

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Vision-Based Position Detection and Robot Path Adaptation for Optimization of Additive Manufacturing Processes

Bernardo de Sá Melo e Castro



Mestrado em Engenharia Mecânica

Supervisor: Prof. Vítor Hugo Pinto

Second Supervisor: Eng. Luís Regueiras

July 21, 2026

Resumo

Os processos de Deposição Direta de Energia (DED) exigem um alinhamento de alta precisão entre a ferramenta robótica de deposição e o substrato, para garantir a máxima utilização do substrato e uma elevada qualidade de impressão. Em muitos contextos industriais, como o que motivou este estudo, esta configuração continua a ser um processo manual e iterativo: os operadores têm de deslocar manualmente o robô, medir os desvios e atualizar as coordenadas no *Flex Pendant*. Esta investigação propõe uma alternativa baseada na visão computacional, selecionada pela sua adaptabilidade superior e pela relação custo-precisão, com o objetivo de obter uma extração de contornos submilimétrica para corrigir as posições de impressão, tanto em cenários de impressão como de reparação.

O cerne deste trabalho reside na conceção e avaliação comparativa de arquiteturas de segmentação baseadas em *Deep Learning* para imagens industriais de alta resolução, utilizadas para extrair contornos precisos de substratos, impressões e grampos de fixação. Foi empregue uma estratégia de divisão em fatias ao longo de todo o processo para preservar os detalhes finos dos contornos, permitindo simultaneamente que os modelos processassem de forma eficaz imagens de 20,2 MP captadas por uma câmara Basler ace 2.

O sistema integra ainda um módulo robusto de calibração e teste, permitindo ao operador gerir de forma intuitiva os parâmetros intrínsecos e extrínsecos e a focagem da lente, incluindo um procedimento de calibração *Hand-Eye* para garantir a extração precisa de coordenadas em toda a área de trabalho do robô. Um analisador RAPID extrai as coordenadas de movimento do percurso da ferramenta da peça, que são depois processadas por um algoritmo de *Nesting* para determinar a correção ideal a aplicar no percurso de impressão. O sistema atualiza o *WorkObject* do robô via TCP/IP assim que a correção computada anteriormente é validada.

Foi também desenvolvido um fluxo de trabalho de reparação como prova de conceito, com o objetivo de alcançar o alinhamento entre a trajetória do percurso de reparação e a região danificada antes do início da deposição, incorporando mapas de profundidade derivados de luz estruturada para superar a semelhança visual entre as superfícies metálicas presentes na imagem.

Palavras-chave: Deposição Direta de Energia • Visão Computacional • Deep Learning • Calibração de Câmara • Algoritmos de Nesting • Luz Estruturada

Abstract

Directed Energy Deposition (DED) processes require high-precision alignment between the robotic deposition tool and the substrate to ensure maximum substrate utilization and high print quality. In many industrial settings, such as the one motivating this study, this setup remains a manual, iterative process: operators must manually jog the robot, measure offsets, and update coordinates in the teach pendant. This research proposes a computer vision-based alternative, selected for its superior adaptability and cost-to-precision ratio, targeting sub-millimeter edge extraction to correct printing positions in both nesting and repair scenarios.

The core of this work lies in the design and comparative evaluation of deep learning segmentation architectures for high-resolution industrial imaging, used to extract accurate contours of substrates, print deposits, and fixturing clamps. A tiling strategy was employed throughout to preserve fine contour detail while allowing the models to process 20.2MP frames from a Basler ace 2 camera effectively.

The system further integrates a robust calibration and testing module, enabling the operator to manage intrinsic and extrinsic parameters and lens focus intuitively, including a hand-eye calibration procedure to ensure accurate coordinate extraction across the robot's working envelope. A RAPID parser extracts movement coordinates from the part's toolpath, which are then processed by a nesting algorithm to determine optimal placement based on the high-precision visual data, and the system updates the robot's WorkObject via TCP/IP once validated.

A proof-of-concept repair pipeline was also developed, with a goal of achieving the complete overlap between the repair toolpath and the damaged region before deposition begins, incorporating depth maps derived from structured light to overcome the visual similarity between metallic substrates and their surroundings.

Keywords: Directed Energy Deposition • Computer Vision • Deep Learning • Camera Calibration • Nesting Algorithms • Structured Light

Acknowledgements

I would firstly like to thank my supervisor, Prof. Dr. Vítor Hugo Martins, for the support given throughout this project. He was one of the most dedicated professors I have had, his genuine interest in this field and his drive for acquiring and spreading knowledge were truly contagious, which was fundamental in igniting my own passion for this area of expertise.

I would also like to express my deepest gratitude to my co-supervisor and colleague, Eng. Luís Regueiras, for always showing availability and a genuine interest in helping me. He not only helped solve my problems and ensured I had everything I needed to progress, but also lifted my spirits and ensured the project was not only interesting but fun to develop. He also pushed me to be as perfectionist as possible and played a big part in ensuring I placed the most effort into every task. I feel lucky to have had him as a supervisor and as a friend.

I extend this appreciation to all my colleagues from my Bachelor's degree at ISEP, from my Master's at FEUP who made this journey so joyful, truly proving it is not only the destination or journey that matters, but mostly the company.

I would also like to thank all my colleagues at INEGI for the companionship and social moments, which helped me relax and disconnect for a while, allowing me to return to work with renewed focus and energy. With a special thanks to my colleague Ricardo Carvalho for the additional guidance and help.

I also feel the need to thank my friends, without whose support I know I could not have achieved all that I have in the last few years. My special thanks go to my girlfriend, Joana, who has been with me almost since the beginning of my adulthood and who has not only been one of my greatest sources of motivation but has made me a better person at all levels of my life. I would also like to thank my friend António, who has been my friend since I can remember and has deeply influenced the person I became. I will forever regard him as one of the most important reasons for the successes I will achieve.

Lastly, I want to thank my family for all the love and support given to me. I truly consider us a perfect family, not in the sense that we have no flaws, but in the sense that there is nothing I would change. Their support is, without a doubt, the primary reason for all that I have achieved and will achieve. A final special thanks to my mother Carla, who remains my major source of inspiration and the person I hope most not to disappoint.

Bernardo Castro

Declaration of honour

I declare that this dissertation is my own work and has not previously been submitted or assessed at this or any other institution. References to other authors (statements, ideas, thoughts), as well as the use of published works, strictly comply with the rules of attribution and are duly identified in the text and in the bibliographic references, in accordance with the applicable referencing standards. I am aware that the practice of plagiarism or self-plagiarism constitutes an academic offence, as established in the U.Porto Code of Ethics.

I also declare that I have used Artificial Intelligence tools to complete part(s) of this dissertation, and that this use and its results are duly noted and have been carefully checked for errors, inaccuracies, unfounded attributions or other forms of bias in content and arguments, as well as determining authorship.

Bernardo de Sá Melo e Castro

Bernardo Castro

*“The important thing is not to stop questioning.
Curiosity has its own reason for existence.”*

— ALBERT EINSTEIN

Contents

1	Introduction	1
1.1	Motivation and Industrial Context	1
1.2	The Proposed Vision-Based Approach	1
1.3	System Features and Document Structure	2
2	Background and related Work	3
2.1	Automated Localization Systems	3
2.2	Camera Models and Spatial Calibration	4
2.3	Deep Learning for Industrial Image Segmentation	5
2.4	3D Scene Reconstruction and Depth Estimation	8
2.5	Multimodal Fusion for Enhanced Segmentation	10
2.6	Nesting Algorithms	12
2.7	Rigid Alignment of 2D Contours	13
3	System Architecture and Methodology	15
3.1	Vision System Selection	15
3.2	Robotic Integration and Setup	16
3.3	Camera calibration	17
3.4	Projector System Calibration	32
3.5	Slicing Aided Hyper Inference (SAHI)	33
3.6	Dataset Preparation	33
3.6.1	Dataset for the Nesting System	34
3.6.2	Dataset Preparation for the Repair System	35
3.7	Segmentation Pipeline Architecture	37
3.7.1	Object Detection for Prompt Generation	37
3.7.2	Object Segmentation for Contour Extraction	39
3.8	Depth Map Generation	46
3.9	RGB-D Segmentation	48
3.10	Nesting Algorithm	55
3.11	Repair Toolpath Alignment	59
3.12	App-Robot Communication	61
4	System Validation and Performance Evaluation	63
4.1	Calibration Pipeline	63
4.2	Nesting System Workflow	68
4.3	Repair System Workflow	72

5	Conclusions and Future Work	76
5.1	Conclusions	76
5.2	Future Work	79
	References	80
A	Nesting Benchmark Detailed Results	83

List of Figures

2.1	Example of COLMAP	9
2.2	Example of photometric stereo setup	10
3.1	Overview of the experimental DED cell featuring an ABB IRB4600 robot. The camera mounted on the end-effector is highlighted with a solid circle, the printing table with a dash-dot circle, and the WLAM system with a dashed circle.	17
3.2	Distribution of positional errors across markers.	22
3.3	Comparison of hand-eye calibration solvers by XY standard deviation. Lower values indicate better cross-pose consistency.	25
3.4	Sample images from the dataset prior to augmentation.	35
3.5	Representative samples from the repair dataset: (a) RGB image, (b) instance annotation, (c) depth map visualization, and (d) annotation detail.	36
3.6	Yolo training results	38
3.7	Yolo predictions confusion matrix	39
3.8	Visual comparison between the default HQ-SAM (a, c) and the fine-tuned model (b, d) on two representative substrate examples.	41
3.9	IoU comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.	42
3.10	DICE coefficient comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.	43
3.11	Boundary IoU comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.	43
3.12	Inference comparison between Mask2Former and HQ-SAM	45
3.13	Structured light acquisition outputs. (a,b) Colourised depth maps of the printing table under two substrate configurations. (c,d) Reconstructed point clouds from a single camera pose and from multi-pose fusion with station rotation, respectively.	48
3.14	Validation mIoU evolution over 80 training epochs for DFormer-Tiny and CMX-B0. Dashed horizontal lines indicate the peak mIoU reached by each architecture.	50
3.15	Qualitative comparison of segmentation predictions produced by CMX-B0 and DFormer-Tiny on two representative samples of the held-out test set. The green overlay corresponds to the predicted <i>PrintingArea</i> mask, the blue overlay to the predicted <i>grampo</i> , and the thin white contour to the ground-truth boundary.	52
3.16	Intersection-over-Union (IoU) for the <i>PrintingArea</i> class achieved by both architectures on the held-out test set.	53

3.17	DICE score for the <i>PrintingArea</i> class achieved by both architectures on the held-out test set.	53
3.18	Boundary F-score with 3-pixel tolerance (BF@3px) for the <i>PrintingArea</i> class achieved by both architectures on the held-out test set.	54
3.19	Example of nesting scenario	56
3.20	Sequential packing test results: percentage of substrate area occupied by each nesting method until no further placements were possible.	57
3.21	Repair toolpath alignment on two simulated cases. The segmented <i>PrintingArea</i> contour (green + red) is shown together with the parsed toolpath (blue), first placed arbitrarily (a, c) and then after applying the computed $(\Delta x, \Delta y, \theta)$ correction (b, d).	61
4.1	Workflow of the Mode 2 calibration (camera intrinsic and extrinsic). Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.	65
4.2	Projector's intrinsic calibration flowchart. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.	66
4.3	Projector's extrinsic calibration flowchart. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.	67
4.4	Overview of the full nesting system workflow. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.	69
4.5	Overview of the full nesting system communication flow	70
4.6	Nesting pipeline interface steps and laser validation	71
4.7	Visual verification of spatial alignment in simulation mode, illustrating the result of the integrated pipeline.	73
4.8	Flowchart of the repair system workflow. Green boxes denote operator-only manual steps, orange boxes represent autonomous system actions, and blue boxes indicate combined human-machine interactions.	74

List of Tables

3.1	Marker error analysis	20
3.2	Summary of marker error categories and system baseline	20
3.3	Statistical analysis of bias and interpretation	21
3.4	Marker corner coordinate data	23
3.5	ArUco marker ID0H23 — corner position measurements and reprojection error .	23
3.6	Comparative statistical analysis of bias: original vs. after improvements	24
3.7	Comparative error analysis: system baseline, original, and improved results	24
3.8	Comparison of pose-set diversity metrics between the original 20-pose set and the revised 12-pose set used for hand-eye calibration.	27
3.9	Hand-eye calibration validation results.	29
3.10	Hand-eye calibration validation from multiple robot starting positions. Same validation markers and methodology as Table 3.9.	29
3.11	Comparative summary of extrinsic calibration methods evaluated. Percentage improvements are relative to the preceding method in the sequence.	30
3.12	Hand-eye calibration validation at 960 mm working distance after intrinsic-only recalibration.	30
3.13	Peak validation metrics achieved by the trained architectures.	49
3.14	Nesting Method Benchmark — Summary Statistics Across All Scenarios	55
A.1	Nesting method benchmark — individual scenario results.	83

List of Acronyms

AM	Additive Manufacturing
AP	Average Precision
ArUco	Augmented Reality University of Cordoba
BCE	Binary Cross Entropy
BF	Boundary F-score
BLF	Bottom-Left Fill
CAD	Computer-Aided Design
COCO	Common Objects in Context
DED	Directed Energy Deposition
DICE	Dice coefficient
DMD	Digital Micromirror Device
FCOS	Fully Convolutional One-Stage Object Detection
GPU	Graphics Processing Unit
GSD	Ground Sampling Distance
HHA	Horizontal disparity, Height above ground, Angle with gravity
HQ-SAM	Segment Anything in High Quality Model
ICP	Iterative Closest Point
IoU	Intersection over Union
IP	Internet Protocol
Mask2Former	Masked-attention Mask Transformer
mIoU	mean Intersection over Union
MP	Megapixel
NFP	No-Fit Polygons
NMM	Non-Maximum Merging
NMS	Non-Maximum Suppression
OEE	Overall Equipment Effectiveness
PQ	Panoptic Quality
RGB	Red, Green, Blue
RGB-D	Red, Green, Blue plus Depth
RMS	Root Mean Square

ROI	Region of Interest
RT-DETR	Real-Time Detection Transformer
SAHI	Slicing Aided Hyper Inference
SAM	Segment Anything Model
SfM	Structure from Motion
TCP	Transmission Control Protocol
TOOD	Task-aligned One-stage Object Detection
UI	User Interface
VFNet	VarifocalNet
VRAM	Video Random Access Memory
WLAM	Wire Laser Additive Manufacturing
YOLO	You Only Look Once

Chapter 1

Introduction

1.1 Motivation and Industrial Context

The paradigm shift toward Industry 4.0 has Additive Manufacturing (AM) as a cornerstone of modern industrial production. Within this field, Directed Energy Deposition (DED) has gained significant prominence due to its capacity for large-scale fabrication and the high-value repair of complex metallic components. However, the operational efficiency of DED remains heavily reliant on manual preparation phases. In current industrial practice, the spatial alignment between the robotic toolpath and the physical substrate is often an iterative, manual process. Operators are required to manually jog the robot and update coordinates, a method that is not only time-consuming but also prone to human error. This directly reduces the profitability of the process, especially in laboratory and industrial settings where setup time represents a significant fraction of total machine time, directly impacting process efficiency and throughput.

1.2 The Proposed Vision-Based Approach

To address these limitations, this research proposes a computer vision system designed to automate the nesting of new prints and to show how depth maps could be used to extend the system's features to help in the alignment of repair jobs. Unlike traditional contact probes or expensive laser profilometers, a vision-based alternative offers a superior balance between adaptability and cost-to-precision ratio. The developed solution employs a high-resolution imaging pipeline using a 20.2 Megapixel (MP) Basler ace 2 industrial camera. By implementing state-of-the-art deep learning segmentation models and a Slicing Aided Hyper Inference (SAHI) algorithm, the system achieves sub-millimeter edge extraction. This provides the foundational data for two distinct workflows: a robust Nesting System aimed at maximizing substrate utilization through rapid spatial optimization, and a Proof-of-Concept (PoC) repair system. Developed under current hardware limitations, the latter incorporates depth map reconstruction and Red, Green, Blue plus Depth (RGB-D) foundation models to generate high-fidelity surface contours, ensuring the precise volumetric alignment required for additive repair in a laboratory setting. Ultimately, this setup serves to evaluate

whether the future procurement of specialized hardware is justified before committing to a significant financial investment.

1.3 System Features and Document Structure

The system's architecture is defined by the integration of advanced computer vision with robotic path planning. A key feature of the pipeline is its robust calibration module, which uses intrinsic parameters to correct lens distortion and map pixel coordinates into the camera frame, and extrinsic parameters to transform these into the robotic coordinate frame. This data is utilized by a nesting algorithm that processes the toolpath coordinates to determine the optimal arrangement of prints, or, in repair scenarios, to verify the complete overlap between the repair toolpath and the damaged component printing surface. By establishing a Transmission Control Protocol (TCP)/Internet Protocol (IP) communication bridge, the system updates the robot's work object automatically, effectively bridging the gap between digital planning and physical execution.

This dissertation is organized as follows: Chapter 2 explores previous works in industrial segmentation and nesting; Chapter 3 details the implementation of the deep learning pipeline, SAHI architecture, and hardware integration; Chapter 4 evaluates the performance and accuracy of both the nesting and repair workflows; and Chapter 5 presents the final conclusions and future work.

Chapter 2

Background and related Work

The precision and reliability of the Directed Energy Deposition (**DED**) process is directly influenced by the accurate synchronization of the robotic manipulator with the printing substrate and other surroundings. Achieving a high-quality deposition requires an accurate definition of the work object, ensuring that the programmed toolpath aligns perfectly with the physical geometry where the deposition should be performed. This section reviews the present day localization methods, contrasting traditional manual techniques with automated contact and non-contact sensing technologies, as well as other relevant approaches.

As introduced in Chapter 1, the manual alignment of the toolpath to the substrate is a major source of non-productive time. This penalty is compounded in complex geometries and repair tasks, where the operator must repeat the jogging and offset-verification cycle multiple times, degrading the Overall Equipment Effectiveness (**OEE**) and increasing the likelihood of misalignment defects.

2.1 Automated Localization Systems

Contact Based Localization Systems

To mitigate human error, several automated solutions have been integrated into Additive Manufacturing (**AM**) systems. One of them is touch-trigger probes, these probes allow the robot to autonomously detect the substrate's surface through physical contact. Bastas [1] showed that touch-trigger probes allow autonomous surface localization with considerably high accuracy. However, contact sensing presents inherent drawbacks: the process is relatively slow due to the low speed necessary for safe probing, and the sensors can suffer mechanical wear. Furthermore, contact probes lack the capacity to interpret the semantic context of the workspace, failing to identify previously printed features or other structures that continue to obstruct the active build volume. These systems are also ill-suited for evaluating complex geometries, frequently encountered in repair operations as resolving such shapes requires an impractical density of physical measurement points, hard to match with the theoretical Computer-Aided Design (**CAD**) ones.

Laser Profilometry and 3D Scanning Technologies

Non-contact sensing, particularly laser line scanners and profilometers, represents the high-end spectrum of localization technology. These sensors project laser patterns onto the substrate to generate high-density 3D point clouds. Liu et al. [2] showed that while laser-based systems offer exceptional geometric accuracy, their implementation in DED environments often implicates very high costs. The metallic and often reflective nature of DED substrates can cause significant noise in laser data, requiring advanced filtering algorithms. This creates a technological gap where there is a need for a solution that offers the flexibility and high precision of 3D scanning, but at a lower cost, a gap that computer vision is uniquely positioned to fill.

2.2 Camera Models and Spatial Calibration

The implementation of an industrial vision system for robotic position correction requires a robust mathematical mapping between the two-dimensional image plane and the three-dimensional robotic workspace. This is fundamentally achieved through the pinhole camera model, which describes the projection of 3D points into a 2D plane using a set of intrinsic and extrinsic parameters. Geometric calibration, such as the method proposed by Zhang [3], remains the best approach in industrial vision, utilizing planar patterns to estimate these parameters while simultaneously compensating for radial and tangential lens distortions. The intrinsic parameters, typically represented by a calibration matrix K , account for the camera's focal length (f_x and f_y), principal point (c_x and c_y) and skew coefficient (γ):

$$K = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

Accurate determination of this matrix is essential to compute the necessary scaling factor to translate pixel data into metric units. To accurately model the transformation between 3D world coordinates and the 2D image plane, a calibration process is required. Zhang's method recommends capturing images of a planar chessboard pattern in various positions and orientations to compute the distortion vector and the intrinsic and extrinsic matrices. Lenses often deviate from the ideal pinhole model, resulting in geometric distortion caused by the curvature of the lens elements, leading to radial and tangential effects. The calibration process uses the detected corners of the chessboard to calculate a set of distortion coefficients (k_1, k_2, k_3, p_1, p_2), where k_1, k_2 and k_3 are the radial distortion coefficients, which model the barrel or pincushion distortion introduced by the lens curvature, and p_1, p_2 are the tangential distortion coefficients, which account for the misalignment between the lens and the image sensor. These coefficients form the distortion vector used to undistort the image, a critical step for high-precision position detection.

Furthermore, the extrinsic matrix describes the camera's location and orientation relative to the world, consisting of a rotation matrix R and translation vector t . Together, these parameters form the extrinsic matrix $[R|t]$, which maps a point from the 3D world coordinate system to the 3D camera coordinate system:

$$[R|t] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \quad (2.2)$$

where R is a 3×3 rotation matrix whose columns represent the orientation of the world axes relative to the camera frame, and $t = [t_x, t_y, t_z]^T$ is the translation vector representing the position of the world origin in the camera coordinate system. The computation of this matrix is vital for eye-in-hand applications, as the spatial relationship between the camera mounted on the end effector and the robot base must be calculated to enable the autonomous update of the work object. Maintaining this calibration is essential in industrial environments where vibrations or thermal expansion may affect the camera's focus and orientation.

2.3 Deep Learning for Industrial Image Segmentation

The transition from raw high-resolution pixel data to actionable robotic coordinates requires a segmentation pipeline designed to maintain spatial integrity while providing operational flexibility. By analysing literature with similar objectives, it is evident that traditional computer vision methodologies, such as simple edge detection or thresholding, are often ineffective in DED environments. As highlighted by "A Survey of Vision-Based Methods for Surface Defects' Detection and Classification in Steel Products" [4], such approaches struggle to generalise across varying surface conditions, lighting conditions, and material textures. The presence of significant visual noise, low contrast between metallic substrates and the metallic work table, and the presence of modular holes in the table surface tend to confuse contour extraction functions. While fine-tuning thresholds might offer a temporary solution, such systems become overly rigid and fail to adapt to variations. Consequently, deep learning-based segmentation has emerged as a promising direction for this class of problems, as such methods offer markedly improved robustness to visual variation and finer spatial accuracy than traditional approaches as shown in Ulku and Akagündüz [5].

Segmentation Models

Correctly segmenting the substrates, prints and clamps is the core task of this pipeline, and it requires separating individual object boundaries at the pixel level. There are three types of segmentation. Semantic segmentation labels each pixel with a broad category, but cannot tell apart different objects of the same class. Instance segmentation goes further and gives every detected object a unique identity, which is essential for distinguishing between the several clamps and prints that may appear in the same frame. Panoptic segmentation combines the two, producing a single,

complete interpretation of the scene. For a nesting pipeline, instance or panoptic segmentation are the suitable choices: they ensure that each physical part is recognized as a separate geometry, preventing adjacent objects from merging, a failure that would otherwise compromise the spatial optimization and collision-avoidance logic of the nesting system. Beyond this, adopting a panoptic or instance segmentation model opens the door to additional features that rely on assigning a unique identifier to each segmented body.

Among the most influential recent contributions in this area is the Segment Anything Model (SAM), introduced by Kirillov et al. [6], which established a new paradigm in promptable segmentation. SAM was trained on a dataset of over one billion masks and demonstrated impressive zero-shot generalization across a wide variety of image distributions. However, Ke et al. [7] identified two fundamental limitations in SAM’s outputs: coarse mask boundaries and frequent errors when dealing with objects presenting intricate or thin structures. To address these shortcomings, the same authors proposed the Segment Anything in High Quality Model (HQ-SAM), which introduces a learnable High-Quality Output Token injected into SAM’s mask decoder. This token, fused with both early and final Vision Transformer (ViT) encoder features, is trained on a curated dataset of 44K fine-grained masks, enabling the model to preserve SAM’s zero-shot capabilities while substantially improving boundary accuracy.

A key limitation of prompt-dependent models such as SAM and HQ-SAM is that they require an external input: either a point, a set of points, or a bounding box, to identify the region of interest for segmentation. This dependence on manual prompting renders them unsuitable for fully autonomous inference pipelines. To overcome this, a lightweight object detection neural network can be employed upstream in the pipeline to automatically generate the bounding box proposals that serve as prompts for the segmentation model.

The You Only Look Once (YOLO) family of models represents one of the most established and widely adopted approaches to real-time object detection. One of the latest major iterations, YOLOv8, introduced by Jocher, Chaurasia, and Qiu [8], builds upon prior versions with an anchor-free split detection head and enhanced backbone and neck architectures, achieving an optimized trade-off between inference speed and accuracy. YOLOv8 is further distinguished by its extensive documentation, straightforward training interface, and demonstrated capacity to generalize effectively even when trained on small custom datasets. These characteristics make it particularly well-suited for industrial applications with limited annotated data.

Other object detection architectures were considered as alternatives. Real-Time Detection Transformer (RT-DETR), offers competitive accuracy through attention-based feature aggregation and eliminates the need for the Non-Maximum Suppression (NMS) post-processing step required by

YOLO-based models, potentially improving inference determinism. Similarly, YOLOv9, as explained in Wang, Yeh, and Liao [9], introduces programmable gradient information and a generalized efficient layer aggregation network architecture, achieving improved accuracy at comparable speeds. Despite these advances, YOLOv8 remains notable for its maturity, ecosystem support, and lower barrier to custom training.

An alternative approach to instance segmentation is offered by the Masked-attention Mask Transformer (**Mask2Former**), proposed by Cheng et al. [10]. Mask2Former introduces a universal architecture capable of addressing panoptic, instance, and semantic segmentation tasks within a single framework, setting state-of-the-art results on the Common Objects in Context (**COCO**) benchmark. Its key innovation is a masked attention mechanism that constrains cross-attention within predicted mask regions, leading to faster convergence and improved performance. Unlike SAM and HQ-SAM, Mask2Former is not a promptable model which can be an advantage when considering it can work independently, but it also makes it impossible to have a semi automatic system where the user defines the interest area. While it can be fine-tuned on custom datasets, achieving effective domain adaptation, typically requires updating a substantially larger number of parameters compared to the ultra-lightweight decoder fine-tuning strategy available in HQ-SAM. Consequently, adapting Mask2Former is more computationally demanding and time-consuming. Furthermore, this dense parameter update significantly increases the risk of overfitting, particularly when training on specialized, low-volume datasets. Additionally, the absence of zero-shot capabilities means that any new object geometry encountered at inference time, such as the highly variable cross-sections of components arriving for repair should have been represented in the training set, which could create a challenge in industrial repair scenarios where the variety of damaged part geometries is practically unlimited.

High-Resolution Inference and Image Slicing

With the goal of detecting sub millimetric features, this work will likely require the use of a high-resolution camera. This introduces a significant computational challenge: modern neural network detectors are typically trained and optimized for input images of moderate resolution (e.g., 1024×1024 pixels), and directly downsampling high-resolution imagery to these dimensions results in a proportional loss of spatial detail, which can cause small or fine-features to become entirely undetectable.

A principled solution to this problem is the use of image slicing strategies, which partition high-resolution images into overlapping sub-regions, run inference on each sub-region at native resolution, and subsequently merge the resulting predictions back into the original image space. The Slicing Aided Hyper Inference (**SAHI**) framework, proposed by Akyon, Altinuc, and Temizel [11], formalizes this approach into an open-source, model-agnostic pipeline. SAHI demonstrated consistent and significant improvements in the Average Precision (**AP**) metric across multiple detectors and benchmark datasets: inference-time slicing alone increased AP by 6.8%, 5.1%, and

5.3% for the Fully Convolutional One-Stage Object Detection (**FCOS**), VarifocalNet (**VFNet**), and Task-aligned One-stage Object Detection (**TOOD**) detectors respectively on the VisDrone and xView datasets.

An alternative slicing strategy is the uniform sliding window approach studied by Yang et al. [12] in the context of aerial imagery. That work highlights both the effectiveness of tiling for enlarging small objects and the risk of boundary truncation when objects span multiple slices. The authors propose a mixed data strategy that simultaneously processes the original image at global scale and the sub-patches at local scale, partially mitigating truncation artefacts. SAHI addresses this concern through configurable overlap ratios between slices and a post-processing merging step based on Non-Maximum Merging (**NMM**), which reconstructs detections across slice boundaries without discarding information. The availability of mature tooling, direct integration with the YOLO model family, and the demonstrated quantitative performance advantages position SAHI favourably among available slicing strategies.

Concluding Remarks

Based on the literature reviewed in this chapter, a multi-stage inference pipeline consisting of SAHI-based image slicing, followed by bounding box detection, followed by high-quality instance segmentation, and concluded by SAHI-based result reconstruction, appears to be the most promising architecture for the problem at hand. Additionally, the Mask2Former architecture will be trained and integrated into an analogous pipeline to serve as a comparative benchmark, allowing for a rigorous evaluation of which model adapts more effectively to the specific geometric and environmental constraints of the DED workspace.

The HQ-SAM model will be addressed first in the experimental phase, serving as a baseline for the development of the vision pipeline. This prioritization is motivated by the model’s capacity for immediate deployment, providing an initial ground truth against which subsequent iterations can be benchmarked. Furthermore, the lightweight nature of its adaptation strategy is particularly advantageous for environments with limited computational resources, as the accelerated training cycle facilitates the parallel testing of auxiliary pipeline features. The viability of this approach is reinforced by the findings of Gandomi et al. [13] and Wu et al. [14], which demonstrate that targeted fine-tuning on relatively small, domain-specific datasets can yield highly accurate segmentation results.

2.4 3D Scene Reconstruction and Depth Estimation

A primary objective of integrating depth sensing into the vision pipeline is to resolve ambiguities that 2D segmentation models struggle to handle. In many DED environments, the metallic substrate and the work table exhibit nearly identical textures, colors, and reflective properties, leading to semantic confusion where the model fails to distinguish the object from the background. By

incorporating a third dimension, the system can leverage height discontinuities to refine the masks generated by the neural networks, ensuring that the detected boundaries correspond to physical elevations rather than visual artifacts. This section evaluates various techniques for obtaining depth maps using 2D camera sensors, weighing their industrial feasibility against the specific constraints of the robotic DED cell.

One widely documented approach for 3D reconstruction is Structure from Motion (**SfM**), frequently implemented via the COLMAP framework. COLMAP utilizes a series of overlapping 2D images taken from different viewpoints to estimate both the camera's spatial trajectory and the 3D structure of the scene through feature matching and bundle adjustment. Figure 2.1 illustrates the results of this framework. While this method can produce highly detailed point clouds, it was discarded for this application due to its high setup time and motion requirements. Effective reconstruction requires a vast number of images captured from diverse angle, however, the robotic manipulator used in this work, specifically within the constraints of the Wire Laser Additive Manufacturing (WLAM) system, has a restricted range of motion due to cable management and the bulk of the end-effector. Performing the necessary sweeping movements for COLMAP would significantly increase non-productive cycle time, contradicting the goal of rapid localization. As shown in Figure 2.1 the number needed for precise reconstruction can exceed 100 images.



Figure 2.1: Example of COLMAP

Photometric Stereo was also considered as a potential solution for surface normal estimation and depth mapping. This technique relies on capturing multiple images from a fixed camera position while varying the direction of the light source, as shown in Figure 2.2. By analyzing the shading variations on the object's surface, the local surface orientation can be mathematically derived. Despite its precision in capturing fine surface details, this method was discarded for practical and economic reasons. Implementing Photometric Stereo in an industrial DED cell would require a specialized lighting rig with multiple, precisely controlled, light sources mounted around the workspace. This would not only increase the hardware cost and system complexity but also introduce challenges regarding the high reflectivity of metallic substrates, which produces specular highlights that violate the Lambertian assumption required by standard Photometric Stereo algorithms and degrade the accuracy of the reconstructed normals Moske et al. [15].

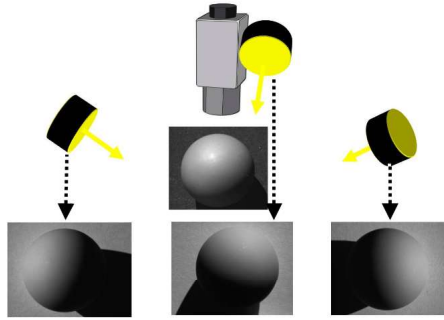


Figure 2.2: Example of photometric stereo setup

With the recent advancements in artificial intelligence, "Monocular Depth Estimation" models, such as Depth Anything, have emerged as a software-only alternative for generating depth maps from a single 2D image. These neural networks are trained on massive datasets to infer depth based on contextual cues. However, using these models would likely result in a depth map with the same level of reliability and precision as the 2D segmentation models themselves. Since the goal is to use depth data as an independent "ground truth" to overcome the texture similarities between the table and the substrate, relying on another predictive model that may suffer from the same visual biases, would not be adequate for high-precision industrial localization.

Among the reviewed technologies, 3D Structured Light represents the most viable solution for this work. This technique involves projecting a known light pattern (such as a grid or a series of stripes) onto the scene and capturing the deformation of that pattern with a camera. The geometric displacement of the pattern allows for the triangulation of depth at every pixel with high accuracy. Structured light was selected as the methodology for this research due to its relatively low hardware requirements at the proof-of-concept stage, needing only a standard projector and camera, coupled with its fast execution time and the well-established calibration literature. Furthermore, its ability to provide an objective, metric depth map independently of surface texture makes it the ideal candidate to assist the segmentation pipeline in distinguishing between the printing substrate and the work table. It should be noted, however, that these modest requirements apply only to the validation of the concept. An industrial-grade implementation, with the accuracy and robustness demanded by production, would still entail a considerably higher hardware investment.

2.5 Multimodal Fusion for Enhanced Segmentation

Beyond the direct use of depth maps for geometric measurement, integrating depth data into the image segmentation pipeline provides a significant advantage in resolving visual ambiguities. When the reflectiveness and texture of the substrate and the table is nearly identical, the depth channel acts as an additional feature space that provides the necessary geometric separation for accurate delineation. This section explores various methodologies for fusing Red, Green, Blue

(RGB) and depth (RGB-D) data, ranging from traditional geometric filtering to state-of-the-art deep learning architectures.

Before employing complex neural networks, a more direct way to use depth data is through geometric point cloud processing. Once the depth map is converted into a 3D point cloud using the camera’s intrinsic parameters, algorithms such as Alpha Shapes, introduced by Edelsbrunner, Kirkpatrick, and Seidel [16], can be used to extract the substrate boundary. An Alpha Shape is a mathematical definition of the shape of a set of points, controlled by a parameter α that sets how much detail the resulting boundary retains. By keeping only the points within a given height range above the table and computing their Alpha Shape, the system produces a contour that does not depend on visual texture or lighting, which can serve as a fallback or as a refinement mask for the 2D segmentation results. This contour, however, is only as accurate as the depth map behind it: the approach is only worth using when the depth sensor captures the surface at a density close to that of the vision system. In the proof-of-concept setup used in this work, the available depth resolution would produce a coarser mask that might miss finer or more complex contours, which is why this option was left as a documented direction rather than tested experimentally.

Before depth data is combined with RGB in a deep learning pipeline, it can be encoded in a form that neural networks can better exploit. A common approach in early RGB-D segmentation pipelines is the Horizontal disparity, Height above ground, Angle with gravity (HHA) encoding Gupta et al. [17]. By transforming the raw depth map into these three geometrically meaningful channels, HHA exposes scene structure in a form readily consumed by convolutional neural networks (CNNs) originally designed to process three-channel RGB inputs, allowing pretrained backbones to be repurposed for depth data. However, this encoding carries a non-negligible per-frame computational cost, relies on assumptions such as a known gravity direction and ground plane, and amplifies sensor noise through the surface-normal estimation required for the angle channel.

Once a suitable depth representation is chosen, the next design decision concerns how the RGB and depth streams are actually merged. In the realm of deep learning, this strategy is categorized by the stage at which the two modalities are combined. Early Fusion merges the RGB and depth channels at the input level, which is computationally efficient but often fails to capture the complex, non-linear correlations between appearance and geometry. Mid-level Fusion integrates features at various stages within the encoder backbones, allowing the model to learn cross-modal dependencies at different scales of abstraction. Late Fusion processes each modality through separate networks and merges their final predictions, which, while robust, often lacks the fine-grained spatial alignment required for precise DED applications.

Building on these fusion strategies, several state-of-the-art RGB-D segmentation models were considered for this work. CMX employs a dual-stream architecture with a Cross-Modal Feature Rectification module to handle misalignments and noise in the depth map Zhang et al. [18]. DFormer, a transformer-based approach, prioritizes depth features by using a powerful encoder to extract high-level geometric representations Yin et al. [19]. HBDFormer utilizes a Hybrid Binocular Discriminative Transformer to enhance feature diversity between the two modalities Wei et al. [20]. AsymFormer, on the other hand, adopts an asymmetric architecture where the depth branch is lighter than the RGB branch, optimizing for real-time performance without sacrificing accuracy Du et al. [21].

Based on the literature and the specific requirements of the DED environment, two distinct architectures have been selected for the experimental phase: DFormer utilizing a DFormer-Tiny backbone, and CMX configured with a lightweight backbone variant. The specific particularities of the custom DED dataset present unique challenges, as its industrial features and geometric characteristics diverge significantly from standard public benchmark datasets. This domain mismatch implies that conventional fine-tuning of large, general-purpose models is not ideal, as standard pre-trained weights fail to adequately capture the unique textures and geometries of metallic deposition tracks. However, because the custom dataset is relatively small, employing high-capacity, heavy architectures would inevitably lead to severe overfitting during training. Consequently, compact versions, specifically the DFormer-Tiny variant and a lightweight CMX configuration were selected to balance architectural expressiveness with strong generalization capabilities under data-constrained conditions.

2.6 Nesting Algorithms

As one of the main goals of this project is to help the operator in the nesting of new prints, once the substrate, previous prints and clamps are segmented and located, there is still the need to apply an algorithm that correctly shifts the new print position to an ideal one, minimizing substrate use and avoiding collisions with previous prints, or the clamps. This chapter focuses on the study of available algorithms and methods to solve the nesting problem.

Two broad families of approaches have been studied in the literature: geometry-based methods and raster-based methods. Geometry-based methods rely on exact representations of part contours and employ mathematical constructs to evaluate feasibility and proximity of placements. Among these, the No-Fit Polygons (NFP) is one of the most widely adopted tools, as it encodes all valid non-overlapping positions of one part relative to another as a single polygon, enabling fast collision and containment checking, without exhaustive point-by-point evaluation. Bennell and Song [22] proposed a comprehensive and robust procedure for computing the NFP of arbitrary polygonal shapes using Minkowski sums, a geometric operation that effectively sweeps the area of one polygon around the boundary of another to define the limit of feasible placements. This

approach specifically addresses numerical robustness issues, such as the handling of degenerate cases, touching edges, and complex non-convex geometries. For this work, such robustness is critical because the contours generated by the vision system often contain high-frequency noise or irregular concavities, so utilizing an NFP method that can reliably process these imperfections is essential to ensuring that the nesting logic does not fail when faced with real-world, segmented imagery.

Raster-based methods, by contrast, represent both the sheet and the parts as binary pixel grids, using morphological image operations to determine valid placements. Bouganis and Shanahan [23] introduced a vision-based intelligent system that exploits this pixel representation alongside computer vision techniques to solve the online irregular packing problem, achieving high packing densities with low computational times per placement decision, making it particularly well-suited to industrial applications where parts arrive sequentially and decisions must be made rapidly. However, the raster approach introduces a dependency on image resolution: finer pixel grids improve placement accuracy at the cost of increased memory and computation, while coarser grids can introduce placement errors that are unacceptable in precision manufacturing contexts.

As the DED process requires the gaps between prints to be in the order of, at least, 2 mm and the distance from the nozzle to the clamp cannot be smaller than 23 mm to avoid contact, the use of geometry based methods, such as NFP, whose speed and required computational power don't scale so linearly with the increase of resolution, seems to be the most promising solution.

2.7 Rigid Alignment of 2D Contours

Once a part has been placed on the printing table, the toolpath generated offline has to be brought into the same frame as the surface actually observed by the camera. This is a two-dimensional instance of *rigid point set registration*: given a source and a target contour known to describe the same physical region, recover the rigid transform $(\Delta x, \Delta y, \theta)$ that maps one onto the other.

The most widely used technique for this task is the Iterative Closest Point (ICP) algorithm, originally introduced by Besl and McKay [24] and refined through a large number of variants surveyed by Rusinkiewicz and Levoy [25]. ICP alternates between assigning each source point to its nearest target point and solving in closed form for the rigid transform that minimises the sum of squared distances over those pairs. The method converges monotonically to a local minimum but is well known to be sensitive to the initial pose: when the starting guess is far from the true alignment, ICP can converge to an incorrect local optimum, and most modern implementations therefore rely on a coarse initialization step that aligns the centroids and the principal axes of the two point sets before any iteration begins Kolpakov and Werman [26]. The same coarse-alignment step can itself be used as a stand-alone solver whenever the conditions justify it, exploiting global shape statistics such as image moments, which provide sampling-independent estimates of the centre of mass Flusser, Suk, and Zitová [27], and the minimum-area bounding rectangle, computable in

linear time on the convex hull Toussaint [28], which provides a robust estimate of orientation that is insensitive to how the boundary was sampled.

For the repair workflow considered in this dissertation, the source and target contours describe the same physical region and differ only by the small rigid offset introduced when the part is placed manually on the table. There is no occlusion, no partial overlap, and no scale change. Under these conditions, the assumptions underlying the centroid-and-rectangle scheme are exactly met, and the most common failure modes of ICP (sensitivity to initialization, local minima from poor initial overlap) bring no advantage in return for their added complexity. The lightweight scheme adopted in Section 3.11 is therefore a deliberate match between the structure of the problem and the simplest method that solves it.

Chapter 3

System Architecture and Methodology

Following the literature review presented in the previous chapter, it is evident that while traditional localization methods like tactile probing or laser scanning are precise, they lack the flexibility and cost-effectiveness required for dynamic DED environments. To address these limitations, this research proposes a high-resolution computer vision system integrated with state-of-the-art deep learning models.

This chapter details the technical implementation of the proposed system, starting from the hardware selection to the development of a robust segmentation and communication pipeline.

3.1 Vision System Selection

The selected camera was the Basler ace 2 (model: a2A4504-5gcIP67) paired with a lens with a 25 mm focal length (model: C12-2524-25M). The selection of this model was based on the abundant number of articles where a Basler ace 2 is used for similar industrial applications such as Merz et al. [29] or Remani et al. [30]. The choice of the model itself was motivated by two primary aspects: the decision to select an RGB sensor to keep the system versatile for future applications, and the necessity to detect surface features with a size of approximately 0.2 mm. This level of detail is crucial for identifying surface textures, small cuts, or groove marks that allow the system to distinguish the substrate from the background table. Consequently, this target feature size also establishes the baseline for the minimum achievable positioning error, the lower the spatial resolution, the higher the potential accuracy of the system.

According to the Nyquist-Shannon sampling theorem, the spatial resolution must be fine enough to provide at least two pixels across the smallest feature to be detected. The spatial resolution (R) is defined by the ratio between the field of view (FOV) and the optical format of the sensor in pixels (O_p):

$$R = \frac{FOV}{O_p} \quad (3.1)$$

For a FOV of 500×500 mm and a 20.2MP sensor (4504×4504 pixels), the resulting spatial resolution is approximately 0.11 mm/pixel. This means that the smallest detectable feature is expected to be in the order of 0.22 mm, establishing the theoretical detection limit of the system under this specific hardware configuration.

This detection limit was selected considering the cumulative errors expected from the vision pipeline, namely those arising from camera intrinsic and extrinsic calibration, as well as the precision of the segmentation masks produced by the neural network. Accounting for these error sources, the nesting system is expected to have an average positioning error below 1 mm and a maximum error below 3 mm. For the repair jobs system, where the increased available setup time allows for a more complex pipeline including camera movement and a FOV reduction, the goal is to achieve an average positioning error below 0.3 mm and a maximum error below 0.6 mm. This maximum allowable error of 0.6 mm corresponds to exactly half of the deposition width (1.2 mm). Restricting the maximum deviation to half of the deposition width ensures that the center of the deposition tool never strays outside the nominal path boundary, guaranteeing a robust weld bead overlap and preventing lack-of-fusion defects along the substrate interface.

3.2 Robotic Integration and Setup

The camera was integrated into the robot's end-effector in an eye-in-hand configuration, as shown in Figure 3.1. This configuration was selected to provide high flexibility in sensor positioning while maintaining mechanical rigidity, thereby ensuring high repeatability. This rigidity is crucial for the extrinsic calibration matrix calculated in future chapters to remain valid after robot movement. By mounting the camera directly on the manipulator, the system can be positioned optimally for each specific substrate or part, ensuring that the high-resolution acquisition required for the segmentation pipeline is consistent.

After correctly fixed, the focal lens and aperture were adjusted to ensure the best picture quality using Basler's software, Pylon viewer, that has a sharp contour feature designed specifically to this end.



Figure 3.1: Overview of the experimental DED cell featuring an ABB IRB4600 robot. The camera mounted on the end-effector is highlighted with a solid circle, the printing table with a dash-dot circle, and the WLAM system with a dashed circle.

3.3 Camera calibration

To ensure the images captured by the camera truly reflect the environment and to allow for the transformation from the pixel coordinate system to the real world robot coordinate system, both the intrinsic and extrinsic calibrations were performed.

Intrinsic Calibration

To calculate the intrinsic matrix and distortion vector Zhang's method was followed. A minimum of 10 images containing a ChArUco board, a board combining Augmented Reality University of Cordoba (ArUco) markers and square chessboard-style markers, were captured from multiple angles and distances to ensure adequate coverage of the camera's field of view, maximising the accuracy of the calibration. The ChArUco board used comprises 20×15 squares with a square side length of 8 mm and ArUco markers of 6 mm drawn from the `DICT_5X5_1000` dictionary. Corner detection and sub-pixel refinement were performed using OpenCV's `CharucoDetector` with `CORNER_REFINE_SUBPIX`, and only images yielding at least 20 detected corners were retained. This refinement step is critical at high resolutions, as the Basler camera operates at 4504×4504 pixels, where even sub-pixel corner localization errors can translate to meaningful reprojection inaccuracies.

The detected 2D image coordinates were paired with their corresponding known 3D coordinates defined on the ChArUco board via `CharucoBoard.matchImagePoints`, and used as input to the `calibrateCamera` function to estimate the camera matrix $\mathbf{K}$ and distortion coefficients $\mathbf{d}$. No additional calibration flags were applied, allowing OpenCV to estimate the full set of radial distortion coefficients. The resulting calibration achieved a Root Mean Square (RMS) reprojection error of 0.505 pixels, indicating a good fit between the projected and detected corner positions. The calibration provides estimates of the focal lengths f_x and f_y , the principal point (c_x, c_y) , and the radial distortion coefficients k_1, k_2, k_3, k_4 , and k_5 . These allow lens distortion correction of any captured image prior to further processing. An example of the resulting intrinsic matrix $\mathbf{K}$ and distortion coefficient vector $\mathbf{d}$ are shown in Equations 3.2 and 3.3.

$$\mathbf{K} = \begin{bmatrix} 9768.73 & 0 & 2228.37 \\ 0 & 9773.23 & 2155.19 \\ 0 & 0 & 1 \end{bmatrix} \text{ px} \quad (3.2)$$

$$\mathbf{D} = \begin{bmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{bmatrix} = \begin{bmatrix} -0.0354 & 0.3037 & 0 & 0 & -0.2080 \end{bmatrix} \quad (3.3)$$

Extrinsic Calibration

Once the intrinsic calibration is computed and the internal parameters of the camera are known, the parameters that relate the camera with the 3D world need to be calculated. To calculate these parameters, there are two primary approaches: the first involves capturing a series of images of a calibration pattern, such as a ChArUco board, from multiple known robot positions to solve the hand-eye calibration problem; the second, relies on a direct correspondence between known physical coordinates on the workspace and their observed image projections.

While the hand-eye calibration method generally offers higher accuracy and the potential for full automation, it necessitates manoeuvring the robot through a series of complex positions. In this setup, such movements are constrained by the physical dimensions and cabling of the Meltio system integrated into the robot's end-effector. Furthermore, this approach requires an industrial-grade calibration target, as even minor surface deformations can introduce significant geometric errors. Consequently, the direct correspondence method was prioritized and evaluated first.

Direct Correspondence Extrinsic Calibration

To implement this method, 3 ArUco markers were placed on the printing table and their position was measured by jogging the robot to the 4 corners of mentioned markers. To ensure the robot was in the correct position the laser pointer, already integrated in the Wire laser Additive Manufacturing Wire Laser Additive Manufacturing (WLAM) system was used. To avoid the error caused by incorrect measuring, 3 measurements were performed for each corner, with robot movement in between each measurement. With the corner positions of the calibration markers measured in the robot coordinate frame, the extrinsic parameters can be computed. The extrinsic

calibration solves for the rotation vector $\mathbf{r}$ and translation vector $\mathbf{t}$ that together describe the rigid transformation from the world coordinate frame, defined by the robot, to the camera coordinate frame. This is achieved using the Perspective-n-Point (PnP) algorithm, specifically the iterative variant implemented in OpenCV's `solvePnP` function with the `SOLVEPNP_ITERATIVE` flag.

For each calibration marker, the four corner coordinates measured by the robot provide a set of 2D–3D point correspondences: the 3D positions are the robot's measured coordinates expressed in millimetres, and the 2D positions are the corresponding pixel coordinates detected in the undistorted camera image. Prior to corner detection, the captured frame is corrected for lens distortion using the intrinsic matrix $\mathbf{K}$ and the distortion coefficients $\mathbf{D}$ obtained in the previous step. ArUco corner detection is then performed on the undistorted image using OpenCV's `ArucoDetector` with the `DICT_4X4_50` dictionary, and the resulting pixel coordinates are passed directly to the solver.

The solver receives the undistorted pixel coordinates, the corresponding 3D world points at the known marker height, and the refined camera matrix $\mathbf{K}'$, yielding the rotation vector $\mathbf{r}$ and translation vector $\mathbf{t}$ that defines the camera pose relative to the robot world frame. These are saved alongside z_{calib} for use during inference. An example of the resulting rotation and translation vectors are presented below.

$$\mathbf{r} = \begin{bmatrix} 2.168 \\ 2.234 \\ 0.022 \end{bmatrix} \text{ rad}, \quad \mathbf{t} = \begin{bmatrix} 0.055 \\ -1.640 \\ 1.024 \end{bmatrix} \text{ m}$$

Direct Correspondence Extrinsic Calibration Validation

In order to validate the camera calibration performed previously the following test was performed: 3 new ArUco markers were placed on the printing table, 2 at the same height as the calibration markers (named ID4 and ID5) and one at substrate level (ID0H23) which also allows to verify the height is being correctly compensated by the software. Their positions were measured following the same methodology as in the previously mentioned chapter, measuring each corner 3 times with robot movement in between measurements. To create a system baseline an A4 paper sheet containing 4 ArUco markers was printed. In this sheet all relative positions are known, which allows to test the calibration pipeline effectiveness removing the influence of errors caused by incorrect measurements. Some error sources still remain: marker bending as the paper sheet is not perfectly flat (this error is increased in the calibration markers as they were attached to the printing table using double faced tape), camera resolution limits and light reflections which affect the corner detection capabilities. With this data the overall error of the system can be evaluated and compared to a baseline. It is also possible to evaluate the effectiveness in height compensation, which is crucial, since the substrates can have a wide range of thicknesses. Finally, bias was

also addressed to verify if the system is consistently either under or overestimating the marker's position.

The first statistical test performed was on the measurements done to extract the markers position. A standard deviation of 0.08 mm on X coordinates and 0,109 mm on Y coordinates was recorded. Max delta between measurements on X coordinates and on Y coordinates, were also addressed, returning the values 0.217 mm and 0.193 mm, respectively, showing the manual measuring process was stable and precise.

Table 3.1 shows the overall mean error computed as the average euclidean error of all 4 corners across the 3 measurements performed on markers ID5, ID4 and ID0H23 as shown in equation 3.4

$$\bar{E} = \frac{1}{12} \sum_{j=1}^4 \sum_{k=1}^3 \sqrt{(x_{jk} - \hat{x}_j)^2 + (y_{jk} - \hat{y}_j)^2} \quad (3.4)$$

Table 3.1: Marker error analysis

Marker	N corners	Overall Mean Err (mm)	Overall Max Err (mm)	Overall Std Dev (mm)
ID5	4	0.625	0.893	0.194
ID4	4	0.344	0.608	0.152
ID0H23	4	0.419	0.622	0.154

Table 3.2 groups the measurements into 4 categories: system baseline, ground level markers, elevated markers and all markers combined. This system baseline was established by detecting the ArUco marker corners printed on a flat A4 sheet. Because the exact relative positions of these corners were known a priori, the error analysis was conducted entirely within the paper's local coordinate frame, bypassing any transformation or extrapolation to the robot's world frame.

Table 3.2: Summary of marker error categories and system baseline

Category	N total points	Overall Mean Error (mm)	Overall Max Error (mm)	Std Dev (mm)
System baseline	36	0.207	0.440	0.148
Flat markers	24	0.485	0.893	0.223
Elevated marker	12	0.419	0.622	0.154
All test markers combined	36	0.463	0.893	0.203

The results indicate that the increase in error caused by manual measurement, slightly different marker height and marker bending, further exacerbated by the use of adhesive tape, is approximately 0.25 mm (average error) and 0.45 mm (maximum error). To mitigate this error, the use of

rigid markers and contact-based position measurement could be implemented. Additionally, the height compensation is validated, as the overall mean error and overall maximum error did not present an observable increase in value.

A bias analysis was also conducted to assess the presence of systematic directional tendencies in the measurements. Table 3.3 shows that measurements on the X axis exhibit a tendency to overestimate corner positions (75 % of measurements), while on the Y axis there is a tendency to underestimate (66.7 % of measurements). It is also noticeable that this bias is more pronounced on the Y axis, as evidenced by the Max and Min Δ values.

Table 3.3: Statistical analysis of bias and interpretation

Statistic	All ΔX	All ΔY
Mean ΔX / Mean ΔY	0.193	-0.188
Std Dev ΔX / Std Dev ΔY	0.217	0.374
Min ΔX / Min ΔY	-0.120	-0.890
Max ΔX / Max ΔY	0.560	0.500
% Negative ΔX / ΔY	25.0 %	66.7 %

Figure 3.2 shows a graph that allows for an easier understanding of the measurement error distribution across both axes, where the concentration of the points outside the center shows evidence of a bias.

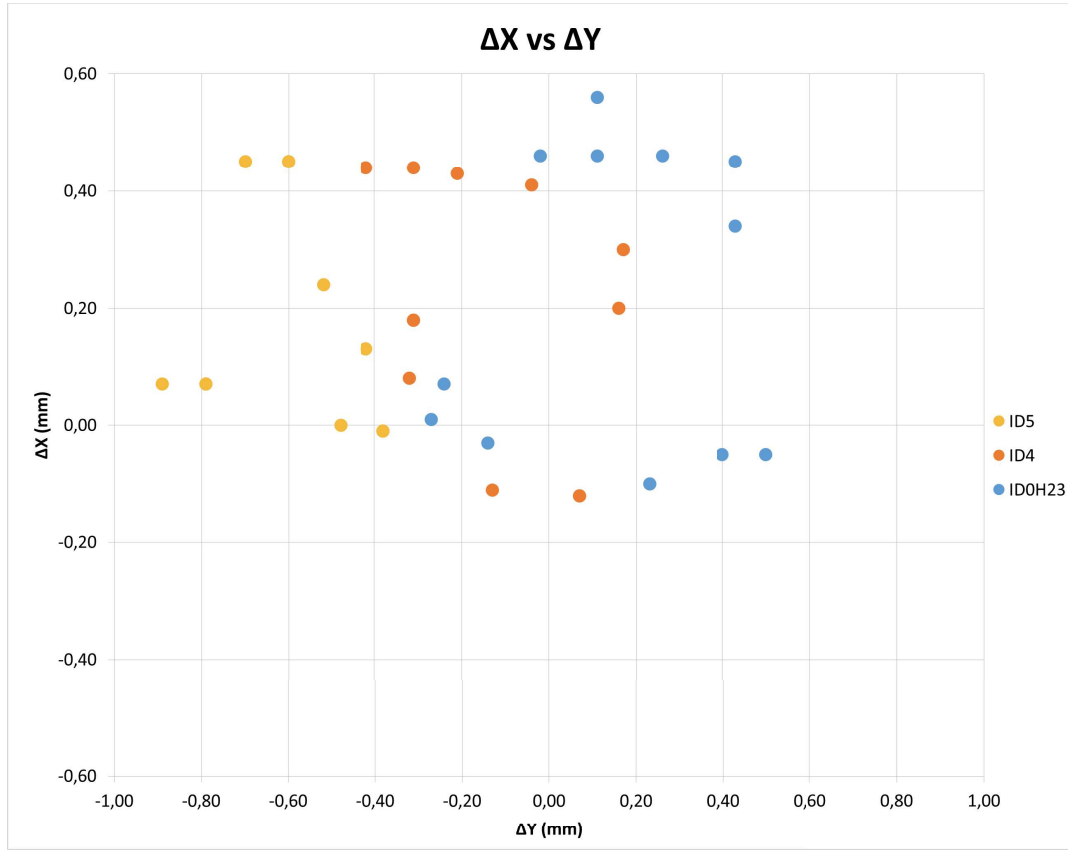


Figure 3.2: Distribution of positional errors across markers.

This graph also reveals that marker ID0H23 exhibits an opposite bias to that observed in the remaining two markers along the Y axis. Examining the spatial coordinates of the markers in Table 3.4, a clear correlation emerges between the Y coordinate of the marker and the sign of the observed ΔY bias: ID5, located at $Y \approx -105\text{mm}$, exhibits a strongly negative mean ΔY of -0.581mm ; ID4, located near $Y \approx -9\text{mm}$, exhibits a small negative mean ΔY of -0.132mm ; and ID0H23, located at $Y \approx +110\text{mm}$, exhibits a positive mean ΔY of $+0.150\text{mm}$. This specific error pattern is typically linked with a small residual tilt of the camera mount around the X axis. A tilted camera introduces a parallax effect in which the projected ray for any given pixel diverges from the ideal vertical ray by an amount proportional to the distance of the measured point from the tilt axis. Points located at negative Y values are displaced further in the negative Y direction, whilst points at positive Y values are displaced in the positive direction, producing the sign reversal observed in the data.

To verify this hypothesis, the elevated marker (ID0h23) was moved to a position where its Y coordinates are negative as shown in Table 3.5.

Table 3.4: Marker corner coordinate data

Marker	Corner	X Coordinate	Y Coordinate
ID5 (Test)	C0	1 513.750	-126.413
	C1	1 504.140	-88.090
	C2	1 542.650	-78.327
	C3	1 552.173	-116.880
ID4 (Test)	C0	1 737.283	6.297
	C1	1 775.960	14.390
	C2	1 784.147	-24.243
	C3	1 745.360	-32.527
ID0H23 (Z=23)	C0	1 601.510	127.917
	C1	1 641.017	131.680
	C2	1 644.740	92.057
	C3	1 605.277	88.603

Table 3.5: ArUco marker ID0H23 — corner position measurements and reprojection error

Corner	Axis	Measured Position (mm)			Mean (mm)	Error (mm)	
		Test1	Test2	Test3		Mean	Std
C0	X	1651.65	1651.80	1651.62	1651.69	+0.61	0.06
	Y	-151.55	-151.60	-151.20	-151.45	-0.99	0.15
C1	X	1611.94	1612.04	1612.09	1612.02	+0.20	0.05
	Y	-152.07	-151.86	-151.59	-151.84	-1.06	0.24
C2	X	1611.27	1611.49	1611.40	1611.39	+0.28	0.11
	Y	-112.29	-112.34	-112.14	-112.26	-0.72	0.10
C3	X	1651.14	1651.09	1651.20	1651.14	+0.49	0.06
	Y	-111.85	-111.85	-111.76	-111.82	-0.69	0.05
Global Mean Error							
X axis						+0.40	0.08
Y axis						-0.87	0.14

The bias changed as predicted, which confirms our hypothesis that the camera tilt around the X axis is influencing the coordinate extraction abilities of the system. This negative effect can be corrected in 2 ways, either ensuring perfect physical alignment between the camera and the printing table, which is easy with the current setup as the camera position is directly connected to the robots end effector. Another possible solution is to compute the function that relates the bias error with the markers position and every time a position is calculated a corrective offset is applied based on the marker's position along the Y axis. The selected solution was to improve camera to

table alignment. To perform this correction a python script capable of calculating the tilt angle across both X and Y axis was developed. The tilt angle was manually corrected, going from a 1.7° and 0.5° to 0.3° and 0.2° across the X and Y axis respectively. A perfect 0° tilt was not achieved, as the developed script presented three limitations that prevented finer correction: a processing latency of approximately 1 second, a measurement variability of around 0.1° between consecutive readings of the same static position, and the fact that the correction itself had to be applied manually by the operator. Overcoming these limitations would require an automatic closed-loop system, where the camera communicates directly with the robot controller to perform small, continuous tilt adjustments without operator intervention. Whether this additional development would be justified depends on the calibration method ultimately adopted, since this approach is inherently more sensitive to residual tilt error than the alternative hand-eye calibration. Given the marginal accuracy gains it would provide relative to the time required for its development and validation, this improvement was not pursued at this stage.

Once the tilt angle was improved a new extrinsic calibration was conducted. Additionally to the tilt angle correction, the ArUcO markers were directly glued to an aluminium block to ensure better flatness than the previous taped ones. Tables 3.6 and 3.7 show the slight improvement achieved both on overall bias reduction and overall accuracy. The bias was reduced, on average, in 0.042 mm (21,8 %) and 0.077 mm (41 %) on the X and Y axis respectively. The % of overestimations went from 75 % to 39.9 % on the X axis and from 33.3 % to 61,1 %. The overall error had a 25 % reduction, on average, and a 3.6 % reduction on maximum error.

Table 3.6: Comparative statistical analysis of bias: original vs. after improvements

Statistic	All ΔX		All ΔY	
	Original	Improved	Original	Improved
Mean	0.193	-0.151	-0.188	0.111
Std Dev	0.217	0.301	0.374	0.202
Min	-0.120	-0.830	-0.890	-0.260
Max	0.560	0.360	0.500	0.450
% Negative	25.0 %	61.1 %	66.7 %	38.9 %

Table 3.7: Comparative error analysis: system baseline, original, and improved results

Configuration	N points	Mean Error (mm)	Max Error (mm)	Std Dev (mm)
System Baseline	36	0.207	0.440	0.148
Original (Combined)	36	0.463	0.893	0.203
Improved (Combined)	36	0.347	0.861	0.210

Further improvements could be achieved by utilizing ArUcO markers with higher resolution, improving the lighting system and improving the manual measurement placing a specific pointer for this end on the robot's end effector.

Hand-Eye Extrinsic Calibration

The limitations identified in the direct correspondence approach, principally the requirement to re-measure reference points every time the robot is repositioned, and the susceptibility to manual measurement uncertainty, motivated the evaluation of a hand-eye calibration methodology. Hand-eye calibration solves for the rigid transformation $\mathbf{T}_{he} \in SE(3)$ between the camera frame and the robot end-effector frame, a quantity that remains constant regardless of robot configuration once the camera is physically fixed to the flange. With $\mathbf{T}_{he}$ known, the world-frame position of any point observed by the camera can be recovered from the robot's forward kinematics alone, without any further physical measurement at the new position.

The classical formulation of hand-eye calibration is the $\mathbf{AX} = \mathbf{XB}$ problem, where $\mathbf{A}$ represents the relative motion of the end-effector between two poses, $\mathbf{B}$ represents the corresponding relative motion of the calibration target in the camera frame, and $\mathbf{X} = \mathbf{T}_{he}$ is the unknown transform to be recovered. Multiple closed-form and iterative solvers have been proposed for this system; in this work four standard OpenCV implementations were evaluated at each calibration attempt: the methods of Park and Martin [31], Tsai and Lenz [32], Horaud and Dornaika [33], and Andreff, Horaud, and Espiau [34]. The mean XY standard deviation for each solver is reported in Figure 3.3, with the Andreff solver consistently achieving the lowest cross-pose consistency error.

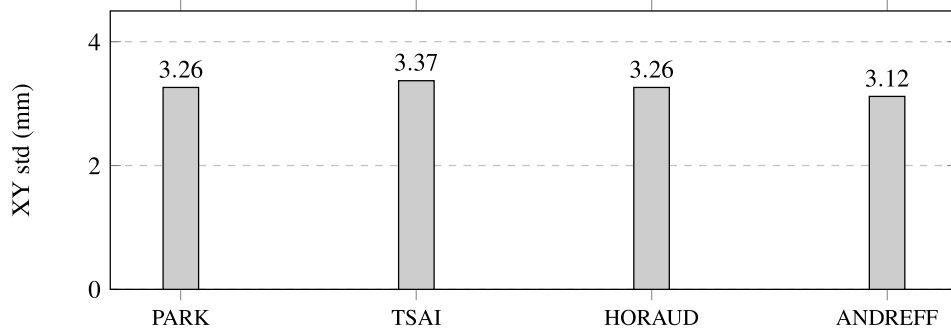


Figure 3.3: Comparison of hand-eye calibration solvers by XY standard deviation. Lower values indicate better cross-pose consistency.

Initial calibration runs used a set of 20 joint poses selected to maximise angular diversity and board tilt variation, with the camera at a nominal sensor-to-board distance of approximately 960 mm. The primary metric used to assess $\mathbf{T}_{he}$ quality is the cross-pose consistency standard

deviation, defined as the standard deviation of a fixed board corner's predicted world-frame position across all calibration poses. Across all four solver methods, this metric was approximately 3.2 mm, well above the 0.6 mm target.

To isolate the root cause, pose diversity was investigated as a primary suspect. The rotation axis coverage of the pose set was quantified by the eigenvalues of the pose conditioning matrix, which returned values of 1.000, 0.960, and 0.279, confirming adequate coverage of all three rotation axes. Pairwise rotation angles between poses spanned 27° to 66° . A singularity filter was subsequently applied, removing all poses with $|j_5| < 15$; the cross-pose standard deviation shifted from 3.233 mm to 3.296 mm, a negligible change that conclusively ruled out both joint-space singularity and pose diversity as contributors to the elevated error. Notably, the five excluded poses were also the only low-tilt poses in the set, implying that board tilt was simultaneously tested and similarly excluded as a cause at this stage.

Attention then shifted to the board itself. The ChArUco calibration target used in the direct correspondence evaluation was fabricated by printing markers on paper and adhering them to an aluminium plate. As previously noted, this introduced minor surface deformations that degraded corner detection accuracy. For all subsequent hand-eye calibration work, this target was replaced with an industrial-grade rigid ChArUco board (20×15 20×15 squares, 8 mm square size, DICT_5X5_1000), whose dimensional accuracy is guaranteed within a fraction of a millimetre.

With detection quality improved, the pose set was also reconsidered. The investigation described above had shown that the elevated cross-pose standard deviation was not attributable to pose diversity or joint singularity, and the exclusion of the five low-tilt poses during the singularity filter test had not worsened the result. This suggested that high board tilt, which reduces the apparent size of the calibration squares in the image and degrades the sub-pixel corner localization accuracy of the ChArUco detector, was a contributing factor that had not been isolated independently. A new set of 12 joint configurations was therefore selected, prioritising lower board tilt angles relative to the camera optical axis and a tighter working envelope around the nominal repair position. With this revised setup, the best AX=XB result (ANDREFF solver) achieved a cross-pose standard deviation of **0.802 mm**, representing an improvement of approximately 84 % over the initial 3.2 mm result.

Table 3.8: Comparison of pose-set diversity metrics between the original 20-pose set and the revised 12-pose set used for hand-eye calibration.

Metric	Original (20 poses)	Revised (12 poses)
Number of poses	20	12
Condition number	3.35	164.14
Pairwise rotation range (°)	83.53	49.04
Tilt min (°)	8.05	1.10
Tilt max (°)	36.29	27.79

As summarised in Table 3.8, the revised pose set is in fact significantly less diverse than the original. The condition number, defined as the ratio between the largest and the smallest eigenvalue of the pose covariance matrix, rises from 3.35 to 164.14, indicating that the revised set concentrates most of its variation along a single dominant axis rather than spreading evenly across the three principal directions of the rotational space. The pairwise rotation range nearly halves as well. Despite this loss of diversity, the calibration accuracy improved substantially. This confirms that the dominant source of error in the original set was not insufficient diversity but rather the high board tilt angles, which the revised set reduces from a range of 8.05°–36.29° down to 1.10°–27.79°.

Despite the improvement achieved by the revised pose set, the $AX=XB$ result was anchored only by cross-pose consistency, a relative constraint that minimises the scatter of predictions across poses but provides no absolute position reference. In practice, this manifested as a systematic translational offset: when the calibration was validated by commanding the nozzle tip to a board corner, errors of 2–3 mm were consistently observed, with high repeatability but wrong absolute position.

To introduce an absolute anchor, a hybrid refinement strategy was developed, inspired by the simultaneous hand-eye and kinematic calibration approach of Cao et al. [35], who used the known distance between precision sphere centres as a constraint to recover both hand-eye and kinematic parameters simultaneously. The constraint structure analogous to their sphere-distance method for a planar board camera setup is the cross-pose consistency condition: the same board corner must project to the same world-frame position from all robot configurations. An additional absolute anchor was introduced by physically measuring the world-frame position of six ArUco marker top-left corners (IDs 0, 9, 60, 69, 140, 149) using the robot nozzle tip in contact with each corner and reading the pendant position display. To maximise the information available to the optimiser, the 12-pose endpoint set was expanded: 4 intermediate joint-space interpolated stops were added between each consecutive pair of endpoint poses, bringing the total number of calibration captures to 56. The full optimization minimized a combined objective:

$$\mathbf{T}_{he}^* = \arg \min_{\mathbf{T}_{he}} \underbrace{\sum_i \sum_{m \in \mathcal{M}} \left\| \mathbf{T}_{fk}^{(i)} \mathbf{T}_{he} \mathbf{p}_m^{(i)} - \mathbf{g}_m \right\|^2}_{\text{point constraint}} + \lambda \underbrace{\sum_m \sum_{i \neq j} \left\| \mathbf{T}_{fk}^{(i)} \mathbf{T}_{he} \mathbf{p}_m^{(i)} - \mathbf{T}_{fk}^{(j)} \mathbf{T}_{he} \mathbf{p}_m^{(j)} \right\|^2}_{\text{cross-pose consistency}} \quad (3.5)$$

where $\mathbf{T}_{fk}^{(i)}$ is the robot forward kinematics transform at pose i (obtained directly from the ABB controller via `CROBT`), $\mathbf{p}_m^{(i)}$ is the camera-frame position of marker corner m at pose i (obtained from `solvePnP`), $\mathbf{g}_m$ is the physically measured world-frame position of corner m , and $\lambda = 0.5$ is the relative weight of the consistency constraint. The RAPID controller FK was used directly rather than an independently computed DH model, since the controller's internal kinematic model, including any AbsAcc compensation that may have been applied, is the authoritative source of flange position and cannot be improved upon without a laser tracker.

The optimization was initialised at the best AX=XB result and solved using a trust-region reflective algorithm with bounds of ± 15 mm on translational corrections and $\pm 3^\circ$ on rotational corrections. Convergence was confirmed with final corrections of $\Delta x = +4.99$ mm, $\Delta y = +1.69$ mm, $\Delta z = +1.91$ mm, $\Delta r_x = +0.27^\circ$, $\Delta r_y = -0.23^\circ$, $\Delta r_z = -0.02^\circ$. The combined cross-pose consistency standard deviation reduced from 1.611 mm to 0.832 mm. The resulting $\mathbf{T}_{he}$ was selected over the AX=XB-only result based on this improvement.

The AX=XB formulation estimates the rotation component of $\mathbf{T}_{he}$ from the relative angular motion of the board across calibration poses. The positional error induced by a residual rotation error $\delta\theta$ scales linearly with the sensor-to-board distance d as $\epsilon_{xy} \approx d \cdot \tan(\delta\theta)$. Halving the working distance therefore halves the position error contribution from rotation uncertainty, independently of the number of calibration poses. For this reason, the calibration was repeated with the camera repositioned to a sensor-to-board distance of approximately 500 mm, reducing the approximate FOV from 500×500 mm to 300×300 mm and the Ground Sampling Distance (**GSD**) from 0.111 mm/pixel to approximately 0.067 mm/pixel. At the closer working distance the calibration produced a cross-pose standard deviation of **0.522 mm** (ANDREFF solver).

Hand-Eye Calibration Validation

Validation followed the same methodology applied in the direct correspondence evaluation, ensuring a fair and directly comparable assessment. Three ArUco marker top-left corners (IDs 5, 65, and 145) not used in the calibration procedure were selected as validation points, placed at board positions X=89 mm (centre column), Y=1 mm, 49 mm, and 113 mm respectively. For each marker, the nozzle tip was placed directly above the corner and the world-frame position was recorded by reading the robot controller's position display. To mitigate manual measurement uncertainty, each corner was measured three times with robot movement between measurements and the results averaged, consistent with the procedure described in Section ?? . A single image was then captured

from the robot’s operating pose and `solvePnP` was used to predict the corner positions, which were compared against the averaged physical measurements. Results are presented in Table 3.9.

Table 3.9: Hand-eye calibration validation results.

Marker	Pred X (mm)	Pred Y (mm)	Pred Z (mm)	Meas X (mm)	Meas Y (mm)	E_{XY} (mm)
ID 5	74.243	17.791	55.033	74.314	17.967	0.189
ID 65	27.870	5.401	54.903	27.946	5.578	0.192
ID 145	-33.961	-11.120	54.730	-34.013	-10.825	0.299
$\bar{E}_{XY}$						0.227
$E_{XY}^{\max}$						0.299
s						0.051

These results represent a mean XY error of **0.227 mm** and a maximum XY error of **0.299 mm**, both within the 0.3 mm average and 0.6 mm maximum targets respectively, and corresponding to improvements of **34.6 %** in mean error and **65.3 %** in maximum error relative to the direct correspondence result (0.347 mm mean, 0.861 mm maximum after improvements, displayed in Table 3.7). The standard deviation of 0.051 mm reflects high consistency across markers, indicating no significant spatial bias at this working pose.

A key advantage of the hand-eye approach over direct correspondence is that the $\mathbf{T}_{he}$ is valid from any robot configuration, not only the pose at which it was calibrated. To validate this flexibility, the same three-marker validation was repeated from two additional robot starting positions, distinct from the pose used in Table 3.9. This test evaluates whether the calibration generalizes across the workspace, which is the primary practical motivation for adopting the hand-eye method. Results are presented in Table 3.10.

Table 3.10: Hand-eye calibration validation from multiple robot starting positions. Same validation markers and methodology as Table 3.9.

Robot pose	Marker	Mean E_{XY}	Max E_{XY}
Pose A	ID 5 / ID 65 / ID 145	0.293	0.374
Pose B	ID 5 / ID 65 / ID 145	0.076	0.130
Pose C (original)	ID 5 / ID 65 / ID 145	0.227	0.299

A comparative summary across all methods evaluated is presented in Table 3.11.

Table 3.11: Comparative summary of extrinsic calibration methods evaluated. Percentage improvements are relative to the preceding method in the sequence.

Method	Mean E_{XY} (mm)	Max E_{XY} (mm)	Std Dev (mm)	Improvement mean / max
Direct correspondence (original)	0.463	0.893	0.203	—
Direct correspondence (improved)	0.347	0.861	0.210	25.1 % / 3.6 %
AX=XB, original poses, 500 mm FOV	$\sim 3.2^\dagger$	—	—	—
AX=XB, revised poses, 500 mm FOV.	$0.802^\dagger$	—	—	75 % [†]
AX=XB + refinement, 300 mm FOV.	0.227	0.299	0.051	56.4 % / 65.3 % [‡]

[†] Cross-pose consistency std; live validation not performed at this stage.

[‡] Improvement relative to the cross-pose consistency std of the AX=XB, revised poses with 500 mm FOV.

In the intended deployment scenario, the system must operate at two distinct working distances: a wider field of view at approximately 960 mm for nesting tasks, where the full table surface must be visible, and the reduced 300×300 mm field of view at 500 mm for repair tasks requiring sub-millimetre precision. Performing a full hand-eye recalibration every time the camera is repositioned between these two configurations would not be ideal and would reduce the flexibility advantage of the hand-eye approach. Since $\mathbf{T}_{he}$ is a physical constant of the camera mount and independent of working distance, the hypothesis was that only the intrinsic matrix $\mathbf{K}$ would need to be recomputed after raising the camera and refocusing the lens, as adjusting the focus ring shifts the effective focal length and principal point. To validate this, the same three-marker validation was performed at the 960 mm configuration after intrinsic recalibration at the new focus distance, with $\mathbf{T}_{he}$ unchanged from the 500 mm calibration. The expected increase in absolute error due to the larger working distance, where any residual rotation error in $\mathbf{T}_{he}$ projects over a longer lever arm, was quantified and is presented in Table 3.12.

Table 3.12: Hand-eye calibration validation at 960 mm working distance after intrinsic-only recalibration.

Marker	Pred X (mm)	Pred Y (mm)	Pred Z (mm)	Meas X (mm)	Meas Y (mm)	E_{XY} (mm)
ID 5	75.016	18.180	27.837	74.314	17.967	0.734
ID 65	28.662	5.719	27.563	27.946	5.578	0.730
ID 145	-33.142	-10.895	27.197	-34.013	-10.825	0.873
<i>Mean</i>						0.779
<i>Maximum</i>						0.873
<i>Std Dev</i>						0.067

The results indicate that intrinsic-only recalibration is insufficient to maintain the target accuracy after a substantial change in the lens focus. The XY error increased from 0.227 mm to 0.779 mm at 960 mm, and a subsequent test at the original 500 mm working distance, after refocusing and redoing intrinsics only, also degraded to 0.885 mm mean XY error, confirming that the degradation is not solely attributable to the increased working distance but occurs whenever the lens is significantly refocused without repeating the extrinsic calibration. This suggests that adjusting the focus ring introduces mechanical changes beyond a simple focal length shift, likely small physical displacements of the lens assembly that alter the effective camera-to-flange transform. In practice, however, small focus corrections, such as those required when we simply change substrates of similar size, were observed to leave the extrinsic calibration within tolerance, and only larger refocusing operations between substantially different working distances were found to demand a full recalibration. A precise quantification of the focus-shift threshold above which extrinsic recalibration becomes necessary was not pursued in this work, as it would require a dedicated experimental campaign with controlled, repeatable focus adjustments and is left for future study. As a conservative operational rule, a full intrinsic and extrinsic recalibration is performed whenever the lens focus is meaningfully adjusted between deployment configurations.

Part-Height Compensation

A final practical consideration concerns the relationship between the calibrated extrinsic transform and parts of varying thickness. The hand-eye calibration recovers $\mathbf{T}_{he}$ relative to a specific calibration plane, namely the surface of the ChArUco board. However, during operation the camera observes substrates whose top surface lies at a different height than this reference plane, and because the camera is rigidly mounted off-axis on the robot flange, any deviation between the actual part height and the calibration plane introduces a parallax offset: the same image pixel corresponds to different (X, Y) world coordinates depending on which Z-plane the corresponding optical ray is intersected with. Left uncompensated, this offset would manifest as a systematic translational error proportional to the height difference and to the off-axis distance of the camera from the nozzle.

To address this, the pixel-to-world projection is implemented as a ray back-projection onto a parametric Z-plane rather than as a fixed homography. For a pixel (u, v) and a known part top-surface height z_{part} , the camera-frame ray is recovered as $\mathbf{r}_{\text{cam}} = \mathbf{K}^{-1} \begin{bmatrix} u & v & 1 \end{bmatrix}^\top$, rotated into the world frame via the extrinsic transform $\mathbf{r}_{\text{world}} = \mathbf{R}_{\text{cam} \rightarrow \text{world}} \mathbf{r}_{\text{cam}}$, and intersected with the horizontal plane $z = z_{\text{part}}$:

$$\mathbf{p}_{\text{world}} = \mathbf{o}_{\text{world}} + \frac{z_{\text{part}} - o_{\text{world},z}}{r_{\text{world},z}} \mathbf{r}_{\text{world}}, \quad (3.6)$$

where $\mathbf{o}_{\text{world}}$ is the camera optical centre expressed in world coordinates, extracted from the hand-eye matrix $\mathbf{T}_{he}$. The part height is supplied by the operator at the start of each repair job and

propagated through the pipeline, ensuring that the ray is always intersected with the actual top surface rather than with the fixed calibration plane. This eliminates the parallax-induced translational error without requiring any recalibration when the part geometry changes.

Selected Method and Possible Improvements

The hybrid $AX=XB$ with fixed-point simultaneous refinement, calibrated at a 500 mm working distance, is selected as the extrinsic calibration method for this system. Compared to the direct correspondence approach, it achieves a similar mean error (0.227 mm versus 0.347 mm) while delivering substantially more consistent results, with a maximum error of 0.299 mm against 0.861 mm, a reduction of 65.3 %. This consistency is particularly relevant for repair applications, where a single outlier prediction can misalign an entire toolpath. Beyond precision, this method offers greater flexibility since the $\mathbf{T}_{he}$ is a physical constant of the camera mount, remaining valid until the camera is physically disturbed or a big adjustment is performed in the lens. Even though this hinders the transition from a system calibrated for a nesting operation with a larger FOV to repair operation with a much smaller FOV, the flexibility is still an advantage when developing new features, and since the time necessary for both calibrations is similar, direct correspondence offers no substantial advantage. In a scenario where the aruco markers can be permanently fixed to the printing table with known positions, the direct correspondence approach could become the best option.

For the precision requirements of the current repair tasks, the achieved accuracy is sufficient. Should a future application demand sub 0.1 mm accuracy, the primary limiting factor is the absolute accuracy of the robot's forward kinematics, which places a floor that cannot be reduced by improving the calibration procedure alone. In that scenario, a laser tracker measurement could be used to characterise and compensate the kinematic model at each joint configuration, following the approach of Cao et al. [35], which demonstrated that simultaneous hand-eye and kinematic calibration can reduce positioning errors to well below 0.1 mm on comparable industrial manipulators.

3.4 Projector System Calibration

The calibration of the projection system is mathematically treated as an inverse camera model. In this framework, the projector is considered a device that captures light from the scene, where the 2D image coordinates correspond to the pixels of the projector's Digital Micromirror Device (DMD) array. The calibration procedure is divided into two independent phases: intrinsic calibration of the projector and stereo calibration of the camera-projector pair.

In the first phase, the projector intrinsic parameters are estimated using the robot to drive the camera through a set of 20 poses spanning a wide range of orientations relative to a new large ChArUco board (12×12 squares, 32 mm side length). At each pose, the full Gray code sequence

is projected and captured. The ChArUco corners detected in the white-illuminated frame are decoded to their corresponding projector pixel coordinates (p_x, p_y) by querying the decoded correspondence map at those pixel locations. This process generates a set of 3D-to-2D correspondences between the physical coordinates of the ChArUco board and the projector pixel space, which are used as input for `cv2.calibrateCamera` to estimate the projector intrinsic matrix K_p and distortion coefficients D_p . The resulting intrinsic reprojection error was 0.54 pixels.

In the second phase, the stereo extrinsic parameters are estimated independently of the intrinsics. With the robot fixed at the operating camera position, the original smaller ChArUco board was manually tilted to 10 different orientations within the projection area. At each orientation, the Gray code sequence was projected and decoded, and the detected board corners were mapped to their corresponding projector pixel coordinates. The stereo calibration was then performed using `cv2.stereoCalibrate` with the `CALIB_FIX_INTRINSIC` flag, keeping both the previously computed camera intrinsics and the projector intrinsics fixed. This yields the rotation matrix R and translation vector t describing the rigid transformation between the camera and the projector coordinate frames, with a stereo reprojection error of 0.56 pixels. This extrinsic relationship is fundamental for the triangulation process, as it defines the stereo baseline required to recover metric depth from the structured light correspondences.

3.5 Slicing Aided Hyper Inference (SAHI)

To perform both image slicing and reconstruction after inference the SAHI algorithm was used, this allowed for the use of 4504×4504 pixel images without overwhelming the deep learning models. The created slices for both training and inference have a 1024×1204 pixel dimension and a 30 % overlap. The overlap was chosen following the articles recommendation of 25 %, slightly increased by 5 % to reduce the computational weight of this step ensuring the performance of the model was not undermined. The application of the slicing algorithm to the 4504×4504 pixel imagery resulted in 36 tiles per image at a resolution of 1024×1024 pixels.

3.6 Dataset Preparation

Once the calibration procedure was concluded and the hardware is now ready to be implemented in our systems, we now need to acquire the necessary data upon which the deeplearning models will be developed and tested. This chapter addresses the creation of two distinct datasets developed to support the machine learning components of the system. The first dataset supports the nesting pipeline, enabling the model to identify and localise substrates and existing prints on the work-table. The second dataset supports the repair pipeline, targeting the detection of repair zones. For each dataset, the relevant details are described, including substrate types and geometries, number of images and capture poses, annotation methodology, and the augmentation strategy applied to expand the available training data.

3.6.1 Dataset for the Nesting System

In order to train the models to be used by the nesting system, the creation of a dataset capable of representing the variety of substrates(geometry and material), fixing configurations and lighting conditions encountered in practice is imperative. To this end, 70 undistorted images were captured, encompassing 8 distinct substrate geometries, fixed with either one or two clamps, in multiple positions, and with varying states of previous deposition, ranging from none to multiple prior prints. To expand the available data, a position-focused augmentation strategy was applied. Given that the primary sources of variability: substrate geometry, material and texture, presence of previous prints, and clamp configuration were already represented in the original dataset, and that the lighting conditions in the cell where the robot operates are stable, the augmentation was focused on systematic rotations and flips. This approach accurately simulates the positional entropy inherent to manual substrate placement, where operators do not follow any fixed orientation convention. Figure 3.4 presents a selection of the images captured prior to augmentation, provided for contextual reference.

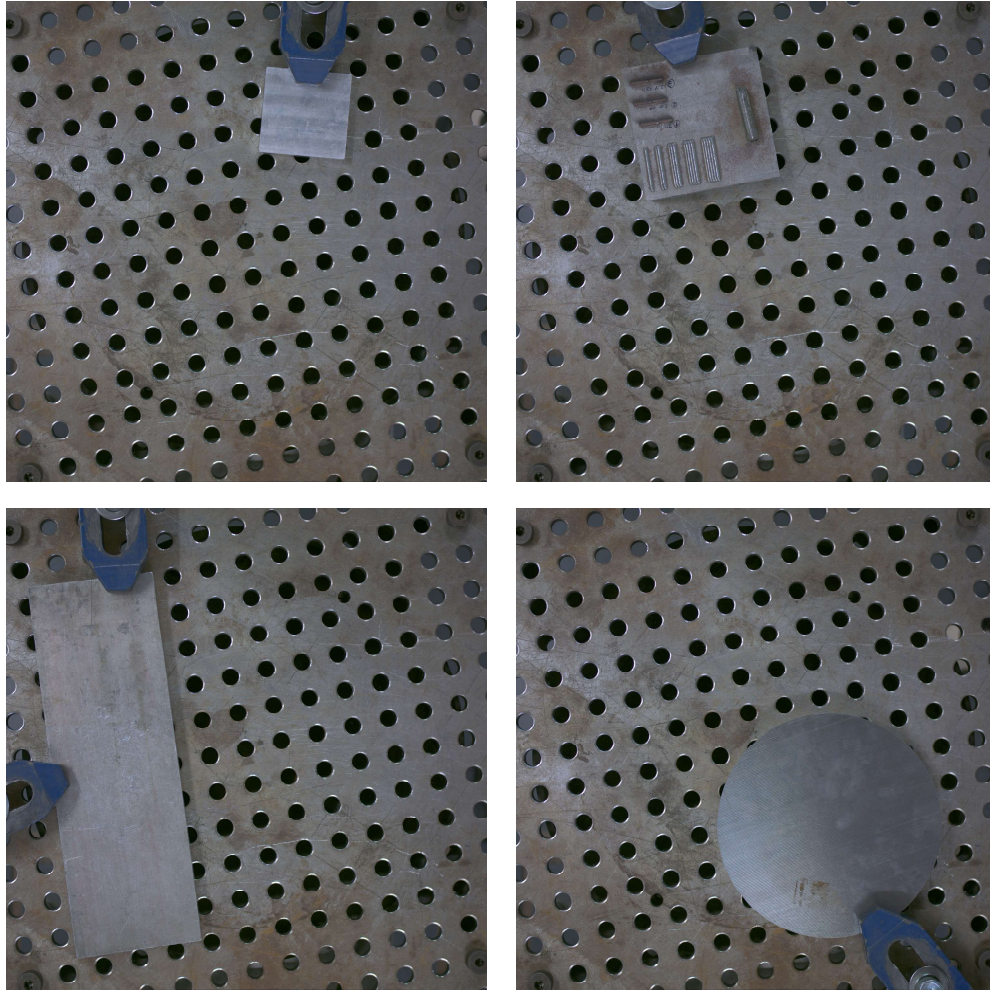


Figure 3.4: Sample images from the dataset prior to augmentation.

3.6.2 Dataset Preparation for the Repair System

In order to train the segmentation model used by the repair pipeline, a dataset capable of representing the variety of substrate geometries, surface conditions, and spatial configurations encountered in practice is required. A fundamental challenge in this context is the absence of real repair specimens with actual material defects suitable for training purposes. To address this limitation, a proxy dataset was created by placing rectified parts or additional substrates on top of base substrates, simulating the elevated surface topology characteristic of a repair zone. Only the top face of the overlaid component was annotated as the repair area, approximating the region that would require material deposition in a real repair scenario. A total of 27 RGB-D image pairs were captured using the Basler camera and the structured light projector system described in Chapter 3.8. Each capture consists of a raw RGB image, a metric depth map, and the corresponding instance segmentation annotation. The dataset encompasses 8 distinct substrate geometries and clamp configurations, captured from the fixed camera position used during system operation. An additional

4 RGB-D pairs, featuring previously unseen substrate and clamp arrangements, were captured separately and held out as a test set for the final evaluation of the trained model.

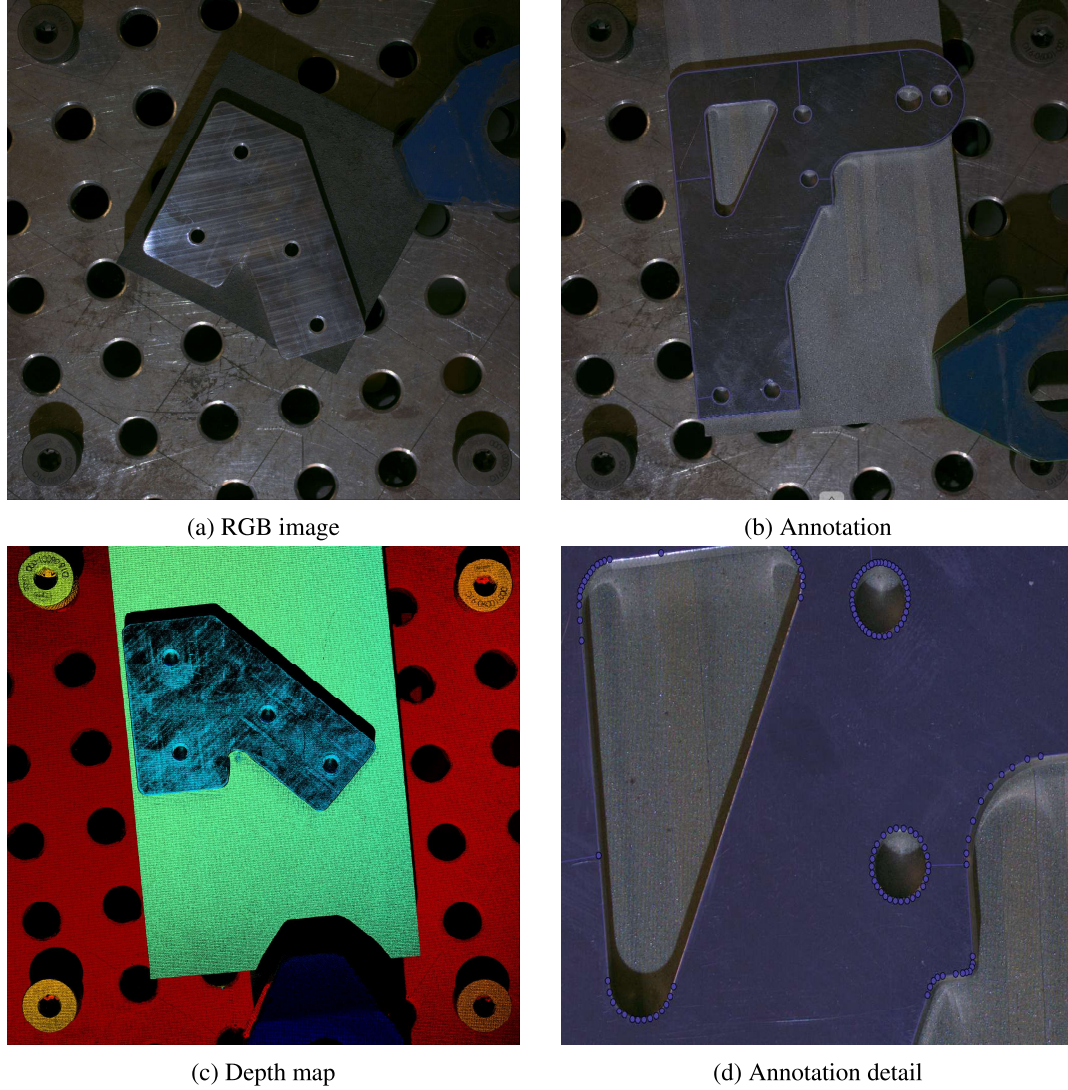


Figure 3.5: Representative samples from the repair dataset: (a) RGB image, (b) instance annotation, (c) depth map visualization, and (d) annotation detail.

To expand the available training data, a position-focused augmentation strategy was applied. Given that the primary sources of variability: substrate geometry, surface condition, and clamp configuration, are already represented in the original captures, and that the illumination conditions within the DED cell remain stable during operation, the augmentation was focused on systematic rotations, flips, and transpositions. This approach accurately simulates the positional variability inherent to manual substrate placement, where operators do not follow a fixed orientation convention. The combined RGB and depth maps were transformed with identical geometric operations

to preserve pixel-level alignment between modalities, while photometric augmentations such as brightness and contrast variation were applied exclusively to the RGB channel. The 27 original captures were expanded into 300 full-resolution augmented samples, which were then split at the specimen level into training and validation subsets following an 80/20 ratio, ensuring that all augmentations derived from a given original belong to the same subset and thus preventing data leakage. Because the native sensor resolution is too large to be processed in a single forward pass by the segmentation backbone, each augmented image was subsequently partitioned into 1024×1024 tiles with a 30% overlap between adjacent tiles. RGB, depth, and label maps were tiled coherently using identical coordinates, and tiles containing no valid annotation were retained to provide negative background context. This two-stage expansion, augmentation followed by overlapping tiling, ultimately yielded 17,280 tiles for training and 6,120 tiles for validation, while the 4 held-out captures were reserved for full-image evaluation.

3.7 Segmentation Pipeline Architecture

With the dataset preparation phase concluded, the training of the deep learning models responsible for the segmentation pipeline was initiated. This section details the development and evaluation of the two-stage vision system, starting with the training of the object detection model used to generate the prompts required by the segmentation stage. It then addresses the segmentation task itself, describing the fine-tuning process, training strategy, and results obtained for both the HQ-SAM and Mask2Former architectures. Finally, a comparative evaluation between the two segmentation approaches is presented, culminating in the justification for the model selected for integration into the final system.

3.7.1 Object Detection for Prompt Generation

As mentioned before, both SAM and HQ-SAM are prompt dependent models, so in order to create a fully autonomous operation, a YOLOv8 model was used to create the bounding boxes where the segmentation model will operate. The motivation to chose this model follows the line of thought discussed in Section 2.3, where it is stated that this model has a great trade-off between training complexity and required computational power to object detection capacity. If the combined results of HQ-SAM and YOLOv8 do not satisfy the project's requirements other models will be tested.

The training phase utilized the AdamW optimizer with an initial learning rate of 5×10^{-4} with cosine decay schedule. The total loss function is defined as a weighted sum of three distinct components:

$$L_{total} = \lambda_{box}L_{box} + \lambda_{cls}L_{cls} + \lambda_{dfl}L_{dfl} \quad (3.7)$$

where:

- L_{box} (**Complete IoU Loss**): Optimizes the overlap, distance, and aspect ratio between the predicted and ground truth bounding boxes.
- L_{cls} (**Binary Cross Entropy**): Measures the classification error for each category using Binary Cross Entropy (**BCE**) over the predicted class probabilities.
- L_{dfl} (**Distribution Focal Loss**): Refines the boundaries of the bounding boxes by modelling the probability distribution of the box edges, particularly useful for datasets with imprecise boundaries.

Throughout the 100 epochs the loss functions converged consistently as observed in Figure 3.6.

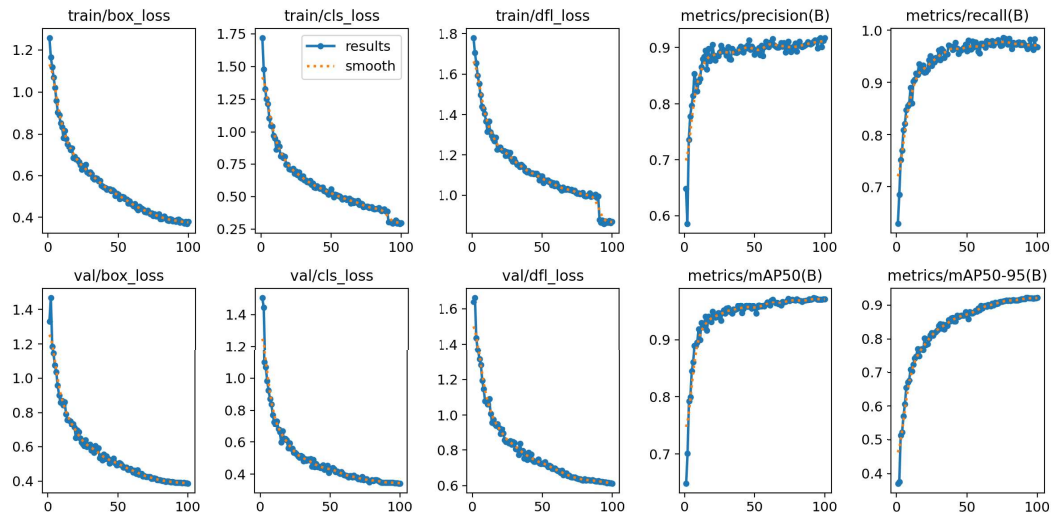


Figure 3.6: Yolo training results

Additionally, it is also possible to observe the resulting model reached its peak performance at epoch 92, achieving a mAP_{50} of 0.9739 and a mAP_{50-95} of 0.9221. Such high values for the mAP_{50-95} metric are critical for this application, as they signify not only that the objects are correctly identified but that the bounding boxes are localized with the high precision required for the subsequent segmentation and nesting calculations. With a recall of 0.9696 and a precision of 0.9103, the detector maintains an exceptional balance between sensitivity and specificity, ensuring that almost every relevant feature is captured while minimizing false alarms that could disrupt the autonomous workflow. To verify the output of the model a confusion matrix was created, as shown in Figure 3.7, from which we can conclude that most predictions across all classes were correct.

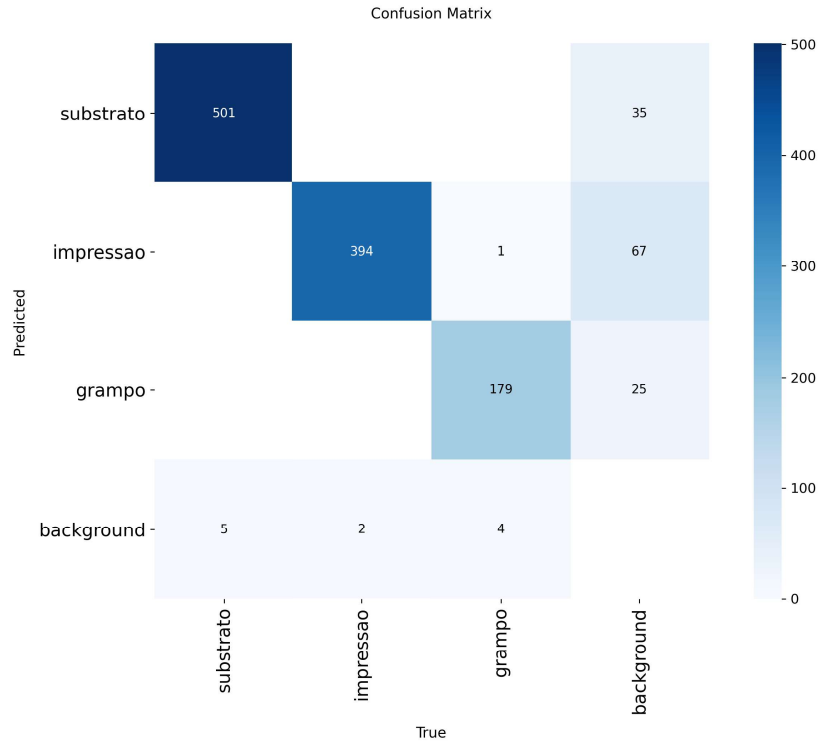


Figure 3.7: Yolo predictions confusion matrix

3.7.2 Object Segmentation for Contour Extraction

After object detection, the final segmentation still needs to be performed. Two candidate models were trained and evaluated for this task: HQ-SAM, which relies on the previously trained YOLOv8 detector to provide the bounding box prompts, and Mask2Former, which performs segmentation independently without requiring an auxiliary detection stage.

Training HQ-SAM

The HQ-SAM ViT-L backbone was fine-tuned on a dataset of annotated DED cell images to adapt its segmentation behaviour to three target classes: "*substrato*", "*impressão*", and "*grampo*". Following the adapter-based fine-tuning strategy proposed in Medical SAM Adapter Wu et al. [14], a parameter-efficient approach was adopted by combining Low-Rank Adaptation (LoRA) on the image encoder with full fine-tuning of the mask decoder. The training dataset consisted of tiled images at 1024×1024 pixels with 30 % overlap, resulting in 4.302 training samples(75.5 %), 1.086 validation samples (19 %), and 308 test samples(5.5 %).

Instead of retraining the full 307M-parameter encoder, Low-Rank Adaptation (LoRA) was employed by inserting trainable adapter modules into the query, key, and value projection layers of

the final eight transformer blocks. The adaptation used a rank (r) of 8, constraining the weight updates to a low-dimensional subspace (only 8 dimensions in 1024, for the ViT-L encoder) to minimize computational overhead, and a scaling factor (α), (the updates are multiplied by $\frac{\alpha}{r}$), of 1.0 to balance the influence of the adapters against the frozen pre-trained weights. This configuration introduced only a small fraction of additional trainable parameters, approximately 1 % of the total encoder, while preserving the rich visual representations of the foundation model. While the encoder neck remained trainable and the rest of the backbone was frozen, the 4.1M, parameter HQ mask decoder was fully fine-tuned. This ensures that the high-fidelity mask generation remains highly responsive to the specific geometries of the prints and substrates, guided by the bounding box prompts provided by the preceding detection stage.

The training objective combined multiple complementary loss terms to balance region accuracy and contour quality. The loss function included focal loss to address class imbalance, Dice coefficient (**DICE**) loss to enforce overlap accuracy, mean squared error for pixel-level supervision, and a boundary-aware term to improve contour delineation. Class-specific weighting was applied, with a significantly higher focal weight assigned to the substrate due to its large spatial extent and visual similarity to the surrounding table. This weighting encouraged the model to better resolve ambiguous regions that are commonly misclassified by the baseline model.

Optimization was performed using separate Adam optimizers for the encoder adaptations and the decoder, reflecting their different learning dynamics. The decoder was trained with a higher learning rate and a plateau-based scheduler, while the LoRA and neck parameters followed a lower learning rate with cosine annealing. The cosine annealing scheduling strategy gradually reduces the learning rate following a cosine curve over the course of training, allowing for larger update steps early on and progressively finer adjustments as the model converges, which helps avoid abrupt drops that could destabilize the LoRA adaptation. Regularization included weight decay, gradient clipping, and gradient accumulation to maintain stable training under memory constraints. Additionally, a knowledge distillation term was incorporated to constrain the fine-tuned decoder to remain close to the original HQ-SAM predictions, preventing degradation of its general segmentation capabilities.

Training was conducted for 50 epochs, with the best model selected based on validation IoU at epoch 40, achieving a global validation IoU of 0.953. Figure ?? displays a visual comparison between the default and fine tuned model highlighting the improvements mentioned earlier in this section.

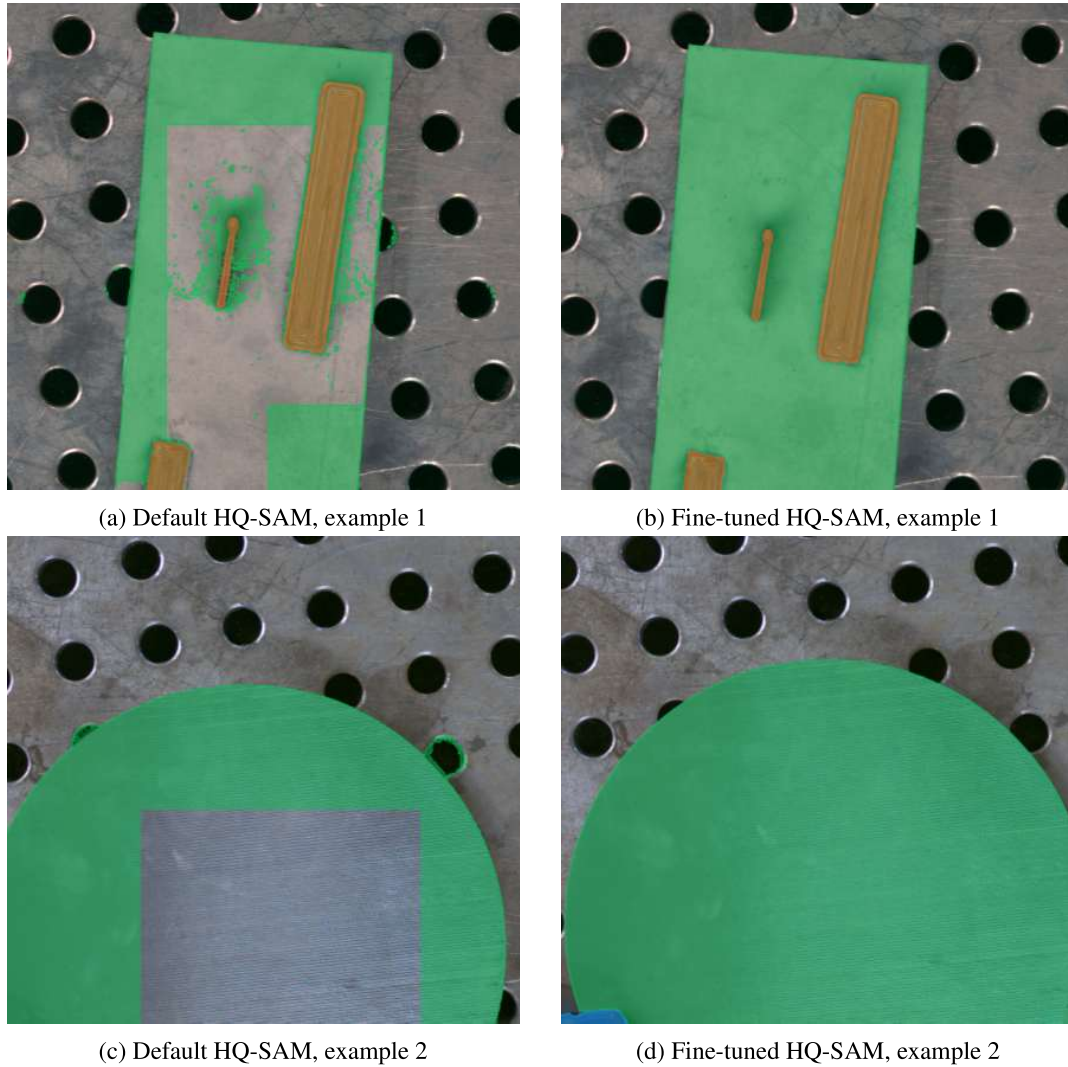


Figure 3.8: Visual comparison between the default HQ-SAM (a, c) and the fine-tuned model (b, d) on two representative substrate examples.

Training Mask2Former

In order to exploit a different approach, a Mask2Former model was fine-tuned with an identical approach, utilizing a Swin-Small backbone, previously trained on the COCO (Common Objects in Context) dataset, to balance architectural complexity with the hardware constraints of an 8GB Video Random Access Memory (VRAM) Graphics Processing Unit (GPU). A dual-optimizer scheme was employed to apply differential learning rates: a conservative rate for the pre-trained backbone to retain feature extraction capabilities, and a more aggressive rate for the Mask2Former decoder and transformer modules to specialize in the specific morphology of the 3 dataset classes. Unlike the HQ-SAM model this Mask2former model was trained to perform Panoptic segmentation, meaning both instance and semantic segmentation at the same time, this helps not only to

avoid the same pixel being assigned more than one category but also the ability to differentiate multiple objects of the same class.

While the global IoU reached a plateau almost instantly, the Panoptic Quality (PQ) and class-specific metrics revealed a more nuanced learning curve. In the initial epochs, the "*substrato*" class exhibited lower IoU and PQ scores (e.g., 0.4540 in Epoch 1) compared to the "*grampo*" and "*impressão*" classes, which consistently stayed above 0.85. This disparity was mitigated by the dual-optimizer training scheme. As the transformer decoder progressively specialized in class-level representations, the PQ of the "*substrato*" improved, though it remained the most challenging morphology to segment accurately. To ensure the model focused on improving this class more than the others, as this is the most important one for nesting, the training was restarted from scratch by reconfiguring the "*substrato*" from a *STUFF* label to a *THING* label and assigning all unannotated pixels to an ignore index (255), completely eliminating background flooding and forcing the loss function to evaluate gradients strictly from explicit mask instances.

Final Evaluation

A final evaluation was conducted to validate the selected model. During this phase, new substrate images were acquired to rigorously verify that the models had not overfitted or developed a dependency on features specific to the training set. The established metrics were recalculated and, consistent with previous iterations, a subset of images was reserved for qualitative visual validation. Figures 3.9, 3.10 and 3.11 display the IoU, DICE and Boundary IoU comparisons across the three models.

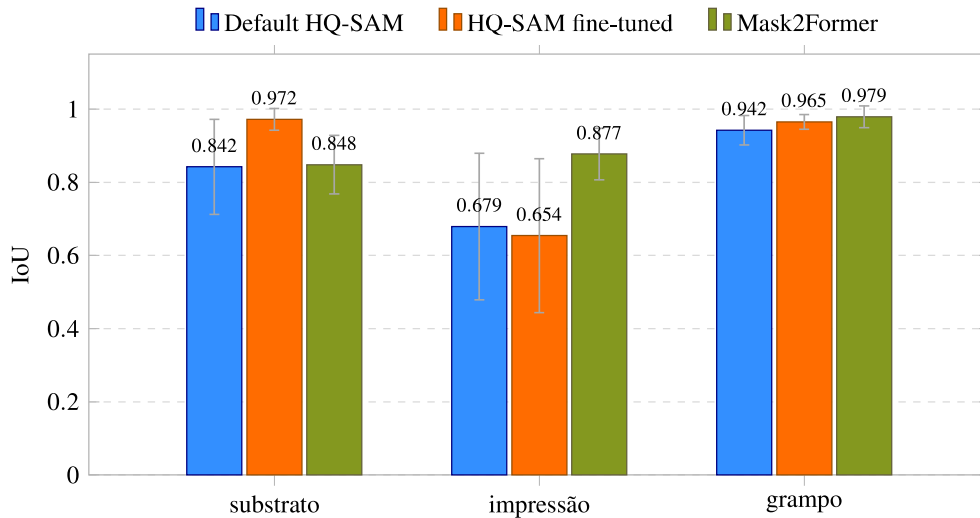


Figure 3.9: IoU comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.

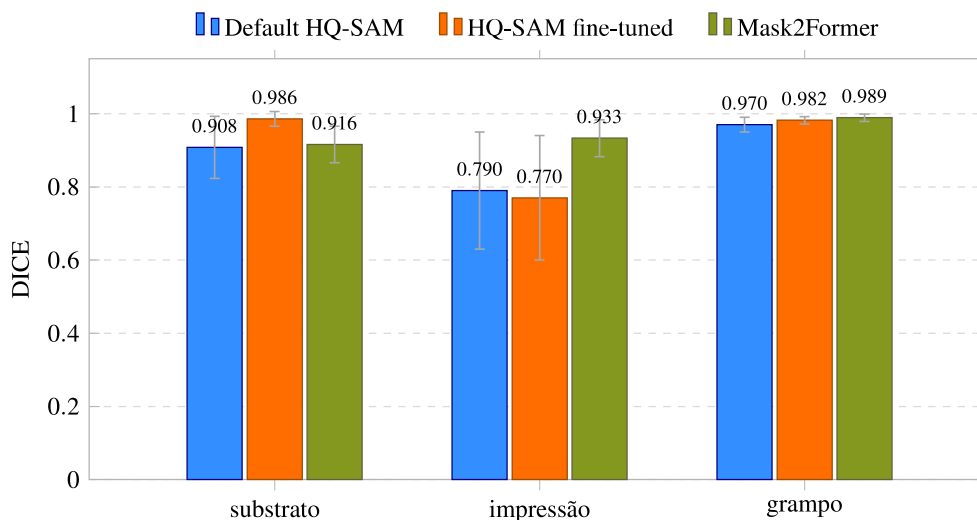


Figure 3.10: DICE coefficient comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.

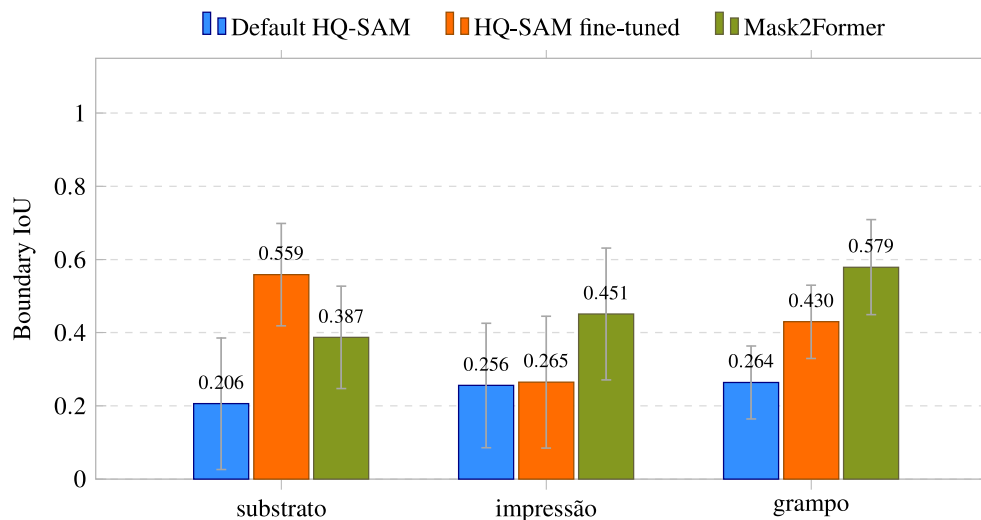
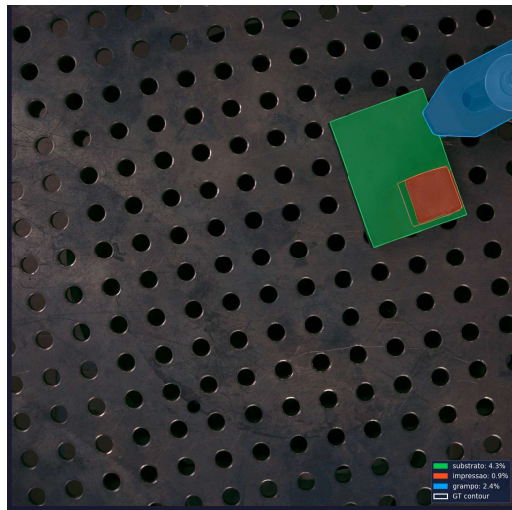


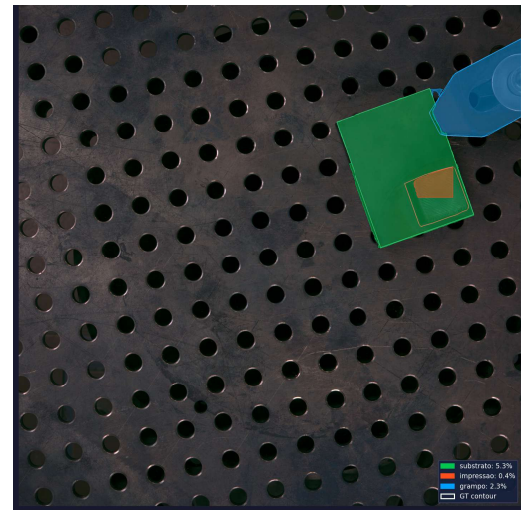
Figure 3.11: Boundary IoU comparison between the original HQ-SAM model, the fine-tuned HQ-SAM and the Mask2Former model, evaluated on the held-out test set. Error bars represent the standard deviation across test images.

Across the three metrics, no single model dominates on every class. Mask2Former leads on the "*impressão*" and "*grampo*" classes, whereas the fine-tuned HQ-SAM achieves the highest scores on the "*substrato*" class, particularly on the area-based IoU and DICE. The default HQ-SAM consistently trails both, confirming the benefit of the fine-tuning stage. The Boundary IoU values are markedly lower than the area-based metrics for all models, reflecting the intrinsic difficulty of pixel-accurate contour delineation under the low-contrast conditions of this task.

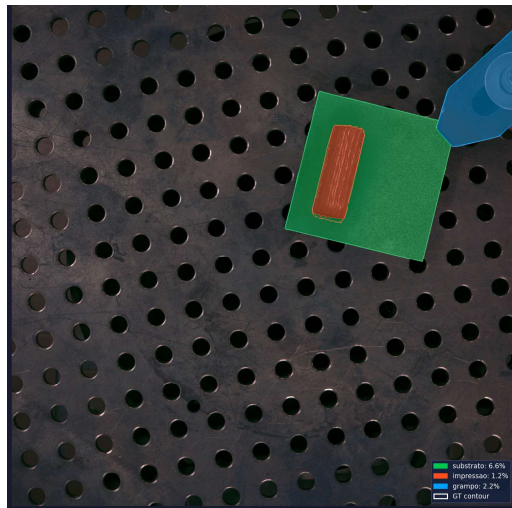
Figure 3.12 shows the outputs of the three models, with the predicted contours overlaid on the original images. This allows a direct comparison with the ground truth and helps identify localization errors and challenging cases.



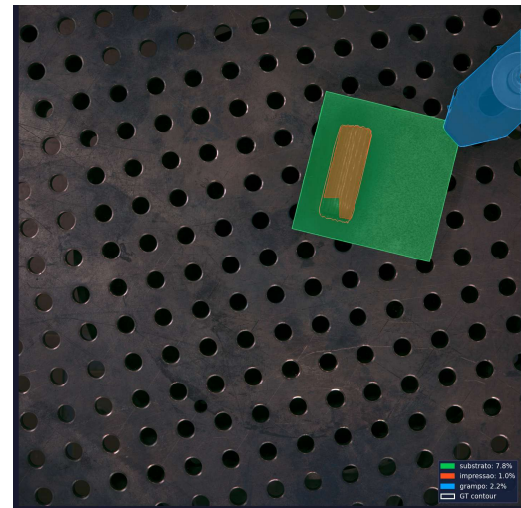
(a) Mask2Former mask1



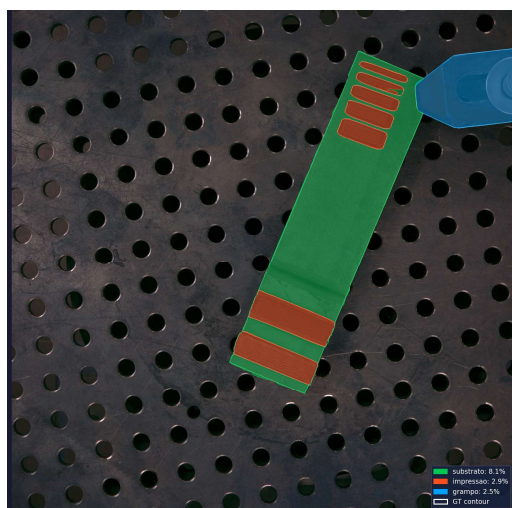
(b) HQ-SAM FT mask1



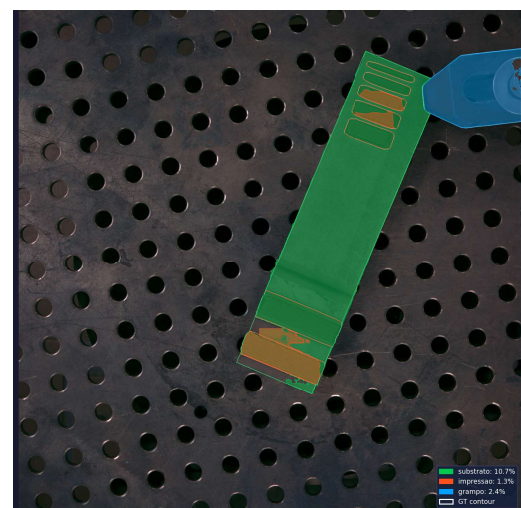
(c) Mask2Former mask2



(d) HQ-SAM FT mask2



(e) Mask2Former mask3



(f) HQ-SAM FT mask3

Figure 3.12: Inference comparison between Mask2Former and HQ-SAM

The qualitative results confirm this reading. The predicted masks are not perfect, but they are visually accurate and their fidelity is sufficient for the nesting scenarios targeted in this work. A recurring weakness is visible in the fine-tuned HQ-SAM predictions for the prints, where the contours are less reliable, a shortcoming that could compromise the precision of the printing process if left unaddressed.

It is also critical to note that the HQ-SAM architecture requires a secondary support model, YOLO, which increases the overall computational overhead and introduces additional points of failure. A suboptimal output from the YOLO detector directly compromises the downstream HQ-SAM segmentation mask. Conversely, the panoptic capabilities of the Mask2Former architecture streamline the pipeline by reducing the requirement for extensive post-processing to resolve overlapping predictions.

Synthesizing the evaluations presented in this chapter, Mask2Former was selected for integration into the final system. This decision is based on its superior consistency and precision in predicting the "*impressão*" and "*grampo*" classes. Regarding the "*substrato*" class, although Mask2Former yielded lower quantitative metrics than the fine-tuned HQ-SAM model, this performance gap is primarily attributed to Mask2Former's lower output mask resolution. This resulted in coarser, more aliased contours compared to HQ-SAM, which utilizes a specialized high-quality refinement token to generate significantly sharper and more detailed boundaries. However, these marginal discrepancies in contour fidelity are effectively encompassed by the safety buffers implemented within the downstream nesting system. Consequently, Mask2Former remains the superior selection, as its overall architectural consistency outweighs the minor geometric aliasing observed in substrate detection.

The primary bottleneck in substrate segmentation is the low textural contrast between the substrate and the worktable. Given that architectures relying exclusively on RGB data are unlikely to overcome these contrast-based limitations, a multimodal RGB-D approach was developed to improve edge detection performance. Since the current imaging setup is strictly two-dimensional, a 3D structured light acquisition pipeline was implemented to generate depth maps, as detailed in Chapter 3.8.

3.8 Depth Map Generation

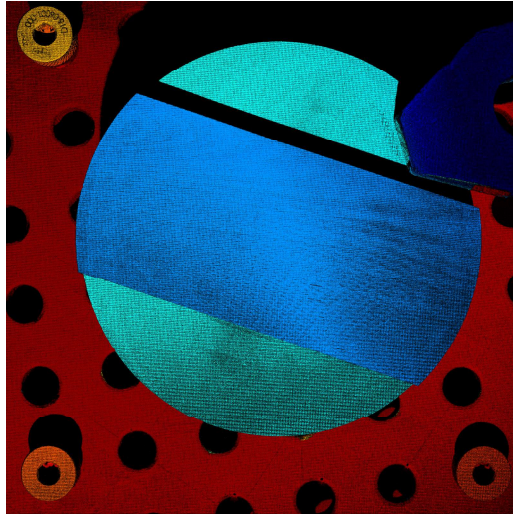
Following the literature review discussed in Section 2.4, a structured light methodology was adopted to extract depth maps of the printing table. The acquisition pipeline is based on a binary Gray code projection sequence, which provides a robust spatial correspondence map between the camera sensor and the projector resolution.

The Gray code sequence is generated using OpenCV's `structured_light_GrayCodePattern` module, producing 44 patterns for a 1920×1080 projector resolution. These patterns encode the horizontal and vertical projector coordinates in binary Gray code, where each bit is represented by a pair of complementary patterns, a pattern and its inverse, to improve robustness against ambient light variation and surface reflectivity. Prior to the Gray code sequence, a uniform white and a uniform black frame are captured to establish per-pixel contrast references, enabling normalization of the projected intensity and compensation for the non-uniform gain of the camera sensor and the varying albedo of the metallic substrate.

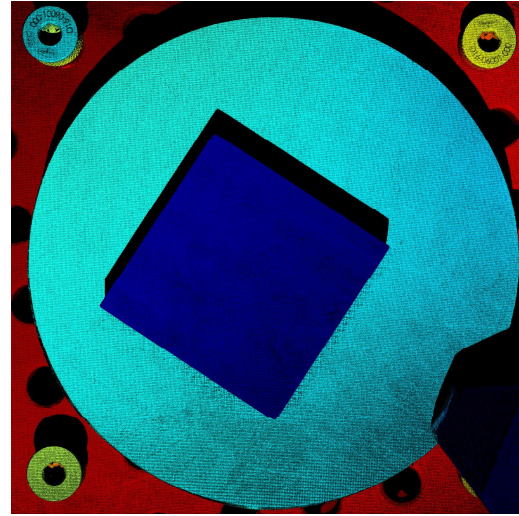
Decoding proceeds pixel-by-pixel: for each camera pixel, the captured intensities of each pattern pair are compared to determine the corresponding binary bit, and the resulting Gray code word is converted to a binary integer representing the projector column and row coordinate. Pixels where the local white-black contrast falls below a minimum threshold, or where the pattern and its inverse differ by less than a reliability margin, are flagged as invalid and excluded from the reconstruction. The output of the decoding stage is a dense correspondence map associating each valid camera pixel with its corresponding projector pixel coordinate.

Triangulation is performed using the calibrated stereo geometry between the camera and the projector. The camera is treated as the reference frame, and the projector is modelled as an inverse camera. Using the calibrated intrinsic parameters of both devices and the extrinsic stereo transformation obtained during calibration, each camera-projector pixel correspondence is triangulated via `cv2.triangulatePoints`, yielding a three-dimensional point in the camera coordinate frame. The resulting depth map is expressed in millimetres and subsequently normalized to a floating-point range of $[0, 1]$ using the known working distance range of the system, with invalid pixels assigned a value of zero.

Figure 3.13 presents four visual outputs of the structured light acquisition pipeline. The first two images show colourised depth maps of the printing table, where the colour gradient encodes metric depth and allows a qualitative assessment of the system's ability to discriminate height differences between the table surface, other part surfaces and the printing area. The remaining two images show point clouds reconstructed from the depth maps: the first obtained from a single camera pose, and the second resulting from the fusion of multiple poses captured at different table rotation angles. These representations are provided to allow a visual inspection of the spatial resolution, point density, and noise characteristics of the depth data produced under the operating conditions of the DED cell.



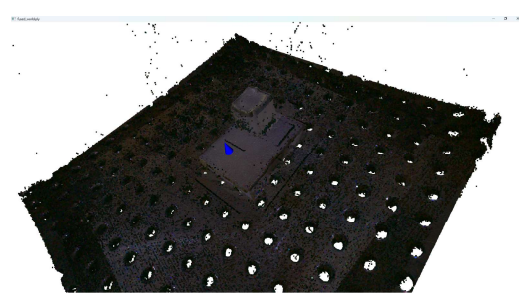
(a) Depth Map Example 1



(b) Depth Map Example 2



(c) Point Cloud from Single Pose



(d) Point Cloud with Multi-pose Fusion

Figure 3.13: Structured light acquisition outputs. (a,b) Colourised depth maps of the printing table under two substrate configurations. (c,d) Reconstructed point clouds from a single camera pose and from multi-pose fusion with station rotation, respectively.

3.9 RGB-D Segmentation

Based on the comprehensive literature review presented in section 2.5, two state-of-the-art RGB-D semantic segmentation models were selected for evaluation, namely CMX-B0 and DFormer-Tiny.

Training details

Both models were trained using a consistent framework to ensure a fair and rigorous comparative evaluation on the custom dataset. The training pipeline was structured for a total duration of 80 epochs using the AdamW optimizer, incorporating an 8 epoch linear warmup phase followed by a cosine annealing learning rate scheduler to stabilize early gradient updates. A multi-layer learning rate strategy was adopted to balance feature preservation and network adaptation, assigning a

lower learning rate of 1×10^{-5} to the pretrained backbones to retain robust generalized representations, while the segmentation heads were configured with a higher learning rate of 1×10^{-4} to rapidly adjust to the specific target classes of the dataset. To resolve class imbalances and guarantee high geometric precision around object boundaries, a joint objective function was optimized, combining a Boundary-Weighted Cross-Entropy loss and a Lovász-Softmax loss, expressed as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE_weighted}} + 0.3 \cdot \mathcal{L}_{\text{Lovász}} \quad (3.8)$$

The first component uses a Laplacian filter to locate class boundaries in the ground-truth labels and applies a maximum penalty weight of 2.0 to pixels near these edges, forcing the network to minimize boundary blurriness, while the Lovász-Softmax loss directly optimizes the intersection-over-union metric at the continuous logit level. Furthermore, an additional online data augmentation strategy was integrated directly into the training pipeline to further mitigate overfitting and enhance model robustness under varying industrial conditions. This online phase applied random horizontal and vertical flips, along with random 90 degree rotations, uniformly across both RGB and depth modalities to enforce geometric invariance. Crucially, color jittering was applied exclusively to the RGB channels to protect the physical meaning of the depth data. It is important to emphasize that these on-the-fly transformations function as an additional layer of regularization, complementing the baseline offline augmentations already established during the initial dataset preparation phase presented in section 3.6.

The models were monitored throughout the 80 epochs using key performance indicators including mean Intersection over Union (**mIoU**), Boundary F-score (**BF**) evaluated with a 3-pixel tolerance window, training loss and validation loss. The peak metrics achieved during the validation phase indicate that CMX-B0 reached a peak validation mIoU of 0.964 and a Boundary F-score of 0.949, whereas DFormer-Tiny converged at a lower performance level, achieving a peak validation mIoU of 0.936 and a Boundary F-score of 0.915, as summarized in Table 3.13.

Table 3.13: Peak validation metrics achieved by the trained architectures.

Architecture	Peak Validation mIoU	Peak Boundary F-score (BF@3px)
DFormer-Tiny	0.936	0.915
CMX-B0	0.964	0.949

The validation mIoU trajectories of both architectures over the 80 training epochs are presented in Figure 3.14. CMX-B0 consistently outperforms DFormer-Tiny across the entire training run, demonstrating rapid initial learning by surpassing an mIoU of 0.80 within the first 10 epochs, while DFormer-Tiny exhibits a smoother but slower upward trajectory that ultimately converges at a lower asymptote.

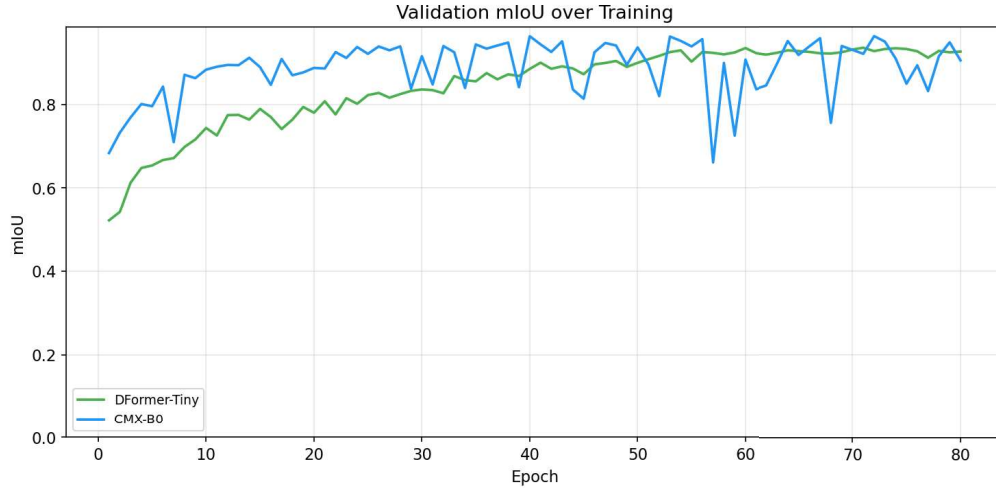


Figure 3.14: Validation mIoU evolution over 80 training epochs for DFormer-Tiny and CMX-B0. Dashed horizontal lines indicate the peak mIoU reached by each architecture.

A clear behavioral asymmetry is observed between the two architectures regarding the stability of the validation metric. While DFormer-Tiny displays a near-monotonic and smooth convergence curve, CMX-B0 exhibits noticeably higher epoch-to-epoch variance, particularly during the second half of training, with several transient single-epoch dips followed by an immediate recovery in the subsequent epoch. This behavior is attributable to two interacting factors inherent to the architectural design and training configuration rather than to training divergence. First, the high-resolution 512×512 input patches required to preserve fine substrate boundaries impose a memory-constrained batch size of four, which inherently increases the variance of the running statistics computed by the Batch Normalization layers present in the depth encoder and in the cross-modal fusion blocks; this variance propagates through the network and produces a higher dispersion of the validation metric across consecutive epochs. Second, the cross-modal feature rectification mechanism that defines the CMX architecture relies on channel-wise sigmoid gates that reweight each modality based on global descriptors of the complementary modality; these gates are intrinsically more sensitive to small shifts in feature distribution than the interaction-based attention employed by DFormer, which translates into the localized dips visible in Figure 3.14. Importantly, the upper envelope of the CMX-B0 curve is stable from approximately epoch 30 onward and the transient dips are systematically recovered in the following epoch, confirming that the oscillations occur around a converged operating point and do not indicate model divergence or instability of the optimization process.

Test Set Evaluation

Given the validation mIoU fluctuations observed for CMX-B0 and the recognized risk of optimistic bias when reporting peak validation metrics, an independent held-out test set was employed

to obtain an unbiased estimate of the generalization performance of both architectures and to verify the absence of overfitting. The test set was constructed from images that were never seen during training or validation, preserving the same acquisition conditions and class distribution of the original dataset. For each model, the checkpoint corresponding to the best validation mIoU was loaded and evaluated on the test set using three complementary metrics: Intersection over Union (IoU), DICE score, and Boundary F-score with a 3-pixel tolerance (BF@3px). The evaluation was restricted to the *PrintingArea* class, which is the only class that directly affects the offset calculation in the repair pipeline. The background class is uninformative for this purpose and the *grampo* class, although useful as a contextual cue during training, is not required as an output of the segmentation stage.

A qualitative comparison of the predictions produced by both models on two representative test samples is presented in Figure 3.15. The first sample (top row) shows a geometrically complex L-shaped substrate placed on the perforated table next to a fixturing clamp, while the second sample (bottom row) shows a simpler rectangular substrate with multiple bolt-hole features. In both scenes, CMX-B0 produces a noticeably tighter contour around the substrate, closely tracking the rounded corners and the small notches, whereas DFormer-Tiny exhibits noisy boundary artefacts and small spurious detections along the substrate edges, particularly visible in the right-hand side of both images where the model leaks the *grampo* mask into the surrounding background.

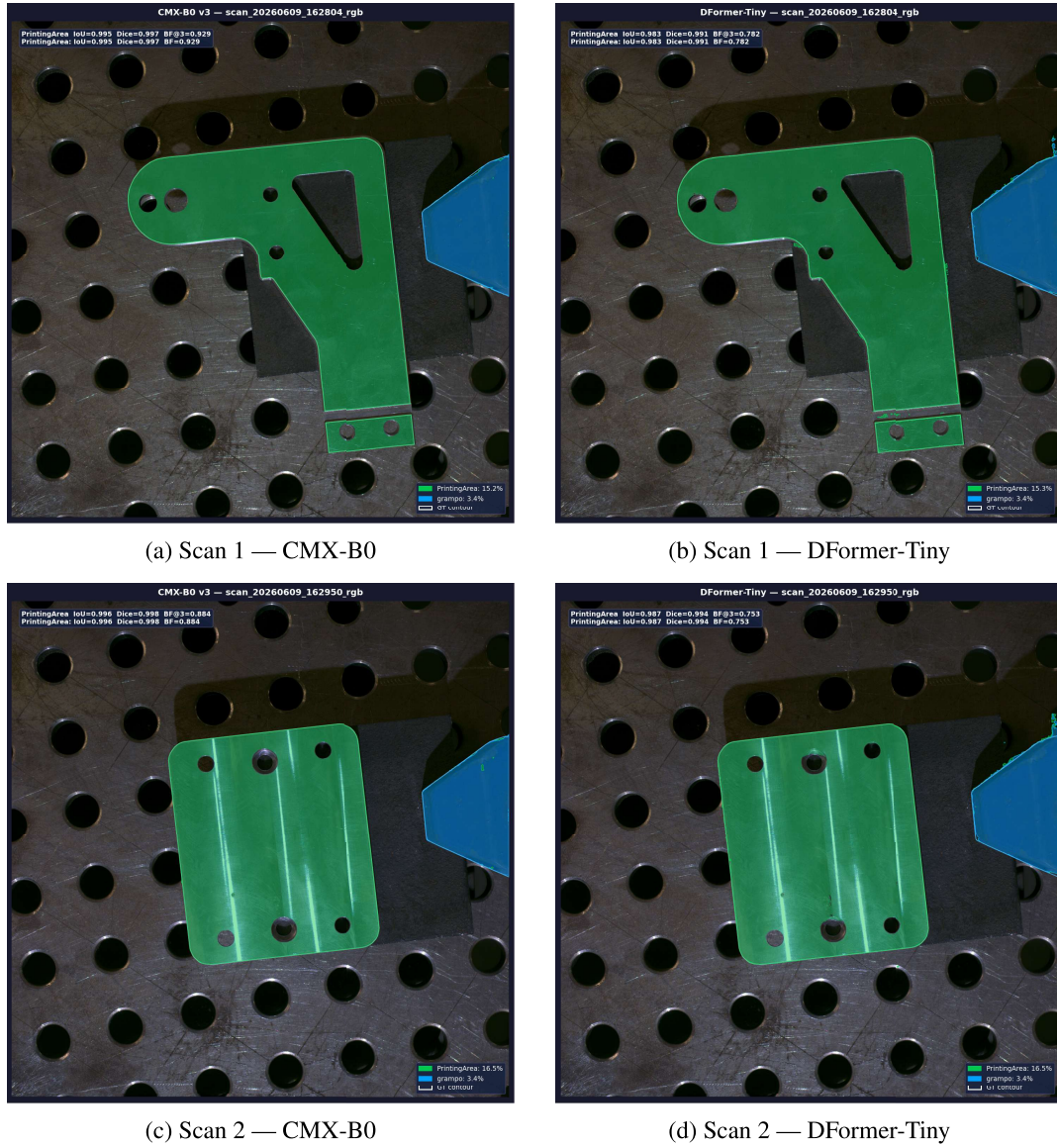


Figure 3.15: Qualitative comparison of segmentation predictions produced by CMX-B0 and DFormer-Tiny on two representative samples of the held-out test set. The green overlay corresponds to the predicted *PrintingArea* mask, the blue overlay to the predicted *grampo*, and the thin white contour to the ground-truth boundary.

Figure 3.16 reports the IoU obtained by both architectures for the *PrintingArea* class on the test set. The IoU directly measures the overlap between predicted and ground-truth segmentation masks and is the primary global accuracy indicator for semantic segmentation tasks.

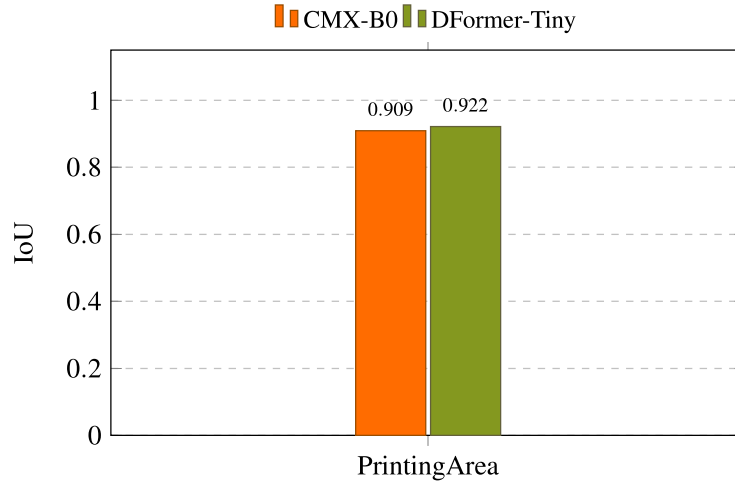


Figure 3.16: Intersection-over-Union (IoU) for the *PrintingArea* class achieved by both architectures on the held-out test set.

Figure 3.17 reports the Dice score for the *PrintingArea* class. The Dice coefficient is defined in Equation (3.9) as:

$$\text{Dice} = \frac{2|P \cap G|}{|P| + |G|}, \quad (3.9)$$

where P and G denote the predicted and ground-truth pixel sets. The DICE score is closely related to the DICE loss minimized during training through the relation $\mathcal{L}_{\text{Dice}} = 1 - \text{DICE}$ and provides a complementary view of overlap that places relatively more weight on the agreement between predicted and ground-truth foreground regions.

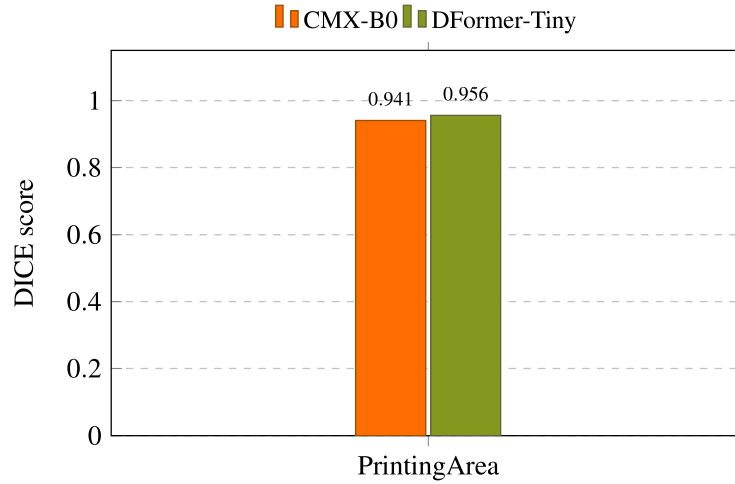


Figure 3.17: DICE score for the *PrintingArea* class achieved by both architectures on the held-out test set.

Finally, Figure 3.18 reports the Boundary F-score with a 3-pixel tolerance (BF@3px), which specifically quantifies the geometric accuracy of the predicted class boundaries and is the most relevant metric for the downstream contour-driven nesting pipeline described in Section 3.6.

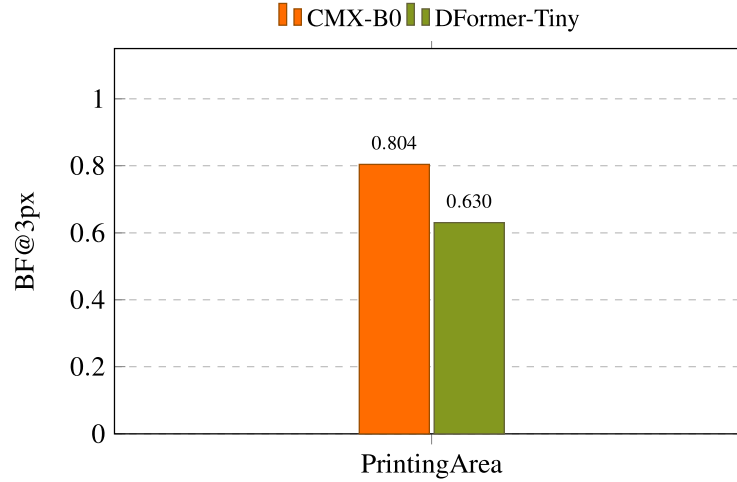


Figure 3.18: Boundary F-score with 3-pixel tolerance (BF@3px) for the *PrintingArea* class achieved by both architectures on the held-out test set.

The aggregated test set results expose a clear trade-off between the two architectures. DFormer-Tiny marginally outperforms CMX-B0 on area-based metrics, with an IoU advantage of +0.013 (0.922 versus 0.909) and a DICE score advantage of +0.016 (0.956 versus 0.941). However, CMX-B0 substantially outperforms DFormer-Tiny on the BF@3px metric, achieving 0.804 against 0.630, a margin of +0.174 in absolute terms. Since the downstream pipeline operates exclusively on the extracted contour of the segmented *PrintingArea*, geometric boundary fidelity is the decisive criterion. For this reason, CMX-B0 was selected as the segmentation backbone for the downstream contour extraction of the proposed repair pipeline. It should nonetheless be acknowledged that the current test set, while sufficient to compare relative architectural behaviour, does not fully reflect the geometric distribution of substrates encountered in the repair workflow; both models should therefore be retested on a dataset whose substrate poses, scales and surface conditions are deliberately offset to match the actual repair scenarios before either is committed to production use.

A further evaluation that was not included in this comparison is a pure RGB segmentation model retrained on the same dataset. Although the comparison between RGB-only and RGB-D models would in principle be informative, the current dataset, which relies on overlapping objects to demonstrate the contribution of the depth channel, is not the appropriate vehicle for it. The overlapping configuration is well suited to illustrate how depth resolves vertical ambiguity between objects of distinct geometry, but it does not reproduce the more challenging failure mode of a repair workflow, in which the substrate to be repaired and a neighbouring reference surface may share

an almost identical visual appearance and differ only in their depth profile. In such a scenario the RGB-only model is expected to be at a significant disadvantage, while in the present dataset the same model would likely perform competitively because the visual cues alone are sufficient to separate the classes. Conducting the RGB-only comparison on a dedicated dataset that explicitly contains identically-textured surfaces, something that is typically found in real repair scenarios is a necessary step for a fair and conclusive assessment of how much the depth would increase the before mentioned metrics.

3.10 Nesting Algorithm

With the print path polygon obtained from the *RAPID* file and the substrate and obstacle contours projected into the robot coordinate frame, the nesting problem can be formulated: find a collision-free placement for the print path inside the substrate boundary, maintaining a minimum clearance from both the substrate edges and any previously deposited material, whilst maximizing the use of the available substrate area.

Testing Nesting Algorithms

To select the nesting algorithm that best fits the requirements of this project, a comparative evaluation was performed assessing three metrics: computation time, placement precision, and overall substrate utilization. Four methods were considered: the geometry-based NFP and the Bottom-Left Fill (BLF) heuristic (a raster based method), both described in Chapter 2.6, a random-based method consisting of generating 5000 random candidate placements and selecting the one closest to the nearest corner, and manual placement included as a human baseline. Two tests were conducted. The first comprised 9 scenarios in which each method was used to nest three print geometries: square, triangle, and circle onto three substrate geometries: square, rectangle, and circle, each containing two pre-placed obstacles representing previous prints at randomised positions. Figure 3.19 illustrates an example of a test scenario while Table A.1, in Appendix A, and table 3.14 present the individual scenario results and the summarized aggregate statistics respectively.

Table 3.14: Nesting Method Benchmark — Summary Statistics Across All Scenarios

Method	Computation Time (s)			Corner Distance (mm)		
	Average	Max	Min	Average	Max	Min
NFP	3.4	10.8	0.3	100.5	150.4	74.0
BLF	17.9	55.6	2.3	103.1	151.4	74.4
Random	0.8	2.1	0.3	105.0	151.5	78.6
Manual	21.9	30.9	10.8	108.5	194.2	75.4

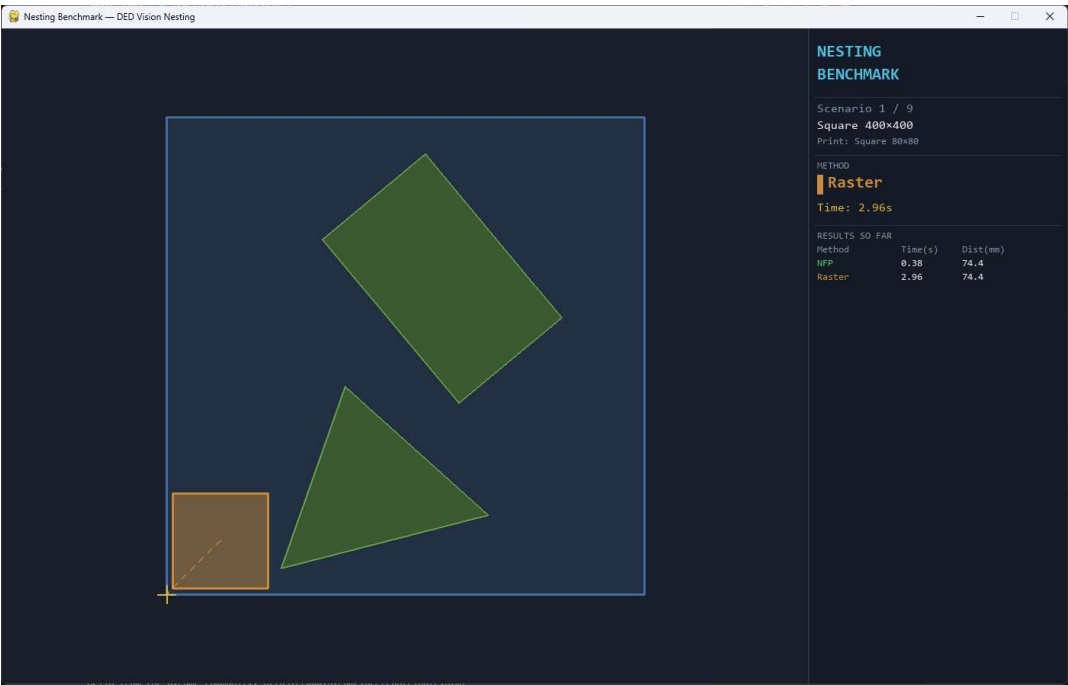


Figure 3.19: Example of nesting scenario

From these results it can be concluded that NFP is the best-performing method with respect to placement quality, consistently yielding the lowest average corner distance across all scenarios. The random method achieved the shortest computation times, remaining consistently below 2.1 s in all cases. The BLF method was slower than both NFP and Random, primarily due to the computational time associated with circular print geometries: the absence of vertices requires the evaluation of a significantly larger number of candidate positions, as evidenced by the computation times of 40–56 s in scenarios 3, 6, and 9. Despite this limitation, BLF produced a placement quality close to NFP across the remaining scenarios.

A second test was conducted to evaluate the capacity of each method to maximise substrate use. In this test, shapes were sequentially nested onto a single substrate until no further placements were possible, after which the percentage of occupied area was recorded. Figure 3.20 presents the results of this test. NFP achieved the highest substrate use and was therefore selected as the nesting algorithm for this project.

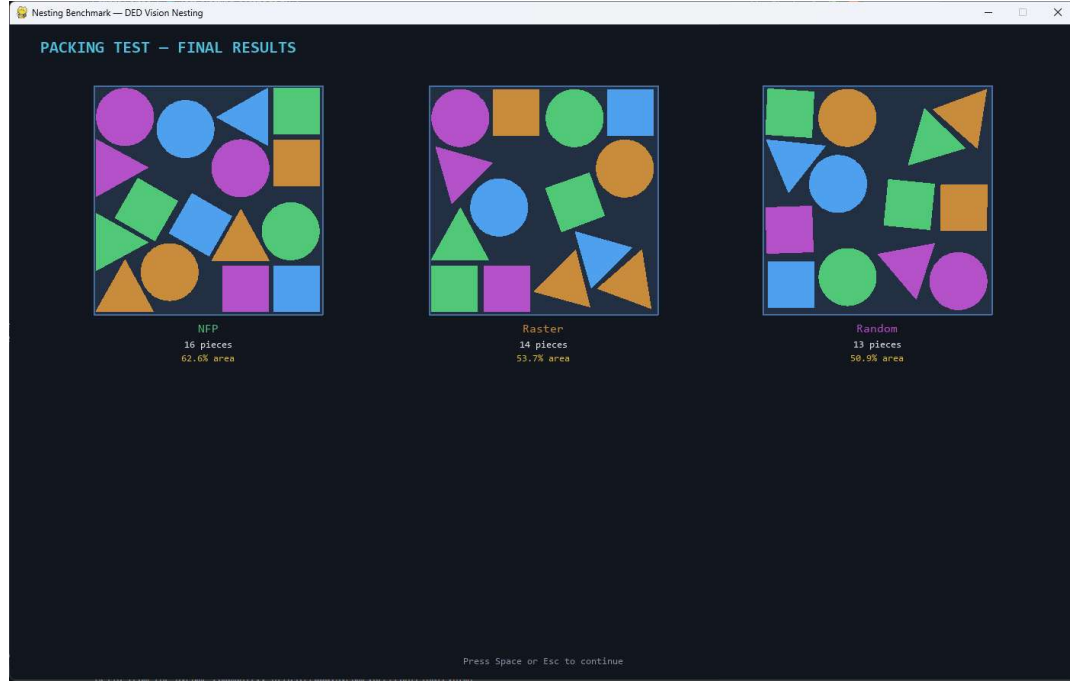


Figure 3.20: Sequential packing test results: percentage of substrate area occupied by each nesting method until no further placements were possible.

No-Fit Polygon Nesting

The placement algorithm is based on the NFP method. The NFP answers a precise geometric question: given a piece P and an obstacle O , what is the complete set of positions for P 's reference point such that P and O are touching but not overlapping. Any position of the reference point strictly inside the NFP causes overlap. Any position outside guarantees a gap. This formulation allows the valid placement region to be computed analytically rather than by exhaustive grid search.

The valid placement region $\mathcal{V}$ for a given rotation angle θ is defined as:

$$\mathcal{V}(\theta) = \text{IFP}(P_\theta, S) \setminus \bigcup_i \text{NFP}(P_\theta, O_i)$$

where P_θ is the print path convex hull rotated by θ , S is the substrate boundary eroded by the substrate clearance margin d_s , IFP is the Inner Fit Polygon, the set of centroid positions for which the piece lies entirely within S , and $\text{NFP}(P_\theta, O_i)$ is the forbidden centroid region induced by obstacle O_i expanded by the obstacle clearance margin d_o . Any point in $\mathcal{V}(\theta)$ is a geometrically valid, collision-free placement.

Inner Fit Polygon Computation

The IFP is computed via exact Minkowski erosion. For a piece with vertices $\mathbf{v}_1, \dots, \mathbf{v}_n$ expressed relative to its centroid, the IFP is the intersection of n translated copies of the eroded substrate:

$$\text{IFP} = \bigcap_{i=1}^n (S \oplus (-\mathbf{v}_i))$$

This is exact for any piece shape and avoids the over-erosion introduced by isotropic buffer methods, which assume the piece extends equally far in all directions and can waste significant substrate area for elongated or rectangular geometries.

No-Fit Polygon Computation

The NFP of the piece with respect to each obstacle is computed via the Minkowski sum $\text{NFP}(P, O) = O \oplus \text{reflect}(P)$, where $\text{reflect}(P)$ denotes the piece polygon reflected through the origin, mapping every vertex (x, y) to $(-x, -y)$. Rather than implementing the full edge-merging Minkowski algorithm, the sum is approximated by translating the obstacle O by each vertex offset of $\text{reflect}(P)$ and taking the convex hull of the resulting union as Bennell and Song [22] showed. For convex piece hulls this approximation is exact; for mildly non-convex geometries it is slightly conservative, expanding the forbidden zone marginally beyond the true boundary, which is the safe direction for collision avoidance.

Placement Scoring and Rotation

The algorithm evaluates a discrete set of rotation angles $\theta \in \{0, \pm 5, \pm 10, \pm 15, \pm 20, \pm 30\}$, testing $\theta = 0$ first so that an unrotated solution is preferred whenever one exists, minimising unnecessary reorientation of the print path. For each angle, $N = 120$ candidate centroid positions are sampled uniformly along the boundary of $\mathcal{V}(\theta)$. Boundary positions are preferred over interior ones because they represent the tightest possible fit, placing the centroid in the interior would leave recoverable substrate area unused. Each candidate is scored by its minimum Euclidean distance to the four corners of the substrate bounding box, with the bottom-left corner weighted most favourably. This corner heuristic is standard in industrial nesting literature and tends to compress successive pieces toward the substrate origin, maximising the contiguous free region available for subsequent placements. The candidate with the lowest score that passes a final hard containment and collision check is selected as the optimal placement.

Clearance Margins

Three independent clearance margins are exposed as operator-configurable parameters at runtime. The substrate margin d_s defines the minimum distance the print path must maintain from the substrate boundary, protecting the substrate edges from deposition. The print margin d_o defines the minimum separation between the new print path and any previously deposited material. A third

margin for the mechanical clamp, following the same geometric logic, is planned as a straightforward extension of the existing obstacle avoidance nozzle collision. All three margins are applied during the NFP and IFP computations described above, so no separate collision pass is required.

Manual Position Correction

To ensure robustness in the presence of detection uncertainties or substrate irregularities, the operator retains full control over the final placement. After the automatic nesting solution is computed and displayed in the User Interface (UI), fine-adjustment controls allow the operator to apply incremental translations Δx , Δy and a rotation $\Delta\theta$ to the positioned print path. These adjustments are applied sequentially, rotation first around the current centroid, followed by translation, and the updated path is immediately re-rendered in the nesting preview, allowing the operator to visually verify the final position before committing the result to the robot's programme.

3.11 Repair Toolpath Alignment

Once the surface to be repaired has been segmented and the original toolpath parsed from the RAPID module, the two must be brought into the same placement before the robot can execute the repair. This section describes the rigid correction $(\Delta x, \Delta y, \theta)$ computed for that purpose, using the same work-object update mechanism designed for the nesting pipeline.

The problem reduces to finding the translation and rotation that map the parsed toolpath, called the *source* contour, onto the detected surface, called the *target* contour. Both are closed planar shapes described by a list of points in world coordinates (mm), and they describe the same physical region, they differ only by the small displacement and rotation introduced when the part was placed manually on the table. The alignment is therefore decomposed into two independent steps: matching their centers and matching their orientations.

The first step is to find where the centre of each shape lies, so that the source can be slid onto the target. A naive approach would be to average the coordinates of the contour's points, but this is misleading whenever the points are not evenly spaced along the boundary, which is the typical case here: the parsed toolpath is denser on curves and sparser on long straight segments, while the segmented contour follows a roughly uniform pixel spacing. To avoid this bias, the centre is computed from the *image moments* of the filled shape using `cv2.moments`. Intuitively, the moments treat the shape as a flat plate of uniform density and return the position of its centre of mass, which is independent of how the boundary was sampled. Formally, for a polygon with moments M_{ij} , the centroid is

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}}, \quad (3.10)$$

where M_{00} is the enclosed area and M_{10} , M_{01} are its first-order moments. The translation $(\Delta x, \Delta y)$ is then simply the vector connecting the source centroid to the target centroid.

The second step is to find how much the source must be rotated to match the target's orientation. To make this measurable, each contour is wrapped in its minimum-area bounding rectangle using `cv2.minAreaRect`, and the angle of that rectangle is taken as the orientation of the shape. This method is robust even for sparse contours and insensitive to how the boundary was sampled. A first estimate of the rotation is given by the difference between the two rectangle angles. However, a rectangle looks identical to itself when rotated by 90° , so this estimate is ambiguous: four different angles, spaced 90° apart, are equally valid from the rectangle's point of view, and only one of them actually aligns the shapes. The ambiguity is resolved by trying all four candidates,

$$\theta_k = \theta_0 + k \frac{\pi}{2}, \quad k \in \{-1, 0, 1, 2\}, \quad (3.11)$$

and, for each one, measuring how well the rotated source overlaps the target. The measure of overlap is the mean distance from each transformed source point to its nearest target point,

$$s(\theta) = \frac{1}{N} \sum_{i=1}^N \min_j \|\mathbf{p}_i^{\text{src}, \theta} - \mathbf{p}_j^{\text{tgt}}\|, \quad (3.12)$$

which is small when the two contours sit on top of each other and large when they do not. The candidate with the smallest $s(\theta)$ is taken as the final rotation θ . Together with the translation from the previous step, this produces the rigid correction $(\Delta x, \Delta y, \theta)$, which is returned along with the residual distance between the centroids after alignment as a sanity check.

This formulation was deliberately kept lightweight: it requires no point-to-point correspondences between the contours, has a closed-form solution for both the translation and the rotation up to the four-way ambiguity, and runs in milliseconds even for contours with thousands of points.

The procedure was validated qualitatively on two independent simulations. In each, a previously segmented `PrintingArea` contour was paired with the toolpath parsed from the corresponding RAPID file, deliberately displaced and rotated to mimic an arbitrary manual placement of the part on the table. Figure 3.21 shows, for both cases, the toolpath before and after applying the computed correction.

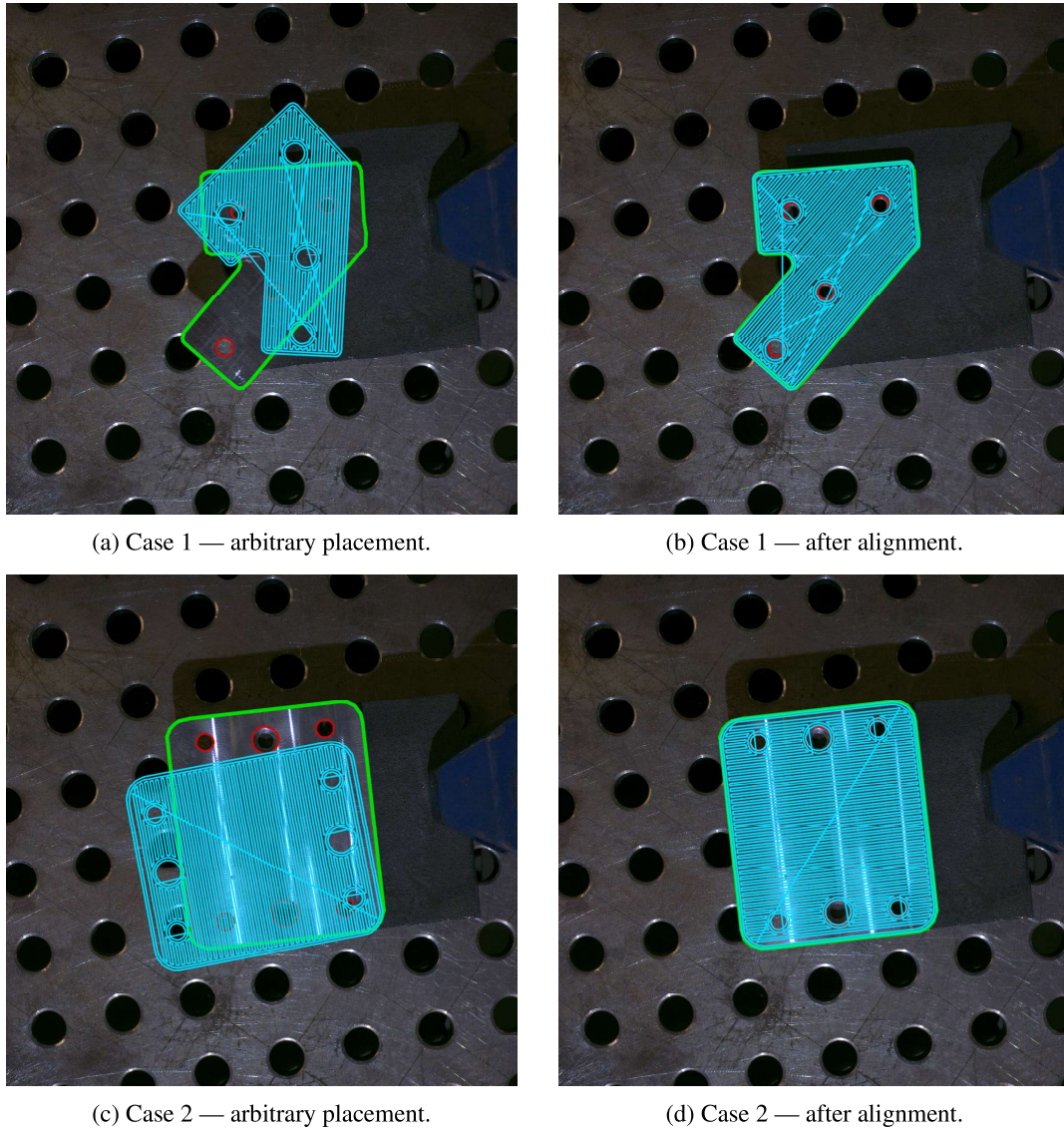


Figure 3.21: Repair toolpath alignment on two simulated cases. The segmented `PrintingArea` contour (green + red) is shown together with the parsed toolpath (blue), first placed arbitrarily (a, c) and then after applying the computed $(\Delta x, \Delta y, \theta)$ correction (b, d).

3.12 App-Robot Communication

Upon completion of the nesting operation, the spatial offset determined for the new print path must be applied to the work object (*wobj*) defined in the robot controller, to which all coordinates in the *RAPID* file are referenced. To establish this communication channel, modifications were introduced to the post-processor of the slicing software where print jobs are previously prepared.

At the beginning of a print job, the operator is prompted, through the teach pendant, to indicate whether the developed system should be engaged. Upon confirmation, a secondary prompt requests authorization to move the robot to the predefined image capture position. Once the robot reaches this position, the operator is instructed to launch the developed software and ensure that the complete nesting or repair verification pipeline is correctly executed. Upon pipeline completion, the computed offset is transmitted from the host computer to the robot controller in the format (X, Y, θ) , where X and Y denote the translational corrections along the respective axes and θ the rotational correction to be applied.

Communication between the host computer and the robot controller is implemented via TCP/IP sockets, using the *RAPID* built-in functions `SocketCreate`, `SocketConnect`, `SocketReceive` and `SocketClose`, whose specifications are documented in the ABB *RAPID* reference manual- ABB Robotics [36]. On the software side, the standard Python `socket` library Python Software Foundation [37] was employed to manage the connection and transmit the offset data.

Chapter 4

System Validation and Performance Evaluation

This chapter details the operational workflows for the three developed systems: the calibration pipeline, the nesting system, and the repair system. The calibration pipeline underpins both operational systems and is responsible for establishing the geometric relationships between the camera, the robot end-effector, and the workspace. It supports multiple modes designed to minimise recalibration overhead while maintaining the accuracy requirements of each task. The primary objective of the nesting system is to minimise setup latency through a robust framework that achieves a placement accuracy of ± 2 mm while simultaneously optimising substrate utilization. The repair system employs a more sophisticated multi-stage pipeline, leveraging depth map reconstruction and an RGB-D foundation model to generate high-fidelity surface contours, ensuring the precise alignment of new print trajectories with the existing volumetric geometry of the repair zone, with a precision of ± 0.6 mm.

4.1 Calibration Pipeline

The calibration system is structured around four modes that cover all practical scenarios, from a quick intrinsic update after refocusing the lens to a complete end-to-end calibration including projector characterization. Separating the pipeline into modes ensures that only the necessary calibrations are run in each situation, avoiding unnecessary robot motion and downtime.

The four available calibration modes are as follows. **Mode 1 (Intrinsic only)** recalibrates the camera intrinsic matrix $\mathbf{K}$ and distortion coefficients $\mathbf{D}$ without modifying the hand-eye transform $\mathbf{T}_{he}$. The user selects one of two predefined paths: the lower path, at an average sensor-to-board distance of around 500 mm and used for smaller FOV tasks, or the upper path, at an average distance of around 960 mm for if a larger FOV is required. In both cases the robot visits 12 endpoint poses, the images are collected, and $\mathbf{K}$ is updated in the calibration file. This mode is intended for use after any lens refocusing operation and typically completes in under five minutes.

Mode 2 (Intrinsic and extrinsic) performs a full hand-eye calibration along the same two paths, but stopping at four interpolated intermediate poses between each consecutive endpoint pair, for a total of 56 poses. Before the robot run, the user is asked for the world-frame positions of six board corners measured by jogging the robot and touching them with the nozzle tip. After the poses are collected, intrinsics are computed, $AX=XB$ is solved across all four solver methods, and the simultaneous fixed-point refinement is applied. The best result by cross-pose consistency standard deviation is saved automatically. Because the time required to compute the intrinsics on top of the already-collected images is negligible (three to five seconds) compared to the total robot path time (ten to fifteen minutes), no separate extrinsic-only mode is provided: whenever the hand-eye transform is recomputed, the intrinsics are refreshed as well. This mode is the primary calibration mode and is expected to be run infrequently, only when the camera is physically disturbed, including big adjustments to the lens, or the system is installed in a new environment. The workflow of this mode is illustrated in Figure 4.1.

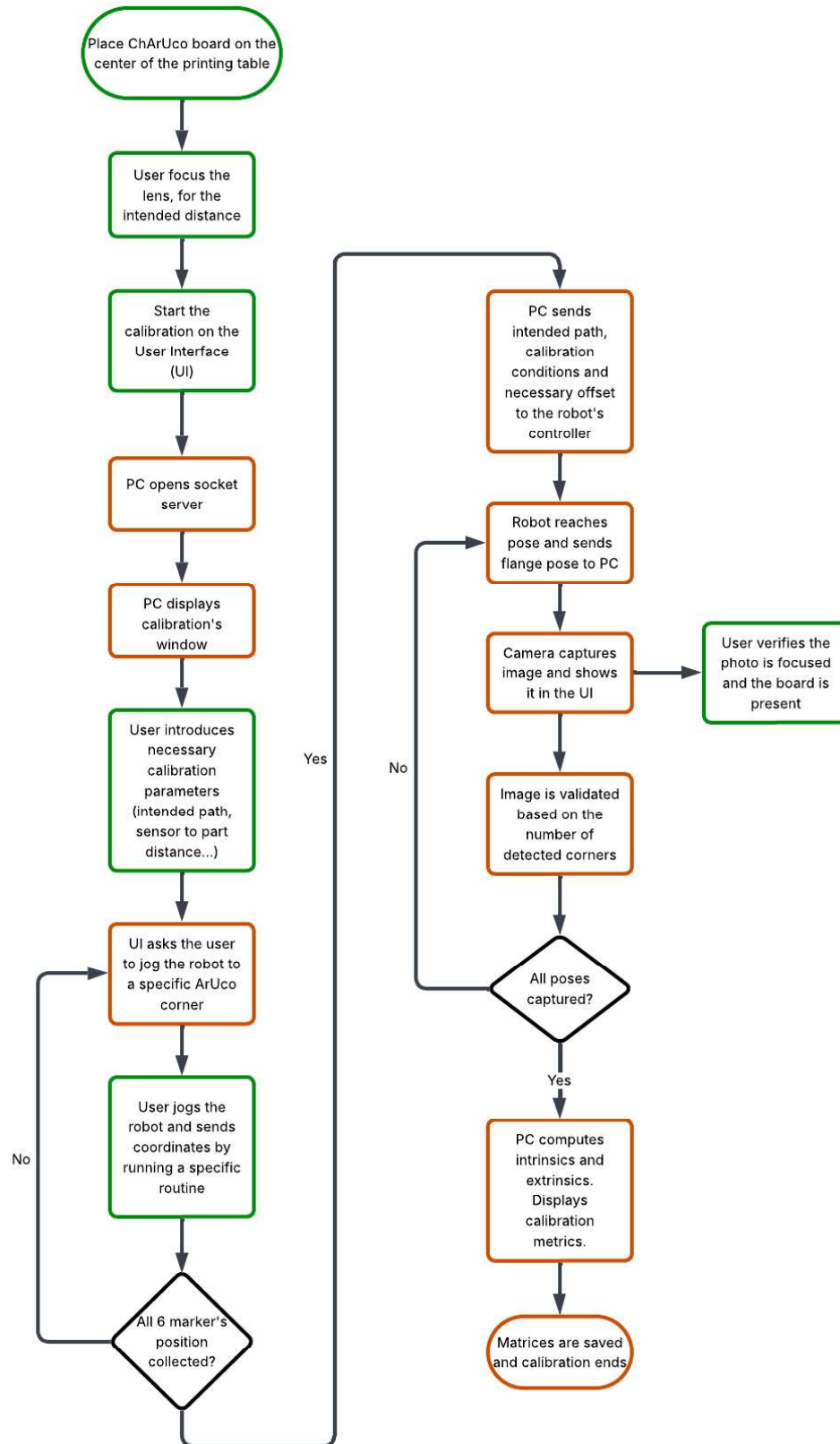


Figure 4.1: Workflow of the Mode 2 calibration (camera intrinsic and extrinsic). Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.

Mode 3 (Projector intrinsics) characterizes the projector intrinsic matrix and distortion coefficients. For this mode, the small ChArUco board is replaced by a larger A2 board placed on the table because the projector's field of projection is almost 2 m wide in the minimum-zoom configuration and a board of this size provides feature points spread further apart, which helps constrain the projector distortion model reliably. The robot either follows the previously defined path, or an offsetted one, where a translation in the z axis is performed to ensure the perfect sensor-target distance is achieved in order to maximize image sharpness. The resulting projector calibration is saved independently of the hand-eye file. The setup for this mode is shown in Figure 4.2.

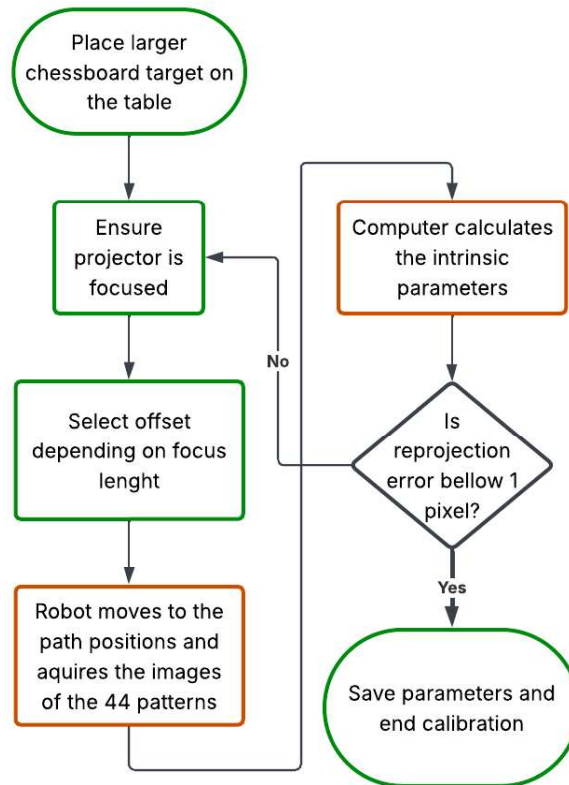


Figure 4.2: Projector's intrinsic calibration flowchart. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.

Mode 4 (Projector stereo) computes the stereo relationship between the camera and the projector, and requires Mode 3 to have been run beforehand so that the projector intrinsics are already available. Because a stereo calibration is valid only for a specific camera pose, the robot stays still throughout this mode and the operator manually repositions the small ChArUco board between captures. The operator must collect at least 10 board positions, taking care to vary not only the in-plane location but also the angle and inclination of the board, so that the stereo extrinsics are well constrained. The setup for this mode is shown in Figure 4.3.

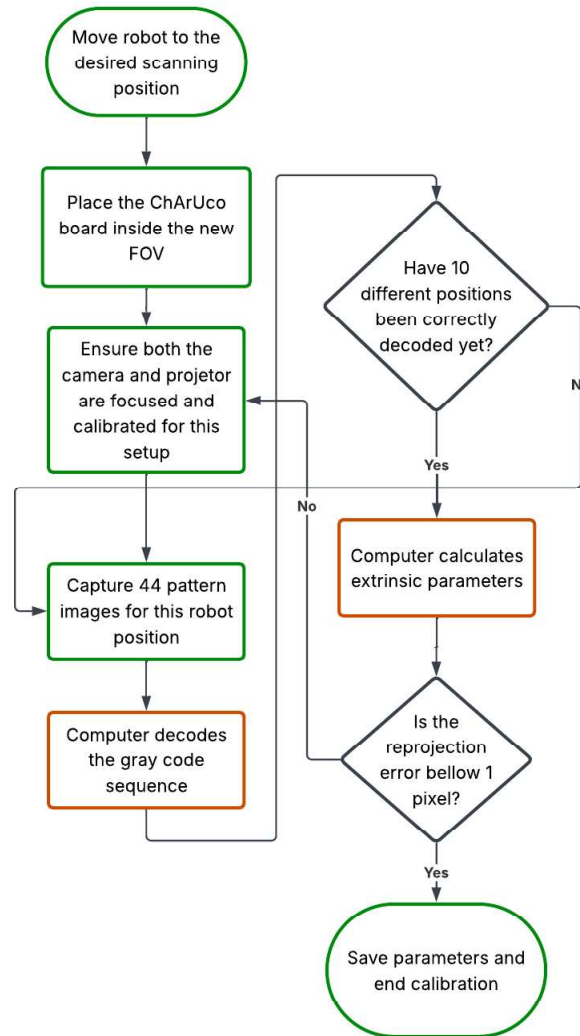


Figure 4.3: Projector’s extrinsic calibration flowchart. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.

Across all four modes, a working distance offset feature is available to accommodate parts of varying height, for which the nominal 500 and 960 mm paths may not be adequate. Whenever the sensor-to-part distance changes, the lens has to be refocused and the intrinsics recalibrated at the new distance. Rather than requiring a new set of joint poses to be recorded for every possible working height, the user can specify the target sensor-to-board distance and the height of the board surface in the robot coordinate frame. The script then automatically computes the required Z offset and applies it to each robot pose during data collection, ensuring the calibration board is always at the correct distance from the sensor without any manual pose re-recording.

Because repair jobs vary in both height and placement on the table, and because the alignment between the toolpath and the actual surface demands a very accurate positioning, the relevant calibration steps are not treated as a one-off setup but are instead embedded in the normal operating workflow, as described in Section 4.3. They can in principle be skipped when the exact same repair is to be performed twice in a row, but in practice this is rarely the case.

4.2 Nesting System Workflow

Figure 4.4 presents a comprehensive overview of the full system workflow, illustrating the sequential steps from the upload of the printing job to the robot controller through to the onset of the printing process. The diagram captures the two parallel threads of operation: the robotic sequence, in which the robot transitions from its home position to the image capture position, and the software pipeline, in which the image is acquired, contours are extracted, and the nesting optimization is performed. The computed offset is subsequently transmitted to the robot controller via TCP/IP, which corrects the work object accordingly, finalising the preparation sequence and allowing the printing process to commence. To better explain the communication nuances, Figure 4.5 displays the communication diagram of the before mentioned system.

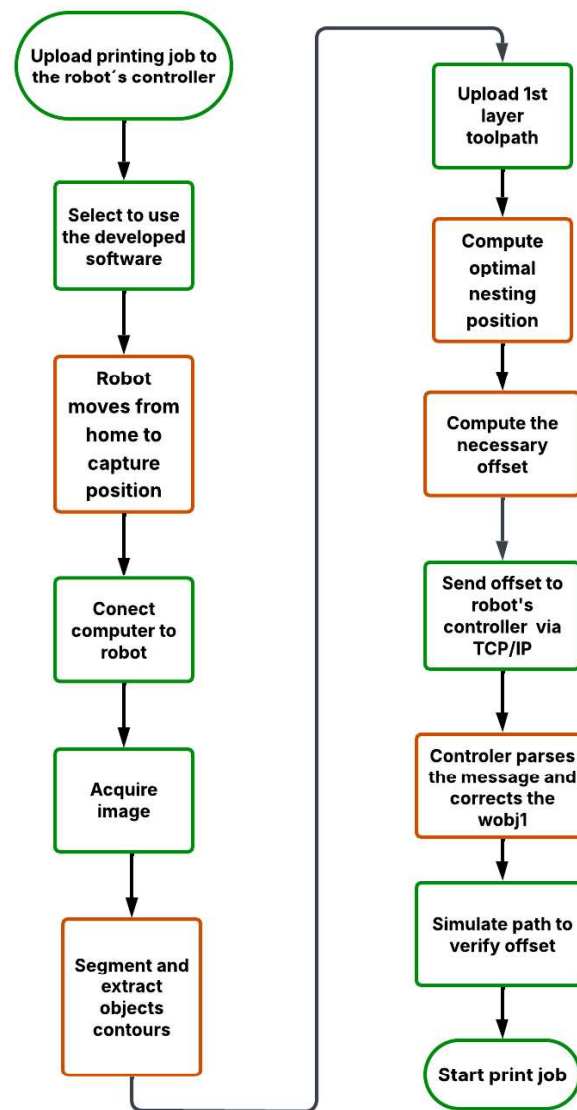


Figure 4.4: Overview of the full nesting system workflow. Green boxes denote operator-only manual steps and orange boxes represent autonomous system actions.

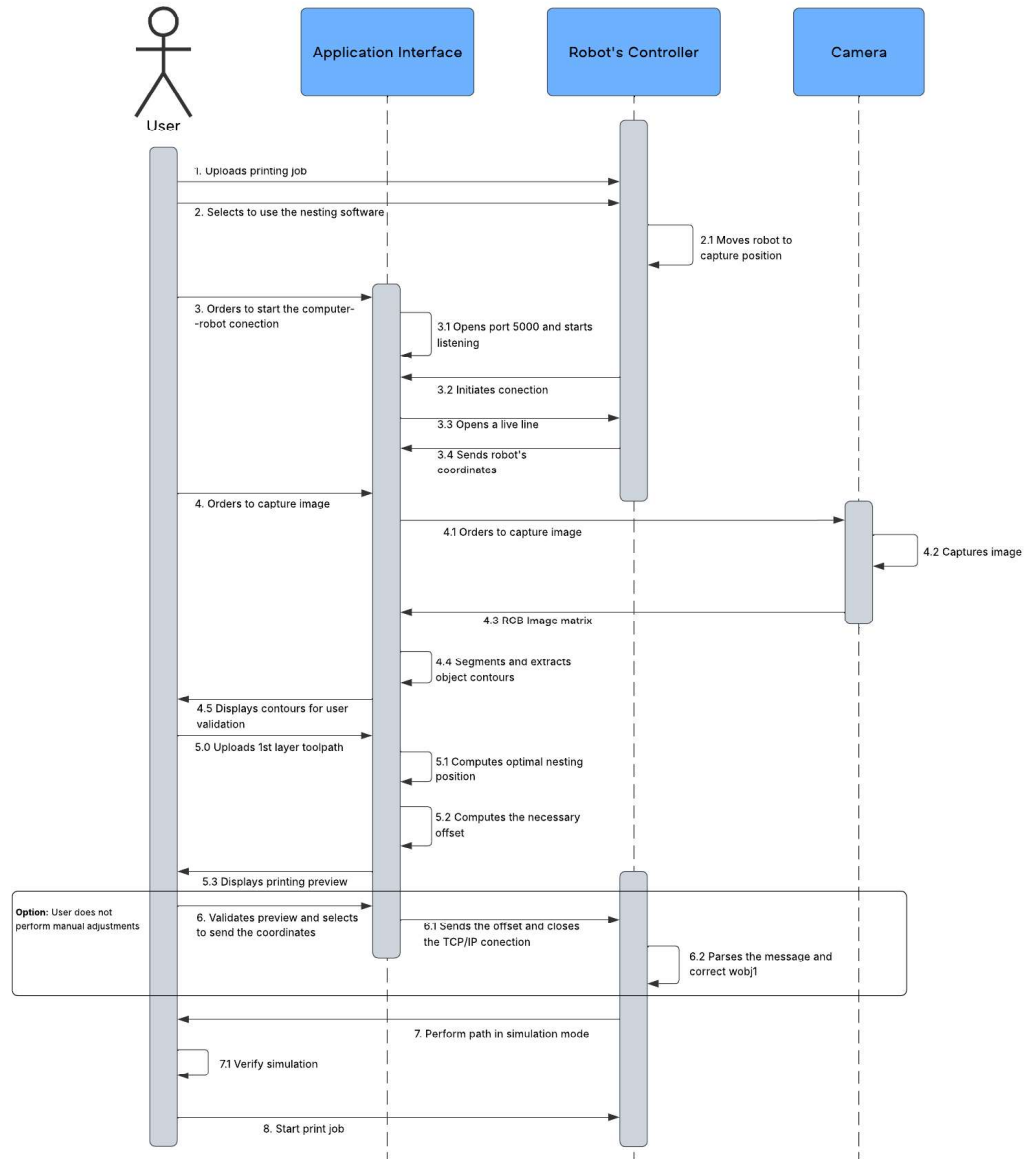


Figure 4.5: Overview of the full nesting system communication flow

Figure 4.6 displays multiple stages of the user's interface during a nesting job and the last step where the user simulates the corrected printing path to verify that the applied offset, correctly shifted the toolpath.

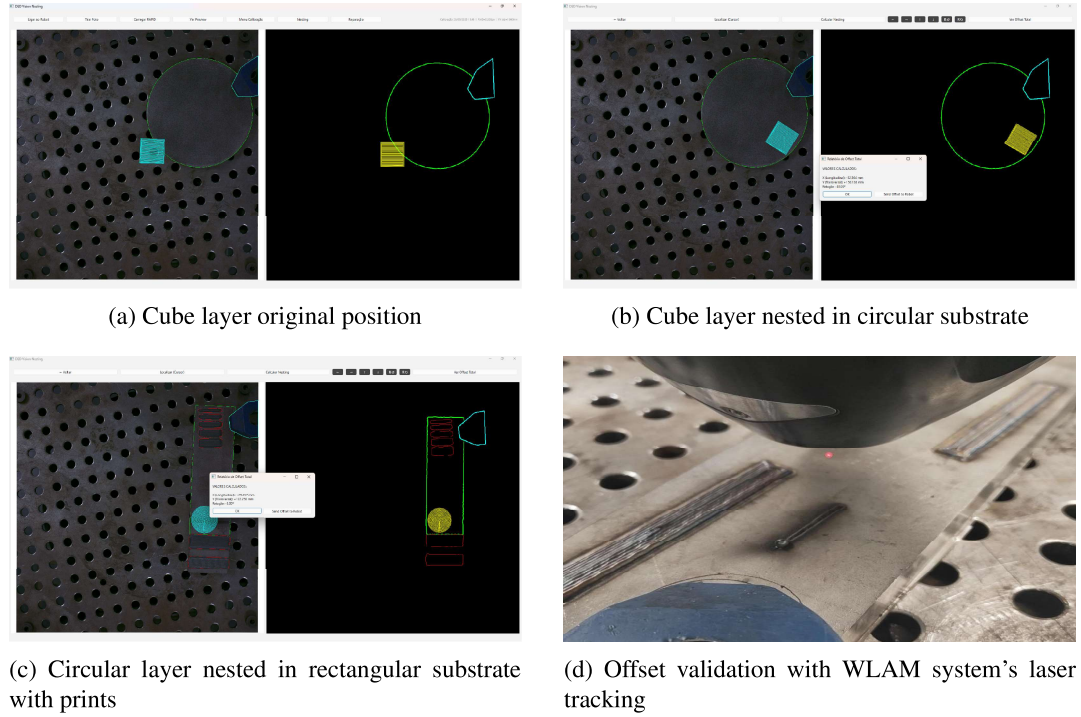


Figure 4.6: Nesting pipeline interface steps and laser validation

To validate the performance of the implemented system, a comparative analysis of setup times was conducted by nesting diverse toolpath geometries across multiple substrates of varying complexity. This experimental framework served to quantify the reduction in preparation latency and to evaluate the system's operational practicality and intuitive interface. The resulting setup times suffered a reduction from an average of 13 min to an average of 4 min, approximately a 69 % reduction. The results were further validated through simulation-mode execution, confirming that the system effectively translates vision-based coordinates into viable print trajectories within the robotic workspace.

While these findings demonstrate a substantial increase in operational efficiency, it is important to note that absolute setup times may vary depending on the specific robotic hardware, ambient lighting conditions, and substrate characteristics of different manufacturing environments. However, adapting the system to these new environmental variables simply requires recapturing reference images under the updated conditions, such as altered lighting, varying substrate geometries, or new clamping setups. Furthermore, the deliberate choice of a panoptic model, combined with a modular script configuration, ensures a highly flexible architecture that streamlines the integration of future feature implementations.

4.3 Repair System Workflow

As previously detailed, the development of the repair system prioritized the enhancement of mask precision and geometric fidelity by incorporating depth map integration. This approach accepted a deliberate trade-off, prioritizing higher spatial accuracy over raw processing speed. The resulting pipeline is illustrated below and executes across a sequence of initialization, dual-system calibration, computer vision processing, and path-alignment verification phases.

The workflow initiates when the operator selects the repair option within the application user interface and positions the workpiece at the center of the printing bed. The target FOV and workpiece height are then provided as inputs to the application, prompting the robot to autonomously move to the coordinates required to achieve the desired FOV. This positioning minimises the GSD while ensuring the target area remains within the frame. Subsequently, the operator manually adjusts the camera lens focus aided by the application's real-time sharpness indicator, after which the application executes the full camera calibration procedure, as detailed in Section 3.3.

The calibration procedure then transitions to a multi-step projector routine. First, the operator positions an A2 calibration board at the designated printing height, allowing the application to perform the projector intrinsic calibration, as described in Section 3.4. Once completed, the physical workpiece is securely fixed into its final printing position, and the operator initializes the printing program via the robot teach pendant. The robot then autonomously moves to a centralized position establishing a baseline 500×500 mm FOV. Within this frame, the operator interactively defines the Region of Interest (ROI) through the application user interface.

With the ROI defined, the robot autonomously aligns itself to center the active printing area, dynamically adjusting its position to preserve the exact sensor-to-surface distance established during calibration. At this final camera position, the system executes the projector's stereo extrinsic calibration to finalize the optical coordinate transformations. The camera and projector are then triggered to acquire a co-registered RGB image and depth map. This RGB-D data is processed by the CMX-B0 model trained in Section 2.5, generating a high-precision mask of the physical component contours. Simultaneously, the RAPID file containing the toolpath for the initial deposition layer is uploaded. The application's placement algorithm evaluates the segmented contours against the nominal toolpath, computing the necessary spatial offset to align the planned deposition path precisely with the physical component boundaries, following the methodology in Section 3.11.

Prior to material deposition, a validation step is then initiated, where the operator executes a printing simulation, allowing for the verification of the calculated offset toolpath. If the alignment is confirmed to be correct, the print operation is ready to be performed.

To validate this integrated pipeline, the visual verification described above was performed in simulation mode to confirm spatial alignment. The results, verified through visual inspection, are shown in Figure 4.7.

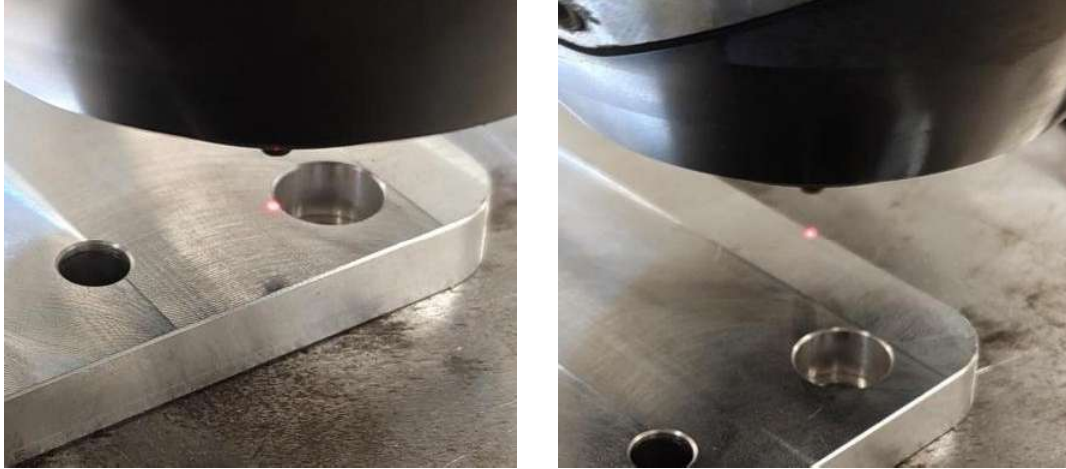


Figure 4.7: Visual verification of spatial alignment in simulation mode, illustrating the result of the integrated pipeline.

Figure 4.8 presents a flowchart summarising the complete repair system workflow described above, from initial part placement through calibration, depth map acquisition, contour alignment, and final deposition. With the current setup, a complete repair job configuration cycle lasts, on average, approximately 33 min.

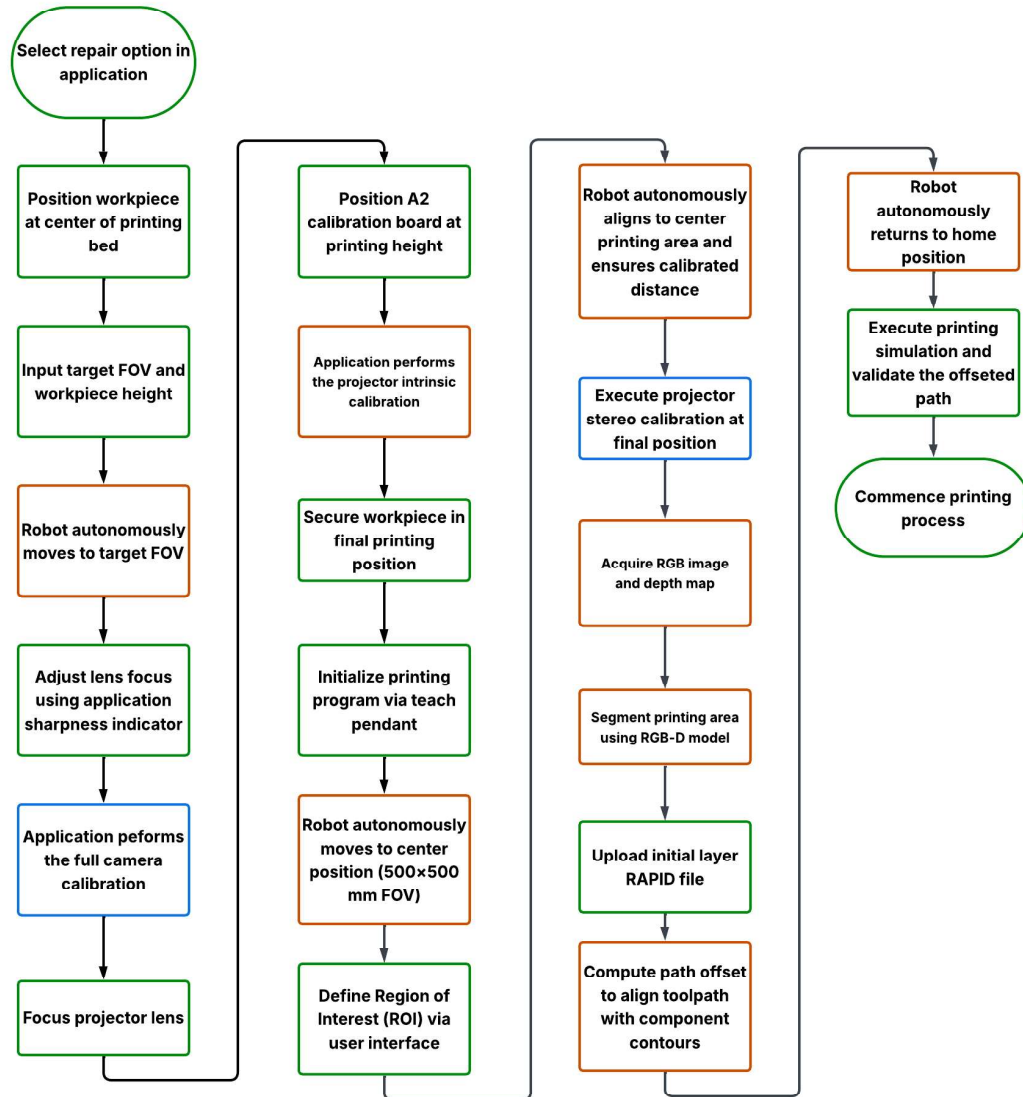


Figure 4.8: Flowchart of the repair system workflow. Green boxes denote operator-only manual steps, orange boxes represent autonomous system actions, and blue boxes indicate combined human-machine interactions.

It is important to contextualise the repair system described in this chapter as a proof of concept rather than a fully production-ready solution. While the pipeline successfully demonstrates the feasibility of RGB-D guided repair alignment, the current implementation requires a complete re-calibration of both the camera and the projector whenever the working distance changes, resulting in a setup time that is impractical unless multiple consecutive repairs are performed on parts of identical geometry, a scenario that is uncommon in typical DED operations. Two directions are identified to address this limitation in future work. First, the adoption of a dedicated stereo vision system, in which the spatial relationship between the two sensors is fixed and factory-calibrated,

would eliminate the need to recalibrate the projector for each intervention. Second, the integration of fixed fiducial targets on the printing table would enable a fully automated calibration routine, reducing operator involvement and allowing setup to proceed in parallel with other preparation tasks, thereby significantly reducing the overall cycle time.

Chapter 5

Conclusions and Future Work

This final chapter closes the dissertation by drawing together the outcomes of the work developed across the preceding chapters. It also reflects on the current limitations of the system as a whole and outlines directions for future development that would extend its accuracy, robustness, and applicability beyond the scope validated in this work.

5.1 Conclusions

This dissertation presented the design, implementation, and validation of a dual-purpose computer vision system for the automation of Directed Energy Deposition operations, addressing two distinct and practically relevant workflows: the nesting of new print toolpaths on different substrates, and the alignment of repair toolpaths with previously deposited surfaces. The following sections summarise the principal findings and contributions of each subsystem.

Camera Calibration

The intrinsic calibration, based on Zhang’s method with a ChArUco board, established the camera matrix and distortion coefficients required to correct lens distortion prior to any metric processing. It achieved an RMS reprojection error of 0.505 pixels at the full sensor resolution of 4504×4504 pixels, providing a reliable foundation for the subsequent extrinsic stage.

Building on these intrinsic parameters, the extrinsic calibration pipeline underwent a structured progression from a direct correspondence method to a hybrid hand-eye formulation that combines the classical $AX=XB$ solver with a simultaneous fixed-point refinement step. The direct correspondence approach, while operationally simple, was limited to a single robot configuration and yielded a mean XY error of 0.347 mm after improvements. The hand-eye calibration, by contrast, produces a transformation that is valid from any robot configuration, as validated across three independent robot poses in Table 3.10. The final calibration achieved a mean XY error of 0.227 mm and a maximum error of 0.299 mm, both within the design targets of 0.3 mm and 0.6 mm respectively.

The investigation also characterised the interaction between lens refocusing and calibration validity: it was established that a major mechanical adjustment of the focus ring requires a full intrinsic and extrinsic recalibration, as focus changes introduce small but non-negligible physical displacements in the lens assembly that alter the effective camera-to-flange transform. This finding showed one of the major problems of the current system, and motivated many of the future improvements discussed in Section 5.2.

Nesting System

The nesting system demonstrated consistent and reliable performance across all nine benchmark scenarios. The No-Fit Polygon algorithm was selected as the optimal method following a comparative evaluation against BLF, random search, and manual placement, achieving the best placement quality with an average corner distance of 100.5 mm and the highest substrate utilisation in the sequential packing test. The complete nesting workflow, from image acquisition to offset transmission, reduced the average setup time from 13 minutes to 4 minutes, a reduction of 69%, while maintaining a spatial placement accuracy within the established safety margins. Laser-pointer simulation confirmed that the computed offsets are correctly applied to the robot work object.

Deep Learning RGB Segmentation

The deep learning segmentation pipeline supporting the nesting system achieved strong quantitative results across both evaluated architectures. The fine-tuned HQ-SAM model, evaluated per class, achieved IoU values of 0.972, 0.654, and 0.965 for the substrate, print deposit, and clamp classes respectively, with corresponding Dice scores of 0.986, 0.770, and 0.982, while Mask2Former achieved IoU values of 0.848, 0.877, and 0.979 for the same three classes, with corresponding Dice scores of 0.916, 0.933, and 0.989, as shown in Figures 3.9 and 3.10. Boundary IoU values of 0.559, 0.265, and 0.430 for HQ-SAM, against 0.387, 0.451, and 0.579 for Mask2Former, as shown in Figure 3.11. This results reflect the inherent difficulty of precisely delineating metallic contours under structured illumination, with each model showing markedly weaker boundary delineation on the print deposit and substrate classes respectively. Despite outperforming Mask2Former on IoU and Dice for the substrate and clamp classes, HQ-SAM was not selected for system integration: Mask2Former was ultimately preferred based on its superior visual consistency, producing structurally coherent masks without the fragmentation artefacts observed in HQ-SAM outputs, and its elimination of the secondary YOLO detection step, which reduces pipeline complexity and removes an additional point of failure.

Depth Map Acquisition and Projector Calibration

A structured light depth acquisition pipeline was implemented and validated using a commercial off-the-shelf digital projector. The Gray code projection and decoding pipeline, combined with a two-phase projector calibration procedure, produced intrinsic and stereo reprojection errors of 0.54 and 0.56 pixels respectively. The resulting depth maps successfully distinguish height

differences between the table surface and the repair printing surface, under the operating conditions of the DED cell. While the proof-of-concept demonstrates the technical feasibility of the approach, the use of a consumer projector with a 1920×1080 native resolution introduces a spatial sampling asymmetry relative to the 4504×4504 camera sensor. An industrial-grade projection unit with a resolution equal to or exceeding that of the camera would be required for full-precision deployment.

RGB-D Segmentation

The RGB-D segmentation pipeline evaluated for the repair workflow achieved consistently high performance across both compared architectures. On the held-out test set, DFormer-Tiny attained a marginally higher area-based accuracy than CMX-B0 for the *PrintingArea* class, with an IoU of 0.922 against 0.909 and a Dice score of 0.956 against 0.941. However, CMX-B0 substantially outperformed DFormer-Tiny on the Boundary F-score (BF@3px), reaching 0.804 against 0.630, a margin of +0.174. Given that the downstream nesting and repair pipeline relies exclusively on the extracted contour of the segmented *PrintingArea*, boundary fidelity rather than raw area overlap was adopted as the decisive selection criterion, and CMX-B0 was therefore selected as the segmentation backbone for the proposed system, despite its higher epoch-to-epoch validation variance during training, which was shown to stem from architectural and batch-size factors rather than from optimization instability. It should be noted that the current test set, while adequate for comparing the relative behaviour of the two architectures, does not fully capture the pose and surface variability expected in real repair scenarios, and a dedicated dataset explicitly featuring identically-textured, overlapping surfaces would be required to conclusively quantify the contribution of the depth channel relative to an RGB-only baseline.

Repair System

The repair pipeline was implemented and validated as a proof of concept, demonstrating end-to-end operation from structured light acquisition through RGB-D segmentation to toolpath alignment and physical deposition. A simulation-mode laser-tracing step confirmed that the computed offsets correctly position the repair toolpath relative to the detected surface contour. A subsequent physical print would be recommended to verify the geometric feasibility of the alignment approach, where the initial deposition layers being visually consistent with the intended repair zone would be an assurance of the pipeline effectiveness.

In its current form, however, the system requires a complete recalibration of both the camera and the projector whenever the working distance changes, which occurs with every new repair part of different geometry. This results in a setup time that is impractical unless multiple consecutive repairs are performed on parts of identical geometry, a scenario that is uncommon in typical DED operations. Two hardware-level changes are identified as the primary path to industrial readiness. First, the adoption of a dedicated stereo vision system, in which the geometric relationship between

the two sensors is fixed and factory-characterised, would eliminate the need to recalibrate the projector for each intervention. This improvements are discussed further in Section 5.2.

5.2 Future Work

Beyond the improvements directly motivated by the limitations identified above, three directions for future development are proposed.

The first concerns calibration accuracy. As noted, the absolute accuracy of the robot's forward kinematics places a floor on the achievable hand-eye error that cannot be reduced by improving the calibration procedure alone, currently limiting this error to approximately 0.2 mm. Incorporating laser tracker measurements to characterise and compensate the robot's kinematic model at each joint configuration, as demonstrated in Cao et al. [35], would allow this floor to be reduced to well below 0.1 mm, which would open the door to repair applications with tighter geometric tolerances than those considered in this work. A complementary improvement that would reduce operator workload is the permanent installation of fixed fiducial markers on the printing table. With known, stable reference points available in every image, the absolute anchor step of the hand-eye refinement could be performed fully automatically at the start of each session, eliminating the need for manual nozzle-contact measurements.

The second concerns the depth acquisition system. The transition from a consumer projector to an industrial structured light head with a native resolution matched to the camera sensor would simultaneously improve depth map resolution, reduce calibration complexity, and eliminate the projector recalibration step entirely. Systems of this type, such as active stereo cameras with integrated illuminators, have the further advantage of producing calibrated depth maps in a coordinate frame that is fixed relative to the camera, removing the need to treat the projector as a separate device in the calibration pipeline.

The third and most expansive direction is the potential extension of the system toward dimensional verification and surface inspection. If the hardware improvements described above are implemented, specifically the adoption of a high-resolution integrated depth sensor, the system would be capable of comparing post-deposition geometry against the intended CAD model. This would enable automatic detection of dimensional deviations and, potentially, visible surface defects such as porosity, cracking, or incomplete fusion, without requiring any additional instrumentation.

References

- [1] Alper Bastas. “Comparing the probing systems of coordinate measurement machine: Scanning probe versus touch-trigger probe”. In: *Measurement* (2020). DOI: [10.1016/j.measurement.2020.107569](#).
- [2] Yang Liu et al. “Laser line scanner based real-time geometry monitoring using Encoder-Decoder network during Directed Energy Deposition”. In: *Measurement* 239 (Jan. 2025), p. 14.
- [3] Zhengyou Zhang. “A Flexible New Technique for Camera Calibration”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (Nov. 2000), pp. 1330–1334. DOI: [10.1109/34.888718](#).
- [4] “A Survey of Vision-Based Methods for Surface Defects’ Detection and Classification in Steel Products”. In: ().
- [5] Irem Ulku and Erdem Akagündüz. “A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D Images”. In: *Applied Artificial Intelligence* 36.1 (2022), p. 2032924. DOI: [10.1080/08839514.2022.2032924](#).
- [6] Alexander Kirillov et al. “Segment Anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 4015–4026. DOI: [10.1109/ICCV51070.2023.00371](#).
- [7] Lei Ke et al. “Segment Anything in High Quality”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 36. 2023.
- [8] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. Version 8.0.0. 2023. URL: <https://github.com/ultralytics/ultralytics>.
- [9] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. “YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information”. In: *arXiv preprint arXiv:2402.13616* (2024). DOI: [10.48550/arXiv.2402.13616](#). arXiv: 2402.13616 [cs.CV].
- [10] Bowen Cheng et al. “Masked-attention Mask Transformer for Universal Image Segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 1290–1299. DOI: [10.1109/CVPR52688.2022.00135](#).
- [11] Fatih Cagatay Akyon, Sinan Onur Altinuc, and Alptekin Temizel. “Slicing Aided Hyper Inference and Fine-tuning for Small Object Detection”. In: *2022 IEEE International Conference on Image Processing (ICIP)* (2022), pp. 966–970. DOI: [10.1109/ICIP46576.2022.9897990](#).
- [12] Ziyue Yang et al. “An Efficient Detection Framework for Aerial Imagery Based on Uniform Slicing Window”. In: *Remote Sensing* 15.17 (2023), p. 4122. DOI: [10.3390/rs15174122](#).

- [13] Iman Gandomi et al. “A Deep Dive Into Capsulorhexis Segmentation: From Dataset Creation to SAM Fine-tuning”. In: *2023 11th RSI International Conference on Robotics and Mechatronics (ICRoM)*. 2023. DOI: [10.1109/ICRoM60803.2023.10412370](https://doi.org/10.1109/ICRoM60803.2023.10412370).
- [14] Junde Wu et al. “Medical SAM Adapter: Adapting Segment Anything Model for Medical Image Segmentation”. In: *Medical Image Analysis* 102 (2023), p. 103547. DOI: [10.1016/j.media.2025.103547](https://doi.org/10.1016/j.media.2025.103547).
- [15] Jonas Moske et al. “Photometric stereo for tool wear monitoring: addressing challenges of specular surfaces in sheet metal forming”. In: *Manufacturing Review* 12 (2025), p. 29. DOI: [10.1051/mfreview/2025026](https://doi.org/10.1051/mfreview/2025026).
- [16] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. “On the Shape of a Set of Points in the Plane”. In: *IEEE Transactions on Information Theory* 29.4 (1983), pp. 551–559. DOI: [10.1109/TIT.1983.1056714](https://doi.org/10.1109/TIT.1983.1056714).
- [17] Saurabh Gupta et al. “Learning Rich Features from RGB-D Images for Object Detection and Segmentation”. In: *European Conference on Computer Vision (ECCV)*. 2014, pp. 345–360. DOI: [10.1007/978-3-319-10584-0_23](https://doi.org/10.1007/978-3-319-10584-0_23).
- [18] Jiaming Zhang et al. “CMX: Cross-Modal Fusion for RGB-X Semantic Segmentation With Transformers”. In: *IEEE Transactions on Intelligent Transportation Systems* 24.12 (2023), pp. 14679–14694. DOI: [10.1109/TITS.2023.3300537](https://doi.org/10.1109/TITS.2023.3300537).
- [19] Bowen Yin et al. *DFormer: Rethinking RGBD Representation Learning for Semantic Segmentation*. 2024. arXiv: [2309.09668](https://arxiv.org/abs/2309.09668) [cs.CV]. URL: <https://arxiv.org/abs/2309.09668>.
- [20] Shuobin Wei et al. “HDBFormer: Efficient RGB-D Semantic Segmentation with A Heterogeneous Dual-Branch Framework”. In: *arXiv preprint arXiv:2311.14470* (2023).
- [21] Siqi Du et al. *AsymFormer: Asymmetrical Cross-Modal Representation Learning for Mobile Platform Real-Time RGB-D Semantic Segmentation*. 2024. arXiv: [2309.14065](https://arxiv.org/abs/2309.14065) [cs.CV]. URL: <https://arxiv.org/abs/2309.14065>.
- [22] Julia A. Bennell and Xiang Song. “A comprehensive and robust procedure for obtaining the nofit polygon using Minkowski sums”. In: *Computers & Operations Research* 35.1 (2008), pp. 267–281. DOI: [10.1016/j.cor.2006.02.026](https://doi.org/10.1016/j.cor.2006.02.026).
- [23] Alexandros Bouganis and Murray Shanahan. “A vision-based intelligent system for packing 2-D irregular shapes”. In: *IEEE Transactions on Automation Science and Engineering* 4.3 (2007), pp. 382–394. DOI: [10.1109/TASE.2006.887158](https://doi.org/10.1109/TASE.2006.887158).
- [24] Paul J. Besl and Neil D. McKay. “A Method for Registration of 3-D Shapes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256. DOI: [10.1109/34.121791](https://doi.org/10.1109/34.121791).
- [25] Szymon Rusinkiewicz and Marc Levoy. “Efficient Variants of the ICP Algorithm”. In: *Proceedings of the 3rd International Conference on 3-D Digital Imaging and Modeling*. 2001, pp. 145–152. DOI: [10.1109/IM.2001.924423](https://doi.org/10.1109/IM.2001.924423).
- [26] Alexander Kolpakov and Michael Werman. “An Approach to Robust ICP Initialization”. In: *arXiv preprint arXiv:2212.05332* (2022). URL: <https://arxiv.org/abs/2212.05332>.
- [27] Jan Flusser, Tomáš Suk, and Barbara Zitová. *Moments and Moment Invariants in Pattern Recognition*. John Wiley & Sons, 2009. ISBN: 978-0-470-69987-4.

- [28] Godfried T. Toussaint. “Solving Geometric Problems with the Rotating Calipers”. In: *Proceedings of IEEE MELECON’83*. 1983, A10.02/1–4.
- [29] Benjamin Merz et al. “Camera-based high precision position detection for hybrid additive manufacturing with laser powder bed fusion”. In: *The International Journal of Advanced Manufacturing Technology* 125.5 (Mar. 2023), pp. 2409–2424. ISSN: 1433-3015. DOI: [10.1007/s00170-022-10691-5](https://doi.org/10.1007/s00170-022-10691-5).
- [30] Afaf Remani et al. “Design of a multi-sensor measurement system for in-situ defect identification in metal additive manufacturing”. In: *Proceedings of the euspen Special Interest Group Meeting: Additive Manufacturing*. 2021. URL: <https://www.euspen.eu/knowledge-base/AM21107.pdf>.
- [31] Frank C. Park and Bryan J. Martin. “Robot sensor calibration: solving $AX=XB$ on the Euclidean group”. In: *IEEE Transactions on Robotics and Automation* 10.5 (1994), pp. 717–721.
- [32] Roger Y. Tsai and Reimar K. Lenz. “A new technique for fully autonomous and efficient 3D robotics hand/eye calibration”. In: *IEEE Transactions on Robotics and Automation* 5.3 (1989), pp. 345–358.
- [33] Radu Horaud and Fadi Dornaika. “Hand-eye calibration”. In: *The International Journal of Robotics Research* 14.3 (1995), pp. 195–210.
- [34] Nicolas Andreff, Radu Horaud, and Bernard Espiau. “On-line hand-eye calibration”. In: (1999), pp. 430–436.
- [35] Dahu Cao et al. “Simultaneous calibration of hand-eye and kinematics for industrial robot using line-structured light sensor”. In: *Measurement* 221 (2023), p. 113508. DOI: [10.1016/j.measurement.2023.113508](https://doi.org/10.1016/j.measurement.2023.113508).
- [36] ABB Robotics. *Technical Reference Manual: RAPID Instructions, Functions and Data Types*. J. Document ID: 3HAC16581-1. ABB Robotics. URL: https://library.e.abb.com/public/688894b98123f87bc1257cc50044e809/Technical%20reference%20manual_RAPID_3HAC16581-1_revJ_en.pdf.
- [37] Python Software Foundation. *socket — Low-level networking interface*. Python 3 Standard Library documentation. Python Software Foundation. URL: <https://docs.python.org/3/library/socket.html>.

Appendix A

Nesting Benchmark Detailed Results

Table A.1: Nesting method benchmark — individual scenario results.

Sc#	Substrate	Print	Method	Time (s)	Dist (mm)	OK
1	Square 400×400	Square 80×80	NFP	0.375	74.4	✓
			BLF	4.975	75.7	✓
			Random	0.339	78.6	✓
			Manual	16.756	75.6	✓
2	Square 400×400	Triangle 100×90	NFP	0.336	74.0	✓
			BLF	2.913	83.2	✓
			Random	0.291	78.7	✓
			Manual	29.181	75.4	✓
3	Square 400×400	Circle $r=50$	NFP	10.764	86.0	✓
			BLF	55.627	86.0	✓
			Random	2.054	87.2	✓
			Manual	19.266	87.3	✓
4	Rectangle 500×300	Square 80×80	NFP	0.402	74.4	✓
			BLF	2.303	74.4	✓
			Random	0.271	85.4	✓
			Manual	20.434	75.6	✓
5	Rectangle 500×300	Triangle 100×90	NFP	0.328	74.0	✓
			BLF	2.416	83.2	✓
			Random	0.281	81.4	✓
			Manual	30.899	76.4	✓

continued on next page

Table A.1 – *continued from previous page*

Sc#	Substrate	Print	Method	Time (s)	Dist (mm)	OK
6	Rectangle 500×300	Circle $r=50$	NFP	10.246	86.0	✓
			BLF	40.019	86.0	✓
			Random	1.585	88.9	✓
			Manual	24.285	87.3	✓
7	Circle $r=220$	Square 80×80	NFP	0.315	145.9	✓
			BLF	3.788	147.4	✓
			Random	0.280	151.0	✓
			Manual	10.781	162.2	✓
8	Circle $r=220$	Triangle 100×90	NFP	0.276	139.8	✓
			BLF	3.760	140.6	✓
			Random	0.276	142.4	✓
			Manual	26.125	142.3	✓
9	Circle $r=220$	Circle $r=50$	NFP	7.197	150.4	✓
			BLF	45.433	151.4	✓
			Random	1.576	151.5	✓
			Manual	19.026	194.2	✓