Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

In Situ Error Correction for 3D Printed Objects with Integrated Electronics on 5-Axis Printers

Dissertation
zur Erlangung des akademischen Grades
Dr. rer. nat
an der Fakultät für Mathematik, Informatik und
Naturwissenschaften
der Universität Hamburg

Eingereicht beim Fachbereich Informatik
von Daniel Ahlers

Dezember 2025

Gutachter:
Prof. Dr. Jianwei Zhang
Prof. Dr.-Ing. Jörg Franke

Tag der Disputation: 08.05.2026

Abstract

3D printing has altered manufacturing by enabling the production of complex objects. The emerging field of printed electronics makes it possible to integrate functional parts with embedded electronics. The full potential of combining 3D printing and printed electronics is unlocked by 5-axis printing. This technique allows circuits to be routed along the surfaces of complex objects. This thesis presents a method to reliably print electronics, correct upcoming errors during printing, and generate a digital twin of the printed object, all achieved on low-cost hardware.

5-axis printing on low-cost printers is challenging due to misalignments in the rotary axes. This challenge is amplified for printed electronics that require a constant distance to the surface and continuous deposition along the path. To address this issue, this work presents a software pipeline that measures the misalignments of the printer's rotary axes and incorporates them into a URDF based model of the printer. An IK solver uses this model to generate compensated toolpaths that account for these deviations. To achieve continuous deposition, surface normals extracted from the underlying object are incrementally adapted along the path. This enables reliable deposition of printed electronic wires onto arbitrary surfaces using imprecise low-cost hardware.

Even with a well calibrated system, errors can still occur during printing, and printed electronics are highly sensitive to these faults. A small imperfection in a wire can make the circuit nonfunctional and cause failure of the entire object. This work presents a method for in itu error detection and repair to address this problem. A neural network segments the wires in images captured during the printing process. By comparing the segmented wires with the intended toolpath, defects are identified. From the identified defects, repair toolpaths are generated to fix them. This results in more reliable electronics with known circuit properties.

For structural objects, reliability of the printing process is also crucial, especially in critical industries such as medical or aerospace. The first step toward quality control and certification is the creation of a digital twin of the printed object. Each printed layer is reconstructed by segmentation with a neural network using two inputs: an image of the current layer and an image of the previous layer. By stacking these individual layer segmentations, a 3D reconstruction of the printed object is created. The reconstruction archives high precision with a resolution of 12 µm per pixel and a mean geometric deviation of 61.5 µm. This digital twin is accurate

enough to enable future quality inspection, adjust printing parameters, and serve as a basis for certification.

In conclusion, the methods proposed in this thesis enable 5-axis printing of objects with embedded electronics on low-cost hardware, ensure the reliability of the printed electronics, and generate accurate reconstructions of structural objects.

Kurzfassung

3D-Druck hat die Fertigung grundlegend verändert, da er die Herstellung komplexer Objekte ermöglicht. Das aufstrebende Gebiet der gedruckten Elektronik erlaubt die Integration funktionaler Bauteile mit eingebetteter Elektronik. Das volle Potenzial der Kombination aus 3D-Druck und gedruckter Elektronik wird erst durch 5-Achsen-Druck erschlossen, weil diese Technik das Verlegen von Schaltkreisen entlang der Oberflächen komplexer Objekte ermöglicht. Diese Dissertation stellt Methoden vor, um Elektronik zuverlässig zu drucken, während des Drucks auftretende Fehler zu korrigieren und einen digitalen Zwilling des gedruckten Objekts zu erstellen - alles auf kostengünstiger Hardware.

5-Achsen-Druck auf kostengünstigen Druckern ist aufgrund von Fehlstellungen der Rotationsachsen herausfordernd. Diese Problematik verschärft sich bei gedruckter Elektronik, da sie einen konstanten Abstand zur Oberfläche sowie einen kontinuierlichen Materialauftrag entlang des Pfades erfordert. Um dieses Problem zu lösen, präsentiert diese Arbeit eine Softwarelösung, die die Fehlstellungen der Rotationsachsen des Druckers präzise misst. Auf Basis dieser Messungen wird ein URDF-basiertes Modell des Druckers erstellt. Ein Inverse-Kinematik-Solver nutzt dieses Modell, um kompensierte Werkzeugwege zu generieren, die die erkannten Abweichungen ausgleichen. Für einen Materialauftrag mit gleichmäßiger Geschwindigkeit werden die Oberflächennormalen des zugrunde liegenden Objekts extrahiert und entlang des Pfades schrittweise angepasst. Dies ermöglicht den zuverlässigen Auftrag gedruckter elektronischer Leitungen auf beliebigen Oberflächen auf unpräziser, kostengünstiger Hardware.

Auch bei einem gut kalibrierten System können während des Drucks Fehler auftreten. Gedruckte Elektronik ist für solche Abweichungen besonders anfällig. Bereits eine kleine Unregelmäßigkeit in einer Leitung kann die gesamte Schaltung funktionslos machen und damit das Objekt unbrauchbar werden lassen. Zur Lösung dieses Problems stellt diese Arbeit eine In-situ Methode zur Fehlererkennung und -reparatur vor. Hierfür segmentiert ein neuronales Netzwerk die Leitungen in Bildern, die während des Druckvorgangs aufgenommen werden. Der Vergleich der segmentierten Leitungen mit dem vorgesehenen Werkzeugpfad identifiziert Defekte. Die identifizierten Defekte werden anschließend mithilfe eines generierten Reparaturpfads behoben. Diese Technik führt dazu, dass zuverlässigere Elektronik mit bekannten Schaltungseigenschaften gedruckt werden kann.

Auch bei strukturellen Objekten ist die Zuverlässigkeit des Druckprozesses von entscheidender Bedeutung, insbesondere für Anwendungen in kritischen Branchen wie Medizin oder Luft- und Raumfahrt. Ein erster Schritt hin zu Qualitätskontrolle und Zertifizierung ist die Erstellung eines digitalen Zwillings des gedruckten Objekts. Diese Arbeit stellt eine Methode vor, jede gedruckte Schicht zu rekonstruieren, indem ein neuronales Netzwerk jede einzelne Schicht segmentiert. Als Eingangsdaten dienen dabei neben dem Bild der aktuellen Schicht auch die Aufnahme der vorherigen Schicht. Durch das Stapeln der segmentierten Einzelschichten entsteht eine 3D-Rekonstruktion des gedruckten Objekts. Die Rekonstruktion erreicht eine hohe Präzision mit einer Auflösung von 12 µm pro Pixel und einer mittleren geometrischen Abweichung von 61.5 µm. Der digitale Zwilling ist hinreichend genau, um zukünftige Qualitätsinspektionen zu ermöglichen, Druckparameter anzupassen und als Grundlage für eine Zertifizierung zu dienen.

Zusammenfassend ermöglichen die in dieser Dissertation vorgestellten Methoden den kostengünstigen 5-Achsen-Druck von Objekten mit eingebetteter Elektronik, sichern die Zuverlässigkeit der gedruckten Elektronik und erlauben eine präzise Rekonstruktion von Strukturteilen.

Contents

Chapter 1

Introduction

3D printing, also known as Additive Manufacturing (AM), is a technology that has revolutionized the way objects are designed and manufactured. The first concept of 3D printing was developed by Hideo Kodama in 1981 [1] where he created objects from a resin by hardening layers of resin with Ultraviolet (UV) light and stacking them onto each other in a specifically designed machine. Later, Scott Crump invented the Fused Deposition Modeling (FDM) process [2] in which a thermoplastic filament is extruded through a nozzle and deposited layer by layer to create a solid object. Both of these early 3D printing processes where patented and commercialized in the late 1980s and early 1990s and were mostly used in industrial applications due to their high cost.

The expiration of these patents, especially the FDM patent, marked a major turning point in the accessibility of 3D printing technology. The Replicating Rapid-prototyper (RepRap) project introduced self-replicating 3D printers that could manufacture most of their own components [3]. This led to a drastic decrease in costs and triggered rapid growth and innovation across the 3D printing and Additive Manufacturing industry. Multiple companies began developing their own 3D printers based on the RepRap project and made them widely available to the public. Through this trend, other technologies like Stereolithography (SLA) also became more affordable and accessible for consumers.

Today, 3D printing has fundamentally altered manufacturing by enabling the production of complex geometries, highly customized features, and more resource efficient processes [4]. It has moved from niche prototyping to mainstream manufacturing, driven by falling equipment costs, broader material capabilities, and growing demand for product personalization. The technology is widely adopted across industries, including aerospace, automotive, healthcare, and consumer products [5]. For example, SpaceX reduced the weight of its Raptor 3 engine by 7% while increasing thrust by 21% using 3D-printed metal parts [W1]. The primary advantage over traditional manufacturing is its flexibility. Designers can iterate rapidly (rapid prototyping) and create geometries that are not achievable with other manufacturing methods [6]. From an economic perspective, 3D printing is preferable to conventional manufacturing in scenarios that

combine low production volumes, highly complex part geometries, and highly customized designs [7]. The global 3D printing market was valued at 22 billion USD in 2024, with annual growth of about 10-20% over the last decade [8].

Early research and development in 3D printing focused primarily on structural performance and aesthetic quality. As the technology matured, researchers began to explore functional applications, particularly the direct integration of electronics into printed parts. This shift enables electronics to be embedded within complex 3D geometries, resulting in functional objects. While the first printed electronics were relatively simple and limited in capability [9], ongoing research has pushed these boundaries. Today, a wide range of processes and materials is available for printing electronics [10], enabling single-piece, fully integrated devices without the constraints of traditional PCB integration.

The layer-based printing on 3-axis machines limits both design freedom and achievable mechanical properties. To overcome these limitations, Curved Layer Fused Deposition Modeling (CLFDM) was introduced [11] and has been shown to increase strength [12] and reduce surface roughness [13]. To further expand this freedom, additional axes were added to printers to print cylindrical objects [14], to print with a 6 Degrees of Freedom (DOF) robotic arm [15], or to print segments in arbitrary orientations [16]. For printed electronics, this greater freedom is especially beneficial because it allows wires to follow the object's surfaces and contours, resulting in a more compact design [17].

Transitioning from conventional 3-axis systems to advanced 5-axis printers makes path planning significantly more complex. Calibration methods must also be adapted to achieve the accuracy required for multi-axis printing. With integrated electronics, these challenges intensify. Because the electronics are embedded, conventional post-production inspection and repair are not feasible. As a result, robust in-process error detection and correction become essential for reliability, since defects in printed conductive paths can render an entire device faulty. For structural parts, an additional challenge is the growing demand for methods that can reconstruct, inspect, and validate the internal structure immediately after fabrication. This would help bridge the gap between the digital design and its physical realization, especially for safety-critical parts, such as those used in aerospace or medical applications.

1.1 Aim of this Thesis

To overcome the challenges outlined above, this thesis aims to develop and validate an integrated toolchain that enables reliable, support efficient 5-axis printing of objects with embedded electronics on low-cost platforms, complemented by in-process defect detection/correction and layer-wise digital twin reconstruction for quality assurance. The main objectives are:

Calibration Develop and validate a machine specific calibration methodology that measures geometric and kinematic errors and converts them into a compensation model for accurate 5-axis motion.

Path Planning Create a framework for conformal printed electronics that generates continuous, surface following toolpaths with correct tool orientation and integrates compensation and machine constraints.

Wire Detection Design and evaluate an in situ, vision based monitoring system that detects and localizes printed conductive structures across materials, geometries, and machine setups.

Error Detection and Correction Develop real-time methods to detect common defects (e.g., connection breaks and shorts) during printing, and implement autonomous strategies to repair these defects, resulting in a working circuit.

Layer Segmentation Investigate generalizable imaging and segmentation methods that reliably capture the actual printed structure of each printed layer.

Object Reconstruction Develop a accurate 3D reconstruction method that creates a digital twin of the printed object from the segmented layers, enabling detailed quality inspection and traceability.

Because Fused Filament Fabrication (FFF) is the most widely used technology on low-cost printers, hardware development concentrates on FFF-based systems, with additional validation on an industrial 5-axis platform (Neotech PJ15X). Reliability is demonstrated through experiments and case studies that cover multiple geometries, materials, imaging setups, and printer configurations.

1.2 Research Question and Contribution

The objective of this thesis is to advance the state of the art in 3D printing of objects with integrated electronics on multi-axis systems by addressing key challenges in calibration, path planning, in-process defect detection and correction, and layer-wise reconstruction for quality assurance. This leads to the following research questions, which are addressed in this dissertation:

Path Planning for 5-Axis Printing How can toolpaths be derived from arbitrary 3D geometries that ensure continuous, surface-conformal deposition while respecting machine kinematics? How can calibration data be robustly modeled and integrated into a path planning framework to enable accurate, reliable 5-axis 3D printing of geometrically complex objects with embedded electronics?

In Situ Error Correction of Printed Electronics What reliable vision based methods can detect and localize conductive wires during printing? How can errors such as connection breaks and short circuits be identified in real time, and which autonomous repair strategies can be executed to correct them?

Layer Segmentation and Object Reconstruction How can the actual printed structure be captured layer by layer? How can this information be used to create a high-fidelity digital twin of the printed object for quality inspection and certification?

1.3 Publications

Most of the content in this thesis has already been published in research papers. These publications are listed below with a short summary of their content.

1.3.1 Core Publications

This section lists the core publications that are directly related to this thesis.

Daniel Ahlers. "In-Situ Verification of 3D-Printed Electronics Using Deep Convolutional Neural Networks". In: *Proceedings of the 32nd Annual International Solid Freeform Fabrication Symposium*, 2021 [18]

This work introduces a vision based method that detects and localizes printed conductive wires during the printing process. A neural network is trained to segment the wires in images captured in situ. The resulting segmentation is then compared with the intended toolpath to identify errors such as connection breaks, shorts, and unreached points. Chapter 5 is partly based on this publication.

Daniel Ahlers, Florens Wasserfall, Johannes Hörber, and Jianwei Zhang. "Automatic in-situ error correction for 3D printed electronics". In: *Additive Manufacturing Letters 7*, 2023 [19]

This work builds on the previous publication by introducing in situ detection and correction of errors in printed electronics. From the detected errors, a repair toolpath for connection breaks is generated and then executed to repair the defect. The approach is evaluated on an industrial 3D printer across various test cases. Chapter 5 is partly based on this publication.

Daniel Ahlers, Tom Schmolzi, German Junca, Jianwei Zhang, and Florens Wasserfall. "Calibration and compensation of 5-axis 3D-printers for printed electronics". In: *Additive Manufacturing Letters 12*, 2025 [20]

This work presents a calibration and compensation approach for 5-axis 3D printers to improve accuracy. It introduces a touch-probe based calibration procedure, and stores the measured parameters in a Unified Robot Description Format (URDF) based compensation model. This model is then integrated into a path-planning framework to generate accurate toolpaths and is evaluated on a low-cost 5-axis printer. Chapter 4 is based on this publication.

Daniel Ahlers, Niklas Fiedler, Florens Wasserfall, and Jianwei Zhang. "Camera Based In Situ Layer Segmentation and Object Reconstruction for Digital Twins in FFF 3D Printing". Submitted to: *Journal of Intelligent Manufacturing*, 2025 [21]

This work is currently under review. It presents an approach for creating a digital twin of a printed object from images captured during the printing process. For each printed layer, a neural network segments the images to identify the regions that were actually deposited in this layer. By stacking these segmented layers, a 3D reconstruction of the printed object is generated forming a digital twin. The reconstruction approach is evaluated on its ability to generalize to unseen objects, its capability to preserve fine details, and its geometric accuracy. Chapter 6 is based on this publication.

1.3.2 Other Publications

The publications listed below are only marginally related to this thesis, or were published before the thesis period started.

Florens Wasserfall, **Daniel Ahlers**, Norman Hendrich, Jianwei Zhang, "3D-Printable Electronics - Integration of SMD Placement and Wiring into the Slicing Process for FDM Fabrication", In: *Proceedings of the 27th International Solid Freeform Fabrication Symposium*, 2016 [22]

This work presents an approach for integrating printed wires and Surface Mounted Device (SMD) components into the slicing process of FFF printers. It introduces a software toolchain that models the circuit schematic, generates the corresponding wire deposition tool paths, and inserts the required pick-and-place commands into the resulting G-code.

Daniel Ahlers, Florens Wasserfall, Norman Hendrich, Jianwei Zhang, "3D Printing of Nonplanar Layers for Smooth Surface Generation", In: *15th IEEE Conference on Automation Science and Engineering (CASE)*, 2019 [23]

This work presents a path-planning approach for printing smooth surfaces with three-axis printers. It automatically generates nonplanar layers on top of the given object while avoiding collisions.

Florens Wasserfall, **Daniel Ahlers**, Norman Hendrich, "Optical In-Situ Verification of 3D-Printed Electronic Circuits", In: *15th IEEE Conference on Automation Science and Engineering (CASE)*, 2019 [24]

This work presents an Support Vector Machine (SVM)-based method for detecting wires in images captured during the printing process. The SVM classifies pixels by their color in the Hue Saturation Value (HSV) color space. A sliding-window detector is applied along the wire to identify missing material. This work is the predecessor of [18].

Florens Wasserfall, Norman Hendrich, **Daniel Ahlers**, Jianwei Zhang, "Topology-Aware Routing of 3D-Printed Circuits", In: *Additive Manufacturing Journal 36*, 2020 [25]

This work introduces an automated wire-routing method for 3D-printed electronics. Connections are extracted from the circuit schematic and routed through the object using an A* algorithm that follows the object's overall layer structure. The object's toolpath is then adapted to automatically create cavities around the routed wires.

Daniel Ahlers, Florens Wasserfall, Norman Hendrich, Arne Büngener, Jan-Tarek Butt, Jianwei Zhang, "Automated In-Situ Placing of Metal Components Into 3D Printed FFF Objects", In: *IEEE/ASME Transactions on Mechatronics 26*, 2021 [26]

This work presents a method for automatically inserting metal components into 3D-printed objects during the printing process. A low-cost FFF printer is extended with a magnetic pick-and-place unit and a camera system, enabling the placement of nuts and other small components into automatically generated cavities during printing.

1.4 Structure of the Thesis

The remainder of this thesis is structured as follows:

Chapter 2 This chapter introduces the fundamental concepts of this thesis. It first reviews 3D printing, outlining the main technologies and processes, explains the FFF workflow from the design phase through toolpath generation to the actual printing, and describes the basic components of a typical FFF printer. Then, it introduces printed electronics, summarizing the main processes for applying conductive traces onto a substrate and the common path-planning approaches used for printed electronics. Finally, it describes the neural network architectures used in this thesis.

Chapter 3 This chapter describes the hardware developed and used in this thesis. It first covers the two modified E3D ToolChanger printers used in most experiments, detailing the

toolheads and the mechanical, electrical, and software modifications that enable printed electronics, five-axis printing, and in situ monitoring. It then briefly introduces the industrial Neotech PJ15X printer used in selected experiments.

Chapter 4 This chapter presents a novel path-planning framework for 5-axis printing of objects with embedded electronics. It begins by introducing a calibration procedure that measures and models inaccuracies in 5-axis printers. Then it explains how this model is integrated into the path-planning framework to compensate for these inaccuracies. It also presents methods for generating surface-conformal toolpaths that ensure continuous deposition on 5-axis printers. Finally, it evaluates the framework on a low-cost 5-axis printer across various test cases.

Chapter 5 This chapter focuses on in situ error detection and correction for 3D-printed electronics. It begins by presenting a vision based method that uses a neural network to detect and localize printed conductive wires during printing. It then explains how errors such as connection breaks and short circuits are identified in real time, and how autonomous repair strategies are executed to correct them. Finally, it evaluates the approach on a range of test cases.

Chapter 6 This chapter introduces a method for creating a digital twin of a printed object without the need for expensive hardware. It captures an image of each printed layer and uses a neural network to segment these images, identifying the regions actually deposited in each layer. Then, the resulting sequence of segmented layers is reconstructed into a 3D model of the printed object. The performance and accuracy of both segmentation and reconstruction are evaluated in multiple experiments.

Chapter 7 This chapter summarizes the main findings of this thesis, highlights its key contributions, and suggests directions for future research.

Chapter 2

Basics

2.1 3D Printing

The first 3D printing process was described by Hideo Kodama in 1981 [1]. He created objects from a resin by hardening layers of resin with UV light and stacking them onto each other in a specifically designed machine. In 1986 Charles Hull developed a process that uses a laser to cure the resin into a solid object. This process is known as SLA [27]. The first commercial 3D printer with this technique was released in 1987 by 3D Systems [W2].

In 1989 Scott Crump came up with the FDM process where a thermoplastic filament is extruded through a nozzle and deposited layer by layer to create a solid object [2]. The first FDM 3D printer was released in 1992 by Stratasys [W3]. The term FFF describes the same process and is often used to avoid the trademarked term FDM by Stratasys. All of these early 3D printers were expensive and used mostly in industry applications.

The expiration of the patent on the FDM process in 2009 paved the way for the development of affordable 3D printers. Following this, the RepRap project introduced self-replicating 3D printers that could manufacture most of their own components [3]. This democratization of 3D printing technology resulted in a dramatic decrease in costs, which triggered rapid growth and innovation throughout the entire 3D printing and Additive Manufacturing industry. Multiple companies started to develop their own 3D printers based on the RepRap project and made them easily available for everyone. Through this trend, other technologies like SLA also became more affordable and available for consumers.

Today, 3D printing is a widely used technology in various industries, including aerospace, automotive, healthcare, and consumer products [5]. 3D printing offers several advantages: it enables rapid prototyping and design iteration, supports the creation of complex geometries and customization that are difficult or impossible with traditional manufacturing, and reduces material waste compared to subtractive methods. However, there are also notable limitations: the range of available materials and their mechanical properties is more restricted than with

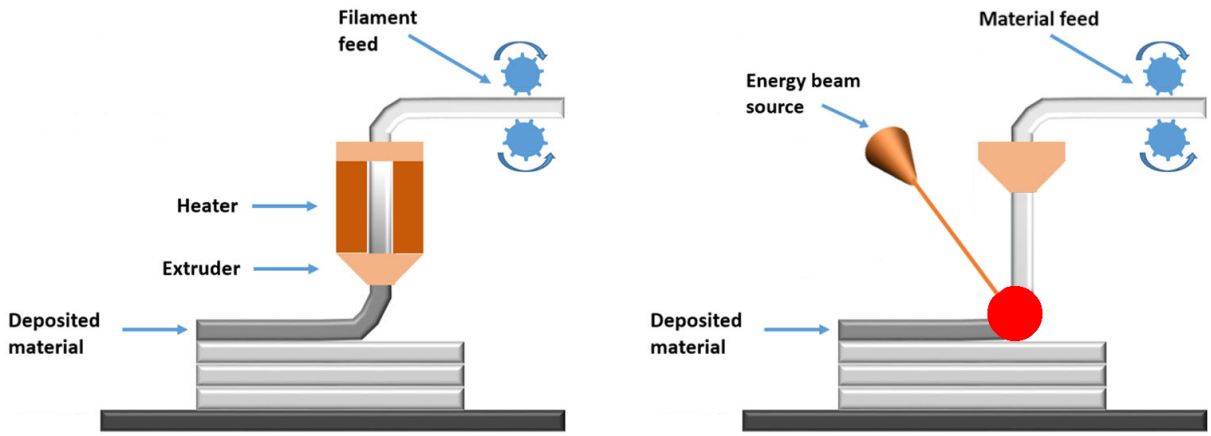
conventional manufacturing, surface finish and dimensional accuracy may be lower, and printing speed and scalability can be limiting factors for mass production.

2.1.1 3D Printing Technologies

Through the rise of 3D printing and additive manufacturing, new applications and technologies were developed. The various 3D printing technologies are systematically categorized by the International Organization for Standardization (ISO) in the ISO/ASTM 52900:2021 standard [28]. This classification provides a clear framework for understanding the major additive manufacturing processes. The following subsections describe each technology according to this standard.

Material Extrusion (MEX)

MEX is the most common printing process for consumer grade 3D printers. It is what most people have in mind when they think of 3D printing. This work primarily focuses on this process. In MEX, material is extruded through a nozzle to create a continuous line that is precisely deposited onto the printbed. This extruded line follows a precalculated path, forming both the outer contours and the internal structure of a single horizontal slice of the object, known as a layer. By stacking multiple layers on top of each other, a complete 3D printed object is constructed. The MEX process is versatile and can utilize a wide range of materials, including thermoplastics, biomaterials, concrete, and others. Thermoplastics are molten and extruded through a nozzle, where they cool down and solidify to form the final object. This process is called FFF or FDM and is further explained in Section 2.1.2. MEX also used in bio printing, where biomedical



(a) Material Extrusion (MEX) **(b) Direct Energy Deposition (DED)**

Figure 2.1 – In both processes the material is deposited in lines onto the printbed. While in MEX the material is extruded through a heated nozzle, in DED the material is deposited and immediately fused together by an energy source after deposition. [29]

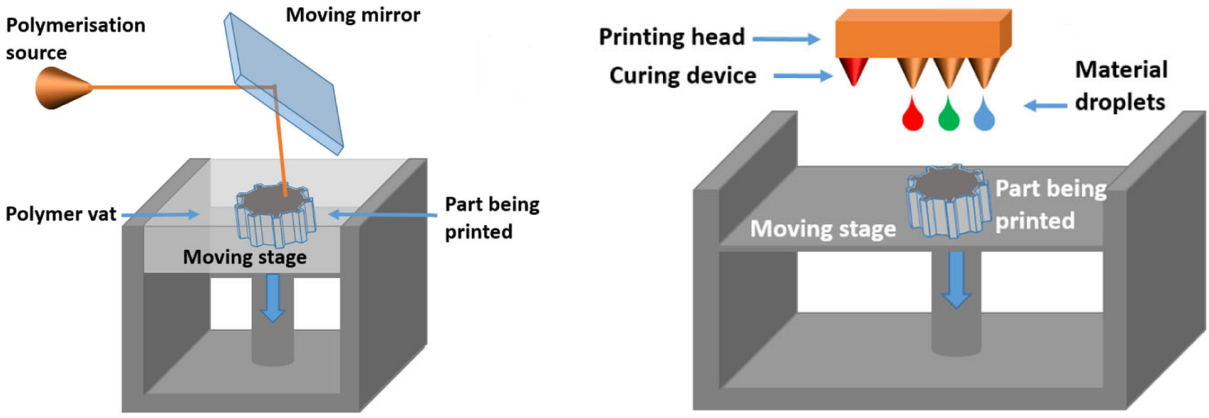
material or cells are extruded into a gel to create 3D structures, or in concrete printing where special concrete is pumped through a nozzle to create buildings. Although material extrusion differs, the basic principle stays the same. The deposition of material in continuous lines, layer by layer, often results in visible layer lines on the surface of the printed object. Additionally, thermoplastics can deform as they cool, which may compromise the dimensional accuracy of the final part. Because each new layer is built upon the previous one, any features that overhang the previous layer require additional support during printing. These support structures are made from the same material as the object or from a specialized support material designed to be easily removed or dissolved after the print is complete. Figure 2.1a shows a schematic of the MEX process.

Direct Energy Deposition (DED)

DED is generally similar to MEX, but in this process, the raw material is deposited and immediately fused together by an energy source after deposition. Common energy sources include lasers, electron beams, plasma arcs, or wire arcs. DED covers a wide range of specialized processes, most of which are used for industrial metal printing. The various sub-processes are primarily distinguished by the type of energy source and the method of material feed. For instance, in Wire Arc Additive Manufacturing (WAAM), a wire is fed through a nozzle and melted by an arc, much like in traditional arc welding. This technique is used not only to fabricate new objects but also to repair or add material to existing components. Most DED processes result in a rough surface finish, which typically requires postprocessing, such as CNC milling, to achieve the desired smoothness and precision. However, the use of a high-energy source in these processes enables the deposited material to fuse together exceptionally well, leading to a printed object with high mechanical strength and strong interlayer bonding. Figure 2.1b shows a schematic of the DED process.

Vat Photopolymerization (VPP)

The second major category of consumer 3D printers is VPP. In this process, a photosensitive resin is selectively cured in a vat using a UV light source. To create each layer, the printbed is lowered into the vat so that a precise layer of resin forms above the printbed. The UV light then hardens the resin only in the areas where the object is intended to form. After the layer is cured, a fresh layer of resin is spread on top of the previously hardened layer. Overhanging structures can be hardened, but will sag if they are not hold up by support structures. These support structures need to be from the same material and have to be removed after printing. VPP produces objects with a smooth surface finish and high resolution, making it suitable for applications that require fine detail and precision like jewelry prototyping or dental applications. There are different processes to harden the resin. In Stereolithography (SLA), a laser is used to trace the outlines of each layer. The laser provides higher energy compared to other methods, but it is more difficult



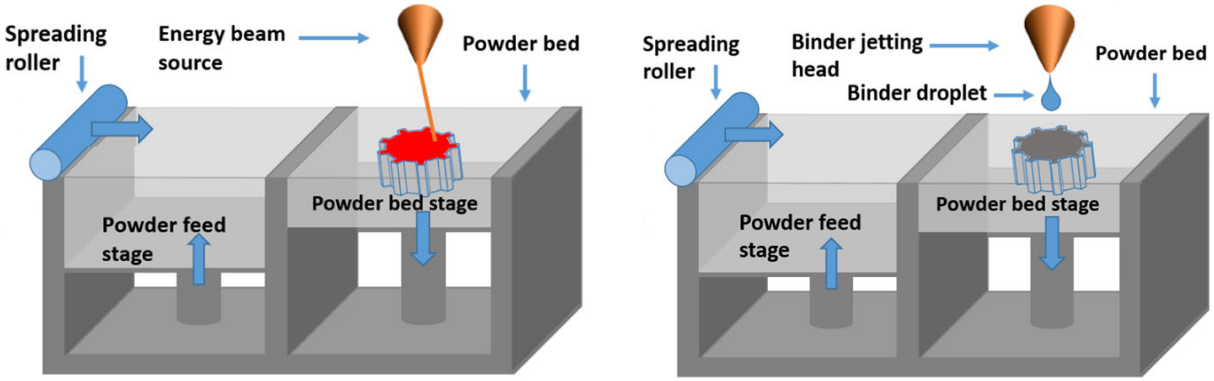
(a) Vat Photopolymerization (VPP) **(b) Material Jetting (MJT)**

Figure 2.2 – Both processes create objects by hardening photosensitive resin layer by layer using a UV light source. In the VPP process, the printbed is submerged in a vat of resin, and each layer is selectively cured within the vat. In contrast, the MJT process jets resin precisely onto the printbed, and each deposited layer is then cured. [29]

to control and more costly. In Digital Light Processing (DLP) printing, a projector is used to expose and cure an entire layer of resin simultaneously, while in Liquid Crystal Display (LCD) printing, a screen selectively hardens the whole layer at once. Both methods are faster and more cost-effective than SLA, as they eliminate moving parts and simplify the layer preparation process. However, their resolution is lower because the light source is not as precisely focused as in SLA. Figure 2.2a shows a schematic of the VPP process.

Material Jetting (MJT)

To create an object using MJT, a photosensitive resin is deposited onto the printbed in a line-by-line fashion through an array of nozzles. The resin is jetted precisely in the areas where the object is intended to form, in a manner similar to inkjet printing on paper. Once the resin is deposited, a UV light source cures the material at the locations where it has been applied. After each layer is cured, the printbed is lowered, and a new layer of resin is jetted on top, repeating the process until the object is complete. In this process, the printing speed is independent of the object's size because the entire layer is deposited in a single pass by jetting material across the printbed with multiple nozzles. Additionally, MJT technology allows for the use of colored inks, enabling the production of full-color prints. This process is primarily used in industry to produce parts with high resolution, good surface finish, and precise dimensional accuracy. Because inkjet droplets require a surface to land on, the technology is not suitable for printing overhanging features without additional support. To address this, support structures are jetted beneath any overhangs during printing. These supports must be made from a different material that can be removed after the print is complete. Most printers use a wax-based support material, which can be dissolved either by applying heat or using a suitable solvent. Special resins containing



(a) Powder Bed Fusion (PBF) **(b) Binder Jetting (BJT)**

Figure 2.3 – Both processes create objects by fusing powder particles layer by layer. In PBF, a thermal energy source selectively fuses the powder, while in BJT, a liquid binder is deposited onto the powder to bond the particles together. [29]

suspended metal or ceramic particles are also used to produce metal or ceramic parts. After printing, the resin is either sintered or burned out, leaving behind the desired metal or ceramic component. Additionally, single-nozzle jetting is commonly used in the fabrication of printed electronics, as presented in more detail in Section 2.2.1. Figure 2.2b shows a schematic of the MJT process.

Powder Bed Fusion (PBF)

In PBF, a thin layer of powder is evenly distributed across the printbed using a spreading roller. A thermal energy source, such as a laser or electron beam, then selectively sinters or melts the powder particles together by scanning the surface of the layer according to the desired geometry. After a layer is fused, the printbed is lowered, and a fresh layer of powder is spread on top. Throughout the printing process, the part is completely surrounded by loose powder, which provides support for overhanging features. Once printing is complete, the loose powder is removed from the finished part and can be reused for subsequent prints. This process provides high resolution and produces parts with a smooth surface finish. However, the equipment required is very costly and is almost exclusively used in industrial settings. In the Selective Laser Sintering (SLS) process, polymer powder particles are sintered together to form the final part. For metals, the fusion is achieved either by a laser in the Laser Powder Bed Fusion (LPBF) process or, when higher energy is required, by an electron beam in the Electron Beam Melting (EBM) process. Figure 2.3a shows a schematic of the PBF process.

Binder Jetting (BJT)

In the BJT process, each layer of the object is formed from a thin bed of powder, similar to the PBF process. However, instead of using heat to fuse the powder particles, a liquid binder is

selectively deposited onto the powder layer. This binder is applied using inkjet nozzles where the object is intended to form. In some processes the binder is mixed with coloured ink to create full color prints. After the printing process is complete, the part is extracted from the surrounding powder bed and any loose powder is removed. The BJT process is a cold technology because it does not use a direct energy source to fuse the powder. The powders can be from a wide range of materials like sand, ceramics, metals, and more. Some materials like ceramics or metals are sintered or fused after printing in an additional process step. In this step, the whole object shrinks since the binder is removed and the powder is sintered or fused together. BJT is mostly used in industry for creating sand molds for casting or to create complex geometries from metal or ceramics. Figure 2.3b shows a schematic of the BJT process.

Sheet Lamination (SHL)

SHL differs significantly from other 3D printing processes. In this method, each layer of the object is created by cutting a prefabricated sheet of material into the desired shape. These individual sheets are then stacked and bonded together through heat, pressure, or adhesive to form a solid object. The sheet lamination process can use a variety of materials, including paper, metal, polymers, or carbon fiber mats. This approach enables the fabrication of objects composed of multiple materials, as each sheet can be made from a different material and bonded together during the lamination process. However, sheet lamination is not widely adopted, as the machines required are expensive and other 3D printing processes often provide superior properties. Additionally, the cutting process generates significant material waste, which is a notable drawback of this method. Figure 2.4 shows a schematic of the SHL process.

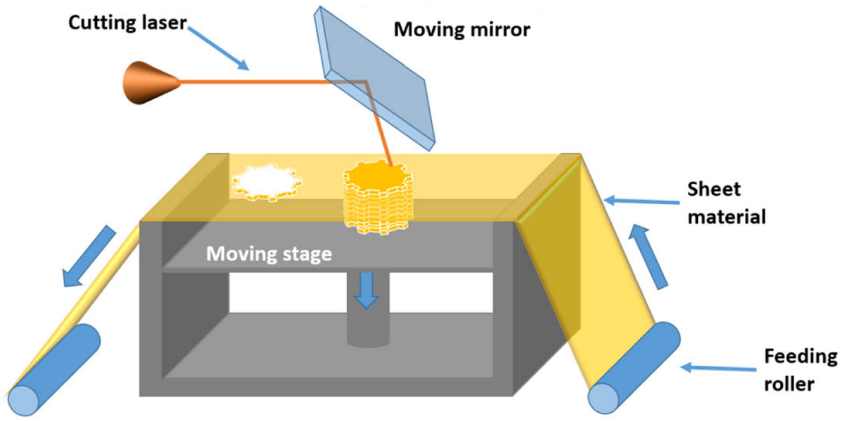
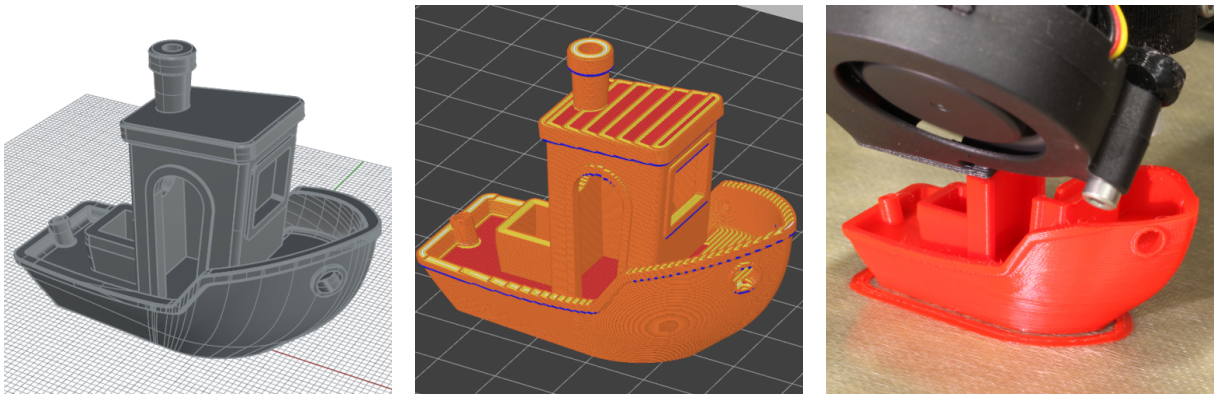


Figure 2.4 – In the SHL process, each layer of the object is formed by cutting a prefabricated sheet of material into the desired shape. These sheets are then fused together using heat, pressure, or adhesive to build up the final object layer by layer. [29]



(a) **(b)** **(c)**

Figure 2.5 – The three steps of the FFF printing process: **(a)** design, where the 3D model is created. **(b)** Toolpath generation, where the model is sliced into layers and the toolpath is generated. **(c)** Printing, where the printer follows the toolpath to fabricate the object layer by layer.

2.1.2 FFF Printing Process

The 3D printing process consists of three basic steps. First is the design step, where the object's model is created from scratch or imported using 3D modeling software. Next is the toolpath generation step, in which the model is sliced into layers and the printer's toolpath is generated. Finally, in the printing step, the printer follows the toolpath instructions to fabricate the object layer by layer. While these steps are common to all 3D printing technologies, the specific details of each step vary depending on the process. Since this work mainly focuses on the FFF process, the following sections describe each step as it applies to FFF printing. Figure 2.5 illustrates the three steps of the FFF printing process.

Design

The first step in the printing process is creating the object to be printed. The user must define the desired model specifications and determine whether these requirements can be achieved using the chosen 3D printing process. During the design phase, it is important to consider process-specific requirements. For example, ensure the model has a flat surface for the first layer, minimize large overhangs, and avoid shallow slopes that can result in visible stair-stepping. The orientation of the part also influences its mechanical strength, printing time, and material usage. These factors should be carefully evaluated when designing the model.

A 3D model can be created from scratch using Computer Aided Design (CAD) software, downloaded from large online databases such as Thingiverse [W4] or Printables [W5], or obtained by scanning an object with a 3D scanner. For successful printing, the model must be a solid surface, watertight, and free of self-intersections. FFF slicers require mesh-based models for processing which are usually provided in the STL file format. Surface-based models in

the STEP format are also supported by modern slicers, but they are converted to meshes before slicing.

Toolpath Generation

The toolpath generation step transforms the 3D model into a set of machine instructions that the printer can execute to fabricate the object. In 3D printing, this process is commonly known as slicing, and the software responsible for it is referred to as a slicer. Open-source projects dominate the development of FFF slicers, with CuraEngine and libslic3r being the two major slicing engines in use. These engines serve as the core of popular slicer applications such as Cura [W6], Prusaslicer [W7], and Orcaslicer [W8]. Figure 2.6 shows the interface of the PrusaSlicer software.

To generate the toolpath, the mesh model is first loaded into the slicer software. The slicer cuts the model into horizontal slices, known as layers. This approach is used for almost all 3D printers, except for advanced path planning techniques such as nonplanar printing and multi-axis printing, which are discussed in Chapter 4. For each layer, the slicer generates the toolpath by first identifying the outer contour of the layer and then offsetting this outline inward to create a shell toolpath. Typically, the shell consists of 2-3 lines that form the perimeter of the layer. The remaining interior area is filled with an infill pattern or filled solid. Infill patterns are designed to save material, reduce print time and weight, while still maintaining structural strength. Regions that are a top or bottom surface are filled solid to ensure a smooth, flat finish on the outside of the object.

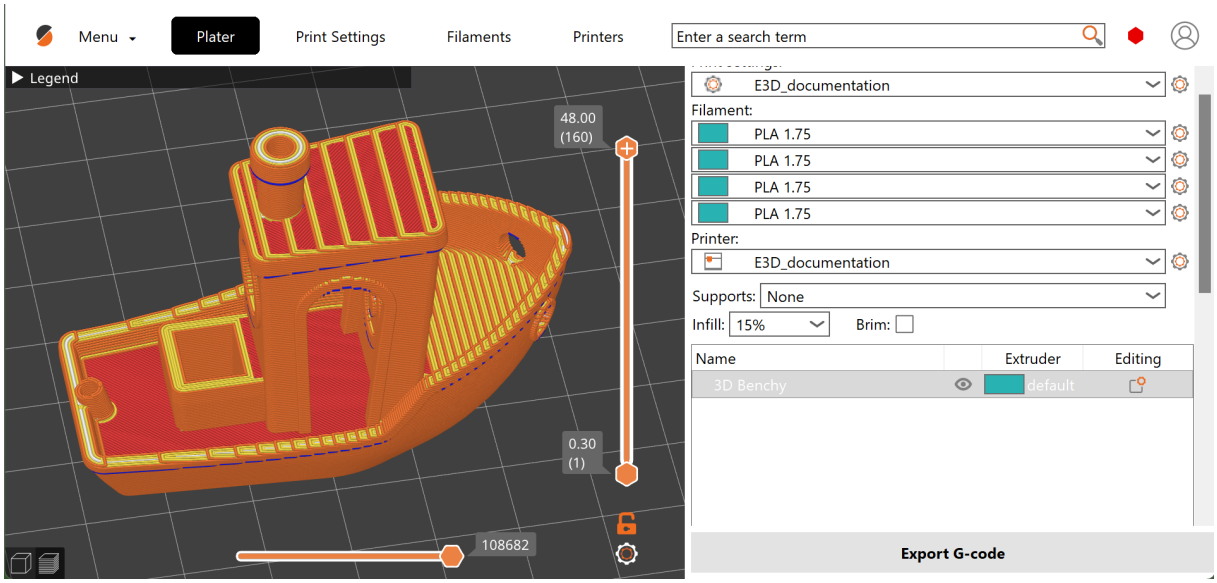


Figure 2.6 – The interface of the PrusaSlicer software, showing a sliced model with the generated toolpath. In the right panel, various presets can be selected. The individual settings can be adjusted through the tabs on the top side.

If a layer contains areas that overhang the layer below too much, the slicer automatically generates support structures underneath these areas to prevent sagging or collapse during printing. After all paths are generated, the slicer organizes them in an optimal order for printing. Each path is divided into segments, with each segment typically represented as a straight line. The slicer then calculates the volume and coordinates for each segment. Finally, these coordinates and instructions are translated into machine-readable commands called G-code.

G-code is a control language commonly used for machine tools in Computer Aided Manufacturing (CAM) processes such as Computer Numerical Control (CNC) milling, laser cutting, and 3D printing. It consists of a human-readable list of commands that instruct the printer on how to operate. Each command is represented by a letter followed by a number, along with optional parameters that specify details for that command. There are various dialects of G-code, tailored for different types of printers. Commands beginning with "G" are general instructions, such as movements, while those starting with "M" are miscellaneous instructions, like setting temperatures. For example, the command `G1 X100 Y100 Z0 F1000` directs the print head to move to the coordinates X100, Y100, Z0 at a speed of 1000 mm/min. Because G-code is highly machine-specific, it must be generated to match the requirements and capabilities of each particular printer model.

Printing

To begin printing, the generated G-code file is loaded into the printer. The printer processes the G-code sequentially, executing each command in the order it appears. The typical workflow starts with the printer homing its axes to establish a known reference position. Next, the printer heats the hotend and printbed to the temperatures specified for the chosen material.

Printing begins with the first layer, which is deposited directly onto the printbed by following the movement instructions in the G-code. The slicer has already calculated the precise amount of material required for each line. Subsequent layers are then printed on top of the previous ones, gradually building up the object until completion.

A typical print job consists of several hundred thousand individual commands. The printer operates without active feedback, relying solely on the instructions provided. After homing, it maintains its position by counting the steps sent to the stepper motors, without verifying its actual location.

2.1.3 FFF Printers

FFF printers have transitioned from costly industrial equipment to accessible consumer grade devices. Today, they represent the most prevalent type of 3D printer available to consumers. Every FFF printer consists of a motion system, which controls the movement of the print head, and an extrusion system, which is responsible for depositing the printing material. Figure 2.7 shows a schematic of a typical FFF printer.

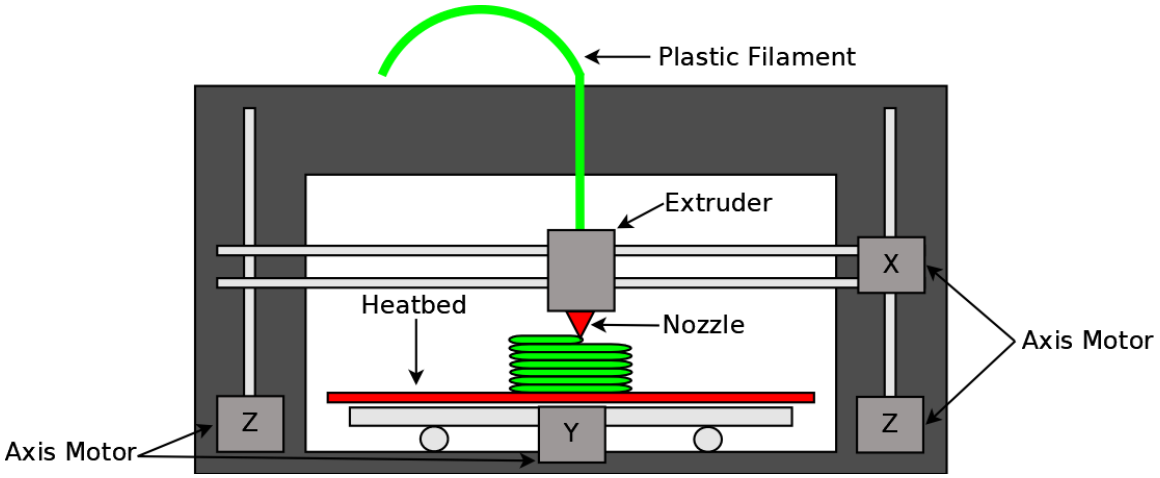


Figure 2.7 – Schematic of a typical Prusa i3 style FFF printer. The motion system with the frame, the motors and the rails moves the printhead in the X and Z direction, while the printbed moves in the Y direction. The extrusion system with the hotend, the extruder, and the heatbed are responsible for melting and depositing the filament onto the printbed.

Motion System

The motion system in a 3D printer controls the movement of the print head and printbed within three dimensional space. Typically, this system is composed of three orthogonal linear axes: X, Y, and Z. While alternative configurations such as delta or polar exist, they are relatively uncommon. This thesis primarily examines cartesian motion systems, which use three linear axes and, in some cases, additional rotary axes.

Each axis in the motion system is powered by a stepper motor. Stepper motors are commonly used because they offer high torque, are inexpensive, and are straightforward to control. While some printers use DC motors with encoders, these are uncommon. The X and Y axes typically use belts to convert the rotational motion of the stepper motors into precise linear movement. For the Z axis, a leadscrew is usually used, as it allows the stepper motor to reliably support the weight of the printbed or printhead against gravity. There are two common configurations for the motion system in 3D printers. In the first configuration, the printhead moves along the X and Z axes while the printbed moves along the Y axis. This setup is typical for printers such as the Prusa i3 series [W9] and the printer described in Section 3.3. In the second configuration, the printhead moves in the X and Y directions, and the printbed moves vertically along the Z axis, as seen in the Bambulabs X1 Series [W10] and the printer described in Section 3.1. For printers with XY moving printheads, a CoreXY [W11] mechanism is often used. In this design, the belts cross each other and no motor is mounted on a moving axis, which reduces the moving mass. This reduction in mass improves print quality and enables higher printing speeds.

In all of these configurations, the printhead can be positioned at any location within a rectangular build volume, which typically measures 200-300 mm along each axis. The movement of the stepper motors are driven by a microcontroller-based control board. The control board

sends pulses to the stepper motor drivers, enabling precise control of the printhead's position. The control board communicates with a computer, usually via USB or SD card. More recent control boards also support direct connections through Wi-Fi or Ethernet, allowing the printer to be operated via a web interface or directly from the slicer software.

Extrusion System

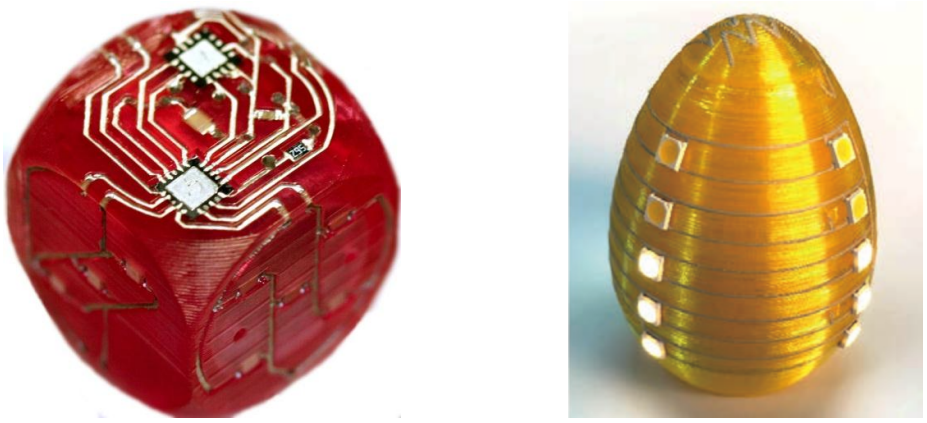
The extrusion system is the core component of the printer responsible for melting and depositing filament onto the printbed. Filament consists of a long strand of plastic wound on a spool, typically made from thermoplastics such as Polylactic Acid (PLA), Acrylonitrile Butadiene Styrene (ABS), Polyethylene Terephthalate Glycol (PETG), or Thermoplastic Polyurethane (TPU), with a standard diameter of 1.75 mm. The filament is pushed by a feeder motor into the hotend where it is heated and melted. The hotend is a metal block with a nozzle at its tip, heated by a heater cartridge. A thermistor is attached to the hotend to monitor its temperature. The control board continuously regulates the hotend temperature to maintain the value specified by the slicer. The nozzle itself has a conical shape with a small opening at the tip, allowing it to produce thin lines of extruded material. A nozzle diameter of 0.4 mm is commonly used, but nozzle sizes can range from 0.1 mm to 1 mm. The width of the extruded line can be adjusted by varying the amount of material extruded through the nozzle. Typically, lines with widths ranging from 50% to 200% of the nozzle diameter are considered printable.

The first layer of material is deposited onto a flat surface known as the printbed. The printbed is typically heated to improve the adhesion of the material and to minimize warping during printing. Before starting a print, the printbed must be carefully leveled to ensure the first layer is laid down evenly. If the bed is not level, some areas of the first layer may be pressed down too much, which can cause nozzle clogs, while other areas may not adhere properly, resulting in print failures. To further support proper adhesion, the printbed is often coated with a material such as Polyetherimide (PEI). These coatings also make it easier to remove the finished print, as they lose their adhesive properties once cooled.

2.2 Printed Electronics

3D printed objects are usually passive objects. They can be used as prototypes, tools, or decorative objects, but they are not able to interact with their environment. To enable objects to interact with their environment, electronic components must be incorporated directly into the object itself. Traditionally, this is achieved by embedding prefabricated Printed Circuit Boards (PCBs) into the object.

The concept of printed electronics is to combine the electronics fabrication directly within the 3D printing process. This is a revolutionary advancement in 3D printing and electronics, allowing complex 3D structures with embedded electronics. Electronic wires are printed



(a) **(b)**

Figure 2.8 – **(a)** A Cubesat with integrated printed electronics. [31] **(b)** A fully 3D printed egg-shaped timer with integrated electronics. [32]

with conductive material onto the substrate and electronic components like sensors, resistors, or Integrated Circuits (IC) are added. The goal is to create a fully functional object with integrated electronics in one process without the need for assembly. This allows for new device architectures that are lighter, flexible, compact, or not possible with traditional manufacturing methods. Also, it allows shorter time to markets for new products, have a shorter assembly time, and the process is capable for mass product customization without expensive tooling [30]. MacDonald et al. [17] showed that a benchmark circuit designed by the National Aeronautics and Space Administration (NASA) Johnson Space Center (JSC) can be printed with a volume reduction of 27% compared with traditional manufacturing methods. The first 3D printed electronic circuits where created by printing mold for a circuit and injected a metal that melts at 70° with a syringe into the 1.2 mm wide cavities to create a small mobile robot [9].

3D printed electronics are also used to add functionality to traditional fabricated objects by printing just the electronics onto them. This approach saves time compared to traditional assembly and enables features such as eliminating separate PCBs, additional wiring, and integrating sensors directly onto the object's surface. However, these circuits and components are limited to the surface of the object and remain visible unless covered in a subsequent manufacturing or assembly step.

3D printed electronics enable a wide range of applications by integrating electronic functionality directly into objects. For example, Gutierrez et al. [31] and Shemelya et al. [33] demonstrated that complex circuits and sensors can be fabricated directly onto the structure of a Cubesat, improving its functionality while keeping weight and size to a minimum (Figure 2.8a). Ankenbrand et al. [32] developed a fully printed egg-shaped timer with integrated electronics produced in a single manufacturing process (Figure 2.8b). De Nava et al. [34] created a 3D printed magnetic flux sensor. Hossain et al. [35] incorporated sensors into a combustion engine using 3D printed electronics. Glasschröder et al. [36] integrated strain gauges into robotic grip-

pers to measure the force applied to objects. Most of these applications incorporate traditional electronic components into 3D printed objects, but it is also possible to print the components themselves directly. For example, researchers have demonstrated 3D printed transistors [37], capacitors [38], and RFID tags [39]. 3D printed electronics can also be used to fabricate sensors directly onto objects, such as multi-axial force sensors [40], pressure sensors [41], and tactile sensors [42]. Yang et al. [43] developed a 3D printed solar cell with integrated electronics. It is also feasible to print batteries [44], including batteries designed to serve as structural components [45]. These examples illustrate the broad range of applications and capabilities enabled by 3D printed electronics technology.

There are several challenges in printed electronics that must be overcome for 3D printed electronics to become a viable alternative to traditional electronics. One major challenge is the limited selection of printable materials. Conductive inks are commonly used to print circuit traces, but these inks are expensive and have a negative environmental impact. Additionally, most conductive inks require a post-printing sintering process to achieve electrical conductivity. Many low-resistance inks need to be sintered at high temperatures, which is incompatible with most 3D printed substrate materials. Inks that can be sintered at lower temperatures generally have higher resistivity.

Another challenge is the tradeoff between scalability and resolution in most printing processes. Processes that achieve high resolution tend to operate slowly, resulting in low throughput and making them unsuitable for mass production. Conversely, processes designed for higher speed and throughput typically produce lower resolution prints, which are inadequate for creating fine traces. As a result, there is a dilemma: high-resolution processes are limited in scalability, while scalable processes cannot achieve the fine detail required for certain applications.

A major challenge is ensuring the reliability of printed electronics. Printed electronics tend to be more prone to failure compared to traditional electronics. The manufacturing process must be carefully controlled and tested to guarantee consistent product quality. Additionally, when electronics are embedded within objects, they become much more difficult to inspect and are nearly impossible to repair. As a result, quality control must be integrated into the production process itself to detect and address issues as they arise.

Lastly, the cost effectiveness of printed electronics compared to traditional electronics remains a significant challenge. Traditional electronics manufacturing is a mature industry characterized by high levels of automation and throughput. In contrast, the automation of 3D printed electronics is still in its early stages, and the process cannot yet match traditional methods in terms of cost, production speed, or reliability. Even with ongoing advancements, 3D printed electronics are currently not competitive with conventional fabrication for large-scale production. As a result, their use is best suited to specialized applications where the unique capabilities of 3D printed electronics or a high degree of customization are required.

2.2.1 Processes for 3D Printed Electronics

There is a wide range of processes for applying conductive material onto a substrate. These processes are primarily distinguished by the method used to deposit the conductive traces. Each process has its own specific capabilities, is suited to particular applications, and is typically combined with different 3D printing techniques for fabricating the substrate.

Direct Writing

In direct writing, a conductive material is directly deposited onto a substrate to form a conductive trace. This can be done with a motorized syringe [47], a worm drive based extruder [48] or a pressure based extrusion system [18]. The conductive material is deposited directly onto the surface using a needle or nozzle. Typically, this material is a conductive paste or ink composed of silver nanoparticles suspended in a solvent, or carbon nanotubes. In earlier approaches, low melting point metals were used to form the traces [9]. Because the material is applied directly to the surface, it is essential to maintain a precise distance between the nozzle and the substrate. The surface tension of the conductive material helps bridging small gaps, but it makes it difficult to produce fine lines. This can result in the formation of blobs, which may cause electrical shorts. The direct writing process can be integrated into MEX printers [47]. Lopez et al. [49] integrated a direct writing system into an SLA printer by using a syringe to dispense conductive material directly into the resin bath, then hardening it with a laser. Figure 2.9a shows a schematic of the direct writing process.

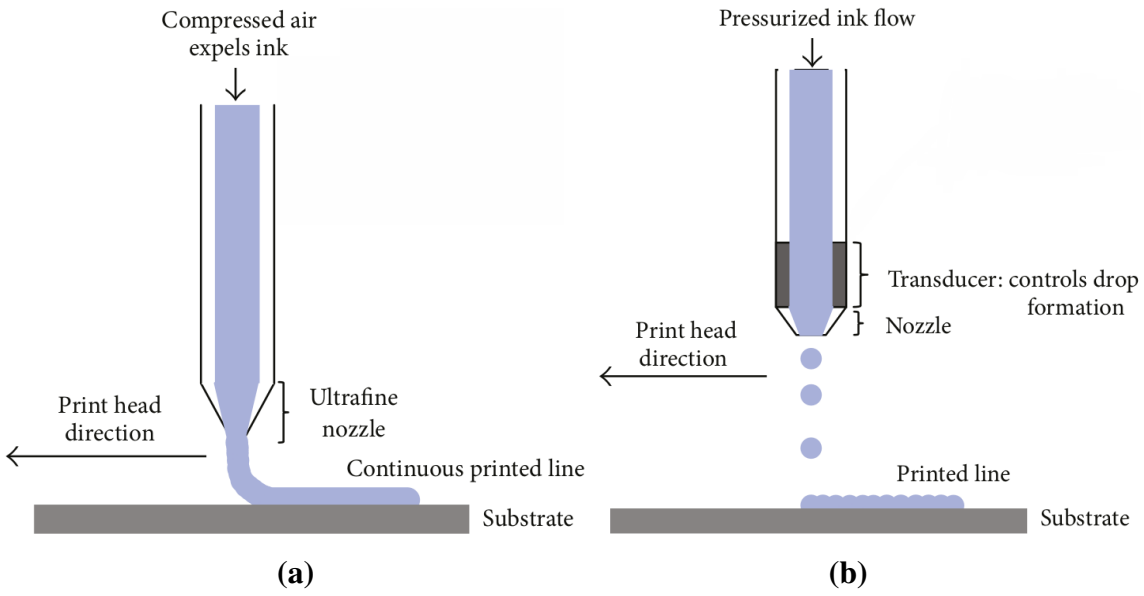


Figure 2.9 – **(a)** In the direct writing process, a conductive material is deposited directly onto the substrate using a nozzle. **(b)** In the inkjet printing process, an inkjet printhead uses either a piezoelectric crystal or a heater to generate pressure, which propels droplets of ink onto the substrate. [46]

Inkjet Printing

Another method for applying conductive silver nanoparticle ink is contactless inkjet printing [50]. In this process, an inkjet printhead uses either a piezoelectric crystal or a tiny resistor that heats up to generate pressure, which propels controlled droplets of ink onto the substrate, similar to the operation of a standard paper inkjet printer. These droplets are deposited sequentially in a line, forming a conductive trace with a consistent width. The inkjet printhead enables the creation of fine lines. Because the process is contactless, it can be used to print on surfaces where the exact distance to the nozzle may vary slightly. However, since the ink droplets lack substantial surface tension, bridging small gaps during printing is challenging. Figure 2.9b shows a schematic of the inkjet printing process.

Aerosol Jet

For even finer lines, the aerosol jet printing process can be used [51] . Aerosol jet printing creates an aerosol mist of ink by atomizing the ink into fine droplets, which are then carried by a gas stream through a deposition nozzle. This focused stream of aerosol is directed onto a substrate, to create a fine line of ink. Often, multiple passes are needed to build up the desired thickness and conductivity of a printed trace. Because Aerosol Jet printing produces extremely fine lines as small as 25 µm, the process is highly sensitive to the surface quality of the substrate. Any surface roughness or gaps can negatively affect the printed traces. Therefore, it is essential that the substrate surface is very flat and smooth. For this reason, Aerosol Jet printing is typically combined with 3D printing processes that produce smooth surfaces, such as PBF [52], or with prefabricated objects produced by VPP printing. Figure 2.10a shows a schematic of the aerosol jet printing process.

Wire Embedding

Another approach for creating conductive traces is to embed prefabricated wires directly into the 3D printed object. This method results in traces with significantly lower resistance and greater mechanical strength compared to other techniques. The improved conductivity is particularly advantageous for power distribution within the object, as it exceeds the performance of other printed electronics processes. One approach to embed the wires is to insert the wires after printing by heating them with an electric current and pressing them into the underlying substrate [53]. Alternatively, the substrate itself can be heated using a laser or ultrasonic energy source, and the cold wire is then pressed into the softened material [54]. Another method is to embed the wire during the printing process by integrating a wire embedding system into the FFF printer head, allowing the wire to be surrounded by the printed material as the object is built [55]. To connect prefabricated wires within a 3D printed object, the wires must be automatically cut to the correct length when connecting components [53], or at interconnects and intersections [55]. A common

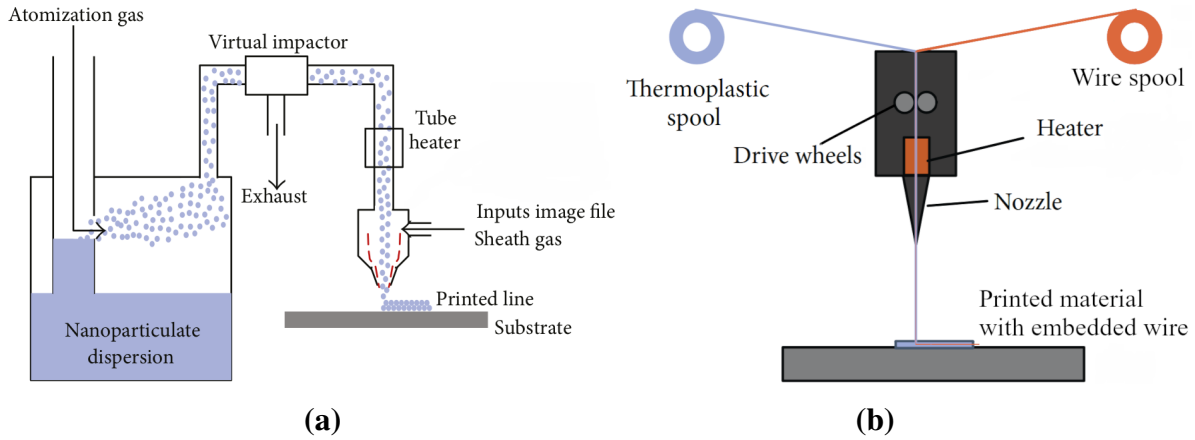


Figure 2.10 – **(a)** In the aerosol jet printing process, an aerosol mist of ink is generated and directed onto the substrate to create fine lines. **(b)** In the wire embedding process, a prefabricated wire is inserted into the substrate to create conductive traces. [46]

challenge in wire embedding is the precise placement of the wire, as wires tend to take the shortest path and may deviate from the intended route, especially around corners. This issue can be addressed by applying a geometric correction model that compensates for the wire's diameter and position during the embedding process [56]. Figure 2.10b shows a schematic of the wire embedding process.

Conductive Filament

For FFF printers, conductive filament is available for printing conductive traces. This method is inexpensive and does not require any specialized equipment, as the filament can be extruded using a standard FFF printer nozzle. However, the resulting traces have much lower conductivity compared to those produced by other processes. Conductive filaments are produced by blending conductive particles, such as carbon black or graphene, into a thermoplastic base material. Filaments containing carbon black filler typically achieve a resistivity of approximately $0.1\,\Omega\,\mathrm{m}$ [57]. Lower resistivity can be obtained by using filaments made from low-melting-point metal alloys, such as Sn70Bi30 or Sn66Bi30Ag4, which have resistivities of 257 $\mathrm{n}\Omega\mathrm{m}$ and 230 $\mathrm{n}\Omega\mathrm{m}$, respectively [58]. However, printing with metal alloy filaments is challenging and often unreliable.

Laser Direct Structuring

Laser direct structuring is a process that is close to traditional PCB manufacturing. In this method, an object is made from a special plastic that can be activated by a laser. The laser selectively activates specific regions of the material, and an electroless plating process is then used to deposit copper onto these activated areas wile the rest of the surface remains uncoated. This approach enables the creation of complex circuits with high resolution. Since the metal

plating occurs as a separate step after activation, this process is limited to producing circuits only on the surface of objects.

2.2.2 Toolpath Generation

All of the processes described above require specialized toolpath generation to control the printer. Typically, the toolpath for a printed wire consists of a line that is deposited onto the substrate. This approach is similar to the toolpath generation used in MEX printers, but it involves additional constraints and requirements that vary depending on the specific process.

When integrating SMD components into a 3D printed object, the toolpath generation must also account for their placement and incorporation. Additionally, the design of electronic circuits, typically performed in Electronic Computer-Aided Design (ECAD) software, must be included in the process [59]. Krebs et al. [60] used the commercial tool NEXTRA, a 3D CAD system that combines mechanical and electrical CAD, to design electronics on non-flat surfaces. This tool enables routing of electronic traces on the surface of 3D objects, but does not provide path planning or printing functionalities. MacDonald et al. [17] used standard PCB ECAD software to design circuits, then wrapped the board layout around simple surfaces such as a die [17]. Swensen et al. [61] processed ECAD board layout files with a Matlab script to generate hollow channels, which were later filled with low-temperature molten conductive alloy. Autodesk developed specialized software called Project Wire [W12], which integrated ECAD design with printer toolpath generation. This software enabled the placement of components, generation of cavities, and routing of wires both through and on the object. However, Project Wire was discontinued after a few years and is no longer available. In parallel, Wasserfall et al. [47] extended the Slic3r FFF slicing software to add toolpath generation for 3D printed electronics. Their software imports the netlist from ECAD tools and generates the printers toolpath, including component placement, cavity generation [22], and topology-aware automatic wire routing [25]. A similar approach was developed by Bailey et al. [62], who created a SolidWorks plugin combined with a Cura plugin. The SolidWorks plugin imports ECAD schematics and generates cavities and traces, while the Cura plugin produces the printers toolpath. The commercial software Motion 3D, developed by Neotech AMT [W13], is used to create toolpaths for 5-axis printing of electronic circuits [32]. Based on the Pictures CAD/CAM software [W14], it enables toolpath generation for 5-axis printing of electronic wires using different processes and supports component placement. Figure 2.11 shows a screenshot of the Motion 3D software. This tool allows the generation of toolpaths for 5-axis printing of electronic wires, but without the integration of components, cavities, or routing.

Several software solutions exist for generating toolpaths for 3D printed electronics, each offering distinct advantages and disadvantages. An ideal software would support importing or designing ECAD schematics, generating printer toolpaths, and integrating component placement, cavity creation, and wire routing. It should also be capable of producing toolpaths for

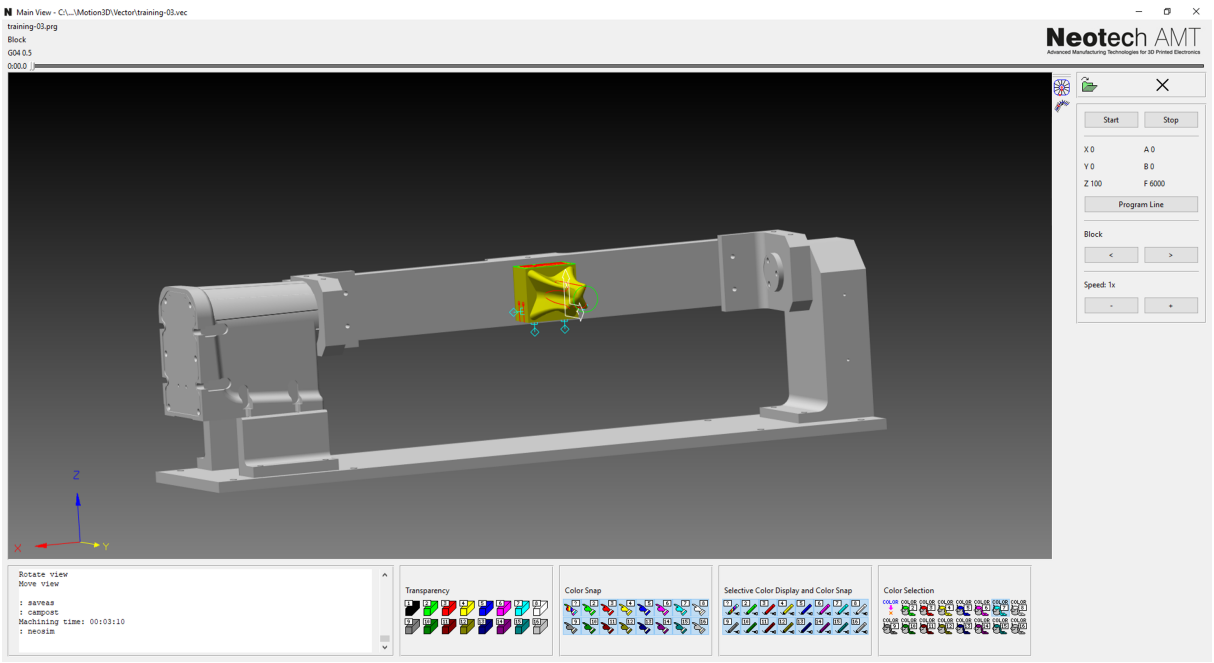


Figure 2.11 – A screenshot of the toolpath planning software Motion 3D by Neotech AMT, which is used to generate toolpaths for 5-axis printing of electronic circuits. [W13]

various processes and printers, including both 3-axis and 5-axis motion systems. At present, no software solution exists that fulfills all these requirements. As a result, toolpath generation for 3D printed electronics continues to be an active area of research.

2.2.3 Classification of 3D Printed Electronics

The Fachverband Elektronikdesign und -fertigung e.V. (FED) proposes a classification system for 3D printed electronics organized by the degree of electronic integration within the printed object [63]. This system provides a common framework to categorize approaches and techniques in 3D printed electronics. The classes are hierarchical-each builds on the previous one-and describe integration levels rather than specific processes or materials. The classes are defined as follows:

Class 1 In this class, conductive traces are printed onto the surface of a prefabricated, planar substrate. The resulting circuit is two-dimensional, comparable to traditional PCB-style layouts produced on materials such as metal, fabric, or plastic. Figure 2.12a illustrates a class 1 printed circuit on a flat metal substrate.

Class 2 This class involves creating a circuit on an existing three-dimensional object. The circuit can be folded or wrapped around the object, yet its layout remains mainly planar. If components are added, they are placed only on the object's surface. Figure 2.12b illustrates a circuit wrapped around a cylindrical object.

(a) **(b)** **(c)** **(d)**

Figure 2.12 – The four realized classes of 3D printed electronics as defined by the FED. **(a)** A planar printed class 1 circuit on a substrate. **(b)** A class 2 circuit wrapped around a cylinder. **(c)** Schematic of a class 3 layer-based printing process for a 3D circuit. **(d)** A fully printed class 4 light bulb with integrated SMD components. Class 5 has not yet been realized in the literature. [63]

Class 3 In this class, the substrate and the conductive traces are printed together in a single process. This enables true 3D circuits with freeform wire paths that can extend in any direction. Complex internal structures can be fabricated, allowing electronics to be integrated within the object's volume. Figure 2.12c shows a schematic of a layer-based printing process for a class 3 circuit.

Class 4 This class extends the integration to include additional SMD components within the printed object. These components are automatically placed during the process and are contacted by the printed conductive traces. This enables highly complex circuits in minimal space. Figure 2.12d shows an example of a fully printed light bulb with integrated electronics.

Class 5 This class incorporates 4D printing capabilities into the object, allowing it to change shape or material properties over time or in response to external stimuli such as heat or mechanical force. Such behavior may be achieved through printed mechanical features that enable the object to adapt or transform after fabrication. While 4D printing is an active research area, its combination with integrated electronics has not yet been demonstrated in the literature.

2.3 Segmentation Networks

In the segmentation tasks presented in Chapter 5 and Chapter 6, neural networks [64] are employed to segment the images. These networks are specifically trained to perform semantic

segmentation, which involves assigning a class label to every pixel in an image. This approach is widely used in computer vision to analyze images at the pixel level.

2.3.1 U-Net

The U-Net architecture is a widely used Convolutional Neural Network (CNN) model for semantic segmentation. Originally introduced by Ronneberger et al. [65] for biomedical image segmentation, it has become popular in many domains because of its effectiveness and adaptability. U-Net features a symmetric encoder-decoder structure: the encoder extracts features from the input image, while the decoder reconstructs the segmentation map using these features. The architecture proposed in the original paper is illustrated in Figure 2.13.

The encoder in the U-Net architecture is composed of several steps, each containing two convolutional layers with a Rectified Linear Unit (ReLU) activation function, followed by a max-pooling layer. With each step, the spatial resolution of the feature maps is halved, while the number of feature maps is doubled. This design enables the network to learn both fine details and broader contextual information. A key feature of U-Net is the use of skip connections: the output from each encoder layer is concatenated with the corresponding decoder layer. These

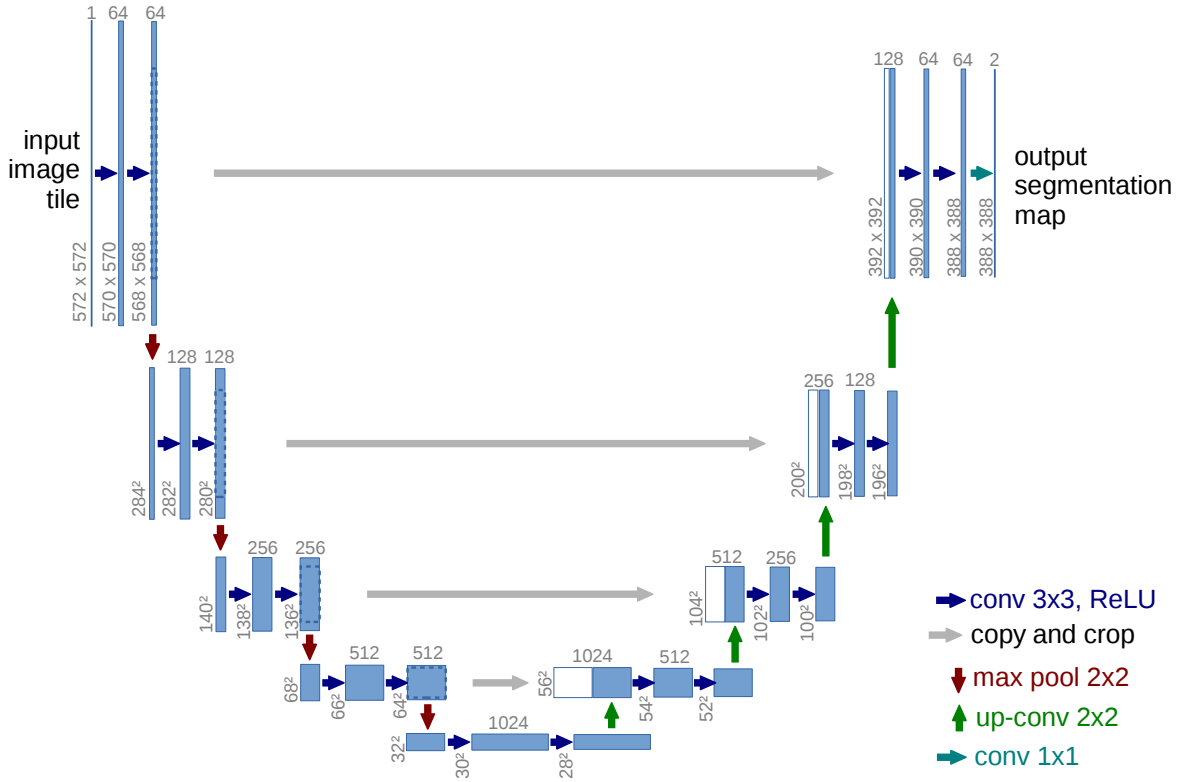


Figure 2.13 – U-Net architecture for semantic segmentation. The encoder extracts the features from the input image, and the decoder reconstructs the segmentation map from the features. [65]

skip connections help preserve spatial information that might otherwise be lost during the downsampling process. At the deepest part of the network, known as the bottleneck, the convolutions have the lowest spatial resolution and the highest number of feature maps. After the bottleneck, the decoder begins to reconstruct the segmentation map by upsampling the feature maps back to the original image size. The decoder consists of a series of transposed convolutional layers, each of which doubles the spatial resolution and halves the number of feature maps. After each upsampling step, the feature maps from the encoder are concatenated with those from the decoder, and the combined maps are processed by two convolutional layers with a ReLU activation function. This process is repeated until the feature maps reaches the original image size. Finally, a 1x1 convolutional layer is applied to produce the segmentation map, with the number of output channels corresponding to the number of classes.

The U-Net architecture is suitable for both single-class and multi-class segmentation tasks, meaning that each pixel in the segmentation map is assigned a probability for every class. To generate the final segmentation map, a softmax activation function is applied to the output of the last layer, so each pixel is classified according to the highest probability among the available classes. U-Net models are commonly used in applications such as medical imaging, remote sensing, and other fields where accurate pixel-level segmentation is required.

2.3.2 U-Net++

The U-Net++ architecture, introduced by Zhou et al. [66], builds upon the original U-Net design to improve segmentation accuracy while managing model complexity. Figure 2.14 illustrates the U-Net++ architecture as presented by the original authors.

The U-Net++ architecture enhances the original U-Net by introducing nested dense skip connections, each containing additional convolutions. After every encoder block, an upsampling path is created that connects to the skip connection of the corresponding upper block. These skip connections are concatenated with the feature maps from the encoder block as well as all previous skip connections, resulting in a densely connected structure. This design improves feature reuse and improves the flow of information between the encoder and decoder. The architecture of the individual blocks remains unchanged from U-Net. On the final upsampling path, each iteration is concatenated with all previous skip connections, enabling more precise segmentation along object boundaries and helping to bridge the semantic gap between the encoder and decoder. However, this dense connectivity increases the number of parameters and computational requirements compared to the original U-Net. The original authors also proposed a deep supervision strategy, in which the network is trained with multiple outputs at different stages of the decoder, as indicated by the red connections in Figure 2.14. Deep supervision was not used in this work.

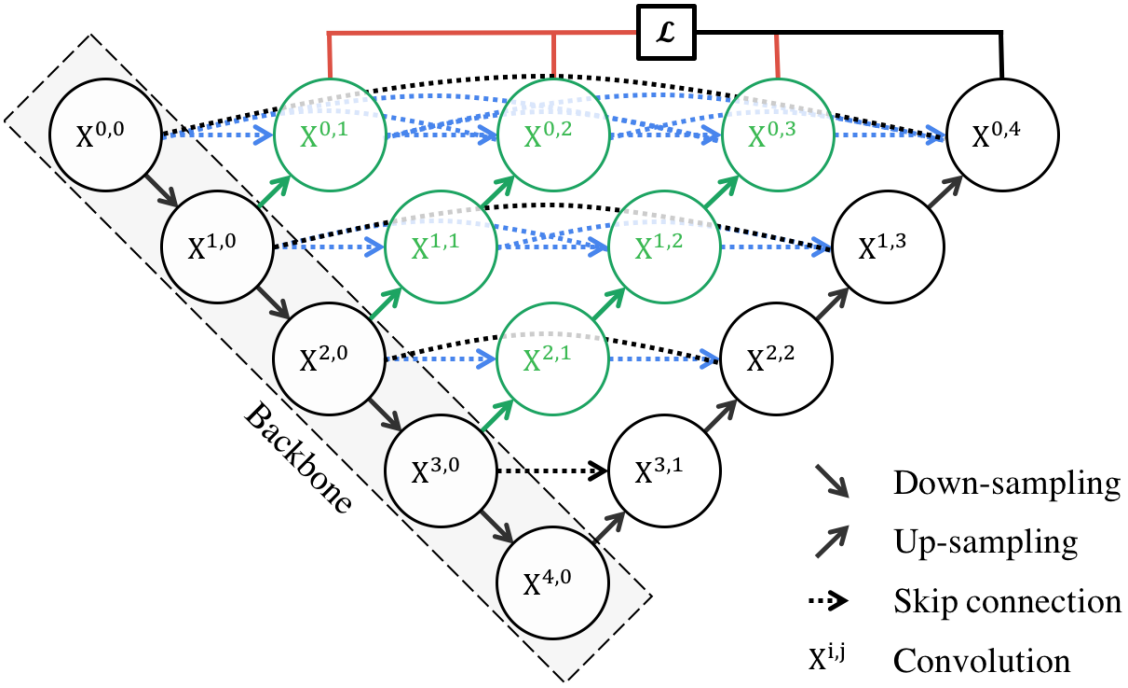


Figure 2.14 – U-Net++ architecture for semantic segmentation. The nested skip connections help to bridge the semantic gap between the encoder and decoder. [66]

2.3.3 DeepLabV3+

DeepLabV3+ is an architecture for semantic segmentation proposed by Chen et al. [67] from Google Research. It is based on previous DeepLab models and introduces several enhancements to improve segmentation performance. DeepLabV3+ is initially designed for semantic segmentation tasks in scene understanding and object detection. Figure 2.15 shows the DeepLabV3+ architecture as proposed by the original authors. The architecture is separated into two main components: the encoder and decoder. The encoder takes the input image and extracts the features using a backbone network like ResNet [68] or Xception [69]. These backbone networks are great for understanding where things are in the image and preserving details. Afterwards, the Atrous Spatial Pyramid Pooling (ASPP) module is applied where Atrous Convolutions with different dilation rates are used to capture multi-scale contextual information. The different scales are concatenated and passed through a 1x1 convolutional layer. This stage allows the network to understand what things are in the image without losing spatial resolution due to downsampling. On the decoder side, the output of the backbone is fed through a 1x1 convolutional layer to reduce the number of channels. The output of the Atrous Convolutions is upsampled by a factor of 4 to match the output size of the backbone network. Both outputs are concatenated and processed with a 3x3 convolutional layer and are upsampled to the original image size to get the final segmentation map. The DeepLabV3+ architecture is designed to handle objects at

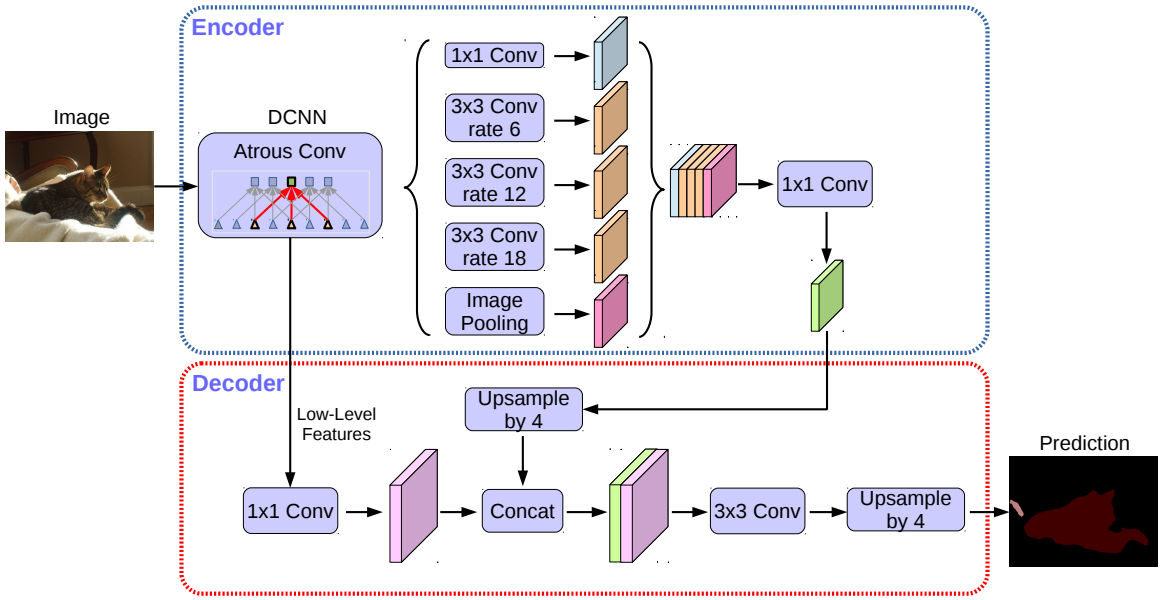
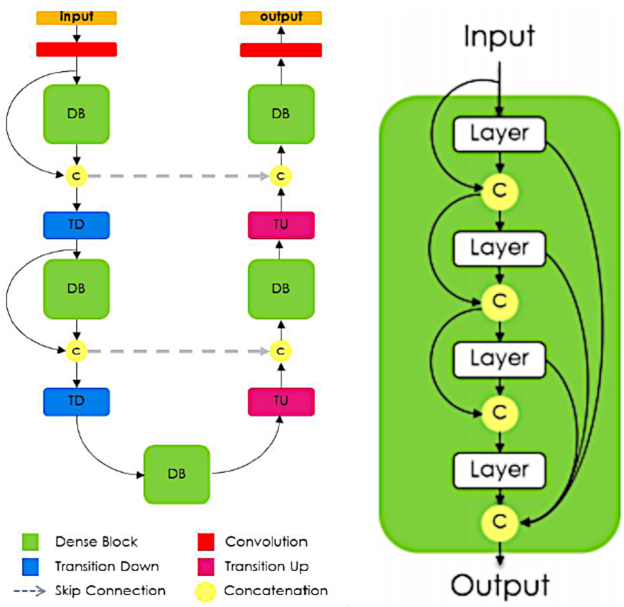


Figure 2.15 – DeepLabV3+ architecture for semantic segmentation. The DCNN backbone extracts features from the input image, the Atrous Spatial Pyramid Pooling extracts multi-scale contextual information, and the decoder reconstructs the segmentation map. [67]

multiple scales and complex scenes. It is particularly effective in scenarios where objects vary significantly in size and shape, making it suitable for applications such as autonomous driving and urban scene understanding.

2.3.4 DenseNet

The DenseNet architecture is a convolutional neural network originally introduced for image classification by Huang et al. [71], and later adapted for semantic segmentation by Jegou et al. [70]. Its overall structure closely resembles the U-Net architecture, but it differs in the design of its convolutional blocks. Figure 2.16 illustrates the DenseNet architecture as presented by the original authors. The convolutional blocks in DenseNet are known as dense blocks and consist of multiple convolutional layers. In each dense block, every layer receives as input the concatenated feature maps from all previous layers within that block. This means that each layer outputs a fixed number of feature maps, and the outputs of all layers in the dense block are concatenated together. On the encoder side, the output of each dense block is concatenated with its input. To reduce the spatial resolution in the encoder, each dense block is followed by a transition layer, which consists of a 1x1 convolutional layer and a 2x2 average pooling layer. Unlike U-Net, DenseNet does not double the number of feature maps at each step. Instead, the number of feature maps increases by a fixed amount with each layer, resulting in a gradual growth in the number of feature maps as the network goes deeper. The bottleneck of the network is also



(a) **(b)**

Figure 2.16 – **(a)** DenseNet architecture for semantic segmentation. The encoder extracts features from the input image, and the decoder reconstructs the segmentation map. **(b)** The DenseNet building blocks. Each layer in a dense block receives the same input as the previous layer by concatenating the feature maps after each convolution. [70]

implemented as a dense block and is followed by the decoder. In the decoder, the input is upsampled to increase the spatial resolution and concatenated with the corresponding skip connection from the encoder. The feature maps are then processed by a dense block, followed by another upsampling transition layer. In the decoder, the number of feature maps decreases because, unlike in the encoder, the input is not concatenated with the output of the dense block. At the end of the network, a 1x1 convolutional layer is applied to produce the final segmentation map, with the number of output channels corresponding to the number of classes in the segmentation task. The DenseNet architecture is designed to improve feature reuse and reduce the total number of parameters in the network. While DenseNets often require fewer parameters than a U-Net of similar depth, their more complex structure and extensive use of concatenations result in higher memory requirements compared to U-Net.

Chapter 3

Hardware

During the research for this thesis, two different 3D printers were modified and used. Both are based on the E3D ToolChanger platform and were extensively adapted to meet the project requirements. The first system is a three-axis motion platform intended for planar electronics printing and in-process monitoring. It was used in the studies presented in Chapter 5 and also in Chapter 6. The final results of Chapter 5 were additionally evaluated on the commercial Neotech PJ15X printer from project partner Neotech AMT GmbH during the KamEl project. That machine is a fully additive 3D electronics printer with a five-axis motion system. The second modified E3D ToolChanger is a five-axis motion platform designed for printing 3D conformal electronics and plastics. It was employed in the research described in Chapter 4. The following sections describe the individual hardware platforms used in the projects in more detail.

3.1 Three Axis E3D ToolChanger

The E3D ToolChanger [W15] is a modular 3D printer platform built for multi-tool operation. It is designed for easy adaptation, making it well suited for research and development in 3D printing. The platform relies on open-source hardware and software, enabling customization and modification. Its key advantage is the integrated tool changing system, which can switch between up to four interchangeable toolheads. These tools are parked at the back of the printer and are individually picked up via a kinematic coupling mechanism. This coupling provides an accuracy of smaller than 1 µm during every tool change. E3D supplies an FFF toolhead in both Bowden and direct drive variants. For creating additional tools, mounting plates are provided so custom tools can be attached to the printer. The tool pickup head includes a fixed switch used for homing the Z axis and leveling the print bed. Because this switch is covered when a tool mounting plate is engaged, it can only be used with no tool mounted. The printer is shown in Figure 3.1 with four different tools mounted.

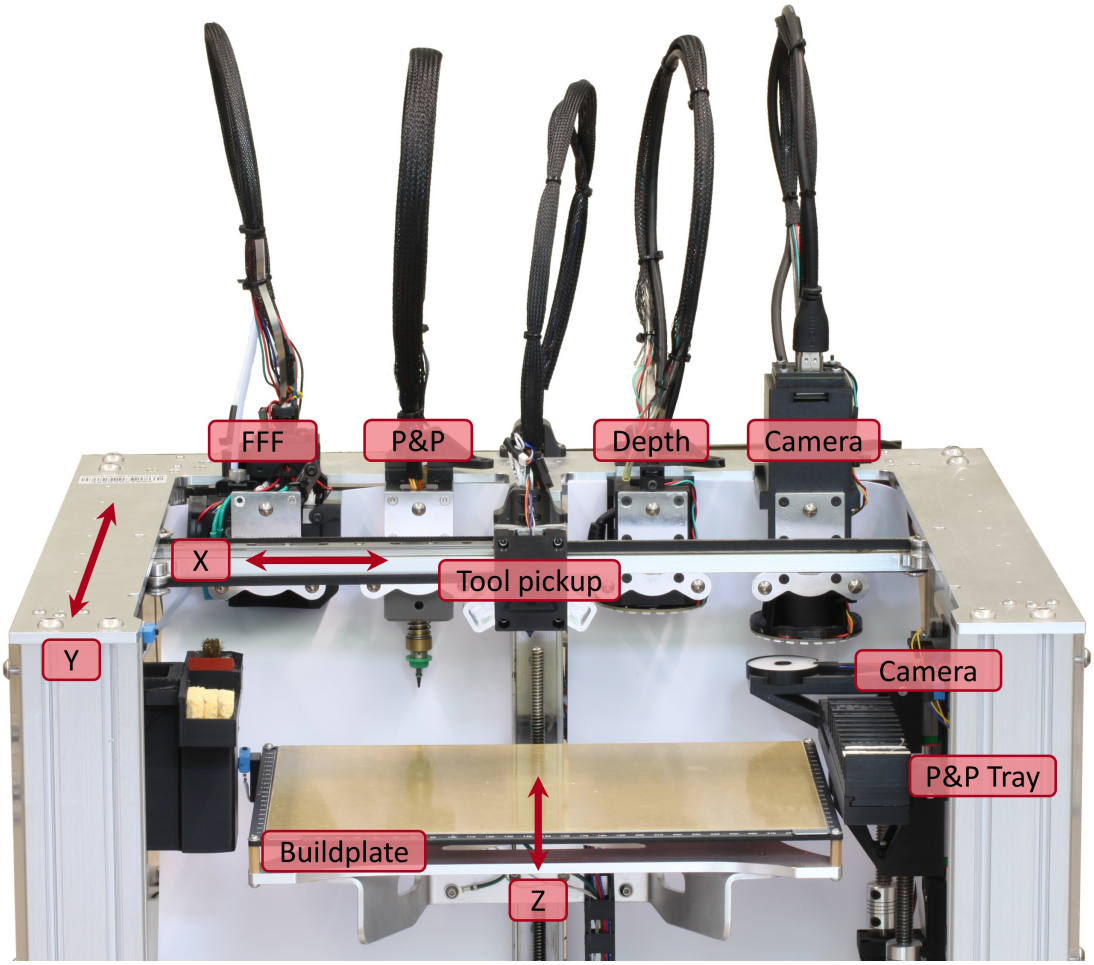


Figure 3.1 – The modified three-axis E3D ToolChanger features a tool pickup head that is capable of holding multiple tools. The four tools, an FFF toolhead, a pick-and-place tool, a depth camera, and an RGB camera, are parked at the rear of the printer. The component tray and the calibration camera are mounted on the right side of the printer.

3.1.1 Motion System

The E3D ToolChanger uses a Cartesian three-axis motion platform with CoreXY kinematics. All axes are driven by stepper motors with 200 steps per revolution (1.8°per step) and run on linear rails. With 6.25 full steps per mm and 256 microsteps between each full step, the theoretical resolution is below 1 µm. In practice, the motion system has a repeatability and accuracy of ~10 µm. The Z axis is driven by a leadscrew and guided on a linear rail. The heated printbed is mounted on the Z axis and has a size of 300 × 200 mm. This results in a build volume of 300 × 200 × 300 mm with an overall axis travel of 363 × 280 × 325 mm, leaving space for additional tools and sensors with full access to the print bed. Excluding the motors integrated into the tools, the E3D ToolChanger has four stepper motors. The first two move the X and Y axes, the third drives the Z axis, and the fourth locks the tool in place. Most tools have an additional stepper motor to drive the tool itself, depending on the tool. Additional stepper motors were

added for the pick and place hardware and for the calibration camera height adjustment. This is described in more detail in Section 3.1.3.

3.1.2 Control Board

The entire printer is managed by a Duet 3 [W16] control board. The Duet 3 is a 32-bit controller intended for both 3D printers and CNC machines. It uses an ARM Cortex-M7 microcontroller and a CAN bus architecture to communicate with expansion boards and other peripherals. The base Duet 3 board provides six stepper motor drivers. Because the proposed machine needs more, two Duet 3HC expansion boards are added, supplying six additional drivers plus extra I/O. These I/O ports connect endstops, sensors, heaters, fans, and other peripherals. The entire system is operated via a web interface that enables straightforward configuration and monitoring. Simple scripts can be defined to automate tasks and control printer functions. Figure 3.2 shows a schematic of the control board and its connections to the various printer components.

A Raspberry Pi running OctoPrint [W17] manages the printing process. The slicer's generated G-code file is uploaded to OctoPrint, which then sends the individual G-code commands via the serial connection to the Duet 3 control board. OctoPrint also supports installing additional plugins to extend its functionality. In this research, two plugins are used to extend the printer's functionality.

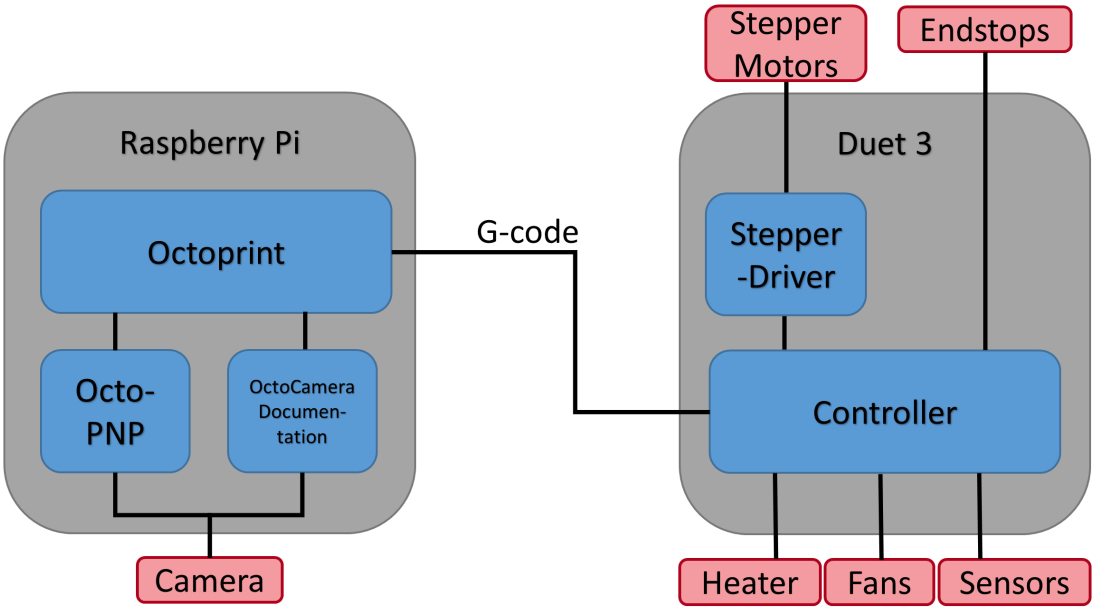


Figure 3.2 – Schematic of the control board and the connected OctoPrint. The OctoPrint (left) sends G-code commands to the Duet 3 via a serial connection and manages the camera. The Duet 3 mainboard (right) controls the stepper motors via dedicated motor drivers, endstops, heaters, fans, and sensors.

The OctoPNP [W18] plugin implements the pick and place workflow [47]. It monitors the G-code stream sent to the printer and pauses it whenever a component must be placed. At that point, the plugin takes control of the printer to execute the placement routine. During this routine it performs the required motion sequences and captures images of the component. It analyzes these images to determine the component's center and rotational orientation, then places the component onto the printed part.

The OctoCameraDocumentation [W19] plugin is used to document the printing process [24]. It monitors the G-code and, when a layer is finished, triggers image capture. To cover the entire layer, it takes multiple images and stitches them together to create a single complete image of the printed layer. This process is described in more detail in Section 5.2.

3.1.3 Toolheads and Accessories

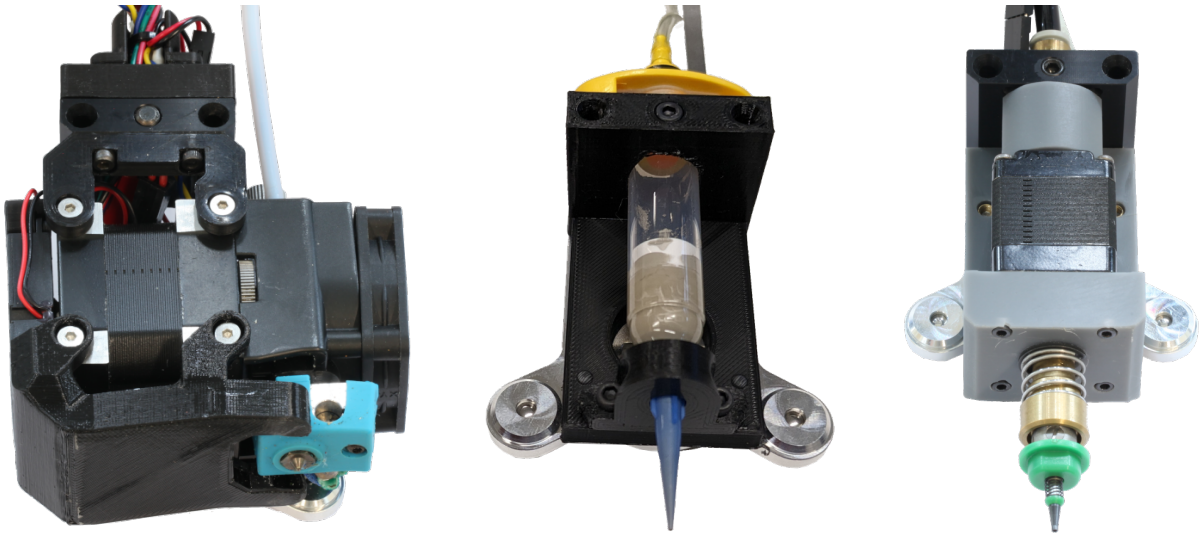
Besides the standard FFF toolhead, several additional custom toolheads and accessories were designed and built for the E3D ToolChanger. These comprise a conductive printhead for wire deposition, a pick and place tool, an in-process monitoring camera, a depth camera for printed layer measurement, and further supporting accessories. The following sections describe each of these toolheads and accessories in more detail.

FFF Printhead

The FFF printhead is the standard toolhead supplied by E3D. It is based on the E3D Hemera extruder, a compact, lightweight direct-drive unit with a dual gear mechanism that grips the filament from both sides. In this direct drive design, the stepper motor that moves the filament forward is mounted directly on the printhead. The filament path is kept very short to reduce the risk of jams and to improve retraction performance. This configuration supports printing with a wide range of materials, including flexible filaments. The hotend uses a standard 0.4 mm nozzle with a cartridge heater and a thermistor for temperature measurement. The heatbreak is integrated directly into the extruder drive assembly. This integration reduces the required space and results in a more compact overall design. The system uses 1.75 mm filament and supports a maximum nozzle temperature of 300 °C.

Conductive Printhead

To print conductive traces, a dedicated conductive printhead was developed for mounting on the E3D ToolChanger. This printhead uses a direct writing process, dispensing conductive ink via a pressure-driven system. Pressure is supplied by a small compressor and adjusted manually using a pressure regulator. Ink flow is controlled by a solenoid valve, which switches the pressure on and off. Both the compressor and valve are operated by the Duet 3 control board through G-code commands.



(a) **(b)** **(c)**

Figure 3.3 – Three toolheads for the E3D ToolChanger: **(a)** The standard FFF printhead, **(b)** the conductive printhead for printing conductive traces, and **(c)** the pick and place tool for placing electronic components.

The ink is contained in a single-use barrel with a plunger, which is pushed by the generated pressure. The barrel has a Luer-Lock connection, allowing various off-the-shelf nozzles to be attached, including single-use plastic nozzles of different diameters or small metal tips commonly used in medical applications.

Conductive traces are printed by moving the tool over the substrate while activating the solenoid valve to apply pressure. Ink is dispensed in a fixed amount determined by the nozzle tip size, applied pressure, and ink viscosity. The width and height of the printed wire can be adjusted by changing the nozzle size, pressure, or tool speed. For consistent results, pressure and speed must be manually fine-tuned for each nozzle and ink type before printing.

Pick and Place System

To integrate electrical components into the printed parts, we developed a pick and place tool that can be mounted on the E3D ToolChanger, together with a component tray to hold the components during the printing process. The system uses a vacuum nozzle to pick up components from the tray and place them on the printed part. The nozzle has a spring-loaded tip and can freely rotate 360° around the Z axis using a hollow shaft stepper motor. Vacuum generation is provided by a small pump and a solenoid valve, similar to the conductive printhead. The component tray is mounted on the right side of the printer and moves up and down along the Z axis to allow component pickup. The tray is interchangeable and is currently equipped with multiple strips of reeled SMD components. To place a component, the printer first picks up the pick and place tool and moves to the component tray. The component tray then rises, pressing the component

onto the vacuum nozzle. Vacuum captures the component, after which the tray lowers again. Using the calibration camera (described in Section 3.1.3), the component is oriented correctly. Finally, the printer moves the tool to the printed part and places the component.

Camera

A camera system was developed to be mounted on the E3D ToolChanger for in-process documentation and monitoring of the printed parts. Images of the printed object can be captured at any time during the printing process. The system is also used to assist in calibration of the other printer components and to locate components on the component tray. All versions are based on Raspberry Pi camera modules connected via the Camera Serial Interface (CSI) to a dedicated Raspberry Pi. Communication is performed over the network with the main Raspberry Pi that controls the printer. Image streaming was first provided by mjpg-streamer through a web interface. Later, this was replaced by a custom image server that can supply calibrated images via the web interface. Multiple revisions of the camera system are described in the following sections.

Raspberry Pi Camera The first generation camera is the Raspberry Pi camera module (Figure 3.4a). It is a compact board-level unit with a resolution of 2592×1944 pixels. After removing the locking glue, the lens focus point was adjusted to be only 7 cm from the lens. This allowed it to capture detailed images of the printed objects. The camera is mounted directly on the tool pickup head, so it moves with every tool and can be used without a tool change. It is held in a small 3D printed mount that secures both the board and the lens. For improved lighting, a Light Emitting Diode (LED) module with a white diffuser was mounted around the camera. Because the inexpensive lens showed strong optical distortion, the cameras Intrinsic parameters

(a) **(b)** **(c)** **(d)**

Figure 3.4 – Four imaging tools for the E3D ToolChanger: **(a)** first-generation Raspberry Pi camera module, **(b)** Raspberry Pi HQ camera with standard lens, **(c)** Raspberry Pi HQ camera with telecentric lens, and **(d)** Intel RealSense D405 depth camera.

were calibrated and compensated using OpenCV (described in more detail later in this section). Even after calibration, significant residual distortion remained and only the central part of the image was reliably usable. The images were therefore cropped to 1000 × 1000 pixels for analysis. The field of view was 18 × 18 mm with a resolution of 18 µm per pixel. This camera was used for the initial experiments described in Chapter 5 and was later replaced by the Raspberry Pi HQ camera due to its lower effective resolution and heavy distortion.

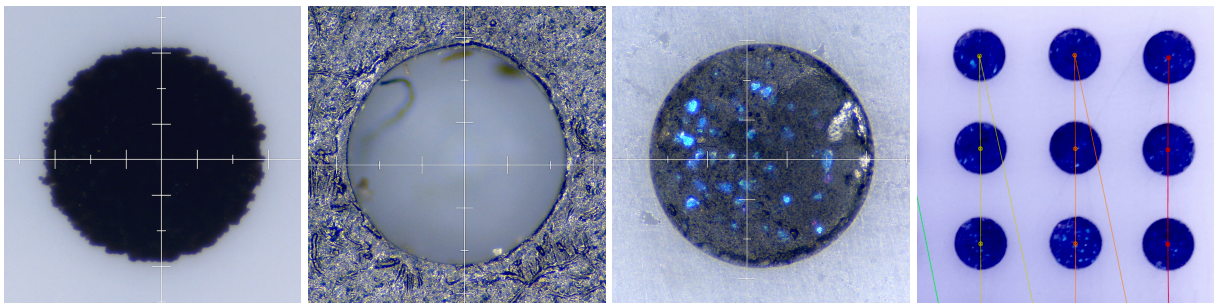
Raspberry Pi HQ Camera The second generation camera uses the Raspberry Pi HQ camera module (Figure 3.4b). Because this module is larger, it cannot be mounted on the tool pickup head and is instead installed on a dedicated tool that the tool changer must pick up before capturing images. It provides a resolution of 4056 × 3040 pixels and has a CS mount for interchangeable lenses. A Basler C125-0818-5M lens with an 8 mm focal length is mounted using a distance ring to shift the focus point to 5 cm from the lens. This close focus results in a field of view of 40 × 30 mm and a resolution of 10 µm per pixel. For illumination, a ring of 16 addressable LEDs is mounted around the lens. The improved sensor and lens produce significantly better images with reduced distortion. However, a noticeable fisheye effect remains. Distortion was compensated with OpenCV, producing cropped 2000 × 2000 pixel images (images had to be square at that development stage). The resulting images are of high quality, but residual distortion still causes offsets of up to 15 pixels (150 µm) at the edges when stitching multiple images. Accurate measurement in the images is not possible because objects that are only slightly out of focus appear smaller or larger, and the distortion is not constant across the image. This camera was used for the final experiments in Chapter 5 and the initial experiments in Chapter 6 and was later replaced by the Telecentric Raspberry Pi HQ camera.

Telecentric Raspberry Pi HQ Camera The telecentric camera is not a completely new camera, but the Raspberry Pi HQ camera fitted with a different lens (Figure 3.4c). Therefore, the image sensor specifications remain unchanged from the previous setup. The lens used is the telecentric TC12024 from Opto Engineering. Telecentric lenses are designed to maintain constant magnification across the entire field of view and eliminate perspective distortion. They are intended for high-precision applications and are well suited for dimensional measurement in images. The selected lens is comparatively large and heavy, measuring 120 mm in length and weighing 342 g. Because of its size, both the camera tool and its parking position had to be redesigned. The lens has a working distance of 67.2 mm, which is roughly twice that of the Basler lens. Combined with the lens length, this increases the required Z travel when switching to the camera and reduces the available Z height for printing. With a magnification of 0.255, the resulting field of view is 24 × 18 mm at a resolution of 6 µm per pixel. A supplied test protocol indicates that lens distortion is below 2 pixels (12 µm) across the entire image. The very low distortion and absence of the previous fisheye effect make it possible to use the full image without correction or cropping. Measurements are accurate over the complete image area, not

only near the center. The camera is equipped with a 24-LED addressable ring around the lens similar to the previous camera. It is used for the experiments described in Chapter 6.

Intrinsic Calibration All camera setups exhibit inaccuracies and systematic errors in the camera's optics. These include lens distortions (e.g., fisheye effect), the optical center of the camera, and the focal length. These factors can be compensated for up to a certain point using OpenCV. Calibration is performed by employing a known pattern, such as a checkerboard or circular pattern, and capturing multiple images of this pattern from different orientations and angles. Because the pattern is known, the calibration parameters can be calculated by detecting the key points in the pattern and matching them to their real-world coordinates. These parameters are then used to correct the images captured by the camera.

Most checkerboards are quite large since most camera applications have a much larger field of view. An off-the-shelf checkerboard fitting the camera's field of view was not available, so a custom one was designed and manufactured. Because a standard chessboard pattern is difficult to produce at this scale, a circular dot pattern was chosen instead. The first attempt was to print the pattern with a laser or inkjet printer on photo paper so the ink would not be absorbed. However, due to low printer resolution, the pattern edges were not sharp and the circles were not perfectly round (Figure 3.5a). A symmetric circular pattern was then produced using a 1 mm drill on a CNC milling machine. Holes were drilled into a 2 mm white acrylic plate. Placing this plate over a black background failed because chips and dirt inside the holes were visible. Filling the holes with black silicone created an uneven surface and reflections. Pre-painting the plate and drilling holes in the painted acrylic led to an uneven surface and hole edges lacking sharpness (Figure 3.5b). The final solution was to fill the holes with black carbon infused paste (Figure 3.5c). This produced a smooth surface with sharp edges and low reflections. The resulting pattern enabled the algorithm to detect it reliably and perform the calibration (Figure 3.5d). Using the resulting calibration matrix, the image server corrected images during acquisition.

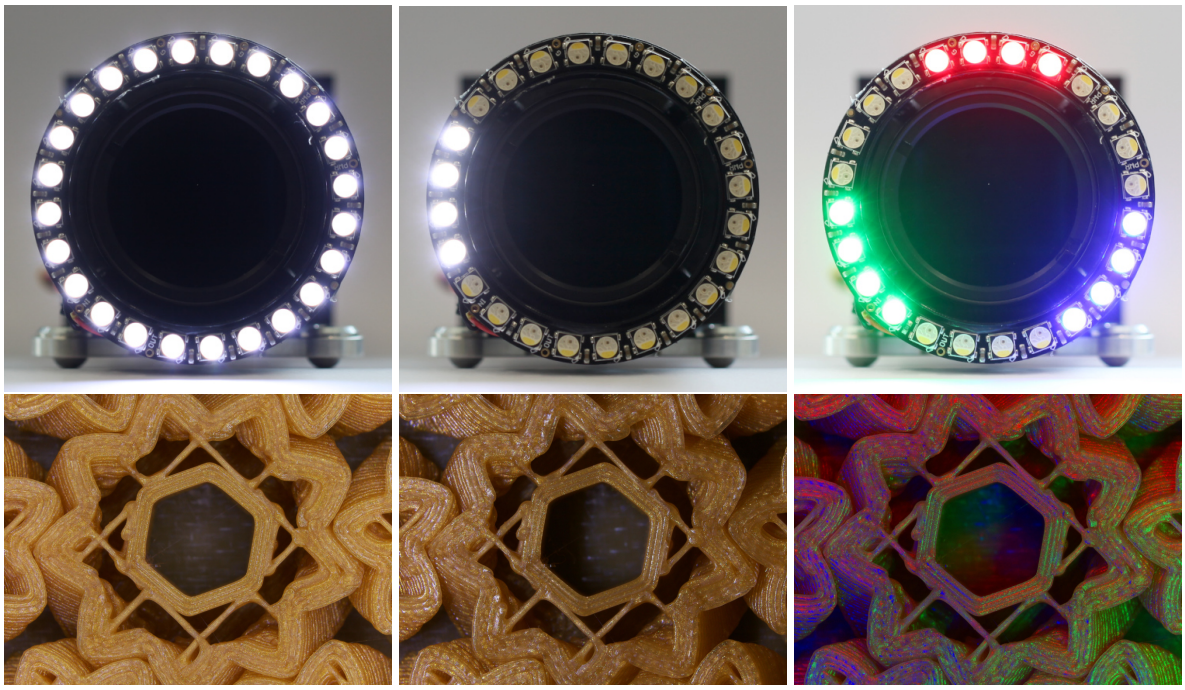


(a) **(b)** **(c)** **(d)**

Figure 3.5 – Three different methods for manufacturing the calibration pattern: **(a)** printing on photo paper, **(b)** drilling into a pre-painted acrylic plate, **(c)** filling drilled holes with black toothpaste, and **(d)** the resulting image after detection of the calibration pattern.

Because fisheye compensation introduces black pixel regions along the image borders, the corrected images must be cropped before further processing. Nevertheless, they are far more usable for image analysis than the original raw images.

Dynamic LED Ring The first camera used simple Light Emitting Diode (LED) lighting that could only be switched on or off. For further experiments, adjustable brightness, directional illumination, and different colors were required. This enables experiments with shadows cast by the printed part and the use of color to emphasize different features. The new lighting system is based on a NeoPixel LED ring with 16 or 24 addressable RGBW LEDs (depending on the camera lens). Each LED package contains a red, green, blue, and white LED plus a built-in controller chip that sets each color independently with 256 brightness levels. An Arduino Nano controls the LED ring and is connected to the Duet 3 control board via a Pulse Width Modulation (PWM) signal. Different light patterns are selected through this PWM signal. The patterns are hardcoded on the Arduino and include: full white at 100%, full white at 25%, illumination only from the left, right, top, or bottom, and a pattern with red, green, and blue light spaced 120° apart. Figure 3.6 shows three different light patterns and the resulting images. The LED ring is mounted around the camera lens using a 3D printed holder and a lens hood that prevents direct glare from the LEDs into the camera.



(a) **(b)** **(c)**

Figure 3.6 – Three different light patterns of the dynamic LED ring for the E3D ToolChanger camera (top) and the resulting images (bottom): **(a)** full white at 100%, **(b)** illumination only from the left, and **(c)** red, green, and blue light spaced 120° apart.

Calibration Camera

The calibration camera uses the same camera module and LED ring as the first generation camera tool. It is employed to calibrate the X and Y positions of the different tools and to assist with aligning SMD components after they are picked up. The camera is mounted upside down on the right side of the printer's frame and can move up and down along the Z axis via an additional stepper motor. This vertical motion shifts the focus point to match tools of different heights or the height of a picked SMD component. The camera is also used to calibrate the individual tools relative to one another. Calibration is performed by capturing images of each tool and measuring the distance between the tool's tip and the center of the image.

Depth Camera

The Intel RealSense D405 is a compact depth camera that determines depth using stereoscopic vision. In stereoscopic vision, two cameras capture images from slightly different viewpoints. The camera matches corresponding points in the two images and computes the disparity (difference in pixel positions). Using this disparity together with the known distance between the cameras, depth is obtained by triangulation. The resulting depth data is then filtered to reduce noise, handle occlusions, and improve the accuracy of the depth map. The D405 has a specified operating distance of 7 to 50 cm, but in practice it exhibits issues below about 8 cm. It provides images at 1280 × 720 pixels with a depth resolution of 0.1 mm. It outputs both depth and RGB images with a field of view of 165 × 92 mm, resulting in 150 µm per pixel. Figure 3.4d shows the depth camera mounted on a tool-plate with additional LED lighting.

3.2 Five Axis E3D ToolChanger

The second printer used in this thesis is the five-axis E3D ToolChanger. It is based on the same platform as the three-axis E3D ToolChanger and uses similar controller hardware. Figure 3.7 shows the five-axis E3D ToolChanger. The base Cartesian motion system is unchanged from the three-axis version. However, the original printbed and mounting plate were removed and replaced with a T-slot aluminum plate supported by three wedges attached to the former printbed holder. On this plate, two rotary axes are mounted that allow the build platform to tilt in two directions. These additional axes are referred to as the A and B axes, where the A axis rotates around the X axis and the B axis rotates around the Z axis. The A axis is mounted to the Z axis with ball bearings on both sides for stability and is driven by a stepper motor via a belt drive with a 1:3 reduction. This reduction is required because the A axis must hold up the weight of the axis against gravity during printing. The B axis is mounted on the A axis, and the build platform is attached directly to the stepper motor shaft. Both axes are driven by Nema 17 stepper motors with 400 steps per revolution (0.9° per step). All rotary axis components are 3D printed from

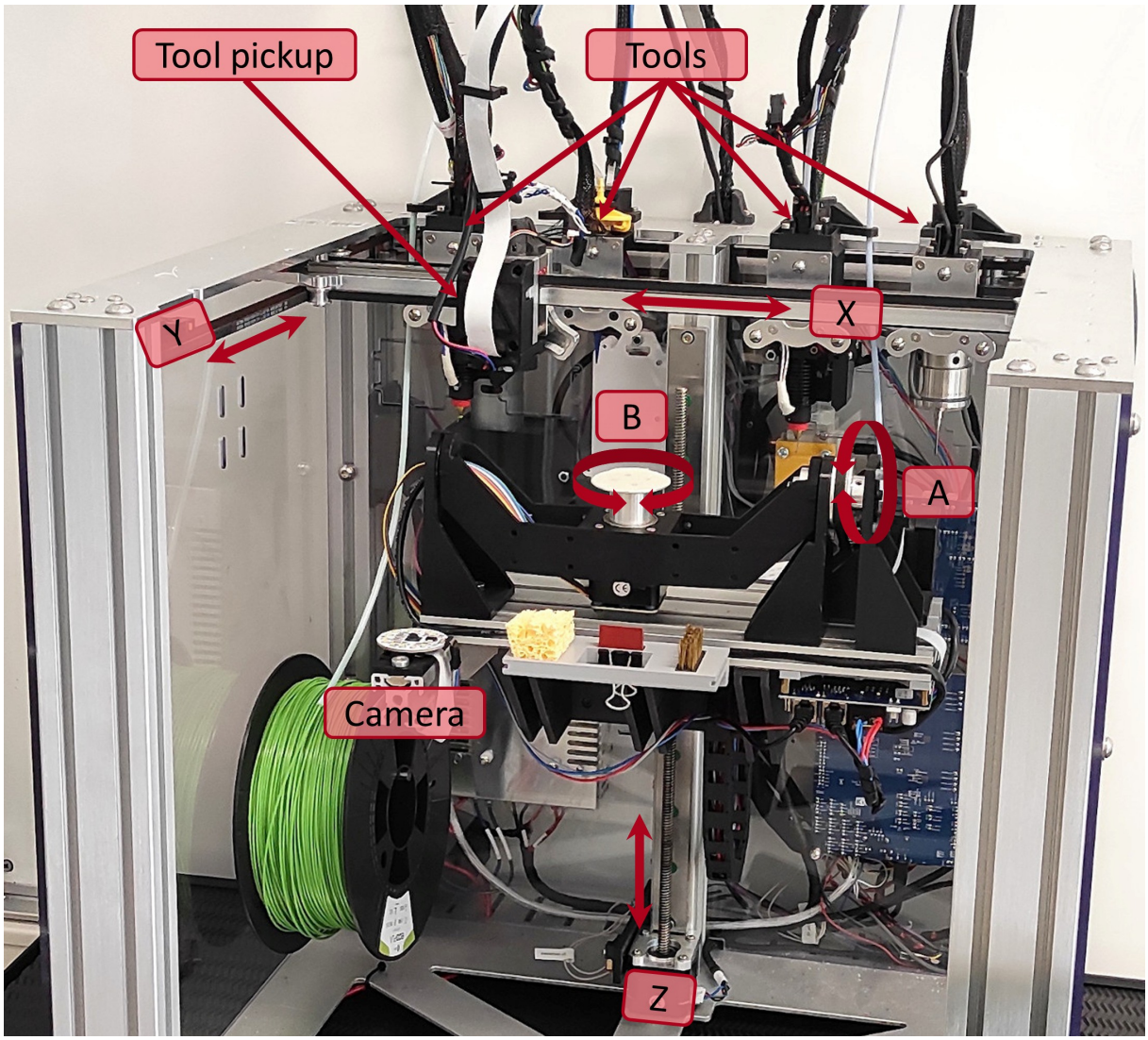


Figure 3.7 – The modified five-axis E3D ToolChanger with the A and B axes mounted on the Z axis and the X and Y axis moving the tool pickup head. The four tools are parked at the back of the printer. The built in camera is used for tool calibration.

ABS plastic, which tends to warp and shrink during printing, resulting in some inaccuracies in the printed parts. Figure 3.8 shows a CAD model of the A and B axes.

The five axis printer uses the same Duet 3 control board as the three axis E3D ToolChanger. Each rotational axis is driven by a Duet 1HCL expansion board that controls its stepper motor. This board supports magnetic encoder based position feedback. A magnetic encoder is mounted on the rotation shaft of the A axis and on the motor shaft of the B axis. It supplies position feedback to the control board with a resolution of 0.17° and is used to home the axes. The expansion board software also supports closed loop position control, but this feature has not worked properly so far.

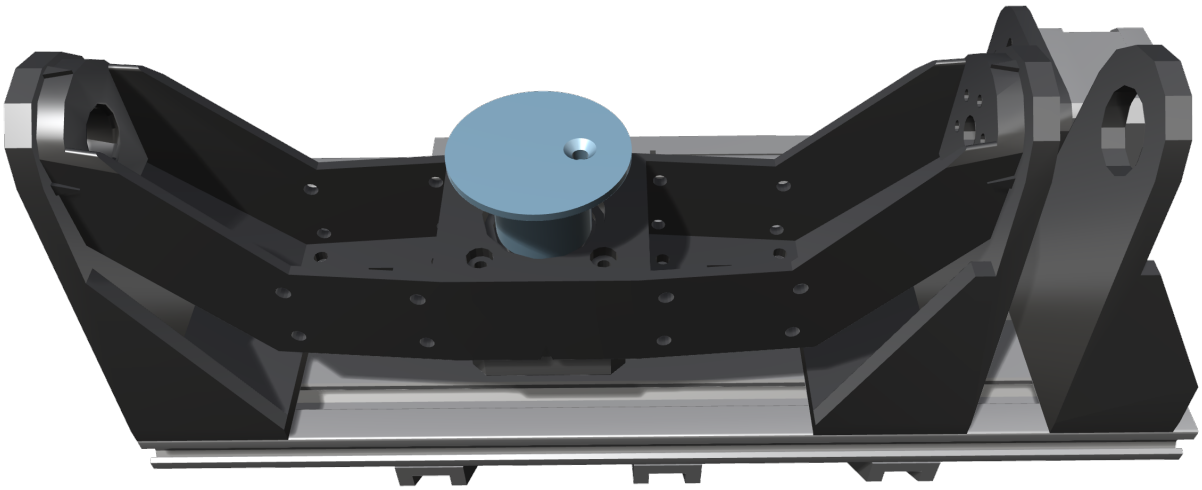


Figure 3.8 – A CAD model of the A and B axes mounted on the Z axis. The A axis rotates around the X axis, and the B axis rotates around the Z axis. The build platform (blue) is mounted directly on the motor shaft of the B axis.

3.2.1 Toolheads and Accessories

The five axis E3D ToolChanger employs two specially designed FFF toolheads optimized for printing with the 5-axis motion system (Figure 3.9a). It also uses the printhead for conductive material described in Section 3.1.3 (Figure 3.9b). A dedicated touch probe tool is mounted on the tool changer to measure axis deviations and calibrate the printer (Figure 3.9c). For tool calibration, a calibration camera similar to that described in Section 3.1.3 is mounted near the A axis.

FFF Printhead

The FFF toolhead has a deliberately slim, elongated design to minimize the risk of collisions with the printed part. It is built around the E3D Hemera extruder and equipped with an E3D Revo hotend. The installed nozzle is a Revo belt nozzle, specified for printing angles of up to 45°. The part cooling fan is mounted high on the toolhead and guides airflow through a long duct down towards the nozzle.

Touch Probe

A dedicated touch probe tool is used to calibrate the rotary axes and the Z axis. The CNC-Step 3D-Finder [W20] is a commercial probe that detects surface contact in the $\pm X$, $\pm Y$, and $+Z$ directions. It includes a 25 mm stylus with a 2 mm ruby ball tip and is spring loaded. When the ball touches a surface the stylus retracts and the contact is registered. The required actuating force is 0.5-1.0 N in X and Y, and 2.5 N in Z. The deflection needed to trigger the probe differs between XY and Z, with Z typically requiring greater travel. Its specified repeatability is 1 µm.

(a) **(b)** **(c)**

Figure 3.9 – Three toolheads for the five-axis E3D ToolChanger: **(a)** the FFF printhead designed to minimize collisions (two are installed), **(b)** the conductive printhead for printing conductive traces, and **(c)** the touch probe tool for calibrating the rotary axes.

On deflection the internal contact opens so the probe functions electrically as a simple switch to the controller. When the controller registers a change in the signal, it stops moving and records the current position. Because the probe is normally held in a CNC spindle, it has an 8 mm shank. A 3D printed clamp mount adapts this shank to a ToolChanger plate. The probe must be collected by the tool changer before use. Its position relative to the machine is calibrated with the calibration camera.

3.3 Neotech PJ15X

Some of the experiments described in Chapter 5 were evaluated on a Neotech PJ15X printer. This printer is installed in the laboratory of the ZIM-KamEl project partner Neotech AMT GmbH. The PJ15X is an industry-grade commercial 3D printer designed for fully additive printing of electronics and other functional applications, combining additive manufacturing with a 5-axis motion system. It is a benchtop machine with a build volume of 400 × 300 × 140 mm. Similar to the Five Axis E3D ToolChanger, the PJ15X uses a Cartesian XYZ motion system with additional A and B rotational axes. In this machine, the tool mounting plate moves along the Z and X axes while the AB rotation unit moves along the Y axis. The motion system is specified with a repeatability of $\pm 10\,\mu m$ in XYZ and 0.002° in AB. The printer is controlled by specialized proprietary software and a printer controller running on an industrial PC. It is modular in design

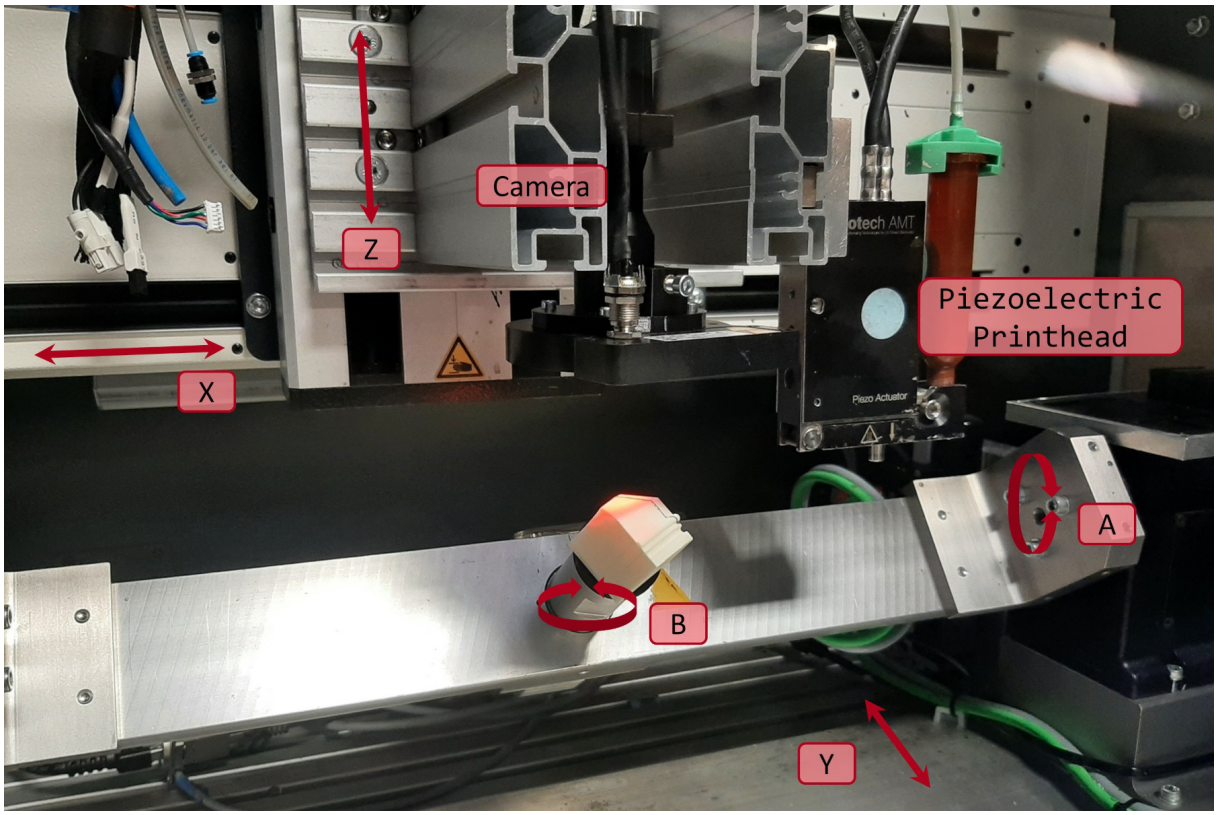


Figure 3.10 – The Neotech PJ15X printer with the rotating A and B axes mounted on the Y axis. the substrate camera and the Piezoelectric Printhead are mounted on the Z axis which moves along the X axis.

and can be extended with different tools and accessories. Figure 3.10 shows the printer setup with the Piezoelectric Printhead and the substrate camera mounted.

3.3.1 Toolheads and Accessories

Neotech AMT GmbH offers modular printing systems that can be configured with different tools according to application and customer requirements. Each tool can be selected as needed and mounted individually on the machine. The system architecture emphasizes flexibility, allowing straightforward integration of additional or updated tool technologies. Because the machine lacks an automatic tool changer, tools must be installed manually and then remain on the printer during operation. This becomes problematic during 5-axis printing, as unused tools can collide with the printed part. Therefore, for a specific task usually only the required tools are mounted. In the experiments described in Chapter 5, the Piezoelectric Printhead and the camera system were used, so only these two tools were mounted on the printer.

Piezoelectric Printhead

The Piezoelectric Printhead is a inkjet based device for printing conductive traces. It ejects the ink droplets using a piezoelectric actuation principle. The ink is held in a replaceable Luer-Lock cartridge that screws onto the printhead. From the cartridge the ink flows into a small internal fluid chamber feeding a precision ceramic nozzle. Inside the printhead a piezoelectric stack drives a plunger that pressurizes this chamber to expel a droplet through the ceramic nozzle. Applying a voltage pulse to the stack moves the plunger and the generated pressure creates the emitted droplet. Adjusting the pulse amplitude (voltage) and duration controls the stack motion, and thereby the plunger stroke and speed. The printhead is specified to deposit 350 µm wide traces using a polymer-based conductive silver ink that is low-temperature sintered.

Substrate Camera

The camera system is used for in-process monitoring and documentation of the printed parts, and also for precise substrate alignment. The camera is a monochrome Basler acA2440-20gm featuring a global shutter and a network interface. It provides a native sensor resolution of 2448×2048 pixels and is fitted with the telecentric lens TC5M-10-65C. Because this lens has a magnification of 1.0, the field of view equals the sensor dimensions of 8.45×7.07 mm, giving the whole setup an effective resolution of 3.45 µm per pixel. Illumination is provided by two complementary sources: a coaxial light input mounted inline with the lens and an external ring light surrounding the lens barrel. The coaxial light directs light straight through the lens so it follows the same optical path as the captured image, suppressing shadows and reducing specular glare, which improves uniformity and overall image quality. A drawback is that this very even lighting can reduce edge contrast on the substrate surface, making boundaries harder to distinguish. In contrast, the ring light projects light obliquely from the side, intentionally creating mild shadows and controlled reflections. The camera is fully controlled by the Neotech software, and image acquisition can be triggered via G-code commands at arbitrary machine positions and orientations during the printing process.

Chapter 4

Path Planning for 5-Axis Printing

Multi-axis printing has recently received increased attention because it addresses the limitations of 3-axis printing. It enables printing from arbitrary directions onto an object's surface. This improves direction dependent properties of printed parts, reduces the need of support structures, and enables printing of objects with complex geometries. For printed electronics, printing wires along the surface enables connections along the Z axis and allows components to be placed in arbitrary orientations. These capabilities are not possible on conventional 3-axis printers.

A 5-axis printer is a printer augmented with additional rotary axes that enables tilting and rotating the printbed. This added degree of freedom requires adapted path planning, since the toolpath not only contains cartesian positions but also orientations for every point. Path planning thus becomes more complex because there are often multiple valid solutions (axis configurations) for the same point, potential collisions with the object strongly affect the planning decisions, and the order of printing becomes important.

5-axis printers are significantly more complex than comparable 3-axis machines. Precise calibration is essential because the machine must know the exact positions and orientations of the rotary axes and the intersection point between them. Because of the leverage of the rotary axes, even small misalignments produce large errors at the tool position. Ideally, the rotary axes would be parallel to the main axes, perpendicular to each other, and intersect in a single point. In reality, these ideal conditions are never fully achieved. The A and B axes are not perfectly aligned, and they often do not intersect in a single point. This is usually mitigated by using more expensive motion systems that can be finely aligned. However, that conflicts with the goal of low-cost 5-axis printing.

The approach in this chapter starts by measuring the position and orientation of the rotary axes together with their intersection point. Using these measurements, a kinematic model of the printer is constructed that records the deviations in a Unified Robot Description Format (URDF) [W21] description of the machine. This model is then applied for toolpath compensation by a specialized analytical Inverse Kinematics (IK) solver that computes the rotary axis angles from the measured offsets. For generating the toolpath on the model, the surface normals

are extracted from the object directly beneath the path. The extracted normals are then adapted to ensure a continuous surface velocity. With the compensation and the adapted toolpath, the printer can follow the surface of the object even when the rotary axed show large deviations from their ideal positions.

This chapter is based the work on the closed source Aion-5X software [W22] by Neotech AMT GmbH and Kronos Mechatronics GmbH and on the publication:

- Daniel Ahlers, Tom Schmolzi, German Junca, Jianwei Zhang, and Florens Wasserfall. "Calibration and compensation of 5-axis 3D-printers for printed electronics". In: *Additive Manufacturing Letters 12* (2025). doi: 10.1016/j.addlet.2024.100265 [20]

4.1 State of the Art

Today's path planning in 3D printing is mostly based on planar slicing, where the object is sliced into horizontal layers [72]. Modern slicers like Cura [W6], PrusaSlicer [W7], or OrcaSlicer [W8] still use planar slicing to generate the toolpath. The process has been refined in recent years with features like adaptive layer heights [73] and dynamic line widths to fill layers faster and more accurately [74]. Although the process is well established and works for most objects, it has several limitations. The most obvious limitation is that only horizontal layers can be printed. Planar 3D printing is effectively a 2.5D process where the layers are planar and the third dimension is only used for layer changes. This limits the geometries that can be printed. Overhangs need to be held up by support structures that have to be be removed after printing. This is time consuming and can leave marks on the object. Another issue is the stair-stepping effect where the layer lines remain visible on the object. This can be reduced by using a thinner layer height, but that increases the print time.

To overcome these limitations, Curved Layer Fused Deposition Modeling (CLFDM) was introduced by Singamneni et al. [11], where the layers are no longer planar but curved and the third dimension is used to form the layer curvature. This reduces surface roughness and produces a more uniform appearance [13]. Because the layers are not flat, the process is often referred to as nonplanar slicing. Later, the approach was extended by combining curved and planar layers in a mixed-layer strategy [75]. The authors showed that the mechanical properties of printed parts can be improved by using nonplanar layers [12]. The first toolpaths for nonplanar layers were generated with scripts that follow mathematical functions to create the curved layers. To generate nonplanar toolpaths for arbitrary objects, a projection-based approach was integrated into a 2.5D slicer [23]. In the projection-based approach, each layer that should be printed nonplanar is first projected onto a plane parallel to the build plate. Then the layer's toolpath is generated on that plane and finally projected back onto the original layer, forming a nonplanar toolpath. Shan et al. [76] proposed an isothermal-surface-based algorithm to generate nonplanar layers. As not all surfaces are printable on a three-axis machine, Ottonello et al. [77] proposed a method to

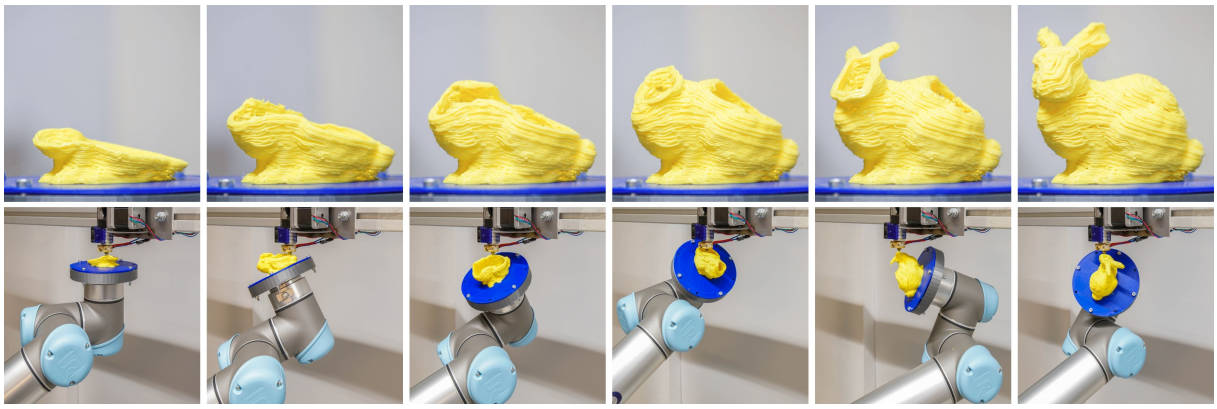


Figure 4.1 – Printing a bunny with a 6 DOF robotic arm using deformed layers to avoid support structures. [79].

automatically select printable surfaces. A major issue in nonplanar printing with 3-axis printers is that the nozzle is not oriented orthogonally to the surface. This causes slight collisions with the surface and creates artifacts [78]. This issue can be reduced by using a multi-axis printer that can tilt the nozzle relative to the surface.

Printers with more than three axes are often referred to as multi-axis printers. These printers can tilt the nozzle or the printbed to print on nonplanar surfaces while keeping the nozzle oriented orthogonally to the surface. This is a major advantage for nonplanar printing because the nozzle does not collide with the surface. However, the process is more complex since the toolpath must be generated for the additional axes and collisions are much more likely. Furthermore, these printers require much higher accuracy and calibration because errors are amplified by the additional axes.

Early feasibility studies on multi-axis printing were conducted by Rieger et al. [15], using a 6 Degrees of Freedom (DOF) robotic arm with a mounted printhead to print curved layers. The authors identified challenges such as determining process-specific kinematic parameters, developing new slicing and path generation algorithms, handling singularities and self-collisions, assessing the feasibility of non-standard build directions, and related issues. Later, Dai et al. [79] proposed multi-axis printing strategy that avoids the need for support structures by deforming the toolpath of the object (Figure 4.1). Besides 6 DOF robotic arms, delta-style printers with a moving printbed were also used for multi-axis printing [80]. The authors showed that surface quality can be improved by aligning the toolpath with the surface normal.

A strategy to avoid collisions in multi-axis printing is cylindrical slicing. In this method, a central core tube is either placed on the build plate or first printed there, and the object is then built concentrically around this tube. Toolpath generation proceeds by virtually unwrapping each cylindrical layer into a planar strip, generating the path in that plane, and then wrapping it back onto the cylinder. Because the layer is only transformed in one (circumferential) dimension, this introduces no toolpath distortion. Cylindrical printing reduces the need for support struc-

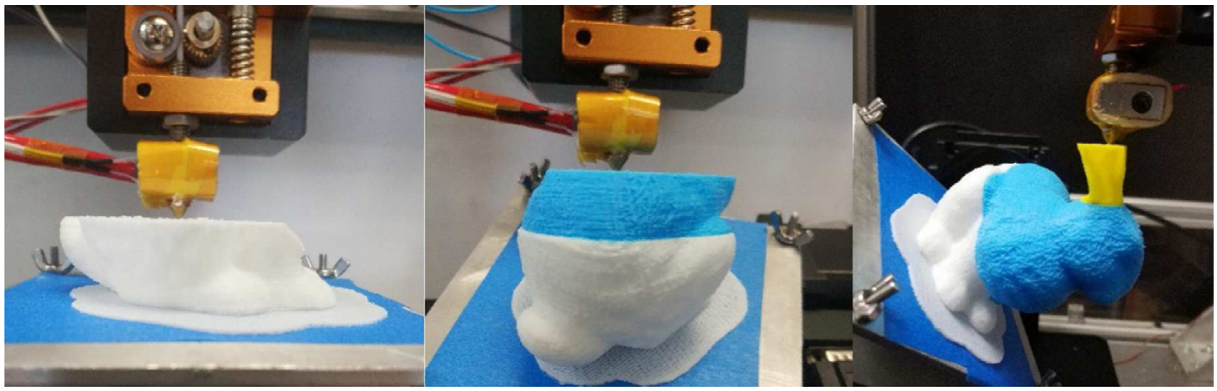


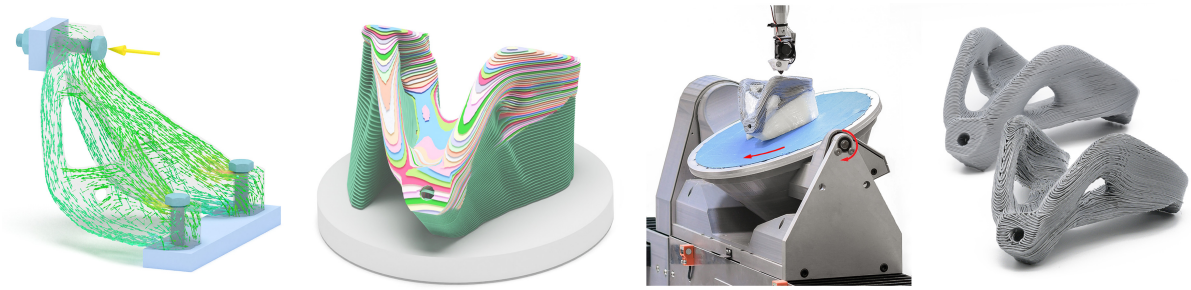
Figure 4.2 – Printing a decomposed bunny with three cutting planes using a 5-axis printer. [83].

tures and decreases the number of layers, thereby reducing manufacturing time and material waste [14]. Another approach to avoid collisions is spherical slicing, where the object is divided into concentric spherical layers. Here, toolpath generation is more difficult than in cylindrical slicing because spherical layers cannot be unwrapped and rewrapped without distortion [81]. To eliminate this distortion, a dedicated slicing algorithm was proposed that directly generates spherical layers and corresponding toolpaths for multi-axis printing [82].

For tube-like structures, a helical printing strategy was proposed in which the object is produced along a continuous helical path [84]. In this approach, the toolpath is generated by tracing a single helix that follows the geometry of the part. This method is commonly applied to vase-like structures, allowing the entire object to be fabricated as one uninterrupted path. Later, Bhatt et al. [85] proposed combining a 3 DOF build platform with a 3 DOF extrusion tool (total 6 DOF) to build thin shell asymmetric parts without using support structures.

These approaches are limited to specific geometries and are therefore not suitable for arbitrary objects. One way to print arbitrary objects with multi-axis printers is to decompose the object into subsections that can each be printed with planar layers. Wang et al. [16] proposed a pipeline that decomposes the object into subsections with different printing directions. The pipeline determines the assembly (printing) order and chains the individual segment toolpaths together, resulting in a continuous toolpath that can be printed with a multi-axis printer. As calculating the optimal cutting planes is quite challenging, heuristic methods are used to approximate an optimal decomposition [83] (Figure 4.2). Compared to planar printing, these decomposition-based approaches reduce the need for support structures, improve the surface quality of the printed parts [86], and can significantly improve the mechanical strength [87].

Decomposing objects into planar printable regions is an effective way to fabricate arbitrary objects with multi-axis printers. However, the process is complex, and the resulting object is not produced in a single continuous path. To address these limitations, curved layers were proposed, in which the object is printed with nonplanar layers that follow its surface. Fang et al. [88] proposed a field-based optimization framework to generate such curved layers and corresponding toolpaths for multi-axis printing. They showed that filament alignment can be optimized



(a) **(b)** **(c)** **(d)**

Figure 4.3 – Printing curved layers along the main stress directions using a 5-axis printer. **(a)** The simulated stress distribution in the object. **(b)** Curved layers aligned with the stress directions. **(c)** Printing of the object. **(d)** Comparison of a planar print (top) and the optimized curved print (bottom) which can withstand a double the load. [88]

along directions with large stresses and that printing can be performed without collisions (Figure 4.3). Later, Li et al. [89] proposed a lattice infill structure generation algorithm that ensures self-supporting conditions for both the infill and the boundary surface of the part. They also proposed a printing sequence optimization algorithm to determine a collision-free order of printing. Zhang et al. [90] introduced a quaternion-based formulation to generate curved layers with local collision avoidance. They demonstrated that the toolpath can be optimized for support-free printing, strength reinforcement, or surface quality depending on the requirements of the printed part.

A major issue in planar printing is the anisotropic strength of printed parts. The tensile strength of a part can be increased by up to 167% by optimizing the printing direction [91]. For parts requiring increased strength and stiffness, fibers can be incorporated into the design. These parts greatly benefit from multi-directional nonplanar surfaces, which allow fibers to be aligned with the stress directions [92]. Fang et al. [93] proposed a pipeline that generates optimized 3D continuous fiber toolpaths for models with complex geometry using two 6 DOF robotic arms. Their enhanced method calculates an optimized scalar field that produces curved layers aligned with maximum stress directions, resulting in a 644% increase in failure load and a 240% improvement in stiffness compared to conventional planar printing without fiber reinforcement.

A major issue in all multi-axis approaches is the positioning accuracy of the printer. The achievable accuracy is limited by the mechanical construction and the quality of the calibration. In most cases, specialized 5-axis printers are more precise than 6 DOF robotic arms where errors get multiplied by the long kinematic chain. Lopez-Arrabal et al. [94] showed that the positioning accuracy of their 6 DOF robotic arm is limited to $\pm 350\ \mu m$. For 5-axis machines, Lei et al. [95] demonstrated that accuracy can be improved by fabricating reference objects and measuring them to construct error models [96], allowing real-time error correction. Bohez et al. [97] used an expensive touch probe to measure systematic errors in 5-axis NC machining and compensated these errors directly in the inverse kinematics equations. The R-Test was used by Weikert et

al. [98] to measure geometric errors in 5-axis machines using three incremental probes arranged orthogonally. Fu et al. [99] developed a model for squareness errors in multi-axis machine tools using the Denavit-Hartenberg (DH) matrices to improve geometric error compensation. For even higher accuracy, a laser interferometer has been used to measure geometric errors in 5-axis machines [100]. These measurements capture both position-independent and position-dependent geometric errors. The position-independent errors are compensated by analyzing their influence on the local frames of the rotary axes [101]. Besides the accuracy and repeatability of different probes, their cost is also a major factor [102]. Liu et al. [103] measured their machine with a trigger pin mounted to the nozzle to close an electric circuit. This low-cost method showed that the accuracy of the A and B axes of their low-cost 5-axis printer is limited to $400\ \mu m$. This error is too large to print useful parts with all five axes moving. For low-cost 5-axis printers, a reliable method to measure and compensate these errors is therefore required to enable useful parts to be printed with all five axes moving.

4.2 Calibration and Kinematic Modeling

Accurate path planning for 5-axis printing depends on knowing the precise positions and orientations of all printer axes. Calibration therefore measures the exact location and alignment of each individual axis together with the point where the two rotary axes nearly intersect. These measurements form the basis for constructing a kinematic model and for compensating mechanical deviations during printing. The procedure is performed on the 5-axis printer described in Section 3.2. That machine is equipped with a touch probe for calibration, two FFF extruders, and a conductive ink dispenser. The X, Y, and Z axes are homed against endstop switches on one side of the frame in the same manner as common 3-axis printers. During calibration, the global zero position is reassigned to the (intended) intersection point of the A and B rotary axes. Calibrating the A and B axes is more complex because neither their centers of rotation nor their rotation axes are directly accessible. Therefore, their exact positions and orientations relative to the Cartesian coordinate system must be determined. Because the B axis is mounted on the A axis, the position and orientation of B depend directly on those of A. Figure 4.4 visualizes the axes in 3D space, their kinematic relationship, and the minimum distance (closest approach) between the two rotary axes.

The position of an axis refers to where that axis is located within the global Cartesian coordinate system. The orientation of an axis refers to how that axis is directed relative to the global Cartesian coordinate system. Thus, the parameters that must be measured are: A position, B position, A orientation, and B orientation. From these parameters, the kinematic model of the printer can be constructed and the closest approach between the two rotary axes can be calculated to define the global zero position. The initial implementation of the calibration procedure was introduced by Tom Schmolzi in his master's thesis [104]. The calibration procedure directly

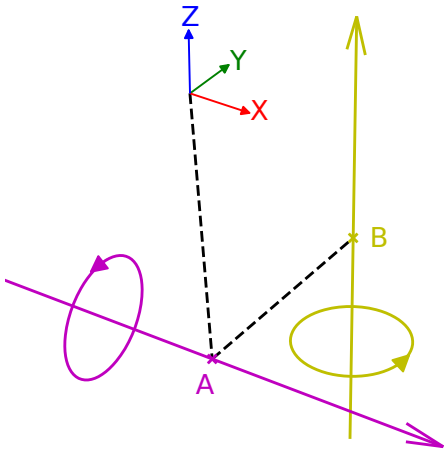


Figure 4.4 – 3D illustration of the rotary axes A and B. The short gray segment connecting them denotes their nearest approach. The gray line from the global origin to the B shaft indicates the translation to the center point on the B axis.

probes the physical physical axis hardware with a touch probe mounted on the printer. There is no additional measurement object mounted on the printers axes.

B Axis Position

To determine the precise B position, the metal shaft of the B stepper motor is probed along the negative and positive x and y directions. The shaft is probed at the height corresponding to the expected middle of the shaft ($Z = -10$). To obtain the B position, the two measurements for X (X_{p_1}, X_{p_2}) and for Y (Y_{p_1}, Y_{p_2}) are averaged. This average gives the B position relative to the 3D probe's tip. The global Cartesian coordinate system is then updated so that the B position is at the origin.

$$B_{pos} = \left[\frac{X_{p_1} + X_{p_2}}{2}, \frac{Y_{p_1} + Y_{p_2}}{2}, -10\right] \tag{4.1}$$

A Axis Position

The next step in the calibration is to determine the position of the A axis. To do this, the top of the B axis shaft is probed while the A axis is rotated to three angles (A_0, A_{90}, and A_{-90}). At A_0, the probing motion is along the Z axis, whereas at A_{90} and A_{-90} the probing motion is along the Y axis. These three contact points lie on a circle in the YZ plane, and the center of that circle defines the A axis position. Since this circle is probed from the inside, the radius of the shaft (r_p) must be added to the calculated center position to obtain the actual A axis position.

$$A_{pos} = \left[0,\ \frac{A_{90} + A_{-90}}{2},\ A_0 + \frac{|A_{90} - A_{-90}|}{2} + r_p\right] \tag{4.2}$$

A Axis Orientation

To determine the orientation of the A axis, its square crossbar is directly probed. Four points are measured in the z direction ($Z_{X,1}, Z_{X,2}, Z_{Y,1}, Z_{Y,2}$), forming a rectangle positioned as far away from the B axis shaft as possible. Using this initial set of four probe points, the slopes along the X (m_X) and Y (m_Y) axes can be computed. To obtain the slope along the Z (m_Z) axis, four further points are probed from the positive and negative Y directions at two distinct X positions ($Y_{X_1,1}, Y_{X_1,2}, Y_{X_2,1}, Y_{X_2,2}$).

$$\begin{aligned} m_X &= \frac{Z_{X,1} - Z_{X,2}}{2d_{ZX}} \\ m_Y &= \frac{Z_{Y,1} - Z_{Y,2}}{2d_{ZY}} \\ m_Z &= \frac{1}{d_{YX}} \cdot \left(\frac{Y_{X_1,2} - Y_{X_1,1}}{2} - \frac{Y_{X_2,2} - Y_{X_2,1}}{2}\right) \\ A_{dir} &= \frac{[1, m_Y, m_Z]}{|[1, m_Y, m_Z]|} \end{aligned} \tag{4.3}$$

B Axis Orientation

Finally, the orientation of the B axis is measured by probing the shaft a second time: now both the top and the bottom of the shaft are probed from all four sides ($p_{Z_j,i}$). With these eight measurements, the shaft's center positions at the top (p_1) and bottom (p_2) are calculated by averaging the opposing probe points. Because the B axis is mounted on the A axis, its measured orientation is influenced by the A axis orientation. Therefore the previously determined orientation offsets of the A axis must be subtracted to obtain the correct (isolated) orientation of the B axis. Rotations of the B axis about the Y axis can be compensated by adjusting the zero position of the A axis. All other rotational components cannot be compensated by the motion system and must be handled in the toolpath generation. The zero positions of all axes are set on the B axis at the point where the A axis is closest to the B axis.

$$\begin{aligned} p_1 &= \frac{1}{4}\sum^{i} p_{Z_1,i} \\ p_2 &= \frac{1}{4}\sum^{i} p_{Z_2,i} \\ B_{dir} &= \frac{p_2 - p_1}{|p_2 - p_1|} \end{aligned} \tag{4.4}$$

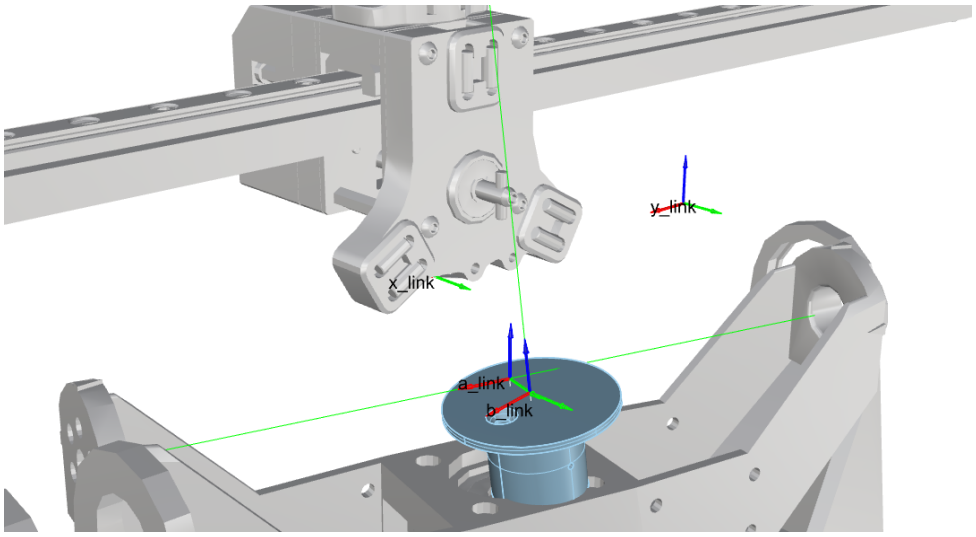


Figure 4.5 – Axes offsets of the generated URDF model of the printer displayed in Aion-5X. The axes alignment is deliberately bad to show the offsets as the typical offsets are too small to be visible.

Kinematic Model in URDF

To make later use of the measured parameters in toolpath generation and to visualize the printer's axes, a kinematic model of the printer is constructed. This model encodes the measured positions and orientations of the A and B axes, as well as those of the linear X, Y, and Z axes. It additionally describes the complete kinematic chain linking all axes, the tools, and the printbed. The model is expressed in the Unified Robot Description Format (URDF) [W21] format and contains 3D models for visualization for every axis.

URDF is a standardized format for describing the kinematic model of a robot. It was introduced by the Robot Operating System (ROS) and is widely used in robotics and simulation. The model is composed of links and joints that define the robot's structure. A link is a rigid body that can be positioned in space, and a joint connects two links and allows relative motion between them. This enables supporting different kinematic configurations across different printers. The URDF can also include separate geometries for visualization and collision detection for each link.

All these features are especially useful when multiple printers with different kinematic configurations must be supported. Because the format is text based, it can be easily generated for any printer and is still easily understood by humans. The generated URDF model includes all measured parameters and offsets, as well as the general kinematic structure of the printer. This model is later used in the path planning stage to generate the toolpath and apply the required compensation. A URDF file including the measured offsets is provided in Listing 4.1. Figure 4.5 shows this URDF model inside the Aion-5X software. In that example, the offsets are intentionally exaggerated to make them clearly visible.

Listing 4.1 – URDF model of the 5-axis E3D ToolChanger with measured calibration parameters. Note that the visual tag that contain the 3D mesh for each link are shortened here.

```
<?xml version="1.0"?>
<robot name="5x_e3d_toolchanger">
        <!-- Links (visual tags shortened) -->
        <link name="base_link">
                <visual>...</visual>
        </link>
        <link name="y_link">
                <visual>...</visual>
        </link>
        <link name="x_link">
                <visual>...</visual>
        </link>
        <link name="z_link">
                <visual>...</visual>
        </link>
        <link name="a_link">
                <visual>...</visual>
        </link>
        <link name="b_link">
                <visual>...</visual>
        </link>
        <link name="tool_link" is_tool_link="true" is_toolchanger="true"/>
        <link name="workpiece_link"/>

        <!-- Joints -->
        <joint name="x_joint" type="prismatic">
                <axis xyz="1 0 0"/>
                <parent link="y_link"/><child link="x_link"/>
        </joint>
        <joint name="y_joint" type="prismatic">
                <axis xyz="0 1 0"/>
                <parent link="base_link"/><child link="y_link"/>
        </joint>
        <joint name="z_joint" type="prismatic">
                <axis xyz="0 0 -1"/>
                <parent link="base_link"/><child link="z_link"/>
        </joint>
        <joint name="a_joint" type="revolute">
                <origin rpy="0.0 -0.0010621 0.0286015" xyz="0 -0.0001302 0"/>
                <axis xyz="1 0 0"/>
                <parent link="z_link"/><child link="a_link"/>
        </joint>
        <joint name="b_joint" type="continuous">
                <origin rpy="0.0 0.0011657 -0.0286015" xyz="0 0.0001302 0"/>
                <axis xyz="0 0 1"/>
                <parent link="a_link"/><child link="b_link"/>
        </joint>
        <joint name="tool_joint" type="fixed">
                <origin xyz="0.0 0.019425 0.027578" rpy="0 0 3.14159265359"/>
                <parent link="x_link"/><child link="tool_link"/>
        </joint>
        <joint name="workpiece_joint" type="fixed">
                <parent link="b_link"/><child link="workpiece_link"/>
        </joint>
</robot>
```

4.3 Compensated Toolpath Generation

In planar 3-axis printing, the toolpath is created by slicing the object into horizontal layers and generating a path for each layer. Converting these paths into printer movements is straightforward, as each point's coordinates map directly to the printer's X, Y, and Z axes. On 5-axis printers, the toolpath generation is more complex because the additional rotary axes must also be controlled.

Toolpath Generation for 5-Axis Printing

In 5-axis printing, each point in the toolpath must provide not only X, Y, and Z coordinates, but also an orientation that specifies how the tool is angled relative to the surface at that location. Therefore, every point in the toolpath includes a normal vector that represents the tool's orientation. Computing these normal vectors is not straightforward because it depends on the object's local surface geometry and requires special handling at edges and corners. Because the normal vector is not directly represented by the printer's axes, it must be converted into angles for the rotary axes A and B. This conversion is performed by the IK solver during toolpath generation. As some point and orientation combinations can be achieved with different A and B angles, the IK solver must select one valid solution that produces a smooth continuous toolpath while avoiding singularities.

Compensated Toolpath Generation

The previously measured inaccuracies of the printer that are encoded in the URDF model must be compensated somewhere between the toolpath generation and the actual printing on the machine. Compensating positional offsets is straightforward: they can be added directly to the coordinates, and such adjustments could even be handled by the printer firmware. By contrast, rotary-axis compensation is more involved because it requires non-linear corrections that depend on the IK solution. Since printer firmware typically does not support custom IK solvers, these computations are performed in the toolpath generation software instead.

Aion-5X Software

Aion-5X [W22] is a proprietary toolpath generation system developed by Kronos Mechatronics GmbH, provided as a C++-based plugin for the CAD software Rhino [W23]. It specializes in generating 5-axis toolpaths for printed electronics, but can also be used for other processes such as 5-axis FFF printing. It currently supports generating free-form lines on arbitrary surfaces and filling closed, conformal surfaces with different patterns. One core feature is the generation of 5-axis toolpaths that take into account the measured kinematic model of the printer. This

enables compensated toolpaths that follow the surface of the object even on printers with large deviations.

4.3.1 Toolpath Generation

Toolpath generation in Aion-5X begins by loading the object's 3D model into the software. If the model does not include wire paths, the user can draw them directly onto the object's surface in Rhino. Because Rhino often represents curves as splines, these paths are converted into a sequence of straight line segments. These segments define a 3D path that closely follows the object's surface and are used to generate the toolpath.

Normal Extraction

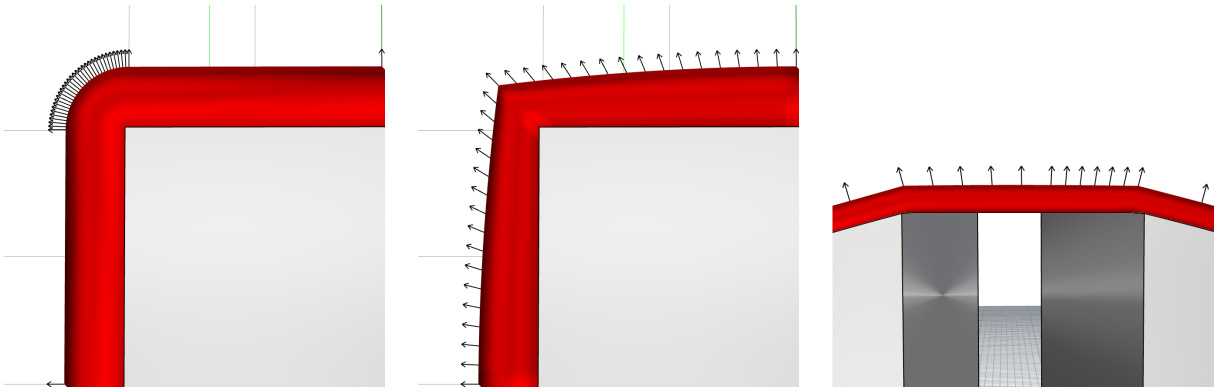
For 5-axis printing, the toolpath must also include normals to orient the tool relative to the surface. These normals are taken from the surface directly beneath each point. This is done by finding the surface with the smallest Euclidean distance to the point. The underlying surface may be either a mesh or a Boundary Representation (Brep). Because some points are closest to edges or corners where multiple surfaces meet, a single point can yield more than one normal. To handle this correctly, all normals from surfaces at approximately the same distance to the point are collected. In practice, a small range ϵ is used to account for numerical imprecision and to ensure all relevant normals are included.

Normal Extraction on Edges and Corners

At object edges and corners, multiple surfaces meet, resulting in multiple valid normals. Because not all of them are relevant for the toolpath, special handling is required. Only normals belonging to surfaces in the direction of the previous point are kept, as well as normals belonging to surfaces in the direction of the next point. All other normals are discarded. For the first and last points of the path, this yields a single normal. For interior points that lie on edges or corners, this yields two normals. These two normals are then ordered by their angle relative to the normals of the previous and next points to ensure they are processed in the correct order.

Printing Around Corners

There are two ways to handle the two normals on corners during toolpath generation. The first way is to keep the printhead perpendicular to the surface while approaching the corner. Then, at the corner, the tool stops, angled towards the surface of the incoming path, then rotates around the corner to the normal of the outgoing path, and continues printing (Figure 4.6a). In the toolpath generation this is done by adding an additional point to the toolpath at the corner that contains the second normal. So the corner point is added twice to the toolpath, once with the incoming normal and once with the outgoing normal. With this method, the toolpath is not printed



(a) **(b)** **(c)**

Figure 4.6 – Two approaches to handle corners in the toolpath: **(a)** The tool stops at the corner and rotates to the next normal. **(b)** The tool is angled along the path towards the corner point and then tilted back towards the next normal. **(c)** When printing over gaps, the normals are interpolated between the last and next valid surface normal. This allows a smooth transition between the two normals.

in a continuous movement, and the tool has to stop at every corner. But the tool remains perpendicular to the surface while moving. Stopping along a path is an issue for some processes like FFF printing, where the tool has to be continuously moving to avoid material accumulations.

In the second method, the tool is gradually angled along the incoming path toward the corner and then tilted back along the outgoing path (Figure 4.6b). This avoids stopping at the corner so the path can be executed as a continuous movement. Here, no additional point is added at the corner, instead the average normal $\vec{V}$ is computed with the following formula:

$$\vec{V} = \frac{1}{2}\left(\vec{N}_{in} + \vec{N}_{out}\right)$$

This averaged normal is then assigned as a single point at the corner. As a result, the tool starts tilting toward the corner while approaching it, reaches the averaged normal at the corner, and then tilts back toward the outgoing normal while moving away. With this approach, the tool is not always perpendicular to the surface, which can be an issue for processes that must remain perpendicular at all times.

Therefore, the best approach depends on the corner angle and the specific process used to print the wire. There is also the option to combine both approaches: the tool is slightly angled along the path toward the corner point and then stops at the corner to rotate only a small amount before continuing along the outgoing path. With this, the motion can be kept as continuous as possible while still ensuring that the angle between the surface and the tool remains small.

Printing Over Gaps

Sometimes the toolpath must cross gaps where no surface exists beneath the point, so no local normal can be extracted. To handle this while maintaining a smooth, continuous movement, the toolpath over the gap uses normals interpolated between the last and next valid surface normal (Figure 4.6c). This interpolation is performed along the path, with the normal at each point determined by its distance to the two sides of the gap. When only one valid normal exists on one side of the gap, such as when the path ends in mid-air, the last valid normal is used for the entire gap.

4.3.2 Inverse Kinematics (IK) Compensation

An IK solver converts the desired position and orientation of each point in the toolpath into an axis configuration that the printer can execute. The solver mathematically determines the joint angles required to position and orient the tool at each target point along the path. Given the desired position and orientation, it calculates the necessary angles for each axis, ensuring accurate tool movement while avoiding singularities. The IK solver in Aion-5X is specifically designed for 5-axis printers with non-orthogonal axes. It uses the measured axis offsets and orientations encoded in the URDF model to compute axis configurations that reach the target points even on printers with significant mechanical deviations.

For each point in space, the task is to orient the tool vector v so that it points directly opposite to the tool's downward direction. In Figure 4.7a, the small black arrow depicts the tool vector v at a specific point in space. Accordingly, we seek the A and B axis angles that make v point towards the $+Z$ axis. When the axes are orthogonal, this is achieved by first rotating v around

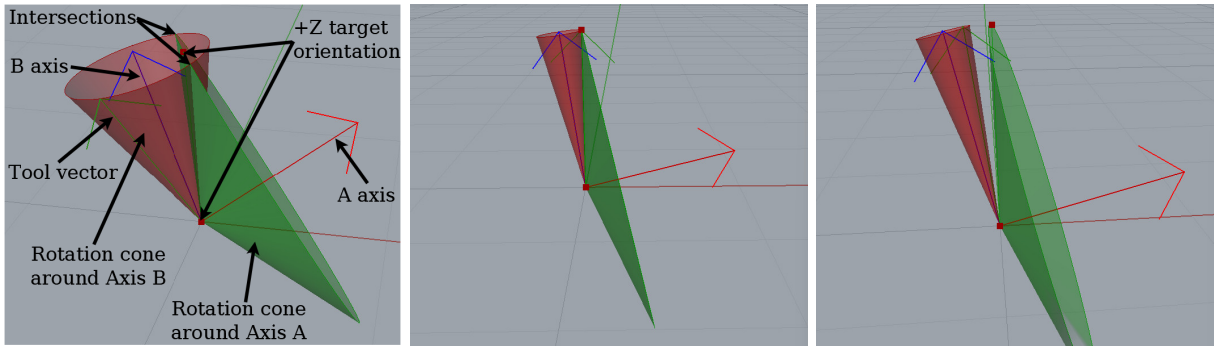


Figure 4.7 – Visualization of the inverse kinematics for orienting the tool vector v: The red cone shows all possible orientations of v when rotated around the B axis, and the green cone shows all possible orientations of the $+Z$ vector when rotated around the A axis. Valid IK solutions are found at the intersection points of both cones. **(a)** Two intersection points: two valid solutions. **(b)** One intersection point: one valid solution. **(c)** No intersection: no valid solution.

the B axis until it lies in the $\overrightarrow{YZ}$ plane, and then rotating it around the A axis until it points towards $+Z$.

When the axes are not orthogonal, finding the correct angles becomes more challenging. The possible orientations that are reachable by each axis can be visualized as cones in 3D space. In Figure 4.7, the red cone represents all orientations reachable by rotating around the B axis, and the green cone represents those reachable by rotating around the A axis. The intersection points of these cones correspond to the valid solutions found by the IK solver. Depending on the orientation of the tool vector v and the axis configuration, there can be zero, one, or two valid solutions. In Figure 4.7a, the cones intersect at two points, which yields two possible solutions for the tool orientation. The solver selects the one that results in the smallest change in angle from the previous point. In Figure 4.7b, the cones are tangent at a single point, so only one solution exists. Here the solver uses that single solution. In Figure 4.7c, the cones do not intersect, making it impossible at that position to orient the tool vector v towards $+Z$.

To overcome the issue of non-solvable cases, the proposed IK solver allows small angular tolerances. When the distance between the cones is smaller than this tolerance, the solution with the smallest distance between the cones is used as a valid solution. This is done by calculating the intersection of the B-cone with the plane defined by the vector $\overrightarrow{AB}$, which is the line connecting the A and B axes. This results in a small deviation of the tool angle from the desired orientation which is much less problematic than a deviation in the tool position. As long as the allowed deviation fits within the misalignment of the axes, the solver can find valid configurations for both the A and B axes for every point in space.

The XYZ position is then computed by applying the offsets defined in the URDF model. This ensures that the resulting path is positioned correctly with respect to the machine calibration. The result is a compensated toolpath that follows the object's surface, enabling accurate printing even on printers with significant mechanical deviations.

4.4 Evaluation

To demonstrate the proposed method, the printer described in Section 3.2 is first calibrated and an URDF model is generated. Several test objects are then printed to show that, with the proposed calibration and active compensation, an otherwise unusable low-cost 5-axis printer can produce useful parts. To illustrate the effect of the compensation, two identical objects are printed, one without compensation and one with compensation. Additionally, a fully FFF-printed globe with filled landmasses is produced, as closed surfaces are very sensitive to the distance between the nozzle and the surface. Finally, a "duck" with integrated electronics is printed to demonstrate the capabilities of printing electronics with conductive paste. All volumetric objects are sliced using PrusaSlicer [W7] and printed with the A and B axes kept fixed, because Aion-5X currently does not support volumetric printing.

4.4.1 Calibration Process and Results

Because the printer loses the position of each axis when powered off, the calibration process must be executed every time the printer is turned on. On average, running the calibration adds 167 seconds, though it can take longer because outlier measurements are repeated. Nevertheless, this overhead is negligible compared to the printing time, which can take several hours. Calibration sets the zero positions of the axes and outputs the offsets for the URDF model of the A and B axes. These offsets in the URDF model only need to be updated after significant changes to the printer, such as a crash or a mechanical modification. It might also change over time due to wear and tear, although this has not been tested yet.

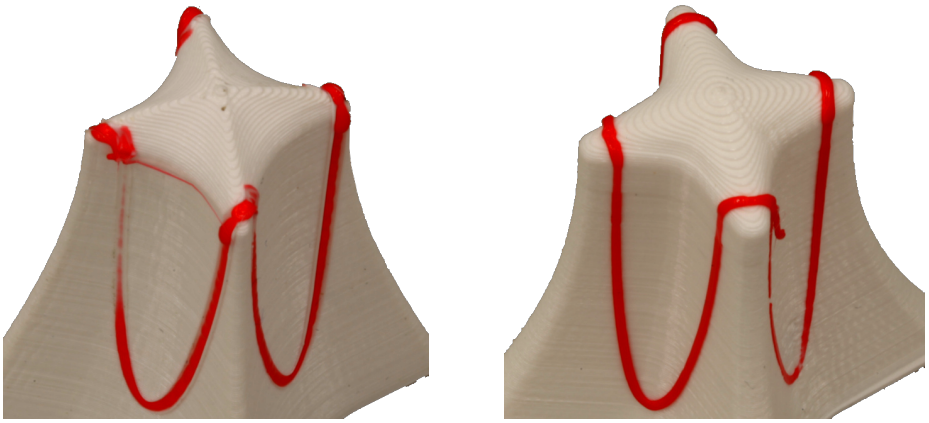
On the proposed E3D 5-axis toolchanger, the calibration results in a tilt of the A axis by $1.6°$ around the Z axis and $0.02°$ around the Y axis. The B axis is tilted by $0.12°$ around the Y axis and has an offset of 118 µm in the Y direction compared to the A axis. In practice, this means that the point 50 mm above the print bed with the A axis at $90°$ is 1450 µm off from the position it should be. This huge disparity between the expected and actual position is caused by the leverage effect of the A axis. The experiments by Tom Schmolzi [104] showed that after compensation, the measured offset is less than 50 µm, which is a significant improvement. Without compensation, this offset makes the printer unusable for 5-axis printing, as the tool would crash into the surface of the object or print in mid-air.

4.4.2 Comparison to Non-Compensated Printing

Two identical objects are printed to make the effect of the compensation clear. The first object is printed without compensation, assuming that the rotation axes are perfectly aligned with the main axes and that the A and B axes intersect exactly at the printer's global zero point. The second object is printed with active compensation enabled, taking into account the measured axis offsets described above, including the offset between the A and B axes. The printed object is designed to require simultaneous movement of both rotary axes and all three Cartesian axes. Without compensation (Figure 4.8a), the head repeatedly collides with the surface, causing squished lines and misalignment with the edges. At some points the nozzle pushes into the underlying material, which only works because the nozzle is hot enough to melt away the material and the object is hollow and can deform slightly under the crashing nozzle. With compensation (Figure 4.8b), these problems are eliminated. The lines show only minor squishing and follow the intended path as it was designed. The results show the effectiveness of the compensation, as the print is now usable and the lines align with the edges.

4.4.3 Application Example: Printed Globe

To show the capabilities in real world applications, a fully 3D printed globe is created. The printed globe consists of a blue sphere with a diameter of 45 mm, representing the northern



(a) **(b)**

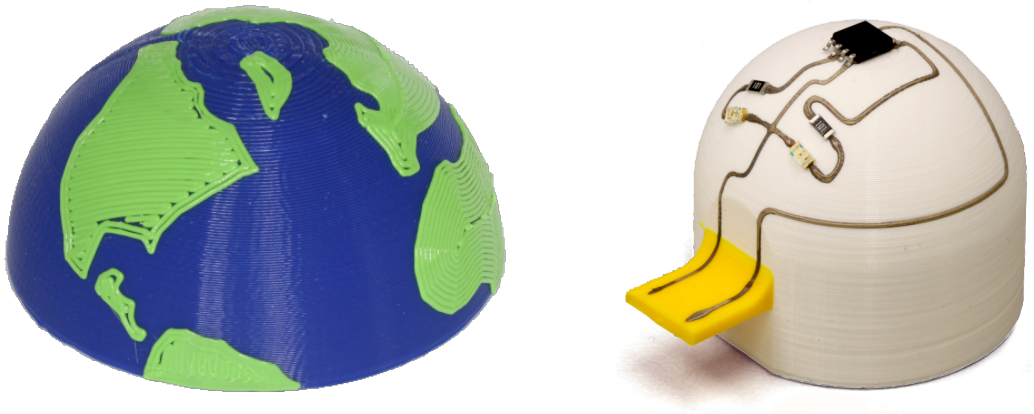
Figure 4.8 – 5-axis simultaneous motion test prints. **(a)** Printed without compensation, resulting in misaligned and squished lines due to axis errors. **(b)** Printed with active positioning compensation, showing accurate alignment and improved surface quality. The ”shoulder” features require coordinated movement of all axes, highlighting the effectiveness of compensation.

hemisphere of the Earth. The landmasses are printed using a surface filling strategy [81] in green PLA, making the continents stand out clearly on the globe. Filled surfaces are highly sensitive to the nozzle-to-surface distance because the toolpath is computed to deposit only the material needed to fill the area directly beneath the nozzle. Unlike single lines, where excess material can spread sideways, filled regions cannot push material outward because the area is enclosed by the previously printed surrounding perimeter. As a result, excess material is forced upward, creating a rough surface that follows the fill pattern. If the nozzle is too far from the surface, too little material is deposited, leading to visible gaps and uneven areas.

The printed northern hemisphere of the Earth is shown in Figure 4.9a. The landmasses are smooth and well-defined, demonstrating that the compensation achieves a consistent surface finish. However, two small areas show a slight deviation in nozzle-to-surface distance. Over North America the path is slightly too high, leading to a small gap in the surface. Over South America it is slightly too low, producing a rough area with some excess material pushed upward. These two regions are relatively close to each other and not on opposite sides of the globe. This occurs because the two landmasses are printed in different orientations, one with the A axis rotated in the negative direction and the other in the positive direction. This indicates a small remaining discrepancy between the kinematic model and the real printer, but it is minor enough to not significantly affect the overall print quality.

4.4.4 Application Example: Printed Electronics

To demonstrate the printer’s capabilities in printed electronics, a small ”duck” with integrated electronics is printed. The printed object is an FED class 3 circuit. Printed electronics are very



(a) **(b)**

Figure 4.9 – Application examples for compensated 5-axis printing. **(a)** A fully 3D printed northern hemisphere of the Earth, showcasing the effectiveness of the compensation in achieving a smooth surface finish. **(b)** Fully 3D printed ”duck” with functioning a electronic circuit on its head to pulse the LEDs. The active compensation ensures a constant distance between the conductive paste dispenser and the previously printed surface.

sensitive to the distance between the nozzle and the surface, as the conductive paste must be applied as a continuous line with a constant distance to the surface. Printing with a too large distance between the nozzle and the surface causes the ink stream to break and create gaps in the wire. If the distance is too small, the nozzle can collide with the surface and smear the ink. The dispenser nozzle used is very delicate. Even small collisions can bend the nozzle tip and prevent the material from being dispensed properly. A damaged nozzle tip must be replaced, which requires recalibrating the tools relative to each other, as the position of the nozzle tip varies with each replacement.

The ”duck” is a raised, 3D-printed half-sphere with an electronic circuit integrated on its top surface. The circuit consists of a microcontroller, two LEDs, and two resistors. Power is supplied via a printed male USB port, which also serves as the ”duck’s” beak. The LEDs are placed as the ”duck’s” eyes, while the resistors form the eyebrows. The microcontroller is programmed to blink the LEDs, producing a blinking effect. The ”duck” is printed in two steps: first the FFF part is printed, and then the conductive paste dispenser applies the wires. All wire paths were generated in Aion-5X using surface projection and then converted into toolpaths. Although, there are only small edges along the wire paths, they are still printed as continuous movements by tilting the tool toward the edge point, as described in Section 4.3.1. All wires are printed to be 0.4 mm wide and 0.2 mm high. The buck’s toolpath is generated in PrusaSlicer and then manually rotated and positioned to align correctly on the surface. This was very time-consuming, as the toolpath position needs to be calculated by hand from the kinematic model and the object. All G-code files are combined into a single file to print the whole object in a single run that uses

both FFF and the conductive paste dispenser. Since the printer does not support pick and place, all components are placed manually into the fresh ink directly after printing.

The resulting printed ”duck” with the electronic components on its head is shown in Figure 4.9b. The conductive paste is applied as a continuous line without any collisions between the nozzle and the surface. The wires are smooth and well-defined, indicating that the distance between the nozzle and the surface was maintained correctly. To obtain a working circuit, multiple attempts were required. However, nearly all difficulties were not caused by the toolpath generation or the printer itself. They were mainly due to human errors, such as smearing the conductive paste when positioning the SMD components by hand. Additional retries were necessary to get the positioning of the ”duck” correct, because the rotation and position offsets had to be calculated by hand and entered into PrusaSlicer. There were also instances of stringing caused by expired ink. As these problems are unrelated to the proposed method, they are not considered further here. Overall, the printed ”duck” works as intended, and when power is supplied via USB the microcontroller boots up and starts blinking the LEDs as programmed. This demonstrates that, with active compensation, the inaccurate 5-axis printer can produce complex objects with multiple processes in a single run, which was not possible before.

4.5 Conclusion

This chapter presented novel calibration methods and path planning algorithms that compensate for the mechanical inaccuracies of low-cost 5-axis printers. The calibration process probes the axes with a touch probe to determine the positions and orientations of the A and B axes, as well as the offset between them. These measurements are used to construct a kinematic model of the printer, which is then exported as a URDF model.

This URDF model is used by the Aion-5X path planning software to generate 5-axis toolpaths that follow object surfaces while actively compensating for the measured inaccuracies. Aion-5X, a C++ plugin for Rhino, generates toolpaths with continuous tool motion, including around corners. The IK solver accounts for the previously measured offsets, enabling precise multi-axis printing on low-cost printers. The overall approach is not limited to FFF printing and can be applied to other processes like printed electronics as well.

The effectiveness of the calibration and path planning was evaluated by printing the same objects with and without compensation. Two test objects were used to showcase the approach: a globe with filled landmasses and a ”duck” with integrated electronics. The results show that these objects are printable even on the presented low-cost 5-axis printer with significant mechanical deviations.

Overall, the presented approach significantly improves the accuracy and usability of low-cost 5-axis printers. It provides a method to plan and execute complex 5-axis toolpaths even on machines that would otherwise be unusable for such tasks, because they cannot consistently

maintain a constant distance between the nozzle and the surface during printing. This approach is not limited to low-cost printers but can also be applied to high-end machines, further improving their accuracy for fine structures.

4.5.1 Future Work

For future work, several improvements can be made to both the printer and the path planning software. A more reliable printer could be built that eliminates sources of uncertainty, such as backlash and wobble, ideally without increasing the cost. The calibration process can be improved to better handle rough printed surfaces, because the current method relies on probing the axes on the printed surface. This can be achieved by making the workflow more user-friendly and automatically updating the URDF model in the path planning software, eliminating the need for the user to copy and paste the output into the configuration file. The Aion-5X path planning software can be extended to support additional strategies, such as volumetric printing and multi-directional printing. This would enable printing more complex objects without the need to manually stitch together G-code files from different programs. Additionally, both the software and the hardware could be extended to support pick and place operations, removing the need for manual component placement. This would enable fully automated printing of complex objects with integrated electronics. Finally, the effectiveness of the compensation can be further investigated regarding the printer's reliability and accuracy. This includes testing its impact on high-end printers to assess whether the same approach can be applied to them. This would provide a better understanding of the presented approach's limitations and capabilities and of its applicability to different printers and processes.

Chapter 5

In Situ Error Correction of Printed Electronics

3D printing is a rapidly growing field that includes a wide range of additive technologies and materials. While most single-material processes are already established in industry, 3D printed electronics is still developing to become reliable enough for industrial applications. Printed electronics make it possible to fabricate wires and other electronic components directly onto or into objects without a dedicated Printed Circuit Board (PCB) or additional assembly steps. Although this approach will not replace classical PCBs, there are multiple use cases where 3D printed electronics adds extra design freedom. Currently, 3D printed electronics is mostly used for prototyping or novel applications, integrating sensors, antennas, and other electronic components directly into the printed object. Reliability is a crucial factor for electronics. In structural 3D printing, small defects in the build usually have only a minor impact on the final part quality. By contrast, printed electronics are highly sensitive to errors: a single interrupted connection generally disables the circuit and makes the entire object unusable. Such errors can arise during the printing process from issues such as a clogged nozzle, inconsistent material flow, or irregularities in the underlying surface. This is a major challenge for 3D printed electronics, because after fabrication the embedded wires are often inaccessible and, due to their integrated nature, cannot be replaced or tested in advance. Consequently, this limited reliability remains a significant barrier to the industrial adoption of the technology.

To address this issue, this chapter presents a novel approach for in situ error correction of 3D printed electronics. The machine acquires multiple images as the object is being printed. A deep convolutional neural network segments the conductive wires from the plastic substrate and the background. Subsequently, the algorithms analyze the intended G-code paths together with the segmented wire geometry to detect errors, including connection breaks, shorts, and points the printed wire failed to reach. The algorithms als estimate the width of each wire and can automatically repair connection breaks by reprinting missing material. Figure 5.1 provides an overview of the proposed process.

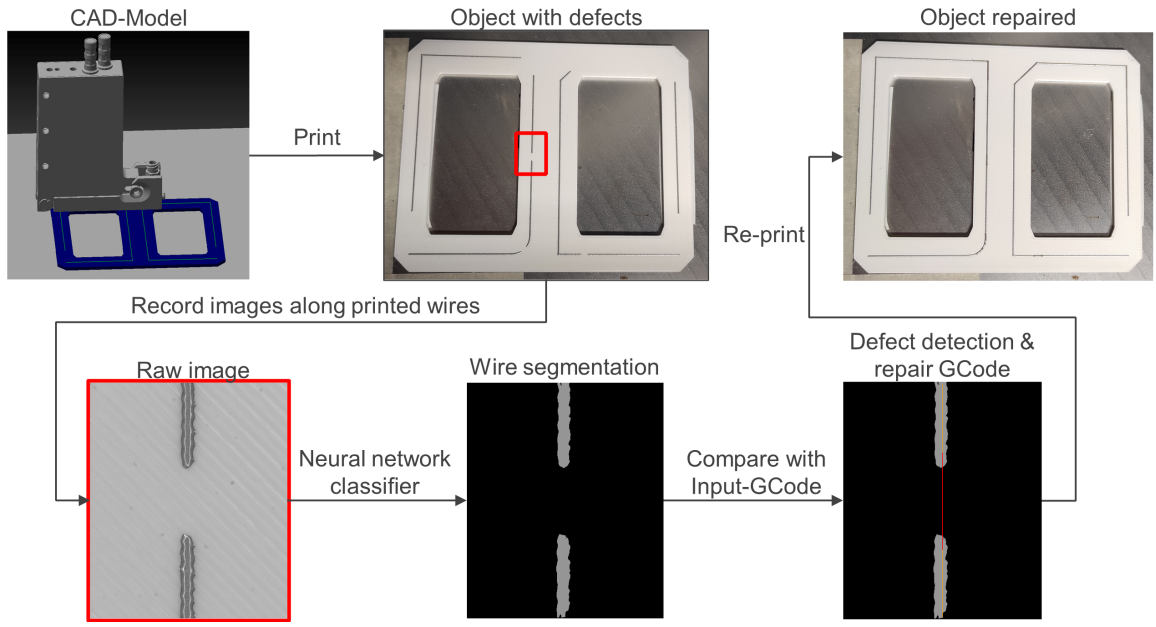


Figure 5.1 – In situ error correction process overview. The object is printed after each wire is deposited, the camera captures images of the wire. A neural network then segments the wire, and the segmented result is compared to the intended G-code paths to detect errors. If errors are detected, a repair path is generated to correct them.

This chapter is based on the publications:

- Daniel Ahlers. "In-Situ Verification of 3D-Printed Electronics Using Deep Convolutional Neural Networks". In: *Proceedings of the 32th Annual International Solid Freeform Fabrication Symposium* (2021). doi: 10.26153/TSW/17557 [18]
- Daniel Ahlers, Florens Wasserfall, Johannes Hörber, and Jianwei Zhang. "Automatic in-situ error correction for 3D printed electronics". In: *Additive Manufacturing Letters 7* (2023). doi: 10.1016/j.addlet.2023.100164 [19]

5.1 State of the Art

Production errors appear in every manufacturing process. In additive manufacturing, these errors occur even more frequently due to the complexity of the process and the typically more complex geometries of the products. When electronics are integrated into the printed parts, the overall complexity increases further. Electronics are very sensitive to errors, as even small defects can lead to failure of the whole system. Effective error detection is crucial in PCB manufacturing to ensure the functionality of the final product. Errors are identified using classical image processing [105], deep learning methods [106], and X-ray inspection for internal structures [107]. Most image processing techniques rely on comparison with reference images [108] or on image subtraction [109]. In recent years, deep learning approaches have been applied to detect

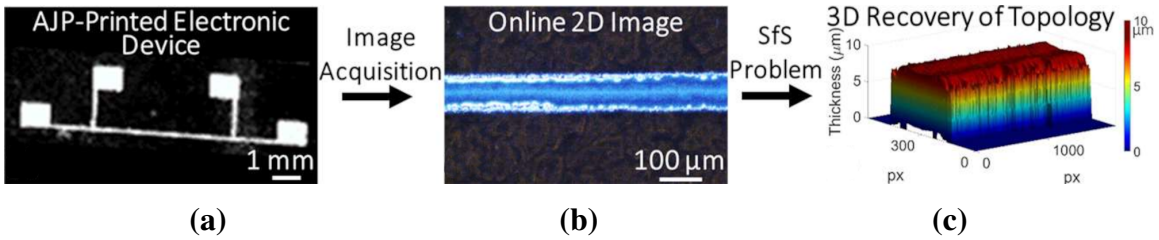


Figure 5.2 – In situ monitoring of aerosol jet printed electronics. **(a)** The printed circuit. **(b)** The captured image of the printed wire. **(c)** The reconstructed 3D topology of the printed wire. [118]

common defects arising during the etching process of PCBs. These defects are detected using convolutional neural networks [110, 111], ResNet50 [112], YOLO variants [113, 114, 115], and transformer-enhanced YOLO architectures [116]. The observed occurrence and characteristics of specific defects are used to adjust production parameters. Because of the low-cost per unit, defective PCBs are typically discarded rather than repaired.

The error detection methods used in PCB manufacturing are not directly transferable to 3D printed electronics, primarily because of the different manufacturing processes and the three-dimensional structure of the printed parts. Current approaches in printed electronics therefore concentrate on preventing errors by stabilizing the printing process. Salary et al. [117] introduced a method to quantify key aerosol jet printing attributes, including line width, line density, line edge quality/smoothness, overspray, line discontinuity, and internal connectivity. In later work, they applied Shape-from-Shading to estimate the cross sectional profile of electronic traces [118]. Figure 5.2 shows an example of the printed circuit, the captured image, and the reconstructed 3D topology of the printed wire. Li et al. [119] implemented an in situ process control for Aerosol Jet Printing that dynamically adjusts process parameters to improve print quality. Zhang et al. [120] used a hybrid machine learning approach to determine the optimal operating process window for Aerosol Jet Printing across various design spaces. Schirmer et al. [121] presented new semi automatized and objective methods for assessing print quality using image processing and statistical techniques. Lombardi et al. [122] recorded the printed line with an integrated camera and applied image processing to close the control loop and actively adjust parameters during printing.

All these approaches focus solely on stabilizing the printing process and do not address error detection and correction of the complete printed circuit. In our previous work [24], images were taken during printing with an integrated camera. The pixels in the region where the wire is expected are classified using an SVM to segment the actual wire from the background. Afterward, the algorithm searches for connection breaks by moving a sliding window along the wire and reports them to the user.

Current research in the field of 3D printed electronics still mainly focuses on improving the printing process, developing new materials, and stabilizing process parameters. Because 3D printed electronics are sensitive to errors, error detection and correction are a crucial part of the

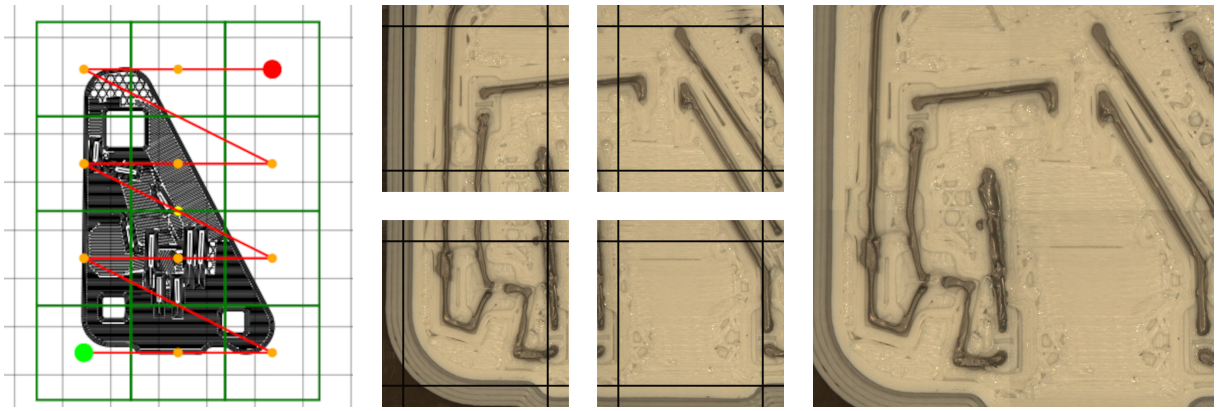
workflow. Printed electronic processes are well suited for error correction, as printed parts can often be repaired simply by printing over the defect. This must be further addressed in research to make 3D printed electronics a reliable and cost-effective manufacturing process.

5.2 Wire Segmentation

In the proposed approach, the object is inspected in situ with an integrated camera while it is being printed. A neural network processes the captured images and segments the conductive wires from the background. Because the wires may be covered by future printing steps shortly and would then not be visible anymore, each wire must be imaged immediately after it is printed. This chapter is based on research conducted within the ZIM-KamEl project. Therefore, two different systems were used for data acquisition. The first system is a modified FFF printer from the University of Hamburg, described in Section 3.1. The second system is a 5-axis printer from the project partner Neotech AMT GmbH, described in Section 3.3. Because the error detection and correction algorithms were designed to operate on both systems, the data recording procedures for each are presented in the following sections.

5.2.1 Layer Based Data Recording

As layer-based printed objects are built from stacked two-dimensional layers, the data recording for this process is likewise performed on a per-layer basis. After each layer is printed, the camera records the entire layer. Because of the short distance between the camera and the printed surface, and the requirement for low distortion, the camera's field of view is relatively small. So, only parts of larger objects can be captured in a single image. To capture these objects entirely,



(a) **(b)** **(c)**

Figure 5.3 – The OctoCameraDocumentation plugin records multiple images of each layer and stitches them together. **(a)** The user interface of the plugin. **(b)** The individual images taken of a layer with the overlap between them. **(c)** The stitched image of the whole layer.

multiple images of the same layer are taken and then stitched together to form one image of the whole layer.

The layer documentation is implemented in OctoPrint [W17] as the OctoCameraDocumentation plugin [W19], developed at the University of Hamburg and released as open source. At the beginning of a print, the plugin parses the G-code file and determines the dimensions of each layer. From this it computes the layer outline and generates a grid of image positions covering the entire layer. Within each layer, images are scheduled in a bottom-to-top, left-to-right order. To avoid gaps between images and to improve stitching robustness, a user-configurable overlap can be added. The plugin calculates and stores the center coordinate of every planned image position for later use. Image acquisition is triggered by a dedicated G-code command (M942) inserted into the slicer's layer change G-code. When the plugin encounters this command, it pauses the print, switches to the camera, moves the camera sequentially to the precomputed positions, and captures an image at each one. After all positions are recorded, the images are stitched into a single composite representing the full layer. Figure 5.3 shows the interface, the captured individual tiles, and the resulting stitched layer image. Because the raw images do not align perfectly, the plugin applies a correlation based stitching algorithm that uses the overlapping regions to accurately register and merge them into a seamless image of the whole layer [24]. Before the print is resumed, the printhead must be wiped to ensure consistent material flow and remove ooze that gathered during the pause.

5.2.2 5-Axis Data Recording

The PJ15X printer from the project partner Neotech AMT GmbH is equipped with additional rotary axes and can therefore print onto objects from arbitrary directions. This capability enables true 3D printing of electronics on complex geometries, allowing the wires to follow the surface contours. However, for such complex geometries the image recording process becomes more challenging. Due to occlusions and varying surface orientations, covering the entire object's surface with images is no longer practical, and a different strategy is required. This challenge is addressed by recording images per wire rather than per layer. In this approach, images are

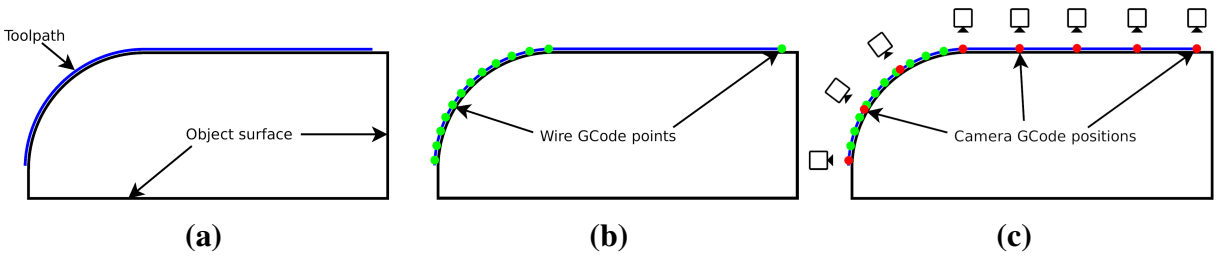


Figure 5.4 – Sampling positions for 5-axis data recording on a curved surface. **(a)** The object with the wire toolpath shown in blue. **(b)** The original non-uniform toolpath points shown in green. **(c)** The re-sampled, equidistant camera recording positions projected onto the surface shown in red.

captured immediately after each wire is printed by moving the camera along the wire's path and taking images. A downside of the wire based image recording is that when multiple wires are printed close together, the same wires may be recorded multiple times with only slight offsets between images, leading to redundant data and increased processing effort.

Because the calculation of the camera recording positions requires prior knowledge of both the printed wire paths and the underlying surface geometry, the wire toolpath must be generated in the CAD/CAM software before printing rather than dynamically during printing. Because the spacing between successive toolpath points is non-uniform, special care is required when determining the camera recording positions. Capturing images at every original toolpath point would oversample curved sections and undersample straight segments. Therefore, the toolpath is resampled to obtain equidistant recording positions along the wire, and the surface normal at each position is computed to set the camera orientation correctly. Figure 5.4 illustrates the problem of non-uniform toolpath point spacing and the solution via resampling of a wire toolpath on a 3D surface. To later recover the exact position of every recorded image, each image receives a unique identifier. This identifier is stored together with its 5D coordinate (X, Y, Z, A, B) in a list that is saved along with the G-code file and the images.

5.2.3 Training Dataset Generation

Because the proposed approach relies on training a neural network to segment the wires from the background, a suitable training dataset is required. Since the color images taken on the modified FFF printer differ significantly in appearance from the monochrome images recorded on the 5-

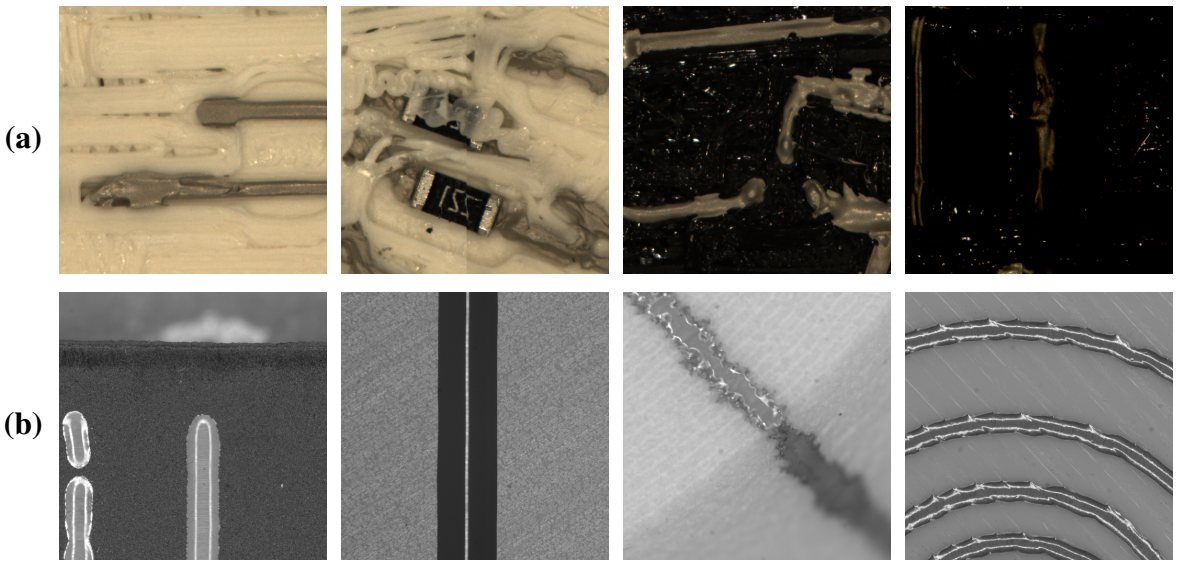


Figure 5.5 – Eight example images from the printed electronics datasets. **(a)** Examples from the *planar-dataset* showing different substrate and also SMD components. **(b)** Examples from the *5-axis dataset* showing different surface textures, wire appearances, and lighting conditions.

Table 5.1 – Overview of the samples produced as a base for the *5-axis dataset*. Each sample has a different appearance and different printed structures. Both the coaxial and the ring light were used to record the images. In total, 4200 images were captured.

ID	Material	Manufacturing Process	Appearance	Application
PET	PET	Film-Extrusion	Transparent	Sensor
PC	PC	Film-Extrusion	Black	Heating
FFFblack	PC	FFF	Black	Consumer electronics
FFFwhite	PETG	FFF	White	Consumer electronics
PA	PA	Injection molding	Grey	Automotive Interior rigid
PU	PU	Molding	Black	Automotive Interior flex
5AxisSLS	PA	SLS	White	Automation/Robotics

axis printer, two separate training datasets are created. For both datasets, the original images are too large to be used directly as input for the neural network. Therefore, they are split into smaller tiles of 512×512 pixels that serve as the network inputs. To distinguish them, the three-axis, layer-based dataset is referred to as the *planar-dataset*, and the five-axis dataset is referred to as the *5-axis dataset*.

As the layer documentation feature had already been in use before the ZIM-KamEl project and many images were available, the *planar-dataset* was created from previously recorded documentation images. These images were captured with a color camera and the spatial resolution was 20 µm per pixel. Each original 1000×1000 pixel image was divided into four tiles of 512×512 pixels. To enlarge the dataset, the color of the plastic substrate was modified to different tones using GIMP. The conductive regions were then manually labeled to generate the ground truth. For more robust detection, the dataset was further expanded through data augmentation: images were randomly flipped, rotated, and adjusted in brightness and contrast. The final dataset consists of 5,742 image tiles and is split into 80% training and 20% validation subsets. The split was performed before augmentation to ensure no overlap between training and validation data. Figure 5.5a shows example images from the *planar-dataset*.

For the *5-axis dataset*, images were recorded during the ZIM-KamEl project by the project partner Neotech AMT GmbH with a spatial resolution of 3.45 µm per pixel. To obtain a broad variety of printed wires, multiple builds were produced using different materials and application scenarios. The dataset contains dense wire networks, isolated single wires, contact pads, and wire geometries that include both curved and straight segments. The samples were printed with varied process parameters, leading to differences in line appearance, including cases with overspray. To achieve stable detection performance, the dataset spans a wide range of substrate surface textures, which affect both image appearance and the geometry of the printed conductive tracks. In total, 4,014 images containing conductive traces were recorded. Table 5.1 summa-

rizes the materials and processes used to generate the samples. Because many images look quite similar and manual annotation is time-consuming, 211 images with deliberately varied appearances were selected for labeling. Given the high spatial resolution, the images can be reduced to a height of 512 pixels without losing wire visibility. Because the images are not square, each one is split into two tiles of 512×512 pixels. To increase variety in the dataset, the same data augmentation used for the *planar-dataset* is applied. Additionally, the images are cropped with varying offsets so that the wires do not always appear centered. Finally, the dataset is split into 80% training and 20% validation subsets. Figure 5.5b shows example images from the *5-axis dataset*.

5.2.4 Segmentation

The first step in the error detection process is to segment the printed wires from the background. In our previous work [24], a simple SVM was used for this task. While this approach worked for the layer-based images, it struggled with low-contrast images and failed completely on monochrome data. However, reliable segmentation is crucial for the presented error detection approach. Therefore, a more robust method is required. Neural networks are well suited for image segmentation and can adapt to different substrate materials, lighting conditions, and machine types. The neural network's task is to classify each pixel in the image as either part of a wire or part of the background. As multiple architectures are available, four common variants were selected for evaluation on both datasets: U-Net, U-Net++, DenseNet, and DeepLabV3+, as described in Section 2.3. All four are convolutional neural networks suited to segmentation tasks and have been successfully applied in various domains. For training, all models use the binary cross-entropy loss and the Adam optimizer with a learning rate of 0.001.

U-Net Two U-Net configurations with different depths were tested. Both configurations start with 16 feature maps, and the number of feature maps is doubled after each downsampling step. Each convolutional layer uses a dropout of 0.2 to improve reliability, and ReLU is used as the activation function. The first configuration, referred to as *shallow*, has three down- and upsampling layers, resulting in a 128 64×64 feature maps in the bottleneck and a total of 482,033 parameters. The second configuration, referred to as *deep*, has four down- and upsampling layers, resulting in a 256 32×32 feature maps in the bottleneck and a total of 1,941,105 parameters.

U-Net++ The U-Net++ architecture keeps the overall structure of U-Net, but adds additional skip connections between the intermediate downsampling and upsampling stages, which increases the number of trainable parameters in the network. Accordingly, the *shallow* configuration has a total of 552,113 parameters, while the *deep* configuration has a total of 2,261,889 parameters.

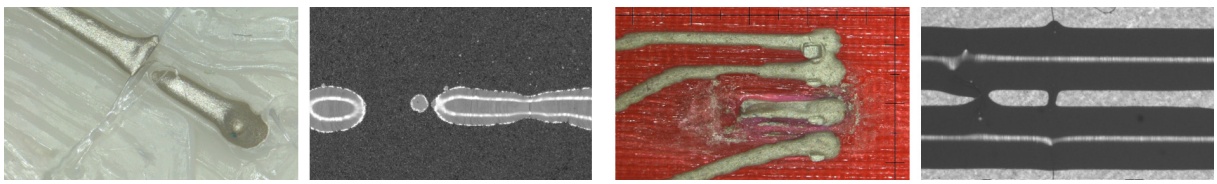
DenseNet The DenseNet architecture is evaluated in two configurations with different depths. In both, a dropout of 0.2 follows every convolutional layer and ReLU is used as the activation function. The first configuration, referred to as *shallow*, contains four dense blocks (with 3, 4, and 5 layers) and a 7-layer bottleneck with 320 64×64 feature maps, resulting in a total of 1,273,697 parameters. The second configuration, referred to as *deep*, contains five dense blocks (with 3, 4, 5, and 7 layers) and a 10-layer bottleneck with 480 32×32 feature maps, resulting in 2,787,393 parameters.

DeepLabV3+ The DeepLabV3+ architecture is implemented as described in the original paper[67]. It uses a ResNet50 backbone with a depth of 50 layers up to 1024 filters. As the backbone can be either trained from scratch with randomly initialized weights or with pre-trained weights from ImageNet [123], both configurations are tested. Both versions have 11,852,353 parameters.

5.3 Error Detection

Printed electronics are highly sensitive to errors, as the involved processes can be unstable. In addition, the substrate onto which the wires are printed is not always perfectly flat, which can introduce further defects. Figure 5.6 shows typical errors that may arise during the printing process. In traditional PCB manufacturing, defects are usually identified only after fabrication, and the PCB is discarded if anything abnormal is found. This practice is wasteful and impractical for the small batch sizes typical of 3D printed electronics. Moreover, once printed wires are covered by plastic, they are no longer accessible for probing or electrical testing in the finished object, as is common practice in PCB manufacturing. Afterward, the only non-destructive way to inspect such embedded structures is the use of expensive X-ray systems. Therefore, detecting errors during the printing process itself is crucial to ensure the functionality of printed electronics.

The error detection in this approach is based on images taken during printing and on the corresponding segmentation masks produced by the neural network. To obtain the intended wire



(a) **(b)** **(c)** **(c)**

Figure 5.6 – Common errors in printed electronics: examples from the *planar-dataset* **(a)** & **(b)** and the *5-axis dataset* **(c)** & **(d)**. Connection breaks caused by an obstacle on the surface **(a)** and by insufficient material flow **(b)**. Shorts resulting from excess material at the end of a wire **(c)** and from wires printed too close together **(d)**.

paths, the G-code file of the current object is parsed. All conductive path segments are extracted from the G-code and stored for further processing. Because wires in the G-code are represented as individual straight line segments, these segments must be grouped into connected components that represent continuous wires. Grouping is performed by identifying segments whose endpoints coincide with endpoints of other segments. Two endpoints are considered to be at the same position if they lie within a defined distance threshold. This threshold is set to half the wire width, since wires are expected to expand to this radius and thus be connected by design. Intersections between segments are also checked to account for wires that may cross each other or form a junction. To evaluate the segmented wires, their coordinates must be transformed into pixel coordinates, taking into account the camera position in the machine, the image resolution, and the measured pixels per millimeter. Once a consistent mapping between G-code coordinates and image coordinates is established, the segmented wires can be evaluated. The current implementation only supports images aligned with two of the main machine axes. Projecting the G-code into images with arbitrary orientations is possible but not yet implemented. With the wire segments corresponding to the current image and their segmentation represented in pixel coordinates, the error detection can begin. Currently, three types of errors are detected: connection breaks, shorts, and unreached points.

Connection Breaks Connection breaks are unwanted interruptions in the conductive material. Common causes for connection breaks are:

- Clogs in the syringe or piezoelectric extruder nozzle
- Inconsistent extrusion caused by air bubbles in the material
- Non-uniform droplet formation in piezoelectric dispensers
- Thin FFF plastic strings crossing and interrupting the wire
- Cracks due to material shrinkage or mechanical stress
- Uneven surfaces causing adjacent wire segments to lose electrical contact

To detect connection breaks, the algorithm verifies whether the segmented wire forms a continuous path without interruptions. First, it takes the segmented image of the whole layer or of a single wire segment and applies an erosion to slightly shrink the wire region. The erosion helps compensate for small inaccuracies in the segmentation mask and removes thin bridges in the segmentation that do not exist in the actual wire. Next, the algorithm uses OpenCV to perform a flood fill starting from the first point of the wire. Flood fill recursively fills all connected pixels with the same color until it reaches a pixel of a different color or the image border. If a segmented connection exists between two points, the pixels along that connection are also filled. The algorithm then checks all points extracted from the G-code that are supposed to be connected to

(a) **(b)** **(c)**

Figure 5.7 – Error detection of connection breaks and shorts. The segmented wires are shown in grey, the filled region is filled in red, the wire paths extracted from the G-code are green, and the tested points are marked with circles. **(a)** A wire without connection breaks: all points are reached (blue circles). **(b)** A wire with a connection break: some points are not reached (yellow circles). **(c)** A wire with a short to a neighboring wire: points of the neighboring wire are also reached (yellow circles).

the starting point. If any such point is not filled, this indicates a connection break between the filled region and the unfilled point. If all end points are filled, no connection break is present in that wire, and the algorithm proceeds to the next wire. Figure 5.7a shows connection break detection, where the wire is filled in green, and all points are reached (blue circles). Figure 5.7b shows a connection break, where only part of the wire is filled red and some points are reached (blue circles) while others are not (yellow circles).

Shorts Shorts are unintended connections between two or more wires that are not meant to be electrically connected. Shorts can be caused by several factors like:

- Excessive material accumulating on the nozzle tip
- Leaking conductive material from the printhead
- Wires placed too close together
- Imprecise printhead motion
- Non-uniform droplet formation in piezoelectric dispensers
- Overspray (small unintended droplets not belonging to the wire)

To detect such shorts, the algorithm tests whether any point of the current wire is reachable via segmented conductive pixels from another wire that should remain isolated. For this test, the

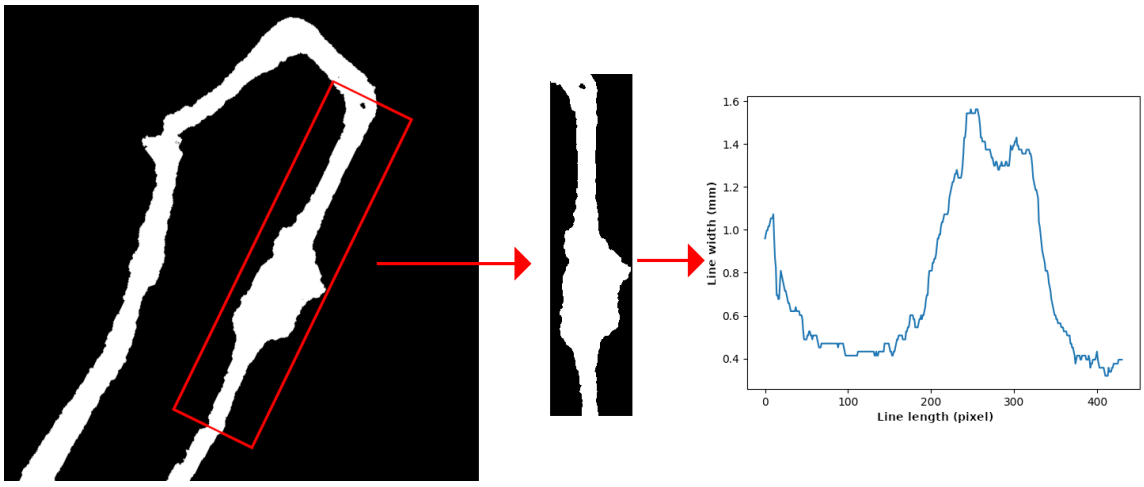
segmentation mask of the current wire is dilated rather than eroded to compensate for small inaccuracies and to bridge tiny gaps that in reality are connected. The algorithm then performs a flood fill starting from a point of the current wire. After the flood fill, every point belonging to any other wire group that should not be connected is checked to see whether it was also filled. If at least one point from a different wire group is filled, this indicates a conductive path between the wires and a short is reported. If no point belonging to any other wire is filled, no short is detected for the current wire and the algorithm proceeds with the next wire. Figure 5.7c illustrates a detected short: the current wire (filled in red) connects to the neighboring wire because that wire is also filled (yellow circles).

Unreached Points Unreached points are locations along a wire, including its endpoints, where the deposited material does not fully reach the intended position or the material is slightly misaligned. In contrast to a connection break, there are no two disconnected segments. The wire remains continuous but does not pass through all designated points or ends exactly at the specified endpoint. Typical causes include:

- Improperly calibrated printhead position
- Backlash in the motion system
- General positioning imprecision
- Surface tension effects in the conductive material at sharp corners
- Inconsistent material flow at the start or end of a wire

An unreached point is not necessarily a functional defect, since electrical continuity may still be intact. However, it is a deviation from the designed path and should be reported to the user. Unreached points are identified during the connection break detection step by examining each point along the intended wire path and checking whether it is classified as material in the segmentation mask. If a point is not classified as material, it is marked as an unreached point, indicating that no material was deposited at that exact location. This differs from a connection break, where the connection between two segments is lost. Here, only specific points along a continuous wire is not reached.

Wire Width Estimation The width of a wire is critical because it directly determines its resistance and impedance and thus its electrical performance. These quantities in turn determine how much current can flow through the wire and are especially important for high-frequency applications and antennas. A wire can be classified as correct by the previous tests, but if it is too thin it may not carry the required current or may have a higher resistance than intended. Therefore, knowing the actual width of a wire is important to ensure its functionality in the printed circuit.



(a) **(b)** **(c)**

Figure 5.8 – The wire width estimation process. One segment of the whole segmented wire (**a**) is extracted and rotated vertically (**b**). The number of segmented pixels is counted in each row to determine the width along the wire segment (**c**).

To estimate the width of a wire, the algorithm uses both the segmented image and the G-code path of that wire. Because wires in the G-code are described as straight line segments, each segment is evaluated individually and the per-segment results are then merged to obtain the complete width profile of the wire. For each segment, the corresponding region is cropped from the segmentation mask with an additional margin so that material extending slightly beyond the nominal path is still captured (Figure 5.8a). This margin is currently set to twice the expected wire width but can be adjusted for processes that typically produce wider or narrower traces. The cropped image is then rotated so that the wire segment is vertically aligned (Figure 5.8b), and the number of segmented wire pixels is counted in each row. This yields the local wire width along the segment (Figure 5.8c). After converting these width values to millimeters using the known pixel size, they are compared to the nominal width specified in the G-code. Once all segments have been processed and their measurements concatenated, the algorithm identifies regions where the wire is too thin or too thick and reports these deviations to the user. Because the pixel counting is performed perpendicular to the local wire direction, corners and crossings, where this assumption breaks, are excluded and not measured.

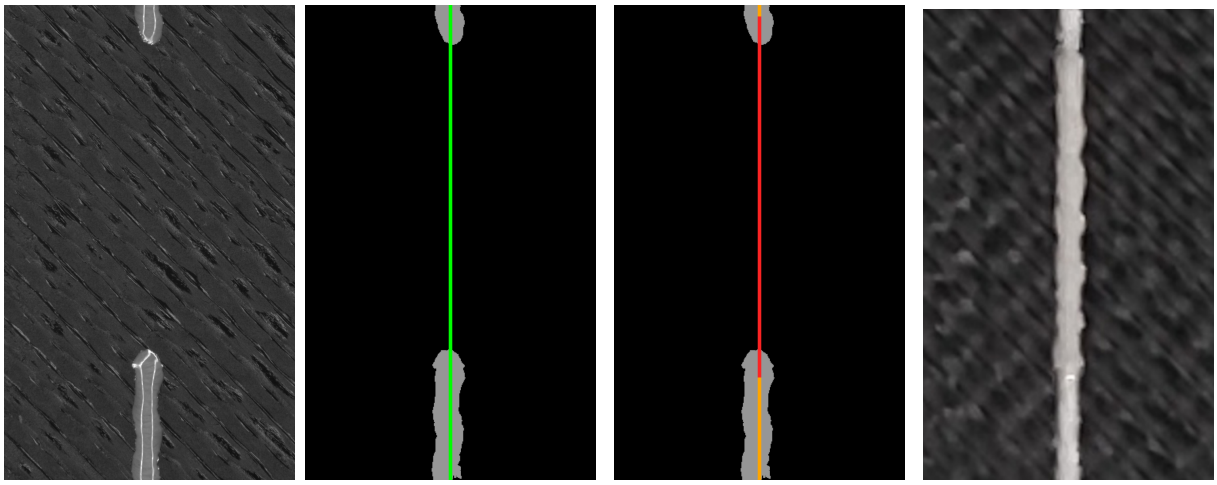
5.4 Repair Path Generation

Because defects in printed electronics almost always result in a non-functional circuit, repairing detected errors is essential to ensure the final object operates as intended. To address the errors identified in the previous step, a dedicated repair path is generated that the printer can execute to restore the affected regions.

Not all detected errors can be repaired with the current setup. Connection breaks can be repaired by reprinting across the gap to restore continuity between the separated segments. Unreached points cannot be corrected automatically because it is uncertain whether reaching that exact location is required for functionality. Their handling therefore depends on the specific context and may require user input to decide the appropriate action. Wires that are narrower than intended can be reinforced by adding an additional pass to increase their width, although this may result in a wire thicker than specified. Wires that are already too thick cannot be corrected with the present machine configuration and materials, since doing so would require removing material, something the current machine cannot perform. The same limitation applies to shorts, which would require removing unintended conductive material to eliminate the unwanted connection. Possible material removal methods would include CNC milling or laser ablation, but these are not part of the current approach.

5.4.1 Reprinting Wires

In the current setup, the only repair action the machine can execute is reprinting missing material. At present, only connection breaks can be reliably repaired this way. To generate a repair toolpath for a wire with a detected connection break, the algorithm must first localize the region where material is absent. To find the start of the missing segment, it steps pixel by pixel along the segmented wire from the first wire point until it encounters a pixel not classified as material in the segmentation mask. This marks the beginning of the connection break. It then continues iterating along the intended path until it reaches a pixel classified again as material, which marks the end of the break. The segment between these two points defines the region where material is missing and must be reprinted to restore the connection. Thus, the start and end points of the missing material are identified and form the basis of the repair path. To ensure a reliable reconnection, the repair path is extended slightly beyond both boundaries to create overlap with existing material. The extracted points are then transformed back into machine coordinates, and new G-code is generated to reprint the missing segment along this path. Other print parameters such as speed and extrusion rate are taken from the original wire G-code and adapted to the shorter repair path. Figure 5.9 illustrates this process for a wire with a connection break. Because multiple breaks may occur in a single wire, the algorithm repeats the procedure until the end of the wire is reached. The repair process is currently semi-automatic, as the user must upload the generated repair G-code to start execution. After the repair is performed, the error detection procedure can be run again to verify that the defects were resolved and no new ones were introduced. If no further issues are found, printing proceeds normally, and the printed electronics should function as intended provided no unrepairable errors remain.



(a) **(b)** **(c)** **(d)**

Figure 5.9 – Repair path generation for a connection break. **(a)** Original image with a connection break. **(b)** The segmented wire with the intended path in green. **(c)** The generated repair path in red. **(d)** The repaired wire after reprinting the repair path.

5.5 Evaluation

To evaluate the proposed in situ error correction system, the most suitable neural network architecture for wire segmentation was first determined by comparing several architectures on both datasets. Next, the error detection algorithms were applied to images with known defects to assess their reliability. Finally, the repair path generation was examined by repairing connection breaks and verifying the success of the repair.

5.5.1 Training Results

To identify the optimal neural network architecture for the wire segmentation task, several candidates were evaluated on both datasets. The evaluated architectures are U-Net, U-Net++, DenseNet, and DeepLabV3+. All networks except DeepLabV3+ were trained in two depth configurations: three (*shallow*) and four (*deep*) downsampling/upsampling blocks. DeepLabV3+ was trained once with pre-trained *ImageNet* weights and once with *randomly* initialized weights. Because the two datasets differ substantially, each architecture was trained separately on each dataset to determine the best-performing model per dataset. All models were trained for 20 epochs with a batch size of 4 using the Adam optimizer and a learning rate of 0.001. Training was executed on two 2080 Ti GPUs with 11 GB memory each. The best-performing epoch for each architecture and dataset is summarized in Table 5.2 and the training and validation Intersection over Union (IoU) over epochs for the two best models on each dataset are shown in Figure 5.10 and the remaining training curves are provided in Figure A.1 in the appendix.

Table 5.2 – Best-performing epoch for all trained models on the *planar* and *5-axis* datasets. For each model, the training and validation Intersection over Union (IoU) are reported, and the highest validation value for each dataset is highlighted in bold.

	planar-dataset		***5-axis dataset***	
Model	train (IoU)	val (IoU)	train (IoU)	val (IoU)
U-Net shallow	0.95	**0.91**	0.85	0.83
U-Net deep	0.91	0.89	0.95	**0.91**
U-Net++ shallow	0.95	**0.91**	0.82	0.76
U-Net++ deep	0.93	0.88	0.93	0.89
DenseNet shallow	0.98	0.00	0.97	0.14
DenseNet deep	0.97	0.00	0.95	0.15
DeepLabV3+ ImageNet	0.97	0.00	0.96	0.00
DeepLabV3+ random	0.97	0.00	0.96	0.00

The training results show that U-Net and U-Net++ perform well on both datasets. On the *planar-dataset*, the shallow U-Net and shallow U-Net++ achieves the highest score, whereas on the *5-axis dataset* the deep U-Net perform best. Overall, shallower variants work better on the *planar-dataset*, while deeper variants are advantageous on the *5-axis dataset*. A plausible explanation is the smaller field of view in the *5-axis dataset*, which makes the wires appear wider relative to the image and may benefit from the increased representational capacity of deeper models. It is also possible that the monochrome images in the *5-axis dataset* require more complex features to distinguish wires from the background, which deeper networks can better capture. DenseNet and DeepLabV3+ are overfitting on both datasets (good training IoU but very low validation IoU). Further tuning and more data might improve them, but this was not pursued because the U-Net-based models already provide high accuracy. Implementation errors are unlikely: the DeepLabV3+ code is a widely used public implementation, and the DenseNet implementation has already performed well in the segmentation in Chapter 6.

Overall, the neural networks achieved promising segmentation performance, despite being trained on relatively small datasets. They showed some misclassification at the image borders, which is typical for convolutional neural networks because successive convolutions reduce contextual information near the edges. To avoid this issue in the final segmentation process, the images were processed with overlap and the border regions were discarded. Although performance is good on both datasets, more diverse data would further improve robustness, especially on previously unseen structures.

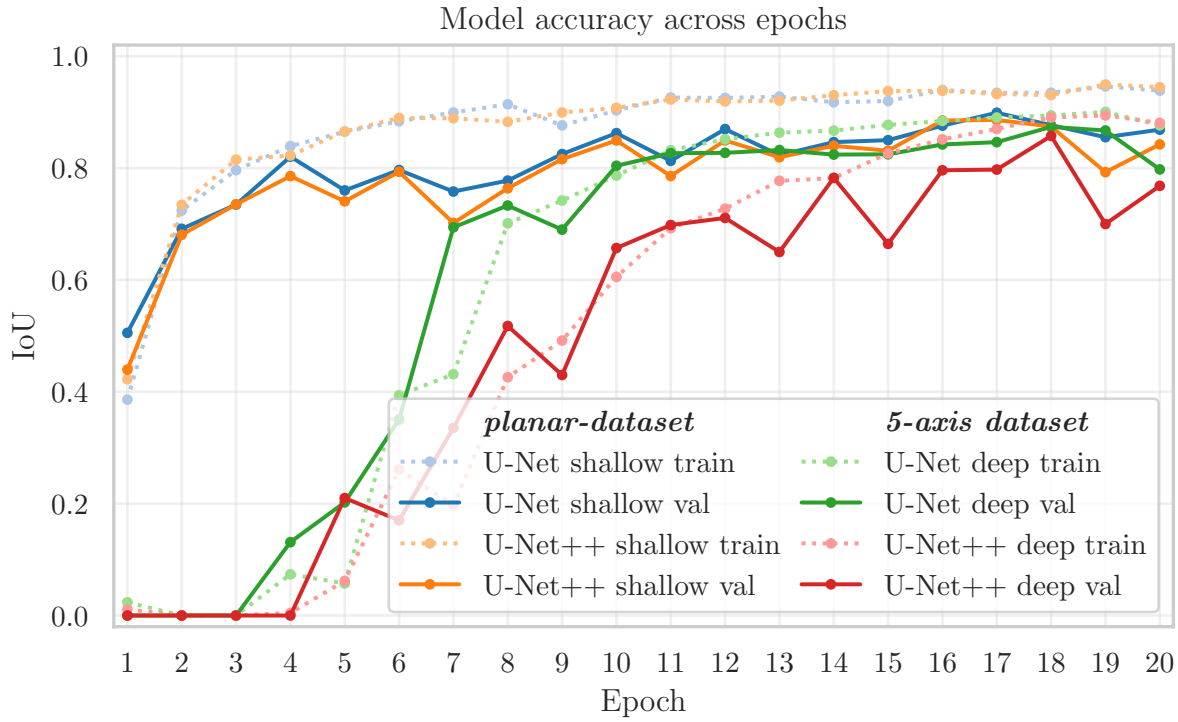


Figure 5.10 – Training and validation IoU across epochs for the top two models per dataset: on the *planar-dataset*, U-Net shallow and U-Net++ shallow, and on the *5-axis dataset*, U-Net deep and U-Net++ deep. The plots are averaged over five training runs with different random seeds.

5.5.2 Segmentation performance

To compare the performance of the neural network-based wire segmentation with the earlier SVM method [24], both were evaluated on a small set of previously unseen images similar in appearance to the *planar-dataset*. Table 5.3 reports the pixel-wise classification outcomes for both methods, listing the counts of true-positive, false-positive, and false-negative pixels, as well as the resulting IoU score. All percentage values are referenced to the total number of

Table 5.3 – Pixel-wise classification results for wire segmentation on unseen data similar to the *planar-dataset*. Shown are the results for the previous SVM approach and the best-performing U-Net model from this work. The results are shown as true-positive, false-positive, and false-negative pixels. The values are given in absolute numbers and percentages where 100% represents all conductive pixels in the image data.

Architecture	True-Positive	False-Positive	False-Negative	IoU
U-Net shallow	916,304 (96.6%)	60,008 (6.3%)	32,407 (3.4%)	0.91
SVM [24]	904,626 (95.4%)	279,645 (29.5%)	44,085 (4.6%)	0.73

conductive (wire) pixels present in the evaluated image set. A false positive denotes a pixel wrongly classified as wire although it belongs to background or plastic. A false negative denotes a pixel that is truly part of a wire but is misclassified as background or plastic.

The results indicate that the shallow U-Net architecture clearly outperforms the previous SVM approach. Although both methods achieve comparable true-positive and false-negative rates, the U-Net model produces a much lower false-positive rate than the SVM. This reduction in false positives results in a more accurate segmentation, which is essential for reliably detecting connection breaks. Precise identification of these breaks and missing material is particularly important, since they are currently the only error types that can be automatically repaired using the presented method. The SVM approach does not work on the *5-axis dataset* at all, because it relies on the Red Green Blue (RGB) and HSV color spaces, which are not available in the monochrome images.

The SVM approach particularly struggles in dark image regions and often misclassifies them as conductive material. In contrast, the U-Net model is less prone to these dark-area errors. However, if a dark region lies between two conductive wires, U-Net may still occasionally misclassify the gap as part of a wire and thus merge them. U-Net also handles reflections better than SVM, although very bright reflections can still lead to incorrect classifications.

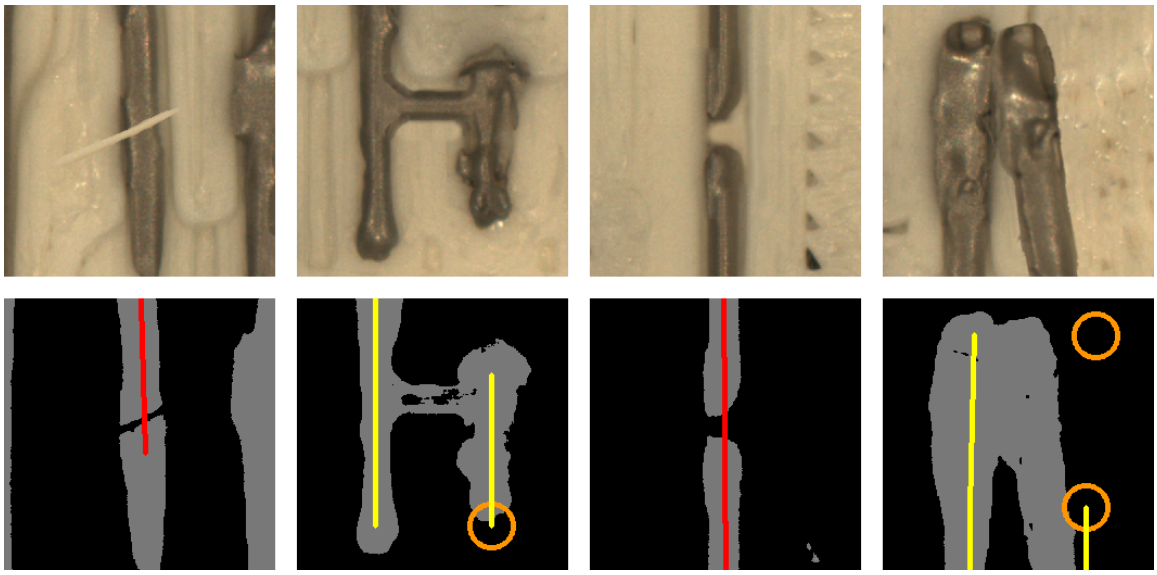
5.5.3 Error Detection

To evaluate the error detection algorithms, a set of wire images was generated, each containing different errors. All defects in this experiment were produced on the layer-based FFF printer, and the resulting images are similar in appearance to the *planar-dataset*. The objective is to assess whether the algorithms can reliably identify connection breaks, shorts, and unreached points in the segmented images. The algorithm is evaluated on its ability to detect the following defect types in the segmented images:

- Connection breaks caused by inconsistent extrusion
- Connection breaks caused by thin plastic strings from a leaking plastic extruder
- Intentional short created by adding an extra wire segment between two connections
- Intentional short created by shifting and rotating a wire so that it touches other wires

Since these experiments were conducted during the COVID-19 lockdown and no real images containing shorts were available, the short circuits were artificially generated by editing images in GIMP. The original images are not part of the training or validation data. Figure 5.11 shows the different defects together with the corresponding output of the error detection algorithms.

All connection breaks are correctly detected and highlighted in red. All created shorts are likewise correctly identified and shown in yellow. Unreached points are indicated with orange



(a) **(b)** **(c)** **(d)**

Figure 5.11 – Four different defects and their results from the error detection algorithms. **(a)** A connection break caused by a plastic string over the conductive trace. **(b)** A short circuit from an unintended printed line. **(c)** A connection break due to missing material. **(d)** A short circuit from a shifted conductive trace. The top row shows the original images and the bottom row the segmentation with detected errors. Connection breaks are marked in red, shorts in yellow, and unreached points with orange circles.

circles. In the image, three unreached points are detected. The first is at the end of a line where the deposited material does not reach the intended endpoint, most likely due to limited motion system precision combined with the material's surface tension. The other two unreached points are expected, as the material is not at its intended positions because the wire was artificially shifted in the image.

The layer documentation images are taken with the first-generation Raspberry Pi Camera as described in Section 3.1.3. Taking images of the whole layer with this camera adds about 10% extra time to the print time of that layer. The print time of a layer is further increased when conductive wires are printed, since the syringe-based extrusion process is comparatively slow. Segmenting each layer image with the trained neural network, and error detection on the segmented images requires less than 1 second. Thus, while the image documentation process noticeably increases the overall print time, the segmentation and error detection steps themselves have a negligible impact.

Overall, the error detection performed well: all presented errors were correctly detected. The algorithm remains robust across different wire geometries and substrates because it relies solely on the segmentation mask and the G-code describing the printed wires.

5.5.4 Repairing Errors

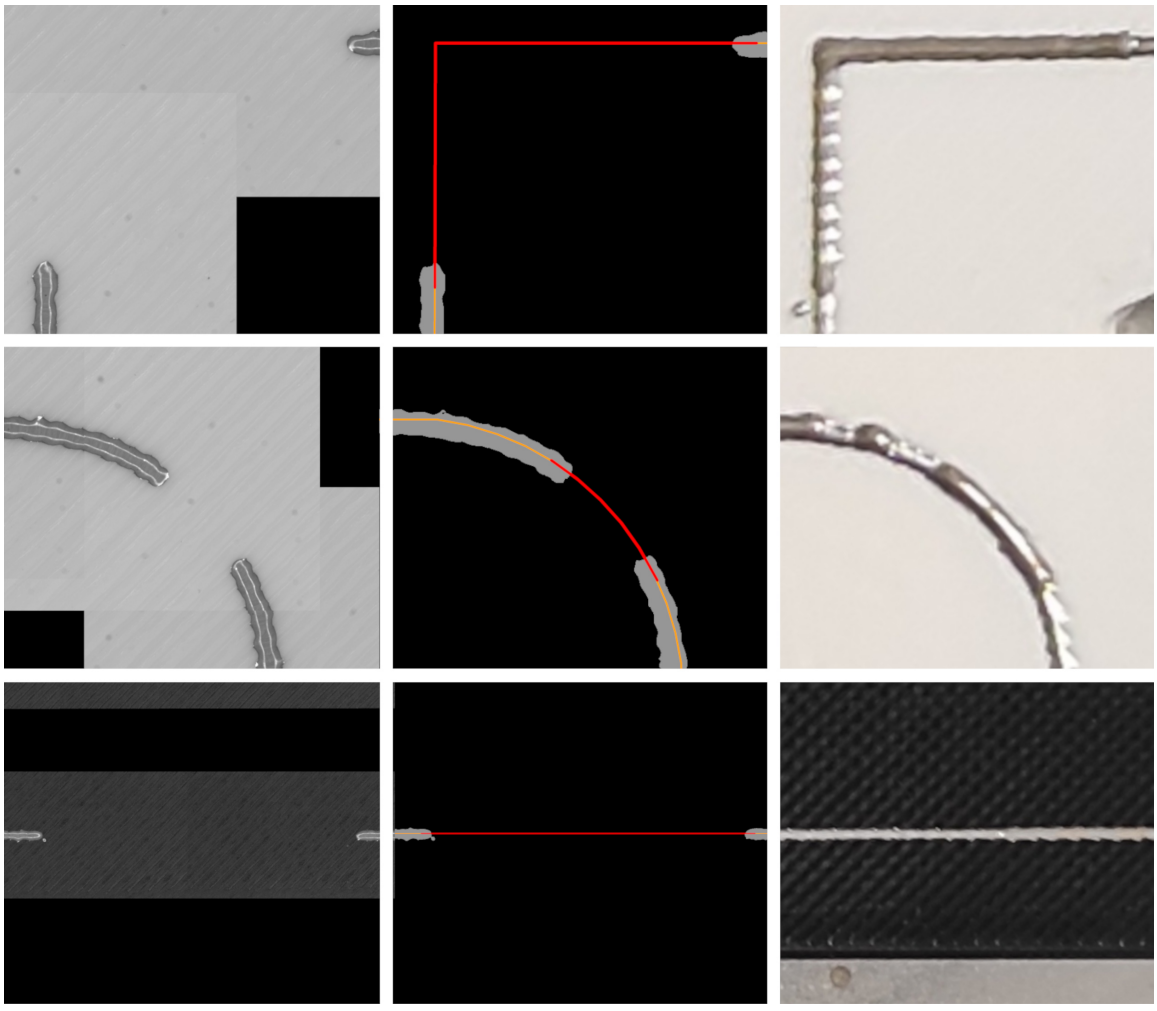
To evaluate the repair path generation and the overall repair process, a series of experiments was performed on the 5-axis printer of our project partner within the ZIM-KamEl project. Accordingly, the images used in these experiments resemble those in the *5-axis dataset*. The objective of these experiments is to demonstrate that the repair path generation can produce suitable repair toolpaths for connection breaks in printed wires.

The objects were created by generating multiple toolpaths in the Motion3D CAD/CAM software. The software's documentation feature was activated to produce image recording positions along the toolpaths. Using these positions, the machine captures images of the printed wires after each wire is deposited. These images are then segmented by the trained neural network, and the error detection algorithm is applied to the resulting segmentations.

The wires were produced on the Neotech PJ15X printer with a piezoelectric extruder. Connection breaks were created by deliberately switching off the piezoelectric extruder during printing, simulating realistic process faults. This procedure emulates real-world errors that may arise during the printing process. To evaluate robustness of the repair path generation algorithm, breaks were inserted at diverse positions along the wire paths: straight sections, corners, curved segments, and at both start and end points. Their lengths were varied to assess performance on small as well as large defects, ensuring coverage of different geometrical and positional variations. Experiments were conducted on three substrate types (Polyvinyl Chloride (PVC) sheets, white FFF-printed PETG, and black FFF-printed PETG) to confirm consistent behavior across materials and surface textures.

Figure 5.12 presents three representative connection breaks (a subset of all tested connection breaks) together with the corresponding generated repair paths and the resulting repaired wires after executing those paths. The first row shows a break at a corner, demonstrating that the algorithm correctly processes corners. The second row shows a break along a curved section, confirming that curved wires are also supported. The third row shows a larger break spanning multiple images, testing the algorithm's ability to handle more extensive defects across several images. For visualization purposes the images are stitched together, while the algorithm itself only combines the generated repair paths from the separate images. Repair path generation proved effective in all tested scenarios, producing appropriate repair paths for every connection break. Missing material is accurately identified, and sufficient overlap is included to ensure reliable connections. The generated repair G-codes were executed without modification, resulting in repaired wires that appear as intended. In most cases it is difficult to discern that these wires were repaired at all.

However, the experiments also revealed several limitations of the current approach. Because the camera's field of view is very small, image recording consumes considerable time during printing and must be improved for greater efficiency. In practice, the image acquisition often took longer than the preceding wire deposition, substantially increasing the overall print time.



(a) **(b)** **(c)**

Figure 5.12 – Three different connection breaks repaired with the proposed method. **(a)** Images captured during printing. **(b)** Segmentation results with the generated repair toolpaths. **(c)** Photographs of the conductive wires after repair.

Another issue was that the PiezoJet dispenser occasionally clogged while printing the repair paths, leaving the connection break unresolved. Clearing the clog always required manual intervention, which is not ideal for an automated workflow. To achieve consistent and reliable repairs, the dispenser's resistance to clogging needs to be improved. Furthermore, since execution of the repair path is currently started manually by the user and no additional image recording step follows it, unrepaired connection breaks remain undetected. At present, it is therefore not guaranteed that the repair was successful.

Lastly, the current implementation of the error detection and the repair path generation is limited to planar circuits aligned with the machine's main axes (XY-plane, XZ-plane, and YZ-plane). To correct circuits printed on freeform surfaces, a mapping from machine coordinates to image coordinates and back to machine coordinates must be implemented to support

all five printer axes. This would allow the algorithm to handle more complex geometries and orientations, making it truly applicable to 5-axis printed objects.

5.6 Conclusion

This chapter introduced a novel method to segment printed wires in images, detect defects in printed circuits, and generate repair toolpaths for connection breaks. Multiple neural network architectures were assessed for the wire segmentation task (U-Net, U-Net++, DenseNet, DeepLabV3+). Among these, U-Net and U-Net++ delivered the best segmentation accuracy on both datasets. The resulting segmentation works reliably on both monochrome and color images, making it applicable across different substrates and printing processes. Compared to the earlier SVM-based method, the neural network approach is more robust and produces far fewer false-positives. A tendency toward overfitting is still observable, and expanding the training set with more diverse substrates and circuit geometries is expected to further improve stability and overall segmentation performance.

The error detection algorithm presented here reliably identifies connection breaks, shorts, and unreached points in printed wires. It also estimates the width of the printed wires, which is important for high-frequency or high-current applications. Because it relies solely on the segmentation mask and the G-code of the printed wires, it is robust with respect to different wire geometries and substrates. The entire error detection process is fully automatic and does not require user input or supervision. By comparing the intended G-code toolpaths with the actually printed wires, the system reports all detected errors. Additionally, all data is archived for high-resolution documentation, which is essential for quality assurance in printed electronics.

The presented repair path generation algorithm produces appropriate repair paths for connection breaks in printed wires. It locates the damaged segment in the G-code and derives a new toolpath to reprint the missing material, ensuring that the repaired wire properly reconnects to the existing circuit. This capability increases the robustness of 3D printed electronics by enabling in situ correction of connection breaks. At present, the repair workflow is semi-automatic and still requires user interaction to initiate the individual steps. Nevertheless, it can be integrated directly into the printing workflow to operate fully automatically, forming a closed-loop error correction system in which the machine detects and repairs errors without user intervention. Because several subsystems across the printing process are involved and have been adapted for this method, the current implementation is limited to the two setups described in Chapter 3. However, the general approach is applicable to any 3D printing system that can capture images of the printed wires and uses a G-code-based path planning system.

In conclusion, in situ error detection and correction substantially increases the reliability of 3D printed electronics and enhances their suitability for industrial applications. The capability to

repair defects during fabrication lowers waste, improves yield, and ensures that the final printed circuits function as intended.

5.6.1 Future Work

For future work, several improvements and extensions to the current approach are possible. Because the current image recording process is still quite slow, especially on the 5-axis printer, it is necessary to improve it. This could be achieved by using a faster camera, enlarging the camera's field of view, or adopting a different image recording method.

Segmentation performance can be improved by expanding the training dataset with a wider variety of substrates and circuit geometries. Greater diversity would increase robustness and reliability, especially on previously unseen structures. Additional improvements may be achieved by adopting more advanced architectures or by further fine-tuning the existing ones.

Currently, the error detection process does not take into account component placement or the connections between components and wires, although these are essential aspects of printed electronics. Inter-layer interconnects are likewise not evaluated in the current approach. Integrating component placement and interconnect verification into the error detection process would further enhance the reliability of printed electronics and help ensure that all functional requirements are met.

The dataset already contains images with arbitrary orientations, and segmentation works on these images. However, extending error detection to complex 5-axis printed objects would be beneficial. At present, the error detection essentially works only for mostly planar circuits aligned with the main axes of the machine. In the future, it would be beneficial to implement the mapping from machine coordinates to image coordinates and back again, thereby supporting real 5-axis printed objects.

At present, the repair process is limited to reprinting missing material to close connection breaks. Extending the system to handle additional error types, including those currently only detected or not yet detected, would be beneficial. Future work should therefore investigate methods tailored to these unresolved defect categories, as they cannot be repaired with the existing approach. For shorts, research should focus on subtractive techniques to remove unintended conductive material and thus enable the repair of short circuits in printed electronics.

The current repair workflow remains semi-automatic and still requires the user to trigger each step individually. To achieve a fully automatic repair process, the repair path generation and its execution must be directly integrated into the printing workflow. A closed-loop error correction system would then allow the machine to automatically detect and repair errors without user intervention, significantly improving the reliability of printed electronics and making them more robust against errors.

Chapter 6

Layer Segmentation and Object Reconstruction

FFF printed 3D objects are produced by executing pre-calculated motion commands that move the printhead to defined coordinates while extruding material. However, these commands are executed by the printer in open-loop control-essentially blind-with only minimal feedback available to the printer's control board. Modern printers can sense whether the nozzle temperature is correct or whether a stepper motor skipped a step and failed to reach its intended position. However, this provides no direct information about the object being printed. As a result, substantial discrepancies between the intended design and the actually printed object can go unnoticed during printing. The internal structure of the object remains unknown and may contain hidden defects like under-/over-extrusion, or a temporarily clogged nozzle. These errors significantly weaken the part along the Z axis, which is already the weakest direction in printed parts [124]. This poses a major challenge in 3D printing, especially for safety-critical parts in aerospace or medical applications. Currently, inspection in FFF printing typically relies on visually checking the outer perimeter, monitoring the extrusion system, or using slow or low-resolution layer scanning methods. High-resolution inspection of internal structures can only be performed with expensive Computed Tomography (CT) scans [125] or destructive testing. If the actually printed structure could be captured, it would enable creation of a digital twin of the printed object. This digital twin can be be used for quality assurance, geometric accuracy assessment, for stress analysis, and as a basis for certification of safety-critical parts.

This chapter presents a novel method to create this digital twin of the actually printed object without requiring expensive hardware. It captures images of every printed layer and segments them with a neural network to identify the regions deposited in each layer. After that, the sequence of segmented layers is reconstructed to form a 3D model of the printed object. This reconstructed model functions as a digital twin of the physical print. In future work, this digital twin can be compared with the intended CAD model to identify discrepancies and defects in the printed object. Figure 6.1 illustrates the overall reconstruction process.

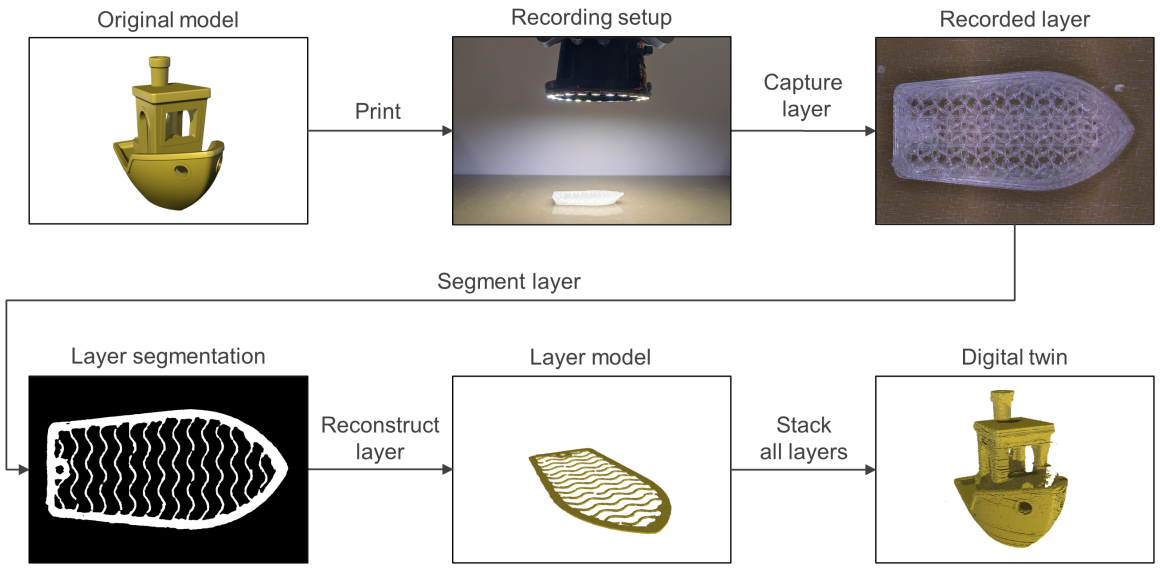


Figure 6.1 – Overview of the reconstruction process. Images of each printed layer are captured during printing. A neural network segments the images to identify the deposited regions in each layer. Finally, the segmented layers are stacked to reconstruct a 3D model of the printed object, which serves as its digital twin.

This chapter is based on the submitted publication (currently under review):

- Daniel Ahlers, Niklas Fiedler, Florens Wasserfall, and Jianwei Zhang. "Camera Based In Situ Layer Segmentation and Object Reconstruction for Digital Twins in FFF 3D Printing". Submitted to: *Journal of Intelligent Manufacturing*, 2025 [21]

6.1 State of the Art

Errors and defects can arise in 3D printing due to the layer-by-layer process and the geometric complexity of the parts. These errors may be caused by material properties, machine parameters, or environmental conditions. To ensure the quality and reliability of printed objects, it is essential to detect them. The earlier an error is detected, the easier it is to correct or to stop the printing process and avoid further defects. In additive manufacturing, error detection is challenging because of the sequential nature of deposition and the complex geometries involved. The main challenges include limited understanding of the materials used, insufficient standards for mechanical and non-destructive testing, and difficulties in defect characterization and detection in geometrically complex parts [126]. To address these issues, various approaches have been proposed that try to detect issues during printing, stabilize the process, and make printed parts more reliable.

One issue that can arise during printing is an offset in the mechanical positioning of the printhead. Such offsets cause misalignments between successive layers and can produce defects in the final object. Jeong et al. [127] presented a method that tracks the printhead motion with

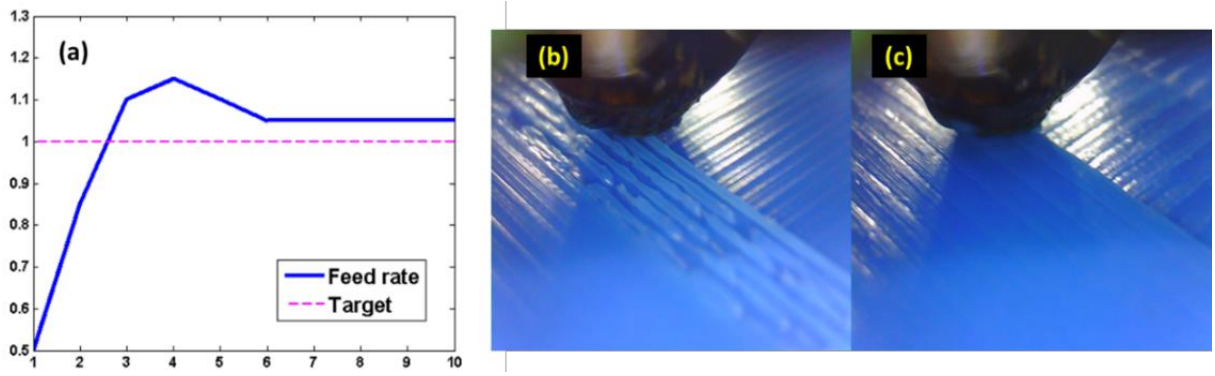


Figure 6.2 – Camera based nozzle flow monitoring: **(a)** The controller adjusts the nozzle flow. **(b)** Initial under-extrusion is detected by the camera. **(c)** The surface with optimal flow after the controller adjusted the flow rate. [131]

a camera and compares it to the commanded G-code path. By reconstructing the printed model and comparing it to a laser-scanned reference, they were able to detect and correct printhead motion offsets. Another approach to improve path accuracy uses a low-cost closed-loop controller that measures the actual printhead position with a linear magnetic encoder [128]. Additionally, some printers integrate closed-loop kinematic control to compensate positioning errors in real time [W24, W25], or monitor stepper motor feedback to identify skipped steps [W9, W10]. All these methods aim to prevent or detect errors introduced by the mechanical positioning system of the printer.

Another frequent source of issues in 3D printing is over- or under-extrusion, and material clogging in the printer's deposition system. To detect these faults, several methods have been proposed. Kim et al. [129] monitor the supply current that drives the material feed to identify defects such as nozzle clogging and thermal substrate deformation without placing a sensor near the high-temperature nozzle. Tlegenov et al. [130] introduced a technique to detect nozzle clogging in FFF 3D printing using a vibration sensor that measures real-time slipping of the filament in the extruder.

Another approach is to use cameras to monitor the printing process. The extrusion at the nozzle can be observed with a camera placed nearby to detect whether material is flowing as intended [132, 133]. Jin et al. [134] placed a camera close to the nozzle, extracted real-time images, and used a convolutional neural network to identify over- and under-extrusion. Liu et al. [131, 135] proposed a Proportional-Integral-Derivative (PID)-controller to close the control loop and correct extrusion during printing. As shown in Figure 6.2, the camera captures images of the extruded layer, which are then processed to correct the nozzle flow rate in real time. Brion et al. [136] combined these ideas and developed a generalizable 3D printing error detection and correction system using a multi-head neural network. The system enables real-time detection and rapid correction of errors such as flow rate errors, lateral speed errors, Z offset errors, and hotend temperature errors. When mounted from the side, the camera view of the extruded contours can

be analyzed to detect missing layers or failed overhangs [137]. In Electron Beam Melting (EBM) processes, the melt pool can be monitored with a combination of camera and sensors to make real-time adjustments that ensure consistent geometries and mechanical properties [W26].

Because the material is melted before extrusion in most printing processes, its temperature can be monitored to detect extrusion errors. Thermal imaging has been used to detect freshly deposited material and compare it with the G-code [138]. It has also been applied to identify excess material or large voids in the printed object [139]. In metal 3D printing, the melt pool temperature can be monitored to detect deposition errors [W27, W28]. Melt pool temperature measurements can also be used to control laser power and optimize the deposition process [140]. But thermal cameras either have a very low resolution or are very expensive, making them impractical for many applications.

Monitoring the motion and extrusion system can detect some errors, but it does not identify geometric defects in the printed layer. To locate such irregularities, an image of the entire layer can be captured after it is printed [141, 142]. This approach, however, does not account for the fact that the previous layer may still be visible in the image. To address this, a mask generated from the G-code can be applied to separate printed from non-printed regions [143]. This assumes, though, that the mask matches the actual printed layer precisely, which is usually not the case. Petsiuk et al. [144] proposed a method that uses a Gaussian Mixture Model to cluster regions of similar texture in the image. These regions are then analyzed with an Agglomerative Hierarchical Clustering Analysis (AHCA) to detect anomalies in the printed layer. They demonstrated that this method can detect holes in the perimeter and blobs in the infill of the printed object. The resolution of their setup is relatively low at 175 µm per pixel. Bowoto et al. [145] compared the image of the printed layer with a theoretical image generated from the G-code using image subtraction to highlight differences in the printed layer.

Most camera-based approaches lack depth information of the printed layer. However, this depth information is very useful for identifying the actually deposited geometry and its height at specific points. To obtain depth data, a point-based laser scanner can be used to scan the printed layer [147]. While the in-plane (layer) resolution depends on the number of sampled points, the achievable height resolution is about 10 µm. A faster alternative is to employ a line scanner to capture the layer [148]. The scanned layer can then be compared to the theoretical layer derived from the CAD model using a convolutional neural network to detect over- and underfill [149]. Another method projects a line onto the printed layer and observes it with a camera to generate a depth image of that layer [150, 151]. This depth image can be converted into a point cloud and compared with a theoretical point cloud generated from the G-code [152]. Such point clouds can also be produced using structured light with a camera mounted above the build area [153, 154]. The resolution of these setups is around 100 µm per pixel. Using a telecentric lens together with a structured light projector improves the resolution to 15 µm per pixel [155]. Singh et al. [146] integrated a custom line scanner into a 6 DOF arm to scan the printed layer and actively correct errors during printing. Their system can detect and repair both positive and negative defects in

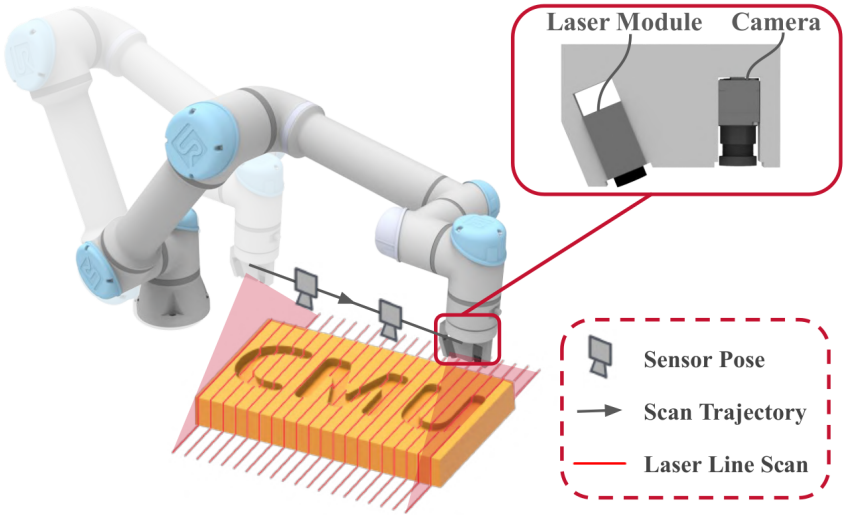


Figure 6.3 – The line scanner based layer segmentation system. The UR5 robot arm moves the line scanner over the printed layer to capture it. The captured layer is then segmented and compared to the theoretical layer to identify defects. Finally the defects can be repaired. [146]

the printed layer while avoiding regions with obstacles. The resolution of this setup is about 270 µm per pixel. To achieve the required accuracy for the deposition as well as the scanning, they scaled the whole system (nozzle diameter, line width, and layer height) by a factor of 10 compared to standard 3D printers. Figure 6.3 shows their system based on a UR5 robot arm.

In other 3D printing processes besides FFF, layer detection can be simpler, because in some systems only the most recently produced layer is exposed while the previous regions remain covered by powder or resin. Caltanissetta et al. [156] used a camera in Laser Powder Bed Fusion (LPBF) to capture the printed layer and compared its contour with the theoretical layer derived from the CAD model. The inkbit system [W29] similarly scans the printed layer after deposition with a camera to generate a topographical map of that layer. This map is then used to adjust the subsequent layer to achieve an accurate object.

While the previous approaches concentrate on detecting errors within individual printed layers, other methods target defects in the finished object as a whole. Side-view images can be classified to determine whether a print is failing or successful [157], or to detect issues such as stringing [158]. Similarly, the Obico system [W30] applies machine learning to side-mounted camera streams to identify problems like layer shifts, tangled webs of plastic (spaghetti), and other common defects during printing. A fundamental limitation of a single side camera is that only the regions visible from that viewpoint can be analyzed. To address this, multi-view setups with two cameras [159], five cameras [160], or even a rotating camera [161] are employed to obtain more complete coverage of the object. These multi-camera (or multi-view) systems compare captured images with synthetically generated renderings from the CAD model at matching viewpoints. The differences are then analyzed using convolutional neural networks to detect discrepancies. Because all these whole-object monitoring strategies remain restricted to visible

surfaces, Orth et al. [162] introduced an approach that uses scattered light in translucent objects to reconstruct a volumetric 3D model, including interior structures that are not externally visible.

With all these detection approaches, the central question is how to utilize the gathered information. One approach is to feed the sensor data into a neural network to predict the strength of the printed part [157]. After defects are detected, their impact can be assessed with a convolutional neural network together with a support vector machine [163]. This enables either cancelling a print mid-process when defects are critical or continuing it if the detected defects are not impactful.

Several challenges in existing methods remain and must be addressed in future research. Approaches that focus on monitoring the extrusion system cannot detect geometric errors in the deposited layer. Current techniques for observing printed layers often either fail to clearly distinguish the newly printed layer from the previous one, or they rely on masks for segmentation whose accuracy is limited. Some solutions depend on costly hardware, such as laser scanners or structured light projectors with cameras. In many cases, the achievable resolution is insufficient to identify small defects in the printed layer. Methods that analyze the entire object from the outside are unable to detect internal defects hidden within the part. As of today, there is no existing method that can reconstruct the entire object with a high resolution, including its internal structures, using affordable hardware.

6.2 Data Recording

The objective of this work is to detect errors in printed structures using images captured during the printing process. So, the first task is to capture the data needed for that detection. This data must be acquired during the printing process to enable identification of defects inside the object that would otherwise remain hidden after completion. Only errors detected while the corresponding layer is being printed can be corrected. Internal defects discovered later cannot be repaired without damaging or destroying the part. Initial experiments were performed with the Raspberry Pi Camera, but the image quality proved insufficient for the segmentation task. Subsequently, the Raspberry Pi HQ Camera with a standard lens was tested. However, noticeable optical distortions in these images made accurate layer segmentation difficult. Ultimately, the Raspberry Pi HQ Camera combined with a telecentric lens was used, providing highly accurate, dimensionally reliable images. In parallel, a depth camera was employed to record depth information for the printed layers. All cameras and the hardware setup are described in detail in Chapter 3.

6.2.1 Image Acquisition

The layer images are captured with a camera mounted as a separate tool in the E3D Toolchanger. After each layer is completed, the OctoCameraDocumentation plugin [W19] records the images,

similar to the procedure described in Section 5.2. To use all features of the Raspberry Pi HQ Camera, MJPGStreamer was replaced by a custom image server that can preprocess the images before sending them to the OctoPrint server. To keep lighting constant throughout a print, the window blinds remain closed and the room light stays on for the entire duration. The grid calculation for image positions was modified so that every layer is photographed at exactly the same coordinates. This is achieved by determining the combined minimum and maximum X and Y extents over all layers and generating a single grid layout that is used for every layer in the print. This uniform grid allows images of one layer to be directly aligned with images of other layers. The drawback is that empty regions in layers are also captured, leading to an increased number of images taken and longer print times. Additionally, the OctoCameraDocumentation plugin was modified so that at each grid position it captures multiple images with different lighting patterns to enable photometric stereo and compare different lighting conditions for the segmentation task. In total, 6 images are taken at every grid position with the following light patterns:

1. All LEDs on (25% brightness) (referred as *all_led*)
2. Light from top
3. Light from left
4. Light from right
5. Light from bottom

   (referred as *four_directions*)
6. Red, green, and blue light from 120°angle (referred as *rgb_led*)

The images are stored using the filename pattern Layer_<layer>_Tile_< tile >_<light>.png in a folder specific to the print job. Because the tiles are derived from the object's size and the camera field of view, the total number of tiles strongly depends on the object dimensions. The frequent switching between the camera and the printhead significantly increases the overall printing time.

6.2.2 Depth Sensing

The Intel RealSense D405 depth camera, described in Section 3.1.3, was used to capture depth information of the printed layers. Images were acquired with the same OctoCameraDocumentation plugin version used for the telecentric camera. A dedicated image acquisition server based on the pyrealsense2 library was implemented to retrieve the depth frames. Since the depth camera's field of view is larger than the printed objects, exactly one depth image was recorded for each layer. This removed the need for tiled captures and therefore significantly sped up the printing process compared to the RGB image acquisition. The raw depth images contained substantial noise and many invalid (hole) regions. These artifacts were mitigated using the postprocessing functions provided by pyrealsense2 library . The processing pipeline first applies a temporal

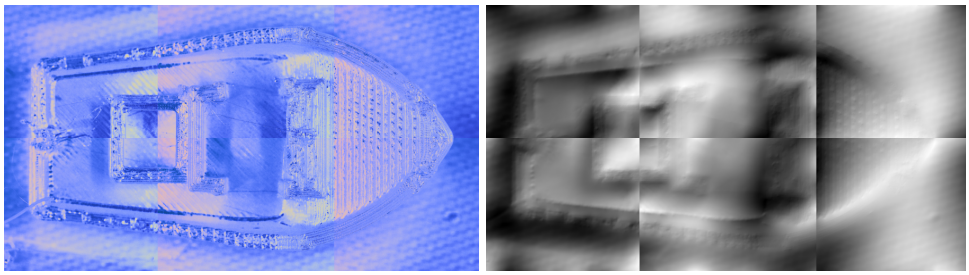
filter initialized (prefilled) with four frames of the object. This temporal filter uses parameters $\alpha = 0.3$, $\delta = 15$, and accepts a pixel only if it is valid in at least two of the four frames. After temporal filtering, a hole-filling filter is applied, inserting values from neighboring pixels closest to the sensor. A subsequent threshold filter retains only depth values within the 5-15 cm range. The filtered depth image is then scaled by a factor of 10 to better utilize the 8-bit dynamic range of a PNG image. The resulting PNG is stored as the layer's depth image in a folder specific to the corresponding print job. Because of its large field of view, the depth camera delivers an in-plane resolution of only 128 µm per pixel, which is insufficient to detect small defects in the printed layer. Although the depth (Z) resolution is specified as 100 µm, the actual measurements frequently deviate by more than 1,000 µm.

6.2.3 Photometric Stereo

Another approach to obtain depth-related information is to apply photometric stereo to the RGB images captured under different lighting conditions. Photometric stereo, introduced by Woodham in 1980 [164], uses multiple images of the same object illuminated from varying directions to estimate the surface normals. It assumes a Lambertian surface[165], meaning the surface reflects light uniformly in all directions, so the observed intensity depends on the angle between the light source direction and the surface normal. With a point light source, the measured intensity is proportional to the cosine of this angle. Using at least three images with distinct light source directions allows recovery of the surface normals. Using more than three light sources improves robustness and accuracy even further. The algorithm works as follows:

1. Capture multiple images of the same object under different lighting directions.
2. For each pixel, measure the intensity values from the images.
3. Formulate the intensity values as a linear system: $I = L \cdot n$, where:
   - I: vector of known intensities for a pixel.
   - L: matrix of light source directions $3 \times m$ where m is number of light sources.
   - n: surface normal vector.
4. Solve for n: $n = (L^T L)^{-1} L^T I$.
5. Normalize n to obtain unit surface normals.

The same computation is applied independently to every pixel, producing a surface normal map in which each pixel encodes the local surface normal. From this normal map, per-pixel gradients can be derived and then numerically integrated to reconstruct a heightmap. The resulting heightmap exposes fine surface features and supplies depth information (relative height variations) without the need for a dedicated depth sensor.



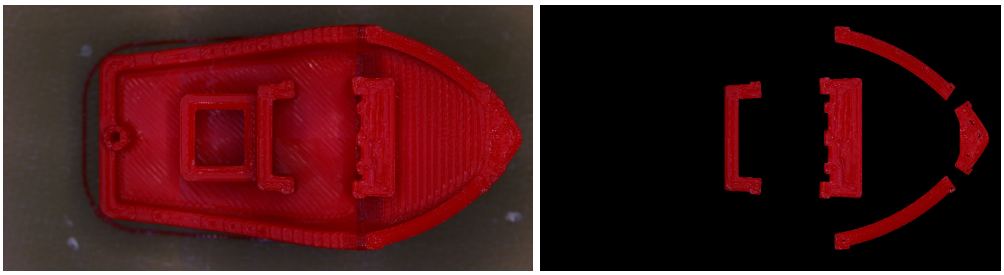
(a) **(b)**

Figure 6.4 – Stitched normalmap **(a)** and heightmap **(b)** of a printed layer generated with photometric stereo from four images taken with directional lighting. The normalmap shows the surface orientation encoded into the color channels of the image, while the heightmap shows the calculated height variations of the surface.

In reality, surfaces are not perfect Lambertian reflectors, and the light sources are not true point sources. In this setup, the short distance between the light sources and the object causes noticeable non-uniform light intensity across the scene. Moreover, the effective illumination angle varies across the field of view. Between the left and right image borders it differs by roughly 17°. These effects lead to incorrectly estimated surface normals that appear bent, similar to a fish-eye distortion of a camera lens. To mitigate this, a light correction factor is applied to the entire image. It is obtained by capturing multiple images of a white reference surface illuminated from each of the four directions. Each image is smoothed with a Gaussian filter, and a per-source light intensity map is computed. This intensity map is inverted and used as a correction factor for the corresponding source image that is added to each image. To account for the angular variation, the L matrix is then computed per pixel, incorporating the local angle between the light source direction and the surface normal at that pixel. Together, these two corrections produce more accurate surface normal estimates and a noticeably less warped normal map. Figure 6.4 shows an example of a stitched normal map and the corresponding heightmap generated from four images taken with directional lighting.

6.2.4 Dataset generation

To train a neural network that can segment layers in printed objects, a dataset with labeled images is required. The ground truth for all datasets is generated by reading the G-code files used to print the objects and generating a mask for each layer. These masks are created by transforming the printer's coordinates into image coordinates using the known camera field of view and pixel size. Currently, the line width must be fixed for the entire print because the ground truth generation cannot handle varying line widths. The generated mask matches the printed layer almost perfectly, though small deviations remain especially at the corners of the printed paths. Since the mask quality is sufficient for neural network training and there is too much data to correct



(a) **(b)**

Figure 6.5 – A full layer image **(a)** and the ground truth mask **(b)** generated from G-code overlayed with the image of the actual printed object. The mask matches the printed layer almost perfectly, but there are some small deviations especially at the corners.

manually, these minor deviations are not adjusted. Figure 6.5 shows an example of a generated mask overlaid on the image of the printed layer.

The first dataset was generated using the Raspberry Pi HQ camera together with the standard Basler lens (second camera generation). It includes ten different objects printed in six different colors. To ensure sufficient variability, the selected objects differ in geometry and size. All were printed with a 0.4 mm nozzle, a 0.3 mm layer height, and a 0.4 mm line width. No support structures were used so that the printed layers remained comparatively simple. The image acquisition grid was recalculated for every layer, so the tile positions did not align across layers. This is important because the network architectures described later require consistent inter-layer alignment, which this dataset does not provide. In total, 9,445 regions were recorded, each with four images per layer using *four_directions* lighting, resulting in 37,780 images. Even after distortion correction, the ground truth masks did not align precisely enough with the printed layers to achieve good segmentation performance. Additionally, further experiments required more lighting variants to extract more information from the printed layers. Also the first dataset suffered from inconsistent tile positions across layers. So a new dataset was recorded with the telecentric lens and additional lighting patterns.

The second dataset was captured using the telecentric lens and contained six different objects printed in three different colors. For this dataset, the layer tile positions were kept aligned across layers to enable experiments with multiple layer inputs. In total, 7,060 regions were recorded, each with five images per layer using *all_led* and *four_directions* light. This resulted in 35,300 images. Additionally, for each layer a normal map and a heightmap were generated with the photometric stereo approach described above.

The final dataset was generated in the same manner, but with more objects and a broader range of colors. A total of eleven different objects were printed in eleven different colors. Three different infill patterns were used. Altogether, 7,972 regions were recorded, each with six images per layer using the *all_led*, *four_directions*, and *rgb_led* lighting. This resulted in 47,832 images. In the initial tests, images from the previous dataset were temporarily included to increase the

Table 6.1 – Overview of the models name, the color used for printing, the infill pattern used, the number of layers printed, the grid size used for the image acquisition, the print time and the estimated time for printing without image acquisition. The final dataset is generated from these printed objects.

Model	Color	Infill	Layers	Grid	Print Time	Est Time
3DBenchy [W31]	brown	Gyroid	160	3 × 2	437 min	93 min
Beer crate [W32]	grey	no infill	173	3 × 3	741 min	206 min
Cookies cutter [W33]	blue	no infill	27	4 × 3	152 min	36 min
Motor plate [W34]	dark green	Cubic	39	2 × 3	140 min	59 min
Gear bearing [W35]	gold	Grid	50	3 × 3	273 min	118 min
Hex-Scraper [W36]	light blue	no infill	16	2 × 6	127 min	65 min
Octopus [W37]	red	Gyroid	89	4 × 4	465 min	117 min
Easter egg [W38]	white	Grid	213	2 × 2	431 min	85 min
Measuring-Tool [W39]	black	Cubic	35	3 × 3	206 min	102 min
Spiral-Planter [W40]	yellow	no infill	160	3 × 3	630 min	173 min
Whistle [W41]	green	Cubic	74	2 × 8	314 min	110 min

dataset size. They were later removed to keep the results comparable, since the earlier data did not contain *rgb_led* images. Table 6.1 and Figure A.3 in the appendix show an overview of the datasets used for training the neural networks.

The raw images have a resolution of 4056 × 3040 pixels with a pixel size of 6 µm, which is far higher than required for the segmentation task and impractical for neural network training. To reduce data volume, all images are uniformly downscaled to 2028 × 1520 pixels, giving a pixel size of 12 µm (83.3 pixels per mm), which remains sufficient for segmentation. The downscaled images are then divided into 512 × 512 pixel tiles with a minimal overlap of 5 pixels to ensure full coverage. This scaling and tiling process is applied to all image types, including the *normalmap* and *heightmap*. As some tiles do not contain any printed material, 99% of these empty tiles are discarded to focus training on relevant regions. In total, this produces 68,809 tiles per image type for the final dataset used to train the neural networks described in the next section.

6.3 Layer Segmentation

3D printed objects are built layer by layer, each layer deposited sequentially. Every layer is a two-dimensional slice with a defined thickness. The complete object geometry can be reconstructed from the collection of all layer outlines. Accurately identifying the material belonging only to the current layer is difficult because parts of the previous layer are still visible in the captured images. These lines from the previous layer look very similar to the newly printed lines, making it challenging to distinguish between them. Consequently, conventional image process-

ing alone often fails to produce a clean separation. To overcome this, several strategies have been investigated for segmenting printed layers. Their shared objective is to isolate the current layer from the one beneath it to enable precise reconstruction. The explored methods comprise depth sensing, photometric stereo, and neural network-based segmentation.

6.3.1 Depth Sensing

The first approach explored was using a depth-sensing camera to segment the printed layers. This idea was motivated by researchers such as Singh et al. [146], who used a line scanner to scan each printed layer and actively correct errors during large-scale printing. The plan was to capture depth images of every layer and derive the segmentation from the depth information. However, the selected depth camera could not deliver sufficiently reliable depth data. Its images exhibited substantial noise and many invalid (hole) regions, which prevented accurate layer segmentation. In addition, the spatial resolution was too low to identify small defects in the printed layer. At present, there is no known affordable depth camera that offers the necessary combination of resolution and accuracy for this task. Therefore, this approach was discontinued in favor of further experiments with the RGB camera. Figure 6.6 shows a depth image of a printed layer in which the object can only be vaguely recognized.

6.3.2 Photometric Depth Images

The second approach applied photometric stereo to derive per-pixel surface normals and a relative *heightmap* for each printed layer. The intention was to use this depth-related representation to separate the current layer from the one below. Although the resulting *heightmap* shows prominent model edges and larger geometric transitions, its accuracy is insufficient for direct layer segmentation. The apparent height variations within a single layer exceed the nominal layer thickness especially when regions of the same layer are not connected (e.g., the hull and the house of the 3DBenchy in Figure 6.5). Without a stable, layer height, a purely height-based segmentation and therefore a reliable layer extraction is not feasible. However, the *heightmap*

Figure 6.6 – A depth image of the same area as in Figure 6.5 taken with the Intel RealSense D405. While the printed layer is clearly visible on the RGB image, the depth image is very noisy, and the object can only be vaguely recognized.

together with the *normalmap* still encode shape and shading cues not present in a standard RGB image and may therefore serve as additional inputs to a neural network to improve layer segmentation.

6.3.3 Neural Network Segmentation

The third and final approach for segmenting the layers is to train a neural network to learn the layer segmentation directly. Because a single image does not contain enough information (the previous layer is still partly visible), the neural network requires additional context. The strategy is to supply both the image of the current layer and the image of the previous layer so the model can isolate the newly deposited material. To determine the best configuration, different neural network architectures as well as different input variants were tested. The evaluated architectures include U-Net [65], U-Net++ [66], and DenseNet [70]. Each network therefore has at least two input images: one of the current layer and one of the previous layer. The different inputs with the number of resulting input channels (in brackets) are:

- *all_led* (6): image of the layer with all LEDs on
- *four_directions* (24): four images of the layer with light from four directions
- *normalmap* (12): *all_led* with additional normalmaps
- *heightmap* (8): *all_led* with additional heightmaps
- *rgb_led* (6): image of the layer with red, green and blue light from 120°angle

All listed input variants are evaluated with every neural network architecture. The resulting number of input channels therefore depends on the chosen variant and ranges from six up to 24 channels (see list above). The input of the neural networks always have a shape of $512 \times 512, \times n$, where n is the number of input channels depending on the selected input variant. All networks use the ReLU activation function in every layer except the final layer, which uses a sigmoid activation. The output is always a 512×512 single-channel image representing the binary segmentation mask: background = 0, layer = 1. Training uses binary cross-entropy loss with the Adam optimizer with learning rate 0.001. The validation set was created by randomly selecting 20% of the images from the dataset. No data augmentation is used because the dataset size is considered sufficient. Since the reference implementations support at most three input channels, all architectures were reimplemented, following the original publications, with minor adjustments in Keras[W42] and Tensorflow[166]. The architectures were configured to have a comparable number of parameters to ensure a fair comparison.

U-Net The U-Net architecture consists of 4 down- and 4 up-sampling layers, with 32 feature maps in the first layer. Each down-sampling step doubles the number of feature maps, while each up-sampling step halves them. At the bottleneck, 512 feature map are used with a size of 32×32. SpatialDropout2D with a rate of 0.2 is applied in the decoder and at the bottleneck, while no dropout is used in the encoder. In this configuration, the model has a total of 7,761,537 trainable parameters.

U-Net++ The U-Net++ architecture likewise consists of 4 down- and 4 up-sampling layers, starting also with 32 feature maps. This also leads to a bottleneck of 512 32×32 feature maps. SpatialDropout2D with a rate of 0.2 is applied in the decoder and at the bottleneck, while no dropout is used in the encoder or on the nested skip connections. In this configuration, the model has a total of 9,043,617 trainable parameters.

DenseNet The DenseNet architecture has also 4 down- and 4 up-sampling layers, with 24 feature maps in every convolutional layer. The number of convolutions per dense block is 3 for the first block, and 4, 5, and 7 for the subsequent lower-level blocks in both encoder and decoder. The bottleneck has a spatial size of 32×32 with 10 convolutions and up to 720 feature maps. SpatialDropout2D with a rate of 0.2 is applied in the decoder and at the bottleneck, with no dropout in the encoder. Overall, DenseNet comprises 6,177,697 trainable parameters and 45,984 non-trainable parameters.

6.4 Object Reconstruction

After successfully segmenting the printed layers in the previous step, the next task is to reconstruct the object from this data to create a digital twin. The first step is to reconstruct each layer from the taken images by processing them with the trained neural network to generate a binary mask of the printed layer. Because the segmentation accepts inputs of 512×512 pixels, the images are split into tiles of this size with an overlap of 64 pixels. This overlap reduces edge artifacts and inaccuracies near tile boundaries that arise from limited context and padding in convolutional neural networks. The segmented tiles are then stitched together to produce a complete segmented image of the layer. This layer mask is used both to analyze the layer for errors and defects and to reconstruct the whole object as a digital twin of the printed part.

6.4.1 3D Reconstruction

As 3D-printed objects are built layer by layer, the reconstruction is performed in the same way by stacking the segmented layers on top of one another. Because the segmentation provides only the flat 2D layer footprint without any height information, the object is reconstructed by extruding each segmented layer to a specified height. This height is not taken from sensor data but is set

(a) **(b)** **(c)**

Figure 6.7 – 3D model reconstruction from segmented layers: **(a)** Stitched segmented layer mask from the neural network, **(b)** extruded layer at the correct height, and **(c)** final 3D model composed of all layers.

from the G-code used to print the object. In this setup, the reconstruction would be wrong if there is any issue on the Z axis, such as skipped steps on the Z axis. Apart from the layer height, no additional information from the print is needed to reconstruct the object. The reconstruction process is implemented with a Python script that calls external tools to generate 3D bodies from the segmented layer images and to combine them into one 3D model. The process is shown in Figure 6.7 and the steps are as follows:

1. Stitch the segmented tiles to generate a full layer image.
2. Filter the layer image with morphological operations to remove small artifacts and noise.
3. Apply Gaussian blur to smooth the edges of the segmented layer.
4. Apply a threshold to obtain a clean binary mask.
5. Take the mask image and convert it to an SVG using Potrace [W43].
6. Generate OpenSCAD [W44] code to extrude the SVG to the specified layer height.
7. Translate the layer to the correct height and XY position based on the layer number.
8. Combine all layers into one OpenSCAD file.

The resulting STL file is typically very large because it retains an unnecessarily high resolution of the object. To significantly reduce the file size, the STL is simplified using PrusaSlicer's integrated mesh simplification tool [W7]. Since this feature is not available via the command line, the STL must be opened in the GUI and simplified manually. The resulting model is a digital twin of the printed object that can be used for further analysis, such as measuring or comparing it with the original model.

6.5 Evaluation

To evaluate the proposed method for layer segmentation and object reconstruction, several aspects are examined. First, the additional time introduced by image acquisition during the printing process is measured. Next, results from different neural network architectures and input types are compared to identify the most effective approach for layer segmentation. To test generalization, the trained neural networks are applied to multiple unseen prints. The ability to preserve small details is evaluated by printing multiple copies of a small object with fine features. Finally, the accuracy of the reconstructed object is assessed by comparing measurements taken from the segmented model with those of the original object.

6.5.1 Image Acquisition Time

The images required for layer segmentation must be captured during printing. Otherwise, the printed layers structure would be covered by the next layer. For this approach, the FFF printing process is interrupted to switch to the camera tool. When a layer is finished, the printhead moves to the park position and the camera tool is picked up. Then the print bed is lowered to provide enough clearance for imaging. Next, the camera moves to the first acquisition position which takes 9.76 seconds on average. The LEDs are then switched on and the first image is taken. This takes on average 2.94 seconds for each image, including a safety delay to ensure that the printer has actually reached the position and there are no vibrations from the movement, as well as the actual image capture, the transfer over network, and saving to the disk of the controlling Raspberry Pi. With six images captured for the dataset, this amounts to about 18 seconds. This time decreases if fewer lighting patterns are used during acquisition. Moving the camera to the next tile position takes, on average, 0.20 seconds. The number of tiles depends on the object size. After all tiles are captured, the LEDs are switched off, the camera returns to the park position, and the FFF tool is picked up again. Since the print bed was lowered, it is raised back to the previous position. Because FFF printheads tend to ooze a small amount of material even when the filament is retracted, the nozzle must be cleaned before printing resumes. To do so, the extruder moves to the dumping and cleaning station, where the nozzle is primed and cleaned. Overall, this step takes 15.11 seconds on average. For a layer with four tiles, the total additional time for image acquisition is 97 seconds. This represents a significant increase in print time compared to printing without image acquisition. Table 6.2 shows the print times of the dataset objects with and without image acquisition and the corresponding percentage increase.

The per-layer print time strongly depends on the object's shape and size, as well as the infill density. In contrast, the documentation time per layer depends only on the number of image tiles required to cover the object. Thus, for a given object the layer documentation time is fixed, while the print time can vary significantly. This results in a particularly high relative time increase for small objects with fine structures or for mostly hollow objects, such as the *Easter egg*. The only

Table 6.2 – Print times of the dataset objects with and without image acquisition. No images is the time that the print takes without image acquisition, while with images is the time that the print takes with image acquisition. Time increase is the percentage of the time increase compared to the print without image acquisition.

Model	Layers	Grid	No images	With images	Time increase
3DBenchy [W31]	160	3 × 2	93 min	437 min	368%
Beer crate [W32]	173	3 × 3	206 min	741 min	260%
Cookies cutter [W33]	27	4 × 3	36 min	152 min	322%
Motor plate [W34]	37	2 × 3	59 min	140 min	137%
Gear bearing [W35]	50	3 × 3	118 min	273 min	131%
Hex-Scraper [W36]	16	2 × 6	65 min	127 min	**95%**
Octopus [W37]	89	4 × 4	117 min	465 min	297%
Easter egg [W38]	213	2 × 2	85 min	431 min	**407%**
Measuring-Tool [W39]	35	3 × 3	102 min	206 min	102%
Spiral-Planter [W40]	160	3 × 3	173 min	630 min	264%
Whistle [W41]	74	2 × 8	110 min	314 min	186%
Cube 100% filled	20	2 × 2	47 min	82 min	**74%**

object in the dataset where the actual printing still takes more time than the image acquisition is the *Hex-Scraper*. To demonstrate the minimal time increase due to image acquisition, a small box that covers exactly 2 × 2 tiles and is fully filled was printed. Even for this object, the print with image acquisition still takes 74% longer than a print without.

In its current state, the image acquisition process is too slow for regular use. The recent trend toward faster printers with higher speeds and shorter layer times further worsens this issue. There are multiple approaches to reduce image acquisition time. Currently, six images are captured for each layer to provide comprehensive data for neural network training. However, once the best-performing neural network architecture and input type have been identified, only the images required for that specific approach need to be acquired during actual use. This allows the number of images per layer to be reduced, minimizing acquisition time. Since the current resolution of 6 µm per pixel is much higher than necessary for accurate segmentation, the camera's field of view could be increased to lower the resolution. This would reduce the number of image tiles required to cover the object, significantly decreasing the total image acquisition time. The 2.94 seconds required to capture each image could also be reduced by implementing a faster image acquisition process.

A fixed, high-resolution camera could also be mounted above the print bed to capture a single image per layer. With this setup, a trade-off between field of view and image resolution is required, since the camera cannot be moved during the print. However, layer documentation

would take only a few seconds: the printhead just needs to be moved out of the field of view, and printing can continue immediately after the image is taken similar to time lapse photography.

6.5.2 Neural Network Training

To identify the best-performing combination of neural network architecture and input type, the models were trained on the generated dataset. Each architecture was trained on the same dataset with five input types: *all_led*, *four_directions*, *normalmap*, *heightmap*, and *rgb_led*. Training was run for 10 epochs with a batch size of 4, except for DenseNet, which used a batch size of 2 due to memory limitations. All trainings were performed on two NVIDIA RTX 2080 Ti GPUs with 11 GB of memory each together with an AMD Ryzen 9 3900X 12-core processor with 24 threads, and 128 GB of RAM. The training results are listed in Table 6.3, showing the IoU scores on both the training and validation sets . For each input type, the highest IoU score is highlighted in bold. As the U-Net++ architecture generally performed best, the training and validation IoU over epochs for this architecture is shown in Figure 6.8. The plots in the figure are averaged over three training runs with different random seeds to account for variability in training. The the training and validation IoU over epochs for all architecture and input type combinations are plotted in Figure A.2 in the appendix for reference.

Across all input types, U-Net++ consistently achieves the highest IoU scores on the validation set, with a best result of 0.937 using the *rgb_led* input. U-Net performs slightly worse overall but remains close to U-Net++ for *all_led*, *four_directions*, and *rgb_led*. DenseNet generally underperforms compared to the other two architectures, shows signs of overfitting, and trains significantly slower while using more memory. Although the difference between U-Net and U-Net++ is not large, U-Net++ is preferred for layer segmentation because of its consistently strong results. However, because the ground truth used for evaluation is derived from the G-code and does not show the physically printed layer, a mismatch can occur between the predicted layer and the ground truth. If the network has learned to predict the actually printed layer more accurately

Table 6.3 – Validation results for layer segmentation: IoU scores for different neural network architectures and input types, as well as the average training time per epoch. For each input type, the highest IoU score is shown in bold. Values are reported for the best epoch of each architecture.

	U-Net		**U-Net++**		**DenseNet**	
Input	train (IoU)	val (IoU)	train (IoU)	val (IoU)	train (IoU)	val (IoU)
all_led	0.911	0.922	0.917	**0.927**	0.954	0.680
four_directions	0.915	0.923	0.914	**0.924**	0.958	0.744
normalmap	0.789	0.811	0.916	**0.926**	0.957	0.778
heightmap	0.856	0.854	0.923	**0.934**	0.952	0.707
rgb_led	0.912	0.922	0.927	**0.937**	0.951	0.831

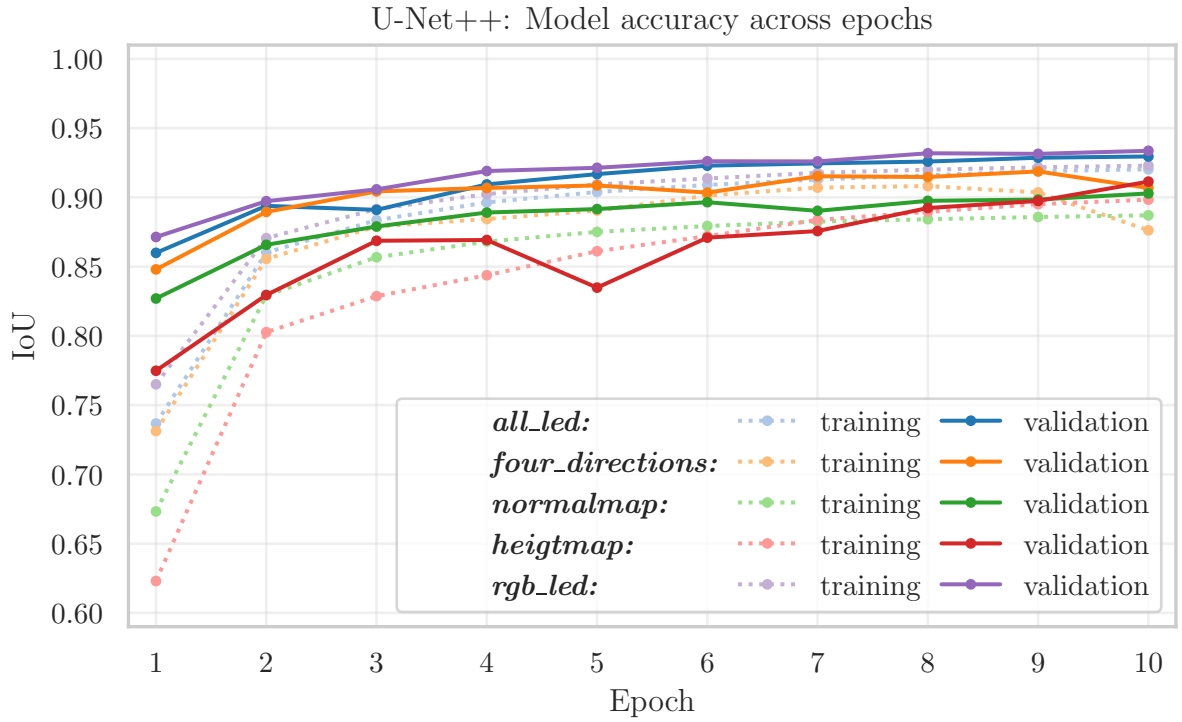


Figure 6.8 – Training and validation IoU across epochs for the U-Net++ architecture with different input types. The plots are averaged over three training runs with different random seeds. The plots are cut at an IoU of 0.6 for better visibility.

than the G-code derived ground truth, the IoU values reported here may underestimate the true segmentation performance and need to be interpreted with care. But currently there is no better metric available to quantify segmentation performance.

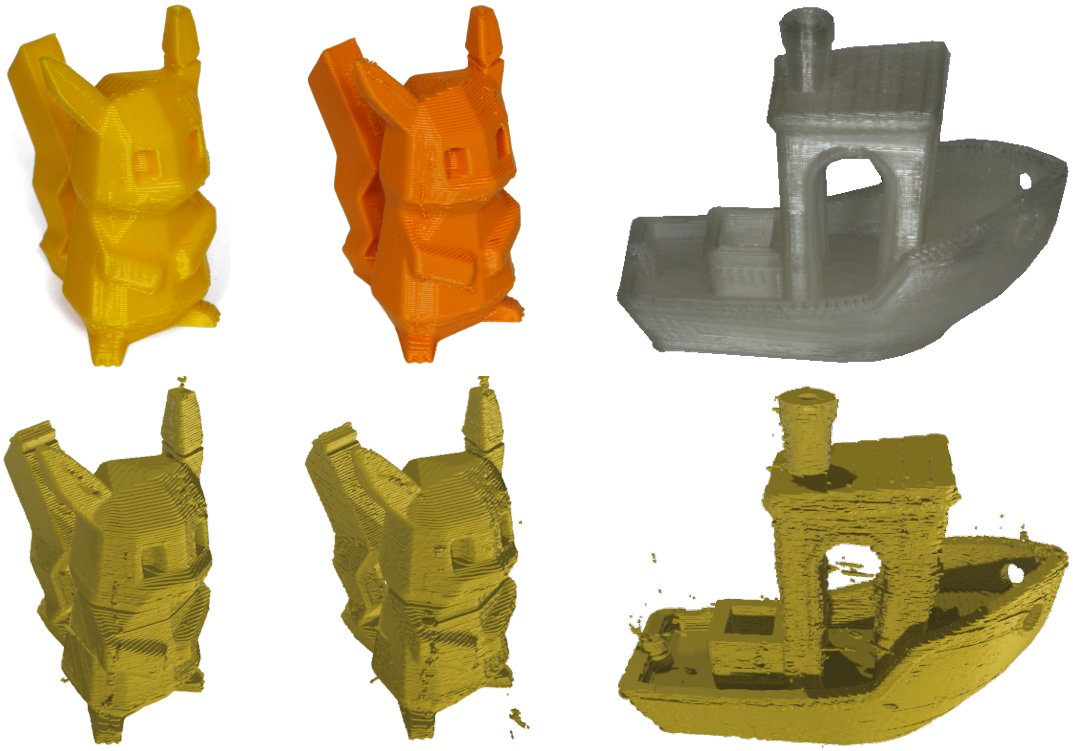
So one possible improvement for both the training and evaluation process is to use more precisely annotated ground truth masks. This could be achieved by using a depth sensor only for dataset generation, while still relying only on the RGB camera for the actual segmentation. Overall, the experiments with different input types show that neural networks can segment images of printed layers reliably. Including images captured under different lighting conditions and adding depth-related inputs slightly improves the segmentation, but these gains are not significant.

6.5.3 Generalization and Robustness Tests

Because the validation set is randomly sampled from the same dataset used for training, it includes images of the same objects and all their colors as in the training set. So the validation results do not indicate how well the trained networks generalize to unseen objects and colors. To evaluate this, a resilience test was carried out where three objects not present in the training dataset were printed.

The objects used for this test are as follows: First, a low-poly squirrel was printed in yellow PLA. Although yellow is present in the training dataset, this specific squirrel model was not included during training. Second, the same squirrel model was printed again, this time using orange PLA, a color that does not appear in the dataset. Third, a 3DBenchy was printed in translucent natural PLA. Figure 6.9 shows the printed objects and their segmentation results. Although the 3DBenchy model is part of the dataset, this particular color is not. The translucent PLA gives the object a distinctly different appearance, with the previous layer visibly shining through the current layer due to its transparency. Since there are no translucent filaments in the training dataset, this test is particularly challenging. All three neural network architectures (U-Net, U-Net++, and DenseNet) were tested with all five input types (*all_led*, *four_directions*, *normalmap*, *heightmap*, and *rgb_led*). The results are listed in Table 6.4. The IoU scores are calculated by comparing the segmentation results with ground truth masks generated from the G-code. For each object and input type, the best IoU score is highlighted in bold.

The segmentation performance on unseen objects is slightly lower than on the validation dataset, yet remains good overall. In this test, the U-Net++ architecture again provides the best



(a) **(b)** **(c)**

Figure 6.9 – The tree objects used for the generalization and robustness tests: **(a)** low poly squirrel [W45] printed in yellow PLA, **(b)** the same squirrel model printed in orange PLA, and **(c)** a 3DBenchy [W31] printed in translucent natural PLA. The bottom row shows the segmentation results from the U-Net++ architecture with *rgb_led* input type.

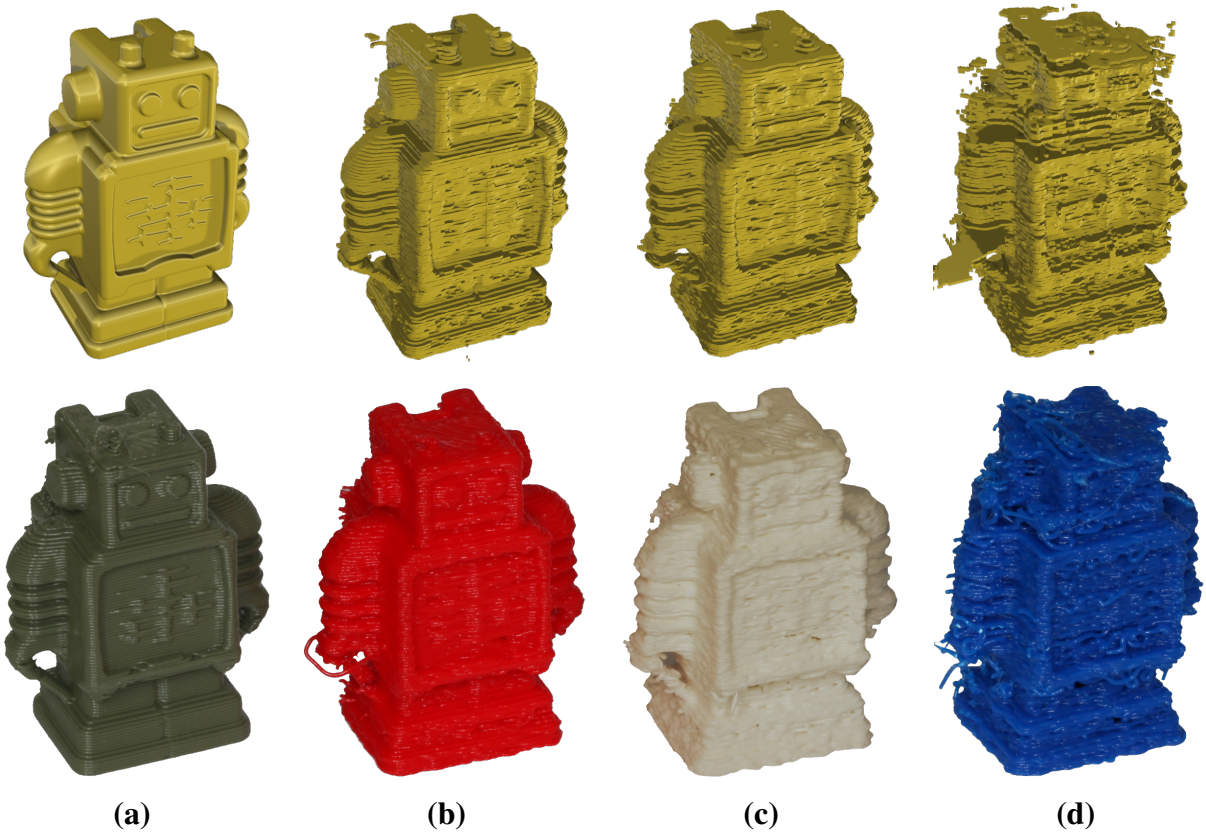
Table 6.4 – Test results for generalization and robustness evaluation: IoU scores for different neural network architectures and input types on unseen objects. The highest IoU score for each object and input is shown in bold.

Object	Architecture	*all_led*	*four_directions*	*normalmap*	*heightmap*	*rgb_led*
Squirrel [W45] yellow	U-Net	**0.817**	0.735	**0.760**	0.784	0.686
	U-Net++	0.816	**0.836**	0.692	**0.838**	**0.832**
	DenseNet	0.541	0.397	0.676	0.633	0.670
Squirrel [W45] orange	U-Net	0.793	0.736	0.535	0.651	0.532
	U-Net++	**0.816**	**0.821**	0.713	**0.839**	**0.831**
	DenseNet	0.738	0.750	**0.749**	0.756	0.732
3DBenchy [W31] trans. natural	U-Net	0.779	0.669	0.698	0.654	0.525
	U-Net++	**0.802**	**0.804**	**0.705**	**0.825**	**0.823**
	DenseNet	0.571	0.684	0.611	0.601	0.809

results, especially with the *heightmap* input type. The *rgb_led* input type also performs well. But from a practical standpoint, the most favorable choices are *rgb_led* or *all_led*, since the other inputs require capturing four additional images. So *rgb_led* delivers the best balance between segmentation performance and acquisition time and is therefore used for further evaluations.

The first layer of the clear Benchy is segmented noticeably worse, because the translucent filament lets the build plate color shine through, reducing contrast. While printing one squirrel ear, the nozzle partially clogged, recovered, and then clogged again after a few layers, leaving a visible gap and a missing tip (see Figure 6.9). This may be caused by frequent retractions and the small print area of the ears. The network correctly identifies and segments the regions with missing material, even though such defects were not present in the training data. Also, the seam line is clearly visible in the segmentation results of all objects. The seam line occurs in the training data, but it is not marked by the ground truth masks. These findings indicate that the network has learned to segment the actually printed material rather than just reproducing the G-code geometry. Segmenting regions that are not in the ground truth masks results in a lower IoU score, even though the segmentation is actually more accurate. The results also show small artifacts around the objects. Some of these artifacts are from stringing during printing, while others are due to minor misclassifications by the network.

Overall, the segmentation results on unseen objects are good and indicate that the trained neural networks can generalize well to new objects and colors. The segmentation performance could likely be improved with additional training data and a fine-tuned network architecture specifically designed for this task.



(a) **(b)** **(c)** **(d)**

Figure 6.10 – Results of the recursive segmentation test: **(a)** Original model printed in dark green, **(b)** first copy printed in red, **(c)** second copy printed in white, **(d)** third copy printed in blue. The top row shows the original model and the three reconstructions from the previous segmentation, while the bottom row shows the printed objects.

6.5.4 Recursive Segmentation

To evaluate how well the segmentation preserves small details and to demonstrate its robustness, a recursive segmentation test is performed. In this test, a small object with fine surface details is printed and segmented using the proposed method. The resulting segmentation is then used to reprint the object, which is segmented again. Analogous to repeatedly photocopying a document, the segmentation quality is expected to degrade with each iteration. The magnitude of this degradation indicates how well fine details are preserved and how robust the segmentation is to noise and artifacts. For this experiment, the process is repeated three times, resulting in four printed objects in total, with the filament color changed at each iteration. Each of the four colors is already included in the dataset to ensure that the neural network can handle them.

The four printed robots are shown in Figure 6.10, with the original model on the left and the three copies on the right. The overall shape of the object is preserved across all iterations. However, small, pointy details such as the antennas are nearly lost after the first copy. The robot's face is still barely visible in the second copy, while the gap between the legs appears only in the

first copy. The ribs on the arms are still visible in the third copy. The hook on the back of the head remains functional in the second copy but becomes clogged in the third copy. Over the iterations, the layer surfaces tend to develop small holes and uneven surfaces, which can lead to issues when printing the copies. These issues worsen with each iteration as the filament is extruded incorrectly and sometimes without proper support from the previous layer.

Segmentation results can likely be improved with additional training data and a fine-tuned network architecture specifically designed for this task. Furthermore, the post-processing of the segmentation masks could be improved, as they still contain small artifacts and holes, given that the printed structures are known to be at least 0.4 mm wide and cannot be smaller than the printed lines.

Although small details are lost over the iterations, the overall shape of the object is preserved. This is still a good result, given that the object is only 33.8 mm tall and has many small details. To our knowledge, no other approach achieves similar results with in situ inspection. Methods that perform similarly rely on post-production inspection with high-resolution 3D scanners or CT scans, which have other disadvantages such as high cost or no ability to inspect the inside of the object. With this approach, active repair during printing is also possible, similar to the approach in Chapter 5.

6.5.5 Accuracy Measurements

Because precise segmentation is essential for accurate object reconstruction, experiments were conducted to quantitatively assess segmentation accuracy. Accuracy was evaluated by printing objects with precisely known dimensions and by measuring both the physical parts and the corresponding segmentation masks. The measurements include inaccuracies due to camera positioning, camera optics, the stitching process, and the segmentation itself. The test objects consisted of a square (30 × 30 mm, 2 mm wall thickness) and a circle (20 mm diameter, 2 mm wall thickness), each printed on a solid base to ensure good adhesion and minimize deformation. The objects are shown in Figure A.4 in the appendix. Each object was printed in four colors (black, blue, red, and yellow) and in two orientations: as modeled and rotated by 45°.

For each printed object, measurements are taken at the center: from top to bottom and from left to right. This is done for both the inner and outer dimensions of the square and the circle. As a result, each object yields eight measurements, two for each dimension type (inner and outer) for both shapes (square and circle). With eight objects in total, this results in 64 measurements.

The measured offsets, expressed in µm, are presented in Figure 6.11. Each point corresponds to an individual measurement offset and is color-coded by the object's color. The × markers represent measurements taken from objects printed at a 45° orientation, while the + markers denote measurements from objects printed without rotation. The thick black line indicates the mean absolute offset for each measurement group. Across all measurements, the mean offset between the physical and the segmented measurements is 61.5 µm. Strangely, the segmentation offsets are

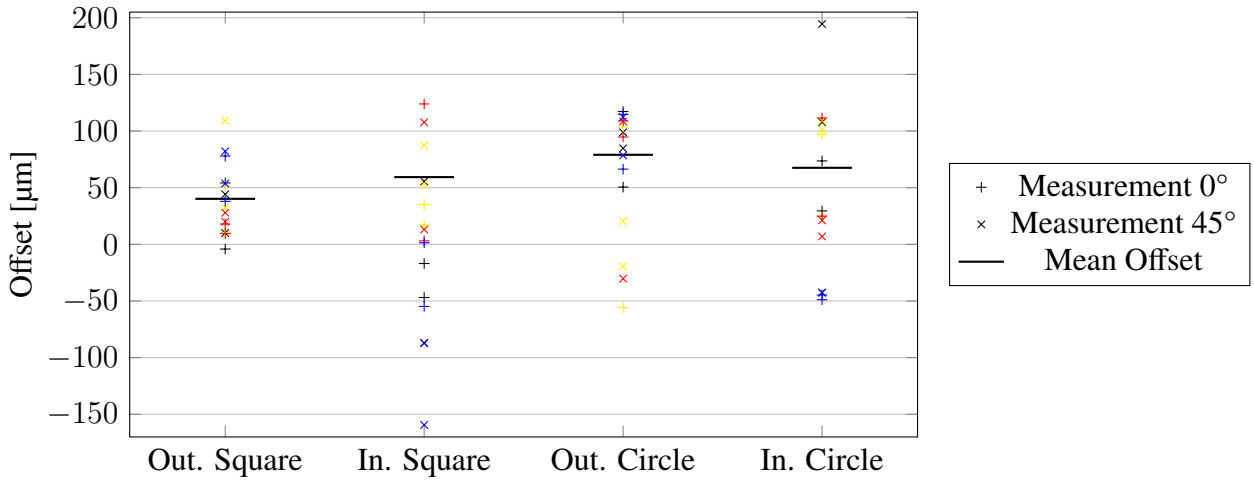


Figure 6.11 – Measurement offsets between the physical measurements of the printed objects and the corresponding measurements taken from the segmentation masks. Each point represents an individual offset, colored according to the object's color. × indicates measurements taken from the object printed at 45° orientation, while + indicates measurements taken from the object printed without rotation. The thick black line shows the mean absolute offset for each group.

mostly positive, meaning the segmented dimensions tend to be slightly larger than the physical part for both outer and inner measurements. This is unexpected, as for inner measurements the segmentation would normally be smaller than the physical measurements. Since the segmentation mask is stitched together from individual images, this positive offset could be caused by the stitching process. The positive offset may also be influenced by greater measurement errors on inner dimensions, which are more challenging to measure than outer dimensions. Additionally, printed lines often shrink inward during cooling, causing a mismatch between the actual printed object and the generated ground truth mask. This could lead the network to compensate for the shrinkage effect, resulting in a positive offset in the segmentation results. With more precisely annotated ground truth, this issue might be reduced.

Nevertheless, this offset is very small, as it reflects the combined effect of measurement errors, stitching errors, segmentation errors and the calipers uncertainty of 30 µm. This indicates that the segmentation is highly accurate and can be used to reconstruct the object with high precision.

6.5.6 Limitations

The main drawback of this approach is the significantly longer print time resulting from the image acquisition step. This overhead can be reduced by using cameras with a larger field of view or by decreasing the time needed to capture the images. However, even with such improvements, this approach will always take longer than a normal print without image acquisition.

For practical deployment in production systems, segmentation accuracy must be improved further. The limited dataset with only two different infill patterns, a fixed line width, and a constant layer height may restrict the networks' ability to generalize. Also all prints were made on the same printer, so the networks may learn printer-specific artifacts that do not generalize to other printers. The dataset also lacks failures and artifacts that may occur during printing, such as layer shifts, under-extrusion, or stringing. Including such data into the training dataset could improve robustness and help the network learn to segment actual printed layers more accurately.

Because the camera captures images strictly from above, the process has no direct information about the layer heights in the object, as there is currently no sensor data providing this information. Therefore, the process can only assume that the layer height specified in the G-code is correct. Additionally, the layer height must currently be fixed for all layers in the object and adaptive layer heights are not supported. Other issues that are not visible from above like layer warping, layer separation, vibration and ringing on the outside of the object, and poor bridging and overhanging layers are also not detected by this approach.

6.6 Conclusion

This chapter presented a comprehensive approach to layer segmentation and reconstruction of 3D-printed objects. It described the image acquisition process used to record each printed layer and compared several segmentation strategies, namely depth sensing, photometric stereo, and neural networks. Only the neural network-based approach produced sufficiently accurate results. To identify the most effective model, multiple architectures with different input types were evaluated. The best-performing configuration was U-Net++ using the *rgb_led* images. Tests on unseen objects show that the trained networks generalize well to new geometries and colors. The reconstruction is robust enough to preserve the overall shape of an object even after multiple iterations of printing and re-segmentation. Segmentation accuracy was quantified by comparing physical measurements of printed parts with corresponding measurements taken from the segmentation masks. The results indicate high accuracy, with a mean offset of 61.5 µm, which is sufficient for object reconstruction. However, the image acquisition time is currently too high for practical use, increasing print time by 95% to 407% depending on the object. While the accuracy is adequate for reconstruction, further improvement is needed for practical deployment to increase confidence in the segmentation results.

Overall, this chapter demonstrates that camera-based layer segmentation using neural networks enables reliable reconstruction and defect detection in 3D printed objects, laying the groundwork for future automated quality assurance.

6.6.1 Future Work

Future research should prioritize optimizing the image acquisition process to reduce the time needed for both image capture and processing. This could include using smaller, lower-cost cameras with a larger field of view, or alternative approaches that capture the entire layer in a single image. The system should also be adapted to work with more common printers, such as Prusa or BambuLab printers, which are widely used in the 3D printing community. Together, these changes would make the system more accessible and usable for a broader audience. To improve segmentation accuracy, future work should evaluate more modern neural network architectures, such as vision transformers, as well as other models specifically designed for image segmentation. In addition, fine-tuning the neural networks hyperparameters could lead to better results. To further improve robustness, the dataset should be expanded to include a broader variety of objects, colors, and infill patterns. This broader coverage would help the neural networks generalize more reliably to unseen objects and colors. Additionally, ground truth generation should move away from relying on G-code and use a more accurate method for producing labels.

At present, the workflow involves multiple manual steps and depends on more than one computer. Future work should focus on fully automating the process and consolidating all stages into a single system directly connected to the printer.

To enable closed-loop printing, the process should be extended to include real-time defect detection and automated repair. This requires feedback mechanisms that identify defects during printing and generate repair paths for regions with insufficient material, similar to the approach for printed electronics described in Chapter 5. With such feedback, the printer can adapt and correct errors as they occur, which would be a significant step toward fully autonomous 3D printing.

In the future, the approach should also be tested at larger scales and on different printers to confirm robustness and reliability across a variety of printing scenarios. Finally, it should be extended to address additional defect types, such as warping, layer separation, and ringing, which are common in 3D printing and are not covered by the current approach.

Chapter 7

Conclusion

This thesis presented a set of tools and methods that enable reliable 5-axis printing of geometrically complex objects with embedded electronics on low-cost platforms. It also introduced in situ monitoring for real-time defect detection and correction, and layer-wise reconstruction for quality assurance. In conclusion, the research questions can be answered as follows:

Path Planning for 5-Axis Printing Complex 3D geometries with embedded electronics can be reliably printed on low-cost 5-axis machines using a calibration-aware path planning framework that compensates for machine misalignments. Calibration measures A and B axis misalignments and intersection offsets, which are then modeled in a URDF-based kinematic model. This model is applied in inverse kinematics to correct every tool pose, turning previously colliding prints into smooth, aligned toolpaths. Surface-conformal toolpaths are generated from arbitrary shapes by sampling surface curves and computing per-point normals. Adjacent normals are intelligently combined and tilted at corners to keep deposition continuous while respecting nozzle tilt bounds. Experiments showed that this approach eliminates surface collisions and over-squishing, resulting in accurate and reliable prints with embedded electronics.

In Situ Error Correction of Printed Electronics To develop a reliable vision based wire detection and repair system, a 5-axis printing platform was equipped with a camera and lighting setup to capture images of printed traces during pauses in the print process. Different neural network architectures were evaluated for segmenting conductive traces from the images, with U-Net performing best. The developed image processing pipeline detects common defects such as connection breaks, shorts, and unreached points in the segmented images. These defects are repaired by extracting the missing segments from the original G-code and executing them as a repair routine during printing. Experiments demonstrated that this approach reduces failed prints by autonomously detecting and repairing defects during the printing process.

Layer Segmentation and Object Reconstruction To capture the true printed geometry of each layer and to distinguish it from previously printed layers, a layer-wise imaging and segmentation approach was developed. To improve segmentation quality, different illumination strategies were tested, showing that they improve segmentation. The best-performing segmentation model, U-Net++, was then fed with images of the current and previous layers to generate high-fidelity per-layer masks. The layer masks are then stacked to create a 3D digital twin of the printed object. These twins enable quality inspection and traceability of the printed objects. Generalization tests on unseen objects and colors, and a recursive copying experiment, showed robustness and the ability to adapt to new colors. The reconstruction accuracy is promising, with a mean offset of 61.5 µm.

The research questions have successfully been answered with the presented methods. The core contributions of this thesis are a calibration-aware path planning framework for 5-axis printing on low-cost hardware, a vision based error detection and repair system, and a layer-wise imaging and segmentation approach for digital twin generation. These contributions together push the boundaries of what is achievable with consumer-grade 3D printers, bringing them closer to industrial capabilities. Detecting errors in real time and correcting them autonomously increases the reliability and yield of prints, while layer-wise imaging and segmentation enables quality inspection and traceability for safety-critical applications.

7.1 Future Work

Although this thesis has made significant progress in 5-axis printing with embedded electronics, several questions for future research remain open. Based on the six main objectives of this thesis, the following directions for future work are suggested:

Calibration The resulting prints, even after calibration and compensation, still show some errors from an slightly incorrect nozzle height. Future research should focus on further investigating other error sources that are not yet modeled, such as thermal drift, backlash, and non-linearities. Furthermore, the probing process needs to be more stable, especially when probing 3D-printed rotation axes. Lastly, investigating how a closed loop control for the A/B axes can be integrated into the system would also be a valuable direction for future research.

Path Planning The presented path planning framework only supports line-style toolpaths for 5-axis printing. Future work should extend the framework to also support volumetric printing. In addition, the 5-axis capabilities should extended further to print complex models, for example through multi-directional printing or related strategies. Future research could also investigate how to further improve the surface speed continuity by continuously tilting the nozzle, rather than only at corners.

Wire Detection As image acquisition and segmentation are time-consuming, future work should focus on speeding up the entire process. Segmentation performance could also be improved with more training data and better segmentation networks. Finally, fine-tuning the network architecture and hyperparameters could further improve performance.

Error Detection and Correction Currently, the error detection and correction system supports only wires aligned with the three main axes. Future work should implement a coordinate mapping to also support arbitrary wire orientations. Additionally, a method for inspecting inter-layer connections should be developed. The repair strategies could also be improved and extended to handle defects that require material removal, such as short circuits or over-extrusion.

Layer Segmentation As image acquisition consumes most of the runtime, future work should focus on accelerating this step. Segmentation performance could be improved with more training data and better segmentation networks. Revising the ground-truth generation to no longer rely on G-code would also help, since the current G-code-based ground truth is inaccurate. Finally, extending this method to 5-axis prints would be an interesting direction for future research.

Object Reconstruction Because the reconstruction process still involves multiple manual steps, future work should automate the entire pipeline and integrate it into a single system directly connected to the printer. Currently, reconstruction of the individual layers relies on the slicer's nominal layer height rather than the actually measured layer height. Future research should measure the true layer height and use it for reconstruction. Finally, using the digital twin for quality inspection and for reprinting faulty parts would be a valuable direction for future work.

References

[1] Hideo Kodama. "Automatic method for fabricating a three-dimensional plastic model with photo-hardening polymer". In: *Review of Scientific Instruments* 52.11 (Nov. 1981), pp. 1770–1773. ISSN: 1089-7623. DOI: 10.1063/1.1136492.

[2] S Scott Crump. *US5121329A - Apparatus and method for creating three-dimensional objects*. 1992. URL: https://patents.google.com/patent/US5121329A/en.

[3] Rhys Jones, Patrick Haufe, Edward Sells, Pejman Iravani, Vik Olliver, Chris Palmer, and Adrian Bowyer. "RepRap – the replicating rapid prototyper". In: *Robotica* 29.1 (Jan. 2011), pp. 177–191. ISSN: 1469-8668. DOI: 10.1017/s026357471000069x.

[4] Tao Peng, Karel Kellens, Renzhong Tang, Chao Chen, and Gang Chen. "Sustainability of additive manufacturing: An overview on its energy demand and environmental impact". In: *Additive Manufacturing* 21 (May 2018), pp. 694–704. ISSN: 2214-8604. DOI: 10.1016/j.addma.2018.04.022.

[5] Terry Wohlers, Robert Ian Campbell, Olaf Diegel, Ray Huff, and Joseph Kowen. *Wohlers Report 2024: 3D printing and additive manufacturing global state of the industry*. Wohlers Associates, 2024.

[6] Ian Gibson, David Rosen, and Brent Stucker. *Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing*. Springer New York, 2015. ISBN: 9781493921133. DOI: 10.1007/978-1-4939-2113-3.

[7] Martin Leary. "AM production economics". In: *Design for Additive Manufacturing*. Elsevier, 2020, pp. 7–31. ISBN: 9780128167212. DOI: 10.1016/b978-0-12-816721-2.00002-6.

[8] Mahdi Jamshid, Ray Huff, Joseph Kowen, Ismail Fidan, and Eujin Pei. *Wohlers Report 2025: 3D printing and additive manufacturing global state of the industry*. Wohlers Associates, 2025. URL: https://wohlersassociates.com/product/wr2025/.

[9] Edward Sells and Adrian Bowyer. *Rapid Prototyped Electronic Circuits*. Tech. rep. University of Bath Department of Mechanical Engineering, 2004. URL: https://fennetic.net/irc/reprap_circuits.pdf.

[10] C. Hanumanth Rao, Kothuru Avinash, B. K. S. V. L. Varaprasad, and Sanket Goel. "A Review on Printed Electronics with Digital 3D Printing: Fabrication Techniques, Materials, Challenges and Future Opportunities". In: *Journal of Electronic Materials* 51.6 (Apr. 2022), pp. 2747–2765. DOI: 10.1007/s11664-022-09579-7.

[11] Sarat Singamneni, Asimava Roychoudhury, Olaf Diegel, and Bin Huang. "Modeling and evaluation of curved layer fused deposition". In: *Journal of Materials Processing Technology* 212.1 (Jan. 2012), pp. 27–35. ISSN: 0924-0136. DOI: 10.1016/j.jmatprotec.2011.08.001.

[12] Bin Huang and Sarat B Singamneni. "Curved Layer Adaptive Slicing (CLAS) for fused deposition modelling". In: *Rapid Prototyping Journal* 21.4 (June 2015), pp. 354–367. ISSN: 1355-2546. DOI: 10.1108/rpj-06-2013-0059.

[13] Sebastian Atarihuana, Felipe Fernández, José Erazo, Mateo Narváez, and Víctor Hidalgo. "Optimal Strategies for Filament Orientation in Non-Planar 3D Printing". In: *Processes* 12.12 (Dec. 2024), p. 2811. ISSN: 2227-9717. DOI: 10.3390/pr12122811.

[14] Gang Zhao, Guocai Ma, Jiangwei Feng, and Wenlei Xiao. "Nonplanar slicing and path generation methods for robotic additive manufacturing". In: *The International Journal of Advanced Manufacturing Technology* 96.9–12 (Mar. 2018), pp. 3149–3159. ISSN: 1433-3015. DOI: 10.1007/s00170-018-1772-9.

[15] Michael Rieger, Benjamin Johnen, and Bernd Kuhlenkoetter. "Analysis and development of the fused layer manufacturing process using industrial robots". In: *Proceedings of ISR 2016: 47st International Symposium on Robotics*. VDE. 2016, pp. 1–8.

[16] Weiming Wang, Cédric. Zanni, and Leif Kobbelt. "Improved Surface Quality in 3D Printing by Optimizing the Printing Direction". In: *Computer Graphics Forum* 35.2 (May 2016), pp. 59–70. ISSN: 1467-8659. DOI: 10.1111/cgf.12811.

[17] Eric Macdonald, Rudy Salas, David Espalin, Mireya Perez, Efrain Aguilera, Dan Muse, and Ryan B. Wicker. "3D Printing for the Rapid Prototyping of Structural Electronics". In: *IEEE Access* 2 (Dec. 2014), pp. 234–242. ISSN: 2169-3536. DOI: 10.1109/access.2014.2311810.

[18] Daniel Ahlers. “In-Situ Verification of 3D-Printed Electronics Using Deep Convolutional Neural Networks”. In: *Proceedings of the 32th Annual International Solid Freeform Fabrication Symposium* (2021). DOI: 10.26153/TSW/17557.

[19] Daniel Ahlers, Florens Wasserfall, Johannes Hörber, and Jianwei Zhang. “Automatic in-situ error correction for 3D printed electronics”. In: *Additive Manufacturing Letters* 7 (Dec. 2023), p. 100164. ISSN: 2772-3690. DOI: 10.1016/j.addlet.2023.100164.

[20] Daniel Ahlers, Tom Schmolzi, German Junca, Jianwei Zhang, and Florens Wasserfall. “Calibration and compensation of 5-axis 3D-printers for printed electronics”. In: *Additive Manufacturing Letters* 12 (Feb. 2025), p. 100265. ISSN: 2772-3690. DOI: 10.1016/j.addlet.2024.100265.

[21] Daniel Ahlers, Niklas Fiedler, Florens Wasserfall, and Jianwei Zhang. “Camera Based In Situ Layer Segmentation and Object Reconstruction for Digital Twins in FFF 3D Printing”. In: *Submitted to Journal of Intelligent Manufacturing* (2025).

[22] Florens Wasserfall, Daniel Ahlers, Norman Hendrich, and Jianwei Zhang. “3D-printable electronics-integration of SMD placement and wiring into the slicing process for FDM fabrication”. In: *Proceedings of the 27th International Solid Freeform Fabrication Symposium*. University of Texas at Austin, 2016, pp. 1826–1837. URL: https://hdl.handle.net/2152/89717.

[23] Daniel Ahlers, Florens Wasserfall, Norman Hendrich, and Jianwei Zhang. “3D Printing of Nonplanar Layers for Smooth Surface Generation”. In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, Aug. 2019, pp. 1737–1743. DOI: 10.1109/coase.2019.8843116.

[24] Florens Wasserfall, Daniel Ahlers, and Norman Hendrich. “Optical In-Situ Verification of 3D-Printed Electronic Circuits”. In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, Aug. 2019, pp. 1302–1307. DOI: 10.1109/coase.2019.8842835.

[25] Florens Wasserfall, Norman Hendrich, Daniel Ahlers, and Jianwei Zhang. “Topology-aware routing of 3D-printed circuits”. In: *Additive Manufacturing* 36 (Dec. 2020), p. 101523. ISSN: 2214-8604. DOI: 10.1016/j.addma.2020.101523.

[26] Daniel Ahlers, Florens Wasserfall, Norman Hendrich, Arne Niklas Bungener, Jan-Tarek Butt, and Jianwei Zhang. “Automated In Situ Placing of Metal Components Into 3-D

Printed FFF Objects". In: *IEEE/ASME Transactions on Mechatronics* 26.4 (Aug. 2021), pp. 1886–1894. ISSN: 1941-014X. DOI: 10.1109/tmech.2021.3078409.

[27] Charles Hull. "StereoLithography: Plastic prototypes from CAD data without tooling". In: *Modern Casting* 78.8 (1988), p. 38.

[28] ISO - International Organization for Standardization. *ISO/ASTM 52900:2021 Additive manufacturing - General principles - Fundamentals and vocabulary*. 2021. URL: https://www.iso.org/obp/ui/#iso:std:iso-astm:52900:ed-2:v1:en.

[29] Marc-Antoine de Pastre, Yann Quinsat, and Claire Lartigue. "Effects of additive manufacturing processes on part defects and properties: a classification review". In: *International Journal on Interactive Design and Manufacturing (IJIDeM)* 16.4 (Feb. 2022), pp. 1471–1496. ISSN: 1955-2505. DOI: 10.1007/s12008-022-00839-8.

[30] Phil Reeves, Chris Tuck, and Richard Hague. "Additive Manufacturing for Mass Customization". In: *Mass Customization*. Springer London, 2011, pp. 275–289. ISBN: 9781849964890. DOI: 10.1007/978-1-84996-489-0_13.

[31] Cassie Gutierrez, Rudy Salas, Gustavo Hernandez, Dan Muse, Richard Olivas, Eric MacDonald, Michael D. Irwin, Ryan Wicker, Mike Newton, Ken Church, and Brian Zufelt. "CubeSat Fabrication through Additive Manufacturing and Micro-Dispensing". In: *International Symposium on Microelectronics* 2011.1 (Jan. 2011), pp. 1021–1027. DOI: 10.4071/isom-2011-tha4-paper3.

[32] Markus Ankenbrand, Yannic Eiche, and Jörg Franke. "Programming and Evaluation of a Multi-Axis/Multi-Process Manufacturing System for Mechatronic Integrated Devices". In: *2019 International Conference on Electronics Packaging (ICEP)*. IEEE, Apr. 2019, pp. 273–278. DOI: 10.23919/icep.2019.8733548.

[33] Corey Shemelya, Luis Banuelos-Chacon, Adrian Melendez, Craig Kief, David Espalin, Ryan Wicker, Gijs Krijnen, and Eric MacDonald. "Multi-functional 3D printed and embedded sensors for satellite qualification structures". In: *2015 IEEE SENSORS*. IEEE, Nov. 2015, pp. 1–4. DOI: 10.1109/icsens.2015.7370541.

[34] Erick DeNava, Misael Navarrete, Amit Lopes, Mohammed Alawneh, Marlene Contreras, Dan Muse, Silvia Castillo, and Eric MacDonald. "Three-Dimensional Off-Axis Component Placement and Routing for Electronics Integration using Solid Freeform Fabrication". In: *Proceedings of the 19th International Solid Freeform Fabrication-Symposium*. 2008. DOI: 10.26153/TSW/14967.

[35] Mohammad Shojib Hossain, Jose A. Gonzalez, Ricardo Martinez Hernandez, Mohammad Arif Ishtiaque Shuvo, Jorge Mireles, Ahsan Choudhuri, Yirong Lin, and Ryan B. Wicker. "Fabrication of smart parts using powder bed fusion additive manufacturing technology". In: *Additive Manufacturing* 10 (Apr. 2016), pp. 58–66. ISSN: 2214-8604. DOI: 10.1016/j.addma.2016.01.001.

[36] Johannes Heinrich Glasschröder. "Additiv gefertigte Werkstücke mit integrierten elektrischen Schaltungen unter Nutzung des 3D-Druckprozesses". PhD thesis. Technische Universität München, 2018. URL: https://mediatum.ub.tum.de/doc/1421825/1421825.pdf.

[37] Gerd Grau and Vivek Subramanian. "Dimensional scaling of high-speed printed organic transistors enabling high-frequency operation". In: *Flexible and Printed Electronics* 5.1 (Mar. 2020), p. 014013. ISSN: 2058-8585. DOI: 10.1088/2058-8585/ab739a.

[38] Keun-Ho Choi, JongTae Yoo, Chang Kee Lee, and Sang-Young Lee. "All-inkjet-printed, solid-state flexible supercapacitors on paper". In: *Energy and Environmental Science* 9.9 (2016), pp. 2812–2821. ISSN: 1754-5706. DOI: 10.1039/c6ee00966b.

[39] Vivek Subramanian, Jean M. J. Frechet, P.C. Chang, Daniel C. Huang, J.B. Lee, Steven Molesa, A.R. Murphy, David H. Redinger, and Steve Volkman. "Progress Toward Development of All-Printed RFID Tags: Materials, Processes, and Devices". In: *Proceedings of the IEEE* 93.7 (July 2005), pp. 1330–1338. ISSN: 0018-9219. DOI: 10.1109/jproc.2005.850305.

[40] Kyuyoung Kim, Jaeho Park, Ji-hoon Suh, Minseong Kim, Yongrok Jeong, and Inkyu Park. "3D printing of multiaxial force sensors using carbon nanotube (CNT)/thermoplastic polyurethane (TPU) filaments". In: *Sensors and Actuators A: Physical* 263 (Aug. 2017), pp. 493–500. ISSN: 0924-4247. DOI: 10.1016/j.sna.2017.07.020.

[41] Ruben Goos, Akash Verma, and Eleonora Ferraris. "Development of a free-form piezoresistive pressure sensor using advanced printing methods". In: *Procedia CIRP* 113 (2022), pp. 335–340. ISSN: 2212-8271. DOI: 10.1016/j.procir.2022.09.139.

[42] Omar Faruk Emon, Faez Alkadi, Mazen Kiki, and Jae-Won Choi. "Conformal 3D printing of a polymeric tactile sensor". In: *Additive Manufacturing Letters* 2 (Apr. 2022), p. 100027. ISSN: 2772-3690. DOI: 10.1016/j.addlet.2022.100027.

[43] Ping Yang, Tianqi Zhai, Boyang Yu, Gengxin Du, Baoxiu Mi, Xinyan Zhao, and Weiwei Deng. "Toward all aerosol printing of high-efficiency organic solar cells using environ-

mentally friendly solvents in ambient air". In: *Journal of Materials Chemistry A* 9.32 (2021), pp. 17198–17210. ISSN: 2050-7496. DOI: 10.1039/d1ta02890a.

[44] Ke Sun, Teng-Sing Wei, Bok Yeop Ahn, Jung Yoon Seo, Shen J. Dillon, and Jennifer A. Lewis. "3D Printing of Interdigitated Li-Ion Microbattery Architectures". In: *Advanced Materials* 25.33 (June 2013), pp. 4539–4543. ISSN: 1521-4095. DOI: 10.1002/adma.201301036.

[45] Ana C. Martinez, Alexis Maurel, Bharat Yelamanchi, A. Alec Talin, Sylvie Grugeon, Stéphane Panier, Loic Dupont, Ana Aranzola, Eva Schiaffino, Sreeprasad T. Sreenivasan, Pedro Cortes, and Eric MacDonald. "Combining 3D printing of copper current collectors and electrophoretic deposition of electrode materials for structural lithium-ion batteries". In: *Advances in Manufacturing* (July 2024). ISSN: 2195-3597. DOI: 10.1007/s40436-024-00514-z.

[46] Eva S. Rosker, Rajinder Sandhu, Jimmy Hester, Mark S. Goorsky, and Jesse Tice. "Printable Materials for the Realization of High Performance RF Components: Challenges and Opportunities". In: *International Journal of Antennas and Propagation* 2018 (2018), pp. 1–19. ISSN: 1687-5877. DOI: 10.1155/2018/9359528.

[47] Florens Wasserfall. "Embedding of SMD populated circuits into FDM printed objects". In: *Proceedings of the 26th International Solid Freeform Fabrication Symposium* (2015). URL: https://hdl.handle.net/2152/89317.

[48] Markellos Ntagios, Habib Nassar, and Ravinder Dahiya. "Closed-loop direct ink extruder system with multi-part materials mixing". In: *Additive Manufacturing* 64 (Feb. 2023), p. 103437. ISSN: 2214-8604. DOI: 10.1016/j.addma.2023.103437.

[49] Amit Joe Lopes, Eric MacDonald, and Ryan B. Wicker. "Integrating stereolithography and direct print technologies for 3D structural electronics fabrication". In: *Rapid Prototyping Journal* 18.2 (Mar. 2012), pp. 129–143. ISSN: 1355-2546. DOI: 10.1108/13552541211212113.

[50] P. J. Smith, D.-Y. Shin, J. E. Stringer, B. Derby, and N. Reis. "Direct ink-jet printing and low temperature conversion of conductive silver patterns". In: *Journal of Materials Science* 41.13 (July 2006), pp. 4153–4158. ISSN: 1573-4803. DOI: 10.1007/s10853-006-6653-1.

[51] Martin Hedges and Aaron Borras Marin. "3D aerosol jet printing-adding electronics functionality to RP/RM". In: *DDMC 2012 conference*. 2012, pp. 1–5. URL: https://

optomec.com/wp-content/uploads/2014/04/Optomec_NEOTECH_DDMC_3D_Aerosol_Jet_Printing.pdf.

[52] Johannes Hoerber, Johannes Glasschroeder, Michael Pfeffer, Johannes Schilp, Michael F. Zaeh, and Jörg Franke. "Approaches for Additive Manufacturing of 3D Electronic Applications". In: *Procedia CIRP* 17 (2014), pp. 806–811. ISSN: 2212-8271. DOI: 10.1016/j.procir.2014.01.090.

[53] Jacob Bayless, Mo Chen, and Bing Dai. *Wire embedding 3D printer*. Tech. rep. Engineering Physics Department, University of British Columbia, 2010. URL: https://www.reprap.org/mediawiki/images/archive/2/25/20100412225510!SpoolHead_FinalReport.pdf.

[54] David Espalin, Danny W. Muse, Eric MacDonald, and Ryan B. Wicker. "3D Printing multifunctionality: structures with electronics". In: *The International Journal of Advanced Manufacturing Technology* 72.5–8 (Mar. 2014), pp. 963–978. ISSN: 1433-3015. DOI: 10.1007/s00170-014-5717-7.

[55] Fabian Ziervogel, Lukas Boxberger, Andre Bucht, and Welf-Guntram Drossel. "Expansion of the Fused Filament Fabrication (FFF) Process Through Wire Embedding, Automated Cutting, and Electrical Contacting". In: *IEEE Access* 9 (2021), pp. 43036–43049. ISSN: 2169-3536. DOI: 10.1109/access.2021.3065873.

[56] Valentin Wilhelm Mauersberger, Fabian Ziervogel, Linda Weisheit, Lukas Boxberger, and Welf-Guntram Drossel. "Development of a Stable Process for Wire Embedding in Fused Filament Fabrication Printing Using a Geometric Correction Model". In: *Materials* 18.1 (Dec. 2024), p. 41. ISSN: 1996-1944. DOI: 10.3390/ma18010041.

[57] Simon J. Leigh, Robert J. Bradley, Christopher P. Purssell, Duncan R. Billson, and David A. Hutchins. "A Simple, Low-Cost Conductive Composite Material for 3D Printing of Electronic Sensors". In: *PLoS ONE* 7.11 (Nov. 2012). Ed. by Jeongmin Hong, e49365. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0049365.

[58] Kimball Andersen, Yue Dong, and Woo Soo Kim. "Highly Conductive Three-Dimensional Printing With Low-Melting Metal Alloy Filament". In: *Advanced Engineering Materials* 19.11 (June 2017). ISSN: 1527-2648. DOI: 10.1002/adem.201700301.

[59] Neeraj Panhalkar, Ratnadeep Paul, and Sam Anand. "A Novel Additive Manufacturing File Format for Printed Electronics". In: *Volume 2A: Advanced Manufacturing*.

IMECE2013. American Society of Mechanical Engineers, Nov. 2013. DOI: 10.1115/imece2013-64678.

[60] Thomas Krebs and Blaženko Šegmanović. "Integrating the CAD Worlds of Mechanics and Electronics with NEXTRA". In: *Concurrent Engineering Approaches for Sustainable Product Development in a Multi-Disciplinary Environment*. Springer London, Aug. 2012, pp. 777–788. ISBN: 9781447144267. DOI: 10.1007/978-1-4471-4426-7_66.

[61] John P. Swensen, Lael U. Odhner, Brandon Araki, and Aaron M. Dollar. "Injected 3D electrical traces in additive manufactured parts with low melting temperature metals". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2015, pp. 988–995. DOI: 10.1109/icra.2015.7139297.

[62] Callum Bailey, Efrain Aguilera, David Espalin, Jose Motta, Alfonso Fernandez, Mireya A. Perez, Christopher Dibiasio, Dariusz Pryputniewicz, Eric Macdonald, and Ryan B. Wicker. "Augmenting Computer-Aided Design Software With Multi-Functional Capabilities to Automate Multi-Process Additive Manufacturing". In: *IEEE Access* 6 (2018), pp. 1985–1994. ISSN: 2169-3536. DOI: 10.1109/access.2017.2781249.

[63] Hanno Platz, Michael Matthes, Markus Biener, Daniel Ernst, Michael Schleicher, and Wolfgang Kühn. *Klassifizierung der additiven Fertigungs- und 3D-Druckprozesse für die Elektronik*. Tech. rep. FED-Arbeitskreis 3D-Elektronik, June 2023. URL: https://www.fed.de/3d-elektronik/.

[64] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[65] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Springer International Publishing, 2015, pp. 234–241. ISBN: 9783319245744. DOI: 10.1007/978-3-319-24574-4_28.

[66] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer International Publishing, 2018, pp. 3–11. ISBN: 9783030008895. DOI: 10.1007/978-3-030-00889-5_1.

[67] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmen-

tation". In: *Computer Vision – ECCV 2018*. Springer International Publishing, 2018, pp. 833–851. ISBN: 9783030012342. DOI: 10.1007/978-3-030-01234-2_49.

[68] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016. DOI: 10.48550/ARXIV.1512.03385.

[69] Francois Chollet. "Xception: Deep Learning With Depthwise Separable Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017. DOI: 10.48550/arXiv.1610.02357.

[70] Simon Jegou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. "The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, July 2017, pp. 1175–1183. DOI: 10.1109/cvprw.2017.156.

[71] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv, 2017. DOI: 10.48550/ARXIV.1608.06993.

[72] A. Dolenc and I. Mäkelä. "Slicing procedures for layered manufacturing techniques". In: *Computer-Aided Design* 26.2 (Feb. 1994), pp. 119–126. ISSN: 0010-4485. DOI: 10.1016/0010-4485(94)90032-9.

[73] Florens Wasserfall, Norman Hendrich, and Jianwei Zhang. "Adaptive slicing for the FDM process revisited". In: *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*. IEEE, Aug. 2017, pp. 49–54. DOI: 10.1109/coase.2017.8256074.

[74] Tim Kuipers, Eugeni L. Doubrovski, Jun Wu, and Charlie C.L. Wang. "A Framework for Adaptive Width Control of Dense Contour-Parallel Toolpaths in Fused Deposition Modeling". In: *Computer-Aided Design* 128 (Nov. 2020), p. 102907. ISSN: 0010-4485. DOI: 10.1016/j.cad.2020.102907.

[75] Bin Huang and Sarat Singamneni. "A mixed-layer approach combining both flat and curved layer slicing for fused deposition modelling". In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 229.12 (Oct. 2014), pp. 2238–2249. ISSN: 2041-2975. DOI: 10.1177/0954405414551076.

[76] Yujie Shan, Yiyang Shui, Junyu Hua, and Huachao Mao. "Additive manufacturing of non-planar layers using isothermal surface slicing". In: *Journal of Manufacturing Processes* 86 (Jan. 2023), pp. 326–335. ISSN: 1526-6125. DOI: 10.1016/j.jmapro.2022.12.054.

[77] Emilio Ottonello, Pierre-Alexandre Hugron, Alberto Parmiggiani, and Sylvain Lefebvre. *QuickCurve: revisiting slightly non-planar 3D printing*. Tech. rep. 2024. DOI: 10.48550/ARXIV.2406.03966.

[78] Yuan Jin, Jianke Du, Yong He, and Guoqiang Fu. "Modeling and process planning for curved layer fused deposition". In: *The International Journal of Advanced Manufacturing Technology* 91.1–4 (Nov. 2016), pp. 273–285. ISSN: 1433-3015. DOI: 10.1007/s00170-016-9743-5.

[79] Chengkai Dai, Charlie C. L. Wang, Chenming Wu, Sylvain Lefebvre, Guoxin Fang, and Yong-Jin Liu. "Support-free volume printing by multi-axis motion". In: *ACM Transactions on Graphics* 37.4 (July 2018), pp. 1–14. ISSN: 1557-7368. DOI: 10.1145/3197517.3201342.

[80] Mohammed A. Isa and Ismail Lazoglu. "Five-axis additive manufacturing of freeform models through buildup of transition layers". In: *Journal of Manufacturing Systems* 50 (Jan. 2019), pp. 69–80. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2018.12.002.

[81] Paul Bartel. "Conformal Surface Printing on a 5-Axis 3D Printing System". Bachelor Thesis. University of Hamburg, Aug. 2024. URL: https://tams.informatik.uni-hamburg.de/publications/2024/BSc_Paul_Bartel.pdf.

[82] Ismail Enes Yigit and I. Lazoglu. "Spherical slicing method and its application on robotic additive manufacturing". In: *Progress in Additive Manufacturing* 5.4 (May 2020), pp. 387–394. ISSN: 2363-9520. DOI: 10.1007/s40964-020-00135-5.

[83] Yisong Gao, Lifang Wu, Dong-Ming Yan, and Liangliang Nan. "Near support-free multi-directional 3D printing via global-optimal decomposition". In: *Graphical Models* 104 (July 2019), p. 101034. ISSN: 1524-0703. DOI: 10.1016/j.gmod.2019.101034.

[84] Mingqian Wang, Haiguang Zhang, Qingxi Hu, Di Liu, and Herfried Lammer. "Research and implementation of a non-supporting 3D printing method based on 5-axis dynamic slice algorithm". In: *Robotics and Computer-Integrated Manufacturing* 57 (June 2019), pp. 496–505. ISSN: 0736-5845. DOI: 10.1016/j.rcim.2019.01.007.

[85] Prahar M. Bhatt, Rishi K. Malhan, Pradeep Rajendran, and Satyandra K. Gupta. "Building free-form thin shell parts using supportless extrusion-based additive manufacturing". In: *Additive Manufacturing* 32 (Mar. 2020), p. 101003. ISSN: 2214-8604. DOI: 10.1016/j.addma.2019.101003.

[86] Bruna Ramos, Diana Pinho, Daniela A.L. Martins, A. Ismael F. Vaz, and Luis Nunes Vicente. "Optimal 3D printing of complex objects in a 5–axis printer". In: *Optimization and Engineering* 23.2 (Apr. 2021), pp. 1085–1116. ISSN: 1573-2924. DOI: 10.1007/s11081-021-09624-0.

[87] Danjie Bi, Molong Duan, Tak Yu Lau, Fubao Xie, and Kai Tang. "Strength-enhanced volume decomposition for multi-directional additive manufacturing". In: *Additive Manufacturing* 69 (May 2023), p. 103529. ISSN: 2214-8604. DOI: 10.1016/j.addma.2023.103529.

[88] Guoxin Fang, Tianyu Zhang, Sikai Zhong, Xiangjia Chen, Zichun Zhong, and Charlie C. L. Wang. "Reinforced FDM: multi-axis filament alignment with controlled anisotropic strength". In: *ACM Transactions on Graphics* 39.6 (Nov. 2020), pp. 1–15. ISSN: 1557-7368. DOI: 10.1145/3414685.3417834.

[89] Yamin Li, Kai Tang, Dong He, and Xiangyu Wang. "Multi-Axis Support-Free Printing of Freeform Parts with Lattice Infill Structures". In: *Computer-Aided Design* 133 (Apr. 2021), p. 102986. ISSN: 0010-4485. DOI: 10.1016/j.cad.2020.102986.

[90] Tianyu Zhang, Guoxin Fang, Yuming Huang, Neelotpal Dutta, Sylvain Lefebvre, Zekai Murat Kilic, and Charlie C. L. Wang. "S 3 -Slicer: A General Slicing Framework for Multi-Axis 3D Printing". In: *ACM Transactions on Graphics* 41.6 (Nov. 2022), pp. 1–15. ISSN: 1557-7368. DOI: 10.1145/3550454.3555516.

[91] Yuan Yao, Yichi Zhang, Mohamed Aburaia, and Maximilian Lackner. "3D Printing of Objects with Continuous Spatial Paths by a Multi-Axis Robotic FFF Platform". In: *Applied Sciences* 11.11 (May 2021), p. 4825. ISSN: 2076-3417. DOI: 10.3390/app11114825.

[92] Guoquan Zhang, Yaohui Wang, Ziwen Chen, Xuguang Xu, Ke Dong, and Yi Xiong. "Robot-assisted conformal additive manufacturing for continuous fibre-reinforced grid-stiffened shell structures". In: *Virtual and Physical Prototyping* 18.1 (May 2023). ISSN: 1745-2767. DOI: 10.1080/17452759.2023.2203695.

[93] Guoxin Fang, Tianyu Zhang, Yuming Huang, Zhizhou Zhang, Kunal Masania, and Charlie C.L. Wang. "Exceptional mechanical performance by spatial printing with continuous fiber: Curved slicing, toolpath generation and physical verification". In: *Additive Manufacturing* 82 (Feb. 2024), p. 104048. ISSN: 2214-8604. DOI: 10.1016/j.addma.2024.104048.

[94] Adrián López-Arrabal, Álvaro Guzmán-Bautista, William Solórzano-Requejo, Francisco Franco-Martínez, and Mónica Villaverde. "Axisymmetric non-planar slicing and path planning strategy for robot-based additive manufacturing". In: *Materials and Design* 241 (May 2024), p. 112915. ISSN: 0264-1275. DOI: 10.1016/j.matdes.2024.112915.

[95] W.T. Lei and Y.Y. Hsu. "Accuracy test of five-axis CNC machine tool with 3D probe–ball. Part I: design and modeling". In: *International Journal of Machine Tools and Manufacture* 42.10 (Aug. 2002), pp. 1153–1162. ISSN: 0890-6955. DOI: 10.1016/s0890-6955(02)00047-0.

[96] W.T Lei and Y.Y Hsu. "Accuracy enhancement of five-axis CNC machines through real-time error compensation". In: *International Journal of Machine Tools and Manufacture* 43.9 (July 2003), pp. 871–877. ISSN: 0890-6955. DOI: 10.1016/s0890-6955(03)00089-0.

[97] E.L.J. Bohez. "Compensating for systematic errors in 5-axis NC machining". In: *Computer-Aided Design* 34.5 (Apr. 2002), pp. 391–403. ISSN: 0010-4485. DOI: 10.1016/s0010-4485(01)00111-7.

[98] Sascha Weikert. "R-Test, a New Device for Accuracy Measurements on Five Axis Machine Tools". In: *CIRP Annals* 53.1 (2004), pp. 429–432. ISSN: 0007-8506. DOI: 10.1016/s0007-8506(07)60732-x.

[99] Guoqiang Fu, Jianzhong Fu, Hongli Gao, and Xinhua Yao. "Squareness error modeling for multi-axis machine tools via synthesizing the motion of the axes". In: *The International Journal of Advanced Manufacturing Technology* 89.9–12 (Aug. 2016), pp. 2993–3008. ISSN: 1433-3015. DOI: 10.1007/s00170-016-9259-z.

[100] Ming Deng, Huimin Li, Sitong Xiang, Puling Liu, Xiaobing Feng, Zhengchun Du, and Jianguo Yang. "Geometric errors identification considering rigid-body motion constraint for rotary axis of multi-axis machine tool using a tracking interferometer". In: *International Journal of Machine Tools and Manufacture* 158 (Nov. 2020), p. 103625. ISSN: 0890-6955. DOI: 10.1016/j.ijmachtools.2020.103625.

[101] Guoqiang Fu, Kunlong Lin, Yue Zheng, Sipei Zhu, Caijiang Lu, and Xi Wang. "PIGEs Identification of Rotary Axes Based on PIGEs Influences on Their Owner Rotary Axes for the Five-Axis Machine Tool". In: *IEEE Transactions on Instrumentation and Measurement* 74 (2024), pp. 1–14. ISSN: 1557-9662. DOI: 10.1109/tim.2024.3507052.

[102] Soichi Ibaraki, Koki Onodera, Wen-Yuh Jywe, Chia-Ming Hsu, and Yu-Wei Chang. "Experimental comparison of non-contact and tactile R-Test instruments in dynamic measurement". In: *The International Journal of Advanced Manufacturing Technology* 128.11–12 (Sept. 2023), pp. 5277–5288. ISSN: 1433-3015. DOI: 10.1007/s00170-023-12178-3.

[103] Hao Liu, Lei Liu, and Kai Shen. "Rotary axis calculation for five-axis FDM printer using a point-fitting optimization method". In: *Applied Mathematics-A Journal of Chinese Universities* 37.2 (June 2022), pp. 258–271. ISSN: 1993-0445. DOI: 10.1007/s11766-022-4586-3.

[104] Tom Schmolzi. "Calibrating a Low-cost, 5 Axis 3D Printer". MA thesis. Universität Hamburg, June 2024. URL: https://tams.informatik.uni-hamburg.de/publications/2024/MSc_Tom_Schmolzi.pdf.

[105] Madhav Moganti, Fikret Ercal, Cihan H. Dagli, and Shou Tsunekawa. "Automatic PCB Inspection Algorithms: A Survey". In: *Computer Vision and Image Understanding* 63.2 (Mar. 1996), pp. 287–313. ISSN: 1077-3142. DOI: 10.1006/cviu.1996.0020.

[106] Xing Chen, Yonglei Wu, Xingyou He, and Wuyi Ming. "A Comprehensive Review of Deep Learning-Based PCB Defect Detection". In: *IEEE Access* 11 (2023), pp. 139017–139038. ISSN: 2169-3536. DOI: 10.1109/access.2023.3339561.

[107] Alexandra Roberts, John True, Nathan T. Jessurun, and Dr. Navid Asadizanjani. "An Overview of 3D X-Ray Reconstruction Algorithms for PCB Inspection". In: *ISTFA 2020: Papers Accepted for the Planned 46th International Symposium for Testing and Failure Analysis*. Vol. 83348. ISTFA2020. ASM International, Dec. 2020, pp. 188–197. DOI: 10.31399/asm.cp.istfa2020p0188.

[108] Vikas Chaudhary, Ishan R. Dave, and Kishor P. Upla. "Automatic visual inspection of printed circuit board for defect detection and classification". In: *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. IEEE, Mar. 2017, pp. 732–737. DOI: 10.1109/wispnet.2017.8299858.

[109] Ajay Pal Singh Chauhan and Sharat Chandra Bhardwaj. "Detection of bare PCB defects by image subtraction method using machine vision". In: *Proceedings of the world congress on engineering* 2 (2011), pp. 6–8.

[110] Weibo Huang and Peng Wei. "A PCB Dataset for Defects Detection and Classification". In: *arXiv preprint arXiv:1901.08204* (2019). DOI: 10.48550/arXiv.1901.08204.

[111] Sasmita Mohapatra, Arpit Kabra, D. H. Gowda, Sudesh S. Gaonkar, and Supriyo Sadhukha. "PCB Defect Detection Using CNN-Based Deep Learning". In: *Soft Computing for Security Applications*. Springer Nature Singapore, 2023, pp. 363–376. ISBN: 9789819936083. DOI: 10.1007/978-981-99-3608-3_25.

[112] Bing Hu and Jianhui Wang. "Detection of PCB Surface Defects With Improved Faster-RCNN and Feature Pyramid Network". In: *IEEE Access* 8 (2020), pp. 108335–108345. ISSN: 2169-3536. DOI: 10.1109/access.2020.3001349.

[113] Yu-Ting Li, Paul Kuo, and Jiun-In Guo. "Automatic Industry PCB Board DIP Process Defect Detection System Based on Deep Ensemble Self-Adaption Method". In: *IEEE Transactions on Components, Packaging and Manufacturing Technology* 11.2 (Feb. 2021), pp. 312–323. ISSN: 2156-3985. DOI: 10.1109/tcpmt.2020.3047089.

[114] Anagha K Lailesh, Jolinson A Richi, and N Preethi. "A Pre-trained YOLO-v5 model and an Image Subtraction Approach for Printed Circuit Board Defect Detection". In: *2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*. IEEE, Jan. 2023, pp. 140–145. DOI: 10.1109/iitcee57236.2023.10090861.

[115] Archana Chaudhari, Ved Upganlawar, Tasmay Barve, Ram Vaidya, and Dheeraj Shelke. "Analysis of YOLO V3 for Multiple Defects Detection in PCB". In: *2024 Parul International Conference on Engineering and Technology (PICET)*. IEEE, May 2024, pp. 1–6. DOI: 10.1109/picet60765.2024.10716153.

[116] Wei Chen, Zhongtian Huang, Qian Mu, and Yi Sun. "PCB Defect Detection Method Based on Transformer-YOLO". In: *IEEE Access* 10 (2022), pp. 129480–129489. ISSN: 2169-3536. DOI: 10.1109/access.2022.3228206.

[117] Roozbeh (Ross) Salary, Jack P. Lombardi, M. Samie Tootooni, Ryan Donovan, Prahalad K. Rao, Peter Borgesen, and Mark D. Poliks. "Computational Fluid Dynamics Modeling and Online Monitoring of Aerosol Jet Printing Process". In: *Journal of Man-*

ufacturing Science and Engineering 139.2 (Oct. 2016). ISSN: 1528-8935. DOI: 10.1115/1.4034591.

[118] Roozbeh (Ross) Salary, Jack P. Lombardi, Prahalad K. Rao, and Mark D. Poliks. "Online Monitoring of Functional Electrical Properties in Aerosol Jet Printing Additive Manufacturing Process Using Shape-From-Shading Image Analysis". In: *Journal of Manufacturing Science and Engineering* 139.10 (Aug. 2017). ISSN: 1528-8935. DOI: 10.1115/1.4036660.

[119] Yifu Li, Karuniya Mohan, Hongyue Sun, and Ran Jin. "Ensemble Modeling of In Situ Features for Printed Electronics Manufacturing With In Situ Process Control Potential". In: *IEEE Robotics and Automation Letters* 2.4 (Oct. 2017), pp. 1864–1870. ISSN: 2377-3774. DOI: 10.1109/lra.2017.2713242.

[120] Haining Zhang, Seung Ki Moon, and Teck Hui Ngo. "Hybrid Machine Learning Method to Determine the Optimal Operating Process Window in Aerosol Jet 3D Printing". In: *ACS Applied Materials and Interfaces* 11.19 (Apr. 2019), pp. 17994–18003. ISSN: 1944-8252. DOI: 10.1021/acsami.9b02898.

[121] Julian Schirmer, Jewgeni Roudenko, Marcus Reichenberger, Simone Neermann, and Jorg Franke. "Print Quality Assessment by Image Processing Methods for Printed Electronics Applications". In: *2018 41st International Spring Seminar on Electronics Technology (ISSE)*. IEEE, May 2018, pp. 1–6. DOI: 10.1109/isse.2018.8443617.

[122] Jack P. Lombardi, Roozbeh (Ross) Salary, Darshana L. Weerawarne, Prahalada K. Rao, and Mark D. Poliks. "Image-Based Closed-Loop Control of Aerosol Jet Printing Using Classical Control Methods". In: *Journal of Manufacturing Science and Engineering* 141.7 (May 2019). ISSN: 1528-8935. DOI: 10.1115/1.4043659.

[123] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2009. DOI: 10.1109/cvpr.2009.5206848.

[124] Kazem Fayazbakhsh, Mobina Movahedi, and Jordan Kalman. "The impact of defects on tensile properties of 3D printed parts manufactured by fused filament fabrication". In: *Materials Today Communications* 18 (Mar. 2019), pp. 140–148. ISSN: 2352-4928. DOI: 10.1016/j.mtcomm.2018.12.003.

[125] Shuai Leng, Kiaran McGee, Jonathan Morris, Amy Alexander, Joel Kuhlmann, Thomas Vrieze, Cynthia H. McCollough, and Jane Matsumoto. "Anatomic modeling using 3D

printing: quality assurance and optimization". In: *3D Printing in Medicine* 3.1 (Mar. 2017). ISSN: 2365-6271. DOI: 10.1186/s41205-017-0014-3.

[126] Qing Yang Lu and Chee How Wong. "Additive manufacturing process monitoring and control by non-destructive testing techniques: challenges and in-process monitoring". In: *Virtual and Physical Prototyping* 13.2 (July 2017), pp. 39–48. ISSN: 1745-2767. DOI: 10.1080/17452759.2017.1351201.

[127] Haedong Jeong, Minsub Kim, Bumsoo Park, and Seungchul Lee. "Vision-Based Real-Time Layer Error Quantification for Additive Manufacturing". In: *Volume 2: Additive Manufacturing; Materials*. MSEC2017. American Society of Mechanical Engineers, June 2017. DOI: 10.1115/msec2017-2991.

[128] Benjamin Weiss, Duane W. Storti, and Mark A. Ganter. "Low-cost closed-loop control of a 3D printer gantry". In: *Rapid Prototyping Journal* 21.5 (Aug. 2015). Ed. by Dr Eujin Pei, pp. 482–490. ISSN: 1355-2546. DOI: 10.1108/rpj-09-2014-0108.

[129] Chiyen Kim, David Espalin, Alejandro Cuaron, Mireya A. Perez, Eric MacDonald, and Ryan B. Wicker. "Unobtrusive In Situ Diagnostics of Filament-Fed Material Extrusion Additive Manufacturing". In: *IEEE Transactions on Components, Packaging and Manufacturing Technology* 8.8 (Aug. 2018), pp. 1469–1476. ISSN: 2156-3985. DOI: 10.1109/tcpmt.2018.2847566.

[130] Yedige Tlegenov, Geok Soon Hong, and Wen Feng Lu. "Nozzle condition monitoring in 3D printing". In: *Robotics and Computer-Integrated Manufacturing* 54 (Dec. 2018), pp. 45–55. ISSN: 0736-5845. DOI: 10.1016/j.rcim.2018.05.010.

[131] Chenang Liu, David Roberson, and Zhenyu Kong. "Textural analysis-based online closed-loop quality control for additive manufacturing processes". In: *IIE Annual Conference. Proceedings*. Institute of Industrial and Systems Engineers (IISE). 2017, pp. 1127–1132.

[132] Santiago Blandon, Juan Camilo Amaya, and Alvaro Jose Rojas. "Development of a 3D printer and a supervision system towards the improvement of physical properties and surface finish of the printed parts". In: *2015 IEEE 2nd Colombian Conference on Automatic Control*. IEEE, Oct. 2015, pp. 1–7. DOI: 10.1109/ccac.2015.7345179.

[133] Alvaro J Rojas Arciniegas and Juan C Amaya Hurtado. "Development of a Supervision System: Towards Closing the Control Loop in 3D Printing Systems". In: *NIP and*

Digital Fabrication Conference 32.1 (Sept. 2016), pp. 221–226. ISSN: 2169-4451. DOI: 10.2352/issn.2169-4451.2017.32.221.

[134] Zeqing Jin, Zhizhou Zhang, and Grace X. Gu. "Autonomous in-situ correction of fused deposition modeling printers using computer vision and deep learning". In: *Manufacturing Letters* 22 (Oct. 2019), pp. 11–15. ISSN: 2213-8463. DOI: 10.1016/j.mfglet.2019.09.005.

[135] Chenang Liu, Andrew Chung Chee Law, David Roberson, and Zhenyu (James) Kong. "Image analysis-based closed loop quality control for additive manufacturing with fused filament fabrication". In: *Journal of Manufacturing Systems* 51 (Apr. 2019), pp. 75–86. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2019.04.002.

[136] Douglas A. J. Brion and Sebastian W. Pattinson. "Generalisable 3D printing error detection and correction via multi-head neural networks". In: *Nature Communications* 13.1 (Aug. 2022). ISSN: 2041-1723. DOI: 10.1038/s41467-022-31985-y.

[137] Hongyao Shen, Weijun Sun, and Jianzhong Fu. "Multi-view online vision detection based on robot fused deposit modeling 3D printing technology". In: *Rapid Prototyping Journal* 25.2 (Mar. 2019), pp. 343–355. ISSN: 1355-2546. DOI: 10.1108/rpj-03-2018-0052.

[138] Fabio Caltanissetta, Gregory Dreifus, Anastasios John Hart, and Bianca Maria Colosimo. "In-situ monitoring of Material Extrusion processes via thermal videoimaging with application to Big Area Additive Manufacturing (BAAM)". In: *Additive Manufacturing* 58 (Oct. 2022), p. 102995. ISSN: 2214-8604. DOI: 10.1016/j.addma.2022.102995.

[139] Youssef AbouelNour and Nikhil Gupta. "Assisted defect detection by in-process monitoring of additive manufacturing using optical imaging and infrared thermography". In: *Additive Manufacturing* 67 (Apr. 2023), p. 103483. ISSN: 2214-8604. DOI: 10.1016/j.addma.2023.103483.

[140] Abdalla R. Nassar, Jayme S. Keist, Edward W. Reutzel, and Todd J. Spurgeon. "Intralayer closed-loop control of build plan during directed energy additive manufacturing of Ti–6Al–4V". In: *Additive Manufacturing* 6 (Apr. 2015), pp. 39–52. ISSN: 2214-8604. DOI: 10.1016/j.addma.2015.03.005.

[141] Wu Yi, He Ketai, Zhou Xiaomin, and Ding Wenying. "Machine vision based statistical process control in fused deposition modeling". In: *2017 12th IEEE Conference on*

Industrial Electronics and Applications (ICIEA). IEEE, June 2017, pp. 936–941. DOI: 10.1109/iciea.2017.8282973.

[142] Ugandhar Delli and Shing Chang. "Automated Process Monitoring in 3D Printing Using Supervised Machine Learning". In: *Procedia Manufacturing* 26 (2018), pp. 865–870. ISSN: 2351-9789. DOI: 10.1016/j.promfg.2018.07.111.

[143] Ammar Malik, Hugo Lhachemi, Joern Ploennigs, Amadou Ba, and Robert Shorten. "An Application of 3D Model Reconstruction and Augmented Reality for Real-Time Monitoring of Additive Manufacturing". In: *Procedia CIRP* 81 (2019), pp. 346–351. ISSN: 2212-8271. DOI: 10.1016/j.procir.2019.03.060.

[144] Aliaksei L. Petsiuk and Joshua M. Pearce. "Open source computer vision-based layer-wise 3D printing analysis". In: *Additive Manufacturing* 36 (Dec. 2020), p. 101473. ISSN: 2214-8604. DOI: 10.1016/j.addma.2020.101473.

[145] Oluwole K. Bowoto, S. Abolfazl Zahedi, and Seng Chong. "Enhancing dimensional accuracy in 3D printing: a novel software algorithm for real-time quality assessment". In: *The International Journal of Advanced Manufacturing Technology* 129.7–8 (Oct. 2023), pp. 3435–3446. ISSN: 1433-3015. DOI: 10.1007/s00170-023-12543-2.

[146] Manpreet Singh, Fujun Ruan, Albert Xu, Yuchen Wu, Archit Rungta, Luyuan Wang, Kevin Song, Howie Choset, and Lu Li. "Toward Closed-Loop Additive Manufacturing: Paradigm Shift in Fabrication, Inspection, and Repair". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2023. DOI: 10.1109/iros55552.2023.10342148.

[147] Ling Li, Ryan McGuan, Robert Isaac, Pirouz Kavehpour, and Robert Candler. "Improving precision of material extrusion 3D printing by in-situ monitoring and predicting 3D geometric deviation using conditional adversarial networks". In: *Additive Manufacturing* 38 (Feb. 2021), p. 101695. ISSN: 2214-8604. DOI: 10.1016/j.addma.2020.101695.

[148] Weiyi Lin, Hongyao Shen, Jianzhong Fu, and Senyang Wu. "Online quality monitoring in material extrusion additive manufacturing processes based on laser scanning technology". In: *Precision Engineering* 60 (Nov. 2019), pp. 76–84. ISSN: 0141-6359. DOI: 10.1016/j.precisioneng.2019.06.004.

[149] Javid Akhavan, Jiaqi Lyu, and Souran Manoochehri. "A deep learning solution for real-time quality assessment and control in additive manufacturing using point cloud data".

In: *Journal of Intelligent Manufacturing* 35.3 (Apr. 2023), pp. 1389–1406. ISSN: 1572-8145. DOI: 10.1007/s10845-023-02121-4.

[150] Marc Preissler, Chen Zhang, Maik Rosenberger, and Gunther Notni. "Platform for 3D inline process control in additive manufacturing". In: *Optical Measurement Systems for Industrial Inspection X*. Ed. by Peter Lehmann, Wolfgang Osten, and Armando Albertazzi Gonçalves. Vol. 10329. SPIE, June 2017, 103290R. DOI: 10.1117/12.2270493.

[151] M Preissler, J Broghammer, M Rosenberger, and G Notni. "Inline process monitoring method for geometrical characteristics in additive manufacturing". In: *Journal of Physics: Conference Series* 1044 (June 2018), p. 012035. ISSN: 1742-6596. DOI: 10.1088/1742-6596/1044/1/012035.

[152] Marc Preissler, Chen Zhang, Maik Rosenberger, and Gunther Notni. "Approach for Process Control in Additive Manufacturing Through Layer-Wise Analysis with 3-Dimensional Pointcloud Information". In: *2018 Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, Dec. 2018, pp. 1–6. DOI: 10.1109/dicta.2018.8615803.

[153] Xinyue Zhao, Qiaolong Lian, Zaixing He, and Shuyou Zhang. "Region-based online flaw detection of 3D printing via fringe projection". In: *Measurement Science and Technology* 31.3 (Dec. 2019), p. 035011. ISSN: 1361-6501. DOI: 10.1088/1361-6501/ab524b.

[154] Paschalis Charalampous, Ioannis Kostavelis, Charalampos Kopsacheilis, and Dimitrios Tzovaras. "Vision-based real-time monitoring of extrusion additive manufacturing processes for automatic manufacturing error detection". In: *The International Journal of Advanced Manufacturing Technology* 115.11–12 (June 2021), pp. 3859–3872. ISSN: 1433-3015. DOI: 10.1007/s00170-021-07419-2.

[155] Jack Girard and Song Zhang. "Fast error detection method for additive manufacturing process monitoring using structured light three dimensional imaging technique". In: *Optics and Lasers in Engineering* 184 (Jan. 2025), p. 108609. ISSN: 0143-8166. DOI: 10.1016/j.optlaseng.2024.108609.

[156] Fabio Caltanissetta, Marco Grasso, Stefano Petrò, and Bianca Maria Colosimo. "Characterization of in-situ measurements based on layerwise imaging in laser powder bed fusion". In: *Additive Manufacturing* 24 (Dec. 2018), pp. 183–199. ISSN: 2214-8604. DOI: 10.1016/j.addma.2018.09.017.

[157] Jianjing Zhang, Peng Wang, and Robert X. Gao. "Deep learning-based tensile strength prediction in fused deposition modeling". In: *Computers in Industry* 107 (May 2019), pp. 11–21. ISSN: 0166-3615. DOI: 10.1016/j.compind.2019.01.011.

[158] Konstantinos Paraskevoudis, Panagiotis Karayannis, and Elias P. Koumoulos. "Real-Time 3D Printing Remote Defect Detection (Stringing) with Computer Vision and Artificial Intelligence". In: *Processes* 8.11 (Nov. 2020), p. 1464. ISSN: 2227-9717. DOI: 10.3390/pr8111464.

[159] Siranee Nuchitprasitchai, Michael Roggemann, and Joshua M. Pearce. "Factors effecting real-time optical monitoring of fused filament 3D printing". In: *Progress in Additive Manufacturing* 2.3 (June 2017), pp. 133–149. ISSN: 2363-9520. DOI: 10.1007/s40964-017-0027-x.

[160] Jeremy Straub. "Initial Work on the Characterization of Additive Manufacturing (3D Printing) Using Software Image Analysis". In: *Machines* 3.2 (Apr. 2015), pp. 55–71. ISSN: 2075-1702. DOI: 10.3390/machines3020055.

[161] Yuanbin Wang, Jiakang Huang, Yuan Wang, Sihang Feng, Tao Peng, Huayong Yang, and Jun Zou. "A CNN-Based Adaptive Surface Monitoring System for Fused Deposition Modeling". In: *IEEE/ASME Transactions on Mechatronics* 25.5 (Oct. 2020), pp. 2287–2296. ISSN: 1941-014X. DOI: 10.1109/tmech.2020.2996223.

[162] Antony Orth, Kathleen L. Sampson, Yujie Zhang, Kayley Ting, Derek Aranguren van Egmond, Kurtis Laqua, Thomas Lacelle, Daniel Webber, Dorothy Fatehi, Jonathan Boisvert, and Chantal Paquet. "On-the-fly 3D metrology of volumetric additive manufacturing". In: *Additive Manufacturing* 56 (Aug. 2022), p. 102869. ISSN: 2214-8604. DOI: 10.1016/j.addma.2022.102869.

[163] Yanzhou Fu, Austin R.J. Downey, Lang Yuan, and Hung-Tien Huang. "Real-time structural validation for material extrusion additive manufacturing". In: *Additive Manufacturing* 65 (Mar. 2023), p. 103409. ISSN: 2214-8604. DOI: 10.1016/j.addma.2023.103409.

[164] Robert J. Woodham. "Photometric Method For Determining Surface Orientation From Multiple Images". In: *Optical Engineering* 19.1 (Feb. 1980). ISSN: 0091-3286. DOI: 10.1117/12.7972479.

[165] Johann Heinrich Lambert. *Photometria*. 1760.

[166] Martín Abadi. “TensorFlow: learning functions at scale”. In: *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*. ICFP’16. ACM, Sept. 2016. DOI: 10.1145/2951913.2976746.

Appendix

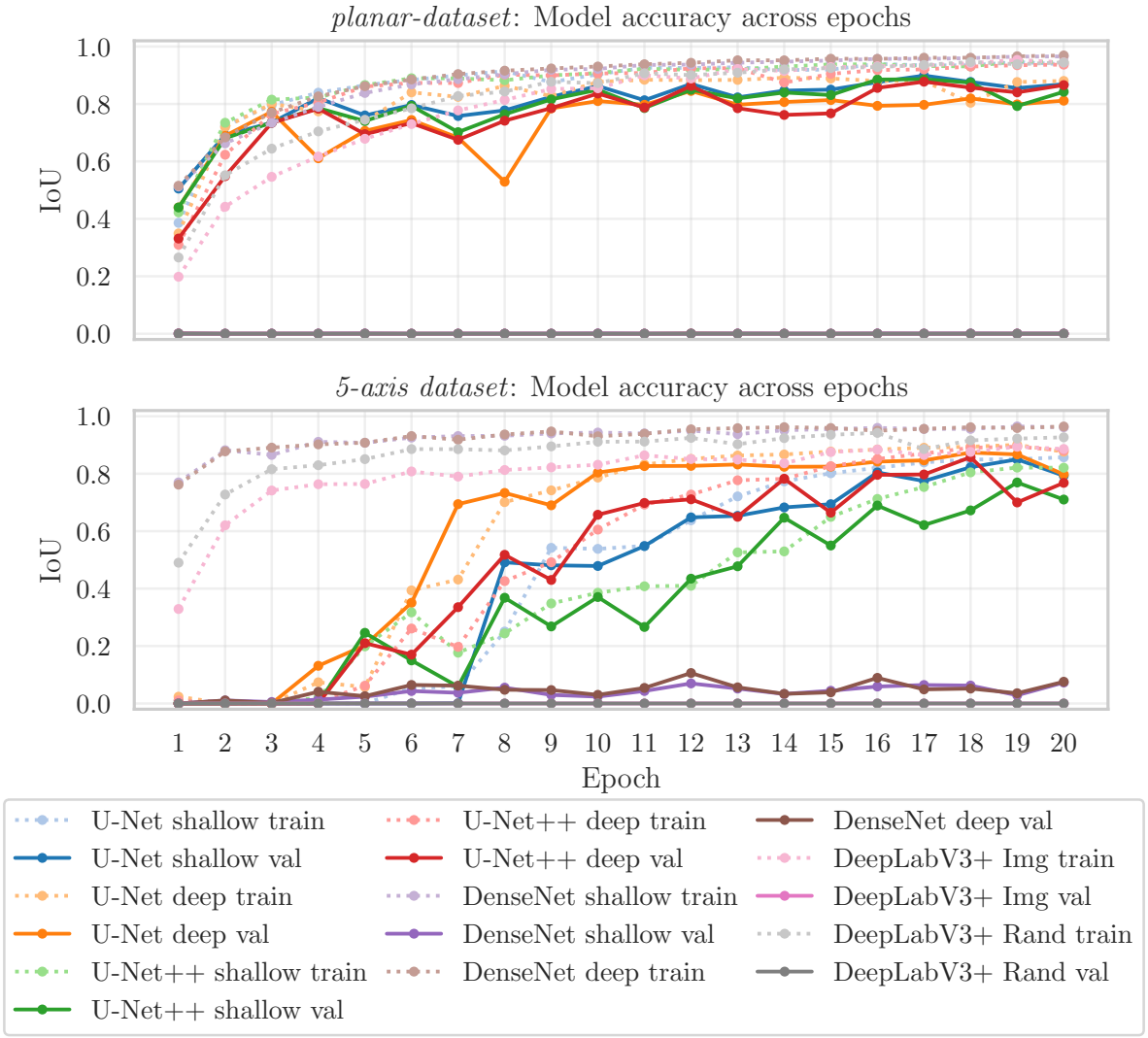


Figure A.1 – Training and validation IoU over epochs of all models. The *planar-dataset* is shown on the top and the *5-axis dataset* on the bottom. The plots are averaged over five training runs with different random seeds.

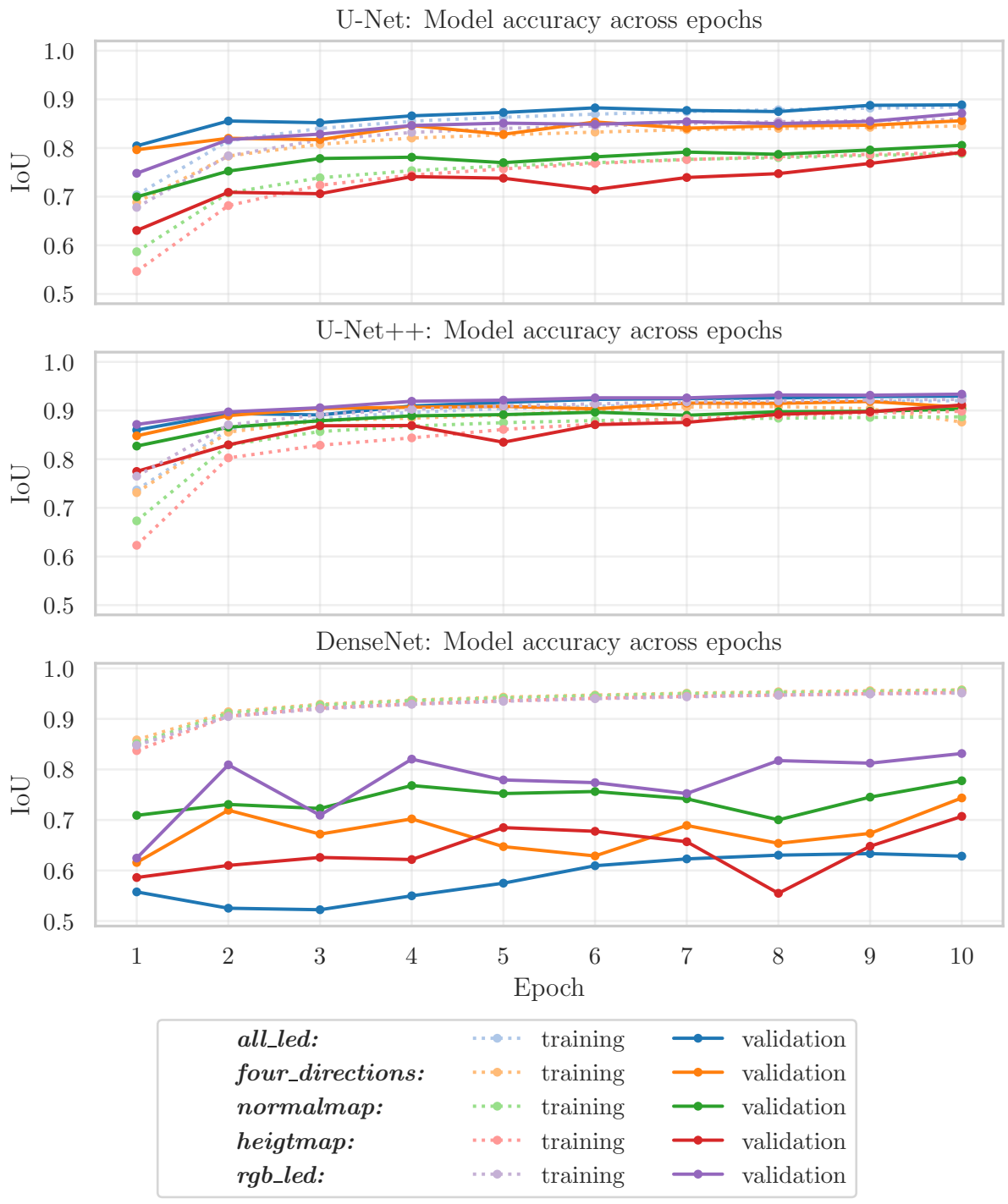


Figure A.2 – Training and validation IoU over epochs of all different inputs. The top plot shows the results in a U-Net, the middle plot in a U-Net++ and the bottom plot in a DenseNet. The plots are averaged over three training runs with different random seeds. The plots are cut at an IoU of 0.5 for better visibility.

Figure A.3 – Printed objects for the layer segmentation dataset. The objects are: 3DBenchy (brown), Beer crate (grey), Cookies cutter (blue), Motor plate (dark green), Gear bearing (gold), Hex-Scraper (translucent blue), Octopus (red), Easter egg (white), Measuring-Tool (black), Spiral-Planter (yellow) and Whistle (green).

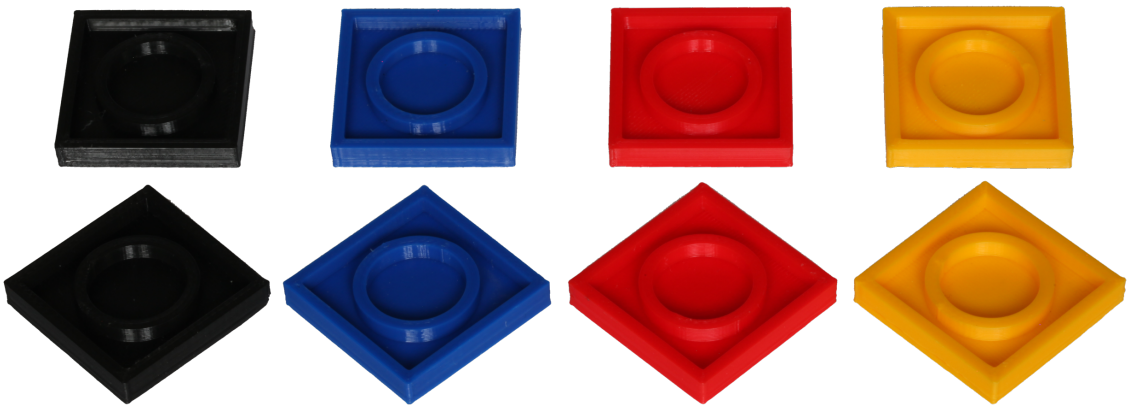


Figure A.4 – Printed test objects used to measure dimensional accuracy. For each color, two prints were produced: one in the default orientation and one rotated by 45 °.

List of Web-Adresses

[W1] Metal AM. *SpaceX reveals its Raptor 3 engine, further enhanced with metal AM*. Accessed: 26.09.2025. URL: https://www.metal-am.com/spacex-debuts-raptor-3-engine-further-enhanced-with-metal-additive-manufacturing/.

[W2] 3D Systems. Accessed: 21.01.2025. URL: https://www.3dsystems.com/.

[W3] Stratasys. Accessed: 21.01.2025. URL: https://www.stratasys.com.

[W4] Thingiverse. Accessed: 31.01.2025. URL: https://www.thingiverse.com/.

[W5] Printables. Accessed: 31.01.2025. URL: https://www.printables.com/.

[W6] Cura. Accessed: 31.01.2025. URL: https://github.com/Ultimaker/Cura.

[W7] PrusaSlicer. Accessed: 31.01.2025. URL: https://github.com/prusa3d/PrusaSlicer.

[W8] OrcaSlicer. Accessed: 31.01.2025. URL: https://github.com/SoftFever/OrcaSlicer.

[W9] Prusa i3 MK4S. Accessed: 31.01.2025. URL: https://www.prusa3d.com/product/original-prusa-mk4s-3d-printer-5/.

[W10] Bambu Lab X1. Accessed: 31.01.2025. URL: https://bambulab.com/en-us/x1.

[W11] Ilan E. Moyer. *CoreXY*. Accessed: 18.11.2025. URL: https://corexy.com.

[W12] Autodesk Inc. *Voxel8 Wires Up a New Era in 3D Printing Technology*. Accessed: 15.01.2025. URL: https://www.autodesk.com/products/fusion-360/blog/voxel8-wires-up-a-new-era-in-3d-printing-technology.

[W13] Neotech AMT. Accessed: 14.03.2025. URL: neotech-amt.com.

[W14] Schott Pictures by PC. Accessed: 14.03.2024. URL: https://www.schott-systeme.de/index.php/de/.

[W15] E3D Online. *ToolChanger repository*. Accessed: 18.11.2025. URL: https://github.com/e3donline/ToolChanger.

[W16] Duet3D. *Duet 3 Main Board*. Accessed: 18.11.2025. URL: https://www.duet3d.com/duet3mainboard6xd.

[W17] Octoprint. Accessed: 11.04.2025. URL: https://octoprint.org/.

[W18] OctoPNP. Accessed: 11.04.2025. URL: https://github.com/platsch/OctoPNP.

[W19] OctoCameraDocumentation. Accessed: 11.04.2025. URL: https://github.com/Fragjacker/OctoCamDox.

[W20] CNC-Step. *3D-Finder – Messtaster*. Accessed: 18.11.2025. URL: https://www.cnc-step.de/3d-finder-messtaster-taster/.

[W21] *Unified Robot Description Format*. Accessed: 15.07.2025. URL: https://wiki.ros.org/urdf.

[W22] Kronos Mechatronics GmbH. *Aion-5X Software*. Accessed: 17.07.2025. URL: https://kronos-mct.com/produktportfolio/#software-aion-5x.

[W23] Robert McNeel & Associates. *Rhino 8*. Accessed: 15.07.2025. URL: https://www.rhino3d.com/.

[W24] Zmorph S.A. *ZMorph Closed Loop System*. Accessed: 12.12.2024. URL: https://zmorph3d.com/blog/closed-loop-system-explained/.

[W25] Stellamove. *Stellamove 3D-print system*. Accessed: 12.12.2024. URL: https://www.stellamove.com.

[W26] Sciaky Inc. *IRISS Interlayer Realtime Imaging and Sensing System*. Accessed: 11.12.2024. URL: https://www.sciaky.com/additive-manufacturing/iriss-closed-loop-control.

[W27] NIT New Infrared Technologies. *Clamir closed-loop laser power control system*. Accessed: 12.12.2024. URL: https://www.clamir.com/specifications/.

[W28] Stratonics Inc. *ThermaViz Sensor*. Accessed: 16.12.2024. URL: https://stratonics.com/systems/.

[W29] Inkbit. *Additive Manufacturing Production System*. Accessed: 16.12.2024. URL: https://inkbit3d.com/.

[W30] obico. Accessed: 15.01.2025. URL: https://www.obico.io/.

[W31] *3DBenchy*. Accessed: 03.05.2025. URL: https://www.printables.com/model/3161-3d-benchy.

[W32] *beer_crate_AA.stl: Beer Crate Battery Holder AA AAA Boxes*. Accessed: 03.05.2025. URL: https://www.printables.com/model/149774-beer-crate-battery-holder-aa-aaa-boxes.

[W33] *Cookies cute dinosaurs kit.stl: Dino Cookie Cutters*. Accessed: 03.05.2025. URL: https://www.printables.com/model/387301-dino-cookie-cutters.

[W34] *extruder-motor-plate.stl: i3 MK3S Printable Parts*. Accessed: 03.05.2025. URL: https://www.printables.com/model/88272-i3-mk3s-printable-parts.

[W35] *gear_bearing.stl: Herringbone Planetary Gear*. Accessed: 03.05.2025. URL: https://www.printables.com/model/3119-herringbone-planetary-gear.

[W36] *hexScraper.stl: Hexscraper Printbed Scraper*. Accessed: 03.05.2025. URL: https://www.printables.com/model/122894-hexscraper-printbed-scraper.

[W37] *Octopus_spiral_sup_v6.stl: Cute Mini Octopus*. Accessed: 03.05.2025. URL: https://www.printables.com/model/178035-cute-mini-octopus.

[W38] *Osterhase_Ei.stl: Easter Egg with Bunny Ears*. Accessed: 03.05.2025. URL: https://www.printables.com/model/1235856-easter-egg-with-bunny-ears.

[W39] *Screw-Measuring-Tool-50mm-M5.stl: Rapid Metric Screw Measuring Tool M2-M5 up to 50mm*. Accessed: 03.05.2025. URL: https://www.printables.com/model/208880-rapid-metric-screw-measuring-tool-m2-m5-up-to-50mm.

[W40] *spiral_planter_solidBottom.stl: Modern Spiral Planter*. Accessed: 03.05.2025. URL: https://www.printables.com/model/225251-modern-spiral-planter.

[W41] *V29.2_Whistle_V29.stl: V29 Whistle*. Accessed: 03.05.2025. URL: https://www.printables.com/model/285368-v29-whistle.

[W42] Keras. Accessed: 21.05.2025. URL: https://keras.io/.

[W43] *Potrace*. Accessed: 03.06.2025. URL: https://potrace.sourceforge.net/.

[W44] OpenSCAD. Accessed: 03.06.2025. URL: https://openscad.org/.

[W45] *pikachu_lowpoly_remastered_flowalistik_0.3.stl: Low Poly Pikachu Remastered*. Accessed: 03.05.2025. URL: https://www.printables.com/model/369173-low-poly-pikachu-remastered.

List of Figures

List of Tables

Acronyms

ABS	Acrylonitrile Butadiene Styrene
AHCA	Agglomerative Hierarchical Clustering Analysis
AM	Additive Manufacturing
ASPP	Atrous Spatial Pyramid Pooling
BJT	Binder Jetting
Brep	Boundary Representation
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CLFDM	Curved Layer Fused Deposition Modeling
CNC	Computer Numerical Control
CNN	Convolutional Neural Network
CSI	Camera Serial Interface
CT	Computed Tomography
DED	Direct Energy Deposition
DH	Denavit-Hartenberg
DLP	Digital Light Processing
DOF	Degrees of Freedom
EBM	Electron Beam Melting
ECAD	Electronic Computer-Aided Design
FDM	Fused Deposition Modeling
FED	Fachverband Elektronikdesign und -fertigung e.V.
FFF	Fused Filament Fabrication
HSV	Hue Saturation Value
IC	Integrated Circuit
IK	Inverse Kinematics
IoU	Intersection over Union
ISO	International Organization for Standardization
JSC	Johnson Space Center
LCD	Liquid Crystal Display
LED	Light Emitting Diode

LPBF	Laser Powder Bed Fusion
MEX	Material Extrusion
MJT	Material Jetting
NASA	National Aeronautics and Space Administration
PA	Polyamide
PBF	Powder Bed Fusion
PC	Polycarbonate
PCB	Printed Circuit Board
PEI	Polyetherimide
PET	Polyethylene Terephthalate
PETG	Polyethylene Terephthalate Glycol
PID	Proportional-Integral-Derivative
PLA	Polylactic Acid
PU	Polyurethane
PVC	Polyvinyl Chloride
PWM	Pulse Width Modulation
ReLU	Rectified Linear Unit
RepRap	Replicating Rapid-prototyper
RGB	Red Green Blue
ROS	Robot Operating System
SHL	Sheet Lamination
SLA	Stereolithography
SLS	Selective Laser Sintering
SMD	Surface Mounted Device
STEP	Standard for the Exchange of Product Data
STL	Standard Tessellation Language
SVM	Support Vector Machine
TPU	Thermoplastic Polyurethane
URDF	Unified Robot Description Format
UV	Ultraviolet
VPP	Vat Photopolymerization
WAAM	Wire Arc Additive Manufacturing
ZIM-KamEl	Zentrales Innovationsprogramm Mittelstand-Kamerabasierte Fehlererkennung für 3D-gedruckte Elektronik

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Sofern im Zuge der Erstellung der vorliegenden Dissertationsschrift generative Künstliche Intelligenz (gKI) basierte elektronische Hilfsmittel verwendet wurden, versichere ich, dass meine eigene Leistung im Vordergrund stand und dass eine vollständige Dokumentation aller verwendeten Hilfsmittel gemäß der Guten wissenschaftlichen Praxis vorliegt. Ich trage die Verantwortung für eventuell durch die gKI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate.

Hamburg, 16. Dezember 2025

Daniel Ahlers