FOLHAR - EXPLORANDO FOLHAS DE ÁRVORES COM REALIDADE AUMENTADA

Bruno Geisler Vigentas, Dalton Solano dos Reis – Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

bvigentas@furb.br, dalton@furb.br

Resumo: *Este artigo apresenta o processo de desenvolvimento e testes de um aplicativo que tem como objetivo auxiliar as pessoas no conhecimento de folhas de árvores por intermédio da Realidade Aumentada Imersiva. O aplicativo foi desenvolvido na plataforma Unity utilizando a linguagem de programação C# em conjunto com os packages Barracuda e AR Foundation. A aplicação faz a detecção do formato da folha da árvore sobre um fundo de cor branca e renderiza uma versão desse formato com Realidade Aumentada, permitindo o usuário a interagir com esse objeto e aprender mais sobre a folha. Para avaliação da ferramenta foram feitos testes funcionais e avaliações com uma professora e um aluno do curso de Ciências Biológicas da Universidade Regional de Blumenau. Com os resultados obtidos, a aplicação se mostrou capaz de cumprir com seu objetivo.*

Palavras-chave*: Realidade aumentada. Folhas de árvores. Detecção de objetos.*

1 INTRODUÇÃO

Os professores possuem várias maneiras de diversificar suas aulas de forma que seus alunos aprendam os conteúdos propostos da melhor maneira, uma destas são as aulas de campo. As aulas de campo no ensino da biologia são práticas que propiciam aos alunos a oportunidade de levar para fora da sala de aula o conteúdo estudado, transcendendo assim as barreiras do ambiente escolar para a realidade e possibilitando ao aluno um novo meio de aprendizagem (SOUSA *et al.*, 2016). Como Araujo *et al.* (2015) demonstram em sua pesquisa realizada com alunos antes e depois da execução de aulas de campo, essas despertam interesse e desenvolvimento cognitivo, proporcionando uma maior relação com o conteúdo teórico. Nas perguntas da pesquisa foi possível notar um melhor desempenho dos estudantes e, por consequência, um maior aproveitamento dos conteúdos ministrados pelos professores, tendo uma reflexão na melhoria da aprendizagem (ARAUJO *et al.*, 2015).

De acordo com Campos (2012), as aulas de campo representam muito mais do que uma simples visita ao meio ambiente. Com elas os alunos são capazes de compreender a dinâmica do ecossistema que os rodeia, instruindo-se sobre sua fauna e flora local tendo maior ciência sobre sua conservação. Tal atividade permite aos alunos a exploração de conceitos, procedimentos e atitudes que se tornam de grande valia para programas de educação ambiental (VIVEIRO; DINIZ, 2009). Os ambientes fora da sala de aula aguçam a mente dos estudantes e fazem com que eles tenham vontade de aprender, dado que se caracterizam como espaços estimulantes que, quando bem aproveitados, se tornam um relevante cenário para a aprendizagem, preenchendo lacunas que vão sendo criadas ao decorrer do ensino na sala de aula. (CARBONNEL, 2002, apud SOUSA *et al.*, 2016). As aulas de campo são oportunidades nas quais os alunos podem descobrir novos ambientes fora da sala de aula, incluindo a observação e o registro de imagens (MORAIS; PAIVA, 2009).

Algumas ferramentas tecnológicas são capazes de ajudar os alunos no âmbito da observação e conhecimento dos elementos do campo. Uma dessas ferramentas é o PlantSnap, que permite ao usuário a identificação de forma precisa e instantânea de mais de 600 mil espécies a partir de fotos de plantas e árvores tiradas pelo próprio aplicativo, proporcionando ainda pequenas interações com o uso de Realidade Aumentada (PLANTSNAP, 2020).

Como Schmalstieg e Höllerer (2016) comentam, a Realidade Aumentada promete a criação automática e direta entre o mundo físico e a informação eletrônica, possibilitando com que essa informação pareça parte do mundo real na percepção do usuário. Na visão de Azuma (1997, p. 03, tradução nossa) "A Realidade Aumentada melhora a percepção e interação do usuário com o mundo real. Os objetos virtuais mostram informações que o usuário não consegue detectar diretamente com seus próprios sentidos. A informação transmitida pelos objetos virtuais ajuda o usuário a realizar tarefas do mundo real". Como Zorzal e Kirner (2005) relatam, a tecnologia de Realidade Aumentada na educação tem o potencial de enriquecer os materiais didáticos, estimulando o aluno a visualizar, conhecer e explorar os conteúdos ministrados pelos professores, possibilitando um aprendizado interativo e dinâmico.

Diante do contexto apresentado, este trabalho tem como objetivo auxiliar no conhecimento de folhas de árvores, por intermédio da Realidade Aumentada. Os objetivos específicos são: utilizar as folhas das árvores como marcadores para apresentação do conteúdo em Realidade Aumentada e analisar a eficácia do aplicativo com usuários da área da biologia.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção expõe os principais aspectos que fundamentaram a implementação do aplicativo. Serão apresentados conceitos e ferramentas utilizados na construção, como Realidade Aumentada, Morfologia das Folhas, AR Foundation e Barracuda, bem como serão apresentados alguns trabalhos que se relacionam com a proposta.

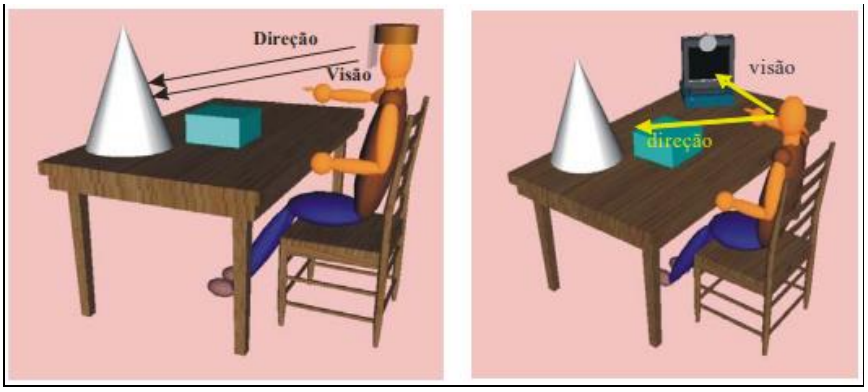
2.1 REALIDADE AUMENTADA

Segundo Azuma (2017, p. 01, tradução nossa), "Realidade Aumentada (AR) é uma experiência imersiva que sobrepõe objetos virtuais 3D sobre a visão direta de um usuário entorno do ambiente real, gerando a ilusão de que esses objetos virtuais existem naquele espaço". Através da Realidade Aumentada é possível estender as fronteiras do mundo real, complementando este com informações virtuais que podem auxiliar os usuários em variadas áreas. Como Kirner *et al*. (2006) relatam, a Realidade Aumentada se beneficiou graças ao avanço da multimídia, que foi influenciada positivamente pelo aumento de banda das redes e permitiu a transferência de imagens e outros fluxos de informação com eficiência, e pelo desenvolvimento da Realidade Virtual, proporcionado pelo incremento na potência dos computadores. Sem essas evoluções, as principais características da Realidade Aumentada descritas por Azuma (2001) não seriam possíveis, como a combinação do mundo real e virtual executando de forma interativa em tempo real.

A Realidade Aumentada é apresentada ao usuário com o auxílio de equipamentos tecnológicos. Estes podem ser capacetes *Head Mounted Display* (HMD) com câmeras acopladas, que filmam a visão do usuário e permitem mesclar o mundo real e virtual trazendo a informação virtual à tela que é vista através do capacete. Assim como também podem ser aplicativos para celulares ou computadores, onde a câmera do celular ou webcam monitoram o ambiente real e o conteúdo virtual é mesclado a ele e exposto na tela do celular ou no monitor do computador (KIRNER *et al*, 2006).

Kirner *et al* (2006) comentam que a Realidade Aumentada pode ser classificada de duas maneiras segundo a forma com que o usuário observa o mundo mesclado através dos dispositivos. Quando o usuário vê a mescla diretamente olhando para a posição real da cena por vídeo, a Realidade Aumentada é de visão direta (Imersiva). Já quando o usuário a observa através de um monitor, não olhando diretamente para o local ao qual os componentes virtuais estão sendo projetados, a Realidade Aumentada é de visão indireta (Não imersiva). As duas modalidades são ilustradas na Figura 1.

Figura 1 - Realidade aumentada de forma direta com HMD e de forma indireta, através de monitor, respectivamente



Fonte: Kirner *et al* (2006, p. 28).

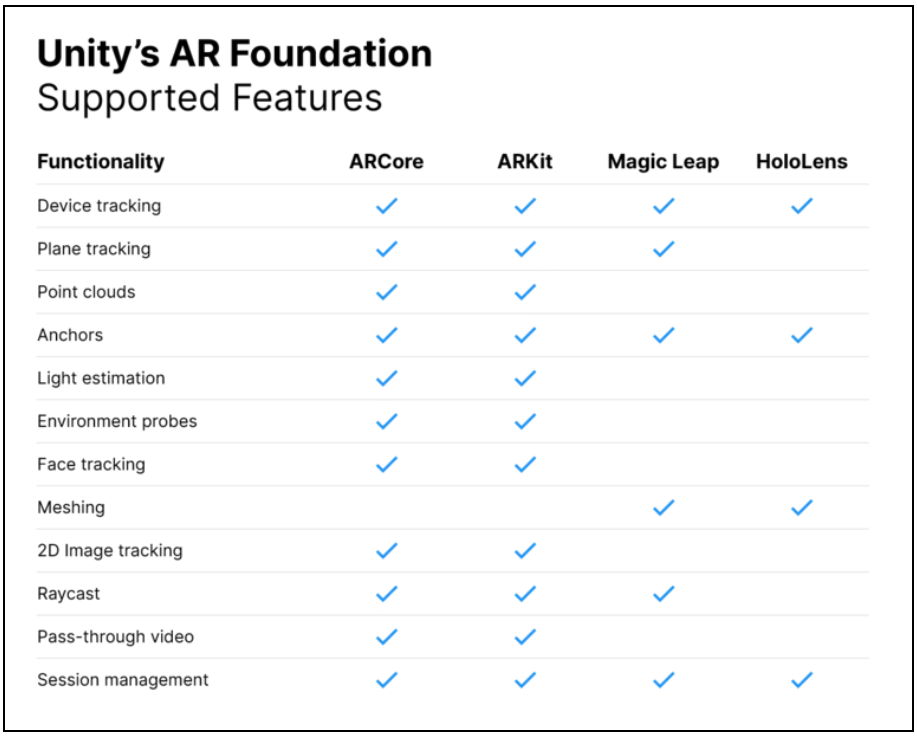
Como Kirner e Siscoutto (2007) relatam, a escolha da visão a ser usada pode interferir na meta da Realidade Aumentada, que é criar um ambiente tão realista ao ponto em que o usuário não consiga distinguir as diferenças entre o real e o virtual.

2.2 AR FOUNDATION

O AR Foundation é um *framework* criado com o intuito de facilitar a criação de aplicações em Realidade Aumentada dentro do Unity para diversas plataformas, como Android e iOS. A biblioteca apresenta uma série de interfaces que fazem com que as bibliotecas de Realidade Aumentada para outras plataformas, como o `ARCore` para o Android e o `ARKit` para o iOS, funcionem da mesma maneira. Seguindo os contratos inferidos na interface e possibilitando que o desenvolvedor faça apenas a chamada de métodos comuns, que executam conforme a biblioteca correspondente ao da plataforma que a `build` for gerada posteriormente (UNITY, 2021b).

O *framework* não inclui funções próprias, ao invés disso define uma API multiplataforma para trabalhar com as funções indispensáveis das principais bibliotecas de Realidade Aumentada no mercado. Entre essas funções definidas, ele conta com funções para *Plane Tracking*, para fazer a detecção de superfícies horizontais e verticais; *Anchors*, que age como uma posição arbitrária rastreada pelo dispositivo; *Device Tracking*, que objetiva rastrear a posição e orientação do dispositivo no ambiente físico; *Raycast*, que consulta os arredores de planos detectados e diversas outras funções que podem ser visualizadas na Figura 2 (UNITY, 2021a). A configuração de um projeto simples com AR Foundation é apresentada no APÊNDICE B.

Figura 2 - Lista de funções presentes no AR Foundation



Unity's AR Foundation
Supported Features

Functionality	ARCore	ARKit	Magic Leap	HoloLens
Device tracking	✓	✓	✓	✓
Plane tracking	✓	✓	✓	
Point clouds	✓	✓		
Anchors	✓	✓	✓	✓
Light estimation	✓	✓		
Environment probes	✓	✓		
Face tracking	✓	✓		
Meshing			✓	✓
2D Image tracking	✓	✓		
Raycast	✓	✓	✓	
Pass-through video	✓	✓		
Session management	✓	✓	✓	✓

Fonte: Unity (2021).

2.3 BARRACUDA

Como descrito por Fleck *et. al.* (2016), redes neurais artificiais são sistemas projetados para simular o processo de aprendizado de um organismo vivo. Para realizar isso, Matos *et. al.* (2020) relatam que o aprendizado nestas redes se dá por meio de técnicas matemáticas estruturadas em algoritmos que adquirem experiência e conhecimento a partir de dados de entrada.

O pacote Barracuda traz uma rede neural artificial leve e multiplataforma para dentro do Unity, permitindo a utilização de redes pré treinadas em aplicações Unity desktop, mobile e para consoles (UNITY, 2019). O pacote Barracuda utiliza dos compiladores Unity Compute Shader, que permitem que programas sejam executados em placas de vídeo (UNITY, 2021c), e Unity Burst, que realiza a compilação para código nativo altamente optimizado (UNITY, 2020b). Desta forma o Barracuda permite com que se consiga executar as redes neurais artificiais rapidamente tanto em GPU como em CPU (UNITY, 2019).

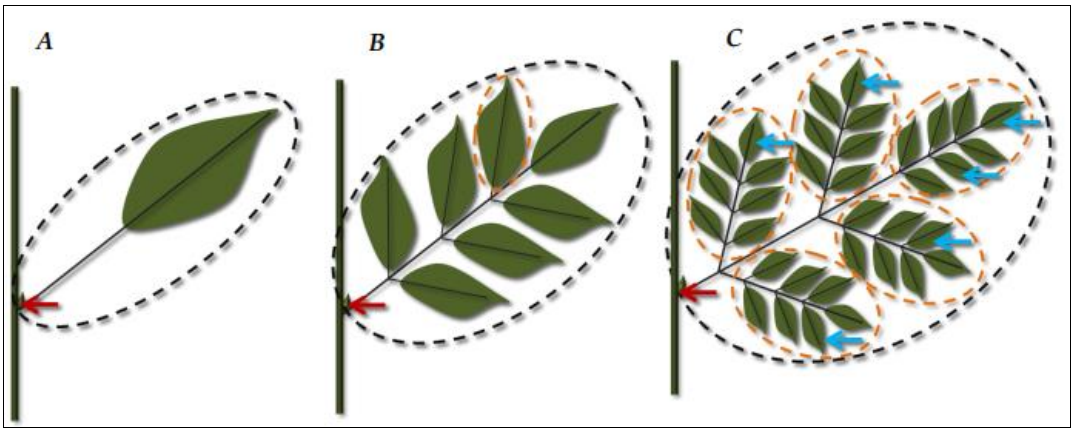
O fluxo de importação de redes neurais artificiais do Barracuda foi criada em cima do formato de modelos ONNX (Open Neural Network Exchange), que possibilita o intercâmbio de vários frameworks de Machine Learning (MICROSOFT, 2019), como Pytorch, TensorFlow e Keraz (UNITY, 2021d). Em virtude disso, uma grande variedade de modelos e arquiteturas são suportadas pelo Barracuda, tais como os classificadores de imagens MobileNet, detector de objetos Tiny YOLO, todos os modelos ML-Agents entre outros. (UNITY, 2020a).

2.4 MORFOLOGIA DAS FOLHAS

Almeida e Almeida (2018) definem as folhas como um apêndice lateral ao caule distribuídas em um intervalo regular, e explicam que a grande diferença morfológica e fisiológica que afeta sobretudo as folhas, ocorre devido ao poder da vegetação de se adaptar a diferentes condições e ambientes. Essa variedade morfológica é foco de pesquisas para o reconhecimento das espécies, que segundo Wiggers (2008), é de grande importância para estudos taxonômicos, para o auxílio em trabalhos científicos sobre a flora e fauna, para a descoberta de plantas medicinais e tóxicas, e definir as espécies presentes em um inventário. Para uma identificação precisa, Almeida e Almeida (2018) relatam que alguns componentes das folhas devem ser observados, tais como limbo, pecíolo, bainha, pulvino e nervuras, bem como a observação de suas estruturas, se são simples ou compostas.

Almeida e Almeida (2018) salientam que a maioria das folhas das plantas com sementes são constituídas por duas partes nitidamente definidas, o pecíolo e o limbo. Pecíolo sendo a haste que proporciona movimento à folha, servindo como uma ponte entre o limbo e o caule ou entre o limbo e a bainha, outro componente da folha. Limbo, também chamado de lâmina foliar, é a folha propriamente dita, onde se encontram as nervuras e onde podem ocorrer variedades morfológicas. Uma dessas variedades é a composição do limbo, que pode ser observada na Figura 3, em que A representa um limbo simples, B um limbo composto e C um limbo recomposto. (ALMEIDA; ALMEIDA, 2018).

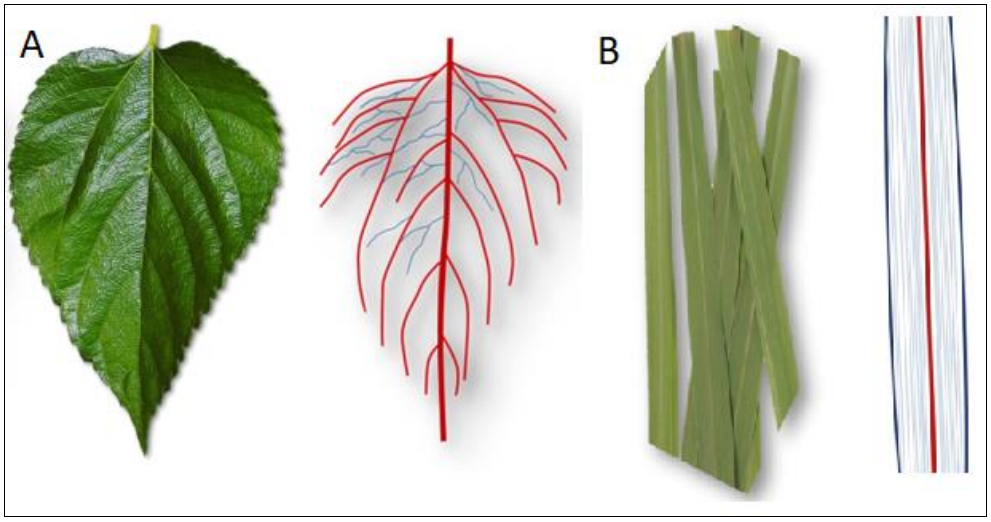
Figura 3 - Representação esquemática do limbo



Fonte: Almeida e Almeida (2018, p.58).

Presentes no limbo, as nervuras das folhas, responsáveis por fazerem a distribuição de água, nutrientes, compostos orgânicos, além de fornecerem sustentação, flexibilidade e resistência a ela, também servem de instrumento para a distinção das folhas, por possuírem diversos formatos. Como exemplo desses formatos temos a paralelinérvea, nome atribuído a nervura quando duas ou mais nervuras secundárias seguem em paralelo, e a peninérvea, que ocorre quando as nervuras secundárias se iniciam ao longo da nervura principal de maneira regular e espaçosa (ALMEIDA; ALMEIDA, 2018). Exemplos podem ser visualizados na Figura 4.

Figura 4 - Nervuras peninérvea e paralelinérvea, respectivamente



Fonte: Almeida e Almeida (2018, p.62).

2.5 TRABALHOS CORRELATOS

A seguir são apresentados dois trabalhos acadêmicos e um produto que desenvolvem características semelhantes aos objetivos do trabalho. O primeiro detalhado no Quadro 1, se trata de um aplicativo móvel que permite o cadastro e classificação de espécies baseadas no registro de folhas de plantas através de imagens capturadas a partir do aplicativo (BORTOLON, 2014). O segundo, apresentado no Quadro 2, é um aplicativo baseado em Realidade Aumentada que objetiva auxiliar a escolha e prover informações sobre o cultivo de plantas (OLIVEIRA; PRADO, 2018). Por último, o Quadro 3 demonstra um aplicativo capaz de identificar 90% das espécies de plantas, folhas, cogumelos, suculentas e cactos a partir da câmera do celular em segundos, contando ainda com interações com o uso de Realidade Aumentada (PLANTSNAP, 2020

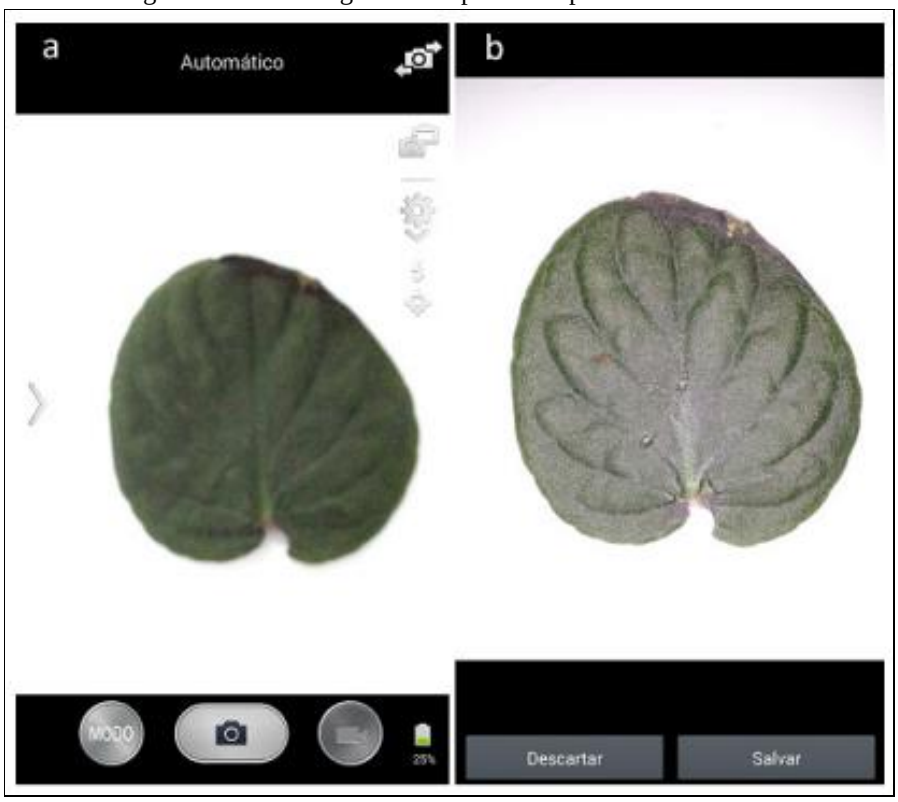
Quadro 1 – Plantarum

Referência	Bortolon (2014)
Objetivos	Disponibilizar um aplicativo Android que permite interagir com um *webservice* para realizar o cadastro e classificação de espécies de plantas.
Principais funcionalidades	O aplicativo permite ao usuário capturar fotos com o celular e as enviar para o servidor escolhendo se deseja fazer a classificação, cadastrar uma amostra de uma espécie já existente na base de dados ou adicionar uma nova.
Ferramentas de desenvolvimento	A parte do aplicativo foi desenvolvida em Java voltado para Android. Enquanto a parte do servidor foi desenvolvido em C# com o framework .Net 4.0 e o modelo de desenvolvimento Windows Communication Foundation (WCF), fazendo ainda a integração com a API Plantarum.
Resultados e conclusões	Bortolon (2014) relata que foram realizados dois cenários de testes, no primeiro com a utilização de um banco de imagens o sistema não classificou apenas 2 imagens e atribuiu falso positivo a outras quatro, em um total de 32. Após a adição dessas imagens sem classificação e com falso positivo a base de dados, como resultado apenas uma delas acusou falso positivo, tendo reconhecido todas as 32 imagens. No segundo cenário de teste, foram utilizadas fotos de espécies locais. O resultado do teste nesse cenário identificou apenas um falso positivo. No final Bortolon (2014) aponta algumas limitações no trabalho, como a necessidade de um alto contraste entre a folha e o fundo que limita algumas detecções.

Fonte: elaborado pelo autor.

A Figura 5 apresenta a tela de registro de espécie, onde o usuário pode capturar uma foto de uma nova espécie e cadastrá-la na base de dados para futuras detecções.

Figura 5 - Tela de registro de espécie no aplicativo Plantarum



Fonte: Bortolon (2014, p. 53)

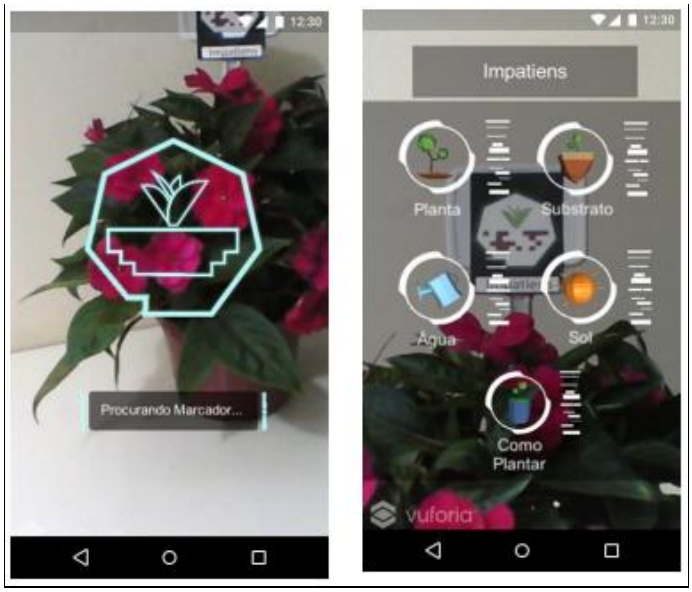
Quadro 2 - Gaia

Referência	Gaia (2018)
Objetivos	Desenvolver um aplicativo Android com uso da Realidade Aumentada visando oferecer informações necessárias e confiáveis para auxiliar pessoas na escolha de flores e plantas ornamentais.
Principais funcionalidades	Gaia utiliza a Realidade Aumentada para trazer informações ao mundo real em forma de textos e exibe informações para auxiliar o cliente na compra de plantas, como a quantidade diária de água e luz necessária além de trazer um passo-a-passo 3D de como plantar a espécie.
Ferramentas de desenvolvimento	O sistema foi desenvolvido no Unity, com a linguagem C#, e a *software development kit* (SDK) Vuforia para parte da Realidade Aumentada em conjunto com marcadores do tipo VuMarks.
Resultados e conclusões	Oliveira e Prado (2018) relatam que completam os objetivos iniciais do projeto, entregando ao usuário acesso às informações necessárias para tomada de decisão quanto à escolha das plantas ornamentais, enfatizando a satisfação no uso da Realidade Aumentada para melhorar o mundo real através da inserção de informações geradas virtualmente, assim como descrevem algumas limitações encontradas, como a dificuldade de funcionamento do aplicativo em dispositivos com câmeras de baixa qualidade ou em condições de luminosidade ruim.

Fonte: elaborado pelo autor.

A Figura 6 demonstra o marcador correspondente a uma planta durante e após a detecção, onde o cliente tem acesso a diversas informações sobre a planta, como a quantidade diária necessária de água e luz, um guia de como fazer o plantio e qual o substrato ideal para a espécie.

Figura 6 - Informações sobre a planta com o aplicativo GAIA



Fonte: Oliveira e Prado (2018, p. 21

Quadro 3 - PlantSnap

Referência	PlantSnap (2020)
Objetivos	Tem como objetivo a identificação de plantas de forma rápida.
Principais funcionalidades	Além de realizar o reconhecimento de espécies, o aplicativo também conta com algumas interações por meio de Realidade Aumentada e uma seção social, onde os usuários podem compartilhar suas descobertas na natureza com outras pessoas.
Ferramentas de desenvolvimento	Utiliza um algoritmo de aprendizado de máquina de código fechado treinado com mais de 250 mil imagens.
Resultados e conclusões	Possibilitando o reconhecimento de mais de 620 mil espécies de plantas com uma acurácia de 94%, o aplicativo conta com mais 32 milhões de instalações e notas 4.5 e 3.6 (escala máxima 5) nas lojas de aplicativo App Store e Play Store, respectivamente.

Fonte: elaborado pelo autor.

O aplicativo PlantSnap (2020) utiliza a Realidade Aumentada de uma maneira mais simples, identificando se a imagem se trata de uma flor ou folha. Identificado uma folha, a aplicação demonstra todas as âncoras encontradas, conforme Figura 7, para que o usuário selecione uma delas para a renderização do objeto 3D, demonstrando o processo de fotossíntese.

Figura 7 - Âncoras para renderização de objetos 3D

Fonte: digitalizado do aplicativo PlantSnap (2020).

3 DESCRIÇÃO DO APLICATIVO

Este capítulo tem como objetivo apresentar os aspectos mais importantes relacionados ao desenvolvimento do aplicativo, bem como as técnicas e ferramentas utilizadas. Ela é dividida em três seções, na 3.1 é apresentada a especificação do aplicativo, a 3.2 apresenta uma visão geral do sistema, mostrando o seu funcionamento e a forma de uso, e, por fim, na 3.3 são demonstrados os principais pontos de implementação para a construção da aplicação.

3.1 ESPECIFICAÇÃO

Nesta seção estão presentes os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF), bem como um diagrama de casos de uso para melhor entendimento do processo que ocorre dentro da aplicação.

Quadro 4 - Requisitos Funcionais

Requisitos Funcionais
RF01: permitir ao usuário iniciar o *scan* a partir de um menu na tela inicial
RF02: permitir ao usuário utilizar a câmera do celular para realizar o reconhecimento do formato da folha
RF03: renderizar um modelo 3D da folha
RF04: exibir informações sobre a folha detectada
RF05: permitir ao usuário iniciar o *scan* no modo quiz

Fonte: elaborado pelo autor.

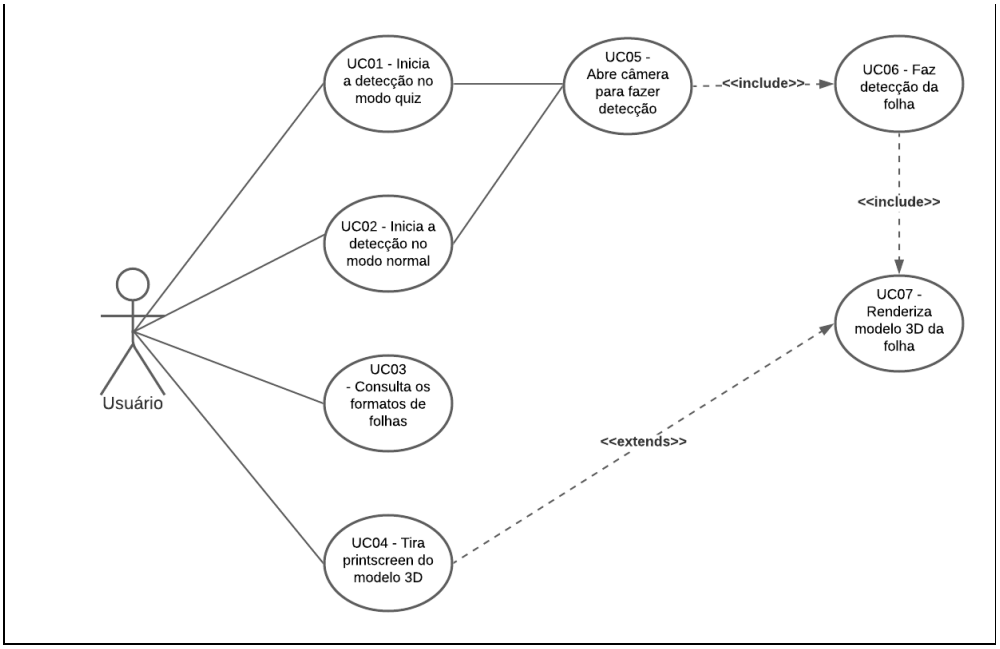
Quadro 5 - Requisitos Não Funcionais

Requisitos Não Funcionais
RNF01: permitir ao usuário capturar uma foto do conteúdo sendo mostrado em sua tela e salvar em sua galeria
RNF02: utilizar a folha da árvore como marcador para ancoragem do conteúdo virtual
RNF03: utilizar o ambiente de desenvolvimento Unity
RNF04: ser desenvolvido para plataforma Android
RNF05: utilizar a biblioteca OpenCV

Fonte: elaborado pelo autor.

Os requisitos e o funcionamento do sistema podem ser melhores entendidos através do Diagrama de Classes da Figura 8.

Figura 8 - Diagrama de Casos de Uso



Fonte: elaborado pelo autor.

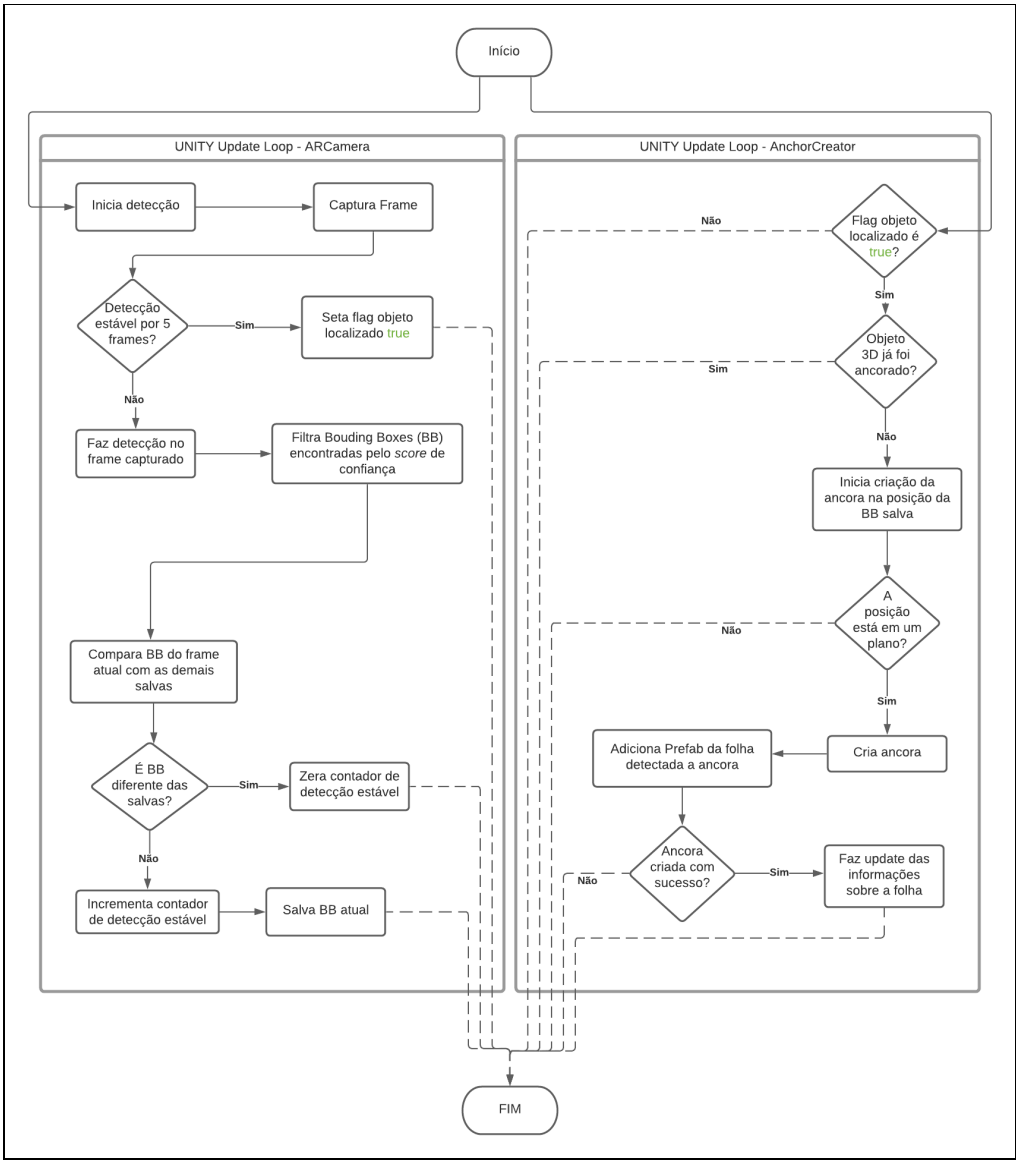
O aplicativo conta com um ator, o `Usuário`. Esse ator pode realizar algumas ações, representadas nos casos de uso `UC01`, `UC02`, `UC03` e `UC04`. O usuário pode iniciar a aplicação de duas maneiras, uma representada através do `UC02 – Inicia a detecção no modo normal`, onde o modelo 3D da folha é renderizado com todas as informações sobre ela. E outra através do `UC01 – Iniciar a aplicação no modo quiz`, onde o modelo 3D renderizado conta com lacunas nas informações, que devem ser preenchidas pelo usuário para a realização de um teste sobre seus conhecimentos sobre folha identificada. Outra ação disponível para o usuário é representada no caso de uso `UC03 – Consulta os formatos de folhas`, que o permite consultar os formatos de folhas que são detectáveis pela aplicação.

O caso de uso `UC05 – abre câmera para fazer detecção`, faz com que a câmera traseira do celular seja acionada para iniciar a detecção. Com isso, o caso de uso `UC06 – Faz detecção da folha` inicia-se e nele é executada a rotina que identifica qual é o formato da folha que está sendo capturado através da câmera. Em seguida, o caso de uso `UC07 – Renderiza modelo 3D da folha` é executado para renderizar a folha com suas informações na tela do aparelho. Com o `UC07` executado, o ator `Usuário` pode então optar por executar mais uma ação, a `UC04 – Tirar printscreen do modelo 3D`, que objetiva tirar uma foto da tela do aparelho e salvar em sua galeria.

Para se aprofundar ainda mais nas duas principais rotinas da aplicação, detecção e criação do objeto 3D no mundo real, foi criado o fluxograma da Figura 9. O fluxograma apresenta as rotinas encapsuladas no `Loop Update` dos *scripts*

C# que executam no Unity. O quadro à esquerda demonstra o fluxo de detecção de objeto, navegando sobre seus caminhos de sucesso e de falha. Já o da direita apresenta o fluxo de criação do objeto 3D, que depende de elementos do loop de detecção para uma execução com sucesso.

Figura 9 - Fluxograma da detecção e criação do objeto 3D



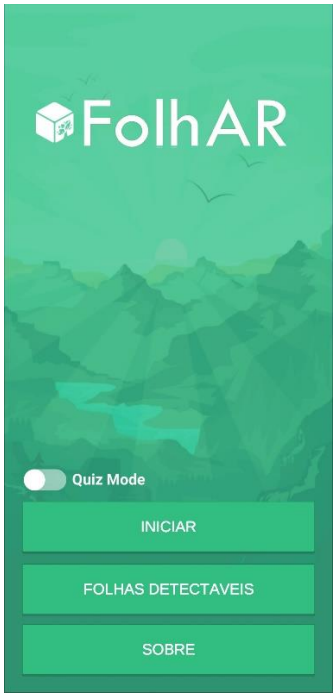
Fonte: elaborado pelo autor.

3.2 VISÃO GERAL

O principal objetivo do aplicativo é auxiliar a busca de conhecimentos sobre folhas de árvores por intermédio da Realidade Aumentada, visando realizar a detecção de vários formatos e, a partir desta detecção, realizar a ancoragem de um modelo 3D da folha com informações pertinentes a ela. Nesta seção é apresentada uma visão geral do aplicativo para possibilitar a observação de como ele foi projetado para atender aos objetivos propostos.

A Figura 10 mostra a tela inicial do aplicativo, o menu inicial. Esta tela conta com 4 botões; um que quando selecionado altera a aplicação para o `Quiz Mode`, outro para iniciar a detecção, outro para visualizar quais são os tipos e formatos de folhas detectadas pela aplicação e, por fim, um botão para constatar algumas informações sobre o desenvolvimento do aplicativo.

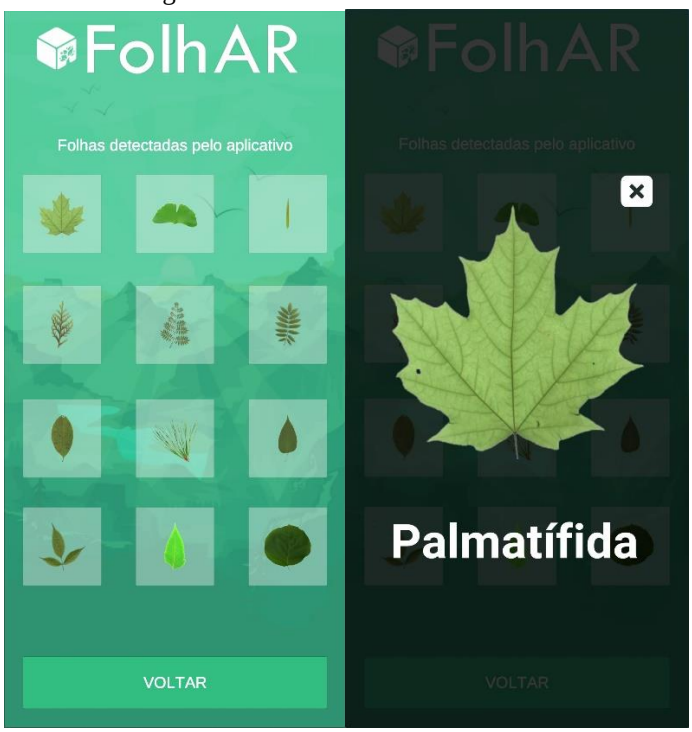
Figura 10 - Menu Inicial



Fonte: elaborado pelo autor.

Por existirem formatos e tipos de folhas demasiadamente vastos, a aplicação atua com um universo reduzido para um reconhecimento com maior assertividade. Para ajudar o usuário que está utilizando o aplicativo a identificar quais são os formatos de folhas detectáveis foi elaborada uma tela, que é ativada ao clicar-se o botão `Folhas Detectáveis`. A tela, que é demonstrada na Figura 11, exibe todos os tipos de folhas detectáveis em forma de botões, que quando pressionados ativam uma tela `pop up` com uma imagem detalhada e o nome do formato da folha.

Figura 11 - Tela de Folhas Detectáveis



Fonte: elaborado pelo autor.

Ao clicar no botão `Iniciar`, é aberta a tela principal da aplicação onde ocorre a detecção e, posteriormente, a renderização do objeto 3D da folha identificada. Nessa tela está presente a imagem obtida através da câmera do celular, assim como alguns botões, um para voltar à tela inicial e outro para reiniciar a detecção.

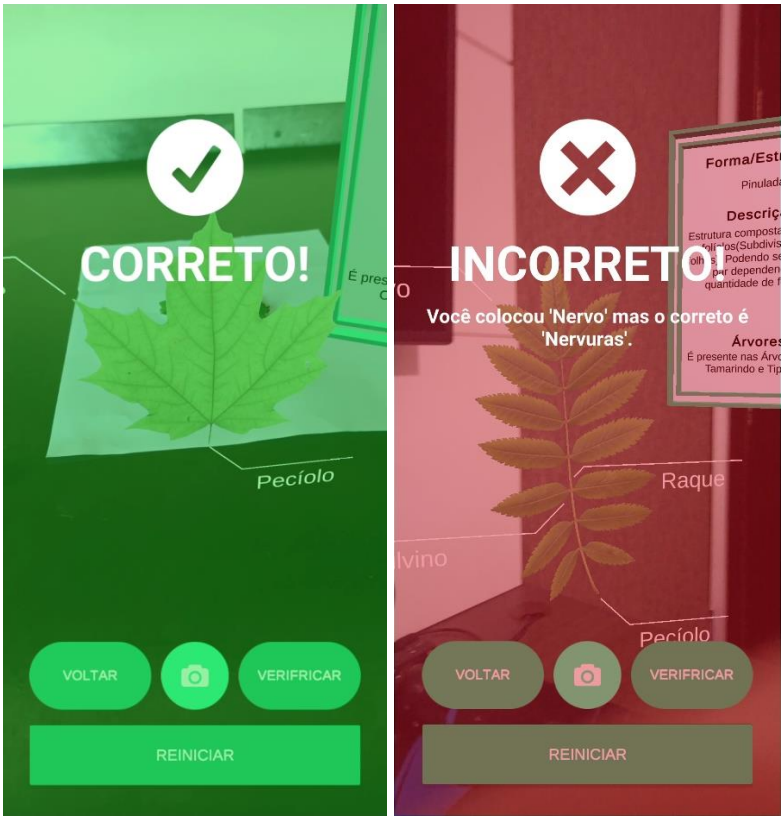
O processo de detecção sempre ocorre até que uma folha seja detectada e seu objeto 3D correspondente seja renderizado. Quando uma folha entrar no campo de visão da câmera, a folha começara a ser detectada e um aviso será exibido na tela, instruindo o usuário a estabilizar a câmera para uma detecção rápida e precisa.

Figura 12 - Aviso instruindo o usuário

Fonte: elaborado pelo autor.

Após detectado o formato da folha, o usuário deverá movimentar o celular ao redor do ambiente, para que uma rede de `planos` seja encontrada na superfície para realização da ancoragem e renderização do objeto 3D, conforme é demonstrado na Figura 12. A renderização do objeto é feita conforme a opção `Quiz Mode` da tela inicial. Caso habilitada, o objeto 3D é apresentado com algumas informações sobre a folha em um quadro. Entretanto, os nomes dos componentes das folhas aparecem em branco, para que o usuário preencha o nome dos componentes para avaliar seus conhecimentos. Com os nomes preenchidos, o usuário pode então pressionar o botão `Verificar`, que fará a validação das lacunas preenchidas através de mensagens de erro ou de acerto, como pode ser observado na Figura 13.

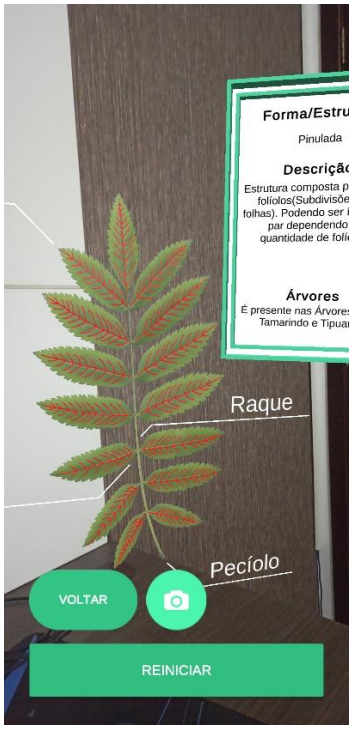
Figura 13 - Validação do Quiz Mode



Fonte: elaborado pelo autor.

Caso a opção de `Quiz Mode` não esteja habilitada, o objeto será renderizado com as lacunas já preenchidas e sem botão para realizar a validação. No entanto o usuário pode a qualquer momento clicar no botão `Voltar`, para ir à tela inicial e alterar a opção `Quiz Mode`. Ainda no objeto renderizado, o usuário pode clicar sobre os componentes da folha para realçá-los, possibilitando um melhor entendimento e visão sobre ele, conforme Figura 14, e caso se queira detectar uma nova folha, o botão `Reiniciar` deverá ser pressionado para que, o processo recomece.

Figura 14 - Destaque nas nervuras das folhas



Fonte: elaborado pelo autor.

3.3 IMPLEMENTAÇÃO

A aplicação foi desenvolvida na plataforma Unity através de scripts C# e com o uso de alguns *packages*, tais como o Barracuda, para a parte de detecção de imagem, e o AR Foundation, para realizar a função de Realidade Aumentada.

O Barracuda necessita de um modelo ONNX de aprendizado de máquina para realizar a detecção conforme os dados do modelo. Assim sendo, o primeiro passo do desenvolvimento consistiu-se em fazer a criação do modelo. Como não foi possível encontrar um dataset pronto contendo os formatos das folhas necessárias, foi organizado um dataset a partir da junção de três datasets diferentes para se ter uma variedade maior de imagens, sendo esses datasets o LeafSnap de Kumar *et al.* (2012), Flavia Dataset de Wu *et al.* (2007) e o dataset de Chouhan (2019). Efetuou-se a junção de algumas imagens de cada um desses datasets e realizou-se a rotulação dessas imagens conforme o formato de folha presente em cada uma. Este processo foi realizado com a ferramenta LabelImg. A utilização da ferramenta é melhor descrita no APÊNDICE C. O dataset foi então treinado para gerar um modelo de extensão `weights`, que foi posteriormente convertido para a extensão `onnx`, esperado pelo Barracuda.

O modelo ONNX gerado é importado para o Unity através do `ModelLoader.Load` do *package* Barracuda. Este método faz com que o modelo, que se encontra em formato binário, seja transformado em um objeto `Model`, esperado pelo método `GraphicsWorker.GetWorke`r do Barracuda, responsável por criar um objeto `Worker`. O objeto faz com que o modelo gerado seja quebrado em pequenas `tasks` que são agendadas para serem executadas na GPU ou na CPU do dispositivo.

A rotina de detecção é iniciada através do método `Update` da classe `ARCamera`. A rotina sempre verifica se a *flag* `isDetecting` é `true`, e caso ela não seja, o reconhecimento está apto a iniciar. Este controle é feito para que a aplicação não tenha mais de uma detecção rodando ao mesmo tempo, dado que o método `Update` é chamado uma vez por quadro e a rotina de detecção é executada de forma assíncrona, podendo demorar alguns quadros para finalizar. O bloco de código no **Erro! Fonte de referência não encontrada.**Quadro 6 é responsável por iniciar a detecção.

Conforme visível no Quadro 6, a primeira ação a ser tomada no método é modificar o valor da *flag* `isDetecting` para `true`. Posteriormente, o método `ProcessImage` é chamado e é responsável por capturar o quadro atual e redimensioná-lo para um tamanho melhor adaptado para a detecção. O resultado dessa execução, uma imagem, é então utilizado como parâmetro para o método `Detect` da classe `Detector` que retorna as `BoudingBoxes` encontradas na detecção. Por último, a *flag* `isDetecting` é alterada novamente para `false`, para que outras detecções possam ocorrer nos quadros seguintes.

Quadro 6 - Método responsável por iniciar a detecção

```
private unsafe void StartToDetect()
{
    this.isDetecting = true;
    StartCoroutine(ProcessImage(this.detector.IMAGE_SIZE, result =>
    {
        StartCoroutine(this.detector.Detect(result, boxes =>
        {
            this.boxOutlinesFromThisFrame = boxes;
            Resources.UnloadUnusedAssets();
            this.isDetecting = false;
        }));
    }));
}
```

Fonte: elaborado pelo autor.

No método `Detect` criado por Ashikhmin (2021) e modificado no trabalho, exibido no Quadro 7, é possível visualizar a criação do tensor (uma estrutura de dados) a partir da imagem do quadro, que será adicionada aos `inputs` da rede neural do Barracuda. Com os `inputs` configurados, o `Worker` inicia a detecção, e uma vez que ela termina, os resultados de duas *convulutional layers* diferentes da rede neural são convertidas para `BoudingBoxes` por intermédio do método `ParseOutputs`. Por fim, os resultados são unidos e filtrados no método `FilterBoudingBoxes`, que aplica a estratégia *Intersect Over Union* para encontrar `BoudingBoxes` que podem se tratar de um mesmo objeto (Quadro 8).

Quadro 7 - Método que faz a detecção

```
public IEnumerator Detect(Color32[] picture, System.Action<IList<BoundingBox>> callback)
{
    using (var tensor = BarracudaHelper.CreateTensorFromPicuture(picture, IMAGE_SIZE, IMAGE_SIZE))
    {
        var inputs = new Dictionary<string, Tensor>();
        inputs.Add(INPUT_NAME, tensor);
        yield return StartCoroutine(worker.StartManualSchedule(inputs));
        var output_l = worker.PeekOutput(OUTPUT_NAME_L);
        var output_m = worker.PeekOutput(OUTPUT_NAME_M);

        var results_l = ParseOutputs(output_l, MINIMUM_CONFIDENCE, params_l);
        var results_m = ParseOutputs(output_m, MINIMUM_CONFIDENCE, params_m);
        var results = results_l.Concat(results_m).ToList();

        var boxes = FilterBoundingBoxes(results, 1, MINIMUM_CONFIDENCE);
        callback(boxes);
    }
}
```

Fonte: elaborado pelo autor.

Quadro 8 - Método para detectar bouding boxes pertencentes ao mesmo objeto

```
private float IntersectionOverUnion(Rect boundingBoxA, Rect boundingBoxB)
{
    float areaA = CalculateArea(boundingBoxA);

    if (areaA <= 0)
        return 0;

    var areaB = CalculateArea(boundingBoxB);

    if (areaB <= 0)
        return 0;

    float intersectionArea = CalculateIntersectionArea(boundingBoxA, boundingBoxB);
    float unionArea = CalculateUnionArea(areaA + areaB, intersectionArea);
    return intersectionArea / unionArea;
}
```

Fonte: elaborado pelo autor.

Caso o mesmo objeto seja detectado por cinco quadros seguidos, a *flag* `localization` é setada para `true` e, com isso, o processo para criação do objeto 3D inicia-se na classe `AnchorCreator`. A primeira ação executada nesta classe é adquirir a `BoudingBox` com o maior *confidence score* encontrada na detecção. As coordenadas X e Y centrais desta `BoudingBox` são então extraídas para servirem como ponto de ancoragem para o objeto 3D. Contudo, antes disso a aplicação utiliza o `ARRaycastManager` do *package* AR Foundation para verificar se este ponto faz intersecção com algum plano através de um teste de *ray casting*. Caso faça, o método mostrado no Quadro 9 retorna uma lista denominada `resultHits`, com pontos onde a intersecção ocorreu.

Quadro 9 - Verifica se X e Y intersectam com um plano

```
private bool VerifyIfPointIntersectsWithPlanes(float x, float y, TrackableType trackableTypes)
{
    return m_RaycastManager.Raycast(new Vector2(x, y), resultHits, trackableTypes);
}
```

Fonte: elaborado pelo autor.

O primeiro `hit` da lista é transferido para o método `CreateAnchor`, que cria uma âncora através da utilização do método `AttachAnchor` do `ARAnchorManager`, utilizando como parâmetro o plano em que a intersecção ocorreu e a posição do `hit`. Esta âncora é originada com um `prefab` associado a ela como demonstrado no Quadro 10**Erro! Fonte de referência não encontrada.**, este `prefab` corresponde ao objeto 3D da folha que foi reconhecida e, portanto, o objeto 3D será renderizado juntamente com a âncora.

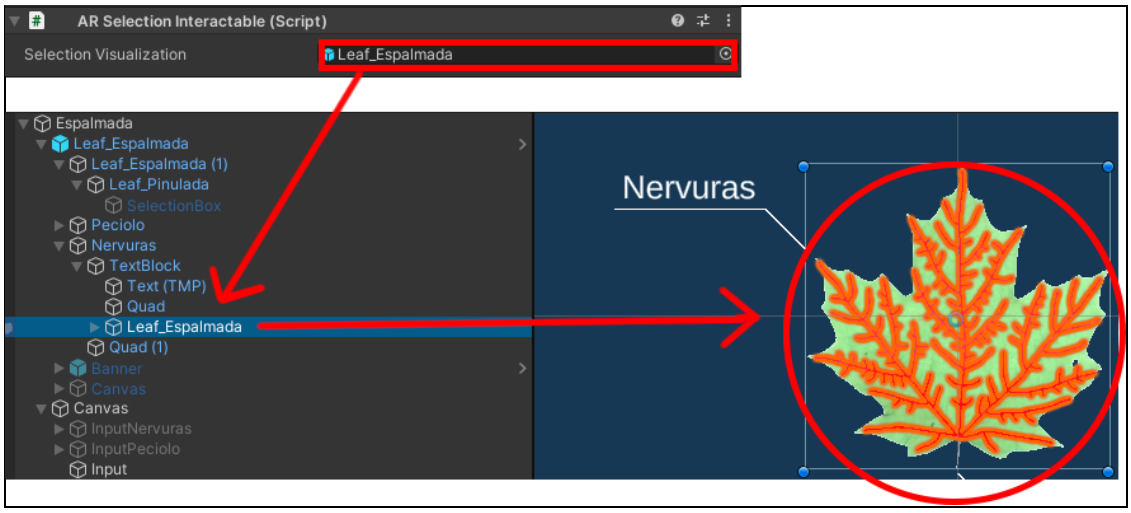
Quadro 10 - Criação da âncora e associação do prefab da folha

```
m_AnchorManager.anchorPrefab = prefab[dicPreFab.IndexOf(aRCamera.foundedLeafString)];
anchor = m_AnchorManager.AttachAnchor(plane, hit.pose);
```

Fonte: elaborado pelo autor.

A interação com os componentes da folha no objeto 3D se dá por meio do script `ARSelectionInteractable`, do AR Foundation. Adicionado o script a um `GameObject`, ele detecta quando um clique ocorre neste `GameObject` e é capaz de tomar uma série de ações, como por exemplo realizar a ativação de outro `GameObject` adicionado ao script, que no caso dos componentes das folhas, são objetos 3D que fazem o destaque do componente. Como por exemplo as nervuras, como é observado na Figura 15.

Figura 15 - Exemplo de script de seleção das nervuras



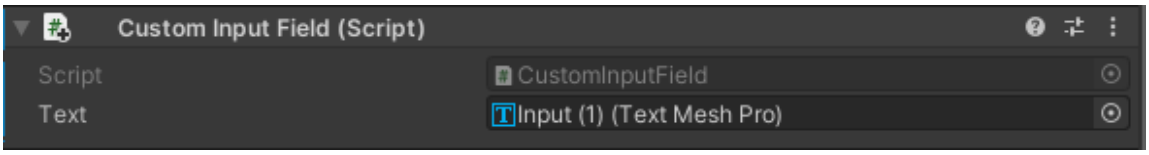
Fonte: elaborado pelo autor.

O *input* para o modo Quiz do aplicativo utiliza o mesmo *script* adicionado ao `GameObject` do componente da folha, mas com a diferença de que no *listener* do método `OnSelectEntered` (Figura 16) do *script* `ARSelectionInteractable`, o método `click` do script `CustomInputField` é chamado e tem como funcionalidade abrir o teclado do celular para receber o input do usuário e armazená-lo em um elemento do tipo `TextMeshPro` (Figura 17) que é passado como atributo do script `CustomInputField`. O conteúdo desse campo é então validado ao clique do botão `Validar`.

Figura 16 - Listener do método `OnSelectEntered` do `ARSelectionInteractable`

Fonte: elaborado pelo autor.

Figura 17 - `TextMeshPro` passado como atribulo do *script* `CustomInputField`



Fonte: elaborado pelo autor.

4 RESULTADOS

Este capítulo tem como objetivo demonstrar os testes realizados com a aplicação. Serão apresentadas três seções, nas quais abordam-se os resultados obtidos através dos testes das ferramentas, dos testes funcionais, e dos testes realizados com acadêmicos e profissionais da área de biologia.

4.1 TESTES DE FERRAMENTAS

Por se tratar de tecnologias novas e que estão em constante alterações, alguns testes foram realizados nas ferramentas selecionadas inicialmente para determinar se elas eram capazes de atender os objetivos do trabalho. Com isso, alguns projetos de testes foram criados a fim de se aprofundar nas funcionalidades dessas ferramentas.

A definição inicial do projeto tinha como ferramenta de detecção o *package* OpenCV for Unity. Entretanto, os testes iniciais nessa tecnologia não se mostraram satisfatórios, uma vez que a aplicação apresentava uma média de 15 FPS (*Frames Per Second*) enquanto a detecção ocorria. Após pesquisas para substitutos, se optou pela utilização da tecnologia Barracuda, que em testes atendeu todas as necessidades do projeto com um ótimo desempenho no dispositivo móvel.

Outro ponto alterado durantes os testes das ferramentas foi a forma com que o objeto 3D viria a ser ancorado. A princípio a tecnologia a ser utilizada para realizar este processo também seria o *package* OpenCV for Unity. Porém, após testes de implementação, observou-se que o objeto 3D se apresentava instável na cena, muitas vezes se movendo para os lados ou girando. Desta forma, a rotina de ancoragem do objeto foi alterada para a utilização do sistema de ancoragem do AR Foundation.

4.2 TESTES DE FUNCIONALIDADE

No decorrer do desenvolvimento da aplicação diversos testes foram executados na plataforma Android, visando garantir que as funcionalidades do aplicativo se comportassem conforme o esperado. Os testes foram realizados em um dispositivo móvel Xiaomi Redmi Note 8 Pro que suporta a tecnologia ARCore, um pré-requisito para o dispositivo uma vez que o AR Foundation necessita desta tecnologia para funcionar. Nele foram realizados testes para verificar a rotina de detecção, a rotina de ancoragem do objeto 3D e alguns cenários relacionados a elas aos quais os usuários poderiam vir a enfrentar.

Os testes na aplicação foram realizados constantemente para garantir que cada nova funcionalidade implementada durante o desenvolvimento atendesse o propósito inicial de forma correta e sem impactar no restante da aplicação. Desta maneira, o desenvolvimento se deu de forma iterativa com a realização de frequentes refatorações no código para melhor atender a novas funcionalidades sem romper com o objetivo inicial do aplicativo.

Foram realizados testes a fim de garantir um bom funcionamento da aplicação em diferentes cenários para identificar aspectos que podem vir a variar na utilização do aplicativo pelo usuário. Estes testes foram divididos em duas categorias; os testes ligados a detecção, como qual a melhor posição da folha da árvore para uma detecção correta, a necessidade ou não de um fundo de cor sólida para um melhor destaque da folha, diferentes graus de luminosidade no cenário, entre outros. Bem como os testes relacionados a Realidade Aumentada, como testes movimentando o dispositivo móvel pelo ambiente para verificar se os objetos 3D se mantinham estáveis e testes garantindo a funcionalidade dos componentes tangíveis.

4.3 TESTES COM ACADÊMICOS E ESPECIALISTA DA ÁREA DE BIOLOGIA

Para validar se o projeto cumpriu com o objetivo proposto, a aplicação foi testada in loco na sala S-203 da FURB no dia 02 de junho 2021 pela Professora Roberta Andressa Pereira do curso de Licenciatura em Ciências Biológicas da Universidade Regional de Blumenau e por um bolsista do curso. Fotos dos testes podem ser vistos no APÊNDICE A.

Foram recolhidas algumas folhas de árvores no campus I da FURB e levadas à sala para testes. No local, as pessoas puderam testar o aplicativo, que estava ativo em dois dispositivos, e responder um formulário para recolher alguns resultados desse teste. A primeira etapa do formulário consistia em um levantamento do perfil dos entrevistados.

Tabela 1 - Perfil dos entrevistados

Sexo	Feminino Masculino	50% 50%
Idade	38 26	50% 50%
Grau de escolaridade	Ensino superior completo	100%
Utiliza dispositivos móveis com frequência	Frequentemente	100%
Já utilizou aplicações com Realidade Aumentada	Sim	100%

Fonte: elaborado pelo autor.

Conforme evidenciado na Tabela 1, os entrevistados tinham 38 e 26 anos e ensino superior completo. Todos utilizavam dispositivos móveis com frequência e já haviam usado outras aplicações com Realidade Aumentada.

A segunda etapa do questionário trazia um passo a passo demonstrando como utilizar a aplicação FolhAR. Passando pelas principais funcionalidades, testando a usabilidade da aplicação para um usuário que acabou de iniciar o aplicativo, e verificando se ele é capaz de navegar na aplicação sem problemas. Como resultado desta etapa, todos os usuários conseguiram completar o passo a passo sem dificuldades, necessitando ajuda em menos de três etapas. Para terceira parte do questionário foram formuladas algumas perguntas utilizando a escala Likert, visando-se classificar a usabilidade do aplicativo e verificar se cumpriu com o papel proposto, de ajudar os usuários a aprender sobre folhas de árvores com o auxílio da Realidade Aumentada. Os resultados podem ser observados na Tabela 2.

Tabela 2 - Opinião dos entrevistados sobre o aplicativo

Usabilidade do aplicativo	5	100%
Cumpriu o objetivo de auxiliar no conhecimento de folhas de árvores com auxílio da Realidade Aumentada	5	100%
Recomendaria o aplicativo para quem deseja aprender mais sobre folhas de árvores	5	100%

Fonte: elaborado pelo autor.

Como constatado na Tabela 2, ambos os usuários concordam que o aplicativo cumpre seu objetivo de auxiliar no conhecimento de folhas de árvores com apoio da Realidade Aumentada. Ademais, a Professora Pereira (2021) comenta que o aplicativo é de grande valia para os estudantes do curso ministrado por ela e vê o mesmo com uma ótima forma de passar o conhecimento sobre folhas de árvores de uma maneira mais dinâmica e, por fim, aponta alguns ajustes de nomenclatura e descrição em alguns formatos de folhas.

Em comparação com os trabalhos correlatos, é possível notar que o trabalho apresentado faz a união das principais características dos trabalhos de Bortolon (2014) e de Oliveira e Prado (2018). Trazendo a detecção de formatos folhas, em comparação com a detecção de folhas de Bortolon (2014), e a apresentação de informações sobre esses formatos, em comparação com as informações sobre as plantas exibidas no trabalho de Oliveira e Prado (2018). Em relação ao PlantSnap (2020), é observado que embora o trabalho apresentado não tenha um reconhecimento tão avançado quanto, ele realiza uma maior interação com a Realidade Aumentada, trazendo conteúdos sobre o formato da folha detectado.

5 CONCLUSÕES

Com base nos resultados obtidos, o aplicativo alcançou seu objetivo de auxiliar no conhecimento de folhas de árvores por intermédio da Realidade Aumentada. O objetivo específico de utilizar a folha da árvore como marcador para ancoragem do objeto 3D de Realidade Aumentada foi alcançado, porém necessitando de um fundo controlado para detecção da folha, como uma folha de papel branca, enquanto o objetivo específico de analisar a eficácia do aplicativo com usuários da área de biologia foi alcançado com êxito e sem limitações. Mesmo os testes sendo realizados com um grupo pequeno de usuários, os resultados obtidos foram satisfatórios, uma vez que foram executados por usuários pertencentes ao ramo de estudo da biologia e se mostraram positivos à ideia de o aplicativo contribuir na busca de conhecimentos de folhas de árvores em campo ou na sala de aula.

Após os empecilhos gerados pelas tecnologias inicialmente escolhidas, o projeto teve um desenvolvimento isento de grandes problemas na plataforma Unity com a utilização dos *packages* Barracuda e AR Foundation. O primeiro se mostrou muito eficiente na detecção de objetos de forma rápida e performática em dispositivos móveis, mesmo não possuindo um modelo de aprendizagem de máquina ideal para a identificação das folhas com uma precisão maior. Já o segundo, se apresentou eficiente nas rotinas de Realidade Aumentada, proporcionando fluidez e agilidade no desenvolvimento dessas rotinas.

Embora o trabalho cumpra com todos os objetivos propostos, possíveis melhorias e extensões de pesquisa foram levantadas para continuações neste projeto:

a) melhorias da detecção e classificação das folhas, evitando a necessidade da utilização de um fundo controlado atrás das folhas das árvores;
b) recortar a folha detectada e utilizar a mesma como textura do objeto 3D;
c) realizar não apenas a classificação de formatos das folhas, mas sim da espécie;
d) permitir que a detecção seja feita diretamente nas folhas presas as árvores;
e) salvar os resultados do modo quiz e criar um ranking entre os usuários.

REFERÊNCIAS

ALMEIDA, Marcílio de; ALMEIDA, Cristina Vieira de. **Morfologia da folha de plantas com sementes**. Piracicaba: ESALQ/USP, 2018. (111 p.). (Coleção Botânica, 3). ISBN: 978-85-86481-64-2. Disponível em: https://www.esalq.usp.br/biblioteca/pdf/morfologia_folha.pdf. Acesso em: 31 mai. 2021.

ARAUJO, Joniel Mendes de et al. EDUCAÇÃO AMBIENTAL: importância das aulas de campo em ambientes naturais para a disciplina de biologia no ensino médio da escola Joaquim Parente na cidade de Bom Jesus - PI. **Ensino, Saúde e Ambiente**, Bom Jesus, v. 8, n. 2, p. 25-36, set. 2015.

ASHIKHMIN, Andrey. **TFClassify Unity Barracuda**. [S.l.], [2021]. Disponível em: https://github.com/Syn-McJ/TFClassify-Unity-Barracuda. Acesso em: 17 jun. 2021.

AZUMA, Ronald T. A Survey of Augmented Reality. **Presence**: Teleoperators And Virtual Environments, Malibu, v. 6, n. 4, p. 355-385, set. 1997.

AZUMA, Ronald T.; BAILLOT, Y.; BEHRINGER, R.; FEINER, S.; JULIER, S.; MACINTYRE, B. Recent advances in augmented reality. **Ieee Computer Graphics And Applications**, [S.L.], v. 21, n. 6, p. 34-47, 2001. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/38.963459.

AZUMA, Ronald T. Making Augmented Reality a Reality. **Imaging And Applied Optics 2017 (3D, Aio, Cosi, Is, Math, Pcaop)**, [S.L.], p. 1-3, 2017. OSA. http://dx.doi.org/10.1364/3d.2017.jtu1f.1.

BORTOLON, Matheus. **PLANTARUM**: uma aplicação Android para consultas de plantas. 2014. 83 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau, 2014.

CAMPOS, Carlos Roberto Pires. A saída a campo como estratégia de ensino de ciências. **Revista Eletrônica Sala de Aula em Foco**, v.1, n.2, p. 25-30, 2012.

CHOUHAN, Siddharth Singh et al. **A data repository of leaf images: Practice towards plant conservation with plant pathology**. In: International Conference on Information Systems and Computer Networks, 4., 2019. Madhav Institute of Technology & Science, 2019.

FLECK, Leandro; TAVARES, Maria Hermínia Ferreira; EYNG, Eduardo; HELMANN, Andrieli Cristina; ANDRADE, Minéia Aparecida de Moraes. Redes neurais artificiais: princípios básicos**. Revista Eletônica Ciência Inovação e Tecnologia**, Cascavel, v. 7, n. 15, p. 47-57, jun. 2016. ISSN 2175-1846.

KIRNER, Claudio et al. **Fundamentos e Tecnologia de Realidade Virtual e Aumentada**. Belém: SBC, 2006.

KIRNER, Claudio; SISCOUTTO, Robson. **Realidade Virtual e Aumentada**: Conceitos, Projeto e Aplicações. Petrópolis: SBC, 2007.

KUMAR, Neeraj et al. **Leafsnap**: A computer vision system for automatic plant species identification. In: European conference on computer vision. Springer, Berlin, Heidelberg, 2012. p. 502-516.

MATOS, Wendler Luis Nogueira; LOPES, Carlos Henrique Gurjão; PORTAL, Lucas Manoel Moraes, SILVA, Orlando Fonseca. REDES NEURAIS ARTIFICIAIS: fundamentos e aplicações em Engenharia Elétrica. In: **Encontro Regional dos Grupos do Programa de Educação Tutorial da Região Norte**, 7., 2020. Online. UNIR, 2020.

MICROSOFT. **ONNX models**. [S.l.], [2019]. Disponível em: https://docs.microsoft.com/en-us/windows/ai/windows-ml/get-onnx-model. Acesso em: 29 maio 2021.

MORAIS, M. B.; PAIVA, M. H. **Ciências – ensinar e aprender**. Belo Horizonte: Dimensão, 2009.

OLIVEIRA, Guilherme Mello; PRADO, Ely F. GAIA: desenvolvimento de sistema baseado em realidade aumentada para auxiliar a escolha e cultivo de plantas ornamentais em centros de paisagismo. **Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica**, Franca, v. 9, n. 1, p. 1-27, jan. 2018.

PEREIRA, Roberta Andressa**. Apresentação e testes aplicativo FolhAR**. 2021. Entrevistador: Bruno Geisler Vigentas. Blumenau. 2021. Entrevista feita através de conversação – não publicada.

PLANTSNAP. **Plant Identifier App, #1 Mobile App for Plant Identification**, 2020. Disponível em: https://www.plantsnap.com/. Acesso em: 19 set. 2020.

SCHMALSTIEG, Dieter; HÖLLERER, Tobias. **Augmented Reality**: principles and practice. Boston: Addison-Wesley, 2016. 473 p.

SOUSA, Cristiane Aureliano de; MEDEIROS, Monalisa Cristina Silva; SILVA, José Adailton Lima; CABRAL, Laíse Nascimento. A aula de campo como instrumento facilitador da aprendizagem em Geografia no Ensino Fundamental. **Revista Educação Pública**, Rio de Janeiro, v. 16, n. 22, p. 1-1, 25 out. 2016. Disponível em: https://educacaopublica.cecierj.edu.br/artigos/16/22/a-aula-de-campo-como-instrumento-facilitador-da-aprendizagem-em-geografia-no-ensino-fundamental. Acesso em: 08 out. 2020.

UNITY. **About AR Foundation**. [S.l.], [2021a]. Disponível em: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@4.1/manual/index.html. Acesso em: 29 mai. 2021.

UNITY. **AR Foundation**. [S.l.], [2021b]. Disponível em: https://unity.com/unity/features/arfoundation. Acesso em: 29 mai. 2021.

UNITY. **Barracuda package brings Neural Networks to Unity**. [S.l.], [2020a]. Disponível em: https://forum.unity.com/threads/barracuda-package-brings-neural-networks-to-unity.962922/. Acesso em: 29 mai. 2021.

UNITY. **Burst User Guide**. [S.l.], [2020b]. Disponível em: https://docs.unity3d.com/Packages/com.unity.burst@1.3/manual/. Acesso em: 29 mai. 2021.

UNITY. **Compute shaders**. [S.l.], [2021c]. Disponível em: https://docs.unity3d.com/Manual/class-ComputeShader.html. Acesso em: 29 mai. 2021.

UNITY. **Introduction to Barracuda**. [S.l.], [2021d]. Disponível em: https://docs.unity3d.com/Packages/com.unity.barracuda@2.0/manual/index.html. Acesso em: 29 maio 2021.

UNITY. **Unity Barracuda**. [S.l.], [2019]. Disponível em: https://docs.unity3d.com/Packages/com.unity.barracuda@0.3/manual/index.html. Acesso em: 29 maio 2021.

VIVEIRO, Alessandra Aparecida; DINIZ, Renato Eugênio da Silva. Atividades de campo no ensino das ciências e na educação ambiental: refletindo sobre as potencialidades desta estratégia na prática escolar. **Ciência em Tela**, v. 1, n. 2, p. 1-12, jul. 2009.

WIGGERS, Ivonei; STANGE, Carlos E. B**. Manual de instruções para coleta, identificação e herborização de material botânico**. 2008. Disponível em http://www.diaadiaeducacao.pr.gov.br/portals/pde/arquivos/733-2.pdf. Acesso em: 25 mai. 2021.

WU, Stephen Gang, et al. A leaf recognition algorithm for plant classification using probabilistic neural network. In: IEEE international symposium on signal processing and information technology, 2007. p. 11-16.

ZORZAL, Ezequiel Roberto; KIRNER, Claudio. **Jogos Educacionais em Ambiente de Realidade Aumentada**. In: II WORKSHOP SOBRE REALIDADE AUMENTADA, 2005, Piracicaba, 2005. p. 52-55.

APÊNDICE A – FOTOS DOS TESTES DA APLICAÇÃO

Este apêndice mostra algumas fotografias que foram retiradas enquanto o aplicativo era testado na Universidade Regional de Blumenau no mês de junho de 2021.

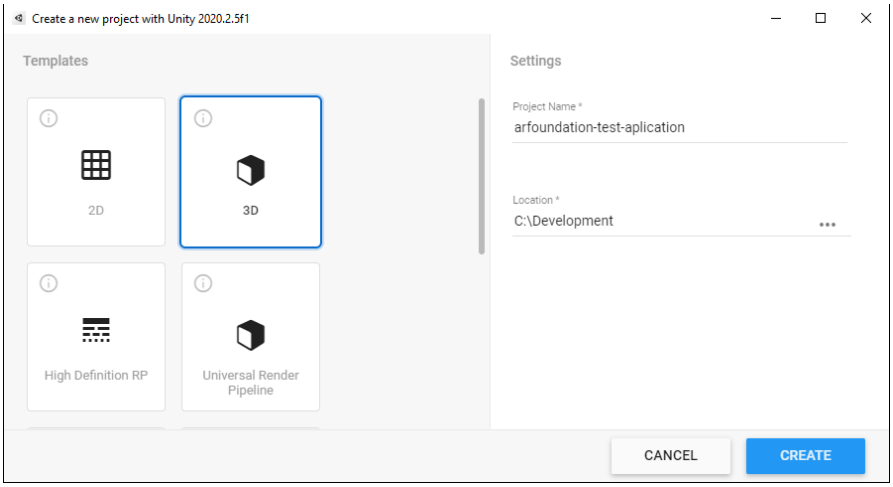
Figura 18 - Teste da aplicação

Fonte: elaborado pelo autor.

APÊNDICE B – APLICAÇÃO SIMPLES COM AR FOUNDATION

Este apêndice tem como objetivo mostrar os passos da criação de um simples aplicativo Android com o AR Foundation, capaz de renderizar objetos 3D em um ambiente de Realidade Aumentada. Mostrando desde a configuração do ambiente até o aplicativo executando.

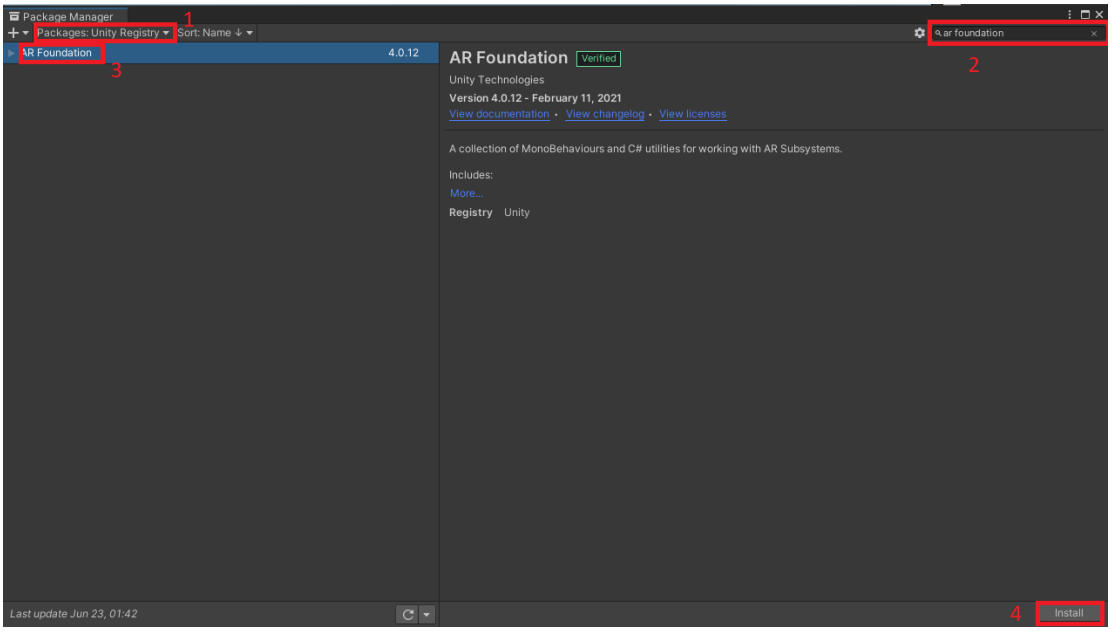
Figura 19 - Criação de um projeto Unity



Fonte: elaborado pelo autor.

O primeiro passo a ser efetuado é a criação um projeto 3D Unity para Android, com nome, um local onde será salvo e certificando-se que a versão do Unity utilizada é igual ou superior a versão 2020.2.5f1. Com o Unity aberto após a criação do projeto, a próxima ação é fazer a adição dos pacotes necessários para o AR Foundation funcionar. Para isso é preciso abrir o `Package Manager` no Unity.

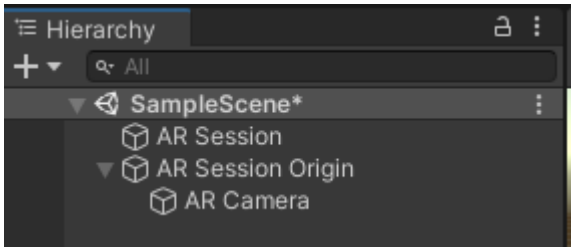
Figura 20 - `Package Manager` no Unity



Fonte: elaborado pelo autor.

No `Package Manager`, demostrado na Figura 20, primeiramente deve-se verificar se o repositório selecionado se trata do `Unity Registry`, como demonstrado em 1. Com ele selecionado, um filtro pelo pacote `AR Foundation` deve ser feito no campo demonstrado em 2. Com o filtro feito, deve-se selecionar o pacote do AR Foundation, demonstrado em 3 e clicar no botão `install`, demonstrado em 4. O mesmo processo deve ser feito para o pacote `ARCore XR Plugin`, uma vez que o AR Foundation depende dele para criação de aplicações Android.

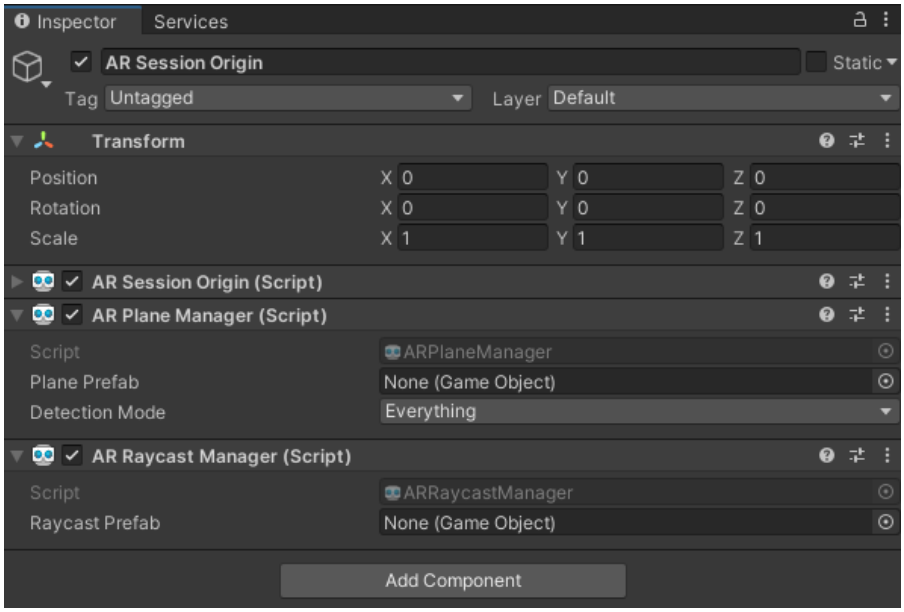
Figura 21 - Elementos essenciais do AR Foundation



Fonte: elaborado pelo autor.

Com os pacotes devidamente instalados, segue-se para a `scene` que é criada automaticamente no projeto Unity. Nela é feita a exclusão de todos os componentes existentes para a adição dos elementos do AR Foundation. Clicando com o botão direito nos elementos da `scene`, deve-se navegar para `XR > AR Session`, e `XR > AR Session Origin` para adicionar os dois componentes fundamentais do AR Foundation. Ao final desse passo, a `scene` deve ficar como demonstrada na Figura 21.

Figura 22 - Componentes adicionados ao `AR Session Origin`



Fonte: elaborado pelo autor.

Conforme visível na Figura 22, a adição de dois componentes deve ser feita no `AR Session Origin` para o gerenciamento algumas funcionalidades. O Primeiro sendo o `AR Plane Manager`, responsável por achar os planos no ambiente, e o segundo o `AR Raycast Manager`, para realização de testes de raycast na aplicação.

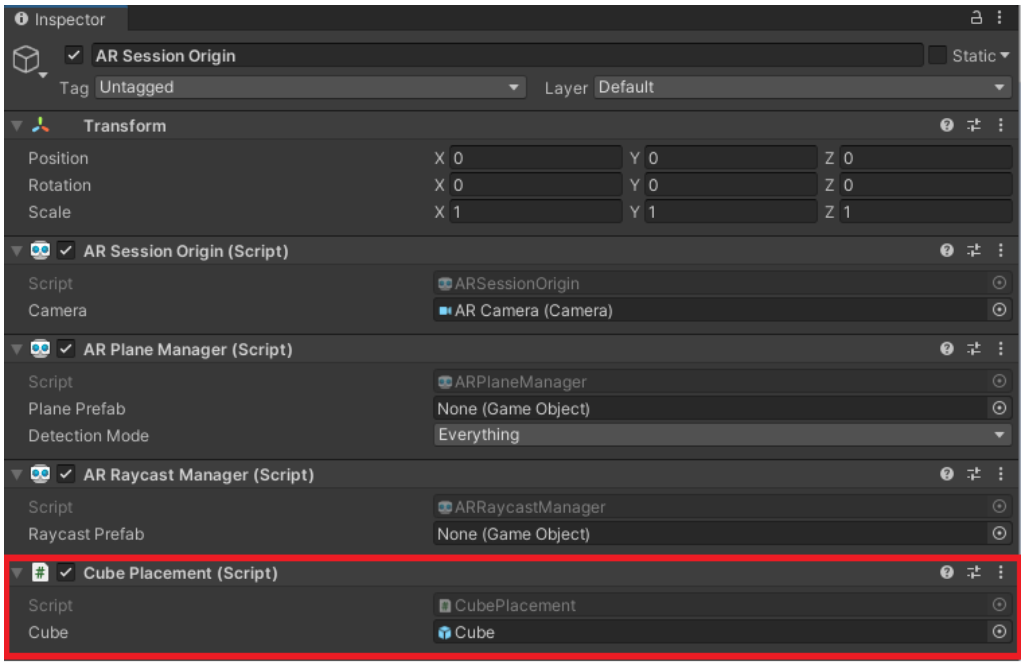
Quadro 11 - Código responsável por fazer a criação do objeto no ambiente de Realidade Aumentada

```
public class CubePlacement : MonoBehaviour
{
    private static List<ARRaycastHit> raycastHits = new List<ARRaycastHit>();

    ARRaycastManager raycastManager;

    [SerializeField]
    GameObject cube;

    Unity Message | 0 references
    void Awake()
    {
        raycastManager = GetComponent<ARRaycastManager>();
    }

    Unity Message | 0 references
    void Update()
    {
        if (Input.touchCount > 0)
        {
            var touchPosition = Input.GetTouch(0).position;

            if (raycastManager.Raycast(touchPosition, raycastHits, TrackableType.PlaneWithinPolygon))
            {
                var hit = raycastHits[0].pose;

                Instantiate(cube, hit.position, hit.rotation);
            }
        }
    }
}
```

Fonte: elaborado pelo autor.

É então realizada a criação de um script C# (Quadro 11) que tem como objetivo localizar os planos no ambiente, fazer a detecção dos cliques na tela, e adicionar um objeto 3D de um cubo no local do clique caso esse seja em cima de um plano.

Figura 23 - Script adicionado ao `AR Session Origin`



Fonte: elaborado pelo autor.

Como apresentado na Figura 23, o script é adicionado ao `AR Session Origin`, e nele é adicionado um `Prefab` de um cubo 3D. Enfim, as seguintes configurações devem ser feitas para realização do build do aplicativo:

a) em `Project Settings > XR Plug-in Management > Android`, garantir que a opção `Initialize XR On Startup` está marcada;
b) em `Project Settings > XR Plug-In Management > Android`, garantir que o `AR Core` está marcado como `Plug-in Providers`;
c) em `Project Settings > Player > Android > Other Settings`, garantir que a opção `Auto Graphics API` está marcada;
d) em `Project Settings > Player > Android > Other Settings`, garantir que a configuração `Minimum API Level` está com `Android 7.0 'Nougat' (API level 24)`, ou superior.

Com as configurações feitas e a build gerada, o aplicativo executando é demostrado na Figura 24.

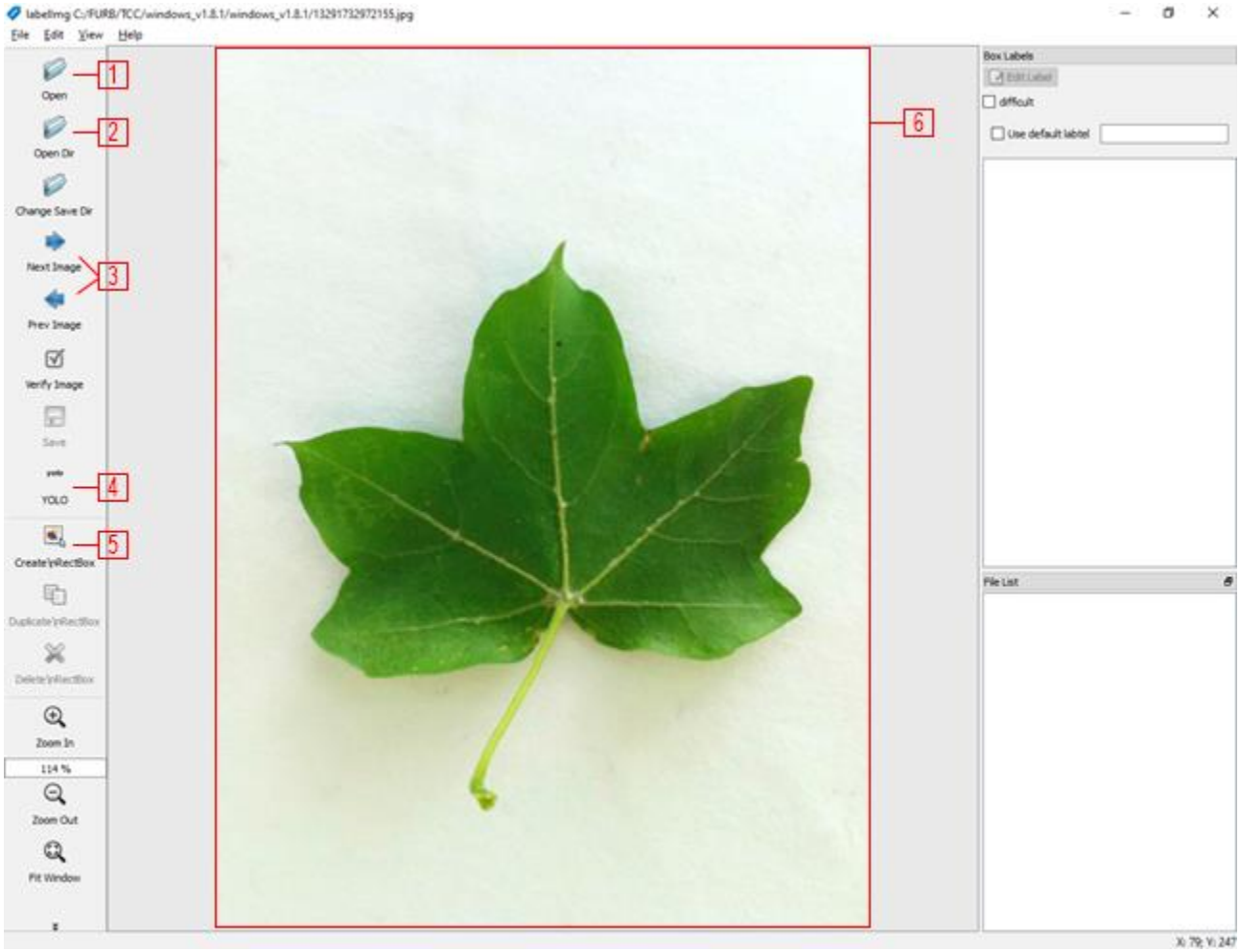
Figura 24 - Aplicativo funcionando

Fonte: elaborado pelo autor.

APÊNDICE C – UTILIZAÇÃO DA FERRAMENTA LABELIMG

Este apêndice demostra a utilização da ferramenta LabelImg para fazer o processo *de image labeling* do dataset de formatos de folhas árvores.

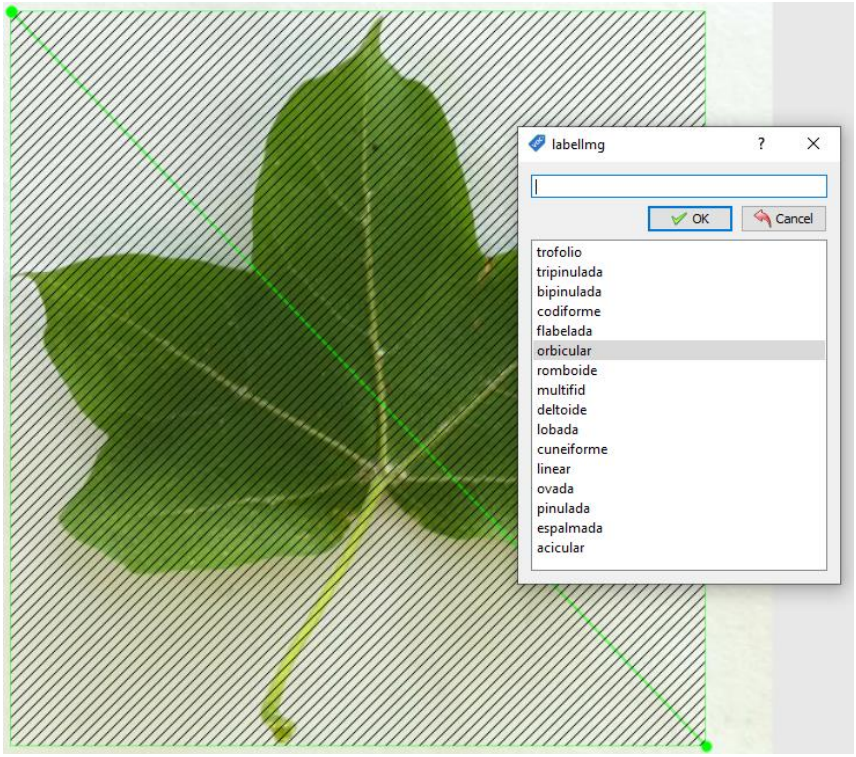
Figura 25 - Ferramenta LabelImg



Fonte: elaborado pelo autor.

A Figura 25 apresenta a ferramenta e seus principais componentes. Em 1 é mostrado o botão para abrir uma imagem para realização da etiquetação dela, em 2 é mostrado o botão para abrir um diretório com várias imagens para facilitar o fluxo, em 3 é apresentado os botões para navegação nas imagens do diretório, em 4 é demonstrado o botão para alterar o formato da anotação, em 5 é apresentado o botão para criar a seleção do objeto e em 6 é apresentada a imagem que foi aberta no programa.

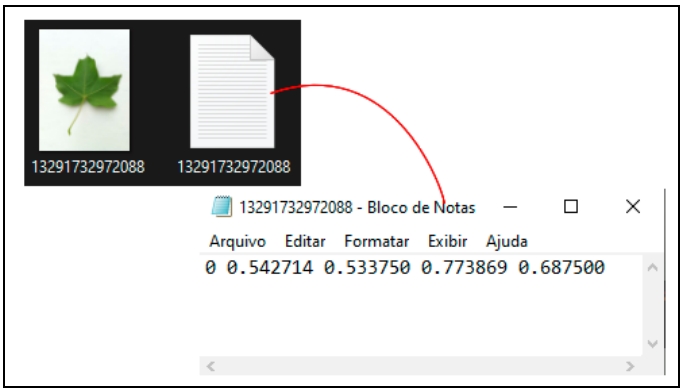
Figura 26 - Seleção do objeto na imagem



Fonte: elaborado pelo autor.

Na Figura 26, é apresentado a seleção do objeto na imagem. Feita a seleção, uma tela pop up é aberta para escolher a etiqueta do objeto selecionado. Nela o usuário pode escolher uma existente ou criar uma nova. Ao final, o salvamento do arquivo faz a geração de um arquivo de texto com o mesmo nome da imagem com as informações de posicionamento da seleção. Um exemplo do arquivo é demonstrado na Figura 27.

Figura 27 - Exemplo do arquivo gerado na ferramenta



Fonte: elaborado pelo autor.