UNIVERSIDADE ESTADUAL DE PONTA GROSSA
MESTRADO EM COMPUTAÇÃO APLICADA

SULLIVAN LOURENÇO DA SILVA

FUSÃO DE SENSORES NA AUTOLOCALIZAÇÃO DE AGROBOTS EM AMBIENTES INTERNOS: USO DE INFORMAÇÕES EXTRAÍDAS PELO ALGORITMO YOLO COM DADOS DE RSSI

PONTA GROSSA
2024

SULLIVAN LOURENÇO DA SILVA

FUSÃO DE SENSORES NA AUTOLOCALIZAÇÃO DE AGROBOTS EM AMBIENTES INTERNOS: USO DE INFORMAÇÕES EXTRAÍDAS PELO ALGORITMO YOLO COM DADOS DE RSSI

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação Aplicada, curso de Mestrado em Computação Aplicada da Universidade Estadual de Ponta Grossa, como requisito para obtenção do título de Mestre

Orientador: Prof. Dr. José Carlos Ferreira da Rocha

PONTA GRO SSA

2024

Silva, Sullivan Lourenço da
S586 Fusão de sensores na autolocalização de agrobots em ambientes internos: uso de informações extraídas pelo algoritmo YOLO com dados de RSSI / Sullivan Lourenço da Silva. Ponta Grossa, 2024.
73 f.

Dissertação (Mestrado em Computação Aplicada - Área de Concentração: Computação para Tecnologias em Agricultura), Universidade Estadual de Ponta Grossa.

Orientador: Prof. Dr. José Carlos Ferreira da Rocha.

1. Yolov8. 2. Casa de vegetação. 3. Autolocalização. 4. Fusão de informação. 5. Visão computacional. I. Rocha, José Carlos Ferreira da. II. Universidade Estadual de Ponta Grossa. Computação para Tecnologias em Agricultura. III.T.

CDD: 004

Ficha catalográfica elaborada por Maria Luzia Fernandes Bertholino dos Santos- CRB9/986

UNIVERSIDADE ESTADUAL DE PONTA GROSSA
Av. General Carlos Cavalcanti, 4748 - Bairro Uvaranas - CEP 84030-900 - Ponta Grossa - PR - https://uepg.br

TERMO

TERMO DE APROVAÇÃO

Sullivan Lourenço da Silva

"FUSÃO DE SENSORES NA AUTOLOCALIZAÇÃO DE AGROBOTS EM AMBIENTES INTERNOS: USO COMBINADO DO ALGORITMO YOLO E DADOS DE RSSI"

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre no Programa de Pós-Graduação em Computação Aplicada da Universidade Estadual de Ponta Grossa, pela seguinte banca examinadora:

Prof. Dr. José Carlos Ferreira da Rocha (UEPG/PR - Presidente)

Prof. Dr. Alceu de Souza Britto Júnior (UEPG/PR)

Prof. Dr. Marco Antônio Simões Teixeira (PUC/PR)

Ponta Grossa, 05 de novembro de 2024.

Documento assinado eletronicamente por **Marco Antonio Simões Teixeira, Usuário Externo**, em 05/11/2024, às 10:56, conforme Resolução UEPG CA 114/2018 e art. 1°, III, "b", da Lei 11.419/2006.

Documento assinado eletronicamente por **Alceu de Souza Britto Junior, Professor(a)**, em 05/11/2024, às 10:58, conforme Resolução UEPG CA 114/2018 e art. 1°, III, "b", da Lei 11.419/2006.

Documento assinado eletronicamente por **Jose Carlos Ferreira da Rocha, Coordenador(a) do Programa de Pós-Graduação em Computação Aplicada - Mestrado**, em 05/11/2024, às 11:00, conforme Resolução UEPG CA 114/2018 e art. 1°, III, "b", da Lei 11.419/2006.

A autenticidade do documento pode ser conferida no site https://sei.uepg.br/autenticidade informando o código verificador **2249377** e o código CRC **1369D9E3**.

Dedico a toda minha família, em especial a minha esposa e filho,

Emanuelle e Jonathan.

AGRADECIMENTOS

Agradeço a Deus por me sustentar, conduzir, e conceder sabedoria para realizar esta pesquisa, além de colocar pessoas que me guiaram e apoiaram ao longo dessa jornada.

Agradeço à minha esposa, Emanuelle, pelo incentivo, compreensão e por estar ao meu lado em todas as etapas da minha vida, sempre me dando força para seguir em frente. Ao meu filho, Jonathan, por me ajudar a equilibrar os estudos com a rotina cotidiana.

Agradeço aos meus pais, Aristides e Idazil, por terem me dado a base para a vida e me incentivado durante toda minha trajetória acadêmica.

Agradeço ao Prof. Dr. José Carlos, por me tranquilizar nos momentos difíceis, mantendo-me na direção correta, e por suas orientações e compartilhamento de conhecimento em todas as etapas deste trabalho.

Agradeço também a todos os professores e amigos que, direta ou indiretamente, contribuíram para a conclusão desta pesquisa.

RESUMO

Este trabalho avalia o efeito do método de fusão de informações (sensores) no desempenho de procedimentos de autolocalização de *agrobots* móveis em casas de vegetação. Para isso, compara os erros de estimativa das coordenadas e o tempo de processamento de dois procedimentos de autolocalização. O primeiro procedimento, CVAutoL, explora informações extraídas de imagens via técnicas de visão computacional. O segundo, FusionAutoL, realiza a fusão de informações de dados extraídos de imagens com dados de RSSI. A detecção do objeto de referência para estimativa da posição em imagens foi efetuada com o algoritmo YOLOv8. Ambos os processos aplicaram uma rotina de regressão não linear, baseada em máquina de vetores de suporte, para ajustar uma função que relaciona os atributos com as coordenadas do *agrobot*. Os erros da distância entre as coordenadas estimadas e as coordenadas reais do *agrobot* (E_2) e a diferença entre as distâncias estimada e real do *agrobot* ao marcador (E_O) foram comparados. Três versões do YOLOv8-Seg foram utilizadas durante os testes, *nano, medium e extra-large*, com o objetivo de verificar se a fusão de dados possibilita o emprego de redes rasas sem que haja perdas de precisão na autolocalização. Os resultados obtidos indicam que o procedimento que efetuam a fusão de informação tiveram erros menores, com E_2 e E_O. Também sugerem que a fusão de informações viabilizou o uso de versões rasas, e mais rápidas, do YOLO sem aumento do erro nas estimativas de posição.

Palavras-chave: Autolocalização. Agrobot. Fusão de informação. YOLOv8. Casa de Vegetação. Visão Computacional. Máquinas de vetor de suporte.

ABSTRACT

This work evaluates the effect of the data fusion method (sensors) on the performance of self-localization procedures for mobile agrobots inside greenhouses. To do so, it compares the estimation errors of coordinates and the processing time of two self-localization procedures. The first procedure, CVAutoL, utilizes information extracted from images via computer vision techniques. The second, FusionAutoL, fuses data extracted from images with RSSI data. The detection of the reference object for position estimation in the images was performed using the YOLOv8 algorithm. Both procedures employed a nonlinear regression routine, based on a support vector machine, to fit a function that relates the attributes to the agrobot's coordinates. The errors in the distance between the estimated coordinates and the actual agrobot coordinates (E_2) and the difference between the estimated and actual distance from the agrobot to the marker (E_O) were compared. Three versions of YOLOv8-Seg were used during the tests, nano, medium, and extra-large to verify whether data fusion allows the use of shallow networks without compromising the precision of self-localization. The results obtained indicate that the data fusion procedure resulted in lower errors, with E_2 and E_O. They also suggest that the data fusion enabled the use of shallow and faster versions of YOLO without increasing the error in position estimates.

Keywords: Self-localization. Agrobot. Data Fusion. YOLOv8. Greenhouse. Computer Vision. Support Vector Machine

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

As casas de vegetação têm um papel importante na agricultura. Elas oferecem um ambiente controlado que favorece o cultivo de determinadas plantas ao reduzir o efeito do ambiente interno sobre o seu desenvolvimento. Segundo Farooq *et al*. (2022), uma casa de vegetação é uma estrutura semelhante a uma casa que é coberta com plástico ou vidros e majoritariamente desenhada para cultivar múltiplas culturas em qualquer estação. Além disso, como em diversas outras atividades econômicas, o plantio em casas de vegetação vem passando por processo de automação com o objetivo de aumentar a produtividade e diminuir custos.

Nesse contexto, Bagagiolo *et al*. (2022) destacam o crescimento de pesquisas sobre o uso de *agrobots* (robôs agrícolas) em casas de vegetação. Em particular, ele destaca o interesse no uso de *agrobots* autônomos móveis em decorrência da possibilidade desses executarem tarefas em vários pontos de uma casa de vegetação. Para ser considerado móvel e autônomo, segundo Tzafestas (2013), o robô deve ser capaz de se deslocar de um lugar para outro sem a interferência de um operador humano.

O planejamento da locomoção e a tomada de decisão quanto à execução de uma ação depende da capacidade do *agrobot* determinar sua localização no ambiente de operação. Isto é, da capacidade do mesmo utilizar sensores e/ou outros meios computacionais para estimar suas coordenadas no ambiente. Esta tarefa é denominada de autolocalização. Segundo Thrun, Burgard e Fox (2005), utilizando-se de um procedimento de detecção de um objeto de referência em um ambiente e de inteligência artificial é factível estimar a posição do robô em relação a esse objeto.

A autolocalização de robôs em uma casa de vegetação com um único pavimento plano objetiva a estimativa das coordenadas (x, y) que especificam a posição do robô em um mapa bidimensional (Berz; Tesch; Hessel, 2015). Em alguns casos, como para evitar colisões, também é útil calcular a distância do robô até um objeto de referência (Azurmendi *et al*., 2023). Este problema pode ser visto como uma tarefa de regressão que explora uma relação funcional entre (x, y) e um conjunto de informações do ambiente extraídas via sensores. Para Jondhale *et al*. (2022), uma máquina de vetor de suporte (SVM) é uma ferramenta de aprendizado de máquina que apresenta capacidade de generalização e tem se destacado em problemas de regressão. Wu *et al*. (2007), destacam a possibilidade de se empregar máquinas de vetor de suporte em tarefas de regressão (Regressão de Vetor de Suporte - SVR).

Segundo Foroughi *et al*. (2021), uma forma de abordar a autolocalização de robôs em ambientes internos é com o emprego de técnicas derivadas da visão computacional. Nessa abordagem, a ideia é usar o processamento de imagens e técnicas de reconhecimento de padrões para extrair informações das imagens capturadas por uma câmera e utilizá-las para estimar as coordenadas relativas do robô, dado um ponto de referência. Uma alternativa à visão computacional é a utilização das informações recebidas pelo robô por antenas emissoras de sinal WiFi, mais especificamente o RSSI (Indicador de intensidade do sinal recebido). A localização baseada em WiFi, parte do pressuposto que a intensidade do sinal depende da posição do robô no ambiente (Husen; Lee, 2014). Portanto é possível estimar sua distância às antenas que emitiram o sinal e, uma vez que a posição daquelas seja conhecida, sua localização.

Ambas as abordagens possuem suas limitações. No caso da visão computacional, uma delas é que a distância do *agrobot* ao objeto de referência durante a aquisição das imagens pode afetar o erro na estimativa das distâncias (Hao et al., 2022). Além disso, o processo de extração de informações tem impacto nas estimativas da posição e no tempo de processamento. Estes elementos devem ser levados em conta quando se considera a operação do robô em tempo real. O sinal do WiFi pode sofrer alterações devido a obstruções, umidade, temperatura e reflexões em superfícies. Estes eventos diminuem a qualidade do sinal.

Para Tzafestas (2013), a fusão de dados de diversos sensores objetiva reduzir as incertezas que podem afetar a estimativa da posição de um robô. Nesta técnica, as informações obtidas de diferentes sensores são utilizadas em conjunto. Adicionalmente, é necessário considerar aspectos práticos quanto a escolha das tecnologias combinadas. Neste sentido, Lukasik, Szott e Leszczuk (2024) observam que a ampla disponibilidade e o custo baixo de dispositivos com WiFi, fazem com que essa tecnologia seja atrativa para sistemas de autolocalização de *agrobots*. O Governo do Estado do Paraná, lançou em 2024 o programa Conectividade Rural do Paraná (2024), com o objetivo de tornar o sinal de internet ainda mais acessível para produtores rurais. De maneira análoga, o custo de câmeras digitais também tem apresentado uma redução significativa ao longo do tempo (Azurmendi *et al.*, 2023).

Considerando o exposto, esse trabalho avaliou o desempenho da fusão de sensores na autolocalização de *agrobots* em casas de vegetação. Para tanto, comparou o desempenho de um procedimento de autolocalização baseado somente em visão computacional, denominado CVAutoL, com outro, FusionAutoL, que combina dados extraídos de imagens e dados de RSSI. O procedimento CVAutoL emprega o algoritmo YOLO (*You only look once*) para detectar um

objeto no ambiente de teste. Os dados extraídos pelo YOLO são tratados para aquisição de informações que então são inseridas em um procedimento de regressão com múltiplas variáveis dependentes baseados em SVR que ajustou uma função para estimar a coordenada (x_e, y_e). A partir destes valores também foi estimada a distância do *agrobot* ao objeto de referência. O procedimento FusionAutoL adiciona leituras de RSSI às entradas da SVR.

A principal motivação para o emprego do algoritmo YOLO é que ele dispõe de funcionalidades que viabilizam o desenvolvimento de procedimentos de autolocalização que, segundo Thrun, Burgard e Fox (2005), detectam os pontos de referência (marcadores) no ambiente e então usam métodos estatísticos ou de inteligência artificial para estimar a posição do robô em relação aos marcadores. Além disso, a eficiência da autolocalização, em termos de tempo de processamento, é fundamental para implementação de robôs móveis e, de acordo com Wang et al. (2022), o YOLO pode ser executado em tempo real.

A demanda por processamento em tempo real tem motivado o emprego de algoritmos de detecção de objetos que exploram implementações rasas de redes neurais na execução de suas tarefas (Khan *et al*., 2024; Susanto *et al*., 2023; Dwijayanti et al., 2024; Pitts, 2024). Nesse contexto, este trabalho avaliou três diferentes versões do YOLOv8, n (*nano*) (mais rápida), m (*medium*) e x (*extra-large*) (mais lenta). O objetivo foi verificar se a fusão de informações permite incrementar o desempenho em termos de velocidade de processamento e manter a precisão do procedimento de autolocalização.

Os procedimentos CVAutoL e FusionAutoL foram testados em uma casa de vegetação. Os testes realizados tiveram o objetivo de avaliar a eficácia preditiva e o tempo de processamento de cada procedimento para cada uma das configurações do YOLO. A regressão fez uso de uma SVR, com função de base radial, para etapa de estimação das coordenadas do *agrobot*. Os resultados obtidos foram submetidos a análise estatística com o objetivo de determinar a existência de diferenças significativas nos desempenhos dos procedimentos de autolocalização baseado em visão computacional e fusão de sensores. Adicionalmente, também são avaliados o efeito do emprego das diferentes versões da YOLO sobre o desempenho dos procedimentos em termos de tempo de processamento.

Os resultados mostraram que as configurações de teste que empregaram fusão de informação obtiveram valores menores para as médias dos erros absolutos nas estimativas das ordenadas absolutas da posição do *agrobot*. O mesmo comportamento foi observado quando se quantificou o erro como distância entre as coordenadas estimadas e as coordenadas reais do

agrobot (E_2) ou como a diferença entre as distâncias estimada e real do robô ao marcador (E_O). As diferenças observadas foram significativas segundo o teste ANOVA.

No que se refere ao tempo de processamento, o ganho de desempenho obtido pelas versões rasas do algoritmo de detecção de objetos, YOLO, é suportado por evidência estatística. A análise multiobjetivo, sugere que a fusão de sensores foi eficaz na construção de procedimento de autolocalização que produziram soluções que dominaram as demais abordagens em termos de tempo de processamento e correção nas estimativas de posição.

1.1 OBJETIVOS

1.1.1 Geral

a) Avaliar a eficácia da fusão de informações extraídas por visão computacional, a partir da detecção de objetos da cena com o algoritmo YOLO, e de dados de intensidade de sinal WiFi na autolocalização de *agrobots* em casas de vegetação.

1.1.2 Específicos

a) Verificar a existência de ganho de desempenho preditivo na autolocalização quando do emprego da fusão das informações.
b) Comparar o desempenho do procedimento de autolocalização, em termos de tempo de processamento quando do uso de diferentes implementações do algoritmo YOLO na detecção dos objetos da cena.

Este estudo propõe uma solução para autolocalização de agrobots em ambientes internos, combinando dados de visão computacional, por meio do YOLOv8-Seg, com sinais RSSI (WiFi) para aumentar a precisão da localização. Foram desenvolvidos e comparados dois procedimentos: CVAutoL, baseado apenas em visão computacional, e FusionAutoL, que integra visão com RSSI, demonstrando que a fusão de dados reduz significativamente os erros de localização. A análise incluiu diferentes versões do YOLOv8-Seg, explorando o uso de modelos mais rasos para aplicações em tempo real. Além disso, o estudo contextualiza seus

resultados com a literatura e sugere aprimoramentos futuros, como o uso de múltiplas câmeras e implementações customizadas, com potencial aplicação em sistemas de mapeamento e localização simultânea (SLAM) em agricultura de precisão.

Este trabalho está organizado em seções que detalham o desenvolvimento e avaliação dos modelos propostos. A seção 2, Revisão da Literatura, introduz os conceitos de autolocalização e discute tecnologias de visão computacional e WiFi. Em Metodologia (seção 3), são descritos os processos de coleta de dados, o treinamento das redes YOLO e os procedimentos CVAutoL e FusionAutoL. A seção 4, Experimentos, apresenta os testes realizados e as métricas de desempenho. Em seguida, a seção 5, Resultados e Discussão, traz uma análise dos experimentos. Finalmente, a seção 6 encerra o trabalho com Considerações Finais, que sumarizam os resultados e sugerem futuras linhas de pesquisa.

2 REVISÃO DA LITERATURA

A autolocalização de robôs por fusão de informação pode ser abstraída em três etapas (Morar et al., 2020; Li et al., 2004; Thrun; Burgard; Fox, 2005). Neste trabalho, a primeira etapa, a aquisição das informações, considera o emprego de câmeras RGB para a captura de imagens e ativos de rede emissores de sinal WiFi para a coleta de dados RSSI. A segunda etapa, extração de informações, aplicou o modelo YOLOv8 e técnicas *ad hoc* de processamento de imagens produzir um conjunto de atributos descritores (características) da cena e do objeto marcador. A terceira etapa, usou um procedimento de regressão por vetor de suporte para estimar a localização do agrobot no ambiente.

Este capítulo apresenta os principais elementos empregados no desenvolvimento deste trabalho. A seção 2.1 apresenta conceitos sobre autolocalização de robôs móveis, a aplicação de visão computacional como meio para autolocalização e os passos necessários para esse processo. Na seção 2.2, aborda-se o uso do procedimento YOLOv8-Seg para realizar a detecção, classificação e segmentação de um objeto marcador na cena, além dos principais avanços tecnológicos introduzidos nessa versão. A seção 2.3 explora o uso de dados RSSI como uma alternativa para realizar a autolocalização. A seção 2.4 descreve o procedimento de regressão por vetor de suporte. Finalmente, a seção 2.5 apresenta alguns trabalhos relacionados ao tema de autolocalização que utilizam técnicas de visão computacional e fusão de dados.

2.1 AUTOLOCALIZAÇÃO DE ROBÔS

Segundo Bekey (2017), um robô é um dispositivo que interage com o ambiente, através de sensores, processa os dados obtidos e toma uma decisão quanto a ação a ser realizada por meio de seus atuadores. Russell e Norving (2013), classificam os robôs em três categorias. Os robôs manipuladores, que são fixos no local de atuação, os móveis, que têm a capacidade de se deslocar pelo ambiente, e os manipuladores móveis, que combinam a mobilidade com a manipulação.

Os robôs móveis autônomos são os que se locomovem no ambiente de operação sem supervisão de um humano (Fragapane *et al.,* 2021; Rubio; Valero; Llopis-Albert, 2019). A fim de desempenhar suas atividades, os mesmos precisam estimar sua posição (autolocalização), determinar a localização dos objetos de interesse e planejar o caminho a ser percorrido. Um

robô pode operar em um ambiente conhecido ou não. No primeiro caso, ele dispõe de um mapa de seu local de operação, no segundo, geralmente, precisa elaborar o mapa. Esse trabalho considera *agrobots* que atuam em ambientes conhecidos.

Thrun, Burgard e Fox (2005), definem o problema de autolocalização de um robô como a tarefa de estimar suas coordenadas e orientação no dado ambiente a partir dos dados coletados pelos sensores. Entre as tecnologias usadas na coleta de dados em ambientes internos destacam-se: Visão computacional (Acharya; Khoshelham; Winter, 2019; Xie *et al*., 2022), WiFi (Bi *et al*., 2023; Ibwe *et al*., 2023), BLE[1] (Junoh; Subedi; Pyun, 2023), Luz Visível (Su *et al*., 2023), Magnetômetro (Kim *et al*., 2023) e LiDAR[2] (Luo *et al.,* 2023). Uma das tecnologias consideradas neste trabalho é aquela que explora a visão computacional. Nesta abordagem, a autolocalização é realizada utilizando dados extraídos de imagens adquiridas com uma câmera digital. Com a imagem capturada é possível identificar objetos ou cenários de interesse e estimar a posição onde a imagem foi obtida (Meng *et al*., 2020).

Uma forma de representar a posição de *agrobot* em um ambiente conhecido é por meio de coordenadas espaciais (x_r, y_r). O emprego desta abstração permite elaborar a autolocalização como um problema de regressão em que a tarefa é estimar a hipótese de localização mais provável, dadas as informações coletadas pelos sensores (Berz; Tesch; Hessel, 2015).

A autolocalização usando métodos de visão computacional pode ser tratada como estimar a posição do robô em relação a determinados objetos da cena (marcadores) (Morar *et al.,* 2020). A realização desta estimativa pode ser decomposta em um procedimento com três passos (Morar *et al*., 2020; LI *et al*., 2004; Thrun; Burgard; Fox, 2005):

a) coleta de imagens e dados do ambiente por câmeras e outros sensores;
b) extração de características que permitam a identificação dos objetos na cena por meio de classificação digital, e;
c) emprego de algoritmos de regressão para estimar a posição em que o robô está posicionado.

Jondhale *et al*. (2022) propõe o emprego de modelos baseados em máquinas de vetor de suporte (SVR) na estimação de coordenadas. Nesta abordagem as características extraídas das imagens, $z=\{ x_1 \ldots x_n\}$ são valores numéricos que são inferidos em uma função f(z) que

[1] BLE: Bluetooth de baixa energia (*Bluetooth Low Energy*)
[2] LiDAR: *LIght Detection And Ranging*

retorna as coordenadas do objeto no ambiente. Este trabalho considera o emprego de um SVR com duas variáveis dependentes x_r e y_r.

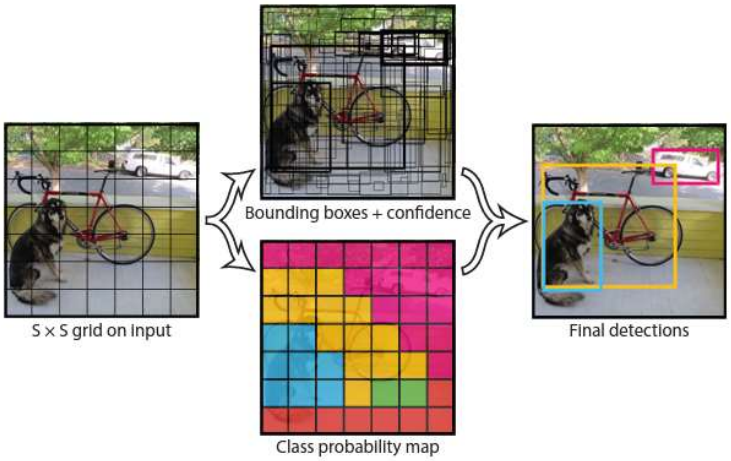
É comum utilizar-se de imagens obtidas a partir de sensores passivos, do tipo câmeras digitais 2D ou 3D. Para Morar *et al.* (2020), o uso de câmeras em localização de robôs, pode ocorrer basicamente de três maneiras, com câmeras fixadas no robô, câmeras fixadas no ambiente e a fusão de dados das câmeras com outros tipos de sensores. Este trabalho considera que as câmeras estão fixadas ao robô. Além disso, explora o fato de que esta abordagem é adequada para ambientes que possuem objetos com características únicas (Roy; Chowdhury, 2021) e que podem ser empregados como marcadores. Sabendo a posição dos marcadores, é possível utilizá-los como referência para realizar a localização do robô.

2.2 YOLO

O algoritmo YOLO (*You Only Look Once*) é um procedimento para detecção de objetos de interesse em uma cena. Este algorimo utiliza uma rede neural convulacional (CNN) para detecção e classificação do objeto e transforma o problema de detecção de objeto em um problema de regressão (Ma *et al*., 2020; Roy; Bose; Bhaduri, 2021). O procedimento de regressão localiza um objeto, *O*, através da estimativa de caixas delimitadoras que circundam *O* na imagem *I* e efetua sua classificação na imagem inteira (Diwan; Anirudh; Tembhurne, 2023). A detecção e classificação ocorre em um passo integrado.

Suponha que *O* pode ser classificado em *C* diferentes categorias (classes). Inicialmente, o procedimento YOLO divide *I* em regiões retangulares que compõem uma grade s X s, $s \in$ N. Na sequência, o procedimento avalia cada célula da grade para verificar se ela contém uma projeção de um objeto. Paralelamente, realiza uma predição das caixas delimitadoras, utilizando-se de uma abordagem sem âncora, onde as caixas delimitadoras são previstas diretamente sem depender de âncoras predefinidas, que se adequam ao tamanho do objeto (referência). Caso uma caixa seja detectada por mais de uma célula para o mesmo objeto, um procedimento chamado "supressão não máxima", elimina as caixas com menores probabilidades resolvendo as sobreposições. Para determinar o objeto alvo, o algoritmo estima a probabilidade da célula conter *O*. Quando a probabilidade de uma hipótese é superior a um limiar pré-especificado, o algoritmo obtém sucesso na detecção de um objeto. (Redmon *et al.,* 2016). Na Figura 1 temos um esquema dessas etapas.

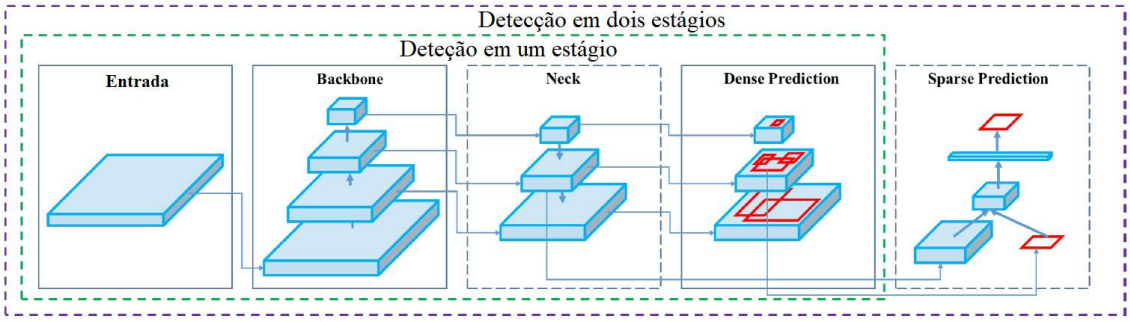
Figura 1- Etapas principais do procedimento YOLO.



Fonte: Redmon e Farhadi, A. YOLO9000: Better, Faster, Stronger. **Web**. 2016

Segundo Bochkovskiy, Wang e Liao, (2020), a arquitetura das redes usadas em procedimentos para detecção de objetos em imagens, pode ser dividida em dois blocos: O primeiro bloco, normalmente treinado na base *ImageNet*, é responsável pela extração de características e é chamado de *backbone*. O segundo, denominado *Head*, é responsável por realizar a classificação e calcular as caixas delimitadoras sobre os alvos detectados. O *Head* pode ter um estágio (*Dense prediction*) ou dois estágios (*Sparse Prediction*). A esses dois blocos principais podem ser adicionados outras camadas para extração de características semânticas da imagem (Roy; Bose; Bhaduri, 2021). Essas camadas são conhecidas por *Neck*. A Figura 2 ilustra a arquitetura de um detector de objetos moderno.

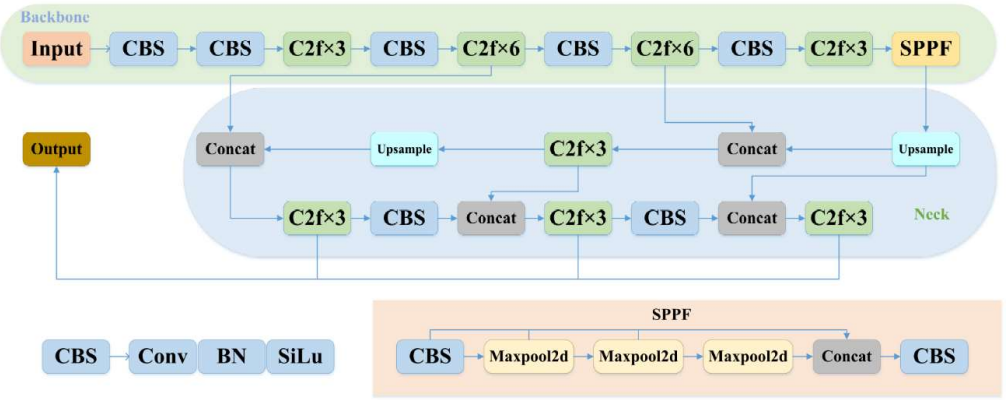
Figura 2 - Detector de objeto moderno.



Fonte: Adaptado de: Bochkovskiy, A.; Wang, C.Y., Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

O YOLOv8 , lançado em 2023 pela empresa Ultralytics, realiza tarefas de detecção, classificação, segmentação, estimativa de pose e rastreamento (Terven; Córdova-Esparza; Romero-González, 2023). A Figura 3 ilustra a arquitetura da rede neural do procedimento YOLOv8. Yue *et al*. (2023) descreve o *backbone* do YOLOv8 como sendo similar ao YOLOv5, contudo, utilizando uma versão adaptada da rede CSPDarknet53 (Wang *et al*., 2019). Uma adaptação descrita, conecta as saídas dos módulos CSP[3] (Sun *et al*., 2024) da rede original a dois fluxos de gradientes paralelos, gerando informações de gradientes mais robustas (módulos C2f[4]) . Além disso, o YOLOv8 incorpora um módulo denominado *Spatial Pyramid Pooling Fast* (SPPF) ao *backbone*, que realiza *pooling* (Géron, 2019) espacial em série e paralelo, aumentando a área de cobertura dos mapas de características e integrando informações em múltiplas escalas (Uygun; Ozguven, 2024).

Figura 3 - Arquitetura simplificada do YOLOv8.



Fonte: LOU, H. *et al*. DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor, 2023

Segundo Terven, Córdova-Esparza e Romero-González (2023) o YOLOv8 implementa dois conceitos importantes: cabeça desacoplada (*decoupled head*) e a não utilização de âncora (*anchor free*). O primeiro conceito refere-se a utilização de ramificações (*heads*) independentes para realização das tarefas de avaliação da probabilidade de uma caixa delimitadora conter um objeto, a classificação e a regressão para localização do tamanho da caixa, melhorando a acurácia do modelo. Enquanto o segundo, sem âncora, elimina a necessidade da utilização de âncoras de referência pré-definidas. Com o objetivo de melhorar a

[3] CSP: Parcial de estágio cruzado (*Cross Stage Partial)*
[4] C2f: Duas camadas de convolução unificas com uma fusão de estágio completo (*full-stage fusion*)

detecção das caixas delimitadoras, principalmente em objetos menores, o YOLOv8 utiliza CioU[5] e DFL[6] (Wu; Dong, 2023) como funções de perda (*loss function*), a função de entropia cruzada binária para a perda de classificação (*classification loss*).

A saída padrão para cada predição do YOLO consiste em uma caixa delimitadora, um conjunto de coordenadas que especifica a região da imagem contendo um objeto detectado. No YOLOv8, as caixas delimitadoras são indicadas pelas coordenadas dos cantos superior esquerdo (x_{se}, y_{se}) e inferior direito (x_{id}, y_{id}), fornecendo uma descrição mais direta dos limites da caixa. O YOLOv8 também pode gerar saídas da forma (x_c, y_c, w, h) no qual x_c e y_c, referem-se ao centroide da caixa, enquanto w e h correspondem à largura e altura da mesma. Além disso, o algoritmo YOLO inclui medidas de confiança para cada detecção, estimando a probabilidade do objeto detectado pertencer a uma das classes predefinidas (Redmon; Farhadi, 2016).

O YOLOv8 possui atualmente cinco versões: *nano* (N), *small* (S), *medium* (M), *large* (L) e *extra-large* (X). A segmentação semântica de instâncias é um tarefa suportada na versão 8 do YOLO, o modelo é conhecido como YOLOv8-Seg. Nessa tarefa o objetivo é, além de obter a caixa delimitadora, identificar os segmentos de pixels que compõe o objeto identificado. Com o detalhamento a nível de pixel é possivel identificar qual pixel faz parte do objeto, distinguindo o da cena (Uygun; Ozguven, 2024; Bai *et al*., 2024). Como saída de uma rede YOLOv8-seg temos uma informação adicional que é referente a máscara de segmentação do objeto localizado em cada caixa delimitadora. Essa é a principal diferença da arquitetura do YOLOv8 para o YOLOv8-seg, segundo Uygun e Ozguven (2024). Essa saída adicional é gerada devido a um módulo de rede convolucional completo denominado Proto, o qual é responsável por produzir as máscaras que representam a segmentação dos objetos, essa máscara é então adicionada ao vetor de saída final do YOLO.

A transferência de aprendizado é uma técnica muito utilizada para treinamentos de rede neurais, em que os pesos de uma rede pré-treinada é utilizado para iniciar o treinamento de uma rede mais especializada (Shin *et al*., 2016). Segundo a documentação da Ultralytics (Ultralytics, 2024), a rede YOLOv8-Seg disponibiliza os pesos pré-treinados no conjundo de dados COCO, enquanto a YOLOv8, no conjunto de dados ImageNet.

As versões disponibilizadas são: YOLOv8n-Seg, YOLOv8s-Seg, YOLOv8m-Seg, YOLOv8l-Seg e YOLOv8x-Seg. A Tabela 1 contém informações, disponibilizadas pela

---

[5] CioU: Interseção sobre União completa (*Complete Intersection over Union*)
[6] DFL: Perda focal por Distribuição (*Distribution Focal Loss*)

Ultralytics, que permitem caracterizar essas implementações em termos de complexidade da arquitetura da rede e velocidade de processamento com imagens de 640 *pixels*. Neste trabalho empregou-se os procedimentos YOLOv8n-Seg (mais simples e rápida), YOLOv8m-Seg (intermediária) e YOLOv8x-Seg (mais complexa e lenta).

Tabela 1 - Métricas de performance do YOLOv8-Seg

Modelo	mAPbox	mAPmask 50-95	Velocidade CPU ONNX (ms)	Velocidade A100 TensorRT (ms)	Params (M)	FLOPs (B)
YOLOv8n-seg	36,7	30,5	96,1	1,21	3,4	12,6
YOLOv8s-seg	44,6	36,8	155,7	1,47	11,8	42,6
YOLOv8m-seg	49,9	40,8	317	2,18	27,3	110,2
YOLOv8l-seg	52,3	42,6	572,4	2,79	46	220,5
YOLOv8x-seg	53,4	43,4	712,1	4,02	71,8	344,1

Fonte: Ultralytics**,** YOLOv8-Seg Documentation

2.3 AUTOLOCALIZAÇÃO BASEADA EM WIFI

Uma tecnologia de sensores que também fornece um mecanismo para coletar dados para autolocalização de robôs móveis é a das redes sem fio (WiFi) (Farid; Nordin; Ismail, 2013; Ibrahim *et al*., 2020; Qin; Zuo; Wang, 2021;). Entre as informações disponibilizadas por sistemas de rede sem fio, o RSSI é um indicador que quantifica a intensidade do sinal de rede recebido por uma antena receptora em relação à energia do sinal enviado pela antena emissora. Dado que o valor de RSSI muda com a distância entre receptor e emissor, os dados representados desta forma têm sido utilizados para estimar a distância de um robô à uma antena de referência e posteriormente, combinado com outras informações, sua posição no ambiente (Xue *et al*., 2017; Yu *et al*., 2014; Zhang *et al*., 2020).

Segundo Rizk, Elmogy e Yamaguchi (2022) uma técnica comum ao utilizar dados RSSI na autolocalização é o RSSI-*fingerprinting*. Essa técnica é dividida em duas fases, *offline* e *online*. Na fase *offline* os dados de sinal RSSI são coletados em diversas posições de referências conhecidas gerando uma base de dados ou um modelo que relaciona as ‘impressões

digitais' (*fingerprinting*) com as localizações. Na fase *online*, a informação derivada dos *fingerprintings* é usada para estimar as posições dos objetos em novos locais.

Os dados obtidos na fase *offline* podem ser utilizados para o treinamento de um modelo de regressão. Em seu estudo Jondhale *et al*. (2022) apontam que esse modelo pode ser um SVR. Segundo eles, a regressão baseada em vetores de suporte é eficaz para capturar relações não lineares entre as entradas e saídas. Além disso, indicam que os SVRs são adequados para localização de alvos móveis.

2.4 REGRESSÃO POR VETORES DE SUPORTE

A regressão por vetor de suporte (SVR) é uma extensão da máquina de vetores de suporte (Anandhi; Chezian, 2013). O procedimento de regressão numérica baseada em máquinas de vetor de suporte objetiva ajustar uma função y=f(x), definida em $R^n \rightarrow R$, a partir de um conjunto de amostras (x_i, y_i) em que $x_i \in X \subset R^n$ e $y_i \in Y \subset R$. f é uma função linear da forma w' x + b em que $w \in R^n$ são os parâmetros numéricos da função e $b \in X$ é o intercepto (Rivas-Perea *et al*., 2021). As estimativas de valores de y_i, z_i, partir de $f(x_i)$ devem atender as seguintes restrições:

- $||w||^2$ é mínimo;
- $-(\varepsilon + \xi_i^*) \leq (y_i - z_i) \leq \varepsilon + \xi_i$.

Nesta expressão ξ_i e ξ_i^* são duas variáveis de folga (Papadimitriou; Steiglitz, 1998) e $\varepsilon > 0$ denota o erro aceitável nas predições. A estimação dos parâmetros de *f* é obtida pela solução do problema abaixo. *C* é uma constante positiva que indica a penalidade por ultrapassar o limite imposto por ε.

$$\text{minimizar} \quad \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{l}(\xi_i + \xi^*) \tag{1}$$

$$objetivo \begin{cases} y_i - r_i - b \leq \varepsilon + \xi_i \\ (\omega, x_i) + b - y_i \leq \varepsilon + \xi_i^*. \\ \xi, \xi \qquad\quad \geq 0 \end{cases} \tag{2}$$

A regressão pode ser aplicada para ajustar relações não lineares entre os elementos de X e Y (Basak; Pal; Patranabis, 2007). Para tanto, é possível aplicar transformações não lineares (funções de kernel) aos elementos de X. Isto gera o conjunto X' que passa a ser uma das entradas do algoritmo de regressão. Este trabalho emprega o *kernel* denominado função de base radial (RBF), conforme equação 3.

$$\boldsymbol{K}\,(\boldsymbol{x},\boldsymbol{y})\;=\;\boldsymbol{exp}(-\Upsilon\;||\,\boldsymbol{x}\;-\;\boldsymbol{y}\,||\,\boldsymbol{2}) \qquad (3)$$

Gunter e Zhu (2007) descrevem um procedimento para estimar os parâmetros do SVR. Chang e Lin (2011) apresentaram uma biblioteca de software para implementação do SVR. Tran, Kuhle e Klau (2024) descrevem uma abordagem que permite empregar o SRV como parte de uma estratégia de regressão com múltiplas variáveis independentes em que cada uma das variáveis alvos tem seus parâmetros ajustados separadamente.

2.5 TRABALHOS RELACIONADOS

Duas revisões sobre tecnologias aplicadas à autolocalização de robôs foram realizadas por Huang *et al*. (2023) e Lukasik *et al.* (2024). O primeiro autor enfatiza a importância da computação visual aliada à fusão de sensores, como de movimento, rádio e LiDARs, além de destacar o papel das técnicas de aprendizado de máquina nesse contexto. Para Huang *et al*. (2023), os sensores amplamente disponíveis, como câmeras e interfaces de rádio (WiFi e Bluetooth), representam uma tendência em evolução. Por outro lado, Lukasik *et al.* (2024), aborda os desafios de localização e mapeamento simultâneo (SLAM), destacando o uso de LiDARs em conjunto com visão computacional, com ênfase no Visual-SLAM. No contexto de tarefas de localização, ambos os autores, exploram o uso combinado de uma gama de tecnologias, incluindo radiofrequência (WiFi, BLE, Bluetooth, entre outros), infravermelho, ultrassom, e IMUs[7], salientando que cada uma possui vantagens e limitações. Assim, a fusão de dados entre diferentes sensores surge como uma abordagem promissora para superar essas limitações.

[7] IMU**:** Unidade de Medição Inercial **(***Inertial Measurement Unit***).**

Azurmendi *et al.*(2023) utilizaram uma câmera monocular para realizar a localização e estimar a distância dos objetos. Os autores empregaram o YOLOv5 para realizar a detecção dos mesmos nas imagens. Os resultados obtidos foram de mAP50-95 superior a 75% e um erro absoluto médio de 0,71 cm. O erro absoluto percentual médio entre os valores reais e valores previstos, MAPE (*Mean Absolute Percentage Error*), foi de 14%. Quatro versões diferentes do YOLOv5 foram avaliadas durante o estudo: N (*nano*), S (*small*), M(*medium*) e L (*large*). Os melhores resultados foram encontrados utilizando a versão M, seguido por N.

Taromi e Klidbary (2024) utilizaram duas imagens obtidas por câmeras *pinhole,* as quais são convertidas para uma imagem 2D. As imagens são processadas com o YOLOv8 para realizar a detecção do objeto na cena. Os dados extraídos do YOLOv8 foram inseridos em uma rede neural multicamadas para realizar a estimativa da altura, largura e distância do objeto. Os resultados obtidos nos experimentos indicam um MAPE de aproximadamente 1,84% em um ambiente com objetos de 50 cm a 200 cm de distância, o tempo de processamento do modelo foi de 0.355 segundos. Um estudo semelhante, também utilizando duas câmeras *pinhole*, foi realizado por Adil, Mikou e Mouhsen, (2022), nele as imagens obtidas são processadas por um algoritmo proposto pelos autores baseado em mapa de disparidade, os resultados foi um MAPE de 1,81% com objetos de 60 cm a 200 cm de distância.

Um trabalho semelhante foi desenvolvido por Fabricio *et al.* (2023). Os autores usaram imagens de um ambiente hospitalar simulado e o algoritmo YOLOv8 para realizar a detecção dos objetos de interesse. O cálculo da distância foi efetuado a partir de uma base de dados com as medidas reais do objeto a uma distância de 2,5m e uma equação matemática utilizando a largura do objeto obtida pelo YOLO. A média de erro variou de 2% a 17% conforme o objeto, elevador e maca, respectivamente.

O estudo de Vajgl, Hurtik e Nejezchleba (2022) discute efeito do desempenho computacional ao utilizar um procedimento baseado no YOLOv3 sobre tempo gasto por uma rotina de cálculo de distância de um objeto. Nesse trabalho, a versão V3 do YOLO foi adaptada para realizar a detecção, classificação e cálculo da distância de forma simultânea. O tempo de processamento para o cálculo da localização foi de 22ms e o erro médio ficou próximo de 11%. Para isso, entre outros ajustes, durante a fase de treinamento o rótulo do objeto de interesse foi identificado juntamente com a distância real.

Sun *et al.* (2021) propôs a utilização de sinalizador BLE para realizar a identificação da posição de *smartphones* e consequentemente pessoas, em um ambiente. Nesse caso foi

utilizado uma CNN cujos parâmetros foram otimizados pelo algoritmo PSO[8] (Kennedy; Eberhart, 1995). Os dados de RSSI obtidos através dos sinalizadores BLE definiram a entrada da rede e foram associados às coordenadas de aquisição x e y e submetidos a CNN para realizar a localização. A acurácia obtida foi de 97.92 %. Os autores destacam o fato de algumas medições terem sidos afetadas por obstruções no sinal de RSSI, devido a objetos, estruturas do ambiente e outros aparelhos.

Cheng *et al*. (2022), também chama a atenção para os ruídos em dados RSSI. Eles utilizam um método não supervisionado DBSCAN[9] que analisa a saída de uma CNN e seleciona as referências necessárias para cada situação. Em seguida utilizam da técnica KNN[10] para realização da localização. Dados RSSI e AOA[11] de dez antenas, foram transformados em uma imagem RGB e utilizada como dado de entrada da CNN. Segundo os autores, o modelo proposto obteve uma acurácia entre 0,1 a 0,3m superior aos modelos tradicionais, sem a utilização de DBSCAN

Este capítulo apresentou as principais tecnologias e métodos utilizados para a autolocalização de robôs, com foco nas técnicas de visão computacional e dados de redes sem fio. A análise da literatura indicou que o uso de informações extraídas a partir da detecção de objetos de referência em imagens, posteriormente tratadas com processamento de imagens, e dados de RSSI provê uma estratégia para autolocalização de robôs em ambientes internos. Esse resultado apoia as decisões metodológicas descritas no próximo capítulo no desenvolvimento de procedimentos de autolocalização baseados em fusão de informação.

---

[8] PSO: Otimização por Enxame de Partículas (*Particle Swarm Optimziation)*

[9] DBSCAN: Agrupamento espacial de aplicações com ruído baseado em densidade (*Density-based spatial clustering of applications with noise*)

[10] KNN: K-enézimo vizinho mais próximo (K-Nearest Neighbour)

[11] AOA: *Angle of arrival*

3 METODOLOGIA

Este capítulo apresenta a metodologia experimental adotada na definição de dois procedimentos para avaliação da fusão de informaçõe na autolocalização do *agrobots* em um ambiente internos. Primeiramente, descreve-se o ambiente de teste e o processo de coleta de dados. Em seguida, detalha-se o treinamento das redes YOLOv8 para detecção e segmentação do objeto de referência na cena e a validação das versões YOLOv8n-seg, YOLOv8m-seg e YOLOv8x-seg. Na sequência, as etapas de treinamento, aplicação e experimentação com os procedimentos de autolicação CVAutoL e FusionAutoL são descritos.

3.1 AMBIENTE DE TESTE E COLETA DE DADOS

3.1.1 Ambiente de teste

O treinamento e os testes foram realizados sobre um conjunto de imagens e dados RSSI coletados em uma casa de vegetação com iluminação natural da Universidade Estadual de Ponta Grossa, denominada Casa de Vegetação 1. Esta foi construída com recursos do projeto CT-INFRA/ FINEP 2013 para campi Estaduais e Municipais. As características do ambiente, que tem aproximadamente 100m^2, podem ser visualizadas na Figura 4.

Figura 4 - Casa de vegetação do projeto CT-INFRA/ FINEP 2013.

Fonte: O autor

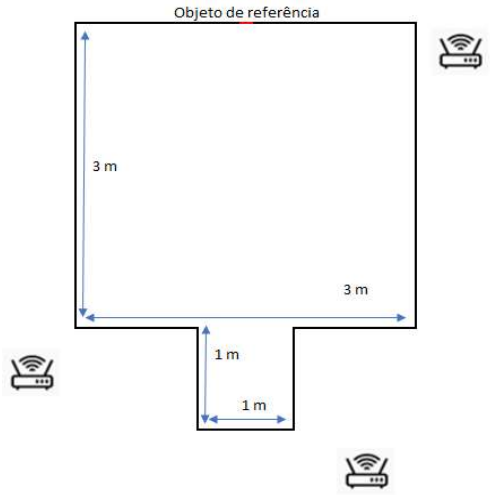
O procedimento de autolocalização envolveu a determinação da posição de um *agrobot* simulado. As coordenadas foram estimadas a partir de um objeto 3D localizado em

uma das paredes internas da casa de vegetação. O objeto em questão era uma caixa de passagem elétrica medindo aproximadamente 15 cm X 15 cm X 10 cm, conforme Figura 5. Os dados de RSSI foram obtidos de três ativos de rede que emitiam sinal de WiFi. A área de movimentação do *agrobot* foi limitada em 10m². As dimensões e forma desta área podem ser visualizadas na Figura 6.

Figura 5 - Caixa de passagem elétrica utilizada como objeto de referência.

Fonte: O autor.

Figura 6 - Layout do ambiente de movimentação do *agrobot.*



Fonte: O autor.

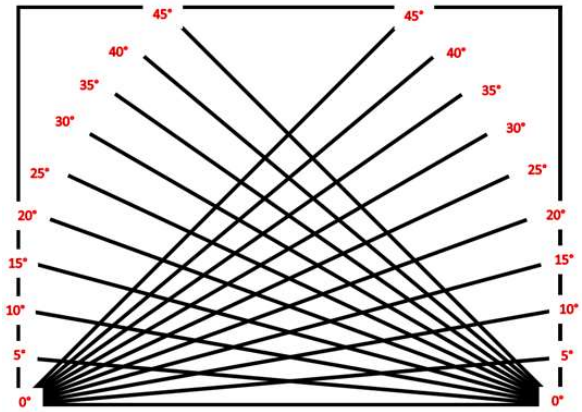
3.1.2 Coletas de dados

A coleta de dados foi dividida em duas fases. Na primeira fase foram coletadas 400 imagens coloridas aleatórias contendo o objeto de referência. Isso deu origem ao conjunto de dados, imgYOLO. Essas imagens foram rotuladas manualmente com emprego da ferramenta online Roboflow (2024) e utilizadas para treinamento e validação das redes YOLOv8-Seg. Na

segunda fase, foram coletadas 20.000 imagens que foram utilizadas para treinamento e validação do procedimento CVAutoL e FusionAutoL. Ao fim deste processo obteve-se o conjunto de imagens, imgAuto.

Durante a aquisição das imagens de imgAuto, a região de operação foi dividida em uma grade de seções quadrangulares de 10 cm x 10 cm. Para cada célula da grade foram capturadas vinte imagens do objeto de referência em diferentes ângulos de visada. As imagens foram obtidas com rotações à esquerda e à direita, conforme a Figura 7. Disto resulta que a base imgAuto possui 20.000 imagens, com dimensões de 3264 x 1836 pixels de três canais de cores, RGB. A Figura 8 ilustra quatro imagens obtidas em uma única posição. Simultaneamente foram obtidas leituras de três ativos de rede emissores de sinal RSSI, posicionadas conforme Figura 6, vinculando-as a cada uma das imagens.

Figura 7 - Ângulos que foram obtidos os dados.



Fonte: O autor

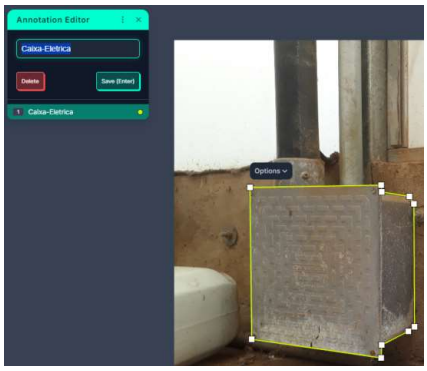
Figura 8 - Exemplos de imagens obtidas na mesma posição.

Fonte: O autor

3.2 TREINAMENTO DO YOLO PARA DETECÇÃO DO OBJETO

O treinamento das diferentes versões do YOLO (YOLOv8n-seg, YOLOv8m-seg e YOLOv8x-seg) foi realizado sobre o conjunto de imagens imgYOLO. O processo de rotulagem e segmentação do objeto de referência foi realizado manualmente utilizando a ferramenta online Roboflow. Para cada imagem do conjunto de dados, uma máscara de segmentação foi desenhada ao redor do objeto, conforme Figura 9. Além disso a base imgYOLO foi submetida a um processo de aumento de dados (*data augmentation*) (Yang et al., 2023) usando a ferramenta Roboflow. Para tanto, foram aplicadas as seguintes transformações sobre as imagens: espelhamento horizontal, rotação entre -15° e 15º e rotação das imagens em 90° no sentido horário e anti-horário. Este procedimento gerou um total de 939 imagens.

Figura 9 - Rotulagem e segmentação de imagem na ferramenta Roboflow.



Fonte: O autor

A base aumentada foi particionada em dois subconjuntos. Um conjunto de imagens de treinamento com 70% (657) das imagens e um conjunto de teste com 30% (282). Durante o treinamento empregou-se o método de transferência de aprendizado (Shin *et al*., 2016) com o uso dos pesos pré-treinados, no conjunto de dados COCO (Ultralytics, 2024), de cada uma das redes, conforme disponibilizados pela empresa Ultralytics.

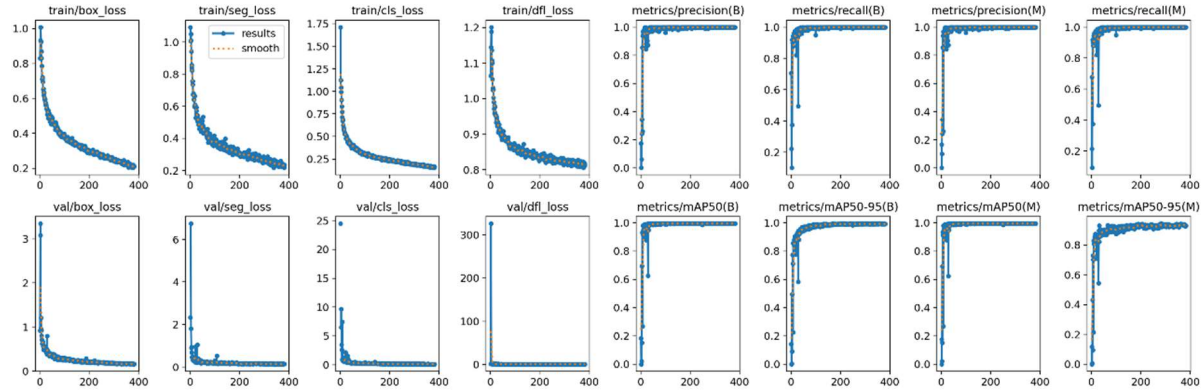
Todas as versões do YOLO foram configuradas com apenas uma classe de objeto para detecção, o objeto de referência. O treinamento foi realizado por 400 épocas, com uma paciência de 100 (padrão no YOLOv8-seg). Estes valores foram definidos empiricamente. A altura e largura das imagens de entrada foram definidas como 640 X 640

O treinamento da rede YOLOv8n-seg, cuja rede tem o menor número de parâmetros e camadas, exigiu a iteração de 278 épocas durante a aprendizagem. A aprendizagem dos

parâmetros da rede YOLOv8m-seg demandou a iteração de 382 épocas. Por fim, foram necessárias 388 épocas para treinar a rede YOLOv8x-seg, a mais complexa das três em termos de número de parâmetros e camadas.

Os resultados do treinamento obtidos pela ferramenta YOLOv8-Seg, vistos na Figura 10, mostram a convergência e desempenho do modelo YOLOv8m-seg. As perdas de treinamento e validação cairam ao longo das épocas e as métricas de precisão e revocação (próximas a 1) indicam capacidade de detecção (Li *et al*., 2023). As métricas mAP50 e mAP50-95 mantiveram valores elevados, acima de 0,9. Os resultados para a rede YOLOv8n-Seg e YOLOv8x-Seg foram similares.

Figura 10- Métricas do treinamento da rede YOLOv8m-Seg.



Fonte: O autor

Após a especificação e o treinamento das redes do YOLO, o desempenho das mesmas foi mensurado com as métricas: precisão (P), revocação (R), mAP50 e mAP50-95. A Tabela 2 enumera os resultados obtidos no treinamento das três redes.

Tabela 2 - Estatísticas do treinamento das redes YOLOv8-Seg

	Caixa delimitadora				Segmentação de Objeto			
YOLO	**P**	**R**	**mAP50**	**mAP50-95**	**P**	**R**	**mAP50**	**mAP50-95**
YOLOv8n-Seg	99,8	99	99,5	98,9	99,8	99	99,5	93,5
YOLOv8m-Seg	99,9	100	99,5	99,3	99,9	100	99,5	94,4
YOLOv8x-Seg	99,9	100	99,5	99,2	99,9	100	99,5	99,9

Fonte: O autor.

Segundo Yu *et al*. (2022), uma métrica utilizada em redes com aprendizado profundo é o IoU (*Intersection over Union*). O IoU, indica a sobreposição entre a caixa delimitadora gerada e a caixa delimitadora prevista. A métrica mAP é calculada sobre diferentes IoUs, obtendo-se a média das precisões (AP) de objetos detectados que possuem um IoUs superior a vários limiares, por exemplo, mAP50 indica que somente as IoUs acima de 50% serão consideradas como corretas, a mAP50-95, é média das detecções onde os limiares variam de 50% a 95%, incrementado em 5%, portanto é uma métrica mais acurada (Padilla; Netto; Silva, 2020).

Neste estudo, as métricas de mAP50-95 para as três redes YOLOv8 foram de 93%, 94% e 99% redes N, M e X, respectivamente. Azurmendi *et al.*(2023), utilizando a base de dados KITTI com YOLOv5 nas versões, N, S, M e L e obteve um mAP50-95 de 73%, 78%, 78 e 81%, respectivamente.

O procedimento de detecção do objeto de referência pelas redes YOLO, foi realizado com as imagens do grupo ImgAuto, que não foram utilizadas no treinamento. As acurácias obtidas foram de 98,8%, 95,3% e 99,1% e o tempo médio em milissegundos de 60, 297 e 935 por imagem, para as versões YOLOv8n-Seg, YOLOv8m-Seg e YOLOv8x-Seg, respectivamente. Conforme observado na Tabela 3.

Tabela 3 - Estatísticas das predições das redes YOLO

YOLO	Objetos Reconhecidos	Falsos Negativos	Falsos Positivos	Confiança média (%)	Tempo médio (ms)	Acurácia
YOLOv8n-Seg	11346	133	249	92,5	60	98,8%
YOLOv8m-Seg	10990	44	533	94,02	297	95,3%
YOLOv8x-Seg	11504	102	127	94,05	935	98,8%

Fonte: O autor

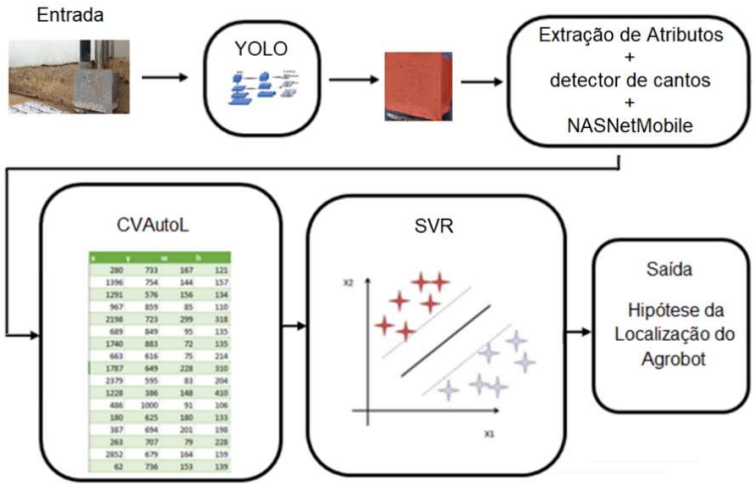
3.3 PROCEDIMENTOS CVAUTOL E FUSIONAUTOL

Esta seção descreve os dois procedimentos de autolocalização usados na realização dos experimentos. O procedimento, CVAutoL explora informações extraídas por processamento digital de imagens e visão computacional para estimar as coordenadas do *agrobot*. O procedimento FusionAutoL, com o objetivo de aumentar a precisão da localização, realiza a fusão de dados obtidos por WiFi com os mesmos dados extraídos das imagens pelo

CVAutoL. Ambos os procedimentos empregam um regressor multiobjetivo com uma SVR para estimar as coordenadas (x_e, y_e) de localização do robô a partir dos dados adquiridos.

A Figura 11 apresenta um esquema do processamento executado pelo procedimento CVAutoL na autolocalização. Como pode ser notado, o processamento começa com a aquisição de uma imagem digital *I* no ambiente de teste, Figura 12, por meio de uma câmera digital instalada no *agrobot*. Esta imagem é usada para alimentar uma rede convolucional YOLO que estima uma caixa delimitadora *B* que contém o objeto em *I* (com confiança igual ou superior a 80%). Um procedimento de extração de atributos é aplicado sobre a segmentação, gerando uma base de dados chamada YoloOut. Para realizar a regressão, essa base é submetida ao procedimento de regressão multiobjetivo. Após o treinamento com o SRV, a máquina de vetores de suporte é usada para estimar a localização do *agrobot*.

Figura 11 - Arquitetura do modelo CVAutoL.



Fonte: O autor

Figura 12 - Exemplo de uma imagem I com o objeto de referência.

Fonte: O autor

A base de dados YoloOut, comum a ambos os procedimentos, possui 55 atributos no total. Esses atributos podem ser divididos em três grupos, de acordo com sua origem. O primeiro grupo é gerado a partir dos dados da saída da rede YOLO e é relacionado às caixas delimitadoras. O segundo grupo é extraído da região delimitada pela máscara de segmentação produzida pelo YOLO usando técnicas de *ad hoc* de processamento digital de imagens e o emprego da rede NasNetMobile (Zoph *et al*., 2018). Por fim, o terceiro grupo contém atributos calculados a partir dos dois primeiros grupos.

O primeiro grupo de informações, obtidos da caixa delimitadora, contém os seguintes dados sobre a localização do objeto na imagem:

a) coordenadas (x_c, y_c) referentes ao ponto central da caixa delimitadora;
b) altura da caixa delimitadora (h);
c) largura da caixa delimitadora (w);
d) coordenada (x_{se}, y_{se}) referente ao canto superior esquerdo;
e) coordenada (x_{id}, y_{id}) referente ao canto inferior direito;
f) valores proporcionais entre o tamanho da imagem e o da caixa delimitadora, gerando os atributos xProp, yProp, wProp e hProp.

As informações acima foram usadas para calcular as informações:

a) coordenada (x_{sd}, y_{sd}) do canto superior direito;
b) coordenada (x_{ie}, y_{ie}) do canto inferior esquerdo.

O segundo grupo foi gerado a partir do objeto *mask*, uma das saídas do YOLOv8-Seg, que contém a máscara que representa a segmentação do objeto da imagem. A Figura 13 mostra um exemplo de segmentação do objeto e sua máscara.

Figura 13 - Fragmento da imagem obtido com os dados da caixa delimitadora com a máscara de segmentação.

Fonte: O autor

A ferramenta *skimage.measure.regionprops* da Scikit-Image (2024) foi empregada para extrair as seguintes informações:

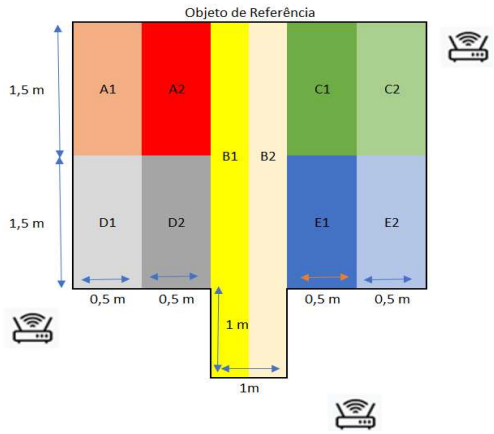
a) Axis_minor_length: Comprimento do menor eixo da elipse que possui os mesmos momentos centrais normalizados de segunda ordem da região;
b) Centroid (xc e yc): posição central da máscara;
c) centroid_local (xcl e ycl): posição central da caixa delimitadora;
d) inertia_tensor_eigvals (iM e im): maior e menor autovalores do tensor de inércia, em ordem decrescente.

Ainda no segundo grupo, um procedimento *ad hoc*, denominado *characteristic_features*, foi desenvolvido para calcular as seguintes características geométricas de um oval aproximado gerado a partir do contorno da máscara.

a) *Center_oval*: centróide do oval (x_{co} e y_{co}): que é a média das coordenadas dos pontos do contorno;
b) *Axis_major_oval*: diâmetro do ponto de maior distância até o centro da elipse;
c) *Area_oval*: calculada utilizando a fórmula de *Shoelace* (Tran, 2017), refere-se a área de um polígono utilizando os vértices da elipse ajustada (Al-Ibadi; Al-Assfor; Al-Ibadi, 2022).

O último atributo do grupo dois, quadrante, reporta qual célula de uma grade de ocupação projetada sobre a área de operação do *agrobot* estava posicionado. Para Siegwart, Nourbakhsh e Scaramuzza (2011) uma grade de ocupação descreve o espaço de movimentação do robô como uma matriz bidimensional cujas células, de dimensões fixas, são projetadas sobre o piso do ambiente. A Figura 14 mostra a grade de ocupação especificada para os experimentos e a identificação de cada célula. Seja Q a grade de ocupação e q a posição do robô na grade durante a aquisição de uma imagem. O valor de q foi estimado com uma rede neural convolucional NasNetMobile (Zoph *et al*., 2018) treinada com as imagens geradas paralelamente a geração da base YOLOOut com a identificação da máscara sobre o objeto de referência encontrado.

Figura 14 - Quadrantes utilizados para rede NasNetMobile.

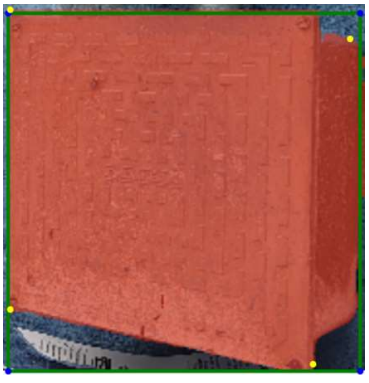


Fonte: O autor

Um segundo algoritmo foi desenvolvido: *find_closest_points*. Ele emprega o cálculo da distância Euclidiana para identificar as coordenadas (x, y) dos quatro pontos do contorno que estão mais próximos aos respectivos cantos da caixa delimitadora. Para cada ponto do contorno da máscara, é calculada a distância em relação a cada canto da caixa delimitadora. O ponto com a menor distância é então selecionado como a coordenada correspondente a esse canto. Essa função retorna as coordenadas destacadas em amarelo na Figura 15, e são referentes aos pontos mais próximos da máscara para os cantos:

a) inferior esquerdo, coordenada (x_{pie}, y_{pie})
b) inferior direito, coordenada (x_{pid}, y_{pid})
c) superior esquerdo, coordenada (x_{pse}, y_{pse})
d) superior direito, coordenada (x_{psd}, y_{psd})

Figura 15 - Caixa delimitadora com a segmentação do objeto e seus respectivos cantos.

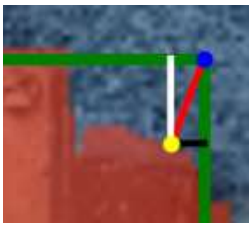


Fonte: O autor

As coordenadas dos pontos mais próximos servem de base para os atributos do terceiro grupo. Nessa etapa, foram calculados dois subgrupos com 12 atributos cada. Ambos os grupos calculam três distâncias para cada um dos quatro cantos do objeto. A diferença entre os grupos está no fato de que, no primeiro, os resultados são expressos em valores absolutos, enquanto no segundo, as distâncias são convertidas proporcionalmente ao tamanho da caixa delimitadora.

Na Figura 16, temos um exemplo dessas medidas: a linha branca representa a distância nas coordenadas Y (DistY), a linha preta indica a distância em X (DistX), e a linha vermelha corresponde à distância euclidiana entre o ponto mais próximo e o canto da caixa delimitadora (DistP).

Figura 16 - Distâncias calculadas entre os cantos da caixa delimitadora e os cantos da segmentação do objeto.



Fonte: O autor

Portanto, os atributos absolutos gerados são:

a) canto superior esquerdo ($DistY_{se}$, $DistX_{se}$ e $DistP_{se}$)
b) canto inferior esquerdo ($DistY_{ie}$, $DistX_{ie}$ e $DistP_{ie}$)
c) canto superior direito ($DistY_{sd}$, $DistX_{sd}$ e $DistP_{sd}$)
d) canto inferior direito ($DistY_{id}$, $DistX_{id}$ e $DistP_{id}$).

Os atributos proporcionais correspondentes são:

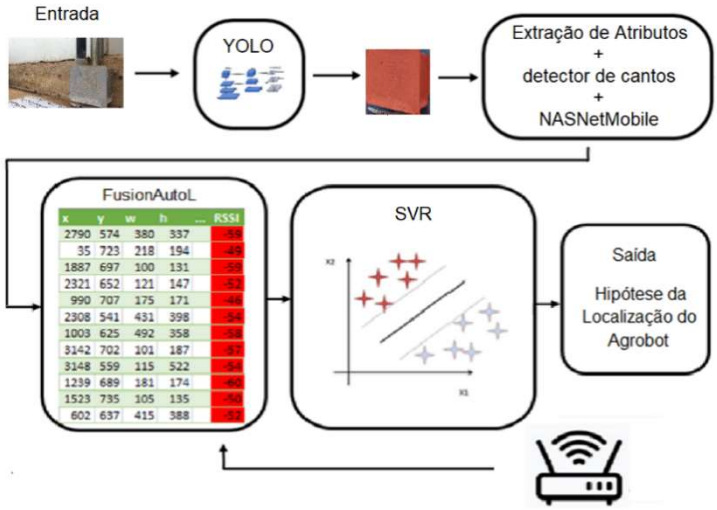
a) canto superior esquerdo ($DistY_{sep}$, $DistX_{sep}$ e $DistP_{sep}$)
b) canto inferior esquerdo ($DistY_{iep}$, $DistX_{iep}$ e $DistP_{iep}$)
c) canto superior direito ($DistY_{sdp}$, $DistX_{sdp}$ e $DistP_{sdp}$)
d) canto inferior direito ($DistY_{idp}$, $DistX_{idp}$ e $DistP_{idp}$).

Para cada uma das três versões do YOLOv8seg, n, m e x, foram geradas uma base de dados YOLOOut, desse modo temos para o procedimento CVAutoL as bases DCV_N, DCV_M e DCV_X, respectivamente.

FusionAutoL

O procedimento FusionAutoL usa uma arquitetura similar ao CVAutoL, a diferença se dará pela adição do valor de três sinais RSSI as bases DCN_N, DCV_M e DCV_X, geradas no procedimento CVAutoL. A Figura 17, mostra a arquitetura FusionAutoL proposta. Nesse modelo os dados são obtidos simultaneamente pela câmera e pela antena receptoras de WiFi (RSSI) do *agrobot*. Os dados RSSI são adicionados a base YOLOOut, gerada pelo CVAutoL, e inseridos como entrada para um regressor multiobjetivo que retorna a hipótese das coordenadas (x_e, y_e) mais provável para localização no *agrobot* no ambiente. Nesse processo são geradas as bases de dados DF_N, DF_M e DF_X, com os dados do YOLOv8-seg, n, m e x, respectivamente.

Figura 17 - Arquitetura do modelo FusionAutoL.



Fonte: O autor

Os atributos adicionais para FusionAutoL são referente as medidas RSSI de três emissores, são eles: rede1, rede2 e rede3.

3.4 BASES DE TREINAMENTO E TESTE

As bases de dados DCV_N, DCV_M, DCV_X, DF_N, DF_M e DF_X foram utilizadas no treinamento e teste da função multiobjetivo para estimar a localização do *agrobot*, mais

especificamente uma parte dos dados foram usados. Isto se deve ao ângulo de visão do *agrobot*, o objeto é visível em 11.479 imagens de imgAuto. Somente essas imagens foram usadas nos testes, as demais foram descartadas.

As imagens também foram inspecionadas para identificar falsos positivos. Apenas aquelas em que o objeto foi reconhecido por, ao menos, uma das redes YOLO foram utilizadas para compor os registros das respectivas bases. Dessa forma, as bases DCV_N e DF_N ficaram com 11.346 registros, as bases DCV_M e DF_M com 10.946 registros, e as bases DCV_X e DF_X com 11.377 registros.

A rede treinada com a arquitetura NasNetMobile, objetiva predizer a célula da grade de ocupação em que as imagens foram adquiridas, fez uso dos pesos pré-treinados da base ImageNet. A função de classificação ajustada foi avaliada em um esquema e validação cruzada estratificada em dez partições. O otimizador Adam com entropia cruzada categórica como função de perda foi empregado no ajuste dos parâmetros da seção multiconectada da rede. Os resultados obtidos nesta etapa de pré-processamento estão listados na Tabela 4.

Tabela 4 - Resultado da classificação da rede NasNetMobile para os modelos do YOLO

YOLO	Acurácia	Desvio Padrão Acurácia	F1-Score	Desvio Padrão F1-Score
YOLOv8n-Seg	90,9	0,07	85,3	0,09
YOLOv8m-Seg	88,9	0,07	88,9	0,09
YOLOv8x-Seg	91,8	0,06	85,6	0,07

Fonte: O autor

O ajuste da máquina de suporte para estimação da localização do *agrobot* no ambiente foi feito com o procedimento *MultiOutputRegressor* do *framework* Scikit-learn (2024a), que ajusta um regressor para cada atributo alvo, coordenadas x_r e y_r. Seu emprego permite realizar uma regressão com algoritmos que não suportam nativamente a regressão multiobjetivo, como é o caso do SVR.

O treinamento do SVR foi implementado em uma rotina com validação cruzada de dez partes com as seguintes configurações: regularização (500), *epsilon* (5), *gamma* (*scale*) e *kernel* (rbf). Esses parâmetros foram obtidos usando *GridSearchCV* do *framework* Scikit-learn (Scikit-learn, 2024b; Buitinck *et al*., 2013). Durante o treinamento, foi utilizada a métrica *neg_mean_squared_error*.

Este capítulo detalhou os procedimentos de autolocalização CVAutoL e FusionAutoL usados para avaliar a eficiência da fusão de informação na estimativa da posição de *agrobots* em casas de vegetação. O próximo capítulo apresenta os experimentos realizados, nos quais essas abordagens foram testadas e avaliadas quanto à precisão e eficiência (tempo de processamento) na autolocalização de *agrobots*.

4 EXPERIMENTOS

Este capítulo apresenta os experimentos realizados com os procedimentos de autolocalização dos algoritmos CVAutoL e FusionAutoL. A Seção 4.1 descreve os procedimentos de seleção e redução dos atributos, ajuste des parâmetros e treinamento do modelo de regressão multiobjetivo com SVR. Cada um dos modelos YOLO foi configurado para detectar e segmentar o objeto de referência, gerando bases de dados com atributos selecionados que foram submetidos a análise preditiva. Na Seção 4.2, são discutidos os métodos de avaliação do desempenho e comparação dos modelos, incluindo as métricas de erro e tempo de processamento, que servirão como indicadores para comparar as configurações e identificar aquelas que oferecem melhor precisão e eficiência.

4.1 TREINAMENTO E TESTE DOS PROCEDIMENTO DE AUTO LOCALIZAÇÃO

Os experimentos de autolocalização dos algoritmos CVAutoL e FusionAutoL podem ser abstraídos em duas etapas. Na primeira etapa, o conjunto de imagens coletadadas, imgAuto, foi inserido na entrada dos algoritmos YOLOv8n-seg, YOLOv8m-seg e YOLOv8x-seg, treinados conforme descrito na Seção 3.2. Para cada uma das imagens foram armazenadas as informações relativas aos três grupos de atributos, descritos na Seção 3.3, gerando assim um registro *r* que foi inserido nas bases DCV_N, DCV_M, DCV_X, DF_N, DF_M e DF_X.

As bases DCV totalizaram 52 atributos, enquanto os conjuntos DF tinham 55 atributos (com o acréscimo dos três atributos relativos a RSSI). Com o objetivo de reduzir a dimensionalidade da base de dados e, consequentemente, a complexidade do modelo, foi implementada uma rotina de seleção de atributos utilizando a função RFE (Eliminação recursiva de Atributos) do *framework* Scikit-learn (Scikit-learn , 2024c; Shah *et al*., 2020). Essa função aplica um processo recursivo para selecionar os *k* melhores atributos para o modelo, eliminando os menos relevantes a cada iteração até encontrar o conjunto ideal. O *score* minimizado foi o erro quadrático.

Uma árvore de decisão (Scikit-Learn, 2024d) foi utilizada como estimador base durante o processo de seleção de atributos. As variáveis-alvo, coordenadas x_r e y_r, foram analisadas separadamente, com processos distintos de seleção de atributos para cada uma. Após

essa análise, os melhores atributos selecionados para cada variável foram combinados, resultando em um único conjunto de características para o modelo.

A quantidade ideal de atributos para o modelo foi identificada utilizando uma rotina exaustiva, que começou com 5 elementos e foi incrementando em blocos de 5 até o total de atributos. Para cada iteração, as métricas MAE (erro absoluto médio - MAE), MSE (erro médio quadrático - EMQ) e RMSE (raiz do erro médio quadrático - REMQ) foram armazenadas. O modelo que apresentou o menor RMSE foi selecionado, indicando aquele com os menores erros.

Para os testes com FusionAutoL a geração dos conjuntos de dados ocorreu de forma similar. A diferença foi a inclusão das leituras de RSSI de forma simultânea a captura das imagems, sendo assim temos três novas base de dados, DF_N, DF_M e DF_X, onde foi adiciona os atributos RSSI às três redes utilizadas.

A estrutura das bases de dados foram adaptadas para cada modelo do YOLO. A Tabela 5, mostra todos os 55 atributos de cada base e indica os atributos selecionados após o processo de redução de dimensionalidade. A Tabela 6 resume a quantidade de atributos.

Tabela 5 - Atributos selecionados por base

(continua)

Atributo	**DCV_N**	**DCV_M**	**DCV_X**	**DF_N**	**DF_M**	**DF_X**
x_c						
y_c		X	X	X	X	X
w						
h	X	X	X	X	X	X
xProp						
yProp					X	
wProp						
hProp						
x_{id}	X					
y_{id}	X	X	X	X		X
x_{sd}				X		
y_{sd}	X	X	X	X	X	X
x_{ie}						

Tabela 5 - Atributos selecionados por base.

(continuação)

Atributo	DCV_N	DCV_M	DCV_X	DF_N	DF_M	DF_X
y_{ie}	X	X	X	X	X	
x_{se}			X			
y_{se}	X	X	X	X	X	X
axis_minor_length			X			
x_{cl}					X	
y_{cl}						
x_{cm}	X	X	X	X	X	
y_{cm}		X				
i_M	X					
i_m		X				
area_oval			X			
axis_major_oval					X	
x_{co}	X	X	X		X	X
y_{co}						
quadrante	X	X	X	X	X	X
$DistP_{id}$	X			X	X	X
$DistX_{id}$		X			X	X
$DistY_{id}$		X				
$DistP_{idp}$						
$DistX_{idp}$					X	
$DistY_{idp}$	X		X			
$DistP_{sd}$		X				
$DistX_{sd}$						
$DistY_{sd}$						
$DistP_{sdp}$						
$DistX_{sdp}$	X	X	X		X	
$DistY_{sdp}$						
$DistP_{ie}$						

Tabela 5 - Atributos selecionados por base.

(conclusão)

Atributo	DCV_N	DCV_M	DCV_X	DF_N	DF_M	DF_X
$DistX_{ie}$			X			
$DistY_{ie}$	X	X	X			
$DistP_{iep}$						
$DistX_{iep}$	X	X				
$DistY_{iep}$	X	X	X	X	X	
$DistP_{se}$						
$DistX_{se}$						
$DistY_{se}$						
$DistP_{sep}$						
$DistX_{sep}$						
$DistY_{sep}$						
rede1				X	X	X
rede2				X	X	X
rede3				X	X	X

Fonte: o Autor

Tabela 6 - Quantidade de atributos por base

DCV_N	DCV_M	DCV_X	DF_N	DF_M	DF_X
16	18	17	14	19	12

Fonte: o Autor

Na segunda etapa do procedimento experimental, o regressor multiobjetivo com uma SVR foi usado para estimar a posição do *agrobot* no ambiente, conforme especificado pelos algorimtos CVAutoL e FusionAutoL. Assim, após a fixação dos hiperparâmetros, o procedimento de ajuste sempre empregou as mesmas configurações ao longo dos testes com a mesma base. O desempenho dos procedimentos foi estimado a partir do processo de validação cruzada em 10 partes.

4.2 ANÁLISE DOS RESULTADOS

A análise dos resultados buscou avaliar o efeito da aplicação da fusão de informações com o desempenho preditivo dos procedimentos de autolocalização. Também pretendeu estimar o impacto de versões simplificadas do algoritmo de detector de objetos (marcador) sobre o tempo de processamento. De forma combinada, o objetivo foi determinar se a fusão de informações permite o emprego de versões simplificadas do algoritmo de detector de objeto sem perda significativa de desempenho.

O desempenho preditivo foi avaliado usando uma métrica de erro, E_2, que calcula a distância euclidiana entre as coordenadas da posição real do *agrobot* e aquela estimada pelos algoritmos CVAutoL e FusionAutoL. De forma análoga foi quantificado a medida de erro E_O, que computa a diferença entre a distância real do *agrobot* para com o ponto de origem do sistema de coordenadas, (0,0), e aquela estimada pelo procedimento. Também calculou-se o EMA das estimativas de x_e e y_e em relação as coordenadas de referência em que as imagens foram tomadas (x_r, y_r).

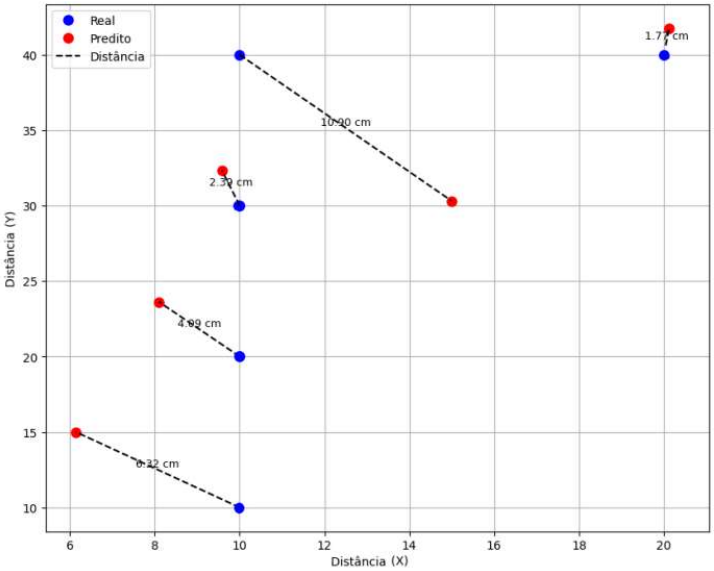
A Tabela 7 lista cada uma das configurações de teste. As Figuras 18 e 19, demonstram os elementos dos cálculos. Na primeira, E_2, o ponto real é mostrado em azul, a hipótese de localização em vermelho e a linha pontilhada entre os dois representa a distância euclidiana entre os pontos. Na segunda, E_O, temos em azul a distância real até a origem, em vermelho a distância predita e em destaque a diferença.

Tabela 7 - Configuração dos componentes testados

Configuração	**Componentes**
CV-N	CVAutoL com YOLOv8n-Seg
CV-M	CVAutoL com YOLOv8m-Seg
CV-X	CVAutoL com YOLOv8x-Seg
F-N	FusionAutoL com YOLOv8n-Seg
F-M	FusionAutoL com YOLOv8m-Seg
F-X	FusionAutoL com YOLOv8x-Seg

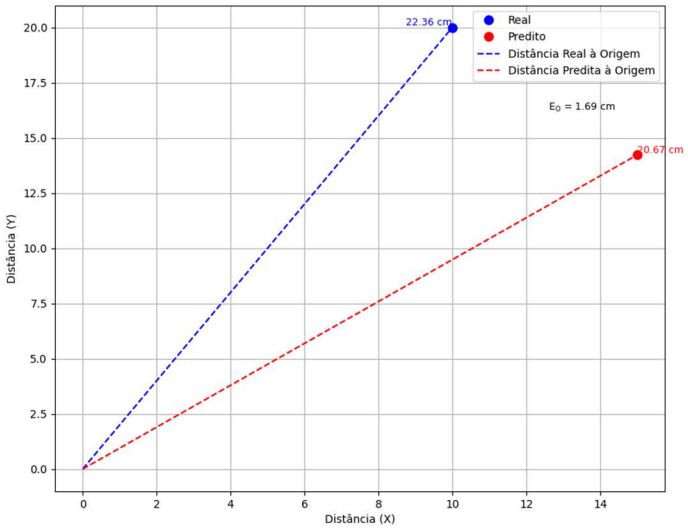
Fonte: O autor

Figura 18 - Exemplo de erros absolutos entre os pontos.



Fonte: O autor

Figura 19 - Diferença entre o ponto real e o predito em relação a origem.



Fonte: O autor

O desempenho preditivo das configurações de CVAutoL e FusionAutoL testadas foram submetidas à análise de variância (ANOVA) de uma via (Kim, 2017) para verificar se existiam diferenças significativas nos desempenhos médios observados. O teste assumiu a seguinte hipótese nula (h_0): os seis modelos avaliados não apresentam diferenças estatísticas significantes em termos do critério de desempenho alvo da análise. A hipótese alternativa foi (h_1), os modelos apresentam diferenças significativas em termos de desempenho. A ANOVA

foi complementada com o teste *post-hoc*, Tukey-Krammer, a fim de identificar as diferenças entre os pares de procedimentos nos casos em que h_0 é rejeitada.

As médias dos erros E_2 e E_O foram submetidas a um ajuste de função quadrática em que a abcissa era a distância em que a imagem foi tomada, definida como d_{xy}. As médias foram calculadas a intervalos de 50 cm (com exclusão mútua). A escolha da função quadrática foi determinada após análise gráfica. A taxa de crescimento destes erros para cada algoritmo foi definida como a primeira derivada da função quadrática (Sviercoski, 2008). O objetivo desta análise foi comparar a taxa de crescimento destes erros entre as diferentes configurações dos procedimentos de autolocalização.

Os resultados relativos ao tempo de processamento (t_p) foram computados em um ambiente do Google Colab, utilizando uma máquina virtual equipada com um processador Intel Xeon CPU @ 2.20GHz (4 núcleos, 8 threads), com cache L3 de 56 MB e 46 GB de memória RAM disponível para uso. Importante destacar que nesse processo não houve o uso de GPU[12] (*Graphics Processing Unit*). O valor para t_p foi registrado em segundos e corresponde ao tempo necessário para o processamento de cada registro nas seguintes etapas: detecção do objeto pela rede YOLO, a extração dos atributos e a predição realizada pela máquina de vetor de regressão. Todos estes tempos também foram comparados usando ANOVA e Tukey-Krammer. A hipótese nula foi que não existe diferença estatística significante no tempo gasto no cômputo da posição pelos modelos avaliados. A hipótese alternativa foi da existência de diferença.

Adicionalmente, a técnica da fronteira de Pareto (Souza, 2021) foi empregada para realizar a análise multiobjetivo do desempenho das diferentes configurações dos algoritmos em termos da precisão na determinação da posição e do tempo de processamento. As funções objetivo avaliadas foram os erros E_2 ou E_O versus o tempo de processamento t_p. O objetivo foi verificar quais configurações eram dominantes em relação a esses dois fatores. A fronteira de Pareto permite identificar as soluções que não são dominadas e que, portanto, representam os melhores compromissos entre objetivos conflitantes. Segundo Talbi (2009), em problemas de maximização, um vetor de objetivo *u* domina um vetor de objetivo *v* se duas condições forem atendidas: (a) todos os elementos de *u* são maiores ou iguais aos elementos correspondentes em *v*; e (b) pelo menos um componente de *u* é estritamente maior.

[12] GPU = Unidade de processamento Gráfico

5 RESULTADOS E DISCUSSÃO

Este capítulo apresenta uma análise dos resultados obtidos com os procedimentos de autolocalização CVAutoL e FusionAutoL. A avaliação foca na precisão das estimativas de posição do *agrobot* e no tempo de processamento dispendido pelos procedimentos para cada configuração de YOLOv8-Seg. Para tanto, os valores de erro EMA_X, EMA_Y, E_2 e E_O para CVAutoL e FusionAutoL foram estimados e comparados por meio de estatística descritiva e análise gráfica. Estes resultados também foram submetidos à análise estatística para avaliar se as diferenças observadas no desempenho dos procedimetos são significativas. A análise multiobjetivo foi realizada para determinar a dominância de cada configuração de teste em termos dos erros de localização e o tempo de processamento.

5.1 DESEMPENHO PREDITIVO

O processo de regressão, descrito na Seção 5.1, foi executado sobre as seis configurações descritas na Tabela 7. O desempenho preditivo com a validação cruzada obtido em cada teste é apresentado na Tabela 8. Naquela tabela, a primeira coluna identifica a configuração e a segunda e terceira coluna enumeram as médias e desvios do EMA de X_e e Y_e. Estes erros são indicados por EMA_X e EMA_Y respectivamente. As duas últimas colunas listam os resultados médios e os desvios de E_2 e E_O.

Tabela 8 - Resultados do procedimento CVAutoL e FusionAutoL

Configuração	EMA_X		EMA_Y		E_2		E_O	
	Média	**Desvio**	**Média**	**Desvio**	**Média**	**Desvio**	**Média**	**Desvio**
CV – N	9,41	0,28	15,42	0,39	19,38	0,36	7,02	0,17
F – N	7,44	0,24	12,39	0,24	15,50	0,32	5,27	0,16
CV – M	9,14	0,31	15,17	0,36	19,04	0,45	6,87	0,18
F – M	6,94	0,19	12,18	0,34	15,06	0,35	4,99	0,10
CV – X	9,11	0,19	14,87	0,51	18,81	0,54	6,82	0,25
F – X	8,53	0,26	13,32	0,32	17,14	0,37	6,29	0,25

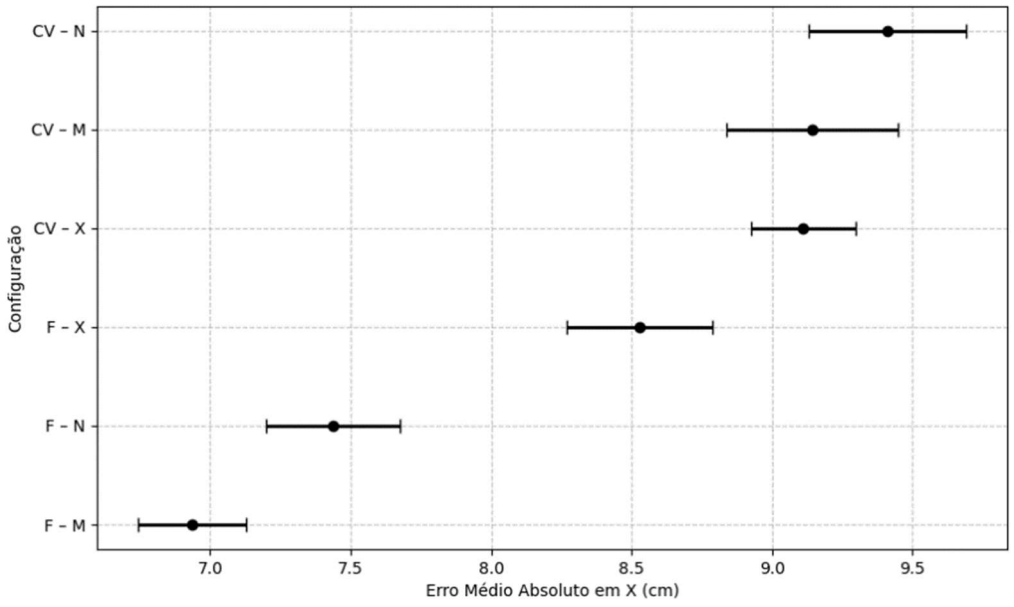
Fonte: O autor

5.1.1 Resultados da coordenada X_e

Os valores obtidos para EMA_X na Tabela 8 indicam que os procedimentos de autolocalização que empregaram fusão de sensores obtiveram erros médios inferiores àqueles obtidos com o CVAutoL com respeito a abscissa da posição estimada. Em particular, a configuração de FusionAutoL que teve o maior erro (F-X), produziu um EMA menor do que a configuração do CVAutoL com o menor erro (CV-X). O gráfico da Figura 20 mostra a dispersão dos erros (desvio padrão) em relação aos valores médios deste *score* e concorda com as observações destacadas.

O teste ANOVA de uma via foi executado e o *valor-P* obtido, 1.2×10^{-31}, levou à rejeição da hipótese nula (h_0). A Tabela 9 enumera os resultados do teste *Post-hoc* de Tukey-Kramer. Os resultados do teste também apoiam as considerações sobre EMA_X. Nessa tabela, também é possível notar que as configurações que utilizaram CVAutoL não apresentaram diferenças estatisticamente significativas entre si. Contudo, aquelas que empregaram FusionAutoL apresentaram diferenças, em média, mais precisas que todas as configurações de CVAutoL

Figura 20 - Comparação do EMA em X por configuração.



Fonte: O autor

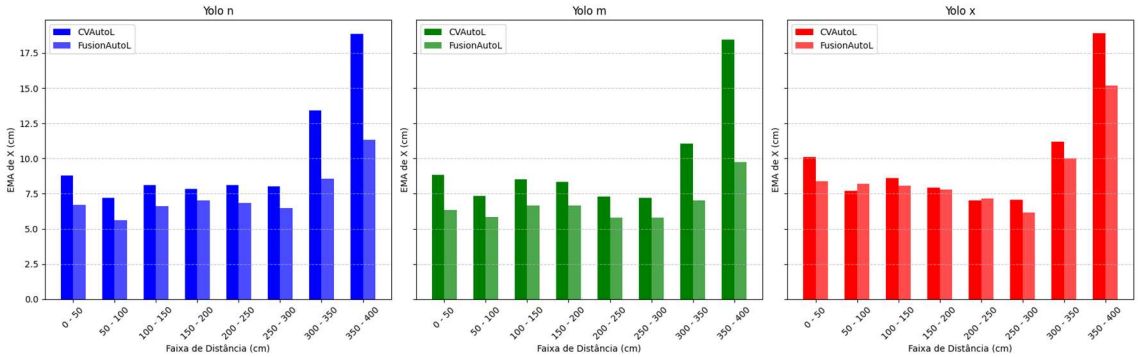
Tabela 9 - Diferenças estatísticas do teste Tukey-Krammer para EMA_X

Configuração 1	Configuração 2	Valor-P	Configuração 1	Configuração 2	Valor-P
CV – M	CV – N	0,169	CV – N	F – X	0
CV – M	CV – X	0,999	CV – X	F – M	0
CV – M	F – M	0	CV – X	F – N	0
CV – M	F – N	0	CV – X	F – X	0
CV – M	F – X	0	F – M	F – N	0,0005
CV – N	CV – X	0,0922	F – M	F – X	0
CV – N	F – M	0	F – N	F – X	0
CV – N	F – N	0			

Fonte: O autor

A Figura 21 ilustra a variação de EMA_X em relação à distância que o *agrobot* encontrava do objeto de referência. Nela é possível observar que a fusão de informações gerou erros médios menores por toda faixa de valores de X. A Figura também destaca que as implementações que usam Fusão mantém o EMA_X abaixo de 10 cm para valores maiores de d_{xy}. Isto sugere que os dados de WiFi contribuem para redução dos erros na estimativa. A configuração FusionAutoL F-M produziu o menor erro médio.

Figura 21 - EMA de X para FusionAutoL e CVAutoL, para diferentes versões do YOLO.



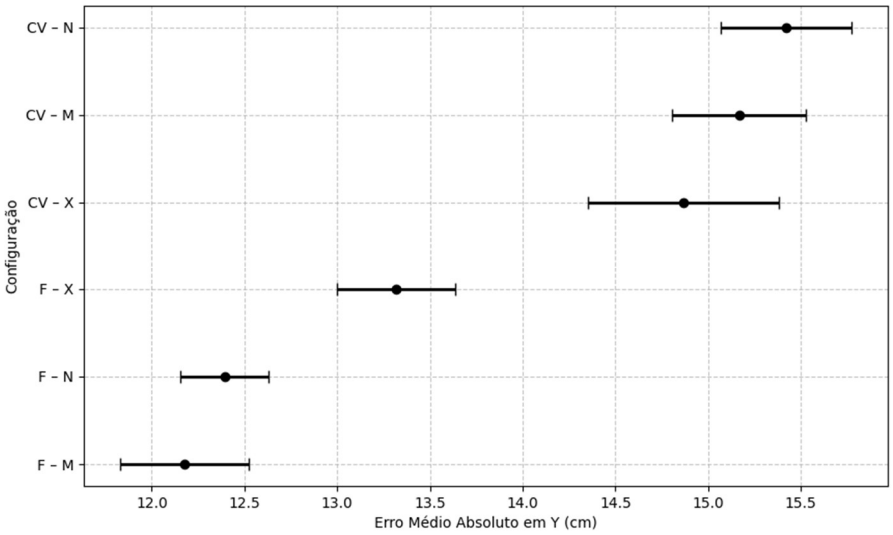
Fonte: O autor

5.1.2 Resultados da coordenada Y_e

Os resultados para EMA_Y estão enumerados na Tabela 8. A qual mostra que as configurações com FusionAutoL produziram valores menores para este *score*. Novamente, a

configuração de FusionAutoL F-M, produziu o menor erro médio. Contudo, há uma sobreposição de F-M e F-N quando se considera os desvios. A Figura 22 permite a visualização destes resultados.

Figura 22 - Comparação do EMA em Y por configuração.



Fonte: O autor

O *valor-P* do teste ANOVA para EMA_Y foi 6.52×10^{-31}. Este valor permitiu a rejeição da hipótese nula (h_0). A análise de pares, via *Post-hoc*, gerou o resultado mostrado na Tabela 10. Mais uma vez, as configurações baseadas em FusionAutoL obtiveram resultados significativamente menores que aquelas de CVAutoL. No entanto, agora, nem todas as cofigurações de FusionAutoL diferem estatisticamente entre si. Isto pode ser notado pela análise dos intervalos dos resultados de FusionAutoL F-M e F-N. Ver Figura 20.

Tabela 10 - Diferenças estatísticas do teste Tukey-Krammer para EMA_Y

(continua)

Configuração 1	**Configuração 2**	**Valor-P**	**Configuração 1**	**Configuração 2**	**Valor-P**
CV – M	CV – N	0,638	CV – N	F – X	0
CV – M	CV – X	0,4471	CV – X	F – M	0
CV – M	F – M	0	CV – X	F – N	0
CV – M	F – N	0	CV – X	F – X	0

Tabela 10 - Diferenças estatísticas do teste Tukey-Krammer para EMA_Y

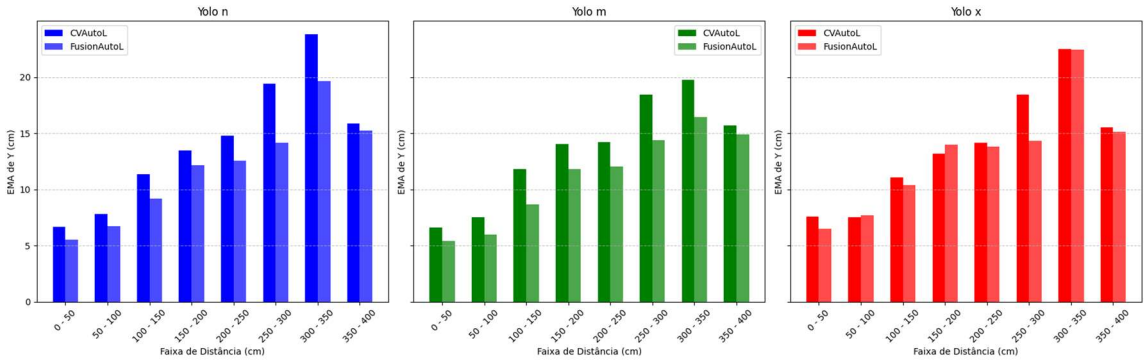
(conclusão)

Configuração 1	Configuração 2	Valor-P	Configuração 1	Configuração 2	Valor-P
CV – M	F – X	0	F – M	F – N	0,7697
CV – N	CV – X	0,0158	F – M	F – X	0
CV – N	F – M	0	F – N	F – X	0
CV – N	F – N	0			

Fonte: O autor

A Figura 23 ilustra a variação do desempenho de CVAutoL e FusionAutoL em termos de EMA_Y ao longo da distância de coleta da imagem. Nela observa-se que o erro na predição de Y ultrapassa o valor de 15 cm quando a distância d_{xy} é maior que 2,5 metros para todas as configurações de CVAutoL. Para FusionAutoL, isto ocorre somente para $d_{xy} > 3m$. Tais resultados sugerem que a fusão do sinal de WiFi com os dados de visão contribuiu, de forma menos preponderante, para redução do erro absoluto na estimativa da ordenada do *agrobot* à medida que este se afastava do objeto de referência.

Figura 23 - EMA de Y para FusionAutoL e CVAutoL, para diferentes versões do YOLO.



Fonte: O autor

5.1.3 Resultados de E_2

As menores médias para o valor de E_2 foram derivadas a partir dos testes com fusão de dados; ver Tabela 8. A ANOVA detectou diferenças significativas nos valores médios de E_2

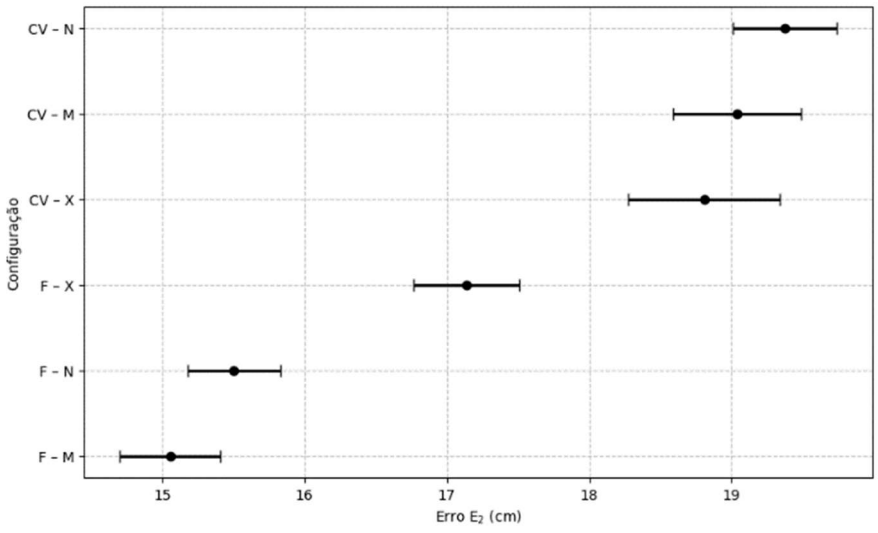
reportados (*valor-P* = 2.54 x10^{-34}). O teste de *Post-hoc* (Tabela 11) permitiu inferir que as diferenças observadas para este tipo de erro foram significativas quando confrontou-se os resultados obtidos com e sem fusão de dados. Como nos casos anteriores, o gráfico com médias e desvios padrões, Figura 24, permite visualizar este padrão de desempenho.

Tabela 11 - Diferenças estatísticas do teste Tukey-Krammer para E_2

Configuração 1	**Configuração 2**	**Valor-P**	**Configuração 1**	**Configuração 2**	**Valor-P**
CV – M	CV – N	0,4433	CV – N	F – X	0
CV – M	CV – X	0,794	CV – X	F – M	0
CV – M	F – M	0	CV – X	F – N	0
CV – M	F – N	0	CV – X	F – X	0
CV – M	F – X	0	F – M	F – N	0,1555
CV – N	CV – X	0,0318	F – M	F – X	0
CV – N	F – M	0	F – N	F – X	0
CV – N	F – N	0			

Fonte: O autor

Figura 24 - Erro médio de E_2 por configuração.

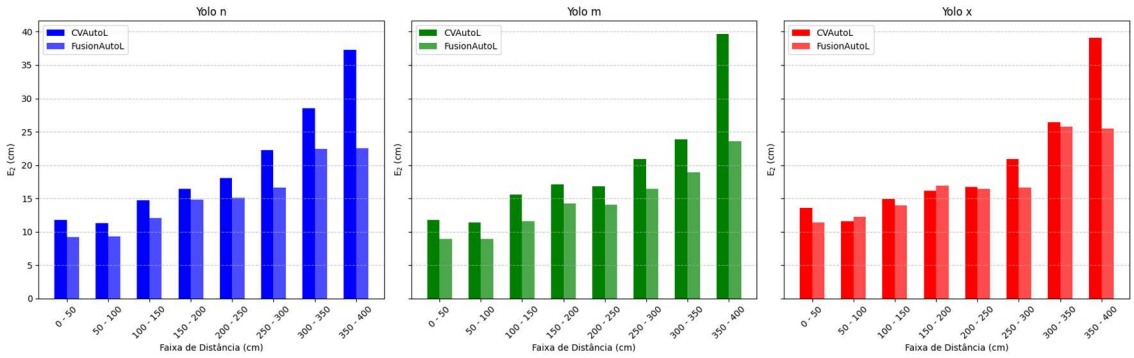


Fonte: O autor

O efeito da distância do *agrobot* ao objeto marcador sobre *E_2* é visto na Figura 25. Nela, se observa que o erro cresce com esta distância para todos os algoritmos. Contudo, o erro

associado às configurações de FusionAutoL são menores que aqueles ligados a CVAutoL e E_2 em todos os intervalos. Além disso, deve-se notar que FusionAutoL produziu um E_2 que se manteve abaixo de 15 cm por uma faixa maior de d_{xy}.

Figura 25 - Erro médio de E_2 para FusionAutoL e CVAutoL, para diferentes versões do YOLO.



Fonte: O autor

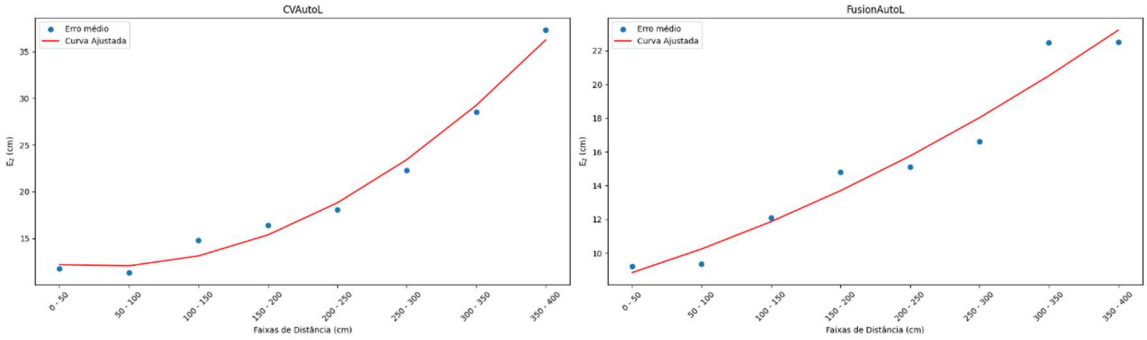
Os dados da Figura 25 foram, para cada uma das configurações testadas, ajustados à uma parábola. Os gráficos referentes ao ajuste aparecem das Figuras 26 a 28. Os valores do coeficiente linear da derivada (2a) das parábolas são listadas na Tabela 12. A Tabela também enumera os valores de R^2; todos acima de 0,9. Os valores de 2a também indicam que E_2 cresceu mais lentamente nas configurações que usaram a fusão de dados.

Tabela 12 - Parábola ajustada para E_2

Configuração	2a	R^2
CV – N	-0,019	0,99
F – N	0,014	0,95
CV – M	0,040	0,96
F – M	0,002	0,96
CV – X	-0,051	0,98
F – X	-0,011	0,92

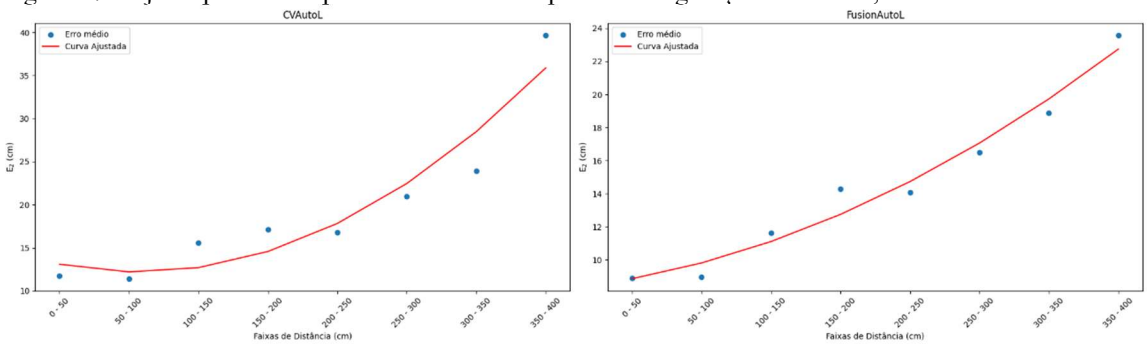
Fonte: O autor

Figura 26 - Ajuste parabólico para YOLO N. A esquerda configuração CVAutoL, a direita FusionAutoL.



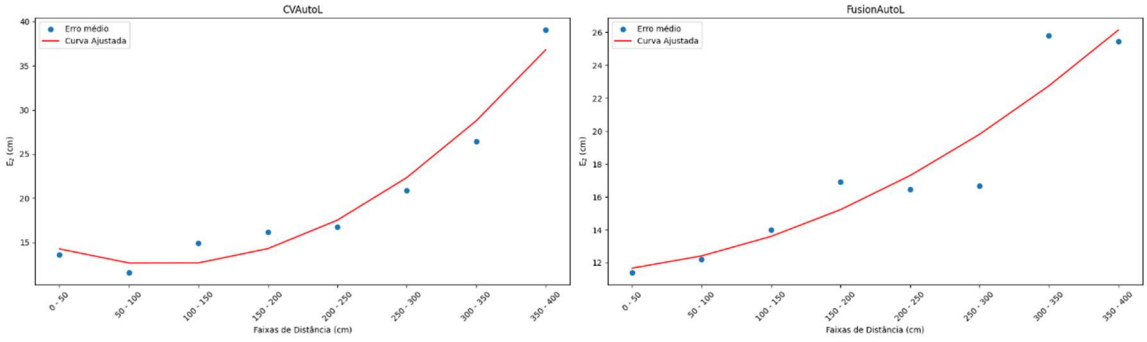
Fonte: O autor

Figura 27 - Ajuste parabólico para YOLO M. A esquerda configuração CVAutoL, a direita FusionAutoL.



Fonte: O autor

Figura 28 - Ajuste parabólico para YOLO X. A esquerda configuração CVAutoL, a direita FusionAutoL.



Fonte: O autor

5.1.4 Resultados de E_O

A autolocalização com fusão de dados apresentou resultados médios menores para E_O (Tabela 8). O *valor-P* da ANOVA de 3,83 x 10^{-34} indica que as diferenças entre as configurações são significativas. O teste *Post-hoc* da Tabela 13 sugere que as diferenças entre as configurações CVAutoL e FusionAutoL são significantes. A Figura 29 ilustra essa diferença de

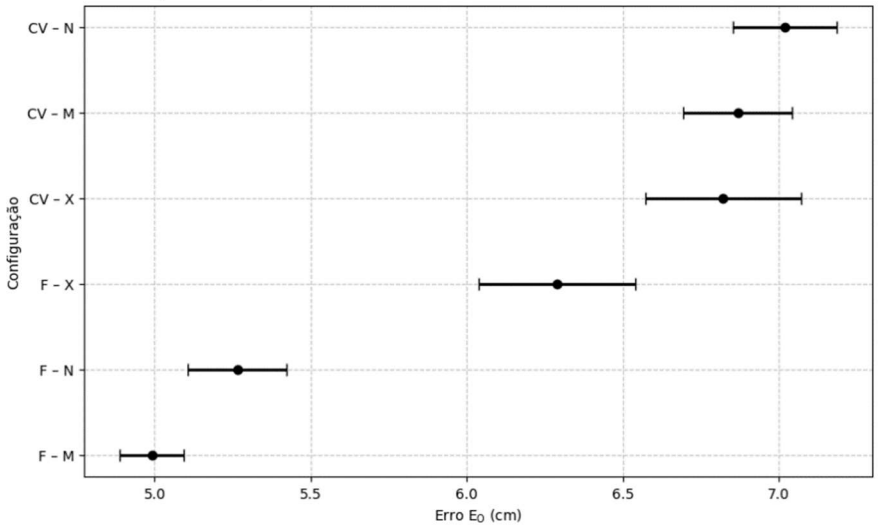
comportamento para FusionAutoL e CVAutoL. Nela também é possível notar que os valores médios de F-M e F-N não diferem estatisticamente.

Tabela 13 - Diferenças estatísticas do teste Tukey-Krammer para E_O

Configuração 1	Configuração 2	Valor-P	Configuração 1	Configuração 2	Valor-P
CV – M	CV – N	0,488	CV – N	F – X	0
CV – M	CV – X	0,994	CV – X	F – M	0
CV – M	F – M	0	CV – X	F – N	0
CV – M	F – N	0	CV – X	F – X	0
CV – M	F – X	0	F – M	F – N	0,0261
CV – N	CV – X	0,2048	F – M	F – X	0
CV – N	F – M	0	F – N	F – X	0
CV – N	F – N	0			

Fonte: O autor

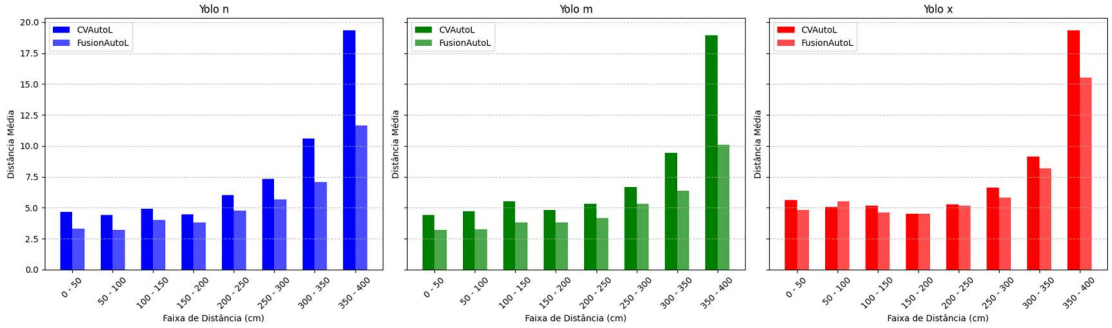
Figura 29 - Erro médio de E_O por configuração.



Fonte: O autor

A Figura 30 mostra que E_O cresceu mais lentamente em relação a d_{xy} nos procedimentos baseados em fusão de dados nos casos do algoritmo YOLO-M e YOLO-N. Adicionalmente, nota-se que o erro médio de todas as configurações ficou entre 3 e 16 cm.

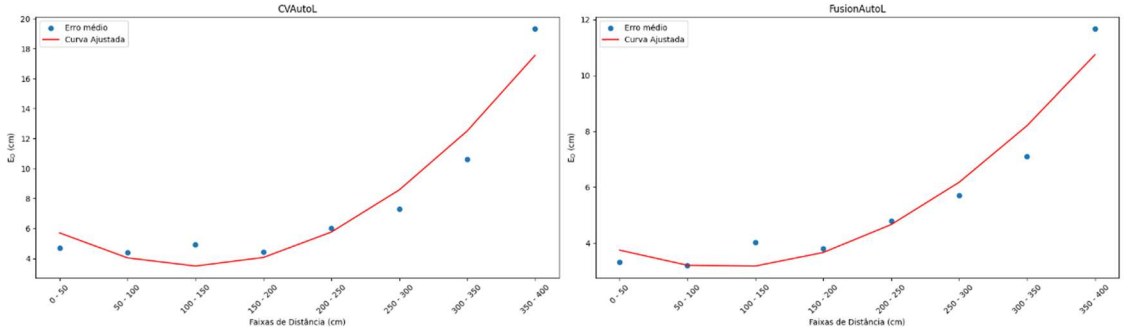
Figura 30 - Erro médio de E2 para FusionAutoL e CVAutoL, para diferentes versões do YOLO.



Fonte: O autor.

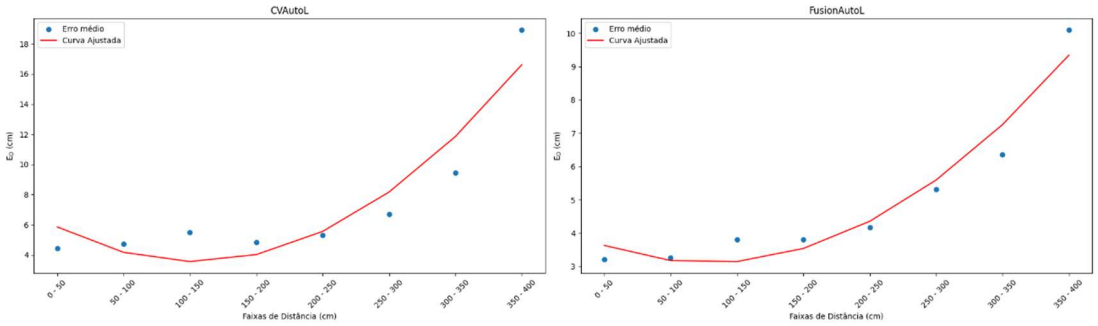
As parábolas ajustadas para $d_{xy} \times E_0$ podem ser vistas das Figuras 31 a 33. Os valores dos coeficientes lineares das derivadas das parábolas e R^2 destes ajustes são listados na Tabela 14. Assim como observado para E_2, este conjunto de resultados fornece evidências que apoiam a conclusão de que E_O cresce mais lentamente nas configurações que usaram FusionAutoL.

Figura 31 - Ajuste parabólico para YOLO N. A esquerda configuração CVAutoL, a direita FusionAutoL.



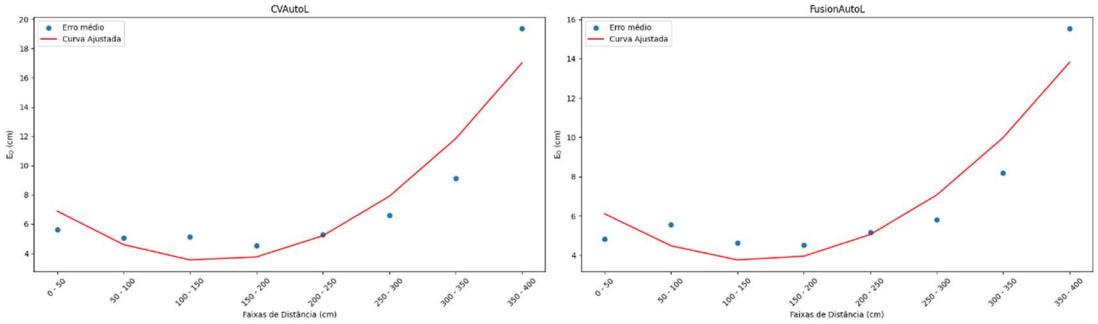
Fonte: O autor

Figura 32 -Ajuste parabólico para YOLO M. A esquerda configuração CVAutoL, a direita FusionAutoL.



Fonte: O autor

Figura 33 - Ajuste parabólico para YOLO X. A esquerda configuração CVAutoL, a direita FusionAutoL.



Fonte: O autor

Tabela 14 - Parábola ajustada para E_O

Configuração	2a	R^2
CV – N	-0,060	0,95
F – N	-0,038	0,89
CV – M	-0,065	0,92
F – M	-0,036	0,86
CV – X	-0,075	0,93
F – X	-0,076	0,86

Fonte: O autor

5.1.5 Tempo de processamento (t_p)

O tempo de processamento (t_p) foi avaliado em segundos. Seu cômputo considerou os tempos de execução descritos na Seção 4.2. As médias dos tempos por configuração estão descritos na Tabela 15. O teste ANOVA resultou em um *valor-P* de $2x10^{-228}$, o que rejeita a hipótese nula (h_0).

Tabela 15 - Tempo médio de processamento por configuração

(continua)

Configuração	T_p YOLO	T_p	Desvio Padrão
CV – N	0,060	0,16	0,00001
F – N	0,060	0,16	0,00004

Tabela 16 - Tempo médio de processamento por configuração

(conclusão)

Configuração	T_p YOLO	T_p	Desvio Padrão
CV – M	0,297	0,40	0,00001
F – M	0,297	0,40	0,00004
CV – X	0,937	1,04	0,00001
F – X	0,937	1,04	0,00002

Fonte: O autor

Os dados obtidos mostram que a autolocalização baseada na versão N foi 15,3 vezes mais rápida do que as configurações que usam YOLO-X e 4,8 vezes mais rápidas do que aquelas que usam YOLO-M. Assim, em resumo, o tempo de processamento variou, basicamente, com a versão do algoritmo YOLO empregado. Esta conclusão é apoiada pela comparação dos tempos indicados na segunda e terceira colunas da Tabela 15. Além disso o teste de *Post-hoc*, Tabela 16, mostra que a comparação de configurações que empregaram diferentes versões do YOLO tiveram tempos médios estatisticamente diferentes.

Os resultados indicados na Tabela 15 também estão de acordo com os dados de desempenho do YOLOv8-Seg disponibilizados pela Ultralytics (Tabela 2).

Tabela 17 - Diferenças estatísticas do teste Tukey-Krammer para t_p

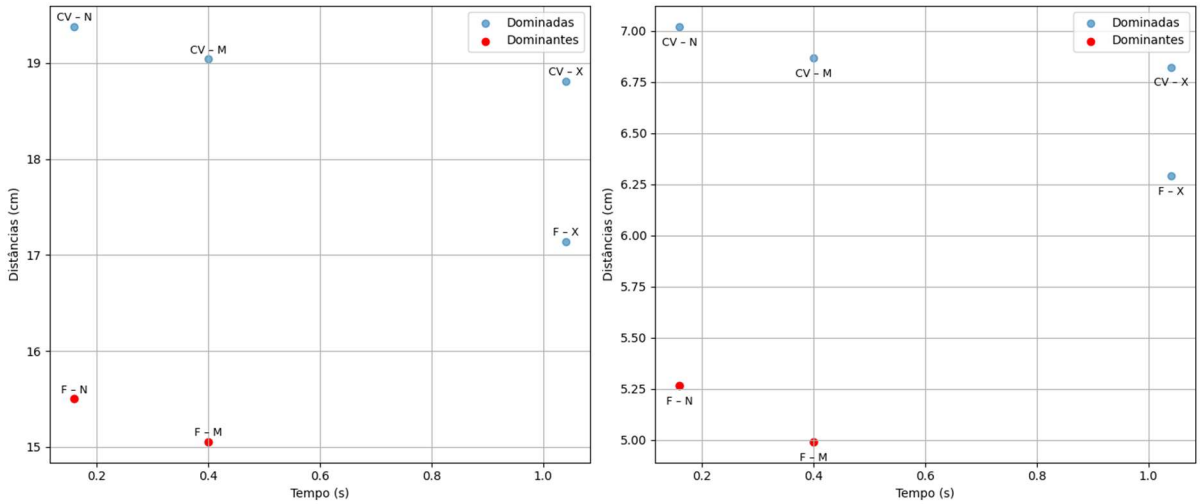
Configuração 1	Configuração 2	Valor-P	Configuração 1	Configuração 2	Valor-P
CV – M	CV – N	0	CV – N	F – X	0
CV – M	CV – X	0	CV – X	F – M	0
CV – M	F – M	0,995	CV – X	F – N	0
CV – M	F – N	0	CV – X	F – X	1
CV – M	F – X	0	F – M	F – N	0
CV – N	CV – X	0	F – M	F – X	0
CV – N	F – M	0	F – N	F – X	0
CV – N	F – N	0,947			

Fonte: O autor

5.1.6 Análise Multiobjetivo

O gráfico da Figura 32 (esquerda) mostra o resultado da fronteira de Pareto com respeito às médias de E_2 e do tempo de processamento para todas as configurações testadas. A Figura 34 (direita) apresenta a fronteira para $E_O \times t_p$. As configurações dominantes são destacadas em vermelho.

Figura 34 - Fronteira de Pareto para Tempo de Processamento e distância.



Fonte: O autor

Ao analisar a fronteira é possível visualizar que, em ambos os gráficos, os procedimentos de autolocalização dominantes empregaram implementações mais rasas, (menos parâmetros e camadas) do algoritmo YOLO e fusão de informações. Obviamente, as diferenças nas soluções dominantes indicam que em situações aplicadas a escolha de um arranjo específico depende dos objetivos específicos da tarefa a ser realizada. Existem diversas abordagens para executar tal escolha (Laurent; Le Mehaute; Schumaker, 1993), contudo, o ponto importante a ser notado aqui é que os resultados obtidos fornecem evidência da viabilidade de se empregar algoritmos baseados em redes rasas na tarefa de detectar objetos marcadores e métodos de fusão de dados para incrementar o desempenho de sistemas de autolocalização em tempo real sem aumento significativo do erro na autolocalização.

A fim de comparar os resultados obtidos com o estudo de Azurmendi *et al.* (2023) que também emprega uma única câmera na aquisição das imagens para estimar a distância de um

objeto até o robô, os resultados sobre E_O foram recalculados para a métrica MAPE. Naquele trabalho, que não faz uso da fusão de informações adicional, o MAPE relativo de E_O foi de 14% . Este valor é cinco vezes maior do que o melhor resultado médio obtido com FusionAutoL (2,6% - YOLOv8m-Seg) e quatro vezes maior que o obtido com CVAutoL (3,4% - YOLOv8m-Seg).

A diferença observada entre o MAPE de FusionAutoL e CVAutoL e aquele descrito no trabalho de Azurmedi *et al* (2023) pode derivar, entre outros fatores, do uso de diferentes versões do YOLO e da fusão de informação por FusionAutoL. Isto porque, aqueles autores também comparam o impacto de diferentes implementações de uma versão do YOLO, no caso, uma versão adaptada do YOLOv5, para determinar a distância de alvos. Assim, as diferenças de desempenho preditivo entre CVAutoL e o procedimento de Azurmedi *et al* (2023) pode estar relacionada a versão do YOLO. Contudo há que se notar que a fusão de dados permitiu um incremento no ganho de CVAutoL. Também deve ser destacado que aqueles autores relatam que o emprego das versões *medium* e *nano* contribuíram para derivação de erros menores, quando comparadas a versão *large.*

Fabricio et al. (2024) utilizam imagens simuladas de uma câmera e o algoritmo YOLOv8 para estimar qual a distância de um robo para com determinados objetos. O erro absoluto da distância estimada para os alvos escolhidos (elevador e cadeira de rodas) variou entre 12 cm e 16 cm. Em termos de erro absoluto médio para E_O, CVAutoL obteve um valor entre 5 cm e 7 cm. Em termos relativos, Fabricio et al. (2024) reportam um erro de 2% a 6%. Comparativamente, FusionAutoL obteve erros relativos de 2,4% a 3,4%. Aqui deve ser notado que as diferenças observadas podem estar relacionadas às diferenças estruturais dos objetos usados em cada trabalho.

O modelo proposto por Vajgl, Hurtik e Nejezchleba (2022), utiliza uma versão do YOLOv3 ajustada pelos autores para estimar distância. O procedimento proposto pelo autor, que obteve o melhor resultado naquele trabalho, atingiu um MAPE de 11%, esse valor é 3 vezes superior ao valor obtido com CVAutoL (3,4% - YOLOv8m-Seg). Dois fatores devem ser considerados para tal diferença, o tamanho dos objetos estudados e a escala de medidas, enquanto CVAutoL utiliza valores até 4 m, Vajgl, Hurtik e Nejezchleba (2022) utiliza objetos de até 90 m de distância. O desempenho de FusionAutoL foi 4 vezes superior (2,6% - YOLOv8m-Seg).

Vajgl, Hurtik e Nejezchleba (2022) também destacam o crescimento do erro absoluto conforme o aumento de d_{xy}. Tal comportamento também foi observado para os procedimentos FusionAutoL e CVAutoL. Nos três procedimentos foi observado um crescimento polinomial do erro em relação a d_{xy}. Adicionalmente, a fusão de informações gerou um procedimento de autolocalização em que o erro cresceu mais lentamente com a distância.

Resultados na mesma ordem de magnitude que CVAutoL e FusionAutoL foram obtidos por Taromi e Klidbary (2024) e Adil, Mikou e Mouhsen (2022). O MAPE ambos os trabalhos apresentaram um valor de 1,84%, e 1,81% para faixa de distância, d_{xy}, entre 60 cm e 200 cm. Para efeitos de comparação o MAPE de FusionAutoL com YOLOv8m-Seg foi recalculado para esta faixa de distâncias. O MAPE foi de 2,8%, o que representa 1,5 vezes o MAPE relatado por Taromi e Klidbary (2024) e 1,5 vezes o MAPE descrito em Adil, Mikou e Mouhsen (2022). Deve ser observado que ambos os trabalhos usaram duas câmeras para aquisição de imagens simultaneamente. Assim, as diferenças aqui indicadas podem ser derivadas deste fator.

O tempo de execução dos modelos FusionAutoL e CVAutoL variaram de forma crescente da versão mais rasa N para a mais profunda M, em valores absolutos de 0,16s e 1,04s respectivamente, considerando o tempo de detecção do objeto, extração de atributo e hipótese de localização. Assim, as implementações de CVAutoL são 2,6 (no melhor caso) e 5 vezes (no pior resultado) mais lentas que aqueles relatados por Azurmendi *et al.*(2023), em que a versão mais rasa, *nano*, teve um desempenho de 0,065 e a versão profunda, 0,223s. Essa diferença, entre outros fatores, pode ser justificada pelo fato de que Azurmendi *et al.*(2023) usou uma versão adapatada do YOLOv5 para fazer a detecção do marcador. Além disso, aquele trabalho não efetua a fusão de dados. Os resultados também foram semelhantes para a análise multiobjetivo entre precisão e tempo de processamento, no estudo de Azurmendi *et al.*(2023) os modelos *nano* e *medium*, dominaram os demais. Isto concorda com os mesmos resultados encontrados pelos procedimentos FusionAutoL e CVAutoL.

Os resultados para EMA_x, EMA_y, E_2 e E_O, também indicam que a fusão de informações reduziu os erros médios indicados por estes *scores*. Além disso, indicam que a fusão permite o uso de versões rasas de redes com o objetivo de ganho computacional. Os ganhos observados quando do uso de FusionAutoL mostram uma redução de 1,31 vezes em termos de EMA_x, 1,24 para EMA_y, 1,26 para E_2 e por fim 1,37 vezes para E_O, considerando os modelos com a versão M do YOLOv8-Seg. Adicionalmente, vale observar que o ganho de desempenho obtido com a

fusão de dados está em acordo com as revisões de literatua de Huang *et al*. (2023) e Lukasik *et al.* (2024).

6 CONSIDERAÇÕES FINAIS

Este trabalho avaliou a possibilidade de ganho de precisão com o uso de fusão de informação de dados obtidos com a aplicação da técnica de visão computacional com dados de RSSI (WiFi) na autolocalização de agrobots em casas de vegetação. A principal motivação para este estudo foi contribuir para o desenvolvimento de agrobots móveis em casas de vegetação. Principalmente, considerando-se o esforço dos governos do Brasil e do estado do Paraná para avançar no uso da tecnologia de informação ao campo.

A tarefa abordada foi a estimativa das coordenadas e distância de um *agrobot* (simulado) a um objeto de referência em uma casa de vegetação usada para pesquisas em Agronomia e Computação Aplicada. Assim, o problema tradado envolvia a autolocalização em ambientes fechados. A estimativa da posição do *agrobot* foi atacada como um problema de regressão não linear.

A fim de avaliar a eficiência da fusão de informações, dois procedimentos de autolocalização foram testados. O primeiro, CVAutoL, explora somente informações extraídas a partir de imagens. O segundo, FusionAutoL, integra dados de RSSI com informações visuais. Em ambos os algoritmos, a detecção do objeto de referência na cena foi efetuada com o algoritmo YOLOv8-Seg. Foram testadas versões completas e rasas deste algoritmo durante os experimentos.

Os resultados indicam que a fusão de informações contribuiu para redução do erro nas estimativas de localização do *agrobot* durante os experimentos. Isto é, os erros absolutos relacionados às estimativas das coordenadas x e y, e a distância do *agrobot* ao marcador diminuíram quando do uso combinado de dados extraídos das imagens e RSSI. Adicionalmente, observou-se que o emprego de redes rasas, mais rápidas, não reduziu o desempenho dos procedimentos em termos de acurácia nas estimativas de posição.

Os trabalhos futuros pretendem avaliar alguns tópicos que puderam ser observados pela análise da literatura a fim de melhorar os resultados obtidos. Primeiro, o emprego de múltiplas câmeras na aquisição do objeto de marcação. Segundo, o emprego de implementações customizadas do algoritmo de detecção de objetos (YOLO). Além disso, pretende-se avaliar a eficiência de empregar-se os procedimentos desenvolvidos como parte de um método para

autolocalização e mapeamento de ambientes (SLAM) baseado em modelos probabilísticos gráficos.

REFERÊNCIAS

ACHARYA, D.; KHOSHELHAM, K.; WINTER, S. BIM-PoseNet: Indoor Camera Localisation Using a 3D Indoor Model and Deep Learning from Synthetic Images. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 150, p. 245–258, 2019.

ADIL, E; MIKOU, M; MOUHSEN, A. A novel algorithm for distance measurement using stereo camera. ***CAAI Transactions on Intelligence Technology***, v.7, p.177-186, 2022.

AL-IBADI, M. A.; AL-ASSFOR, F. K.; AL-IBADI, A. An Automatic Self Shape-Shifting Soft Mobile Robot (A4SMR). **Robotics**, v. 11, n. 6, p. 118, 2022.

ANANDHI, V.; CHEZIAN, R. M. Support Vector Regression to Forecast the Demand and Supply of Pulpwood. **International Journal of Future Computer and Communication**, v. 2, n. 3, p. 266, 2013.

AZURMENDI, I. et al. Simultaneous Object Detection and Distance Estimation for Indoor Autonomous Vehicles. **Electronics**, v. 12, n. 23, p. 4719, 2023.

BAGAGIOLO, G. et al. Greenhouse Robots: Ultimate Solutions to Improve Automation in Protected Cropping Systems—A Review. **Sustainability, Basel**, v. 14, n. 11, p. 6436, 2022.

BAI, R. et al. Automated Construction Site Monitoring Based on Improved YOLOv8-seg Instance Segmentation Algorithm. **IEEE Access**, v. 11, p. 139082-139096, 2023.

BASAK, D.; PAL, S.; PATRANABIS, D. C. Support Vector Regression. **Neural Information Processing – Letters and Reviews**, v. 11, n. 10, p. 203-214, 2007.

BEKEY, G. A. Autonomous Robots: From Biological Inspiration to Implementation and Control (Intelligent Robotics and Autonomous Agents series). **The MIT Press**, p. 594, 2017.

BERZ, E. L.; TESCH, D. A.; HESSEL, F. P. RFID indoor localization based on support vector regression and k-means. **IEEE International Symposium on Industrial Electronics,** v. 24, p. 1-6, 2015.

BI, J. et al. PSOSVRPos: WiFi Indoor Positioning Using SVR Optimized by PSO. **Expert Systems with Applications**, v. 222, 2023.

BOCHKOVSKIY, A.; WANG, C.Y.; LIAO, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. **European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases (2013),** p.108-122, 2013.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A Library for Support Vector Machines. **ACM Transactions on Intelligent Systems and Technology**, v. 2, n. 3, p. 1-27, 2011.

CHENG, F, et al. Improved CNN-Based Indoor Localization by Using RGB Images and DBSCAN Algorithm. **Sensors (Basel, Switzerland)**, v. 22, n. 23, p. 9531, 2022

DIWAN, T.; ANIRUDH, G.; TEMBHURNE, J. V. Object Detection Using YOLO: Challenges, Architectural Successors, Datasets and Applications. **Multimedia Tools and Applications**, v. 82, n. 6, p. 9243–9275, 2023.

DWIJAYANTI, S. at al., Real-time object detection and distance measurement for humanoid robot using you only look once. **Bulletin of Electrical Engineering and Informatics**, v. 13, n. 6, p. 4134-4146, 2024.

FABRICIO, F. et al. Autonomous Mobile Pharmacy (AMPharmy): Identification of hospital objects and their respective positions. **In: Latin American Robotics Symposium, Brazilian Symposium on Robotics, and Workshop on Robotics in Education, LARS/SBR/WRE 2023**. IEEE, p. 532-536, 2023.

FARID, Z.; NORDIN, R.; ISMAIL, M.; Recent Advances in Wireless Indoor Localization Techniques and System, **Journal of Computer Networks and Communications**, v.2013, p. 12, 2013

FAROOQ, M. S. et al. Internet of Things in Greenhouse Agriculture: A Survey on Enabling Technologies, Applications, and Protocols. **IEEE access**, v. 10, p. 53374–53397, 2022

FOROUGHI, F. et al. A CNN-Based System for Mobile Robot Navigation in Indoor Environments via Visual Localization with a Small Dataset. **World Electric Vehicle Journal**, v. 12, n. 3, p. 134, 2021.

FRAGAPANE, G. et al. Planning and Control of Autonomous Mobile Robots for Intralogistics: Literature Review and Research Agenda. **European Journal of Operational Research**, v. 294, n. 2, 2021, p. 405–426, 2021.

GÉRON, A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2. ed. **Sebastopol: O'Reilly Media**, 2019.

GUNTER, L.; ZHU, J. Efficient Computation and Model Selection for the Support Vector Regression. **Neural Computation**, v. 19, n. 6, p. 1633-1653, 2007.

HAO, Y. et al. Understanding the impact of image quality and distance of objects to object detection performance. In: **2022 International Conference on Image Processing (ICIP). IEEE**, 2022.

HUANG, J. et al. Indoor Positioning Systems of Mobile Robots: A Review. **Robotics, Basel**, v. 12, n. 2, p. 47, 2023.

HUSEN, M. N.; LEE S. Indoor Human Localization with Orientation Using WiFi Fingerprinting. **Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication**, p. 1–6, 2014.

IBRAHIM, M. et al. Wi-go: accurate and scalable vehicle positioning using wifi fine timing measurement, **Proceedings of the ACM MobiSys (2020)**, p. 312–324, 2020.

IBWE, K, et al. Indoor Positioning Using Circle Expansion-Based Adaptive Trilateration Algorithm, **Journal of Electrical Systems and Information Technology**, v. 10, n. 1, p. 10–14, 2023.

JONDHALE, S. R. et al. Support Vector Regression for Mobile Target Localization in Indoor Environments. **Sensors**, v. 22, n. 1, p. 358, 2022.

JUNOH, S. A.; SUBEDI, S.; PYUN, J.Y. Crowdsourcing Landmark-Assisted Localization with Deep Learning. **Future Generation Computer Systems**, v. 144, p. 256–270, 2023.

KENNEDY, J.; EBERHART, R. Particle Swarm Optimization. In**: Proceedings of the IEEE International Conference on Neural Networks (ICNN)**, Vol. 4, p. 1942-1948, 1995.

KHAN, F.; RAFIQUE, S.; KHAN, G. M. Low-cost smart metering using deep learning. **International Journal of Information Science and Technology**, Edição Especial, p. 93-104, 2024.

KIM, T. K. Understanding One-Way ANOVA Using Conceptual Figures. **Korean Journal of Anesthesiology**, v. 70, n. 1, p. 22–26, 2017

KIM, Y. H. et al. Sequential Batch Fusion Magnetic Anomaly Navigation for a Low-Cost Indoor Mobile Robot. **Measurement: journal of the International Measurement Confederation**, V.213, 2023.

LAURENT, P.-J.; LE MEHAUTE, A.; SCHUMAKER, L. L. Curves and Surfaces for Geometric Design: A Practical Guide. **1st ed. Berlin: Springer**, 1993.

LI, G. et al. Place Recognition Based on Gabor Filters and SVMs for Mobile Robot Self-Localization. **IEEE Conference on Robotics, Automation and Mechatronics**, v. 2, p. 1112–1116, 2004.

LI, P. et al. Tomato Maturity Detection and Counting Model Based on MHSA-YOLOv8. **Sensors**, *23*, 6701, 2023.

LUKASIK, S; SZOTT, S; LESZCZUK, M. Multimodal Image-Based Indoor Localization with Machine Learning—A Systematic Review. **Sensors, Basel,** v. 24, n. 18, p. 6051, 2024.

LOU, H. et al. DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera **Sensor. Electronics**, v. 12, n. 10, p. 2323, 2023.

LUKASIK, S.; SZOTT, S.; LESZCZUK, M. Multimodal Image-Based Indoor Localization with Machine Learning—A Systematic Review. **Sensors**, v. 24, n. 18, p. 6051, 2024.

LUO, J. et al. Indoor Mapping Using Low-Cost MLS Point Clouds and Architectural Skeleton Constraints. **Automation in Construction**, v. 150, 2023.

MA, H. et al. Detection of Collapsed Buildings in Post-Earthquake Remote Sensing Images Based on the Improved YOLOv3. **Remote Sensing (Basel, Switzerland)** v. 12, n. 1, p. 44, 2020.

MENG, Y. et al. Visual-Based Localization Using Pictorial Planar Objects in Indoor Environment**. Applied Sciences**, v. 10, n. 23, p. 8583, 2020.

MORAR, A. et al. A Comprehensive Survey of Indoor Localization Methods Based on Computer Vision. **Sensors (Basel, Switzerland)**, v. 20, n. 9, p. 2641, 2020.

PADILLA, R.; NETTO, S. L.; SILVA, E. A. B. da. A Survey on Performance Metrics for Object-Detection Algorithms. IEEE International Conference on Systems, Signals and Image Processing (IWSSIP), 2020.

PAPADIMITRIOU, C. H.; STEIGLITZ, K. Combinatorial Optimization: Algorithms and Complexity. New York: **Dover Publications**, 1998.

PARANÁ. Governo do Estado. **Conectividade rural. Inova Paraná,** 2024. Disponível em: https://www.inova.pr.gov.br/Pagina/Conectividade-Rural. Acesso em: 18 out. 2024.

PITTS, H. Warehouse robot detection for human safety using YOLOv8. In: ***SoutheastCon 2024*, Atlanta, GA, EUA**, p. 1184-1188, 2024.

QIN, F.; ZUO, T.; WANG, X. CCpos: WiFi Fingerprint Indoor Positioning System Based on CDAE-CNN. **Sensors (Basel, Switzerland),** v. 21, n. 4, p. 1114, 2021.

REDMON, J. et al. You Only Look Once: Unified, Real-Time Object Detection. **IEEE Conference on Computer Vision and Pattern Recognition (CVPR),** pp. 779–788, 2016.

RIVAS-PEREA, P. et al. Support Vector Machines for Regression: A Succinct Review of Large-Scale and Linear Programming Formulations. **International Journal of Intelligence Science**, v. 3, p. 1-10, 2021.

RIZK, H.; ELMOGY, A.; YAMAGUCHI, H. A Robust and Accurate Indoor Localization Using Learning-Based Fusion of Wi-Fi RTT and RSSI. **Sensors**, v. 22, n. 7, p. 2700, 2022.

ROBOFLOW. **Roboflow: The universal computer vision platform**, 2024. Disponível em: <https://app.roboflow.com>. Acesso: 17 nov. 2024

ROY, A. M.; BOSE, R.; BHADURI, J**.** A Fast Accurate Fine-grain Object Detection Model Based on YOLOv4 Deep Neural Network. **Neural Computing & Applications** v. 34, n. 5, p. 3895-921, 2021

ROY, P.; CHOWDHURY, C. A Survey of Machine Learning Techniques for Indoor Localization and Navigation Systems. **Journal of Intelligent & Robotic Systems** v. 101, n. 3, 2021.

RUBIO, F.; VALERO, F.; LLOPIS-ALBERT, C. A Review of Mobile Robots: Concepts, Methods, Theoretical Framework, and Applications. **International Journal of Advanced Robotic Systems**, v. 16, n. 2, p. 172988141983959–14. 2019.

RUSSELL, S; NORVING P. Inteligência Artificial Tradução da Terceira Edição. **GEN LTC**; 3ª edição, p. 1016, 2013.

SCIKIT-IMAGE. **skimage.measure.regionprops — skimage documentation** 2024. Disponível em: https://scikit-image.org/docs/stable/api/skimage.measure.html#skimage.measure.regionprops. Acesso em:
19 out. 2024.

SCIKIT-LEARN. **sklearn.multioutput.MultiOutputRegressor — scikit-learn documentation.** 2024a. Disponível em: <https://scikit-image.org/docs/stable/api/skimage.measure.html#skimage.measure.regionprops> Acesso em: 18 out. 2024.

SCIKIT-LEARN. **sklearn.model_selection.GridSearchCV — scikit-learn documentation**, 2024b. Disponível em: <https://scikit-learn.org/dev/modules/generated/sklearn.model_selection.GridSearchCV.html>. Acesso em: 18 out. 2024.

SCIKIT-LEARN. **sklearn.feature_selection.RFE — scikit-learn documentation**. 2024c. Disponível em: <https://scikit-learn.org/dev/modules/generated/sklearn.feature_selection.RFE.html>. Acesso em: 18 out. 2024.

SCIKIT-LEARN. **sklearn.tree.DecisionTreeRegressor — scikit-learn documentation.** 2024d. Disponível em: <https://scikit-learn.org/dev/modules/generated/sklearn.tree.DecisionTreeRegressor.html> Acesso em: 18 out. 2024.

SHAH, S. et al. A comparative study of feature selection approaches: 2016-2020. **International Journal of Scientific & Engineering Research**, v. 11, n. 2, p. 469, 2020.

SHIN, H. et al. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. **IEEE Transactions on Medical Imaging**, v. 35, n. 5, p. 1285-1298, maio 2016.

SIEGWART, R; NOURBAKHSH, I. R; SCARAMUZZA, D., Introduction Autonomous Mobile Robots, **MIT Press (MA)**, 2nd ed. 453p, 2011.

SOUZA, L. C. G. Application of a multiobjetive optimization pareto approach to design the sdre controller for a rigid-flexible satellite. **Coleção desafios das engenharias: Engenharia de computação**, v. 2021, p. 119-130, 2021.

SU, D. et al. Four-Dimensional Indoor Visible Light Positioning: A Deep-Learning-Based Perspective. **Journal of the Franklin Institute**, v. 360, n. 6, p. 4071–4090, 2023.

SUN, D. et al. Optimized CNNs to Indoor Localization through BLE Sensors Using Improved PSO. **Sensors (Basel, Switzerland)**, v. 21, n. 6, p. 1995, 2021.

SUN, H. et al. YOLO-FMDI: A Lightweight YOLOv8 Focusing on a Multi-Scale Feature Diffusion Interaction Neck for Tomato Pest and Disease Detection. **Electronics**, v. 13, n. 15, p. 2974, 2024.

SUSANTO, et al. Tiny-YOLO distance measurement and object detection coordination system for the BarelangFC robot. **International Journal of Electrical and Computer Engineering (IJECE)**, v. 13, n. 6, p. 6926-6939, 2023.

SVIERCOSKI, R. F**.** Matemática Aplicada às Ciências Agrárias: Análise de Dados e Modelos. **Editora UFV**. 333 p. 2008.

TAROMI, A.D., KLIDBARY, S.H. A novel data-driven algorithm for object detection, tracking, distance estimation, and size measurement in stereo vision systems. **Multimed Tools Appl**, 2024.

TERVEN, J.; CÓRDOVA-ESPARZA, D.-M.; ROMERO-GONZÁLEZ, J.-A. A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS. **Machine Learning and Knowledge Extraction**, v. 5, n. 4, p. 1680-1716, 2023.

THRUN, S.; BURGARD, W.; FOX, D. Probabilistic Robotics. **The MIT Press**, p. 647, 2005.

TRAN, N. K.; KUHLE, L. C.; KLAU, G. W. A Critical Review of Multi-Output Support Vector Regression. **Pattern Recognition Letters**, v. 178, p. 67-78, 2024.

TRAN, K. H. P. Tango Navigator: an indoor navigation system**.** Dissertação. **Politecnico di Milano**, Milão, p. 111, 2017.

TZAFESTAS, S. G. Introduction to Mobile Robot Control (Elsevier Insights). **Elsevier**: 1ª edição, p.750, 2013.

ULTRALYTICS**. YOLOv8-Seg Documentation**. Disponível em: https://docs.ultralytics.com/models/yolov8/#__tabbed_1_3 Acesso em: 18 out. 2024.

UYGUN, T.; OZGUVEN, M. M. Determination of tomato leafminer: Tuta absoluta (Meyrick) (Lepidoptera: Gelechiidae) damage on tomato using deep learning instance segmentation method. **European Food Research and Technology**, v. 250, p. 1837-1852, 2024.

VAJGL M.; HURTIK P.; NEJEZCHLEBA T. Dist-YOLO: Fast Object Detection with Distance Estimation. **Applied Sciences**. v. 12, n. 3, p. 1354, 2022.

WANG, C. Y., et al. CSPNet: A New Backbone That Can Enhance Learning Capability of CNN. 2019.

WANG, G. et al. Fighting against Terrorism: A Real-Time CCTV Autonomous Weapons Detection Based on Improved YOLO v4. **Digital Signal Processing,** vol. 132, p. 103790, 2022.

WU, T.; DONG, Y. YOLO-SE: Improved YOLOv8 for Remote Sensing Object Detection and Recognition. **Applied Sciences**, v. 13, n. 24, p. 12977, 2023.

WU, Z. L. et al. Location estimation via support vector regression. ***IEEE Transactions on Mobile Computing***, v. 6, n. 3, p. 311-321, 2007.

XIE, T. et al. A Deep Feature Aggregation Network for Accurate Indoor Camera Localization. **IEEE Robotics and Automation Letters**, v. 7, n. 2, p. 3687–3694, 2022.

XUE, W. et al. Improved Wi-Fi RSSI Measurement for Indoor Localization. **IEEE Sensors Journal**, v. 17, n. 7, p. 2224–2230, 2017.

YANG, Z. et al. A Survey of Automated Data Augmentation Algorithms for Deep Learning-Based Image Classification Tasks. **Knowledge and Information Systems**, v. 65, n. 7, p. 2805-2861, 2023.

YU, F. et al. 5 G WiFi Signal-Based Indoor Localization System Using Cluster k-Nearest Neighbor Algorithm. **International Journal of Distributed Sensor Networks**, v. 10, n. 12, p. 247525, 2014.

YU, Y. et al. Identifying Irregular Potatoes Using Hausdorff Distance and Intersection over Union. **Sensors (Basel, Switzerland)**, v. 22, n. 15, p. 5740, 2022.

YUE, X. et al. Improved YOLOv8-Seg Network for Instance Segmentation of Healthy and Diseased Tomato Plants in the Growth Stage. **Agriculture**, v. 13, n. 8, p. 1643, 2023.

ZHANG, H. et al. A Scalable Indoor Localization Algorithm Based on Distance Fitting and Fingerprint Mapping in Wi-Fi Environments. **Neural Computing & Applications**, v. 32, n. 9, p. 5131–5145, 2020.

ZOPH, B. et al. Learning Transferable Architectures for Scalable Image Recognition. **IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),** p. 8697-8710, 2018.