isec
Engenharia

MESTRADO EM ENGENHARIA ELETROTÉCNICA

Manipulação robótica na indústria cerâmica

Autor

Rogério Abel Almeida Torres

Orientador

Professor Coordenador com Agregação

Doutor Nuno Miguel Fonseca Ferreira

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, novembro de 2022

isec
Engenharia

DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA

Manipulação robótica na indústria cerâmica

Relatório de Estágio de Natureza Profissional para a obtenção do grau de Mestre em Engenharia Eletrotécnica

Especialização em Automação e Comunicações em Sistemas Industriais

Autor

Rogério Abel Almeida Torres

Orientador

Professor Coordenador com Agregação

Doutor Nuno Miguel Fonseca Ferreira

Supervisor na empresa CTCV – Centro Tecnológico da Cerâmica e do Vidro

Engenheiro Victor Manuel Mendes Francisco

INSTITUTO POLITÉCNICO DE COIMBRA

INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

Coimbra, novembro de 2022

AGRADECIMENTOS

Começo por agradecer aos meus familiares por todo o apoio e suporte dado ao longo de toda a minha vida. Destaco os meus pais por me terem facultado os meios para concretizar o ingresso ao ensino superior e por estarem sempre dispostos a cometer sacrifícios em prol da minha felicidade e da minha carreira futura.

Ao meu Professor Orientador Doutor Nuno Ferreira que dentre muitas qualidades, destaco a sua disponibilidade, motivação, preocupação e capacidade de incentivo. Agradeço ainda pelas partilhas, oportunidades e experiências dadas pelo mesmo ao longo do meu estágio curricular, que foram sem dúvida muito importantes e enriquecedoras para o tipo de profissional que pretendo ser no futuro.

Ao supervisor da empresa Engenheiro Victor Francisco pelo apoio, compreensão, disponibilidade, confiança e oportunidade da realização do estágio curricular.

Ao Senhor Rui Gouveia por toda a disponibilidade, apoio, trabalho, dedicação, ensinamento e ajuda, sem esquecer toda a boa disposição, companheirismo e paciência presentes em todo o decorrer deste estágio.

Aos meus colegas de curso André e João, que me apoiaram sempre ao longo de todo o mestrado e estágio, demonstrando disponibilidade, companheirismo paciência e motivação, tendo também orientado e demonstrado companheirismo durante todo o decorrer do estágio curricular e da escrita da tese.

Aos meus amigos que sempre me apoiaram, motivaram e se mostraram presentes durante todo o meu percurso académico e de vida.

E por último à Juliana, por ter sido uma inspiração na minha vida, por toda a motivação, paciência, disponibilidade, dedicação, apoio e ajuda desde o primeiro dia. Agradeço também por me ter sempre desafiado a ultrapassar os meus limites, e em me guiar nesta última etapa do meu percurso académico.

A todos os que sempre acreditaram, muito obrigado.

RESUMO

Na área da automação industrial e da robótica, existe a vertente da manipulação, que consiste na utilização de um braço robótico para a orientação e agarre dos objetos desejados por meio de atuadores, tais como vácuo, dedos, entre outros. Alguns objetos, como peças cerâmicas frágeis e de superfície lisa, requerem especial atenção face à força e ao método de agarre exercido sobre estes, onde para tal, existem garras equipadas com sensores de carga capazes de avaliar os valores de carga exercida pelos atuadores da garra sobre o objeto a ser manipulado.

O presente documento encontra-se dividido em duas partes principais, o estado de arte e a demonstração das atividades desenvolvidas, sendo que serão abordados três projetos principais. Estes projetos são a criação de uma réplica digital de um processo de encaixotamento de azulejos, o desenvolvimento de uma garra ventosa rotativa e de uma garra do tipo impacto de três dedos com acionamento independente e capacidade sensorial de carga para aplicar em robôs manipuladores.

Os principais objetivos da realização do estágio foram adquirir conhecimentos gerais na área da automação industrial e robótica referentes à indústria 4.0, e a obtenção de experiência do mercado de trabalho. Por sua vez, estes objetivos foram realizados através da aquisição de vários conhecimentos proporcionados pela experiência de trabalhadores e engenheiros da empresa onde ocorreu o estágio.

Palavras-Chave: Automação, Robótica, Garras, Manipuladores, Indústria 4.0

ABSTRACT

In the field of industrial automation and robotics, there exists the concept of robotic manipulation, which consists of using a robotic arm to guide and grasp the desired objects through actuators, such as vacuum, and fingers, among others. Some objects, such as fragile ceramic pieces with a smooth surface, require special attention due to the force and the gripping method exerted on them, where for this, there are grips equipped with load sensors capable of evaluating the values of load exerted by the grip actuators. onto the object to be manipulated.

This document is divided into two main parts, the state of the art and the demonstration of the activities developed, where three main projects will be addressed. These projects are the creation of a digital replica of a tile boxing process, the development of a rotating suction cup, and a three-finger impact-type gripper with independent drive and sensory loading capability.

The main objectives of the internship were to acquire general knowledge in the area of industrial automation and robotics related to industry 4.0 and to obtain experience in the job market. Nevertheless, these objectives were achieved through the acquisition of various knowledge provided by the experience of workers and engineers from the company where the internship took place.

Keywords: Automation, Robotics, Grippers, Manipulators, Industry 4.0

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

SIMBOLOGIA E ABREVIATURAS

3D	Três Dimensões
A.C.	Antes de Cristo
ABB	ASEA Brown Boveri
ABS	Acrilonitrila Butadieno Estireno
AC	Corrente Alternada
AGV	Automated Guided Vehicle
CAD	Computer-Aided Design
CC	Corrente Contínua
CEN	Comité Europeu de Normalização
cm	Centímetro
CNC	Computer Numerical Control
COM	Comum
CTCV	Centro Tecnológico da Cerâmica e do Vidro
DC	Corrente Contínua
FDM	Fused Deposition Modeling
g	Grama
HMI	Human Machine Interface
I&D	Investigação e Desenvolvimento
I/O	Input/Output
I2C	Inter-Integrated Circuit
IoT	Internet of Things
IPAC	Instituto Português de Acreditação
IPQ	Instituto Português da Qualidade
IRB	Industrial Robot
ISEC	Instituto Superior de Engenharia de Coimbra
ISO	International Organization for Standardization
kg	Quilograma

kgf	Quilograma Força
LCD	Liquid Cristal Display
MIT	Massachusetts Institute of Technology
mm	Milímetros
ms	Milissegundos
N	Newtons
NC	Normalmente fechado
NO	Normalmente aberto
º	Graus
PWM	Pulse Width Modulation
R.U.R.	Rossum's Universal Robots
SA	Sociedade Anônima
SAS	Special Air Service
SCL	Serial Clock
SCT	Sistema Científico e Tecnológico
SDA	Serial Data
SI	Sistema Internacional
SLA	Stereolithography
TCP	Tool Center Point
UV	Ultravioleta
VDC	Volts em Corrente Contínua
VPN	Virtual Private Network

CAPÍTULO 1. - INTRODUÇÃO

O presente documento intitula-se de "Manipulação robótica na indústria cerâmica" e tem como enfoque principal a descrição de estratégias e soluções desenvolvidas em relação à indústria cerâmica. Este surge no âmbito do estágio curricular realizado no Centro Tecnológico da Cerâmica e do Vidro (CTCV), servindo como relatório final do mesmo e com o objetivo da obtenção de grau de mestre no mestrado de Engenharia na Área de Especialização em Automação e Comunicações em Sistemas Industriais lecionado no Instituto Superior de Engenharia de Coimbra (ISEC).

O estágio realizado teve como objetivo principal a aquisição de conhecimentos fulcrais das atividades da empresa associadas à automação industrial e robótica, aprofundando assim os conhecimentos lecionados no mestrado nestas áreas.

Este relatório de estágio apresenta como estrutura quatro capítulos, onde neste primeiro é efetuada uma introdução sobre os objetivos e o local de estágio. O segundo capítulo consiste num estado de arte sobre a evolução industrial e da tecnologia robótica até à atualidade, uma exposição dos tipos de garras existentes e de alguns equipamentos considerados relevantes na temática.

O terceiro capítulo trata-se da descrição das atividades realizadas durante o estágio, tendo estas sido a criação de uma réplica digital de um processo de encaixotamento de azulejos e o desenvolvimento de duas garras robóticas para manipulação cerâmica. Uma destas garras foi desenvolvida para manipular pires de café através de uma ventosa rotativa e a outra garra tem como finalidade agarrar uma chávena de café através da atuação de três dedos independentes, com sensibilidade sensorial.

Por fim, o quarto capítulo expõe as conclusões retiradas da realização deste estágio e alguns pontos a melhorar na realização das tarefas executadas, que podem ser implementadas por pósteros engenheiros.

1.1 - Motivação

A atualidade encontra-se perto de uma quinta revolução industrial, que visa uma indústria aprimorada da atual, a indústria 4.0. Esta aprimora implica o aumento de mão de obra qualificada, de modo a tornar os processos criativos mais “humanizados”.

A principal motivação para a execução deste estágio curricular provem da ambição na toma de conhecimentos do mercado de trabalho associado à robótica e automação industrial, aprofundando em simultâneo os conhecimentos de engenharia eletrotécnica nestas áreas.

Outro fator motivante deste estágio foi o facto de a área da robótica ser proveniente de uma nova iniciativa no local de estágio, o que despertou uma atitude de maior iniciativa e senso crítico.

1.2 - Objetivos

A realização deste estágio teve como principal objetivo a integração no mundo do mercado de trabalho, de modo a obter experiência profissional. A realização deste estágio foi também optada para aprofundar os conhecimentos adquiridos no decorrer do mestrado de eletrotecnia, mais concretamente na área da automação industrial e robótica, através do desenvolvimento de três tarefas correlatadas com robôs manipuladores. A primeira tarefa efetuada trata-se da criação de uma réplica digital de um processo de encaixotamento e paletização de azulejos no software RobotStudio da ABB, processo este pertencente à empresa Aleluia Cerâmicas SA. Esta réplica tem como objetivo substituir a mão de obra presente no processo por robôs manipuladores industriais.

A segunda e terceira tarefa referem-se á criação de garras para aplicar em dois robôs manipuladores pertencentes à empresa de realização de estágio. O primeiro robô trata-se de um YASKAWA, e a garra criada para este robô é uma ventosa rotativa, cuja finalidade consiste na manipulação de um pires de chávena de café para realizar a sua vidragem em mergulho. O segundo robô é um KUKA, e a garra desenvolvida trata-se de uma garra de três dedos de acionamento independente e munidos de sensibilidade sensorial, com o objetivo de manipular uma chávena de café para efetuar o seu polimento junto a uma esponja rotativa.

1.3 - O Centro Tecnológico da Cerâmica e do Vidro

O CTCV foi fundado em 1987, resultado de um comum acordo estabelecido entre Federações Industriais e Organismos Governamentais do Ministério da Indústria e Economia de Portugal, e tem sede no Parque Tecnológico de Coimbra (iParque). Esta instituição foca-se sobretudo no desenvolvimento de habilidades técnicas e tecnológicas de diferentes indústrias (nomeadamente da cerâmica) e é detentora de elevada experiência no que toca ao desenvolvimento de materiais cerâmicos. Posto isto, é importante referir que os seus principais objetivos passam (1) pelo fornecimento de suporte técnico e tecnológico, (2) pela promoção do desenvolvimento e da qualidade de produtos e processos industriais, (3) pela formação de pessoal, (4) pela divulgação de informação científica, técnica e tecnológica, (5) pela realização e promoção de trabalhos de pesquisa e desenvolvimento e, por fim (6) envolver o ambiente construído de forma sustentável. (European Commission-CTCV, 2022)

Atualmente, é uma Entidade Do Sistema Científico e Tecnológico (SCT) certificada pela CERTIF – Associação para a Certificação. Além disso, esta instituição é também detentora de laboratórios acreditados pelo Instituto Português de Acreditação (IPAC), com o objetivo da realização de análises e ensaios. Foi ainda reconhecida pelo Instituto Português da Qualidade (IPQ) como um Organismo de Normalização Setorial que demonstra participação ativa em Comissões Técnicas de Normalização Nacionais, Europeias (CEN) e internacionais (ISO). É ainda membro do centro Habitat, uma plataforma para a construção sustentável. (Quem Somos-CTCV, 2021)

Por fim, importa ainda referir que esta instituição possui vínculo com diversas universidades e centros de investigação, tendo já colaborado em mais de 30 projetos de investigação e desenvolvimento ao longo dos últimos 10 anos pelo que, por consequente, já estabeleceu relação com aproximadamente 100 parceiros nacionais e internacionais. De momento, estão a ser concretizados novos projetos de investigação nos mais recentes edifícios localizados no iParque, nomeadamente no âmbito da área da robótica e automação (ver Figura 1). Foi assim que surgiu a ligação com o Instituto Superior de Engenharia de Coimbra (ISEC), que permitiu o apoio na aquisição de conhecimentos essenciais nesta área tecnológica e na promoção futura de projetos investigação e desenvolvimento (I&D). (CTCV em números, 2022)

Figura 1 - CTCV. Fonte: (www.ctcv.pt/quemsomos.html)

CAPÍTULO 2. - ESTADO DE ARTE

O capítulo 2 tem como principal objetivo a descrição daquilo que foi a evolução da indústria e da tecnologia robótica até ao momento presente. Além disso, discorrer-se-á ainda sobre os tipos de garras existentes para manipuladores robóticos e alguns componentes considerados relevantes para menção.

2.1 - A evolução industrial

No decorrer dos anos e face às constantes inovações tecnológicas, a indústria tem vindo a sofrer mudanças profundas e importantes (Sakurai & Zuchi, 2018). A evolução da tecnologia fez-se ao longo de quatro marcos históricos importantes definidos como revoluções industriais que se traduziram em alterações nos paradigmas de produção. (Pasquini, 2014) (Pereira & Simonetto, 2018)

A primeira revolução industrial (Indústria 1.0) durou cerca de 200 anos e transformou aquilo que era uma economia agrária e artesanal numa dominada pela indústria e máquinas de fabrico a vapor. Em seguida, a introdução da utilização do petróleo e da eletricidade nas máquinas de fabrico permitiu a potencialização da produção em massa e de linha contínua na segunda revolução industrial (Indústria 2.0), que se iniciou 100 anos depois. No final da década de 60, surgiu a terceira revolução industrial (Indústria 3.0), onde a tecnologia da informação foi usada para a programação e automatização da produção. A quarta revolução industrial (Indústria 4.0), contempla a integração dos vários conceitos tecnológicos existentes com a finalidade de criar um novo meio de produção mais rápido e eficiente (Venâncio & Brezinski, 2017). A indústria 4.0 prevê a formação de grandes redes e o fornecimento de produtos e serviços de forma autónoma, através da interligação entre o homem e a máquina (Pereira & Simonetto, 2018). Por outras palavras, esta contempla um mundo onde os indivíduos se movem entre domínio digital e realidade offline através da utilização de tecnologia interligada de modo a permitir a gestão de tarefas. (Xu, David, & Kim, 2018)

Apesar de ainda estar no começo da sua implementação, a quinta revolução industrial (Indústria 5.0) pode ser interpretada como uma complementação da quarta revolução, onde o seu foco está no desenvolvimento de inteligência artificial, sustentabilidade ambiental e económica, e a humanização na personalização dos produtos.

Embora haja a tendência para se considerar cada revolução industrial um evento isolado, a verdade é que o entendimento das mesmas em conjunto permite compreender a sua evolução, e por consequente, o desenvolvimento de mecanismos mais avançados e fiáveis de produção.

2.1.1 - Indústria 1.0

Previamente ao surgimento da indústria, os produtos eram produzidos de forma manual, o que se traduzia em pequenas produções que não eram compatíveis com a evidente necessidade face ao aumento populacional descontrolado. Deste modo, a produção manual já não se constituía como interesse de um regime cada vez mais capitalista, onde a sua essência era a obtenção de lucros. Assim, surgiu na Inglaterra, a indústria 1.0 que depois expandiu-se para outros países como a França, Bélgica, Holanda, Rússia, Alemanha e os Estados Unidos. (Sakurai & Zuchi, 2018)

Este processo de revolução industrial foi uma transição para uma nova era da produção, pela qual ficou marcada por importantes invenções que permitiram sobretudo a evolução do setor produtivo de transportes através da descoberta da utilidade do carvão como fonte de energia. Foi ainda durante esta época que surgiu a locomotiva e a máquina a vapor, tendo sido esta a primeira forma de automação. (Crisanto, 2016)

Além disso, a produção que implicava mão de obra em cada fase do processo de fabrico foi substituída por uma produção que contemplava também a utilização de maquinaria, permitindo uma agilização e aceleração do processo de fabrico, uma vez que as máquinas não necessitavam de parar, por outras palavras, a substituição de trabalhadores por máquinas aumentou significativamente os rendimentos dos processos de fabrico, bem como os tempos de produção (Castanho, 2008). Isto pois cada máquina executava uma etapa do processo de fabrico, originando divisões de trabalho, ou seja, os trabalhadores já não eram responsáveis por todo o processo, mas sim por uma ou várias etapas que o constituíam, de acordo com a Figura 2.

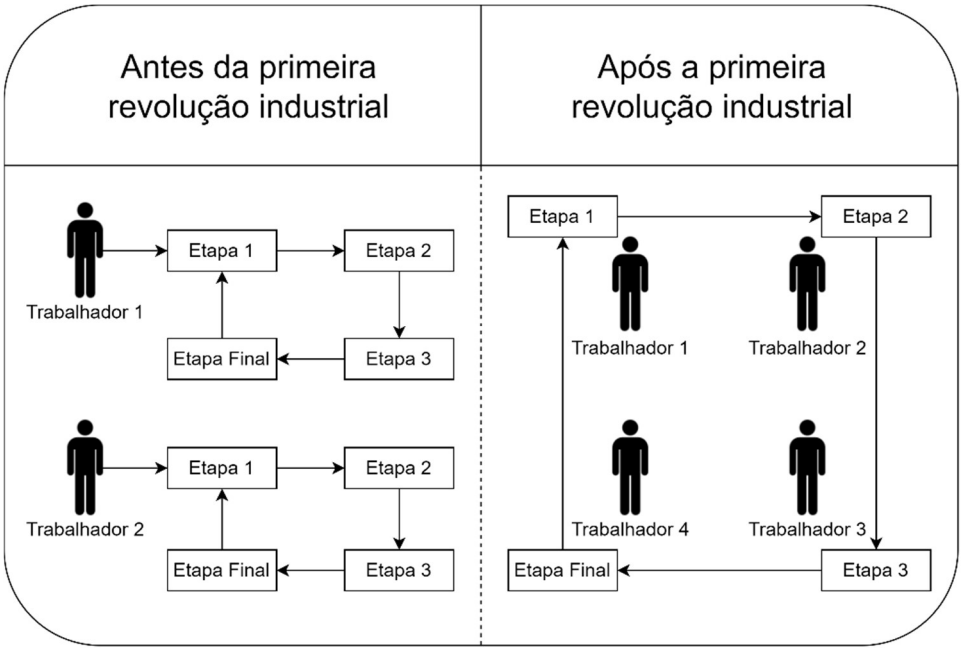


Figura 2 - Distribuição de postos de trabalho antes e depois da revolução industrial. Fonte: (Própria)

Em suma, esta revolução foi caraterizada pelo uso de novas fontes de energia, pela utilização de máquinas movidas a vapor, pelo desenvolvimento de novos meios de comunicação e pela divisão e especialização do trabalho (Crisanto, 2016). Assim, de acordo com (Lima & Neto, 2017), o elemento chave da indústria 1.0 encontra-se nas mudanças tecnológicas e nos avanços que decorreram em três esferas, isto é, (1) na substituição das habilidades humanas por maquinaria, (2) no domínio de uma nova fonte de energia e (3) na melhoria acentuada dos métodos de extração e transformação de matérias primas, que permitiram um aumento de produção e melhoria da qualidade de vida e a instauração de um capitalismo industrial.

2.1.2 - Indústria 2.0

A utilização de novas tecnologias tornou-se um elemento fundamental para o crescimento e modernização ao longo do processo de revolução das indústrias. Dentro deste contexto, o modelo desenvolvido ao longo da primeira revolução industrial foi aprimorado e sofreu mudanças importantes, dando origem à indústria 2.0 que fez frente às novas demandas tecnológicas, movida pelas novas invenções.

Ao longo deste período foi descoberta e introduzida a utilização do petróleo e da eletricidade como meios de combustível para as máquinas industriais, a transformação do ferro em aço, o surgimento e modernização de meios de transporte, o avanço dos meios de comunicação e o desenvolvimento de novas indústrias, tais como a engenharia química (Crisanto, 2016). Além disso, também foram descobertas as primeiras técnicas de automação, posteriormente desenvolvidas por Henry Ford, que se focou sobretudo nos sistemas tecnológicos de produção em massa, através da racionalização da produção capitalista por meio de inovações técnicas, onde por um lado, ocorria a produção em massa, mas por outro, o consumo em massa (Sakurai & Zuchi, 2018). Ford introduziu também a primeira linha de montagem automatizada com o recurso a tapetes transportadores (ou tapetes rolantes).

Outras técnicas de automação importantes desenvolvidas ao longo da indústria 2.0 são conhecidas como automação fixa e recorrem a relés, temporizadores e ligações especificas de cabelagem, conforme a Figura 3. Contudo, este sistema de automação fixa acarretava uma desvantagem que se prendia com a necessidade da paragem da sua utilização para se proceder à sua desmontagem e remontagem quando era necessário efetuar a modificação do sistema, o que se traduzia em problemas ao nível dos custos de produção. Em suma, uma vez montado, o sistema não pode ser reprogramável sem alterações físicas. (Petruzella, 2011)

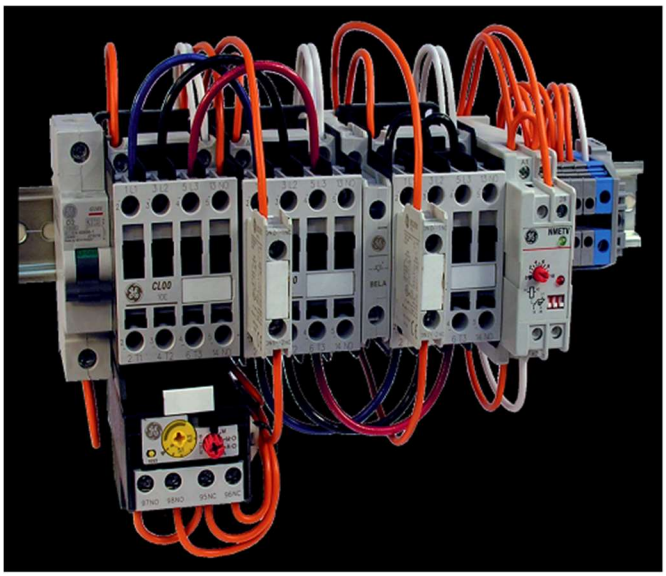


Figura 3 - Lógica com relés. Fonte: (https://tecnologiaelectron.blogspot.com/2013/12/sistemas-de-automatismos-logica.html)

Esta era foi também marcada pelo novo foco em micro inovações, no lugar de macro inovações, isto é, o foco em invenções que melhorassem a produtividade e qualidade do material produzido em vez da criação de invenções inovadoras. Além disso, de acordo com (Boettcher, 2018), todas as inovações supracitadas permitiram lucros cada vez maiores nas indústrias e um maior controlo de gastos, que qualificaram o processo desde a obtenção de matérias-primas até ao consumidor final

No entanto, foi durante este período que também surgiram as primeiras denúncias das condições do trabalho, exigindo mais tempo de descanso, medidas de segurança e também melhores salários.

2.1.3 - Indústria 3.0

A Terceira Revolução Industrial iniciou-se na década de 1950, no pós-segunda guerra mundial. Também conhecida como revolução técnico-científica, esta era foi marcada pelo grande avanço tecnológico nas áreas da robótica, genética, telecomunicações, informática, entre muitas outras, tendo-se também sobressaído nos ramos da indústria metalúrgica e automóvel.

Este avanço tecnológico veio substituir, o que até à data era uma indústria dominada pela tecnologia eletromecânica que recorria à utilização de automação dedicada, repetitiva e não programável, por uma indústria dominada pela tecnologia de base microeletrónica, onde o controlo dos processos industriais é executado através de infraestruturas digitais capazes de programar todo o processo de automação. (Coutinho, 2016)

Esta era foi também marcada pelo surgimento do controlador lógico programável, ou em inglês "*Programmable Logic Controller*" (PLC), que é um computador industrial adaptado para o controlo de processos de fabricação, tais como linhas de montagem, máquinas, dispositivos robóticos ou qualquer outra atividade que exija alta fiabilidade e diagnóstico de falhas de processos. Estes são um exemplo de um sistema de tempo real crítico, querendo com isto dizer que os resultados provenientes da saída do controlador devem ser produzidos consoante os dados obtidos através das variáveis de entrada, dentro de um tempo limitado, caso contrário, o acionamento pretendido é efetuado demasiado tarde, resultando numa operação inválida.

Os PLCs, observados na Figura 4, foram desenvolvidos pela primeira vez na indústria automóvel de modo a providenciar controladores flexíveis, robustos e de fácil programação para substituir os sistemas lógicos de relés desenvolvidos durante a prévia revolução industrial (Wayand, 2020). Desde então, estes têm sido bastante aderidos como controladores de automação de alta fiabilidade, adequados para os ambientes agressivos que a área fabril implica, podendo variar desde pequenos módulos integrados num processador, com algumas dezenas de entradas e saídas lógicas (I/O), até grandes módulos montados em armários com milhares de I/O, e que geralmente estão interligados em rede a outros PLCs. (Tubbs, 2018)

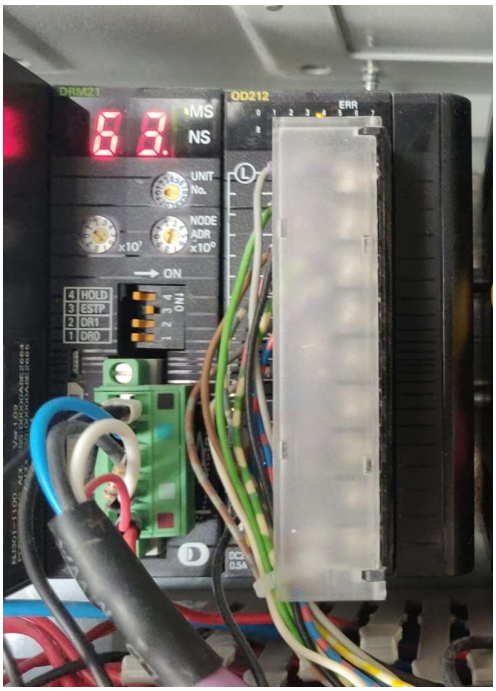


Figura 4 - PLC. Fonte: (Própria)

Em suma, o foco nos avanços tecnológicos levou à criação de máquinas industriais mais rápidas e eficientes, inclusive à criação do primeiro robô industrial, tendo em simultâneo surgido a automação de processos por etapas, proporcionando menores custos de produção e maior rapidez de processo.

2.1.4 - Indústria 4.0

A Quarta Revolução Industrial está a ser vivenciada atualmente, e representa a transição dos avanços tecnológicos desenvolvidos durante a Terceira Revolução para os sistemas no mundo digital, ou seja, a implementação de automação total nos processos industriais.

O fundamento principal desta revolução industrial é que através da interligação da maquinaria, sistemas de produção e restantes equipamentos, seja possível para as empresas a criação de redes munidas de inteligência, capazes de supervisionar e comandar os processos de produção sem a necessidade de interação humana. Existem várias tecnologias que possibilitam esta interligação, sendo uma delas a Internet das coisas, ou *Internet of Things* (IoT).

O IoT é o conceito de interligar todo o tipo de equipamentos (aparelhos de uso quotidiano, meios de transporte, máquinas, entre outros) à internet, possibilitando a interação entre estes e obtendo dados do ambiente em seu redor através de sensores (temperatura, humidade, iluminação, movimento, entre outros), tornando objetos estáticos em elementos dinâmicos numa rede integrada com uma central de processamento inteligente da informação obtida, como se pode observar na Figura 5.

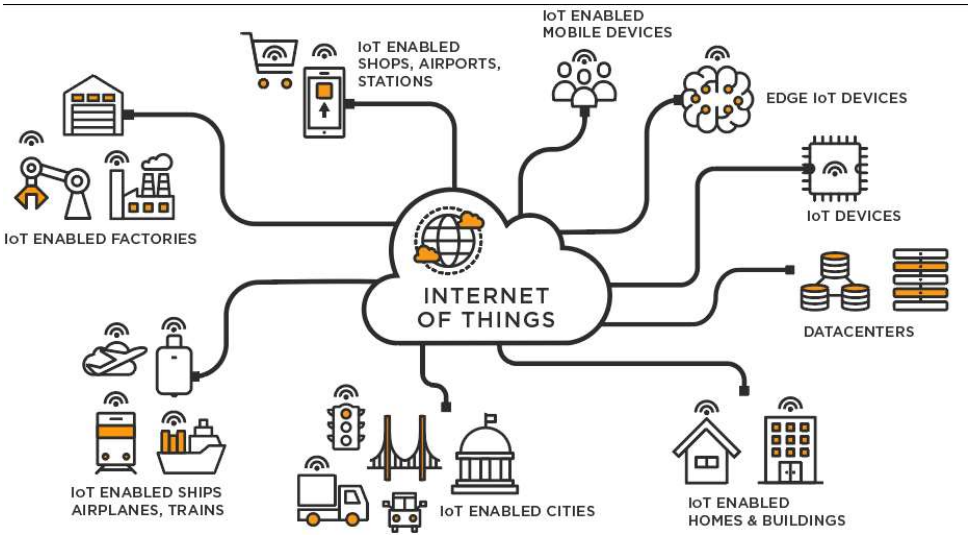


Figura 5 - Internet das coisas. Fonte: (www.tibco.com/pt-br/reference-center/what-is-the-internet-of-things-iot)

Outras tecnologias importantes na indústria 4.0 são as seguintes:

- **Big Data:** Área que estuda métodos de tratamento, análise e obtenção de informação proveniente de conjuntos de dados demasiado grandes para serem tratados por sistemas tradicionais. É através dos computadores de elevada capacidade e redes de comunicação amplas e de baixo custo que é possível armazenar com eficiência e rapidez estas elevadas quantidades de informação,

que após serem analisadas e tratadas em tempo real, facilita a tomada de decisões perante os dados analisados.

- **Computação Cloud e Fog:** A Cloud (nuvem) é um sistema informático onde o armazenamento de dados é efetuado em servidores especializados, onde o acesso e alteração da informação é efetuado através da internet. Desta forma, todos os processos enviam os seus dados para a mesma Cloud, possibilitando a criação de estatísticas globais após a análise desses dados. Na área da indústria, as estatísticas criadas a partir da obtenção de dados recolhidos por sensores físicos permitem a execução de manutenções preventivas de maquinaria. Como o acesso da informação à Cloud é apenas executado via internet, uma vez que este servidor normalmente não está fisicamente presente na unidade fabril, foi criado o conceito de Fog (névoa), que se trata de um conjunto de computadores de fronteira (Edge Computing) que tratam os dados recolhidos localmente, processando-os de forma mais rápida e aliviando a carga dos processadores da Cloud.
- **Réplicas Digitais:** As réplicas digitais de processos fabris são concebidas para substituir os protótipos físicos do chão de fábrica, que por sua vez são mais caros e complexo, obrigando ao procedimento da sua instalação no local. Assim sendo, estas réplicas permitem que sejam testadas diferentes soluções, sem haver a necessidade de se alterar o sistema real da área fabril, possibilitando assim o desenvolvimento de otimizações nos processos, interagir em tempo real, detetar defeitos ou desvios no comportamento das máquinas, entre outros.
- **Cibersegurança:** A criação de servidores com acesso global via internet apresenta ser muito útil na gestão e análise de informação, porém esta facilidade de acesso pode apresentar problemas se segurança. Sem a presença de cibersegurança, hackers podem invadir a rede de informação de uma empresa e expor os dados da mesma a empresas concorrentes ou infetar equipamentos, tais como o HMI (Human-Machine Interface).

Desta forma, comparando a velocidade, alcance e impacto dos novos sistemas provenientes da Quarta Revolução Industrial com as revoluções industriais anteriores, esta evoluiu exponencialmente, em vez de linearmente, apresentando avanços tecnológicos a uma velocidade nunca presenciada.

2.1.5 - Indústria 5.0

A Quinta Revolução Industrial é uma complementação da quarta revolução, onde o foco está no desenvolvimento da inteligência artificial, sustentabilidade e personalização mais humana. Ao contrário da indústria 4.0 que tem como objetivo principal maximizar os lucros sem ter em conta a justiça social ou até mesmo a

sustentabilidade, esta indústria promove a "*utilização da tecnologia para a obtenção de objetivos sociais que vão para além do aumento de lucros e manutenção de empregos*" (Pedro Chaves, Director de Negócio de Manufacturing do SAS Portugal, 2021).

Assim sendo, esta indústria recorre à utilização de robôs colaborativos (Cobot), observado na Figura 6, de modo a reintroduzir o trabalho humano na indústria, possibilitando a criação de customização em massa (Özdemir & Hekim, 2018). A mão de obra passa assim a ser vista como um investimento, e não como um custo extra.

Figura 6 - Robô colaborativo. Fonte: (https://www.assemblymag.com/articles/91862-human-robot-collaboration-comes-of-age)

2.2 - Breve história da robótica

A origem da ideia de humanoides criados artificialmente por seres humanos é, segundo a investigadora *Adrienne Mayor* que publicou o livro "Deuses e Robots: Mitos, Máquinas, e Sonhos Ancestrais da Tecnologia", proveniente de Homero e Hesíodo, poetas gregos que viveram entre 750 e 650 A.C. Como exemplo, a investigadora toma a história de Talos, um homem gigante de bronze criado pelo deus grego da metalurgia Hefesto, que defendia a ilha de Creta, governada pelo rei Minos, de invasores estrangeiros. (Mayor, 2018)

Avançando dois milénios na história, o termo "robot", proveniente do termo checo Robota que significa servidão ou trabalhos forçados, foi divulgado pelo escritor checo Karel Čapek em 1921 para se referir a autômatos artificiais na sua peça de teatro R.U.R (Rossum's Universal Robots). Mais tarde, por volta de 1950, o escritor de ficção científica Isaac Asimov criou as "Três Leis da Robótica" (Santos, 2003-2004), relembrando estas leis:

> "1ª Lei: Um robô não pode ferir um ser humano, ou pela sua passividade deixar que um ser humano seja ferido."
>
> "2ª Lei: Um robô deve obedecer às ordens dadas por um ser humano, excepto se entrarem em conflito com a 1ª lei."
>
> "3ª Lei: Um robô deve proteger a sua própria existência desde que essa proteção não entre em conflito com a 1ª ou 2ª lei." (Ottoni, 2010)

Estas, apesar de rudimentares, devem estar sempre presentes durante a fase de programação do robô para um bom funcionamento do mesmo, mantendo-se a si e a terceiros seguros durante todo o seu curso de trabalho.

No início do século XX, devido à grande necessidade de aumentar a produtividade e de melhorar a qualidade dos produtos, a ideia de construção de robôs começou a ganhar força, tendo sido nesta época que se encontrou as primeiras aplicações para o robô industrial. Até à data desenvolveram-se vários tipos de robôs industriais, aumentando a sua complexidade e capacidade para executar diferentes tarefas com maior liberdade e segurança no passar dos anos.

Em primeiro desenvolveram-se manipuladores industriais que necessitam de barreiras físicas em redor da sua área de trabalho para proporcionar segurança aos trabalhadores, depois criaram-se os robôs móveis capazes de executar tarefas simples como o transporte de peças, também conhecidos por AGV's (Automated Guided Vehicles). Atualmente, a indústria está a investir cada vez mais nos cobots, que não são mais que robôs manipuladores munidos de sensibilidade e inteligência capazes de cooperar com o ser humano no mesmo espaço de trabalho de forma segura e sem a necessidade de barreiras físicas.

Assim sendo, George Charles Devol Jr. é considerado o pai da robótica industrial, tendo criado em 1954 o Unimate, o primeiro robô industrial, presente na Figura 7. Este robô foi implementado em 1961 na linha de produção da General Motors, tendo como funcionalidade manipular pedaços quentes de metal com fim de os implementar nos chassis dos carros. (Automate, s.d.)

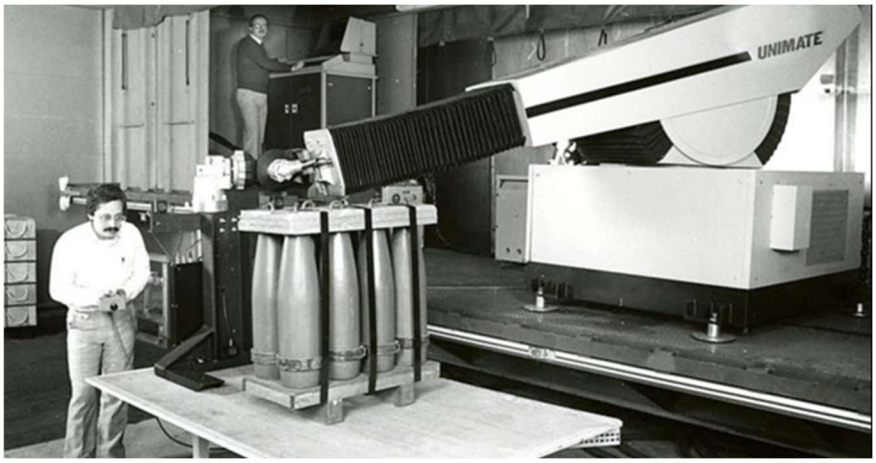


Figura 7 - "Unimate". Fonte: (National Institute of Standards and Technology) (aberto ao público)

Estes robôs industriais são normalmente instalados numa base fixa, parede ou teto e têm um braço (pinça ou ferramenta) para a execução de tarefas. Para segurança, são instaladas barreiras físicas ou de luz, pois estes robôs não apresentam muita "inteligência", tais como sensores de colisão, o que não permite a partilha de espaço de trabalho com humanos, existindo o perigo de este colidir com qualquer obstáculo não previsto na sua programação. Devido ao facto de estes estarem fixos no seu ambiente de trabalho, significa que têm apenas um ambiente imutável, podendo assim executar inúmeras tarefas a altas velocidades e sem a necessidade de interação humana.

Mais tarde, em 1968, foram desenvolvidos no MIT (Massachusetts Institute of Technology) os primeiros robôs móveis, sendo que entre 1966 e 1972, o Standford Research Institute desenvolveu o robô "Shakey", apresentado na Figura 8, que tinha como único comando empurrar um bloco de uma plataforma, recorrendo a planeamento próprio de tarefas. Os robôs móveis, tal como o nome indica, são munidos de movimento, podendo deslocar-se e executar tarefas dentro de uma área de trabalho estipulada. Para segurança, e devido à complexidade de certas tarefas, como por exemplo, que exijam o desvio de obstáculos móveis (pessoas, entre outros), podem existir vários graus de autonomia destes robôs.

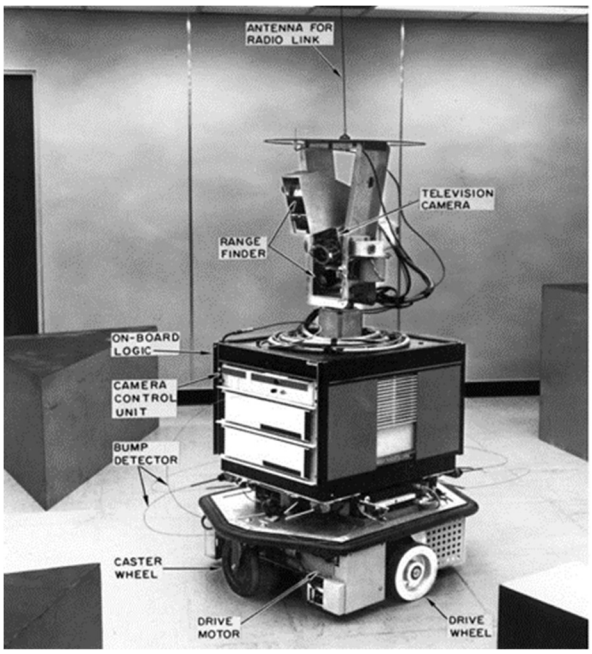


Figura 8 - Robô móvel "Shakey" com inteligência artificial (1970). Fonte: (Nascimento, 2006)

No suceder dos anos foi necessário dar movimento aos robôs industriais, visto que estes apresentam um alcance limitado que, por vezes, obrigam à compra de mais robôs ou de um robô de maiores dimensões que é por vezes mais caro. Criaram-se então linhas com correias de transporte dos robôs, porém estas restringem-se ao deslocamento do robô em linha reta, ocupando também bastante espaço na área de trabalho.

Foi então que surgiram os AGV's (Automated Guided Vehicles) ou sistemas de transporte autónomos que são a junção dos robôs industriais com os robôs móveis. Estes veículos de transporte não necessitam de condutor e são maioritariamente utilizados em armazém e paletização. (Franke, s.d.)

Atualmente, devido à necessidade de cooperação entre robô e trabalhador humano no manuseio de materiais quentes, pesados ou até mesmo para auxílio na soldadura, ou até na área da medicina, em cirurgias prolongadas, houve a necessidade de utilizar robôs com a segurança necessária para evitar qualquer colisão com um ser humano, robôs estes denominados de Cobots, robôs que, em contraste com os robôs industriais que apresentam um risco elevado de colisão e lesão para com os seres humanos, recorrem ao uso de sensores de proximidade e toque, o risco de lesão é praticamente nulo, permitindo trabalho direto com os trabalhadores humanos.

Os Cobots são normalmente programados por áreas de trabalho distintas, como "área de colaboração" onde o robô tem todas as seguranças de colaboração ativadas e move-se a uma velocidade reduzida e a "área livre", onde este move-se à sua velocidade máxima e desativa as seguranças de colaboração, conforme a Figura 9. Recorrendo aos sensores de proximidade, os Cobots podem variar a sua velocidade de trabalho em tempo real, sempre que alguém se aproxima dele, chegando mesmo a parar se necessário.

O mesmo pode-se aplicar aos restantes robôs mencionados acima, porém estes recorrem maioritariamente a barreiras físicas e atuadores de contacto que, por exemplo, enviam um sinal para o robô pausar o seu funcionamento se uma porta da barreira estiver aberta.

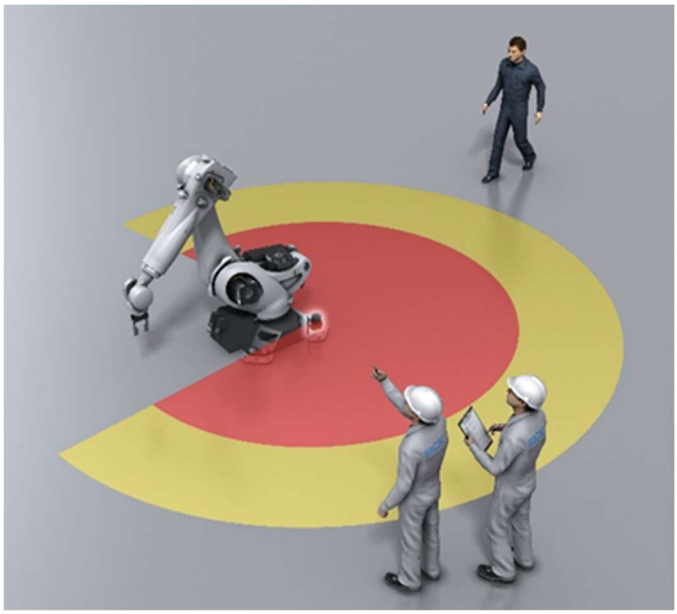


Figura 9 - Área de paragem do Cobot (vermelho) e de velocidade reduzida (amarelo) quando é detetado um obstáculo na respetiva área. Fonte: (CobotTrends.com)

2.3 - Manipuladores industriais

Na área da indústria há cada vez mais adesão á utilização de manipuladores industriais na execução de tarefas perigosas. Os manipuladores industriais são robôs normalmente utilizados para agarrar e manusear cargas pesadas, permitindo aos operadores movimentar objetos de forma rápida, prática e segura, reduzindo o esforço necessário por parte destes, como é possível observar na Figura 10. Assim sendo, contribuem para diminuir o número de acidentes de trabalho no setor industrial, bem como a ocorrência de lesões músculo-esqueléticas, aumentando a produtividade, prevenindo acidentes de trabalho e facilitando a realização de tarefas pesadas. (Benotsmane, Dudás, & Kovács, 2021)

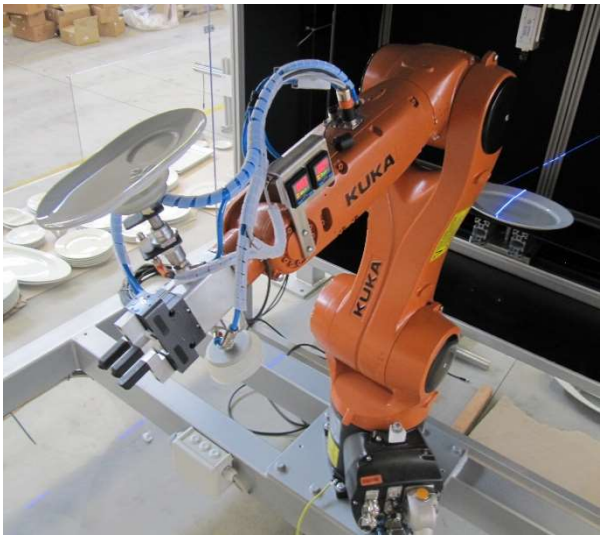


Figura 10 - Robô industrial. Fonte: (Costa Verde - Projeto Inspectware)

O manipulador industrial é constituído, normalmente por seis eixos de rotação, cada um correspondendo a um grau de liberdade, como se pode visualizar na Figura 11. Estes graus de liberdade determinam os movimentos do braço robótico no espaço bidimensional ou tridimensional. Cada junta do robô define um ou dois graus de liberdade. (Ribeiro E. G., 2020), (Ribeiro, Lima, Eckhardt, & Paiva, 2021), (Tuli & Manns, 2019)

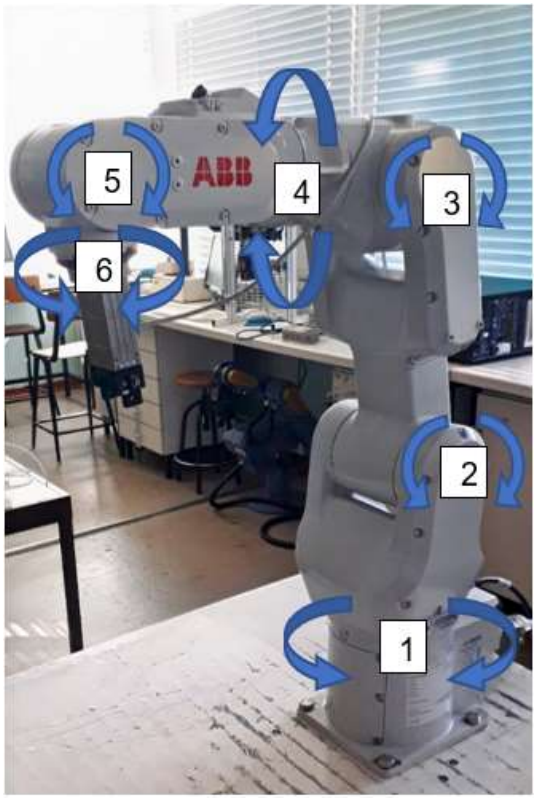


Figura 11 - Eixos de robô ABB IRB 1100-4/0.58 Fonte: (Própria)

2.4 - As diferentes garras dos sistemas robóticos

Uma garra robótica pode ser definida como um subsistema do robô industrial responsável por manter contacto temporário com a peça a ser agarrada (Tai, El-Sayed, Shahriari, Biglarbegian, & Mahmud, 2016), (Shintake, Cacucciolo, Floreano, & Shea, 2018). Garante o manuseamento correto da peça através de mecanismos de geração de pressão entre esta e as terminações da garra. O termo garra pode ser também aplicado em casos onde esta agarre as peças através da sucção (ventosas) ou eletromagnetismo (garra com eletroíman). (Zhang, Xie, Zhou, Wang, & Zhang, 2020)

As garras são os componentes realmente responsáveis por agarrar objetos, que por norma são peças de trabalho que necessitam ser movidas pelo robô para a execução de uma ou várias tarefas. Estas tarefas incluem coleta de peças para um organizador ou palete de transporte (*Pick and Place*), carga e descarga de maquinaria e movimentos precisos de peças junto a ferramentas para a sua aprimora. As peças de trabalho em questão podem ser caixas, chávenas, pratos, matérias-primas, ferramentas, entre outros. (Ribeiro, Lima, Eckhardt, & Paiva, 2021)

Os parâmetros na escolha de um robô manipulador para a realização de uma tarefa podem ser resumidos pela carga do objeto, o alcance da área de trabalho e os seus

graus de liberdade. Enquanto na escolha do atuador final, ou garra, existem inúmeras variantes, tais como a resistência física do objeto, a sua forma, o máximo de deformação ao qual pode ser sujeito, a rugosidade da superfície, entre muitos outros (Ali, Hoi, & Seng, 2011).

As garras podem ser classificadas em quatro tipos, impacto, ingressivo, continuidade e astritivo. (Sousa, 2016)

- **Garras do tipo impacto:** utilizam o movimento dos seus dedos ou pinças de modo a produzir a força necessária para agarrar a peça, assemelhando-se a uma mão humana a agarrar um objeto.
- **Garras do tipo ingressivo:** deformam ou penetram a superfície da peça até uma dada profundidade recorrendo a agulhas, pinos ou ganchos.
- **Garras de continuidade:** implicam contacto direto entre a garra e a peça através de adesão térmica ou química, sendo mais utilizados na manipulação de microcomponentes.
- **Garras do tipo astritivo:** utilizam vácuo ou eletromagnetismo para agarrar o objeto. No caso das garras de vácuo, é necessário assegurar um bom isolamento entre a ventosa e o objeto. (Sousa, 2016)

Dentro dos vários tipos de garras, ou "*grippers*", existe o "*single gripper*", que é uma garra de pressão simples montada no pulso do robô e o "*dual gripper*" que são duas garras de pressão presas ao pulso do robô capazes de segurar dois objetos em simultâneo, com acionamento independente uma da outra (Monkman, 2006), (Samadikhoshkho, Zareinia, & Janabi-Sharifi, 2019).

No lugar de uma garra, pode ser utilizada uma ferramenta para a execução de tarefas. Nestes casos a ferramenta realiza o seu trabalho relativamente a um objeto estático ou em movimento lento, assegurando um processo preciso. Vários exemplos deste tipo de trabalho são soldadura, perfuração, pintura e polimento de peças. Como atuadores finais, existem as seguintes garras. (Qian, QingLong, YongQian, & Lin, 2020)

2.4.1 - Garra mecânica

É um atuador final que usa dedos acionados por um mecanismo com fim de agarrar um objeto. Estes dedos são os apêndices realmente responsáveis por fazer contacto com o objeto a manipular. A utilização de dedos substituíveis permite a sua troca devido a desgaste e à intercambialidade, ou seja, à projeção de diferentes conjuntos de dedos compatíveis com o mesmo mecanismo para acomodar diferentes modelos de peças. Uma garra de dedos substituíveis é denominada por garra intercambiável. (Hu, Wan, & Harada, 2019) Um exemplo de uma garra mecânica encontra-se na Figura 12.

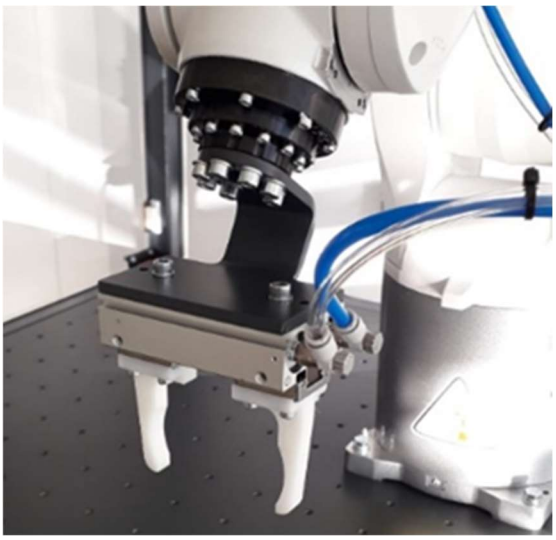


Figura 12 - Garra Mecânica. Fonte: (Própria)

2.4.2 - Dispositivos mecânicos simples

Uma garra pode também ser um dispositivo mecânico simples, como um gancho ou uma concha, onde não existe qualquer acionamento para acomodar um devido material ou peça. Nestes casos é necessário ter em conta o movimento e orientação de entrada da garra para recolher o material, citando a título de exemplo quotidiano, encher uma colher com sopa. (Hu, Wan, & Harada, 2019)

2.4.3 - Garra com feedback sensorial

Garra mecânica com recursos de feedback sensorial nos seus dedos para ajudar a localizar ou determinar a força de pressão correta a ser aplicada na peça de trabalho. Estas garras permitem agarrar peças de trabalho mais frágeis sem correr o risco de as partir ou deformar com o excesso de pressão de agarre de forma automática. Também é possível avaliar se a garra está a segurar corretamente a peça através da comparação da força exercida em cada dedo. Se houver diferença na força exercida de cada dedo significa que a garra não está a agarrar corretamente a peça, podendo isto significar que a peça está descentrada da posição desejada. (Hogreve, Prigge, Köbisch, & Tracht, 2020)

2.4.4 - Garra com vários dedos

São garras mecânicas especialmente projetadas para se assemelhar a uma mão humana. Dependendo do número de juntas por dedo, capacidade sensorial e/ou

complexidade mecânica, estas garras podem ser consideradas “garras universais”, mencionadas mais a frente. Um exemplo deste tipo de garras está presente na Figura 13. (Liu, et al., 2020)

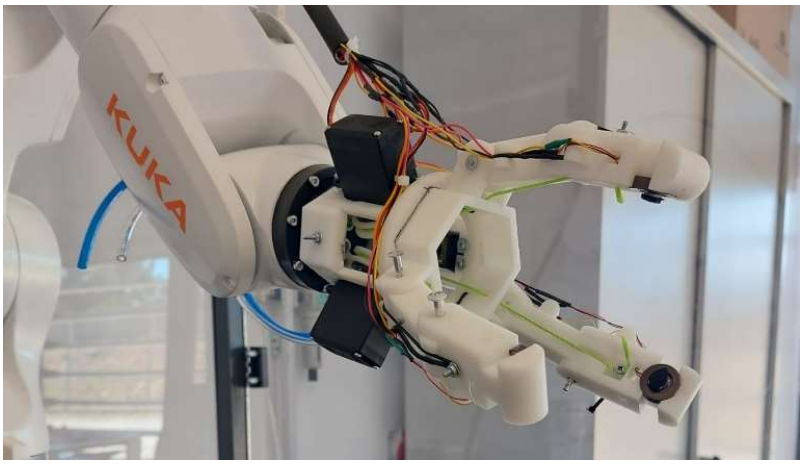


Figura 13 - Garra com vários dedos e feedback sensorial. Fonte: (Própria)

2.4.5 - Garra magnética

As garras magnéticas utilizam o princípio do magnetismo para segurar peças de trabalho metálicas, com a capacidade de carregarem peças de diferentes tamanhos muito rapidamente. Um sistema que recorra ao eletromagnetismo é de controlo mais fácil, porém requer uma alimentação de corrente contínua (CC) e uma unidade de controlador apropriada. Os imãs permanentes apresentam a vantagem de não exigir uma fonte de alimentação CC externa para a sua operação. (Peidró, Tavakoli, Marín, & Reinoso, 2019)

2.4.6 - Garra adesiva

Garras que utilizam uma substância adesiva nas suas extremidades para agarrar peças muito leves como tecidos. As peças a serem manipuladas devem ser seguradas apenas por um dos lados. Novas tecnologias apontam para uma garra que utiliza atuadores de elastômero fluídico e adesivos inspirados no geco. (Modabberifar & Spenko, 2018)

2.4.7 - Garra de vácuo

Recorre ao uso de ventosas para segurar peças com superfícies planas. O vácuo criado entre a ventosa e a superfície da peça cria força de sucção suficiente para

levantar a peça, podendo ser esta uma folha de metal, painéis de plástico, madeira ou pratos cerâmicos.

Podem ser utilizadas uma ou mais ventosas para agarrar a peça. Entre as restantes garras, esta é a que consome menos energia, porém está sujeita a mais acidentes devido a ventosas desalinhadas e a outras fugas que não asseguram uma vedação hermética. (Gabriel, Fahning, & Meiners, 2020) A Figura 14 mostra um exemplo desta garra.

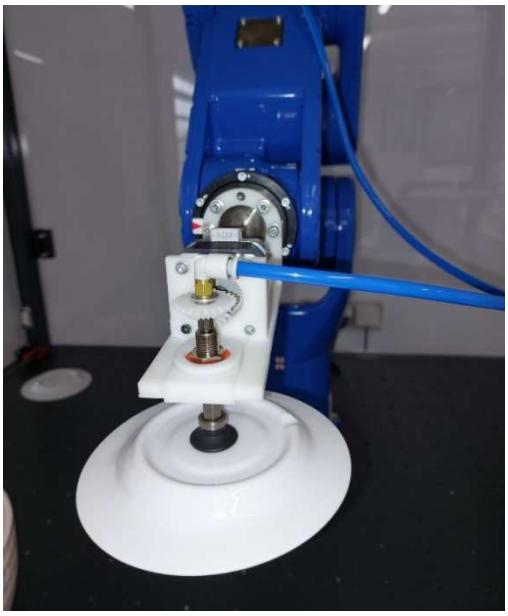


Figura 14 - Ventosa a vácuo. Fonte: (Torres & Ferreira, 2022)

2.4.8 - Garra de expansão

Estas garras, presente na Figura 15, são utilizadas para segurar uma peça oca, como um copo, chávena ou tubo oco pelo seu interior, através da expansão de um material denso de borracha que pressiona contra o interior da mesma, permitindo assim a elaboração do trabalho completo na superfície externa da peça. (Kumar, Yadav, Gupta, & Khatait, 2021)

Figura 15 - Garra de expansão. Fonte: (www.himsale.com/?product_id=209125730_44)

2.4.9 - Garra com vários atuadores finais

É uma garra mecânica com dois ou mais dispositivos de aperto num atuador final normalmente utilizadas para a carga e descarga de peças. No caso das garras com dois atuadores, estas reduzem até metade o tempo de ciclo por peça segurando duas peças de trabalho ao mesmo tempo. Estas garras podem também ter fisometrias diferentes para segurar peças distintas que passem pelo mesmo trajeto de trabalho, ou para diferentes trabalhos em partes diferentes da peça. (Sriskandarajah & Shetty, 2018) Um exemplo de uma garra dupla está representado na Figura 16.

Figura 16 – Garra dupla. Fonte: (Própria)

2.5 - Garras universais

Tarefas de manipulação de objetos consideradas simples no nosso dia a dia podem ser bastante desafiantes para os robôs manipuladores, sendo que para este agarrar um objeto, não basta apenas realizar contacto com o objeto, é necessário também evitar o deslize do objeto durante o seu movimento.

O desenvolvimento de garras universais capazes de agarrar objetos desconhecidos de formas ou quaisquer outras propriedades de superfície variadas requer por norma uma grande complexidade de software e hardware, como por exemplo, as garras baseadas na mão humana. Este tipo de garra inclui um elevado número de juntas controláveis, a necessidade de deteção e controlo da força exercida no objeto para não o deformar e um software complexo para executar os cálculos. (Reddy & Satya Suresh, 2013)

Apesar das garras universais serem bastante flexíveis, estas apresentam bastantes dificuldades face ao manuseio de objetos de grandes dimensões e peso.

2.5.1 - Twister hand

A Twister Hand é outra garra mecânica de dedos acionados por cabos. Apresenta três dedos, cada um com design baseado no origami twisted tower e são movidos em simultâneo por um único servo motor através de cabos, um em cada dedo, que funcionam como tendões, conforme a Figura 17. (Lee, Wang, & Zheng, 2020) Esta tecnologia permite que a garra consiga agarrar objetos frágeis sem os deformar e sem a necessidade de sensibilidade de força.

Figura 17 - Twister Hand. Fonte: (Lee, Wang, & Zheng, 2020)

2.5.2 - Jamming gripper

O jamming gripper é uma garra universal com uma abordagem inovadora. Esta garra em vez de utilizar dedos robóticos, usa uma única massa de material granulado envolto por uma membrana elástica que, quando pressionado contra o objeto a ser manipulado, adapta-se à sua forma, envolvendo-o. (Reddy & Satya Suresh, 2013)

Após a aplicação de vácuo, o material granulado contrai e endurece rapidamente para segurar e prender o objeto. Esta garra abre inúmeras possibilidades para a simplificação dos projetos de garras universais, pois, para além de não necessitar de dedos robóticos, que são de uma construção mecânica complexa, não necessita de feedback sensorial. Para largar o objeto, injeta-se pressão na massa e ela expande, voltando ao seu estado inicial. (Miettinen, Frilund, Vuorinen, Kuosmanen, & Kiviluoma, 2019) A Figura 18 demonstra o funcionamento desta garra no agarre de um objeto.

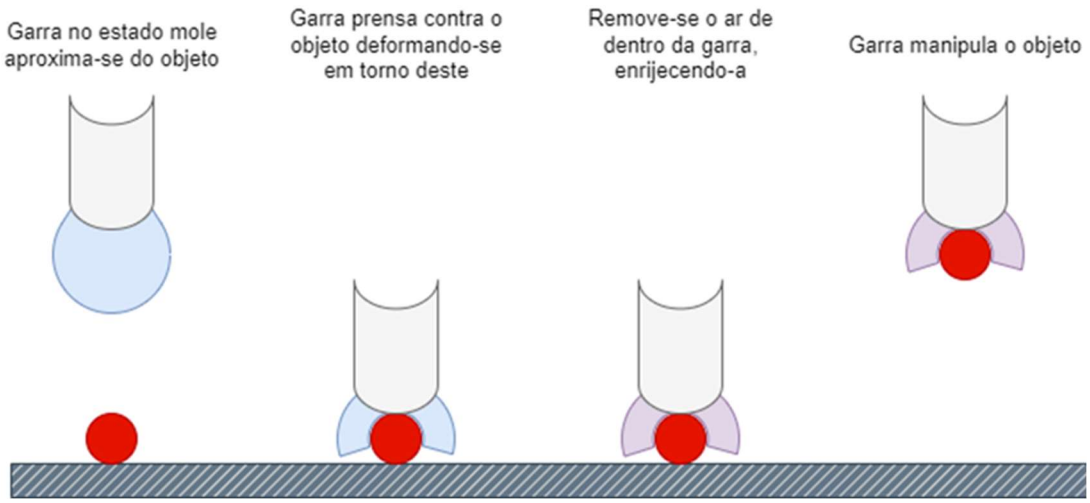


Figura 18 - Método de agarre do Jamming gripper. Fonte: (Própria)

2.6 - Componentes extra

Este subcapítulo aborda alguns componentes inseridos na área da robótica que revelaram ser fundamentais na execução de atividades no decorrer do estágio curricular. Importa ainda referir que também é feita alusão a alguns componentes que não foram utilizados, mas que de algum modo, possuíram caráter imprescindível para a compreensão e aprendizagem.

2.6.1 - Relés

O relé é um interruptor de acionamento elétrico que possibilita o controlo de um circuito com uma determinada potência através de um outro circuito com potência diferente, de forma independente e eletricamente isolados entre si (ver Figura 19).

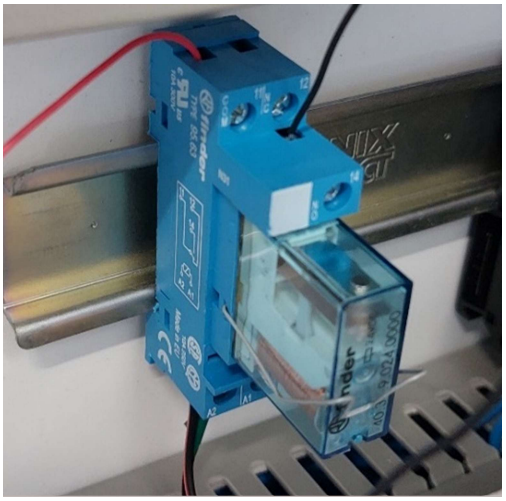


Figura 19 - Relé eletromagnético. Fonte: (Própria)

Por norma, é um circuito de baixa potência que aciona o relé, controlando assim um circuito de alta potência, porém o contrário também pode acontecer. O relé utilizado para interligar a comunicação do robô (24VDC) com a placa Arduíno (12VDC) foi o relé eletromagnético. Este relé, observado na Figura 20 é constituído por um lado primário e um lado secundário.

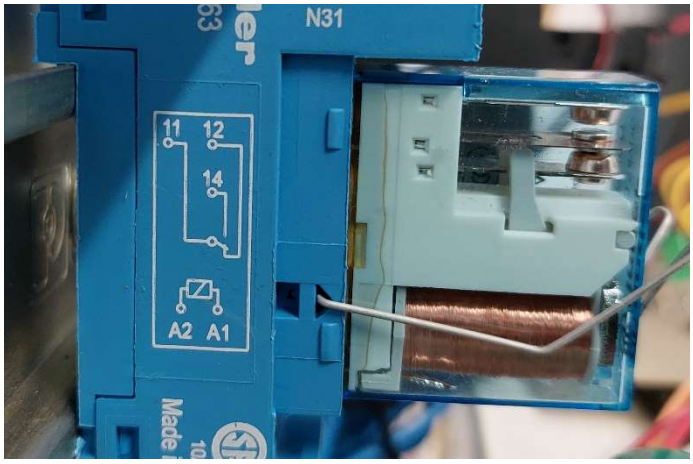


Figura 20 - Esquema de relé eletromagnético. Fonte: (Própria)

O lado primário é responsável por acionar o relé através da corrente que passa na bobine, criando assim um campo eletromagnético que atrai uma alavanca responsável pela mudança do estado dos contactos do lado secundário.

Na Figura 20 o lado primário é representado pelos pinos A1 e A2, podendo estes serem ligados em qualquer sentido. Por norma, o lado secundário destes relés apresenta três pinos, sendo estes o pino comum (COM), o pino normalmente aberto (NO) e o pino normalmente fechado (NC). Na Figura 20 este lado é representado pelos pinos 11, 12 e 14, sendo o pino 11 o COM, o pino 12 o NC e o pino 14 o NO.

O pino COM está sempre em contacto com um outro pino, sendo que quando não passa corrente na bobine o pino COM fica em contacto com o NC e quando passa corrente na bobine o pino COM entra em contacto com o NO, sendo este o parâmetro decisivo na ligação dos pinos secundários do relé.

Existem vários tipos de relés, como por exemplo o relé de "*latching*" que apenas necessita de um impulso energético para mudar o seu estado ou relés puramente eletrónicos que utilizam um led que aciona um foto-transístor, porém o seu princípio é o mesmo. Existem também relés especiais, como os relés temporizadores e os relés de segurança.

2.6.2 - Sensores infravermelhos ON/OFF

Os sensores infravermelhos ON/OFF são normalmente utilizados para detetar objetos a uma distância calibrada. Estes sensores, como mostra a Figura 21, irradiam um feixe

de luz infravermelha, que reflete num objeto, retornando para um recetor. Quando o recetor identifica a luz infravermelha, envia um sinal "*ON*" ou 1 pela saída de "*DATA*". Se o recetor não identificar o feixe de luz infravermelha, envia um sinal "*OFF*" ou 0 pela saída de "*DATA*".

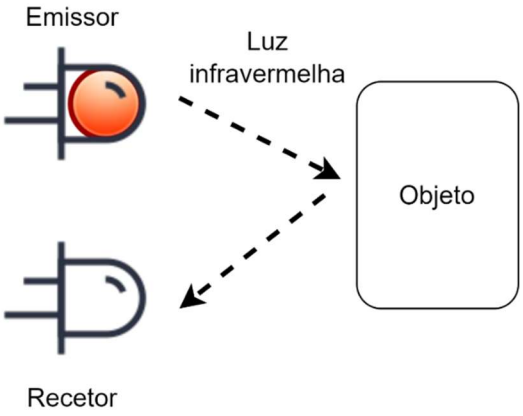


Figura 21 - Funcionamento de sensor infravermelho. Fonte: (Própria)

O sensor utilizado na deteção de proximidade de objeto foi o sensor E18-D80NK. Este sensor utiliza um parafuso regulador para calibrar a distância de deteção, e quando o sensor deteta um objeto, acende também um led, como é demonstrado na Figura 22.

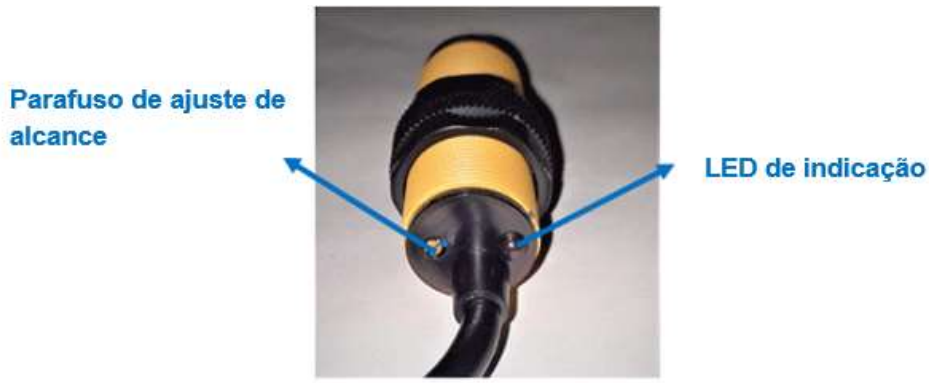


Figura 22 - Sensor E18-D80KN com LED e parafuso de ajuste. Fonte: (https://circuitdigest.com/microcontroller-projects/interfacing-e18d80nk-ir-proximity-sensor-with-arduino)

2.6.3 - Sensores de carga

Na manipulação de materiais frágeis como os cerâmicos, é por vezes importante atribuir sensibilidade às garras do manipulador. Para tal, os sensores de carga são as ferramentas ideais para medir a força exercida pela garra sobre uma superfície.

Estes sensores são compostos por uma ponte de extensómetros elétricos, que se comportam como uma "mola" quando aplicado pressão sobre eles, como mostra a Figura 23. Os extensómetros são condutores elétricos e alteram a sua resistência

elétrica consoante a sua deformação elástica, deformação esta que permite o retorno do material à sua forma inicial.

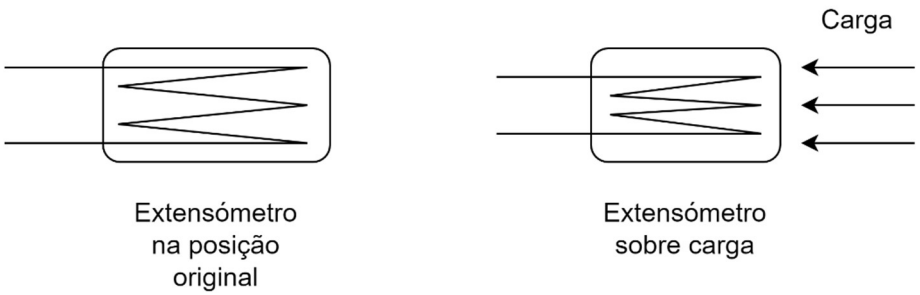


Figura 23 - Funcionamento do extensómetro. Fonte: (Própria)

Quando comprimido sobre a atuação de uma força, este diminui a sua resistência elétrica, e quando descomprime, aumenta a sua resistência elétrica. É através da variação da sua resistência elétrica que o sensor calcula a carga exercida sobre si. A resistência do extensómetro está ligada a uma ponte de "*Wheatstone*", presente na Figura 24, que é um esquema elétrico com a capacidade de calcular uma resistência desconhecida. (Silva, Ferreiro, Cirquera, Santos, & Silva, 2014)

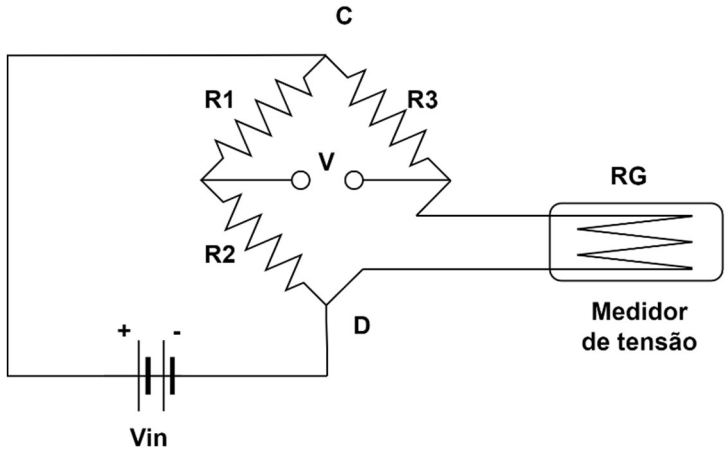


Figura 24 - Ponte de "Wheatstone". Fonte: (Própria)

2.6.4 - Sistemas de criação de vácuo

Na criação de ar comprimido, utilizam-se compressores de ar com fim de o armazenar em cilindros de alta pressão. Como estes cilindros armazenam o ar com pressões superiores à da pressão atmosférica, a atmosfera comporta-se como vácuo e o ar comprimido é expelido quando o circuito de ar é aberto.

Na área da manipulação com garras de ventosas, é necessário criar o efeito oposto, ou seja, é necessário criar uma pressão inferior à da pressão atmosférica para se formar o vácuo que vai agarrar o objeto a manipular. Para tal é necessário implementar sistemas de criação de vácuo, tal como o sistema apresentado na Figura 25.

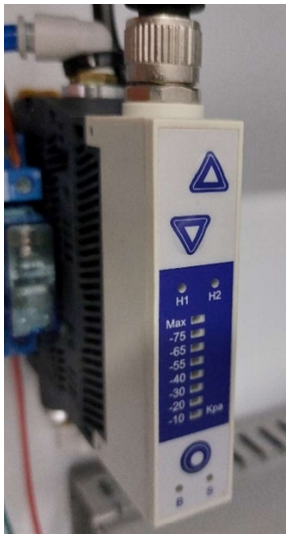


Figura 25 - Gerador de vácuo "Compact Ejector". Fonte: (Própria)

O gerador de vácuo da Figura 25 utiliza a pressão de ar comprimido (pressão positiva) para criar vácuo (pressão negativa) através do efeito de Venturi. O ar comprimido injetado na entrada do aparelho passa por um canal cada vez mais estreito, criando nesse estreitamento uma pressão inferior, porém com maior velocidade que obriga à entrada de ar por um orifício, que é o fornecedor de vácuo da garra. O volume de ar total que sai pelo orifício de saída corresponde à soma do ar comprimido injetado com o ar sugado pelo efeito de Venturi, demonstrado na Figura 26.

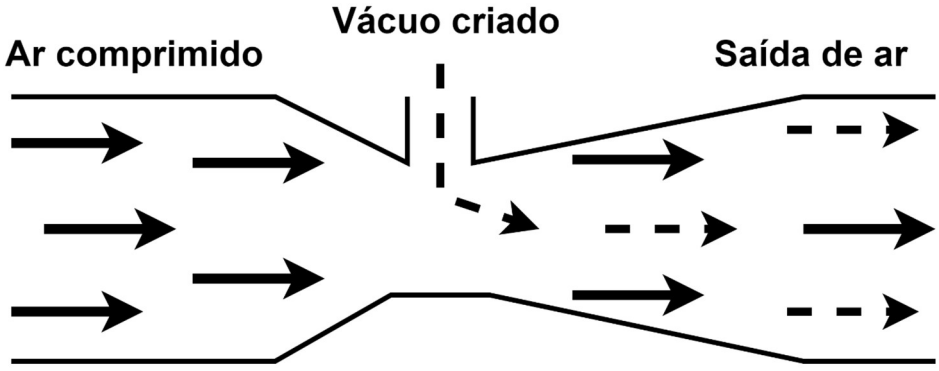


Figura 26 - Efeito de Venturi. Fonte: (Própria)

2.6.5 - Válvulas antirretorno

Durante e processo fabril, existe a possibilidade de acontecer uma falha energética, que por sua vez desliga as bombas de ar comprimido, o que faz com que deixe de haver vácuo na ventosa. Uma implementação mecânica que impede este acontecimento é a válvula de antirretorno, presente na Figura 27.

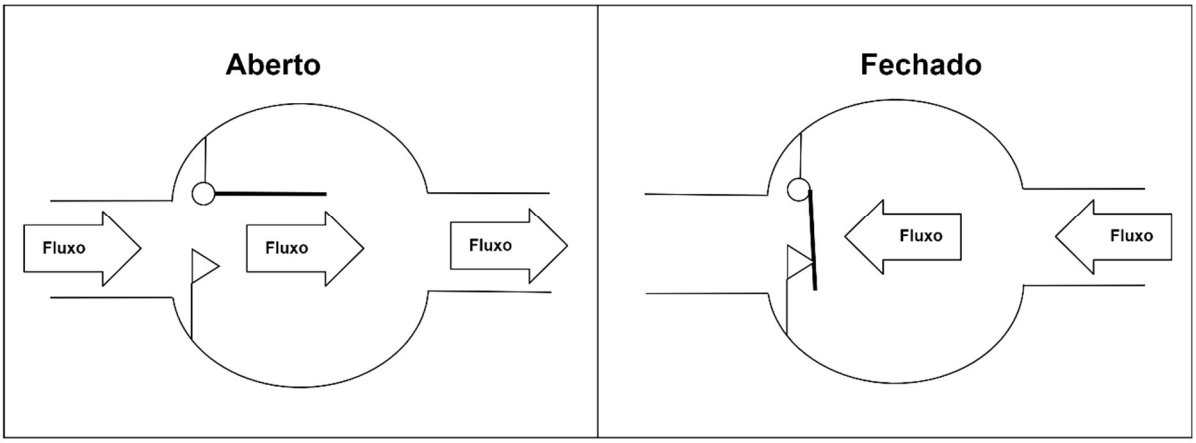


Figura 27 - Princípio de funcionamento de válvula antirretorno. Fonte: (Própria)

A válvula de antirretorno é utilizada para evitar que a direção do fluxo de um fluído se inverta no sistema de tubagem. Esta válvula controla automaticamente o sentido de direção do fluido através do princípio de pressão diferencial, ou seja, a válvula abre quando a pressão do lado de que é suposto prover o fluído é maior que a pressão do lado do qual o fluído deve ser expelido.

Deste modo, quando o ar comprimido é fechado e a garra ventosa está a agarrar um objeto, a válvula impede o ar de fluir na direção oposta, mantendo o vácuo entre a superfície do objeto a ser manipulado e a garra.

2.6.6 - Motores AC e DC

Os motores AC/DC convertem a energia elétrica fornecida em energia rotacional. O motor de corrente contínua (DC) varia a velocidade de rotação do seu eixo consoante a variação da tensão ou corrente. O motor de corrente alternada (AC) varia a velocidade de rotação do seu eixo consoante a variação da frequência da tensão, necessitando de um variador de velocidade para este fim. Estes motores podem variar a sua velocidade de rotação ou até mesmo a sua direção, porém, para parar a sua rotação é necessário cortar a alimentação do motor. Por este motivo, este tipo de motores não consegue parar com precisão numa posição exigida pelo utilizador, nem manter essa posição face a forças externas.

O princípio de funcionamento destes motores é semelhante, as bobinas existentes em torno do rotor são energizadas, criando um campo eletromagnético que faz o eixo girar, porém existe uma grande diferença para além do seu modo de alimentação.

Enquanto o motor contínuo apresenta um campo eletromagnético fixo e uma armadura girante, o motor alternado é o oposto, apresenta um campo eletromagnético girante e uma armadura fixa.

2.6.7 - Motores de passo

Os motores de passo, como o presente na Figura 28, são motores rotativos que se movimentam por incrementos, ou passos, caracterizando-se pelo controlo preciso da posição do seu eixo. Cada modelo de motor de passo possui um número específico de passos por volta, que varia entre 3 e 72 passos.

Figura 28 - Motor de passo NEMA 17. Fonte: (www.ptrobotics.com/motor-stepper/8442-motor-de-passo-nema-17-bipolar-45ncm-15a-42x42x39mm.html)

Estes motores funcionam através do controlo individual de bobinas que, quando ativadas, atraem um dos pólos de um íman presenta no eixo do motor. Dependendo do número de bobinas, o motor pode ter maior ou menor resolução. Existem três tipos principais de motores de passo, que são os de imanes permanentes, de relutância variável e os híbridos.

Motor de imanes permanentes – Possui um rotor magnetizado. Apresenta um íman permanente num eixo liso. Tem como vantagens a simplicidade mecânica, são de baixo custo e devido ao facto de este ter um campo magnético permanente que se soma ao campo eletromagnético gerado pelas bobinas, gera um binário maior no arranque. Apresenta como desvantagem a sua baixa precisão. Um exemplo do funcionamento deste motor encontra-se na Figura 29.

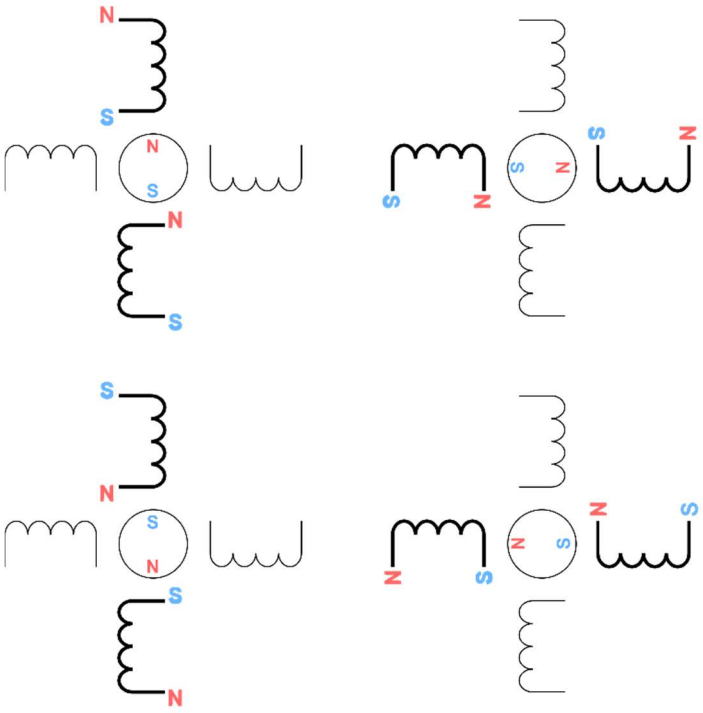


Figura 29 - Funcionamento de motor de imanes permanentes. Fonte: (Própria)

Motor de relutância variável – Não possui rotor magnetizado. Apresenta um número diferente de dentes do rotor em comparação com o número de dentes do estator, de modo que apenas um par de dentes do estator esteja alinhado com um par de dentes do rotor. O rotor roda, alternando a energização das bobines nos dentes do estator (ver Figura 30). Tem como vantagem uma maior precisão de passo, porém como não apresenta um campo magnético permanente, exerce menor binário.

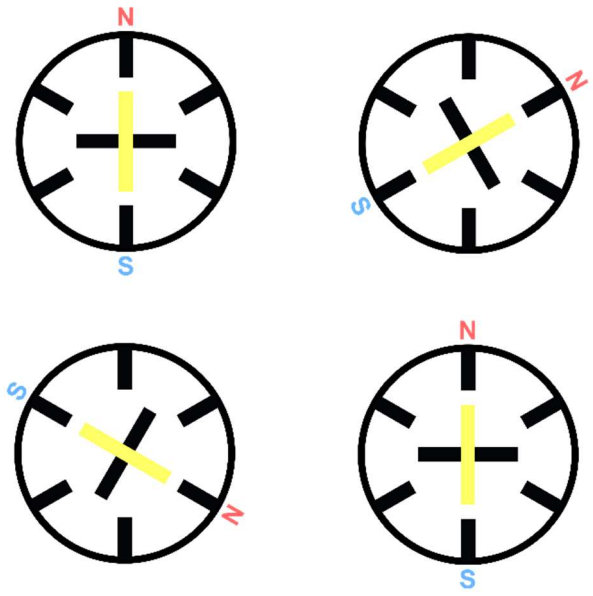


Figura 30 - Funcionamento de motor de relutância variável. Fonte: (Própria)

Motor híbrido – É uma mistura dos dois motores anteriores, ou seja, possui rotor magnetizado e um número de dentes diferente do número de dentes do estator. Este também apresenta as vantagens dos dois tipos de motores anteriores, ou seja, a sua precisão é maior e consegue debitar um maior binário. O eixo deste motor é construído com dois tipos de dentes, numa extremidade o polo Sul e na outra o polo Norte, os dentes destes dois pólos estão desalinhados um com o outro, de modo que estes fiquem alternados. O rotor gira do mesmo modo que os motores anteriores, alternando a energização das bobinas (ver Figura 31). Um exemplo deste motor de passo encontra-se presente na Figura 28.

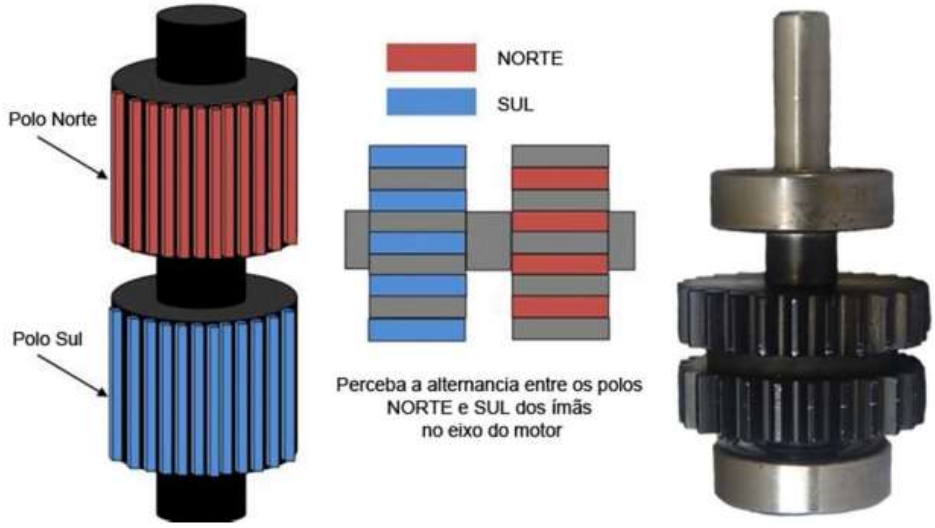


Figura 31 - Funcionamento do motor híbrido. Fonte: (www.curtocircuito.com.br/blog/motor-de-passo/introducao-ao-motor-de-passo)

2.6.8 - Servos motores de posição (180)

Um servo motor é, no fundo um motor AC ou DC controlado por um circuito de controlo e um potenciómetro. Os servos motores AC são geralmente utilizados em ambientes industriais, uma vez que apresentam maior binário, enquanto os servos motores DC são geralmente utilizados em projetos menores, devido ao seu menor custo.

Este motor tem uma rotação máxima de 180 graus, existindo pinos nas suas engrenagens que o impedem de executar rotações superiores. O objetivo deste tipo de motores é fazer o motor rodar para uma posição específica, mantendo memória da sua posição atual.

O motor sabe o valor da sua posição atual através do valor obtido pelo potenciómetro, que por sua vez está ligado diretamente às engrenagens do motor, ou seja, consoante a rotação do motor, o valor da resistência do potenciómetro também varia, sendo possível obter a posição exata de rotação do motor. Para comandar o servo motor a

rodar para uma posição pretendida, é necessário enviar um sinal PWM (pulse width modulation), cuja largura de pulso, que pode variar entre 1ms a 2ms, corresponde a um ângulo entre os 0 e os 180 graus (ver Figura 32).

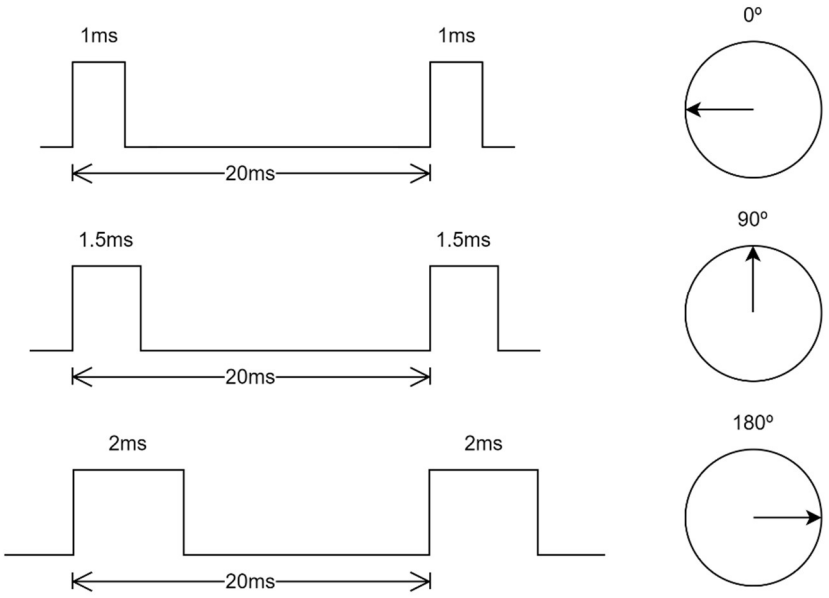


Figura 32 - Pulsos e rotação de servo motor de posição. Fonte: (Própria)

2.6.9 - Servos motores contínuos (360)

O servo motor contínuo funciona do mesmo modo que o servo motor de posição, porém este apresenta duas grandes diferenças. A primeira é que este servo motor não apresenta o pino limitador de posição na engrenagem, permitindo que este rode 360 graus ou mais.

A segunda diferença é o facto de o potenciómetro não estar ligado às engrenagens do motor, ou seja, este não varia o seu valor em função da rotação do motor. Assim sendo, como o valor da resistência do potenciómetro é fixo, o controlador recebe sempre o mesmo valor, estando a rodar ou não. Devido a isto, este motor não guarda memória da sua posição atual, ou seja, dependendo do valor da resistência que o potenciómetro esteja regulado, o controlador irá assumir que o motor está sempre na posição calibrada pelo potenciómetro.

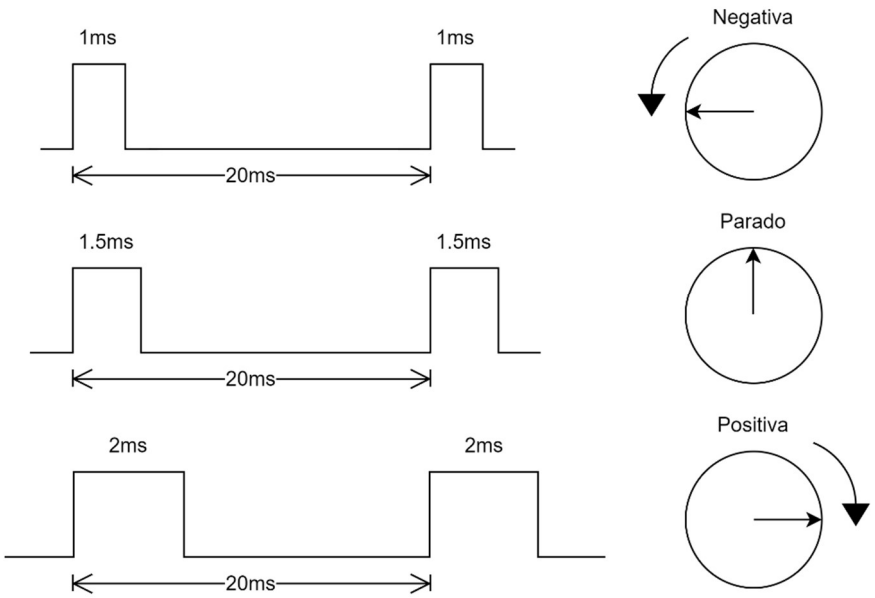


Figura 33 - Pulsos e rotação de servo motor contínuo calibrado para 90º. Fonte: (Própria)

Como é possível verificar na Figura 33, o valor do potenciómetro está calibrado para 90 graus, correspondente à largura de pulso intermédia (1.5ms). Neste caso, quando o valor de posição enviado para o controlador for menor que 90 graus, o motor vai rodar infinitamente no sentido contrário ao dos ponteiros do relógio. Se o valor enviado for maior que 90 graus, o motor vai rodar infinitamente no sentido dos ponteiros do relógio. Por último, se o valor enviado for de 90 graus, o motor vai parar. Nota-se que quanto mais próximo é o valor enviado do valor calibrado, mais lenta é a rotação do motor, e quanto mais afastado é o valor enviado, mais rápida roda.

CAPÍTULO 3. - ATIVIDADES DESENVOLVIDAS

O estágio curricular, decorrido no Centro Tecnológico da Cerâmica e do Vidro (CTCV), deu início no dia 18 de outubro de 2021 e deu termo em 30 de junho de 2022, tendo uma duração total de oito meses e meio. A área da robótica e automação do centro resulta de uma parceria recente com o Instituto Superior de Engenharia de Coimbra (ISEC), permitindo o apoio de aquisição de conhecimentos essenciais nesta área tecnológica, e na promoção futura de projetos I&D.

Assim sendo, no decorrer deste estágio foram desenvolvidas simulações de ambientes de trabalho industrial de clientes do CTCV, tanto simulações digitais como físicas em menor escala.

As simulações digitais foram programadas através da aplicação RobotStudio2021 da ABB para simular a substituição de trabalho manual por robôs industriais ABB. Quanto às simulações reais em menor escala, foram desenvolvidas garras para dois robôs manipuladores disponibilizados pelo centro.

3.1 - Simulação para paletização automática

Foi proposto pela empresa Aleluia Cerâmicas SA a substituição de trabalhadores por robôs manipuladores na paletização e despaletização de azulejos cerâmicos. Atualmente, um funcionário executa a despaletização os azulejos, um a um, para um tapete transportador, como mostra a Figura 34. Na outra extremidade do tapete transportador, outros dois funcionários colocam os azulejos em pilha em cima de uma caixa que depois segue através de outro tapete transportador para um embalador automático. Uma vez embaladas, as caixas seguem num outro tapete transportador para uma doca de paletização, onde um funcionário paletiza as caixas numa palete. Quando a palete está cheia, uma empilhadora move a palete para a doca de carga e coloca uma palete vazia no mesmo local.

Figura 34 - Colocação de azulejos manualmente no tapete. Fonte: (Própria)

A proposta colocada é substituir quatro funcionários por três robôs industriais, formando os funcionários substituídos de modo que apresentem capacidade para supervisionar o processo completo ou colocá-los noutro ambiente de trabalho onde seja necessária intervenção humana. Todos os dados de dimensionamento do local e dos azulejos são estimados através da visualização de vídeos da execução do processo manual, o que implica que estes não correspondam com a realidade.

Assim sendo, este subcapítulo trata-se de um guia cujo enfoque é proporcionar as ferramentas básicas na criação de uma réplica digital através da aplicação RobotStudio 2021 da ABB, dirigida a uma plateia principiante na utilização deste programa, tendo como exemplo, alunos do ensino superior.

3.1.1 - Simulador RobotStudio

A utilização de um simulador na criação de réplicas digitais é de extrema importância, pois este permite desenvolver o programa do robô industrial sem a necessidade de haver deslocação para a fábrica real. Também permite aos projetos de automação serem ensaiados e estruturados sem a necessidade de montar e desmontar máquinas, transportadores, cabos elétricos, e restantes elementos fabris. Por sua vez, também serve para criar e simular novas garras, possibilitando a escolha da mais adequada para a realização da tarefa pretendida.

Para construir uma réplica digital do espaço fabril com a implementação dos robôs manipuladores desejados, recorreu-se à aplicação RobotStudio 2021, presente na Figura 35. Esta aplicação pertence à ABB, é uma aplicação cujo fim é criar réplicas digitais de espaços físicos com a aplicação de equipamentos manipuladores ou posicionadores da marca ABB.

De modo a ser possível programar nesta aplicação é necessária ter uma licença de utilização, tendo esta sido fornecida pelo ISEC. Para a licença funcionar é necessário o utilizador efetuar a ligação à rede de internet do ISEC ou utilizar a VPN da mesma.

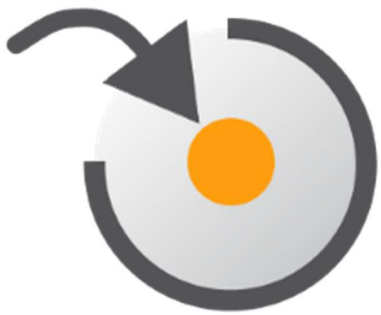


Figura 35 - Ícone da aplicação RobotStudio. Fonte: (https://github.com/simple-icons/simple-icons/issues/1870)

3.1.2 - Criação de um novo projeto

O primeiro passo na criação da réplica digital é criar um projeto RobotStudio. Como se pode observar na Figura 36, é necessário abrir a aplicação e escolher a opção "*New*" → *"Project"* e preencher os seguintes dados com o nome do projeto, local onde se vai guardar o projeto e, selecionar a opção "*Include a Robot and Virtual Controller*".

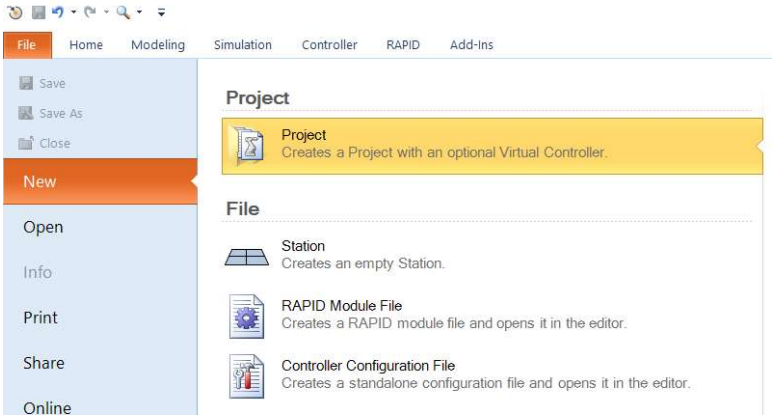


Figura 36 - Criação de um novo projeto. Fonte: (ABB)

Feito isto, é desbloqueado um conjunto de campos a preencher, tais como o nome do controlador, o local onde este vai ser guardado e o modelo do robô. O campo "*RobotWare*" deve ser deixado como padrão, pois corresponde à família do *software* do controlador do robô mais recente e compatível com a lista padrão de robôs, de acordo com a Figura 37.

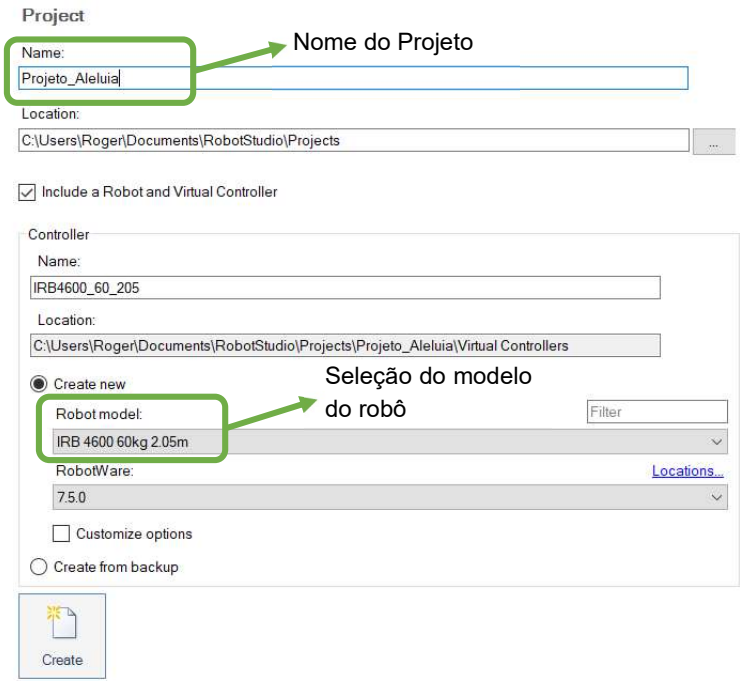


Figura 37 - Campos de inserção para criação de novo projeto. Fonte: (ABB)

Selecionando o campo "*Robot model*" é possível selecionar vários robôs manipuladores e observar as características de carga máxima e alcance. Após a seleção do robô, precede-se em clicar em "*Create*", demonstrado na Figura 38.

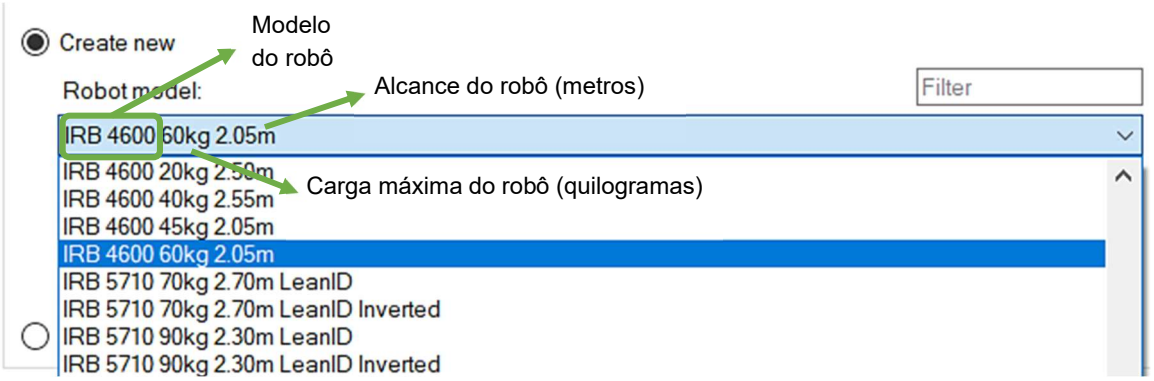


Figura 38 - Seleção do robô manipulador. Fonte: (ABB)

O robô manipulador selecionado, presente na Figura 39, foi o IRB 4600, capaz de suportar 60 quilogramas, com 2.05 metros de alcance do tipo C. Este robô foi escolhido devido ao seu grande alcance e elevado limite de carga. Apesar do peso de uma caixa com azulejos estar perto de 20 kg, é necessário ter em conta que uma garra que suporte este tipo de cargas também apresenta um peso elevado, que tem de ser suportado pelo robô. Tendo em conta o local de operação do robô, o seu alcance deve ser aproximadamente o alcance de duas pessoas, uma vez que é o número máximo de pessoas que um destes robôs irá substituir e estas encontram-se ao lado uma da outra, a escolha do alcance de 2.05 metros é segura e justificada.

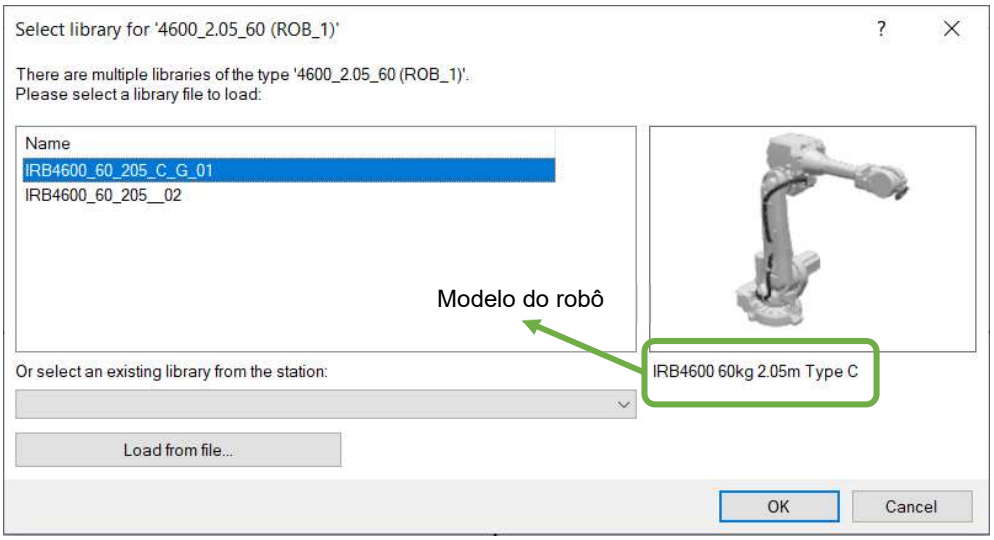


Figura 39 - Escolha do tipo de robô. Fonte: (ABB)

Após escolher o modelo desejado do robô, é criado um ambiente vazio onde o ponto zero da base do robô coincide com o zero do mundo criado. A área de trabalho principal apresenta as seguintes funções apresentadas na Figura 40.

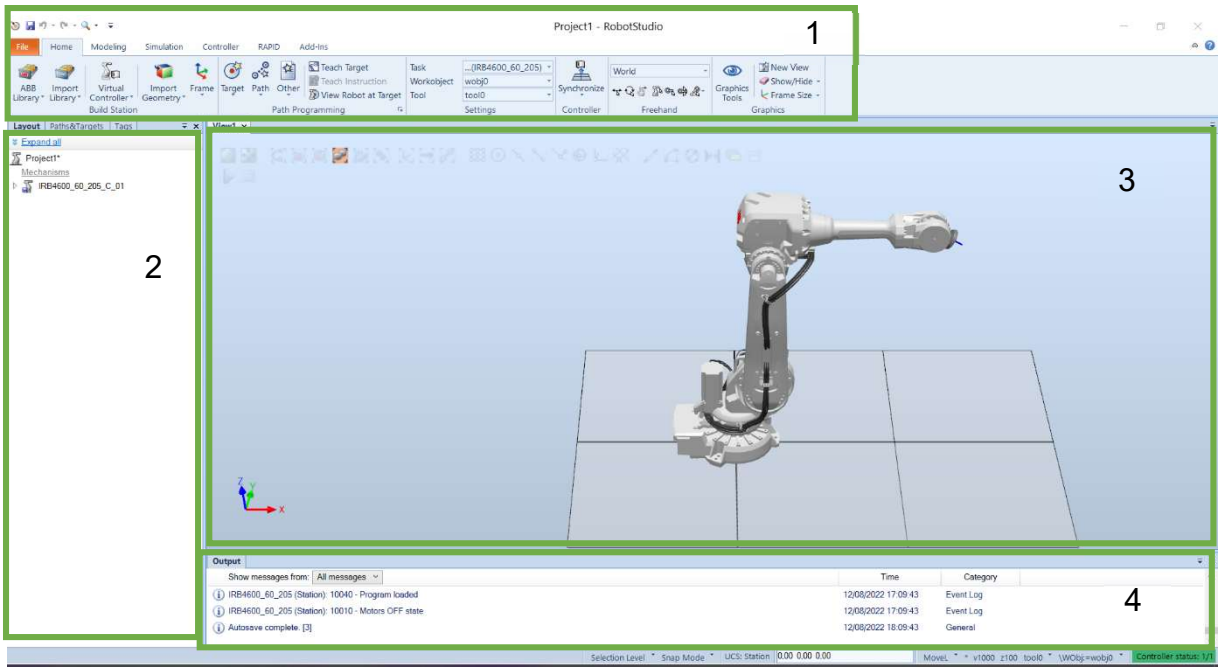


Figura 40 - Ambiente principal de trabalho. (1) - Barra de tarefas; (2) - Janela de esquema; (3) . Vista do mundo; (4) - Saída de mensagens do(s) controlador(es). Fonte: (ABB)

3.1.3 - A Barra de tarefas

É através da barra de tarefas que é possível aceder a todas as funcionalidades do RobotStudio. Esta apresenta sete separadores principais que serão expostos de um modo geral.

- **File** → Separador que permite executar as funções comuns de um separador deste tipo. Através deste separador é possível criar, abrir, ver informações, imprimir, alterar definições, fechar, entre outros.
- **Home** → Separador observado na Figura 41 que permite aceder às ferramentas essenciais na manipulação do robô no mundo.

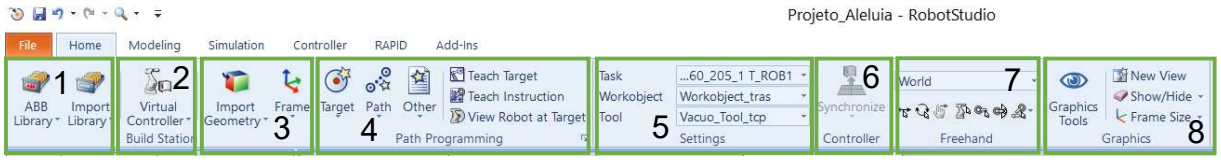


Figura 41 - Separador "Home". Fonte: (ABB)

1. Aceder às librarias ABB (robôs manipuladores, robôs móveis, posicionadores e tapetes rolantes) ou importadas (equipamentos normalmente presentes em ambiente fabril, controladores, garras, objetos criados e guardados, etc.);
2. Criar um controlador;
3. Importar geometrias ou criar referenciais;

4. Ferramentas de auxílio de programação ponto a ponto do robô, por exemplo definição de pontos "*Target*" aos quais o robô pode aceder automaticamente, criar um caminho em contorno da superfície de um objeto, entre outros;
5. Seleção do controlador, ferramenta ou o "*workobject*" pretendido;
6. Sincronizar o código escrito em linguagem Rapid com a simulação criada ou vice-versa;
7. Definir o modo como se movimenta o robô manualmente com o rato, desde um "jog" linear, angular ou de arraste;
8. Controlo dos gráficos e seleções de visibilidade de caminhos executados pelo robô, referenciais e criação de novas vistas.

- **Modeling** → Separador mostrado por partes na Figura 42 e na Figura 44 que permite criar objetos, alterar a física do mundo, executar medições e criar mecanismos, tais como de ferramentas, tapetes transportadores, entre outros.

Figura 42 - Separador "Modeling"(parte 1). Fonte: (ABB)

1. Criação de grupos de componentes, componentes vazios e de "*Smart Components*", que podem ser munidos de lógica programável por blocos, tais como os da Figura 43;
2. Criação de sólidos;
3. Criação de cabos, física de juntas do robô e do mundo;

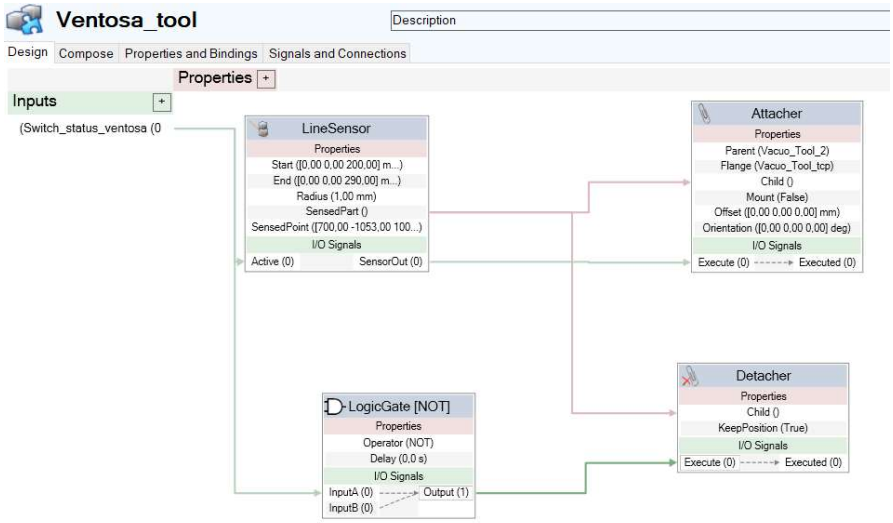


Figura 43 - Exemplo de "Smart Component" com lógica de blocos. Fonte: (Própria)

Figura 44 - Separador "Modeling" (parte 2). Fonte: (ABB)

4. Operações CAD;
5. Medições ponto a ponto, ângulo, diâmetro e de distância mínima entre superfícies ou linhas;
6. Definir o modo como se movimenta o robô manualmente com o rato, desde um "jog" linear, angular ou de arraste;
7. Criação de mecanismos munidos de movimento e lógica.

- **Simulation** → Separador observado na Figura 45 que permite correr a simulação, fazer "*Reset*", gravar, entre outras ferramentas de gestão de vídeo.

Figura 45 - Separador "Simulation". Fonte: (ABB)

1. Configurações de simulação e de colisão;
2. Controlo de simulação;
3. Monitor de sinais I/O, traços de movimento do robô e "*Stopwatch*";
4. Análise de sinais;
5. Ferramentas de gravação.

- **Controller** → Separador mostrado por partes na Figura 46 e na Figura 48 que permite aceder às ferramentas essenciais de gestão do(s) controlador(es).

Figura 46 - Separador "Controller" (parte 1). Fonte: (ABB)

1. Adicionar um novo controlador e outros acessos;
2. Ferramentas do controlador, desde reinício, criação de um "Backup", eventos, um simulador da consola do robô mostrado na Figura 47, entre outros;
3. Gestão dos sinais de entrada/saída;

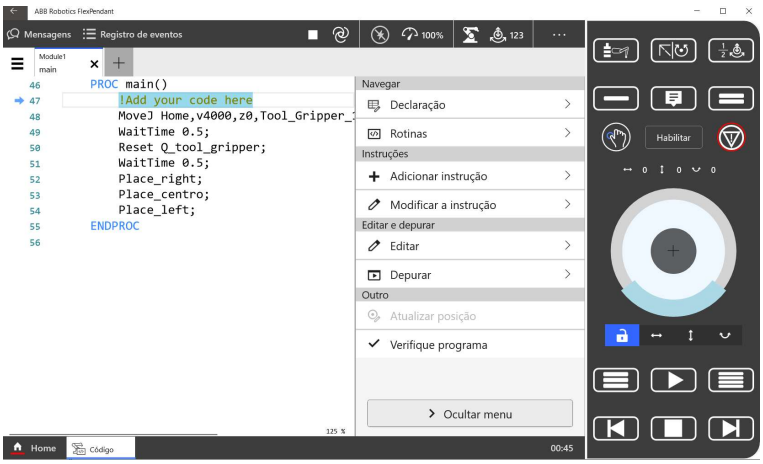


Figura 47 - Simulador de consola FlexPendant. Fonte: (ABB)

Figura 48 - Separador "Controller" (parte 2). Fonte: (ABB)

4. Configuração , carga e salvar de parâmetros e propriedades do controlador;
5. Outras configurações do controlador, desde a instalação de versões do "*RobotWare*", ferramentas de rastreio de tapetes transportadores, entre outros.

- **Rapid** → Separador observado por partes na Figura 49 e na Figura 50 que permite aceder a ferramentas de edição de código em linguagem Rapid presente no controlador. Apresenta ferramentas de auxílio de escrita, de sincronização do código com a simulação, entre outros.

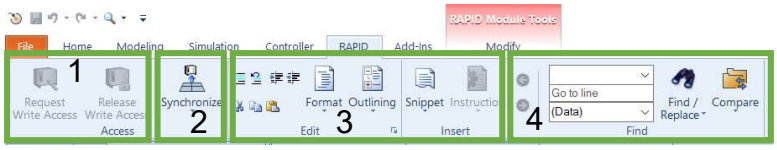


Figura 49 - Separador "Rapid" (parte 1). Fonte: (ABB)

1. Configuração de acessos de escrita;
2. Sincronizar o código escrito em linguagem Rapid com a simulação criada ou vice-versa;
3. Ferramentas de edição de texto e de inserção de instruções do robô ABB;
4. Ferramentas de busca;

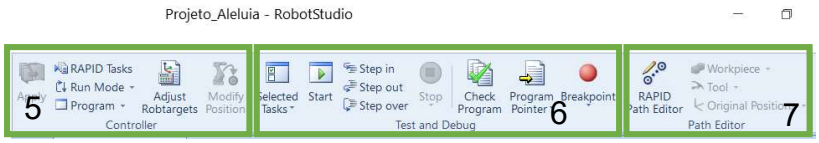


Figura 50 - Separador "Rapid" (parte 2). Fonte: (ABB)

5. Aplicação do código Rapid escrito no controlador, ajuste de pontos, de referenciais da garra e outras ferramentas de edição do controlador;
6. Secção para testes e “Debug”;
7. Edição de caminho.

- **Add-Ins** → Separador responsável pela gestão de pacotes online. Estes pacotes podem conter o download de ferramentas para o robô, mesas de ensaio, tapetes transportadores, máquinas CNC, entre outros.

3.1.4 - Criação da garra do robô

No simulador RobotStudio, a criação de uma garra é o primeiro passo essencial na criação de uma réplica digital. Deste modo é possível, em simulação, escolher o melhor tipo de garra robótica para efetuar as tarefas propostas. Para tal é necessário saber quais os objetos que estes irão manipular. Uma vez que as dimensões do azulejo não foram fornecidas, estas foram estimadas através da visualização de um vídeo do processo fabril, onde se estimou que as suas dimensões são 1000x400x50 mm. Para criar um objeto sólido em forma de paralelepípedo, é necessário aceder a barra de tarefas “*Modeling*” e escolher as opções “*Solid*” → “*Box*”. De seguida preenche-se os campos de acordo com a Figura 51.

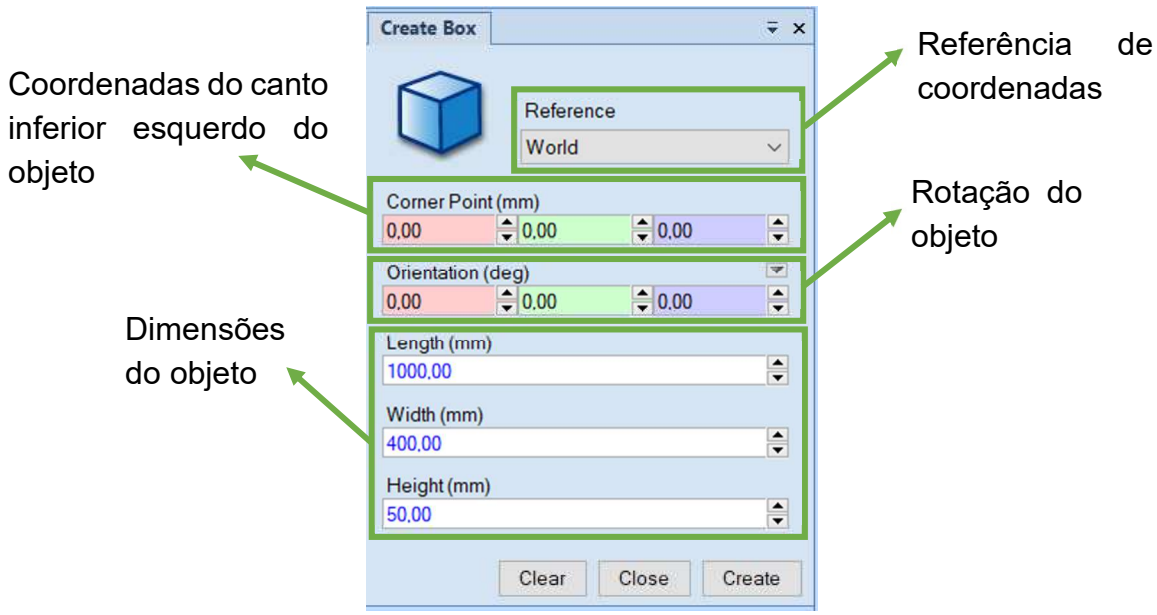


Figura 51 - Criação do azulejo. Fonte: (ABB)

Por motivos de estética, é possível selecionar o azulejo com o botão direito do rato e selecionar "*Modify*" → "*Set Color…*" para alterar a cor ou "*Modify*" → "*Graphic Appearance*" para alterar a cor e textura do objeto, de modo a obter a aparência observada na Figura 52.

Figura 52 - Azulejo criado. Fonte: (Própria)

De seguida, criou-se uma garra com dez ventosas para agarrar este azulejo. O primeiro passo é criar a forma física da garra, utilizando formas sólidas individuais para a compor, executado do mesmo modo que na criação do azulejo, porém colocá-las dentro de um grupo de componentes. Este grupo de componentes é possível criar a partir da barra de tarefas "*Modeling*" →"*Component Group*" e é visto na janela de "*Layout*". Estes resultados podem ser observados na Figura 53 e na Figura 54.

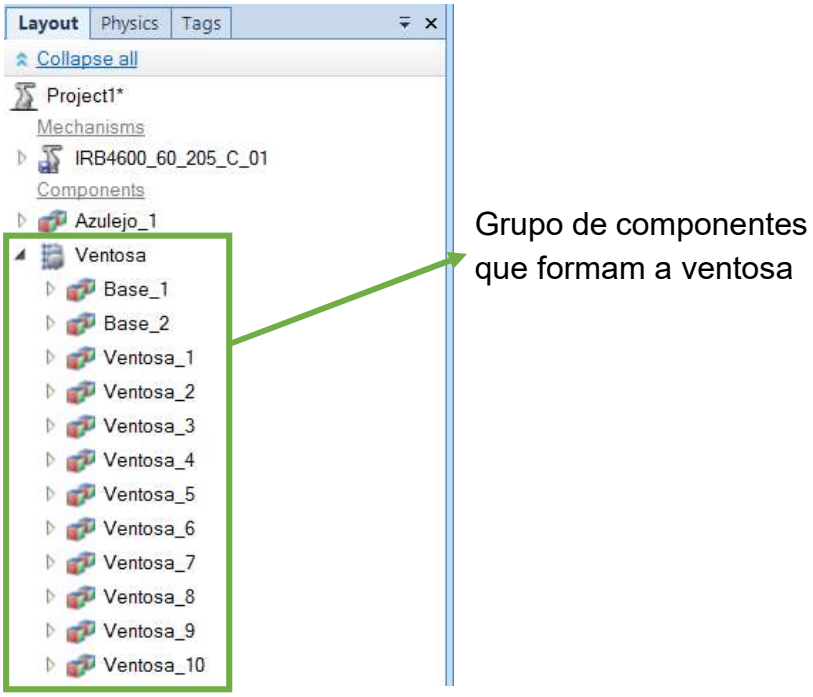


Figura 53 - Janela "Layout" com ventosa. Fonte: (Própria)

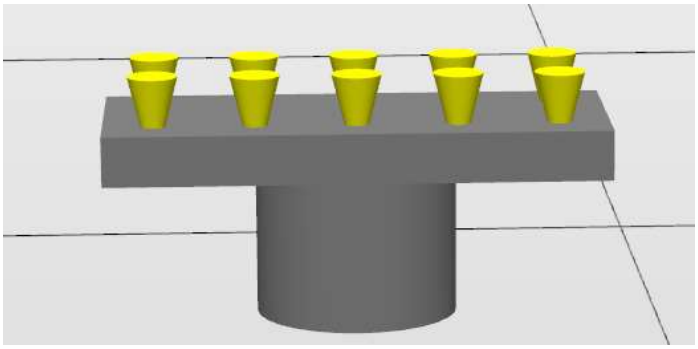


Figura 54 - Garra Ventosa. Fonte: (Própria)

Nota-se que o ponto central da base inferior (correspondente ao ponto de encaixe da garra no robô) deve coincidir com o zero referencial do mundo para obter melhores resultados na criação da garra, tal como é possível observar na Figura 55. Para mover um sólido ou grupo de componentes, basta clicar com o botão direito do rato sobre ela e selecionar "*Position*" → "*Set Position*" e selecionar a posição do canto inferior esquerdo de modo que o centro da base coincida com o referencial.

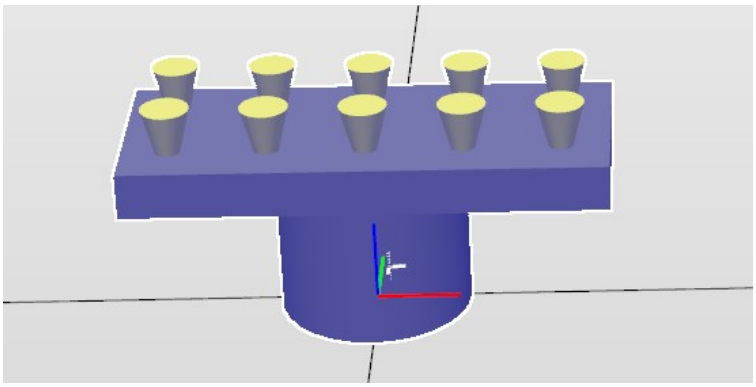


Figura 55 - Garra Ventosa com referencial zero no centro da base inferior. Fonte: (Própria)

Com o corpo sólido criado, é necessário transformá-lo agora numa ferramenta. Na barra de tarefas "*Modeling*" acedendo a "*Create Tool*" é aberto uma janela com campos a preencher com informação da ferramenta. Preenchendo o nome da ferramenta criada, existe a possibilidade de utilizar um "*Dummy*" ou de uma forma geométrica já existente para servir de corpo da garra. Para escolher um corpo já existente coloca-se um visto na opção "*Use Existing*" e escolher o corpo desejado. Nota-se que apenas um corpo pode ser selecionado, daí colocar os sólidos criados dentro de um grupo, de modo a ser possível selecioná-los como um conjunto. Para criar físicas de simulação melhores, é possível adicionar uma massa à garra e a posição do seu centro de gravidade, porém este passo é opcional, tal como mostra a Figura 56.

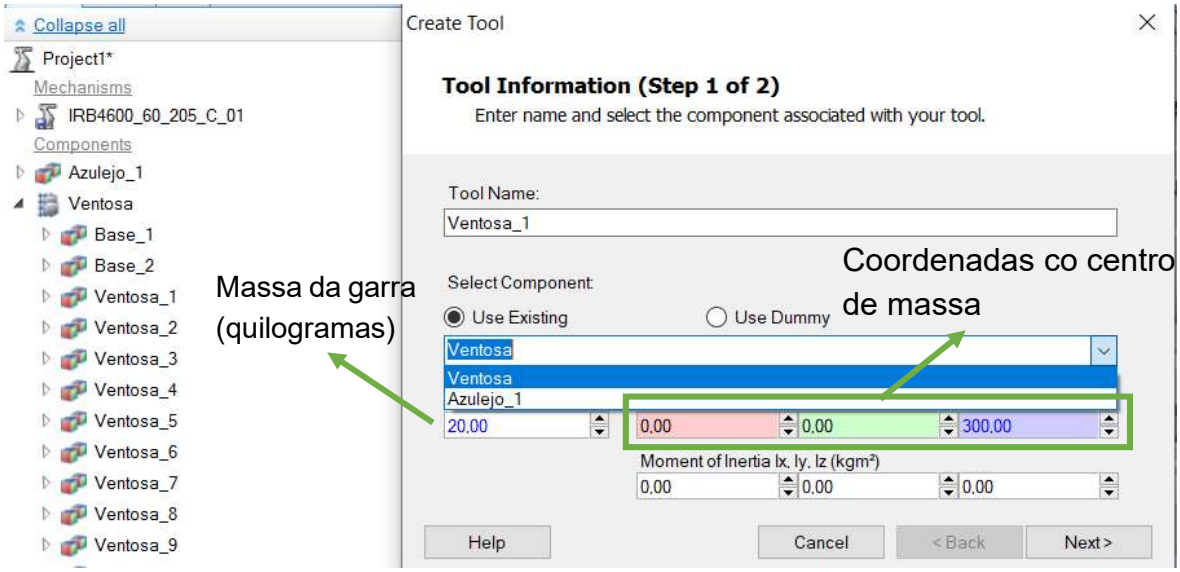


Figura 56 - Passo 1 na criação de uma ferramenta. Fonte: (Própria)

Após selecionar o botão "*Next*", o segundo passo na criação da garra é a definição do ponto TCP (Tool Center Point), ou seja, quando é enviado um ponto de coordenadas para o robô com a garra, este ponto irá coincidir com o TCP. Conforme a Figura 57, o ponto é definido através de um deslocamento em Z de 350 mm em comparação ao ponto de referência zero do objeto, que coincide com a base central inferior, ou com a última junta do robô (zona de encaixe).

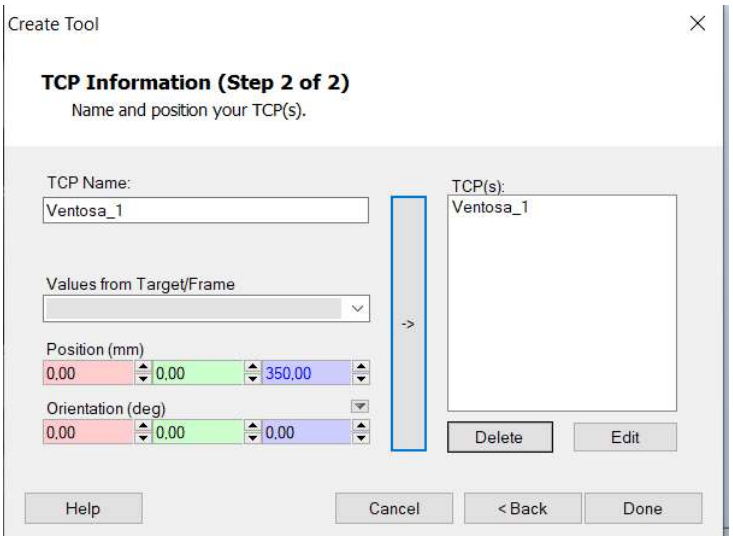


Figura 57 - Passo 2 na criação de uma ferramenta. Fonte: (Própria)

Com a criação da garra, é necessário dar-lhe a capacidade de, quando comandada, agarrar o objeto que se encontra em contacto com esta. Para tal é necessário criar um componente munido de inteligência para colocar um sensor e criar lógica de sinais. Para tal, na barra de tarefas "*Modeling*" seleciona-se a opção "*Smart Component*" e na janela de "*Layout*" arrasta-se a garra criada para dentro do componente inteligente.

Após isto, seleciona-se o componente com o botão direito do rato e clica-se em "*Edit Component*", abrindo uma segunda janela com o conteúdo mostrado na Figura 58.

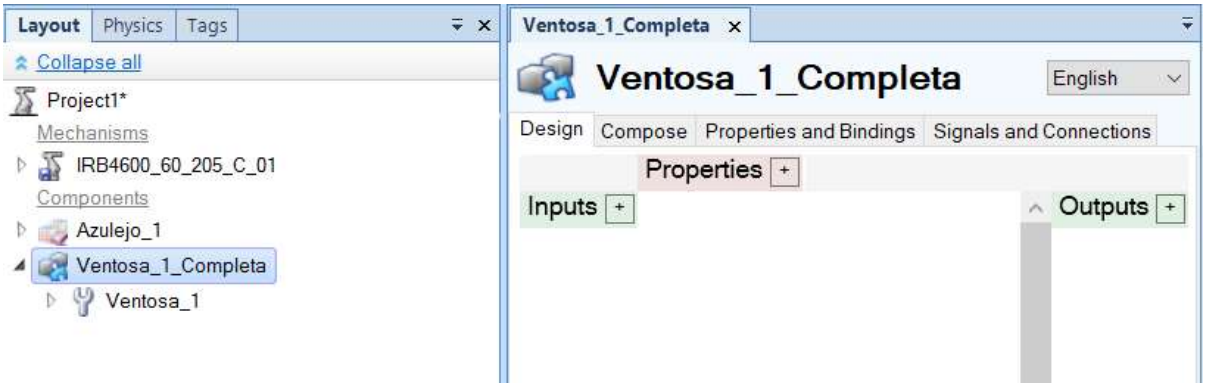


Figura 58 - Criação de um "Smart Component" para atribuir lógica de sinais à garra. Fonte: (Própria)

Nesta janela existem dois retângulos com a descrição "Inputs" e "Outputs". Como esta garra está a simular uma simples ventosa de vácuo, necessita apenas de uma entrada lógica, correspondente ao sinal recebido do robô para ativar o vácuo. Para tal, seleciona-se o a opção "Inputs" e atribui-se um nome identificativo ao sinal, prosseguindo com "*Ok*", tal como monstra a Figura 59.

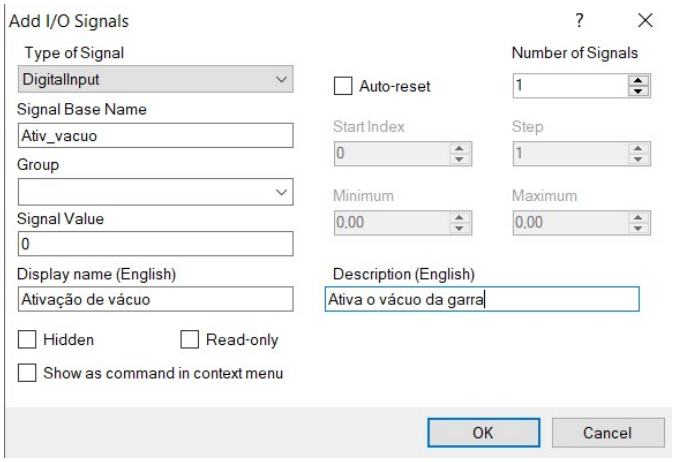


Figura 59 - Criação de entrada de sinal da garra. Fonte: (Própria)

De seguida seleciona-se a janela "*Compose*" e cria-se, selecionando a opção "*Add component*", um sensor linear num local onde este consiga detetar o objeto que a garra irá agarrar, preenchendo os campos observados na Figura 60.

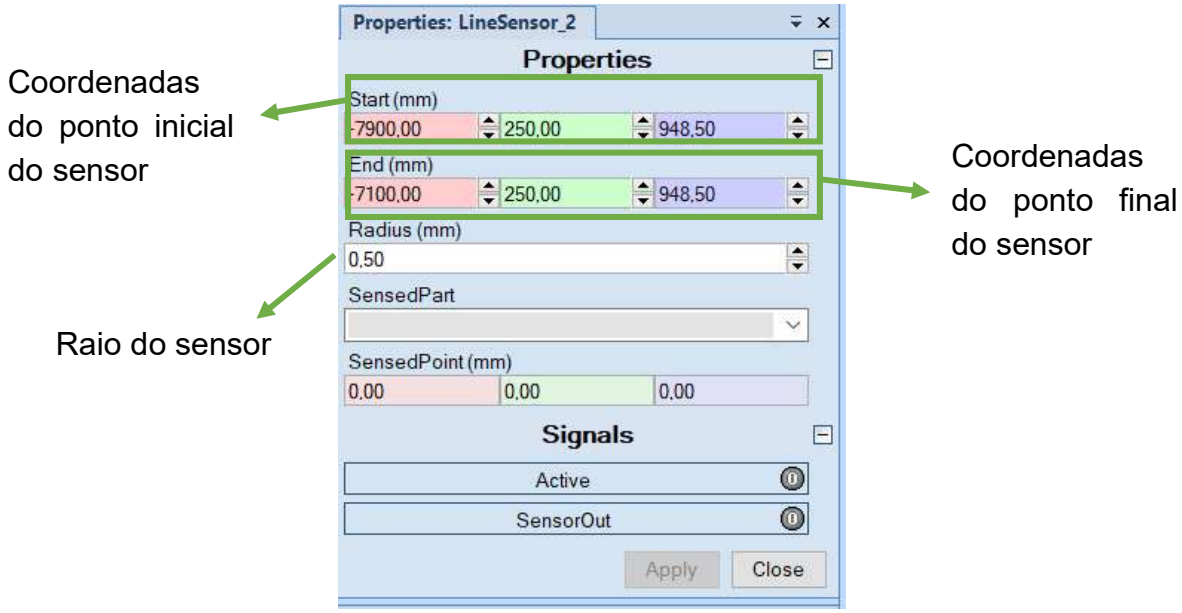


Figura 60 - Criação do sensor linear. Fonte: (Própria)

Cria-se, conforme a Figura 61, uma ação de agarre chamada "*Attacher*" onde o seu "*Parent*" seja a ventosa (ícone da ferramenta "*tool*") e o "*Flange*" seja o TCP criado, e de seguida, uma ação de desgarre chamada "*Detacher*" com um visto na opção "*KeepPosition*" e por último um portão lógico "*NOT*".

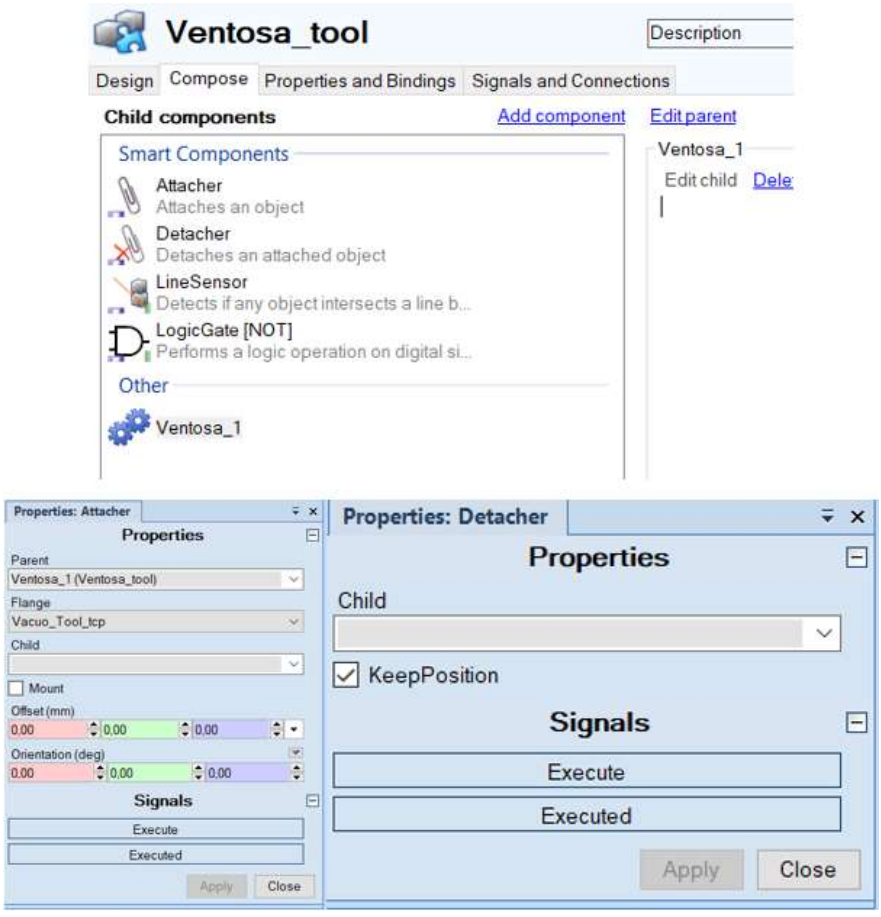


Figura 61 - Criação das componentes lógicas da garra. Fonte: (Própria)

De seguida seleciona-se a janela "*Design*" e efetua-se as seguintes ligações de acordo com a Figura 62.

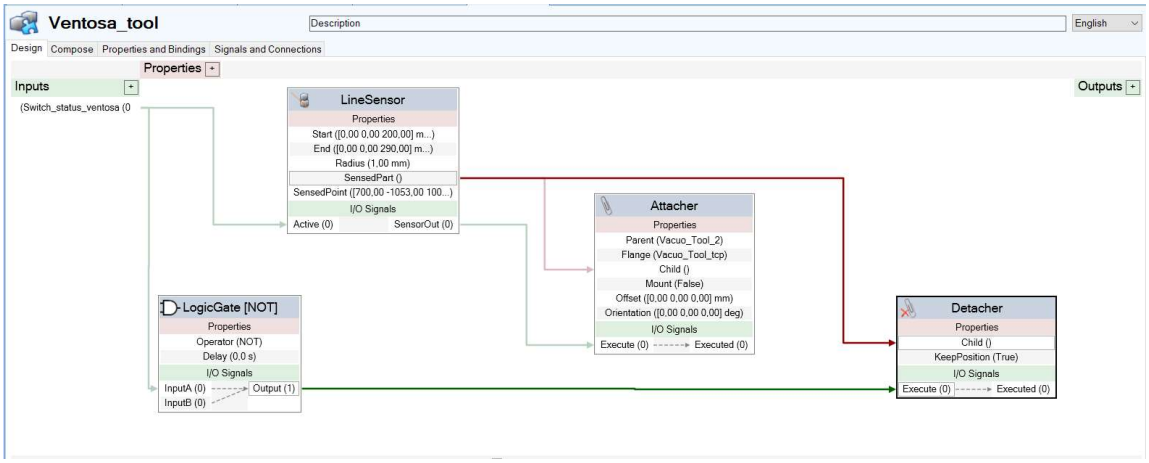


Figura 62 - Lógica de blocos da ventosa. Fonte: (Própria)

Em suma, sempre que o robô envia o sinal de acionamento da ventosa, esta ativa o sensor que deteta o nome do objeto em contacto com este, devolvendo a informação ao "*Attacher*" e ao "*Detacher*". Uma vez detetada a peça, ativa o "*Attacher*". Quando o sinal do robô é desativado, o portão lógico "*NOT*" inverte o sinal de modo a ativar o "*Detacher*" que larga o objeto detetado pelo sensor no local.

Com um sinal de entrada criado, é necessário criar no controlador do robô um sinal de saída e efetuar a comunicar com este. Para tal seleciona-se na barra de tarefas "*Controller*" → "*Configuration*" → "*I/O System*". Na nova janela apresentada na Figura 63, seleciona-se na barra "*Type*" a opção "*Signal*" com o botão direito do rato seleciona-se a opção "*New Signal*".

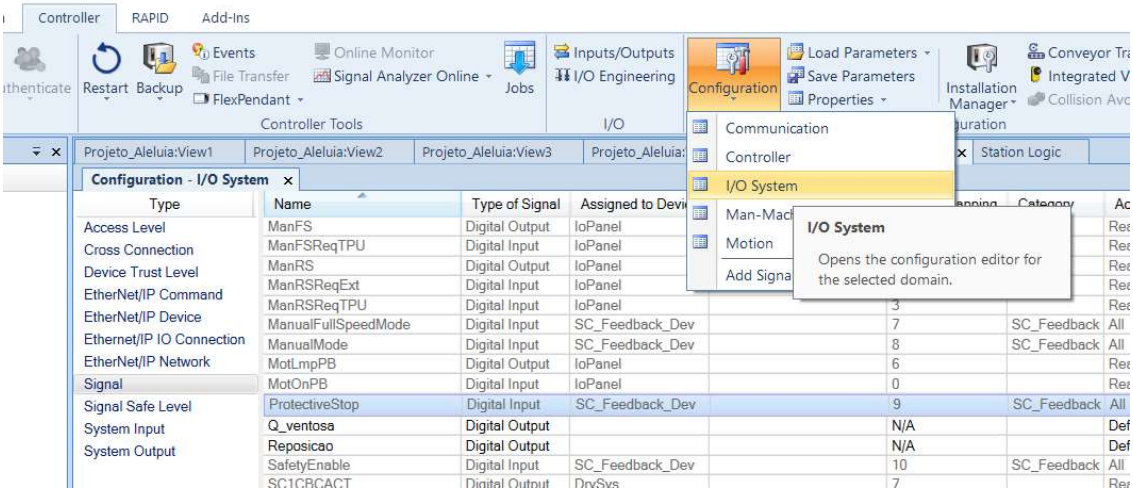


Figura 63 - Criação de sinal de saída do controlador (parte 1). Fonte: (Própria)

Para criar um sinal de saída, aplica-se um nome e em "*Type of Signal*" seleciona-se a opção "*Digital Output*", tal como mostra a Figura 64. De seguida reinicia-se o controlador na opção "*Restart*" presente na barra de tarefas "*Controller*".

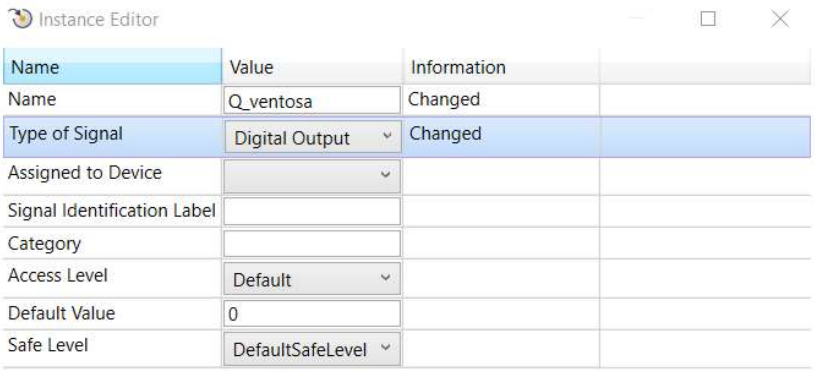


Figura 64 - Criação do sinal de saída do controlador (parte 2). Fonte: (Própria)

Para interligar o sinal de saída do controlador com o sinal de entrada da garra, seleciona-se na barra de tarefas "*Simulation*" → "*Station Logic*", procura-se o sinal pretendido no controlador (neste caso o bloco com o nome "IRB4600_60_205") e liga-se ao sinal de ativação da garra presente no bloco da ventosa, como é possível observar na Figura 65.

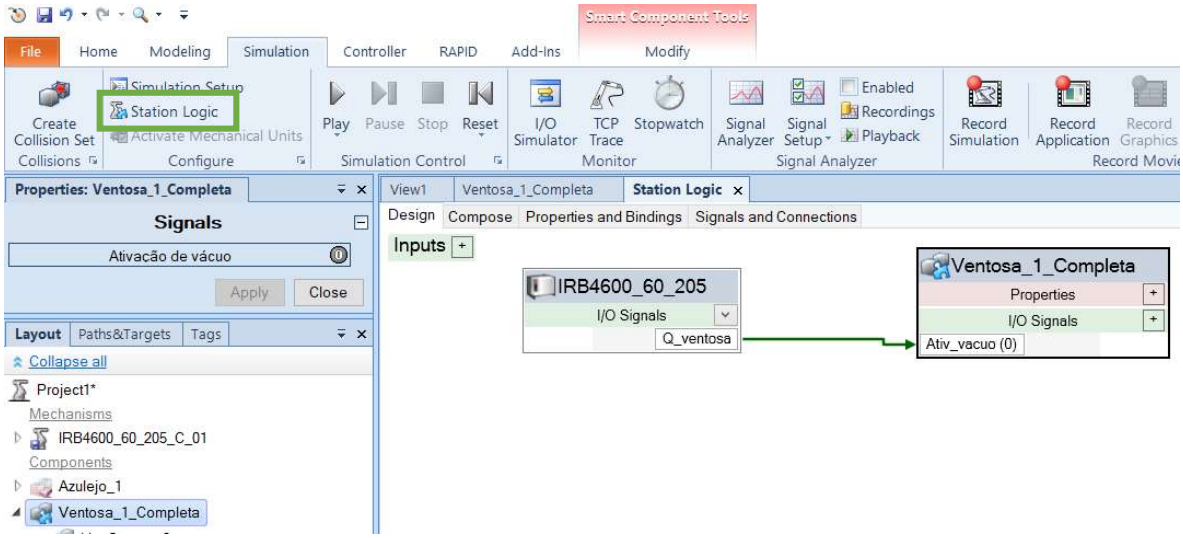


Figura 65 - Interligação do sinal do controlador com o sinal da garra. Fonte: (Própria)

Na janela de esquema, arrasta-se o "Smart component" da garra para o robô, seleciona-se a opção sim e a garra fica colocada na extremidade do robô, obtendo o resultado apresentado na Figura 66. Contudo, os dados da ferramenta "Tooldata" não estão atualizados, sendo necessário efetuar a sua criação.

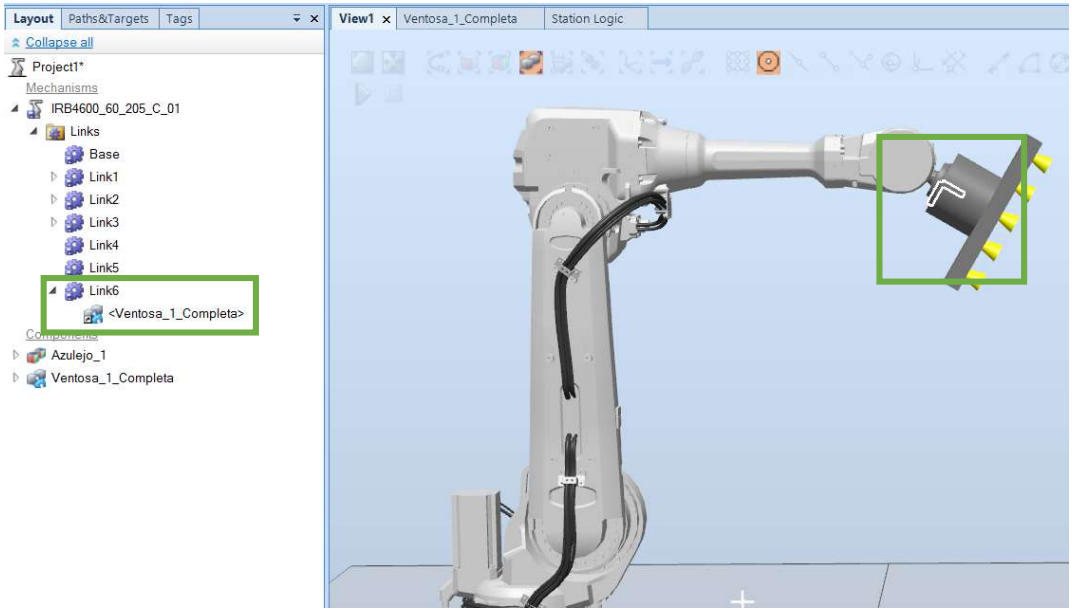


Figura 66 - Ventosa posicionada no robô (Tool Data incorreta). Fonte: (Própria)

Para criar os dados da ferramenta, acede-se na barra de tarefas "*Home*" → "*Path Programming*" → "*Other*" → "*Create Tooldata*", abrindo a janela presente na Figura 67. Escolhe-se o nome e altera-se o campo "*Tool Frame*" → "*Position x,y,z*" conforme o deslocamento do TCP executado na Figura 57, que neste caso foi de 350 mm segundo o eixo Z.

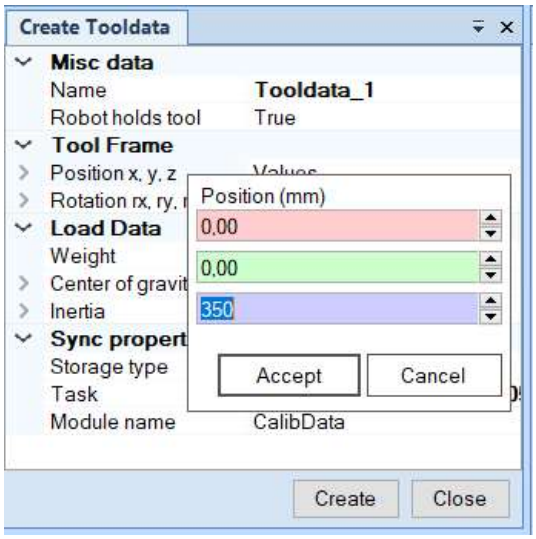


Figura 67 - Criação dos dados da ferramenta. Fonte: (Própria)

Após clicar em "Create", seleciona-se na barra de tarefas "*Home*" → "*Settings*" → "*Tool*" a ferramenta certa e, deste modo, a garra está funcional, tal como mostram os destaques da Figura 68.

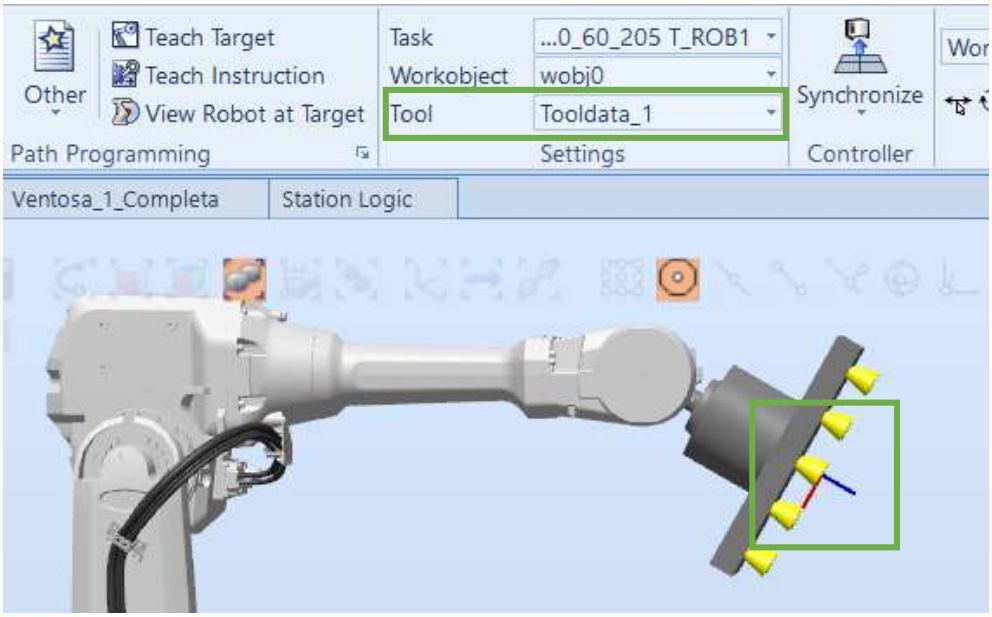


Figura 68 - Robô com dados da ferramenta corretos. Fonte: (Própria)

Para criar uma garra munida de movimento, como por exemplo uma garra mecânica, o processo é bastante semelhante ao da criação da ventosa, porem com algumas alterações e adições. A primeira alteração a efetuar é não criar um grupo de componentes desta garra, de acordo com a Figura 53. Isto deve-se ao facto de ser necessário atribuir para a base da garra e para cada dedo parâmetros importantes, que não poderiam ser atribuídos se estes elementos fizessem parte de um grupo de componentes. Nota-se que mesmo não pertencendo ao mesmo grupo de componentes, os elementos da garra devem estar colocados na posição de abertura ou de fecho, tal como monstra a Figura 69.

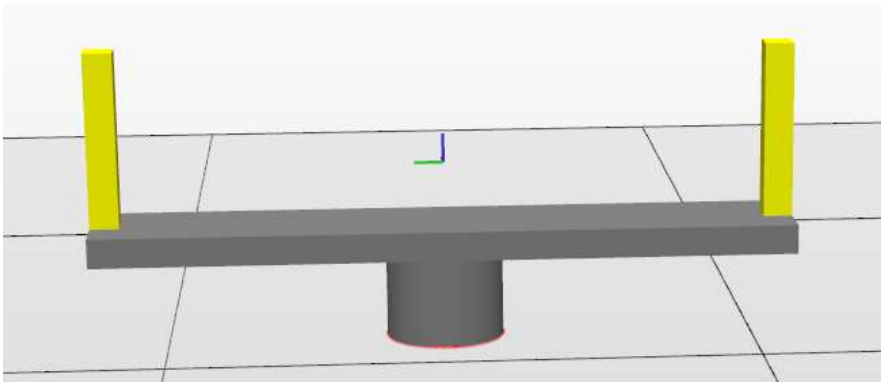


Figura 69 - Corpo sólido de nova garra mecânica. Fonte: (Própria)

Após criar os objetos sólidos da garra, em vez de se criar a garra através da opção "*Create Tool*", seleciona-se a opção "*Create Mechanism*" presente na barra de tarefas "*Modeling*". Nesta nova janela observada na Figura 70, atribui-se o nome à nova garra e seleciona-se a opção "*Tool*" em "*Mechanism Type*".

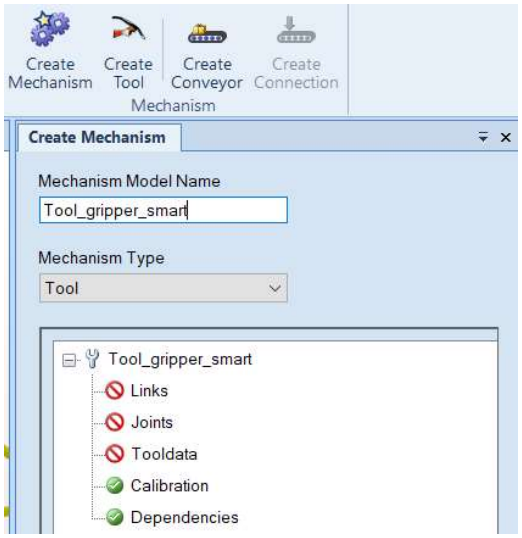


Figura 70 - Criar garra com movimento de dois dedos. Fonte: (Própria)

Existem três parâmetros a preencher antes de compilar a nova garra, os "*Links*", "*Joints*" e a "*Tooldata*". É devido a estes parâmetros que a criação do corpo sólido desta garra não deve ser feita dentro de um grupo de componentes, pois esse grupo é criado automaticamente através da seleção de componentes separados nos parâmetros "*Links*". Fazendo duplo clique na opção "*Links*", aparece uma nova janela onde se deve selecionar o objeto correspondente à base da garra em "*Selected Component*" e clicar no visto da opção "*Set as BaseLink*", conforme a Figura 71.

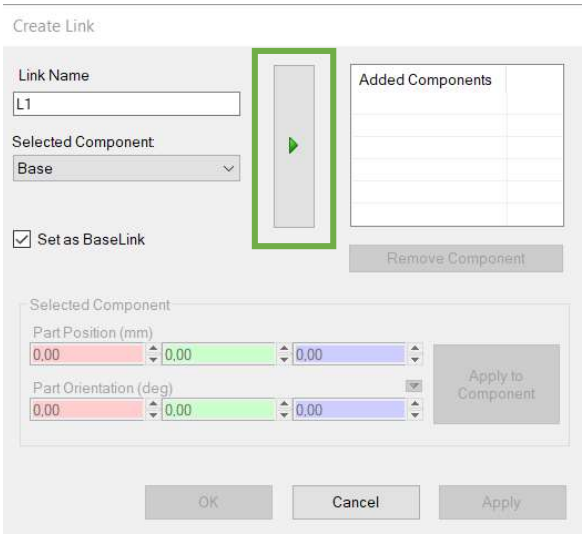


Figura 71 - Criação do link base da garra. Fonte: (Própria)

Procede-se em clicar no triângulo verde realçado acima e na opção "*Apply*". Depois executa-se os mesmos passos para os seguintes componentes da garra, clicando sempre em "*Apply*" quando se pretender criar um link. Nota-se que, como já foi criado o "*BaseLink*", esta opção não pode mais ser selecionada, tal como se pode observar na Figura 72.

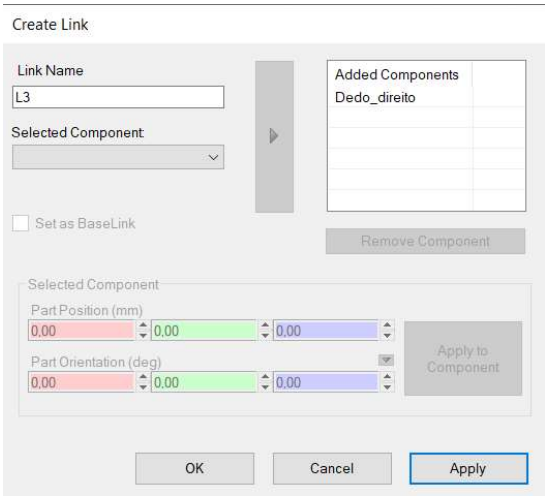


Figura 72 - Criação do último link da garra. Fonte: (Própria)

Após clicar em "*Apply*" e finalizar em "*Ok*", o próximo passo é a opção "*Joints*", responsável por interligar os componentes sólidos entre si. Neste caso, a garra pretendida faz com que os dedos se movam linearmente no sentido Y para abrir e fechar. É necessário selecionar o "*Parent Link*" e o "*Child Link*", sendo que o primeiro corresponde a base e o segundo ao dedo esquerdo da garra. Em "*Joint Type*" seleciona-se a opção "*Prismatic*" e preenche-se os restantes dados apresentados na Figura 73.

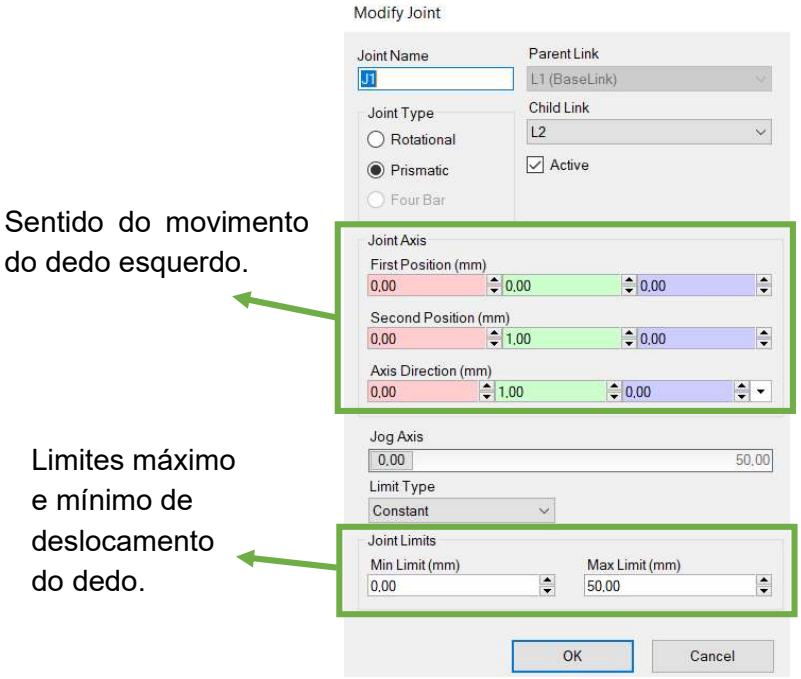


Figura 73 - Criação de juntas. Fonte: (Própria)

De acordo com a Figura 74, procede-se do mesmo modo para o segundo dedo, porém ter em atenção que o movimento deste é inverso em comparação ao primeiro dedo, sendo também necessário selecionar o "*Parent Link*" que é correspondente á base (L1), procedendo com a conclusão clicando em "*Ok*".

Sentido do movimento do dedo direito. Orientações invertidas em relação ao dedo anterior.

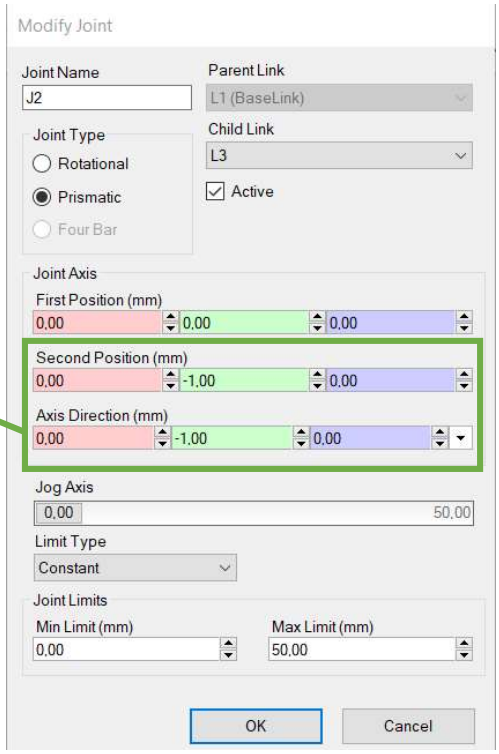


Figura 74 - Comparação de inversão com dedo anterior. Fonte: (Própria)

O último parâmetro a preencher é o do TCP, fazendo duplo clique na opção "*Tooldata*" atribui-se um nome, seleciona-se o link de base, pois o ponto pretendido é o centro da garra segundo o plano XY, adicionando um valor em Z que fique entre os dois dedos criados. Tal como executado na criação da ventosa, atribui-se uma massa e um centro de gravidade de modo a obter-se melhores resultados na simulação na janela presenta na Figura 75.

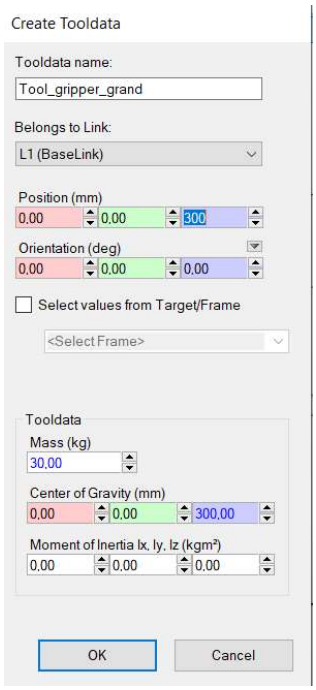


Figura 75 - Criação do TCP da garra. Fonte: (Própria)

Após clicar em "*Ok*" clica-se em "*Compile Mechanism*", o que desbloqueia opções de seleção de posições. As posições criadas são a de garra completamente aberta e completamente fechada, tal como mostra a Figura 76 e a Figura 77, respetivamente.

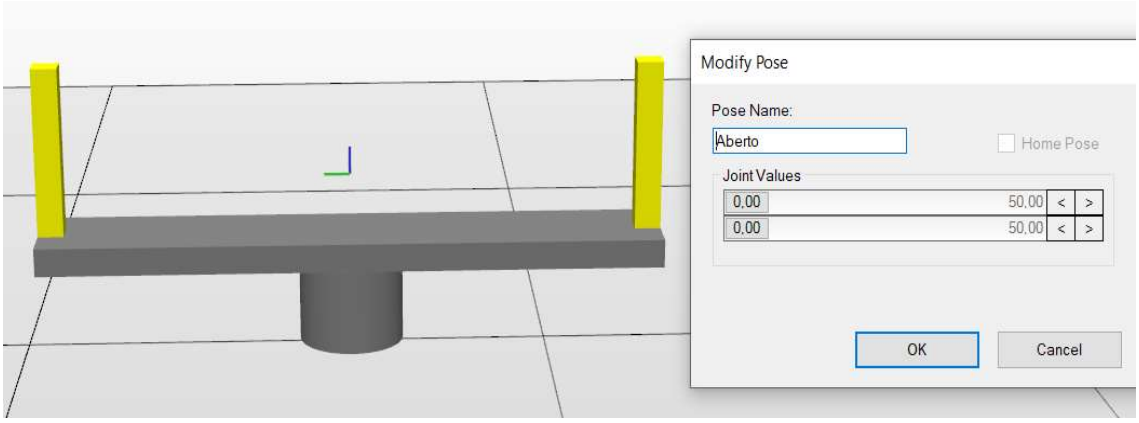


Figura 76 - Garra mecânica na posição "Aberto". Fonte: (Própria)

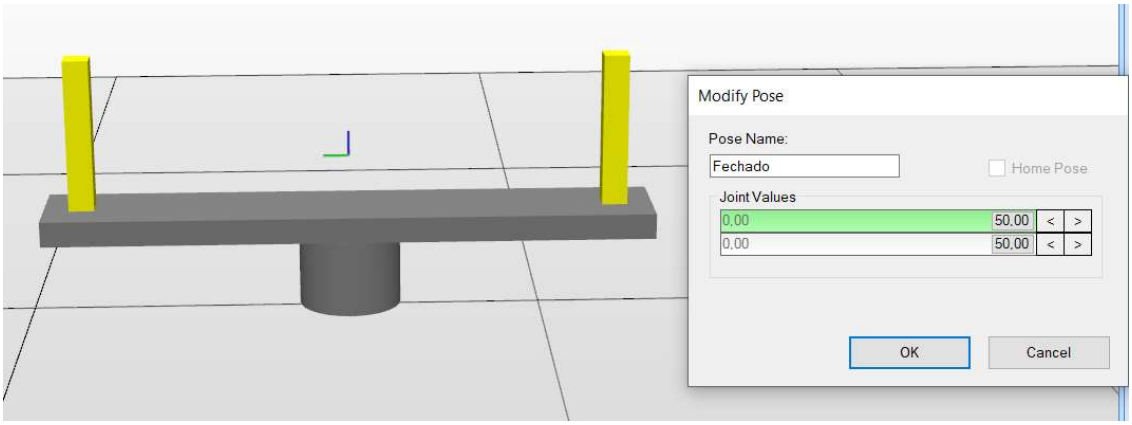


Figura 77 - Garra mecânica na posição "Fechado". Fonte: (Própria)

Uma vez criadas as posições de abertura e feche da garra, é necessário muni-la de inteligência através da criação de um "*Smart Component*", do mesmo modo que efetuado na ventosa, demonstrado na Figura 43 e da Figura 58 até à Figura 62.

Feito isto, é necessário efetuar a comunicação com o controlador, criando uma saída digital no controlador do robô, interligá-la com a entrada da garra e, de modo que a garra fique posicionada no robô, arrastar a garra criada para cima do robô na janela de esquemas e criar os dados da garra, seguindo os passos apresentados da Figura 63 até à Figura 68.

Deste modo, a garra criada fica corretamente posicionada e agarra o objeto em contacto com o sensor, porém esta não abre nem fecha as pinças quando acionada. Para aplicar este efeito visual, é necessário criar dois eventos no controlador, um para quando a saída de acionamento da garra estiver ativa e outro para quando este está desativo. Para aceder a ferramenta de criação de eventos, é necessário ir à barra de tarefas "*Simulation*" → "*Simulation Logic*" → "*Event Manager…"* ou, caso esta opção não exista, clicar no ícone presente na Figura 78.

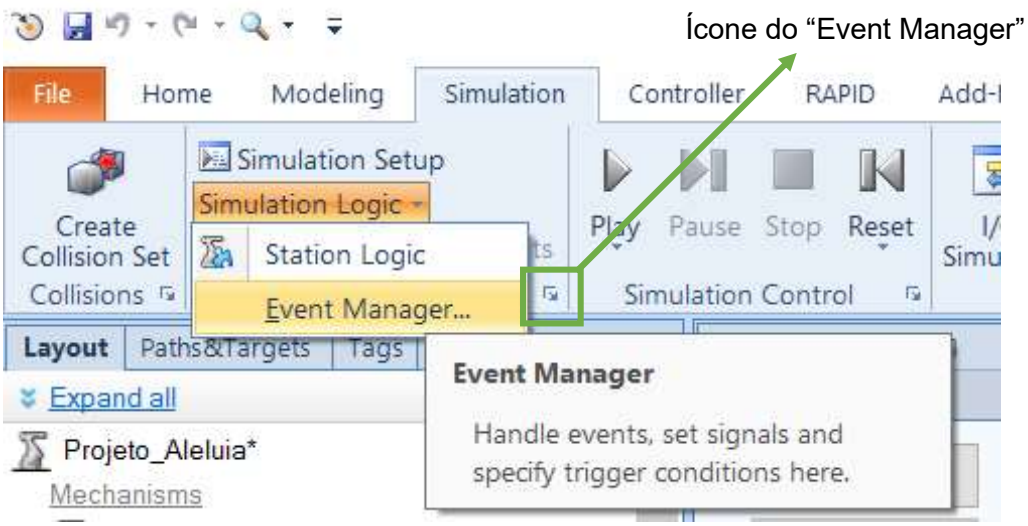


Figura 78 - Acesso ao "Event Manager". Fonte: (Própria)

Na janela “Create New *Event*”, mostrada na Figura 79, clica-se em “*ADD*” para adicionar um novo evento, mantem-se a opção “*Activation*” no modo “*ON*” e seleciona-se no “*Event Trig Type*” a opção “*I/O signals changed*”, criando deste modo um evento quando surge a alteração de um sinal.

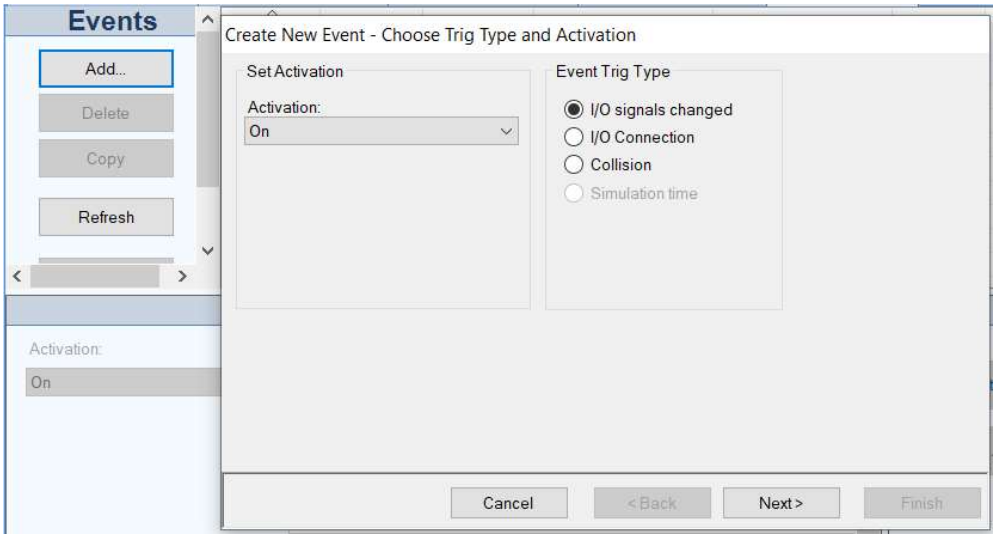
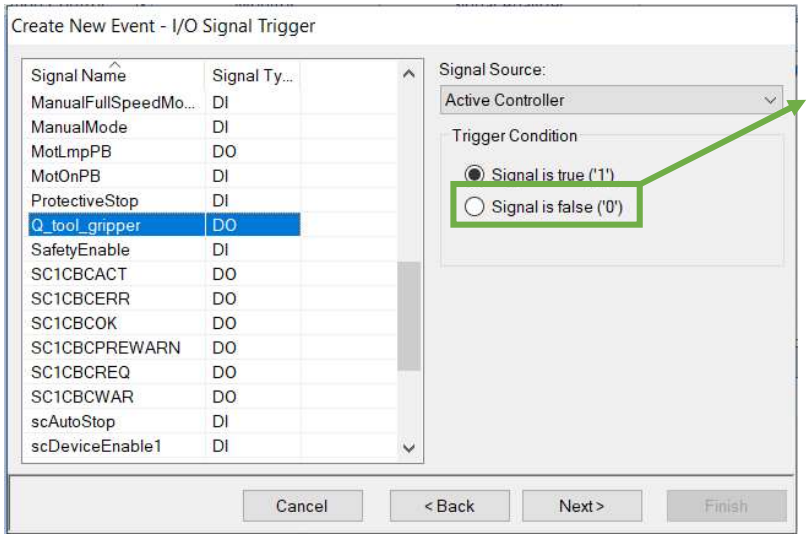


Figura 79 - Criação de evento de fecho de garra (parte 1). Fonte: (Própria)

De seguida é necessário selecionar o controlador e a saída responsável pela ativação da garra, onde neste caso o controlador é “*Active Controller*” e o nome da saída atribuído é “*Q_tool_gripper*”. A “*Trigger Condition*” será escolhida conforme o fim desejado para este evento. Para fechar a garra, o sinal será “*True*” (1) e para a abrir “*False*” (0), tal como é possível observar na Figura 80.

As seguintes figuras mostram a criação do evento de fecho da garra, porém realçam a opção que se deve efetuar para, após a conclusão desta, criar o evento de abertura da garra.



Selecionar para evento de garra aberta.

Figura 80 - Criação de evento de feche de garra (parte 2). Fonte: (Própria)

Após clicar no botão "*Next*" aparece uma janela com a opção "*Set Action Type*", seleciona-se a opção "*Move Mechanism to Pose*", presente na Figura 81. Após clicar em "*Next*", seleciona-se o nome correspondente à garra em "*Mechanism*" e o nome da posição em "*Pose*", tal como mostra a Figura 82.

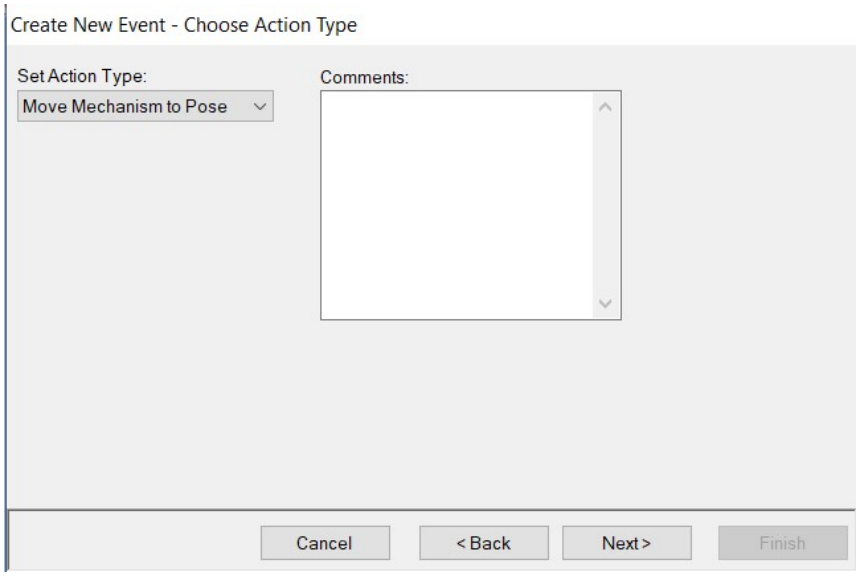


Figura 81 - Criação de evento de feche de garra (parte 3). Fonte: (Própria)

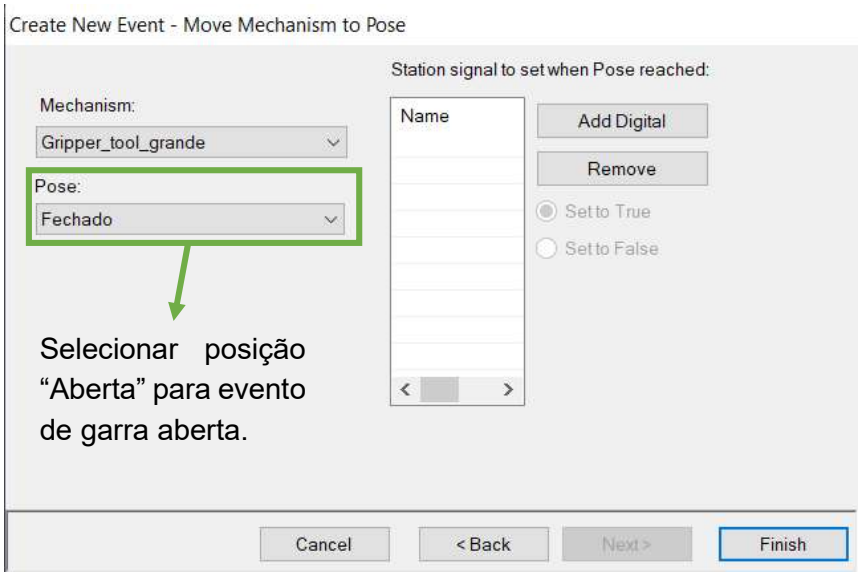


Figura 82 - Criação de evento de feche de garra (parte 4). Fonte: (Própria)

Depois de clicar em "*Finish*", cria-se outro evento do mesmo modo, porém relembrando mais uma vez, com a efetuação das alterações realçadas na Figura 80 e na Figura 82, obtendo assim os dois eventos observados na Figura 83.

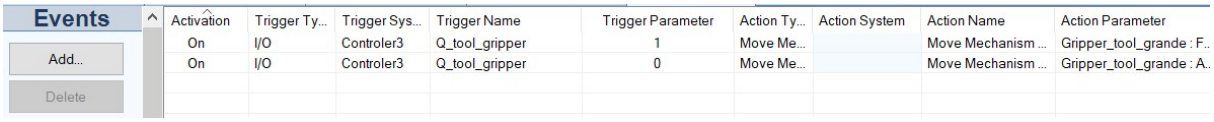


Figura 83 - Eventos criados. Fonte: (Própria)

3.1.5 - Manipulação do robô

Este tópico aborda um conjunto de ferramentas importantes para a manipulação e criação de pontos "Target" do robô, utilizando o cenário mostrado na Figura 84. Informação mais detalhada sobre a linguagem de programação e das suas funcionalidades encontra-se no Anexo 3.

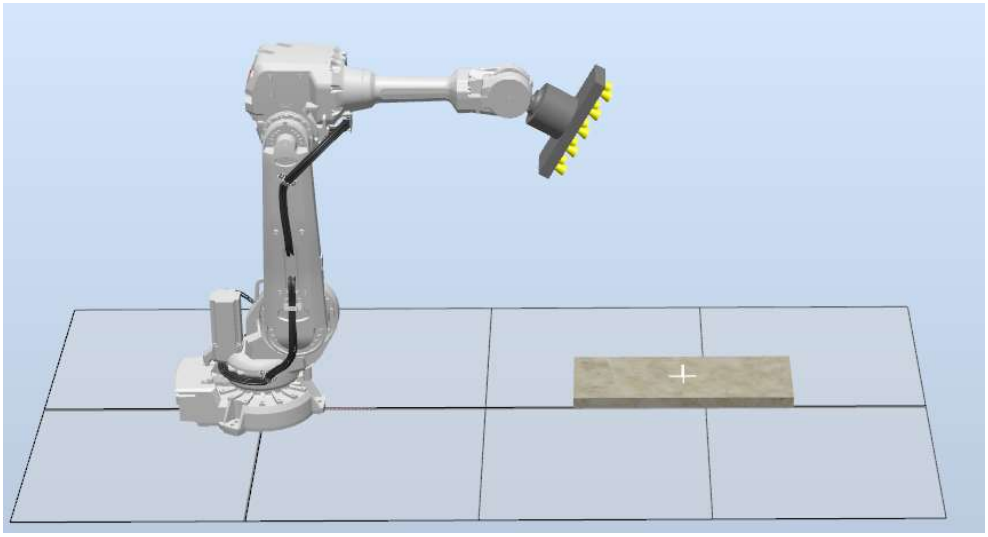


Figura 84 - Cenário de interação de robô com objeto. Fonte: (Própria)

É possível manipular o robô através da atribuição de valores de ângulos de juntas ou da atribuição de um ponto espacial e a sua respetiva orientação. Para tal, na janela "*Layout*", seleciona-se o robô com o botão esquerdo do rato, abrindo assim uma nova janela na barra de tarefas "*Mechanism Tools*" →"*Modify*" → "*Motion*", tal como é possível observar na Figura 85.

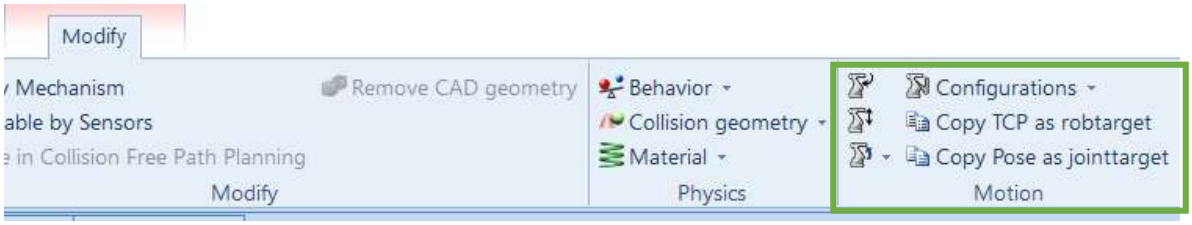


Figura 85 - Separador "Mechanism Tools". Fonte: (Própria)

Nesta secção é possível ver os seguintes ícones:

 Mechanism Joint Jog – Abre uma janela onde é possível alterar os valores de cada junta do robô (Ver Figura 86).

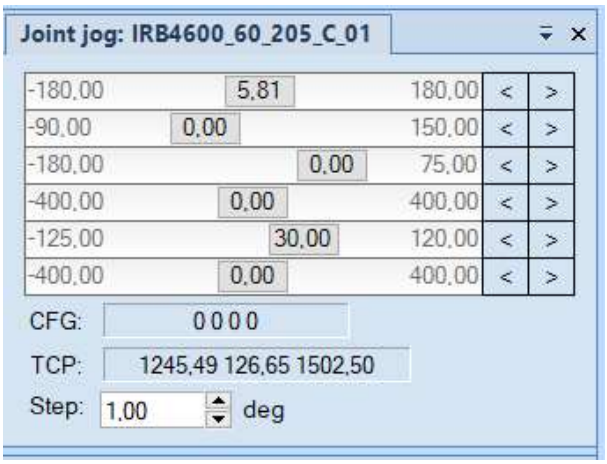


Figura 86 - Janela "Joint jog". Fonte: (Própria)

 Mechanism Linear Jog – Abre uma janela onde é possível alterar os valores do dos eixos e das suas orientações do robô (Ver Figura 87).

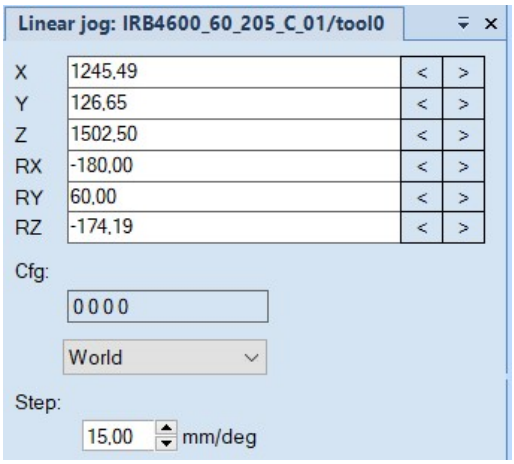


Figura 87 - Janela "Linear Jog". Fonte: (Própria)

É através da manipulação dos campos destas duas janelas que é possível manipular o robô através de coordenadas ou valores de ângulos de juntas. Este método, porém, é bastante trabalhoso e demorado na realização de um programa extenso.

Quando o objeto a manipular já está corretamente posicionado no simulador, é possível selecionar um ponto da superfície deste objeto e declará-lo como um alvo do robô. Este alvo pode ser acedido pelo robô quando o utilizador necessitar.

Para criar um alvo, é necessário ter em atenção a seleção correta do controlador, do objeto de trabalho e da ferramenta, presentes na barra de tarefas "*Home*" → "*Settings*", de acordo com a Figura 88.

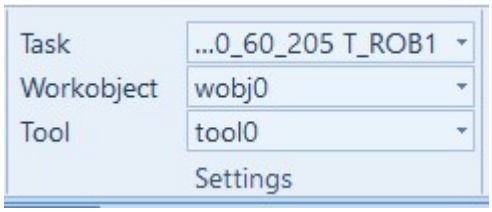


Figura 88 - Controlador, objeto de trabalho e ferramenta. Fonte: (Própria)

Selecionando a opção na barra de tarefas "*Home*" → "*Path Programming*" → "*Target*" → "*Create Target*" é possível selecionar um ponto no mundo, ou introduzir manualmente as coordenadas desejadas para o robô alcançar. É também possível adicionar vários pontos, clicando em "*ADD*" para cada adição e, para finalizar, clica-se em "*Create*", tal como é possível observar na Figura 89.

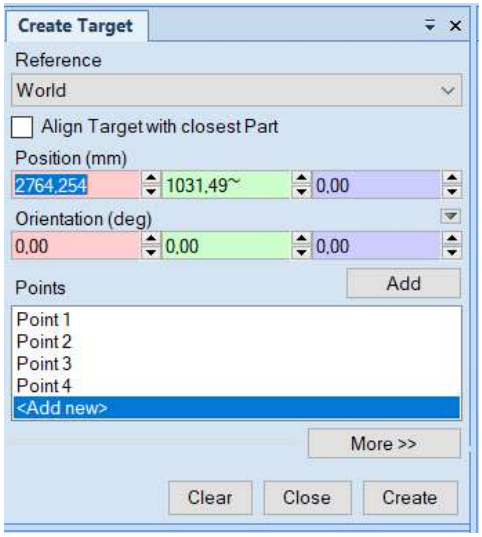


Figura 89 - Pontos de alvo para robô. Fonte: (Própria)

Para selecionar um canto de um objeto, centro de uma superfície, entre outros, existem ferramentas de auxílio de seleção na janela da vista, conforme a Figura 90.

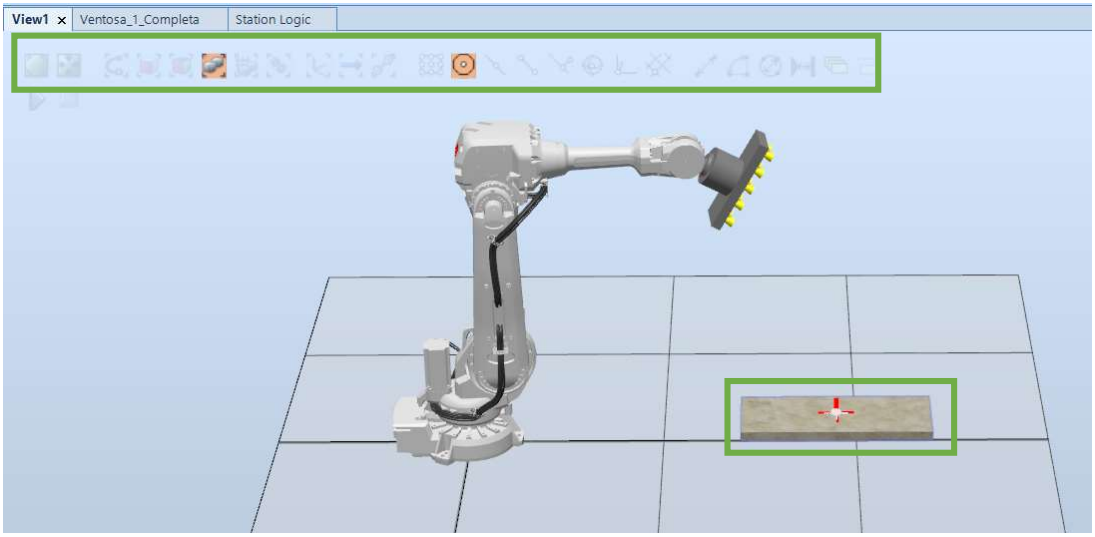


Figura 90 - Ferramentas de auxílio de seleção e alvo. Fonte: (Própria)

Tendo o ponto criado, é possível observá-lo na janela "*Paths&Targets*" na lateral esquerda dentro de "*Nome_do_robô*" →"*T_ROB_nº_do_robô*" → "*Workobjects & Targets*" → "*wobj0*" → "*wobj0_of*". Selecionando o alvo com o botão direito do rato, é possível observar um clone da ferramenta sobre o alvo selecionando "*View Tool at Target*" e escolhendo a ferramenta.

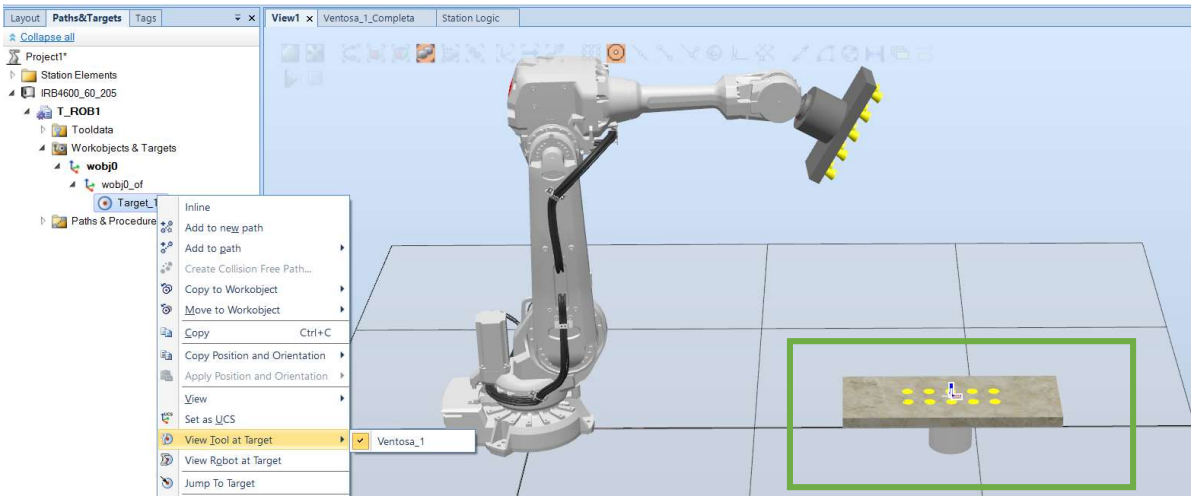


Figura 91 - Ferramenta sobre alvo. Fonte: (Própria)

Também é possível observar o robô no alvo, selecionando a opção "View Robot at Target". Estas duas opções podem ser canceladas, selecionando-as de novo. Como é possível verificar na Figura 91 e na Figura 92, a orientação da ferramenta está errada e, apesar de na Figura 92 a opção "View Robot at Target" estar ativa, o robô não se posiciona no alvo, devendo-se isto ao facto do robô não conseguir alcançar este alvo segundo esta orientação da ferramenta, sendo necessário reajustá-la.

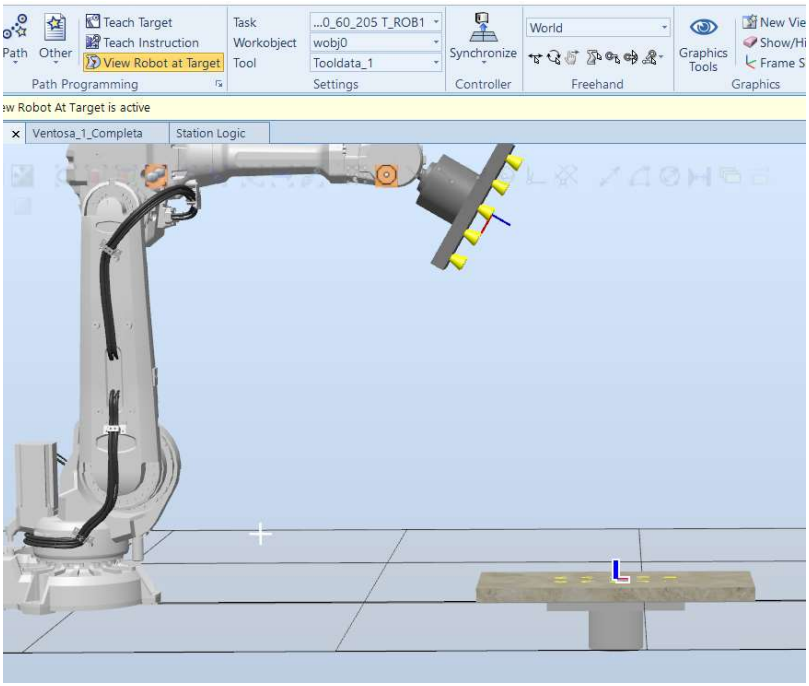


Figura 92 - Robô sem alcance ao ponto alvo selecionado. Fonte: (Própria)

Para ajustar a rotação do alvo, basta clicar no "*Target*" com o botão direito do rato e aceder a "*Modify Target*" → "*Rotate*", abrindo a janela presente na Figura 93.

Existem várias formas de alterar a rotação do eixo nesta janela, porém a recomendada é acrescentar um número em "*Rotation(deg)*", e selecionar um eixo pelo qual a ferramenta irá rodar, neste caso 180 graus em X.

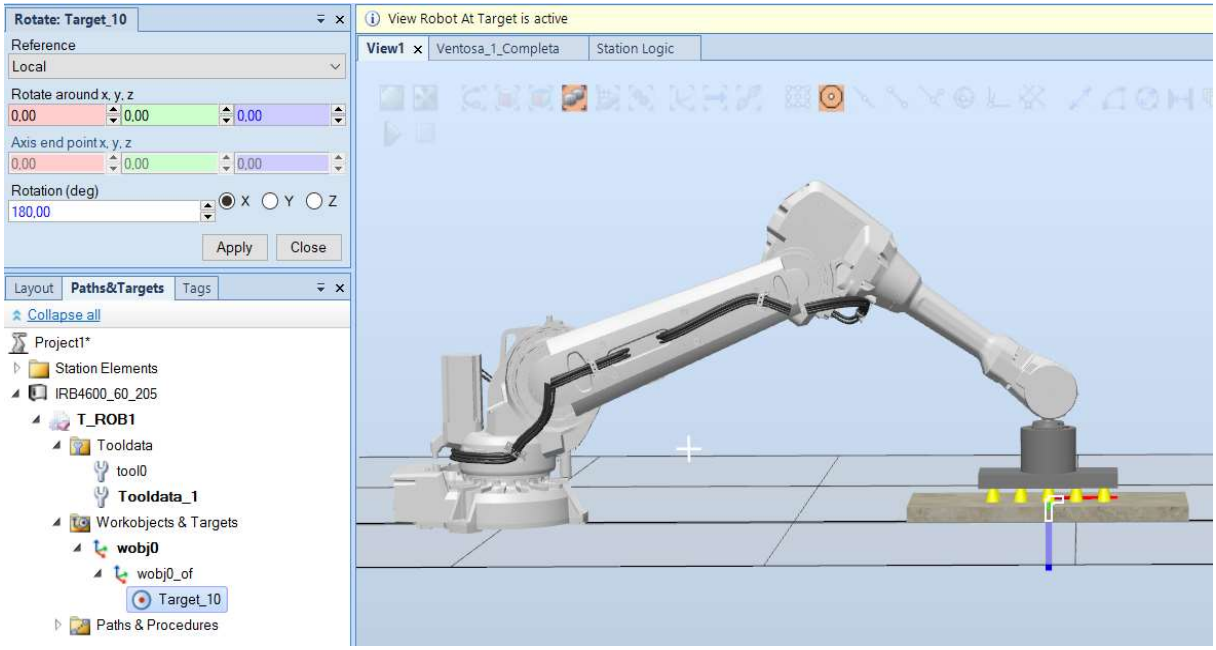


Figura 93 - Robô posicionado corretamente no alvo. Fonte: (Própria)

3.1.6 - Criação de procedimentos e adição de alvos ao controlador

Para adicionar este alvo a um programa do robô, é necessário criar um "*Path*" com o nome "*main*", que poderá ser o único procedimento presente no controlador, ou pode chamar vários procedimentos. Para tal seleciona-se na barra de tarefas "*Home*" → "*Path*" → "*Empty Path*", que cria um procedimento chamado "*Path_10*". Para o renomear, seleciona-se com o botão direito do rato o procedimento criado e "*Rename*".

Após criado o "*Path*", arrasta-se o alvo para cima dele, alterando as propriedades apresentadas na Figura 94, segundo o tipo de movimento, velocidade e margem desejados.

Figura 94 - Parâmetros do movimento. Fonte: (Própria)

Como é possível observar na Figura 95, a adição dos alvos "*Target_10*" e "*Target_20*" apresentam problemas, identificados pelos seguintes símbolos:

 Robô não consegue alcançar o ponto alvo com a configuração atual. Nova configuração necessária.

 Robô não consegue alcançar o ponto alvo. Apagar alvo ou reposicioná-lo.

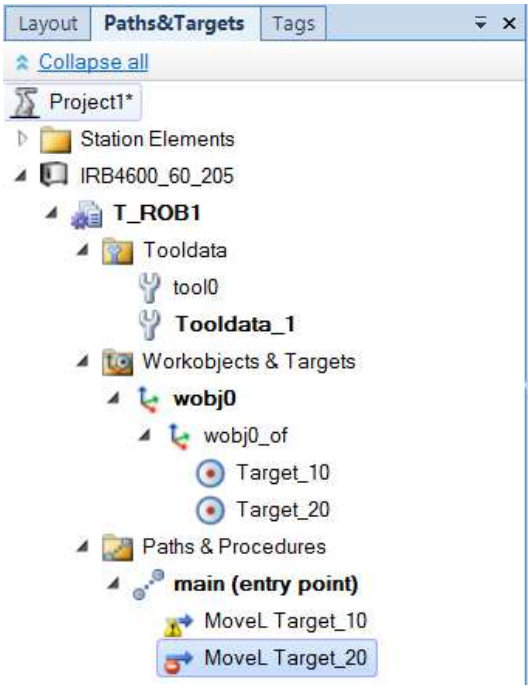


Figura 95 - Alvos no procedimento "main". Fonte: (Própria)

Para configurar de forma automática os movimentos do procedimento é possível aceder em “Paths&Targets” → “Paths & Procedures” ao procedimento “main” com o botão direito do rato e aceder a “Auto Configuration” → “ALL move instructions”, tal como se observa na Figura 96.

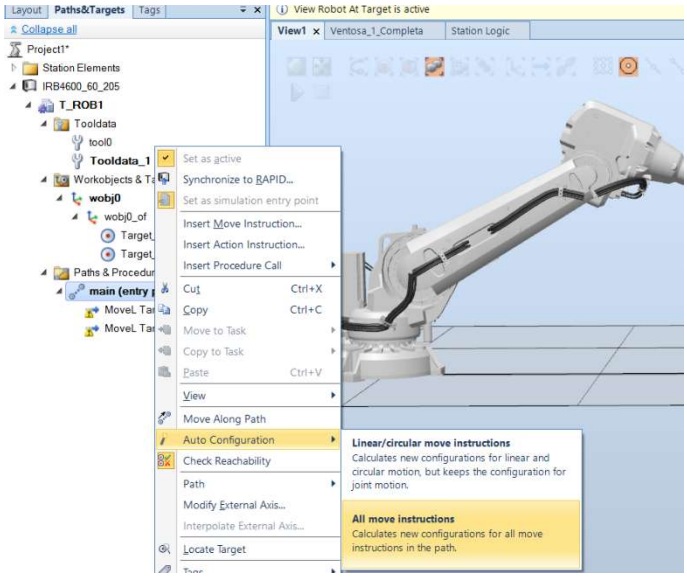


Figura 96 – Configuração automática das posições do robô. Fonte: (Própria)

De seguida, é aberta a janela presente na Figura 97. Esta janela apresenta as várias configurações do robô, onde nas que apresentam problemas de alcance, aparece o símbolo de alerta.

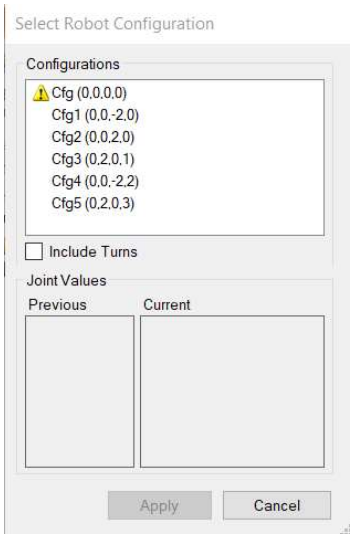


Figura 97 - Configurações do robô. Fonte: (Própria)

Na seleção da configuração, é recomendado escolher a que apresenta mais valores “zero” nos parâmetros imediatamente a frente do nome (Ver Figura 98 e Figura 99).

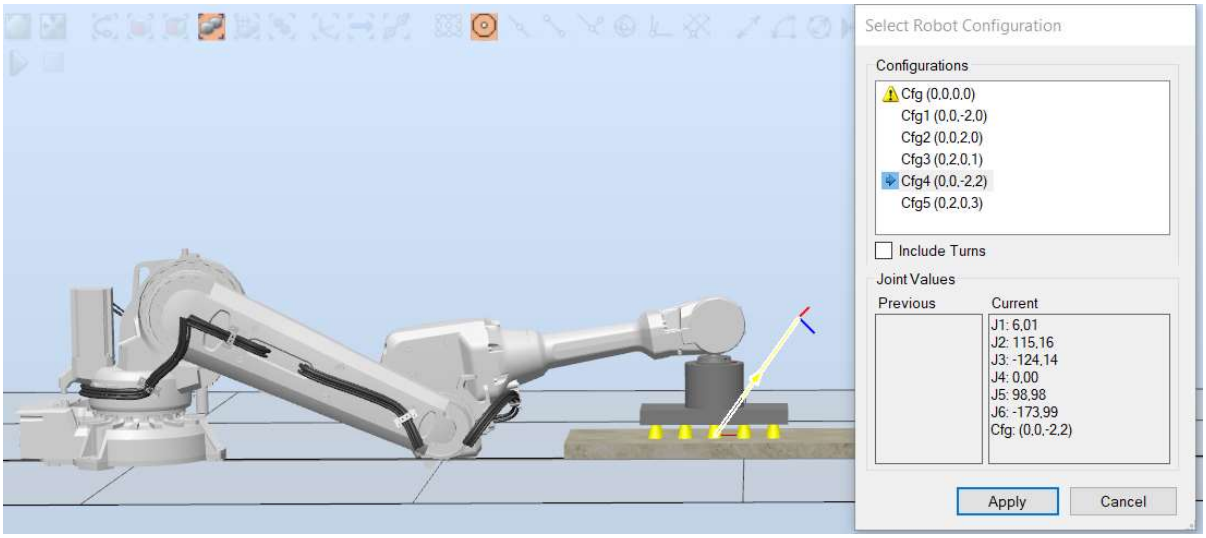


Figura 98 - Configuração do robô (não recomendada). Fonte: (Própria)

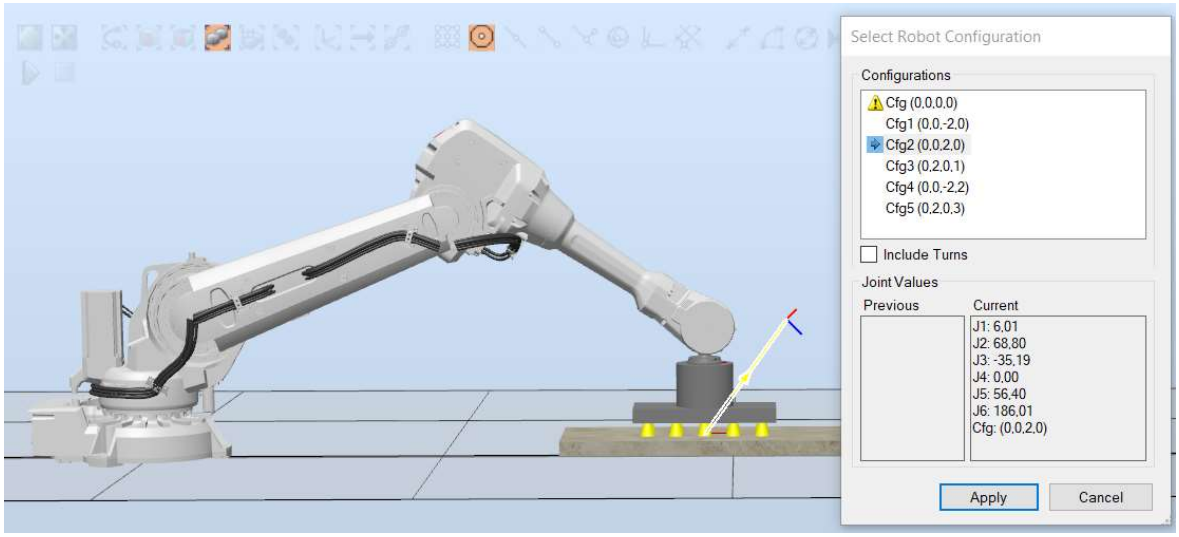


Figura 99 - Configuração do robô (recomendada). Fonte: (Própria)

Após aplicar a configuração desejada, o procedimento irá apresentar o aspeto apresentado na Figura 100.

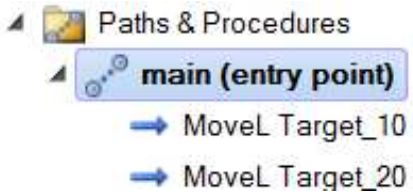


Figura 100 - Procedimento configurado. Fonte: (Própria)

Após serem criados vários movimentos, é necessário guardá-los no controlador. Para tal na barra de tarefas “Home” →”Controller” → “Synchronize” existem duas opções, observadas na Figura 101. A primeira é a “Synchronize to RAPID” que transfere os procedimentos e alvos criados pelo método anterior para a consola, convertendo em código RAPID. A segunda “Synchronize to Station” transfere o Código criado em RAPID para a estação.

Figura 101 - Sincronização de controlador. Fonte: (Própria)

Selecionando uma das sincronizações, coloca-se um visto em todas as opções visíveis e pressiona-se “OK”.

Nota-se que sempre que se cria movimentos ou envios de sinais, deve-se sincronizar a estação com o código RAPID, e vice-versa, dependendo de como se programou. Ao ser efetuado a sincronização, um dos métodos de programação (estação ou RAPID) será totalmente substituído pelo outro, no que se deve tomar especial atenção neste passo.

3.1.7 - Lógica de sinais

Para o robô enviar um sinal, ou executar uma espera, é possível adicionar uma linha de código, clicando com o botão direito do rato sobre o procedimento, e aceder a "Insert Action Instruction…", abrindo a janela presente na Figura 102.

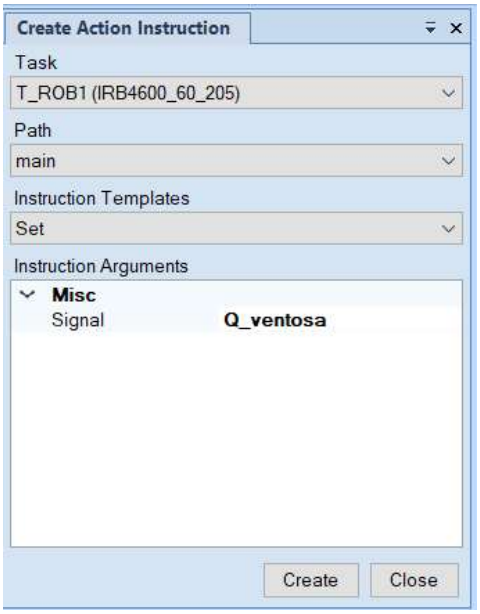


Figura 102 - Criação de ação. Fonte: (Própria)

Nesta janela, é possível selecionar o robô ao qual se vai adicionar a instrução no parâmetro "Task", o procedimento no parâmetro "Path" e o tipo de instrução no parâmetro "Instruction Templates", sendo que todos estes parâmetros são mostrados numa lista. No caso da Figura 102, está a ser criado um sinal "Set" da porta lógica da ventosa. Para escolher qual o sinal a modificar acede-se em "Instruction Arguments" e escolhe-se a saída pretendida da lista. Através deste método, criou-se o programa observado na Figura 103.

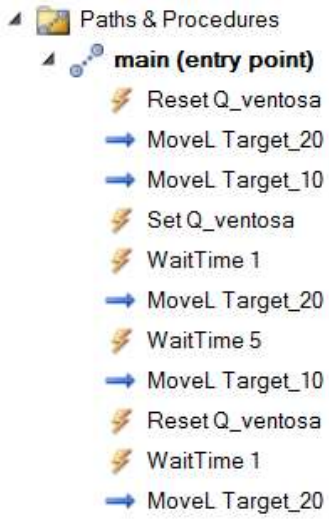


Figura 103 - Programa simples. Fonte: (Própria)

Após sincronizar para código RAPID na opção “Synchronize to RAPID” (ver Figura 101), acede-se na barra de tarefas “RAPID” para abrir uma janela lateral “Controller”, faz-se um duplo clique com o botão esquerdo do rato em cima do “main” e verifica-se que o procedimento criado aparece em código RAPID, de acordo com a Figura 104.

```
24      !***********************************************************
25      PROC main()
26          !Add your code here
27          Reset Q_ventosa;
28          MoveL Target_20,v500,z0,Tooldata_1\WObj:=wobj0;
29          MoveL Target_10,v500,fine,Tooldata_1\WObj:=wobj0;
30          Set Q_ventosa;
31          WaitTime 1;
32          MoveL Target_20,v500,z0,Tooldata_1\WObj:=wobj0;
33          WaitTime 5;
34          MoveL Target_10,v500,fine,Tooldata_1\WObj:=wobj0;
35          Reset Q_ventosa;
36          WaitTime 1;
37          MoveL Target_20,v500,z0,Tooldata_1\WObj:=wobj0;
38      ENDPROC
39  ENDMODULE
```

Figura 104 - Procedimento criado sincronizado para RAPID. Fonte: (Própria)

Qualquer atualização escrita nesta janela deve ser aplicada na barra de tarefas “RAPID” → “Apply” → “Apply ALL”, tal como mostra a Figura 105, para a colocar no controlador e, de seguida, executar a sincronização “Synchronize to Station”.

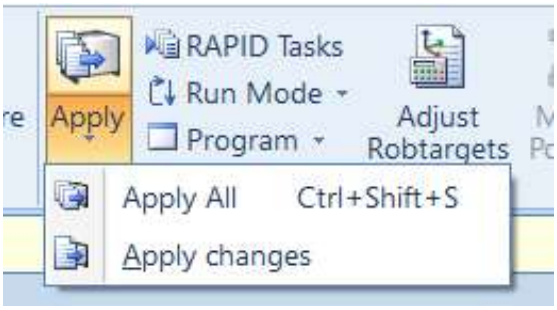


Figura 105 - Aplicar mudanças no código RAPID. Fonte: (Própria)

3.1.8 - Ferramentas de simulação

O simulador é uma ferramenta importante para a atribuição visual ao código criado. Apesar de ser uma ferramenta utilizada para criar vídeos demonstrativos das réplicas digitais, é uma ferramenta bastante útil para identificar erros cometidos durante a criação da mesma.

Após a criação de um programa, ou sua parte, devidamente sincronizado, é necessário criar um estado de simulação inicial para o programa retomar após uma simulação, uma vez que os objetos manipulados e os estados de sinais no fim da simulação podem não coincidir com os iniciais. Para tal acede-se na barra de tarefas "*Simulation*" → "Reset" → "*Save Current State…*", abrindo a janela exposta na Figura 106 na qual é possível adicionar o nome ao estado, uma descrição e escolher quais os estados, valores e projetos a ser guardados. Por norma, seleciona-se todos os projetos com um visto.

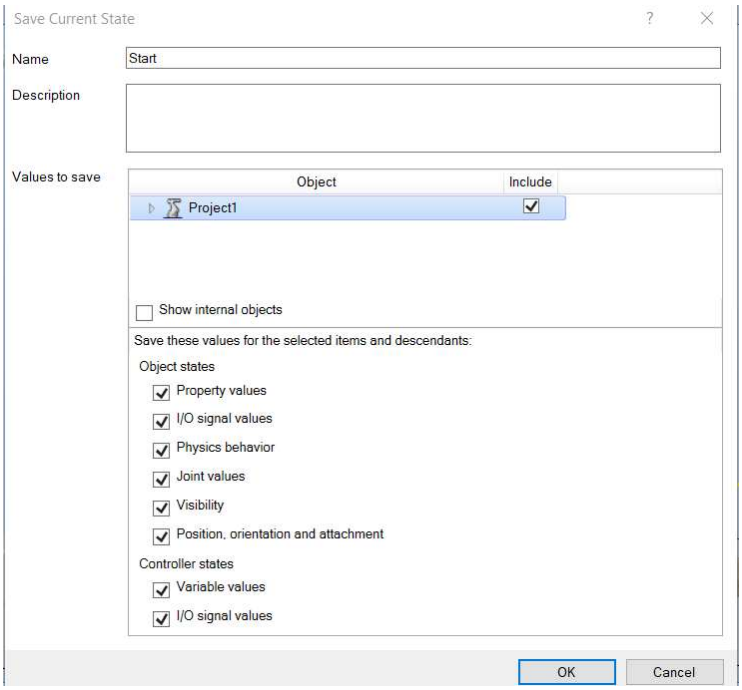


Figura 106 - Guardar estado atual. Fonte: (Própria)

Após clicar em “OK”, é possível visualizar o novo estado criado, acedendo mais uma vez na barra de tarefas “Simulation” → “Reset”, tal como se pode observar na Figura 107. Com o estado inicial guardado, é possível dar início á simulação criada através do ícone “Play” situado na mesma barra de tarefas, mantendo o ambiente de início guardado em segurança.

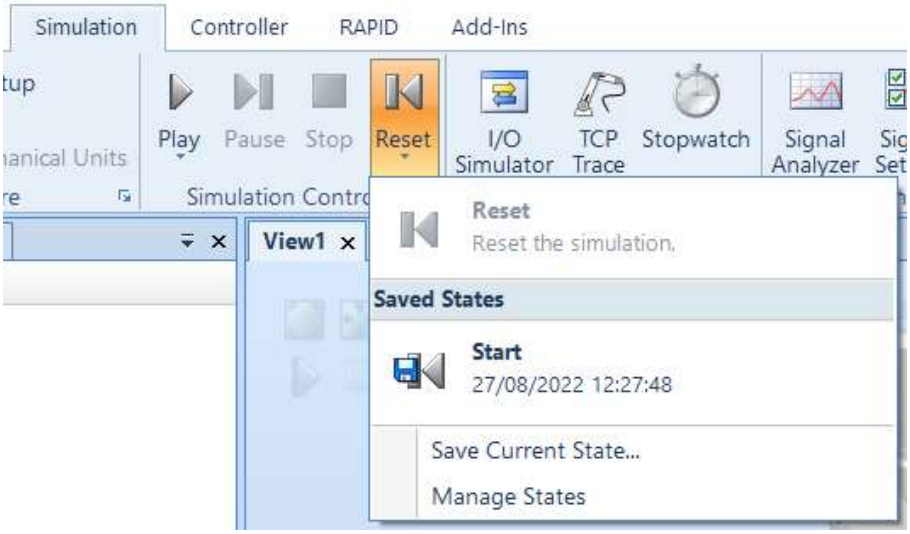


Figura 107 - Barra de tarefas do simulador com estado "START" criado. Fonte: (Própria)

Após clicar no “Play”, é possível observar a réplica criada em execução, conforme a Figura 108, e, no caso da sua existência, a existência de erros. Nota-se que após a finalização do vídeo, é recomendado selecionar o estado inicial guardado duas ou mais vezes seguidas, devido a um “bug” de estados de objetos e de leitura de sensores.

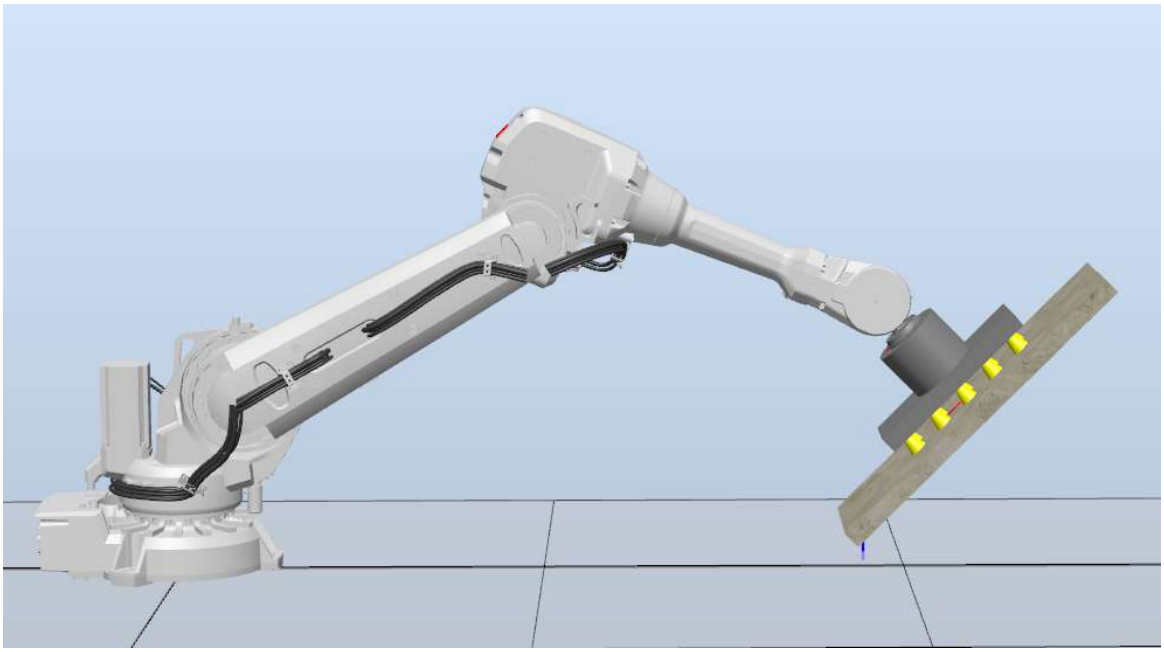


Figura 108 - Robô a agarrar azulejo na simulação. Fonte: (Própria)

3.1.9 - Importar objetos e criar tapete transportador

Criado o robô, o próximo passo é criar o ambiente industrial envolvente. Numa primeira fase serão importados elementos como a palete onde irão estar os azulejos inicialmente e o tapete transportador que os leva para o próximo robô. Para dar movimento ao tapete transportador, existem vários métodos de o fazer, porém o método utilizado foi criar uma superfície invisível retangular em cima do tapete importado e atribuir-lhe movimento.

O primeiro passo na criação do ambiente, mostrado na Figura 109, é importar e posicionar uma palete para os azulejos. Para tal é necessário aceder na barra de tarefas "*Home*" → "*Import Library*" → "*Equipment*" e procurar o objeto "*Euro Pallet*". Coloca-la na posição pretendida, clicando com o botão direito do rato sobre ela e selecionar "*Position*" → "*Set Position*".

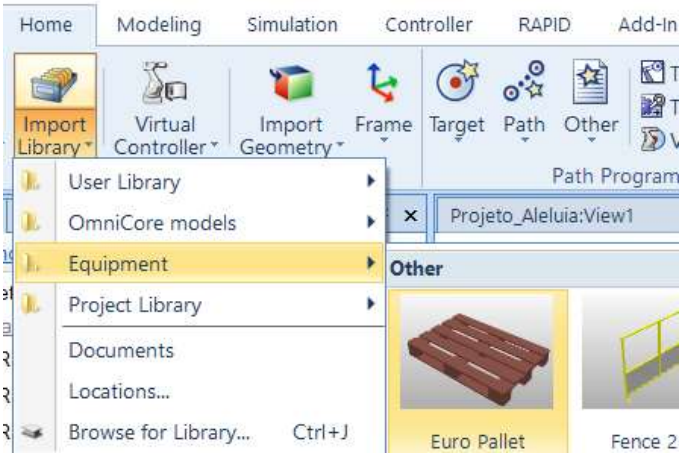


Figura 109 - Inserir palete no ambiente. Fonte: (Própria)

Em seguida, de acordo com a Figura 110, cria-se um grupo de componentes para inserir todos os azulejos da simulação, de modo a ser mais fácil e prático identificá-los na janela de esquemas e movimentar como conjunto no case de ocorrência de alterações futuras. Nota-se que na montagem da paletização, por motivos referidos mais a frente, é importante colocar os azulejos imediatamente em cima da palete e sem estes se encontrarem dentro uns dos outros.

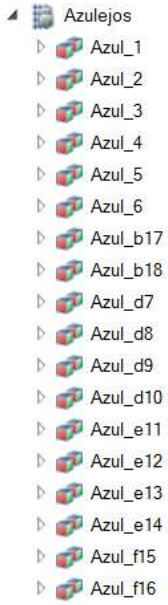


Figura 110 - Grupo de azulejos criados. Fonte: (Própria)

Assim sendo, a partir do primeiro azulejo criado, criaram-se no total dezoito azulejos, identificados de 1-18 e com letras representativas em alguns para melhor identificação dos mesmos. De seguida reposicionou-se estes azulejos de modo a preencher a palete conforme a Figura 111.

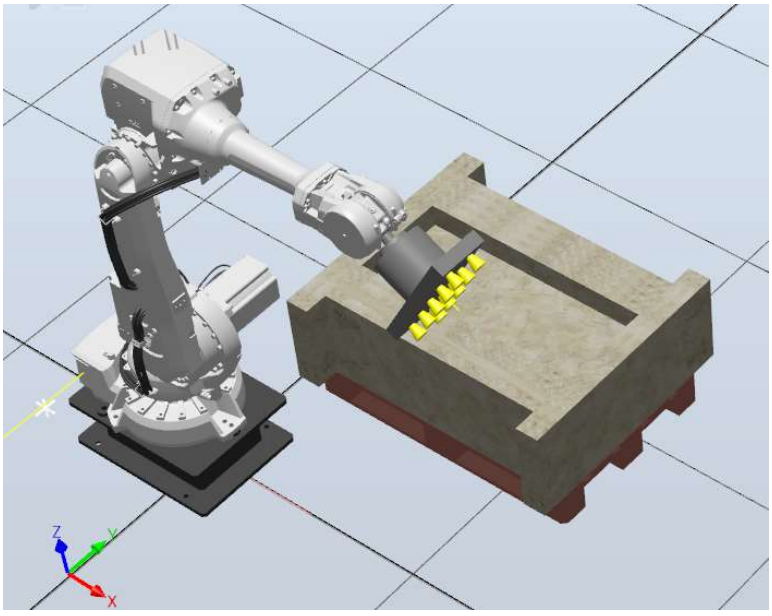


Figura 111 - Posicionamento da paletização. Fonte: (Própria)

O próximo passo é importar o tapete transportador e criar um sólido retangular que cubra a sua face superior. Para importar o tapete transportador procede-se do mesmo modo que na importação da palete, porém seleciona-se o componente "*Conveyor*". Depois de o posicionar no devido local, cria-se um sólido retangular e posiciona-se por cima do tapete, cobrindo-o de uma ponta à outra, tal como se pode observar na Figura 112.

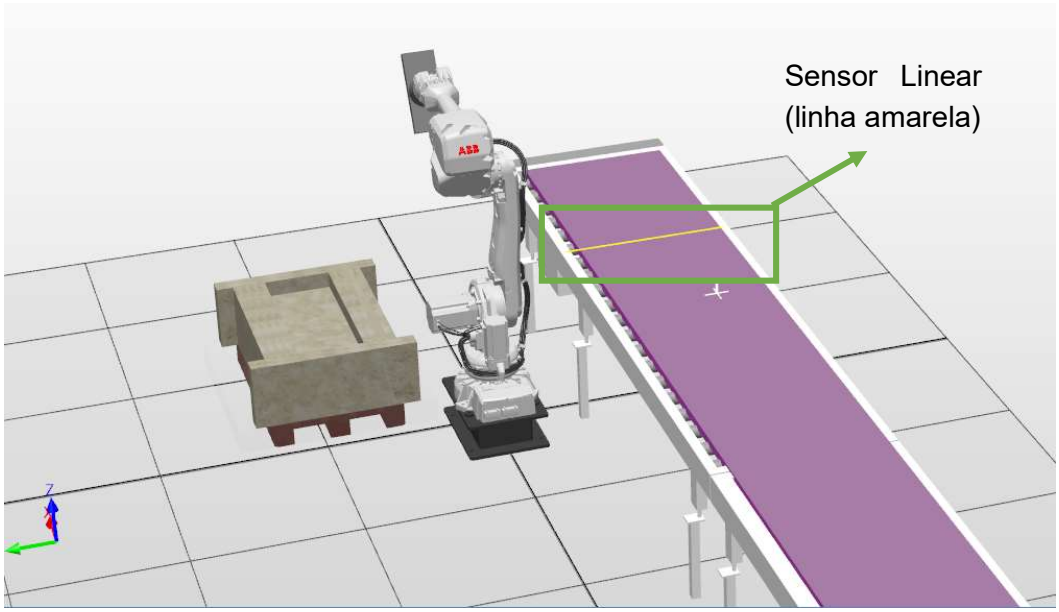


Figura 112 - Tapete transportador. Fonte: (Própria)

Para atribuir movimento à sua superfície, basta selecioná-lo com o botão direito do rato e escolher as opções "*Physics*" → "*Behavior*" → "*Fixed*" de forma a este sólido interagir com outros sólidos, porém estar fixo no ambiente e selecionar a opção "*Physics*" → "*Surface Velocity*" e atribuir uma velocidade e direção de velocidade, de acordo com a Figura 113.

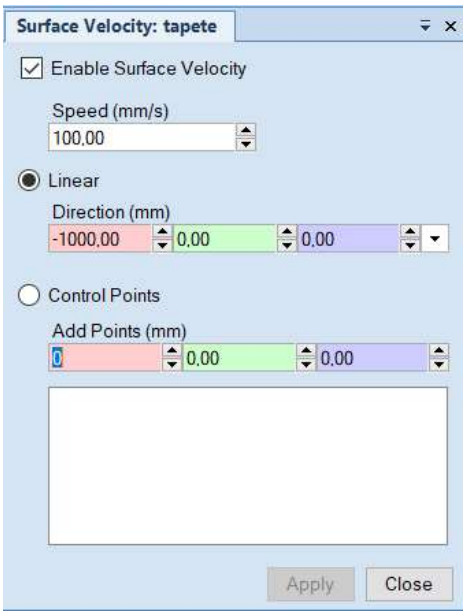


Figura 113 - Atribuição de velocidade na superfície de um sólido. Fonte: (Própria)

Para dar segurança ao posicionamento dos azulejos, criou-se um "*Smart Component*" onde se inseriu o tapete transportador e um sensor linear ligado a um "*Output*", tal como mostra a Figura 114, posicionando-o de acordo com a Figura 112, num local onde seja seguro colocar um azulejo quando o sensor não deteta outro azulejo.

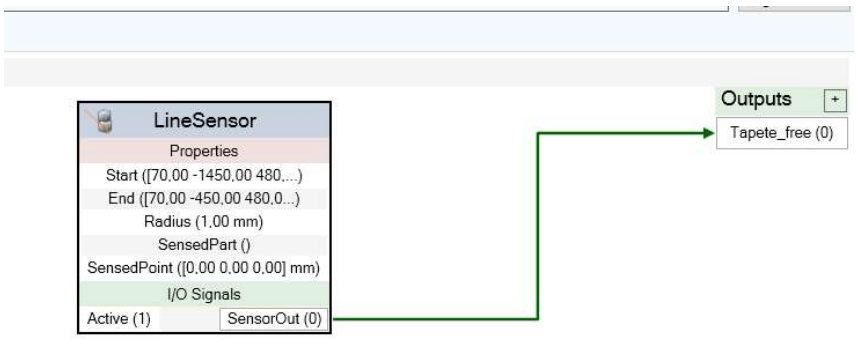


Figura 114 - Sensor linear ligado a "Output". Fonte: (Própria)

Para os azulejos interagirem com o tapete quando entrarem em contacto com este, basta selecioná-los com o botão direito do rato e escolher as opções "*Physics*" → "*Behavior*" → "*Dynamic*" de forma a este sólido interagir com outros sólidos e com a gravidade. É devido a esta funcionalidade que os azulejos não devem encontrar-se dentro uns dos outros, pois caso isto aconteça, ao clicar-se no "*Play*" do simulador, estes são projetados em direções aleatórias até cair.

3.1.10 - Adição de novos controladores

No simulador RobotStudio, é possível controlar vários robôs com apenas um controlador, porém isto não é o que se apresenta num ambiente fabril, até porque os robôs escolhidos são de grande porte e estarão bastante afastados uns dos outros. Para tal optou-se pela criação de um controlador por robô.

Para adicionar um novo controlador é necessário aceder na barra de tarefas "Home" as opções "*Virtual Controller*" →"*New Controller*" e preencher os campos do mesmo modo que na criação do primeiro controlador, de modo a obter-se um mesmo modelo do primeiro robô, de acordo com a Figura 115. A atribuição de nomes identificativos de fácil compreensão e distinção ajuda na programação dos controladores numa fase final da completação da simulação.

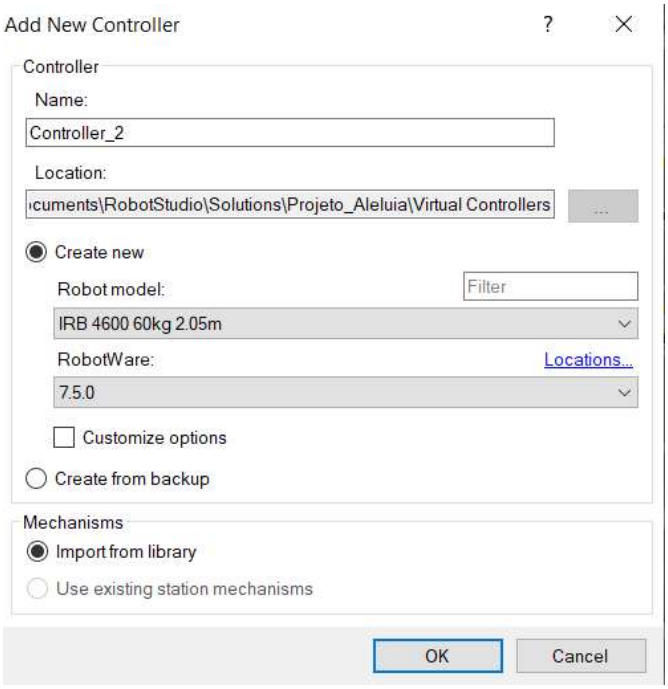


Figura 115 - Adicionar novo controlador. Fonte: (Própria)

Após clicar em "*OK*", aguarda-se que o programa o compile e assim está criado o segundo controlador com o robô pretendido. De seguida posiciona-se o robô no local desejado, que neste caso é na outra extremidade do tapete transportador.

Para terminar a construção base do ambiente fabril, criou-se um terceiro controlador com o mesmo modelo de robô, uma mesa estática para se efetuar a colocação dos azulejos e um segundo tapete transportador para transportar a caixa fechada com os azulejos para a despaletização final, efetuada pelo terceiro robô.

Foi também efetuada a atribuição das ferramentas, sendo que o robô do primeiro e segundo controlador utilizam as ventosas e o robô do terceiro controlador utiliza a garra mecânica, tal como se pode observar na Figura 116.

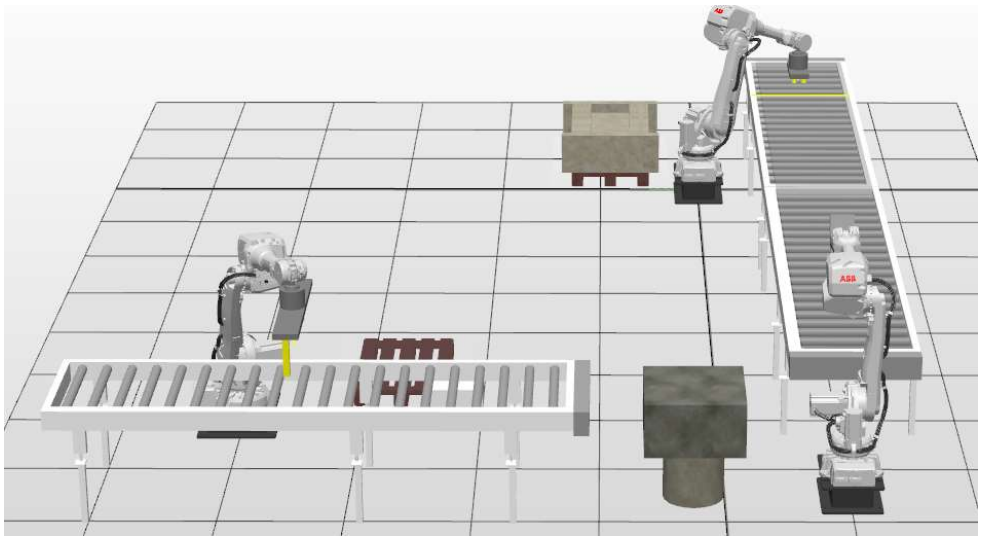


Figura 116 - Ambiente fabril base. Fonte: (Própria)

Para criar uma cópia da ventosa do primeiro robô, acedeu-se à ventosa na janela "*Layout*", efetuou-se a sua cópia através das teclas de atalho "*CTRL + C*" e colou-se sobre o segundo robô com as teclas de atalho "*CTRL + V*", obtendo o resultado observado na Figura 117.

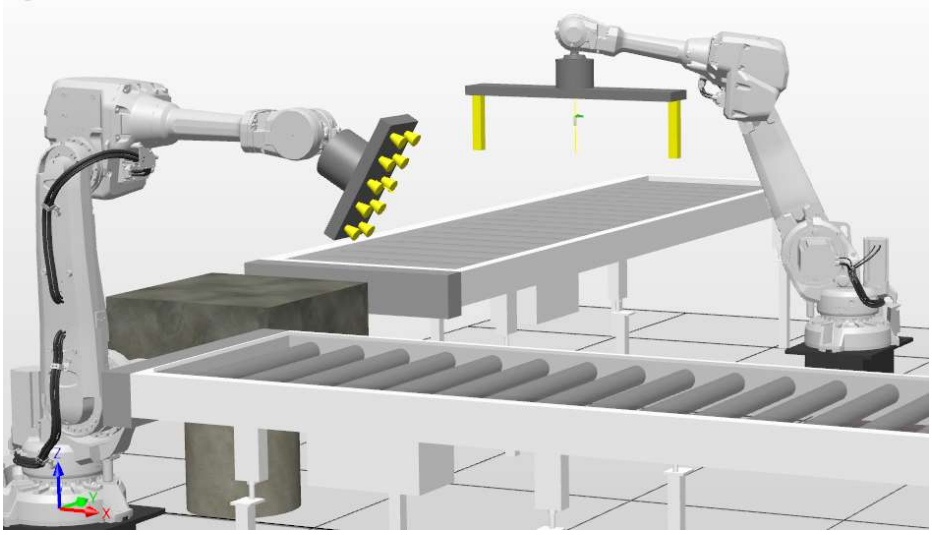


Figura 117 - Atribuição das garras aos robôs criados. Fonte: (Própria)

3.1.11 - Encaixotamento dos azulejos

A certa altura do processo fabril, os azulejos serão colocados dentro de uma caixa de cartão e serão embalados. No simulador este processo pode ser executado de duas maneiras distintas. A primeira, mais semelhante à realidade, seria a criação de uma caixa de cartão aberta, posicioná-la numa mesa, colocar o número de azulejos pretendidos em cima dela (neste caso são três), simular o fecho da caixa através da rotação dos sólidos correspondentes as laterais da caixa dentro de um "Smart Component" e, com o auxílio de quatro sensores, selecionar os três azulejos e a caixa para serem agarradas pela garra, pois cada sensor só pode detetar um objeto de cada vez, logo será necessário um sensor por objeto a agarrar.

Esta solução apresenta complicações desnecessárias, a criação de vários blocos lógicos, tais como quatro blocos "Attacher" e outros quatro "Detacher" e a atribuição de quatro sensores adicionais sempre que for necessário reposicionar a caixa com os azulejos.

Figura 118 - Caixa fechada (esquerda) e caixa aberta (direita). Fonte: (Própria)

O segundo método utiliza a ilusão de ótica para transformar a caixa aberta e os três azulejos num único sólido paralelepípedo que simula a caixa fechada, podendo-se observar estes dois novos sólidos na Figura 118.

Tal como no primeiro método, efetua-se a criação da caixa aberta, posiciona-se a caixa numa mesa e os três azulejos em cima desta. Quando o robô enviar um sinal de que colocou o terceiro azulejo, os quatro sensores detetam os objetos, escondem-nos através de uma "Smart Component" e atribui-se o comportamento de inativo. Este comportamento faz com que o objeto não interaja com nenhum outro objeto, sensor ou gravidade física da simulação. É de seguida criado uma caixa fechada e posicionada instantaneamente num outro tapete transportador que leva a caixa para o terceiro e último robô.

A solução abordada é mais simples e implica menos carga para o programa simulador. Esta abordagem implica a utilização dos quatro sensores mencionados anteriormente, porém apenas para detetar a caixa aberta e os azulejos para lhes alterar as propriedades físicas.

3.1.12 - Criação de um "*Source*" e de um "*Sink*"

A "Source" é uma componente que, através de um modelo sólido escolhido, cria cópias desse modelo ao comando do utilizador. A "Sink" é uma componente que apaga totalmente um objeto á escolha do utilizador. Deste modo é possível criar um objeto temporário e eliminá-lo quando necessário.

O primeiro passo será criar um distribuidor de caixas vazias que chegam até perto do segundo robô através de um tapete transportador. Para tal, cria-se o tapete através da barra de tarefas "*Home*" → "*Import Library*" → "*Equipment*" e seleciona-se o "*Conveyor*" desejado.

Após posicioná-lo num local apropriado, cria-se um "*Smart Component*" vazio, ao qual, selecionando-o através do botão direito do rato na janela "*Layout*" e escolher a opção "*Edit Component*". Na nova janela aberta, abre-se a secção "*Compose*" → "*Add*

Component" e adiciona-se a ação "*Source*". Os dados a preencher são a "Source", que é a escolha do objeto sólido que serve de modelo para as cópias a serem criadas, a posição onde é pretendida a criação das cópias e a orientação da cópia. É importante selecionar a opção "*Transient*", pois esta faz com que as cópias sejam temporárias durante a simulação, ou seja, uma vez terminada a simulação, as caixas, incluindo o registo do nome destas, é apagado do programa, poupando espaço de memória e evitando "*bugs*" de memória. Nota-se, tal como mostra a Figura 119, que na opção "*Physics Behavior*" colocou-se a opção "*None*", pois o modelo original já apresenta o comportamento "Dynamic", que é uma propriedade guardada e transmitida pelo "Source" para a cópia.

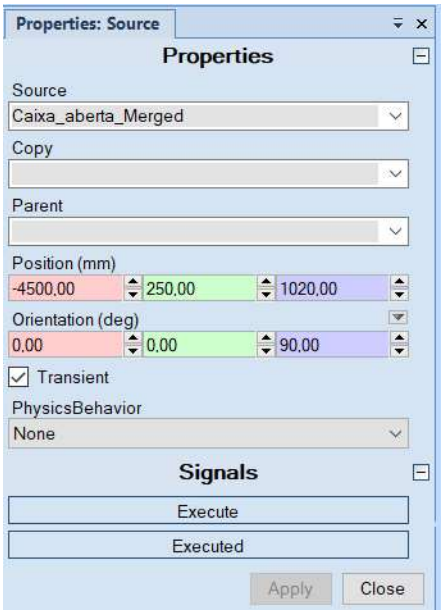


Figura 119 - Criação de um "Source". Fonte: (Própria)

Para a criação do "Sink", procede-se do mesmo modo na procura da ação, porém este não se preenche nenhum dos campos, pois estes campos irão recorrer a sensores e sinais de "status" para apagar a caixa pretendida, tal como se pode observar na Figura 120.

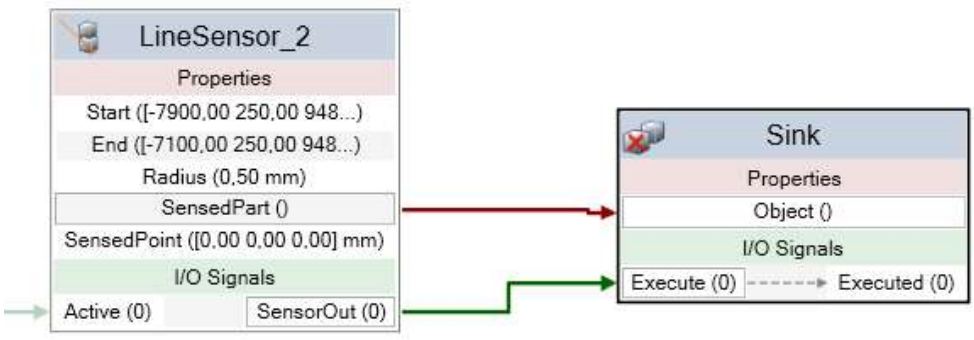


Figura 120 - Acionamento do "Sink" através de deteção do sensor linear. Fonte: (Própria)

Uma vez apagada a caixa aberta, esta é substituída por uma caixa fechada. Para tal. Coloca-se um “Source” que crie cópias da caixa fechada sobre o tapete transportador. Este “Source” será ativado pelo robô quando este coloca o terceiro azulejo sobre a caixa aberta, com um “Delay” de dois segundos, obtendo o resultado presente na Figura 121.

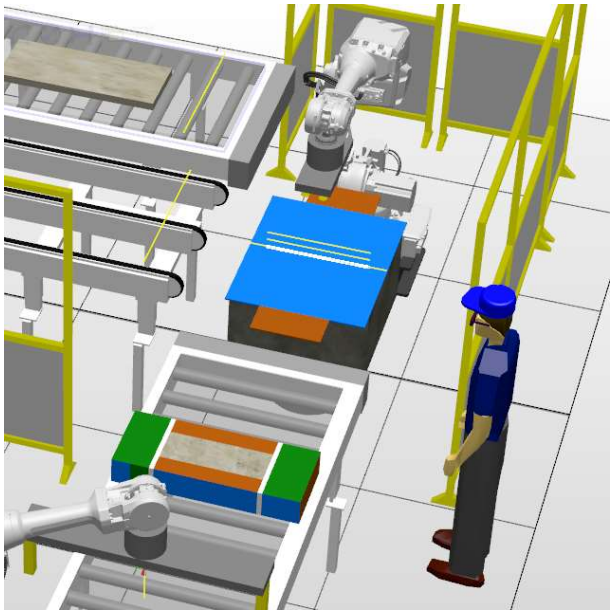


Figura 121 - Caixa aberta e caixa fechada. Fonte: (Própria)

3.1.13 - Movimento linear controlado

A atribuição de velocidade de superfície a um sólido de modo que este transporte os objetos que entram em contacto com este, apresenta alguns problemas na simulação. O primeiro problema é que enquanto o objeto estiver sobre este, não para o seu deslocamento, o que obriga a criar um sistema de rastreio de movimento do tapete ou de temporização precisa quando for necessário ao robô agarrar o objeto. Outro problema é o facto de não ser possível criar um sistema de transporte com pausas, ou seja, os objetos transportados não podem parar para a realização de tarefas.

Tendo isto em conta, criou-se através de um “*Smart Component*” → “*Compose*” → “*Add Component*” um “LinearMover2”. Esta opção, como é possível observar na Figura 122, difere do “LinearMover” pois, enquanto a “LinearMover” apenas fornece direção e velocidade ao objeto (obtendo assim o mesmo efeito que a atribuição de velocidade a uma superfície), o “LinearMover2” possibilita a criação de uma deslocação de uma distância pretendida com a duração de um tempo definido.

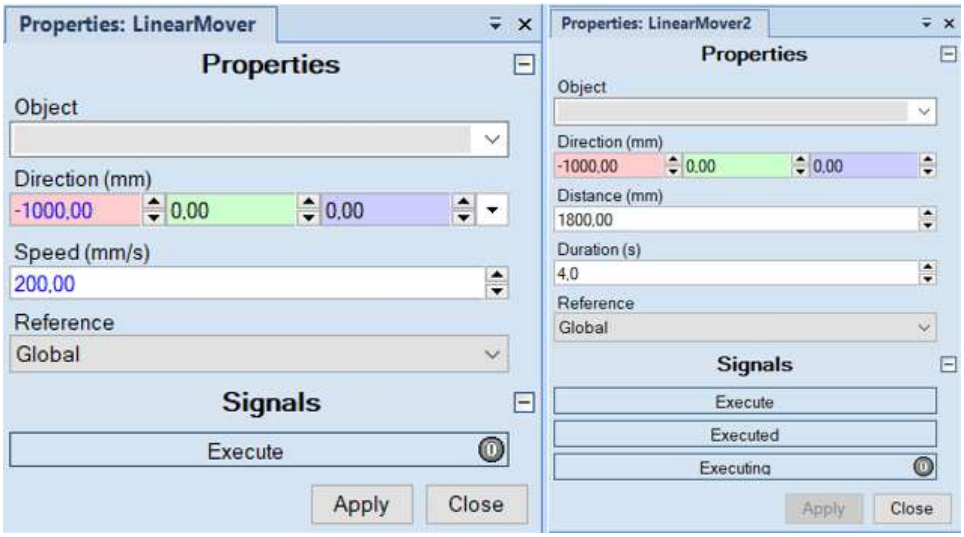


Figura 122 - Parâmetros do "LinearMover" (esquerda) e do “LinearMover2” (direita). Fonte: (Própria)

3.1.14 - Divisão das áreas de trabalho

Uma vez criadas e posicionadas as componentes essenciais da réplica digital, esta encontra-se dividida nas três partes representadas na Figura 123.

- A zona de despaletização (1), onde um robô é responsável por despaletizar os azulejos da palete e colocá-los no tapete transportador.
- A zona de embalamento (2), onde um outro robô é responsável por recolher as caixas de cartão abertas do tapete transportador para uma mesa e encher a caixa com os azulejos provenientes do primeiro tapete, de onde um funcionário irá retirar e colocar num terceiro tapete transportador.
- A zona de paletização (3), onde um terceiro robô recolhe as caixas fechadas provenientes do tapete transportador e coloca-as na palete.

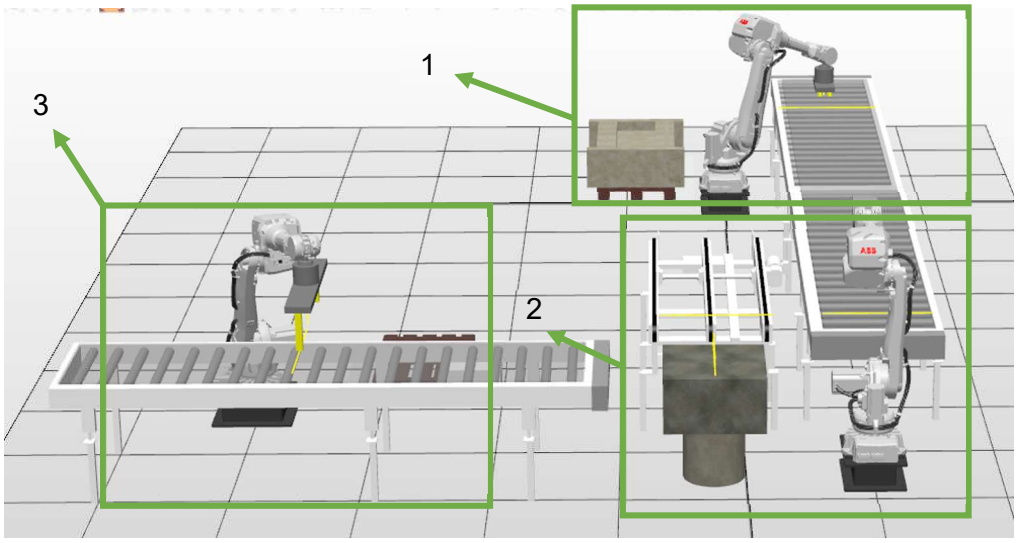


Figura 123 - Secções de trabalho. Fonte: (Própria)

Para finalizar a réplica digital, criou-se redes de proteção em redor dos robôs, conforme a Figura 124.

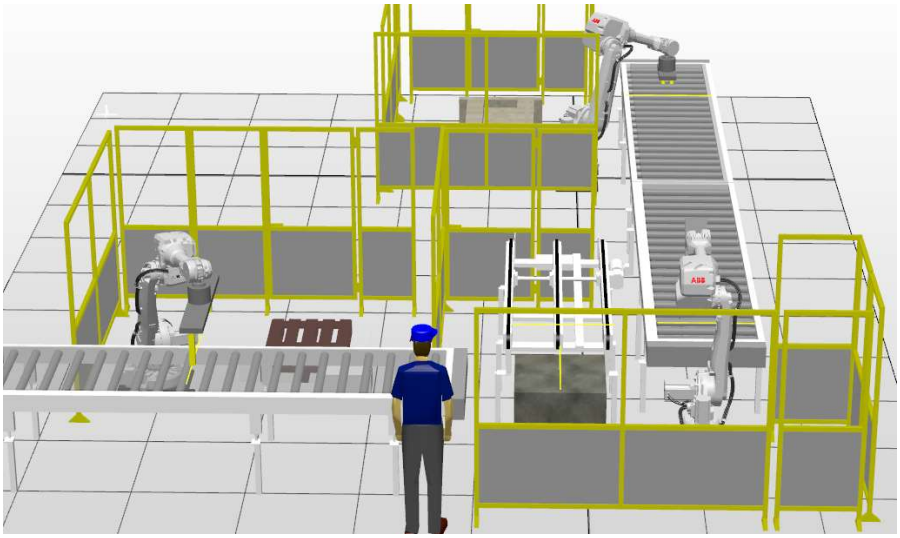


Figura 124 - Réplica digital com redes de proteção. Fonte: (Própria)

3.2 - Desenvolvimento de garras para processos cerâmicos

Foi apresentado pela empresa Porcelanas da Costa Verde SA um conjunto de problemas face aos atuais processos de vidragem a mergulho e polimento de peças cerâmicas, processos estes executados de forma manual por funcionários.

De um modo generalizado, os principais problemas da colocação de mão de obra humana neste setor são o facto de esta ser envelhecida e cada vez mais escassa, e, devido ao facto dos processos serem repetitivos e por vezes requerem manuseio de cargas elevadas, pode provocar problemas físicos e psicológicos nos trabalhadores a longo prazo. (WEBER, FUSSLER, J. F. O'HANLON, & GRANDJEAN, 1980)

Através da substituição dos funcionários por robôs manipuladores neste tipo de trabalhos, é possível eliminar os problemas mencionados acima, enquanto se reduz o tempo do processo. Isto também apresenta melhorias nas condições de trabalho dos funcionários substituídos, uma vez que as capacidades destes podem ser implementadas na supervisão do processo ou em outras tarefas menos perigosas na fábrica.

Devido á especificidade dos processos fabris expostos e a fragilidade do material a manipular, é necessário desenvolver garras adaptadas a cada um dos processos. Assim sendo, os próximos subcapítulos expões estes processos e as garras de teste desenvolvidas para estes. Todas as garras e objetos desenvolvidos para a criação das réplicas em miniatura foram impressos em impressoras 3D de ABS (acrilonitrila

butadieno estireno) ou resina e os seus modelos foram criados virtualmente utilizando o programa FreeCAD, observado na Figura 125.

Figura 125 - Ícone da aplicação FreeCAD. Fonte: (https://pt.wikipedia.org/wiki/Ficheiro:FreeCAD-logo.svg)

3.3 - Vidragem de pires de café

O processo de vidragem cerâmica dá-se pela aplicação de uma película de revestimento composta, na sua maioria, por sílica. Esta película, denominada por vidrado, serve tanto para dar brilho á peça cerâmica, como para aumentar a sua rigidez. O vidrado pode ser aplicado de duas formas: através da pulverização a seco sobre a superfície da peça, ou do mergulho da peça na mistura líquida (Aksoy, Polat, Polat, & Coskun, 2006).

O projeto abordado trata-se de vidragem por mergulho de pratos ou bandejas, onde atualmente este é executado manualmente por funcionárias. Neste caso, as bandejas ou pratos chegam á área de vidragem sem o vidrado, uma funcionária agarra a peça cerâmico pela base inferior com as pontas dos dedos, reduzindo assim a área não vidrada em contacto com os dedos, mergulhando o cerâmico num reservatório grande de líquido vidrado. De seguida retira o cerâmico da tina e executa vários movimentos circulares de modo a espalhar o vidrado na parte superior do cerâmico. Por último, coloca o cerâmico já vidrado numa mesa de forma redonda e manualmente rotativa para secar. Passados alguns minutos, o cerâmico fica seco e é retirado para o próximo processo fabril.

3.3.1 - Réplica física da vidragem

Tendo em mente o espaço fabril já existente, a réplica em miniatura foi criada mantendo o método do processo, porém adaptado ao robô com uma garra ventosa para efetuar o vidrado num pires de café.

Foi escolhida uma garra ventosa, pois esta é a garra mais semelhante ao processo de agarre do cerâmico a vidrar, uma vez que deste modo a área de contacto entre o cerâmico e a garra é reduzido, diminuindo assim também a área do cerâmico não vidrada.

Para automatizar o processo, muniu-se a mesa rotativa com um motor "*stepper*" controlado por um Arduíno e ligado aos sinais de saída do robô. A garra é também controlada por um motor "stepper" de modo a executar o movimento circular falado no subcapítulo anterior.

Contudo, a conclusão deste processo sofreu várias etapas de teste e de adaptação, tendo começado pela criação de um apoio simples para a ventosa e de uma mesa estática.

3.3.2 - Ventosa simples

O primeiro passo para o desenvolvimento deste processo foi criar um suporte que una a ventosa ao robô, de modo que este consiga agarrar o cerâmico pela base inferior e vire a parte superior para cima com facilidade e apelas rodando um dos eixos do robô (neste caso, o último).

O robô disponibilizado já apresentava um apoio para uma pequena ventosa, presente na Figura 126, porém este não é propício para efetuar a rotação do cerâmico mencionada no parágrafo anterior. Este apoio, através da realização de engenharia inversa, serviu de alvo de medição de modo a criar dos furos para montar o apoio ao robô com parafusos e para montar a ventosa.

Figura 126 - Ventosa e apoio original. Fonte: (Própria)

Após efetuadas as medições, criou-se o apoio da ventosa de acordo com a Figura 127, semelhante ao apoio utilizado como referência, porém com um ângulo de noventa graus entre a superfície de apoio do robô e a superfície de apoio da ventosa. Foram também adicionados dois furos para colocar a ventosa em dois perfis distintos.

Aumentou-se o alcance da ventosa e criou-se um triângulo de apoio, devido ao facto do material de impressão 3D a utilizar ser menos resistente.

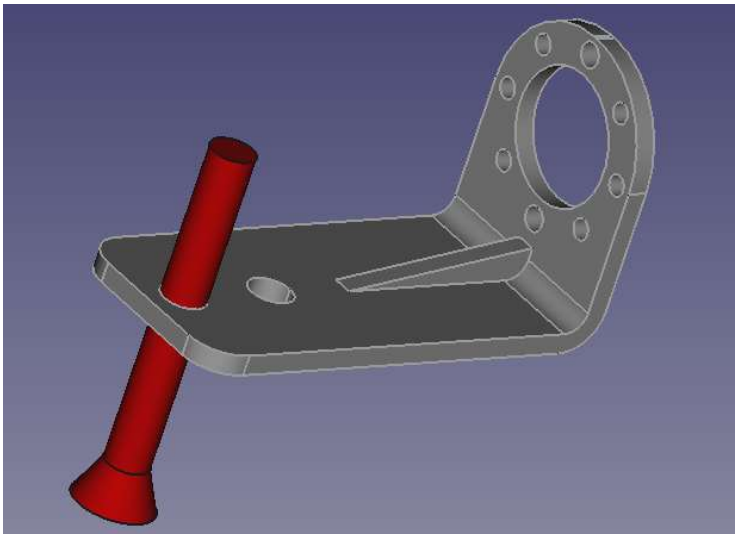


Figura 127 - Modelo de ventosa simples. Fonte (Própria)

Após a finalização do modelo do apoio da ventosa, converteu-se o ficheiro para um formato compatível com a impressora 3D, programou-se a impressão em software próprio (*Slicer*) e imprimiu-se a peça. O material utilizado para a impressão da garra foi resina rígida (SLA), cuja solidificação da resina liquida é efetuada por laser. Após a impressão da peça, esta necessita ser colocada num forno de cura por luz ultravioleta (UV) durante 30 minutos para solidificar na sua totalidade e aumentar a sua resistência física. Após este processo, a peça está pronta a ser montada no robô, conforme a Figura 128.

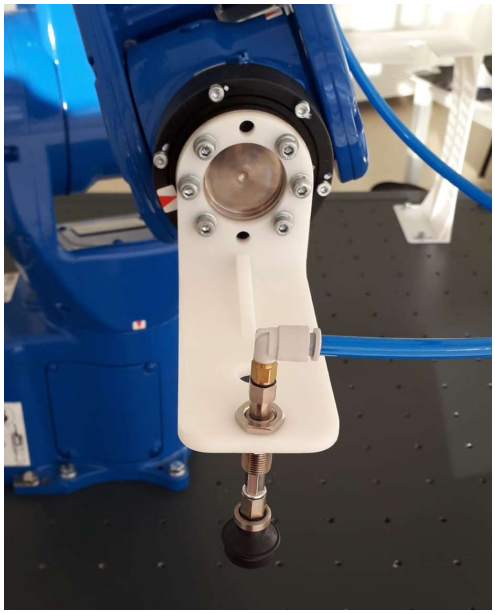


Figura 128 - Garra ventosa simples. Fonte: (Própria)

O acionamento da ventosa dá-se através do envio se sinal "*True*" de uma das saídas do robô que excita um relé que por sua vez ativa um sistema de criação de vácuo.

3.3.3 - Mesa estática e tina de vidrado

O seguinte passo foi a criação de uma mesa estática para o robô pousar os pires após executar o vidrado. Através da manipulação do pires com a ventosa simples, observou-se que para esta executar a sua tarefa com sucesso, é necessário a criação de ranhuras na mesa para a ventosa passar e pousar o pires nela e, para o braço do robô passar por baixo da mesa sem colisão com esta e com o chão, a mesa tem de ser alta, obtendo o modelo mostrado na Figura 129.

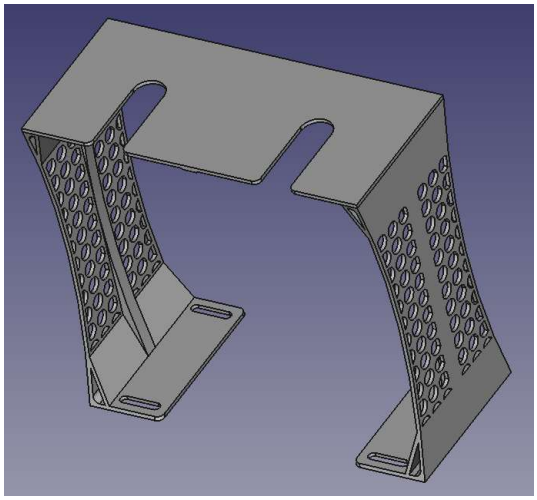


Figura 129 – Modelo de mesa estática com ranhuras. Fonte: (Própria)

Após a impressão 3D da mesa em resina e montagem no espaço, foi fornecido uma tina cerâmica para simular a tina de vidrado do espaço fabril. Porém, devido ao facto do controlador do robô estar debaixo da mesa de montagem e esta apresentar furos na sua base, não foi utilizado qualquer líquido de simulação de vidrado, pois a utilização de líquido neste ambiente pode danificar o controlador, uma vez que não existe a proteção necessária para o fazer.

3.3.4 - Simulação de teste de vidragem por mergulho

Uma vez o espaço montado, criou-se o código de movimentação do robô de modo que este recolhesse um dos dois pires, um de cada vez, com a base inferior voltada para cima, conforme a Figura 130.

Figura 130 - Recolha do pires. Fonte: (Própria)

Após a sua recolha, o robô vira a base superior do pires para a frente, mantendo uma inclinação para que este entre na tina sem colidir com esta, como é possível observar na Figura 131.

Figura 131 - Simulação de mergulho no vidrado. Fonte: (Própria)

De seguida o robô retira o pires da tina e vira a base superior do pires para cima, executando movimentos circulares para simular o espalhar do vidrado. O pires apresenta uma inclinação, observada na Figura 132, durante os movimentos circulares, baseados no movimento visualizado de uma funcionária a executar a mesma tarefa.

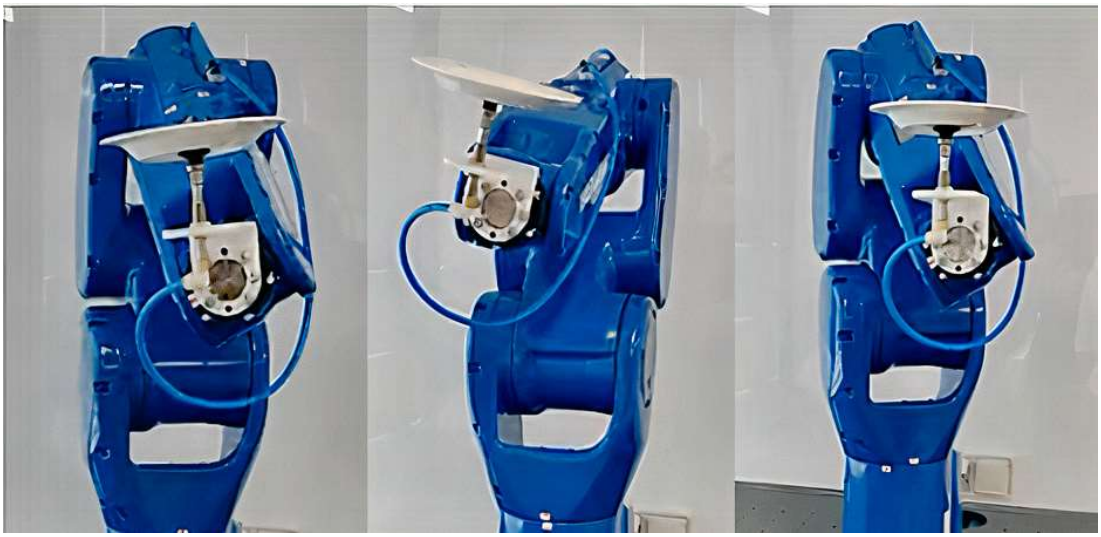


Figura 132 - Simulação do espalhar do vidrado. Fonte: (Própria)

Após executar o movimento circular durante algum tempo, o pires está pronto para ser colocado na mesa estática, simulando assim a secagem do vidrado. O programa depois repete o mesmo processo para o segundo pires, colocando-o por fim na segunda ranhura, de acordo com a Figura 133.

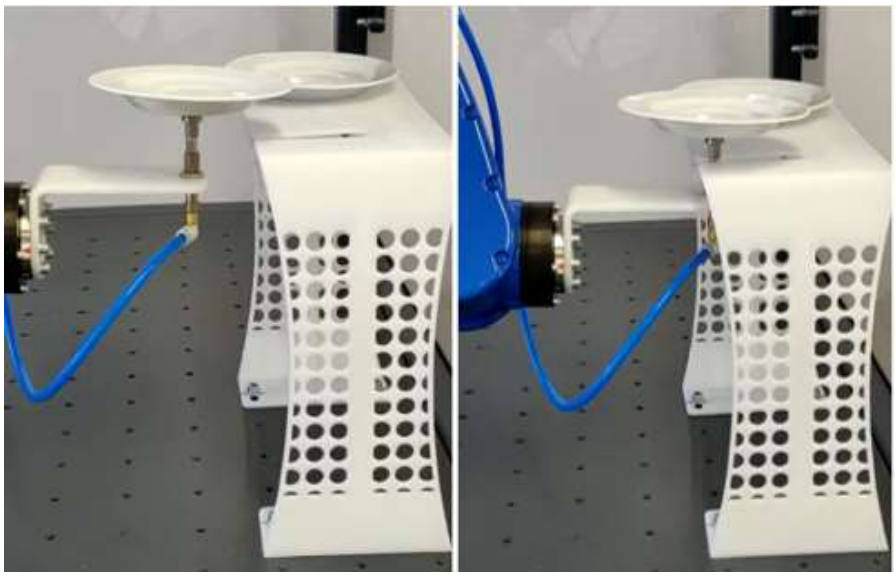


Figura 133 - Robô a colocar pires na mesa. Fonte: (Própria)

A simulação com a ventosa simples e a mesa estática serviu para aferir alguns constrangimentos causados pelo facto da mesa e da garra criadas serem estáticas. O robô executa o movimento circular para espalhar o vidrado com alguma dificuldade, obrigando a várias juntas do mesmo rodarem com bastante velocidade e complexidade, resultando isto num conjunto de movimentos mais lento, o que não é ideal. Outro ponto a salientar é o facto de apenas existir duas ranhuras na mesa e em posições diferentes, ou seja, coordenadas segundo o eixo XY distintas, o que dificulta a criação de ciclos repetitivos.

3.3.5 - Mesa rotativa

O seguinte passo na criação da réplica física em miniatura foi substituir a mesa estática por uma mesa de forma redonda e rotativa de modo a simplificar a criação de ciclos de trabalho. A mesa idealizada apresenta três ranhuras com a mesma finalidade que a mesa estática, porém estas seguem a direção do centro da mesa. Deste modo, quando o robô coloca o pires numa ranhura vazia, esta roda 120 graus, colocando a próxima ranhura vazia no mesmo local que a anterior, sendo assim possível o robô aceder à próxima ranhura executando os mesmos movimentos que na anterior. Para tal, criou-se um modelo em 3D no FreeCAD de modo que esta suporte os três pires de café.

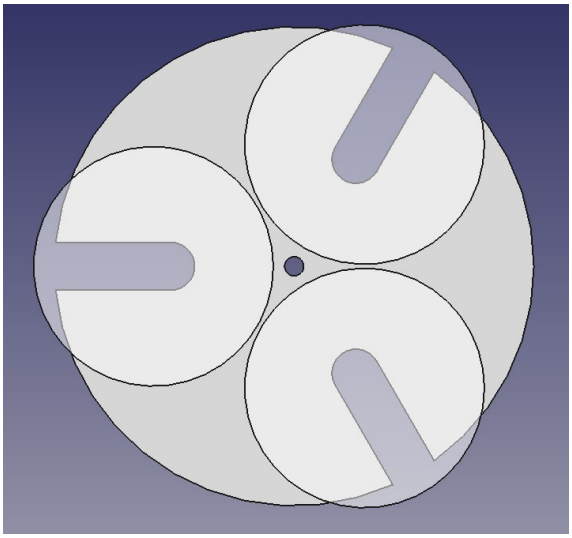


Figura 134 - Vista superior da mesa com ranhuras e pires semitransparentes. Fonte: (Própria)

Como é possível observar na Figura 134, esta apresenta três ranhuras, uma por pires, cada uma afastada exatamente 120 graus em relação ao centro da mesa, que será também o seu eixo de rotação.

O ideal seria a mesa ter um diâmetro maior, de modo que os pires ficassem completamente dentro. Porém, devido ao limite das dimensões da impressora 3D, uma mesa com diâmetro maior obrigava a dividir o círculo em três ou mais partes, criando diferença nas distâncias em graus entre as ranhuras quando fosse executada a colagem das partes (a cola ocupa espaço quando aplicada, alterando as dimensões da mesa sem qualquer rigor).

Esta diferença não controlada no dimensionamento da mesa faria com que a rotação da mesa, controlada de 120 em 120 graus pudesse por vezes não coincidir com o centro da ranhura, criando a hipótese de colisão da ferramenta com a mesa quando esta estivesse a colocar o pires.

Para ser possível controlar o ângulo de rotação utilizou-se o motor de passo (stepper motor) Nema 17. Este motor foi escolhido devido à sua força debitada e ao facto de

ser possível aplicar com precisão uma quantidade de passos cada vez que for necessário rodar a mesa. Estes passos são diretamente proporcionais ao ângulo que a mesa necessita rodar.

Foi adicionado ao tampo da mesa, conforme a Figura 135, um suporte para um veio hexagonal, presente na Figura 136, que está conectado ao veio do motor, de modo que este faça rodar a mesa.

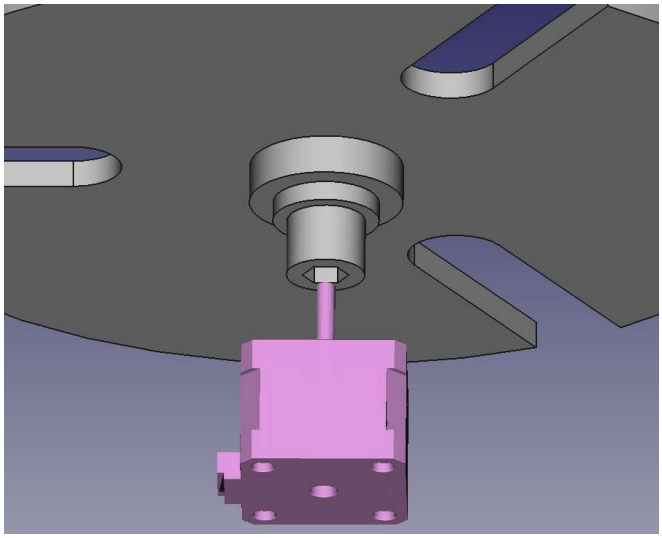


Figura 135 - Guia de veio hexagonal e motor Nema. Fonte: (Própria)

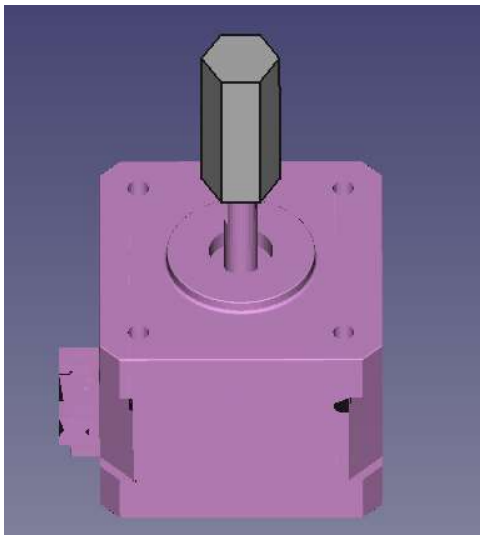


Figura 136 - Veio hexagonal montado no veio do motor. Fonte: (Própria)

Tendo especial atenção às dimensões do motor Nema 17, criou-se o tronco da mesa com uma jaula de apoio ao motor e, para evitar a erosão do material quando o tampo da mesa roda, criou-se também um compartimento para colocar um rolamento, de acordo com a Figura 137.

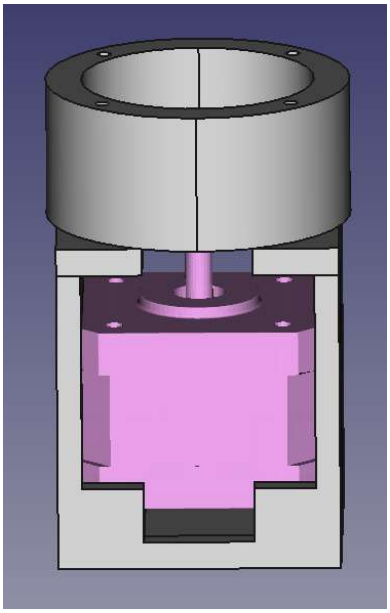


Figura 137 - Jaula do Nema e suporte para rolamento. Fonte: (Própria)

Por último, criou-se a perna e o pé da mesa, de modo que este seja aparafusado na mesa da jaula do robô, obtendo por fim o modelo finalizado da mesa rotativa presente na Figura 138.

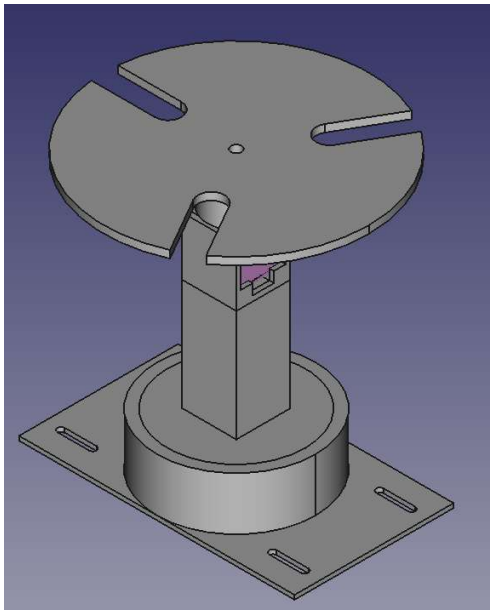


Figura 138 - Modelo da mesa rotativa. Fonte: (Própria)

Sempre que o robô dá a instrução para a mesa rodar, esta roda 120 graus, porém é impossível posicionar corretamente a primeira ranhura sem auxílio de sensores externos. Para tal utilizou-se um sensor de distância integrado na mesa, tal como é possível observar na Figura 139.

Figura 139 - Sensor de distância. Fonte: (Própria)

O material utilizado para a impressão da mesa foi ABS, cujo processo de impressão é de extrusão de material (FDM), e, ao contrário da resina, não necessita de passar por um processo de cura para solidificar. As partes impressas da mesa foram posteriormente coladas ou montadas com auxílio de parafusos, de acordo com a Figura 140.

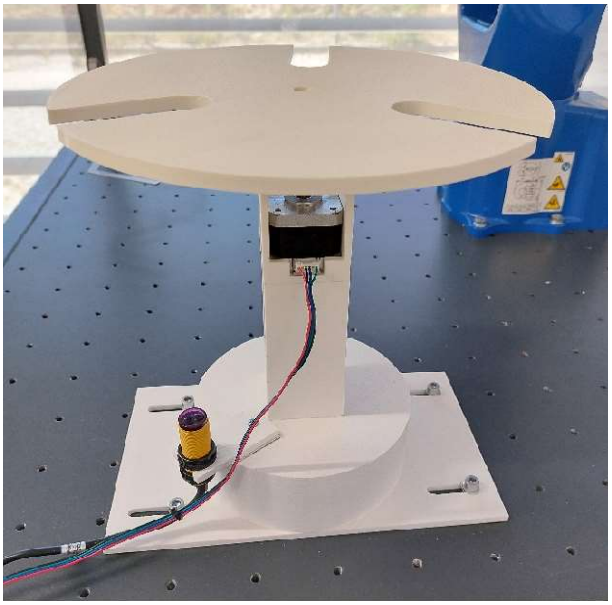


Figura 140 - Mesa rotativa com sensor de ranhuras. Fonte: (Própria)

3.3.6 - Ventosa rotativa

Para que o processo de espalhar o vidrado na face superior do pires seja mais rápido e eficiente, colocou-se a hipótese de dar movimento rotacional à ventosa recorrendo a engrenagens cónicas movidas por um outro motor de passo Nema 17. Nota-se que para este fim, um qualquer motor de corrente contínua com total liberdade rotacional pode ser utilizado, porém este motor foi escolhido devido á sua compatibilidade de

programação com o outro motor Nema responsável por fazer o tampo da mesa rotativa girar.

Devido à complexidade da nova peça de apoio, e ao facto de existir movimento rotacional de engrenagens, o primeiro passo na criação da nova ventosa foi construir uma réplica mais detalhada da ventosa, incluindo o conetor do tubo de vácuo e as porcas de aperto da ventosa ao seu apoio, tendo também sido efetuada a importação de um modelo 3D do motor Nema 17, tal como consta na Figura 141

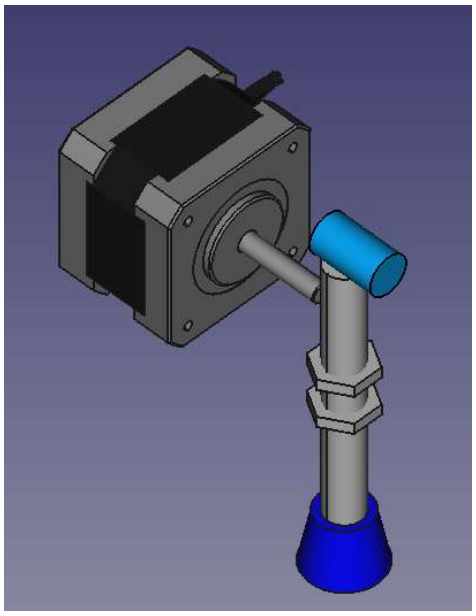


Figura 141 - Modelos 3D da ventosa e do motor de passo. Fonte: (Própria)

De seguida, construiu-se as engrenagens cónicas presentes na Figura 142, com um perfil de apoio adaptado ao motor e à ventosa. Para evitar que a ventosa gire a altas velocidades, criou-se um redutor de velocidade através do número de dentes atribuídos às engrenagens, sendo que a engrenagem do motor de passo apresenta 17 dentes e a engrenagem da ventosa apresenta 25 dentes.

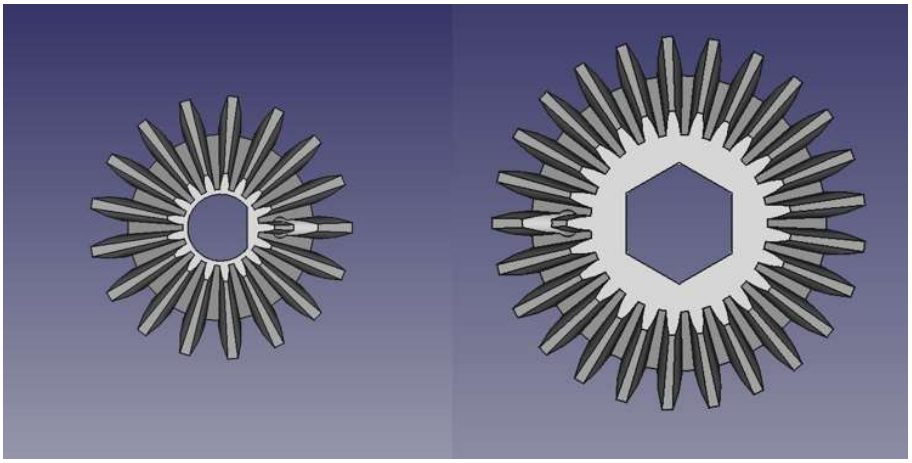


Figura 142 - Engrenagens cónicas para Nema 17 (esquerda) e ventosa (direita). Fonte: (Própria)

De seguida dispôs-se os modelos nas engrenagens de modo que os seus dentes interajam entre si e inseriu-se o modelo do motor de passo e da ventosa sobre o eixo das engrenagens, sendo assim possível obter as dimensões da localização destes componentes no espaço físico, tal como mostra a Figura 143.

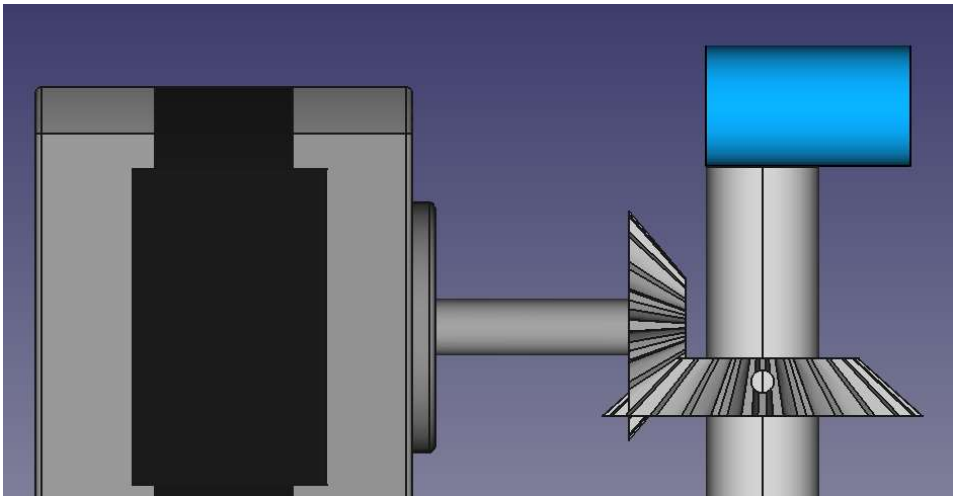


Figura 143 - Modelo de interação entre motor de passo e ventosa. Fonte: (Própria)

Através das dimensões obtidas na criação do primeiro apoio para a ventosa, criou-se um apoio, porém adaptado para o motor de passo e para a inserção de um rolamento sobre o eixo da ventosa, de modo a prevenir o desgaste do material face ao movimento rotacional da ventosa. O triângulo de suporte é de maior dimensões, de modo que a peça ofereça resistência ao peso acrescentado do motor de passo.

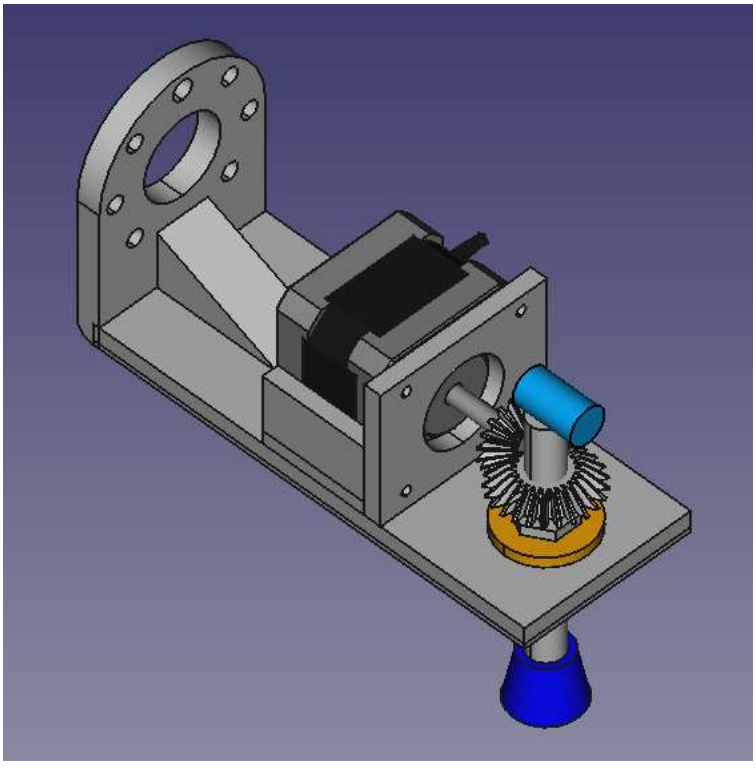


Figura 144 - Apoio de ventosa rotacional. Fonte: (Própria)

De acordo com a Figura 144, o rolamento (laranja) não apresenta segurança face à eventualidade deste apoio ser manipulado do avesso, caindo devido ao seu peso. Para tal criou-se uma peça removível que prende o rolamento ao apoio, mostrado na Figura 145.

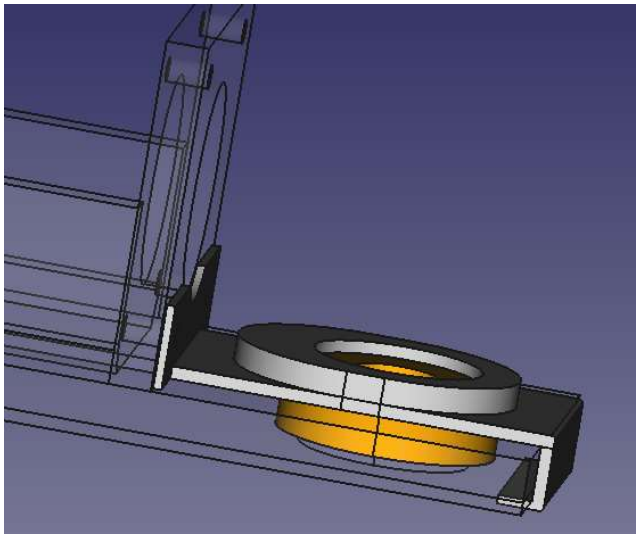


Figura 145 - Apoio do rolamento. Fonte: (Própria)

De seguida, imprimiu-se todos os componentes em resina e efetuou-se a sua montagem, de acordo com a Figura 146.

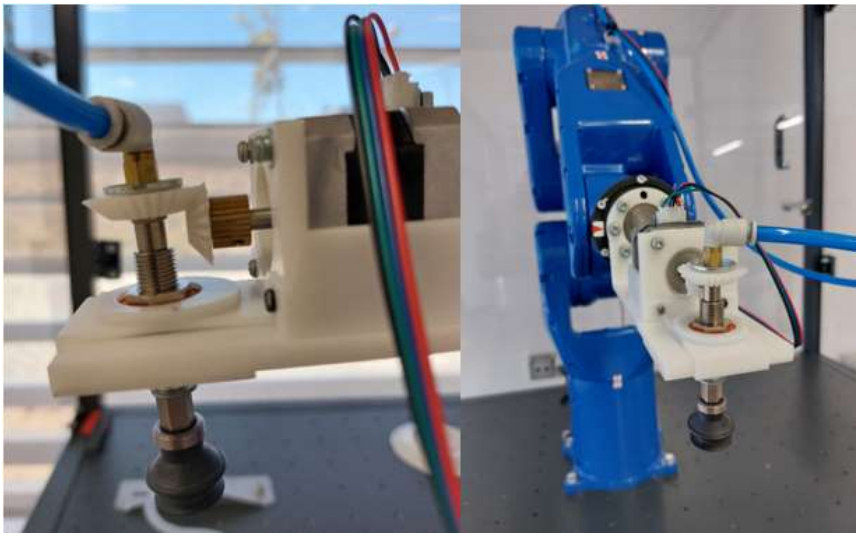


Figura 146 - Montagem de apoio para ventosa rotativa. Fonte: (Própria)

3.3.7 - Simulação final de vidragem por mergulho

Uma vez o espaço montado, criou-se o código de movimentação do robô de modo que este recolhesse um dos três pires, um de cada vez, com a base inferior voltada para cima, tal como é mostrado na Figura 147.

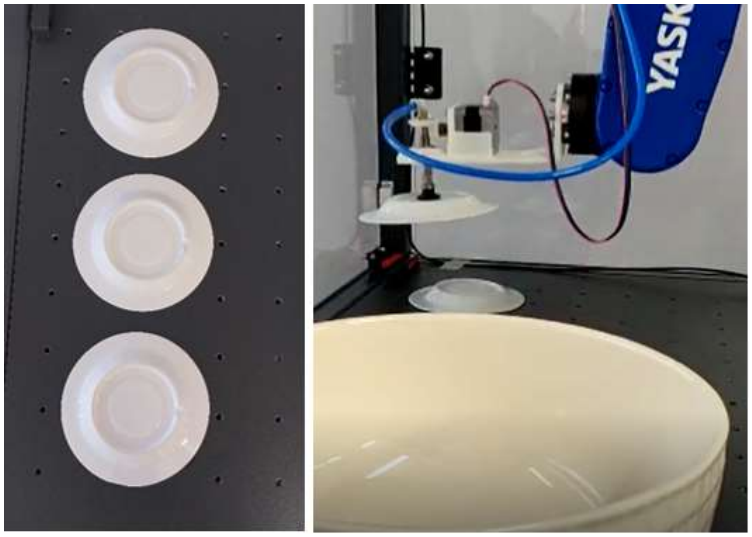


Figura 147 - Disposição e agarre dos pires. Fonte: (Própria)

Após a sua recolha, o robô vira a base superior do pires para a frente, mantendo uma inclinação para que este entre na tina sem haver colisão, e girando o pires para simular o mergulho e aplicação do vidrado de acordo com a Figura 148.

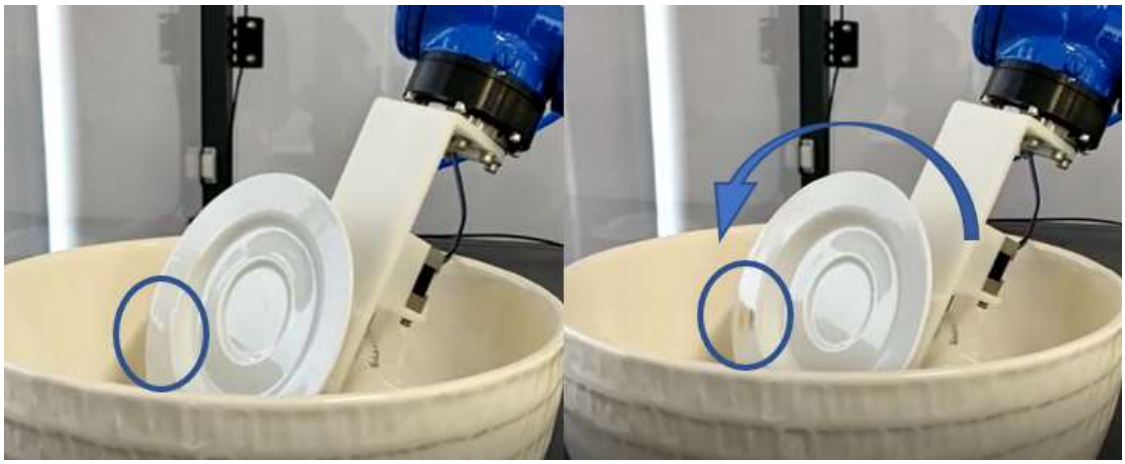


Figura 148 - Simulação de mergulho do pires com rotação. Fonte: (Própria)

De seguida o robô retira o pires da tina e, conforme a Figura 149, vira a base superior do pires para cima, executando movimentos circulares e aplicando rotação através do motor para simular o espalhar do vidrado. Contudo, os movimentos circulares executados pelo robô, em comparação com a simulação real com a ventosa simples, são executados menos vezes e com menor inclinação, tornando este processo mais rápido e eficiente.

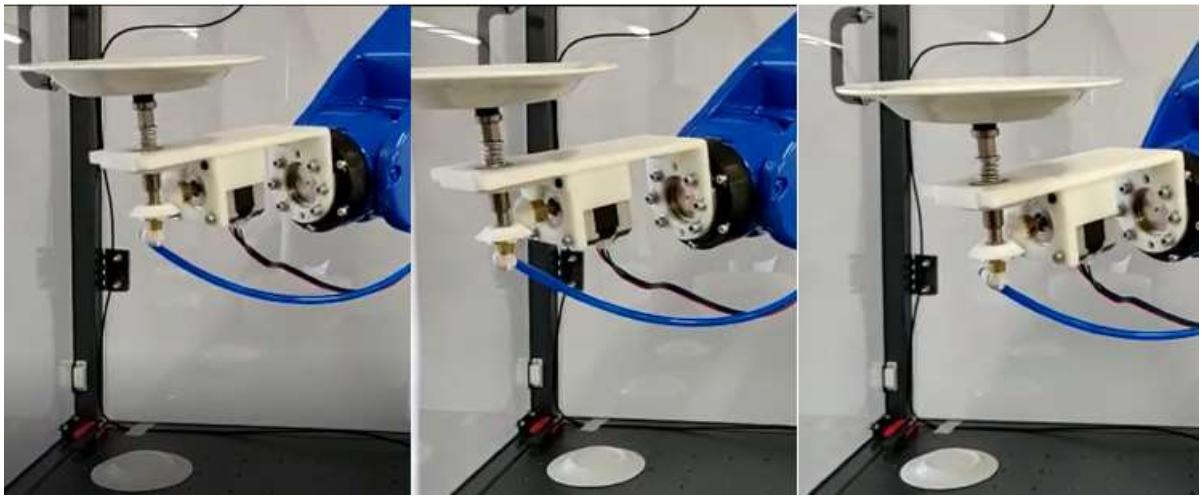


Figura 149 - Simulação do espalhar do vidrado com rotação. Fonte: (Própria)

Após executar o movimento circular durante algum tempo, o pires está pronto para ser colocado na mesa rotativa, simulando assim a secagem do vidrado, de acordo com a Figura 150.

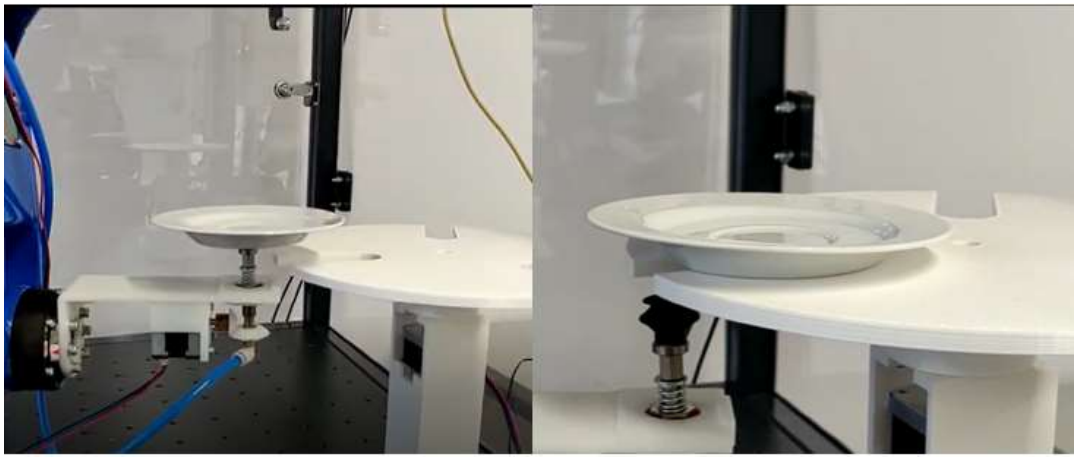


Figura 150 - Colocação do primeiro prato na mesa rotativa. Fonte: (Própria)

Após a colocação do primeiro prato na mesa, esta gira 120 graus no sentido dos ponteiros do relógio e aguarda pelo próximo prato, completando o ciclo mais duas vezes até os três pratos estarem colocados sobre ela, segundo as disposições da Figura 151. Após serem colocados os três pratos na mesa, de acordo com a Figura 152, o robô aguarda alguns segundos e vai buscar os três pratos, um de cada vez, e coloca-os no seu local inicial, possibilitando assim a criação de ciclos para esta tarefa.

Figura 151 - Rotações da mesa de secagem. Fonte: (Própria)

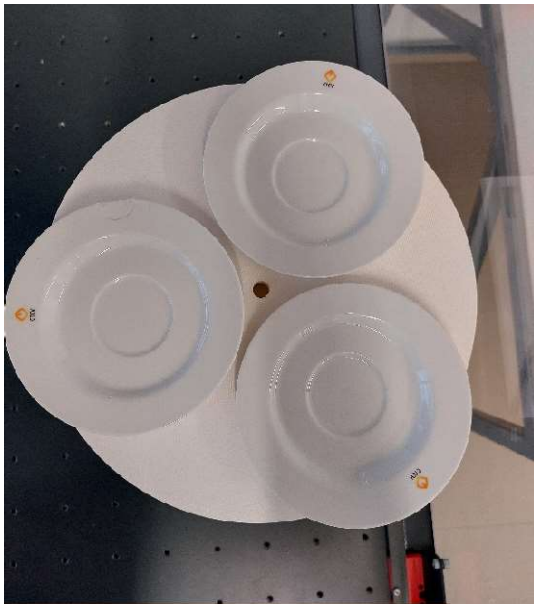


Figura 152 - Mesa rotativa cheia. Fonte: (Própria)

A adição de movimento rotacional à garra e à mesa eliminou todos os constrangimentos mencionados no procedimento da tarefa com a ventosa estática. Devido à eliminação da necessidade de o robô executar movimentos circulares de elevada dificuldade e num elevado número de vezes, o processo no geral tornou-se mais rápido.

A aplicação de movimento rotativo no tampo da mesa, e o facto das ranhuras estarem exatamente distanciadas 120 graus entre si, possibilita a execução de ciclos infinitos do processo e facilita a programação do robô, uma vez que todos os movimentos relacionados com o mergulho, espalhar do vidrado e colocação dos pires na mesa são iguais, poupando tempo de programação e espaço de memória do robô face à criação de novas posições.

3.4 - Polimento de chávenas de café

O processo de polimento cerâmico é utilizado para remover excessos de material ou para dar acabamentos mais suaves às peças cerâmicas. Neste processo é tanto possível aproximar a peça com o auxílio de um manipulador a uma polidora fixa, como incorporar uma polidora na garra do manipulador. (L.G.Li, Z.Y.Zhuob, A.K.H.Kwan, T.S.Zhang, & D.G.Lu, 2020)

O projeto abordado trata-se de polimento da base inferior de chávenas de café, onde atualmente este é executado por funcionárias que agarram na chávena e colocam sobre um tapete polidor em constante movimento.

3.4.1 - Réplica física do polimento

Tendo em mente o método fabril atualmente utilizado, a réplica em miniatura foi criada mantendo o seu princípio de funcionamento, ou seja, o robô agarra a chávena e encosta a sua base inferior a uma polidora. Após ser efetuado o polimento da chávena, a garra coloca-a no local correspondente às chávenas polidas.

Para executar o agarre da chávena, foi desenvolvida uma garra mecânica com vários dedos, uma vez que a chávena vai exercer colisão continuada com a polidora, sendo necessário uma maior área de contacto, força de agarre e estabilidade. Os dedos da garra são de acionamento independente e munidos de sensores de carga, de modo a ser possível aplicar inteligência sensorial à garra.

Contudo, a conclusão deste processo sofreu várias etapas de teste e de adaptação, tendo começado pela criação de uma pinça mecânica adaptada à forma da chávena, seguida de vários ensaios com feedback sensorial.

3.4.2 - Polidora

O primeiro passo na criação da réplica real em miniatura foi o desenvolvimento da polidora presente na Figura 153. Foi utilizada uma esponja de maquilhagem, acionada por um servo motor contínuo.

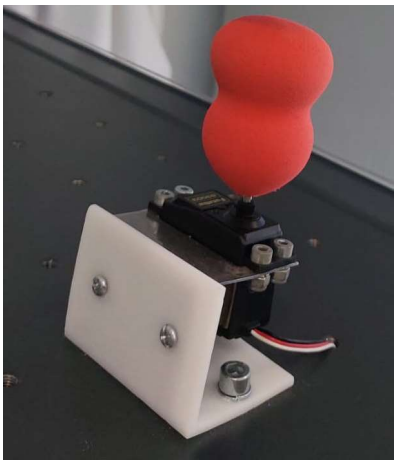


Figura 153 - Polidora. Fonte: (Própria)

Foi escolhida esta esponja devido à sua forma reta na face superior, forma esta propícia para polir a base da chávena, e à curva formada lateralmente, ideal para realizar testes de polimento da asa da chávena.

3.4.3 - Pinça adaptada à chávena

O robô utilizado para executar o polimento da chávena já se encontrava munido por uma garra mecânica com pinças retas. Esta garra é acionada através da aplicação de pressão de ar num dos dois orifícios “O” e “S”, sendo que a pressão de ar colocada no orifício “O” aciona a abertura do mecanismo e a pressão de ar em “S” aciona o seu fecho (ver Figura 154).

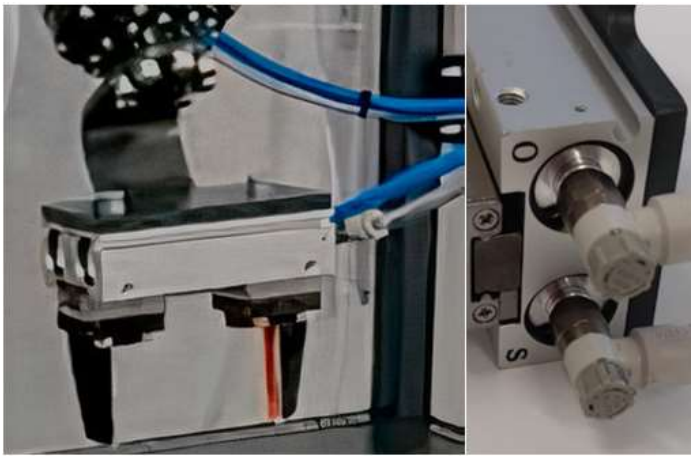


Figura 154 - Garra mecânica com pinças retas e orifícios. Fonte: (Própria)

Realizados alguns testes com esta garra, concluiu-se que esta não é ideal para manipular a chávena, uma vez que a área de contacto entre a chávena e a garra é muito baixa, provocando por vezes a queda e dano do material cerâmico.

Para tornar o processo de manipulação mais eficiente, optou-se por manter o mecanismo de acionamento, do qual é possível efetuar a troca das suas pinças, e desenhar duas pinças com o perfil de agarre adaptado à chávena. O primeiro passo foi obter o modelo 3D da chávena de ensaio de acordo com a Figura 155, disponibilizada pelo designer da empresa.

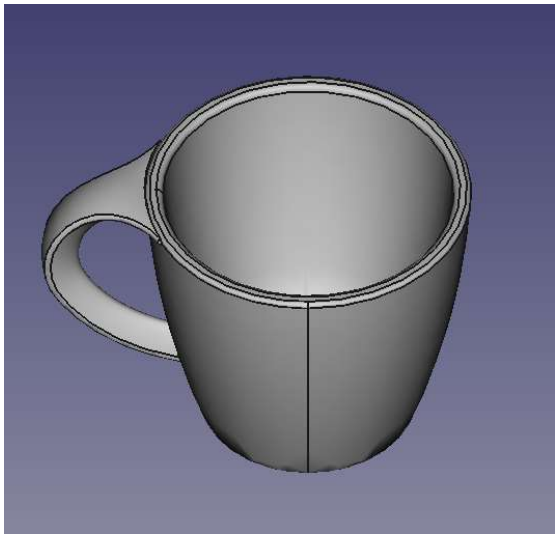


Figura 155 - Modelo 3D da chávena de ensaio. Fonte: (Própria)

Em seguida, efetuou-se um scan das pinças retas para obter o seu modelo 3D. Optou-se por esta técnica devido ao seu complexo perfil face à zona de encaixe da pinça no mecanismo. Após a sua scanerização, moldou-se o modelo de forma que, no invés de apresentar pinças retas, estas formam uma curva coincidente com a curva da chávena, tal como é possível observar na Figura 156.

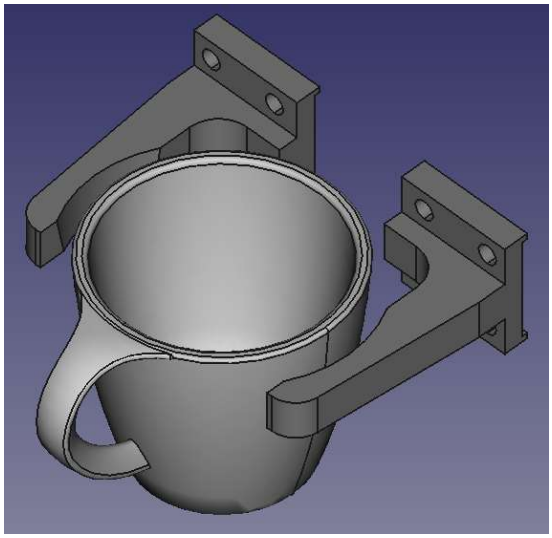


Figura 156 - Modelo de pinças adaptadas à chávena. Fonte: (Própria)

Uma vez impressas as garras em resina, efetuou-se a sua montagem no mecanismo e, por consequente, no robô, como se pode visualizar na Figura 157.

Figura 157 - Garra adaptada montada no robô. Fonte: (Própria)

Para comparar as diferenças entre a pinça reta (padrão) e a pinça adaptada à chávena (de perfil especial), desmontou-se as pinças do mecanismo de acionamento e colocou-se as duas lado a lado, junto à chávena de café. Como é possível verificar na Figura 158, a pinça de perfil especial (a), desenhada com o mesmo perfil da chávena a manipular, apresenta uma maior área de contacto (verde) com a chávena em comparação com a área de contacto da pinça padrão (vermelho). Acrescenta-se ainda que o comprimento da pinça (b) é reduzido, o que por sua vez diminui a área de contacto, dificultando ainda mais a manipulação da chávena.

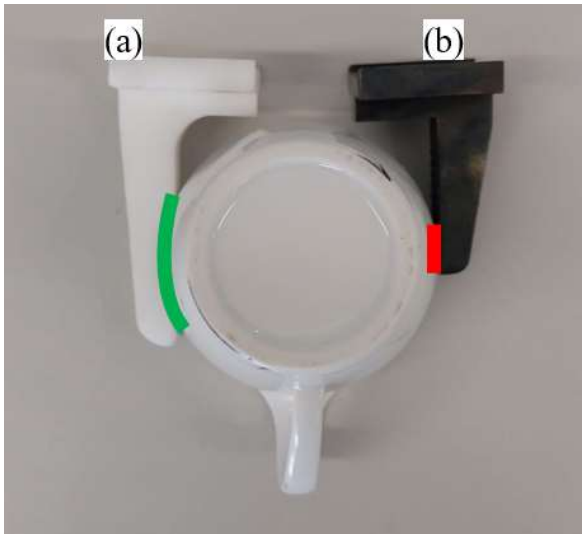


Figura 158 - Pinça de perfil especial (a) e garra padrão (b). Fonte: (Torres & Ferreira, 2022)

Realizaram-se, com a pinça padrão e a pinça de perfil especial, testes de eficiência na manipulação da chávena de café. O primeiro teste foi a manipulação da chávena na sua posição normal. Este teste provou que ambas as garras, sem existência de colisão da chávena com outro meio, conseguem manipular a chávena de café sem esta cair (ver Figura 159).

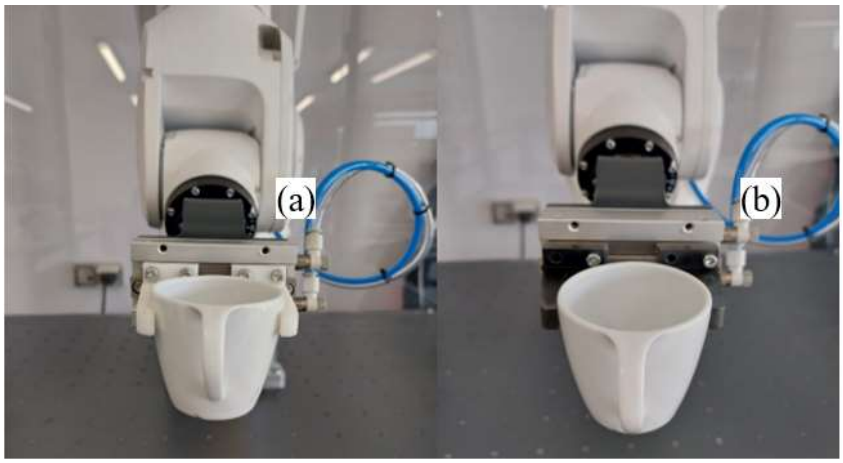


Figura 159 - Teste de agarre com pinças de perfil especial (a) e pinças padrão (b). Fonte: (Própria)

De seguida, efetuou-se o mesmo teste, porém, inverteu-se a posição da chávena de café. Uma vez invertida, a chávena apresenta uma posição menos propícia para o seu agarre devido à sua forma cónica. No teste com a pinça de perfil especial (a), a garra não demonstrou qualquer dificuldade na manipulação da chávena, porém, no teste com a pinça padrão (b) a chávena escorregou e danificou-se na sua asa quando executado o mesmo teste (ver Figura 160).

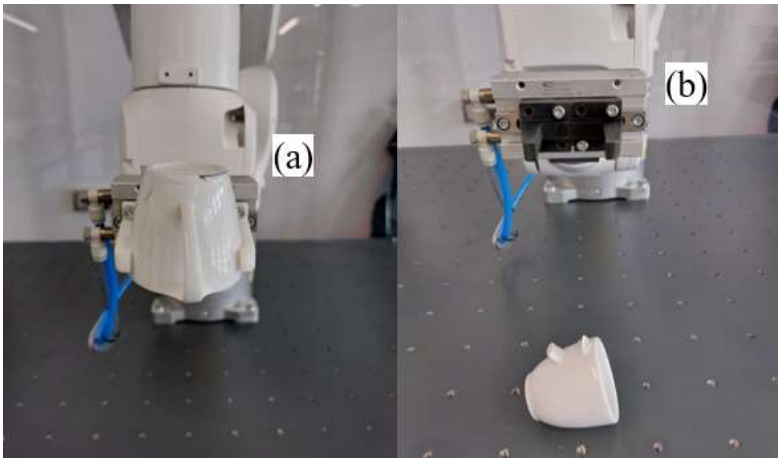


Figura 160 - Teste de agarre invertido com pinças de perfil especial (a) e com pinças padrão (b). Fonte: (Torres & Ferreira, 2022)

Com a execução deste teste final, provou-se que a projeção de pinças adaptadas ao perfil do cerâmico a manipular apresenta benefícios face à estabilidade de manipulação da peça, em comparação com as pinças padrão. Contudo, numa era fabril onde os modelos das peças a produzir são temporários, as pinças adaptadas ao perfil de uma peça que já não é produzida tornam-se obsoletas num curto espaço de vida, refletindo em custos elevados na construção e troca destas pinças.

3.4.4 - Simulação com pinças impressas

Uma vez desenvolvidas e testadas as pinças, criou-se o código de movimentação do robô de modo que este agarre a chávena na sua posição normal, ou seja, com a base inferior para baixo, de acordo com a Figura 161.

Figura 161 - Recolha da chávena. Fonte: (Própria)

De seguida, colocou-se a base da chávena junto à esponja rotativa, e efetuaram-se movimentos circulares de modo a simular o polimento da mesma, conforme a Figura 162.

Figura 162 - Polimento da base da chávena. Fonte: (Própria)

Após finalizar esta tarefa, a chávena é colocada numa posição de segurança acima da esponja e é efetuada a sua rotação, de modo que a asa da chávena esteja no sentido horizontal, tal como mostra a Figura 163.

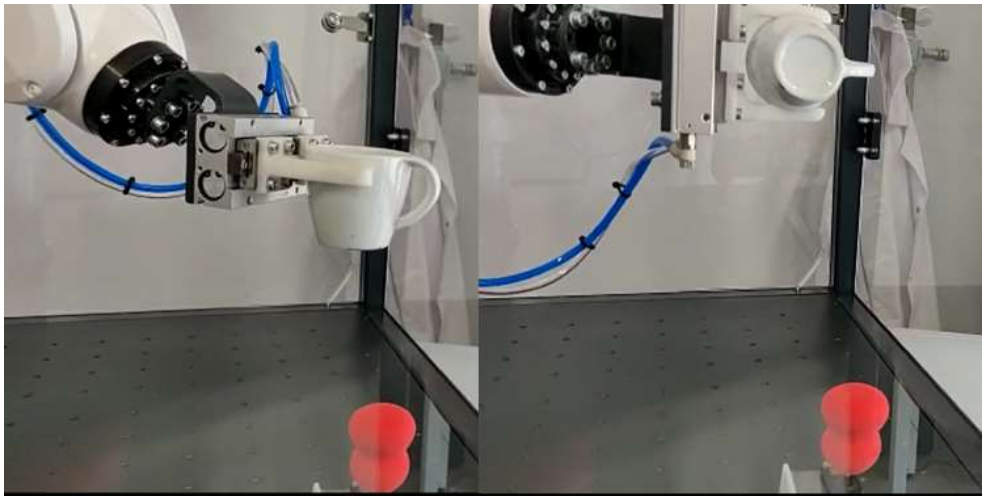


Figura 163 - Rotação da chávena. Fonte: (Própria)

Mantendo esta orientação da chávena, ela desce até encostar a sua asa à polidora e executa um movimento curvilíneo, seguindo o perfil da asa mostrado na Figura 164.

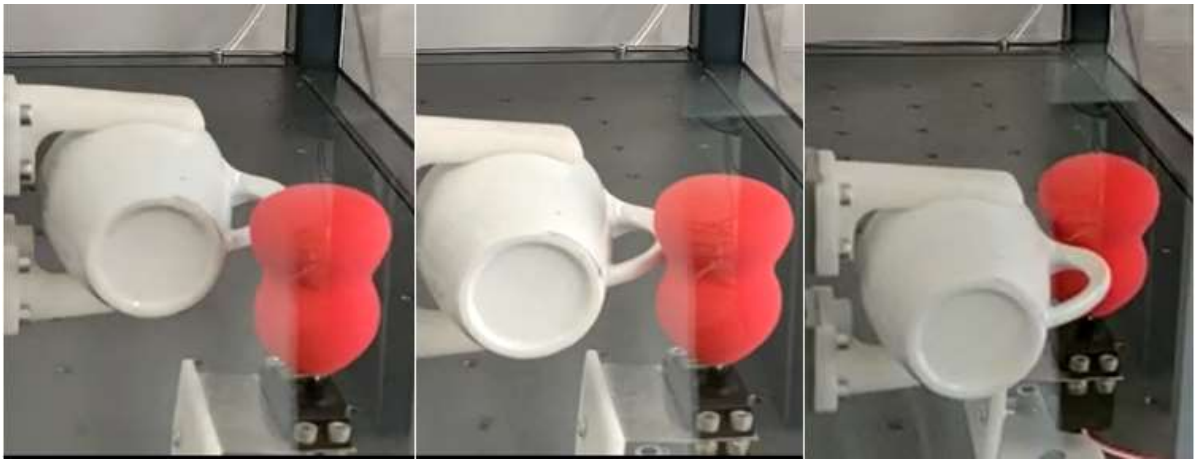


Figura 164 - Polimento da asa da chávena. Fonte: (Própria)

Uma vez executadas as duas tarefas de polimento, o robô coloca a chávena na sua posição inicial, de modo a ser possível criar ciclos destas tarefas com a mesma chávena, de acordo com a Figura 165. Esta garra provou ser bastante eficiente, porém, existem ainda alguns constrangimentos a apontar após a execução de alguns testes.

Quando aplicada força sobre a asa, a chávena tende a deslocar-se ligeiramente, refletindo-se numa alteração da posição de recolocação da chávena no ponto inicial. Não é possível saber qual a força exercida pela pinça sobre a chávena, o que pode fragilizar e até fraturar cerâmicos mais frágeis, tais como as peças cerâmicas cruas, ou em verde, que são peças que ainda não passaram pelo processo de cozedura, responsável por tornar a peça mais rígida.

Figura 165 - Chávena recolocada na sua posição inicial. Fonte: (Própria)

Por estes motivos, criou-se uma garra munida de feedback sensorial, optando pelo desenvolvimento de uma garra de dedos múltiplos, que aplicam maior estabilidade na manipulação da chávena.

3.4.5 - Conversão de unidades do sensor de carga

O primeiro passo na criação da garra com feedback sensorial foi a compra do sensor de carga FX293X-100A-0025-L (da série FX29) presente na Figura 166, capaz de medir até 125 Newtons de força, correspondente a aproximadamente 12.74 kg de força. Este sensor foi escolhido devido à sua forma compacta e rigidez e, apesar de haver sensores do mesmo modelo com menor capacidade de carga (50 N), os preços das duas eram iguais e os valores da força a aplicar com a garra eram ainda desconhecidos, optando-se por manter uma margem de segurança superior.

Figura 166 - Sensor de carga FX29. Fonte: (Própria)

O modelo do sensor utilizado é o amplificado, ou seja, os valores de tensão estão compreendidos entre 0.5VDC e 4.5VDC, apresentando uma variação de 4VDC. Apesar do sensor apresentar quatro pinos, este modelo apenas utiliza três, onde o fio vermelho (positivo) e o preto (negativo) servem para a sua alimentação e o fio amarelo é ligado ao pino de comunicação. O quarto pino, ligado ao fio branco, não é utilizado neste modelo específico, porém é utilizado juntamente com o amarelo para estabelecer comunicação pelo barramento I2C, onde o fio amarelo corresponde ao SCL (Serial Clock) e o fio branco ao SDA (Serial Data), observado na Figura 167.

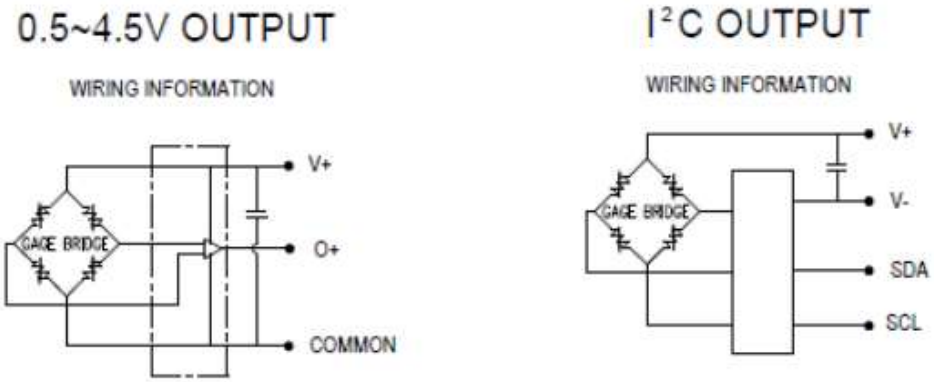


Figura 167 - Informação dos pinos do sensor amplificado (esquerda) e do sensor I2C (direita). Fonte: (Mouser)

Executou-se a montagem de um circuito de teste do sensor de carga numa placa Arduíno NANO, capaz de ler os valores instantâneos obtidos em intervalos de 1 segundo, e mostrá-los numa porta de serie e num display LCD conectado ao driver PCF8574, de modo que seja possível comunicar com o Arduíno recorrendo apenas a 4 pinos (2 de alimentação mais 2 de comunicação) no invés de 16 pinos. O esquema de montagem deste teste encontra-se na Figura 168 e o seu código Arduíno no Anexo 4.

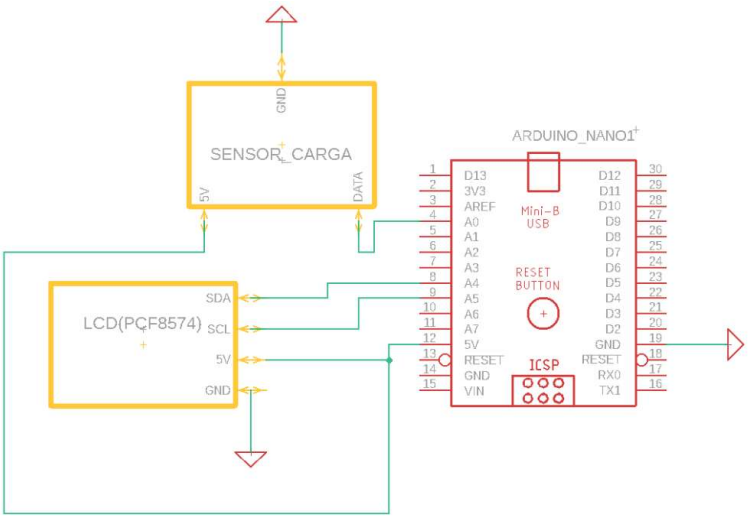


Figura 168 – Esquema de montagem de sensor de carga. Fonte: (Própria)

Com a realização deste teste, observou-se que os valores obtidos no ecrã LCD compreendiam-se na casa das centenas, sendo que, sem ser exercida qualquer força sobre o sensor, este apresentava o valor 106. Nota-se ainda que o valor mostrado no LCD aumentava conforme a força exercida sobre ele.

Como os valores obtidos não constam em nenhuma unidade do Sistema Internacional de Unidades (SI), efetuou-se uma calibração dos valores obtidos, comparando-os na Tabela 1 com valores de massas conhecidos, previamente medidos numa balança de precisão.

Tabela 1 - Massa de objetos e valores obtidos pelo sensor. Fonte: (Própria)

Objeto	Massa (g)	Valor Obtido
Sem carga	0	106
Cubo de madeira	89.9	114
Cubo de madeira c/ haste em ferro	118.3	115
Rolamento	18.4	107
Peso de 6.85 pound (LB)	3105	332
Garrafa de água	643.2	153
Alicate	155.7	118
Peso de 500g	500	143
Peso de 1kg	1000	178
Peso de 2kg	2000	252

Os valores traduzem-se no gráfico de pontos presente na Figura 169, onde se pode aferir que o seu comportamento é aproximadamente linear. O seguinte gráfico foi criado, introduzindo os dados obtidos num ficheiro Excel.

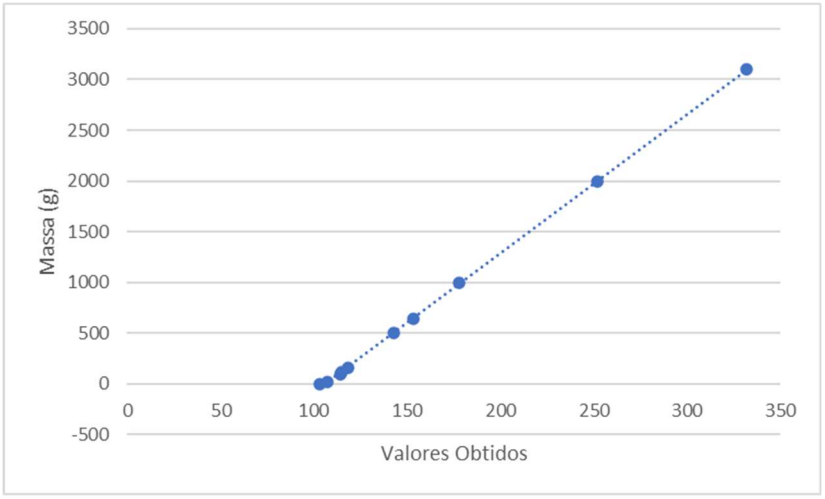


Figura 169 - Valores obtidos e reta aproximada. Fonte: (Própria)

Recorrendo às ferramentas do Excel “DECLIVE” e “INTERCETAR”, foi possível obter os valores de declive e da interceção da reta com o eixo Y de modo a se criar a seguinte equação, onde “Y” representa a massa convertida em gramas [g] e “X” representa o valor obtido pelo sensor.

$$\text{(a)} \quad Y = 13.7172X - 1453.0517\ [g]$$

Simplificou-se a equação (a), colocando o valor multiplicativo de “X” em evidência, obtendo a seguinte equação.

$$\text{(b)} \quad Y = 13.7172 * (X - 106)\ [g]$$

Através da equação (b), é possível verificar que a diferença entre o valor medido pelo sensor e o valor 106 (correspondente na Tabela 1 ao valor lido sem carga) é multiplicada diretamente com o valor colocado em evidência, possibilitando a substituição do valor 106 pelo valor de “TARA”, ou de massa zero. Esta substituição na equação permite a customização do valor zero através da leitura do valor obtido pelo sensor, sem carga exercida.

$$\text{(c)} \quad Y = 13.7172 * (X - TARA)\ [g]$$

O código do Anexo 4 foi adaptado, de modo que quando um botão de pressão é acionado, este dá indicação ao Arduíno para ler o valor do sensor de carga e substituí-lo na equação (c) pela variável “TARA”. Adicionou-se também um valor estipulado de limite mínimo de fecho no código Arduíno, para testagem desta aplicação na garra futura. O código modificado encontra-se no Anexo 5 e o novo esquema de montagem Arduíno na Figura 170.

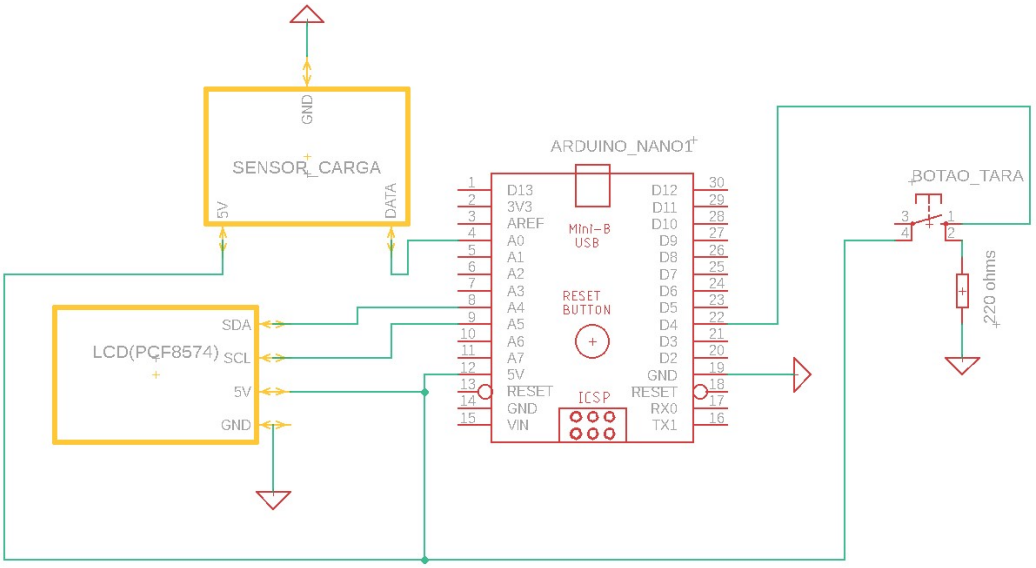


Figura 170 - Esquema de montagem com botão "TARA". Fonte: (Própria)

O LCD mostra no seu ecrã o valor lido diretamente pelo sensor e a sua conversão para quilogramas (kg) através da utilização da equação (c) dividida por 1000, tal como é possível observar na Figura 171.

Figura 171 - Valores mostrados no LCD. Fonte: (Própria)

No caso do valor de carga ultrapassar os 3 quilogramas, é mostrado no LCD uma mensagem de aviso, seguida pelo valor de carga em quilogramas medida pelo sensor, de acordo com a Figura 172.

Figura 172 - Valores acima de 3kg. Fonte: (Própria)

3.4.6 - Protótipo de pinça com sensibilidade

Para testar o funcionamento do sensor acoplado a uma garra, criou-se uma pinça mecânica, acionada pelo servo motor “Tower Pro Sg90”, com binário de 2.5 kgf/cm. Esta pinça foi montada numa caixa impressa em ABS, não sendo possível a sua montagem no braço robótico. As restantes peças (pinças, apoios e parafuso sem fim) foram impressas em resina.

Foi concebido um mecanismo de acionamento das pinças no programa FreeCAD, onde uma engrenagem aciona dois parafusos sem fim, responsáveis por movimentar as pinças. Contudo, este modelo provocou demasiadas oscilações nas suas componentes, mostrando ser pouco efetivo e, por vezes, encravando os dentes das engrenagens. Simplificou-se este mecanismo, sendo apenas necessário um parafuso sem fim ligado diretamente ao motor para acionar as duas pinças, tal como se observa na Figura 173.

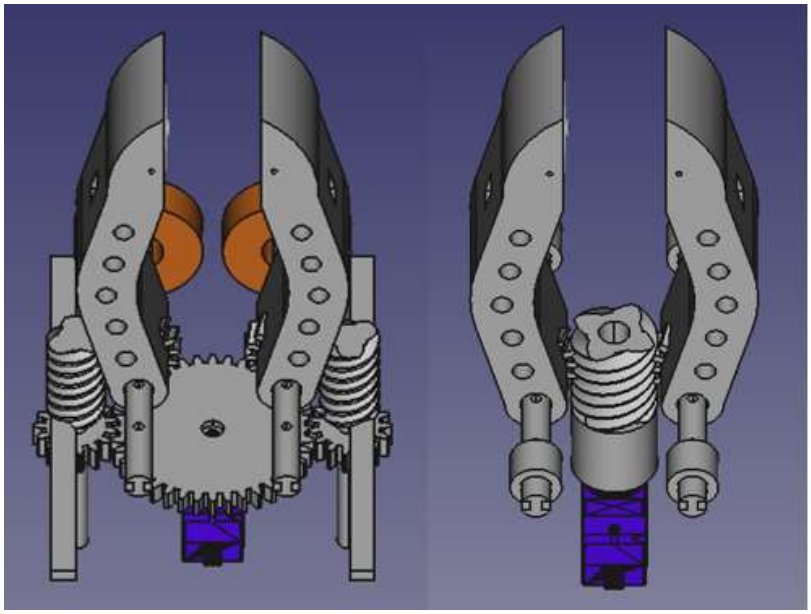


Figura 173 - Mecanismo obsoleto (esquerda) e mecanismo simplificado final (direita). Fonte: (Própria)

Após a impressão das componentes, efetuou-se a sua montagem , obtendo a garra de ensaio presente na Figura 174.

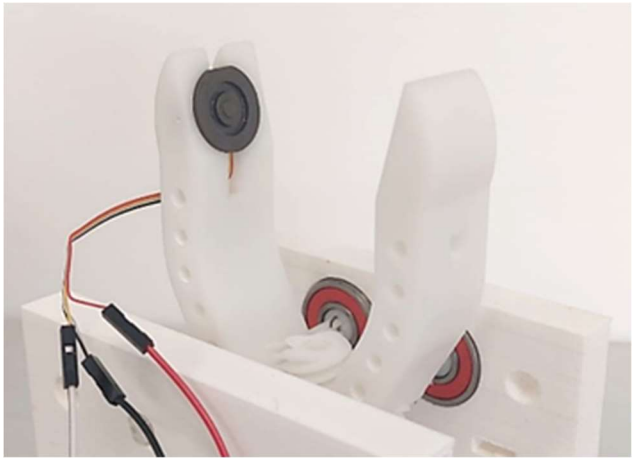


Figura 174 - Protótipo de garra de pinças mecânicas. Fonte: (Própria)

Esta garra foi programada no Arduíno NANO, munida apenas por um sensor de carga para efetuar a medição da força exercida num objeto. O esquema de montagem encontra-se na Figura 175 e o seu código de programação no Anexo 6.

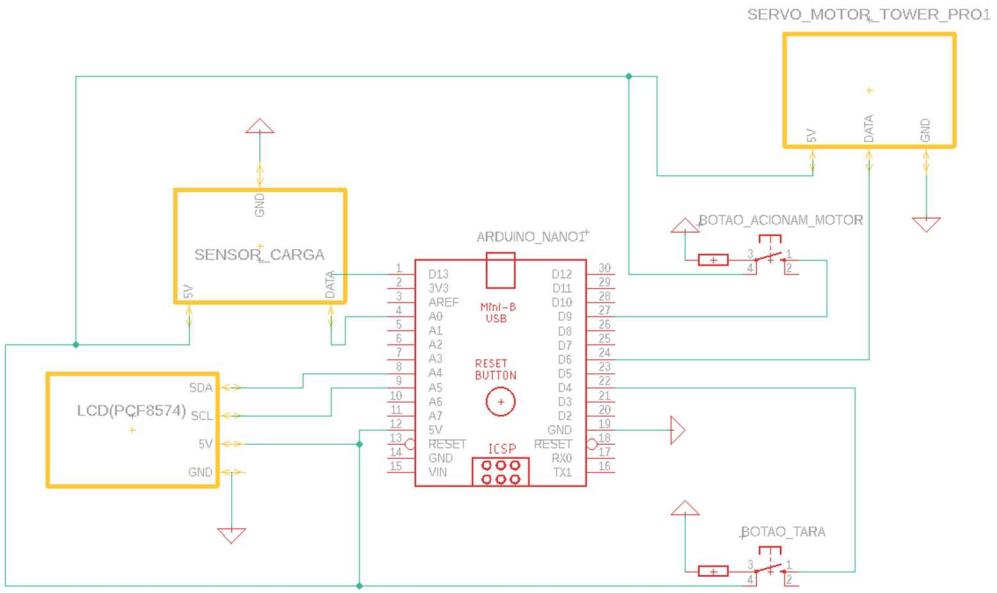


Figura 175 - Esquema de montagem Arduíno de pinça com sensibilidade. Fonte: (Própria)

Este protótipo é constituído pelo circuito presente na Figura 175, que é uma aprimora do circuito da Figura 170, incluindo todas as suas funcionalidades, com a adição de um botão e um servo motor, responsáveis por acionar e movimentar o mecanismo de abertura e fecho das pinças. Este botão funciona por impulsos, ou seja, ao premir o botão, é lido o estado atual da garra (aberta ou fechada), conforme a Figura 176 e a Figura 177, comandando o servo motor a abrir ou fechar as suas pinças.

```
23  s.write(0); //INICIA O MOTOR NA POSIÇÃO 0°
24  // Estado da garra: 0-Aberto;
25  //                  1-Fechado
26  ac_state = 0;
27  pos = 0;
```

Figura 176 - Estados da garra. Fonte: (Própria)

```
34 void loop()
35 {
36   Ler_carga();
37   ac_read = digitalRead(ac_garra);
38   if (ac_read==1) {
39     switch(ac_state) {
40       case 0:
41       //Chama função para fechar garra
42       Fechar();
43       ac_state = 1;
44       break;
45       case 1:
46       //Chama função para abrir garra
47       Abrir();
48       ac_state = 0;
49       break;
50     }
51   }
```

Figura 177 - Código de mudança de estados da garra. Fonte: (Própria)

Como consta a Figura 179, o protótipo criado apresenta dois limites de posição, sendo o primeiro limite de fecho, correspondente ao ângulo de 180º do servo motor, e o segundo limite o de abertura, estipulado para o ângulo de 0º do servo motor. O código redigido permite que a garra, quando acionada, incremente ou decremente a sua posição de ângulo 3º de cada vez, avaliando em simultâneo o valor da carga exercida no sensor. Contudo, apenas após dez incrementos (30º), é efetuada a atualização do valor de carga medida no LCD, criando assim maior conforto visual na sua leitura, uma vez que cada incremento de 3º é efetuado a cada 15 milissegundos (ms), refletindo numa atualização no LCD a cada 150 ms, de acordo com a Figura 178.

```
55 void Fechar(){
56   Serial.print("Closing");
57   while(pos < 180 && ac_state==0){ //ENQUANTO "pos" MENOR QUE 180, INCREMENTA, fechando garra
58     Ler_carga();
59     oldpos = pos;
60     while(pos < oldpos+30 && ac_state==0){
61       pos=pos+3;
62       s.write(pos); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
63       val = analogRead(read_pin);
64       delay(10);
65       carga = (13.7172 * (val-TARA))/1000;
66       if (carga > 4) {
67         ac_state=1;}
68         else {
69         ac_state=0;}
70       delay(5); //INTERVALO DE 5+10 MILISSEGUNDOS
71     }
72   }
73   Serial.print("Closed");
74 }
```

Lê, converte e atualiza valores de carga no display LCD (a cada 150ms)

Apenas lê e converte valores de carga (a cada 15ms)

Figura 178 - Código de fecho da garra. Fonte: (Própria)

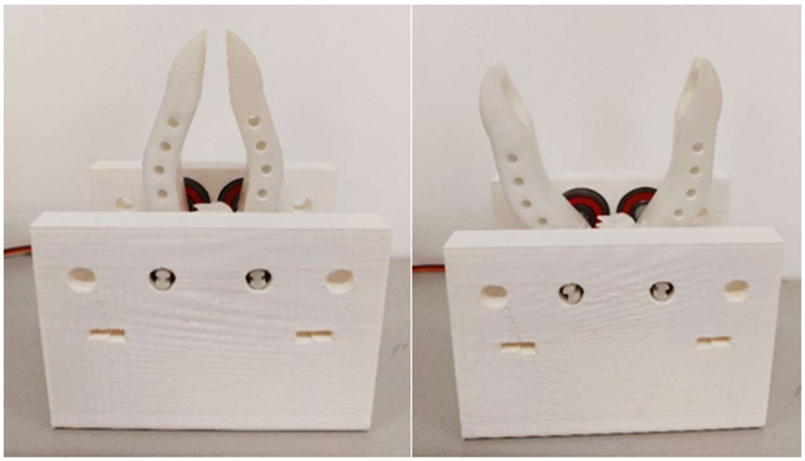


Figura 179 - Protótipo fechado (esquerda) e aberto (direita). Fonte: (Própria)

Para testar a força debitada pela garra, colocou-se uma caixa oca impressa em ABS entre as pinças e procedeu-se ao seu fecho, premindo o botão de acionamento do motor, de acordo com a Figura 180. A caixa foi criada para substituir a chávena de café, que por sua vez revelou ser demasiado pesada, escorregando durante todas as tentativas da sua manipulação. Após alguns ensaios deste tipo, concluiu-se que a força máxima debitada por este motor nas extremidades das pinças mecânicas é de 0.47 kg.

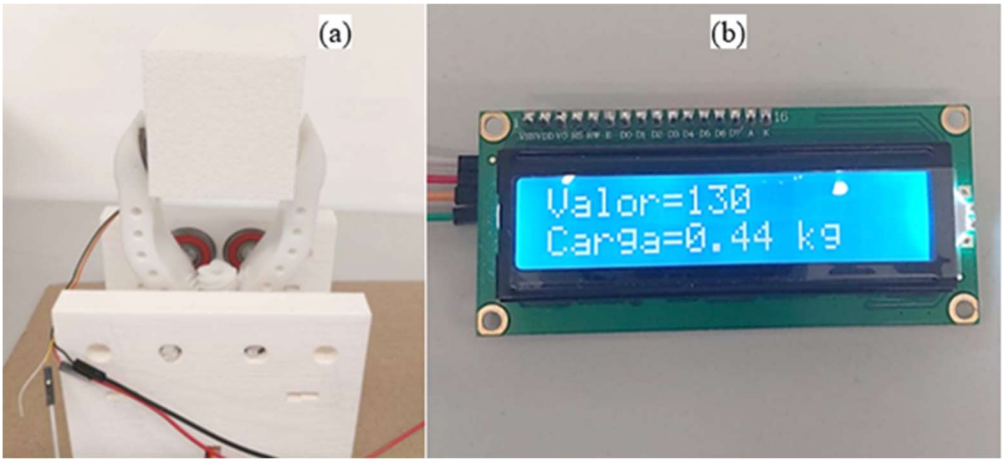
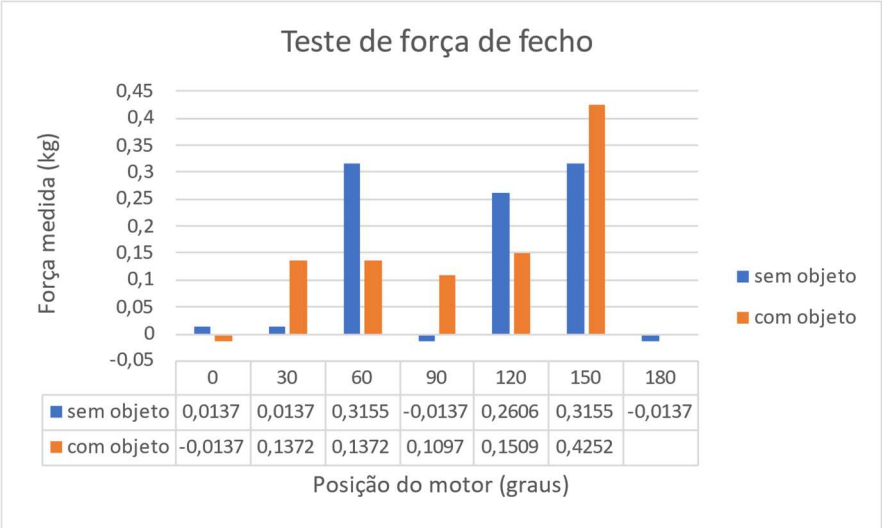


Figura 180 - Garra a agarrar uma caixa (a) e o seu valor de força (b). Fonte: (Torres & Ferreira, 2022)

Observou-se também no ecrã LCD que mesmo sem haver contacto com qualquer objeto, os valores lidos pelo sensor oscilavam, chegando por vezes a 0.35 kg, o que obrigou á criação de um valor mínimo de limite de fecho de 0.4 kg, que é perto do valor de débito máximo (0.47 kg). Pondera-se que esta oscilação de valores se deve a ruído eletrónico criado pelo mau contacto ou rotura dos cabos do sensor, que são multifilares e bastante finos, logo frágeis (ver Figura 181).



	0	30	60	90	120	150	180
sem objeto	0,0137	0,0137	0,3155	-0,0137	0,2606	0,3155	-0,0137
com objeto	-0,0137	0,1372	0,1372	0,1097	0,1509	0,4252	

Figura 181 - Valores obtidos pelo sensor durante o seu fecho sem agarrar um objeto (azul) e até agarrar um objeto (laranja). Fonte: (Própria)

No teste sem objeto, representado na Figura 181 a azul, existem três picos de carga superiores a 0.25 kg, chegando a ultrapassar os 0.30 kg. No teste com objeto, representado a laranja, as oscilações são mais constantes, porém menores. Nota-se que o servo motor parou de rodar aos 150º quando detetou que o valor de carga ultrapassava o valor de limite mínimo estipulado (0.4 kg), não tendo sido obtido nenhum valor aos 180º pois esta posição não foi alcançada neste teste.

Este protótipo é obsoleto face ao agarre da chávena de café pretendida, sendo necessária a realização de algumas melhorias, tais como a substituição por novos motores mais fortes e a eliminação do ruído existente no circuito. A posição onde é

detetada carga superior á carga mínima de limite de fecho, é a posição onde a garra fica fechada até ser comandada para abrir, o que causa problemas quando ocorre o deslize do objeto a manipular. Este deslize provoca uma diminuição da carga lida, onde a garra deveria compensar a força debitada, aumentando-a até ultrapassar o valor mínimo estipulado.

Os sensores de carga utilizados apresentam um grande defeito quanto ao seu formato, onde este mantem contacto com o objeto a medir com apenas um ponto rígido, que retira estabilidade no manuseio do objeto. Para obter maior estabilidade, é necessário adaptar este sensor através da colocação de uma esponja na sua extremidade, ou recorrer a outros sensores mais maleáveis.

3.4.7 - Garra de três dedos

A garra final ponderada para executar a tarefa do polimento é uma garra de três dedos articulados, semelhantes aos dedos de uma mão humana. Cada dedo é acionado de forma independente pelo servo motor correspondente e munido na sua extremidade por um sensor de carga que controla a carga mínima e máxima de debitada na chávena de café. O servo motor controla o movimento do seu dedo através da tração de um fio e de um elástico, ambos presos a um parafuso num dos elementos do dedo, ou seja, quando o servo motor puxa o fio, este é enrolado numa bobina, fechando a garra, quando o servo motor liberta o fio, o elástico puxa o dedo, de modo a abrir a garra.

O primeiro passo no desenvolvimento da garra foi a criação de modelos 3D dos dedos da garra, cada um constituído por duas peças que se ligam por um veio, atribuindo movimento articulado através da sua rotação.

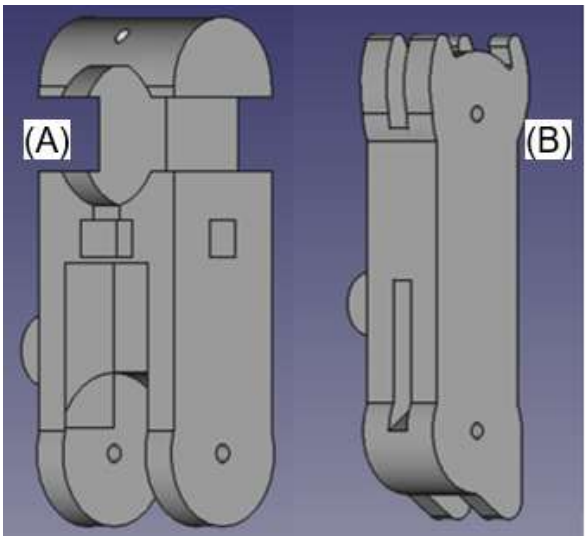


Figura 182 - Constituintes de um dedo da garra. Fonte: (Própria)

Os elementos presentes na Figura 182 são a parte superior do dedo (esquerda), denominada por peça (A), é responsável por acomodar o sensor de carga e agarrar o objeto pretendido, e a parte inferior do dedo (direita), denominada por peça (B), a cargo de ligar a parte superior do dedo à base da garra, mencionada mais a frente. A peça (B) apresenta limitadores físicos de rotação da peça (A), cruciais para a sua movimentação e manipulação da chávena de café, tal como mostra a Figura 183.

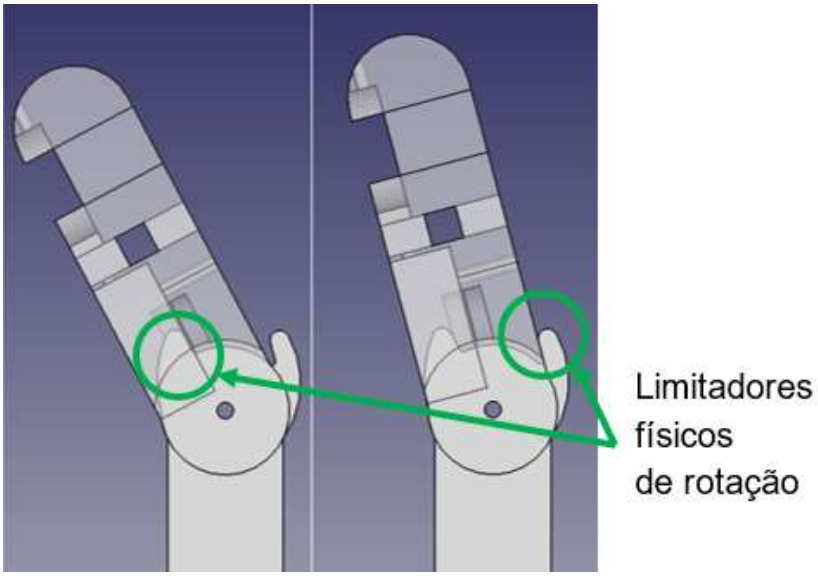


Figura 183 - Limites de posição da peça (A). Fonte: (Própria)

Criaram-se mais duas cópias destes dois elementos, formando assim os três dedos da garra. Estes dedos são unidos, também por veios, a uma base com a forma de um hexágono irregular, formando um mecanismo semelhante a uma “mão”, de acordo com a Figura 184.

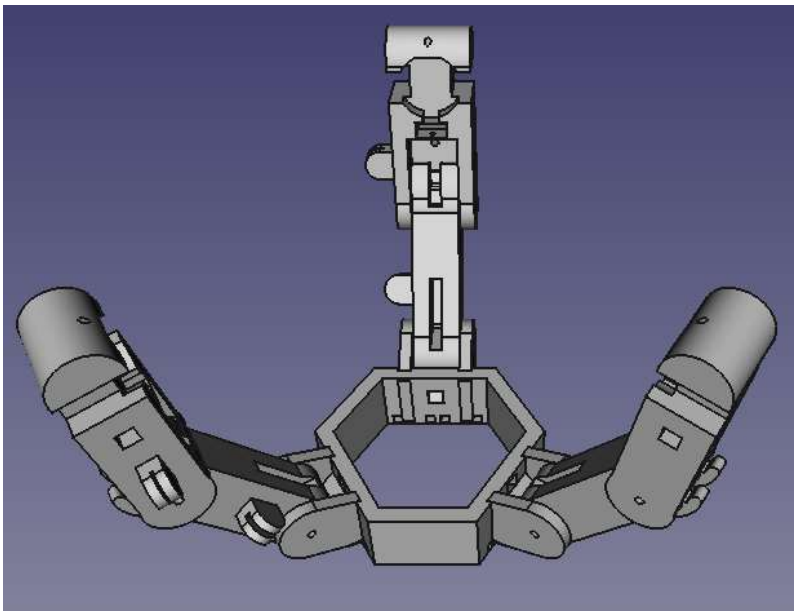


Figura 184 - Três dedos unidos a base. Fonte: (Própria)

A peça (B) apresenta na sua parte inferior um limitador de abertura, não permitindo ao conjunto dos dedos abrir mais que o desejado. São este limitador, presente na Figura 185, e o limitador da Figura 183 que determinam o movimento que a garra efetua durante a sua abertura e o seu fecho, definindo o ponto de início e de fim do referido movimento.

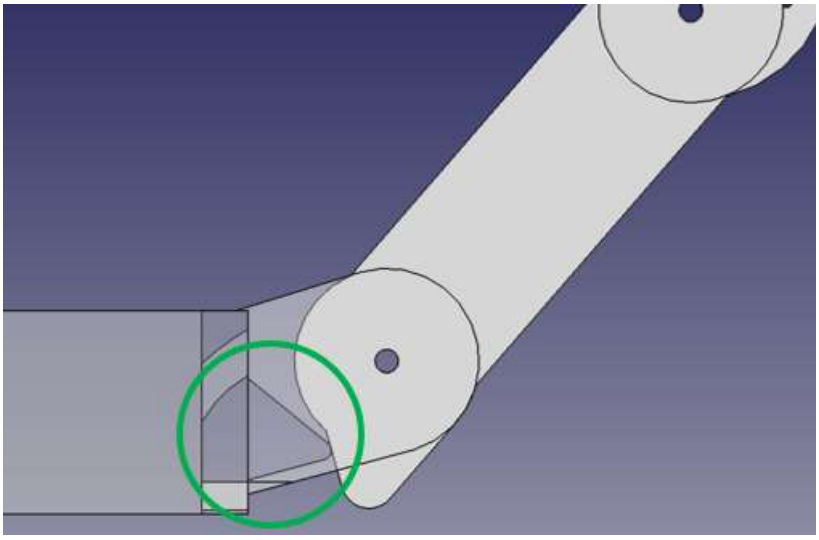


Figura 185 - Limitador de abertura da peça (B). Fonte: (Própria)

Para efetuar a tração do fio que fecha a garra, foram utilizados três servos motores de posição (180º) do modelo "*Tower Pro MG995*", presente na Figura 186, com binário de 10 kgf/cm quando operado a 6VDC, binário este quatro vezes superior ao debitado pelo motor utilizado no protótipo anterior.

Figura 186 - Servo motor Tower Pro MG995. Fonte: (https://www.elementzonline.com/towerpro-mg995-metal-gear-servo-motor)

De modo a ser possível dimensionar um apoio para este servo motor, foi criado o seu modelo 3D na aplicação FreeCAD com as suas dimensões exatas, obtendo o resultado apresentado na Figura 187. Este apoio foi posteriormente copiado mais duas vezes, obtendo um apoio por cada motor.

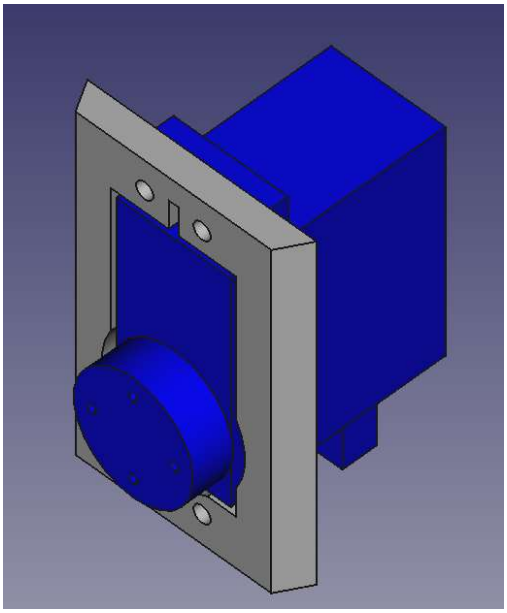


Figura 187 - Modelo 3D do servo motor (azul) e seu apoio (cinza). Fonte: (Própria)

Também recorrendo ao modelo do robô, desenvolveram-se as bobinas responsáveis por enrolar os fios que tracionam os dedos, conforme a Figura 188.

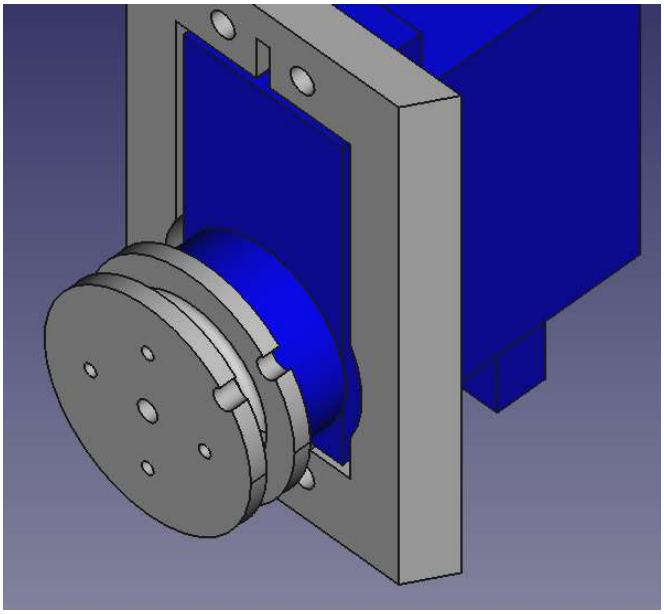


Figura 188 - Bobina do fio de tração. Fonte: (Própria)

Através da observação do modelo 3D da Figura 189, é possível observar que a posição do motor é um fator determinante para o alinhamento do fio de tração com o parafuso que o une ao dedo. Este alinhamento é importante, pois evita que o fio entre em colisão com os restantes elementos da garra que, através da sua fricção, podem danificar o material. Efetuando a rotação segundo o eixo Z do apoio do motor, foi possível simular a posição ideal para colocar este apoio.

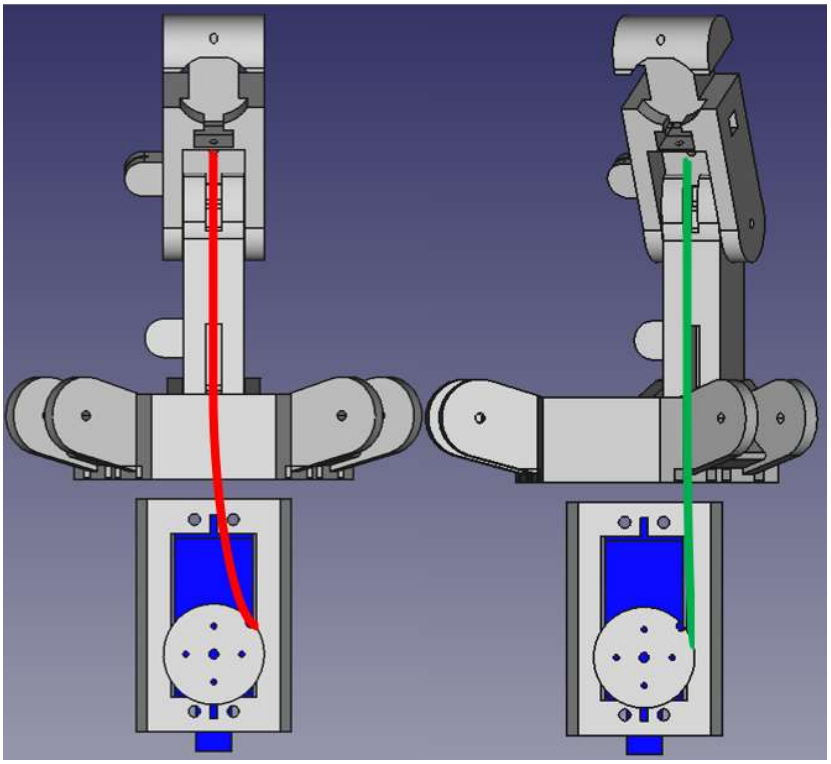


Figura 189 - Simulação da orientação do fio de tração, mudando a posição do apoio do motor. Fonte: (Própria)

Com consciência desta informação, criou-se o cilindro oco observado na Figura 190, que une os três apoios à base da “mão” da garra, à qual também foram adicionadas três ranhuras para maior estabilidade e precisão na colagem destes apoios. Esta forma foi escolhida devido ao facto de ser possível rodá-la, durante a montagem e colagem das peças da garra, de modo a obter um melhor alinhamento do fio de tração.

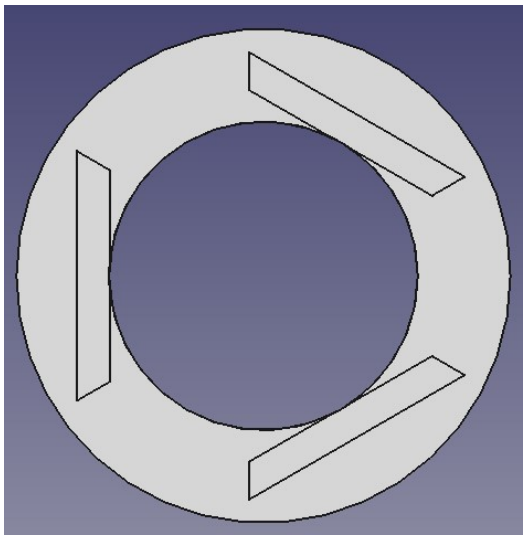


Figura 190 - Cilindro que une os apoios dos motores à "mão" da garra. Fonte: (Própria)

Foram desenvolvidas as placas presentes na Figura 191 para colocar entre os apoios dos motores, atribuindo maior estabilidade à garra e, através da atribuição de um furo no seu centro, possibilita a montagem do elemento que está aparafusado ao robô.

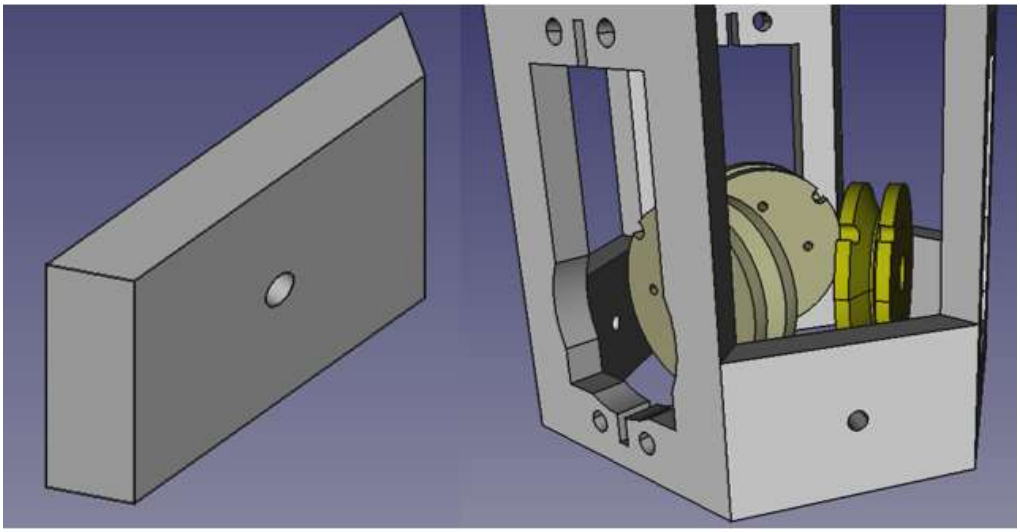


Figura 191 - Placa entre apoios dos motores. Fonte: (Própria)

Utilizando as medidas do apoio da garra presente na Figura 157, criou-se o elemento que une o robô à garra. A peça observada na Figura 192 apresenta furos para a colocação dos parafusos que a apertam ao motor e três furos coincidentes com os furos das placas entre os apoios do motor.

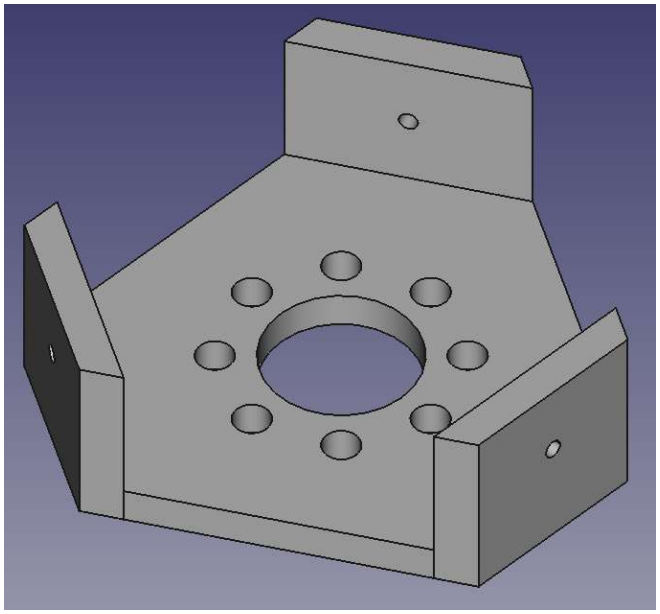


Figura 192 - Apoio que une a garra ao robô. Fonte: (Própria)

Após a impressão 3D em resina dos elementos, efetuou-se a sua montagem e colagem, obtendo, juntamente com a adição dos fios de tração, motores e três sensores de carga, a garra presente na Figura 193.

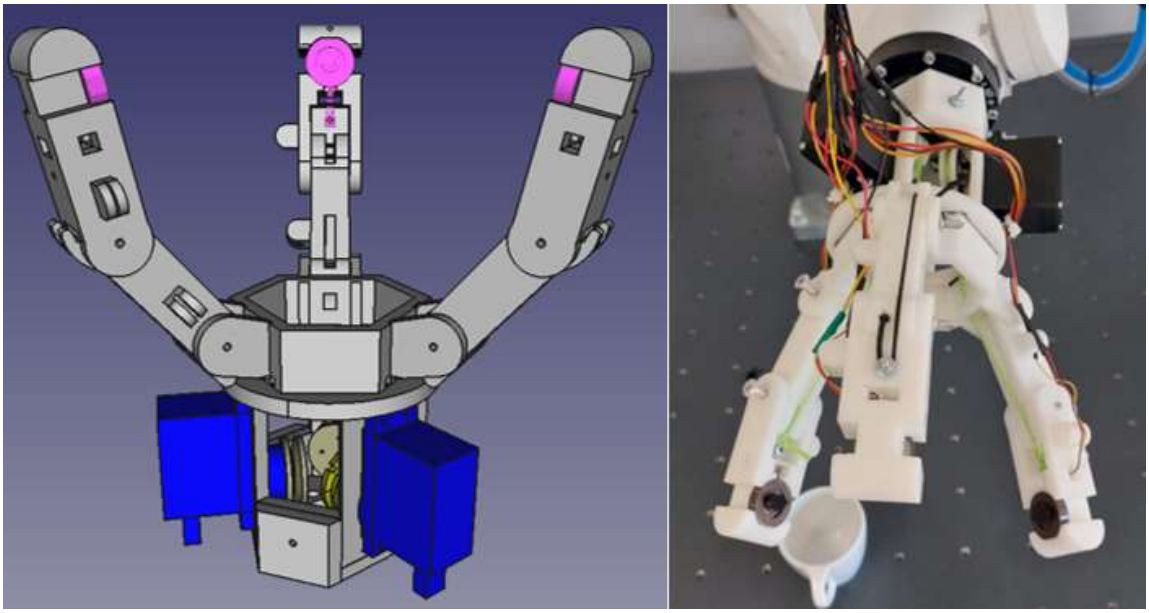


Figura 193 – Modelo e garra de três dedos. Fonte: (Própria)

3.4.8 - Código e funcionamento da garra de três dedos

O código da garra foi uma adaptação do código do Anexo 6, onde o propósito do estado da garra (ver Figura 176 e Figura 177) foi mantido, porém este foi replicado para cada um dos dedos, de acordo com a Figura 194.

```
34 // Estado da garra: 0-Fechado/A Abrir;
35 //                   1-Aberto/A Fechar
36 int ac_state_1 = 1;
37 int ac_state_2 = 1;
38 int ac_state_3 = 1; //Estado de cada dedo
```

Figura 194 - Estados de cada dedo. Fonte: (Própria)

O Loop principal do programa começa por chamar a função “Ler Carga”, mencionada mais à frente. De seguida, quando o robô envia o sinal de valor “1”, no lugar de comparar cada estado individualmente, é comparada a soma destes estados, ou seja, como o estado de cada dedo apenas pode ser “0” (fechado) ou “1” (aberto), a soma dos três estados é sempre um número inteiro compreendido entre “0” (todos os dedos estão fechados) e “3” (todos os dedos estão abertos). Nesta soma, os valores “1” e “2” indicam que um ou dois dedos não se encontram devidamente fechados, por diversos motivos que serão explicados posteriormente. Quando a soma dos valores dos estados é zero, o programa apenas lê os valores de carga de cada dedo para, no caso de um ou mais valores descer do valor estipulado de fecho, podendo esta descida ser causada pelo facto de o objeto a ser manipulado escorregar, atribuir o valor de estado “1” ao dedo correspondente, fazendo com que no próximo ciclo, entre na função de fecho, tal como se pode observar na Figura 195.

```
70 void loop()
71 {
72   Ler_carga();
73   ac_read = digitalRead(ac_garra);
74   Serial.println(ac_read);
75   if (ac_read == 1) {
76     if (ac_state_1 + ac_state_2 + ac_state_1 > 0) {
77       Fechar(); //Chama função para fechar garra
78       ac_state_1 = 0;
79       ac_state_2 = 0;
80       ac_state_3 = 0;
81       delay(10);
82     }
83     if (ac_state_1 + ac_state_2 + ac_state_1 == 0) {
84       Ler_carga();
85       if (carga_1 <= carga_fecho) ac_state_1 = 1;
86       else ac_state_1 = 0;
87       if (carga_2 <= carga_fecho) ac_state_2 = 1;
88       else ac_state_2 = 0;
89       if (carga_3 <= carga_fecho) ac_state_3 = 1;
90       else ac_state_3 = 0;
91       delay(10); //INTERVALO DE 10 MILISSEGUNDOS
92     }
93   }
```

Figura 195 – Loop principal (). Fonte: (Própria)

As funções “Ler_carga” e “Calcular_Carga”, mostradas na Figura 197, são, como o nome indica, funções cujo objetivo é ler os valores de carga dos sensores colocados nos dedos da garra e convertê-los para gramas. A função de cálculo da carga também lê o valor de um potenciómetro que, por sua vez, pode ser alterado de acordo com a preferência do utilizador, sendo este o valor correspondente ao valor mínimo que a garra deve exercer para segurar o objeto. A função “Ler_carga” mostra a leitura destes valores no display LCD, de acordo com a Figura 196, sendo que os valores “C1”, ”C2” e ”C3” são os valores de carga em cada dedo da garra e o valor “F” é o valor mínimo de limite de carga para segurar o objeto.

Figura 196 - Valores de carga em cada dedo do sensor e valor mínimo de força. Fonte: (Própria)

```
156 void Ler_carga() {
157   Calcular_Carga();
158   lcd.clear();
159   lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
160   lcd.print("C1="); //IMPRIME O TEXTO NO DISPLAY LCD
161   lcd.print(carga_1);
162   lcd.setCursor(8, 0);
163   lcd.print("C2=");
164   lcd.print(carga_2);
165   lcd.setCursor(0, 1); //POSIÇÃO DO CURSOR
166   lcd.print("C3="); //IMPRIME O TEXTO NO DISPLAY LCD
167   lcd.print(carga_3);
168   lcd.setCursor(8, 1);
169   lcd.print("F=");
170   lcd.print(carga_fecho);
171   lcd.setCursor(13, 1);
172   lcd.print("[g]");
173 }
174
175 void Calcular_Carga() {
176   carga_fecho = (analogRead(potenc_read)) / 2;
177   val_1 = analogRead(read_pin_1);
178   val_2 = analogRead(read_pin_2);
179   val_3 = analogRead(read_pin_3);
180   delay(5);
181   carga_1 = (13.7172 * (val_1 - TARA_1));
182   carga_2 = (13.7172 * (val_2 - TARA_2));
183   carga_3 = (13.7172 * (val_3 - TARA_3));
184 }
```

Figura 197 - Funções de leitura e cálculo de carga. Fonte: (Própria)

Tal como é possível observar na Figura 198, a função “Fechar” apresenta como primeiro passo a leitura do estado do robô, de modo a obter a confirmação de que o robô continua a enviar o valor de “1”, entrando de seguida num Loop que atua enquanto a soma dos valores de estado dos dedos for superior a “1” e o sinal enviado pelo robô for “1”. Após ler os valores de carga, compara-os individualmente com o valor de limite mínimo de agarre do objeto, sendo que se o valor lido no sensor do dedo for superior ao de limite ou se o valor de posição desse dedo for igual ao de limite de fecho, o programa assume que este dedo já se encontra devidamente fechado, caso contrário, permanece em processo de fecho. Após esta comparação, é executado um processo semelhante ao demonstrado na Figura 178, porém a atualização do display dá-se a cada 20 decrementos, atualizando-o a cada 300ms.

```
107 void Fechar() {
108   ac_read = digitalRead(ac_garra);
109   Serial.println(ac_read);
110   while (ac_state_1 + ac_state_2 + ac_state_3 > 0 && ac_read == 1) {
111     Ler_carga();
112     if (carga_1 >= carga_fecho || pos_1 == 180 - lim_pos) ac_state_1 = 0;
113     else ac_state_1 = 1;
114     if (carga_2 >= carga_fecho || pos_2 == 180 - lim_pos) ac_state_2 = 0;
115     else ac_state_2 = 1;
116     if (carga_3 >= carga_fecho || pos_3 == 180 - lim_pos) ac_state_3 = 0;
117     else ac_state_3 = 1;
118     delay(5); //INTERVALO DE 5 MILISSEGUNDOS
119     oldpos_1 = pos_1;
120     oldpos_2 = pos_2;
121     oldpos_3 = pos_3;
122     while (pos_1>oldpos_1-20 && pos_2>oldpos_2-20 && pos_3>oldpos_3-20 && ac_state_1+ac_state_2+ac_state_3>0 && ac_read==1){
123       Calcular_Carga();
124       if (carga_1 >= carga_fecho || pos_1 <= 180 - lim_pos) ac_state_1 = 0;
125       else ac_state_1 = 1;
126       if (carga_2 >= carga_fecho || pos_2 <= 180 - lim_pos) ac_state_2 = 0;
127       else ac_state_2 = 1;
128       if (carga_3 >= carga_fecho || pos_3 <= 180 - lim_pos) ac_state_3 = 0;
129       else ac_state_3 = 1;
130       pos_1 = pos_1 - ac_state_1;
131       s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
132       pos_2 = pos_2 - ac_state_2;
133       s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
134       pos_3 = pos_3 - ac_state_3;
135       s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
136       delay(10);
137       ac_read = digitalRead(ac_garra);
138       delay(5); //INTERVALO DE 5 MILISSEGUNDOS
139     }
140     ac_read = digitalRead(ac_garra);
141     Serial.println(ac_read);
142   }
143 }
```

Figura 198 - Função "Fechar" da garra. Fonte: (Própria)

Voltando ao Loop principal, quando o valor enviado pelo robô for “0”, é chamada a função de abertura da garra, conforme a Figura 199.

```
94   Ler_carga();
95   ac_read = digitalRead(ac_garra);
96   Serial.println(ac_read);
97   if (ac_read == 0) {
98     Abrir(); //Chama função para abrir garra
99     ac_state_1 = 1;
100    ac_state_2 = 1;
101    ac_state_3 = 1;
102  }
103  Ler_carga();
104  delay(450);
105 }
```

Figura 199 - Loop principal (parte 2). Fonte: (Própria)

A função de abertura da garra, tal como é possível observar na Figura 200, é mais simples que a função de fecho, uma vez que após atualizar o valor no LCD, escreve o valor de “180” em todos os servos motores, de modo a abrir os dedos da garra rapidamente.

```
145 void Abrir() {
146   Ler_carga();
147   pos_1 = 180;
148   pos_2 = 180;
149   pos_3 = 180;
150   s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
151   s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
152   s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
153   delay(20); //INTERVALO DE 20 MILISSEGUNDOS
154 }
```

Figura 200 - Função de abertura da garra. Fonte: (Própria)

Foi utilizada uma placa Arduíno NANO, onde o circuito elétrico da garra encontra-se na Figura 201, sendo este munido por três servos motores, três sensores de força, um LCD, um potenciómetro e um relé de cinco pinos. Posteriormente, foi colocado um LED junto à saída do relé, com a finalidade de indicar quando o robô envia o sinal “1”, responsável por acionar o fecho da garra.

Devido ao facto de o robô apresentar um valor de 24VDC na sua saída digital, valor este cinco vezes superior ao suportado pelas entradas digitais do Arduíno, foi utilizado um relé que, quando energizado pelo sinal de saída do robô, fecha o contacto “NO”, fazendo passar a energia proveniente dos 5VDC disponibilizados pela placa que, por sua vez, é alimentada a 7.5VDC.

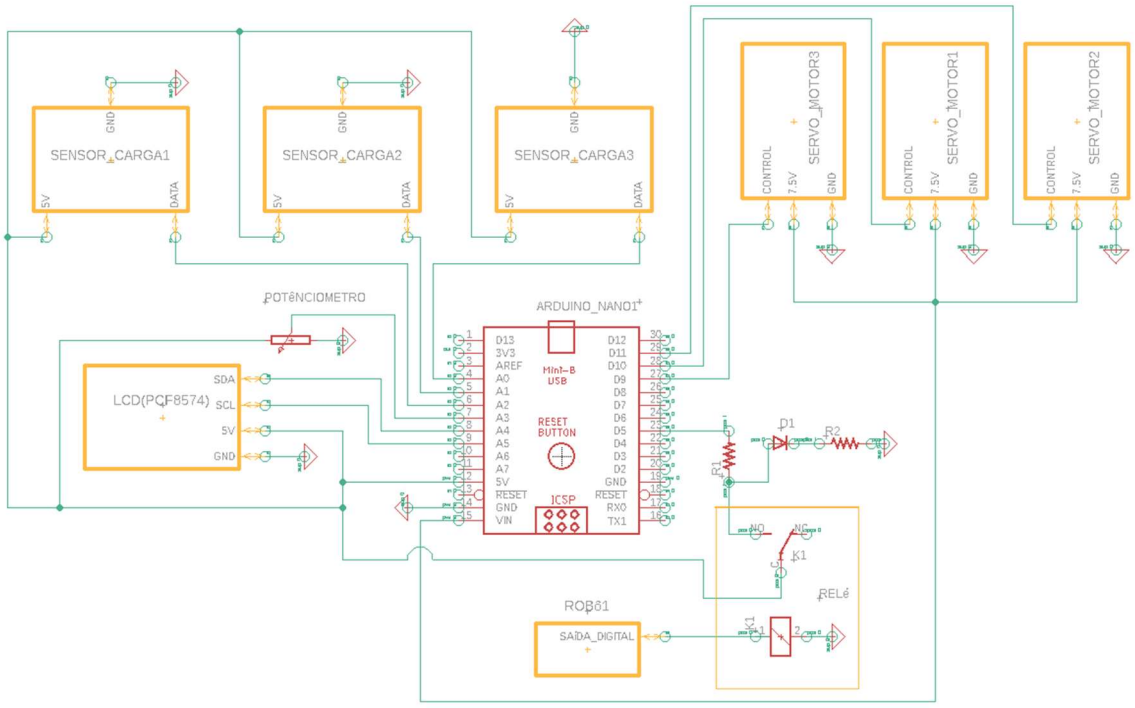


Figura 201 - Esquema de montagem Arduíno de garra com três dedos. Fonte: (Própria)

O potenciómetro envia um sinal entre “1028” e “0”, porém este sinal é dividido por dois, tal como observado na Figura 197, limitando a atribuição do valor mínimo de fecho dos dedos da garra para um valor até 514g, o que por consequente, limita o valor máximo de carga exercida pelos dedos.

3.4.9 - Simulação de teste com chávena de café (sem limite máximo de força)

Uma vez terminada a construção da garra, criou-se um programa no robô KUKA que simula o polimento da base da chávena e, no final da sua execução, vira a base polida para a frente, de modo a testar também a estabilidade da garra ao manipular a chávena numa diferente orientação, voltando por fim, a colocá-la na posição inicial, para ser possível executar o programa em Loop.

O robô começa o programa por colocar a garra junto à chávena para, após acionar a saída digital responsável por excitar o relé, acionar o fecho da garra, conforme mostra a Figura 202.

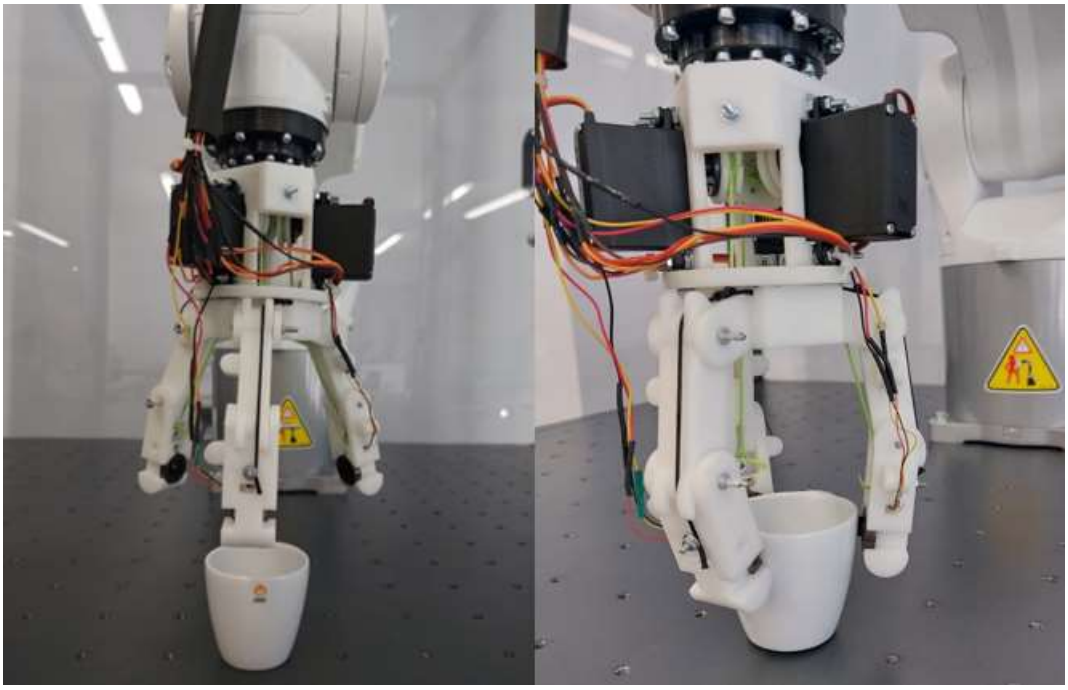


Figura 202 - Agarre da chávena com garra de três dedos. Fonte: (Própria)

Em seguida, o robô eleva a chávena, colocando-a numa posição de segurança sobre a esponja polidora, efetuando depois a sua descida até ao ponto em que a base da chávena colide com a esponja. Após descer o seu braço, o robô efetua duas voltas segundo o eixo Z da garra, de modo que a base da chávena seja polida na sua totalidade, elevando-a no fim para uma posição de segurança, tal como é possível observar na Figura 203.

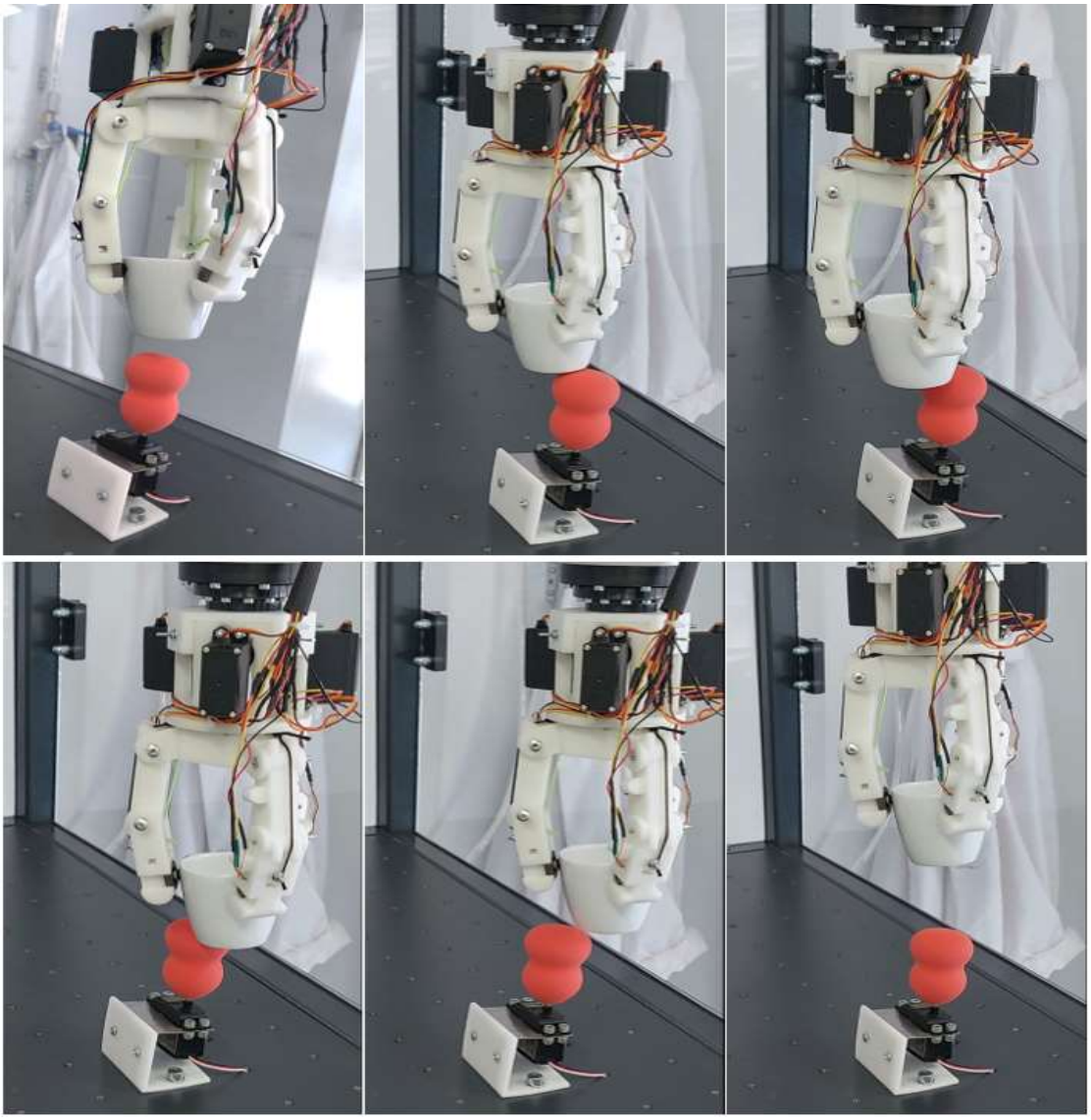


Figura 203 - Polimento da chávena manipulada pela garra de três dedos. Fonte: (Própria)

Uma vez efetuado o polimento da base da chávena, o robô move-a para a sua frente e roda-a, tal como mostra a Figura 204, sendo assim possível observar no mesmo instante a base da chávena polida e o comportamento da garra ao manipular a chávena numa diferente orientação.

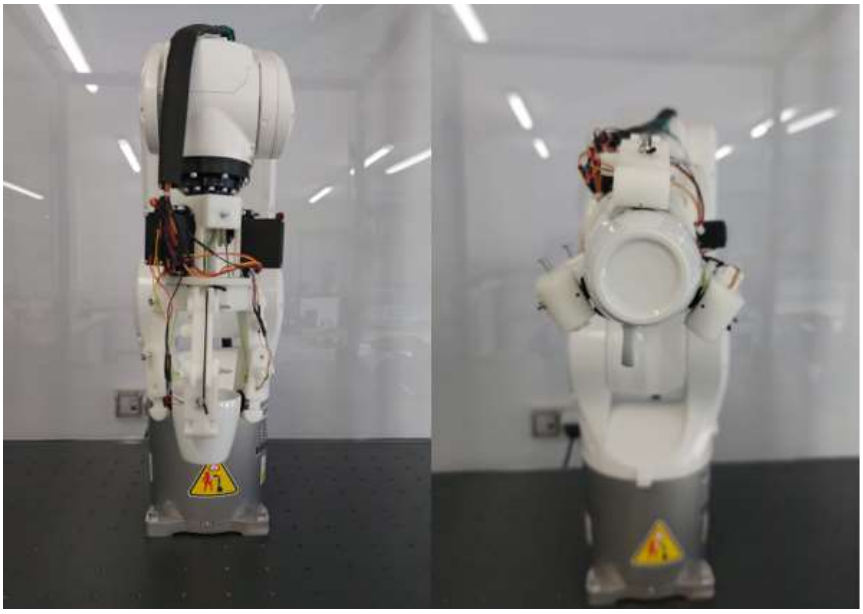


Figura 204 - Mostragem da base da chávena. Fonte: (Própria)

Apos alguns segundos nesta posição, o robô volta a colocar a chávena na sua posição original, retornando em seguida à posição de começo do programa, de modo que seja possível executar um Loop automático, de acordo com a Figura 205.

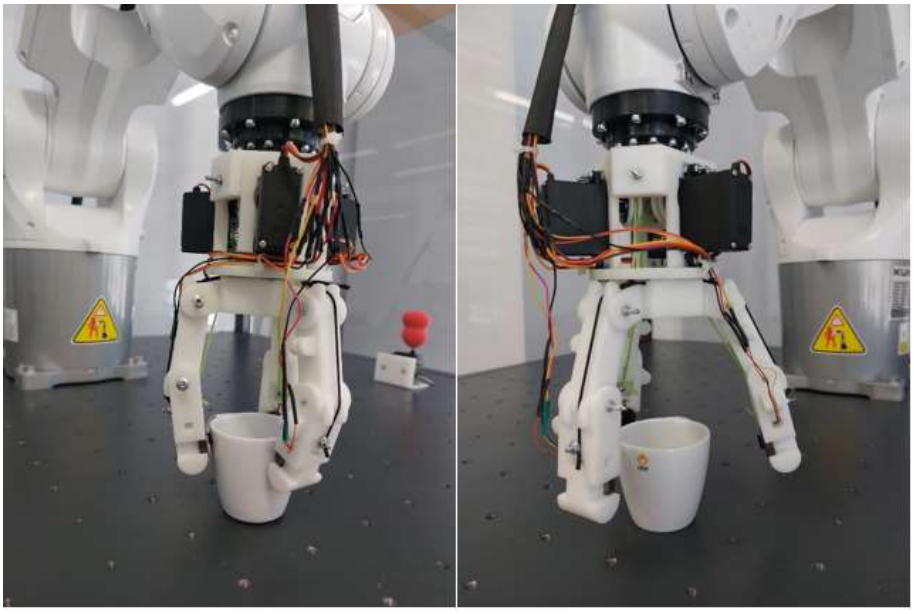


Figura 205 - Colocação da chávena com garra de três dedos. Fonte: (Própria)

No decorrer deste teste, o valor de carga mínima para fecho da garra foi regulado para 160g e, como mostra a Figura 206, os valores lidos pelos sensores nos dedos da garra são superiores a este valor, comprovando a estabilidade na sua manipulação.

Figura 206 - Leitor LCD com valores de cargas do teste sem limite máximo de força. Fonte: (Própria)

Contudo, a ausência da atribuição de um limite de carga máxima exercida pelos dedos da garra pode causar danos nas peças a serem manipuladas. O comportamento desta garra face à mudança brusca de valores de carga mínima de fecho pode ser observado na sequência presente na Figura 207, onde a chávena de café esteve a ser manipulada na sua duração. Durante esta sequência, após a chávena ser agarrada com a força mínima de 160g, reduziu-se o valor mínimo requerido para 79g, não apresentando qualquer alteração na força exercida pela garra. Em seguida, aumentou-se o valor mínimo de carga para 179g, o que por sua vez fez atuar os dedos da garra, que efetuaram mais força sobre a chávena.

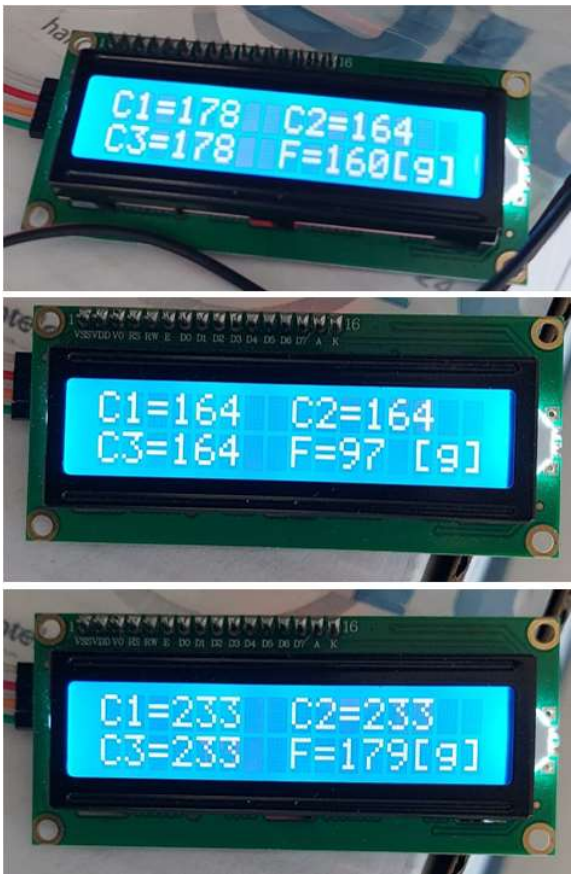


Figura 207 - Força exercida pela garra mediante variação no potenciómetro (sem limite máximo). Fonte: (Própria)

3.4.10 - Simulação final com chávena de café (com limite máximo de força)

O código do Anexo 7 desenvolvido para a garra impõe um limite mínimo de carga exercida pelos seus dedos, porém não acarreta qualquer segurança que estipule o seu limite máximo, podendo isto causar danos nos materiais mais frágeis. Tendo isto em mente, adicionou-se a este código a atribuição de um limite máximo de força de duas vezes o valor obtido pelo potenciómetro, criando assim o código presente no Anexo 8.

As diferenças entre estes dois códigos, visíveis na Figura 208 e na Figura 209, são que, no Loop principal, existe mais uma condição para atribuir o estado "1" ao dedo que ultrapassa o limite máximo estipulado e, durante a função de fecho, o servo motor responsável pelo dedo que exerce força excessiva vai incrementar um passo à sua posição atual, de modo a abrir ligeiramente o dedo em questão sem que o largue completamente.

```
68 void loop()
69 {
70   Ler_carga();
71   ac_read = digitalRead(ac_garra);
72   if (ac_read == 1) {
73     if (ac_state_1 + ac_state_2 + ac_state_1 > 0) {
74       Fechar(); //Chama função para fechar garra
75       ac_state_1 = 0;
76       ac_state_2 = 0;
77       ac_state_3 = 0;
78       delay(10);
79     }
80     if (ac_state_1 + ac_state_2 + ac_state_1 == 0) {
81       Ler_carga();
82       if (carga_1 <= carga_fecho || carga_1 > carga_fecho*2) ac_state_1 = 1;
83       else ac_state_1 = 0;
84       if (carga_2 <= carga_fecho || carga_2 > carga_fecho*2) ac_state_2 = 1;
85       else ac_state_2 = 0;
86       if (carga_3 <= carga_fecho || carga_3 > carga_fecho*2) ac_state_3 = 1;
87       else ac_state_3 = 0;
88       delay(10); //INTERVALO DE 10 MILISSEGUNDOS
89     }
90   }
```

Condição de carga máxima é 2x o valor de carga mínima

Figura 208 - Condição de valor de carga máxima. Fonte: (Própria)

```
103 void Fechar() {
104   ac_read = digitalRead(ac_garra);
105   while (ac_state_1 + ac_state_2 + ac_state_3 > 0 && ac_read == 1) {
106     Ler_carga();
107     if (carga_1 >= carga_fecho || pos_1 == 180 - lim_pos) {
108       if (carga_1 > carga_fecho*2) {
109         pos_1 = pos_1 + 1;
110         s1.write(pos_1);
111       }
112       ac_state_1 = 0;
113     }
114     else ac_state_1 = 1;
115     if (carga_2 >= carga_fecho || pos_2 == 180 - lim_pos) {
116       if (carga_2 > carga_fecho*2) {
117         pos_2 = pos_2 + 1;
118         s2.write(pos_2);
119       }
120       ac_state_2 = 0;
121     }
122     else ac_state_2 = 1;
123     if (carga_3 >= carga_fecho || pos_3 == 180 - lim_pos) {
124       if (carga_3 > carga_fecho*2) {
125         pos_3 = pos_3 + 1;
126         s3.write(pos_3);
127       }
128       ac_state_3 = 0;
129     }
130     else ac_state_3 = 1;
```

Incremento no servo 1 quando a carga máxima é excedida

Incremento no servo 2 quando a carga máxima é excedida

Incremento no servo 3 quando a carga máxima é excedida

Figura 209 - Abertura dos dedos com excesso de carga. Fonte: (Própria)

O código da garra foi de seguida atualizado, tendo-se repetido o processo de polimento com o mesmo sucesso. Testou-se também o comportamento deste código face à mudança brusca de valores de carga mínima e máxima de fecho, tal como é possível observar na sequência da Figura 210, onde a chávena de café esteve a ser manipulada durante o recorrer do teste. Nesta sequência, após a chávena ser agarrada com a força mínima de 160g (1), reduziu-se o valor mínimo requerido para 52g (2), apresentando desta vez alteração na força exercida pelos três dedos da garra, que por sua vez reduziu para dentro dos valores estipulados. Em seguida, aumentou-se o valor mínimo de carga para 135g (3), o que mais uma vez, fez atuar os dedos da garra que exerceram mais força sobre a chávena.

Reduziu-se mais duas vezes, de modo que apenas um ou dois dedos atuassem, ou seja, após reduzir a carga mínima para 57g (4), esperou-se que os valores das cargas estabilizassem para reduzir a carga de limite apenas o suficiente, correspondendo a 45g (5), para que o dedo que está a exercer maior força sobre a chávena compense o seu excesso de força.

Por último, reduziu-se o valor para 37g (6) com a finalidade de testar a estabilidade da garra na aplicação de forças reduzidas, que revelaram ser bastante instáveis em testes anteriores na aplicação de um limite máximo de carga de valor inferior, tendo sido concluído que apenas se observa estabilidade quando o valor de carga máxima de fecho é pelo menos duas vezes superior ao valor de carga mínima de fecho.

Figura 210 - Força exercida pela garra mediante variação no potenciómetro (com limite máximo). Fonte: (Própria)

A instabilidade referida no parágrafo anterior deve-se ao facto de haver pouca diferença entre o valor mínimo e máximo de carga de fecho, o que causou constantes incrementos e decrementos no mesmo dedo, destabilizando também os restantes dedos, que, ao compensar a força exercida na chávena, mudavam a sua posição face à garra, o que torna a criação de ciclos obsoleta.

3.4.11 - Testes com chávena verde

Os seguintes testes foram executados com uma chávena de café idêntica à utilizada anteriormente, porém crua e sem asa, relembrando que a peça cerâmica crua, ou verde, é frágil, pois ainda não passou pelo processo de cozedura que, posteriormente, atribui rigidez à peça. Devido à sua fragilidade e menor massa, reduziu-se a força mínima de carga para 42g, de acordo com a Figura 211.

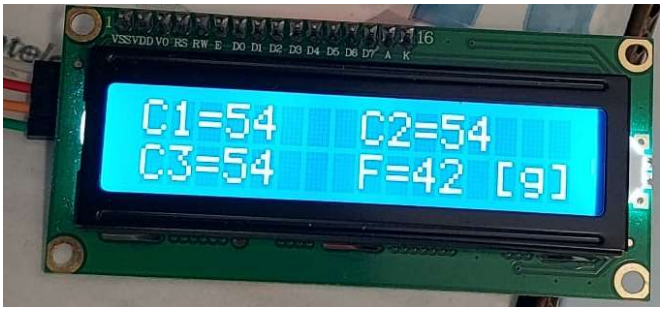


Figura 211 - Valores de carga para teste com chávena verde. Fonte: (Própria)

Uma vez agarrada, a chávena crua passou pelo mesmo processo de manipulação que a chávena de café anteriormente utilizada, presentes na Figura 212 e na Figura 213, sem apresentar dificuldades no seu manuseio, e sem esta partir devido à carga aplicada.

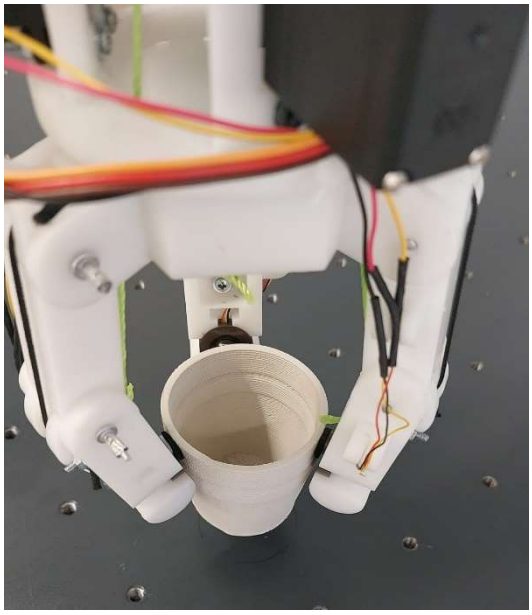


Figura 212 - Garra de três dedos a recolher chávena crua. Fonte: (Própria)

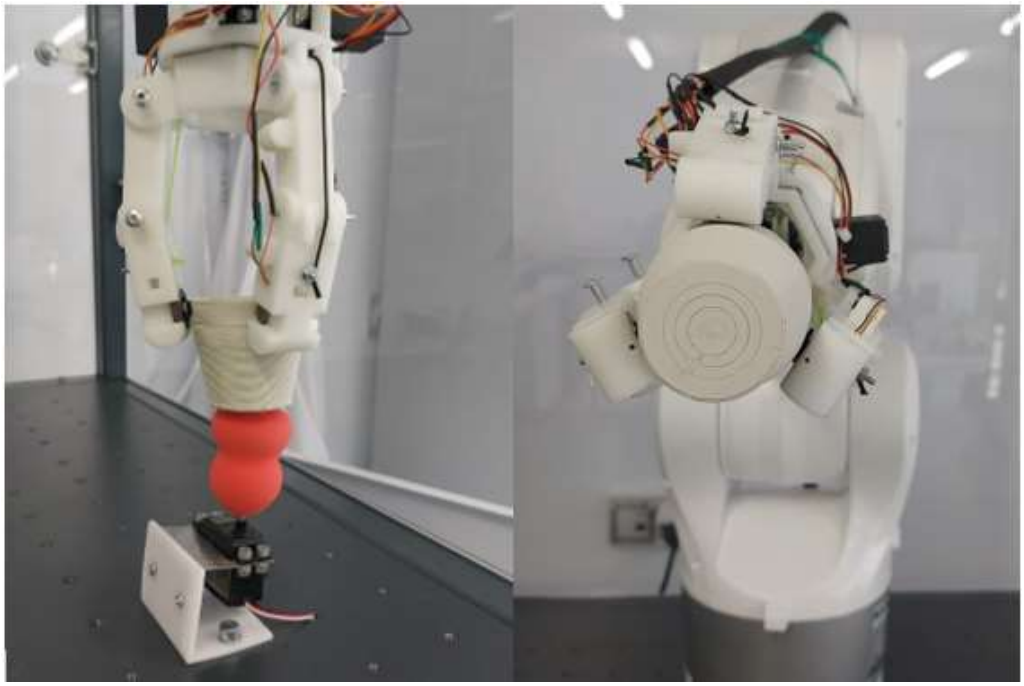


Figura 213 - Manipulação da chávena crua com garra de três dedos. Fonte: (Própria)

De seguida, para testar o limite de resistência da peça crua, o robô manteve a chávena suspensa a poucos centímetros da base da mesa do robô, mesa esta coberta por cartão para proteger o controlador e as suas ligações, observado na Figura 214.

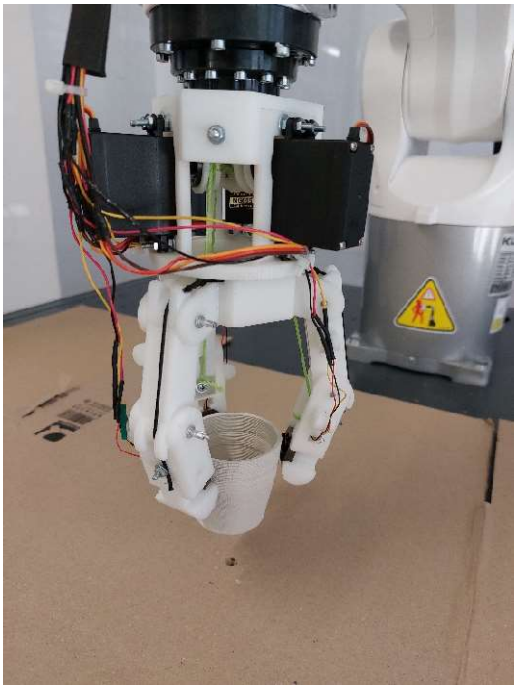


Figura 214 - Teste de rigidez da chávena crua. Fonte: (Própria)

Este teste consistiu em aplicar a força mínima de fecho idêntica à do teste anterior, presente na Figura 211, e subir lentamente até ao valor máximo do potenciómetro ou até a chávena partir, registando os valores obtidos de 40 em 40 gramas de força somados, obtendo os resultados presentes na Tabela 2.

Tabela 2 - Valores de carga do teste de rigidez. Fonte: (Própria)

Cargas nos dedos			Limite e estado da chávena	
C1	C2	C3	Limite mínimo	Chávena
53	47	47	42	OK
91	86	89	85	OK
132	131	128	125	-
182	182	172	165	-
206	214	229	205	-
262	262	262	245	-
316	318	315	285	-
362	371	359	325	-
397	372	397	365	-
425	433	424	405	-
467	482	467	445	Partiu

Quando se aumentou o valor mínimo de carga para 445, a chávena partiu, como mostra a Figura 215, tendo-se observado que os dedos “C1” e “C3” exerciam uma carga de 467g e o dedo “C2” exerceu 482g.

Figura 215 - Chávena crua partida. Fonte: (Própria)

Com a realização deste teste foi possível comprovar que o feedback sensorial é de extrema importância na manipulação de objetos frágeis, uma vez que sem esta particularidade, não seria possível manipular a chávena em verde com a garra desenvolvida.

3.4.12 - Conclusões sobre a garra de três dedos

O desenvolvimento da garra de três dedos com feedback sensorial permitiu estudar o comportamento de garras do tipo impacto na área da manipulação cerâmica, tendo sido um sucesso face à execução da tarefa pretendida. Contudo, esta garra é apenas um protótipo de testes e exposição, o que significa que não pode ser utilizada num ambiente industrial. Para tornar esta garra exequível num ambiente industrial é necessário alterar várias propriedades da mesma, expostas de seguida.

A utilização de uma placa Arduíno para alimentar e programar a garra, no lugar de autómato, deve-se ao facto de esta ser apenas, tal como mencionado anteriormente, um protótipo de testes e de exibição que não requer a capacidade e velocidade fornecidos pelo autómato. Por sua vez, num ambiente industrial é necessária maior capacidade de resposta, velocidade, segurança e capacidade de atuar durante tempos prolongados, que características do autómato, que o Arduíno não apresenta em suficiência.

A resina, utilizada para imprimir as componentes principais da garra, é um material com propriedades físicas semelhantes às do vidro, não sendo muito resistente a colisões e quedas. Por observação própria, nota-se que este material fragiliza com a sua utilização e exposição à luz do sol, mais concretamente aos raios ultravioleta, tornando o material cada vez mais quebradiço. Estas propriedades da resina não são compatíveis com as exigências da indústria, sendo necessário utilizar um material mais robusto para a garra "sobreviver" neste ambiente. Outro aspeto importante a mencionar é a exposição das suas componentes de acionamento de movimento e cabos elétricos, uma vez que no ambiente fabril, sobretudo no polimento cerâmico, existem bastantes poeiras no ar que danificam os motores e os fios de atuação expostos. A solução para este problema é proteger os motores, os fios elétricos e os fios de tração, embutindo-os no corpo da garra.

Outra mudança a ser feita para tornar a garra viável para a indústria é a substituição dos fios de tração por cabos de metal, de modo a prolongar a vida útil da garra e diminuir a sua manutenção.

Os sensores utilizados, apesar de serem robustos, compactos e de suportarem cargas elevadas, estes são pouco flexíveis e apresentam apenas um ponto de medida, o que torna a medida da carga exercida em alguns objetos inválida. Para observar este efeito, realizou-se a tarefa de manipulação de um cubo de madeira presente na Figura 216, que, apesar deste ter sido manipulado sem problemas, a leitura das cargas exercidas pelos dedos da garra não apresentaram o mesmo efeito. Como os três sensores estão afastados 120º entre si e os lados do cubo fazem 90º entre si, quando é efetuada a sua manipulação, apenas um dos sensores consegue fazer contacto permanente com o cubo, enquanto os outros ocasionalmente enviam a resposta de que estão a medir 0g de força.

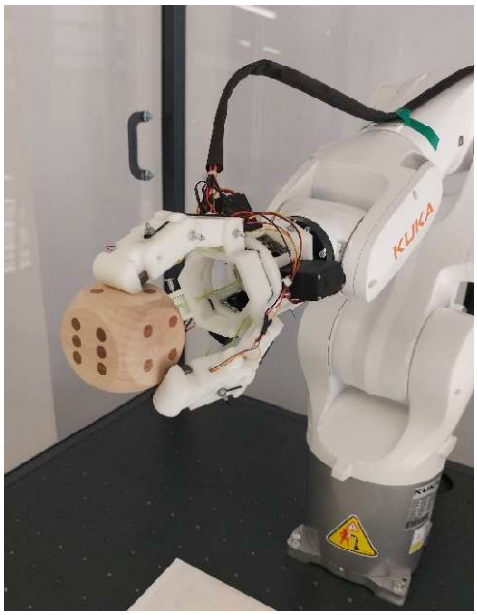


Figura 216 - Manipulação de dado de madeira. Fonte: (Própria)

CAPÍTULO 4. - CONCLUSÃO E DESENVOLVIMENTOS FUTUROS

Neste último capítulo encontram-se as conclusões retiradas da realização deste estágio e os futuros desenvolvimentos para aprimorar os projetos desenvolvidos durante o mesmo.

4.1 - Conclusão

Na área da automação industrial e da robótica, existe uma constante mudança face ao seu mundo tecnológico, evoluindo a passo cada vez mais sistemático. Desde os primórdios da industrialização que o seu objetivo é aumentar a produtividade, onde a automação industrial e a robótica são indispensáveis para a produção da atualidade.

Sendo que o foco deste relatório de estágio é sobre a manipulação robótica na indústria, sinto que os meus conhecimentos de eletrotecnia adquiridos durante todo o meu percurso do ensino superior, em conjunto com os conhecimentos eletromecânicos obtidos na licenciatura, foram empregues com sucesso na realização e desenvolvimento das tarefas propostas.

Um dos grandes desafios na área da robótica é a manipulação de objetos com formatos específicos, sendo por vezes necessário grandes investimentos em garras universais, capazes de manipular estes objetos.

Apesar da indústria 4.0 representar o presente geral, a indústria 5.0 já começa a ser implementada em algumas áreas e com ela, a importância do fabrico aditivo e dos simuladores robóticos, como por exemplo o RobotStudio da ABB, tende a aumentar, de modo a permitir o acompanhamento da velocidade de trabalhos de engenharia, tais como reformas ou instalações de equipamentos industriais, com a velocidade que a implementação da indústria 5.0 requer.

4.2 - Desenvolvimentos futuros

Apesar de todos os objetivos terem sido concluídos com sucesso, existem ainda alguns parâmetros que não foram explorados na sua totalidade. A criação da réplica digital de encaixotamento foi efetuada com sucesso, porém o próximo passo seria a sua implementação em fábrica com a adição de pelo menos três robôs manipuladores.

Quanto às garras desenvolvidas, estas funcionam perfeitamente como protótipo, porém não apresentam durabilidade nem proteções suficientes para serem implementadas num sistema com funcionamento sistemático.

Poderiam ainda ser desenvolvidas na garra de três dedos algumas funcionalidades tais como a deteção de escorregamento do objeto a ser manipulado durante a sua recolha. De modo a permitir intercomunicação entre a garra de três dedos e o robô, era necessário ligar uma das entradas do controlador do robô a um relé alimentado a 24VDC, excitado pelos 5VDC de uma saída do Arduíno, deste modo, quando o robô executasse o movimento de agarre da chávena e a levantasse, era criada uma função na placa Arduíno que, recorrendo à análise em tempo real das cargas medidas pelos três sensores, detetasse se o objeto foi agarrado com sucesso, ou se o robô deve tentar agarrá-lo novamente. O robô, numa posição se segurança, aguardava pela resposta do Arduíno, onde, caso a resposta dada fosse de que o objeto foi agarrado com sucesso, o robô continuava a realização da sua tarefa, caso contrário, voltava a tentar agarrar a chávena.

REFERÊNCIAS BIBLIOGRÁFICAS

(13 de Março de 2013). Obtido de UNESP: https://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/aula3-motor-de-passo-2013-1-13-03-2013-final.pdf

ABB. (s.d.). RobotStudio.

Aksoy, G., Polat, H., Polat, M., & Coskun, G. (2006). Effect of various treatment and glazing (coating) techniques on the roughness and wettability of ceramic dental restorative surfaces. *Colloids and Surfaces B: Biointerfaces, 53*(2), 254-259.

Ali, H., Hoi, L. H., & Seng, T. C. (2011). Design and Development of Smart Gripper with Vision Sensor for Industrial Applications.

Automate. (s.d.). Obtido em 5 de 1 de 2022, de https://www.automate.org/a3-content/joseph-engelberger-unimate

Automation, R. &. (s.d.). Product specification IRB 1100/CRB 1100. *Skribenta version 5.4.005*.

Benotsmane, R., Dudás, L., & Kovács, G. (2021). Survey on New Trends of Robotic Tools in the Automotive Industry. *Vehicle and Automotive Engineering 3*.

Boettcher, M. (10 de Maio de 2018). *Revolução Industrial - Um pouco de história da Indústria 1.0 até à Indústria 4.0.* Obtido de Linkedin: https://pt.linkedin.com/pulse/revolu%C3%A7%C3%A3o-industrial-um-pouco-de-hist%C3%B3ria-da-10-at%C3%A9-boettcher

Castanho, A. M. (2008). Direitos Humanos na Primeira Revolução Industrial. *IV ENCONTRO DE INICIAÇÃO CIENTÍFICA E III ENCONTRO DE EXTENSÃO UNIVERSITÁRIA*.

CobotTrends.com. (s.d.). Obtido em 5 de 1 de 2022, de https://www.cobottrends.com/wp-content/uploads/2019/10/SRAP_1.jpg

Coutinho, L. (2016). A terceira Revolução Industrial e Tecnológica. As Grandes Tendências das Mudanças. Em *Economia e Sociedade* (pp. 69-87). UNICAMP.

Crisanto, A. M. (2016). *Uma tendência mundial em forma de tecnologia.* Jundiaí, Brasil: Instituto de Ciências Exatas e Tecnologia Universidade Paulista - UNIP.

CTCV em números. (3 de Agosto de 2022). Obtido de https://www.ctcv.pt/emnumeros.html

European Commission-CTCV. (27 de Maio de 2022). Obtido de European Commission: https://ati.ec.europa.eu/technology-centre/ctcv-centro-tecnologico-da-ceramica-e-do-vidro

Franke, L. S. (s.d.). *More Than Digital*. Obtido em 5 de 1 de 2022, de https://morethandigital.info/pt-pt/uma-breve-historia-da-robotica-robos-modernos/

Gabriel, F., Fahning, M., & Meiners, J. (2020). Modeling of vacuum grippers for the design of energy efficient vacuum-based handling processes. Em *Prod. Eng. Res. Devel. 14* (pp. 545–554).

Hogreve, S., Prigge, M., Köbisch, K. O., & Tracht, K. (2020). Encapsulation of sensory gripper fingers with silicone rubber. Em *Procedia CIRP* (pp. 439-444).

Hu, Z., Wan, W., & Harada, K. (2019). Designing a Mechanical Tool for Robots With Two-Finger Parallel Grippers.

Kumar, R., Yadav, H., Gupta, V., & Khatait, J. P. (2021). *Flexure based gripper to grasp hollow objects by internal surface interaction.*

L.G.Li, Z.Y.Zhuob, A.K.H.Kwan, T.S.Zhang, & D.G.Lu. (2020). Cementing efficiency factors of ceramic polishing residue in compressive strength and chloride resistance of mortar. *Powder Technology, 367*(1), 163-171.

Lee, K., Wang, Y., & Zheng, C. (2 de Abril de 2020). TWISTER Hand: Underactuated Robotic Gripper.

Lima, E., & Neto, C. (2017). Revolução Industrial: Considerações sobre o pioneirismo Industrial inglês. *Revista Espaço Académico, nº194*.

Liu, C., Cheng, J., Li, Z., Cheng, C., Zhang, C., Zhang, Y., & Zhong, R. Y. (2020). Design of a self-adaptive gripper with rigid fingers for Industrial Internet. Em *Robotics and Computer-Integrated Manufacturing.*

Mayor, A. (2018). *Gods and Robots: Myths, Machines, and Ancient Dreams of Technology.* Princeton University Press.

Miettinen, J., Frilund, P., Vuorinen, I., Kuosmanen, P., & Kiviluoma, a. P. (2019). *Granular jamming based robotic gripper for heavy objects.* Finland.

Modabberifar, M., & Spenko, M. (2018). A shape memory alloy-actuated gecko-inspired robotic gripper. Em *Sensors and Actuators A: Physical* (pp. 76-82).

Monkman, G. (2006). Robot Grippers. *Assembly Automation, 29*(1).

Mouser. (s.d.). Obtido de FX29 - Compact Compression Load Cell : https://pt.mouser.com/datasheet/2/418/5/ENG_DS_FX29_A5-1609196.pdf

Nascimento, S. (2006). Automatizações no inorgânico: aproximações ao estudo social de criaturas artificiais. Em *Análise Social* (p. 1042).

Operating Manual OMNICORE. (s.d.). *Skribenta version 5.4.005*.

Ottoni, A. L. (2010). Introdução à Robótica. Minas Gerais: Universidade Federal de São João Del-Rei.

Özdemir, V., & Hekim, N. (2018). Birth of Industry 5.0: Making Sense of Big Data with Artificial Intelligence, “The Internet of Things” and Next-Generation Technology Policy. *OMICS: A Journal of Integrative Biology*, 65–76.

Pasquini, N. (2014). As Revoluções Industriais: Uma Abordagem Conceitual. *Revista Tecnológica da Fatec Americana, 8(1)*, 29-44.

Pedro Chaves, Director de Negócio de Manufacturing do SAS Portugal. (18 de Junho de 2021). Indústria 5.0, o novo paradigma de transformação digital do sector de manufacturing. *Público*.

Peidró, A., Tavakoli, M., Marín, J. M., & Reinoso, Ó. (2019). Design of compact switchable magnetic grippers for the HyReCRo structure-climbing robot. Em *Mechatronics* (pp. 199-212).

Pereira, A., & Simonetto, E. (2018). Indústria 4.0: Conceitos e Perspectivas para o Brasil. *Revista VALE, 16(1)*, 1-9.

Petruzella, F. D. (2011). *Programmable Logic Controllers (4ª ed.).* 1221 Avenue of the Americas, New York, NY, 10020 ; Estados Unidos da América: The McGraw-Hill Companies, Inc.

Qian, Z., QingLong, M., YongQian, X., & Lin, G. (2020). The Robot Intelligent Spraying Glazing System for Sanitary Ceramics Industry. *IOP Publishing Ltd*.

Quem Somos-CTCV. (27 de Outubro de 2021). Obtido de CTCV: https://www.ctcv.pt/quemsomos.html

Reddy, P. V., & Satya Suresh, V. N. (2013). A review on importance of universal gripper in industrial robot applications. *International Journal of Mechanical Engineering and Robotics Research, 2*(2).

Ribeiro, E. G. (Julho de 2020). Controle Servo-Visual de Robô Manipulador com 7 Graus de Liberdade.

Ribeiro, J., Lima, R., Eckhardt, T., & Paiva, S. (2021). Robotic Process Automation and Artificial Intelligence in Industry 4.0 – A Literature review. Em *Procedia Computer Science, Volume 181* (pp. 51-58).

Sakurai, R., & Zuchi, J. (2018). Revoluções industriais até à indústria 4.0. *Revista Interface Tecnológica, 15(2)*, 480-491.

Samadikhoshkho, Z., Zareinia, K., & Janabi-Sharifi, F. (2019). A Brief Review on Robotic Grippers Classifications. *IEEE Canadian Conference of Electrical and Computer Engineering (CCECE).*

Santos, V. M. (2003-2004). Robótica Industrial. Aveiro: Universidade de Aveiro.

Shintake, J., Cacucciolo, V., Floreano, D., & Shea, H. (2018). Soft Robotic Grippers. *Advanced Materials, 30*(29).

Silva, J. G., Ferreiro, E. K., Cirquera, L. R., Santos, T. C., & Silva, A. d. (2014). EXTENSÔMETRO CAPACITIVO PARA A DETERMINAÇÃO DE DEFORMAÇÕES EM SISTEMAS ESTRUTURAIS. 951-959.

Sousa, E. M. (julho de 2016). Projeto de um Gripper Sub-atuado com Aplicações em Locomoção Robótica.

Sriskandarajah, C., & Shetty, B. (2018). A review of recent theoretical development in scheduling dual-gripper robotic cells. Em *International Journal of Production Research, 56* (pp. 1-2, 817-847).

Tai, K., El-Sayed, A.-R., Shahriari, M., Biglarbegian, M., & Mahmud, S. (2016). State of the Art Robotic Grippers and Applications. *Robotics, 5*(2).

Torres, R., & Ferreira, N. (2022). Development of grippers in the ceramic industry. *MAASE_ISEP*, (p. 5). Porto.

Tubbs, S. P. (2018). Programmable Logic Controller (PLC) Tutorial, Siemens Simatic S7-1200. Publicis MCD Werbeagentur GmbH; 3rd ed.

Tuli, T. B., & Manns, M. (2019). Hierarchical motion control for real time simulation of industrial robots. *52 th CIRP Conference on Manufacturing Systems.* Piegen, Germany.

Venâncio, A. L., & Brezinski, G. L. (2017). *Sistema de avaliação de maturidade industrial baseando-se nos conceitos de Indústria 4.0.* Curitiba: Universidade Tecnológica Federal do Paraná.

Wayand, B. (20 de Março de 2020). *What Is a PLC.* Obtido de MRO Electric: https://www.mroelectric.com/blog/what-is-a-plc/

WEBER, A., FUSSLER, C., J. F. O'HANLON, R. G., & GRANDJEAN, E. (1980). Psychophysiological effects of repetitive tasks. Em *Ergonomics, 23:11* (pp. 1033-1046).

Xu, M., David, J. M., & Kim, S. H. (2018). The Fourth Industrial Revolution: Opportunities and Challenges. *International Journal of Financial Research*, 90-95.

Zhang, B., Xie, Y., Zhou, J., Wang, K., & Zhang, Z. (2020). State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review. Em *Computers and Electronics in Agriculture, Volume 177.*

CAPÍTULO 5. - ANEXOS

Índice de anexos

Anexo 1 Robô manipulador ABB

O robô manipulador utilizado é um ABB IRB 1100-4/0.58, tal como mostra a Figura 217, existente no laboratório de robótica do departamento de eletromecânica do ISEC. Na Tabela 3 encontram-se as suas especificações.

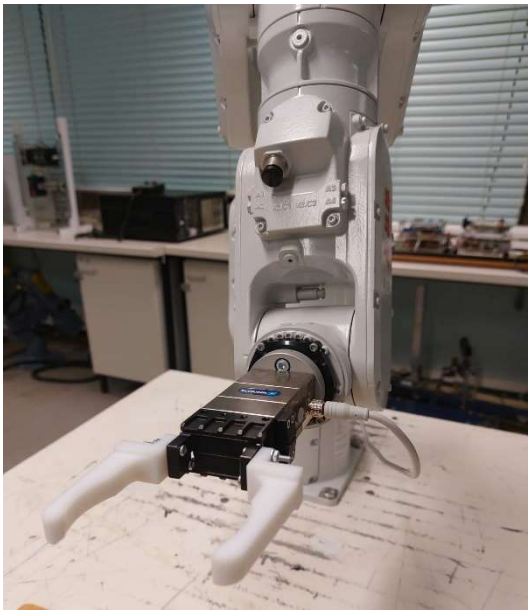


Figura 217 - Robô ABB IRB 1100-4/0.58. Fonte: (Própria)

Tabela 3 - Especificações do robô. Fonte: Chapa do robô

Tipo	Manipulador
Modelo	IRB 1100-4/0.58
Carga máxima	4 kg
Alcance	0.58 m
Peso do robô	21 kg
Graus de liberdade	6

Informações adicionais:

- Todos os seis eixos apresentam restrições de software, porém apenas o eixo 1 (base) apresenta em acréscimo uma restrição mecânica.
- É alimentado a 30V e a 1,5A ou 1A, dependendo da entrada escolhida para alimentação.
- Existem quatro orifícios de conexão de ar no robô com capacidade máxima de 6 bar.

O robô apresenta as seguintes dimensões presentes na Figura 218.

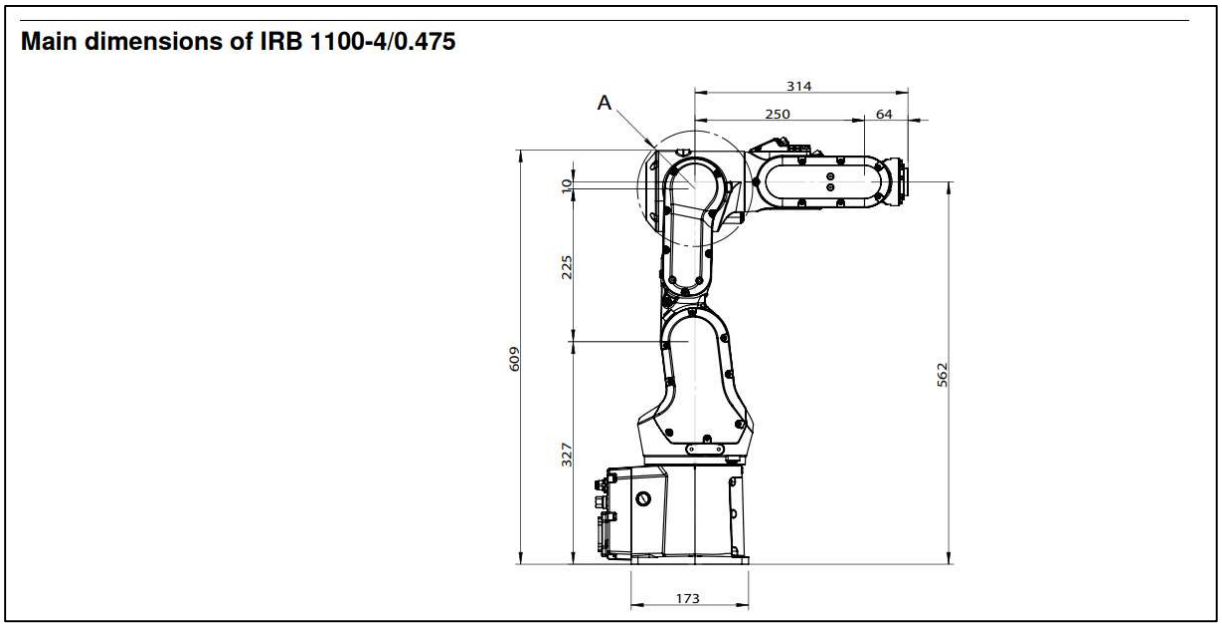


Figura 218 - Dimensões principais do IRB 1100-4/0.58. Fonte: (Automation)

Estes robôs manipuladores apresentam uma carga máxima (neste caso de 4,2 kg), porém esta carga é apenas suportada pelo robô em determinadas circunstâncias. Quando, por exemplo, o braço robótico se encontra nas extremidades do seu alcance, esta carga máxima diminui consideravelmente, tal como mostra a Figura 219. Esta carga também varia dependendo se o robô está instalado numa parede ou teto e do ângulo de rotação do pulso do robô.

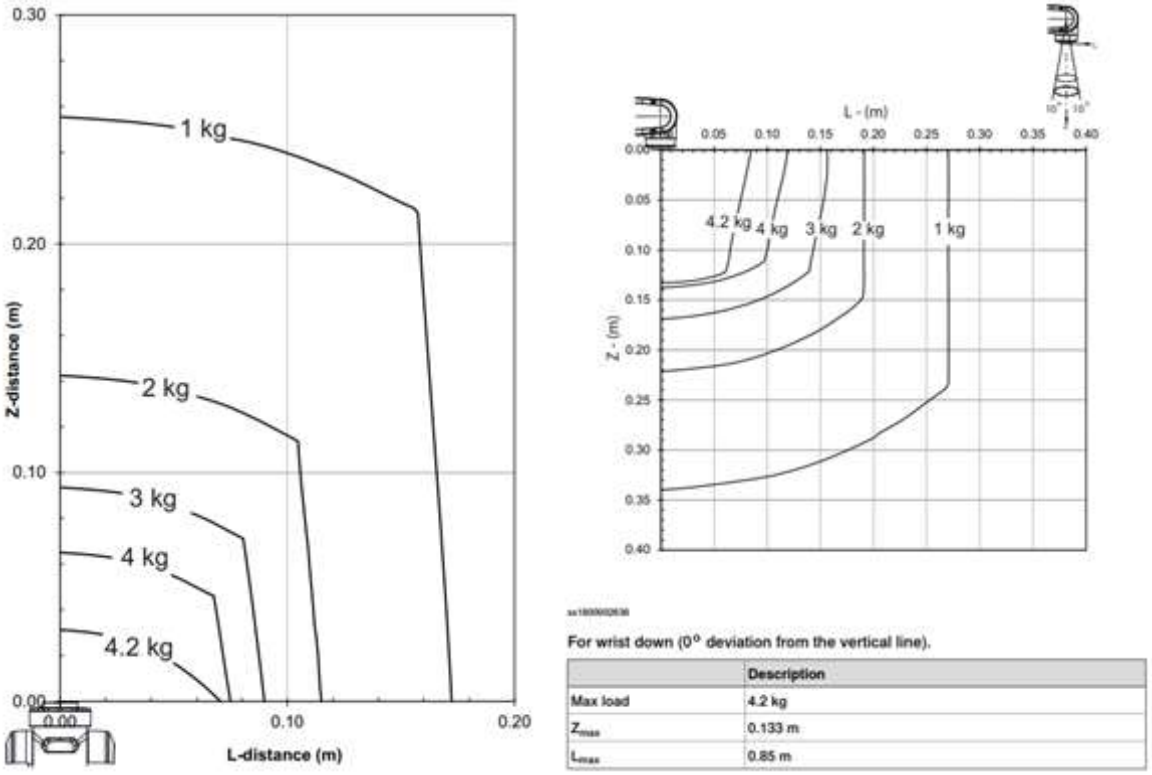


Figura 219 - Diagramas de carga do IRB 1100-4/0.58 (esquerda com pulso vertical ±10º). Fonte: (Automation)

Cada eixo do robô apresenta restrições de trabalho distintas representadas pelo máximo e mínimo de graus segundo o seu eixo de rotação. A Tabela 4 mostra estes valores para os seis eixos do robô.

Tabela 4 - Alcance de trabalho. Fonte: (Automation)

Eixos	Alcance de trabalho	Nota
Eixo 1	±230°	O robô montado na parede tem uma área de trabalho para o eixo 1 que depende da carga útil e das posições de outros eixos. É recomendada simulação no RobotStudio.
Eixo 2	-115°/+113°	-
Eixo 3	-205°/+55°	-
Eixo 4	±230°	-
Eixo 5	-125°/+120°	-
Eixo 6	±400°	Valor Padrão
	±242°	Valor máximo de revolução. A faixa de trabalho padrão para o eixo 6 pode ser estendida alterando os valores dos parâmetros no software.

Anexo 2 Consola Flexpendant (OMNICORE)

[foto da consola]

A consola disponível no ISEC para a manipulação do robô é a FlexPendant mais recente (2019), com touch screen. Esta permite operar o robô correndo programas, modificá-los, controlar o robô em tempo real através do joystick, entre outros. A sua nomenclatura encontra-se na Figura 220 e na Tabela 5.

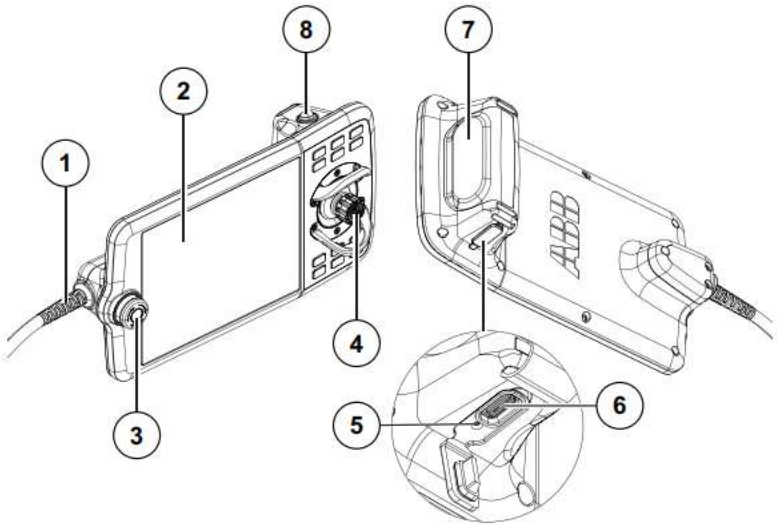


Figura 220 - Nomenclatura da consola FlexPendant. Fonte: (Operating Manual OMNICORE)

Tabela 5 - Nomenclatura da consola FlexPendant. Fonte: (Própria)

Nomenclatura	Descrição
1	Conetor
2	Ecrã tátil
3	Botão de emergência
4	Joystick
5	Botão de Reset
6	Porta USB
7	Botão “homem morto”
8	Botão de polegar (robôs colaborativos)

No lado direito do ecrã tátil, existem os seguintes botões apresentados na Figura 221 e expostos na Tabela 6:

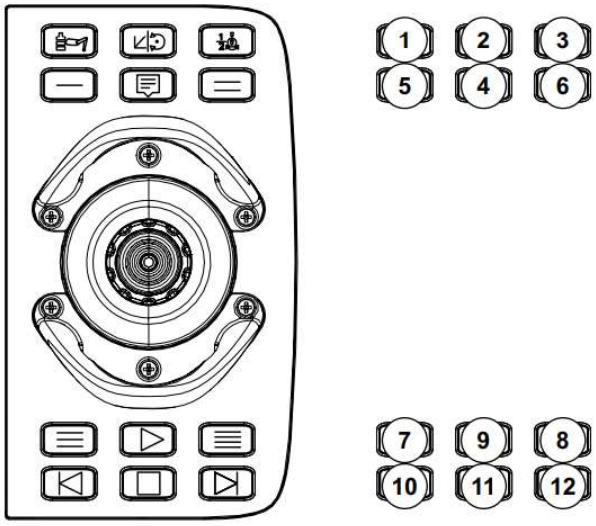


Figura 221 - Botões "Hard Buttons". Fonte: (Operating Manual OMNICORE)7

Tabela 6 - Nomenclatura de botões "Hard Buttons". Fonte: (Operating Manual OMNICORE)

Nomenclatura	Descrição
1	Botão da unidade mecânica. Permite a seleção de uma unidade mecânica.
2	Botão de movimento 1. Permite trocar entre movimento linear ou de juntas.
3	Botão de movimento 2. Permite trocar o movimento entre o conjunto de eixos 1-3 e 4-6
4	Botão de mensagem. Se for impulsionado durante alguns segundos, captura uma *screenshot*.
5,6,7,8	Botões programáveis pelo utilizador.
9	Butão de **START**. Executa o programa selecionado.
10	Botão de **Passo ATRÁS**. Executa uma instrução ao contrário.
11	Butão de **STOP**. Para a execução do programa.
12	Botão de **Passo á FRENTE**. Executa uma instrução.

Anexo 3 Programação RAPID

A linguagem de programação dos robôs ABB é denominada por RAPID, introduzida pela primeira vez em 1994 juntamente com o sistema de controlo S4. Este anexo resume as funcionalidades principais desta linguagem, servindo de um pequeno manual para consulta das funções desta linguagem.

Procedimento

Esta programação funciona através da criação de procedimentos através do código "*PROC*" que terminam com o código "*ENDPROC*". Um procedimento obrigatório para qualquer programa em RAPID é o procedimento "*main*". Este é o único procedimento que inicia automaticamente sem necessidade de o chamar. Pode também ser o único procedimento do programa, ou pode chamar mais procedimentos através do seu nome.

```
66
67      PROC main()
68          !Add your code here
69          MoveJ home,v7000,z0,Vacuo_Tool_tcp\WObj:=wobj0;
70          Pick_frente;
71          Pick_esq;
72          Pick_centro;
73          Pick_right;
74          Pick_tras;
75      ENDPROC
76
77      PROC Pick_centro()
78          MoveJ pick_safe,v4000,z0,Vacuo_Tool_tcp\WObj:=wob
79          SetDO Q_ventosa,0;
80          MoveL Target_10,v4000,fine,Vacuo_Tool_tcp\WObj:=w
```

Figura 222 - Exemplo de procedimento "main" e início de procedimento "Pick_centro". Fonte: (Própria)

A Figura 222 mostra um programa que, através de um movimento de juntas, o robô vai á posição "*HOME*" e após completar este movimento, executa as funções "*Pick_frente*"," *Pick_esq*"," *Pick_centro*"," *Pick_right*" e" *Pick_tras*" por esta ordem.

A ordem da criação das funções não influencia na ordem de execução das mesmas, tal como é possível verificar na Figura 222 que a próxima função a ser criada é a função centro, porém esta só é executada após a finalização de duas outras funções chamadas pela main.

Comandos de movimento

Existem vários comandos para movimentar o manipulador da posição atual para a posição desejada, cada um com propriedades diferentes. É com especial atenção nestas propriedades que se deve escolher o tipo de movimento a programar no manipulador.

Na Tabela 7 encontram-se os tipos de movimento de um manipulador ABB.

MoveAbsJ	Executa um movimento para uma posição de juntas absoluta
MoveC	Executa um movimento circular
MoveCDO	Executa um movimento circular e aciona uma saída digital no canto
MoveCSync	Executa um movimento circular e um procedimento RAPID
MoveExtJ	Move uma ou várias unidades mecânicas sem o TCP (tool center point)
MoveJ	Executa um movimento de juntas
MoveJDO	Executa um movimento de juntas e aciona uma saída digital no canto
MoveJSync	Executa um movimento de juntas e um procedimento RAPID
MoveL	Executa um movimento linear
MoveLDO	Executa um movimento linear e aciona uma saída digital no canto
MoveLSync	Executa um movimento linear e um procedimento RAPID

Tabela 7 - Tipos de movimento programáveis para o robô. Fonte: (Própria)

Dentro destes tipos de movimentos, existem três tipos de movimentos essenciais para a execução de qualquer tarefa pelo manipulador.

MoveJ – O robô executa um movimento de juntas, ou seja, os eixos do robô movem-se para a posição de destino segundo um caminho não linear, de modo que todos os eixos cheguem ao valor da posição de destino ao mesmo tempo.

Um exemplo de uma linha de comando padrão com MoveJ é o seguinte:

MoveJ p1, veloc_1, z0, gripper_1;

Nesta linha de comando, o TCP da ferramenta "*gripper_1*" desloca-se segundo um caminho não linear para a posição "*p1*" com a velocidade "*veloc_1*" para a cota "*z0*".

Nota-se que a cota é definida para indicar a proximidade dos eixos reais á posição programada. Para cota "*z0*" a ferramenta irá exatamente para a posição programada, enquanto para a cota "*z40*" a ferramenta estará desviada 40mm segundo o eixo Z da

posição alvo. Colocar neste parâmetro o código "*fine*" fará com que o programa apenas execute o próximo comando quando a trajetória chegar a sua posição final, evitando pequenos arredondamentos entre posições, de acordo com a Figura 223.

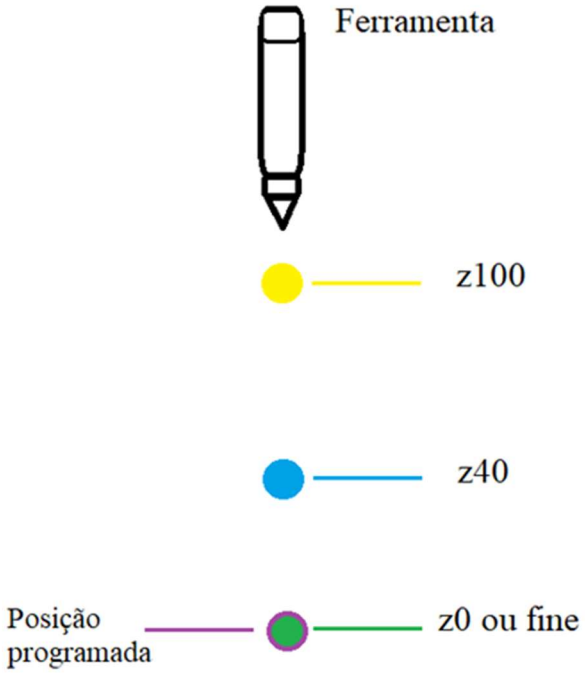


Figura 223 - Cota segundo o eixo Z da ferramenta. Fonte: (Própria)

MoveL – O robô executa um movimento linear, ou seja, desloca o TCP pelo trajeto mais curto entre a posição inicial e a posição de destino. Este movimento pode também ser utilizado para reorientar a ferramenta.

Um exemplo de uma linha de comando padrão com MoveL é o seguinte:

MoveL p1, veloc_1, fine, gripper_1;

Nesta linha de comando, o TCP da ferramenta "*gripper_1*" desloca-se segundo um caminho linear para a posição "*p1*" com a velocidade "*veloc_1*" para a cota "*fine*".

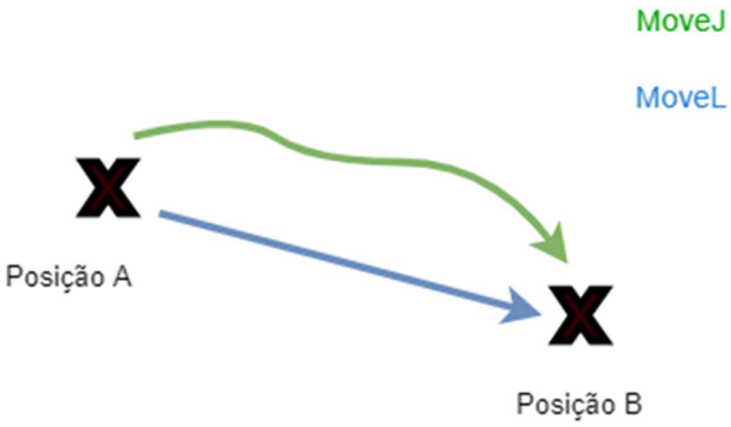


Figura 224 - Movimento linear (azul) e movimento de juntas (verde). Fonte: (Própria)

Como ilustrado na Figura 224, o movimento de juntas executa o trajeto entre a posição A e B de modo não linear para facilitar o movimento das juntas do robô, tornando o movimento mais fluido e rápido.

Já o movimento linear executa o trajeto mais curto entre A e B de modo retilíneo obrigando cada junta do robô a rodar a diferentes velocidades e direções durante todo o percurso. Assim sendo, para caminhos de precisão (e.g. posicionar a garra junto ao objeto a agarrar) é recomendado utilizar sempre movimentos lineares, enquanto para movimentos de maior liberdade e de deslocamento mais longo deve-se usar o movimento de juntas.

MoveC – O robô executa um movimento circular, ou seja, desloca o TCP para uma posição destino de modo circular. Normalmente a orientação da garra permanece a mesma durante este movimento.

Um exemplo de uma linha de comando padrão com MoveC é o seguinte:

MoveC p1, p2, v500, z30, gripper_1;

Nesta linha de comando, o TCP da ferramenta "*gripper_1*" desloca-se segundo um caminho circular da posição inicial seguindo a orientação do ponto circular "*p1*" para a posição "*p2*" com a velocidade "*v500*" para a cota "*z30*". Deste modo, o movimento circular é definido através de três pontos, o ponto de início, o ponto circular "*p1*" e o ponto de destino "*p2*".

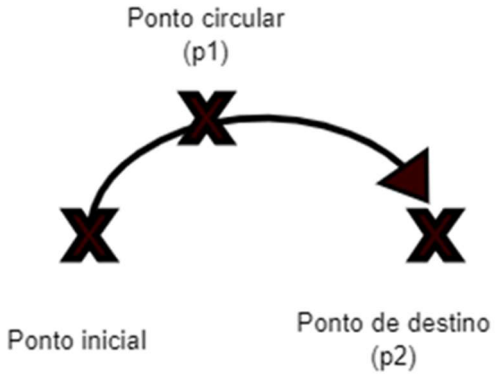


Figura 225 - Movimento circular. Fonte: (Própria)

Como mostra a Figura 225, o movimento circular executa o caminho entre o ponto inicial e "p2" formando um semicírculo segundo a orientação "p1", não passando obrigatoriamente neste ponto.

Este comando de movimento pode também executar círculos completos, porém é recomendado executar duas ou mais linhas de movimento circular, uma vez que esta obriga a que exista uma distância mínima de 0,1 mm entre o ponto inicial e o ponto de destino.

A Figura 226 representa os movimentos circulares dos dois códigos abaixo, onde o código (a) executa o círculo com apenas uma linha de comando e o código (b) executa

com duas linhas de comando, ambos provenientes de um movimento linear que termina no ponto “p1”

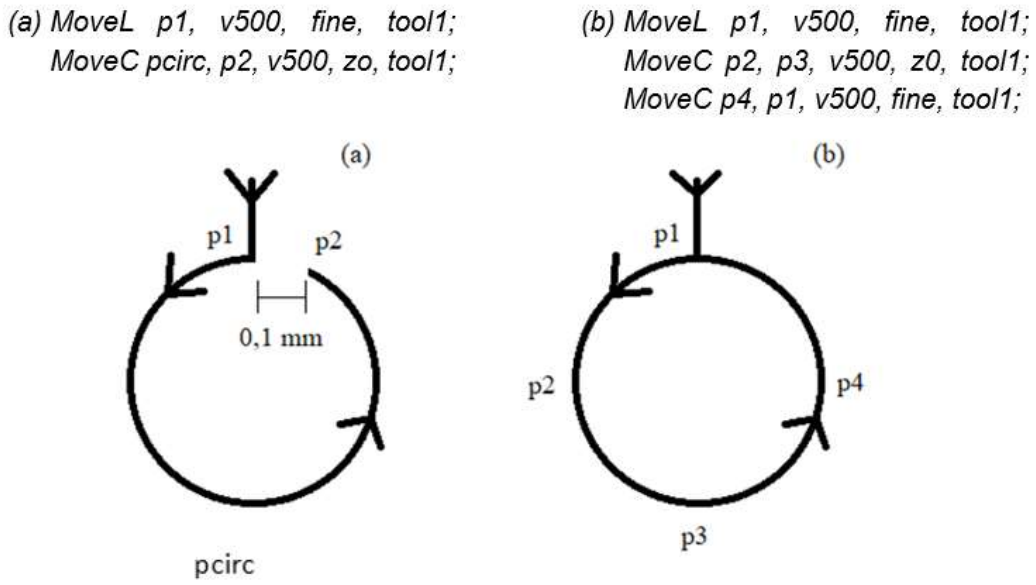


Figura 226 - Círculo completo executado com uma linha de comando (a) e com duas linhas de comando (b). Fonte: (Própria)

Nota-se que existem algumas restrições quanto á colocação do ponto circular e do ponto de destino. Uma delas, já mencionada, é a distância mínima entre o ponto inicial e o de destino que é de 0,1 mm. Outras duas restrições são a distância mínima entre o ponto inicial e o ponto circular também de 0,1 mm e o ângulo formado entre o ponto circular e o ponto de destino com o ponto de início tem de ser superior a 1o. A precisão do movimento circular é menor quando os pontos são colocados perto destes limites.

É possível observar estas restrições na Figura 227.

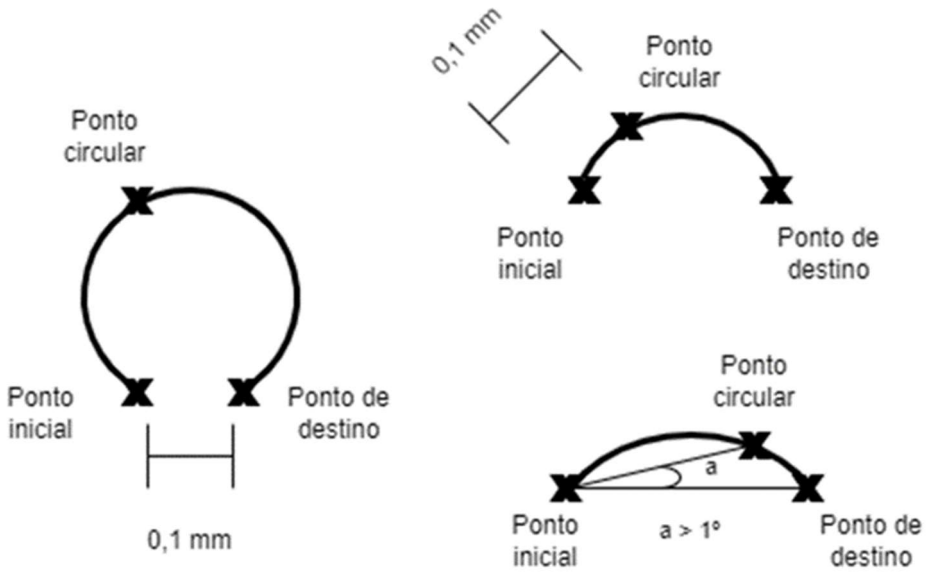


Figura 227 - Restrições mínimas de distância entre os pontos do comando MoveC. Fonte: (Própria)

Sinais de entrada e de saída

Quando um programa necessita de acionar um mecanismo externo, ou quando o manipulador apenas se pode mover quando um evento é concretizado, o controlador é o componente que trata da informação trocada entre mecanismos. Isto é possível através da troca de sinais elétricos com nomenclatura única por cada evento ou estado presente no programa. Estes sinais podem ser de entrada ou de saída (I/O).

Sinais de entrada – São sinais digitais ou analógicos enviados por uma fonte externa para o controlador do manipulador.

Sinais de saída – São sinais digitais ou analógicos enviados pelo controlador do manipulador para um recetor externo.

Estes sinais podem ser atribuídos ou alterados através da utilização dos seguintes códigos.

AliasIO – Utilizado para definir um sinal de qualquer tipo com uma alcunha ou para utilizar sinais em módulos de tarefa integrados. Um exemplo deste comando encontra-se no trecho a seguir:

```
VAR signaldo alias_do;
PROC prog_start()
    AliasIO config_do, alias_do;
ENDPROC
```

A rotina prog_start está conectada ao evento START nos parâmetros do sistema, e o programa que define o sinal de saída digital alias_do está conectado ao sinal de saída digital config_do no início do programa.

PulseDO – Utilizado para criar um pulso com valor padronizado de 200ms numa saída digital do controlador. Um exemplo da utilização deste código encontra-se na linha seguinte:

```
PulseDO acionamento_garra;
```

Set – Utilizado para alterar o valor de uma saída digital para “1”. Existe um pequeno atraso antes do sinal obter fisicamente seu novo valor. Se for pretendido que o programa aguarde até que o sinal tenha recebido o seu novo valor, então pode ser utilizada a instrução SetDO com o parâmetro opcional \Sync. O valor verdadeiro depende da configuração do sinal. Se o sinal for invertido no parâmetros do sistema, esta instrução faz com que o canal físico seja definido como zero. Um exemplo desta instrução encontra-se na linha seguinte.

```
Set acionamento_garra;
```

SetDO – Tal como o comando anterior, este é utilizado para alterar o valor de um sinal digital de saída do controlador, com ou sem atraso ou sincronização. Este comando pode também alterar o valor para zero e definir um atraso no envio do sinal, ao obrigar o robô a esperar que a alteração do sinal seja fisicamente efetuada, antes de realizar a próxima linha de comando. Um exemplo destas aplicações encontra-se nas linhas seguintes.

SetDO \SDelay :=0.2, acionamento_garra, 0;

O sinal *acionamento_garra* é alterado para "0" após um atraso de 0.2 segundos, contudo, o programa continua a executar as próximas linhas de comando, sem esperar pela alteração efetuada nesta linha de comando.

SetDO \Sync ,acionamento_garra, 1;

Altera o valor do sinal *acionamento_garra* para "1" e o programa aguarda que esta alteração seja feita fisicamente, antes de prosseguir para a próxima instrução.

SetAO – Utilizado para alterar o valor de um sinal de saída analógico, funcionando de modo semelhante à instrução anterior, tal como se pode observar na linha seguinte:

SetAO acionamento_garra, 15;

Reset – Utilizado para alterar o valor de um sinal de saída digital para zero, um exemplo deste comando encontra-se na linha seguinte:

Reset acionamento_garra;

Durante a execução de um programa, o controlador pode aguardar a execução de determinadas ações refletidas na alteração de um sinal digital ou analógico. Para tal, existem os seguintes comandos apresentados na Tabela 8:

Tabela 8 - Comandos de espera fundamentais RAPID. Fonte: (Própria)

WaitAI	Aguarda até que um sinal de entrada analógico cumpra a condição (e.g. "*WaitAI ai1, 5*")
WaitA0	Aguarda até que um sinal de saída analógico cumpra a condição (e.g. "*WaitAO ao1, 5*")
WaitDI	Aguarda até que um sinal de entrada digital cumpra a condição (e.g. "*WaitDI di2, 1*")
WaitDO	Aguarda até que um sinal de saída digital cumpra a condição (e.g. "*WaitDO do2, 1*")
WaitGI	Aguarda até que um conjunto de sinais de entrada digital cumpra a condição (e.g. "*WaitGI gi4, 5*")
WaitGO	Aguarda até que um conjunto de sinais de saída digital cumpra a condição (e.g. "*WaitGO do2, 3*")

WaitRob	Aguarda até ao ponto de paragem ou de velocidade zero (e.g. "*WaitRob \InPos*")
WaitTestAndSet	Aguarda até que a variável passe de "*unset*" para colocar em "*set*" (e.g. "*WaitTestAndSet tproutine_inuse*")
WaitTime	Aguarda um determinado intervalo de tempo (e.g. "*WaitTime 1.5*")
WaitUntil	Aguarda até que alguma condição seja cumprida (e.g. "*WaitUntil di2 = 1*")
WaitWObj	Aguarda a chegada de um objeto num transportador (e.g. "*WaitWObj wobj_em_tapete1*")

Recorrendo a estes comandos, é possível comandar o robô a aguardar que determinadas condições sejam cumpridas, podendo tanto servir de segurança, como para sincronizar procedimentos.

Tal como é possível ordenar ao robô para aguardar uma condição, é também possível ordenar que ele execute um conjunto de movimentos em função de certas condições. Para tal, existem os seguintes comandos:

WHILE – Repete um conjunto de comandos enquanto uma condição for verdadeira, tal como se observa no parágrafo seguinte:

```
WHILE condição_1 < condição_2 DO
    …
    condição_1 := condição_1 + 1;
ENDWHILE
```

Neste exemplo, a consição_1 vai somar 1 ao seu valor sempre que a função "WHILE" executa um ciclo, quando o valor desta condição superar o valor da consição_2, o ciclo quebra.

FOR – Repete uma ou um conjunto de instruções um determinado número de vezes, tal como é possível observar no parágrafo seguinte onde repete a Rotina_1 dez vezes:

```
FOR i FROM 1 TO 10 DO
    …
    Rotina_1;
ENDFOR
```

IF – Executa uma ou várias instruções caso a condição seja cumprida. Pode ser suplementado com "ELSE" ou "ELSEIF" de modo a executar várias tarefas diferentes, dependendo das condições que forem, ou não, cumpridas, tal como é possível observar no seguinte parágrafo:

```
IF reg_1 > 5 THEN
        …
ELSEIF reg_1: =5 THEN
        …
ELSE
        …
ENDIF
```

No código apresentado acima é possível verificar três ocorrências diferentes, dependendo do valor de "*reg_1*", a ocorrência para o valor de "*reg_1*" superior a 5, a ocorrência para ser igual a 5, e a ocorrência no caso de este não corresponder a estas duas condições, ou seja, caso seja inferior a 5.

Anexo 4 Código Arduíno de medição de carga

```
#include <Wire.h> //INCLUSÃO DE BIBLIOTECA
#include <LiquidCrystal_I2C.h> //INCLUSÃO DE BIBLIOTECA
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); //ENDEREÇO
DO I2C E OUTRAS INFORMAÇÕES

int val;
char read_pin = A0;

void setup()
{
  lcd.begin (16, 2); //QUANTIDADE DE COLUNAS(16) E DE LINHAS(2) DO DISPLAY
  lcd.setBacklight(HIGH); //LIGA O BACKLIGHT (LUZ DE FUNDO)
  Serial.begin(9600);
  pinMode(read_pin, INPUT);
  delay(10);
}

void loop()
{
  val = analogRead(read_pin);
  delay(10);
  lcd.clear();
  lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
  lcd.print("Valor="); //IMPRIME O TEXTO NO DISPLAY LCD
  lcd.print(val);
  Serial.print("Valor medido:");
  Serial.println(val);
  Serial.println();
  delay(1000);
}
```

Anexo 5 Código Arduíno de medição de carga com TARA

```
#include <Wire.h> //INCLUSÃO DE BIBLIOTECA
#include <LiquidCrystal_I2C.h> //INCLUSÃO DE BIBLIOTECA
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); //ENDEREÇO
DO I2C E OUTRAS INFORMAÇÕES

float carga;
int val;
char read_pin = A0;
char button = 4;
int button_read;
float TARA;

void setup()
{
  lcd.begin (16, 2); //QUANTIDADE DE COLUNAS(16) E DE LINHAS(2) DO DISPLAY
  lcd.setBacklight(HIGH); //LIGA O BACKLIGHT (LUZ DE FUNDO)
  Serial.begin(9600);
  pinMode(read_pin, INPUT);
  pinMode(button, INPUT);
  delay(10);
  TARA = analogRead(read_pin);
}

void loop()
{
  button_read = digitalRead(button);
  if (button_read == 1) {
    TARA = analogRead(read_pin);
    delay(10);
  }
  val = analogRead(read_pin);
  delay(10);
  carga = (13.7172 * (val - TARA)) / 1000;
  if (carga > 6) {
    lcd.clear();
    lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
    lcd.print("Valor elevado!!!"); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.setCursor(0, 1); //POSIÇÃO DO CURSOR
    lcd.print("Carga="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(carga, 2);
    lcd.print(" kg");
    Serial.println("Perigo, está a chegar perto do limite do sensor");
  }
  else {
    lcd.clear();
    lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
    lcd.print("Valor="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(val);
    lcd.setCursor(0, 1); //POSIÇÃO DO CURSOR
    lcd.print("Carga="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(carga, 2);
    lcd.print(" kg");
    Serial.print("Valor medido:");
    Serial.println(val);
    Serial.println();
```

```
    Serial.print("Carga:");
    Serial.print(carga, 4);
    Serial.println(" kg");
    Serial.println();
  }
  delay(1000);

}
```

Anexo 6 Código Arduíno de pinça com sensibilidade de carga

```
#include <Wire.h> //INCLUSÃO DE BIBLIOTECA
#include <LiquidCrystal_I2C.h> //INCLUSÃO DE BIBLIOTECA
#include <Servo.h> //INCLUSÃO DA BIBLIOTECA NECESSÁRIA
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7,3, POSITIVE);
Servo s; //OBJETO DO TIPO SERVO

const int pinoServo = 6; //PINO DIGITAL UTILIZADO PELO SERVO
int pos; //POSIÇÃO DO SERVO
int oldpos; //posição de 30 em 30° antiga
float carga;
int val;
char read_pin = A0;
int button = 4;
int ac_garra = 9;
int button_read;
int ac_read;
float TARA;
int ac_state;

void setup()
{
  lcd.begin (16,2); //QUANTIDADE DE COLUNAS(16) E DE LINHAS(2) DO DISPLAY
  lcd.setBacklight(HIGH); //LIGA O BACKLIGHT (LUZ DE FUNDO)
  s.attach(pinoServo); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO SERVO
  s.write(0); //INICIA O MOTOR NA POSIÇÃO 0°
  // Estado da garra: 0-Aberto;
  //                  1-Fechado
  ac_state = 0;
  pos = 0;
  Serial.begin(9600);
  pinMode(read_pin, INPUT);
  pinMode(button, INPUT);
  delay(10);
  TARA = analogRead(read_pin);
}

void loop()
{
  Ler_carga();
  ac_read = digitalRead(ac_garra);
  if (ac_read==1) {
    switch(ac_state) {
      case 0:
      //Chama função para fechar garra
      Fechar();
      ac_state = 1;
      break;
      case 1:
      //Chama função para abrir garra
      Abrir();
      ac_state = 0;
      break;
    }
  }
  Ler_carga();
  delay(450);
}
```

```
void Fechar(){
  Serial.print("Closing");
  while(pos < 180 && ac_state==0){ //ENQUANTO "pos" for MENOR QUE 180,
INCREMENTA, fechando garra
    Ler_carga();
    oldpos = pos;
    while(pos < oldpos+30 && ac_state==0){
      pos=pos+3;
      s.write(pos); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      val = analogRead(read_pin);
      delay(10);
      carga = (13.7172 * (val-TARA))/1000;
      if (carga > 4) {
        ac_state=1;}
        else {
        ac_state=0;}
      delay(5); //INTERVALO DE 10+5 MILISSEGUNDOS
    }
  }
  Serial.print("Closed");
}

void Abrir() {
  while(pos > 0){
    Ler_carga();
    oldpos = pos;
    while(pos > oldpos-30){
      pos=pos-3;
      s.write(pos); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      delay(15); //INTERVALO DE 15 MILISSEGUNDOS
    }
  }
}

void Ler_carga() {
  // TARA
  button_read = digitalRead(button);
  if (button_read == 1) {
    TARA = analogRead(read_pin);
    delay(10);
  }
  val = analogRead(read_pin);
  delay(5);
  carga = (13.7172 * (val-TARA))/1000;
  if (carga > 6) {
    lcd.clear();
    lcd.setCursor(0,0); //POSIÇÃO DO CURSOR
    lcd.print("Valor elevado!!!"); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.setCursor(0,1); //POSIÇÃO DO CURSOR
    lcd.print("Carga="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(carga,2);
    lcd.print(" kg");
  }
  else if ((carga >= 4) && (carga <= 6)){
    lcd.clear();
    lcd.setCursor(0,0); //POSIÇÃO DO CURSOR
    lcd.print("Carga lim. fecho"); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.setCursor(0,1); //POSIÇÃO DO CURSOR
    lcd.print("Carga="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(carga,2);
```

```
    lcd.print(" kg");
  }
  else {
    lcd.clear();
    lcd.setCursor(0,0); //POSIÇÃO DO CURSOR
    lcd.print("Valor="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(val);
    lcd.setCursor(0,1); //POSIÇÃO DO CURSOR
    lcd.print("Carga="); //IMPRIME O TEXTO NO DISPLAY LCD
    lcd.print(carga,2);
    lcd.print(" kg");
    Serial.print(carga, 4);
    Serial.println();
  }
}
```

Anexo 7 Código Arduíno de garra de três dedos sem limite máximo de força

```
#include <Wire.h> //INCLUSÃO DE BIBLIOTECA
#include <LiquidCrystal_I2C.h> //INCLUSÃO DE BIBLIOTECA
#include <Servo.h> //INCLUSÃO DE BIBLIOTECA

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); //ENDEREÇO
DO I2C E OUTRAS INFORMAÇÕES
Servo s1;
Servo s2;
Servo s3;//OBJETOS DO TIPO SERVO
const int pinoServo_1 = 9;
const int pinoServo_2 = 10;
const int pinoServo_3 = 11; //PINOS DIGITAIS UTILIZADOS PELOS SERVOS
char read_pin_1 = A0;
char read_pin_2 = A1;
char read_pin_3 = A2; //Ler carga de sensores
const int ac_garra = 5; // Acionamento do robô
const int lim_pos = 60; //Limite de variação da posição (180-lim_pos)
char potenc_read = A3;
int carga_fecho; //Valor limite de carga para fecho
int pos_1 = 180;
int oldpos_1 = 180;
int pos_2 = 180;
int oldpos_2 = 180;
int pos_3 = 180; //POSIÇÃO DO SERVO
int oldpos_3 = 180; //posição de 20 em 20° antiga
int carga_1;
int carga_2;
int carga_3;  //valor carga por cada sensor (com conversão)
int val_1;
int val_2;
int val_3; //valor carga por cada sensor (sem conversão)
float TARA_1;
float TARA_2;
float TARA_3; //Valores de TARA
// Estado da garra: 0-Fechado/A Abrir;
//                  1-Aberto/A Fechar
int ac_state_1 = 1;
int ac_state_2 = 1;
int ac_state_3 = 1; //Estado de cada dedo
int button_read;
int ac_read; // Variável que lê o valor do robô

void setup()
{
  Serial.begin(9600);
  lcd.begin (16, 2); //QUANTIDADE DE COLUNAS(16) E O NÚMERO DE LINHAS(2) DO
DISPLAY
  lcd.setBacklight(HIGH); //LIGA O BACKLIGHT (LUZ DE FUNDO)
  s1.attach(pinoServo_1); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
  s1.write(pos_1); //INICIA O MOTOR_1 NA POSIÇÃO 180°
  s2.attach(pinoServo_2); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
  s2.write(pos_2); //INICIA O MOTOR_2 NA POSIÇÃO 180°
```

```
  s3.attach(pinoServo_3); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
  s3.write(pos_3); //INICIA O MOTOR_3 NA POSIÇÃO 180°
  carga_fecho = (analogRead(potenc_read)) / 2;  //Divide o valor lido no
potenciômetro por 2
  pinMode(read_pin_1, INPUT);
  pinMode(read_pin_2, INPUT);
  pinMode(read_pin_3, INPUT);
  pinMode(potenc_read, INPUT);
  pinMode(ac_garra, INPUT);
  delay(10);
  pinMode(pinoServo_1, OUTPUT);
  pinMode(pinoServo_2, OUTPUT);
  pinMode(pinoServo_3, OUTPUT);
  TARA_1 = analogRead(read_pin_1);
  TARA_2 = analogRead(read_pin_2);
  TARA_3 = analogRead(read_pin_3);
  delay(10);
  Serial.println(ac_read);
}

void loop()
{
  Ler_carga();
  ac_read = digitalRead(ac_garra);
  Serial.println(ac_read);
  if (ac_read == 1) {
    if (ac_state_1 + ac_state_2 + ac_state_1 > 0) {
      Fechar(); //Chama função para fechar garra
      ac_state_1 = 0;
      ac_state_2 = 0;
      ac_state_3 = 0;
      delay(10);
    }
    if (ac_state_1 + ac_state_2 + ac_state_1 == 0) {
      Ler_carga();
      if (carga_1 <= carga_fecho) ac_state_1 = 1;
      else ac_state_1 = 0;
      if (carga_2 <= carga_fecho) ac_state_2 = 1;
      else ac_state_2 = 0;
      if (carga_3 <= carga_fecho) ac_state_3 = 1;
      else ac_state_3 = 0;
      delay(10); //INTERVALO DE 10 MILISSEGUNDOS
    }
  }
  Ler_carga();
  ac_read = digitalRead(ac_garra);
  Serial.println(ac_read);
  if (ac_read == 0) {
    Abrir(); //Chama função para abrir garra
    ac_state_1 = 1;
    ac_state_2 = 1;
    ac_state_3 = 1;
  }
  Ler_carga();
  delay(450);
}

void Fechar() {
  ac_read = digitalRead(ac_garra);
  Serial.println(ac_read);
```

```
  while (ac_state_1 + ac_state_2 + ac_state_3 > 0 && ac_read == 1) {
    Ler_carga();
    if (carga_1 >= carga_fecho || pos_1 == 180 - lim_pos) ac_state_1 = 0;
    else ac_state_1 = 1;
    if (carga_2 >= carga_fecho || pos_2 == 180 - lim_pos) ac_state_2 = 0;
    else ac_state_2 = 1;
    if (carga_3 >= carga_fecho || pos_3 == 180 - lim_pos) ac_state_3 = 0;
    else ac_state_3 = 1;
    delay(5); //INTERVALO DE 5 MILISSEGUNDOS
    oldpos_1 = pos_1;
    oldpos_2 = pos_2;
    oldpos_3 = pos_3;
    while (pos_1>oldpos_1-20 && pos_2>oldpos_2-20 && pos_3>oldpos_3-20 &&
ac_state_1+ac_state_2+ac_state_3>0 && ac_read==1){
      Calcular_Carga();
      if (carga_1 >= carga_fecho || pos_1 <= 180 - lim_pos) ac_state_1 = 0;
      else ac_state_1 = 1;
      if (carga_2 >= carga_fecho || pos_2 <= 180 - lim_pos) ac_state_2 = 0;
      else ac_state_2 = 1;
      if (carga_3 >= carga_fecho || pos_3 <= 180 - lim_pos) ac_state_3 = 0;
      else ac_state_3 = 1;
      pos_1 = pos_1 - ac_state_1;
      s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      pos_2 = pos_2 - ac_state_2;
      s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      pos_3 = pos_3 - ac_state_3;
      s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      delay(10);
      ac_read = digitalRead(ac_garra);
      delay(5); //INTERVALO DE 10+5 MILISSEGUNDOS
    }
    ac_read = digitalRead(ac_garra);
    Serial.println(ac_read);
  }
}

void Abrir() {
  Ler_carga();
  pos_1 = 180;
  pos_2 = 180;
  pos_3 = 180;
  s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  delay(20); //INTERVALO DE 20 MILISSEGUNDOS
}

void Ler_carga() {
  Calcular_Carga();
  lcd.clear();
  lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
  lcd.print("C1="); //IMPRIME O TEXTO NO DISPLAY LCD
  lcd.print(carga_1);
  lcd.setCursor(8, 0);
  lcd.print("C2=");
  lcd.print(carga_2);
  lcd.setCursor(0, 1); //POSIÇÃO DO CURSOR
  lcd.print("C3="); //IMPRIME O TEXTO NO DISPLAY LCD
  lcd.print(carga_3);
  lcd.setCursor(8, 1);
  lcd.print("F=");
```

```
  lcd.print(carga_fecho);
  lcd.setCursor(13, 1);
  lcd.print("[g]");
}

void Calcular_Carga() {
  carga_fecho = (analogRead(potenc_read)) / 2;
  val_1 = analogRead(read_pin_1);
  val_2 = analogRead(read_pin_2);
  val_3 = analogRead(read_pin_3);
  delay(5);
  carga_1 = (13.7172 * (val_1 - TARA_1));
  carga_2 = (13.7172 * (val_2 - TARA_2));
  carga_3 = (13.7172 * (val_3 - TARA_3));
}
```

Anexo 8 Código Arduíno de garra de três dedos com limite máximo de força

```
#include <Wire.h> //INCLUSÃO DE BIBLIOTECA
#include <LiquidCrystal_I2C.h> //INCLUSÃO DE BIBLIOTECA
#include <Servo.h> //INCLUSÃO DA BIBLIOTECA NECESSÁRIA

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); //ENDEREÇO
DO I2C E OUTRAS INFORMAÇÕES
Servo s1;
Servo s2;
Servo s3;//OBJETO DO TIPO SERVO
const int pinoServo_1 = 9;
const int pinoServo_2 = 10;
const int pinoServo_3 = 11; //PINOS DIGITAIS UTILIZADOS PELOS SERVOS
char read_pin_1 = A0;
char read_pin_2 = A1;
char read_pin_3 = A2; //Ler carga de sensores
int ac_garra = 5; // Acionamento do robot
int lim_pos = 60; //Limite de variação da posição (180-lim_pos)
int potenc_read = A3;
int carga_fecho; //Valor limite de carga para fecho
int pos_1 = 180;
int oldpos_1 = 180;
int pos_2 = 180;
int oldpos_2 = 180;
int pos_3 = 180; //POSIÇÃO DO SERVO
int oldpos_3 = 180; //posição de 20 em 20° antiga
int carga_1;
int carga_2;
int carga_3; //valor carga por cada sensor (com conversão)
int val_1;
int val_2;
int val_3; //valor carga por cada sensor (sem conversão)
float TARA_1;
float TARA_2;
float TARA_3; //Valores de TARA
  // Estado da garra: 0-Fechado/A Abrir;
  //                  1-Aberto/A Fechar
int ac_state_1 = 1;
int ac_state_2 = 1;
int ac_state_3 = 1; //Estado de cada dedo
int button_read;
int ac_read; // Variável que lê o valor do robô

void setup()
{
  lcd.begin (16, 2); //QUANTIDADE DE COLUNAS(16) E O NÚMERO DE LINHAS(2) DO
DISPLAY
  lcd.setBacklight(HIGH); //LIGA O BACKLIGHT (LUZ DE FUNDO)
  s1.attach(pinoServo_1); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
  s1.write(180); //INICIA O MOTOR_1 NA POSIÇÃO 180°
  s2.attach(pinoServo_2); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
  s2.write(180); //INICIA O MOTOR_2 NA POSIÇÃO 180°
  s3.attach(pinoServo_3); //ASSOCIAÇÃO DO PINO DIGITAL AO OBJETO DO TIPO
SERVO
```

```
  s3.write(180); //INICIA O MOTOR_3 NA POSIÇÃO 180°
  carga_fecho = analogRead(potenc_read);
  pinMode(read_pin_1, INPUT);
  pinMode(read_pin_2, INPUT);
  pinMode(read_pin_3, INPUT);
  pinMode(potenc_read, INPUT);
  pinMode(ac_garra, INPUT);
  delay(10);
  pinMode(pinoServo_1, OUTPUT);
  pinMode(pinoServo_2, OUTPUT);
  pinMode(pinoServo_3, OUTPUT);
  TARA_1 = analogRead(read_pin_1);
  TARA_2 = analogRead(read_pin_2);
  TARA_3 = analogRead(read_pin_3);
  delay(10);
}

void loop()
{
  Ler_carga();
  ac_read = digitalRead(ac_garra);
  if (ac_read == 1) {
    if (ac_state_1 + ac_state_2 + ac_state_1 > 0) {
      Fechar(); //Chama função para fechar garra
      ac_state_1 = 0;
      ac_state_2 = 0;
      ac_state_3 = 0;
      delay(10);
    }
    if (ac_state_1 + ac_state_2 + ac_state_1 == 0) {
      Ler_carga();
      if (carga_1 <= carga_fecho || carga_1 > carga_fecho*2) ac_state_1 =
1;
      else ac_state_1 = 0;
      if (carga_2 <= carga_fecho || carga_2 > carga_fecho*2) ac_state_2 =
1;
      else ac_state_2 = 0;
      if (carga_3 <= carga_fecho || carga_3 > carga_fecho*2) ac_state_3 =
1;
      else ac_state_3 = 0;
      delay(10); //INTERVALO DE 10 MILISSEGUNDOS
    }
  }
  Ler_carga();
  ac_read = digitalRead(ac_garra);
  if (ac_read == 0) {
    Abrir(); //Chama função para abrir garra
    ac_state_1 = 1;
    ac_state_2 = 1;
    ac_state_3 = 1;
  }
  Ler_carga();
  delay(450);
}

void Fechar() {
  ac_read = digitalRead(ac_garra);
  while (ac_state_1 + ac_state_2 + ac_state_3 > 0 && ac_read == 1) {
    Ler_carga();
    if (carga_1 >= carga_fecho || pos_1 == 180 - lim_pos) {
      if (carga_1 > carga_fecho*2) {
```

```
        pos_1 = pos_1 + 1;
        s1.write(pos_1);
      }
      ac_state_1 = 0;
    }
    else ac_state_1 = 1;
    if (carga_2 >= carga_fecho || pos_2 == 180 - lim_pos) {
      if (carga_2 > carga_fecho*2) {
        pos_2 = pos_2 + 1;
        s2.write(pos_2);
      }
      ac_state_2 = 0;
    }
    else ac_state_2 = 1;
    if (carga_3 >= carga_fecho || pos_3 == 180 - lim_pos) {
      if (carga_3 > carga_fecho*2) {
        pos_3 = pos_3 + 1;
        s3.write(pos_3);
      }
      ac_state_3 = 0;
    }
    else ac_state_3 = 1;
    delay(5); //INTERVALO DE 5 MILISSEGUNDOS
    oldpos_1 = pos_1;
    oldpos_2 = pos_2;
    oldpos_3 = pos_3;
    while (pos_1 > oldpos_1 - 20 && pos_2 > oldpos_2 - 20 && pos_3 >
oldpos_3 - 20 && ac_state_1 + ac_state_2 + ac_state_3 > 0 && ac_read == 1)
{
      Calcular_Carga();
      if (carga_1 >= carga_fecho || pos_1 <= 180 - lim_pos) ac_state_1 = 0;
      else ac_state_1 = 1;
      if (carga_2 >= carga_fecho || pos_2 <= 180 - lim_pos) ac_state_2 = 0;
      else ac_state_2 = 1;
      if (carga_3 >= carga_fecho || pos_3 <= 180 - lim_pos) ac_state_3 = 0;
      else ac_state_3 = 1;
      pos_1 = pos_1 - ac_state_1;
      s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      pos_2 = pos_2 - ac_state_2;
      s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      pos_3 = pos_3 - ac_state_3;
      s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
      delay(10);
      ac_read = digitalRead(ac_garra);
      delay(5); //INTERVALO DE 5+10 MILISSEGUNDOS
    }
    ac_read = digitalRead(ac_garra);
  }
}

void Abrir() {
  Ler_carga();
  pos_1 = 180;
  pos_2 = 180;
  pos_3 = 180;
  s1.write(pos_1); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  s2.write(pos_2); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  s3.write(pos_3); //ESCREVE O VALOR DA POSIÇÃO QUE O SERVO DEVE GIRAR
  delay(20); //INTERVALO DE 15 MILISSEGUNDOS
}
```

```
void Ler_carga() {
  Calcular_Carga();
  lcd.clear();
  lcd.setCursor(0, 0); //POSIÇÃO DO CURSOR
  lcd.print("C1="); //IMPRIME O TEXTO NO DISPLAY LCD
  lcd.print(carga_1);
  lcd.setCursor(8, 0);
  lcd.print("C2=");
  lcd.print(carga_2);
  lcd.setCursor(0, 1); //POSIÇÃO DO CURSOR
  lcd.print("C3="); //IMPRIME O TEXTO NO DISPLAY LCD
  lcd.print(carga_3);
  lcd.setCursor(8, 1);
  lcd.print("F=");
  lcd.print(carga_fecho);
  lcd.setCursor(13, 1);
  lcd.print("[g]");
}

void Calcular_Carga() {
  carga_fecho = (analogRead(potenc_read)) / 2;
  val_1 = analogRead(read_pin_1);
  val_2 = analogRead(read_pin_2);
  val_3 = analogRead(read_pin_3);
  delay(5);
  carga_1 = (13.7172 * (val_1 - TARA_1));
  carga_2 = (13.7172 * (val_2 - TARA_2));
  carga_3 = (13.7172 * (val_3 - TARA_3));
}
```