DTU Electro
Department of Electrical and Photonics Engineering

Computer Vision-based Detection and Robotic Handling of Liquids for Use in Cell Culture Automation

Master Thesis

Computer Vision-based Detection and Robotic Handling of Liquids for Use in Cell Culture Automation

Master Thesis
July, 2023

By
Daniel Schober

Cover photo: Generative AI picture using DreamStudio by stability.ai
Published by: DTU, Department of Electrical and Photonics Engineering, Ørsteds Pl. 348, 2800 Kgs. Lyngby Denmark
www.dtu.dk

Approval

This thesis has been prepared over six months at the Automation and Control group, Department of Electrical and Photonics Engineering, at the Technical University of Denmark, DTU, in partial fulfillment for the degree Master of Science in Engineering, MSc Eng.

It is assumed that the reader has basic knowledge in the areas of automation, mathematics, and statistics.

Daniel Schober - s212599

.....

Signature

.....

Date

Abstract

Observing and handling liquids is one of the scientists' most executed and time-consuming tasks in research laboratories. To effectively deploy robots to work alongside research scientists in research laboratories, they must be able to detect transparent containers and liquids and perform pouring tasks like human scientists. One of the main difficulties in automating workflows in research laboratories is the cost of introducing specialized and specifically designed hardware, which is often immutable and not reusable.

This work investigates the opportunities of a vision- and simulation-based approach to make a robot a flexible and adaptable laboratory assistant, demonstrated with a proposed new low-batch system for cell culture automation. Cell culture is one of the fundamental tools in life science research. It describes the process of maintaining and growing cells under controlled physiological conditions in artificial environments outside of living organisms. Automating the required manual processes can free up time for researchers, make working hours more flexible, increase data generation, and reduce the use of disposable plastics. Existing laboratory hardware and consumables used in manual cell culture applications are combined with a robot arm, which executes all human tasks, including the pouring of liquids.

A two-step vision-based system is used for liquid volume estimation. First, segmented depth maps of transparent containers and the liquids inside are generated based on a convolutional neural network (CNN) trained on computer-generated images. In the second step, the volume of the liquid inside the container is estimated based on those results using a CNN trained on a newly generated dataset, which consists of more than 5,000 images of liquids inside laboratory containers and their volume. Based on the liquid volume estimation for the pouring container, the best pouring movement is selected from more than 6,000 simulated pours. Unlike previous autonomous pouring approaches, the pouring container rotates around the liquid exit point, making the trajectory more suitable for pouring into containers with small openings.

The liquid volume can be estimated with a mean absolute percentage error of less than 17% just providing an RGB image and of less than 10% when the container volume is given. The liquid segmentation and volume estimation outcomes demonstrate superior performance when applied to glass containers compared to plastic containers. The simulation-to-reality transfer for pouring using a robot arm shows reasonable accuracy, with potential for improvement when using accurate 3D models of the containers and adapting the liquid parameters. The new pouring trajectory results in low spilling in real executions. The system prototype demonstrates the possibilities of integrating existing hardware and a robot arm for laboratory automation by successfully executing the three required workflows for cell culture automation.

Acknowledgements

I want to express my deep gratitude and appreciation to my two supervisors, who have supported and contributed to the successful completion of my master's thesis:

Lazaros Nalpantidis, Professor of Autonomous Systems at the Department of Electrical and Photonics Engineering, Technical University of Denmark (DTU).
University supervisor of the thesis.

James Love, Vice President, Automation and Process Optimization, Novo Nordisk.
Company supervisor of the thesis.

While it is impossible to mention every individual who has influenced my work, I extend my gratitude to everyone who has played a part, no matter how big or small, in the realization of this thesis.

Contents

1 Introduction

This thesis explores the opportunities for an autonomous cell culture system. For that, existing laboratory hardware already used in the process will be integrated for a proof-of-concept prototype. To execute the required steps, it is aimed to perform human tasks with liquids using one single robot arm. More precisely, computer vision techniques combined with a deep learning approach will be used to detect liquids and estimate their volume, to be then able to simulate and execute the liquid pouring steps. This section presents the larger scientific context of the underlying research problem, followed by potentials and restrictions of current systems and an overview of the proposed solution. Finally, the structure of this work is explained.

1.1 Research Background

The gradual incorporation of autonomous systems into various work settings has notably impacted the productivity, effectiveness, and caliber of a diverse range of industrial procedures [1]. Recent improvements in advanced technology areas like robotics and artificial intelligence have automated processes that were previously only achievable through human labor. Although automated equipment has been increasingly integrated into life science laboratories in the last few years, it is noticeable that a high level of manual manipulation remains. Many experimental procedures still require researchers to carry out protocols manually in the research facility [2]. The life sciences are a group of disciplines that deal with living organisms' biochemical, microbiological, and molecular aspects and cover a wide range of life processes, from microbes to animals, fungi, and plants. Through experimental methods and in-depth investigations, they seek to deepen the understanding of the complexity and diversity of life forms and their biological mechanisms [3].

1.1.1 Motivation

A text mining study by Groth and Cox in 2017 shows that 89% of the analyzed life science articles use a manual protocol that could be automated using current technologies [4]. Possible reasons for this automation gap are the high initial investment costs, high variability in research protocols, and the lack of lower-cost interim labor-saving automation options. Despite ongoing efforts by life scientists to overcome these challenges by isolating and automating specific parts or segments of their workflows, the manual programming required for each subsystem remains a significant obstacle, diverting valuable time and resources from actual research activities. There is, therefore, a growing need for more comprehensive and efficient automation solutions that streamline the entire research process from data collection to analysis and help researchers achieve their goals more effectively.

A recent survey by Holland et al. [2] highlights three significant pain points for autonomous systems in research laboratories: (1) difficulty of use, (2) high initial cost of investment, and (3) inability to use a single system for multiple purposes. Hence, to expand the range of automation options suitable for research laboratories, there is a need for more flexible, modular, and cost-effective designs. This work is motivated by the current lack of affordable labor-saving automation alternatives, particularly in the pharmaceutical wet lab industry. Despite the availability of high-level and large-batch solution machines, scientists often struggle with the inefficiency of performing repetitive low-batch experiments. This results in a significant amount of time being spent on such tasks. Consequently, this work aims to investigate more cost-effective automation options that can enhance laboratory productivity and efficiency. The main advantages of an increased automation level in research laboratories lie in improved efficiency of the researchers, higher safety for operators, and better reproducibility through standardization [5].

Additionally, traditional automation systems are often designed for specific tasks, making them inflexible and challenging to adapt to new applications. Computer vision technology provides a solution to many of these challenges by allowing the development of cost-effective and adaptable autonomous systems. Specifically, in laboratory environments, computer vision could enable robots to detect and handle liquids and transparent laboratory containers in the same precise and efficient manner as humans, which could help make a robot arm an intelligent and adaptable laboratory assistant. Furthermore, by integrating computer vision-based liquid detection and handling capabilities with existing laboratory equipment, it is possible to create modular and scalable systems that can be customized for different applications. This can help reduce automation costs while providing a more flexible solution for laboratory automation.

1.1.2 Significance of Cell Culture Automation

This thesis will explore how computer vision-based liquid detection and handling can be integrated with robotic systems to develop a comprehensive and versatile automation system for cell culture applications for low throughput. Cell culture is one of the fundamental tools in life science research and biotechnology. Applications range from testing drugs or toxins, development of gene and cell therapies, and investigation of the function of biomolecules to the production of biologics or vaccine particles. It enables consistent and reproducible results by utilizing clonal cells, increasing accuracy and reliability in data analysis [6]. It describes the process of maintaining and growing cells under controlled physiological conditions in artificial environments outside of living organisms. Both primary cell cultures and cell lines are valuable models for exploring cell physiology and biochemistry in healthy and diseased states. Primary cell cultures consist of dispersed cells that are directly obtained from tissues and have a limited lifespan, while cell lines are immortalized cells that can be cultured infinitely [7, 8].

The culture conditions for each cell type are highly specific, but generally, the in vitro environment for cell culture requires a suitable vessel containing a range of essential components. These include a media that provides crucial nutrients. In addition, growth factors, hormones, and gases such as oxygen and carbon dioxide are provided to support the survival and development of the cells. The culture environment must also maintain a carefully regulated physicochemical environment, including pH, osmotic pressure, and temperature, to ensure optimal conditions for the cells [8].

There are entirely automated high-level systems for the process of cell culture for high throughput (see subsection 2.1.4). Examples are the StemmCellFactory [9], which can handle up to 60 different cell lines in parallel, and Formulatrix Cell Culture System [10], a fully automated and enclosed platform. However, there are currently no examples of commercially available low-cost solutions for cell culture automation for low throughput [2]. Several advantages could be gained from implementing this solution. Worth mentioning are a decrease in contamination, an increase in the rate of data generation, a reduction in human-induced variability, an efficiency improvement of researchers, and a transition from manual to cognitive labor [11, 12, 2]. Specifically for cell culture automation in research laboratories, one main factor is improving working times and flexibility. Researchers can schedule processes to run autonomously over extended periods, including weekends and evenings, by automating tasks such as media changing and cell passaging. This flexibility not only improves the efficiency of experimental procedures but also reduces the burden on researchers who would otherwise need to be physically present in the laboratory outside of regular working hours. Additionally, the amount of disposable plastic can be reduced by replacing the manual work of pipetting in the process through robotic pouring. Only considering research institutions, laboratories are estimated to produce more than 5.5 million tons of plastic waste each year [13].

1.2 Outline of the Problem and the Proposed Solution

The focus lies on the problem of maintaining adherent cell cultures. This includes three main procedures: (1) Analyzing the cells, (2) changing media, and (3) passaging (also known as subculturing) of cells. For adherent cell cultures, it is necessary to repeatedly remove the spent media and replace it with fresh media. To achieve optimal cell proliferation, it is essential to provide cells with fresh media two to three times per week [8]. When cells reach confluency, it is crucial to passage them. Otherwise, they will experience reduced mitotic index and eventually result in cell death. The interval for cell passaging varies based on the cell line and its growth rate [14]. The growth of cells in culture can be divided into four distinct phases: (1) the lag phase, which is the period of minimal growth before cell growth begins, (2) the log phase, which is characterized by exponential growth, (3) the stationary phase, during which cell growth slows down and reaches a plateau, and (4) the death phase, where cells die due to lack of nutrients and unfavorable living conditions. To maintain optimal growth conditions, it is recommended to change the culture media during the log phase and to passage cells before they reach the stationary phase [15]. An exemplary growth curve of adherent cells with the described phases and a suitable time for passaging is shown in fig. 1.1.

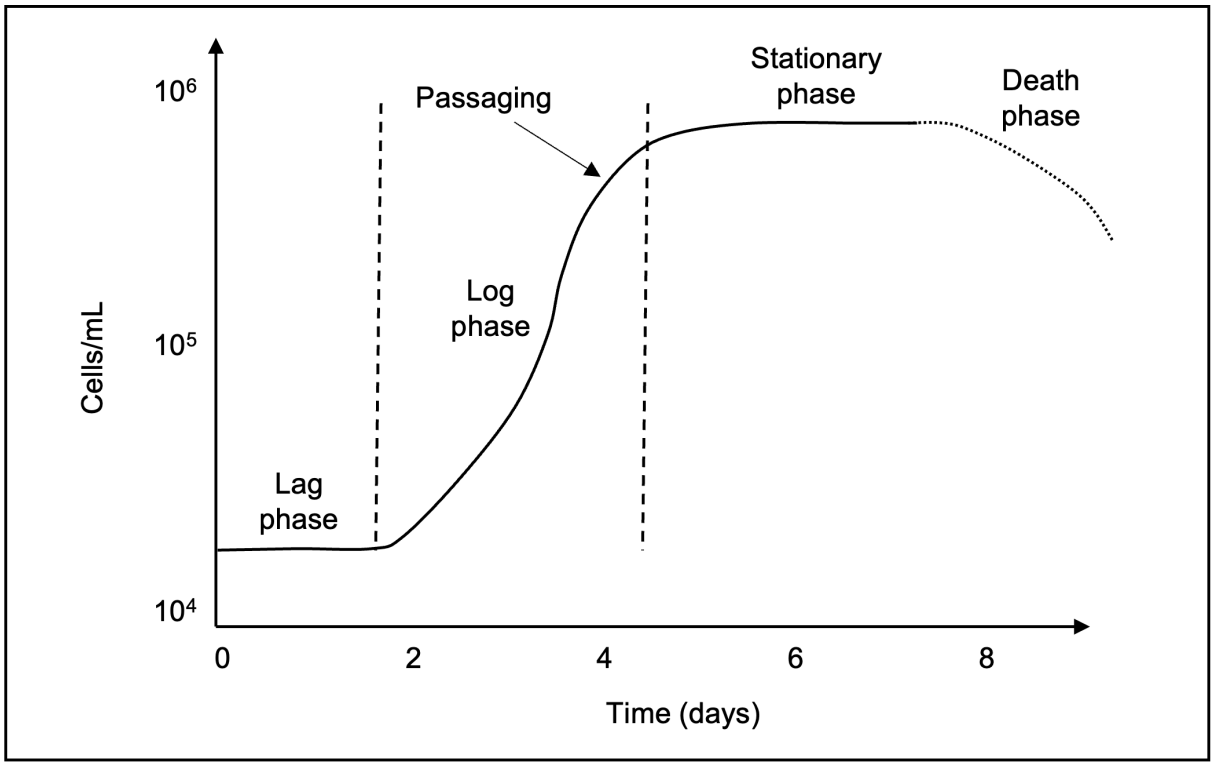


Figure 1.1: Exemplary growth curve of adherent cells (based on [15]).

The manual process for the three procedures starts with getting the selected cell culture flask from the incubator, which has a chamber lined with shelves and equipped with a thermostat, a humidifier, and a carbon dioxide (CO_2) regulator. The cell culture flask is then placed on a microscope to check cell confluency. Cell confluency is typically given as a percentage, which refers to the proportion of the cell culture flask covered by adherent cells. In the case of low confluency, the flask is placed back in the incubator if no media change is planned. If high confluency is detected, passaging of the cell culture is needed.

After, the flask is opened, and the spent cell culture media is removed. A balanced salt washing solution without calcium and magnesium (approximately 3 mL per 10 cm^2 culture surface area)

is added to the flask. After, the flask is moved back and forth several times, and the washing solution is removed. For the procedure of media change, a specific amount of pre-warmed complete growth media is added to the flask, the lid attached, and the flask placed back in the incubator.

For the passaging procedure, a dissociation reagent such as trypsin (approximately 0.5 mL per 10 cm^2) is added. Next, the flask is gently moved to get complete cell layer coverage and secure full detachment. Depending on the used cell line, the cell culture vessel can be incubated for a short time (approximately two minutes). After, the cells are observed for detachment under a microscope. As an optional step, a small number of cells can be removed for cell counting. Pre-warmed growth media is then added to the flask. Depending on the volume and desired cell density, the cells get split into a specific number of new flasks by pipetting. As a final step of the passaging procedure, the lid is attached to the flasks, and the flasks are placed back in the incubator [8, 14]. A simplified process flow diagram for manually performed media change and passaging of adherent cells can be seen in fig. 1.2.

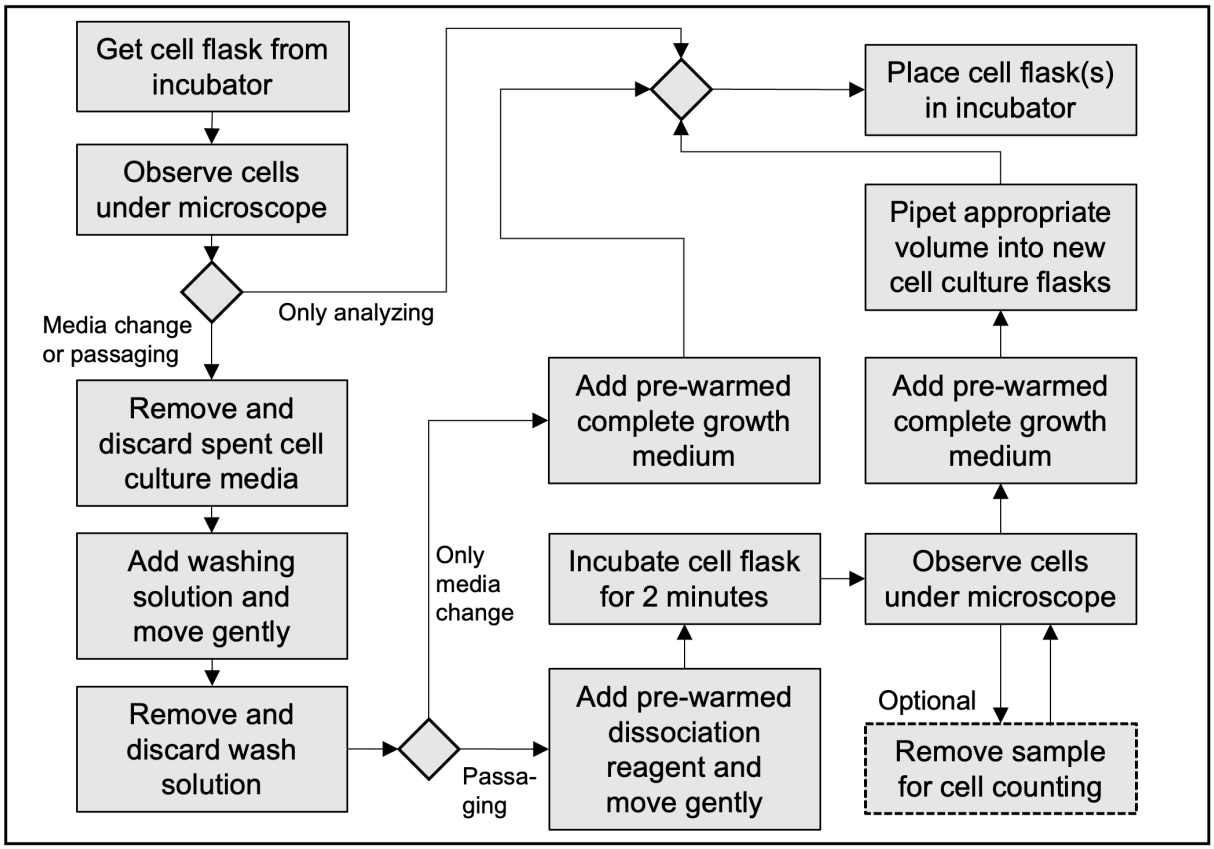


Figure 1.2: Simplified process flow diagram for analyzing cells, media change, and passaging of adherent cells performed manually by laboratory professionals (based on [8]).

The exact execution of the steps can vary depending on the researcher. Pictures of exemplary steps of passaging cells performed manually by laboratory technicians at R&eD of Novo Nordisk can be seen in fig. 1.3. For further visualization of the manual process, a video of the passaging process can be found here in the GitHub repository of the project.

This thesis presents a solution to automate the described manual processes. The main idea is visualized in fig. 1.4 and consists of three main contributions, each taking a significant step towards building a fully autonomous cell culture system for low throughput.

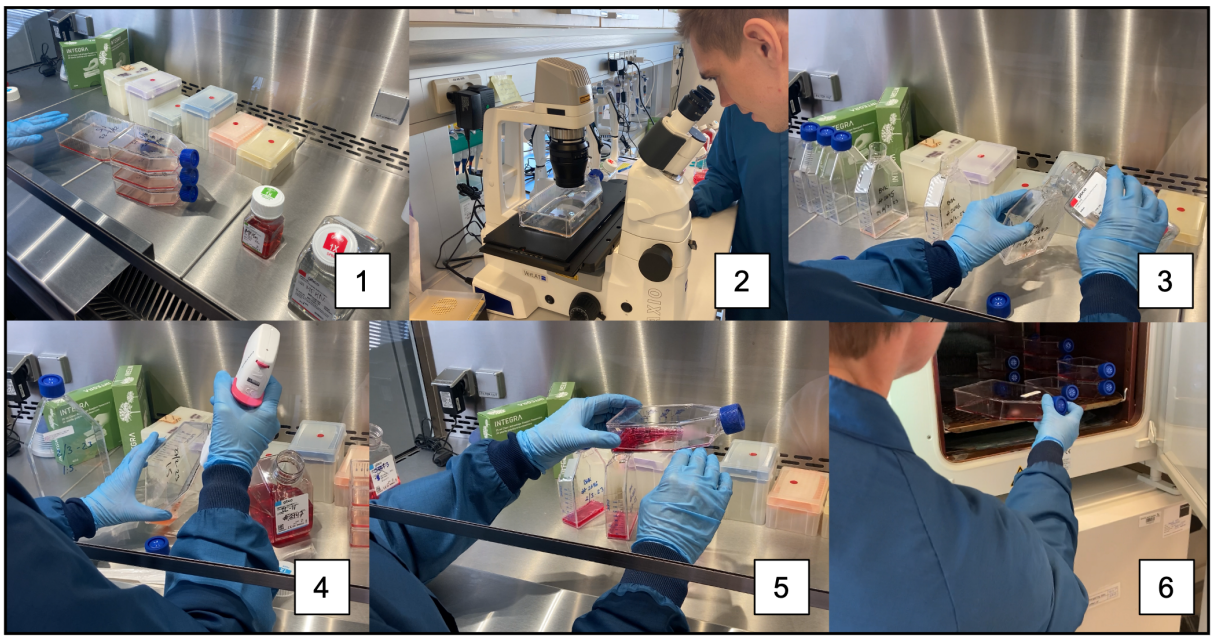


Figure 1.3: Pictures of exemplary steps of passaging performed manually by laboratory technicians at R&eD of Novo Nordisk: (1) Initial setup with cell flasks and materials, (2) Observation of cells under a microscope, (3) Pouring of washing solution, (4) Adding dissociation reagent by pipetting, (5) Gently moving the flask, (6) Removing/Placing of cell flasks from/to an incubator.

- **System prototype for full robotic automation:** The proposed system includes a UR5e collaborative industrial robot with an Intel RealSense D415 camera, a standard incubator that is adapted for automated opening and closing, a microscope, a unit for heating and cooling of liquids, and a capping and decapping unit. All the additional parts are 3D-printed or custom-built. The entire system is made to fit on one table. The robot arm executes the tasks usually done by human labor. The laboratory technician is only responsible for providing empty input flasks and refilling the required liquids.

- **Vision-based liquid and transparent object detection and liquid volume estimation:** A vision-based system is developed to detect transparent objects for process monitoring (e.g., how many flasks are present) and to estimate the volume of liquid in the different transparent containers. The first step to achieve this is a deep-learning approach based on the *TransProteus* dataset [16]. The result of this model is a segmentation and depth estimation of transparent vessels and the liquid inside of them. This model is used to generate a new dataset of images of laboratory containers with liquid content, including their segmentation and depth estimation and the object and liquid volumes. A new model trained on this dataset estimates the volume of liquid given in a transparent container. This estimation builds the base for autonomous robotic pouring from the given containers.

- **Adaptable robotic pouring using fluid simulation:** Instead of pipetting, the autonomous solution is based on robotic pouring. The robot arm movement to pour a desired amount of liquid with varying starting volumes is based on the vision-based estimation of the liquid volume in the pouring container and the results of a simulation of the pouring movements with the particle-based simulator NVIDIA Flex. This pouring simulation can be adapted to different scenarios and objects also outside of research laboratory environments.

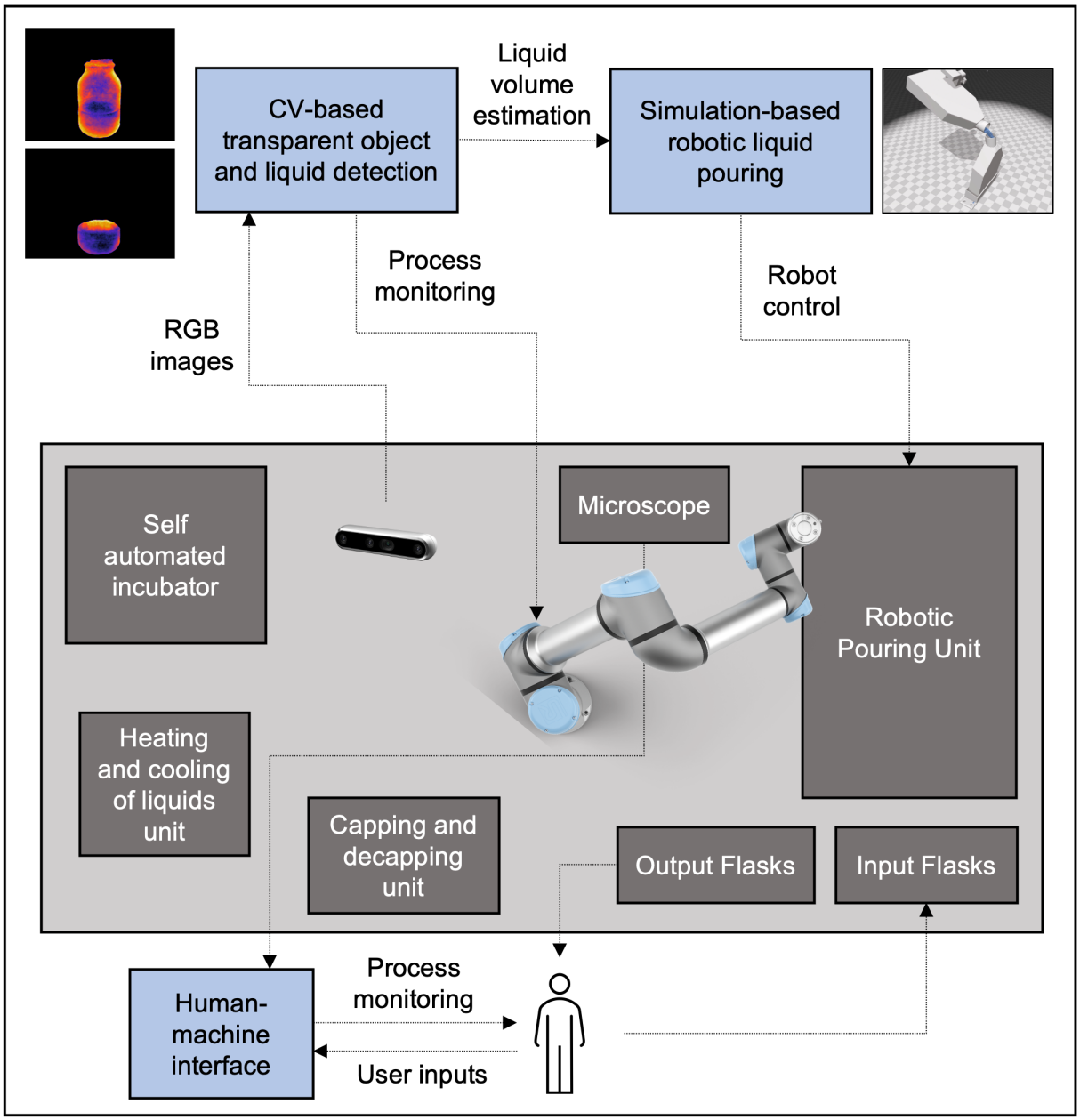


Figure 1.4: Simplified overview of the proposed solution.

1.3 Goals and Research Questions

As mentioned, there is currently no cell culture automation system for low throughput. Also, the research on using computer vision for robotic tasks in research laboratories is still in its beginning, and no approach for vision-based liquid volume estimation in research laboratory environments is available (see section 2.2). That is why the research of this thesis is exploratory, with flexible and open-ended goals to create a basis and provide ideas for further projects and research in this area.

The overall goal is to investigate the opportunities of an automated cell culture system for low throughput. This central idea splits into two subgoals: (1) to integrate existing laboratory hardware for a proof-of-concept prototype and (2) to perform human tasks that include liquids using a camera and a robot arm.

For goal (1), the question of if and how existing laboratory hardware can be combined to create an autonomous cell culture system is to be answered. Desired are a modular, adaptable, and low-cost setup to enhance reusabilty for future laboratory automation projects.

Goal (2) can be divided into two subtopics. First, computer vision techniques are supposed to detect liquids and transparent vessels during the cell culture process. One question to be answered here is how a vision-based system can be used for process monitoring. However, the main focus lies on the question of how accurately the volume of liquid can be estimated based on vision. This focus is because a working and well-generalizing approach for vision-based volume estimation can be used for various tasks in unmanned laboratory systems, including the problem of robotic pouring. Second, based on those results, the pouring process for laboratory containers is to be simulated for autonomous robotic pouring in laboratory environments. The resulting research question is whether a particle-based simulation can predict the required real robot arm movement to pour a specific amount of liquid depending on varying starting volumes. A schematic overview of the goals and research questions can be seen in fig. 1.5

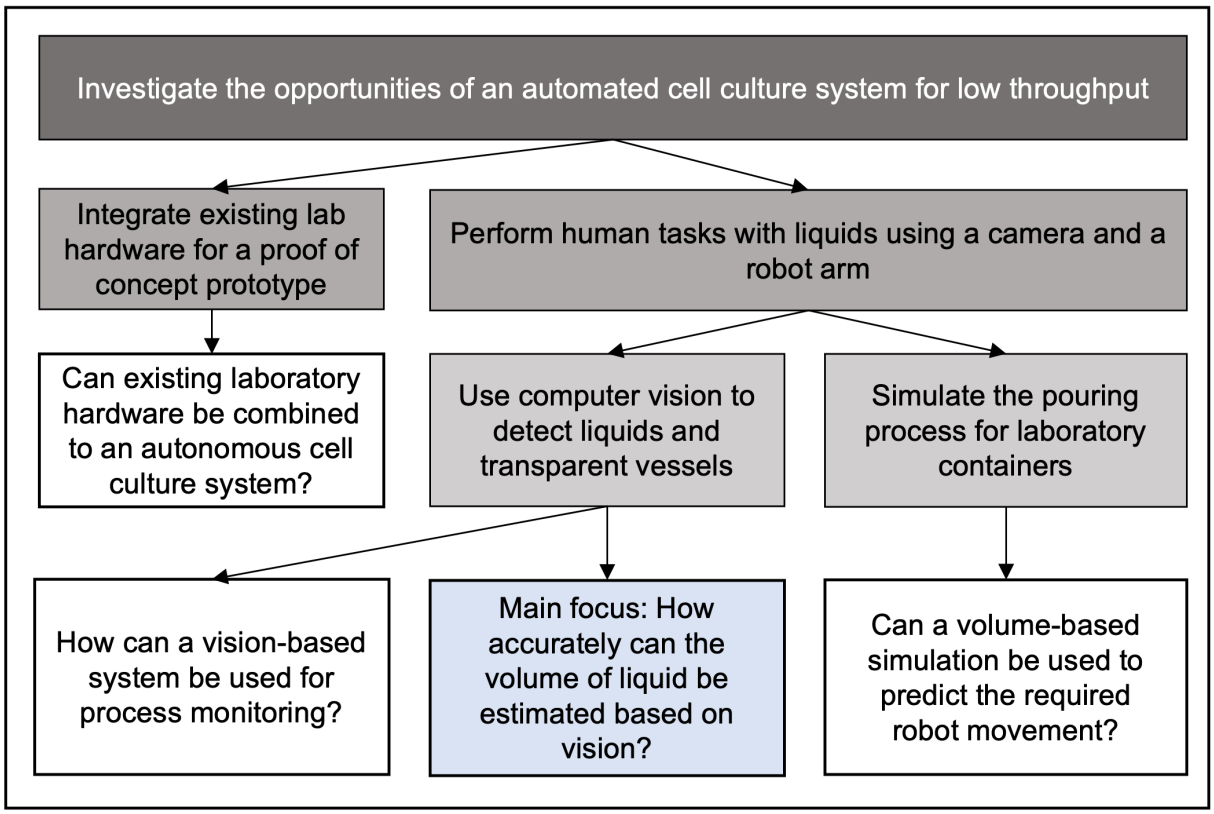


Figure 1.5: Overview of the goals and research questions.

1.4 Structure of the Thesis

This thesis starts by providing the relevant technical background and explaining the relevant work done in the fields of unmanned systems and computer vision in research laboratories, as well as autonomous liquid pouring.

The methodology and procedures chapter is divided into three parts. The first part explains the hardware components, the connection and configuration, and the process and workflows of the system prototype. The second part provides details about the methodology used for the vision-based detection and volume estimation of liquids. This includes data generation, model architectures, and experimental setups. The third part explains the process behind the

approach for simulation-based autonomous pouring from laboratory vessels.

Building on the methodology and procedures chapter, the results chapter is also divided into three sections. First, the liquid detection and volume estimation results are presented, followed by the results of the simulation-based autonomous pouring. Finally, the overall performance of the system is evaluated.

The thesis finishes with a discussion about the results, possible future work, and a short conclusion.

2 Technical Background and Related Work

This chapter will provide the technical background and related work surrounding the three relevant topics. First, it is aimed to provide a comprehensive understanding of unmanned systems in research laboratories (section 2.1) and especially in the field of cell culture automation (subsection 2.1.4). After, general background in computer vision technologies and deep learning will be given (subsection 2.2.1), followed by an overview of the challenges of computer vision in research laboratories as well as the previous work done in liquid detection and volume estimation (subsection 2.2.2). Finally, the current research and boundaries of autonomous liquid pouring will be explained (section 2.3).

2.1 Unmanned Systems in Research Laboratories

2.1.1 Definition and Classification of Unmanned Systems in Research Laboratories

Unmanned systems in research laboratories are robotic systems that are designed to perform various experimental operations without the need for human intervention. They are used in various laboratory applications, including sample preparation, analysis, and processing. These systems can perform tasks such as weighing, mixing, and dispensing reagents and samples, as well as data collection and analysis. Recent developments in unmanned systems for laboratory automation include the integration of robotics and artificial intelligence (AI). Robotics can automate sample handling, while AI can optimize reactions by adjusting experimental conditions in real time based on sensor data recorded during the process (closed-loop approach). Combining machine learning methods and research experiments makes it possible to narrow the gap between the infinite parameter space and the finite experimental capacity [17]. Recent laboratory systems that combine full hardware autonomy with data-based recommendations are known under the term self-driving lab (SDL) [18, 19]. The transition from conventional laboratories, where humans execute manual tasks, to unmanned laboratories, where the researchers are primarily responsible for interpreting the results and solving high-level questions, is visualized in fig. 2.1.

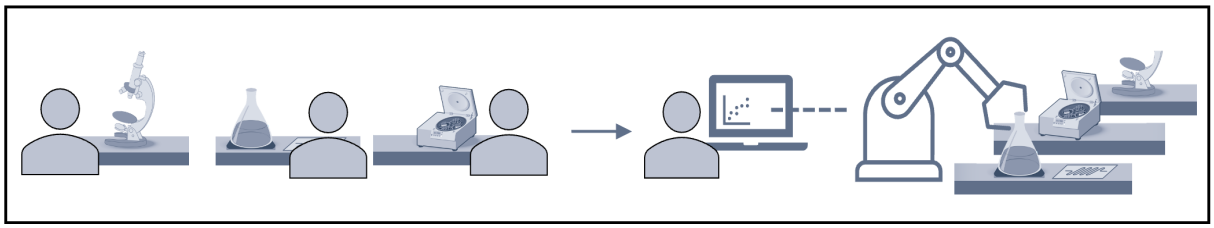


Figure 2.1: Illustration of the transition from a conventional research laboratory (left) to an unmanned laboratory (right) (based on [19]).

To the author's knowledge, a classification system for research laboratory automation equipment has not been published yet. However, several comparable methods have been devised for classifying industrial automation. In a study conducted by Frohm et al. [20], these existing systems were examined, and the authors suggested their own framework consisting of 7 levels of automation. Table 2.1 presents these levels, corresponding descriptions, examples commonly found in research laboratories, and approximate costs.

It demonstrates that most equipment items deemed the most expensive in most present laboratories are classified under level 5. Finding higher-grade 6 and 7 items is uncommon in

Table 2.1: Automation levels from Frohm et al. [20] with example laboratory automation equipment including an indicative cost range (based on [2]).

Level of automation	Description	Research laboratory examples	Indicative costs
1	Totally manual - Totally manual work, no tools are used, only the users own muscle power.	Washing of glasses, pouring of liquids	€0
2	Static hand tool - Manual work with support of static tool. E.g. Screwdriver	Dissection scalpel	€10-30
3	Flexible hand tool - Manual work with support of flexible tool. E.g. Adjustable spanner	Pipetting	€50-250
4	Automated hand tool - Manual work with support of automated tool. E.g. Hydraulic bolt driver	Stripette and handheld dispenser	€250-400
5	Static machine/workstation - Automatic work by machine that is designed for a specific task. E.g. Lathe	Centrifuge, automated freezer, capper/decapper	€500-80,000
6	Flexible machine/workstation - Automatic work by machine that can be reconfigured for different tasks. E.g. CNC-machine	Motorized stage microscope, liquid handler	€80,000-150,000
7	Totally automatic - Totally automatic work, the machine solves all deviations or problems that occur by itself. E.g. Autonomous systems	Automated cell culture system, bespoke laboratory equipment e.g., Labman formulation engine.	€100,000-5,000,000

academic research laboratories, but those systems are increasing rapidly in industrial environments. While mid-range level 5 automation items undoubtedly increase the efficiency of laboratory research, they are designed for specific subtasks and require a large amount of manual manipulation before and after machine usage. Most of the equipment in this category performs subtasks that humans would be incapable of doing (e.g., centrifuging). Level 6 and 7 systems and devices often combine lower-level automation equipment to replace the manual tasks in between [2]. The recent development of unmanned systems from level 7 is explained in subsection 2.1.2. The system prototype proposed in this work can also be classified as a level 7 since it requires only minimal human input and can adapt to the execution of steps itself.

2.1.2 Implementations and Applications of Unmanned Systems in Research Laboratories

One of the first works on unmanned systems in research laboratories was done in 2009. Adam [21] is a fully automated system that uses decision trees and random forests to study the function of genes and enzymes. It is equipped with sophisticated software and hardware to perform various microbiological experiments. In biomedicine, Adam has been used to discover new scientific knowledge related to gene functions and drug targets. Eve [22] is a autonomous

discovery system focusing on drug screening. It is designed to be flexible and can perform different bio-activity analyses. Eve uses machine learning algorithms to predict the biological activity of potential drug candidates. In 2020, MacLeod et al. introduced a flexible and modular self-driving robotic platform called Ada, capable of autonomously synthesizing, processing, and characterizing organic thin films. The configuration of the robotic platform for a specific experimental workflow is achieved by mounting an appropriate collection of experimental modules on the robot [23]. ARChemist, an automated laboratory system proposed by Fakhruldeen et al. in 2022, was developed to carry out experiments such as solubility screening and crystallization without human intervention [24]. One of the main challenges for unmanned systems in laboratory automation is the integration of various components, including sensors, actuators, consumables, and control systems, to enable seamless operation [17]. Most developed systems focus on integrating existing laboratory equipment and combining it with autonomous robotic solutions. One main example of this approach was accomplished by Burger et al. in 2020. The authors used a mobile robot to create an unmanned system to accelerate the search for improved photocatalysts for hydrogen production from water. The strategy focuses on automating the researcher rather than the existing instruments. The mobile robot can autonomously navigate in a laboratory and perform reagent-dispensing and handling operations at different experimental stations [25].

Critically, previous robot-assisted laboratory automation systems were tested in carefully controlled environments to prevent the occurrence of unsatisfied task constraints. Additionally, existing setups cannot incorporate perception for creating scene descriptions, which restricts experiment setups to static and structured environments. The dependency on predefined tasks and motion plans imposes limitations on the adaptability and robustness of those systems to new environments.

2.1.3 Components and Design Considerations for Unmanned Systems in Research Laboratories

The complexity of an unmanned system for a research laboratory is directly related to the variety of the required experimental modules (e.g., single versus multistage experimental stages), the range and number of operating process conditions (e.g., pressure and temperature), type of solvent (aqueous versus organic), required characterization techniques and acceptable precision. Since the complexity of an autonomous system is usually directly correlated to development time and costs, reusability through standardization and modularization of hardware and software is a crucial aspect of the design of future unmanned systems [19]. Designing for flexibility is also an essential factor for laboratories with a high level of protocol variation. In addition, the ability to modify an autonomous system without requiring specialized engineering knowledge is highly desirable. This enables researchers to automate a broader range of process steps without requiring time-consuming and costly redesign of tools or extensive reprogramming. An intriguing advancement of the modular design approach is integrating existing automation equipment into a unified system capable of seamlessly executing the desired protocol in a continuous process stream [2]. Additionally, laboratory automation systems built upon an anthropomorphic design that mimics human movements are more likely to be trusted by laboratory operators [26].

The robot is considered the most critical component in designing an unmanned system for research laboratories. When analyzing successful systems for laboratory automation, three approaches to integrating robots for transferring materials and devices become clear: mobile robots (usually including a moveable arm), stationary robot arms, and fluidic robots. Examples can be seen in fig. 2.2.

The right approach heavily depends on the characterizations of the system. Mobile robots have a distinct advantage in research labs due to their easy access to existing large equipment. Mod-

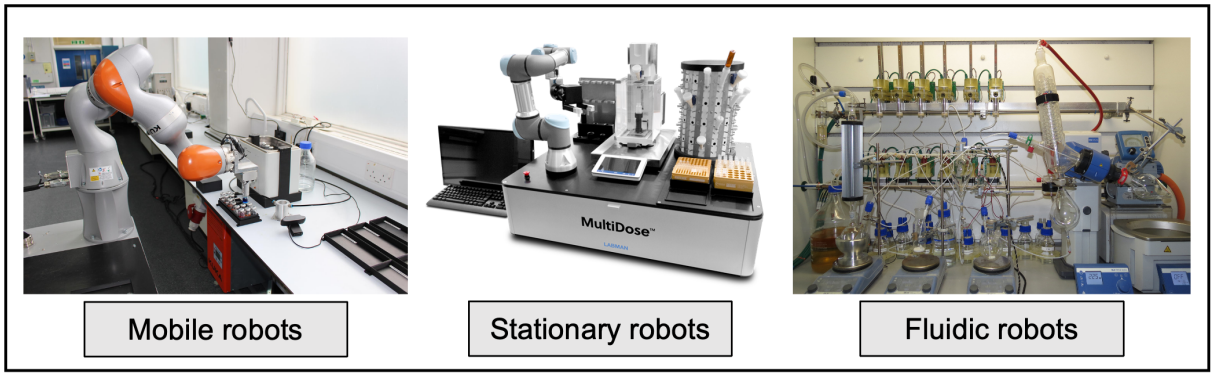


Figure 2.2: Pictures of examples of mobile, stationary, and fluidic robots used to create unmanned laboratory systems. Left: KUKA mobile robot used for the mobile chemist [25], Center: Automated dosing system using a stationary UR5 arm [27], Right: The Chemputer platform using a fluidic robot [28].

ularizing process steps and sharing the workplace with laboratory staff is possible. Examples of mobile robots used in laboratory automation are the KUKA mobile robot used for the mobile chemist by Burger et al. [25] and Kevin, a mobile robot designed by Fraunhofer IPA for life science laboratory tasks [29]. Although Kevin was developed to fulfill non-value-adding activities in the laboratory, it could combine different submodules of a self-driven laboratory. However, due to mobile robots' high initial cost and setup time, they are not commonly used in research laboratories yet [19].

Stationary robots are usually cheaper and can move faster in a protected environment than mobile robots. The main downside is the limited range of movement, which limits the number of reachable laboratory devices. An often-used solution is putting the robot arm on a linear rail. Examples of the use of stationary robots for unmanned systems in laboratory automation can be found in academic research projects [30] as well as in serial industrial products [27].

Another crucial aspect of the system design is the selection of consumables such as flasks, plates, and pipettes. The effectiveness and robustness of autonomous laboratory systems are maximized when input materials or consumables are standardized. In the case of standard-shaped labware, this enables non-adaptive, rigid automation components like grippers to have complete control over the device, resulting in improved placement accuracy and potentially faster operations. However, there is a significant amount of variation in labware not only between different research laboratories but even within the same laboratory. Researchers often switch between different variants of labware based on cost, availability, or personal preference, which introduces challenges for automation processes [2]. Robotic perception using vision modules can help to improve the system's flexibility to handle different input consumables (see subsection 2.2.2). Furthermore, automation-friendly labware can simplify tasks such as decapping containers.

The selection of additional devices depends on the specific tasks of the system. The emergence of cost-effective 3D printing technologies for laboratory applications [31], off-the-shelf actuators, and programmable microcontrollers [32] has empowered research laboratories to combine components that can be easily assembled, controlled, and automated at a relatively low cost for prototyping and final solutions. This technological advancement allows researchers to customize their existing equipment and create new elements without significant expenses [2].

2.1.4 Cell Culture Automation

In the field of cell culture automation, a wide range of solutions has emerged to improve laboratory work and streamline processes. These systems encompass a range of components and technologies, including robotic arrangements for labware handling, sterile hoods with filters to maintain aseptic conditions, liquid handling systems, cell count devices, and incubators set at 37 °C with a controlled CO_2 atmosphere. Additionally, advanced systems may incorporate devices for real-time detection of biological processes and microscopes for precise evaluation of cell confluency [33, 34].

The design and functionality of these systems are aimed toward monitorability, typically facilitated by specific software. The existing systems in this field can be categorized based on the type of labware they are designed for, such as microplates or cell culture flasks. Most of the systems are built with an emphasis on large-scale studies that would otherwise be difficult to undertake (e.g., Formulatrix [10], StemCellFactory [9]). Solutions were proposed both by companies in the laboratory automation sector and as academic work. A non-exhaustive list of cell culture systems can be found in Table 2.2.

Table 2.2: Non-exhaustive list of cell culture systems both commercially available as well as built as an academic project (sorted in ascending order by year of publication).

Cell Culture System	**Type of Labware**	**Applications**	**Throughput**
Cellmate [35]	Flasks	Seeding and harvesting of adherent cells	Medium/High
Cellerity/Freedom [36] EVO	Plates	Adherent cell cultivation and transfection to produce HIV pseudovirus; cell handling	High
BioCel Systems [37]	Plates	Cell expansion and cell biology (high-throughput screening)	High
Cell Culture System [38]	Plates	Cultivation of adherent cells from gingival tissue and bone marrow aspirate	Low
AI.CELLHOST [39]	Plates	Cultivation of adherent cells (SH-SY5Y, BE(2) M17, and HEK293T cell lines)	High
Biomek Cell Workstation [34]	Plates	Cultivation of adherent and suspension cells	Medium
CELP [40]	Flasks	Cultivation of adherent and non-adherent cells	Medium
AUTOSTEM [41]	Plates	Large scale production of clinical-grade mesenchymal stromal cells	High
StemCellFactory [9]	Plates	Automated generation and expansion of human induced pluripotent stem cells	High
CompacTSelecT [42] S.C.	Plates/ Flasks	Regular cell handling	High
Formulatrix [10]	Plates	Large scale cultivation of adherent cell lines	High

Among the notable examples is the CompacTSelecT system, which features a six-axis anthropomorphic robotic arm enabling access to 90 T175 flasks. It also includes a plate incubator, along with integrated components such as a cell counter, a medium pump, and flask cappers/decappers and holders. To maintain a sterile environment, a HEPA filter is employed in the system [42].

Researchers have also investigated the AI.CELLHOST system, which combines automated cell cultivation with fluorescence imaging. The system incorporates the Hamilton Microlab STAR for precise liquid handling, a cell counter, and a Cytomat 24C incubator for accommodating up to 504 SBS plate stackers in a sterile environment. Sterile conditions are maintained by a flow chamber equipped with HEPA filter systems. A MICROLAB SWAP robot arm achieves the connection between the modules [39].

TECAN offers the Cellerity system designed for high-throughput automated cell culturing. This system provides various options for cell culture processes. It includes a housing with a HEPA filter system to ensure sterile handling of samples and reagents [36].

The BioCel system by Agilent Automation incorporates a fast central radial robotic arm for high-throughput processes. The system provides an array of environmental support solutions, including filter systems with ultra-low penetration air filters, unidirectional airflow, and precise control of temperature and humidity [37].

Another notable system is the compact cell culture system developed by Kato et al., specifically designed for the cell expansion of primary tissue, particularly fibroblasts. This system consists of different units, including a supply unit, an incubation unit, and a collection unit. Rotary pumps and valves are incorporated into the system, which is compact (70 × 60 × 86 cm). The required solutions are contained in bags within the supply unit. The system utilizes disposable tubing sets that connect the central units, including the culture dish, aeration filters, and collection tubes [38].

The Biomek Cell Workstation is another noteworthy system that offers compatibility and flexibility for the automated cultivation of various cell types. The workstation supports both two- and three-dimensional cell cultures. Initially designed for adherent cell cultures, the Biomek Cell Workstation has also been enhanced to handle suspension cells. The system includes a Biomek NX for precise liquid handling steps, an incubator, a centrifuge for sample preparation, and a cell counter. Notably, the vertical arrangement of the system allows for a compact footprint, distinguishing it from other existing systems [34].

As shown in the table 2.1, most systems are designed for high throughput and are, therefore, expensive and unsuitable for small-scale laboratory automation, both for small laboratories and small cell cultures. Low throughput systems found during the research are either made for a specialized use case [38] or not commercially available [40].

2.2 Computer Vision in Research Laboratories

The journey of computer vision technologies started in the 1960s. At the initial stage, computer vision illustrates the extraction of information through digital images using computational models [43]. In recent years, there have been significant advances in computer vision technologies, in particular deep learning techniques and convolutional neural networks, enabling more effective and accurate visual recognition and analysis (see subsection 2.2.1). Computer vision is a technology suitable for acquiring, processing, and analyzing visual inputs and, therefore, is and will increasingly be an essential aspect of modern laboratory automation. In unmanned systems in research laboratories, visual recognition involves object identification, position estimation, and material detection as the prerequisites for robotic manipulation tasks. Ideally, a

computer vision system for a research laboratory would be able to replicate a human scientist's observation and analytics skills. While computer vision has been extensively researched and applied in other areas, such as autonomous vehicles and surveillance, its application for robotic tasks in experimental research laboratories is relatively new and limited.

2.2.1 Background on Deep Learning in Computer Vision

AI is a field of computer science whose goal is to create algorithms that mimic human intelligence to solve problems and automate decision-making. Machine learning (ML), a subfield of AI, is widely used in research and various applications. These include text mining, spam detection, video recommendation, and image classification. Deep learning (DL), which is a subset of ML (see fig. 2.3), draws inspiration from the information-processing patterns observed in the human brain [44].

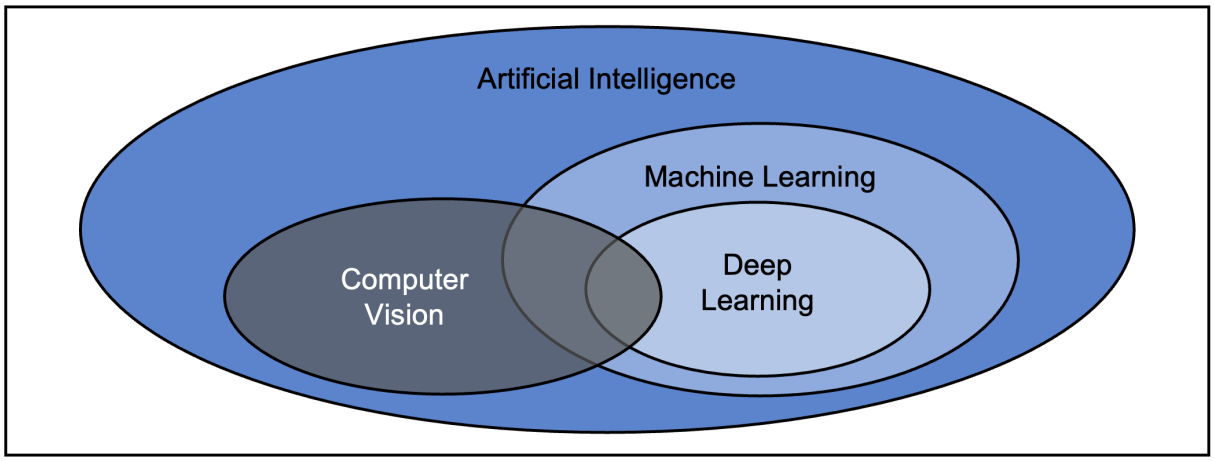


Figure 2.3: Venn diagram of artificial intelligence, machine learning, deep learning, and computer vision (based on [44]).

Unlike traditional rule-based systems, deep learning does not rely on human-designed rules for its operation. Instead, it leverages vast quantities of data to establish mappings between input and corresponding labels. DL uses multiple layers of algorithms known as artificial neural networks (ANNs). Each layer offers a distinct interpretation of the input data it is fed with. For example, to perform a classification task, traditional ML methods involve a series of sequential steps, including preprocessing, feature extraction, careful feature selection, learning, and classification. The effectiveness of those ML techniques depends heavily on accurate feature selection, as biased selection can lead to incorrect class differentiation. In contrast, DL offers the advantage of automating the learning process for feature sets for different tasks. DL facilitates simultaneous learning and classification, eliminating the need for separate steps [45].

Deep learning techniques can be broadly categorized into three main categories: unsupervised, semi-supervised, and supervised learning. Since only labeled data is used in this work, the focus lies on deep supervised learning. In supervised learning, the environment consists of a set of inputs and their resultant outputs ($(x_t, y_t) \sim \rho$) [45].

Convolutional neural network (CNN) is one of the most popular and used deep learning networks, which is especially the case for computer vision applications. The structure of CNNs takes inspiration from the neurons found in human and animal brains, similar to conventional neural networks. Specifically, the visual cortex in a cat's brain, which consists of a complex arrangement of cells, serves as a simulation model for CNNs [46]. The main advantage of CNNs compared to other network types is that it automatically detects significant features without human supervision. The benefits of using CNNs over other traditional neural networks in

the computer vision environment can be summarized as follows [47, 45, 43]:

- The weight-sharing feature reduces the number of trainable network parameters and helps the network to enhance generalization and avoid overfitting.
- When feature extraction layers and the final layer are learned concurrently, it leads to a model output that is well-structured and heavily dependent on the extracted features.
- CNNs offer greater simplicity in implementing large-scale networks than other types of neural networks.

The CNN architecture comprises multiple layers, also known as building blocks. The following section briefly describes each layer in the CNN architecture and its respective function [45, 47]. These will be used later on during this work.

- Convolutional Layer: This layer acts as a primary building block, determining the output by computing dot products between input and convolutional filter (kernel) values, creating a 2D activation map. Through this process, a CNN efficiently learns filters that activate when specific features are observed in the input.
- Pooling Layer: It performs sub-sampling of feature maps, reducing their size while preserving important information. Different types of pooling methods, such as max pooling, min pooling, and global average pooling, can be used. However, the pooling layer's drawback is that it focuses on the location of features rather than capturing all relevant information, potentially reducing the overall CNN performance. Nonetheless, pooling layers help scale down the spatial size of representations, speeding up calculations and preventing overfitting.
- Activation Function (Non-linearity Layer): They map the input to the output. This is achieved by computing the weighted sum of neuron inputs, including bias if applicable, and determining whether the neuron should fire based on the input, generating the corresponding output.
- Fully Connected Layer: This layer is located at the end of each CNN architecture, and connects each neuron to all neurons of the previous layer. It takes a vector input obtained by flattening the feature maps and produces the final output of the CNN.
- Loss Function (Loss Layer): The loss layer plays a crucial role in minimizing the difference between true and predicted values/labels during training, serving as a guide for the neural network's learning process. Various loss functions can be used for different tasks, which will be explained in more detail in the following sections.

Deep learning-based methods for object segmentation and monocular depth estimation are particularly important for the course of the work, which is why they will be briefly discussed.

Semantic Segmentation

CNN-based segmentation has found applications in various domains, including medical imaging, object detection, and autonomous driving. The aim of segmentation is to find distinct and meaningful regions in the picture. Other than in an image classification task, the last layer of the neural network architecture is typically designed to produce a dense output map that represents the segmentation mask for pixels or regions in the input image. For the segmentation of a single object, the segmentation mask is usually predicted as a two-layer probability mask (pixel belongs/does not belong to the object). A fully convolutional network (FCN) is a specific type of CNN architecture designed for pixel-wise predictions, particularly in tasks like semantic segmentation. FCNs are tailored to preserve spatial information and enable end-to-end pixel-level predictions [48, 49].

Monocular Depth Estimation

Estimating scene depth is a crucial aspect of computer vision that enhances perception and understanding of three-dimensional environments in the real world. This capability has significant implications for several fields, including robot navigation, autonomous driving, and virtual reality, as it enables more accurate spatial understanding. Active depth estimation methods often rely on lasers, structured light, and reflections to capture depth information from object surfaces, resulting in depth point clouds, surface models, and scene depth maps [50]. However, these methods often demand significant computing resources and fail for transparent objects given in a research laboratory (see subsection 2.2.2). As a result, image-based depth estimation has gained popularity as a more practical alternative, offering efficient and cost-effective solutions that can be applied in various applications. Monocular depth estimation based on deep learning is a task of learning depth maps from a single two-dimensional color image using a deep neural network, which was first presented by Eigen et al. in 2014 [51]. Since then, many researchers have developed DL methods for monocular depth estimation. Modern approaches are usually based on an encoder-decoder network. The encoder consists of a combination of convolution and pooling layers to capture the depth features, while the decoder network includes deconvolution layers to rebuild the estimated pixel-level depth map with the same size as the input. Additionally, to preserve the features of each scale, the corresponding layers of the encoder and decoder are concatenated with skip connections. With known camera parameters, it is possible to convert the predicted depth map into an XYZ map with the 3D position of each pixel in the world [50, 52].

2.2.2 Computer Vision Challenges in the Research Laboratories

The diverse and complex environments of research laboratories present a challenging scenario for computer-vision-based systems, as a variety of instruments and equipment may become targets for recognition with different distances, locations, and sizes. At the same time, images may vary in brightness, darkness, and mutual occlusion [17]. Especially transparent plastic and glass objects, which are almost always present during research experiments, are challenging. They often appear as noisy or distorted approximations of the surfaces that lie behind them and do not form disjoint boundaries with their environment. Although transparent objects have unique characteristics based on the material and thickness, there has been little research on applying DL methods to this area due to the difficulties in gathering and labeling a sufficiently large dataset [53]. Consequently, segmentation or classification approaches often fail for transparent vessels.

Additionally, experimental research laboratories often lack standardized protocols for capturing images and data, leading to variations in image quality, lighting conditions, and experimental setups. These variations pose challenges for developing robust computer vision algorithms that can work reliably across different labs and experiments. Tackling that problem using a DL approach requires large amounts of annotated data. However, annotating laboratory images can be time-consuming and resource-intensive, often requiring domain expertise and manual labeling. Building comprehensive and accurately labeled datasets can be a significant challenge [17, 3].

To mimic the observation and analysis skills of a human scientist, the vision system must also be able to recognize and analyze the samples used in the experiments. Laboratory samples can have complex and diverse structures, with a high variety in the appearance of liquids and mixtures. These variations make it challenging to develop computer vision algorithms that can accurately analyze different types of samples, which is crucial for autonomous process monitoring and task execution [3, 17]. In the following, the main tasks approached in this work will be explained in more detail, and the current state-of-the-art will be shown.

Transparent Object Detection

In unmanned systems, visual recognition involves identifying and detecting target objects. Detection is a prerequisite for robotic manipulation tasks and process monitoring. Research laboratories often involve dynamic experimental setups, where the arrangement of equipment, containers, and samples may change over time. The computer vision system should be able to adapt to these changes and update the planning of the movement of the robot accordingly. Developing efficient and practical computer vision-based object recognition and detection systems for autonomous laboratory systems remains a significant challenge. While current computer vision technology has improved experiment efficiency, most existing unmanned systems in research laboratories rely on multiple dedicated automation devices, leading to high costs and limited versatility. Most current approaches are still limited to simple conditions with controlled environments and often fail in complex real-world scenarios [16, 17].

The main difference between an object detection system in the research laboratory and a conventional object detection system is the frequency of transparent objects. As described, transparent objects have unique visual properties that make them extremely difficult to perceive. Since they do not adhere to the geometric light assumptions used in classic vision algorithms, deep learning approaches are the current state-of-the-art in transparent object detection. One recent DL-based approach for estimating accurate 3D geometry of transparent objects from a single RGB-D image that could be applied to robotic manipulation in laboratory environments was presented by Sajjan et al. [54]. The CNN was trained on a large-scale synthetic dataset of over 50,000 RGB-D images. The work also demonstrates that the vision system can be applied out-of-the-box to improve grasping for transparent objects algorithms' performance on transparent objects. A similar dataset, which supports caustics, dispersion, and refraction and therefore shows better results, was created by Mousavi et al. [53]. However, both datasets only contain images of empty containers, which is rarely the case in experimental research environments. The primary studies about transparent object detection, specifically in laboratory environments, were conducted by Sagi Eppel et al. and the matter lab at the University of Toronto [55, 56, 57, 16]. In [16], the authors achieved a mean Intersection-over-Union (mIoU) of 87% for transparent vessel segmentation on real images. All these approaches also include the detection of materials and liquids in particular and will therefore be described in the next section.

Liquid Detection

Handling liquids inside mainly transparent containers is one of the main activities performed by human scientists in the laboratory and is, therefore, a crucial task to be solved to achieve completely autonomous laboratory systems. Compared to multiple sensors like pressure sensors, laser distance sensors, or millimeter-wave radars, camera-based vision solutions have the potential to be the most suitable option in terms of both flexibility and economy [17]. There have been attempts for general liquid detection and some specifically in research laboratory environments. An approach based on conventional image processing techniques by Eppel and Kachman in 2014 achieved a miss rate of less than 1% for gas-liquid surfaces [55]. Since the solution is based on edge detection for vessel and liquid level recognition, it requires standardized image-taking conditions with plain backgrounds and fails in more complex environments. Another early method to detect liquids using computer vision was presented by Hara et al. in 2014 [58]. The authors used data from a depth camera and based the detection on light refraction and triangulation principles. The main limitations are the necessity to acquire depth data and that the method only works for the liquid surface visible from the top. Elbrechter et al. introduced a classification system for discriminating liquids in different containers used in kitchens using point cloud data. However, the approach fails for previously unseen liquids due to the lack of a large training dataset and does not provide any information about the liquid quantity. A graph-cut computer vision approach for tracing liquid levels and material boundaries in transparent vessels was proposed by Eppel et al. in 2016 [56]. The method gave high accuracy in

boundary recognition for liquid, solid, granular, and powder materials in various glass containers in chemical environments. Since it relies on the assumption that the bottom of the vessel is completely covered by material and the top is empty, the method fails for small quantities of materials. Schenck and Fox [59] used an LSTM-based neural network trained on a synthetic dataset to track liquids in video sequences. The study demonstrates that aggregating data over multiple frames yields the best liquid detection results compared to standard image segmentation. However, the work focuses mainly on the liquid in motion. Therefore, no conclusions about liquids inside transparent containers are possible.

In 2020, Tara Zepel et al. [60] created an inexpensive and adaptable computer vision system to monitor and control liquid levels in a chemical reactor. Since the liquid-level detection algorithm is based on canny edge detection specifically improved for that reactor, it is unlikely to perform well in other, more general environments. During the same year, the first ML-based approach that specifically targeted research laboratory environments was presented by Eppel et al. [57]. They provided a new dataset, *Vector-LabPics*, which consists of 2,187 annotated images for identifying materials and containers in laboratory environments. The CNN trained on this dataset showed satisfactory accuracy in identifying and separating vessels and material phases and classifying liquids and solids. However, the accuracy was relatively lower regarding segmenting systems with multiple phases, such as liquids undergoing phase separation. The major limitation in increasing the accuracy of the models is the relatively small size of the dataset.

That problem was tackled again by Eppel et al. in 2022 when they introduced a new procedurally generated dataset consisting of 50,000 images of liquids and solid materials inside transparent containers. To create the computer-generated images (CGI), 13,000 distinct objects, 500 environments, and 1,450 material textures were combined with simulated liquids and generated vessels.

The trained CNNs aim to predict the 3D shape and properties of materials, liquids, and objects and the shape of transparent containers. The trained CNN for semantic segmentation achieved a mIoU of 55% for content segmentation on real images. The major challenge described by the authors is increasing the prediction accuracy in more complex chemical environments [16]. Since this dataset can be considered state-of-the-art for liquid and vessel images with depth and segmentation ground truth in terms of variety and size, it is also used in this thesis.

Liquid Volume Estimation

Very little research can be found to estimate the volume of liquid and the container itself. To the author's knowledge, there is currently no existing approach for estimating volumes based on computer vision in research laboratory environments.

A first approach to estimating the volume of transparent containers and their liquid content based on an RGB image was proposed by Mottaghi et al. in 2017 [61]. The CNN trained on the data recorded by the authors classified the container volume into one of ten classes (50, 100, 200, 300, 500, 750, 1000, 2000, 3000 mL, and above) and the liquid volume into one of five classes (0%, 33%, 50%, 66%, 100% of the container volume). Average per-class accuracies of 17.79% for the container volume estimation and 32.01% for the content volume estimation were achieved. In 2022, Zhu et al. [62] introduced a model for real-time volume estimation of liquids based on the inputs from an RGB image and a customized tactile sensor. The multimodal CNN achieves a high precision with an error of 2 mL for volumes of 40-80 mL. The main limitation of this system is the low flexibility since it was only trained for one object containing the liquid. Therefore generalization for complex research laboratory environments with different containers is not possible.

An approach to estimating the liquid volume from single-view RGB images was presented by

Cobo et al. in 2022 [63]. The authors aimed to determine the wine volume in nine different glass containers. The CNN was trained on a newly introduced dataset which consists of more than 24,000 images of glasses filled with red wine. A mean average error lower than 10 mL for volumes of 50-300 mL was achieved. However, the model is limited to using glass containers and liquids similar to the ones used in the dataset and was not tested on other settings or more diverse data.

2.3 Autonomous Liquid Pouring

To be able to create a fully autonomous and flexible laboratory system, the robot must be able to pour precisely, similar to a human scientist. To the author's knowledge, there is currently no approach focussing specifically on the pouring of liquids in laboratory environments, especially using vessels present in a research laboratory. However, various approaches to the general problem of liquid pouring, often focussing on kitchen assistant tasks, can be transferred to laboratory environments. The pouring problem can be divided into three main components: observing the fluid in the pouring and/or target container, modeling the flow dynamics, and controlling the pouring container to reach a specified volume [64]. Possibilities of observing the fluid were described in subsection 2.2.2. One central question is whether observing the liquid in the pouring container or the target container is better. Previous work done in the field used different approaches, which will be described briefly.

Burgard and Do [65] focus on a vision-based height detection in the receiving container and then using a PID controller to execute the movement of the wrist joint of the robot. Narasimhan et al. [66] also detect the fill level in the target container and use a generative model that is capable of translating images of colored liquids into synthetically generated transparent liquid images, trained only on unpaired data of colored and transparent liquid images. Schenk and Fox [67] use a deep learning-based approach to estimate the amount of liquid in an opaque target container and combine it with a simple PID controller to pour specific amounts of liquid. The ground truth data for the pixel-wise liquid classification and subsequent volume estimation is generated from experiments with heated water and a thermal camera.

Huang et al. [68] propose a self-supervised learning approach that learns pouring dynamics, motion, and outcomes from unsupervised demonstrations. The poured volume is only measured in the target container using a scale. To better generalize, Kennedy et al. [64] try to pour precisely from various unknown containers. The approach combines vision- and scale-based volume estimation in the receiver and a pouring simulation in NVIDIA Flex. This solution improved the average pour error and time of 38 mL and 20 seconds achieved by [67] to an average error of 3 mL with pouring varying from 20-45 seconds and can be seen as state-of-the-art in precision. However, the setup requires a scale and a defined environment for the liquid segmentation in the target container. Another simulation-based approach was presented by Guevara et al. [69] in 2017. The work aims to minimize spilling in robotic pouring through approximate fluid simulation in NVIDIA Flex, assuming a known liquid volume in the pouring container.

One of the significant downsides of only detecting the liquid in the target container is the time delay of the pour. Most of the systems are programmed to halt and turn the source container to its original position once the estimated volume has reached the desired level. However, this method may result in over-pouring since liquid may continue to flow out of the source container during its backward rotation. This is especially relevant for pours with a low target volume in the receiving container. Additionally, pouring in multiple target containers requires multiple liquid detection setups, which could be avoided by observing the liquid in the pouring container.

This work is based on the mentioned simulation-based approaches of Kennedy et al. [64] and

Guevara et al. [69], and the same simulation environment NVIDIA Flex will be used. This approach requires an understanding of the physics of fluids and their simulation.

2.3.1 Fluid Simulation and Position-Based Fluids

Depending on the required accuracy, speed, and stability, the physics of fluids can be derived in many different ways [70]. Here, the focus lies on particle methods, which are popular for their simplicity and flexibility. One particle-based approach based on the Lattice-Boltzmann method considers fluids as particles on a lattice that move along a discrete set of directions, which leads to easy parallelization. However, this is inefficient for coarse flows [69]. Another approach sees the fluid as a continuum of particles. It solves the resulting Navier-Stokes-equations to estimate the velocity field within the fluid by sampling at various points (Eulerian) or gathering velocities of particles (Lagrangian). Nevertheless, none of the mentioned methods can be easily extended for two-way coupling between solids/objects and liquids in a mixed simulation, like pouring liquids out of a rigid body [69].

One exception for this issue is using simulation based on position-based dynamics. This fast but approximate method deals with two-way coupling by calculating the dynamics of the interaction of particles based only on their position. The main advantage of a position-based approach is its controllability. Additionally, the positions of vertices and parts of objects can be directly manipulated during the simulation, and an explicit position-based solver is easy to understand and adapt [71]. The concept of position-based fluids is based on the idea of position-based dynamics. This technique achieves comparable incompressibility and convergence as modern smoothed particle hydrodynamic solvers while benefiting from the stability inherent in geometric-based, position-based dynamics methods. Consequently, this method supports larger time steps, making it well-suited for real-time applications [72]. Most recent simulation-based pouring approaches are based on an approximate position-based particle simulator (see [73, 74, 69, 64, 75]).

2.3.2 Digital Twins

Digital twins serve as effective tools for comprehensively addressing real-world phenomena by seamlessly bridging the gap between reality and virtual predictions. By leveraging a wealth of data inputs, it becomes possible to establish connections between physical objects or processes and simulations, thereby enabling the acquisition of meaningful real-time results. Ultimately, the digital twin of the object or process of interest provides a cost-effective solution for making decisions analogous to those we would make when directly interacting with the actual object or process [76]. The main characteristic of using a digital twin is that there are two worlds, the physical and the digital. The digital one mirrors the real state of the physical world, and based on the exchange of data between the worlds, it may be synchronized [77].

Implementing these virtual replicas has already demonstrated their utility across various industries and scientific disciplines. In robotics, digital twins play a crucial role in scene understanding and interaction, enabling a seamless connection between robots and the physical reality they operate in. By leveraging digital twins, robots can effectively comprehend their surroundings and interact with the real world meaningfully [77]. In the case of a liquid pouring environment, a digital twin containing the same pouring and receiving objects, liquids, and robot coordinates must be transferred between the digital and real world. Instead of trying different pouring movements by changing the path, velocity, maximum pouring angle, and start volume in the real world, it is possible to explore the parameter space in the digital world and only execute a suitable movement on the real robot.

3 Methodology and Procedures

3.1 System Prototype

This section will mainly focus on the work to achieve the goal (1) mentioned in section 1.3: Integrating existing laboratory hardware for a proof-of-concept prototype of an autonomous cell culture system. The following sections will explain the selection and design of the hardware components, details on the connection and configuration of the system and the intended autonomous workflows.

In the planning and design phase of the system, several key considerations following the explanations made in subsection 2.1.3 were addressed to achieve a consistent overall design and architecture:

- **Modularity:** Incorporating a modular design enables the system to be easily expandable or adaptable. By dividing the system into smaller, independent modules, changes or upgrades to specific components can be made without affecting the system, which is especially important for a system prototype. Modularity facilitates future enhancements and the reuse of subelements of the system prototype for future projects regarding life science automation.
- **Integration of existing laboratory equipment:** The system integration into existing laboratory workflows should be seamless, minimizing disruptions and maximizing utilizing existing resources. This can be achieved by selecting machines and consumables already used in manual laboratory workflows. This can also be considered a cost-saver for future system scaling.
- **Simplicity:** Designing the system with simplicity aims to make it user-friendly and intuitive for operators. Simplicity enhances usability, decreases the chances of errors, and allows researchers to focus more on their experiments and analysis rather than dealing with complex system operations. This is essential for a low throughput system since various end-users often use the system instead of an experienced operator.

One of the major decisions was the choice of the right robot type. Since only one big piece of equipment is needed (incubator), the reach of a stationary arm was expected to be enough to automate the process. Hence, no mobile robots were considered in the design process. A collaborative robot arm was chosen over an industrial robot arm since the system is to be used alongside humans, and therefore safety and human-robot collaboration were essential criteria. In cell culture automation, there may be instances where human intervention or interaction is required, such as the delicate handling of sensitive samples or troubleshooting unexpected situations. Collaborative robots can assist human operators while ensuring safety and augmenting their capabilities, allowing for efficient and productive teamwork. The system is built on a moveable table with surface dimensions of 121x198 cm, with the robot mounted in the center to reach the different elements around it.

3.1.1 Components of the Hardware Setup

The following describes the purpose, characteristics, and, if applicable, the design process of the hardware components used in the system prototype. A list of the used purchased devices and elements can be seen in table B.3. The technical drawings of the 3D printed parts are located in section B.4. The CAD files can be accessed here.

Robot
The used robot is a UR5e from Universal Robots [78]. It has a maximum payload of 5 kg and a reach of 850 mm. Both parameters are suitable for the setup since the maximum weight in the process is a medium bottle with 0.75 kg, and the reach allows it to go to most positions on the table [78]. The robot is responsible for moving objects between the modules mounted on the table and handling the consumables during the process. The robot is attached to a Vention mounting plate.

Consumables
Only consumables utilized in the manual cell culture process carried out by laboratory labor are used to enable seamless system integration. These are *Nunc EasYFlask 175 cm^2 Cell Culture Flasks* [79] from ThermoFisher Scientific (from now on referred to as flask, cell flask, or cell culture flask) and *Gibco Bottles 500 mL* [80] from ThermoFisher Scientific for media and washing solution (from now on referred to as bottle, media bottle, or media/washing solution bottle). Pictures of the consumables can be seen in fig. B.15.

Multipurpose Gripper
A Hand-E Adaptive Gripper from Robotiq [81] was attached to the robot arm, and custom gripper fingers were designed to create a multipurpose gripper. The goal was to allow the robot to carry out all the tasks required in the workflow with the same gripper fingers to avoid tool changing. These tasks involve gripping the cell culture flask, the media/washing solution bottles, and capping/decapping the flask lid.

Multiple iterations of the gripper fingers were designed and printed using a Raise3D E2 3D printer. The final solution involves a design with two levels to achieve a wide range of strokes for the different sizes of flasks (40 mm) and bottles (100 mm). The lower level of the fingers has a semicircular recess for gripping and capping/decapping the flask lid. The recess's diameter and the lid's outside diameter are equal. For the final solution, adhesive tape was attached to the gripping surfaces to reduce slippage between the consumables and the plastic surface of the fingers. Figure 3.1 provides images of the multipurpose gripper. Further details of the gripper finger design are shown in fig. B.16.

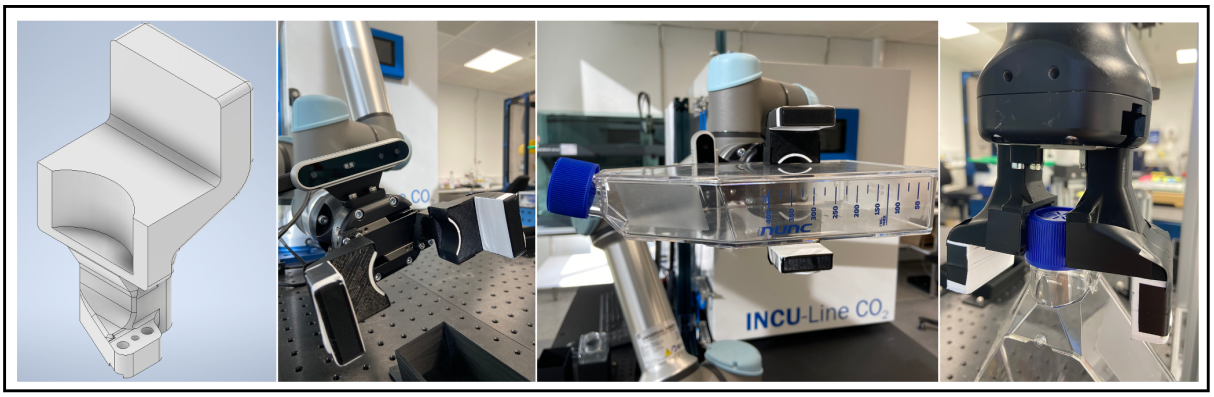


Figure 3.1: Visualisations of the gripper finger design and its functionalities. From left to right: (1) Isometric view of the gripper finger in Autodesk Inventor. (2) Gripper fingers attached to the Hand-E Adaptive Gripper from Robotiq. (3) Gripper holding a cell culture flask. (4) Gripper decapping a cell culture flask.

Incubator
One main challenge was automating a standard laboratory incubator commonly used in manual cell culture automation processes. Automated incubators are rarely commercially available and usually only made for plates, not cell culture flasks. Hence, a standard, non-automated VWR INCU-Line ILCO 180 Premium CO_2 Incubator was selected. It is designed to provide

optimal growth conditions for cell cultures while ensuring maximum sample safety. Additional characteristics include a multi-functional microprocessor controller, adjustable temperature/time profiles, and safety features such as temperature alarms, power failure control, and door alarms [82].

Multiple approaches to opening the handle and door of the incubator with the robot arm and gripper have proved unsuitable. The handle must be turned 90° and then pulled in a circular motion on the door. The biggest obstacle is the resistance when opening and closing the handle since the collaborative robot often blocks the movement because of too high force. Hence, the final solution for opening and closing the incubator is independent of the robot. It includes a pneumatic piston rod cylinder (Festo ISO cylinder DSNU-25-500-PPV-A) connected to 1 bar of compressed air each for opening and closing. The cylinder is attached to the upper back of the incubator on one end and to the door of the incubator on the other end. Extending the cylinder, therefore, leads to opening the door by pushing, and compressing leads to closing the door by pulling. Only by pulling using the cylinder, the door could not be sealed airtight, which is a prerequisite for stable CO_2 levels. To solve this issue, three linear pneumatic clamps (Festo linear/swivel clamp) were attached to the side of the incubator. These clamps are connected to dual 5-bar compressed air inputs, enabling both closing and opening actions. Once the door of the incubator is closed by pulling with the cylinder, the three clamps are closed to lock the door and make it airtight.

Conversely, when opening the incubator, the clamps are released first, followed by the extension of the pneumatic cylinder. Refer to fig. 3.2 for visuals of the incubator and the opening/closing mechanism.

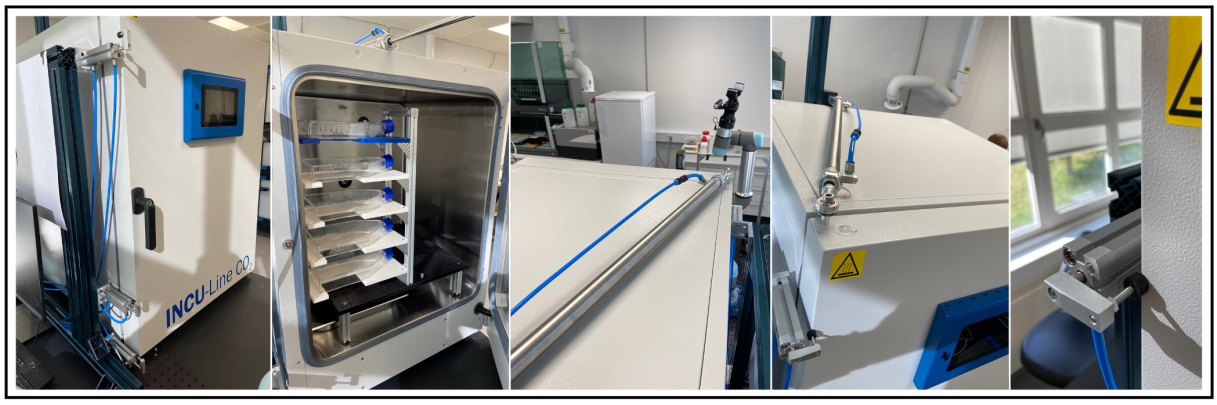


Figure 3.2: Visualisations of the incubator and its opening/closing mechanism. From left to right: (1) Closed, but not airtightly locked, door. (2) Opened door after extending the pneumatic cylinder. (3) Top view of the compressed pneumatic cylinder. (4) Front view of the closed door with locked clamps. (5) Detail view of one of the three closed clamps to lock the door.

Flask Storage

Storage locations both for empty input flasks on the table for cell passaging and for flasks inside the incubator are needed. For this prototype, input and incubator storages of five flasks were considered sufficient. To upscale the system, more storage locations for flasks would be required. This could be achieved by using a rotating flask holder. Similar to the manual process, they should be placed horizontally. Hence, a flask holder plate with a recess was designed to allow the gripper fingers to grasp the flask. Five plates each were fixed on top of each other to a structure made of 20 x 20 mm aluminum profiles. In the experimental setup, two structures are utilized: one is affixed to the table, and the other is inside the incubator. The incubator's functioning necessitates a water bath component at the bottom of the incubator chamber. To

accommodate the placement of the water bath, the flask storage unit was elevated by 10 cm, allowing sufficient space for the water bath to be positioned underneath it. Figure 3.3 offers visual representations of the flask storages.

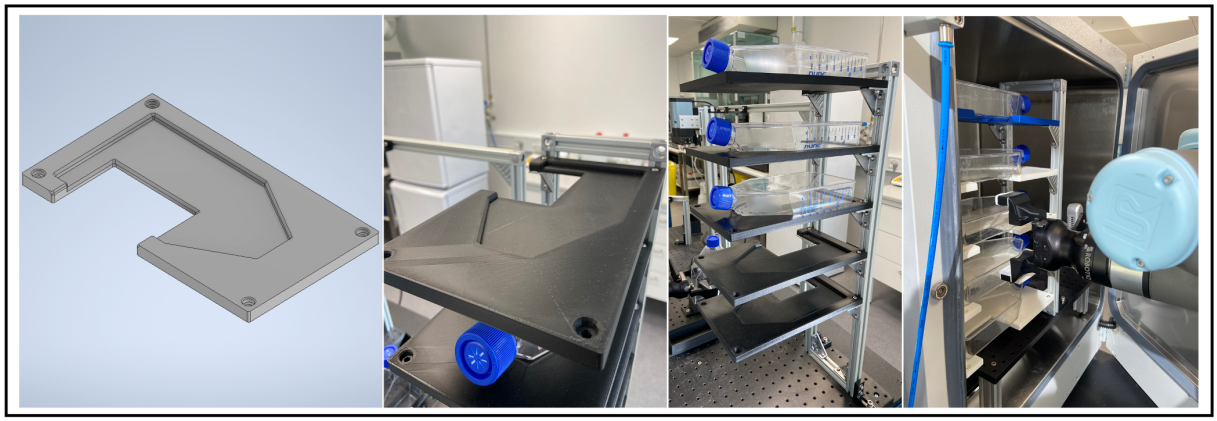


Figure 3.3: Visualisations of the flask holder plate and the final solution for flask storage. From left to right: (1) Isometric view of the flask holder plate in Autodesk inventor. (2) Detail view of a 3D printed flask holder plate. (3) Final solution of the input flask storage affixed to the table. (4) Final solution of the flask storage inside the incubator with a water bath placed underneath it.

Lid Holder

Lid holders are needed during manipulation tasks with open flasks. After decapping with the robot arm, the lid must be placed in a fixed position so that the robot can continue with the process until the flask needs to be capped again. Multiple iterations of the lid holder elements were designed and printed. The final solution is made to hold the lid upside down to avoid cross-contamination between different lids. The holder's width is chosen to fit between the closed gripper fingers when holding the lid. Five lid holder elements were 3D printed and attached to a 550 mm high structure made of 20x20 mm aluminum profiles, which allows horizontal approaching of the robot while holding the lid upside down. The final solution of the lid holder is visualized in fig. B.17.

Flask Holder for Pouring and Capping/Decapping

Multiple iterations of the flask holder as a base for pouring and capping/decapping tasks were designed and printed. Since the flask opening and the lid should be parallel to the XY plane of the robot to enable easier decapping and pouring, and the flask neck is tilted in relation to its body, the flask holder needs to be tilted as well. The final solution is tilted by 13.5° and contains several flattenings on the input level to ensure smooth insertion. Refer to fig. 3.4 for a graphical representation of the flask holder.

Camera and Camera Mounting

An Intel RealSense D415 Depth Camera [83] was selected for capturing images for computer vision tasks. It captures depth at a resolution of up to 1280x720 pixels and color imagery at 1920x1080 pixels [83]. The camera was mounted to the Robotiq gripper to achieve high flexibility and mobility, not stationary in the system. As a result, the robot arm can move the camera to various locations and angles, allowing it to capture different viewpoints. In addition, the camera can monitor different system modules during the process, ensuring continuous functionality. The mount was designed to position the camera perpendicularly to the gripper fingers to enable a wide field of view. The camera mount is attached with two M4 x 8 mm screws to the Robotiq gripper and two M3 x 8 mm screws to the camera. The camera and the camera mount can be seen in fig. B.18.

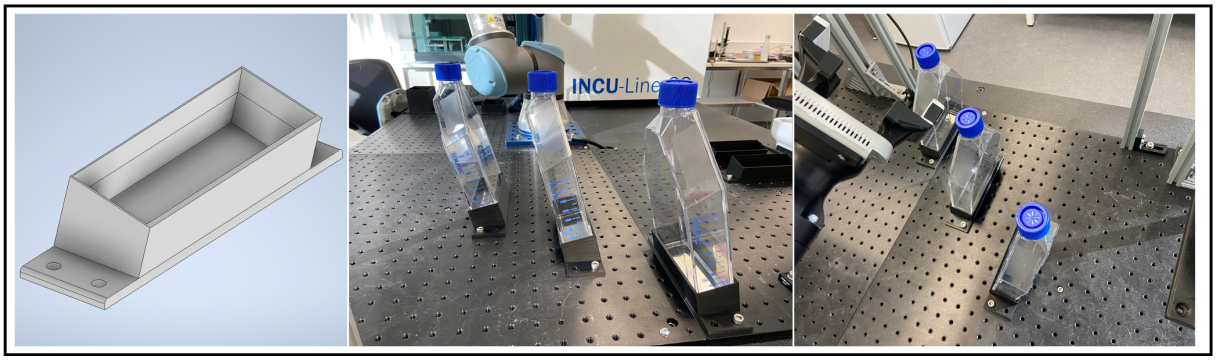


Figure 3.4: Visualisations of the flask holder for pouring and capping/decapping and its placement in the system. From left to right: (1) Isometric view of the flask holder in Autodesk inventor. (2) Detail view of three holders containing cell culture flasks, which lids are parallel to the table. (3) Top view of the flask holders affixed to the table.

Capper/Decapper

Capping/decapping the media/washing solution bottles requires a lid rotation of around 750°. Since the last joint of the UR5e has a maximum range of rotation of 720°, using the arm for this task is unpractical. The robot would need to grip the lid, rotate for 720°, open the gripper, rotate back, grip again, and finish the capping/decapping process. Therefore, an RGI 100 Rotary Electric Gripper Jaw from DH Robotics is used [84]. Four M3 x 16 mm pins are used as gripper fingers. Since it is an infinite rotating gripper, no regripping is needed. Therefore, the robot arm can hold the bottle below the gripper module, and after decapping, the module can keep the lid while the robot executes tasks with the bottle. The gripper module is lifted by attaching it to a 650 mm high 40x40 mm aluminum profile to allow easy access for the robot arm. Figure B.18 shows a picture of the capping/decapping unit.

Microscope

A CytoSMART Lux3 BR [85] was selected as a microscope to analyze the confluency of the cells. It was developed especially for cell culture applications and utilizes brightfield and digital phase-contrast microscopy to capture high-quality images of living cells. The image size of 2072x2072 pixels combined with the 1.45x1.45 mm field of view provides a 0.7 µm/pixel resolution. Another advantage is the Python-based CytoSMART Lux Open API, which allows a more straightforward incorporation of the microscope into the cell culture automation system [85]. In addition, a microscope holder was designed, 3D printed, and attached to the table to ensure a fixed position of the device and, therefore, consistent imaging quality. A picture of the microscope is shown in fig. B.18.

Trypsin Unit

Since only a very low amount of dissociation reagent such as trypsin (approximately 0.5 mL per 10 cm^2) needs to be added to the cell culture flask, pouring is unsuitable. Instead, a bottle dispenser (1-10 mL) was attached to a trypsin bottle and mounted on a structure of aluminum profiles to allow vertical insertion of the reagent into the flask. Images of the unit can be seen in fig. B.19.

Heating and Cooling of Liquids Unit

Different options were evaluated for cooling the media and the washing solution during storing and heating before pouring it into the cell culture flask. Adding a fridge to the system would require another door-opening mechanism and significant space. Instead, the final solution includes a thermoelectric chiller (TCube edge [86]). It has a temperature range of 0°C to 65°C. Cooling/heating liquid is pumped through pipes surrounding the tightly fitted media and washing solution holder.

3.1.2 Connection and Configuration

The components described in subsection 3.1.1 are combined on one table to establish a functional system. The positions of the substations were chosen iteratively during prototyping and programming, focussing on keeping a modular approach. One of the main constraints is the position of the incubator since the door can only open to one side. The incubator module, including the opening/closing mechanism, can be moved and reused for future projects. The microscope is placed close to the incubator to ensure easy and fast analysis of the flasks. The capping/decapping unit is located close to the pouring unit to avoid long movements of the robot with open bottles, which could lead to spilling. The positions of the flask holders for pouring and capping/decapping are selected so that they are reachable and approachable from the right angle by the robot. All the elements needed for passaging cells by pouring are combined on one side of the table into one pouring module. This includes three flask holders, the lid holder element, and the flask storage. The setup of the hardware modules on the table can be seen in fig. 3.5. Further images of the setup are shown in fig. B.20.

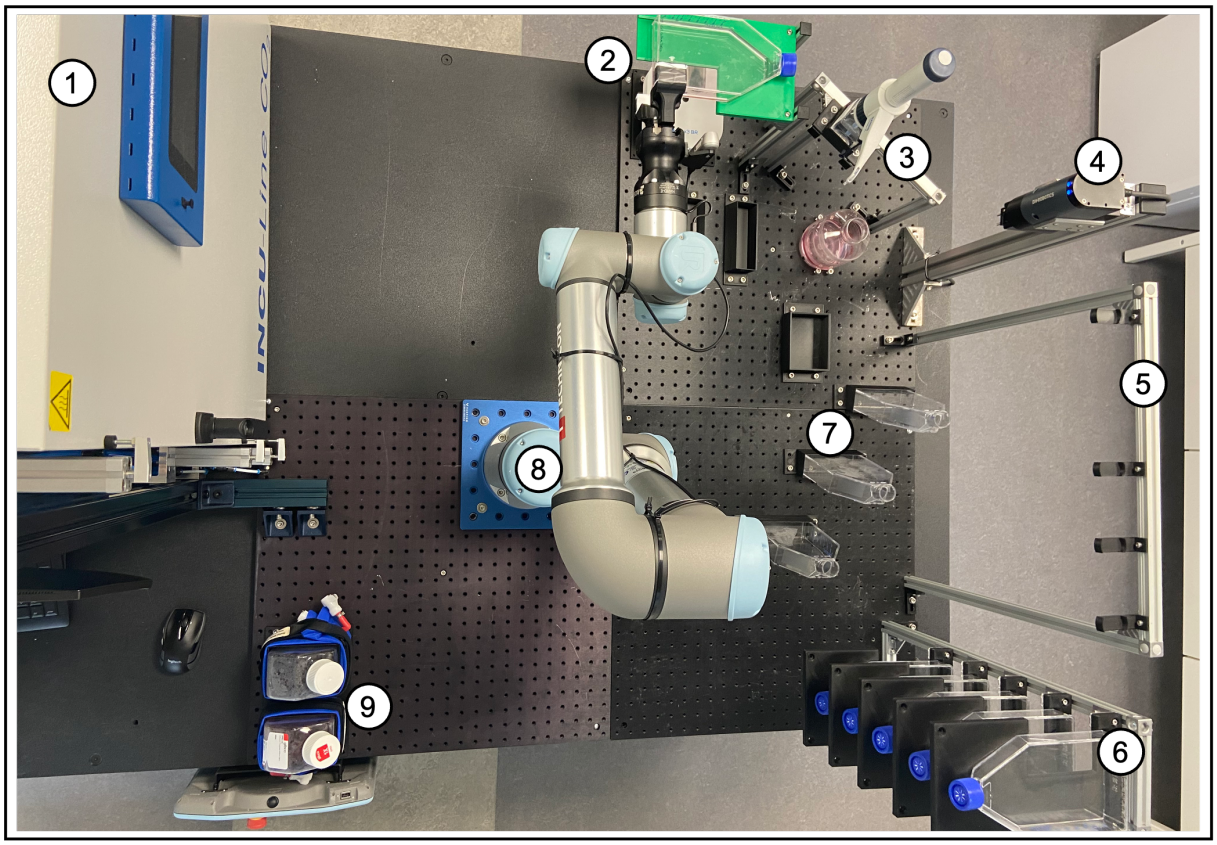


Figure 3.5: Overview of the hardware setup on the table. (1) Automated incubator. (2) Microscope. (3) Trypsin unit. (4) Capper/Decapper. (5) Lid holders. (6) Flask storage. (7) Flask holder for pouring. (8) UR5e with 3D-printed gripper fingers. (9) Heating and cooling of liquids unit.

All the devices are either connected to the UR control box or directly to the PC (Dell Precision 3630 Tower). The connection between the UR5e and its control box and the PC is established using the UR-RTDE library developed by SDU [87]. The library works by establishing a communication link using an Ethernet connection. It utilizes the Real-Time Data Exchange (RTDE) protocol, a proprietary protocol developed by Universal Robots for real-time communication [88]. The connection makes it possible to receive real-time feedback from the robot arm, send commands, and control its motion in real time.

The pneumatic clamps and the cylinder are controlled by using electrically actuated 5/2-way control valves. These are both connected to two digital output connectors of the UR control box. Each output can control the airflow in one output tube of the valve, which is responsible for extending or compressing the clamps and the cylinder. The Robotiq Hand-E gripper is connected to the robot arm by a power and communication cable with a USB adapter. The microscope, camera, and electric rotary gripper are connected to the PC by a USB cable. While the camera and the gripper can be controlled using specific Python libraries (pyrealsense2, minimalmodbus), the microscope needs the CytoSMART driver installed on the PC. The entire system is controlled by an application written in Python. This includes a simple human-machine interface to control the tasks and see the microscope's output. The recorded data (images taken by microscope and camera) are stored locally on the PC. The functional interconnection of hard- and software is visualized in fig. 3.6. The UR control box, the electric rotary gripper, the incubator, and the PC are connected to standard 230V electrical sockets.

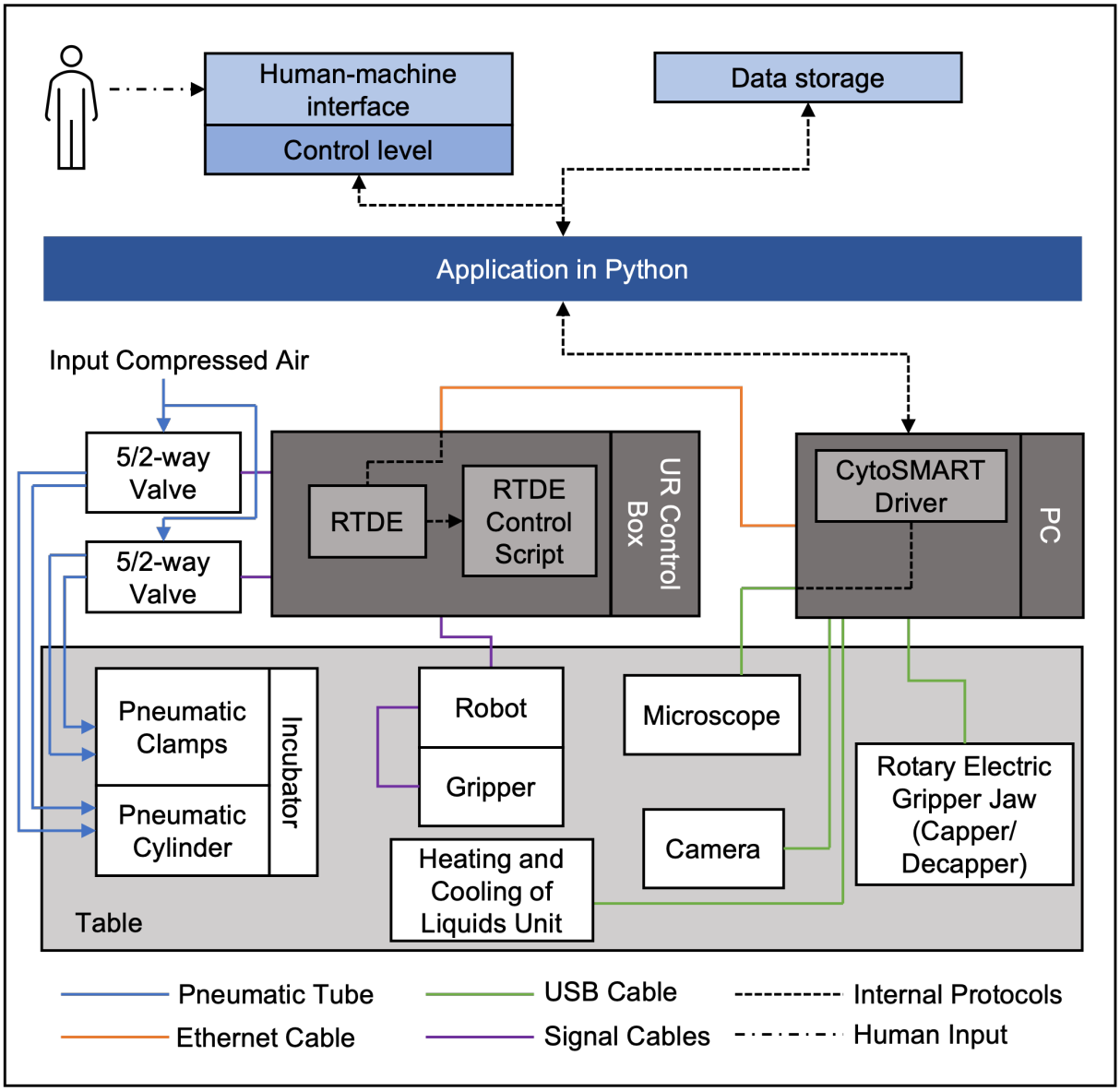


Figure 3.6: Simplified overview of the functional interconnection of hard- and software.

3.1.3 Process and workflow description

Similar to the manual process, the autonomous process is split into three high-level workflows.

- Workflow A: Analyzing cell growth.
- Workflow B: Changing media.
- Workflow C: Passaging.

The system should be able to execute the three workflows independently, depending on the user input or the scheduled plan. To enhance reusability and simplicity, the steps to achieve completion of the workflows are structured into workflow modules.

The following gives a mid-level overview of the purpose of each process module, including a summarized description of the substeps. A low-level description is given in each function block of the Python code.

- Module 1: Get a cell culture flask from the incubator (open clamps, open door, grip the flask, move the flask outside, close door and clamps).
- Module 2: Analyze cells (move flask to regripping station, regrip, place the flask on the microscope, take images at different positions, move back to start position).
- Module 3: Place a flask in a flask holder (move to the flask holder, place the flask inside, and move back to the start position).
- Module 4: Decap flask(s) (move to the flask, grip the lid, take off the lid, place the lid on a lid holder, and move back to the start position).
- Module 5: Remove liquid from a flask (take the open flask from a flask holder, move to the waste container, pour out all the liquid, and place the flask back into the flask holder).
- Module 6: Add liquid to a flask (heat media or washing solution, take the bottle, decap the bottle, regrip the bottle, pour a specific amount into the flask, regrip the bottle, cap the bottle, and place it back into the bottle holder).
- Module 7: Add trypsin to a flask (move to the trypsin unit, move up the bottle dispenser, push it down, and move back to the start position).
- Module 8: Get three empty flasks and place them into the flask holders (move to flask storage, grip an empty flask, move to the flask holder, place the flask inside, and move back to the start position (3 times)).
- Module 9: Split cells into empty flasks (take the full flask from the flask holder, move to the empty flasks, pour liquid three times, and place the empty flask into the flask storage).
- Module 10: Cap flask(s) (get a lid from the lid holder, place the lid on a flask, cap the flask, and move back to the start position).
- Module 11: Place a flask in the incubator (open clamps, open door, move the flask inside, place the flask on the flask storage, close door, and close clamps).

Throughout the implementation of the modules, careful consideration was given to selecting start and end positions, ensuring the flexibility to arrange them in any order, thus enabling diverse workflows to be executed. The process diagram illustrating the three primary workflows can be observed in fig. 3.7. This design approach allows for seamless customization and adaptation of the system to various operational requirements.

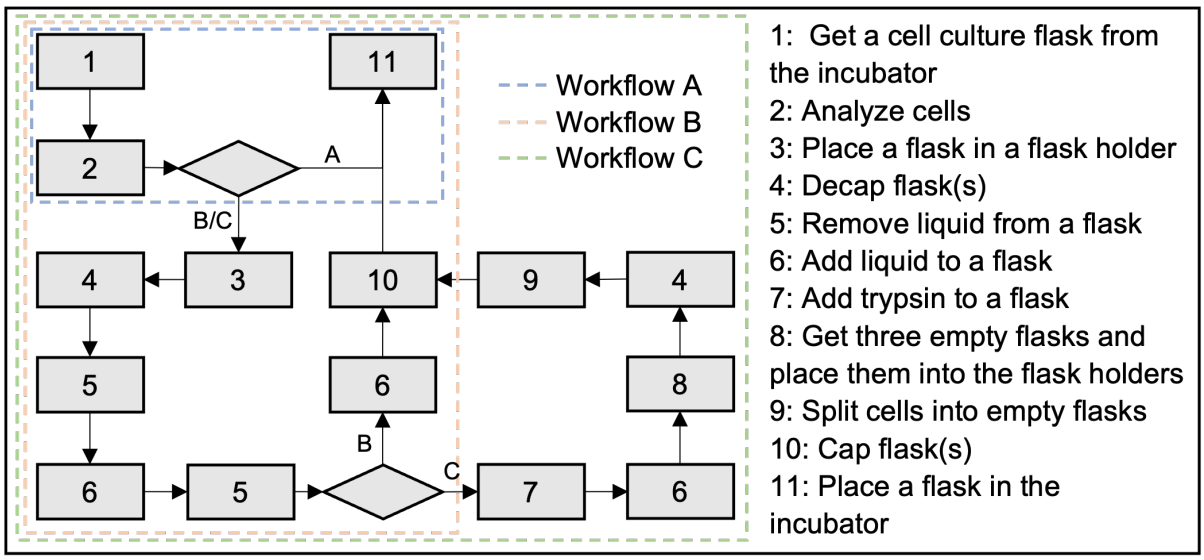


Figure 3.7: Process diagram for the three main autonomous workflows.

3.2 Vision-based Detection and Volume Estimation of Liquids

This chapter explains the methodology used for vision-based detection and volume estimation of liquids. As seen in subsection 2.2.2, very little research has been done in the field of liquid detection in research laboratories. Additionally, to the author's knowledge, there is currently no established approach for estimating volumes based on computer vision in research laboratory environments. This chapter starts by explaining the idea and approach of the vision-based system (subsection 3.2.1). It continues by providing details about the process for segmentation and depth estimation of transparent vessels and liquids inside of them (subsection 3.2.2). After, the methodology and procedures for the vision-based volume estimation of liquids will be explained (subsection 3.2.3).

3.2.1 Idea and Approach

The proposed approach consists of a two-step process for liquid volume estimation. First, train a CNN on the *TransProteus* dataset for segmentation and monocular depth estimation of liquids and transparent vessels. The resulting model can be used for liquid and transparent object detection for process monitoring (e.g., is a flask present or not). Segmented vessel and liquid XYZ maps can be created using camera intrinsics and combining the segmentation and depth maps. Subsequently, a custom dataset is generated, containing images of transparent vessels commonly used in research laboratories and the liquid volumes stored as information. The initial custom dataset is then converted into a new dataset containing segmented depth maps of liquids and vessels using the trained segmentation and depth estimation model. Finally, a new CNN is trained on the converted dataset for liquid volume estimation. The approach is based on the idea that a model can learn from the shapes of vessel and liquid as well as from the relationship between liquid and vessel maps (e.g., the percentage filled) to infer a better estimation of the volume than just from providing an image. The approach is visualized in fig. 3.8.

3.2.2 Segmentation and Depth Estimation

The aim is to create segmentation and depth maps of both transparent vessels and liquids present in research laboratories based on a single input RGB image. This can be used for process monitoring during experiments carried out in the laboratory and for flexible robotic tasks (e.g., grasping containers in an unknown environment).

The approach is based on the idea and data provided by Eppel et al. in 2020 [57] and 2022

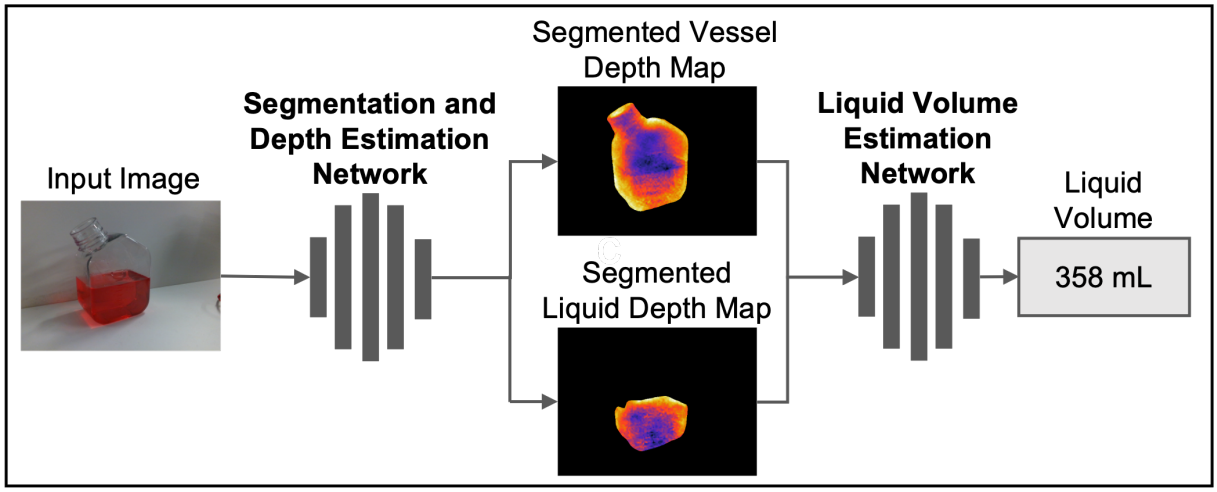


Figure 3.8: Visualization of the two-step approach for liquid volume estimation.

[16]. However, the model trained in [16] tries to predict the XYZ map directly from an RGB input image. As suggested by the authors, an approach to predicting a depth map and then transforming it into XYZ coordinates using the known camera parameters is preferable over predicting the 3D model as an XYZ map. In autonomous systems in research laboratories, the camera does not change, and the camera parameters can be assumed to be known. Hence, the model from [16] is modified to predict the depth maps instead of the XYZ maps. Also, the model in this work is only trained on liquid content inside the vessels rather than on object and liquid content.

Dataset Selection and Description

Deep learning-based approaches for computer vision rely on large annotated datasets encompassing images and their associated properties. However, manually creating such datasets is a labor-intensive process that is typically limited to simple properties such as material class (liquid/solid) and region within the image. Unfortunately, annotating 3D structures manually proves challenging due to limitations in accuracy. Moreover, viewing an object through a transparent surface introduces significant shape distortion, which human annotators are unable to account for fully. Therefore, an alternative approach that circumvents these challenges is the utilization of computer-generated images (CGI). Synthetic datasets containing transparent objects were available for some years (e.g., *ClearGrasp* [54], *SuperCaustics* [53]), but datasets containing transparent objects with liquid content were just recently published. The biggest and most diverse one is the *TransProteus* dataset [16]. That is why it was chosen for the segmentation and depth estimation in this work.

The *TransProteus* dataset consists of more than 50,000 CGI with annotated 3D models of transparent vessels and their contents, including segmentation masks and material properties such as color, transparency, reflectance, and roughness. Generated using Blender, the dataset encompasses over 500 high-definition backgrounds and more than 13,000 random objects for diverse and realistic scenes. It also includes simulated liquids with various properties using the Blender MantaFlow tool, featuring effects like splashing, foam, and bubbles. Material textures were generated using the principled BSDF shader in Blender, providing control over visual properties such as color, transparency, reflectance, and roughness. Examples of simulated vessels with liquid content from the dataset can be seen in fig. 3.9.

The dataset is available online and publically accessible [89]. It is split into eight subfolders, resulting in a total size of around 220 GB. Each subfolder is split into a training and testing folder.

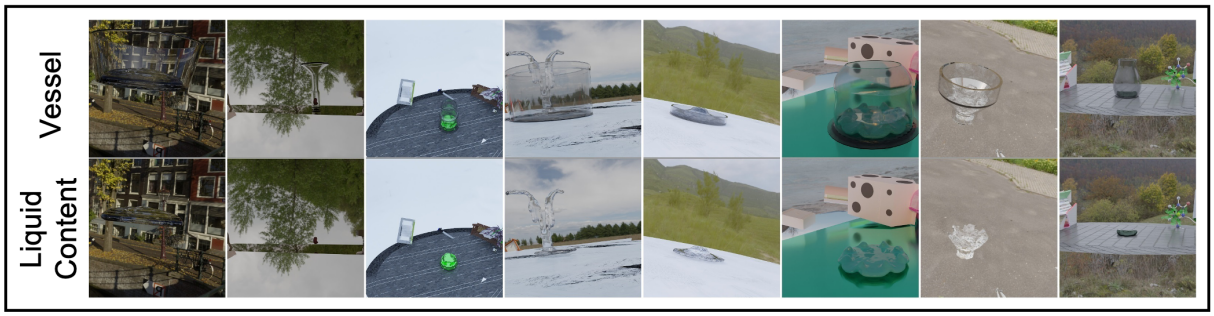


Figure 3.9: Examples of simulated vessels with liquid content from the *TransProteus* dataset.

Both of them are divided into four subdirectories depending on the vessel content (liquids, flat liquids, single objects, objects). Each folder in the subdirectories contains one sample. That includes the RGB image, content material, vessel material, camera parameters, and depth and segmentation maps of the empty vessel, the vessel opening, and the content, respectively. Examples of the detailed content of the folders can be seen in figure fig. 3.10.

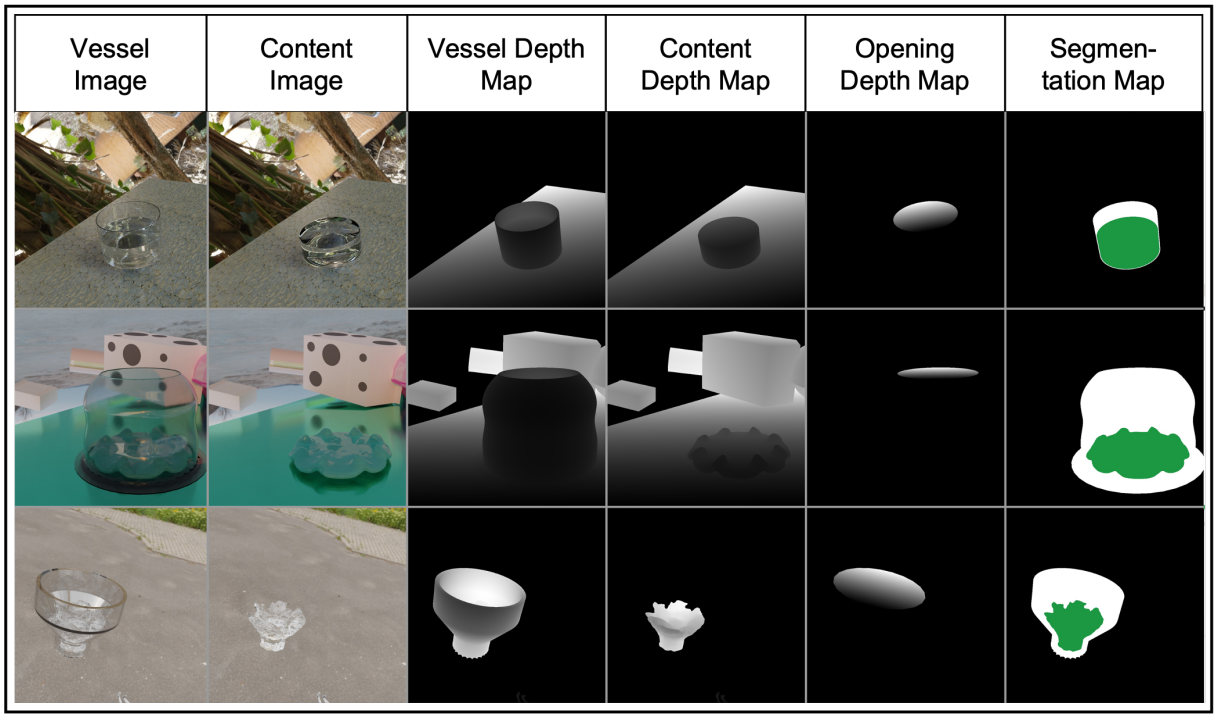


Figure 3.10: Examples of the content of a folder from three samples of the *TransProteus* dataset.

Since the model in this work should specifically focus on liquids inside the transparent vessels, the model explained in the following section is trained only on the images containing liquids. Only the subfolders called *LiquidContent* are used for that. That sums up to a dataset size of around 14,500 samples. To improve results on real-world images, the *Vector-LabPics* data [57] is used as additional training data for vessel and liquid segmentation mask prediction. The *Vector-LabPics* dataset comprises 7,900 images capturing materials in diverse stages and procedures observed in transparent vessels within chemistry labs, medical labs, hospitals, and other relevant settings. The depth loss is set to zero since no ground truth depth maps are available for the *Vector-LabPics* data. A description to download the datasets is provided in the subfolder for segmentation and depth estimation in the GitHub repository of the project.

Model Architecture

The model architecture for the segmentation and depth estimation of liquids and transparent vessels is based on the work presented by Eppel et al. [16]. The used final architecture is a modified version of a standard fully convolutional network (FCN) based on the DeepLab-v3 architecture [90, 49] with the following components:

- **Encoder:** The model uses a pre-trained ResNet-101 [91] model as the encoder. ResNet-101 is a deep residual network consisting of convolutional layers and pooling layers to extract features from the input RGB images.
- **ASPP (Atrous Spatial Pyramid Pooling) Layers:** The decoder component employs an ASPP module with dilated convolutions as in the DeepLabv3 architecture [90]. To capture multi-scale contextual information, these layers have different dilation rates (1, 2, 4, 12, 16). Using multiple scales, the model can effectively analyze objects of various sizes and preserve spatial resolution. Each of the five ASPP layers applies a dilated convolution operation followed by batch normalization and ReLU activation.
- **Squeeze ASPP Layer:** This layer applies a regular convolution operation to reduce the number of channels in the feature maps obtained from the ASPP layers from 2560 to 512. It consists of a convolutional layer, batch normalization, and ReLU activation.
- **Skip Connection Layers:** The model includes skip connections to fuse the features from the encoder with the upsampled features. These connections help retain fine-grained details from the early stages of the network during upsampling. Three different skip connection layers are added after specific layers of the encoder. Each skip connection consists of a convolutional layer, batch normalization, and ReLU activation.
- **Squeeze Upsample Layers:** These layers upsample the fused features from the three skip connections. They increase the spatial resolution of the features while reducing the number of channels. They consist of a convolutional layer, batch normalization, and ReLU activation. The three layers of skip connections and upsampling follow the U-Net structure [92].
- **Output Layers:** The model has separate output layers for each depth and segmentation mask prediction (vessel mask, vessel depth, liquid mask, liquid depth, vessel opening mask, and vessel opening depth). Each output layer is a convolutional layer that produces the predicted depth maps or segmentation masks. The vessel, liquid, and vessel opening masks are predicted as a two-layer probability mask. The depth maps for vessel, liquid, and vessel openings are predicted as a one-layer map.

Overall, the architecture is designed to take advantage of the ResNet-101 encoder's ability to extract image features. The ASPP layers and skip connections enable the model to capture both global and local context, while the squeeze layers and upsample layers refine and upsample the features to produce the final predictions. The input and outputs of the model are visualized in fig. 3.11.

Implementation Details

The model and all required scripts for training, evaluating, and using the model were implemented using Python and the PyTorch library. The project folder for the usage of the segmentation and depth estimation of liquids and transparent vessels is a subfolder of the main project repository. It follows the structure of the Cookiecutter Data Science project template, which provides a standardized and organized layout for data science projects. Further implementation details, including a description of how to train, evaluate, and use a model, are described in appendix A.

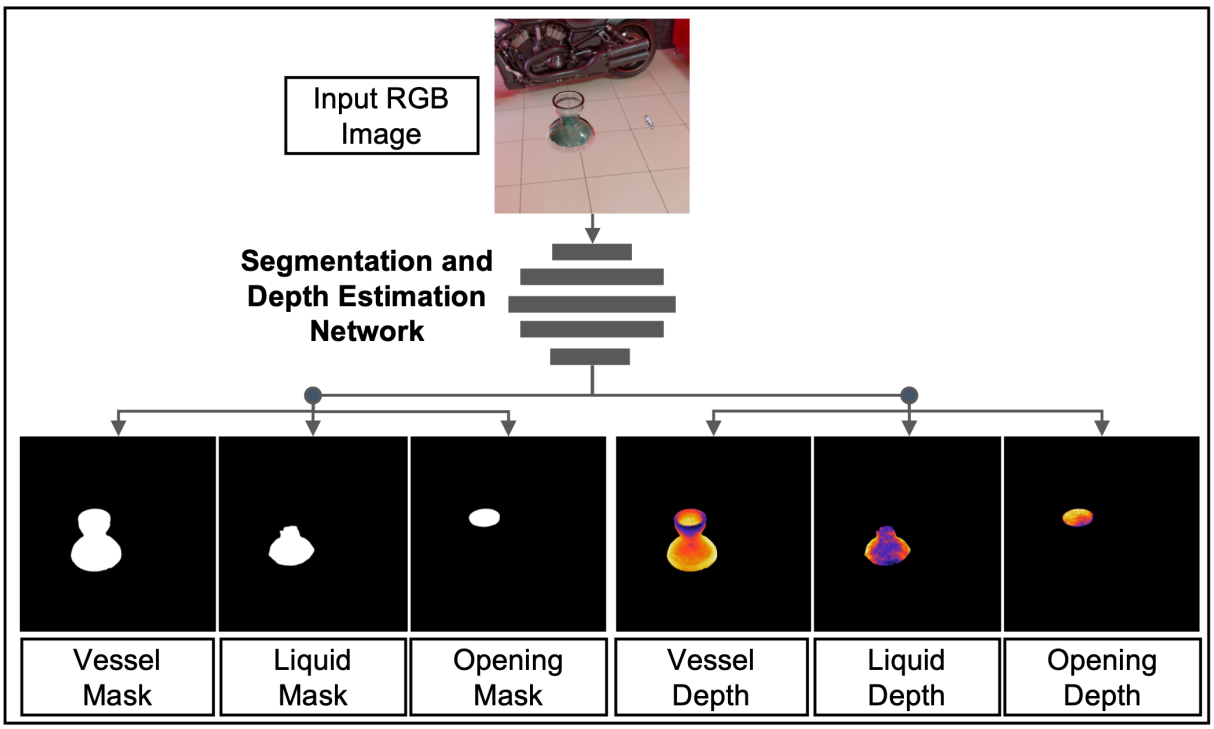


Figure 3.11: Visualization of the input and output layers of the segmentation and depth estimation model for an example of the *TransProteus* dataset.

Experimental Setup

All training processes were carried out on a Tesla A100-PCIE GPU available in the DTU HPC clusters. The final model was trained for 75 epochs using the described subset of the *TransProteus* dataset and the *Vector-LabPics* dataset. The *Vector-LabPics* data was applied in 33% of the training steps. Image augmentation for training data includes resizing, centered cropping, gaussian blurring, decoloring, and darkening. The testing data is already excluded in an extra folder in each folder of the *TransProteus* dataset. An Adam optimizer was used with an initial learning rate of 1×10^{-5} and a weight decay of 4×10^{-5}. The learning rate is adjusted every five epochs based on the average loss. If the average loss does not decrease for ten consecutive epochs, the learning rate is reduced by 10%. The batch size was set to 6. The trained model weights and optimizer state are overwritten after each epoch. After every fifth epoch, the model gets saved permanently. The loss for each segmentation mask is calculated using the standard per-pixel cross-entropy function.

The loss for each depth map is calculated only for the region of the object as given by the specific ground truth mask. As described, for *Vector-LabPics* samples, the depth loss is set to zero. In the *TransProteus* dataset, the scale of the 3D model can not be extracted from the image. Therefore, the loss needs to be independent of the predicted model scale, which is why the scale-invariant depth loss, as introduced by Eigen et al. [51], is used. The scale-invariant depth loss is defined in eq. (3.1):

$$L(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N} {d_i}^2 - \frac{1}{N^2}(\sum_{i=1}^{N} d_i)^2 \tag{3.1}$$

where $d_i = log(y_i) - log(\hat{y}_i)$. y_i is the predicted depth at pixel i, and $\hat{y}_i$ the ground truth depth for pixel i, and N the number of pixels in the region. Note that the network output is $log(y)$. Hence,

the final linear layer predicts the log depth, and the depth maps need to be converted to linear scale before further use.

The loss curves of the category losses of vessel depth, liquid depth, vessel opening depth, vessel mask, liquid mask, and vessel opening mask for the described training are shown in fig. 3.12.

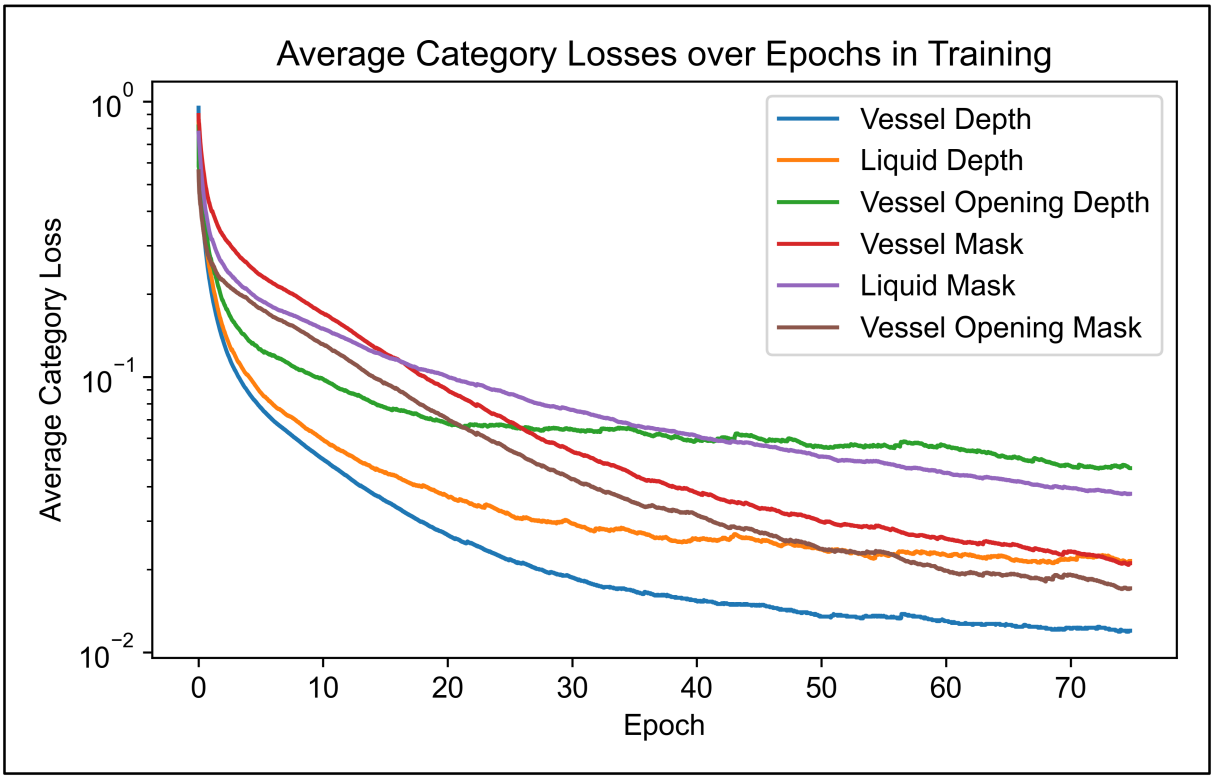


Figure 3.12: Category loss curves of vessel depth, liquid depth, vessel opening depth, vessel mask, liquid mask, and vessel opening mask.

In the evaluation of the segmentation of the vessel, liquid, and vessel opening regions, the standard Intersection over Union (IoU) metric is used. IoU is a widely-used metric for assessing semantic segmentation and is calculated separately for each region. It quantifies the degree of overlap between the pixels identified as belonging to a region in both the network prediction and the ground truth from the dataset. The IoU is computed by dividing the intersection of these pixels by their union. Additionally, $Recall$ is calculated as the intersection divided by the total number of pixels belonging to the region according to the ground truth annotation. $Precision$ is computed as the intersection divided by the total number of pixels assigned to the region based on the model prediction.

Evaluation of the depth map prediction is done using a metric suggested by Eigen et al. [51]. Since the scale of the 3D model in the *TransProteus* dataset is arbitrary, the scale-dependent metrics like the linear root mean squared error ($RMSE$) have little meaning. Therefore, only the $scale-invariant\ log\ RMSE$ is used. It is calculated by taking the mean of the values using the square root of the result of eq. (3.1), resulting in eq. (3.2).

$$RMSE(log, scale-invariant) = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\left(\frac{1}{N}\sum_{i=1}^{N}d_i^2 - \frac{1}{N^2}\left(\sum_{i=1}^{N}d_i\right)^2\right)} \tag{3.2}$$

where T is the number of samples, and d_i and N the same as defined for eq. (3.1).

3.2.3 Liquid Volume Estimation

The aim is to predict the volume of liquids inside transparent vessels present in research laboratories based on a single RGB image. For the input of the volume estimation model, the described segmentation and depth estimation network is used first to create the segmented depth maps of vessels and liquids. A vision-based volume estimation can be used for various robotic tasks in- and outside of research laboratories. Exemplary use cases for research laboratories are the autonomous pouring of liquids based on a known starting volume or the control of liquid materials in a self-driving laboratory. Almost empty vessels could be detected and automatically replaced using a mobile robot. To the author's knowledge, no similar two-step approach for the vision-based estimation of liquid volume has been used. The idea is based on the food volume estimation approach by Graikos et al. [93]. However, instead of the proposed point-cloud-to-volume calculation, another CNN is used.

Data Generation

Since the *TransProteus* dataset does not include a volume ground truth, it can not be used for liquid volume estimation. To the author's knowledge, there is no publicly available dataset that contains images of liquids and their volumes in laboratory environments. Therefore, a new dataset called *LabLiquidVolume*, which fills that gap, was created. It includes 5,451 images of liquids in laboratory containers.

The images were taken using an Intel RealSense D415 camera. The ground truth of the liquid volume was measured using a Mettler Toledo XSR2002S balance with an accuracy of ± 0.5 mL. The camera was connected to a PC via a USB cable. The balance has a digital display for data readout. The images were generated in different environments in the automation laboratory and various research laboratories at the Novo Nordisk R&eD site in Måløv, Denmark. The setup for one of the data generation scenes is shown in fig. 3.13.

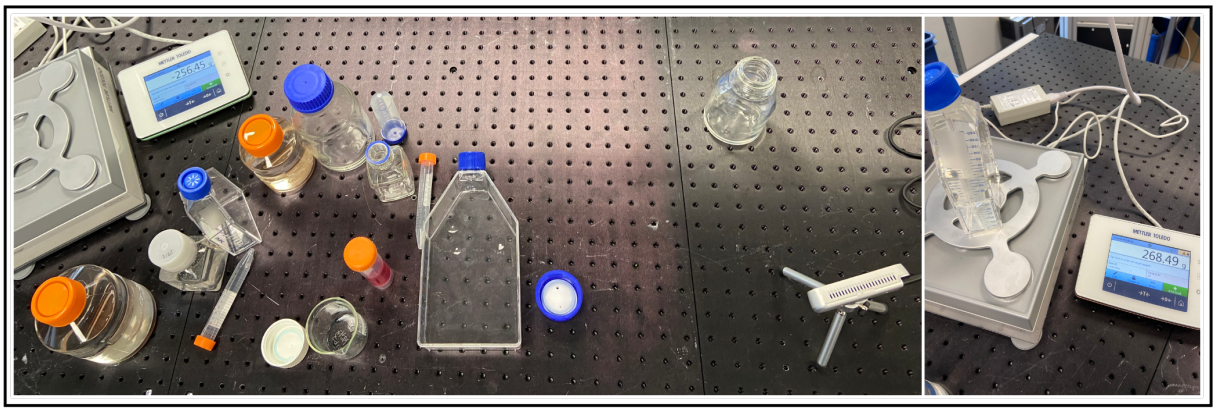


Figure 3.13: An exemplary setup for the generation of the *LabLiquidVolume* dataset. It includes the laboratory vessels, the balance, and the Intel RealSense D415 camera.

For the data generation, a simple user interface was created in Python. It asks the user to select the vessel name and liquid color and to enter the liquid volume. The camera view is streamed in another window. The sample is saved by clicking capture, and the corresponding data is stored in a new subfolder. The user interface during the data generation can be seen in fig. 3.14.

Twelve of the most common research laboratory containers, including the consumables used in cell culture processes, were selected for the dataset. Details of the vessels (official name, company, material, height, opening diameter, max. width) are listed in table B.1. The twelve empty vessels are visualized in fig. 3.15. In consultation with laboratory scientists, three different

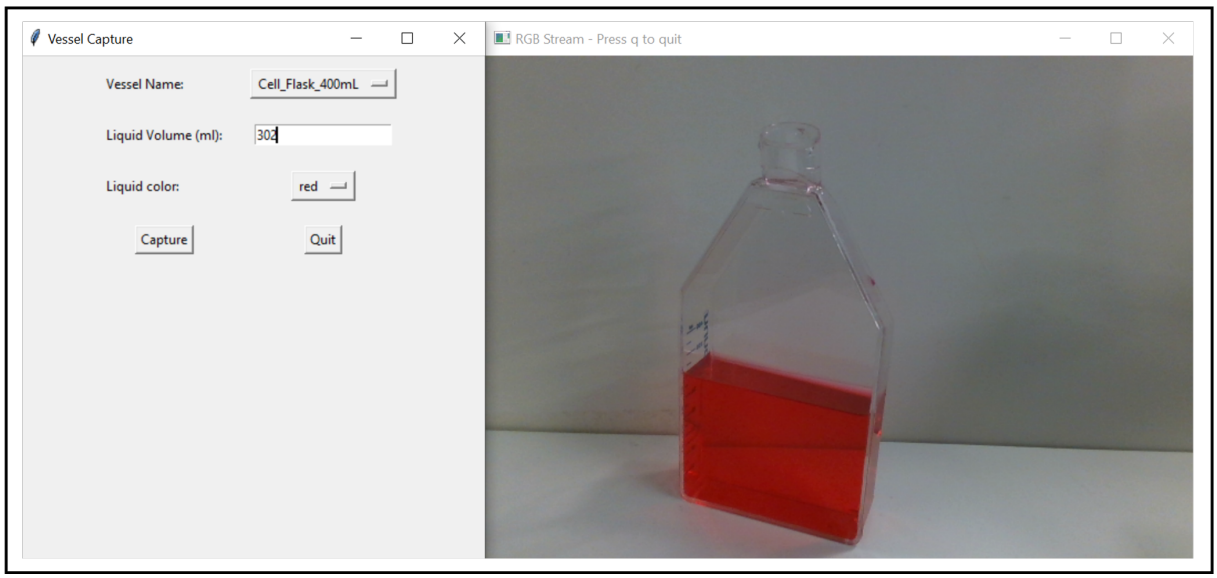


Figure 3.14: The user interface during data generation. Left: User input window for vessel name, liquid volume, and liquid color, and buttons for capturing images and quitting. Right: Live camera RGB stream.

liquid colors were chosen: red, green, and blue. Besides transparent liquids, these are the most frequent colors of liquids in laboratory experiments. Due to a recognized high variability of the results of the segmentation and depth estimation for completely transparent liquids, these were not used for the dataset. To create as diverse a dataset as possible, many different liquid volumes were chosen. Depending on the vessel, these range from 3 to 600 mL. For the same reason, various distances to the object (50 - 600 mm) and different camera angles were used. However, all images were taken from above so that the surface of the liquid is visible. To illustrate the diversity of the dataset, 20 sample images are shown in fig. B.1.

Figure 3.15: The vessels selected for the *LabLiquidVolume* dataset. From left to right (short names used in the dataset): Duran_500mL, Duran_250ml, Duran_100mL, Storage-Bottle_500mL, Storage_Bottle_250mL, Pyrex_100mL, Cell_Flask_400mL, Cell_Flask_160mL, Gibco_500mL, Gibco_125mL, Tube_50mL, Tube_15mL.

An initial dataset is created during the data generation, which is then converted to the final dataset using the trained segmentation and depth model. The initial dataset contains a subfolder for each sample. Each subfolder includes the recorded RGB image and depth map from the camera (as .png and .npy files) and the liquid color, the vessel name, the liquid volume, and the vessel volume in separate text files.

For the processed final dataset, the content of each subfolder is copied and extended by the

output of the segmentation and depth estimation model using the RGB images as input. This includes the segmented liquid and vessel depth maps, the liquid and vessel segmentation masks (each as .png and .npy files), and the unsegmented depth maps of liquid and vessel. Additionally, a combined image is created for easier visualization of the samples, including the original RGB image, the normalized segmented liquid and vessel depth maps, and the liquid and vessel segmentation masks. Consequently, the converted final dataset consists of 5,451 subfolders with the following content:

- *Input_color.txt*: Liquid color (red, blue, or green)
- *Input_ContentDepth_segmented.npy/.png*: Segmented liquid depth map.
- *Input_ContentDepth.npy*: Liquid depth map.
- *Input_ContentMaskClean.npy/.png*: Liquid segmentation mask.
- *Input_DepthMap.npy/.png*: Recorded depth map from camera.
- *Input_EmptyVessel_Depth_segmented.npy/.png*: Segmented vessel depth map.
- *Input_EmptyVessel_Depth.npy*: Vessel depth map.
- *Input_RGBImage.npy/.png*: Recorded RGB image from camera.
- *Input_vessel.txt*: Short name of the vessel.
- *Input_VesselMask.npy/.png*: Vessel segmentation mask.
- *Input_visualize.png*: Combined image showing the original RGB image, the normalized segmented liquid and vessel depth maps, and the liquid and vessel segmentation masks.
- *Input_vol_liquid.txt*: Liquid volume.
- *Input_vol_vessel.txt*: Vessel volume.

Three samples of the generated *LabLiquidVolume* dataset are visualized in fig. 3.16.

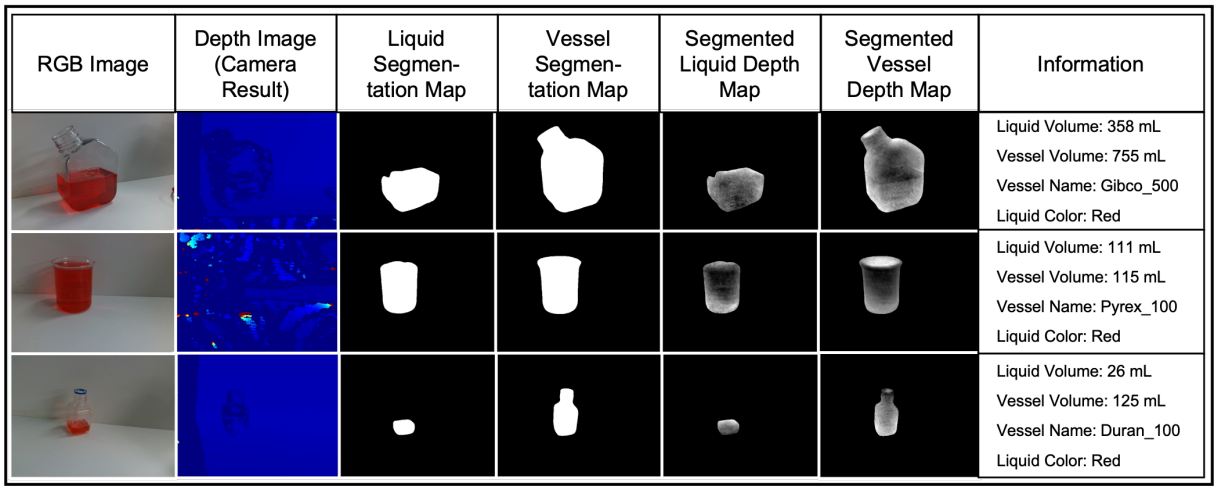


Figure 3.16: Visualization of the content of three samples of the generated *LabLiquidVolume* dataset.

The vessels in the images in the dataset have an average maximum volume of 178 mL and an average fill proportion of 48%. 31% percent of the samples contain green, 57% red, and 12% blue liquid. A more detailed analysis of the distribution of the samples can be seen in table 3.1. A histogram of the distribution of liquid volumes in the dataset is shown in fig. B.2.

Table 3.1: Details of the content of the *LabLiquidVolume* dataset.

Name	Number of Samples	Max. Vessel Vol.	Average Liquid Vol.	Average Fill Level	Prop. of Green Samples	Prop. of Red Samples	Prop. of Blue Samples
Cell_Flask _160mL	417	268	125	47%	40%	41%	19%
Cell_Flask _400mL	665	647	292	45%	45%	34%	21%
Duran_100mL	335	125	58	46%	49%	51%	0%
Duran_250mL	409	302	161	53%	21%	65%	15%
Duran_500mL	631	594	290	49%	24%	63%	13%
Gibco_125mL	652	180	86	48%	23%	47%	8%
Gibco_500mL	512	755	321	43%	34%	72%	21%
Pyrex_100mL	530	115	60	52%	26%	59%	15%
StorageBottle _250mL	401	359	183	51%	32%	68%	0%
StorageBottle _500mL	423	682	281	41%	22%	78%	0%
Tube_15mL	119	16	9	56%	32%	68%	0%
Tube_50mL	357	55	31	56%	33%	53%	14%
Total	**5451**	**385**	**178**	**48%**	**31%**	**57%**	**12%**

Model Architecture

The neural network architecture employed for the volume estimation is a basic CNN as explained in [45]. It comprises multiple layers, including convolutional, batch normalization, and fully connected layers. The input of the main presented network consists of two components: vessel depth map and liquid depth map. The network processes these depth maps to generate a corresponding liquid volume prediction.

The architecture begins with convolutional layers, which extract relevant features from the input depths using a series of 2D convolutions. The first convolutional layer takes in two channels, representing the vessel and liquid depth. It applies a kernel size of 5x5 and utilizes padding to maintain spatial dimensions. The subsequent convolutional layers gradually increase the number of output channels, allowing for more intricate patterns and features to be extracted.

To enhance the training process and reduce overfitting, batch normalization layers are incorporated after each convolutional layer. These layers normalize the outputs, ensuring more stable and efficient learning throughout the network. Max pooling layers are inserted to downsample the feature maps, reducing spatial dimensions while retaining important information. This helps to capture and preserve the most relevant features while discarding less significant details. Dropout layers are included to randomly deactivate a portion of the neurons during training, reducing interdependencies and encouraging the network to learn more robust representations. The architecture has a configurable dropout rate, allowing control over the dropout intensity during training.

Following the convolutional and pooling layers, the feature maps are flattened and passed

through fully connected layers. The fully connected layers allow the network to learn complex relationships and mappings between the input depths and the desired volume output. The number of neurons gradually decreases across the fully connected layers. ReLU activation is applied after each convolutional and fully connected layer, introducing non-linearity and enabling the network to model more complex and nonlinear patterns. The final fully connected layer outputs a single value, representing the predicted liquid volume.

The second main model, besides the mentioned model trained with the segmented depth maps as input, is the vessel volume input model. In addition to the segmented depth maps, the model takes the maximum volume of the vessel as an input. The vessel volume is transformed using a fully connected layer before being concatenated with the features extracted from the convolutional layers of the segmented depth maps.

Implementation Details

The implementation of the model, along with all the necessary scripts for training, evaluation, and utilization, was implemented in Python using the PyTorch library. Specifically, the project folder dedicated to the volume estimation of liquids is a subfolder within the main project repository. It follows the same organized structure as the previously described subproject for segmentation and depth estimation.

The `main.py` file serves as the main entry point for running the liquid volume estimation. It provides three modes of operation: training the model, predicting the volume based on an RGB image, and evaluating the model on a folder of images containing the segmented liquid and vessel depth maps.

Further implementation details, including a description of how to train, evaluate, and use a model, are described in appendix A. An explanation of how to get the dataset and the trained models depicted in this section is also included.

Experimental Setup

The training processes for the liquid volume estimation were performed on a Tesla V100 GPU, which is available in the DTU HPC clusters. The final model was trained for 200 epochs using 90% of the *LabLiquidVolume* dataset. 10% of the training data is used for validation. All testing results are based on the remaining 10% of the data (545 samples). The input images undergo a downsampling process as part of the preprocessing stage. A grid search was done for hyperparameter tuning to find the best learning rate, batch size, and dropout rate. Learning rates of 0.0001, 0.001, and 0.01, batch sizes of 2, 4, 8, and 16, and dropout rates of 0.1, 0.15, 0.2, and 0.25 were used. The final model was trained with a learning rate of 0.001, a batch size of 8, and a dropout rate of 0.2.

In this experimental setup, the chosen loss function is the Mean Squared Error (MSE), which is commonly used for regression problems. The MSE loss is calculated by taking the average of the squared differences between the predicted values and the actual target values. The Adam optimizer is used for iteratively adjusting the model's parameters to minimize this loss. The loss curves of the two main models during training are visualized in fig. 3.17.

In addition to the two mentioned main models, two additional approaches were chosen for comparison:

1. Segmentation mask model: Instead of the segmented depth maps, this model only takes the segmentation masks of vessels and liquids as inputs.
2. Scaled segmented depth maps model: To provide a better global scale, the segmented depth maps are first normalized and then multiplied by the median of the segmented real depth maps captured by the RealSense camera. The two scaled segmented depth maps are the input of the model.

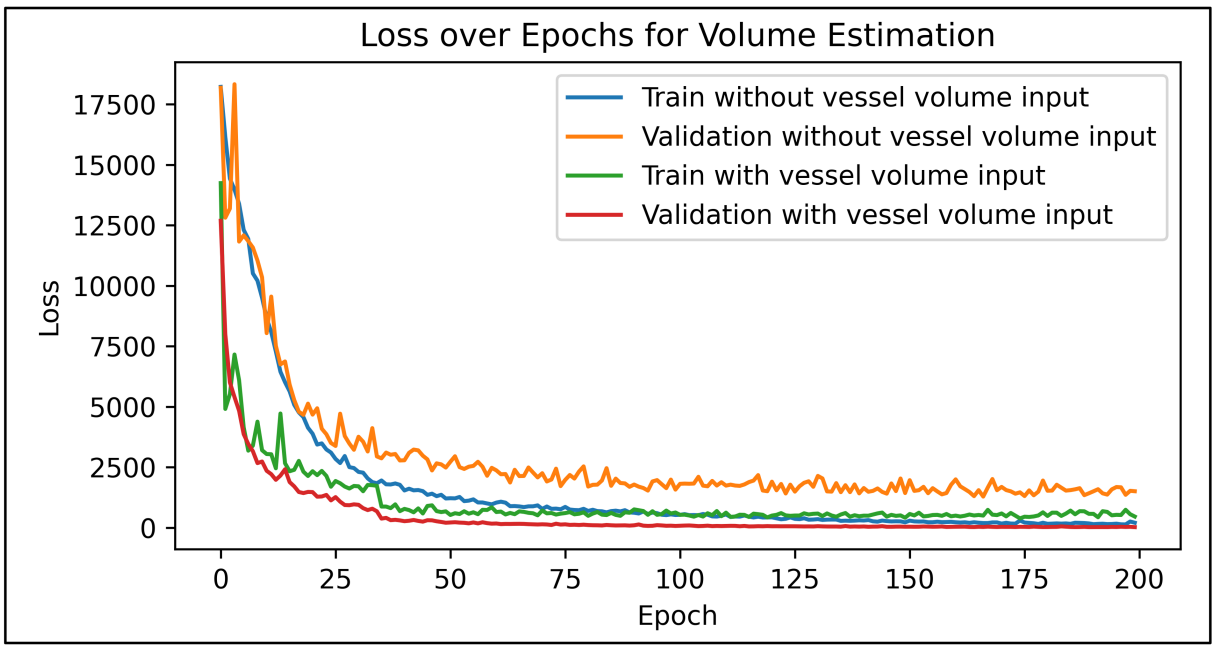


Figure 3.17: Loss curves of the two main models during training. (1) Only segmented depth maps as input. (2) Segmented depth maps and vessel volumes as input.

All the additional models were trained and tested with the same data and parameters described above. An overview of the four approaches is visualized in fig. 3.18.

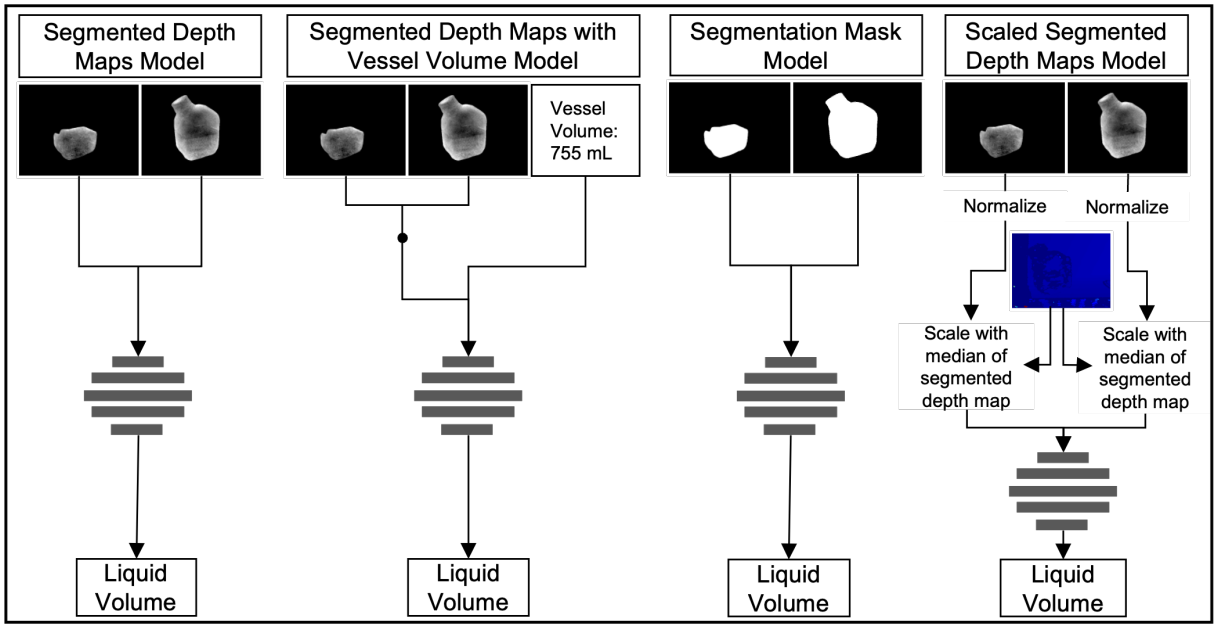


Figure 3.18: Overview of the four approaches for liquid volume estimation based on the *LabLiquidVolume* dataset.

Three different metrics, which are commonly used for regression problems, are used for evaluating the performance of the described models (all based on [94]): The root-mean-squared error ($RMSE$, eq. (3.3)), the mean absolute percentage error ($MAPE$, eq. (3.4)), and the R^2 score (coefficient of determination, eq. (3.5)).

The results are calculated based on the real and predicted liquid volumes. In the following, V_{real} stands for the actual liquid volume inside the vessel, $V_{predicted}$ for the predicted liquid volume from the model, n for the number of samples, and $\bar{V}_{real}$ for the mean of the real liquid volumes.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(V_{predicted,i} - V_{real,i})^2} \tag{3.3}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{V_{predicted,i} - V_{real,i}}{V_{predicted,i}}\right| \tag{3.4}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(V_{real,i} - V_{predicted,i})^2}{\sum_{i=1}^{n}(V_{real,i} - \bar{V}_{real})^2} \tag{3.5}$$

3.3 Simulation-based Autonomous Pouring

This chapter explains the methodology behind the approach for simulation-based autonomous pouring from laboratory vessels. The aim is to be able to use the vision-based liquid volume prediction for the pouring container to subsequently take the results from the simulation to get the required robot movement to pour a specific amount of the predicted liquid volume into a target vessel. The idea is visualized in Figure 3.19. This chapter starts by giving an overview of the simulation framework (subsection 3.3.1), followed by the details about the implemented pouring simulation (subsection 3.3.2). Finally, the simulation-to-reality transfer to the robot is explained (subsection 3.3.3).

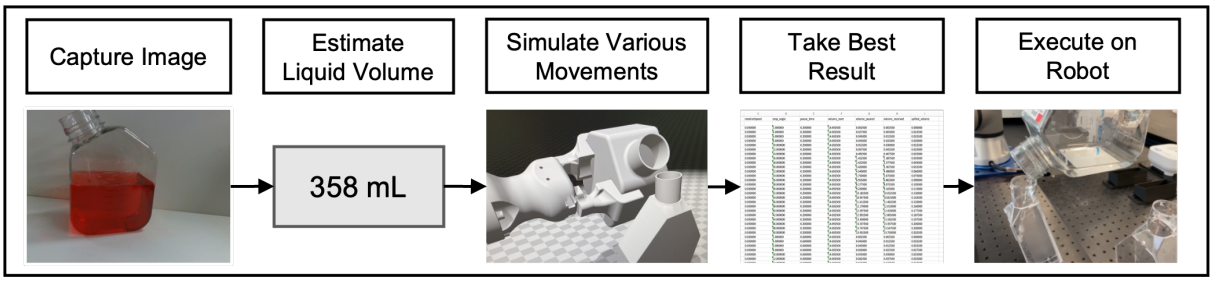


Figure 3.19: Visualization of the overall idea of the autonomous robotic pouring approach based on simulation.

3.3.1 Simulation framework

As mentioned in section 2.3, this work is based on the same simulation framework used in many recent approaches for liquid pouring: NVIDIA Flex. NVIDIA Flex is a particle-based simulation library designed for real-time applications. It is based on position-based dynamics [71] and unified particle physics for real-time applications [95]. NVIDIA Flex revolves around a fundamental concept: representing everything as a system of particles interconnected by constraints. As explained in section 2.3, this representation offers several advantages, including efficient modeling of diverse materials and seamless interaction between elements of different types. Notably, it enables two-way coupling between rigid bodies and fluids, facilitating a comprehensive simulation of their interactions. By adopting this particle-based approach, NVIDIA Flex allows the simulation and study of a wide range of materials with complex behaviors in a computationally efficient manner.

The NVIDIA Flex library consists of two main parts: the core solver (NvFlex.h) and the extensions library (NvFlexExt.h). The core solver operates on flat arrays of particles and constraints, efficiently handling large numbers of elements. The extensions library provides object/asset management features. The core solver is closed source, while the extensions library and demo application come with full source code. The API follows a C-style design and requires a structure-of-arrays data layout for optimal simulation performance. Particles serve as the fundamental building blocks. Each particle contains essential dynamic attributes such as position, velocity, and inverse mass. The inverse mass is stored with the position in the format [x, y, z, 1/m]. This is especially useful for counting particles to calculate volumes of liquids inside various vessels at different timesteps. Additionally, Flex assigns a per-particle phase value that governs the behavior of the particles.

All simulations were executed on a Vision Gaming Windows 11 PC with an NVIDIA GeForce RTX 4090 graphics card and 64 GB of RAM.

3.3.2 Pouring simulation

Two scenes are created to simulate the two main liquid pouring movements required for cell culture automation. One for pouring from a cell culture flask, and one for pouring washing solution or media from a Gibco 500 mL bottle, both into another flask standing in one of the flask holders mounted on the table.

Object Import

The 3D models of the pouring and receiving objects are required for a precise simulation. The vendor of the specified vessels was contacted, but because of reasons of confidentiality, the 3D models were not received. Therefore, the 3D models were created using Autodesk Inventor based on the publicly available 2D drawings. This is particularly difficult for complex structures such as the opening neck of the cell culture flask. The origin of the coordinate system was placed in the center of the objects to allow easier later usage as a tool-center-point (TCP) for the robot movement. A visualization of the 3D models can be seen in fig. B.5.

The 3D models are loaded to the simulation as object files (.obj) and converted into a triangle mesh object. The flask holder, including the receiving flask, is placed on the ground plane of the simulation and centered on the origin of the coordinate system. The starting setup in the simulation for pouring with the cell culture flask and the media/washing solution bottle can be seen in fig. 3.20

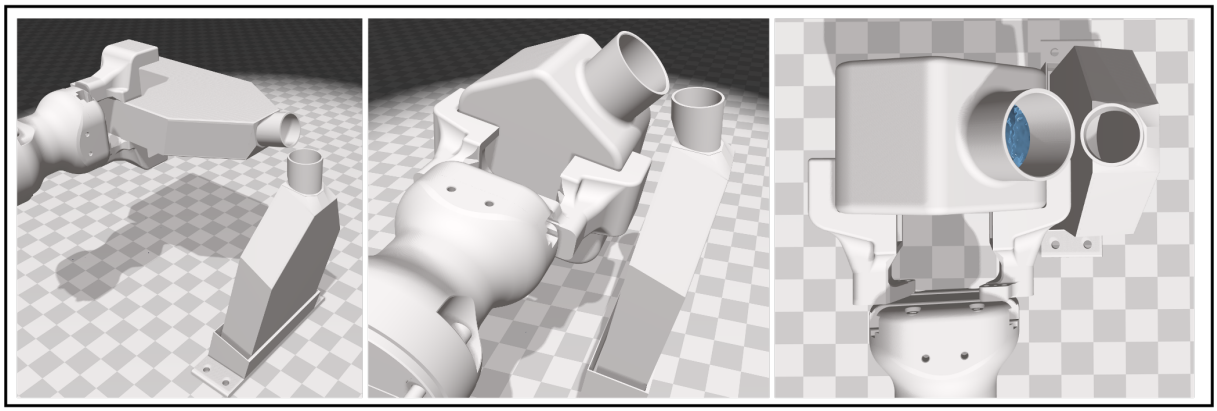


Figure 3.20: The starting setup of the pouring scenes in NVIDIA Flex. Left: Cell culture flask and receiving container. Center and right: Washing solution/media bottle and receiving container.

Liquid Generation and Calibration

At the start of every pouring simulation scene, the pouring container is filled with a certain amount of liquid. A liquid emitter is created in the center of the containers. At every selected

simulation timestep, this emitter creates new liquid particles. The number of particles is proportional to the area of the emitter. After the required amount of particles is created, ten seconds of waiting time is added to let the water settle. Since the stored attributes of the particles at each time step include the coordinates, the amount of liquid inside of regions inside and outside of the pouring and receiving container can be calculated.

All liquids used in the cell culture process are assumed to behave similarly to water. To achieve a good simulation-to-reality transfer, the liquid parameters for the simulation must be varied, transferred, and tested in reality. Multiple approaches, including grid search and Bayesian optimization, were used for this problem. Since this procedure was done in [74] and [73], the best liquid parameters found for water in these works are used for the simulation. These can be found in table 3.2.

Adhesion	Cohesion	Surface Tension	Viscosity
0.0001	0.001	0.005	0.01

Table 3.2: Liquid parameters used for the simulation in NVIDIA Flex.

To use the simulation results in reality, the number of particles must be converted to liquid volume. For that, a calibration scene was created. This includes a cylindrical container with an inner volume of 500 mL. This container was designed in Autodesk Inventor, imported to the scene, and then filled with particles with the mentioned parameters. The maximum amount of particles that fit into the container was iteratively determined. The ratio between the number of particles and the volume is used as a conversion factor. The calibration scene is visualized in fig. B.6.

Pouring Movement

Most recent approaches for autonomous robotic pouring only consider a rotation of the wrist of the robot (e.g., [64, 65, 67, 68, 69, 73]). That means that the x, y, and z coordinates of the TCP of the robot stay constant. Consequently, the exit point of the liquid is constantly changing. Although this is possible for receiving containers with a large opening surface and pouring containers with a small height, pouring into objects with a small opening, such as cell culture flasks, poses problems. Because of the large movement of the liquid exit point, the liquid will only reach the receiving container for specific pouring angles. This issue is visualized in fig. 3.21.

More advanced approaches for trajectory planning usually consider the entire robot movement, including approaching the receiving container from longer distances [74, 96].

The suggested new approach for pouring liquids into containers with a small opening surface is based on human movement. When manually executing the task, humans usually first move the opening of the pouring container close to the opening of the receiving container. Once the liquid starts pouring, the liquid exit point remains constant and is used as a center of rotation. Consequently, the pouring container rotates around the liquid exit point.

To execute that movement, the TCP changes both in position and orientation. Assuming a movement in the xy plane, the x and y coordinates must be calculated based on the rotation angle θ around the liquid exit point. From now on, the liquid exit point will be referred to as the center of rotation (CoR).

The start position for the cell culture flask is horizontal since there will never be more than 150 mL inside the container during the process. As a result, the fill level always remains below the lowest opening point of the flask. When gripping the flask horizontally, the TCP will be in the middle between the upper and lower flask surface. For this case, it is valid that $TCP_y = CoR_y$

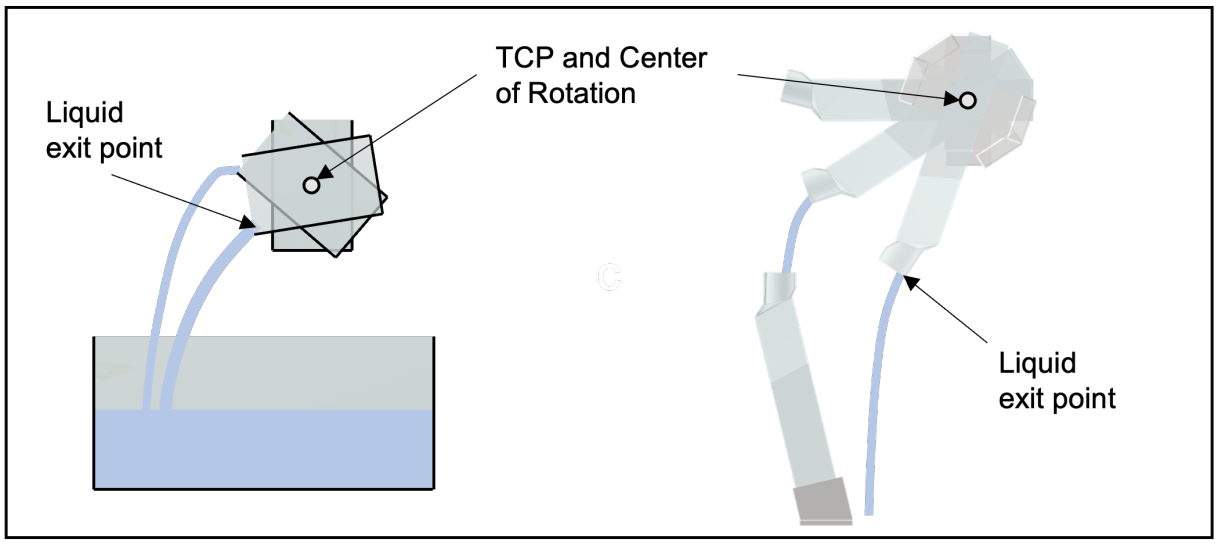


Figure 3.21: Visualization of the issue with pouring movements only considering a wrist rotation. Left: Successful pouring only using rotation because of a receiving container with a large opening surface and a pouring container with a small height. Right: Liquid is spilled when only rotating the cell culture flask around the TCP.

(see fig. 3.23). Therefore, the TCP position depending on the rotation angle θ is calculated as presented in eq. (3.6) and eq. (3.7).

$$TCP_x(\theta) = TCP_{x,start} - l * (1 - cos(\theta)) \tag{3.6}$$

$$TCP_y(\theta) = TCP_{y,start} + l * sin(\theta) \tag{3.7}$$

where $l = |TCP_{x,start} - CoR_x|$. An exemplary simulation scene of the cell culture flask executing the described movement can be seen in fig. 3.22.

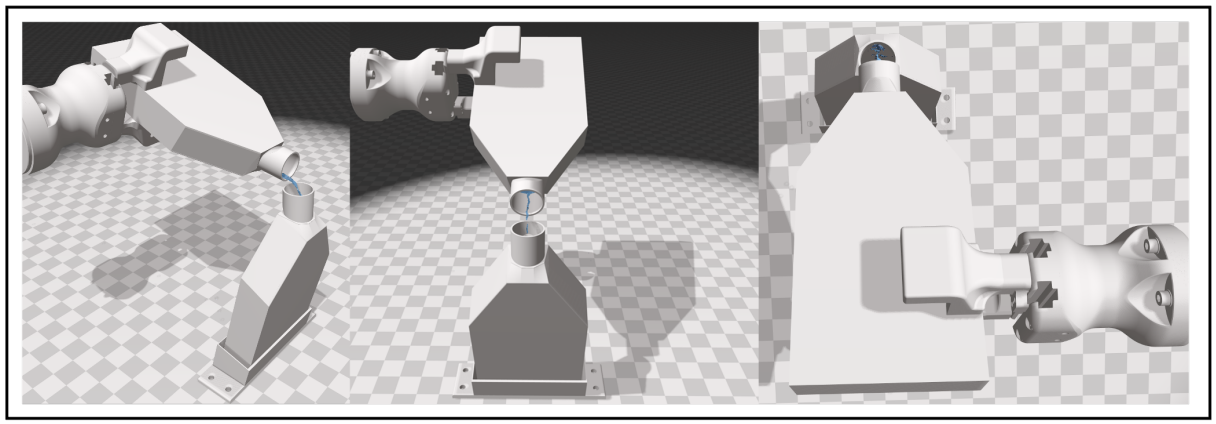


Figure 3.22: An exemplary simulation scene of the cell culture flask executing the new pouring movement.

For the washing solution/media bottle, $TCP_x \neq CoR_x$ (see fig. 3.23). From the 3D model, we get the distance between TCP_{start} and CoR as $l = 88.99mm$ and $\beta = 46.11°$. In a horizontal position, the CoR can not be placed close enough to the opening surface of the receiving flask since the gripper would touch the lower part of the flask. Therefore, the starting orientation of the

bottle is rotated by $\alpha = 14.5°$, which is the same angle as the rotation of the flask holder. From there, the starting angle between TCP_{start} and CoR is calculated as $\alpha_{start} = \beta - \alpha = 31.61°$. TCP_x and TCP_y for a specific time step with the angle of rotation θ can then be calculated as shown in eq. (3.8) and eq. (3.9). These equations can be used to calculate the pouring trajectory for any new pouring container for any starting position TCP_{start} and initial rotation α.

$$\left.\begin{aligned} TCP_x(\theta) &= TCP_{x,start} - l * cos(\alpha_{start}) + l * cos(\alpha_{start} - \theta) \\ TCP_y(\theta) &= TCP_{y,start} + l * sin(\alpha_{start}) - l * sin(\alpha_{start} - \theta) \end{aligned}\right\} for\ \theta \leq \alpha_{start} \tag{3.8}$$

$$\left.\begin{aligned} TCP_x(\theta) &= TCP_{x,start} - l * cos(\alpha_{start}) + l * cos(\theta - \alpha_{start}) \\ TCP_y(\theta) &= TCP_{y,start} + l * sin(\alpha_{start}) + l * sin(\theta - \alpha_{start}) \end{aligned}\right\} for\ \theta > \alpha_{start} \tag{3.9}$$

The suggested new approach for the pouring movement is visualized in fig. 3.23. It also demonstrates the initial rotation of the bottle. An exemplary pouring movement of the bottle in the simulation is shown in fig. 3.24. The final simulated pouring sequences include the movement of the pouring container until θ_{max} is reached and the inverse movement back to the start position.

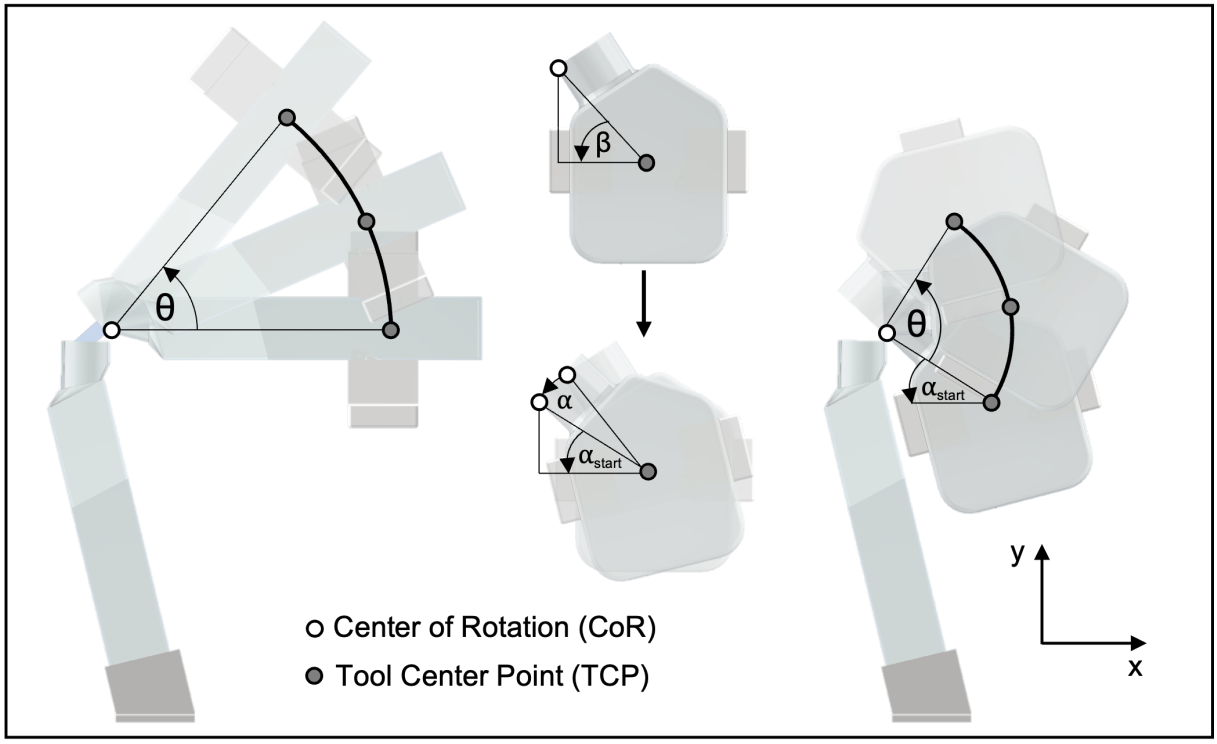


Figure 3.23: Visualization of the pouring movement. θ represents the pouring angle (equivalent to the rotation of the pouring container), β the angle between the TCP and the CoR of the bottle in the horizontal position, α the rotation of the bottle to the same angle as the receiving flask ($\alpha = 14.5°$), and α_{start} the starting angle between the TCP and the CoR. Left: Pouring with the cell culture flask. Center: Explanation of the starting position of the bottle. Right: Pouring with the media/washing solution bottle.

Videos of the simulated pouring movements captured from different viewing angles can be found in the the pouring simulation subfolder of the GitHub repository of the project.

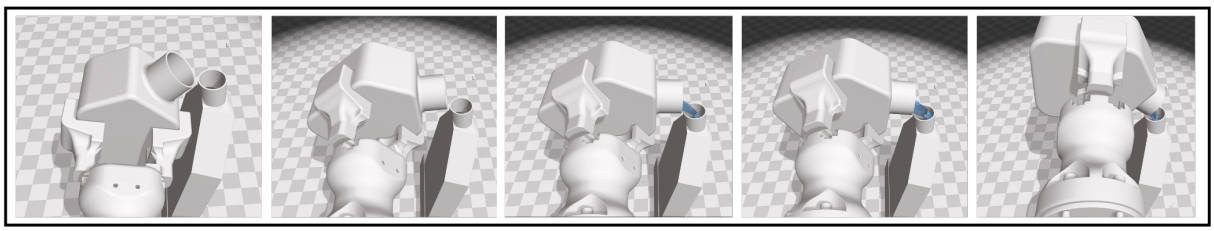


Figure 3.24: An exemplary simulation scene of the media/washing solution bottle executing the new pouring movement.

Parameter Space and Problem Formulation

Using mathematical approaches to find the required robot movement usually involves using simplified models. The calculations are often limited to varying one parameter of the pouring action. Most of the time, the rotation angle around the TCP is used (e.g., [64, 68]). By using simulation, more parameters of the pouring movement can be varied, which results in a more comprehensive and realistic approach to understanding, designing, and optimizing robotic liquid pouring tasks. More detailed output volumes can be achieved by exploring a wider range of the parameter space.

In this study, the objective of the simulation is to pre-evaluate the outcomes of various pouring movements by manipulating different parameter inputs. The goal is to use these simulation results to identify the optimal robot movement for specific input parameters in real-world scenarios.

The complete set of parameters is divided into two categories: *action parameters* and *simulation parameters*. The *action parameters* include all variables that specify the control of the pouring movement as described above. This contains the angular velocity, the starting position and orientation, the maximum rotation angle (stop angle), the duration of the stop at the maximum rotation angle (stop time), the pouring container, and the gripping position of the container. The *simulation parameters* are defined as input variables that must be given to the simulator. This includes the number of particles at the start (start volume), the particle radius, and the liquid parameters (adhesion, cohesion, surface tension, viscosity).

Certain variables are selected to remain constant to reduce the complexity of the parameter space and minimize the time required for pre-running the simulations. These include the liquid parameters, which are set to the values mentioned above, the particle radius, the angular velocity (0.03 rad/s), and the starting position and orientation relative to the receiving container (as visualized in fig. 3.23). The gripping positions were chosen iteratively using the robot and the 3D-printed gripper fingers and were also kept constant throughout the simulations.

We use $\mathbf{p} \in \mathbf{P}$ to denote a set of parameters, $\mathbf{p}^{(i)}$ to denote the i^{th} dimension of the set of parameters, and $\mathbf{p}^*$ to represent an optimal set of parameters. Hence, the combination of the remaining 2D *action parameter* space and the 2D *simulation parameter* space results in a reduced 4D parameter space $\mathbf{p}$. This consists of the pouring container $\mathbf{p}^{(0)} = c$, the start volume in the pouring container $\mathbf{p}^{(1)} = V_{start}$, the stop angle $\mathbf{p}^{(2)} = \theta_{stop}$, and the stop time $\mathbf{p}^{(3)} = t_{stop}$.

As mentioned, the cell culture flask (T175 EasYFlask) and the washing solution/media bottle (Gibco Bottle, 500 mL) were selected as pouring containers. Therefore, $\mathbf{p}^{(0)}$ has a cardinality of 2. The range and step size of the other parameters were chosen iteratively using different test movements and the knowledge of the cell culture automation process. For the cell culture flask, the start volume varies from 35 to 150 mL in 5 mL steps ($\left|\mathbf{p}^{(1)}(c=0)\right| = 25$), the stop angle from 2° to 50° in 2° steps ($\left|\mathbf{p}^{(2)}(c=0)\right| = 25$), and the stop time from 0.2 s to 1.8 s ($\left|\mathbf{p}^{(3)}(c=0)\right| = 5$).

The start volume for the bottle ranges from 50 mL to 500 mL in increments of 10 mL, resulting in a cardinality of 46 ($\left|\mathbf{p}^{(1)}(c=1)\right| = 46$). The stop angle varies from 30° to 90° in 4° steps ($\left|\mathbf{p}^{(2)}(c=1)\right| = 31$). Similarly, the stop time ranges from 0.2 s to 1.8 s ($\left|\mathbf{p}^{(3)}(c=1)\right| = 5$). Using a grid search approach, this results in a total number of scenes of 6,805. The described parameter space is visualized in fig. 3.25. The simulation of the pours with the cell culture flask took around 52 hours. The simulation process for the media/washing solution bottle required approximately 140 hours to complete. A timelapse video of 12 consecutive pouring movements with different parameters can be seen here.

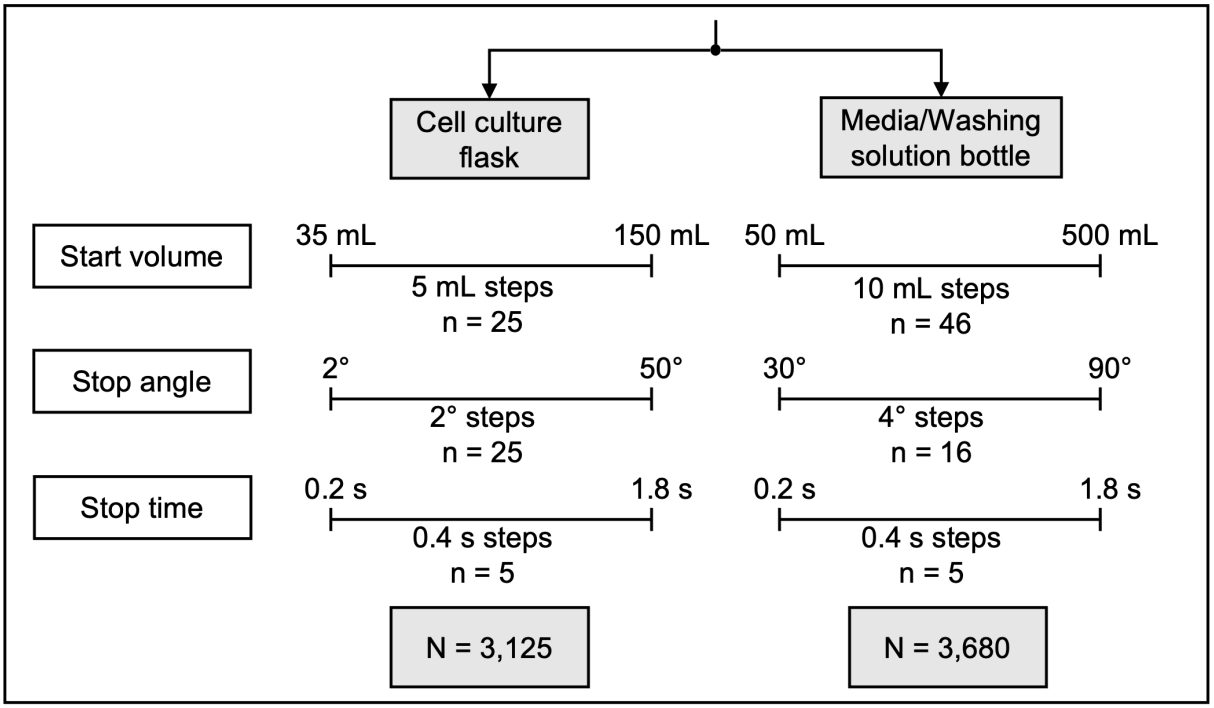


Figure 3.25: Parameter space for the simulation of liquid pouring.

Based on the given real-world parameters, which are the pouring container, the start volume, and the desired volume to be poured V_{goal}, the best-fitting simulated scene with an optimal set of parameters $\mathbf{p}^*$ needs to be selected. For that, the start volume, the received volume, and the spilled volume of each scene are stored. The resulting cost function is defined in eq. (3.10).

$$C(\mathbf{p}) = C_{start} + C_{received} + C_{spill} \tag{3.10}$$

C_{start} is the magnitude of the difference between the start volume in the simulation scene and the start volume in the real-world scenario. $C_{received}$ is the magnitude of the difference between the received volume in the receiving container in the simulation and the desired real-world volume to be received. C_{spill} is the volume of spilled liquid in the simulation. The overall goal is to determine an action with parameters $\mathbf{p}^*$ that minimizes the cost $C(\mathbf{p})$. By iterating through the results of the simulation with the selected pouring container, real-world start volume, and desired output volume, this minimum can be found.

3.3.3 Simulation-to-Reality Transfer for Robot Movement

To achieve the final goal of the pouring task, the simulated movement must be transferred to a real robot. In the context of transferring the policy to a robot, the role of the pouring container changes to that of the manipulator's end effector (TCP). This shift occurs because the grippers

hold and manipulate the pouring container, making it functionally equivalent to an end effector during the motion planning process in the simulation. The coordinates of the TCP and the rotation angle of the pouring container at every time step for every scene are recorded and stored.

However, it is essential to note that these positions and orientations are defined with respect to the coordinate system of the simulation, where the origin is located at the ground plane in the center of the flask holder. By subtracting the starting position coordinates from the TCP, the origin can be moved to the starting position of the pouring movement. The new poses represent the movement of the robot's end effector with respect to the tool's coordinate system.

However, the positions sent to the used UR5e through the RTDE need to be defined with respect to the base coordinate system. Therefore, a pose transformation must be applied to every pose to make them relative to the base coordinate system. For the transformation, the initial start tool position and orientation (before rotating around α) in the base coordinate system is used as the origin of the tool's coordinate system ($Pose_{start}$). The transformation function begins by extracting the position (X, Y, Z) and rotation (R_x, R_y, R_z) components from $Pose_{start}$ and the pouring pose in the tool's coordinate system $Pose_{tool}$ at a specific timestep. The rotation components are initially given as rotation vectors and are first converted to Euler angles. It then calculates the rotation matrices for both poses (R_{tool}, R_{start}). The rotation matrices are multiplied to obtain the combined rotation matrix, which is then converted back to a rotation vector. The translation of $Pose_{tool}$ is rotated by R_{start} and added to the translation of $Pose_{start}$ to achieve the combined position. Finally, the resulting combined position and rotation components are returned as the transformed pose, which represents the pose of the tool (which is equivalent to the pouring container) relative to the base coordinate system.

Another problem to solve was transferring the movement speed from the simulation to the real world. The pose in the simulation is stored at every timestep with a defined frequency. However, the commands sent to the robot need to include velocity. To get this value in meters per second, the distance between two positions at two consecutive timesteps is calculated in meters and then multiplied by the given frequency used in the simulation. For a continuous robot movement, two paths with a blending radius of 0.001 m are created. The first path represents the movement from the start position to the stop angle. There, the movement is stopped for the defined stop time of the sample. After, the second path is executed, which moves the pouring flask back from the stop angle to the start position.

The advantage of performing the movement in the simulation relative to the tool's coordinate system and subsequently converting it to the base coordinate system is that the same movement can be executed from various starting positions. This flexibility is particularly crucial for scenarios involving multiple pouring sequences at different positions, such as cell passaging.

All pouring experiments were executed with the same UR5e robot, Robotiq Hand-E gripper, and 3D-printed gripper fingers as used for the cell culture system prototype. The specifications are described in subsection 3.1.1. The same connection setup was used as mentioned in subsection 3.1.2. A Mettler Toledo XSR2002S balance was used to measure the amount of liquid before the pouring sequence inside the pouring container and after the pouring sequence inside the pouring and receiving containers.

A random selection of 50 samples was made from the pouring results obtained through simulations for both the cell culture flask and the bottle. The experiments for pouring from the cell culture flask were executed as follows:

1. Load the parameters for the sample, which include the start volume and the pouring trajectory from the simulation, and the predicted received and spilled volumes.

2. Manually fill the required amount of liquid into the pouring container by measuring the weight using the balance.
3. Manually place the empty and open receiving flask into the flask holder and the open pouring flask into the start flask holder.
4. Convert the pouring trajectory from the tool coordinate system to the base coordinate system.
5. Grip the pouring flask at the same location as in the simulation with the gripper fingers, and move the robot to the start pose where the pouring flask is above the receiving container.
6. Execute the pouring movement.
7. Open the gripper and manually remove the pouring and receiving flask.
8. Measure the remaining and the received volume with the balance, and write down the results.
9. Restart the procedure.

The procedure of the pouring experiments is visualized in fig. 3.26. A video of a demonstration of the experiment procedure of pouring from the media/washing solution bottle is provided in the simulation subfolder of the GitHub repository of the project. Each execution takes around 150-240 seconds, of which the robot only spends 5-30 seconds for the actual pouring movement. The rest of the time is spent on weighing and cleaning. In total, around six hours were spent performing real pouring executions.

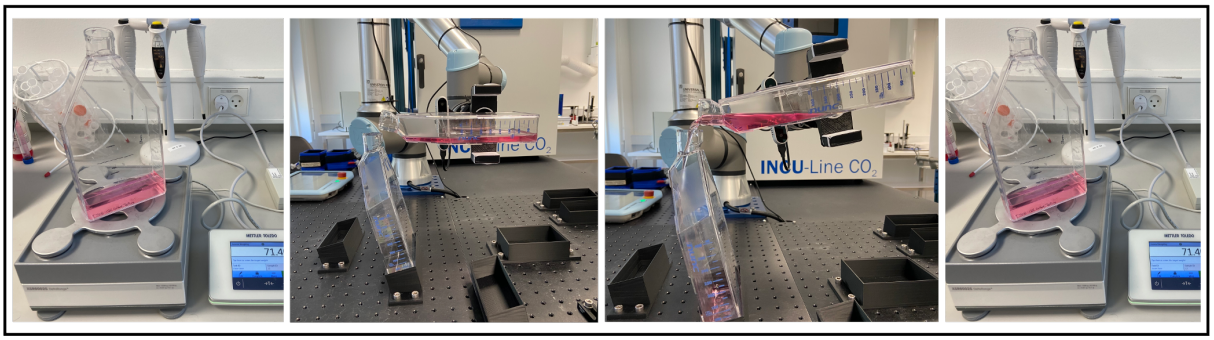


Figure 3.26: Visualization of the pouring experiment procedure and setup. From left to right: (1) Manual filling of the pouring container with the starting volume. (2) Robot holding the pouring container at the start position. (3) Execution of the pouring movement. (4) Measuring the received volume in the receiving container and the remaining volume in the pouring container.

In the evaluation of the simulation-to-reality experiments, three different metrics are used: The root-mean-squared error ($RMSE$, eq. (3.11)), the mean absolute percentage error ($MAPE$, eq. (3.12)), and the R^2 score (coefficient of determination, eq. (3.13)) (all based on [94]). The results are calculated based on the real and simulated received volumes in the receiving container and the real and simulated spilled volumes. In the following, V_{real} stands for volumes measured in the real-world experiments, V_{sim} for volumes from the simulation results, n for the number of samples, and $\bar{V}_{real}$ for the mean of the real volumes.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(V_{sim,i} - V_{real,i})^2} \tag{3.11}$$

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{V_{sim,i} - V_{real,i}}{V_{sim,i}} \right| \tag{3.12}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (V_{real,i} - V_{sim,i})^2}{\sum_{i=1}^{n} (V_{real,i} - \bar{V}_{real})^2} \tag{3.13}$$

4 Results and Evaluation

This chapter provides a detailed analysis of the outcomes obtained from the computer vision-based approach for the detection and volume estimation of liquids (section 4.1), the pouring simulation for the autonomous robotic pouring of liquids (section 4.2), and the overall system performance (section 4.3). The focus lies on the performance, accuracy, and practicality of the developed approaches within the context of cell culture automation.

4.1 Results of Liquid Detection and Volume Estimation

This section presents the results of the models for vision-based detection and depth estimation of transparent vessels and liquids, as well as the estimation of the volume of liquids as described in section 3.2.

4.1.1 Segmentation and Depth Estimation

The results presented in this section are based on the model trained with the parameters described in subsection 3.2.2. The depth maps are normalized and shown as a color map for better visualization. First, the model was tested on test images from the *TransProteus* dataset, which are stored in the testing subfolders of the dataset.

Three examples of the masks and depth maps predicted by the model based on a single RGB input image are shown in fig. 4.1.

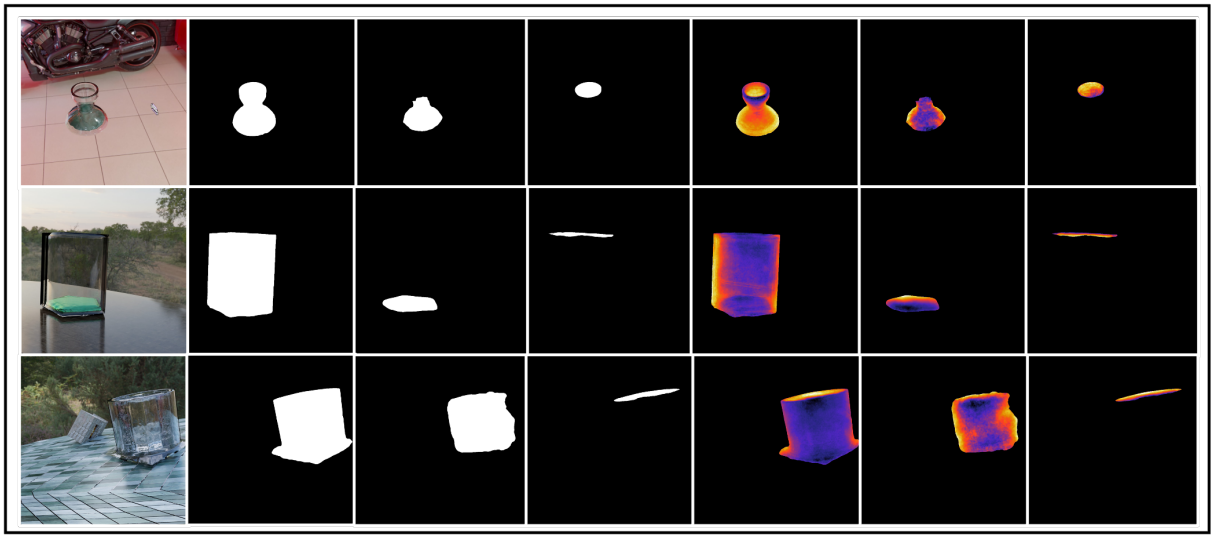


Figure 4.1: Examples of the masks and depth maps predicted by the trained segmentation and depth estimation model. From left to right: (1) Input RGB image. (2) Vessel mask. (3) Liquid Mask. (4) Opening Mask. (5) Vessel depth map. (6) Liquid depth map. (7) Opening depth map.

The model performance was evaluated on 1500 samples from the test set. The results of the segmentation of liquids, transparent vessels, and vessel openings are given in table 4.1. The vessel region is predicted with high accuracy ($IoU > 96\%$), with high precision and recall. These results are similar to the ones shown in [16] ($IoU = 96\%$, $Precision = 99\%$, $Recall = 98\%$). The stored model after 20 epochs of training has an $IoU < 90\%$, while after 40 epochs, a similarly high accuracy was achieved ($IoU = 94\%$).

The vessel opening mask was predicted with similar accuracy as the vessel mask ($IoU = 94\%$). The IoU is equal to the one achieved in [16], with slightly higher precision (98% vs.

96%) and lower recall (96% vs. 98%). By only using the liquid content subset, the liquid mask was predicted with an IoU of 85%, which is slightly higher than the one achieved by training on the whole dataset in [16] ($IoU = 84\%$).

Table 4.1: Results of semantic segmentation on 1500 simulated images from the *TransProteus* test set for vessel, liquid, and vessel opening.

	IoU	$Precision$	$Recall$
Vessel Mask	0.96	0.99	0.97
Liquid Mask	0.85	0.91	0.93
Vessel Opening Mask	0.94	0.98	0.96

The results of the depth estimation of vessels, liquids, and vessel openings are given in table 4.2. As noted in subsection 3.2.2, only the scale-invariant log RMSE is used as a metric. The depth estimation of the vessel shows better results than the one for liquid and vessel opening. Since this is the first approach for depth estimation on this dataset, no baselines for comparison exist.

Table 4.2: Results of depth estimation on 1500 CGI from the *TransProteus* test set for vessel, liquid, and vessel opening.

	Vessel	Liquid	Opening
$RMSE\ (log, scale - invariant)$	0.031	0.034	0.043

Evaluating the depth estimation of liquids on real images is not possible since no samples exist or can be created. Since it is not possible to remove the vessel while keeping the liquid in the same place, no 3D scans of the liquid inside the vessel can be done. However, the prediction can be made on real images of vessels containing liquids taken in the research laboratories to evaluate the results qualitatively. For that, the predicted segmented depth maps of images from the *LabLiquidVolume* dataset described in 3.2.3 based on the trained models after different epochs are compared. Two samples are visualized in fig. 4.2. The analysis reveals that in the case of vessel prediction, the segmentation mask achieves a high level of accuracy within a few epochs. Conversely, the depth prediction, which provides a visualization of the object's shape, produces more detailed results after a longer training period. For the liquid, the shown segmentation masks achieve high accuracy after 55 epochs. The depth predictions of the liquid get more precise after more epochs. However, the edges of the liquid top surfaces are not clearly visible.

As mentioned in subsection 2.2.2, liquids and transparent objects often appear for a camera as noisy or distorted approximations of the surfaces that lie behind them and do not form disjoint boundaries with their environment. Especially for depth cameras, liquids and transparent objects pose challenges due to their unique optical properties. When light passes through a transparent material with different refractive properties, such as a liquid, it undergoes refraction. Refraction causes the path of the light rays to change direction, making it difficult for the depth camera to estimate the object's depth. Transparent objects can reflect light or create specular highlights, further complicating depth estimation. The reflections and highlights can confuse the depth camera by introducing additional sources of light that are not directly related to the object's depth. Consequently, many values in the recorded depth maps are missing.

To quantify that problem, the predicted segmentation masks were overlaid on the captured depth maps obtained from the camera. This allowed for the computation of the proportion of missing depth values within the segmented regions. Specifically, for the vessel depth maps,

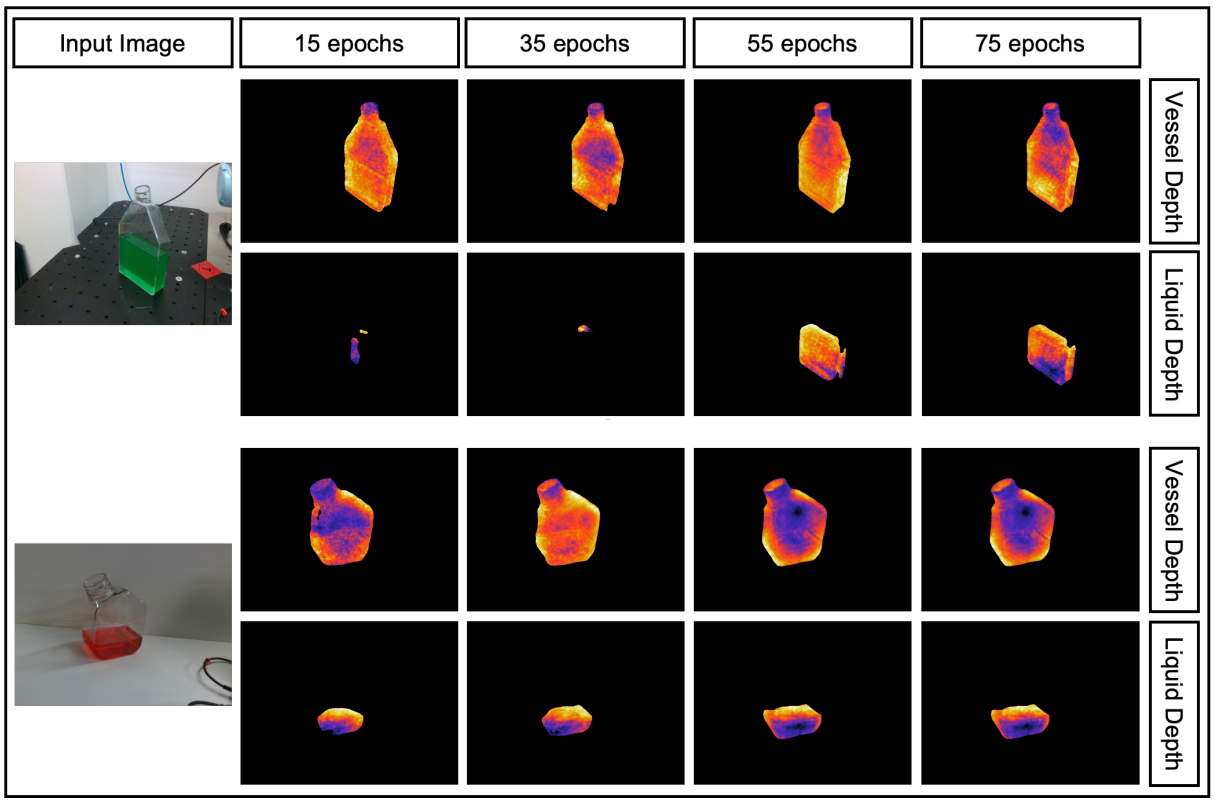


Figure 4.2: Examples of the masks and depth maps predicted by the trained segmentation and depth estimation model. From left to right: (1) Input RGB image. (2) Vessel mask. (3) Liquid mask. (4) Opening mask. (5) Vessel depth map. (6) Liquid depth map. (7) Opening depth map.

an average of 49.8% of depth values were found to be missing. Similarly, for the liquid depth maps, an average of 52.1% of depth values were missing. To illustrate that, two samples of the *LabLiquidVolume* dataset are presented in fig. 4.3. It shows the predicted segmented depth maps of liquid and vessel as well as the segmented depth map recorded by the depth camera. In addition, it should be noted that the model predicts the undistorted liquid shape, which is the liquid as it would be viewed if the vessel was not there. This would not be possible using the depth map recorded by the camera.

As explained, scale-invariant losses focus on learning relative depth, leading to accurate relative depth predictions. This means that it does not consider the scale discrepancy between the ground truth annotation and the output predicted by the model. In other words, two identical liquids at different distances from the camera can have the same predicted depth map. For an accurate liquid volume prediction, the global scale is essential. This raises the question of whether global depth is learned from training in a scale-invariant manner. To perform a quantitative evaluation, 221 images from the *LabLiquidVolume* dataset, specifically featuring the Duran 250 mL vessel, were randomly selected. These images were manually classified into three distinct classes: *close*, *mid-range*, and *far* (referred to as true liquid distance class). After, the mean of the predicted segmented liquid depth map for every sample was calculated. The samples were sorted by the resulting mean value and put into the same classes with the same amount of samples per class (e.g., the 76 samples with the lowest mean liquid depth into the class *close*). The results of this evaluation are shown in the confusion matrix in fig. 4.4. It can be seen that for most of the samples, the predicted liquid distance class is correct (81.90%). No samples were mispredicted from *close* to *far* or from *far* to *close*.

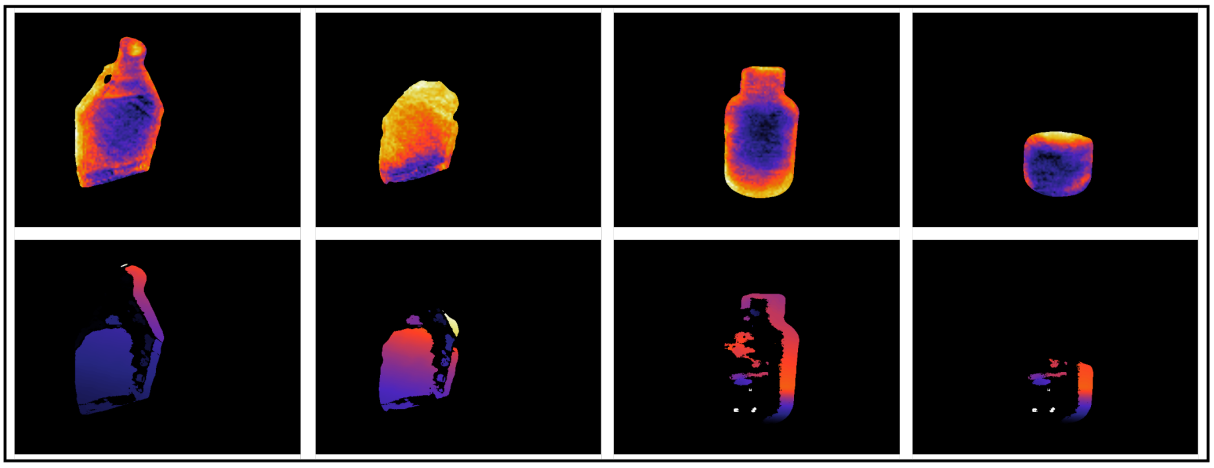


Figure 4.3: Two samples of the *LabLiquidVolume* dataset to illustrate the problem of missing depth values. The first row shows the predicted segmented depth maps, while the second row shows the segmented depth map recorded by the depth camera. From left to right: vessel depth map cell flask 400 mL, liquid depth map cell culture flask 400 mL, vessel depth map Duran 250 mL, liquid depth map Duran 250 mL.

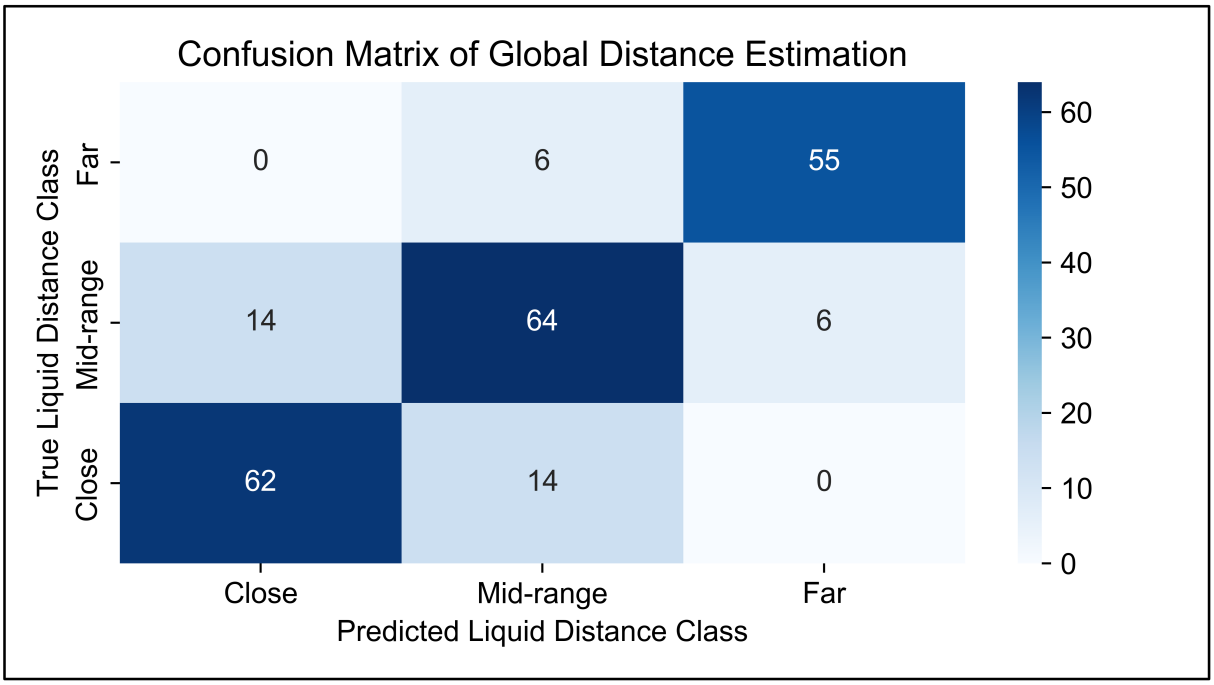


Figure 4.4: Confusion matrix of the global depth evaluation of 221 samples from the *LabLiquid-Volume* dataset.

This means that the liquid depth prediction takes the global scale into account to some degree. A more detailed evaluation would require more manual classifications or manually annotated liquid depth maps.

The segmentation prediction can be easier evaluated on real images. To get results specifically for the containers present in a research laboratory and in the cell culture process, images from the generated *LabLiquidVolume* dataset are used. Ten samples per container were randomly selected, and the liquid and vessel segmentation masks were manually annotated using a custom-made script. A visualization of the interface of the manual annotation can be seen in fig. B.21.

For the evaluation of the manually annotated images, the same metrics as for the CGI are used. The results for every vessel and summarized for all glass and plastic objects are shown in table 4.3. The vessel region is accurately predicted for all objects ($IoU > 80\%$). The vessels with the lowest IoU for vessel segmentation are the 400 mL Cell Culture Flask and the 15 mL Tube. For vessel segmentation, the difference in accuracy for glass and plastic objects is small.

However, there is a gap between glass and plastic objects for liquid segmentation. While the liquid region inside glass vessels is predicted with an IoU of 91.3%, it is significantly lower for liquids inside plastic objects ($IoU = 84.3\%$).

Table 4.3: Results of semantic segmentation for liquids and vessels of 120 manually annotated images from the generated *LabLiquidVolume* dataset. Ten samples were randomly selected per vessel.

	Liquid			**Vessel**		
	IoU	$Recall$	$Precision$	IoU	$Recall$	$Precision$
Cell Flask 160mL	0.844	0.982	0.858	0.959	0.976	0.981
Cell Flask 400mL	0.744	0.984	0.754	0.887	0.975	0.908
Duran 100mL	0.895	0.956	0.934	0.905	0.916	0.987
Duran 250mL	0.917	0.956	0.957	0.936	0.946	0.988
Duran 500mL	0.914	0.948	0.960	0.927	0.937	0.988
Gibco 125mL	0.797	0.980	0.814	0.921	0.959	0.959
Gibco 500mL	0.890	0.985	0.904	0.942	0.975	0.965
Pyrex 100mL	0.925	0.967	0.955	0.941	0.952	0.987
Storage Bottle 250mL	0.934	0.969	0.962	0.950	0.964	0.984
Storage Bottle 500mL	0.928	0.963	0.962	0.949	0.959	0.989
Tube 15mL	0.759	0.948	0.793	0.819	0.913	0.888
Tube 50mL	0.842	0.919	0.907	0.931	0.959	0.969
Glass Objects	**0.913**	**0.957**	**0.952**	**0.928**	**0.938**	**0.988**
Plastic Objects	**0.843**	**0.966**	**0.870**	**0.920**	**0.961**	**0.956**
All Objects	**0.866**	**0.963**	**0.897**	**0.923**	**0.953**	**0.966**

The 400 mL Cell Flask ($IoU = 74.4\%$), the 15 mL Tube ($IoU = 75.9\%$), and the Gibco 125 mL ($IoU = 79.7\%$) show the worst results for the liquid segmentation. Those objects are all made out of plastic. The vessels made out of glass all have an $IoU > 0.89$ for the liquid region. The best prediction for the liquid region was made for the Pyrex 100 mL and the 250 mL and 500 mL Storage Bottles.

The aggregated outcomes from the analysis of the 120 real images indicate relatively inferior performance across all three testing metrics in vessel segmentation compared to the evaluation conducted on the *TransProteus* test set. For the liquid segmentation, the testing on the real images shows better results for IoU (0.87 vs. 0.85) and $Precision$ (0.96 vs. 0.91) and worse results on $Recall$ (0.90 vs. 0.93) compared to the CGI from the *TransProteus* test set.

The visualization in fig. 4.5 shows the two samples with the best predictions and the two with the worst predictions of the liquid region. It includes both the predicted and manually annotated segmentation masks for liquids and vessels. The first row represents a 400 mL Cell Culture Flask with an IoU for the liquid region of 67.6% and an IoU for the vessel of 75.1%. The second row illustrates a Gibco 125 mL bottle, with an IoU score of 23.9% for the liquid region and an IoU score of 81.7% for the vessel. The predictions for the Gibco 500 mL and the Storage Bottle 500 mL in rows three and four have IoU scores of $>$ 96% for both liquid and vessel regions.

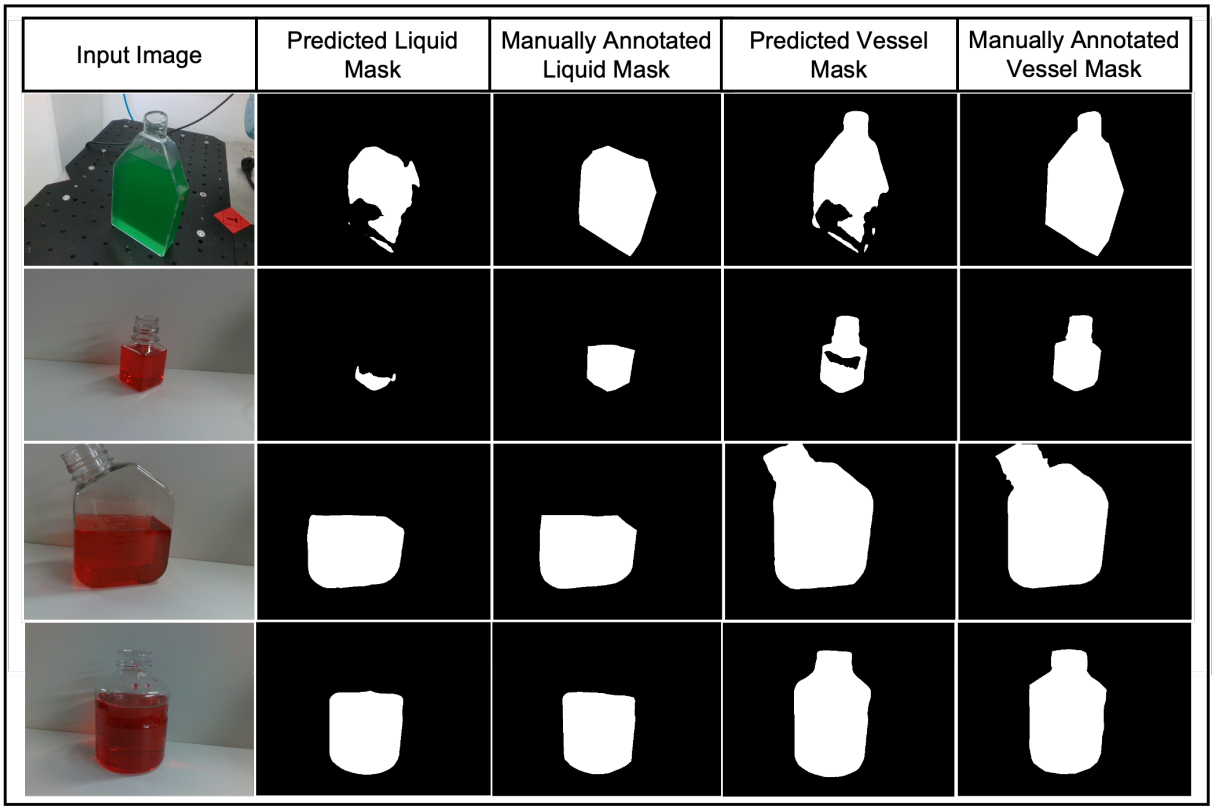


Figure 4.5: Visualizations of the two samples with the best predictions and the two samples with the worst predictions of the liquid region from the selected 120 images from the *LabLiquidVolume* dataset. From top to bottom: (1) Cell Culture Flask 400 mL. (2) Gibco 125 mL. (3) Gibco 500 mL. (4) Storage Bottle 500 mL.

4.1.2 Liquid Volume Estimation

The results presented in this section are based on the models trained with the parameters and data from the *LabLiquidVolume* dataset as described in subsection 3.2.3.

The distribution of real and predicted liquid volumes for the model trained on the segmented depth maps of fluid and vessel only is shown in fig. 4.6. The colors of the data points indicate the vessel present in the image. The dashed line represents a perfect prediction, where the real volume equals the predicted volume. A histogram of the distribution of the $RMSE$ can be seen in fig. B.3. The average amount of liquid inside the vessels in the test set is 185.39 mL.

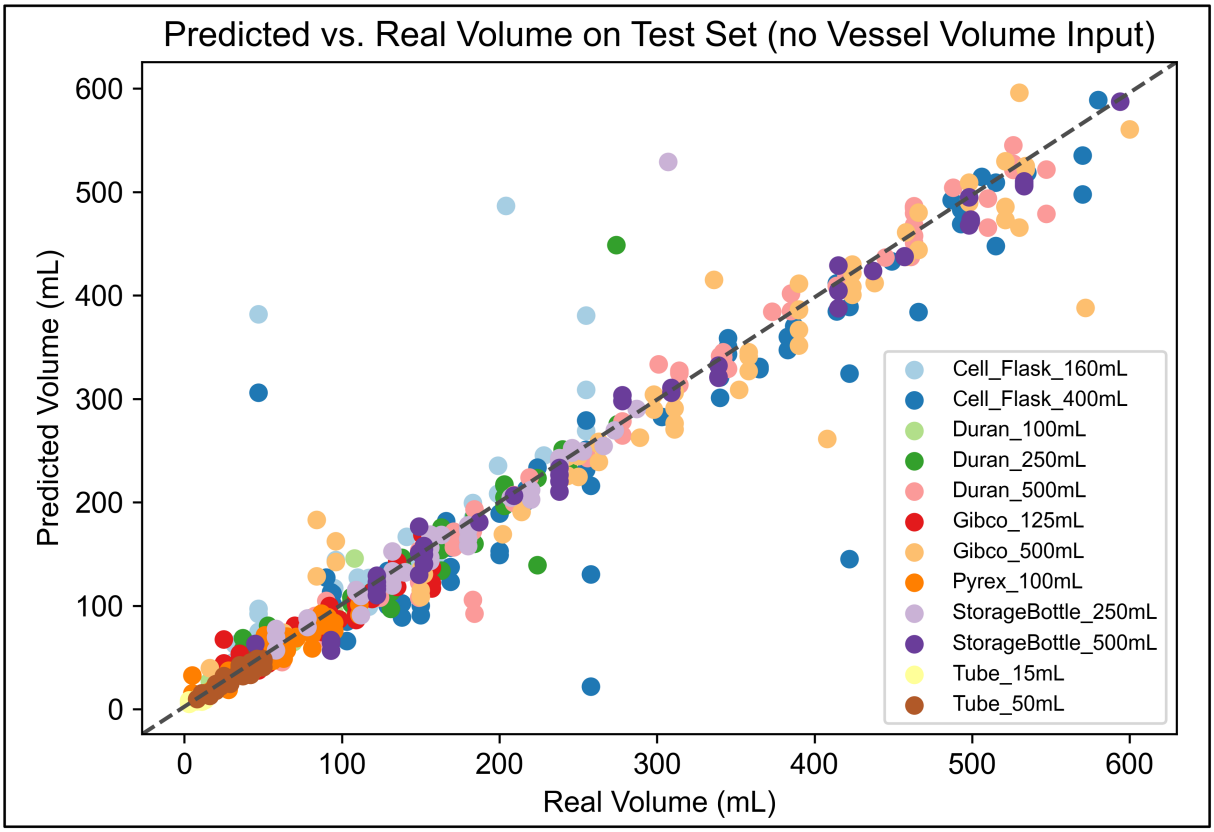


Figure 4.6: Scatter plot of the real and predicted liquid volumes for images from the test set for the model trained only on the segmented depth maps.

The analysis indicates that the model's performance shows an $RMSE$ of 37.90 mL, a mean absolute percentage error of 16.96%, and an R^2 score of 0.94 on the test set. However, the predicted volume exhibits significant discrepancies from the actual volume for some samples, with a maximum error of 320.20 mL. These errors occur in both positive and negative directions.

The equivalent plot for the model trained on the segmented depth maps and the volume of the vessel is shown in fig. 4.7. The histogram of the distribution of the $RMSE$ can be seen in fig. B.4.

The results show an overall $RMSE$ of 17.83 mL, a $MAPE$ of 9.39%, and a R^2 score of 0.99 on the test set. Fewer samples with a predicted volume far from the real volume can be seen, with a maximum error of 118.70 mL.

Detailed results of both models for each vessel are listed in table 4.4. It can be observed that the model which includes the vessel volume has better results for $RMSE$ and $MAPE$ for every vessel. The objects with the most precise results are the Duran 500 mL and the 250 mL and 500 mL Storage Bottles. The vessels with the highest $MAPE$ for both models are The Tube 15 mL, the Cell Flask 160 mL, and the Pyrex 100 mL.

The negative R^2 score for the Cell Flask 160 mL for the segmented depth maps model implies that the model's predictions are performing worse than a horizontal line representing the mean of liquid volume for that particular vessel. The relevant vessels for the cell culture automation show satisfactory results for the segmented depth maps with vessel volume model (Cell Flask 400 mL: $MAPE$=12.4%, Gibco 500 mL: $MAPE$=10.1%).

The five samples with the highest $MAPE$ were extracted and visually inspected for both mod-

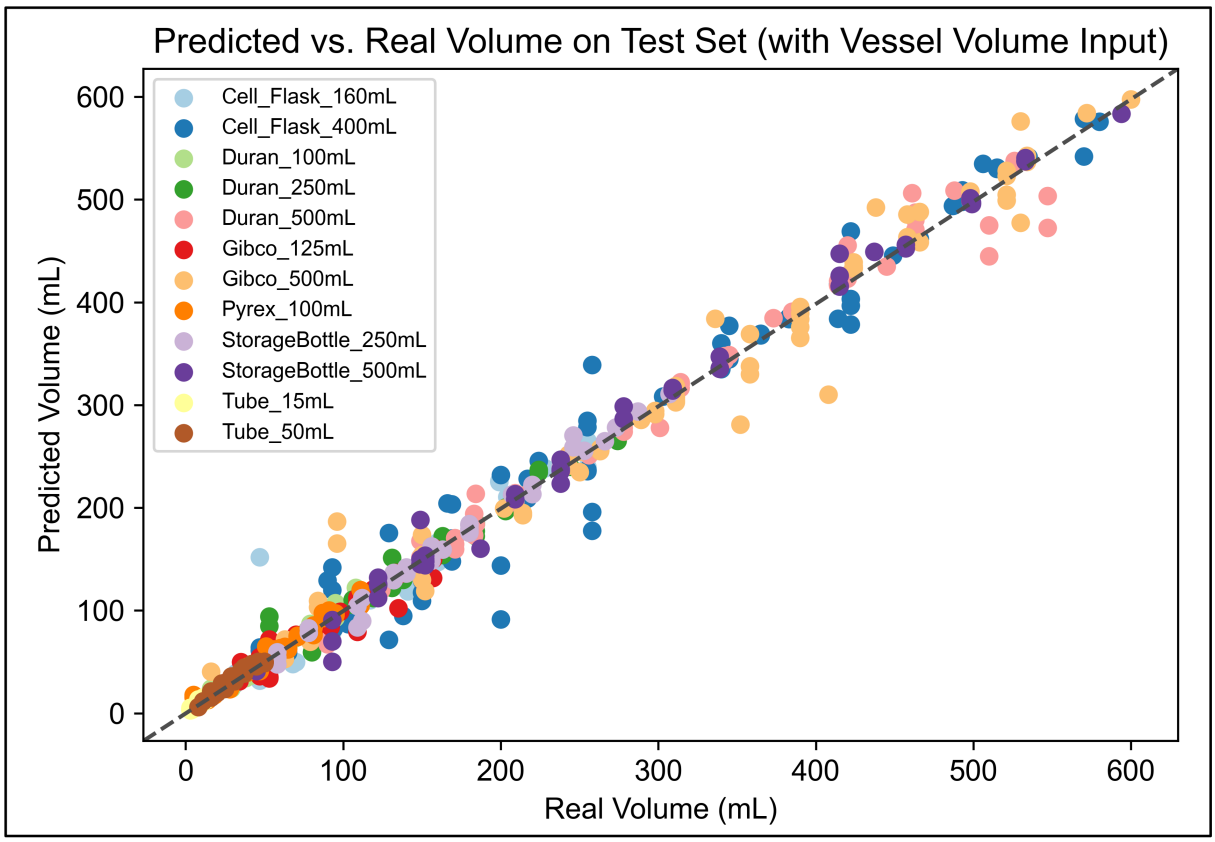


Figure 4.7: Scatter plot of the real and predicted liquid volumes for images from the test set for the model trained on the segmented depth maps and the volume of the vessel.

els. This showed that most bad liquid volume predictions were already based on non-accurate segmentation and depth estimation results, especially regarding the liquid. Three samples with the highest $MAPE$ predicted with the model trained only on the segmented depth maps are visualized in fig. 4.8. The figure shows the original RGB images, the predicted segmented depth maps of liquids and vessels, and the predicted and real liquid volumes.

To quantify this inspection, the Pearson's correlation coefficient between the IoU for liquid segmentation from the manual annotation for each vessel (table 4.3) and the $MAPE$ of the liquid volume estimation for each vessel was calculated. The analysis revealed a moderate negative correlation coefficient of -0.694. This finding suggests that higher IoU values, indicating better segmentation performance, were associated with lower $MAPE$ values, indicating improved accuracy in liquid volume estimation. In other words, when the model excelled at accurately segmenting the liquid volumes, it also made more precise volume predictions.

Besides the two mentioned main models, the two additional models (Segmentation Mask Model, Scaled Depth Maps Model) were analyzed to evaluate the usefulness of the presented approach. The results of this comparison are listed in table 4.5. It shows the final $RMSE$ for the training data after 200 epochs and the mentioned metrics for the test set. The results indicate that the segmented depth maps and vessel volume model outperforms other approaches across all evaluation criteria. The model using only the segmented depth maps as input yields slightly better results compared to the model using scaled segmented depth maps. However, omitting depth information entirely and solely relying on liquid and vessel segmentation masks as input significantly degrades the model's performance.

Table 4.4: Detailed results of both main models for each vessel in the *LabLiquidVolume* dataset.

		Segmented Depth Maps Model			**Segmented Depth Maps with Vessel Volume Model**		
	$Mean\ Liquid\ Volume$ (mL)	$RMSE$ (mL)	$MAPE$ (%)	R^2	$RMSE$ (mL)	$MAPE$	R^2
Cell Flask 160mL	115.22	74.64	38.61	-0.21	20.88	14.44	0.90
Cell Flask 400mL	277.46	62.62	21.60	0.83	29.88	12.40	0.96
Duran 100mL	54.31	7.26	14.14	0.94	5.18	9.80	0.97
Duran 250mL	142.29	31.66	13.08	0.76	11.30	8.79	0.97
Duran 500mL	302.71	22.44	6.84	0.98	18.55	4.11	0.98
Gibco 125mL	85.29	16.80	20.51	0.83	9.67	8.63	0.94
Gibco 500mL	325.45	43.63	13.53	0.92	27.09	10.06	0.97
Pyrex 100mL	57.29	8.21	22.46	0.91	5.37	14.34	0.96
Storage Bottle 250mL	175.05	38.94	7.99	0.65	7.94	3.60	0.99
Storage Bottle 500mL	279.40	17.06	7.39	0.99	11.88	4.90	0.99
Tube 15mL	8.47	2.51	44.52	0.62	1.95	15.79	0.77
Tube 50mL	33.03	4.02	10.62	0.89	3.64	9.67	0.91
All Vessels	**185.39**	**37.90**	**16.96**	**0.94**	**17.83**	**9.39**	**0.99**

Table 4.5: Comparison of the results of the four approaches used for liquid volume estimation.

Model Input	$RMSE\ Training$ (mL)	$RMSE\ Testing$ (mL)	$MAPE\ Testing$ (mL)	$R^2\ Testing$
Segmented Depth Maps	13.00	37.90	16.96	0.94
Segmented Depth Maps and Vessel Volume	4.63	17.83	9.39	0.99
Segmentation Masks	32.94	53.92	24.21	0.88
Scaled Segmented Depth Maps	18.84	41.20	18.20	0.92

4.1.3 Interpretations and Limitations

The segmentation and depth estimation model demonstrates better liquid segmentation results for glass containers compared to plastic containers on real images. This discrepancy might arise because the *TransProteus* dataset exclusively contains glass containers. Transparent plastics and glass differ in visual properties. Glass typically has higher optical clarity and a higher refractive index than most transparent plastics. The refractive index determines how light

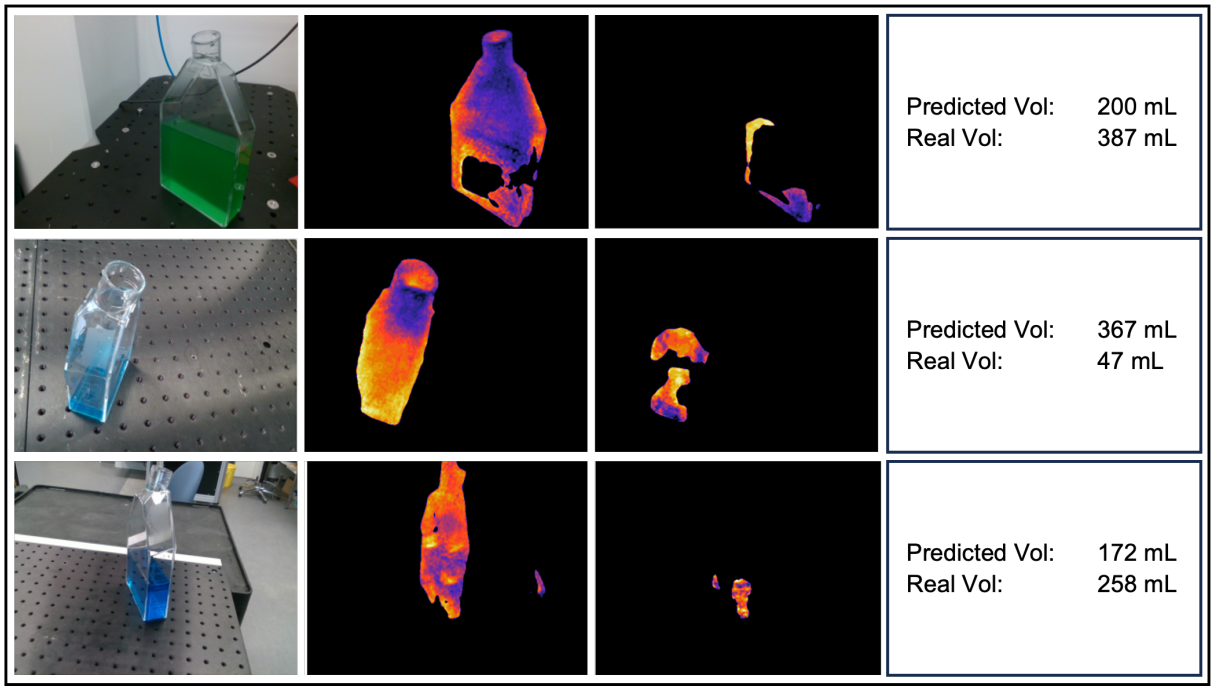


Figure 4.8: Three samples of the test set with the highest $MAPE$ predicted from the model trained with the segmented depth maps.

bends as it passes through a material. A solution would require an adaption of the *TransProteus* dataset using Blender to include plastic containers. Because of the found correlation between segmentation results and the accuracy of the volume estimation, this would most likely also decrease the $MAPE$ of the liquid volume estimation.

Beyond liquid volume estimation, the segmentation and depth estimation results offer potential for other applications in research laboratories. For instance, the segmentation of the opening region can be utilized to determine the success of decapping or capping, identify the presence of required transparent objects, or ascertain which flasks are loaded.

Incorporating depth maps alongside segmentation masks significantly enhances the volume estimation model's performance, underscoring the benefits of utilizing depth information. It is important to note that the volume estimation component of the model is trained solely on images obtained from the Intel RealSense D415 camera. Consequently, using cameras with different camera intrinsics may yield varying results. To address this limitation, a new approach could involve training the model on XYZ maps instead. Another alternative approach to address the limitation of varying camera intrinsics is to incorporate a calibration step. By calibrating the camera intrinsics of different devices used for data capture, it would be possible to transform the images into a standardized coordinate system, ensuring consistency across different cameras. This way, the model could be trained on the transformed depth maps, enabling it to generalize better to different camera setups.

The adequacy of the volume estimation model's accuracy depends on the specific task and context in which it is applied. For research laboratories, the approach can be useful for tasks such as detecting and replacing nearly empty vessels using mobile robots. However, it may not be suitable for experiments that require high precision. The liquid volume estimation model trained with the vessel volume provides a solution to the problem of training the depth estimation using scale-invariant loss. It provides a scaling factor for the size of the container and is, therefore, advantageous when confronted with new objects. The first exemplary trials with one

of the biggest transparent vessels in research laboratories (Glass Media Bottles, GL-45, Schott) showed good results when providing the vessel volume. One example is shown in fig. 4.9. Even though the liquid depth prediction considers the global scale to some degree, the model only using the segmented depth maps shows worse results for known objects and fails for unseen objects. The approach of using the median of the ground truth depth from the depth camera probably failed because of wrong depth values due to the described difficulties with transparent objects.

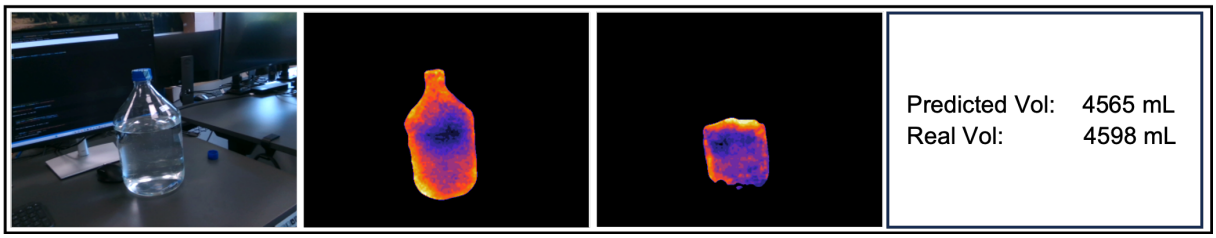


Figure 4.9: An example of the volume estimation trials with new, unseen laboratory vessels.

Alternatively, a new method could use the approach from the *ClearGrasp* paper [54] to infer the accurate 3D geometry of the transparent objects from the recorded RGB-D image and include that information for scaling the predicted liquid and vessel depth maps.

A comparison with existing liquid volume estimation approaches is difficult because of the differences in the variability in the datasets. The only exciting approaches for predicting the liquid volume from a single RGB image are the ones from Mottaghi et al. [61] and Cobo et al. [63]. The suggested method by Mottaghi et al. followed a classification approach, reaching a 32% per-class accuracy. A direct comparison between the regression and classification evaluation metrics is not possible. They divided the volume space of the containers into ten classes and the liquid volume of each container into five classes. The volume spacing between trained classes limits their methodology. This results in an average spacing of 176 mL, significantly higher than the shown $RMSE$ of 37.9 mL for the segmented depth maps model. The model trained for wine volume prediction by Cobo et al. [63] achieved a MAE of 8 mL, which is slightly lower than the achieved MAE of 9.3 mL for the model with vessel volume input and 18.4 mL for the model without. However, given the limited diversity of the dataset, the model's applicability is restricted to scenarios involving red liquid and wine glasses that closely resemble those used in the dataset in terms of shape and size. Despite this limitation, we were fortunate to obtain early access to the API from the authors. To evaluate the model's performance on laboratory containers, we randomly selected twenty images from the *LabLiquidVolume* dataset and utilized the API to predict the liquid volume in each image. However, the results obtained were inconsistent, showing arbitrary predictions. The $RMSE$ was 164 mL, with an average liquid volume of 189 mL.

The usage of the trained models for fully transparent/non-colored liquids is possible, and was tested on some laboratory samples. However, it should be noted that the segmentation and depth estimation results exhibited inferior performance when applied to non-colored liquids, consequently leading to a degradation in the precision of the liquid volume estimation. A possible solution could include a fine-tuning of the segmentation and depth estimation model with manually segmented images of transparent liquids.

4.2 Results of Simulation-based Autonomous Pouring

This section presents the results of the pouring simulation conducted in NVIDIA Flex and the simulation-to-reality transfer experiments described in section 3.3.2. The results of all the pour-

ing simulation scenes are provided in CSV summary files containing the scene number, the path to the recorded trajectory, the stop angle, the stop time, the start volume, the poured volume, the received volume, and the spilled volume. Exemplary samples from the file for the cell culture flask can be seen in table 4.6. For space reasons, the path is not shown.

Table 4.6: Exemplary results provided in the simulation summary output files.

Scene Number	Stop Angle (°)	Stop Time (s)	Start Vol. (mL)	Poured Vol. (mL)	Received Vol. (mL)	Spilled Vol. (mL)
235	20.0	1.8	38.8	17.7	17.6	0.1
1388	26.0	0.2	86.9	72.0	71.2	0.8
2189	28.0	1.0	116.1	105.2	100.7	4.6
1560	20.0	1.0	91.8	60.3	59.8	0.5
1786	22.0	0.6	101.6	74.8	73.0	1.8

4.2.1 Analysis of the Simulation Result Space

The simulations need to cover the whole result space to give valuable predictions with the possibility of selecting a pouring movement with any start and output volume. In other words, for a specific start volume, the maximum distances between the different poured volumes should be as small as possible.

For the cell culture flask, for every starting volume, 125 pours were simulated. An exemplary plot of the course of the remaining volume in the cell culture flask with a start volume of 35 mL over time is illustrated in fig. 4.10. It can be seen that most of the possible output volumes can be covered, with a more detailed distribution for lower remaining volumes. An equivalent visualization of the pouring movements with the media/washing solution bottle for a start volume of 80 mL is shown in fig. B.7.

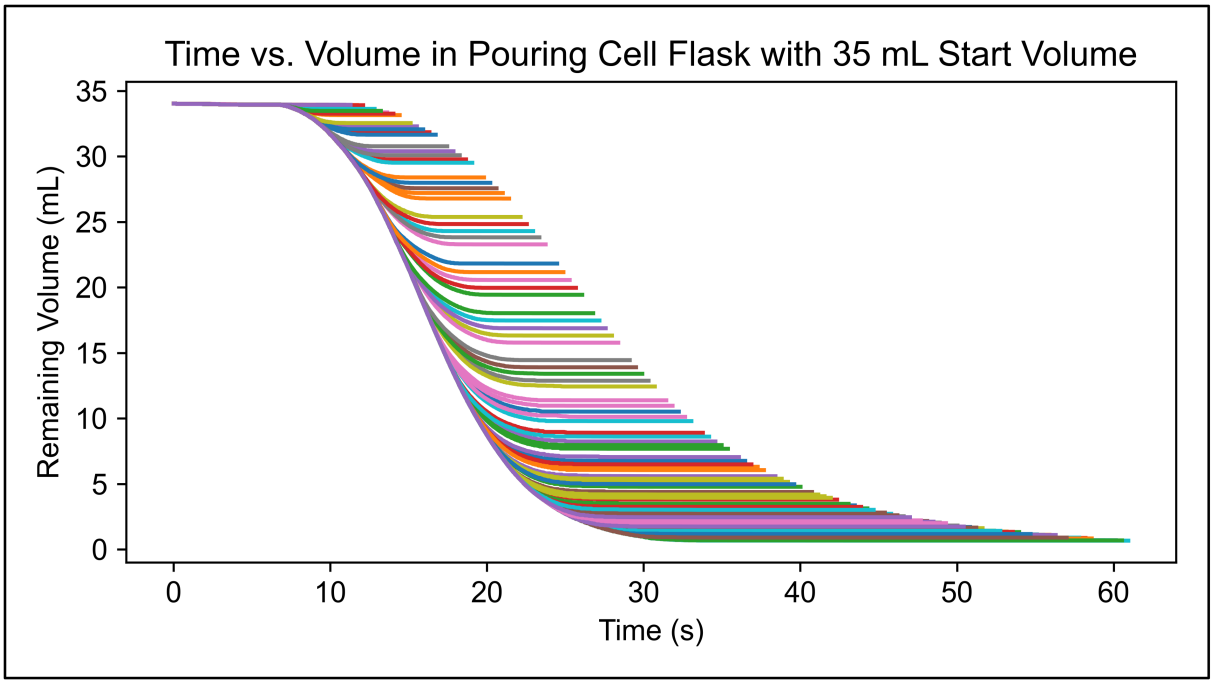


Figure 4.10: Visualizations of the course of the remaining volume in the cell culture flask with a starting volume of 35 mL over time for the 125 simulated pours.

The entire result space must be considered to find the simulated movement with the lowest

cost calculated in eq. (3.10). Important is hereby the coverage for the start and the received volumes. This relation for all simulated pouring movements is visualized in fig. 4.11.

It can be seen that the coverage of possible received volumes in the receiving flask is very high for low start volumes and gets lower for higher start volumes. It can also be observed that for start volumes above 80 mL, not the entire start volume in the pouring flask is received from the receiving container. The higher the stop angle, the higher the received liquid volume.

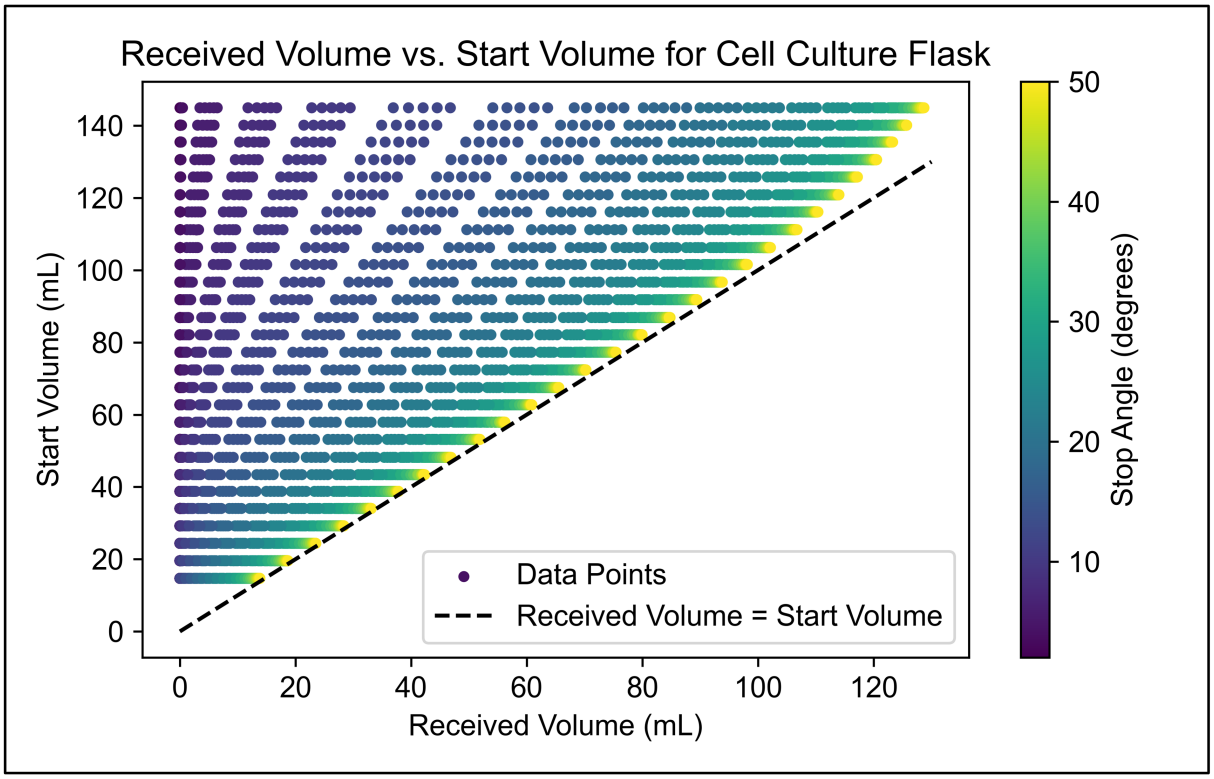


Figure 4.11: Visualization of the start and received volumes for all simulated cell culture flask pouring movements.

The equivalent plot for the media/washing solution bottle is shown in fig. B.11. Similarly to the cell culture flask, higher initial volumes result in reduced coverage, while lower start volumes exhibit higher coverage.

Detailed results of the analysis of the results space for each start volume are shown in table 4.7. It includes the maximum difference between two consecutive received volumes in mL and the mean difference between two consecutive received volumes. Those values are based on the minimum and maximum received volume range. The coverage represents the proportion of the possible output values between the minimum and maximum received volume in 1 mL steps that can be reached from the simulation outputs (received volume) with an error of less than 0.5 mL. The equivalent results for the media bottle are shown in table B.2.

For start volumes below 70 mL, the coverage is > 80%, with a maximum difference between two consecutive received volumes of < 3.5 mL. The higher the start volume, the higher the maximum and mean difference, and the lower the coverage of possible output values.

Further analysis can be done on the influence of varying the different input parameters. A detailed view of the course of the remaining volume for varying start volumes for a fixed stop angle and stop time is shown in fig. B.8. The influence of a varying stop angle on the remaining

Table 4.7: Detailed analysis of the results space for the different start volumes for the simulated pours with the cell culture flask.

Start Volume (mL)	Max. Difference (mL)	Mean Difference (mL)	Coverage (%)
15	0.49	0.11	100
20	0.71	0.15	100
24	0.94	0.19	100
29	1.22	0.23	92.86
34	1.46	0.25	87.88
39	1.78	0.30	92.11
43	2.06	0.34	90.48
48	2.16	0.38	85.11
53	2.47	0.42	80.77
58	2.77	0.45	83.93
63	3.02	0.49	80.33
68	3.32	0.53	80.30
72	3.52	0.57	74.29
77	3.82	0.61	72.00
82	4.04	0.64	67.50
87	4.39	0.68	68.24
92	4.83	0.72	62.92
97	5.06	0.76	60.64
102	5.49	0.79	62.24
106	5.78	0.82	59.80
111	6.10	0.86	57.94
116	6.38	0.89	56.36
121	6.54	0.92	56.14
126	6.79	0.94	55.56
131	7.07	0.97	53.72
135	7.27	0.99	52.85
140	7.28	1.01	53.17

volume over time for a fixed stop time and start volume can be seen in fig. B.9. The impact of changing stop times on the progression of the remaining volume over time, with a fixed stop angle and start volume, is illustrated in fig. B.10.

It can be observed that the variation in the stop angle is responsible for the coarse differences in the poured volume and the variation in the stop time for the more subtle differences.

In research laboratories, particularly where liquids can pose a safety hazard, minimizing the volume of spills is of critical importance. The calculation of the spilled volume in the simulation is based on the number of particles that are neither inside the receiving container nor remaining inside the pouring container. The spilled volume in relation to the received volume for the simulated pours with the cell culture flask is illustrated in fig. 4.12.

It can be observed that the spilled volume increases for higher received volumes. For target volumes below 105 mL, a simulated pour can be found without any spilled volume. In addition, the volume of spills increases for higher stop angles. The highest spilled amount of liquid (11.9 mL) was measured for a start volume of 145 mL, a stop angle of 50°, and a stop time of 2 seconds.

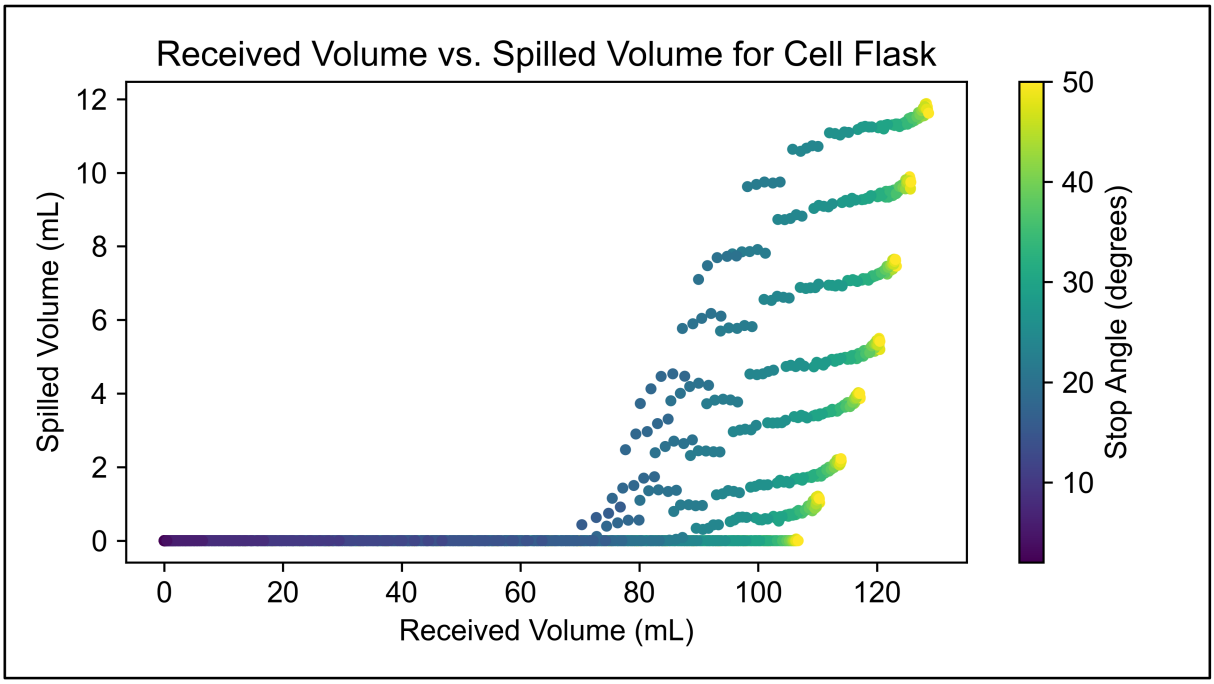


Figure 4.12: Visualization of spilled volume in relation to the received volume inside the receiving container for the simulated pours with the cell culture flask.

The equivalent plot for the media/washing solution bottle can be found in fig. B.12. Similar to the cell culture flask, pouring without spillage is possible for low target volumes. However, the spilled volume in the simulation increases significantly for pours with start volumes above 350 mL and target volumes of > 340 mL.

The goal of the big parameter space in the simulation is to minimize the cost as calculated in eq. (3.10). By finding this minimum, the best pouring movement can be chosen for a given start volume and a desired target volume to be poured. For the analysis, a set of every possible target volume for every possible start volume between 0 mL and the maximum of the received volumes with a step size of 1 mL was created. For the pouring with the cell culture flask, this results in 9,750 possible combinations of start and target volumes. For every point, the simulated pour with minimal cost was found in the list of results from the simulation. The plot in fig. 4.13 shows the minimum cost depending on start and target volumes. The lowest cost among the combinations is 0.045 mL, and the highest is 16.51 mL. The mean cost for the set is 2.27 mL.

It can be observed that the cost is lowest for samples with a start volume that was also used in the simulation. For start volumes below 100 mL, the cost is smaller than 7.5 mL for every possible target volume, with a mean cost of 1.8 mL. The cost increases significantly for high start volumes with target volumes above 90 mL.

The equivalent plot for the media/washing solution bottle can be seen in fig. B.13. The lowest cost among the mentioned combinations for the bottle is 0.005 mL, and the highest is 119.37 mL. The mean cost for the set is 14.24 mL.

4.2.2 Results of the Simulation-to-Reality Transfer

As explained in subsection 3.3.3, the simulated movements were transferred to be executed in a real-world setting on a UR5e. To illustrate this, an exemplary sequence of a simulated pour executed on the robot is shown in Figure fig. 4.14. Videos of both simulation-to-reality transfers

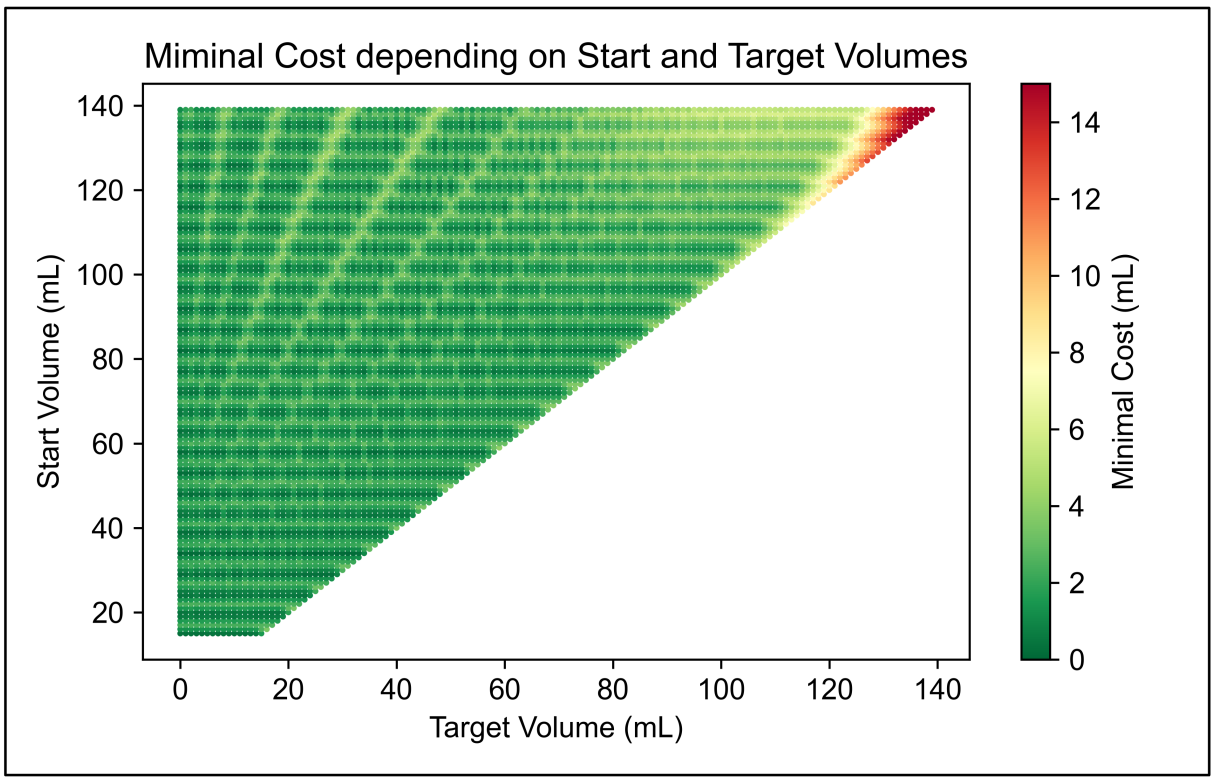


Figure 4.13: Visualization of the minimal cost for a simulated pour with a cell culture flask given a defined start and target volume with step sizes of 1 mL.

for the cell culture flask and the media/washing solution bottle are available here.

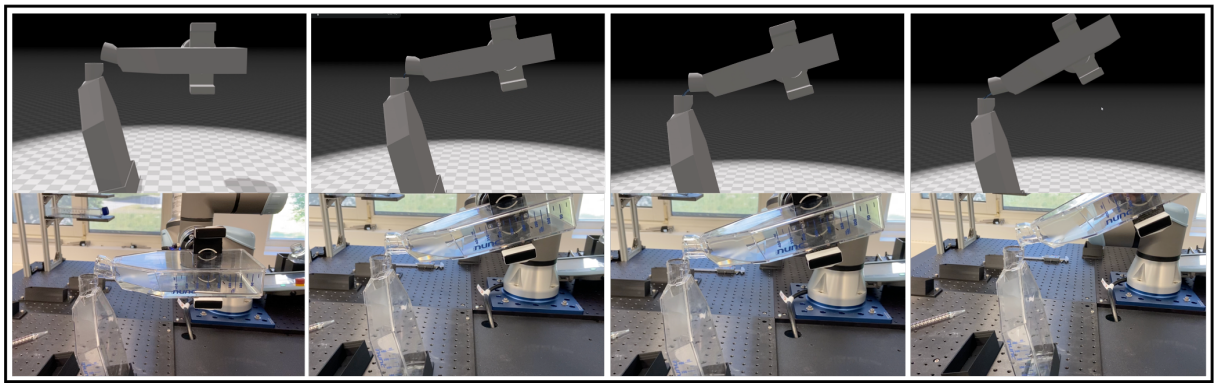


Figure 4.14: Visualization of a simulated pouring sequence with a cell culture flask executed in a real-world setting on a UR5e.

.

The results of the 100 simulated pours executed in reality are provided in two Excel files. Besides the simulation results described above, this includes the poured volume, the received volume, and the spilled volumes from the real execution. The results of the poured volumes in the experiments with the cell culture flask are visualized in fig. 4.15. The colors of the data points represent the ratio of the target volume (received volume in simulation) to the start volume.

It can be seen that the results in the simulation and reality are very similar for low target volumes (< 20 mL) and for cases where the ratio of target volume to start volume is high (yellow points). For nine samples, the amount of poured liquid in the real world is lower than in the simulation,

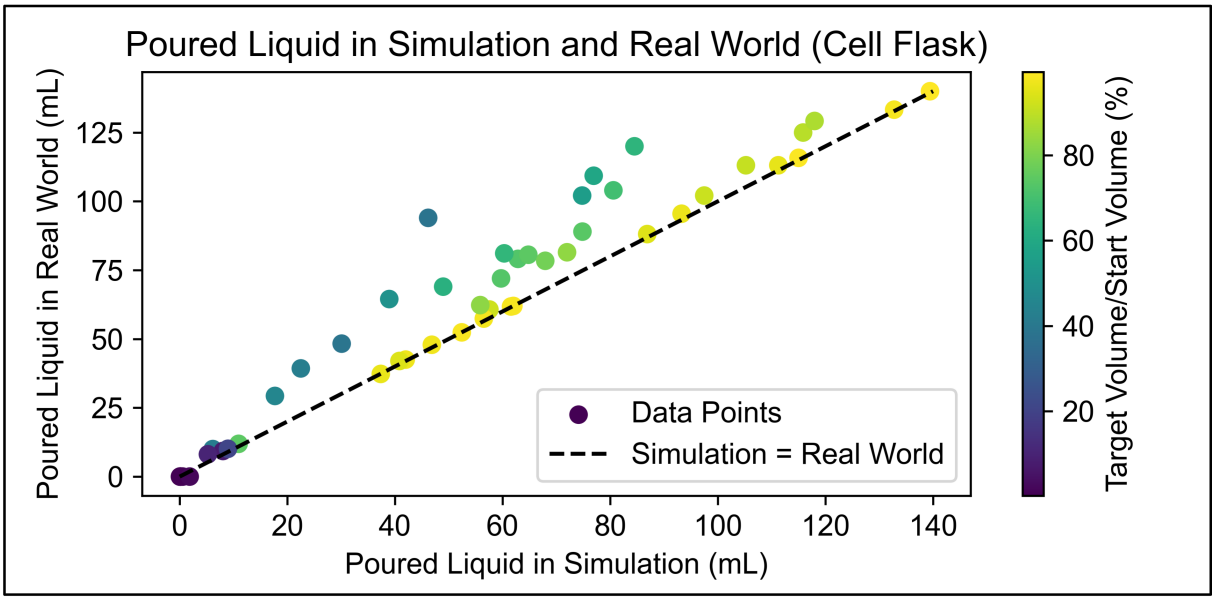


Figure 4.15: Results of the real pexments exwith ae and a cell culture flask as pouring container.

with a maximum difference of 1.8 mL. The real poured volume is higher than the simulated one for all other pours, with a maximum difference of 49.3 mL. The highest differences occur for ratios of the target volume and start volume between 0.3 and 0.7.

The equivalent plot for the media/washing solution bottle is shown in fig. 4.16. The results are similar to the pours with the cell culture flask. For most samples (n=31), the real poured volume is higher than the simulated one, with a mean difference of 29.2 mL and a maximum difference of 52.6 mL. Again, precise results are achieved for cases where the ratio of the target volume to the start volume is high.

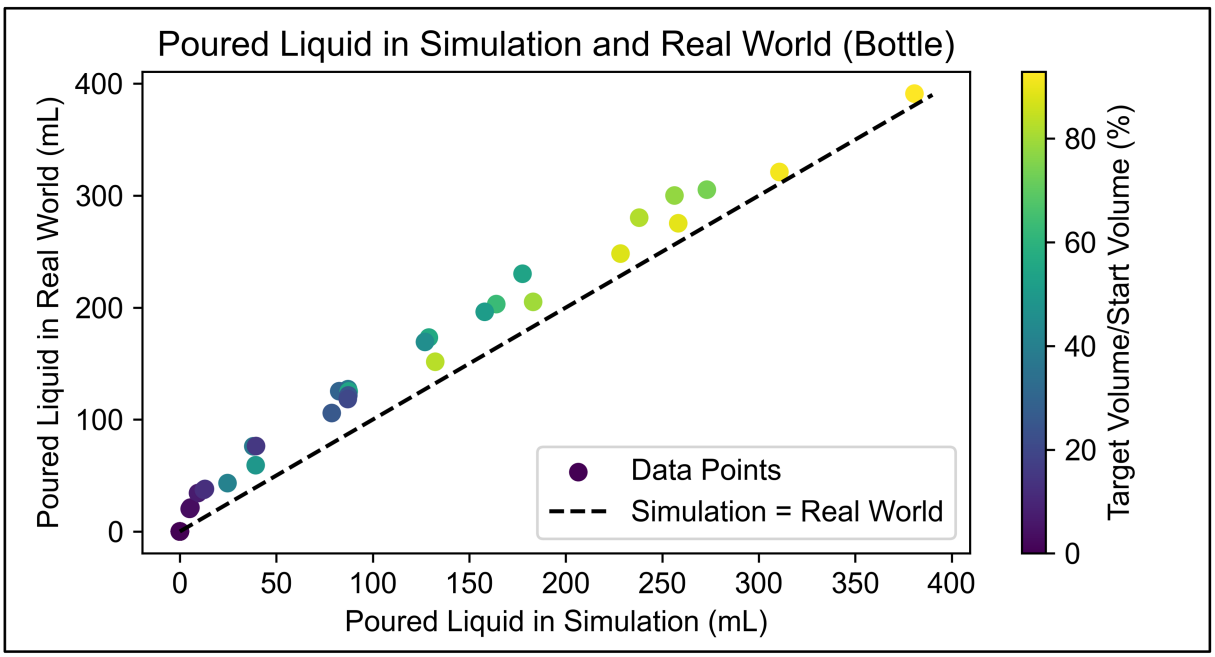


Figure 4.16: Results of the real pouring experiments executed with a UR5e and a Gibco 500 mL media/washing solution bottle as pouring container.

The results of the spilled volumes in the real executions with the cell culture flask are visualized in fig. 4.17. For 35 samples, the real spilled volume is lower than the predicted spilled volume from the simulation, with a maximum difference of 13.2 mL and a mean difference of 2.1 mL. For the remaining 15 pours, the real spilled volume is higher than the simulated one, with a maximum difference of 2.4 mL and a mean difference of 0.5 mL. It can be observed that the real spilled volume is close to the simulated spilled volume for low amounts of spilled liquid and low start volumes. For high start volumes, the volume spilled in the simulated pours is higher than in the real-world experiments.

The equivalent plot for the media/washing solution bottle can be seen in fig. B.14. For 40 samples, the real spilled volume is lower than the predicted spilled volume from the simulation, with a maximum difference of 37.8 mL and a mean difference of 6.4 mL. For the other ten pours, the real spilled volume is equal to or higher than the simulated one, with a maximum difference of 1.17 mL and a mean difference of 0.2 mL.

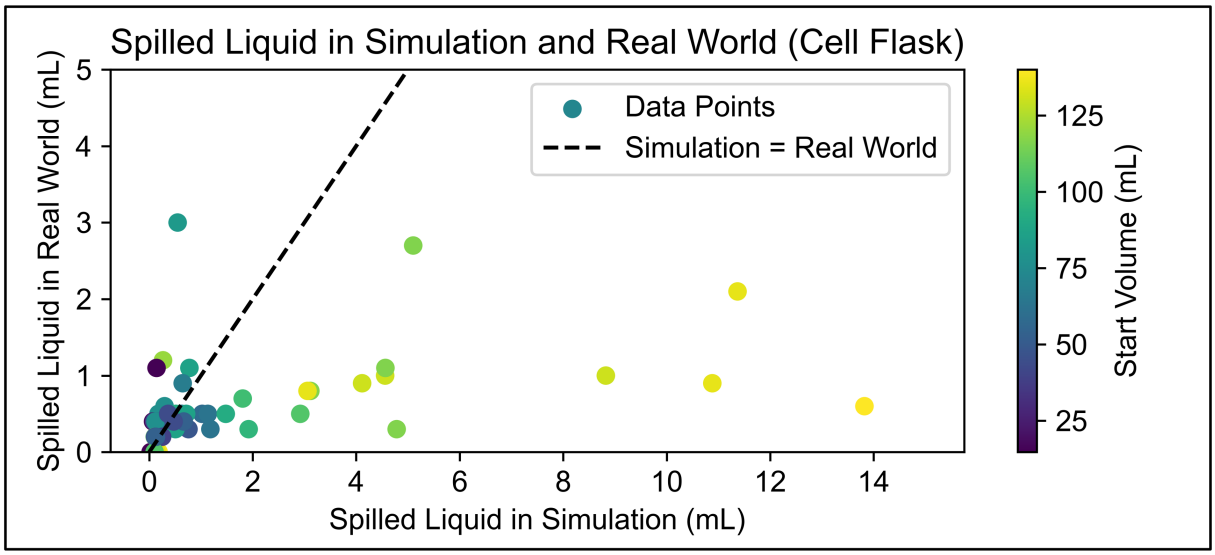


Figure 4.17: Results of the spilled volume in the real pouring experiments executed with a UR5e and a cell culture flask as pouring container.

Detailed results of the simulation-to-reality experiments for the cell culture flask and the media/washing solution bottle are listed in table 4.8. The $RMSE$, $MAPE$ and R^2 values are calculated as shown in eq. (3.11), eq. (3.12), and eq. (3.13).

Table 4.8: Detailed results of the simulation-to-reality experiments for the cell culture flask and media/washing solution bottle. The metrics are calculated as explained in subsection 3.3.2.

		$RMSE$ (mL)	$MAPE$ (%)	R^2
Cell Culture Flask	Received Volume	15.02	21.0	0.83
Cell Culture Flask	Spilled Volume	3.21	63.0	-0.06
Bottle	Received Volume	31.84	53.0	0.88
Bottle	Spilled Volume	9.20	111.0	-0.34

4.2.3 Interpretations and Limitations

The main advantage of using pre-simulated pouring results is the reduction of numerous time-consuming real pouring attempts, allowing for more efficient experimentation and adaptation.

Moreover, a detailed target volume precision achieved in simulations is possible because of the possibility of varying various pouring parameters. Additionally, the presented simulation exhibits adaptability to new pouring containers, enhancing its practicality.

However, certain gaps between the simulation and reality can be encountered. Notably, the lack of available 3D models for the pouring objects poses a challenge in accurately reproducing the pouring behavior. A slightly wrong-modeled bottleneck can significantly change the simulation results. The reliance on approximate liquid parameters in the simulation also contributed to the disparities. An iterative adjustment of the used liquid parameters is a possible solution to improve the accuracy of the simulation. However, this is only useful if the exact 3D model of the pouring containers is available. Otherwise, the parameter adjustment would only cancel out the error of the 3D modeling. The difference in the selected start volumes for the cell culture flask and the actual start volumes in the simulations is due to a minimal 'hole' in the 3D object of the flask, which leads to a leakage of particles during the particle emitting.

The simulation-to-reality gap can be minimized using a correction factor, leading to mean absolute percentage errors of 14% and 25% for the received volume. To interpret the results in a research laboratory context, the opinion of scientists in the R&eD site of Novo Nordisk was obtained. The general tenor is that the practicability is highly dependent on the experimental procedure. For many preparation tasks in the laboratory, a $MAPE$ of around 15% might be sufficient.

The described new pouring trajectory showed very promising results for pouring into receiving containers with a small opening. During the real execution, very low spilling occurred for both pouring containers. Since the trajectory can be easily adapted for new pouring containers, this new approach has many possible applications in automating research experiments, but also for other robotic pouring problems like filling drinks for a kitchen assistant.

4.3 Overall System Performance

This section presents the results of the performance of the system prototype for cell culture automation as presented in section 3.1.

The described modules, which can be combined to execute the three main workflows, were tested individually. Iterative adjustments and improvements were made until 20 consecutive runs of each module were accomplished successfully. In the modules involving pouring tasks, it is important to note that the precision of the pour was not taken into consideration as a determining factor for successful completion. Videos of the successful completion of each module can be seen here.

The precision of the autonomous pouring approach, which includes the vision-based volume estimation of liquids, the selection of the pour with the minimum cost from the simulation, and the real execution of the pour, was tested. To achieve this, module 6 was repeated 20 times for each target volume of 30 mL and 50 mL, utilizing the media bottle. Random start volumes ranging from 100 mL to 500 mL were used.

The target volumes were specifically chosen to align with the typical amount of media poured by human scientists, which commonly falls within the range of 30 mL to 50 mL. A correction factor of 1.15 was applied to the received volumes in the simulation to improve the simulation-to-reality transfer. The results are listed in table 4.9. $RMSE$ and $MAPE$ are calculated as shown in eq. (3.11) and eq. (3.12).

The three main workflows (A: Analyzing cell growth, B: Changing media, C: Passaging) were tested 10 times each. Start conditions were one filled cell culture flask in the flask storage inside the incubator, five empty flasks in the flask storage on the table, a media and a washing solution

Table 4.9: Results of the complete autonomous pouring workflow tested with module 6 (20 times per target volume).

Target Volume (mL)	Average Execution Time (s)	$RMSE$ (mL)	$MAPE$ (%)	Maximum Error (mL)
30	135.3	21.4	71.3	39.6
50	124.8	26.2	52.4	44.8

bottle with a sufficient amount of liquid, and a filled trypsin container. The results of the tests are shown in table 4.10. Exemplary pictures of the execution of the workflows and the simple user interface to start the workflows and visualize the cell confluency are shown in fig. 4.18.

Table 4.10: Results of the autonomous workflows A, B, and C executed on the cell culture system (10 times each).

Workflow	Average Execution Time (s)	Completion Rate (%)
A	78.3	100
B	461.4	100
C	841.2	90

Workflows A and B achieved a completion rate of 100%. Workflow C encountered a single failure, which occurred during the flask capping phase. The execution times for the workflows A and B are similar to a human scientist. However, especially decapping and capping flasks takes more time in the autonomous workflows, which makes workflow C comparatively time-consuming. After further testing, the speed of the individual modules can be increased to reduce the total execution time. Videos of the three workflows can be seen here.

The possibility of using the segmentation and depth estimation model for process monitoring is showcased in a system setup verification phase prior to the execution of the autonomous workflows. This helps to avoid human errors when loading the system. The robot moves to different elements of the system and captures a total of eight images. After, the segmentation masks are predicted, which are then utilized to validate the presence of containers through simple value-counting functions. Specifically, the checks involve examining whether the microscope unit, vertical flask holders, and regripping station for the bottles are empty, as well as ensuring an adequate number of empty flasks (>2), the presence of the trash container, and the availability of media and washing solution bottles. Each of the six checks was conducted 10 times with a container present and 10 times without. Notably, all 120 checks were successfully completed. A video of the system setup verification phase can be seen here. Snapshots of the process can be seen in fig. B.22.

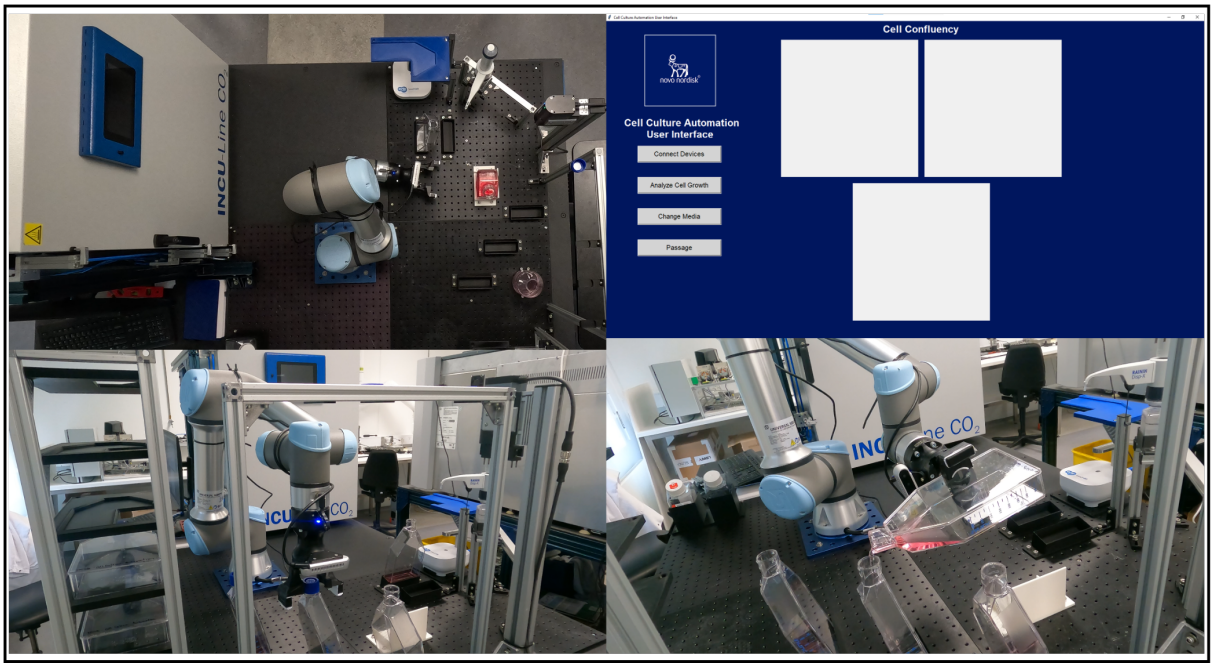


Figure 4.18: Exemplary pictures of the execution of the workflows and the simple user interface to start the workflows and visualize the cell confluency.

5 Discussion and Future Work

The exploratory character of the present work was already emphasized at the beginning. The open-ended goals were defined to create a basis and provide ideas for further projects and research in this area.

Research goal (1), to integrate existing laboratory hardware for a proof-of-concept prototype, was successfully demonstrated by autonomously executing the three main workflows in cell culture. Combining a robot arm, existing laboratory equipment, and 3D-printed parts allows the prototype to be easily rebuilt and extended. Subelements like the incubator with the door opening mechanism can be reused and multiplied for future projects through the achieved modularity. In combination with autonomous pouring, the desired flexibility and adaptability for new applications in laboratory automation are fulfilled. The system is the first low-cost solution for cell culture automation using cell culture flasks. The equipment costs required to build the system are approximately DKK 432,000 (€ 58,000). An overview of the expenses is shown in table B.4. However, some further factors need to be taken into consideration for a final system. First, the system must be encapsulated to ensure a stable atmosphere. Second, more flask storage inside the incubator is needed to allow multiple passaging procedures without human intervention.

Goal (2), to perform human tasks that include liquids using a camera and a robot arm, was approached using a two-step deep learning-based method for liquid volume estimation and a simulation of various pouring movements. The possibilities and limitations of the two elements were evaluated in detail in the results sections. In summary, the vision-based system was successfully used for process monitoring (capping/decapping, presence of flasks), and the volume of liquid can be estimated with satisfactory precision for known laboratory vessels or when the vessel volume is given. Future work could involve the extension of the *TransProteus* dataset with plastic objects, the enlargement of the *LabLiquidVolume* dataset with more vessel types, and an adaption of the volume estimation model to the use of point clouds. The particle-based simulation predicts the required real robot arm movement to pour a specific amount of liquid depending on varying starting volumes with satisfactory precision. However, an adaption to multiple laboratory containers with real 3D models is desirable. Another possibility to improve the simulation-to-reality transfer accuracy is to build a setup for autonomous filling, pouring, and weighing of the poured liquid. This could be used to run many more real pouring executions and subsequently adapt the simulation results with the real ones. Future simulation work for liquid pouring could be done in Isaac Orbit, a unified and modular framework for robot learning powered by NVIDIA Isaac Sim [97]. A pouring simulation module will be added end of 2023, which could help reduce the time for setting up the pouring scenes. Compared to the state-of-the-art approach for autonomous pouring by Kennedy et al. [64], the vision- and simulation-based approach presented here is worse in precision, but offers higher flexibility for pouring into multiple receiving containers and simplicity by using less equipment.

Performing all human pouring tasks in research laboratories with the presented combined approach with the current results is not feasible since the volume estimation and pouring simulation errors add up. However, improving the accuracy of both elements could lead to sufficient precision for many laboratory tasks. When very precise pouring is demanded, an option could include the vision-based monitoring of the liquid in the pouring and the receiving container.

6 Conclusion

This work investigated the opportunities of a vision- and simulation-based approach to make a robot a flexible and adaptable laboratory assistant, demonstrated with a proposed new low-batch system for cell culture automation. The proposed system demonstrates a low-cost solution for the three main workflows in cell culture. It has the potential to enhance scientists' productivity and alleviate the workload of researchers, eliminating the necessity for their physical presence in the laboratory during non-standard working hours. Thanks to the modularity of the cell culture automation prototype, elements can be reused for further laboratory automation projects and for a final solution of the system.

A new method consisting of a two-step vision-based system was proposed for liquid volume estimation from a single RGB image. First, segmented depth maps of transparent containers and the liquids inside are generated based on a convolutional neural network (CNN) trained on computer-generated images from the *TransProteus* dataset. In the second step, the volume of the liquid inside the container is estimated based on those results using a CNN trained on a newly generated dataset called *LabLiquidVolume*. The presented dataset consists of more than 5,000 images of liquids inside the most common laboratory containers and their volume.

More than 6,000 pours covering different start volumes, stop angles, and stop times were simulated. Unlike previous autonomous pouring approaches, the proposed new movement is a rotation of the pouring container around the liquid exit point instead of the center of the container. This makes the trajectory more suitable for pouring into containers with small openings, which often appear in research laboratory environments.

The proposed autonomous pouring approach, which includes the vision-based volume estimation and the selection of the best pour from the simulation, was demonstrated with real executions on a robot arm in the cell culture system.

The proposed vision- and simulation-based approach offers a cost-effective solution that can be easily integrated into existing research laboratory setups. By improving the subelements' accuracy, this approach has the potential to revolutionize research practices through flexible robotic assistance, enabling scientists to focus on critical tasks and driving scientific progress.

Bibliography

[1] Carl Benedikt Frey and Michael A. Osborne. “The future of employment: How susceptible are jobs to computerisation?” In: *Technological Forecasting and Social Change* 114 (Jan. 2017), pp. 254–280. ISSN: 00401625. DOI: 10.1016/j.techfore.2016.08.019.

[2] Ian Holland and Jamie A. Davies. *Automation in the Life Science Research Laboratory*. Nov. 2020. DOI: 10.3389/fbioe.2020.571777.

[3] Chaitanya Kulkarni. “Automating the Experimental Laboratory”. PhD thesis. Ohio: Ohio State University, 2021.

[4] Paul Groth and Jessica Cox. “Indicators for the use of robotic labs in basic biomedical research: A literature analysis”. In: *PeerJ* 2017.11 (2017). ISSN: 21678359. DOI: 10.7717/peerj.3997.

[5] Giuseppe Lippi and Giorgio Da Rin. *Advantages and limitations of total laboratory automation: A personal overview*. June 2019. DOI: 10.1515/cclm-2018-1323.

[6] Irina Alecu et al. “Cell culture metabolomics and lipidomics”. In: *Metabolomics Perspectives: From Theory to Practical Application*. Elsevier, Jan. 2022, pp. 415–456. ISBN: 9780323850629. DOI: 10.1016/B978-0-323-85062-9.00012-X.

[7] Charis P. Segeritz and Ludovic Vallier. “Cell Culture: Growing Cells as Model Systems In Vitro”. In: *Basic Science Methods for Clinical Researchers*. Elsevier Inc., Apr. 2017, pp. 151–172. ISBN: 9780128030783. DOI: 10.1016/B978-0-12-803077-6.00009-6.

[8] Merck KGaA. *Fundamental Techniques in Cell Culture Laboratory Handbook 4th Edition*. Tech. rep. Darmstadt, 2018.

[9] Andreas Elanzew et al. “The StemCellFactory: A Modular System Integration for Automated Generation and Expansion of Human Induced Pluripotent Stem Cells”. In: *Frontiers in Bioengineering and Biotechnology* 8 (Nov. 2020). ISSN: 22964185. DOI: 10.3389/fbioe.2020.580352.

[10] Formulatrix. *Automated Cell Culture*. 2023.

[11] Péter Egri et al. “Bio-inspired control of automated stem cell production”. In: *Procedia CIRP*. Vol. 88. Elsevier B.V., 2020, pp. 600–605. DOI: 10.1016/j.procir.2020.05.105.

[12] P. Moutsatsou et al. *Automation in cell and gene therapy manufacturing: from past to future*. Nov. 2019. DOI: 10.1007/s10529-019-02732-z.

[13] M. Urbina, A. Watts, and E. Reardon. “Labs should cut plastic waste too”. In: *Nature 528* (2015), p. 479. DOI: https://doi.org/10.1038/528479c.

[14] Richard Ricardo and Katy Phelan. “Trypsinizing and subculturing mammalian cells”. In: *Journal of Visualized Experiments* 16 (2008). ISSN: 1940087X. DOI: 10.3791/755.

[15] Barry G. Hall et al. “Growth rates made easy”. In: *Molecular Biology and Evolution* 31.1 (Jan. 2014), pp. 232–238. ISSN: 07374038. DOI: 10.1093/molbev/mst187.

[16] Sagi Eppel et al. “Predicting 3D shapes, masks, and properties of materials inside transparent containers, using the TransProteus CGI dataset”. In: *Digital Discovery* 1.1 (2022), pp. 45–60. DOI: 10.1039/d1dd00014d.

[17] Guoqiang Wang et al. *Machine Learning in Unmanned Systems for Chemical Synthesis*. Mar. 2023. DOI: 10.3390/molecules28052232.

[18] Hector Garcia Martin et al. *Perspectives for self-driving labs in synthetic biology*. Tech. rep. 2022.

[19] Milad Abolhasani and Eugenia Kumacheva. “The rise of self-driving labs in chemical and materials sciences”. In: *Nature Synthesis* (Jan. 2023). DOI: 10.1038/s44160-022-00231-0.

[20] Johan Stahre, Jörgen Frohm, and Veronica Lindström. *Levels of Automation in Manufacturing Factory-in-a-Box View project ProWood View project Jörgen Frohm Swedish transportation administration Levels of Automation in Manufacturing*. Tech. rep. 2008. URL: https://www.researchgate.net/publication/255793362.

[21] Ross King et al. "The Robot Scientist Adam". In: *Computer* 42.8 (2009), pp. 46–54. DOI: 10.1109/MC.2009.270. URL: www.yeastgenome.org.

[22] Andrew Sparkes et al. "Towards Robot Scientists for autonomous scientific discovery". In: *Automated Experimentation* 2 (2010). URL: http://www.aejournal.net/content/2/1/1.

[23] B P Macleod et al. *Self-driving laboratory for accelerated discovery of thin-film materials*. Tech. rep. 2020. URL: https://www.science.org.

[24] Hatem Fakhruldeen et al. "ARChemist: Autonomous Robotic Chemistry System Architecture". In: (Apr. 2022). URL: http://arxiv.org/abs/2204.13571.

[25] Benjamin Burger et al. "A mobile robotic chemist". In: *Nature* 583.7815 (July 2020), pp. 237–241. ISSN: 14764687. DOI: 10.1038/s41586-020-2442-2.

[26] Ewart J. de Visser et al. "Almost human: Anthropomorphism increases trust resilience in cognitive agents". In: *Journal of Experimental Psychology: Applied* 22.3 (Sept. 2016), pp. 331–349. ISSN: 1076898X. DOI: 10.1037/xap0000092.

[27] Labman Automation Ltd. *MultiDose: Automated XPR Solid Dosing System*. 2023. URL: https://www.labmanautomation.com/multidose/.

[28] Sebastian Steiner et al. "Organic synthesis in a modular robotic system driven by a chemical programming language". In: *Science* 363.6423 (Jan. 2019). ISSN: 10959203. DOI: 10.1126/science.aav2211.

[29] Sarah Kleine-Wechelmann et al. "Designing the mobile robot Kevin for a life science laboratory". In: (Apr. 2023). DOI: 10.1109/RO-MAN53752.2022.9900786. URL: http://arxiv.org/abs/2304.09090%20http://dx.doi.org/10.1109/RO-MAN53752.2022.9900786.

[30] Naruki Yoshikawa et al. "Chemistry Lab Automation via Constrained Task and Motion Planning". In: (Dec. 2022). URL: http://arxiv.org/abs/2212.09672.

[31] Andrew J. Capel et al. *3D printing for chemical, pharmaceutical and biological applications*. Dec. 2018. DOI: 10.1038/s41570-018-0058-y.

[32] Brandon G. Wong et al. "Precise, automated control of conditions for high-throughput growth of yeast and bacteria with eVOLVER". In: *Nature Biotechnology* 36.7 (Aug. 2018), pp. 614–623. ISSN: 15461696. DOI: 10.1038/nbt.4151.

[33] Maciej Daniszewski et al. *Automated Cell Culture Systems and Their Applications to Human Pluripotent Stem Cell Studies*. Aug. 2018. DOI: 10.1177/2472630317712220.

[34] R. Lehmann et al. "Biomek Cell Workstation: A Variable System for Automated Cell Cultivation". In: *Journal of Laboratory Automation* 21.3 (June 2016), pp. 439–450. ISSN: 22110690. DOI: 10.1177/2211068215599786.

[35] Christopher J. Bernard et al. "Adjunct Automation to the Cellmate™ Cell Culture Robot". In: *Journal of Laboratory Automation* 9.4 (2004), pp. 209–217. ISSN: 15402452. DOI: 10.1016/j.jala.2004.03.004.

[36] Tecan. "Cellerity". In: *Tecan Journal* 3 (2008), p. 8.

[37] Agilent Technologies. *Agilent BioCel System Configuration*. 2009.

[38] Ryuji Kato et al. *A Compact, Automated Cell Culture System for Clinical Scale Cell Expansion from Primary Tissues*. Tech. rep. 2010.

[39] Shushant Jain et al. "The complete automation of cell culture: Improvements for high-throughput and high-content screening". In: *Journal of Biomolecular Screening* 16.8 (Sept. 2011), pp. 932–939. ISSN: 10870571. DOI: 10.1177/1087057111413920.

[40] Advanced Technology Inc. *Automated Cell Culture System*. 2016.

[41] Jelena Ochs et al. “Advances in automation for the production of clinical-grade mesenchymal stromal cells: the AUTOSTEM robotic platform”. In: *Cell and Gene Therapy Insights* 3.8 (Oct. 2017), pp. 739–748. ISSN: 20597800. DOI: 10.18609/cgti.2017.073.

[42] Carlos A Tristan et al. *Robotic High-Throughput Biomanufacturing and Functional Differentiation of Human Pluripotent Stem Cells*. Tech. rep. 2020. DOI: 10.1101/2020.08.03.235242. URL: https://doi.org/10.1101/2020.08.03.235242.

[43] Abdullah Ayub Khan, Asif Ali Laghari, and Shafique Ahmed Awan. “Machine Learning in Computer Vision: A Review”. In: *EAI Endorsed Transactions on Scalable Information Systems* 8.32 (2021), pp. 1–11. ISSN: 20329407. DOI: 10.4108/eai.21-4-2021.169418.

[44] Lucas Mohimont et al. *Computer Vision and Deep Learning for Precision Viticulture*. Oct. 2022. DOI: 10.3390/agronomy12102463.

[45] Laith Alzubaidi et al. “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions”. In: *Journal of Big Data* 8.1 (Dec. 2021). ISSN: 21961115. DOI: 10.1186/s40537-021-00444-8.

[46] D H Hubel and Ad T N Wiesel. *Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex*. Tech. rep. 1962, p. 106.

[47] Anamika Dhillon and Gyanendra K. Verma. *Convolutional neural network: a review of models, methodologies and applications to object detection*. June 2020. DOI: 10.1007/s13748-019-00203-0.

[48] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. “Learning Deconvolution Network for Semantic Segmentation”. In: (May 2015). URL: http://arxiv.org/abs/1505.04366.

[49] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation”. In: (Nov. 2014). URL: http://arxiv.org/abs/1411.4038.

[50] Yue Ming et al. “Deep learning for monocular depth estimation: A review”. In: *Neurocomputing* 438 (May 2021), pp. 14–33. ISSN: 18728286. DOI: 10.1016/j.neucom.2020.12.089.

[51] David Eigen, Christian Puhrsch, and Rob Fergus. “Depth Map Prediction from a Single Image using a Multi-Scale Deep Network”. In: (June 2014). URL: http://arxiv.org/abs/1406.2283.

[52] Chao Qiang Zhao et al. *Monocular depth estimation based on deep learning: An overview*. Sept. 2020. DOI: 10.1007/s11431-020-1582-8.

[53] Mehdi Mousavi and Rolando Estrada. “SuperCaustics: Real-time, open-source simulation of transparent objects for deep learning applications”. In: (July 2021). URL: http://arxiv.org/abs/2107.11008.

[54] Shreeyak S. Sajjan et al. “ClearGrasp: 3D Shape Estimation of Transparent Objects for Manipulation”. In: (Oct. 2019). URL: http://arxiv.org/abs/1910.02550.

[55] Sagi Eppel and Tal Kachman. *Computer vision-based recognition of liquid surfaces and phase boundaries in transparent vessels, with emphasis on chemistry applications*. Tech. rep. 2014. DOI: https://doi.org/10.48550/arXiv.1404.7174.

[56] Sagi Eppel. *Tracing liquid level and material boundaries in transparent vessels using the graph cut computer vision approach*. Tech. rep. 2016.

[57] Sagi Eppel et al. *Computer vision for recognition of materials and vessels in chemistry lab settings and the Vector-LabPics dataset*. Tech. rep. 2020.

[58] Yoshitaka Hara et al. “Detection of liquids in cups based on the refraction of light with a depth camera using triangulation”. In: *IEEE International Conference on Intelligent Robots and Systems*. Institute of Electrical and Electronics Engineers Inc., Oct. 2014, pp. 5049–5055. ISBN: 9781479969340. DOI: 10.1109/IROS.2014.6943280.

[59] Connor Schenck and Dieter Fox. “Detection and Tracking of Liquids with Fully Convolutional Networks”. In: (June 2016). URL: http://arxiv.org/abs/1606.06266.

[60] Tara Zepel et al. *Automated Liquid-Level Monitoring and Control using Computer Vision*. Tech. rep. 2021.

[61] Roozbeh Mottaghi et al. “See the Glass Half Full: Reasoning about Liquid Containers, their Volume and Content”. In: (Jan. 2017). URL: http://arxiv.org/abs/1701.02718.
[62] Fan Zhu et al. “Visual-Tactile Sensing for Real-time Liquid Volume Estimation in Grasping”. In: (Feb. 2022). URL: http://arxiv.org/abs/2202.11503.
[63] Miriam Cobo et al. “Artificial intelligence to estimate wine volume from single-view images”. In: *Heliyon* 8.9 (Sept. 2022). ISSN: 24058440. DOI: 10.1016/j.heliyon.2022.e10557.
[64] Monroe Kennedy et al. “Autonomous Precision Pouring from Unknown Containers”. In: *IEEE Robotics and Automation Letters* 4.3 (July 2019), pp. 2317–2324. ISSN: 23773766. DOI: 10.1109/LRA.2019.2902075.
[65] Chau Do and Wolfram Burgard. “Accurate Pouring with an Autonomous Robot Using an RGB-D Camera”. In: *International Conference on Intelligent Autonomous Systems* 15 (Oct. 2018), pp. 210–221. URL: http://arxiv.org/abs/1810.03303.
[66] Gautham Narayan Narasimhan et al. “Self-supervised Transparent Liquid Segmentation for Robotic Pouring”. In: (Mar. 2022). URL: http://arxiv.org/abs/2203.01538.
[67] Connor Schenck and Dieter Fox. *Visual Closed-Loop Control for Pouring Liquids*. Tech. rep. 2017.
[68] Yongqiang Huang, Juan Wilches, and Yu Sun. “Robot gaining accurate pouring skills through self-supervised learning and generalization”. In: *Robotics and Autonomous Systems* 136 (Feb. 2021). ISSN: 09218890. DOI: 10.1016/j.robot.2020.103692.
[69] Tatiana López Guevara et al. *Adaptable Pouring: Teaching Robots Not to Spill using Fast but Approximate Fluid Simulation*. Tech. rep. 2017, pp. 77–86.
[70] C Pozrikidis. *Fluid Dynamics*. 3rd ed. New York: Springer, 2017. DOI: https://doi.org/10.1007/978-1-4899-7991-9.
[71] Matthias Müller et al. “Position Based Dynamics”. In: *J. Vis. Communications* 18.2 (2007), pp. 109–118. DOI: https://doi.org/10.1016/j.jvcir.2007.01.005.
[72] Miles Macklin and Matthias Müller. “Position based fluids”. In: *ACM Transactions on Graphics* 32.4 (July 2013). ISSN: 07300301. DOI: 10.1145/2461912.2461984.
[73] Tatiana Lopez-Guevara et al. “Stir to pour: Efficient calibration of liquid properties for pouring actions”. In: *IEEE International Conference on Intelligent Robots and Systems*. Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 5351–5357. ISBN: 9781728162126. DOI: 10.1109/IROS45743.2020.9340852.
[74] Edwin Babaians et al. “PourNet: Robust Robotic Pouring Through Curriculum and Curiosity-based Reinforcement Learning”. In: *IEEE International Conference on Intelligent Robots and Systems*. Vol. 2022-October. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 9332–9339. ISBN: 9781665479271. DOI: 10.1109/IROS47612.2022.9981195.
[75] Chau Do, Camilo Gordillo, and Wolfram Burgard. “Learning to Pour using Deep Deterministic Policy Gradients”. In: *IEEE RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 3074–3079.
[76] Beatriz Moya et al. “Physically sound, self-learning digital twins for sloshing fluids”. In: *PLoS ONE* 15.6 (June 2020). ISSN: 19326203. DOI: 10.1371/journal.pone.0234569.
[77] Maria G. Juarez, Vicente J. Botti, and Adriana S. Giret. “Digital twins: Review and challenges”. In: *Journal of Computing and Information Science in Engineering* 21.3 (June 2021). ISSN: 15309827. DOI: 10.1115/1.4050244.
[78] Universal Robots. *UR5e*. 2023. URL: https://www.universal-robots.com/products/ur5-robot/.
[79] ThermoFisher Scientific. *Nunc EasYFlask Cell Culture Flask*. 2023. URL: https://www.thermofisher.com/order/catalog/product/159910.
[80] ThermoFisher Scientific. *Gibco Bottle, 500 mL*. 2023. URL: https://www.thermofisher.com/order/catalog/product/10340001.

[81] Robotiq. *Hand-E Adaptive Gripper*. 2023. URL: https://robotiq.com/products/hand-e-adaptive-robot-gripper.
[82] Avantor. *VWR INCU-Line ILCO 180 Premium, CO2 Incubator*. 2023. URL: https://in.vwr.com/store/product/36173394/vwr-incu-line-ilco-180-premium-co2-incubator.
[83] Intel. *Intel RealSense Depth Camera D415*. 2023. URL: https://www.intelrealsense.com/depth-camera-d415/.
[84] DH Robotics. *RGI Series. Rotary Electric Gripper*. 2023. URL: https://en.dh-robotics.com/product/rgi.
[85] CytoSMART Technologies B.V. *CytoSMART Lux2 Duo Kit Label-free microscopy with high image quality*. Tech. rep. 2021. URL: https://cytosmart.com/sites/default/files/resources/Lux3%20BR%20brochure%20-%20CytoSMART%20technologies.pdf.
[86] Solid State Cooling Systems. *TCube edge*. 2023. URL: https://www.sscooling.com/product/t-cube-edge/.
[87] SDU. *Universal Robots RTDE C++ Interface*. 2023. URL: https://sdurobotics.gitlab.io/ur_rtde/#.
[88] Universal Robots. *Real-Time Data Exchange (RTDE) Guide*. 2023. URL: https://www.universal-robots.com/articles/ur/interface-communication/real-time-data-exchange-rtde-guide/.
[89] Sagi Eppel. *TransProteus Dataset*. 2023. URL: https://e.pcloud.link/publink/show?code=kZfx55Zx1GOrl4aUwXDrifAHUPSt7QUAIfV.
[90] Liang-Chieh Chen et al. "Rethinking Atrous Convolution for Semantic Image Segmentation". In: (June 2017). URL: http://arxiv.org/abs/1706.05587.
[91] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: (Dec. 2015). URL: http://arxiv.org/abs/1512.03385.
[92] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: (May 2015). URL: http://arxiv.org/abs/1505.04597.
[93] Alexandros Graikos et al. "Single image-based food volume estimation using monocular depth-prediction networks". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 12189 LNCS. Springer, 2020, pp. 532–543. ISBN: 9783030491079. DOI: 10.1007/978-3-030-49108-6{_}38.
[94] Alexei Botchkarev. "Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology". In: *Interdisciplinary Journal of Information, Knowledge, and Management* 14 (2019), p. 4579. DOI: https://doi.org/10.28945/4184.
[95] Miles MacKlin et al. "Unified particle physics for real-time applications". In: *ACM Transactions on Graphics*. Vol. 33. 4. Association for Computing Machinery, 2014. DOI: 10.1145/2601097.2601152.
[96] Zherong Pan, Chonhyon Park, and Dinesh Manocha. "Robot Motion Planning for Pouring Liquids". In: *Proceedings of the Twenty-Sixth International Conference on Automated Planning and Scheduling*. 2016, pp. 518–526.
[97] Mayank Mittal et al. "ORBIT: A Unified Simulation Framework for Interactive Robot Learning Environments". In: (Jan. 2023). URL: http://arxiv.org/abs/2301.04195.

A GitHub Repository and Reproducibility

The GitHub repository containing all the code of the project can be found under the following link:

https://github.com/DaniSchober/LabLiquidVision.

The repository is divided into four different subfolders:

- System Prototype for Cell Culture Automation: *cell_culture_automation*
- Vessel and Liquid Segmentation and Depth Estimation *segmentation_and_depth*
- Liquid Volume Estimation: *volume_estimation*
- Pouring Simulation: *pouring_simulation*

The *cell_culture_automation* subfolder also contains the drawings of the 3D printed parts. The *pouring_simulation* subfolder includes videos of the simulation and the simulation-to-reality transfer.

To reproduce the described results, do the following:

```
git clone https://github.com/DaniSchober/LabLiquidVision

conda env create -f environment.yml

conda activate labliquidvision
```

Detailed descriptions of how to train and test models, run a prediction on a model, use the simulator, and use the system prototype are included in the specific subprojects. The specific requirements are also included in the *ReadMe* of the subfolders.

All the models and the *LabLiquidVolume* dataset are stored in Google Cloud. To get them, go to the specific subfolder, and run:

```
dvc pull
```

To only get the models, run this inside the specific subfolder:

```
dvc pull models
```

To only download the *LabLiquidVolume* dataset, run this in the *volume_estimation* subfolder:

```
dvc pull data
```

Implementation Details of the Segmentation and Depth Estimation

The `main.py` file serves as the main entry point for running the segmentation and depth prediction. It provides three modes of operation: training the model, predicting the segmentation and depth of an image, and evaluating the model on a folder of images.

To run the script, the following command-line arguments can be used:

- `--mode`: Specifies the mode of operation, which can be `train`, `predict`, or `evaluate`.
- `--cuda`: Specifies whether to use CUDA for GPU acceleration. It accepts either `True` or `False`.
- `--batch_size`: Specifies the batch size for training (applicable only in the `train` mode).

- `--epochs`: Specifies the number of epochs for training (applicable only in the `train` mode).
- `--load_model`: Specifies whether to load a pretrained model. It accepts either `True` or `False`.
- `--use_labpics`: Specifies whether to use lab pictures for training. It accepts either `True` or `False`.
- `--model_path`: Specifies the path to the trained model file.
- `--image_path`: Specifies the path to the image for prediction (applicable only in the `predict` mode).
- `--folder_path`: Specifies the folder path for evaluation (applicable only in the `evaluate` mode).

Exemplary usage and further information are described in the ReadMe of the folder. The main training code is provided in the `train.py` file.

Implementation Details of the Liquid Volume Estimation

To get the two main models (`volume_model_no_vol.pth`, `volume_model_with_vol.pth`), run `dvc pull models` inside this subproject. The models get saved in `volume_estimation/models`.

This repository contains code for training, predicting, and testing models for volume estimation of liquids and possibilities for new data generation and data conversion to include the segmentation and depth maps. The main file, `main.py`, handles these operations based on the provided arguments.

The `main.py` can be used with the following command:

```
python main.py --mode <mode> [--image_path <image_path>] [--folder_path <folder_path>]
[--num_epochs <num_epochs>] [--vessel_volume <vessel_volume>] [--use_vessel_volume]
[--no_GPU] [--path_input <path_input>] [--path_output <path_output>]
[--model_path <model_path>]
```

The available modes are:

- `train`: Trains a model. If `--use_vessel_volume` is used, a model that takes the volume of the vessel as an additional input is trained. Otherwise, only the segmented depth maps of liquid and vessel are used as input.
- `predict`: Predicts the volume of liquid in an RGB image. The image can be changed by using `--image_path`. If `--use_vessel_volume` is used, the model, which includes the vessel volume input, is used. The vessel volume can be provided using `--vessel_volume`.
- `test`: Tests the model. If `--use_vessel_volume` is used, the model that takes the volume of the vessel as an additional input is tested. Otherwise, only the model trained with the segmented depth maps of liquid and vessel is tested. The folder for testing can be changed using `--folder_path`.
- `record`: Records new samples for the dataset. A user interface is opened to simplify the data generation. The vessel name, liquid volume, and liquid color need to be provided by the user. Every sample gets saved in a new subfolder in `data/interim`.
- `convert`: Converts the generated data to the processed dataset. The path for the input data can be defined using `--path_input`, the path for the processed data using `--path_output`. The segmentation and depth maps are generated using the model defined using `--model_path`.

- `predict_on_depth_maps`: Predicts the volume of liquid based on already generated segmented depth maps of liquid and vessel. A random sample from `data/processed` will be selected for the prediction.

Optional arguments:

- `--no_GPU`: Disables GPU usage for prediction.
- `--image_path`: Path to the image for prediction (default: ”example/image.png”).
- `--folder_path`: Path to the folder containing the converted LabLiquidVolume dataset (default: ”data/processed/”).
- `--use_vessel_volume`: If that is selected, the training, testing, and predictions are made using the volume of the vessel (in mL) as an additional input.
- `--vessel_volume`: Volume of the vessel as input for the prediction when `--use_vessel_volume` is used (default: 0).
- `--num_epochs`: Number of epochs to train for (default: 200).
- `--model_path`: Path to the model for dataset conversion (default: ”../segmentation_and_depth/models/se
- `--path_input`: Path to the input dataset for conversion (default: ”data/interim”).
- `--path_output`: Path to the output dataset for conversion (default: ”data/processed”).

Please note that some modes require additional arguments. Make sure to provide the necessary arguments based on the chosen mode. Exemplary usage and further information are described in the ReadMe of the folder. The main training code is provided in the `train.py` file.

B Appendix

B.1 LabLiquidVolume Dataset

Name in Dataset	Name	Company	Material	Height (mm)	Opening Diameter (mm)	Max. Width (mm)
Cell_Flask _160mL	T75 EasYFlask	Thermo Fisher	Plastic	150	24	81
Cell_Flask _400mL	T175 EasYFlask	Thermo Fisher	Plastic	228	114	26
Duran_100mL	DURAN GL 45 Blue Laboratory Bottle 100 mL	Duran	Glas	106	17	50
Duran_250mL	DURAN GL 45 Blue Laboratory Bottle 250 mL	Duran	Glas	139	30	71
Duran_500mL	DURAN GL 45 Blue Laboratory Bottle 500 mL	Duran	Glas	176	30	85
Gibco_125mL	Gibco Bottle, 100 mL	Thermo Fisher	Plastic	94	37	50
Gibco_500mL	Gibco Bottle, 500 mL	Thermo Fisher	Plastic	149	37	100
Pyrex_100mL	PYREX Griffin Low Form 100 mL Beaker	Corning	Glas	68	54	55
StorageBottle _250mL	Corning 250 mL Easy Grip Polystyrene Storage Bottle	Corning	Plastic	117	38	74
StorageBottle _500mL	Corning 500 mL Easy Grip Polystyrene Storage Bottles	Corning	Plastic	139	38	94
Tube_15mL	Corning 15mL Centrifuge Tubes	Corning	Plastic	118	15	17
Tube_50mL	Corning Self-standing 50 mL Centrifuge Tube	Corning	Plastic	115	27	29

Table B.1: Details of the vessels in the *LabLiquidVolume* dataset.

Figure B.1: 20 samples of the recorded RGB images of the *LabLiquidVolume* dataset.

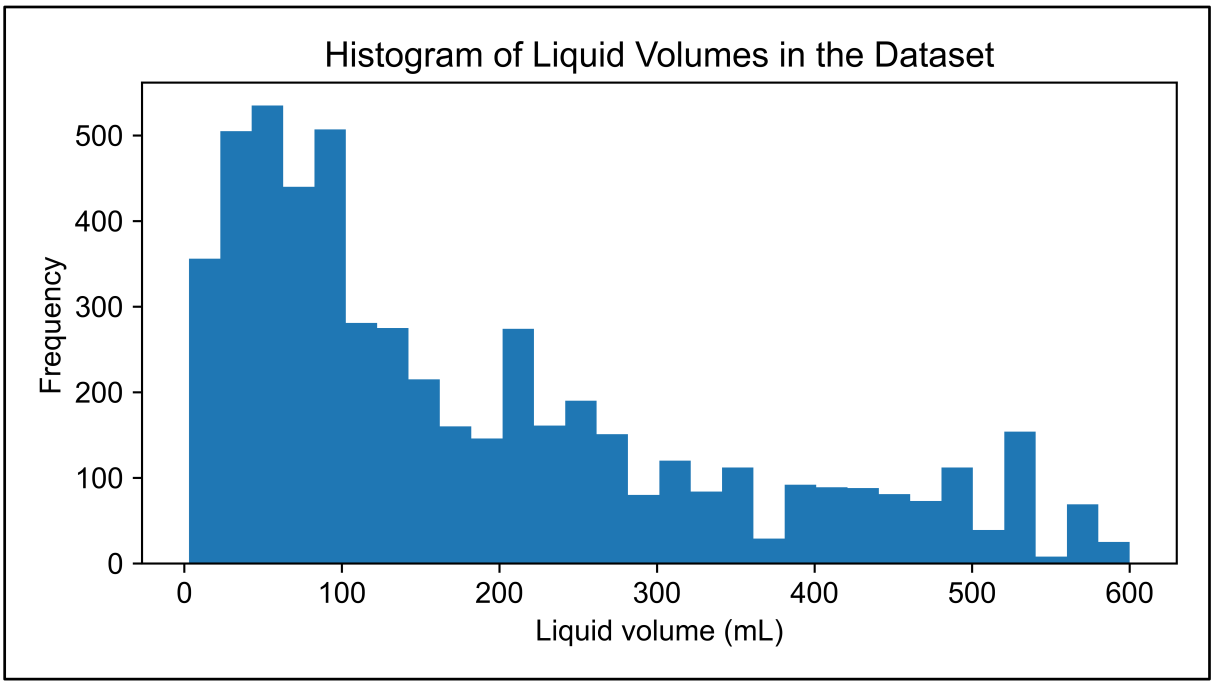


Figure B.2: Histogram of the distribution of volumes of liquids in the *LabLiquidVolume* dataset.

B.2 Volume Estimation

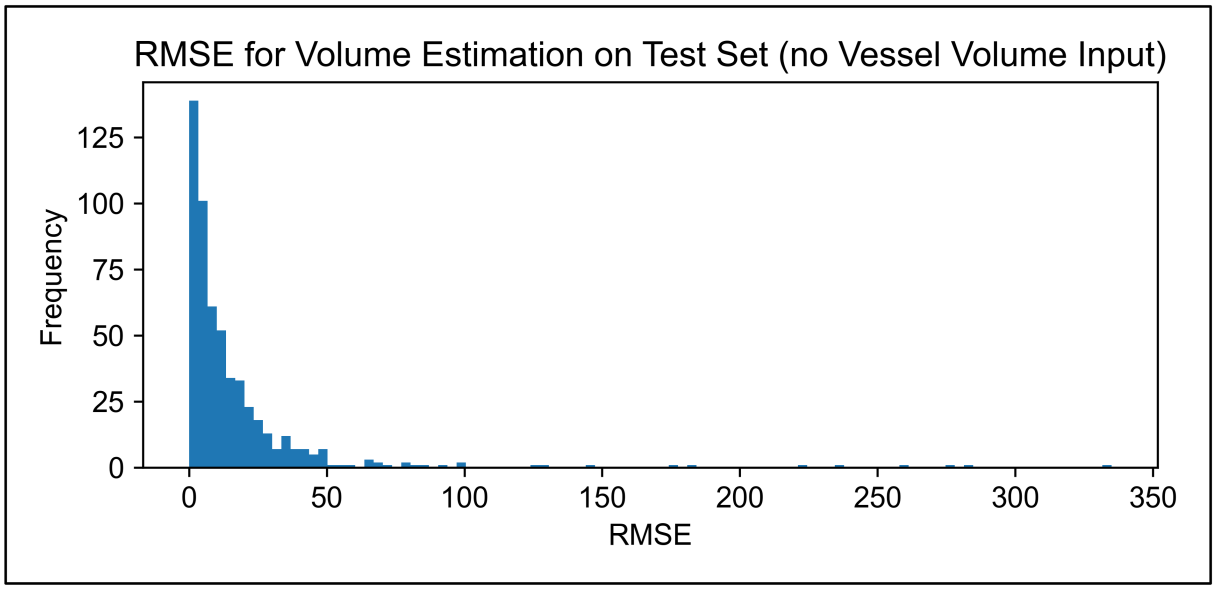


Figure B.3: Histogram of the RMSE results from the testing on the test set of the *LabLiquidVolume* dataset for the model without vessel volume input (only segmented depth maps).

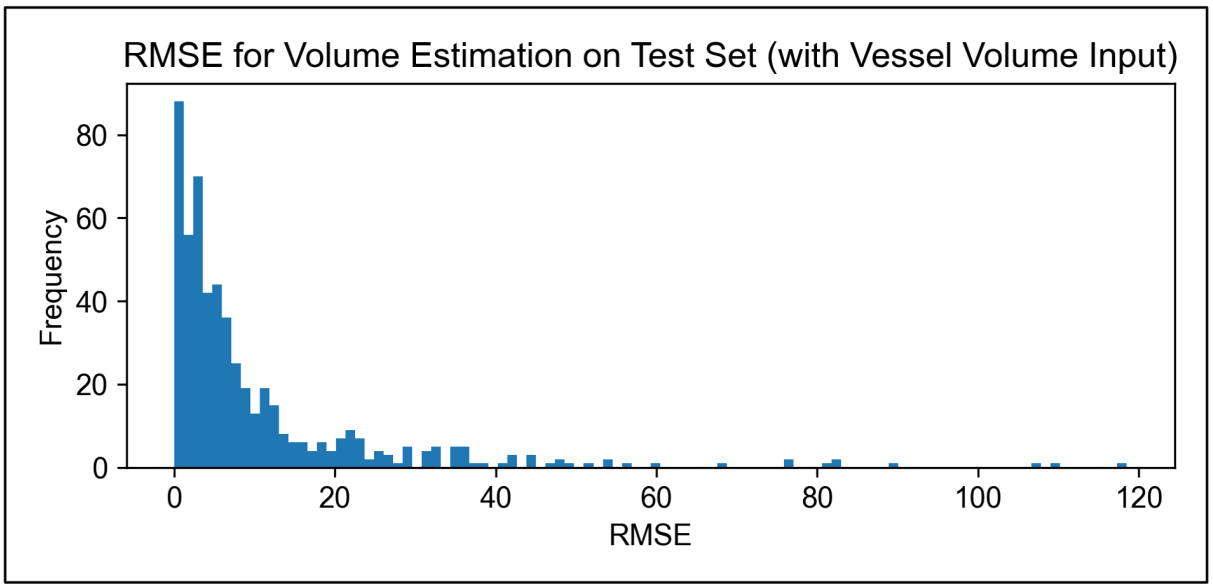


Figure B.4: Histogram of the RMSE results from the testing on the test set of the *LabLiquidVolume* dataset for the model with vessel volume input.

B.3 Pouring Simulation

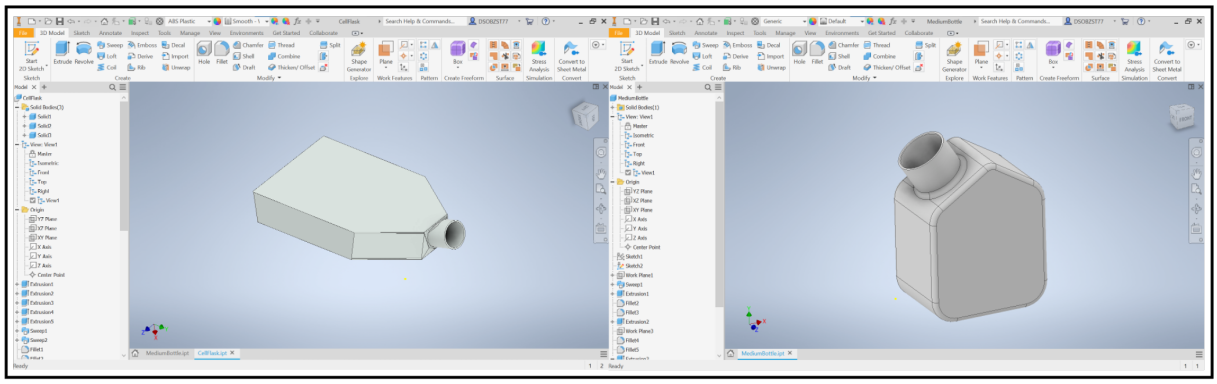


Figure B.5: Visualization of the created 3D models of the cell culture flask and the washing solution/media bottle in Autodesk Inventor.

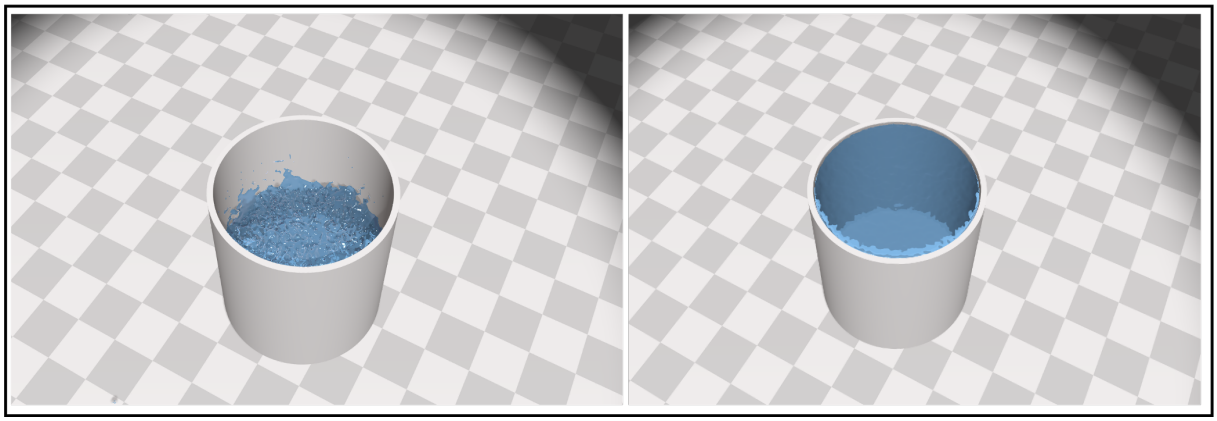


Figure B.6: Visualization of the filling of the container in the calibration scene of the simulation.

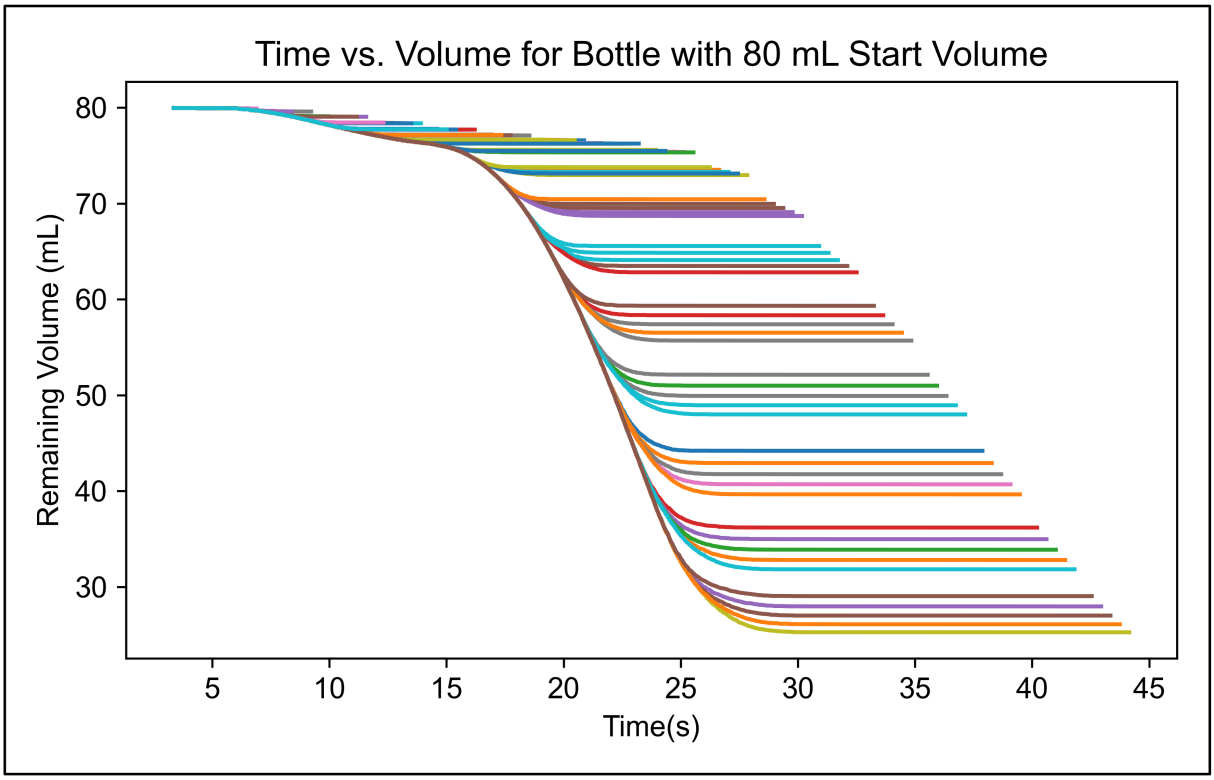


Figure B.7: Visualization of the course of the remaining volume in the media/washing solution bottle with a starting volume of 80 mL over time for the 155 simulated pours.

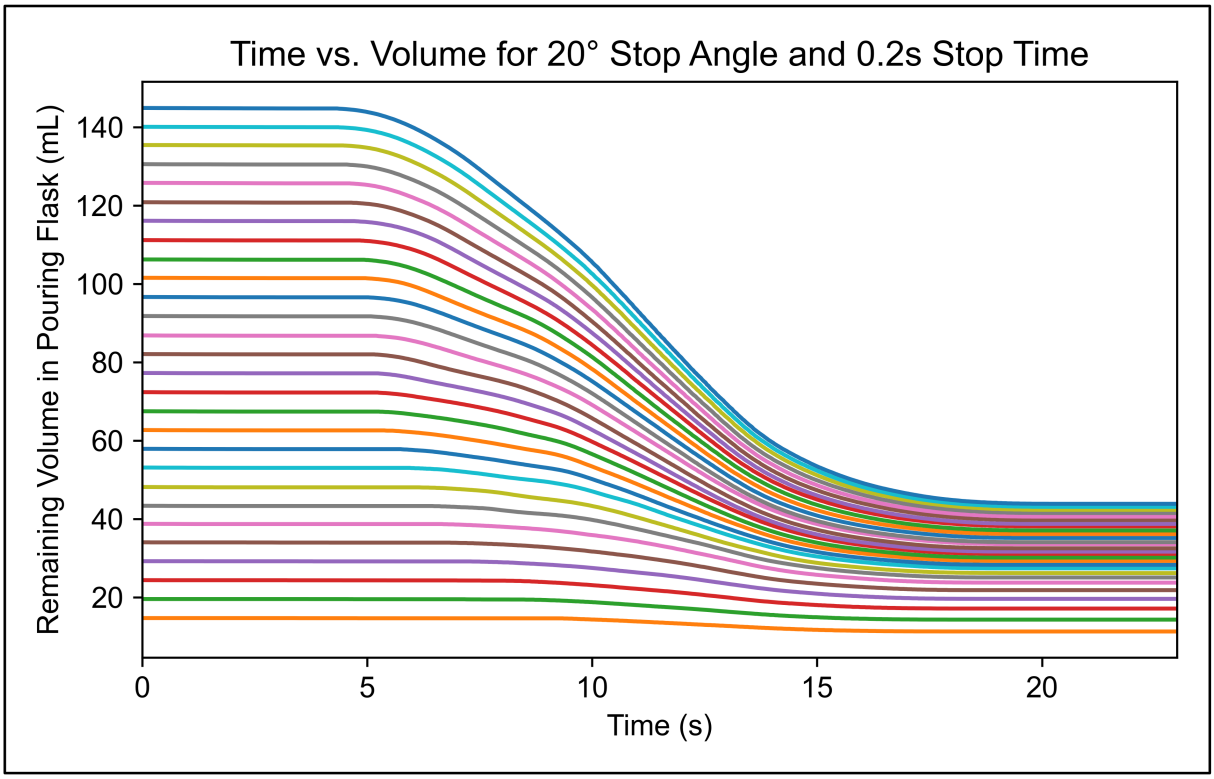


Figure B.8: Visualization of the course of the remaining volume in the cell culture flask for different start volumes and a fixed stop angle and stop time.

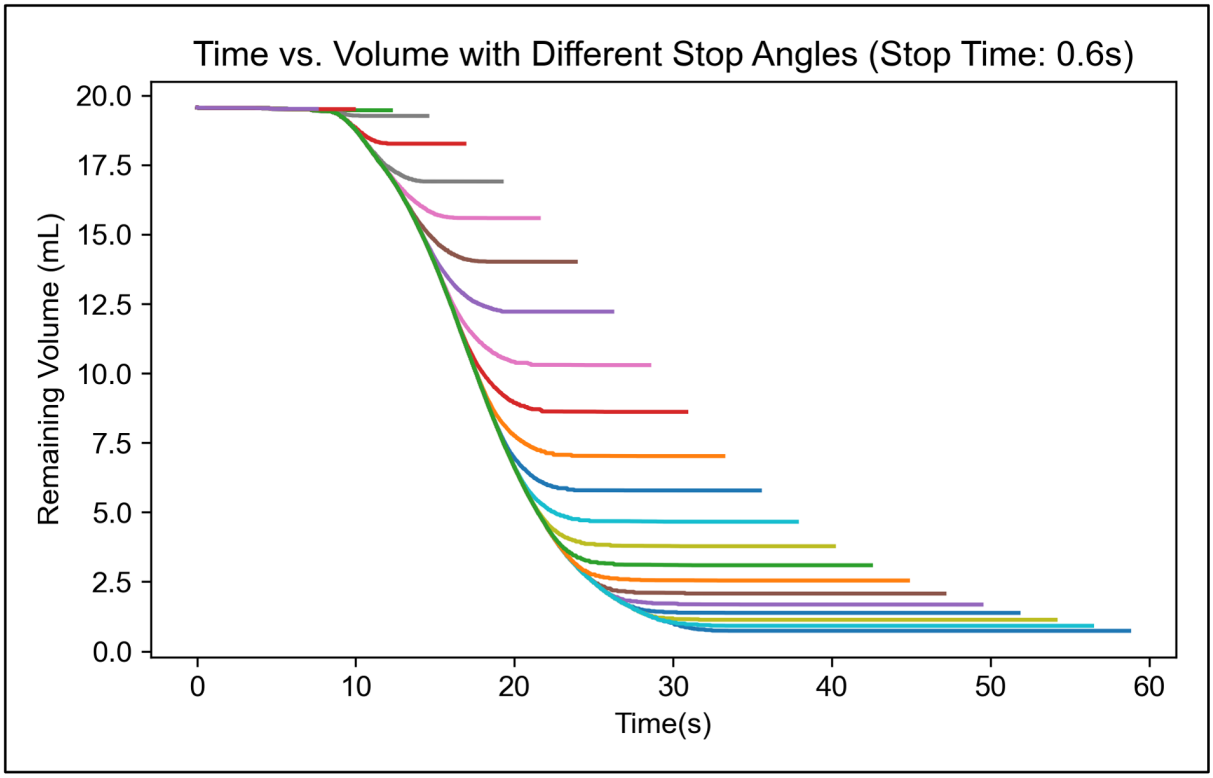


Figure B.9: Visualization of the course of the remaining volume in the cell culture flask for different stop angles and a fixed stop time and start volume.

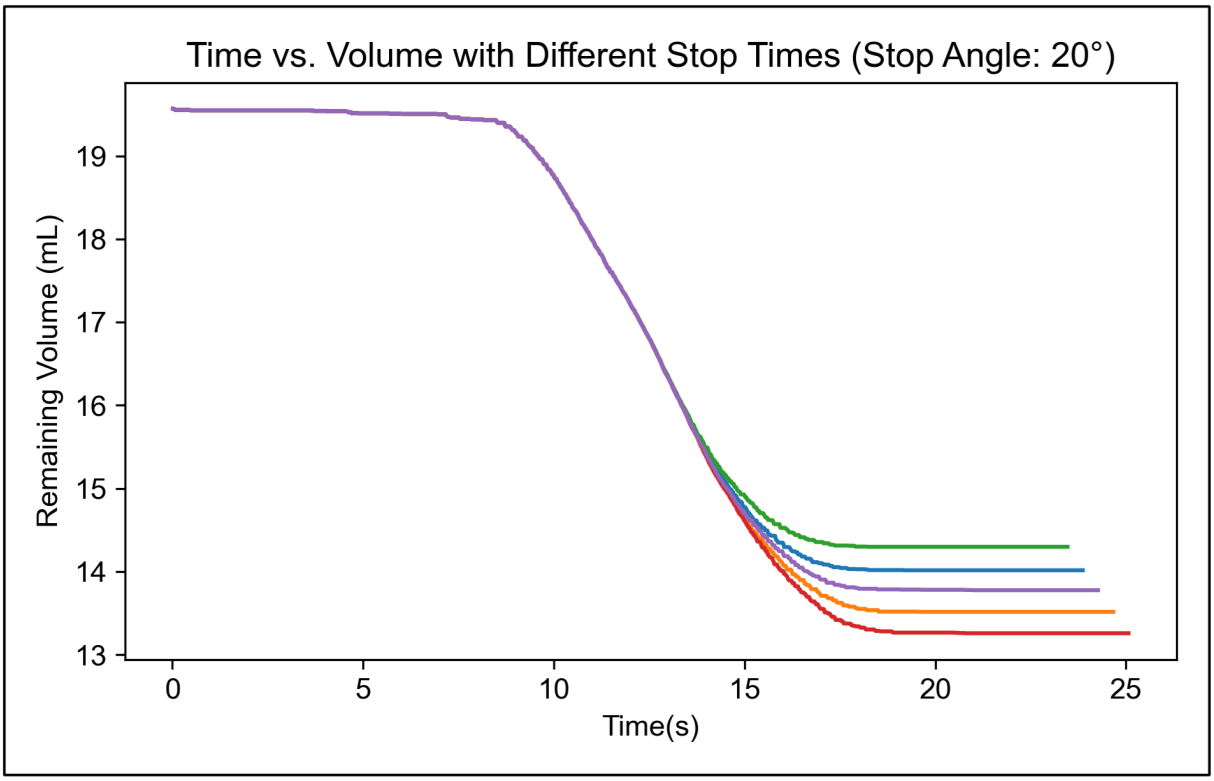


Figure B.10: Visualization of the course of the remaining volume in the cell culture flask for different stop times and a fixed stop angle and start volume.

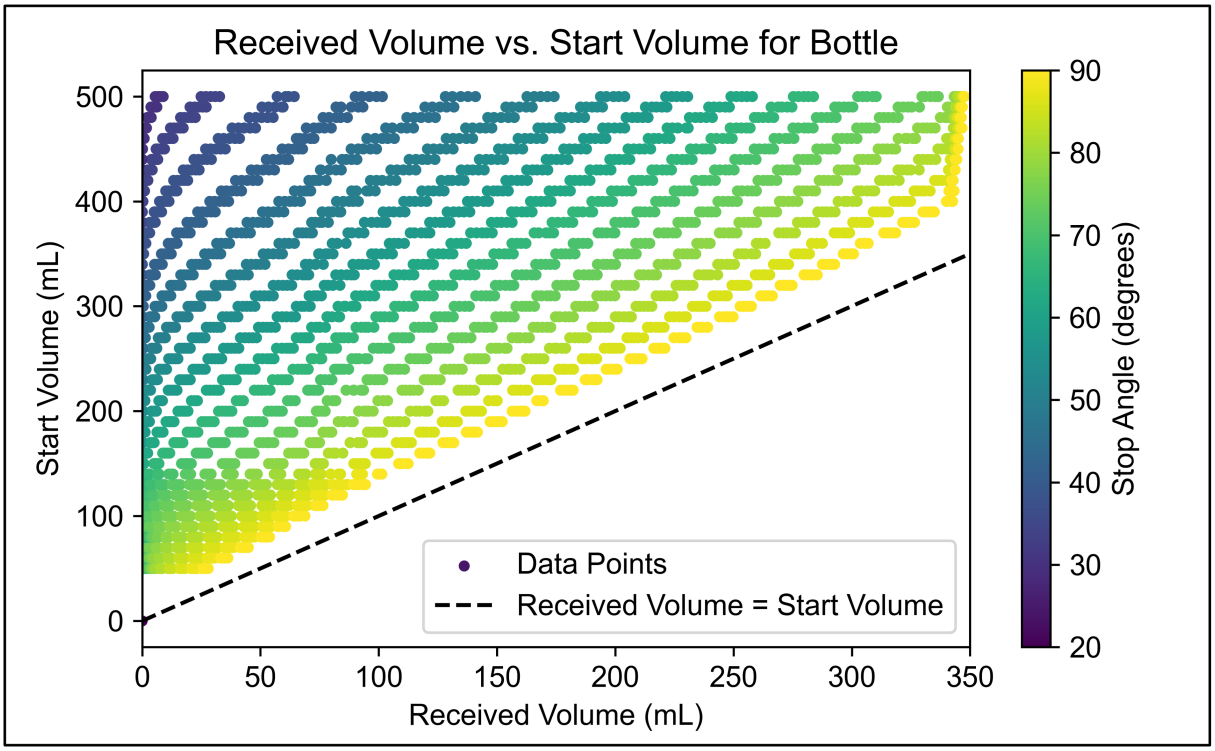


Figure B.11: Visualization of the start and received volumes for all simulated media/washing solution bottle pouring movements.

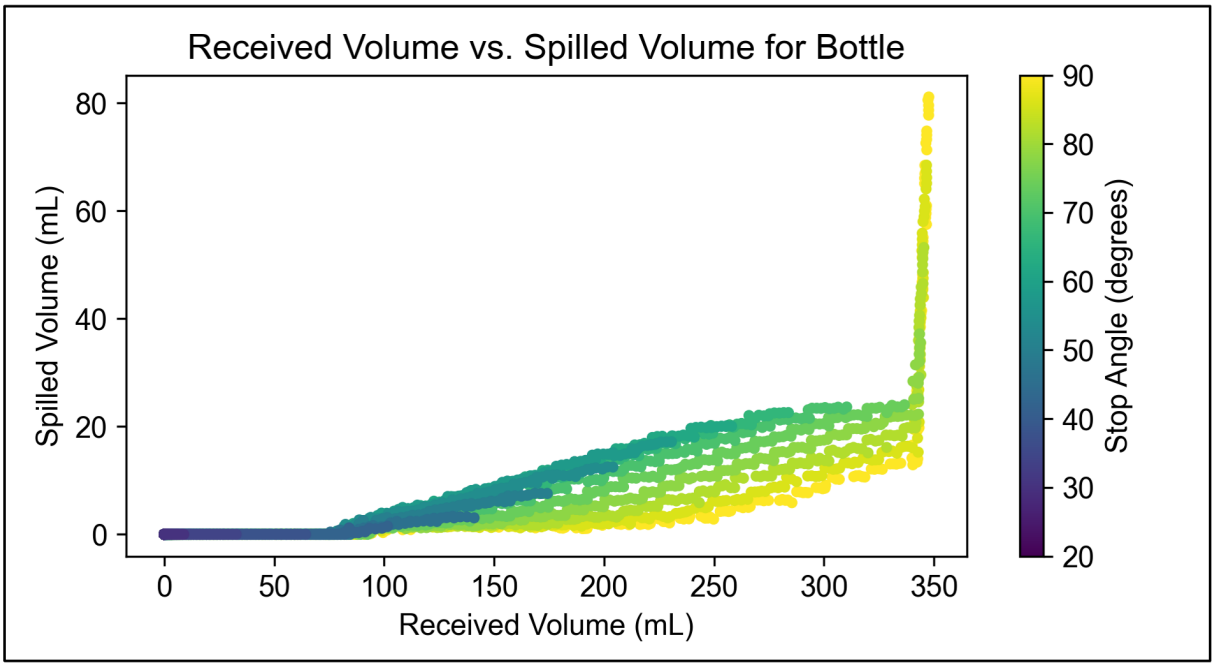


Figure B.12: Visualization of spilled volume in relation to the received volume inside the receiving container for the simulated pours with the media/washing solution bottle.

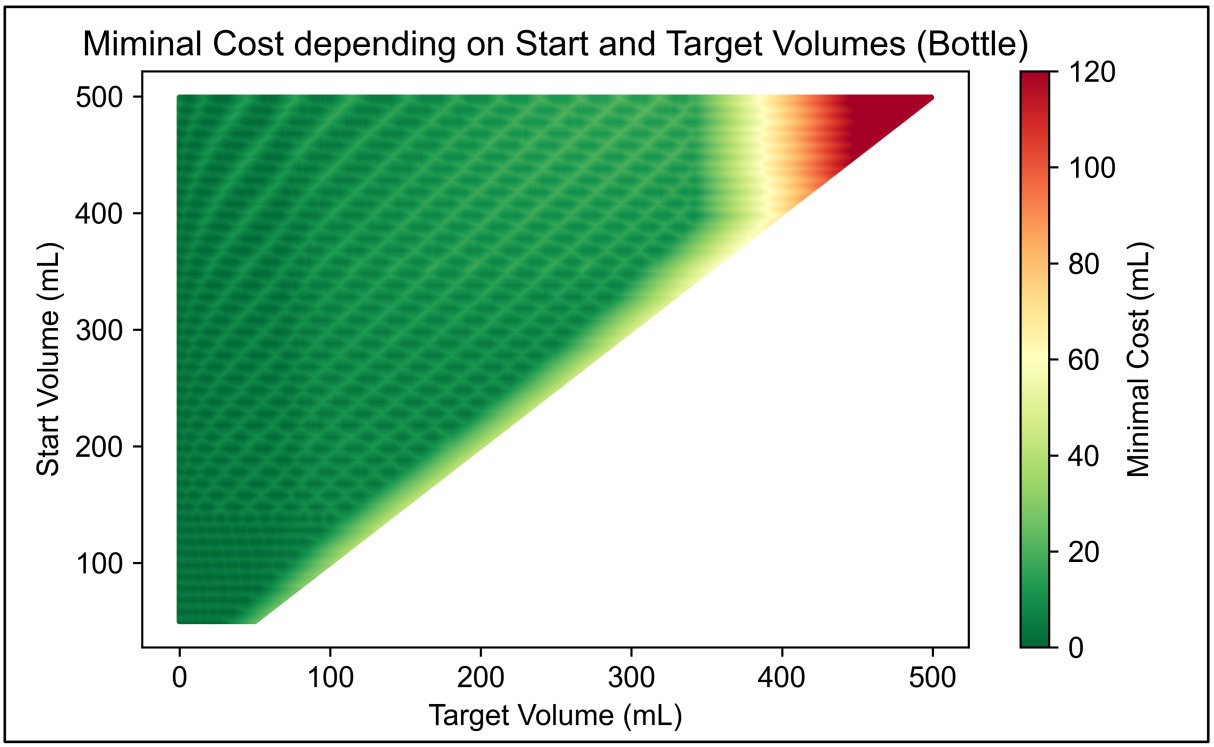


Figure B.13: Visualization of the minimal cost for a simulated pour with a media/washing solution bottle given a defined start and target volume with step sizes of 1 mL.

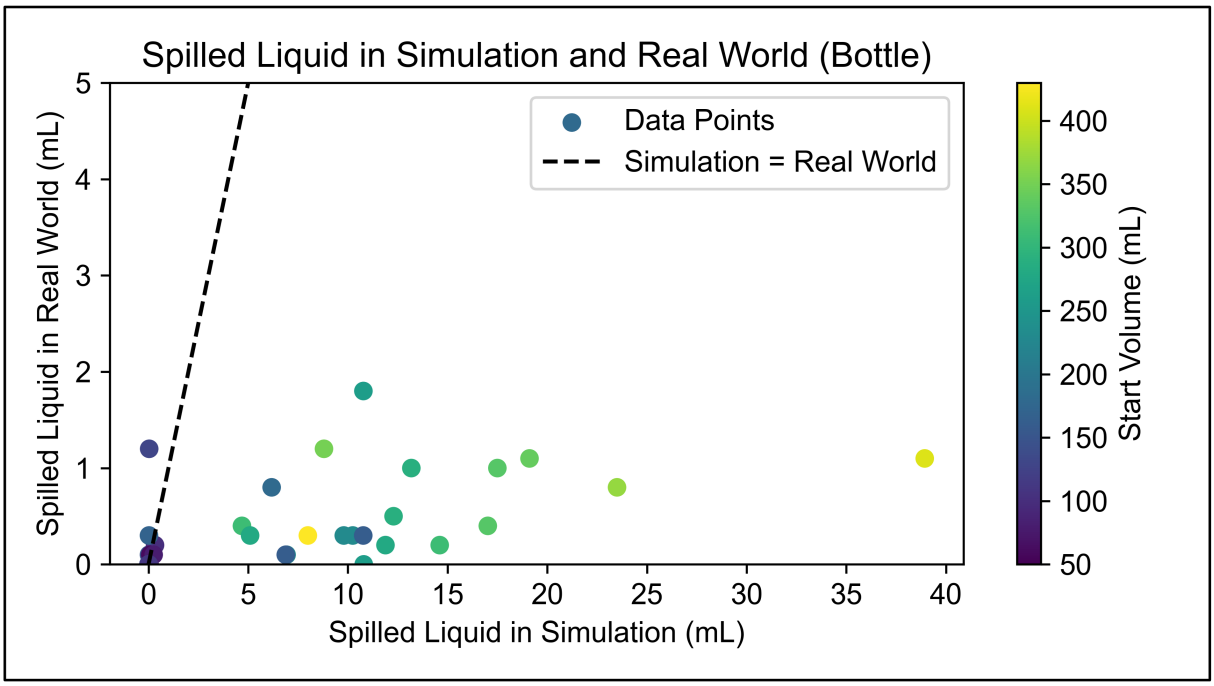


Figure B.14: Results of the spilled volume in the real pouring experiments executed with a UR5e and a Gibco 500 mL bottle as pouring container.

Table B.2: Detailed analysis of the results space for the different start volumes for the simulated pours with the media bottle.

Start Volume (mL)	Max. Difference (mL)	Mean Difference (mL)	Coverage (%)
50	2.80	0.17	82.23
60	3.23	0.20	80.56
70	3.68	0.25	73.33
80	3.73	0.30	68.52
90	4.12	0.34	60.66
100	4.34	0.38	59.42
110	4.70	0.43	61.84
120	4.63	0.47	58.82
130	4.56	0.53	55.32
140	7.57	0.80	37.62
150	14.28	1.42	29.46
160	14.63	1.54	28.69
170	15.30	1.67	28.03
180	15.65	1.79	27.46
190	15.75	1.93	26.32
200	16.06	2.05	25.93
210	20.42	3.20	16.47
220	16.83	2.37	26.37
230	17.57	2.43	25.0
240	17.15	2.54	25.37
250	17.86	2.68	24.17
260	17.92	2.80	23.98
270	18.15	2.91	23.48
280	17.77	3.03	23.01
290	17.98	3.16	22.4
300	19.69	3.30	22.48
310	17.94	3.37	22.56
320	18.20	3.48	22.18
330	18.52	3.61	22.46
340	18.18	3.71	21.84
350	18.26	3.81	21.59
360	20.88	5.12	16.29
370	20.26	4.08	20.69
380	18.87	4.13	21.1
390	19.97	4.24	20.6
400	21.20	4.34	19.83
410	22.33	4.34	20.12
420	21.56	4.34	19.83
430	22.00	4.35	19.19
440	23.22	4.36	19.48
450	24.00	4.37	19.13
460	24.02	4.36	19.42
470	22.99	4.38	19.02
480	25.14	4.36	18.21
490	26.2	4.34	18.44
500	26.3	4.39	18.34

B.4 System Prototype

Figure B.15: Pictures of the Gibco Bottles 500 mL [80] from ThermoFisher Scientific for media and wash solution and the Nunc EasYFlask Cell Culture Flask [79] from ThermoFisher Scientific (Catalog number: 159920).

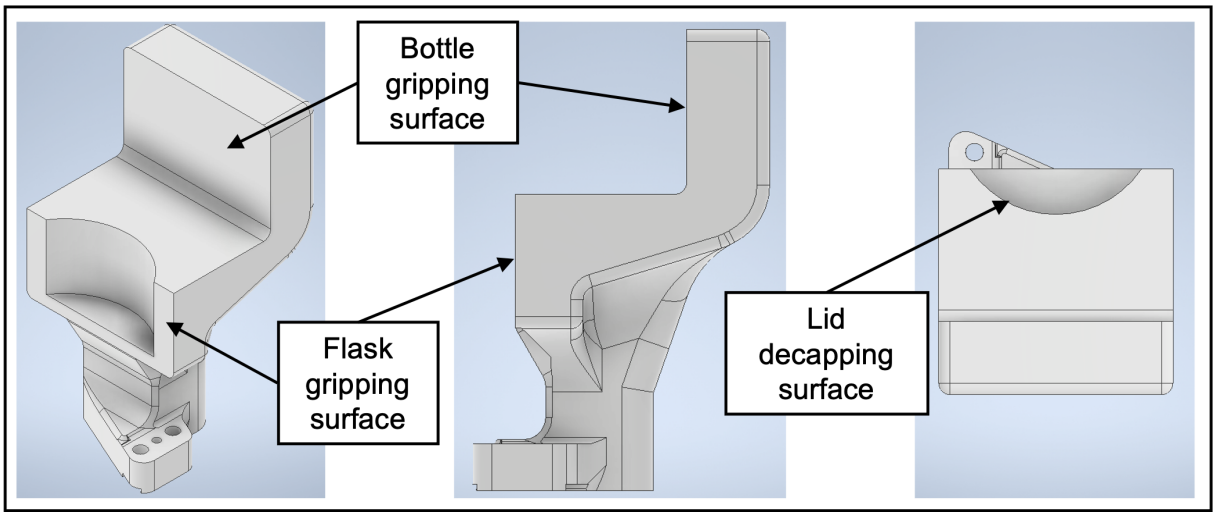


Figure B.16: Visualisations of the details of the gripper finger design. From left to right: (1) Isometric view in Autodesk Inventor. (2) Side view. (3) Top view.

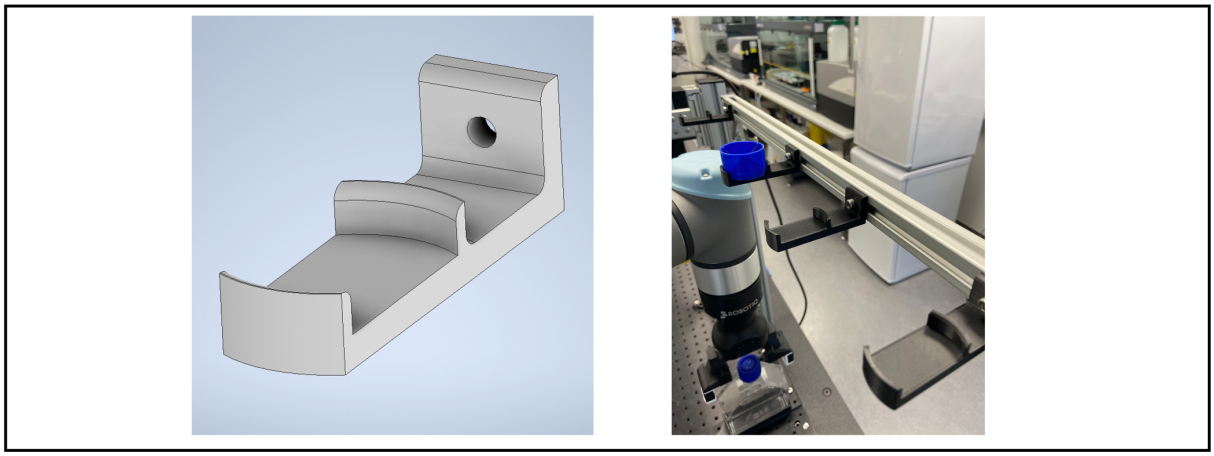


Figure B.17: Visualisations of the lid holder. From left to right: (1) Isometric view of the lid holder part in Autodesk Inventor. (2) The lid holders attached to the aluminum structure.

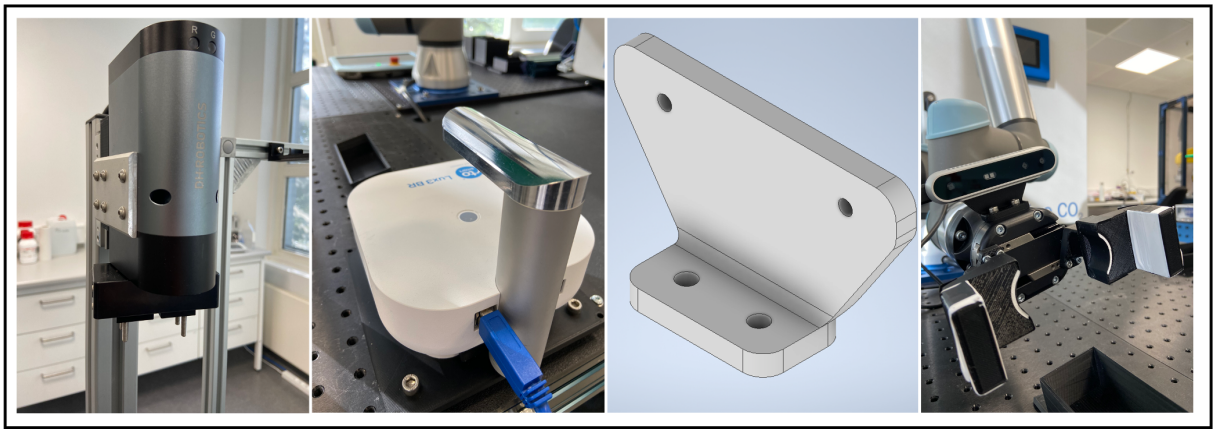


Figure B.18: Visualisations of the different components of the system. From left to right: (1) Infinite rotating gripper module for capping/decapping the bottles. (2) The microscope in the system. (3) Isometric view of the camera mount in Autodesk Inventor. (4) The camera mount, including the camera attached to the gripper.

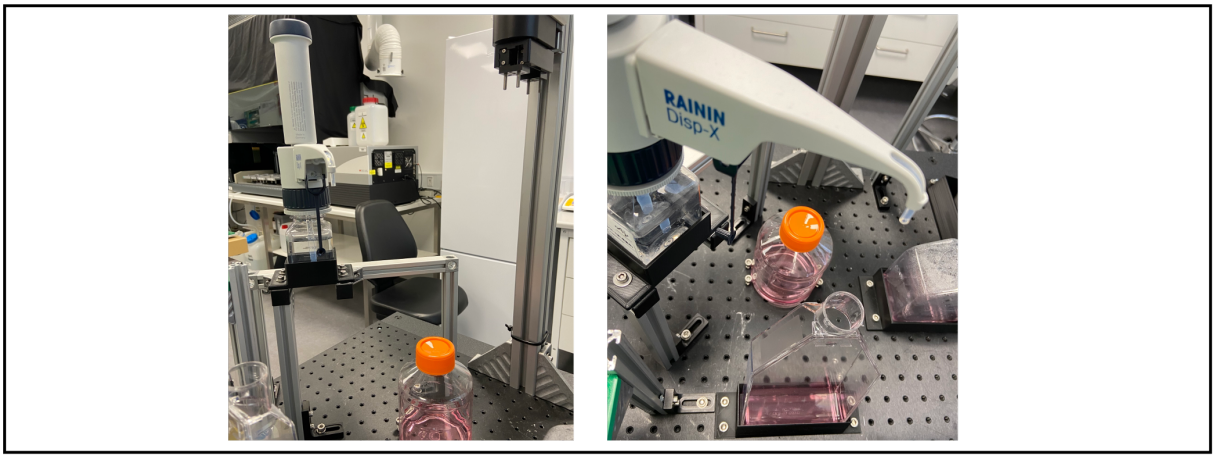


Figure B.19: Visualisations of the trypsin station including a Disp-X Bottle Dispenser 1-10mL (30373528).

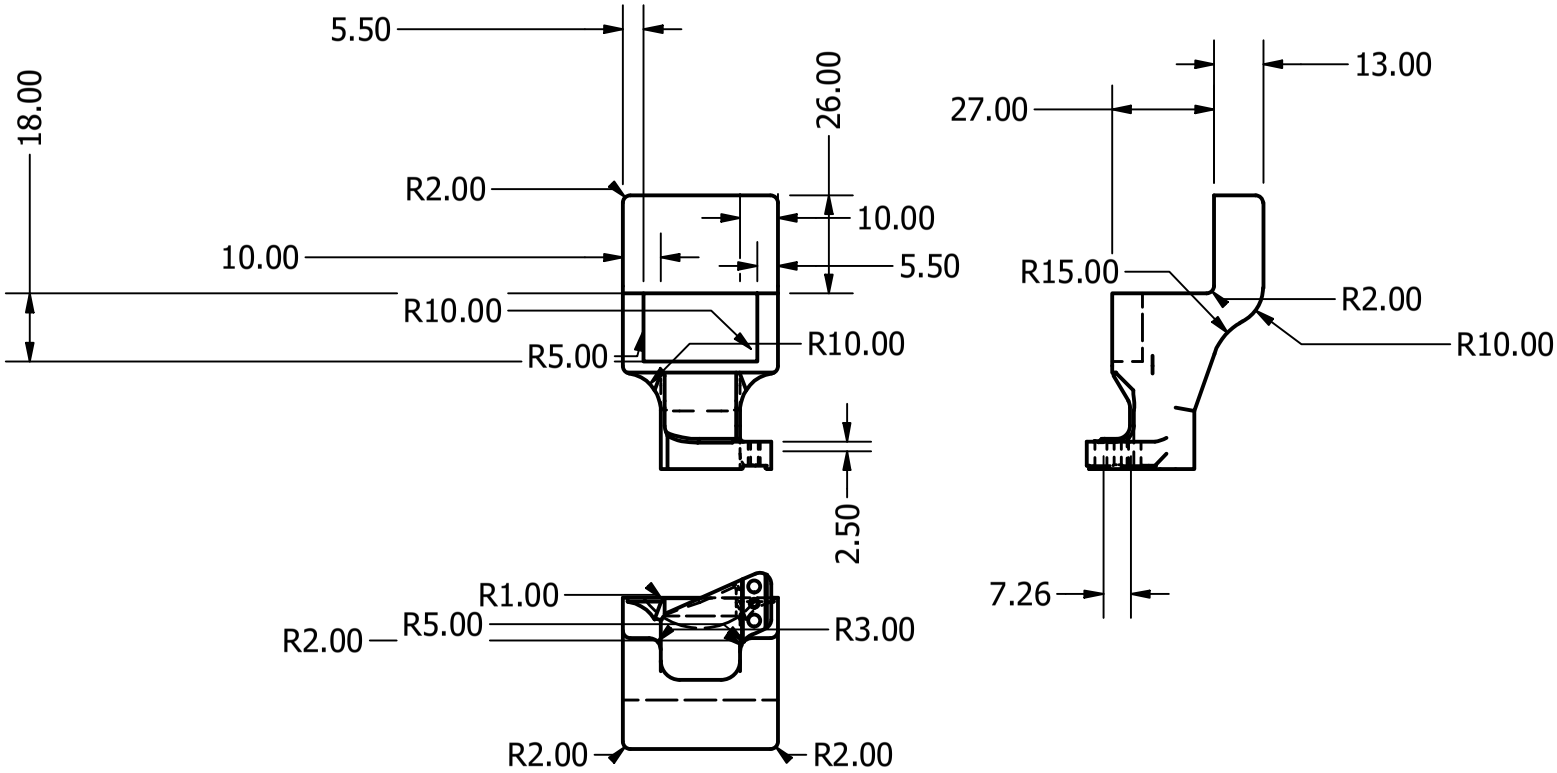
Table B.3: Equipment list for the system prototype.

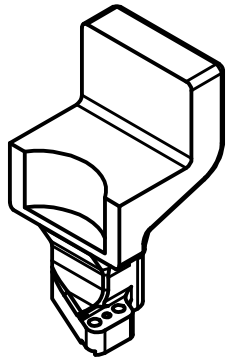
Quantity	Equipment
3	Festo linear/swivel clamp (8072624)
1	Universal Robots UR5e
1	VWR INCU-Line ILCO 180 Premium, CO_2 Incubator
1	Disp-X Bottle Dispenser 1-10mL (30373528)
1	RGI 100 Rotary Electric Gripper Jaw
1	CytoSMART Lux3 BR
1	Festo ISO cylinder DSNU-25-500-PPV-A
1	Festo Compressed air filter regulator LFR (8003637)
1	Vention table (121 x 198 cm)
1	TCube Edge Thermoelectric Chiller (52-14072-1)
2	Festo Solenoid Valve (VUVS-LK20-B52-D-G18-1C1-S)
1	Festo Basic valve LR-D-MINI (546430)
1	Robotiq Hand-E Adaptive Gripper

Figure B.20: Multiple perspectives of the system setup on the table.

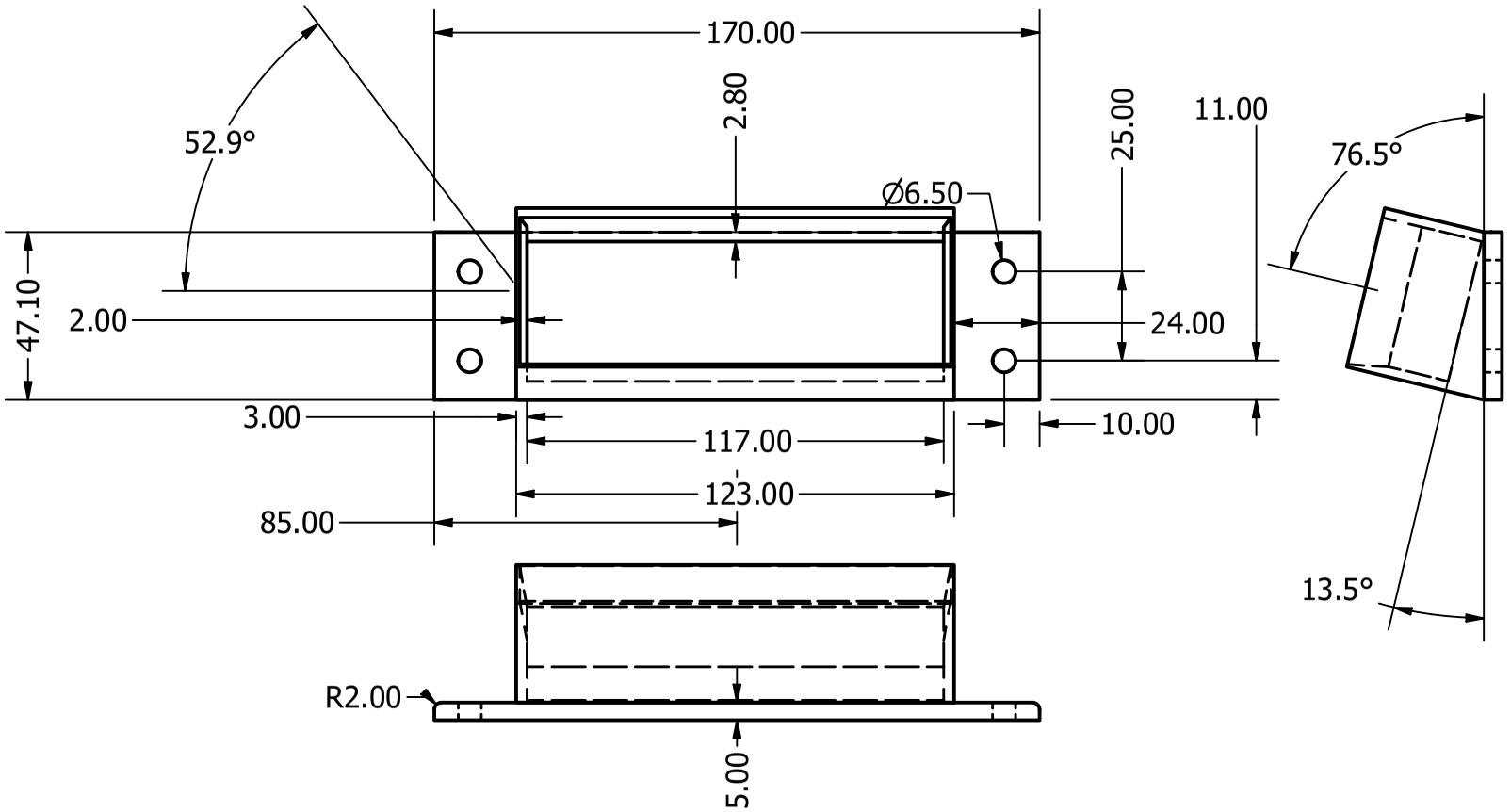
Table B.4: Cost overview for rebuilding the system.

Quantity	Equipment	Price (DKK)
3	Festo linear/swivel clamp (80726224)	4,399.08
1	Universal Robots IJR5e	191,093.00
1	VWR INCU-Line ILCO 180 Premium, CO2 Incubator	63,727.50
1	Disp-X Bottle Dispenser 1-10mL (30373528)	2,630.91
1	RGI 100 Rotary Electric Gripper Jaw	19,400.00
1	CytoSMART Lux3 BR	80,337.65
1	Festo ISO cylinder DSNU-25-500-ppv-A	508.15
1	Festo Compressed air filter regulator LFR (8003637)	382.43
1	Robotiq Hand-E Adaptive Gripper	29,835.00
1	Cooling Circulator and ThermoWraps	39,328.45
	Total	431,642.17

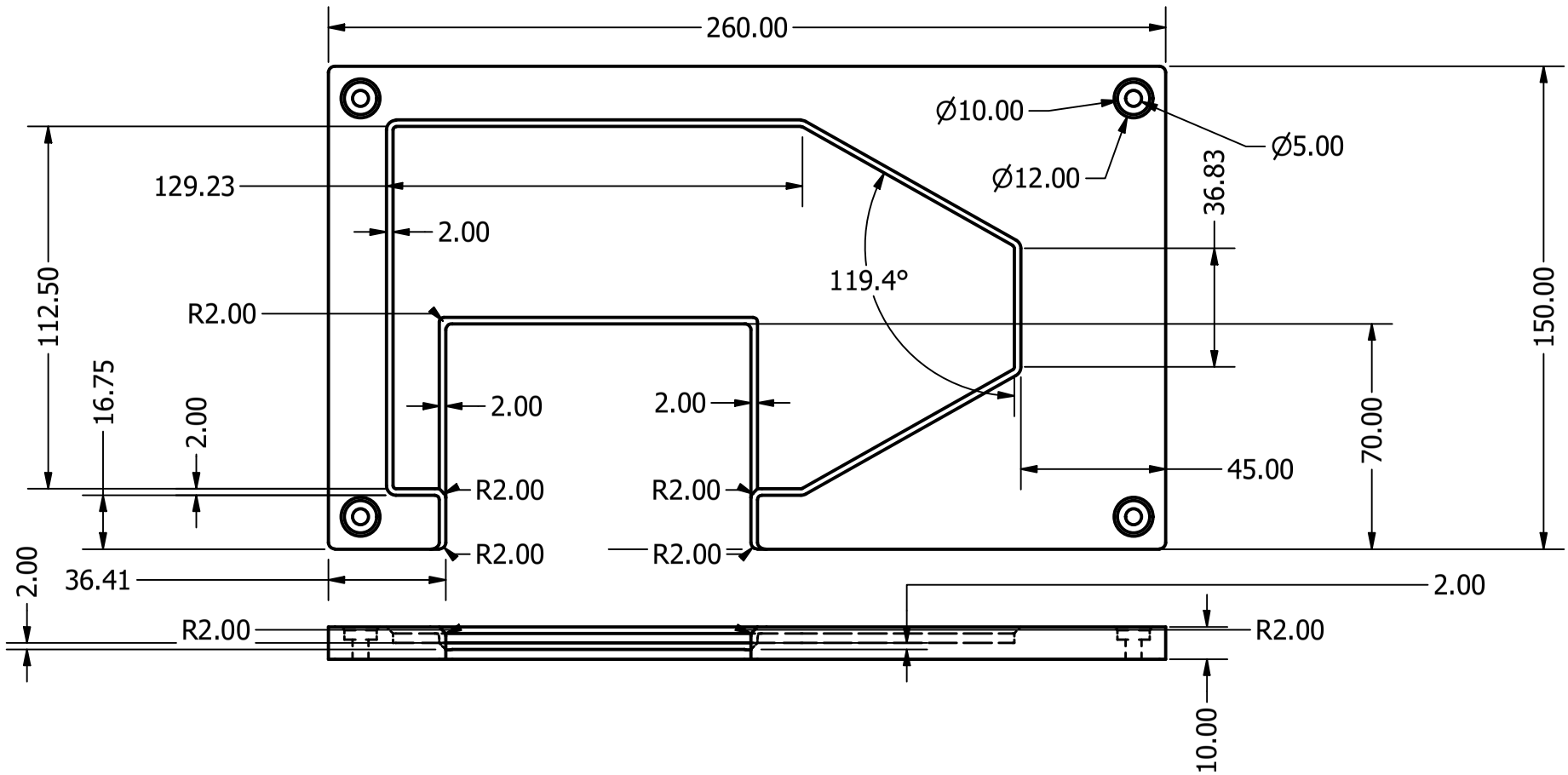




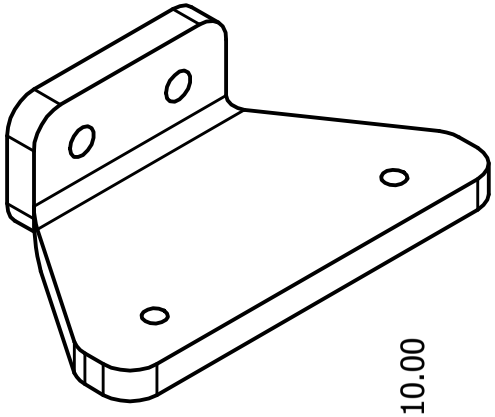
DRAWN DSOB	03-06-2023	TITLE Gripper Fingers		
CHECKED				
QA				
MFG				
APPROVED				
		SIZE A4	DWG NO Gripper	REV
		SCALE 1 / 2		SHEET 1 OF 1

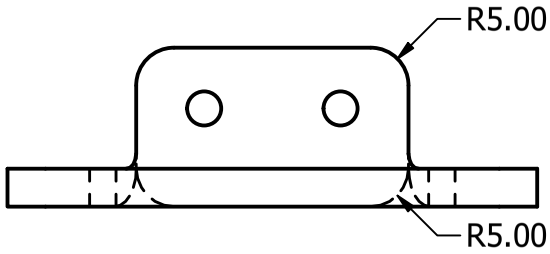


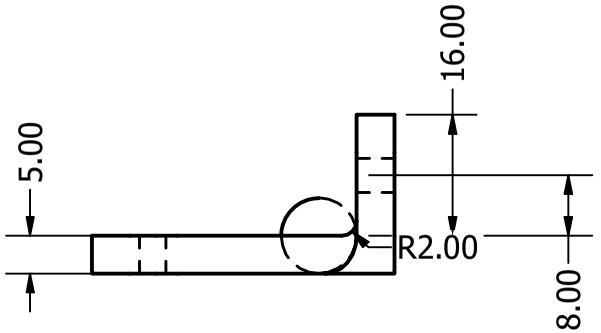
DRAWN DSOB	03-06-2023	TITLE Flask Holder		
CHECKED				
QA				
MFG				
APPROVED				
		SIZE A4	DWG NO Flask_holder_receiver	REV
		SCALE 1 : 1		SHEET 1 OF 1

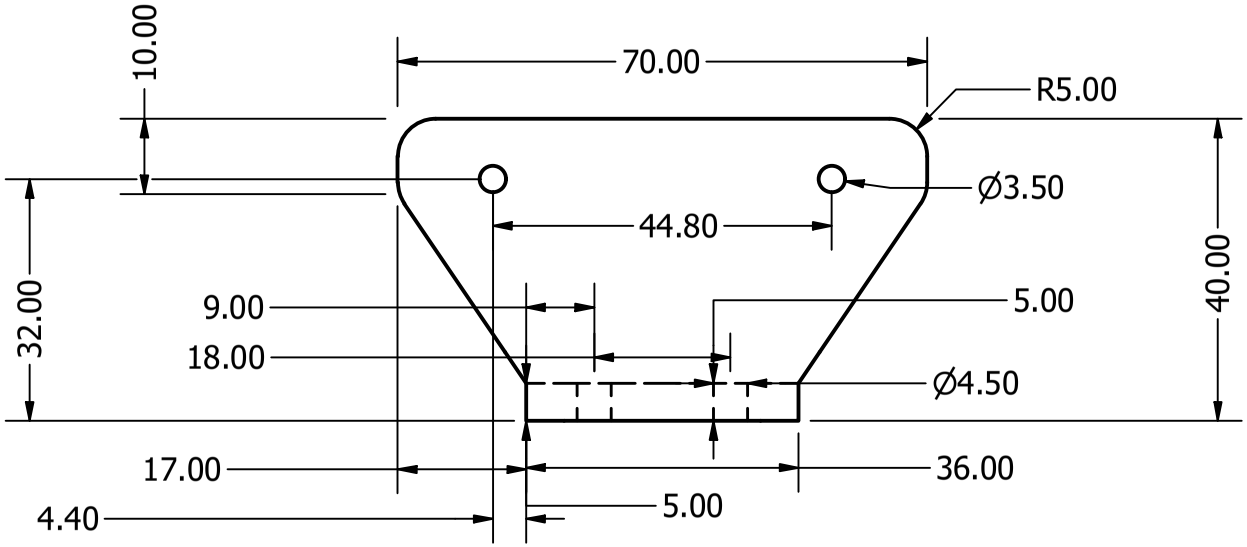


DRAWN DSOB	03-06-2023	
CHECKED		TITLE: Flask Holder Horizontal
QA		
MFG		
APPROVED		
		SIZE A4 | DWG NO T-175_holder | REV
		SCALE 1 / 2 | SHEET 1 OF 1

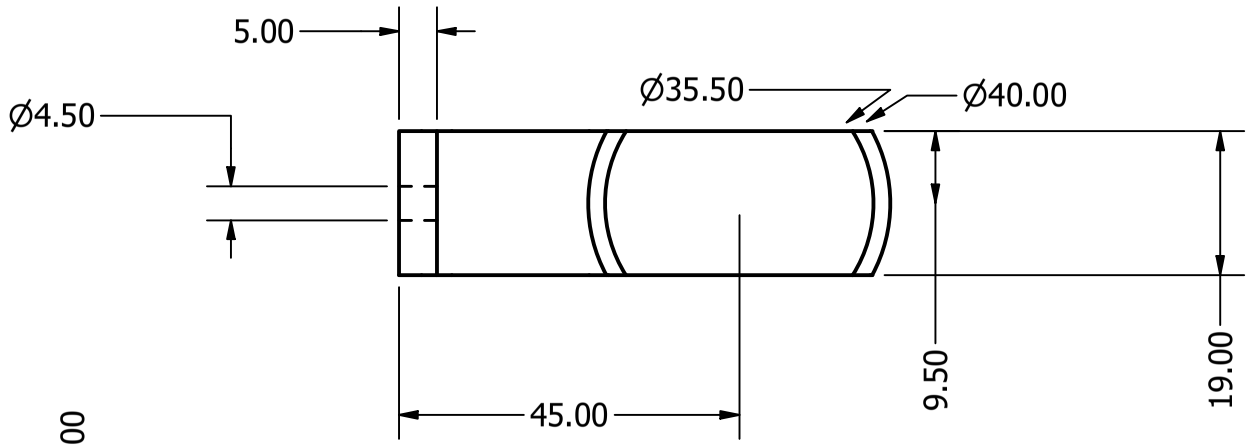


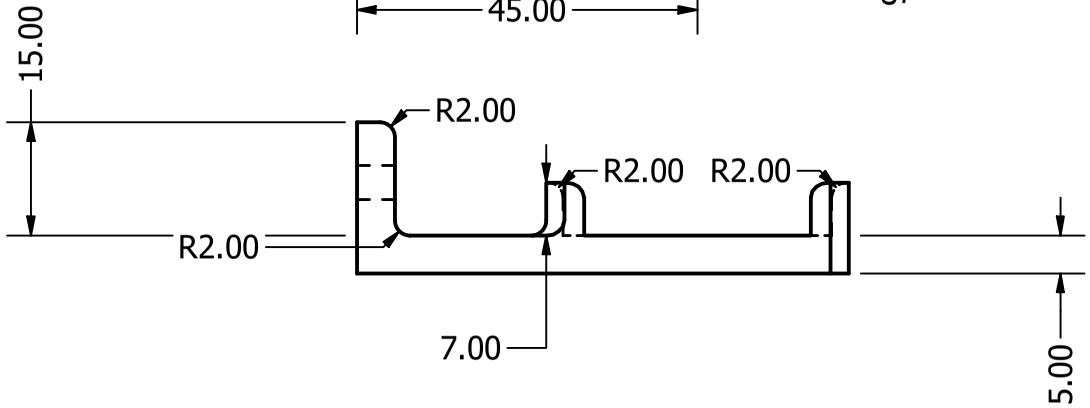


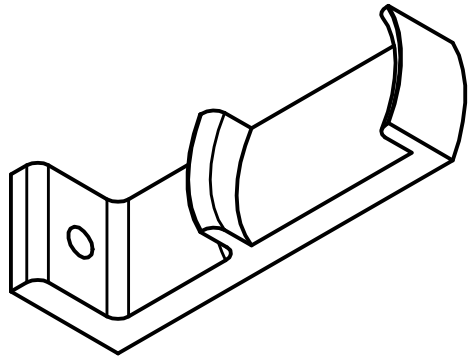




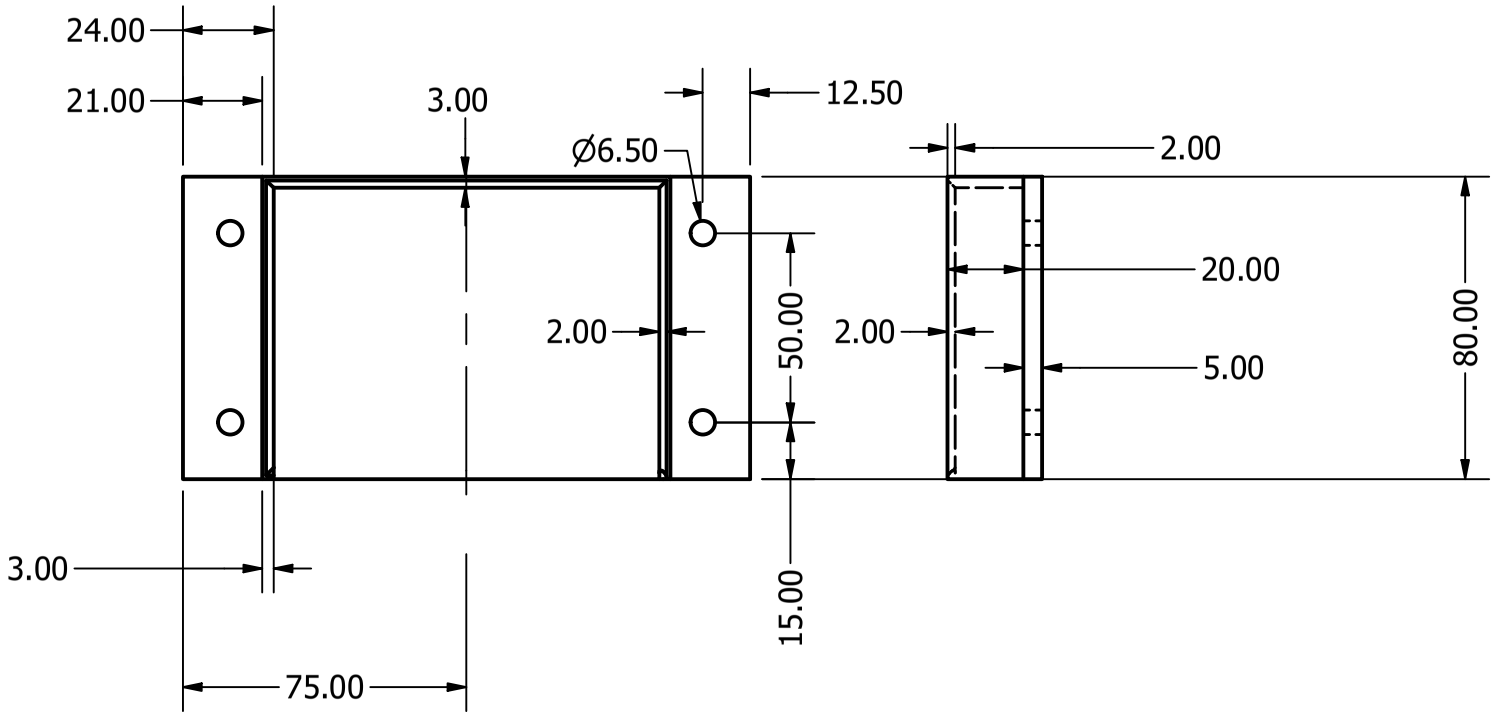
DRAWN DSOB	03-06-2023	TITLE: Camera Mount
CHECKED		
QA		
MFG		
APPROVED		
		SIZE A4 | DWG NO CameraHolder | REV
		SCALE 1 : 1 | SHEET 1 OF 1

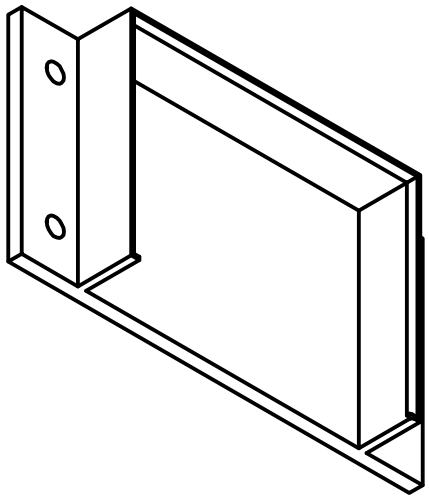






DRAWN DSOB	03-06-2023	TITLE Lid Holder		
CHECKED				
QA				
MFG				
APPROVED				
		SIZE A4	DWG NO Lid_holder	REV
		SCALE 1 : 1		SHEET 1 OF 1





DRAWN DSOB	29-06-2023	
CHECKED		TITLE: Medium Holder
QA		
MFG		
APPROVED		
		SIZE A4 | DWG NO MediumHolder | REV
		SCALE 1 / 2 | SHEET 1 OF 1

B.5 Additional Figures

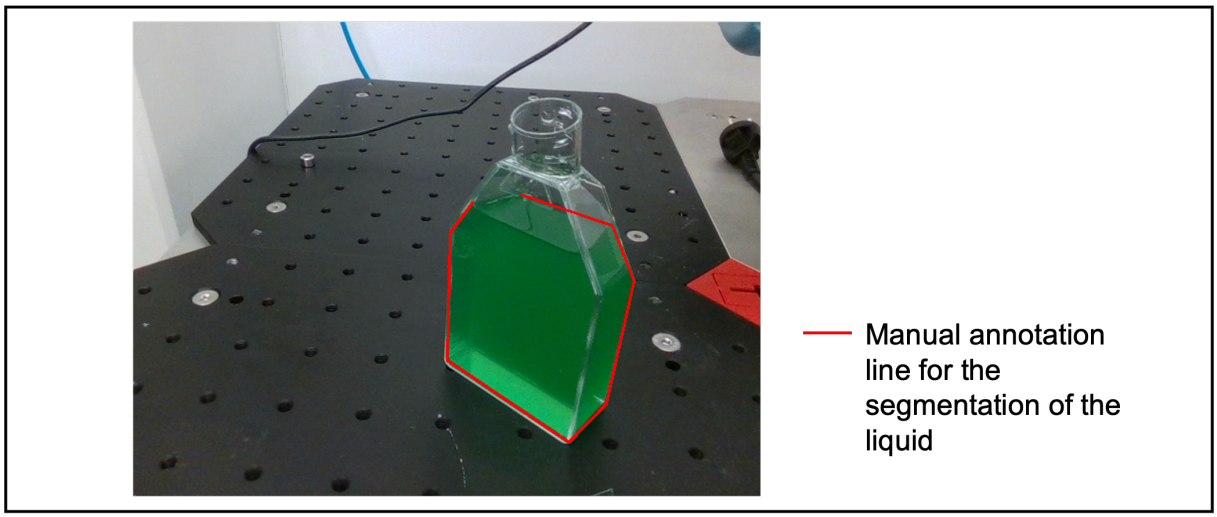


Figure B.21: Visualization of the process of manual annotation for the segmentation of liquids and vessels.

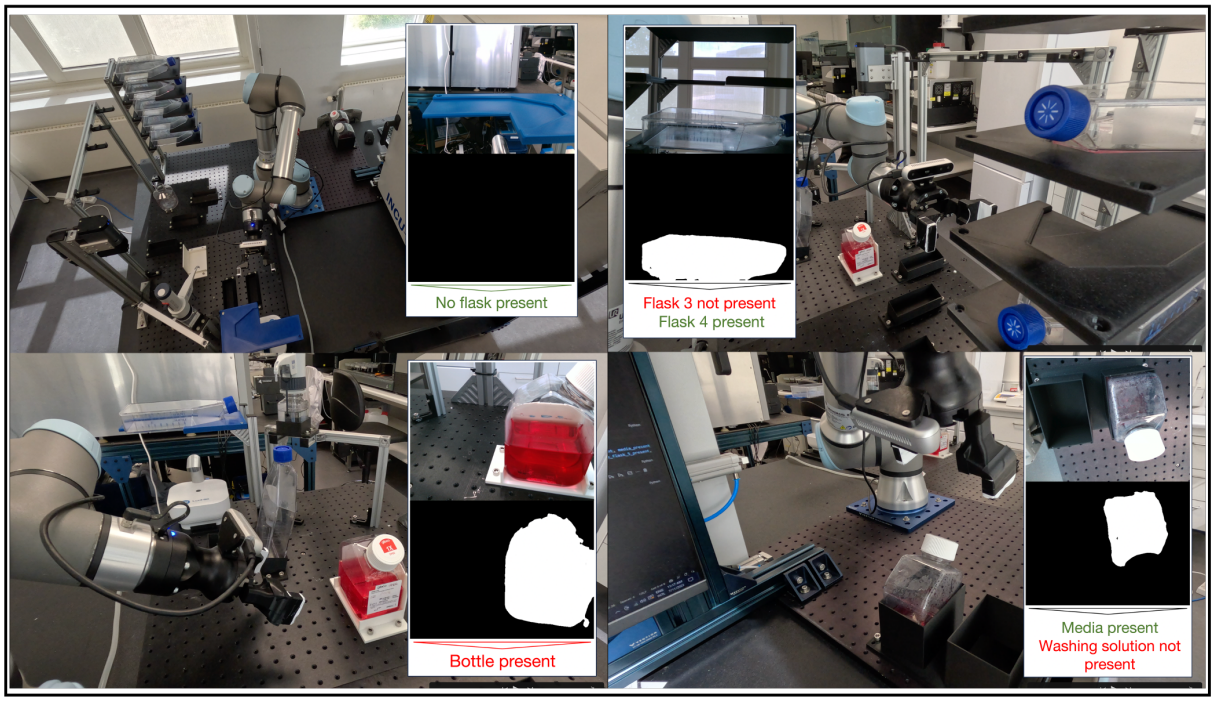


Figure B.22: Snapshots of the process monitoring using the segmentation maps predicted from the segmentation and depth estimation model.

Technical
University of
Denmark

Ørsteds Pl. 348
2800 Kgs. Lyngby
Tlf. 4525 1700

www.dtu.dk