ESTUDO E APLICAÇÃO DE LOCALIZAÇÃO E MAPEAMENTO SIMULTÂNEOS EM AMBIENTES INDUSTRIAIS[1]

Thiago Lablonsk de MEDEIROS[2]
Aluno de Engenharia de Controle e Automação/IFSP
Aluno/IFSP

Muriell de Rodrigues e FREIRE[3]
Doutor em Engenharia Elétrica/UNIFEI
Professor/IFSP

RESUMO
Os sistemas inteligentes são ferramentas poderosas que estão revolucionando o mundo como o conhecemos. O veículo autônomo faz parte da tecnologia que vai revolucionar o mundo como o conhecemos onde novas técnicas podem ser testadas, combinadas e assim melhores e otimizados sistemas de navegação não tripulados. O mesmo está acontecendo em fábricas ou indústrias onde a tendência é que tudo seja automatizado, inclusive os veículos de transporte de insumos. O ideal seria um sistema sem qualquer auxílio de dispositivos como GPS e que fosse robusto (seja imune a falhas ou trabalhasse com desempenho semelhante ao humano), além de responder bem a diferentes ambientes que possam conter ruídos (eletromagnéticos, acústicos, luminosos, etc.). Para a indústria, o sistema precisa ser robusto, pois a indústria é um ambiente cheio de perturbações, onde pessoas, ou outros robôs, motores, construções com alvenaria grossa podem ser encontrados em todos os lugares e isso pode comprometer aplicações que envolvam o uso de GPS, é necessária uma adaptação na planta para o correto funcionamento dos sistemas não tripulados. A Localização e Mapeamento Simultâneos ou Localização e Mapeamento Simultâneos (SLAM) pode ser uma boa alternativa, pois não depende de nenhum dispositivo externo, apenas de sensores locais, como câmera e sensores de distância. Este trabalho descreve um cenário hipotético onde um veículo deve trafegar e mapear o local por onde passar e gerar um mapa a partir das informações coletadas e processadas. Os dados aqui apresentados podem contribuir para trabalhos futuros.
Palavras-chave: Localização e Mapeamento Simultâneos, Sensores, Algoritmo, Navegação, Simulação, Veículo autônomo.

APPLICATION OF THE BAYESIAN OTIMIZATION ALGORITHM WHITH MUTATION ON THE PLANNINGTHE EXPANSION OF POWER TRANSMISSION SYSTEMS

ABSTRACT
This work aims to apply the Bayesian Optimization Algorithm (BOA) to the Transmission Expansion Planning (PET) problem. The formulation of PET will be based on short term planning, considering the "N-0" safety criteria. The best solutions to the problem were found through a specialized evolutionary algorithm and used as reference for the implementation of the

[1] TCC do Curso de Engenharia de Controle e Automação – *Campus* São João da Boa Vista.
[2] thiagolablonsk@gmail.com
[3] muriell@ifsp.edu.br

BOA algorithm. From an adaptation of the BOA, a mutation operator was implemented to generate more diversity and thus improve searches. The analyses were made comparing the original BOA with the modified version. The comparison between the algorithms was done using a graphical method (convergence graph) comparing the cases for discussion. The algorithm can be used not only for PET, but also for other problems that involve evolutionary algorithm Smart systems are powerful tools that are revolutionizing the world as we know it. Autonomous vehicle is part of the technology that will revolutionize the world as we know it where new techniques can be tested, combined and thus improving and optimizing unmanned navigation systems. The same is happening in factories or industries where the trend is for everything to be automated, including supplement transport vehicles. The ideal system would be a system without any aid from devices such as GPS and be robust (either immune to failures or minimally like humans), in addition to responding well to different environments that may contain noise (electromagnetic, acoustic, lighting, etc). For the industry, the system needs to be robust, because the industry is an environment full of disturbances, where people or other robots, engines, construction with thick masonry can be found everywhere and this can compromise applications that involve the use of GPS, being an adaptation in the plant is necessary for the correct functioning of unmanned systems. The Simultaneous Location and Mapping or Simultaneous Localization And Mapping (SLAM) can be a good alternative as it does not depend on any external device, only local sensors, such as camera and distance sensors. This work describes a hypothetical scenario where a vehicle must travel and map the place where to pass and generate a map from the information collected and processed. The data presented here may contribute to future work.
KEYWORDS: Simultaneous Localization and Mapping, Sensors. Algorithm, Navigation, Simulation, Autonomous vehicle.

Introdução

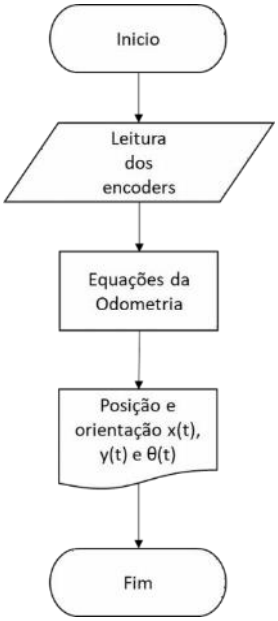
A introdução de sistemas inteligentes dentro de ambientes industriais é cada dia mais comum, e isso está se tornando uma necessidade, pois a demanda por produtos e bens de consumo só aumenta. Sistemas inteligentes são sistemas baseados em conhecimento para resolver problemas, sendo capaz de manipular informações de forma inteligente (REZENDE, 2003). Uma das áreas da Inteligência Artificial (IA) tenta tirar pessoas de atividades repetitivas e perigosas para remaneja-las para trabalhos cada vez mais nobres que precisam de criatividade e que tratam de incerteza, como a criação de algoritmos (SCHWAB, 2016). Isto envolve também redução dos custos, pois não necessitam de retrabalho e podem trabalhar o tempo todo, precisando apenas de manutenção e investimentos que são recuperados a longo prazo. A implementação de veículos autônomos na quarta revolução industrial já é uma realidade em algumas empresas, para movimentar cargas sem a necessidade de um operador (veículo não tripulado), onde tudo é gerenciado por um algoritmo inteligente. Algoritmos de navegação em sua maioria utilizam o GPS para se localizar e se mover, ou dependendo da aplicação é feita uma adaptação no local, para que o algoritmo possa executar rotinas e processar informações para conseguir tomar decisões de forma inteligente, porém,

algumas aplicações como em pouso de “drones” o uso do GPS não é uma boa opção, existem falhas na comunicação (YANG, 2018), o que pode comprometer a integridade do sistema ou dos dados coletados que são utilizados durante o processo de navegação, provocando acidentes. Na indústria o problema é semelhante, onde paredes e máquinas, entre outros, podem afetar o sinal do GPS, ou eventos que ocorrem próximos são capazes de afetar o desempenho do sistema e não estão previstas na programação do robô. Pensando nos problemas que podem surgir, aplicar o SLAM dentro de um ambiente industrial pode ser uma boa escolha para tornar o processo robusto, seguro e ainda ser uma solução de baixo custo, porque não é necessário adaptar a planta para implementar um veículo autônomo com SLAM.

A principal diferença da técnica de localização e mapeamento é que nenhum mapa é fornecido ao algoritmo, sua tarefa é descobrir através de sensores a própria localização, que deve construir um mapa em tempo real, sendo um problema muito estudado com nome de Localização e Mapeamento Simultâneos (LEMS) (NORVIG, 2013). No campo de estudos sobre navegação de robôs autônomos existe a odometria que é um campo de estudos forte em mapear o ambiente, com o movimento incremental através de sensores encoders para determinar sua localização (GADIOLI, 2017), também existem aplicações que utilizam imagens para realizar a odometria, mas que não será abordada neste trabalho. A Figura1 ilustra como GADIOLI, implementou a odometria que é chamado por interrupção a cada 30 milissegundos. À medida que o robô se movimenta é feito uma contagem dos pulsos para determinar a distância percorrida pelo robô ou para que lado ou o sentido em que o robô se encontra, isso facilita a construção do mapa e melhora a robustez do processo.

No campo de estudo de técnicas de Odometria Visual (OV) existe um portal online onde são divulgados procedimentos e resultados de OV (Karlsruhe, 2021), sendo o SLAM um destes mecanismos. Para OV detectar a movimentação é necessário analisar uma sequência de imagens e observar translação e rotação de um determinado ponto (YOUSIF, 2015). Este ponto é determinado utilizando técnicas como a geometria epipolar que consiste em duas câmeras que captam o mesmo ponto, sendo assim duas imagens em perspectiva de um único objeto (ZHANG, 1998), mas também existem técnicas que utilizam uma câmera.

Figura 1- Diagrama do algoritmo de cálculo da odometria.



Fonte: Gadioli (2017, p. 2).

O SLAM foi proposto pela primeira vez por Smite em 1986 (SMITH, 1986), a proposta ajudou a criar aplicações de realidade aumentada que também pode ser utilizada na robótica. De acordo com Stachniss (STACHNISS, 2009) o problema que o SLAM tenta responder é: Como o mundo é? Onde estou? E como posso chegar à determinado lugar? (MACARIO BARROS, 2022). O SLAM tenta responder a estas perguntas utilizando diversos algoritmos para processar informações que podem ser obtidas por sensores de diversos tipos.

O SLAM trabalha com a premissa de que não existe nenhum mapa para o robô inicialmente é a única informação é a distância de barreiras físicas em sua volta, sua posição e orientação inicial é zero. Sendo assim o SLAM trabalha com a localização local e não a global, como é realizado no GPS. A partir da determinação de pontos dados pelos sensores é possível construir um mapa e em seguida estimar a localização do robô no mapa construído, e tudo é feito em tempo real (tempo de execução). Todo processo começa pela inicialização onde variáveis são definidas (como calibração em sistemas que utilizam câmeras para fotografar o caminho percorrido), rastreamento (onde é estimado a pose dos sensores) e por fim o mapeamento onde é construído um mapa em tempo real

do que está em volta do robô (TAKETOMI,2016). O mapa pode ser em duas dimensões (2D) ou em três dimensões (3D). O mapa em 3D possui detalhes de relevo e quando se utiliza câmeras é possível construir modelos do mundo real, pois além de detalhes de relevo é possível adicionar cores, mas também alguns algoritmos estudam o caminho percorrido e fornecem somente o caminho e comparam com o do GPS para estudar a precisão do algoritmo.

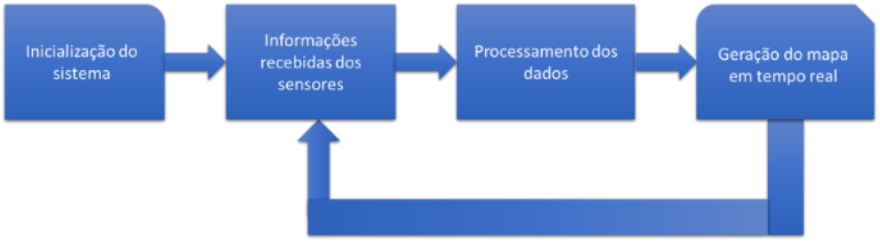
Para estimar as poses (local em que o robô está em cada instante de tempo) do SLAM é necessário a utilização de filtros como “Extended Kalman” (EKF) e Bayesiano (FILIPENKO, 2018).

O EKF é uma versão não linear do Filtro Kalman, onde o algoritmo faz uso de uma série de medições, incluindo, imprecisões, ruídos estatísticos e produz uma estimativa de variáveis incertas (desconhecidas), assim estimando uma distribuição de probabilidades conjunta em cada estado (instante de tempo).

O filtro Bayesiano classifica informações, através de dados estatísticos e atribui probabilidades, utilizando de inferências que dependem de um banco de dados para que o filtro funcione. Muito utilizado em e-mails. A primeira etapa (para o SLAM) de do filtro Bayesiano são previsões, utilizando informações de estados anteriores e o estado atual do sistema (veículo), comandos de controle de entrada (atuadores). A segunda etapa é a atualização de medição onde acontece a combinação das informações dos sensores com o estado previsto (FILIPENKO, 2018).

A Figura 2 mostra como é de forma simplificada o funcionamento do SLAM (baseado em visão e distância).

Figura 2- Processo de criação e atualização do mapa (SLAM) em tempo real.



Fonte: Elaboração própria.

Existem várias técnicas e algoritmos SLAM. Às duas técnicas mais comuns são a

utilização de sensores de proximidade como o Lidar (sensor de distância a “laser”) e o Visual (utilizando câmeras). Para se aplicar o SLAM é importante que não se tenha nenhuma informação prévia da localização do robô. Partindo da premissa da não existência de um mapa, é iniciado a etapa de construção de um mapa do ambiente em 2D ou 3D do local de onde o robô se encontra (YOUSIF, 2015), realizando cálculosbaseando-se nas informações fornecidas por sensores.

Os sensores dão a capacidade do robô autônomo de “sentir” o ambiente a sua volta (CHONG, 2015), então a escolha do sensor adequado para cada cenário afeta diretamente a capacidade que o robô tem de observar o que está ao seu redor e para isto existem diversos sensores. Para navegação de robôs autônomos sensores acústicos são melhor empregados debaixo d’água, diferente de sensores visuais e a laser (CHONG, 2015). Telêmetros a laser por outro lado são mais populares e robustos em ambientes internos e externos com alta velocidade e precisão (CHONG, 2015), no caso do Lidar é o sensor mais encontrado na literatura. Existem vários sensores de visão, como o RGB obtém informações de distâncias a partir de imagens com regiões texturizadas (geralmente é marcado pontos geométricos na imagem a cada varredura). Para que o robô consiga estimar as poses é necessário que o algoritmo encontre pontos de interesse na imagem.

Nos resultados esses pontos serão apresentados. Para sensores de visão existem também o RGB-D que são capazes de identificar a profundidade através da projeção de uma luz no espectro do infravermelho que é captado por outro sensor, assim é possível determinar a distância que o sensor se encontra do objeto, porem este é sensível a luz externa. Um sensor RGB-D muito utilizado é o Microsoft Kinect, por ser robusto e ser muito citado em diversos artigos prometendo ser promissor para a aplicação SLAM.

Como já mencionado técnicas SLAM mais comuns são utilizando o sensor Lidar (sensor de distância a “laser”) e o visual (utiliza câmeras), mas também existem aplicações que tentam misturar os métodos com a finalidade de aumentar a robustez do método. Também existem outras aplicações de realidade aumentada, como o Google Tango e o Structure Sensor, mas não é o foco deste trabalho.

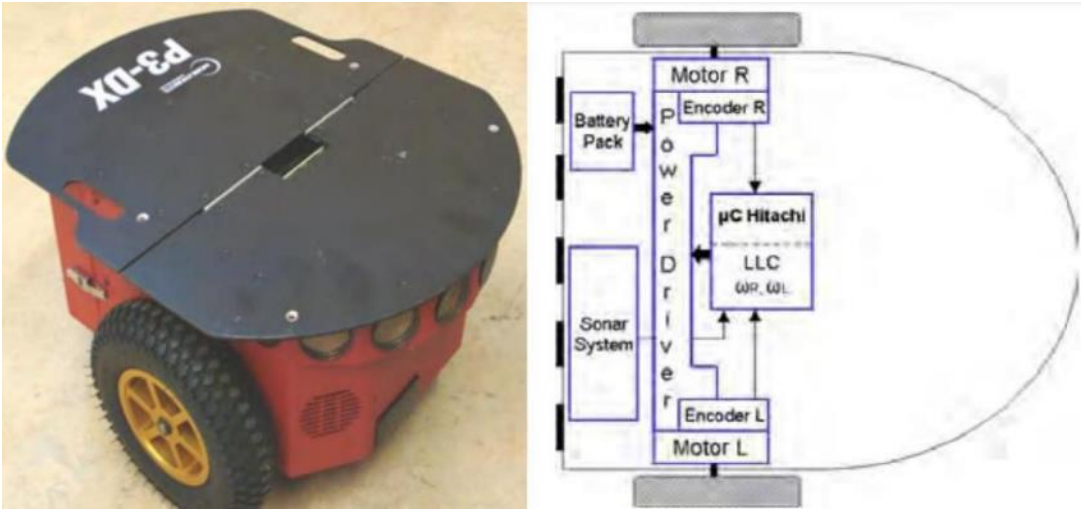
No presente trabalho, além de um estudo sobre o tema SLAM, deve-se verificar a factibilidade da implementação de algoritmo SLAM Visual e do Lidar dentro de um ambiente simulado (ambientes de realidade virtual 3D similares ao de uma fábrica). A aplicação deve fornecer informações e conseguir gerar um mapa que condiz com a

realidade da simulação. Ademais, deve-se verificar quais influências o ambiente tem sobre o robô que utiliza o SLAM. Ao final, confrontar dados teóricos obtidos em artigos com informações experimentais a partir da realização de testes (simulação).

Desenvolvimento

Para o estudo e a coleta de dados, foi utilizado o robô Pioneer P3, que geralmente é encontrado em experimentos, por estar disponível na plataforma CoppeliaSim e por se tratar de um robô simples e de uso didático (Figura 3) que possui sensores como encoder e atuadores, como dois motores. O modelo físico possui baterias e uma estrutura compacta para protótipos.

Figura 3 – Robô Pioneer e esquema interno.



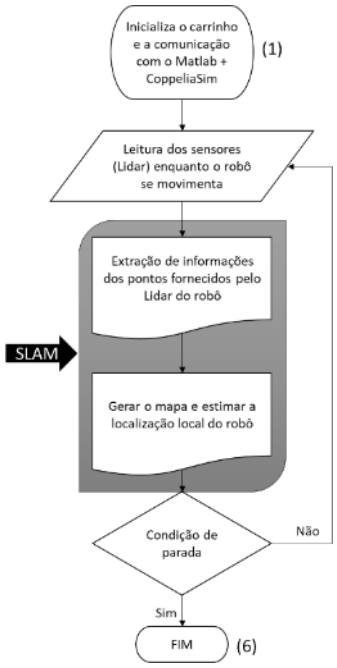
Fonte: Espinosa (2010).

Para o estudo do SLAM Lidar foi utilizado o projeto feito no Matlab RMAproject (BARBOSA, 2017). Este projeto permite executar o SLAM Lidar, onde o robô percorre pela planta, enquanto gera o mapa e ao final salvar o mapa gerado em cadateste, basta inserir o robô em robô em um ambiente virtual do CoppeliaSim e executar o projeto. O mapa é gerado Matlab enquanto caminha pelo ambiente simulado. A leitura éfeita por sensores de distância que enviam informações do CoppeliaSim em tempo real para o Matlab. A Figura 4 apresenta um ambiente teste, do qual será gerado um mapa e aFigura 5 um fluxograma desta aplicação.

Figura 4 - Planta base para testes iniciais (planta 1).

Fonte: Elaboração própria.

Figura 5 - Fluxograma SLAM - Lidar.

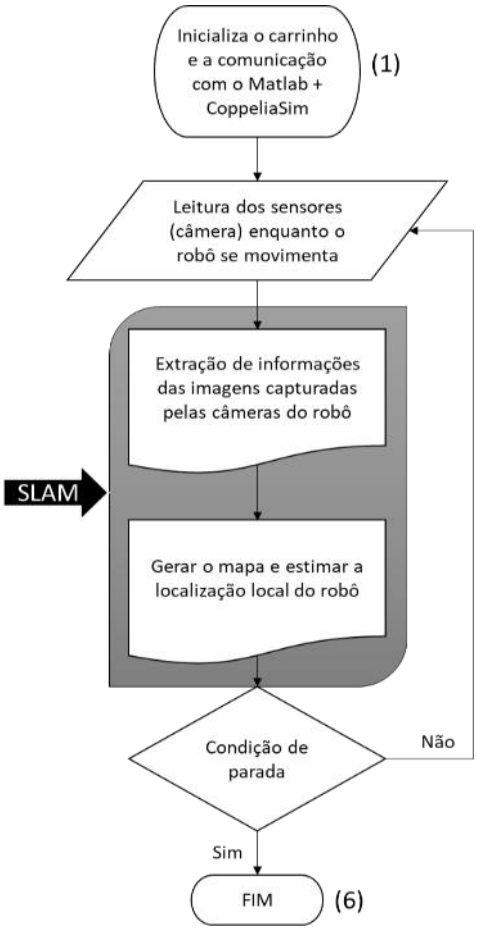


Fonte: Elaboração própria.

Com a finalidade de evitar retrabalho, para o vSLAM (SLAM visual) foi necessário criar um supervisório (simples) para controlar o robô e o código captura a

imagens da câmera instalada no robô a uma taxa de 1 segundo ou 1 Hz e salva as imagens em uma pasta. Após a coleta de imagens os arquivos (imagens) salvos são lidos por um "script" do Matlab usando como base o código disponível na plataforma MatWorks (MathWorks, 2022) que gera os pontos na imagem e calcula o deslocamento e por fim gera um mapa com o percurso percorrido. A Figura 6 mostra o fluxograma para este processo.

Figura 6 - Fluxograma vSLAM



Fonte: Elaboração própria.

A Figura 6 (para melhor visualização o autor enumerou apenas o primeiro passo e o último), mostra passos do 1 e 6 (no caso o SLAM lê as imagens diretamente de uma pasta onde estão armazenadas as imagens de todo o trajeto) para executar o vSLAM. Os passos são:

- **Primeiro passo:** inicialização do carrinho, onde toda a parte de sensores e atuadores (motores) são iniciados, onde também é feita a comunicação do

simulador (CoppeliaSim) com o Matlab utilizando a DLL disponibilizada pelo CoppeliaSim.

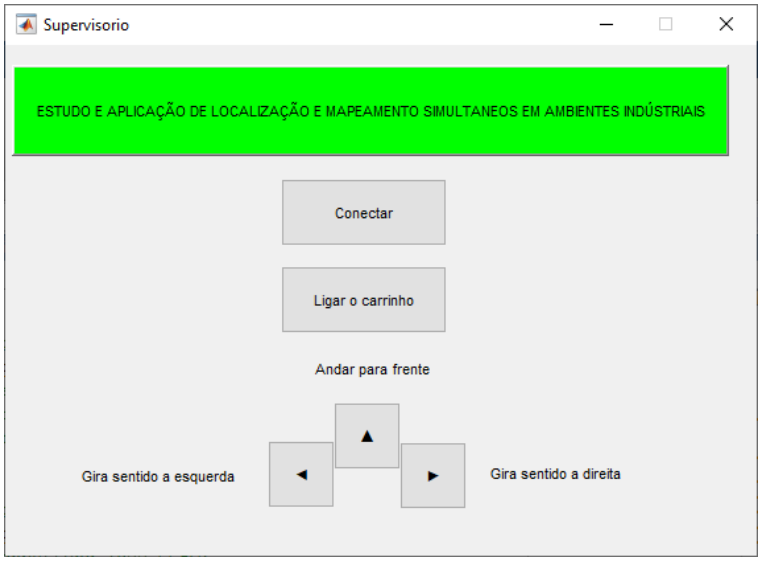
- **Segundo passo:** o robô coleta imagens enquanto se move pelo ambiente simulado.
- **Terceiro passo:** as imagens recebidas pelas câmeras, são processadas no Matlab.
- **Quarto passo:** após o processamento das imagens é gerado parte do mapae estimado a localização do robô utilizando o mapa.
- **Quinto passo:** uma condição de parada deve ser estabelecida, saindo de um local e voltar para o mesmo local, formando um "loop" (YOUSIF, 2015). Caso a condição não seja satisfeita o algoritmo volta para o passo 2.
- **Sexto passo:** caso a condição de parada 5 seja satisfeita o robô pode ser desligado e a coleta de dados é encerrada, o mapa não deve ser mais atualizado.

A condição de parada "loop" é uma técnica utilizada para se estimar o erro gerado ao longo da execução do sistema, geralmente é utilizada como um dos parâmetros para comparar eficiência entre algoritmos (no caso o erro é uma das medidas) SLAM. O processo da geração do mapa para o vSLAM e o SLAM com lidar é semelhante, diferindo do sensor utilizado e da técnica utilizada para estimar as posições, já os sensores fornecem informações diferentes (distância e imagem).

As ferramentas utilizadas foram o Matlab (MATLAB, 2021), CoppeliaSim (ROHMER, E. et al.) e uma DLL ("Dynamic-link library" - pacote de ferramentas em forma de códigos que executam uma instrução) do próprio CoppeliaSim para realizar a comunicação entre os dois recursos.

A Figura 7 mostra o supervisório criado pelo autor para navegar com o robô durante a coleta de imagens, seu funcionamento consiste em clicar nas "setas" onde pode-se girar o robô sentido a esquerda, direita e seguir em frente. O supervisório simples só para implementar os comandos, foi construído no próprio Matlab utilizando a toolbox gráfica disponibilizada pela ferramenta. Com isso é possível coletar as imagens que podem ser utilizadas no vSLAM, evitando retrabalho.

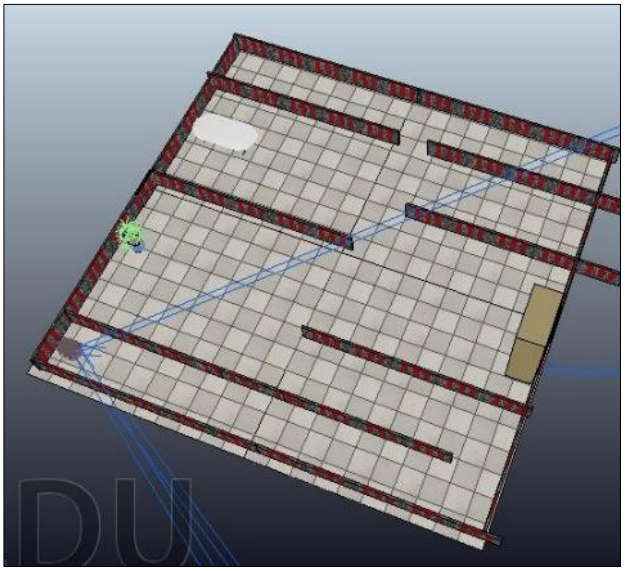
Figura 7 – Supervisório.



Fonte: Elaboração própria.

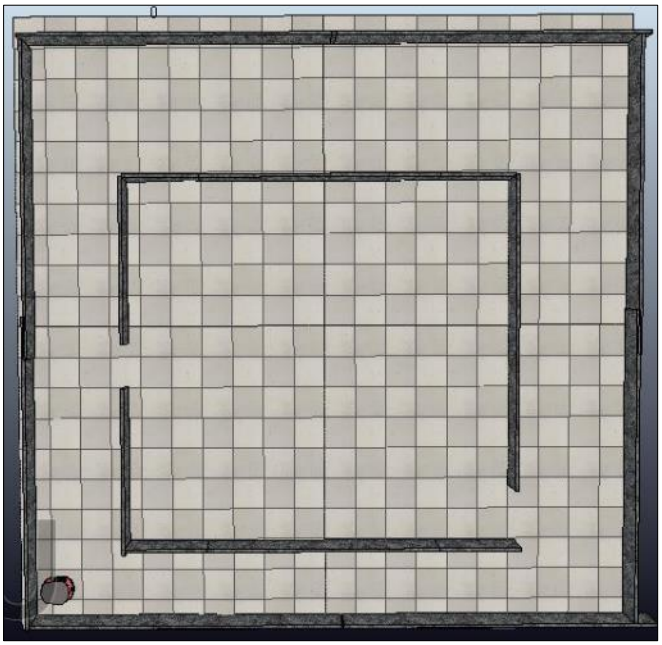
Foram utilizadas três plantas, para estudar os resultados, sendo eles Figura 4, 8 e 9. A planta 3 é referência à uma parte de uma planta onde o autor já trabalhou, então o robô teria que ser capaz de mapear o ambiente, pois se trata de um local similar ao que seria encontrado fora da simulação. O robô utilizado é o “pioneer p3” do próprio simulador (CoppeliaSim).

Figura 8 – Planta 2.



Fonte: Elaboração própria.

Figura 9 – Planta 3.



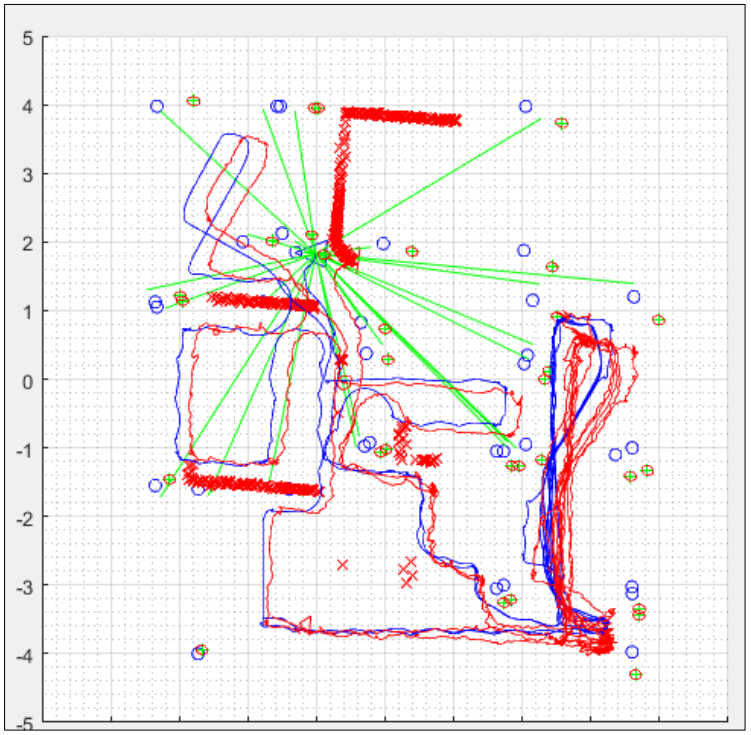
Fonte: Elaboração própria.

Resultados obtidos

A Figura 10 e 11 são resultado do mapeamento utilizando o Lidar, o caminho percorrido pelo robô é a linha em azul e a estimada é a vermelha e tudo está registrado nográfico (Figura 10) o mapa gerado (Figura 11) é o resultado do mapa estimado. O que pode ser observado é que o resultado da Figura 11 se mostra similar a planta (Figura4), “real”, porém ainda à muitos pontos de incerteza, pois o mapa contém regiões “borradas”onde existem espaços em branco e deveria existir uma parede, reforçando ainda mais a inconsistência dos dados.

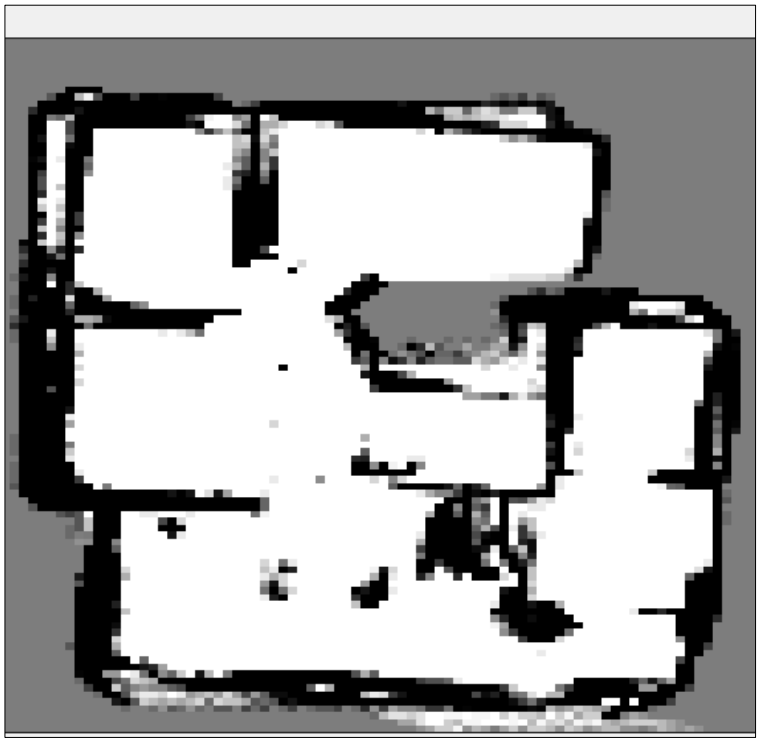
Para testar melhor o método, foram criadas outras duas plantas (2 e 3) para testare tentar entender o possível problema por trás do mapeamento, por se tratar de um ambiente simulado o resultado deveria ser mais próximo possível da planta utilizada.

Figura 10 – Caminho percorrido pelo robô.



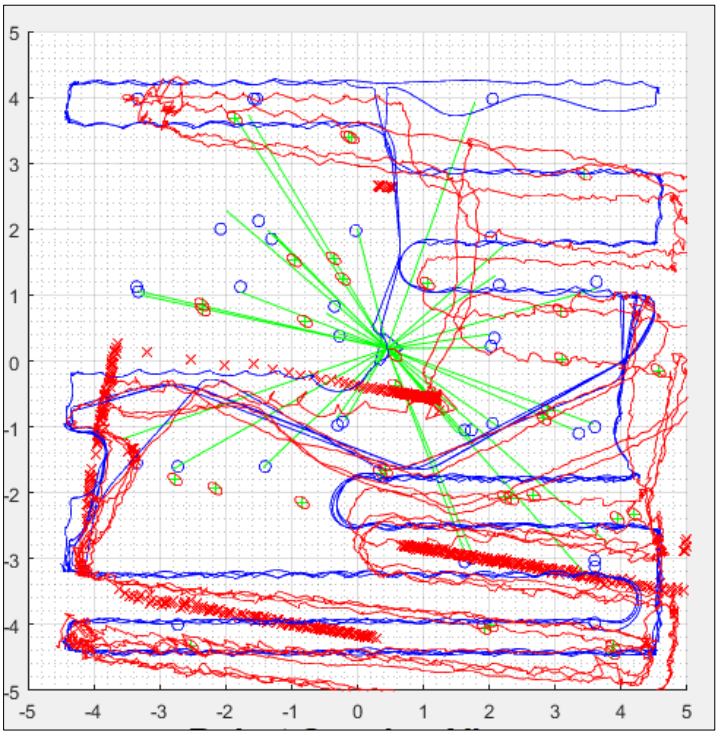
Fonte: Elaboração própria.

Figura 11 – Mapa gerado da planta 1.



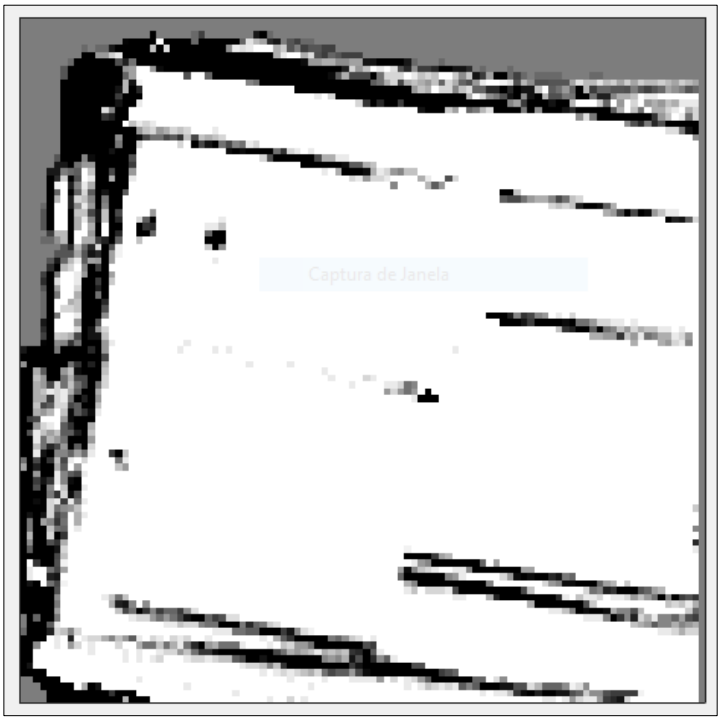
Fonte: Elaboração própria.

Figura 12 – Caminho percorrido pelo robô planta 2.



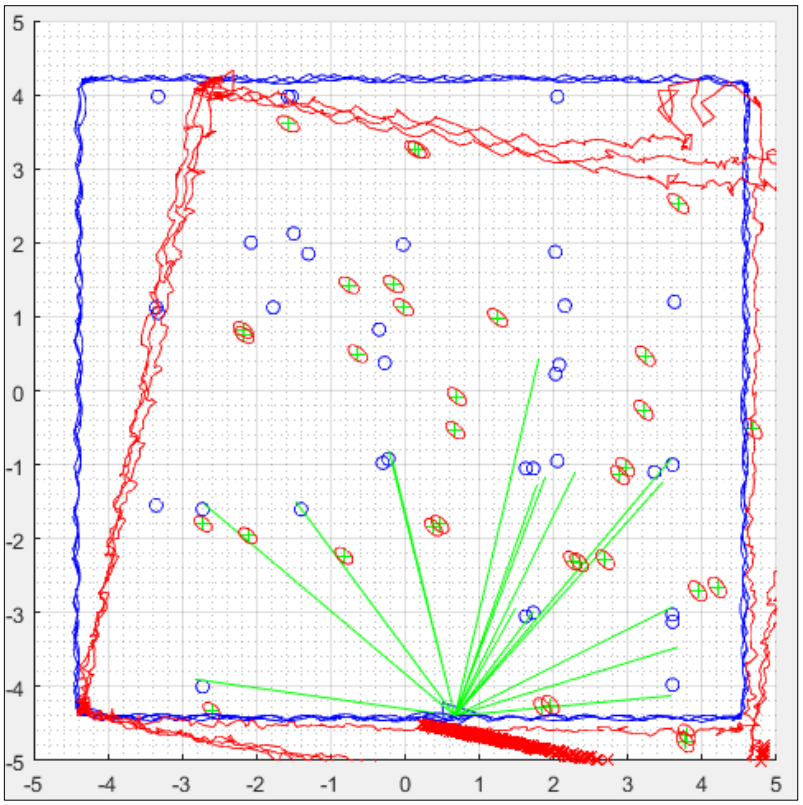
Fonte: Elaboração própria.

Figura 13 – Mapa gerado da planta 2.



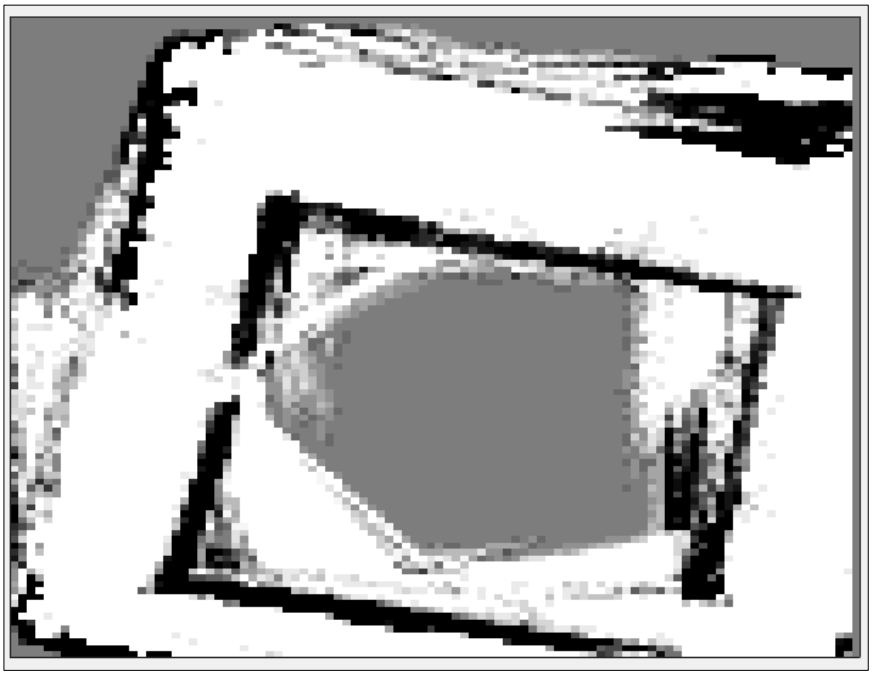
Fonte: Elaboração própria.

Figura 14 – Caminho percorrido pelo robô planta 3.



Fonte: Elaboração própria.

Figura 15 – Mapa gerado da planta 3



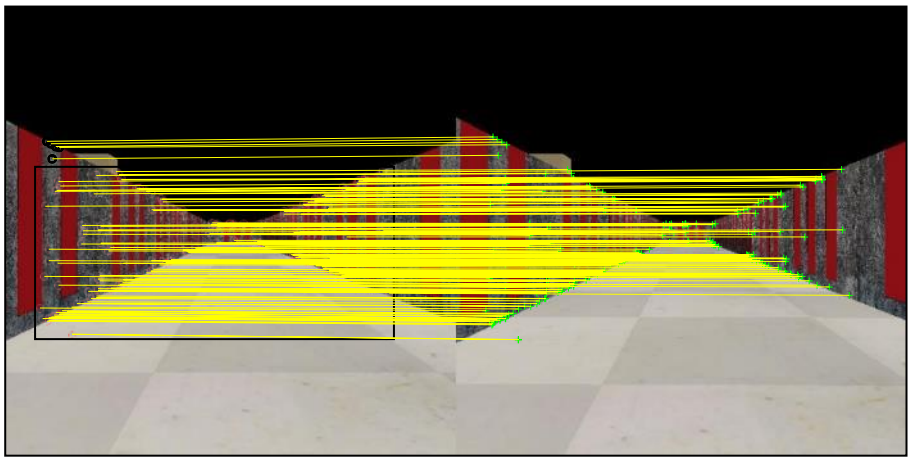
Fonte: Elaboração própria.

Os resultados da Figura 12 até a 15 possuem inconsistências assim como no

primeiro caso, mesmo em plantas mais simples como a da Figura 15 o resultado não foi um mapa que descrevesse bem o local (com inconsistência nos dados), embora tenha construído um mapa que lembra o mapa “real” na indústria isso pode se tornar problema, podendo causar acidentes como colisões, já que as informações parecem não ser totalmente claras. O resultado fornecido nas Figuras 10, 12 e 14 possuem uma linha de duas cores, a vermelha representa o robô em sua posição estimada e a azul a “real” (na simulação), olhando para estes gráficos a posição estimada quase nunca coincide com a posição em ele deveria estar. Esse problema para o SLAM é bem comum e existe diversas propostas para resolver problemas de inconsistências nas posições estimadas, mas serão melhor abordadas para o vSLAM, de modo geral a mistura de técnicas tem tornado o SLAM como um todo mais preciso.

Na Figura 16, a imagem mostra o algoritmo do vSLAM marcando pontos para dar início aos cálculos da posição estimada do robô, esta etapa é fundamental para que o algoritmo encontre a posição inicial para realizar as estimativas de forma correta durante todo percurso percorrido. Para isso o algoritmo as duas primeiras imagens e as compara. As marcações em vermelho e verde, são pontos de interesse e utilizando as duas primeiras imagens, o algoritmo consegue estimar a posição inicial, onde é traçado linhas em amarelo (Figura 16), comparando a primeira e segunda imagem, com os mesmos pontos para estimar o deslocamento inicial. Uma imagem cheia de detalhes é fundamental para que o algoritmo consiga encontrar pontos geométricos em imagem, e marcar regiões como ponto de referência (geométricos) para imagens subsequentes que também vão encontrar os mesmos pontos ou não e ao final relacionar com imagens anteriores e estimar uma nova posição a cada imagem nova que entra para o sistema para ser processada.

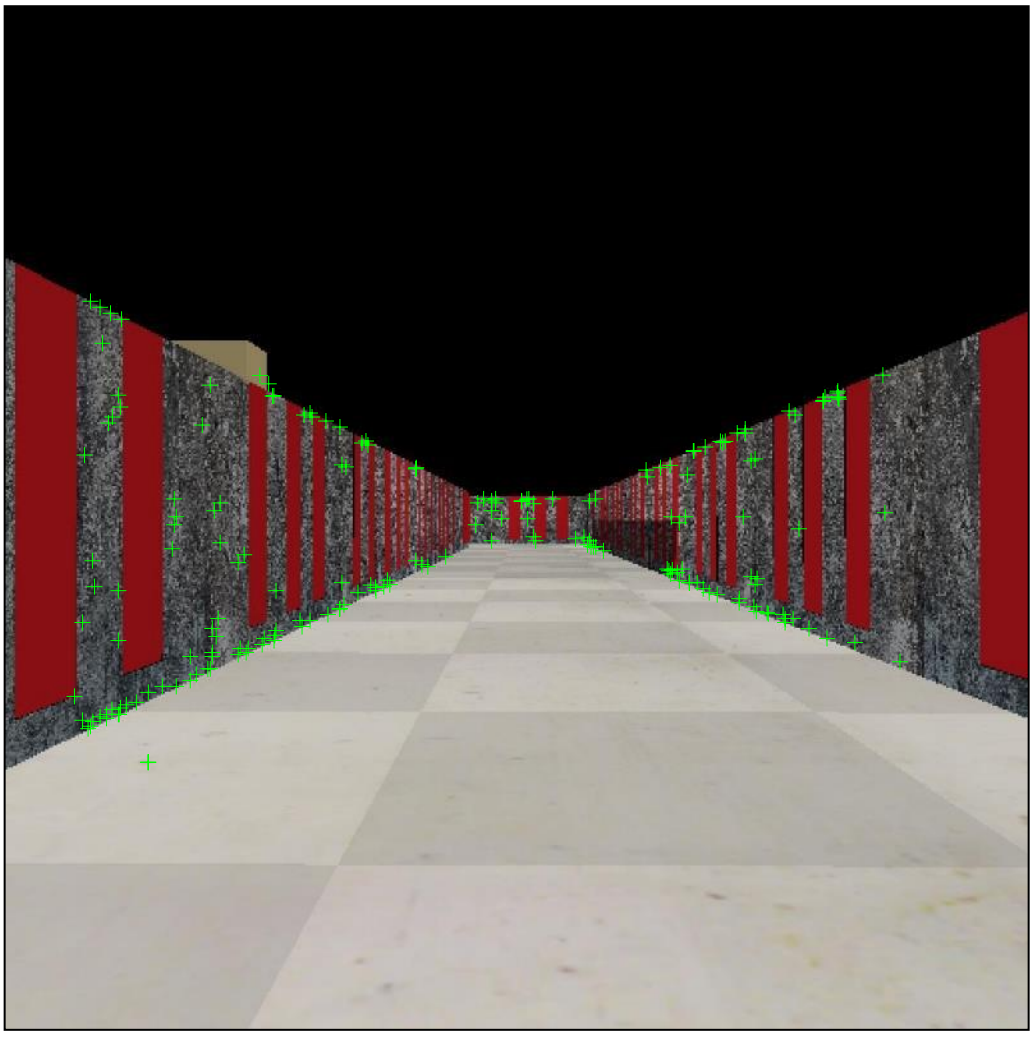
Figura 16 – Definindo estados iniciais – Planta 2.



Fonte: Elaboração própria.

Após definir o ponto inicial que é dado pela comparação das duas primeiras imagens o algoritmo segue recebendo imagens para estimar sua posição à medida em queas imagens são inseridas no sistema, assim é possível estimar o percurso percorrido pelorobô ao longo do processo. Os pontos na terceira imagem que vem logo em seguida mostrando o ponto inicial pode ser visualizada na Figura 17. Quanto mais pontos o algoritmo conseguir identificar, melhor será a estimativa e menor é chance do robô se “perder”, visto que em imagens menos texturizadas o algoritmo para sua execução por falta de referência, por esse motivo, decidiu-se “pintar” as paredes da simulação com faixas vermelhas e de forma desordenada.

Figura 17 – Processamento das imagens.



Fonte: Elaboração própria.

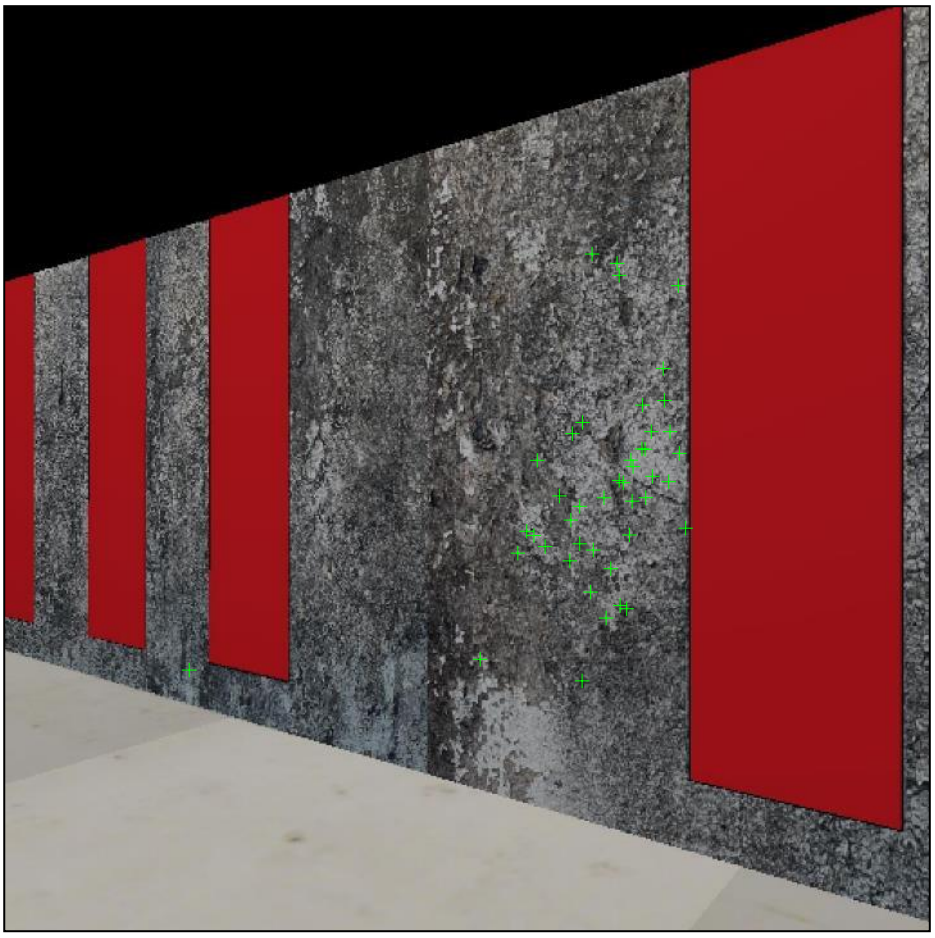
Na Figura 18, logo após iniciar a curva acontece um erro no algoritmo (não há pontos o suficiente para prosseguir então as referências do mapa são perdidas ou não podem ser calculadas) e uma exceção é gerada, parando o processo e o resultado gerado é mostrado na Figura 19, onde 3 linhas são traçadas (trajetória real, estimada e a otimizada) o quarto item é a nuvem de pontos que está relacionada as geometrias de referências encontradas em cada imagem. O vSLAM é uma técnica sensível a diversos fatores, dentre eles a luz, movimentos rápidos do robô ou câmera. Para isto o sistema deve contar com uma ferramenta de relocalização e otimização do mapa:

> A relocalização é necessária quando o rastreamento falha devido ao movimento rápido da câmera ou a alguns distúrbios. Neste caso, é necessário calcular

> novamente a pose da câmera em relação ao mapa. Portanto, esse processo é chamado de “relocalização”. Se a relocalização não for incorporada aos sistemas vSLAM, ossistemas não funcionarão mais depois que o rastreamento for perdido e tais sistemas nãoserão praticamente úteis. Portanto, um método rápido e eficiente para a relocalização tem sido discutido na literatura. Observe que isso também é conhecido como problemasde robô sequestrado em robótica (TAKETOMI, 2017).

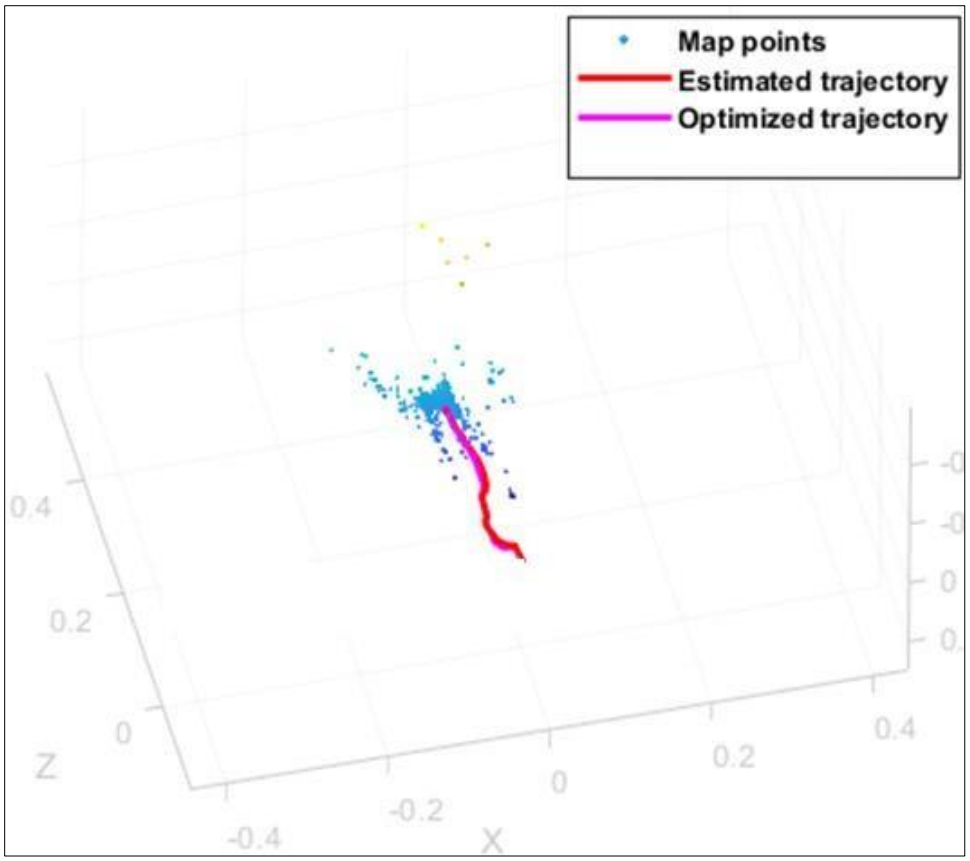
A otimização de mapa é uma tentativa de diminuir o erro acumulativo.

Figura 18 – Local onde acontece o erro.



Fonte: Elaboração própria.

Figura 19 – Mapa 3D da trajetória do robô.

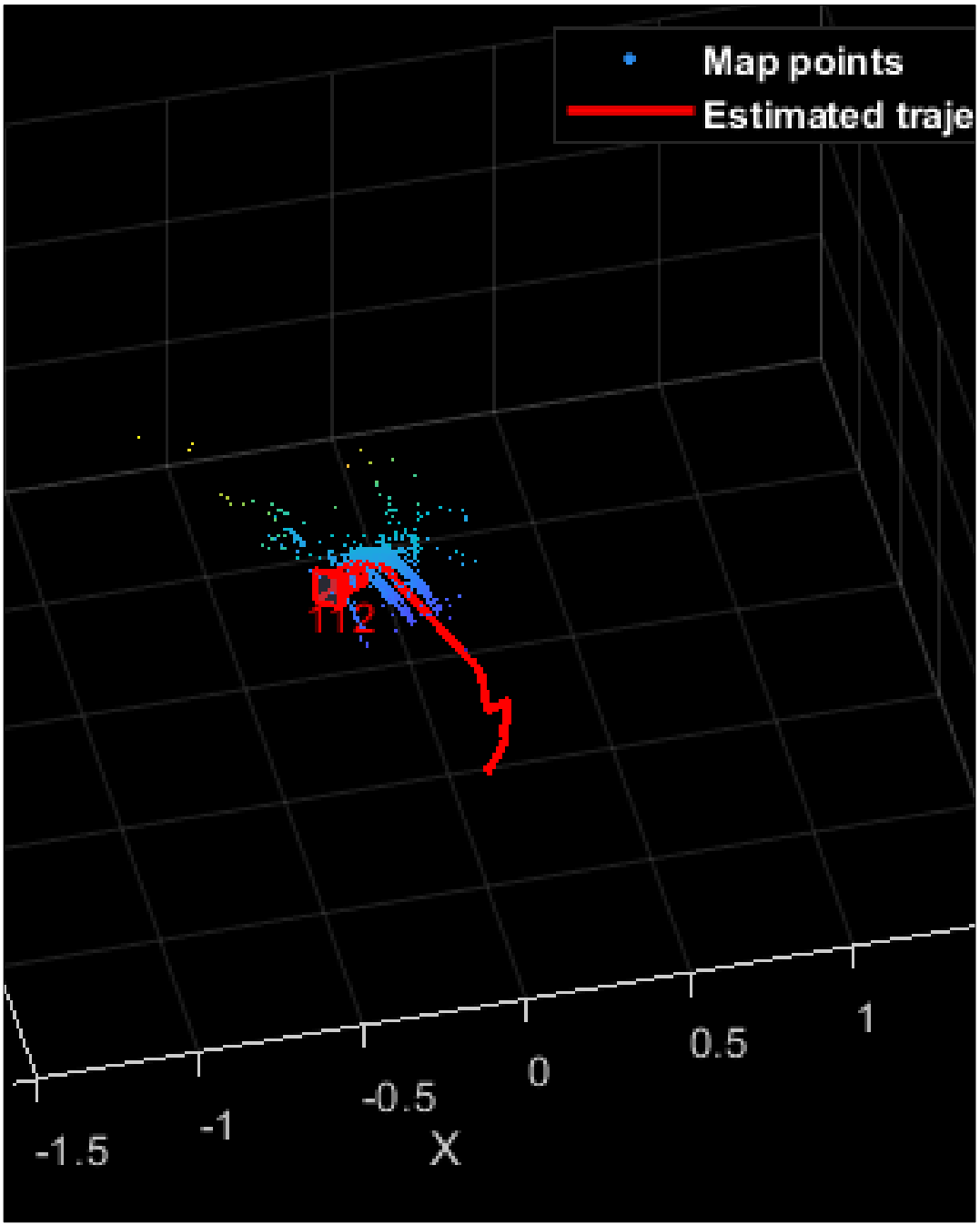


Fonte: Elaboração própria.

O *script* utilizado, não utiliza nenhuma ferramenta de relocalização, então quando acontece movimentos de rotação pura ou as imagens são muito semelhantes o algoritmo acaba estimando sua localização de forma incorreta, resultando na perda de sua localização e ocasionando erros no "script". Para tentar melhorar o desempenho do método, foi testado aumentar a frequência de amostragem. Como o equipamento (computador) utilizado não iria conseguir aumentar o ciclo de varredura (interrupção do sistema à uma taxa maior) do programa (pois o processamento não é suportado) o meio utilizado foi reduzir a velocidade do veículo em 5 vezes (de 2,5 m/s para 0,5 m/s), assim foi possível capturar 5 vezes mais imagens no mesmo espaço percorrido. Os resultados foram melhores (Figura 20) que o anterior (Figura 19), levando o robô a percorrer uma trajetória maior e chegou a concluir o movimento de pura rotação, mas logo em seguida parou, então o método não foi o suficiente, pois em alguns momentos o algoritmo "perdeu" sua localização e o processo parou, o que levou a crer que o maior motivo é o

local não possuir diversidade de informações (pouco rica em detalhes distintos).

Figura 20 – Melhoria.

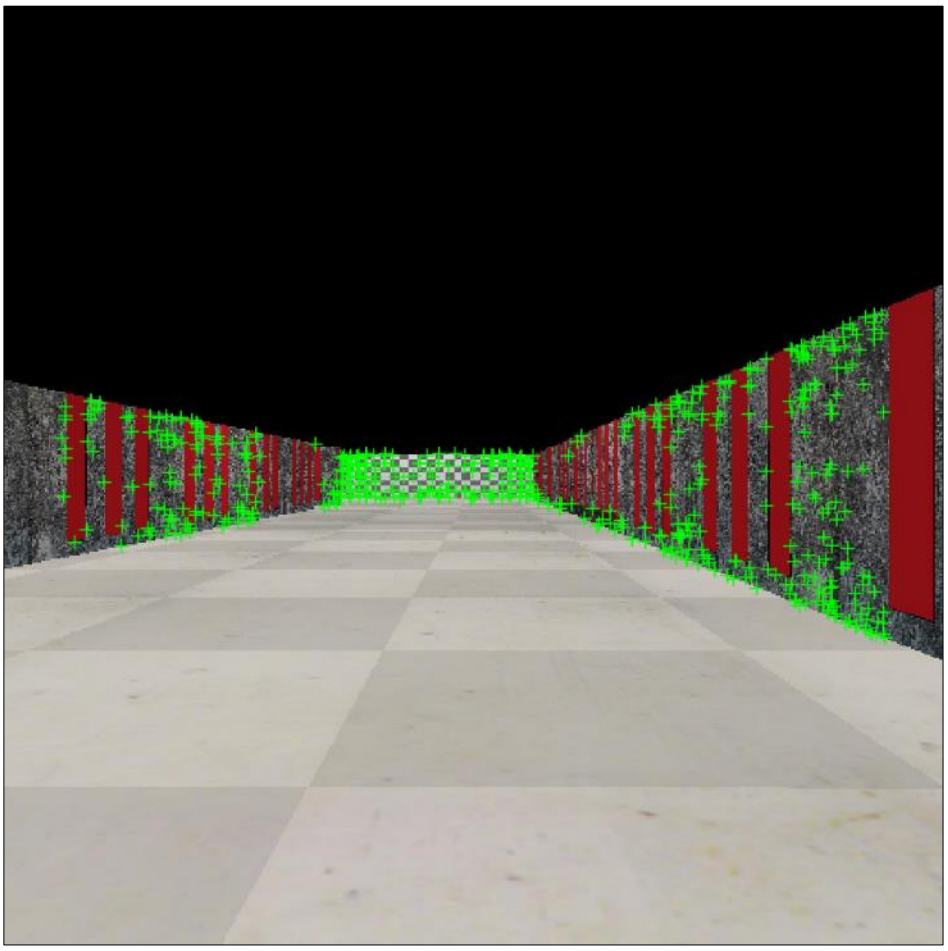


Fonte: Elaboração própria.

Uma última tentativa foi mudar a imagem do ponto inicial da simulação, com a finalidade de diferenciar o ponto final (o “loop” que deve ser realizado), do restante do percurso, já que o ambiente simulado é parecido em cada canto da simulação, também foi

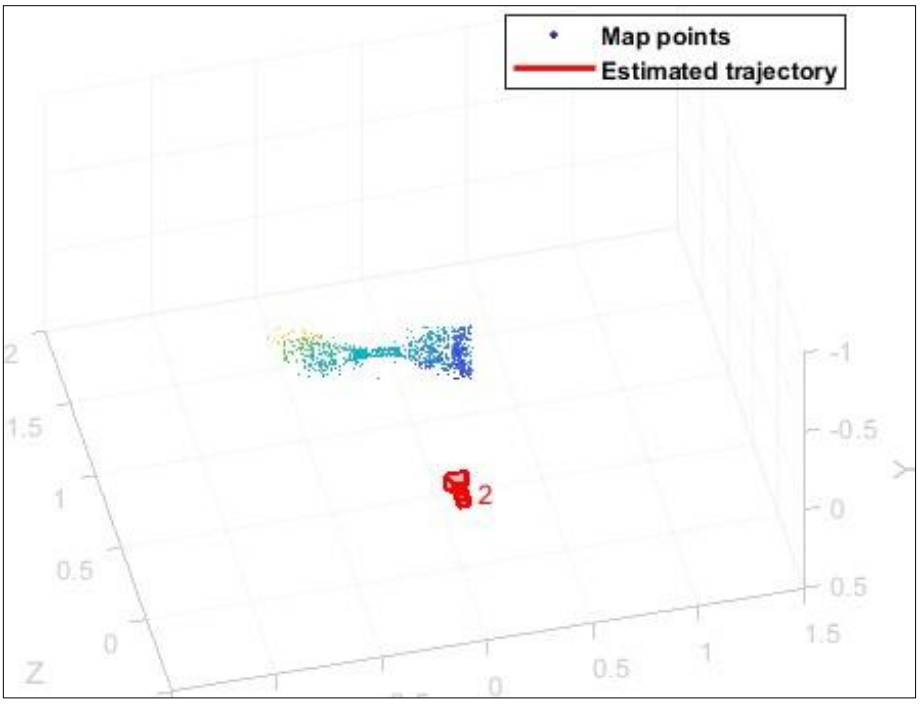
aumentado o número de pontos (aumentado o número de pontos marcados em uma imagem) e o resultado pode ser observado na Figura 21.

Figura 21 – Modificações na planta



Fonte: Elaboração própria.

Figura 22 – Pontos



Fonte: Elaboração própria.

A Figura 21 mostra uma quantidade maior de pontos (Figura 22), isso facilita na construção do mapa, pois os pontos demarcados são utilizados nas imagens subsequentes que entram para o algoritmo, que utilizam os pontos já calculados em execuções passadas N(t-1) onde N é a varredura e t é o tempo atual, assim quanto mais referências forem encontradas, mais fácil para o algoritmo estimar sua localização local em relação ao que está a sua volta, porém isso tem um custo computacional maior, porque o espaço de busca é aumentado ao espalhar e aumentar o número de pontos isso exige mais da máquina (computador). O mesmo teste foi realizado com as demais plantas e o resultado foi o mesmo. A máquina utilizada pelo autor não permite uma taxa maior de aquisição de imagens, então os testes foram finalizados por aqui, sem completar a trajetória.

Conclusão

Os resultados para o SLAM Lidar foram promissores, mesmo simples, foi possível ter uma noção do poder que a ferramenta tem, já o vSLAM possui algumas complicações que podem impedir sua implementação dentro da indústria ou até mesmo a necessidade de adaptações o que pode inviabilizar o método já que possuem outros meios que podemser menos susceptíveis a falhas devido a variáveis externas, porém em ambos os

casos foram aplicados utilizando a forma mais básica do SLAM, sendo que o campo de estudoSLAM é bem mais vasto, alguns artigos descrevem outras técnicas que foram desenvolvidas ao passar dos anos que podem melhorar a aplicação, alguns são mencionados nos artigos: *A Comprehensive Survey of Visual SLAM Algorithms. Robotics* (MACARIO BARROS, 2022) e *Visual SLAM algorithms: A survey from 2010 to 2016. IPSJ Transactions on Computer Vision and Applicatio*ns (TAKETOMI, 2017). Existe outro artigo que aborda um SLAM utilizando a técnica *Swarm* (enxame) que consiste em descentralizar o SLAM e realizar uma "exploração" de forma coletiva (KEGELEIRS, 2021).

Outro artigo que também busca melhoria no SLAM é o *Kimera-Multi: Robust, Distributed, Dense* (TIAN, 2022), diferente de outras técnicas a inicialização utiliza um mapa global para iniciar as variáveis e a estimativa da trajetória é distribuída, trabalhando com paralelismo. O resultado demonstrado para esta técnica parece promissor pelos resultados do artigo. Com isso, pode-se perceber que o SLAM é uma ferramenta que promete mudar o modo como vemos os meios de navegação autônomos e com este trabalho, também pode-se concluir que aplicar somente o SLAM não é factível, em ambientes mais sofisticados (com grandes perturbações), já que em ambientes simples o desempenho não foi robusto, mas serve como aprendizado e dar uma visão básica do que é o SLAM, veículos autônomos e uma perspectiva diferente do habitual onde nós temos um dispositivo fornecendo informações da localização global o tempo todo. Se tornando assim um desafio a ser vencido, pois o SLAM pode melhorar também outros sistemas, como a navegação autônoma de robôs de exploração no mar e de exploração espacial.

Referências

BARBOSA, Laercio. RMAproject: SLAM with Matlab and V-REP. In: BARBOSA, Laercio. **RMAproject: SLAM with Matlab and V-REP**. 1. 1. ed. Github: Instituto de Computação e Matemática Computacional, Universidade de São Paulo, USP, 2017. Disponível em: https://github.com/laercio-barbosa/rmaProject. Acesso em: 26 jun.2021.

CHONG, T. J. et al. Sensor technologies and simultaneous localization and mapping (SLAM). **Procedia Computer Science**, v. 76, p. 174-179, 2015.

ESPINOSA, Felipe et al. Electronics proposal for telerobotics operation of P3-DX units. In: **Remote and Telerobotics**. IntechOpen, 2010.

FILIPENKO, Maksim; AFANASYEV, Ilya. Comparison of various slam systems for mobile robot in an indoor environment. **International Conference on Intelligent Systems (IS)**. IEEE, 2018. p. 400-407.

GADIOLI, Alefe Vitor Almeida; SILVA, Rafael Leal. Odometria: comportamento em trajetória retilínea e curvilínea e a utilização regressões comoforma de redução de erros. **Mostra Nacional de Robótica (MNR)**, 2017. Disponível em: http://sistemaolimpo.org/midias/uploads/048f8060e90df2d30a7540a8b88448dd.pdf. Acesso em: 25 set. 2023.

KARLSRUHE INSTITUTE OF TECHNOLOGY. **The KITTI Vision Benchmark Suite.** Cvlibs, 2021. Disponível em: http://www.cvlibs.net/datasets/kitti/eval_odometry.php. Acesso em: 30 mai. 2021.

KEGELEIRS, Miquel; GRISETTI, Giorgio; BIRATTARI, Mauro. Swarm SLAM: Challenges and perspectives. **Frontiers in Robotics and AI**, v. 8, p. 23, 2021.

MACARIO BARROS, Andréa et al. A Comprehensive Survey of Visual SLAM Algorithms. **Robotics**, 2022. v. 11, n. 1, p. 24, 2022.

MathWorks. Help Center: **Monocular Visual Simultaneous Localization and Mapping.** cvlibs. 2021. 1 p. Disponível em: https://www.mathworks.com/help/vision/ug/monocular-visual-simultaneous-localization-and-mapping.html. Acesso em: 28 fev. 2022.

MATLAB. [S. l.]: **The Mathworks**, Inc., 2024. Disponível em: https://www.mathworks.com/. Acesso em: 10 mar. 2021.

RUSSELL, Stuart; NORVIG, Peter; **Inteligencia artificial**. Tradução Regina Célia Simille de Macedo. 3a. ed. Rio de Janeiro: Elsevier Editora, 2013.

REZENDE, Solange Oliveira. **Sistemas inteligentes:** fundamentos e aplicações. Barueri: Manole, 2003.

ROHMER, E. et al. CoppeliaSim (formerly V-REP): a Versatile and Scalable Robot Simulation Framework. **CoppeliaSim,** v.4.10. Proc. of The International Conference on Intelligent Robots and Systems (IROS), 2013. Disponível em: www.coppeliarobotics.com. Acesso em: 14 abr. 2021.

SCHWAB, KLAUS. **A quarta revoluçao industrial**. Tradução Daniel MoreiraMiranda. São Paulo: Edipro, 2016.

SMITH, Randall C.; CHEESEMAN, Peter. On the representation and estimation of spatial uncertainty. **The international journal of Robotics Research**, v. 5, n. 4, p. 56-68, 1986.

STACHNISS, Cyrill. **Robotic mapping and exploration**. Springer, 2009.

TAKETOMI, Takafumi; UCHIYAMA, Hideaki; IKEDA, Sei. Visual SLAM algorithms: A survey from 2010 to 2016. **IPSJ Transactions on Computer Vision and Applications**, v. 9, n. 1, p. 1-11, 2017.

TIAN, Yulun et al. Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems**. IEEE Transactions on Robotics**, 2022.

YANG, Tao et al. Monocular vision SLAM-based UAV autonomous landing in emergencies and unknown environments**. Electronics**, v. 7, n. 5, p. 73, 2018.

YOUSIF, Khalid; BAB-HADIASHAR, Alireza; HOSEINNEZHAD, Reza. An overview to visual odometry and visual SLAM: Applications to mobile robotics. **Intelligent Industrial Systems**, v. 1, n. 4, p. 289-311, 2015.

ZHANG, Zhengyou. Determining the epipolar geometry and its uncertainty: Areview. **International journal of computer vision**, v. 27, n. 2, p. 161-195, 1998.