Revista de Sistemas de Informação da FSMA

n. 31 (2023) pp. 48-68

SISTEMAS DE INFORMAÇÃO

http://www.fsma.edu.br/si/sistemas.html

TRILHA PRINCIPAL

Dronapple: Um Modelo para Colheita de Maçãs Utilizando Aprendizado de Máquina e Drones

Maxwell F. Barbosa, Universidade do Vale do Rio dos Sinos – Unisinos, Brasil
Gilson A. Helfer, Universidade de Santa Cruz do Sul – UNISC, Brasil
Jorge L. V. Barbosa, Universidade do Vale do Rio dos Sinos – Unisinos, Brasil

***Resumo*— Este artigo apresenta o modelo *Dronapple*, que utiliza um drone equipado com uma câmera e uma ventosa para colher maçãs em pomares de macieira no sistema *Tall Spindle*. O modelo é capaz de reconhecer as maçãs nas árvores, colhê-las com a ventosa e depositá-las em uma despensa, utilizando algoritmos de visão computacional e aprendizado por reforço. A avaliação e treinamento do modelo proposto foram realizados com a implementação de dois ambientes de simulação. Os resultados no simulador mostraram que, em média, foram necessários 5,32 segundos para realizar a colheita da maçã e 17,75 segundos para locomover o drone até a despensa e retornar à árvore, com uma taxa de sucesso de 97,93% na colheita de maçãs maduras em posições não obstruídas. O vídeo[1] que demonstra o funcionamento do modelo no ambiente de simulação desenvolvido e o código-fonte[2] estão disponíveis.**

***Palavras-chave*—Aprendizado por reforço, Drone autônomo, Pomar de macieira, Visão Computacional, Simulador.**

***Abstract*—This article presents the Dronapple model, which uses a drone equipped with a camera and a suction cup for apple orchards in the Tall Spindle system with the ability to recognize apples in trees, harvest them with a suction cup and deposit them in a pantry box, using algorithms of computer vision and reinforcement learning. The validation and training of the proposed model were carried out with the implementation of two simulation environments. The simulator results showed that it took an average of 5.32 seconds to harvest the apple and an average of 17.75 seconds to move the drone to the pantry box and return to the tree with a 97.93% success rate in harvesting ripe apples in unobstructed positions. A video[1] showing the model working in the developed simulator environment and the source code[2] are available.**

***Index Terms*—Apple orchard, Autonomous drone, Computer Vision, Reinforcement learning, Simulator.**

I. INTRODUÇÃO

Com a escassez de mão de obra na colheita de maçãs devido ao crescimento das plantações, muitas maçãs apodrecem nas árvores ou são colhidas após o período ideal de colheita. Além disso, o trabalho é muito exigente, possui baixa remuneração e é temporário. Este cenário tende a piorar devido à migração de pessoas para as cidades em busca de melhores condições [30] [14]. Em alguns casos, os agricultores relatam uma escassez de 30% de coletores, resultando no apodrecimento de plantações e, algumas vezes, no encerramento de suas operações seis semanas antes do esperado [11]. Uma pesquisa realizada em pomares de macieira em Washington estimou que o custo para colher um acre pode chegar a aproximadamente US$ 2.200 [19]. Por ser um trabalho manual e exigente, sem alternativas para automação nas colheitas, os produtores ficam sem opções para suprir a falta de trabalhadores, ocasionando a falta de produtos e preços elevados.

Com as estimativas apontando para um aumento populacional, a necessidade de aumentar a produção de frutas se mostra ainda mais importante. O cultivo de frutas está relacionado a dois objetivos da Organização Mundial da Saúde (OMS): acabar com a fome e propiciar uma vida saudável. Segundo a OMS, é recomendado consumir pelo menos 400 g diárias de frutas e vegetais. Estima-se que em 2017, 3,9 milhões de óbitos em todo o mundo podem ser atribuídos à falta de consumo de frutas e vegetais [44]. Estima-se que a falta desses alimentos tenha ocasionado 14% dos óbitos por câncer gastrointestinal, 11% das doenças cardíacas isquêmicas e cerca de 9% de mortes por derrame cerebral pelo mundo [1].

O Brasil possui um papel importante neste cenário. Segundo o Departamento de Economia Rural do Paraná (DERAL), em 2017 o Brasil foi o terceiro maior produtor de frutas do mundo, sendo a maçã a oitava fruta mais cultivada no país [17]. Segundo o relatório *The State of Food Security and Nutrition in the World* de 2020, estima-se que o crescimento atual na área agrícola não está sendo suficiente e que em 2030 não será possível atingir as metas de fome zero no mundo [41]. Para suprir essa demanda, novas tecnologias e métodos precisam ser desenvolvidos e adotados em todas as etapas dos cultivos, desde o preparo do solo até a colheita. Para isso, muitos processos tendem a ser automatizados para aumentar a eficiência e qualidade.

Os drones, também chamados de Veículos Aéreos Não Tripulados (VANTs), surgiram para atender necessidades militares e, comparados com os modelos atuais, eram pesados e

M. F. Barbosa, Universidade do Vale do Rio dos Sinos, São Leopoldo, Brasil (e-mail: maxwell.fbarbosa@gmail.com).
G. A. Helfer, Universidade de Santa Cruz do Sul, Santa Cruz do Sul, Brasil (e-mail: ghelfer@unisc.br)
J. L. V. Barbosa, Universidade do Vale do Rio dos Sinos, São Leopoldo, Brasil (e-mail: jbarbosa@unisinos.br).

[1] https://youtu.be/YIr9OHIYEUc
[2] https://github.com/MaxwellFB/Dronapple

com alto custo de fabricação. Diferentemente de outros veículos aéreos, os drones não possuem tripulantes e podem ser controlados de longa distância por uma pessoa ou até mesmo ser autônomos, sem a necessidade de interação humana [13]. Devido a estas e outras vantagens, os drones estão sendo adotados em diversas áreas, além do uso militar, como na agricultura e em pomares para automatizar processos, reduzir custos e aumentar o monitoramento das plantações [8].

O aprendizado de máquina trouxe diversas possibilidades de automação, principalmente com a utilização do aprendizado por reforço. Diversos estudos mostram a possibilidade de treinar robôs para realizarem trabalhos repetitivos. Essa tecnologia possibilitou criar drones autônomos mais sofisticados para navegar em locais que antes eram acessados apenas por profissionais ou para realizar a entrega de produtos com maior eficiência, sem a necessidade de um piloto [5]. Um impedimento inicial era a falta de hardware para processar as informações, já que o aprendizado de máquina é computacionalmente custoso, mas nos últimos anos isso vem mudando: o hardware está cada vez mais poderoso, menor e com menor custo de fabricação.

O objetivo deste trabalho é associar tecnologias como drones para realizar a colheita de maçãs, o avanço do poder computacional para processamentos de imagens e o aprendizado de máquina para desenvolver um modelo, com potencial de ajudar a reduzir a escassez de trabalhadores em pomares de macieira no sistema *Tall Spindle*, em que as árvores são plantadas em fileiras e podadas para possuir poucas folhas e galhos, facilitando a colheita. Além disso, o trabalho tem como objetivo permitir a expansão de plantações com maior previsibilidade, reduzindo o risco da falta de trabalhadores em épocas de colheita.

Para tanto, este artigo propõe o modelo *Dronapple*, organizado em 4 componentes principais para realizar a locomoção dos drones pelas árvores, detectar e colher as maçãs não obstruídas e depositá-las em uma despensa, tudo de forma autônoma. Para isso, foram desenvolvidos neste trabalho dois ambientes de simulação para realizar os treinamentos dos algoritmos de aprendizado de máquina e avaliar o modelo proposto, assim como a implementação da simulação de um drone equipado com um braço e uma câmera digital. O trabalho tem como principal contribuição científica a especificação de um modelo aéreo para colheita de maçãs não obstruídas.

Este artigo está organizado em 8 seções. A seção II apresenta o referencial teórico encontrado na literatura, que contribuiu para a elaboração do presente trabalho. Na seção III, são elencados os trabalhos relacionados na literatura e comerciais, os critérios de escolha dos mesmos e uma análise comparativa entre os trabalhos analisados e o modelo *Dronapple*. A seção IV apresenta a arquitetura e o funcionamento do modelo proposto. A seção V descreve os aspectos de implementação dos ambientes de simulação e dos componentes do modelo. A seção VI descreve como foram realizados os treinamentos e as avaliações, apresentando os resultados na seção VII. Por fim, a seção VIII apresenta as conclusões e os trabalhos futuros originados deste estudo.

II. REFERENCIAL TEÓRICO

Para o desenvolvimento do drone autônomo para colher maçãs, é necessário analisar e detalhar conceitos e ferramentas essenciais. Nesta seção, serão descritos os principais conceitos e ferramentas utilizados durante o desenvolvimento do presente trabalho.

A. Aprendizado Profundo

Há muitos anos, estudos têm sido realizados sobre o funcionamento do cérebro humano e como replicá-lo em máquinas. Em 1943, o neurofisiologista Warren McCulloch e o matemático Walter Pitts publicaram um artigo sobre o funcionamento dos neurônios e elaboraram um modelo computacional para redes neurais baseado em matemática, intitulado de lógica de limiar (*Threshold Logic*) [29].

Em 1958, Rosenblatt apresentou um modelo chamado Perceptron, com capacidade de reconhecer padrões por meio de dados de entrada [34]. Alguns anos depois, Rumelhart et al. trouxeram para o segmento o *backpropagation*, que possibilitou expandir o *Perceptron* para diversas camadas de neurônios totalmente conectadas e permitiu realizar o treinamento de redes maiores, mantendo tempo computacional viável para resolver problemas mais complexos. Segundo os autores, o processo ajusta constantemente os pesos de cada um dos neurônios, buscando representar características relevantes, minimizando a diferença entre o resultado da rede neural e o esperado [35].

As redes com múltiplas camadas costumam possuir uma função de ativação para ajustar a saída do valor do neurônio antes de entrar no próximo. De acordo com Li et al., as funções de ativação têm como principal objetivo adicionar não-linearidade à saída de cada neurônio. Caso não seja utilizado uma função de ativação não linear, não importará quantas camadas a rede possuir, ela não será capaz de aprender problemas não lineares, assim como o *Perceptron*, que não consegue resolver problemas complexos. Teoricamente, uma rede com função de ativação não linear consegue se aproximar de qualquer função objetivo, aumentando a habilidade da rede neural de aprender com os dados [26].

Embora as redes neurais sejam capazes de aprender utilizando dados vetoriais, elas não conseguem apresentar os mesmos benefícios para dados matriciais, como imagens, principalmente devido à alta necessidade de poder computacional. Para resolver esse problema, LeCun et al. publicaram um artigo em 1998 mostrando os benefícios da utilização da *Convolutional Neural Network* (CNN), em conjunto com múltiplas camadas de redes neurais totalmente conectadas e *backpropagation*, quando aplicados a muitas imagens de exemplo. No estudo, constataram que a CNN consegue detectar padrões relevantes em imagens, ignorando características irrelevantes, além de possuir uma quantidade menor de pesos (parâmetros), consequentemente consumindo menos memória [25].

B. Aprendizado por Reforço

O aprendizado por reforço é uma categoria diferente de aprendizado, que possui um agente — como um robô, um carro ou um drone. Sua principal característica é a falta de uma base de dados formada, pois o agente interage em tempo real com o ambiente, observa as alterações e recebe pontuações pré-definidas para saber se a ação realizada é considerada positiva e, como tal, deve continuar sendo realizada. O agente fica neste ciclo, aprendendo e se adaptando por tentativas e erros, de acordo com a pontuação recebida. Segundo [4], a união do aprendizado profundo com o sistema de interações do aprendizado por reforço permitiu o desenvolvimento de agentes com capacidade de se adaptarem ao ambiente, inclusive utilizando imagens como entrada para o sistema, simulando a visão humana. O computador é capaz de aprender a realizar tarefas complexas, como jogar videogame e fazer robôs interagirem com objetos de forma autônoma, tudo isso utilizando uma câmera para captar imagens do cenário.

Pode-se considerar o aprendizado profundo como o cérebro do agente e o aprendizado por reforço o método de aprendizado, também chamado de política. Diversas políticas foram propostas, com benefícios diferentes, sendo uma delas a *Proximal Policy Optimization* (PPO) [37], que mostrou conseguir ensinar robôs a aprenderem a realizar atividades com ações contínuas, incluindo aprender a nadar, pular e caminhar. Outra política é a *Deep Deterministic Policy Gradient* (DDPG) [27], que utiliza a união de diversas técnicas desenvolvidas na área em busca de resolver problemas mais complexos e atingir resultados melhores, sendo um deles a possibilidade de ensinar o agente a dirigir um carro utilizando imagens como entrada.

C. Drones aplicados em pomares

Com o avanço das tecnologias de drones, que também podem ser chamados VANT (Veículo Aéreo Não Tripulado) ou VARP (Veículo Aéreo Remotamente Pilotado), de acordo com tipo de veículo considerado, esses equipamentos têm sido aplicados em diversos campos, incluindo pomares. No mapeamento sistemático realizado por Barbosa et al. [8], foram exploradas publicações que uniram as vantagens de aprendizado de máquina e drones para auxiliar em tarefas em pomares de laranjeira e macieira. Embora o mapeamento tenha se concentrado apenas em pomares de duas frutas, já é possível identificar o crescimento da utilização dessas tecnologias na área.

Segundo o estudo, entre os anos de 2010 e 2021, foram realizadas 12 publicações que atenderam aos critérios de pesquisa do artigo. O objetivo mais pesquisado foi o mapeamento das árvores, o que possibilitou ter um mapa completo do pomar, sabendo a localização de cada uma das árvores. Outro ponto relevante foi o equipamento mais utilizado, a câmera RGB, que teve destaque, mostrando que, por mais que seja um equipamento simples, ele pode ser utilizado para realizar as tarefas de aprendizado de máquina propostas pelas publicações.

No mapeamento sistemático foram identificados dois artigos referentes à detecção de frutas, utilizando algoritmos de detecção de objetos, como *Faster R-CNN* [33] e *YOLOv4* [9], que mostraram a capacidade desses algoritmos em realizar detecções precisas de frutas e flores de frutas utilizando imagens de câmeras RGB equipadas em drones. Além disso, esses artigos também mostraram que os drones não apresentam riscos para as frutas e árvores devido ao vento gerado durante as operações de curta distância.

D. Simulador de drones

O aprendizado por reforço é uma atividade que pode exigir relevante esforço computacional para realizar seu treinamento, já que seu aprendizado consiste em realizar a atividade diversas vezes. Ademais, a necessidade de aprender por erros, muitas vezes pode danificar o equipamento devido a um movimento incorreto. Para resolver isso, os treinamentos costumam ocorrer em simuladores e após o treinamento estar completo, é transferido para o mundo real sendo efetuados pequenos ajustes, caso seja necessário.

O *AirSim* [38] é um simulador de código aberto projetado para simular carros e drones a fim de treinar algoritmos de aprendizado por reforço, visando alta fidelidade das imagens, simulação da gravidade, arrasto e atrito. No caso de drones, é utilizado um quadricóptero equipado com diversos sensores, incluindo um Sistema de Posicionamento Global (GPS) e câmeras posicionadas em três regiões diferentes do drone. Para a construção dos ambientes, o simulador *AirSim* tem um plugin para *Unreal Engine*[3] e sua interface de programação de aplicação (API) com *Unreal Engine* permite não apenas a comunicação com o drone, mas também a manipulação de objetos e cenários para gerar ambientes dinâmicos em cada época de treinamento.

III. TRABALHOS RELACIONADOS

Nesta seção, serão explorados trabalhos que aplicam técnicas autônomas em pomares de macieira, buscando entender quais metodologias, tecnologias e ferramentas apresentaram resultados promissores.

A seção está organizada da seguinte forma: a subseção III.A apresenta os critérios para a escolha dos trabalhos analisados; as subseções III.B e III.C analisam, respectivamente, trabalhos com sistemas autônomos para a colheita de maçãs encontrados na literatura e em contextos comerciais; e, finalmente, a subseção III.D apresenta as diferenças entre o trabalho proposto e os trabalhos relacionados.

A. Critérios para a escolha dos trabalhos

Para explorar os trabalhos realizados na área de interesse, utilizou-se a ferramenta de pesquisa de trabalhos acadêmicos Google Acadêmica por ser possível pesquisar em diversas bases de dados. Os trabalhos encontrados foram priorizados pela relevância apresentada pela ferramenta e pela data de publicação. Os termos utilizados foram: "*automation agriculture deep learning*", "*harvesting robots*", "*drone deep reinforcement learning*" e "*fruit picking*".

Na pesquisa, foram selecionados os seguintes mapeamentos:

[3] https://www.unrealengine.com/en-US

Saleem et al. [36], para explorar a utilização de técnicas de aprendizado de máquina e aprendizado profundo na agricultura; [6], para observar trabalhos com automação de coletores nas mais diversas áreas do plantio; [5], para identificar quais são as técnicas de aprendizado de máquina utilizadas em drones autônomos em diversos cenários de atuação; [40], para entender as técnicas e estratégias utilizadas na automação de robôs para detectarem e coletarem frutas, tanto na literatura quanto em aplicações comerciais; [10], para expandir o conhecimento sobre estratégias que estão sendo utilizadas comercialmente em robôs para coletar maçãs.

Além dos mapeamentos pesquisados, foi utilizado o mapeamento realizado pelos autores deste artigo para conhecer diversos estudos semelhantes que utilizaram aprendizado de máquina e drones para auxiliar nas tarefas em pomares de macieira e laranjeira [8]. Também foram realizadas pesquisas adicionais na ferramenta Google Acadêmico com o objetivo de analisar trabalhos mais relacionados à área de interesse e considerados relevantes pela ferramenta. Utilizaram-se as seguintes frases de pesquisa: "*Apple harvesting"*, "*Autonomous drone"*, "*UAV reinforcement learning continuous actions*", "*picking robot"* e "*UAV detection fruits"*.

Como critério de escolha, considerou-se a utilização de qualquer tipo de sistema autônomo para a colheita de maçãs. Em casos em que os mesmos autores tenham publicado mais de um artigo, referenciou-se somente o trabalho mais recente.

B. Coletores de maçãs na literatura

Baeten et al. [7] realizaram a montagem de um robô coletor vinculado a um trator para realizar a movimentação e fornecer energia para os componentes. Seu sistema de colheita é realizado por um braço robótico com a ponta equipada com um aspirador para pegar a maçã da árvore e uma câmera instalada na abertura do coletor. Para realizar a localização das maçãs, basearam-se no reconhecimento de objetos arredondados com tamanhos pré-configurados e a aproximação do braço é realizada utilizando várias imagens para calcular, por triangulação, a distância até a maçã. A colheita de cada maçã levou de 8 a 10 segundos. Foi possível colher 80% das frutas com diâmetro entre 6 e 11 cm no campo de visão. Considerando todas as maçãs disponíveis, 70% das frutas estavam no espaço no qual o robô conseguia coletar, o restante estava muito alto.

De-An et al.[16] trabalharam com um robô equipado com um braço robótico para realizar a colheita de maçãs. Sua garra é equipada com diversos sensores com um sistema de corte para conseguir identificar quando a maçã é agarrada e cortar o talo. Para detectar as frutas foi efetuado um pré-processamento, aplicando filtro mediano vetorial para remover ruídos e destacar a maçã, mantendo a qualidade das bordas. Em seguida, foram aplicadas diversas técnicas para extrair as cores e formas para, após, passarem para um *Support Vector Machine* (SVM) e *Radial Basis Function* (RBF). Para aproximar o braço foram utilizadas as coordenadas da localização da fruta e, nos testes realizados no campo durante 10 minutos, 77% das frutas detectadas foram colhidas com sucesso, com tempo médio de 15,4 segundos para colher cada maçã.

Ahlin et al. [2] propuseram um algoritmo para realizar a movimentação de um braço robótico para coletar maçãs, inclusive aquelas que estão obstruídas por galhos e folhas. As avaliações foram realizadas em um simulador, e para detectar as frutas, foi utilizado um limiar na intensidade das cores, e com a localização da maçã, o braço foi movimentado. Nas avaliações realizadas, 92% das maçãs foram detectadas e colhidas com sucesso, mas não foi informado o tempo de colheita.

Silwal et al. [39] desenvolveram um robô para realizar a colheita de maçãs, utilizando um braço robótico e uma garra. O sistema de localização das maçãs consistiu na utilização de dois algoritmos. Como o carrinho de movimentação era estático, o sistema processou a imagem somente uma vez para localizar as frutas, e quando coletou todas as maçãs detectadas, o carrinho foi movido manualmente para uma nova região e o processo foi repetido. A câmera é posicionada atrás do braço robótico e calibrada de forma que seja possível calcular os eixos para se aproximar e colher a maçã com o braço. Durante as avaliações, algumas maçãs foram removidas para evitar problemas na colheita, devido a limitação do robô para alcançá-las e as colheitas foram realizadas na segunda e terceira linhas verticais das árvores (meio da árvore), onde o sistema conseguiu colher com sucesso 84,6% das tentativas, enquanto 13% não foram possíveis devido a colisões entre os dedos da garra e outras frutas. Para colher a primeira maçã, levou-se em média 6,2 segundos, e 5,8 segundos para as próximas frutas do ciclo, além de 6 segundos iniciais para cada ciclo de identificação e localização dos frutos, resultando em uma média de 4 maçãs colhidas por ciclo.

Em um trabalho realizado por Hohimer et al., uma garra robótica impressa em 3D foi desenvolvida e dois algoritmos foram utilizados para detectar maçãs e movimentar um braço robótico em resposta às coordenadas da localização da fruta. Algumas maçãs obstruídas por galhos foram removidas em um pomar e, dentre as tentativas realizadas com as frutas restantes, 67% foram bem-sucedidas, com um tempo médio de 7,3 segundos por maçã [20].

No trabalho de Kang et al., foi apresentado um modelo de detecção e segmentação de maçãs e galhos, no qual a localização da fruta detectada foi utilizada para mover o braço robótico. As avaliações foram realizadas em laboratório, e foram colhidas de 81% a 91% das maçãs, dependendo da complexidade da localização das frutas, com um tempo médio de 7 segundos para colheita [22].

Zhang et al. [46] utilizaram o algoritmo *Mask R-CNN* para detectar as maçãs com o auxílio de uma câmera RGB-D. Com as coordenadas da localização da fruta detectada, foi movimentado um braço robótico para realizar a colheita de maçãs com um sistema de sucção semelhante a uma ventosa que segura a fruta até ser solta em uma despensa. Os autores detectaram alguns casos que o sistema de sucção conseguiu segurar corretamente a maçã, mas não foi possível desvinculá-la da árvore devido as maçãs estarem em galhos longos e

flexíveis e, dificuldades para o sistema de localização das maçãs localizar as frutas parcialmente obstruídas. Nas avaliações realizadas em um pomar, das 41 tentativas, 64,06% das maçãs foram colhidas com sucesso em um tempo médio de 8,8 segundos por fruta.

Hu et al. [21] utilizaram uma câmera binocular para captar as imagens e conseguir estimar a distância entre a câmera e a fruta com maior precisão em comparação com uma câmera RGB, para então, detectar as maçãs utilizado um limiar na intensidade das cores e para movimentar o braço equipado com uma ventosa é utilizado as coordenadas da localização da fruta. Em avaliações realizadas em um pomar, 47,37% das maçãs foram colhidas com sucesso, com um tempo de 4 segundos por colheita.

Yu et al. [45] desenvolveram um robô humanoide, o qual foi aprimorado por Kang et al. [23]. Ele utiliza uma câmera binocular e para fazer a detecção das maçãs são utilizados filtros de pontos e o algoritmo *SIFT*. A movimentação é realizada baseada nas localizações das frutas. Nas avaliações realizadas em laboratório, o robô conseguiu colher 89,2% das maçãs, levando em média 12,3 segundos por maçã.

Wang et al. [42] desenvolveram uma garra com quatro dedos flexíveis e uma ventosa no centro para segurar a maçã. Para realizar a detecção das frutas, foi utilizado segmentação com sensor *LiDAR*, que é um dos métodos com maior precisão para calcular a distância e, com estes valores, realizar a colheita. Das tentativas de colheita, 73,33% foram bem-sucedidas, com tempo médio de colheita de 14,69 segundos.

C. Coletores de maçã comerciais e protótipos

Bogue [10] analisou diversas empresas com soluções para colheita autônoma, seja em desenvolvimento ou já comercializadas. Uma delas é a empresa *FFRobotics*[4] que desenvolveu uma máquina para colheita de maçãs, equipada com 12 braços robóticos com garras, 6 em cada lado da máquina. Sua movimentação é feita por um veículo onde possui todo maquinário embutido e requer um operador. São utilizadas câmeras para identificar as frutas e caso estejam maduras, realizar a colheita. A empresa acredita que será possível colher 90% das maçãs com uma taxa de danos durante a colheita de 3% a 5%. Para o funcionamento adequado da máquina, é necessário ter um distanciamento adequado entre as linhas de plantações das árvores, caso contrário, não é possível entrar com a máquina ou colher ambos os lados simultaneamente, limitando o funcionamento em 50% da capacidade.

Abundant Robots[5] também é uma empresa que possui uma máquina para colheita de maçãs que consiste em um trator adaptado para carregar o maquinário, equipado com um *LiDAR* para se locomover pela plantação. Sua operação necessita de um supervisor, a localização das maçãs é feita por algoritmos de detecção de objetos e a colheita, por um robô com sucção para coletar a maçã, a empresa planeja coletar 1 maçã por segundo. Ao contrário da *FFRobotics*, esta máquina colhe somente de um lado, não sendo necessário ter um determinado distanciamento entre as linhas de plantações das árvores, apenas o suficiente para a máquina entrar.

A *Tevel Aerobotics Technologies*[6] é uma empresa que focou em um sistema diferente de colheita de frutas. Esta empresa planeja realizar colheita de abacates, mangas, laranjas e maçãs utilizando drones. Seus drones são ligados via cabo por um veículo que também é usado para armazenar as frutas colhidas. Possui a capacidade de realizar a colheita de frutas durante o dia e a noite e, para seu sistema de visão, é utilizado uma câmera frontal para localizar as frutas maduras e uma inferior para realizar a leitura de diversos *QR Code* que estão na plataforma onde deve ser depositado a maçã colhida. A coleta inicialmente era realizada por uma garra com sistema de rotação para soltar a fruta da árvore, mas recentemente a empresa substituiu a garra por uma ventosa.

D. Diferencial do trabalho proposto

O Quadro I apresenta uma comparação dos trabalhos de colheita autônoma de maçãs analisados na literatura. Todos os trabalhos exploraram robôs terrestres, o que limita a altura que o braço pode atingir para realizar a colheita. Para detecção das maçãs, um dos trabalhos utilizou modelos de aprendizado de máquina (AM), três utilizaram aprendizado profundo (AP) e os outros trabalhos utilizaram modelos matemáticos sem aprendizado, o que em alguns casos dificultou a adaptação do modelo a mudanças de iluminação. Todos os trabalhos realizaram cálculos e lógicas baseados na localização das maçãs detectadas para movimentar o robô e realizar a colheita da fruta. Grande parte dos trabalhos utilizou uma garra para a colheita, um deles utilizou um cano com sucção, dois trabalhos utilizaram ventosas e um utilizou uma combinação de ventosa e garra.

O modelo *Dronapple* tem como objetivo solucionar o problema da limitação de altura para a colheita de maçãs, utilizando um drone. Para isso, emprega técnicas avançadas, como algoritmos de aprendizado profundo (AP), que podem se adaptar a mudanças de iluminação e cenários, para otimizar a detecção das maçãs em tempo real. Além disso, o aprendizado por reforço (AR) é utilizado para treinar o drone a se movimentar em direção à maçã e coletá-la de forma eficiente, identificando situações em que pode se mover mais rapidamente sem perder a estabilidade dos movimentos.

Não foi possível compreender o funcionamento completo dos produtos das empresas, pois não foram encontradas informações suficientes nos sites dos fabricantes. Por serem produtos comerciais, é possível que a liberação de certas informações possa comprometer a competitividade da empresa. No entanto, é possível concluir que dois deles são terrestres, sendo que um deles possui limitação devido à distância entre as linhas de plantação das árvores, pois caso sejam muito próximas, o robô não terá espaço para operar, e se estiverem muito distantes, só será possível utilizar metade da capacidade de colheita da máquina. O terceiro modelo analisado é aéreo e utiliza drones, assim como o modelo proposto neste trabalho.

[4] https://www.ffrobotics.com

[5] https://waxinvest.com/projects/abundant-robots

[6] https://www.tevel-tech.com

Quadro I: Comparação entre os trabalhos relacionados na literatura e o modelo proposto

Referência	Ambiente	Detector	Navegador	Coletor
[7]	Terrestre	Matemático	Coordenadas	Sucção
[16]	Terrestre	AM	Coordenadas	Garra
[2]	Terrestre	Matemático	Coordenadas	Garra
[39]	Terrestre	Matemático	Coordenadas	Garra
[20]	Terrestre	Matemático	Coordenadas	Garra
[22]	Terrestre	AP	Coordenadas	Garra
[46]	Terrestre	AP	Coordenadas	Ventosa
[21]	Terrestre	Matemático	Coordenadas	Ventosa
[23]	Terrestre	Matemático	Coordenadas	Garra
[42]	Terrestre	AP	Coordenadas	Garra e ventosa
Dronapple	Aéreo	AP	AR	Ventosa

Infelizmente, há poucas informações disponíveis sobre seu funcionamento completo, e outra desvantagem é a necessidade de utilizar múltiplas câmeras, o que eleva o custo do coletor, além de necessitar de um veículo adaptado com diversos tipos de *QR Code*, onde o drone realiza sua leitura para localizar a despensa e depositar as maçãs colhidas.

O modelo *Dronapple* possui o código-fonte aberto, ampliando a divulgação de conteúdo para o desenvolvimento de coletores de maçãs e outras frutas utilizando drones. O modelo tem como objetivo resolver o problema da falta de trabalhadores na colheita de maçãs, permitindo uma previsibilidade maior para o crescimento das plantações, diminuindo a necessidade de trabalhadores em épocas específicas do ano e garantindo que os drones autônomos estejam disponíveis para realizar as colheitas. A principal contribuição deste trabalho é a especificação de um modelo aéreo para realizar a colheita de maçãs utilizando algoritmos de aprendizado de máquina e suas características são descritas a seguir.

IV. Modelo Dronapple

Esta seção descreve o modelo *Dronapple*, sendo apresentado na subseção IV.A uma visão geral do funcionamento do modelo e na subseção IV.B são detalhados sua arquitetura e os seus quatro componentes principais.

A. Visão geral

O modelo *Dronapple* apresenta uma proposta para realizar a colheita de maçãs diretamente das árvores em pomares de macieira no sistema *Tall Spindle*, caracterizado por árvores plantadas em fileiras e podadas de forma que as maçãs cresçam em posições mais acessíveis para a colheita. Para isso, utiliza-se um drone equipado com câmera para observar as árvores e maçãs, além de um braço equipado com uma ventosa com sistema de rotação para soltar a maçã da árvore e levá-la até uma despensa próxima ao drone, onde será depositada.

O *Dronapple* foi projetado para localizar as maçãs e identificar o estado da fruta, verificando se estão maduras o suficiente para serem colhidas e se estão obstruídas, o que impossibilita a colheita. Dessa forma, evita-se a colheita precoce de maçãs e também que o drone colida com galhos e folhas, o que pode danificar tanto a árvore quanto o próprio drone. Para realizar essa tarefa, é utilizada uma arquitetura de visão computacional para detectar e classificar as frutas por meio da câmera instalada no drone.

A arquitetura do modelo *Dronapple* é composta por quatro componentes principais. O Componente 1 realiza a navegação inicial do drone até uma determinada região na árvore utilizando um sistema de posicionamento global (GPS). Em seguida, o Componente 2 captura uma imagem utilizando a câmera e, com algoritmos de visão computacional, realiza a detecção e classificação das maçãs, escolhendo uma delas para ser colhida. Com a localização da maçã que deverá ser colhida, o Componente 3 realiza a movimentação do drone até a fruta para colhê-la e, por fim, o Componente 4 movimenta o drone até a despensa para depositar a maçã.

Essa divisão dos componentes permite o gerenciamento do sistema de forma modular, possibilitando a avaliação e melhoria de cada parte de forma isolada. Além disso, permite a alteração das estratégias sem afetar significativamente o restante do sistema, exceto pela detecção e localização das maçãs presente no Componente 2, já que as coordenadas da maçã obtidas por este componente são utilizadas pelo Componente 3 para movimentar o drone até a fruta.

Na criação do modelo, foram utilizados algoritmos com capacidade de serem executados em tempo real, mesmo em configurações de computadores com poucos recursos computacionais, o que possibilita o uso de microcomputadores no drone para realizar todo o processamento necessário. Todos os algoritmos e configurações serão explicados na próxima seção.

O fluxo do modelo *Dronapple* concentra-se principalmente nos Componentes 2 e 3, destacados na região em verde na Figura 1. Inicialmente, o Componente 1 posiciona o drone na primeira região configurada da árvore. O Componente 2, em seguida, localiza e atualiza a localização da maçã a cada passo que o Componente 3 realizar, buscando rastrear a mesma maçã até o momento em que a fruta for coletada. Caso o Componente 2 não consiga localizar pelo menos uma maçã que seja possível colher, é retornado para o Componente 1 para

movimentar o drone para uma nova região na árvore. Se o Componente 3 conseguir coletar a maçã com a ventosa, será passado para o Componente 4 para realizar o deslocamento do drone até a despensa e soltar a fruta. Com a maçã depositada, o Componente 1 move o drone para a região onde a colheita da fruta foi realizada, para buscar uma nova maçã para ser colhida.

B. Arquitetura Dronapple

A Figura 2 apresenta a arquitetura do modelo, composta por quatro componentes: Aproximar Árvore, Detector, Coletor e Largar Maçã. As definições de cada componente são detalhadas nas subseções a seguir.

Componente 1 - Aproximar Árvore

O Componente 1 tem como objetivo aproximar o drone da árvore em uma distância de 70 a 90 cm, permitindo uma visão significativa da região, e mantendo uma distância segura tanto da árvore da frente quanto da de trás. Essa distância foi calculada por meio de testes realizados em simuladores, que serão mencionados na seção de Implementação.

O *Dronapple* pressupõe que já foi realizado um mapeamento do pomar, semelhante aos trabalhos de Ampatzidis et al. [3] e Osco et al. [31], para saber a localização das árvores. Essas informações são armazenadas em uma base de dados, à qual o modelo *Dronapple* tem acesso, permitindo que o drone se movimente linearmente através do uso de GPS até as regiões desejadas.

Após a colheita de uma fruta, a posição atual do drone é armazenada para que ele possa retornar à mesma localização após ter realizado o depósito da maçã na despensa. Este componente também é responsável por movimentar o drone para a próxima região, caso o Componente 2 não consiga localizar nenhuma fruta com a possibilidade de ser colhida, podendo ser em sentido vertical ou horizontal.

Componente 2 – Detector

O Componente 2 utiliza as imagens capturadas em tempo real pelo drone e as processa utilizando arquitetura de visão computacional para localizar todas as maçãs visíveis, classificando-as quanto às condições para serem colhidas. Com as maçãs detectadas, é utilizado um algoritmo de rastreamento baseado nas coordenadas de localização das maçãs para decidir qual delas, dentre todas as classificadas como possíveis de serem colhidas, será priorizada para que suas coordenadas sejam passadas para o Componente 3.

Devido à comunicação direta entre as informações geradas por este componente e o próximo, ele é o único componente que não pode ser alterado livremente. Deve-se tomar os devidos cuidados para evitar danificar o funcionamento do componente seguinte, pois os resultados dos cálculos das coordenadas de localização da maçã são utilizados pelo Componente 3.

Componente 3 – Coletor

O Componente 3 recebe as coordenadas da localização da maçã, enviadas pelo componente anterior, e, com base nessas coordenadas, utiliza uma rede neural para mover o drone até que a ventosa esteja encostada na maçã. Para determinar o momento em que o sistema de sucção da ventosa deve ser acionado, é utilizado outro algoritmo que leva em consideração a localização da maçã e a velocidade do drone. Quando a maçã está fixada na ventosa, um sistema de rotação localizado na região da ventosa, com um ângulo mínimo de 54º no sentido anti-horário, é ativado [15], a fim de rotacionar a fruta e soltá-la da árvore. Após soltar a maçã da árvore, o drone realiza um movimento para trás, de aproximadamente 50 cm[7], o que é suficiente para se afastar a uma distância relativamente grande da árvore, diminuindo as chances de colisão e, em seguida, passar para o próximo componente.

Componente 4 - Largar Maçã

O Componente 4 tem como objetivo levar o drone até a despensa mais próxima para depositar a maçã colhida. Para isso, a despensa é equipada com um sensor de GPS que fornece sua localização exata. Todas as informações de localização das despensas podem ser armazenadas na mesma base de dados mencionada anteriormente, que também armazena a localização das árvores. Caso a despensa se mova conforme o drone se move para as próximas árvores, as informações de localização devem ser atualizadas na base de dados. No entanto, não será abordado neste trabalho como movimentar a despensa, assumindo que algum componente terceiro, como o *Thovald plataform*[8], esteja sendo utilizado.

Utilizando as coordenadas da despensa e a localização do GPS do drone, este componente, primeiramente, verifica se o drone está em uma posição mais alta do que a despensa. Caso contrário, realiza uma movimentação do drone para cima, a fim de evitar colidir com as laterais da despensa. Em seguida, o componente movimenta o drone linearmente até o centro da despensa e libera a maçã da ventosa para concluir a coleta de uma maçã.

V. Aspectos de Implementação

Para fins de avaliação do modelo proposto, foi implementado um protótipo em ambientes de simulação. Para isso, foi utilizada a linguagem de programação *Python* para construir a arquitetura do modelo e se comunicar com os ambientes de simulação.

A. Simulador

Na implementação do protótipo, foram desenvolvidos dois ambientes de simulação para treinamentos e validações do modelo proposto. Devido ao uso de uma câmera para realizar a visão do cenário, optou-se pelo simulador *Unreal Engine*, que apresenta flexibilidade para o desenvolvimento do ambiente necessário e gráficos semelhantes aos do mundo real. Todos os objetos no ambiente de simulação estão disponíveis dentro da *Unreal Engine* ou gratuitamente em sua loja, sendo eles:

[7] Se a maçã estiver mais para dentro, ela provavelmente estará obstruída e não será marcada para colheita pelo Componente 2.

[8] https://sagarobotics.com/thorvald-platform

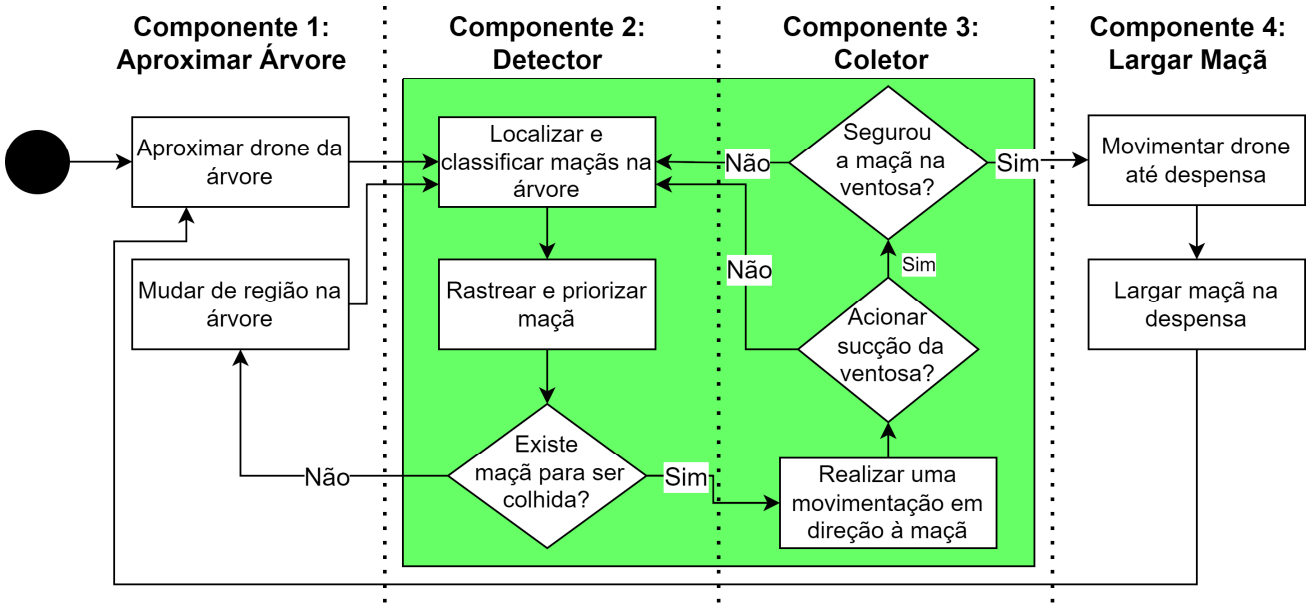

Fig. 1. Fluxo do Dronapple

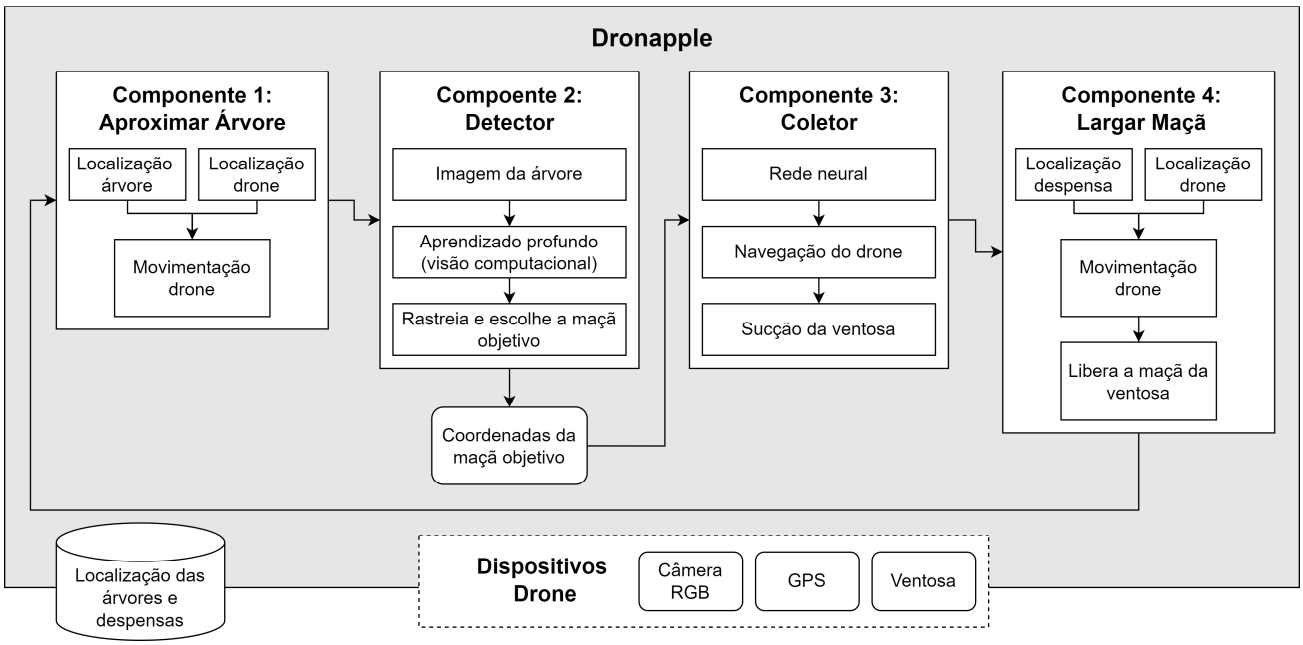

Fig. 2. Arquitetura do Dronapple

"*Megascans - Definitive Fruits*"[9] para fazer as maçãs, "*Megascans Trees: European Black Alder (early access)*"[10] para fazer a árvore e "*Temperate Vegetation: Optimized Grass Library*"[11] para fazer o chão.

O primeiro ambiente, denominado Ambiente 1, foi desenvolvido principalmente para realizar o treinamento do Componente 3. Neste ambiente, foram adicionados dois cenários. O primeiro, representado na Figura 3, apresenta a visão da câmera do drone na parte inferior direita, com uma maçã em um fundo estático e liso, a fim de diminuir a complexidade na detecção da fruta pelo Componente 2. O segundo cenário, representado na Figura 4, foi desenvolvido para avaliar o funcionamento do rastreamento da maçã que será colhida. Este cenário também possui o mesmo fundo estático e liso, mas com nove maçãs próximas uma da outra, possuindo três diferentes distanciamentos entre elas: laterais de 0,14 cm, 1,14 cm e 1,64 cm e superiores de 0,2 cm, 0,7 cm e 1,7 cm, respectivamente.

[9] https://www.unrealengine.com/marketplace/en-US/product/fa6a85e22fd24701b6d158ca08801598

[10] https://www.unrealengine.com/marketplace/en-US/product/megascans-trees-european-black-alder-early-access

[11] https://www.unrealengine.com/marketplace/en-US/product/dynamic-optimized-grass-library

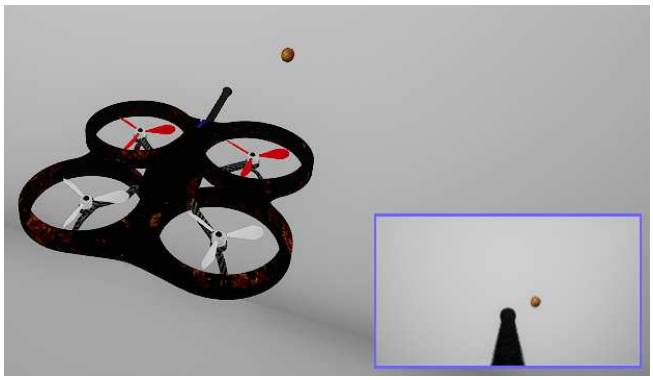

Fig. 3. Simulação Ambiente 1 - Com 1 Maçã

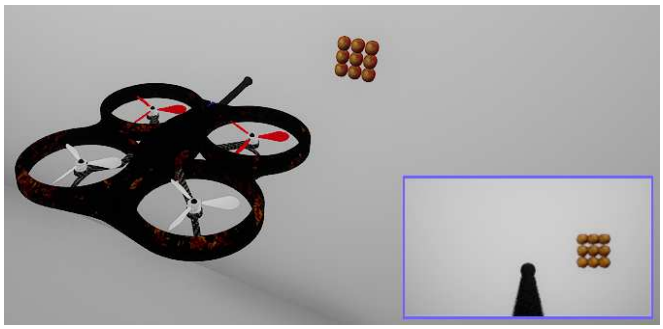

Fig. 4. Simulação Ambiente 1 - Com 9 Maçãs

Apesar da falta de realismo inerente a este tipo de cenário, eles são um passo necessário no desenvolvimento e no treinamento do Componente 3 e ambos serão substituídos por versões mais realistas em um momento posterior.

No segundo ambiente, denominado de Ambiente 2, foi construído um terreno ao ar livre com uma árvore. Devido à falta de macieiras semelhantes às do sistema *Tall Spindle* na loja da *Unreal Engine*, foi utilizado outro tipo de árvore que se assemelha ao desejado.

Com a finalidade de atender às necessidades de ambientes dinâmicos para os treinamentos de um modelo de aprendizado de máquina, cada um dos ambientes é alterado automaticamente no início de cada época. No Ambiente 1, a posição da maçã é alterada, podendo estar a uma distância entre 80 cm e 120 cm do drone e em qualquer parte do campo de visão da câmera, com exceção das bordas e abaixo do braço.

No Ambiente 2, foram implementadas 100 posições diferentes onde maçãs com possibilidade de serem colhidas podem aparecer e outras 101 posições com maçãs obstruídas. Estas posições podem ser observadas na Figura 5 com as maçãs vermelhas representando as posições de maçãs possíveis de serem colhidas e as laranjas representando maçãs obstruídas. Algumas não estão visíveis por estarem atrás das maçãs vermelhas. Ao iniciar cada época de treinamento ou de teste, cada uma das maçãs tem 40% de chances de aparecer no cenário. As que aparecerem possuem 60% de chances de estarem maduras e 40% de estarem verdes. Todas as maçãs maduras possuem o mesmo tamanho, 7,2 cm de diâmetro e 7,6 cm de altura. Devido às configurações do pacote de frutas utilizado, as maçãs verdes possuem tamanhos diferentes das maduras, tendo 8,3 cm de diâmetro e 8 cm de altura. Foi mantido os tamanhos mencionados devido as configurações padrões dos pacotes utilizados e por mais que os tamanhos não estejam representando completamente os tamanhos de maçãs reais, será suficiente para validar se o Componente 2 será capaz de localizar as maçãs.

Fig. 5. Simulação Ambiente 2 - Posições Maçãs

A Figura 6 apresenta uma visão geral do Ambiente 2, contendo as maçãs vermelhas e verdes conforme as probabilidades de aparecerem. Na parte inferior esquerda encontra-se a despensa onde são depositadas as maçãs colhidas e na parte inferior direita, a visão da câmera do drone de uma região da árvore.

Fig. 6. Simulação Ambiente 2 - Visão Geral

A simulação do drone nos ambientes desenvolvidos foi realizada utilizando o *plugin Airsim [38]*, por possuir configurações e física semelhantes às de um drone real, o que permitiu realizar movimentações realistas e customizações, como adicionar um braço com a ventosa ligada ao drone. Como o drone do *Airsim* foi desenvolvido para manter a fidelidade de um drone real, não foi realizada nenhuma alteração nas configurações padrão disponíveis na versão 1.8.1 para *Windows*, no repositório oficial do projeto no *GitHub*. Para todos os experimentos, foi utilizada a câmera RGB central disponível, com resolução de captura de 256×144 pixels.

O desenvolvimento do braço e da ventosa foi inspirado no trabalho de Liu et al. [28], que desenvolveu uma ventosa para ser utilizada em drones, assim como na empresa *Tevel Aerobotics Technologies*. Para isso, foi utilizado o programa de modelagem *3D Blender*[12] e exportado para *Unreal Engine*, onde foi vinculado na parte central da frente do drone, com um comprimento total de 41 cm para evitar que o drone colida com a árvore quando for coletar uma maçã.

Não foi possível simular a resistência da maçã na árvore, por isso não foi implementado o sistema de rotação da ventosa ao segurar a maçã. Dessa forma, considerou-se uma colheita bem-sucedida no momento em que a ventosa encosta na maçã e o sistema de sucção é acionado para segurar a fruta. Devido à falta de simulação de todos os elementos reais, como o vento, o critério de colheita bem-sucedida foi rigoroso. Qualquer toque, por menor que fosse, nos galhos ou nas folhas foi considerado uma colisão, e caso o encaixe da ventosa não estivesse centralizado corretamente na maçã, a colheita foi considerada bem-sucedida, mas com potencial problema.

B. Componentes 1 e 4

Os Componentes 1 e 4, responsáveis por aproximar o drone da árvore e movimentá-lo para largar a maçã colhida na despensa, não possuem algoritmos de aprendizado de máquina e receberam pouca atenção neste trabalho. Por esse motivo, considerou-se que esses componentes utilizam sensores de GPS ou outros sensores com precisão suficiente para locomover o drone para o local desejado. Para realizar as movimentações, foram utilizadas as funções de locomoção *moveToPositionAsync* e *moveToZAsync* disponíveis no *Airsim*, sem prévias consultas à qualidade dos serviços em drones reais.

Com o objetivo de facilitar a construção do protótipo, não foi utilizado um banco de dados externo para o armazenamento da localização da árvore e da despensa, mas sim adicionado diretamente no código-fonte. No entanto, para armazená-las, poderia ser utilizado o banco de dados *SQLite*[13], por ser um banco de dados leve e que atende às necessidades.

O Componente 1 foi responsável por posicionar o drone a uma distância de aproximadamente 75 cm da árvore e 35 cm acima do solo, onde se encontram as maçãs mais baixas. Sempre que nenhuma maçã com possibilidade de colheita foi detectada pelo Componente 2 na região atual, o drone foi movido cerca de 35 cm para cima em relação à posição atual. Isso permitiu observar uma nova região da árvore sem perder completamente de vista a região anterior, diminuindo as chances de não coletar uma maçã. Esse movimento foi configurado para ser realizado até atingir a altura máxima onde poderia haver maçãs, resultando em 7 vezes e 2,45 metros de altura. Após atingir o limite de altura, o drone foi movido cerca de 85 cm para a esquerda e, quando nenhuma maçã foi detectada na região, desceu aproximadamente 35 cm de altura, no máximo 7 vezes, até atingir uma altura de cerca de 35 cm acima do solo. Desta forma, o Componente 1 movimentou o drone em um formato semelhante a um “S” deitado, representado na Figura 7, até explorar todas as regiões da árvore.

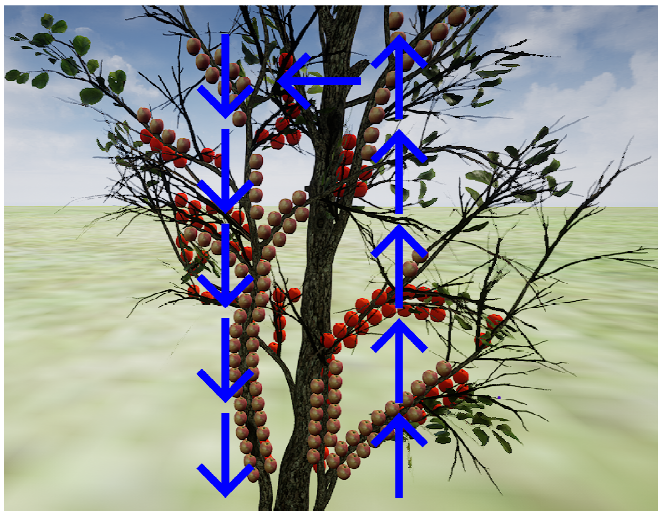

Fig. 7. Movimento semelhante a um “S” realizado pelo drone

A despensa, presente na parte inferior esquerda da Figura 6, foi construída com uma abertura de 1 m², 48 cm de altura e posicionada a 2 metros de distância da árvore, para que houvesse um espaço entre a despensa e a árvore no qual o drone pudesse se movimentar livremente para colher as maçãs.

Após fixar a maçã na ventosa e realizar o movimento para trás, o Componente 4 foi responsável por verificar se a altura do drone estava acima de 80 cm. Caso estivesse abaixo, o drone deslocou-se até atingir a altura desejada. Em seguida, o Componente 4 movimentou o drone linearmente até que a ventosa se encontrasse aproximadamente no centro da despensa, a uma altura de 51 cm, para liberar a maçã da ventosa, deixando-a cair na despensa. Com a maçã liberada, o Componente 1 movimentou o drone para as coordenadas X e Y da última região em que se encontrava antes de colher a maçã. Caso a altura (Y) fosse abaixo de 80 cm, o drone primeiro deslocava-se para essa altura antes de ser movimentado para a altura da última região. Devido à instabilidade da movimentação do drone pela ferramenta utilizada, esperava-se 5 segundos para que o drone se estabilizasse antes de passar para o Componente 2.

[12] https://www.blender.org
[13] https://www.sqlite.org/index.html

C. *Componente 2*

O Componente 2 utilizou a arquitetura *YOLOv7* para realizar a detecção e localização das maçãs, por ser considerada uma das melhores arquiteturas de detecção de objetos em tempo real na base de dados *COCO* [32]. Para diminuir o tempo de processamento, foi utilizada a versão *YOLOv7-Tiny*, a mais compacta entre todas as disponíveis no repositório oficial do projeto no *GitHub*, e as imagens foram redimensionadas para a resolução de 416x416 pixels, com *stride* de 32. Com o objetivo de simplificar a saída da rede e facilitar possíveis customizações futuras, foram normalizadas as coordenadas dos *bounding boxes*, que consistem nas coordenadas de localização da parte superior esquerda (X e Y), da largura (W) e da altura (H) das caixas de detecção das maçãs na imagem. Dessa forma, mesmo que a resolução da captura das imagens pelo drone seja alterada, as coordenadas da localização das maçãs continuarão tendo o mesmo efeito, não prejudicando o resultado do Componente 3, que recebe essas coordenadas como entrada.

Visando obter uma melhor descrição do estado atual da plantação, cada maçã detectada foi classificada em 4 categorias diferentes pelo *YOLOv7-Tiny*, sendo elas: *normal red* (*NR*) para não obstruída e madura, *normal green* (*NG*) para não obstruída e verde, *obstructed red* (*OR*) para obstruída e madura e *obstructed green* (*OG*) para obstruída e verde.

Para simplificar a entrada de informações no Componente 3, foi utilizado um algoritmo de rastreamento no Componente 2, responsável por identificar a maçã mais próxima, utilizando as coordenadas de localização das maçãs. Isso permitiu passar apenas a localização de uma única maçã, que era o objetivo no momento, ignorando as outras frutas presentes no campo de visão da câmera. Dessa forma, a cada movimento realizado pelo drone, o objetivo era sempre a mesma fruta até que sua colheita fosse concluída.

Com o objetivo de aumentar a eficiência do sistema e reduzir as chances de ocorrerem problemas devido ao ângulo de visão das maçãs distantes, foram desconsideradas as maçãs próximas das bordas ou muito abaixo do braço. Os limites dessas regiões foram ilustrados com retângulos pretos na Figura 8. Essas maçãs foram deixadas para serem colhidas quando o drone se locomovesse para a próxima região. Foram utilizadas as seguintes configurações de coordenadas: X e Y menores que 0,15, e X e Y maiores que 0,75 e 0,70, respectivamente.

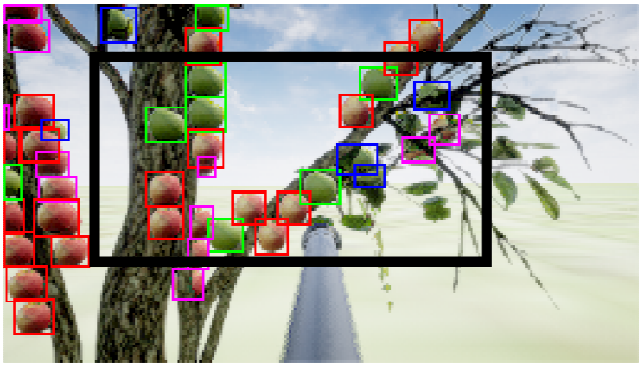

Fig. 8. Limites para colher maçãs

D. *Componente 3*

Por apresentar diversos algoritmos de aprendizado por reforço e flexibilidade para realizar customizações nos hiperparâmetros, foi utilizado o *framework Tensorforce*. Desta forma, foi possível avaliar diversos algoritmos e hiperparâmetros para o Componente 3 mover o drone até a maçã. Para acelerar o aprendizado, foi desenvolvido um pré-treinamento, chamado de Sistema Manual, com o objetivo de realizar o mesmo trabalho que o Componente 3 precisa fazer, porém, utilizando regras lógicas semelhantes ao trabalho de [16], além das coordenadas da maçã utilizadas pelo componente oficial, foram usadas informações adicionais da distância entre o drone e a maçã. Com estas informações que estão disponíveis apenas no ambiente de simulação, foi possível criar regras lógicas, como: Se o drone estiver a mais do que determinados centímetros de distância da maçã, deve se movimentar mais rápido e quando estiver com poucos centímetros de distância, deve se movimentar mais devagar e o mesmo para as direções horizontais e verticais. Sendo que foi priorizado na construção destas regras lógicas a velocidade para realizar a colheita, sem sacrificar a segurança do drone.

Com os dados de diversas colheitas bem-sucedidas realizadas pelo Sistema Manual, foi realizado o pré-treinamento dos algoritmos de aprendizado por reforço avaliados neste trabalho. Tal procedimento permitiu ter um conhecimento prévio das ações que precisam ser realizadas visando melhorar o desempenho e realizar a colheita mesmo sem as informações de distância entre o drone e a maçã utilizadas pelo Sistema Manual. Esta abordagem permitiu avaliar o desempenho de diversos algoritmos de aprendizado por reforço aplicados em 3 arquiteturas diferentes em pouco tempo. Caso contrário, poderia levar vários dias para realizar o treinamento de cada um dos algoritmos até alcançar resultados satisfatórios no hardware utilizado.

Por fim, foi desenvolvido um sistema baseado nas informações das coordenadas da maçã e na velocidade atual do drone, para identificar quando o sistema de sucção presente na ventosa deve ser acionado para segurar a maçã. Para realizar cada movimento do drone, foi utilizado o comando *moveByVelocityAsync* do *Airsim*, com tempo de movimentação de 0,001 segundo.

VI. Treinamentos e Avaliações

Os treinamentos e avaliações do modelo *Dronapple* foram realizados em dois ambientes de simulação desenvolvidos conforme a necessidade do algoritmo. Para as avaliações finais do modelo completo, foi utilizado o Ambiente 2. Um computador com processador Intel I5-3330, 12 GB de memória RAM DDR3, placa de vídeo Nvidia GTX 760 Ti e sistema operacional Windows 10 foi utilizado para realizar todas as etapas. Devido à incompatibilidade da placa de vídeo com os sistemas de aprendizado de máquina atual, o modelo foi avaliado apenas com CPU, e o simulador foi executado no mesmo computador que o modelo, o que limitou a velocidade de execução. Todos os treinamentos foram realizados no computador mencionado, exceto para a rede *YOLOv7-Tiny*, que

exigiu um computador com mais poder computacional para o treinamento, utilizando o sistema em nuvem *Google Colab*[14].

Devido aos Componentes 1 e 4, que movimentam o drone até a árvore e, posteriormente, até a despensa para depositar a maçã colhida, não apresentarem aprendizado de máquina, apenas regras lógicas e utilizarem ferramentas disponíveis no *plugin Airsim*, foi avaliado apenas o tempo necessário para executar suas ações.

No Componente 2, foram realizados dois treinamentos distintos da rede *YOLOv7-Tiny* para detecção e classificação das maçãs: um para cada ambiente. No Ambiente 1, contendo somente a categoria *NR*, cada imagem possuía apenas uma maçã, totalizando 1239 imagens para treinamento e 413 imagens para validação, com um *batch size* de 64 por 300 épocas. Já no Ambiente 2, foram utilizadas as quatro categorias: 1174 *NR*, 653 *NG*, 1111 *OR* e 589 *OG*, com 150 imagens para treinamento e 50 imagens para avaliação, totalizando 362 *NR*, 204 *NG*, 358 *OR* e 192 *OG*. O *batch size* foi de 64 por 2300 épocas. Todas as imagens foram coletadas manualmente utilizando a câmera do drone em cada um dos ambientes, simulando movimentos e distâncias possíveis de serem realizadas pelo Componente 3. As anotações foram realizadas utilizando a ferramenta *LabelImg*[15], seguindo os padrões de *bounding box* da arquitetura *YOLO*.

No sistema de rastreamento das maçãs, foi avaliada a possibilidade de localizar a maçã mais próxima, utilizando o cálculo da distância euclidiana entre as coordenadas normalizadas X e Y da localização superior esquerda do *bounding box* da maçã gerada pelo *YOLOv7-Tiny*, em relação às coordenadas 0,45×0,55. Essa posição encontra-se aproximadamente na parte superior esquerda da ventosa. Com a maçã objetivo definida, foi avaliada a utilização do algoritmo *DeepSORT* [43], para melhorar a eficiência no rastreamento. Para a implementação, foi utilizado o repositório "yolov7-deepsort-tracking"[16], que possui a implementação do algoritmo em conjunto com a arquitetura *YOLOv7*.

As avaliações do sistema de rastreamento foram realizadas no Ambiente 1, utilizando o cenário com nove maçãs próximas uma da outra e com três diferentes distanciamentos entre elas. Foram analisados os comportamentos do sistema quando existem maçãs muito próximas e mais afastadas, sendo os distanciamentos laterais de 0,14 cm, 1,14 cm e 1,64 cm e distanciamentos superiores de 0,2 cm, 0,7 cm e 1,7 cm, respectivamente (distâncias pequenas para simular casos reais em que as maçãs estão extremamente próximas, quase como se fossem "cachos"). O sistema de movimentação do drone realizado pelo Componente 3 utilizado para as avaliações foi o mesmo implementado na versão final do modelo *Dronapple*, ou seja, o sistema de rastreamento foi avaliado após o Componente 3 estar treinado e com as configurações finais. Dessa forma, foi possível avaliar o desempenho do algoritmo de rastreamento e sua capacidade de manter o objetivo sempre na mesma maçã.

No Componente 3, foram avaliadas as políticas de aprendizado por reforço *PPO* e *DDPG* por serem consideradas leves e com bons resultados em agentes de ações contínuas. Os treinamentos foram realizados no Ambiente 1, utilizando o Componente 2, que foi treinado neste mesmo ambiente, reduzindo a complexidade e os erros de detecção, além de não exigir a realização de rastreamento, já que apenas uma maçã estava presente no campo de visão do drone.

Na construção da arquitetura, foram utilizados os hiperparâmetros padrões das políticas disponíveis na versão 0.6.5 do *Tensorforce*. O *batch size* foi configurado para 64 e a *learning rate* para 0,0001 para ambas as políticas. O *memory* foi definido como *minimum* para o *PPO* e 5000 para o *DDPG*. Após as avaliações, foram realizados novos testes com a política que apresentou os melhores resultados. Para isso, foram criadas três arquiteturas de redes neurais com quantidades diferentes de neurônios, sendo uma delas a arquitetura padrão do *Tensorforce* e as outras duas inspiradas nela. Essas arquiteturas foram ilustradas utilizando o programa Netron[17] na Figura 9. Cada uma delas foi avaliada com seis funções de ativação diferentes disponíveis no *Tensorforce* entre cada uma das camadas de redes neurais, além de uma avaliação sem qualquer tipo de função de ativação, totalizando 21 avaliações. As funções testadas foram: *Tanh*, *ReLU*, *ELU* [12], *Leaky-ReLU*, *SELU* [24] e *Sigmoid*.

Cada uma das arquiteturas foi dividida em três blocos principais. O primeiro bloco foi composto por uma rede neural que recebeu como entrada as coordenadas da localização da maçã classificada para ser colhida pelo Componente 2 e 4 elementos (X, Y, W, H). O segundo bloco, outra rede neural, recebeu como entrada a velocidade atual X, Y e Z do drone. O terceiro e último bloco foi composto por uma rede neural que recebeu como entrada a concatenação dos resultados das duas redes neurais anteriores. Essa rede neural retornou uma lista com três valores de ponto flutuante para ser passado para o *plugin Airsim*, podendo cada um deles estar entre -1,0 e 1,0 para indicar a intensidade que o drone deve ser movimentar nas coordenadas X, Y e Z, respectivamente.

Todos os treinamentos foram realizados no Ambiente 1 e seguiram o mesmo padrão de pré-treinamento, com um histórico de 100 épocas utilizando o Sistema Manual e mais 400 épocas de treinamento utilizando aprendizado por reforço. Para avaliar os melhores momentos do agente após o treinamento, cópias do modelo foram criadas toda vez que o agente atingia o próprio recorde de maior valor de recompensa ou menor tempo para coletar a maçã com sucesso. Para estimular a exploração, foi utilizado um ruído com 35% de chance em cada uma das coordenadas de velocidade X, Y e Z resultantes do modelo, variando de -0,05 a 0,05. A cada 5 épocas, o modelo foi executado por 1 época completa sem a exploração, para avaliação.

Cada uma das cópias das arquiteturas criadas durante o treinamento foi avaliada no Ambiente 1 com uma maçã durante a execução de 20 épocas. Para cada época, foram armazenadas

[14] https://colab.research.google.com
[15] https://github.com/tzutalin/labelImg
[16] https://github.com/deshwalmahesh/yolov7-deepsort-tracking
[17] https://netron.app

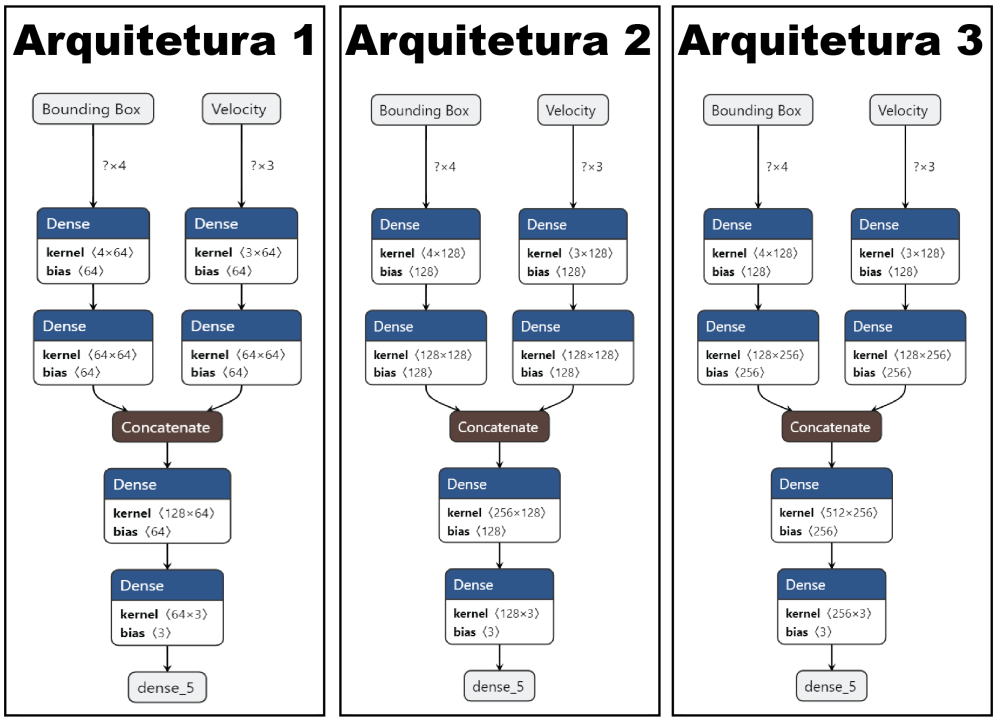

Fig. 9. Arquiteturas avaliadas para o Componente 3

as seguintes informações a serem utilizadas nas avaliações: quantidade de movimentos, tempo até o término da época, tempo médio para processar cada movimento, *FPS* médio para processar cada movimento, recompensa total e última recompensa. Para realizar os cálculos de recompensa, foi criado uma fórmula inspirada no trabalho de Duisterhof et al. [18], sendo ela:

$$r = (1000 - 1000\upsilon)\alpha - 100\beta + 20\Delta \quad (1)$$

A equação 1 consiste em dois valores binários ativados se atenderem determinadas condições, sendo elas: α, ativado caso a maçã seja encaixada corretamente na ventosa, e β, ativado caso sejam realizadas 100 movimentações sem atingir o objetivo α ou caso o drone, incluindo o braço, colida com algum objeto no cenário. Para incentivar o agente a possuir baixa velocidade ao coletar a maçã, foi adicionada a variável υ para reduzir a recompensa conforme a soma dos valores absolutos da velocidade X, Y e Z do drone no momento em que coleta a maçã.

Para determinar se cada movimento realizado foi benéfico, é feito o cálculo Δ, presente na equação 2, que consiste na distância entre o drone e a maçã no tempo anterior, D_{t-1} subtraída pela distância no tempo atual, D_t. O símbolo $_t$ representa o tempo atual e $_{t-1}$ representa o tempo anterior. Com isso, a recompensa tende a ser maior se o drone se aproximou da maçã.

$$\Delta = D_{t-1} - D_t \quad (2)$$

No sistema que decide quando acionar a sucção da ventosa, foi utilizada a biblioteca *Sklearn* para fazer o pré-processamento dos dados e avaliar algoritmos de aprendizado de máquina, e a biblioteca *Pytorch* para avaliar algoritmos de aprendizado profundo. Para criar a base de dados, foi executa as partes dos Componentes que já estavam treinados, sendo os componentes 2 para detectar a maçã e o 3 para movimentar o drone até a maça e armazenado as velocidades X, Y e Z atuais do drone e as coordenadas X, Y, W e H da maçã objetivo de todos os movimentos realizados no Ambiente 1 e toda vez que a ventosa era encostada na maçã (detectado pelo ambiente de simulação) foi reiniciado o ambiente e classificado um movimento antes de encostar a ventosa na maçã como sendo o momento ideal para o sistema de sucção ser acionado, totalizando 12096 dados de movimentações do drone, contendo desde o ponto inicial até dois passos antes de coletar a maçã, e 12096 dados de um passo antes de realizar a coleta da maçã, para o sistema ser acionado com o último movimento realizado pelo drone. A divisão dos dados foi realizada utilizando a função *train_test_split* da biblioteca *Sklearn*, com as seguintes divisões: 60% para treinamento, 20% para validação e 20% para teste.

Durante o treinamento, foram avaliados quatro tipos de pré-processamento para os 7 valores de entrada que são as velocidades X, Y e Z do drone e as coordenadas X, Y, W e H da maçã objetivo no tempo atual. Os dois primeiros consistiram em escalar os dados utilizando a função *StandardScaler* do *Sklearn*, e em um deles, aplicar *Principal Component Analysis* (PCA) contendo de 1 a 6 componentes. Para os outros dois, foram utilizados os valores originais e novamente, em um deles, foi utilizado PCA contendo de 1 a 6 componentes. Para cada pré-processamento, foram avaliados os seguintes algoritmos: *Logistic Regression*, *Support Vector Machine* (SVM), *Random Forest*, *Naive Bayes*, *Decision Tree*,

K-Nearest Neighbors e rede neural, totalizando 98 avaliações.

O modelo da rede neural foi treinado por 300 épocas, e a cada 5 épocas, a acurácia nos dados de validação foi verificada e armazenado somente o modelo com a melhor acurácia. Para a arquitetura, foi utilizada uma camada com 128 neurônios, função de ativação *ReLU* e a segunda camada consistiu na saída da rede com *Sigmoid*. Para o cálculo de *loss*, foi utilizado *Binary Cross Entropy*, e *Adam* como otimizador.

Após as avaliações e a escolha do melhor algoritmo e hiperparâmetros, o sistema foi executado durante 100 épocas no Ambiente 1, com o objetivo de avaliar se o algoritmo seria capaz de acionar a sucção no momento certo. Para evitar prejudicar o resultado deste sistema, foram removidas situações em que o computador processou menos de 5 *FPS, o que limita a estabilidade da movimentação do drone* e o sistema de locomoção não conseguiu levar o drone até a maçã adequadamente, o que não representa problema no sistema de sucção e sim, no Componente 2 de detecção de maçãs ou do 3 para movimentar o drone até a fruta.

Após realizar as comparações e avaliações de todos os algoritmos sugeridos, foram selecionados aqueles que melhor se adequaram a cada uma das tarefas propostas neste trabalho. Como avaliação final, foram realizadas a colheita de 75 árvores utilizando o Ambiente 2, com o objetivo de realizar uma avaliação completa do modelo *Dronapple*, incluindo o funcionamento dos 4 componentes para realizar o fluxo completo. Cada árvore apresentou diversas maçãs em posições e condições conforme as probabilidades mencionadas anteriormente. Considerou-se o fim da colheita de cada árvore quando o Componente 1 atingiu a última região válida e o Componente 2 não detectou maçãs possíveis de serem colhidas no local. Dessa forma, foi possível realizar a comparação do desempenho do modelo proposto com trabalhos relacionados.

VII. RESULTADOS

Na validação dos dados no Ambiente 1, o modelo de detecção e classificação de maçãs do Componente 2 apresentou uma precisão de 100%, o que era esperado devido à baixa complexidade do ambiente. Já no Ambiente 2, que apresentou maiores desafios, o modelo teve uma precisão média de 95,1% e um *recall* médio de 94,5%. A categoria *NR* apresentou tanto precisão quanto *recall* de 97,5%, enquanto a categoria *OG* obteve o pior resultado, com uma precisão e *recall* de 88,5% e 90,1%, respectivamente.

A Tabela I apresenta os resultados detalhados de ambos os modelos, conforme o ambiente de simulação, tendo somente a classe *NR* no Ambiente 1, pois não foram utilizadas as outras classes neste ambiente e as 4 classes (*NR*, *NG*, *OR* e *OG*) presentes no Ambiente 2, além da união delas.

Como um dos objetivos foi priorizar a eficiência, principalmente devido ao poder computacional limitado do computador utilizado, o algoritmo *DeepSORT* não pôde ser mantido devido ao tempo de processamento elevado. Por este motivo, foi alterado para, em vez de utilizar o cálculo euclidiano somente no início para identificar a maçã mais próxima, ser também utilizado durante cada um dos movimentos do drone, com o objetivo de rastrear a maçã atual mais próxima. Em um teste realizado com 1000 imagens do Ambiente 2, foi identificado que o tempo médio de processamento por imagem do sistema euclidiano foi de 0,104 segundo e do *DeepSORT* foi de 1,04 segundo. Nos três cenários de avaliação no Ambiente 1 com 9 maçãs, o sistema euclidiano obteve uma acurácia de 100% nos dois primeiros distanciamentos e 95% no terceiro. Porém, ao analisar o único erro obtido, constatou-se que não foi ocasionado pelo algoritmo de rastreamento, mas sim pelo modelo de aproximação do drone, que não conseguiu centralizar corretamente com a maçã no momento final. Nos testes realizados, isso ocorreu em uma pequena percentagem dos casos, como poderemos ver nos resultados mostrados mais à frente neste artigo.

Os resultados das primeiras avaliações do Componente 3 para identificar a melhor política para movimentar o drone até coletar a maçã, mostraram que a política *DDPG* não apresentou sinais de aprendizado, resultando em uma acurácia de 0%, ou seja, em nenhum dos casos conseguiu movimentar o drone até encaixar a maçã na ventosa. Ao contrário do *PPO* que apresentou uma acurácia de 95% e um tempo de colheita da maçã médio de 7,77 segundos. Com estes resultados, foram realizados os próximos treinamentos utilizando apenas a política *PPO*, conforme discutido na seção anterior.

Dentre todas as arquiteturas e hiperparâmetros testados, diversas tiveram acurácia de 100%, sendo que as funções de ativação *SELU* e *Leaky-ReLU* tiveram os melhores resultados. Como medida de desempate foi utilizada a média da quantidade de movimentos para colher a maçã, pois quanto menos movimentos realizados, menos processamento utilizado e energia gasta, tanto com o dispositivo que estiva executando a arquitetura, quanto o drone para se movimentar. Desta forma, foi escolhida a Arquitetura 3 com função de ativação *SELU*, com média de 37,95 movimentos, tempo médio de colheita de 6,25 segundos e média de 6,09 *FPS*.

Tabela I: Resultados do modelo de detecção e classificação de maçãs do Componente 2

	Classe	***Imagens***	***Anotações***	***Precisão (%)***	***Recall (%)***
Ambiente 1	*NR*	413	413	100	100
Ambiente 2	*NR*	50	362	97,5	97,5
	NG	50	204	98	97
	OR	50	358	96,2	93,6
	OG	50	192	88,5	90,1
	Todas	50	1116	95,1	94,5

Tabela II: Melhores resultados dos modelos de movimentação do drone do Componente 3

Arquitetura	***Função de Ativação***	***Acurácia (%)***	***Movimentos (média)***	***Segundos (média)***
3	*SELU*	100	37,95	6,25
3	*Leaky-ReLU*	100	38,10	6,13
3	*Tanh*	100	38,15	6,49
2	*SELU*	100	40,05	6,48
2	*Leaky-ReLU*	100	40,35	6,52

A Tabela II apresenta os resultados das 5 melhores arquiteturas e funções de ativação, contendo a quantidade média de movimentações e tempo médio para colher cada maçã e a Tabela V do Apêndice apresenta todos os resultados, representados com o símbolo "-" nas situações em que não conseguiram colher nenhuma maçã. Ambas as tabelas estão ordenadas por acurácia e quantidade média de movimentações.

No sistema de sucção, o algoritmo de aprendizado de máquina *Random Forest* apresentou o melhor resultado, ocupando as três primeiras colocações da categoria referentes à acurácia nos dados de teste. Sua melhor configuração ocorreu com os dados escalados como pré-processamento e teve uma acurácia de 98,37% nos dados de teste. Já os algoritmos de aprendizado profundo, ocupando os três primeiros lugares da categoria referentes à acurácia nos dados de teste, tiveram pré-processamentos diferentes, sendo eles: valores escalados com 98,35% de acurácia, valores escalados com *PCA* de 6 componentes com 97,85% de acurácia e valores naturais com *PCA* de 6 componentes com 97,29% de acurácia, respectivamente.

Pode-se visualizar na Tabela III todos os algoritmos que tiveram a maior acurácia nos dados de teste, com informações se os dados foram escalados e caso tenham utilizado *PCA*, foi adicionado a quantidade de componentes, caso contrário, o símbolo "-", além da acurácia nos dados de validação e teste. A Tabela VI do Apêndice apresenta os resultados de todos os algoritmos avaliados.

Como os algoritmos *Random Forest* e rede neural tiveram resultados semelhantes, o tempo de processamento de cada um dos algoritmos foi analisado como critério de desempate. Para medir o tempo, cada algoritmo foi executado 20 vezes e a média foi coletada. Os resultados foram de 0,028 segundo e 0,013 segundo para *Random Forest* e rede neural, respectivamente. Devido aos resultados, a rede neural foi escolhida, que além de possuir acurácia semelhante, é aproximadamente duas vezes mais rápida em comparação ao *Random Forest*.

Em avaliações realizadas no Ambiente 1 com a rede neural escolhida para acionar a sucção, constatou-se que o sistema sempre foi acionado em algum movimento antes de alcançar a maçã, o que garantiu que em todos os momentos que a ventosa encostou na maçã o sistema estivesse acionado. No melhor resultado, o sistema foi acionado faltando 3 movimentos para atingir a maçã e no pior caso, faltando 13 movimentos. O sistema realizou em média 6,07 movimentos precoces com desvio padrão de 2,06 movimentos, em um cenário onde o drone teve que realizar em média 40,19 movimentos com desvio padrão de 3,83 movimentos até chegar na fruta. Considerando a quantidade média de movimentos necessários para alcançar a maçã e a média de acionamentos precoces subtraídas por 1, pois o sistema precisa ser acionado no mínimo um movimento antes de atingir a fruta, o sistema ficou acionado ociosamente 12,6% do tempo.

Devido à utilização das ferramentas do *Airsim* para realizar as operações dos Componentes 1 e 4 e a baixa velocidade do drone, não foram detectados problemas ou situações que ocasionaram a colisão do drone com algum objeto no cenário.

No Ambiente 2, o Componente 1 levou em média 1,08 segundo para realizar a troca de posição superior, 0,79 segundo inferior e 5,89 segundos para movimento horizontal. Para o Componente 2, não foi possível apresentar um resultado quantitativo além dos apresentados durante o treinamento, pois seria necessário realizar as anotações de todas as imagens durante as 75 árvores coletadas. Por este motivo, foram realizadas análises visuais das execuções. Nessas análises, foram identificadas situações em que o sistema de detecção e localização do drone classificou a maçã com a categoria errada, fazendo com que o drone se locomovesse até uma maçã que não deveria ser colhida. Além disso, foram identificadas mudanças de classificação, demonstradas na Figura 10 pela flecha amarela. Na figura, a maçã foi inicialmente classificada como uma categoria e, após realizar alguns movimentos, a mesma maçã foi classificada como outra categoria, fazendo com que o drone mudasse de objetivo e, muitas vezes, colidisse, assim como deixasse de identificar a maçã por ficar obstruída pelo braço do drone.

Tabela III: Melhores resultados da sucção do Componente 3

Algoritmo	***Escalado***	***Componente***	***Acurácia Validação***	***Acurácia Teste***
Random Forest	Sim	-	98,41	98,37
Rede Neural	Sim	-	98,55	98,35
Random Forest	Não	-	98,43	98,33
Rede Neural	Sim	6	98,14	97,85
Random Forest	Não	6	97,91	97,62
Rede Neural	Não	6	97,40	97,29

Fig. 10. Mudança da classificação da maçã

Para o Componente 3 movimentar o drone até a maçã, foram necessárias em média 30,43 movimentações, o que levou em média 5,32 segundos, executado em média com 5,74 *FPS*. O sistema de sucção foi acionado 100% das vezes em algum momento antes de encostar na maçã, tendo em média 6,58 movimentações precoces. Considerando que deveria ser apenas 1 movimentação precoce, o sistema ficou ocioso 18,34% do tempo. Por fim, para o Componente 4 movimentar o drone até a despensa e o Componente 1 mover de volta para a última posição de colheita, foram necessários em média 17,75 segundos. Em resumo, para realizar a colheita de cada maçã foram necessários em média 23,08 segundos.

De todas as frutas disponíveis nas árvores durante as execuções, foram colhidas 2694 maçãs normais vermelhas (*NR*), 1 maçã normal verde (*NG*), 6 maçãs vermelhas obstruídas (*OR*) e nenhuma maçã verde obstruída (*OG*), o que correspondeu a uma acurácia de 97,93%, 99,94%, 99,78%, 100%, respectivamente. Durante as colheitas, após a remoção de dados com menos de 5 *FPS*, 91,05% das maçãs *NR* colhidas foram consideradas colheitas bem-sucedidas, enquanto 8,95% foram consideradas colheitas com problemas potenciais devido à ventosa não estar totalmente encaixada na maçã, demora na comunicação com o simulador devido ao baixo *FPS* ou por estar com velocidade suficiente para empurrar a maçã para dentro da árvore após a colheita, o que poderia ocasionar colisões caso houvesse algum objeto atrás da maçã colhida. Após a colheita da maçã, não foram identificadas colisões do braço com a árvore ou outras maçãs além daquela presa na ventosa.

As situações consideradas malsucedidas e nas quais a colheita de maçã não foi realizada corresponderam a 213 ocasiões (7,73%) das 2755 tentativas de colheitas com mais de 5 *FPS*. Dessas ocasiões, em 31,46% o braço colidiu com algum galho ou tronco da árvore, em 66,20% as colisões ocorreram ao tentar encaixar corretamente a ventosa na maçã e em 2,35% o braço colidiu com alguma maçã obstruída. Não foram detectadas colisões do corpo do drone com algum objeto no cenário e, de todas as colisões mencionadas, nenhuma foi considerada grave, pois o drone estava em baixa velocidade.

A Tabela IV apresenta a comparação dos resultados do modelo *Dronapple* no simulador desenvolvido com os resultados apresentados pelos próprios autores em suas publicações, mencionadas na seção de trabalhos relacionados. Ressalta-se que o modelo *Dronapple* foi executado e avaliado em ambientes de simulação, enquanto alguns dos trabalhos relacionados foram realizados no mundo real. Dessa forma, pode-se considerar a comparação injusta, porém optou-se por realizá-la para demonstrar o potencial do modelo proposto.

Tabela IV: Comparação dos resultados com trabalhos relacionados

Referência	***Ambiente***	***Tipo avaliação***	***Maçãs colhidas (%)***	***Tempo para colher (seg.)***
[7]	Real	Disponíveis	80	8-10
[16]	Real	Sucesso	77	15,4
[2]	Simulador	Sucesso	92	-
[39]	Real	Sucesso	84,6	5,8-6,2
[20]	Real	Sucesso	67	7,3
[22]	Laboratório	Disponíveis	81-91	7
[46]	Real	Sucesso	64,06	8,8
[21]	Real	Sucesso	47,37	4
[23]	Laboratório	Sucesso	89,2	12,3
[42]	Real	Sucesso	73,33	14,69
Dronapple	Simulador	Sucesso	91,05	23,08

Pode-se observar que o modelo *Dronapple* apresentou uma das melhores acurácias em tentativas de colheita, tendo apenas o trabalho de Ahlin et al. [2] que também foi executado em simulação com 0,95% a mais de sucesso. Também pode-se destacar que todos os trabalhos possuem uma limitação de altura específica para realizar a colheita. O modelo *Dronapple* apresentou o benefício de conseguir colher até mesmo as maçãs posicionadas nas partes mais altas das árvores, possibilitando a realização da colheita de mais maçãs. Além disso, possui recursos para detectar maçãs que estão obstruídas, evitando-as para não danificar o equipamento. No entanto, destaca-se as vantagens dos trabalhos de Ahlin et al. [2], Kang et al. [22] e Kang et al. [23] que conseguem colher maçãs obstruídas.

Devido aos trabalhos de Baeten et al. [7] e Kang et al. [22] não apresentarem a taxa de sucesso de colheita, foi utilizado o valor disponibilizado que corresponde à quantidade de maçãs colhidas dentre as que o robô teoricamente poderia colher. Como medida de comparação, pode-se colocar que o *Dronapple* apresentou uma taxa de sucesso de colheita de 97,73%.

Comparando o tempo necessário para realizar a colheita de cada uma das maçãs, os trabalhos relacionados apresentam a vantagem de possuírem a despensa acoplada no próprio dispositivo, reduzindo o tempo para levar e largar a maçã na despensa e em alguns casos não foi realizada esta medição. Se considerarmos somente o tempo de colheita do modelo

Dronapple, ele conseguiu colher em média 5,32 segundos, mesmo utilizando um computador com baixo poder computacional para modelos de aprendizado de máquina. Devido à limitação de não possuir uma despensa acoplada e à pouca dedicação neste trabalho à otimização do processo de depósito da maçã, resultou em um tempo consideravelmente maior em comparação com os trabalhos relacionados.

VIII. Conclusões e Trabalhos Futuros

Este trabalho propôs o modelo *Dronapple* para realizar a colheita de maçãs utilizando drones equipados com uma câmera RGB e uma ventosa, com a utilização de algoritmos de aprendizado de máquina para realizar as operações sem a necessidade de auxílio humano, com o intuito de automatizar parcialmente o trabalho de colheita de maçãs.

O *Dronapple* foi modelado em componentes para facilitar customizações e utiliza algoritmos considerados de baixo consumo, o que possibilita que todos os processamentos possam ser realizados em um microcomputador equipado no próprio drone ou em um servidor de baixo custo.

Para realizar o treinamento dos algoritmos de aprendizado de máquina e comprovar o funcionamento do modelo, foram desenvolvidos dois ambientes de simulação, um especificamente para treinamentos e o segundo para avaliações. Com base nos resultados, o modelo se mostrou promissor, com capacidade de movimentar o drone para coletar maçãs e conseguir evitar regiões onde se encontram maçãs com algum potencial risco de colisão do drone devido à presença de galhos e folhas, além de conseguir levar a maçã coletada até uma despensa, o que possibilitou realizar a colheita de diversas maçãs sem a necessidade de interação humana.

A principal contribuição deste trabalho é a especificação de um modelo aéreo para realizar a colheita de maçãs, possibilitando que seja automatizada a colheita de maçãs não obstruídas sem a necessidade de alterações na superfície terrestre e sem limitação na altura das árvores, pois o drone possui a capacidade de voar a dezenas de metros de altura. Da mesma forma, a despensa pode estar localizada em qualquer região da plantação e ser compartilhada entre diversos drones. Devido à falta de um operador, a colheita não fica limitada pela mão de obra e o plantio pode ser parcialmente automatizado, ficando a cargo de um coletor humano apenas a colheita de maçãs obstruídas, além da possibilidade de vincular outras funcionalidades como detecção de pragas e doenças nas árvores.

Com base nos estudos deste trabalho, surgem novas possibilidades para trabalhos futuros. Para o Componente 4, se mostra necessário melhorias para diminuir o tempo de colheita, pois este componente possui um tempo elevado para ser executado, podendo-se aplicar aprendizado de máquina para aumentar a velocidade das movimentações. O mesmo ocorre para o Componente 1, para aumentar a estabilidade e diminuir a probabilidade de não visualizar uma maçã na região. Além disso, considera-se importante a utilização de um hardware com maiores poderes computacionais, para melhorar a estabilidade dos movimentos do drone, buscar diminuir o tempo total de colheita e implementar um sistema mais robusto de rastreamento como o *DeepSORT*.

Outro trabalho futuro consiste na aplicação do modelo em um drone real para identificar possíveis desafios e como solucioná-los, como a limitação atual das baterias que possuem poucos minutos de operação e como será realizado o funcionamento da ventosa. Como este trabalho utilizou uma simulação, podem existir aspectos que não foram considerados ou possíveis de simular, como o vento e a resistência da maçã nas árvores para avaliar o sistema de rotação da garra e que poderiam influenciar nas estimativas de aproximação do drone da maçã e a estabilidade do drone, por este motivo é importante avaliar o modelo no mundo real, pois pode ser necessário, para aumentar a confiabilidade, adotar a utilização de um sensor LiDAR, câmeras RGB-D ou câmeras binoculares.

Referências

[1] Afshin, A., Sur, P. J., Fay, K. A., Cornaby, L., Ferrara, G., et al. 2019. Health Effects of Dietary Risks in 195 Countries, 1990–2017: A Systematic Analysis for the Global Burden of Disease Study 2017. The Lancet, 393 (10184): 1958-1972.

[2] Ahlin, K. J., Hu, A.-P., e Sadegh, N. 2017. Apple Picking Using Dual Robot Arms Operating Within an Unknown Tree. In 2017 ASABE Annual International Meeting, page 1. American Society of Agricultural and Biological Engineers.

[3] Ampatzidis, Y., Partel, V., Meyering, B., e Albrecht, U. 2019. Citrus Rootstock Evaluation Utilizing UAV-based Remote Sensing and Artificial Intelligence. Computers and Electronics in Agriculture, 164:104900.

[4] Arulkumaran, K., Deisenroth, M. P., Brundage, M., e Bharath, A. A. 2017. A Brief Survey of Deep Reinforcement Learning. arXiv preprint arXiv:1708.05866.

[5] Azar, A. T., Koubaa, A., Ali Mohamed, N., Ibrahim, H. A., Ibrahim, Z. F., et al. 2021. Drone Deep Reinforcement Learning: A Review. Electronics, 10(9):999.

[6] Bac, C. W., Van Henten, E. J., Hemming, J., e Edan, Y. 2014. Harvesting Robots for High-value Crops: State-of-the-art Review and Challenges Ahead. Journal of Field Robotics, 31(6):888–911.

[7] Baeten, J., Donné, K., Boedrij, S., Beckers, W., e Claesen, E. 2008. Autonomous Fruit Picking Machine: A Robotic Apple Harvester. In Field and service robotics, pages 531–539. Springer.

[8] Barbosa, M., Helfer, G., e Barbosa, J. 2021. Aprendizado de Máquina e Drones Aplicados em Pomares: Um Mapeamento Sistemático. Revista de Sistemas de Informação da FSMA, (28):27–34.

[9] Bochkovskiy, A., Wang, C.-Y., e Liao, H.-Y. M. 2020. Yolov4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[10] Bogue, R. 2020. Fruit Picking Robots: Has Their Time Come? Industrial Robot: The International Journal of Robotics Research and Application.

[11] Butler, S. 2019. Tonnes of Crops Left to Rot as Farms Struggle to Recruit EU Workers. Disponível em: https://www.theguardian.com/business/2019/oct/11/tonnes-of-crops-left-to-rot-as-farms-struggle-to-recruit-eu-workers. Acesso em: 21 de maio de 2022.

[12] Clevert, D., Unterthiner, T., e Hochreiter, S. 2015. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). arXiv preprint arXiv:1708.05866.

[13] Custers, B. 2016. Drones Here, There and Everywhere Introduction and Overview. In The future of Drone Use, pages 3–20. Springer.

[14] Danesckhu, S. 2017. Migrant Labour Shortage Leaves Fruit Rotting on UK Farms. Financial Times.

[15] Davidson, J. R., Silwal, A., Hohimer, C. J., Karkee, M., Mo, C., e Zhang, Q. 2016. Proof-of-concept of a Robotic Apple Harvester. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 634–639. IEEE.

[16] De-An, Z., Jidong, L., Wei, J., Ying, Z., e Yu, C. 2011. Design and Control of an Apple Harvesting Robot. Biosystems engineering, 110(2):112–122.
[17] Departamento de Economia Rural. 2020. Fruticultura Análise da Conjuntura. Disponível em: https://www.agricultura.pr.gov.br/sites/default/arquivos restritos/files/documento/2020-01/fruticultura 2020.pdf. Acesso em: 23 de maio de 2022.
[18] Duisterhof, B. P., Krishnan, S., Cruz, J. J., Banbury, C. R., Fu, W., et al. 2019. Learning to Seek: Autonomous Source Seeking with Deep Reinforcement Learning Onboard a Nano Drone Microcontroller. arXiv preprint arXiv:1909.11236.
[19] Gallardo, R. e Galinato, S. 2019. 2019 Cost Estimates of Establishing, Producing, and Packing Gala Apples in Washington State. Washington State University.
[20] Hohimer, C. J., Wang, H., Bhusal, S., Miller, J., Mo, C., e Karkee, M. 2019. Design and Field Evaluation of a Robotic Apple Harvesting System with a 3D-printed Soft-robotic End-effector. Transactions of the ASABE, 62(2):405–414.
[21] Hu, G., Chen, C., Chen, J., Sun, L., Sugirbay, A., et al. 2022. Simplified 4-dof Manipulator for Rapid Robotic Apple Harvesting. Computers and Electronics in Agriculture, 199:107177.
[22] Kang, H., Zhou, H., e Chen, C. 2020. Visual Perception and Modeling for Autonomous Apple Harvesting. IEEE Access, 8:62151–62163.
[23] Kang, M., Fan, Z., Yu, X., Wan, H., Chen, Q., et al. 2022. Division Merge Based Inverse Kinematics for Multi-dofs Humanoid Robots in Unstructured Environments. Computers and Electronics in Agriculture, 198:107090.
[24] Klambauer, G., Unterthiner, T., Mayer, A., Hochreiter, S., 2017. Self-Normalizing Neural Networks. arXiv preprint arXiv:1706.02515.
[25] LeCun, Y., Bottou, L., Bengio, Y., e Haffner, P. 1998. Gradient-based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11):2278–2324.
[26] Li, Z., Liu, F., Yang, W., Peng, S., e Zhou, J. 2021. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. IEEE Transactions on Neural Networks and Learning Systems.
[27] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., et al. 2015. Continuous Control with Deep Reinforcement Learning. arXiv preprint arXiv:1509.02971.
[28] Liu, S., Dong, W., Ma, Z., e Sheng, X. 2020. Adaptive Aerial Grasping and Perching with Dual Elasticity Combined Suction Cup. IEEE Robotics and Automation Letters, 5(3):4766–4773.
[29] McCulloch, W. S. e Pitts, W. 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity. The Bulletin of Mathematical Biophysics, 5(4):115–133.
[30] Oliveira, V. d. S. E. 2011. As Relações de Trabalho na Colheita da Maçã em Vacaria (RS): Da Autonomia Camponesa ao Controle do Capital.
[31] Osco, L. P., de Arruda, M. d. S., Gonçalves, D. N., Dias, A., Batistoti, J., et al. 2021. A CNN Approach to Simultaneously Count Plants and Detect Plantation-rows from UAV Imagery. ISPRS Journal of Photogrammetry and Remote Sensing, 174:1–17.
[32] Papers With Code 2022. Real-time Object Detection on COCO. Web page. Disponível em: https://paperswithcode.com/sota/real-time-object-detection-on-coco. Acesso em: 8 de agosto de 2022.
[33] Ren, S., He, K., Girshick, R., e Sun, J. 2015. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Setworks. Advances in Neural Information Processing Systems, 28.
[34] Rosenblatt, F. 1958. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. Psychological review, 65(6):386.
[35] Rumelhart, D., Hinton, G., e Wiliams, R. 1986. Learning Representations by Backpropagation Errors. Nature, 323(533-536):10.
[36] Saleem, M. H., Potgieter, J., e Arif, K. M. 2021. Automation in Agriculture by Machine and Deep Learning Techniques: A Review of Recent Developments. Precision Agriculture, 22(6):2053–2091.
[37] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., e Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347.
[38] Shah, S., Dey, D., Lovett, C., e Kapoor, A. 2018. Airsim: High-fidelity Visual and Physical Simulation for Autonomous Vehicles. In Field and Service Robotics, pages 621–635. Springer.
[39] Silwal, A., Davidson, J. R., Karkee, M., Mo, C., Zhang, Q., e Lewis, K. 2017. Design, Integration, and Field Evaluation of a Robotic Apple Harvester. Journal of Field Robotics, 34(6):1140–1159.
[40] Tang, Y., Chen, M., Wang, C., Luo, L., Li, J., et al. 2020. Recognition and Localization Methods for Vision-based Fruit Picking Robots: A Review. Frontiers in Plant Science, 11:510.
[41] UNICEF, et al. 2020. The State of Food Security and Nutrition in the World 2020.
[42] Wang, X., Kang, H., Zhou, H., Au, W., Wang, M. Y., e Chen, C. 2023. Development and Evaluation of a Robust Soft Robotic Gripper for Apple Harvesting. Computers and Electronics in Agriculture, 204:107552.
[43] Wojke, N., Bewley, A., e Paulus, D. 2017. Simple Online and Realtime Tracking with a Deep Association Metric. In 2017 IEEE International Conference on Image Processing (ICIP), pages 3645–3649. IEEE.
[44] World Health Organization. 2019. Increasing Fruit and Vegetable Consumption to Reduce the Risk of Non-communicable Diseases. e-Library of Evidence for Nutrition Actions (eLENA).
[45] Yu, X., Fan, Z., Wang, X., Wan, H., Wang, P., et al. 2021. A Lab-Customized Autonomous Humanoid Apple Harvesting Robot. Computers & Electrical Engineering, 96:107459.
[46] Zhang, K., Lammers, K., Chu, P., Li, Z., e Lu, R. 2021. System Design and Control of an Apple Harvesting Robot. Mechatronics, 79:102644.

APÊNDICE

Tabela V: Todos os resultados dos modelos de movimentação do drone do Componente 3

Arquitetura	***Função de Ativação***	***Acurácia (%)***	***Movimentos (média)***	***Segundos (média)***
3	*SELU*	100	37,95	6,25
3	*Leaky-ReLU*	100	38,10	6,13
3	*Tanh*	100	38,15	6,49
2	*SELU*	100	40,05	6,48
2	*Leaky-ReLU*	100	40,35	6,52
3	*ReLU*	100	41,25	6,56
2	*ReLU*	100	41,70	6,68
2	*Tanh*	100	42,40	8,14
1	*SELU*	100	44,55	7,27
1	Sem ativador	100	62,20	9,99
1	*ReLU*	95	40,05	6,54
1	*Tanh*	95	40,11	6,87
3	*ELU*	95	40,11	6,53
1	*Leaky-ReLU*	95	41,32	6,82
2	Sem ativador	95	42,63	6,82
3	Sem ativador	90	39,56	6,34
1	*ELU*	90	42,28	6,89
2	*ELU*	90	42,61	7,08
2	*Sigmoid*	-	-	-
1	*Sigmoid*	-	-	-
3	*Sigmoid*	-	-	-

Tabela VI: Todos os resultados da sucção do Componente 3

Algoritmo	***Escalado***	***Componentes PCA***	***Acurácia Validação***	***Acurácia Teste***
Random Forest	Sim	-	98,41	98,37
Rede Neural	Sim	-	98,55	98,35
Random Forest	Não	-	98,43	98,33
Rede Neural	Sim	6	98,14	97,85
Random Forest	Não	6	97,91	97,62
Rede Neural	Não	6	97,40	97,29
Random Forest	Sim	6	97,46	97,19
Random Forest	Não	5	97,25	97,15
Decision Tree	Sim	-	97,06	96,96
Rede Neural	Sim	5	97,25	96,94
Random Forest	Não	4	96,96	96,78
Decision Tree	Não	-	96,88	96,78
Random Forest	Sim	5	97,00	96,71
Rede Neural	Não	-	96,90	96,67
KNN	Não	-	96,63	96,51

Continua na próxima página

Algoritmo	***Escalado***	***Componentes PCA***	***Acurácia Validação***	***Acurácia Teste***
Random Forest	Não	3	96,51	96,47
KNN	Não	6	96,24	96,42
Rede Neural	Não	5	96,55	96,42
Rede Neural	Sim	4	96,73	96,40
Random Forest	Sim	4	96,80	96,20
KNN	Sim	-	96,22	96,16
Decision Tree	Não	6	96,65	96,01
Rede Neural	Não	4	96,20	96,01
Rede Neural	Não	3	96,09	95,91
KNN	Sim	6	95,60	95,87
SVC	Sim	-	96,03	95,72
Decision Tree	Não	5	95,74	95,68
Logistic Regression	Sim	-	96,09	95,66
Random Forest	Sim	3	96,13	95,64
KNN	Não	5	95,85	95,64
Rede Neural	Sim	3	96,40	95,58
Decision Tree	Não	4	95,74	95,35
SVC	Não	6	95,47	95,00
KNN	Não	4	94,92	94,98
Decision Tree	Não	3	95,37	94,87
KNN	Sim	5	94,89	94,85
Decision Tree	Sim	6	95,60	94,83
SVC	Não	5	94,94	94,81
Decision Tree	Sim	5	94,42	94,79
SVC	Sim	6	95,25	94,73
Decision Tree	Sim	4	94,56	94,61
KNN	Sim	4	94,50	94,46
SVC	Não	3	94,46	94,40
SVC	Não	4	94,34	94,34
SVC	Sim	5	94,54	94,26
KNN	Não	3	94,03	93,90
KNN	Sim	3	93,61	93,80
Decision Tree	Sim	3	94,01	93,76
Logistic Regression	Sim	6	94,23	93,66
SVC	Sim	4	93,76	93,37
Naive Bayes	Não	-	92,29	92,79
Naive Bayes	Sim	-	92,29	92,79
SVC	Não	-	92,70	92,19
SVC	Sim	3	91,94	91,96
Rede Neural	Não	2	91,75	91,75
Rede Neural	Sim	2	91,34	91,49
Random Forest	Não	2	91,63	91,44

Continua na próxima página

Algoritmo	***Escalado***	***Componentes PCA***	***Acurácia Validação***	***Acurácia Teste***
Rede Neural	Não	1	91,19	91,36
Naive Bayes	Sim	3	90,99	90,89
Naive Bayes	Não	3	90,70	90,70
Naive Bayes	Não	5	90,70	90,68
Random Forest	Sim	2	90,84	90,58
Naive Bayes	Não	6	90,57	90,41
Naive Bayes	Não	1	90,39	90,39
Naive Bayes	Não	2	90,16	90,04
Naive Bayes	Sim	4	90,24	89,89
Naive Bayes	Não	4	89,98	89,77
SVC	Não	1	89,67	89,65
SVC	Não	2	89,44	89,52
Logistic Regression	Não	-	89,44	89,44
Logistic Regression	Não	6	89,29	89,32
Naive Bayes	Sim	5	89,58	89,17
Naive Bayes	Sim	6	89,27	88,65
Rede Neural	Sim	1	88,74	88,63
Decision Tree	Não	2	88,84	88,49
SVC	Sim	2	88,55	88,22
Logistic Regression	Não	5	88,22	88,22
Naive Bayes	Sim	1	88,59	88,16
SVC	Sim	1	87,99	87,91
Decision Tree	Sim	2	87,85	87,41
KNN	Não	2	86,38	86,92
Logistic Regression	Sim	5	86,81	86,79
Random Forest	Não	1	86,44	86,79
Logistic Regression	Sim	4	86,81	86,77
Decision Tree	Não	1	86,09	86,57
Naive Bayes	Sim	2	87,49	86,44
Logistic Regression	Não	4	86,25	86,22
KNN	Sim	2	85,45	85,93
Logistic Regression	Sim	3	84,48	85,04
KNN	Não	1	84,19	84,25
Random Forest	Sim	1	83,22	83,10
Decision Tree	Sim	1	83,17	83,10
Logistic Regression	Não	3	83,17	82,85
KNN	Sim	1	79,39	79,67
Logistic Regression	Não	2	66,95	65,76
Logistic Regression	Sim	2	60,21	59,14
Logistic Regression	Não	1	39,89	40,52
Logistic Regression	Sim	1	31,91	32,09

Fim da tabela