Development of an Intelligent Personal Assistant to Empower Operators in Industry 4.0 Environments

Matheus Teixeira Talacio - 42802

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering.

Work oriented by:

Prof. Paulo Leitao

Prof. Marcos Banheti Rabello Vallim

Bragança

November 2020

Development of an Intelligent Personal Assistant to Empower Operators in Industry 4.0 Environments

Matheus Teixeira Talacio - 42802

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Industrial Engineering through the Dual Degree Program

Work oriented by:

Prof. Paulo Leitao

Prof. Marcos Banheti Rabello Vallim

Bragança

November 2020

Abstract

Industry 4.0 brings a high level of automation to industrial environments and changes the way companies operate, both in operational aspects and in human relations. It is important to define the role of the millions of operators affected in this new socioeconomic paradigm, integrating new technologies and empowering the workforce to take advantage of aspects such as the flexibility and versatility that human operators bring to production lines.

To advance the implementation of this objective, this work proposes the development of an intelligent personal assistant, using concepts of human-in-the-loop cyber-physical systems and context awareness, to assist operators during manufacturing tasks, providing the necessary information for the fulfillment of operations and verifying the accuracy to inform them about possible errors.

The implementation is divided in two parts. The first part focuses on an application that supports the real-time operations that can be present in the industry, such as pick and place in warehouses and the assembly of complex equipment on an assembly line. Through an interface, the instruction is given and, using artificial vision techniques with images coming from an IntelRealsense camera, it verifies if the operation is being correctly performed. The gathering of this information occurs in a context awareness algorithm, fulfilling the requirement of intelligent personal assistant and providing feedback to the operator so that the tasks are performed with efficiency and lower incidence of errors.

The second part includes the training of these operators in an immersive environment through a virtual reality equipment such as the Oculus Go. The immersive scenario, developed in Unity3D, uses as a model the real workbench, bringing the possibility of

performing these trainings in any environment and excluding the need to use real equipment, which could be damaged by the user's inexperience.

The results achieved during the validation tests performed in these two parts, commenting on the strengths, challenges and failures found in the system in general. These results are also qualitatively compared with traditional applications of the proposed case studies in order to prove the fulfillment of the objectives proposed in this work. Finally, the usability test is presented, which provides data on weak points in the user experience for possible improvements in future work.

Keywords: Industry 4.0; Human-In-The-Loop Cyber-Physical Systems; Context Awareness; Intelligent Personal Assistant; Virtual Reality; Machine Vision.

Resumo

A indústria 4.0 traz um nível elevado de automação a ambientes industriais e muda a forma em que empresas funcionam, tanto em aspectos operacionais quanto em relações humanas. É importante a definição do papel dos milhões de operadores afetados neste novo paradigma socioeconômico, fazendo a integração das novas tecnologias e capacitando a mão de obra para fazer proveito de aspectos como a flexibilidade e versatilidade que operadores humanos trazem às linhas de produção.

Para avançar a implementação deste objetivo, este trabalho propõe o desenvolvimento de uma assistente pessoal inteligente, utilizando conceitos de human-in-the-loop cyber-physical systems e context awareness, para auxiliar operadores durante tarefas de manufatura, provendo informações necessárias para o cumprimento de operações e verificando a acurácia para informá-lo sobre possíveis erros.

A implementação está dividida em duas partes. A primeira parte foca em uma aplicação de operações em tempo real que podem estar presentes na indústria como pick-and-place em armazéns e a montagem de equipamentos complexos em uma linha de montagem. Através de uma interface é dada a instrução a ser realizada e, utilizando técnicas de visão artificial, com imagens vindas de uma câmera IntelRealsense, verifica se a operação está sendo corretamente executada. A junção dessas informações ocorre em um algoritmo de context awareness, cumprindo o requisito de assistente pessoal inteligente e fornecendo o feedback ao operador para que as tarefas sejam realizadas com eficiência e menor incidência de erros.

Já a segunda parte engloba o treinamento destes operadores em um ambiente imersivo

através de um equipamento de realidade virtual como o Oculus Go. O cenário, desenvolvido no Unity3D, utiliza como modelo a bancada real, trazendo a possibilidade de se realizar esses treinamentos em qualquer ambiente, excluindo a necessidade da utilização de equipamentos reais e possíveis danos originados de inexperiência do usuário.

Os resultados apresentam os testes de validação realizados nestas duas partes, comentando os pontos fortes, desafios e falhas encontradas no sistema em geral. Estes resultados também são comparados qualitativamente com aplicações tradicionais dos casos de estudo propostos de forma a comprovar o cumprimento dos objetivos propostos neste trabalho. Por fim, é apresentado o teste de usabilidade que fornece dados em pontos fracos na experiência de usuários para possíveis melhorias em futuros trabalhos.

Palavras Chave: Indústria 4.0; Human-In-The-Loop Cyber-Physical Systems; Context Awareness; Assistente Pessoal Inteligente; Realidade Virtual; Visão Artificial.

Contents

List of Tables

List of Figures

Acronyms

AI Artificial Intelligence.

API Application Programming Interface.

AR Augmented Reality.

BGR Blue-Green-Red.

CA Context Awareness.

CPS Cyber-Physical Systems.

HiLCPS Human-in-the-Loop Cyber-Physical Systems.

HSV Hue-Saturation-Value.

ICT Information and Communications Technology.

IoT Internet of Things.

IPA Intelligent Personal Assistant.

JSON JavaScipt Object Notation.

REST Representational State Transfer.

RGB Red-Green-Blue.

SDK Software Development Kit.

SUS System Usability Survey.

TOF Time-of-Flight.

VR Virtual Reality.

Chapter 1

Introduction

The 4th industrial revolution is pushing the adoption of emergent technologies, e.g., Internet of Things (IoT), Artificial Intelligence (AI), Big data, Virtual Reality (VR) and Augmented Reality (AR), aiming to transform the way industries operate to increase the intelligence, responsiveness and reconfigurability. Therefore, Industry 4.0 is the joining of these new technologies to modify the interaction between all components, technological or social, within a manufacturing. [1]

In particular, Industry 4.0 enables the increasing level of automation and digitization in the factories of the future [2]. Cyber-Physical Systems (CPS) act as a backbone to develop such emergent production systems, contributing to the digital transformation of production processes towards smart processes, machines and products. CPS aims to connect the various physical components, e.g. sensors and actuators, with cyber systems composed by controllers and communication networks to achieve a common goal [3].

However, this category of automation is convenient for companies due to its level of productivity and long term cost reduction. Effectively, it brings social problems, since there is a great substitution of human labor, eliminating existing jobs today. According to the report published by McKinsey Global Institute, by 2030, between 400 and 800 million people will have their jobs taken by automatic systems and will need to look for new jobs [4]. Therefore, it is necessary to find ways to empower human capabilities to integrate them into new technologies.

In this emergent CPS environment, humans can have a very important role since being the most flexible elements in the automated system, possessing a greater cognitive capacity for decision making in unforeseen situations, thus being required for their symbiotic integration. In particular, instead of performing repetitive and monotonous tasks that can be fully automated, humans will be requested to perform add-value tasks, e.g. assembly of complex and customized products or performing maintenance interventions.

One way to implement this integration is through Human-in-the-Loop Cyber-Physical Systems (HiLCPS). HiLCPS incorporates human and physical components via an embedded system interacting with data collector sensors on the operator's intentions [5] and enable tasks such as adjusting set-points, directly commanding the system and providing data (identification, error prevention and insights), through its users. [6]

In a factory, where there is a wide variety of products, it is common to use manual assembly, and to compete with mass production - where automatic processes are present -, the training and support of skilled operators is necessary [7]. Currently, manuals referring to these tasks are mostly static documents that do not include the operation context [8], making the browsing of information difficult and not offering feedback to the operator. These factors increase the time required for the task execution and the effectiveness of training.

In this context, AI-based assistants and virtual interfaces can be used to assist humans to realize manual operations in a faster and more cost-effective manner, taking advantage of the huge amount of data available at the shop floor, as well as the emergent Information and Communications Technology (ICT), e.g., AI, VR and AR [9]. These intelligent assistants can support the online monitoring of the equipment condition during the execution of assembly and maintenance operations, and combine this information with diagnostic reports and their previous experience in analogous situations, determining the best action plans to be carried out.

Specifically, VR and AR technologies allow operators to perform these tasks in immersive and augmented scenario, promoting the human-machine interface in industrial

environments and contributing for a more efficient, cost-effective and time-effective operation execution.

The use of VR also allows the creation of interactive, effective and low-cost training scenarios, supporting flexibility during the session, as well as the possibility to test the reaction to failures and the automatic production of performance reports. Also, the hardware currently available for VR visualization has a small size factor, increasing its applicability since it replaces training stations that can occupy a large space and, depending on the complexity, a high cost, where in VR the value is fixed independently.

1.1 Objectives

This thesis has the objective of developing an Intelligent Personal Assistant (IPA) that supports human operators to perform faster and more cost-effectively their assembly operations. In particular, the developed IPA's application use AI and image processing to guide operators to assemble correctly customized products. These solutions could allow an increase in the productivity of operators through the direct integration with emerging technologies, thus adding value to human operations that must compete with highly automated industries.

A virtualization of one of IPA's real scenarios to virtual reality will also be developed with the objective of enabling the training of operators in an immersive environment. This application can increase the training efficiency as it can present a scenario that is identical to its industrial application, where the user performs the same actions as the latter anywhere, due to the fact that the virtual reality equipment is portable.

1.2 Thesis Structure

This thesis is structured in introduction, state of the art, case studies, development and architecture, results and conclusion.

In the introduction a contextualization about the purpose of the operator in an industrial 4.0 environment will be presented, along with the motivations and objectives of this work. The state of the art contains a literature review on human-in-the-loop cyber-physical systems, intelligent personal assistant applications with their enabling technologies and applications in augmented/virtual reality. It will also be presented the case studies in which this platform will be built, pointing out the requirements of each system and how they will be implemented and validated.

The development chapter explains the development of the case studies, presenting their respective architectures and algorithms used in the implementation and results present the data obtained from the validation of each case, discussing the main characteristics observed and their failures. Finally, the conclusion summarizes the results obtained from the initial objectives and proposes some points for future work.

Chapter 2

State of Art

Industry 4.0 relies on the integration of human operators into the production processes for enhanced product value. This achieves manufacturing flexibility in complex human-machine systems, where humans and machines can not be considered in an isolated manner but instead regarded as a collaborative team [10]. This requires the presence of operators on the design of human-centred adaptive manufacturing systems, allowing a symbiotic integration.

However, the execution of some operations, e.g., assembly and maintenance is normally complex and requires high-level expertise from the operators for an efficient execution of these operations in short time, meeting the desirable quality and with the minimal impact on the normal production cycles. This requires to enable human-machine communication and collaboration, through the use of innovative technologies, e.g. machine learning, VR and AR, that will support the operators during the realization of their tasks.

An important topic regarding these HitLCPS applications is Context Awareness (CA). CA refers to a system's ability to analyze and capture properties that make up the aspects of a specific operation [11]. This technology has a high range of applications, enabling everything from energy efficiency analysis in wireless connectivity [12] to seamless mobile network handovers [13]. It is also a key part in systems that provide support to operators as they can adapt in real time and increase efficiency in decision making, important features in implementing an IPA.

2.1 Intelligent Personal Assistants

An Intelligent Personal Assistant (IPA) is a guidance software system that can support the execution of operations, facilitating the interaction between the operators and machines or computers. It is based on data mining from images, video, and voice commands captured from sensors distributed in the environment [14], [15]. The use of IPA as an intelligent assistant to inform the operator with real-time and historical data, guiding through the best action plan to perform the operation [16] can improve the quality of the executed operation, particularly in case of complex and/or customized ones. As an example, maintenance technicians can use an IPA system to obtain useful real-time information about the machine condition, recommendations and actions to be taken, as well as additional useful information, e.g., documents, websites or videos [17].

These systems can provide real-time information and instructions, overlapped on the workstation, replacing traditional methods that rely on printed manuals and real machines that can be damaged [18].

The more common commercial versions of IPA are Amazon Alexa, Microsoft Cortana, Google Assistant and Apple Siri. In industrial environments, IPA are being used for training as a means of learning through the first-person experience [19], [20], to train operators in new machine maintenance procedures [21] and to support operators in performing complex and customised assembly tasks and maintenance operations [22].

An important dimension is the data value in this digital transformation era. In fact, the huge amount of data continuously being generated on the shop floor can be used to boost these intelligent personal assistants. For this purpose, the access to historical and real-time data related to the equipment condition can be performed by using IoT technologies and provided to the user through proper interfaces. AI techniques can be used to digest and analyze big amounts of data to determine the best action plans to be carried out during the maintenance operations. These would contribute to intelligent maintenance within the industry 4.0 context [23].

2.2 Intelligent Personal Assistant Technologies

The core part in an IPA is the task management software. It is necessary to integrate the peripheral equipment with an intelligent algorithm that controls the flow of operations through CA to provide adequate feedback to assist in user decision making. Currently there is a range of programming language options with different features, e.g. Python, C++ and Java, and their use is dependent on initial project requirements and compatibility with the technologies used.

While there has been a matching increase in research, there is still a need to build upon this research, namely in combining automation of core technologies (haptic devices, stereo graphics, adaptive content, assessment and autonomous agents) [24], and developing reusable automation frameworks for virtual training that allows the easy integration of new contents and procedures.

2.2.1 Machine Vision

One of these technologies, which is emerging in the development of intelligent applications is artificial vision, is also known as machine vision. This technique is especially important in the supervision of assembly operations because it allows the acquisition of data through images, which would not be possible only by inference of context in software. Artificial vision has already proven itself as indispensable in human support systems, as its most remarkable applications refer to the use in stand-alone vehicles for the detection of characteristics of a road [25], inspection of products [26] and assembly aid in production lines such as the MOON project, conducted by Airbus, which implements an integration with digital manuals and, through a camera and tablet, superimposes the instructions on the object in real time [27], demonstrated in figure 2.1.

Not always just a two-dimensional image is enough to meet the requirements of a solution. For this, there are specialized cameras that have the ability to measure the distance of points in the field of view, enabling more complex applications that use three-dimensional data such as gesture recognition [28] and pedestrian detection on a road for

Figure 2.1: Instruction superimposition in the Airbus MOON project.[27]

anti-collision systems [29]. There are several technologies used to accomplish this goal and should be considered according to its necessity. The two most popular ones are Time-of-Flight (TOF), used by Microsoft Kinect 2.0 [30] and Microsoft Azure Kinect [31], and stereo vision, used by Intel Realsense [32], shown in figure 2.2 and Stereolabs ZED [33]. Table 2.1 presents the result of a research conducted by Seeed on the comparison of these two technologies [34].

Figure 2.2: Intel Realsense D415 depth camera. [32]

To use data received by camera capture, it is convenient to use image processing libraries such as OpenCV [35], Google Cloud Vision [36] and Tensorflow [37]. These libraries implement essential tools for recognizing shapes and objects, e.g. contour matching, morphological operations and feature detection. In addition to the common techniques of

Table 2.1: Comparison between Stereo Vision and Time of Flight sensors [34]

Features	**Stereo Vision**	**ToF**
Working distance	<2m	0.4-5m
Accuracy	5%-10% of distance	<0.5% of distance
Resolution	medium	low
Power consumption	comparatively high	medium
Use environment	environment with ambient light or outdoor	indoor and outdoor
Frame rate	high	variable
Hardware cost	low	medium
Software processing requirement	high	low

property extraction, it is also possible, through the aforementioned libraries, to use machine learning to train neural networks for recognition of complex shapes like wildlife monitoring in a nature reserve [38].

2.2.2 User Interaction

It is important that an IPA system communicates with the user, receiving input from operations and providing feedback. For the reception of control information from the operator, there are not only artificial vision techniques, described in section 2.2.1. Developments in this area increase traditional technologies, such as mice and keyboards, with gesture recognition, allowing a wider range of control possibilities and increasing the accessibility of the system. This technology enables application controls from vehicle multimedia systems [39] to smart-home devices [40], without the need to interact directly with the dashboards.

User feedback is the end point of an IPA system. At this stage it is necessary that the information is demonstrated to the user in a concise and intuitive way so that it can be interpreted correctly and provide support to the task being performed.

Historically, the most common way to demonstrate feedback is through displays. This technology has its appeal linked to its ease of use, cost and amount of variations available in the market such as cell phones, tablets and monitors. For the use of displays, it is

necessary the use of computer equipment, paired with interface software to render the information. The choice of these equipments is related to the platform requirements and basically can be divided between web and binary applications. Figure 2.3 shows a collaborative robot demonstration using a tablet to interface with the operator.

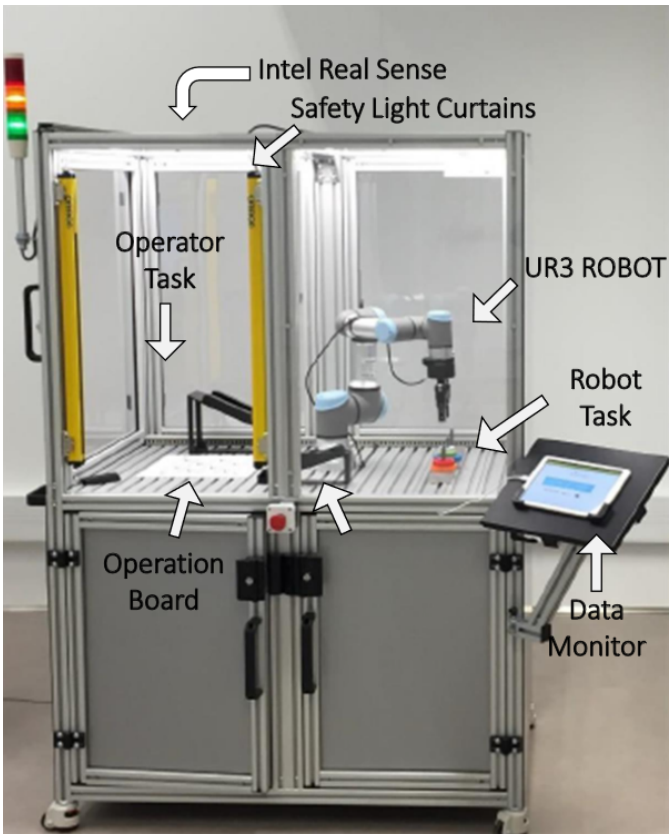

Figure 2.3: Collaborative robot demonstration with data being displayed via a tablet.

Web applications have their code located on servers that distribute the content over the Internet and are consumed through browsers. Because they are distributed over the Internet, web applications can be accessed from any point connected to the network without the need to install the files on the device to be used. However, these applications need a constant connection. If the hosting is done by a third party company, when losing access to the internet, it is not possible to use the system. Also one cannot forget security issues, if access to the interface is public, it is necessary to create systems to prevent leakage of critical information and prevent attacks. Among the most used frameworks

for the development of web applications, according to Stackoverflow [41], are: Django (Python), React (Javascript) and Laravel (PHP).

Binary applications are software compiled to run locally on the devices to which the displays are connected. These, unlike the previous one, do not need to be connected and can be executed even if the devices are not on a network. However, it is necessary that the software be installed for each machine you want to use and they introduce compatibility problems between devices e.g. for a Windows machine and a cell phone with Android, making it necessary to adapt them, or in extreme cases remake them, to work on different platforms.

The use of displays is the most common, but they have limitations as they are locked in place, causing the operator to change the focus of the task to seek information. To mitigate this problem, some technologies present feedback through the overlapping of data in the work area. Graphic projectors use lenses to enlarge an image generated internally by a device and use the same software workflow explained earlier. Therefore, they have easy replacement of common displays because they can use the same applications and have the same video connection e.g. HDMI or VGA. However, graphic projectors need a low brightness in the environment for better visualization of their output, making it unfeasible in certain situations such as outdoor use. Uses such as data demonstration of a robotic system [42], shown in figure 2.4, and support for human-machine applications [43] present the advantages of having easily accessible information projection on surfaces near the operator.

Another technology used for overlapping information are laser projectors. Unlike graphical projectors, these use laser diodes combined with motorized reflectors to draw contours in areas of interest. These projectors have a superior performance in different levels of ambient lighting, but have a higher cost [44]. Through this technology it is possible, to precisely demonstrate to operators instructions using CAD models interpreted by a control software to assist in assembly tasks [45], as presented in figure 2.5.

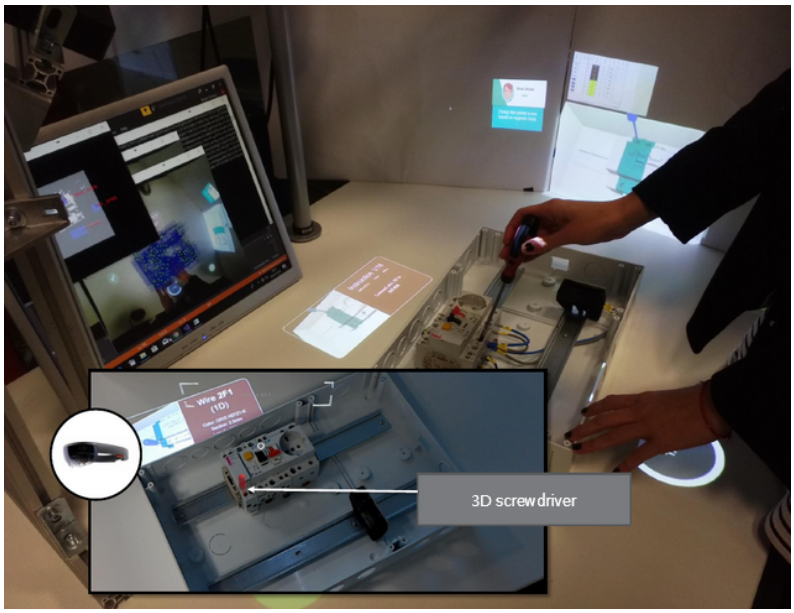

Figure 2.4: Data demonstration via a standard projector [42]

Figure 2.5: Laser projector assisted assembly [45]

2.2.3 Speech Recognition

In terms of user input without the need for direct interaction with equipment e.g. keyboards, buttons and touch screens, the recognition and analysis of voice commands brings the ability to interact with IPA systems quickly. The area of voice recognition advances every year developing new techniques that increase reliability in detections [46], with emphasis on research using deep neural networks to solve problems of external noise [47].

This technology has several applications in different areas such as communication aid between speech impediment patients and physicians in clinical environments, voice commands on airplane flight decks and control of residential IoT devices. In industry we have the development of speech recognition to command actuators, particularly, [48] uses this technique to control an motor connected to a conveyor belt to support the development of intelligent industries.

It is convenient to use tools ready for the inclusion of speech recognition in projects under development, as these have already been tested and have their results published, eliminating the need to create models from scratch. Among the many existing platforms for local processing, CMUSphinx [49] stands out using the Hidden Markov Model, a statistical model and Kaldi [50] using deep neural networks. Although these libraries enable offline processing, the cloud based Application Programming Interface (API) have a more robust architecture as they have a larger database for training and are always being updated. Google Speech API [51], IBM Watson [52] and SpeechAPI [53] are some of these APIs available in the market.

2.3 Augmented and Virtual Reality

Usually, the IPA software is supported by VR and AR technologies, adopting e.g., head-mounted devices, exampled by figure 2.6 (e.g., Google Glasses [54], Microsoft HoloLens [55] and Facebook Oculus VR, [56]) or regular displays supported by cameras. These technologies improve the transfer of information from the digital to the real world and can provide strong benefits for human integration.

Figure 2.6: Facebook Oculus Go

In particular, AR can complement intelligent assistants by supporting the flexible and extensive presentation of virtual information together with real world objects, this makes possible a quick consultation of information directly in the area of interest, since the operator does not need to change the focus of work to look for it in other sources [57]. AR also has the ability to make these instructions more intuitive, as they can demonstrate the steps to be performed by superimposing the objects being manipulated in real time according to their progress. One use of this feature is its use in training sessions for operators in product assembly [58].

On the other hand, VR can support immersive experiential learning, to train for the complex interplay between the physical and digital world, as it uses a head mounted display to immerse the user in the environment, making the operator fulfill the task objective with a greater sense of being in the real situation. In addition to the immersion benefits, VR also allows the simulation of risk situations e.g. training in the mining industry [59], shown in figure 2.7, and practical surgical exercises for medical students [60].

Additionally, some companies, e.g ITI VR [61], DISTI [62] and ECA [63] Group, develop VR and AR training solutions for industrial applications, but mostly devoted to large corporations and addressing specific training scenarios focusing on heavy and expensive machines. Lowering the costs of design and development of VR and AR environments

Figure 2.7: Training in the mining industry using a virtual reality application. [59]

for Industry 4.0 is also interconnected with reusing components of the VR framework and the ease implementation of new maintenance or assembly procedures.

For the development of VR/AR applications it is useful to use software capable of generating graphics, communicating with the head mounted display and, if used, performing the physical calculations of the simulation. These software, also known as graphics engines, were born to assist in the development of virtual games. However, since they are easy to develop with, have a large number of guides and a variety of functions and support libraries, they attract other academic applications [64]–[66]. Even though graphics engines like Unity3D [67] and Unreal Engine [68] were created with the main reason as game development, there are already software like Amazon Sumerian that is focused on diverse uses, e.g. tourism, retail and education [69].

Chapter 3

Description of the Case Studies

All the developments presented in this thesis were performed considering two scenarios for the applicability tests. These scenarios were developed considering the objectives of this work and merge the use of context awareness, augmented reality, virtual reality and machine vision systems in different ways in each situation.

These scenarios are suggested as following:

- Gift-box assembly
- Complex product assembly
- Virtual training for gift-box assembly

3.1 Gift-box Assembly

In this study scenario the user must carry out the organization of a box of different products, considering the order of disposal of the products requested by the customer. Each of the gift boxes has four spaces that must carry the products in the order presented. The order of disposition of the products is presented by the algorithm to the operator through a screen and the activities performed by the operator will be supervised by an intelligent system that will verify the assertiveness of the assembly. If the operator

performs the assembly incorrectly, the intelligent system will indicate the error and the operation cycle will only end when the system detects that the assembly is correct.

This scenario presents a simple assembly, but it can be recurrent in industries, as for example a pick-and-place operation in warehouses, where orders must be placed in boxes different for each customer and with a wide variety of products. The effectiveness of the algorithm, in relation to its symbiosis with the operator, can represent a gain in efficiency and quality of products by companies, especially those with activities with similar characteristics, but different details or random order of activities.

To collect the results, this scenario was implemented in a real environment where it is possible to perform the tests and evaluate the behavior of the implemented algorithms. The tests were performed with 50 consecutive operations that appeared randomly to the user on the screen. The arrangements in this case were chosen by the algorithm itself, since for this case study it was not necessary to enter specific instructions.

3.2 Complex Product Assembly

In this case study the scenario presents more complex three-dimensional assemblies. The presentation of the assembly is made through a step by step to the operator who must follow it to complete the final product. The intelligent checking system is also active in this scenario, however, in this case, each step is understood as a complete activity and the next step of the assembly is only presented to the operator when its previous step is completed perfectly.

At the beginning of the assembly cycle the operator chooses which part he wants to assemble in a list presented by the interface presented through a monitor. After this choice the activity planner informs the system which activities will be executed and the system presents the first step to the operator.

For the test of this scenario, it was chosen 4 assemblies to be performed 5 times each to validate the expected results and find incorrect detections through on purpose operator errors.

It was also performed tests with 5 users performing an assembly in order to obtain an average duration between setup, assembly and completion time.

For comparison, tests were also performed with users using traditional paper instructions. The need for manual confirmation of the assembly was also added before proceeding to the next step as well as at the end of the scenario to simulate a filling of a report.

The advantage of this scenario for industries is that the operator does not need to have prior knowledge of the assembly to be performed and allows the flexibility of operators between different production lines. Also, in very complex assemblies, the operator has the proper supervision so that errors do not happen, also increasing the efficiency and quality of production.

3.3 Virtual Training for Gift-boxes Assembly

A third case is also performed using virtual reality to simulate the real operation described in the case of section 3.1. This scenario, different from those mentioned above, has the proposal of helping in the training of the operator in a virtual and immersive way, since it is not necessary to use real equipment and allows the realization of non-viable situations such as property breakage and failures.

The virtual visualization was built based on a real training scenario and the operator must follow the instructions presented by the system and be able to perform the assembly of the gift-box. The same verification characteristics of the original case study should be followed to provide feedback to the user and assist him in the task.

Tests of this application were done around the user experience during the use of the platform, as the results of the IPA application are already presented with the original case study.

This type of training can be very beneficial, especially for work in dangerous environments, as the operator can learn all safety aspects before being placed in an environment where an error can have major consequences.

Chapter 4

Development/System Architecture

4.1 Workbench System Architecture

The IPA application has two macro divisions, which are hardware and software. The hardware includes the workbench, where the assembly is done and has the technologies to capture data and show information about the operations performed. The software processes the information received by the workbench in order to confirm the instructions and return their outcome.

The behavior of this intelligent assistant is formed not by a single module, but by the combination of all the components that must communicate constantly to support the accomplishment of the given task in an efficient way.

4.1.1 Hardware Architecture

To capture the image and depth of the scene, the Intel Realsense D415 is used. This camera has a stereo infrared module capable of generating depth by capturing two images in two different positions and, through the disparity in the position of the objects, it is possible to estimate the distance they are from the camera. There is also an Red-Green-Blue (RGB) sensor for acquiring color images. It is also important to mention that a code library, developed by the manufacturer, is provided for communication with the camera

in the most used languages, such as Python, C++ and C#.

The workbench is also equipped with two options for image display, which displays IPA's graphical interface, being a monitor mounted in front of the operator and a projector facing the table. The choice of use is at the discretion of the operator.

All the components mentioned are connected to a computer, responsible for hosting both the web server and the assembly verification algorithm. Therefore, the communication of the hardware components is made directly with the software, without the need for wireless protocols and providing the possibility of performing the exercises without the need for Internet.

Finally, the structure has components to support the operation, such as, a upper direct illumination by LED to illuminate the scene and assist the image recognition, boxes to store the parts used in the assemblies and an RGB LED strip, mounted below the storage boxes, to indicate to the operator which part should be used in the instruction being performed. The control of this ribbon is made through an Arduino UNO microcontroller.

Figure 4.1 demonstrates the structure with its components explained.

4.1.2 Software architecture

The software architecture is subdivided into three systems: web server, database and verification algorithm.

The web server contains the graphical interface, responsible for generating the information and instructions about the scenarios, and the backend, which maintains the interface states, communicates securely with the database and exposes the Representational State Transfer (REST) API for internal communication, with the interface itself, and external, with the verification algorithm.

A MySQL database is used to store the data of all assemblies. The other two subsystems have direct communication with this server to query and register instructions or flags needed to perform their respective routines.

In the verification algorithm, the image recognition, associated with flow control codes,

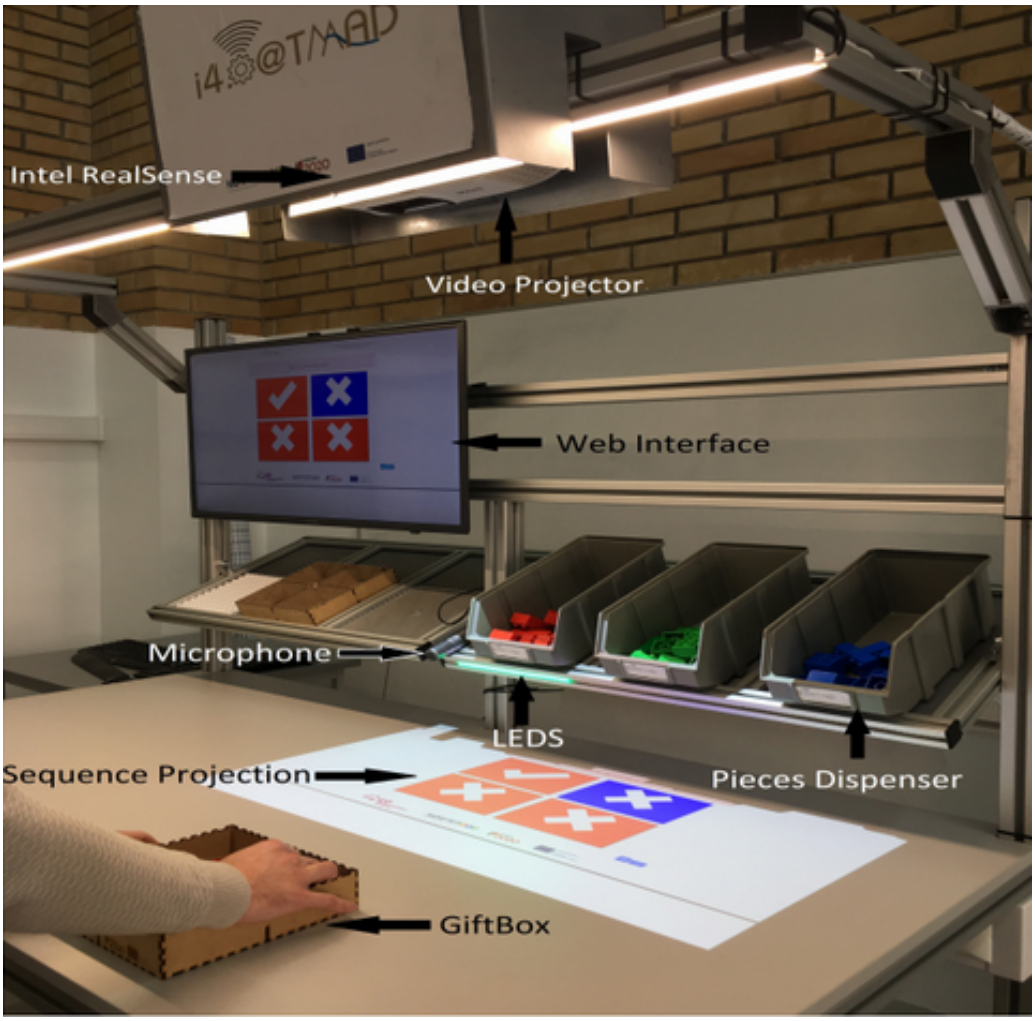

Figure 4.1: Workbench structure with its components presented.

specific for each scenario, forms the most important behavior of IPA, which is to analyze and process the current context of the scene against the assembly instructions. Through this module, it is verified if the actions performed by the operator are in accordance and, with this analysis, the web interface is kept updated to provide feedback and continuity of the operation. This algorithm was codified in Python, ensuring fast runtime, easy development, camera compatibility and support of image manipulation libraries.

The figure 4.2 shows the representation of all systems and data flow between them. The systems are highlighted through a dotted line, with their subtitles, being their internal components the subsystems.

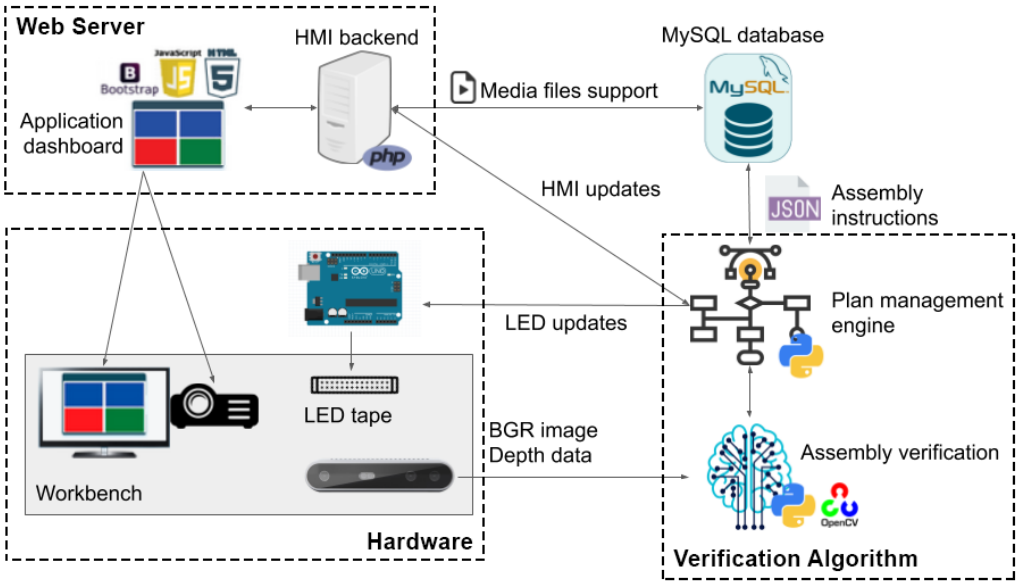

Figure 4.2: Relationship between IPA systems with data flow.

4.2 Image Recognition and Classification Algorithm

The image processing algorithm is used to obtain information about the parts presented in the scene and classify their aspects such as color, size and position. This algorithm is used in both scenarios, as they use the same type of piece. So, when executing it, only a list with the detections is returned, and the upper layer of the program uses it as needed.

This code is supported by two libraries: OpenCV and IntelRealsense2.

OpenCV is a computational vision tool focused on real-time applications and has functions that assist in operations such as filters, morphology, contour detection and color space conversion. IntelRealsense2 is responsible for the communication with the camera, and it is used to obtain the image and the depth information of the scene.

At the beginning of the execution, the camera starts with the saturation parameter higher than the default because, as the proposed algorithm uses a Hue-Saturation-Value (HSV) filter, the colors are accentuated, facilitating the detection of the components. After initialization, a frame is captured and converted from Blue-Green-Red (BGR) (standard format of the camera software) to HSV. Figure 4.3 represents the HSV color space.

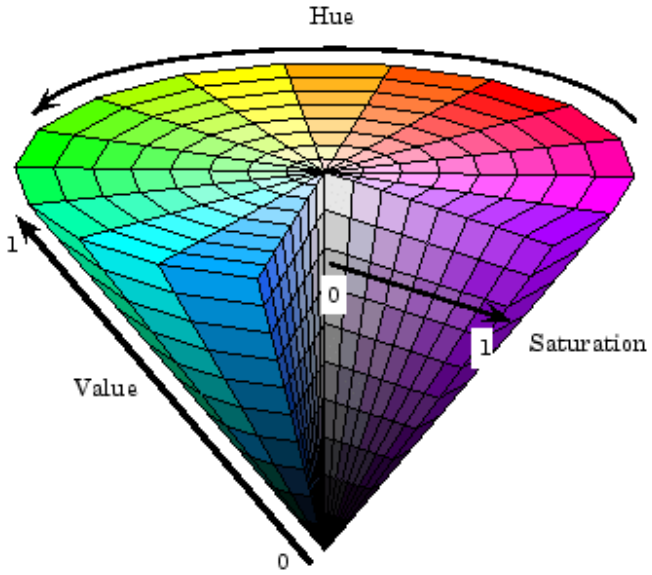

Figure 4.3: HSV color space representation. [70]

The main reason for the use of this space is the ease of representing the desired colors for the filter because it is closer to color perception in relation to human eyes and presents better performance in artificial vision algorithms [71]. If the RGB space was used, to describe the colors it would be necessary to change three values of red, green and blue, thus being a three-dimensional representation. In HSV space, by leaving saturation constant, a two-dimensional representation of colors is obtained, being more intuitive to select desired values and debug the program if necessary. Figure 4.4 shows the HSV space

representation with constant saturation.

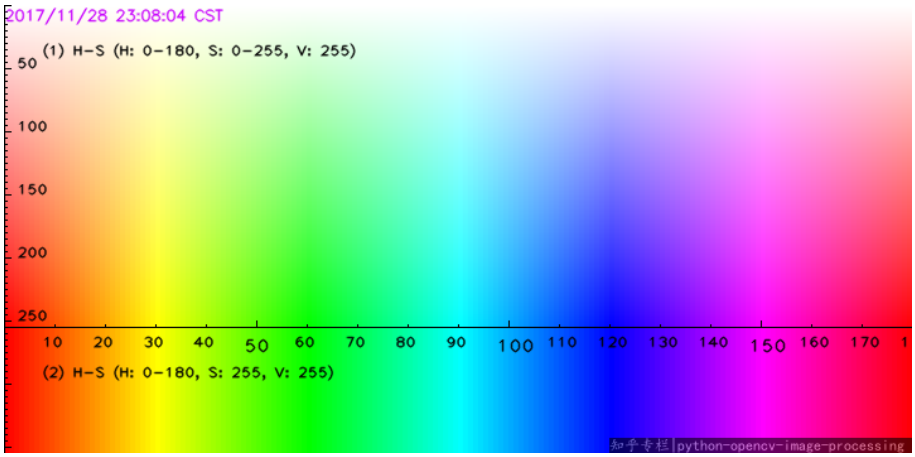

Figure 4.4: HSV color space representation with constant saturation. [72]

After the color space conversion, there is a height versus width matrix of the image, where the fields correspond to the HSV values of each pixel obtained by the camera and thus it is possible to apply the filter. This filter is intended to isolate pixels from the image according to a pre-specified color range. Through a loop, the matrix is iterated and, if the pixel is within the range, in a second matrix a value of 0 or 1 is set for the two possibilities, ultimately creating a black (0) and white (1) mask. This operation is carried out by OpenCV's "inRange" function. As there are 4 different colors, 3 for the pieces and 1 for the box, this filter is applied 4 times, obtaining 4 different matrices.

The difference in lighting in the scene may present noise in the filter, making the edges not well defined, therefore, to mitigate this problem, a small morphological dilatation operation is applied to the obtained masks, making the edges smoother.

A particularity occurs in the selection of the red color filter. It is possible to observe through the figure 4.4 that, as it is a color disfigured circle to a rectangular one, there is a break in the red tones, which are present in the smallest and largest values of hue. To mitigate this problem, two different filters are applied to the red color, one for each separation, which later their masks are interpolated through an AND operation, finally obtaining a single mask. The table 4.1 presents the chosen intervals, in HSV values, for each color.

Table 4.1: Chosen HSV intervals for each color in the filter.

Colors	**Lower Boundary**	**Upper Boundary**
Red Low	0, 146, 20	7, 255, 255
Red High	163, 146, 20	189, 255, 255
Green	50, 100, 20	77, 255, 255
Blue	100, 100, 20	125, 255, 255
Brown (Box)	0, 80, 20	31, 255, 255

From the mask, OpenCV is used to obtain the contours of the white pixel cluster areas. The "findContours" method implements the algorithm proposed by Suzuki and Abe to extract topological information through a sequence of coordinates obtained by following borders between white and black pixels in the mask [73]. This method returns a array with all the contours found. From these contours it is possible to execute other methods in the library to obtain area, center of mass and vertex position.

Each contour of each mask is analyzed to assemble the return values of this algorithm. To simplify the analysis, using the "minRectArea" method it is obtained a rectangle with the smallest possible area that fits on top of the contours obtained and, as the parts have this shape and the contours are not always perfect, it is possible to perform the checks on regular rectangles with straight edges. For the analysis, it is also necessary to exclude invalid contours. Assuming that the area of valid objects has a fixed value range, it is possible to obtain these values empirically and, by comparing the area of all the contours obtained using the "contourArea" method, unwanted detections are excluded.

For the return values of this algorithm, a list is initialized where its members correspond to the identified objects, the table 4.2 presents the specific information returned for each object. The color of the current outline is the simplest to be identified because, as each mask corresponds to a specific color, the color of the mask being analyzed is assigned to the current object.

The second property to be determined is the shape, this step is ignored when analyzing the box because it has only one shape. The assembly pieces can be categorized into three: square, horizontal rectangle and vertical rectangle. It is known that a rectangular piece will always have an area larger than a square piece, so after this threshold value is found

Table 4.2: List of object properties classified with this algorithm.

Property Name	**Value Type**	**Example**
Color	Integer (1: Red, 2: Green, 3: Blue, 4: Box)	2
Position	Integer Pair	(30, 25)
Vertices	Array of Integer Pairs	[(14, 30), (14, 50), (33, 30), (33, 50)]
Orientation and Shape	Integer (-1: Square, 0: Horizontal Rectangle, 1: Vertical Rectangle)	1

empirically, a category can already be classified. If the piece is not a square, the height is compared to the width of the contour. To do this, the vertices must first be arranged in a logical order such as: top left, top right, bottom right, bottom left.

To achieve this, each part has a list containing the coordinates of the four vertices, composed of their values corresponding to the positions on the horizontal (X) and vertical (Y) axes of the image. First the list is sorted according to their values on the X axis, i.e. the vertex with the smallest horizontal position is first on this list and the subsequent ones follow in ascending order. Thus, when checking the 4.5 figure, with the two possible rotations found in an assembly, it is concluded that the first two items correspond to the left vertices. In these two vertices an operation similar to the previous one may be carried out and they may be ordered along the Y axis because, between the two, the one with the largest value on the vertical axis will correspond to the upper left corner, successively the remaining point will be the lower left corner.

Having been the top left corner identified, a two-dimensional Euclidean distance operation is performed, according to the equation 4.1 between this point and the others because they make a rectangular triangle relation as observed in figure 4.6 and, according to the Pythagorean theorem, the value of the hypotenuse is always higher than a catheter's, in this case, the connection forming the hypotenuse corresponding to the top left and bottom right vertices. Thus, the remaining labels can be inferred, with the shortest distance being the top right corner and the longest the bottom right corner.

$$distance = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \tag{4.1}$$

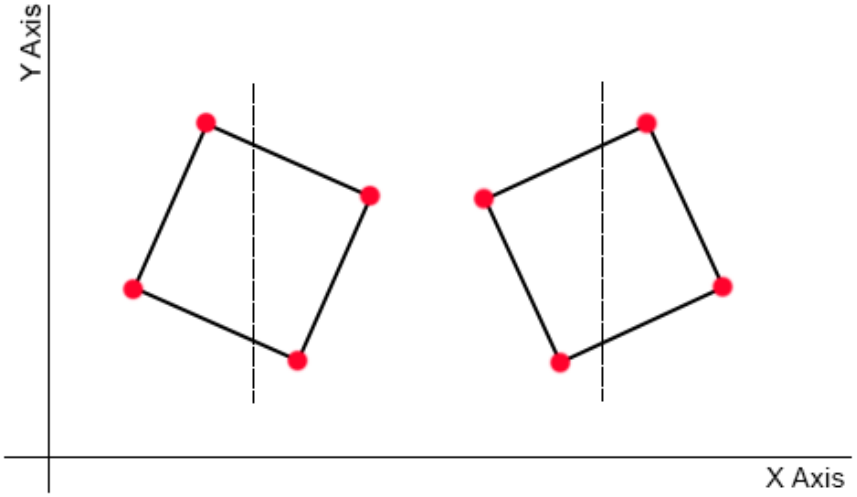

Figure 4.5: Separation between the two most extreme vertices of a piece with two possible rotations.

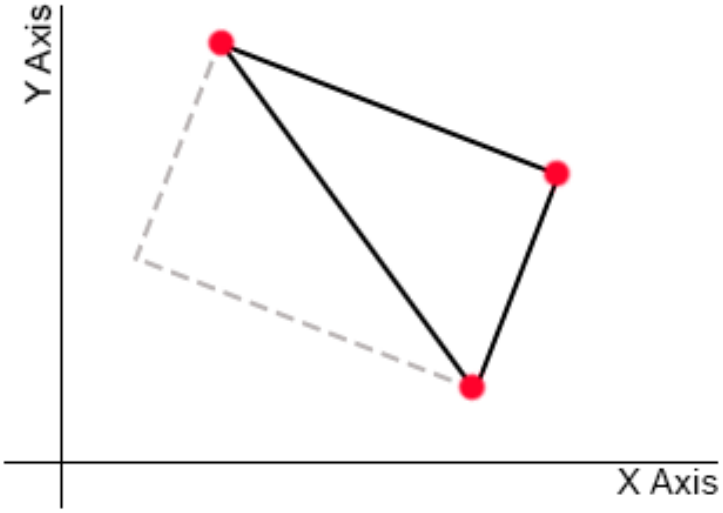

Figure 4.6: Rectangular triangle relation between the top left vertex and the right most vertices.

With the ordered corners, the width of the piece is considered being the distance between the top right corner and the top left corner and the height being the distance between the top left corner and the bottom left corner, this value is obtained from the two-dimensional Euclidean distance. If the piece has a height greater than the width, it is classified as vertical rectangular, otherwise horizontal rectangular.

The last attributes to be assigned are the absolute position in relation to the image, obtained through the contour method executed previously, and its depth in relation to the camera. This depth is acquired by capturing the information coming from the Intel Realsense stereo sensor and, through the library provided by the manufacturer, it is possible to obtain a matrix height versus width of a camera frame, where its fields correspond to the distance from the pixel to the sensor. This distance has a raw value, therefore it does not correspond to any specific unit. This algorithm does not need a particular unit because it only uses the values in code and the conversion would only serve to format the values to metric units.

It is important that the color and depth matrices be acquired from the same camera frame because, if they were taken from different frames, the pieces could be in different positions, causing incorrect relations between the two information.

Since this algorithm is executed in real time, to increase execution efficiency, a check to verify if an operator is making a modification in the assembly is included. For this, if a shorter than expected distance is detected in the depth matrix, considering the expected distance, it is determined that there is, for example, a hand above the assembly manipulating the exercise, thus, the analysis is not performed and the function awaits until the situation is normalized.

After making the described analysis of all the contours obtained, a list with all the parts is returned to the calling method.

4.3 Gift-box Assembly Verification Algorithm

When the operator selects this exercise, the web application will randomly generate a color sequence, which is stored in the database for later use. The role of the user is to place a piece in each position following the generated instruction. There are two requirements for the confirmation of a correct assembly: the piece in a position must have the same color as the instruction and there can only be one piece in each position.

Once the main verification algorithm is started, image recognition, described in subsection 4.2, is used to retrieve the list of the objects present at the scene. Figure 4.7 presents the gift-box assembly confirmation algorithm.

After confirming that there is the box between the recognized objects, it checks whether all the parts are inside the box through the OpenCV's "pointPolygonTest" function, which in a simplified way calculates the distance between a point and an contour, in this case any point of the piece and the box respectively, and returns its position as outside, on the border or inside. All the parts outside the box are ignored.

Next, the position of the part must be classified in one of the four possibilities. To do this, the top left corner of the piece is used to calculate the distance between this vertex and the four vertices of the box. As can be seen in figure 4.8, the position of the part can be inferred through the shortest distance obtained, that is, if the shortest distance is that of the part with the top right corner of the box, this will be its classification.

After the pieces are classified, it is checked if there are two pieces with the same designation, if so, an error message is sent to the interface to indicate that there is a problem with the assembly, because, according to the requirement, only one piece can be in one position.

The answer to be sent to the interface is a string, of size 4, corresponding to the colors present in the spaces of the box, if there is no piece in any of the positions, instead of the colors, an "X" is inserted in the string which, when analyzed by the web application, is interpreted as an invalid value, therefore incorrect.

In this part of the process it is necessary to ensure that the connection between the

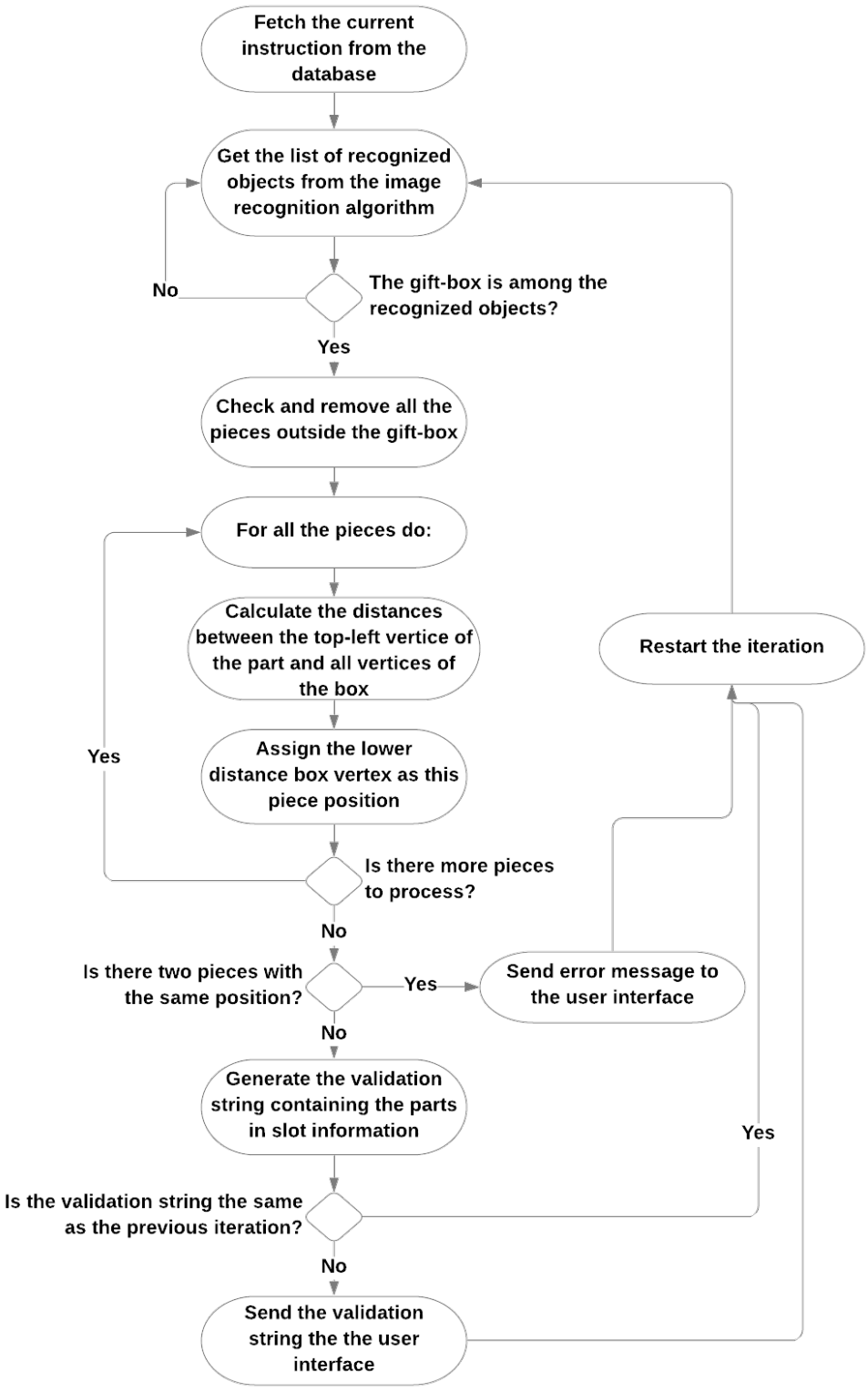

Figure 4.7: Gift-box verification algorithm.

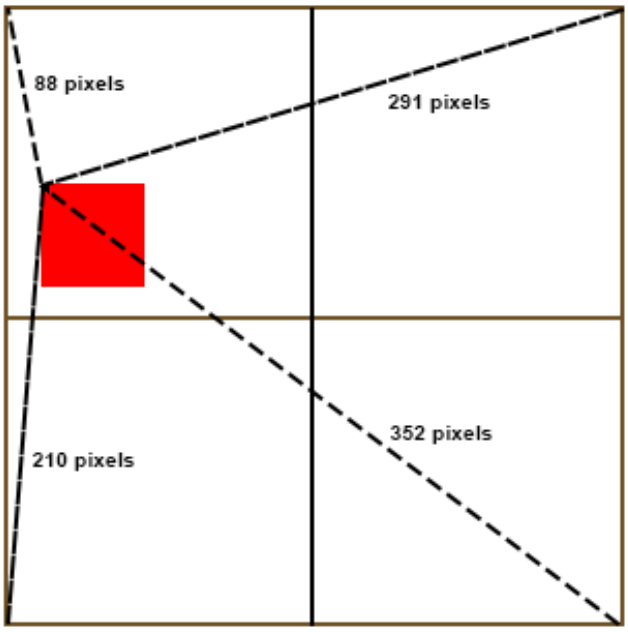

Figure 4.8: Example of a position classification through distance between the box vertices.

systems is not congested, as detection is always running, thus, the last message sent is stored and compared with the new response, if they are the same, there is no need to resend it.

Although the assembly is mainly validated by the web application, this function is also implemented locally. This is done in order to implement the behavior of generating a new instruction simply by placing an empty box after a complete assembly. When it is detected that the instruction has been carried out, the algorithm enters a routine waiting for the return of the recognition of only one box on the scene, with no pieces, and when this occurs, a reload command is sent to the web interface through the REST API, generating a new instruction.

4.4 Complex Assembly Verification Algorithm

When the assembly verification application is executed, the first action is to fetch the active assembly and search for the respective instruction file for the current step.

Since it is a three-dimensional mount, i.e. there is the possibility of parts being on

top of each other, it is important to know the current height of these objects, which is possible by subtracting the distance from the sensor to the table with the distance from the sensor to the part. However, not all points of the working area have a fixed distance, due to reasons such as surface irregularity and the camera assembly not being perfectly parallel to the table. To mitigate this problem, as soon as the application is started, an empty table frame is captured and the depth matrix is obtained, creating a height map that is later consulted to calculate the heights.

4.4.1 Assembly Files Structure

The files, describing the instruction to be checked, are composed of information and relationships of all the pieces that must be present on the scene through the JavaScipt Object Notation (JSON) format. To understand the verification algorithm, it is first necessary to describe how these instructions are generated.

A master piece is present in all instructions, being responsible for the basis of calculation to establish the position of the other pieces in scene. In this algorithm it has been decided that the master is the one in the top left corner in relation to the rotation of the mounting surface as illustrated in figure 4.9 by piece 2. In order that this piece, regardless of the addition of other components, is always the master, no other piece can be added so as to overwrite its state.

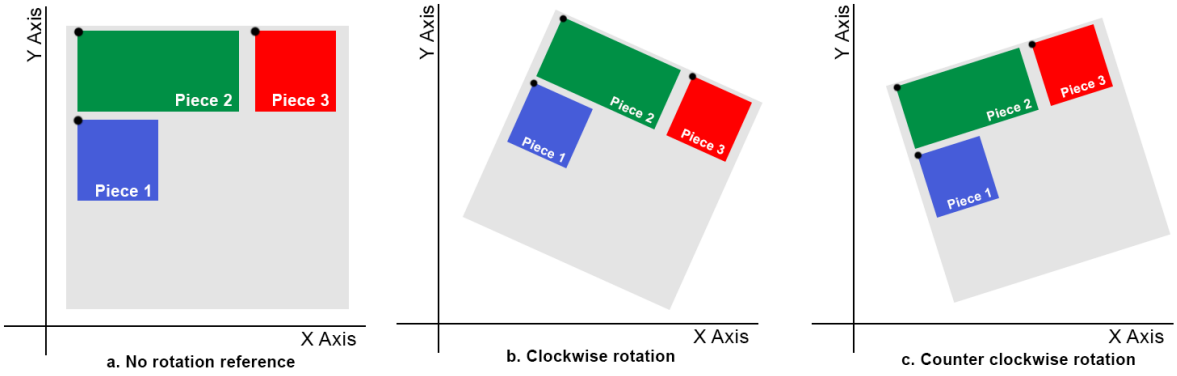

Figure 4.9: Possible assembly rotations.

So, in order for the JSON file to be populated, first the master must be found. Figure 4.10 presents the algorithm used to achieve this goal.

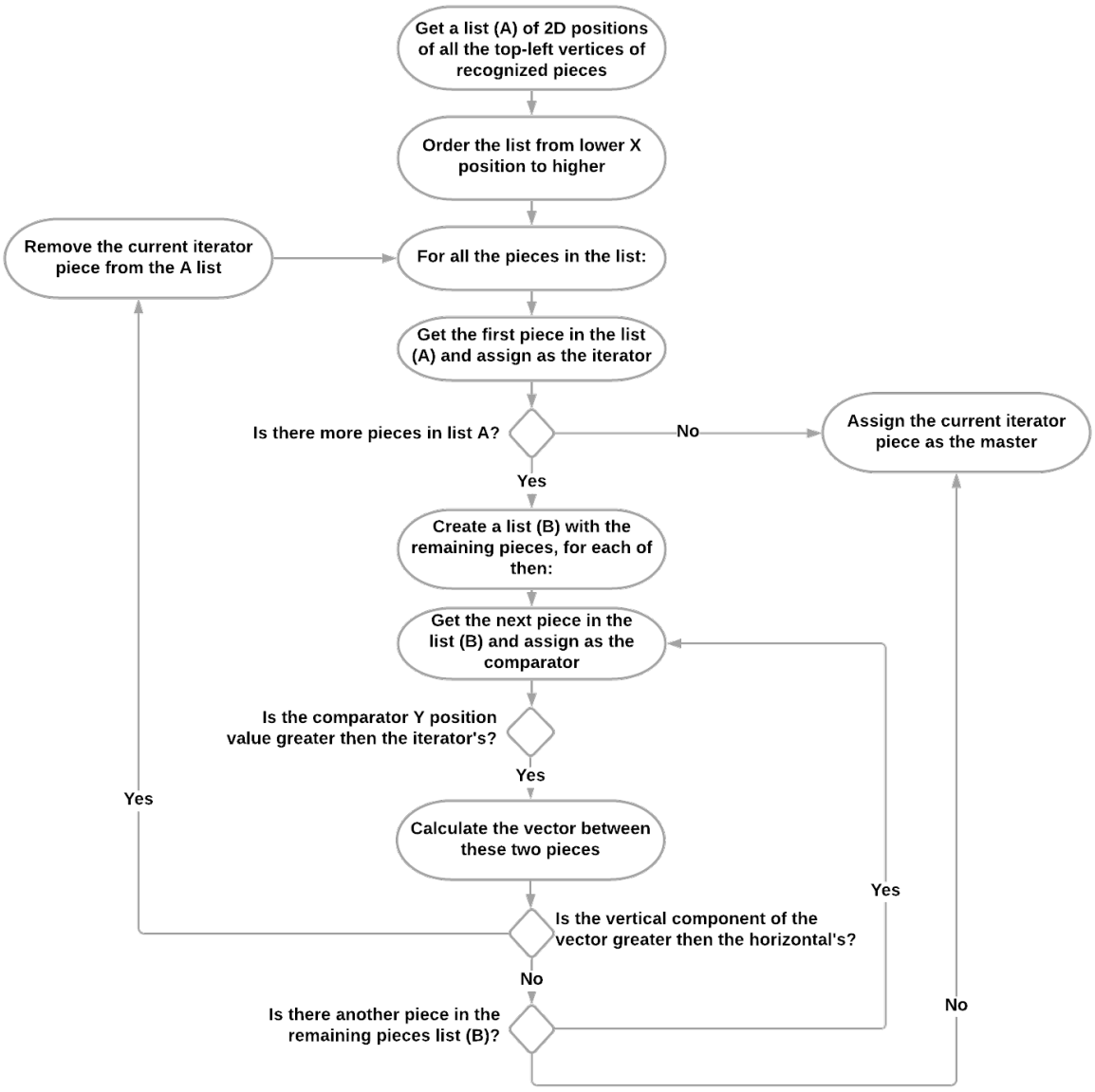

Figure 4.10: Master identification algorithm.

First of all, the list of the positions in relation to the horizontal axis is arranged from smaller to larger or from left to right. From the first member of the list, the remaining pieces are iterated to check if there are any with values greater than the current one on the vertical axis, i.e. a piece that is above it, and if found, the algorithm interrupts the

iteration and uses the found piece to repeat the same loop until there is no part that meets the requirement.

Having in mind case b. of figure 4.9, the algorithm starts from piece 1, checks the position of piece 2 and concludes that the vertical position is greater than its own, therefore the iteration is stopped. The piece 2 is selected for the second iteration, which compares it with the remaining pieces, in this case 3 which is below the current one and, since there are no more pieces to the right, 2 is determined as master.

However, when applying the same algorithm to case c. of figure 4.9, during the iteration with piece 2, piece 3 is found as vertical position superior and would eventually be defined as master, but when observing the arrangement of the assembly, the piece 2 is the one in the top left corner.

As a solution, a step is added in this algorithm. When a piece is found vertically above the current one, the vector of distance between the points is generated. The piece is selected only if the vertical component of this vector is larger than the horizontal component, because if the vector has a predominantly horizontal tendency, it is concluded that in fact the piece is beside and not above, and it will be ignored.

Applying the modified algorithm to case c we have: starting with piece 2 we verify piece 1, as its vertical position is smaller than the one of 2 we move to the next one. In relation to part 2, part 3 is above, when generating the distance vector, it is verified that it has a horizontal tendency, so it is ignored. As there are no more pieces, the 2nd is considered as master and, visually, it can be concluded that correctly.

After the master piece is found, all other pieces are iterated to establish their positions.

For each part, it is necessary to find a metric that characterizes its relation in the context of the assembly. For this, the application searches the shortest distance between a corner of the master and the piece in question. Figure 4.11 illustrates an example where this characteristic is found between the top right corner of the master and the top left corner of the piece. In this situation, another pair could be found between the bottom left corner and the bottom right corner. However, for the verification algorithm it would make no difference. The distance, the reference point on the master and on the piece are

filled in the JSON file.

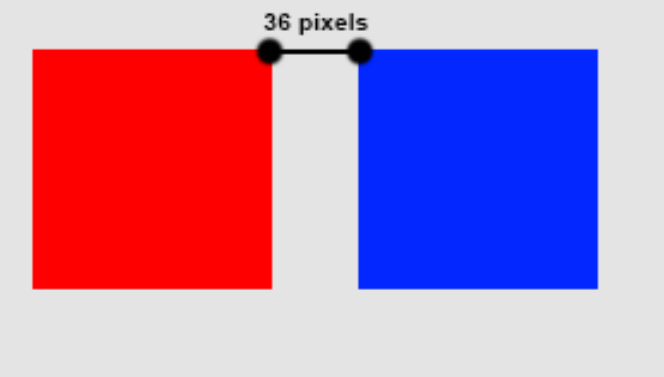

Figure 4.11: Lowest distance found between two examples pieces and its reference corners.

It is also necessary to fill in the color, orientation and shape of the piece in the file, these values are obtained directly from the image recognition algorithm.

However, it is not always possible to analyze all the parts on the scene, especially those that are covered by another part in a later step, causing the detection not to find all the objects and rendering the established relationship rules useless. To solve this problem, the assemblies were separated into layers of heights, each with a master piece to serve as a reference for other pieces in the same plane. A new layer is considered if any part is mounted on top of another and the number of the layer to which the instruction refers is recorded in the JSON file.

When a layer change is required to verify if the new piece is inserted correctly, the same calculation of distances and reference points is made with the previous master piece, however, a limitation has been introduced in the assembly freedom. It is necessary that the assembly area is not moved between this change, because the last position of the previous master part is stored, and, as it can be covered by the new part, it would not be possible, in this method, to search it again. In case the surface is moved, it is possible to return to the previous step to synchronize the positions again.

Another problem to be solved in the layer change is the possibility of the same distance being found for two different positions, illustrated in figure 4.12. For this, a new field is added to the file to check the trend of the distance vector between the two parts and,

according to the values of the vertical or horizontal components, this tendency is stored. The parts have specific positions where they can be placed on the mounting surface, leaving this particularity simple to be solved, or else the same distance would be found in an arc of infinite points around the master reference. Some examples of JSON instruction files are illustrated in figure 4.13.

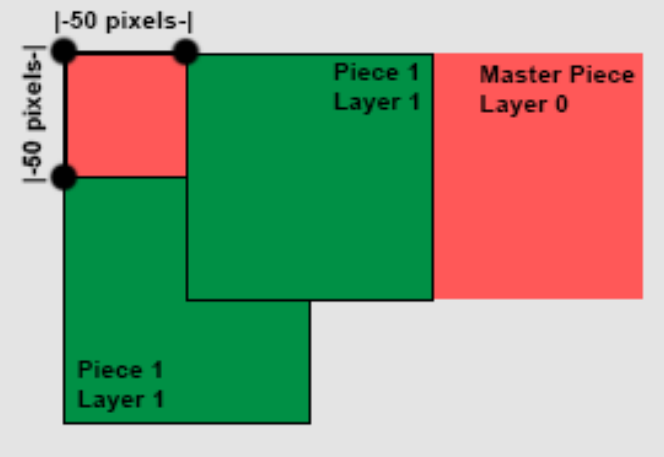

Figure 4.12: Example of two positions that could be found using the relations rules.


```
{
    "layer": 0,
    "assembly": {
        "0": {
            "color": "B",
            "master": true,
            "orientation": 1
        },
        "1": {
            "color": "R",
            "master": false,
            "distance": 3,
            "piece_ref": 0,
            "master_ref": 1,
            "orientation": 1
        }
    }
}
```

```
{
    "layer": 1,
    "assembly": {
        "0": {
            "color": "G",
            "master": true,
            "distance": 13,
            "movement": -1,
            "piece_ref": 1,
            "master_ref": 0,
            "orientation": -1
        },
        "1": {
            "color": "G",
            "master": false,
            "distance": 35,
            "movement": -1,
            "piece_ref": 0,
            "master_ref": 1,
            "orientation": -1
        }
    }
}
```


Figure 4.13: Examples of JSON files describing assembly steps.

To facilitate the creation of these instruction files a tool has been developed in Python. This tool does not have its own interface, but it uses the Windows console with instructions

and also a window illustrating all the detections performed to be verified. The user must enter the parts in the assembly and confirm, through the command line, step by step. After the complete assembly has been finished, the user enters the end command, in turn, the software generates all the files, with photos taken when a correct step was confirmed, and inserts them into the database as a new operation. This way, it is not necessary to create the JSON files manually, increasing the accuracy of the instructions so that there are no problems during normal operation.

4.4.2 Assembly Files Verification

When the JSON file is fetched, it is composed of a list of pieces containing the information of the relationships that make up an assembly, so, the first step is to compare the number of objects expected and the actual, if different, it is already possible to send an error message to the interface saying that it is not correct. Figure 4.14 presents this verification algorithm.

After this first confirmation, the application iterates over each instruction and searches all the detected parts for one that fits the required profile. It is important to note that the distance values present in the file are absolute and the probability that, for lighting reasons and position in relation to the camera, the values calculated by the recognition are not equal but close. Thus, a delta value at the expected distance is included, making the query within a range. For each piece in the instruction, at the end of the search, true or false values are added to a Boolean array. This vector has the size corresponding to the amount of pieces present in the step and at the end of the iteration, the assembly is correct if all its values are true.

To mitigate false positives in the analysis, this algorithm is executed five times and if and only if all five are equal, the answer will be sent to the server. This number of searches was found empirically, because if it is large, there will be a long delay between confirmations, and if it is small, the accuracy of the analysis is lower. Five searches was defined as a comfortable value of certainty and execution time. It is important to note

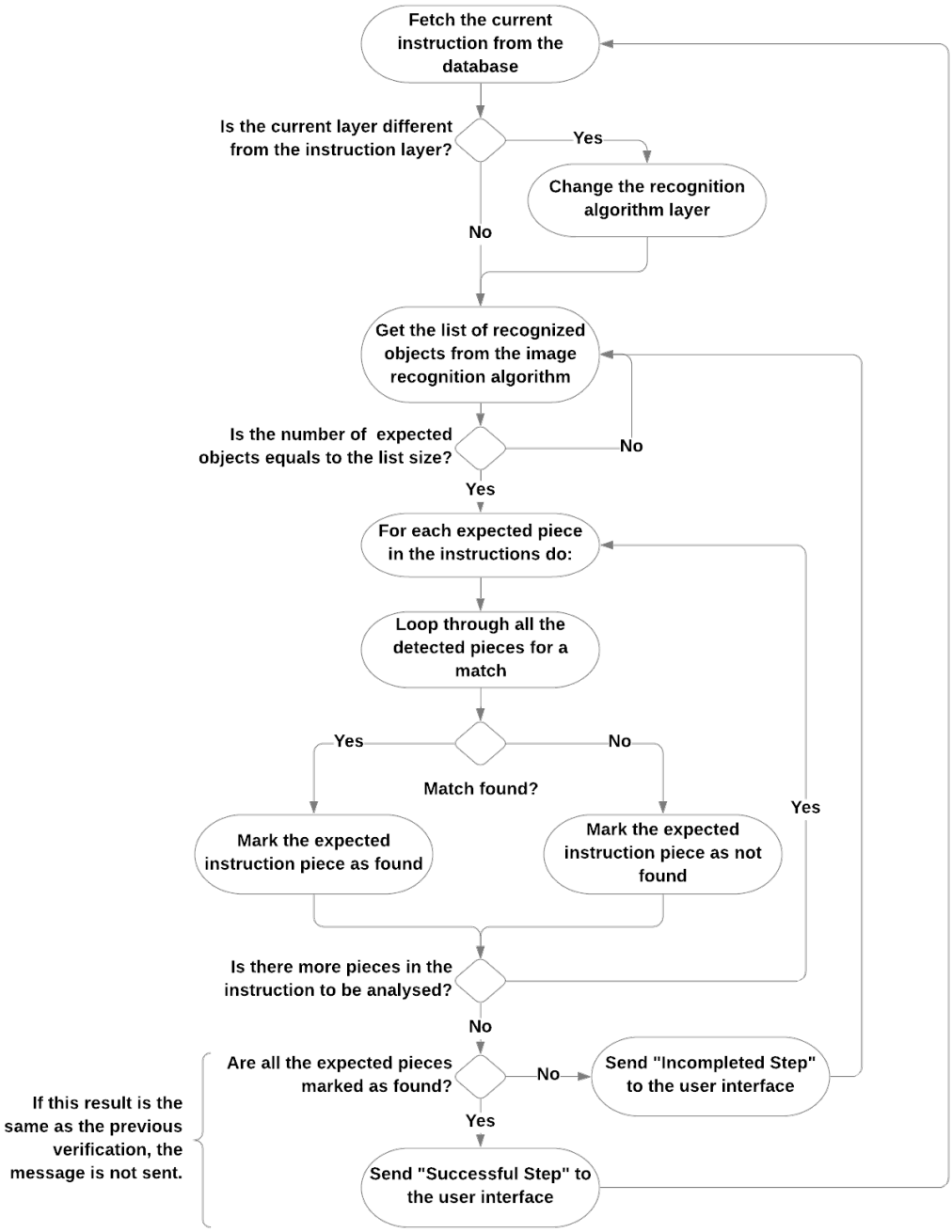

Figure 4.14: Complex product assembly verification algorithm.

that the same artifice against network overflow mentioned in topic 4.3 was used, with only messages different from the previous ones being sent.

As this algorithm has the memory of the previous instructions, it is possible to identify the piece included in the new step, this way, it is sent to the LED ribbon of the workbench, through a UDP connection, which piece is needed, helping the operator in the execution of the scenario.

Finally, if all five checks are correct, through the REST API of the interface, it is sent that the instruction was completed correctly, or else if not. If the instruction is correct, the application returns to the initial state in which it will fetch the next JSON instruction from the database to restart the verification routine.

4.5 Web User Interface

The web application is responsible for interacting with the user through controls and feedback of actions being performed during the scenarios. This application was developed through Laravel. Laravel is a framework for creating web-oriented applications using PHP as the language and was created with the principle of providing the possibility of developing applications quickly without the need for programming a complicated foundation, as the framework takes care of this.

The application backend is responsible for communicating securely with the database for query and data entry operations. These functions are exposed to the other components through a REST API, used as much by the frontend, for user interaction, as by the verification algorithm, to receive the state of the assembly to be performed.

The database in use is MySQL, containing three tables: "scenarios", responsible for the data of the gift-box assembly, "sequences", used to store the instructions of all the assemblies of the complex assembly scenario and "sequences_history", responsible for maintaining the current state also from the complex assembly. In table 4.3, 4.4 and 4.5 one finds the fields of the three tables, with explanations, respectively, in the order they were introduced.

Table 4.3: "Scenarios" database table with row explanation.

Row	**Description**	**Data example**
id	Identification number of this assembly	21
order	String representing the colors of this assembly	"G,G,B,R"
validation	String with four integers representing the correctness of the slots being 1 a correct color	"1,0,1,1"
is_valid	Status of this assembly, 1 for correct.	0
status	Field for the new assembly request to the interface, 1 for requested	0

Table 4.4: "Sequences" database table with row explanation.

Row	**Description**	**Data example**
sequence	Identification number of the assembly	1
step	Step identification of this instruction	3
image	Image URL of this current step	"storage/Scenarios/s1/1_3.JPG"
assembly	JSON instruction file of this current step	{"layer": 0, "assembly": {"0": {"color": "G", "master": true, "orientation": 0}}}

Table 4.5: "Sequences_history" database table with row explanation.

Row	**Description**	**Data example**
sequence	String representing the correctness of the assembly steps	"1,1,1,0" (3 correct, 1 incorrect)
status	Status step, 1 for correct	0
currentstep	Current step being performed	3

The interface is divided into three pages: main, gift-box assembly and sequence operation. The main page is the first to be presented to the operator starting the application, and is responsible for presenting two training options to be chosen. Each option has its own layout and logic to form the behavior determined by the requirements of each scenario

4.5.1 Gift-Box Assembly User Interface

As explained in section 4.3, when the operator starts this scenario by selecting it on the main page, a random sequence of colors is generated as an instruction in the backend of the application, inserted into the database following the table 4.3 rules and used as parameters passed to the renderer so that these colors are shown on the page.

This page is composed of four squares, corresponding to the four spaces in the box, which have their colors representing the instruction to be followed by the operator. Also, in these squares, when the assembly is processed, symbols are superimposed to identify if that position is with the right piece. Finally, in the upper part there is a feedback text, to provide pass textual information about the current state of the scenario, and in the lower right part a button to generate another instruction, if necessary. The figure 4.15 demonstrates the interface and figure 4.16 presents the main update algorithm.

After being rendered, it is necessary to update and show the assembly information, for this, the code performs a loop every 100 milliseconds. At each loop, a call is made at the API through "GET /checkScenario" that returns a Boolean array representing the colors of each piece in the box. From this return, the algorithm checks each color in the interface and superimposes a confirmation or denial to represent if that position was properly assembled. If all the options are correct, a success message is given, which is then modified to inform that the user must insert a new empty box to generate a new color sequence. If one or more positions are incorrect, an error message is given. Figure 4.17 illustrates the interface behavior, given a certain response.

To define the behavior of being able to generate a new scenario by placing an empty box in the scene, the interface must be able to detect this request. Since it is not possible

Figure 4.15: Gift-box assembly user interface.

to send messages directly to the frontend, the database fields are used to simulate this feature.

When verifying that all colors have been correctly populated, the interface performs, through the API, a request "POST /setScenarioStatus", causing the field "is_valid" in the database to be modified to true, or 1. This check is also being done simultaneously in the verification algorithm which, when confirming the assembly, goes on hold for an empty box that, when detected, a "POST /updateScenario" is sent to the web API. This method checks if the "is_valid" field is true, and if, only if it is, it changes the "status" field to true. The interface, when requesting the assembly state, with the "status" field true, receives a command to update the page, causing a new color arrangement to be generated.

4.5.2 Complex Assembly User Interface

When selecting this scenario, internally a random assembly is chosen and registered in the database. Before the page is loaded for the first time, the controller query the "sequences" table for the first step of the current assembly in order to obtain the images that should

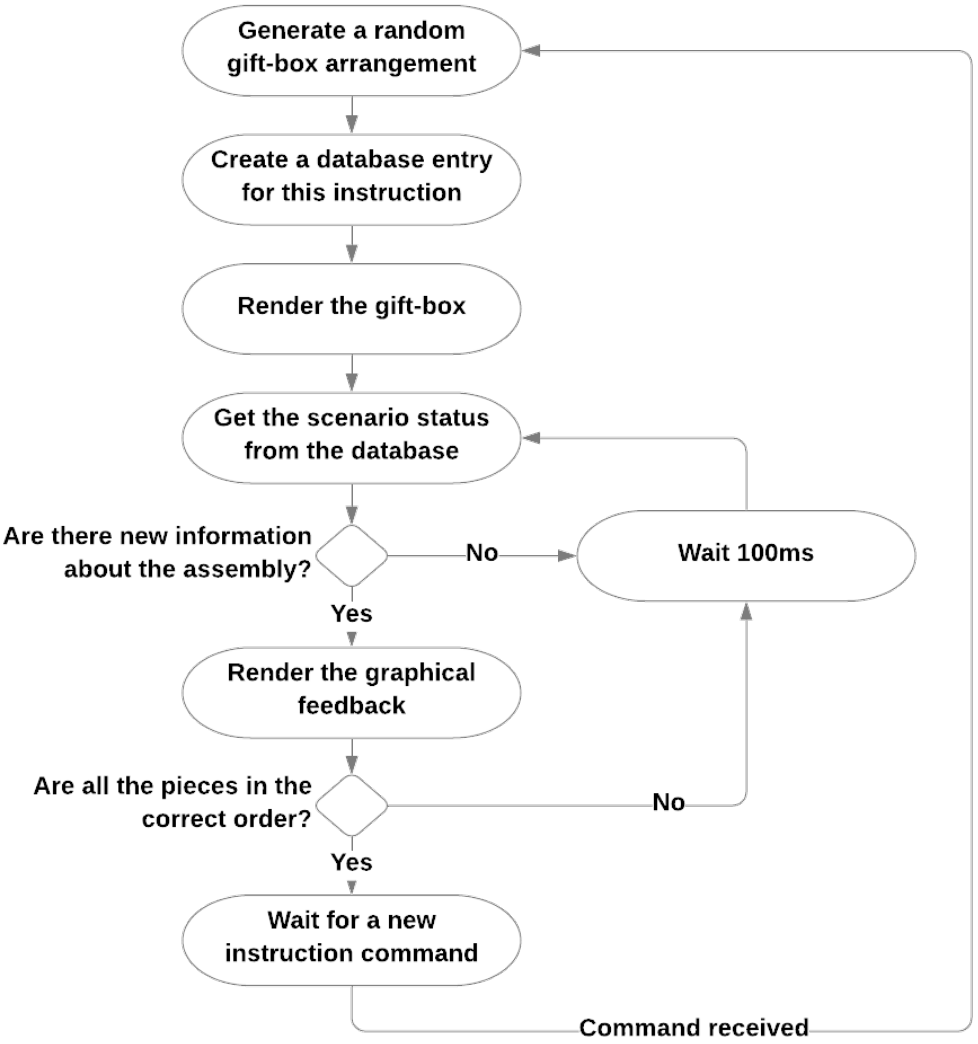

Figure 4.16: Gift-box assembly interface main update routine.

Figure 4.17: Gift-Box assembly interface feedback.

be displayed. This interface consists of pictures of the complete assembly and current step to be performed with visual instructions. As in the interface described in the previous section, there is a feedback panel for error or success messages, and a button to generate a new mount, if necessary. Figure 4.18 illustrates the components of this page and figure 4.19 presents the main update algorithm.

After rendering, the interface enters a routine similar to the one described in the previous section. A call is made to the API through "GET /checkSequence", returning three possible states: incorrect assembly, correct assembly with next step and fully correct assembly. These states are calculated from the data of the "sequence_history" table, updated by the verification algorithm. The "sequence" field of this table contains a string of integers that follow the structure shown in figure 4.20. If there is a 0 in the string, it indicates that there is an active step not completed and respectively the opposite otherwise. On the condition that all the steps given are already completed, it is necessary to know if there is more to be done, for this, the next hypothetical step number is calculated

Figure 4.18: Complex product assembly user interface.

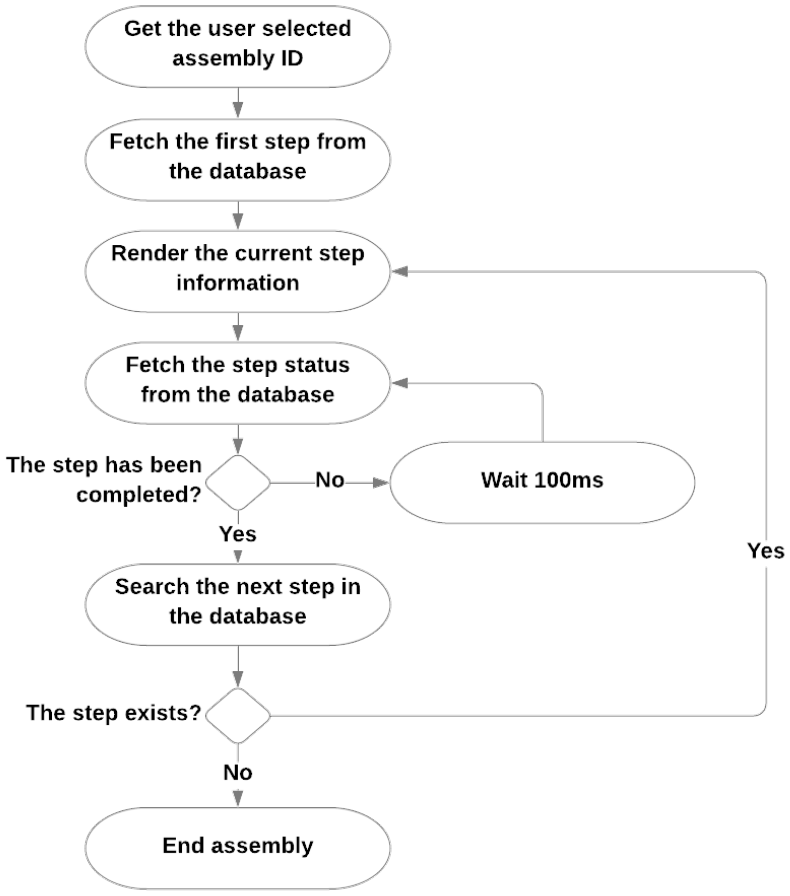

Figure 4.19: Complex product assembly interface main update routine.

and searched in the table with the instructions. Depending on this query returning valid results, the system renders the page with the figure of the next stage or generates a new assembly.

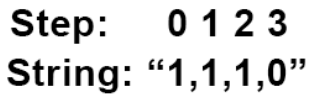

Figure 4.20: Validation string structure.

4.6 Virtual Reality Application

For the development of the scenario in virtual reality it was used the Unity3D game engine. This software was chosen for the amount of educational resources available on VR integration, support for the equipment employed and the author's previous experience, eliminating the need for the platform to be learned from scratch in order to achieve the project requirements.

To fulfill the immersive feature of VR training, it is necessary that the scene, a three-dimensional environment where the user interacts, is designed to look as close as possible to the real situation, described in section 4.3. For the composition of the scene it was used a mixture of 3D models preexisting in Unity3D and models made in Blender [74], this last a free and open-source modeling software. In figure 4.21 the composition of the scene is presented.

In front of the user is present the mounting box arranged on top of a workbench. On this workbench there are also three containers, separated according to the color of their contents, where there is a single piece for assembly. When the exit of a part from its container is detected, another one is generated in its place, decreasing the amount of memory required for rendering the scene if several parts were placed in the environment. Finally, the feedback monitor shows the current instruction, with confirmation of the correctly assembled slots and a general status message.

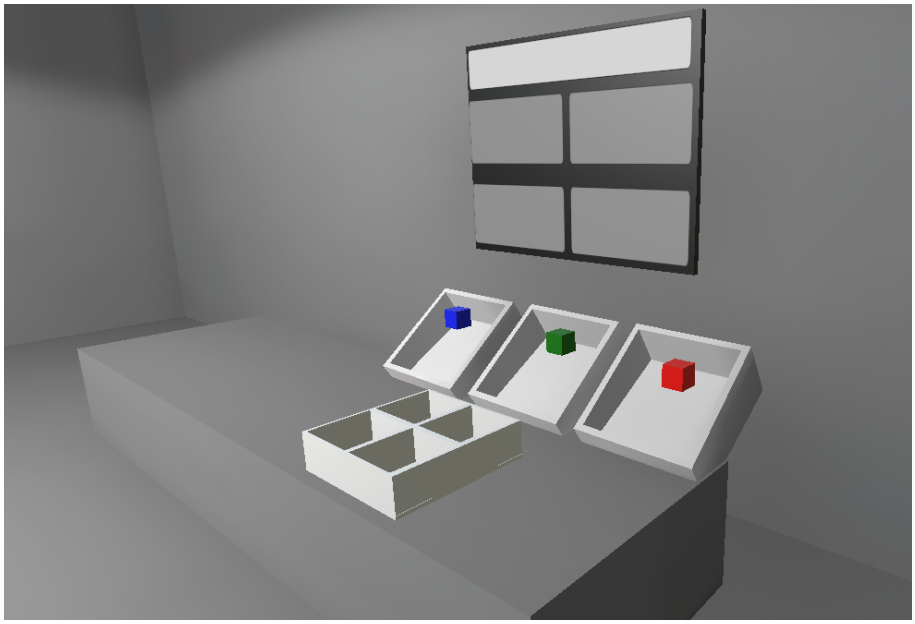

Figure 4.21: Virtual reality main scene.

The architecture of the virtual reality system is presented, in summarized format, in figure 4.22.

The assembly instruction is random and automatically generated at the beginning of the session and, for recognition, there are detection areas in each gift-box slot, demonstrated in figure 4.23. Through the Unity3D physical engine, it is possible to identify part collisions with the detection areas and retrieve the object's tag, previously assigned according to its color, and sent to the control script that compares with the expected tag and updates a status list that represents the assembly's accuracy. Finally, this control script sends the aforementioned list to the interface control to update the information shown to the user.

The VR device selected for integration with this simulation was the Oculus Go for its availability during development and for being a standalone device, that is, it is not necessary to connect it to a computer to run the simulation, as it is integrated with an Android embedded system with its own internal storage.

The adaptation of the project in Unity3D is carried out using the libraries and guides provided by the manufacturer of Oculus. These libraries offer ready-made codes that

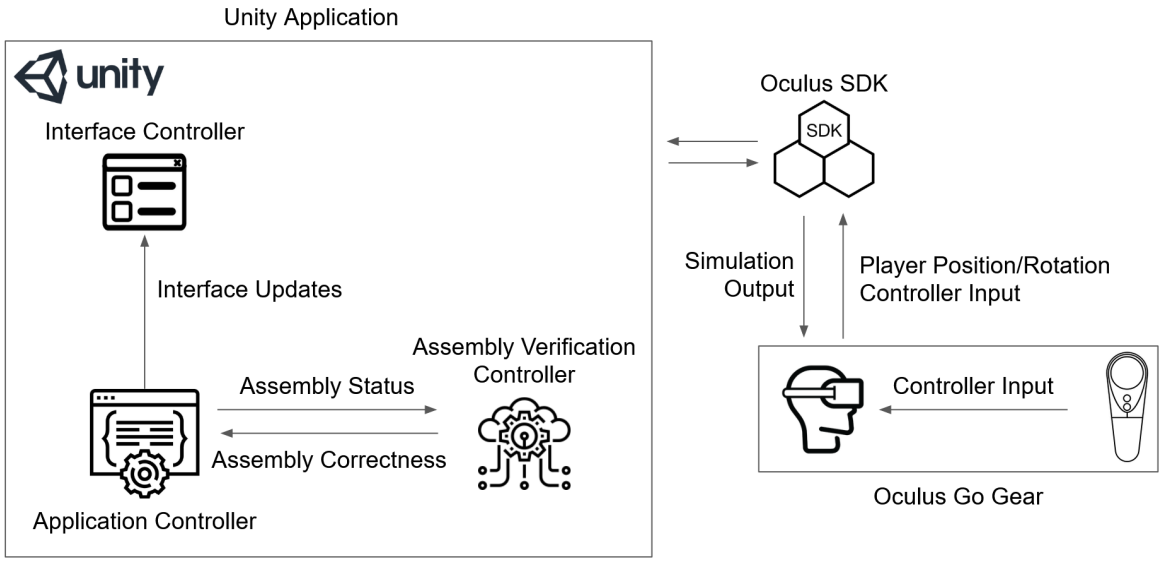

Figure 4.22: VR system architecture.

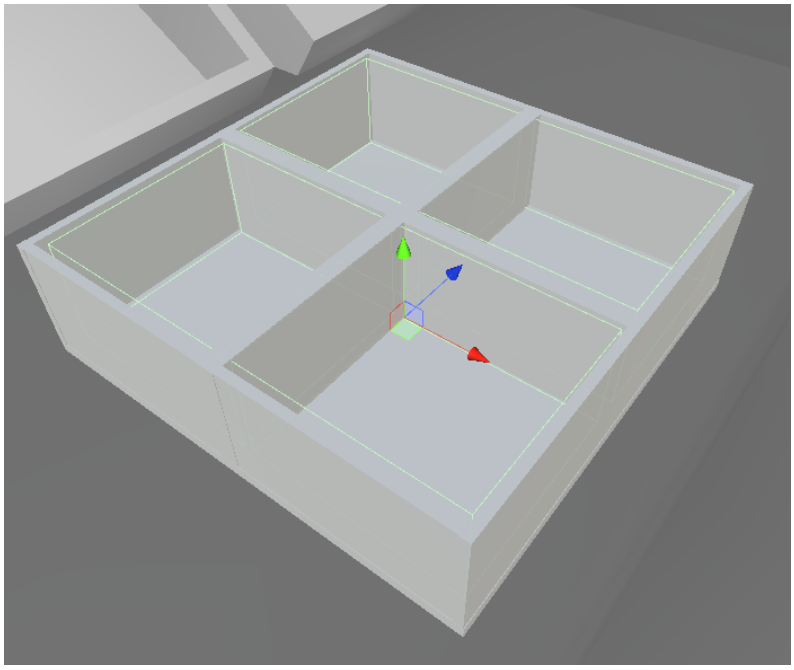

Figure 4.23: Gift-Box slot detection areas (outlined in green).

make it simple to transfer from a normal scene to a VR scene, performing tasks such as moving the camera according to the user's vision and individually rendering each eye on the glasses automatically. Because the VR device uses a different operating system, it is also necessary to install and connect the Android Software Development Kit (SDK) with the project for the final compilation.

As user input for interaction with the scene, the controller included with Oculus Go is used. Through the library, the position and rotation in relation to the user's own position and button activations are automatically detected, being only necessary to assign the actions in the Unity3D user input system. The Oculus library also provides a three-dimensional model to represent the control in the simulation. Figure 4.24 shows the settings of these buttons in the controller.

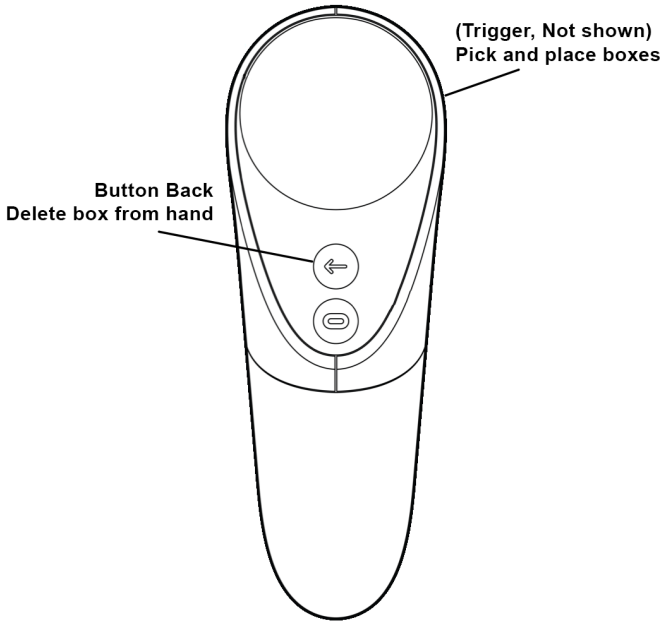

Figure 4.24: Oculus Go controller functions mapping.

To interact with the parts the user must point the control to select it, and this behavior is performed through the raycast system of Unity3D. Raycast works by casting a ray in one direction, in this case the direction in which the controller is pointing, and returns the first object on which this ray collided. In this way, when pointing at a box, it is possible to "pick it up" and anchor it with the position of the controller, as it was being holden.

By having this piece anchored, the user has the possibility to point the controller to the gift-box and insert it or discard the piece, freeing its "hand".

Without any visual help these tasks can become difficult, as it is not always possible to infer which piece you are pointing to. This problem is solved by rendering a visible line that follows the same position and direction as the detection raycast, creating a laser pointer. Also, to improve the user experience, when having a part in "hand" and pointing the controller to the gift-box, a representation is shown of where the part will be when placing it. Figure 4.25 shows the control's help line and figure 4.26 demonstrates the future representation of the pieces in the gift-box.

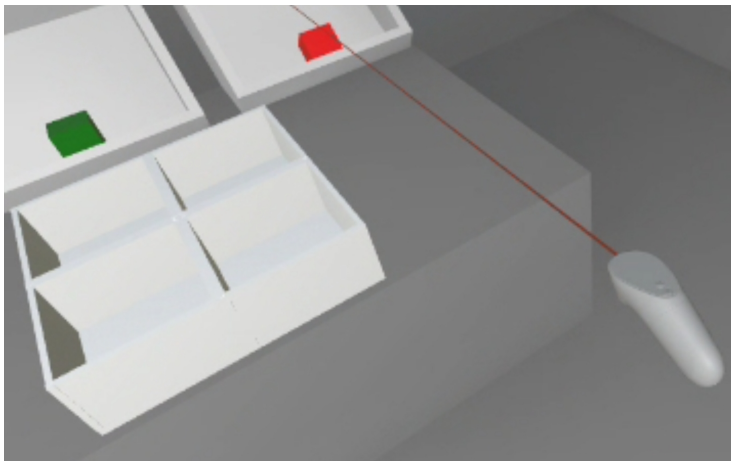

Figure 4.25: Controller pick and place trajectory helper.

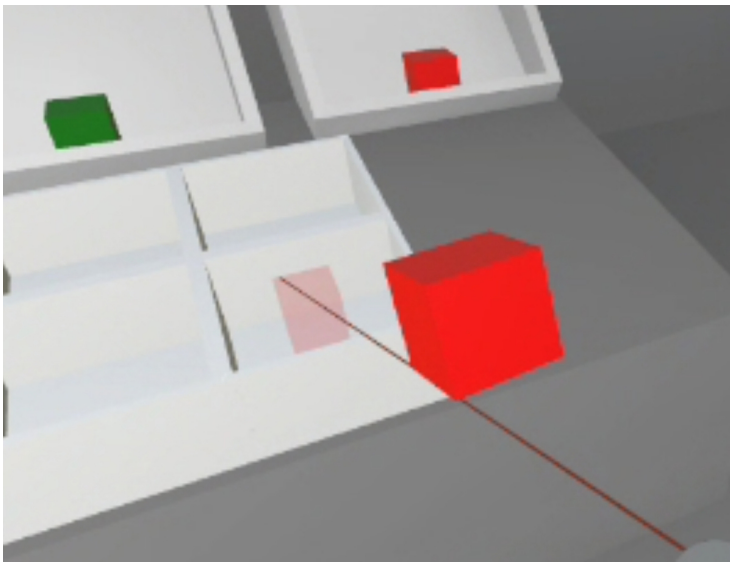

Figure 4.26: Piece future position representation at the gift-box slot.

Chapter 5

Results and Discussions

The results are divided among all the developed technologies that cover the objectives of this work. Performance, tests and weaknesses are presented followed by a discussion about the fulfillment of these objectives and possible improvements detected during the evaluations.

5.1 Artificial Vision Algorithm

The artificial vision algorithm was tested alongside its applications in both workbench scenarios, so from its use, it is possible to infer its performance and failures through use in real cases.

As explained in section 4.2, multiple checks were incorporated to identify only the objects pertinent to the exercises. After the platform execution, it is possible to notice that these checks fulfill their role because no false positives were obtained, be it of color or incorrect classification.

However, it was also noticed that some pieces were being excluded because they were below the lower limit of required area. This problem was corrected with a new collection of area parameters and, as in the algorithm, cases like this were already thought ahead, these new parameters were easily injected into the code. After this modification, no part was excluded by area verification.

To help the gift-box assembly scenario, a check was incorporated so that the recognition returns only the parts that are inside the box, this one also detected. This behavior was correctly tested and confirmed with the need to have only one box in the scene. As in the scenario the presence of two boxes is considered incorrect, this was not classified as problematic.

The biggest problem found during the assembly detection was its position in relation to the table. On the workbench there is a illumination directly above the work area and, because the parts are reflective, in the image capture, the parts are completely or partially saturated, not being possible to recognize them, shown by figure 5.1. Some solutions can be added to mitigate this situation.

Figure 5.1: Reflected light (left) causing obstruction of the filter mask (center) resulting in failure to detect the blue piece (right).

In the context of physical solutions, it is possible to rearrange the workbench lighting and add diffusers so that it does not cause unwanted reflections. On the software level, a solution would be the reconstruction of detections according to the neighborhood of the objects, however, this would only be possible in cases of partial and not total reflection, and the reconstruction time would increase the total time of detection of the assembly. The not ideal solution employed in this case was to delimit the working area within sections of the table where the lighting angle does not cause reflections.

Finally, the classification of the heights of the pieces has been proven efficient by using a height map, because, through the debugging of the algorithm, it is possible to find the same height for pieces in the same plane. Since a height map technique is used, it is added the requirement that the algorithm be started with no object above the table, otherwise

it is necessary to clean it and restart the detection. Also, if the camera is misaligned perpendicularly to the working area, incorrect heights are obtained, so it is necessary to perform this verification ahead.

5.2 Gift-box Assembly Operation

To validate the gift-box checking algorithm 50 consecutive runs were performed including incorrect assemblies with the following situations:

- Incorrect color order within the box.
- Two pieces in the same slot.
- Pieces outside the box.

During these tests no false positives were obtained. However, as explained in section 5.1, when the box was outside the detection area some false negatives occurred for reasons of light reflection.

A particularity identified in this exercise was the incorrect detection of red pieces when very close to the box wall. It is observed by the debugging of the artificial vision algorithm that, in some situations of low brightness, the HSV values for brown and red coincide, thus causing that the filter cannot separate the two objects. Therefore, it is necessary to improve the lighting layout and fine tune the filter values to better separate them.

A timed operation test was not performed in this case because, in initial validations, it was noticed that a larger number of available pieces and possible combinations would be needed to obtain a significant difference between the times when using conventional methods and this IPA. However, the results of this application are more in the qualitative than the quantitative scope. Its advantages in using it to assist the user during customized assemblies include the automatic presentation of the required products and the verification of the operations being performed, excluding the possibility of human errors coming, for

example, from fatigue, thus increasing the efficiency of the assembly line and general comfort of the operator.

Figure 5.2 presents an operator performing a instruction as well the algorithm output and figure 5.3 shows the steps performed by the visual algorithm do detect the parts.

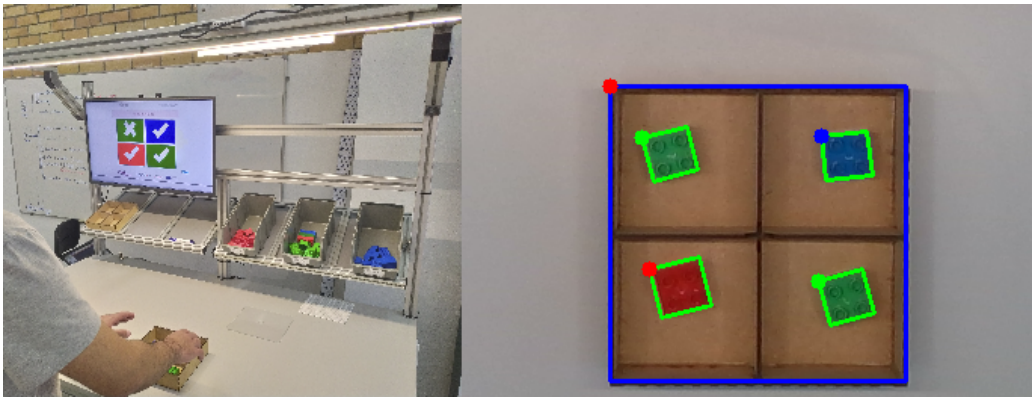

Figure 5.2: Gift-box assembly in the IPA (right) with the recognition output (left).

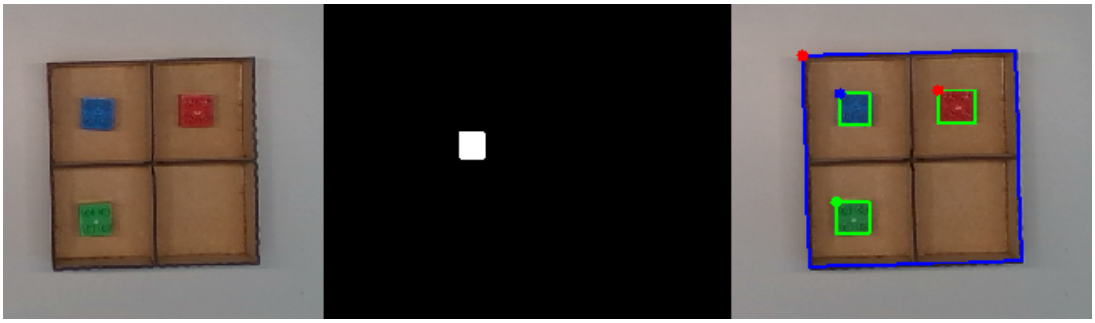

Figure 5.3: Image recognition output with the original image (left), HSV filter threshold (center) and final results (right) for the blue piece.

5.3 Complex Assembly Operation

To validate this use case, tests were performed with 4 different assemblies, 5 times each, being 2 of these assemblies with the instructions filled manually, that is, the JSON file being created from the measurement of the relations by a human, and the others filled automatically by the tool presented in section 4.4.1.

As with the test carried out in the previous section, erroneous situations were presented to identify possible false positives in the recognition of the steps. The following situations were introduced:

- Switch between color, position or wrong orientation.
- Parts mounted in incorrect layers.
- Movement of the mounting base in instructions of the same layer.
- Removal of previous parts.

Table 5.1 presents the results of the tests above for 4 different assembly instructions performed 5 times each, with 2 being filled manually and 2 created by the JSON tool described in section 4.4.1.

Table 5.1: Complex assembly tests for 4 assembly instructions.

	Assembly 1	**Assembly 2**	**Assembly 5**	**Assembly 6**
Assembly files type	Manually created	Manually created	Created by software	Created by software
Number of steps	5	4	5	4
Fully detected in first try	No	No	Yes	No
Non-detections (total)	18	5	0	2
False detections	0	0	0	0

As expected, the steps performed with manually filled instructions showed a higher rate of non detection than the automatic ones, and in one of the assemblies the files had to be created again with new parameters. However, during the tests, no false positives occurred.

One problem found was the insertion of parts of the same color as other parts in lower layers. Although the performance of the algorithm when ignoring the layers not being worked on was satisfactory, a behavior of pieces of inferior parts being included in the proximity with the parts in the superior layer was detected, observed in figure 5.4.

Thus, when the colors of these pieces coincide, they are considered one piece only, causing the incorrect calculation of their edges and consequently unable to be confirmed by the algorithm.

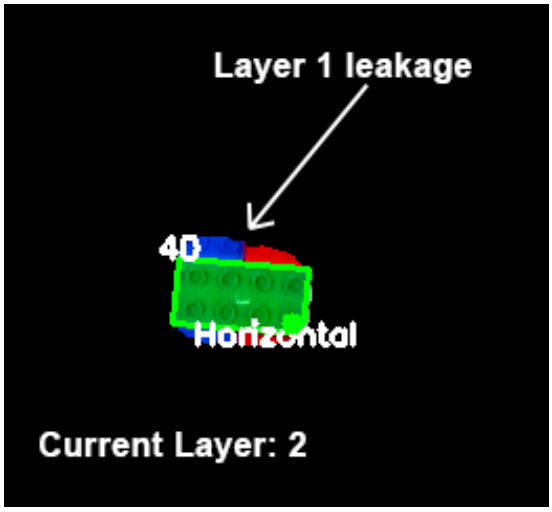

Figure 5.4: Wrong inclusion of bottom layer parts.

It was also proven the theorized problem during the development of this application in which during the change of layers, if the parts change their position on the workbench, it is not possible to obtain the relation of the new part and therefore it is not possible to confirm the step. To solve this problem it is necessary to use another type of algorithm that can identify the position of the parts, independent of the relations between the rest of the assembly.

To perform the execution time tests, an assembly has been selected obtaining the setup duration (platform initialization and assembly selection), time per step and finalization (in this case the report will be considered automatic by the system). In order to compare with a conventional method, another assembly with the same difficulty and number of steps was presented to the operators with paper instructions. In the latter the user must fill out a report containing initial data considered as setup, confirmations of each step performed with observations, if necessary, and finally, signature. Table 5.2 presents the result of both tests.

It can be observed that the biggest difference between the two methods is related to

Table 5.2: Average elapsed assembly time of IPA and paper instructions.

Time Elapsed (s)	Conventional Method	IPA method
Setup	84.25	15.75
Step	18.88	13.25
Total	182,00	77.75

the setup time, which is composed by the selection of the assembly procedure and the pre-filling of the report, where, in the IPA, the information would be filled automatically and uploaded to the database accordingly. The time taken to complete the step when performed on the workbench shows a reduction of 29.8% in relation to conventional methods. Through the use of assembly recognition associated with the IPA method, it is ensured the certainty that the instruction was correctly executed, while in the conventional method this function is up to the operator, thus introducing the possibility of human error.

Figure 5.5 shows the user performing a workbench operation and an example of the recognition algorithm output. The black points present in the algorithm's output are due to the layer system because, as the distances are subtracted from the height map, it is possible to obtain negative values by oscillating the camera data, causing the algorithm to exclude these areas.

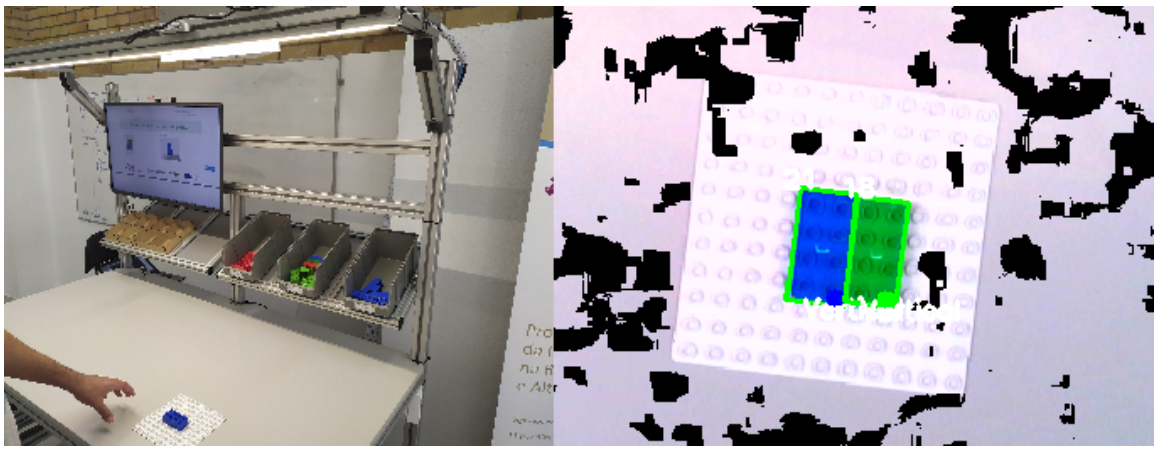

Figure 5.5: Operator performing a complex product assembly (left) and a recognition algorithm output example (right).

5.4 User Interface

During all the tests performed in sections 5.2 and 5.3, the interface behavior for possible problems was also observed. The browser used during the test was Google Chrome and no errors were detected. However, when using Mozilla Firefox, the interface was not updated correctly and when inspecting the web console, it was discovered that calls to fetch the software status were not being made. Unfortunately the cause of this anomaly was not found, but it is theorized that it is in a blocking by the browser of multiple calls in quick succession.

5.5 Virtual Reality Application

As this application is a simulation of the gift-box assembly and performs the detections by code and not by cameras, no tests were performed with the characteristics of section 5.2, as there is no possibility of them happening. The tests performed have the objective of finding possible bugs and erroneous behavior by the software or the VR device.

The tests involved 50 consecutive tasks, observing the interface behavior, the fidelity of synchronization of user movements with those mimic by the simulation and possible crashes.

When the first testing routine was started, it was noticed that the glasses screen was flickering, causing nausea during use. After further inspections, it was determined that the Unity3D's output refresh rate was incorrect, 60Hz, compared to that required by the VR device, 75Hz. After modification of the parameters this problem was corrected.

It was also detected that, in rare situations, the pieces got stuck in the box when they were selected from the container for manipulation, illustrated in figure 5.6. For the animation of the pieces to leave the container and go towards the user's hand, they are physically moved in the scene, therefore, depending on the user's angle, it is possible that the animation path passes through the gift-box and the system considers that it was being placed in one of the slots, causing two pieces to be together. To solve this

problem, a condition was added that the piece can only be placed in one slot if it is leaving the user's hand. This solved the problem of the verification system considering it when unintentionally placed, but visually, the part is still showed stuck. The visual problem has not been solved, but it does not influence the confirmation of the task and it can be unstuck using the "remove part" button of the control.

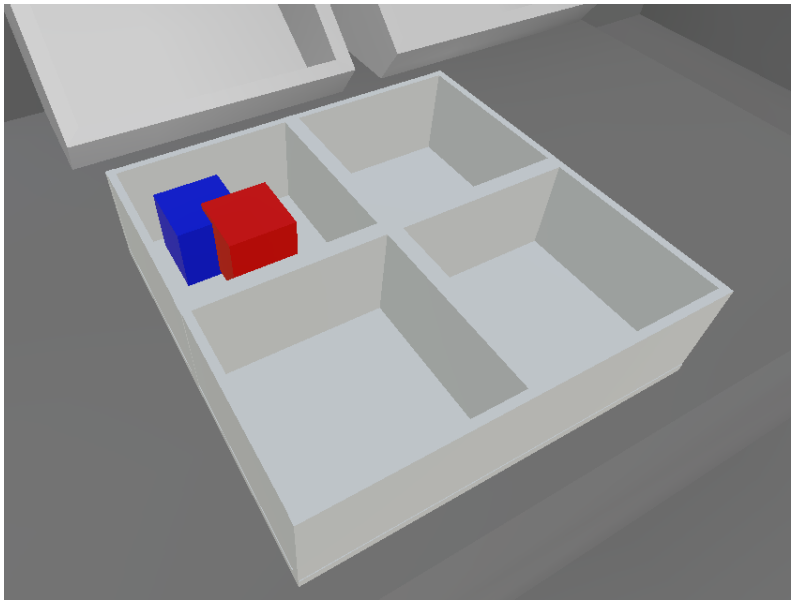

Figure 5.6: Two pieces in the same slot.

Disregarding this error in rare situations, the simulation normally occurred within the tested parameters, interpreting the user input, recognizing the assembly and presenting the instructions in the feedback panel successfully.

Figure 5.7 presents the vision of a user performing an assembly in the simulation.

5.6 System Usability Survey

A questionnaire was applied to measure the user experience while using the IPA workbench and the VR application. This is derived from the System Usability Survey (SUS) [75], chosen because it is a reliable and tested method to measure the usability of computer applications. The questionnaire, illustrated in Table 5.3, is composed of 10 statements, 5

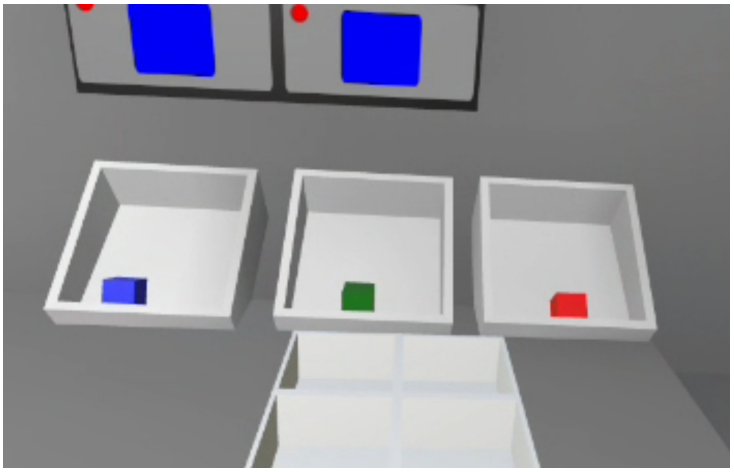

Figure 5.7: Operator point of view while performing a gift-box assembly in the VR environment.

in positive tone and 5 in negative tone, and the user should rank according to the scale from strongly disagree to strongly agree. The usability test was applied, in 6 operators for the workbench and 6 for the VR application, starting from the completion of some tests followed by the filling out of the questionnaire.

Table 5.3: SUS 10 items questionnaire.

I think that I would like to use this system frequently
I found the system unnecessary complex
I think the system was easy to use
I would need the support of a technical person to be able to use the system
I found the various functions in the system were well integrated
I thought there was too much inconsistency in the system
I think that most people would learn to use the system very quickly
I found the system very cumbersome to use
I needed to learn a lot of things before I could use the system
I felt very confident using the system

The SUS for the IPA workbench was conducted considering both scenarios and the average achieved SUS score was 88.13. Since according to the SUS classification, a score higher than 80.3 is considered excellent, the achieved results were very satisfactory. The lowest individual scores were related to previous knowledge and need for supervision, which may be due to the lack of an explanation of how to use the tool, and can be solved by using a tutorial to explain the interface.

As for the VR application, the survey was looking for information about possible problems during the use of the system and general feedback about the simulation and integration with the VR. With a 92.5 grade being obtained, considered excellent, the lowest scores were for the statement "I think that I would like to use this system frequently". This is consistent with comments received regarding discomfort in using VR glasses for a prolonged time, causing headaches and muscle distress from the extra weight of the device.

Chapter 6

Conclusions

This thesis presented an approach to support the integration of humans in cyber-physical systems. The application of a workbench equipped with an intelligent personal assistant demonstrates, through two different scenarios, how emergent ICT technologies can be applied to help operators to perform assembly tasks in a faster and more efficient way.

Some changes are necessary in the detection algorithm of section 4.4 to solve the limitations presented in the results. It is also necessary to carry out research to improve image recognition in order to solve the reflection problems of the parts, be they physical or software solutions from filters and reconstruction of lost areas.

Several aspects can influence the use and confidence of the IPA technology, e.g., usability, security and privacy. At the practical level, there are some challenges still to be faced regarding supporting technologies for IPA applied to industrial environments such as the implementation of intelligent sensors capable of better interacting with the user, artificial intelligence and advanced user interfaces, achieving a synchronized symbiosis between operation and operator to further increase efficiency. Solutions need to be matured to avoid creating entropy within the maintenance process and operators need to be adequately trained to operate the system in a proper manner. The IPA's human-interaction barrier models need to be improved, and AI systems need to prove reliability.

In parallel, using VR technology, the virtualization of the workbench was presented through Unity3D and Oculus Quest. This allows the sessions to be carried out in an

immersive environment through a smaller equipment and with a relatively smaller price in comparison with a full workbench, with no concern of damage to the components handled during the training.

Another aspect is related to the ergonomic evolution of the head mounted devices, which need to be more comfortable for operators to use during the entire shift or even complete the maintenance intervention.

Future work includes the use of more robust detection algorithms to handle more complex parts e.g. motors and hardware, and AI techniques to provide best action plans along the assembly process. Regarding the virtual reality application, future work must develop more sophisticated scenarios, such as assembling complex products, possibly using more robust controllers that can detect both the user's hands and allow the detection of the finger movement to bring the simulation behaviors closer to the scenarios to be simulated, increasing immersion.

Also, further development is needed in the range of AR applications, enabling the deployment of an IPA in tasks such as valve inspection and obtaining real-time information on equipment to assist in maintenance operations.

Bibliography

[1] A. Gilchrist, *Industry 4.0*. Jan. 2016. DOI: 10.1007/978-1-4842-2047-4.

[2] ——, *Industry 4.0: the industrial internet of things*. Springer, 2016.

[3] P. Leitão, A. W. Colombo, and S. Karnouskos, "Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges," *Computers in Industry*, vol. 81, pp. 11–25, 2016.

[4] Mckinsey, *Jobs lost, jobs gained: What the future of work will mean for jobs, skills, and wages*, https://www.mckinsey.com/featured-insights/future-of-work/jobs-lost-jobs-gained-what-the-future-of-work-will-mean-for-jobs-skills-and-wages, Accessed: 2020-09-16.

[5] G. Schirner, D. Erdogmus, K. Chowdhury, and T. Padir, "The future of human-in-the-loop cyber-physical systems," *Computer*, vol. 46, no. 1, pp. 36–45, 2013.

[6] P. Fantini, G. Tavola, M. Taisch, J. Barbosa, P. Leitão, Y. Liu, M. S. Sayed, and N. Lohse, "Exploring the integration of the human as a flexibility factor in cps enabled manufacturing environments: Methodology and results," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2016, pp. 5711–5716.

[7] S. Hoedt, A. Claeys, H. Van Landeghem, and J. Cottyn, "The evaluation of an elementary virtual training system for manual assembly," *International Journal of Production Research*, vol. 55, no. 24, pp. 7496–7508, 2017.

[8] A. Barthelmey, D. Störkle, B. Kuhlenkötter, and J. Deuse, "Cyber physical systems for life cycle continuous technical documentation of manufacturing facilities," *Procedia CIRP*, vol. 17, pp. 207–211, 2014, Variety Management in Manufacturing, ISSN: 2212-8271. DOI: https://doi.org/10.1016/j.procir.2014.01.050. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2212827114002856.

[9] P. Fantini, G. Tavola, M. Taisch, J. Barbosa, P. Leitao, Y. Liu, M. S. Sayed, and N. Lohse, "Exploring the integration of the human as a flexibility factor in CPS enabled manufacturing environments: Methodology and results," in *Proc. of IEEE IECON Conference*, 2016, pp. 5711–5716.

[10] C. Krupitzer, S. Müller, V. Lesch, M. Züfle, J. Edinger, A. Lemken, D. Schäfer, S. Kounev, and C. Becker, "A survey on human machine interaction in industry 4.0," *ACM*, vol. 1, no. 1, 2020.

[11] P. Makris, D. Skoutas, and C. Skianis, "A survey on context-aware mobile and wireless networking: On networking and computing environments' integration," *IEEE Communications Surveys & Tutorials*, vol. 15, Jan. 2012. DOI: 10.1109/SURV.2012.040912.00180.

[12] A. Rahmati and L. Zhong, "Context-based network estimation for energy-efficient ubiquitous wireless connectivity," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 54–66, 2011.

[13] S. Fernandes and A. Karmouch, "Vertical mobility management architectures in wireless networks: A comprehensive survey and future directions," *Communications Surveys & Tutorials, IEEE*, vol. 14, pp. 45–63, Mar. 2012. DOI: 10.1109/SURV.2011.082010.00099.

[14] D. Romero, J. Stahre, T. Wuest, O. Noran, P. Bernus, A. Fasth, and D. Gorecky, "Towards an operator 4.0 typology: A human-centric perspective on the fourth industrial revolution technologies," in *Proc. Int'l Conf. on Computers and Industrial Engineering*, 2016.

[15] J. Hauswald, M. A. Laurenzano, Y. Zhang, *et al.*, "Sirius: An open end-to-end voice and vision personal assistant and its implications for future warehouse scale computers," in *20th Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, 2015, pp. 223–238.

[16] S. Hoedt, A. Claeys, H. V. Landeghem, and J. Cottyn, "The evaluation of an elementary virtual training system for manual assembly," *Int'l Journal of Production Research*, vol. 55, no. 24, pp. 7496–7508, 2017. DOI: 10.1080/00207543.2017.1374572. [Online]. Available: https://doi.org/10.1080/00207543.2017.1374572.

[17] M. A. Frigo, E. C. da Silva, and G. F. Barbosa, "Augmented reality in aerospace manufacturing: A review," *Journal of Industrial and Intelligent Information*, vol. 4, no. 2, 2016.

[18] R. Pierdicca, E. Frontoni, R. Pollini, M. Trani, and L. Verdini, "The use of augmented reality glasses for the application in industry 4.0," in *Int. Conf. on Augmented Reality, Virtual Reality and Computer Graphics*, 2017, pp. 389–401.

[19] G. Mantovani, "Vr learning: Potential and challenges for the use of 3d," *Towards cyberpsychology: Mind, cognitions, and society in the Internet age*, pp. 208–225, 2003.

[20] M. Abidi, A. Al-Ahmari, A. Ahmad, W. Ameen, and H. Alkhalefah, "Assessment of virtual reality-based manufacturing assembly training system," *The International Journal of Advanced Manufacturing Technology*, vol. 105, Dec. 2019. DOI: 10.1007/s00170-019-03801-3.

[21] Z. Zhu, V. Branzoi, M. Wolverton, G. Murray, N. Vitovitch, L. Yarnall, and R. Kumar, "Ar-mentor: Augmented reality based mentoring system," in *Proc. of the IEEE international symposium on mixed and augmented reality (ISMAR'14)*, 2014, pp. 17–22.

[22] S. Webel, U. Bockholt, T. Engelke, N. Gavish, M. Olbrich, and C. Preusche, "Augmented reality training for assembly and maintenance skills," *Robotics and Auton. Systems*, vol. 61, no. 4, pp. 398–403, 2013.

[23] R. Roy, R. Stark, K. Tracht, S. Takata, and M. Mori, "Continuous maintenance and the future – foundations and technological challenges," *CIRP Annals*, vol. 65, no. 2, pp. 667–688, 2016.

[24] J. Q. Coburn, J. L. Salmon, and I. Freeman, "Effectiveness of an immersive virtual environment for collaboration with gesture support using low-cost hardware," *J. Mech. Design*, vol. 140, no. 4, 2018.

[25] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta, "Artificial vision in road vehicles," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1258–1271, 2002.

[26] S. Cubero, N. Aleixos, E. Moltó, J. Gómez-Sanchis, and J. Blasco, "Advances in machine vision applications for automatic inspection and quality evaluation of fruits and vegetables," *Food and bioprocess technology*, vol. 4, no. 4, pp. 487–504, 2011.

[27] J. Servan, F. Mas, J. L. Menendez, and J. Rios, "Using augmented reality in airbus a400m shop floor assembly work instructions," in *Aip conference proceedings*, American Institute of Physics, vol. 1431, 2012, pp. 633–640.

[28] C. Yang, Y. Jang, J. Beh, D. Han, and H. Ko, "Gesture recognition using depth-based hand tracking for contactless controller application," in *2012 IEEE International Conference on Consumer Electronics (ICCE)*, IEEE, 2012, pp. 297–298.

[29] S. Nedevschi, S. Bota, and C. Tomiuc, "Stereo-based pedestrian detection for collision-avoidance applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 380–391, 2009.

[30] *Microsoft kinect 2.0*, `https://developer.microsoft.com/en-us/windows/kinect/`, Accessed: 2020-09-22.

[31] *Microsoft azure kinect*, `https://azure.microsoft.com/en-us/services/kinect-dk/`, Accessed: 2020-09-22.

[32] *Intel realsense*, `https://www.intelrealsense.com/`, Accessed: 2020-09-22.

[33] *Stereolabs zed*, `https://www.stereolabs.com/`, Accessed: 2020-09-22.

[34] Seeed, "Tof (time of flight) vs stereo vision – 3d imaging technology comparison," Tech. Rep., Jul. 2020.

[35] *Opencv*, `https://opencv.org/`, Accessed: 2020-09-23.

[36] *Google cloud vision api*, `https://cloud.google.com/vision`, Accessed: 2020-09-23.

[37] *Tensorflow*, `https://www.tensorflow.org/`, Accessed: 2020-09-23.

[38] A. R. Elias, N. Golubovic, C. Krintz, and R. Wolski, "Where's the bear?-automating wildlife image processing using iot and edge cloud systems," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, IEEE, 2017, pp. 247–258.

[39] K. A. Smith, C. Csech, D. Murdoch, and G. Shaker, "Gesture recognition using mm-wave sensor for human-car interface," *IEEE Sensors Letters*, vol. 2, no. 2, pp. 1–4, 2018.

[40] Q. Wan, Y. Li, C. Li, and R. Pal, "Gesture recognition for smart home applications using portable radar sensors," in *2014 36th annual international conference of the IEEE engineering in medicine and biology society*, IEEE, 2014, pp. 6414–6417.

[41] *Stackoverflow 2020 survey*, `https://insights.stackoverflow.com/survey/2020/`, Accessed: 2020-10-22.

[42] B. Simões, R. De Amicis, I. Barandiaran, and J. Posada, "X-reality system architecture for industry 4.0 processes," *Multimodal Technologies and Interaction*, vol. 2, no. 4, p. 72, 2018.

[43] J. Shen and N. Gans, "Robot-to-human feedback and automatic object grasping using an rgb-d camera–projector system," *Robotica*, vol. 36, no. 2, pp. 241–260, 2018.

[44] T. Wengefeld, D. Höchemer, B. Lewandowski, M. Köhler, M. Beer, and H.-M. Gross, "A laser projection system for robot intention communication and human robot interaction," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, IEEE, 2020, pp. 259–265.

[45] Y. Yi, X. Liu, and Z. Ni, "Digital twin-based human-machine collaboration and application approach for laser projection aided assembly of complex product," in *2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, IEEE, vol. 2, 2020, pp. 100–103.

[46] S. Nivetha, "A survey on speech feature extraction and classification techniques," in *2020 International Conference on Inventive Computation Technologies (ICICT)*, 2020, pp. 48–53. DOI: 10.1109/ICICT48043.2020.9112582.

[47] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *2013 IEEE international conference on acoustics, speech and signal processing*, IEEE, 2013, pp. 7398–7402.

[48] J. Yao, Y. Liu, and Y. Shi, "A speech control system for intelligent industry," in *2016 International Conference on Computer and Information Technology Applications*, Atlantis Press, 2016.

[49] *Cmusphinx speech recognition*, https://cmusphinx.github.io/, Accessed: 2020-10-22.

[50] *Kaldi asr*, https://kaldi-asr.org/, Accessed: 2020-10-22.

[51] *Google speech api*, https://cloud.google.com/speech-to-text/, Accessed: 2020-10-22.

[52] *Ibm watson*, https://www.ibm.com/watson/, Accessed: 2020-10-22.

[53] *Web speech api*, `https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API/`, Accessed: 2020-10-22.

[54] *Google glass*, `https://www.google.com/glass/start/`, Accessed: 2020-09-24.

[55] *Microsoft hololens*, `https://www.microsoft.com/en-us/hololens`, Accessed: 2020-09-24.

[56] *Facebook oculus*, `https://www.oculus.com/`, Accessed: 2020-09-24.

[57] N. Navab, "Developing killer apps for industrial augmented reality," *IEEE Computer Graphics and applications*, vol. 24, no. 3, pp. 16–20, 2004.

[58] P. Horejsi, "Augmented reality system for virtual training of parts assembly," *Procedia Engineering*, vol. 100, no. January, pp. 699–706, 2015.

[59] E. Van Wyk and R. De Villiers, "Virtual reality training applications for the mining industry," in *Proceedings of the 6th international conference on computer graphics, virtual reality, visualisation and interaction in Africa*, 2009, pp. 53–63.

[60] K. S. Gurusamy, R. Aggarwal, L. Palanivelu, and B. R. Davidson, "Virtual reality training for surgical trainees in laparoscopic surgery," *Cochrane database of systematic reviews*, no. 1, 2009.

[61] *ITI VR*, `https://www.iti.com/vr`, Accessed: 2020-09-24.

[62] *DiSTI*, `https://disti.com/`, Accessed: 2020-09-24.

[63] *ECA group*, `https://www.ecagroup.com/`, Accessed: 2020-09-24.

[64] S.-G. Ouyang, G. Wang, J.-Y. Yao, G.-H.-W. Zhu, Z.-Y. Liu, and C. Feng, "A unity3d-based interactive three-dimensional virtual practice platform for chemical engineering," *Computer Applications in Engineering Education*, vol. 26, no. 1, pp. 91–100, 2018.

[65] H. Zhang, "Head-mounted display-based intuitive virtual reality training system for the mining industry," *International Journal of Mining Science and Technology*, vol. 27, no. 4, pp. 717–722, 2017.

[66] V. Liagkou, D. Salmas, and C. Stylios, "Realizing virtual reality learning environment for industry 4.0," *Procedia CIRP*, vol. 79, pp. 712–717, 2019.

[67] *Unity3d*, https://unity.com/, Accessed: 2020-09-24.

[68] *Unreal engine*, https://www.unrealengine.com/en-US/, Accessed: 2020-09-24.

[69] *Amazon sumerian*, https://aws.amazon.com/pt/sumerian/, Accessed: 2020-09-24.

[70] *Understanding color models: A review*, https://www.researchgate.net/figure/HSV-color-model-single-hex-cone-1014_fig8_266462481/, Accessed: 2020-11-16.

[71] N. Mohd Ali, "Performance Comparison between RGB and HSV Color Segmentations for Road Signs Detection," *App. Mech. and Mater.*, 2013. DOI: 10.4028/www.scientific.net/AMM.393.550.

[72] *Choosing the correct upper and lower HSV boundaries for color detection with'cv::inRange' (openCV)*, https://www.xspdf.com/help/50641240.html/, Accessed: 2020-11-16.

[73] S. Suzuki *et al.*, "Topological structural analysis of digitized binary images by border following," *Computer vision, graphics, and image processing*, vol. 30, no. 1, pp. 32–46, 1985.

[74] *Blender*, https://www.blender.org/, Accessed: 2020-10-22.

[75] *Measuring and interpreting system usability scale*, https://uiuxtrend.com/measuring-system-usability-scale-sus/, Accessed: 2020-05-29.