Politécnico
de Viseu

Escola Superior
de Tecnologia
e Gestão de Viseu

Indústria 4.0: Identificação e Seriação com Recurso a Visão Artificial e Monitorização IoT de um Sistema de Armazenamento

Joel Lopes dos Santos

Dissertação

Mestrado em Energia e Automação Industrial

Trabalho efetuado sob a orientação de
Professor Doutor Miguel Francisco Martins de Lima

Abril de 2026

Politécnico
de Viseu

Escola Superior
de Tecnologia
e Gestão de Viseu

Indústria 4.0: Identificação e Seriação com Recurso a Visão Artificial e Monitorização IoT de um Sistema de Armazenamento

Joel Lopes dos Santos

Dissertação

Mestrado em Energia e Automação Industrial

Trabalho efetuado sob a orientação de

Professor Doutor Miguel Francisco Martins de Lima

Abril de 2026

AGRADECIMENTOS

Agradeço aos meus pais e à minha irmã. Tenho a felicidade de poder dizer que sou grato pela presença dos demais e reconheço a sorte que tenho em poder contar com eles, em todos os momentos.

Agradeço, igualmente, a toda a comunidade que integra e representa o Instituto Politécnico de Viseu, nomeadamente, aos professores, orientador, bem como a todos os meus colegas. A vontade em conhecer o mundo e a coerência racional nasce do convívio com os demais.

Acima de tudo, estou grato a mim mesmo, por ter concretizado este desafio, com resiliência e bondade.

RESUMO

A digitalização de processos ao nível industrial, associada ao uso de princípios com base na Indústria 4.0, tem motivado a integração de sistemas físicos com tecnologias de análise e/ou tratamento de dados. Neste contexto, desenvolveu-se um trabalho, que consistiu na modernização de uma maquete existente no Departamento de Engenharia Eletrotécnica da Escola Superior de Tecnologia e Gestão de Viseu, originalmente realizada como sistema de triagem de peças.

Foram impressas peças de forma a melhorar o funcionamento mecânico do sistema e foi criado um modelo tridimensional completo da maquete em ambiente CAD. Desta forma permite-se, não só a possibilidade de identificar melhorias futuras a nível mecânico com uma maior facilidade, mas também faz da mesma passível de ser utilizada em ambientes de simulação ou desenvolvimentos que integram o âmbito dos *Digital Twins* (DT).

O controlo do sistema foi possível através de um PLC Siemens S7-1214C programado em linguagem SCL, sendo este responsável pela gestão das tarefas a serem realizadas nos diferentes modos de operação. A identificação do tipo de peça e a consequente validação foi realizada através de um sistema de visão artificial recorrendo a um ambiente *python* num *Raspberry Pi*. Para a integração do sistema de visão com o PLC foi utilizado o protocolo de comunicação S7.

Adicionalmente, foi desenvolvido um ambiente de interface *web* para monitorização de dados em tempo real.

Palavras-chave: Internet of Things; Digital Twins; Indústria 4.0; Automação; Eficiência; Processos Produtivos, Snap7; Raspberry Pi.

ABSTRACT

The principle of process digitalisation at the industrial level, associated with the use of Industry 4.0 concepts, has encouraged the integration of physical systems with data analysis and processing technologies. In this context, the present work consisted of modernising an existing didactic model located at the Department of Electrical Engineering of the School of Technology and Management of Viseu, originally developed as a part-sorting system.

New components were manufactured using additive manufacturing in order to improve the mechanical operation of the system, and a complete three-dimensional CAD model of the platform was created. This approach not only facilitates future mechanical improvements but also enables the model to be used in simulation environments or in future developments related to Digital Twin integration.

The system control was implemented using a Siemens S7-1214C PLC programmed in Structured Control Language (SCL), responsible for managing the tasks executed in the different operating modes. Part identification and subsequent validation were performed through a machine vision system implemented in Python on a Raspberry Pi platform. Integration with the PLC and the required communication were achieved using the S7 protocol.

Additionally, a web-based interface was developed to enable real-time data monitoring

Keywords: Internet of Things; Digital Twins; Industry 4.0; Automation; Industry; Efficiency; Productive Processes; Snap7; Raspberry Pi.

ÍNDICE GERAL

ÍNDICE DE TABELAS / QUADROS

ÍNDICE DE FIGURAS

LISTA DE SIGLAS / ABREVIATURAS

AI – *Artifical Inteligence*

API – *Application Programming Interface*

CAD – *Computer Aided-Design*

CLP – Controlador Lógico Programável

CPS – *Cyber-Physical Systems*

DB – *Data Block*

DEE – Departamento de Engenharia Eletrotécnica

DT – *Digital Twin*

EOL – *End of Line*

ESTGV – Escola Superior de Tecnologia e Gestão de Viseu

IA – *Inteligência Artificial*

IIoT – *Industrial Internet of Things*

IoT – *Internet of Things*

JSON – *JavaScript Object Notation*

HMI – *Human-Machine Interface*

HSV – *Hue, Saturation, and Value*

IPV – Instituto Politécnico de Viseu

PA – Poliamida

PLC – *Programmable Logic Controller*

RGB – *Red, Green, Blue*

ROI – *Region of Interest*

RPI – Reposição da Posição Inicial

1. INTRODUÇÃO

A crescente adoção da digitalização de sistemas industriais tem conduzido a profundas transformações ao nível do controlo de processos produtivos, bem como na forma como a sua monitorização e análise são concebidas. A evolução tecnológica tem impulsionado a integração de sistemas físicos com plataformas digitais, permitindo que os processos produtivos deixem de ser apenas sequenciais e passem a ser orientados por dados, com capacidade de adaptação dinâmica com as condições de operação (Abayadeera & Ganegoda, 2024).

Neste contexto, as soluções industriais modernas baseiam-se na recolha contínua de dados provenientes de sistemas físicos, os quais são posteriormente processados e disponibilizados em plataformas que permitem a sua visualização, análise e interpretação. Esta interligação entre o mundo físico e o digital constitui um dos pilares fundamentais da atual transformação industrial, sendo suportada por tecnologias como sensores inteligentes, redes de comunicação e sistemas computacionais distribuídos (Xu et al., 2014).

A evolução destes sistemas enquadra-se no conceito de Indústria 4.0, que promove a integração de tecnologias como a Internet das coisas, os gémeos digitais e o processamento de dados recorrendo a técnicas de inteligência artificial. Estas tecnologias permitem a criação de sistemas capazes de monitorizar o seu próprio estado, antecipar comportamentos e otimizar o seu funcionamento, contribuindo para uma maior eficiência e fiabilidade dos processos produtivos (Bottazzi et al., 2025).

Neste enquadramento, os PLC assumem um papel central no controlo dos sistemas industriais, sendo responsáveis pela gestão das entradas e saídas e pela execução da lógica de controlo. A sua elevada robustez e determinismo tornam-nos elementos fundamentais no controlo dos processos físicos. No entanto, a necessidade de executar

tarefas mais complexas, como o processamento de imagem ou a análise de grandes volumes de dados, conduz à integração de sistemas externos que complementam as suas funcionalidades (Xu et al., 2014).

Em paralelo, os sistemas de visão artificial têm vindo a assumir um papel cada vez mais relevante na automação industrial, permitindo a automatização de tarefas de inspeção, classificação e controlo de qualidade. Estes sistemas baseiam-se na aquisição e processamento de imagens digitais, sendo o seu desempenho diretamente influenciado pela qualidade da imagem e pelos parâmetros de aquisição, como iluminação, exposição e foco (Roberg et al., 2025).

Os algoritmos de visão são capazes de identificar defeitos, realizar medições dimensionais e efetuar classificações automáticas, podendo ainda integrar técnicas de aprendizagem automática que aumentam a sua robustez em ambientes industriais complexos (Ha et al., 2025).

Outro conceito relevante neste contexto é o de gémeos digitais, que consiste na criação de uma representação digital de um sistema físico, capaz de refletir o seu comportamento em tempo real. Estes sistemas combinam dados provenientes de sensores, dados históricos e modelos computacionais, permitindo não só a monitorização, mas também a simulação e previsão de comportamentos futuros (Matta & Lugaresi, 2024).

Em seguida, é apresentado um exemplo representativo (figura 1, adaptada do *website* da *Nvidia*), resultante da colaboração entre a *Dassault Systemes* e a *Omron*, no âmbito da investigação do conceito de gémeos digitais.

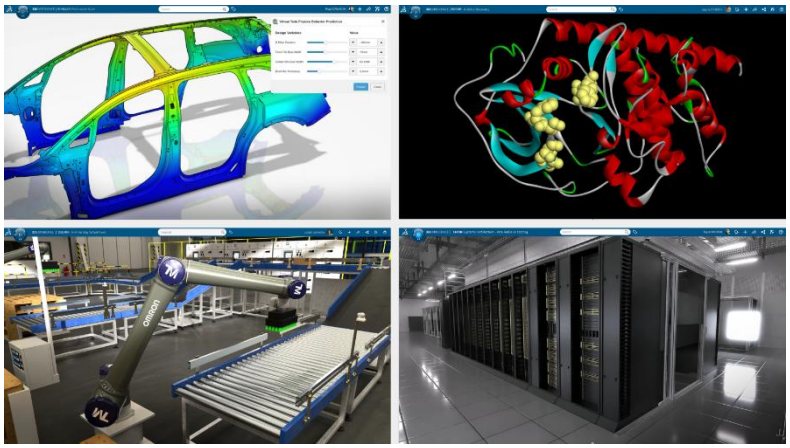


Figura 1 - Representação de diferentes ambientes virtuais baseados em modelos físicos.

A integração destas tecnologias permite criar sistemas industriais mais eficientes, flexíveis e adaptáveis, sendo este o principal enquadramento do trabalho desenvolvido na presente dissertação.

1.1. Enquadramento

A automação industrial tem desempenhado um papel determinante na modernização dos processos produtivos, contribuindo para o aumento da eficiência, melhoria da qualidade dos produtos e redução da intervenção humana em tarefas repetitivas. A evolução destes sistemas tem sido fortemente influenciada pela integração de tecnologias digitais, permitindo a criação de sistemas mais flexíveis e adaptáveis (Baltazar, 2021).

Os PLC continuam a ser amplamente utilizados como elemento central de controlo, garantindo fiabilidade e robustez na execução das tarefas. No entanto, apresentam limitações ao nível da execução de tarefas computacionalmente exigentes, o que conduz à necessidade de integração com sistemas externos.

Neste contexto, dispositivos como o *Raspberry Pi* (Figura 2, adaptada do *website* do *Raspberry Pi*) surgem como soluções viáveis para a execução de tarefas como o processamento de imagem e visão computacional, permitindo a implementação de algoritmos complexos com um custo reduzido.

Figura 2 - Raspberry Pi 5.

A integração destes sistemas com plataformas de monitorização, nomeadamente interfaces *web*, permite a visualização de dados em tempo real e a análise do desempenho do sistema, contribuindo para uma gestão mais eficiente dos processos.

1.2. Objetivos da investigação

O objetivo principal da presente dissertação consiste na modernização de uma maquete de triagem ("*sorter*") já existente, e a criação do seu modelo 3D em ambiente CAD, o que permitirá, numa futura aplicação, a sua integração numa solução de ambiente *Digital Twin*.

De uma forma direta, com o trabalho desenvolvido pretende-se atingir os seguintes objetivos:

- Modernizar o sistema mecânico da maquete, utilizando novos componentes modelados em ambiente CAD e impressos em 3D, com o objetivo de melhorar a repetibilidade do sistema.
- Desenvolver um modelo tridimensional completo de forma a permitir, com uma maior facilidade, alterações futuras e, principalmente, possibilitando a utilização do modelo em ambientes virtuais.
- Implementar um sistema de controlo do sistema de triagem recorrendo a um PLC Siemens 1214C, programado em SCL, elemento gestor das diferentes fases do processo de triagem.
- Desenvolver um sistema de visão artificial utilizando um *Raspberry Pi* e uma *webcam* Logitech C920, com capacidade de identificar o tipo de peça alimentada e reconhecendo o seu rigor dimensional recorrendo a um algoritmo escrito em linguagem *Python*.
- Integrar o sistema de visão artificial ao sistema de controlo através de um protocolo de comunicação em rede, permitindo troca de dados e ordens entre os sistemas.
- Implementar uma interface simples de monitorização baseada numa página *web*, permitindo visualizar valores de produção.

Assim, pretende-se demonstrar que a integração destes sistemas, a um nível académico, pode contribuir para uma aproximação entre as entidades formadoras e os formandos a temas diretamente relacionados com a realidade industrial.

Adicionalmente, pretende-se evidenciar a viabilidade da utilização de arquiteturas distribuídas, onde sistemas de controlo determinísticos (PLC) coexistem com sistemas computacionais dedicados (*Raspberry Pi*), permitindo a execução de tarefas

complementares como o processamento de imagem e a disponibilização de dados em tempo real, em alinhamento com os princípios da Indústria 4.0 (Xu et al., 2014).

1.3. Abordagem metodológica

A abordagem metodológica adotada baseia-se na adaptação de um sistema de automação existente, combinada com a adição de novos elementos responsáveis pela sua modernização ao nível pretendido, tornando-o capaz de realizar funções adicionais, nomeadamente ao nível do controlo auxiliar e do processamento de imagem.

Numa primeira fase, foi realizada uma análise ao equipamento existente com o objetivo de identificar possíveis limitações e entravamentos, tanto a nível mecânico como elétrico, de forma a serem corrigidos no âmbito deste trabalho. Com base nesta análise, foram desenvolvidas e implementadas soluções mecânicas que permitiram melhorar o funcionamento do sistema, contribuindo para uma maior estabilidade e repetibilidade do processo de triagem.

Posto isto, foi desenvolvido um novo programa em linguagem *SCL*, de forma a permitir o controlo pelo *PLC Siemens* S7-1214C, responsável pela gestão dos diferentes modos de funcionamento, execução da sequência de triagem e comunicação com o sistema externo. Este sistema assegura o controlo determinístico do processo, garantindo a correta execução das operações e a coordenação entre os diferentes atuadores e sensores.

Paralelamente, foi desenvolvido um algoritmo para o sistema de visão artificial, implementado em *Python*, que permite ao *Raspberry Pi* aceder à *webcam*, capturar imagens e proceder à identificação da peça e à validação da sua integridade. O algoritmo desenvolvido baseia-se em técnicas de processamento digital de imagem, permitindo extrair características relevantes da peça e classificá-la de acordo com critérios definidos.

Após o desenvolvimento individual de cada uma das soluções, procedeu-se à sua integração, com o objetivo de obter um sistema completo de controlo e visão artificial. Esta integração foi realizada através de comunicação em rede, recorrendo ao protocolo S7, permitindo a troca de informação entre o PLC e o *Raspberry Pi*. Desta forma, o *PLC* emite um pedido de análise ao sistema de visão, sendo posteriormente recebido o resultado da classificação, que é utilizado na tomada de decisão do processo de triagem.

Adicionalmente, foi implementado no *Raspberry Pi* um sistema de exportação de dados para uma interface *web*, recorrendo a um servidor desenvolvido em *Flask*. Este sistema estabelece comunicação direta com o PLC através da leitura de blocos de dados, permitindo recolher dados estatísticos e disponibilizá-los numa plataforma de monitorização acessível através da rede. A aplicação desenvolvida integra uma *API* em formato JSON para acesso aos dados e uma interface gráfica simples (*dashboard*), que apresenta os valores em tempo real.

Desta forma, a arquitetura do sistema assenta na separação de funções entre os diferentes componentes: o *PLC* responsável pelo controlo determinístico do processo, o *Raspberry Pi* responsável pelo processamento de imagem e pela intermediação de dados, e a interface *web* responsável pela monitorização. Esta abordagem permite uma maior flexibilidade e escalabilidade do sistema, estando alinhada com os princípios da Indústria 4.0 e com arquiteturas distribuídas baseadas na Internet industrial das coisas (*IIoT*) (Xu et al., 2014).

1.4. Estrutura da dissertação

Esta dissertação encontra-se organizada em cinco capítulos distintos.

No capítulo 1 é apresentada a introdução aos diferentes assuntos, de forma a fornecer uma contextualização geral dos temas em estudo. O mesmo inclui um

enquadramento do problema, os objetivos da investigação, a abordagem metodológica adotada e a estrutura global da dissertação.

O capítulo 2 apresenta uma revisão da literatura, espelhando o estado da arte, onde são abordados temas relacionados com a automação industrial, com a indústria 4.0, visão artificial e gémeos digitais.

No capítulo 3 descreve-se o desenvolvimento da arquitetura do sistema e referem-se todos os materiais envolvidos que tornaram o projeto exequível e os métodos utilizados que permitiram a validação dos resultados. É também apresentada toda a lógica utilizada para o controlo do PLC, bem como o algoritmo utilizado pelo *Raspberry Pi* e o respetivo processo de *setup*.

No capítulo 4 apresenta-se uma análise de resultados extraídos da fase de desenvolvimento das soluções do sistema e a discussão relativa aos resultados obtidos que avaliam todo o desempenho do sistema e as limitações encontradas.

Por fim, no capítulo 5 são apresentadas conclusões quanto ao trabalho realizado e possíveis direções futuras de forma a melhorar o sistema.

2. REVISÃO DA LITERATURA

A crescente digitalização dos sistemas industriais, aliada à evolução das tecnologias de informação e comunicação, tem conduzido à necessidade de uma análise aprofundada dos conceitos e tecnologias que suportam os sistemas de automação modernos. Neste contexto, a revisão da literatura assume um papel fundamental na compreensão dos principais paradigmas que caracterizam a indústria atual.

A Indústria 4.0 surge como o principal enquadramento desta transformação, promovendo a integração entre sistemas físicos e digitais através de tecnologias como a Internet industrial das Coisas, os sistemas ciber-físicos e os gémeos digitais. Estas tecnologias permitem não só a monitorização e controlo dos processos produtivos, mas também a sua otimização contínua com base em dados recolhidos em tempo real (Bottazzi et al., 2025; Xu et al., 2014).

Paralelamente, o desenvolvimento de sistemas de visão artificial e de técnicas de processamento digital de imagem tem vindo a assumir um papel cada vez mais relevante na automação industrial, permitindo a automatização de tarefas de inspeção, classificação e controlo de qualidade, com elevados níveis de precisão e repetibilidade.

Adicionalmente, o conceito de gémeo digital tem ganho particular destaque ao possibilitar a criação de representações digitais de sistemas físicos, sendo capazes de refletir o seu comportamento em tempo real. Esta abordagem permite não só a monitorização e análise dos sistemas, mas também a simulação e previsão de comportamentos futuros, contribuindo para uma melhoria da tomada de decisão e da eficiência dos processos produtivos.

Neste sentido, o presente capítulo tem como objetivo apresentar uma visão geral dos principais conceitos e tecnologias associados à automação industrial moderna,

servindo de base teórica para o desenvolvimento do sistema de triagem apresentado nesta dissertação.

2.1. Introdução à automação

A digitalização de soluções industriais tem conduzido a uma transformação iminente da forma como são concebidos, monitorizados e controlados os processos produtivos. Ao longo das últimas décadas, com a própria evolução das tecnologias em geral, a integração de sistemas físicos e sistemas computacionais tornou-se possível, originando novos paradigmas associados à indústria 4.0. Estes paradigmas assentam no uso de tecnologias de ponta, como é o caso da própria *IIoT*, os modelos virtuais gerados e a análise/tratamento de dados, como fontes capazes de representar todo o modelo físico real (Iranshahi et al., 2025; Xu et al., 2014).

A figura 3 (adaptada de Xu et al., 2014) apresenta um conjunto de tecnologias que se podem encontrar num processo industrial baseadas na evolução da *internet* das coisas.

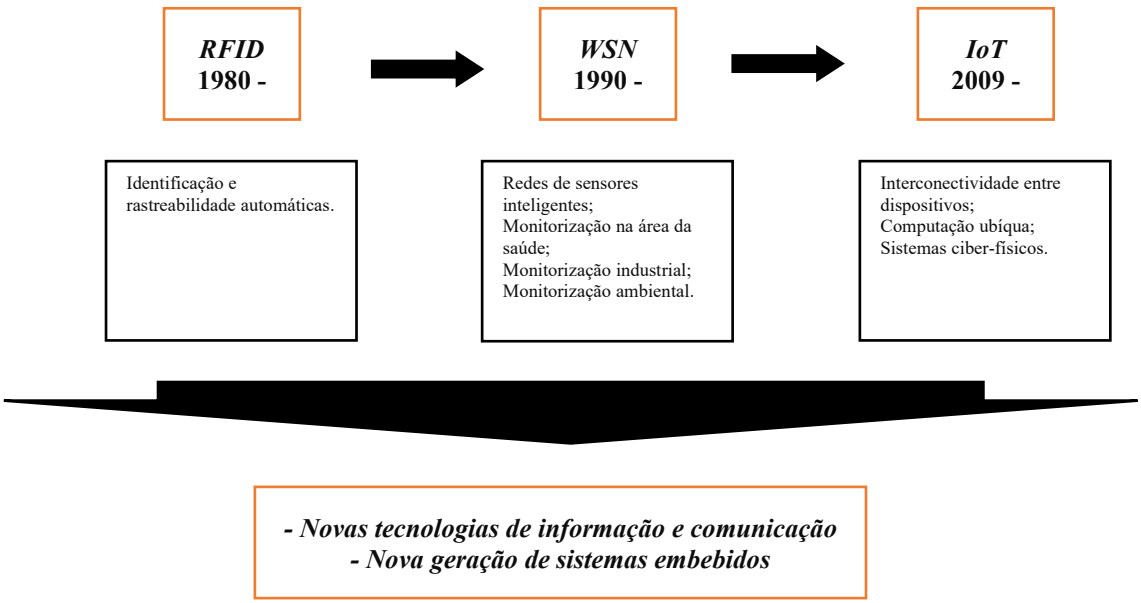


Figura 3 - Evolução das tecnologias associadas à Internet das Coisas, desde sistemas RFID até à integração em sistemas IoT e ciber-físicos.

No âmbito da automação industrial, os controladores programáveis continuam a desempenhar um papel central no controlo do processo industrial. Contudo, e dado o aumento do grau de complexidade exigida pelos sistemas atuais e perante a necessidade de integrar capacidades modernas de processamento da informação e validação de resultados, são frequentemente implementados sistemas externos com capacidade de executar tarefas com um maior grau de exigência computacional, tais como o processamento de imagem, a análise de dados ou algoritmos de inteligência artificial.

Esta abordagem permite uma separação funcional entre o controlo determinístico do processo, assegurado pelo *PLC*, e o processamento de informação mais complexo, delegado em sistemas computacionais dedicados, contribuindo para uma maior eficiência e flexibilidade do sistema global (Bellavista et al., 2023).

Historicamente, os sistemas eram caracterizados pela baixa modularidade e elevada complexidade aquando de uma reorganização. Ainda assim, tendo em consideração todos os avanços tecnológicos no setor da comunicação industrial, tornou-se possível a integração de sensores, controladores, sistemas com função de supervisão e plataformas de análise e tratamento de dados mais diversificadas (Xu et al., 2014).

Esta evolução permitiu o desenvolvimento de sistemas mais modulares e escaláveis, capazes de se adaptar a diferentes requisitos de produção e de integrar novas funcionalidades sem alterações estruturais significativas.

É neste contexto que surge o conceito de Indústria 4.0. Este é representado por um novo paradigma industrial baseado na integração entre sistemas físicos e sistemas digitais. O paradigma promove a utilização de sistemas ciber-físicos, a Internet das coisas e plataformas de tratamento de dados com capacidade de mimetizar o estado físico real dos diversos sistemas (desde sistemas de triagem, de montagem ou até de auxílio a operações manuais) (Bottazzi et al., 2025).

Entre os maiores protagonistas, os conceitos que têm vindo a ganhar maior popularidade face à sua utilidade são os sistemas de visão artificial e os *Digital Twins*. O aumento de fiabilidade dos resultados gerados por sistemas em que sejam aplicadas este tipo de soluções de visão constitui uma prova do correto uso e do potencial que a tecnologia carrega. Da mesma forma, com a virtualização de um ambiente real como forma de reconhecimento de problemas durante um dado processo, monitorização de estados e acessível tratamento de dados, torna-se possível reconhecer, de forma prévia, erros gerados ou potenciais encravamentos do sistema (Leon-Medina et al., 2025; Matta & Lugaresi, 2024).

Para além disso, a integração destas tecnologias permite não só a monitorização, mas também a implementação de estratégias de manutenção preditiva, baseadas na análise contínua do comportamento do sistema ao longo do tempo (Benhanifia et al., 2025).

Neste contexto, o presente capítulo visa apresentar uma visão generalizada das principais tecnologias associadas ao desenvolvimento/modernização deste sistema de triagem, incluindo temas sobre a automação industrial com base em PLC, Indústria 4.0, sistemas ciber-físicos, *Industrial Internet of Things*, visão artificial, processamento de imagem e gémeos digitais.

2.2. A evolução da automação industrial

A automação tem vindo a desempenhar um papel cada vez mais fundamental no desenvolvimento de soluções aplicadas na indústria moderna. Desde os primeiros sistemas de controlo eletromecânicos até aos sistemas digitais que atualmente são apresentados nas integrações, a automação contribuiu com um papel fundamental na

redução de falhas, aumento de eficiência, aumento de qualidade do produto final e aumento da produtividade dos processos (Baltazar, 2021).

Numa primeira fase da automação industrial, este tipo de sistema de controlo era baseado numa arquitetura plena de dispositivos eletromecânicos, relés, temporizadores e contactores. Estes equipamentos permitiam a implementação de uma sequência lógica, porém simples, dada a complexidade e dimensão das soluções. A mesma demonstrou, numa época inicial, potenciais entravamentos a nível da sua flexibilidade em aplicações mais modernas, bem como de manutenção (Baltazar, 2021; Oliveira et al., 2024).

A introdução dos controladores lógicos programáveis no final da década de 1960 representou um avanço crítico que revolucionou a indústria da automação. Estes novos equipamentos seriam já providos da capacidade de implementar uma lógica programável através de *software* (Oliveira et al., 2024).

Estes controladores apresentam inúmeras vantagens em relação aos sistemas tradicionais com base num funcionamento por relés, oferecendo uma maior flexibilidade, facilidade de programação da solução e a capacidade de integração com sistemas mais modernos de automação industrial (Baltazar, 2021; Oliveira et al., 2024).

Esta evolução permitiu a transição de sistemas rígidos para sistemas mais flexíveis e reconfiguráveis, capazes de responder a diferentes necessidades de produção, sendo este um dos fatores que impulsionou o desenvolvimento dos sistemas industriais modernos.

Adicionalmente, a introdução de sistemas programáveis possibilitou a integração progressiva de redes de comunicação industrial, abrindo caminho à interligação entre diferentes equipamentos e sistemas, característica fundamental dos ambientes industriais atuais.

A tabela 1 (adaptado de Pérez-Domínguez et al., 2023) apresenta a evolução dos sistemas industriais ao longo do tempo.

1784	1870	1969	2011	Futuro
Indústria 1.0	**Indústria 2.0**	**Indústria 3.0**	**Indústria 4.0**	**Indústria 5.0**
- Produção mecânica; - Energia proveniente da água e vapor.	- Divisão do trabalho; - Produção em massa; - Energia elétrica.	- Eletrónica; - Sistemas de IT; - Automatização; - Produção;	- *IoT* (Internet das coisas); - Robótica e inteligência artificial; - *Big Data;* - Computação na nuvem.	- Robótica e inteligência artificial; - Sustentabilidade; - Recursos renováveis; - Biónica;
Primeiro tear mecanizado	Primeira linha de montagem	Primeiro PLC	Sistemas físicos cibernéticos	Trabalho colaborativo humano-robô. Bioeconomia

Tabela 1 - Evolução dos sistemas de automação industrial ao longo das diferentes revoluções industriais.

2.3. A indústria 4.0

A origem do conceito de Indústria 4.0 surgiu nos inícios da segunda década deste século através da procura pela modernização da indústria com base em sistemas de digitalização de processos produtivos. Esta fase representa a quarta revolução industrial e caracteriza-se pela integração destas tecnologias digitais emergentes a sistemas de produção (Bottazzi et al., 2025).

Das principais tecnologias que representam diretamente sistemas implementados na filosofia da Indústria 4.0 destacam-se os sistemas ciber-físicos, a *Industrial Internet of Things*, a análise e tratamento de dados em larga escala, o *cloud computing* e a inteligência artificial (Bottazzi et al., 2025; Xu et al., 2014).

Com a integração destas tecnologias é possível criar fábricas inteligentes, nas quais, as soluções munidas de sensores e ações e os sistemas computacionais comunicam entre si, de forma a melhorar o funcionamento do processo produtivo.

Esta integração permite ainda a recolha contínua de dados ao longo do processo produtivo, possibilitando a sua análise em tempo real e a implementação de estratégias de otimização baseadas em dados, contribuindo para uma maior eficiência e capacidade de adaptação dos sistemas industriais (Iranshahi et al., 2025).

A implementação de cenários da Indústria 4.0 baseia-se em quatro princípios fundamentais: intercomunicação entre sistemas, transparência da informação, assistência técnica prestada e a descentralização das tomadas de decisão (Bottazzi et al., 2025).

Estes princípios permitem a criação de sistemas industriais mais autónomos e interligados, capazes de operar de forma descentralizada, promovendo uma maior flexibilidade e eficiência nos processos produtivos.

2.4. Sistemas ciber-físicos

Os sistemas *cyber*-físicos (*Cyber-Physical Systems* – CPS) são a representação da integração de sistemas computacionais e os próprios processos físicos desempenhados pelo processo. Nestas soluções, elementos tais como sensores são capazes de realizar a recolha de informação de forma que, posteriormente, seja processada por um algoritmo computacional capaz de tomar decisões e enviar comandos de ação para o processo (por exemplo, avançar um dado cilindro pneumático após recebida a informação de que o sensor de recuado se encontra ativo).

Segundo Lee et al., os CPS são um dos pilares fundamentais da Indústria 4.0, permitindo a criação de sistemas industriais com capacidade de monitorização e controlo dos processos físicos, possibilitando que o mesmo seja feito em tempo real, permitindo o seu acesso posterior.

Estes sistemas são contemplados com a possibilidade de estabelecer uma ligação contínua entre o mundo físico (real) e o mundo digital, criando uma ponte que permite a criação de sistemas industriais capazes de adaptar o seu comportamento em função dos modos e condições de operação (Bellavista et al., 2023; Iranshahi et al., 2025).

Esta interação contínua entre os dois domínios permite uma maior capacidade de adaptação do sistema, sendo um dos elementos fundamentais na implementação de sistemas inteligentes no contexto industrial.

No contexto da automação industrial (figura 4), os CPS constituem a base para a integração de tecnologias como a *IIoT* e os gémeos digitais, permitindo a sincronização entre dados reais e modelos digitais, característica essencial nos sistemas modernos.

Figura 4 - Exemplo de ambiente industrial automatizado moderno.

2.5. *Internet* industrial das coisas (IIoT)

A *internet* das coisas (*IoT*) consiste na interligação dos dispositivos físicos, de métrica real, através de redes de comunicação que satisfazem a recolha e transmissão de dados em diferentes sistemas, numa só direção ou na bidirecionalidade dos comandos a serem executados (Xu et al., 2014).

Em contexto industrial, este conceito é, muitas vezes, referido como *internet* industrial das coisas (ou *IIoT*). Este *IIoT* possibilita a ligação de sensores, máquinas e sistemas de controlo a redes de comunicação que integram e promovem a comunicação entre os sistemas, permitindo a comunicação e transmissão de dados sobre o funcionamento do processo em tempo real.

Esta capacidade de comunicação contínua entre dispositivos permite a criação de sistemas distribuídos, onde diferentes componentes colaboram entre si para a execução de tarefas, característica essencial dos sistemas industriais modernos (Xu et al., 2014).

Na figura 5 (adaptado de Xu at Al., 2014) apresentam-se algumas tecnologias associadas à internet das coisas no âmbito da indústria 4.0.

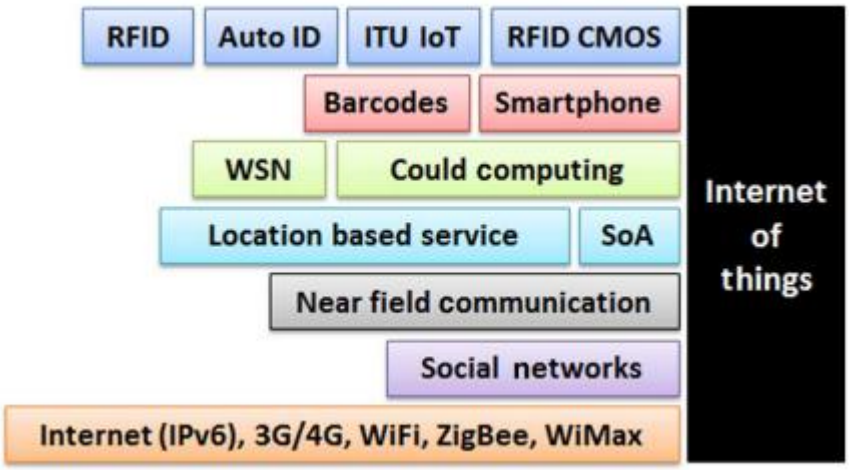


Figura 5 - Exemplo de tecnologias associadas à Indústria 4.0.

Segundo Xu et al., a utilização desta ferramenta visa integrar sistemas de informação a sistemas de produção, oferecendo a possibilidade de implementação em aplicações avançadas de monitorização, controlo e análise de informação.

Adicionalmente, a *IIoT* constitui uma base fundamental para a implementação de conceitos como os gémeos digitais, permitindo a recolha de dados em tempo real

necessários para a sincronização entre o sistema físico e o modelo digital (Iranshahi et al., 2025).

2.6. Sistemas de visão artificial industriais

Os sistemas de visão artificial desempenham hoje um papel importante no setor industrial da automação. Estas técnicas são utilizadas, geralmente, para tarefas de supervisão, identificação de peça (ou apenas de deteção de presença de peça) e controlo dimensional.

Segundo Szeliski, a visão computacional consiste no desenvolvimento de métodos capazes de extrair informação útil a um dado processo através de imagens digitais onde são aplicados complexos algoritmos de processamento de imagem e reconhecimento de formas/padrões (Szeliski, 2010).

A figura 6 (adaptado de Martins Almeida, 2021) apresenta, como exemplo, uma aplicação de um sistema de visão desenvolvido em meio académico.

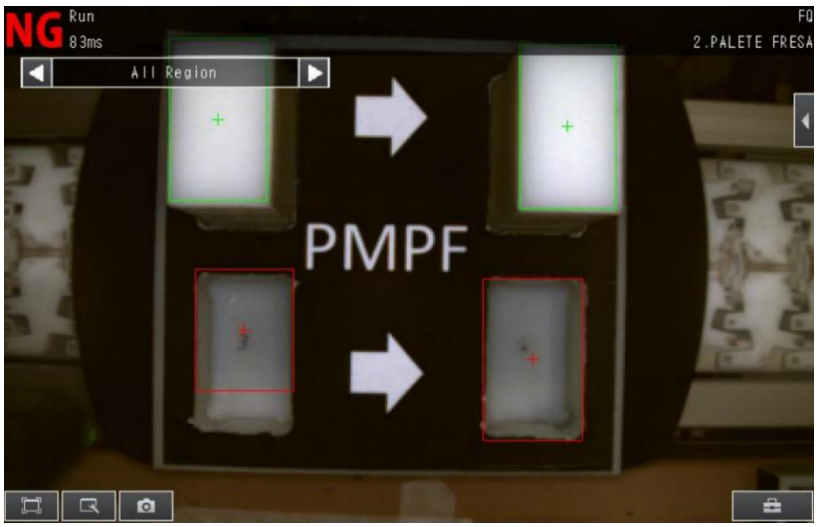


Figura 6 - Captura de imagem num processo de enchimento de paletes de uma CFF.

Este tipo de sistemas apresenta diferentes vantagens em aplicações industriais, oferecendo elevada velocidade de análise, elevada repetibilidade e redução de erros

causados por inspeção humana motivados, por exemplo, por fadiga (Martins Almeida, 2021).

2.6.1. Processamento digital de imagem

O processamento digital de imagens contribui diretamente para a base do funcionamento destes sistemas de visão artificial. O conjunto de técnicas permite uma infinidade de processos que se complementam de forma ideal na direção do objetivo final da aplicação. Desta forma, é possível obter um resultado com base em informação que permite facilitar, por exemplo, a tomada de decisões (Martins Almeida, 2021).

Por entre as etapas mais comuns características deste tipo de processamento de imagem encontram-se o pré-processamento da imagem, a segmentação, a extração dos conteúdos que caracterizam o resultado e, por fim, a devida classificação dos objetos presentes na imagem processada.

Este conjunto de etapas constitui um *pipeline* típico de processamento de imagem, amplamente utilizado em sistemas de visão industrial.

Uma das técnicas mais utilizadas para a segmentação automática e aplicada neste trabalho, é o método de Otsu. Este permite determinar de forma automática um limite que separa diferentes classes de pixéis com base no aumento da variação dos valores entre classes (Otsu, 1979).

Para além disso, operações como a erosão, dilatação, fecho (também utilizada neste trabalho) são frequentemente utilizadas para melhorar os resultados obtidos após realizada a segmentação, como forma de eliminar ruído remanescente.

A figura 7 apresenta um exemplo das diferentes etapas do processamento de imagem.

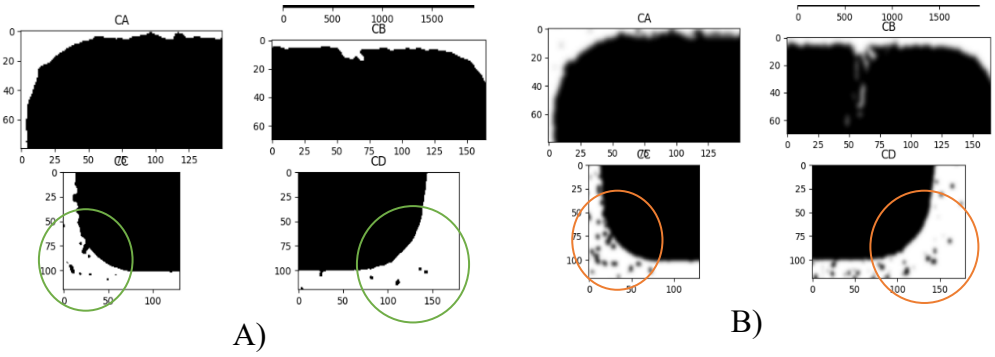


Figura 7 - Etapas do processamento de imagem. A) Imagem após tratamento; B) Imagem pré-tratamento.

2.6.2. Sistemas de visão artificial baseados em hardware de baixo custo

A investigação e desenvolvimento em torno de plataformas de computação de baixo custo permitiu, de certa forma, uma facilidade acrescida na implementação de sistemas de visão artificial tornando, em geral, as soluções mais acessíveis e robustas o suficiente para diversas aplicações industriais (Martins Almeida, 2021).

Estas soluções assumem particular relevância em contextos académicos e de prototipagem, onde o custo reduzido e a flexibilidade de implementação são fatores determinantes.

O *Raspberry Pi* (figura 8) e o Arduíno, são exemplos reais deste mesmo conceito de hardware compacto de baixo custo. O *Raspberry Pi* destaca-se, porém, devido à sua excelente capacidade de processamento capaz de executar algoritmos de processamento de imagem utilizando linguagens de programação tal como *Python*.

Esta capacidade permite a implementação de sistemas de visão artificial distribuídos, onde o processamento é realizado localmente, reduzindo a necessidade de recursos computacionais centralizados.

A utilização de bibliotecas como Skimage, OpenCV, entre outros, permitem implementar algoritmos avançados que apoiam a vertente da visão artificial com relativa

facilidade, criando, uma vez mais, um atalho no desenvolvimento de protótipos de sistemas com recurso à visão artificial.

Figura 8 - Exemplo de aplicação de um *Raspberry Pi* a nível industrial.

2.7. Comunicação entre PLC e sistemas externos

Apesar da elevada robustez e fiabilidade demonstradas pelos PLC, estes dispositivos apresentam também limitações ao nível da execução de algoritmos complexos e exigentes, como o processamento de imagem ou o tratamento avançado de dados, que normalmente não são viáveis devido às próprias restrições do dispositivo (Baltazar, 2021)

Por esse motivo, é comum integrar soluções que permitam tirar partido do melhor de dois mundos, combinando o controlo determinístico assegurado pelo PLC com sistemas computacionais externos, como o *Raspberry Pi*, responsáveis pela execução de tarefas mais exigentes do ponto de vista computacional e fora do alcance do PLC (Martins Almeida, 2021).

Esta abordagem permite uma separação funcional entre o controlo do processo e o processamento de dados, contribuindo para uma maior eficiência, flexibilidade e escalabilidade do sistema global.

A comunicação entre os dispositivos pode ser realizada através de protocolos de comunicação industriais (figura 9, retirada do website direcionado ao protocolo S7 da *Siemens*), onde se incluem:

- OPC/UA;

- Modbus;

- Protocolo S7,

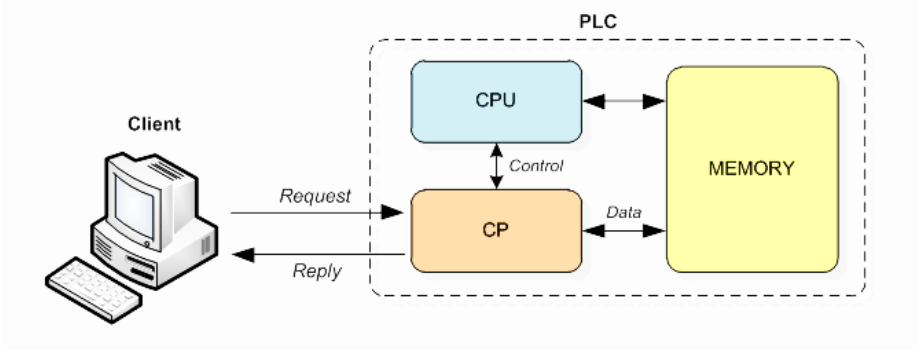


Figura 9 - Estrutura de comunicação entre um sistema externo e um PLC, evidenciando a troca de dados entre cliente, CPU e memória interna.

Estes protocolos permitem a troca de dados entre controladores industriais e sistemas externos, possibilitando a integração de diferentes tecnologias no mesmo sistema (Xu et al., 2014).

No contexto deste trabalho, a comunicação entre o PLC e o sistema externo é realizada através do protocolo S7, permitindo a leitura direta de dados a partir dos blocos de dados e a sua utilização em aplicações externas, nomeadamente para processamento e monitorização.

2.8. *Digital Twins*

O conceito de *Digital Twin* refere-se, em tradução direta, à criação de uma representação digital gémea de um sistema físico real, capaz de refletir o seu comportamento em tempo real (Iranshahi et al., 2025). Um *Digital Twin* estabelece uma ligação contínua entre o sistema físico e o modelo digital, permitindo, desta forma, a troca permanente e consistente de dados entre ambos (Tao et al., 2019).

Ao contrário de modelos de simulação tradicionais, os gémeos digitais são continuamente atualizados com dados provenientes do sistema físico, permitindo uma representação dinâmica e evolutiva do sistema ao longo do tempo(Matta & Lugaresi, 2024).

Com este tipo de soluções disponíveis, torna-se possível realizar simulações, prever falhas e otimizar processos produtivos sem necessidade de interação direta com o sistema físico (Leon-Medina et al., 2025).

Esta abordagem permite não só a monitorização do sistema, mas também a implementação de estratégias de manutenção preditiva e otimização baseada em dados, contribuindo para uma maior eficiência operacional (Benhanifia et al., 2025).

A figura 10 ilustra a interação entre o sistema físico e o modelo digital no contexto de um *Digital Twin*.

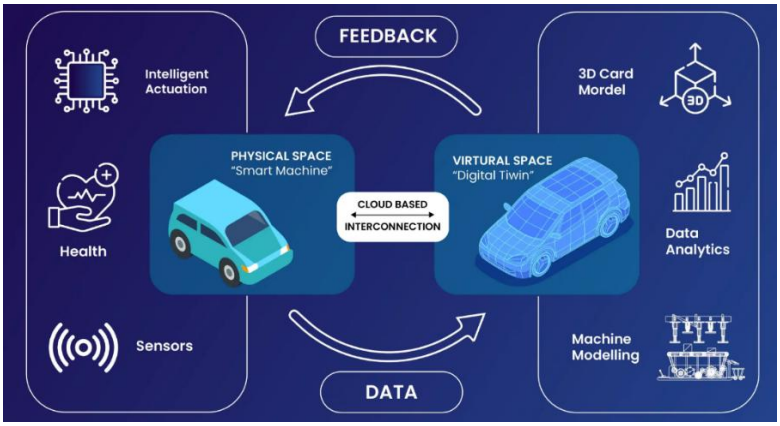


Figura 10 - Representação do conceito de Digital Twin.

No contexto da indústria 4.0, os *Digital Twins* assumem um papel fundamental na integração entre sistemas físicos e digitais, sendo frequentemente suportados por tecnologias como a *IIoT* e plataformas de computação em nuvem.

2.9. Síntese do estado da arte

A revisão literária apresentada mostra que a integração entre tecnologias de automação industrial, visão computacional e sistemas digitais constitui um elemento base para o desenvolvimento de soluções de sistemas industriais modernas (Iranshahi et al., 2025; Xu et al., 2014).

O uso de um PLC para o controlo integral do sistema físico, aliado a sistemas externos capazes de executar tarefas como a de processamento de imagem, permite desenvolver soluções flexíveis, tal como a apresentada neste trabalho (Baltazar, 2021; Martins Almeida, 2021).

Para além disso, a integração de tecnologias associadas à Indústria 4.0, como os sistemas ciber-físicos e os *Digital Twins*, abre novas possibilidades para uma monitorização mais eficiente e para a otimização contínua de sistemas industriais (Bottazzi et al., 2025; Leon-Medina et al., 2025).

A utilização destas abordagens permite ainda a recolha e análise contínua de dados, suportando a implementação de estratégias de manutenção preditiva e tomada de decisão baseada em dados.

O sistema desenvolvido no âmbito deste trabalho enquadra-se neste contexto tecnológico, integrando diferentes camadas de controlo, processamento e monitorização. A solução proposta envolve a combinação de um PLC industrial, um sistema de visão artificial baseado num *Raspberry Pi* e uma interface *web* para monitorização de dados em tempo real.

Desta forma, demonstra-se a viabilidade da integração destas tecnologias em ambiente académico, aproximando a implementação prática dos conceitos abordados ao contexto industrial real.

3. IMPLEMENTAÇÃO DA SOLUÇÃO PROPOSTA

O presente capítulo descreve a metodologia adotada no desenvolvimento do sistema proposto, bem como os materiais e métodos utilizados ao longo da sua implementação. Tendo como base a modernização de uma maquete de triagem existente, procurou-se integrar diferentes tecnologias associadas à indústria 4.0, nomeadamente a visão artificial, comunicação entre sistemas e monitorização de dados em tempo real.

Numa primeira fase, foi realizada a modelação tridimensional da bancada em ambiente CAD, permitindo não só a análise e melhoria da componente mecânica, mas também a criação de uma base para futuras integrações em ambientes de simulação e desenvolvimento de gémeos digitais. Esta etapa revelou-se fundamental para a validação de soluções físicas e para a otimização do posicionamento dos diferentes elementos do sistema.

Seguidamente, procedeu-se à implementação do sistema de controlo recorrendo a um PLC *Siemens* S7-1214C, responsável pela gestão do funcionamento da máquina, incluindo a coordenação dos atuadores, sensores e lógica de operação. Paralelamente, foi desenvolvido um sistema de visão artificial baseado num *Raspberry Pi*, encarregue da identificação e validação das peças através de processamento de imagem.

A comunicação entre os diferentes sistemas foi assegurada através do protocolo S7, permitindo a troca de informação entre o PLC e o *Raspberry Pi*, garantindo assim a integração entre o controlo do processo e o processamento de dados. Adicionalmente, foi implementada uma interface *web* simples, capaz de apresentar dados de produção em tempo real, contribuindo para a monitorização do sistema.

Por fim, são descritos os algoritmos desenvolvidos, tanto ao nível do processamento de imagem como da exportação de dados para a plataforma *web*, detalhando o seu funcionamento e integração no sistema global.

3.1. Modelação 3D da bancada

Num momento inicial foi elaborada uma representação tridimensional completa da maquete em ambiente CAD (figura 13), permitindo não só a visualização global do sistema, mas também o estudo de soluções alternativas para o posicionamento da *webcam* e restantes componentes envolvidos no processo.

Para além da modelação da estrutura existente, foram também desenvolvidas e integradas novas peças com o objetivo de melhorar a integridade, funcionalidade e praticidade da maquete. Estas peças foram desenhadas em ambiente CAD e posteriormente fabricadas por impressão 3D, sendo apresentadas com maior detalhe no Apêndice A e Apêndice B.

A modelação integral da bancada permitiu uma melhor compreensão da interação entre os diferentes elementos mecânicos, facilitando a identificação de limitações do sistema original e possibilitando a introdução de melhorias ao nível do posicionamento dos componentes, estabilidade estrutural e repetibilidade dos movimentos, aspetos fundamentais quando são implementadas soluções com recurso a visão artificial.

A figura 11 apresenta o modelo tridimensional completo da bancada, onde é possível observar a estrutura principal, os sistemas de transporte, os atuadores pneumáticos e os elementos de suporte ao sistema de controlo.

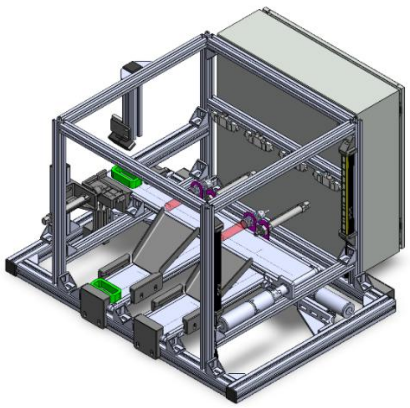


Figura 11 - Representação tridimensional da bancada em ambiente CAD.

Com a representação apresentada, foi possível obter uma visão global do sistema, bem como validar a integração dos diferentes componentes antes da sua implementação física.

Para além disso, o modelo desenvolvido constitui uma base sólida para futuras aplicações em ambiente de simulação, permitindo a criação de diferentes configurações e estados da máquina de forma controlada.

Neste sentido, a modelação 3D não só contribuiu para a melhoria da componente mecânica da maquete, como também facilitou a preparação do sistema para integrações mais avançadas, nomeadamente no contexto de um gémeo digital.

De forma a complementar esta abordagem, apresenta-se de seguida uma descrição dos principais componentes que constituem o sistema desenvolvido.

3.1.1. Componentes

De forma a complementar a modelação tridimensional apresentada, procede-se à identificação dos principais componentes que constituem a bancada desenvolvida.

O sistema é composto por uma estrutura base em perfis de alumínio, que suporta os diferentes elementos mecânicos e garante a estabilidade do conjunto. O transporte das peças é assegurado por um sistema de *conveyor*, responsável pela deslocação das mesmas ao longo das diferentes etapas do processo.

O sistema de atuação é constituído por cilindros pneumáticos SMC, utilizados para o posicionamento e desvio das peças, sendo o seu funcionamento controlado pelo PLC *Siemens* S7-1214C. A deteção de posições e estados é realizada através de sensores fotoelétricos *Omron*.

Ao nível do controlo e supervisão, o sistema integra ainda uma interface HMI *Siemens* KTP700 *Basic*, permitindo a interação com o utilizador, bem como um *Raspberry Pi* 3B responsável pelo processamento de imagem e comunicação com o PLC.

A aquisição de imagem é realizada através de uma *webcam Logitech* C920, posicionada de forma a capturar a zona de inspeção das peças.

Para além dos componentes comerciais, foram também desenvolvidos elementos mecânicos adicionais em ambiente CAD, posteriormente fabricados por impressão 3D, com o objetivo de melhorar o desempenho global do sistema.

De forma a sistematizar os elementos utilizados, o Apêndice C apresenta a tabela 2, onde se encontram identificados os principais componentes da bancada. O Apêndice D apresenta diversas páginas adicionadas ao esquema elétrico dadas as adições físicas.

3.1.2. Integração de um *Digital Twins*

É possível, para já, embora de uma forma ainda simplificada, realizar uma representação dos diferentes estados da máquina através da apresentação de imagens correspondentes a diferentes configurações do modelo CAD, numa plataforma web, conforme os avanços de estados do sistema.

Esta abordagem baseia-se na utilização do modelo tridimensional desenvolvido, permitindo representar diferentes posições dos atuadores e elementos mecânicos de acordo com o estado atual da máquina. Como exemplo, é possível observar a representação de um cilindro pneumático em duas configurações distintas: posição recolhida e posição avançada, permitindo ilustrar de forma clara a alteração do estado físico do sistema (figura 12).

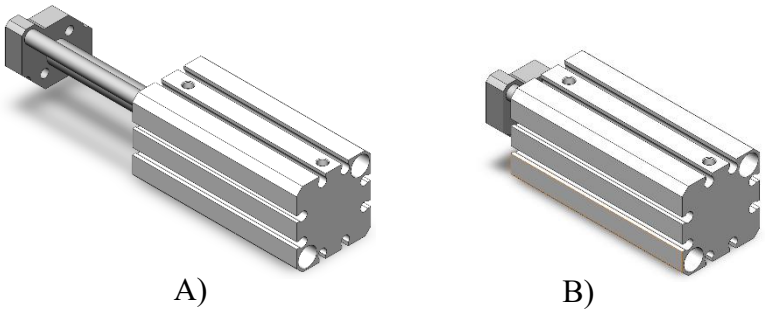


Figura 12 - Configurações aplicadas a um cilindro pneumático: A) Cilindro avançado; B) Cilindro Recuado.

A associação destas diferentes configurações a estados específicos do funcionamento da máquina permite criar uma correspondência visual entre o comportamento real e a sua representação digital, ainda que sem ligação direta em tempo real.

Neste contexto, surge a possibilidade de integrar esta funcionalidade na própria página *web* desenvolvida para a monitorização do sistema, onde, consoante o estado atual da máquina, seria apresentada uma imagem correspondente ao modelo CAD nesse mesmo estado. Desta forma, o utilizador poderia acompanhar visualmente o funcionamento do sistema através de uma representação simplificada, mas intuitiva, complementando a informação numérica já disponibilizada.

Apesar de não ter sido implementada nesta fase, esta abordagem constitui uma aproximação inicial ao conceito de *Digital Twin*, permitindo a visualização interpretativa do funcionamento do sistema.

Desta forma, estabelece-se uma base para desenvolvimentos futuros, onde será possível evoluir para uma integração mais avançada, com atualização automática dos estados do modelo digital em função dos dados provenientes do sistema físico.

3.1.3. Produto representativo

É de referir que, na configuração original da maquete, a lógica de funcionamento passava pela distinção entre dois tipos de peças: um bloco maquinado em poliamida (PA) e um outro modelo de igual geometria, contendo uma chapa metálica numa das suas faces. Esta distinção era realizada através de uma combinação binária de dois sensores, um fotoelétrico e outro indutivo.

A figura 13 apresenta uma representação em ambiente CAD das peças originalmente utilizadas no sistema, evidenciando a diferença entre o bloco simples e o bloco com inserção metálica.

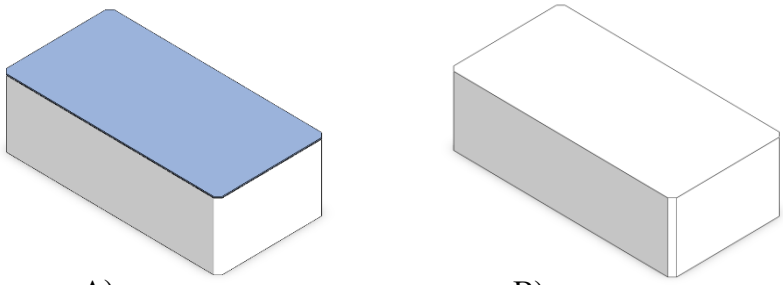


Figura 13 – Peças substituídas: A) Peça com chapa; B) Peça sem chapa.

No âmbito deste trabalho, o objetivo passou pela substituição desta abordagem por um sistema de visão artificial, recorrendo a uma câmara para identificar defeitos e distinguir diferentes tipos de peças.

Para este efeito, foram desenvolvidas novas peças, apresentadas na figura 14, com diferentes configurações geométricas que permitem a sua classificação através de processamento de imagem.

A) B) C) D)

Figura 14 – Peças utilizadas: A) Peça A – OK; B) Peça A – NOK; C) Peça B – OK; D) Peça B – NOK.

As peças desenvolvidas (impressas a vermelho) correspondem a diferentes variantes do produto, designadas por tipos A, B, C e D, sendo diferenciadas com base em características geométricas específicas, nomeadamente ao nível dos filetes internos.

Em contraste, as peças originais ilustram a abordagem inicial baseada em sensores, permitindo evidenciar a evolução do sistema para uma solução baseada em visão artificial.

A prioridade do algoritmo desenvolvido passou pela identificação dos filetes existentes nas peças conformes, sendo o mesmo capaz de distinguir, adicionalmente, o tipo de peça A, B, C ou D.

Para validação do sistema, foram produzidas três unidades de cada tipo de peça utilizando uma impressora de filamento da *Bambu Labs*, permitindo a realização de testes, tanto em modo manual como em modo automático, com o objetivo de validar o funcionamento do algoritmo e a comunicação entre os diferentes sistemas integrados.

3.2. Programação do CPU Siemens S7-1214C

A maquete desenvolvida incorpora um CPU *Siemens* S7-1214C AC/DC/RLY, responsável pelo controlo integral da estação, incluindo a gestão dos atuadores, leitura de sensores e coordenação das diferentes etapas do processo de triagem. Para além disso, o controlador permite a comunicação com sistemas externos, nomeadamente o *Raspberry Pi*, recorrendo ao protocolo S7.

De forma a garantir a comunicação entre os diferentes dispositivos do sistema, nomeadamente o PLC, a HMI e o *Raspberry Pi*, foi necessário configurar todos os elementos na mesma rede local. Neste contexto, foi atribuído ao PLC o endereço IP 192.168.30.1, permitindo a sua identificação e acesso pelos restantes dispositivos.

Para que a comunicação com o *Raspberry Pi* fosse possível, foi ainda necessário desativar a opção de otimização de acesso aos blocos de dados (DB), presente no TIA Portal. Esta configuração permite o acesso direto à memória do PLC por parte de aplicações externas, sendo essencial para a leitura e escrita de dados através do protocolo S7. A figura 15 ilustra a configuração referida.

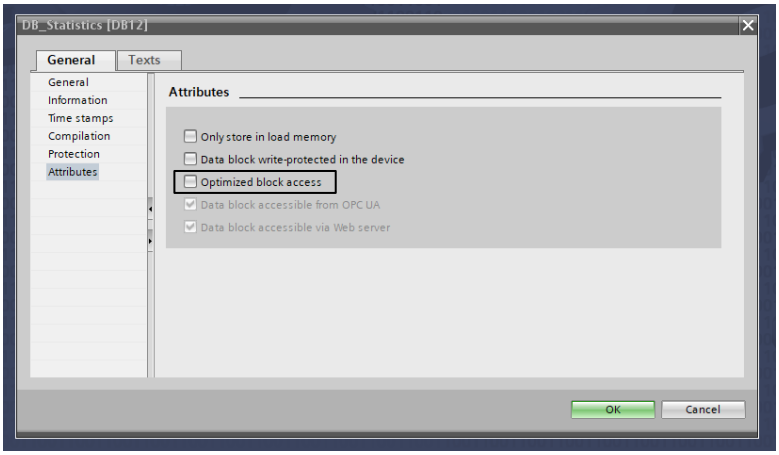


Figura 15 - Configuração da opção de acesso otimizado ao bloco.

Esta abordagem permite estabelecer uma ligação direta entre o sistema de controlo e o sistema de processamento de imagem, garantindo a troca de informação necessária ao funcionamento integrado do sistema.

Do ponto de vista da programação, os controladores da série S7-1200 seguem uma estrutura baseada em blocos, conforme definido pela norma IEC 61131-3, permitindo o desenvolvimento de aplicações modulares, reutilizáveis e de fácil manutenção.

A execução do programa do utilizador é gerida pelo sistema operativo do PLC, que organiza a chamada dos diferentes blocos de acordo com eventos e ciclos de execução. Esta estrutura hierárquica permite que um bloco principal invoque outros blocos, formando uma arquitetura em árvore que facilita a organização do código.

Neste contexto, a estrutura do programa foi organizada com base nos seguintes tipos de blocos:

- **Blocos de Organização** (OB – *Organization Blocks*): Representam a interface entre o sistema operativo e o programa do utilizador, sendo executados automaticamente pelo PLC. Destaca-se o OB1, responsável pelo ciclo principal de execução do programa (ciclo de scan), funcionando como ponto de entrada para a lógica implementada.
- **Blocos de Função** (FC – *Functions*): Utilizados para a implementação de rotinas reutilizáveis sem memória associada. Estes blocos não retêm valores entre ciclos, sendo adequados para operações lógicas e cálculos simples.
- **Blocos de Função com Memória** (FB – *Function Blocks*): Permitem a implementação de lógica com retenção de estado, recorrendo a blocos de dados de instância (DB). São particularmente adequados para a

modelação de comportamentos sequenciais, como máquinas de estados e controlo de dispositivos.

- **Blocos de Dados** (DB – *Data Blocks*): Utilizados para armazenamento de dados, podendo ser globais ou associados a instâncias de FBs. Estes blocos permitem a partilha de informação entre diferentes partes do programa, bem como a integração com sistemas externos e interfaces HMI.

Esta organização modular foi adotada no desenvolvimento do sistema, permitindo separar as diferentes funcionalidades da maquete em blocos distintos, facilitando a leitura, manutenção e expansão futura do programa.

A implementação da lógica de controlo foi realizada recorrendo à linguagem *Structured Control Language* (SCL), suportada pelos controladores Siemens S7-1200. A escolha desta linguagem deve-se à sua adequação à implementação de algoritmos com maior complexidade lógica, permitindo uma representação mais estruturada e direta de condições, sequências e interações entre variáveis. A utilização de estruturas como *IF* e *CASE* possibilita uma organização clara do código, reduzindo a complexidade visual associada a linguagens gráficas como o *Ladder*, especialmente em sistemas com múltiplos estados e temporizações.

Adicionalmente, a natureza textual do SCL facilita a leitura, manutenção e diagnóstico do código, permitindo uma identificação mais rápida de erros e uma maior flexibilidade na implementação de alterações. Do ponto de vista do desempenho, a utilização desta linguagem não compromete o funcionamento do sistema, uma vez que o código é compilado para uma forma equivalente no controlador, garantindo o comportamento determinístico exigido. Desta forma, o SCL revelou-se uma solução

adequada ao tipo de aplicação desenvolvida, promovendo uma implementação mais eficiente e organizada da lógica de controlo.

De forma a organizar a programação dos diversos processos implementados na maquete, descrevem-se de seguida as principais funções de bloco utilizadas para controlar as ações requeridas pelo sistema, bem como a sua integração na organização geral do programa.

3.2.1. *Main* (OB1)

O bloco de organização OB1 constitui o ciclo principal de execução do programa, sendo responsável pela integração e sequenciação das diferentes rotinas implementadas no sistema.

De acordo com o seu modo de funcionamento, a OB1 é executada de forma cíclica, permitindo a atualização contínua do estado do sistema. Neste contexto, são priorizadas as ações relacionadas com os modos de operação, pedidos de arranque e paragem, deteção de erros de processo e verificação das condições necessárias à execução de cada etapa.

A organização da lógica neste bloco permite garantir um comportamento determinístico do sistema, assegurando que, em cada ciclo de execução, são avaliadas todas as condições relevantes antes da tomada de decisão.

Neste bloco são efetuadas as chamadas às diferentes funções de bloco (FB e FC), responsáveis pela implementação das várias funcionalidades do sistema, incluindo gestão de modos, sequência de operação, controlo de atuadores e tratamento de dados.

A integração de cada uma destas funções na OB1 será descrita nos subcapítulos seguintes, com o objetivo de facilitar a compreensão da arquitetura e da lógica global do sistema desenvolvido.

3.2.2. *Manager Mode* (FB1)

A função "*FB_ModeManager*" representa uma série de condições de funcionamento, estabelecendo o controlo do modo de operação de toda a máquina. Esta função serve como cérebro da gestão de modos da máquina, gerindo também os pedidos realizados pelos potenciais operadores da máquina.

De forma a priorizar eventos humanos, foram criados de início dois *triggers* de forma a garantir que os mesmos são lidos pelo sistema Siemens S7-1200. Existe tanto a possibilidade de ativação dos comandos ser feita através dos botões físicos "*Start_PB*" ou "*Stop_PB*", bem como no ambiente de modo automático da HMI.

A figura 16 apresenta a lógica programada para esses mesmos *triggers* de pedidos do operador.

```
1  // Triggers Gerais
2
3  #rStart(CLK := #StartCmd);
4  #rStop(CLK := #StopCmd);
5  #rRPI(CLK := #RPICmd);
6
7  // Start
8
9  IF #rStart.Q THEN
10     #RunLatched := TRUE;
11     #StopLatched := FALSE;
12 END_IF;
13
14 // Pedido Stop
15
16 IF #rStop.Q THEN
17     #StopLatched := TRUE;
18 END_IF;
19
```

Figura 16 - Triggers de comando e ativação dos bits de ação.

Foi também implementado um sistema para o modo automático bem como um modo manual caso seja necessária a intervenção na máquina. Os mesmos podem ser configurados através da janela de modo manual da HMI. Esta programação encontra-se descrita na figura 17.

```
21  // Modos Automático e Manual
22
23 IF #HMI_Auto THEN
24      #AutoMode := TRUE;
25      #ManualMode := FALSE;
26  ELSIF #HMI_Manual THEN
27      #AutoMode := FALSE;
28      #ManualMode := TRUE;
29  END_IF;
30
```

Figura 17 - Seleção do modo manual e do modo automático.

De forma a serem limpos os *bits* presos, inseriu-se o código presente na figura 18.

```
32  // Limpeza dos Bits
33
34 IF #StopDone THEN
35      #RunLatched := FALSE;
36      #StopLatched := FALSE;
37  END_IF;
38
39 IF #RPIDone THEN
40      #RPILatched := FALSE;
41  END_IF;
42
```

Figura 18 - Limpeza dos bits presos.

Aqui é apresentado, também, o comando de reposição da posição inicial (RPI), utilizado em caso de erro na máquina, facilitando o restabelecimento das condições que permitem o correto arranque e funcionamento da máquina.

A figura 19 apresenta todas as ligações a entradas e saídas associadas à função de gestão dos modos de operação.

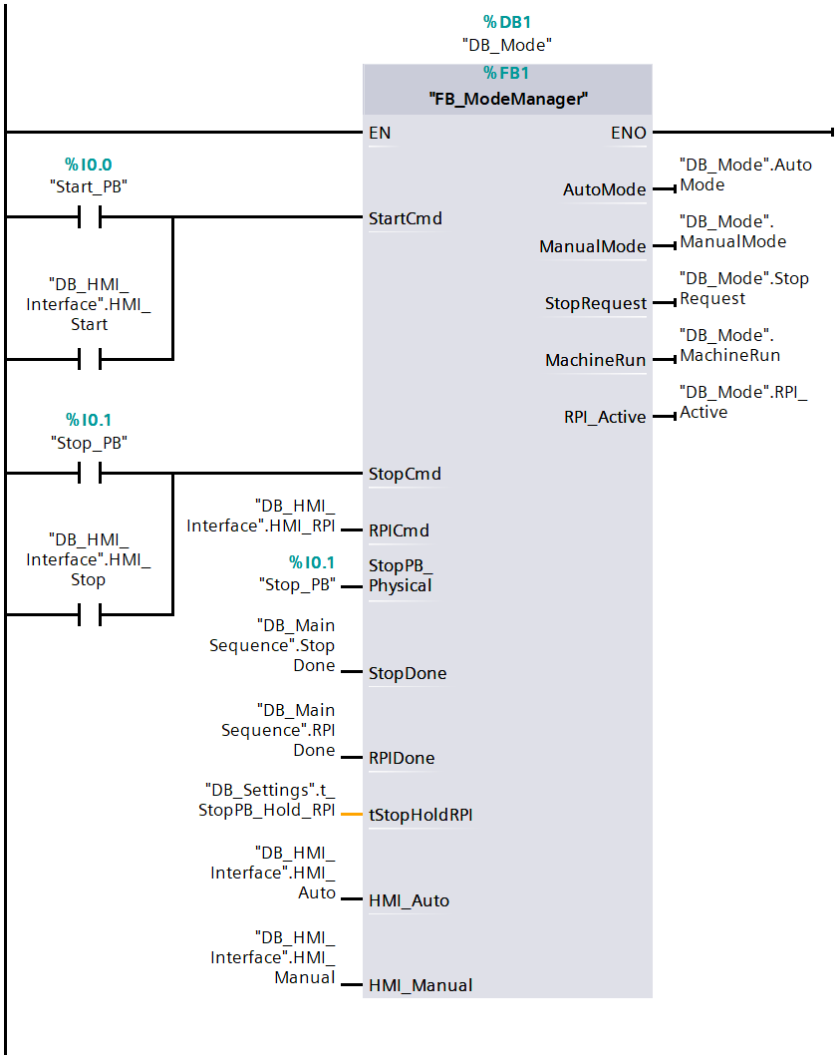


Figura 19 - Bloco de funções do modo de gestão e as suas correspondências.

3.2.3. *Main Sequence* (FB7)

A função "*FB_MainSequence*" é responsável pela gestão de todas as etapas a percorrer ao longo do ciclo de funcionamento da máquina. Na figura 20 é possível observar todo o *Grafcet* que gere este sistema de controlo.

Foi necessário implementar diferentes posições finais para que cada peça se deva encontrar em posição distinta das demais após o processo de triagem. Tendo a maquete apenas duas rampas de fuga do material, optou-se por criar o sistema de organização descrito pelas figuras 21, 22 e 23.

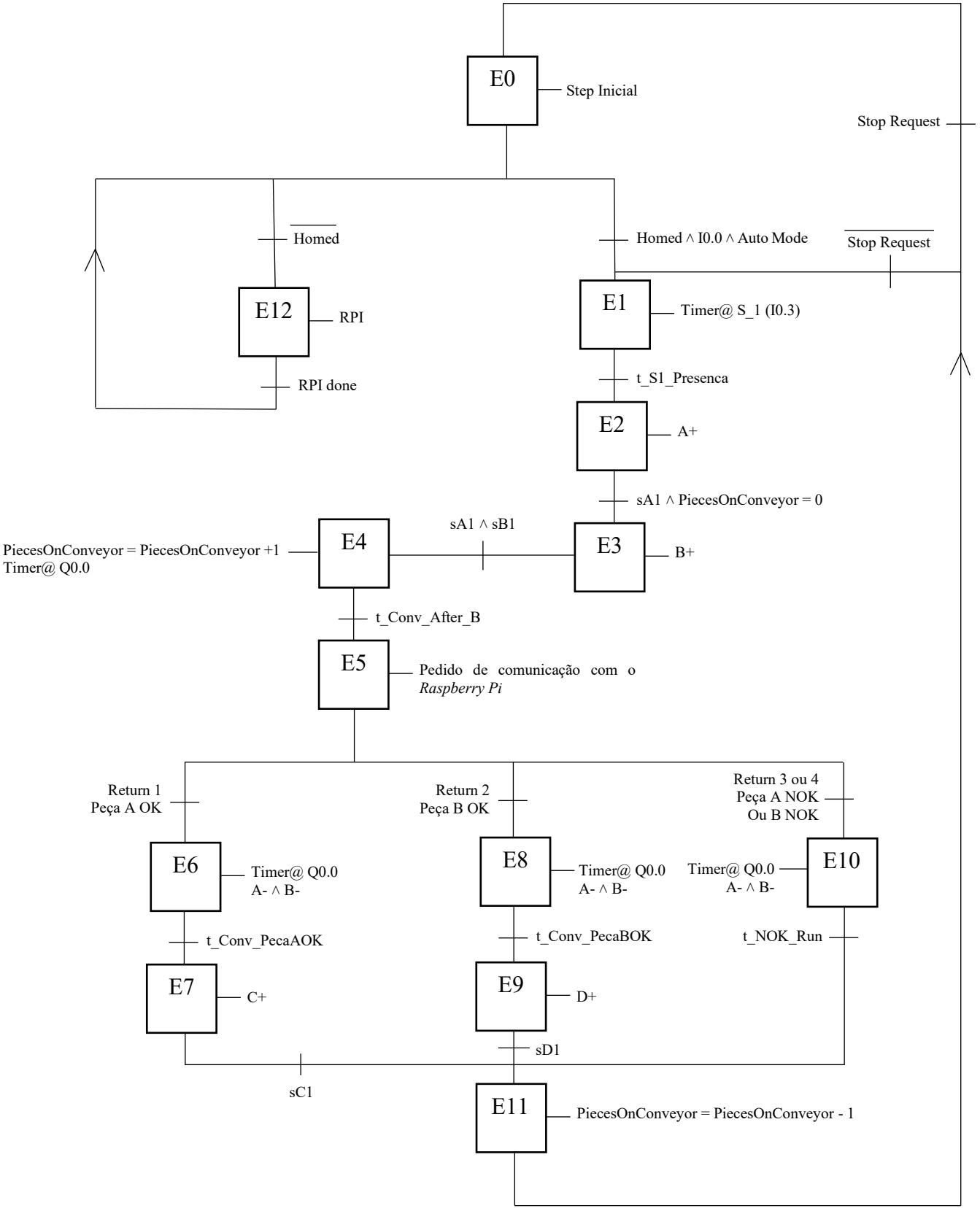


Figura 20 - Grafcet do bloco de funções da sequência principal.

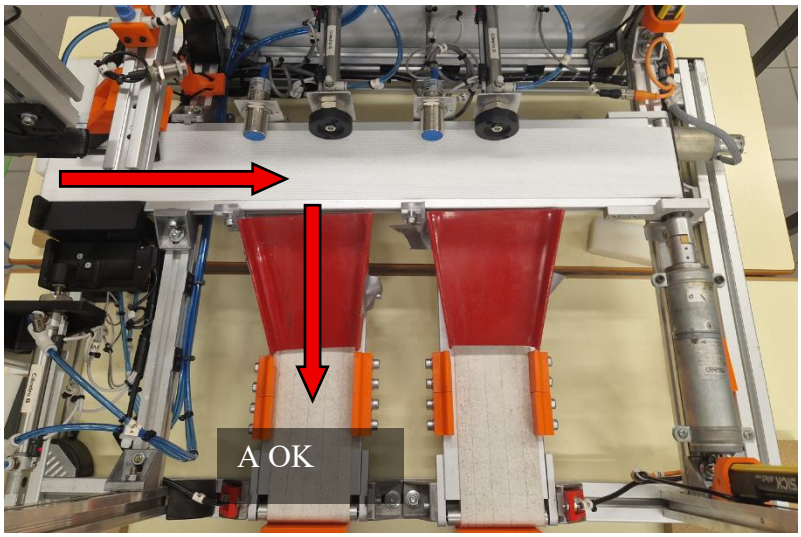


Figura 21 - Rota executada pelas peças A avaliadas como conformes.

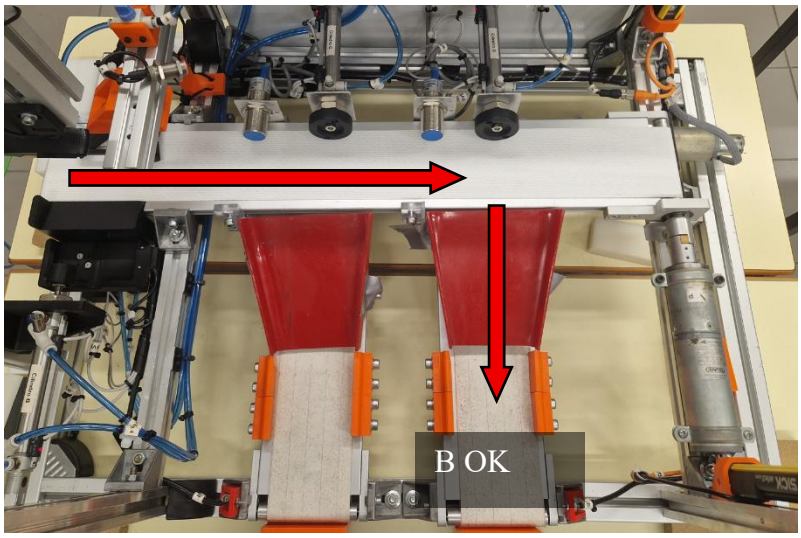


Figura 22 - Rota executada pelas peças B avaliadas como conformes.

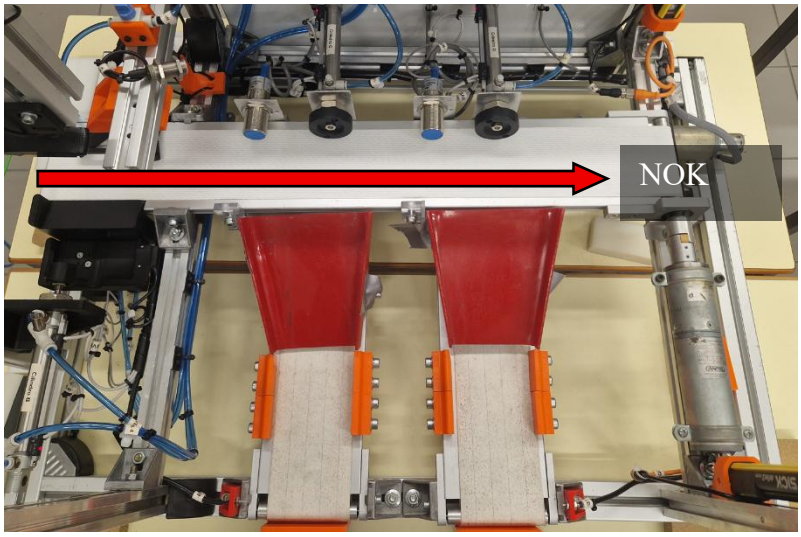


Figura 23 - Rota executada pelas peças avaliadas como não conformes.

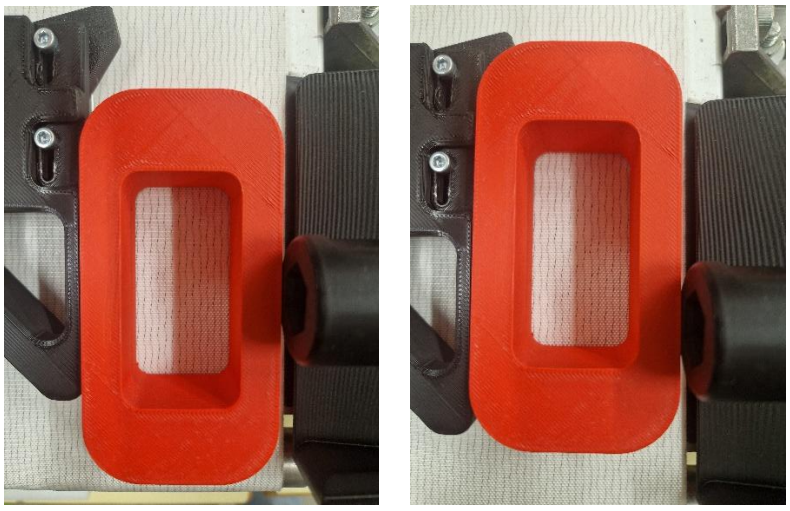
Com base neste método, a seleção entre as três opções para a recolha de peças:

- A OK: a peça é enviada pelo *conveyor* principal até ao cilindro pneumático C onde, de seguida, é empurrado para a zona de peças A conformes;

- B OK: a peça é enviada pelo *conveyor* principal até ao cilindro pneumático D onde, de seguida, é empurrado para a zona de peças B conformes;

- NOKs: as peças são processadas quanto a eventos e contadores de uma forma independente, sendo que o timer é comum para ambas as peças B não conforme e A não conforme, de forma que as mesmas sejam expulsas pelo fim do *conveyor*.

Foi implementado um temporizador "t_Conv_After_B" de forma a garantir um posicionamento mais consistente ao longo de várias medições e testes. A figura 24 apresenta a posição inicial da peça assim que empurrada pelo cilindro pneumático B até ao *conveyor*. Reparou-se que a posição de chegada era inconsistente, então, desta forma, garantiu-se um posicionamento capaz de elevar a fiabilidade do equipamento.



A) B)

Figura 24 – Posição da peça: A) Posição da peça depois da descarga do cilindro B; B) Posição da peça após concluído o temporizador "t_Conv_After_B".

Em seguida, na figura 25, são apresentadas as ligações feitas à função "*FB_MainSequence*" na OB1.

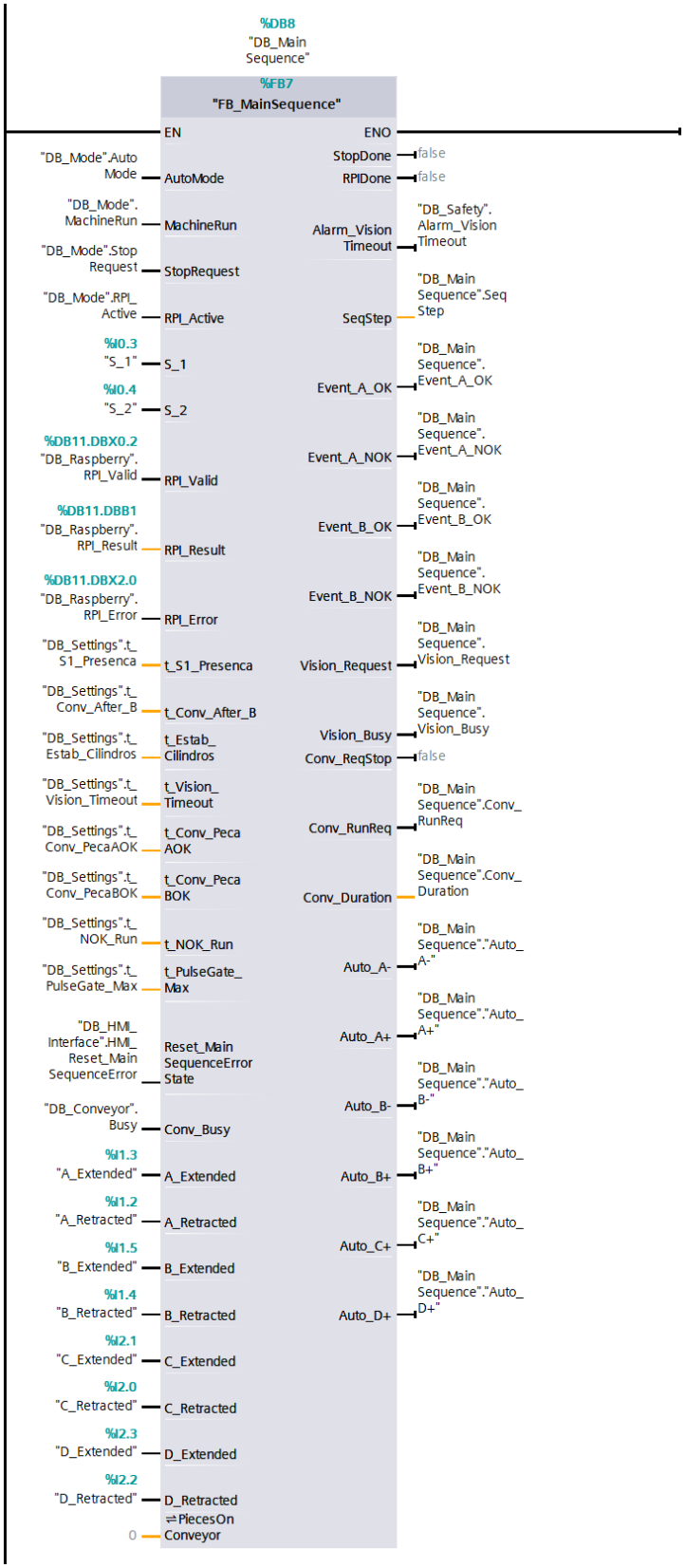


Figura 25 - Bloco de funções da sequência principal e as suas correspondências.

3.2.4. *Conveyor* (FB3)

Esta função ("*FB_Conveyor*") é uma função de rotina, chamada de forma recorrente em auxílio ao *grafcet* da sequência principal. Uma vez que há diferentes temporizadores de ativação do motor, diferentes ocasiões em que o mesmo é chamado a entrar em funcionamento, foi criada esta rotina, por ser capaz de atualizar o seu valor de tempo ativo com o objetivo de cumprir com o pedido enviado pela função "*FB_MainSequence*".

A figura 26 apresenta a escrita lógica para o funcionamento do *conveyor* conforme os pedidos realizados pela sequência principal.

```
1  #rReq(CLK := #RunReq);
2
3  IF #rReq.Q AND NOT #Active THEN
4      #Active := TRUE;
5
6      #LatchedRunTime := #RunTime;
7
8
9  END_IF;
10
11 IF #Active THEN
12     #Motor_Conv := TRUE;
13     #tRun(IN := TRUE,
14           PT := #LatchedRunTime);
15
16     IF #tRun.Q THEN
17         #tRun(IN := FALSE,
18               PT := #LatchedRunTime);
19         #Motor_Conv := FALSE;
20         #Active := FALSE;
21     END_IF;
22 ELSE
23     #Motor_Conv := FALSE;
24     #tRun(IN := FALSE,
25           PT := #LatchedRunTime);
26 END_IF;
27
28 #Busy := #Active;
```

Figura 26 - Programação SCL da função de rotina do conveyor.

A figura 27 apresenta-nos as correspondências feitas para a função do *conveyor* no OB1.

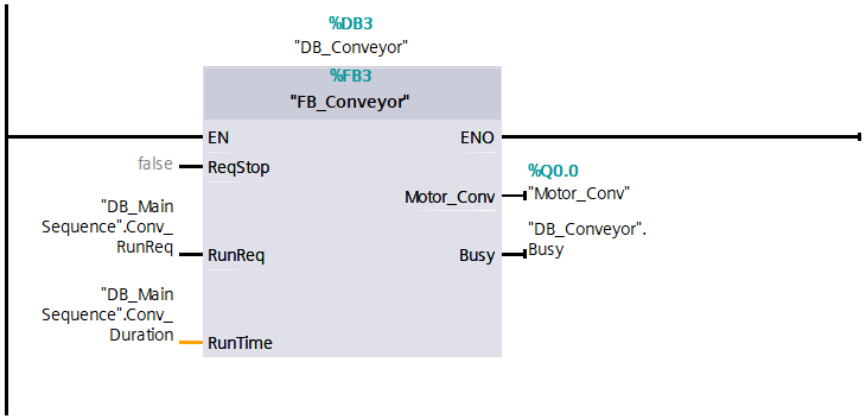


Figura 27 - Bloco de funções da rotina conveyor e as suas correspondências.

3.2.5. *Actuators* (FB6)

A função "*FB_Actuators*" tem dois modos de funcionamento. Os mesmos definem-se pelo estado em que a máquina se encontra (automático ou manual), sendo esse o modo operacional autorizado a comandar os cilindros pneumáticos. Existindo a opção de haver um ciclo automático e um ciclo manual para a atuação pneumática, de forma a garantir o correto comportamento entre as ações realizadas por ambos os intervenientes, foi necessária a implementação de uma função genérica como fonte de segurança.

Através da figura 28, somos capazes de observar como foram feitas as ligações entre a função de atuadores e os demais operadores.

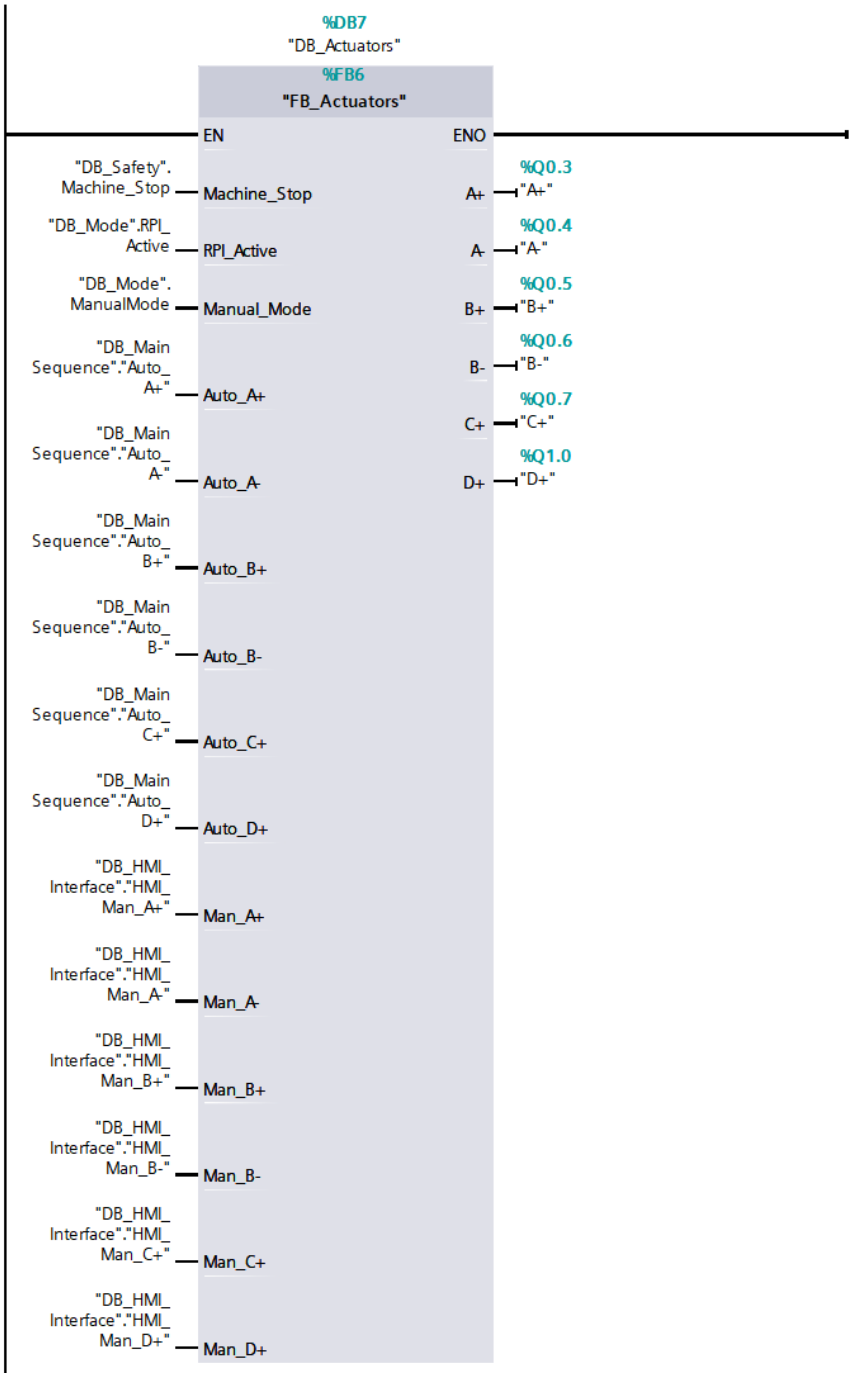


Figura 28 - Bloco de funções dos atuadores e as suas correspondências.

3.2.6. *Counters* (FB5)

Esta função é responsável pela atualização de dados a serem, mais tarde, apresentados numa plataforma *web*, bem como no ecrã da HMI. A contagem das peças do tipo A, quando é recebido o sinal sC1 (sensor de avançado do cilindro C), é

incrementada uma unidade. O mesmo processo repete-se para as peças do tipo B, contudo, no cilindro pneumático D, quando é ativado o sinal de sD1. Para ambas as peças NOK, presume-se que, após o tempo de extração das mesmas (t_NOK_Run) o valor é incrementado dependendo do resultado. O aumento de peças dependerá sempre do resultado proveniente do algoritmo corrido pelo *Raspberry Pi*.

Na figura 29, é possível ver a lógica utilizada para atingir esse objetivo. Quando, na função da sequência principal, são ativados *triggers* com eventos associados, a função de contadores recolhe a informação e acrescenta o valor internamente. A sua DB é responsável por guardar os valores a serem enviados, depois, para o *website* gerado.

```
#rAOK(CLK := #Event_A_OK);
#rANOK(CLK := #Event_A_NOK);
#rBOK(CLK := #Event_B_OK);
#rBNOK(CLK := #Event_B_NOK);

IF #ResetCounters THEN
    #Total_OK := 0;
    #Total_NOK := 0;
    #Total_All := 0;
    #A_OK := 0;
    #A_NOK := 0;
    #B_OK := 0;
    #B_NOK := 0;
ELSE
    IF #rAOK.Q THEN
        #A_OK := #A_OK + 1;
        #Total_OK := #Total_OK + 1;
    END_IF;

    IF #rBOK.Q THEN
        #B_OK := #B_OK + 1;
        #Total_OK := #Total_OK + 1;
    END_IF;

    IF #rANOK.Q THEN
        #A_NOK := #A_NOK + 1;
        #Total_NOK := #Total_NOK + 1;
    END_IF;

    IF #rBNOK.Q THEN
        #B_NOK := #B_NOK + 1;
        #Total_NOK := #Total_NOK + 1;
    END_IF;

    #Total_All := #Total_OK + #Total_NOK;
END_IF;
```

Figura 29 - Algoritmo de programação utilizado para contagem de peças.

A figura 30 apresenta como foram alocados esses *in/outs* e apresenta, também um *input* que possibilita a limpeza dos valores atuais dos diversos contadores.

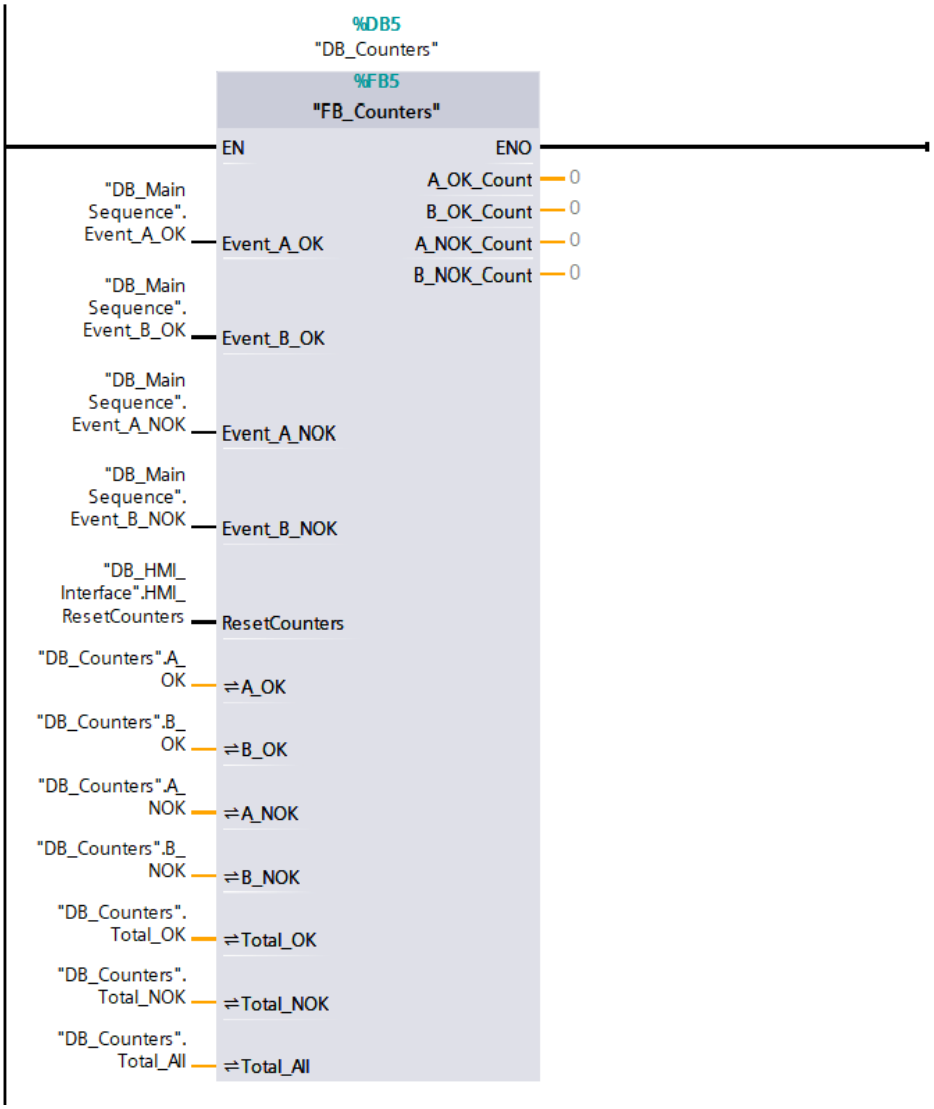


Figura 30 - Bloco de funções dos contadores e as suas correspondências.

3.2.7. *Publisher* (FB2)

A função *Publisher* é responsável por trocar informação com o *Raspberry*, enviando-lhe, sempre que necessário, um pedido de captura e processamento da imagem de forma a ser devolvido o resultado do teste executado. Através de um bit flanco gerado

na sequência principal, a função publisher faz a mensagem chegar ao PLC através do protocolo *S7*.

Para esta função foram utilizadas duas variáveis de entrada representativas da *Main Sequence* com *outputs* diretos para a base de dados criada com o fim de comunicar com esses mesmos pedidos ao *Raspberry*.

A figura 31 apresenta as ligações estabelecidas, na OB1, que permitem o funcionamento desta função.

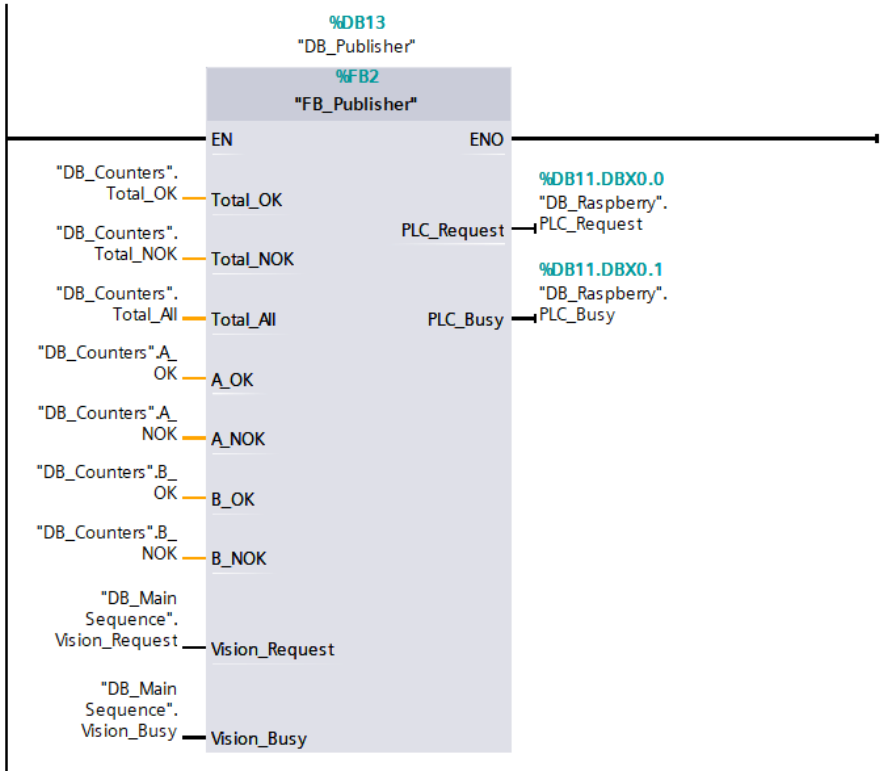
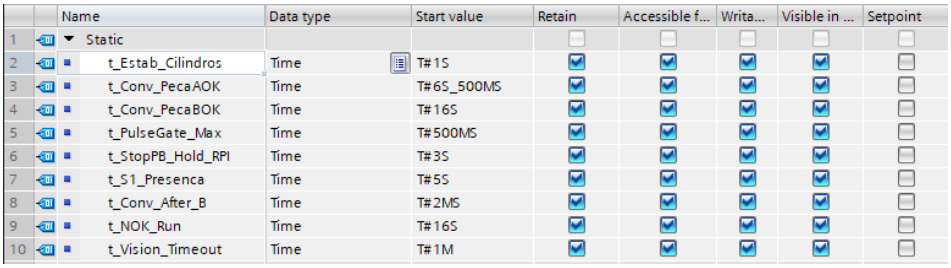


Figura 31 - Bloco de função publisher e as suas correspondências.

3.2.8. *Settings* (DB6)

Este bloco de dados é responsável por integrar os valores dos diversos temporizadores acessíveis através da HMI. Os parâmetros são configuráveis através do menu apresentado pela figura 41. É possível observar este DB na figura 32.



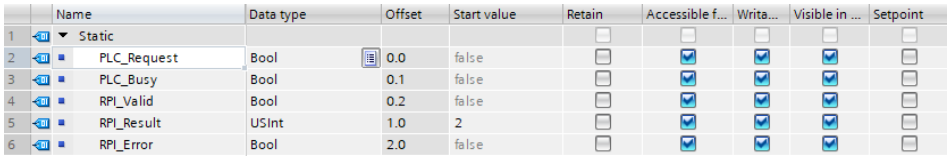
	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static							
2	t_Estab_Cilindros	Time	T#1S	☑	☑	☑	☑	☐
3	t_Conv_PecaAOK	Time	T#6S_500MS	☑	☑	☑	☑	☐
4	t_Conv_PecaBOK	Time	T#16S	☑	☑	☑	☑	☐
5	t_PulseGate_Max	Time	T#500MS	☑	☑	☑	☑	☐
6	t_StopPB_Hold_RPI	Time	T#3S	☑	☑	☑	☑	☐
7	t_S1_Presenca	Time	T#5S	☑	☑	☑	☑	☐
8	t_Conv_After_B	Time	T#2MS	☑	☑	☑	☑	☐
9	t_NOK_Run	Time	T#16S	☑	☑	☑	☑	☐
10	t_Vision_Timeout	Time	T#1M	☑	☑	☑	☑	☐

Figura 32 - Bloco de dados utilizado para os temporizadores do processo.

Repare-se que foram selecionados todos os campos na secção de "retenção" pois, desta forma é possível garantir que, quando alterados os parâmetros na HMI, após um encerramento do PLC, os dados permanecem guardados. Desta forma é evitada a necessidade de reconfiguração aquando do arranque do sistema.

3.2.9. *Raspberry* (DB11)

Este *data block* é a ponte criada entre o *Raspberry* e o PLC *Siemens*. O *Raspberry* acede diretamente a esta DB e, assim que obtido o resultado da aplicação do algoritmo de processamento de imagem, o mesmo escreve-o diretamente nesta DB. A figura 33 apresenta as diferentes variáveis necessárias para este tipo de interação.

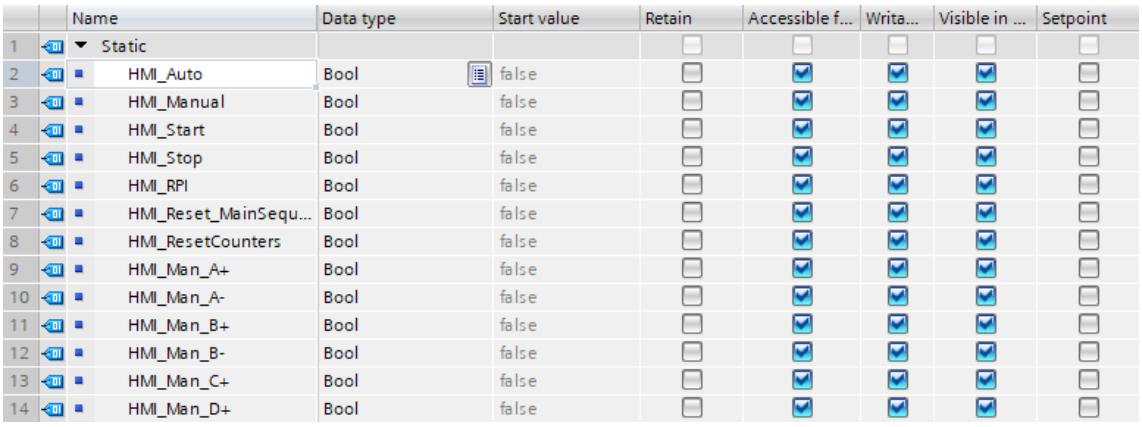


	Name	Data type	Offset	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static								
2	PLC_Request	Bool	0.0	false	☐	☑	☑	☑	☐
3	PLC_Busy	Bool	0.1	false	☐	☑	☑	☑	☐
4	RPI_Valid	Bool	0.2	false	☐	☑	☑	☑	☐
5	RPI_Result	USInt	1.0	2	☐	☑	☑	☑	☐
6	RPI_Error	Bool	2.0	false	☐	☑	☑	☑	☐

Figura 33 - Bloco de dados utilizado para a comunicação com o Raspberry Pi.

3.2.10. HMI (DB9)

Este bloco de dados é responsável pelos diferentes comandos e controlos para a HMI. No próximo subcapítulo explica-se a aplicação das restantes variáveis aqui apresentadas.



	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static							
2	HMI_Auto	Bool	false					
3	HMI_Manual	Bool	false					
4	HMI_Start	Bool	false					
5	HMI_Stop	Bool	false					
6	HMI_RPI	Bool	false					
7	HMI_Reset_MainSequ...	Bool	false					
8	HMI_ResetCounters	Bool	false					
9	HMI_Man_A+	Bool	false					
10	HMI_Man_A-	Bool	false					
11	HMI_Man_B+	Bool	false					
12	HMI_Man_B-	Bool	false					
13	HMI_Man_C+	Bool	false					
14	HMI_Man_D+	Bool	false					

Figura 34 - Bloco de dados utilizada para comandos executados pela HMI.

3.3. Programação da HMI

A programação da HMI está estruturada em diversos ecrãs. Neste subcapítulo descrevem-se os diferentes ecrãs acessíveis ao utilizador.

A HMI utilizada é o modulo KTP700 Basic do fabricante *Siemens*. Trata-se de uma HMI que comunica através do protocolo *profinet*, pelo que foi necessária a atribuição de um IP e, em conjugação do valor utilizado para o próprio PLC, utilizou-se o valor de IP 192.168.30.2.

3.3.1. *Home Screen*

O *home screen* é o ecrã/ambiente que permite a navegação entre os diversos menus. Será, porém, sempre necessário regressar ao menu *home screen* no caso de ser pretendido entrar num segundo ecrã.

A figura 35 apresenta o ecrã *home screen* com todos os botões de acesso aos respetivos ecrãs.

Figura 35 - Ecrã Home Screen da HMI.

3.3.2. Auto

Este ambiente é caracterizado pelos botões de "*Start*" e pedido de paragem ("*Stop*"), bem como um botão de *reset* que força o sistema a uma reposição da posição inicial (RPI). É possível, ainda, obter *feedback* do resultado gerado pelo processamento de imagem através deste menu.

A figura 36 apresenta o ecrã de modo automático.

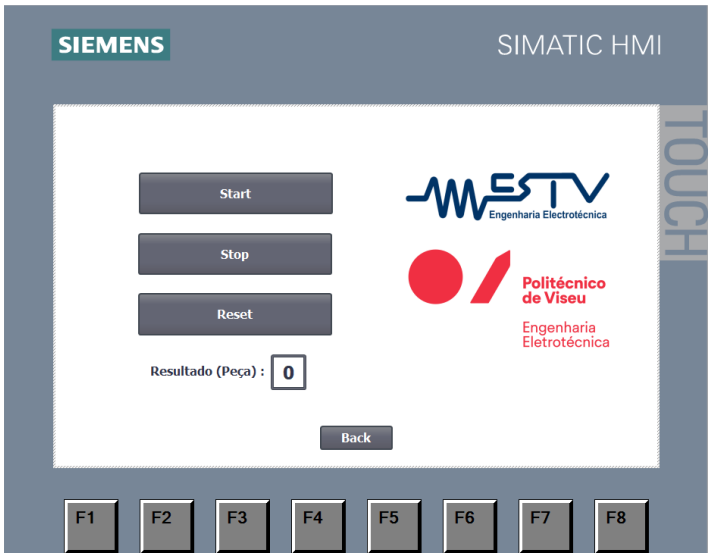


Figura 36 - Ecrã do modo automático da HMI.

3.3.3. Manual

Este é um ambiente que permite as diversas ações manuais de forma a serem testados os diversos componentes de forma independente, sem necessidade de validar os seus movimentos através de um modo automático.

A figura 37 apresenta o ecrã do modo manual.

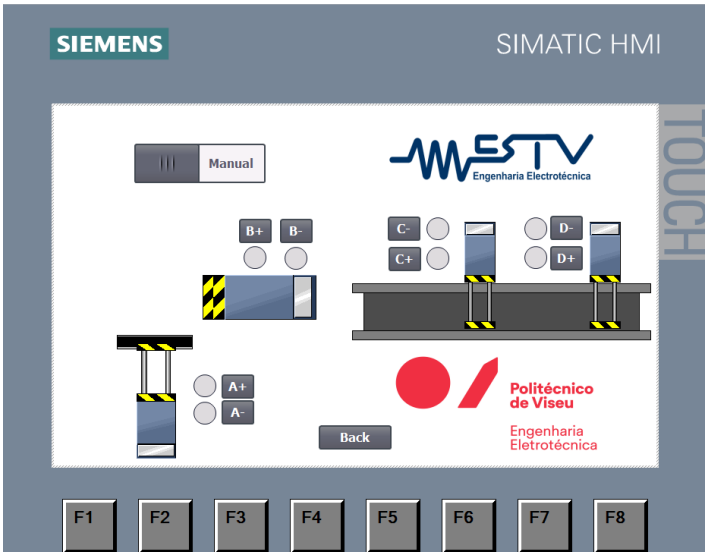


Figura 37 - Ecrã do modo manual da HMI.

Através da figura 38 é possível observar que, quando ativo um dado sinal, é enviado um *feedback* para este ecrã, dando mais informação em casos, por exemplo, de afinação dos mesmos.

É possível obter este tipo de *feedback* transmitido através da HMI, partindo da seleção do menu propriedades e, de seguida, animações (do objeto a tratar), onde somos apresentados com a possibilidade de criar diferentes interações com os elementos visuais. Dá-se como exemplo o sensor de avançado do cilindro A.

Figura 38 - Fotografia do menu manual da HMI.

A figura 39 apresenta a forma de configuração dos elementos que permitem executar as animações descritas anteriormente.

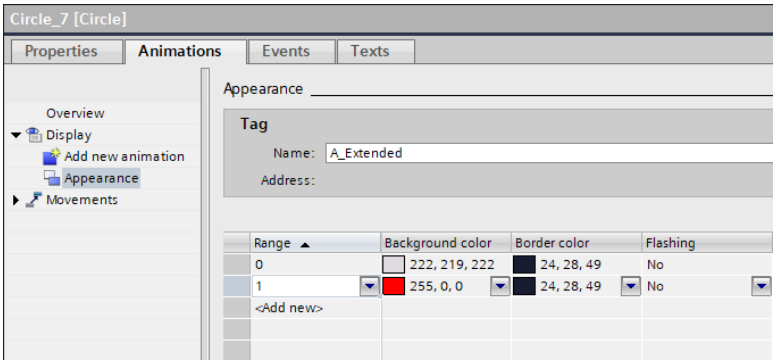


Figura 39 - Menu de customização de elementos da HMI.

3.3.4. Contadores

Neste ecrã apresenta-se uma estatística dos diferentes valores de produção e a possibilidade de limpar os mesmos, caso se pretenda (figura 40).

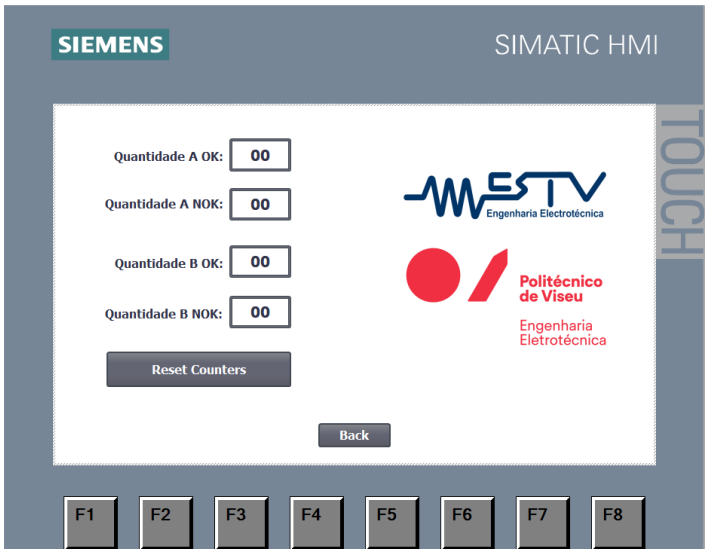


Figura 40 – Ecrã do menu contadores da HMI.

3.3.5. Definições

Este ecrã (figura 41) permite a edição dos tempos pré-definidos para todos os temporizadores utilizados no processo. Desta forma, a parametrização torna-se mais simples, o que facilita a configuração em caso de necessidade ao nível da manutenção do sistema.

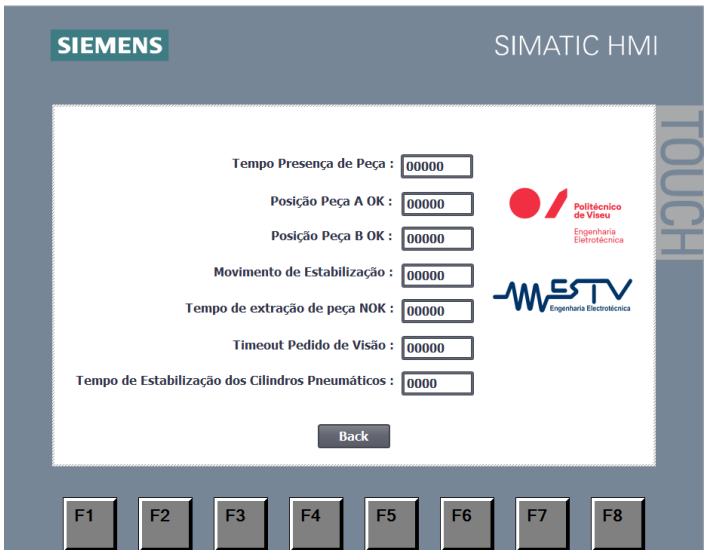
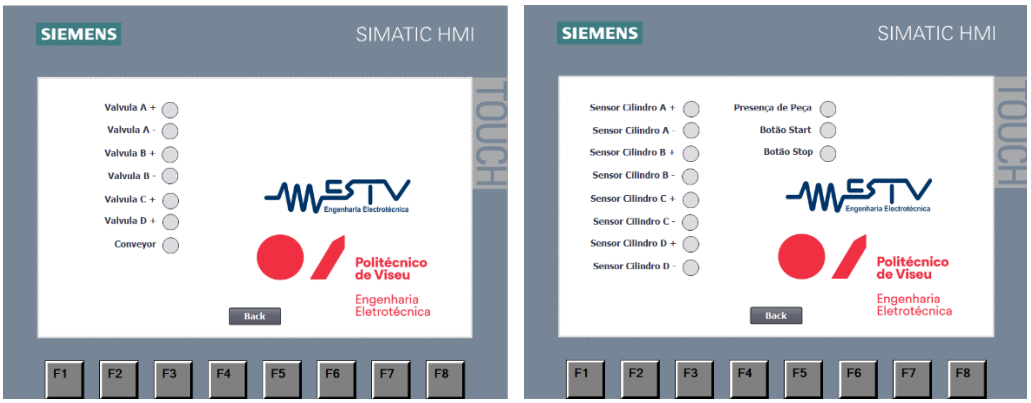


Figura 41 - Ecrã do menu de definições da HMI.

3.3.6. Entradas/Saídas

Apesar de se tratar de dois ecrãs independentes (figura 42), são ambientes semelhantes, onde estão expostos todos os seus componentes, com uma abordagem de funcionamento idêntica à apresentada anteriormente pela figura 38.



A) B)

Figura 42 – Menus de *in/outs*: A) Ecrã das entradas; B) Ecrã das saídas.

3.3.7. *Tags*

Estas *tags* são variáveis de ligação da HMI com variáveis existentes na memória do PLC. Estas podem ser lidas, ou levadas a interagir com o ambiente físico a partir da HMI. A figura 43 apresenta uma listagem de *tags* utilizadas pela HMI.

Name	Tag table	Data type	Connection	PLC name	PLC tag	Address
A+	Default tag table	Bool	HMI_Conne...	PLC_1	"A+"	
A-	Default tag table	Bool	HMI_Connectio...	PLC_1	"A-"	
B+	Default tag table	Bool	HMI_Connectio...	PLC_1	"B+"	
B-	Default tag table	Bool	HMI_Connectio...	PLC_1	"B-"	
C+	Default tag table	Bool	HMI_Connectio...	PLC_1	"C+"	
D+	Default tag table	Bool	HMI_Connectio...	PLC_1	"D+"	
A_Extended	Default tag table	Bool	HMI_Connectio...	PLC_1	A_Extended	
A_Retracted	Default tag table	Bool	HMI_Connectio...	PLC_1	A_Retracted	
B_Extended	Default tag table	Bool	HMI_Connectio...	PLC_1	B_Extended	
B_Retracted	Default tag table	Bool	HMI_Connectio...	PLC_1	B_Retracted	
C_Extended	Default tag table	Bool	HMI_Connectio...	PLC_1	C_Extended	
C_Retracted	Default tag table	Bool	HMI_Connectio...	PLC_1	C_Retracted	
DB_Counters_A_NOK	Default tag table	DInt	HMI_Connectio...	PLC_1	DB_Counters.A_NOK	
DB_Counters_A_NOK_Count	Default tag table	Int	HMI_Connectio...	PLC_1	DB_Counters.A_NOK_Co...	
DB_Counters_A_OK	Default tag table	DInt	HMI_Connectio...	PLC_1	DB_Counters.A_OK	
DB_Counters_A_OK_Count	Default tag table	Int	HMI_Connectio...	PLC_1	DB_Counters.A_OK_Count	
DB_Counters_B_NOK	Default tag table	DInt	HMI_Connectio...	PLC_1	DB_Counters.B_NOK	
DB_Counters_B_NOK_Count	Default tag table	Int	HMI_Connectio...	PLC_1	DB_Counters.B_NOK_Co...	
DB_Counters_B_OK	Default tag table	DInt	HMI_Connectio...	PLC_1	DB_Counters.B_OK	
DB_Counters_B_OK_Count	Default tag table	Int	HMI_Connectio...	PLC_1	DB_Counters.B_OK_Count	
DB_HMI_Commands_HMI_Aut...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_A...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Man...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_...	
DB_HMI_Commands_HMI_Reset	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_RPI	
DB_HMI_Commands_HMI_Res...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_R...	
DB_HMI_Commands_HMI_Res...	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_R...	
DB_HMI_Commands_HMI_Start	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_S...	
DB_HMI_Commands_HMI_Stop	Default tag table	Bool	HMI_Connectio...	PLC_1	DB_HMI_Interface.HMI_S...	
DB_Raspberry_RPI_Result	Default tag table	USInt	HMI_Connectio...	PLC_1	DB_Raspberry.RPI_Result	
DB_Settings_t_Conv_After_B	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_Conv_Afte...	
DB_Settings_t_Conv_PecaAOK	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_Conv_Pec...	
DB_Settings_t_Conv_PecaBOK	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_Conv_Pec...	
DB_Settings_t_Estab_Cilindros	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_Estab_Cili...	
DB_Settings_t_NOK_Run	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_NOK_Run	
DB_Settings_t_PulseGate_Max	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_PulseGate...	
DB_Settings_t_S1_Presenca	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_S1_Prese...	
DB_Settings_t_Vision_Timeout	Default tag table	Time	HMI_Connectio...	PLC_1	DB_Settings.t_Vision_Ti...	
D_Extended	Default tag table	Bool	HMI_Connectio...	PLC_1	D_Extended	
D_Retracted	Default tag table	Bool	HMI_Connectio...	PLC_1	D_Retracted	
Moto_3	Default tag table	Bool	HMI_Connectio...	PLC_1	Moto_3	
Motor_2	Default tag table	Bool	HMI_Connectio...	PLC_1	Motor_2	
Motor_Conv	Default tag table	Bool	HMI_Connectio...	PLC_1	Motor_Conv	
S_1	Default tag table	Bool	HMI_Connectio...	PLC_1	S_1	
S_2	Default tag table	Bool	HMI_Connectio...	PLC_1	S_2	
Start_PB	Default tag table	Bool	HMI_Connectio...	PLC_1	Start_PB	

Figura 43 - Listagem de tags utilizadas pela HMI.

3.4. *Set-up* do *Raspberry Pi*

De maneira a permitir que um sistema de visão artificial e a própria comunicação com o PLC Siemens seja executado, foi necessário preparar o *Raspberry Pi* como plataforma secundária de tratamento de dados, aproveitando o seu potencial de processamento. Deu-se início ao processo através da instalação do sistema operativo, a configuração ao acesso remoto, a instalação de todas as bibliotecas necessárias ao funcionamento pretendido e as configurações utilizadas de forma a potenciar o melhor resultado na captura de imagem, através da *webcam*.

3.4.1. Preparação do *Raspberry OS*

Primeiramente realizou-se a instalação do sistema operativo Rapsberry Pi OS Lite (versão 64-bit). O mesmo foi selecionado dada a exclusão de um ambiente gráfico, não seria necessário em aplicações semelhantes, permitindo dessa forma, reduzir o consumo de recursos do sistema, garantindo uma maior estabilidade para o processo requerido.

Iniciou-se o processo formatando o cartão SD utilizando a ferramente disponibilizada pela *Raspberry* (*Raspberry Pi Imager*), disponível no website do fabricante.

Concluída a instalação da aplicação no computador, foi criada a imagem no cartão através da seleção de:

- Raspberry Pi OS Lite (64-bit);
- O cartão SD a ser colocado no Raspberry;

A figura 44 apresenta o ambiente onde é necessária a escolha do sistema operativo a instalar no cartão SD do *Raspberry Pi*.

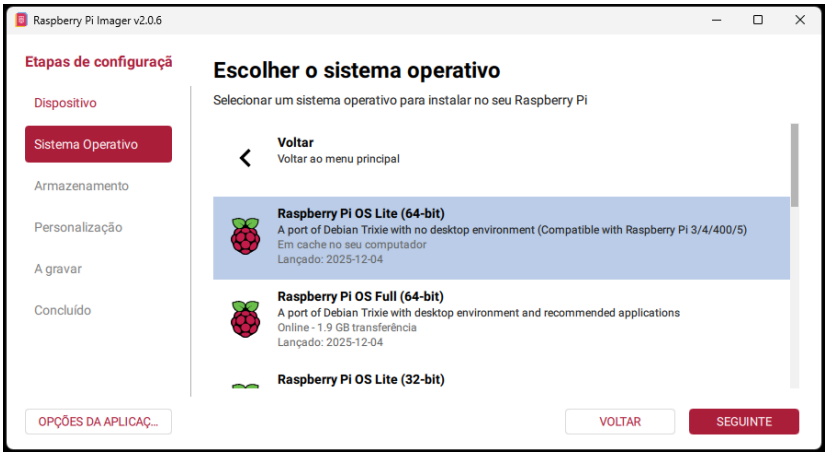


Figura 44 - Ambiente de seleção do sistema operativo no *Raspberry Pi Imager*.

Nas opções avançadas do *Imager* foram, também, configurados o utilizador, o acesso à rede e, fator importante, a ativação de acessibilidade remota via SSH, como apresentado através da figura 45.

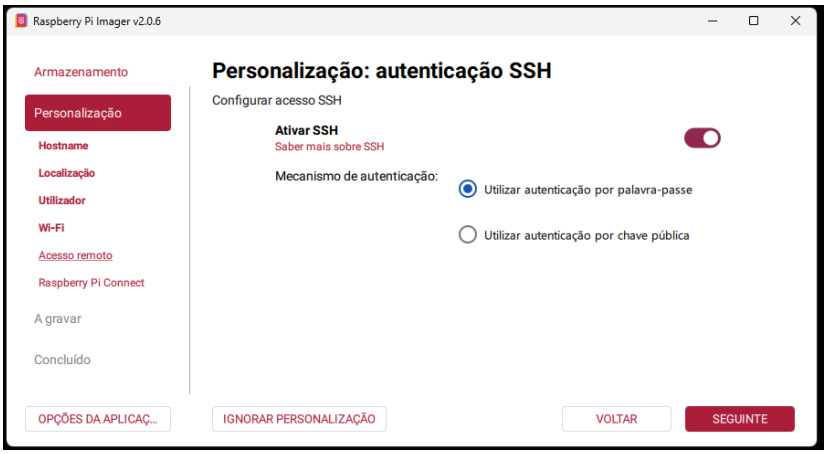


Figura 45 - Ambiente de seleção da opção de acesso remoto ao Raspberry Pi Imager.

3.4.2. Inicialização do *Raspberry Pi*

Após instalado o sistema operativo no cartão SD, o mesmo foi inserido no *Raspberry Pi.* Em seguida, foi efetuada a ligação ao dispositivo em rede local através de um cabo *ethernet* ligado ao mesmo *switch* onde se encontram o PLC e a HMI.

Após iniciado o sistema, foi necessário identificar o endereço de IP associado ao próprio *Raspberry* e proceder à sua atualização. Para isto usaram-se os seguintes comandos:

- sudo apt update
- sudo apt upgrade -y

Através do código apresentado seguidamente foi possível aceder ao ficheiro dhcpcd.conf de forma a editar parâmetros que tornaram o endereço de IP estático. Desta forma, a ligação remota tornou-se mais simples. Para o mesmo selecionou-se o IP 192.168.30.10.

- sudo nano /etc/dhcpcd.conf

Após definido um IP estático e conhecido (e estando o computador a utilizar o IP 192.168.30.200) foi possível estabelecer-se a ligação remota com o equipamento. Essa ligação realiza-se através do seguinte comando:

- ssh maquetedee@192.168.30.10

De notar que "maquetedee" é o nome do perfil do utilizador do *Raspberry*, definido durante a configuração através do *Imager*.

Numa fase inicial deve se recorrer às definições de rede e ethernet do próprio computador de forma a estabelecer o correto endereço de IPv4. Em seguida foi feita uma confirmação de que o Raspberry estava num endereço alcançável.

A figura 46 apresenta o ambiente de edição das definições de IP do *Windows* e a figura 47 o ambiente após estabelecida a ligação remota.

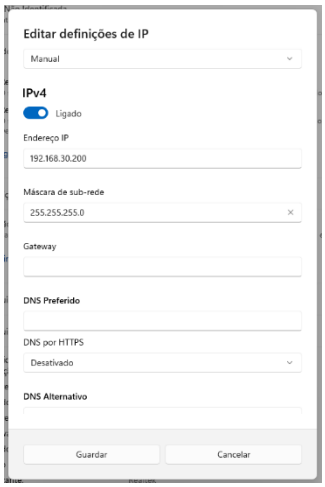


Figura 46 - Janela de configuração do endereço de IP em ambiente Windows.

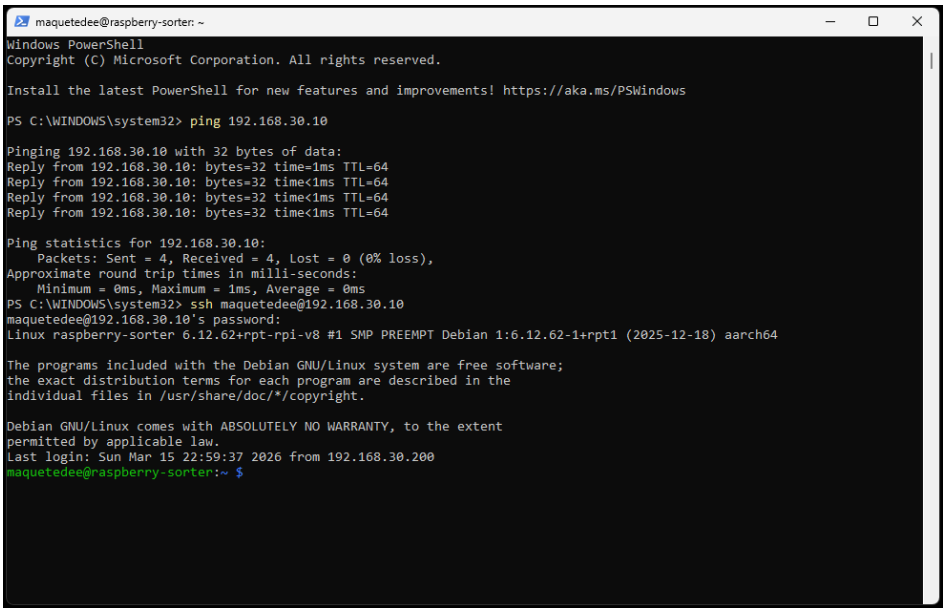


Figura 47 - Ligação remota por SSH concluída com sucesso.

3.4.3. Instalação de ferramentas necessárias

De forma a permitir a execução e desenvolvimento do sistema de visão artificial, foi necessário instalar um conjunto de ferramentas base neste equipamento, incluindo bibliotecas de Python, entre outros.

A instalação foi feita através da seguinte linha de comando:

- sudo apt install -y python3 python3-pip python3-venv git

Adicionalmente foram instaladas ferramentas mais específicas para o processamento da imagem, tratamento e captura.

- sudo apt install -y python3-opencv
- sudo apt install -y v41-utils

Este pacote (v41-utils) permite controlar diretamente parâmetros da *webcam* através do sistema operativo, tais como o brilho, saturação, foco, exposição, entre outros.

De forma a evitar conflitos entre bibliotecas do sistema e as dependências do projeto em si, foi criado um ambiente virtual dedicado para este fim. O comando utilizado foi:

- python3 -m venv snap7_env

Desta forma, é possível obter um ambiente isolado e destinado aos programas a serem utilizadas. De modo a aceder a esse mesmo ambiente é necessário, sempre que se liga o *raspberry*, executar o comando:

- source ~/snap7_env/bin/activate

Na figura 48 é possível verificar que estamos dentro desse ambiente gerado pelo *feedback* oferecido pelo próprio terminal.

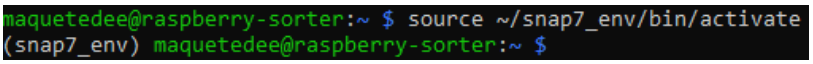


Figura 48 - Verificação da entrada no ambiente python criado.

Em seguida, dentro do ambiente virtual, foi atualizado o gestor pip de forma a garantir assim compatibilidade com as bibliotecas mais recentes. Para o mesmo recorreu-se ao comando:

- pip install -upgrade pip

3.4.4. Instalação das bibliotecas necessárias

Após definido o ambiente, foi possível dar seguimento à instalação de todas as bibliotecas necessárias para o correto funcionamento do algoritmo de visão e da comunicação com o PLC.

Em seguida, referem-se as principais bibliotecas utilizadas:

- pip install numpy
- pip install matplotlib
- pip install scikit-image
- pip install opencv-python

- pip install python-snap7
- pip install flask

A biblioteca *Snap7* é a que permite estabelecer a comunicação entre o *raspberry* e os controladores *Siemens*, sendo este o protocolo escolhido para a troca de informação ao longo deste projeto.

3.4.5. Identificação e controlos da *webcam*

Após instaladas todas as dependências necessárias, foi identificada a *webcam* nos dispositivos instalados no *Raspberry* de forma a testar-se o modo de captura. Para observar todos os dispositivos ativos foi utilizado o comando:

- v4l2-ctl --list-devices

O resultado dessa pesquisa apresenta-se na figura 49.

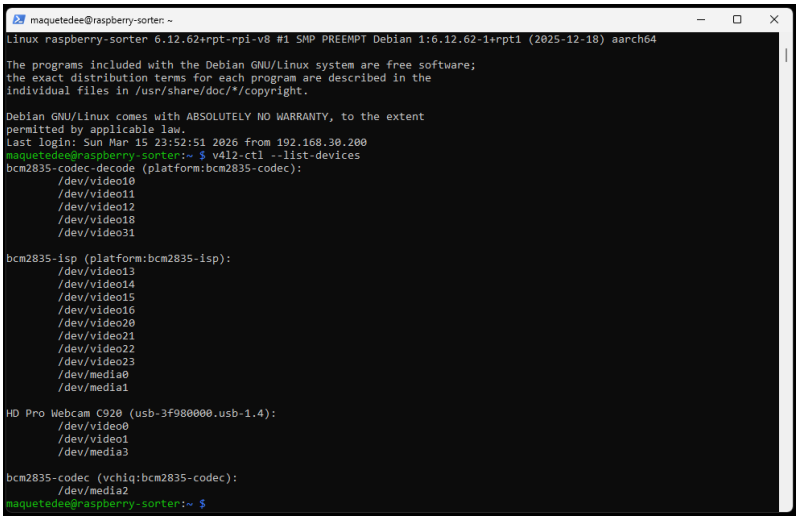


Figura 49 - Dispositivos instalados.

Desta forma foi possível confirmar o endereço associado à *webcam* (neste caso /dev/video0).

Para facilitar a parametrização, foi adicionada à programação um comando que permite o tratamento de todas as opções indicadas seguidamente. De modo a saber que parâmetros são atingíveis para balancear a qualidade de imagem de uma forma otimizada (figura 50), utilizou-se o comando:

- v4l2-ctl -d /dev/video0 --list-ctrls

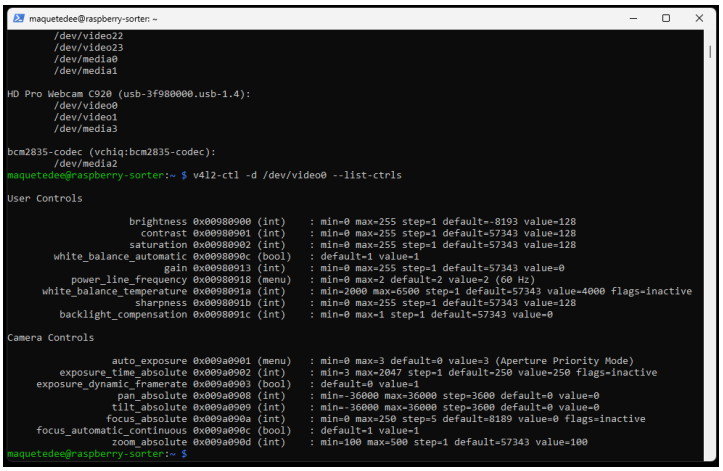


Figura 50 - Lista de controlos completa da webcam.

3.4.6. Transferência de ficheiros

Durante a definição dos parâmetros a serem utilizados para a câmara, foi necessário aceder ao ambiente do *Raspberry* com o objetivo de extrair as imagens produzidas. Desta forma, utilizou-se o protocolo scp de modo a permitir-se copiar ficheiros através da ligação ssh. Em seguida apresenta-se o comando executado para esse fim:

- scp capture.jpg maquetedee@192.168.30.10:/home/maquetedee

A figura 51 apresenta o momento da transferência de uma imagem proveniente do *Raspberry Pi*.

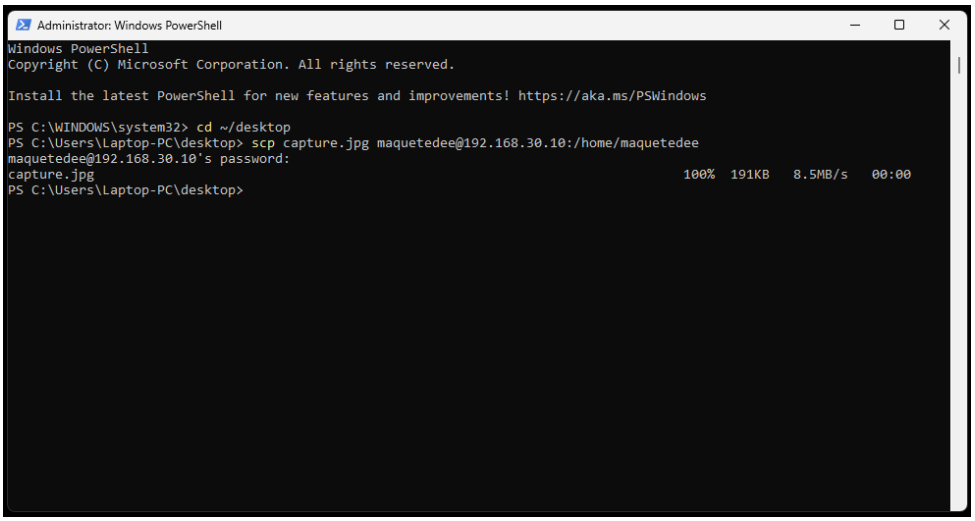


Figura 51 - Momento de transferência de uma imagem utilizando o scp.

É importante que se realize aquele comando inicial "cd ~/[caminho]" de forma a guardar o ficheiro no [caminho] definido, caso contrário a imagem seria guardada na pasta *system32*.

3.5. Algoritmo de Reconhecimento por Visão Artificial

O objetivo do algoritmo criado visa não só a identificação, mas também a validação das peças, recorrendo à visão artificial. Este programa é executado através de um *Raspberry* Pi em comunicação direta com um PLC *Siemens* S7-1214C através do protocolo *S7*, utilizando a biblioteca snap7.

O fluxo de funcionamento baseia-se na ordem de pedido executada pelo PLC e recebida pelo *Raspberry*. Este, captura uma imagem e, através de máscaras, recortes e morfologias aplica um tratamento à imagem para que a informação, depois de processada, classifique a peça conforme os parâmetros definidos, sendo, posteriormente, devolvido um valor ao PLC para que o mesmo progrida no seu *grafcet*.

3.5.1. Importação de Bibliotecas

A primeira secção do código apresenta-nos as diversas bibliotecas importadas (figura 52) para que seja possível a manipulação pretendida da imagem. São utilizadas bibliotecas padrão de *Python*, bem como bibliotecas especializadas para o tratamento da imagem e estabelecimento de comunicação com o PLC.

```
import time
import subprocess
from pathlib import Path

import cv2
import numpy as npy
import matplotlib.pyplot as plt

from skimage import img_as_float, transform, data, measure
from skimage.color import rgb2gray, rgb2hsv, hsv2rgb
from skimage.measure import label, regionprops, ransac
from skimage.feature import SIFT, match_descriptors#, plot_matches
from skimage.filters import threshold_local,prewitt_v,prewitt_h,sobel_v,sobel_h,laplace, unsharp_mask, gaussian, threshold_otsu
from skimage.restoration import inpaint
import skimage.morphology as mph

import snap7
from snap7.util import get_bool, set_bool
```

Figura 52 - Cabeçalho de chamada das diferentes bibliotecas utilizadas.

3.5.2. Configuração dos Parâmetros Gerais

Depois de importadas as bibliotecas, são definidos parâmetros de configuração do equipamento físico (Logitech C920). Esses parâmetros servem de apoio à configuração do programa em si. Aqui é, também, definido o caminho onde é guardada a captura.

A figura 53 apresenta alguns parâmetros de configuração da *webcam* embutidos no código de forma a aumentar a repetibilidade do sistema.

```
24  VIDEO_DEVICE = "/dev/video0"
25  WIDTH = 1920
26  HEIGHT = 1080
27  FOCUS = 25
28
29  CAPTURE_PATH = "capture.jpg"
```

Figura 53 - Exemplo de parâmetros configuráveis inseridos no código de visão artificial.

3.5.3. Parâmetros de Comunicação *Snap7*

Depois de importadas as bibliotecas e configurada a *webcam*, é definido um *set* de parâmetros que permite a comunicação entre o *Raspberry* e o PLC. Aqui são especificados valores do endereço de IP, rack e slot utilizados na ligação e o número do bloco de dados a ser utilizado para a troca de informação.

Para além dessa informação, são também apresentados os *bits* a serem interpretados e escritos na DB do PLC. Estes sinais permitem implementar um protocolo simples de comunicação capaz de proceder ao pedido do processamento feito por parte do PLC, registo da validade do resultado, indicação de erro de leitura e a consequente validação da peça em teste (figura 54).

```
PLC_IP = "192.168.30.1"
RACK = 0
SLOT = 1
DB_NUMBER = 11

BYTE_FLAGS = 0
BIT_PLC_REQUEST = 0
BIT_PLC_BUSY = 1
BIT_RPI_VALID = 2
BIT_RPI_ERROR = 3
BYTE_RPI_RESULT = 1
```

Figura 54 - Configurações de comunicação entre o PLC e o Raspberry Pi.

Estes bits representam a ação de pedido de teste, a capacidade do PLC entender se o *Raspberry* se encontra em uso, a validade do resultado, o erro no resultado e o próprio valor resultante do teste.

3.5.4. Configuração dos Parâmetros da *Webcam*

Como previamente referido através da figura 49, neste código, foi adicionada uma secção que força a integração das opções selecionadas de forma a garantir uma

estabilidade nas diferentes parametrizações relacionadas com a webcam. Nesta secção procedeu-se à desativação da opção de autofoco, definição do foco manual, desativação do balanceamento de brancos automático e a sua definição de forma manual, definição da temperatura de cor e a configuração de tempo de exposição.

3.5.5. Aquisição da Imagem

A captura da imagem a ser trabalhada é realizada através da função apresentada na figura 55.

```
def capture_image():

    cap = cv2.VideoCapture(VIDEO_DEVICE, cv2.CAP_V4L2)

    if not cap.isOpened():
        raise RuntimeError("Erro: webcam nao abriu")

    cap.set(cv2.CAP_PROP_FRAME_WIDTH, WIDTH)
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, HEIGHT)

    time.sleep(0.5)

    frame = None

    for _ in range(5):
        ret, frame = cap.read()

    cap.release()

    if frame is None:
        raise RuntimeError("Erro a capturar imagem")

    cv2.imwrite(CAPTURE_PATH, frame)

    print("[CAM] imagem guardada:", CAPTURE_PATH)

    print("[CAM] resolucao:", frame.shape)
```

Figura 55 - Rotina de captura de imagem.

Esta função estabelece a ligação à *webcam* utilizando a biblioteca *OpenCV* e, consequentemente, verifica a integridade da ligação e define a resolução da imagem

capturada. É utilizado um temporizador de forma a permitir que a camara estabilize antes de executada a captura.

Em seguida, são capturados 5 *frames*, mantendo, apenas, o último. Posto isto a imagem é arquivada de forma a ser utilizada para o seu processamento posterior.

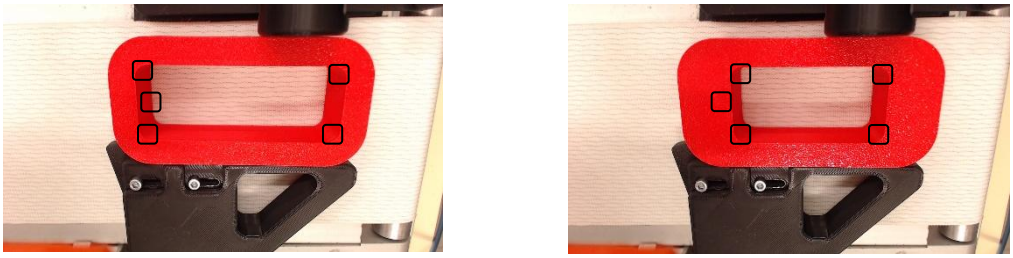
3.5.6. Processamento da Imagem

A função detect_piece() constitui a base do código de reconhecimento e processamento da imagem capturada de forma a identificar o tipo e a conformidade da mesma. Para cada uma das categorias são analisadas diferentes regiões da imagem que correspondem a zonas específicas de tratamento das peças.

3.5.7. Segmentação da Imagem

De forma a facilitar a análise destas zonas específicas das peças, a imagem foi dividida em submatrizes da imagem real, com dimensão de 70 píxeis por 70 píxeis, processo geralmente designado por ROI (*Region of Interest*).

É possível observar estas regiões através da figura 56.



A) B)

Figura 56 - Representação das regiões de interesse: A) Peça A; B) Peça B.

A configuração das fronteiras das ROI foi realizada de forma a manter as suas coordenadas entre a análise de peças conformes e não conformes, tanto para as peças do

tipo A, como do tipo B. Esta abordagem garante que a classificação depende exclusivamente das características geométricas e visuais das peças, e não de variações na localização da análise.

Posteriormente, o espaço de cores dessas regiões é transformado de RGB para HSV permitindo maior facilidade na procura pela cor vermelha.

Depois de convertidos esses espaços, são extraídos três canais do modelo HSV: a tonalidade, saturação e a intensidade luminosa.

3.5.8. Máscaras de cor

De forma a facilitar a análise destas sub-regiões, foi necessário aplicar uma segmentação baseada na cor vermelha presente nas peças.

Numa primeira fase é aplicado um filtro Gaussiano com objetivo de reduzir o ruído presente na imagem.

Em seguida é criada a máscara de cor, que identifica os píxeis cuja tonalidade corresponde à faixa de vermelho aceitável. Esses parâmetros/limites facilitam a remoção dos elementos que não constituem a peça a ser identificada.

O resultado deste processo apresenta um grande pormenor já em torno da peça.

A figura 57 apresenta, de forma simples, três etapas distintas ao longo do processamento visual industrial de uma imagem.

Figura 57 - Diferentes fases do processamento da imagem, desde a captura ao resultado final.

3.5.9. Operações morfológicas

Depois de concluída a segmentação inicial, são aplicadas operações morfológicas para melhorar a qualidade da máscara obtida.

Através da operação de *binary closing*, que resumidamente força a imagem a passar por uma dilatação, seguida de uma erosão, é possível obter uma grande diminuição de pontos de ruído que surjam com todos os processos prévios. Desta forma, a aparência de granulado em algumas imagens fica menos notável ou acaba, em alguns casos, por fechar por completo a peça, produzindo um melhor resultado.

3.5.10. Validação da peça

Para cada uma das submatrizes é calculada a soma dos valores dos píxeis presentes nesta combinação já binária.

Este cálculo resulta numa soma de quantidade de píxeis com tonalidade branca, podendo, assim, distinguir uma peça com o filete das que não o têm. Em simultâneo, foi implementado um ponto de *poka-yoke*, o que permite a diferenciação das peças do tipo A, das peças do tipo B.

Cada região gera, através deste princípio, uma verificação lógica que indica se a zona foi corretamente identificada ou não.

3.5.11. Classificação final

Após a análise destas regiões, o sistema distingue os valores e valida conforme o seu desempenho na comparação com os valores permitidos pelos intervalos definidos.

Os resultados possíveis para esta operação são:

- Peça tipo A – conforme;

- Peça tipo A – não conforme;
- Peça tipo B – conforme;
- Peça tipo B – não conforme;
- Resultado inconclusivo.

Estes valores são, depois, enviados para o PLC de forma a este prosseguir com o ciclo em ação.

3.6. Algoritmo de Exportação de Dados para a Página *Web*

O código desenvolvido visa implementar uma pequena plataforma de monitorização capaz de estabelecer comunicação direta com o PLC utilizado neste projeto, lê a sua DB reservada aos dados estatísticos (DB12 – *DB_Statistics*) e disponibiliza esses valores recolhidos diretamente num servidor *web*. O servidor intervém através de dois acessos distintos: um através de uma API em JSON, destinada à captura dos dados requeridos, e outra *dashboard* que apresenta os valores em tempo real. A aplicação foi desenvolvida com recurso à linguagem *Python* utilizando bibliotecas *Flask* e *Snap7*, estando esta última já instalada no âmbito do processamento de imagem previamente descrito. Desta forma foi possível integrar dados do sistema de controlo numa interface simples de monitorização de dados, através da rede.

3.6.1. Importação de bibliotecas

Para este algoritmo foi necessário importar as bibliotecas apresentadas na figura 58.

```
from flask import Flask, jsonify
import snap7
import struct
```

Figura 58 - Bibliotecas importadas para a plataforma web.

A biblioteca *Flask* é responsável por criar o ambiente *web*. A função *flask* cria a aplicação que estabelecerá correspondência com os pedidos HTTP, enquanto a função *jsonify* é responsável por converter as estruturas de dados de *Python* em JSON, formato mais comum e amplamente utilizado em soluções semelhantes. A biblioteca *Snap7* permite a comunicação direta entre o *Raspberry* e o PLC *Siemens*, disponibilizando a informação de forma à mesma ser trocada com os *data blocks* existentes.

3.6.2. Parametros da comunicação *Snap7*

Após a importação das bibliotecas, define-se um conjunto de parâmetros no programa (figura 59). Por exemplo, PLC_IP define o IP do controlador na rede. As variáveis *Rack* e *Slot* identificam a posição do CPU na configuração do controlador Siemens. A variável *DB_Number* indica qual dos *datablocks* está a armazenar a informação.

```
PLC_IP = "192.168.30.1"
RACK = 0
SLOT = 1
DB_NUMBER = 12
```

Figura 59 - Parâmetros de comunicação.

3.6.3. Criação da aplicação Web e conexão

De forma a ser criada a aplicação utilizou-se o comando:

- app = Flask(__name__)

Esta simples linha inicializa a criação através de um *framework Flask*. O valor "__name__" permite que o *Flask* localize recursos que poderão servir esta aplicação, contudo, a mesma foi limitada ao código HTML diretamente introduzido.

De forma a ser estabelecida a ligação ao PLC recorreu-se aos comandos apresentados na figura 60.

```
client = snap7.client.Client()
client.connect(PLC_IP, RACK, SLOT)
```

Figura 60 - Comandos de arranque da comunicação Raspberry – PLC.

Desta forma, cria-se um objeto cliente da biblioteca, representativo da ligação ao PLC. Em seguida, através do "client.connect", é feita a ligação para o PLC no caso de os parâmetros estarem conformes. Estabelecida a ligação, o *raspberry* já é capaz de realizar leituras dos *Data Blocks* existentes.

Assim, após o estabelecimento da ligação descrita, é possível aceder aos dados que se pretende mostrar na página *web*, utilizando a instrução:

- data = client.db_read(DB_NUMBER, 0, 28)

Este método permite a leitura direta de blocos de dados na memória do CPU. Como cada variável do tipo DInt ocupa 4 bytes, o espaço criado possibilita a integração de 7 variáveis deste tipo.

Na figura 61 apresenta-se uma função que procura os *bytes* pretendidos e devolve-os cada um no seu espaço de forma que, através da função "*stats*" seja possível individualizar cada um deles.

```
def dint(offset):
    return struct.unpack(">i", data[offset:offset+4])[0]

stats = {
    "Total_OK":  dint(0),
    "Total_NOK": dint(4),
    "Total_All": dint(8),
    "A_OK":      dint(12),
    "A_NOK":     dint(16),
    "B_OK":      dint(20),
    "B_NOK":     dint(24)
}

return stats
```

Figura 61 - Leitura das variáveis do PLC através de offsets no Data Block.

Desta forma somos capazes de fazer a separação do espaço de funcionamento de cada variável, ficando a mesma acessível.

3.6.4. API - *Endpoint*

A Figura 62 ilustra a definição do *endpoint* utilizado para expor os dados do PLC sob a forma de objeto JSON.

A anotação apresentada no início obriga a função *stats*() a ser executada sempre que existe um novo cliente a aceder ao endereço. A função chama a "*read_stats*()" de forma a obter os dados necessários para proceder ao *Jsonify*. Feito isto, o código fica formatado num objeto JSON.

```
@app.route("/api/stats")
def stats():
    return jsonify(read_stats())
```

Figura 62 - Disponibilização de dados do PLC através de endpoint.

3.6.5. Programação HTML

A função inicial da *dashboard* devolve uma página feita em HTML como estrutura base, sendo a aplicação da linguagem CSS responsável pela formatação visual e a linguagem JavaScript permite atualizar os valores apresentados.

A secção HTML é responsável por definir os vários elementos identificados, como apresenta a figura 63.

```
<body>

<div class="wrap">

<div class="card">
<h1>Monitorização — DB_Statistics</h1>
<div class="muted">Atualiza automaticamente a cada 1 s.</div>
</div>

<div class="grid">

<div class="card">

<div class="kpi">
<div class="label">Total (todas)</div>
<div class="value" id="Total_All">0</div>
</div>

<div class="kpi">
<div class="label">Total OK</div>
<div class="value" id="Total_OK">0</div>
</div>

<div class="kpi">
<div class="label">Total NOK</div>
<div class="value" id="Total_NOK">0</div>
</div>

</div>

<div class="card small">

<div class="kpi">
<div class="label">A — OK</div>
<div class="value" id="A_OK">0</div>
</div>

<div class="kpi">
<div class="label">A — NOK</div>
<div class="value" id="A_NOK">0</div>
</div>

<div class="kpi">
<div class="label">B — OK</div>
<div class="value" id="B_OK">0</div>
</div>

<div class="kpi">
<div class="label">B — NOK</div>
<div class="value" id="B_NOK">0</div>
</div>
```

Figura 63 - Estrutura HTML da interface de monitorização de dados do PLC.

Cada um destes elementos é responsável por configurar uma das variáveis a serem apresentadas e atualizadas no ambiente *web*.

3.6.6. Programação JavaScript

Já no fim da programação encontramos um script responsável pela atualização dos valores mostrados na página. Em suma, a função de *update*() executa pedidos HTTP ao /api/stats através da função apresentada na figura 64.

```
async function update(){

let r = await fetch('/api/stats');
let j = await r.json();

document.getElementById("Total_OK").innerText = j.Total_OK;
document.getElementById("Total_NOK").innerText = j.Total_NOK;
document.getElementById("Total_All").innerText = j.Total_All;

document.getElementById("A_OK").innerText = j.A_OK;
document.getElementById("A_NOK").innerText = j.A_NOK;

document.getElementById("B_OK").innerText = j.B_OK;
document.getElementById("B_NOK").innerText = j.B_NOK;

}
```

Figura 64 - Atualização dinâmica dos indicadores da interface web através de JavaScript.

A função *fetch()* envia um pedido ao servidor e, quando recebida a resposta, converte os valores JSON para objeto JavaScript. De seguida o valor é recebido e inserido no HTML.

De forma que a página seja constituída por valores em tempo real, foi implementada uma função de atualização dos dados a cada 1000ms.

Por fim, o ambiente virtual é iniciado através da função apresentada na figura 65.

```
# ==============================
# START SERVER
# ==============================

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

Figura 65 - Inicialização do servidor Flask e configuração de acesso em rede.

A função condiciona o servidor a ser iniciado só e apenas quando o script é executado de forma direta. A secção *app.run()* inicia o *Flask* com os parâmetros *host*=”0.0.0.0” que possibilita ao servidor ligações de diferentes dispositivos da rede e a *port*=5000 define a porta de comunicação.

3.6.7. Síntese do código aplicado

Resumidamente, o código implementado consiste na execução de um servidor *Flask* que lê de forma continua os dados estatísticos armazenados no *Data Block* do PLC através do protocolo S7. Os dados são tratados de forma a serem apresentados em JSON e mostrados numa plataforma *web* com atualizações cíclicas de 1s. Desta forma é possível acompanhar o sistema de triagem a partir de qualquer dispositivo ligado à rede, em qualquer momento.

4. ANÁLISE E DISCUSSÃO DOS RESULTADOS

Após diversos testes de funcionamento verifica-se que a máquina executou ciclos em modo automático e foi capaz de manter a sua fiabilidade, sempre que a configuração se manteve consistente.

De forma a validar o trabalho desenvolvido, foram utilizados 4 conjuntos de peças diferentes (2 unidade de peças A OK, 2 unidades de peças A NOK, 2 unidades de peças B OK e 1 unidade de peças B NOK). As peças foram colocadas na máquina sendo o resultado apresentado na figura 66.

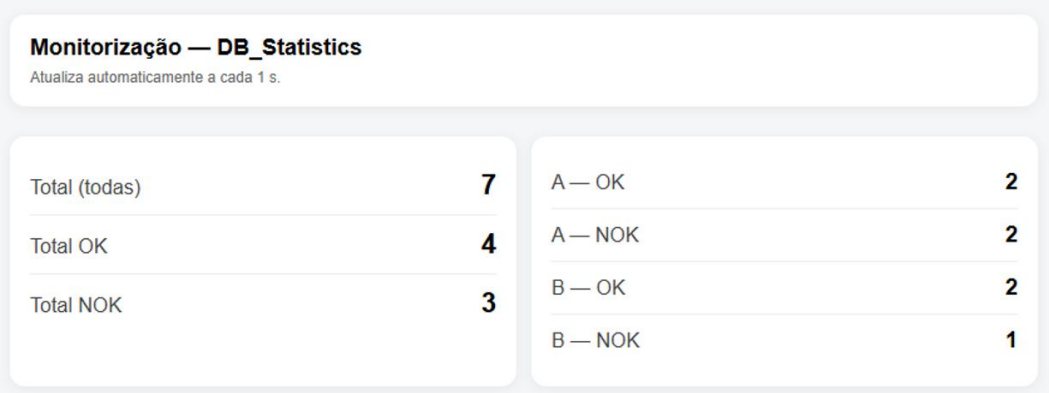


Figura 66 - Resultados do teste na plataforma web.

Esta figura apresenta o ambiente *web* desenvolvido com a estatística relativa às quantidades de peças classificadas durante o ciclo de validação. É possível verificar que os valores apresentados correspondem ao conjunto de peças introduzidos no sistema, evidenciando a coerência entre o comportamento esperado e os resultados obtidos.

Observa-se a correta distinção entre peças conformes e não conformes, bem como entre os tipos A e B, validando a integração entre o PLC, o sistema de visão artificial e a plataforma *web*.

A tabela 2 apresenta os intervalos de valores aceitáveis, definidos para avaliar o desempenho em cada ROI, dependendo do tipo de peça em análise.

	Número de pixeis aceitáveis por região de interesse									
	Canto A		Canto B		Canto C		Canto D		Binário	
	Valor Ref	Range	Valor Ref	Range	Valor Ref	Range	Valor Ref	Range	Valor Ref	Resultado
Peça A_OK	1009	900 - 1100	1041	950 - 1150	1217	1100 - 1300	1738	1650 - 1850	0	Peça A
Peça A_NOK	438	50 - 650	106	50 - 650	184	50 - 650	556	50 - 650	0	Peça A
Peça B_OK	1092	1000 - 1200	1287	1200 - 1400	1187	1100 - 1300	1219	1100 - 1300	3742	Peça B
Peça B_NOK	19	10 - 650	481	50 - 650	577	50 - 650	404	50 - 650	4900	Peça B

Tabela 2 - Valores de referência com base nas peças testadas.

É possível, através da análise da tabela 3, observar os valores obtidos em cada ROI examinado para a peça A OK e fazer a sua comparação com os intervalos definidos.

Peça Teste: A_OK	Número de pixeis aceitáveis por região de interesse									
	Canto A		Canto B		Canto C		Canto D		Binário	
	Valor	Range	Valor	Range	Valor	Range	Valor	Range	Valor	Resultado
Peça A_OK	1009	900 - 1100	1041	950 - 1150	1217	1100 - 1300	1738	1650 - 1850	0	Peça A
Peça A_NOK	1009	50 - 650	1041	50 - 650	1217	50 - 650	1738	50 - 650	0	Peça A
Peça B_OK	0	1000 - 1200	433	1200 - 1400	3728	1100 - 1300	221	1100 - 1300	0	Peça A
Peça B_NOK	198	10 - 650	334	50 - 650	2598	50 - 650	0	50 - 650	0	Peça A

Legenda	
Resultado validado	
Resultado ignorado	

Tabela 3 - Resultados por região de interesse para a peça A OK e respetiva legenda.

O critério de validação prioriza a execução da distinção do tipo de peça (A ou B). Este processo realizou-se recorrendo a um ponto *poka-yoke*, isto é, um ponto de clara diferença geométrica entre os produtos. A figura 67 apresenta a interseção entre as peças conformes A e B e a localização do ponto de *poka-yoke*.

Figura 67 - Interseção entre as peças A e B; Localização do *poka-yoke*

A figura 68 apresenta, na primeira linha, as regiões de interesse extraídas (canto A, canto B, canto C, canto D e região de *poka-yoke*) e, na segunda linha, o resultado da segmentação binária após aplicado o algoritmo de processamento.

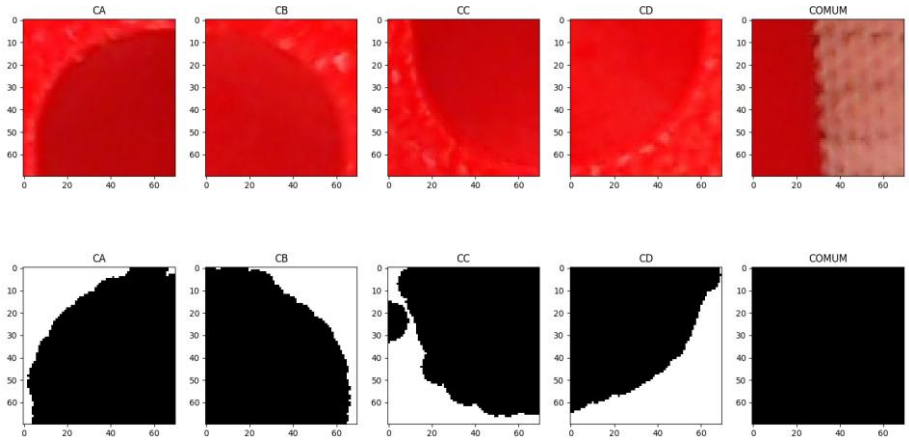


Figura 68 - ROI e segmentação binária - Análise da peça A OK através da ROI de peças A

A análise é feita pela avaliação do desempenho de cada peça alimentada tanto na região de interesse definida para as peças A, como para as peças do tipo B. De forma consequente, a condição para a validação do resultado relativo ao tipo de peça (isto é, se é peça A ou peça B) é definido pelo resultado do desempenho do algoritmo relativamente à região de *poka-yoke*.

De forma semelhante, a figura 69, apresenta a análise feita à peça A OK pela região de interesse das peças B.

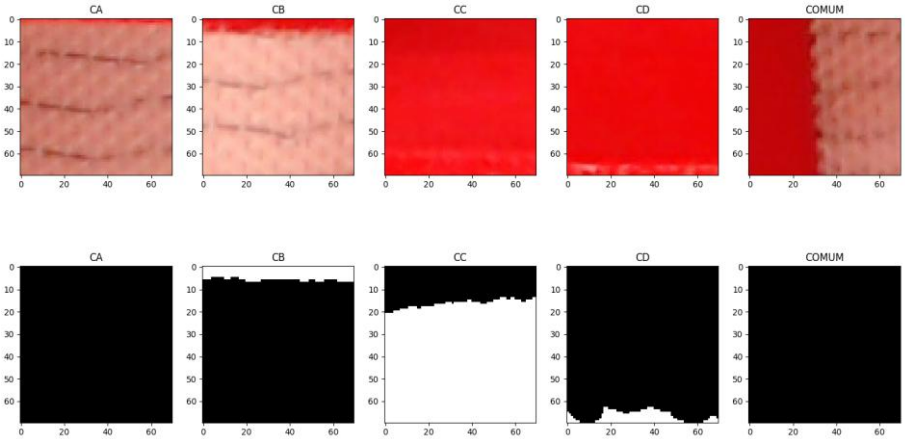


Figura 69 - ROI e segmentação binária - Análise da peça A OK através da ROI de peças B.

Com base na análise da região de *poka-yoke*, verifica-se que o número de píxeis se encontra abaixo do limite definido (50 píxeis), permitindo ao sistema identificar a peça como pertencente ao tipo A.

Após essa identificação, são avaliadas as quatro regiões correspondentes aos cantos da peça. Quando os valores obtidos em cada uma dessas regiões se encontram dentro dos intervalos parametrizados, a peça é classificada como conforme.

Caso uma ou mais regiões não cumpram os critérios definidos, o sistema mantém a identificação do tipo de peça, mas classifica-a como não conforme, sendo ainda possível determinar quantos cantos não satisfazem as condições estabelecidas. Os restantes resultados estão apresentados no Apêndice E e Apêndice F.

O intuito do projeto passou pela aproximação da maquete existente a um sistema de indústria 4.0. Esse objetivo ficou cumprido, contudo, há diversas situações que invalidam garantir o perfeito funcionamento e a sua coerência segundo uma diretiva máquina, por exemplo.

O facto do circuito de segurança, quando violado, desligar a máquina por completo é algo não visto em aplicações industriais.

As barreiras são, geralmente, colocadas em zonas de perigo iminente para o operador e, nesta bancada as mesmas encontram-se numa lateral da máquina, podendo essa ficar protegida com o uso de proteções mecânicas, por exemplo, dando a possibilidade de criar uma barreira na zona de inserção de peça, com capacidade de funcionar a par com o operador.

O desenho 3D serviu para criar uma aproximação da aplicação da maquete em ambientes de simulação e gémeos digitais.

Deverá, caso seja aplicado um sistema de visão semelhante, existir um foco de luz no interior da máquina e uma cortina capaz de bloquear a luz exterior da mesma, de forma

que os resultados provenientes do algoritmo de deteção das peças seja mais imune a condições exteriores. Desta forma, dependendo da intensidade luminosa exterior à sala, o resultado que a máquina apresentava não era totalmente consistente, gerando falsos erros e falsas peças NOK.

5. CONCLUSÃO E MELHORIAS FUTURAS

Conclui-se que é possível a integração de sistemas industriais modernos com o objetivo de melhorar processos já existentes, tendo a solução desenvolvida demonstrado capacidade de funcionamento em modo automático e coerência com os objetivos propostos. Ainda assim, esta abordagem apresenta margem de evolução, podendo ser aprimorada com a introdução de elementos adicionais que aumentem não só a fiabilidade do sistema, mas também a segurança dos seus utilizadores.

A integração da maquete num contexto associado a gémeos digitais encontra-se validada, uma vez que, recorrendo a equipamentos de processamento reduzido e de baixo custo, foi possível implementar mecanismos de aquisição, tratamento e disponibilização de dados, bem como a validação da integridade das peças processadas. Esta abordagem demonstra a viabilidade das arquiteturas utilizadas, onde diferentes sistemas cooperam para atingir um objetivo comum, em alinhamento com os princípios da indústria 4.0.

Numa perspetiva de evolução futura, destaca-se a possibilidade de enriquecer a interface *web* através da introdução de elementos visuais representativos do estado atual da máquina. A apresentação de imagens associadas a diferentes fases do ciclo de funcionamento permitiria, ao utilizador, acompanhar o processo de forma mais intuitiva, criando a perceção de movimento e evolução do sistema, mesmo na ausência de uma representação dinâmica contínua. Esta abordagem constitui uma forma simplificada de virtualização do sistema, permitindo uma leitura visual imediata do seu estado e aproximando o utilizador de uma interação mais imersiva com o processo.

Adicionalmente, a implementação de um conjunto de variáveis parametrizáveis associadas ao sistema de visão artificial representa uma melhoria relevante. A possibilidade de ajuste de parâmetros como brilho, exposição, foco e outros fatores relacionados com a aquisição da imagem permitiria aumentar a robustez do sistema face

a variações do ambiente, nomeadamente alterações de iluminação. Esta abordagem tornaria o sistema mais adaptável, reduzindo a necessidade de intervenção direta no código e promovendo uma configuração mais eficiente e flexível.

Para além dos aspetos referidos, outras melhorias podem ser consideradas, tais como a introdução de mecanismos de deteção de erro mais robustos e a eventual integração de técnicas mais avançadas de processamento de imagem. Estas evoluções permitiriam não só aumentar a fiabilidade do sistema, mas também aproximá-lo de soluções industriais mais completas.

Por fim, a evolução natural do sistema poderá passar por uma maior aproximação ao conceito de *Digital Twin*, através da sincronização contínua entre o estado físico e a sua representação digital. A integração de dados históricos e a análise do comportamento do sistema poderão, numa fase mais avançada, permitir a identificação de padrões de funcionamento e antecipação de falhas, abrindo caminho para abordagens associadas à manutenção preditiva. Ainda que de forma introdutória, o trabalho desenvolvido estabelece uma base sólida para este tipo de evolução, demonstrando que mesmo sistemas de pequena escala podem servir como plataforma para a exploração destes conceitos.

De forma geral, o trabalho desenvolvido evidencia que é possível, em contexto académico, implementar soluções integradas que combinam controlo, visão artificial e monitorização remota, recorrendo a tecnologias acessíveis. Esta abordagem contribui para a compreensão prática dos conceitos associados à Indústria 4.0 e constitui uma base sólida para desenvolvimentos futuros na área da automação e integração de sistemas.

6. REFERÊNCIAS BIBLIOGRÁFICAS

Abayadeera, M. R., & Ganegoda, G. U. (2024). Digital Twin Technology: A Comprehensive Review. *International Journal of Scientific Research*, *10*(4).

Baltazar, A. F. A. B. (2021). *INDÚSTRIA 4.0 E SUSTENTABILIDADE*. Dissertação de Mestrado, Lisbon School of Economics & Management, Universidade de Lisboa.

Bellavista, P., Bicocchi, N., Fogli, M., Giannelli, C., Mamei, M., & Picone, M. (2023). Requirements and design patterns for adaptive, autonomous, and context-aware digital twins in industry 4.0 digital factories. *Computers in Industry*, *149*, 103918. https://doi.org/10.1016/j.compind.2023.103918

Benhanifia, A., Cheikh, Z. B., Oliveira, P. M., Valente, A., & Lima, J. (2025). Systematic review of predictive maintenance practices in the manufacturing sector. *Intelligent Systems with Applications*, *26*, 200501. https://doi.org/10.1016/j.iswa.2025.200501

Bottazzi, D., Foschini, L., & Talamo, R. (2025). Digital Twins in Industry 4.0: A Practitioner's Perspective. *IT Professional*, *27*(4), 25–32. https://doi.org/10.1109/MITP.2025.3530300

Dassault Systèmes and NVIDIA Partner to Build Industrial AI Platform Powering Virtual Twins. (2026, fevereiro 3). https://nvidianews.nvidia.com/news/dassault-systemes-nvidia-industrial-ai

Ha, C. K., Nguyen, H., & Van, V. D. (2025). YOLO-SR: An optimized convolutional architecture for robust ship detection in SAR Imagery. *Intelligent Systems with Applications*, *26*, 200538. https://doi.org/10.1016/j.iswa.2025.200538

Hibbard, K. (2019, junho 18). *Benefits of Automation*. https://kingstar.com/benefits-of-automation/

Iranshahi, K., Brun, J., Arnold, T., Sergi, T., & Müller, U. C. (2025). Digital twins: Recent advances and future directions in engineering fields. *Intelligent Systems with Applications*, *26*, 200516. https://doi.org/10.1016/j.iswa.2025.200516

Lee, J., Bagheri, B., & Kao, H.-A. (2015). *A Cyber-Physical Systems architecture for Industry 4.0*. *Vol. 3*, 18–23.

Leon-Medina, J. X., Tibaduiza, D. A., Parés, N., & Pozo, F. (2025). Digital twin technology in wind turbine components: A review. *Intelligent Systems with Applications*, *26*, 200535. https://doi.org/10.1016/j.iswa.2025.200535

Martins Almeida, F. D. (2021). *Visão artificial no controlo da Célula Flexível de Fabrico da ESTGV*. Dissertação de Mestrado, Escola Superior de Tecnologia e Gestão de Viseu, Instituto Politécnico de Viseu.

Matta, A., & Lugaresi, G. (2024). An Introduction to Digital Twins. *2024 Winter Simulation Conference (WSC)*, 1281–1295. https://doi.org/10.1109/WSC63780.2024.10838793

Mechanical Engineering World. (2026, março 2). *How IoT, AI, and Digital Twins are Reshaping Modern Manufacturing.* https://www.linkedin.com/pulse/how-iot-ai-digital-twins-reshaping-modern-nvnlc

Oliveira, P. V. S. D., Santos, L. D. F., & Ferreira, M. P. (2024). INTELIGÊNCIA ARTIFICIAL NA AUTOMAÇÃO DE PROCESSOS INDUSTRIAIS E SEUS IMPACTOS. *Revista de Economia Mackenzie*, *21*(1), 162–182. https://doi.org/10.5935/1808-2785/rem.v21n1p.162-182

Otsu, N. (1979). *A Threshold Selection Method from Gray-Level Histograms*. IEEE Transactions on Systems, Man, and Cybernetics, vol. *9*, 62–66.

Pérez-Domínguez, L. A., Ávila-Lopez, J. R., & Luviano-Cruz, D. (2023). Tendencia de la Industria 4.0 comparado con Industria 5.0. *Mundo FESC*, *13*(25), 7–19. https://doi.org/10.61799/2216-0388.1248

Raspberry Pi 5. (s.d.). Obtido 3 de março de 2026, de https://www.raspberrypi.com/products/raspberry-pi-5/

Roberg, J.-H., Mosiichuk, V., Silva, R., & Rosado, L. (2025). OPT-IQA: Automated camera parameters tuning framework with IQA-guided optimization. *Intelligent Systems with Applications*, *26*, 200520. https://doi.org/10.1016/j.iswa.2025.200520

Siemens communications overview. (s.d.). [Siemens]. Obtido 2 de abril de 2026, de https://snap7.sourceforge.net/siemens_comm.html

Szeliski, R. (2010). Computer Vision: Algorithms and Applications. *Springer*.

Tao, F., Zhang, H., Liu, A., & Y. C. Nee, A. (2019). *Digital twin in industry: State-of-the-art*. *Vol. 15, No. 4*, 2405–2415.

Xu, L. D., He, W., & Li, S. (2014). Internet of Things in Industries: A Survey. *IEEE Transactions on Industrial Informatics*, *10*(4), 2233–2243. https://doi.org/10.1109/TII.2014.2300753

APÊNDICES

APÊNDICE A

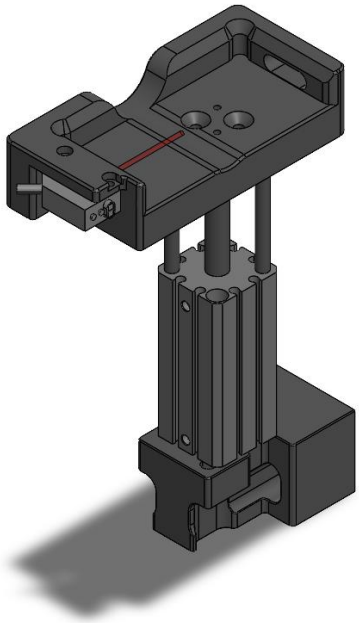


Figura 70 - Conjunto Suporte Cil. Pneumático A.

Figura 71 - Ninho para colocação de peça.

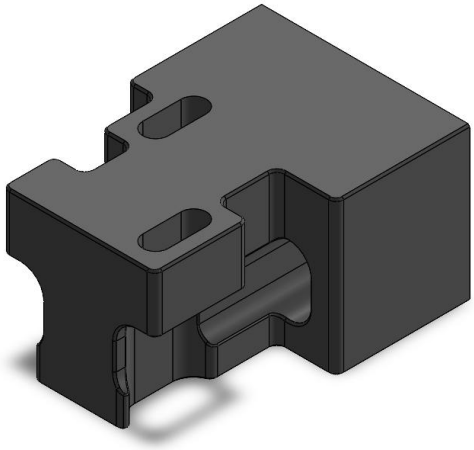


Figura 72 - Peça suporte do cilindro pneumático A.

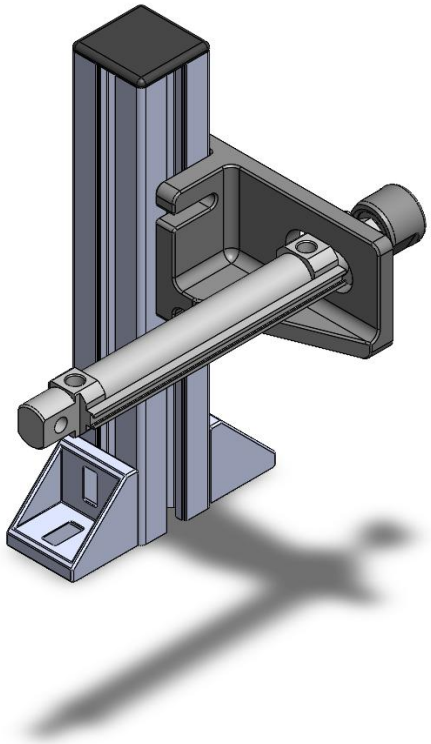


Figura 73 - Conjunto Suporte Cil. Pneumático B.

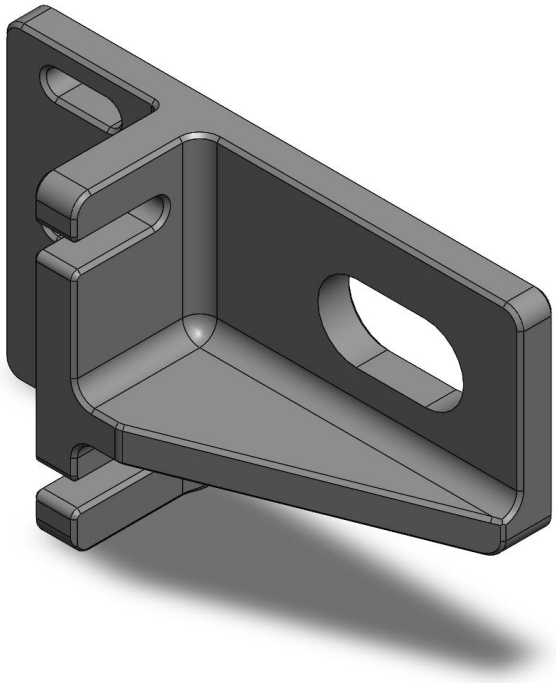


Figura 74 - Peça suporte do cilindro pneumático B.

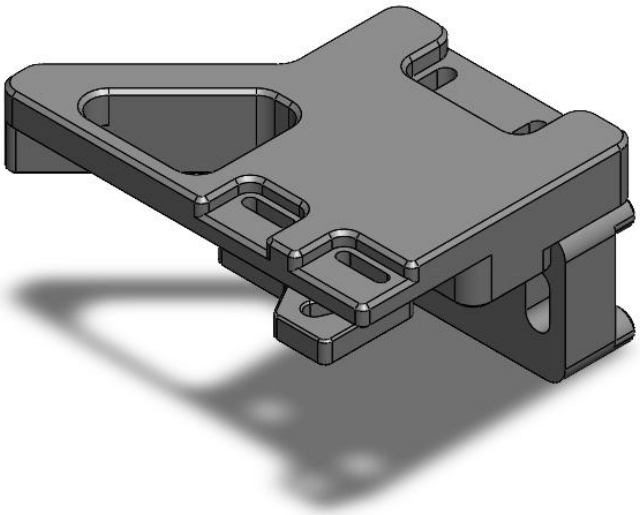


Figura 75 - Conjunto Batente.

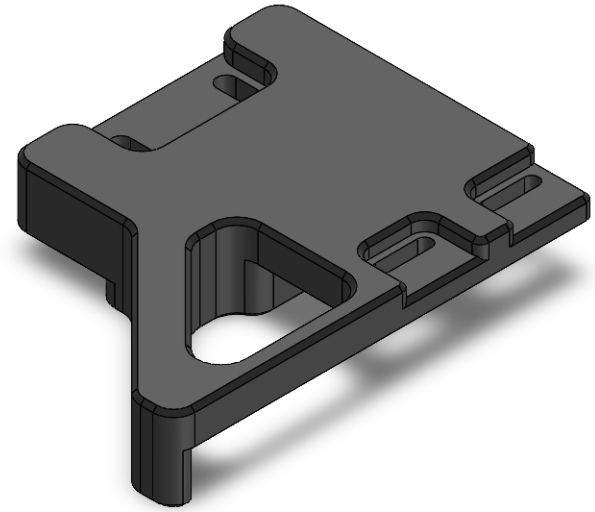


Figura 76 - Peça batente.

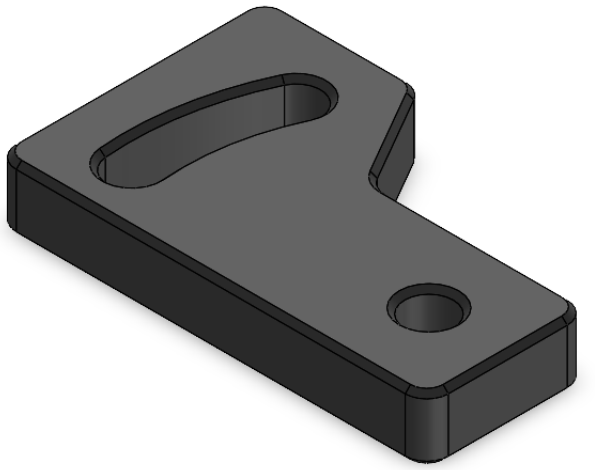


Figura 77 - Peça de ajuste de paragem da peça.

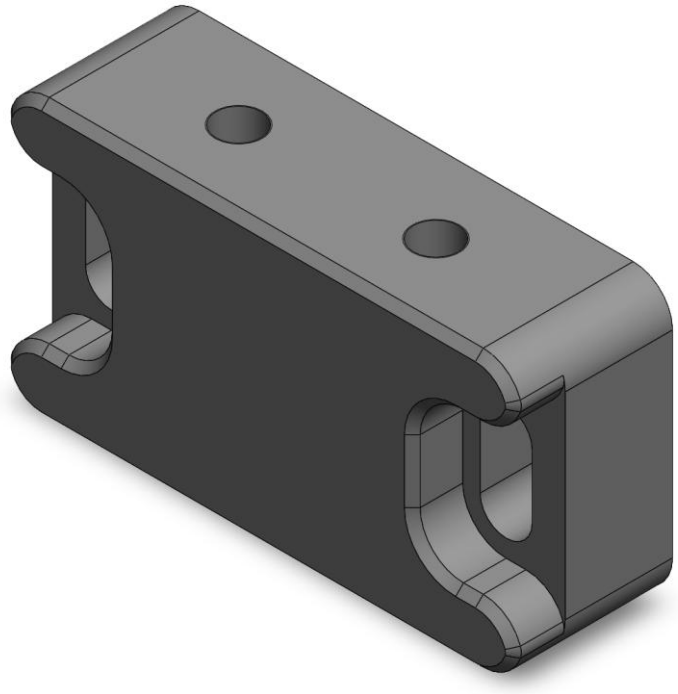


Figura 78 - Peça suporte do batente

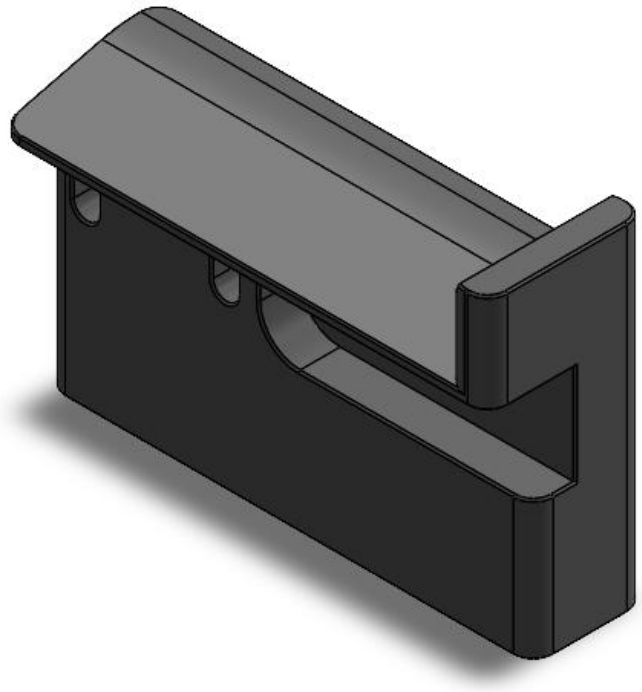


Figura 79 - Peça de guiamento do produto.

APÊNDICE B

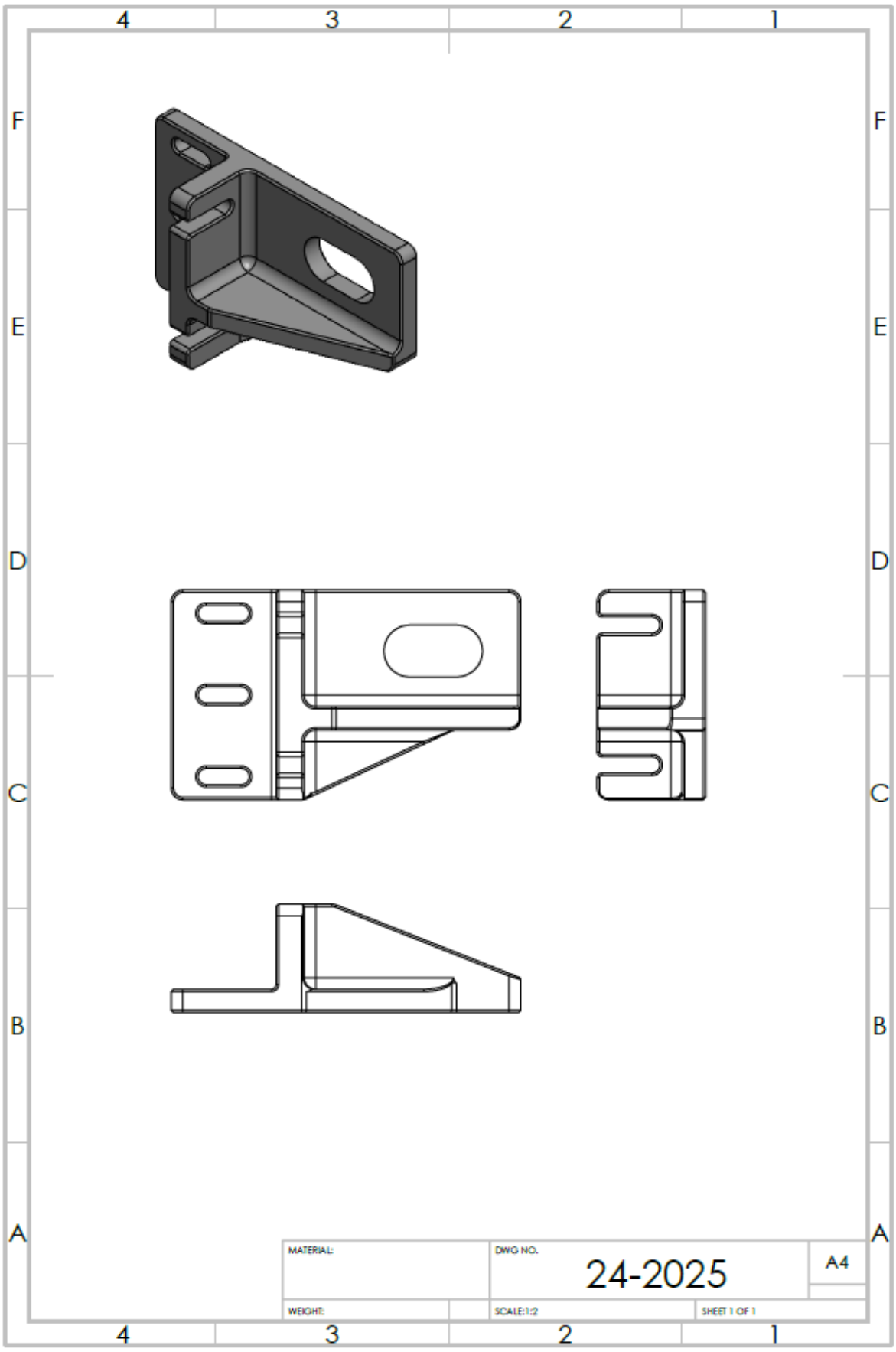


Figura 80 - Vistas simplificadas da peça de fabrico com referência 24.

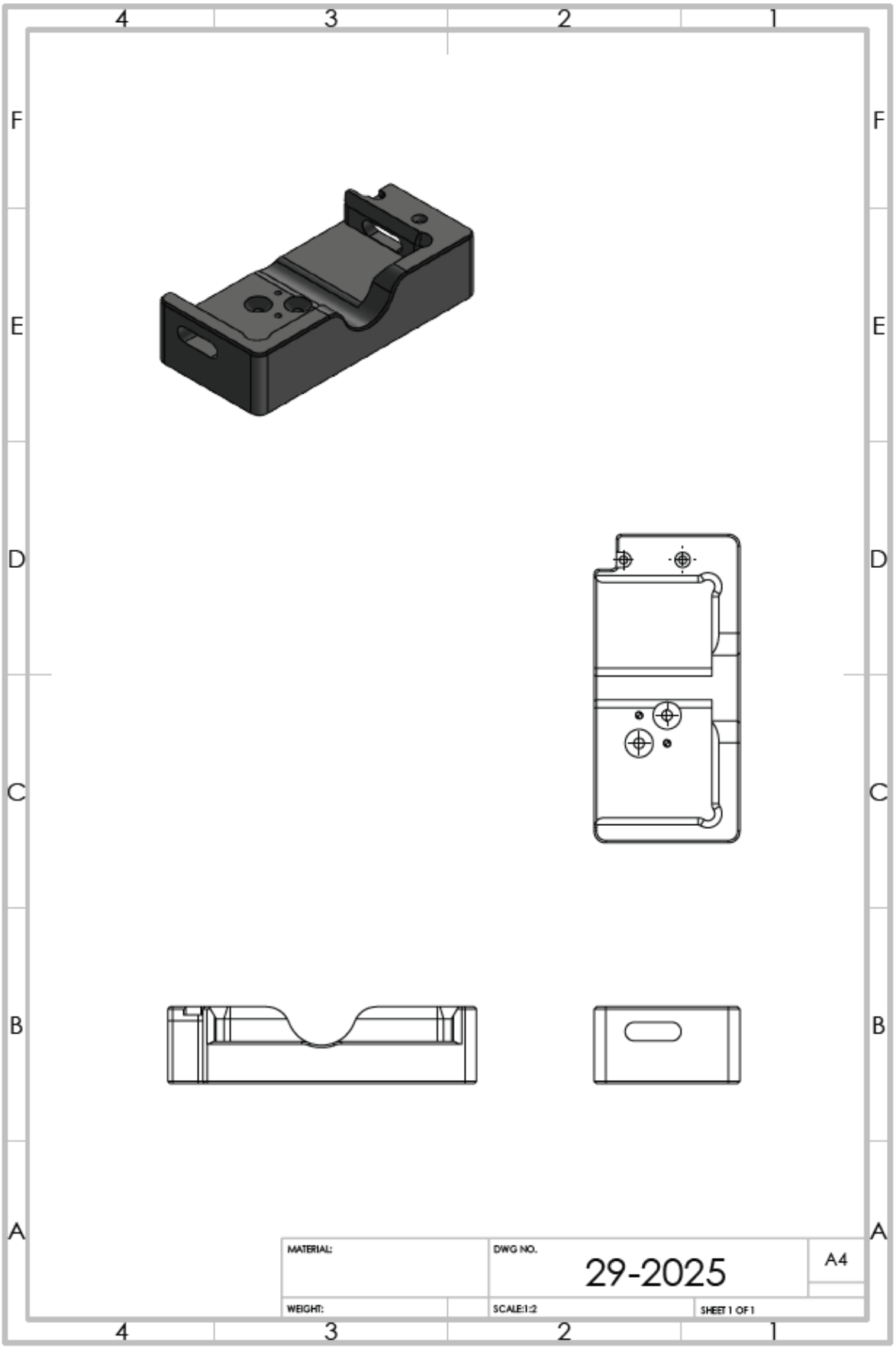


Figura 81 – Vistas simplificadas da peça de fabrico com referência 29.

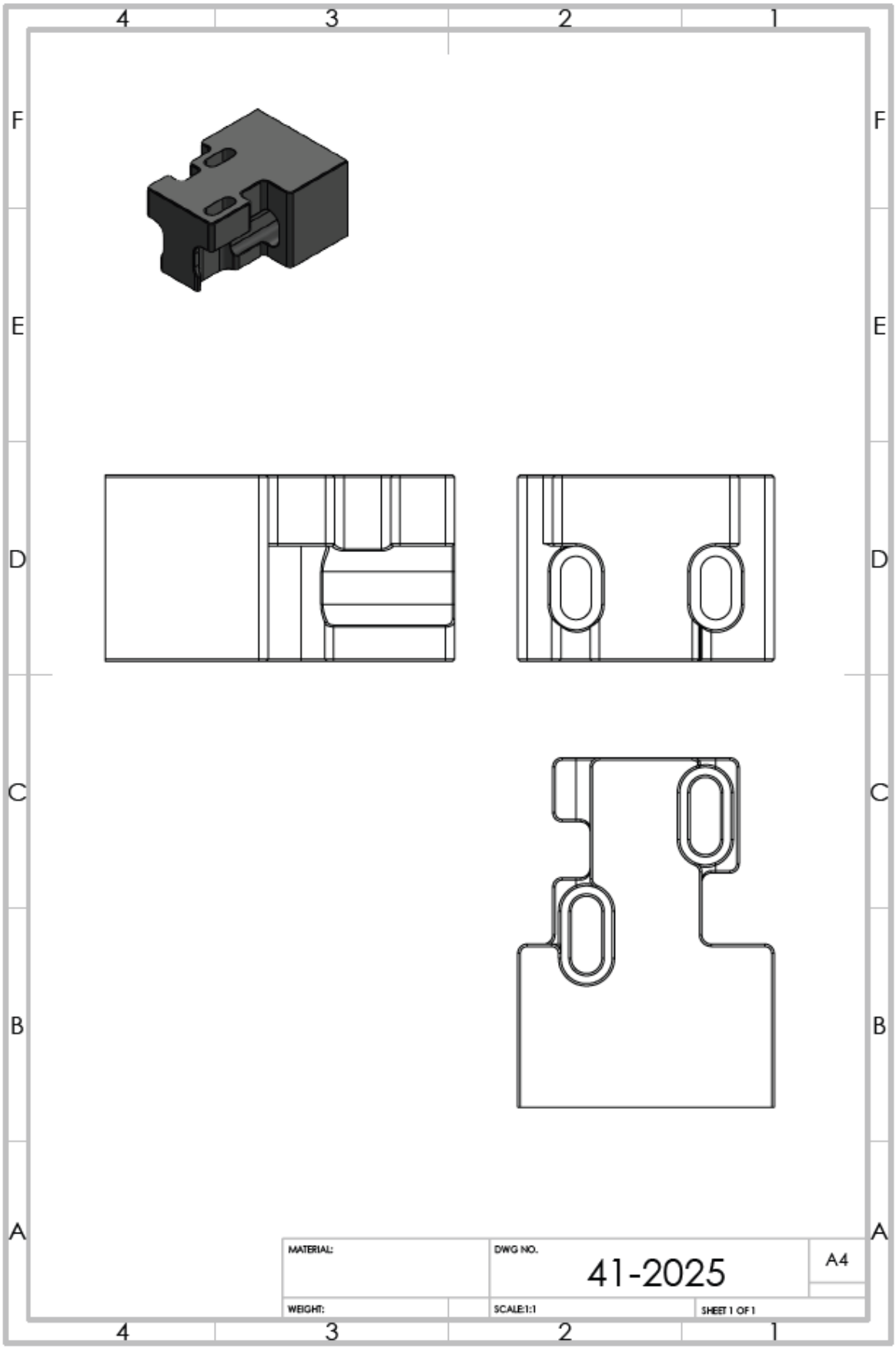


Figura 82 - Vistas simplificadas da peça de fabrico com referência 41.

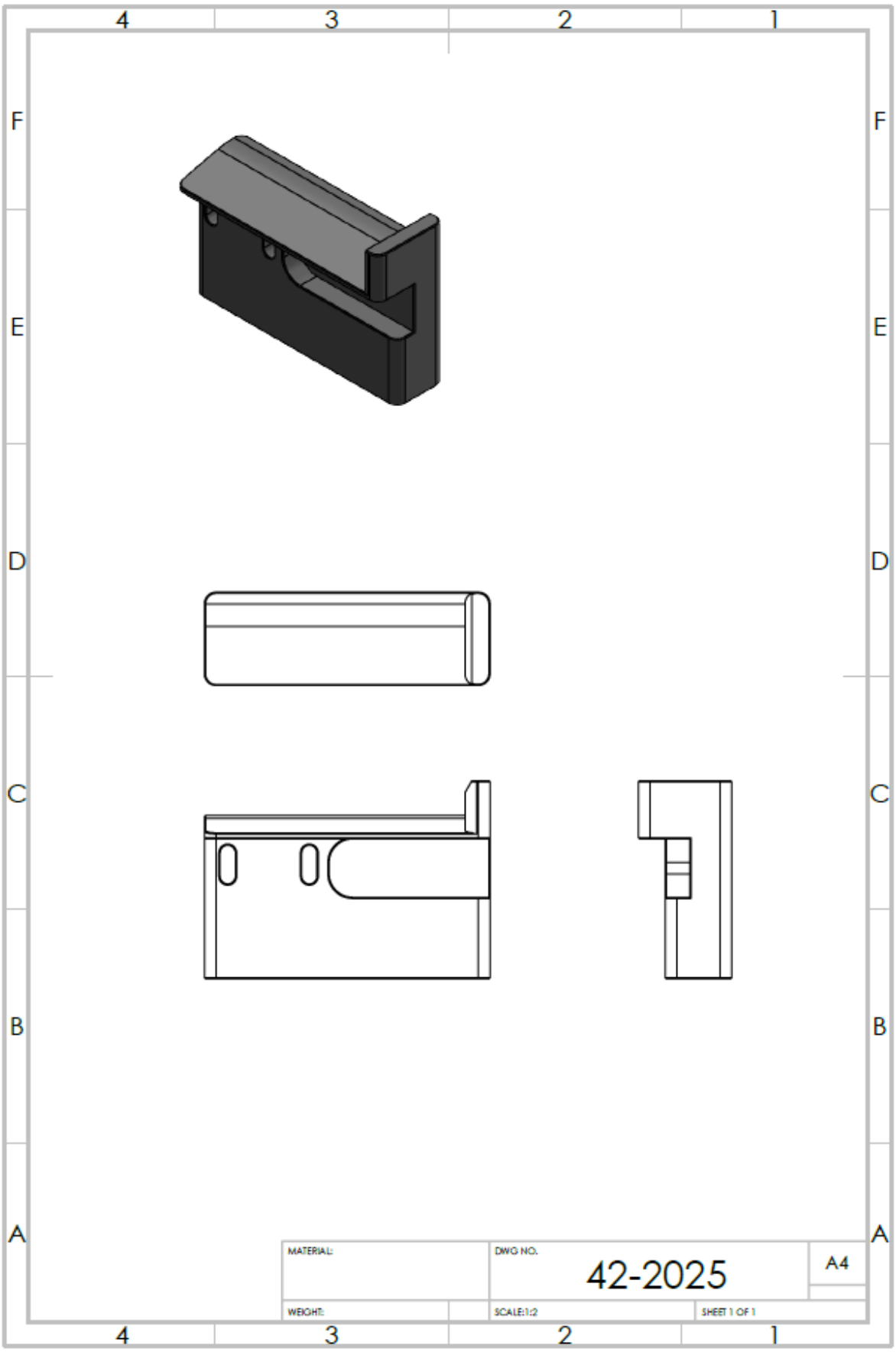


Figura 83 - Vistas simplificadas da peça de fabrico com referência 42.

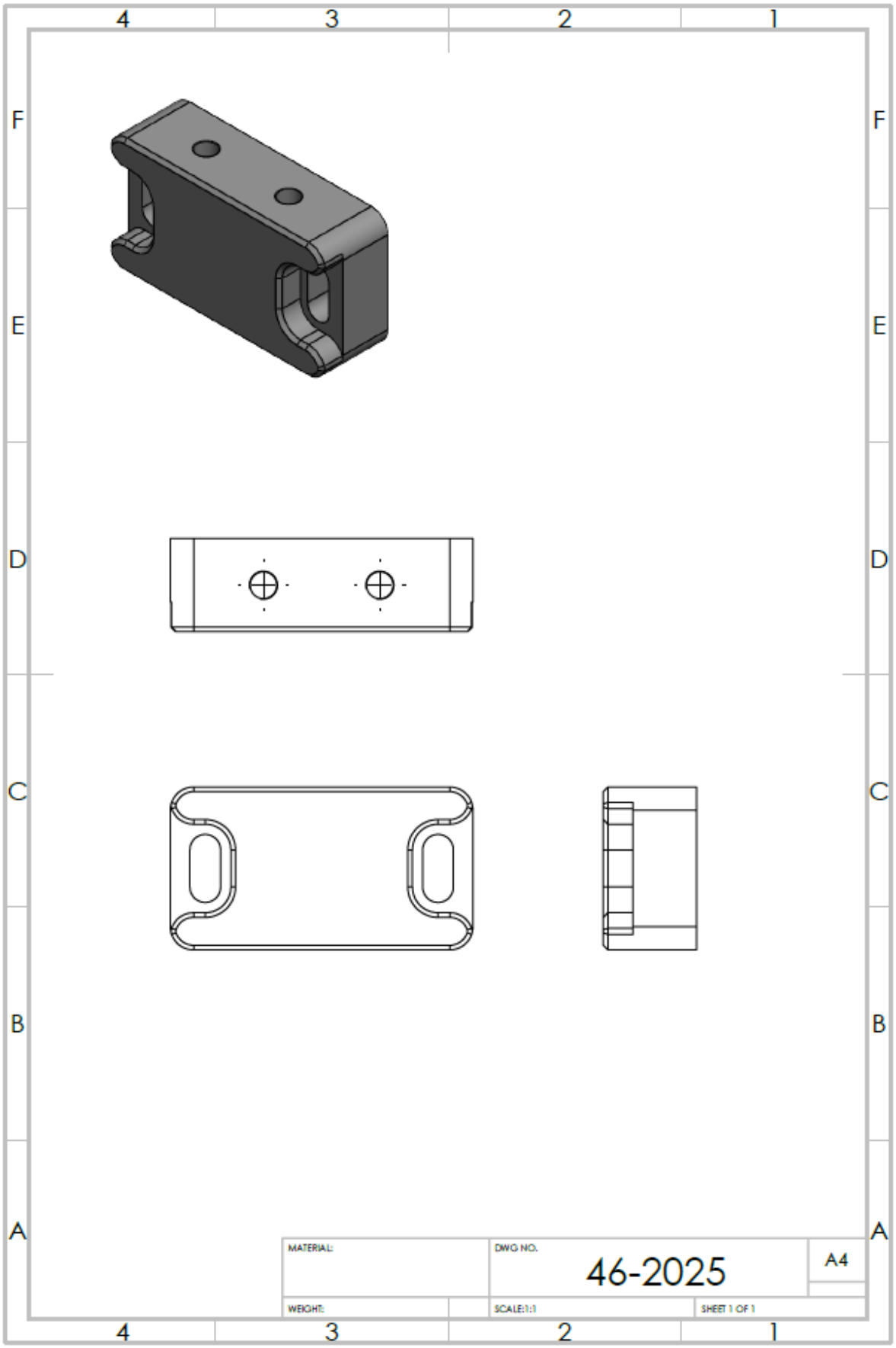


Figura 84 - Vistas simplificadas da peça de fabrico com referência 46.

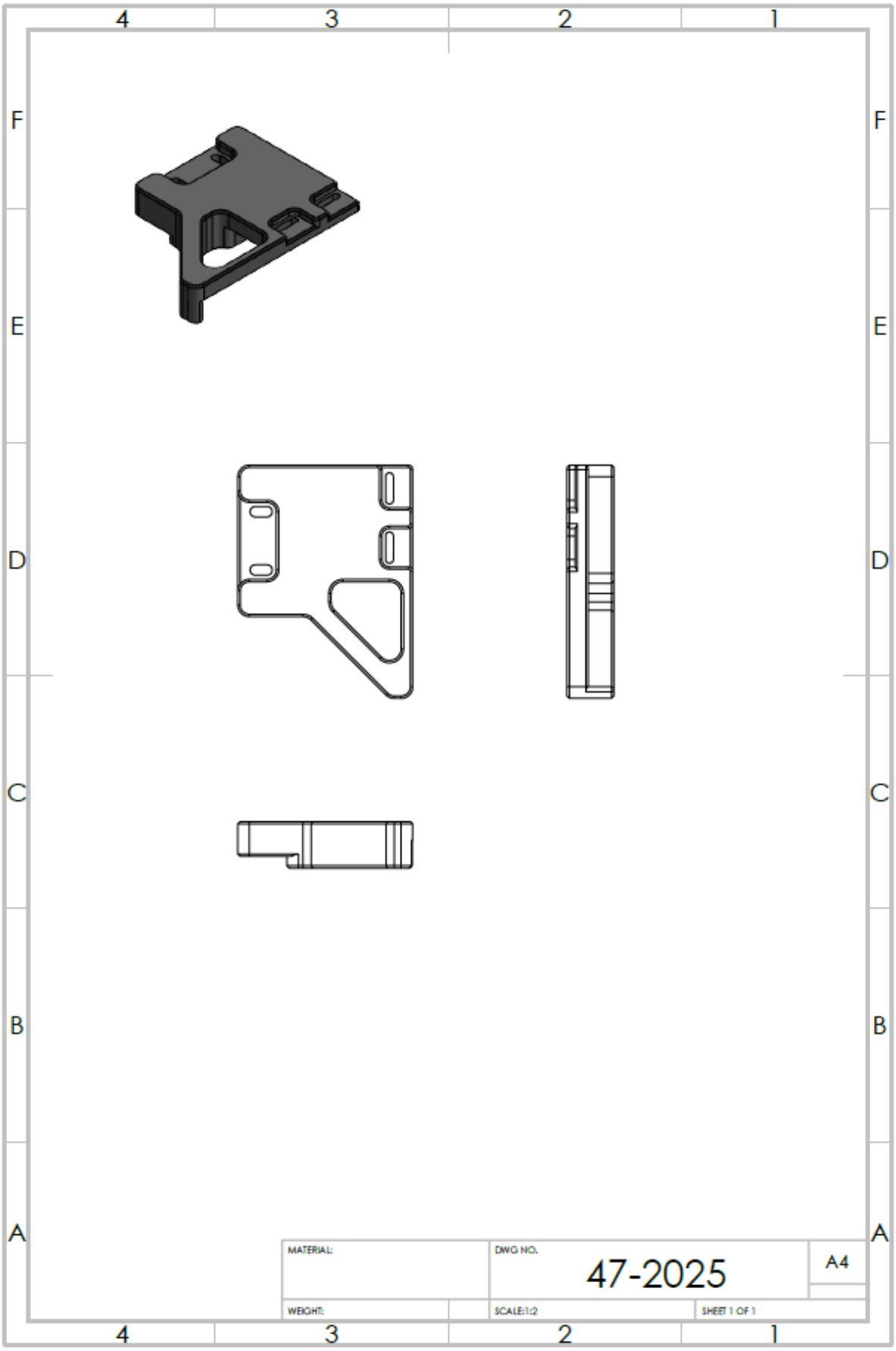


Figura 85 - Vistas simplificadas da peça de fabrico com referência 47.

APÊNDICE C

Desc. Técnica	Nomenclatura	E/S	Modelo/Características técnicas	Fabricante	Descrição
-FQ1	Barreira de Segurança	-	C2C-SA03030A10000	Sick	Emissor
-FQ2	Barreira de Segurança	-	C2C-EA03030A10000	Sick	Recetor
-PH1	HMI	-	6AV2 123-2GB03-0AX0	Siemens	HMI KTP700 Basic PN
-KC2	PLC	-	6ES7 214-1BE30-0XB0	Siemens	CPU S7-1214C AC/DC/RLY
-KC2.1	Módulo de expansão	-	6ES7 223-1BL32-0XB0	Siemens	Módulo de E/S SM1223 DC/DC
-KC3	Ethernet switch	-	RS-PRO 144-8673	RS-PRO	Ethernet switch c/ 5 portas
-KE1	Relé de segurança	-	PNOZ X2,1	Pilz	Relé de segurança Pilz
-B1	Sensor fotoelétrico	I0.3	E3Z-D82	Omron	Sensores de presença no ninho
-B2	Sensor fotoelétrico	I0.4	E3Z-D82	Omron	Sensores de presença no conveyor
-BVB0	Sensor magnético	I1.4	D-A93	SMC	Sensor magnético para cil. Pneumático
-BVB1	Sensor magnético	I1.5	D-A93	SMC	Sensor magnético para cil. Pneumático
-BVA0	Sensor magnético	I1.2	D-A73	SMC	Sensor magnético para cil. Pneumático
-BVA1	Sensor magnético	I1.3	D-A73	SMC	Sensor magnético para cil. Pneumático
-BVC0	Sensor magnético	I2.0	D-A73	SMC	Sensor magnético para cil. Pneumático
-BVC1	Sensor magnético	I2.1	D-A73	SMC	Sensor magnético para cil. Pneumático
-BVD0	Sensor magnético	I2.2	D-A73	SMC	Sensor magnético para cil. Pneumático
-BVD1	Sensor magnético	I2.3	D-A73	SMC	Sensor magnético para cil. Pneumático
-M1	Motor elétrico	Q0.0	Un=24 V; In=2,9 A	-	Motor DC
-VA0	Válvula 5/2	Q0.4	SY5220-5LOU-C6	SMC	Válvula SMC 5/2
-VA1		Q0.3			
-VB0	Válvula 5/2	Q0.6	SY5220-5LOU-C6	SMC	Válvula SMC 5/2
-VB1		Q0.5			
-VC1	Válvula 5/2	Q0.7	SY5120-5LOU-C6F	SMC	Válvula SMC 5/2
-VD1	Válvula 5/2	Q1.0	SY5120-5LOU-C6F	SMC	Válvula SMC 5/2
-S1	Botão Start	I0.0	XB4BA31	Schneider	Botão de pressão
-S2	Botão Stop	I0.1	XB4BA21	Schneider	Botão de pressão

-SE1	Cogumelo de Emergência	-	XB4BS8444	Schneider	Cogumelo de emergência
-PF1	Sinalizador de cor Verde	Q2.0	XVBC2B3	Schneider	Sinalizador de luz verde
-PF2	Sinalizador de cor Vermelha	Q2.1	XVBC2B4	Schneider	Sinalizador de luz vermelha

Tabela 4 - *Bill of Materials* e respetiva identificação de entradas e saídas.

APÊNDICE D

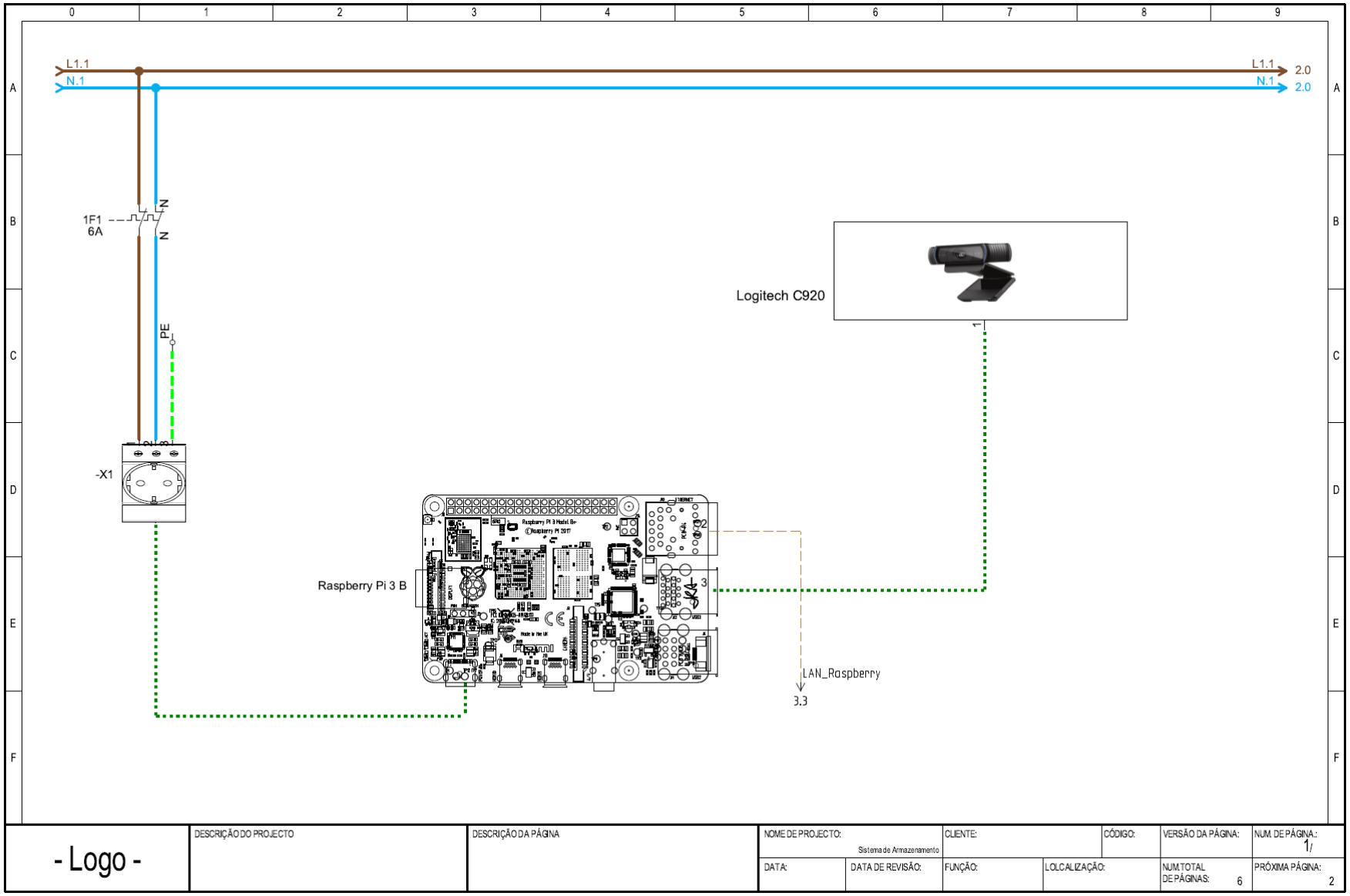


Figura 86 - Página 1 adicionada ao esquema elétrico existente.

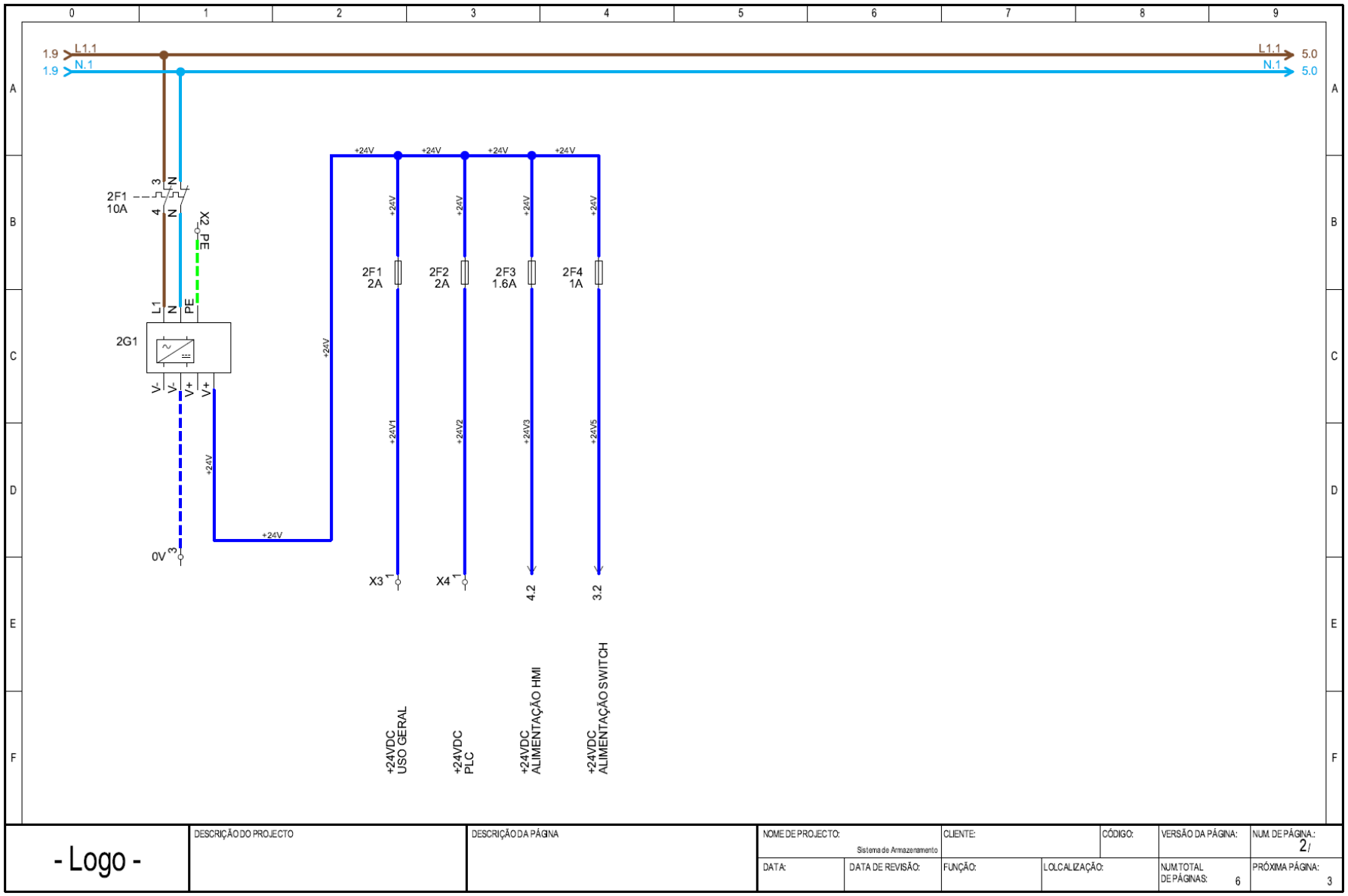


Figura 87 - Página 2 adicionada ao esquema elétrico existente.

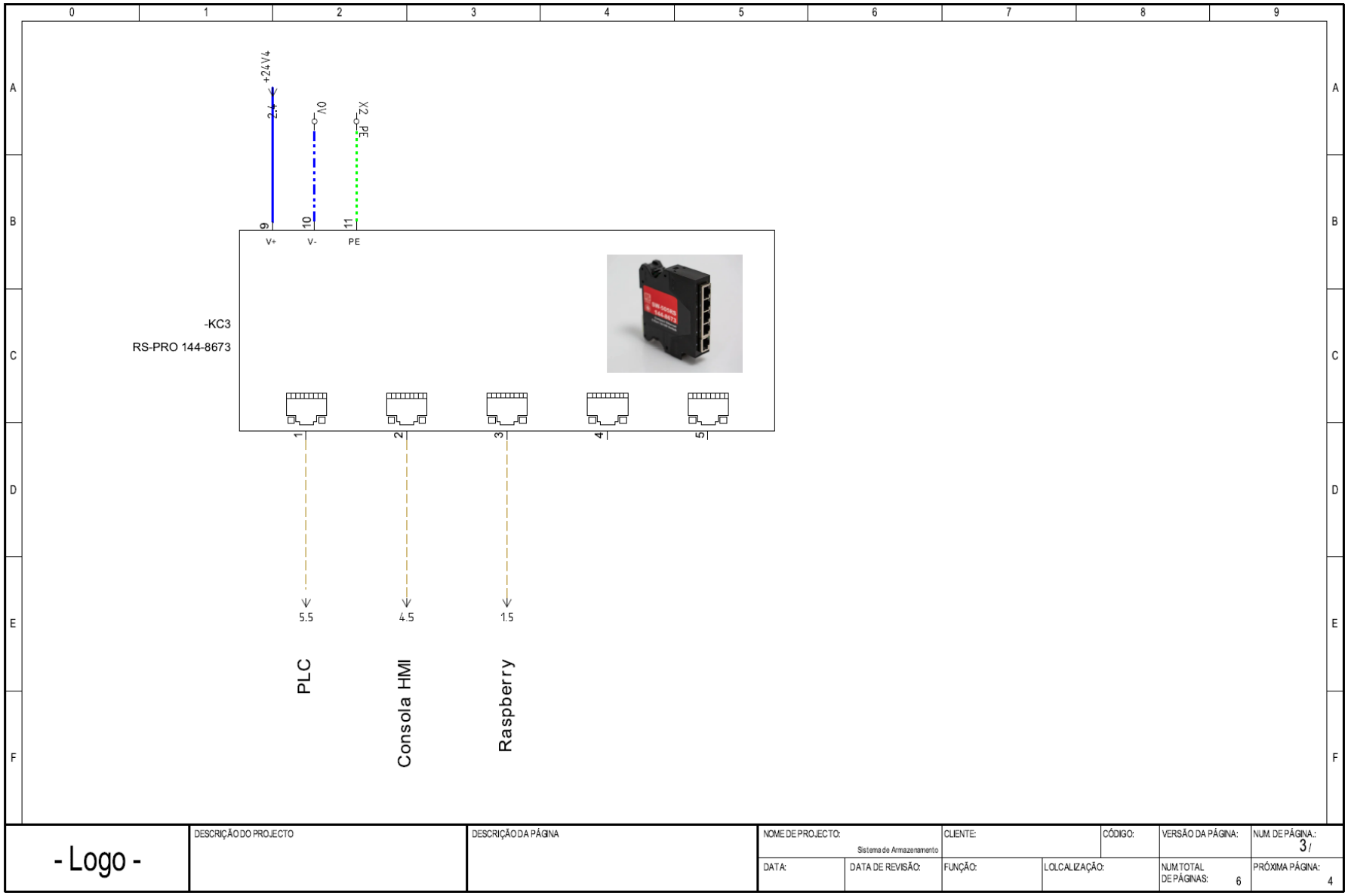


Figura 88 - Página 3 adicionada ao esquema elétrico existente.

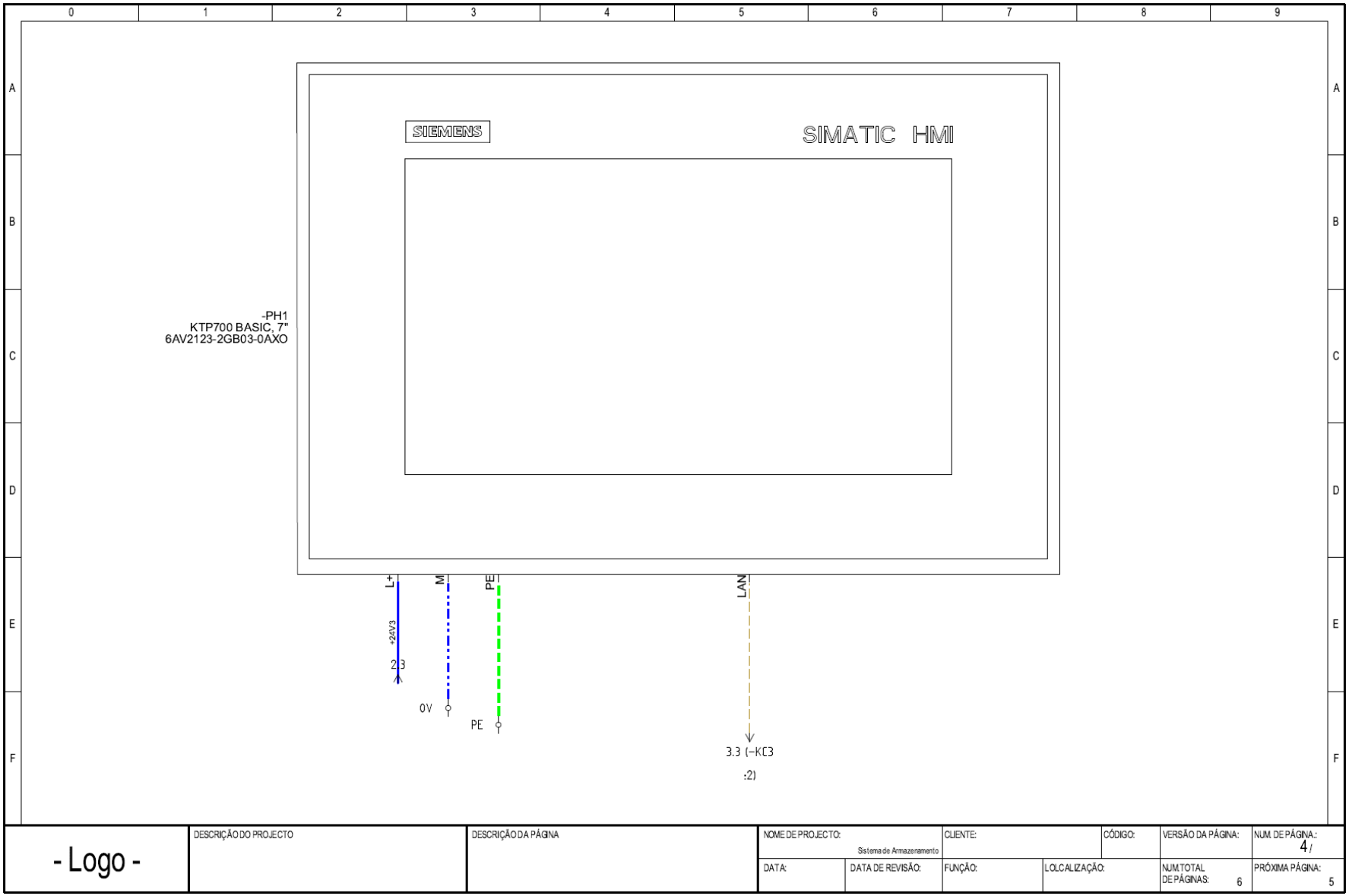


Figura 89 - Página 4 adicionada ao esquema elétrico existente.

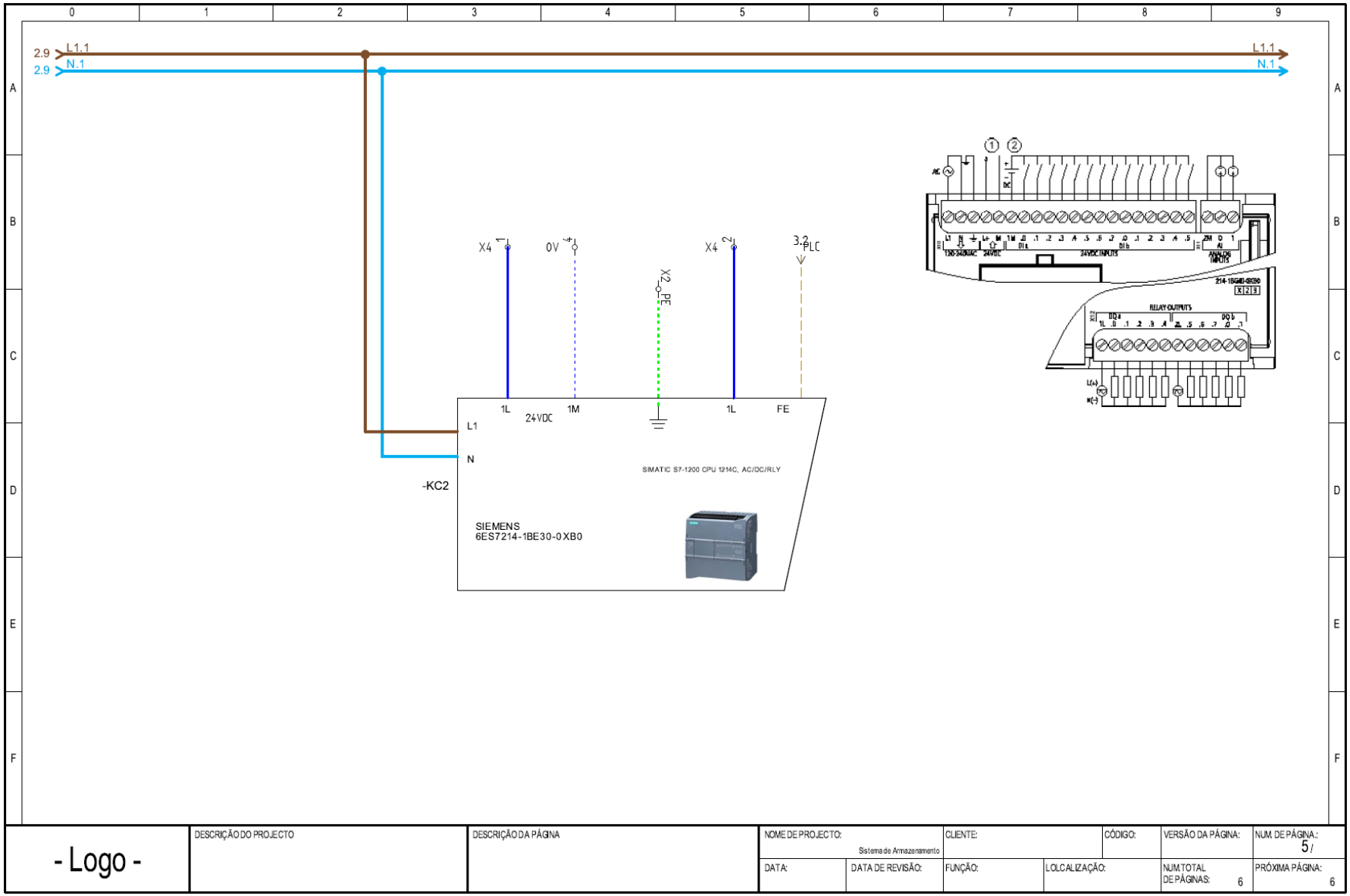


Figura 90 - Página 5 adicionada ao esquema elétrico existente.

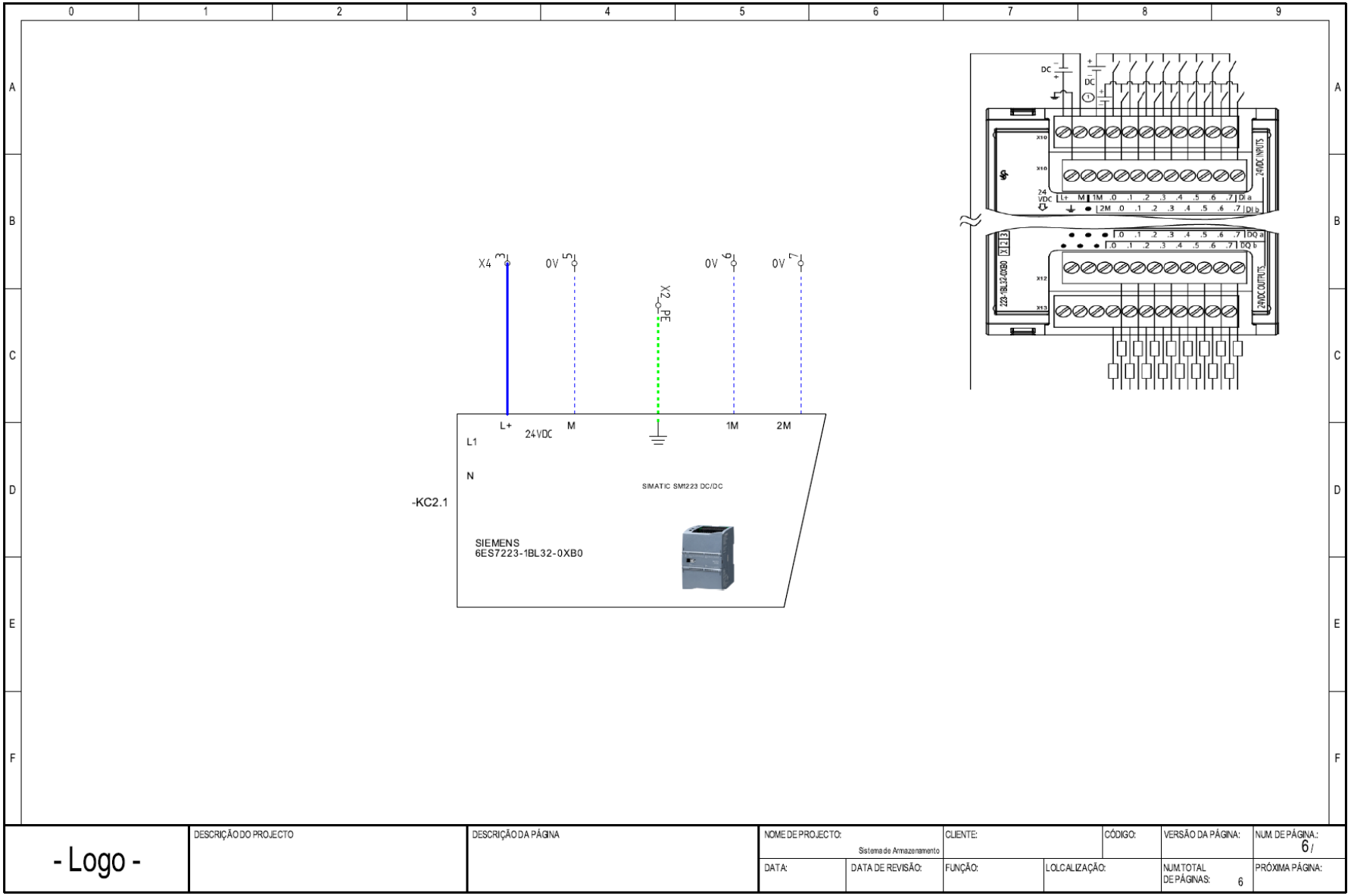


Figura 91 - Página 6 adicionada ao esquema elétrico existente.

APÊNDICE E

Peça Teste: A_NOK	Número de pixeis aceitáveis por região de interesse									
	Canto A		Canto B		Canto C		Canto D		Binário	
	Valor	Range	Valor	Range	Valor	Range	Valor	Range	Valor	Resultado
Peça A_OK	438	900 - 1100	106	950 - 1150	184	1100 - 1300	556	1650 - 1850	0	Peça A
Peça A_NOK	438	50 - 650	106	50 - 650	184	50 - 650	556	50 - 650	0	Peça A
Peça B_OK	7	1000 - 1200	370	1200 - 1400	3374	1100 - 1300	242	1100 - 1300	0	Peça A
Peça B_NOK	258	10 - 650	310	50 - 650	2213	50 - 650	0	50 - 650	0	Peça A

Tabela 5 - Resultados por região de interesse para a peça A NOK.

Peça Teste: B_OK	Número de pixeis aceitáveis por região de interesse									
	Canto A		Canto B		Canto C		Canto D		Binário	
	Valor	Range	Valor	Range	Valor	Range	Valor	Range	Valor	Resultado
Peça A_OK	4900	900 - 1100	4321	950 - 1150	30	1100 - 1300	4783	1650 - 1850	3742	Peça B
Peça A_NOK	4900	50 - 650	4321	50 - 650	30	50 - 650	4783	50 - 650	3742	Peça B
Peça B_OK	1092	1000 - 1200	1287	1200 - 1400	1187	1100 - 1300	1219	1100 - 1300	3742	Peça B
Peça B_NOK	1092	10 - 650	1287	50 - 650	1187	50 - 650	1219	50 - 650	3742	Peça B

Tabela 6 - Resultados por região de interesse para a peça B OK.

Peça Teste: B_NOK	Número de pixeis aceitáveis por região de interesse									
	Canto A		Canto B		Canto C		Canto D		Binário	
	Valor	Range	Valor	Range	Valor	Range	Valor	Range	Valor	Resultado
Peça A_OK	4900	900 - 1100	4688	950 - 1150	1580	1100 - 1300	4900	1650 - 1850	4900	Peça B
Peça A_NOK	4900	50 - 650	4688	50 - 650	1580	50 - 650	4900	50 - 650	4900	Peça B
Peça B_OK	19	1000 - 1200	481	1200 - 1400	577	1100 - 1300	404	1100 - 1300	4900	Peça B
Peça B_NOK	19	10 - 650	481	50 - 650	577	50 - 650	404	50 - 650	4900	Peça B

Tabela 7 - Resultados por região de interesse para a peça B NOK.

APÊNDICE F

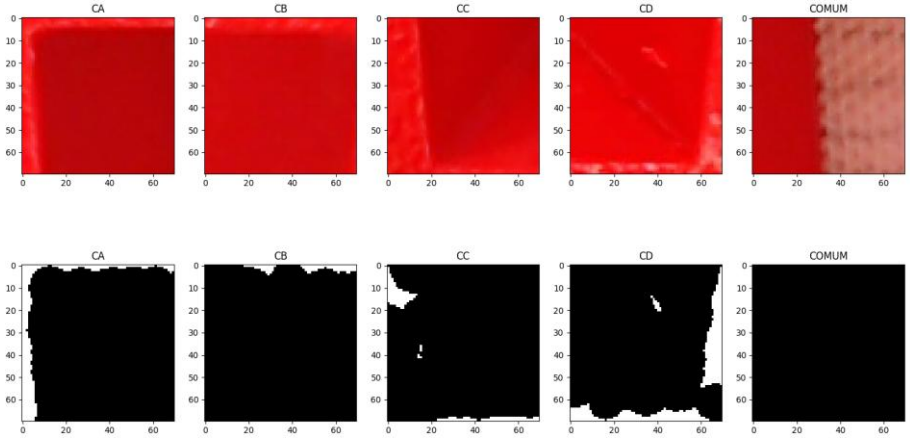


Figura 92 - ROI e segmentação binária - Análise da peça A NOK através da ROI de peças A.

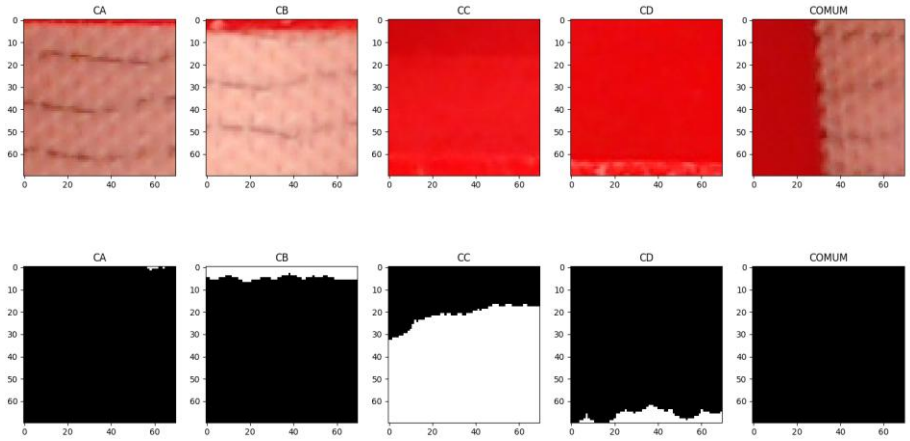


Figura 93 - ROI e segmentação binária - Análise da peça A NOK através da ROI de peças B.

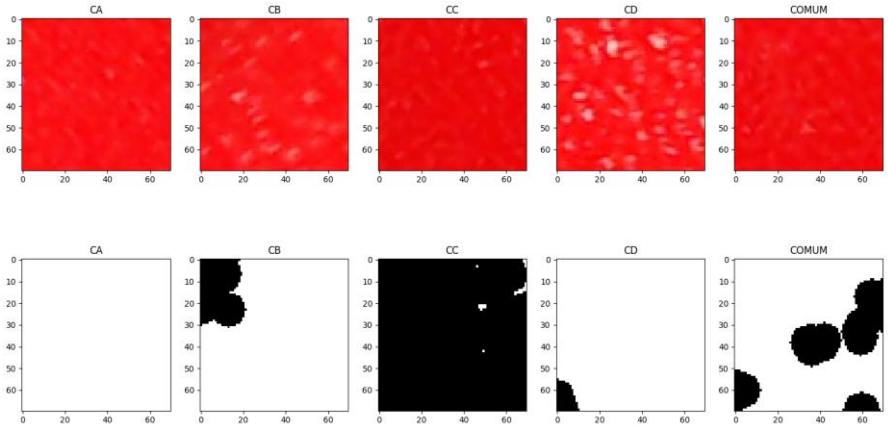


Figura 94 - ROI e segmentação binária - Análise da peça B OK através da ROI de peças A.

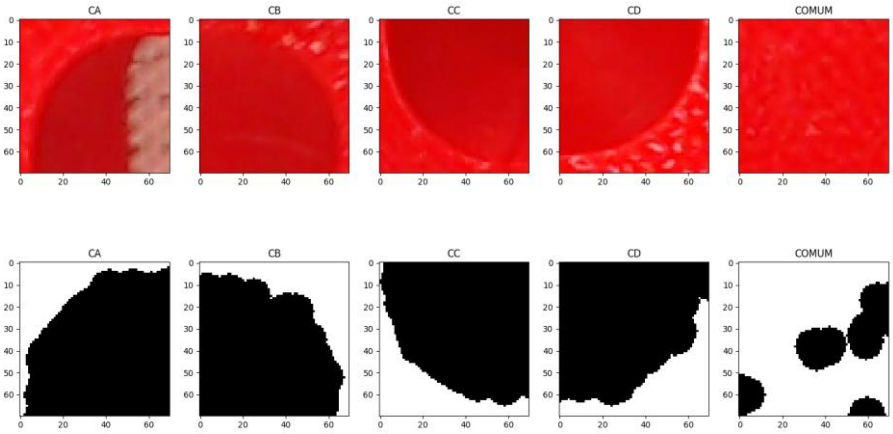


Figura 95 - ROI e segmentação binária - Análise da peça B OK através da ROI de peças B.

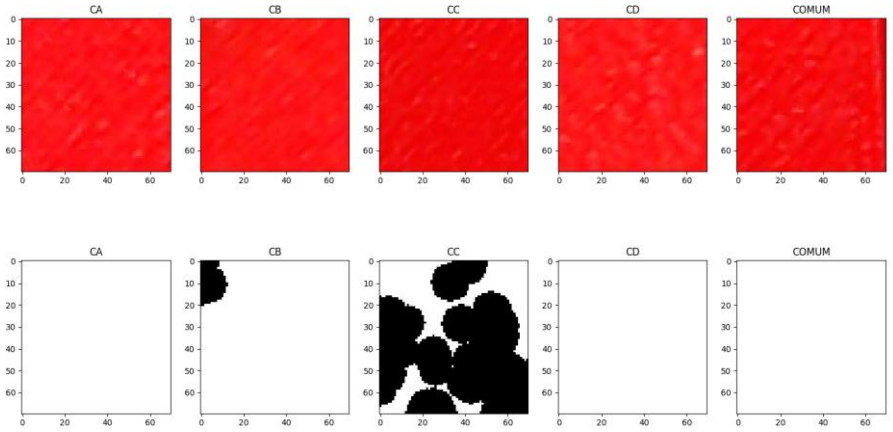


Figura 96 - ROI e segmentação binária - Análise da peça B NOK através da ROI de peças A.

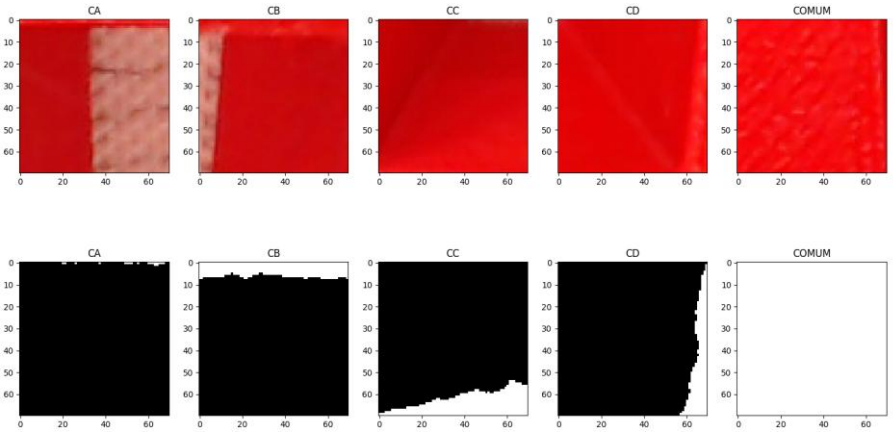


Figura 97 - ROI e segmentação binária - Análise da peça B NOK através da ROI de peças B.