FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Smart And Cost-Effective Manipulator's End-Effector For Tomato Harvesting

Francisco Adrião Silva Oliveira

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Filipe Baptista Neves Dos Santos

July 27, 2022

This work was carried out with the support of the host institution: INESC TEC and supervised at the institution by: INESC TEC

The INESC TEC supervising team was: Filipe Baptista Neves dos Santos, Sandro Augusto Costa Magalhães and Vítor Daniel Veloso Tinoco

Resumo

Tem havido um aumento na variedade de manipuladores de colheita desenvolvidos. No entanto, por vezes a falta de eficiência destes manipuladores, em comparação com as tarefas de colheita executadas por humanos, dificulta o seu uso como solução para a falta de mão de obra existente ou como auxílio para a realização de tarefas repetitivas. Um dos componentes fundamentais destes manipuladores é a ferramenta, responsável por retirar o fruto da planta. Esta dissertação estuda o projeto e a construção de uma ferramenta robótica capaz de colher tomate para ser incluída num manipulador com base num Selective Compliance Assembly Robot Arm (SCARA).

De modo a cumprir o objetivo de desenvolver a ferramenta robótica, o trabalho desta dissertação contribui com:

- Uma revisão literária sobre ferramentas robóticas disponíveis atualmente.
- Um projeto *open source* para uma ferramenta robótica chamada FruitGrip.
- Teste e validação da ferramenta na realização de tarefas de colheita num laboratório em ambiente artificial.

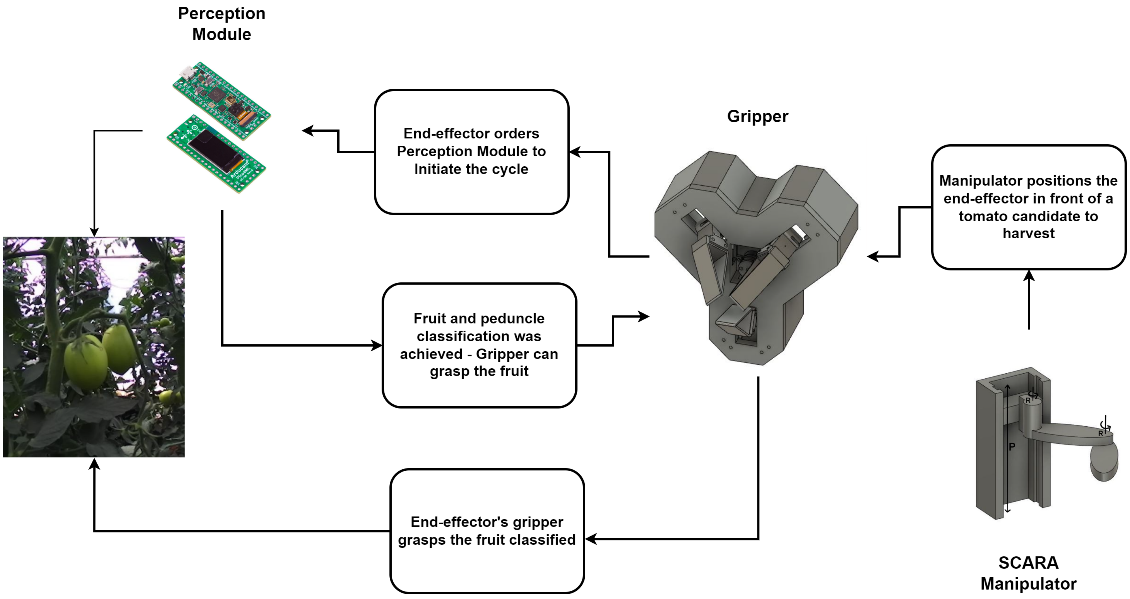
Abstract

There has been an increase in the variety of harvesting manipulators developed. However, sometimes the lack of efficiency of these manipulators, when compared with harvesting tasks performed by humans, makes it difficult to be used as a solution to the lack of labour available or as a tool to help with repetitive tasks. One of the key components of these manipulators is the end-effector, responsible for picking the fruits from the plant. This dissertation studies the design and making of an end-effector capable of harvesting tomato to be included on a custom Selective Compliance Assembly Robot Arm (SCARA) manipulator.

To achieve the objective of developing the end-effector, this dissertation work contributes with:

- A systematic review about end-effectors already available.
- An open source design for an end-effector called FruitGrip.
- Test and validation of the tool on doing harvesting tasks in a laboratory with an artifical environment.

Graphical Abstract



Agradecimentos

Ao meu orientador, Filipe Santos, pela oportunidade de fazer esta dissertação, assim como pelo apoio e conhecimento que me foi transmitido durante o semestre.

Ao Sandro Magalhães e ao Vítor Tinoco, pela disponibilidade para me ajudarem nas diversas fases do projeto e me guiarem para atingir os objetivos pretendidos.

A todos os elementos da equipa do CRIIS, por me acolherem durante este semestre no laboratório não só como aluno, mas como colega de trabalho. Foi um enorme gosto poder trabalhar diariamente com vocês.

A todos os meus amigos e família, por me apoiarem em todo o meu percurso académico, motivando-me nos momentos mais complicados.

Francisco Oliveira

"I have no idols. I admire work,
dedication and competence."

Ayrton Senna

Contents

List of Figures

List of Tables

Abbreviations and Symbols

ADC	Analogue to Digital Converter
CAD	Computer Aided Design
DoF	Degrees of Freedom
FSR	Force Sensing Resistor
GPIO	General-Purpose Input/Output
IMU	Inertial Measurement Unit
LCD	Liquid Crystal Display
LiDAR	Light Detection And Ranging
MSB	Most Significant Bit
NN	Neural Network
PETG	Polyethylene Terephthalate Glycol
PLA	Polylactic Acid
PWM	Pulse Width Modulation
RGB	Red Green Blue
SCARA	Selective Compliance Assembly Robot Arm
SPI	Serial Peripheral Interface
TPU	Thermoplastic Polyurethane
TRL	Technology Readiness Level
UART	Universal Asynchronous Receiver/Transmitter
UI	User Interface
USB	Universal Serial Bus

Chapter 1

Introduction

1.1 Context

The use of robots in agriculture aims to simplify arduous agricultural tasks which were solely performed with human labour. The increasing world population has risen the necessity of an increment in the food production, namely in agricultural goods [7]. However, there is less human labour available to complete the necessary tasks, mainly due to the physical difficulties associated with this type of work. On the other hand, the cost of hiring labour has seen an increase, since the urbanization of the rural areas leads to more job opportunities other than farming; furthermore, the youth on those areas can achieve better qualifications compared to the same section of the population in the past [8].

The previously mentioned tasks, such as harvesting, require high precision to guarantee that the fruit picked is not damaged in the process, nor the surrounding fruits or the plant itself. The success of a task is not just dependent on the type of manipulator. The robot requires a complex set of interconnected modules [9] to guarantee the well function of the system, including an end-effector. The end-effector is an external tool which is coupled to a manipulator and assists in fulfilling the tasks of the robot, such as cutting, grasping, drilling, among others [10]. In harvesting manipulators, the end-effector requires - besides other functionalities in some cases - the capability of grasping the fruits and take them out of the plant without damaging them while having the necessary strength to not let them fall.

1.2 Motivation

As previously mentioned, agricultural robotics is an area of increasing interest which is becoming more relevant nowadays to assist the human population in arduous tasks.

Considering the importance of this area on the modern agriculture, which aims to be part of the solution to increment the food production, this project is an opportunity to study how to increase the efficiency of the harvesting robots by developing a crucial tool needed to perform harvesting tasks and analysing its performance.

1.3 Objectives

The main objective of this dissertation is to design and build a smart and cost-effective end-effector capable of harvesting tomatoes.

To fulfil this objective there is the need to complete the following objectives:

- Develop an electrical end-effector capable of grasping tomato. – To complete this step, it is required to design the tool structure and assembly of the different parts, developing also the software needed for the end-effector to complete the tasks needed to harvest successfully.
- Implement a perception module capable to distinguish between tomato and peduncle. – To fulfil this objective, the perception module must be developed using modern technologies, such as machine learning, in order to optimize the tool to achieve better results.
- The end-effector must be safe to use on the harvesting task, not damaging the tomato.

1.4 Requirements

For this work is expected the end-effector fulfils the following requirements:

- The end-effector needs to be attached to a robot based on a custom Selective Compliance Assembly Robot Arm (SCARA) manipulator already being developed.
- The end-effector is to be designed considering the different steps of the harvesting task, which implies being capable of approaching the fruit without causing damage to any parts of the plant and grasp it.
- The end-effector is expected to have a perception module to be able of detecting the fruit it was designed to grasp, as well as the peduncle of that fruit, to adjust its position for optimal harvesting.
- Establish a communication protocol between the end-effector and the manipulator's Raspberry Pi.
- The end-effector is expected to be cost-effective, fulfilling the previous requirements without exceeding on the cost of production.

1.5 Dissertation Structure

This dissertation is organized in eight chapters.

- In chapter 2 are analysed different solutions of end-effectors already available.
- In chapter 3 is explained the architecture and design of the end-effector built.

- Chapter 4 lists the hardware needed during the project, as well as analyses the different characteristics of those components.
- In chapter 5 is analysed the software developed during the dissertation.
- In chapter 6 are presented the individual tests and validation of the different components of the end-effector.
- In chapter 7 are analysed the results obtained while testing the complete tool assembled, overviewing the performance of the different parts.
- Chapter 8 contains the conclusions of the work developed during the dissertation, as well as the aspects that need to be addressed in the future.

Chapter 2

Harvesting Manipulators - A Systematic Review

In this chapter is done a literature review on different end-effectors developed in the last few years. There is also done a background overview on relevant characteristics of manipulators, which may influence the efficiency of a robot in the task of harvesting.

2.1 Methodology

The information gathered for this document is the result of research in platforms such as Scopus, ResearchGate, Google Scholar and IEEE Xplore about end-effectors for harvesting manipulators. The following keywords were used: End-effectors, Harvesting End-effectors, Harvesting Manipulator.

As result of a simple research done in the Scopus database for harvesting manipulators (search: harvesting AND manipulator) and harvesting end-effectors (search: harvesting AND end-effector) it is possible to analyse that from 2017 to 2021 the number of documents published as seen a general increase, as shown in Figure 2.1 and 2.2, meaning that this has been a topic with an increasing interest over the last years.

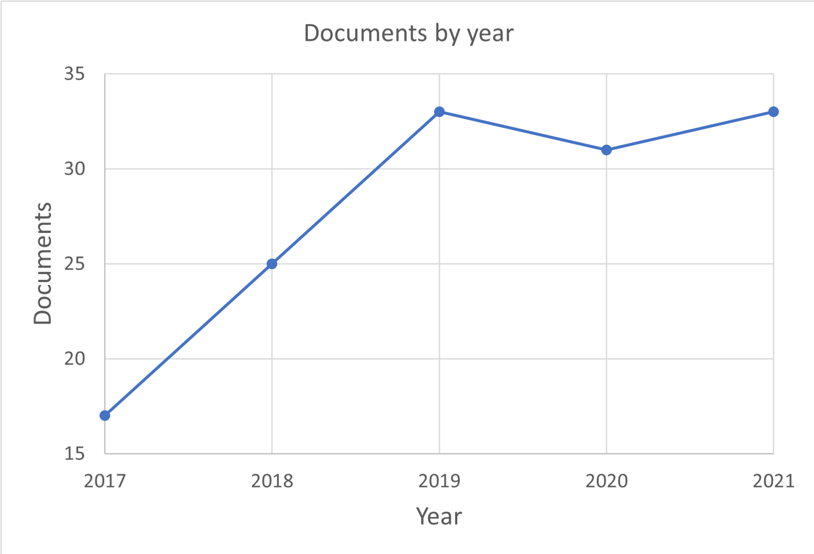


Figure 2.1: Number of Harvesting Manipulators Documents Published Between 2017-2021 - Scopus

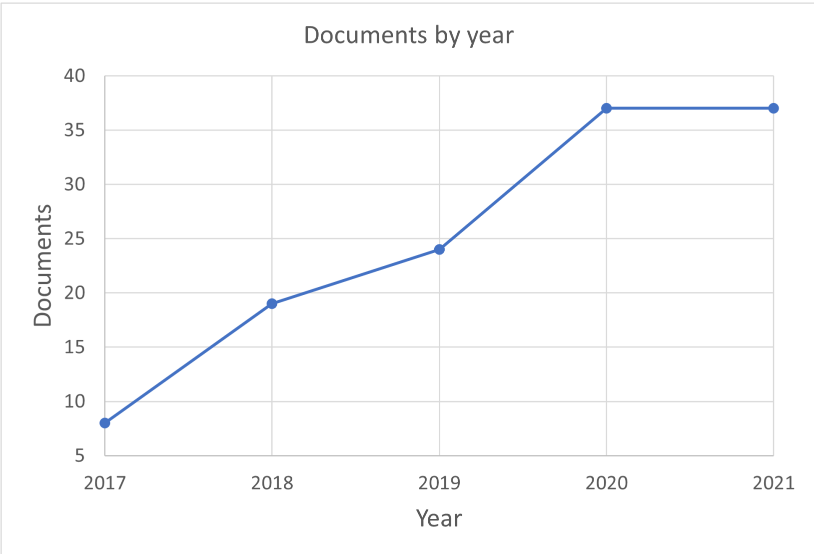


Figure 2.2: Number of Harvesting End-Effectors Documents Published Between 2017-2021 - Scopus

2.2 Background

While the main goal of the dissertation is to develop an end-effector for a manipulator, it is fundamental to analyse the different manipulators and their characteristics as a whole, without focusing exclusively on the end-effector. This was addressed in the research done by Tinoco *et al.* [11] [12] and consists on achieving an understanding on the role of the different parts of the manipulators in a given context, relating them to the success of the tasks. Tinoco *et al.* [11] [12] concluded that there is a link between the type of end-effector used and the number of Degrees of Freedom (DoF) required by the manipulator to perform the tasks with success. The Degrees of Freedom are consequently related to the number of joints in the manipulator, which can be prismatic or revolute. The prismatic joint movement is linear along an axis, while the revolute joint confers rotational movement [13].

The manipulator currently being built, as mentioned before, consists of a custom SCARA manipulator, articulated by 1 prismatic joint and 2 revolute joints (PRR), resulting in 3 Degrees of Freedom (DoF), similar to the sketch in Figure 2.3. The end-effector can be designed with a configuration that will give more DoF to the manipulator by adding more joints to the tool system, allowing for a more complex range of movements, which will possibly increase the robot efficiency.

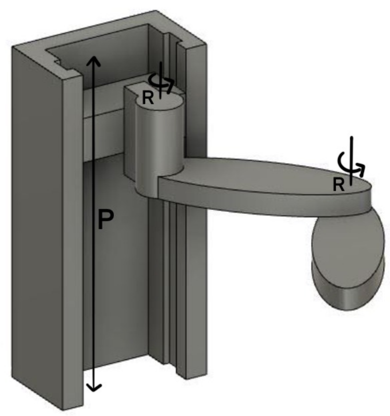


Figure 2.3: Sketch of a Custom SCARA Manipulator Configuration With PRR Articulation

Tinoco *et al.* [11] [12] also realized that most of the harvesting manipulators were developed for a type of harvesting. They were designed to be capable of harvesting fruits with specific characteristics and could eventually not be successful trying to harvest fruits with different shapes or sizes. Other aspect noticed was that, even in the intended tasks, sometimes the manipulators would still have difficulties to perform; one of the reasons was related to the inefficiency of gripping the fruits.

Combining the above is possible to understand that the design of the end-effector will have influence on the efficiency of the manipulator for the desired tasks. However, it is crucial to understand that a good solution for a type of manipulator and harvest can be a bad one on a different scenario. This means that while analysing the different solutions developed it is important to compare them with the environment the end-effector is intended to be used. The same technology applied to different fruits or types of manipulators can result in a different efficiency when compared to the studied cases.

2.3 State Of The Art Analysis

In the state of the art analysis it was chosen a total of six articles containing end-effectors developed to complete harvesting tasks, ranging from the year of 2009 to 2021. These end-effectors are analysed in the following subsections in chronological order, where it was made an overview of their characteristics and efficiency on completing the tasks they were designed to achieve.

Integrated Gripper And Cutter For Harvesting Greenhouse Products

In 2009, Jia *et al.* [1] developed a tool capable of gripping and cutting to be integrated on harvesting manipulators. This tool was intended to be lightweight, simple and low-cost, while being versatile so it could be used on different manipulators to harvest different fruits, such as tomatoes or grapes. For this purpose, the authors designed an end-effector consisting mainly of pliers (for the gripper) and a scissor (for the cutter) (Figure 2.4). This simple system was conceived with the objective of avoiding force sensors, since it would grab the peduncle instead of the fruit, making it possible for the manipulator to place the fruit/cluster on a desired place since it wouldn't fall after the cutting.

Jia *et al.* [1] conducted tests in two different scenarios. The first test was to pick tomatoes by the peduncle from shelves. The authors observed that the system took 37 s to pick each tomato and it was successful on all tries. The second test had the objective to measure the necessary strength to cut tomatoes' peduncles. The authors verified values ranging from 31 N to 71 N with the average of 39.22 ±12.36N.

It was concluded this tool design is capable of performing harvest tasks of several different fruits, on the condition that the peduncles are long enough for the size of the designed end-effector.

Figure 2.4: Integrated Gripper And Cutter For Harvesting Greenhouse Products - Adapted From [1]

Tomato Cluster Harvesting Robot

In 2010, Kondo *et al.* [2] developed an end-effector for a tomato cluster harvesting robot. The authors noted that focusing on harvesting clusters, instead of a single fruit at a time, would be more efficient as the robot could pick multiple fruits at once. The disadvantage of using this method, as previously stated, is the increased probability of harvesting unripe fruit.

Harvesting clusters implies cutting the plant's peduncle, which can vary in diameter and consequently compromise the efficiency of the process. This fact was also a part of the study in the research from Kondo *et al.* [2], where it was stated that the maximum cutting resistance on the peduncles was approximately 60 N for the largest ones. This makes it a relevant aspect while designing the end-effector, since the cutting tool needs to apply this force to be able to cut the largest peduncles.

The authors developed the robot based on a SCARA manipulator (Mitsubishi Electric Corporation [14] RH-6SH5520) resulting on a manipulator with four degrees of freedom. The authors concluded this was a capable choice, since the plants are mostly the same height. As a consequence, the prominent movements are horizontal and easily reachable with the SCARA robotic arm. Analysing the manipulator used by Kondo *et al.* [2], it is possible to reflect on some aspects of the development of the end-effector. The authors developed a complex mechanism consisting mainly of two upper fingers, where the cutter is placed, and two lower fingers, all assembled on a structure capable of rotating on itself with the actuation of a servomotor (Figure 2.5). These fingers wouldn't grasp the peduncle of the cluster but surround the main stem of the plant. After that, the end-effector would move along the main stem until the "peduncle detection sensor" on the lower fingers detected a peduncle and then the cutter would cut it.

Kondo *et al.* [2] stated that the end-effector worked correctly as it was capable of harvesting the tomato clusters with an execution time of about 15 s for each cluster. The test, performed on 20 tomato clusters, resulted on a success rate of 50 %. The authors concluded that in seven of the tomato clusters not harvested the explanation for the failed procedure was due to the end-effector not being able to surround the main stem in high-density plants, ending up hitting the clusters near the actuation zone of the end-effector. One possible solution mentioned in the article is the reduction of the actuator height, making it more compact to reach the main stem in high-density plants, increasing then its efficiency.

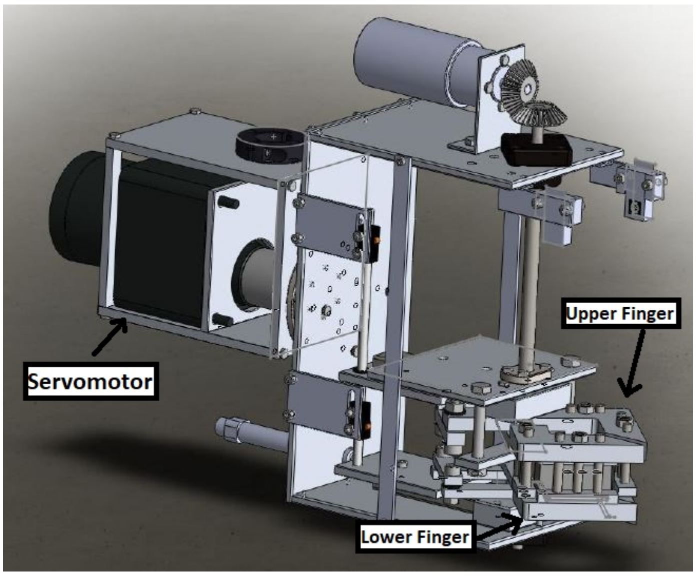


Figure 2.5: Tomato Cluster Harvesting End-Effector - Adapted From [2]

Autonomous Tomato Harvesting Robot With Rotational Plucking Gripper

In 2016, Yaguchi *et al.* [3] developed an autonomous tomato harvesting robot. In a previous work, the authors developed a humanoid robot with a scissor hand that achieved 60 % success rate but needed teleoperation support of a human [15]. The authors realized that one of the tasks that could compromise the harvest was related with cutting the stem of the fruit, which requires recognition of the stem. This may be more difficult than just fruit recognition because the fruit has size and colour features, making it simpler to distinguish from the surroundings [3]. Having that in consideration, the authors opted for a rotational plucking mechanism instead of blades to cut the stem.

The perception module on the work developed by Yaguchi *et al.* [3] was achieved using a stereo camera. The algorithm uses colour extraction to find the possible candidates, find the nearest cluster, and uses the sphere fitting method to identify the tomatoes. The use of a stereo camera allowed to measure the distance from the end-effector to the tomatoes.

The end-effector designed by Yaguchi *et al.* [3] consisted of a hand with two degrees of freedom, one being the fingers (three rigid fingers) whose movement allowed them to open and close and the other being the infinite rotation of the wrist, as shown in Figure 2.6. The process of harvesting consists in grasping a single tomato by closing the fingers around it and then actuate the rotation of the whole hand. This allows for the pedicel to brake and separate from the rest of the stem, picking the fruit from the plant. The authors concluded this method could harvest a single tomato in 23 s and, after experimental testing, the success rate was 62.2 %. However, the authors believed that this success rate is inaccurate because more harvesting trials need to be done.

Figure 2.6: Rotational Plucking Gripper - Adapted From [3]

Yaguchi *et al.* [3] detected some common cases of failure during the harvesting tests. In some attempts the position of the fruit was miscalculated or the manipulator would move part of the plant, causing the tomato not to be grasped by the hand. Another type of failed attempts consisted in the hand, while trying to reach the tomato in a cluster, grasping multiple fruits together. This situation occurred in cases of dense tomato clusters.

Sweet Pepper Harvesting

In 2017, Lehnert *et al.* [4] developed a robotic harvester with the purpose of harvesting sweet peppers, which was equipped with a novel end-effector also developed by the authors.

The end-effector consisted mainly of two fundamental parts, a suction cup to grip the pepper and an oscillating blade to separate the pepper from the plant. While designing the robot, the authors analysed different types of grippers and besides considering contact-based grippers, for example with mechanical fingers, effective on gripping the fruits, didn't opt to use them since this type of gripper can suffer interference from the surroundings, such as branches. Lehnert *et al.* [4] decided to use a suction cup for its simplicity and also because it only requires to reach a face of the pepper. The suction cup is coupled to the end-effector with a flexible strip and magnetically attached to the cutting mechanism of the end effector, an oscillating blade. The perception algorithm combines different frames obtained from a Red Green Blue-Depth (RGB-D) camera to obtain a 3D model. Subsequently, the system segments the peppers to distinguish them from the background. This configuration allows the end-effector to detect and reach the pepper, and grip it with the suction cup. After it is successfully secured, the cup separates from the blade, allowing the arm to move and get the blade closer to the pepper stem and cut it, as shown in Figure 2.7 and Figure 2.8. After dropping the pepper, the robot only has to point the end-effector down and using gravity to its advantage it is possible to re-connect again the suction cup to the cutting mechanism.

Figure 2.7: Sweet Pepper Harvester - Grip The Pepper - Adapted From The Video by Lehnert *et al.* [4]

Figure 2.8: Sweet Pepper Harvester - Cut The Stem - Adapted From The Video by Lehnert *et al.* [4]

Lehnert *et al.* [4] conducted two different field trials, on a total of 75 peppers. From these, 24 peppers were harvested in the first trial and 26 in the second, being the remaining 25 an additional test, however not under strict conditions. For the second trial, the team performed some modifications on the manipulator to improve the system behaviour. These modifications included changes to the end-effector, which was provided with a vacuum sensor to detect the attachment to the suction cup and a micro-switch to detect the decoupling of the suction cup and the cutting blade.

Overall, in the whole process (Attachment and Detachment) the robot achieved a success rate of 58 % and 48 % in the first and second trial, respectively. However, this means the robot could harvest the fruit, but does not guarantee it was not damaged in the process. Analysing the individual harvesting results, it is possible to understand that some successful harvested peppers were in fact damaged. This damage was mostly caused by the blade actuation. Large irregularities on the pepper would cause a poor estimation of the pose and the blade cutting the peduncle would also cut part of the pepper.

The authors then concluded that in future work it would be better if the robot could detect also the peduncle and not only the pepper, since at this stage the cutting point was determined assuming the peduncles were always vertical to the centre of the pepper, which wasn't always the case.

Integrated End-Effector For Picking Kiwifruit

In 2020, Mu *et al.* [5] proposed an automated method to pick kiwifruits, separating the fruit from the stem. The authors recognized there were already several studies on the harvest of this fruit, however the methods previously developed wouldn't completely combine the task of grab, separate from the plant and safely unload the fruit. For this purpose the authors conducted a study to design and test an end-effector capable of complete these three tasks without damaging the kiwis, whose peel can be easily damaged during the process.

The end-effector developed consisted mainly of a structure with two rigid fingers designed according to the dimensions of the fruits, a fibre sensor, a position sensor, a pressure sensor and a stepper motor, as shown in Figure 2.9. This configuration allows the end-effector to reach for an individual kiwifruit and embrace it from below with the two fingers, assuring it successfully enveloped it with the readings from the fibre sensor. Then the motor initiates the movement of the fingers' structure, pulling the fruit from the plant. The fingers then drop the fruit on a designated place and the motor resets to the original position to be ready to actuate again.

Mu *et al.* [5] conducted a field test where 240 fruits were harvested. The authors concluded the harvesting success rate was affected by the surroundings, such as presence of branches, length of the stem, as well as the fruit maturity. Failed tries might be caused by the separation of nearby fruits, failure to grab the fruit and the fingers slipping. The average success rate of this field test was 94.2 % and only 4.9 % of the fruits picked had damaged peels, which the authors concluded is lower than with humans harvesting because the manipulator drops fewer fruits. The average picking time calculated was around 4 to 5 seconds for each kiwi.

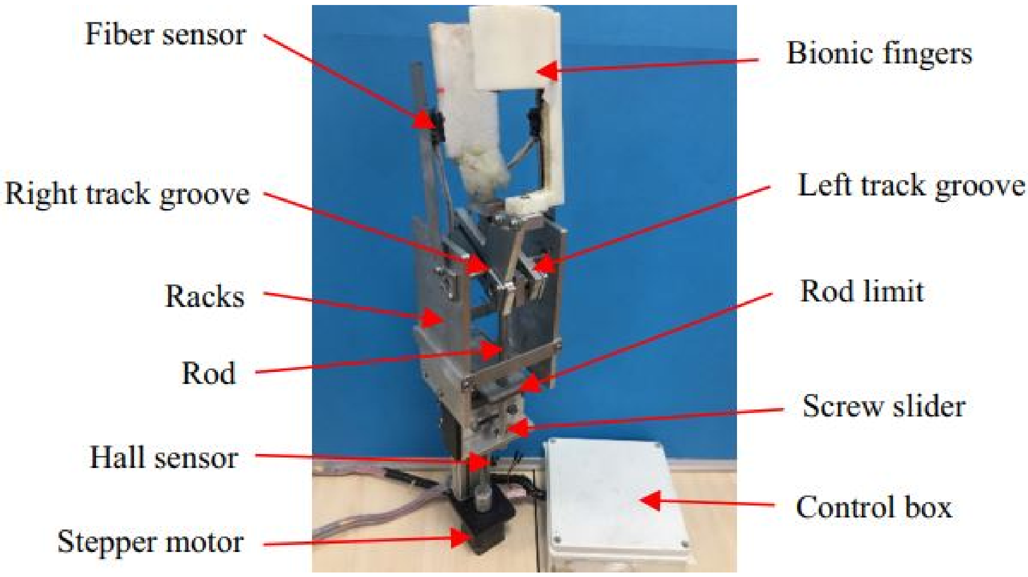


Figure 2.9: End-Effector For Picking Kiwifruit - Adapted From [5]

A Field-Tested Harvesting Robot For Oyster Mushroom in Greenhouse

In 2021, Rong *et al.* [6] proposed an oyster mushroom harvesting robot to be used in a greenhouse. The authors considered that two of the main components of the robot would be the perception module and the end-effector. The perception module consisted of an RGB-D camera, capable of providing colour images and depth, and a light source to be able to operate in low light conditions. The authors opted to use soft grippers for the end-effector and mounted these grippers on a structure capable of rotating, as shown in Figure 2.10. The objective was to simulate manual picking, due to the fact the mushrooms could be easily damaged while harvesting. The soft-gripper, with four fingers, is capable of wrapping around the mushroom surface, which would not be possible with rigid fingers. Subsequently, the structure rotates 180° to take the mushroom out of the cultivation bag.

The field experiments were conducted in daylight, with clear weather and good light conditions. Rong *et al.*[6] performed several system tests to evaluate the performance of the robot. After three experiments, the robot achieved a harvesting success rate of 86.8 %, taking an average of 8.85 s to pick each mushroom. The authors concluded the failed tries were mainly related to the perception module, such as plastic from the cultivation bags interfering with the detections or the existence of areas with poor illumination where the mushroom could not be identified. However, in some cases, the problem was also due to the gripper, where the mushrooms would slip from the fingers or some would cause the gripper to damage them while grasping. These situations were due to the inconsistent size and shape of the mushrooms, and also they tend to adhere together, causing the perception module to detect several mushrooms as one.

Figure 2.10: Mushroom Harvester End-Effector - Adapted From [6]

2.4 Discussion

Reflecting on the previous end-effectors, whose main characteristics are presented in Table 2.1, and comparing them, it is possible to better understand the functionalities of manipulators' end-effectors and what aspects are needed when building one.

Table 2.1: Harvesting End-Effectors

Source	Gripper	Application Field
Jia et al. [1]	Pliers & Cutter	Tomatoes
Yaguchi et al. [3]	3 Fingers	Tomatoes
Kondo et al. [2]	4 Fingers & Cutter	Tomatoes
Lehnert et al. [4]	Suction Cup & Cutter	Sweet Pepper
Mu et al. [5]	2 Fingers	Kiwifruit
Rong et al. [6]	4 Soft Fingers	Mushroom

Firstly, Kondo *et al.* [2], developed a complex and functional end-effector, with relevant features that are a good "first step" to develop a tool for a similar outcome. The idea of adding a servomotor to allow the end-effector to rotate appears to be an efficient method to add another DoF to the manipulator in a way that becomes possible to approach stems/peduncles that are not vertically aligned with the manipulator. Yet there were also some particularities that compromised the harvesting efficiency. One of these, mentioned before, was related to the size of the tool in plants with high-denisty. While it is crucial to have parts that guarantee the end-effector is on the correct stem/peduncle, it is important to have all of them on a compact tool that can approach the plant safely without damaging fruits or branches nearby. The chosen method to harvest, holding the stem and finding the peduncle by moving the tool along that stem, guarantees the peduncles belonging to that stem are found and cut. However, in non-ideal cases, where the main stem is below a high quantity of leaves or other branches, even a more compact version of this end-effector could have problems to reach it.

The solution presented by Jia *et al.* [1] does not appear as robust as the previous one. It doesn't add any extra movement to the manipulator, since it is just the tool without additional joints, and it does not have a perception module to detect the fruits. Although, considering the dimensions, which would be advantageous on high-density plants by having lower risk of compromising the harvesting while moving between clusters and branches, the design becomes an interesting alternative. Combined with an effective perception module and using additional joints to minimize the movement limitations, it could possibly be adapted to be used as a complete end-effector capable of completing harvesting tasks successfully. The technique used by Jia *et al.* [1] is also different from the one from Kondo *et al.* [2]: the end-effector, instead of embracing the main stem and "search" for the peduncles, is designed for a direct approach. Thereby, the tool actuates directly on the peduncle (the part that is intended to be grasp and cut). This can be useful in circumstances the main stem is unreachable, when the high-density of clusters or branches covers all the possible path.

The work done by Lehnert *et al.* [4] follows a completely different strategy. Instead of grasping the peduncle in a try for harvesting clusters, this work, developed for pepper harvesting, is focused on picking a single pepper at a time, grasping this one instead of its stem. The use of a detachable suction cup to grasp directly the pepper and a blade to cut the peduncle seems a more direct approach to a successful harvest, since it tries to avoid the problem of cruising between leaves and branches to reach for the peduncle. However, it can lead to high failure rates. This might occur because, to work properly, the system expects that some irregularities, both on the shape of the pepper and the orientation of the peduncle, do not exist. That is to say, the system tends to work but it is limited to peppers with a surface the suction cup can grip and a vertically aligned stem that the cutting blade can reach. These assumptions, not being always the case on the field, led to the failed tries and damaged peppers mentioned by the author.

Mu *et al.* [5] managed to find an efficient solution for the kiwifruit harvest. The method developed and the end-effector designed showed to be one with a good success rate in harvesting. However, while focusing only on develop the end-effector to harvest kiwifruits in the described environment increases the efficiency of the tool for that task, it is simple to understand this efficiency will be difficult to acquire in different scenarios. That is to say, this end-effector is more specific than the other solutions analysed since the tool contains two fingers that embrace the kiwifruit from below, this configuration would be difficult to implement with different fruits. Even with kiwifruits with a more irregular shape or size the end-effector could face operational problems, as even the authors stated that fruits that were too small would slip from the fingers. The fact that the approach is only done from below can be affected by branches and leaves, since the end-effector can't find another path to the same fruit and it also limits to plants like some vines, where the fruit can be accessible from below.

The rotational plucking mechanism developed by Yaguchi *et al.* [3] does not use a blade similar to the one by Mu *et al.* [5]. The main difference is while the previous one pulled the fruit downwards, the end-effector designed by Yaguchi *et al.* [3] rotates the hand while grasping the fruit. This method, as demonstrated by the authors, can safely pick the tomatoes most of the time without the danger of cutting the fruit, as seen in examples with a cutting mechanism associated with the end-effector. However, apart from the difficulties already mentioned by the authors, there is the fact that the hand is designed for picking a single tomato at a time. The end-effector can harvest this fruit because the rotational movement can break the pedicel, but the same technique might not work on fruits with a stronger or thicker pedicel/stem, where it could compromise the harvesting by damaging the fruit or the rest of the plant.

The end-effector developed by Rong *et al.* [6], similar to the previously two analysed, did not contain a cutting tool, opting for a rotational plucking mechanism as Yaguchi *et al.* [3]. However, the authors used soft grippers instead of rigid fingers. This technology allows to grip more sensitive materials that can be easily damaged, since the gripper tends to match the shape of the picked object without applying excessive force. However, the authors noticed, due to the fragility of the mushrooms, the uncontrolled force applied by the fingers sometimes could still cause damage to the mushrooms. Rong *et al.* [6] concluded that in future work the end-effector should be equipped with force feedback sensors to control the force applied and provide information if the mushroom was successfully picked.

2.5 Conclusion

These end-effectors can successfully complete the tasks they were designed to fulfil; yet, they are only guaranteed to have success in ideal conditions and assumption based situations. The authors of those end-effectors concluded the difficulties faced on the completion of the tasks can be minimized by improving the already existing designs to adapt to irregularities of the work environment. These irregularities include different shapes of fruits, density of branches and leaves and obstruction in the path to the fruit.

The problems associated with the obstruction in the path to the fruit might be difficult to minimize without altering the environment. However, by implementing an end-effector with increased movement freedom, as seen in the coupling of the tool by Kondo *et al.* [2], it would be possible to consider different alternatives to the path initially stipulated. Also, it is crucial to understand that for each type of harvest there are different solutions with different efficiencies. For example, the single fruit harvesting solutions analysed might achieve good results on harvesting fruits such as tomatoes. However, they are non-practical for harvesting smaller fruits, such as grapes, where it is more efficient to harvest the entire cluster together. A general end-effector capable of working in both environments, or even being easily modified to suit different situations could be beneficial, since it would not require producing a different tool for each type of harvest.

These are the type of considerations to have in mind while designing a tool for harvesting, understanding also that these already existing end-effectors can be improved to serve as a starting point to develop a more general tool that, apart from being designed with the purpose of harvesting a specific type of fruit, can be adapted to complete the task on different types successfully.

Chapter 3

FruitGrip - End-Effector Architecture And Design

This chapter analyses the structure chosen for the different parts of the end-effector developed, the FruitGrip. The design of the different components, mainly made in the software Fusion360[1], was developed having in consideration the variations on the shape and size of the objects that were intended to be grasped by the tool. The decisions made during the process are presented in this chapter, along with the explanation for the different components and configurations used.

3.1 Gripper Design

The gripper of the end-effector was the first part designed and assembled. The objective was to develop a gripper capable to grasp tomatoes. However, to increase the versatility of the tool, it was needed to design a gripper that can grab tomatoes of different sizes and shapes.

Having the above in consideration, the first step to develop the gripper consisted on choosing the type of mechanism to grasp the fruits. To avoid pneumatic actuators, keeping the system low-cost and less complex, mechanical fingers were considered to be the more efficient choice available. Rigid fingers could be inefficient since it would be more rigorous on the sizes and shapes of fruits capable of grasping, since the fingers can not deform during the actuation. The most fitting option was the use of soft grippers, flexible fingers that could adapt to the surface of the fruit while closing, which was already shown by the work of Rong *et al.* [6] that is the most secure solution to avoid damaging the fruits.

In the beginning of the design phase, several shapes of fingers were designed in the CAD software (Figure 3.1) and some of them were 3D printed in Thermoplastic Polyurethane (TPU) to test the flexibility of the fingers and the range of object sizes that was possible to grasp. However, the fingers printed were too flexible and lacked force to be able to grasp. Analysing the different solutions already available of soft grippers, the fingers based on the Fin Ray® Effect (a trademark

[1]Fusion 360: Software cad/cam 3d, Jan 2022. URL: http://www.autodesk.pt/products/fusion-360/overview

of Evologics GmbH[2]) seemed to be versatile for the intended task. The Fin Ray® Effect is a configuration based on the deformation of fish fins. The soft finger with origin on this effect consists of a V-shaped finger with interconnections inside, and when an external force is applied to the perimeter of the finger it deforms, matching the shape of the object applying the force [16]. These fingers can easily adapt to the surface of the object while having the necessary strength to maintain that shape without bending too much.

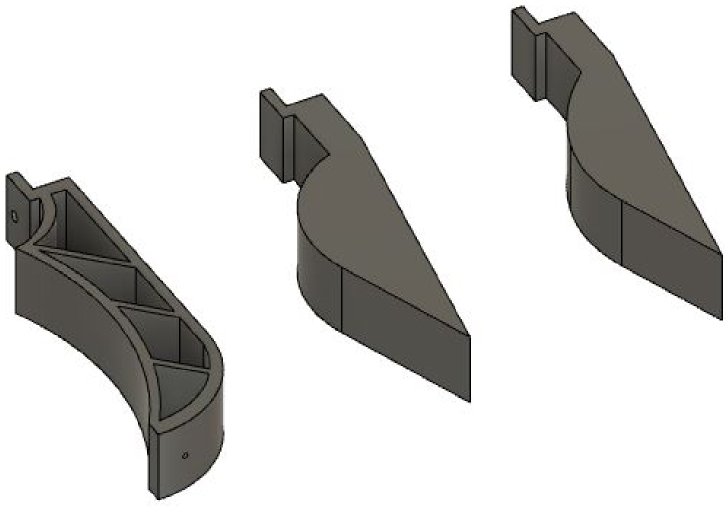
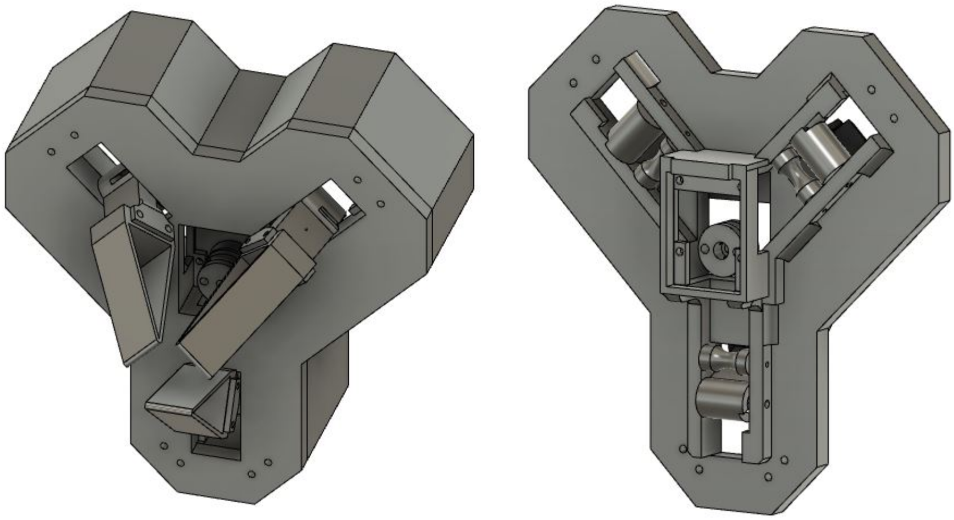


Figure 3.1: Gripper's Fingers Prototypes Designed on Fusion360

Having chosen the fingers, the second part of the problem related to grasping was their actuation. The need of a mechanism compact enough to not exceed the end-effector proportions (compromising its viability) was crucial, while guaranteeing the fruit was not damaged or fell during the process. The solution implemented consisted on using three Fin Ray® Effect based fingers. One of the fingers would be positioned on the bottom, to serve as a holder for the tomato, and the other two would be positioned on the top, making an angle of $45°$ and $-45°$ to the vertical axis (Figure 3.2). This configuration allowed the stabilization of the fruit, as well as it is adaptable to different sizes and shapes of tomatoes. At first, the idea was to actuate each finger independently with different motors, but this configuration would increase significantly the size and weight of the end-effector and since the fingers can deform to the shape of the fruit it is not strictly necessary to close each finger on their own. Having this in consideration, the chosen method was to control the three fingers with only one servomotor. The servomotor would be connected to the three fingers by strings and, when activated, the strings would be pulled, consequently closing all fingers simultaneously. The three strings were guided to the servomotor axis and the servomotor movement would roll them around that axis. However, the strings could not be tied directly on the servomotor axis, since it would put the servomotor in danger by tangling around it. The solution

[2]Evologics gmbh. URL: https://evologics.de/

was to develop an auxiliary part that was screwed to the axis and received the strings, having a shape capable of keeping the strings around it without tangling (Figure 3.3).



(a) Configuration (b) Interior Mechanism

Figure 3.2: Gripper External Configuration (Left) And Interior Mechanism (Right)

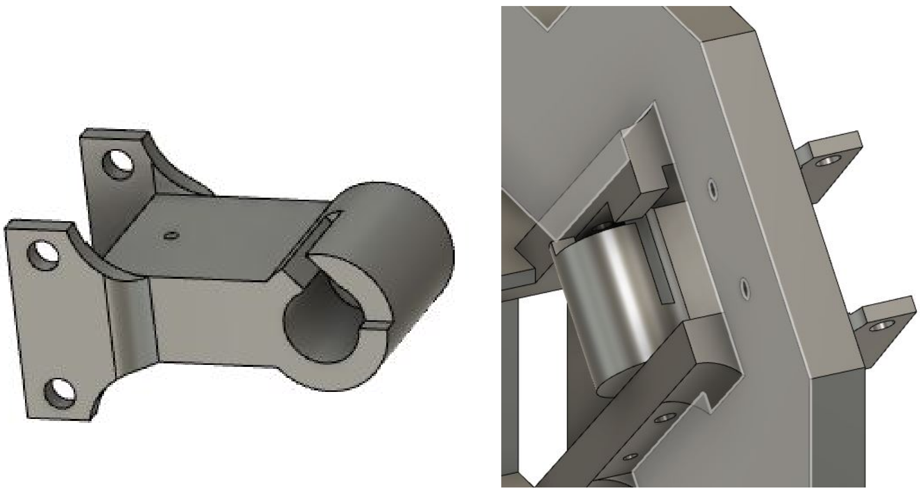
Figure 3.3: Servomotor Auxiliary Part

The gripper was designed to allow the end-effector to reach fruits with a maximum width of around 90 mm and height of around 100 mm. Closing to its maximum, it was supposed to be capable of grasping fruits as small as 20 mm width and height, approximately.

The servomotor was capable of applying a force to close the three fingers. However, a mechanism to apply an opposite force on the fingers was required, so they could maintain their original position when the servomotor stopped pulling the strings. The finger's mechanism was achieved on a spring based structure. The soft-gripper was attached to a structure that was connected to the front plate of the gripper (Figure 3.4) allowing it to rotate on the point of connection to guarantee the movement of the finger. On the rotating axis of this structure a torsion spring was included, whose legs would have the following configuration:

- One on the interior of the structure, which would follow its movement when the finger was closing.
- The other one positioned outside the structure to press against the front plate, creating the opposite force of the fingers' closing movement by generating tension on the spring.

The method applied was effective on maintaining the fingers open when the strings were not being pulled, as well, it was a low-cost solution to the intended objective.



(a) Finger Structure (b) Finger Structure in Front Plate

Figure 3.4: Finger Structure (Left) And The Component Assembled on The Front Plate (Right)

When approaching the fruit, it was crucial that the end-effector did not damage it. To minimize the risk, sensors could be used to understand with more accuracy the fruit's position, instead of only relying on a camera module. Force or weight sensors allow the system to be aware of the force exert by the fingers while grasping the tomato. The sensors capable of achieving this are, for example, load cells and force sensing resistors. However, since the Fin Ray® Effect based fingers would deform while grabbing the fruits, bending and consequently creating an angle, it would be possible to also use flexibility sensors, since these vary their resistance in order of the angle of the sensor strip.

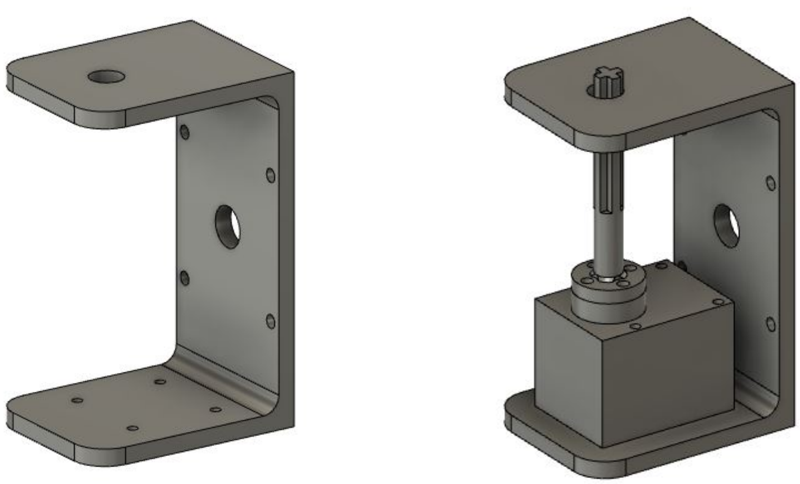
Load cells can achieve higher precision than the force sensing resistors, but their size and shape would be difficult to mount in order to detect the force applied by the soft gripper on the fruit. Nevertheless, the possibility of using load cells was not discarded, and it could still be useful to control the gripper's actuation. Having that in mind, a load cell was placed parallel to the front plate of the gripper on a support designed to cover the "servomotor window". This load cell had the objective to touch the tomato when the end-effector approached the fruit to grasp it, by doing this it was possible to measure the force applied in order to avoid the end-effector to damage the tomato with its structure before closing the fingers.

3.2 End-Effector Coupling

The end-effector's coupling structure was one of the last structures to be designed. The design of this part was delayed, since it was convenient to wait for the major parts of the gripper to be printed and assembled, so it would not be needed to redesign the coupling structure on each modification of the original design.

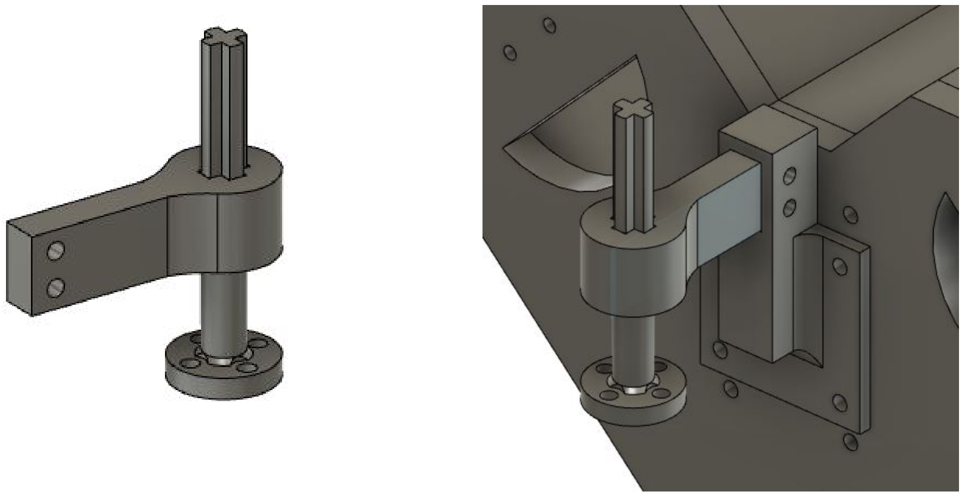
The requirements for the coupling structure consisted in being capable of holding the end-effectors weight without breaking or compromising the success of the operation, and also it needed to rotate around the z axis in order to improve the manipulator's movements to reach for the fruits from different horizontal paths, increasing, in this way, the manipulator's manoeuvrability.

Having the above in consideration, the structure was composed by four 3D printed parts and a servomotor. The main component consisted of a 'U' shaped piece that contained the servomotor inside and it was bolted to the manipulator's arm (Figure 3.5(a)). The servomotor contained an axis rod extension (Figure 3.5(b)) allowing it to hold the part that would work as a bridge between the servomotor and the gripper's main structure. The bridge was secured to a frame screwed to the backplate of the gripper (Figure 3.6).



(a) ‘U’ Shaped Piece (b) Axis Rod Extension

Figure 3.5: Coupling Structure Main Component (Left) And Servomotor Axis Rod Extension (Right)



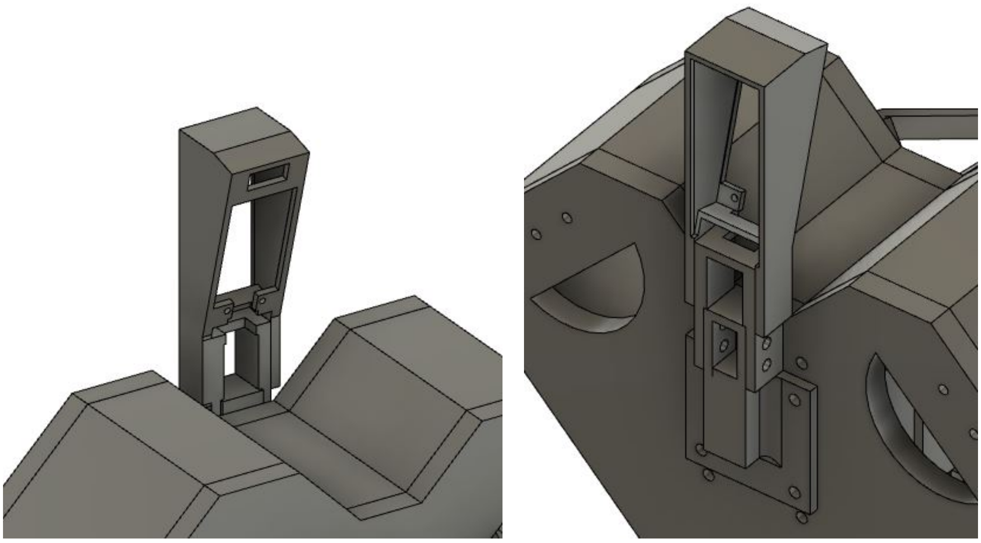
(a) Bridge (b) Backplate Frame

Figure 3.6: Coupling Structure Bridge (Left) And Backplate Frame (Right)

3.3 Arducam Support

The perception module's Arducam needed to be positioned on the gripper, allowing the camera to capture the frame containing the tomato to be harvested. To achieve this, the preferred place would be above the gripper to be possible to start the detection after the manipulator placed the end-effector in front of the fruit. However, there are some difficulties on this approach. Firstly, it was crucial to place the camera without needing to modify the gripper's main structure design or drilling extra holes on it. Also, the frame captured should not contain additional objects, for example parts of the gripper, which could compromise the efficiency of the perception module.

Analysing the structure, it was considered the best method would be to design the Arducam support to be capable of attaching to the backplate frame of the coupling structure (Figure 3.7). This choice fitted the requirements, since the camera would be positioned above the gripper's structure and would not capture it, nor the fingers. Also, this makes use of the already existing holes, without needing to redesign already mounted parts of the end-effector.



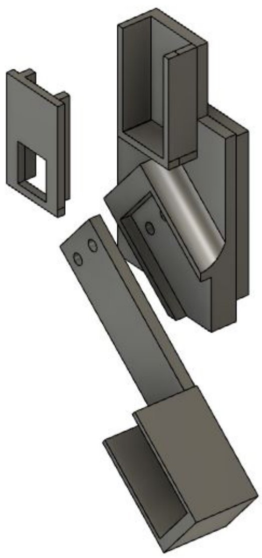
(a) Front (b) Back

Figure 3.7: Arducam Support Front View (Left) And Back View (Right)

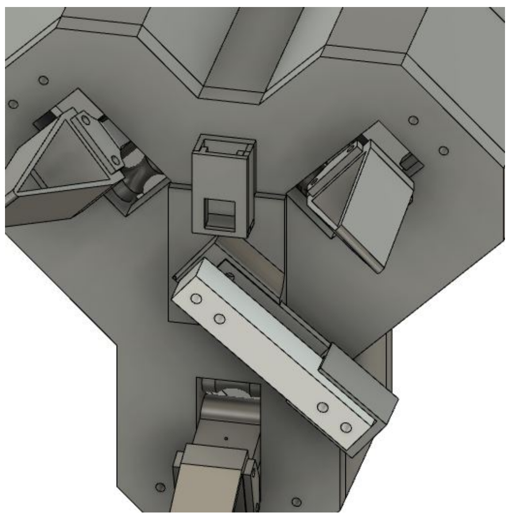
3.4 Load Cell & LiDAR Support

The load cell was expected to be placed on the centre of the front plate of the gripper, since the work principle consisted on sensing the tomato bumping against the end-effector when the manipulator did the approach to initiate the harvest of the fruit. However, the LiDAR needed to be on the front of the end-effector. The LiDAR needs to be positioned facing the detected fruit to confirm its presence at a pre-defined distance to the end-effector. To guarantee the distance is measured correctly, the ideal position would be near the centre of the gripper's front plate; however, this place would be already occupied by the load cell.

The solution to this problem consisted of designing a structure capable of holding both the components near the centre of the gripper without compromising any of their respective tasks. Avoiding drilling additional holes on the gripper, the structure was also designed for the purpose of covering the servomotor window on the front plate (Figure 3.8).



(a) Support Parts



(b) Assembly on The Gripper

Figure 3.8: LiDAR & Load Cell Support (Left) And The Component Assembled on The Gripper (Right)

Chapter 4

Electrical Design of FruitGrip

This chapter contains a short description of the electric hardware used on the FruitGrip. The components are divided in two major parts of the end-effector: the gripper and the perception module. The work principle of each component is explained, as well as the characteristics that contributed to their choice.

4.1 Gripper

RP2040 Microcontroller - Raspberry Pi Pico

The microcontroller used to control the different parts of the end-effector was the RP2040[1] (Figure 4.1), the 32-bit microcontroller released by Raspberry Pi in 2021 integrated on the Raspberry Pi Pico board[2]. The choice was mainly made because of the features of this microcontroller, which shows to be a powerful chip in comparison to other existent microcontrollers, for example the Atmega328p used on the Arduino UNO[3]. The RP2040 benefits, for example, of having a C/C++ toolchain, dual-core, 264 KB of SRAM and 2 MB of flash storage[4]. Also, it has a high number of availability in stock, while other types of microcontrollers are currently suffering from low stock during the realization of this work. The low price of the RP2040 boards, compared to other types (the Raspberry Pi Pico costs around 5 €, while the Arduino UNO costs around 20€) was also another determinant factor for the decision.

[1]Raspberry Pi. Buy a rp2040. URL: https://www.raspberrypi.com/products/ rp2040/
[2]Raspberry Pi. Buy a raspberry pi pico. URL: https://www.raspberrypi.com/products/raspberry-pi-pico/
[3]Arduino uno rev3. URL: https://store.arduino.cc/products/arduino-unorev3
[4]rp2040 - raspberry pi. URL: https://datasheets.raspberrypi.com/rp2040/rp2040-product-brief.pdf

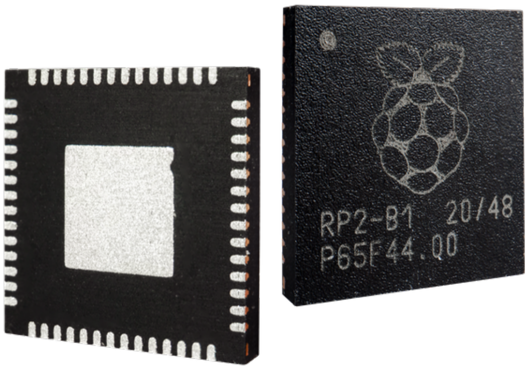


Figure 4.1: RP2040 Chip - Taken From Raspberry Pi Website[1]

ABRS-8034DH Servomotor

The ABRS-8034DH servomotor[5] (Figure 4.2) was the one used on the gripper for pulling the strings attached to the soft fingers and on the end-effector/manipulator coupling to allow the rotation around the vertical axis. There was the need to choose a servomotor with an angle range capable of closing the three fingers completely, and also able to exert the necessary torque to move the whole end-effector. This model has a range from 0° to 280° and an operating voltage range from 7.4 V to 12 V. The servomotor, when plugged to a 12 V source, can also apply up to 120 kg/cm[5]. Having the above in consideration, this type of servomotor seemed to be a capable choice to integrate in the system, fulfilling both the objective of pulling the strings to close the gripper's fingers and moving the end-effector on the coupling structure.

Figure 4.2: ABRS-8034DH Servomotor - Taken From Hobby King Website[5]

[5]Robostar abrs-8034dh 280° digital metal gear high voltage robot servo 120kg/0.38sec/83g. URL: https://hobbyking.com/en_us/robostar-abrs-8034dh-280-digital-metal-gear-high-voltage-robot-servo-120kg-0-38sec-83g.html

Load Cell & HX711 Module

The load cell (Figure 4.3) chosen to detect the presence of a tomato on the gripper and control the movement limit of the end-effector was capable to sense up to 1 kg applied to its structure. This choice was made having mainly two factors in consideration. Firstly, it was not necessary to have a load cell capable of sensing weight higher than 1 kg, since the weight applied by the end-effector on the tomato should not cause damage to the fruit. The other factor was the size of the load cell, which should not be excessive to not interfere with the fingers closing but also should be bigger enough to guarantee a contact surface with the fruit. The output signal from the load cell has low magnitude, in order to guarantee a correct reading from the sensor it is necessary to have an amplifier between its output and the microcontroller's input pins. To amplify this signal and also to convert from analogue to digital was used the HX711 module[6] (Figure 4.4). This module is then connected to a pin of the board to receive a clock signal and another GPIO pin of the board to send the information, a bit for each clock cycle.

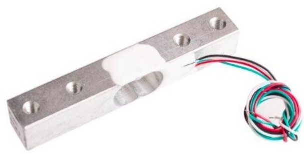


Figure 4.3: Load Cell - Taken From Electrofun Website[7]

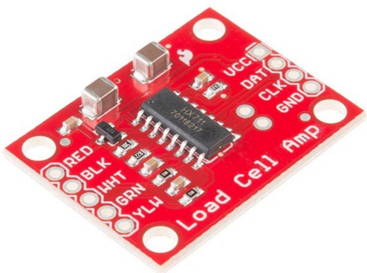


Figure 4.4: HX711 Module - Taken From Botnroll Website[6]

Force Sensor

The force sensor used consisted on a Force Sensing Resistor (FSR) (Figure 4.5) that decreases its resistance when a force is applied to the surface. The force sensitivity of this sensor was up to around 10 N. When compared to a load cell, this type of sensor is more inaccurate, suffering deviations from the correct value and being less precise[8]. However, to sense the force applied by

[6] Módulo amplificador para células de carga - hx711. URL: https://www.botnroll. com/pt/forca-pressao-vibracao/1638-modulo-amplificador-paracelulas- de-carga-hx711.html

[7] Sensor de peso 1kg célula de carga. URL: https://www.electrofun.pt/sensoresarduino/celula-carga-1kg-sensor-peso

[8] Force sensitive resistor hookup guide. URL: https://learn.sparkfun.com/tutorials/force-sensitive-resistor-hookup-guide?_ga=2.78027863.237030943.1650974973-662338554.1646775835

the lower finger of the gripper on the fruit, the FSR was the most fitting solution when analysing the specific situation.

The compromising factor when choosing a sensor was related to the structure design. A load cell would be difficult to combine with the soft fingers without altering the fingers' physical characteristics, since this type of sensors is rigid and would stop the finger from matching the fruit's shape. The FSR structure is flexible and could be easily glued to the surface of the finger, bending with the movement of the latter. Since the objective was to understand when the finger had successfully grasped the fruit, not measuring exactly the force applied but a range of values, the sensor being not as accurate as a load cell would not compromise the expected result of the task.

Figure 4.5: Force Sensing Resistor - Adapted From Botnroll Website[9]

Flexible Sensor

The flexible sensor (Figure 4.6) is a sensor which resistance varies according to the angle of the sensor's strip, having the lowest resistance when flat. This technology is patented by SpectraSymbol[10] and it was developed to be used on the Nintendo Glove[11], which objective was to detect the finger's movement (open/close).

Having in consideration the original purpose of these sensors, it is possible to understand the usability on the gripper's actuation. While the fingers are closing to grasp the fruit, they deform, matching the shape of the embraced object. Placing the flexible sensor on the back of the lower finger it was possible to calculate the angle made by the deformed finger and with that value estimate if the fruit was successfully secured.

Since the flexible sensor is not very accurate and the readings sometimes suffer interference resulting in false values, it was not possible to rely only on the value obtained. The results of the

[9] Sensor de força resistivo (fsr) pequeno 4mm. URL: https://www.botnroll.com/pt/ forca-pressao-vibracao/390-sensor-de-forca-resistivo-4mm.html

[10] Resistive flex sensors: Spectra symbol flex sensors, Sep 2021. URL: https://www.spectrasymbol.com/product/flex-sensors/

[11] Flex sensor hookup guide. URL: https://learn.sparkfun.com/tutorials/flex-sensor-hookup-guide

flexible sensor were always combined with the information gathered by the FSR to understand if the tomato was embraced without causing damage.

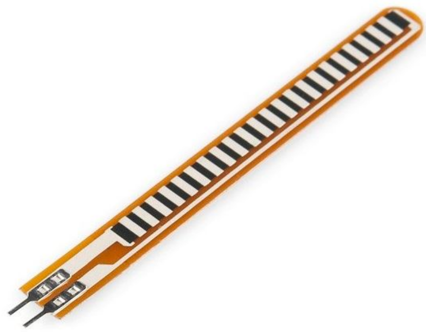


Figure 4.6: Flexible Sensor - Taken From Botnroll Website[12]

4.2 Perception Module

Arducam Pico4ML

The board used for the tomato and peduncle detection was the Arducam Pico4ML[13] (Figure 4.7). This board is also based on the RP2040 microcontroller, having a configuration similar to the Raspberry Pi Pico and, apart from the camera module (the monochromatic QVGA HiMax HM01B0), it also contains an LCD SPI display and a IMU module (ICM-20948)[13]. Combining with the above, the board also has the advantage of being reasonably cheap for the specifications provided.

[12]Sensor de flexibilidade 2.2"/ 56mm. URL: https://www.botnroll.com/pt/forcapressao-vibracao/389-flex-sensor-22.html

[13]Pico4ml: Raspberry pi rp2040 based board for machine learning. URL: https://www.arducam.com/pico4ml-an-rp2040-based-platform-fortiny-machine-learning/

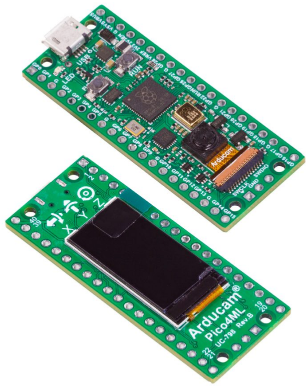


Figure 4.7: Arducam Pico4ML - Taken From Arducam Website[13]

VL53L1X - Time-Of-Flight Ranging Sensor

The VL53L1X sensor[14] (Figure 4.8) works as a LiDAR, emitting a laser and receiving the reflected beam to measure the distance to the object in front of it. This sensor was used with the objective to measure the distance to the tomato detected by the manipulator on the beginning of the task, detecting if this one disappears from the front of the end-effector, which results on the reset of the task. The VL53L1X has the advantage of being compact and integrated on a small board, allowing to be easily placed on the structure of the tool. The range admitted by this sensor to accurate measure the distance is up to 4 m and 50 Hz[14].

Figure 4.8: VL53L1X Sensor - Adapted From The Data Brief on ST Website[15]

[14] Vl53l1x. URL: https://www.st.com/en/imaging-and-photonicssolutions/vl53l1x.html
[15] Vl53l1x-satel. URL: https://www.st.com/en/evaluation-tools/vl53l1xsatel.html#overview

Chapter 5

FruitGrip - High Level Software Architecture

On this chapter, it is explained the software that was implemented on both the gripper and the perception module of the end-effector. There are also included the details of the state machines used, as well as the configuration of the external platforms needed to complete the logic implemented, such as the platform used to train the neural network of the perception module.

5.1 Gripper Control

The gripper control was built in state machine structure. This method allowed the control to work in a non-stop loop, while doing all the steps needed repeatedly to harvest successfully. In order to optimize the process, since the gripper control would need to communicate with both the manipulator and the perception module to complete some steps, the code was divided in two state machines.

Main State Machine

The main state machine (*state_control*) (Figure 5.1) consisted on the general control of the gripper and the transition between the main steps of the harvesting process.

In the beginning of the cycle, the control is continuously waiting to receive messages from the manipulator to control the coupling servomotor. The manipulator breaks this loop by confirming the servomotor is in the desired position, which allows proceeding to the next state. The next state is responsible for the whole detection step, containing the "detection state machine" and only exits this state after the detection is complete or if it was not possible to do it. In case the detection is not possible, the state machine goes back to the beginning and resets the cycle, otherwise it transits to the next state where it communicates with the manipulator to approach the tomato for grasping. The following states are all related to the mechanical actuation of the gripper in doing the task of safely grasp the tomato and drop it in the end, while maintaining communication with the manipulator.

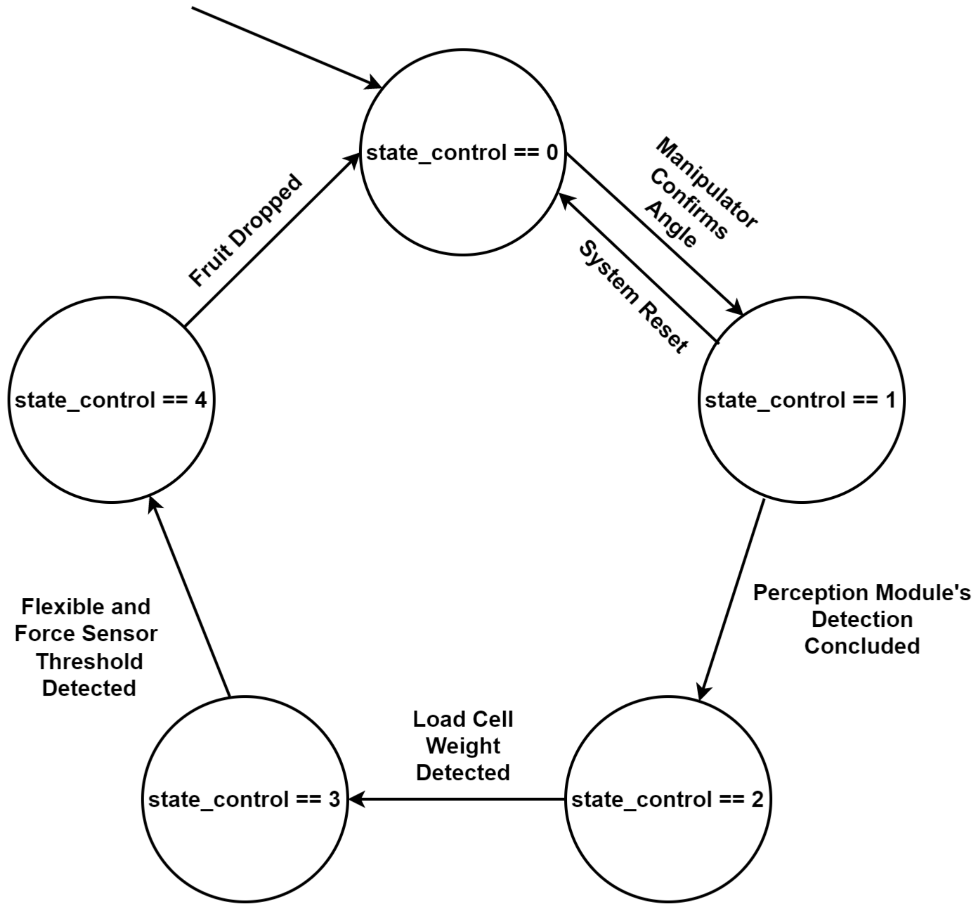


Figure 5.1: Main State Machine

Detection State Machine

The state machine auxiliary to the detection step (*state*) (Figure 5.2) is implemented inside the main state machine. The role of this state machine is to ensure the communication with the perception module while the image classification is being done on this module. It starts by ordering the perception module to initiate the code cycle and, after starting, it blocks on a state which does not have actions but works as a bridge to the different following states available. From this state, the next one on the cycle depends on the message sent by the perception module via Universal Asynchronous Receiver/Transmitter (UART). The states containing actions will, essentially, send commands to the manipulator to adjust the arm's position.

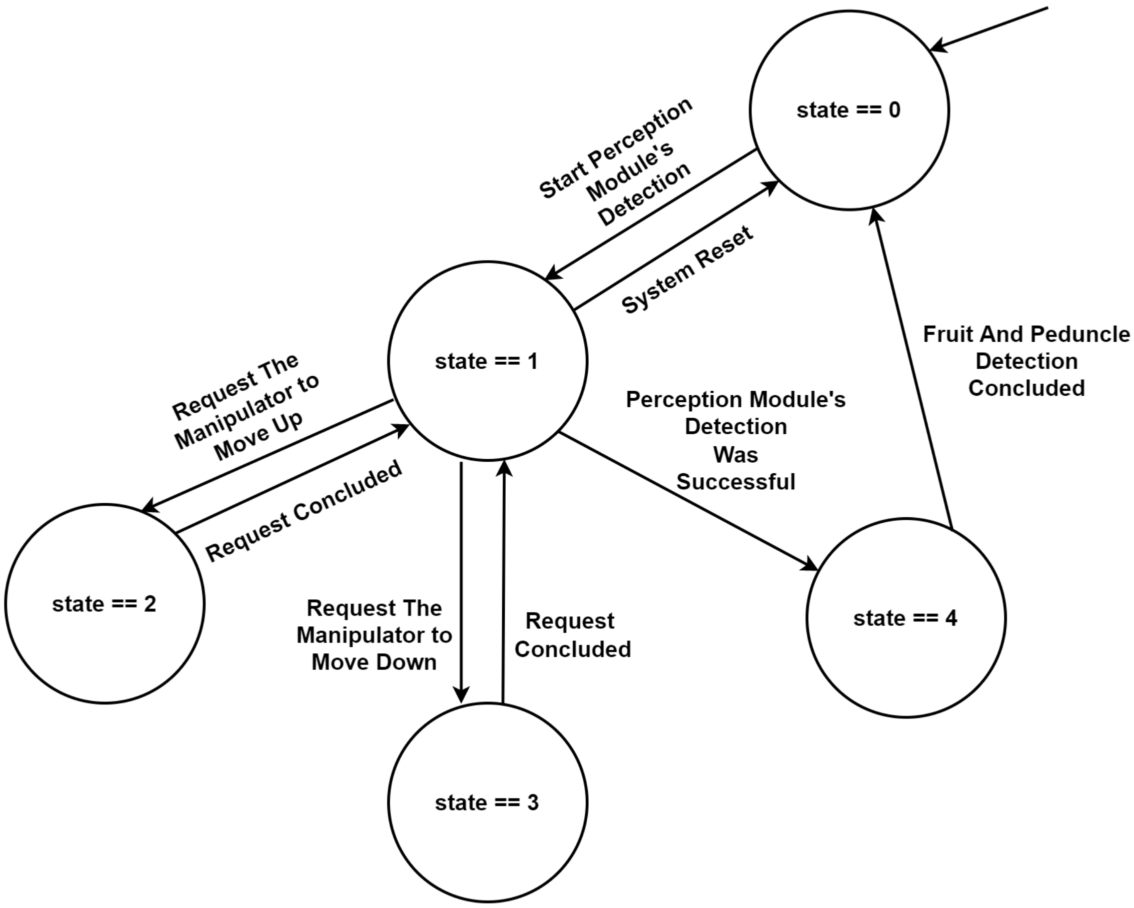


Figure 5.2: Detection State Machine

5.2 Perception Module

The perception module of the end-effector was based on the Arducam Pico4ML. The objective was to confirm the object in front of the end-effector after the manipulator's positioning was a tomato, and to give information to the end-effector on how the manipulator should adjust its height to safely approach the tomato for grasping.

To achieve the perception module's purpose, there was the need to implement an algorithm to detect the desired object in the image captured. Object detection is an algorithm capable of detecting multiple objects on the same image and, since it is computational heavy, the Arducam Pico4ML could not handle this algorithm. Considering this, there was the need to implement an image classification divided in two different frames, which requires less memory, since it classifies each image with the label of the predominant object in the image. The first frame was responsible for confirming the presence of a tomato and, after that classification, the frame above was responsible for detecting the presence of a peduncle. The image classification algorithm was implemented considering three distinct types of labels, images containing tomatoes, containing peduncles and containing "field", which consisted of photos of the surroundings that can be normally found near tomatoes, such as groups of leaves.

Image Classification - Edge Impulse

Edge Impulse[1] is a free platform for machine learning development. This platform was used to generate and train an impulse based on a Neural Network (NN) to be deployed on a RP2040 chip, more specifically on the Arducam Pico4ML module. This platform has the main advantage of not requiring hardware to train the impulse, since it is done online. Edge Impulse has the advantage of containing an intuitive and simple User Interface (UI) and documentation containing basic tutorials, which allows developing the impulse without previous experience in machine learning. The platform also calculates the memory needed to run the impulse and an approximate value for the inferencing time, depending on the board selected by the user at the beginning of the project creation.

The dataset used consisted in a training set with 284 items and a test set of 63 items, the images were distributed as shown in Table 5.1 and Table 5.2, respectively. This dataset was the final version, which achieved better results when testing. However, previously to this version there were several tested datasets with slightly different images and variation in quantity that were continuously modified until the last version to guarantee the well functioning of the classifier.

Table 5.1: Training Set - Distribution

Label	Number of Images
Tomato	88
Field	112
Peduncle	84

Table 5.2: Test Set - Distribution

Label	Number of Images
Tomato	25
Field	21
Peduncle	17

Edge impulse suggests the image size to be of 96×96 pixels or 160×160 pixels in order to achieve optimal accuracy on the learning blocks. However, even the smaller size was enough to overload the RP2040 memory when running the classifier together with the needed code for the perception module's logic. The image size used was 64×64 pixels and the learning block used was *Transfer Learning*, which was the recommended on the platform for image classification (Figure 5.3).

Due to being a small NN, in order to use less memory to optimize the perception module's firmware, the architecture used was the MobileNetV1. The NN was trained in 120 epochs, where each epoch consists of a complete training cycle, had a learning rate of 0.0005, which defines how much the weights are being adjusted during training, and a validation set size of 20% of the training set to evaluate the classifier obtained after training. These values were chosen by varying

[1] Edge impulse. URL: https://www.edgeimpulse.com/

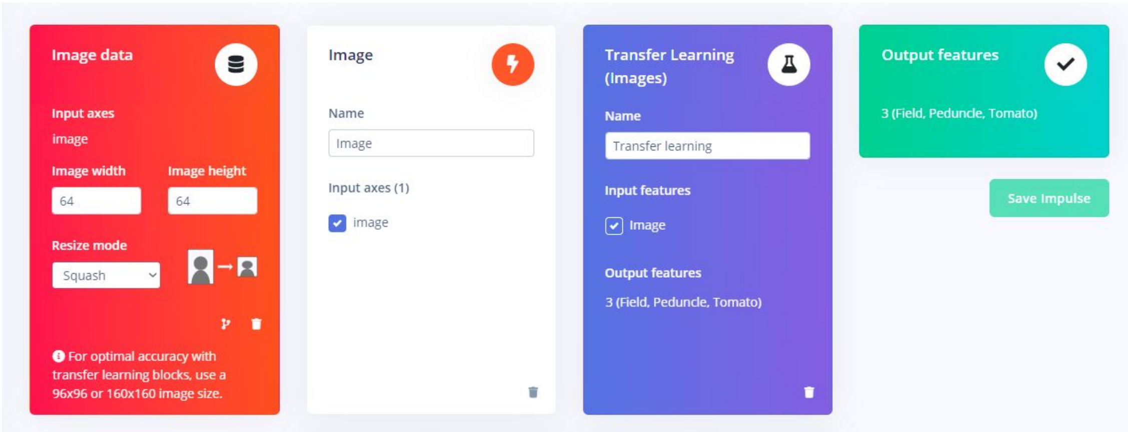


Figure 5.3: Impulse Design - Edge Impulse

them and testing the different classifiers, starting with the recommended values by the platform and checking the performance of the classifier on each trial[2]. It was also used data augmentation, which increased the diversity of the dataset by applying random transformations on the images, such as rotations, image cropping and brightness variations[3]. The objective of this setting is to increase the accuracy of the classifier. The results obtained on the validation set after the model training are the following, shown on Table 5.3, resulting on the confusion matrix of Table 5.4 and the remaining metrics values calculated of Table 5.5 [17].

Table 5.3: Model Training - Validation Set - Results

Parameter	**Result**
Accuracy	89.5 %
Loss	0.19

Table 5.4: Model Training - Validation Set - Confusion Matrix

	Field	Peduncle	Tomato
Field	100%	0%	0%
Peduncle	0%	88.2%	11.8%
Tomato	0%	20%	80%
F1 Score	100%	83%	84%

[2]Neural network settings. URL: https://docs.edgeimpulse.com/docs/edgeimpulse-studio/learning-blocks/transfer-learning

[3]Data augmentation - tensorflow core. URL: https://www.tensorflow.org/tutorials/images/data_augmentation

Table 5.5: Model Training - Validation Set - Metrics

	Field	Peduncle	Tomato
Precision	1	0.79	0.89
Recall	1	0.88	0.8
Specificity	1	0.33	0.92
False Positive Rate	0	0.1	0.05
False Negative Rate	0	0.12	0.2
Wrong Classifications Rate	0	0.11	0.11
Correct Classifications Rate	1	0.89	0.89

The errors that occurred on the NN training corresponded to false detections on the tomato and peduncle labels of the validation set. There were two tomato detections on the 17 images which the actual label was peduncle. The opposite situation also occurred, where four out of 20 images labelled as tomato were predicted as being peduncle.

Analysing some examples of the cases mentioned before, it was possible to assume some of the causes that might have led to the errors occurred. As it is possible to observe on Figure 5.4, the image used, apart from having low resolution, contains a tomato which is not as round as usual and part of the tomato's peduncle. These aspects could have contributed to the false detection, as there are features that usually are not present on the tomato images.

On the opposite situation, represented on Figure 5.5, this example of a peduncle image detected as tomato can possibly be explained due to the presence of additional elements on the image that are not peduncles. Dividing the image in three parts of equal size, it is possible to observe about one third of the image corresponds to almost a complete tomato that was caught on the image frame and there's also part of another tomato in the bottom right corner of the image. The presence of these tomatoes could have led the classifier training to give a false detection, even being the peduncle the bigger and more centred object in the frame.

Figure 5.4: Case of Tomato Detected as Peduncle

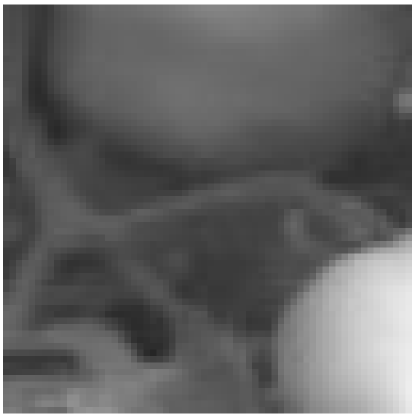


Figure 5.5: Case of Peduncle Detected as Tomato

The model testing was done with the test set previously shown in Table 5.2. This data set contained images similar to the images on the training set, whose system had not evaluated before. The results, as shown in Table 5.6, were slightly better than the ones obtained with the validation set in the training phase, resulting in higher values of precision, recall and correct classifications rate in some labels, as shown in Table 5.7. This allows to conclude the model is not overfitting, since it classifies unknown images with efficiency identical or even higher as it classified the validation set. On the 63 images of the test set, the classifier did not achieve the expected result in only three images, where it considered uncertain the classification for two of the peduncle images and one image of a tomato, which did not reach the threshold implemented by the platform for any label.

Table 5.6: Model Testing - Confusion Matrix

	Field	Peduncle	Tomato	Uncertain
Field	100%	0%	0%	0%
Peduncle	0%	88.2%	0%	11.8%
Tomato	0%	0%	96%	4%
F1 Score	100%	94%	98%	

Table 5.7: Model Testing - Metrics

	Field	Peduncle	Tomato
Precision	1	1	1
Recall	1	0.88	0.96
Specificity	1	1	1
False Positive Rate	0	0	0
False Negative Rate	0	0.12	0.04
Wrong Classifications Rate	0	0.03	0.02
Correct Classifications Rate	1	0.97	0.98

After the NN was trained and tested, it was deployed as a C++ library. The library was then included to the code of the image acquisition to classify the frames captured by the camera of the

Arducam Pico4ML module.

Image Acquisition And Classification

The image acquisition was implemented on the Arducam Pico4ML. The code used the multicore feature of the RP2040 microcontroller and initially was overclocked to 250 MHz to achieve better results without blocking the rest of the program. However, by overclocking the microcontroller, it was not possible to exchange messages via UART with the gripper control, since the frequency change would influence the bitrate of the transmission, resulting in errors.

The core 1 was in control of the communication between the RP2040 and the HM01B0 camera and contained the data received from it. This data consisted of a value ranging from 0 to 255 for each pixel, corresponding to a gray level, since the sensor used is monochromatic. However, the classifier was supposed to read a float array containing the value of each pixel in RGB representation. Having this in consideration, it was needed to convert the grayscale value received to RGB. A gray level is represented in RGB with all the components having the same value as the original value received (i.e. R = G = B = Gray Level), so the data acquired was converted to RGB giving each component the value of the pixel.

The core 1 also controlled the LCD of the Arducam Pico4ML via SPI, which was useful to better understand what was captured by the board camera at each moment without having the need to send the data via Universal Serial Bus (USB) to the computer.

The main function, running on core 0, was responsible to launch the multicore functions and run the image classifier of the C++ library generated by Edge Impulse. The classifier analysed the input and the NN predicted which label was more probable of being represented on the frame received, saving this classification on a struct containing the info about the results obtained. The core 0 also contained a state machine, responsible for the steps needed to conclude the image classification. This state machine acted as a *mirror* of the detection state machine on the gripper control, transiting not only when the conditions of both the LiDAR and the classification allowed it, but also when the messages sent to the gripper control were acknowledged via UART. On the first analysed frame, the classification value of the probability of being a tomato was evaluated, considering how many consecutive times it occurred, by establishing a minimum value for occurrence and classification. In case the evaluation concluded that the label expected was in the image, the perception algorithm transits to the frame above and initializes the search for peduncle on the frame.

5.3 Communication Between Gripper Control And Perception Module

There was the need to establish a communication between the Arducam Pico4ML and the control module. The objective was for the Arducam Pico4ML to acquire the images from the camera, detect the tomato on one frame, detect peduncle on the frame above, and then pass information to the control module to actuate, initiating the grasping. In order to create a starting point for this objective, it was crucial to implement and test the communication between these two boards. Due to its simplicity and since it would only be necessary to pass simple and short commands, the two boards were connected by UART. It uses only two wires, to send and receive information, and as it is asynchronous it does not need a clock signal [18].

At first, the objective was to establish a communication sequence, with messages containing starting and ending characters. However, due to the limitations associated with the memory available at the Arducam Pico4ML and the time the image classification took to complete on each cycle, it was implemented a single character communication. This type of communication can easily suffer interference on the channel used, resulting in noise that can be interpreted as a character. Having that in consideration, most of the messages sent between the two microcontrollers had a confirmation from the receiver which consisted of the same character sent back to the origin.

The information passed had the following commands, which are shown in Table 5.8.

Table 5.8: Gripper-Perception Module Communication - Commands

Command	Instruction	From	To
s	Orders the perception module to initiate the image classification sequence	Gripper	Perception Module
u	Informs the gripper it is needed to increment the vertical position	Perception Module	Gripper
d	Informs the gripper it is needed to decrement the vertical position	Perception Module	Gripper
x	The request made by the perception module could not be fulfilled	Gripper	Perception Module
k	The image classification cycle was complete and the current end-effector position is appropriate to approach the detected tomato to harvest	Perception Module	Gripper
r	The classification could not be completed, which can be due to the classifier reaching the number of maximum tries to do the classification or the LiDAR stopping detecting the tomato	Perception Module	Gripper

5.4 Communication With The Manipulator

There was the need to establish a communication between the module controlling the end-effector's logic and the manipulator main computer. The objective of this communication was for the manipulator to give the orders to the end-effector to actuate and the last one to acknowledge those orders and request for adjustments on the manipulator height.

Since the computer controlling the manipulator is a Raspberry Pi 4B, which contains USB ports, it was possible to connect the gripper's Raspberry Pi Pico via this method to the manipulator computer and use the stdio functions to exchange characters.

In order to guarantee the communication between these two components was reliable and fault proof, it was established that the information passed followed a communication sequence, containing starting and ending characters. By implementing this sequence, both parts communicating can understand if the information received was intentional or noise in the communication channel.

The information passed had the following commands, which are shown in Table 5.9.

Table 5.9: Manipulator-End-Effector Communication - Commands

Command	**Instruction**	**From**	**To**
ang*xxx*d	Change the angle of the servomotor that connects the end-effector to the manipulator *xxx* is the angle value with a range from 0 º (000) to 180 º (180)	Manipulator	End-Effector
angok}\n	Confirm that the angle of the servomotor that connects the manipulator and the end-effector is on the requested position	End-Effector	Manipulator
angokff	Order the end-effector to proceed with the operation, confirming the coupling servomotor is on the desired position	Manipulator	End-Effector
movup}\n	Request to increase the vertical position of the manipulator's arm	End-effector	Manipulator
movupff	Confirm the vertical position of the manipulator was increased has requested by the end-effector	Manipulator	End-Effector
movdw}\n	Request to decrease the vertical position of the manipulator's arm	End-effector	Manipulator
movdwff	Confirm the vertical position of the manipulator was decreased has requested by the end-effector	Manipulator	End-Effector
dropt}\n	Confirm to the manipulator the end-effector is ready to receive the order to drop the tomato	End-Effector	Manipulator
dropokf	Order the end-effector to open the gripper and drop the tomato	Manipulator	End-Effector
adjef}\n	Request the manipulator to bring the end-effector closer to the tomato	End-Effector	Manipulator
adjok}\n	Request the manipulator to stop, since the tomato is already touching the load cell	End-Effector	Manipulator
reset}\n	Informs the manipulator that the image perception module was not successful and the harvesting task was reset	End-Effector	Manipulator

Chapter 6

Test And Validation

The test and validation phase was done with bottom-up approach implemented during the whole dissertation. The different parts of the end-effector's system were tested individually to be corrected and validated. Only after validating the correct functioning of the different subsystems, they were integrated on the main system.

6.1 Gripper Movement

The first part tested and validated was the gripper of the end-effector. On this subsystem, it was possible to distinguish different components that needed to be tested separately before being joined together. The mechanism to control the fingers' movement with the servomotor was the first step completed during the project. The fingers and their structure were assembled on the gripper front plate and the strings were attached to the servomotor. This part was tested by implementing a Pulse Width Modulation (PWM) signal on the RP2040 to control the servomotor. The signal's Duty Cycle was then modified by serial input on a computer's keyboard, changing the servomotor's axis position. This allowed to close and open the gripper, testing on objects with different shapes and sizes to validate it was capable of grasping them and have the necessary strength to hold without letting them fall.

After this validation, the next step was implementing a small library containing the functions necessary to control the servomotor's PWM without needing to invoke all of them on the main code.

6.2 End-Effector/Manipulator Coupling

The servomotor used on the structure connecting the end-effector and the manipulator is the same type of servomotor used on the gripper to close the fingers. To control the motor a PWM signal is required to adjust the axis position, as previously explained. Since the task of gradually changing the servomotor position to its limits, by pressing two different keyboard keys, had already been tested, the test needed for this specific part consisted on changing to an angle value requested by the manipulator's controller. In order to fulfil that purpose, two functions were created on the servomotor library previously built. The function *ang2micro* consisted on a linearization of the PWM values in order to the correspondent angle, the objective consisted in having as input the angle requested by the manipulator and it would return the PWM value to change the servomotor axis to the correct position. The function *setmicros_prog* was an improvement on the previous function used to change the servo position, while the first function developed had as input the PWM value desired, this new function also received the current PWM value on the servomotor and the objective was to change the applied PWM with a determined rate (defined inside the function by a timer), so it was not changing instantaneously when the function was called. This last function was noticed to be useful even for the gripper's servomotor, replacing the original function used to close the fingers. The transitions while closing and opening the gripper's fingers were smoother, resulting in an increase on the precision of the gripper when grasping the tomatoes.

6.3 UART Communication

Since the first steps were tested before having the Arducam Pico4ML available, the procedure was to establish a communication between two Raspberry Pi Pico as both boards share the RP2040 microcontroller, which would result in a similar implementation.

One Raspberry Pi Pico was connected by USB to a computer and by UART to the second Raspberry Pi Pico that was controlling the gripper. In the computer, the keyboard was used to send commands to open and close the gripper, which was confirmed by serial prints that the first Raspberry received the commands correctly. The commands were sent from the first Raspberry Pi Pico to the second by UART, resulting on the gripper to close and open accordingly.

Later, the communication by UART was implemented on the Arducam Pico4ML having as structure the code tested on this step between the two Raspberry Pi Pico and assuring the communication commands previously stated.

6.4 USB Communication

The USB communication to be tested had the objective to simulate the communication between the end-effector and the manipulator's raspberry.

The test procedure consisted on establish a simple communication between the end-effector's Raspberry Pi Pico responsible for the gripper control and a computer with USB ports. The messages consisted of arrays of 8-bit characters, the character on each position was sent from the Raspberry Pico via the *putchar()* function and the receiving characters were read via the function *getchar()*. On the computer's end, there was a Putty terminal which printed the messages received and allowed to write the commands to send back to the end-effector.

After assuring the communication worked as expected, it was implemented the interpretation and sending of the commands previously defined to assure the liability of the communication protocol.

6.5 Load Cell

The load cell implementation followed two major steps to be considered ready to use on the end-effector.

For the first step, as mentioned before, a HX711 amplifier was used to read the values measured on the load cell, that have low magnitude, as a digital signal on a GPIO of the Raspberry Pi Pico. In order to achieve this, there was the need to create two functions. The first function to be called was the *InitHX711*, which simplified the process of initializing the clock and data pins as output and input, respectively, putting the clock signal with low value. The second function, *ReadHX711*, consisted on the procedure needed to be able to read information from the amplifier connected to the load cell. On the amplifier datasheet[1], it is possible to understand how this function needs to be implemented to retrieve information. It is stated, when not reading data, the clock signal should be low and the data signal should be high. When there is data available, the data signal goes low and then it is necessary to apply between 25 and 27 clock pulses to shift the data to be received, which will be received bit by bit on the data pin, starting on the Most Significant Bit (MSB). The 25^{th} pulse of the clock signal will pull the data signal back to high, reaching the end of the data transfer.

Considering these instructions, the function created waits for the data signal to go low and then on a *for* cycle it sends the clock pulses while saving the data received. In the end of the cycle, the function returns the value read of the weight on the load cell. However, the values read need to be calibrated to be interpreted as weight in SI units (g). For that purpose, in the second step of the load cell configuration it was made a linearization of the values, using a known weight and a reading without any weight on the load cell. Using these two references, the linear function was calculated to calibrate the load cell. The calibration was done with the load cell on a horizontal position to be possible to put the weight on top of it; however, the load cell would be positioned vertically on the end-effector to detect weight change when it bumped against the tomato. When it was positioned correctly, it was possible to notice the weight varied a few grams when compared to the horizontal position and there was the need to calibrate once more, adding the weight difference for the new position to achieve a value closer to the reality.

6.6 Force And Flexible Sensors

Both the force and flexible sensors have a similar implementation. These sensors are used on a voltage divider configuration and act as a variable resistor on the circuit. The output is then read on an Analogue to Digital Converter (ADC) pin of the microcontroller, without needing any extra integrated circuit or external amplifier to function.

The main difference between the two sensors is the data interpretation after it is read. While the voltage variation from the flexible sensors corresponds to an angle variation on the body of the

[1] 24-bit analog-to-digital converter (adc) for weight scales. URL: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf

sensor, in the case of the force sensor this voltage variation is caused by a weight applied to the tip of the sensor. Considering this, in a method similar to the one implemented on the load cell, there was the need to calibrate the values read by linearization to give the most accurate estimation of the parameters analysed. The force sensor was calibrated using a known weight to be possible to linearize the values obtained. However, to calibrate the flexible sensor, it was more difficult to obtain a correct estimation of the angle, since the calibration was done by bending the sensor's body to positions which are easier to predict the formed angle, such as 90 º, resulting in values not as accurate when compared to the force sensor.

The next step consisted on attaching both sensors to the lower finger of the gripper. The force sensor was glued on the top of the finger and the flexible sensor was glued on the bottom. After concluding this task, it was noticed that the values read differed from the values obtained before placing the sensors on the finger's surface. The reason for this change was related to the contact between the finger and the sensors, these being glued on the finger would slightly stretch when the finger moved, causing variations on the reading even when there was no object being grasped by the gripper. Considering this external error, there was the need to once more calibrate the logic implemented on the sensors, this was achieved by increasing the offset of the readings.

6.7 Image Classification

The perception module implemented on the Arducam Pico4ML had the first tests by acquiring images printed on paper as well as images on the computer screen. The first classification algorithm was not implemented to classify images also containing peduncles, but instead just containing tomatoes. This choice was made considering there were already available datasets containing tomatoes, while peduncles were not initially available as a dataset. Also, for testing purposes, it would be easier to detect only tomatoes, which have more details printed on paper than peduncles. Apart from needing only to detect tomatoes, there is the need to have more than one label to be used as comparison for the label of interest. In order to fulfil the requirements, the classifier used, developed on Edge Impulse, contained two different labels, tomato and field. The dataset used consisted of a training set with 150 items and a test set of 50, the images were distributed as shown in Table 6.1 and Table 6.2, respectively.

Table 6.1: First Classifier Training Set - Distribution

Label	Number of Images
Tomato	75
Field	75

Table 6.2: First Classifier Test Set - Distribution

Label	Number of Images
Tomato	25
Field	25

The code used verified if there were at least two close readings with probability higher than 60 % of being a tomato, and if the conditions met, it was assumed a tomato was on the front of the camera module. The algorithm was capable to classify images containing tomatoes. However, each classification took around 1700 ms.

Chapter 7

Results

To evaluate the viability of the end-effector built, there was the need to conduct several trials on the different components, as well as on the whole system.

On the end of the development period was expected the end-effector fulfilled several tasks related to movement, perception and sensory. As previous stated, the different parts separately were able to perform a simplified version of those type of tasks, yet the complexity increases when all the components have to work together to complete the harvesting task.

7.1 Hardware

The first step after validating all the different parts individually consisted in assembling the end-effector to its designed structure.

The different structural parts were mainly composed of Polyethylene Terephthalate Glycol (PETG) and Polylactic Acid (PLA). PETG was used essentially on parts requiring more strength due to the material's durability [19], such as the main structure of the gripper. PLA, being cheaper and highly available, was used on the remaining parts that sometimes required several test prints to adjust sizes and shapes to optimize their functioning, not requiring much strength associated, such as the LiDAR and load cell support.

After the assembly of the different parts (Figure 7.1), the end-effector was mounted on the coupling structure attached to the manipulator's arm (Figure 7.2). During the project, it was made an effort to opt for low weight materials, both on the structure and the remaining components needed, in order to not exceed the end-effector's weight to avoid overloading the manipulator's arm. The end-effector was attached to the manipulator successfully, being the latter able to support the additional weight added on its extremity. Still, a small bending on the manipulator's arm was noticed and caused by the end-effector. However, being this one a prototype, it is expected in the future to be more resistant, not being affected by the current end-effector's weight.

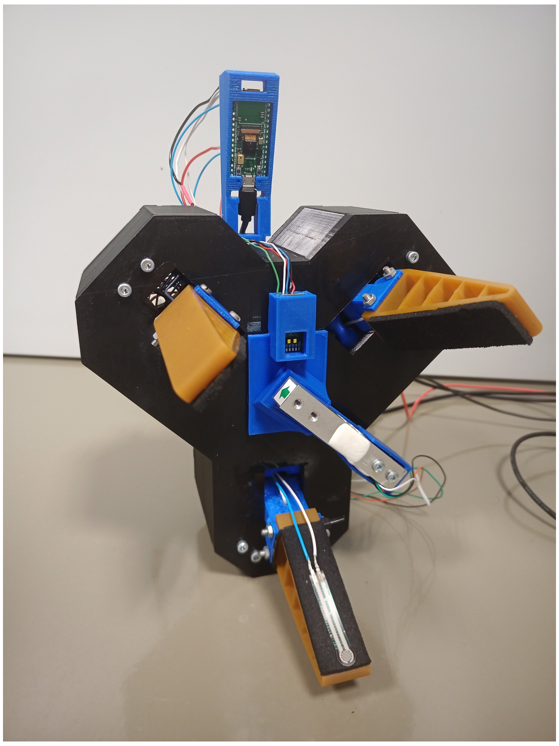


Figure 7.1: End-Effector Assembled

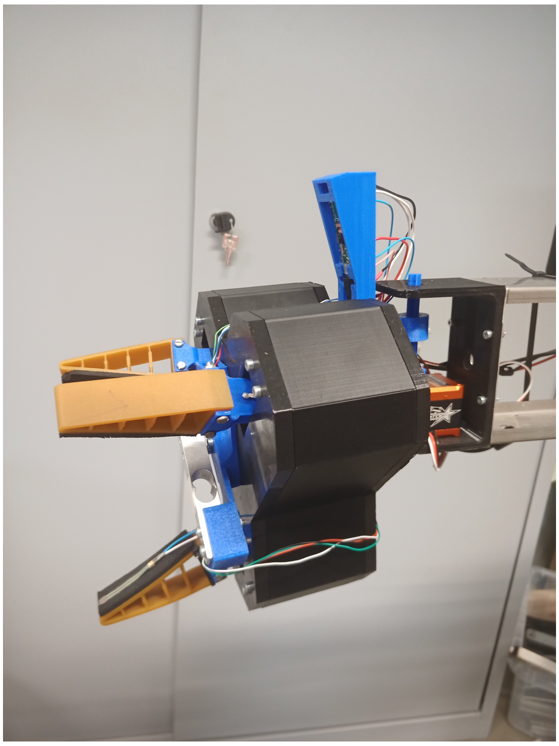


Figure 7.2: End-Effector Mounted

7.2 Perception Performance

On the validation section, it was noted the classifier took around 1700 ms to do a complete classification with two different labels. However, as previously stated, the complete classifier used consisted of three labels, containing also a label associated to the presence of peduncles. This classifier was noticeably slower than the first one, having an inference time of around 3550 ms (Figure 7.3).

This delay also affected the rest of the harvesting task, slowing down the whole process. In the perception module, the classifier must assume the detection of a tomato by doing two classification cycles with at least 60 % of probability of being a tomato in the image frame. Considering each classification cycle takes around 3550 ms, this process will take at least 7100 ms, having this timestamp increased when the probability is not very stable between measurements, requiring more than two classification cycles. In the second image frame, where the probability of containing a peduncle is classified, the evaluation method changes, not requiring two positive classifications. However, there was implemented a limit of ten tries for the end-effector to detect the peduncle and adjust the height of approach, where each trial runs the classifier cycle to analyse the probability of the frame containing peduncle. This chosen method is supported by one characteristic of the SCARA manipulator to which the end-effector was coupled. The manipulator also has a perception module to detect the tomatoes on the plant in order to initially choose which to approach. Considering this characteristic, it is assumed the manipulator positions the end-effector in front of it, not requiring the end-effector's perception module to find the fruit in the whole plant. However, it was implemented the tomato classifier on the first image frame as a double verification to guarantee the object in front is in fact a tomato, conducting the two classification cycles without a maximum limit of tries. In the case of the peduncle detection, it is crucial this one is accurate and limited in number of tries, since the manipulator does not previously know if the peduncle is reachable and uncovered, relying on the end-effector's gathered information. The end-effector is responsible to understand if there is a peduncle visible on the tomato to be harvested, avoiding to damage tomatoes nearby and working as a starting step for an additional tool to separate the peduncle and the fruit.

Overall, it is possible to estimate the complete classification time required by the perception module, considering the number of cycles done on each part of the algorithm, as shown in Table 7.1

Table 7.1: Perception - Classifier Time

Frame of Interest	**Minimum Number of Cycles**	**Minimum Time Necessary (ms)**	**Maximum Number of Cycles**	**Maximum Time Necessary (ms)**
Tomato	2	$\pm$ 7100	Not Defined	Not Defined
Peduncle	1	$\pm$ 3550	10	$\pm$ 35500

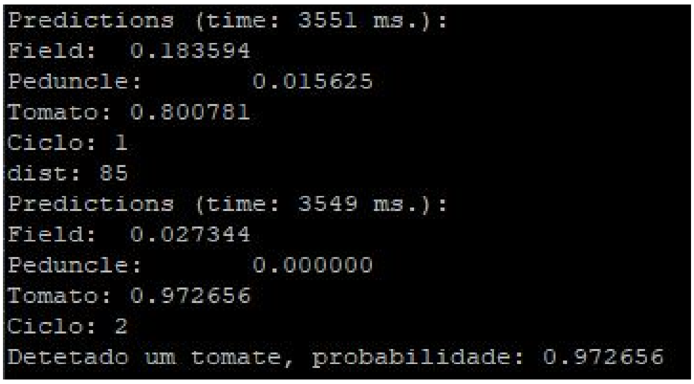


Figure 7.3: Classification Cycle Inferencing Time - PuTTY Terminal

Since a limit for the number of tries to do the tomato classification on the first frame was not implemented, it is not possible to define a maximum time for the perception module algorithm. However, it is possible to calculate the minimum time needed. Considering the minimum number of classification cycles needed to fulfil the perception module's tasks, the complete algorithm should take around 10 650 ms.

To analyse the perception's module performance, several classifications were executed, both for the tomato and the peduncle frame (20 classifications for each of the frames), and the results were registered. Executing this test allowed to understand how frequently the classifier only requires to run the minimum number of cycles, achieving the lower execution times and consequently the most optimized execution of the perception's module.The results obtained for the tomato and peduncle classification are organized on Figure 7.4 and Figure 7.5, respectively,

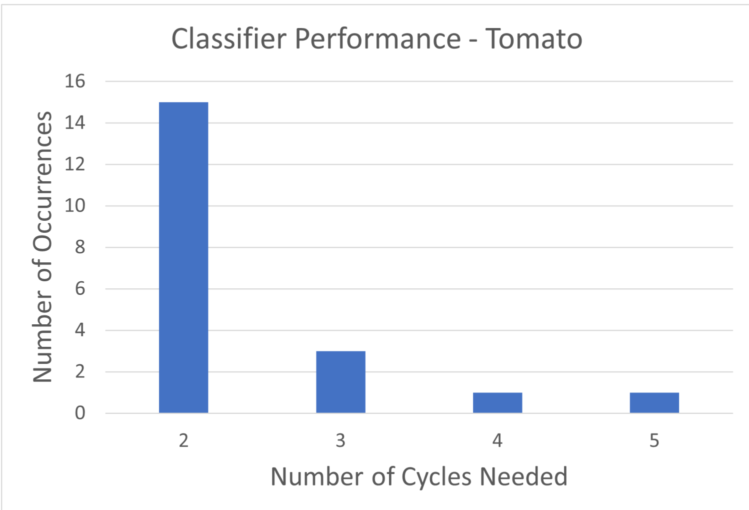


Figure 7.4: Chart - Classifier Performance - Tomato

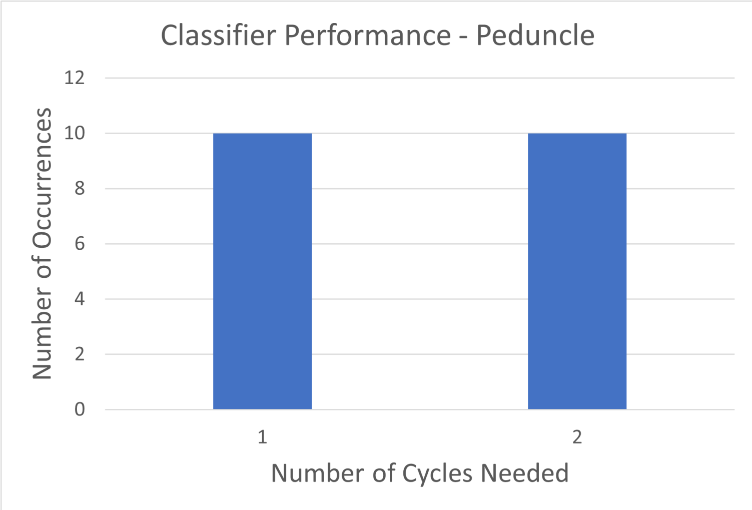


Figure 7.5: Chart - Classifier Performance - Peduncle

This test allowed to take a few conclusions about the performance of the perception module. The tomato classifier was able to detect the tomato on all the tries, not being stuck on the cycle. It was also possible to confirm its efficiency, since on a total of 20 tries, the classifier needed more than the minimum required (two cycles) to classify the frame as tomato on just five of the tries, achieving the optimal timestamp in 75 % of the times. On the peduncle classifier, being this one easier to be confused with the surroundings, since it contains less distinctive features, the number of cycles needed did not show tendency for a single value. However, on the 20 trials conducted, the number of cycles required to classify the frame as containing peduncle were divided equally between one and two, not reaching the maximum number of tries when the visual path to the peduncle is clear.

7.3 Grasping Performance

Both the fingers movement and the sensors readings were previously tested and validated individually. To evaluate the grasping performance, there was the need to conduct several trials in grasping tomatoes, considering both the capability of the fingers to hold the fruit and the sensors to do correct measures to avoid the tomato being crushed by the gripper closing.

Having this in mind, after the end-effector was mounted on the manipulator's arm, it performed several complete harvesting cycles. On each trial, the observed result was registered to analyse in detail the system's behaviour on the grasping task and reflect if there were improvements to be made to optimize the process.

On the first set of trials, it was concluded that the flexible and force sensors readings were actuating on the gripper's control for values higher than the needed to successfully hold the fruit, resulting in the gripper grasping excessively the tomato. Since this type of actuation can cause damage to the tomato, the data gathered on the tests was used to link the sensor's readings with the servomotor position. It was noticed, most of the time, the force sensor touched the tomato and started to have readings when the servomotor had a PWM with a pulse duration of 1.8 µs (on a range from 0.9 µs to 2.1 µs), and the lowest value read was around 34 g (Figure 7.6). The flexible sensor contained different readings for this same position on each trial, as well as oscillations between measures (Figure 7.7). In order to obtain a threshold capable to actuate on the control correctly, it was considered the average value given of all the trials done with a PWM with a pulse duration of 1.8 µs, which was around 15.465°.

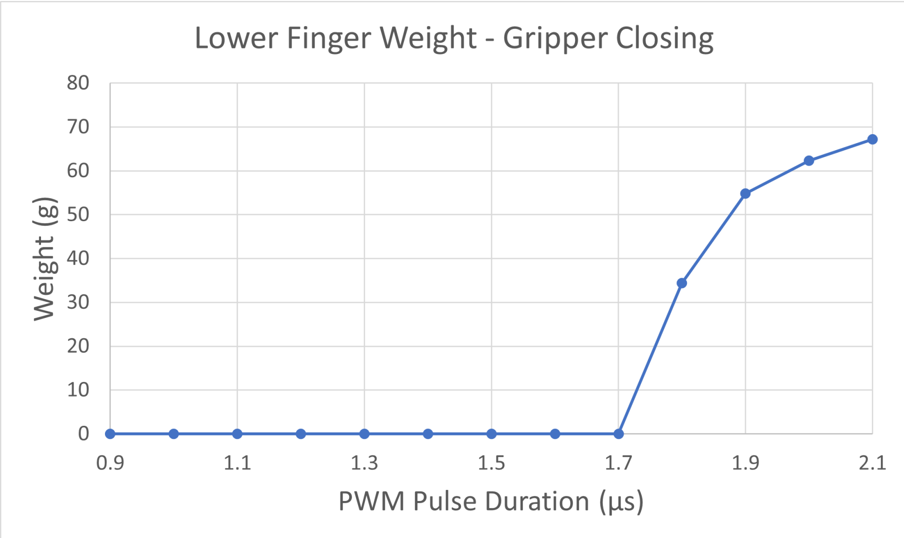


Figure 7.6: Lower Finger Weight - Gripper Closing

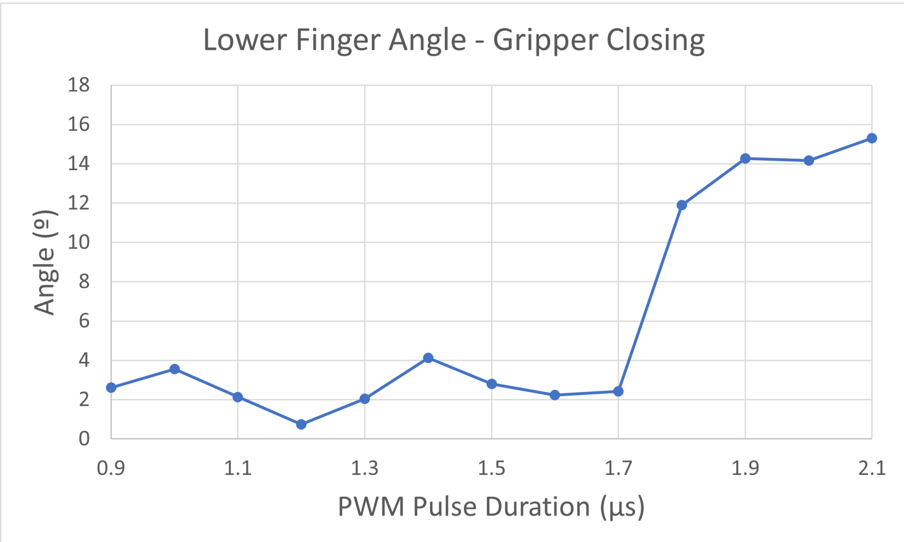
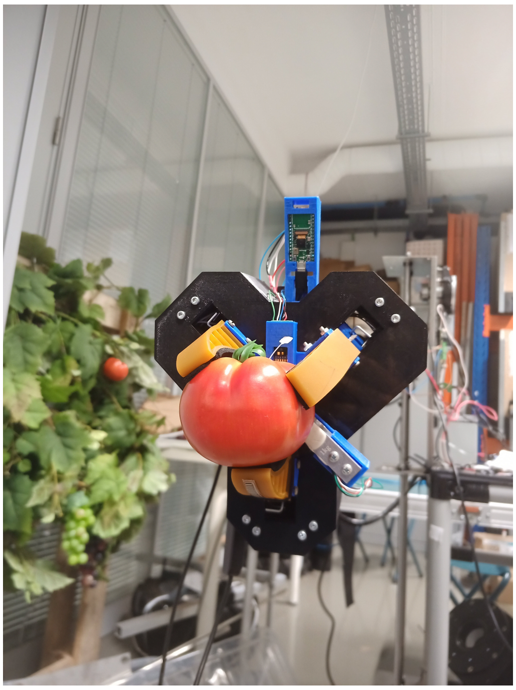


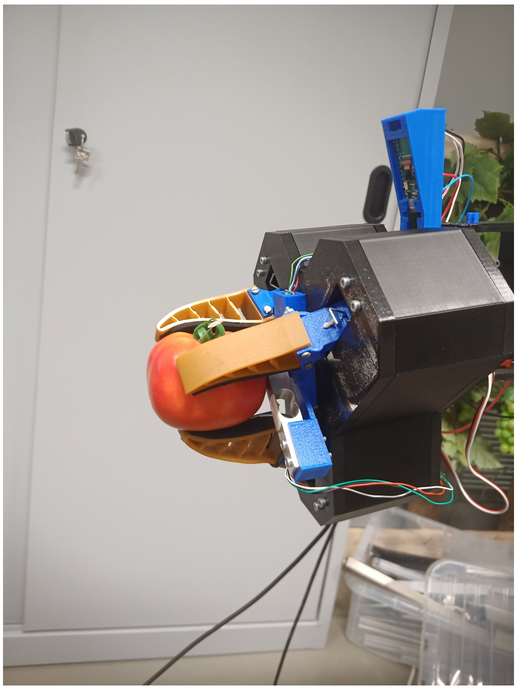
Figure 7.7: Lower Finger Angle - Gripper Closing

Considering these results, the new values for the sensors thresholds were given with the following logic: For the flexible sensor, the threshold considered was the closest integer higher than the average value calculated, being 16°. Considering the flexible sensor's readings are inconsistent, while the force sensor was more accurate, the latter had two different thresholds. In case the flexible sensor had readings bigger than its threshold, the force sensor's value to be considered should be higher than 30 g, which allows it to consider the lower value read on the initial tests (34 g). Otherwise, if the flexible sensor did not achieve the required value, the force sensor's threshold is increased to 60 g, forcing the gripper to close slightly more to guarantee the fruit is not dropped.

The new threshold worked as expected, allowing the end-effector to consider it successfully grasped the tomato when the PWM pulse duration was near 1.8 µs. This position is enough to grasp the tomato securely without applying excessive force on the grasped fruit (Figure 7.8).



(a) Front View



(b) Side View

Figure 7.8: Tomato Grasping Front View (Left) And Side View (Right)

7.4 Communication Efficiency

Testing the whole system, there were two types of communication protocols implemented at the same time, USB for the communication between the end-effector control and the manipulator's Raspberry Pi and UART between the end-effector control and the perception module.

These communications were crucial to assure the correct functioning of the whole system, which requires being constantly exchanging information between the different parts to proceed on the tasks. Comparing this implementation to the tests and validation of the two protocols separately, there are no significant differences associated. Since the two types of communication use different GPIOs and interfaces, they do not interfere with each other, being able to operate separately on the code. Considering this, the communication on the final version of the project was successful, guaranteeing the transmission of orders and information between the different parts.

7.5 Overall Cost

The end-effector developed was capable of complete the tasks it was designed to fulfil and meet the requirements defined in the beginning of the project, including being cost-effective. Overall, the cost of the components used on the end-effector, displayed in Table 7.2, was around 336 €.

Table 7.2: List of Components

Type of Component	**Quantity**	**Unit Cost (€)**	**Total (€)**
Soft-Fingers (Approximate value based on similar fingers)	3	37.86	113.58
Arducam Pico 4ML	1	25	25
Raspberry Pico	1	4.59	4.59
Force Sensing Resistor	1	6.10	6.10
Flexible Sensor	1	11.9	11.9
Load Cell	1	7.56	7.56
HX711 Module	1	9.95	9.95
ABRS-8034DH Servomotor	2	60.38	120.76
VL53L1X Sensor	1	17.91	17.91
3D prints (Considering 20€/kg)	-	-	13.64
Various Hardware (Screws, etc)	-	-	5
TOTAL			335.99

Chapter 8

Conclusion

Despite the end-effector meeting the requirements established on the beginning of the project, it is not possible to state the end-effector is fully completed and able to be used as a definitive version on the field.

In order to be completely validated, there is the need to conduct more tests. The trials already done on the laboratory were conducted with plastic tomato models, images of tomato and images of peduncles. The main reason for this type of tests was to avoid damaging and waste real tomatoes, which would oppose to the purpose of developing a tool to complete harvesting tasks efficiently. Only on the end of this project, having all the parts working safely and accordingly, it would make sense to perform tests on real fruit for final adjustments on the sensor's thresholds.

Conducting field tests would also be crucial to understand the efficiency of the perception module detecting different types of tomato on natural sunlight. The tests performed on the laboratory were done under artificial light, leading to the need of adjusting the camera's exposure to be able to detect the tomato without getting white spots on the frame caused by the light, which compromised the efficiency of the classifier.

Overall, the end-effector developed can be considered an efficient first step of a tool to be implemented on the final version of the manipulator, needing only the missing cutting mechanism and final calibrations to be considered complete and ready to use on the field to harvest tomato.

8.1 Outcomes

On the end of the project, the end-effector designed and built was able to fulfil the initial requirements defined, both structural and software. The different parts work properly, both individual and together, as previously demonstrated.

Considering this, it was possible to achieve:

- The article "End-Effectors For Harvesting Manipulators - State Of The Art Review" [20] published at ICARSC 2022 (International Conference on Autonomous Robot Systems And Competitions 2022)

- An open source design for a cost-effective end-effector called FruitGrip
- Software design for FruitGrip
- Deep learning model for tomato and peduncle detection
- A prototype of FruitGrip at TRL5 (Technology Readiness Level 5)

8.2 Challenges

The challenges associated with this work were mainly caused by the current lack of stock available on electronic components, which resulted in several delays on some material needed to complete the end-effector. This resulted in some software for the project being developed without the sensors available, needing a debug phase more extensive when all the parts finally arrived, which compromised the bottom-up approach chosen.

The other aspect that raised concern was due to the Raspberry Pico and the RP2040 microcontroller being a novelty on the market. Since its release in the beginning of the year 2021, there were made several projects with the microcontroller, yet this number is not comparable with the availability of projects and documentation found for boards containing, for example, the ATmega328p microprocessor. The lack of information available made the work of developing code more complex, specially since there was the need to use components which required a complex understanding of the capability of the RP2040 to handle some algorithms, mainly related to the code used for the perception module.

Despite these difficulties associated, the project was capable to advance and surpass those problems.

8.3 Future Work

There are still some improvements and calibrations to be done on the end-effector developed, mainly related to the adjustment of sensor values while harvesting real fruit. Also, considering the manipulator used consists of a prototype which is still being developed, there might be needed to adjust the number of requests sent from the end-effector to the manipulator to adjust its height. Currently, the end-effector is doing five requests per adjustment, however if the manipulator movement is altered on a next phase of the prototype this value might need to be adjusted.

To modify this end-effector to be able to complete harvesting tasks successfully in the field, there is the need to include a cutting mechanism to cut the peduncle, separating the fruit from the plant. This mechanism was not possible to develop during the project, where it was given focus to the perception module and the approach to grasp the tomato. However, the perception module already searches for the peduncle, commanding the manipulator to adjust its height to guarantee an optimal harvest position. This step is crucial for the integration of a cutting mechanism, since it will not be needed additional logic implemented on the end-effector. The cutting mechanism is supposed to actuate only when the fruit is grasped. Considering after the fruit is grasped, the end-effector waits for the manipulator to give the order to open the gripper, the end-effector can be unaware of the existence of the cutting tool, being the latter commanded by the manipulator's computer.

In the future might be relevant to consider change the perception module hardware. As previously stated on the obtained results, the perception module works properly as expected; however, the inference cycle time is around 3550 ms. On the field, this inference time might be disadvantageous and can compromise the efficiency of the robot when compared to human labour. Having the same classifier and logic implemented on a more powerful microcontroller might be an important modification to improve the performance of the end-effector by reducing the time needed for each classification cycle. There is also the possibility of a more powerful microcontroller being capable of image detection, which the Arducam Pico 4ML was not capable of doing, which lead to the use of image classification on the algorithm. As previously analysed in the bibliography review, the size of the tool can compromise its efficiency on harvesting. This characteristic might also be a factor to consider change in the future, mainly after testing the end-effector on the field. During the project, the design of the different components was made considering the size of the tool; yet, there were parts reinforced to assure the fruit's weight on the end-effector would not be an obstacle to the success of the latter on the completion of the tasks. The structural reinforcements on the components resulted on the increase of the end-effector dimensions, which might be reduced after testing on the field, analysing the resulting stress on the structure and redesigning some parts.

References

[1] B. Jia, A. Zhu, S.X. Yang, and G.S. Mittal. Integrated gripper and cutter in a mobile robotic system for harvesting greenhouse products. pages 1778–1783, 2009. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-77951483544&doi=10.1109%2fROBIO.2009.5420430&partnerID=40&md5=cff1461efc6025d4f63fd6116233270d, doi:10.1109/ROBIO.2009.5420430.

[2] N. Kondo, K. Yata, M. Iida, T. Shiigi, M. Monta, M. Kurita, and H. Omori. Development of an end-effector for a tomato cluster harvesting robot. *Engineering in Agriculture, Environment and Food*, 3(1):20–24, 2010. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-79954608604&doi=10.1016%2fS1881-8366%2810%2980007-2&partnerID=40&md5=eda55f9e40153a70fb7f55b5e4ebf5f9, doi:10.1016/S1881-8366(10)80007-2.

[3] Hiroaki Yaguchi, Kotaro Nagahama, Takaomi Hasegawa, and Masayuki Inaba. Development of an autonomous tomato harvesting robot with rotational plucking gripper. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 652–657, 2016. doi:10.1109/IROS.2016.7759122.

[4] C. Lehnert, A. English, C. McCool, A.W. Tow, and T. Perez. Autonomous sweet pepper harvesting for protected cropping systems. *IEEE Robotics and Automation Letters*, 2(2):872–879, 2017. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85058585408&doi=10.1109%2fLRA.2017.2655622&partnerID=40&md5=c2826a918ad2b17e12b6600fb904fc73, doi:10.1109/LRA.2017.2655622.

[5] L. Mu, G. Cui, Y. Liu, Y. Cui, L. Fu, and Y. Gejima. Design and simulation of an integrated end-effector for picking kiwifruit by robot. *Information Processing in Agriculture*, 7(1):58–71, 2020. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85067246778&doi=10.1016%2fj.inpa.2019.05.004&partnerID=40&md5=8d4f2f648a30f35fc5f9f0d6a09e092a, doi:10.1016/j.inpa.2019.05.004.

[6] Jiacheng Rong, Pengbo Wang, Qian Yang, and Feng Huang. A field-tested harvesting robot for oyster mushroom in greenhouse. *Agronomy*, 11(6), 2021, DOI: 10.3390/agronomy11061210. doi:10.3390/agronomy11061210.

[7] Charles Godfray, John Beddington, Ian Crute, Lawrence Haddad, David Lawrence, James Muir, Jules Pretty, Sherman Robinson, Sandy Thomas, and Camilla Toulmin. Food security: The challenge of feeding 9 billion people. *Science (New York, N.Y.)*, 327:812–8, 02 2010. doi:10.1126/science.1185383.

[8] L. Depenbusch, C.R. Farnworth, P. Schreinemachers, T. Myint, M.M. Islam, N.D. Kundu, T. Myint, A.M. San, R. Jahan, and R.M. Nair. When machines

take the beans: Ex-ante socioeconomic impact evaluation of mechanized harvesting of mungbean in bangladesh and myanmar. *Agronomy*, 11(5), 2021. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85106489067&doi=10.3390%2fagronomy11050925&partnerID=40&md5=452faebff46b2c93036c821e32e918eb, doi:10.3390/agronomy11050925.

[9] Redmond Shamshiri, Cornelia Weltzien, Ibrahim Hameed, Ian Yule, Tony Grift, Siva Balasundram, Lenka Pitonakova, Desa Ahmad, and Girish Chowdhary. Research and development in agricultural robotics: A perspective of digital farming. *International Journal of Agricultural and Biological Engineering*, 11:1–14, 07 2018. doi:10.25165/j.ijabe.20181104.4278.

[10] J. Becedas, I. Payo, and V. Feliu. Two-flexible-fingers gripper force feedback control system for its application as end effector on a 6-dof manipulator. *IEEE Transactions on Robotics*, 27(3):599–615, 2011. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-79958771568&doi=10.1109%2fTRO.2011.2132850&partnerID=40&md5=419c4b2713338b05b60ab140eb790154, doi:10.1109/TRO.2011.2132850.

[11] V. Tinoco, M.F. Silva, F.N. Santos, L.F. Rocha, S. Magalhaes, and L.C. Santos. A review of pruning and harvesting manipulators. pages 155–160, 2021. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85107147336&doi=10.1109%2fICARSC52212.2021.9429806&partnerID=40&md5=325ef2cdc06b30b6a515657a20f6804e, doi:10.1109/ICARSC52212.2021.9429806.

[12] V. Tinoco, M. F. Silva, F. N. Santos, A. Valente, L. F. Rocha, S. A. Magalhães, and L. C. Santos. An overview of pruning and harvesting manipulators. *Industrial Robot*, 2021. doi:10.1108/IR-07-2021-0139.

[13] Serdar Kucuk and Z. Bingul. The inverse kinematics solutions of industrial robot manipulators. *Proceedings of the IEEE International Conference on Mechatronics, 2004. ICM '04.*, pages 274 – 279, 07 2004. doi:10.1109/ICMECH.2004.1364451.

[14] Mitsubishi electric corporation. URL: https://emea.mitsubishielectric.com/fa/products/rbt.

[15] Xiangyu Chen, Krishneel Chaudhary, Yoshimaru Tanaka, Kotaro Nagahama, Hiroaki Yaguchi, Kei Okada, and Masayuki Inaba. Reasoning-based vision recognition for agricultural humanoid robot toward tomato harvesting. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6487–6494, 2015. doi:10.1109/IROS.2015.7354304.

[16] Whitney Crooks, Gabrielle Vukasin, Maeve O'Sullivan, William Messner, and Chris Rogers. Fin ray effect inspired soft robotic gripper: From the robosoft grand challenge toward optimization. *Frontiers in Robotics and AI*, 3, 11 2016. doi:10.3389/frobt.2016.00070.

[17] Pedro Machado, Andreas Oikonomou, João Filipe Ferreira, and T. Martin Mcginnity. Hsmd: An object motion detection algorithm using a hybrid spiking neural network architecture. *IEEE Access*, 9:125258–125268, 2021. doi:10.1109/ACCESS.2021.3111005.

[18] Eric Peña and Mary Grace Legaspi. Uart: A hardware communication protocol understanding universal asynchronous receiver/transmitter. *Analog Dialogue [online]*, 4(54):1–5, 2020. URL: https://www.analog.com/media/en/analog-dialogue/volume-54/number-4/uart-a-hardware-communication-protocol.pdf.

[19] T Balint, J Živčák, R Hudák, T Tóth, M Kohan, and S Lancoš. Destructive and non-destructive testing of samples from PLA and PETG materials. *IOP Conference Series: Materials Science and Engineering*, 1199(1):012045, nov 2021. URL: https://doi.org/10.1088/1757-899x/1199/1/012045, doi:10.1088/1757-899x/1199/1/012045.

[20] Francisco Oliveira, Vítor Tinoco, Sandro Magalhães, Filipe N. Santos, and Manuel F. Silva. End-effectors for harvesting manipulators - state of the art review. In *2022 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 98–103, 2022. doi:10.1109/ICARSC55462.2022.9784809.