PUC Minas

A B A K Ó S

Instituto de Ciências Exatas e Informática

Módulo de Software para Detecção de Falhas na Pista Guia de um Veículo Autoguiado*

Software Module for an Automated Guided Vehicle Lane Wear Detection

Fernanda Elias Couto[1]
Gabriel Araujo da Silva[2]
Jaiane Nunes da Silva[3]
Jéssica Miranda da Costa[4]
Júlia Vitória de Almeida Oliveira[5]
Wallace Pereira Neves dos Reis[6]

Resumo

Veículos Autoguiados, ou AGVs (*Automated Guided Vehicles*), seguidores de linha têm sua operação baseada na contínua detecção da faixa guia usada como referência. Em ambientes industriais, as faixas guia podem sofrer desgaste ao longo do tempo, comprometendo a navegação e exigindo manutenção frequente. O presente trabalho propõe um módulo de software para a detecção automática de falhas na faixa guia de AGVs utilizando processamento de imagens. Os experimentos conduzidos com vídeos de diferentes pistas demonstraram que o módulo é capaz de identificar falhas na faixa guia com alta exatidão, distinguindo entre pistas saudáveis e degradadas. Os resultados indicam que a solução pode reduzir a necessidade de inspeção manual e minimizar interrupções no transporte de materiais, contribuindo para uma maior eficiência operacional em ambientes industriais.

Palavras-chave: Processamento de imagem. Identificação de padrão. Veículos autoguiados.

*Submetido em 11/08/2023 - Aceito em 07/06/2025

[1]Curso Técnico em Automação Industrial, IFRJ - Volta Redonda, Brasil- fernanda.eliasct@gmail.com.
[2]Curso Técnico em Automação Industrial, IFRJ - Volta Redonda, Brasil- gabriel.dasilvaa780@gmail.com.
[3]Curso Técnico em Automação Industrial, IFRJ - Volta Redonda, Brasil- silvajaiane212@gmail.com.
[4]Curso Técnico em Automação Industrial, IFRJ - Volta Redonda, Brasil- jessicamir009@gmail.com.
[5]Curso Técnico em Automação Industrial, IFRJ - Volta Redonda, Brasil- julia.vitoria.almeida21@gmail.com.
[6]Doutor em Ciência da Computação, Mestre em Engenharia Elétrica, Docente do IFRJ - Volta Redonda, Brasil- wallace.reis@ifrj.edu.br.

Abstract

Automated Guided Vehicles (AGVs) that follow a line operate based on the continuous detection of a guide strip or track used as a reference. In industrial environments, guide tracks may deteriorate over time, compromising navigation and requiring frequent maintenance. This study proposes a software module for the automatic detection of faults in AGV guide tracks using image processing. Experiments conducted with videos of different tracks demonstrated that the module can accurately identify faults in the guide track, distinguishing between healthy and degraded tracks. The results indicate that the solution can reduce the need for manual inspection and minimize disruptions in material transport, contributing to greater operational efficiency in industrial environments.

Keywords: Image processing. Pattern identification. Automated guided vehicles.

1 INTRODUÇÃO

No ano de 1953, foi criado o primeiro veículo autoguiado, do Inglês *Automated Guided Vehicle* (AGV), que funcionava autonomamente em caminhos pré-definidos. As fábricas, no final dessa mesma década, implementaram a utilização desse sistema, o que continuou ocorrendo até a atualidade, onde ele é muito empregado em indústrias de automóveis e de distribuição de mercadorias. Seu objetivo é fazer o transporte de matérias-primas, produtos e informações, fazendo atividades repetitivas e difíceis de serem realizadas por humanos (Almeida, 2016).

Esses veículos contribuem para a segurança nos ambientes industriais, bem como na preservação das mercadorias produzidas e na rapidez, levando ao aumento da produção e à redução dos custos, principalmente aqueles relacionados à mão de obra humana. Tais benefícios refletem no crescimento do mercado de AGVs, que é estimado, para o ano de 2025, alcançar 7,3 bilhões de dólares (Monteiro, 2018).

Eles têm sido uma das principais ferramentas nos Sistemas de Fabricação Flexíveis. Tais sistemas de transporte são responsáveis por conectar as máquinas e os robôs envolvidos nos processos industriais a outros setores como os de montagem e de armazéns, reunindo as informações dos materiais enviados às máquinas e dos produtos finais em um computador principal. Sendo assim, é um sistema que monitora e controla o uso de materiais na produção, auxiliando na diminuição do consumo exacerbado da matéria–prima (Dreger, 2001).

Dentro da lógica dos Sistemas Flexíveis de Fabricação, Groover (2008) aborda que "o manuseio de materiais deve ser realizado de maneira segura, eficiente, com baixo custo e em tempo hábil, precisamente e sem danos aos materiais". Optando pela mão de obra humana para essa atividade, há um alto custo com funcionários que não agregam valor ao produto, além de não existir a garantia de segurança a operação, já que os humanos são suscetíveis a erros que podem causar acidentes e danos materiais.

De acordo Brito et al. (2020), uma pesquisa identificou que a indústria tende a optar por mão de obra que agrega valor no produto final, o que não ocorre na etapa de transporte, por esse motivo, os AGVs são aplicados. Eles permitem que o custo que a indústria teria no deslocamento de materiais e produtos seja direcionado a uma outra etapa do processo que agrega valor no produto final.

Os veículos autoguiados são capazes de ofertar um manuseio de materiais da forma abordada por Groover (2008). Eles oferecem rapidez e eficiência por conseguirem transportar uma diversidade de produtos e materiais a vários pontos diferentes, não sendo limitado a um só percurso, como aconteceria se a indústria optasse por utilizar uma esteira para o transporte. Além disso, por serem automáticos, conseguem realizar trabalhos repetitivos sem pausa, alcançando regiões da indústria onde o acesso humano é difícil. Sendo assim, a mão de obra humana é dispensada, diminuindo os gastos e os danos materiais ocasionados às mercadorias, que são um grande problema quando o transporte é realizado por humanos (Tonin, 2018).

Por oferecerem tantas vantagens, os AGVs possuem muitos ambientes de aplicação. Os mais conhecidos são a indústria automotiva, portos e linhas de montagem. Nos portos, eles são utilizados no transporte de contêineres durante o embarque e desembarque dos navios, o que é

mais eficiente porque não depende dos veículos portuários para fazer essa tarefa. A implementação desse recurso da automação na logística de carga e descarga de navios de contêineres tem sido observada, por exemplo, nos portos de Singapura e Xangai (Brito et al., 2020).

Dentre os tipos de AGVs, há os seguidores de linha, cujo percurso é demarcado no piso por uma faixa de cor contrastante. O veículo se desloca apenas por cima dessa faixa. No caso de detecção ótica ou por imagem, os veículos possuem sensores responsáveis por identificar a diferença entre o piso e a faixa guia e utilizam algoritmos para reconhecimento de borda, o que permite a navegação tendo a faixa como referência. Por dependerem da identificação da faixa, ela deve ser mantida sob manutenções constantes, que permitam que o AGV distingua o piso e pista guia delimitada pela faixa (Bandinelli, 2018). A Figura 1 mostra o AGV utilizado para implementação e para validação da proposta. A estrutura da câmera possui vedação da iluminação do ambiente. Essa caixa que envolve a câmera fica localizada no centro do veículo, rente ao chão e possui iluminação próxima, usando fitas de LED, de forma a mitigar a influência da iluminação do ambiente.

Figura 1 – Veículo autoguiado utilizado como base para o desenvolvimento e testes do módulo de software

Fonte: autoria própria.

Caso essa manutenção não seja realizada, o acúmulo de sujeiras, a queda de objetos sobre a linha, ou o desgaste pela própria movimentação de veículos e cargas, pode impedir que o AGV de identifique a faixa corretamente, prejudicando seu desempenho e afetando sua operação (Nascimento et al., 2019). Para Santos (2013), esse problema se torna um grande empecilho para uso desse tipo de veículo, já que o processo precisa ser parado para que haja a manutenção da faixa danificada, afetando diretamente a produtividade. Em empresas que possuem muitos AGVs, caso uma linha fique inoperante durante 1 hora, isso pode representar um grande prejuízo. De acordo com Belo (2016), se algo ocasiona dificuldades na chegada das matérias-primas de maneira rápida e na quantidade necessária para suprir as necessidades do processo, o custo de operação aumenta significativamente e há uma diminuição na capacidade de produção da indústria. Ainda, considerando o uso desse veículo em portos, esse tempo fora de operação pode retratar um atraso significativo no tempo de carga ou descarga de uma embarcação. Dessa forma, situações como essas podem diminuir o retorno de investimento esperado para esse tipo de tecnologia.

A identificação de falhas na pista de AGVs é um problema relevante tanto do ponto de

vista prático quanto teórico. Do ponto de vista prático, falhas não detectadas podem comprometer a continuidade das operações, gerando impactos na produtividade de sistemas automatizados e até acidentes. Já do ponto de vista teórico, o desenvolvimento de técnicas de visão computacional para AGVs apresenta desafios como variações de iluminação, diferentes condições do piso, degradação progressiva da pista que envolvem a necessidade de pesquisa em algoritmos que sejam capazes de abarcar diferentes cenários operacionais. Dessa forma, a proposta contribui para o aumento da confiabilidade da navegação dos AGVs atuando como uma ferramenta de manutenção, além de propor uma aplicação de processamento de imagem para AGVs não identificada pelos autores na literatura.

Este trabalho propõe um módulo de software embarcado para AGVs, baseado em processamento de imagem, capaz de identificar falhas na pista por meio de câmera instalada no veículo, conforme abordagem de Reis (2023). O módulo visa automatizar a fiscalização da faixa de rodagem em áreas de difícil acesso ou com baixa frequência de inspeção humana, reduzindo a necessidade de intervenção manual e os custos operacionais.

O desenvolvimento do módulo segue princípios de engenharia de software, com foco em independência e reusabilidade. Os objetivos específicos incluem: (i) estruturar o código com alta coesão e baixo acoplamento; (ii) documentar o serviço com UML, facilitando sua compreensão e reaproveitamento; (iii) adotar a arquitetura orientada a serviços (SOA) para promover escalabilidade e flexibilidade; e (iv) aplicar o Padrão Fachada para garantir uma interface eficiente com os demais subsistemas do AGV.

Para desenvolver o módulo de software proposto, será empregada uma abordagem baseada em processamento de imagens, utilizando técnicas de segmentação e detecção de contornos para identificar falhas na faixa guia, tendo como padrão um valor definido para a faixa saudável. Os testes experimentais realizados visam validar a precisão e a confiabilidade do módulo na identificação de falhas sob diferentes condições operacionais, representadas por diferentes vídeos. Espera-se que essa solução contribua para a automação do monitoramento da faixa guia, reduzindo a necessidade de intervenção humana e aumentando a eficiência dos processos industriais que fazem uso de AGVs.

O presente artigo está dividido nas seções descritas a seguir. Em Trabalhos Correlatos são apresentadas pesquisas recentes sobre sistemas de visão computacional de veículos autoguiados relacionados com o trabalho proposto. Na seção Proposta é descrito o funcionamento do módulo desenvolvido e a composição de sua construção. Na seção Resultados e Discussão são apresentados os testes realizados com o software e as análises o seu desempenho na identificação das falhas. Por fim, a seção Conclusão apresenta as considerações finais sobre a aplicação do software e os trabalhos futuros identificados.

2 TRABALHOS CORRELATOS

Reis e Morandin Junior (2021) analisaram o uso de sensores e de técnicas de detecção usadas na visão computacional e sua influência no controle de posicionamento dos AGVs. Se-

guindo a divisão do avanço tecnológico desses veículos em quatro períodos, Ullrich (2015), os autores destacam que, na atualidade, ainda estão sob desenvolvimento sistemas com técnicas de detecção mais modernas e menos custosas, o que demonstra que a visão computacional ainda é um ramo em avanço. O motivo disso ocorrer é que diversos fatores devem ser considerados na construção de novas tecnologias para a visão de máquinas, como a influência de ruídos nos sensores utilizados para o posicionamento dos AGVs. Esses têm uma gama de origens, como a sujeira, a iluminação e o próprio sensor usado. No caso do uso de câmeras como sensor, o que está em crescimento dentro das indústrias, ainda deve-se atentar para o peso carregado pelo veículo, o que pode causar vibrações que comprometem a captura de quadros da imagem, ou *frames*. Como resultado de sua pesquisa, eles observaram a importância dos estudos recentes e também dos futuros para a conexão de sistema instalados em AGVs antigos à indústria 4.0 através do surgimento de sensores e técnicas que atendam aos padrões e ao mercado.

Ainda sobre a influência dos sensores no sistema de funcionamento do AGV, Oliveira et al. (2019) abordam em sua pesquisa a utilização de câmeras USB como detectores para o sistema de controle dos veículos autoguiados. Os sensores mostram-se tão importantes nesse cenário devido a sua atuação no controle de posicionamento desses veículos, os tornando mais precisos e mais estáveis, ofertando maior segurança ao transporte de mercadorias. Entretanto, os autores observam que é preciso atentar-se para como o tratamento e o processamento da imagem influenciam o sistema de funcionamento do AGV, já que podem ocasionar degradação do sistema de controle. Para a análise do desempenho do AGV e sua exatidão de posicionamento sob diferentes condições, usando uma câmera USB, foi desenvolvido um protótipo de AGV com tração omnidirecional ensaiado em uma pista com formato de oito. Os fatores da câmera modificados ao longo do teste incluíram as configurações do processamento de imagem, como o uso de diferentes filtros morfológicos e resoluções de imagem. Os resultados indicam que a câmera pode ser utilizada como detector de linha no sistema de controle de posição de um AGV, porém, deve-se estudar os efeitos dos parâmetros do sistema de processamento de imagem na malha de controle.

Ao buscar a aplicação de técnicas de visão computacional em veículos embarcados de menor custo, Girbés et al. (2010) criaram o projeto PISALA (Protótipo Industrial para Rastreamento Automático de Linhas com Visão Artificial), onde construíram um sistema de baixo custo para um seguidor de linha usando visão computacional e as vantagens da utilização de câmeras para a leitura das faixas guias em ambientes diversos. Para avaliar a robustez e precisão dos algoritmos, testes qualitativos e quantitativos foram realizados. Como resultado, obtiveram êxito na implementação do AGV seguidor de linha, apresentando bom desempenho. Entretanto, destacaram a necessidade de uma melhoria na leitura das imagens em lugares onde a faixa está danificada, com superexposição ou com falta de luz.

Zhang et al. (2022) também observam essa necessidade, direcionando os estudos para os problemas encontrados na detecção de faixas guias de navegação de um veículo autoguiado. Como as linhas estão presentes no ambiente, elas sofrem com a superexposição, que é identificada na câmera utilizada como sensor. O método de navegação visual é muito importante para

os sistemas industriais, principalmente devido ao menor custo, por isso, é preciso solucionar tal problema, que diminui a precisão na identificação da faixa e prejudica a orientação e movimentação do veículo. O método utilizado por eles para solucioná-lo foi um sistema de visão computacional combinando imagem em pintura e segmentação de imagem.

Em relação aos controladores CPU de veículos autoguiados que utilizam uma câmera como sensor para o sistema de visão computacional, Dewantoro et al. (2021) compararam em seus estudos o desempenho da Raspberry Pi e Jetson Nano. Tais controladores foram utilizados no processamento de imagem do AGV, onde um método foi empregado para detectar os contornos da pista de navegação presente no *frame* capturado pela webcam e usá-los como referência para a movimentação do robô.

A metodologia aplicada no estudo foi a construção de um seguidor de linha que usava separadamente a Raspberry PI e a Jetson Nano, submetidas ao mesmo sistema de processamento de imagem. Para a visão do veículo, é preciso que essas CPUs processem as imagens da webcam, controlando, ao mesmo tempo, a movimentação. Os resultados obtidos nos testes comprovaram que a Raspberry executou o algoritmo sem erros. O sucesso na identificação do percurso que o robô deve seguir está ligado diretamente a precisão na detecção das bordas da pista, o que a Raspberry conseguiu oferecer. Ela foi capaz de identificar 6 linhas distintas, dentre 8 tipos, com uma precisão de 100%. Além disso, obteve 98% de sucesso para completar as voltas da pista de teste. Concluiu-se, a partir disso, que a Raspberry funciona bem e já é aplicada como CPU para processamento de imagem e detecção de faixas guias de navegação de AGVs.

Como se pode observar, diversos autores abordam a utilização da Raspberry PI e de câmeras no sistema de visão computacional para a navegação dos AGVs, o que comprova que esses métodos têm sido usados recentemente e ainda estão sob desenvolvimento e pesquisa. É de destaque ainda o fato de muitos teóricos reconhecerem os problemas de leitura de imagem dos veículos autoguiados ocasionados por desgastes na faixa, mas os solucionarem de formas distintas da proposta nesse projeto, o que o torna inédito.

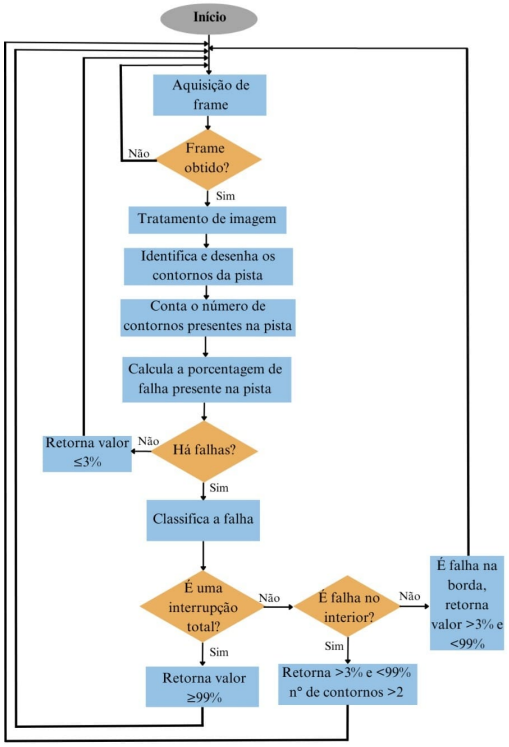
Existem pesquisas que tentam solucionar a problemática por meio da substituição da faixa guia, como é o caso dos sistemas filoguiados e os sistemas que integram a localização e mapeamento simultâneo (SLAM) nos veículos autoguiados. No entanto, o primeiro sistema depende da inserção de condutores elétricos no chão para a criação de um campo magnético a ser identificado por uma antena instalada no AGV, o que o faz não ser flexível, visto que para fazer a modificação da rota é preciso instalar novos condutores no chão de fábrica, aumentando o custo de implementação (Santos, 2013).

O segundo sistema substitui as faixas pelo mapeamento do ambiente e a localização do veículo no mesmo. Contudo, para ele ser aplicado é necessário alterar o sistema principal de navegação do veículo autoguiado, tendo um custo maior (Tonin, 2018). Tal fato representa uma desvantagem às indústrias que já utilizam os seguidores de linha e querem apenas solucionar as paradas repentinas de operação devido ao desgaste das faixas, sem influenciar no sistema de navegação já existente e sem precisar inserir novos sensores no veículo.

3 PROPOSTA

Quando o veículo autoguiado entrar em funcionamento, o módulo de software será iniciado. Conforme a Figura 2, após a inicialização do sistema, um *frame* será aquistado.

Figura 2 – Fluxograma de funcionamento geral do módulo de software



Fonte: autoria própria.

Caso não seja identificada a aquisição de um *frame* do vídeo, o processo retornará ao início e solicitará novamente a imagem. Caso o *frame* tenha sido obtido corretamente, ele passará pelo tratamento de imagem. O processamento inicial, indicado no fluxograma como "Tratamento de Imagem", converte o quadro para escala de cinza, aplica um filtro de suavização (*GaussianBlur*) para reduzir ruídos, com elemento estruturante quadrado de tamanho 9×9 pixels, e realiza a binarização da imagem, com limiar configurado para o valor 100, na escala de 0 a 255, como representado na Equação 1, onde cada pixel é convertido para dois estados, preto (0) ou branco (255), dependendo de seu valor original comparado com o valor de limiar.

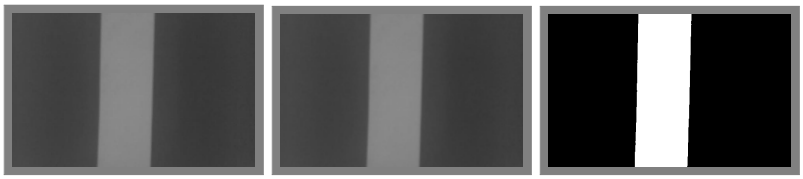
$$\text{pixel}_{binarizado}(x,y) = \begin{cases} 255, & \text{se pixel}_{escala\_de\_cinza}(x,y) \geq T \\ 0, & \text{se pixel}_{escala\_de\_cinza}(x,y) < T \end{cases} \tag{1}$$

A Figura 3 mostra os quadros processados após a conversão de escala de cinza, suavização de ruídos e binarização, respectivamente, para dois casos: pista saudável, na Figura 3(a) e pista com falha, na Figura 3(b). A moldura cinza foi adicionada para facilitar a visualização dos quadros e não está presente durante o processamento de imagem.

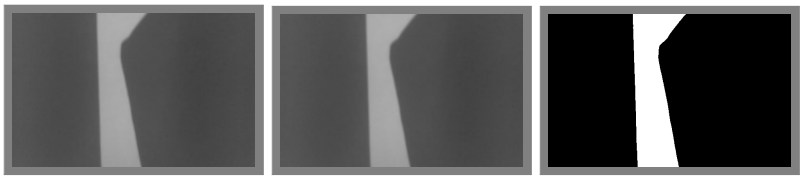
Após o processamento, os contornos da pista são identificados, delimitados e, em seguida, contabilizados. Essas ações são realizadas na etapa indicada no fluxograma como "Identifica e desenha os contornos da pista" e "Conta o número de contornos presentes na pista".

Figura 3 – Resultados do tratamento inicial da imagem que prepara o quadro para a identificação e medição das falhas

(a) *Frames* processados com trecho de uma pista saudável.



(b) *Frame* com trecho de uma pista com 25% de falha.



Fonte: autoria própria.

Para a identificação dos contornos na imagem é usada a função `findContours` da biblioteca OpenCV. A própria função possui o atributo de tamanho, do qual é extraído o número de contornos encontrados na imagem. Em uma situação normal, espera-se encontrar três contornos.

Usa-se a função `contourArea` para calcular a área da faixa guia no quadro. A função utiliza a Equação 2 para calcular a área do polígono definido pela função `findContours`. Com isso, é possível calcular a porcentagem de falhas presentes na pista e classificá-las.

$$A = \frac{1}{2}\left|\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i)\right| \tag{2}$$

A área da falha é calculada usando a Equação 3. A $\text{Área}_{\text{Padrão}}$ é um valor determinado que indica a área da faixa guia saudável.

$$percentual_{falha} = 100 - \left(100 \times \frac{\text{área}}{\text{Área}_{\text{Padrão}}}\right)[\%] \tag{3}$$

Seguindo o fluxograma da Figura 2, caso o valor percentual de falha seja menor do que 3%, o próximo quadro é processado, significando que não há falha no presente quadro. Caso haja falhas, existem três possibilidades de classificação. A interrupção total é quando a pista possui um corte que inicia em uma borda e termina na outra, interrompendo o caminho. Nessas circunstâncias, o valor retornado será $\geq 99\%$ de falha na pista.

A falha no interior da faixa ocorre quando nenhuma de suas bordas estão danificadas, mas existe um desgaste na parte interna da pista. Para diferenciá-la de uma pista interrompida, além de serem considerados valores de porcentagem de falha $> 3\%$ e $< 99\%$, são observados o número de contornos da faixa. Em interrupções totais, em pistas saudáveis e em pistas com falhas no seu contorno, existem duas bordas sendo identificadas e traçadas no quadro. No primeiro caso, há uma borda superior e outra inferior, sendo divididas pela interrupção. No segundo e terceiro caso, existem a borda da lateral esquerda e a borda da lateral direita da pista sendo traçadas. Independentemente de ter falhas no contorno da pista, continuam sendo

identificadas duas bordas. A única situação em que ocorre um número de contornos maior que dois é quando, além de serem traçadas as bordas das laterais da pista, é traçado um contorno no interior dela, indicando falhas no centro da faixa.

Caso não seja uma interrupção e nem uma falha interna, será detectada uma falha na borda da pista. Para essa situação, os valores retornados serão $> 3\%$ e $< 99\%$. Caso não haja falhas, o valor retornado será $\leq 3\%$ de falha. Para determinar essa última porcentagem, foram utilizados três vídeos como teste, dois vídeos com uma pista falhada e um vídeo com a pista sem falhas, e observado que, em *frame*s onde a pista estava saudável, ou seja, sem apresentar falhas, a porcentagem de falha variava de 0 a 3%. Em todas as situações, após o valor ser retornado, o sistema volta ao início e faz a aquisição do próximo *frame* do vídeo[1].

A proposta apresentada possui suas limitações. Em sua versão atual, é reconhecido que a precisão da detecção e da medição da falha na faixa guia pode ser afetada por fatores externos, como as condições de iluminação, a velocidade do AGV e condições adversas da faixa guia, como sujeiras que podem ser confundidas com falhas. O projeto do AGV, ainda em desenvolvimento, prevê estruturas e outras técnicas para mitigar alguns desses problemas e minimizar a quantidade de ajustes durante a operação.

A câmera está instalada em uma estrutura que isola a iluminação externa e conta com iluminação própria por fitas de LED, conforme inicialmente proposta em (Oliveira et al., 2019) e aprimorada no AGV deste trabalho (Reis, 2023). A velocidade do AGV influencia a qualidade da imagem da faixa guia, sendo necessário ajustar a frequência de aquisição de imagens considerando a maior velocidade do veículo (Oliveira et al., 2019). Condições adversas da faixa não foram consideradas nesta etapa, pois o ambiente industrial para montagem de equipamentos eletrônicos apresenta piso limpo, conforme (Reis, 2023).

Além disso, é importante salientar que o presente módulo faz parte de uma arquitetura maior de operação e controle do veículo e que os dados gerados a partir do processamento de imagem não serão utilizados sozinhos. Por exemplo, para utilização do módulo de detecção de falha como um apoio à manutenção, é interessante que as informações de falha da faixa guia sejam acompanhadas de informações do local da falha. Para isso, outro módulo de software será utilizado em conjunto, o serviço de Odometria.

4 MATERIAIS E MÉTODOS

O estudo consistiu na elaboração de um componente de software voltado para a avaliação da condição da pista guia de um AGV, com a vantagem de sua implementação não exigir a existência prévia de um sistema integrado. A codificação foi realizada na linguagem C++ utilizando a biblioteca OpenCV para o tratamento e processamento de imagens. O veículo autoguiado utilizado já possui uma estrutura de hardware e software determinada. Portanto, foi fundamental que o módulo em questão apresentasse elevado nível de coesão e baixo acoplamento, utilizando-se de diversas estratégias de elaboração para alcançar esse objetivo.

[1]Os vídeos citados no trabalho estão disponíveis no link: https://youtu.be/saBIzhj4LEQ.

Foi aplicada a arquitetura SOA (orientada a serviços), com o objetivo de gerenciar de maneira mais efetiva a programação orientada a objetos construída, possibilitando a fragmentação da lógica em partes (Erl, 2005). Além disso, foram feitas documentações iniciais e posteriores à conclusão do projeto, utilizando diagramas de forma gráfica através da linguagem UML (Linguagem de Modelagem Unificada) (Jones, 1999).

Ademais, foi utilizado o padrão de projeto Fachada durante o desenvolvimento do software, de modo com que as interfaces do serviço fossem representadas de maneira mais simplificada, apresentadas em um alto nível (Shalloway; Trott, 2004). A linguagem de programação foi utilizada pois seu código fonte é compilado para execução, que como Nametala et al. (2018) afirmam, possui melhor performance no sistema operacional Ubuntu, utilizado no projeto. Além disso, é compatível com a biblioteca OpenCV, utilizada para o tratamento de imagem, sendo essencial para a identificação de falhas na pista.

A Raspberry Pi é o microprocessador utilizado para embarcar o sistema de controle do AGV. Além de ser um dispositivo com grande flexibilidade e baixo custo, ele também conta com recursos importantes, como WiFi, Bluetooth, entradas USB e saída HDMI. Ele funciona executando uma versão do sistema operacional Linux, com interface gráfica e pacotes de aplicativo de código aberto (Souza, 2022).

Em complemento ao que foi destacado, o microprocessador apresenta um ótimo desempenho em aplicações que requerem processamento em tempo real. Além disso, apresenta uma ampla capacidade de trabalhar com diversos periféricos mesmo com seu tamanho reduzido (Oliveira, 2017). O veículo utiliza um módulo de câmera como sensor principal para seguir a faixa guia. Esse módulo foi desenvolvido para a Raspberry Pi, conhecido como RaspiCam ou PiCam v1.3. As especificações mais relevantes do dispositivo estão listadas na Tabela 1. O serviço de detecção de falhas processa os mesmos quadros que são utilizados para manter o AGV seguindo a faixa.

Tabela 1 – Características da câmera empregada no AGV

Características	**Valor**	**Características**	**Valor**
Tamanho do módulo	$25 \times 24 \times 9$ mm	Peso	3 g
Resolução	5 Megapixels	Modos de vídeo	1080p30, 720p60, 640×480p60/90
Integração Linux	Driver V4L2 disponível	Sensor	OmniVision OV5647
Resolução do Sensor	2592×1944 pixels	Área da imagem no sensor	$3{,}76 \times 2{,}74$ mm
Tamanho do pixel	$1{,}4 \times 1{,}4\,\mu$m	Tamanho óptico	$1/4$”
Campo de visão vertical, α	$41{,}41 \pm 0{,}11°$	Campo de visão horizontal, β	$53{,}5 \pm 0{,}13°$
Distância focal	3,60 mm		

Fonte: (Reis, 2023).

4.1 Padrão Fachada

Padrões de design são meios de melhorar códigos preexistentes, os tornando fáceis de serem implementados e manuseados. Eles tornam programas mais visuais, facilitando seu entendimento e manutenção. Ademais, apenas fazem sentido quando implementados em linguagens de programação orientadas a objetos, pois estas costumam encapsular linhas de código dentro de classes ou funções, que podem ser facilmente representadas em módulos de lógica (Lasater, 2006).

O padrão fachada pode ser definido como uma interface de serviços de alto nível. Nesteruk (2018) usa como exemplo desse padrão uma casa, onde há menor preocupação com os sistemas internos, como elétrico ou hidráulico, onde só se deseja que estejam funcionando. Desse modo, há maior importância com a parte externa, como móveis ou pintura, o que é de fato visto pelos moradores.

Ao criar uma fachada para um sistema, uma interface simplificada também é desenvolvida, podendo ser utilizada para esconder o código existente. Além disso, é possível acessar facilmente qualquer subsistema sem a necessidade de interferência em outros preexistentes. Esse padrão não necessita ser definido apenas como uma interface mais simples de visualização, mas também como uma maneira de reduzir a quantidade de informação que o usuário deverá lidar. De forma resumida, é necessário simplificar um sistema complexo preexistente e o método fachada apresenta uma interface simplificada para que o usuário consiga interagir com essa de maneira dinâmica (Shalloway; Trott, 2004).

É possível que os utilizadores do código customizem ferramentas dentro de funções criadas com o método fachada, de modo com que consigam modificar uma parte desejada sem que necessitem manusear uma parte complexa do código (Nesteruk, 2018).

4.2 Arquitetura Orientada a Serviços (SOA)

Para se entender o que é a arquitetura orientada a serviços, primeiramente é preciso saber o que é um serviço. Guimarães Júnior (2015) define serviço como "a unidade fundamental de uma arquitetura SOA, é um elemento computacional que tem como propósito desempenhar uma função específica e que pode ser utilizado por um cliente."

Um serviço geralmente é composto por duas partes, uma implementação e uma interface. A implementação consiste em um módulo de software que é responsável pela execução das funções do serviço, já a interface tem como função ser a parte acessível do serviço, onde são detalhadas suas funções e suas condições de uso, entretanto, a interface não apresenta detalhes de como o serviço foi construído. Pode-se dizer, então, que a interface é responsável por "encapsular" o módulo de software, tornando visível ao cliente apenas os detalhes necessários.

A arquitetura SOA, por apresentar serviços que constituem funções completas e com um baixo acoplamento, ou seja, não sendo limitados apenas a um programa ou software específico, podem ser fornecidos, usados e reusados em uma organização ou entre organizações, através de redes de comunicação, como a internet (Fugita, 2009).

A arquitetura SOA estrutura sistemas em três papéis principais: provedor, registro e cliente. O provedor implementa e mantém o serviço, o registro atua como diretório para divulgação das descrições e requisitos, e os clientes consultam o registro para consumir os serviços. Para ser considerada orientada a serviços, a arquitetura deve cumprir requisitos específicos.

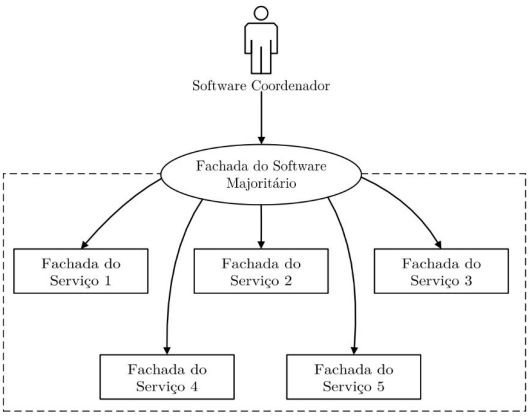
Entre esses requisitos, destacam-se a reutilização dos serviços por múltiplos clientes, a existência de um contrato formal que define funções, requisitos e interface técnica, e características essenciais como baixo acoplamento, autonomia — permitindo que o serviço funcione independentemente — e a restrição de comunicação direta entre serviços (Mendes et al., 2008; Souza, 2021).

Além das condições citadas acima, também é importante destacar que somente é necessário conhecer as entradas e saídas do serviço, não sendo necessário conhecer sua lógica. Os serviços podem também se compor de forma que a saída gerada por um serviço pode ser encaminhada para outro a partir da intermediação do software consumidor. A última característica compreendida é a necessidade de uma interface pública que permita que os serviços sejam descobertos e utilizados (Erl, 2007). Todos os requisitos citados anteriormente são cumpridos quando a arquitetura orientada a serviços é aplicada em *web services*. Entretanto, tratando de sistemas embarcados, é possível, ao compará-los, notar certa diferença entre as aplicações.

Por possuírem um contrato formal para utilização dos serviços, no caso dos AGVs, as interfaces dos serviços são unificadas em uma interface do padrão fachada, simplificando a comunicação entre o software cliente e os subsistemas. Essa interface unificada atua como um contrato formal e uma interface técnica para os serviços dentro do subsistema, além de atuar como interface pública (Souza, 2021).

Em serviços embarcados, a velocidade de processamento dos dados é essencial. Para evitar complicações e garantir um melhor desempenho, os serviços são pré-instalados no sistema embarcado, eliminando a necessidade de um software intermediário, como o *Enterprise Service Bus* (ESB), para conectar o provedor do serviço ao consumidor (Souza, 2021). A Figura 4 mostra um exemplo de aplicação do padrão fachada.

Figura 4 – Exemplo de uso do Padrão Fachada na SOA



Fonte: (Reis, 2023).

Dentro da fachada, há diversos serviços que compreendem um subsistema. O software coordenador consegue, ao se conectar com aquela fachada, usufruir de todas as implementações existentes dentro do subsistema, porém, de forma controlada. O software coordenador não tem acesso direto à todos os métodos, apenas àqueles disponíveis na fachada. Outros métodos podem ser chamados, porém, o software coordenador não tem acesso à esses fluxos de dados, recebendo somente a resposta final do serviço.

Dessa forma, pode-se concluir que a aplicação de SOA em serviços web e em sistemas embarcados é semelhante, com a principal diferença sendo a ausência da necessidade de um software exclusivo para realizar a conexão entre o provedor e o consumidor nos sistemas embarcados, já que todos os serviços necessários estão instalados no dispositivo.

4.3 Linguagem de Modelagem Unificada

A UML (*Unified Modeling Language*) é definida por Booch et al. (2006) como "uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas complexos de software". Essa linguagem é utilizada para modelar softwares utilizando orientação a objetos e sua aplicação acontece principalmente nas etapas de análise de requisitos. É uma linguagem-padrão utilizada internacionalmente pelo mercado industrial.

A UML não depende de métodos de desenvolvimento ou linguagens específicas e deve ser usada conforme a necessidade do sistema, sem a obrigatoriedade de todos os seus diagramas. Aplicável em diversos domínios, como sistemas corporativos, telecomunicações e serviços financeiros, a UML foi criada para padronizar a modelagem de software, superando a diversidade de linguagens anteriores que dificultavam a comunicação entre desenvolvedores. Essa padronização é realizada por meio de diagramas que detalham suas finalidades, componentes e aplicações.

O propósito de haver tantos diagramas é prover muitas visões diferentes sobre o projeto a ser modelado. Para que vários aspectos sejam levados em consideração na hora da modelagem e o projeto fique mais completo. Cada diagrama é responsável por analisar um projeto, ou parte dele, por uma determinada visão (Guedes, 2018). Para documentar o módulo de software proposto, foram construídos os diagramas de atividades, sequência, casos de uso e classes, explicados a seguir.

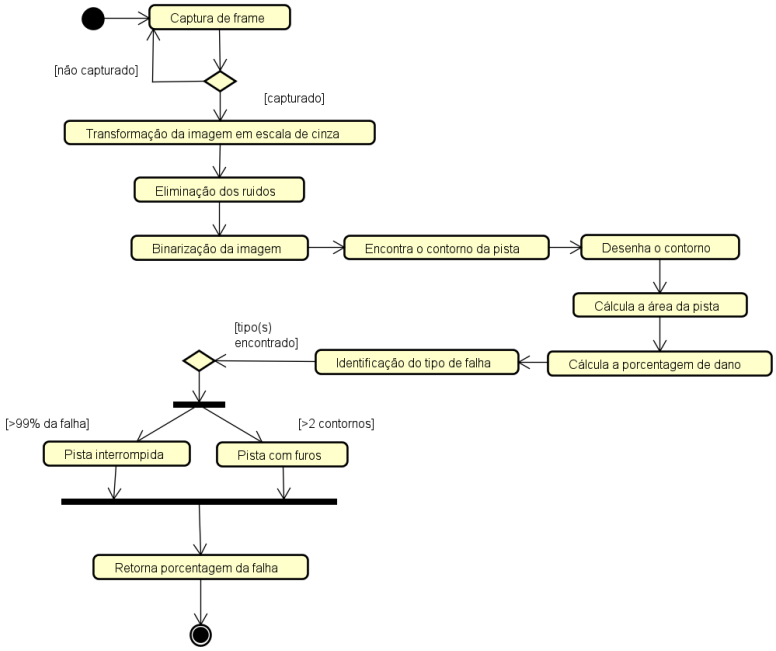
4.3.1 Diagrama de Atividades

O diagrama de atividades funciona como um gráfico de fluxo, apresentando o fluxo de um sistema de controle de um serviço para outro, tem como prioridade indicar os passos a serem seguidos para a finalização de um processo específico, com a possibilidade de ser apresentado por um método específico, com certo grau de dificuldade. O diagrama foca em representar o fluxo do monitoramento de um processo (Booch et al., 2006).

O diagrama de atividades é uma representação gráfica que descreve de maneira sistemá-

tica e organizada o fluxo de atividades e a lógica envolvida em um processo ou procedimento. No contexto específico do código em questão, o diagrama de atividades da Figura 5 é utilizado para apresentar as etapas sequenciais a serem seguidas no processo de detecção de falhas na pista. Essa representação visual auxilia na compreensão e no entendimento de como o processo é executado, destacando as atividades envolvidas e as relações de fluxo entre elas.

Figura 5 – Diagrama de Atividades do serviço implementado



Fonte: autoria própria.

O código inicia com a captura do *frame*, repetindo o processo caso falhe. Em seguida, a imagem é tratada por conversão para escala de cinza, aplicação de filtro de desfoque médio para reduzir ruídos e binarização em pixels preto e branco. Com a imagem processada, identifica-se o contorno da faixa e calcula-se a área da pista. A análise da porcentagem de dano e da quantidade de contornos permite detectar falhas ou interrupções na faixa, avaliando assim as condições da pista e retornando a porcentagem de dano encontrada.

4.3.2 Diagrama de Sequência

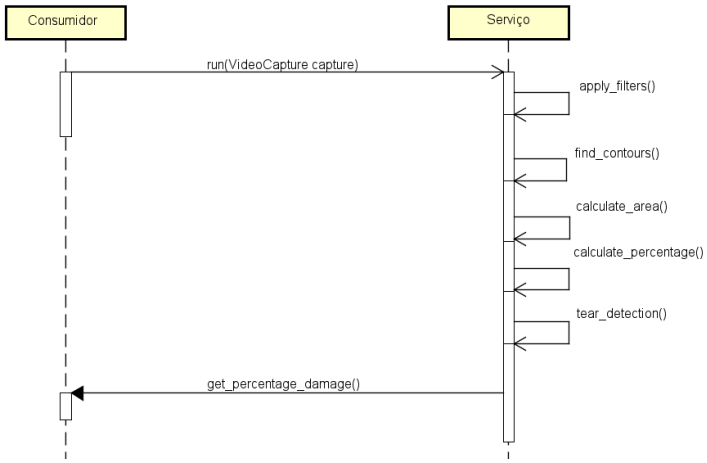
Esse diagrama foca na ordem temporal em que as informações são compartilhadas entre os objetos participantes de um processo específico. Normalmente identifica a ocorrência que gera todo o processo modelado, assim como o ator que dá origem a essa ocorrência e aponta como o processo deve se desenvolver e ser finalizado através de uma chamada de métodos que são recebidos por mensagens enviadas em meio aos objetos (Guedes, 2018).

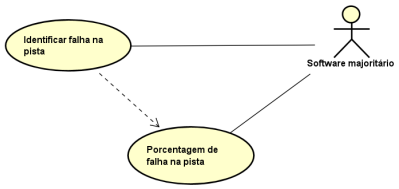
Com o objetivo de ilustrar a execução do propósito estabelecido pelo diagrama de casos de uso, o diagrama de sequência da Figura 6(a) oferece uma descrição detalhada e sequencial do processo de programação a ser implementado. Com a finalidade de avaliar o estado das faixas guias, é realizado o cálculo da porcentagem da pista, permitindo, dessa forma, estimar o nível

de desgaste ao longo do tempo.

Figura 6 – Diagramas de sequência e de casos de uso do serviço implementado

(a) Diagrama de sequência **(b) Diagrama de casos de uso**





Fonte: autoria própria.

Após o cálculo, será possível determinar a porcentagem de falha, considerando que a presença de falhas ocorre quando a pista não se encontra em condições ideais (ou seja, não está 100% saudável). Portanto, por meio da porcentagem de erro, é possível identificar essas falhas. A função encarregada dessa tarefa é a `calculate_area`.

Para identificar o tipo de falha presente, a função chamada `tear_detection` é utilizada. Essa função estabelece que, se a porcentagem de falha for superior a 99%, indica que a falha está separando a pista em duas partes, resultando em um corte total no meio. A função também verifica se existem mais de dois contornos nas falhas. Se houver, isso indica que a falha está localizada dentro da área da pista. Por fim, o último passo consiste em coletar as porcentagens encontradas para criar um gráfico. Essa função recebe o nome de `get_percentage_damage`.

4.3.3 Diagrama de Casos de Uso

O diagrama de casos de uso tem como característica uma linguagem simples e de fácil entendimento para os desfrutadores. Ele apresenta uma ideia mais básica a respeito do funcionamento do sistema, mostra como os atores interagem com o sistema e quais funções e serviços o sistema oferece. Sua aplicação é mais comum em etapas de levantamento e análise de requisitos, embora em todo o processo possa ser verificado (Booch et al., 2006).

O diagrama da Figura 6(b) apresenta as funcionalidades do sistema a partir do ponto de vista do usuário, demonstrando a interação dos atores que podem ser usuários, sistemas externos ou certas entidades que interajam com o sistema para a realização de determinada ação, assim como a interação desses elementos entre si.

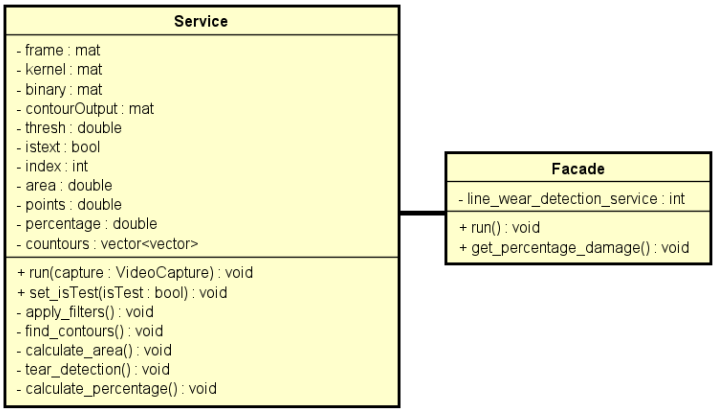
Pode-se observar a partir do diagrama que há dois casos de uso. Caso uma falha seja identificada, há um relacionamento de inclusão e um de associação, sendo o relacionamento de inclusão entre identificar falha na pista e porcentagem de falha na pista. Tanto a identificação

de falhas quanto a porcentagem retornada se conectam ao software majoritário que é um ator.

4.3.4 Diagrama de Classes

Há grande possibilidade de ser o diagrama mais utilizado, sendo de grande importância. Serve como complemento de grande parte dos diagramas e é utilizado para determinar a estrutura das classes presentes no sistema. Ele estabelece como as classes estão relacionadas e como trocam informações (Guedes, 2018). O diagrama da Figura 7 demonstra as classes implementadas no projeto, através de informações presentes na programação, estruturadas em: nome, atributos e métodos.

Figura 7 – Diagrama de Classes



Fonte: autoria própria.

Contém informações sobre as classes *Service* e *Facade*. A primeira apresenta informações sobre a programação de maneira efetiva, onde de fato o software foi desenvolvido e essas informações são passadas de forma mais completa, bem similar ao código. A segunda é um tipo de padrão de projeto cujo objetivo é prover uma interface mais simples para o entendimento de um sistema complexo, utilizada como um modo de abstração entre o cliente e os elementos internos de um sistema.

Informações sobre o tipo de variáveis como se uma variável é pública, podendo ser acessada e modificada por qualquer parte do programa, ou privada, podendo ser acessadas e modificadas apenas pela classe que a declara, também estão presentes. Esse diagrama se assemelha muito a programação, trazendo uma boa base do que deve ser inserido no código para que esse programa execute a função desejada.

5 RESULTADOS E DISCUSSÃO

Para testar a capacidade do módulo de software na detecção de falhas na pista, foram usados os três vídeos adquiridos, por meio da câmera embarcada no AGV, no início do desenvolvimento do serviço. Esses vídeos contemplam as situações de faixa guia saudável e faixa guia com falhas, mas com o AGV percorrendo o caminho nos dois sentidos. Além deles, duas outras pistas, uma reta e outra curva, foram filmadas 4 vezes cada, sendo 2 vídeos com as faixas guias falhadas e 2 vídeos com as mesmas saudáveis. Na pista reta, os testes foram feitos com o veículo partindo do mesmo ponto em todas as gravações. Já na pista curva, o ponto de partida do AGV em um dos vídeos com a faixa guia falhada foi invertido em relação à outra filmagem.

Apesar de existirem vídeos representando a pista com a mesma condição física, falhada ou não, esses são considerados vídeos diferentes, já que existe a possibilidade de variação da iluminação do quadro, causada pela própria variação no reflexo da iluminação forçada em trechos distintos do piso e pela porção de faixa branca presente no quadro, e da velocidade do veículo para um mesmo trecho da pista cada vez que ele fez o percurso.

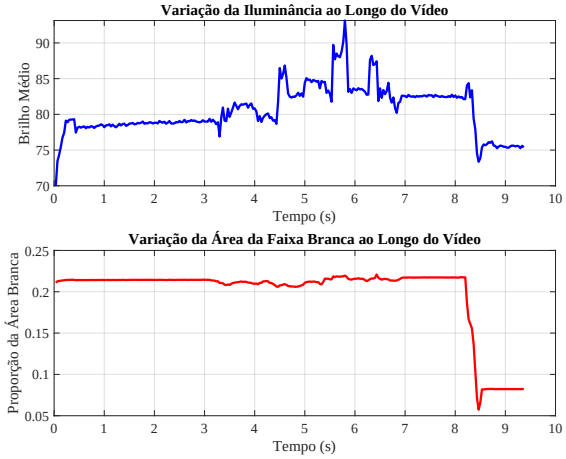
Nesse sentido, antes de analisar os resultados do módulo proposto, serão apresentados gráficos de análise de brilho da imagem, de forma a representar a relação da iluminância do quadro processado. Para isso, foram usados três vídeos iniciais da pista reta. Quatro análises foram feitas para cada vídeo. O brilho médio de cada quadro em escala de cinza foi calculado, usando a escala de 0 a 255, assim como a porção da imagem que é formada por pixels brancos, sendo a área da imagem igual a 1, ou 100%. Esses valores foram plotados em gráficos, de forma a poder relacionar a quantidade de pixels brancos na imagem com a variação no brilho. Além disso, para cada vídeo foram determinados a amplitude do brilho, ou seja, a variação máxima encontrada, e o desvio padrão do brilho, indicando o nível de oscilação ao longo do vídeo.

Os gráficos mostrados na Figura 8(a) mostram, respectivamente, o brilho médio e a porção da área do quadro que equivale à faixa guia ao longo do vídeo da pista reta saudável. Nele é possível perceber uma variação do brilho médio no intervalo de tempo de 3 a 7 segundos, enquanto o AGV está em movimento. Entre 7 e 8 segundos o AGV está parado, até que avança para o final da faixa, entre 8 e 9 segundos, quando a faixa acaba e maior parte do quadro representa apenas o piso preto. Assim, pode-se relacionar a variação no brilho com o movimento do veículo, pois a iluminação forçada da câmera reflete de formas distintas na faixa e no piso. Entretanto, a amplitude do brilho médio foi de $23,11$ unidades e o desvio padrão na ordem de $3,34$.

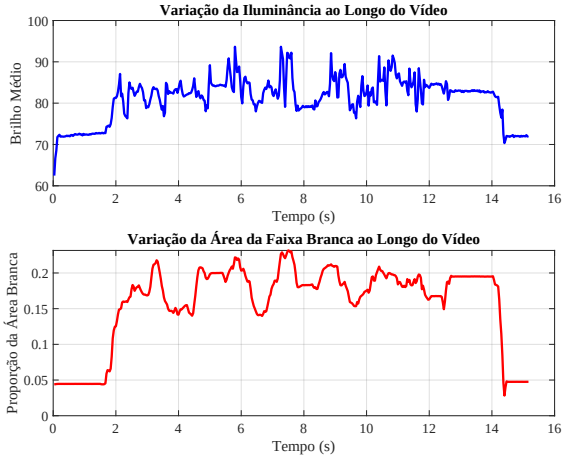
Similarmente, os gráficos mostrados na Figura 8(b) mostram, respectivamente, o brilho médio e a porção da área do quadro que equivale à faixa guia ao longo do vídeo da pista reta com falhas. Nessa situação, observa-se uma variação do brilho médio no intervalo de tempo de 2 a 12 segundos, aproximadamente, enquanto o AGV está em movimento. Próximo dos 14 segundos, o AGV chega ao final da faixa. Com maior variação da área branca do quadro, a amplitude do brilho médio foi de $31,18$ unidades e o desvio padrão na ordem de $5,10$, como poderia ser esperado.

Figura 8 – Análise do brilho em diferentes situações

(a) Análise no vídeo da pista reta saudável



(b) Análise em vídeo da pista reta com falhas



Fonte: autoria própria.

A análise da variação de brilho nos vídeos revelou que, apesar da mesma condição física da faixa guia, diferenças na iluminância dos quadros processados são inevitáveis devido à interação entre a iluminação forçada da câmera e a superfície da pista. Esses resultados indicam que, quanto maior a variação da faixa branca no quadro devido às falhas na pista, maior é a oscilação no brilho médio. Além disso, a variação da iluminância ao longo do tempo está fortemente correlacionada com o deslocamento do AGV e sua interação com a luz refletida na faixa e no piso. Entretanto, relacionando os resultados anteriores com os resultados obtidos da detecção de falhas, não houve um impacto preponderante da variação do brilho, destacando que vídeos diferentes da mesma pista obtiveram resultados similares.

Foram analisados 11 vídeos, resultando na geração de 22 gráficos, divididos em dois tipos: percentual de falhas na pista e interrupções totais ao longo do tempo. Ambos apresentam os resultados por *frame*, com atualização a cada 33 ms, correspondente à taxa de 30 *frames* por segundo.

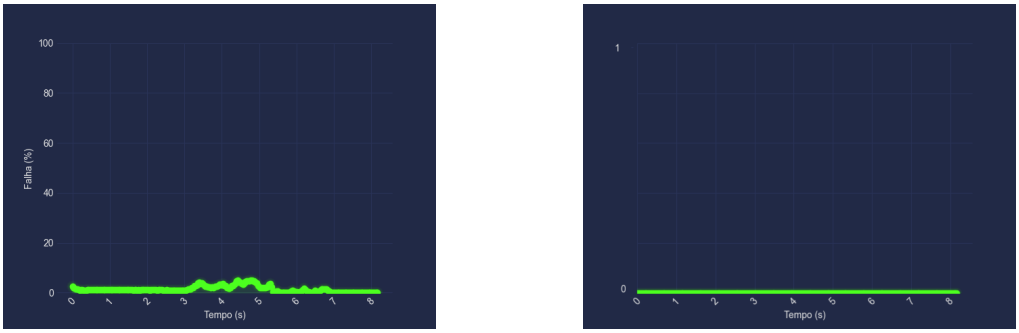
No caso do primeiro tipo de gráfico, o eixo das ordenadas (y) possui um intervalo de 0 a 100%, que corresponde a porcentagem de falha que a pista pode conter. Nele poderá ser observado o valor retornado pelo software para cada situação de falha descrita no tópico 3. Já o segundo tipo de gráfico indica se foram detectadas interrupções totais ou não em cada *frame* do vídeo. Dessa maneira, seu eixo das ordenadas possui apenas dois estados ou valores, 0 e 1 ou falso e verdadeiro.

Os primeiros gráficos analisados correspondem aos três vídeos que serviram como base para a construção do software. A faixa guia em tais gravações era uma reta. A Figura 9(a) se trata do gráfico de porcentagem de falha ao longo do vídeo com a pista saudável. Em consonância com a proposta do software, a porcentagem retornada nessa situação não ultrapassou os 3% de falha. Sendo assim, o gráfico da Figura 9(b) não detectou interrupções totais, que ocorreriam caso apresentasse uma porcentagem de falha maior ou igual a 99%.

Já as Figuras 10(a) e 10(c) correspondem aos gráficos com o percentual de falhas nos vídeos com a pista reta com falhas. Como pode-se observar, ambos tiveram respostas similares

Figura 9 – Gráficos resultantes para a pista reta saudável do vídeo 1

(a) Gráfico de porcentagem de falhas **(b) Gráfico de interrupção total**



Fonte: autoria própria.

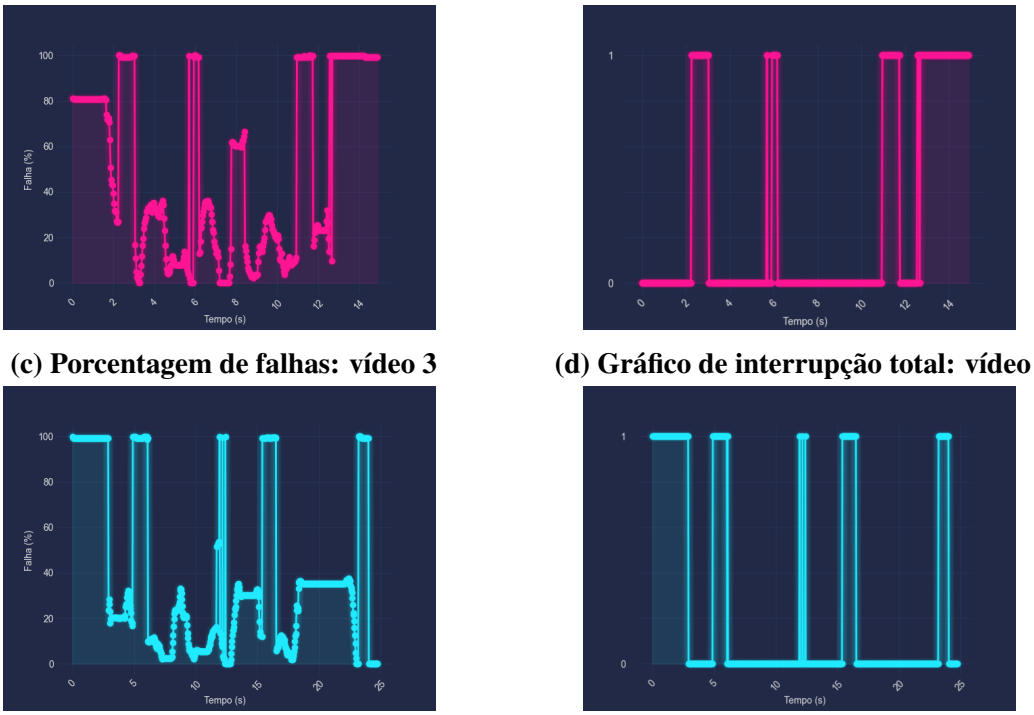
em relação a porcentagem de falha, só que de maneira invertida nas figuras. Isso ocorreu porque as gravações são do AGV percorrendo o mesmo trajeto, mas partindo de lados opostos em cada aquisição de imagens.

No gráfico da Figura 10(a), foi indicada uma porcentagem maior ou igual a 99% de falhas nos últimos segundos do vídeo, o que resultou na indicação de uma interrupção total no gráfico da Figura 10(b), que corresponde a mesma filmagem. Tal resultado se repetiu nas Figuras 10(c) e 10(d), só que nos segundos iniciais da gravação.

Figura 10 – Gráficos resultantes para a pista reta com falhas dos vídeos 2 e 3

(a) Porcentagem de falhas: vídeo 2 **(b) Gráfico de interrupção total: vídeo 2**

(c) Porcentagem de falhas: vídeo 3 **(d) Gráfico de interrupção total: vídeo 3**



Fonte: autoria própria.

Essa interrupção indicada corresponde ao final da pista nos gráficos das Figuras 10(a) e 10(b) e ao início da faixa guia nos gráficos das Figuras 10(c) e 10(d). É considerada uma interrupção total nessa situação porque houve um contraste entre o piso onde a faixa foi instalada e o seu início.

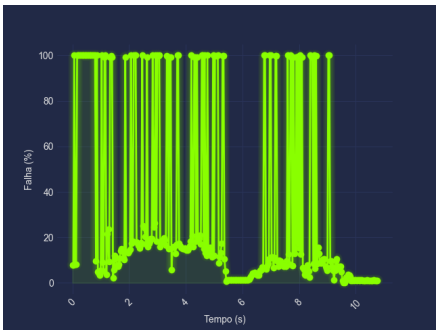
Os vídeos adquiridos para os testes do módulo, após seu desenvolvimento, ao serem submetidos ao serviço, comprovaram a influência de alguns fatores na detecção de falhas. As Figuras 11(a) e 11(c) se tratam do gráfico com o percentual de defeitos da pista reta e sem falhas das gravações dos vídeos 4 e 5.

Diferentemente do gráfico da Figura 9(a), que permaneceu ao longo do vídeo entre os limites percentuais determinados para uma faixa sem falhas, a porcentagem de diversos *frames* foi maior ou igual a 99%, indicando interrupções totais que não existiam na pista. Por esse motivo, os gráficos das imagens 11(b) e 11(d), que correspondem aos gráficos de interrupção das Figuras 11(a) e 11(c), respectivamente, plotaram várias ocorrências de descontinuidades nos instantes em que as porcentagens atingiram, aproximadamente, 100%.

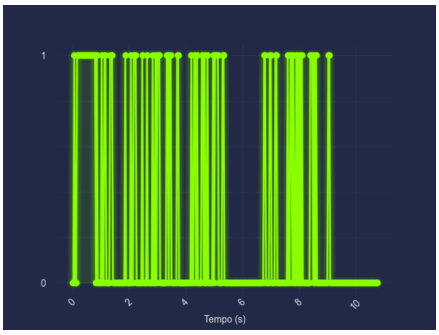
Além da velocidade do AGV nesses testes ter sido maior do que nos primeiros três vídeos submetidos ao serviço, deve-se destacar a condição da fita branca usada como guia para a navegação. A faixa era mais antiga, por isso, estava com coloração amarelada, o que influencia no contraste com o fundo preto onde é instalada. Fora isso, deve-se considerar as diferenças de iluminação externa, pois, mesmo que o veículo autoguiado tenha uma iluminação forçada para limitar os efeitos dessas variações, elas ainda influenciam.

Figura 11 – Gráficos resultantes para uma pista reta saudável dos vídeos 4 e 5

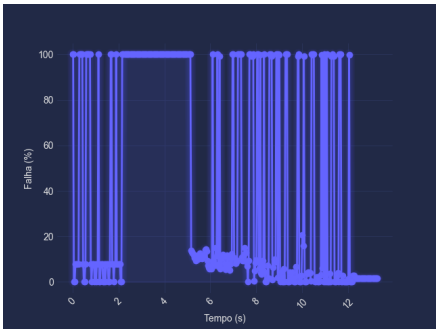
(a) Porcentagem de falhas: vídeo 4



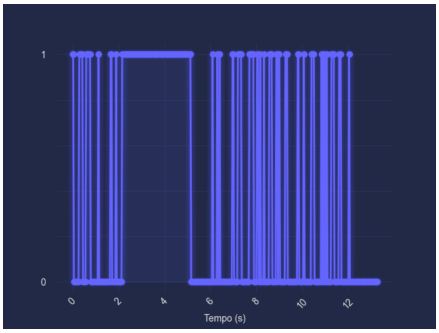
(b) Gráfico de interrupção total: vídeo 4



(c) Porcentagem de falhas: vídeo 5



(d) Gráfico de interrupção total: vídeo 5



Fonte: autoria própria.

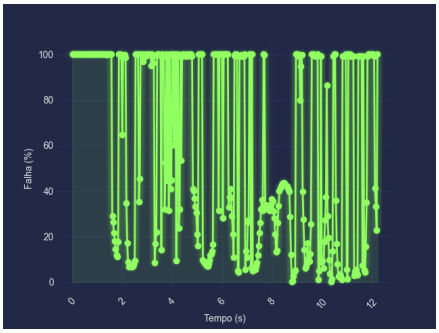
Apesar disso, nota-se que, em alguns *frames* do fim do vídeo das duas pistas, onde o veículo está com uma velocidade mais baixa porque ele está parando, o módulo de software consegue detectar que a pista não possui interrupções totais e o valor percentual fica dentro dos limites determinados na Seção 3. Outra situação em que ocorreu uma resposta diferente do serviço foi entre 5 e 6 segundos do gráfico da Figura 11(a). Pode-se perceber que o percentual de falhas ficou entre 0 e 3% e não houve nenhuma interrupção total identificada na imagem 11(b),

mas isso ocorreu devido a uma parada que o AGV fez nesse instante do vídeo. Considerando que, em uma parada, o serviço conseguiu ler corretamente a faixa, mesmo ela estando amarelada e em diferentes condições de iluminação, percebe-se a influência também da velocidade unida aos fatores anteriores.

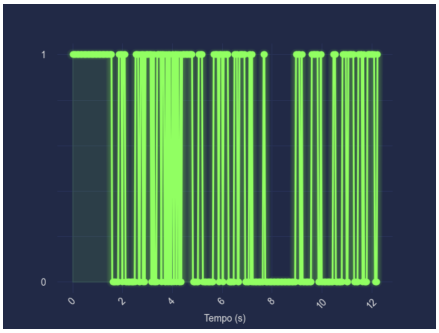
A pista reta e sem falhas analisada a partir dos vídeos 4 e 5 teve trechos recortados com o intuito de se criar interrupções totais, falhas nas bordas e falhas no interior da faixa. O veículo foi colocado para fazer mais duas vezes o percurso, gerando os vídeos 6 e 7, e foram gerados os gráficos de porcentagem e interrupção total para ambas as gravações. É possível notar que muitos *frames* dos gráficos das Figuras 12(a) e 12(c) ainda tiveram um percentual de falhas alto, superior a 99%, o que levou a plotagem de um número expressivo de interrupções nas Figuras 12(b) e 12(d). Apesar de terem sido criadas descontinuidades na faixa guia, os resultados dos gráficos de interrupção, assim como nas Figuras 11(b) e 11(d), indicaram cortes totais em trechos onde não possuía. Isso se deve aos mesmos fatores que influenciaram os resultados da mesma pista, porém sem falhas.

Figura 12 – Gráficos resultantes para uma pista reta com falhas dos vídeos 6 e 7

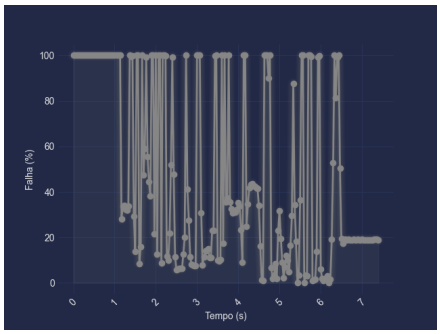
(a) Porcentagem de falhas: vídeo 6



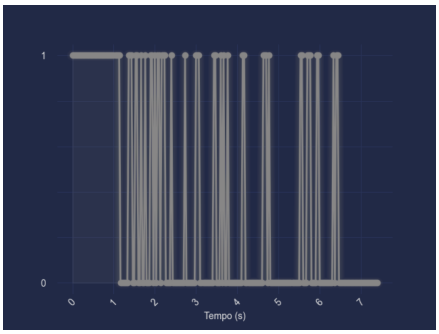
(b) Gráfico de interrupção total: vídeo 6



(c) Porcentagem de falhas: vídeo 7



(d) Gráfico de interrupção total: vídeo 7



Fonte: autoria própria.

No entanto, deve-se observar que o software apresentou repetibilidade na detecção de falhas, já que, quando o AGV percorreu duas vezes a mesma pista, os gráficos de porcentagem e de interrupção tiveram plotados resultados similares, mesmo que equivocados em alguns pontos.

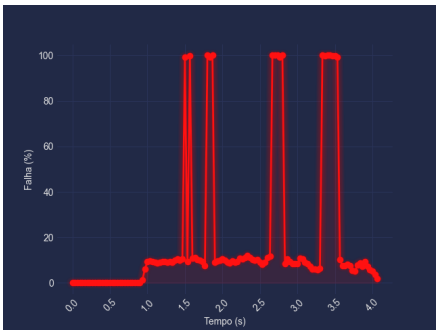
Junto com os vídeos da pista reta, foram adquiridos os quatro vídeos de um percurso com curva. Os dois primeiros vídeos, 8 e 9, foram da faixa guia saudável e com o veículo autoguiado partindo do mesmo ponto. Em comparação com os resultados da pista reta e sem falhas adquiridas no mesmo dia, as porcentagens de falha retornadas pelo módulo de software

foram muito mais exatas. Deve-se considerar como um dos motivos o fato da pista com curva ter sido construída no dia da aquisição dos vídeos. Isso significa que a fita que serve como guia para o veículo estava mais nova do que a fita da pista reta, levando a um contraste maior com o fundo preto no qual foi instalada.

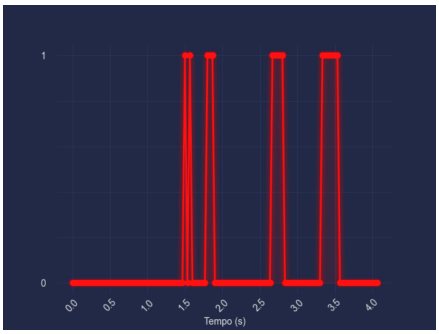
Os gráficos de percentual de falha das Figuras 13(a) e 13(c) tiveram mais *frames* com resultados em conformidade com os valores determinados na Seção 3 para uma pista saudável. Além disso, ambos ficaram parecidos, demonstrando novamente a precisão do serviço. No entanto, não se deve desconsiderar que ainda foram plotadas porcentagens de falha maiores que 99%, resultando na indicação de descontinuidades inexistentes na pista nos gráficos das Figuras 13(b) e 13(d).

Figura 13 – Gráficos resultantes para uma pista pista em curva saudável dos vídeos 8 e 9

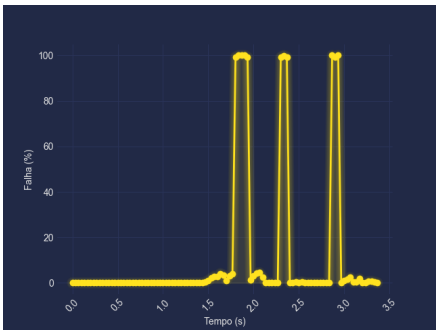
(a) Porcentagem de falhas: vídeo 8



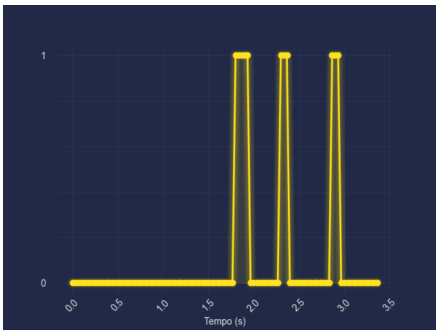
(b) Gráfico de interrupção total: vídeo 8



(c) Porcentagem de falhas: vídeo 9



(d) Gráfico de interrupção total: vídeo 9



Fonte: autoria própria.

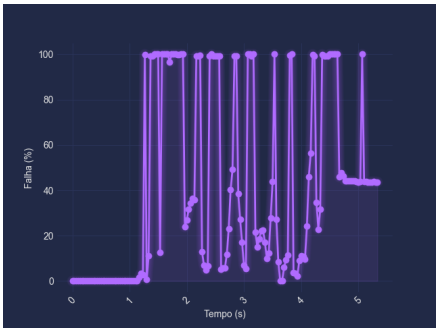
A pista curva também foi recortada para gerar falhas e foram adquiridos dois vídeos, 10 e 11, do AGV fazendo o percurso, sendo que, em um deles, o veículo partiu da posição inversa. No vídeo 10, o AGV demorou, aproximadamente, um segundo para partir. Durante esse período de tempo, ele ficou sobre uma parte da faixa guia que não havia falhas, ou seja, estava saudável. Como é possível observar na Figura 14(a), que representa o gráfico de porcentagem de falha, durante o primeiro segundo, os *frames* tiveram um resultado percentual entre 0 e 3%, mostrando que o software foi capaz de reconhecer a ausência de defeitos nesse trecho da pista, o que também é indicado no gráfico da Figura 14(b).

Já no início do vídeo 11, o veículo autoguiado ficou, aproximadamente, 2 segundos parado sobre o seu ponto de partida, que, por apresentar um contraste entre o fundo preto e o início da faixa, deveria ser classificado como uma interrupção total. Percebe-se por meio da Figura 14(c), correspondente ao gráfico de porcentagem de falha, que o serviço indicou um

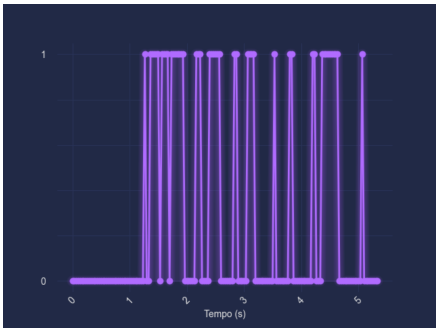
percentual acima de 99%, por consequência, o gráfico 14(d) detectou a interrupção total da forma que foi proposta na construção do módulo.

Figura 14 – Gráficos resultantes para uma pista em curva com falhas dos vídeos 10 e 11

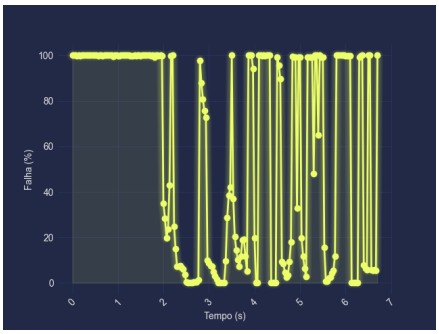
(a) Porcentagem de falhas: vídeo 10



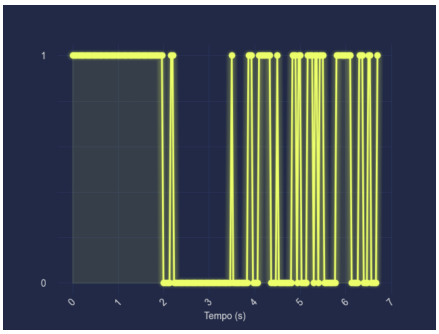
(b) Gráfico de interrupção total: vídeo 10



(c) Porcentagem de falhas: vídeo 11



(d) Gráfico de interrupção total: vídeo 11



Fonte: autoria própria.

Se comparados com os gráficos de porcentagem e interrupção da pista reta com falhas, os resultados mostrados na Figura 14 tiveram menos resultados falsos. Percebe-se isso pela diminuição da plotagem de interrupções totais em trechos do vídeo que não contém descontinuidades.

Com base no primeiro teste, é possível determinar que, para diferentes condições de funcionamento do AGV, o serviço desenvolvido pode ter seu desempenho diminuído. Assim, é necessário aprofundar as pesquisas nos filtros utilizados a fim de que o software seja mais eficaz na detecção de falhas sob condições diversas.

O objetivo do segundo experimento foi avaliar a exatidão do serviço de software em retornar a porcentagem de falha em pistas. Para realizar esse teste, foram montados *frames* de pistas com porcentagens conhecidas de falhas, com base na área de pixels de referência para o cálculo das porcentagens. Essas imagens foram criadas já binarizadas, com o mesmo tamanho das imagens geradas pela câmera. O objetivo foi criar imagens-padrão, ou seja, imagens cuja área em falha é conhecida, de forma a testar o módulo de software proposto. Essas imagens são mostradas na Figura 15(a). A moldura na cor cinza foi utilizada apenas para separar claramente as imagens. Apenas a área em pixels pretos ou brancos foi analisada pelo software.

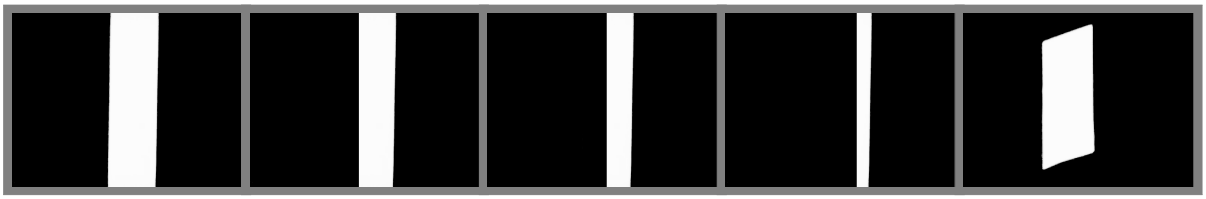
Foram criadas imagens contendo áreas com diferentes porcentagens de falha na pista. Essas porcentagens incluíram 0%, 25%, 50% e 75% de desgaste. Adicionalmente, uma simulação de interrupção total foi realizada, na qual a porcentagem de falha é superior a 99%. Os

frames contendo essas diferentes porcentagens foram submetidos ao módulo de software em análise.

Para o início das análises, o *frame* demonstrado na Figura 15(a) contendo 0% de falha foi selecionado como referência, utilizando uma pista completamente saudável de um dos vídeos registrados pela câmera do AGV analisados. Ao submeter esse *frame* ao serviço de software, o resultado retornado foi de 0% de falha, o que demonstrou a capacidade do serviço de software em identificar corretamente a ausência de falhas.

Figura 15 – Imagens utilizadas para o experimento de validação da exatidão do serviço de software em retornar a porcentagem de falha em pistas

(a) Saudável **(b) 25% de falha** **(c) 50% de falha** **(d) 75% de falha** **(e) Interrompida**



Fonte: autoria própria.

A Figura 15(b) demonstra a imagem de pista com 25% de falha criado para avaliação a partir da modificação da imagem utilizada anteriormente, a qual originalmente apresentava 0% de falha. Ao submeter esse novo *frame* ao serviço de software, o resultado retornado foi de 24,9715% de falha, considerando o cálculo de porcentagem retornado pelo software, apresentando um erro de aproximadamente 0,03%. A diferença entre o valor calculado pelo software (24,9715%) e a porcentagem real de falha (25%) é relativamente pequena, demonstrando uma exatidão satisfatória do serviço nesse teste com um valor baixo. No *frame* com 50% (Figura 15(c)) de falha na pista, a porcentagem retornada pelo serviço foi de (45,6054%), apresentando um maior erro (4,3946%) em relação a imagem de porcentagem menor. A imagem da Figura 15(d), com 75% de desgaste na pista foi submetido ao software. O resultado retornado foi de 69,9238%, com erro de 5,0762%.

Com base nos dados analisados, pode-se inferir que conforme a porcentagem de falha na pista aumenta, o erro também cresce. Desse modo, o software apresenta melhor comportamento em pistas com menor porcentagem de desgaste, além de ser capaz de retornar 0% para pistas com nenhuma porcentagem de falha. Ademais, foi analisado o *frame* com interrupção total da pista (Figura 15(e)), ou seja, com uma porcentagem maior que 99% de falha. O retorno do software foi de 99,40% de falha, alertando corretamente um desgaste que interrompe a pista.

6 CONCLUSÃO

AGVs são muito utilizados no processo logístico por oferecerem rapidez, eficiência e por conseguirem transportar diversos produtos e materiais a vários pontos diferentes. Veículos do tipo seguidores de linha percorrem um caminho que pode ser demarcado com faixas pintadas ou fitas colocadas no chão de fábrica. Entretanto, como a faixa está no chão de fábrica, o acúmulo

de sujeiras ou a queda de objetos sobre a linha impedirá o AGV de ler a faixa, pois a mesma estará com falhas. Visando solucionar esse problema, foi construído um módulo de software para ser instalado em AGVs já existentes capaz de identificar as falhas presentes na pista a partir da câmera presente nos veículos.

Para a análise do desempenho do módulo de software, foram utilizados 11 vídeos que geraram 22 gráficos, sendo divididos em dois tipos, gráfico de porcentagem de falha presente na pista e gráfico de interrupção total. Além desse, foram realizados mais dois testes, um deles consistia em uma análise da exatidão do módulo de software ao retornar a porcentagem de falhas na pista, onde foram testadas pistas com 0%, 25%, 50%, 75% de falha e outra com uma interrupção total. Os resultados obtidos demonstraram que quanto maior a porcentagem de falha na pista, maior o erro gerado.

A partir de uma revisão da literatura, foi observado que não há qualquer estudo que trate sobre medição do desgaste da pista, portanto a criação desse trabalho pode auxiliar como base para a criação de projetos futuros nessa área. É importante destacar que a velocidade, a iluminação, a câmera, a presença de ruídos e a condição da fita usada como pista são alguns dos fatores que podem influenciar nos resultados da identificação de falhas. Nos sistemas de Visão Computacional dos veículos autoguiados, esses são itens que podem levar à necessidade de alteração dos parâmetros definidos para os filtros usados no tratamento de imagem.

O software foi documentado utilizando Linguagem de Modelagem Unificada e construído de forma a possibilitar que um usuário que deseja fazer a instalação do módulo no seu AGV, mesmo que não tenha tido participação na sua construção, entenda e consiga adaptar o serviço conforme a sua necessidade. Portanto, para projetos futuros, podem ser realizados estudos que se aprofundem no uso dos filtros utilizados, além de buscar formas de utilizar uma binarização adaptativa para melhorar o processamento da imagem. Ademais, podem ser pesquisadas e desenvolvidas formas de indicar a localização de falhas em uma pista, introduzindo um sistema que opere em conjunto com um serviço de odometria, por exemplo, diminuindo o tempo gasto por um funcionário que necessita percorrer a faixa para localizar o desgaste.

REFERÊNCIAS

ALMEIDA, L. **Veículo Auto Guiado (AGV -Automated Guided Vehicle) - Protótipo Seguidor de Linha**. 2016. 83 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia Elétrica, Centro Federal de Educação Tecnológica de Minas Gerais, Minas Gerais.

BANDINELLI, Giovana Cunha. **Análise de implementação de AGVS em ambiente industrial.** 2018. 37 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia de Controle e Automação, Universidade Federal do Rio Grande do Sul, Porto Alegre, Rio Grande do Sul.

BELO, João Renato Fialho de. **Análise do desempenho de movimentação de carga com plataforma: seguidor de linha ou trilho.** 2016. 37 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia de Produção, Universidade Federal do Pampa, Bagé, Rio Grande do Sul.

BOOCH, Grandy; RUMBAUGH, James; JACOBSON, Ivar. **UML: Guia do Usuário.** 1. ed. Rio de Janeiro, RJ, Brasil: Campus / Elsevier Brasil, 2006. 552 p.

DA MOTA BRITO, José Marcos et al. Aplicação de veículo guiado automaticamente nas diversas áreas de produção da indústria: revisão sistemática da literatura. **Brazilian Journal of Development**, v. 6, n. 2, p. 9486–9502, 2020.

DEWANTORO, Gunawan; MANSURI, Jamil; SETIAJI, Fransiscus Dalu. Comparative study of computer vision based line followers using raspberry pi and jetson nano. **Jurnal Rekayasa Elektrika,** v. 17, n. 4, 2021.

DREGER, Rubem Sprenger. **Construção e avaliação do desempenho de um veículo autoguiado: AGV, de baixo custo, para uso em ensino e pesquisa.** 2001. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Porto Alegre, RS, Brasil.

ERL, Thomas. **Service-Oriented Architecture: Concepts, Technology, and Design.** Upper Saddle River, NJ, USA: Prentice Hall PTR, 2005. 792 p.

ERL, Thomas. **SOA: Principles of Service Design.** 1. ed. Upper Saddle River, NJ, EUA: Prentice Hall PTR, 2007. 608 p. Prentice Hall Service-Oriented Computing Series.

FUGITA, Henrique Shoiti. **MAPOS: método de análise e projeto orientado a serviços.** 2009. Dissertação (Mestrado em Engenharia de Computação) — Escola Politécnica da Universidade de São Paulo, São Paulo, SP, Brasil.

GIRBÉS, Vicent; ARMESTO, Leopoldo; TORNERO, Josep. Pisala project. intelligent sensorization for line tracking with artificial vision. In: VDE. **Anais ISR 2010 (41st International Symposium on Robotics) e ROBOTIK 2010 (6th German Conference on Robotics).** Munique, Alemanha, 2010. p. 1–6.

GROOVER, Mikell P. **Automation, Production Systems, and Computer-Integrated Manufacturing.** 3. ed. Upper Saddle River, NJ, EUA: Pearson Prentice Hall, 2008. 736 p.

GUEDES, Gilleanes TA. **UML 2 - Uma abordagem prática.** 1. ed. São Paulo, SP, Brasil: Novatec Editora, 2018. 400 p.

GUIMARÃES JÚNIOR, Carlos Solon Soares. **Proposta de um framework baseado em arquitetura orientada a serviços para a robótica.** 2015. Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Porto Alegre, RS, Brasil. Disponível em: <https://hdl.handle.net/10183/132636>.

PAGE-JONES, Meilir. **Fundamentals of Object-Oriented Design in UML.** 1. ed. Boston, MA, EUA: Addison-Wesley Professional, 1999. 336 p.

LASATER, Christopher G. **Design Patterns.** 1. ed. Burlington, MA, EUA: Jones & Bartlett Publishers, 2006. 512 p.

MENDES, J. Marco et al. Service-oriented control architecture for reconfigurable production systems. In: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (IEEE). **Anais da 6º IEEE International Conference on Industrial Informatics (INDIN).** Daejeon, South Korea, 2008. p. 744–749. 13–16 July 2008.

MONTEIRO, Matheus Martins. **Projeto de instrumentação e controle de um veículo autoguiado.** Campina Grande, PB, Brasil, 2018.

NAMETALA, Ciniro Aparecido Leite; NUNES, Gustavo Henrique; DO AMARAL, Petrúcio Júnio. Análise comparativa de tempos de execução de código r interpretado e compilado em sistemas windows e linux. In: SEMINÁRIO INTERNACIONAL DE ESTATÍSTICA COM R, 3., 2018, Niterói. **Anais do III Seminário Internacional de Estatística com R.** Niterói: Universidade Federal Fluminense, 2018.

NASCIMENTO, Elaine et al. Tecnologia na movimentação de materiais: uma ferramenta de auxílio na redução de custos. **Revista Diálogos Interdisciplinares,** v. 8, n. 5, p. 132–143, 2019.

NESTERUK, Dmitri. **Design Patterns in Modern C++: Reusable Approaches for Object-Oriented Software Design.** 1. ed. Berkeley, CA, EUA: Apress, 2018. 336 p.

PUPPIM DE OLIVEIRA, Diogo; PEREIRA NEVES DOS REIS, Wallace; MORANDIN JUNIOR, Orides. A qualitative analysis of a usb camera for agv control. **Sensors,** MDPI, v. 19, n. 19, p. 4111, 2019.

DE OLIVEIRA, Sérgio. **Internet das coisas com ESP8266, Arduino e Raspberry PI.** 1. ed. São Paulo, SP, Brasil: Novatec Editora, 2017. 256 p.

REIS, Wallace Pereira Neves dos. **Um controlador Fuzzy-Adaptativo em cascata com Multi-Sensores para o aumento da exatidão na posição do AGV seguidor de linha.** 2023. Tese

(Doutorado em Ciência da Computação) — Universidade Federal de São Carlos, São Carlos, SP, Brasil.

DOS REIS, Wallace Pereira Neves; MORANDIN JUNIOR, Orides. Sensors applied to automated guided vehicle position control: A systematic literature review. **The International Journal of Advanced Manufacturing Technology,** Springer, v. 113, n. 1-2, p. 21–34, 2021.

SANTOS, Eduardo António da Silva. **Logística baseada em AGVs.** 2013. Dissertação (Programa em Engenharia Eletrotécnica e de Computadores) — Universidade do Porto, Porto, Portugal.

SHALLOWAY, Alan; TROTT, James R. **Design Patterns Explained: A New Perspective on Object-Oriented Design.** Boston, MA, USA: Pearson Education, 2004.

SOUZA, L. F. F. de. **Arquitetura de Software Orientada a Serviços para AGV.** 2021. 80 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia de Computação, Universidade Federal de São Carlos, São Carlos, São Paulo.

SOUZA, R. S. F. **Monitoramento contínuo de temperatura utilizando Raspberry PI e Zabbix.** 2022. 53 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia Eletrônica, Universidade de Brasília, Brasília, Distrito Federal.

TONIN, Carlos Miguel Prescendo. **Desenvolvimento de um sistema de navegação para sistemas AGV através do método SLAM.** 2018. 48 p. Trabalho de Conclusão de Curso — Faculdade de Engenharia de Controle e Automação, Universidade Federal do Rio Grande do Sul, Porto Alegre, Rio Grande do Sul.

ULLRICH, Günter. **Automated guided vehicle systems: A primer with practical applications.** Heidelberg: Springer Berlin, 2015.

ZHANG, Houzhong et al. Research on Guide Line Identification and Lateral Motion Control of AGV in Complex Environments. **Machines,** MDPI, v. 10, n. 2, p. 121, 2022.