UNIVERSIDADE
BEIRA INTERIOR

Aplicação do robô colaborativo UR3e no ensino na área da robótica

Luís Carlos Baptista Nunes

Dissertação para obtenção do Grau de Mestre em
Engenharia Eletromecânica
(2º ciclo de estudos)

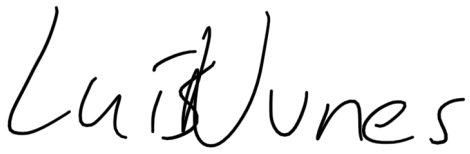
Orientador: Prof. Doutor Pedro Miguel de Figueiredo Dinis Oliveira Gaspar
Co-orientador: Mestre Martim Lima de Aguiar

junho de 2023

Declaração de Integridade

Eu, Luís Carlos Baptista Nunes, que abaixo assino, estudante com o número de inscrição M11809 do 2º Ciclo em Engenharia Eletromecânica da Faculdade de Engenharia, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o **Código de Integridade da Universidade da Beira Interior**.

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.



Universidade da Beira Interior, Covilhã 6 /06 /2023

Agradecimentos

Em todas as fases da vida, como em qualquer projeto, é necessária uma ajuda de todos. Esta ajuda pode ser de várias formas e feitios, um ombro amigo, uma palavra amiga, uma saída de descompressão… resumindo, o mais importante é a presença das pessoas que realmente nos ajudam a sermos melhores. Um obrigado, é a única de agradecer pelo apoio que todas essas pessoas me deram, não me deixando desistir, para terminar o meu percurso académico.

Ao meu orientador, o Professor Doutor Pedro Miguel de Figueiredo Dinis Oliveira Gaspar, pela ajuda, orientação e esclarecimento e motivação. Por nunca me ter deixado desistir do tema escolhido e me mostrando a importância e relevância que o mesmo têm. Um muito obrigado!

Ao meu co-orientador, Mestre Martim Lima Aguiar, pela paciência que teve comigo e a força para não desistir, pela orientação e as palavras de incentivo. Agradecer também pelos momentos de riso que tínhamos, que ajudava na descompressão em momentos mais stressantes.

Aos meus colegas de laboratório pela ajuda e pelo apoio a não me deixarem desistir, o Rodrigo Antunes e o Nuno Pereira, sem eles não teria sido possível chegar ao fim.

À minha família, em especial atenção aos meus pais e à tia Luísa, pela presença, carinho e aquelas palavras de força nos momentos certos para não deixar ir abaixo. Pela paciência nos dias menos bons em que a paciência já faltava. Um muito obrigado do fundo do coração!

À minha namorada, Maria Fonseca, pelo ombro amigo e por estar presente sempre que precisei. Pela paciência que teve comigo, mesmo quando eu não estava no meu melhor, pelo carinho e pela força incansável que me deu para conseguir completar esta etapa. Um muito, muito obrigada!

Aos meus amigos, as amizades criadas, aos momentos vividos, as bebedeiras e noitadas. Aqueles que estiveram sempre presentes e prontos a ajudar, serviram de apoio e motivação. Um especial obrigado aos amigos da casa 32, que se tornaram na família da Covilhã. Agradecer aos que já não são amigos, mas sim família, por todas as palavras de apoio e pelo percurso que fizemos juntos ao longo destes anos magníficos.

Um especial agradecimento, a cidade da Covilhã, uma cidade que me acolheu de braços abertos, que me proporcionou anos de pura magia, uma cidade lindíssima e acolhedora. À Universidade da Beira Interior, por tudo, pelo apoio e por terem sido a minha segunda casa. Aos funcionários incríveis do bar das engenharias, pelas histórias e pelos momentos incríveis que nos proporcionaram.

Muito obrigado a todos aqueles que tornaram estes anos possíveis e principalmente a esta conquista. Sem vocês não seria possível.

Obrigado!

Resumo

Nos dias de hoje vive-se uma constante evolução ao nível da robótica. Podemos encontrar robôs em qualquer lado, numa farmácia, num hospital, numa fábrica, até mesmo em casa, e a lista continua. O que para as gerações antigas era novidade, ver um robô, para as gerações mais recentes e uma situação comum.

A falta de mão-de-obra qualificada é um problema. Nomeadamente, a falta de pessoal qualificado e com experiência é mais visível na indústria, o que compromete a capacidade de produção nacional.

Este défice de mão-de-obra qualificado tem vindo a sofrer uma ligeira melhoria. Com o avanço tecnológico e com a maior facilidade de as pessoas poderem frequentar um curso superior ou até mesmo cursos de especialização profissional. No entanto tem de existir um esforço por parte das entidades formadoras em atualizar os cursos com base nas necessidades das entidades empregadoras.

Nesta dissertação é apresentada a criação de uma infraestrutura de apoio às atividades laboratoriais em cursos e disciplinas de Robótica Industrial. A infraestrutura de forma matricial é composta por um conjunto de elementos, ponto de partida, ponto de chegada, obstáculos, seções de poste e seções de pista com diferentes comprimentos, que podem estar em diferentes posições e organização, sendo variável a colocação dos três primeiros tipos de elementos, por parte do utilizador. Os diferentes elementos foram desenvolvidos por fabrico aditivo em PLA e possuem diferentes cores para serem identificáveis. Esta infraestrutura permite a construção, de forma autónoma, de uma pista por parte de um robô colaborativo, em particular um manipulador UR3e da Universal Robots, que utiliza visão computacional para identificar os pontos de partida e de chegada, assim como dos obstáculos. Por via da aplicação do algoritmo do caminho mais curto, o robô constrói um trajeto para um berlinde a percorrer, desde o ponto de partida até ao ponto de chegada, evitando os obstáculos, com base na quantidade de movimento adquirida pela ação da gravidade decorrente das diferentes alturas a que são colocadas as seções de pista. A programação dos processos (deteção da posição dos elementos pontos de partida e de chegada, e obstáculos por visão computacional, determinação do caminho mais curto, estratégia de construção da pista, e posicionamento e movimento do robô) foi realizado na linguagem de programação Python.

Os protocolos para realização do trabalho laboratorial desenvolvido estão divididos em 3 níveis de conhecimento e consequentemente de dificuldade. O primeiro nível de dificuldade, o mais simples, consiste em mover peças (peça da posição inicial, peça da posição final e as peças obstáculo), de uma posição onde estão armazenadas para as posições definidas no protocolo. No protocolo que diz respeito ao segundo nível de dificuldade, as peças vão ser colocadas no tapete rolante por uma determinada ordem, e o robô, à medida que as vai buscar ao tapete rolante, coloca-as consoante a ordem definida no protocolo. Por fim, no protocolo nível três, o de maior grau de dificuldade, as peças vão ser colocadas no tapete rolante de uma forma aleatória. O robô terá que detetar a cor da peça fazendo uso da câmara que tem acoplada na sua flange e consoante a cor da peça, terá de a colocar de acordo com a disposição que se encontra definida no protocolo.

Este trabalho tem como objetivo ajudar a colmatar a necessidade de mão-de-obra qualificada na indústria afeta a trabalhos com manipuladores robóticos. A realização dos trabalhos nas componentes laboratoriais de cursos e disciplinas de robótica industrial vão auxiliar a aquisição de competências dos alunos ao nível da programação de manipuladores robóticos.

Palavras-chave

Robótica; Robótica Industrial; Robótica na educação; Robô Colaborativo; UR3e; Protocolos laboratoriais; Infraestrutura de apoio ao ensino.

Abstract

In today's world, robotics is constantly evolving. We can find robotic elements anywhere, in a pharmacy, in a hospital, in a factory, even at home, and the list goes on. What was strange for the older generations, seeing a robot, for the more recent generations is commonplace.

Increasingly the shortage of skilled labor is a problem. The shortage of qualified and experienced personnel is most visible in the labor market, compromising the nation's production capacity, which means that the education and training of qualified labor is of great importance.

This shortage of skilled labor has been improving slightly. With technological advances and the greater ease with which people can attend higher education or even professional specialization courses. However, there must be an effort on the part of training entities to update courses based on the needs of employers.

This dissertation presents the creation of an infrastructure to support laboratory activities in courses and disciplines of Industrial Robotics. The matrix-shaped infrastructure is composed by a set of elements, starting point, ending point, obstacles, column sections and track sections with different lengths, which can be in different positions and configurations, being variable the placement of the first three types of elements, by the user. The different elements have been developed by additive manufacturing in PLA and have different colors to be identifiable. This infrastructure allows the autonomous construction of a track by a collaborative robot, in particular the Universal Robots UR3e, by image recognition of the starting and ending points, as well as obstacles. By applying the shortest path algorithm, the robot constructs a path for a marble to travel, from the starting point to the finish point, avoiding the obstacles, based on the amount of movement acquired by the action of gravity arising from the different heights at which the track sections are placed. The programming of the processes (detection of the position of the elements starting and ending points, and obstacles by computer vision, determination of the shortest path, track construction strategy, and positioning and movement of the robot) was performed in the Python programming language.

The protocols of the laboratory work are divided into 3 levels of knowledge and consequently of difficulty. The first level of difficulty, the simplest, consists in moving pieces (the initial position piece, the final position piece and the obstacle pieces), from a

position where they are stored to the positions defined in the protocol. In the protocol concerning the second difficulty level, the pieces will be placed on the conveyor belt in a certain order, and the robot, as it gets them from the conveyor belt, places them according to the order defined in the protocol. Finally, in protocol level three, the most difficult, the pieces will be placed on the conveyor belt in a random order. The robot will have to detect the color of the piece using the camera attached to its flange and, depending on the color of the piece, it will have to place it according to the arrangement defined in the protocol.

This work aims to help fill the need for qualified labor in the industry related to work with robotic manipulators. The work carried out in laboratory during the industrial robotics courses and disciplines will help students to acquire skills in terms of programming robotic manipulators.

Keywords

Robotics; Industrial robotics; Robotics in education; Collaborative robotics; UR3e; Lab protocols; Infrastructure to support teaching.

Índice

Lista de Figuras

Lista de Tabelas

Nomenclatura

Acrónimos:

ABB *Asea Brown Boveri;*

AM *Additive Manufacturing* (Fabrico Aditivo)*;*

ASTD *American Society for Training and Development;*

CAD *Computer-Aided Design* (Desenho Assistido por Computador);

CNC *Computer Numeric Control* (Controlador Numérico Computorizado);

EUA Estados Unidos da América;

FDM *Fused deposition modeling* (Modelação por deposição de material fundido);

IFR *International Federation of Robotics;*

MDF *Medium Density Fiberboard* (Placa de Média Densidade);

PETG *Polyethylene terephthalate glycol* (Politereftalato de etileno glicol);

PLA *Polylactic acid* (Ácido poliláctico);

PLCs *Programmable Logic Controller* (Programador Lógico Programável);

RGB *Red Green Blue*;

STEM *Science, Technology, Engineering and Mathematics* (Ciência, Tecnologia, Engenharia e Matemática);

UBI Universidade da Beira Interior;

UR *Universal Robots;*

1. Introdução

Neste capítulo, será dada uma abordagem/contextualização do tema da dissertação desenvolvida neste documento. Serão ainda abordadas as motivações que levaram a realização deste trabalho, bem como os objetivos do mesmo. Por último, vai ser apresentado a metodologia e a estrutura da dissertação.

1.1. Enquadramento

Nos últimos anos têm-se observado um avanço significativo a nível da robotização e automação, tanto nas tarefas domésticas, como na medicina, indústria, forças armadas, etc. O principal objetivo desta evolução é que a haja uma adaptação "*das funções dos robôs com as necessidades humanas* (p. 1)." (Do *et al.*, 2022).

Nos dias de hoje, a robótica faz parte do cotidiano, onde quer que se vá pode-se estar na presença de um sistema robotizado, no entanto, é preciso mais abertura a cerca deste tema do ponto de vista da sua mais-valia para a sociedade e economia. Existe o paradigma de que quanto mais a robótica tomar conta da sociedade, mais aumentará o desemprego. O que para uma sociedade pouco informada pode ser uma justificação para que estes avanços não aconteçam tão rápido como têm acontecido. Por um lado, pode dizer-se que de facto existem postos de trabalho que se vão perder, pela substituição de algumas tarefas, mais monótonas, repetitivas e alienadoras para o ser humano. No entanto, essas tarefas vão ser substituídas por tecnologia, sistemas robotizados e automatizados. Por outro lado, esta evolução nos sistemas de produção na indústria terá um "impacto positivo dos robôs industriais no emprego" (p. 1), segundo a *International Federation of Robotics (IFR)* prevê que devido à robótica vão ser criados cerca de 4 a 6 milhões de postos de trabalho, no entanto se for contabilizado o número de postos de trabalho que indiretamente também vão ser criados, o número sobe para os 10 milhões (Do *et al.*, 2022; Sergeyev *et al.*, 2017).

O rápido avanço da robotização e automação de sistemas, como referido acima em várias áreas, tem criado uma grande procura de profissionais de robótica. Esta procura tem de ser colmatada pela formação e qualificação de novos profissionais, onde a educação e a formação em sistemas robotizados irão ter um papel fundamental e fulcral para a elevada procura/necessidade de profissionais na área de robótica. Com estes avanços, a necessidade de mão-de-obra é grande, no entanto, existe uma necessidade de pessoas qualificadas e com a formação certa para ir de encontro com os avanços tecnológicos. A *American Society for Training and Development* (ASTD) reportou que as organizações nos Estados Unidos da América (EUA) gastam, cerca de 164,2 mil milhões de dólares em formação de funcionários (Do *et al.*, 2022a; Sergeyev *et al.*, 2017).

Para que as organizações empresariais não precisem de gastar quantias avultadas em formações específicas para a maior parte dos seus colaboradores, pode colmatar-se esta evolução tecnológica e a grande procura da indústria por mão-de-obra qualificada/especializada. A evolução da robótica na educação vai desempenhar um papel muito importante na preparação de novos talentos para o mercado (Do *et al.*, 2022a).

Nos dias de hoje, ainda existe uma grande discrepância entre a educação e as necessidades profissionais, mais propriamente na indústria. No entanto, se se conseguir uma maior aproximação entre as duas, será uma mais-valia, tanto para quem tenta tirar partido da educação, no caso educação superior universitária, tanto para quem emprega diplomados com esta graduação, pois vai-se sentir mais à vontade em contratar um colaborador sem ter a preocupação em gastos económicos e tempo em formações para que este esteja apto para as funções desejadas. Sabe-se que a formação específica vai ter sempre de existir, devido à constante evolução de conceitos, mas se se reduzir esta formação, por pouco que seja, já e uma mais-valia tanto para o colaborador como para o empregador (Santos & Júnior, 2020; Atman *et al.*, 2022; Do *et al.*, 2022).

Quando se pensa na robótica educacional é necessário ter em conta uma abordagem multidisciplinar, do ponto de vista, que quando se aborda o tema "robótica" tem de se ter as facetas associadas à mecânica, elétrica/eletrónica, a programação, a automação, etc. O *STEM (Science, Technology, Engineering and Mathematics)* é um método educativo que visa a interdisciplinaridade, entre outros aspetos, abordando as várias áreas de estudo, matemática, ciência e tecnologia, oferecendo novos benefícios na educação a todos os níveis. Esta abordagem faz a interligação de todas as áreas de estudo necessárias para uma melhor abordagem da robótica industrial, acrescentando que a robótica estimula a capacidade de resolução de problemas, capacidade de comunicação, capacidade de trabalho em equipa, independência, imaginação e criatividade (Sergeyev, 2010; Santos & Júnior, 2020; Gao *et al.*, 2020; Wang *et al.*, 2020; Navy *et al.*, 2021).

1.2. Objetivos

A robótica está muito presente no dia-a-dia da sociedade, no entanto, no ponto de vista da robótica a nível industrial verifica-se que houve um grande investimento por parte das entidades patronais. Estima-se que existe um crescimento de 18-25% na implementação de robôs industriais. Para este crescimento, a procura de novos colaboradores com formações e capacidades adequadas, vai ser cada vez maior. Para este elevado crescimento da implementação de robôs na indústria também vai ser necessário um grande número de colaboradores dotados com capacidades em robótica, com grande capacidade para uma evolução na tecnologia e com capacidade para dar pequenas formações aos utilizadores (Sergeyev *et al.*, 2017).

O principal objetivo desta dissertação reside no desenvolvimento de um protocolo com vários graus de dificuldade de um trabalho laboratorial envolvendo programação e teste em sistema real de aplicações de robótica industrial, com o intuito de providenciar um maior entendimento, perceção e uma menor discrepância entre a robótica educacional e a robótica industrial, em alunos do ensino superior.

Para que o objetivo principal seja cumprido, foram definidos os seguintes objetivos:

- Realizar uma revisão bibliográfica sobre as temáticas em estudo – Robótica, Robótica Industrial, Robótica educativa e STEM;
- Criação de uma infraestrutura e, consequentemente, um sistema de apoio ao ensino, adaptável a várias unidades curriculares;
- Desenvolvimento de uma atividade laboratorial e criação de dois protocolos laboratoriais com vários graus de dificuldade, baseado em funções que replicam tarefas que se observam em ambiente industrial;
- Análise dos resultados obtidos.

1.3. Metodologia

A metodologia utilizada para o desenvolvimento desta dissertação teve por base o objetivo principal e os objetivos específicos referidos no ponto anterior. Logo numa primeira fase procedeu-se uma pesquisa descritiva, na qual se estudou e analisou-se o conceito de robótica industrial e robótica educativa, com forma de conhecer algumas diferenças entre ambas.

A revisão bibliográfica serviu como base para todo o trabalho desenvolvido, bem como para orientar o caso prático, a realização de novos protocolos. Com esta abordagem conseguiu-se uma melhor análise e compreensão da literatura existente sobre o tema. Para a obtenção destes dados foram utilizadas bases de dados científicos, principalmente, Science *Direct* e *Web of Science*. Desta forma as fontes foram utilizadas para ter um conhecimento da informação referenciada por diversos autores.

Numa primeira fase do trabalho, foram definidos os objetivos, gerais e específicos, a desenvolver, numa segunda fase realizou-se a revisão bibliográfica dos temas em estudo, robótica industrial e robótica na educação, para ser obtida uma base teórica e prática, com análise de alguns trabalhos e testes já realizados, para sustentar a componente prática do trabalho. Em seguida, foi desenvolvida e realizada a atividade laboratorial proposta. Depois da realização dos protocolos, fez-se uma verificação dos protocolos, com o intuito de conseguir perceber se existiriam alguns

erros ou alguns passos pouco específicos. Por consequência, pretende-se que estes protocolos sejam testados nas unidades curriculares das áreas.

1.4. Estrutura da dissertação

A presente dissertação é composta por cinco capítulos para apresentar de uma forma simples e sucinta o trabalho desenvolvido.

O capítulo 1, correspondente à introdução ao trabalho, é feita uma contextualização do tema em análise e são apresentados os seus principais objetivos. É apresentada a metodologia e a estrutura deste trabalho.

No capítulo 2 é abordado o enquadramento teórico e a revisão bibliográfica do conceito, robótica industrial, bem como o de robótica na educação, bem como a importância de cada um individualmente e principalmente a importância e dependência entre estes dois temas.

No capítulo 3 é apresentado o caso prático, o desenvolvimento do material de apoio, o desenvolvimento do programa para a resolução dos protocolos, bem como alguns testes no funcionamento do programa.

No capítulo 4 apresenta-se a análise dos resultados obtidos, a infraestrutura criada para as atividades de robótica, bem como a análise do programa desenvolvido.

Por fim, no capítulo 5 são apresentadas as conclusões ao trabalho desenvolvido e as propostas de trabalhos futuros, baseados em fatores que ao longo do desenvolvimento desta dissertação se identificaram como sendo pertinentes para serem desenvolvidos de uma forma mais aprofundada.

2. Estado da Arte

No presente capítulo será apresentada a contextualização teórica dos temas abordados nesta dissertação. No subcapítulo referente a Robótica, será abordada a sua evolução ao longo da história, o conceito de robótica industrial e o conceito de robótica colaborativa. Robótica na Educação, o subcapítulo seguinte, abordará a importância da robótica na educação, bem como um método de desenvolvimento dos alunos e uma comparação entre a robótica industrial e robótica na educação.

2.1. Robótica

O que é a robótica? Quando se pensa na resposta a esta questão está presente um leque muito abrangente de respostas, no entanto, uma resposta simples a esta questão, a robótica é: a ciência e o estudo de robôs. A palavra robótica surgiu na série de "I, Robot", onde o escritor Isaac Asimov a refere pela primeira vez (Arulkirubakaran *et al.*, 2022)

O que é um robô? Outra pergunta na qual se pode obter uma imensa quantidade de respostas onde qualquer resposta pode estar certa. A palavra robô vem da palavra checa "*robota*" que significa "trabalho forçado" nos tempos atuais. Um robô pode ser entendido como um utensilio de auxílio ao ser humano na realização de um determinado número de tarefas (Azevedo *et al.*, 2010; Gasparetto, 2016; Arulkirubakaran *et al.*, 2022)

Na robótica foram criadas três leis para salvaguardar o bem-estar de todos, e para se conseguir uma melhor interligação entre os robôs e o sere humano, para que estes convivam em plena harmonia (Barua *et al.*, 2020; Zamalloa *et al.*, 2017):

1º. Um robô não pode prejudicar fisicamente e mentalmente uma pessoa nem que a mesma sofra qualquer dano pessoal por uma atividade sua;

2º. Um robô deve obedecer a todas as ordens que um ser humano lhe dá, no entanto não pode ir contra a 1º Lei, ou seja não pode prejudicar fisicamente uma pessoa;

3º. Um robô deve proteger a sua própria integridade, no entanto não pode ir contra a 1º e 2º Lei.

2.1.1. Evolução do Conceito

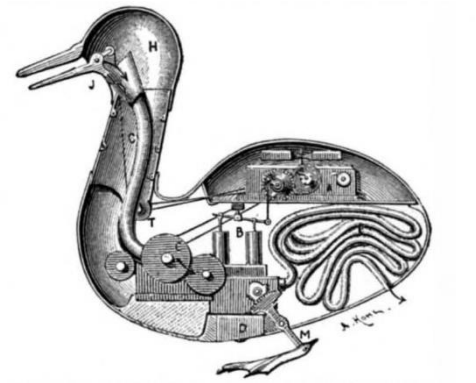
Historicamente existem algumas razões para acreditar que cerca de 285-222 a.C. os primeiros "aparelhos robóticos", foram construídos pelo Engenheiro e Matemático grego Ctesibius, que consistia num relógio a água. Também grego, Heron de Alexandria, Engenheiro e Geómetra,

construiu vários objetos aos quais se pode chamar de "sistemas robotizados", como a primeira máquina de bebidas, que colocando uma moeda, saia dela um jato de água, ou ainda um objeto que andava para a frente e para trás, movido por engrenagens usando a energia cinétia de grãos de trigo. Como curiosidade, criou o primeiro motor a vapor comprovado na história (Azevedo *et al.*, 2010).

Historicamente, a robótica fica também marcada por Leonardo Da Vinci, "Cientista, Matemático, Engenheiro, Inventor Anatomista, Pintor, Escultor, Arquiteto, Botânico, Poeta e Músico", Da Vinci ainda nos dias de hoje é reconhecido devido à sua mente brilhante. Contudo devido à época em que se encontrava, grande parte das suas invenções não passaram do papel, no entanto o "Robô da Vinci", Fig.1, é umas das suas invenções, e era usado por altos membros da realeza com o intuito de os entreter (Azevedo *et al.*, 2010; Gasparetto, 2016).

Figura 1. Modelo do robô de Leonardo (Moran, 2006).

Em 1730, Jacques de Vaucanson, inventor e artista francês, teve também um papel importante nesta área. Ficou conhecido pelas suas invenções: o pato mecânico, o qual conseguia imitar um pato real como se este estivesse a alimentar-se, Fig. 2, e também o tocador de flauta, Fig. 3. A medida que a tecnologia se ia desenvolvendo as pessoas cada vez mais iam ficando preocupadas com os seus postos de trabalho, pois estavam a sentir que os seus postos de trabalho começavam a ficar em risco devido ao aparecimento de equipamentos robóticos (Azevedo *et al.*, 2010; Gasparetto, 2016; Ge *et al.*, 2020).



INTERIOR OF VAUCANSON'S AUTOMATIC DUCK.
A, clockwork; *B*, pump; *C*, mill for grinding grain; *F*, intestinal tube; *J*, bill; *H*, head; *M*, feet.

Figura 2. Pato Digestivo (Enrica Merlo, 2018).

Figura 3. Tocador de Flauta (Enrica Merlo, 2018).

A evolução da robótica está relacionada com outras áreas e com a evolução da própria sociedade. A robótica é uma área multidisciplinar que abrange diversas áreas, como: a matemática, ciência, tecnologia, física e a ciência computacional. A robótica apresenta também grande dependência na criação e evolução dos componentes necessários às construções, tais como: transístores, computadores digitais, controladores numéricos, circuitos integrados, entre outros. Com a evolução destas tecnologias as características dos robôs também foram evoluindo, tanto a nível mecânico como a nível da programação, mais autónomos, mais capazes a nível mecânico, capazes de suportar cargas mais elevadas, mais ágeis, etc. (Ge *et al.*, 2020; Zamalloa *et al.*, 2017; Atman *et al.*, 2022; David & Nourbakkhsh, 2016).

Com a análise das suas características mecânicas e as suas propriedades a nível de autonomia, eficácia de movimentos (precisão), pode fazer-se uma classificação a nível de gerações. Existindo 6 gerações sendo elas um pouco diferentes entre si, consoante a evolução da tecnologia, estando descritas de seguida (Zamalloa *et al.*, 2017).

2.1.1.1. Geração 0: até 1950

Esta geração fica marcada pelo humanóide de Leonardo Da Vinci, Fig 1, desenvolvido em 1495 e por algumas invenções de Jacques de Vaucanson, em 1730, como o tocador de flauta e o pato digestivo, no entanto uma geração muito rudimentar comparando com a atualidade.

Com a primeira revolução industrial é que se começa a pensar em automação com o objetivo de facilitar processos e de os tornar mais rápidos, para isto começou a usar-se sistemas pneumáticos[1] e sistemas hidráulicos[2]. A utilização destes equipamentos dá-se pois estes, não necessitavam de ser comandados por nenhum sistema eletrónico, mas sim acionados pelos trabalhadores. O grande objetivo desta Era foi a automatização destes processos, e a automatização surge mais tarde com o aparecimento do primeiro computador, *Colossus* (Azevedo *et al.*, n.d.; Zamalloa *et al.*, 2017; Azevedo *et al.*, n.d.; Gasparetto, 2016).

2.1.1.2. 1ª Geração: Primeiros Manipuladores (1950-1967)

As duas características que marcam esta geração são: a falta de informação durante o processo, ou seja, os robôs não obtêm nenhuma informação do seu ambiente em redor. Este faziam "movimentos cegos", os movimentos eram contínuos mesmo que tenham um objeto à sua frente o que pode acarretar alguns problemas, como por exemplo danificar o próprio robô ou até mesmo danificar objetos no ambiente em redor. A outra grande característica é a sua simplicidade de controlo, estes robôs simplesmente moviam-se de um ponto para o outro, e assim sucessivamente, o que facilitava a sua programação na medida em que não havia nenhum fator externo para alterar previamente o movimento (Ge *et al.*, 2020; Zamalloa *et al.*, 2017).

O primeiro robô de que há conhecimento foi o UNIMATE, Fig. 5, criado por George Devol, este foi considerado como o primeiro robô industrial. Em 1960, este robô foi instalado numa empresa (GM) onde conseguiram ver logo resultados muito promissores com a sua utilização, o que

[1] Um sistema pneumático consiste num conjunto de componentes interligados que usam ar comprimido para realizar uma determinada tarefa (movimentar, rodar, fixar, aparafusar, etc.) (Jiang *et al.*, 2021).

[2] Um sistema hidráulico consiste na execução de uma determinada tarefa através da utilização de um fluido sob pressão. Este líquido sob pressão contra uma superfície cria uma força ou potência (Jiang *et al.*, 2021).

motivou muitas empresas tanto a implementar como a fazer uma pesquisa em robótica (Gasparetto & Scalera, 2019; Zamalloa *et al.*, 2017).

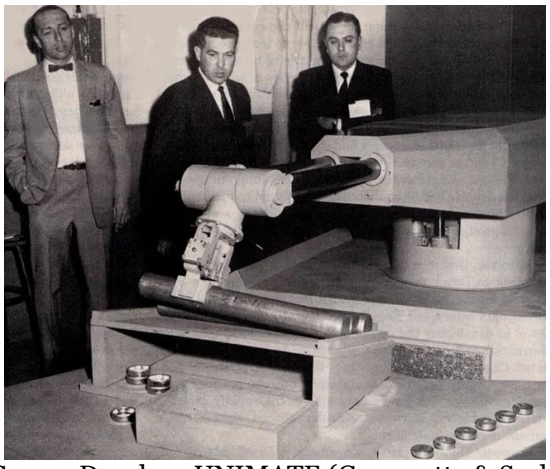


Figura 4. George Devol e o UNIMATE (Gasparetto & Scalera, 2019).

2.1.1.3. 2ª Geração: Robôs dotados de sensores (1968-1977)

Nesta geração, como o subtítulo caracteriza-se por se ter robôs com sensores, onde estes se tornam mais autónomos e eficazes. As características mais marcantes nesta geração, é: o melhor conhecimento a nível do espaço que rodeia os robôs, maior sensorização a nível da força, binário[3] e visão e a capacidade de aprenderem por demonstração (Giralt *et al.*, 1990; Zamalloa *et al.*, 2017).

A grande vantagem da integração de sensores, em comparação com os robôs de 1ª geração é o facto de estes serem capazes de interagir com o ambiente em seu redor, como por exemplo: (1) evitar obstáculos (2) parar quando tocavam em algum objeto na sua trajetória, (3) capacidade de ultrapassar desafios e (4) eram muito mais completos que os anteriores. Estes avanços só foram possíveis devido à criação dos PLCs (*Programmable Logic Controllers* - Controlador Logico Programável), um equipamento baseado numa unidade de processamento capaz de controlar vários tipos de equipamentos industriais e sistemas automatizados (Garcia Elena *et al.*, 2007; Gasparetto, 2016).

A KUKA, um dos maiores fabricantes de robôs no mundo, construiu o primeiro robô industrial, com seis eixos acionados com motores elétricos, designado de FAMULUS, Fig. 5, em 1973, um ano mais tarde o robô designado de T3, Fig. 6, foi introduzido no mercado por Cincinnati

[3] Binário consiste numa grandeza vetorial responsável pela produção da rotação de um corpo (Park *et al.*, 2019).

Milacron. Este foi o primeiro a ser comercializado, que era controlado por um microcomputador (Mark & Paul, 1982; Gasparetto & Scalera, 2019).

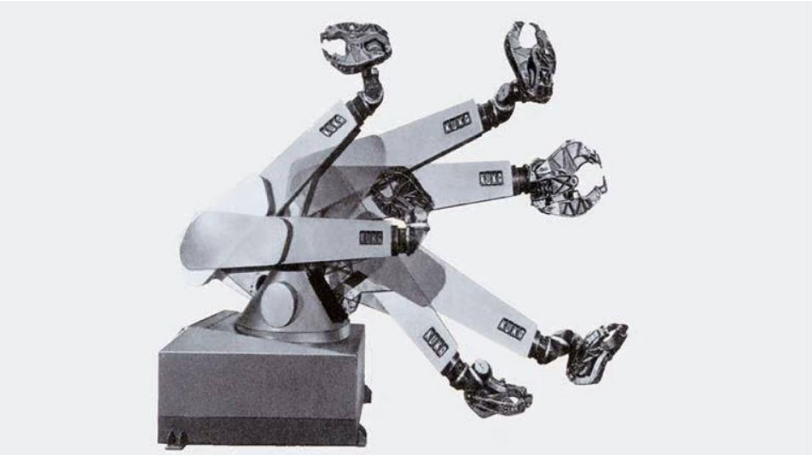


Figura 5. KUKA- FAMULUS (Shepherd & Buchstab, 2014).

Figura 6. Robô T3 (Gasparetto & Scalera, 2019)(.

2.1.1.4. 3ª Geração: Robôs Industriais (1978-1999)

Esta geração fica marcada pela "Era dos Robôs" (Maeda, 2005), em 1980, onde verbas avultadas foram investidas pelas empresas no seu desenvolvimento e avanço tecnológico, com o objetivo de tornarem as suas linhas de produção mais rápidas e eficientes. As atividades onde se investiu mais foram: Pintura; Solda; Posicionamento (*Pick and Place*) e Montagem. O investimento global na compra de robôs obteve um aumento de cerca de 80 % face aos anos anteriores (Maeda, 2005; Zamalloa *et al.*, 2017; Gasparetto & Scalera, 2019).

Nesta época, existiram alguns avanços na tecnologia que proporcionaram desenvolvimentos significativos na área da robótica, alguns deles não só para a robótica, como o acesso geral à *internet*, como por exemplo, que proporcionou a troca de informação em tempo real. A linguagem

da programação da robótica também foi algo que evoluiu. Por exemplo, a FANUC usou a linguagem chamada de *Karel,* Fig. 7, em 1988 e a ABB (*Asea Brown Boveri)* criou a *Rapid*, o que possibilitou que os robôs fossem reprogramáveis, ou seja após programados uma vez podiam voltar a ser programados com outro programa para uma tarefa diferente (Jack, 1994; Lapham, 1999; Castells, 2002).

```
PROGRAM MOVER
VAR
  original: POSITION
  destination: POSITION
  count: INTEGER
CONST
  gripper = 1
  num_of_parts = 10
BEGIN
  $SPEED = 200.0
  $MOTYPE = LINEAR
  OPEN HAND gripper
  FOR count = 1 TO num_of_parts DO
    MOVE TO original
    CLOSE HAND gripper
    MOVE TO destination
    OPEN HAND gripper
  ENDFOR
END mover
```

Figura 7. *Karel* linguagem (Lapham, 1999).

2.1.1.5. 4ª Geração: Robôs Inteligentes (2000-2017)

Na quarta geração, as características que se evidenciaram mais foram (Zamalloa *et al.*, 2017; Gasparetto & Scalera, 2019):

- Utilização de capacidades informáticas avançadas, que possibilitaram que os robôs suportassem mais informação e a processassem de uma forma mais rápida e com menos erros;
- Os robôs eram capazes de realizar tarefas não só com os programas já carregados, mas também estes mesmos programas lógicos eram capazes de se adaptarem ou até mesmo aprender com o ambiente de trabalho;
- A inteligência artificial começou a ter as suas primeiras aparições de uma forma parcial e experimental;
- Sensorização muito mais avançada que eram enviadas e analisadas dependendo da estratégia do controlador;
- Introdução de robôs colaborativos, mas ainda numa fase inicial e experimental.

O robô Roomba, Fig. 8, foi o primeiro robô de limpeza doméstico, um aspirador, que podia aspirar a casa autonomamente sem ninguém o estar a controlar ou a fornecer informação. A ABB introduziu o primeiro robô colaborativo, YuMi, Fig. 9, que é conhecido pela sua segurança em trabalhar com um operador ao seu lado sem este correr algum perigo, melhorando o processo de produção e a postura do operador (ABB, 2015; Barua *et al.*, 2020; Jones *et al.*, 2005).

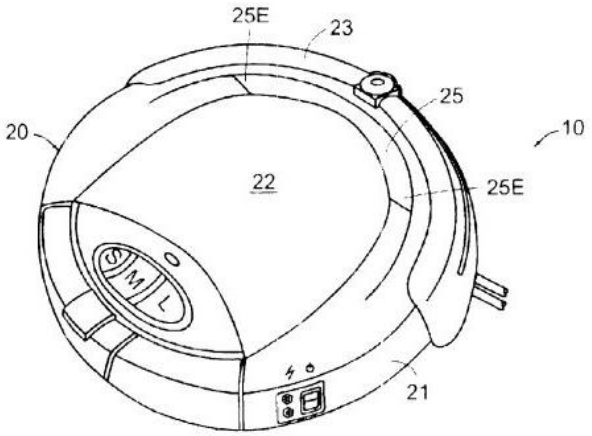


Figura 8. Robô Roomba (Jones *et al.*, 2005).

Figura 9. Robô Yumi (ABB, 2015).

2.1.1.6. 5ª Geração: Robôs Colaborativos e Pessoais (2017-)

Esta é a atual geração, no entanto será uma geração que ainda vai estar em desenvolvimento. Apresenta uma vantagem muito importante em relação as anteriores, nesta geração os robôs podem trabalhar em perfeita sincronia com os colaboradores e no mesmo espaço sem ser preciso uma célula de trabalho, onde esta separava o ambiente de trabalho do robô, do restante espaço. Ainda nesta geração existem robôs que podem ser configurados, ou seja, pode ter-se o mesmo robô com configurações diferentes o que os torna muito mais versáteis, pois o mesmo robô pode fazer vairas atividades com uma simples alteração na sua configuração (Barua *et al.*, 2020; Do *et al.*, 2022).

Um robô colaborativo que surgiu nesta época, que apresenta grandes vantagem em relação aos robôs de gerações anteriores, por exemplo, é o UR3, Fig. 10, um robô criado pela UR (Universal

Robots), um robô onde se pode ter várias configurações a nível de garras e acessórios que o torna-o muito versátil.

Figura 10. Robô Colaborativo UR3e (Universal Robots, 2021).

2.1.2. Robótica Industrial

O avanço tecnológico, tem vindo a ser muito associado à necessidade de aumentar a rapidez de processos. Também cada vez mais as indústrias têm de tentar arranjar soluções para conseguir reduzir os tempos de produção ao máximo. Outra necessidade das empresas é o carregamento de material mais pesado bem como necessidade de movimentação de objetos de uma forma mais rápida e sem tanto esforço dos operadores (Hägele *et al.*, 2016; Grau *et al.*, 2017).

Surge então assim a Robótica Industrial, estando descrita de uma forma breve no capítulo seguinte. Será analisado o avanço das tecnologias, e as grandes diferenças entre as mesmas.

O primeiro robô industrial apareceu em 1959, desenvolvido por George Devol e Joseph Engelberger, este pesava cerca de 2 toneladas e era controlado por um programa num tambor magnético, com atuadores hidráulicos (convertem a anergia de trabalho em energia mecânica) que eram programados por coordenadas, designado de Unimation, Fig. 4. Apresar de ser inventado em 1959, só dois anos depois é que este é instalado numa linha de produção nos EUA. Posteriormente em 1962, surge o primeiro robô cilíndrico, Versatran, Fig. 11, que também era atuado por componentes hidráulicos, muito semelhante ao Unimation (Arulkirubakaran *et al.*, 2022).

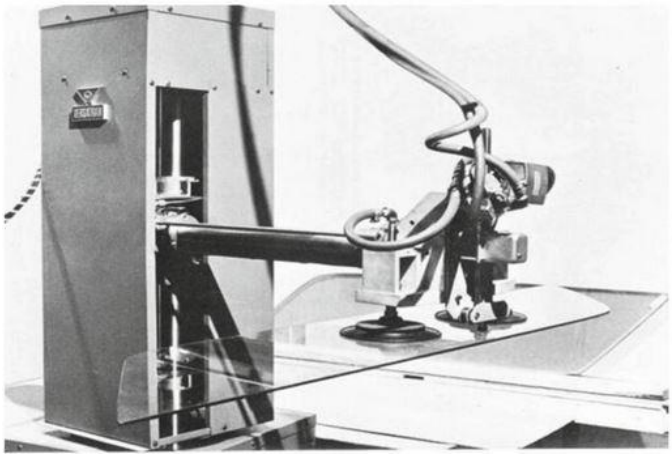


Figura 11. Robô Versatran (Arulkirubakaran *et al.*, 2022).

Posteriormente, em 1969, Victor Scheinman, um estudante de Engenharia Mecânica, contruiu o primeiro robô elétrico, Stanford Arm, Fig. 12. Este robô era controlado por um microprocessador, composto por 5 juntas rotativas e uma junta prismática, ou seja, tinha 6 graus de liberdade[4]. O robô era dotado de sensores para medir a posição e a velocidade. Com este tipo de construção e interface, observou se uma revolução na área de robótica industrial, isto porque consequentemente foi possível observar que muitas das ideias advinham da ideia desenvolvida por Victor Scheinman, o Standford Arm (Gasparetto & Scalera, 2019).

Em 1973, com a crise do petróleo muitas empresas tiveram de criar soluções para terem uma eficiência superior nos seus processos de fabrico, o que proporcionou um aumento de cerca de 30 % na compra de robôs industriais. Ainda no mesmo ano a KUKA desenvolveu o primeiro robô com 6 eixos eletromecânicos, denominado de Famulus, Fig. 5. No ano seguinte, o robô T3 *"The Tomorrow Tool"* (Em Portuges: "A Ferramenta do Amanhã"), foi desenvolvido e este tinha incorporado o primeiro microcomputador, que veio revolucionar o mercado, onde este teve grande procura pelas empresas do setor automóvel. Nesse mesmo ano, a empresa ASEA, mais conhecida por ABB, começou a desenvolver a gama de robôs IRB que foi uma grande aposta da empresa. Esta gama de robôs foi muito conhecida e muito bem-sucedida em toda a robótica industrial. O IRB-6, Fig. 12, foi considerando por muitos um robô "lendário", muito devido às suas capacidades. Este era capaz de desempenhar inúmeras tarefas, tais como: manuseamento de material, transporte, embalamento, soldadura, lixamento de material, e polimento. (Karlsson, 1991; Gasparetto & Scalera, 2019; Arulkirubakaran *et al.*, 2022)

[4] Capacidade de deslocamentos e rotações que um sistema consegue fazer, 1 grau de liberdade- um deslocamento ou rotação, 2 graus de liberdade- 2 deslocamentos ou rotações, por norma, cada grau de liberdade esta relacionado com o número de articulações que um robô têm (Meng, 2022).

Figura 12. Robô Stanford Arm (Gasparetto & Scalera, 2019).

Figura 13. Robô IRB-6 (Arulkirubakaran *et al.*, 2022).

Posteriormente, em 1978, Hiroshi Makino criou o robô de montagem SCARA, Fig. 14, um robô rápido, com boa precisão e principalmente a um custo baixo. Este veio revolucionar a produção de componentes elétricos e produtos consumíveis, devido à sua rapidez e exatidão de movimentos. Dada a sua construção simples e barata, este sofreu algumas alterações/ melhorias ao longo do tempo, o exemplo de uma adaptação é a versão da Fig. 15. Estes robôs já eram dotados de uma tecnologia mais avançada onde era possível programar *online* usando uma caixa de aprendizagem ou *offline* usando um PLC ou um computador (Hägele *et al.*, 2016; Arulkirubakaran *et al.*, 2022).

Figura 14. Robô SCARA (Arulkirubakaran *et al.*, 2022).

Figura 15. SCARA adaptação (Arulkirubakaran *et al.*, 2022).

Contudo, estes robôs também eram capazes de identificar objetos na sua área de trabalho usando a visão ou sistemas de perceção, estas vantagens eram usadas em determinadas atividades onde a visão era uma das principais funções, como por exemplo a verificação de defeitos nas peças, inspeção de peças ou até mesmo soldadura (Hägele *et al.*, 2016; Arulkirubakaran *et al.*, 2022).

Mais tarde, em 1992, uma empresa Suíça, a Demaurex, apresentou a primeira célula/estação de trabalho com 6 robôs Delta, Fig. 16. Pouco tempo depois, a ABB contruiu um robô da mesma categoria, robôs paralelos, denominado "Flex-Picker", Figura 17. Estes eram conhecidos pela sua rapidez nos movimentos e precisão, no entanto apresentam a desvantagem de não serem tão flexíveis no que toca à área de trabalho, mais pequenas em comparação com os robôs acima mencionados (Hägele *et al.*, 2016).

Figura 16. Estação robôs Delta (Arulkirubakaran *et al.*, 2022).

Figura 17. Robô Flex-Picher (Arulkirubakaran *et al.*, 2022).

2.1.3. Robótica Colaborativa

Quando se pensa em robótica colaborativa surge a ideia de qualquer tipo de colaboração entre o robô e o seu operador, o ser humano. A essência é essa, o ser humano junta as suas mais-valias em colaboração com as do robô, ou seja, é uma união entre o robô e o ser humano, Fig. 18. Este tipo de procedimento tem muitas vantagens do ponto de vista de trabalhos possíveis, pois como se sabe, um robô pode trabalhar horas a fio sem ter perdas no seu rendimento e na sua precisão de trabalho. Já quando se falado ser humano não e bem assim, o ser humano chega a um certo ponto e vai estar cansado, o seu nível de concentração já não é o mesmo e isso vai causar falta de precisão e perda de eficiência, o que para uma empresa significa perda de lucros ou despesas adicionais (Hägele *et al.*, 2016).

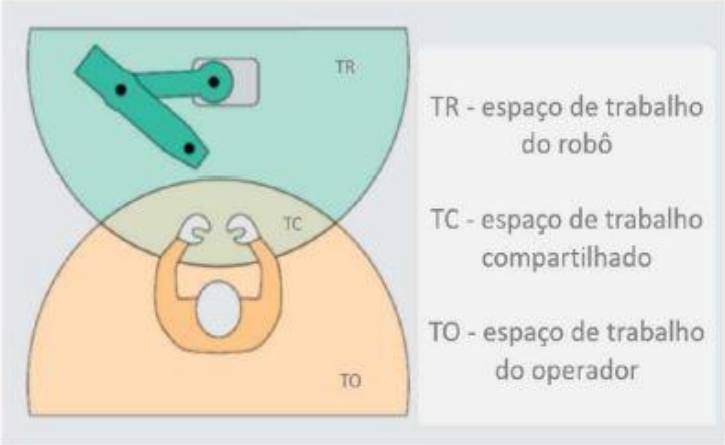


Figura 18. Espaço de trabalho colaborativo (Soares & Lucato, 2021).

Em termos ergonómicos, a capacidade de carregar/suportar peso, por parte do ser humano, está limitada, quer por questões de dimensionamento, quer por capacidade de suportar cargas. Esta limitação, para além de impedir o desempenho de determinadas tarefas, também pode provocar fadiga ou até mesmo lesões musculoesqueléticas, difíceis de reverter. Quando se aplica o exemplo de suportar cargas, aplicado a um robô, este também apresenta limitações. Sendo estas limitações apenas de quantidade de carga suportada, basta adaptar a capacidade de carga do robô ao peso/carga que se pretende que este seja capaz de suportar. E ao contrário, das limitações físicas do ser humano, uma das vantagens de utilização de robôs reside na possibilidade de este estar em constante carga sem interrupções (Hägele *et al.*, 2016; Kirgis *et al.*, 2016).

Quando se observa o lado do ser humano, pode concluir-se que este vai ter vantagem onde um robô não as vai ter, como a adaptação rápida a uma nova tarefa, um determinado objeto que esta no espaço de trabalho, situações inesperadas que possam acontecer ou até a capacidade de resolver problemas que possam surgir (Hägele *et al.*, 2016; Marques *et al.*, 2020).

No que toca à robótica colaborativa tem de se ter sempre em atenção à segurança, tanto do operador do robô como do meio que o rodeia. No entanto, já não é necessário considerar as células de trabalho, pois neste caso os robôs já não estão confinados a um espaço fechado. Estes vão se encontrar num ambiente aberto onde o operador pode interagir com o robô e com os objetos da determinada tarefa. Para que esta junção aconteça de uma forma segura, estes robôs estão dotados de inúmeros sensores para detetar e saber a distância de obstáculos ou até mesmo pessoas para evitar contactos indesejáveis evitando acidentes. Caso os sensores não consigam detetar algum obstáculo, este está preparado para quando existir algum contacto que não esteja previamente considerado no *software* de programação, parar. No entanto, este só para se sofrer uma força, sobre o braço robótico, maior do que a definida nas medidas de segurança.

No entanto, existem vantagem e desvantagens na robótica colaborativa, e também alguns desafios, visto que ainda é um tema recente e com uma margem muito grande de investigação e

progressão. Como mostra a Tabela 1, a robótica colaborativa apresenta algumas vantagens principalmente na indústria, como: viabilidade económica, facilidade de programação, vantagens a nível de esforço, segurança, a postura que traz para o ser humano, colaboração humano-robô e a alta versatilidade. Contudo, nem tudo pode ser perfeito, e como desvantagens tem-se: algumas incertezas na eficácia operacional, baixas velocidades e por vezes baixas capacidades de carga devido a uma baixa robustez. No que toca aos desafios, coloca-se logo a hipótese de algo que se pode melhorar para ultrapassar os desafios, logo esta área ainda necessita de ser desenvolvida (Marques *et al.*, 2020; Soares & Lucato, 2021).

Tabela 1. Vantagens, desvantagens e desafios da robótica colaborativa (adaptado de Marques *et al.*, 2020; Soares & Lucato, 2021).

Vantagens	Viabilidade económica
	Colaboração humano-robô
	Alta versatilidade
	Segurança
	Facilidade de programação
	Aumento da ergonomia
Desvantagens	Baixa Velocidade
	Pouca capacidade de carga
	Incertezas sobre a eficácia operacional
Desafios	Divisão de tarefas
	Adaptabilidade

Na atualidade, existem algumas marcas comerciais que estão na vanguarda da robótica colaborativa com os seus baraços robóticos colaborativos, sendo algumas delas das principais responsáveis pelo desenvolvimento e aprimoramento desta tecnologia tão importante para a indústria e outras áreas que também usam a robótica para melhorarem a sua eficiência. A ABB e a Universal Robots são algumas das empresas que mais se tem destacado no desenvolvimento e na produção de braços robóticos colaborativos, como por exemplo, a ABB com o GoFa CRB 15000, Fig. 19, e a Universal Robots com o UR3e, Fig. 20.

Figura 19. ABB GoFa CBR 15000 (ABB, 2021).

Figura 20. UR3e (Universal Robots, 2021).

2.2. Robótica na Educação

"*Muitos trabalhos existentes serão automatizados nos próximos 20 anos*" (Sergeyev *et al.*, 2015). O mundo está a sofrer um avanço tecnológico muito grande, onde uma grande parte dos empregos que irão estar disponíveis vão ser na área da robótica, o que do ponto de vista educacional é muito importante que se comece desde cedo a introduzir o conceito de robótica na educação (Sergeyev *et al.*, 2015; Sergeyev *et al.*, 2017).

A educação é o bem essencial para qualquer sociedade dita desenvolvida, pois a educação é base da sociedade. Da educação advém o desenvolvimento de um país. Com a educação vem o desenvolvimento social de uma nação. Muito resumidamente a educação é a base de tudo. Uma criança não nasce ensinada, tem de aprender a falar, andar, etc., para isto existe a educação que pode ter várias formas e feitios, no entanto neste trabalho interessa-nos uma educação em ambiente escolar. A educação em ambiente escolar pode depender de muitos fatores: (1) meio em que se encontra, (2) área de estudo em causa, (3) grau de dificuldade necessário, entre outros.

Estes mesmos fatores vão ser a chave para o desenrolar da abordagem da educação a adotar, ou seja, por exemplo, o tipo de aula que vai ser lecionada (aulas mais práticas ou aulas mais teóricas), uma aula de campo ou uma aula na própria infraestrutura da instituição em causa, uma abordagem onde os alunos têm de fazer o seu próprio estudo para resolverem um problema ou o professor ensina como resolver o mesmo problema, existem muitas possibilidades (Shmatko & Volkova, 2020; Do *et al.*, 2022).

A robótica na educação é uma área muito importante devido à sua multidisciplinaridade, ou seja, a necessidade de várias áreas do conhecimento como a física, biologia, geografia, matemática, ciência, eletrónica e mecânica. Estas serão proporcionadoras de outras competências como escrever, ler, pesquisar, colaboração, pensamento crítico, tomada de decisões, resolução de problema, a comunicação, o *design* e o pensamento computacional (Eguchi, 2017).

Quando se investiga sobre a multidisciplinaridade da robótica surge sempre a sigla "STEM" pois está relacionada com a multidisciplinaridade na educação. STEM (*Science* - Ciência, *Technology* - Tecnologia, *Engineering* - Engenharia e *Mathematics* - Matemática) é um termo muito usado na área da educação. Alguns autores defendem que a interdisciplinaridade será muito importante na educação nos tempos de hoje. Como o mundo está com uma evolução tão rápida a nível tecnológico, se se adotar uma educação STEM, os alunos vão estar mais bem preparados para a constante evolução tecnológica (Gao *et al.*, 2020; Li *et al.*, 2020; Navy *et al.*, 2021).

A robótica na educação pode proporcionar uma experiência muito mais iterativa dos alunos, uma abordagem mais motivadora e apelativa, para estes, onde vão seros mesmos a fazer alguma coisa, irão "meter as mãos na massa". Esta abordagem torna as aulas mais dinâmicas, do que as aulas tradicionais onde os alunos limitam-se a ouvir os professores (Eguchi, 2017).

Por outro lado, quando se fala em robótica, e no seu ensino, têm de ser os alunos a desenvolver todos os passos para o projeto final, para uma melhor apreensão de conhecimentos. A montagem e a remontagem dos materiais para o *design* do robô, enquanto passam por um processo de resolução de problemas durante o processo de *design* do mesmo e durante a sua programação (Eguchi, 2017).

Quando se pensa na abordagem da atividade da robótica na educação, esta tem de ser realizada pelos alunos em pequenos grupos, não podem ser grupos muito grandes, mas nunca atividades de um só elemento, pois o objetivo das atividades são proporcionar uma colaboração entre as disciplinas, mas também uma colaboração entre os próprios alunos. A colaboração entre os alunos vai fornecer competências de trabalho em equipa, como expressar as suas ideias, tomada de decisões em grupo, criticar construtivamente e uma das mais importantes as competências na comunicação. Trabalhar em grupo também vai proporcionar uma aprendizagem em áreas diferentes Nem todas as pessoas são iguais o que umas vão ser melhores em algumas áreas do que outras. Esta diferença que cada um vai ter nas suas competências vai permitir que as pessoas

aprendam umas com as outras e que se ajudem umas às outras nas áreas que são melhores (Pires, 2003; Alareje, *et al.*, 2017; Eguchi, 2017; Berry *et al.*, 2020; Gao *et al.*, 2020; Rahman, 2021).

2.2.1. Robótica Industrial vs Robótica Industrial na Educação

Uma das grandes diferenças entre a robótica industrial e a robótica industrial na educação será na área da nova tecnologia, onde esta chega sempre primeiro ao meio industrial do que ao meio educacional, muito devido à busca de eficiência do meio industrial face à concorrência existente no mesmo (Do *et al.*, 2022).

A robótica industrial é uma área de evolução constante, todos os anos surgem materiais novos, técnicas novas, novos *softwares*, ou seja, uma área que se encontra em constante desenvolvimento. Para que esta constante evolução seja acompanhada pelas instituições de ensino, básico, secundário ou superior, neste caso em particular, tem de existir uma ponte entre a indústria e a entidade educacional para reduzir as diferenças entre ambas. No entanto, no caso do ensino superior, as universidades têm de tentar colmatar estas discrepâncias para que os seus alunos saiam bem preparados para o mundo atual de trabalho, e que saiam com competências mais próximas das necessitadas no mundo industrial. Segundo Do *et al.* (2022) e Tijdens *et al.* (2018), a falta de profissionais técnicos e especializados muitas das vezes é por causa da discrepância de competências entre o sistema de educação e a real necessidade das indústrias ou então, devido à diferença entre a aprendizagem e a formação, que não correspondem às reais necessidades industriais (Pires, 2003; Malinetskii & Sirenko, 2017; Tijdens *et al.*, 2018; Do *et al.*, 2022).

Segundo Sergeyev *et al.* (2017), atualmente as indústrias precisam de contratar cerca de 2.5 milhões de trabalhadores STEM, ou seja, profissionais que adquiriram formação STEM, onde a maior parte proveem de programas de engenharias baseados no STEM, que criam as competências necessárias para estes estudantes entrarem no mercado de trabalho, preparados para os desafios reais. Para que estes estudantes saiam das universidades preparados, as entidades formadoras têm de adotar novas formas de ensino, a necessidade de um ensino mais focado nas necessidades da indústria, que passa por os alunos terem mais atividades práticas de resolução de problemas, para que as intuições de ensino ajudem a colmatar a falta de mão-de-obra qualificada e com as competências corretas e necessárias (Alaraje *et al.*, 2010; Sergeyev *et al.*, 2017; Sergeyev *et al.*, 2018).

Apesar de existir algum esforço por parte das entidades educadoras para diminuírem a discrepância entre as competências lecionadas para as competências necessárias para o mundo industrial, existem alguns entraves. Alguns deles são monetários (falta de apoios), falta de vontade de alguns membros das entidades educadoras de fazer diferente, e muitas das vezes também e complicado para uma empresa parar a sua produção ou atividade laboral para que

alguns alunos possam experienciar um pouco do que é o mundo industrial e se familiarizar com alguns problemas que possam surgir (Almaleh *et al.*, 2019).

No entanto, outra possibilidade é a mudança do método de ensino nas instituições de ensino, pois este só depende das pessoas da entidade educacional para haver mudança. Esta mudança/melhoria vai ter de ser contínua, pois, a área da robótica está em contante desenvolvimento, logo a área educacional também vai ter de acompanhar o desenvolvimento da robótica, que com a sua evolução vai verificar-se uma expansão de áreas de aplicação, o que vai criar a necessidade de novas competências multidisciplinares (Mcmurtrey *et al.*, 2008; Almaleh *et al.*, 2019).

2.2.1.1. Atividades Industriais Práticas

A robótica industrial cada vez está mais enraizada na indústria a nível mundial, o que consequentemente, conduz à necessidade de mão-de-obra qualificada e especializada, sendo para isto, necessário aproximar a educação para tal realidade. Com a introdução de atividades práticas nas atividades educativas está a proporcionar-se cenários para simular as atividades que são realizadas na indústria. Com estas simulações proporciona-se aos alunos uma melhoria nas suas competências para o mercado industrial como a sua educação STEM. De seguida encontra-se caracterizadas as atividades práticas mais próximas da realidade industrial, possíveis de desenvolver em contexto de sala de aula (Sergeyev *et al.*, 2017).

2.2.1.2. Posicionamento (*Pick and Place*)

Uma atividade de *Pick and Place*, Fig. 21, traduzido "pegar e colocar", consiste simplesmente em mover uma peça de um lugar para o outro. Como por exemplo, tirar material de uma palete e colocar num determinado local, ou até mesmo colocar material de uma linha de produção para uma palete para passar ao próximo passo, ou mudar o material de uma estação de produção para a estação seguinte, rodar o material para este ser trabalhado dos dois lados, etc. Estes tipos de atividades são muito versáteis pois são a base de quase todas as outras atividades de uma linha de produção(Sergeyev *et al.*, 2017).

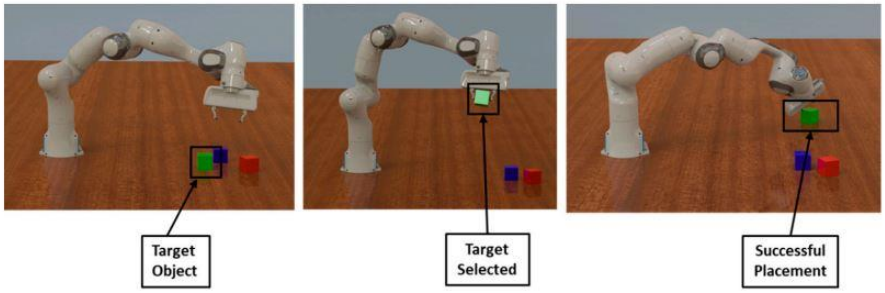


Figura 21. *Pick and Place* aplicação adaptado de (Lobbezoo *et al.*, 2021).

2.2.1.3. Lixamento de uma Superfície

Este tipo de atividade não é tão propício a existir numa instituição de ensino, pois vai requer um ambiente particular devido ao lixo e pó que o lixamento vai criar. No entanto, existem muitas empresas que optam por colocar robôs a fazer este tipo de trabalhos, tanto no lixamento de peças como no seu polimento, Fig. 22, dois procedimentos muito parecidos.

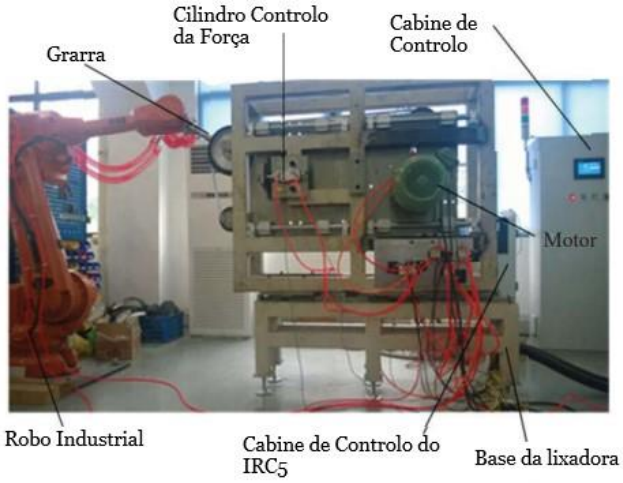


Figura 22. Sistema de lixamento robotico adaptado de (Verl *et al.*, 2018).

2.2.1.4. Soldadura

Esta atividade, soldadura, Fig. 23, é muito frequente nas empresas, devido à facilidade que um robô tem em concretizar esta tarefa, para além da precisão com que a faz. No entanto, esta atividade para ser realizada em ambiente educacional será muito difícil, pois é necessário muito equipamento e uma sala onde seja seguro fazer este tipo de aplicações.

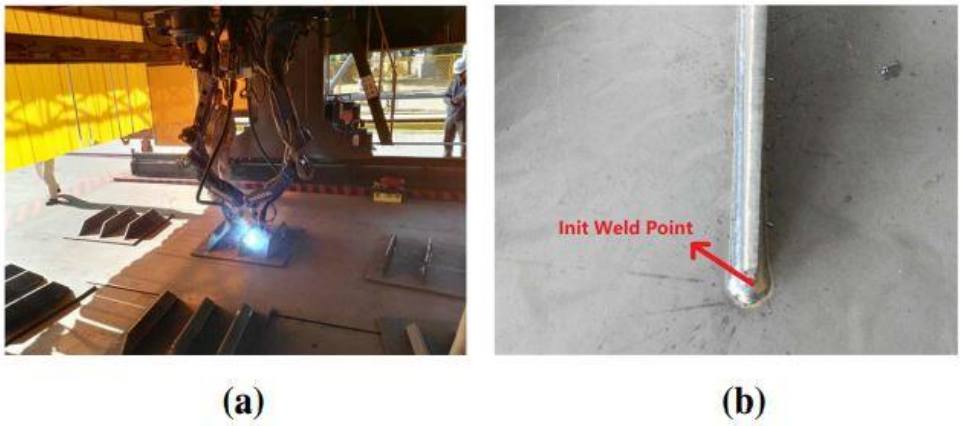


Figura 23. a) Processo de Soldadura. B) Resultado final da soldadura. Adaptado de (Yang *et al.*, 2020).

2.2.1.5. Deteção / Separação

No que toca à deteção/separação tem de se ter em conta que pode ser qualquer tipo de deteção. Para esta deteção vai necessitar-se do auxílio de uma câmara, que irá detetar as peças, detetar erros de forma, erros de cor, se esta está pintada ou não, se têm o acabamento pretendido ou até se tem o tratamento final ou não. Pode haver outro tipo de deteção de defeitos, através do peso das peças, no entanto será necessário mais material auxiliar (Herakovic, 2010; Pérez *et al.*, 2016).

A separação, Fig. 24, não é mais nem menos como o nome indica, a separação de dois ou mais tipos de peças que estejam juntas, de diferentes formas ou até mesmo de diferentes cores, sendo que para esta atividade vai ser necessário o auxílio da ajuda de visão (Pérez *et al.*, 2016).

Figura 24. Separação de material (Pérez *et al.*, 2016).

3. Caso Prático

Neste capítulo, é concebido um problema, que será resolvido por meio de ferramentas da robótica. Para tal, será construída toda a infraestrutura de apoio à robótica educacional, com o objetivo de simular operações de manipulação, sensorização e controlo que se podem encontrar na indústria. apresentado todo o processo de criação dos protocolos laboratoriais da robótica aplicada à educação, bem como a sua simulação, com o objetivo de mostrar que estes podem ser uma melhor abordagem entre a robótica na educação e a robótica industrial.

Será apresentado, todo o processo de conceção das peças necessárias para a realização dos protocolos, bem como o método utilizado para a sua execução, e por fim a simulação dos mesmos.

3.1. Desenvolvimento de Protocolos

Para o desenvolvimento do material de apoio foi usado um *software* de desenho assistido por computador - CAD (*Computer-Aided Design*) 3D, o *Solidworks*, para o desenvolvimento modular dos mesmos. Para a produção dos modelos CAD foi usado uma técnica chamada de manufatura aditiva, mais propriamente a impressão 3D, por FFF, e maquinação numa CNC.

O uso do *software* de desenho CAD 3D deve-se à familiarização com o mesmo, e pela utilização do *software* na resolução de outros projetos, o que facilitou o processo de desenvolvimento das peças bem como a licença e utilização disponibilizada pela Universidade da Beira Interior.

Para a produção/impressão das peças necessárias aos protocolos foi utilizada a *Artillery Genius*, Fig. 25 e a *Artillery Sidewinder*, Fig. 26, duas impressoras 3D disponíveis no Departamento de Engenharia Eletromecânica da Universidade da Beira Interior. Numa impressora 3D podem ser utilizados vários tipos de filamentos, de acordo com à finalidade da peça e o acabamento final pretendido.

Para o desenvolvimento deste trabalho, os materiais utilizados na impressora 3D, foram o PLA (*Polylactic acid*) e o PETG (*PolyEthylene Terephthalate Glycol*), devido às suas propriedades mecânicas e também por serem os mais fáceis de trabalhar e de mais fácil acesso.

Figura 25. Artillery Genius (Artillery, 2020a).

Figura 26. Artillery Sidewinder (Artillery, 2020b).

O PLA, é um tipo de polímero, termoplástico e biodegradável, ou seja, um bioplástico. Quanto ao seu comportamento mecânico, é um material muito duro, o que o torna muito pouco flexível, provocando maior probabilidade de rotura, num curto espaço de tempo. No entanto, o PETG é um polímero muito mais flexível, logo não parte com tanta facilidade, e quando se fala no atrito entre duas superfícies o PETG também é melhor que o PLA, pois este não se deforma tão facilmente, pela sua capacidade de suportar temperaturas mais altas (Szykiedans *et al.*, 2017; Djokikj *et al.*, 2022).

Para uma comparação rápida da rigidez entre o PLA e o PETG, basta observar o gráfico da Fig. 27, onde é possível observar um teste de fadiga entre 3 peças iguais, uma de PLA, outra PETG e a última de PC (*Polycarbonate*), este material não foi usado no trabalho. É possível verificar que o

PETG é mais flexível do que o PLA, pois este parte com mais facilidade (Szykiedans *et al.*, 2017; Djokikj *et al.*, 2022).

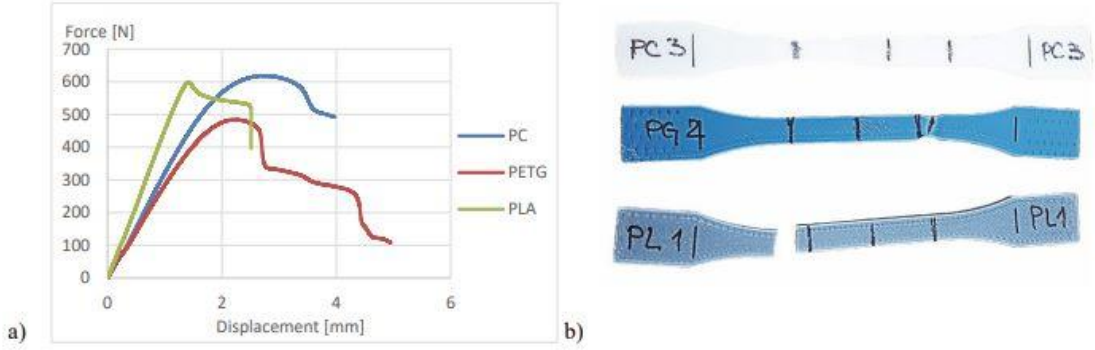


Figura 27. Ensaio de teste à tração. A) grafico da força de tração. B) exemplos das peças depois dos testes. Adaptado de (Djokikj *et al.*, 2022).

O processo de Fabrico Aditivo - AM (*Additive Manufacturing*) é essencialmente composto por três fases (Brown & Beer, 2013):

1. Desenvolver o modelo 3D num *software* de desenho CAD;
2. Converter o ficheiro CAD num ficheiro com o formato estereolitográfico (*StereoLithography* - STL) e através deste ficheiro gerar o percurso por camadas, cujo pode ser lido por uma máquina de controlo numérico computadorizado (*Computer Numeric Control* - CNC), por exemplo G-*codes*;
3. Produção da peça numa máquina de AM.

O código G é um conjunto de cortes transversais do desenho da peça em 3D, que forma uma série de desenhos 2D separados entre si pela altura da camada de material. Esta altura de camada vai depender da qualidade e da rigidez que se pretende dar à peça. No entanto, também é necessário ter em atenção que a altura da camada vai interferir com as propriedades mecânicas da peça final (Brown & Beer, 2013).

Este tipo de impressão 3D - modelação por deposição de material fundido (FDM), constrói os objetos por camadas, como referido acima. Consiste na fundição de um polímero termoplástico, que sai através de um bocal de extrusão, neste caso, a impressão é feita deslocando a cama em y e a cabeça da impressão em *x* e *z*, ou seja, ambas movem-se em simultâneo, designada impressão cartesiana pois a deslocação é feita em relação aos 3 eixos cartesianos (*x*, *y* e *z*) (Huang *et al.*, 2015; Kim *et al.*, 2015).

Para o processo de impressão 3D, Fig. 28, o filamento armazenado no rolo (5) é puxado pelas engrenagens de extrusão até chegar ao bocal de extrusão, onde este vai ser aquecido (1). O filamento é aquecido até chegar a um estado semilíquido, no bocal de extrusão (4) que é o que vai determinar o diâmetro de extrusão. O termoplástico fundido que sai pelo bocal de extrusão (4) é depositado por camadas, na cama (6) que pode ser aquecida ou não. Neste caso, por ser usado o PLA e PETG a cama é aquecida. As camadas são dispostas consoante o ficheiro que contêm o G-*code*, que contém descrito o planeamento de cada camada, Fig, 29 (Huang *et al.*, 2015; Kim *et al.*, 2015).

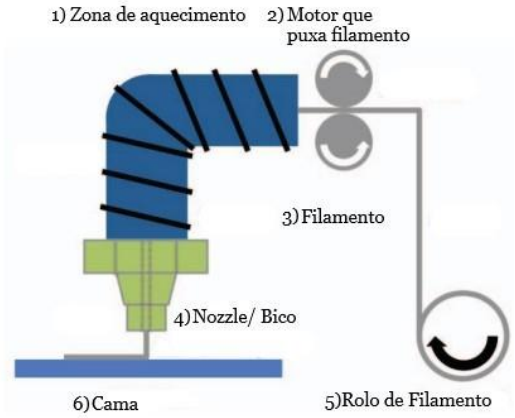


Figura 28. Esquema de funcionamento de uma impressora FDM. Adaptado de Huang *et al.* (2015).

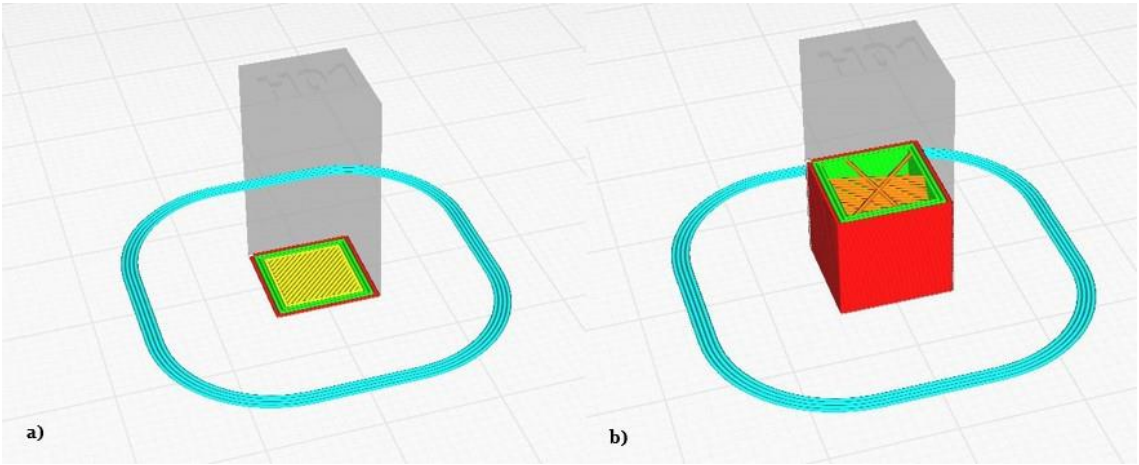


Figura 29. Impressão 3D, visualização do código G. a) simulação da 1º camada. b) simulação a meio da peça.

Para que o PLA e o PETG saiam do bocal de extrusão, estes têm de estar a uma determinada temperatura, como também a temperatura da plataforma de impressão (cama), temperaturas

estas que variam consoante as especificações do material utilizado. As temperaturas, em graus celsius (ºC), necessárias a atingir no processo, estão descritas na Tabela 2 (Djokikj *et al.*, 2022).

Tabela 2. Parâmetros Impressão (Adaptado de: Djokikj *et al.*, 2022).

Material	PLA	PETG
Temperatura de extrusão (ºC)	200 a 220	220 a 250
Temperatura da cama (ºC)	60	80

Para protocolos da área de robótica, é necessário a ferramenta mais importante, um robô. Para a realização deste trabalho, foi usado um robô colaborativo com 6 graus de liberdade, da *Universal Robots*, UR3e, Figura 30 a). Este robô é dotado de 6 juntas rotativas (base, ombro, cotovelo, pulso 1, pulso 2 e pulso 3), que apresentam uma amplitude entre os -360º e os 360º, com exceção da última junta (pulso 3) que consegue rodar infinitamente, para permitir certos trabalhos mais específicos. Este braço robótico, também conhecido por robô colaborativo, apresenta uma carga útil de 3 kg, um alcance de 500 mm e tem um peso de 12 kg. Podem ser consultados mais detalhes em anexo na *datasheet*.

Foi usada a garra 2F-85 da *Robotiq*. Caracteriza-se por ter uma amplitude de abertura máxima de 85 mm, força de 20 a 235 N, pode suportar uma carga útil de até 5 kg e pesa cerca de 0,9 kg. As pinças da garra fecham a uma velocidade entre os 20 a 150 mm/s (Figura 30 b). Também foi usada a câmara do pulso da *Robotiq*. Esta apresenta um campo de visão mínimo de 10 x 7.5 cm, e máximo de visão de 36 x 27 cm. Para que a câmara consiga captar com nitidez a peça que a garra está a manipular, esta tem de variar o seu campo de visão de 10 % a 60%. A câmara apresenta ainda 6 LEDs de luz branca e um alcance de foco mínimo de 7 cm (Figura 30 c).

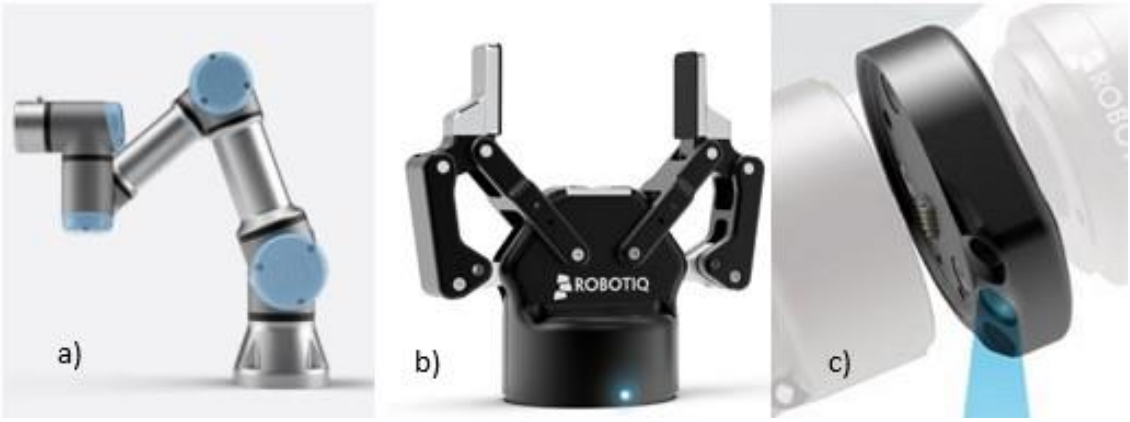


Figura 30. a) Braço robótico UR3e b) Garra 2F-85 c) Câmara de pulso (Robotic, 2020; Robotiq, 2020; Universal Robots, 2021)

3.1.1. Desenvolvimento da pista

Para a realização do protocolo desenvolveu se uma pista, cujo objetivo consiste em que o robô colaborativo a consiga montar e em seguida colocar um berlinde no início do percurso, e este chegue até ao ponto final da mesma. Sendo que seriam colocados obstáculos pelo caminho para dificultar o processo.

O desenvolvimento da pista apresentou vários conceitos e várias fases de desenvolvimento. Numa primeira fase, começou-se por desenvolver os desenhos CAD da pista, posteriormente a impressão para a visualização e a realização de alguns testes.

Nesta fase inicial, foi importante testar as dimensões da pista, bem como as tolerâncias de encaixes e rosca, para uma melhor perceção de qual a opção que preenchia de melhor forma os requisitos pretendidos.

Inicialmente, procedeu-se ao desenvolvimento da base da pista, que tem como objetivo dar suporte a todos os encaixes que a constituem, e ao mesmo tempo servir de local de armazenamento de toda as peças, e obstáculos desenvolvidos para o efeito. Como mostra a Figura 31, a base desenvolvida tem encaixes de forma que todas as peças envolvidas possam ser encaixadas, a configuração da mesma justifica-se pela necessidade de se adaptar a configuração do robô que manipulará as peças.

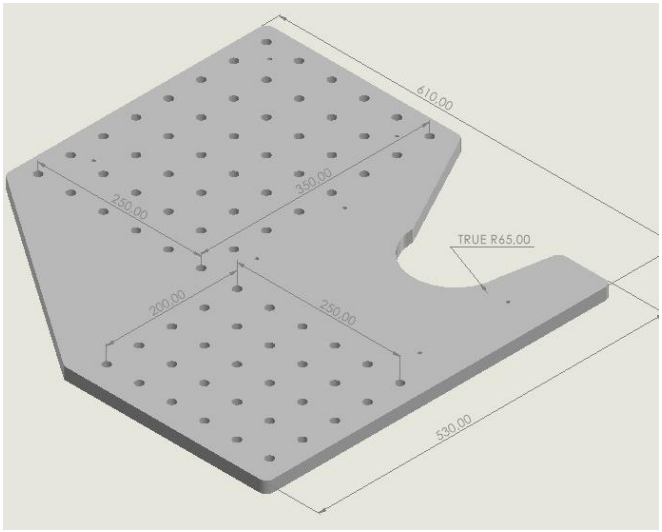


Figura 31. Base da Pista e Armazenamento.

O material escolhido para a base da pista foi o MDF (*Medium Density Fiberboard*). Esta foi maquinada numa CNC Pronun (Figura 32 a). De seguida são apresentadas algumas imagens que descrevem o processo de maquinação (Figura 80 a), b), c), d), e) e f)).

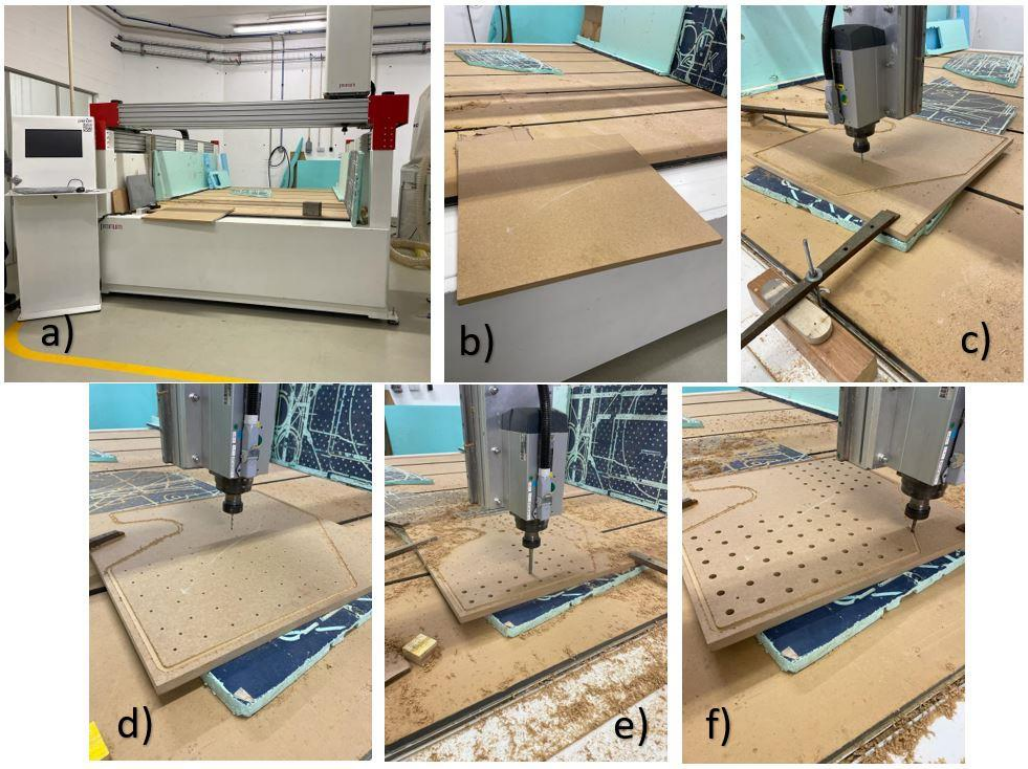


Figura 32. Processo de fabrico da placa de madeira: a) Máquina CNC Pronum; b) Placa de madeira (MDF) 650×650×18mm; c) Fixação da placa; d) Furação de 5mm de diametro; e) Furação de 13 mm de diâmetro; f) Fresagem do contorno exterior.

Posteriormente, a placa de madeira sofreu um processo de tratamento de superfície com o objetivo de lhe proporcionar maior durabilidade e resistência, conforme exposto na Figura 33 a). As seguintes imagens apresentam o processo de tratamento de superfície realizado maquinação (Figura 33 b), c), d) e e).

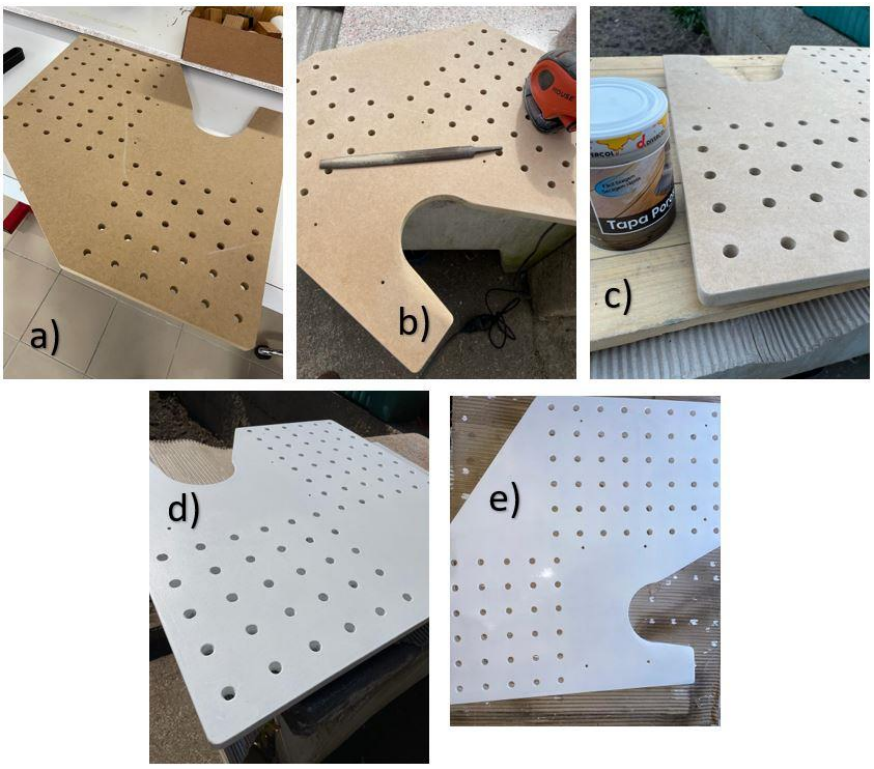


Figura 33. Processo de tratamento: a) Placa de madeira em bruto; b) Processo de lixagem da placa de madeira; c) Aplicação de tapa poros na placa de madeira; d) Aplicação da tinta na placa; e) Aplicação do verniz mate na placa.

Por fim, foram criados uns pinos, apresentados na Figura 34, que vão servir de guias para a montagem da pista. Estes pinos são encaixados nas furações da base, de acordo com a Figura 35. Para os pinos, foram desenvolvidos alguns testes de tolerância de diâmetro, entre os 12 e 13 mm. Sendo que, o que se ajustava mais ao requisito, é o diâmetro de 12,2 mm, pois entrava justo e era necessário fazer alguma força para o retirar, o que para a sua finalidade era perfeito. Foi também criado um suporte para o berlinde, Figura 36, onde este como o nome indica serve para guardar o berlinde até ao robô o ir buscar.

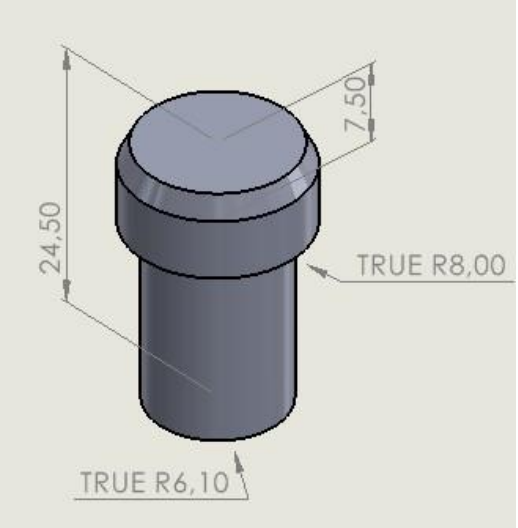


Figura 34. Pino de 12,2 mm.

Figura 35. Pinos inseridos na base.

Um dos últimos elementos criado para a finalização da pista, foi o suporte da esfera, tal como se pode observar na Figura 36. Este foi desenhado para que fosse possível ficar fixo, através do perno inferior, onde o robô pode ir buscar a esfera com alguma facilidade. O pormenor deste componente será que a esfera é assente em dois elementos que permitem que fique centrada em relação à peça, o que vai facilitar a sua manipulação.

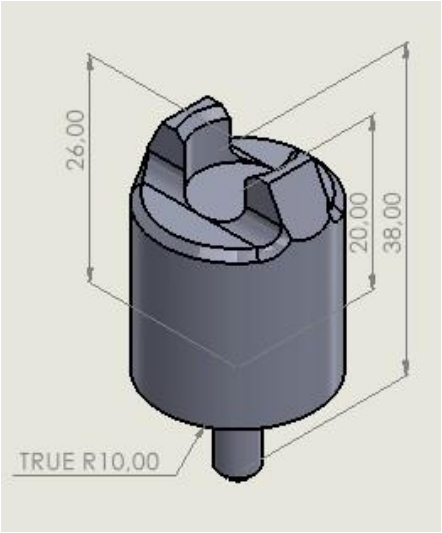


Figura 36. Suporte do berlinde.

3.1.1.1. 1ª iteração do projeto da pista

Numa fase inicial, eram conhecidos dois requisitos básicos que a pista tinha de apresentar:

- Ser montada por partes, que leva à necessidade da mesma apresentar encaixes;

- Possuir inclinação suficiente que permitisse fazer com que a bola se deslocasse.

Assim, e numa fase exploratória, criaram-se encaixes para as várias partes da pista - componentes retos, sendo que estes mostraram-se desde logo um erro.

Uma solução, que se mostrou desde início pouco favorável, foi a construção das diferentes colunas de suporte à pista, serem de alturas diferentes, de forma a permitir inclinação suficiente para a bola/ berlinde rolar. Pois quanto mais comprida a pista fosse, mais peças com diferentes alturas eram necessárias para compor a totalidade da mesma. Pode observar-se na Figura 37 e 38, que quando se monta a pista com 3 suportes são necessárias 11 peças, e só dois comprimentos, o que vai dificultar muito a sua montagem e vai ser uma montagem demorada. Este tempo e dificuldade aumentam à medida que se aumentava o comprimento da pista.

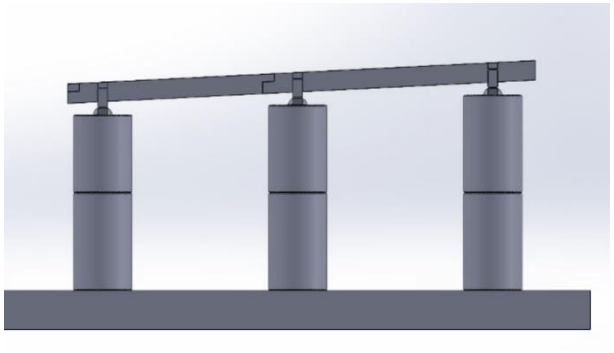


Figura 37. Vista lateral da primeira pista.

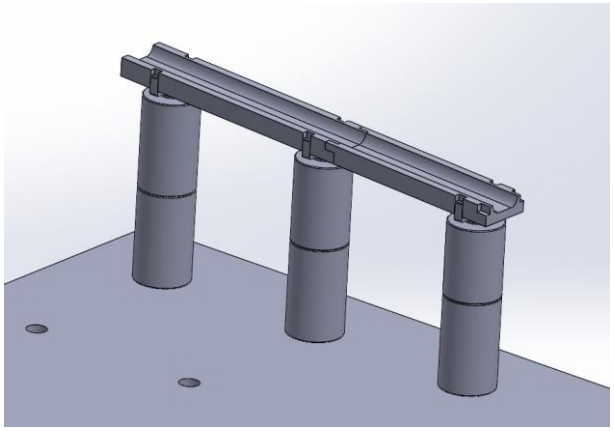


Figura 38. Vista isométrica da primeira pista.

3.1.1.2. 2ª iteração do projeto da pista

Numa segunda iteração (Figura 39), o ponto de partida consistiu em obter-se uma solução onde fosse possível regular a altura de cada pilar da pista. Para tal, foi desenhado um pilar fixo roscado, "macho", conforme apresentado na Figura 40 a), onde entrava outro pilar roscado, "fêmea", que na sua parte superior tinha um encaixe, onde a parte da pista reta e a parte da pista encaixavam. A base desta pista era uma semicircunferência com furação e com rosca espaçados entre si por 10 mm, e possuíam roscas onde o pilar roscado," macho", era fixo.

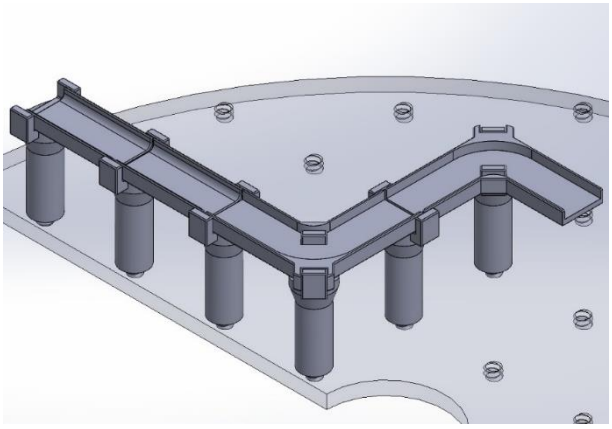
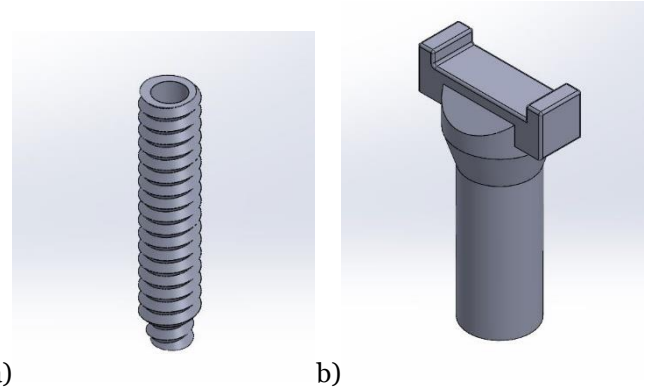


Figura 39. Segunda iteração da montagem da pista.

A montagem da pista consistia também nos componentes retos e curvas (superfície onde a esfera circulavam), como se pode ver na Figura 39. Esta pista apresentava alguns problemas, como a sua complexidade de produção e o tempo que ia demorar na sua montagem, o que inviabilizou esta iteração de desenvolvimento da pista.



a) b)

Figura 40. a) Pilar roscado fixo, "macho". b) Pilar móvel com suporte, "fêmea".

3.1.1.3. 3ª iteração do projeto da pista

A terceira iteração da pista ao conceito de montagem, das colunas "macho" e "fêmea". A grande diferença notada, é na componente reta e curva. Onde estes tinham diferenças nos encaixes e na pista onde a esfera circulava. Esta iteração apresenta duas versões. A primeira onde a componente reta é centrada em relação ao suporte, de acordo com a Figura 41 e 42.

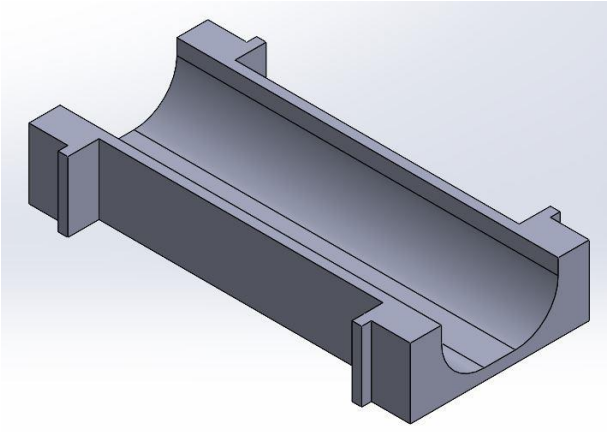


Figura 41. Componente reto da terceira iteração, primeira versão.

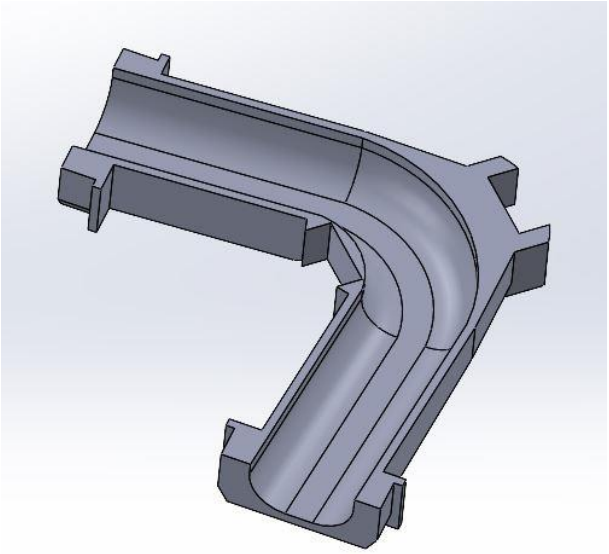


Figura 42. Componente curvo da terceira iteração, primeira versão.

E a segunda versão, que apresenta poucas diferenças em relação à primeira versão. Tanto a componente reta e a curva não estão centradas em relação aos apoios, o que vai permitir à componente curva, conforme Figura 43, fazer uma curva tangente em relação à componente reta, exposta na Figura 44.

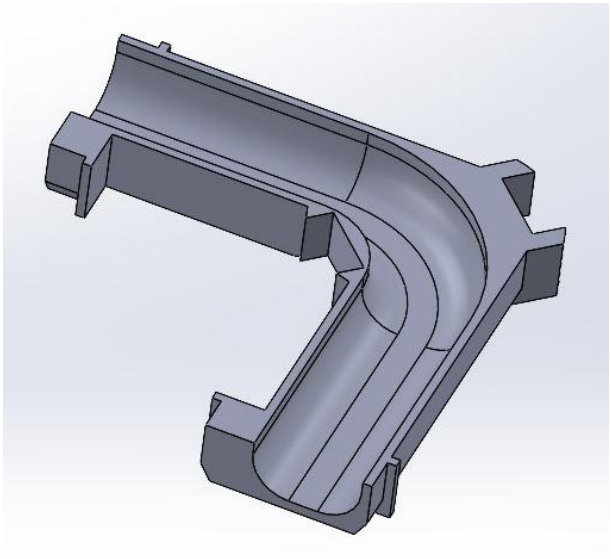


Figura 43. Componente curva da terceira iteração, segunda versão.

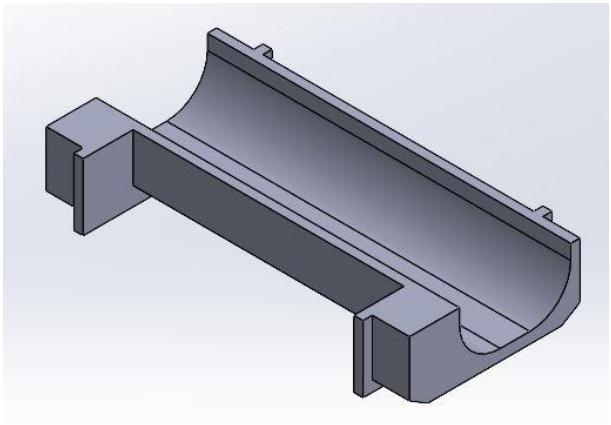


Figura 44. Componente reta da terceira iteração, segunda versão.

3.1.1.4. 4ª iteração do projeto da pista

Nesta iteração aconteceram várias alterações, uma delas, a redução da largura de todos os componentes, e a curva tinha uma curvatura mais suave em relação a curva da iteração anterior, o que facilitava a deslocação da esfera, Figura 45.

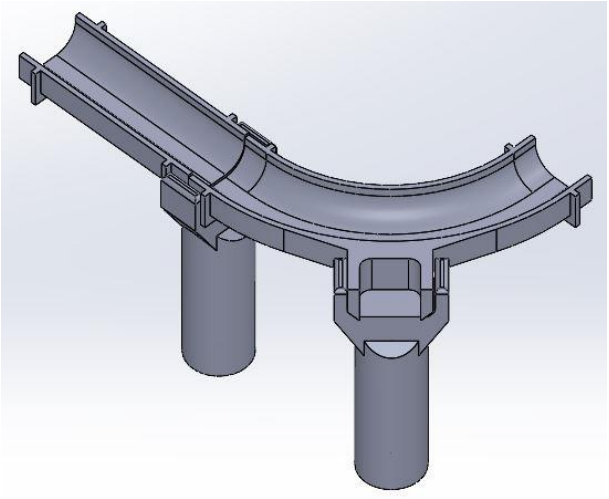


Figura 45. Pista da quarta iteração.

Um ponto negativo desta versão da pista é quando se pretende inclinar o componente curvo, este fica torto em relação às extremidades, o que possivelmente cria um degrau na pista, que poderá impedir que a esfera complete o seu trajeto. Foi pensada uma solução para este problema criando-se uma curva flexível, Figura 46, no entanto era uma tarefa difícil a nível da produção e uma solução pouco robusta.

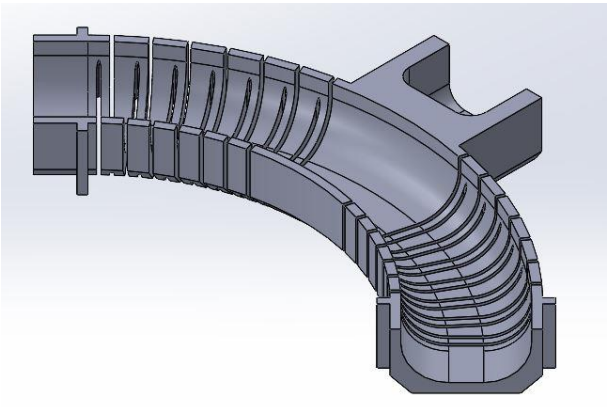


Figura 46. Componente curva da quarta iteração flexível.

3.1.1.5. 5ª iteração do projeto da pista

A última iteração foi inspirada num conceito diferente das iterações anteriores. Tentou-se uma abordagem mais simples, mas ao mesmo tempo mais versátil. Contudo, esta iteração apresenta duas versões, sendo que a segunda versão permite ter maior versatilidade que a primeira.

Esta iteração consiste em 9 componentes diferentes. As versões são semelhantes, sendo a grande diferença entre elas a otimização dos encaixes e algumas partes mais específicas como por exemplo o interior delas que na segunda versão tem um túnel. No entanto, existe o componente obstáculo (obstáculo), de cor vermelha que apresenta duas diferenças entre a primeira versão, e a segunda, Figura 47. Na primeira versão possui 30 mm e na segunda possui 35 mm de altura e o encaixe para a base onde vai ser construída a pista, na parte inferior do componente.

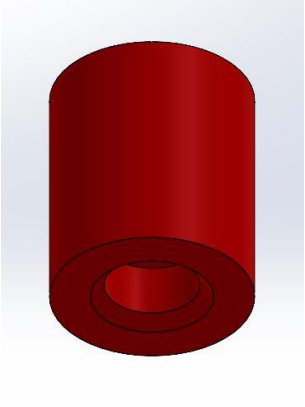


Figura 47. Obstáculo da segunda versão.

Primeira versão

Todas as peças têm a mesma altura em relação ao eixo *z*, o que vai facilitar a montagem. Assim sendo, o que vai realmente ser diferenciador é o comprimento das componentes retas (retas). Existem 5 diferentes retas, onde o comprimento varia de 78 a 263 mm. No comprimento total, no entanto, o que vai realmente interessar é o comprimento entre os seus encaixes, exposto na Figura 48, que variam entre 50 e os 250 mm, com aumentos de 50 mm em cada peça.

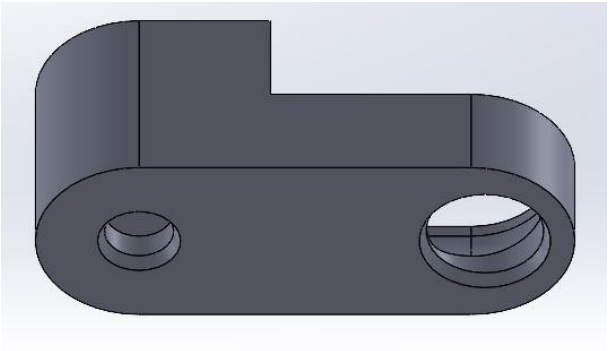


Figura 48. Encaixes da quinta iteração, da primeira versão.

Na Figura 49, pode se ver um exemplo da montagem desta versão com todos os elementos. No entanto existiam alguns problemas, como por exemplo, os encaixes não serem todos iguais e a esfera não podia passar por dentro das alturas o que se tornaria numa mais-valia, do ponto de vista da versatilidade.

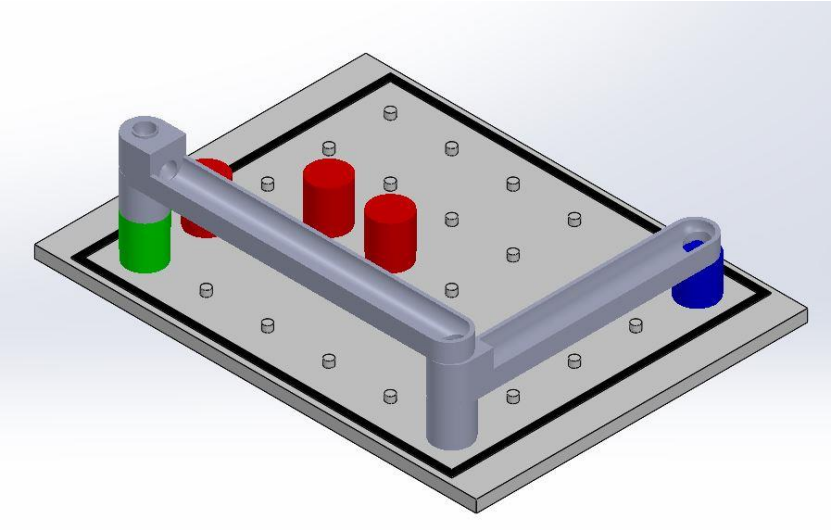


Figura 49. Possibilidade de montagem.

Segunda versão

Esta versão é a selecionada para a pista final. O aspeto é muito semelhante a versão descrita anteriormente, tendo sido os encaixes alterados. Estas alterações servem para uma maior facilidade de montagem e estabilidade da pista, para que esta, a quando, percorrida pela esfera não se desmonte.

Nesta versão, uma das principais diferenças é que a componente "altura", possui uma furação interior permitindo aa esfera, Figura 50, que em vez de passar obrigatoriamente de uma reta para outra reta, pode passar de uma reta para uma altura e só depois para outra reta, de acordo com a Figura 51.

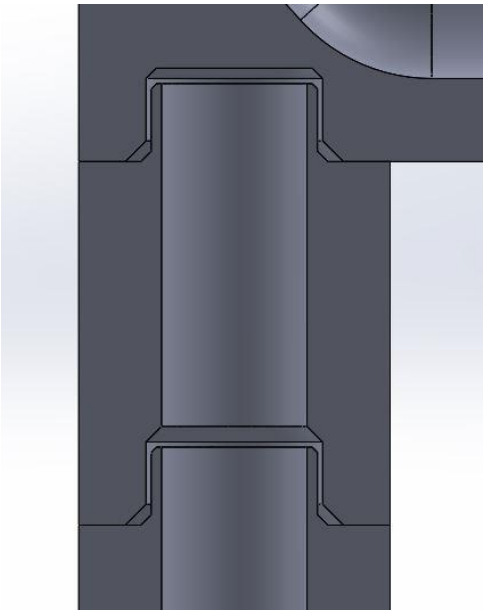


Figura 50. Túnel das alturas – vista em corte.

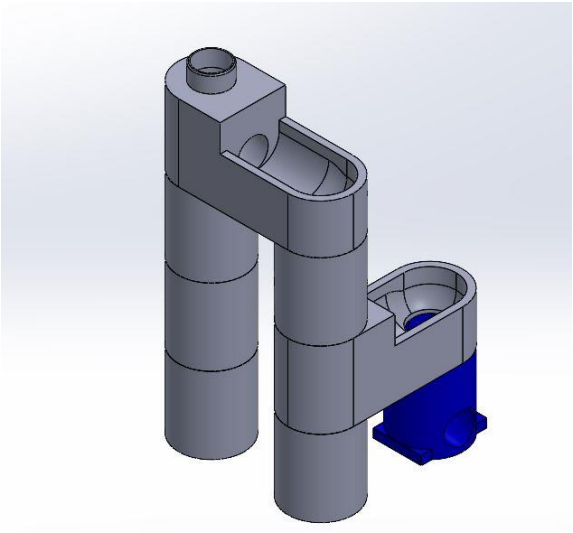


Figura 51. Passagem reta, altura para reta.

Nesta versão, pode-se observar que o encaixe superior, Figura 52, é maior do que na versão 1, o que facilita a estabilidade da pista. Os encaixes inferiores nesta versão são todos iguais, Figura 53, o que vai proporcionar mais versatilidade e não vai criar problemas na montagem. O túnel interior de cada componente reto, Figura 54, começa e acaba tangente á superfície da componente, o que facilita a deslocação do berlinde/ bola.

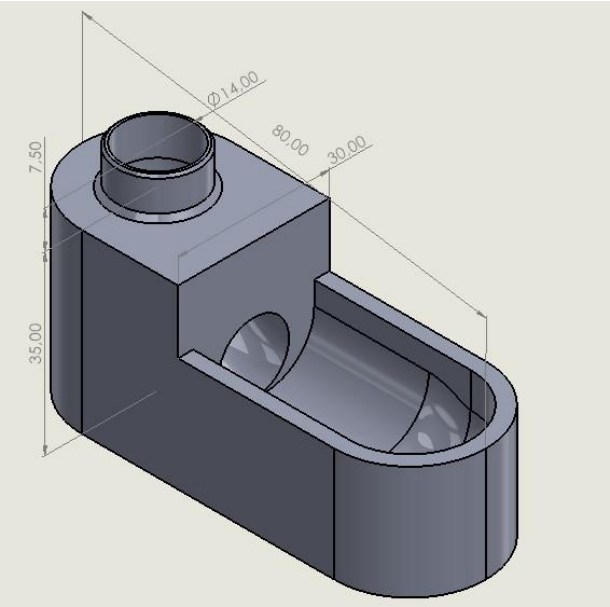


Figura 52. Reta 1, versão 2.

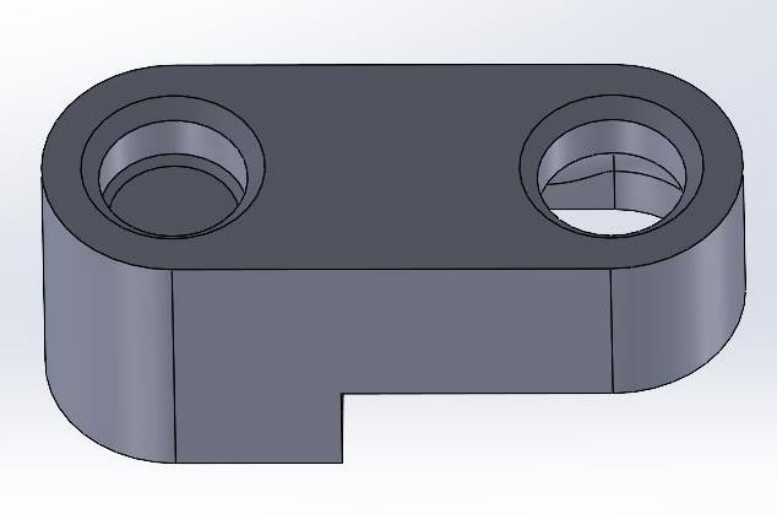


Figura 53. Encaixes reta 1, da quinta iteração da segunda versão.

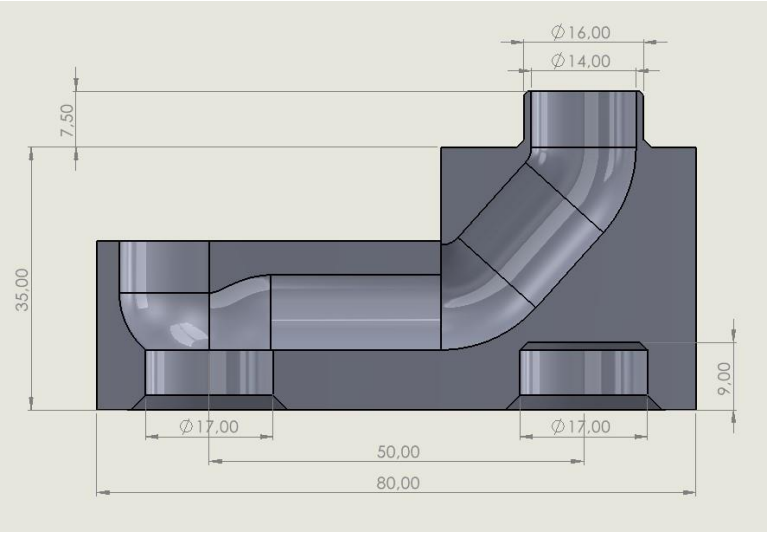


Figura 54. Túnel interior da reta 1, quinta iteração da versão 2. Vista em corte.

Nesta versão, a peça final foi pensada para que a esfera quando chega ao fim do seu percurso fique armazenada na peça de transporte, apresentada na Figura 55. Esta tem uma concavidade redonda no seu centro para que a bola não baloice tanto no seu transporte. Possui ainda umas patilhas laterais para servir de guia quando encosta no componente final, exposta na Figura 56, que também está preparada para que as patilhas laterais do componente de transporte encostem sem dificuldade, como mostram as Figuras 57 e 58.

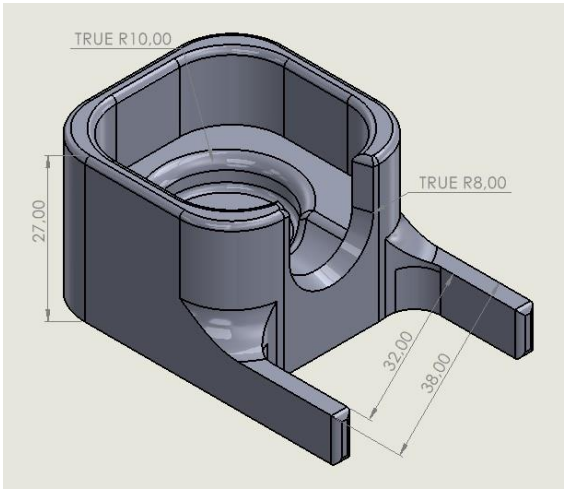


Figura 55. Componente Transporte.

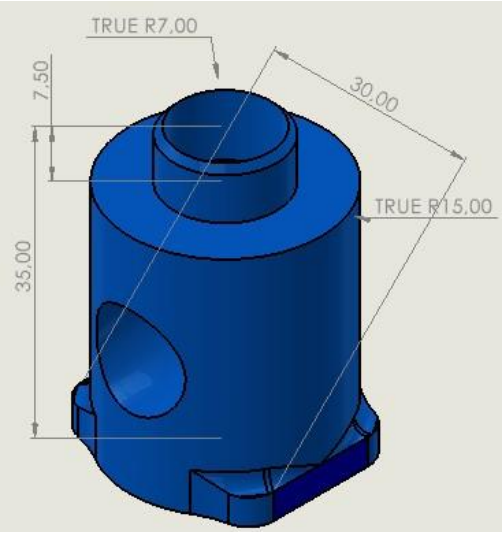


Figura 56. Componente Final.

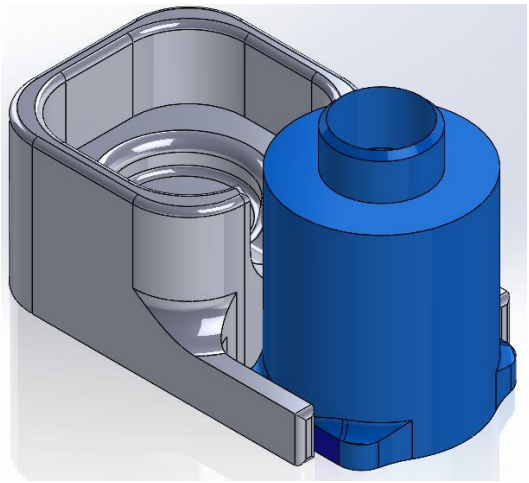


Figura 57. Encaixe do componente final e a peça de transporte.

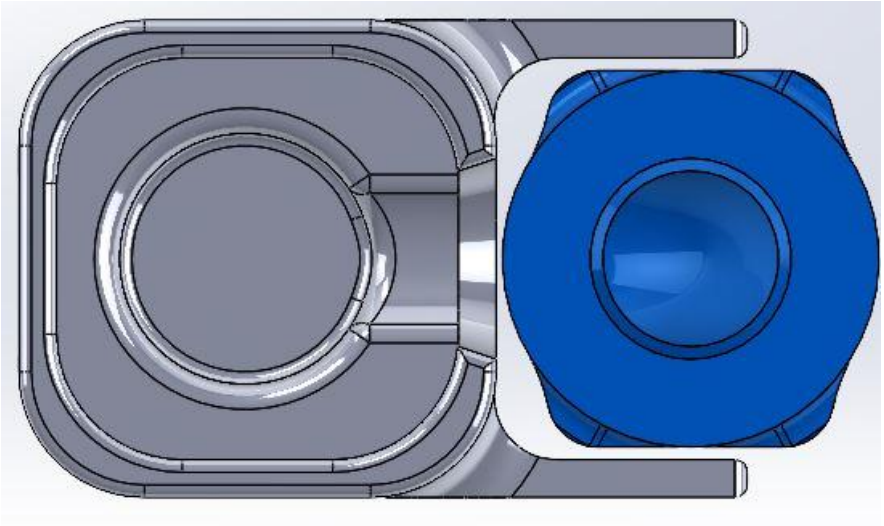


Figura 58. Encaixe do componente final e da peça de transporte, vista topo.

As restantes componentes retas, a reta 2, exposta na Figura 59, a reta 3, exposta na Figura 60, a reta 4, exposta na Figura 61, e a reta 5, exposta na Figura 62, são idênticas à reta 1, exposta na Figura 52, a única diferença que apresentam é o comprimento entre os encaixes na parte inferior, que variam de 50mm, 100 mm, 150 mm, 200 mm, 250 mm, entre cada encaixe, respetivamente. O objetivo de criação de todas estas retas é usar o menor número de componentes na construção da pista final, onde se consegue usar uma reta 4, por exemplo, em vez de várias retas 1 ou até mesmo duas retas 2.

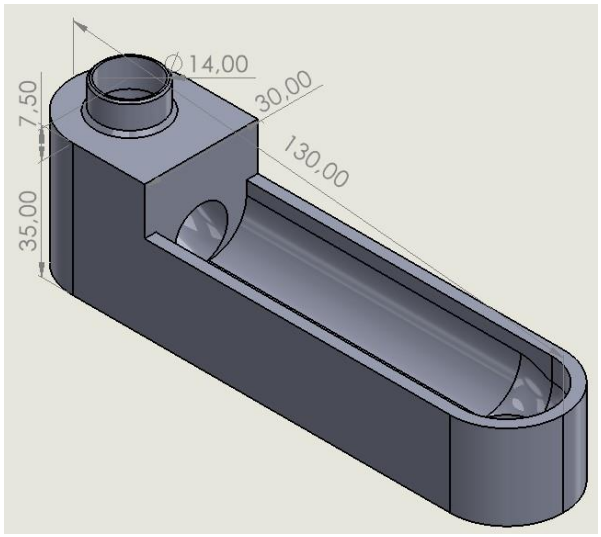


Figura 59. Reta 2, segunda versão.

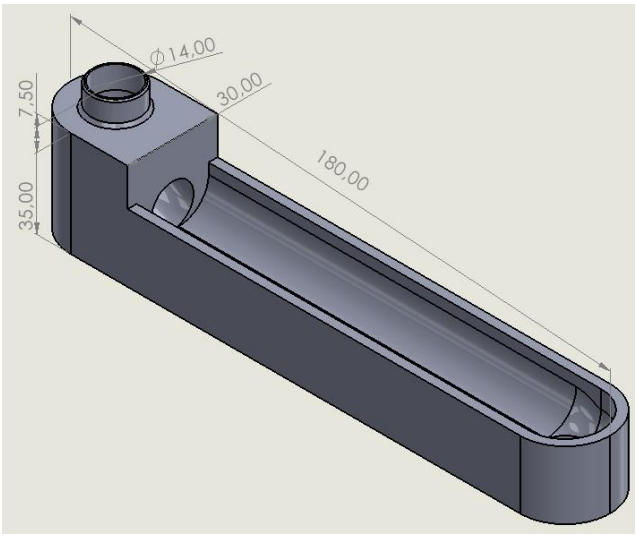


Figura 60. Reta 3, segunda versão.

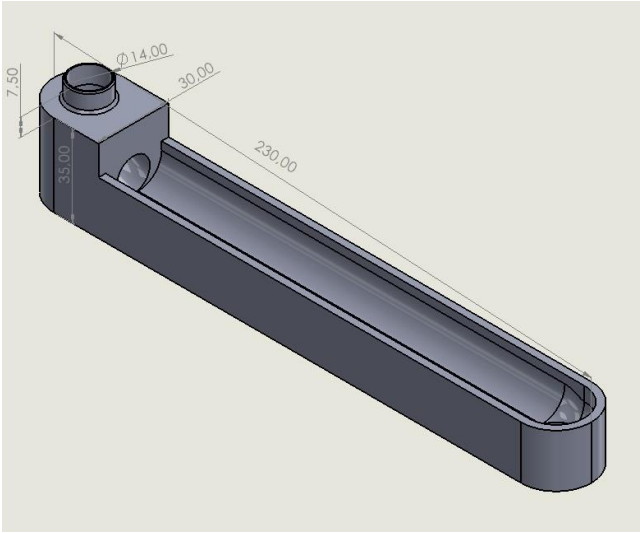


Figura 61. Reta 4, segunda versão.

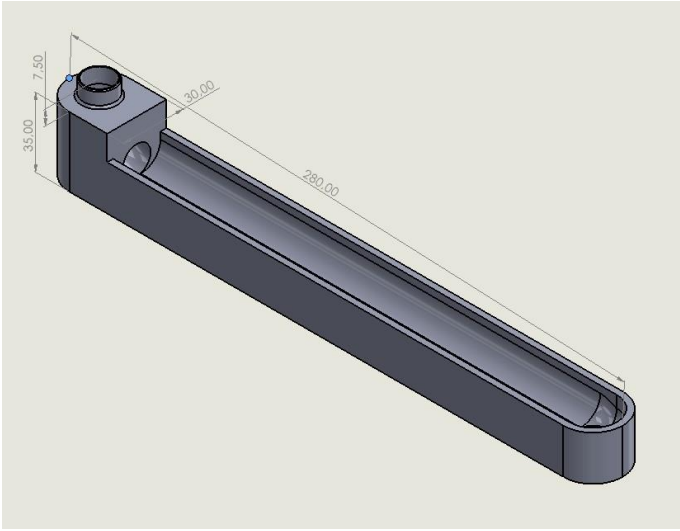


Figura 62. Reta 5, segunda versão.

Uma das alterações nesta versão em relação à primeira versão foram os encaixes. Estes encaixes foram alterados pelo facto de que, na primeira versão eram muito pequenos e não proporcionavam grande estabilidade. Para que a estabilidade da pista melhorasse, e não ser uma preocupação, a pista cair, os encaixes foram adaptados. Os encaixes, expostos na Figura 63 passaram de cerca de 2 mm, para 7.5 mm, de acordo com a Figura 54, o que permitiu ter mais área de contacto, que por sua vez garante mais estabilidade.

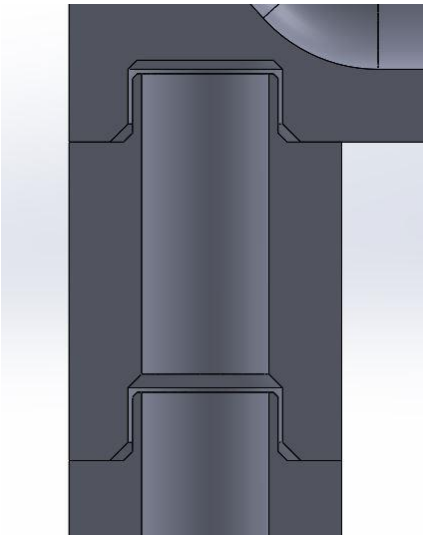


Figura 63. Vista de corte dos encaixes.

Com estes componentes apresentados é possível criar-se a segunda versão da pista 5, exposta na Figura 64, uma versão que foi testada e considerada como aquela que cumpre os requisitos pretendidos.

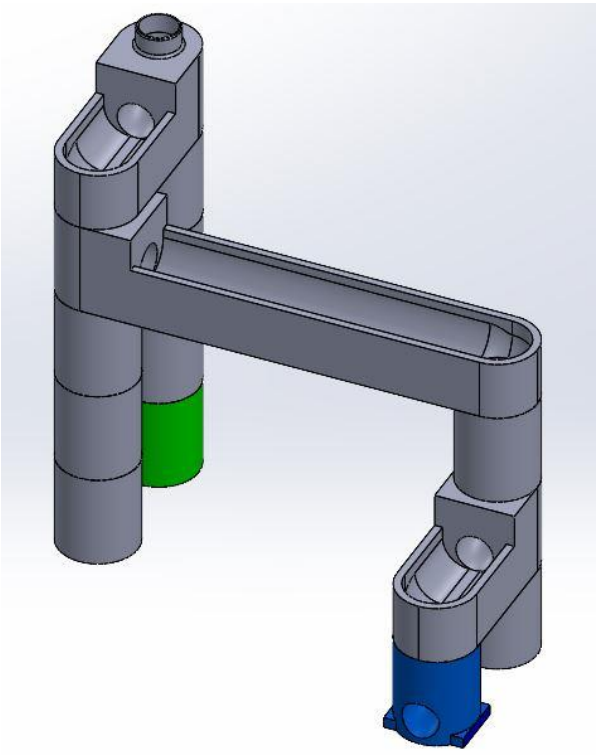


Figura 64. Pista da iteração 5, segunda versão.

3.2. Desenvolvimento do programa

Neste capítulo abordar-se-á todo o desenvolvimento do programa em linguagem de programação *Python*[5]. Devido à elevada extensão, este capítulo encontra-se dividido em 3 partes: a análise de imagem, a análise do caminho mais curto, e por fim a montagem da pista com o robô UR3e.

Como referido acima, este programa foi dividido em 3 partes. Assim sendo, numa primeira fase do programa vai existir uma análise de uma fotografia, que vai ser tirada com o pendente do robô, depois vai ser aberta como um ficheiro de imagem e o programa vai analisá-la, cuja explicação mais detalhada se encontra no subcapítulo 3.2.1. Numa fase seguinte, o programa vai analisar qual é o caminho mais rápido entre os dois pontos, pontos de partida e de chegada, evitando os obstáculos, cujo detalhe se encontra no subcapítulo 3.2.2. Numa fase final, depois do caminho mais curto ser determinado, o programa vai enviar os comandos necessários para que o robô monte a pista correta correspondente ao caminho mais curto, com as restrições identificadas.

À medida que o programa se foi desenrolando foi necessário a adição de extensões, conforme Figura 65, para que fosse possível a execução do programa, sem qualquer erro. Na linha 4, na Figura 65, pode-se observar a extensão que permite abrir ficheiros de imagem/fotografias. Na linha seguinte, esta extensão permite abrir ficheiros guardados no *Windows*. Na linha 6, a extensão indicada serve para a análise de cores. Na linha 7, a extensão tem como finalidade poder-se criar gráficos. Na linha 8, a extensão permite que se trabalhe com *arrays* (estrutura de dados que armazena um conjunto de elementos, que também podem ser chamados de vetores ou matrizes). Na linha 9, a extensão permite inserir ou eliminar elementos de um *queue* (fila - uma coleção de elementos). Na linha 10, a extensão permite que se possa usar blocos de comparação ao longo do programa. Na linha 11, é invocada a extensão para que seja possível a conexão ao computador para controlar o robô, e por fim, na linha 12, a extensão chamada serve para que se possa manipular o tempo ao longo da montagem.

[5] *Python*- linguagem de programação.

```
2   #########  Extenções
3
4   from PIL import Image
5   import os
6   from colorthief import ColorThief
7   import matplotlib.pyplot as plt
8   import numpy as np
9   from collections import deque
10  import heapq
11  import socket
12  import time
13
```

Figura 65. Extensões.

3.2.1. Análise de imagem

Na análise da imagem foi necessário importar a imagem através do nome que o utilizador definir. Depois de abrir a imagem, Figura 66, conforme exposto na linha 18 da Figura 67, esta vai ser cortada para que a base da pista, onde a pista vai ser montada, fique o mais centrada possível, de acordo com o código exposto nas linhas 20 à 27. Quando este processo é realizado, nas linhas 21 a 24, e se define o número de pixéis pretendidos cortar, para se chegar aos valores demonstrados, cortar 70 pixéis á esquerda, 15 na parte superior e inferior e cortar 20 pixéis do lado esquerdo. Estes valores foram obtidos a partir de testes experimentais. Na linha 27, a imagem foi cortada e gravada com o nome: "Teste_robo.jpg", Figura 68, conforme linha 30. Na linha 32, a imagem vai ser mostrada no ecrã do computador.

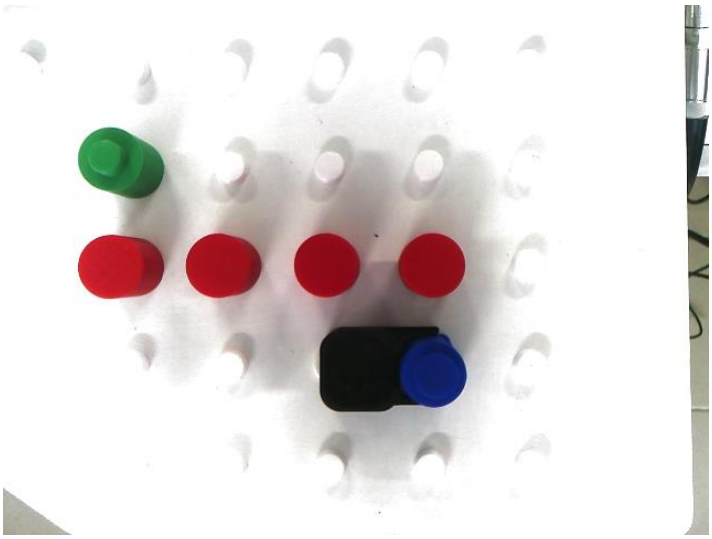


Figura 66. Fotografia tirada pelo robô.

```
15  ##############recortar imagem
16
17  # abrir a imagem
18  imagem = Image.open("G_2_c.jpg")
19
20  # definir as dimensões para cortar a imagem
21  esquerda = 70
22  superior = 15
23  direita = imagem.width - 20
24  inferior = imagem.height - 15
25
26  # recortar a imagem
27  imagem_recortada = imagem.crop((esquerda, superior, direita, inferior))
28
29  # salvar a imagem recortada
30  imagem_recortada.save("Teste_robo.jpg")
31
32  imagem_recortada.show()
33
```

Figura 67. Dimensionamento da imagem.

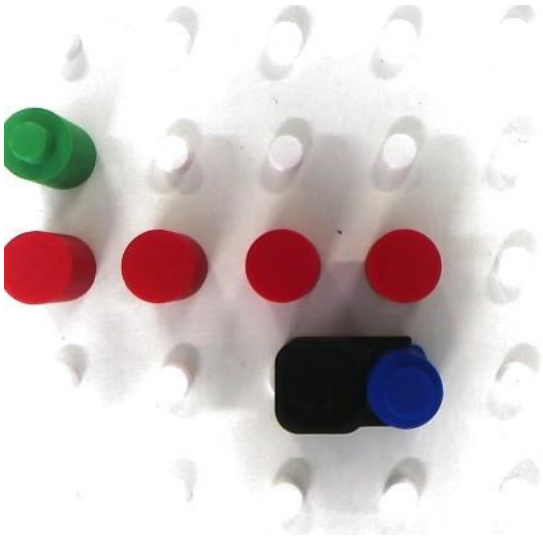


Figura 68. Imagem dimensionada.

Numa fase seguinte, conforme Figura 68, abriu-se a imagem já recortada (linha 35), e em seguida converteu-se a imagem para o sistema de cores (RGB) com o código da linha 38. O código das linhas 43 a 51 realiza uma iteração por todos os pixéis da imagem e guarda o valor dos pixéis se forem pretos. A linha 60 realiza o redimensionamento da imagem para 300x250 pixéis, isto porque numa fase seguinte vai haver uma divisão da imagem original em 30 quadrados iguais, e 300 e 250 são múltiplos de 6 e de 5, respetivamente. Nas últimas linhas da Figura 69, linhas 63 e 66, servem para converter a imagem de novo para o sistema de cores ativas (RGB) e gravar a imagem com o nome “imagem_corte.png”.

```
34  # Abrir a imagem nova
35  img = Image.open("Teste_robo.jpg")
36
37  # Converter a imagem para RGB
38  img = img.convert("RGBA")
39
40  # Imagem data
41  pixels = img.load()
42
43  # Iterate over the pixels
44  for x in range(img.width):
45      for y in range(img.height):
46          # get the pixel value
47          r, g, b, a = pixels[x, y]
48          # check if the pixel is black
49          if r == 0 and g == 0 and b == 0:
50              # set the alpha channel to 0
51              pixels[x, y] = (r, g, b, 0)
52
53  # Get the bounding box of the non-transparent pixels
54  bbox = img.getbbox()
55
56  # Crop the image to the bounding box
57  img = img.crop(bbox)
58
59  # Resize the image
60  img = img.resize((300,250), resample=Image.BICUBIC)
61
62  #converter para RGB
63  img = img.convert("RGB")
64
65  # Save the image
66  img.save("image_corte.png", "PNG")
```

Figura 69. Conversão da imagem no sistema de cores (RGB).

Numa fase seguinte, Figura 70, abre-se a imagem cortada, com o código da linha 71. O código da linha 74 permite obter as dimensões da imagem, definidas anteriormente. Nas linhas 76 a 93 é apresentado o código que realiza a divisão da imagem em 30 quadrados iguais, onde o comprimento é dividido por 6 e a altura por 5. Em seguida, estes quadrados são guardados como uma nova imagem, apresentada na Figura 71.

```
69  ##############    divisão da imagem
70  # abre a imagem
71  im = Image.open("image_corte.png")
72
73  # obtém as dimensões da imagem
74  width, height = im.size
75
76  # calcula o tamanho de cada quadrado
77  square_width = width // 6
78  square_height = height // 5
79
80  # inicializa uma lista vazia para armazenar as imagens divididas
81  squares = []
82
83  # divide a imagem em 30 quadrados iguais
84  for i in range(0, width, square_width):
85      for j in range(0, height, square_height):
86              # recorta o quadrado
87          square = im.crop((i, j, i + square_width, j + square_height))
88              # adiciona o quadrado à lista
89          squares.append(square)
90
91  # salva as imagens divididas
92  for i, square in enumerate(squares):
93      square.save("square{}.png".format(i))
```

Figura 70. Divisão da imagem em 30 quadrados.

Figura 71. Exemplo de um quadrado, onde tem a peça inicial.

De seguida, na Figura 72, é apresentado o trecho de código para cada imagem dos quadrados obtidos anteriormente ser analisada uma a uma, com o objetivo de saber qual a cor do centro de cada quadrado. Desta forma pode saber-se onde está a peça verde, peça que indica o início da pista (ponto de partida), e a peça azul, peça que indica o ponto final da pista (ponto de chegada) e por fim saber em que quadrado se encontra a(s) peça(s) vermelha(s) que indica(m) onde se

encontra(m) o(s) obstáculo(s). Esta verificação das cores, vermelha, azul e verde, não é necessário que seja uma cor perfeito na gama RGB. Podem ser cores mais claras ou mais escuras, como se pode observar nas linhas 105, 111 e 117, onde o código verifica um intervalo de cores e não uma só cor. Numa fase seguinte, guarda-se um vetor com as cores de cada quadrado, de uma forma ordenada para uma análise futura.

Depois de determinar as cores dos quadrados, é possível verificar em que posição se encontra a peça inicial, a peça final e os obstáculos. No entanto, para uma melhor perceção e organização, foi criada uma grelha 6x5, com as cores obtidas anteriormente, o que indica se a cor verde existir está presente a peça inicial, se a cor for azul está-se perante a peça final, se a cor for vermelha está-se perante um obstáculo e se a cor for branca não existe nada naquela posição.

```
95  ########   ver a cor do centro da imagem
96  #def ana_cor():
97  vetor_cor_centro=[]
98  vetor_colors=[]
99
100 for i in range (30):
101
102     def color_name(rgb_color):
103         r, g, b = rgb_color
104         if r > g and r > b:
105             if abs(b-r)>25 and abs(g-r)>25:
106                 return "red"
107             else:
108                 return "white"
109
110         elif g > r and g > b:
111             if abs(g-r)>25 and abs(g-b)>25:
112                 return "green"
113             else:
114                 return "white"
115
116         elif b > r and b > g:
117             if abs(b-r)>25 and abs(g-b)>25:
118                 return "blue"
119             else:
120                 return "white"
121         else:
122             return "white"
123
124     im = Image.open("square{}.png".format(i))
125      # abre a imagem
126     image_files=("square{}.png".format(i))
127
128         # obtém as dimensões da imagem
129     width, height =im.size
130
131         # calcula as coordenadas do centro da imagem
132     center_x = width // 2
133     center_y = height // 2
134
135         # obtém a cor do pixel no centro da imagem
136     center_color = im.getpixel((center_x, center_y))
137     vetor_cor_centro.append(color_name(center_color))
138
139         # imprime a cor do pixel no centro da imagem
140     print(f" A imagem {image_files} tem a cor do centro {center_color} que é {color_name(center_color)}")
141     #criação da grelha
```

Figura 72. Verificação da cor do centro do quadrado.

Na Figura 73, pode observar-se a criação de um gráfico com os objetos nas suas posições, e as diferentes cores. Na linha 149, cria-se uma figura vazia que vai ser preenchida com as cores obtidas anteriormente. A linha 157 à 181 permitem definir os vários parâmetros da grelha, como o tamanho em *x* e em *y*, bem como a orientação dos eixos. A linha 185 permite a criação da grelha com as diferentes cores. Na Figura 74, pode ver-se um exemplo de uma grelha.

```
148  # cria uma figura vazia
149  fig, ax = plt.subplots()
150
151  # lista de cores para as células da grade
152  colors = vetor_cor_centro
153
154  print(len(colors))
155  x =0
156
157  # loop para desenhar retângulos coloridos na grade
158  for i in range(6):
159      for j in range(5):
160
161          ax.add_patch(plt.Rectangle((i, j), 1, 1, color=colors[x]))
162          x=x+1
163
164  # configura o eixo x e y
165  ax.set_xlim(0, 6)
166  ax.set_ylim(0, 5)
167  ax.set_aspect("equal")
168
169  #plt.gca().invert_xaxis()
170  plt.gca().invert_yaxis()
171
172  # mostra a figura
173
174  graph=plt.show()
175
176  #criar grid
177
178  import numpy as np
179  colors = vetor_cor_centro
180  grid_row = 5
181  grid_col = 6
182
183  #####  criação da grid
184
185  grid = np.array(colors).reshape(5, 6, order='F')
186
187  print(grid)
188
```

Figura 73. Criação da grelha.

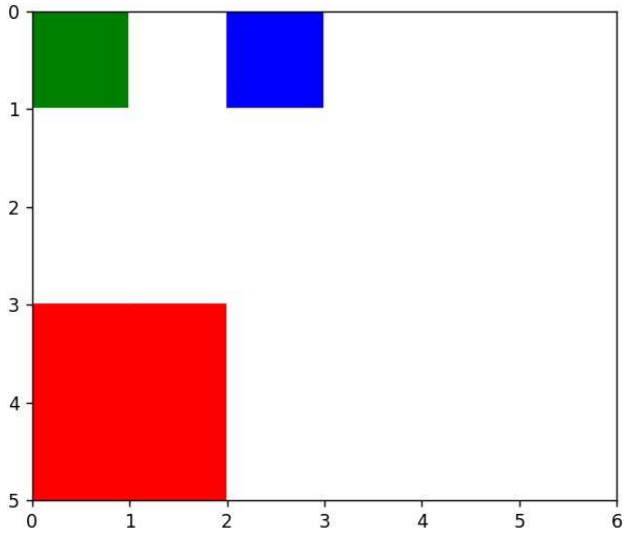


Figura 74. Imagem da grelha.

3.2.2. Determinação do caminho mais curto

Com a grelha já feita já é possível verificar qual o caminho mais curto entre a peça inicial (verde) e a peça final (azul). No entanto, se existir objetos no caminho o percurso vai ser desviado.

Numa primeira fase, analisando a Figura 75, as linhas 193 à 204 permitem percorrer toda a grelha e encontrar o ponto inicial e o ponto final, e guardar os valores das suas coordenadas (x, y). Depois de encontrar o ponto inicial, vai ser inicializado um *while*, que consiste numa repetição de um comando até um objetivo final. Neste caso consiste numa verificação até encontrar o ponto final, evitando os obstáculos. Ou seja, partindo do ponto inicial, vai percorrer a grelha em todas as direções possíveis «(1, 0), (-1, 0), (0, 1), (0, -1)», conforme linha 209, adicionando sempre um valor. Sempre que este percorrer uma direção, de acordo com o exposto na linha 224, e quando encontrar o ponto final, vai comparar qual o caminho, de todos os caminhos possíveis, o que têm um valor mais baixo. O que tiver o valor mais baixo vai ser o caminho mais curto para ir do ponto inicial ao ponto final, evitando os obstáculos.

Ainda na Figura 75, é indicado que depois de se obter o caminho mais curto, dado pela linha 236. Este caminho vai ser dado num conjunto de coordenadas, criando-se um novo gráfico, de acordo com o código da linha 238 à 251, Figura 77, mas agora em vez de ser a grelha equivalente à da fotografia tirada pelo robô, cria-se uma grelha que mostra o caminho mais curto entre o ponto inicial e o ponto final, conforme exposto na Figura 78. Como este caminho é dado em coordenadas vai facilitar a sua análise para a construção da pista.

```python
# caminho mais rapido  ##############################################################################
def find_shortest_path_Astar_curve(grid):
    start = None
    end = None
    for i in range(len(grid)):
        for j in range(len(grid[0])):
            if grid[i][j] == "green":
                start = (i, j)
            elif grid[i][j] == "blue":
                end = (i, j)
            if start and end:
                break
        if start and end:
            break
    if not start or not end:
        return None
    # cria a fila de prioridade com o nó inicial
    heap = [(0, start)]
    previous = {start: None}
    distance = {start: 0}
    directions = [(1, 0), (-1, 0), (0, 1), (0, -1)]
    # loop até encontrar o ponto final
    while heap:
        # pega o nó com menor distância
        current = heapq.heappop(heap)[1]
        if current == end:
            break
        for direction in directions:
            neighbor = (current[0] + direction[0], current[1] + direction[1])
            if 0 <= neighbor[0] < len(grid) and 0 <= neighbor[1] < len(grid[0]) and grid[neighbor[0]][neighbor[1]] != "red" and neighbor not in previous:
                cost = distance[current] + 1 + abs(end[0] - neighbor[0]) + abs(end[1] - neighbor[1]) #calcula distância de Manhattan
                if len(current)>2 and current[2] is not None and current[2] != direction: # verifica se houve mudança de direção
                    cost += 1 # adiciona penalidade para curva
                heapq.heappush(heap, (cost, neighbor, direction))
                previous[neighbor] = current
                distance[neighbor] = distance[current] + 1
    # cria o caminho marcando os pixels com a cor amarela
    path = []
    current = end
    while current:
        path.append(current)
        current = previous[current]
    for pixel in path:
        grid[pixel[0]][pixel[1]] = "yellow"
    return path[::-1]
#mostrar o caminho
path = find_shortest_path_Astar_curve(grid)
print(path)
```

Figura 75. Caminho mais rápido.

```python
#mostrar o caminho a amarelo

grid_row = 5
grid_col = 6

grid1 = [[0 for _ in range(grid_col)] for _ in range(grid_row)]

for coord in path:
    grid1[coord[0]][coord[1]] = 1

print(grid1)

plt.imshow(grid1)
plt.show()
```

Figura 76. Grelha com o caminho mais rápido.

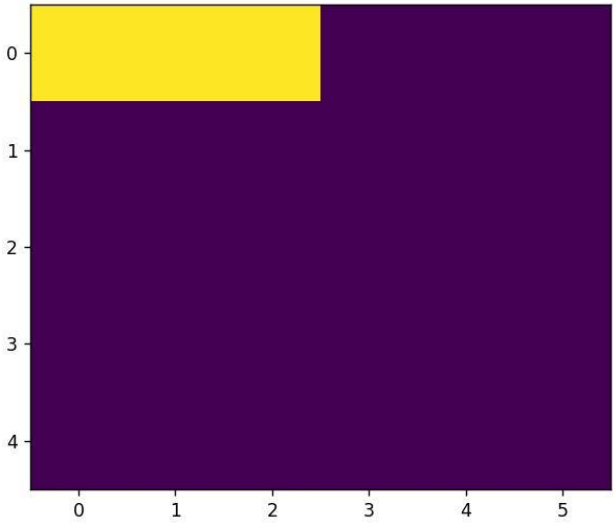


Figura 77. Exemplo de um caminho mais curto.

3.2.3. Análise do caminho

A análise do caminho, tem como objetivo analisar onde é que é necessário levar ou não uma altura, e qual a coordenada onde vai ter de ser colocada, qual é a direção e o comprimento da reta ou das retas que vai ter de ser montada. Para esta análise foram criados vários ciclos *for*, isto é, uma iteração ao longo de um conjunto de valores, um vetor, onde neste caso é o vetor que indica qual é o caminho mais curto.

Numa fase inicial criaram-se várias variáveis e vetores onde à medida que se analisa o caminho, se consiga guardar a informação necessária para a montagem futura, como se pode observar na Figura 78.

```
253  ############################   DAR AS POSIÇÕES DOS PILARES E O COMPRIMENTO DAS PEÇAS DA PISTA
254
255  retavcb=0 #reta vertical cima-baixo
256  retavbc=0 #reta vertical baixo-cima
257
258  retahed=0 #reta horizontal esquerda-direita
259  retahde=0 #reta horizontal direita-esquerda
260  direção=0
261
262  vetor_retahed=[]   #vetor reta horizontal esq-dir
263  vetor_retahde=[]   #vetor reta horizontal dir-esq
264  vetor_retavbc=[]   #vetor reta vertical baixo-cima
265  vetor_retavcb=[]   #vetor reta vertical cima-baixo
266  vetor_direção=[]   #vetor direção
```

Figura 78. Criação de variáveis e vetores.

De seguida, analisa-se em que direção é que o caminho é percorrido, conforme exposto na Figura 79, sendo que existem quatro possibilidades:

- Andar na horizontal da esquerda para a direita, corresponde ao valor 1;
- Andar na horizontal da direita para a esquerda, corresponde ao valor 2;
- Andar na vertical de cima para baixo, corresponde ao valor 3;
- Andar na vertical de baixo para cima, corresponde ao valor 4.

Esta análise, dos respetivos valores foi guardada no "vetor direção" que ajuda quando se estiver a montar a pista, pois é necessário saber em que direção vai ser colocada a reta.

```
for i in range(len(path)):
    try:
        if path[i][0]==path[i+1][0] and path[i][1]<path[i+1][1]:  #verificar se esta a andar horizontal, esquerda-direita
            retahed=retahed+1
            vetor_retahed.append(retahed)
            print("Está a andar para a direita")
            vetor_direção.append(1)


        if path[i][0]==path[i+1][0] and path[i][1]>path[i+1][1]:   #verificar se esta a andar na horizontal, direita-esquerda
           retahde =retahde+1
           vetor_retahde.append(retahde)
           print("Está a andar para a esquerda")
           vetor_direção.append(2)

           #andar na vertical

        if path[i][1]==path[i+1][1] and path[i][0]<path[i+1][0]:  #verificar se esta a andar vertical, cima-baixo, eixo vertical ao contrario
            retavcb=retavcb+1
            vetor_retavcb.append(retavcb)
            print("Está a andar na vertical, B-C")
            vetor_direção.append(3)

        if path[i][1]==path[i+1][1] and path[i][0]>path[i+1][0]:   #verificar se esta a andar na vertical, baixo-cima, eixo vertical esta ao contrario
           retavbc =retavbc+1
           vetor_retavbc.append(retavbc)
           print("Está a andar na vertical, C-B")
           vetor_direção.append(4)
        if i==1:
            vetor_direção.insert(0,vetor_direção[i-1])    ########alterado

    except IndexError:
            print("Acabou a iteração")

print("Andou",retahde, "quadriculas na horizontal direita--> esquerda")
print("Andou",retahed, "quadriculas na horizontal esquerda--> direita")
print("Andou",retavcb, "quadriculas na vertical cima-->  baixo")
print("Andou",retavbc, "quadriculas na vertical baixo--> cima")
print("Vetor direção=",vetor_direção)
```

Figura 79.Verificação da direção das retas.

Como os pilares e as retas vão ser montadas sempre em relação ao ponto inicial e aos pontos onde existem curvas, é necessário saber em que posição do caminho é que vai levar uma peça de altura. Para isso analisasse o "vetor direção" para saber onde existe curva. Esta análise é feita sempre que o valor muda, assim é feita uma análise ao "vetor direção", sempre que o valor do "vetor direção" mudar está presente uma mudança de direção. Esta mudança de direção vai ser guardada num novo vetor o "vetor posição", Figura 80.

```
#dizer em que posição e que existe curva
vetor_posição=[]
vetor_pilar=[]
for i in range(len(vetor_direção)):
    try:
        if vetor_direção[i] != vetor_direção[i+1]:
            print("curva")
            vetor_posição.append(i)
        #    print(posição)
        #posição.clear()
        i=i+1
    except IndexError:
        print()

print("Vetor posição=" ,vetor_posição)
for i in range(len(vetor_posição)):   #saber onde vai levar um pilar

    print("Na posição", vetor_posição[i]+1, "existe uma curva e vai ter de levar um pilar")
    vetor_pilar.append(vetor_posição[i]+1)

print("Pilares nas posições",vetor_pilar)  #posições que vai levar pilar,
#nao esquecer que a pisição inicial e a posição 0
```

Figura 80. Verificação das posições dos pilares.

Para a montagem também é preciso verificar o comprimento de cada reta, conforme exposto na Figura 81, pois estas podem variar de um comprimento de 1 a 6 quadrados. Para se obter o comprimento de cada peça, analisou-se o "vetor posição" onde numa primeira análise verifica-se o comprimento da primeira reta. Em seguida verifica se o comprimento da reta intermédia e por fim o comprimento da reta final. Estes comprimentos vão ficar guardados nas variáveis "peça1" correspondente a reta inicial, "peçax" correspondente a reta intermédia e "peçaf" correspondente à reta final, onde na fase de montagem voltam a ser usadas estas variáveis para verificar qual reta é que o robô tem de montar.

```
#analisar a distancia entre curvas
print(len(vetor_posição))
print(len(vetor_direção))
contar=1

for i in range(len(vetor_posição)):
    try:
        if i==0:
            peça1=vetor_posição[i]+1
            print("A primeira peça tem um comprimento de",peça1)
        if i!=0:
            contar=contar+1
            peçax=(vetor_posição[i]-vetor_posição[i-1])+1
            print("A peça",contar,"tem um comprimento de",peçax)
        if i==len(vetor_posição)-1:
            peçaf=len(vetor_direção)-vetor_posição[i]
            print("A ultima peça tem um comprimento de",peçaf)

    except IndexError:
        print()

```

Figura 81. Verificação do comprimento das retas.

Para a montagem da pista usam-se como referência as coordenadas dos pinos que estão na base, de acordo com a Figura 35. Para saber estas coordenadas, tem de se verificar em que posição no vetor do caminho mais curto existe uma curva, que consequentemente vai levar uma altura, conforme indicado na Figura 82.

Existe uma particularidade. Quando a pista só é constituída por uma reta, o robô só vai ter de montar uma reta, e para isso verifica-se o tamanho da reta novamente, mas para a condição de que só existe uma reta, conforme exposto na Figura 82, da linha 379 à 385. Esta particularidade só é necessária fazer devido ao facto que quando a pista só tem uma reta o "vetor posição" é nulo.

Por fim, analisa-se quantos pilares são necessários para cada posição intermédia. Caso exista uma curva, só existe uma posição intermédia. Caso existam duas curvas, vão existir duas posições intermédias, conforme exposto na Figura 82, da linha 387 à 395.

```
360  ######## coordenadas dos pilares  ###################
361
362
363  print("As coordenadas da posição inicial é",path[0])
364
365  p=1
366  vetor_p=[]          #ordem do pilar, pilar inicial, pilar 2,...
367  pilar=0
368  vetor_pilar=[]       #guarda o numero de pilares que é necessário
369
370  for i in vetor_posição:
371      p=p+1
372      vetor_p.append(p)
373      pilar=pilar+1
374      vetor_pilar.append(pilar)
375      print("As coordendas do pilar", p ,"são", path[i])
376
377  print("As coordenadas da posição final é",path[-1])
378
379  ############## caso a pista so tenho uma reta    ##########
380
381  if vetor_posição == []:
382      peçaf=0
383      peçaf=len(path)
384      vetor_pilar=[0]
385
386
387  ####################   quantidade de pilares em cada posição   ####
388
389  vetor_pilar.reverse()
390
391  for k in range(len(vetor_p)):
392      try:
393          print("O pilar", k+2 , "tem", vetor_pilar[k], "alturas")
394      except IndexError:
395          print()
```

Figura 82. Verificação das coordenadas dos pilares.

3.2.4. Montagem da pista

Para a montagem da pista é necessário que o robô pegue nas peças, nas retas, com as várias dimensões, nas alturas e por fim na esfera. Estas peças vão estar armazenadas sempre na mesma disposição, apresentada na Figura 83, na base de armazenamento. O facto de as peças estarem sempre na mesma posição vai facilitar a programação para que o robô não cometa erros ao ir buscar as peças. Como referido, as peças vão estar armazenadas sempre com a mesma disposição, no entanto foram criados dois pontos, Figura 84, um em cima da base de armazenamento, "home_p_base", Figura 85, e outro em cima da base da pista, "home_p_pista", 86, onde permite que a movimentação do robô não interfira com a pista que possa já estar montada. Estas posições são definidas antes do início da montagem, de acordo com a Figura 84, uma vez que como vão ser chamadas várias vezes ao longo do programa, em vez de se estar a escrever as coordenadas (x, y, z, R_x, R_y, R_z) [6] das duas posições sempre que se pretende invocá-las, basta defini-las no início e depois só invocar pelo nome.

Figura 83. *Layout* das peças armazenadas.

[6] As coordenadas x, y, z, são os valores em metros em relação ao centro da base do robô. R_x, R_y e R_z são as rotações segundo o eixo x, y e z, respetivamente.

```
397  ####################################    DICIONARIOS DE PONTOS ARMAZEM ##################
398
399  pontobasearmazenar_0_0=(0.2591 , -0.1731 , 0.0300 , 3.140 , 0.000 , 0.000)
400  ##### ponto home da base
401  home_p_base=(0.106, -0.30, 0.2673, 2.22, -2.22,0)
402
403  ##### ponto home pista
404  home_p_pista=(0.2869, 0.0285, 0.2673, 2.22, -2.22,0)
405
406  ######## ponto aproximação bola
407
408  ponto_bola_aproximação = (0.2591-0.055 , -0.1173+0.225 ,0.035+0.2 , 2.22 , -2.22 , 0.000)
409
410  ######## ponto apanhar bola
411  ponto_bola = (0.2591-0.055 , -0.1173+0.225 ,0.035 , 2.22 , -2.22 , 0.000)
```

Figura 84. Posições de segurança.

Figura 85. Ponto imediatamente acima da base armazenar.

Figura 86. Ponto imediatamente acima da base da pista.

Como referido acima, a peça vai estar sempre armazenadas no mesmo sítio. Para tal foram guardadas as suas posições numa biblioteca, de acordo com a Figura 87, e também as posições imediatamente acima, que se chamam posições de aproximação, para não existir colisões nas outras peças. Foram criadas duas bibliotecas. Uma com a posição para apanhar a peça, Figura 88, e outra com a posição imediatamente acima, Figura 89. Esta posição é igual às posições para apanhar a peça, sendo a única diferença a adição de 40 mm ao valor inicial do *z*.

```
basearmazenar0 ={
    "Altura1" : (0.2591-0.35 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Altura2" : (0.2591-0.3 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Altura3" : (0.2591-0.25 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Altura4" : (0.2591-0.2 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Altura5" : (0.2591-0.35 , -0.1731-0.2 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta2.1" : (0.2591-0.3 , -0.1731-0.2 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta2.2" : (0.2591-0.1 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta2.3" : (0.2591-0.05 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta3.1" : (0.2591-0.3 , -0.1731-0.1 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta3.2" : (0.2591 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta3.3" : (0.2591 , -0.1731-0.1 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta4.1" : (0.2591-0.35 , -0.1731-0.15 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta4.2" : (0.2591-0.1 , -0.1731-0.15 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta4.3" : (0.2591-0.1 , -0.1731-0.05 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta4.3" : (0.2591-0.05 , -0.1731-0.2 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta5.1" : (0.2591-0.25 , -0.1731-0.2 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta5.2" : (0.2591-0.2 , -0.1731-0.2 , 0.0300 , 3.140 , 0.000 , 0.000),
    "Reta6" : (0.2591-0.15 , -0.1731-0.25 , 0.0300 , 3.140 , 0.000 , 0.000),
}
```

Figura 87. Biblioteca peças armazenadas.

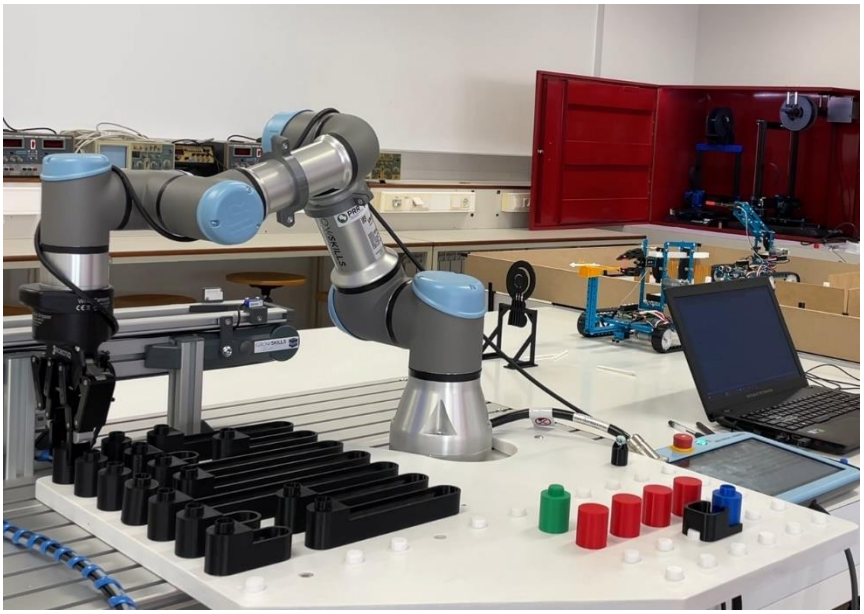


Figura 88. Ponto para apanhar a peça.

Figura 89. Ponto de aproximação.

As posições da base da pista também são guardadas da mesma forma. Criaram-se cinco bibliotecas, onde cada uma corresponde a cada nível da pista. O nível 0 foi denominada de "basepista0", o nível 1, "basepista1" e assim sucessivamente até ao nível 4, "basepista4". As posições do nível 0, de acordo com a Figura 90, têm todas a mesma orientação da garra. No entanto, na parte da montagem das retas, essa orientação vai ter de ser alterada consoante a direção da reta. As bibliotecas dos outros níveis são iguais à biblioteca do nível 0, sendo que a única alteração é o valor do eixo do *z*. A biblioteca do nível 1 vai ter um valor no eixo *z* de 0,07

metros. A biblioteca do nível 2 vai ter um valor no eixo *z* de 0,105 metros. A biblioteca do nível 3 vai ter um valor no eixo *z* de 0,14 metros e a biblioteca do nível 4 vai ter um valor no eixo *z* de 0,175 metros. A criação destas bibliotecas dos diferentes níveis permite que se consiga ter um ponto de aproximação usando o nível anterior.

```
basepista0 = {
    "Ponto_0_0_0" :(0.2591,-0.1173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_0_0" :(0.3091,-0.1173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_0_0" :(0.3591,-0.1173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_0_0" :(0.4091,-0.1173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_0_0" :(0.4591,-0.1173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_0_1_0" :(0.2591,-0.0673,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_1_0" :(0.3091,-0.0673,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_1_0" :(0.3591,-0.0673,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_1_0" :(0.4091,-0.0673,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_1_0" :(0.4591,-0.0673,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_0_2_0" :(0.2591,-0.0173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_2_0" :(0.3091,-0.0173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_2_0" :(0.3591,-0.0173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_2_0" :(0.4091,-0.0173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_2_0" :(0.4591,-0.0173,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_0_3_0" :(0.2591,0.0327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_3_0" :(0.3091,0.0327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_3_0" :(0.3591,0.0327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_3_0" :(0.4091,0.0327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_3_0" :(0.4591,0.0327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_0_4_0" :(0.2591,0.0827,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_4_0" :(0.3091,0.0827,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_4_0" :(0.3591,0.0827,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_4_0" :(0.4091,0.0827,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_4_0" :(0.4591,0.0827,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_0_5_0" :(0.2591,0.1327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_1_5_0" :(0.3091,0.1327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_2_5_0" :(0.3591,0.1327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_3_5_0" :(0.4091,0.1327,0.034, 3.140 ,0.0 , 0.000),
    "Ponto_4_5_0" :(0.4591,0.1327,0.034, 3.140 ,0.0 , 0.000),
}
```

Figura 90. Biblioteca das posições 0 da pista.

Antes do início da montagem da pista é necessário conectar o computador com o controlador do robô, seguindo o código exposto na Figura 91. A conexão é realizada através de uma ligação *Ethernet* onde é necessário dar o endereço do computador e a porta onde este vai estar ligado no controlador do robô.

```
623   HOST = "169.254.148.38" # The remote host
624   PORT = 30002 # The same port as used by the server
625
626   s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
627   s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
628   s.connect((HOST, PORT))
629   time.sleep(0.5)
```

Figura 91. Conexão via socket.

Para o início da montagem, começa-se por abrir a garra cerca de 45% da abertura máxima da garra. Sempre que se pretende abrir a garra, esta só vai abrir os 45% da sua abertura máxima. A abertura máxima da garra foi limitada a 45% para que esta quando for aberta para largar as peças, as pinças não tocarem nas peças imediatamente ao lado e não deitarem a pista abaixo, ou seja não interferirem com o resto da montagem. Quando se pretende fechar a garra, esta vai fechar até estar totalmente fechada. Caso tenha uma peça no meio, vai fechar e segurar a peça até fazer uma força para que consiga transportar uma peça com o peso máximo de 1 kg. Para a garra abrir é necessário dar o comando exposto na Figura 92, e conforme a Figura 93, quando se pretender fechar a garra.

```
631   opengripper = open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')
632   closegripper = open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/close.script', 'rb')
633   #'''
634   o = opengripper.read(1024)
635   while (o):
636       s.send(o)
637       o = opengripper.read(1024)
638   time.sleep(3)
```

Figura 92. Abrir a garra.

```
674       ###### close gripper
675       closegripper = open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/close.script', 'rb')
676       c = closegripper.read(1024)
677       while (c):
678           s.send(c)
679           c = closegripper.read(1024)
680       time.sleep(3)
```

Figura 93. Fechar a garra.

Qualquer montagem deve seguir um procedimento. Na montagem da pista não é diferente, tem de se saber, numa primeira fase, se se está perante uma pista composta só com uma reta, com uma curva (duas retas), ou uma pista com duas curvas (três retas). Esta primeira condição é necessária saber para que o programa entre na condição certa.

Se entrar na condição de uma reta, ou seja, que não existe curvas na pista, o programa vai verificar qual é o tamanho da reta. Depois de saber o tamanho da reta entra na condição. Em seguida, vai

pegar na reta armazenada na base com o tamanho indicado. Para isto o robô vai da posição "inicial", até à posição "casa pista", Figura 86, para a posição "casa base", Figura 85. Em seguida para a posição de "aproximação" à reta a apanhar, Figura 94, de seguida vai para a posição "imediatamente abaixo da posição de aproximação", Figura 95 e quando chega a esta posição fecha a garra e volta a posição "aproximação", já com a peça na garra. Para segurança, o robô volta à posição "casa base" e a seguir "casa pista".

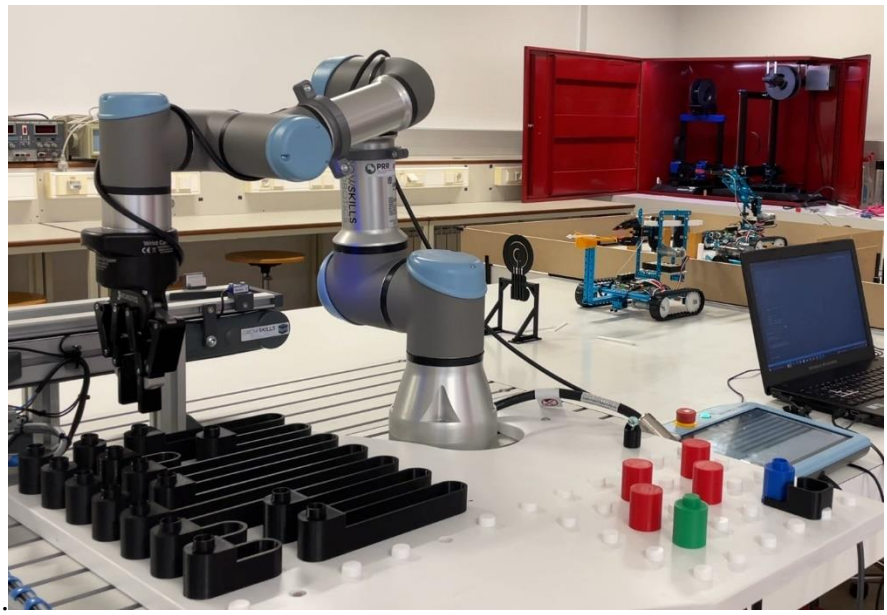


Figura 94. Ponto de aproximação, exemplo.

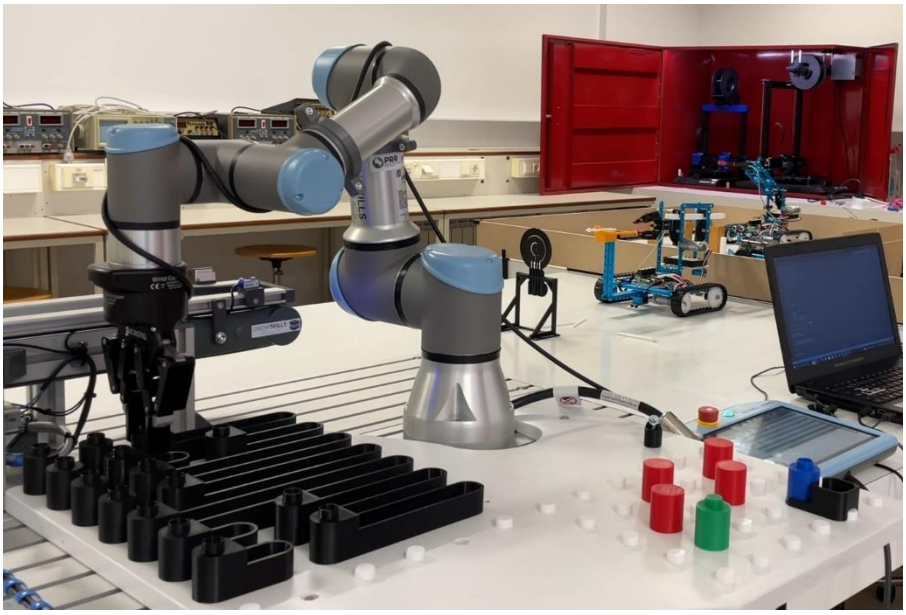


Figura 95. Ponto apanhar reta, exemplo.

Quando chega ao ponto “casa pista”, Figura 96, vai entrar num ciclo para ir identificar qual é o ponto onde tem de colocar a reta. Neste caso é o ponto inicial do caminho. Sendo necessário verificar qual a direção que a reta tem, estas direções estão definidas com as variáveis “1” se a reta tiver a direção esquerda para a direita, “2” se a reta estiver a direção direita esquerda, “3” se a reta tiver a direção cima baixo, sentido positivo do eixo *x* do robô, e “4” se a reta tiver a direção baixo cima, sentido negativo do eixo *x* do robô.

Figura 96. Home pista com reta, exemplo.

Por fim, ao verificar em que direção a reta está, o robô vai do ponto “casa pista”, Figura 96, para o ponto de “aproximação”. No entanto, neste trajeto vai a corrigir a direção da reta caso seja necessário consoante a direção pretendida, Figura 97. Quando chega ao ponto de “aproximação”, Figura 98, ponto do nível acima, já está com a direção pretendida. Logo desta posição vai para a posição abaixo, posição do nível 1, e abre a garra. Depois de largar a peça, volta a ir ao ponto de “aproximação” e logo a seguir para o ponto “casa pista”, Figura 86.

Figura 97. Correção da direção, exemplo.

Figura 98. Ponto de aproximação, exemplo.

Com a pista montada, o único passo em falta é a colocação da esfera para percorrer a pista. Para tal, o robô vai para o ponto de aproximação da esfera que é um ponto acima do ponto para pegar a esfera, Figura 99, em seguida vai para o ponto onde apanha a esfera, Figura 100, e fecha a garra. Depois de apanhar a esfera, volta ao ponto de aproximação e volta para o ponto “casa pista”, Figura 86.

Figura 99. Ponto de aproximação da esfera.

Figura 100. Ponto para apanhar o berlinde.

Quando chega ao ponto “casa pista”, Figura 86, vai ter de verificar qual é o ponto inicial do caminho para colocar a esfera. Quando verifica qual é a posição, este vai para a posição de aproximação, dois níveis acima da posição a que a pista está montada. Em seguida vai para a posição para largar a esfera, Figura 101, que é a posição do nível acima da qual a pista foi montada, mas com uma mudança de menos 15 mm no eixo do *z*, para que a esfera fique o mais próximo possível da entrada para o caminho e não correr o risco de a esfera não percorrer a pista. Existe uma exceção para a largada da esfera. Caso a pista tenha duas curvas (3 pistas), a mudança no

eixo do *z* é só menos 12 mm. Para finalizar, o robô depois de largar a esfera volta à posição "casa pista".

Figura 101. Ponto de largada do berlinde, exemplo.

Na Figura 102, está presente a seção do programa que se a reta for igual a 2 unidades, Figura 47, e a pista tiver só uma camada, vai buscar a reta com tamanho 2 à base de armazenamento e passando pelo processo descrito em cima, volta à posição "casa pista".

```
658 if vetor_posição == []:    #### alturas em funçaõ da peça inicial se for ==2 so vai existir uma reta
659     print("Entrei uma camada")
660     print(peçaf)
661 
662     if peçaf== 2:           #### se a reta tiver um tamanho igual a 2 vai buscar a reta com tamanho 2
663 
664         ###### moverj/l  ponto aproximação######
665         x,y,z,rx,ry,rz=list(basearmazenar1["Reta2.1_1"])
666 
667         s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
668         time.sleep(7)
669         ###### moverj/l ponto apanhar
670         x,y,z,rx,ry,rz=list(basearmazenar0["Reta2.1"])
671 
672         s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.25, r=0)" + "\n").encode('utf8'))
673         time.sleep(5)
674         ###### close gripper
675         closegripper = open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/close.script', 'rb')
676         c = closegripper.read(1024)
677         while (c):
678             s.send(c)
679             c = closegripper.read(1024)
680         time.sleep(3)
681 
682         #### ponto de aproximação
683         x,y,z,rx,ry,rz=list(basearmazenar1["Reta2.1_1"])
684 
685         s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.25, r=0)" + "\n").encode('utf8'))
686         time.sleep(5)
687 
688         #### move ponto home base armazenar
689 
690         x,y,z,rx,ry,rz=home_p_base
691 
692         s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
693         time.sleep(7)
694 
695         ###move home base pista
696 
697         x,y,z,rx,ry,rz=home_p_pista
698 
699         s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
700         time.sleep(7)
```

Figura 102. Condição do comprimento da reta for igual a 2 unidades.

Na Figura 103, está presente a seção do programa onde vai existir a verificação da posição na qual vai ser colocada a reta. Neste caso vai ser o ponto inicial, conforme linhas 704 e 705, em seguida, na linha 706, vai verificar qual é a direção que a reta vai ter. Na linha 713, vai verificar se a direção é igual a "1", e em caso afirmativo vai entrar nesta condição e colocar a peça na posição vista acima.

```
    for i in range(6):
        for j in range(6):
            if path[0]==(i,j):
                #### ponto aproximação
                #x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])

                #s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
                #time.sleep(7)

                if vetor_direção[0]==1:       ######### direção esqueda direita

                    #### ponto aproximação
                    x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
                    time.sleep(7)

                    #### ponto place
                    x,y,z,rx,ry,rz=list(basepista1["Ponto_"+str(i)+"_"+str(j)+"_1"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
                    time.sleep(5)

                    opengripper=open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')   #Robotiq Gripper
                    o = opengripper.read(1024)

                    while (o):
                        s.send(o)
                        o = opengripper.read(1024)
                    time.sleep(3)

                    #### ponto aproximação
                    x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
                    time.sleep(5)
```

Figura 103. Verificação do ponto de colocar a reta e verificação se a direção for igual a “1”.

Na Figura 104 está presente a seção do programa onde se verifica se a direção da reta é igual a “2”, esquerda direita. Se esta condição se verificar, o robô vai rodar a garra para a direção pretendida, através da rotação da sua última junta de rotação. Na Figura 105 e Figura 106, o processo vai ser todo igual, sendo a única diferença a condição relativamente à direção da reta. Se a direção é igual a “3”, cima baixo, ou se a direção é igual a “4”, baixo cima, respetivamente.

```
                if vetor_direção[0]==2:       ##########direção 2, direita esquerda

                    #### ponto aproximação
                    x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",0,3.14,"+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
                    time.sleep(7)

                    #### ponto place
                    x,y,z,rx,ry,rz=list(basepista1["Ponto_"+str(i)+"_"+str(j)+"_1"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",0,3.140,0], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
                    time.sleep(5)

                    opengripper=open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')   #Robotiq Gripper
                    o = opengripper.read(1024)
                    while (o):
                        s.send(o)
                        o = opengripper.read(1024)
                    time.sleep(3)

                    #### ponto aproximação
                    x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])

                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",0,3.140,0], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
                    time.sleep(5)
```

Figura 104. Verificação se a direção for igual a "2".

```
769                 if vetor_direção[0]==3:     ######## direção cima baixo
770 
771                     #### ponto aproximação
772                     x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])
773 
774                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,-2.220,"+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
775                     time.sleep(7)
776 
777                     #### ponto place
778                     x,y,z,rx,ry,rz=list(basepista1["Ponto_"+str(i)+"_"+str(j)+"_1"])
779 
780                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,-2.220,"+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
781                     time.sleep(5)
782 
783                     opengripper=open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')
784                     o = opengripper.read(1024)
785                     while (o):
786                         s.send(o)
787                         o = opengripper.read(1024)
788                     time.sleep(3)
789 
790                     #### ponto aproximação
791                     x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])
792 
793                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,-2.220,"+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
794                     time.sleep(5)
```

Figura 105. Verificação se a direção for igual a "3".

```
796                 if vetor_direção[0]==4:
797 
798                     #### ponto aproximação
799                     x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])
800 
801                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,2.220,"+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
802                     time.sleep(10)
803 
804                     #### ponto place
805                     x,y,z,rx,ry,rz=list(basepista1["Ponto_"+str(i)+"_"+str(j)+"_1"])
806 
807                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,2.220,"+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
808                     time.sleep(5)
809 
810                     opengripper=open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')
811                     o = opengripper.read(1024)
812                     while (o):
813                         s.send(o)
814                         o = opengripper.read(1024)
815                     time.sleep(3)
816 
817                     #### ponto aproximação
818                     x,y,z,rx,ry,rz=list(basepista2["Ponto_"+str(i)+"_"+str(j)+"_2"])
819 
820                     s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+",2.220,2.220,"+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
821                     time.sleep(5)
```

Figura 106. Verificação se a direção for igual a "4".

Na Figura 102 está presente a condição de que se a reta tiver um comprimento de 2 unidades (para os restantes comprimentos da pista, 3, 4, 5 e 6 unidades, funciona da mesma forma. A única diferença é que em vez de ir à base de armazenamento buscar a reta com comprimento de 2 unidades, vai buscar a reta com o comprimento desejado).

Quando for a montagem de uma pista com uma curva, condição que diz que a posição inicial tem 3 níveis, de acordo com a Figura 107, a seção da montagem das retas é igual à seção explicada acima para uma pista de uma reta. No entanto, esta pista vai levar dois componentes “altura” na posição onde se encontra localizada a curva e um componente “altura” imediatamente acima da peça início. Esta pista tem uma ordem na montagem, pois é uma montagem por camadas. Esta montagem começa pela componente “altura”, que é colocada no nível 0, na posição onde se encontra localizada a curva, sendo a primeira camada. Na segunda camada, é colocada uma altura imediatamente acima da peça início, ou seja, a posição inicial do caminho. Em seguida, é colocada

a reta final que tem como posição onde se encontra localizada a curva, no nível 1. Por fim, na terceira camada é colocada a reta inicial que vai ter como posição, a posição inicial do caminho, no entanto no nível 2.

```
1484    ##########################      duas retas
1485
1486    print(path)
1487    if vetor_pilar[0]+2==3:
1488        print("entrou")
1489
```

Figura 107. Verificação da condição inicial, montagem de pista com duas retas.

Quando a montagem já é de uma pista com duas curvas, ou seja, três retas, a montagem é muito idêntica à anterior. Como é uma pista maior, vai conter mais elementos e consequentemente demorar mais tempo a ser montada.

Assim sendo, quando for necessário que o robô monte uma pista de três retas, numa fase inicial vai ser necessário a entrada na condição onde entra o ciclo da montagem pretendida, conforme Figura 108. Depois de entrar na condição, este vai colocar duas peças "altura" no nível 0, nas posições onde existirem curvas. O processo de ir buscar as peças "altura" é o mesmo, já explicado anteriormente. Em seguida, passa-se para a montagem do nível acima, nível 1, onde o robô vai começar por montar duas peças "altura", uma na posição inicial e uma na primeira curva, respetivamente. Por fim, ainda no nível 1, vai montar a última reta do caminho, tendo em conta o seu tamanho e a posição da última curva, bem como a direção em que se encontra através da verificação do vetor direção. No segundo nível, o robô vai montar uma peça "altura" na posição inicial e em seguida, vai montar a reta intermédia, tendo em conta o seu tamanho e a posição da primeira curva e a direção em que esta se encontra no caminho, através da verificação do vetor direção. No terceiro nível e último, vai ser montada a última reta, tendo em conta a posição inicial, o seu tamanho e a sua direção. Numa fase final, o robô vai pegar na esfera e colocá-lo na posição inicial, tendo em conta que se está no nível 3. Na Figura 109, pode ver-se a seção do programa em que são dadas as ordens para o robô ir buscar a esfera. Esta seção é sempre igual para todas as montagens, e coloca-o no ponto inicial, conforme Figura 110. Para o caso de ser uma montagem com três retas, esta seção vai depender da condição inicial que determina quantos níveis vai ter a pista, de acordo com a Figura 111.

```
3443    ########################## tres retas ############ duas curvas
3444
3445    print(path)
3446    if vetor_pilar[0]+2==4:
```

Figura 108. Verificação da condição inicial, montagem de pista com três retas.

```
6695  ################################################ pegar no berlinde e colocao no ponto inical#################################
6696
6697  ####  ponto aproximação pegar bola
6698  x,y,z,rx,ry,rz= ponto_bola_aproximação
6699
6700  s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
6701  time.sleep(6)
6702
6703
6704  ##### ponto pegar bola
6705  x,y,z,rx,ry,rz= ponto_bola
6706
6707  s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
6708  time.sleep(7)
6709
6710  ###### close gripper
6711
6712  closegripper = open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/close.script', 'rb')
6713  c = closegripper.read(1024)
6714  while (c):
6715          s.send(c)
6716          c = closegripper.read(1024)
6717
6718  time.sleep(3)
6719
6720  ####  ponto aproximação pegar bola
6721
6722
6723  x,y,z,rx,ry,rz= ponto_bola_aproximação
6724
6725  s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
6726  time.sleep(4)
6727
6728  #### mover home pista
6729
6730  x,y,z,rx,ry,rz=home_p_pista
6731
6732  s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
6733  time.sleep(6)
```

Figura 109. Ordens para ir buscar a bola.

```
6738    for i in range(6):
6739            for j in range(6):
6740                if path[0]==(i,j):
```

Figura 110. Verificação de qual é o ponto inicial.

```
6797                if vetor_pilar[0]+2==4:     #### existem duas pistas logo a bola e largada imediatamente a cima da pista
6798                    print("berlinde 3º camada")
6799                    ####  ponto aproximação
6800                    x,y,z,rx,ry,rz=list(basepista4["Ponto_"+str(i)+"_"+str(j)+"_4"])
6801
6802                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
6803                    time.sleep(7)
6804                    #### ponto place
6805                    x,y,z,rx,ry,rz=list(basepista4["Ponto_"+str(i)+"_"+str(j)+"_4"])
6806
6807                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z-0.012)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.4, r=0)" + "\n").encode('utf8'))
6808                    time.sleep(5)
6809
6810                    ###### Abrir a garra
6811                    opengripper=open('C:/Users/lucab/OneDrive - Universidade da Beira Interior/DIssertação_/Python/1win0in13uro/open.script', 'rb')   #Robotiq
6812                    o = opengripper.read(1024)
6813                    while (o):
6814                        s.send(o)
6815                        o = opengripper.read(1024)
6816                    time.sleep(3)
6817
6818                    #####
6819                    x,y,z,rx,ry,rz=list(basepista4["Ponto_"+str(i)+"_"+str(j)+"_4"])
6820
6821                    s.send (("movej(p["+str(x)+","+str(y)+","+str(z)+","+str(rx)+","+str(ry)+","+str(rz)+"], a=0.3, v=0.6, r=0)" + "\n").encode('utf8'))
6822                    time.sleep(7)
```

Figura 111. Verificação dos niveis da pista e largada do berlinde.

4. Análise de Resultados

De uma forma geral, o objetivo foi alcançado, um trabalho com algumas dificuldades e contratempos, que se conseguiram ultrapassar. Neste capítulo é abordado o processo de todo o trabalho realizado e as suas etapas, por qual se passou para o conseguir completar.

4.1. Infraestruturas - Análise das etapas

Com a realização da análise de artigos científicos e de trabalhos já realizados no âmbito de tentar interligar a robótica industrial e a robótica na educação, foi possível observar que não existem muitos trabalhos que interliguem estas duas dimensões.

A realização de uma infraestrutura para a qual fosse possível realizar trabalhos onde se pode interligar a robótica industrial e a robótica na educação, foi sempre o ponto de partida e o grande objetivo final. Esta infraestrutura como foi explicada acima, passou por várias versões, sendo que estas várias versões decorreram da tentativa de criar a versão mais versátil e ao mesmo tempo mais simples para poder ser usada por uma faixa etária mais alargada de estudantes. A versão final pode ser usada para ser montado o caminho mais curto evitando os objetos, no entanto também foi pensada para poder ser adaptada, em trabalhos futuros, para que seja o caminho mais curto, mas em vez de evitar os obstáculos, passar por cima dos obstáculos.

De uma certa forma, a base que foi criada para dar apoio à montagem e ao armazenamento das peças para a montagem da pista. Esta pode ser adaptada com novas peças e disposições para atividades diferentes, como por exemplo uma linha de montagem, análise de defeitos em determinadas peças no seu formato ou defeito no seu acabamento (pintura, revestimento, etc.), para uma simulação de soldadura, etc. As oportunidades de continuação deste trabalho até agora desenvolvido são infindáveis.

Com a realização deste trabalho foi possível criar um sistema de apoio ao ensino, o que pode ser o início de um projeto para a dinamização de disciplinas e cursos de robótica industrial ou até mesmo em demonstrações mais dinâmicas e interativas para os alunos. Com algumas alterações e adaptações, este material de apoio pode ser usado para outras disciplinas no âmbito da automação industrial, o que seria uma junção muito interessante do ponto de vista industrial, uma vez que existem muitas indústrias que interligam a automação com a robótica.

4.2. Protocolos laboratoriais

Os dois protocolos desenvolvidos encontram-se em anexo. Estes vão se diferenciar pelo seu grau de dificuldade, onde o segundo protocolo, o mais complexo, está subdividido em três níveis. O

protocolo da atividade 1, Anexo 2, serve para os alunos conhecerem o UR3e e se familiarizarem com o robô. Este protocolo consiste em montar a pista através do pendente do robô e no fim fazer com que o berlinde percorra a pista.

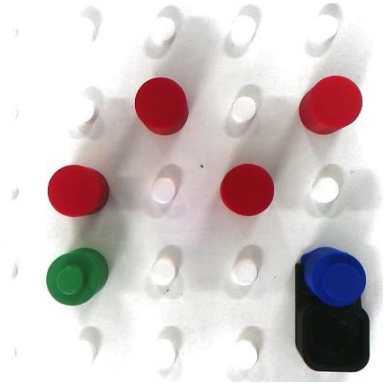
O protocolo da atividade 2, Anexo 2, o mais complexo, vai estar dividido em três níveis. O primeiro nível onde as peças, peça inicial, peça final e os obstáculos, peça verde, peça azul e peças vermelhas, respetivamente, vão estar ordenadas numa determinada posição e o robô tem de as ir buscar e colocá-las no lugar determinado no protocolo.

No nível dois, as peças vão sendo colocadas no tapete rolante de uma determinada forma e o robô vai ter que as ir buscar ao tapete rolante e montá-las no sítio definido no protocolo.

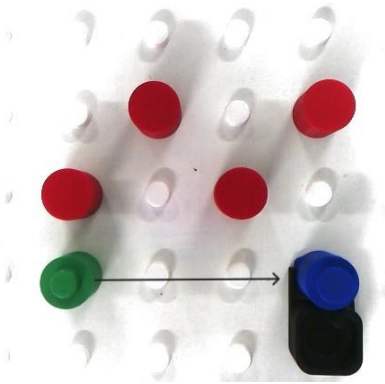
O terceiro nível, o mais complexo o robô vai ter de ir buscar as peças ao tapete rolante, onde estas vão sendo colocadas de forma aleatória no tapete, e vai ser necessário diferenciar qual e a peça que apanhou, através da sua cor, com câmara de pulso.

No decorrer deste trabalho foi desenvolvido com sucesso um programa em *python* que realiza a análise de imagem, consegue descobrir o caminho mais curto através dessa imagem e em seguida montar uma pista consoante o caminho mais curto detetado anteriormente. A pista pode apresentar três tipos de montagem, pista com uma reta, pista com duas retas e uma curva ou uma pista com 3 retas e duas curvas. Esta pista vai depender da disposição das peças quando é tirada a fotografia.

O resultado obtido quando se observa a montagem da pista só com uma reta é possível verificar que o caminho mais rápido vai ser em linha reta, o que facilita na montagem da pista. No entanto, podem existir vários caminhos consoante a disposição da peça inicial e a peça final. Analisando a Figura 112 (a), consegue-se analisar qual é o caminho mais rápido entre o ponto inicial e o ponto final, Figura 112 (b). Para a montagem desta versão vai ser necessário 1 peça, 1 retas. No nível 1, é colocada a única reta da montagem, Figura 112 (c). De seguida, o robô vai colocar a esfera para que este percorra a pista do ponto inicial, peça verde, ao ponto final, peça azul, pelo caminho mais curto, Figura 112 (d).



(a) Exemplo de disposição das peças.



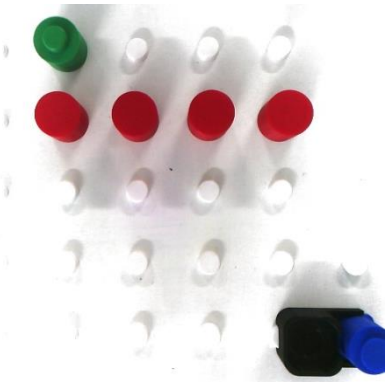
(b) Caminho mais curto

(c) Montagem nível 1.

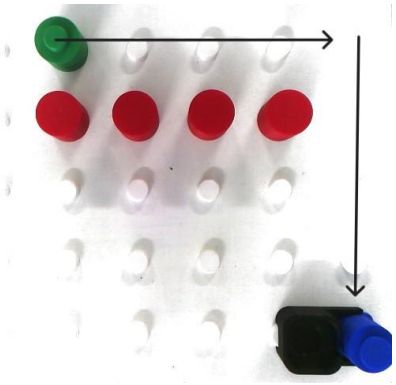
(d) Colocar o berlinde a percorrer a pista.

Figura 112. Exemplificação do Protocolo 1 - pista com uma retas.

Numa pista com duas retas e uma curva, a situação já poderá ser diferente, pois consoante a disposição dos obstáculos pode haver duas possibilidades de um caminho mais curto ou não. O exemplo a seguir, Figura 113 (a), consegue-se facilmente visualizar qual é o caminho mais curto, Figura 113 (b). Para a montagem desta versão vão ser necessárias 4 peças, 2 alturas e 2 retas, divididas em 3 níveis. No nível 1, é colocada uma altura, Figura 113 (c). No nível 2 é colocada a reta final e uma altura, Figura 113 (d). Por fim, no último nível, o nível 3, é colocada a reta inicial, Figura 113 (e). De seguida, o robô vai colocar a esfera para que este percorra a pista do ponto inicial, peça verde, ao ponto final, peça azul, pelo caminho mais curto, Figura 113 (f).



(a) Exemplo de disposição das peças.



(b) Caminho mais curto.

(c) Montagem nível 1.

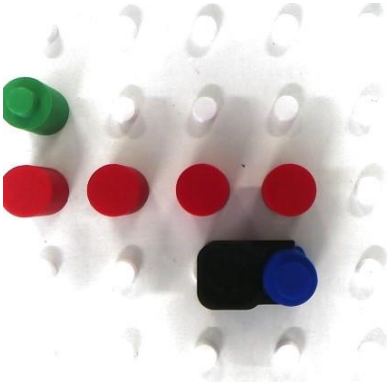
(d) Montagem nível 2.

(e) Montagem nível 3.

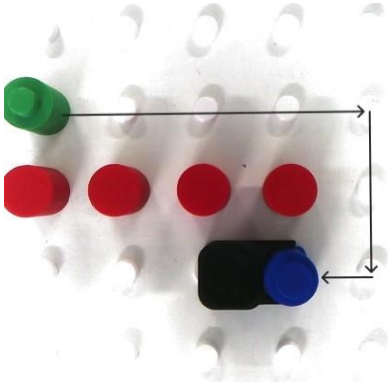
(f) Colocar o berlinde a percorrer a pista.

Figura 113. Exemplificação do Protocolo 1 - pista com duas retas e uma curva.

Numa pista com três retas e duas curvas, consoante a disposição dos obstáculos, pode haver duas possibilidades de um caminho. O exemplo a seguir, Figura 114 (a), consegue-se visualizar qual é o caminho mais curto, Figura 114 (b). Para a montagem desta versão vão ser necessárias 8 peças, 5 alturas e 3 retas, divididas em 4 níveis. No nível 1, são colocadas duas alturas, Figura 114 (c). No nível 2 são colocadas a reta final, a altura intermédia e a altura na posição inicial, Figura 114 (d). Em seguida, no nível 3 são colocadas a reta intermédia e uma altura na posição inicial, Figura 114 (e). No último nível, nível 4, o robô vai colocar a reta inicial, Figura 114 (f). Por fim, o robô vai colocar a esfera para que este percorra a pista do ponto inicial, peça verde, ao ponto final, peça azul, pelo caminho mais curto, Figura 114 (g).



(a) Exemplo de disposição das peças.

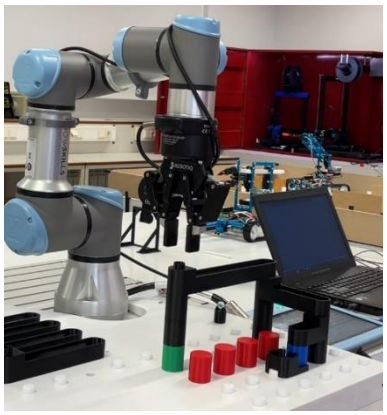


(b) Caminho mais curto.

(c) Montagem nível 1.

(d) Montagem nível 2.

(e) Montagem nível 3.



(f) Montagem nível 5.

(g) Colocar o berlinde a percorrer a pista.

Figura 114. Exemplificação do Protocolo 1 - pista com três retas e duas curva.

5. Conclusões

No desenrolar deste capítulo, são apresentadas as conclusões ao trabalho desenvolvido, bem como algumas propostas de trabalhos futuros, com o intuito de melhoria do sistema criado bem como a melhoria da infraestrutura criada com base em alguns dados tirados após os testes realizados.

5.1. Conclusões gerais

Com as mudanças na sociedade, tanto a nível social, quer a nível económico, a necessidade de satisfazer a procura dos consumidores aumentou. As indústrias vêm-se obrigadas a otimizar as suas produções para colmatar a procura dos consumidores e os aumentos do custo da matéria-prima, e ainda, se não o mais importante, obrigadas a apostarem nestas otimizações para se tornarem competitivos em relação à concorrência.

Com estas mudanças na sociedade e adaptações das indústrias, existe uma necessidade de as entidades educacionais colmatarem estas mudanças com uma melhoria e uma aposta em soluções mais avançadas, soluções mais práticas para que os alunos que frequentem estas unidades curriculares saiam cada vez mais preparados para estes desafios no mundo industrial.

O principal objetivo foi cumprido na sua totalidade, que consistia na criação de um sistema de apoio à educação, na interligação entre a robótica industrial e a robótica na educação. Os objetivos mais específicos foram todos concluídos.

Ao longo da realização deste trabalho, e em particular no desenvolvimento da pista, houve várias iterações. O resultado foi otimizado em termos de versatilidade e maior facilidade de utilização. Esta versão foi capaz de resolver o problema relativo à diferença de alturas, sendo possível montar a pista sem existir desníveis entre o início e o final da reta. Outro problema resolvido foi a não existência de peças curvas, sendo estas conseguidas consoante a direção da componente reta. Para a montagem da pista pelo robô, foi desenvolvido um programa, onde numa primeira fase foram definidas todas as posições necessárias e adicionadas às bibliotecas das posições. Posteriormente, este mesmo programa faz a análise da fotografia tirada pela câmara do braço robótico, onde esta vai ser analisada de forma a detetar o caminho mais curto. Ao qual o robô monta a pista consoante o caminho encontrado. Este processo todo vai ser estipulado pelos protocolos da atividade laboratorial, que apresenta vários níveis de dificuldade.

O primeiro protocolo, um protocolo mais simples, mas ao mesmo tempo desafiante, a montagem da pista com o pendente do robô. O segundo protocolo, um protocolo que vai estar dividido em três níveis, no entanto mais complexo que o primeiro pois este vai ser necessário usar o pendente do robô e o programa externo em *python*. Este trabalho foi realizado em âmbito académico o que

proporcionou um conjunto de ajudas, tanto a nível de apoio pessoal como a nível de apoio logístico, tanto a nível de ferramentas, como do espaço para desenvolver este trabalho. Serviços como a maquinaria CNC, para a maquinação da base, as impressoras 3D, para a produção de toda as peças necessárias para a montagem da pista e que permitissem com que a base ficasse completa e também o espaço disponibilizado para a realização deste projeto.

Por fim, espera-se que esta dissertação tenha contribuído para uma melhor perceção da importância quer ao nível da robótica industrial, como da robótica educacional. Contudo, espera-se também que este trabalho tenha demonstrado a importância que é a junção destes dois grandes temas num ambiente educacional, com o objetivo de uma melhor preparação, quer do pessoal docente, quer dos alunos, que poderão beneficiar e sentir-se mais capazes quando entrarem para o mundo do trabalho.

5.2. Sugestões de trabalhos futuros

À medida que o trabalho é desenvolvido, vão havendo conceitos, temas que sendo estudados trazem novas perguntas e novas ideias de aplicar o objetivo principal deste trabalho. Assim sendo, vão ser apresentado algumas propostas de trabalhos futuros ou formas de abordagem diferente em consequência deste trabalho, e da pesquisa desenvolvida.

Uma das propostas, é a criação de uma nova base e material de apoio que consigam interligar uma variedade maior de trabalhos desenvolvidos. Consequentemente, interligar várias disciplinas e vários materiais existentes.

Outra proposta, é a criação de mais protocolos para outras aplicações da robótica na indústria, como a soldadura, a separação de material defeituoso, etc.

A última proposta, é a melhoria e otimização do programa de montagem da pista, otimizá-lo ao nível do seu tamanho e tentar tornar a montagem mais rápida.

Referências Bibliográficas

ABB. (2015, April 13). *ABB introduces YuMi, world's first truly collaborative dual-arm robot*. https://new.abb.com/news/detail/12952/abb-introduces-yumir-worlds-first-truly-collaborative-dual-arm-robot

ABB. (2021). *Introducing GoFa*.

Alaraje N., Sergeyev A., & Le B. (2010). *Promoting Robotics Education: Curriculum and State-of-the-Art Robotics Laboratory Development Related papers An Int erdisciplinary Laborat ory for Teaching Art ificial Int elligence and Manufact uring Promoting Robotics Education: Curriculum and State-of-the-Art Robotics Laboratory Development*. http://technologyinterface.nmsu.edu/Spring10/

Almaleh A., Aslam M. A., Saeedi K., & Aljohani N. R. (2019). Align my curriculum: A framework to bridge the gap between acquired university curriculum and required market skills. *Sustainability (Switzerland)*, *11*(9). https://doi.org/10.3390/su11092607

Artillery. (2020a). *Artillery Genius*. https://artillery3d.com/products/artillery-genius-3d-printer-kit-220220250mm-print-size-with-ultra-quiet-stepper-motor-tft-touch-screen

Artillery. (2020b). *Artillery Sidewinder*. https://artillery3d.com/products/artillery-sidewinder-x2-upgrade-version-abl-auto-calibration-3d-printer-550405640mm-larger-printed-size-118111811575-inches-high-precision-dual-37

Arulkirubakaran D, Rasalin Prince Malkiya R., Nagarajesh T. C. S., Palanisamy D., Neil Anand Koppisetti, Siddharth S., Nagasuresh Manni, Kishore K. CH. S., & Samuel Abhishek Nalla. (2022). Evolution of Industrial Robots During Mid of Nineteenth Century–Beginning of Twentieth Century—A Review. In *Recent Advances in Materials An Modern Manufacturing* (pp. 142–153). https://doi.org/10.1007/978-981-19-0244-4_15

Atman Uslu N., Yavuz G. Ö., & Koçak Usluel Y. (2022). A systematic review study on educational robotics and robots. In *Interactive Learning Environments*. https://doi.org/10.1080/10494820.2021.2023890

Azevedo S., Aglaé A., & Pitta R. (2010). *Minicurso: Introdução a Robótica Educacional*. http://www.robotics.org/

Barua R., Datta S., Mengade A., & Patil P. (2020). Modernization of Robotics Application in 21st Century: A Review. *Journal of Mechanical Robotics*, *5*(2), 24–34. https://doi.org/10.46610/jmmdm.2020.v05i02.005

Berry C. A., Gennert M. A., & Marie Reck R. (2020). *Practical Skills for Students in Mechatronics and Robotics Education*.

Brown A. C., & Beer D. de. (2013). *Development of a stereolithography (STL) slicing and G-Code generation algorithm for an entry level 3-D printer*.

Cukosky R. Mark, & Wright K. Paul. (1982). *POSITION SENSING WRISTS FOR INDUSTRIAL MANIPULATORS*.

D. Arulkirubakaran, R. MAlkiya Rasalin Price, T. C. S. Nagarajesh, D. Palanisamy, Koppisetti Neil Anand, S. Siddharth, Manni Nagasuresh, K. CH. S. Kishore, & Nalla Samuel Abjishek. (2022). *Evolution of Industrial Robots During Mid of Nineteenth Century–Beginning of Twentieth Century—A Review*. https://doi.org//https://doi.org/10.1007/978-981-19-0244-4_15

Djokikj J., Tuteski O., Doncheva E., & Hadjieva B. (2022). Experimental investigation on mechanical properties of FFF parts using different materials. *Procedia Structural Integrity*, *41*(C), 670–679. https://doi.org/10.1016/j.prostr.2022.05.076

Do H. D., Tsai K. T., Wen J. M., & Huang S. K. (2022). Hard Skill Gap between University Education and the Robotic Industry. *Journal of Computer Information Systems*, *00*(00), 1–13. https://doi.org/10.1080/08874417.2021.2023336

Eguchi A. (2017). Bringing robotics in classrooms. In *Robotics in STEM Education: Redesigning the Learning Experience* (pp. 3–31). Springer International Publishing. https://doi.org/10.1007/978-3-319-57786-9_1

Enrica Merlo. (2018, March 17). *A MODERNIDADE DE JACQUES DE VAUCANSON: O PATO DIGESTIVO E A MELHOR OFERTA*. http://www.mimancanoifondamentali.com/2018/03/la-modernita-di-jacques-de-vaucanson.html

Gao X., Li P., Shen J., & Sun H. (2020). Reviewing assessment of student learning in interdisciplinary STEM education. In *International Journal of STEM Education* (Vol. 7, Issue 1). Springer. https://doi.org/10.1186/s40594-020-00225-4

Garcia Elena, Jimenez Maria Antonia, De Santos Pablo Gonzales, & Armada Manuel. (2007). The evolution of robotics research. *Robotics & Automation Magazine*.

Gasparetto A. (2016). Robots in history: Legends and prototypes from ancient times to the industrial revolution. *History of Mechanism and Machine Science*, *32*, 39–49. https://doi.org/10.1007/978-3-319-31184-5_5

Gasparetto A., & Scalera L. (2019). A Brief History of Industrial Robotics in the 20th Century. *Advances in Historical Studies*, *8*, 24–35. https://doi.org/10.4236/cm.2019.81002

Ge S. S., Zhao D., Li D., Mao X., & Nemati A. (2020). Historical and futuristic perspectives of robotics. *Artificial Life and Robotics*, *25*(3), 393–399. https://doi.org/10.1007/s10015-020-00613-7

Giralt G., Chatila R., & Vaisset M. (1990). An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots. *Autonomous Robot Vehicles*.

Grau A., Indri M., lo Bello L., & Sauter T. (2017). *Industrial Robotics in Factory Automation: from the Early Stage to the Internet of Things*.

Hägele M., Nilsson K., Pires J. N., & Bischoff R. (2016). Robots at Work. In *Robots at Work* (pp. 1385–1421).

Herakovic N. (2010). *Robot Vision in Industrial Assembly and Quality Control Processes*.

Hollingum Jack. (1994). ABB focus on "lean robotization." In *Industrial Robots* (Vol. 21, pp. 15–16).

Huang T., Wang S., & He K. (2015). Quality Control for Fused Deposition Modeling Based Additive Manufacturing: Current Research and Future Trends. *The First International Conference on Reliability Systems Engineering*.

Jiang Z., Xiong W., Du H., Wang Z., & Wang L. (2021). Energy-saving methods in pneumatic actuator stroke using compressed air. *The Journal of Engineering*, *2021*(5), 241–251. https://doi.org/10.1049/tje2.12000

Joseph L. Jones, Newton E. Mack, David M. Nugent, & Paul E. Sandin. (2005). *UTONOMOUS FLOOR-CLEANING ROBOT*.

Karlsson J. (1991). *A decade of robotics; analysis of the diffusion of industrial robots in the 1980s by countries, application areas, industrial branches and types of robots*. Mekanförbundets Förlag.

Kim C., Espalin D., Cuaron A., Perez M. A., MacDonald E., & Wicker R. B. (2015, July 10). A Study to Detect a Material Deposition Status in Fused Deposition Modeling Technology. *International Conference on Advanced Intelligent Mechatronics*.

Kirgis F.-P., Katsos P., & Kohlmaier M. (2016). Collaborative Robotics. In *Robotic Fabrication in Architecture, Art and Design 2016* (pp. 448–453). Springer International Publishing. https://doi.org/10.1007/978-3-319-26378-6_36

Lapham J. (1999). RobotScript: The introduction of a universal robot programming language. *Industrial Robot*, *26*(1), 17–25. https://doi.org/10.1108/01439919910250188

Li Y., Wang K., Xiao Y., & Froyd J. E. (2020). Research and trends in STEM education: a systematic review of journal publications. In *International Journal of STEM Education* (Vol. 7, Issue 1). Springer. https://doi.org/10.1186/s40594-020-00207-6

Lobbezoo A., Qian Y., & Kwon H. J. (2021). Reinforcement learning for pick and place operations in robotics: A survey. In *Robotics* (Vol. 10, Issue 3). MDPI. https://doi.org/10.3390/robotics10030105

M. Castells. (2002). Lessons from the history of the internet. In *The internet galaxy: Reflections on the Internet, Business, and Society* (pp. 20–33).

Maeda J. (2005). CURRENT RESEARCH AND DEVELOPMENT AND APPROACH TO FUTURE AUTOMATED CONSTRUCTION IN JAPAN. *Construction Research Congress*.

Malinetskii G. G., & Sirenko, S. N. (2017). Robotics and education: A new approach. *Herald of the Russian Academy of Sciences*, *87*(6), 527–534. https://doi.org/10.1134/S1019331617060107

Marques A. L., Santos Carrijo R., & Silva De Morais A. (2020). *ROBÓTICA COLABORATIVA: IMPORTÂNCIA E DESAFIOS COLLABORATIVE ROBOTIC: IMPORTANCE AND CHALLENGES*.

Mcmurtrey M. E., Downey J. P., Zeltmann S. M., & Friedman W. H. (2008). Critical Skill Sets of Entry-Level IT Professionals: An Empirical Examination of Perceptions from Field Personnel Critical Skill Sets of Entry-Level IT Professionals. In *Journal of Information Technology Education* (Vol. 7).

Meng Z. (2022). Trajectory Tracking Control Algorithm of Six Degrees of Freedom Industrial Robot. In *Jordan Journal of Mechanical and Industrial Engineering* (Vol. 16, Issue 1).

Navy S. L., Kaya F., Boone B., Brewster C., Calvelage K., Ferdous T., Hood E., Sass L., & Zimmerman M. (2021). "Beyond an acronym, STEM is…": Perceptions of STEM. *School Science and Mathematics*, *121*(1), 36–45. https://doi.org/10.1111/ssm.12442

Park J., Choi S., Oh J., & Eo J. (2019). ENGINE NET TORQUE COMPENSATION THROUGH DRIVELINE TORQUE ESTIMATION IN A PARALLEL HYBRID VEHICLE. *International Journal of Automotive Technology*, *20*(3), 619–627. https://doi.org/10.1007/s12239–019–0059–y

Pérez L., Rodríguez Í., Rodríguez N., Usamentiaga R., & García D. F. (2016). Robot guidance using machine vision techniques in industrial environments: A comparative review. *Sensors*, *16*(3). https://doi.org/10.3390/s16030335

Pires J. N. (2003, April 28). Os Desafios da Robótica Industrial Da interdisciplinaridade às vantagens da cooperação entre empresas e universidades. *Público*.

Rahman S. M. M. (2021). Assessing and benchmarking learning outcomes of robotics-enabled stem education. *Education Sciences*, *11*(2), 1–25. https://doi.org/10.3390/educsci11020084

Robotiq. (2020). *2F-85 Gripper*. https://robotiq.com/products/2f85-140-adaptive-robot-gripper?ref=nav_product_new_button

Robotiq. (2020). *Wrist Camera*. https://robotiq.com/products/wrist-camera?ref=nav_product_new_button

Santos F. C., & Júnior G. A. S. (2020). A DIMENSÃO DA ROBÓTICA EDUCACIONAL COMO. *Dialogia*, 50–65.

Sergeyev A., & Alaraje N. (2010). Promoting Robotics Education: Curriculum and State-of-the-Art Robotics Laboratory Development. *The Technology Interface Journal*, *10*(3).

Sergeyev A., Alaraje N., Kuhl S., Hooker J., Druschke V., Kinney M., & Highum M. (2018). *Revamping Robotics Education to Meet 21st Century Workforce Needs-Years 1-2 Progress Reports*.

Sergeyev A., Alaraje N., Kuhl S., Meyer M., Kinney M., & Highum M. (2015). Innovative Curriculum Model Development in Robotics Education to Meet 21st Century Workforce Needs. In *ASEE Zone III Conference (Gulf Southwest-Midwest-North Midwest Sections)*.

Sergeyev A., Alaraje N., Parmar S., Kuhl S., Druschke V., & Hooker J. (2017). Promoting industrial robotics education by curriculum, robotic simulation software, and advanced robotic workcell development and implementation. *11th Annual IEEE International Systems Conference, SysCon 2017 - Proceedings*. https://doi.org/10.1109/SYSCON.2017.7934754

Shepherd S., & Buchstab A. (2014). KUKA Robots On-Site. In *Robotic Fabrication in Architecture, Art and Design 2014* (pp. 373–380). Springer International Publishing. https://doi.org/10.1007/978-3-319-04663-1_26

Shmatko N., & Volkova G. (2020). Bridging the Skill Gap in Robotics: Global and National Environment. *SAGE Open*, *10*(3). https://doi.org/10.1177/2158244020958736

Soares R., & Lucato A. V. R. (2021). ROBÓTICA COLABORATIVA NA INDÚSTRIA 4.0, SUA IMPORTÂNCIA E DESAFIO. *Revista Interface Tecnológica*, *18*(2), 747–759. https://doi.org/10.31510/infa.v18i2.1298

Szykiedans K., Credo W., & Osiński D. (2017). Selected Mechanical Properties of PETG 3-D Prints. *Procedia Engineering*, *177*, 455–461. https://doi.org/10.1016/j.proeng.2017.02.245

Tijdens K., Beblavy M., & Thum-Thysen A. (2018). Skill mismatch comparing educational requirements vs attainments by occupation. *International Journal of Manpower*. https://www.emerald.com/insight/content/doi/10.1108/IJM-10-2018-0328/full/html

Universal Robots. (2021, May). *UR3e*. https://www.universal-robots.com/br/produtos/ur3/

Verl A., Xu W., & Institute of Electrical and Electronics Engineers. (2018). *Design and Experiment of Robotic Belt GrindingSystem with Constant Grinding Force*.

Yang L., Liu Y., & Peng J. (2020). Advances techniques of the structured light sensing in intelligent welding robots: a review. In *International Journal of Advanced Manufacturing Technology* (Vol. 110, Issues 3–4, pp. 1027–1046). Springer. https://doi.org/10.1007/s00170-020-05524-2

Zamalloa I., Kojcev R., Hernández A., Muguruza I., Usategui L., Bilbao A., & Mayoral V. (2017). *Dissecting Robotics - historical overview and future perspectives*. http://arxiv.org/abs/1704.08617

Anexo 1

Especificações técnicas do robô UR3e (Universal Robots, 2021).

USA 10/2015

UNIVERSAL ROBOTS

UR3 Technical specifications

Item no. 110103

6-axis robot arm with a working radius of 500 mm / 19.7 in

Weight:	11 kg / 24.3 lbs
Payload:	3 kg / 6.6 lbs
Reach:	500 mm / 19.7 in
Joint ranges:	+/- 360° Infinite rotation on end joint
Speed:	All wrist joints: 360 degrees/sec. Other joints: 180 degrees/sec. Tool: Typical 1 m/s. / 39.4 in/s.
Repeatability:	+/- 0.1 mm / +/- 0.0039 in (4 mils)
Footprint:	Ø128 mm / 5.0 in
Degrees of freedom:	6 rotating joints
Control box size (WxHxD):	475 mm x 423 mm x 268 mm / 18.7 x 16.7 x 10.6 in
I/O ports:	Digital in: Controlbox 16, Tool conn. 2 Digital out: Controlbox 16, Tool conn. 2 Analog in: Controlbox 2, Tool conn. 2 Analog out: Controlbox 2, Tool conn. -
I/O power supply:	24 V 2A in control box and 12 V/24 V 600 mA in tool
Communication:	TCP/IP 100 Mbit: IEEE 802.3u, 100BASE-TX Ethernet socket & Modbus TCP
Programming:	Polyscope graphical user interface on 12 inch touchscreen with mounting
Noise:	Comparatively noiseless
IP classification:	IP64
Power consumption:	Approx. 100 watts using a typical program
Collaboration operation:	15 advanced adjustable safety functions
Materials:	Aluminum, PP plastic
Temperature:	The robot can work in a temperature range of 0-50°C*
Power supply:	100-240 VAC, 50-60 Hz
Cabling:	Cable between robot and control box (6 m / 236 in) Cable between touch screen and control box (4.5 m / 177 in)

*) At high continuous joint speed, ambient temperature is reduced.

Universal Robots USA, Inc.
11 Technology Drive
East Setauket, New York 11733
USA
+1 631 610-9664

ur.na@universal-robots.com

Universal Robots A/S
Energivej 25
DK-5260 Odense S
Denmark
+45 89 93 89 89

www.universal-robots.com
sales@universal-robots.com

Anexo 2

Departamento de
Engenharia Eletromecânica

Protocolos para atividades laboratoriais

Sugestões de atividades práticas no âmbito da resolução de um labirinto

Robótica Industrial

Proposta por: Luís Nunes

Professor: Pedro Dinis Gaspar

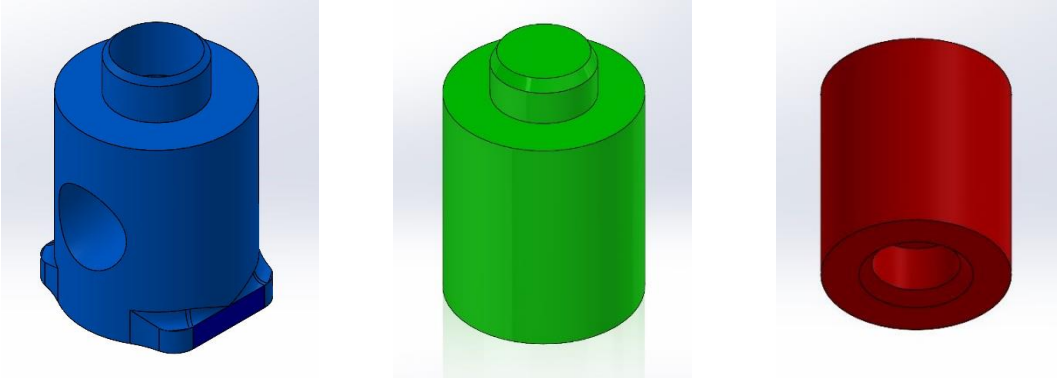
Departamento de Engenharia Eletromecânica

Universidade da Beira Interior

Covilhã e UBI, junho de 2023

Este guia tem como objetivo apoiar o desenvolvimento de trabalhos laboratoriais que pretendam desenvolver o conhecimento sobre a utilização do pendente do robô UR3e, demonstrando a rapidez e eficiência da programação. Através de exercícios práticos e estimulantes relacionados com a resolução de um labirinto em que serão aplicadas operações comuns de robôs industriais.

Para isso, o robô será programado para montar um desafio aleatório, criado previamente. De seguida, um programa que faz uso da visão computacional irá controlar o robô para resolver este problema. Após a montagem da estrutura, será possível colocar uma esfera na área da peça verde, que deslizará pelo caminho mais curto em torno das peças vermelhas até chegar à peça azul. A primeira parte do trabalho consiste na definição do desafio, usando seis peças cilíndricas, ilustradas na Figura 1.



(a) Peça inicial (b) Peça final (c) Obstáculos

Figura 115 Peças usadas para definir o desafio.

Utilizando as ferramentas de manipulação do robô, estas peças serão posicionadas de acordo com o esquema da Figura 2, que pode ser alterado para que diferentes grupos façam o mesmo trabalho com configurações de montagem diferentes.

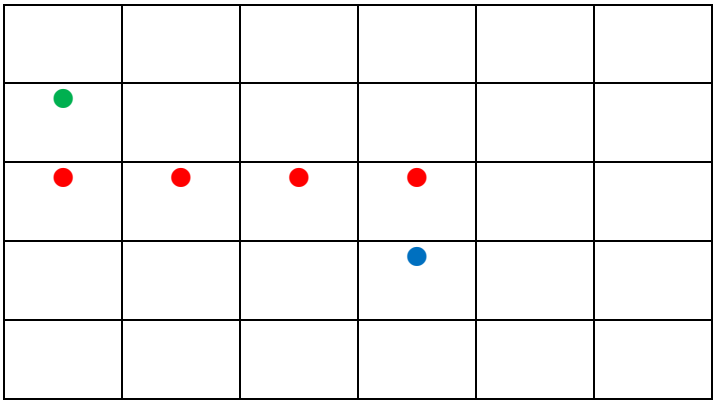


Figura 2. Esquema de montagem do desafio.

As peças que definem o desafio encaixam na base representada na Figura 3 (a), com um resultado semelhante ao apresentado na Figura 3 (b). Com o desafio montado, o objetivo passa a ser a criação de uma pista que começa na peça verde, termina na peça azul, e permite a navegação ao redor dos obstáculos vermelhos, pelo caminho mais curto, como exemplificado na Figura 2 (c).

(a) Base (b) Definição do desafio (c) Montagem concluída

Figura 3. Estágios da montagem.

Para criar esta pista, será executado um programa para navegar pelo labirinto com base no *setup* que foi montado nesta atividade.

O objetivo é apresentar esta tarefa em diferentes configurações, permitindo ajustar o nível de dificuldade às ferramentas que se pretendem desenvolver, especificamente:

Atividade 1: Manipulação simples.

Objetivo:

Montar as 6 peças que definem o desafio, de uma posição pré-determinada à posição indicada na Figura 2.

Procedimento:

- As peças estão armazenadas em posições específicas;
- Programar o manipulador para recolher as peças de um local pré-definido e movê-las para a posição indicada na Figura 2;
- Deslocar o manipulador a uma posição que permita tirar uma fotografia, que posteriormente será guardada numa *pen drive*;
- O utilizador vai pegar na foto guardada no dispositivo de armazenamento e vai-lhe dar um nome à sua escolha, de preferência um nome fácil.
- Transferir de sítio para o mesmo sítio, mesma pasta, do programa principal de montagem, para esta parte de tirar a foto e guardá-la já existe um programa feito que vai ser fornecido;
- Numa fase seguinte vai à linha de código, linha 18, do programa de montagem onde vai abrir a foto que o manipulador tirou para que este proceda a montagem da pista.
- Em seguida, correr o programa e verificar se manipulador faz a montagem da pista adequadamente.

Atividade 2: Inputs e outputs.

Objetivo:

Montar as 6 peças que definem o desafio. As peças serão colocadas individualmente por uma ordem específica no tapete transportador, e transportadas até ao ponto de recolha do manipulador, que as colocará na posição indicada na Figura 2.

Procedimento:

- As peças vão ser colocadas por uma ordem específica no tapete rolante (1º a peça inicial; 2º a peça final; 3º- as 4 peças vermelhas;);
- O manipulador é programado para aguardar até que o sensor de infravermelhos dê a indicação que a peça se encontra no local de recolha.
- Programar o manipulador para recolher as peças no ponto de recolha do tapete e movê-las para a posição indicada na Figura 2;
- Deslocar o manipulador a uma posição que permita tirar uma fotografia, que posteriormente será guardada numa *pen drive*;
- O utilizador vai pegar na foto guardada no dispositivo de armazenamento e vai lhe dar um nome a sua escolha, de preferência um nome fácil.
- Transferir de sítio para o mesmo sítio, mesma pasta, do programa principal de montagem, para esta parte de tirar a foto e guardá-la já existe um programa feito que vai ser fornecido;
- Numa fase seguinte vai a linha de código, linha 18, do programa de montagem onde vai abrir a foto que o manipulador tirou para que este proceda a montagem da pista.
- Em seguida, verificar se esta tudo pronto para dar *run* no programa, verificar se o computador está devidamente ligado ao robô.

Atividade 3: Visão.

Objetivo:

Montar as 6 peças que definem o desafio. As peças serão colocadas individualmente por uma ordem não especificada no tapete transportador, e transportadas até ao ponto de recolha do manipulador, que as deverá reconhecer, utilizando as ferramentas de visão, e colocar na posição indicada na Figura 2.

Procedimento:

- As peças vão ser colocadas aleatoriamente no tapete rolante;
- O manipulador é programado para aguardar até que o sensor de infravermelhos dê a indicação que a peça se encontra no local de recolha.
- Programar o manipulador para recolher as peças no ponto de recolha do tapete e movê-las para a posição indicada na Figura 2;
- Deslocar o manipulador a uma posição que permita tirar uma fotografia, que posteriormente será guardada numa *pen drive*;
- O utilizador vai pegar na foto guardada no dispositivo de armazenamento e vai lhe dar um nome a sua escolha, de preferência um nome fácil.
- Transfere a foto de sítio para o mesmo sítio, mesma pasta, do programa principal de montagem, para esta parte de tirar a foto e guardá-la já existe um programa feito que vai ser fornecido;
- Numa fase seguinte vai a linha de código, linha 18, do programa de montagem onde vai abrir a foto que o manipulador tirou para que este proceda a montagem da pista.
- Em seguida, verificar se esta tudo pronto para dar *run* no programa, verificar se o computador está devidamente ligado ao robô.

Atividade 4: Montagem da pista sem processamento de imagem.

Objetivo:

Montar a pista para um desafio previamente definido e montado manualmente. As várias peças que compõem o labirinto serão montadas pelo manipulador, que deverá ser programado para esta tarefa de manipulação que exige uma maior complexidade, dada a importância da precisão, e orientação das peças.

Procedimento:

- As peças que definem o desafio são colocadas à mão na base, de acordo com o esquema da Figura 2;
- Definir o caminho mais curto que pode ser feito, obedecendo às restrições do desafio;
- Programar o manipulador para montar a pista, tendo em atenção a orientação das pistas, e a ordem de montagem que permite diminuir o risco de colisões;

Iniciar e programar o UR3e:

1. Ligar o robô através do botão do pendente indicado na Figura 4, e verificar se o robô já esta operacional, luz verde no canto inferior esquerdo do pendente.

Figura 4. Pendente UR3e.

2. Ativar a garra, no menu instalação na barra superior esquerda. Assim que a garra esta ativada podemos passar a criação do programa.

3. Na barra superior, ao centro, criar um programa.

4. Para criar um programa, escolher as funções disponíveis no menu programa na barra superior, do lado esquerdo. São as funções principais:

    a. Marcar posições:

        i. **waypoint**: marcar uma posição, posição para largar a peça, posição para buscar peça ou até mesmo posição intermedia. Para guardar uma posição, movemos o robô para a

posição que nos queremos, usando a mão, carregando no botão atrás do pendente, ou então usando os controlos do pendente.

b. Movimento do robô:

   i. **movej**: Comando utilizado para mover o robô para uma posição pré-definida no espaço das juntas. Especifica as posições das articulações do robô.

   ii. **movel**: Comando para mover o robô para uma posição pré-definida no espaço cartesiano. Especifica as coordenadas XYZ e a orientação do efetuador.

   iii. **movep**: Comando para mover o robô em uma trajetória interpolada no espaço das juntas, permitindo um movimento suave e contínuo.

c. Controlo de velocidade:

   i. **speedj**: Comando para definir a velocidade angular máxima das articulações do robô durante o movimento.

   ii. **speedl**: Comando para definir a velocidade linear máxima do robô durante o movimento.

d. Controlo de fluxo:

   i. **if**: Comando condicional utilizado para executar determinadas ações somente se uma condição for verdadeira.

   ii. **while**: Comando de ciclo que repete um bloco de código enquanto uma condição for verdadeira.

   iii. **wait**: Comando utilizado para aguardar até que uma determinada condição seja cumprida antes de continuar a execução do programa.

e. Controlar a garra:

   i. **opengripper**: abrir a garra.

   ii. **closegripper**: fechar a garra

   Caso seja necessário usar o tapete, temos de ter atenção a vários fatores:

f. Aos sensores, que quando a peça chegar ao final do tapete, e ativar o sensor, tem de se ter cuidado em programar o tapete a parar, *"set_convey_OFF=OFF",* ou seja, sempre que a peça chega ao final do tapete e acionar o sensor o tapete para.

g. Assim que a peça são do tapete este tem de começar a andar para que a próxima peça chegue ao fim do tapete, *"set_convey_On=ON"*.

h. Para que a programação do tapete, ir ao menu *UR Caps*, exemplos:

   i. *set_convey_OFF=OFF*: desligar o tapete.

   ii. *set_convey_ON=ON*: ligar o tapete.

   iii. *get_digital_in:* ler o sinal do sensor, da entrada digital.

   iv. *get_analog_in:* ler o sinal do sensor, da entrada analógica.

   v. Vai ser necessário usar condições *"if"*, por exemplo, se o sensor do fim do tapete estiver com sinal negativo o tapete para.

5. Por fim, o robô vai ter de pegar no berlinde e colocá-lo no início da pista, de forma que este percorra a pista.

Tirar a fotografia e gravar na *pen drive*.

a. *camlocate*: função para tirar a foto, a partir da localização dos objetos na base.

b. *Save Image*: Comando para guardar a foto, e necessário selecionar o locar onde vai ser guardada.

c. Após guardar a fotografia na *pen drive*, abrir a *pen drive* no computador e mudar o nome da fotografia para um nome simples, por exemplo o nome do grupo.

d. Abrir o programa "Montagem_Pista" e alterar o nome do ficheiro que se encontrar na linha 18 para o nome dado a fotografia tirada.

e. De seguida, ligar o robô ao computador utilizando um cabo ethernet.

f. Abrir a linha de comandos, no menu inicial do Windows, em seguida escrever na linha de comandos "ipcongif".

g. Assim que aparecer a lista de dados ir buscar o valor do *IPv4 address*, um número com o seguinte formato "xxx . xxx . x . xx".

h. O IP do computador tem de ser igual ao IP no programa, linha 623.

i. Abrir no pendente do robô as definições, ir as conexões, *ethernet* e colocar o número do IPv4 do computador no pendente do robô.

j. Por fim, e necessário permitir que seja um dispositivo externo a controlar o robô. Para isso, no canto superior direito do pendente, consegue-se mudar de remoto para locar, ou vice-versa. É necessário carregar no símbolo do pendente, onde diz local, e mudar para remoto, Figura 5. Assim o robô vai estar preparado para receber ordens externas. Muito importante colocar no pendente modo remoto.

6. Por último, dar *run* no programa.

O trabalho de laboratório requer:

- Apresentação e discussão do trabalho laboratorial.
- Simulação do processo em ambiente sala de aula com o UR3e.
- Entrega de relatório (data-limite de entrega a designar).
- Entrega do programa de controlo do manipulador (data-limite de entrega a designar).