BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

ÚSTAV AUTOMATIZACE A INFORMATIKY

DESIGN AND IMPLEMENTATION OF THE ROBOTIC PLATFORM FOR AN EXPERIMENTAL LABORATORY TASK

NÁVRH A IMPLEMENTACE ROBOTICKÉ PLATFORMY PRO EXPERIMENTÁLNÍ LABORATORNÍ ÚLOHU

MASTER'S THESIS

DIPLOMOVÁ PRÁCE

AUTHOR Bc. Martin Juříček

AUTOR PRÁCE

SUPERVISOR Ing. Roman Parák

VEDOUCÍ PRÁCE

BRNO 2022

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF MECHANICAL ENGINEERING

Assignment Master's Thesis

Institut:	Institute of Automation and Computer Science
Student:	**Bc. Martin Juříček**
Degree programm:	Applied Computer Science and Control
Branch:	no specialisation
Supervisor:	**Ing. Roman Parák**
Academic year:	2021/22

As provided for by the Act No. 111/98 Coll. on higher education institutions and the BUT Study and Examination Regulations, the director of the Institute hereby assigns the following topic of Master's Thesis:

Design and implementation of the robotic platform for an experimental laboratory task

Brief Description:

The thesis will focus on an experimental laboratory task in the field of medicine using a collaborative robotic arm and a camera system. The thesis will include a research in the field of collaborative robotics and a more detailed description of the Universal Robots UR3 robot. The theoretical part of the thesis will also include research in the field of machine vision and analysis of the use of robots in medicine. The main goal of the thesis will be the design of an advanced control system for a selected experimental laboratory task using the robotic system ROS (Robotic Operating System). The conclusion of the thesis will be devoted to the implementation and verification of the proposed solution.
The work will be carried out in the Research and Development Laboratory of Cybernetics and Robotics at the Institute of Automation and Computer Science, FME, Brno University of Technology, and will expand existing research in the use of advanced robotics in medical applications.

Master's Thesis goals:

– Explore collaborative robots from Universal Robots and describe the selected cooperating robot UR3 in more detail.
– Examine the machine vision area.
– Analysis of the use of robots in medicine
– Design a modular end–effector.
– Design a control system for a selected experimental laboratory task in the field of medicine.
– Implement a control algorithm.
– Verify the functionality of the created solution using simulation and on a real robot.

Recommended bibliography:

SICILIANO, B., KHATIB, O. Ed. Springer Handbook of Robotics. Berlin, Heidelberg: Springer-Verlag, 2016. ISBN 978-3-319-32550-7.

LYNCH, KM, PARK, FC. Modern Robotics: Mechanics, Planning, and Control. Cambridge Univeristy Press, 2017. ISBN 978-1107156302.

QUIGLEY, M., CONLEY, K., GERKEY, B. P., FAUST, J., FOOTE, T., LEIBS, J., WHEELER, R., Ng, A. Y. ROS: an open-source Robot Operating System. In ICRA Workshop on Open Source Software. 2009.

BRADSKI, G. The OpenCV Library. Dr. Dobb's Journal of Software Tools. 2000.

Deadline for submission Master's Thesis is given by the Schedule of the Academic year 2021/22

In Brno,

L. S.

doc. Ing. Radomil Matoušek, Ph.D. Director of the Institute	doc. Ing. Jaroslav Katolický, Ph.D. FME dean

ABSTRACT

Advanced robotics does not always have to be associated with Industry 4.0, but can also be applied, for example, in the Smart Hospital concept. Developments in this field have been driven by the coronavirus disease (COVID-19), and any improvement in the work of medical staff is welcome. In this thesis, an experimental robotic platform was designed and implemented whose main function is the swabbing samples of the nasal vestibule. The robotic platform represents a complete integration of software and hardware, where the operator has access to a web-based application and can control a number of functions. The increased safety and collaborative approach cannot be overlooked. The result of this work is a functional prototype of the robotic platform that can be further extended, for example, by using alternative technologies, extending patient safety, or clinical testing and studies.

ABSTRAKT

Pokročilá robotika se nemusí vždy pouze pojit s Průmyslem 4.0, nýbrž nachází své uplatnění i kupříkladu v konceptu Smart Hospital. Pokrok v této oblasti umocnilo onemocnění koronaviru (COVID-19), přičemž každé ulehčení práce zdravotnímu personálu je vítáno. V rámci této diplomové práce byla navrhnuta a implementována experimentální robotická platforma s hlavní funkcí stěru vzorků z předsíně dutiny nosní. Robotická platforma představuje kompletní integraci softwaru a hardwaru, kde má operátor přístup k webově založené aplikaci a může ovládat řadu funkcí. Opomenout nelze také zvýšenou bezpečnost a kolaborativní přístup. Výsledkem práce je funkční prototyp robotické platformy, který je možné dále rozšiřovat například v podobě použití alternativních technologií, rozšíření bezpečnosti či klinického testování a studie.

KEYWORDS

Robotic platform, smart hospital, nasal vestibule swab, advanced robotics, collaborative robotics, machine vision, convolution neural networks, point cloud, Universal Robots UR3, Intel RealSense D435i, ROS, Flask, Robo Medicinae I

KLÍČOVÁ SLOVA

Robotická platforma, smart hospital, stěr z nosní předsíně, pokročilá robotika, kolaborativní robotika, strojové vidění, konvoluční neuronové sítě, mračno bodů, Universal Robots UR3, Intel RealSense D435i, ROS, Flask, Robo Medicinae I

ROZŠÍŘENÝ ABSTRAKT

Od roku 2019 sužuje celosvětově celou populaci onemocnění COVID-19. Tato diplomová práce vznikla jako bezprostřední reakce na světové dění. Hlavní myšlenkou práce je návrh a implementace experimentální robotické platformy s hlavní funkcí stěru vzorků z předsíně nosní dutiny. V této oblasti sbírání vzorků pro diagnostiku onemocnění, se zabíralo již několik vědecko-technických týmů. Většinou tyto týmy postavily svá řešení na sběru vzorků z nosohltanu, buď dutinou nosní či ústní. V rámci sekce kybernetiky a robotiky, Ústavu automatizace a informatiky se jedná o rozšiřující práci, která nepřímo navazuje na aplikaci Open Tube. Open Tube představuje ve zkratce robotické pracoviště, které pomáhá laborantům zpracovat infekční vzorky.

Hlavním cílem práce je návrh funkčního prototypu experimentální robotické platformy. K řešení pak byly použity základní technologie jako je Python Flask a Robotický Operační Systém. Díky implikaci těchto nástrojů bylo možné vytvořit takzvaně webově založenou aplikaci. Flask a Robotický Operační Systém této robotické platformy běží na centrální jednotce, která může představovat kupříkladu stolní počítač či výkonný jednodeskový počítač. Operátor pak má přístup ke rozhraní člověk-stroj pomocí webového prohlížeče ze zařízení, které může například reprezentovat dotykový monitor. Důležitým faktorem vyvstává to, že zařízení musí být připojeno do stejné lokální sítě jako je centrální jednotka.

Pozornost je třeba dát na propojení softwaru a hardwarového vybavení. Zařízení na kterém stojí celá aplikace je kolaborativní robot UR3 od firmy Universal Robots. K robotu pak byl navrhnut vlastní uchopovač, který se sekládá z Onrobot HEX-E senzoru, 3D kamery Intel RealSense D435i a dvouprstého chapadla Onrobot RG2. Toto vybavení je pak řízeno pomocí již zmiňovaného Flask a Robotického Operačního Systému. Integraci těchto části má pak operátor možnost ovládat ze svého zařízení, přičemž se mu naskýtá širokého pole funkcí. V tomto duchu byly implikovány funkce jako je ovládání jednotlivých kloubů robota s vizualizací digitálního dvojčete, informativní panel pro zobrazení aktuální teploty a proudu v kloubech, přístup k databázi pacientů a mnoho dalších samotných či dílčích funkcí.

Klíčovou funkcí je pak polo-automatizovaný proces sběru vzorků z předsíně nosní dutiny. Tento proces lze zcela automatizovat, avšak za cenu snížení bezpečnosti. Celý proces začíná uchopení tyčinky chapadlem, která je určená pro sběr vzorků, poté je proveden pohyb do inicializační pozice. Následuje kooperace operátora, kde je provedeno nalezení daného pacienta v databázi (například načtení QR kódu). Pokračuje funkce rozpoznávání obličeje za účelem ověření, následná validace pacientovy hlavy ve snímaném rozsahu kamery, detekce a výpočet středu nosních dírek a rekonstrukce pacientovy hlavy ve 3D jako mračno bodů. Výsledek této části procesu je pak použit pro výpočet trajektorie pohybu robotického ramene do

nosní předsíně s následným provedením krouživého pohybu. Na závěr je vykonán pohyb zpět do inicializačních pozic a odevzdání tyčinky. Při procesu je důležité dbát na bezpečnost, kterou především zajišťuje silovo-momentový senzor HEX-E. Ten zabezpečuje, že pokud bude dosažena maximální hodnota povolené síly působící na pacienta robot je bezprecedentně zastaven. Svou velkou výhodu pak naskýtá, že zastavení nezpůsobí vypnutí kontroléru robota či Robotického Operačního Systému. V tomto ohledu se lze také spolehnout na bezpečnostní prvky kolaborativního robota samotného, kdy při pacientovi nezpůsobí bolest či zranění.

Pod lupou detekce nosních dírek je pak vymezena oblast segmentace obrazu, přesněji využití dvou segmentačních modelů konvolučních neuronových sítích. V práci byl použit model U-Net a ASPOCRNet, každý model byl trénován se třemi páteřními sítěmi, kde jedna z nich reprezentovala "state-of-the-art". Výstupem učení je fakt, že segmentační model U-Net si vedl oproti ASPOCRNet patrně lépe.

Práce pojednává průřezem o kolaborativní robotice, strojovém vnímání především zaměřeno na strojové vidění, aplikací robotů ve zdravotnictví, dále je v této diplomové práci vyzdvihnut koncept Smart Hospital, analýza využití robotů ve zdravotnictví, návrh, simulace a testování experimentální robotické platformy. V části strojového vidění je přiblížena problematika segmentace obrazu s využitím umělých neuronových sítí. Praktický výstup také zahrnuje i například funkce základního testování Flask aplikace, či jednoduchý útok silou na přihlašovací systém aplikace. Závěrem praktické části je pak vymezena oblast simulace a následného otestování v laboratoři. Výsledkem otestování je fakt, že byl zkonstruován funkční protyp experimentální robotické platformy, avšak s otevřnými možnostmi na další rozšiřování. Díky samotné komplexnosti celé práce je pak možné provést mnoho dalších úprav vylepšení. Jedním z vylepšení je kupříkladu využití většího kolaborativního robota, či případně dvou-ramenného kolaborativního robota jako je ABB YuMi. V softwarové části je pak možné provést modifikace například za účelem zvětšení bezpečnosti, implementace vlastních plánovacích algoritmů či čisté modifikace s cílem nasazení do provozu. Celá práce je postavena na modularitě, rozšiřitelnosti a upravitelnosti.

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

2022

BIBLIOGRAPHIC CITATION

JUŘÍČEK, Martin. *Design and implementation of the robotic platform for an experimental laboratory task.* Brno: Brno University of Technology, Faculty of Mechanical Engineering, Institute of Automation and Computer Science, 2022, 99 p. Master's Thesis. Supervised by Ing. Roman Parák

AUTHOR'S DECLARATION

I declare that this thesis is my original work, that I have written it independently under the supervision of my thesis supervisor, and that I have used professional literature and other sources of information, all of which are cited in the thesis and listed in the reference list.

As the author of this work, I further declare that in connection with the creation of this work I have not infringed the copyrights of third parties, in particular, I have not infringed in an unauthorised manner the personality rights of others and I am fully aware of the consequences of the violation of the provisions of Articles § 1 of the Copyright Act No. 121/2000 Coll. including possible criminal consequences.

In Brno 20. 5. 2022 ....

Martin Juříček

ACKNOWLEDGEMENT

I would like to thank the supervisor of the master's thesis Ing. Roman Parák. For professional consultation, friendly cooperation, helpful assistance, and above all, guidance. I would also like to thank all my colleagues and friends who helped me with my thesis. Last but not least, I would like to thank my beloved one and family, who have supported me throughout my studies.

CONTENTS

Chapter 1

Introduction

The influx of COVID-19 has caused mortality and often painful diseases around the world. Intensive research and development to combat this disease is extending to all possible areas, including robotics. In this struggle, robotics plays the role of assistant to the medical staff. Therefore, this thesis also focuses on an experimental laboratory task in the medical field using a collaborative robotic arm and camera system. The aim is to create an advanced robotic platform with the main objective of collecting samples from the nasal vestibule. To solve this task, a UR3 collaborative robot from Universal Robots was used with a customized gripper design consisting of an OnRobot RG2 gripper and a HEX-E sensor, and last but not least, an Intel RealSense D435i 3D camera. The entire solution is then integrated into the software infrastructure, which is mainly powered by Python Flask and the Robotic Operating System. Artificial intelligence, more specifically convolutional neural networks, was also used in this thesis. Primarily to detect the centre of the nostrils.

The theoretical part begins with an overview of the trend, and future of collaborative robotics, introducing the reader to several collaborative robots from the stable of leading robot manufacturers, including the UR3 robot. The reader then delves into the field of machine perception, focusing primarily on machine vision, and the conclusion of this research is devoted to image segmentation using artificial neural networks. This is followed by a chapter on the field of robots in medicine, describing interesting applications and analyses of medical applications. The practical section offers a look at a complete robotic platform solution, from gripper design to software solutions. The different application functions and their roles in the system are mentioned. In particular, it is about training convolutional neural networks to detect nostrils. Finally, the practical part describes the simulation of the process, the application tests, and the subsequent real-world tests.

1.1 Motivation

The outbreak of the global COVID-19 pandemic has highlighted that one of the most crucial sectors of human society is healthcare. In hospitals, clinics, and surgeries, doctors and nurses were exposed to almost direct contact with potentially infected patients. Due to the infection of this personnel, many entire departments have been forced to suspend scheduled examinations and possible operations for a while. Therefore, in these difficult times and with the prospect of another pandemic, this thesis aims to innovate and provide a solution to collecting samples from the nasal vestibule. This thesis has been developed at the Research and Development Laboratory of Cybernetics and Robotics at the Institute of Automation and Informatics, Faculty of Mechanical Engineering, Brno University of Technology, to extend the existing research on the use of advanced robotics in medical applications.

Within this laboratory, a project has already been created for the University Hospital Brno, which is called Open Tube [20]. The application aims to solve automated sample testing on the COVID-19. This robotic workstation, equipped with an ABB collaborative robot, an automatic tube opener SMC, and other instruments, creates an almost fully automated process. The project has applications in biohazard enclosures and toxic cytostatic laboratories.

1.2 Related Work

Around the world, several research teams have tried to contribute their inventions to the fight against COVID-19 disease, including those related to oropharyngeal swab sampling. A team from the University of Southern Denmark and the company Lifeline Robotics have devised the first automated robotic solution for COVID-19 sampling [12]. This was a solution using a robot from Universal Robots, while their Gripper integrated a swab holder and a camera. The head of the patient is then placed in the bracket for patients. The bracket prevents deflecting and endangering the patient.

The team that also tackled the problem of oropharyngeal swab sampling for COVID-19 diagnosis was from the Chinese University of Hong Kong [22]. Their solution is also based on a collaborative robot from Universal Robots company, whose main invention is a complex gripper that grips a rigid-flexible coupled manipulator. The complete system thus consists of a collaborative robot, a linear manipulator with a servo motor for rotation, which holds a micro pneumatic actuator. Sensors such as the 3D camera, the endoscope, and the force detection system are also mounted on the gripper. A ring illuminator for front illumination on the camera axis is also an integral part. As for the software, the authors of this article addressed the detection of the oral cavity phantom. For this detection, they used one of the artificial intelligence methods, Mask Regions with Convolutional Neural Network Features. The result of the work is a robotic application that has already been validated by several volunteers and has areas that can be improved and subsequently tested clinically.

Chapter 2

Collaborative Robotics

Over time, across development and industrial revolutions, progress has defined human society. Today's era is characterized by smart devices, robotics, and artificial intelligence, which have an impact on everyday life. Robotics has a growing number of applications and is becoming more accessible, thanks to mobile and collaborative robotics.

Collaborative robotics is a field of study that is defined as cooperation between humans and robots. The main task of a robot is to assist the human in his work. Collaborative robots and operators usually work in a shared workspace. This is a major advantage over conventional robotic arms, which are separated by strict isolation in the form of mechanical fencing or sensor barriers. The tasks performed by collaborative robots typically include arduous routine tasks, difficult operations, or processes that cannot be fully automated.

Fig. 1: Collaborative robotics [47]

2.1 Trend and Future

KUKA, one of the best-known companies in the field of industrial robotics, launched its first collaborative robot in 2004. This launch can be considered a milestone in the history of collaborative robots, and the term cobot can be linked to it. The word cobot was created by combining the words collaborative and robot. Collaborative robotics has become especially popular in small and medium-sized enterprises, where robot performance, accuracy, and repeatability must be combined with human expertise. Among the largest and most well-known manufacturers of collaborative robots are Universal Robots, FANUC, KUKA, and ABB. These robots have common features such as safety, affordability, flexibility, and ease of use.

Affordability depends on the company and provider, but generally, the purchase and maintenance prices are lower than for conventional industrial robots. In the case of conventional industrial robots, a safe zone has to be created, whereas in the case of collaborative robots it is already guaranteed by a series of safety elements and sensors. Another key feature is flexibility due to low weight and ease of assembly. This means that robots can be moved to other habitats or cells in a short time. Last but not least, these robots are generally easy to use, thanks to teach pendant devices[1]. These devices provide operators with limited, but easy access to control and programming cobots.

Safety is the highest priority for collaborative robots, which is why it has many different safety features and sensors. Starting with the installation of soft materials on the joints and parts of the robots themselves, to the use of cameras and lasers. Some safety features are used to slow the cobot down and others to stop it immediately. For manufacturers and developers to be able to mark their robots as collaborative, they must follow ISO 10218-1 and the ISO/TS 15066 specifications. [7]

The future of collaborative robotics will be defined by cobots approaching the performance of conventional industrial robots, but retaining their safe collaborative approach, as well as efforts to make cobots more design and user-friendly. Collaborative robotics will find a new field of application as an assistant in laboratories, hospitals, shops, warehouses, etc.

2.2 Collaborative Robots

With the advent of Industry 4.0, several new concepts have emerged that are changing industrial values. Cobots are being innovated by big companies like ABB or Stäubli, but also start-ups with new ideas such as Slovak start-up Spinbotics.

2.2.1 ABB GoFa, Swifti and Yumi

One of the latest cobots with a modern collaborative robotics approach is the GoFa robot from the Swedish-Swiss company ABB. GoFa's name stands for "Go far. Go Faster. Go further". This 6-axis robot with a reach of 950 mm, not only has a nice modern design, but also a 5 kg payload [2]. The GoFa can move at speeds of up to 2.2 $\mathrm{m \cdot s^{-1}}$, but its weight of 27 kg is higher compared to other cobots. Swifti (Superfast) robot, equally modern, doesn't have such a nice design as GoFa, but ABB put more emphasis on the industrial parameter of speed. This 6-axis has

[1] The Teach Pendant is a device provided by collaborative robot manufacturers to easily control and manage robots. It is usually a tablet or a small computer with control peripherals.

a 4 kg payload and is 5 kg lighter than the GoFa, at the cost of a lower reach range [4]. GoFa cobot has 0.04 mm less position repeatability compared to Swifti. ABB makes Swifti in two versions with a 475 mm reach range and a maximum speed up to 4.32 $\text{m} \cdot \text{s}^{-1}$ or 580 mm reach range and a maximum speed up to 5.05 $\text{m} \cdot \text{s}^{-1}$. In terms of safety, GoFa has an integrated torque sensor in each of its six joints offering superior power and force limiting performance. On the other hand, Swifti safety is more complex because combining ABB's SafeMove and safety laser scanner. This safety system allows the cobot to use its maximum speed if the operator is outside the working area, if the operator approaches the working area the cobot automatically slows down or halts. This technology allows Swifti to work quickly and accurately with pose repeatability ±0.01 mm, which is about ±0.04 mm more accurate than GoFa.

In 2015, ABB introduced the world's first two-arm collaborative robot YuMi, at the Hannover Messe. YuMi (You and Me) has two arms, which each have a 7-axis of freedom [5]. Compared to the new generation, this robot can only be mounted on a table. One of the main disadvantages is the payload, which is only 0.5 kg. The speed of the robot itself reaches a maximum of 1.5 $\text{m} \cdot \text{s}^{-1}$, but pose repeatability is ±0.02 mm. ABB Yumi weighs 38 kg and has a reach of 559 mm, thanks to its two arms it is capable of multi-tasking.

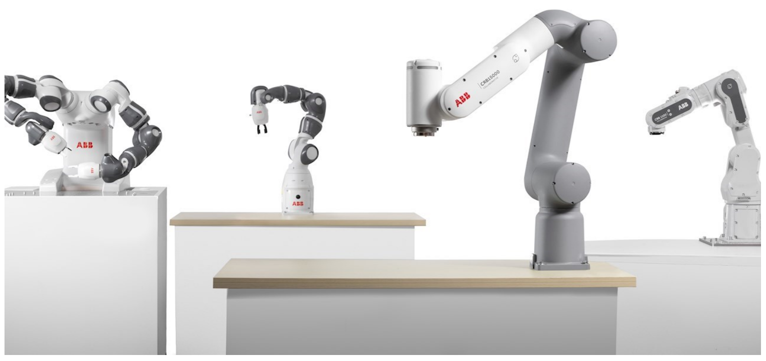


Fig. 2: ABB - Yumi (left), IRB 14050 (back), GoFa (front), Swifti (right) [3]

2.2.2 KUKA LBR Med

One of the leading companies in the field of robotics is KUKA, which provides a medical cobot called LBR Med [32]. This is a 7-axis collaborative robot, which the company distributes in two variants LBR Med 7 R800 and LBR Med 14 R820. The range of the LBR Med 7 R800 type is 800 mm with a load capacity of 7 kg and a self-weight of 25.5 kg, and its position repeatability is ±0.1 mm. Whereas the type with the designation LBR Med 14 R820 has a range of 820 mm with a load capacity of 14 kg and a weight of 32.3 kg. The position repeatability thus differs from the first type by ±0.05 mm. The LBR Med collaborative robot is CB certified following IEC 60601-1 and IEC 62304, while KUKA guarantees many safety features such as configurable safety movements, single fault, or circuit safety. To achieve maximum safety, the cobot integrates force and torque sensors, as well as other supporting sensors to avoid collisions.

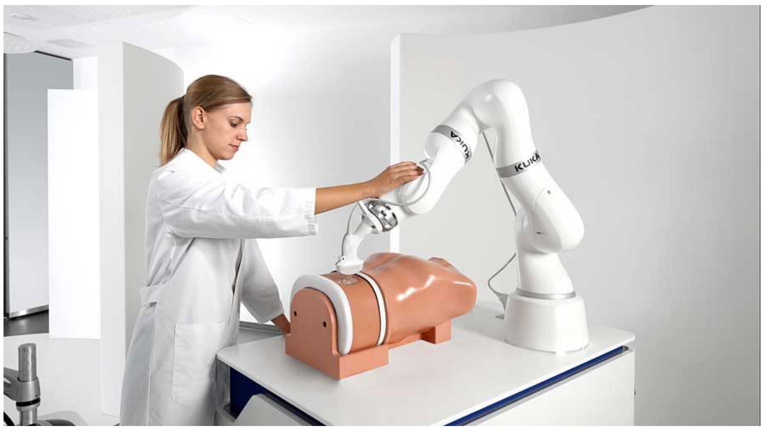


Fig. 3: KUKA LBR Med [38]

2.2.3 Universal Robots UR3

A major influence in the field of collaborative robotics is the Danish company Universal Robots. The company based in city Odens was founded in 2005, with Esben Østergaard, Kasper Støy and Kristian Kassow. At the turn of the 20th and 21st centuries, the market was dominated by mainly large, heavy, and expensive robotic arms that only the larger companies could afford. This led to the idea of making robotic arms available to smaller companies. The first cobot was therefore launched on the Danish and German markets in 2008. The company's first cobot is the UR5 [61]. Universal Robots has already released two generations of cobots, the latest is called the e-series and the previous CB series. The new generation consists for example of collaborative robots UR3e, UR5e, UR10e, and UR16e.

The ultra-lightweight collaborative robot UR3 from the CB series of the Universal Robots was released in 2015. This six-axis cobot weighs only 11 kg and has a reach of up to 500 mm, but can rotate 360° at all joints [59]. This feature is one of the main advantages over competing for collaborative robots. The payload of the cobot is 3 kg with a repeatability position accuracy of ±0.1 mm. The cobot can operate in the range of 0–50°C using a speed up to 1 m $\cdot$ s^{-1}. Users can mount the robot either horizontally on the ceiling or floor, as well as vertically on a wall. The level of protection of the UR3 is 64, so according to the standard, protection against dangerous contact with any device and splashing water from all angles.

Safe collaborative operation is guaranteed by TÜV NORD certification. Rated and tested by UR3 according to EN ISO 13849:2008 PL d. The cobot can be connected via two digital inputs, two digital outputs, and two analogue inputs. To be able to operate and program the robot, a controller and a control panel in the form of a 12" touchscreen display where the UR Polyscope Human-Machine Interface (HMI) is installed. Communication is possible via industrial buses Modbus TCP, Profinet, and EthernetIP or communication over a computer network via the TCP/IP protocol.

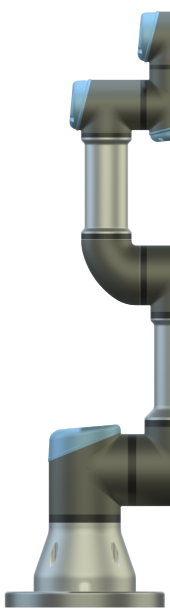


Fig. 4: Universal Robots UR3 CB-Series

Chapter 3

Machine Perception

The tools of Industry 4.0 and smart machines also include the use of artificial intelligence, for example in robotics, automotive, or in the processing of large amounts of data from logistics, manufacturing, or distribution systems. The essence of artificial intelligence is to mimic the human mind in learning or understanding when solving complex problems through computing. Artificial intelligence is a complex branch in the field of computer science, that brings various scientific disciplines such as natural language processing, machine learning, and even machine perception. These areas are gradually developing and becoming intertwined.

One of the most progressive and researched areas is machine perception. Machine perception is concerned with extracting information about a scene by analyzing signals from sensors. It is similar to the way a person perceives the world around him with his senses [58]. Machine perception includes processing data from all kinds of sensors such as cameras, ultrasonic sensors, or even microphones. This area of artificial intelligence is therefore divided into several sub-areas, such as machine vision, machine hearing, and others.

Research in machine perception focuses on a wide range of applications especially such as image, video, and sound understanding. In robotics, it can also be a perception of force or torque and many others. In recent years, this research and application have been accelerated mainly by more powerful hardware. The resulting applications can be seen in everyday life, for example when using smartphones. One of the most interesting applications may be solving an equation from the image, where the smartphone camera captures the image, the extraction of the elements into digital form is performed and then the problem is calculated.

The more specific demonstration may be a robot that uses a camera as vision and touch perception is represented by a force sensor. This can be used, for example, when dignifying the mechanical fixation of manufactured parts. The control unit receives the sensor data and processes it so that the image data detect the obstacles and the workpiece itself. The force sensor signals are processed so that the control unit performs a diagnosis.

One area that is still in its beginnings is that of machine olfaction. In this area, scientific and technical teams are trying to detect and measure odours using machines. This may have enormous potential for example in the form of detecting various chemicals in the air, etc.

3.1 Machine Vision

In the last few years, one of the most popular fields of machine perception is machine vision. Machine vision mimics human vision by processing data in the form of a digital image, which is produced by one or more image sensors. This field is used in a huge number of deployments, such as advanced robotics, autonomous driving, and industrial or medical applications. An example might be quality inspection in factories, car traffic sign detection from images, or diagnosis from X-ray scans.

Many other disciplines can be included in machine vision, such as the design of camera hardware and software, the use of filters, lenses, extension tubes, or the appropriate illumination of the scene. Other areas are image acquisition, correction of noise and unwanted effects, image processing, and rendering of the final image. For example, the latest generation of smartphones has all these functions integrated to maximize image production and user experience.

Before algorithms or artificial intelligence methods can be applied to image processing, sensor data must be acquired. Data is acquired by collection from image sensors which are often cameras. The basic physical principle of the cameras is based on incident photons that eject electrons out of the valence layer in semiconductors, which are then trapped in a potential pit (electron trap). 3D cameras are used less often. 3D cameras augmented classical 2D sense with additional technology that simulates human binocular vision, and therefore captures three-dimensional images in point clouds form. Augmentation technologies in 3D cameras can be in many forms, from IR sensors to software 3D reconstruction from 2D images. For proper machine vision system design, it is also necessary to address lighting and the appropriate selection of hardware and software.

After the data is acquired, first, the image pre-processing phase is carried out. At this phase, data correction, such as noise, salt, and pepper, or gamma correction, is performed, sometimes followed by data augmentation in the sense of image or photometric modification, such as horizontal flipping, and image cropping, or brightness and contrast adjustment. Secondly, image processing continues, in the last years, often characterized by convolutional neural networks. Due to the use of the convolutional neural network, higher computational power requirements are imposed. However, some methods use other mathematical models and don't require high computing complexity. The final stage of processing generally provides a solution to a specific problem and takes follow-up action given the context and nature of the problem being solved. [54]

An illustrative example is the classification of objects from the input image. As shown in Fig. 5, the entire machine vision process depends on the context of the task at hand, how the desired scene is captured, and other criteria.

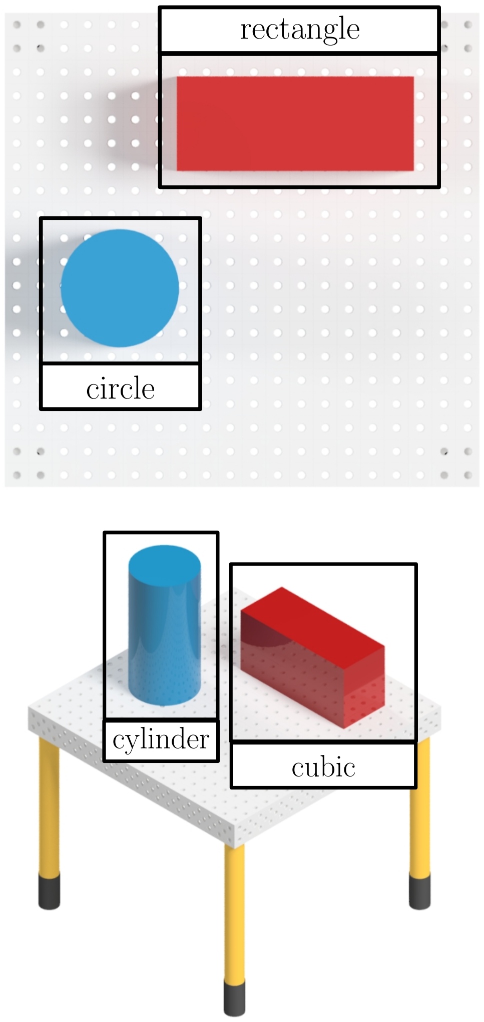


Fig. 5: Simple classification

3.1.1 Machine Vision in Robotics

The field, where machine vision finds its inherent representation, is robotics. Cameras become the eyes of robots and are used differently for different tasks (Fig. 6). In industry, robotic arms or SCARA robots are combined with machine vision mainly for bin picking tasks. Nowadays, bin picking still presents many challenges due to the nature of the material, location, visibility, or arrangement. Whereas mobile robotics uses machine vision for orientation in space, as well as for obstacle avoidance or object tracking.

An example of a company combining machine vision and robotics is the iRVision system. iRVision is a plug-and-play visual detection system from FANUC [16]. This system aims to enable quick installation, ease of use, and flexibility. It is fully integrated into the robot, so it does not require any additional hardware for configuration and operation, nor any interface to external devices. It uses 2D or 3D part recognition and can locate parts regardless of their size, shape, or position.

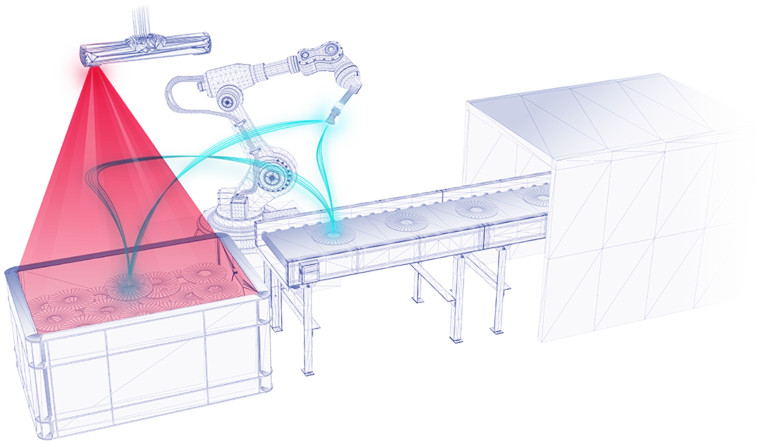


Fig. 6: Bin picking process [43]

3.2 3D Cameras

In recent years, there has been an increase in the research field of machine vision, and fast development in the sensors themselves, including 3D cameras. A 3D camera is a sensor that can perceive the depth of an image to replicate the three dimensions as perceived by a person.

A 3D camera is one possible concept that classically has two or more cameras integrated with additional sensors or lasers. In the context of 3D cameras, terms such as a colour image or depth image are also used, which are closely related to the types of cameras used in the whole concept of the 3D camera.

- colour image - RGB camera creates a 2D pixel grid in which each pixel is assigned a value, usually RGB (Red Green Blue). Each attribute has a number from 0 to 255, so for example, a black pixel can be denoted as (0,0,0).
- depth image - A depth camera has pixels that have a different numerical value associated with them, which indicates the distance of the camera, or so-called depth.

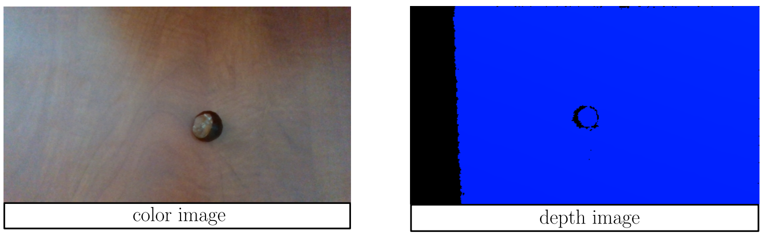


Fig. 7: Colour and depth image from Intel RealSense D435i

One of the most commonly used 3D camera methods is structured light. The principle of the structured light method is characterized by projecting coded light onto an object in the scene. The projected light has a specific pattern. Since the projected pattern is known, the depth information is provided by the way the camera sensor sees this pattern in the scene. Thus, the object in the scene is observed and decoded by the camera, and the decoding process is based on the comparison of paired points in two views and the triangulation of the rays emanating from the optical centres. For example, if the pattern is a series of stripes projected onto an object, the stripes will deform and bend around the surface of the object in a certain way. If the subject moves closer to the emitter, the pattern will change. Using the disparity between the expected image and the actual image displayed by the camera, the camera distance for each pixel can be calculated. [33, 25]

Another method used in 3D cameras is stereoscopic vision. This method uses depth from stereoscopy, a classical computer vision algorithm inspired by human binocular vision. The whole system is based on two parallel views-port and calculates depth by estimating disparities between matching key points in the left and right images [15]. 3D cameras based on this principle use stereo depth cameras that project infrared light onto the scene to increase the accuracy of the data [25]. As a result, these cameras can use any light to measure depth. For a stereo camera, all infrared noise is good noise. Stereo cameras for depth detection have two sensors separated by a short distance. The distance these cameras can measure is directly related to the distance between the two sensors: the wider the baseline, the further the camera can see. Stereo cameras work in the same way as when using two eyes, with our brain calculating the difference between each eye.

3D cameras based on LiDAR technology are a type of camera that uses laser light to calculate depth [25]. The measurement method is based on measuring the distance to the target by illuminating it with laser light and then measuring the reflected light utilizing a sensor. Depending on the intensity and wavelength of the light, time-of-flight sensors can measure depth over considerable distances, which is used, for example, for terrain mapping. The main disadvantage of cameras is that they can be sensitive to other cameras in the same area, while they can also perform worse in outdoor conditions. The sun, or any other light source hitting the sensor, can degrade the quality of the depth image.

3.2.1 Photoneo

Photoneo, a Slovak startup founded in 2013, is one of the most successful startups in the field of 3D cameras. This company has won several awards including the platinum-level VSD 2019 Innovators Award.

The flagship product is MotionCam-3D, which is the world's highest resolution and highest accuracy area 3D camera for dynamic scenes [44]. An integral part of this 3D camera is Parallel Structured Light technology which contains a clever sensor, that is designed to perform the acquisition in one snapshot as opposed to the sequential scanning of a standard image sensor. The camera with the largest scanning range is marked as MotionCam-3D L. This camera has a scanning range of 778—3034 mm, with a weight of 1.5 kg. The camera is capable of inspecting objects moving at speeds up to 40 $\mathrm{m \cdot s^{-1}}$, with a 3D throughput of 15 million points per second.

Photoeno also offers one of the best cameras on the market for resolving static scenes PhoXi Fig. 8 [45]. It uses similar technology to the MotionCam-3D series, focusing mainly on thermal stability, relatively lightweight, and energy efficiency. All under the umbrella of an IP factor of 65 and easy system integration. Compared to the MotionCam-3D L, the camera weighs only 0.95 kg, with a scanning range of 458—1118 mm. The camera itself has a relatively small size of $677 \times 68 \times 416$ mm, with a 3D throughput of 16 million points per second.

Fig. 8: Photoneo - Phoxi M (down), Phoxi XL (up)

3.2.2 Intel RealSense

One of the most flexible and widely used 3D cameras in the world is the RealSense series from Intel Corporation. Intel made world history with the introduction of the first microprocessor, the Intel 4004. This 4-bit microprocessor ushered in a new era of computers as we know them today. Intel RealSense technology is not just a hardware device, but a combination of hardware and software. Intel provides the ability to control 3D cameras using libraries for various development tools or programming languages such as Unreal Engine, Unity, C++, Matlab, and Python [28].

Intel RealSense serie offers the D415, D435/D435i, D455 and L515 cameras. The L515 is the flagship of the RealSense 3D camera family. This camera integrates LiDAR technology to capture depth images [27]. Thanks to this technology, it can capture depth images with a resolution of 1024×768 pixels and 30 FPS (frames per second). This camera is designed with a compact cylindrical shape with a diameter of 61 mm and a height of 26 mm. It is primarily designed for indoor use with the RGB sensor using a rolling shutter[1], while it can achieve a frame rate of 30 FPS and a resolution of 1920 × 1080 pixels. The camera is controlled and powered by the Vision ASIC processor. An important feature is that this camera provides an eye-safe Class 1 laser.

The most versatile camera in the RealSense series is the D435i. This camera combines the robust depth-sensing capabilities with the addition of an Inertial Measurement Unit (IMU)[2] [26]. Thanks to its small size of 90 × 25 × 25 mm, it has a large number of applications in robotic navigation or object recognition. The D435i features the Vision Processor D4, which controls RGB and depth cameras, and an infrared projector. The total maximum range of the D435i is around 3 m, while the RGB camera uses rolling shutter technology to achieve a maximum resolution of 1920 × 1080 pixels at a frame rate of 30 FPS. Stereoscopic depth camera uses a global shutter[3] and reaches a maximum resolution of 1280 × 720 pixels with a maximum of 90 FPS. Thanks to this design, it can be used both indoors and outdoors. The design comparison as shown in Fig. 9.

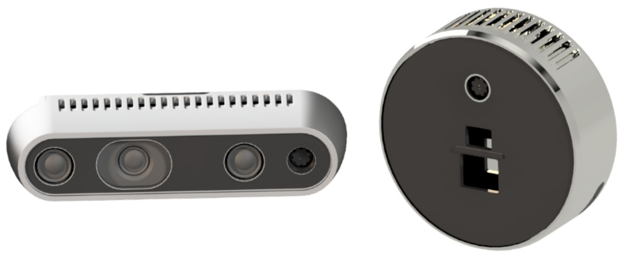


Fig. 9: Intel RealSense D435i and L515 CAD models

3.3 Image Segmentation

One of the integral tasks of machine vision is image segmentation (Fig. 10). The main goal is to divide the image into parts that are related to objects or areas of the real

[1] Rolling shutter is a method of capturing images or video in which a single frame or multiple frames do not capture the entire scene at a single moment, but quickly scans the scene in a vertical, horizontal, or rotational direction.

[2] IMU is a combination of several sensors with gyroscopes and is used to detect movements and rotations in 6 degrees of freedom [24].

[3] Global shutter is a method of capturing images video in which one or more frames record the entire scene at a single moment.

world or to separate objects from the background. This task can therefore be classified as a lower level of processing, however, segmentation is widely used in further processing for higher-level processing tasks such as 3D measurement, classification, object detection, or dimensional measurement. The process itself is the division of an image into multiple segments that meet certain criteria. For example, if the pixels in a segmented part have the same grey level or brightness, this can lead to ambiguity.

There are many different techniques with distinct approaches for solving image segmentation from a digital image. One of the oldest, but also the simplest is called thresholding. This technique can be used either on its own or as part of more sophisticated methods. It uses different levels of object and background intensity. The thresholding is then used to determine the pixel membership of either the background or the object.

The well-known method is edge-based segmentation. Edge-based segmentation takes advantage of the fact that there is a significant difference in surrounding pixels, which allows edge detectors to identify local edges. Edge detectors are algorithms whose output is a set of edges in the image. A related method is a region-based segmentation, which uses edges and then creates regions after contour closure.

Knowledge-based image segmentation is among the more advanced algorithms as they can make segmentation very easy because they use atlas models or templates of segmented objects that are generated automatically based on the training data. These advanced techniques can be also included in hybrid methods, these ones cannot be clearly classified but they are methods based on, for example, morphology, amplitude projection, or neural networks. [55, 54]

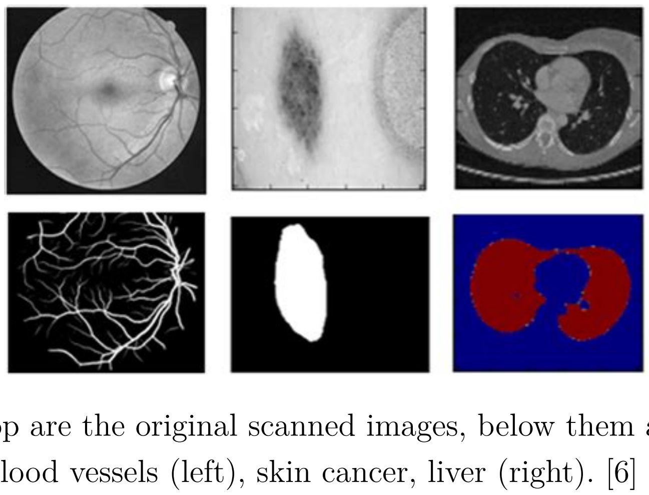


Fig. 10: At the top are the original scanned images, below them are the segmented images. Retinal blood vessels (left), skin cancer, liver (right). [6]

In the context of solving the image segmentation problem, it can be divided this task into two groups, semantic and instance segmentation [42]. Semantic segmentation is a task in which detected objects in an image and grouped according to defined categories. For example, in a face segmentation scene, it can detect the nose, the mouth, the right or left ear, and the eye. Whereas instance segmentation involves not only the detection of objects within the segmentation scene but also their categorization. An example then could be in the face segmentation scene the category of eyes and including the right and left eye in this category.

3.3.1 Artificial Neural Networks

A major development in the field of artificial intelligence has been made possible by artificial neural networks in short ANN. In its design, an artificial neural network has similarities to the biological computational model of the human brain, where signals are transmitted through a network of neurons. The first neural network can be considered the McCulloch-Pitts model (Fig. 11), which was published in 1943. Since then, the development in the field of artificial neural network has undergone great evolution, and from the theoretical model, nowadays artificial neural network is applied to a wide range of tasks.

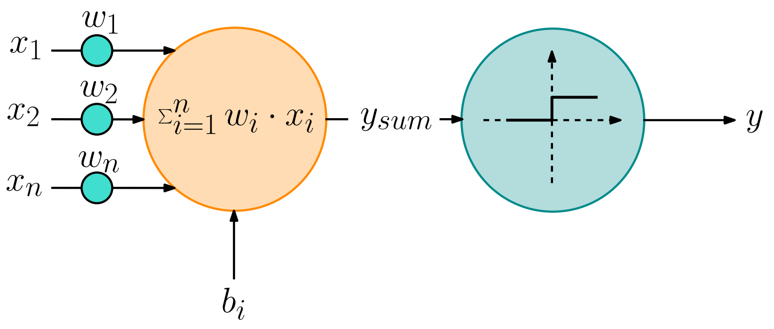


Fig. 11: McCulloch-Pitts model

The basic function of an artificial neural network is inspired by a biological neural network. In a biological neural network in the brain, the dendrites of the neuron receive input signals that are then processed in the cell body (soma). In addition, the processed signals are transmitted along the axon to the axonal terminals, where there are connections between other neurons are found. Thus, connections between neurons (synapses) allow signals to be transmitted from one neuron to another, which can then process and send them.

In an artificial neural network, first of all, the input data is transformed using a nonlinear function into a weighted sum of the inputs. This transformation is defined as a neural layer with a neural unit function. The outputs of one layer are then used as input to the next layer, where they are combined using iterative transformations into a final layer that produces a prediction. Training the neural network involves learning from the provided data, using changes in the weights and network parameters to minimize the difference between predictions and desired values. The artificial neural network has several basic ability properties:

- extract and represent dependencies in data that are not obvious
- solve strongly non-linear problems
- learn and generalise

These properties, however, come at the price of the need for computing power. A neural network is formed when multiple neurons are connected by inputs and outputs, but always consist of the same components, which can be expressed as the mathematical model of a neuron:

$$y(x) = f(\sum_{i=1}^{n} w_i \cdot x_i - b_i) \tag{1}$$

- y output signal
- f activation function
- x input signal
- w weights
- b bias

If there is at least one hidden layer in the network topology, it is a deep neural network. However, there can also be neural networks that are divided only into input and output layers. To date, there are many modifications and types of ANN such as Echo State Network, Neural Turing Machine, Extreme Learning Machine, Generative Adversarial Network, or Deep Convolutional Network Fig. 12. But, in general, we can divide the ANN into forward and recurrent. [35, 21]

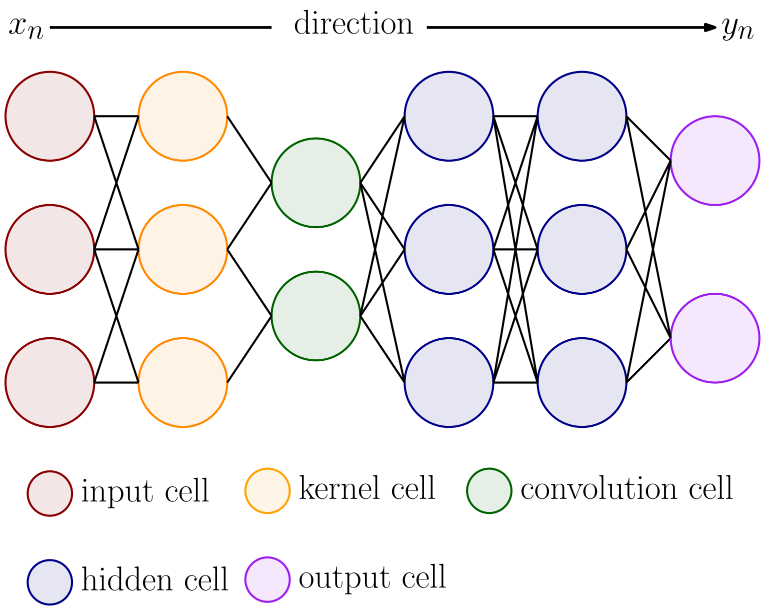


Fig. 12: Deep Convolutional Network scheme

3.3.2 Functions used in artificial neural networks

For the implementation and learning of artificial neural networks, it is convenient to use basic functions including optimization algorithms, loss functions, and activation functions.

Activation Functions

Activation functions determine the output of a neuron depending on its inputs [35]. To provide approximations of any continuous function, activation functions must be non-linear and continuously differentiable. However, not all known and used activation functions satisfy this condition.

One of the most popular activation functions is ReLu (Rectified Linear Unit). The ReLu function can be described mathematically by the Eq. 2. Compared to others, it provides faster learning and less computational complexity [21]. Although this activation function has many variations, it still retains two basic drawbacks:

- absence of derivation at the point $x = 0$
- zero value of the gradient in the interval $H(f) \in (0, \infty)$ that creates neurons incapable of learning

$$f(x) = max(0, x) \tag{2}$$

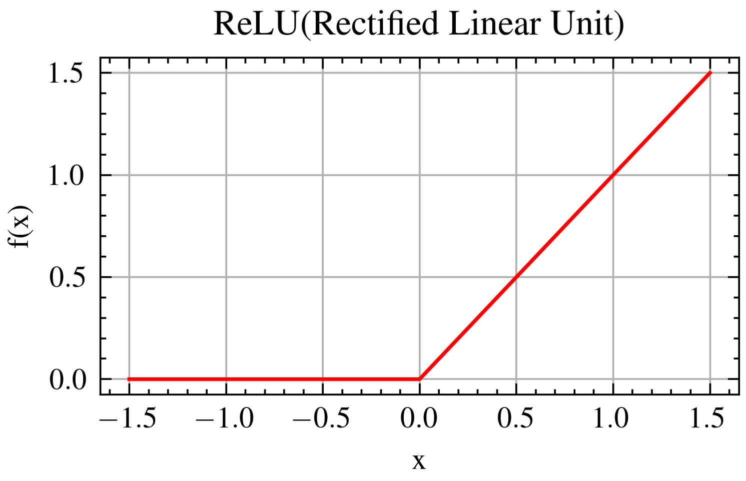


Fig. 13: ReLU activation function illustration

An activation function that finds its appropriate application in the output layer of the classification network is the Softmax (normalized exponential function). This activation function can be defined according to the Eq. 3, where the individual values determine the degree of classification.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{n} e^{x_j}} \tag{3}$$

Loss Functions

The loss function is a function used to measure the prediction error rate against the training data [35]. The loss function is chosen according to the type of problem to be solved. The choice greatly influences the speed and quality of learning of the neural network. At the same time, a validation loss function is also used, which is a function used to measure the prediction error rate against the validation data. One of the most widely used loss functions for image segmentation using ANN is the cross-entropy loss function.

Cross-entropy loss functions have many modifications [21]. Basic variants include binary cross-entropy. This method is used for two-class classification. While categorical cross-entropy is mainly used for classification into three or more classes. Each class corresponds to one output neuron. The function is defined by the relation:

$$E = -\sum_{i=1}^{N} y_i \log(\widehat{y}_i) \tag{4}$$

Optimizer

The optimizer is an optimization method used to minimize the value of the loss function. Depending on the nature of the problem, the optimization algorithm is selected. Several optimization algorithms can be used and the most famous ones are Adam, Adamax, Adatela, Adagrad (Fig. 14) or RMSProp.

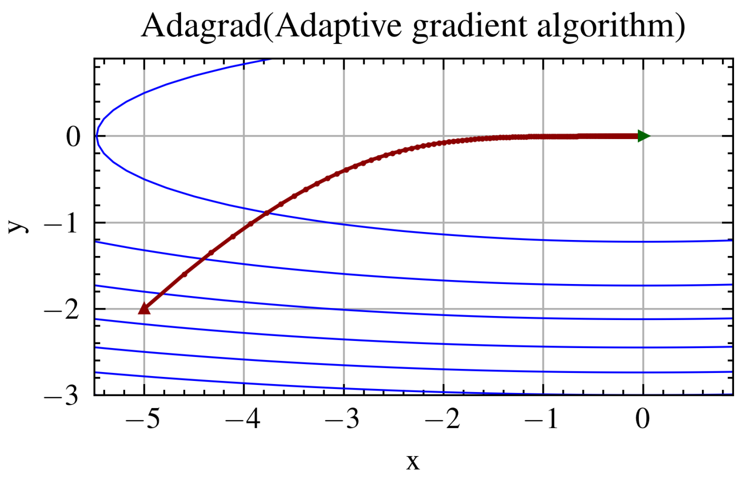


Fig. 14: Example of Adagrad optimizer during finding the minimum in point [0,0] from start point [-5,-2]. Adagrad in this example took 250 steps and an ellipsoidal set was used as the optimization function.

Adam (Adaptive moment estimation) is an optimization method that stands out for its computational efficiency and low memory requirements [21]. Based on these properties, this optimization method has become one of the most widely used methods for solving problems involving a large amount of data or parameters.

Back-propagation
So far, the most widely used learning method is backpropagation, also known as backprop. In this sense, learning can be seen as an optimization process in which the input signal of the training patterns is first forward-propagated, and then the optimizer (section 3.3.2) tries to minimize the loss function (section 3.3.2) arising from the difference between the actual and desired output [21]. Subsequently, the gradient of the loss function is calculated, back-propagated through the network, and the weights and thresholds of the neurons in each layer are updated. Published results describing this algorithm as early as 1986 by Rumelhart *et al.* [52].

3.3.3 Neural Networks for Segmentation

As already mentioned, the method using artificial neural networks for segmentation is classified as a hybrid method. The disadvantage of other methods is the non-universal approach but the advantage is less computational complexity. Whereas the use of the artificial neural network is very flexible and can be applied to a wide range of problems, but at the price of higher computational complexity. Image segmentation becomes the basis for further image processing such as classification or detection. Convolutional neural networks rely on the use of convolutional, pooling, and fully-connected layers, where if ANN has at least one convolutional neural layer we refer to them as a convolutional neural network in short CNN.

The basic idea of the convolutional layer is to extract information and reduce data from the image. The input image is processed by the convolutional kernel. The convolutional kernel can be expressed as a linear filter, which processes a small neighbourhood of representative pixels (pixel that is processed). The kernel size is usually odd (much smaller than image size) and must satisfy the condition, that for data reduction to occur, the kernel shift during convolution must be greater or equal to one. Several kernels can be applied to a single convolutional layer, and for an RGB image that perceives one kernel as a 3D array. The number of output transformations is determined by the number of kernels. The actual convolution operation is performed by already mentioned shifting the convolution kernel to the source image and then multiplying the shifted elements. The output of the layer is called a feature map. Write the resulting sum to the output, move the kernel and repeat the process. Mathematically, it can be expressed as:

$$g(x,y) = h(x,y) * k(x,y) = \sum_{(i,j)} \sum_{\in O} h(x-i, y-j) \cdot k(i,j) \tag{5}$$

- g result image
- h source image

- k convolution kernel
- O small neighbourhood of the representative pixel (currently processed)

Another very important layer for image segmentation is the pooling layer. The pooling layer, like convolution, is also used for data reduction to preserve the information. This fact has a major role in reducing the dimensionality of feature maps and reducing computational complexity. The objective of this layer is to perform a transformation, such that one pixel in the new image is represented as a group of neighbouring pixels, from the original one. The difference to the convolutional layer is that the transformation is known in advance. However, the weights and biases are unknown in the convolutional layer, this is determined from the training data. Pooling layers can be divided into max-pooling and average-pooling layers, which can be described as:

$$g(x,y) = \max_{(i,j)\in O} h(i,j) \tag{6}$$

$$g(x,y) = \sum_{(i,j)} \sum_{\in O} \frac{1}{mn} h(i,j) \tag{7}$$

- g result image
- h source image
- m width of window
- m height of window
- O small neighbourhood of the representative pixel (currently processed)

Building and designing a CNN is a very challenging task because there is no golden rule. Every dataset needs a unique approach, however, there are certain recommendations, that come from time-proven solutions. First of all ReLu activation function and its modifications are very often used. Batch normalization and appropriate initialization of parameters before training should prevent failure generalization and vanishing gradient problems. For most tasks, a deep neural network is needed. The vanishing gradient problem is an important issue in solving the training of convolutional neural networks, hence several typical network models emerged. [54, 21]

In recent years, several typical segmentation models have emerged. These models have their specific architectures but can use different backbones. Backbone is not a universal technical term used in *DeepLab* to refer to the feature extractor network [10]. The segmentation models from their original designs are modifiable in the form of these backbones, which have their specific uses, an example of these backbones being ResNet, DenseNet, MobileNet, and others.

U-Net
One of the most emblematic models in the field of image segmentation using a convolutional neural network is the U-Net [51]. U-net was first published in 2015 by the Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, and has been applied to biomedical image segmentation. The name U-Net is due to the architecture that can be drawn in the shape of a U. This segmentation model was developed as an improvement of the FCN (Fully Convolutional Networks for semantic segmentation) model. The main idea behind this segmentation model is to be able to work with less training data to achieve the most accurate segmentation.

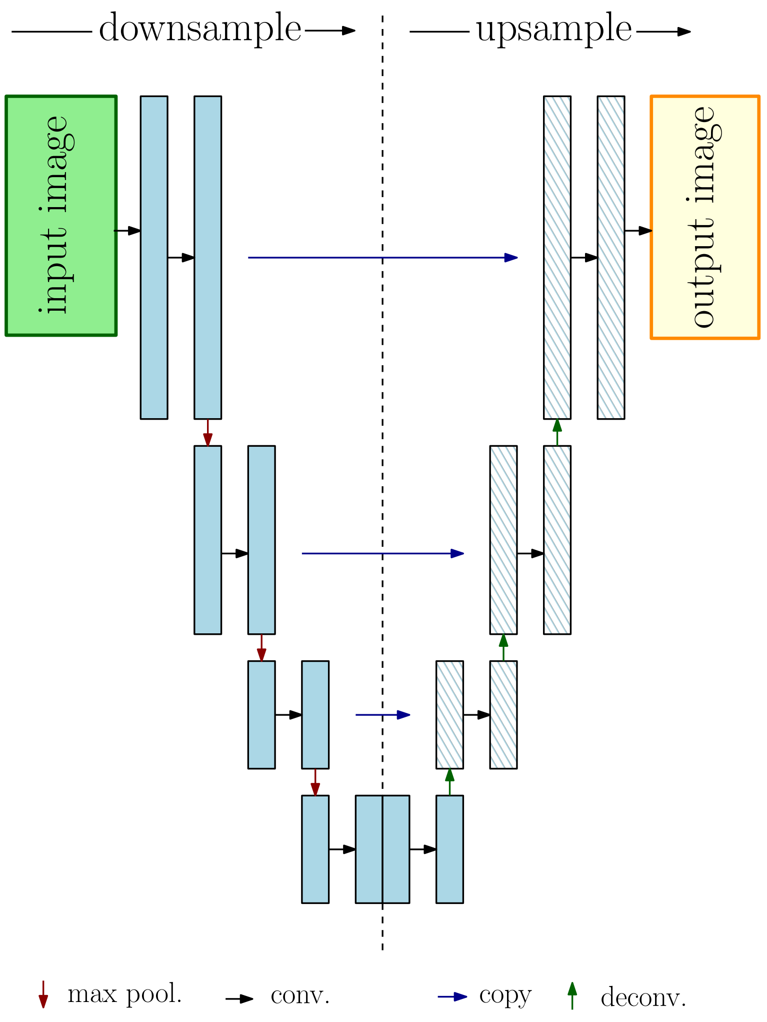


Fig. 15: U-Net architecture

The architecture of the U-Net model consists of two basic parts. The first part is called the contracting path and the second part is called the expanding path. The contracting path is modelled as a typical convolutional neural network architecture, with the main objective of performing downsampling and creating feature maps. During the downsampling process, the resolution of the input image data decreases, and the number of channels increases as they pass through the network. While the expansive path is designed for upsampling, in which the effort is to extract an image with as many channels as expected classes from the feature maps.

The original U-Net design of 2015 was based on asymmetrical left and right part Fig. 15. However, today a modification of this original design is used in which the contracting pathway is implicit in the form of a robust architecture such as MobileNet, EffecieNet, and others, which are called backbones.

ASPOCRNet

One of the most recent segmentation models with a modern approach can be identified as ASPOCRNet. However, this is an unofficial name, this model was officially called "HRNet + OCR + SegFix". This proposed segmentation model was published in 2021 by the Institute of Information Technology, Chinese Academy of Sciences University [66]. ASPOCRNet consists of the acronyms ASPP and OCR. Atrous Spatial Pyramid Pooling (ASPP) denotes a segmentation module that is used for semantic segmentation and performs a resampling of the given feature layer at multiple rates before convolution. This process performs data extraction from the original image using various parallel filters. These filters rely on each other to capture objects and useful image context at multiple scales [9]. OCR in the context of this segmentation model is defined as an equivalent approach to the Transformer encoding-decoding scheme [11]. It can be thought of the Transformer scheme as a substitution of convolutional and recurrent layers. When compared to convolutional neural networks, it is observed that, for example, kernel width directly affects the long-term dependencies that can be established between pairs of input and output positions. In the long run, convolutional layers require more computational power, which is time consuming.

The architecture of this segmentation model consists of two parts: the backbone and the OCR module. The backbone is used to predict the coarse segmentation in term object regions, and this prediction is fed as input to the OCR module. Next, the implemented OCR module performs a coarse segmentation prediction in terms of the soft object region, where the final segmentation is predicted from a linear function. A larger number of backbones can be used for the solution than those published in the original article, but this model is generally based on robust backbones architectures.

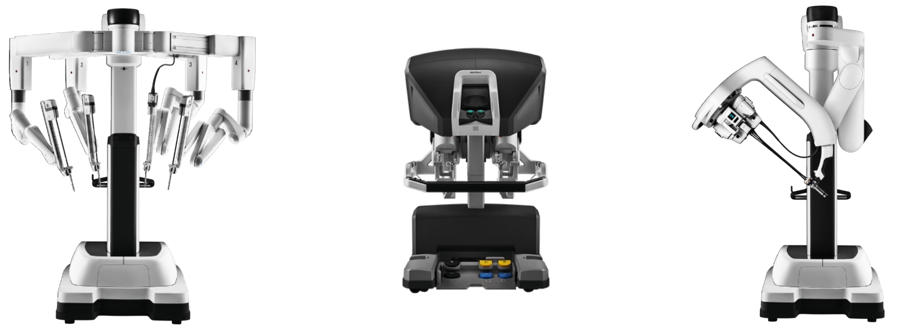
Chapter 4

Robots in Healthcare

The use of robots in healthcare dates back to the 1980s, when the first surgical robot, the PUMA 560, was used. The PUMA 560 robot was first deployed in 1985 during surgery when the robot's task was to insert a needle into the brain for a biopsy (medical diagnostic methods) [37]. The use of the robot reduced the risk of possible shaking of the surgical instrument, resulting in a more precise operation and proof that robots have their place in healthcare.

4.1 DaVinci

One of the most famous surgery robots is Da Vinci. The first series of Da Vinci surgery robot saw the light of day in 1999 [29]. The latest generation is called Da Vinci X, Xi, SP, and is the fourth generation represented by the California company Intuitive. This surgery system consists of a visual system, a surgeon's console, and robotic wrists with instruments.



(a) Da Vinci Xi (b) Surgery console (c) Da Vinci Sp

Fig. 16: Da Vinci [30]

The Da Vinci system does not primarily operate autonomously but requires an operator to control it. The operator can control the robotic arms with instruments and a binocular camera from the surgeon's control console. This system has gained its popularity by meeting the requirements of a minimally invasive approach with the elimination of unwanted shaking. The dominant feature was the possibility of 3D visualization using the binocular camera system, which resulted in much greater dexterity even in confined spaces. It also has the indisputable advantage of allowing the operator to sit in a comfortable position during long and demanding operations. The robotic arms have integrated many safety features and sensors such as hall, optical, magnetic, or infrared sensors. The Da Vinci system integrates its own operating and control system managed by Intuitive and offers a sophisticated platform for robotic-assisted surgery in urology, cardiology, neurology, and others.

4.2 Stäubli SEON

Robots can also help in the fight against cancer and are finding their place in oncology. The SEON project at the University Hospital Erlangen aims to develop magnetic nanoparticles that as a drug delivery transport in the fight against cancer [57]. This project aims to use an alternative robotic method to chemotherapy that would not have such an impact on the human body, since solid tumours would be precisely targeted. The result is a sophisticated connection between the Stäubli TX200 robot and the magnetic head as an end-effector. The magnetic head hovers over the patient's body, directing the magnetic nanoparticles with the active ingredient directly to the tumour. This application brings unquestionable facilitation of medical stuff. With the robotic arm, it is possible to achieve precise tracking of patient-specific treatment coordinates and avoidance of inaccuracies compared to manual movement of the magnet.

4.3 UVD Robots

Another meaningful use of robots in healthcare is a robotic disinfection system. UVD robots have come up with a disinfection system that consists of a mobile platform and a powerful UV-C emitter [62]. The company distributes two models (model-C and model-D). The latest model-C is the 3rd generation, this model is capable of disinfecting small and large areas simply by driving through. The robot can disinfect all environments that potentially contain dangerous microorganisms while being very gentle on surfaces. Its width is only 55 cm, so it can move even in very confined spaces. These autonomous robots are equipped with laser scanners and 3D cameras. The main purpose of these sensors is to detect obstacles and provide route information for route planning. The robot can also be manually controlled remotely and is capable of creating a temperature map and exposure map for the operator. The robot system includes several safety features that detect and stop processes if someone enters the room while being disinfected. Another safety feature is that the robot monitors if it is too close to an obstacle even when manually operated. Active disinfection time is estimated at two to two and half hours, with a charging time of three hours. The maximum speed is up to 5.4 $\mathrm{km} \cdot \mathrm{h}^{-1}$ and it can move in all directions.

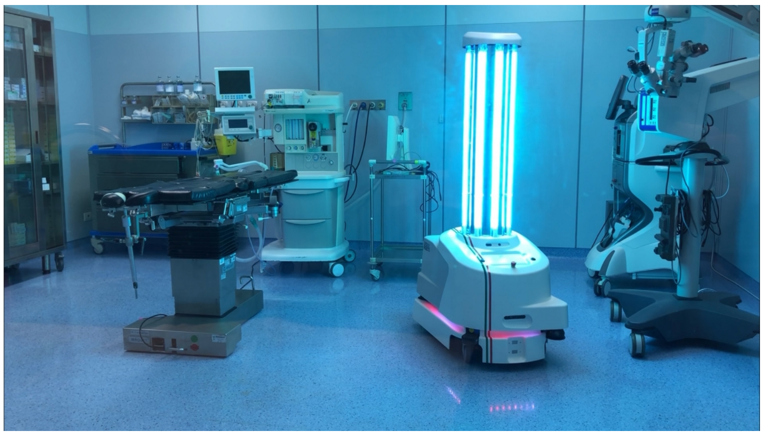


Fig. 17: UVD Robots Model B [62]

4.4 Smart Hospital

Hospitals, an essential part of the healthcare system, are complex systems that link different branches of medicine, with the primary goal of providing maximum care to patients. The future of healthcare in hospitals will be defined by the integration of new advanced technologies such as artificial intelligence, robotics, imaging methods, e-charging or building digital twin and others. The aim of incorporating the new technology is to make the work of medical staff easier and thus achieve maximum possible patient care with reduced mistakes. The main challenges of this new concept are to reduce operating costs, enable their work more efficiently and precisely, and optimize space efficiency. But at present, the most important challenge is cybersecurity and safety.

Robotics is also part of the smart hospital concept. Especially mobile and collaborative robots find their applications. An example would be a disinfection mobile platform or autonomous mobile platform for food, drugs, and medical supplies delivery. Larger autonomous platforms will also be able to transport patients accompanied by a nurse to the operating theatre. The project to combine an autonomous platform and a collaborative robot YuMi was undertaken by ABB company in partnership with the Texas Medical Center campus in Houston (Fig. 18) [1]. The project aims to design a robotic platform, that will assist medical and laboratory staff with laboratory and logistics tasks in hospitals. This robotic platform has the potential to perform a wide range of repetitive and time-consuming activities, including drug preparation, loading, and unloading centrifuges, pipetting and handling liquids, and picking and sorting tubes.

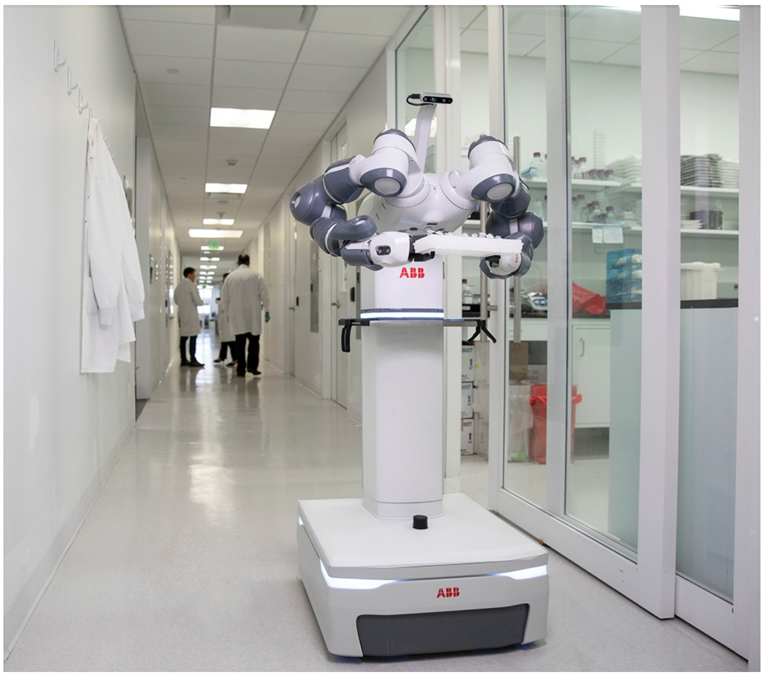


Fig. 18: ABB YuMi medical robotic mobile platform [1]

Collaborative robots are also finding applications as medical device operators. The Sawyer robot from Rethink Robotics is being tested in the ICU (Intensive Care Unit) environment at the Mayo Clinic Medical Simulation Center, Jacksonville Florida [19]. The Sawyer Black edition robot was chosen because of the relatively powerful integrated Cognex camera, but mainly because of the very easy programming and control. There's also the option to control using basic voice commands too. This 7-axis collaborative robot has been tested for basic device operations in the ICU. The operations are principally designed to respond to various alerts and to procure devices largely on their own, autonomously. An example of one of the many tasks is the operation of a device, where the robot holds a push-button for intravenous drug infusion. Some tasks may still be very complex and require the presence of medical personnel.

A huge role in the field of modern medicine is represented by the use of artificial intelligence in diagnostics, imaging, patient digital twin, surgery planning, telesurgery, surgery robots control, and others. An application example, where is used artificial neural network for diagnosis is Deep Learning-based Automatic Detection (DLAD) [23]. This ANN is designed for the diagnosis of major thoracic diseases on chest radiographs. The aim of diagnose the most common chest diseases from chest X-rays such as lung cancer, tuberculosis, pneumonia, and pneumothorax. However, artificial intelligence can be also used in medicine to solving problems such as the estimation of suture tensile force in robotic surgery. Poorly stitched sutures can break inside the patient postoperatively, where the patient may suffer bleeding or can also cause peritonitis. This experimental robotic application aims to solve this problem [31]. An integral part of the whole robotic application is the design of an artificial neural network that processes data from both the two image inputs and the end-effector position input. The network predicts the interaction forces during the suturing process.

VR (Virtual Reality) and AR (Augmented Reality) technology will have their specific use in the smart hospital (Fig. 19). The doctor will be able to see a digital twin model of the patient's body, so this will increase better navigation and orientation for diagnosis or surgery. This technology goes hand in hand with 3D reconstruction from CT, X-rays, MRI (Magnetic Resonance Imaging), PET (Positron Emission Tomography), or ultrasounds scans, using artificial intelligence. Which is also related to the planning of the surgery. The use of AR technology also offers patients the opportunity to draw maps and pathways through complex hospital buildings.

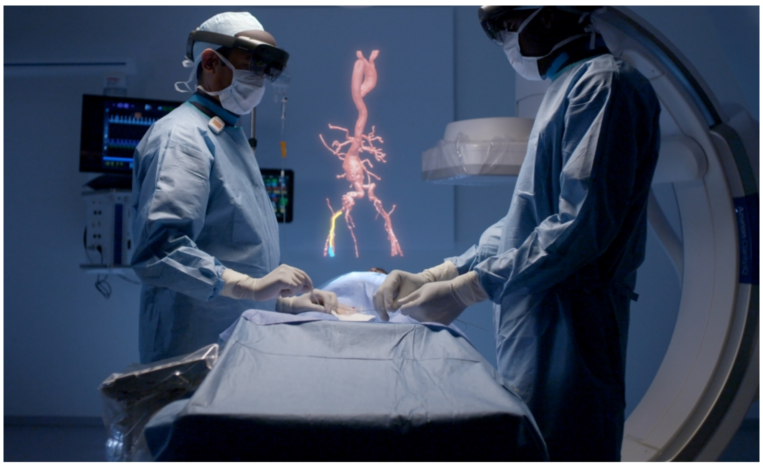


Fig. 19: Microsoft HoloLens 2 AR technology [8]

Thanks to advanced imaging technology, even the bold idea of remote surgery can be considered. The first complete remote surgery was conducted in 2001, with French surgeon Dr Jacques Marescaux in New York City performing a cholecystectomy on a patient in Strasbourg, France [36]. With the new 5G network, the idea of remote operations can be expanded, explored, and made more accessible.

4.5 Analysis Robots Applications in Medicine

Today's robotics already offers many possibilities for the development and use of robots in medicine. One of the most attractive applications in terms of development is cleaning and disinfection robots for hospitals, as these robots can be easily built and programmed thanks to current technology. A more complex application is to create a robotic platform for delivering drugs or testing samples. Surgical robots are the most difficult sector to develop, as they have to meet many requirements such for example maximum precision, reliability, and hygiene.

Among the more difficult tasks in terms of the development of a robotics platform for medicine may be also collecting samples from the nasal vestibule. This thesis aims to design an experimental robotic platform for swabbing from the nasal vestibule. To solve this problem, it is necessary to select robots. Due to the nature of the problem, it is advisable to use a robotic arm. There are plenty of manufacturers of robotic arms such as ABB, Stäubli, Epson, and Universal Robots. For solving the barrier-free problem, it is advisable to choose a collaborative robot from the entire range of robot types.

The main advantage of a collaborative robot over an industrial arm is a safety feature, cobot doesn't need a mechanical fence or sensor barriers, also, if a cobot hits an operator or patient, it shouldn't cause injury on impact and should stop immediately. Collaborative robots include the UR3 robot from Universal Robots CB-series, which was used to solve the defined experimental problem [59].

UR3 was chosen for the lower purchase price, availability and parameters. For applications to bigger spaces, UR3 can be substituted with, for example, UR5 in both variants (CB-series, e-series) [60]. The UR5 has a 330 mm longer reach range than the UR3, allowing it to work on longer distances, but this is at the cost of more weight and price. The main advantages of collaborative robots from Universal Robots are that they can move all joints in the range of ±360° and, thanks to their worldwide distribution, many packages for different robotic software can be used.

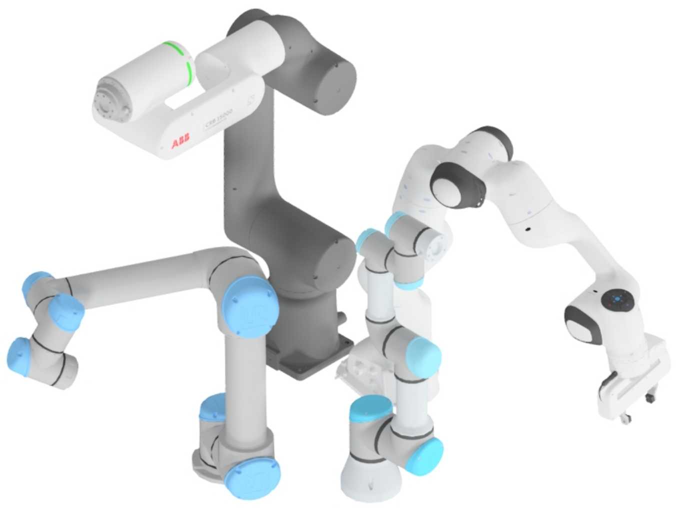


Fig. 20: ABB GoFa, UR5 (left), UR3, Franka Emika (right in back)

An ABB GoFa robot with a reach of 950 mm and a payload of 5 kg could also be used for this task [2]. However, the biggest disadvantage is the weight and limited movement of the robot axis. It also has its limitations in software packages and bigger prices. One of the biggest competitors for solving the given task can be considered Franka Emika [18]. This 7-axis collaborative robot is distributed with a two-finger gripper and a force sensing sensor, making it flexible compared to UR3 or ABB GoFa.

ABB's two-arm collaborative robot YuMi is also worth mentioning. Thanks to its unique design, it offers another wide range of applications compared to the aforementioned single-arm collaborative robots. In addition to its low payload, it is possible to use two robot arms simultaneously, while its own body is directly integrated with the controller. It can therefore be used advantageously with a combination of autonomous platform or static positioning, as appropriate. It offers advantages, for example, when programming several functions coincidentally, such as drug delivery, room disinfection, or the handling of other medical devices.

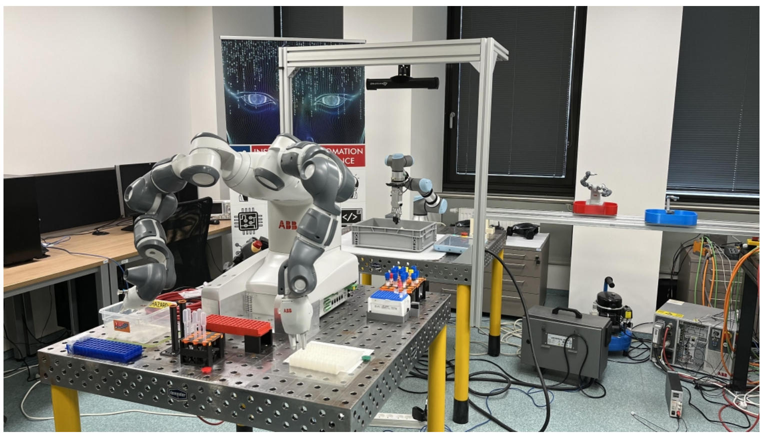


Fig. 21: Institute of Automation and Computer Science - YuMi and UR3

Chapter 5

Experimental Robotic Platform

The outbreak of the COVID-19 pandemic brought to light the fact that the most important sector of human society is healthcare. One of the frequent problem in health facilities was that they faced staff shortages. In order to, at least partially, help the medical staff, the idea of an robotic platform for automated antigen-testing was conceived. This is why Robo Medicinae I was created.

Robo Medicinae I is an experimental robotic platform that enables human-machine collaboration. It involves an operator connection that has control over the process of collecting nasal vestibule samples for antigen testing from patient. In this thesis, a semi-automated process was proposed, however, it is possible to change this configuration to a fully automated one, but this comes at the cost of safety. This robotic platform integrates the UR3 cobot with designed gripper on which the Intel RealSense D435i 3D camera, the OnRobot RG2 two-finger gripper and the OnRobot HEX-E force-torque sensor are mounted. In terms of software control, there is a web-based system that the operator can use to control processes such as access to the patients database, robot movement, streaming image from camera and so on. The whole system is designed to be simulated and then used in real-world applications. Several different technology areas are connected in the application, such as pick and place, digital twin, virtualisation, databases, machine vision and 3D data processing. All of this is assembled on a complex but modular platform, that creates a great base for tuning to real-world deploy.

It is a completely open-source project, where all materials are provided on *GitHub* server. This platform may not only be used for nasal vestibule sampling, but can be modified for other operations as well. Great emphasis was placed to design robotic platform as modular as possible, which could serve as a basis for various extensions.

robo medicinae I.

Fig. 22: Robo Medicinae I - Logo

5.1 Robotic Platform Design

The main device of the application is a collaborative robot. As part of the solution, the UR3 cobot from the Cybernetics and Robotics Laboratory in the Institute of Automation and Computer Science (IACS) was chosen. Although during the development phase the ABB YuMi collaborative robot was added, which would also be useful for the solution of the experimental robotic application.

Other integral parts of the design is the Intel RealSense D435i, the RG2, and the HEX-E sensor. These sensors and the gripper came together in the design of the customized gripper. Other parts to be designed were the rack and thread for the swab stick. The rendered CAD model is shown in Fig. 23

Everything is then connected to the control unit, which can take the form of a powerful single-board or desktop computer. As a web-based application, the operator has the possibility to control it from his or her device (e.g. tablet) using a web browser. However, the condition is that the device must be connected to the same local network.

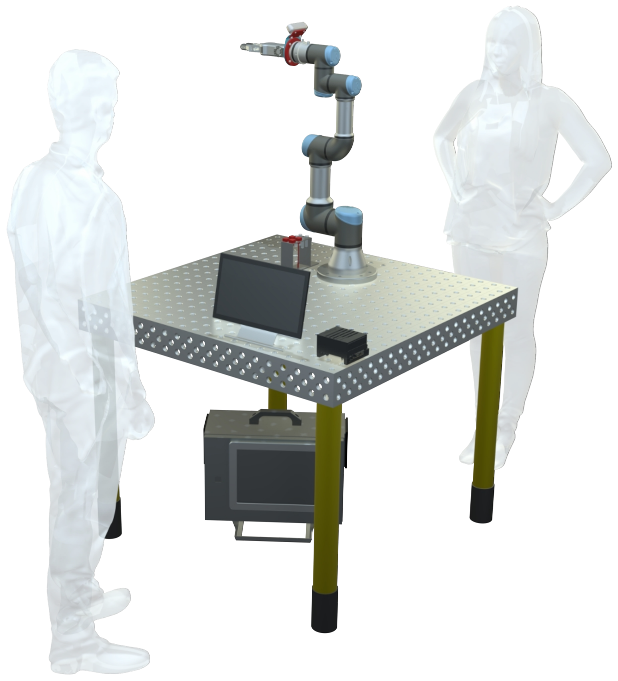


Fig. 23: Robo Medicinae I - Visualisation of robotic platform

5.1.1 Intel RealSense D435i

It is necessary to add eye sense to the robot for the defined problem solution. The main perception will be a vision. For the eyes of the UR3 robot, the Intel D435i camera was chosen. Another camera under consideration may be Photoneo Phoxi M (Fig. 25), which is also part of Cybernetics and Robotics Laboratory, IACS.

The Intel camera was chosen because of its smaller size and flexibility of use, while the Photoneo Phoxi M camera is larger in size and limited in use. The physical parameters of the cameras are crucial for the solution due to the mounting. If the camera is mounted statically (eye-to-hand), problems arise of limited scanning of the space. On the other hand, the camera attached to the robot (eye-in-hand), must be small enough not to limit the movement and path planning of the robot.

The Photoneo Phoxi M camera is based on structured light technology, whose major advantage is that the resolution can be up to submicron, but at the cost that any vibration or movement can lead to distortion of the 3D images and poor geometric measurement. These features come with the additional drawback of the high computing power required due to the stacking of several frames to obtain a 3D image. This is why the Photoneo camera has its computing unit centered on an Nvidia GPU Pascal. Another disadvantage is that it emits a lot of infrared light during the scan, which can be uncomfortable for the patient. The Intel RealSense D435i camera combines active IR stereo with a global shutter and IMU. This technology makes, the camera very flexible. Compared to the Photon PhoXi camera, it is not as precise, but it does not require a lot of computing power, which means that the physical parameters are smaller. The computing unit of the Intel camera is housed in the Intel RealSense Vision Processor D4. Overall, the Photoneo camera has its applications in medicine, but for tasks such as live measuring of human body shapes or the microstructures of human skin, Intel RealSense thanks to flexibility are used for a defined problem.

5.1.2 HEX-E Sensor

To ensure safety during testing, the solution also included the use of an OnRobot HEX-E sensor [41]. This sensor, when properly mounted, can give a true measurement of force and torque from 6 axes. This data provides a basis for a solution to a safety stop when the robot approaches a short distance to a patient and starts to push too hard. Measurement data is only provided from the centre of the HEX-E sensor. For example, if a patient tries to push on a robot joint, the sensor does not register this force. The HEX-E sensor is capable of measuring force $\pm$200N in the x,y,z-axis.

A competitor in this sensor category is for example the FT 300-S sensor from Robotiq [50]. The HEX-E sensor has a smaller measurement range of 100 N but has a lower purchase price than FT 300-S. The manufacturer Optoforce also claims that the noise-free resolution in the x,y-axis is 0.2 N and in the z-axis is 0.8 N, while in Robotiq it is 1 N.

5.1.3 RG2 Gripper

For the pick and place task, it was also necessary to mount a gripper. The solution was an OnRobot RG2 two-finger gripper [40]. In the application, it is necessary to pick the thread with the swab stick from the rack and then drop it into the tube. Thanks to this gripper, it can be used for different thread sizes, as it also has a maximum payload of 2 kg and a maximum spread of up to 110 mm.

A Robotiq 2F-140 two-finger gripper was also available for usage (Fig. 25) [49]. However, the design of this gripper is much larger than the RG2, with a maximum span of 30 mm and a payload of 0.5 kg more. Since the application requires gripping an 18×18 mm thread, the RG2 was chosen because it is much less robust. However, its disadvantage is in software control. When the command to open or close the gripper was executed, the Universal Robots controller stopped. This is because the RG2 gripper is connected directly to the UR3 and the RG2 URCap[1] must be installed on the UR3 control panel. Whereas the Robotiq 2F-140 is not connected directly to the robot but is connected to the control box.

5.1.4 Gripper

All these sensors and the gripper had to be combined and integrated into a complete gripper. The challenge was to create a modular gripper with individual parts that could be disassembled and replaced, for example, with another type of 3D camera. At the same time, it was also necessary to design the parts in such a way that they could be printed on a 3D printer in the shortest possible time.

Three parts are required for assembly. The first part is the base, which is attached to the HEX-E sensor. The camera mount and the RG2 can be attached to this base. Two variants have been developed for the Intel RealSense D435i 3D camera mount, the first variant is a 40° angular camera mount and the second variant is a 90° straight mount. Within the modularity, these mounts can be interchanged (illustrated in Fig. 24). Subsequently, a spacer was designed for the camera mount that serves as a connection between the camera mount and the camera itself. A simple rack and thread for the swab stick were designed to handle the entire testing process. All parts were designed in *Autodesk Inventor* and printed out on Prusa 3D printers of PLA plastic.

The gripper is mounted on the UR3 robot as follows. The HEX-E sensor is mounted on the UR3 end effector. Next, the base part with the 3D camera mount attached to the HEX-E sensor and the RG2 sheet metal holder is placed, where these parts are assembled. Then the RG2 gripper itself is mounted on the base and the spacer with the Intel RealSense D435i camera already mounted can be attached to the 3D camera mount.

[1] URCap is a Java archive, with modifiers for loading into the Universal Robots system.

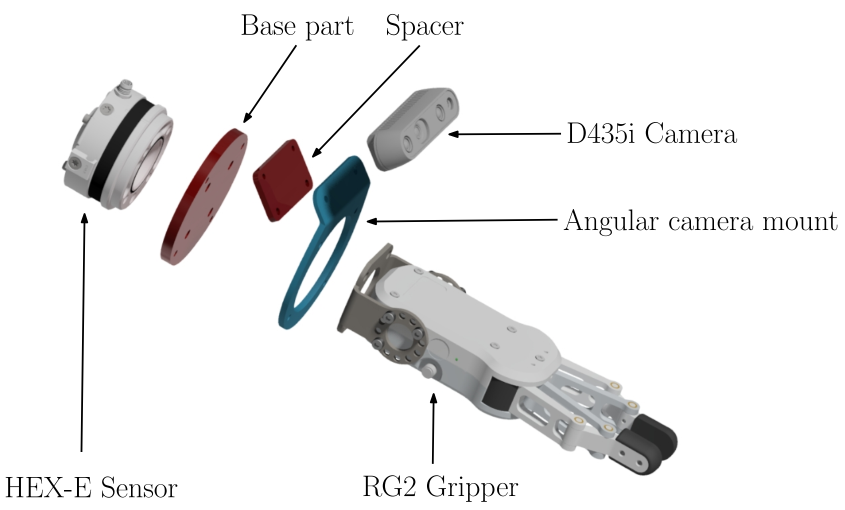
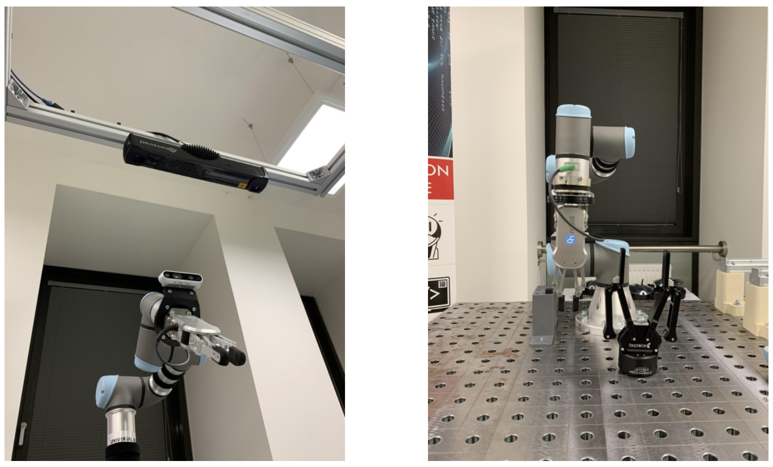


Fig. 24: Robo Medicinae I - Custom gripper design



(a) Photoneo Phoxi M and Intel RealSense D435i

(b) Robotiq 2F-140 and OnRobot RG2

Fig. 25: Comparison to alternative camera and two-finger gripper

5.2 Software Topology

The basic idea of the software topology is to create a local web server that the operator can connect to from another device on the same local network. This local server is essentially a central unit and controls various parts such as the robot, the sensors, access to the database, etc (Fig. 26). The operator can control the whole process and the different parts from a web browser, where the whole HMI is produced.

This software topology leads to a division into back-end and front-end. The back-end is understood as all processes that the operator cannot see from the remote access, i.e. everything that the server computer runs in the background. Within the application, this includes, for example, the mapping of URLs, image processing or compute functions. However, in terms of front-end, this can be defined as everything that the user can see from his remote access in a web browser. This includes, for example, page rendering, animations, etc.

The development of the back-end part of the server was based on the Python 3 with auxiliary scripts in the Bash programming language. The *Poetry* package manager handles all updates and the management of the various Python libraries, which were used. The front-end of the application was based on the programming languages HTML, CSS, and JavaScript. Working with JavaScript has been made easier thanks to the *jQuery* library. A Robotic Operating System (ROS) was chosen to simulate and control the UR3 robot, and also control the RG2 two-finger gripper and the HEX-E sensor during real-world testing.

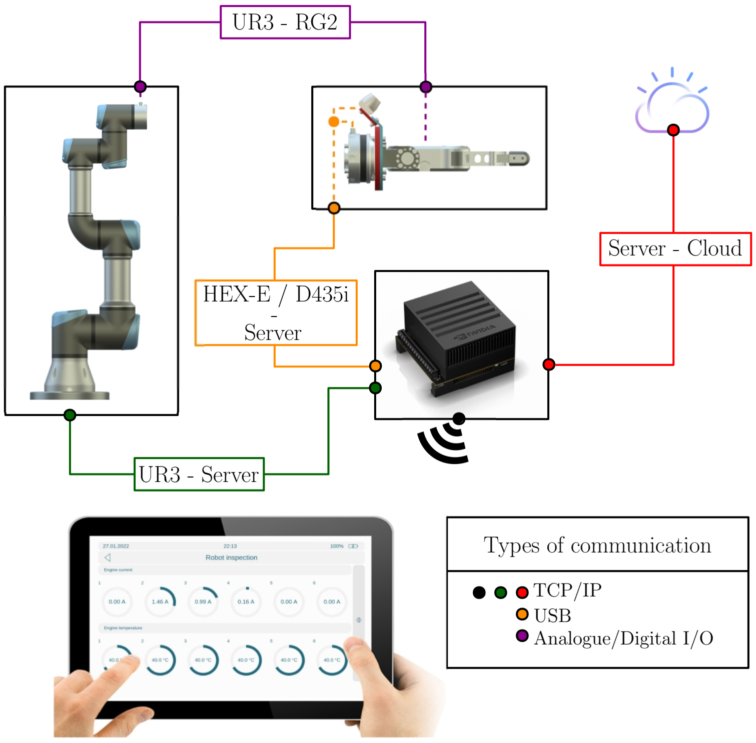


Fig. 26: Robo Medicinae I - Software topology

Different libraries were used for the back-end, the front-end, or the ROS. Everything had to be incorporated to make the system work on such a complex scale. Thanks to these technologies, it has been possible to create an application that offers the operator the following possibilities:

- Acces to patient and operators databases
- Search the patient database based on a Personal Identification Number (PIN) or a scanned QR code with a PIN from a colour image.
- Display stream colour, depth, infrared image from 3D camera

- FaceID - matching the patient from the streamed colour image to a photo from the patient database
- Face Detection - detection of human face landmarks
- Identification centre of nostril - segmentation model CNN to detect nostril, based on this centroid calculation
- Face Scan - scan align colour to depth image, based on this reconstruct 3D point cloud
- Show point cloud in a web browser
- Connect or disconnect robots - switch multiple UR3
- Manual control of robot rotation before scan face is processed
- Start the process of motion to the detected centre of the nostril
- Manual control of robot - simple HMI for control of cobot joints
- Display digital twin of cobot in the web browser
- Display basic data from cobot as a temperature, current or coordinates
- Generate PDF document based on HTML input
- Basic data about weather

5.2.1 Flask

The cornerstone of a web-based application is the web framework. It is not necessary to use it, but it will facilitate the developer's work. In general, the web application framework is primarily designed to support web application development. Among these web frameworks, it may include Flask. Flask is not exactly a web framework in the true sense of the word, but a micro web framework [17]. Micro classification means, that it maintains a simple but extensible core, which makes Flask a very powerful and efficient tool. This modularity makes it possible to extend this server core with any desired extensions. In this regard, extensions are understood as validation forms, database abstraction layers or Simple Mail Transfer Protocol (SMTP) interfaces, and so on.

A well-known and popular challenger is *Django*, compared to Flask it is a complete web framework that integrates many pre-built features such as user authentication or geolocation and others. One of the advantages of *Django* is that it comes with all the necessary security features to protect the web applications. Thanks to this full range of functions are particularly suitable for large projects where a lot of traffic (user accesses) is to be expected. However, *Django* is full-stack web framework status that is redeemed at the price of less customization. Customization is dominated by Flask, which finds its preference in smaller and medium-sized projects. Because of this flexibility and the possibility to extend only the necessary extensions, Flask was chosen for the solution.

Core library Flask integrates *Werkzeg's WSGI*[2] toolkit and the *Jinja2* template engine. Afterwards, the *SQLAlchemy*[3] library was installed in these models to handle database access and modification options. To access the database and create a simple sign-in system, it was necessary to install the *Flask-WTF* library. This library provides functions such as internationalization, and important CSRF[4] protection and others. The latest Flask extension to be installed is *Flask-login.* This extension provides user session management. It handles the common tasks of signing in, signing out, and remembering operators' sessions over periods.

Flask blueprints encapsulate a template, static files, and other elements that can be applied to an application [46]. In their essence, they can be considered a mould. Their main function is to create an organizable and extendable structure, and also avoid cyclic imports. With blueprints, it is possible to create several complex structures, that include both different and equal components. It is possible to imagine this in a structure where it is needed to have a user, administrators, an authors section, and so on. Although the Flask is very modular, it has some rules regarding blueprints for it to work properly. There must be two folders in the file system, templates and static. The templates folder contains only the files that the template engine can work with. The Static folder contains the assets used by the templates, such as CSS files, JavaScript files, or 3D models. An example of this structure is shown in Fig. 27.

```
./flask_server
├── run.py
├── config.py
└── /app
    ├── __init__.py
    ├── routes.py
    ├── routes_auth.py
    ├── /templates
    └── /static
```

Fig. 27: Basic Flask software topology

In the application of the experimental robotic application, two blueprints were used in the Flask. This is because one level, where the individual routes handle e.g. data collection and further streaming, and the next level is the organizational

[2] Web Server Gateway Interface (WSGI) is a standard for Python web application development, the specification describes how a web server communicates with web applications and how web applications can be chained together to process a single request [64].

[3] SQLAlchemy is the Python SQL toolkit that gives the power and flexibility of SQL in databases [56].

[4] Cross-Site Request Forgery (CSRF) is a web security vulnerability that allows an attacker to induces users to perform actions that they do not intend to perform [53].

level, where e.g. sign in, sign out, or validation of either admin or user access is solved.

One of the features that needed to be addressed is multi-level access. The resulting application has two levels, a basic user level, and the second level, which is the administrator level. The administrator level provides user database management, access management, and much more. This extension has been programmed as a Python wrapper itself, which compares whether the signed-in in member has administrator or user status in the user database.

5.2.2 ROS

As this is a robotic application, it is also necessary to control and simulate the UR3 robot. A wide range of many tools that can be used, is also ROS. Robot Operating System is one of the most popular frameworks in the field of robotics. ROS is neither a framework nor an operating system in the traditional sense of the term, but a meta operating system that provides a highly structured communication layer on top of the host operating system of a heterogeneous computing cluster [48]. Thus, as a conventional operating system, the ROS meta operating system provides a variety of services such as a hardware abstraction lawyer, low-level device control, as well as functions such as inter-process message passing and package manager. One of the main parts is the tools and libraries for writing and executing code, although it is possible to integrate with real-time code, it is not a real-time framework. The main advantage is its strong community of developers, thanks to which it is now possible for many additional tools and libraries can be integrated, allowing for greater programming convenience and the necessary resources to solve certain problems. The basic tools that have in the full version are the ROS Core, the *Gazebo* simulation environment, and the 3D visualization tool *Rviz* (Fig. 28). The main programming languages offered are Python and C++. However, there is also a variant of ROS2, in which the new generation provides developers with a completely new dimension of uses compared to the first generation. The main improvement came in the form of the ability to control and use multiple robots simultaneously, thus providing robotic safety and the long-awaited real-time control. The disadvantage of this version was the limited number of developers creating packages, but the situation is changing and the transition to this new generation is gradual. ROS distribution Melodic Morenia was chosen for the solution because it offers a wide range of tools and the possibility of rapid development in simulation and transfers to real-world testing, basic version of programming languages is Python 2 and C++ 11.

Other tools can be used. One possible option is, for example, to use the *Socket* library and send commands in the URScript programming language. Another option is to use a pre-programmed library to access UR interfaces that facilitate the use of

UR robotic manipulators by external applications. Or use, for example, the 3D simulation software *Visual Components* (Fig. 29). This program has a plug-in for post-processing simulations. This plug-in works by putting a simulation programmed in Python into URScript code. It can then be loaded into the UR Polyscope.

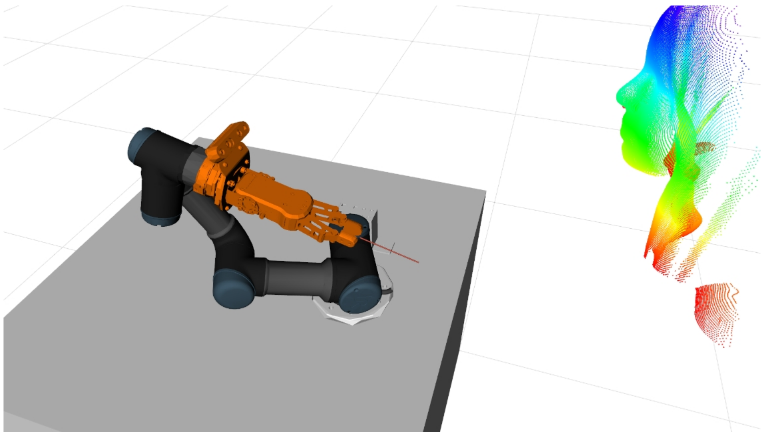


Fig. 28: Robo Medicinae I - ROS Rviz simulated scene

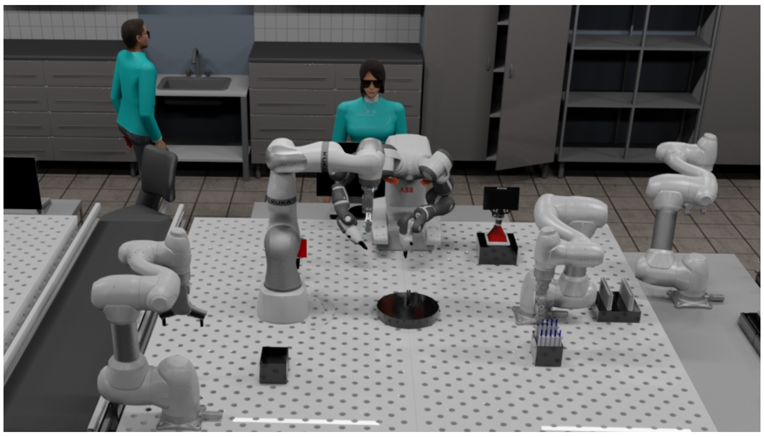


Fig. 29: Visual Components - Example of a test laboratory

The platform that has been added to ROS is *MoveIT*. *MoveIT* is one of the cornerstones of robotic arm control using ROS. The platform has a wide range of interlocking add-ons. In particular, the platform provides a motion planning function using planning algorithms from library OMPL (Open Motion Planning Library), alternatively from others such as STOMP (Stochastic Trajectory Optimization for Motion Planning), etc. Another indispensable function of the platform is, for example, collision detection or obstacle avoidance. However, other development packages had to be used such as *Universal_Robots_ROS_Driver*, *rosbridge*, *tf2_web_republisher*, and *optoforce*. Its structure in the catkin workspace can be seen in Fig. 30.

ROS plays a vital role in this application, it is used to simulate and then control the robot, sensors, and gripper in real-world testing. This flexibility makes it a capable development tool. When solving the application a package was created, called *robo_medicinae*, this package integrates three basic parts:

- *robo_platform* - integrates the scripts for running the *Gazebo* simulation scene and the modified scripts from *Universal_Robots_ROS_Driver* for real-world control cobot
- *robo_moveit* - integrates the *MoveIT* package with modified scripts, together with the execution of *Rviz*
- *robo_control* - implies Python 2 control scripts

```
./catkin_workspace
└── /src
    ├── /robo_medicinae
    │   ├── /robo_platform
    │   ├── /robo_moveit
    │   └── /robo_control
    ├── /optoforce
    ├── /universal_robots
    └── /Universal_Robots_ROS_Driver
```

Fig. 30: Robo Medicinae I - ROS structure

5.2.3 Communication Server-ROS

As part of the robotic platform implementation, it was necessary to ensure direct communication between the Flask - Client - ROS. Therefore, to mediate this communication it was necessary to install the *rosbridge* package. *Rosbridge* provides the JSON API to functionality for non-ROS programs. This means that it is possible to connect from .NET applications or use WebSocket communication from a web browser. The solution was simply to use WebSocket communication whereby using the ROS JavaScript libraries it was possible to connect to *rosbridge.* Thanks to this communication it was possible to send and receive messages of different data types such as String, Float32, or special types such as JointState.

The main usage of this communication is the management of the digital twin in a web browser. To view the digital twin it was necessary to install another package, *tf2_web_republisher.* The main purpose of this package is to provide to throttle and pre-compute the TF[5] transformation. By running *rosbridge* and theappropriate JavaScript libraries on the client-side, the digital twin will be transmitted, but one of the important conditions is to have in the static Flask directory (Fig. 31),

[5] TF is a tool that keeps track of multiple coordinate frames over time.

the appropriate file system hierarchy along with the 3D models where the files have for example the .stl or .dae extension.

```
./flask_server
└─/app
   └─/static
      ├─/ur_description
      │  └─/meshes
      └─/robo_platform
         └─/meshes
```

Fig. 31: Digital twin Flask structure

5.2.4 Functions of Robotic Platform

As mentioned above, the application offers several different functions, from facial identification to the digital twin robot. These functions have been developed based on various Python and JavaScript libraries. To keep the Python libraries up to date and maintainable, the *Poetry* tool was used for their management (Fig. 32). However, it is also possible to use, for example, *PIP* or less suitably *Anaconda*. No library manager has been used to maintain the JavaScript libraries, but they are imported directly using *cdnjs*[6].

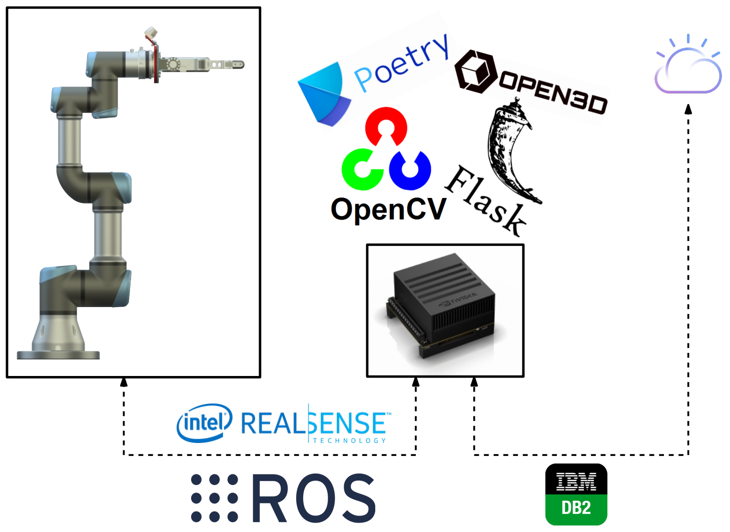
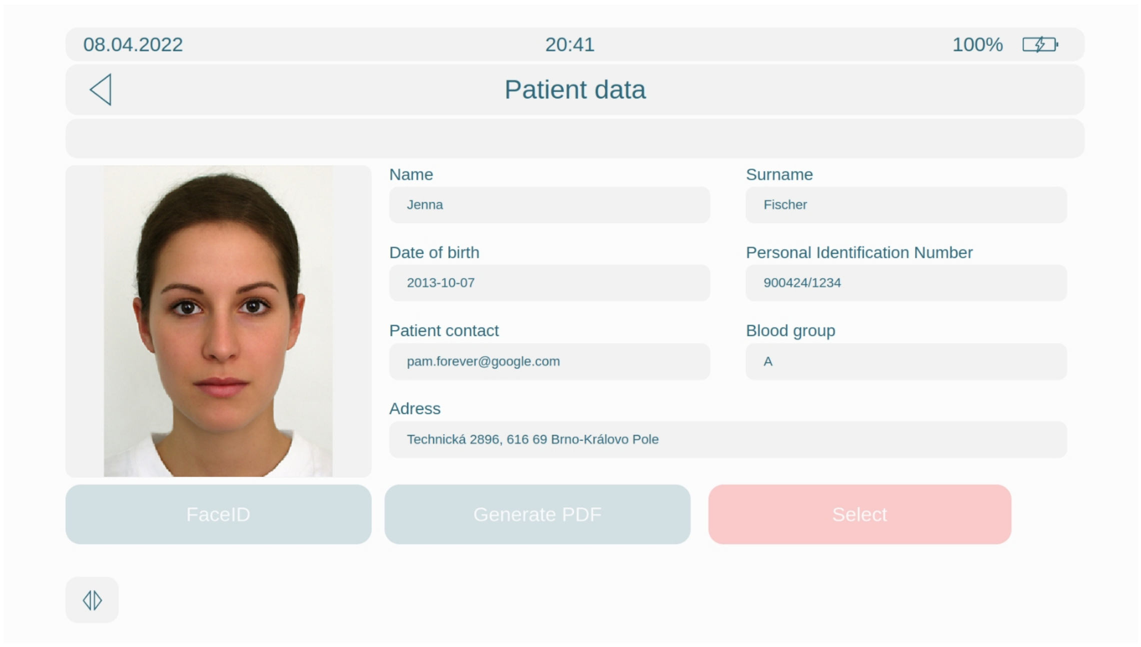


Fig. 32: Robo Medicinae I - Main libraries and frameworks

[6] cdnjs is a free and open-source software content delivery network hosted by web-based open-source archive Cloudflare.

Databases Access

One of the basic functions of the application is the possibility of signing in via the user database. This is a local database, so it is located on the same device as Flask. Access to the database is handled by the *SQLAlchemy* library and the database engine used is *SQLite*. It is a relational database that is not branched but contains a single table with attributes user id, name, surname, title, department, password, email, role, and session token.



Name	Surname
Jenna	Fischer
Date of birth	Personal Identification Number
2013-10-07	900424/1234
Patient contact	Blood group
pam.forever@google.com	A
Adress	
Technická 2896, 616 69 Brno-Královo Pole	

Fig. 33: Robo Medicinae I - Patient data

Once signed into the application, the operator has the possibility to search in the patient database. The patient database is not located at the local centre but is mediated by a third party, such as Google, Amazon, or IBM. In the context of the whole application, it is mainly a demonstration, as no hospital or healthcare institution lends out its real patient database and tries to strongly protect this data. Therefore, in solving this application, the patient database was stored in the IBM cloud for the demonstration. This demo represents the connection to the database outside the local network, as if it were deployed. IBM provides a relational database but uses a database engine called *DB2*. The attributes of this database are id, PIN, name, surname, address, dob (day of birth), contact, blood, and photo (Fig. 33). There are two search options:

- Searches in the database based on the input label. When the search is based on PIN or surname matching.
- Search based on QR code retrieval. The function is based on the principle of processing a colour image from a 3D camera and applying the QR code read algorithm to this image. If the code is read correctly, a search is performed in

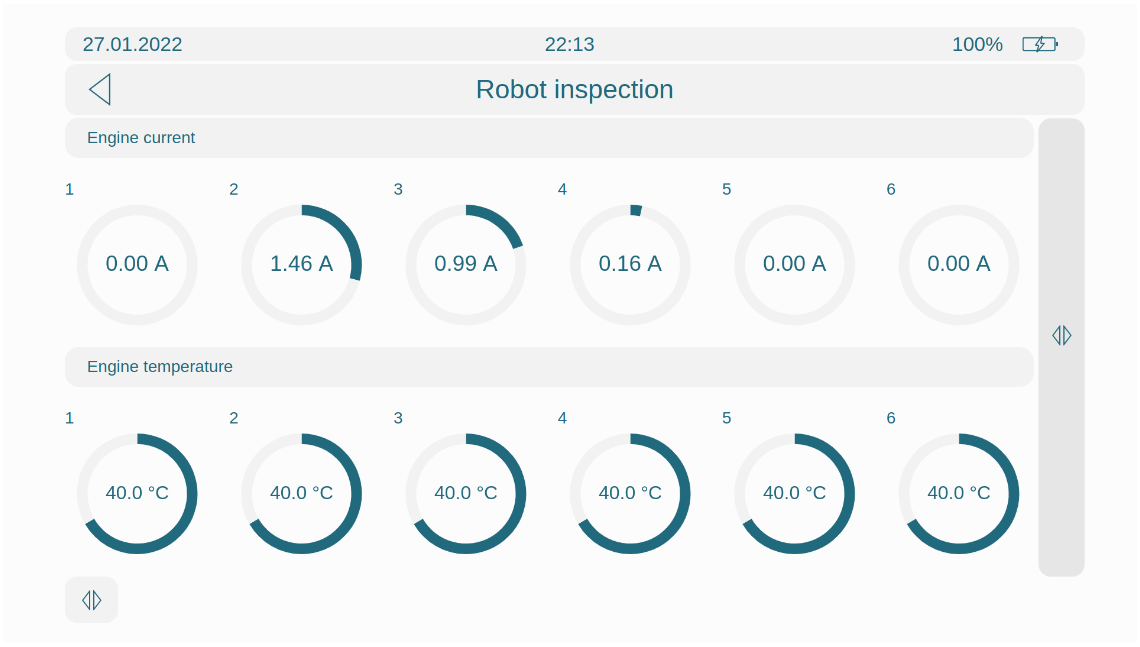
the database based on the PIN. The *Pyzbar* library was used for the QR read and *Pyrealsense* to control 3D camera.

Data Provision

Among other, not very important functions are the provision of information to the operator. This includes, for example, information about the current weather, humidity, and pressure. The essence of this informative function is to send a query to the *wttr.in* server and this response is sent back to the client-side.

There is also the option to stream colour, depth, and infrared images. This process of controlling the 3D camera and then streaming it to the client is implemented by using the *Pyrealsense* library, which allows the Intel RealSense D435i 3D camera to be controlled. The images are streamed to the client at 640 × 480 pixels, with a hardware resolution of 60 FPS. This is due to the run of all cameras drivers, which have different limits in terms of resolution and frames per second.

Added operator option is to obtain information about the current temperature and current in the motors from cobot UR3 (Fig. 34). This data is obtained via Socket communication with the cobot on the IP address of the UR3 and port 30003, which is designed for real-time communication. On the right side, the operator can click on an additional sidebar panel, which shows the .svg format of the UR3 cobot. After clicking on each of the joints, explanations are displayed.


27.01.2022 22:13 100%

Robot inspection

Engine current

1	2	3	4	5	6
0.00 A	1.46 A	0.99 A	0.16 A	0.00 A	0.00 A

Engine temperature

1	2	3	4	5	6
40.0 °C	40.0 °C	40.0 °C	40.0 °C	40.0 °C	40.0 °C

Fig. 34: Robo Medicinae I - Robot inspection

FaceID and Face Detection

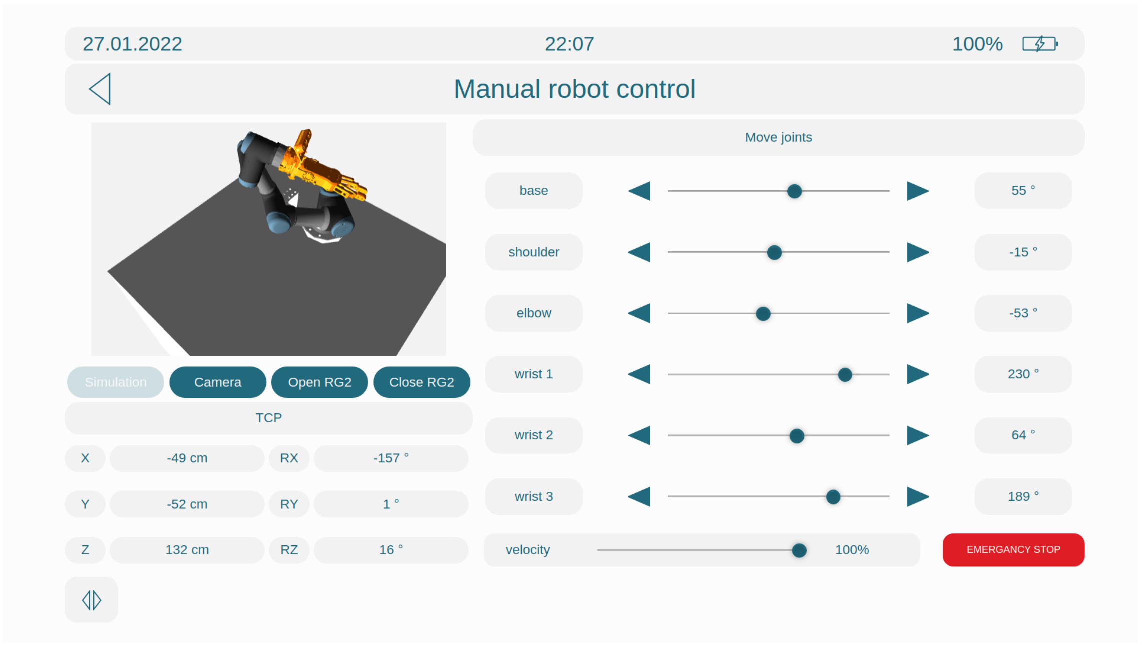
Interesting features before performing a face scan are functions to validate, whether the patient matches the patient selected from the patient database. For this function, the colour image from the camera has been used, by using the *Pyrealsense* library. The *face_recognition* library was used for the validation. This is a library that has implemented a convolutional neural network that compares two images, one is the template for validation and the other is the image that is validated. Where the validated key is the face. This process is very similar to the modern technology used to unlock smartphones, for example.

The next function is face detection, this function again involves an already trained neural network that detects landmarks of the face. This function is preferably used when the patient is too tall or too short so that the cobot subsequently rotates to a suitable scanning position before the face scan process. This proper position is defined by a fixed frame, where if the reference points of the face are outside this frame, the operator will not be able to continue the process. This function was programmed using the Python *OpenCV* library and *Pyrealsense*.

Manual Robot Control

Another option for the operator is to connect and disconnect the ROS cobot controllers from Flask. To connect the UR3 to the application it must be on the same local network. This function was created because of the possibility to control multiple robots, but not at the same time. The ability to control multiple robots at the same time would be available when using ROS2. It is a simple process, the operator checks the IP address of the cobot on the UR Polyscope control panel and then enters it into the application in the web browser. If the operator enters an IP address in the form 127.0.0.1, the *Gazebo* simulation environment, ROS, and the corresponding modified *MoveIT* control script are started. If the IP address is different, then the modified *Universal_Robots_ROS_Driver* control script is run and the corresponding modified *MoveIT* control script is also run. All these scripts are integrated in packages *robo_platform* and *robo_moveit*. When disconnected, these programs simply stop working. Communication is realized between the client device and the server by *rosbridge*.

Within the application, a simple control panel has been designed that allows the operator to control the individual joints of the robot, either utilizing sliders or buttons that rotate in two direction rotation angles. Other control elements are the adjustment of the movement speed or a stop button. In the robot's manual control, there is also a twin digital view of the robot, which can be toggled with the colour image of the Intel RealSense D435i. Next to the buttons for switching between the digital twin and the camera image there are buttons for opening and closing the RG2 gripper. There is also a small user information panel with eulerian rotations and Cartesian coordinates of the x,y,z axes. Design and implementation of the output in Fig. 35.


27.01.2022 22:07 100%

Manual robot control

Move joints

base	55 °
shoulder	-15 °
elbow	-53 °
wrist 1	230 °
wrist 2	64 °
wrist 3	189 °
velocity	100%

TCP

X	-49 cm	RX	-157 °
Y	-52 cm	RY	1 °
Z	132 cm	RZ	16 °

Fig. 35: Robo Medicinae I - Manual robot control

Generate PDF Document

An additional interesting feature is the PDF generation based on the .html input script. This is a function that acts as a validation certificate in the application that the patient has undergone the testing process. For the solution, the *pdfkit* library was used. Input is .html script with data from the patient database and then thanks to the tools of this library it is converted into .pdf format. Finally converted PDF format is sent to the client, where the client has the option to download it.

5.3 Face Scan Process

An integral part of the whole application is the facial scanning process. This is a function in which the patient's head is scanned by using a 3D camera, followed by detection of the nostrils using an convolutional neural network. Then 3D reconstruction of the image from depth and colour images is processed. To start this process, the operator must first perform faceID and facial validation to ensure that the correct patient is scanned and in a good position.

One of the main problems encountered in the solution was the detection of nostrils. Since there is no pre-trained model for image segmentation with nostril detection, it was necessary to design a customized solution. The solution to this problem was based on a complex artificial neural network tool, more specifically a convolutional neural network segmentation model.

5.3.1 CNN for Detection Nostrils

Two convolutional neural network segmentation models, namely U-Net and AS-POCRNet, were selected to solve the nostril detection. However, before these models can be learned, it is necessary to collect the dataset and perform so-called masking. Masking is the process of preparing a dataset by contrasting the areas of interest in the original image and creating a new image. This process is used in neural network learning methods with a supervisor. To solve the problem of nostril detection, regions of interest have been labelled using a polygon.

The data were collected using a 3D camera, where it was necessary to capture a colour image aligned with the depth image. This alignment is justified by the 3D reconstruction. The resolution of the captured colour image is 640×480 pixels with suppression of the environmental scene where the disparity offset has a fixed value. A total of 100 images were captured, in three different environments, and three persons were captured.

Afterwards, the images were labelled using the open-source tool *Make-Sense*. This was followed by masking in which three classes of masks were created. The labelled left nostril has an image value of 1, the labelled right nostril has a value of 3 and the rest of the scene has a value of 2. The data was then split into a ratio of 80% training and 20% validation. The simplified process represents Fig. 36.

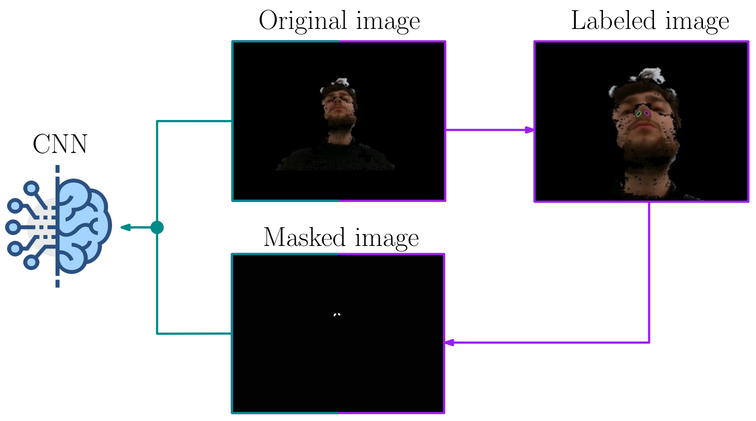


Fig. 36: Pre-process: Original image - Labeled image - Masked image

Latterly, a .ipynb (Python Jupyter Notebook) script was created. This script was programmed to enable neural network training both on local devices and in the cloud such as *JupyterLab*, *Google Colab*, or *Microsoft Azure*. The following functions have been implemented in this script:

- Load images and masks from folder
- Images and masks augmentation
- Image and masks preprocessing
- Build cnn model

- Train cnn model

Since the dataset contains only 100 original images with masks, which were subsequently divided into training and validation images, it was also necessary to augment the data to create a larger training set. The augmentation was performed using the *Albumentation* library, which supports image transformation functions such as vertical or horizontal flipping, pseudo-random contrast change, and others. These transformations were applied only to the training data. The validation data were transferred only using the padding function (padding space on all sides of the image). To create a larger training data set, 90 transitions for each training image and 10 transformations for each test image were augmented. Before using this data in the learning process, a pre-processing had to be done, in which the data is converted into the correct data type and batches.

Another the most important part is the compilation of the convolutional neural network model. This is the step before the actual learning, where parameters such as the optimizer, the number of classes, the loss function, and others are set. For the solution, libraries were used that already have some segmentation models pre-programmed with the possibility of choosing the backbones as well. These are the *Segmentation_Models* and the *TensorFlow_Advanced_Segmentation_Models* Python libraries.

5.3.2 CNN Training and Results

Among the final parts of the nostril, the detection process is the learning of the neural network. In this part, all parameters such as the number of epochs, callbacks, and others are configured. Both convolutional neural networks models were trained on a desktop computer with the *TensorFlow* framework optimized to transfer the computation to an Nvidia GTX 1050 Ti graphics card. To compare these models the following common parameters were set:

The learning process was then monitored using the machine learning tool *Weights & Biases*. Thanks to this tool, it is possible to monitor the current loss function, the accuracy of the model, or possibly the GPU temperature or the GPU utilization after each epoch. The training termination criterion is set to reach the maximum number of epochs.

From the learning results of the U-Net segmentation model, it can be observed that the Resnet, Seresnet, and Mobilenet backbone networks achieved the same accuracy value in epoch 15, i.e. 0.9999. Regarding the loss function, the learning process of the Seresnet backbone achieved the best value in the last episode, although in the validation process in the same epoch it was the worst among the compared methods. The Seresnet represents the "state-of-the-art" backbone in this comparison and can be identified as the backbone with the fastest learning process. The Resnet

Tab. 1: CNN parameters

Parameters of trained CNN	
Number of epochs	15
Number of classes	3
Optimizer	Adam
Learning rate (α)	0.001
Steps per train epoch	number of train images / 4
Steps per validation epoch	number of test images / 4
Loss function	Sparse categorical crossentropy
Activation function	ReLu, Softmax
Backbones	Mobilenet, Resnet, SeResnet, Xception
Pretrained weights	Imagenet dataset

backbone, which has a slower learning process but does not tend to diverge, retains its constancy of comparison. Represented in Fig. 37.

However, the results of the ASPOCRNet learning process are more different. In Fig. 38, the Mobilenet and Xception backbones achieve the same accuracy value of 0.9996 in episode 15, while the Resnet backbone had a significantly lower value. In the same episode, Xception achieves the best value for the learning loss function and also for the validation loss function among the methods already compared. As in the previous case, the Xception backbone in this context represents a "state-of-the-art" backbone and can be considered the best backbone of the compared backbones in terms of the ASPOCRNet segmentation model.

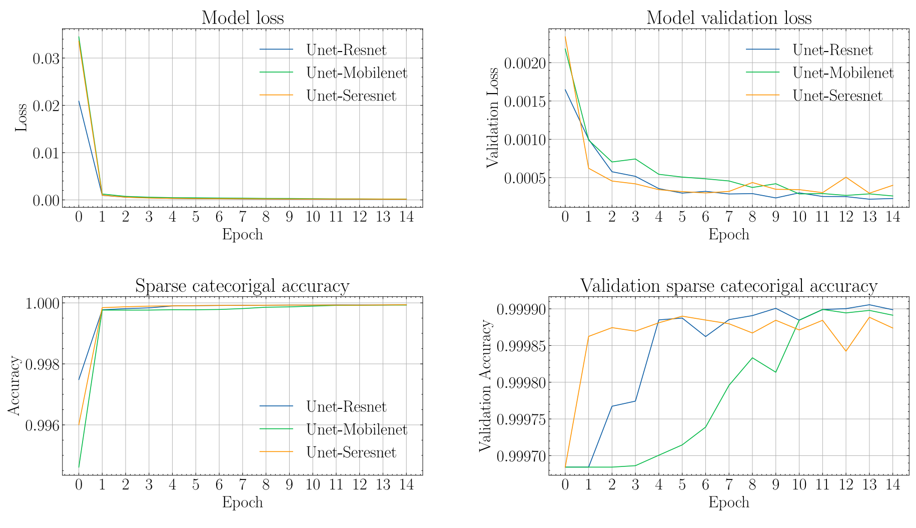


Fig. 37: U-Net results

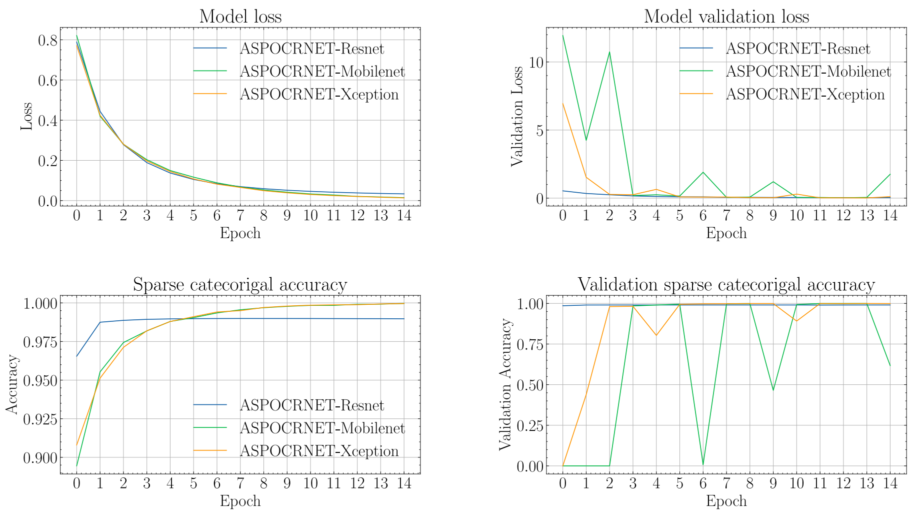


Fig. 38: ASPOCRNet results

In this work, it was necessary to create a function to detect nostrils, from among several possible methods that solve image segmentation an artificial neural network was used more precisely a convolutional neural network. In the solution, two segmentation models were compared, the older U-Net (2015) and the newer ASPOCRNet (2021) model. The result of this comparison is that to solve the defined problem it is not necessary to apply the complex segmentation model represented by ASPOCRNet, which finds its use mainly for solving robotic segmentation scenes with a larger number of classes. Whereas U-Net has demonstrated good learning properties for this relatively simple segmentation task. An important difference compared to U-Net is that the ASPOCRNet model offers the advantage of significantly smaller size due to its architecture. In comparison, ASPOCRNet was half the size of U-Net.

To be able to use the trained model output comprehensively, the *ONNX (Open Neural Network Exchange)* tool was used to convert the *TensorFlow* .pb model format to the general .onnx format. This format facilitates the use of the model in various frameworks and libraries such as *OpenCV*, *PyTorch*, or even *Wolfram*. Since some functions of the web application already rely on the *OpenCV* library, e.g. for face detection, it was afterwards used to load and use the trained model for nostril detection Fig. 39. The size of the trained model itself also plays an important role, when exporting to .onnx the size was reduced by about 33%.

In the context of the application, the centre of the nostril is then calculated from the classified pixels. The relationship for the calculation can be defined as follows:

$$c_{x,y} = \frac{\sum_{n=1}^{N} p_{x,y}}{l_{x,y}} \tag{8}$$

Where $c_{x,y}$ represent centre on the x,y-axis, $p_{x,y}$ pixels of interest on the x,y-axis and $l_{x,y}$ is number of pixels of interest on the x,y-axis.

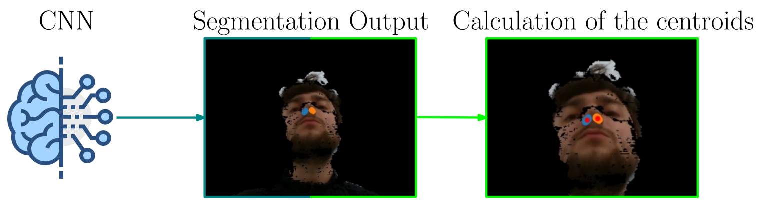


Fig. 39: Post-process: Segmentation image - Centroid of nostrils image

5.3.3 Face Reconstruction

The final part of this process is the subsequent reconstruction of the 3D point cloud from the colour image aligned with the depth image and the depth image itself. Once the centre of the nostril is detected, see section 5.3.2, this point is transferred to the original colour image. The process of creating the RGBD image is then performed. The conversion function of the *open3D* library [67] creates an RGBD image from the pair of the colour image and the depth image. Where the depth image is stored in a data type representing the depth value in metres.

The *Open3D* library also provides a function for the subsequent calculation of point clouds by conversion of the RGBD image. To perform the conversion, it must be specified the 3D camera parameters. These parameters includes depth scale d_s, the focal length (f_x, f_y) and the optical centre (c_x, c_y). The parameters of the RGBD image are the resolution (a, b) and the depth d [39]. The calculation of the point cloud is then performed based on the equations:

$$z = \frac{d}{d_s} \tag{9}$$

$$x = \frac{(a - c_x) \cdot z}{f_x} \tag{10}$$

$$y = \frac{(b - c_y) \cdot z}{f_y} \tag{11}$$

The result is a point cloud in the *open3D* PointCloud data type which is used for subsequent rendering in the web browser Fig. 40 and also as a renderer in ROS visualization tool *Rviz*. A Point cloud result of 3D face reconstruction plays a crucial role in calculating the trajectory of the robot's movement towards the detected centre of the patient's nostril. Another approach to perform the face reconstruction is to save the scene directly as a point cloud in .ply or .pcd format, but there is the problem of transferring the detected nostril centre into 3D space arises.

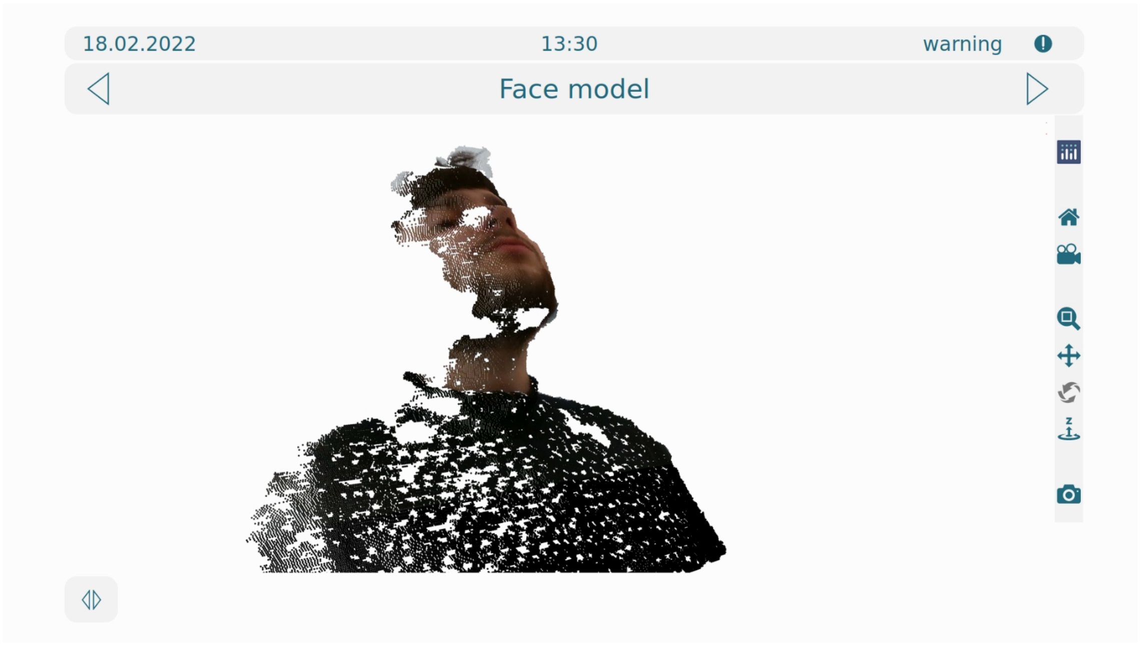


Fig. 40: Robo Medicinae I - Face model

5.4 Human-Machine Interface

Also one of the important parts that needed to be addressed and mentioned for this application is the HMI. For this part, cooperation with Kateřina Monsportová, who created the design in *Adobe Illustrator*, was organized. My colleague also created the logo and vector graphics for the UR3 robot. The aim was to create a nice graphic design, which is nevertheless modular for different additional components and also to keep it simple.

The technical implementation was done by exporting the CSS code from *Adobe Illustrator* and then modifying and integrating it into the application. However, for rapid development and integration, a more advanced script pre-processor, such as *Sass* or *Less*, which can be used for larger projects, was not used. When implementing the graphic design of HMI from my colleague, it was necessary to solve the automatic recalculation of the resolution for rendering the content in the operator's web browser. To speed up the integration, the design solution was set to the original design resolution of 1280×960 pixels and automatically recalculated to a resolution of 1920×1080 pixels. JavaScript libraries *jQuery*, *Plotly*, *Anime*, and others were used for animations, charts, graphics, and other interesting elements of the app. From this design, a user-friendly environment with a medical touch was created.

Chapter 6

Simulation and Tests

An essential tool for robotics developers is a simulation environment in which they can test their programs, algorithms, and robot designs and perform regression tests. In this way, dangerous situations can be avoided, both for the robot itself and humans. To avoid such situations, developers use simulation environments such as *Gazebo*, *CoppeliaSim*, or *Nvidia Isaac*, for example, in which they try to model the real environment and test the simulation as close as possible.

Another very useful tool for developers is test libraries. With these libraries, developers can create large-scale projects without compromising the functional integrity of the application or system by modifying or adding functions. There are several libraries within Python such as *Unittest*, *Pytest*, *Nose2* and etc.

6.1 Test Application

Due to the large scale of the whole experimental robotic application, code tests lead to maintainability, quality, and in a sense also security. The tests were mainly performed on the Python Flask, as it is the central unit. For these tests, the default Python *Unittest* library was used. With this library, it was possible to test the basic routes of the Flask and possibly detect weaknesses, bad code, or unhandled exceptions.

The application was also tested with one of the easiest password cracking attacks, the brute force attack. In cryptography, this type of attack means that the attacker tries to crack the login by using a large number of passwords or passphrases attempts to obtain the correct combination. This method can be effective if the password consists of a small number of characters in the order of three. If the password has at least five characters, where alphanumeric characters alternate with special characters, this method is highly ineffective. For example, the limited number of possible requests to process or the CSRF protection can be a struggle against this method.

To perform this test, CSRF had to be disabled in the Flask configuration. The test searched for the password "aaaa", and only upper and lower case letters were generated during the test. The password was in position 140 610 in the order of the characters tested. An attempt to send with a server response took approximately 0.01 s. The hacking script was programmed in Python 3.

6.2 Laboratory Task Process

The main task and function of this experimental robotic platform is the collection of samples by swabbing from the nostril vestibule. As mentioned in section 5.3, it was necessary to program a comprehensive facial scanning function that would take care of the detection of the centre of the nostril and the actual 3D reconstruction into a point cloud form. This was done within the back-end server and front-end client integration. However, within the overall application, this is preceded by several tasks from the operator's perspective. Assuming the operator makes the device operational, it can be defined in a few basic steps:

1. The patient is positioned in a defined examination area.
2. The operator searches the database for the patient (surname, PIN or QR) and selects it.
3. The operator activates the faceID function and patient identification is performed.
4. A face detection process is then performed to ensure that the robotic arm rotates in the correct position.
5. If everything is validated, the process of scanning, nostril detection and 3D image reconstruction follows.
6. In the final stage, nasal sampling is carried out.

As part of the solution to this task, a working prototype was created in which, however, it is necessary to perform these processes semi-automatically due to the safety of real-world testing.

6.2.1 Patient Space

The application places great emphasis on patient freedom. However, it is necessary to pre-allocate space for the patient to be in due to the limited perception of space. Limits come with the restricted range of the robotic arm and also the limited view range of the camera. A safe and sensed area for the patient must be marked. The application also depends on the patient not moving, if the patient moves the application does not have a dynamic replanning process. The safety feature of cobot or a force-torque sensor stops the collaborative robot in case of a collision with the patient.

6.2.2 Hand-Eye Calibration

To perform an accurate movement to the identified nostril from the 3D model reconstructed in the application, it is necessary to perform a so-called hand-eye calibration. Alternatively, if the application solution used a camera statically positioned outside the robot, and eye-to-hand calibration would have to be performed. The differences represented in Fig. 41. However, within the solution, due to flexibility and limited space, the first option was used.

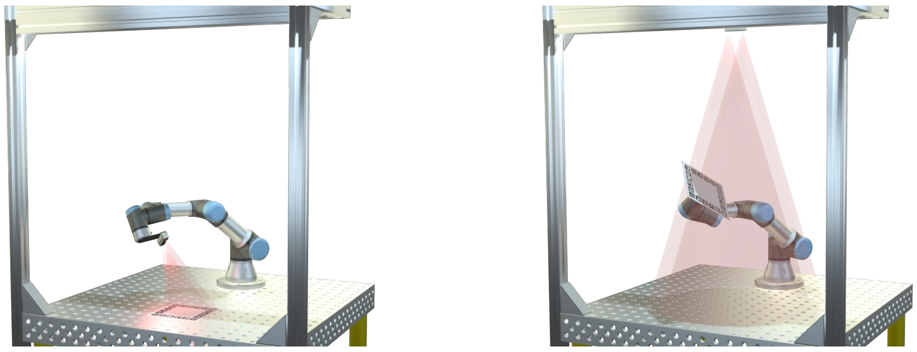

(a) Hand-eye calibration (b) Eye-to-hand calibration

Fig. 41: Hand and eye calibration methods to determine the geometric relationship between the robot and the 3D camera.

The hand-eye calibration process can be defined as finding the relative position and orientation between the fixed camera mounted on the end-effector and the last joint of the robotic gripper [65]. One or more cameras can be calibrated by using a calibration pattern (Fig. 42). Essentially, this calibration aims to transform the detected x,y coordinates from the vision frame to the robot coordinate frame. Hand-eye-calibration can be generally mathematically described as:

$$\mathbf{AX} = \mathbf{XB} \tag{12}$$

- **X** unknown hand-eye transformation
- **A** robot's joint data
- **B** corresponding camera movement

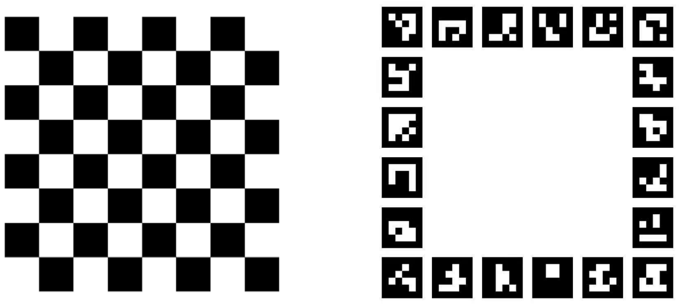

(a) Chessboard (b) ArUco board

Fig. 42: Calibration patterns [34]

The ROS *Robotic Hand-eye Calibration Workspace* package was selected for the hand-eye calibration solution [34]. This is a package that was developed primarily for the ROS Melodic Morenia distribution. The ArUco board was chosen as the calibration pattern. The chessboard may also be taken into account. The first phase consisted of preparations, so the dependencies had to be installed on the control system running ROS, the Gripper had to be built and the cobot and camera had to be connected. The next part consisted of running the program itself, which recorded the individual positions of the collaborative robot with the camera capturing the statically placed calibration pattern. Once a sufficient number of these positions have been recorded, the result calculation was performed. The output was recorded in a .launch file, where the result position in the x,y,z axis, and the quaternions were written. This result was used to calculate the motion of the cobot.

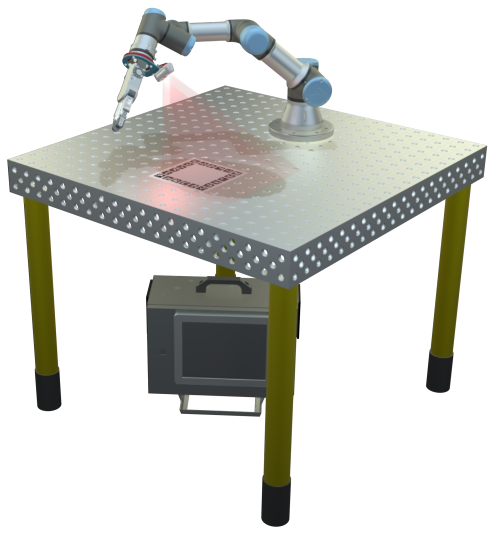


Fig. 43: Robo Medicinae I - Hand-eye calibration process

6.2.3 Robot Motion

As already mentioned in section 5.2.2, ROS was chosen to solve the motion and also the simulation. This meta operating system offers the visualization tool *Rviz*, but for full-motion simulation, a simulation environment is required. The simulation environment *Gazebo*, which is the default simulation environment of ROS, was used in the solution of the given task. The *Gazebo* has physics engines such as ODE (Open Dynamics Engine), Bullet, Simbod, or DART (Dynamic Animation and Robotics Toolkit). An alternative solution could be *Webots*, for example. Like *Gazebo*, *Webots* uses the ODE physics engine, but with its own modifications.

However, before the actual simulation process could be carried out, it was necessary to create a simulation scene (Fig. 44). It was assembled from a number of models to create the most faithful-looking scene for real-world testing. Therefore, the original ROS scripts were modified to involve only UR3 for the *Gazebo* and *MoveIT* simulation environments. However, in the case of the framework, it should be noted that, as part of the creation of the *robo_moveit* package using the *MoveIT Wizard*, the algorithms calculate the collision scene from the given models. Therefore, the more complicated and robust the models imported into the *MoveIT* scripts, the slower the trajectory calculation will be. Therefore, the modified simplified models were imported for *MoveIT*, while the original models exported from *Autodesk Inventor* were kept for *Gazebo*.

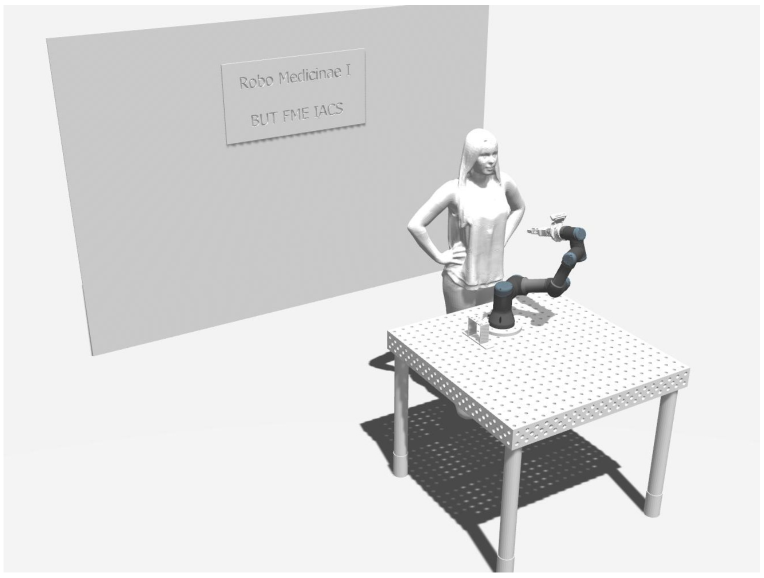


Fig. 44: Robo Medicinae I - Gazebo simulation scene

As part of the solution to the process of collecting samples from the nostrils, it was also necessary to integrate the movement of the UR3 itself. This cobot movement process is divided into three basic parts:

1. Initiating movement
2. Alignment movement
3. Movement for sampling

The initialization motion is initiated after the operator connects the UR3 to the application. This process represents the execution of the ROS control script from the *Universal_Robots_ROS_Driver* and the associated *MoveIT* script. Once these scripts are executed, the process of picking the swab stick with thread from the rack follows. This movement also contains the control of the RG2 gripper in the open and close direction. This is followed by the movement to the initialization position. Once the initialization position is reached, the alignment movement follows, which is coupled to the operator interaction. This is a process that is part of face detection, where it is necessary to correctly align the cobot so that the face is correctly captured from the camera's perspective. The core of the movement itself involves rotation in eulerian coordinates on the y-axis (pitch).

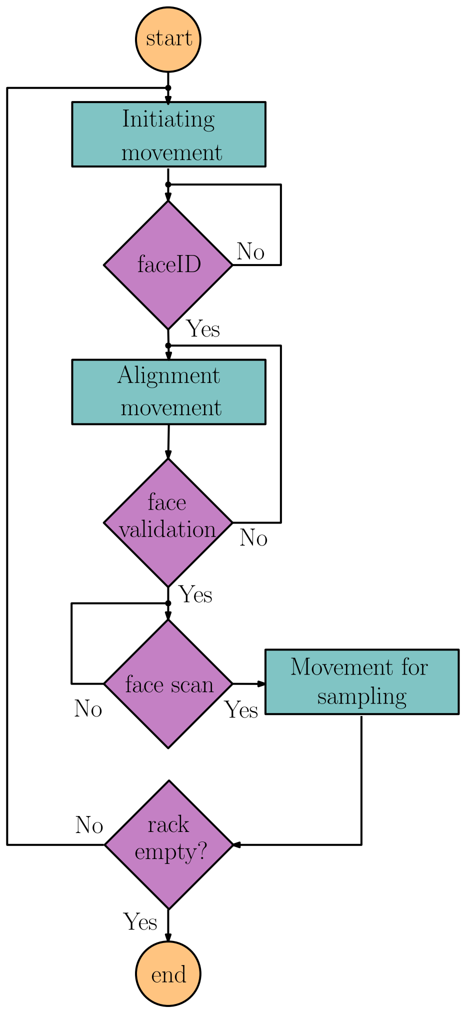


Fig. 45: Floatchart of cobot movement process

Last but not least in this process is the movement for collecting samples. This movement of the cobot is the most robust, as it represents a series of actions. The process is activated after the operator confirms the correct scan of the face and initiates the process. This complex process must include the alignment action to the colour camera to achieve the correct position, followed by a linear movement towards the centre of the nostril, and then a circular movement is performed. The next part is a linear movement back to the initialization position. If everything is done correctly, the collaborative robot moves back to the rack and places the swab stick. The process is illustrated in Fig. 46. After this last movement, the whole task can be performed again for a new patient in which the cobot will pick the swab stick from the other stilts of the rack.

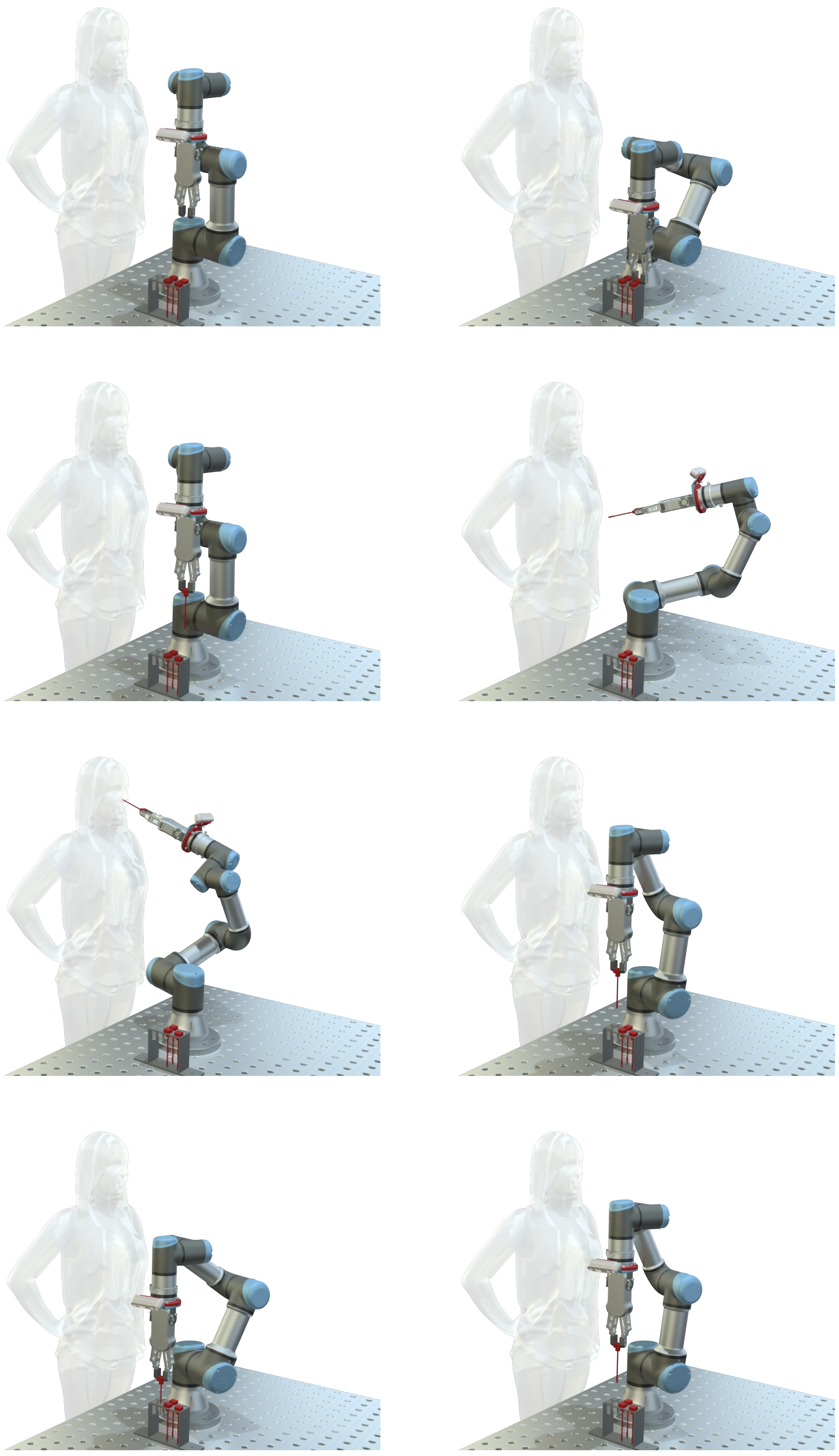


Fig. 46: Robo Medicinae I - Visualisation process of collecting samples from nostril

6.2.4 Safety

As this is an unprecedented process that can quickly lead to personal injury during real-world testing, it was necessary to ensure safety. Therefore, the HEX-E sensor, discussed in section 5.1.2, was also integrated. From the application point of view, the data is acquired via ROS, where a Python script is programmed to watch the deviation of the applied force. Thus, if a value higher than the lower limit is reached, the cobot stops immediately. For example, simply touching the gripper can cause the collaborative robot to stop. One of the challenges in addressing this safety is to stop so that the *Universal_Robots_ROS_Driver* themselves or even the cobot are not overridden.

As mentioned, the HEX-E sensor itself can only detect this force from its centre to the end of the gripper. However, when using the UR3 collaborative robot, patient safety is also ensured by the safety features of the cobot itself. This is the fact that the cobot will not cause significant injury to the patient. An extreme example of this is if the patient moves from his scanning position and the collaborative robot is already in motion, he then intuitively tries to stop the cobot with his hands, which activates the safety features of the UR3 and the cobot shuts down in emergency mode and stops. In this case, the operator must respond to determine if the patient is fine and restart the cobot from the UR Polyscope control panel.

6.3 Application Integration

As part of the solution, it was necessary to resolve the portability of the entire control system so that it could be run both on a local computer and, if necessary, by extending the entire system on a dedicated server. For this purpose, the use of *Docker* is proposed. *Docker* is an open-source tool that aims to provide a unified interface for isolating containerised applications in *macOS*, *Linux* and *Windows* environments [13]. In this sense, containers are a standard unit of software. This unit includes the code for all its dependencies so that the application runs reliably in different operating systems. In this area, the most prominent competitor is the *Kubernetes*.

To solve the problem of an experimental robotic application, the *Docker Compose* (Docker extension) tool was used, which allows the entire application to be split into several containers and then linked together (Fig. 47). One of the containers is service marked nginx. Nginx uses the *NGINX* software of the same name to provide the application with web server services such as reverse proxy, caching, or load balancing, among others [14]. To complete the picture, the *uWSGI* web application server is used, which provides a standard web server interface and mediates requests

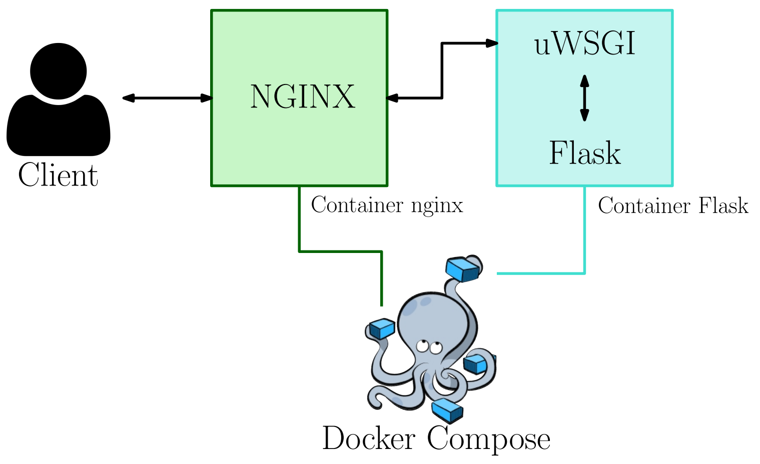


Fig. 47: Robo Medicinae I - Docker Compose structure

to Flask [63]. Very often this framework is used in conjunction with web servers such as *Nginx* or *Cherokee*.

In the final implementation, ROS was not integrated as a container, mainly due to development, simulation, and testing. It is the containerization of the ROS control program that provides an extension of the work. In this case, it is already a modification for live deployment. As for the solution, this is a very simple task given the availability of already solved ROS containers. Another way to deploy in operation can be to export the trajectories defined by ROS and use, for example, the *python-urx* or *ur_rtde* library to control the UR3 collaborative robot. The most complicated way is through the export of ROS programs, where the robot is controlled by URSCript commands sent through Socket communication. However, due to the emphasis on modularity, the resulting deployment application can be modified in any way.

The entire application Robo Medicinae I is stored in a *Github* repositories. The *Github* contains a header repository and four sub-repositories. Thus, there is the complete software, CAD models, and documentation. All Python scripts conform to Black's coding style.

6.4 Real-World Test and Result

To verify the functionality of the entire robotic platform, wiring and real-world testing was carried out. The devices used for the tests were:

- Collaborative robot Universal Robots UR3
- 3D camera Intel RealSense D435i
- Force-Torque sensor OnRobot HEX-E
- 2-finger gripper OnRobot RG2

- Single-board computer Nvidia Jetson Xavier
- Touch screen monitor pc HP Slate 21 Pro

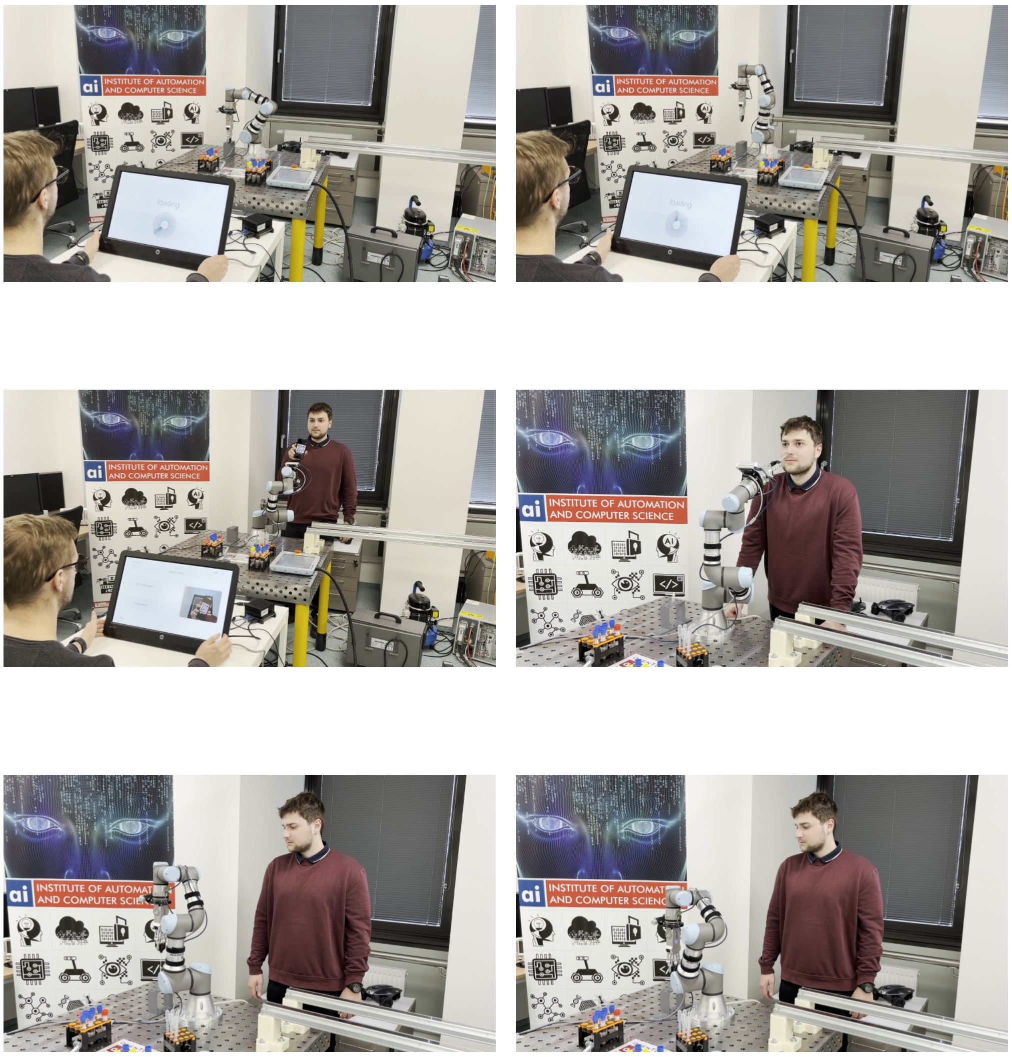


Fig. 48: Robo Medicinae I - Real-world process of collecting samples from nostril

The control unit was represented by an Nvidia Jetson Xavier single board computer in development kit configuration. The use of this single-board computer, which is based on the ARM processor architecture, can have its drawbacks in the compatibility of some libraries that have to be built and installed from the source. This incompatibility also poses a problem when using docker. However, during testing in the lab, the solution was run on Flask's development server. The drawback then arose with the need to connect the control unit to the touch screen monitor via a cable. Once all devices were connected to the central unit, it was possible to test the systems, the communication with ROS, and the functions of the robotic platform itself. However, the main function, i.e. collecting samples from the nasal vestibule, has to be prioritized. These tests of the main function were carried out in the Robotics and Cybernetics Laboratory at IACS, and the results are shown in the Tab. 2.

In tests, high accuracy was achieved in detecting nostrils. Poor or no recognition was mainly due to poor head tilt, where the nostrils were covered by the tip of the nose in the captured image. Subsequent motion simulation then achieved the same accuracy. The lower success rate in real-world tests is due to several external influences. These influences include spontaneous movement of the patient, and inaccurate calibration of the camera, the robot or the hand-eye. However, the spontaneous patient movement has the greatest influence on inaccuracy, as it has been observed when only a deviation of a few millimetres has occurred. The image output from lab shown in Fig. 48.

Tab. 2: Table of laboratory test results

Laboratory test results	
Number of attempts: 75	
Number of tested persons: 5	
Task	**Succes rate**
Nostrils detection	96.00%
Motion simulation	$93.3\overline{3}\%$
Real-world movement	$38.3\overline{3}\%$

Chapter 7

Discussion and Future Work

The main objective of the thesis was to create an experimental robotic platform with verification of the functionality of the created solution by simulation and on a real robot. The work itself was a challenge in the form of a combination of several technologies. From the design of the gripper itself to the entire topology of the software. The interconnection of software and hardware also played a crucial role. The development of this application took many months with frequent changes to achieve a result of a functional prototype of the robotic platform.

In the first development phase, the first thing to do was to choose a technology that was able to render the HMI and control the processes at the same time. Portability and compatibility with different devices also played an important role. The ideas led to using B&R's programmable logic controller (PLC) as the main control device with MappView technology. This turned out to be a bad idea during development, due to poor performance. Another interesting idea would be to use the powerful Siemens SIMATIC IPC System. Siemens SIMATIC IPCs for AI offer incredible performance based on special devices from both Nvidia and Google. Therefore, the solution was based on Python Flask and a possible Docker extension that can involve any central device based on architecture x86. Worse compatibility arose with the ARM architecture. In this phase, the second step was to properly use the Intel RealSense D435i camera and integrate it into the system. In this regard, it was also necessary to configure the camera correctly and to use the tools and libraries to process the camera data. The processing and configuration of the camera were done to maximize the accuracy of the human head capture. In this second part of the first phase, there was already a simple and very poor HMI programmed. The challenge was to integrate and control the camera so that the results could be displayed in a web browser. Gradually, functions such as faceID, face detection, or the acquisition of data from the 3D camera or the UR3 cobot were integrated. In this first part, it also had to be decided whether to use the ROS or to do without it. A crucial role in the choice was then played by the additional ROS tools that would have to be additionally programmed as part of the work. This would lead to a further extension of the development.

In the next phase, integration and database work followed, with the involvement of a nice HMI by my colleague Kateřina Monsportová. In addition, the first simulation was created in the *Webots* simulation environment, and this environment was later replaced by *Gazebo*. A simple UR3 control panel was also created together with a digital twin display. Gradually, more functions and animations for the user were added. In addition, an own gripper was designed. This phase was followed by the first real-world tests to check if the topology of the software worked correctly and if the designed gripper would fit and perform its functionality. This was followed by a series of tests to make the UR3 collaborative robot operational and connected, and finally, the hand-eye calibration was carried out. Another important part of this phase was the commissioning and control of the RG2 gripper and the HEX-E sensor.

The third phase was marked by the training of the convolutional neural networks and the implications for the exploration process of the application. Modifications were also made and final features were added to the application, designing the rack and the thread, and making the robot move. Linking the application to the robot's movements, programming tests, and performing a simple hacking attack. Finally, the resulting system was installed on an Nvidia Jetson Xavier controller.

In the field of learning convolutional neural networks, data collection, labelling, pre-processing, training, and post-processing of the program were also necessary. Two segmentation models, U-Net and ASPOCRNet, were compared. Each model was trained on similar backbones, where one of them represented the state of the art. As a result, in general, U-Net performed much better in the learning process. I can point out that this was not a very complex segmentation scene where ASPOCRNet would have been more successful. However, it is necessary to highlight the fact that ASPOCRNet, due to its architecture, offers a considerably smaller trained model.

The practical part of the master's thesis was completed by testing the robotic platform designed in the IACS laboratory. The primary function tested was to collect samples from the nasal vestibule. The result is shown in Tab. 1. The main cause of unsuccessful movement attempts in the nostril is mainly due to external influence, when the patient is not mechanically fixed and therefore able to move spontaneously while breathing. Millimetric deviations also play a role in this process. In the laboratory, it was observed that the variance of the deviations was then in the order of about 5 mm. Other factors influencing the failed movement are inaccurate calibration of the camera, the robot, and, finally, the hand-eye.

Compared to existing solutions (section 1.2), this application differs in the primary function of swabbing samples mainly because it does not go directly into the patient's nasopharynx, but only into the nasal vestibule, leading to increased safety and speed. For eventual deployment in the field, it is envisaged to have antigen sticks and tests that can diagnose only from samples from the nasal vestibule. This application also offers a user-friendly design with simple but effective controls. The main emphasis in this work is on openness, customization, and extensibility. However, the open space provided to the patient without any mechanical fixation brings its disadvantage in the form of the external influence of the spontaneous movement. This leads to a considerable degree of failure in the actual sample collection process. However, this effect can be overcome with modifications.

An example of a modification could be the use of a two-armed collaborative robot, YuMi. Thanks to its ingenious design, a camera can be statically mounted on it, resulting in the possibility of a fairly accurate scan of the patient, addressing whether or not and to what extent he or she has moved. These calculations would be fed into further calculations for dynamic trajectory planning, resulting in a much higher success rate. It should also be noted that this collaborative robot not only has two arms but also integrates a controller at the bottom, so it can be involved in a mobile platform, for example. This can lead to completely expanded application possibilities, such as analyzing, pipetting, or handling samples, thanks to its two arms. This would speed up and automate the whole process of antigen diagnosis. Improvements can also be made, for example, to mark samples with a barcode and automate the transport. Also worth mentioning is the idea of drug administration or room disinfection, added to the nasal chamber sample collection process, and much more.

Within the complexity of the project, there are many other possibilities for expansion, from the software part to possible hardware modifications or replacement of the cobot itself. Another substantiated extension is to improve the segmentation model. In the resolution of the work, when an emergency condition with limited assistance was in force, a small dataset of people was taken. To obtain a better and more accurate segmentation, it would be preferable to acquire a larger dataset and perform transfer learning, alternatively, if the dataset is large enough, there is the possibility to perform the learning completely anew with a fine-tuning technique. Also worth mentioning is the use of possibly a different segmentation technique or a convolutional neural network segmentation model.

Other desired improvements include the calculation of the relative tilt for the movement towards the nostril. This proposal is based on the estimation normals of the point cloud by using the k-d tree search parameters for the hybrid k-nearest neighbours algorithm compute. With this calculation, the most optimal inclination for the movement of the nostrils can be calculated. On the practical side, this calculation is already implicit using the *open3D* library but has not been completed until the secure testing phase.

Chapter 8

Conclusion

An experimental robotic platform with the main function of collecting samples from the nasal vestibule has been presented. In this context, it concerns the design, implementation, and testing of a robotic platform that is a complete integration of software and hardware with a range of functions. The work aimed to create a web-based application that is under the control of an operator. This application has involved a central unit, which can represent, for example, a powerful single-board computer while controlling all devices together with the collaborative robot. The operator then connects to this central unit via for example a tablet. A complete design of the software topology was carried out, where an integral part was programmed and a convolutional neural network model was learned to solve the detection of the nostril. The emphasis on patient safety cannot be overlooked. The actual hardware part was also built, commissioned, and tested. The result is a working prototype of an experimental robotic platform.

One of the main shortcomings of this robotic platform was that the external influence of the patient's spontaneous movement was not taken into account, given the emphasis on open space design. However, this is an external influence that can be addressed by improving the design and applying advanced methods. These methods include both patient monitoring for displacement detection and subsequent application for dynamic trajectory planning. This is the most important improvement that would contribute to a rapid increase in the success rate of sample collection.

Chapter 9

Bibliography

[1] ABB. *ABB demonstrates concept of mobile laboratory robot for Hospital of the Future* [online]. 2019 [cit. 2022-02-22]. Available at: `https://new.abb.com/news/detail/37301/abb_demonstrates_concept_of_mobile_laboratory_robot_for_hospital_of_the_future`.

[2] ABB. *GoFa CRB 15000* [online catalogue sheet]. 2021 [cit. 2021-09-21]. Available at: `https://assets.ctfassets.net/gt89rl895hgs/1MBowsjHDvAEykEwKKBWwl/0fd8f6c1512862195b23b1613db0db16/GoFa_CRB15000-datasheet_digital_20210408_ms.pdf`.

[3] ABB. *Právě představené inovativní koboty ABB cílí na méně zkušené uživatele* [online]. ABB s.r.o., 2021 [cit. 2022-03-20]. Available at: `https://www.strojirenstvi.cz/prave-predstavene-inovativni-koboty-abb-cili-na-mene-zkusene-uzivatele`.

[4] ABB. *SWIFTI CRB 1100* [online catalogue sheet]. 2021 [cit. 2021-09-21]. Available at: `https://assets.ctfassets.net/gt89rl895hgs/7lxW2lwo38EpADo8WYr0ja/753a223995a8cbd611ee78e18763af5e/SWIFTI_CRB1100_digital_20210331_ms.pdf`.

[5] ABB. *YuMi IRB 14000* [online catalogue sheet]. 2021 [cit. 2021-09-21]. Available at: `https://search.abb.com/library/Download.aspx?DocumentID=9AKK106354A3254&LanguageCode=en&DocumentPartId=&Action=Launch`.

[6] Alom, M. Z., Yakopcic, C., Hasan, M., Taha, T. and Asari, V. Recurrent residual U-Net for medical image segmentation. *Journal of Medical Imaging*. march 2019, vol. 6, [cit. 2022-03-20]. DOI: 10.1117/1.JMI.6.1.014006.

[7] Bi, Z., Luo, C., Miao, Z., Zhang, B., Zhang, W. et al. Safety assurance mechanisms of collaborative robotic systems in manufacturing. *Robotics and Computer-Integrated Manufacturing*. 2021, vol. 67, p. 102022, [cit. 2022-02-22]. DOI: https://doi.org/10.1016/j.rcim.2020.102022. ISSN 0736-5845. Available at: `https://www.sciencedirect.com/science/article/pii/S0736584520302337`.

[8] BOYLE, A. *XR experts see health care as the killer app for VR, AR, MR . . . or whatever you call it* [online]. GeekWire,

2020 [cit. 2022-03-20]. Available at: https://www.geekwire.com/2020/xr-experts-see-health-care-killer-app-vr-ar-mr-whatever-call/.

[9] Chen, L., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A. L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *CoRR*. 2016, abs/1606.00915, [cit. 2022-03-14]. Available at: http://arxiv.org/abs/1606.00915.

[10] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F. and Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. arXiv. 2018, [cit. 2022-04-23]. DOI: 10.48550/ARXIV.1802.02611. Available at: https://arxiv.org/abs/1802.02611.

[11] Cristina, S. *The Transformer Model* [online]. 2021 [cit. 2022-03-14]. Available at: https://machinelearningmastery.com/the-transformer-model/.

[12] Crowe, S. *Danish startup develops throat swabbing robot for COVID-19 testing* [online]. 2020 [cit. 2022-03-20]. Available at: https://www.therobotreport.com/danish-startup-develops-throat-swabbing-robot-for-covid-19-testing/.

[13] Docker. *What is a Container?* [online]. 2022 [cit. 2022-03-23]. Available at: https://www.docker.com/resources/what-container/.

[14] NGINX. *What is NGINX?* [online]. 2022 [cit. 2022-03-23]. Available at: https://uwsgi-docs.readthedocs.io/en/latest/.

[15] Dorodnicov, S. *The basics of stereo depth vision* [online]. 2018 [cit. 2022-02-22]. Available at: https://www.intelrealsense.com/stereo-depth-vision-basics.

[16] FANUC. *Kamerové funkce pro roboty* [online]. [cit. 2022-02-22]. Available at: https://www.fanuc.eu/cz/cs/roboty/p%C5%99%C3%ADslu%C5%A1enstv%C3%AD/vid%C4%9Bn%C3%AD.

[17] Flask. *Foreword* [online]. 2010 [cit. 2022-03-01]. Available at: https://flask.palletsprojects.com/en/2.0.x/foreword/#what-does-micro-mean.

[18] Franka Emika. *PANDA - DATASHEET* [online catalogue sheet]. 2019 [cit. 2021-03-04]. Available at: https://wiredworkers.io/wp-content/uploads/2019/12/Panda_FrankaEmika_ENG.pdf.

[19] Freeman, W. D., Sanghavi, D. K., Sarab, M. S., Kindred, M. S., Dieck, E. M. et al. Robotics in Simulated COVID-19 Patient Room for Health Care Worker Effector Tasks: Preliminary, Feasibility Experiments. *Mayo Clinic Proceedings: Innovations, Quality and Outcomes*. 2021, vol. 5, no. 1, p. 161–170,

[cit. 2021-03-09]. DOI: https://doi.org/10.1016/j.mayocpiqo.2020.12.005. ISSN 2542-4548. Available at: https://www.sciencedirect.com/science/article/pii/S2542454820302599.

[20] Gajdošíková, V. *Vědci z brněnského VUT navrhli robotické pracoviště, pomůže laborantům zpracovat infekční vzorky* [online audio]. 2020 [cit. 2022-03-20]. Available at: https://brno.rozhlas.cz/cro_soundmanager/files/8205399/field_main_audio.

[21] Goodfellow, I., Bengio, Y. and Courville, A. *Deep Learning* [online]. MIT Press, 2016 [cit. 2022-04-01]. Available at: http://www.deeplearningbook.org.

[22] Hu, Y., Li, J., Chen, Y., Wang, Q., Chi, C. et al. Design and Control of a Highly Redundant Rigid-flexible Coupling Robot to Assist the COVID-19 Oropharyngeal-Swab Sampling. *IEEE Robotics and Automation Letters*. 2022, vol. 7, no. 2, p. 1856–1863, [cit. 2022-03-20]. DOI: 10.1109/LRA.2021.3062336.

[23] Hwang, E. J., Park, S., Jin, K.-N., Kim, J. I., Choi, S. Y. et al. Development and Validation of a Deep Learning–Based Automated Detection Algorithm for Major Thoracic Diseases on Chest Radiographs. *JAMA Network Open*. march 2019, vol. 2, no. 3, p. e191095–e191095, [cit. 2022-03-03]. DOI: 10.1001/jamanetworkopen.2019.1095. ISSN 2574-3805. Available at: https://doi.org/10.1001/jamanetworkopen.2019.1095.

[24] Intel. *Depth Camera D435i* [online]. [cit. 2022-03-03]. Available at: https://www.intelrealsense.com/depth-camera-d435i/.

[25] Intel. *Beginner's guide to depth* [online]. 2019 [cit. 2022-03-04]. Available at: https://www.intelrealsense.com/beginners-guide-to-depth/.

[26] Intel. *Intel RealSense Product Family D400 Series* [online catalogue sheet]. 2020 [cit. 2022-02-22]. Available at: https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf.

[27] Intel. *LiDAR Camera L515* [offline catalogue sheet]. 2021 [cit. 2022-03-03]. Available at: https://www.intelrealsense.com/lidar-camera-l515.

[28] IntelRealSense. *Intel RealSense SDK* [online]. GitHub, 2022 [cit. 2022-02-22]. Available at: https://github.com/IntelRealSense/librealsense.

[29] Intuitive. *Da Vinci X User manual* [offline catalogue sheet]. Intuitive Surgical [cit. 2021-03-09]. Available at: https://manuals.intuitivesurgical.com/home.

[30] Intuitive. *Da Vinci Surgical Systems* [online]. 2022 [cit. 2022-03-20]. Available at: `https://www.intuitive.com/en-us/products-and-services/da-vinci/systems`.

[31] Jung, W.-J., Kwak, K.-S. and Lim, S.-C. Vision-Based Suture Tensile Force Estimation in Robotic Surgery. *Sensors*. 2021, vol. 21, no. 1, [cit. 2022-03-03]. DOI: 10.3390/s21010110. ISSN 1424-8220. Available at: `https://www.mdpi.com/1424-8220/21/1/110`.

[32] KUKA. *Medical Robotics, LBR Med* [offline catalogue sheet]. 2017 [cit. 2022-02-22]. Available at: `https://www.kuka.com/en-gb/industries/health-care/kuka-medical-robotics/lbr-med`.

[33] Leroux, T., Ieng, S.-H. and Benosman, R. Event-Based Structured Light for Depth Reconstruction using Frequency Tagged Light Patterns. november 2018, [cit. 2022-02-22]. Available at: `https://www.researchgate.net/publication/329234540_Event-Based_Structured_Light_for_Depth_Reconstruction_using_Frequency_Tagged_Light_Patterns`.

[34] lixiny. *Robotic Hand-eye Calibration Workspace* [online]. GitHub, 2022 [cit. 2022-02-22]. Available at: `https://github.com/lixiny/Handeye-Calibration-ROS`.

[35] Matoušek, R. *Neuronové sítě a hluboké učení* [lecture]. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky, CZ, 2021 [cit. 2022-03-20].

[36] Minkel, J. *First Complete Trans-Atlantic Robotic Surgery* [online]. 2001 [cit. 2022-02-22]. Available at: `https://www.scientificamerican.com/article/first-complete-trans-atla/`.

[37] Moore, E. J. Robotic surgery. Encyclopedia Britannica. 2018, [cit. 2021-03-09]. Available at: `https://www.britannica.com/science/robotic-surgery`.

[38] oneindustry. *Roboty jsou nepostradatelným partnerem člověka ve výrobě i službách* [online]. 2020 [cit. 2022-03-20]. Available at: `https://www.oneindustry.cz/automatizace-robotizace/roboty-jsou-nepostradatelnym-partnerem-cloveka-ve-vyrobe-i-sluzbach/`.

[39] Open3D. *Create point cloud from rgbd image* [online documentation]. 2019 [cit. 2022-03-01]. Available at: `http://www.open3d.org/docs/0.6.0/python_api/open3d.geometry.create_point_cloud_from_rgbd_image.html`.

[40] Optoforce. *RG2 Gripper Datasheet* [online catalogue sheet]. 2015 [cit. 2022-03-03]. Available at: https://www.universal-robots.com/media/1226143/rg2-datasheet-v14.pdf.

[41] Optoforce. *HEX-E* [online catalogue sheet]. 2017 [cit. 2022-03-03]. Available at: https://manualzz.com/doc/36650788/datasheet---optoforce.

[42] Pardesi, R. *Why the Difference is Crucial for Image Annotation* [online]. 2020 [cit. 2022-03-14]. Available at: https://www.linkedin.com/pulse/computer-vision-semantic-vs-instance-segmentation-ravi-pardesi/.

[43] Photoneo. *Bin Picking Studio The most versatile robotic intelligence software* [online]. 2022 [cit. 2022-04-28]. Available at: https://www.photoneo.com/bin-picking-studio/.

[44] Photoneo. *MotionCam-3D* [online]. 2022 [cit. 2022-02-22]. Available at: https://www.photoneo.com/motioncam-3d/.

[45] Photoneo. *PhoXi 3D Scanner M* [online]. 2022 [cit. 2022-02-22]. Available at: https://www.photoneo.com/products/phoxi-scan-m/.

[46] Picard, R. *Blueprints — Explore Flask 1.0 documentation* [online documentation]. 2014 [cit. 2022-03-01]. Available at: http://exploreflask.com/en/latest/blueprints.html.

[47] Pirolini, A. *The Rise of Advanced Robotics* [online]. AZO Robotics, 2014 [cit. 2022-03-20]. Available at: https://www.azorobotics.com/Article.aspx?ArticleID=192.

[48] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T. et al. ROS: an open-source Robot Operating System. *ICRA Workshop on Open Source Software.* march 2009, vol. 3, [cit. 2022-03-03]. Available at: https://www.researchgate.net/publication/233881999_ROS_an_open-source_Robot_Operating_System.

[49] Robotiq. *Adaptive grippers* [online catalogue sheet]. 2020 [cit. 2022-03-03]. Available at: https://blog.robotiq.com/hubfs/Product-sheets/Adaptive%20Grippers/Product-sheet-Adaptive-Grippers-EN.pdf.

[50] Robotiq. *FT 300-S FORCE TORQUE SENSOR* [online catalogue sheet]. 2021 [cit. 2022-03-03]. Available at: https://blog.robotiq.com/hubfs/Product-sheets/FT%20300/ur/Product_sheet-FT300S_FC_EN.pdf.

[51] Ronneberger, O., Fischer, P. and Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *CoRR*. 2015, abs/1505.04597, [cit. 2022-03-14]. Available at: `http://arxiv.org/abs/1505.04597`.

[52] Rumelhart, D. E., Hinton, G. E. and Williams, R. J. Learning representations by back-propagating errors. *Nature*. Oct 1986, vol. 323, no. 6088, p. 533–536, [cit. 2022-04-01]. DOI: 10.1038/323533a0. ISSN 1476-4687. Available at: `https://doi.org/10.1038/323533a0`.

[53] S., K. *Cross Site Request Forgery (CSRF)* [online]. 2022 [cit. 2022-03-03]. Available at: `https://owasp.org/www-community/attacks/csrf`.

[54] ŠKRABÁNEK, P. *Strojové vidění* [lecture]. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky, CZ, 2021 [cit. 2022-03-20].

[55] SLUNSKÝ, T. *Vícetřídá segmentace 3D lékařských dat pomocí hlubokého učení*. Brno, CZ, 2019. [cit. 2022-02-22]. Master's thesis. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací. Available at: `http://hdl.handle.net/11012/177588`.

[56] SQLAlchemy. *The Python SQL Toolkit and Object Relational Mapper* [online]. [cit. 2022-03-03]. Available at: `https://www.sqlalchemy.org/`.

[57] Stäubli. *Targeting nanoparticles with robotics* [online]. [cit. 2021-03-09]. Available at: `https://www.staubli.com/global/en/robotics/industries/medical-robotics/cancer-treatment-of-the-future.html`.

[58] Tatum, M. *What Is Machine Perception?* [online]. 2022 [cit. 2022-02-22]. Available at: `https://www.easytechjunkie.com/what-is-machine-perception.htm`.

[59] Universal Robots. *Technical details UR3* [online catalogue sheet]. [cit. 2022-02-22]. Available at: `https://www.universal-robots.com/media/1801288/eng_199901_ur3_tech_spec_web_a4.pdf`.

[60] Universal Robots. *UR5e technical details* [online catalogue sheet]. [cit. 2021-03-04]. Available at: `https://www.universal-robots.com/media/1802778/ur5e-32528_ur_technical_details_.pdf`.

[61] Universal Robots. *About Universal Robots / History* [online]. 2022 [cit. 2022-04-23]. Available at: `https://www.universal-robots.com/about-universal-robots/our-history/`.

[62] UVD. *MODEL C REVOLUTIONIZING INFECTION PREVENTION* [online]. UVD Robots [cit. 2021-03-09]. Available at: `https://uvd.blue-ocean-robotics.com/robots`.

[63] uWSGI. *The uWSGI project* [online documentation]. 2022 [cit. 2022-03-23]. Available at: `https://uwsgi-docs.readthedocs.io/en/latest/`.

[64] WSGI. *What is WSGI?* [online documentation]. [cit. 2022-03-03]. Available at: `https://wsgi.readthedocs.io/en/latest/what.html`.

[65] Yang, L., Cao, Q., Lin, M., Zhang, H. and Ma, Z. Robotic hand-eye calibration with depth camera: A sphere model approach. In: *2018 4th International Conference on Control, Automation and Robotics (ICCAR)*. 2018, p. 104–110 [cit. 2022-03-20]. DOI: 10.1109/ICCAR.2018.8384652.

[66] Yuan, Y., Chen, X. and Wang, J. Object-Contextual Representations for Semantic Segmentation. *CoRR*. 2019, abs/1909.11065, [cit. 2022-03-14]. Available at: `http://arxiv.org/abs/1909.11065`.

[67] Zhou, Q.-Y., Park, J. and Koltun, V. Open3D: A Modern Library for 3D Data Processing. *ArXiv:1801.09847*. 2018, [cit. 2022-04-23].

Chapter 10

List of Appendicies

Appendix A

Contents of the Included Storage Media

- Electronic version of this master's thesis in PDF format.
- Poster of this master's thesis in PDF format.
- *Adobe Illustrator* Human-Machine Interface graphical designe.
- **Readme.md** file of instructions.
- 🔗 Robo Medicinae I - Server repository.
- 🔗 Robo Medicinae I - ROS repository.
- 🔗 Robo Medicinae I - Segmentation models CNN repository.
- 🔗 Robo Medicinae I - Gripper repository.