UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UM FRAMEWORK PARA APOIO AO DESENVOLVIMENTO DE APLICAÇÕES EDUCACIONAIS DE REALIDADE AUMENTADA MÓVEL BASEADA EM MARCADORES

MARCOS TULIO SOTO DE LA VEGA

ORIENTADOR: PROF. DR. DELANO MEDEIROS BEDER

São Carlos - SP
Novembro/2018

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UM FRAMEWORK PARA APOIO AO DESENVOLVIMENTO DE APLICAÇÕES EDUCACIONAIS DE REALIDADE AUMENTADA MÓVEL BASEADA EM MARCADORES

MARCOS TULIO SOTO DE LA VEGA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Engenharia de Software
Orientador: Dr. Delano Medeiros Beder.

São Carlos - SP
Novembro/2018

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Marcos Tulio Soto de La Vega, realizada em 27/11/2018:

Prof. Dr. Delano Medeiros Beder
UFSCar

Profa. Dra. Marilde Terezinha Prado Santos
UFSCar

Profa. Dra. Fátima de Lourdes dos Santos Nunes Marques
USP

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Fátima de Lourdes dos Santos Nunes Marques e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

Prof. Dr. Delano Medeiros Beder

Dedico este trabalho à minha família, amigos, companheiros e a todas as pessoas que ao longo da minha vida tem me encorajado a perseguir os meus sonhos, e a ser uma melhor pessoa.

AGRADECIMENTOS

Agradeço primeiramente a Deus por todas as suas bênçãos na minha vida, por ter colocado diante de mim sempre pessoas maravilhosas, e por ter me dado força e sabedoria para levar a cabo com sucesso tanto este trabalho, quanto todos os objetivos propostos na minha vida.

Agradeço imensamente aos meus pais: Levys De La Vega e Manuel Soto, pois devo a eles o que sou hoje, por seu apoio incondicional e por ter me ensinado os princípios e valores que agora possuo, e que considero fundamentais para ter sucesso na vida, assim como para dar nossa contribuição em fazer deste mundo um lugar melhor. Agradeço também aos meus irmãos: Manuel, Diego, Juan de Dios e Juan Diego, pelo seu apoio, e por terem sido para mim fonte de motivação tanto para finalizar este trabalho quanto na minha vida.

Meus sinceros agradecimentos ao meu orientador, Delano, pelo acompanhamento contínuo neste trabalho, obrigado pela sua disposição de ajudar sempre no que for possível, por ter compreendido as minhas dificuldades (além de conhecimento, por causa de enfrentar o desafio de uma nova cultura, um novo idioma etc.), e finalmente por todas as suas acertadas orientações. Sem seu apoio, seria impossível ter concluído este trabalho.

Agradeço aos meus companheiros do LOA, pela cooperação neste trabalho, assim como pela companhia em incontáveis tardes no LOA, sempre estando dispostos a me ajudar quando precisei. Obrigado a: Douglas, Frederico, Gabriel, Julia, Layon, Lucas, Mariana, Miguel, Paulito, Thiago e Valério.

Agradeço por fim meus companheiros e amigos de mestrado pela receptividade e apoio, especialmente a: Daniel, Dennys, Guisella, Steve e Vivi, que sempre me encorajaram em momentos de dificuldade durante esta etapa da minha vida.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

A disciplina, cedo ou tarde, vencerá a inteligência.

Provérbio Japonês

RESUMO

Nos últimos anos, diferentes tecnologias têm começado a ser utilizadas em cenários educativos, fornecendo ganhos nos processos de ensino-aprendizagem. A Realidade Aumentada (RA), e mais especificamente a Realidade Aumentada Móvel (RAM), é uma tecnologia que fornece grandes oportunidades na educação, permitindo misturar elementos do mundo real com informação virtual, fornecendo ao aprendiz um ambiente mais rico e com maiores possibilidades de interação do que seu ambiente natural, e ainda incentivando a mobilidade. Visando fomentar o desenvolvimento de aplicativos educacionais com essa tecnologia, fazem-se necessários *frameworks* e metodologias que facilitem o desenvolvimento e guiem o processo de criação tentando minimizar a complexidade. No entanto, é necessário também que *frameworks* criados sejam voltados especificamente para a educação, e, portanto, que forneçam funcionalidades que permitam apoiar necessidades educativas (p.ex., apoiar a utilização de questionários com perguntas/respostas). Atualmente existem muitos *frameworks* para criar RA, no entanto eles estão voltados para criar aplicativos de RA genéricos, e não especificamente educativos. No presente projeto, é proposto um *framework* de apoio a uma metodologia para o desenvolvimento de aplicações educativas com RAM baseada em marcadores. O *framework* permite a desenvolvedores implementar um conjunto de funcionalidades orientadas na educação sem necessidade de programação, e, no lugar disso, arrastar e soltar classes, montando e customizando cenários com os recursos necessários (modelos 3D, imagens, etc.). A metodologia está composta de uma série de orientações teóricas (*guidelines*) a serem seguidas no desenvolvimento do aplicativo. Foi realizado um estudo de caso para avaliar o *framework* proposto, principalmente quanto a usabilidade, e, adicionalmente, o *framework* também foi avaliado por meio de experimentos com desenvolvedores, além de questionários para coleta de dados. Finalmente, através dos experimentos realizados, conclui-se que o *framework* proposto pode permitir o desenvolvimento de aplicações educativas de RA com facilidade, fornecendo funcionalidades customizáveis de acordo com as necessidades do usuário, e de cunho geral na educação (úteis em qualquer área da educação).

Palavras-chave: Realidade Aumentada, Dispositivos Móveis, Frameworks, Ambientes de Aprendizagem.

ABSTRACT

In recent years, different technologies have begun to be used in educational scenarios, providing gains in the teaching-learning processes. Augmented Reality (AR), and more specifically Mobile Augmented Reality (MAR), is a technology that provides great opportunities in education, allowing to blend real-world elements with virtual information, giving the learner a richer environment and greater possibilities of interaction than their natural environment, still encouraging mobility. Aiming to foster the development of educational applications with this technology, frameworks and methodologies are necessary in such way that facilitate the development and guide the creation process trying to minimize the complexity. However, it is also required that created frameworks are specifically oriented towards education, and therefore provide functionalities that support educational needs (e.g. to support the use of questionnaires with questions/answers). Currently there are many frameworks for creating AR, however they are oriented towards the creation of generic, rather than specifically educational, AR applications. In this project, a framework is proposed to support a methodology for the development of educational applications with MAR based on markers. The framework allows developers to implement a set of education-oriented functionalities without programming, and instead drag and drop classes, creating and customizing scenarios with the required resources (3D models, images, etc.). The methodology is composed of guidelines to be followed in the development of the application. A case study was carried out to evaluate the framework, mainly regarding usability, additionally the framework was also evaluated through experiments with developers, and using questionnaires for data collection. Finally, through the experiments carried out, it can be concluded that the proposed framework can allow the development of educational applications of AR with ease, providing functionalities that are customizable according to the needs of the user, and of a general nature in education (useful in any area of education).

Keywords: Augmented Reality, Mobile Devices, Frameworks, Learning Environment.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AODDEI - *Analise, Obtenção, Desenho, Desenvolvimento, Avaliação, Implementação*

API - *Application Programming Interface*

AR - *Augmented Reality*

ARLE - *Augmented Reality Learning Environment*

ESBC - *Engenharia de Software Baseada em Componentes*

GPS - *Global Positioning System*

GPU - *Unidade de Processamento Gráfico*

HMD - *Head-Mounted Display*

HUD - *Head-Up Display*

J2ME - *Java 2 Micro Edition*

MCQ - *Multiple Choice Questions*

OHMD - *Optical Head-Mounted Display*

OST - *Optical See-Through*

OVA - *Objeto Virtual de Aprendizagem*

PDA - *Personal Digital Assistants*

PIP - *Personal Interaction Panel*

PR - *Public Relations*

RA - *Realidade Aumentada*

RAM - *Realidade Aumentada Móvel*

RV - *Realidade Virtual*

RM - *Realidade Mista*

SDK - *Software Development Kit*

SEPA-M - *The Science Expedition: The Pop-Up Adventure multimedia*

SEPA-AR - *The Science Expedition: The Pop-Up Adventure augmented reality*

TC - *Tomografia Computadorizada*

UI - *User Interface*

UWP - *Universal Windows Platform*

VA - *Virtualidade Aumentada*

VC - *Virtuality Continuum*

VST - *Video See-Through*

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Contexto

Nos últimos anos, os avanços da tecnologia têm mudado grande parte da vida das pessoas em diferentes áreas, fornecendo grandes vantagens e permitindo fazer coisas que pareciam impossíveis nas épocas antigas. Nesse contexto, pesquisadores têm começado a notar inúmeras possibilidades e ganhos que podem ser atingidos quando se faz uso dessas tecnologias para o ensino-aprendizagem. Uma das tecnologias que pode ser útil para apoiar os processos educativos é a Realidade Aumentada (RA), que permite misturar em telas de diversos dispositivos a visão do mundo real com elementos virtuais, também chamados elementos aumentados. A RA, não sendo uma tecnologia tão nova quanto parece, tem começado a ser utilizada em contextos educacionais graças ao uso massivo de dispositivos móveis tais como *smartphones*.

O atual projeto está inserido na área da Realidade Aumentada Móvel (RAM), apresentando um *framework*, voltado para a criação de aplicativos educativos de cunho geral e de apoio a uma metodologia proposta para o desenvolvimento de aplicativos educativos com RAM baseada em marcadores. Neste sentido, uma vez que o *framework* está voltado para a criação de aplicativos do tipo educativo, ele fornece funcionalidades para apoiar processos (ou atender necessidades) especificamente para o contexto educativo. Isto é diferente das ferramentas de RA de propósito geral, as quais fornecem funcionalidades para apoiar necessidades

genéricas no desenvolvimento de sistemas com RA (e não especificamente na educação), e, portanto, não levam em consideração necessidades que surgem quando se tenta aplicar a RA em cenários educativos. Por outro lado, desde que o *framework* proposto também é de cunho geral na educação, significa que suas funcionalidades são igualmente úteis para as diferentes áreas da educação, sem estar limitado a áreas específicas (p.ex., fornecendo um suporte para definição de questionários com perguntas e respostas, o qual seria útil em aplicativos educativos de qualquer área do conhecimento).

O presente trabalho visa servir de base para que desenvolvedores com conhecimento de programação possam criar, de maneira simples, aplicativos educativos com RAM baseada em marcadores e partir de um conjunto de funcionalidades prontas que podem ser customizadas e estendidas de acordo com as suas necessidades. Para atingir esse objetivo, é proposto um *framework* de RAM baseada em marcadores, voltado para o desenvolvimento de aplicativos educativos de cunho geral, e que é usado em uma metodologia que fornece um roteiro claro através de orientações teóricas *(guidelines)* a serem seguidas durante o desenvolvimento do aplicativo.

1.2 Justificativas

O desenvolvimento de sistemas educativos com RA pode fornecer grandes vantagens na educação, no entanto também supõe grandes desafios para os criadores desse tipo de software, pois além da complexidade de todo projeto de software, a RA acrescenta necessidades adicionais complexas (p.ex., criar modelos 3D, comumente necessários em aplicativos de RA e para o qual normalmente se precisa de habilidades de modelagem tridimensional), portanto são necessárias ferramentas e metodologias que permitam apoiar e guiar o máximo possível os desenvolvedores na criação deste tipo de software, visando fazer que em geral os projetos sejam desenvolvidos do jeito mais simples possível.

O presente trabalho tem sua principal justificativa devido às poucas ferramentas e *frameworks* existentes voltados para o desenvolvimento de aplicativos educativos com RAM, além disso o pouco número de metodologias de RAM para cenários educativos foi outro fator que motivou o presente trabalho. Deve-se notar que RA, e especificamente a RAM, são tecnologias que só recentemente começaram a ser utilizadas no contexto educativo, e ainda fazem-se necessários esforços de pesquisa nessas áreas, como afirma Fabregat Gesa (2012): “[...] esta combinação [RA com tecnologias móveis criando a RAM] e sua aplicação em cenários educativos continua sendo uma área aberta para a pesquisa”.

Inicialmente, por meio da pesquisa bibliográfica realizada, pode-se notar que a maioria de ferramentas e *frameworks* existentes para a criação de aplicativos com RAM, e ainda com RA, não tinha como foco a educação, ou seja, não estava voltada para a criação de aplicativos especificamente do tipo educativo, pois eram ferramentas de RA de propósito geral e não consideravam aspectos ou necessidades educacionais. Esta situação também tem sido notada em outras pesquisas, como é apresentado a seguir: “[...] análise do estado da arte de *frameworks* para criação de experiências educacionais de RA mostra uma falta de ferramentas especificamente desenhadas para a entrega de conteúdo educativo e avaliativo de RA” (CUBILLO et al., 2014). Além disso, as poucas ferramentas voltadas para a educação que foram encontradas têm como público alvo professores, e, portanto, consistem em abordagens focadas na criação de aplicações sem nenhum tipo de codificação, e com grandes limitações quanto aos tipos de aplicativos que podem ser criados. Considera-se muito importante também oferecer ferramentas que facilitem o desenvolvimento destes aplicativos usando programação (que tenham como público alvo programadores), o que permite conservar total liberdade, controle e flexibilidade na criação do software.

Cabe observar ainda que das metodologias encontradas para o desenvolvimento de aplicativos educativos com RAM, grande parte delas não possui *frameworks* nem ferramentas para apoiar a criação do aplicativo, além de fornecer uma série de passos ou orientações claramente definidas, assim este tipo de metodologias consiste basicamente em orientações teóricas que servem de guia na criação do software.

Portanto, por meio da pesquisa bibliográfica realizada, parece-nos que há muito poucas ferramentas e *frameworks* de criação de software de RAM voltados especificamente para a educação, assim como muito poucas metodologias de RAM para cenários educativos que forneçam ferramentas de apoio no desenvolvimento dos aplicativos, além de fornecer orientações teóricas. Dessa forma, uma vez que não foi encontrado um *framework* de RAM voltado para a criação de aplicativos especificamente do tipo educativo de cunho geral, tendo como público alvo desenvolvedores, e que seja usado de apoio a uma metodologia para o desenvolvimento de aplicativos educativos com RAM, surge a necessidade da presente pesquisa.

Procurando dar solução ao problema encontrado, neste trabalho foi proposto o desenvolvimento de um *framework*, voltado para a educação e de cunho geral, tendo como público alvo desenvolvedores com conhecimento de programação, e utilizado em uma metodologia para o desenvolvimento de aplicações educativas com RAM baseada em marcadores. Em geral, com a presente proposta, visou-se oferecer um suporte para que desenvolvedores possam criar, de maneira simples, experiências educativas de RAM baseadas em marcadores, em que o *framework* proposto fornece funcionalidades prontas voltadas na educação, servindo de base para o processo de codificação, e a metodologia proposta fornece orientações teóricas (*guidelines)*, que permitem guiar o desenvolvedor em todo processo de criação do aplicativo, considerando o uso do *framework*.

1.3 Objetivos

1.3.1 Objetivo Geral

- Criar um *framework,* tendo como público alvo desenvolvedores, de apoio a uma metodologia para o desenvolvimento de aplicativos educativos (de cunho geral) com RAM baseada em marcadores.

1.3.2 Objetivos Específicos

- Criar um *framework* visando apoiar a desenvolvedores na criação de aplicativos educativos com RAM baseada em marcadores, fornecendo funcionalidades prontas, altamente customizáveis, e orientadas a cobrir processos ou necessidades específicas para o contexto educativo e de cunho geral na educação.
- Criar uma metodologia se apoiando no *framework* proposto para o desenvolvimento de aplicativos educativos com RAM baseada em marcadores.

Com o objetivo de validar o *framework*, foram realizadas as seguintes atividades:

- Desenvolver um estudo de caso com o *framework* e metodologia propostos, visando, através da experiência do próprio autor, medir usabilidade do *framework* e determinar se as funcionalidades fornecidas são altamente customizáveis e de cunho geral na educação (podem ser usadas para qualquer área da educação).
- Realizar experimentos usando o *framework* proposto, com um grupo de estudantes de cursos de computação, que permitam mais uma forma de validação do *framework*, para medir usabilidade do *framework*, e determinar se as funcionalidades fornecidas são altamente customizáveis.

1.4 Organização do Trabalho

A dissertação está organizada em 7 capítulos, descritos a seguir:

- Capítulo 1: Apresentou a introdução, contexto, justificativas, e os objetivos da presente pesquisa.
- Capítulo 2: Apresenta conceitos referentes a RA, RAM, definições, características, assim como ferramentas e tecnologias hardware

necessárias para a criação de aplicações com RA. Este capítulo também apresenta informação referente à RA na educação, visando esclarecer a importância desta tecnologia em contextos educativos, assim como definir quais são a suas vantagens e ganhos principais em relação a outras tecnologias similares ou ferramentas usadas nos processos de ensino-aprendizagem, notando que estas questões são abordadas tanto com a tecnologia de RA em geral, quanto com a RAM, que é a área específica onde está inserido o presente trabalho.

- Capítulo 3: Apresenta informação referente a projetos existentes relacionados com o presente trabalho, assim, consiste em uma lista de projetos com metodologias, *frameworks* e ferramentas propostas, voltados ao desenvolvimento de aplicativos com RA (tanto no contexto educativo quanto fora dele) e que tem algum tipo de relação com nosso *framework* e metodologia propostos. Para cada um dos projetos listados é feita tanto uma análise das suas características, quanto uma comparação em relação ao nosso projeto, indicando similaridades e diferenças. Após apresentar os projetos, no final do capítulo apresenta-se uma discussão geral sobre eles e o nosso projeto, visando dar uma visão mais geral das características dos projetos existentes, e aspectos que motivaram a realização do presente trabalho.
- Capítulo 4: Apresenta o *framework* e a metodologia que são propostos no presente trabalho visando solucionar o problema estabelecido inicialmente, fornecendo informação detalhada deles e das suas características particulares.
- Capítulo 5: Apresenta em detalhe o estudo de caso realizado (como uma primeira abordagem de validação do *framework* pelo próprio autor), que consistiu na criação de um aplicativo educacional de RAM para telefones Android com dois cenários disponíveis, fazendo uso do *framework* e metodologia propostos.
- Capítulo 6: Apresenta experimentos realizados usando o *framework* proposto, com um grupo de estudantes de cursos de computação, visando validá-lo com um público externo. Estes experimentos consistiram em treinamentos iniciais, realização de atividades propostas, e, finalmente, coleta de dados por meio de questionários

que permitiram conhecer a percepção dos participantes do *framework* proposto.

- Capítulo 7: Foram apresentadas as conclusões desta dissertação, incluindo contribuições, limitações identificadas e sugestões de trabalhos futuros para continuidade desta dissertação.

Capítulo 2

REALIDADE AUMENTADA (RA)

Uma das definições de RA é apresentada a seguir: “Como uma definição operacional da RA, tomamos o termo para se referir a qualquer caso em que um ambiente real é aumentado por meio de objetos virtuais (Computação Gráfica)” (MILGRAM & KISHINO, 1994).

Por outro lado, visando que a definição de RA fosse independentemente de tecnologias específicas (p.ex., que a RA seja independente de dispositivos como *Head-Mounted Display*[1] (HMD)), Azuma (1997) define a RA como sistemas que têm as seguintes características:

- Combinar o mundo real com o virtual;
- Interativo em tempo real;
- Registrado em 3D.

Algumas pesquisas mais recentes relacionadas com a RA têm proposto outras definições apresentadas a seguir: “A RA não é nada além do que informações digitais situadas no topo de objetos do mundo real, onde o usuários é capaz de interagir com conteúdo virtual no mundo real podendo diferenciar o mundo real e virtual” (SHEIKH & SAWANT, 2016). Uma definição mais simples sugere que: “A RA

[1] HMD, é um dispositivo usado na cabeça que tem um pequeno *Display* óptico em frente de um ou de cada olho. Para mais informações ver em: https://en.wikipedia.org/wiki/Head-mounted_display

refere-se a tecnologias que projetam materiais digitais sobre objetos do mundo real" (CUENDET et al., 2013).

Notam-se que os conteúdos que podem ser aumentados não se limitam a ser só do tipo visual, embora ele seja o tipo de conteúdo geralmente mais usado nos sistemas de RA. Em relação a isso, tem-se que: "A RA não se restringe apenas ao sentido visual, ela pode ser potencialmente aplicada a todos os sentidos, incluindo o sentido do tato, audição, etc." (AZUMA et al., 2001).

A RA é uma tecnologia muito similar à Realidade Virtual (RV), mesmo porque é derivada dela, no entanto ambas as tecnologias com nomes e natureza similares não devem ser confundidas pois têm diferenças muito importantes, sendo a principal que na RA o usuário continua visualizando o mundo real só que aumentado com informação virtual, no entanto na RV o mundo real é substituído completamente pelo virtual e o usuário fica em todo momento visualizando só o novo ambiente virtual. Em relação a isso, tem-se o seguinte:

> "A RA é uma variante dos Ambientes Virtuais ou RV como é mais conhecida. As tecnologias de RV submergem completamente ao usuário dentro de um ambiente sintético, enquanto o usuário está nesse ambiente ele não pode ver o mundo real ao seu redor. Pelo contrário, a RA permite ao usuário ver o mundo real com objetos virtuais sobrepostos ou misturados nele. Portanto, a RA complementa a realidade ao invés de substituí-la completamente". (AZUMA, 1997).

Uma questão importante para entender a RA, e como ela está relacionada com tecnologias similares, é o termo *Virtuality Continuum* (VC), do qual faz parte a RA e outros tipos de tecnologias que misturam a realidade com a virtualidade de algum jeito. No artigo de Milgram & Kishino (1994) foi apresentado o conceito do VC, o qual define em um extremo (esquerda) os ambientes reais (ambientes onde todos os objetos fazem parte da nossa realidade), e no extremo oposto (direita), Ambientes Virtuais ou RV (onde todos os objetos são gerados com computação), assim aquilo que fica no meio dos dois extremos é chamado Realidade Mista (RM), sendo eles

ambientes onde se misturam e são apresentados juntos objetos do mundo real e do mundo virtual.

A Figura 2.1 mostra o VC, verificando que a RA faz parte da RM e está mais perto do ambiente real do que o ambiente virtual, e nota-se que a Virtualidade Aumentada[2] (VA) é outro tipo de tecnologia que também faz parte da RM, porém ela está mais perto do mundo virtual do que do mundo real.

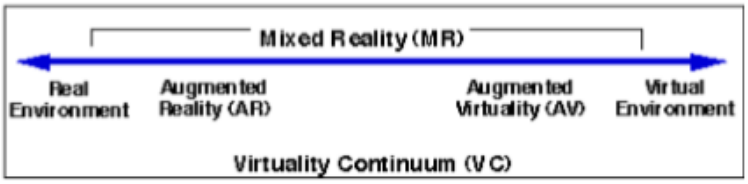

Figura 2.1: Representação simplificada do VC. (MILGRAM & KISHINO, 1994)

Assim, levando em consideração as tecnologias mencionadas e através do VC, é possível notar relações como a seguinte: “Tanto em VA quanto em Ambientes Virtuais (ou RV), o ambiente ao redor é virtual, enquanto que na RA o ambiente ao redor é real” (AZUMA et al., 2001).

2.1 Realidade Aumentada Móvel (RAM)

Segundo Grubert & Grasset (2013), conforme citado por Araújo et al. (2015), “a RAM é a aplicação do conceito de RA em plataformas móveis, que significa misturar conteúdos virtuais e cenas do ambiente real na tela do dispositivo móvel”. Assim, a RAM consiste na mistura de duas tecnologias, que são: A RA e a computação móvel. “Esta sinergia [a sinergia entre RA e computação móvel] permite ter acesso em tempo real a dados e informações virtuais que são inseridas sobre o que nós estamos vendo, enquanto nos movimentamos por ambientes naturais ou urbanos” (ZUÑIGA TORRES, 2008).

[2] VA é uma tecnologia que permite misturar realidade e virtualidade, onde são inseridos elementos reais que complementam um mundo virtual. Para mais informações ver em: https://en.wikipedia.org/wiki/Mixed_reality#cite_note-15

Deve-se notar que o conceito de RAM algumas vezes pode se referir a qualquer tipo de sistema com RA que permita sua movimentação, incluindo ainda sistemas com dispositivos HMD e não especificamente com dispositivos móveis de mão como *smartphones*. “RAM às vezes se refere a qualquer sistema de RA transportável que possa ser usado *Indoors* e *Outdoors*. [...], nós veremos a RAM com a conotação mais popular usada hoje — usando dispositivos móveis de mão, tais como *smartphones* ou *tablets*” (GRUBERT & GRASSET, 2013). No presente trabalho, também será adotada essa conotação para se referir à tecnologia de RAM, podendo também ser encontrada na literatura como o termo *Hand-held AR.*

A RAM surgiu aproveitando os avanços da tecnologia moderna nos dispositivos móveis tais como celulares, *smartphones* e *tablets*, e, quanto ao hardware, potência e a incorporação de novas ferramentas integradas como sensores GPS (*Global Positioning System),* acelerômetro etc., que fornecem uma grande faixa de características adicionais às que se podem obter quando se usa a RA em dispositivos fixos tais como *Desktop*. Além dos fatos anteriores, a computação móvel também tem sido importante para pesquisadores e desenvolvedores de RA devido a vários aspectos como o uso massivo de dispositivos móveis na atualidade (quase todas as pessoas têm acesso a eles), seu relativo baixo custo, etc. O anterior também é apoiado pela afirmação apresentada a seguir:

> “Nos últimos anos, demonstrações e aplicações de *Hand-held AR* tornaram-se comuns à medida que dispositivos inteligentes têm permeado o mercado. *Smartphones* e pequenos *tablets* agora vêm equipados com sensores de rastreamento, câmeras, poderosa GPU (Unidade de Processamento Gráfico) e *Displays* de alta resolução. A integração de todas estas tecnologias tem permitido a desenvolvedores e pesquisadores criar aplicações reais de RA em dispositivos *tablets* e telefones de tamanho pequeno para as massas”. (BARICEVIC et al., 2012).

Uma das características mais importantes (tal vez a mais importante) de utilizar a RA com dispositivos móveis é a possibilidade que esses dispositivos

fornecem de ser usados em qualquer lugar. Papagiannakis et al. (2010), conforme citado por Fabregat Gesa (2012), afirmam o seguinte:

> "Nos últimos anos tem existido uma tendência em misturar as tecnologias móveis com RA para permitir a criação de aplicações de RA que se beneficiam das características de portabilidade e acesso imediato à informação que são atingidas com dispositivos móveis".

No artigo de Araújo et al. (2015) são definidas duas abordagens normalmente consideradas na RAM quando se mistura conteúdo virtual em cenários reais, elas são: Marcadores e localização. Segundo o artigo, a abordagem de marcadores se caracteriza por usar software de reconhecimento de um padrão para seleção do conteúdo que vai ser apresentado ao usuário, por outro lado a abordagem de localização permite definir esse conteúdo a partir do uso do GPS do dispositivo móvel e da orientação da câmera.

Outro tipo de abordagem que pode ser considerado para diferenciar a natureza dos sistemas de RAM da utilizada na RA tradicional para dispositivos *Desktop* com Webcam (*Desktop* AR) é através da analogia de *Magic Lenses* (normalmente referida como *Magic Lenses Metaphor*) para sistemas do tipo de RAM, e, de forma similar, a analogia de *Augmented Mirror* para sistemas de *Desktop* AR. Em relação a essas analogias, quando se tem *Desktop* AR, acontece o seguinte: "A câmera captura a visão do mundo real, e software especializado gera o conteúdo aumentado, o qual é posicionado, orientado e exibido na tela do computador como um espelho aumentado" (CAMBA & CONTERO, 2015). Por outra parte, em relação a RAM tem-se que: "Os telefones móveis com câmeras embutidas tornam possível usar um estilo de lente mágica de RA, usando a imagem da câmera ao vivo, tanto para o monitoramento de visão por computador quanto para a exibição de imagens 3D aumentadas" (SCHMALSTIEG & WAGNER, 2009).

Figura 2.2: Augmented Mirror com Desktop AR (acima) e Magic Lenses com RAM (abaixo). (CAMBA & CONTERO, 2015)

Finalmente, tem que se notar que, assim como aconteceu com o surgimento da RAM, a RA (através da sinergia com outros campos ou tecnologias) poderia chegar a derivar novos subconjuntos de RA no futuro próximo, como explica a seguir o artigo de Wu et al. (2013): “Com a rápida evolução da tecnologia, o conceito de RA poderia ser ampliado ainda mais, porque mais e mais dispositivos hardware e software poderiam ser utilizados para criar RA”.

2.2 Tecnologias de Realidade Aumentada

Nesta seção são apresentadas as tecnologias de RA, apresentando tanto o hardware que pode ser usado para este tipo de aplicativos quanto software que permite seu desenvolvimento.

2.2.1 Hardware de Realidade Aumentada

O hardware de RA se refere aos tipos de dispositivos que podem ser usados para aplicar esta tecnologia. Um dos aspectos mais citados em pesquisas quanto ao hardware de RA é a tecnologia de *Display* usada ou o *Display* de RA, ou que se refere à tecnologia disponível para a apresentação da RA. Nota-se que a RA não é uma tecnologia limitada ao sentido visual, no entanto é claro que a maioria de pesquisas e projetos de RA existentes estão voltados para este sentido. Assim na teoria de hardware deste trabalho é levado em consideração unicamente *Visual Display* de RA (*Display* de RA relacionado com o sentido visual), deixando de lado tipos de *Display* sobre outros sentidos tais como *Aural Display* (relacionado com a audição) e *Haptic Display* (relacionado com o tato).

Para o estudo de *Visual Display* de RA*,* toma-se como referência principal o artigo de Van Krevelen & Poelman (2010), no qual foram classificados os diferentes tipos (ou técnicas) de *Visual Display* para a RA, sendo eles: *Video See-Through* (VST), *Optical See-Through* (OST), e *Projective*. Estes são resumidos a seguir:

- ***Video See-through*** **(VST):** Através desta técnica para apresentar a RA, o mundo real é substituído por um vídeo da realidade, e elementos virtuais se sobrepõem às imagens capturadas (o usuário não vê diretamente a realidade senão um vídeo dela aumentado com conteúdo virtual). Nota-se que esta técnica consiste na mais simples de implementar e mais barata das apresentadas. “Uma vez que a realidade é digitalizada, é mais fácil interpor e remover objetos da realidade” (VAN KREVELEN & POELMAN, 2010). Um fato importante que permite diferenciar esta técnica da OST é o seguinte: “A imagem do vídeo é misturada com algum conteúdo virtual (como você verá em um filme) e enviada de volta para uma tela padrão [...]” (GRUBERT & GRASSET, 2013), ou seja, a mistura dos mundos real e virtual acontece realmente na tela do dispositivo, isto é descrito na figura 2.4.

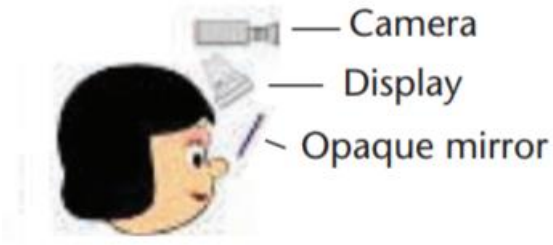

Figura 2.3: Conceito de VST. (AZUMA et al., 2001)

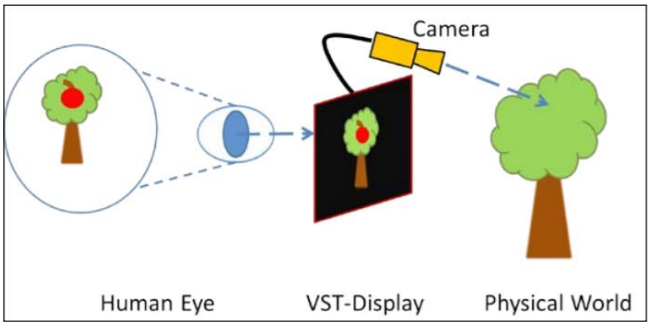

Figura 2.4: VST, mistura dos mundos real e virtual acontecendo na tela do dispositivo. (GRUBERT & GRASSET, 2013)

Figura 2.5: Exemplo de VST *Display* através de um dispositivo móvel. (REALIDAD AUMENTADA, 2017)

- ***Optical See-Through* (OST):** Elementos virtuais (ou sobreposição de RA) são apresentados sobre lentes e espelhos transparentes, onde se mantem a visão da realidade (o usuário vê diretamente a realidade), porém com conteúdo virtual sobreposto sobre ela. Uma característica particular desta técnica é que pode ser considerada muito segura, pois os usuários sempre podem continuar vendo ainda quando existe alguma falha de energia, sendo muito útil para sistemas com fins

médicos ou militares onde resultaria muito crítico criar obstáculos ao sentido da visão mesmo que seja por alguns poucos segundos. Um fato importante que permite diferenciar esta técnica da VST é o seguinte: “A mistura dos mundos real e virtual não acontece na tela do dispositivo, mas diretamente na retina do olho do usuário [...]” (GRUBERT & GRASSET, 2013). Isto pode ser visto na Figura 2.7.

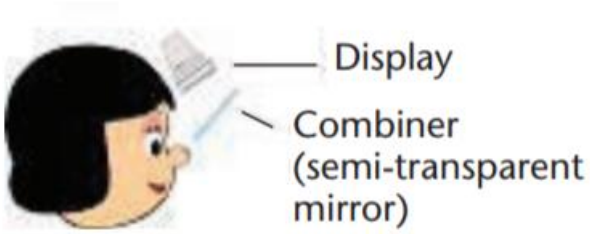

Figura 2.6: Conceito de OST. (AZUMA et al., 2001)

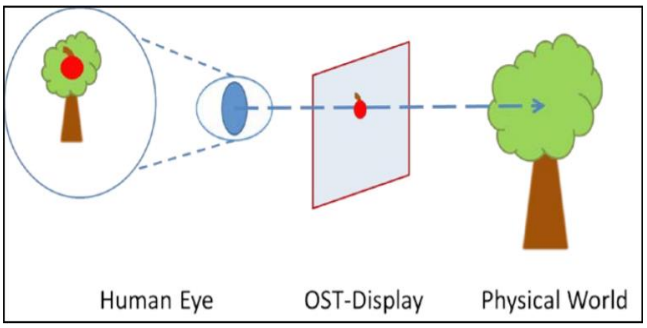

Figura 2.7: OST, mistura dos mundos real e virtual acontecendo só na retina do olho do usuário. (GRUBERT & GRASSET, 2013)

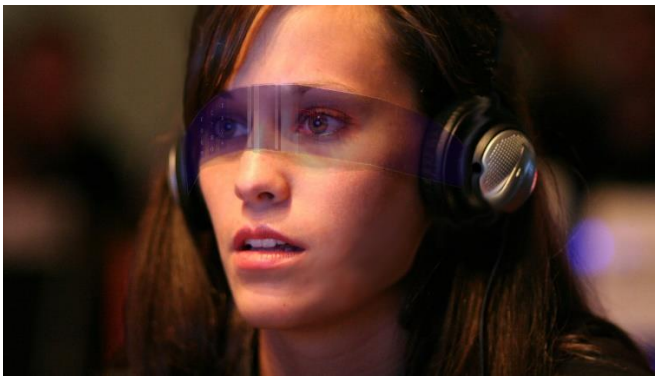

Figura 2.8: Exemplo de OST *Display* através de um OHMD. (OPTICAL HEAD-MOUNTED DISPLAY, 2017)

- ***Projective:*** Os elementos virtuais (ou sobreposição de RA) são projetados sobre os elementos reais (o usuário vê diretamente os

elementos virtuais sobre os elementos do mundo real através de projeções).

Figura 2.9: Exemplo de *Projective Display*. (HOLMES, 2010)

Os mesmos autores também classificam o *Visual Display* de acordo com a sua posição entre o usuário visualizador e o ambiente real, assim, de acordo com a posição, foram definidos três tipos possíveis de *Visual Display* que são: *Head-worn*, *Hand-held* e *Spatial*. Nota-se que cada um dos tipos de *Visual Display* da primeira classificação (VST, OST, e *Projective*) pode se encaixar em qualquer um dos tipos desta nova classificação de acordo com a sua posição (*Head-worn*, *Hand-held* e *Spatial*), existindo diferentes combinações possíveis de *Visual Display* (ver Figura 2.10). Esta nova classificação baseada no posicionamento do *Visual Display* é resumida a seguir:

- ***Head-worn:*** O *Display* é colocado na cabeça do usuário. Idealmente os *Head-worn Display* de RA não deveriam ser maiores do que um par de óculos (AZUMA et al., 2001).

- ***Hand-held*****:** O *Display* é levado na mão do usuário. *Hand-held Display* funciona como uma janela ou lupa que mostra objetos reais com a sobreposição de RA (AZUMA et al., 2001).

- ***Spatial*****:** O *Display* está colocado no ambiente de um jeito estático. Estas técnicas são adequadas para grandes apresentações e exibições com interação limitada (VAN KREVELEN & POELMAN, 2010).

A Figura 2.10 apresenta exemplos de sistemas de RA levando em consideração os dois tipos de classificações apresentadas, ou seja, classificação segundo o tipo (ou técnica) de *Visual Display* (VST, OST, e *Projective*), e segundo o posicionamento do *Visual Display* (*Head-worn*, *Hand-held* e *Spatial*).

Figura 2.10: Classificação dos tipos de *Visual Display* de RA. (TOBAR et al., 2013)

2.2.2 Software de Realidade Aumentada

Ao longo dos anos, muitas ferramentas que permitem criar software com RA têm sido criadas por desenvolvedores e pesquisadores, sendo disponibilizados com diferentes licenças. A seguir serão apresentadas algumas dessas ferramentas que foram pesquisadas no presente trabalho. É claro que as ferramentas apresentadas não são as únicas existentes, no entanto podem ser consideradas como algumas das ferramentas importantes e de grande utilidade para criar este tipo de software.

Deve-se notar que na criação de software com RA, embora existem muitas ferramentas, diferente do que acontece em outros tipos de software, na maioria dos

casos cada uma dessas ferramentas tem características muito diferentes das outras e fornece um suporte específico para algum tipo de RA, ou para certas plataformas, por exemplo: Para *Desktop* AR, RAM, RA seguindo a abordagem de marcadores (RA baseada em marcadores), ou seguindo a abordagem de localização, etc. Assim muitas vezes a escolha de uma ferramenta específica vai depender do tipo de RA que vai ser implementada e da plataforma alvo da aplicação. Comentários similares em relação a isso também podem ser encontrados em pesquisas existentes, como o seguinte:

> "As ferramentas de RA são difíceis de comparar, devido a que algumas delas são especializadas para uma finalidade (p.ex. *tracking* baseado em marcadores ou ambientes móveis), algumas só suportam certas plataformas (p.ex. Windows ou iOS) e outras suportam várias plataformas e são usadas para vários fins". (SILTANEN, 2012).

As ferramentas de RA pesquisadas foram:

- ARToolkit (ARTOOLWORKS, 2017a): Consiste em uma das ferramentas ou biblioteca de software mais conhecidas e utilizadas para criar software de RA, foi inicialmente desenvolvida por Hirokazu Kato, e está sendo mantida pelo Human Interface Technology Laboratory (HIT Lab) da Universidade de Washington, HIT Lab NZ da Universidade de Canterbury, Nova Zelândia, e ARToolworks, Inc, Seattle. ARToolkit permite a criação de software de RA realizando o processo de *tracking* (ver Seção 2.3) especificamente através de marcadores, assim, está voltado para criar aplicativos de RA baseada em marcadores. Por outro lado, ARToolkit é usada para *Desktop AR*, sendo suportadas diferentes plataformas (Windows, Linux, Mac OS X, SGI), e permitindo a codificação através da linguagem de programação C++. Atualmente está disponível de forma *Open Source* com licença GPL, e também através de licença comercial, e deve-se notar que, ao longo dos anos, muitas outras bibliotecas têm sido derivadas a partir dela, (NyARToolkit, FLARToolKit etc.) que permitem suportar as características de ARToolkit em uma grande variedade de plataformas.

- NyARToolkit (NYARTOOLKIT, 2017): Projeto ou biblioteca desenvolvida no Japão, e baseada no ARToolkit, que permite criar software de RA para várias plataformas (tanto *Desktop* quanto móveis), tendo como característica principal, semelhante a ARToolkit, de ser voltado para RA baseada em marcadores. NyARToolkit tem suporte para várias linguagens de programação como Java, C# e ActionScript, sendo disponível de forma *Open Source* com licença GPL, com algumas funcionalidades disponíveis na licença MIT, e também por meio de licença comercial.

- FLARToolKit (ARTOOLWORKS, 2017b): Projeto ou biblioteca baseada no ARToolkit considerada como a versão do ARToolkit para Adobe Flash ActionScript 3, que permite criar software de RA na web. Sendo uma versão portada do ARToolkit, também está voltada para RA baseada em marcadores e atualmente está disponível de forma *Open Source* com licença GPL, e também através de licença comercial.

- Layar (LAYAR, 2017): Ferramenta que permite o desenvolvimento de software de RA para dispositivos móveis (iOS e Android) sobrepondo uma camada com informação virtual, e fazendo uso da câmara e sensores tais como GPS, acelerômetro e bússola para obter a posição e orientação do dispositivo, assim, uma vez que se tem a posição e orientação do dispositivo, pode se sobrepor informação virtual em certos pontos quando são apontados pelo dispositivo. Os pontos de referência (pontos que devem ser aumentados com conteúdo virtual), assim como a informação virtual de aumento, serão fornecidos externamente por um servidor. Nota-se que, diferentemente das ferramentas anteriores, essa ferramenta está voltada para RA seguindo a abordagem de localização, e está disponível unicamente com licença comercial. Este tipo de ferramenta é muito usado na atualidade para aplicações de RAM em lugares abertos, como aplicações de busca, guias turísticas etc.

- Vuforia (PTC, 2017): É um SDK (*Software Development Kit*) que permite a criação de software de RA para dispositivos móveis e óculos digitais. As plataformas suportadas são Android, iOS e UWP (*Universal Windows Platform*), e está disponível para o desenvolvimento através de Android Studio, XCode, Visual Studio, e o *game engine* Unity[3]. Vuforia está voltado para RA baseada em marcadores, visto que usa visão computacional para reconhecer e rastrear marcadores (*tracking*), ainda sendo suportados o rastreamento de alguns objetos simples tais como caixas. Vuforia consiste em uma ferramenta muito poderosa para a RA em dispositivos móveis, e são fornecidas características adicionais de muita utilidade, como *Virtual Buttons* (que permitem usar áreas específicas do marcador para executar eventos quando for cobertas por um objeto), reconhecimento de palavras em inglês, reconhecimento do terreno do mundo real (*Smart Terrain*), etc. Finalmente, Vuforia está disponível atualmente com licença comercial e uma opção *free* para desenvolvimento sem fins lucrativos.

Finalmente, após pesquisar e analisar diferentes ferramentas, foi escolhida Vuforia como a plataforma de RA para suportar as características de rastreamento de marcadores, sendo utilizada sobre o *game engine* Unity, o qual será usado como ferramenta principal para criar o *framework* proposto.

2.3 Tracking

Uma vez que a RA precisa ser registrada em 3D, ou seja, precisa alinhar elementos virtuais com reais de modo que pareçam coexistir no mesmo mundo e apresentá-los de um jeito interativo sem que os elementos virtuais permaneçam estáticos ou fixos em uma determinada posição, é preciso realizar o processo de *tracking*. Em relação a esta necessidade da RA, se tem o seguinte: “A RA precisa

[3] Unity é um *game engine* que permite o desenvolvimento de *videogames* e simulações para computador, consoles de jogo e dispositivos móveis. Para mais informações ver em: https://en.wikipedia.org/wiki/Unity_(game_engine)

conhecer sobre o conteúdo real e virtual. Precisa conhecer onde as coisas estão no espaço (*Registration*) e rastrear aonde elas estão se movimentando (*Tracking*)" (GRUBERT & GRASSET, 2013). Nesse contexto, Siltanen (2012) opina que: "Na prática, os sistemas de RA precisam determinar a localização e orientação da câmera. Com uma câmera calibrada, o sistema é capaz de renderizar objetos virtuais no lugar correto". Ainda sobre essa necessidade, também se tem a seguinte afirmação:

> "A RA requer um posicionamento e um rastreamento de orientação muito precisos para alinhar, ou registrar, informações virtuais com os objetos físicos que estão sendo anotados. Sem isso, é bastante difícil enganar os sentidos humanos para acreditar que os objetos virtuais gerados por computador coexistem no mesmo espaço físico que os objetos do mundo real". (PAPAGIANNAKIS et al., 2008).

Uma vez esclarecida a anterior, várias definições de *tracking* são apresentadas a seguir: "O termo *tracking* significa calcular a pose relativa (localização e orientação) de uma câmara em tempo real. Isto é um dos componentes fundamentais da RA" (SILTANEN, 2012). Outra definição sugere que: "*Tracking* refere-se ao processo de rastrear a posição e rotação do usuário, de acordo com a cena que está sendo observada" (TOBAR et al., 2013).

Nota-se a importância do *tracking* para a RA, pois se este processo não fosse levado em consideração, os elementos virtuais aumentos ficariam totalmente estáticos para a nossa visão do mundo. Esta situação foi analisada em Grubert & Grasset (2013), onde afirma-se que nesse caso:

> "[...], qualquer conteúdo virtual (tal como textos, ou imagens) permanecerá fixo na sua posição na tela. A superposição será realmente estática, seu *Display* de RA atuará como um *Head-Up Display*[4] (HUD), porém não será realmente RA [...]". Ver Figura 2.11.

[4] HUD é um *Display* transparente que apresenta dados sem precisar que os usuários se afastem de seus pontos de vista. Para mais informações ver em: https://en.wikipedia.org/wiki/Head-up_display

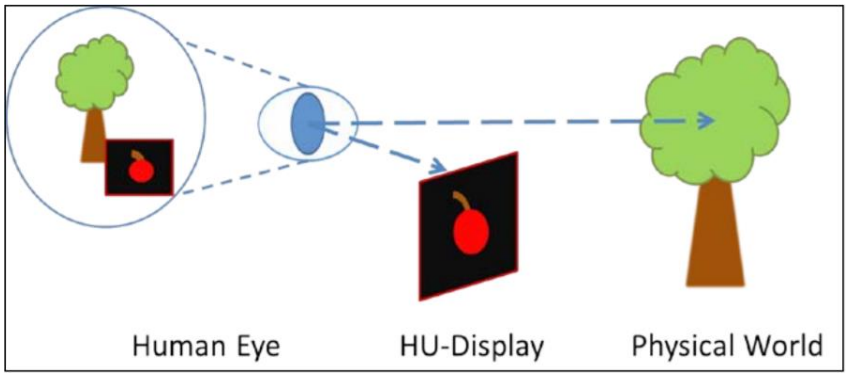

Figura 2.11: Superposição de elementos virtuais acontecendo de jeito estático através de um HUD. (GRUBERT & GRASSET, 2013)

Além da anterior, nota-se que esta característica de *tracking* é particular da tecnologia de RA, e não está presente em outros tipos de tecnologias ou técnicas que também fazem mistura de elementos reais e virtuais. Por exemplo, em filmes de ficção científica, onde são frequentemente misturados elementos reais e virtuais, o processo de *tracking* não existe pois os elementos virtuais são acrescentados na fase de pós-produção após ter sido feita a captura de cena (após gravação das cenas). Assim, consequentemente, se tem que: “Em um filme, você apenas olha passivamente para a cena do ponto de vista onde ela foi filmada. Em RA, você pode-se mover ativamente ao redor, para a frente, e retroceder e virar seu *Display* de RA — você ainda verá um alinhamento entre os mundos” (GRUBERT & GRASSET, 2013). Em Grubert & Grasset (2013) também foi comentado outro exemplo similar onde se mistura realidade com virtualidade sem existir o processo de *tracking*, que é quando se apresentam anúncios em televisão com banners de texto na parte inferior da tela (o qual ainda pode acontecer em tempo real como na RA), assim, uma vez que não existe processo de *tracking*, os elementos virtuais ficam estáticos sendo apresentados de jeito similar ao caso da Figura 2.11.

2.4 Realidade Aumentada Baseada em Marcadores

Neste tipo de RA, o processo de *tracking* é feito através de marcadores de RA (esta abordagem de RA é chamada *tracking* baseado em marcadores). Uma simples definição deste elemento é a seguinte: “Marcadores são imagens que podem ser detectadas por uma câmara e um software como localização de elementos virtuais

colocados em uma cena" (KATIYAR et al., 2015). Outras definições mais técnicas são também encontradas na literatura, como a apresentada a seguir:

> "Um marcador é um sinal ou imagem que um sistema computacional pode detectar desde uma imagem de vídeo usando processamento de imagem, reconhecimento de padrões e técnicas de visão computacional. Uma vez detectado, se define tanto a escala correta quanto a pose da câmera". (SILTANEN, 2012).

Um marcador de exemplo pode ser visto na Figura 2.12 a seguir.

Figura 2.12: Exemplo de marcador de RA tradicional. (GAO et al., 2017)

Assim, na RA baseada em marcadores, através de uma câmara de qualquer tipo (câmera web, câmara integrada em *smartphones*), são capturadas imagens do mundo real em busca do marcador e, uma vez encontrado e determinado a sua posição e orientação (*tracking*), o elemento virtual é sobreposto sobre as imagens capturadas de jeito adequado, dando a impressão de estar convivendo com os elementos do mundo real. Nota-se que a RA baseada em marcadores não depende de *Displays* específicos. Nas Figuras 2.13, 2.14 e 2.15 podem ser vistos exemplos desta tecnologia com diferentes *Displays* de RA.

Figura 2.13: Exemplo de RA baseada em marcadores com um *Display* de tipo *Spatial* através de um *Desktop.* (SLIJEPCEVIC, 2016)

Figura 2.14: Exemplo de RA baseada em marcadores com um *Display* de tipo *Hand-held* através de um dispositivo móvel. (MULTIDOTS.COM, 2015)

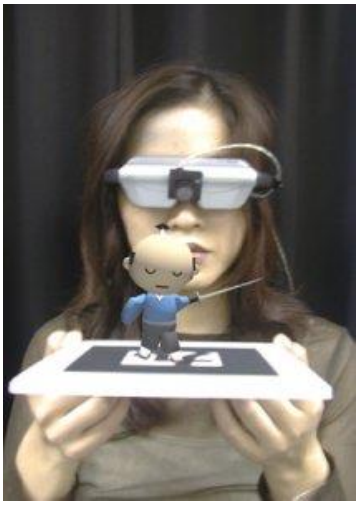

Figura 2.15: Exemplo de RA baseada em marcadores com um *Display* de tipo *Head-worn* através de um dispositivo HMD. (ARTOOLWORKS, 2017a)

2.5 Marcadores De Controle

Um Marcador de Controle pode ser definido como um marcador adicional que tem como único objetivo ativar algum tipo de evento sempre que ficar visível pela câmera junto ao marcador principal vinculado ao objeto. Eventos comuns ativados através de marcadores de controle podem ser: trocar o elemento virtual por outro, acrescentar novos elementos sobre ele, mudar alguma característica particular etc. Na Figura 2.16, apresenta-se o processo do uso de um marcador de controle para trocar um elemento em RA (Texto 3D) por outro novo (Novo Texto 3D), deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o objeto original voltará a ficar na cena como estava inicialmente.

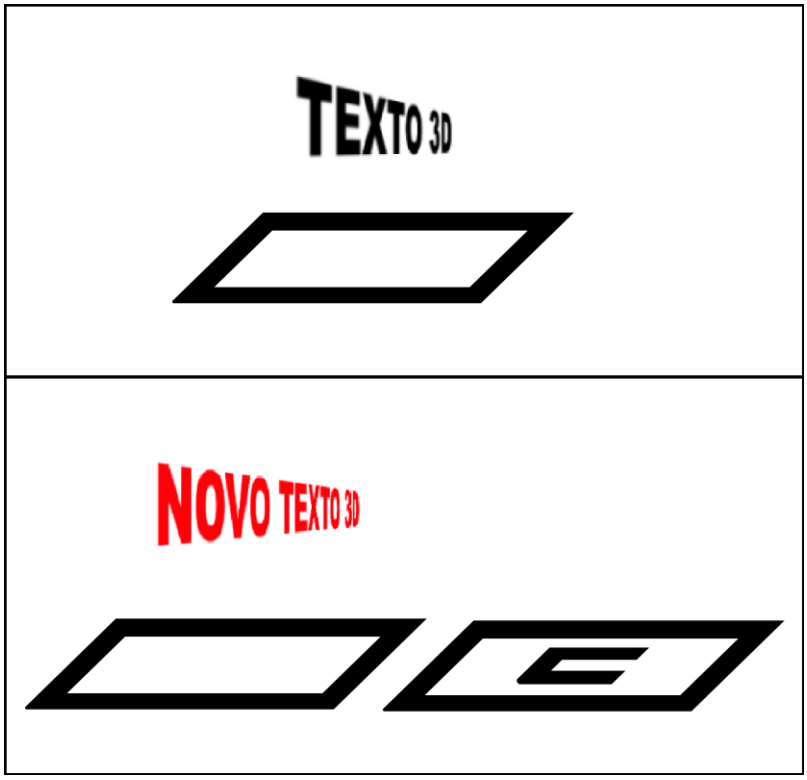

Figura 2.16: A imagem acima apresenta um objeto em RA (Texto 3D) com seu marcador vinculado já reconhecido pelo sistema, a imagem abaixo apresenta a mesma cena acrescentando um marcador de controle "C", o qual faz trocar o objeto original por outro (Novo Texto 3D).

Um exemplo do uso de marcadores de controle pode ser visto no projeto SICARA em Kirner (2013), na aplicação chamada "Aritmetica", onde um marcador principal apresenta uma imagem em RA com uma pergunta da área de aritmética, e, uma vez que um marcador de controle "C" também for visível pelo sistema

colocando-o junto do marcador principal, então o resultado da pergunta ficará disponível. Este processo, em detalhe, pode ser visto nas Figuras 2.17 e 2.18.

Figura 2.17: A figura esquerda apresenta o marcador principal ainda sem ser reconhecido pelo sistema, já na figura direita o marcador principal é reconhecido pelo sistema e apresenta a imagem em RA com a pergunta. (KIRNER, 2013)

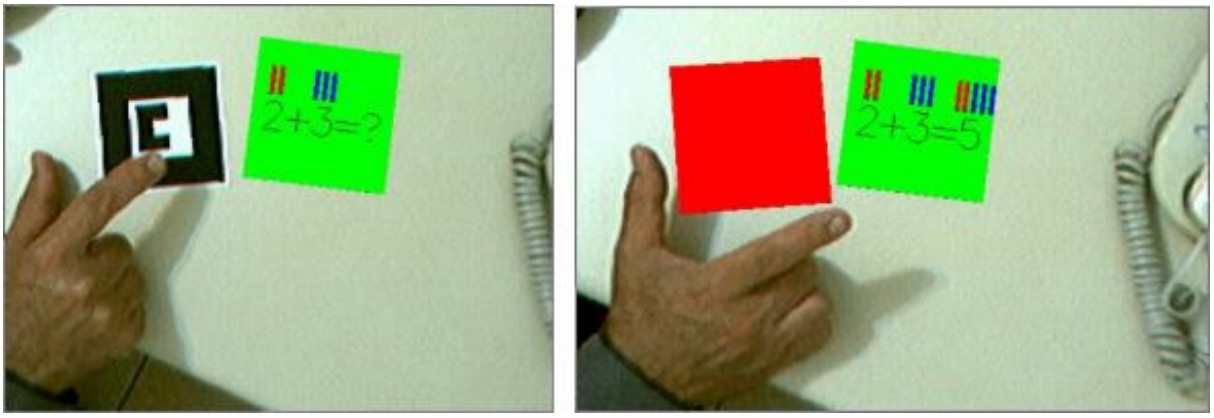

Figura 2.18: A figura esquerda apresenta o marcador principal e o de controle "C", porém o marcador de controle ainda sem ser reconhecido pelo sistema. Já a figura direita apresenta o marcador de controle sendo reconhecido pelo sistema e ativando o resultado da pergunta. (KIRNER, 2013)

Através da pesquisa realizada, foi possível concluir que os marcadores de controle são de muita utilidade nas aplicações de RA baseada em marcadores, e ainda mais nas aplicações educacionais, nas quais fornecem a possibilidade ao aprendiz de realizar diferentes eventos sobre os objetos em RA que estão sendo estudados, permitindo maiores possibilidades de exploração. Finalmente, devido a sua relevância na RA baseada em marcadores e em contextos educacionais, o apoio à utilização de marcadores de controle foi um dos objetivos principais na definição das funcionalidades providas pelo *framework* proposto (ver Seção 4.1.3).

2.6 Experiências Prévias com Realidade Aumentada

Nesta seção é apresentado um breve resumo de experiências prévias ou aplicações desenvolvidas com tecnologia de RA para contextos educativos, tanto com RA em geral quanto com RAM. Estas experiências são resultado de pesquisas feitas sobre sistemas com RA aplicada à educação. É evidente que não foi feito um levantamento de todas as pesquisas existentes sobre o tema, mas sim de algumas pesquisas significativas que permitiram fornecer uma ideia geral do que pode ser conseguido quando se aplica a tecnologia da RA no contexto educativo. Este estudo foi útil para entender como pode ser utilizada a RA e especificamente a RAM para criar conteúdo educativo, que tipos de aplicativos existem, quais são suas características e suas tendências, etc. Incialmente são apresentadas experiências prévias com RA na educação (com qualquer tipo de dispositivos hardware e não especificamente para dispositivos móveis), e, finalmente, são apresentadas experiências especificamente com RAM na educação, que é o contexto onde está inserido o presente trabalho.

2.6.1 Experiências Prévias com Realidade Aumentada na Educação

O artigo de Blum et al. (2012) consiste em um aplicativo chamado *Mirracle,* o qual fornece um novo cenário para estudar a anatomia humana de uma forma intuitiva e fazendo uso da RA. *Mirracle* tem como característica principal permitir a utilização de um espelho mágico no qual modelos 3D de órgãos internos do corpo humano são sobrepostos sobre o usuário em tempo real através da RA, isto permite criar a ilusão de estar vendo as partes internas do seu próprio corpo, facilitando o estudo da anatomia humana. Para isso foi aplicado um novo conceito de interação baseada em gestos, onde, através de gestos, diferentes fatias de Tomografia Computadorizada (TC) e um conjunto de dados fotográficos podem ser selecionados para visualização.

Além da característica anterior de espelho mágico, o sistema (através de outro modo onde não se usa o espelho mágico) também permite mostrar modelos 3D de órgãos, informação textual, e imagens sobre anatomia. Para interagir com

esses dados (rotacionar, dar zoom), apresenta-se uma nova metáfora de interação que faz uso da câmera de profundidade, chamada *Metaphor of Frosted Glass* (ou metáfora do vidro fosco), esta metáfora consiste em que os objetos sejam visualizados similares a como se existisse um vidro fosco na cena. Assim, objetos que estão muito longe do vidro não são vistos, objetos mais perto do vidro são vistos desfocados, e objetos que tocam o vidro são bem vistos. Ao usar a metáfora para uma interface de usuário, informações como textos se apresentam como se estivessem sobre o vidro, e a mão do usuário como se estivesse atrás do vidro, permitindo ações como selecionar um ponto de um jeito mais intuitivo.

O hardware utilizado pelo *Mirracle* consiste em um dispositivo de visualização que funciona como o espelho, com uma câmera colorida colocada na superfície da tela desse dispositivo de visualização, e uma câmera de profundidade junto à câmera colorida. Através das câmeras se captura a cena real onde está o usuário, e imagens obtidas são apresentadas no dispositivo de visualização, o qual se torna parecido com um espelho normal, no entanto o fato de acrescentar elementos virtuais com RA que se sobrepõem ao corpo do usuário na cena real torna o espelho um espelho mágico.

Figura 2.19: *Mirracle.* (BLUM et al., 2012)

No artigo de Kaufmann (2002) apresenta-se *Construct3D*, que é uma ferramenta de construção de Geometria tridimensional para a educação nas áreas de Matemáticas e Geometria, a qual visa melhorar as habilidades espaciais e maximizar a transferência de aprendizagem, podendo ser usada tanto em ensino médio quanto em educação universitária. *Construct3D* permite manter comunicação cara a cara e de forma colaborativa entre professor e estudantes, ao mesmo tempo

que os estudantes podem visualizar objetos que normalmente tinham que criar com métodos tradicionais. O sistema fornece um conjunto de funcionalidades para a construção de primitivas geométricas tais como pontos, linhas, planos, cubos, esferas, etc. As funcionalidades de construção incluem interseções, linhas normais e planos, operações de simetria, etc.

Para a visualização, utiliza-se um dispositivo HMD, e o processo de criação das figuras geométricas é através de manipulação 3D usando uma *Stylus* com 6 graus de liberdade e um *Personal Interaction Panel* (PIP) para interação com elementos. O PIP é usado para permitir a integração de elementos de interfaces 2D tradicionais como botões, controles deslizantes, etc., sendo necessário para operações adicionais como selecionar figuras, carregar, remover, etc. Usando esta forma de interação, o aprendiz pode criar os elementos de um jeito intuitivo e muito parecido como que se estivesse utilizando uma caneta normal.

Figura 2.20: Estudantes trabalhando com *Construct3D*. (KAUFMANN, 2002)

O artigo de Forte & Kirner (2009) consiste em um software de RA chamado *ARTutor*, que permite auxiliar no ensino de tópicos de Física e Matemática, sendo criado a partir das características do sistema de autoria em RA, SACRA (SANTIN, 2008).

O sistema *ARTutor* utiliza um computador para a visualização, e marcadores de RA, permitindo a realização de diversas tarefas nas duas áreas de estudo definidas. Por exemplo, adotando uma tarefa de sólidos geométricos da área de Matemática, inicialmente são mostrados os tópicos mais importantes do assunto em

questão, que consiste na parte teórica ou explicação do conceito (no caso informação das figuras 3D e fórmulas), o usuário terá uma opção de desafio para realizar uma atividade relacionada à questão de estudo, como construir uma figura 3D (p.ex., um cubo) com as medidas certas para resolver o exercício, adicionalmente o usuário tem a possibilidade de experimentar com RA e visualizar o modelo 3D da figura que ele criou colocando o marcador na vista da câmara do computador. Além disso, o estudante poderá alterar o objeto virtual apresentado (utilizando um marcador de controle) por um texto 3D que apresenta o resultado do exercício, indicando também se o exercício estava certo ou errado. Nota-se que a opção de experimentação através de RA dependerá totalmente do tipo de exercício, assim no caso em que o assunto escolhido for de Física, como, por exemplo, a queda livre, o estudante poderá visualizar em 3D o movimento de queda de uma esfera virtual desde uma altura configurável com o objetivo que possa estudar o comportamento da mesma.

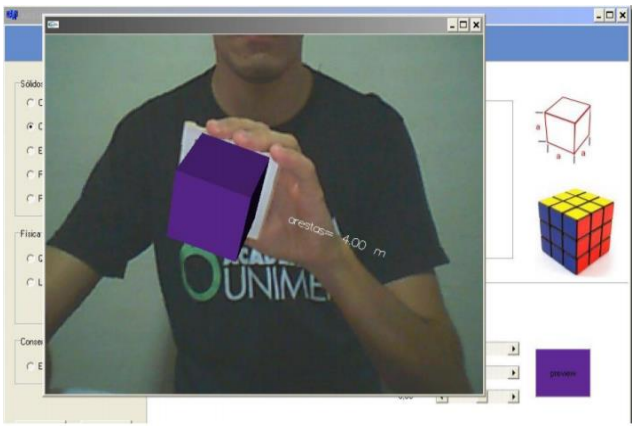

Figura 2.21: Usuário interagindo com objeto virtual criado com *ARTutor* (FORTE & KIRNER, 2009)

2.6.2 Experiências Prévias com Realidade Aumentada Móvel (RAM) na Educação

No artigo apresentado por Mahale & Yeddu (2016), é proposto um aplicativo de RAM baseado em marcadores para a plataforma Android, o qual permite a apresentação de objetos virtuais na sala de aula tais como frutas, vegetais, e animais em 3D, visando descrever esses elementos em maior profundidade para crianças de escola primária. Esses elementos normalmente são apresentados pelo professor sem fazer uso de RA e com métodos de ensino tradicional na lousa,

imagens de livros, ou ainda através de ferramentas como PowerPoint. A RA, ao contrário de outros métodos e ferramentas, permite que os elementos sejam apresentados como se estivessem realmente na sala de aula, sendo possível mudar sua escala e serem visualizados a partir de diferentes posições, tais como: a esquerda, a direita, acima e abaixo. Para o desenvolvimento do aplicativo e a criação dos marcadores, foram utilizadas as ferramentas Unity em conjunto com Vuforia, adicionalmente, para criar os modelos 3D, foi utilizado o software Blender.

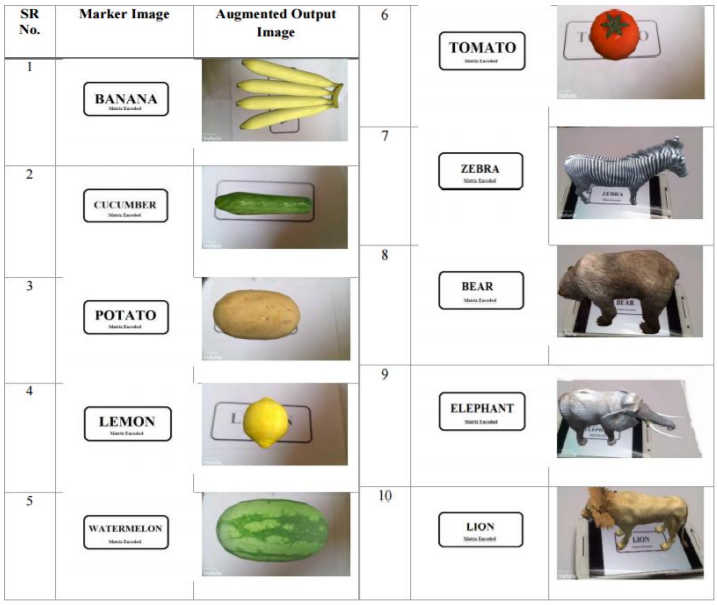

Figura 2.22: Marcadores e objetos aumentados com o aplicativo. (MAHALE & YEDDU, 2016)

O artigo de Chowdhury et al. (2013) apresenta *The Science Expedition: The Pop-Up Adventure multimedia and augmented reality* (SEPA-M e SEPA-AR), que consiste em duas aplicações de aprendizado móvel com conteúdo baseado no programa de estudos de ciências de estudantes de quarta série. SEPA-AR consiste em um aplicativo móvel de RA, e SEPA-M em um aplicativo móvel multimídia. Nesta pesquisa, é aprofundado o aplicativo SEPA-AR, o qual é do nosso interesse, uma vez que é um aplicativo móvel com RA. SEPA-AR consiste no uso de um livro que possui marcadores de RA, onde, uma vez que um marcador é detectado pela câmera do dispositivo móvel, elementos virtuais que correspondem a um nível de estudo específico são apresentados sobre o dispositivo como elementos aumentados que se sobrepõem à realidade. No aplicativo, apresentam-se vários níveis de estudo com conteúdo correspondente ao programa de estudos de ciências de quarta série, visando apresentar um material mais claro e interativo que permita

melhorar o aprendizado das crianças. O tipo de informação que o aplicativo pode inserir no mundo real, uma vez detectado o marcador, é de tipo visual e auditivo, apresentando-se imagens 2D de elementos relacionados com o tema, sons relacionados aos elementos, e, finalmente, vídeos. O usuário pode interagir com o aplicativo através do botão do menu, e escolher a informação que quer aumentar em um momento dado (ou seja, escolher o objeto, som ou vídeo). Quanto às ferramentas de desenvolvimento do aplicativo, foi usado OpenGL ES (KHRONOS GROUP, 2017b) para desenhar os diferentes objetos (moedas de metal, uma cadeira de madeira, uma bolsa de couro, brinquedos de plástico, elásticos, e pneus) e ARToolkit para desenvolver a visão de RA do aplicativo e suportar as funcionalidades relacionadas com RA. A linguagem de codificação usada foi Java.

Figura 2.23: Menu de funções com SEPA-AR em funcionamento. (CHOWDHURY et al., 2013)

No artigo apresentado por Wagner & Barakonyi (2003), apresenta-se um jogo colaborativo de RAM para o aprendizado de símbolos kanji chamado *AR Kanji*. Visando que o aplicativo fosse de tipo móvel, e além disso, atingir um público amplo, ele foi projetado para *Personal Digital Assistants* (PDA), rodando juntamente com um módulo de rastreamento baseado em marcador óptico. O aplicativo foi um dos primeiros jogos de tipo *Hand-held AR* que foi possível executar de um jeito totalmente autônomo através de PDAs não modificados e equipados com câmera e WiFi. O jogo consiste em que dois jogadores se situem juntos um em frente ao outro, com uma pilha de cartas, e cada uma delas tem um kanji em ambos os lados. Na parte da frente de uma carta, além do símbolo do kanji, está cercada por uma borda preta para permitir o reconhecimento do marcador e o processo de aumento quando o dispositivo a detectar. O fluxo do jogo é por turnos, iniciando com um grupo de cartas com o verso para cima. O dispositivo apresenta um ícone de um elemento

(por exemplo, uma árvore) e pede para o jogador em turno que procure o kanji desse elemento no grupo de cartas, uma vez que o jogador escolhe a carta, ele deve virá-la para ficar a parte da frente visível à câmara, assim o aplicativo reconhece o marcador e o elemento em questão é apresentado em 3D através de RA. Se o jogador escolheu a carta correta, ele ganha um ponto, mantém a carta, e poderá continuar com a próxima busca, no caso de errar, o jogador deve deixar a carta com o grupo e o jogo passa para o turno do oponente. Para o desenvolvimento do aplicativo, foi utilizado um *framework* de *Hand-held AR* próprio dos autores. O sistema executa uma versão nativa portada e com velocidade optimizada de ARToolkit. O processo de optimização de velocidade foi feito utilizando a biblioteca chamada *Intel Fixed Point library* que é parte de *Intel* GPP ("Intel Graphics Performance Primitives for the Intel® PXA250 Applications Processor," n.d.). Por outra parte, a renderização é feita com SoftGL (WAGNER, n.d.), o qual consiste em um subconjunto OpenGL (KHRONOS GROUP, 2017a) desenvolvido *in-house* que implementa as primitivas gráficas 3D principais.

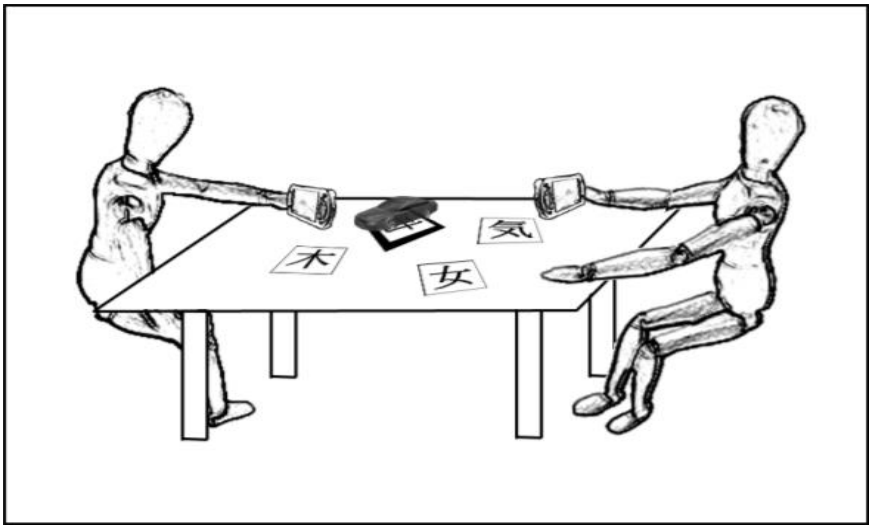

Figura 2.24: *AR Kanji* mostrando um carro. (WAGNER & BARAKONYI, 2003)

2.7 Realidade Aumentada na Educação

Antes de falar sobre RA na educação, é preciso saber que a RA não é uma tecnologia tão nova quanto parece, o termo RA foi proposto na década de noventa, já desde essas datas era pesquisada por muitos autores e vários sistemas foram

desenvolvidos fazendo uso dela, no entanto sua aplicabilidade estava muito restringida devido, principalmente, ao hardware necessário e os altos custos inerentes. Um usuário comum tinha pouco ou nenhum conhecimento sobre esse tópico, no entanto no transcurso dos anos, a evolução dos computadores, e principalmente o surgimento de dispositivos móveis (tais como *smartphones* e *tablets)* cada vez mais potentes e mais acessíveis ao usuário comum, permitiu que a RA tivesse mais linhas de atuação e fosse utilizada em ambientes antes impensados. Apoiando as afirmações anteriores se tem o seguinte:

> "A RA que até faz pouco tempo era uma tecnologia experimental, restringida ao entorno de especialistas técnicos e pesquisadores, se torna cada vez mais acessível. Atualmente, diversos dispositivos já disponíveis para um setor da população Ibero-americana e que tendem a atingir uma maior penetração (móveis, consoles de jogos, PDA e *tablet*PC, etc.) já possuem as ferramentas necessárias para implementar RA". (GARCÍA et al., 2010).

Levando em consideração o apresentado no parágrafo anterior, uma vez que a RA tornou-se mais acessível e mais conhecida, sua aplicabilidade no decorrer dos anos tem sido massiva em inúmeras áreas como turismo, publicidade, entretenimento, e aos poucos também se tem começado a inserir na educação, apresentando ganhos e abrindo novos cenários de aprendizado antes não contemplados. Em relação a isso tem-se que:

> "O que uma vez foi visto por muitos como um mero truque com poucas aplicações fora do treinamento, *marketing*/PR (*Public Relations*), esporte e entretenimento, agora está se tornando uma corrente principal com oportunidades reais para que seja usado para fins educativos". (FITZGERALD et al., 2013).

Pelo seu potencial, nos últimos anos, a RA está se tornando de interesse para pesquisadores e desenvolvedores no cenário educativo, ainda esta tecnologia foi listada pelo Relatório Horizont 2010 (GARCÍA et al., 2010) como uma das tecnologias emergentes a serem vistas e que em próximos anos teria um uso muito

difundido para o ensino, aprendizado e pesquisa criativa. A seguir apresentam-se algumas das principais vantagens de aplicar a RA no contexto educativo.

Uma das vantagens mais citadas da RA na educação é o alto nível de interesse e motivação inerente que a própria tecnologia produz nos estudantes, sendo uma tecnologia futurista que parece tirada de filmes de ficção científica. Em relação a isso, tem-se o seguinte: “Se tem demostrado que os ambientes de RA poderiam aumentar a motivação e interesse dos alunos, o qual por sua vez poderia ajudá-los a desenvolver uma melhor compreensão nos conteúdos de aprendizado” (NINCAREAN et al., 2013).

Mais uma vantagem importante é que a RA permite interagir com elementos que, por sua própria natureza ou por diferentes razões, são difíceis de inserir nas salas de aula, nos ambientes educativos, ou ainda algumas vezes até em qualquer ambiente de tipo geral. Pode-se pensar, por exemplo, em elementos de grandes escalas como animais gigantes ou extintos, planetas, etc., os quais se querem estudar, ou ainda em outros casos, elementos na escala microscópica invisíveis ao olho humano. Todos esses elementos poderiam ser situados numa sala de aula através da RA, da forma mais realista possível, e numa escala adequada para ser explorados de um jeito colaborativo, ainda com o acréscimo que fornece a RA em relação a outras tecnólogas similares como a RV, de não ter que substituir o ambiente real (no caso a sala de aula com o professor como ator principal da comunicação) para interagir com esses elementos virtuais.

Em relação a isso, Rodríguez Lomuscio (2011) afirma o seguinte:

> “Ao permitir interagir com distintos elementos, a RA permite que os estudantes sejam capazes de perceber e manipular objetos que de outra forma seria impossível. Por outro lado, não eliminando o contexto do mundo real, esta tecnologia permite que isto seja feito sem perder a comunicação e colaboração que podem ser necessários em distintos contextos educacionais”.

Outra das principais vantagens da RA é seu caráter experimental, ou seja, que permite a experimentação do usuário, sendo uma tecnologia que torna o aprendiz o ator principal na execução de uma atividade, onde ele mesmo pelos seus próprios meios executa, confere e obtém a suas conclusões, precisando de uma mínima ajuda do professor, o qual se torna mais em um guia na experiência do próprio aprendiz do que o ator principal da aprendizagem do aluno na atividade. “A RA é uma tecnologia ativa, não uma tecnologia passiva, os estudantes podem usá-la para a construção de novas formas de compreensão sobre a base das interações com os objetos virtuais que são subjacentes aos dados à vida real” (ABDULMUSLIH ALSIRHANI, 2012). Esta característica da RA é muito importante, mesmo considerando algumas tecnologias utilizadas atualmente na educação. Em relação a isso, tem-se:

> “[...], apresentações em PowerPoint, vídeos animados estão sendo usados generalizadamente nas salas de aula ao redor do mundo. O problema com estas tecnologias é que o estudante permanece como um elemento passivo do processo de aprendizado. Estas tecnologias da informação devem visar uma melhor e maior participação dos estudantes”. (SINGHAL et al., 2012).

Embora as características e vantagens de RA apresentadas até aqui sejam muito importantes, e certamente podem levar a fortalecer os processos de ensino-aprendizagem, algumas delas são também comuns em muitos outros tipos de tecnologias que podem ser utilizadas ou ainda que são utilizadas atualmente na educação (p.ex., a RV, vídeos etc.). Portanto, a seguir são apresentadas vantagens mais específicas da própria tecnologia de RA, quando aplicada na sala de aula ou em ambientes educativos. Para isso, toma-se como referência principal o trabalho apresentado por Billinghurst (2002).

Segundo Billinghurst (2002), a educação é uma área muito valiosa para aplicar RA pelas seguintes razões:

- Suporte de *Seamless Interaction* entre ambientes reais e virtuais;
- Uso de uma *Tangible Interface Metaphor* para manipulação de objetos;

- Permitir *Transition smoothly* entre realidade e virtualidade.

Explicam-se, resumidamente, os conceitos:

***Seamless Interaction*:** Nas salas de aula, os estudantes trabalham bem melhor se estão situados em um mesmo espaço de trabalho. É possível atingir isso usando RA, pois o ambiente, no qual vão interagir os estudantes, continua sendo o mesmo (por exemplo, a sala de aula), podendo ter uma interação direta tanto com outros estudantes quanto com elementos virtuais aumentados. Isso é considerado importante, pois, nesse caso, o espaço de tarefas pertence ao espaço de comunicação, os estudantes têm uma comunicação direta suportada no ambiente comum, onde podem se olhar e compartilhar sinais de comunicação enquanto fazem experimentos com os elementos virtuais. Já em outros casos, com outras tecnologias de computação, por exemplo fazendo uso de algum tipo de software educativo *Desktop*, mesmo estando em um mesmo computador, os estudantes focam-se no computador, e a atenção torna-se à máquina.

***Tangible Interface Metaphor*:** Pessoas sem nenhum conhecimento e interação com o mundo da computação podem ainda ter uma experiência rica de RA, pois sistemas de RA baseados em uma *Tangible Interface Metaphor* permitem manipular elementos virtuais através de objetos físicos de uma forma intuitiva (p.ex., através de cartões com imagens reconhecidas pelo sistema ou marcadores de RA), sendo facilmente manipuláveis para qualquer tipo de pessoa, desde crianças muito novas até pessoas sem nenhum conhecimento de computadores.

Transition smoothly: A RA permite fazer uma transição fluida dos usuários sobre o VC, experimentando uma transição entre ambientes reais e virtuais.

Como visto, a RA fornece novas experiências de ensino-aprendizagem ricas e com grandes benefícios tanto para o professor quanto para os estudantes, sendo que esta tecnologia, atualmente, encontra-se madura o suficiente, podendo ser, por isso, considerada uma opção viável para os docentes, e utilizada em dispositivos tão comuns como computadores e *smartphones* que quase qualquer pessoa possui na atualidade. Porém, ainda com as atuais facilidades, a aplicação da RA na educação

tem sido pouca até agora, e, portanto, espera-se que mais pesquisadores possam focar seu interesse nesta área para que possam ser aproveitadas suas inúmeras vantagens. No entanto, deve-se notar que para obter boas experiências, esta tecnologia deve ser adotada corretamente nos processos educativos, focando-se no aprendiz e nos objetivos de aprendizado que ele deve atingir ao utilizar o aplicativo. Além disso, é necessário superar desafios pedagógicos relacionados ao processo de inserir a tecnologia de RA no contexto educativo, como o citado a seguir:

> "Assim, pode haver uma lacuna entre os métodos de ensino-aprendizagem atualmente usados nas salas de aula e a natureza centrada nos estudantes e exploratória da aprendizagem promovida por sistemas de RA. Designers de ambientes de aprendizagem de RA precisam perceber a lacuna e fornecer um possível suporte pare ajudar a professores e alunos a superá-la". (WU et al., 2013).

Em geral, pode-se afirmar que, embora a RA forneça muitas vantagens, também supõe muitos desafios para poder ser aplicada de um jeito adequado na educação. Em relação a isso, também se tem o seguinte:

> "Apesar de uma grande quantidade de pesquisas nas duas últimas décadas, a adoção da RA na educação e formação ainda é bastante desafiante devido a problemas com sua integração com métodos de aprendizagem tradicionais, custos para o desenvolvimento e manutenção dos sistemas de RA, e resistência geral às novas tecnologias". (KANGDON, 2012).

2.8 Realidade Aumentada Móvel na Educação

A RAM mistura as tecnologias de RA e computação móvel. Além da RA por si só e suas vantagens na educação, a computação móvel é outra tecnologia que abre novas portas no contexto educativo, e ainda foi listada pelo Relatório Horizont 2010

(GARCÍA et al., 2010) como uma das tecnologias emergentes que em próximos anos teria um uso bastante difundido para o ensino, aprendizado e pesquisa criativa.

A combinação de RA com a computação móvel para criar o que se conhece como RAM, foi muito importante para que a RA fosse mais difundida e chegasse a ter maior influência no mundo real, além do entorno de pesquisas. Ainda na atualidade, a RA principalmente através de RAM tem o potencial para influenciar na vida cotidiana das pessoas (por exemplo, com aplicativos que funcionam como guias turísticos, para procurar locais comerciais como restaurantes, etc.). Resumindo, a computação móvel tem potencializado o alcance da RA, assim como as suas possibilidades, podendo ainda afirmar, com certeza, que uma das razões pelas quais a RA ultimamente tem atingido algumas áreas a mais, incluindo a educação, do que os anos anteriores, é graças ao fato de que pode ser utilizada em dispositivos móveis e aproveitar suas vantagens. “Na verdade, são esses dispositivos [dispositivos de mão tais como *smartphones* e *tablets*] os que são em grande parte responsáveis por trazer a RA para o mercado de consumo geral” (BARICEVIC et al., 2016). A combinação das duas tecnologias mencionadas para formar a RAM tem representado grandes oportunidades no cenário educativo, e sua importância já tem sido citada em algumas pesquisas, como no artigo de Fabregat Gesa (2012), onde se afirma o seguinte:

> “Combinando computação móvel com técnicas de RA, cria-se um grande potencial para fornecer experiências de aprendizagem contextuais e *in situ* valiosas e de exploração e descoberta fortuita da informação conectada no mundo real. Espera-se que a experiência de interação com estes conteúdos seja principalmente benéfica para aqueles estudantes que precisam de maior nível de exploração”.

A seguir apresentam-se algumas das principais vantagens de utilizar a RAM no contexto educativo.

Além dos inúmeros benefícios que a tecnologia da RA fornece por si só, com a RAM são acrescentadas vantagens da computação móvel, eles principalmente estão relacionados com a liberdade quanto ao espaço físico onde poderiam ser

utilizados os sistemas criados com tecnologia de RA quando são misturados com a tecnologia de computação móvel. Muitos autores têm opinado sobre isso: “Significativamente, ela [a RAM] fomenta a mobilidade dos usuários, aumenta os lugares físicos onde a aprendizagem pode ocorrer, serve como ponte entre esses lugares, e habilita conexões entre aprendizagem formal e informal” (FITZGERALD et al., 2013). Sobre isso, segundo Klopfer & Sheldon (2010), conforme citado por Wu et al. (2013), “a mobilidade fornecida por dispositivos de mão poderia alavancar a autenticidade de um ambiente de aprendizagem e aumentaria as interações dos aprendizes com os outros”. Nesse contexto, Billinghurst & Duenser (2012) afirmam que: “Com dispositivos móveis, os usuários podem ter uma experiência de AR em qualquer lugar, o que significa que os alunos podem permanecer ativamente envolvidos no processo de aprendizagem tanto fora como dentro da sala de aula”.

Outra das vantagens da RAM na educação tem sido o fato de que os dispositivos utilizados (principalmente *smartphones* e *tablets*) tornam-se ferramentas muito adequadas para aplicar a RA no contexto de ensino-aprendizagem devido a muitas razões como: sua simplicidade (a comparação com outros tipos de dispositivos de RA mais complexos), seu baixo custo, o alto nível de difusão na sociedade destes dispositivos, e ainda seus diversos tipos de sensores. Em relação a isso, afirma-se o seguinte:

> “[...] *smartphones* contêm um conjunto cada vez mais sofisticado de sensores, permitindo assim à AR se tornar mais pessoalmente significativa e situada, experiências que anteriormente requeriam o empréstimo de equipamentos especializados agora provavelmente são acessíveis aos estudantes ou ao público em geral, fornecendo assim uma tecnologia mais sustentável para a aprendizagem quotidiana”. (FITZGERALD et al., 2013).

Mais uma vantagem da RAM na educação é que graças à faixa de componentes adicionais, como todo tipo de sensores (p.ex., GPS, acelerômetro) integrados atualmente em *smartphones* e *tablets*, se abrem novas oportunidades de desenvolvimento para novos tipos de experiências de ensino-aprendizagem utilizando a RA, assim como para novos tipos de público alvo (p.ex., estudantes com algum tipo de deficiências, etc.). Estes componentes, presentes em dispositivos

móveis atuais, já têm sido usados no contexto de aprendizado móvel e ainda sem ser combinados com a tecnologia de RA tem mostrado um impacto positivo, assim, sua relevância poderia ser ainda maior na educação se fossem utilizados junto com a RA em contextos adequados. Quanto a sensores, tais como acelerômetros e seu impacto na educação, segundo Mehigan (2009), tem-se que: “a incorporação de tecnologias tais como o acelerômetro, que é padrão em muitos telefones celulares modernos, significa uma oportunidade para desenvolver novas aplicações de aprendizagem móvel inclusivas”. Por outro lado, em relação a outros componentes úteis para a educação e presentes nestes dispositivos, o mesmo autor, Mehigan (2009), se refere da seguinte forma: “A incorporação de tecnologias sem fio, como o *Bluetooth*, permitiria o fornecimento de um ambiente colaborativo para a aprendizagem móvel, permitindo que todos os alunos se comuniquem através de seu dispositivo de mão, independentemente das deficiências enfrentadas por alunos individuais”. Finalmente, considerando estes componentes e suas possibilidades, especificamente na área de RA, observa-se que:

> “Atuais *smartphones* e *tablets* combinam um rápido processador com hardware gráfico, uma grande tela sensível, e sensores *on-board* (câmera, GPS, bússola, acelerômetros), tornando eles ideais para experiências de RA tanto *Indoor* quanto *Outdoor*. Pesquisadores e desenvolvedores comerciais tem usado estas plataformas para criar aplicativos educativos, fornecendo experiências de aprendizagem inovadoras”. (BILLINGHURST & DUENSER, 2012).

Quanto ao nível de colaboração atingido com sistemas de RAM, notam-se vantagens ainda quando comparadas aos sistemas similares que utilizam tecnologia móvel, porém sem RA. Por exemplo, no artigo de Morrison et al. (2009) apresenta-se um estudo comparativo entre dois sistemas de mapas com ambas as tecnologias (um sistema com RAM e outro com tecnologia móvel unicamente). O aplicativo com RAM permitiu aumentar um mapa real através de conteúdo virtual (como fotos) graças à RA, por outro lado, o outro sistema consistiu em um mapa digital 2D. Ambos sistemas foram utilizados com grupos de usuários em atividades que consistiam na procura de informação baseada em localização sobre objetivos de tarefas. O estudo levantou evidências que, através do sistema com RAM, os

participantes foram motivados a trabalhar juntos, interagindo entre eles para a solução das tarefas, por outro lado, no caso do sistema sem RA, o comportamento dos participantes foi associado com estratégias de solução de tarefas que foram mais solitárias, notando-se um fomento à colaboração com a RAM.

Finalmente, outra vantagem da RAM na educação é ser divertido e motivador para os aprendizes, pois é fornecido todo tipo de experiências com a característica comum de serem de alto nível de interesse para eles, sendo experiências de aprendizagem parecidas com jogos, o qual é diferente de muitas ferramentas atualmente usadas na educação que, pelo contrário, oferecem pouco nível de motivação nos estudantes. “Seja que eles estejam *Outdoor* ou *Indoor*, os usuários de aplicações com RAM curtem se envolvendo em tópicos através de experiências semelhantes a jogos, e os desenvolvedores já tem criado algumas aplicações educativas interessantes, [...]” (BILLINGHURST & DUENSER, 2012).

Capítulo 3

TRABALHOS RELACIONADOS

Neste capítulo, é apresentado um conjunto de metodologias, *frameworks* e ferramentas existentes propostos para o desenvolvimento de aplicativos com RA, e que foram encontrados na pesquisa bibliográfica. Nota-se que os projetos foram encontrados em bancos de dados de artigos científicos tais como: IEEE Xplore Digital Library[5], SciELO[6], assim como em algumas bibliotecas digitais universitárias, usando as palavras-chave: Metodologias de Realidade Aumentada*, Frameworks* de Realidade Aumentada, e ferramentas de Realidade Aumentada.

Os projetos encontrados são apresentados em três categorias: Frameworks de RA Orientados a Desenvolvedores; Ferramentas Para Criação de Aplicações de RA Sem Precisar de Programação; e, finalmente, Metodologias Para Criação de Aplicações de RA, Sem Ferramentas Para Apoiar o Desenvolvimento, apresentadas nas Seções 3.1, 3.2 e 3.3 respetivamente. Para cada um dos projetos, inicialmente foi feita uma análise das suas características, para finalizar com uma comparação em relação ao *framework* e à metodologia propostos no presente trabalho. Também é apresentada uma tabela comparativa visando esclarecer ainda mais as similaridades e diferenças dos projetos existentes com o proposto neste trabalho. No

[5] IEEE Xplore Digital Library é um banco de dados com conteúdo científico e técnico publicado pelo IEEE. Para mais informações ver em:
https://ieeexplore.ieee.org/Xplore/home.jsp

[6] SciELO é um banco de dados bibliográfico e biblioteca digital de periódicos científicos de acesso aberto. Para mais informações ver em:
http://www.scielo.org/php/index.php?lang=en

final do capítulo, será apresentada uma discussão geral dos projetos estudados e o projeto da presente dissertação, assim como uma tabela comparativa geral incluindo todos os projetos juntos.

3.1 Frameworks de RA Orientados a Desenvolvedores

AR Learning Videogame For Kids With ADHD Symptoms (TOBAR et al., 2013):

Nesta pesquisa foi apresentado um *framework* chamado *AR Interaction Framework*. O *AR Interaction Framework* permite realizar interações usando marcadores de realidade aumentada, voltadas para processos lógico matemáticos em crianças. São suportados diferentes tipos de interação, as quais consistem em:

- Através do marcador, arrastar e soltar objetos virtuais;
- Através do marcador, colocar objetos;
- Através do marcador, manipular a *transform*[7] de objetos (rotacionar e movimentar).

O *AR Interaction Framework* foi criado visando apoiar tanto necessidades específicas de interação com RA para um jogo educativo (chamado *Gremlings in my Mirror*), quanto para futuros projetos, pois essas interações também correspondem a tipos de interação possíveis em jogos classificados pelo autor como *Spatial AR-Game*[8].

[7] *Transform* se refere a uma matriz 4x4 que armazena características tais como posição e rotação de um objeto. Para mais informações ver em: https://en.wikipedia.org/wiki/Transformation_matrix

[8] *Spatial AR-Game* é um tipo de jogo de RA, no qual o usuário está sempre na frente da câmera e do monitor. Para mais informações ver em: https://www.researchgate.net/publication/272997565_AR_Learning_Videogame_For_Kids_With_ADHD_Symptoms

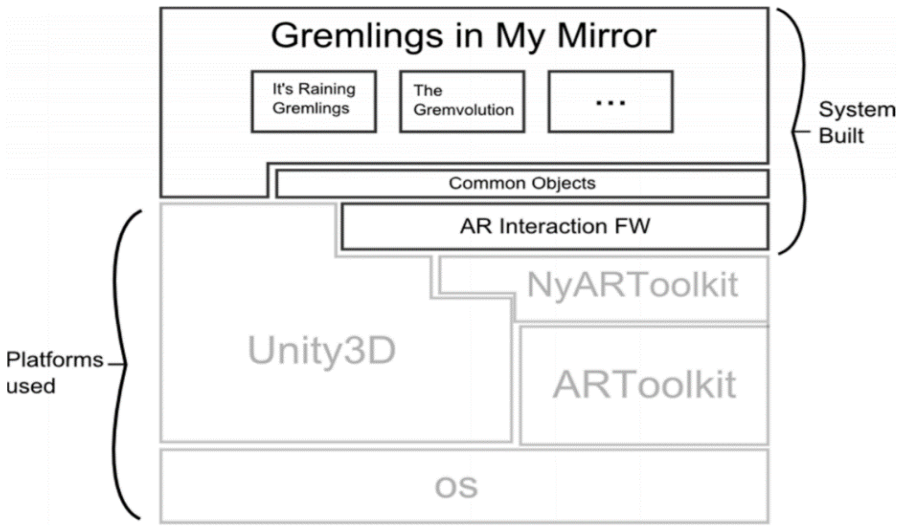

Figura 3.1: Arquitetura do sistema *Gremlings in my Mirror* criado com o *AR Interaction Framework*. (TOBAR et al., 2013)

Quanto à comparação do atual projeto com o nosso, pode-se concluir que estão relacionados, uma vez que em ambos são propostos *frameworks* que fornecem a desenvolvedores um suporte de funcionalidades úteis para apoiar o desenvolvimento de aplicações de RA. O *framework* proposto por esse trabalho, semelhante ao nosso, pode ser considerado como voltado para a criação de aplicações especificamente de tipo educativo, pois suas funcionalidades visam apoiar interações de RA úteis para o jogo educativo desenvolvido nesse projeto, assim como para *Spatial AR-Game*, podendo ser de utilidade para criar outros projetos educativos em contextos similares. Por outro lado, pode-se dizer que as funcionalidades fornecidas por esse *framework* não são de cunho geral na educação, desde que estão limitadas a processos educativos lógico-matemáticos, como, por exemplo, emparelhamento e ordenamento de elementos (para o que foi útil no jogo educativo em questão). Dessa forma, esse *framework* não teria a mesma utilidade em outras áreas da educação, diferente do que acontece com o *framework* proposto por esse trabalho, que, pelo contrário, fornece funcionalidades que, além de ser voltadas para a criação de aplicativos educativos, também são de cunho geral na educação, e, portanto, visa apoiar o desenvolvimento de aplicativos educativos com RA sem estar focado em nenhuma área particular da educação.

Finalmente, o atual *framework* difere também do nosso, uma vez que está orientado para aplicativos de RA baseada em marcadores e *Desktop*, e não para aplicativos de RAM baseada em marcadores, como em nosso caso.

A tabela comparativa a seguir, irá usar os seguintes elementos de comparação, representados através de números na tabela:

1. Fornece metodologia projetada para cenários educativos.
2. Fornece ferramenta ou *framework* para apoiar a criação do aplicativo.
3. Ferramenta ou *framework* está voltada para a criação de aplicativos especificamente de tipo educativo (fornece funcionalidades específicas para educação).
4. As funcionalidades fornecidas são de cunho geral na educação.
5. Público alvo ou usuário final.
6. Tipo de RA suportado e plataforma.

Tabela 3.1: Tabela comparativa entre projeto (TOBAR et al., 2013) e o nosso projeto

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(TOBAR et al., 2013)	Não	Sim (*AR Interaction Framework)*	Sim	Não (Voltadas para processos logico matemáticos)	Desenvolvedores	RA baseada em marcadores e *Desktop (Desktop AR)*
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

Deve-se notar que será utilizado o mesmo critério (com os mesmos elementos de comparação representados por números) para as tabelas comparativas restantes.

Uma Metodologia para o Desenvolvimento de Aplicações de Realidade Aumentada em Telefones Celulares Utilizando Dispositivos de Sensoriamento (ZUÑIGA TORRES, 2008)**:**

Neste projeto, propõe-se uma metodologia para o desenvolvimento de aplicações de tipo *Hand-held AR* (ou RAM) com marcadores, e, principalmente, para aquelas aplicações que utilizam sensores com interfaces de comunicação sem fio. A metodologia proposta consiste em: orientações (ou *guidelines*) para a criação de sistemas *Hand-held AR*, modelagem conceitual e semântica dos dados e do usuário e bibliotecas de software encarregadas de processar e apresentar informação gerada pelo aplicativo como objetos virtuais, isso é feito na plataforma *Java 2 Micro Edition* (J2ME) (ORACLE, 2017). As orientações (ou *guidelines*) propostas na metodologia foram as seguintes: Definição da Aplicação, Definição da Aquisição de Dados, Definição de Modelagem Semântica, Definição do Dispositivo Móvel e Disponibilização das informações. Por outro lado, as classes criadas foram encarregadas especificamente de: aquisição e processamento dos dados gerados pelos dispositivos sensores e pala câmara, reconhecimento de marcadores, renderização de imagens virtuais, efeitos sonoros, efeitos de vibração e modelagem das preferências do usuário.

Para a criação das bibliotecas, foram utilizadas outras bibliotecas existentes que serviram de base, como foi o caso da biblioteca *Mobile Media API* (MMAPI) que proporciona funcionalidades de multimídia e foi útil para a captura de vídeo implementada na classe de nome *CameraPAD,* e também para a renderização de efeitos sonoros na classe *SomPAD*. Por outro lado, a classe *SemacodePAD*, encarregada do reconhecimento de marcadores, foi criada utilizando a biblioteca *Semacode Reader* do projeto *Open Source* chamado *Semacode* (SEMACODE, n.d.). As classes restantes foram implementadas de um jeito similar e são apresentadas na Figura 3.3. A Figura 3.2 apresenta a arquitetura da metodologia proposta, e a Figura 3.3 as classes criadas com as respetivas APIs (*Application Programming Interface*), sobre as quais foram implementadas.

Figura 3.2: Arquitetura proposta para aplicações de tipo *Hand-held AR* que usam sensores com interfaces de comunicação sem fio. (ZUÑIGA TORRES, 2008)

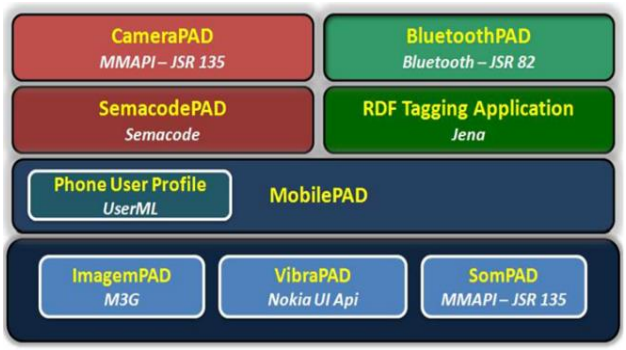

Figura 3.3: Arquitetura baseada na plataforma J2ME que resume as classes implementadas. (ZUÑIGA TORRES, 2008)

Comparando o nosso projeto com o atual, pode-se concluir que estão relacionados, uma vez que em ambos são propostas metodologias projetadas para o desenvolvimento de aplicações de RAM baseada em marcadores, porém a atual metodologia está voltada para aplicativos que usam sensores com interfaces de comunicação sem fio, e não para aplicativos educativos como no nosso caso.

O atual projeto semelhante ao nosso também fornece suporte de tipo software (no caso, bibliotecas de software criadas) para apoiar o desenvolvimento dos aplicativos, no entanto esse suporte difere muito do nosso principalmente porque seu foco não é aplicativos educativos. Inicialmente, grande parte desse suporte consiste em funcionalidades relacionadas com a utilização de sensores, como a aquisição e processamento dos dados gerados pelos sensores. Por outro lado, o suporte relacionado com RA propriamente dito, corresponde principalmente às funcionalidades básicas necessárias para todo sistema de RA baseada em marcadores, como a captura de vídeo, reconhecimento de marcadores, etc., os quais foram feitos sobre APIs existentes isoladas. Estas funcionalidades, na

atualidade, podem ser suportadas ao mesmo tempo e de um jeito simples por muitas ferramentas ou *framework*s de RA. Nesse sentido, o nosso projeto consiste no uso de ferramentas, tais como as discutidas, disponíveis na atualidade (que já integram as funcionalidades básicas para todo sistema de RA baseada em marcadores) e sobre elas é desenvolvido um *framework* que aproveita essas funcionalidades já existentes e fornece novas funcionalidades para apoiar processos educativos de cunho geral.

Finalmente, o atual projeto também tem como público alvo desenvolvedores com conhecimento de programação, e, portanto, nesse aspecto, se relaciona com o nosso projeto.

Tabela 3.2: Tabela comparativa entre projeto (ZUÑIGA TORRES, 2008) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(ZUÑIGA TORRES, 2008)	Não (Embora é fornecida uma metodologia, ela não está projetada para aplicativos educativos)	Sim	Não	Não	Desenvolvedores	RAM baseada em marcadores e dispositivos sensores com interfaces de comunicação sem fio
Nosso Projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

3.2 Ferramentas Para Criação de Aplicações de RA Sem Precisar de Programação

From Reality to Augmented Reality: Rapid Strategies for Developing Marker-Based AR Content Using Image Capturing and Authoring Tools (CAMBA & CONTERO, 2015):

Descreve-se uma metodologia para a criação de conteúdo de RA baseado em marcadores. Nesta metodologia se propõe o uso de modelagem 3D baseado em imagens através de ferramentas existentes, permitindo a criação de modelos 3D a partir de fotografias e sem precisar de conhecimento técnico em software de modelagem. Além disso, se faz uso de uma ferramenta de autoria desenvolvida pelos autores que permite criar os cenários de RA de um jeito simples, a partir dos modelos já criados. Nota-se que na criação de conteúdo de RA normalmente é necessário tanto conhecimento de modelagem tridimensional (para criar os modelos 3D dos sistemas de RA), quanto habilidades de programação, o que torna-se um desafio na hora de os instrutores tentarem criar suas próprias experiências de RA. Assim, esta proposta representa uma oportunidade para que os professores possam criar seus próprios conteúdos de RA de um jeito simples e sem depender de habilidades comumente necessárias para este tipo de software.

A metodologia apresentada consiste em duas fases: *Creation of 3D Models* e *Creation of AR Content*. Na fase inicial se propõe o uso de alguma ferramenta de modelagem baseada em imagens para realizar o processo de modelagem dos elementos 3D necessários para a cena de RA. Uma vez executada a fase inicial, na segunda fase se utiliza um aplicativo de autoria de RA chamado *Aumentaty Author*, desenvolvido pelos autores, que faz a conversão do elemento 3D, criado na fase anterior, a um elemento interativo de RA (vinculando esse modelo a um marcador). A metodologia com suas duas fases e as tarefas em cada uma delas é apresentada na Figura 3.4.

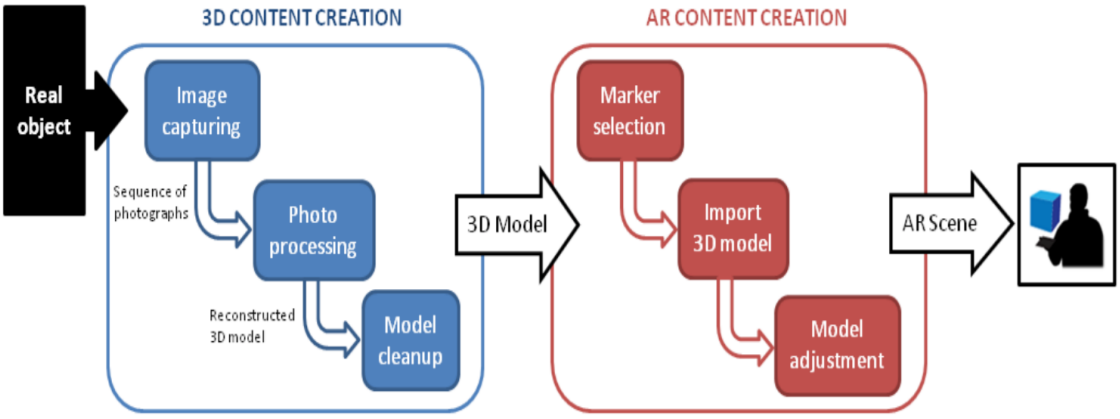

Figura 3.4: Metodologia proposta para criação de conteúdos de RA. (CAMBA & CONTERO, 2015)

Comparando-o ao nosso projeto, pode-se concluir que estão relacionados, pois inicialmente em ambos projetos são propostas metodologias projetadas para o desenvolvimento de aplicativos educativos com RA baseada em marcadores, tendo como diferença que a nossa metodologia é especificamente para dispositivos móveis, enquanto que a discutida nesse trabalho é tanto para dispositivos móveis quanto para *Desktop*.

Por outro lado, este trabalho também fornece uma ferramenta para apoiar a criação do aplicativo (no caso, um software que não requer de programação), que é chamada *Aumentaty Author*. No entanto, pode-se dizer que a ferramenta *Aumentaty Author* (ao contrário do nosso *framework*) não está voltada para a criação de aplicativos especificamente do tipo educativo, pois não fornece funcionalidades específicas para a educação (p.ex., para suportar realização de perguntas, etc.). *Aumentaty Author* foca em automatizar o máximo possível o processo de criação de sistemas de RA, sendo fornecida uma abordagem simples e rápida para criar aplicativos de RA. No entanto, os sistemas que podem ser criados são cenários de RA simples e com poucas características educativas (p.ex., conteúdo avaliativo), assim, pode-se afirmar que *Aumentaty Author* não leva em consideração necessidades educativas que surgem na criação de aplicativos com RA, além de permitir que professores possam criar aplicativos simples com esta tecnologia.

Uma diferença importante entre os dois projetos, é que este tem como público alvo professores sem habilidades em programação. Pelo contrário, nosso projeto tem como público alvo desenvolvedores, assim, levando em consideração o usuário final, eles se orientaram a questões diferentes. Inicialmente, neste projeto, através da sua metodologia e sua ferramenta *Aumentaty Author,* visa-se permitir que professores possam criar conteúdo de RA de um jeito simples e rápido (ainda sem as habilidades comumente necessárias para este tipo de software), no entanto, como falado, *Aumentaty Author* não está voltado para a criação de aplicativos especificamente do tipo educativo, e, além disso, os tipos de aplicativos que podem ser criados estão limitados pela ferramenta, o que significa uma perda de flexibilidade quanto ao desenvolvimento dos aplicativos. Pelo contrário, o nosso projeto através do *framework* proposto, fornece funcionalidades específicas para educação e de cunho geral (ou seja, está voltado para a criação de aplicativos especificamente do tipo educativo), e uma vez que tem como foco desenvolvedores, permite manter uma flexibilidade maior para a criação de aplicativos educativos com RA, do que ferramentas tais como *Aumentaty Author*.

Tabela 3.3: Tabela comparativa entre projeto (CAMBA & CONTERO, 2015) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(CAMBA & CONTERO, 2015)	Sim	Sim (Software *Aumentaty Author*)	Não	Não	Professores	*Desktop* AR, e RAM baseada em marcadores
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

A Learning Environment for Augmented Reality Mobile Learning (CUBILLO et al., 2014)**:**

Apresenta-se uma ferramenta de autoria, fácil de usar, chamada ARLE (*Augmented Reality Learning Environment*), que permite a professores criar seus

próprios cenários de RA baseada em marcadores, contendo objetos, descrições sobres esses objetos e perguntas. As características ou funcionalidades de ARLE podem ser apresentadas, resumidamente, da seguinte maneira:

- Permite ao professor integrar diferentes tipos de recursos multimídia (vídeos, áudios, imagens, e objetos 3D) de um jeito simples, para criar seu cenário de RA.
- Permite ao professor inserir descrições sobre elementos que são exibidos, o que corresponde a informação adicional (textos, imagens, ou vídeos).
- Permite ao professor inserir *Multiple Choice Questions* (MCQ) sobre os elementos exibidos.
- Inclui uma biblioteca que armazena os conteúdos virtuais de RA criados. Esta biblioteca permite que professores possam, de um jeito simples, compartilhar seus recursos e reutilizar os conteúdos de RA previamente criados por outros.

Em geral, ARLE está dividido em duas partes baseando-se na arquitetura Cliente-Servidor (ver Figura 3.5). Os componentes de ARLE são explicados, resumidamente, a seguir:

- *ARLE Client*: contém a aplicação móvel *ARLE Mobile Client*, pela qual o estudante pode interagir com os recursos de RA com operações tais como: modificar o objeto (rotacionar, movimentar, escalar etc.), executar vídeos, e responder perguntas.
- *ARLE Server*: contém a ferramenta de autoria *ARLE Web Authoring Tool*, na qual o professor pode inserir recursos digitais e criar os cenários de RA levando em consideração as características definidas anteriormente.

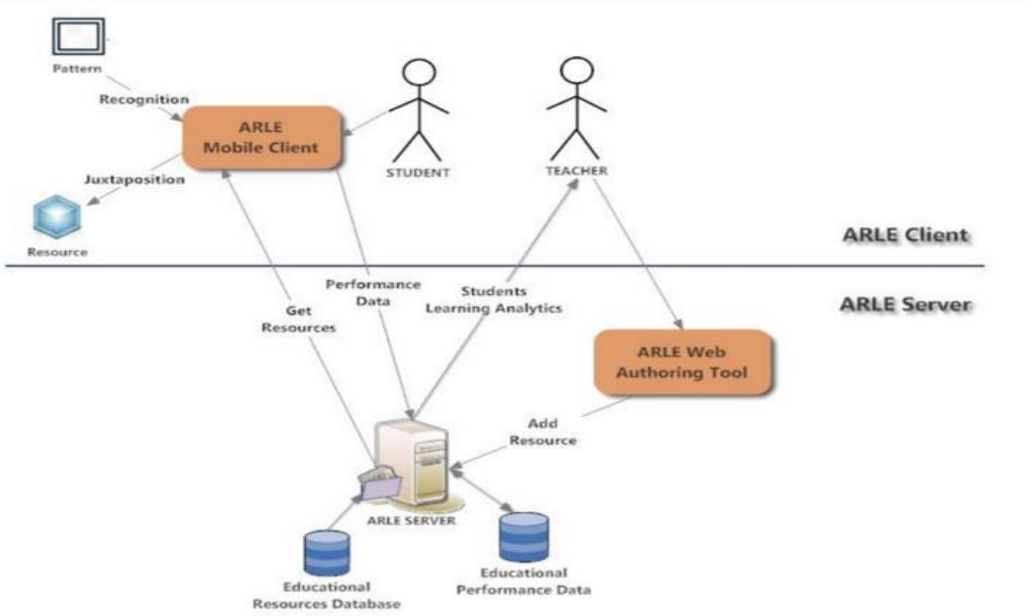

Figura 3.5: Arquitetura que apresenta componentes e funcionamentos do sistema ARLE. (CUBILLO et al., 2014)

Quanto à comparação do atual projeto com o nosso, pode-se concluir que estão relacionados, uma vez que em ambos são apresentadas ferramentas para apoiar a criação de aplicativos educativos com tecnologia de RAM baseada em marcadores. Nesse projeto é fornecido o software ARLE, enquanto que no nosso projeto é proposto um *framework*. Ambas as ferramentas fornecem funcionalidades específicas para a educação e de cunho geral, ou seja, estão voltadas para a criação de aplicativos educativos de cunho geral. Em relação às funcionalidades fornecidas, existem grandes diferenças entre ambas as ferramentas. No nosso *framework*, embora também tenha um suporte para definição de questionários com perguntas e respostas, este suporte é muito mais abrangente, permitindo diferentes tipos de perguntas (perguntas com alternativas de respostas através de textos, imagens ou ainda modelos 3D em RA) e, por outro lado, o Software ARLE suporta unicamente perguntas com alternativas através de textos, ficando assim muito mais limitado nesse aspecto. Por fim, o nosso *framework*, embora também permita a incorporação de informação adicional, essa informação adicional será de tipo 3D em RA (textos 3D ou imagens), diferentemente da informação adicional no ARLE que será sempre em 2D. Finalmente, o nosso *framework* fornece várias funcionalidades através do uso de marcadores de controle (o qual não foi considerado no ARLE), e,

em geral, fornece um pacote de funcionalidades com a característica de serem altamente customizáveis num nível maior do que ARLE.

Uma diferença importante entre as duas ferramentas fornecidas é que o Software *ARLE* tem como público alvo professores sem habilidades em programação, enquanto o nosso *framework* tem como público alvo desenvolvedores. Assim, uma vez que *ARLE* é um software que não precisa de habilidades de programação, os tipos de aplicativos que podem ser criados estão mais limitados, tendo uma perda de flexibilidade no desenvolvimento dos aplicativos. Dessa forma, como o *framework* proposto tem como foco desenvolvedores (usa-se programação), pode-se manter maior flexibilidade para a criação dos aplicativos.

Tabela 3.4: Tabela comparativa entre projeto (CUBILLO et al., 2014) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(CUBILLO et al., 2014)	Não	Sim (Software *ARLE*)	Sim	Sim	Professores	RAM baseada em marcadores.
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

3.3 Metodologias Para Criação de Aplicações de RA, Sem Ferramentas Para Apoiar o Desenvolvimento

Propuesta Metodológica para la Construcción de Objetos Virtuales de Aprendizaje basados en Realidad Aumentada (TOVAR et al., 2014)**:**

Propõe-se uma metodologia mista para o desenvolvimento de objetos virtuais de aprendizagem (OVAs) com RAM baseada em marcadores, a qual foi formada a partir da metodologia para o desenvolvimento de OVAs chamada AODDEI (análise, obtenção, desenho, desenvolvimento, avaliação, implementação) (MUÑOZ et al., 2006), e a metodologia de desenvolvimento de software ESBC (engenharia de software baseada em componentes) (PRESSMAN, 2011). A metodologia AODDEI

foi usada como base na estrutura da metodologia proposta, e a ESBC permitiu complementar a AODDEI e fazer reuso de componentes já criados, o que permite diminuir o tempo de desenvolvimento. A metodologia proposta surgiu devido à necessidade de uma metodologia para criar OVAs suportando o uso de tecnologias emergentes como a RA em dispositivos móveis, pois as metodologias existentes para OVAs foram criadas para o desenvolvimento de conteúdo do tipo páginas web e software de escritório, tendo assim um foco diferente e que leva a inconsistências. Esta metodologia consistiu unicamente na definição de uma série de fases que foram as seguintes: Análise do Negócio, desenho e escolha das ferramentas, construção e adaptação das ferramentas de engenharia e testes e implantação.

Comparando-o ao nosso projeto, pode-se concluir que estão relacionados, uma vez que em ambos são propostas metodologias projetadas para o desenvolvimento de aplicações educativas de RAM baseada em marcadores, no entanto, na atual metodologia além das fases (ou orientações teóricas) definidas para o processo de criação das OVAs com RAM, não se fornece nenhuma ferramenta ou *framework* de apoio quer permita facilitar codificação ou em geral apoiar o desenvolvimento do aplicativo. Isto é diferente da nossa metodologia, em que é fornecido um *framework* com esse objetivo. Finalmente, esta metodologia também se relaciona com a nossa, uma vez que está orientada a desenvolvedores de software com conhecimento de programação, que, no caso, estarão encarregados de seguir cada uma das fases definidas pela metodologia até conseguir criar suas OVAs com tecnologia de RAM baseada em marcadores.

Tabela 3.5:Tabela comparativa entre projeto (TOVAR et al., 2014) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(TOVAR et al., 2014)	Sim (Metodologia para OVAs)	Não	Não	Não	Desenvolvedores	RAM baseada em marcadores.
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

Construção de Ambientes Educacionais com Realidade Aumentada: Processo Centrado no Usuário (NAKAMOTO et al., 2010)**:**

Define-se uma metodologia de análise de requisitos para a construção de objetos educacionais com RA, centrada no usuário e na usabilidade do aplicativo, isto visando que os projetistas deste tipo de sistemas possam fazê-los mais usáveis e diminuir sua carga cognitiva, levando em consideração a complexidade das suas interfaces (se comparado a interfaces de software 2D com as quais os usuários geralmente já estão habituados) e o fato de que as orientações disponíveis para construção de software estão voltadas somente para ambientes 2D tradicionais. Na Figura 3.6 é apresentado o modelo de análise de requisitos proposto com todas as suas fases, as quais são: ambientes genéricos de RA, experiências com usuário, análise do perfil do usuário, análise do contexto da tarefa, análise das capacidades e restrições da plataforma, usabilidade, e guia do projeto.

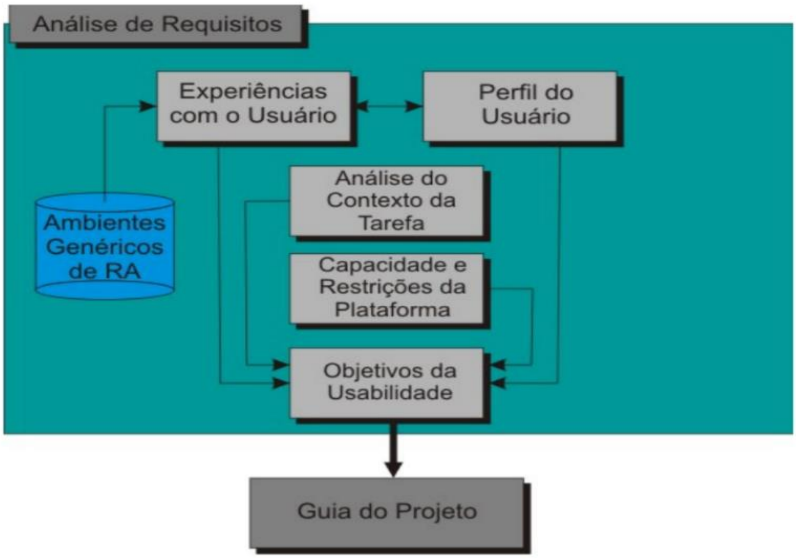

Figura 3.6: Análise de requisitos para ambientes de RA. (NAKAMOTO et al., 2010)

Comparando o nosso projeto com o atual, pode-se concluir que estão relacionados, uma vez que em ambos são propostas metodologias projetadas para o desenvolvimento de ambientes educacionais de RA. No entanto, a atual metodologia difere da nossa em vários aspectos como: está centrada no usuário e na usabilidade, portanto tem um foco educativo bem mais específico do que a nossa, e, além disso, uma das diferenças mais importantes é que esta metodologia é unicamente de análise de requisitos e não atinge outras fases no processo de criação do software de RA. Em outras palavras, a metodologia não fornece um guia completo que auxilie os projetistas no desenvolvimento de softwares educativos com RA, e, portanto, também é claro que não fornece nenhuma ferramenta ou *framework*

de apoio no desenvolvimento, ao contrário do que acontece com a nossa metodologia, a qual apoia-se em um *framework* proposto.

Por outro lado, pode-se dizer que a atual metodologia semelhante à nossa, tem como público alvo desenvolvedores de software, no entanto, uma vez que a atual metodologia atinge unicamente o processo de análise de requisitos, esse perfil pode ser definido mais especificamente para analistas de requisitos. Finalmente, a atual metodologia difere também da nossa, uma vez que está orientada para aplicativos educativos de RA em geral, e não especificamente para aplicativos educativos de RAM baseada em marcadores, como em nosso caso.

Tabela 3.6: Tabela comparativa entre projeto (NAKAMOTO et al., 2010) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(NAKAMOTO et al., 2010)	Sim	Não	Não	Não	Desenvolvedores (Analistas de requisitos)	RA
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

Análisis de sistemas de realidad aumentada y metodología para el desarrollo de aplicaciones educativas (ABDULMUSLIH ALSIRHANI, 2012)***:***

É definida uma metodologia para o desenvolvimento de aplicativos educativos utilizando RA. Esta metodologia foi orientada na docência, assim o papel do professor é ativo, fazendo parte da equipe que criará o conteúdo de RA (em tarefas como o assessoramento didático), visando permitir a correta definição dos objetivos a nível educativo a atingir fazendo uso do aplicativo. Esta metodologia consistiu na definição de uma série de fases que foram as seguintes: seleção do conteúdo, análise do conteúdo, e desenvolvimento.

Comparando o nosso projeto com o atual, pode-se concluir que estão relacionados, uma vez que em ambos são propostas metodologias projetadas para o desenvolvimento de aplicativos educativos de RA. No entanto, a metodologia atual

está voltada para aplicativos com tecnologia de RA em geral, e a nossa para aplicativos especificamente de RAM baseada em marcadores. Por outro lado, a atual metodologia, além das fases teóricas que define para o processo de criação dos aplicativos, não fornece nenhuma ferramenta ou *framework* para facilitar codificação ou, em geral, para apoiar o desenvolvimento do aplicativo. Isto é diferente do nosso caso, em que a metodologia apoia-se em um *framework* proposto. Finalmente, esta metodologia também se relaciona com a nossa, uma vez que está orientada a desenvolvedores de software com conhecimentos de programação, os quais estarão encarregados de seguir cada uma das fases definidas pela metodologia para criar o aplicativo educativo de RA.

Tabela 3.7: Tabela comparativa entre projeto (ABDULMUSLIH ALSIRHANI, 2012) e o nosso projeto.

Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(ABDULMUSLIH ALSIRHANI, 2012)	Sim	Não	Não	Não	Desenvolvedores	RA
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

3.4 Considerações Finais

Foi apresentada uma discussão geral sobre os projetos existentes encontrados e o nosso, visando dar uma visão mais geral das características dos projetos existentes, assim como os aspectos que motivaram à realização do presente trabalho.

Através da pesquisa bibliográfica e dos projetos encontrados, parece-nos que há muito poucos *frameworks*, e ainda muito poucas ferramentas em geral, voltados para a criação de aplicativos de RA especificamente de tipo educativo, pois, como pôde ser visto, a maioria de ferramentas de RA existentes que foram encontradas

estão voltadas para a criação de aplicativos de RA genéricos. A análise de várias destas ferramentas é apresentada a seguir.

Quanto a *frameworks* de RA existentes, pôde-se concluir que a maioria deles não estão relacionados com a educação, inicialmente considerando os principais *frameworks* de RA de grande difusão na comunidade (p.ex., ARToolKit, Vuforia etc.) que foram estudados na seção 2.2.2. Deve-se notar que são ferramentas de RA genéricas, assim suportam funcionalidades básicas necessárias para software de RA (captura de vídeo, reconhecimento de marcadores etc.), e outras funcionalidades de tipo genéricas. Estas ferramentas fornecem um suporte necessário para os sistemas de RA, mas que não atingem necessidades educativas concretas, portanto com este tipo de ferramentas os desenvolvedores dos aplicativos, uma vez que querem encarar o processo de criação de aplicativos educacionais, são obrigados a codificar completamente suas próprias funcionalidades voltadas às necessidades educativas.

Nos trabalhos relacionados, foram encontradas também ferramentas similares às anteriores, tais como as bibliotecas de software criadas na metodologia proposta por Zuñiga Torres (2008) na seção 3.1, as quais estão voltadas para o desenvolvimento de aplicativos de RA genéricos suportando o uso de dispositivos sensores com interfaces de comunicação sem fio. Com essas bibliotecas de software, foram fornecidas funcionalidades básicas necessárias para todo sistema de RA baseado em marcadores, como: a captura de vídeo, reconhecimento de marcadores etc. e funcionalidades adicionais relacionadas com a utilização de sensores.

Por outro lado, o *framework* encontrado nos projetos relacionados chamado *AR Interaction Framework* proposto em Tobar et al. (2013) na seção 3.1, que pode ser considerado como voltado para aplicativos de tipo educativo, tem como desvantagem que não é de cunho geral na educação, e está focado em apoiar unicamente processos ou necessidades educativas na área de Matemáticas, (p.ex., apoiar processos de emparelhamento e ordenamento de elementos), portanto não tem a mesma utilidade em outras áreas da educação, e não consiste em uma

solução completa que permita apoiar a criação de software de RA educativo em geral.

Quanto às ferramentas para criação de RA que não precisam de programação também existem muito poucas relacionadas com a educação, visto que nos trabalhos relacionados foram achadas duas dessas ferramentas, chamadas: *Aumentaty Author* proposta em Camba & Contero (2015), e *ARLE* proposta em Cubillo et al. (2014), ambas apresentadas na seção 3.2. Sobre a primeira delas, *Aumentaty Author*, ainda que relacionada com a educação, pôde-se concluir que não está voltada para a criação de aplicativos com RA especificamente de tipo educativos, tendo como objetivo principal facilitar e automatizar os processos necessários para criar aplicações simples de RA baseada em marcadores (para que professores criem os aplicativos), no entanto não fornece funcionalidades que permitam apoiar necessidades educativas (p.ex., a realização de perguntas etc.). Ao contrário, a segunda ferramenta encontrada, chamada *ARLE,* igual ao nosso *framework* proposto, está voltada para a criação de aplicativos com RA de tipo educativos de cunho geral (ou seja, para qualquer área da educação). Neste ponto, deve-se notar que, embora este tipo de ferramentas forneça um jeito simples de criar aplicativos de RA, os tipos de aplicativos que podem ser criados ficam limitados pela ferramenta (diferente do que acontece com *frameworks* cujo foco são desenvolvedores), o que significa uma perda de flexibilidade significativa quanto ao desenvolvimento dos aplicativos.

Através da pesquisa bibliográfica e projetos encontrados, também nos parece que há muito poucas metodologias para a criação de aplicativos educativos de RA apoiando-se no uso de ferramentas ou *frameworks*. Como pôde ser visto, a maioria das metodologias encontradas somente consistem em orientações teóricas, tais como: a metodologia mista para o desenvolvimento de OVAs com RAM baseada em marcadores, proposta em Tovar et al. (2014), a metodologia de análise de requisitos para a construção de objetos educacionais com RA centrada no usuário e na usabilidade do aplicativo, proposta em Nakamoto et al. (2010) e a metodologia para o desenvolvimento de aplicativos educativos utilizando RA e orientada na docência, proposta em Abdulmuslih Alsirhani (2012), todas apresentadas na seção 3.3. Neste sentido, é necessário que as metodologias criadas tenham como apoio *frameworks*

ou ferramentas que visem apoiar a criação dos sistemas, para que, além de guiar os desenvolvedores com roteiros teóricos (ou *guidelines*), os desenvolvedores também possuam ferramentas úteis no processo de codificação ou desenvolvimento do software.

Assim, após feita a análise dos projetos relacionados encontrados na pesquisa bibliográfica, e uma vez que não foi encontrado um *framework* de RAM voltado para a criação de aplicativos do tipo educativo de cunho geral tendo como público alvo desenvolvedores e que seja usado de apoio a uma metodologia, surge a necessidade da presente pesquisa.

Finalmente, uma tabela comparativa geral incluindo todos os projetos juntos é apresentada a seguir:

Tabela 3.8: Tabela comparativa geral de todos os projetos e o nosso.

Frameworks de RA Orientados a Desenvolvedores						
Projeto	**1**	**2**	**3**	**4**	**5**	**6**
(TOBAR et al., 2013)	Não	Sim (*AR Interaction Framework)*	Sim	Não (Voltadas para processos logico matemáticos)	Desenvolvedores	RA baseada em marcadores e *Desktop (Desktop AR)*
(ZUÑIGA TORRES, 2008)	Não (Embora é fornecida uma metodologia, ela não está projetada para aplicativos educativos)	Sim	Não	Não	Desenvolvedores	RAM baseada em marcadores e dispositivos sensores com interfaces de comunicação sem fio
Ferramentas Para Criação de Aplicações de RA Sem Precisar de Programação						
(CAMBA & CONTERO, 2015)	Sim	Sim (Software *Aumentaty Author*)	Não	Não	Professores	*Desktop* AR, e RAM baseada em marcadores

(CUBILLO et al., 2014)	Não	Sim (Software *ARLE*)	Sim	Sim	Professores	RAM baseada em marcadores.
Metodologias Para Criação de Aplicações de RA, Sem Ferramentas Para Apoiar o Desenvolvimento						
(TOVAR et al., 2014)	Sim (Metodologia para OVAs)	Não	Não	Não	Desenvolvedores	RAM baseada em marcadores.
(NAKAMOTO et al., 2010)	Sim	Não	Não	Não	Desenvolvedores (Analistas de requisitos)	RA
(ABDULMUSLIH ALSIRHANI, 2012)	Sim	Não	Não	Não	Desenvolvedores	RA
Nosso projeto	Sim (Metodologia proposta)	Sim (*Framework* proposto)	Sim	Sim	Desenvolvedores	RAM baseada em marcadores

Capítulo 4

FRAMEWORK E METODOLOGIA PROPOSTOS

No presente trabalho, visando solucionar os problemas já comentados, é apresentada como proposta a criação de um *framework* nomeado “AR Educational framework” que será usado de apoio a uma metodologia proposta para o desenvolvimento de aplicativos educativos com a tecnologia de RAM baseada em marcadores.

4.1 AR Educational Framework

O *framework* proposto corresponde a classes reusáveis (e ainda objetos pré-fabricados de Unity chamados *prefabs*[9], que fazem uso dessas classes), que permitirão ao desenvolvedor do aplicativo automatizar processos na criação de software educativo com RAM baseada em marcadores, isto através do suporte de várias funcionalidades que foram classificadas como funcionalidades gerais e funcionalidades específicas para educação (ver mais detalhes na Seção 4.1.3). Mais especificamente, o *framework*, através de suas classes, permite implementar um

[9] *Prefabs* são objetos de Unity que armazenam componentes (classes) e propriedades, e, a partir deles, podem ser criados novas instâncias, funcionando como *templates* ou objetos pré-fabricados prontos para serem reusados facilmente. Para mais informações ver em: https://docs.unity3d.com/Manual/Prefabs.html

conjunto de funcionalidades sem necessidade de programação, e, no lugar disso, arrastar e soltar classes, montando e customizando cenários com os recursos necessários (modelos 3D, imagens, marcadores etc.).

Quanto às funcionalidades disponíveis, sumariamente, o *framework* inclui os seguintes aspectos:

- A possibilidade de criar perguntas de dois tipos possíveis;
- A possibilidade de incorporar informação adicional sobre objetos através de interação com marcadores;
- A possibilidade de mudar a *transform* (rotação e escala) de objetos, através de botões da UI ou de interação com marcadores;
- A possibilidade de mudar a cor de um objeto ou o objeto todo por outro, através de interação com marcadores.

O *framework* foi criado pensando em manter a maior flexibilidade possível em relação aos aplicativos que podem ser criados, portanto, deixando disponíveis diferentes opções de configuração (através de *flags* etc.) que permitem customizar as funcionalidades dependendo das necessidades do desenvolvedor. Assim, o *framework* fornece funcionalidades prontas (especificamente para o contexto educativo) e que não limitam a flexibilidade no desenvolvimento dos aplicativos, uma vez que podem ser configuráveis dependendo das preferências do desenvolvedor, e, além disso, o desenvolvedor fica com total acesso ao código para realizar mudanças ou acrescentar o que for necessário.

4.1.1 Arquitetura de Software

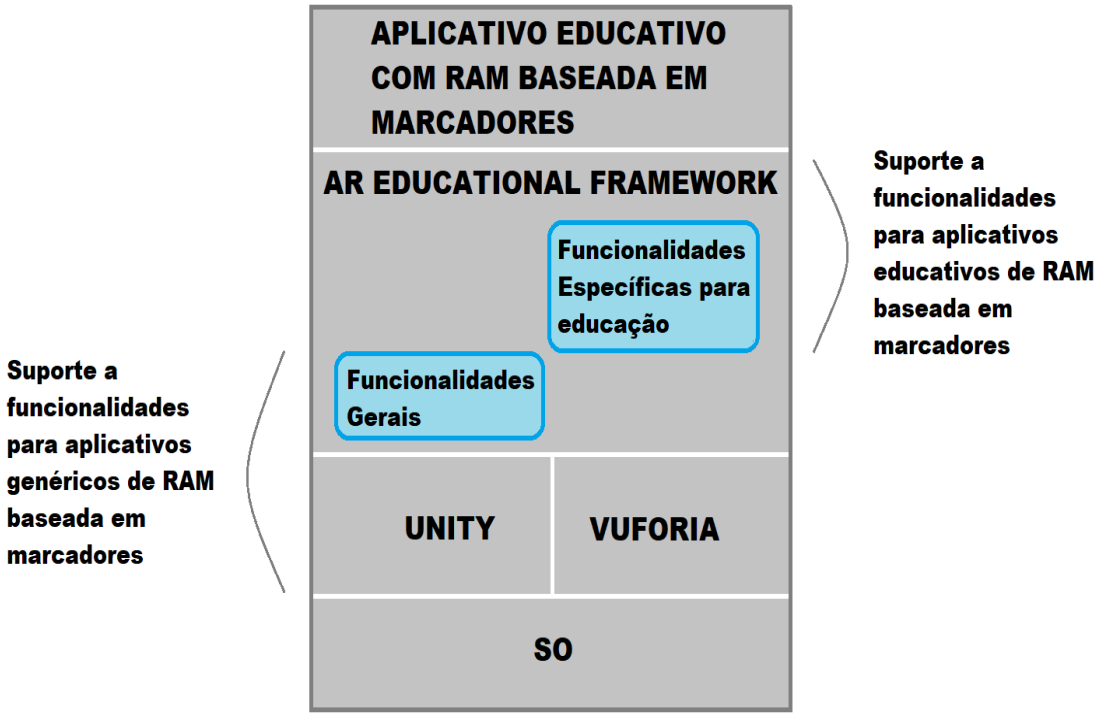

Figura 4.1: Arquitetura de software para aplicativos educativos criados com o AR Educational Framework.

A Figura 4.1 apresenta a arquitetura de software para aplicativos que fazem uso do *framework* proposto, e sobre a qual foi criado o aplicativo educativo com RAM baseada em marcadores, do estudo de caso no presente trabalho (ver Capítulo 5). O *framework* proposto funciona como camada de software intermediária construída sobre o *game engine* Unity e o SDK de RA Vuforia, os quais, em conjunto, já fornecem as funcionalidades básicas necessárias para todo sistema de RAM baseado em marcadores (captura de vídeo, reconhecimento de marcadores etc.). Assim, o *framework* proposto aproveita as funcionalidades providas por essas ferramentas e fornece novas funcionalidades para apoiar processos especificamente para o contexto educativo no desenvolvimento desse tipo de software.

4.1.2 Software Utilizado

Unity: foi escolhido o *game engine* Unity como ferramenta principal de criação do *framework* proposto, pois fornece um ambiente muito poderoso para desenvolver cenários em 3D e ainda cenários de RA sempre que *plugins* ou *frameworks* de RA

adicionais, tais como o SDK Vuforia, forem instalados em conjunto. Unity permite uma fácil integração de elementos 3D e sua curva de aprendizagem é relativamente menor do que outras ferramentas similares. De fato, Unity é uma das ferramentas mais importantes para desenvolver RA para dispositivos móveis na atualidade. “Mais do que 91% de experiências de *hololens* e 64% de aplicações de RA móveis são criados com Unity” (VERGARA, 2017). Assim, como pode ser visto, atualmente Unity ainda sendo um *game engine* voltado principalmente para a criação de *videogames*, também é considerado uma das ferramentas principais para criar RA em dispositivos móveis, sempre que for usado com ferramentas que suportem as características da RA. Unity está disponível atualmente com licença comercial e uma opção *free* que pode ser utilizada para projetos sem fins lucrativos.

Vuforia: foi escolhido o SDK Vuforia como plataforma para suportar as características da RA baseada em marcadores (tais como reconhecimento e rastreamento de marcadores) pois fornece um suporte muito poderoso e um jeito simples de usar essa tecnologia, principalmente quando é utilizada sobre o *game engine* Unity. De fato, pode-se afirmar que, embora Vuforia está disponível para desenvolver diretamente para Android e iOS, o jeito mais simples e amigável é usando Unity. Além do anterior, Vuforia também permite criar quase qualquer tipo de experiência de RA baseada em marcadores sobre dispositivos móveis, fornecendo uma grande faixa de características, e, embora não ser *open source*, suporta um tipo de licença *free* (para desenvolvimento sem fins lucrativos), que pode ser utilizada para projetos educativos.

4.1.3 Funcionalidades do AR Educational Framework

Para entender as funcionalidades do *framework* proposto, deve-se esclarecer que estas são funcionalidades de alto nível, diferentemente das funcionalidades básicas necessárias para todo software de RAM baseado em marcadores (captura de vídeo, reconhecimento de marcadores etc.). Atualmente existem já um grande número de *frameworks* ou ferramentas de desenvolvimento que suportam essas funcionalidades, e este projeto visou aproveitar essas ferramentas (no caso Unity e Vuforia) para sobre elas construir um novo *framework* que permite oferecer um

suporte voltado para a educação e de utilidade para desenvolvedores de conteúdos educativos com RAM baseada em marcadores.

Uma vez esclarecido esse ponto, as funcionalidades do *framework* têm sido classificadas em funcionalidades gerais, e funcionalidades específicas para a educação. Isso permite diferenciar funcionalidades genéricas das funcionalidades voltadas às necessidades educativas e orientadas para cenários educativos com RAM baseada em marcadores. Várias das funcionalidades foram baseadas em operações ou interações frequentes encontradas em aplicativos educacionais de RAM baseada em marcadores durante a pesquisa bibliográfica.

Deve-se notar que, embora o *framework* seja voltado para a educação, considerou-se importante que ele também forneça funcionalidades que não estão totalmente relacionadas com a educação, mas que sejam muito úteis para criar cenários bem sucedidos com RAM baseada em marcadores (funcionalidades gerais). Um exemplo deste tipo de funcionalidades é o suporte para modificação da *transform* dos objetos através de interfaces de interação adicionais (por exemplo, através de botões de UI), ao invés de unicamente através do marcador, o que permitiria mudar características de objetos virtuais aumentados tais como rotação e escala de diferentes maneiras e permitindo maior exploração. Assim, embora este suporte não esteja relacionado especificamente com cenários educativos, considera-se de grande utilidade em todo sistema de RAM baseada em marcadores. E, de fato, se isso não for considerado, os objetos aumentados seriam estáticos em relação ao marcador e teriam menor nível de exploração (o único jeito para interagir com eles seria da forma implícita através do marcador, ou seja, movimentando o objeto através do marcador), tendo desvantagens notáveis em relação a interfaces adicionais faladas, como as seguintes: quanto à rotação, a orientação do objeto em relação ao marcador estaria sempre fixa, o que dificultaria a visão do lado inferior do objeto (ou lado mais próximo ao marcador) e, quanto à escala, o usuário estaria sempre obrigado a se aproximar ou afastar do marcador para perceber mudança de tamanho no objeto. Portanto, esta situação não permite aproveitar totalmente uma das principais vantagens da RAM baseada em marcadores, e, em geral, da RA, que é poder fornecer elementos virtuais dinâmicos suportando a maior interação possível com o usuário.

4.1.3.1 Funcionalidades Gerais

São funcionalidades de grande utilidade na criação de software com RAM baseada em marcadores, que podem ser usadas não apenas em contextos educacionais. Neste projeto, este tipo de funcionalidades pode ser considerado o de menor relevância, uma vez que o *framework* foca no desenvolvimento de aplicativos educativos, e, portanto, em suportar funcionalidades especificamente para o contexto educativo. As funcionalidades deste tipo que foram incluídas no *framework*, são apresentadas a seguir:

- **Suportar a modificação da *transform* de objetos através de botões da UI (*User Interface*):** permitir, através de botões, rotacionar e escalar objetos em RA no cenário. A seguir serão explicadas, com maior detalhe, cada uma destas operações: (ver as classes do *framework* **`ButtonOperations`** e **`ButtonsPack`**, que fornecem a presente funcionalidade, no Apêndice B).

  - Rotacionar: o *framework* proposto fornece um suporte que permite rotacionar objetos em RA, de diferentes formas e através de botões. Esta rotação pode ser em vários sentidos, em torno de um dos eixos de coordenadas X, Y ou Z do espaço tridimensional, como pode ser visto na Figura 4.2. O suporte também fornece uma opção para restabelecer o objeto à rotação original e permitir fazer a rotação mudando valores, tais como velocidade de rotação, definido previamente pelo usuário.

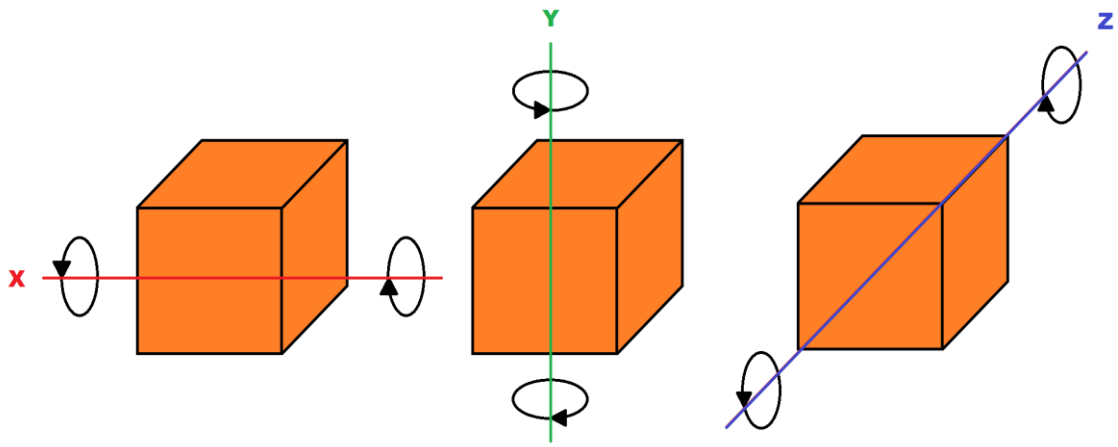

Figura 4.2: Rotação nos dois sentido possíveis em torno do eixo X (esquerda), em torno do eixo Y (meio), e em torno do eixo Z (direita), suportados pelo *framework*.

- Escalar: o *framework* proposto fornece um suporte que permite mudar o tamanho de objetos em RA através de botões. Especificamente, é possível aumentar ou diminuir o tamanho do objeto de forma proporcional (na mesma magnitude) em todos seus eixos de coordenadas (X, Y e Z), este foi considerado o jeito de escalar objetos tridimensionais mais útil tanto para aplicativos de RA gerais quanto educativos, em que comumente se quer aumentar ou diminuir os objetos de jeito proporcional e não em eixos específicos, permitindo ao objeto manter a sua forma original enquanto for escalado. O suporte também fornece uma opção para restabelecer o objeto ao seu tamanho original, e permitir escalar mudando valores tais como a velocidade do escalado, definida previamente pelo usuário. Nota-se que o processo de escalar objetos é feito mantendo a posição do objeto em relação a sua parte inferior, ou seja, mantendo a distância em relação ao chão (vide Figura 4.3). Isto evita que o objeto possa se sobrepor ou penetrar superfícies sobre as quais está colocado, e, neste caso, se sobrepor ao marcador, relembrado que na RA baseada em marcadores, objetos 3D aumentados são apresentados comumente acima de marcadores.

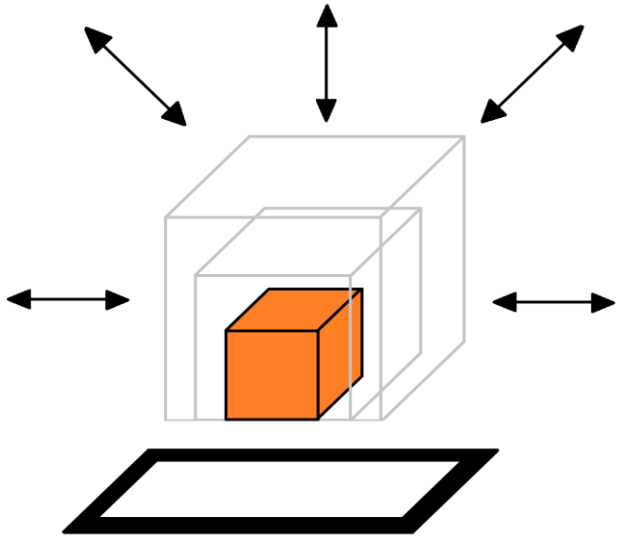

Figura 4.3: Esboço do processo de escalar objetos suportado pelo *framework.*

Como pode ser visto na Figura 4.3, o processo de escalar acontece de forma proporcional em todos os eixos, assim, na figura, o cubo aumenta seu

tamanho em X, Y, e Z na mesma magnitude ao mesmo tempo. Nota-se que, ainda no processo de escalar, a posição do objeto em relação a sua parte inferior se mantém, e, portanto, o lado inferior do cubo fica na mesma posição, embora seja mudado seu tamanho, evitando que possa se sobrepor na posição do marcador.

- **Suportar a modificação da *transform* de objetos através de marcadores de controle:** permitir, através do uso de marcadores de controle, rotacionar e escalar objetos em RA no cenário. A seguir serão explicadas, com maior detalhe, cada uma destas operações: (ver a classe do *framework* **`ChangeElement`**, que fornece a presente funcionalidade, no Apêndice B).

  - Rotacionar: o *framework* proposto fornece um suporte que permite rotacionar objetos em RA através do uso de marcadores de controle. Neste caso, permite-se mudar a rotação original do objeto por uma nova rotação definida pelo usuário, sempre que um marcador de controle estiver junto ao marcador principal vinculado ao objeto. Figura 4.4 apresenta esse processo para um objeto cubo sendo rotacionado no eixo Y. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o objeto voltará a sua rotação original.

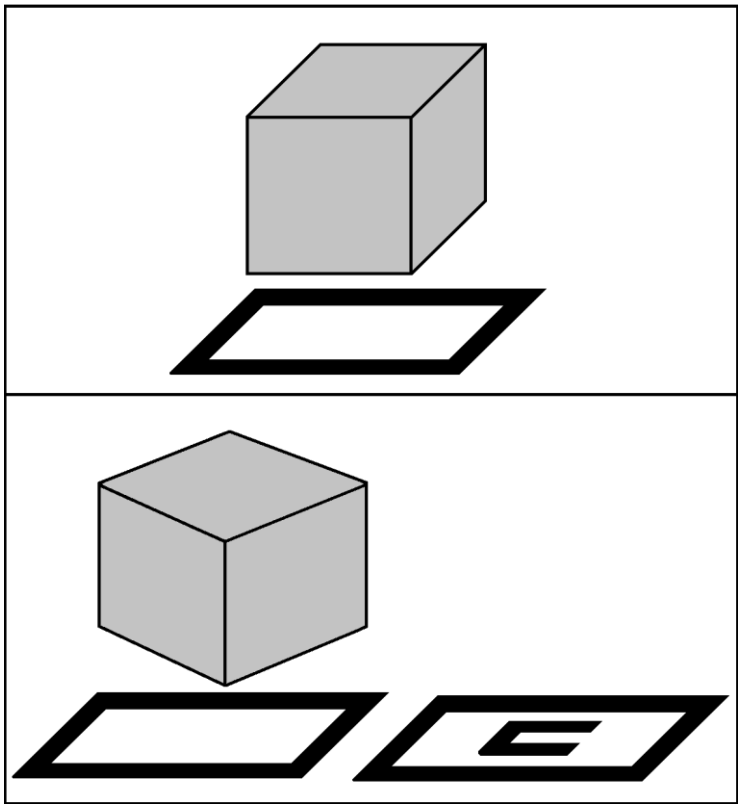

Figura 4.4: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz mudar a rotação original do objeto por uma nova.

- Escalar: o *framework* proposto fornece um suporte que permite escalar objetos em RA através do uso de marcadores de controle. Neste caso, permite-se mudar a escala original do objeto por uma nova escala definida pelo usuário, sempre que um marcador de controle estiver junto ao marcador principal vinculado ao objeto. Figura 4.5 ilustra esse processo para um objeto cubo sendo escalado em um tamanho maior. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o objeto voltará a sua escala original. Uma diferença em relação ao processo de escalar através de botões é que, deste outro jeito, o processo de escalar não precisa ser proporcional, assim a nova escala poderia mudar o objeto em diferentes magnitudes para cada eixo (X, Y e Z).

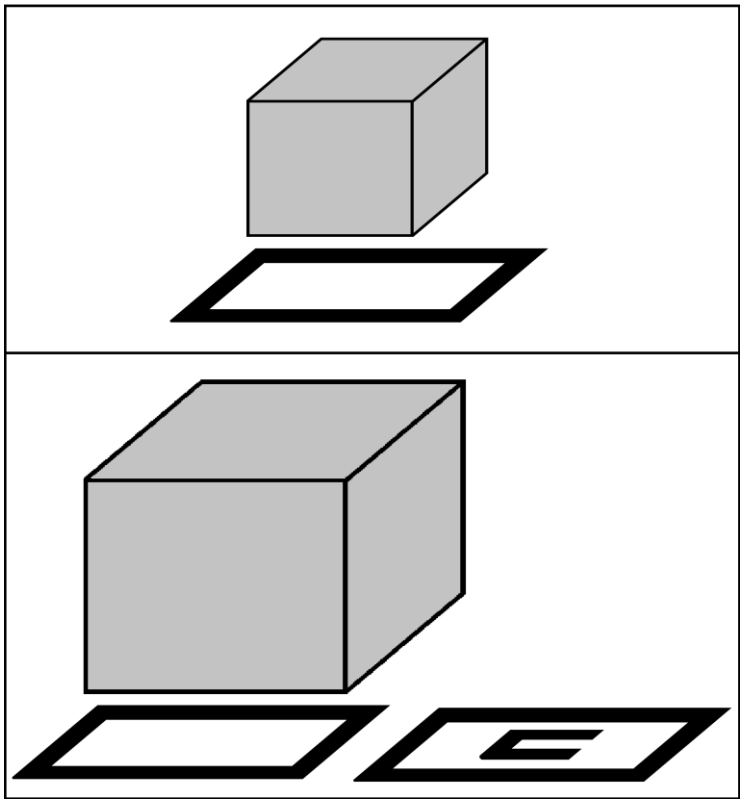

Figura 4.5: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz mudar a escala original do objeto por uma nova.

4.1.3.2 Funcionalidades Específicas para Educação

São funcionalidades voltadas especificamente para o contexto educativo na criação de software com RAM baseada em marcadores, e correspondem às funcionalidades de maior peso e relevância neste projeto. No caso do *framework* proposto, estas funcionalidades além de serem específicas para educação, também têm as características de serem de cunho geral na educação, o que significa que são úteis para qualquer tipo de aplicativos educativos sem estar focado em nenhuma área particular da educação (ou seja, são igualmente úteis para Matemática, Química etc.). As funcionalidades deste tipo que foram incluídas no *framework*, são apresentadas a seguir:

- **Suporte à modificação dos materiais de objetos através de marcadores de controle:** o *framework* proposto fornece um suporte que permite mudar os

Materiais[10] de um objeto em RA por novos materiais definidos pelo usuário, sempre que um marcador de controle estiver junto ao marcador principal vinculado ao objeto. Figura 4.6 ilustra esse processo, em que é mudado o material original de cor cinza em um objeto cubo por um novo material de cor azul. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o objeto voltará a ficar com seu material original. (Ver a classe do *framework* **`ChangeElement`**, que fornece a presente funcionalidade, no Apêndice B)

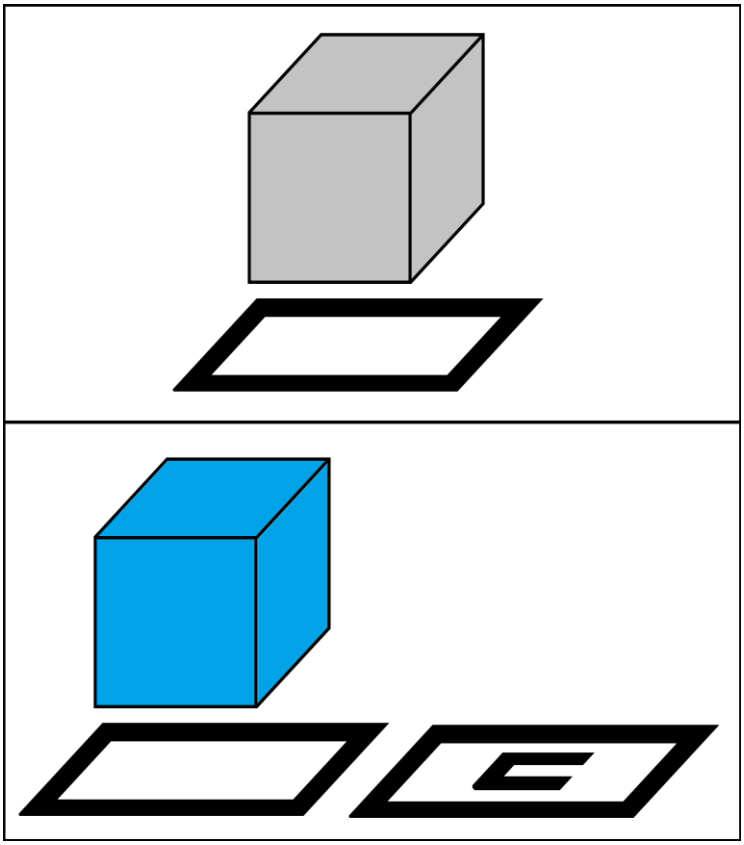

Figura 4.6: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz mudar o material original do objeto por um novo de cor azul.

- **Suporte à troca de objetos através de marcadores de controle:** o *framework* proposto fornece um suporte que permite trocar um objeto em RA por outro novo definido pelo usuário, sempre que um marcador de controle

[10] Um Material é um elemento do Unity que permite definir características de um objeto tais como cor, textura, opacidade etc.). Para mais informações, observar: https://docs.unity3d.com/Manual/Shaders.html

estiver junto ao marcador principal vinculado ao objeto. Figura 4.7 apresenta esse processo, em que um objeto cubo original é trocado por uma esfera. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o objeto original voltará a ficar na cena como estava inicialmente. (Ver a classe do *framework* **`ChangeElement`**, que fornece a presente funcionalidade, no Apêndice B)

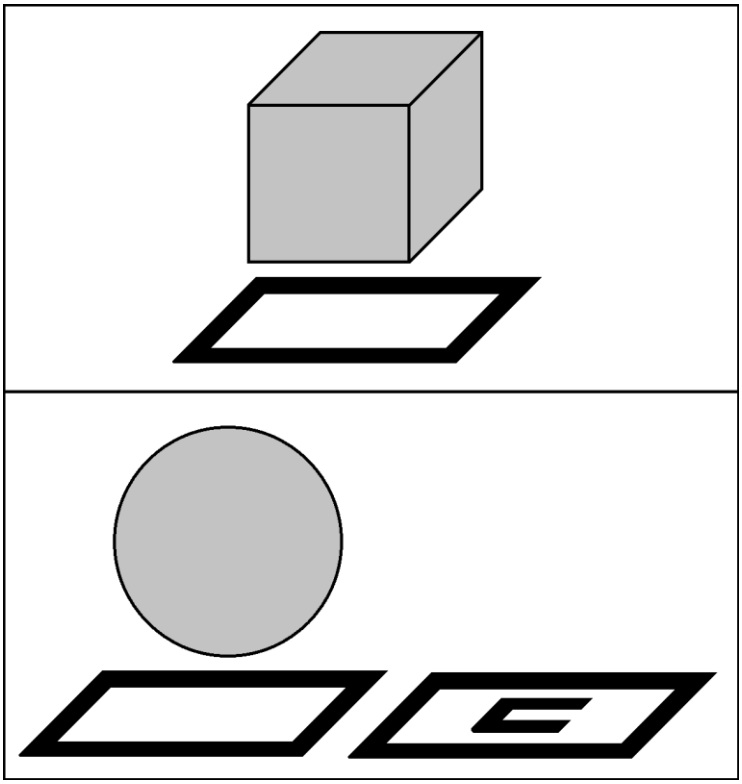

Figura 4.7: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz trocar o objeto original (cubo) por um novo (esfera).

- **Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle:** permitir, através de marcadores de controle, ativar diferentes tipos de informações (textos 3D ou imagens) sobre objetos em RA. A seguir será explicada, com maior detalhe, a operação descrita para cada um dos tipos de informações adicionais suportados. (Ver a classe do *framework* **`ActivateInformation`**, que fornece a presente funcionalidade, no Apêndice B)

  - Textos 3D: o *framework* proposto fornece um suporte que permite ativar textos 3D sobre objetos em RA, sempre que um marcador de controle estiver

junto ao marcador principal vinculado ao objeto. Figura 4.8 ilustra esse processo, em que um texto 3D é ativado sobre um objeto cubo. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, o texto 3D será desativado, ficando a cena como estava inicialmente.

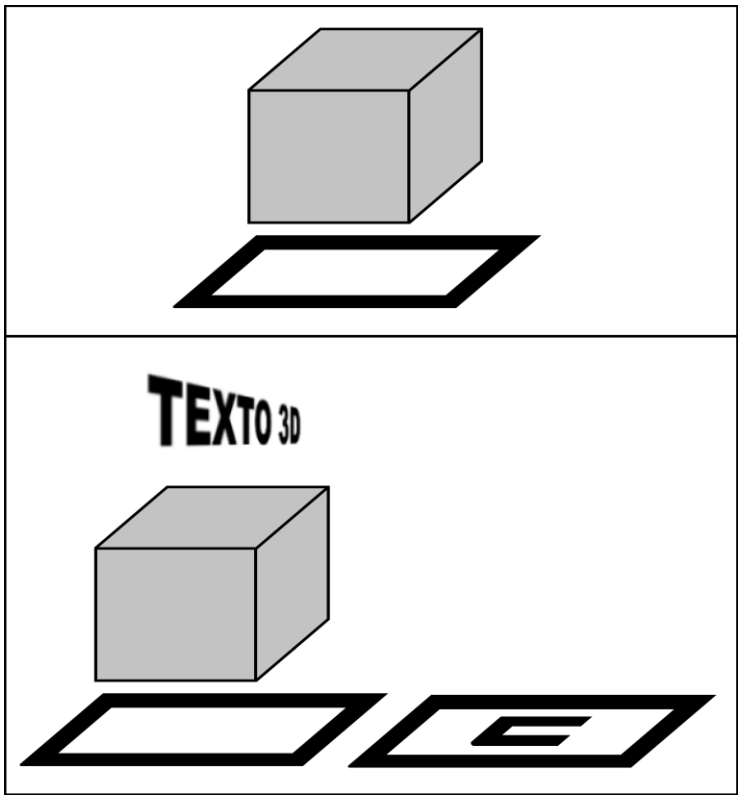

Figura 4.8: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz ativar um texto 3D (usado como informação adicional) sobre o objeto.

- Imagens: o *framework* proposto fornece um suporte que permite ativar imagens sobre objetos em RA, sempre que um marcador de controle estiver junto ao marcador principal vinculado ao objeto. Figura 4.9 ilustra esse processo, em que uma imagem é ativada sobre um objeto cubo. Deve-se notar que, uma vez que o marcador de controle for separado do marcador principal, a imagem será desativada, ficando a cena como estava inicialmente.

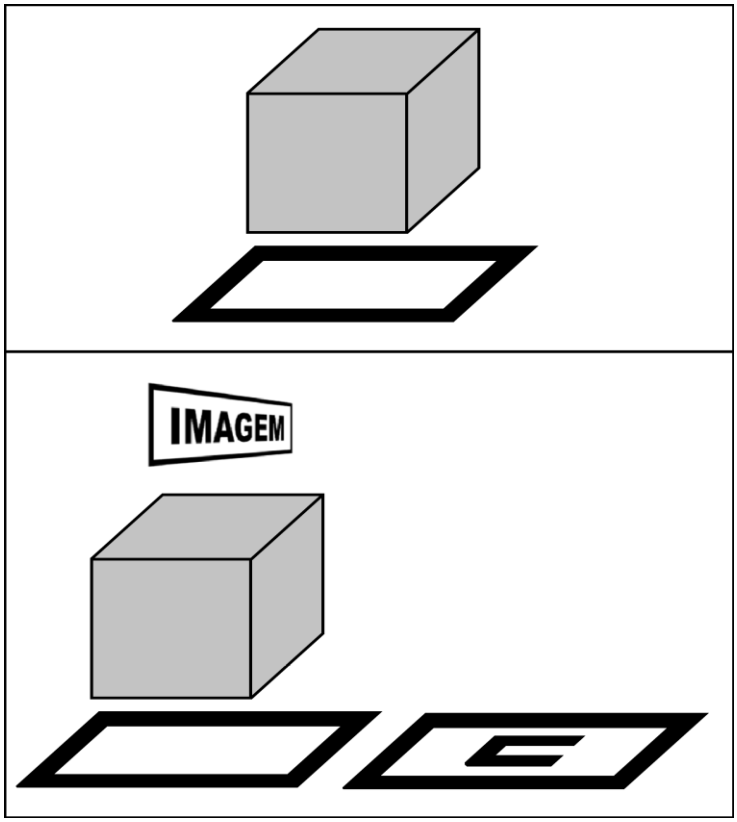

Figura 4.9: A imagem acima apresenta um objeto com seu marcador vinculado, já a imagem abaixo apresenta a mesma cena acrescentando o marcador de controle "C", o qual faz ativar uma imagem (usada como informação adicional) sobre o objeto.

- **Suporte de Perguntas:** consiste em um suporte para permitir a criação de perguntas do tipo *Multiple Choice Questions* (MCQ). Duas abordagens que permitem diferentes tipos de perguntas foram definidas dependendo do tipo de alternativas de respostas utilizado nas perguntas, ambas estão detalhadas a seguir:

  - Abordagem 1: abordagem de perguntas em que as alternativas de resposta são elementos 2D (tipo UI tradicional) apresentados ao mesmo tempo. Para esta abordagem, o *framework* proposto permite como alternativas de resposta o uso tanto de textos 2D quanto de imagens. (Ver a classe do *framework* **`QuestionController1`**, que fornece a presente funcionalidade para este tipo de perguntas, no Apêndice B)

  - Abordagem 2: abordagem de perguntas em que as alternativas de resposta são elementos em RA apresentados por turno. Para esta abordagem, o *framework* proposto permite como alternativas de resposta o uso de todo tipo de elementos em RA (p.ex., modelos 3D, imagens em RA). (Ver a classe do

framework **`QuestionController2`**, que fornece a presente funcionalidade para este tipo de perguntas, no Apêndice B)

Um esboço com a aplicação de cada abordagem será apresentado mais adiante nesse capítulo.

Como pode-se notar, as duas abordagens comentadas permitem utilizar diferentes tipos de elementos virtuais como alternativas das perguntas (textos 2D, imagens ou ainda elementos 3D em RA), isto permite oferecer maiores possibilidades na realização de perguntas além dos tipos de perguntas comuns em que as alternativas são textos 2D. Por outro lado, nota-se que, na Abordagem 2, os objetos são apresentados por turno (e não ao mesmo tempo como na Abordagem 1), isso devido ao fato de que essa foi considerada a maneira mais adequada de apresentar elementos virtuais em RA como alternativas, que garante que cada um dos elementos possam ser explorados sem problemas.

Seja qual for a abordagem utilizada, o suporte de perguntas deve permitir ao aluno realizar certas operações para cada pergunta. A seguir são apresentadas aquelas que foram consideradas mais relevantes. (Para ver todas as possibilidades das perguntas, ver a documentação das classes do *framework* **`QuestionController1`** e **`QuestionController2`**, as quais fornecem os dois tipos de perguntas para as duas abordagens definidas, respetivamente, estão disponíveis nos Apêndices B e C)

- Possibilidade de definir um número qualquer (maior do que um) de alternativas de resposta, e indicar qual das alternativas corresponde à alternativa certa. Assim por exemplo, se vai ser utilizada a Abordagem 1, com imagens como as alternativas de resposta, e o usuário quer usar 3 alternativas, ele deverá fornecer as 3 imagens necessárias para representar cada uma das alternativas, e indicar qual delas corresponde a alternativa certa.
- Disponibilidade de um *feedback* automático de resposta correta ou errada, após a pergunta ser respondida.

- Possibilidade do uso de tempo, permitindo ao usuário customizá-lo segundo sua necessidade.
- Possibilidade de repetir uma pergunta, imediatamente após ter sido respondida.
- Possibilidade de apresentar as alternativas em ordem aleatória.
- Possibilidade de ver a resposta correta da pergunta.
- Possibilidade de utilizar elementos adicionais, além dos elementos principais gerenciados pelo *framework*, fazendo o suporte ficar mais flexível e permitindo que o usuário possa acrescentar, no contexto da pergunta, outros elementos gerenciando-os pela sua responsabilidade.

Diferentes elementos que serão gerenciados pelo suporte de perguntas do *framework* permitirão realizar as operações anteriores, estes elementos serão definidos a seguir, sendo indicados em dois esboços de cenários usando as duas abordagens possíveis. Inicialmente, será apresentado o esboço da Abordagem 1 com seus elementos, e, após isso, o esboço da Abordagem 2 com seus respectivos elementos.

As Figuras 4.10, 4.11 e 4.12 ilustram o esboço de um cenário criado com o suporte de perguntas do *framework* aplicando a Abordagem 1, a Figura 4.10 ilustra quando está sendo respondida a pergunta, a Figura 4.11 ilustra após responder a pergunta, e a Figura 4.12 ilustra quando está sendo apresentada a resposta certa.

Na Figura 4.10, o número 1 indica um marcador de RA, e os outros números indicam os elementos virtuais relacionados com o suporte do *framework*, o número 2 indica um cubo em RA sobre o que se faz a pergunta, o número 3 indica um elemento adicional (no caso, um texto de pergunta), o número 4 indica o texto do tempo, e o número 5 as alternativas de resposta (nota-se que na figura esquerda são usados textos 2D como alternativas de resposta, e na figura direita são usadas imagens).

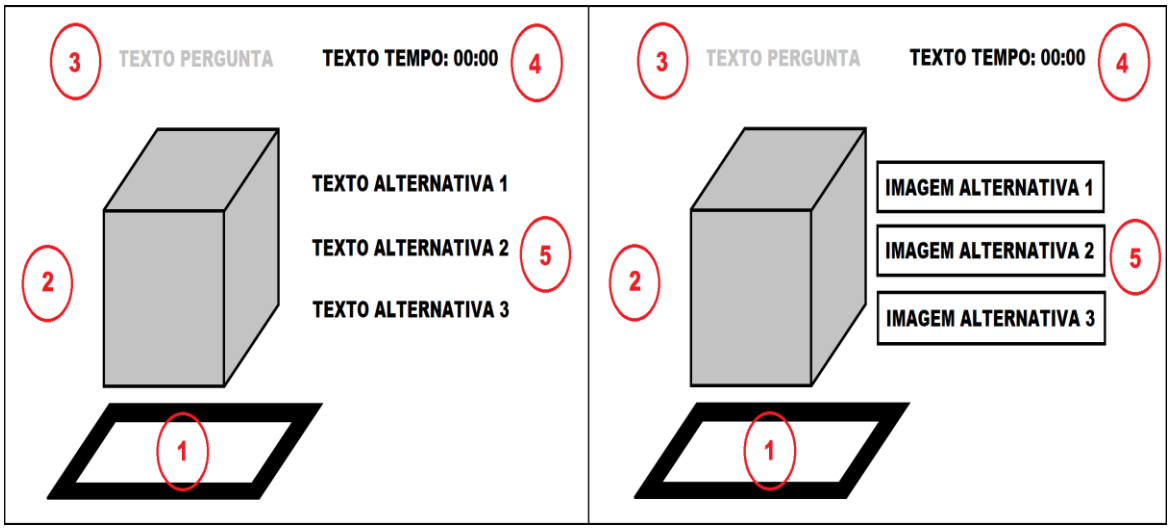

Figura 4.10: Aplicação da Abordagem 1 em um esboço de um cenário, enquanto está sendo respondida a pergunta. Na figura esquerda são usados textos 2D como alternativas de resposta, e na figura direita imagens (elementos número 5).

Na Figura 4.11, o número 1 indica o botão repetir pergunta, o número 2 o botão de ver resposta correta da pergunta, e o número 3 o elemento de *feedback* (uma imagem).

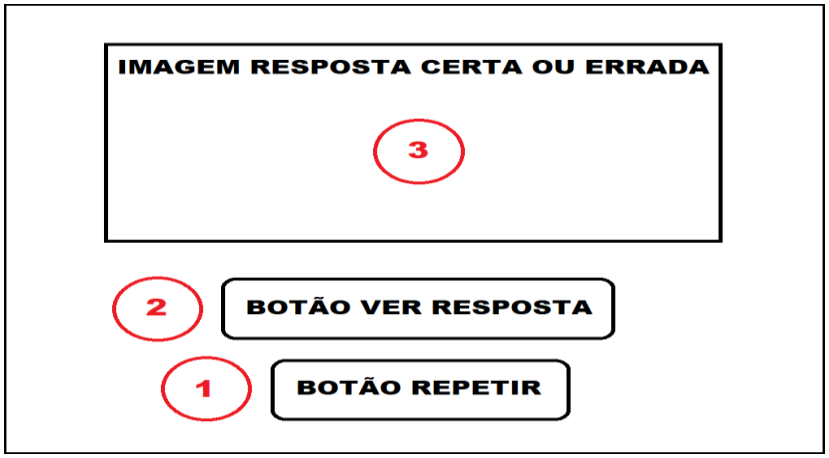

Figura 4.11: Aplicação da Abordagem 1 em um esboço de um cenário, após ser respondida a pergunta.

Já a Figura 4.12 apresenta vários dos elementos presentes na Figura 4.10, porém com as seguintes diferenças: o elemento adicional, neste caso, é um texto de resposta (ao invés do texto de pergunta inicial). Além disso, uma das alternativas está marcada com uma imagem com o símbolo "*Check*" para informar que corresponde à resposta certa. Finalmente, fica também disponível um botão de repetir a pergunta.

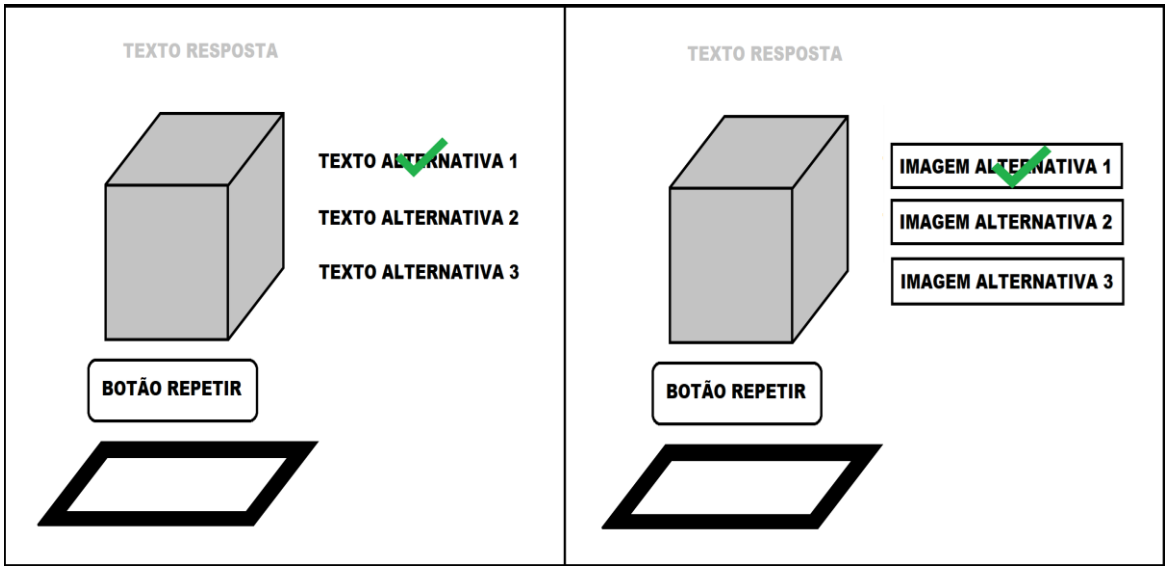

Figura 4.12: Aplicação da Abordagem 1 em um esboço de um cenário, enquanto está sendo apresentada a resposta correta da pergunta. Na figura esquerda foram usados textos 2D como alternativas de resposta, e na figura direita imagens.

A seguir, serão apresentados detalhes dos elementos presentes com a Abordagem 1:

- Alternativas de Resposta (ver Figura 4.10, elementos número 5): são elementos virtuais representando as alternativas de resposta da pergunta. Nesta abordagem, podem ser textos 2D, ou imagens, e poderão ser selecionados como resposta pressionando sobre eles.
- Texto do Tempo (ver Figura 4.10, elemento número 4): texto que funciona como relógio (cronômetro), formatado corretamente, e que apresenta o tempo disponível para responder a uma pergunta, podendo ser configurado pelo usuário. Uma vez que for utilizado, fica relacionado com a resolução da pergunta, assim, finalizar o tempo significa que o usuário falhou em responder à pergunta corretamente.
- Elementos de *Feedback* (ver Figura 4.11, elemento número 3): são imagens que serão ativadas imediatamente após ter sido respondida a pergunta, e que irão indicar se a pergunta foi respondida corretamente ou incorretamente.
- Botão Ver Resposta (ver Figura 4.11, elemento número 2): botão que permite ver a resposta correta da pergunta, após ter sido respondida. No caso de usar a abordagem padrão de ver a resposta fornecida pelo *framework*, este botão fará a cena ficar como apresentado na Figura 4.12.

- Botão Repetir Pergunta (ver Figura 4.11, elemento número 1): botão que permite repetir a pergunta atual após ter sido respondida, permitindo reiniciar o estado de todos os elementos da pergunta.
- Elementos Adicionais (ver Figura 4.10, elemento número 3): são elementos adicionais que poderiam ser requeridos para uma pergunta. O suporte de perguntas do *framework* terá uma gestão menor sobre eles, unicamente para ativá-los e desativá-los quando for necessário, garantindo que o fluxo de pergunta e resposta (com os elementos associados) aconteça da maneira correta. Estes elementos podem ser imagens, botões, elementos em RA, ou qualquer tipo de elemento virtual requerido, e deverão ser gerenciados independentemente pelo usuário. São permitidos diferentes tipos de elementos adicionais, dependendo do seu estado de ativação no fluxo da pergunta, eles são: elementos adicionais ativos unicamente quando está sendo respondida a pergunta (Estado apresentado na Figura 4.10), elementos adicionais ativos unicamente após responder a pergunta (Estado apresentado na Figura 4.11), elementos adicionais ativos quando está sendo apresentada a resposta correta (Estado apresentado na Figura 4.12), e, finalmente, elementos adicionais ativos durante a pergunta toda (em todos os estados apresentados nas Figuras 4.10, 4.11 e 4.12).

As Figuras 4.13, 4.14 e 4.15 ilustram o esboço de um cenário criado com o suporte de perguntas do *framework* aplicando a Abordagem 2, a Figura 4.13 ilustra quando está sendo respondida a pergunta, a Figura 4.14 ilustra após responder a pergunta, e a Figura 4.15 ilustra quando está sendo apresentada a resposta certa.

Na figura 4.13, o número 1 indica um marcador de RA, e os outros números indicam os elementos virtuais relacionados com o suporte do *framework*, o número 2 indica o botão de confirmar alternativa escolhida, o número 3 indica os botões de mudar alternativa (em diferentes sentidos), o número 4 as alternativas de resposta (no caso, são modelos 3D em RA de figuras geométricas apresentadas por turno, a esfera laranja está ativada e o

cilindro desativado), o número 5 indica um elemento adicional (texto de pergunta), e o número 6 indica o texto do tempo.

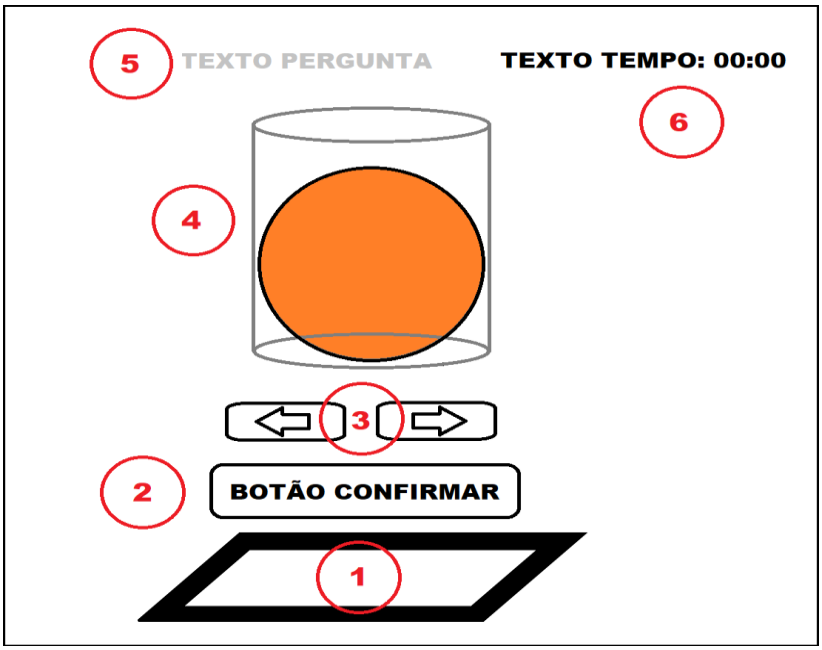

Figura 4.13: Aplicação da Abordagem 2 em um esboço de um cenário, enquanto está sendo respondida a pergunta.

Na Figura 4.14, o número 1 indica o botão repetir pergunta, o número 2 o botão de ver resposta correta da pergunta, e o número 3 o elemento de *feedback* (uma imagem).

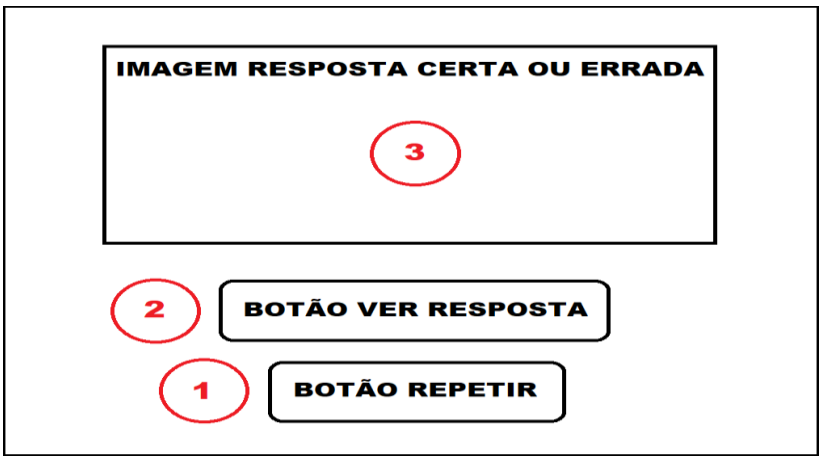

Figura 4.14: Aplicação da Abordagem 2 em um esboço de um cenário, após ser respondida a pergunta.

A Figura 4.15 apresenta vários dos elementos presentes na Figura 4.13, porém com as seguintes diferenças: o elemento adicional, neste caso, é um texto de resposta, além disso, fica ativo unicamente o modelo 3D em RA

que corresponde à resposta correta (no caso, a esfera). Finalmente, fica também disponível um botão de repetir a pergunta.

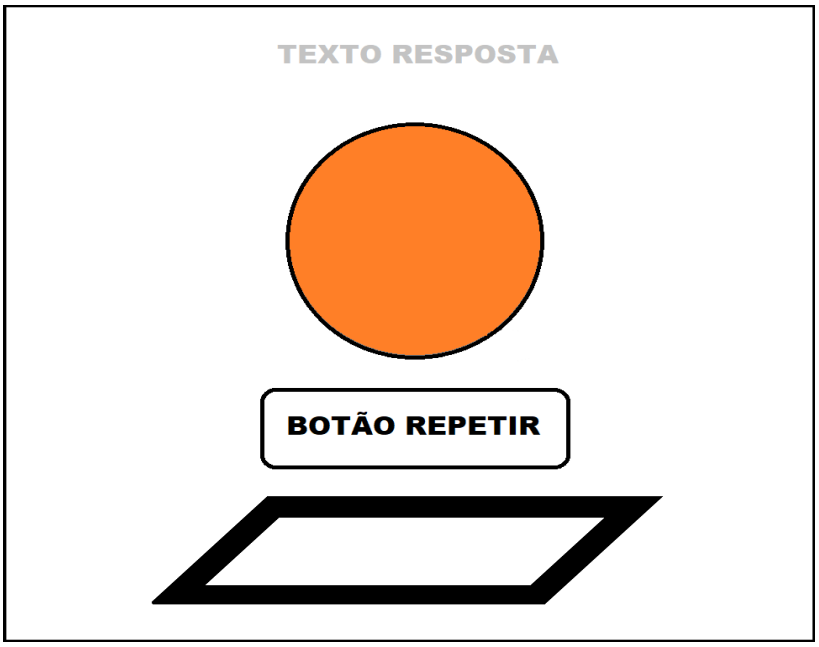

Figura 4.15: Aplicação da Abordagem 2 em um esboço de um cenário, enquanto está sendo apresentada a resposta correta da pergunta.

A seguir, serão apresentados detalhes dos elementos presentes na Abordagem 2. Nota-se que muitos deles mantêm o mesmo comportamento da Abordagem 1:

- Alternativas de Resposta (ver Figura 4.13, elementos número 4): são elementos em RA apresentados por turno, e que podem ser explorados um por um e selecionados como resposta através de botões.
- Botões de Mudar Alternativa (ver Figura 4.13, elementos número 3): são dois botões que permitem explorar todas as alternativas disponíveis uma por uma (relembrando que para esta abordagem cada alternativa é mostrada por turno), para isso estes botões vão permitir desativar a alternativa atual e ativar uma próxima (ou seja, desativar o elemento em RA atual e ativar outro).
- Botão Confirmar Alternativa (ver Figura 4.13, elemento número 2): botão que permite confirmar a alternativa selecionada atualmente.
- Texto do Tempo (ver Figura 4.13, elemento número 6): tem o mesmo comportamento na Abordagem 1.

- Elementos de *Feedback* (ver Figura 4.14, elemento número 3): têm o mesmo comportamento na Abordagem 1.
- Botão Ver Resposta (ver Figura 4.14, elemento número 2): tem comportamento similar presente na Abordagem 1. Neste caso, se o usuário usar a abordagem padrão de ver a resposta fornecida pelo *framework*, este botão fará a cena ficar como ilustrado na Figura 4.15.
- Botão Repetir Pergunta (ver Figura 4.14, elemento número 1): tem o mesmo comportamento na Abordagem 1.
- Elementos Adicionais (ver Figura 4.13, elemento número 5): têm o mesmo comportamento na Abordagem 1.

4.2 Metodologia Proposta

A metodologia proposta tem como objetivo guiar os desenvolvedores em todo processo de criação de aplicativos educativos com a tecnologia de RAM. Para isso, esta metodologia está composta de uma série de orientações teóricas *(guidelines)* a serem seguidas no desenvolvimento do aplicativo, que consideram o uso do *framework* proposto. Estas *guidelines* são apresentadas a seguir.

4.2.1 Guidelines Para Desenvolvimento De Aplicativos De RA Educacionais Com o AR Educational Framework

1) **Definir Os Cenários Do Aplicativo De RA Educacional:**
   a) **Definir Cenários Do Aplicativo:** Neste passo, devem-se definir os cenários do aplicativo com as atividades a serem realizadas, incluindo definição de perguntas a serem realizadas e a escolha da abordagem de perguntas mais adequada em cada cenário, das duas definidas pelo *framework* (ver Abordagem 1 e Abordagem 2, na funcionalidade “Suporte de Perguntas”, descrita na Seção 4.1.3.2). Neste passo, tem grande relevância definir claramente objetivos de aprendizagem a serem atingidos em cada cenário (o que se quer que o aprendiz aprenda).

2) **Criar Recursos:**
   a) **Criar Marcadores De RA:** No caso de querer usar marcadores com imagens próprias e não os marcadores *default* da Vuforia, neste passo, devem ser criados os novos marcadores. Este processo deverá ser feito através da página web[11] da Vuforia, obtendo um pacote Unity do banco de dados dos marcadores criados.
   b) **Criar Elementos Virtuais:** Neste passo, devem ser criados os elementos audiovisuais necessários para o aplicativo, tais como: modelos 3D, imagens ou texturas 2D, etc., fazendo uso de programas para esses fins (p.ex. Blender, Autodesk Maya, Photoshop etc.).

3) **Configuração Da Vuforia:**
   a) **Configurar O Projeto Do Unity Para Suportar O Uso Da Vuforia:**
      - **Importar Recursos Da Vuforia:** Substituir a **`MainCamera`** *default* do Unity por uma **`ARCamera`** da Vuforia (criar fazendo: **`GameObject/Vuforia/ARCamera`**). Nesta operação, se importarão, automaticamente, todos os recursos necessários da Vuforia no projeto atual do Unity.
      - **Habilitar Vuforia:** Habilitar Vuforia no projeto do Unity para a plataforma destino do aplicativo, em: **`File/BuildSettings`**, selecionando a plataforma (Android), fazendo click em **`PlayerSettings`**, e, finalmente, nas propriedades do inspetor do Unity em: **`XRSettings`**, ativar a opção: **`Vuforia Augmented Reality Supported`**.
   b) **Importar Marcadores De RA E Habilita-los Para Seu Uso:** No caso de usar marcadores próprios e não marcadores *default* da Vuforia, importar no projeto o pacote do banco de dados dos marcadores criado na fase 2. Finalmente, para poder usar os marcadores importados, deve-se redefinir o banco de dados dos marcadores utilizados no projeto, para isso: selecionar a **`ARCamera`**, então nas propriedades do

[11] A página web da Vuforia pode ser acessada através do seguinte link: https://developer.vuforia.com/

inspetor do Unity fazer click em "**Open Vuforia Configuration**", e copiar o código de licença do banco de dados importado no campo: **App License Key**.

4) **Configuração Do Framework:**
   a) **Configurar Número Máximo De Marcadores Simultâneos:** Selecionar a **ARCamera**, fazer click em "**Open Vuforia Configuration**", e, finalmente, no campo **Max Simultaneous Tracked Images** indicar o número máximo de marcadores simultâneos a suportar. Para usos básicos do *framework*, normalmente será suficiente com configurar o valor 2 neste campo.
   b) **Importar Framework:** Importar no projeto, o pacote "**AR Educational Framework.unitypackage**", que possui os recursos do *framework,* disponível no link: https://www.dropbox.com/s/oftxi7jj2kt7dkn/AR%20Educational%20Framework.unitypackage?dl=0[12]
   c) **Configurar Marcadores Do Framework Na Cena:** O *framework* utiliza marcadores próprios para suportar suas funcionalidades (ao invés dos marcadores criados de jeito normal), portanto para usar marcadores com o *framework*, deve-se arrastar na cena do Unity o marcador disponível chamado **ImageTarget1Prefab**, localizado na pasta com o caminho: **AR Educational Framework/Resources**. Este marcador possui as configurações necessárias para ser usado com o *framework*, e será só com marcadores com esta configuração que deverá ser utilizado o *framework*.
   d) **Configuração Adicional Sugerida Usando Vários Marcadores:** Sempre que for necessário usar vários marcadores simultâneos com o *framework*, recomenda-se, adicionalmente, configurar a **ARCamera** da cena como **ESPECIFIC_TARGET** na sua opção ***World Center Mode***, e arrastar o marcador vinculado aos objetos (ou seja, o marcador que tem como filhos os objetos) no campo ***World Center***.

[12] Todas as referências ao pacote "AR Educational Framework.unitypackage" neste documento, se referem a este link.

Para facilitar as fases 3 e 4 (correspondentes às configurações necessárias para o *framework*), é fornecido um tutorial que inclui imagens de capturas de tela chamado “Tutorial Configuração AR Educational Framework” no Apêndice A.

5) **Criação Dos Cenários:**
   a) **Importar Elementos Virtuais:** Importar no projeto os elementos virtuais criados na fase 2, e que serão usados no aplicativo.
   b) **Criação dos Cenários:** Criar os cenários de RA definidos, usando os marcadores necessários, e fazendo uso das funcionalidades do *framework* através das suas classes.

Capítulo 5

ESTUDO DE CASO

Esse capítulo apresenta um estudo de caso realizado que consistiu na criação de um aplicativo educacional de RAM para telefones Android com dois cenários disponíveis, um deles com conteúdo para crianças na área de Geometria e o outro na área de Biologia, seguindo as *guidelines* definidas para o apoio no desenvolvimento de aplicativos de RA educacionais com o *framework* proposto, e levado a cabo pelo próprio autor. O objetivo deste estudo de caso consiste em avaliar o *framework* por meio de sua utilização no desenvolvimento de um aplicativo educacional completo com a tecnologia de RAM baseada em marcadores, tentando determinar se é possível o desenvolvimento desse tipo de aplicativos com facilidade, quais são os principias ganhos do *framework*, qual o seu nível de flexibilidade no desenvolvimento, e, além disso, determinar se as funcionalidades fornecidas podem ser usadas para qualquer área da educação, afinal são de cunho geral na educação.

Os modelos do primeiro cenário do aplicativo foram criados pelo próprio autor com o software Blender, já para o segundo cenário foi utilizado o modelo 3D "Human_heart" de um coração humano (ver Figura 5.6), disponível no link:

https://www.blendswap.com/blends/view/73924[13]

O APK do aplicativo desenvolvido, o projeto Unity, e as imagens dos marcadores necessários podem ser obtidos na pasta: Aplicativo Estudo De Caso, no link:

[13] Todas as referências ao modelo 3D "Human_heart" neste documento, se referem a este link.

https://www.dropbox.com/sh/fvtky1q8t42qtt3/AACUBY030KvDiU_tzRwL8qSaa?dl=0[14]

Instruções do uso do aplicativo podem ser vistas em Apêndice E, além disso tutoriais destelhando os passos para a criação da primeira pergunta de cada cenário do aplicativo, podem ser vistos em Apêndice F.

5.1 Aplicação Das Guidelines Para Desenvolvimento De Aplicativos De RA Educacionais Com O AR Educational Framework, No Aplicativo Desenvolvido No Estudo De Caso

1) **Definir Os Cenários Do Aplicativo De RA Educacional:**
   a) **Definir Cenários Do Aplicativo:** Foram definidos dois cenários para o aplicativo, que são explicados a seguir:

      Cenário 1: Este cenário está focado no estudo da área de Geometria, e, especificamente, no pensamento espacial. O cenário consistiu numa fase inicial de perguntas (sem RA), e uma fase final de exploração em RA para cada pergunta. Na fase de perguntas é apresentada uma imagem 2D de uma figura geométrica sobre a qual vai ser realizada a pergunta (ver Figura 5.1), e três imagens representando possíveis visões da figura (visões da frente, topo, e lado) usadas como as alternativas de resposta (ver Figura 5.2), então o aprendiz deverá escolher a única imagem alternativa que representa uma visão certa da figura em questão.

[14] Todas as referências à pasta “Aplicativo Estudo De Caso” neste documento, se referem a este link.

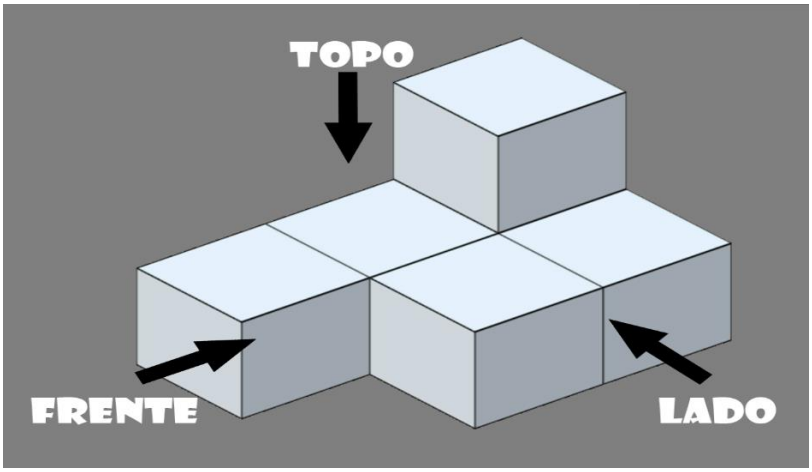

Figura 5.1: Imagem 2D de figura geométrica sobre a qual vai ser realizada a pergunta.

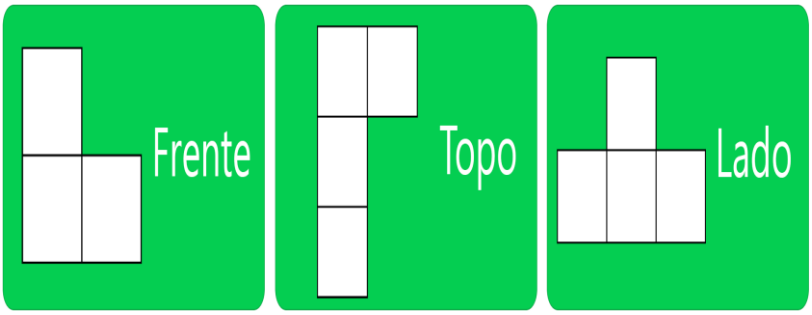

Figura 5.2: Imagens das alternativas representando possíveis visões (visões da frente, topo, e lado) da figura geométrica da pergunta. Para a figura geométrica da Figura 5.1, a alternativa correta é a imagem da esquerda (com o texto Frente).

Finalmente, a fase de exploração permite explorar a figura geométrica da pergunta através de um modelo 3D em RA, e ver a resposta certa, sendo possível mudar o seu tamanho e rotação através de botões, e ativar eventos, por meio de um marcador de controle, que permitem tanto mostrar um texto 3D, indicando as faces da visão certa da figura, quanto mudar essas mesmas faces para a cor vermelha a fim de melhorar sua visibilidade.

Nota-se que, para este cenário, foram definidas perguntas usando a primeira abordagem de perguntas definida pelo *framework* (alternativas são imagens 2D apresentadas ao mesmo tempo). Finalmente, para este cenário, foi definido como objetivo de aprendizagem melhorar a capacidade para visualizar mentalmente objetos 3D a partir de diferentes posições.

Cenário 2: Este cenário está focado no estudo da área de Biologia, e, especificamente, no estudo do coração humano. O cenário consistiu numa fase de treinamento em RA e uma fase de perguntas também em RA. Na fase de treinamento, é apresentado um modelo 3D de um coração em RA, sendo possível mudar seu tamanho e rotação com botões, e, além disso, ativar um evento, por meio de um marcador de controle, que permite mostrar textos 3D indicando os componentes principais do coração, nesta fase o aprendiz deve se familiarizar com os componentes do coração. Na fase de perguntas são apresentados, como alternativas, três modelos 3D em RA apresentados por turno e indicando componentes específicos do coração, sendo que em cada um destes modelos 3D só o componente do coração em questão está com cor, ficando o resto do modelo com transparência (ver Figura 5.3). O usuário deverá escolher o modelo com o componente do coração solicitado, e, após responder à pergunta, será possível ver o modelo 3D da resposta certa, e ativar eventos, por meio de um marcador de controle, que permitem tanto mostrar um texto 3D indicando o componente do coração, quanto mudar a cor do modelo para fazer com que ele fique sem nenhuma transparência. Nesta fase, igual a fase de treinamento, também será possível mudar o tamanho e rotação dos objetos através de botões.

Nota-se que, para este cenário, foram definidas perguntas usando a segunda abordagem de perguntas definida pelo *framework* (alternativas são modelos 3D em RA apresentados por turno). Finalmente, para este cenário, se define como objetivo de aprendizagem aprender os componentes principais do coração: aorta, artéria pulmonar, ventrículos etc.

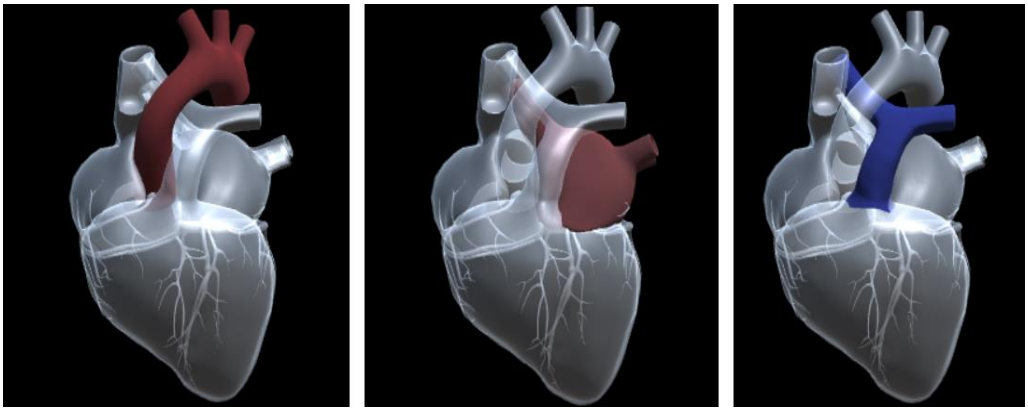

Figura 5.3: Modelos 3D indicando componentes do coração, e usados como alternativas para uma pergunta. O primeiro modelo indica a aorta (figura esquerda), o segundo o átrio esquerdo (figura centro), e o último a artéria pulmonar (figura direita).

2) Criar Recursos:

a) Criar Marcadores De RA: Através da página web da Vuforia foram criados dois marcadores usados nos dois cenários, estes marcadores foram criados a partir de imagens obtidas na internet, e podem ser vistos na Figura 5.4, o marcador da esquerda na figura foi usado como o marcador principal vinculado aos objetos 3D (sobre o qual aparecem os objetos 3D), já o marcador da direita foi usado como marcador de controle, permitindo ativar diferentes eventos tais como mostrar textos 3D com informação adicional, e mudar cor dos objetos 3D.

Figura 5.4: Marcadores usados nos dois cenários do aplicativo, e criados através da página web da Vuforia, a partir de imagens obtidas da internet. (PTC, n.d.), (ROBERTSON, 2011)

b) Criar Elementos Virtuais:

Cenário 1: Para este cenário foram criados quatro modelos 3D de figuras geométricas (ver Figura 5.5), além disso, quatro imagens dessas figuras usadas para fazer as perguntas (tais como a apresentada na Figura 5.1), e um total de dez imagens para as alternativas de resposta (tais como as apresentadas na Figura 5.2). Os modelos do cenário foram criados com o software Blender, enquanto que, para as imagens, foram utilizados os softwares Paint e Photoshop.

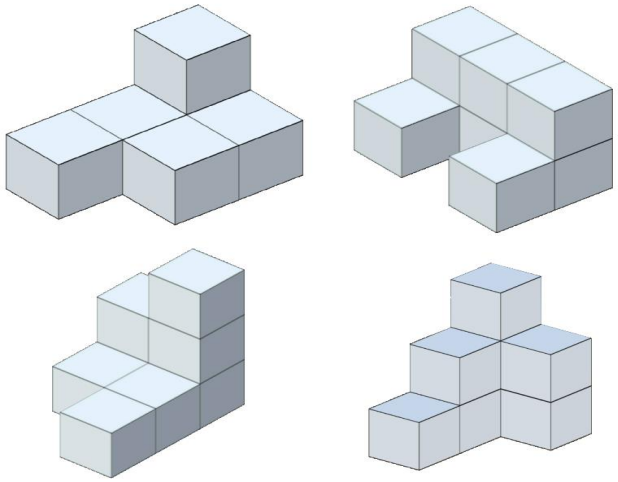

Figura 5.5: Modelos 3D de figuras geométricas criados com Blender.

Cenário 2: Para este cenário, foi usado unicamente o modelo 3D “Human_heart” de um coração humano, disponível na internet (ver Figura 5.6). Nota-se que, a partir desse modelo, foram modificados seus materiais que definem a cor, fazendo transparentes certos componentes do coração, para obter todos os modelos usados nas alternativas de resposta das perguntas, como os apresentados na Figura 5.3.

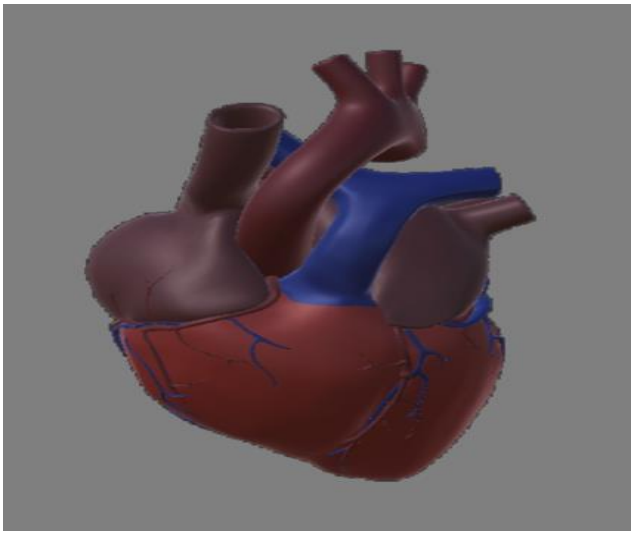

Figura 5.6: Modelo 3D do coração usado no Cenário 2.

3) **Configuração Da Vuforia:** Foram feitas as configurações necessárias no projeto do Unity, ficando a Vuforia pronta para ser utilizada. Para o estudo de caso foi usada a versão da Vuforia 7.1.31.

4) **Configuração Do Framework:** Foram feitas as configurações necessárias no projeto do Unity, ficando com o *framework* instalado e pronto para ser utilizado. Para o estudo de caso foi usada a versão do Unity 2018.1.0f2.

5) **Criação Dos Cenários:**
   a) **Importar Elementos Virtuais:** Foram importados no projeto, os elementos virtuais criados na fase 2.
   b) **Criação dos Cenários:** Com o *framework* instalado, e fazendo uso tanto dele quanto dos recursos obtidos das fases anteriores, foram criados os dois cenários com suas fases definidas. A seguir serão explicadas, resumidamente, as funcionalidades do *framework* usadas em cada cenário, assim como essas funcionalidades foram aplicadas através de suas classes.

      Cenário 1: Foram usadas as classes **`QuestionController1`**, **`ButtonsPack`**, **`ActivateInformation`** e **`ChangeElement`**, e, quando necessário, foram usadas também as versões dessas classes para fazer suas inicializações padrões (classes **`QuestionController1E`**, **`ButtonsPackE`**,

`ActivateInformationE`, e **`ChangeElementE`**) permitindo agilizar o desenvolvimento. O cenário com suas duas fases definidas pode ser visto nas Figuras 5.7, 5.8 e 5.9.

O uso das classes principais no cenário, e alguns detalhes importantes de configuração serão explicados a seguir.

- Classe **`QuestionController1`**: foi usada para criar as perguntas. Neste caso, foi configurada cada pergunta para usar alternativas de resposta através imagens 2D (ao invés de textos). Além disso, para as perguntas foi utilizada a opção *default* de ver reposta correta através de um botão.
- Classe **`ButtonsPack`**: foi usada para criar os botões de rotacionar e escalar os modelos 3D de figuras geométricas, na fase de exploração.
- Classe **`ActivateInformation`**: foi usada para criar o evento de mostrar um texto 3D indicando as faces da visão certa da figura, na fase de exploração (ver evento ativado na Figura 5.9 - direita). A classe foi configurada para usar textos 3D como a informação adicional.
- Classe **`ChangeElement`**: foi usada para criar o evento de mudar para cor vermelha as faces da visão certa da figura, na fase de exploração (ver evento ativado na Figura 5.9 - direita). A classe, portanto, foi configurada para trocar os Materiais originais das faces da visão certa da figura, por Materiais com cor vermelha.

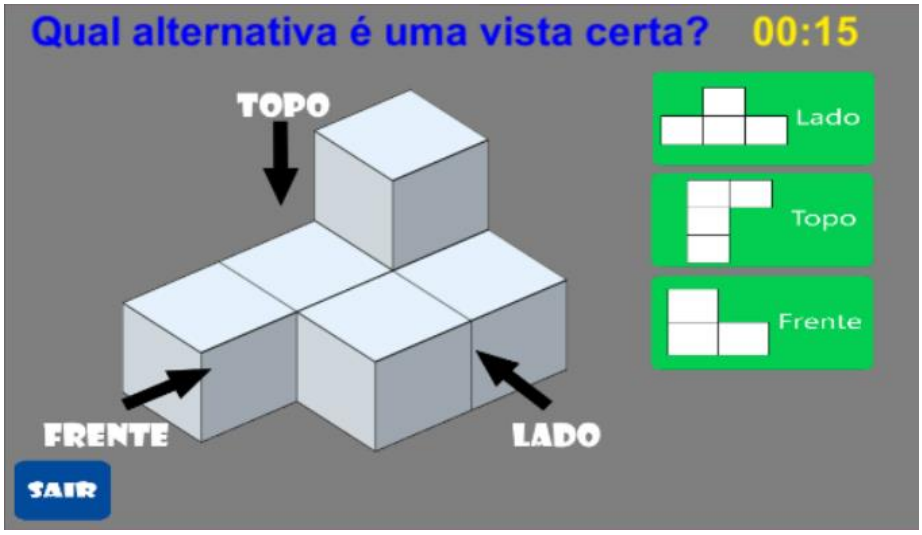

Figura 5.7: Fase inicial de perguntas sem RA para uma pergunta do Cenário1, enquanto está sendo respondida.

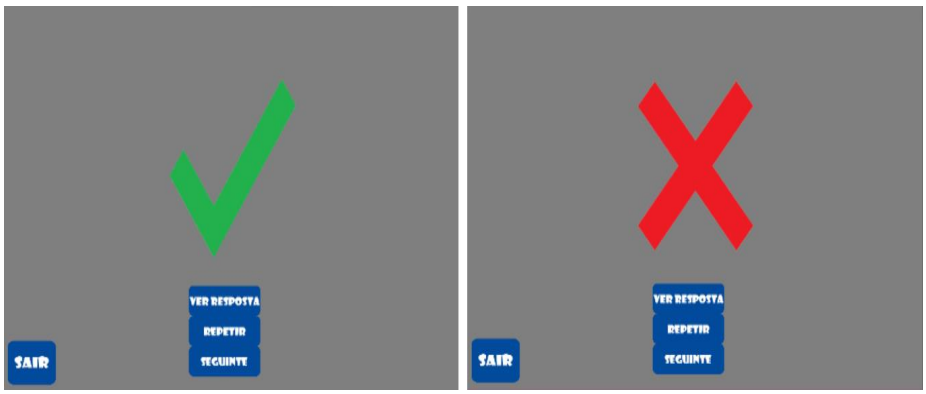

Figura 5.8: Fase inicial de perguntas sem RA para uma pergunta do Cenário1, após ser respondida corretamente (figura esquerda) e incorretamente (figura direita).

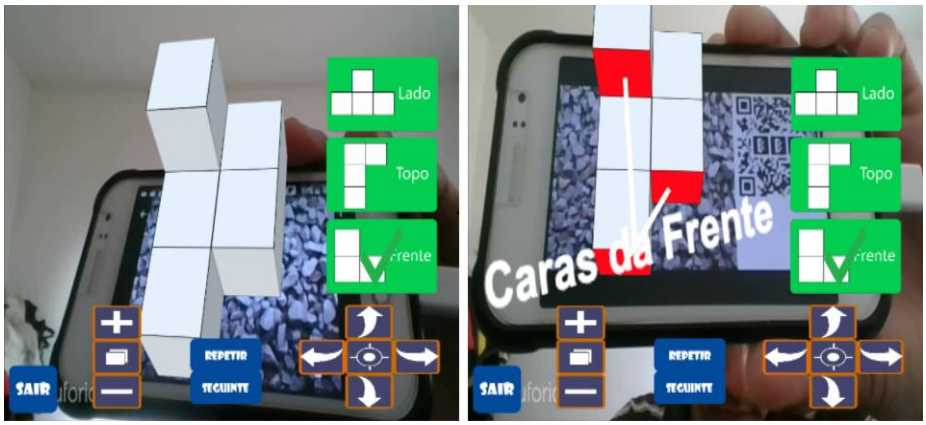

Figura 5.9: Fase final de exploração em RA para uma pergunta do Cenário 1, sem eventos ativados (figura esquerda) e com eventos ativados (figura direita).

Cenário 2: Foram usadas as classes **`QuestionController2`**, **`ButtonsPack`**, **`ActivateInformation`** e **`ChangeElement`**, e, quando necessário, foram usadas também as versões dessas classes para fazer suas inicializações padrões (classes **`QuestionController2E`**, **`ButtonsPackE`**, **`ActivateInformationE`**, e **`ChangeElementE`**) permitindo agilizar o desenvolvimento. O cenário com suas duas fases definidas pode ser visto nas Figuras 5.10, 5.11 e 5.12.

O uso das classes principais no cenário, e alguns detalhes importantes de configuração serão explicados a seguir.

- Classe **`QuestionController2`**: foi usada para criar as perguntas na fase de perguntas.
- Classe **`ButtonsPack`**: foi usada para criar os botões de rotacionar e escalar os modelos 3D do coração usados no cenário, tanto na fase de treinamento quanto na fase de perguntas.
- Classe **`ActivateInformation`**: foi usada para criar o evento de mostrar textos 3D indicando os componentes principais do coração, na fase de treinamento (ver evento ativado na Figura 5.10 - direita). Além disso, a classe também foi usada para criar o evento de mostrar um texto 3D indicando o componente do coração certo, quando está sendo visualizada a resposta correta na fase de perguntas (ver evento ativado na Figura 5.12 - direita). A classe foi configurada para usar textos 3D como a informação adicional.
- Classe **`ChangeElement`**: foi usada para criar o evento de mudar a cor do modelo 3D do coração para ficar sem nenhuma transparência, quando está sendo apresentada a resposta correta na fase perguntas (ver evento ativado na Figura 5.12 - direita). A classe, portanto, permite trocar os Materiais dos componentes do coração que possuem transparência por novos

Materiais com as cores adequadas para esses componentes, fazendo possível ao usuário, ver o coração todo sem transparências.

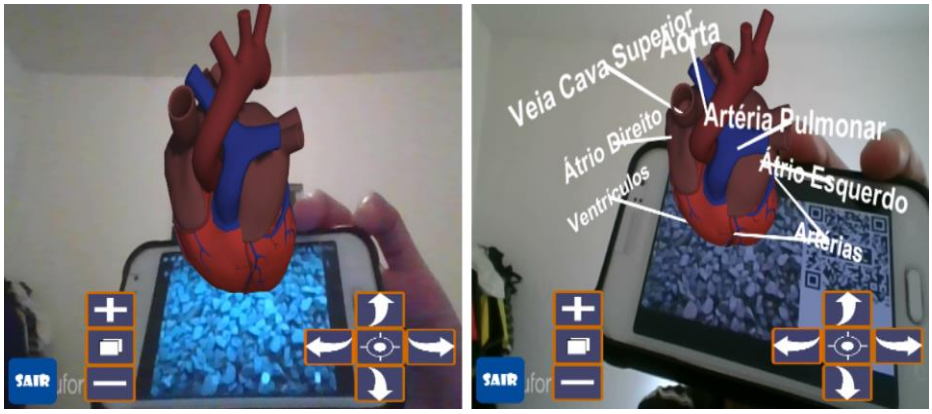

Figura 5.10: Fase de treinamento em RA do Cenário 2, sem o evento ativado (figura esquerda), e com o evento ativado (figura direita).

Figura 5.11: Fase de perguntas em RA do Cenário 2 enquanto está sendo respondida uma pergunta.

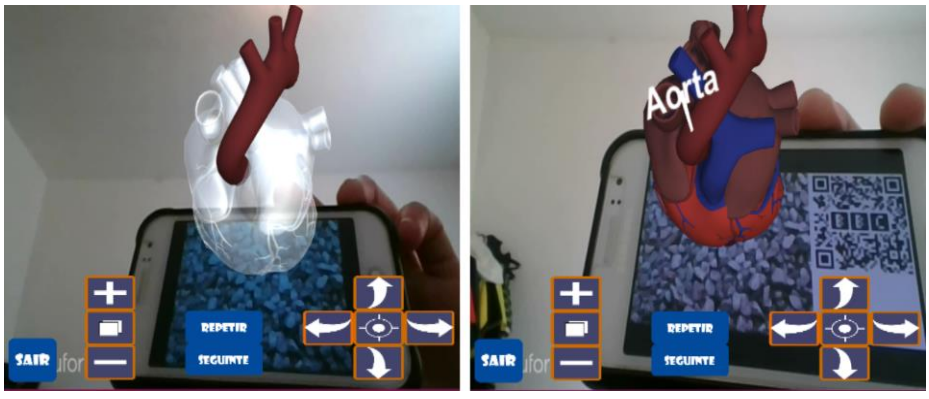

Figura 5.12: Fase de perguntas em RA do Cenário 2 enquanto está sendo apresentada a resposta certa, sem eventos ativados (figura esquerda) e com eventos ativados (figura direita).

Finalmente, como visto, os dois cenários foram feitos completamente usando o *framework*, portanto foi necessário unicamente utilizar as classes do *framework* disponíveis, e montar e customizar os cenários com os recursos necessários (modelos 3D, imagens, marcadores etc.). Porém, deve-se notar que, neste estudo de caso, para o aplicativo, codificação extra, ou seja, sem utilização do AR Educational Framework, foi necessária para definir os botões de sair e para os diferentes menus disponíveis. Este código adicional está presente na pasta chamada **`MenuScripts`** do projeto Unity do aplicativo do estudo de caso, disponível dentro da pasta: Aplicativo Estudo De Caso.

5.2 Arquitetura De Software Para O Aplicativo Educativo Criado Com O AR Educational Framework, No Estudo De Caso

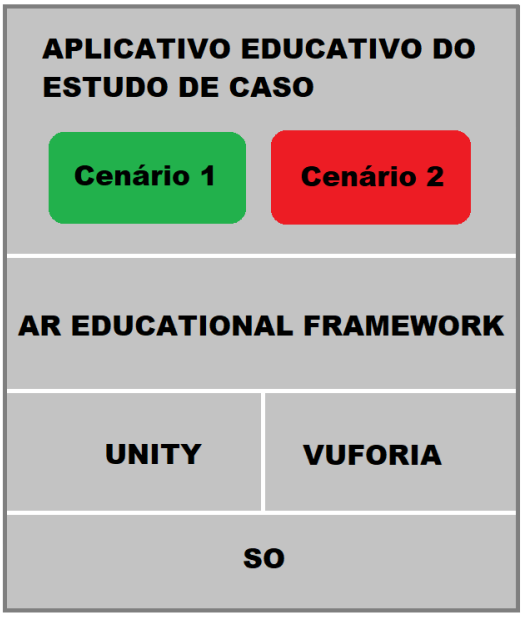

Figura 5.13: Arquitetura de Software para o aplicativo educativo criado com o AR Educational Framework, no estudo de caso.

Na Figura 5.13, pode ser vista a arquitetura de software sobre a qual foi criado o aplicativo educativo com RAM baseada em marcadores do estudo de caso

apresentado nesse capítulo. No caso particular deste aplicativo, foi utilizada a grande parte das funcionalidades suportadas pelo AR Educational Framework, fazendo com que quase todo o processo de codificação necessário para o aplicativo fosse substituído pelo processo de arrastar e soltar classes do AR Educational Framework, sendo necessária codificação extra (sem a utilização do AR Educational Framework) unicamente para os botões de sair, e os diferentes menus do aplicativo.

5.3 Diagrama De Casos De Uso Geral Do Aplicativo Criado Com O AR Educational Framework No Estudo De Caso

Nesta seção apresenta-se o diagrama de casos de uso geral do aplicativo criado com o AR Educational Framework, o qual pode ser visto na Figura 5.14.

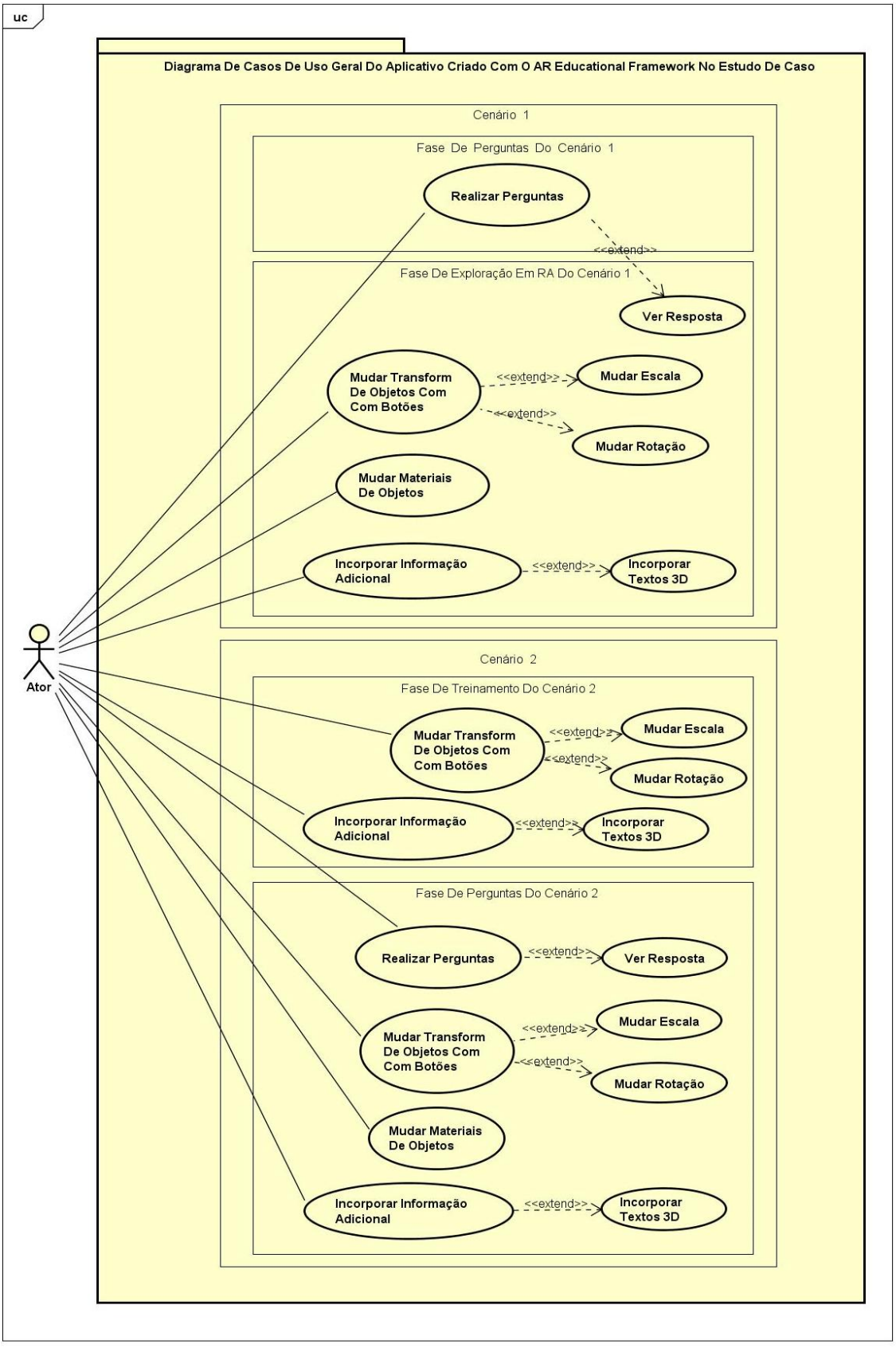

Figura 5.14: Diagrama de casos de uso geral do aplicativo criado com o AR Educational Framework.

O diagrama de casos de uso mostra os requisitos do aplicativo criado com o AR Educational Framework, os quais serão descritos a seguir. Neste caso, a fim de permitir uma melhor análise, o diagrama foi estruturado isolando cada um dos dois cenários disponíveis no aplicativo, com seus respectivos casos de uso.

Realizar Perguntas: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte de Perguntas”, descrito na Seção 4.1.3.2. As perguntas suportadas em cada cenário foram descritas na fase “Definir Os Cenários Do Aplicativo De RA Educacional”, na Seção 5.1, sendo usada a classe “**`QuestionController1`**” para criar as perguntas do cenário 1, e “**`QuestionController2`**” para criar as perguntas do cenário 2.

Mudar Transform De Objetos Com Botões: Este requerimento é suportado pelo *framework* através da funcionalidade “Suportar a modificação da transform de objetos através de botões da UI (User Interface)”, descrito na Seção 4.1.3.1. Para cada um dos dois cenários, esta funcionalidade é fornecida através da “Classe **`ButtonsPack`**”, tal como descrito na fase “Criação Dos Cenários”, na Seção 5.1.

Mudar Materiais De Objetos: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte a modificação dos materiais de objetos através de marcadores de controle”, descrito na Seção 4.1.3.2. Para cada um dos dois cenários, esta funcionalidade é fornecida através da “Classe **`ChangeElement`**”, tal como descrito na fase “Criação Dos Cenários”, na Seção 5.1.

Incorporar Informação Adicional: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle”, descrito na Seção 4.1.3.2. Para cada um dos dois cenários, esta funcionalidade é fornecida através da “Classe **`ActivateInformation`**”, tal como descrito na fase “Criação Dos Cenários”, na Seção 5.1.

5.4 Considerações Finais

Neste capítulo, foi apresentado o desenvolvimento de um aplicativo educacional usando o *framework* proposto, e seguindo as *guidelines* da metodologia proposta, sendo explicado, de forma geral, como foram usadas as funcionalidades do *framework* para criar o aplicativo. Nota-se que a escolha dos dois cenários do aplicativo, cada um deles em áreas diferentes da educação (cenário 1 na área de Geometria, e cenário 2 na área de Biologia), foi feita para permitir avaliar o uso do *framework* e suas funcionalidades em áreas diferentes (determinar se o *framework* é de cunho geral na educação).

Através deste estudo de caso, pode-se levantar evidências de que é possível para um desenvolvedor experiente no Unity e Vuforia (no caso o autor), criar aplicações educacionais de RAM, facilmente, usando o *framework*, uma vez que foi possível realizar com sucesso o desenvolvimento do aplicativo do estudo de caso. É notável a facilidade no desenvolvimento com o *framework,* uma vez que a maior parte do aplicativo foi feita arrastando e soltando classes do *framework* (sem precisar de programação) e montando e customizando cenários com os recursos necessários (modelos 3D, imagens, etc.), assim o único código adicional para o aplicativo, e externo ao *framework*, foi o necessário para programar os botões de sair e de navegação dos diferentes menus disponíveis, o qual ainda pode-se considerar como o de menor peso, complexidade e relevância de todo o desenvolvimento. Assim, pode-se afirmar que o *framework* trouxe um significativo ganho tanto de tempo quanto de simplicidade, uma vez que o tempo e complexidade de quase a totalidade de programação requerida foi abstraído pelo processo de arrastar e soltar classes.

Quanto à flexibilidade, o *framework* mostrou ser altamente customizável em relação às funcionalidades fornecidas, permitindo criar perguntas de dois tipos bem diferentes (de acordo com as necessidades de cada cenário), e customizar cada uma delas (informação necessária, elementos 3D e 2D, tempo de resposta etc.), assim como os eventos de mostrar informação adicional, mudar cor de objetos, e mudar escala e rotação. Por outro lado, o *framework* também permitiu se integrar

com sucesso ao código adicional externo a ele, permitindo, neste caso, que através de menus externos ao *framework* seja feita a navegação entre as perguntas.

O estudo de caso também demostrou que as funcionalidades fornecidas pelo *framework* podem ser úteis para aplicativos de diferentes áreas da educação, uma vez que cada cenário do aplicativo está inserido dentro de uma área totalmente diferente (Geometria e Biologia, respetivamente), sendo utilizado, para sua criação, o mesmo conjunto de funcionalidades do *framework*.

Finalmente, apesar de o estudo de caso indicar que o *framework* cumpre com os objetivos pelos quais foi proposto, o próximo capítulo tem como objetivo apresentar a avaliação deste trabalho através de experimentos usando o *framework* com um grupo de estudantes de cursos de computação, para o desenvolvimento de cenários educacionais de RA, o que corresponde a um contexto que permitirá uma avaliação mais adequada e mais uma forma de validação do *framework.*

Capítulo 6

AVALIAÇÃO: EXPERIMENTO

No capítulo anterior foi apresentando o estudo de caso desenvolvido com o objetivo de avaliar a utilização do *framework* no desenvolvimento de um aplicativo educacional de RAM a partir da experiência pessoal do autor desse trabalho. No entanto, é necessário também uma avaliação complementar através de experimentos de utilização do *framework* por outros desenvolvedores. No caso, o público dos experimentos consistiu em um grupo de estudantes de cursos de computação com nível iniciante ou intermediário no Unity e Vuforia. Visto que a criação de um aplicativo completo de RA demandaria bastante tempo, os experimentos foram limitados à criação de cenários educativos simples de RA utilizando apenas algumas das funcionalidades providas pelo *framework*, e reutilizando vários dos modelos 3D do aplicativo desenvolvido no estudo de caso do Capítulo 5.

Assim, foram realizados experimentos visando uma adequada avaliação do *framework* proposto, para tentar definir se através da utilização do *framework* foi atingido o objetivo deste projeto (apoiar desenvolvedores na criação de aplicativos educativos com RAM baseada em marcadores, de cunho geral na educação), e principalmente avaliar a usabilidade e dificuldade de uso do *framework* por outros desenvolvedores. Em geral, os problemas a serem solucionados através do *framework* podem ser apresentados através de questões, que são as seguintes:

1. O AR Educational Framework permite o desenvolvimento de aplicações educativas de Realidade Aumentada com facilidade?

2. O AR Educational Framework é útil tanto para desenvolvedores experientes, quanto iniciantes em Unity e Vuforia?
3. O AR Educational Framework fornece funcionalidades customizáveis de acordo às necessidades do usuário?
4. O AR Educational Framework é de cunho geral na educação, ou seja, útil para qualquer área na educação?

6.1 Experimentos

Visando a validação do *framework* proposto, foram realizados dois experimentos utilizando algumas funcionalidades do *framework*. Foi necessária uma primeira sessão em formato treinamento para conhecer conceitos básicos do Unity e da Vuforia, e, após isso, foram aplicados os dois experimentos usando o *framework*. Cada um destes experimentos consistiu em um treinamento inicial de uma funcionalidade do *framework* e na aplicação de um exercício proposto usando a funcionalidade estudada. Finalmente, após os experimentos, foi realizada a coleta de dados aplicando um questionário, e solicitando informações sobre utilização do *framework,* assim como sugestões. A sessão de treinamento do Unity e Vuforia, e os dois experimentos com a coleta de dados (através do questionário), foram realizados em 3 sessões, em 3 dias distintos, e são detalhados nas próximas seções deste capítulo.

Para a participação nos experimentos não foi definido como requisito ter conhecimentos de programação, sendo unicamente recomendável que os participantes tivessem conhecimentos básicos na interface do Unity, e noções da Vuforia. Deve-se notar que devido à simplicidade do *framework*, ainda sem os participantes terem tal conhecimento recomendado (de Unity e Vuforia), foi previsto que o treinamento inicial no Unity e Vuforia seria suficiente para usar o *framework* corretamente. Os experimentos contaram com a inscrição de seis estudantes de cursos de computação.

A seguir são detalhados o treinamento do Unity/Vuforia, os dois experimentos realizados, e, finalmente, a coleta de dados.

6.1.1 Treinamento Unity/Vuforia

O objetivo deste treinamento consistiu em dar aos participantes os conhecimentos básicos do Unity e a Vuforia que seriam necessários para usar o *framework*, e, além disso, realizar a instalação do *framework* com todas as configurações necessárias, com o objetivo de que nas sessões dos experimentos o foco ficaria apenas na utilização das funcionalidades do *framework*. Deve-se notar que tanto a instalação do Unity (incluindo a Vuforia) quanto a instalação de câmeras, foram realizadas previamente nos computadores utilizados, sendo testados previamente a fim de evitar qualquer tipo de inconveniente no treinamento, e posteriores experimentos.

Embora o treinamento tenha sido majoritariamente expositivo (alguns slides também foram apresentados com conceitos de RA e o *framework*), foi pedido aos participantes que realizassem tarefas simples livremente para se familiarizar com as ferramentas utilizadas. O treinamento teve um tempo aproximado de 1 hora e 40 minutos (o esclarecimento de dúvidas está computado nesse total), e incluiu os seguintes aspectos:

- Apresentação de conceitos básicos do Unity: Foram apresentados os conceitos fundamentais do Unity, as diferentes visões do Unity, navegação pela visão da cena do Unity, mudar *transform* de objetos, posicionar objetos juntos etc. Este ponto teve como principal foco os participantes que tinham pouco ou nenhum conhecimento do Unity.
- Instalação do Framework: Foi realizada a instalação do *framework*, incluindo as configurações necessárias da Vuforia.
- Apresentação de conceitos básicos da Vuforia: Foram apresentados os conceitos fundamentais da Vuforia para permitir mostrar objetos em RA, mudar as imagens dos marcadores, seu tamanho, etc.

Nota-se que os conceitos apresentados foram simples e de nível iniciante nas ferramentas (em geral incluindo configurações, criação e manipulação básica de

elementos 2D e 3D em Unity, e com RA usando Vuforia). Aspectos avançados, assim como de programação, não foram abordados no treinamento, uma vez que não seriam necessários nos experimentos.

6.1.2 Experimento 1

O objetivo do experimento foi utilizar o *framework* usando a funcionalidade de "Suporte de Perguntas", descrita na Seção 4.1.3.2, através da classe **`QuestionController2`**. Para isso, o experimento 1 (semelhante ao experimento 2) esteve dividido em duas partes, que são descritas a seguir:

6.1.2.1 Treinamento

O treinamento teve como objetivo capacitar aos participantes no uso da funcionalidade utilizada (no caso, através do uso da classe **`QuestionController2`** do *framework*), para isso foi necessária a explicação dos conceitos necessários, realização de exemplos com a funcionalidade, sendo fornecida também documentação de apoio consistindo em um tutorial do uso da funcionalidade incluindo imagens de captura de tela (ver "Tutorial Classe **`QuestionController2`**" em Apêndice I). Nota-se que este tutorial foi uma adaptação, com pequenas mudanças do "Tutorial Classe **`QuestionController2`**" do Apêndice D, sendo mais adequado para os fins do experimento. Finalmente, o tempo do treinamento foi aproximadamente de 50 minutos.

6.1.2.2 Atividade

Nesta atividade os participantes deviam realizar individualmente um exercício proposto usando a funcionalidade estudada, podendo ser auxiliados pela documentação do tutorial fornecido no treinamento. Especificamente, a atividade foi criar um cenário de RA com uma pergunta sobre o coração humano, que consistia em identificar qual, entre três modelos 3D usados como as alternativas de resposta (e apresentados por turno), indicava o componente do coração chamado ventrículos (ver cenário Figuras 6.1, 6.2 e 6.3), para isso, foi fornecido um documento com as operações requeridas que deviam ser aplicadas fazendo uso do *framework* (ver "EXPERIMENTO 1 AR EDUCATIONAL FRAMEWORK" em Apêndice J), sendo

possível ter apoio da documentação e tutoriais disponíveis. O tempo para a atividade foi aproximadamente de 10 minutos.

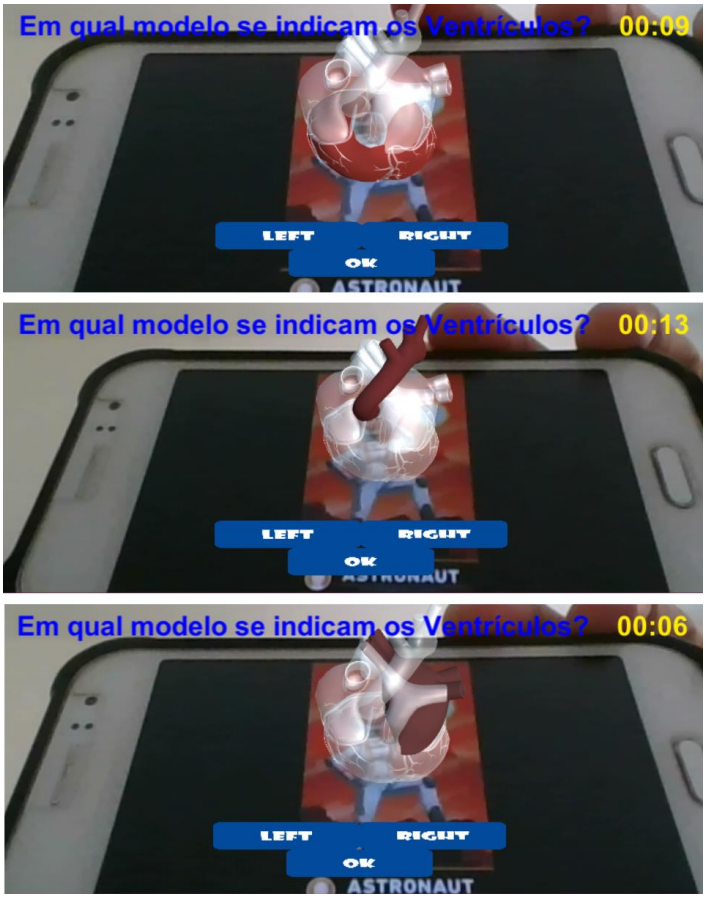

Figura 6.1: Cenário final enquanto está sendo respondia a pergunta, com os três modelos alternativas que podem ser explorados um por um (com os botões "LEFT" e "RIGHT") e escolhidos como resposta com "OK" (figura acima modelo indicando Ventrículos, figura no centro modelo indicando a Aorta, e figura abaixo modelo indicando o Átrio Esquerdo).

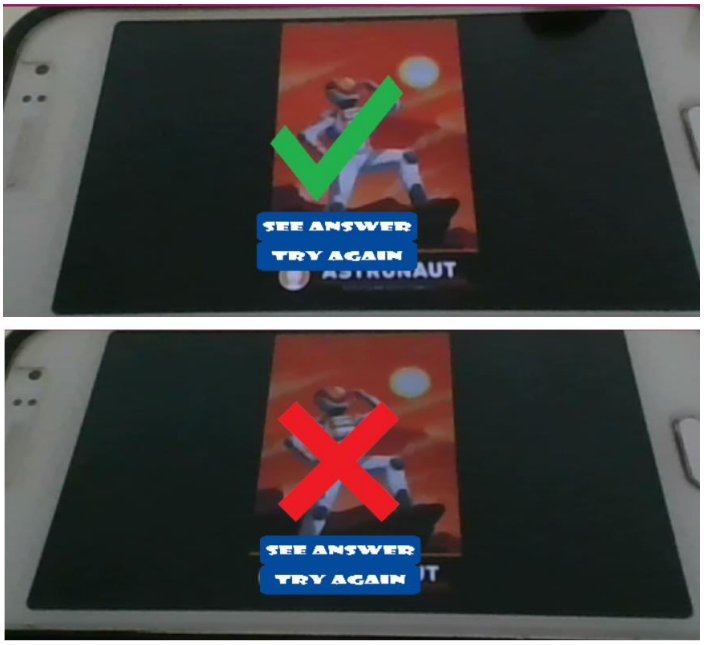

Figura 6.2: Cenário final após responder a pergunta pressionando botão "OK", e enquanto está sendo visto o *feedback* de resposta correta e errada respetivamente.

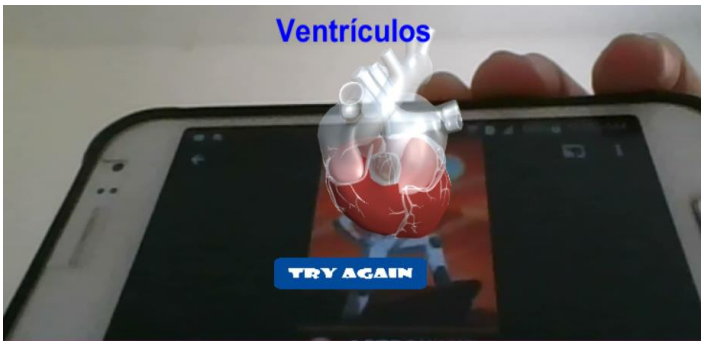

Figura 6.3: Cenário final após ser pressionado o botão "See Answer", e mostrando unicamente o modelo resposta da pergunta (modelo indicando Ventrículos).

Nota-se que o marcador usado para o experimento foi um dos marcadores *default* fornecido pela Vuforia, e apresentado na Figura 6.4.

Figura 6.4: Marcador *default* da Vuforia, usado no experimento 1. (PTC, 2017)

6.1.3 Experimento 2

O objetivo do experimento foi utilizar o *framework* usando a funcionalidade de "Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle", descrita na Seção 4.1.3.2, através da classe **`ActivateInformation`**.

6.1.3.1 Treinamento

O treinamento teve como objetivo capacitar aos participantes no uso da funcionalidade utilizada (no caso, através do uso da classe **`ActivateInformation`** do *framework*), fornecendo também um tutorial de apoio (ver "Tutorial Classe **`ActivateInformation`**" em Apêndice I), e, em geral, seguindo os mesmos lineamentos propostos no treinamento do experimento 1. Nota-se que este tutorial foi uma adaptação, com pequenas mudanças, do "Tutorial Classe **`ActivateInformation`**" do Apêndice D, sendo mais adequado para os fins do experimento. O tempo do treinamento foi aproximadamente de 50 minutos.

6.1.3.2 Atividade

Nesta atividade, semelhantemente à atividade do experimento 1, os participantes deviam realizar individualmente um exercício proposto usando a funcionalidade estudada, podendo ser auxiliados pelo tutorial fornecido no treinamento. Especificamente, a atividade consistiu em criar um cenário de RA no qual um modelo 3D de um coração humano podia ser munido de informação

adicional (textos 3D indicando componentes específicos do coração), quando um marcador adicional (marcador de controle) estiver junto ao marcador original, ou seja, o marcador adicional funcionaria como o *trigger* do evento de ativar informação (ver cenário nas Figuras 6.5, e 6.6), para isso, foi fornecido um documento com as operações requeridas que deviam ser aplicadas fazendo uso do *framework* (ver "EXPERIMENTO 2 AR EDUCATIONAL FRAMEWORK" em Apêndice J), sendo possível ter apoio da documentação e tutoriais disponíveis. O tempo para a atividade foi, aproximadamente, de 15 minutos.

Figura 6.5: Cenário final quando só o marcador principal é visível pela câmera, e mostrando só o modelo do coração.

Figura 6.6: Cenário final quando o marcador adicional (ou de controle) é colocado junto ao marcador principal (ambos os marcadores são visíveis), permitindo ativar a informação adicional.

Nota-se que os marcadores usados para o experimento foram marcadores *default* fornecidos pela Vuforia, o marcador principal foi o mesmo do experimento 1 (Figura 6.4), e o marcador de controle é apresentado na Figura 6.7.

Figura 6.7: Marcador *default* da Vuforia, usado no experimento 2. (PTC, 2017)

6.1.4 Coleta de Dados

Para coletar os dados, foi realizado um questionário (ver Apêndice H), o qual esteve principalmente focado em medir a usabilidade do *framework*. De maneira geral, foram realizadas questões para os seguintes aspectos:

I. Realizar o levantamento do conhecimento inicial dos participantes nas ferramentas e tecnologias necessárias (Unity, Vuforia e RA).
II. Determinar qual foi a relevância dos treinamentos e ou material de apoio para realizar as atividades propostas com o *framework*.
III. Tentar determinar se o *framework* é de fácil uso.
IV. Determinar se as funcionalidades do *framework* resultam altamente customizáveis para o usuário.
V. Conhecer a visão dos participantes para o uso do *framework* no desenvolvimento de aplicativos completos de RA.

O questionário incluiu também três perguntas abertas não obrigatórias em que os participantes do experimento poderiam incluir comentários e opiniões que achassem necessários.

6.1.4.1 Questionário

Com os dados obtidos do questionário (ver questionário em Apêndice H), e apresentados através de gráficos, é realizada uma análise da percepção dos participantes para cada questão.

No levantamento do conhecimento inicial dos participantes nas ferramentas e tecnologias necessárias, a partir dos gráficos da Figura 6.8, pode-se concluir que o conhecimento dos participantes no Unity, Vuforia e Realidade Aumentada, pode ser considerado, no geral, básico, pois a maioria dos participantes mantive suas respostas entre “Nenhum” a “Básico” para todas as questões, tendo só um participante com conhecimento intermediário em Unity e outro em Realidade Aumentada. É importante notar que o Unity foi a ferramenta com maior conhecimento prévio por parte dos participantes, 50% deles indicando conhecimento básico, já no caso da Vuforia e Realidade Aumentada, a maioria dos participantes indicaram ter “Nenhum” conhecimento.

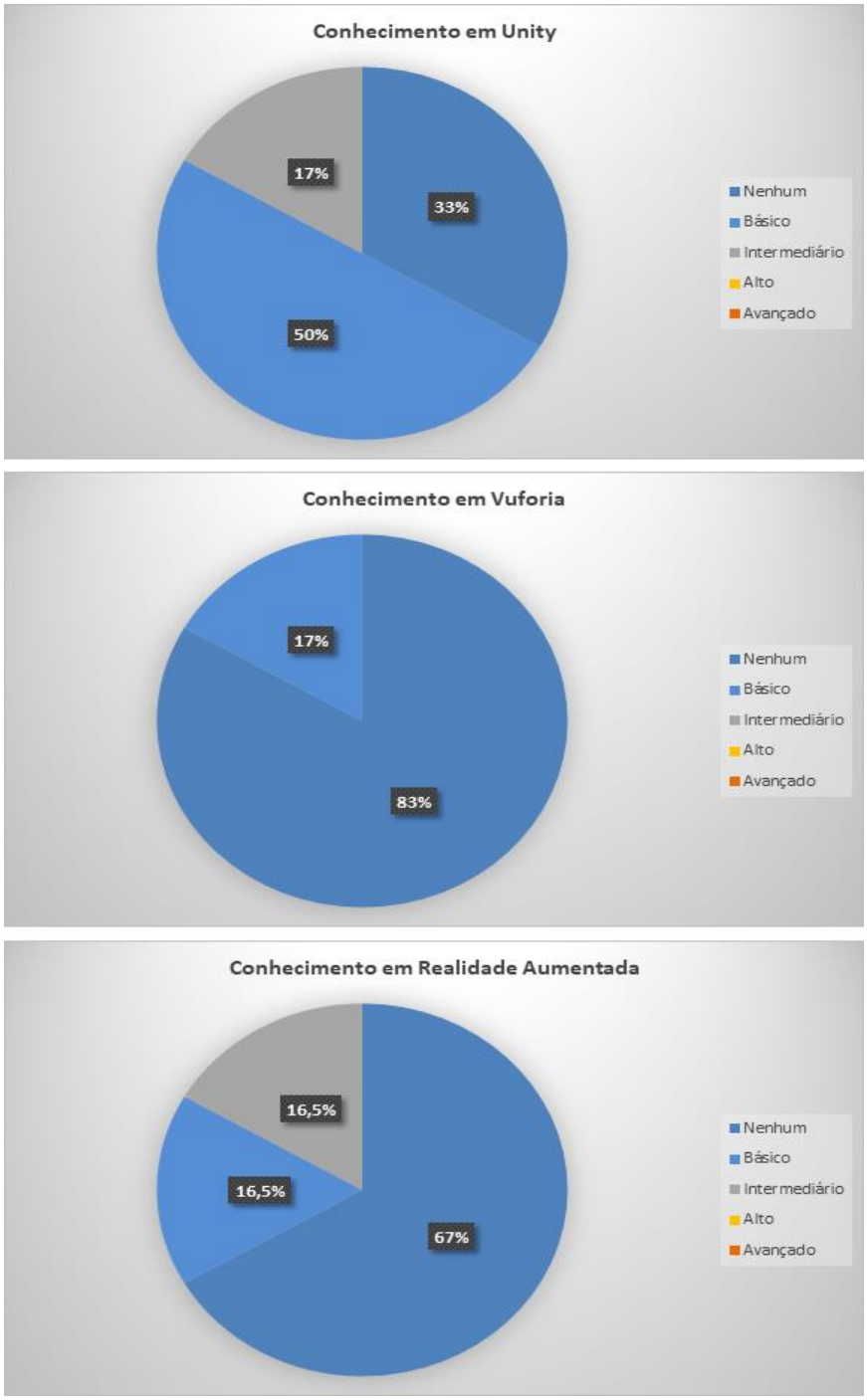

Figura 6.8: Gráficos com porcentagens referentes às questões do levantamento de conhecimento dos participantes nas ferramentas e tecnologias necessárias.

Nas questões visando determinar a relevância dos treinamentos e o material de apoio (tutoriais) para a realização das atividades nos experimentos, a partir dos gráficos da Figura 6.9, pode-se concluir que tanto o treinamento quanto o material de apoio foram essenciais para o desenvolvimento.

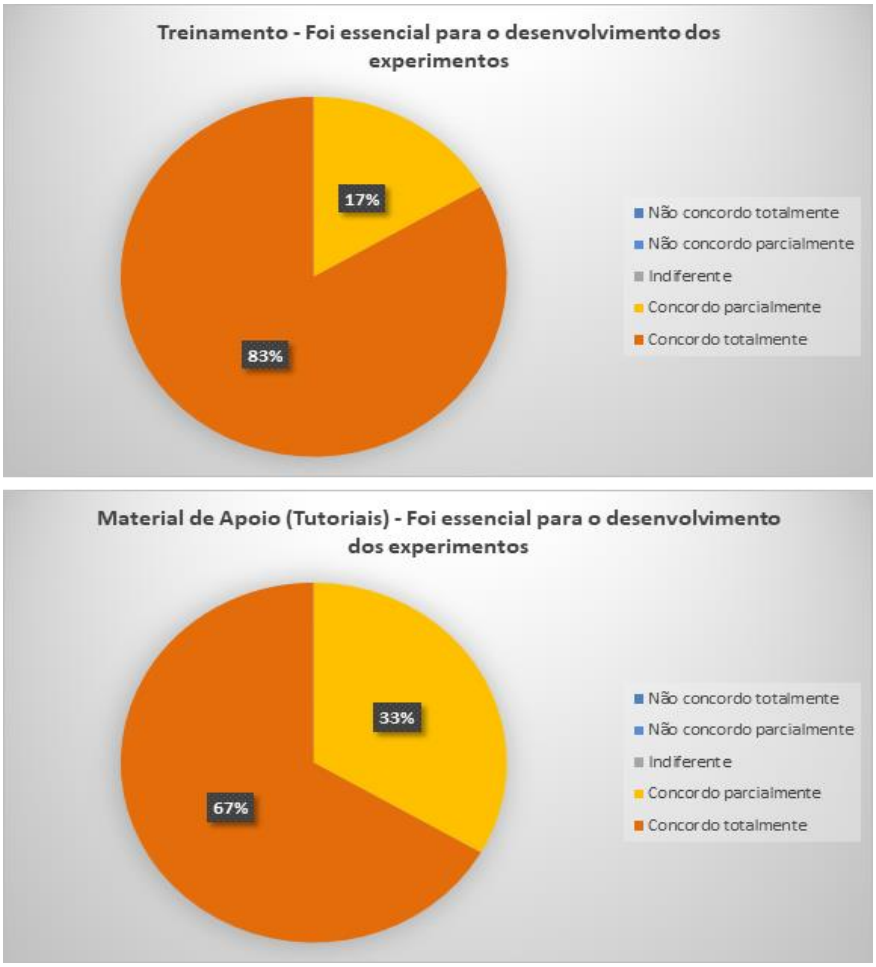

Figura 6.9: Gráficos com porcentagens referentes às questões para determinar relevância dos treinamentos e o material de apoio para a realização das atividades nos experimentos.

Por outro lado, a partir dos gráficos da Figura 6.10, não é possível concluir que o treinamento por si só, nem o material de apoio por si só, seriam suficientes para realizar os experimentos, uma vez que os resultados apresentaram opiniões contraditórias, em geral quantidades aproximadas de pessoas concordaram e discordaram das afirmações.

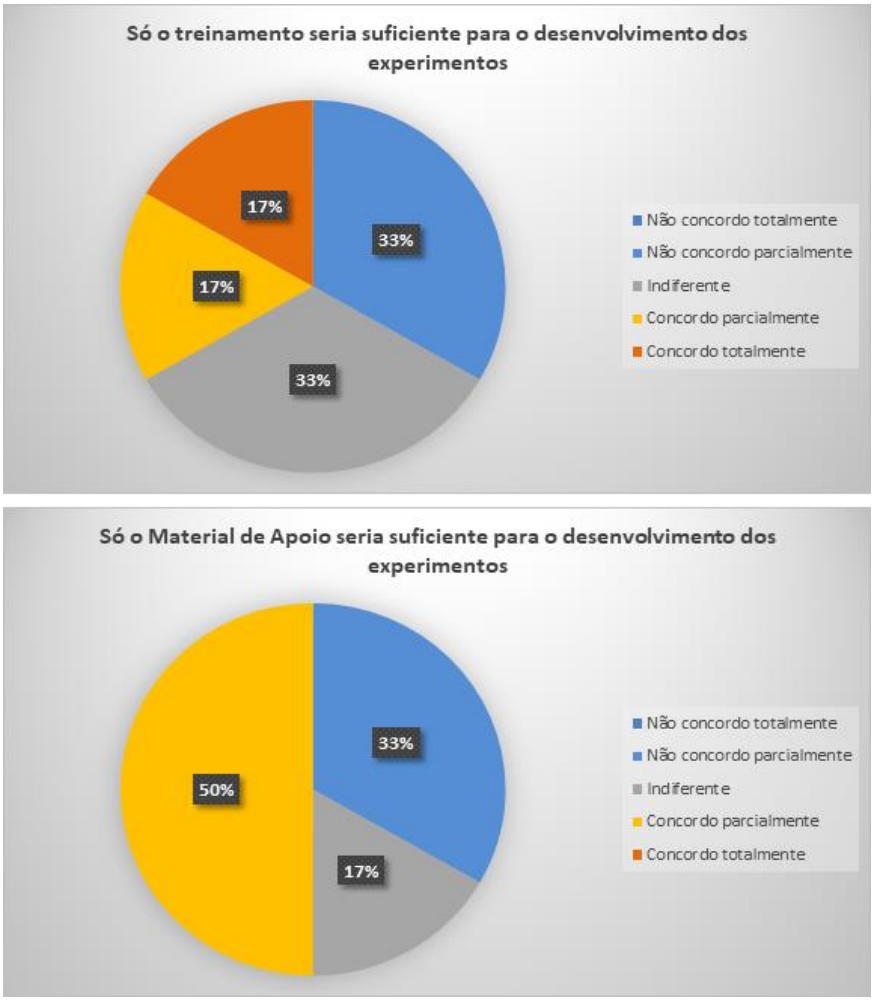

Figura 6.10: Gráficos com porcentagens referentes às questões para determinar relevância dos treinamentos e o material de apoio para a realização das atividades nos experimentos.

Quanto às perguntas focadas em determinar a facilidade ou complexidade do uso do *framework*, os resultados obtidos foram bastante claros e positivos para a pesquisa, resultando em uma percepção de simplicidade no uso. A análise dessas perguntas é apresentada a seguir.

Inicialmente, para as questões “Realizei com sucesso as atividades previstas no experimento 1”, e “Realizei com sucesso as atividades previstas no experimento 2”, os gráficos da Figura 6.11 mostram que, em geral, as atividades propostas foram realizadas sem nenhum tipo de inconveniente, e lembrando que o conhecimento inicial dos participantes nas ferramentas usadas foi básico. Dessa forma, é possível observar indícios de que o *framework* é de fácil uso, e não requer conhecimentos avançados.

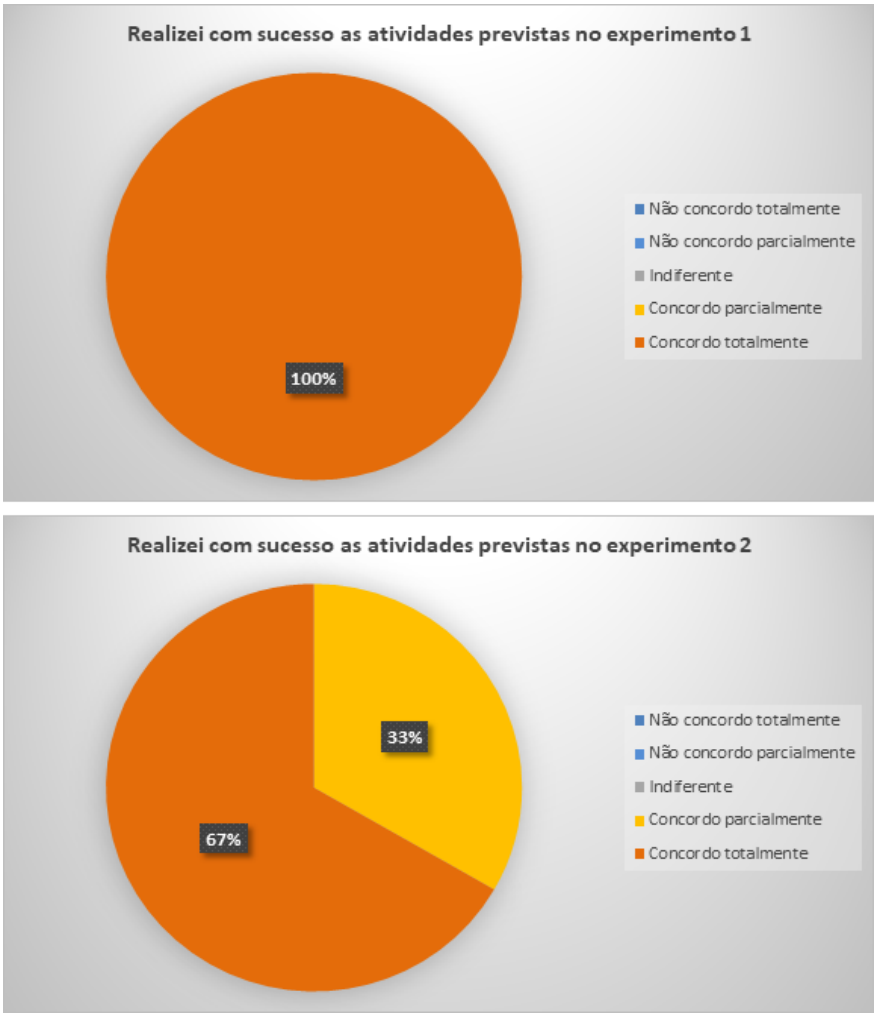

Figura 6.11: Gráficos com porcentagens referentes a questões para determinar se as atividades nos experimentos foram realizadas com sucesso pelos participantes.

Nas perguntas “É necessário que o nível de conhecimento em Unity seja avançado para o desenvolvimento dos experimentos”, “É necessário que o nível de conhecimento em Vuforia seja avançado para o desenvolvimento dos experimentos”, e “É necessário que o nível de conhecimento em Realidade Aumentada seja avançado para o desenvolvimento dos experimentos”, a Figura 6.12 mostra 3 gráficos com os resultados respectivos, dos quais pode-se concluir que, em geral, o nível de conhecimento necessário sobre Unity, Vuforia e Realidade Aumentada, para o desenvolvimento dos experimentos, não foi considerado avançado, pois, para as três questões, a maioria dos participantes mantiveram suas respostas entre “Não concordo totalmente” e “Não concordo parcialmente”.

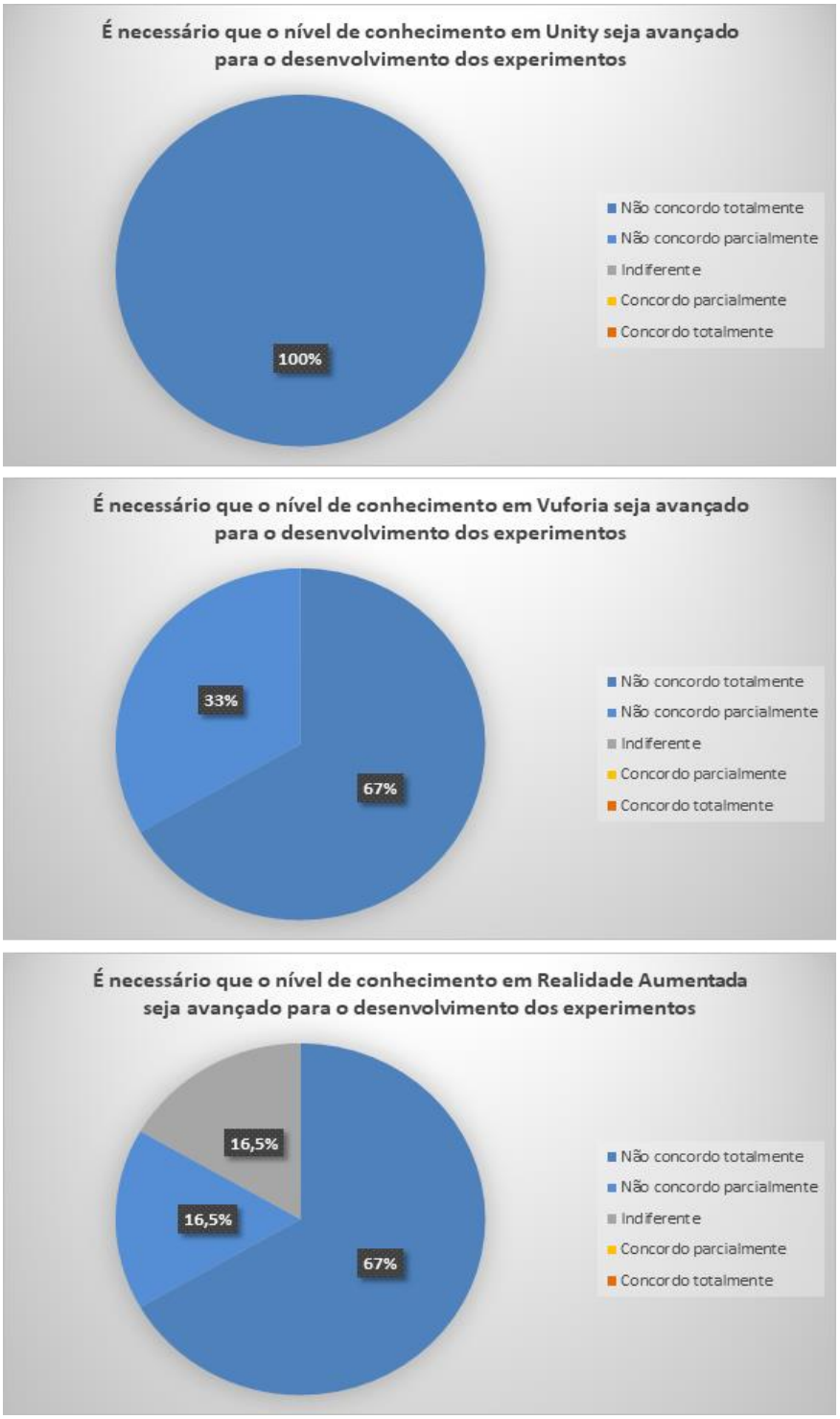

Figura 6.12: Gráficos com porcentagens referentes a questões para determinar o nível de conhecimento necessário nas ferramentas Unity, Vuforia, e na RA, para usar o *framework* nos experimentos realizados.

Na questão realizada “De forma geral, a utilização do Framework pode ser considerada de nível avançado”, como pode ser visto no gráfico da Figura 6.13, somando as escalas “Não concordo totalmente” e “Não concordo parcialmente”,

alcança-se um índice do 67% dos participantes dizendo que o uso do *framework* proposto não é considerado avançado, além disso, nota-se que nenhum dos estudantes concordou com a afirmação da questão, e só 2 deles (correspondentes ao 33%) indicaram “Indiferente”, obtendo-se, através desta questão, uma percepção simples (ao invés de avançada) para o uso do *framework*, e que ainda permite reafirmar os resultados das três questões anteriores.

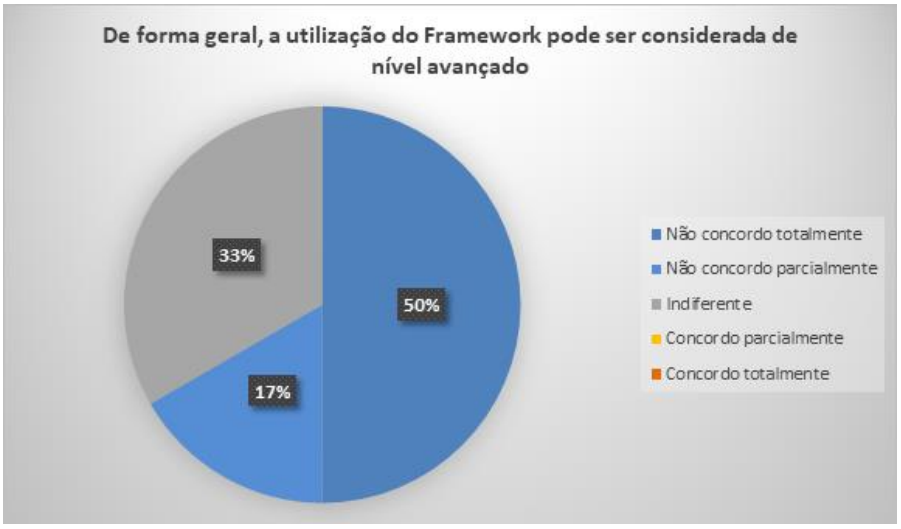

Figura 6.13: Gráficos com porcentagens referentes à questão “De forma geral, a utilização do Framework pode ser considerada de nível avançado” do questionário.

Na pergunta realizada “O Framework permite criar cenários educativos simples de Realidade Aumentada com facilidade”, o gráfico da Figura 6.14 mostra resultados totalmente positivos e ainda mais esclarecedores que confirmam a percepção de simplicidade no uso do *framework*, visto que a totalidade dos participantes mantiveram suas respostas entre “Concordo totalmente” e “Concordo parcialmente”, sendo 67% correspondente a “Concordo totalmente”, e 33% restantes correspondente a “Concordo parcialmente”.

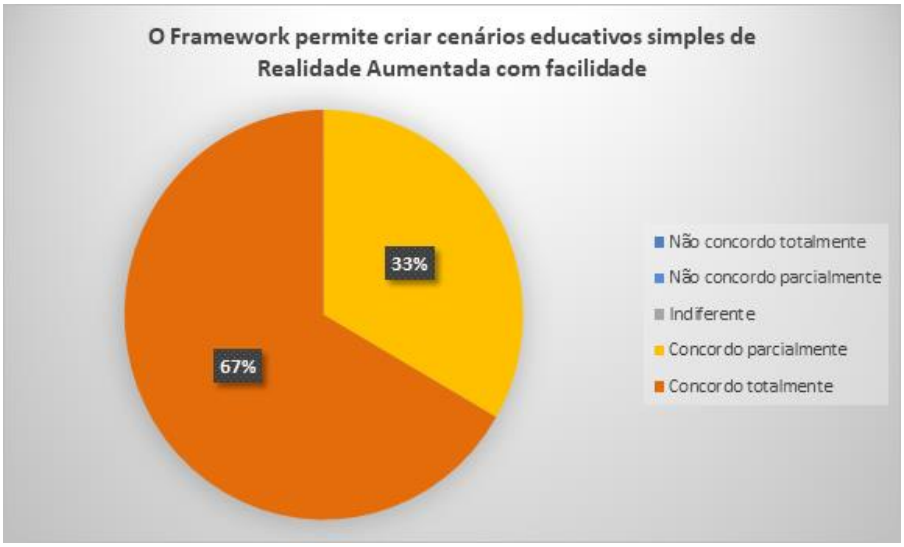

Figura 6.14: Gráficos com porcentagens referentes à questão "O Framework permite criar cenários educativos simples de Realidade Aumentada com facilidade" do questionário.

Assim, uma vez analisados os resultados das perguntas dirigidas a determinar a facilidade do uso do *framework*, em geral, pode-se afirmar que o *framework* mostrou ser fácil de usar, ainda para um público iniciante nas ferramentas e tecnologias requeridas para seu uso.

A questão para determinar o nível de customização do *framework*, que foi a seguinte: "O Framework fornece um pacote de funcionalidades altamente customizáveis de acordo às necessidades do usuário", teve resultados também positivos. Dessa forma, pode-se afirmar que, na percepção dos participantes, o *framework* apresenta alto nível de customização, pois, como mostrado no gráfico da Figura 6.15, a totalidade dos participantes concordaram parcial ou totalmente com a questão.

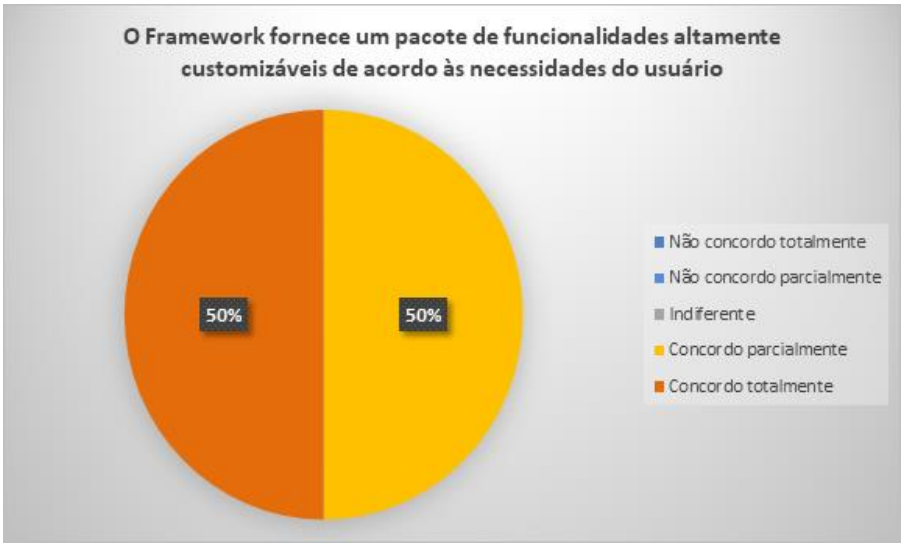

Figura 6.15: Gráficos com porcentagens referentes à questão "O Framework fornece um pacote de funcionalidades altamente customizáveis de acordo às necessidades do usuário" do questionário.

Finalmente, quanto às questões para conhecer a visão dos participantes para o uso do *framework* no desenvolvimento de aplicativos completos de RA, os resultados apresentam uma visão muito positiva do *framework*, pois, como pode ser visto nos gráficos da Figura 6.16, a maior parte dos participantes mantiveram suas respostas entre "Concordo parcialmente" e "Concordo totalmente", para todas as questões, sugerindo que o *framework* se apresenta como uma ferramenta capaz de permitir com sucesso o desenvolvimento de aplicativos educativos de RA, ainda com facilidade. A última das perguntas, na qual todos os participantes concordaram totalmente em que usariam o *framework*, serve para reafirmar ainda mais a percepção positiva que em geral obteve o *framework* para a criação de ambientes de RA. Deve-se notar que para estas questões, os resultados representam uma visão parcial dos participantes, lembrando que as atividades propostas foram para criar cenários simples de RA (não um aplicativo completo), porém este resultado pode ser tomado como confirmação das evidências obtidas no estudo de caso, onde foi desenvolvido (pelo próprio autor) um aplicativo completo de RA com sucesso.

Figura 6.16: Gráficos com porcentagens referentes a questões para conhecer a visão dos participantes para o uso do *framework* no desenvolvimento de aplicativos completos de RA.

6.2 Considerações Finais

Como poderá ser visto ao longo deste capítulo, através da análise das respostas ao questionário, podem ser apresentadas respostas para as perguntas levantadas no início do capítulo, permitindo complementar as evidências que foram obtidas no estudo de caso.

Para a primeira pergunta, “O AR Educational Framework permite o desenvolvimento de aplicações educativas de Realidade Aumentada com facilidade?”, inicialmente através dos resultados dos experimentos, os participantes conseguiram realizar com sucesso as atividades (ver Figura 6.11), tendo uma percepção de facilidade usando o *framework* (ver Figuras 6.12, 6.13, e 6.14), já quanto ao uso do *framework* para criar aplicações completas de RA (que é o objetivo da pergunta), ainda que eles não criaram uma aplicação completa, através dos resultados das perguntas da Figura 6.16, eles dão uma visão positiva do *framework*, indicando que seria possível criar esse tipo de aplicativos com facilidade, o que reafirma as evidências do estudo de caso, no qual o autor criou um aplicativo educativo de RA indicando também alta facilidade. Assim através dos resultados dos experimentos com participantes, e do estudo de caso, a pergunta é respondida positivamente.

Como nota adicional sobre a facilidade percebida com o *framework* nos experimentos realizados, é interessante também indicar algumas respostas da pergunta aberta: “Como você avalia sua experiência de utilizar o Framework para desenvolver cenários educativos de Realidade Aumentada?”, pela qual participantes indicaram que a experiência com o *framework* resultou fácil e agradável, e, até certo, ponto divertida. Algumas das respostas são apresentas a seguir:

Participante A: *“Achei fácil, interessante e até divertido”.*

Participante B: “*Muito boa e divertida, o tutorial prático traz um sentimento de objetivo cumprido”.*

Participante C: *“Foi agradável utilizar o Framework, ele é fácil de usar e intuitivo. Não me senti incapacitado em momento algum e, após entender como usá-lo pela 1ª vez, consegui fazer os experimentos facilmente”.*

Para a pergunta “O AR Educational Framework é útil tanto para desenvolvedores experientes, quanto iniciantes em Unity e Vuforia?”, através dos resultados dos experimentos, uma vez que os participantes conseguiram realizar com sucesso as atividades propostas com o *framework* (ver Figura 6.11), e notando que no levantamento de conhecimentos (ver Figura 6.8) os participantes em geral indicaram ter conhecimento básico nas ferramentas necessárias (Unity e Vuforia), pode-se concluir que o *framework* seria de utilidade para desenvolvedores iniciantes nas ferramentas, e, semelhantemente, mostrou-se útil para desenvolvedores experientes através do aplicativo criado no estudo de caso pelo próprio autor. Assim através dos resultados dos experimentos com participantes e do estudo de caso, a pergunta pode ser respondida positivamente. Como nota adicional sobre a pergunta analisada, é interessante também indicar a resposta de um dos participantes do experimento (o qual indicou não ter nenhum conhecimento no Unity, e na Vuforia) para a pergunta aberta “Como você avalia sua experiência de utilizar o Framework para desenvolver cenários educativos de Realidade Aumentada?”:

Participante A: *“Foi gratificante, uma vez que não possuía conhecimento algum do Framework e mesmo assim consegui atingir com sucesso os objetivos”.*

Para a pergunta: “O AR Educational Framework fornece funcionalidades customizáveis de acordo às necessidades do usuário?”, a partir dos resultados da pergunta da Figura 6.15, a totalidade dos participantes acharam que o *framework* é altamente customizável (50% concordaram totalmente, e o outro 50% concordaram parcialmente), reafirmando as evidências obtidas no estudo de caso através do aplicativo criado. Assim, definitivamente a pergunta foi respondida positivamente. Como nota adicional sobre o alto nível de customização percebida com o *framework* nos experimentos realizados, é interessante também indicar a resposta de um dos participantes para a pergunta aberta “Quanto você acha que o Framework iria apoiar o desenvolvimento de aplicações educativas de Realidade Aumentada?”, que foi a seguinte:

Participante A: *“Por se tratar de um Framework facilmente customizável, acredito que a curva de aprendizado seja menor e, por isso, a satisfação de fazer e poder conferir rapidamente os resultados aumenta a vontade de criação”.*

Finalmente, para a última pergunta, “O AR Educational Framework é de cunho geral na educação, ou seja útil para qualquer área na educação?”, através das evidências obtidas no estudo de caso, no qual o autor conseguiu criar um aplicativo educativo de RA com dois cenários de áreas totalmente diferentes (Geometria e Biologia), usando as mesmas funcionalidades do *framework*, a pergunta pode ser respondida positivamente.

Ao final pode-se notar que as perguntas idealizadas foram respondidas positivamente neste capítulo, demostrando que o *framework* pode permitir o desenvolvimento de aplicações educativas de Realidade Aumentada com facilidade, sendo simples o suficiente para ser usado tanto por desenvolvedores experientes quanto iniciantes no Unity e Vuforia, ainda fornecendo um pacote de funcionalidades altamente customizáveis de acordo com as necessidades do usuário, e de cunho geral na educação (úteis em qualquer área da educação).

Capítulo 7

CONCLUSÃO

Foi apresentado, no decorrer deste trabalho, um *framework* para apoiar o desenvolvimento de aplicativos educativos com RAM baseada em marcadores, visando permitir a desenvolvedores a criação desse tipo de aplicativos com facilidade, fornecendo um conjunto de funcionalidades úteis em qualquer área da educação e altamente customizáveis segundo a necessidade do usuário. Neste trabalho, também foi apresentada uma metodologia para guiar o desenvolvimento fazendo uso do *framework* proposto.

Para criar o *framework* proposto, foi usado como base o *game engine* Unity e o SDK de RA chamado Vuforia. Na Seção 7.1, será possível verificar as contribuições deste trabalho e, na Seção 7.2, as limitações identificadas. Sugestões de trabalhos futuros para continuidade desta dissertação são descritas na Seção 7.3.

7.1 Contribuições

A principal contribuição deste trabalho foi apresentar um *framework* que permite a criação de aplicativos educativos de RAM com facilidade, sendo que, no decorrer deste, têm sido apresentadas evidências de sucesso no cumprimento desse objetivo. No capítulo 6, através dos experimentos e do questionário aplicado aos participantes, os resultados obtidos sugerem que, para os participantes do

experimento, o *framework* é de fácil utilização, e que permite criar com facilidade tanto cenários educativos simples de RA, quanto aplicações completas, reafirmando a opinião dada pelo autor a partir do aplicativo criado no estudo de caso (Capítulo 5). Nota-se que os participantes dos experimentos realizados, tinham, em geral, conhecimento básico no Unity e Vuforia, portanto o *framework* mostrou-se de fácil utilização mesmo para um público iniciante nos tópicos relacionados.

Acredita-se que a facilidade fornecida pelo *framework* se deve principalmente à possibilidade de implementar um conjunto de funcionalidades sem necessidade de programação. Ao invés disso, a programação inerente é abstraída pelo processo de arrastar e soltar classes que possibilita um ganho no tempo e na complexidade do desenvolvimento. Outro ponto importante que presumidamente influencia positivamente na simplicidade do *framework*, é o uso da inicialização *default* (ou inicialização padrão) suportada através das classes da pasta "**`EditMode`**" do *framework*, pois ainda sem ser necessário programação, o desenvolvedor normalmente precisaria montar a cena toda desde o início com todos os elementos necessários para a funcionalidade (ou seja, criar e posicionar recursos tais como marcadores, imagens, textos, botões, modelos 3D etc.). Porém utilizando a inicialização *default* que foi fornecida pelo *framework*, ao invés de ter que criar todos os elementos e montar a cena, esse processo pode ser automatizado fornecendo cenas prontas com elementos padrões do Unity e Vuforia, que só devem ser customizadas com os novos recursos. Neste ponto, deve-se notar que a inicialização padrão não está disponível para a totalidade das funcionalidades fornecidas pelo *framework*, sendo este aspecto discutido posteriormente na Seção 7.2.2.

Outra contribuição importante deste trabalho consiste em fornecer um *framework* que permite alta flexibilidade no desenvolvimento, sendo que, no decorrer deste, têm sido apresentadas evidências de sucesso no cumprimento de mais este objetivo. No capítulo 6, através dos experimentos, os resultados obtidos sugerem que o *framework* é altamente customizável para os participantes, reafirmando a opinião do autor a partir do aplicativo criado no estudo de caso (Capítulo 5), o qual conseguiu criar cenários com perguntas de dois tipos bem diferentes (de acordo com as necessidades de cada cenário), e customizar cada uma delas (informação necessária, elementos 3D e 2D, tempo de resposta etc.), assim como eventos

fornecidos pelo *framework* que foram usados (evento de mostrar informação adicional, mudar cor de objetos, e mudar escala e rotação), ainda integrando com sucesso código adicional ao *framework* que permitiu, através de menus, realizar a navegação entre as perguntas.

Mais uma contribuição importante deste trabalho consiste em fornecer um *framework* (além de fácil de usar, e altamente customizável), com funcionalidades úteis em qualquer área da educação (ou seja, de cunho geral na educação), pela qual, no decorrer deste trabalho, têm sido apresentadas evidências de ter cumprido esse objetivo. Tais evidências foram obtidas a partir do estudo de caso (Capítulo 5), em que o autor conseguiu desenvolver com sucesso um aplicativo com dois cenários de áreas totalmente diferentes (Geometria e Biologia, respetivamente).

Finalmente, como uma última contribuição, além do *framework* proposto, neste trabalho também foi apresentada uma metodologia com *guidelines* para guiar o desenvolvimento, fazendo uso do *framework* proposto.

Os pontos fortes deste trabalho são apresentados resumidamente a seguir:

- *O framework* proposto permite a criação de aplicativos educativos de RAM de um jeito simples.
- O *framework* proposto é útil tanto para desenvolvedores experientes, quanto para iniciantes no Unity e Vuforia.
- O *framework* proposto permite alta flexibilidade no desenvolvimento, visto que fornece funcionalidades altamente customizáveis de acordo com as necessidades do usuário e, além disso, uma vez que as funcionalidades são fornecidas através de classes, essas classes podem ainda ser acessadas diretamente pelo desenvolvedor, ser modificadas e estendidas sem nenhum tipo de limitação, permitindo ainda mais possibilidade de customização das funcionalidades existentes, ou ainda acrescentar novas funcionalidades desenvolvidas pelo usuário.
- O *framework* proposto é de cunho geral na educação, ou seja, útil em qualquer área da educação.

- Além do *framework* proposto, um conjunto de *guidelines* também foi proposto, visando com este projeto, além de facilitar codificação (através do *framework*), guiar o desenvolvedor em todo processo de criação do aplicativo através de uma série de passos bem definidos.

7.2 Limitações

As limitações são apresentadas a seguir, sendo divididas em limitações deste trabalho e limitações do *framework* proposto.

7.2.1 Limitação deste Trabalho

Foram identificadas as principais limitações deste trabalho, podendo ser listadas em:

- Para os experimentos realizados, devido ao tempo requerido, só foram utilizadas duas das funcionalidades fornecidas pelo *framework,* que foram: “Suporte de Perguntas” e “Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle”, descritas na Seção 4.1.3.2.
- Para os experimentos realizados, devido ao tempo requerido, os participantes só criaram cenários simples de RA, e não aplicativos completos.

7.2.2 Limitação do Framework Proposto

Foram identificadas as principais limitações do *framework* proposto, podendo ser listadas em:

- Uma vez que o *framework* foi desenvolvido totalmente utilizando Unity e Vuforia, tem como requisito obrigatório o uso de tais ferramentas, assim como o conhecimento básico de algumas operações

necessárias (principalmente no Unity a navegação básica pelo cenário, e na Vuforia algumas configurações). Porém o conhecimento requerido dessas ferramentas é completamente básico, como demostrado nos experimentos realizados, onde o *framework* foi usado com sucesso por pessoas com conhecimento básico do Unity, e sem nenhum conhecimento da Vuforia. Em relação a este ponto, também deve-se lembrar que Unity e a Vuforia são duas das ferramentas mais utilizadas para o desenvolvimento de aplicativos de RAM, portanto acredita-se que o *framework* terá alta difusão, e poderá ser amplamente utilizado pela comunidade.

Os recursos do AR Educational Framework (pacote do *framework* **"AR Educational Framework.unitypackage"**, documentação etc.) estão disponíveis em Github, através do link:
https://github.com/MarcosTulioSDLV/AR-Educational-Framework

Recursos do aplicativo desenvolvido no estudo de caso usando o *framework* (APK do aplicativo, projeto Unity, e documentação etc.), também estão disponíveis em Github, através do link:
https://github.com/MarcosTulioSDLV/Example-App-Using-AR-Educational-Framework

- O *framework* não forneceu nenhuma funcionalidade voltada ao uso de sons, nem foi considerado o uso de sons em nenhuma das funcionalidades fornecidas, portanto para integrar sons em aplicativos criados com o *framework,* será necessário programação adicional.
- A definição da ordem de renderização de elementos UI (textos 2D, imagens), assim como ajuste automático deles em diferentes tamanhos de tela, deve ser gerenciada pelo próprio desenvolvedor. Porém, deve-se notar que tais operações (embora não fornecidas pelo *framework*), estão disponíveis no Unity de jeito simples sem requerer programação adicional (unicamente realizando configurações nos elementos UI e **`Canvas`**). Assim, mesmo os usuários iniciantes no Unity não deveriam ter maiores problemas nesse aspecto.
- A inicialização *default* (ou inicialização padrão) suportada através das classes da pasta "**`EditMode`**" não está disponível para a totalidade das

funcionalidades fornecidas pelo *framework* (para cada classe da pasta "**`MainScripts`**" só existe uma inicialização *default* disponível, que é feita através de uma única classe associada da pasta "**`EditMode`**"), portanto tem casos nos quais o usuário não poderá usar inicialização padrão e precisará criar todos os elementos necessários (marcadores, imagens, textos, botões, modelos 3D etc.) e será responsável por montar a cena. Um exemplo disso pode ser citado com a classe "**`ChangeElement`**", e usando sua inicialização *default* através de "**`ChangeElementE`**", a qual permitirá realizar unicamente a inicialização *default* da classe "**`ChangeElement`**" para a funcionalidade chamada "Suporte a modificação dos materiais de objetos através de marcadores de controle", descrita na Seção 4.1.3.2, e não para as outras funcionalidades fornecidas com "**`ChangeElement`**".

7.3 Trabalhos Futuros

Algumas sugestões de trabalhos futuros para continuidade desta dissertação, e que foram considerados relevantes, são descritas a seguir:

- Realizar novos experimentos similares aos realizados, porém usando outras funcionalidades do *framework* que não foram consideradas, e, portanto, que permitam uma validação mais ampla do *framework*. Nos novos experimentos, também poderia ser avaliado o *framework* com um público sem nenhum conhecimento de computação (p.ex., professores). Este último ponto teria como objetivo mostrar que, embora o *framework* tenha como público alvo desenvolvedores, ele é simples o suficiente para ser usado ainda por pessoas sem conhecimento de programação (devido principalmente ao fato de que o *framework* permite abstrair programação pelo processo de arrastar e soltar classes).
- Realizar experimentos que permitam validar o conjunto de *guidelines* e *framework* propostos em termos de aprendizagem, visando medir qual o impacto de aplicativos criados no entendimento dos temas por parte dos

estudantes, determinar se esses aplicativos resultam relevantes no processo de ensino-aprendizagem.

- Realizar experimentos com o *framework* proposto, que permitam avaliar as suas vantagens em relação ao desenvolvimento tradicional (sem fazer o uso do *framework*, ou seja, utilizando-se apenas Unity e Vuforia). Nestes experimentos deveria ter especial relevância o tempo de desenvolvimento requerido em cada uma das duas abordagens (usando o *framework* X usando desenvolvimento tradicional).
- Quanto ao *framework*, um ponto que pode ser considerado importante para a sua evolução seria o suporte do uso de sons, seja para complementar funcionalidades existentes (através de novos campos nas classes das funcionalidades existentes), ou ainda para novas funcionalidades totalmente independentes (através de novas classes criadas, para as novas funcionalidades).
- Outro ponto para a continuidade do *framework* pode ser implementar novas funcionalidades além das relacionadas com sons. Um exemplo seria um novo suporte à modificação da *transform* de objetos usando o *touch screen* do dispositivo, ou seja, permitir rotacionar e escalar os objetos em RA, através da tela sensível ao toque dos dispositivos.

REFERÊNCIAS

Abdulmuslih Alsirhani, M. (2012). *Análisis de sistemas de realidad aumentada y metodología para el desarrollo de aplicaciones educativas*. Universidad Rey Juan Carlos. Retrieved from https://eciencia.urjc.es/handle/10115/7805

Araújo, T., Santos, C., Carneiro, N., Meiguins, B., & Miranda, B. (2015). Aplicações Android de Realidade Aumentada em Arquitetura Extensível, Flexível e Adaptável. Retrieved from http://www.lbd.dcc.ufmg.br/colecoes/sbsi/2015/010.pdf

ARToolworks. (2017a). ARToolKit. Retrieved from https://www.hitl.washington.edu/artoolkit/

ARToolworks. (2017b). FLARToolKit. Retrieved from http://www.libspark.org/wiki/saqoosha/FLARToolKit/en

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., & MacIntyre, B. (2001). Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, *21*(6), 34–47. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=963459

Azuma, R. T. (1997). A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, *6*(4), 355–385. Retrieved from http://www.cs.unc.edu/~azuma/ARpresence.pdf

Baricevic, D., Hollerer, T., Sen, P., & Turk, M. (2016). User-Perspective AR Magic Lens from Gradient-Based IBR and Semi-Dense Stereo. *IEEE Transactions on Visualization and Computer Graphics*. Retrieved from http://ieeexplore.ieee.org/document/7460953/

Baricevic, D., Lee, C., Turk, M., Hollerer, T., & Bowman, D. A. (2012). A Hand-Held AR Magic Lens with User-Perspective Rendering. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*. IEEE. Retrieved from http://ieeexplore.ieee.org/document/6402557/

Billinghurst, M. (2002). Augmented reality in education. *New Horizons for Learning*. Retrieved from http://www.solomonalexis.com/downloads/ar_edu.pdf

Billinghurst, M., & Duenser, A. (2012). Augmented reality in the classroom. *Computer*, *45*(7), 56–63. Retrieved from http://ieeexplore.ieee.org/abstract/document/6171143/authors

Blum, T., Kleeberger, V., Bichlmeier, C., & Navab, N. (2012). mirracle: An Augmented Reality Magic Mirror System for Anatomy Education. In *Virtual*

Reality Short Papers and Posters (VRW), 2012 IEEE (pp. 115–116). IEEE. Retrieved from http://ieeexplore.ieee.org/document/6180909/

Camba, J. D., & Contero, M. (2015). From Reality to Augmented Reality: Rapid Strategies for Developing Marker-Based AR Content Using Image Capturing and Authoring Tools. In *Frontiers in Education Conference (FIE), 2015 IEEE* (pp. 1–6). IEEE. Retrieved from http://ieeexplore.ieee.org/document/7344162/?reload=true

Chowdhury, S. A., Obeidy, W. K., Arshad, H., & Parhizkar, B. (2013). A Mobile Augmented Reality and Multimedia Application for Mobile Learning. *International Journal of Digital Content Technology and Its Applications*, *7*(13), 25–32. Retrieved from https://www.researchgate.net/publication/257154768_A_Mobile_Augmented_Reality_and_Multimedia_Application_for_Mobile_Learning

Cubillo, J., Martín, S., Castro, M., Díaz, G., Colmenar, A., & Botički, I. (2014). A learning environment for augmented reality mobile learning. In *Frontiers in Education Conference (FIE)* (pp. 1–8). IEEE. Retrieved from http://ieeexplore.ieee.org/abstract/document/7044039/?reload=true

Cuendet, S., Bonnard, Q., Do-Lenh, S., & Dillenbourg, P. (2013). Designing Augmented Reality for the Classroom. *Computers & Education*, *68*, 557–569. Retrieved from https://infoscience.epfl.ch/record/181631/files/finalVersion.pdf

Fabregat Gesa, R. (2012). Combinando la realidad aumentada con las plataformas de e-elearning adaptativas. *Enl@ce Revista Venezolana de Información, Tecnología Y Conocimiento*, *9*(2), 69–78. Retrieved from https://dialnet.unirioja.es/servlet/articulo?codigo=3971545

FitzGerald, E., Ferguson, R., Adams, A., Gaved, M., Mor, Y., & Thomas, R. (2013). Augmented reality and mobile learning: the state of the art. *International Journal of Mobile and Blended Learning*, *5*(4), 43–58. Retrieved from http://oro.open.ac.uk/38386/

Forte, C. E., & Kirner, C. (2009). Usando Realidade Aumentada no Desenvolvimento de Ferramenta para Aprendizagem de Física e Matemática. In *6º Workshop de Realidade Virtual e Aumentada-WRVA* (pp. 1–6). Retrieved from http://sites.unisanta.br/wrva/st/62200.pdf

Gao, L., Watson, R., Billinghurst, M., Wei, C., & Bai, H. (2017). Creating, Training and Adding Markers. Retrieved from https://envisage-ar.com/envisage-tutorials/7/

García, I., Peña-López, I., Johnson, L., Smith, R., Levine, A., & Haywood, K. (2010). *Informe Horizon: Edición Iberoamericana 2010*. Austin, Texas: The New Media Consortium. Retrieved from http://www.nmc.org/pdf/2010-Horizon-Report-ib.pdf

Grubert, J., & Grasset, R. (2013). *Augmented Reality for Android Application Development*. Birmingham: Packt Publishing Ltd. Retrieved from

http://file.allitebooks.com/20151101/Augmented Reality for Android Application Development.pdf

Holmes, K. (2010). Projective Augmented Reality Turns Cars Inside Out. Retrieved from https://creators.vice.com/en_us/article/8qmkak/projective-augmented-reality-turns-cars-inside-out

Intel Graphics Performance Primitives for the Intel® PXA250 Applications Processor. (n.d.). Retrieved from http://www.intel.com/design/pca/applicationsprocessors/swsup/gppv1.htm

Kangdon, L. (2012). Augmented reality in education and training. *TechTrends*, *56*(2), 13–21. Retrieved from https://link.springer.com/article/10.1007/s11528-012-0559-3?LI=true

Katiyar, A., Kalra, K., & Garg, C. (2015). Marker Based Augmented Reality. *Advances in Computer Science and Information Technology (ACSIT)*, *2*(5), 441–445. Retrieved from http://www.krishisanskriti.org/vol_image/04Jul201511074828 Anuroop Katiyar 441-445.pdf

Kaufmann, H. (2002). Construct3D: An Augmented Reality Application for Mathematics and Geometry Education. In *Proceedings of the tenth ACM international conference on Multimedia* (pp. 656–657). ACM. Retrieved from http://dl.acm.org/citation.cfm?id=641140

Khronos Group. (2017a). OpenGL Headline News. Retrieved from https://www.opengl.org/

Khronos Group. (2017b). opengles. Retrieved from https://www.khronos.org/opengles/

Kirner, C. (2013). SICARA. Retrieved from http://www.ckirner.com/claudio/?PROJETOS:SICARA

Layar. (2017). Layar. Retrieved from https://www.layar.com/

Mahale, P., & Yeddu, S. (2016). Android-based Augmented Reality to Enhance Education System. *International Journal of Computer Applications*, *146*(6), 0975 – 8887. Retrieved from http://www.ijcaonline.org/archives/volume146/number6/mahale-2016-ijca-910790.pdf

Mehigan, T. J. (2009). Harnessing accelerometer technology for inclusive mobile learning. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services* (p. 100). ACM. Retrieved from http://dl.acm.org/citation.cfm?id=1613973

Milgram, P., & Kishino, F. (1994). A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions on Information Systems*. Retrieved from https://cs.gmu.edu/~zduric/cs499/Readings/r76JBo-Milgram_IEICE_1994.pdf

Morrison, A., Oulasvirta, A., Peltonen, P., Lemmela, S., Jacucci, G., Reitmayr, G., … Juustila, A. (2009). Like Bees Around the Hive: A Comparative Study of a Mobile Augmented Reality Map. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1889–1898). Retrieved from http://dl.acm.org/citation.cfm?id=1518991

Multidots.com. (2015). Augmented Reality. Retrieved from https://www.multidots.com/augmented-reality/

Muñoz, J., Osorio, B., Álvarez, F., & Cardona, P. (2006). *Metodología para elaborar Objetos de Aprendizaje e integrarlos a un Sistema de Gestión de Aprendizaje*. *Revista Apertura del Sistema de Universidad Virtual*. México.

Nakamoto, P. T., Carrijo, G. A., & Cardoso, A. (2010). Construção de ambientes educacionais com realidade aumentada: processo centrado no usuário. *RENOTE*, *7*(3), 244–252. Retrieved from http://seer.ufrgs.br/renote/article/view/13564

Nincarean, D., Ali, M. B., Halim, N. D. A., & Rahman, M. H. A. (2013). Mobile Augmented Reality: the potential for education. *Procedia - Social and Behavioral Sciences*, *103*, 657 – 664. Retrieved from http://www.sciencedirect.com/science/article/pii/S1877042813038305

NyARToolkit. (2017). NyARToolkit project. Retrieved from http://nyatla.jp/nyartoolkit/wp/?page_id=198

Optical head-mounted display. (2017). In wikipédia. Retrieved from https://en.wikipedia.org/wiki/Optical_head-mounted_display

Oracle. (2017). JAVA PLATFORM, MICRO EDITION (JAVA ME). Retrieved from http://www.oracle.com/technetwork/java/embedded/javame/index.html

Papagiannakis, G., Singh, G., & Magnenat-Thalmann, N. (2008). A survey of mobile and wireless technologies for augmented reality systems. *Computer Animation and Virtual Worlds*, *19*(1), 3–22. Retrieved from http://dl.acm.org/citation.cfm?id=1348087

Pressman, R. S. (2011). *Engenharia de Software: Uma Abordagem Profissional*. AMGH Editora Ltda. Retrieved from https://fateczlads.files.wordpress.com/2014/08/engenharia-de-software-7c2b0-edic3a7c3a3o-roger-s-pressman-capc3adtulo-1.pdf

PTC. (n.d.). Optimizing Target Detection and Tracking Stability. Retrieved from https://library.vuforia.com/articles/Solution/Optimizing-Target-Detection-and-Tracking-Stability.html

PTC. (2017). Vuforia Developer Portal. Retrieved from https://developer.vuforia.com/

Realidad aumentada. (2017). In wikipédia. Retrieved from https://es.wikipedia.org/wiki/Realidad_aumentada

Robertson, D. (2011). Create your own BBC QRCode. Retrieved from http://www.bbc.co.uk/blogs/researchanddevelopment/2011/09/create-your-own-bbc-qrcode.shtml

Rodríguez Lomuscio, J. P. (2011). *REALIDAD AUMENTADA PARA EL APRENDIZAJE DE CIENCIAS EN NIÑOS DE EDUCACIÓN GENERAL BÁSICA*. Universidad de Chile.

Santin, R. (2008). *SACRA - Sistema de Autoria Em Ambiente Colaborativo com Realidade Aumentada.* UNIMEP - Universidade Metodista de Piracicaba.

Schmalstieg, D., & Wagner, D. (2009). Mobile Phones as a Platform for Augmented Reality. *Connections*, *1*, 3. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.144.6950&rep=rep1&type=pdf

Semacode. (n.d.). Semacode. Retrieved from http://semacode.org/

Sheikh, A., & Sawant, K. (2016). Introduction to Augmented Reality: An overview, Development of AR in Android. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *5*(6). Retrieved from http://ijarcet.org/wp-content/uploads/IJARCET-VOL-5-ISSUE-6-1989-1994.pdf

Siltanen, S. (2012). *Theory and Applications of Marker-based Augmented Reality*. Retrieved from http://mehmetsimsek.net/slides/bm533/S3.pdf

Singhal, S., Bagga, S., Goyal, P., & Saxena, V. (2012). Augmented Chemistry: Interactive Education System. *International Journal of Computer Applications*, *49*(15), 1–5. Retrieved from http://research.ijcaonline.org/volume49/number15/pxc3881041.pdf

Slijepcevic, N. (2016). Augmented Reality in Education. Retrieved from http://www.arined.org/

Tobar, H. F., Fabregat, R., & Baldiris, S. (2013). *AR Learning Videogame For Kids With ADHD Symptoms*. Universitat de Girona. Retrieved from https://www.researchgate.net/publication/272997565_AR_Learning_Videogame_For_Kids_With_ADHD_Symptoms

Tovar, L. C., Bohórquez, J. A., & Puello, P. (2014). Propuesta Metodológica para la Construcción de Objetos Virtuales de Aprendizaje basados en Realidad Aumentada. *Formación Universitaria*, *7*(2), 11–20. Retrieved from http://www.scielo.cl/scielo.php?pid=S0718-50062014000200003&script=sci_arttext

Van Krevelen, D. W. F., & Poelman, R. (2010). A survey of augmented reality technologies, applications and limitations. *International Journal of Virtual Reality*, *9*(2), 1–20. Retrieved from http://kjcomps.6te.net/upload/paper1 .pdf

Vergara, M. (2017). Augmented World Expo: How Unity developers will be part of the future. Retrieved from https://blogs.unity3d.com/es/2017/06/08/augmented-world-expo-how-unity-developers-will-be-part-of-the-future/

Wagner, D. (n.d.). SoftGL, An OpenGL Subset. Retrieved from http://www.ims.tuwien.ac.at/research/handheld_ar/developer/sof tgl.php

Wagner, D., & Barakonyi, I. (2003). Augmented Reality Kanji Learning. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality* (p. 335). IEEE Computer Society. Retrieved from http://dl.acm.org/citation.cfm?id=946816

Wu, H. K., Lee, S. W. Y., Chang, H. Y., & Liang, J. C. (2013). Current status, opportunities and challenges of augmented reality in education. *Computers & Education*, *62*, 41–49. Retrieved from http://www.sciencedirect.com/science/article/pii/S0360131512002527

Zuñiga Torres, J. C. (2008). *Uma Metodologia para o Desenvolvimento de Aplicações de Realidade Aumentada em Telefones Celulares Utilizando Dispositivos Sensores*. Universidade de São Paulo. Retrieved from http://www.teses.usp.br/teses/disponiveis/3/3142/tde-18022009-165517/pt-br.php

Apêndice A

TUTORIAL CONFIGURAÇÃO AR EDUCATIONAL FRAMEWORK

A seguir são apresentados os passos para instalar o AR Educational Framework sobre um projeto do Unity (versões do Unity a partir da 2017.2, que já vem com a Vuforia integrada diretamente, são requeridas para o *framework*). Os passos para configurar o AR Educational Framework, foram divididos em: "Configuração Da Vuforia", que consiste em realizar as configurações necessárias da Vuforia, e finalmente: "Configuração Do Framework", que consiste em instalar e configurar o AR Educational Framework. O AR Educational Framework foi criado com a versão do Unity 2018.1.0f2, e a versão da Vuforia 7.1.31.

Notas:

- Uma cena final chamada SampleScene, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALCONFIGURACAO, disponível no link: https://www.dropbox.com/s/lb3ysul6kqgemba/PROJETOUNITYTUTORIALCONFIGURACAO.rar?dl=0
- Visando facilitar as configurações, cada elemento do Unity que for necessário criar, terá indicado entre parêntesis os passos para sua criação.

Configuração Da Vuforia

a) Configurar O Projeto Do Unity Para Suportar O Uso Da Vuforia:

- Importar Recursos Da Vuforia: Substituir a **`MainCamera`** *default* do Unity por uma **`ARCamera`** da Vuforia (criar fazendo: **`GameObject/Vuforia/ARCamera`**), ver Figura A 1. Nesta operação se importarão automaticamente todos os recursos necessários da Vuforia no projeto atual do Unity. Nota: A câmera **`MainCamera`** *default* deve ser apagada nesse processo.

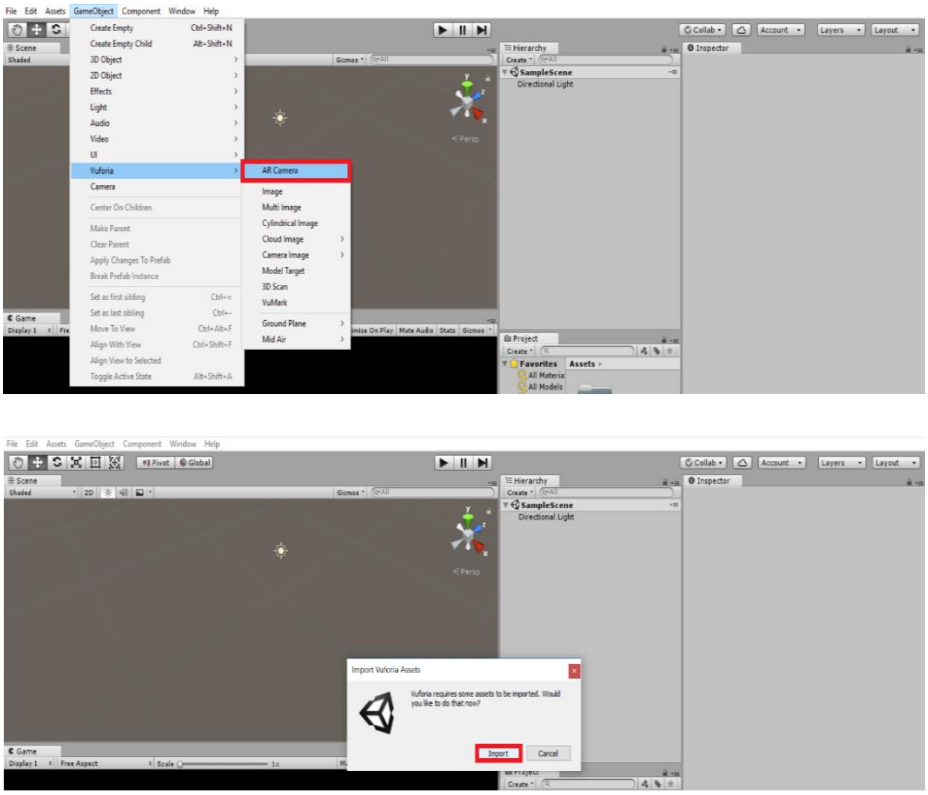

Figura A 1: Processo de criação de uma `ARCamera` que importará automaticamente os recursos da Vuforia.

- Habilitar Vuforia: Habilitar Vuforia no projeto do Unity para a plataforma destino do aplicativo, em: **`File/BuildSettings`**, selecionando a plataforma (Android), fazendo click em **`PlayerSettings`**, e finalmente nas propriedades do inspetor do Unity em: **`XRSettings`**, ativar a opção: **`Vuforia Augmented Reality Supported`**. Ver Figura A 2.

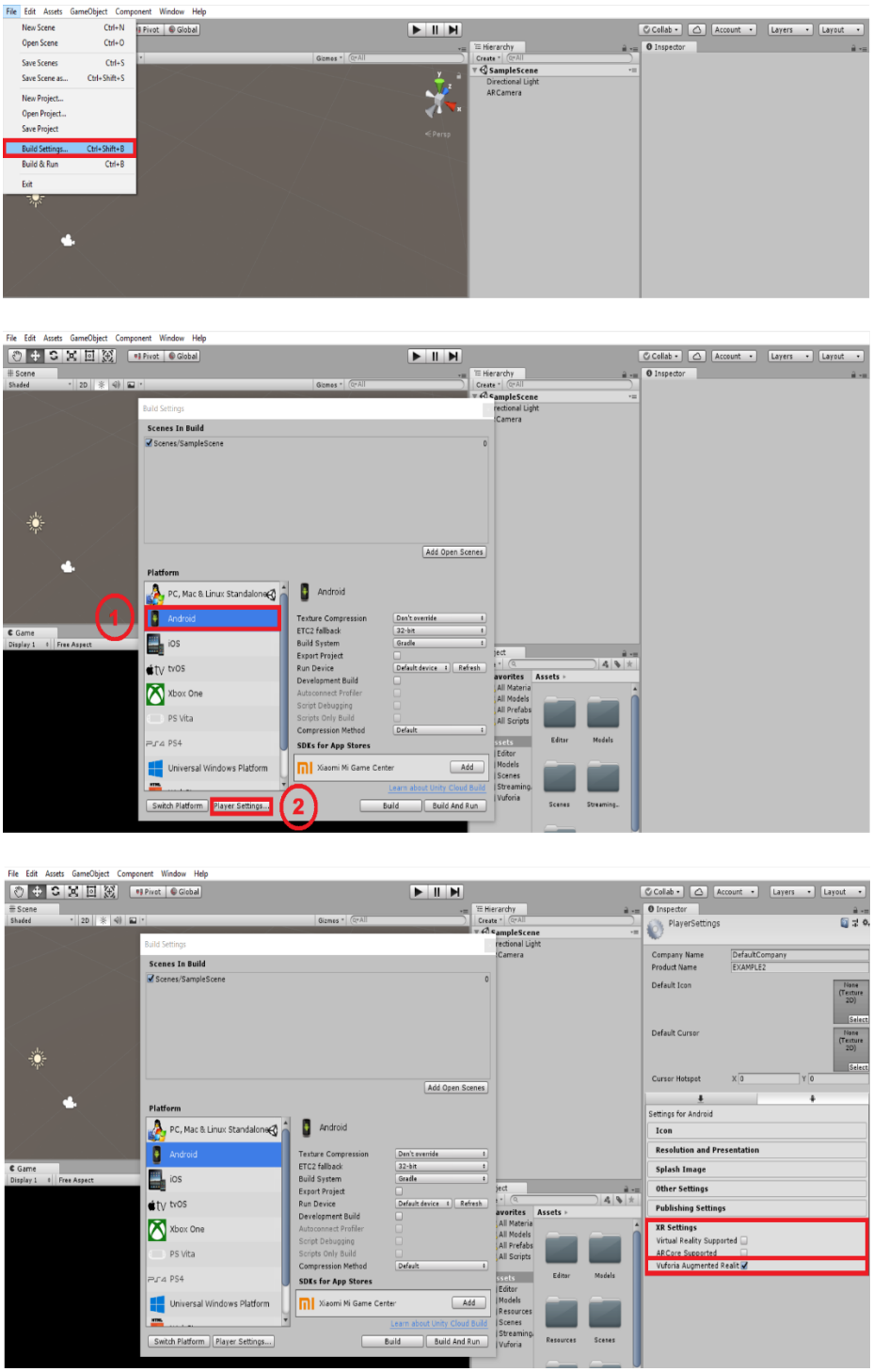

Figura A 2: Processo para habilitar Vuforia no projeto do Unity.

b) Importar Marcadores De RA E Habilita-los Para Seu Uso: No caso de usar marcadores próprios e não marcadores *default* da Vuforia, importar no projeto o pacote do banco de dados dos marcadores novos. Para importar um pacote do Unity (arquivo com extensão unitypackage), se faz através de:

`Assets/ImportPackage/CustomPackage`, após isso selecionando o pacote no diretório e com click em *import* para confirmar.

Finalmente, para poder usar os marcadores importados, deve-se redefinir o banco de dados dos marcadores utilizados no projeto, para isso: Selecionar a **`ARCamera`**, então nas propriedades do inspetor do Unity fazer click em "**`Open Vuforia Configuration`**", e copiar o código de licença do banco de dados importado no campo: **`App License Key`**, ver Figura A 3.

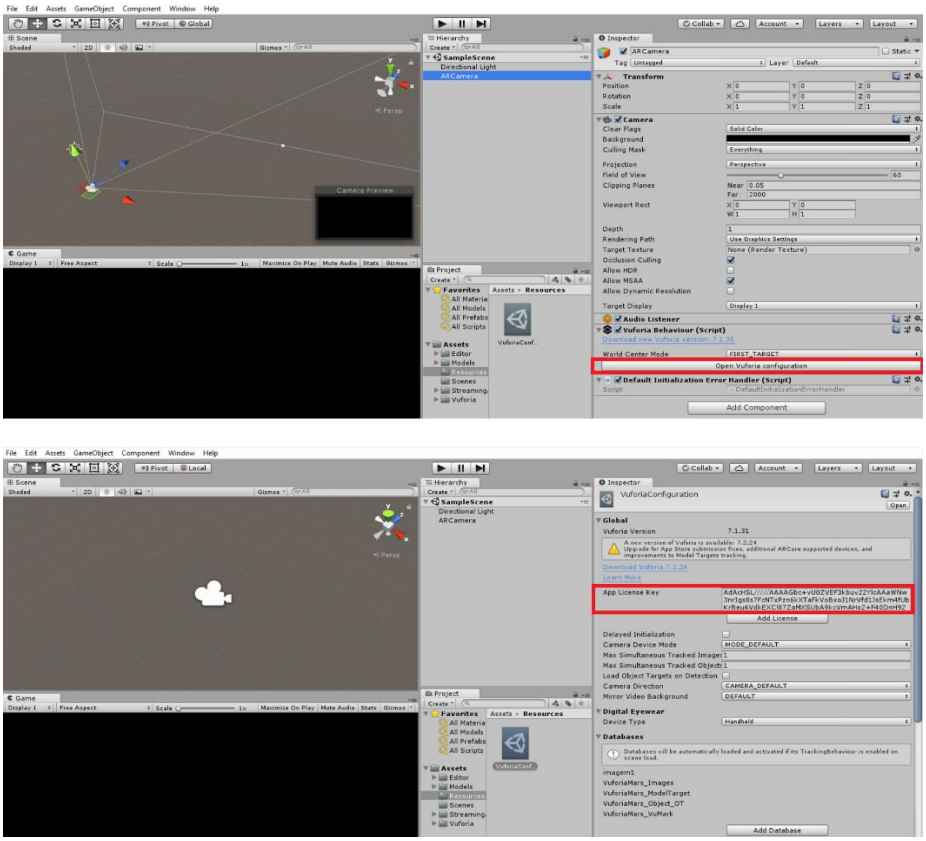

Figura A 3: Copiar código de licença do banco de dados dos novos marcadores.

Configuração Do Framework

a) Configurar Número Máximo De Marcadores Simultâneos: Selecionar a **`ARCamera`**, fazer click em "**`Open Vuforia Configuration`**", e finalmente no campo: **`Max Simultaneous Tracked Images`**, indicar o número máximo de marcadores simultâneos a suportar, ver Figura A 4. Nota: Para

usos básicos do *framework*, normalmente será suficiente com configurar o valor 2 neste campo.

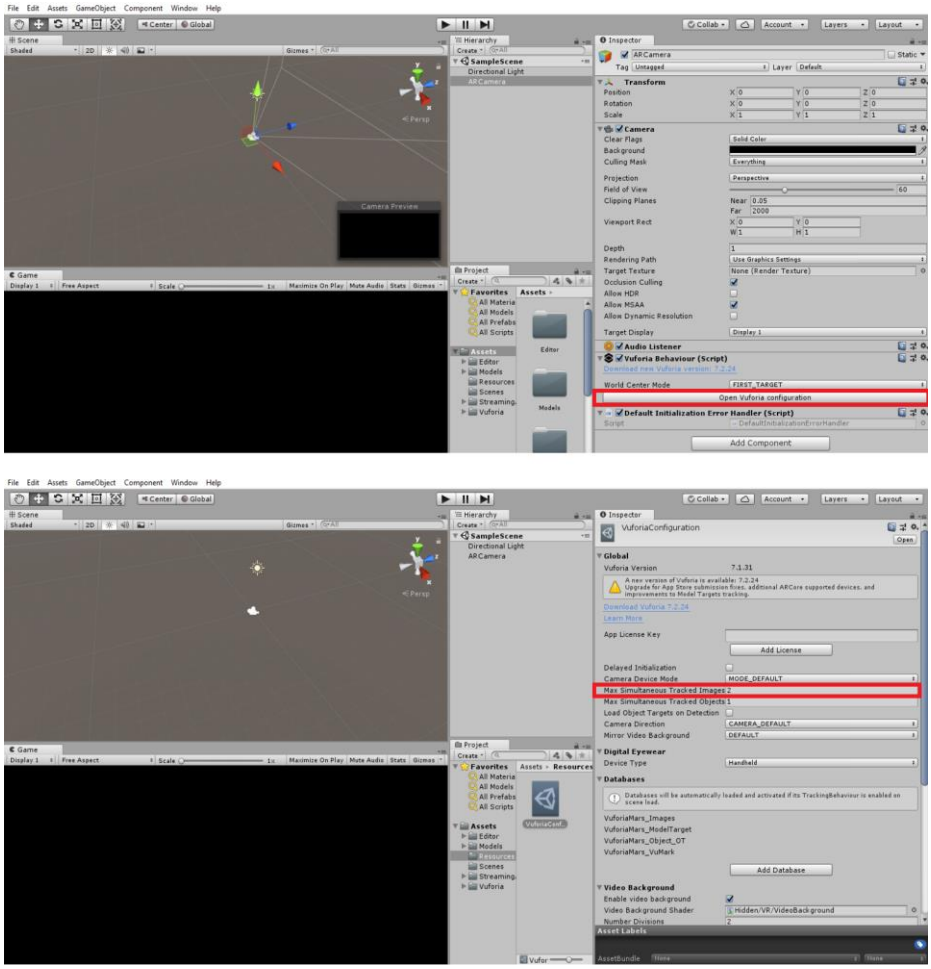

Figura A 4: Configuração número máximo de marcadores simultâneos.

b) Importar Framework: Importar no projeto, o pacote "**`AR Educational Framework.unitypackage`**", que possui os recursos do *framework*.

c) Configurar Marcadores Do Framework Na Cena: O *framework* utiliza marcadores próprios para suportar suas funcionalidades (ao invés dos marcadores criados de jeito normal), portanto para usar marcadores com o *framework*, deve-se arrastar na cena do Unity o marcador disponível chamado **`ImageTarget1Prefab`**, localizado na pasta com o caminho: **`AR Educational Framework/Resources`**, ver Figura A 5. Este marcador possui as configurações necessária para ser usado com o *framework*, e será só com marcadores com esta configuração que deverá ser utilizado o *framework*.

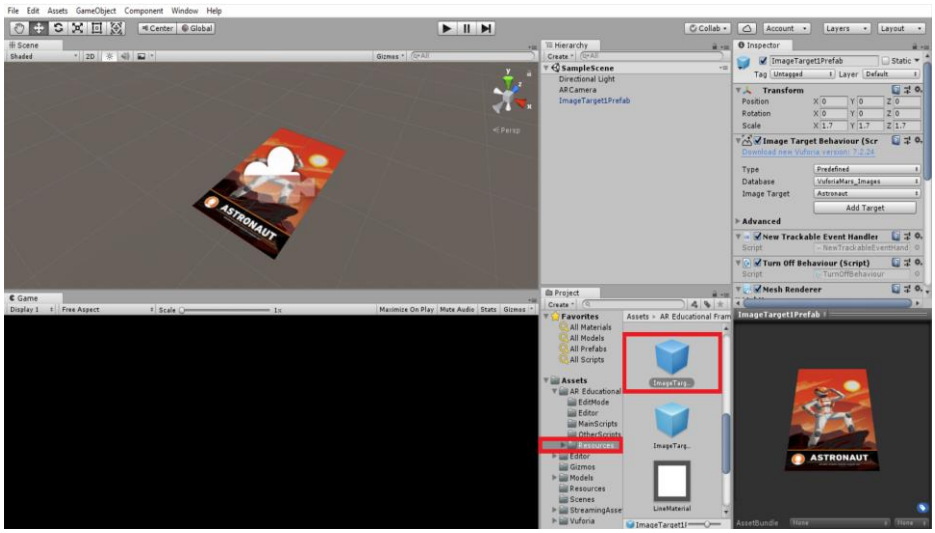

Figura A 5: Marcador *default* `ImageTarget1Prefab` arrastado na cena.

No caso de querer usar outro dos marcadores *default*, deve-se selecionar o marcador atual, e no campo: *ImageTarget*, escolher o que quiser, ver Figura A 6.

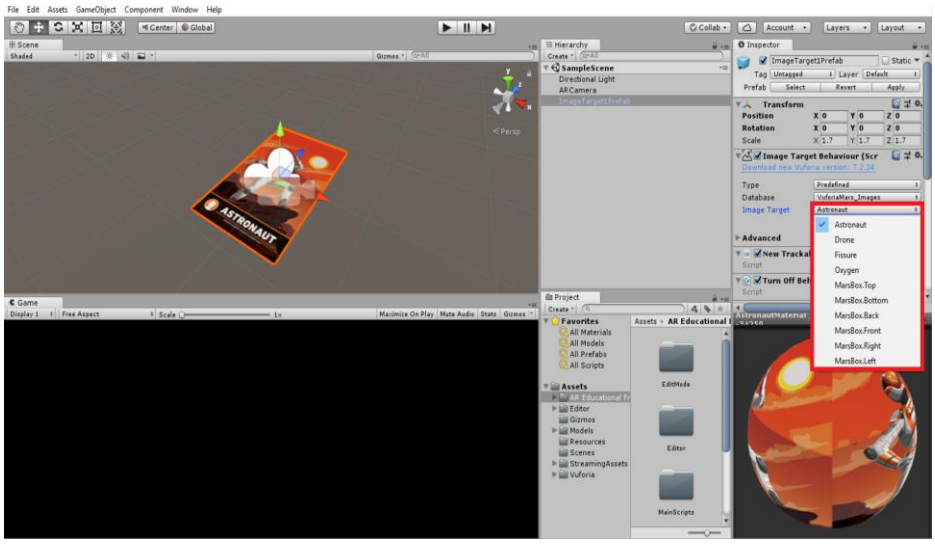

Figura A 6: Lista de marcadores *default* da Vuforia que podem ser escolhidos.

Já no caso de querer usar marcadores próprios, então deverá ser mudado o banco de dados *default* pelo novo importado, no seu campo: **`DataBase`**, ver Figura A 7, e finalmente escolher o marcador do mesmo jeito que foi mostrado na Figura A 6.

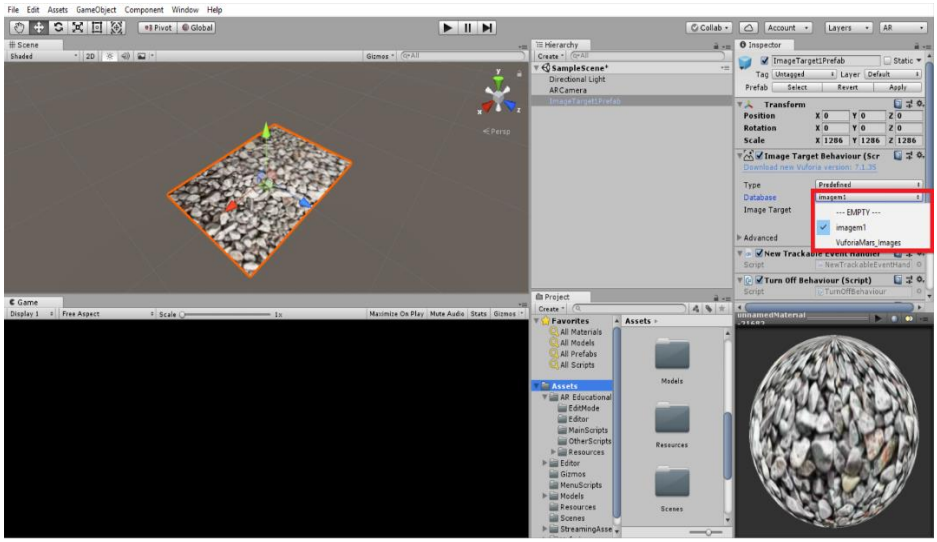

Figura A 7: Lista dos bancos de dados de marcadores disponíveis, imagem1 corresponde a um novo banco de dados já importado.

Um jeito adicional de criar marcadores para o *framework* (ao invés de arrastar o marcador disponível em **`Resources`**) consiste em criar o marcador de jeito normal (**`GameObject/Vuforia/Image`**), e faze-lhe manualmente a configuração necessária para poder usá-lo com o *framework*, que consiste em substituir sua classe **`DefaultTrackableEventHandler`** pela classe **`NewTrackableEventHandler`** do *framework* localizada na pasta **`OtherScripts`**.

Neste ponto, a cena já ficará pronta para vincular objetos 3D no marcador e utilizar as funcionalidades do AR Educational Framework.

d) Configuração Adicional Sugerida Usando Vários Marcadores: Sempre que for necessário usar vários marcadores simultâneos com o *framework*, recomenda-se adicionalmente configurar a **`ARCamera`** da cena como: **`ESPECIFIC_TARGET`**, na sua opção: ***`World Center Mode`***, e arrastar o marcador vinculado aos objetos (ou seja, o marcador que tem como filhos os objetos) no campo: ***`World Center`***, ver Figura A 8.

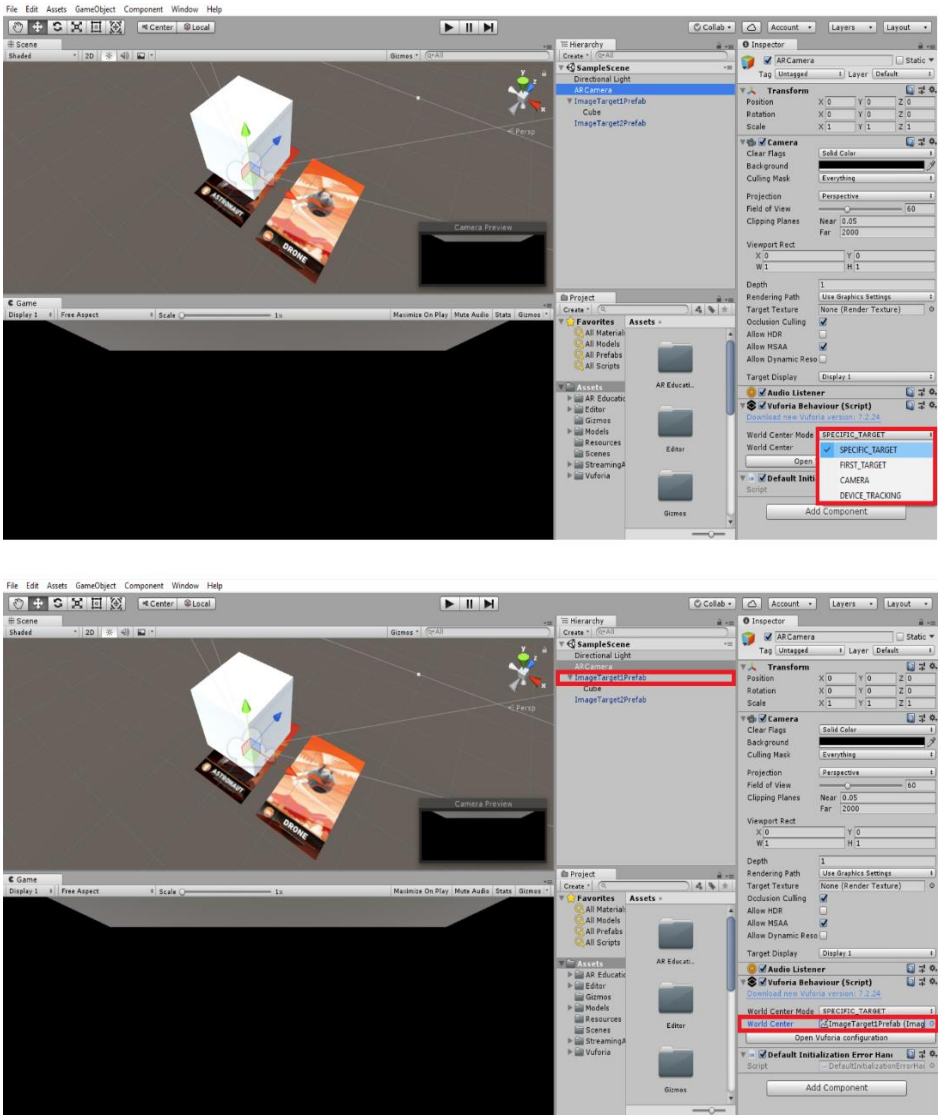

Figura A 8: Configuração adicional recomendada para usar vários marcadores simultâneos com o *framework*.

Apêndice B

CLASSES DO AR EDUCATIONAL FRAMEWORK

Organização De Pastas Do AR Educational Framework

O *framework* possui duas pastas principais chamadas: "**`AR Educational Framework`**" e "**`Gizmos`**", que são importadas automaticamente no projeto do Unity através do pacote "**`AR Educational Framework.unitypackage`**" do *framework*. Estas pastas e seu conteúdo serão explicados a seguir.

Pasta AR Educational Framework

A pasta **`AR Educational Framework`** possui dentro outras cinco pastas principais com scripts e recursos do *framework*, que são as seguintes: **`EditMode`**, **`Editor`**, **`MainScripts`**, **`OtherScripts`** e **`Resources`**. Essas pastas e seu conteúdo serão apresentadas de jeito detalhado a seguir.

- **EditMode:** Esta pasta contém classes que permitem ao usuário fazer uma inicialização padrão (ou inicialização *default*) para as classes das funcionalidades fornecidas pelo *framework* (classes contidas em **`MainScripts`**), o qual consiste em montar automaticamente uma cena com a funcionalidade respetiva e usando elementos padrão do Unity e Vuforia como exemplos (p.ex. cubos, esferas, marcadores *default* da Vuforia etc.)., especificamente, são criados, posicionados e configurados elementos

necessários tais como objetos 2D, 3D, marcadores etc., automaticamente no modo de edição do Unity (sem precisar executar Unity). Nota-se que os elementos criados na inicialização padrão são elementos padrão do Unity e Vuforia, e portanto a cena deverá ser customizada pelo usuário com os novos recursos necessários. Devido a que estas classes são para permitir inicialização padrão, seu uso será opcional (podendo ser criados todos os elementos necessários diretamente pelo usuário), e no caso da utilização dessas, recomenda-se sempre removê-las após ser feita a respectiva inicialização. As classes contidas dentro desta pasta são: **`ActivateInformationE`**, **`ButtonsPackE`**, **`ChangeElementE`**, **`QuestionController1E`** e **`QuestionController2E`**.

- **Editor:** Esta pasta contém classes encarregadas da customização do inspetor do Unity para as classes das funcionalidades fornecidas pelo *framework* (classes contidas em **`MainScripts`**). O uso de classes com inspetores customizados foi importante para que as classes deixassem disponíveis (através dos seus campos) diferentes opções de configuração que permitem customizar as funcionalidades dependendo das necessidades do desenvolvedor. Scripts de customização do inspetor devem sempre estar contidos numa pasta de nome especial para o Unity chamada **`Editor`** (é possível ter várias pastas **`Editor`** em um projeto). Nota-se desde que estas classes são unicamente para customização do inspetor, o usuário não precisa interagir com elas para utilizar as funcionalidades fornecidas pelo *framework*. As classes contidas dentro desta pasta são: **`ActivateInformationI`**, **`ButtonOperationsI`**, **`ChangeElementI`**, **`QuestionController1I`** e **`QuestionController2I`**. A classe **`ActivateInformationI`** está encarregada de customizar o inspetor do Unity para sua versão respetiva na pasta **`MainScripts`** (classe **`ActivateInformation`**), já a classe **`ButtonOperationsI`** faz o mesmo processo para a classe **`ButtonOperations`** da pasta **`MainScripts`**, e assim por diante.

- **MainScripts:** Esta pasta contém as classes das funcionalidades fornecidas pelo *framework*, e portanto cada uma dessas classes representa uma

funcionalidade ou um conjunto de funcionalidades do *framework*. As classes contidas dentro desta pasta são: **`ActivateInformation`**, **`ButtonOperations`**, **`ButtonsPack`**, **`ChangeElement`**, **`QuestionController1`** e **`QuestionController2`**. De jeito resumido, a classe **`ActivateInformation`** permite ativar informações adicionais sobre um objeto em RA através do uso de um marcador de controle, a classe **`ButtonOperations`** permite mudar a *transform* (rotação e escala) de objetos em RA através de botões, porém tendo o usuário que criar cada botão um por um, já a classe **`ButtonsPack`** mais especializada permite criar automaticamente um pacote de botões usando a classe anterior **`ButtonOperations`**, a classe **`ChangeElement`** permite mudar características de um objeto ou trocar ele por outro através de um marcador de controle, e finalmente as classes **`QuestionController1`** e **`QuestionController2`** permitem a criação de perguntas de diferentes tipos (seguindo duas abordagens definidas).

- **OtherScripts:** Esta pasta contém classes adicionais utilizadas pelo *framework*. As classes contidas dentro desta pasta são: **`GeralObjectsContainer`**, **`NewTrackableEventHandler`**, **`ObjectContainerWithRenderer`**, **`OnClickElement`** e **`SubObjectContainerWithMaterial`**. Alguns detalhes relevantes destas classes são apresentados a seguir. As classes **`GeralObjectsContainer`**, **`ObjectContainerWithRenderer`**, e **`SubObjectContainerWithMaterial`**, não fornecem nenhuma funcionalidade além de permitir criar instâncias de objetos que foram necessários em algumas classes da pasta **`MainScripts`**. A classe **`OnClickElement`** permite detectar clicks em elementos do Unity, e especificamente foi usada para permitir a textos 2D e imagens detectar os click do usuário (necessário para a funcionalidade “Suporte de Perguntas” seguindo a “Abordagem 1”, descrita na Seção 4.3.1.2). Finalmente a classe **`NewTrackableEventHandler`** é uma versão customizada da classe **`DefaultTrackableEventHandler`** da Vuforia.

- **Resources:** Esta pasta contém recursos e *prefabs* necessários para o *framework*, e que são carregados automaticamente desde código. Nota-se que recursos a serem carregados desde código devem estar contidos sempre numa pasta de nome especial para o Unity chamada **`Resources`** (é possível ter várias pastas **`Resources`** em um projeto). Estes recursos são detalhados a seguir:

  ButtonOperationsPrefabs: A sub-pasta **`ButtonOperationsPrefabs`** contém um conjunto de botões *prefabs* com todos os botões para rotacionar e escalar que podem ser criados com o *framework* a partir da classe **`ButtonsPack`** e **`ButtonOperations`**.

  ButtonTextures: A sub-pasta **`ButtonTextures`** contém um conjunto de texturas com o atributo **`Texture Type`** configurado em: **`Sprite (2D and UI)`**, usadas na criação de botões padrões quando for feita inicializações padrões das classes.

  ButtonPrefab: Botão *prefab* usado como base para instanciar botões padrões quando for feita inicializações padrões das classes.

  CanvasPrefab: Elemento **`Canvas`** do Unity usado como base para criar instancias de **`Canvas`** quando for feita inicializações padrões das classes.

  CorrectAnswerImage: Textura com o atributo **`Texture Type`** configurado em: **`Sprite (2D and UI)`**, usada para criar a imagem de *feedback* de resposta correta quando for feita inicializações padrões das classes **`QuestionController1`** e **`QuestionController2`**.

  WrongAnswerImage: Textura com o atributo **`Texture Type`** configurado em: **`Sprite (2D and UI)`**, usada para criar a imagem de *feedback* de resposta errada quando for feita inicializações padrões das classes **`QuestionController1`** e **`QuestionController2`**.

  ImageTarget1Prefab: Marcador *prefab* usado como base para instanciar o primeiro marcador (marcador principal) quando for feita inicializações padrões das classes.

  ImageTarget2Prefab: Marcador *prefab* usado como base para instanciar o segundo marcador (marcador de controle) quando for feita inicializações padrões das classes.

LineMaterial: Material com o atributo de **`Shader`** configurado na opção de: **`Particles/Additive`**, e usado para a criação de linhas na classe **`ActivateInformation`**.

Material: Material de cor azul usado quando for feita a inicialização padrão da classe **`ChangeElement`**.

Pasta Gizmos

A pasta **`Gizmos`** (localizada fora da pasta **`AR Educational Framework`** por ser um nome de pasta especial para o Unity), contém ícones a serem usados na classe **`ActivateInformationI`**. No Unity, ícones usados devem estar contidos numa pasta de nome especial para o Unity chamada **`Gizmos`**, colocada diretamente dentro da pasta **`Assets`** (somente é possível ter uma pasta **`Gizmos`** no projeto). Nota-se que no caso de importar o *framework* em um projeto com a pasta **`Gizmos`** já existe, os recursos da pasta **`Gizmos`** do *framework* serão acrescentados na pasta **`Gizmos`** original.

Classes Do AR Educational Framework

A seguir, serão descritas em maior detalhe as classes da pasta **`MainScripts`** e **`EditMode`** do *framework,* que presentam grande relação entre elas e que são as classes com as quais o usuário vai interagir diretamente para usar as funcionalidades suportadas. As classes da pasta **`MainScripts`**, são as classes das funcionalidades fornecidas pelo *framework*, as quais o usuário deve manipular para levar a cabo as funcionalidades, lembrando que cada uma delas representa uma funcionalidade ou um conjunto de funcionalidades fornecidas pelo *framework* (descritas na Seção 4.1.3). Por outro lado, as classes contidas na pasta **`EditMode`**, permitem ao usuário realizar inicializações *default* das classes de **`MainScripts`**, portanto também se faz necessário aprofundar sobre cada uma delas, indicando a relação com sua classe associado da pasta **`MainScripts`**.

Além das classes anteriores, também se considerou importante apresentar a classe **`NewTrackableEventHandler`** contida na pasta **`OtherScripts`**, com alguns detalhes importantes sobre ela.

Classe `ButtonOperations`: Classe localizada na pasta **`MainScripts`** do *framework*, que permite através de um botão, mudar a *transform* de objetos 3D em RA (ou seja, rotacionar e escalar eles). Nota-se que a classe **`ButtonOperations`** permite a rotação e escala de objetos em RA através de botões, porém tendo o usuário que criar cada botão necessário um por um. Ver "Instruções Do Uso Da Classe **`ButtonOperations`**" em Apêndice C. Nota-se que as instruções do Apêndice C, apresentam a totalidade dos campos disponíveis de cada classe do *framework*, assim como sua utilidade, permitindo dar maiores detalhes e aprofundar sobre as classes.

Criação De Um Pacote De Botões Default: O *framework* oferece através da classe **`ButtonsPack`**, a possibilidade de criar automaticamente um pacote de botões padrão, ver mais detalhes na classe **`ButtonsPack`**. Em situações comuns (onde o usuário quer iniciar com um pacote de botões prontos), deveria ser usada a classe **`ButtonsPack`** para criar os botões rapidamente (ao invés de **`ButtonOperations`**), e através de **`ButtonOperations`** ser configuradas as preferências do usuário para as operações.

Classe ButtonsPack: Classe localizada na pasta **`MainScripts`**, que permite (fazendo uso da classe **`ButtonOperations`**), criar automaticamente um pacote de botões padrões com todos os botões necessários para mudar a *transform* de objetos 3D em RA (ou seja, rotacionar e escalar eles em todas as formas). Ver "Instruções Do Uso Da Classe **`ButtonsPack`**" em Apêndice C.

Inicialização Default Da Classe: O *framework* oferece através da classe **`ButtonsPackE`**, a possibilidade de criar um marcador e um objeto padrão para os botões, ver mais detalhes na classe **`ButtonsPackE`**.

Classe ButtonsPackE: Classe localizada na pasta **`EditMode`**, que permite criar automaticamente um marcador padrão, e um objeto cubo padrão (colocado como filho do marcador na hierarquia do Unity), vinculado eles a um pacote de

botões já criado com a classe **`ButtonsPack`**. Ver "Instruções Do Uso Da Classe **`ButtonsPackE`**" em Apêndice C.

Classe ChangeElement: Classe localizada na pasta **`MainScripts`** do *framework*, que permite mudar um objeto 3D em RA, atreves do uso de um marcador de controle (ou seja, colocando um marcador de controle junto do marcador principal vinculado ao objeto 3D). Podem ser mudadas características do objeto (mudar rotação e escala, ou os seus Materiais) ou ainda trocar o objeto por outro novo. Ver "Instruções Do Uso Da Classe **`ChangeElement`**" em Apêndice C.

Inicialização Default Da Classe: O *framework* oferece através da classe **`ChangeElementE`** a possibilidade de fazer uma inicialização padrão para uma das funcionalidades da classe **`ChangeElement`**, ver mais detalhes na classe **`ChangeElementE`**.

Classe ChangeElementE: Classe localizada na pasta **`EditMode`**, que permite criar automaticamente os elementos necessários para fazer uma inicialização padrão da classe **`ChangeElement`**, fazendo com que uma das funcionalidades dessa classe fique pronta para ser executada (no caso, a funcionalidade chamada "Suporte a modificação dos materiais de objetos através de marcadores de controle", descrita na Seção 4.1.3.2). Ver "Instruções Do Uso Da Classe **`ChangeElementE`**" em Apêndice C.

Classe ActivateInformation: Classe localizada na pasta **`MainScripts`** do *framework*, que permite ativar diferentes tipos de informações adicionais sobre um objeto 3D em RA, através do uso de um marcador de controle (ou seja, colocando um marcador de controle junto do marcador principal vinculado ao objeto 3D). As informações adicionais ativadas podem ser textos em 3D ou imagens em RA. Os textos ou imagens podem ainda ser ativados acompanhados de linhas que lhes permitam indicar pontos específicos do objeto. Ver "Instruções Do Uso Da Classe **`ActivateInformation`**" em Apêndice C.

Inicialização Default Da Classe: O *framework* oferece através da classe **`ActivateInformationE`** a possibilidade de fazer uma inicialização padrão para

uma das funcionalidades da classe **`ActivateInformation`**, ver mais detalhes na classe **`ActivateInformationE`**.

Classe ActivateInformationE: Classe localizada na pasta **`EditMode`**, que permite criar automaticamente os elementos necessários para fazer uma inicialização padrão da classe **`ActivateInformation`**, fazendo com que uma das funcionalidades dessa classe fique pronta para ser executada (no caso, a funcionalidade chamada "Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle", usando textos 3D como a informação adicional, descrita na Seção 4.1.3.2). Ver "Instruções Do Uso Da Classe **`ActivateInformationE`**" em Apêndice C.

Classe QuestionController1: Classe localizada na pasta **`MainScripts`**, que permite criar uma pergunta na qual as alternativas são elementos 2D (textos 2D ou imagens 2D) tipo UI tradicional, apresentados no mesmo tempo. Nota-se que uma vez que estes elementos 2D funcionarão como alternativas de resposta, então poderão ser selecionados como resposta pressionando sobre eles. Ver "Instruções Do Uso Da Classe **`QuestionController1`**" em Apêndice C.

Inicialização Default Da Classe: O *framework* oferece através da classe **`QuestionController1E`**, a possibilidade de fazer uma inicialização padrão para a classe **`QuestionController1`**, ver mais detalhes na classe **`QuestionController1E`**.

Classe QuestionController1E: Classe localizada na pasta **`EditMode`**, que permite criar automaticamente os elementos necessários para fazer uma inicialização padrão da classe **`QuestionController1`**, fazendo com que uma pergunta com elementos padrões de exemplo fique pronta para ser executada. Nota-se que a pergunta com elementos padrões criada, irá usar textos 2D como as alternativas, ao invés de imagens 2D, lembrando que são suportadas alternativas desses dois tipos com a classe **`QuestionController1`**. Ver "Instruções Do Uso Da Classe **`QuestionController1E`**" em Apêndice C.

Classe QuestionController2: Classe que permite criar uma pergunta na qual as alternativas são elementos 3D em RA (p.ex. modelos 3D) vinculados a um marcador, e apresentados por turno. Uma vez que estes elementos 3D funcionarão como alternativas de resposta, sendo apresentados por turno, são fornecidos botões para permitir explorar eles um por um, e selecionar o desejado como resposta. Ver "Instruções Do Uso Da Classe **`QuestionController2`**" em Apêndice C.

Inicialização Default Da Classe: O *framework* oferece através da classe **`QuestionController2E`** a possibilidade de fazer uma inicialização padrão para a classe **`QuestionController2`**, ver mais detalhes na classe **`QuestionController2E`**.

Classe QuestionController2E: Classe localizada na pasta **`EditMode`**, que permite criar automaticamente os elementos necessários para fazer uma inicialização padrão da classe **`QuestionController2`**, fazendo com que uma pergunta com elementos padrões de exemplo fique pronta para ser executada. Nota-se que a pergunta com elementos padrões criada, irá usar dois modelos de um cubo e uma esfera como alternativas, lembrando que com a classe **`QuestionController2,`** são suportadas alternativas através de qualquer tipo de elementos 3D em RA (p.ex. modelos, ou imagens em RA). Ver "Instruções Do Uso Da Classe **`QuestionController2`**E" em Apêndice C.

Classe NewTrackableEventHandler: Classe contida na pasta **`OtherScripts`**, que é utilizada pelos marcadores próprios do *framework* (localizados na pasta com o caminho: **`AR Educational Framework/Resources`**), ao invés da classe **`DefaultTrackableEventHandler`** que todo marcador da Vuforia possui por padrão (*default*). Na prática, para criar um marcador a ser usado com o *framework*, se o usuário utilizar o marcador do *framework* chamado **`ImageTarget1Prefab`** (tal como falado em "Configurar Marcadores Do Framework Na Cena" na fase "Configuração Do Framework" da Seção 4.2.1), ou se utilizar alguma inicialização padrão disponível, ele não precisará manipular a classe **`NewTrackableEventHandler`**, devido a que o marcador criado já terá esta classe inserida ao invés da **`DefaultTrackableEventHandler`** original, porém no caso de criar um marcador de jeito normal com

GameObject/Vuforia/Image, e querer usá-lo junto com o *framework*, então será necessário substituir sua classe **DefaultTrackableEventHandler** original pela classe **NewTrackableEventHandler** do *framework*.

NewTrackableEventHandler, consiste em uma versão customizada da classe **DefaultTrackableEventHandler** da Vuforia, realizando sobre ela alguns ajustes necessários. **DefaultTrackableEventHandler** é uma classe da Vuforia que permite manipular eventos relacionados com o rastreamento de marcadores (p.ex. definir eventos necessários quando um marcador está sendo visível ou não pela câmera), assim foi criada a classe **NewTrackableEventHandler** usando a classe original **DefaultTrackableEventHandler** como *template*, e realizando alguns ajustes para as necessidades particulares do *framework* proposto. O código necessário que foi acrescentado em **DefaultTrackableEventHandler** para criar **NewTrackableEventHandler** é discutido a seguir.

Na Figura B 1, as linhas 27 até 34 (região de código nomeada como AR_EDUCATIONAL_FRAMEWORK_VARIABLES), apresentam os novos atributos que foi necessário acrescentar, assim como suas respetivas propriedades. A variável booleana **markerFound**, é usada para indicar quando o marcador está sendo visível ou não pela câmera (seu valor será atribuído a partir dos métodos **OnTrackingFound** e **OnTrackingLost** da classe), e será acessada pelas outras classes do *framework* quando for necessário realizar eventos que dependem da visibilidade do marcador (p.ex. ativar eventos através de marcadores de controle). A lista renderers, é necessária para a funcionalidade de “Suporte à troca de objetos através de marcadores de controle”, (ver Seção 4.1.3.2), e será usada para manter componentes **Renderer** do Unity (de certos objetos), que queremos evitar que sejam ativados quando o marcador for visível pela câmera (ao invés disso, por diversas razões, o próprio *framework* será encarrado desse processo de ativação). Já a linha 11, tem um **Namespace** acrescentado para o uso de listas.

```
9   using UnityEngine;
10  using Vuforia;
11  using System.Collections.Generic;//EDITADO---
12
13  /// <summary>
14  /// A custom handler that implements the ITrackableEventHandler interface.
15  ///
16  /// Changes made to this file could be overwritten when upgrading the Vuforia version.
17  /// When implementing custom event handler behavior, consider inheriting from this class instead.
18  /// </summary>
19  public class NewTrackableEventHandler : MonoBehaviour, ITrackableEventHandler
20  {
21      #region PROTECTED_MEMBER_VARIABLES
22
23      protected TrackableBehaviour mTrackableBehaviour;
24
25      #endregion // PROTECTED_MEMBER_VARIABLES
26
27      #region AR_EDUCATIONAL_FRAMEWORK_VARIABLES
28
29      private bool markerFound;//EDITADO--- VARIAVEL USADA PARA INFORMAR ÀS OUTRAS CLASSES, SE O MARCADOR ESTIVER VIS
30      public bool MarkerFound { get { return markerFound; } }
31      private List<Renderer> renderers = new List<Renderer>();//EDITADO--- NECESSARIO PARA O PROCESSO DE TROCAR OBJET
32      public List<Renderer> Renderers { get { return renderers; } }
33
34      #endregion // AR_EDUCATIONAL_FRAMEWORK_VARIABLES
```

Figura B 1: Conjunto de novos atributos que foram acrescentados na classe `DefaultTrackableEventHandler` original, para criar `NewTrackableEventHandler`.

Na Figura B 2, apresenta-se o código acrescentado no método **`OnTrackingFound`** da classe **`DefaultTrackableEventHandler`** original. A linha 98 consiste em uma validação adicional que permite realizar o processo já falado de evitar ativar certos componentes **`Renderer`** (mantidos na lista renderers), nota-se que o **`DefaultTrackableEventHandler`** original sem essa validação, permitiria ativar todos os **`Renderer`** dos objetos associados a um marcador quando o marcador for visível pela câmera, portanto para o *framework* foi necessário acrescentar está linha para evitar que esse processo seja feito em certos **`Renderer`** de objetos. Já a linha 110, é a atribuição da variável **`markerFound`** para o valor **`True`**, quando o marcador for visível pela câmera.

```
89      protected virtual void OnTrackingFound()
90      {
91          var rendererComponents = GetComponentsInChildren<Renderer>(true);
92          var colliderComponents = GetComponentsInChildren<Collider>(true);
93          var canvasComponents = GetComponentsInChildren<Canvas>(true);
94
95          // Enable rendering:
96          foreach (var component in rendererComponents)
97          {
98              if (!Renderers.Contains(component))//EDITADO--- evita ativar os componentes renderers de certos objetos
99                  component.enabled = true;
100         }
101
102         // Enable colliders:
103         foreach (var component in colliderComponents)
104             component.enabled = true;
105
106         // Enable canvas':
107         foreach (var component in canvasComponents)
108             component.enabled = true;
109
110         markerFound = true;//EDITADO---
111     }
```

Figura B 2: Código acrescentado no método `OnTrackingFound` da classe `DefaultTrackableEventHandler` original, para criar `NewTrackableEventHandler`.

Finalmente na Figura B 3, apresenta-se o código acrescentado no método **`OnTrackingLost`** da classe **`DefaultTrackableEventHandler`** original, que consiste unicamente, na linha 132, a atribuição da variável **`markerFound`** para o valor **`False`**, quando o marcador não for visível pela câmera.

```
114     protected virtual void OnTrackingLost()
115     {
116         var rendererComponents = GetComponentsInChildren<Renderer>(true);
117         var colliderComponents = GetComponentsInChildren<Collider>(true);
118         var canvasComponents = GetComponentsInChildren<Canvas>(true);
119
120         // Disable rendering:
121         foreach (var component in rendererComponents)
122             component.enabled = false;
123
124         // Disable colliders:
125         foreach (var component in colliderComponents)
126             component.enabled = false;
127
128         // Disable canvas':
129         foreach (var component in canvasComponents)
130             component.enabled = false;
131
132         markerFound = false;//EDITADO---
133     }
```

Figura B 3: Código acrescentado no método `OnTrackingLost` da classe `DefaultTrackableEventHandler` original, para criar `NewTrackableEventHandler`.

Diagramas De Classes Do Framework

A seguir são apresentados os diagramas de classes do AR Educational Framework, tal como ele foi projetado, e indicando alguns pontos importantes. Em alguns dos diagramas foi necessário suprimir os métodos e atributos das classes para melhorar a visualização, porém essas mesmas classes são apresentadas de jeito independente com todos seus atributos e métodos.

Os diagramas serão apresentados organizados por pacotes de classes, inicialmente será apresentado o diagrama de classes geral do *framework* com todos seus pacotes, porém devido ao tamanho e complexidade, foi necessário também apresentar diagramas de classes de jeito independente para cada par de pacotes que estiver relacionado dentro do *framework* (o que permitirá uma melhor análise das relações entre os pacotes). Por último, serão apresentados diagramas de classes independentes para cada pacote de classes disponível (o que permitirá uma melhor análise das próprias classes de cada pacote).

Como falado, inicialmente apresenta-se o diagrama de classes geral do *framework* na Figura B 4. Neste diagrama foram suprimidos atributos e métodos das classes, e foram omitidos os relacionamentos das classes com outras classes externas tanto do Unity quanto da Vuforia, isso afim de evitar ainda maior complexidade no diagrama, porém os atributos e métodos, assim como relacionamentos com classes externas, serão incluídos nos diagramas de classes independentes para cada pacote, apresentados no final desta seção.

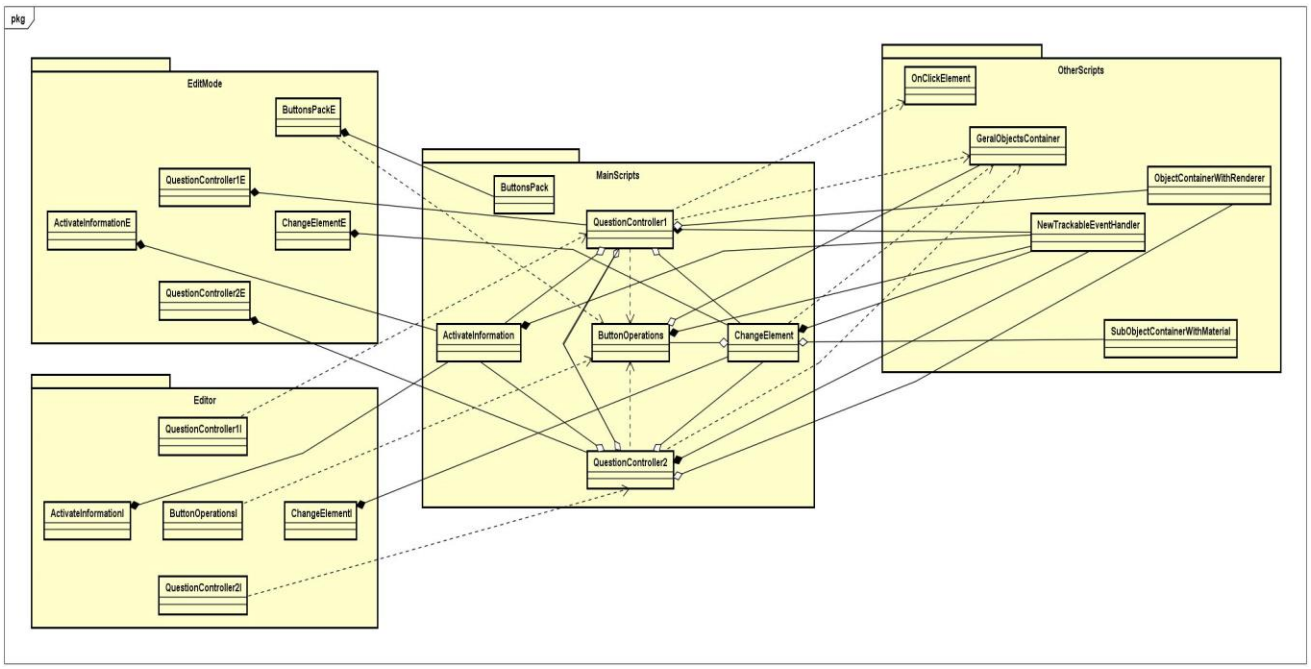

Figura B 4: Diagrama de classes geral do AR Educational Framework.

Devido à quantidade de relacionamentos entre classes, visando maior simplificação e permitir uma melhor análise, se achou adequado apresentar diagramas de classes de jeito independente para os pares de pacotes relacionados, que são:

- **`EditMode`** e **`MainScripts`**.
- **`Editor`** e **`MainScripts`**.
- **`OtherScripts`** e **`MainScripts`**.

A Figura B 5 apresenta o diagrama de classes de jeito independente dos pacotes **`EditMode`** e **`MainScripts`** com seus relacionamentos. Lembre-se que as classes de **`EditMode`** unicamente são usadas para fazer inicializações padrão das classes de **`MainScripts`**, portanto cada classe de **`EditMode`** está relacionada com

uma versão respetiva dela em **`MainScripts`**, que corresponde à classe sobre a qual faz a inicialização padrão.

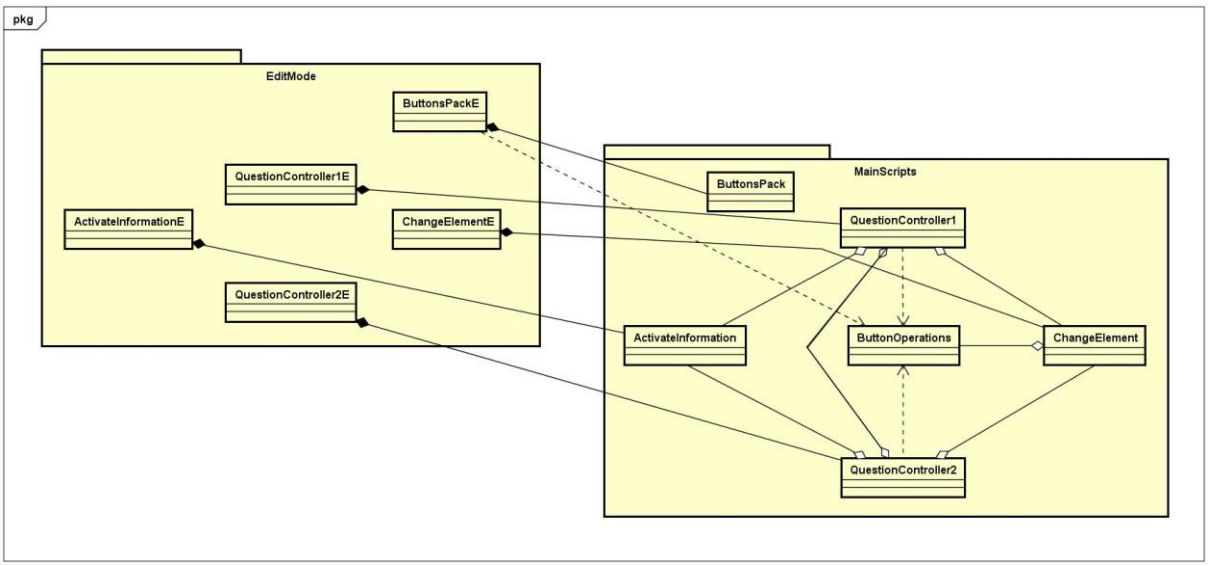

Figura B 5: Diagrama de classes de jeito independente dos pacotes `EditMode` e `MainScripts`.

A Figura B 6 apresenta o diagrama de classes de jeito independente dos pacotes **`Editor`** e **`MainScripts`** com seus relacionamentos. Lembre-se que as classes de **`Editor`** unicamente são usadas para customizar o inspetor do Unity para as classes de **`MainScripts`**, portanto cada classe de **`Editor`** está relacionada com uma versão respetiva dela em **`MainScripts`**, que corresponde à classe sobre a qual faz o processo de customização do inspetor do Unity.

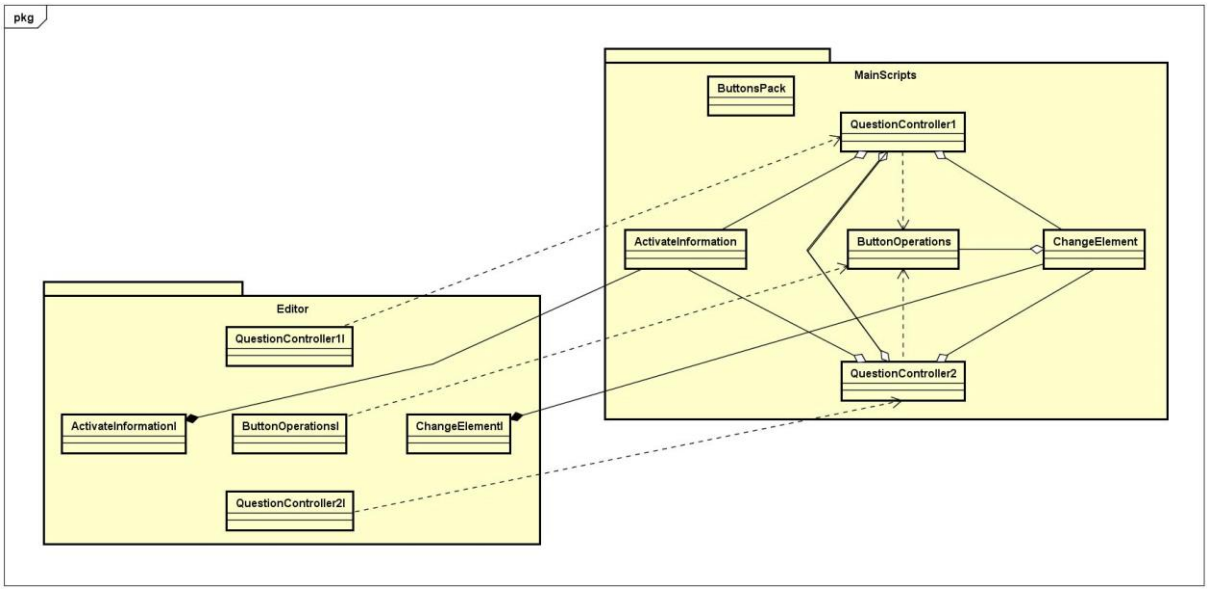

Figura B 6: Diagrama de classes de jeito independente dos pacotes `Editor` e `MainScripts`.

A Figura B 7 apresenta o diagrama de classes de jeito independente dos pacotes **`OtherScripts`** e **`MainScripts`** com seus relacionamentos. Lembre-se que as classes de **`OtherScripts`** chamadas: **`GeralObjectsContainer`**, **`ObjectContainerWithRenderer`**, e **`SubObjectContainerWithMaterial`**, unicamente são usadas para definir tipos de dados necessários em algumas classes da pasta **`MainScripts`**, assim no diagrama, podem ser vistas quais são as classes de **`MainScripts`** que precisam das três classes comentadas (e portanto, que tem algum tipo de relação com elas). Por outro lado, a maioria das classes de **`MainScripts`** (classes das funcionalidades do *framework)*, estão relacionadas com a classe **`NewTrackableEventHandler`** (classe que permite manipular eventos relacionados com marcadores) de **`OtherScritps`**, nota-se que isso foi necessário para permitir que as funcionalidades do *framework* possam realizar eventos que dependam da visibilidade do marcador pela câmera. Finalmente a classe **`QuestionController1`** está relacionada com a classe **`OnClickElement`**, devido a que os tipos de perguntas suportados em **`QuestionController1`** utilizam textos 2D e imagens como alternativas, as quais precisam ser pressionadas pelo usuário, e poder detectar seus clicks (lembre-se que a classe **`OnClickElement`** permite detectar clicks em elementos).

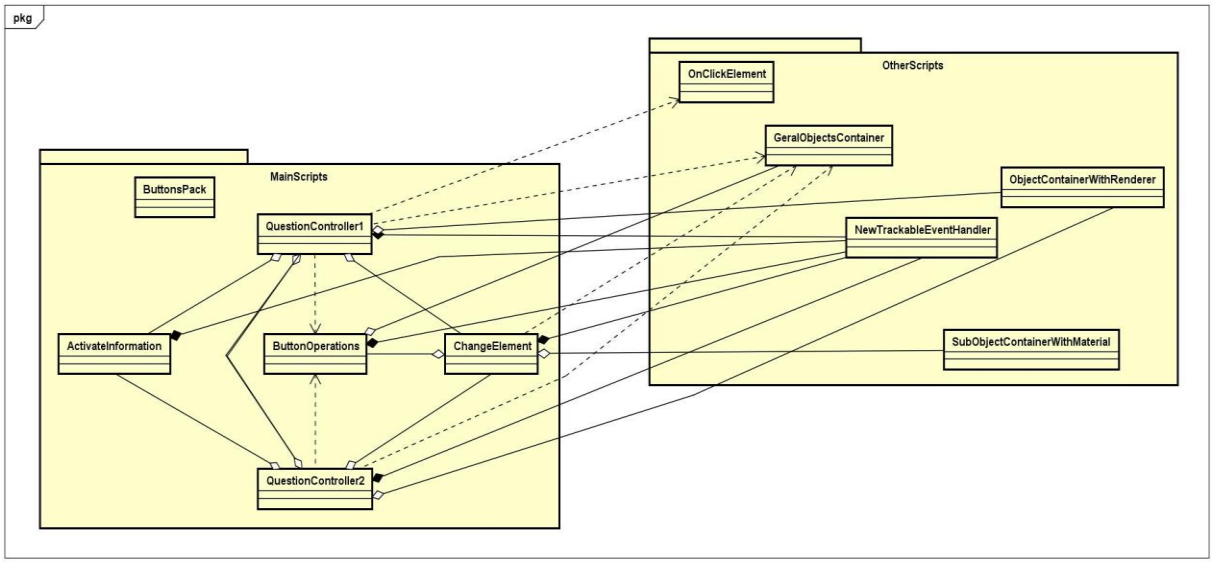

Figura B 7: Diagrama de classes de jeito independente dos pacotes `OtherScripts` e `MainScripts`.

Finalmente, a seguir são apresentados os diagramas de classes independentes para cada pacote de classes disponível dentro do *framework*, incluindo atributos e métodos das classes, assim como relacionamentos com outras classes externas do Unity e da Vuforia.

Na Figura B 8 apresenta-se o diagrama de classes do pacote **`EditMode`** do *framework*, como pode-se notar, todas as classes herdam da classe externa do Unity chamada **`MonoBehaviour`**, a qual consiste na classe principal para scripts no Unity.

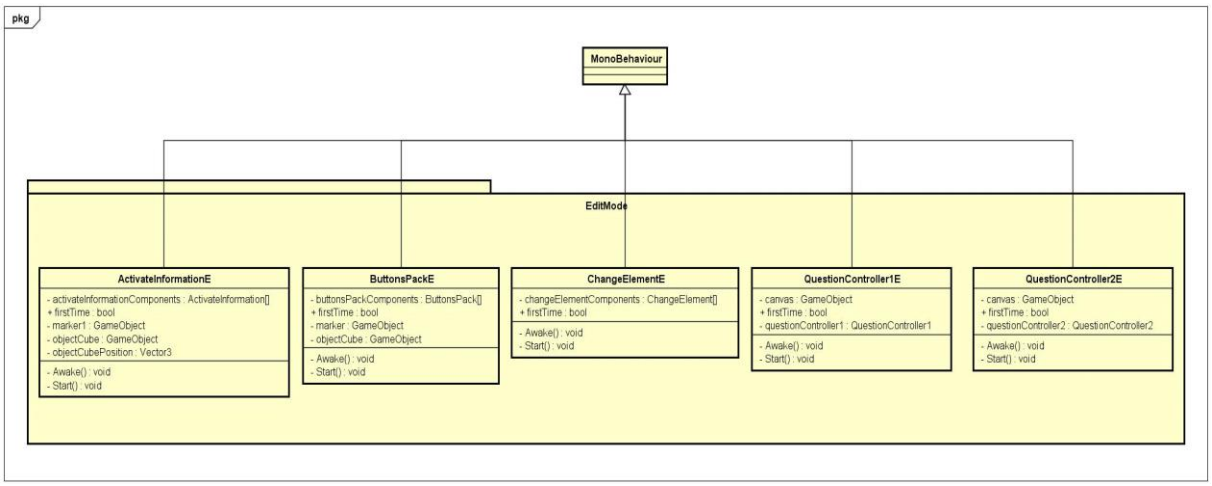

Figura B 8: Diagrama de classes do pacote `EditMode`.

Na Figura B 9 apresenta-se o diagrama de classes do pacote **`Editor`** do *framework*, neste pacote todas as classes herdam da classe externa do Unity chamada **`Editor`**, devido a que, como já falado, elas consistem em classes que permitem customizar o inspetor do Unity, sendo a classe **`Editor`**, a classe base do Unity utilizada para criar inspetores customizados.

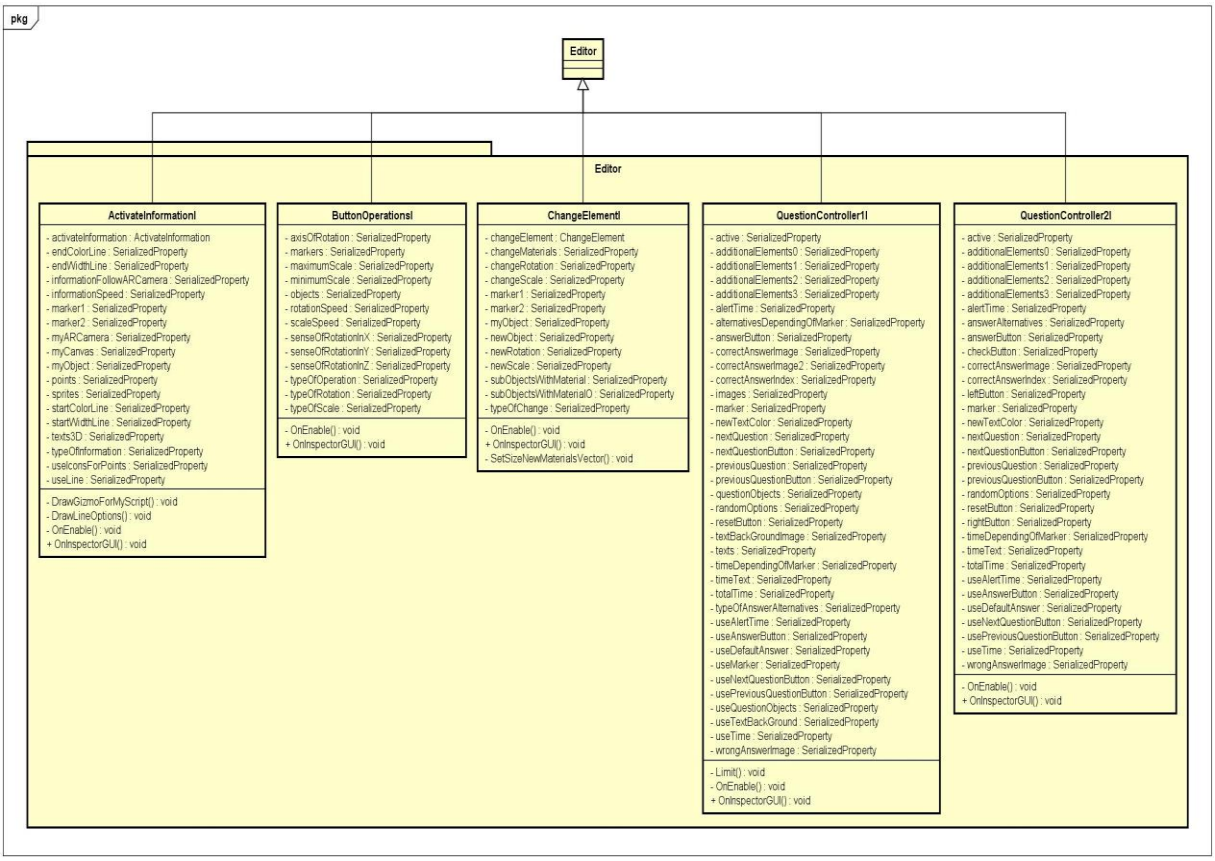

Figura B 9: Diagrama de classes do pacote `Editor`.

Na Figura B 10 apresenta-se o diagrama de classes do pacote **`MainScripts`** do *framework*, nota-se que as classes são apresentadas com seus atributos e métodos suprimidos, porém posteriormente cada uma delas serão apresentadas completas de jeito isolado. Neste pacote todas as classes herdam da classe externa do Unity **`MonoBehaviour`**, e já a classe **`ButtonOperations`** implementa as interfaces do Unity **`IPointerDownHandler`**, **`IPointerUpHandler`** e **`IPointerClickHandler`**, as quais permitem implementar métodos relacionados com o processo de detectar clicks do usuário sobre elementos do Unity (neste caso, necessários para serem usados sobre botões do Unity).

Como pode-se notar no diagrama de classes, existe uma grande quantidade de relacionamentos de diversos tipos entre as próprias classes do pacote **`MainScripts`**, deve-se lembrar que elas consistem nas classes das funcionalidades fornecidas pelo *framework*, e muitos desses relacionamentos se devem a que tais funcionalidades devem poder ser executadas tanto de jeito isoladas, quanto em conjunto, e ainda ser aplicadas sobre um mesmo conjunto de elementos (modelos 3D, marcadores etc.), portanto se faz necessário que todas as funcionalidades estejam prontas para serem executadas juntas, assim como para trabalhar de jeito coordenado, se comunicando entre elas para os processos que for necessários.

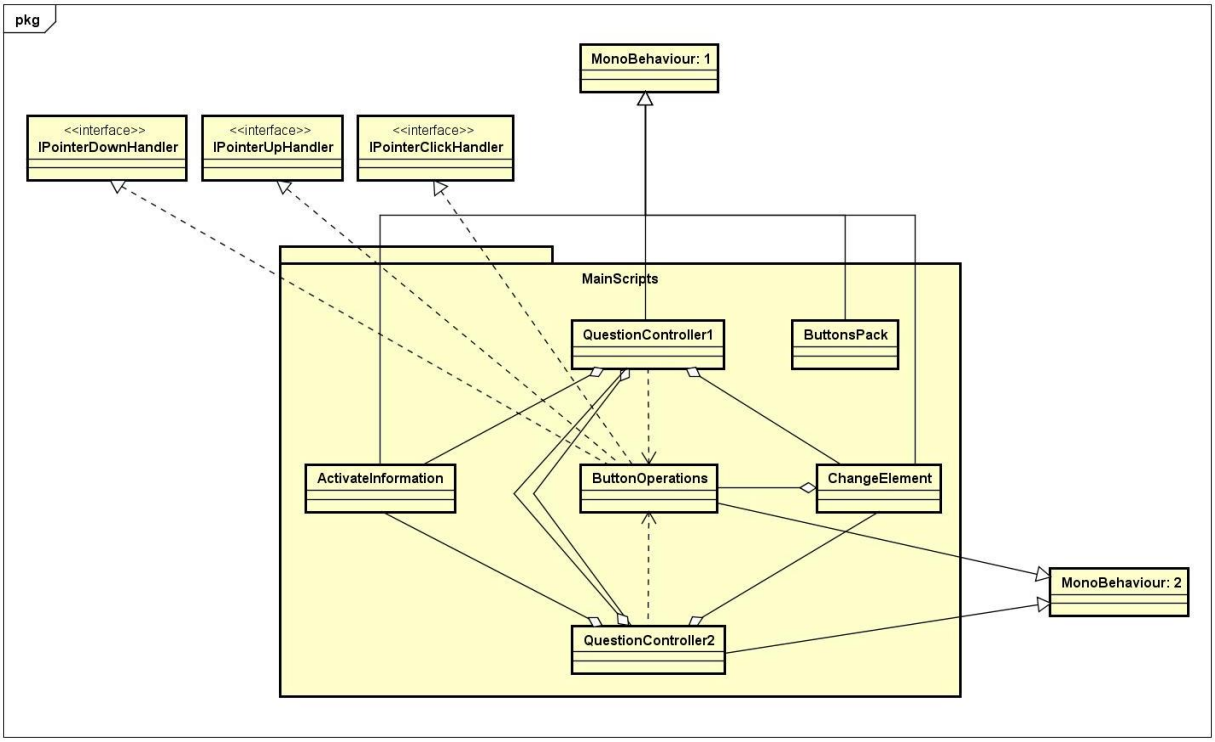

Figura B 10: Diagrama de classes do pacote `MainScripts`.

A seguir são apresentadas cada uma das classes do pacote **`MainScripts`**, com seus atributos e métodos, e de jeito isolado.

Figura B 11: Classe `ActivateInformation`.

Figura B 12: Classe `ButtonOperations`.

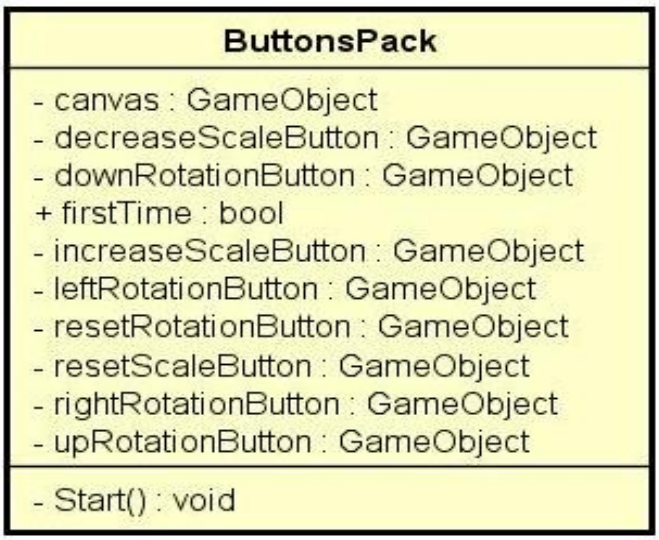

Figura B 13: Classe `ButtonsPack`.

Figura B 14: Classe `ChangeElement`.

Figura B 15: Classe `QuestionController1`.

QuestionController2

- activateInformationComponents : ActivateInformation[]
- active : bool
- actualIndex : int
- additionalElements0 : GameObject[]
- additionalElements0OriginalPosition : Vector3[]
- additionalElements0OriginalRotation : Quaternion[]
- additionalElements0OriginalScale : Vector3[]
- additionalElements1 : GameObject[]
- additionalElements1OriginalPosition : Vector3[]
- additionalElements1OriginalRotation : Quaternion[]
- additionalElements1OriginalScale : Vector3[]
- additionalElements2 : GameObject[]
- additionalElements2OriginalPosition : Vector3[]
- additionalElements2OriginalRotation : Quaternion[]
- additionalElements2OriginalScale : Vector3[]
- additionalElements3 : GameObject[]
- additionalElements3OriginalPosition : Vector3[]
- additionalElements3OriginalRotation : Quaternion[]
- additionalElements3OriginalScale : Vector3[]
- alertTime : int
- allObjects : List<GameObject>
- allObjectsWithRenderer : List<GameObject>
- answerAlternatives : List<GameObject>
- answerAlternativesOriginalPosition : Vector3[]
- answerAlternativesOriginalRotation : Quaternion[]
- answerAlternativesOriginalScale : Vector3[]
- answerButton : Button
- changeElementComponents : ChangeElement[]
- checkButton : Button
- correctAnswerImage : Image
- correctAnswerIndex : int
- correctAnswerObj : GameObject
- countDown : float
- emtyObjects : List<GameObject>
- firstFrameActive : bool
- firstFrameDesactive : bool
- firstTime : bool
- guiTime : float
- inscresedTotalTime : bool
- leftButton : Button
- marker : GameObject
- markerNewTrackableEventHandlerComponent : NewTrackableEventHandler
- newTextColor : Color
- nextQuestion : GameObject
- nextQuestionButton : Button
- objectsContainerWithRenderer : List<ObjectContainerWithRenderer>
- originalColor : Color
- originalObjects : List<GameObject>
- originalTotalTime : int
- paused : bool
- previousQuestion : GameObject
- previousQuestionButton : Button
- questionAnswered : bool
- questionController1Components : QuestionController1[]
- questionController2Components : QuestionController2[]
- randomOptions : bool
- resetButton : Button
- rightButton : Button
- starTime : float
- stopTime : bool
- stopTimeValue : float
- substituteObjects : List<GameObject>
- substituteObjectsOfAnswerAlternatives : List<GameObject>
- substituteObjectsOfAnswerAlternativesOriginalPosition : Vector3[]
- substituteObjectsOfAnswerAlternativesOriginalRotation : Quaternion[]
- substituteObjectsOfAnswerAlternativesOriginalScale : Vector3[]
- substituteObjectsOriginalPosition : Vector3[]
- substituteObjectsOriginalRotation : Quaternion[]
- substituteObjectsOriginalScale : Vector3[]
- timeDependingOfMarker : bool
- timeText : Text
- totalTime : int
- useAlertTime : bool
- useAnswerButton : bool
- useDefaultAnswer : bool
- useNextQuestionButton : bool
- usePreviousQuestionButton : bool
- useTime : bool
- wrongAnswerImage : Image

- AnswerButtonMethod() : void
- Awake() : void
- ChangeQuestion() : IEnumerator
- CheckButtonMethod() : void
- CorrectAnswer() : void
- CountDown() : void
- DisableElements() : void
- InitializeVectObjectsContainerWithRenderer() : void
- LateUpdate() : void
- LeftButtonMethod() : void
- ResetButtonMethod() : void
- ResetElementsTransform() : void
- ResetObjectsColor() : void
- ReshuffleList() : void
- RightButtontMethod() : void
- SaveObjectsOfArrayInAllObjects() : void
- SaveOriginalTransformOfElements() : void
- Start() : void
- StateQuestionAnswered() : void
- Update() : void
- WrongAnswer() : void

Figura B 16: Classe `QuestionController2`.

Na Figura B 17 apresenta-se o diagrama de classes do pacote **`OtherScripts`** do *framework*, neste pacote as classe **`NewTrackableEventHandler`** e **`OnClickElement`** herdam da classe externa do Unity **`MonoBehaviour`**. Lembre-se que as classes **`GeralObjectsContainer`**, **`ObjectContainerWithRenderer`**, e **`SubObjectContainerWithMaterial`**, são usadas unicamente para definir tipos de dados necessários para outras classes e portanto não precisaram herdar de **`MonoBehaviour`**. Além disso, a classe **`NewTrackableEventHandler`** implementa a interface da Vuforia **`ITrackableEventHandler`**, que permite manipular mudanças de estado de marcadores, e finalmente a classe **`OnClickElement`** implementa a interface do Unity **`IPointerDownHandler`**, que permitiu implementar um método para detectar clicks do usuário sobre elementos (neste caso, necessário para ser usado sobre textos 2D e imagens do Unity).

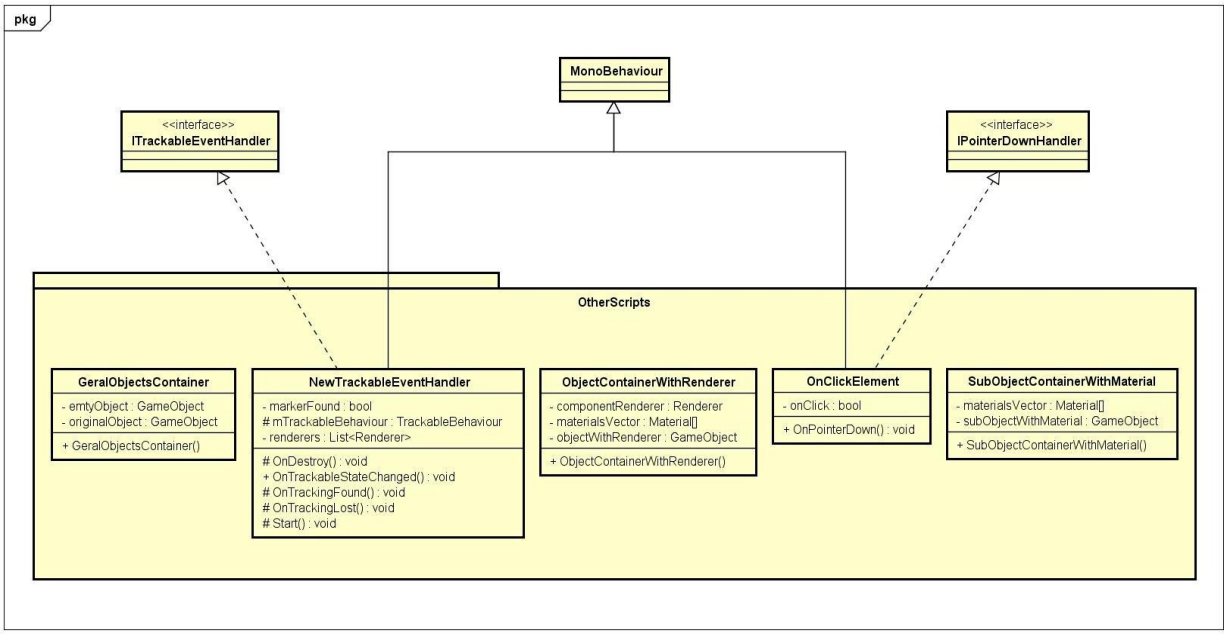

Figura B 17: Diagrama de classes do pacote `OtherScripts`.

Diagrama De Casos De Uso Geral Do Framework

A seguir apresenta-se o diagrama de casos de uso geral do AR Educational Framework, na Figura B 18.

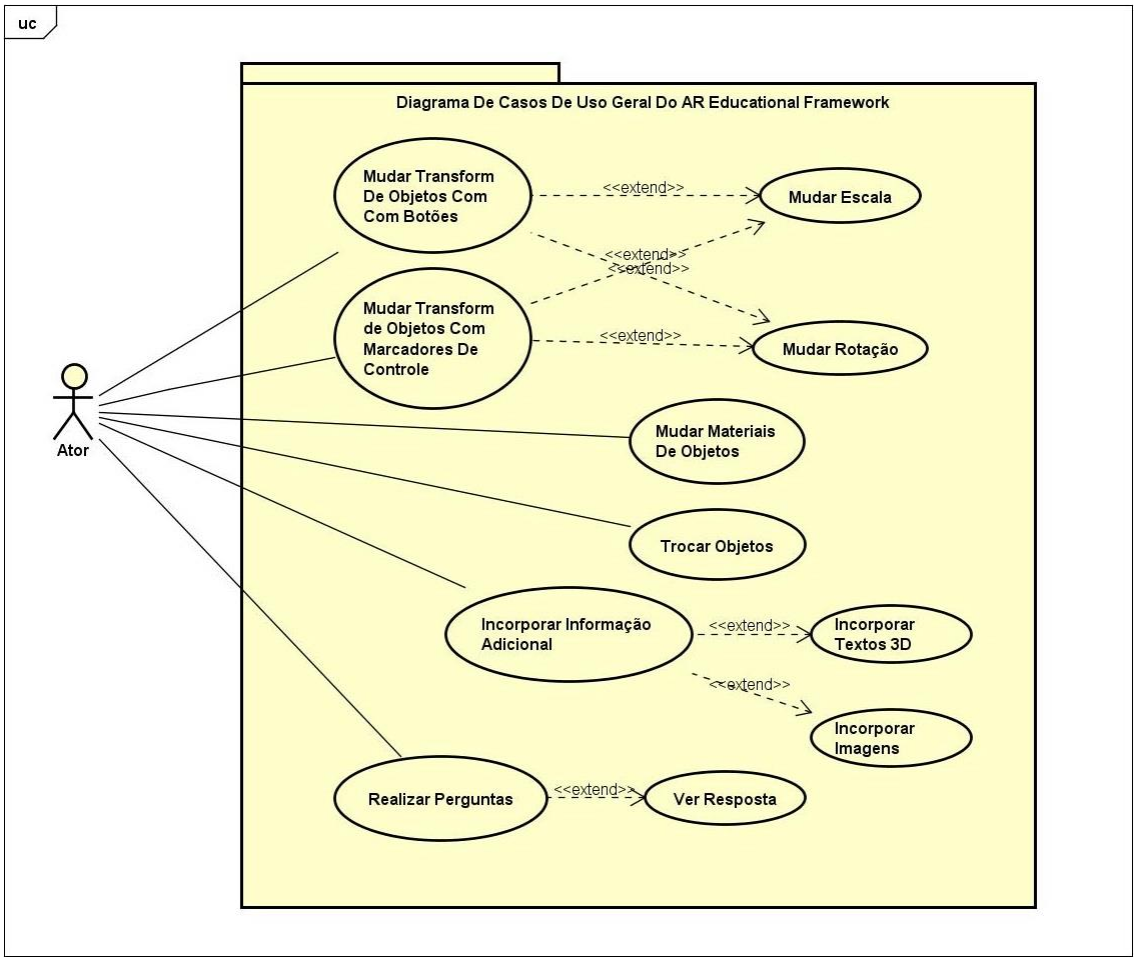

Figura B 18: Diagrama de casos de uso geral do AR Educational Framework.

O diagrama de casos de uso mostra os requisitos do AR Educational Framework, os quais são descritos a seguir:

Mudar Transform De Objetos Com Botões: Este requerimento é suportado pelo *framework* através da funcionalidade “Suportar a modificação da transform de objetos através de botões da UI (User Interface)”, descrita na Seção 4.1.3.1.

Mudar Transform De Objetos Com Marcadores De Controle: Este requerimento é suportado pelo *framework* através da funcionalidade “Suportar a modificação da transform de objetos através de marcadores de controle”, descrita na Seção 4.1.3.1.

Mudar Materiais De Objetos: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte a modificação dos materiais de objetos através de marcadores de controle”, descrita na Seção 4.1.3.2.

Trocar Objetos: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte à troca de objetos através de marcadores de controle”, descrita na Seção 4.1.3.2.

Incorporar Informação Adicional: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte à incorporação de informação adicional sobre objetos através de marcadores de controle”, descrita na Seção 4.1.3.2.

Realizar Perguntas: Este requerimento é suportado pelo *framework* através da funcionalidade “Suporte de Perguntas”, descrita na Seção 4.1.3.2.

Apêndice C

INSTRUÇÕES CLASSES DO AR EDUCATIONAL FRAMEWORK

A seguir, será explicado em detalhe como usar cada uma das classes das funcionalidades do *framework* (classes disponíveis na pasta **`MainScripts`**). Será explicado também como usar as classes que permitem realizar as inicializações padrão das funcionalidades (classes disponíveis na pasta **`EditMode`**). Cada elemento do Unity que for necessário criar, terá indicado entre parêntesis os passos para sua criação.

Instruções Do Uso Da Classe ButtonOperations: A classe deve ser arrastada sobre um botão da interface do Unity (**`GameObject/UI/Button`**) criado na cena pelo usuário. Os campos da classe que ficam visíveis para o usuário no inspetor, permitem definir a operação a ser realizada pelo botão e configurar as preferências do usuário, e em vários desses campos devem ser arrastados os elementos necessários para levar a cabo a operação (marcadores, e elementos 3D). Nota-se que este processo deveria ser repetido para cada botão necessário na cena.

A seguir, são explicados em detalhe os campos da classe e seu uso, nota-se que no caso de não fazer a criação do pacote de botões *default* (disponível através de **`ButtonsPack`**), todos os elementos que for necessários deverão ser criados e posicionados pelo usuário:

- TypeOfOperation: Permite escolher o tipo de operação que vai ser feita sobre o objeto quando o botão for pressionado, entre duas opções possíveis. Scale permite mudar o tamanho do objeto, e Rotate permite rotacionar ele.
- Objects: Através de Size, deve ser definida a quantidade de objetos que vão ser mudados com o botão atual, cada objeto deve estar colocado como filho (na hierarquia do Unity) de algum marcador, e ser arrastado em um dos campos Element disponíveis.
- Markers: Nos campos Element, devem ser arrastados os marcadores vinculados a cada objeto respetivamente (ou seja, o marcador que tem como filho na hierarquia do Unity ao objeto respetivo).

Opções com Scale selecionada em TypeOfOperation:

- TypeOfScale: Permite definir o tipo de escala que vai ser feita entre três opções possíveis. Increase permite aumentar o tamanho dos objetos quando o botão for pressionado, Decrease permite diminuir seu tamanho, e ResetScale permite restabelecer ao tamanho original.

  Opções com Increase selecionada em TypeOfScale:

  - ScaleSpeed: Permite definir a velocidade para aumentar o tamanho dos objetos.
  - MaximumScale: Permite definir a escala máxima até onde os objetos podem ser aumentados, este valor deve ser maior que 1. O valor *default* 2 significa o dobro do tamanho original do objeto.

  Opções com Decrease Selecionada em TypeOfScale:

  - ScaleSpeed: Permite definir a velocidade para diminuir o tamanho dos objetos.
  - MinimumScale: Permite definir a escala mínima até onde os objetos podem ser diminuídos, este valor deve ser maior que 0 e menor que 1. O valor *default* 0,5 significa a metade do tamanho original do objeto.

Opções com Rotate selecionada em TypeOfOperation:

- TypeOfRotation: Permite definir o tipo de rotação que vai ser feita entre duas opções possíveis. InAxis permite definir uma rotação em um dos eixos, e ResetRotation permite restabelecer à rotação original.

  Opções com InAxis Selecionada em TypeOfRotation:

  - RotationSpeed: Permite definir a velocidade para rotacionar os objetos.
  - Axis Of Rotation: Permite escolher o Eixo de Rotação (X, Y ou Z).

    Opções com X selecionada em AxisOfRotation:

    - SenseOfRotationInX: Permite escolher o sentido da rotação no Eixo X entre as duas opções possíveis. Up permite rotacionar o objeto para acima e Down para abaixo.

    Opções com Y selecionada em AxisOfRotation:

    - SenseOfRotationInY: Permite escolher o sentido da rotação no Eixo Y entre as duas opções possíveis. Left permite rotacionar para a esquerda e Right para a direita.

    Opções com Z selecionada em AxisOfRotation:

    - SenseOfRotationInZ: Permite escolher o sentido da rotação no Eixo Z entre as duas opções possíveis. Positive permite rotacionar o objeto no sentido positivo, e Negative no sentido negativo.

Instruções Do Uso Da Classe ButtonsPack: A classe deve ser arrastada sobre um objeto do Unity. Uma vez criado o pacote de botões, recomenda-se remover esta classe. Nota-se que neste caso, é através dos campos da classe **`ButtonOperations`** (disponível em cada botão), que serão configuradas as preferências do usuário para as operações dos botões.

Vincular Botões A Estados Da Pergunta: No caso de usar estes botões em contextos de perguntas (ou seja, usando a classe **`QuestionController1`** ou **`QuestionController2`**), é possível fazer que os botões apareçam unicamente em

alguns dos estados da pergunta, isso é feito arrastando eles nos campos AdditionalElements da classe **`QuestionController1`** ou **`QuestionController2`** da pergunta (ou seja, usando os botões como elementos adicionais), ver mais detalhes nos campos **`AdditionalElements`** das classes **`QuestionController1`** ou **`QuestionController2`**.

Instruções Do Uso Da Classe ButtonsPackE: A classe deve ser arrastada em um objeto do Unity que já possui a classe **`ButtonsPack`**, mantendo os botões do pacote já criado, com os campos da classe **`ButtonOperations`** por *default*. Após fazer isso, recomenda-se remover as duas classes **`ButtonsPack`** e **`ButtonsPackE`**.

Finalmente, um tutorial de exemplo (que inclui imagens de capturas de tela), usando a classe **`ButtonsPack`** e sua inicialização *default* com **`ButtonsPackE`**, é fornecido para maior compreensão em “Tutorial Classe **`ButtonsPack`**” em Apêndice D. Nota-se que o tutorial apresenta como usar **`ButtonsPack`** de um jeito isolado, já um tutorial para usar **`ButtonsPack`** junto com perguntas criadas com o *framework* (usando **`QuestionController1`** ou **`QuestionController2`**), é fornecido em “Tutorial Uso Classe **`ButtonsPack`** Em Perguntas”. Esse último tutorial, também inclui os passos para vincular botões a estados da pergunta, fazendo eles aparecer só em certos estados da pergunta.

Instruções Do Uso Da Classe ChangeElement: A classe deve ser arrastada sobre um objeto do Unity. Os campos da classe que ficam visíveis para o usuário no inspetor, permitem definir a operação a ser realizada, e em vários deles devem ser arrastados os elementos necessários (marcadores, elementos 3D etc.), que deverão ser criados e posicionados na cena do Unity pelo usuário.

A seguir, são explicados em detalhe os campos da classe e seu uso, nota-se que no caso de não fazer a inicialização *default* (disponível através de **`ChangeElementE`**), todos os elementos que for necessários deverão ser criados e posicionados pelo usuário:

- Marker1: Neste campo arrasta-se o marcador principal vinculado ao objeto, este é o marcador que vai ter como filho (na hierarquia do Unity) o objeto que quer se mudar.
- Marker2: Neste campo arrasta-se o marcador de controle, o qual vai permitir mudar o objeto quando estiver junto do outro marcador.
- MyObject: Neste campo arrasta-se o objeto que vai ser mudado quando os dois marcadores estiverem juntos, este objeto deve ser colocado como filho do marcador arrastado em Marker1.
- TypeOfChange: Permite escolher o tipo de mudança que vai ser feita no objeto entre duas opções possíveis. ChangeObjectProperties permite fazer mudanças nas propriedades do objeto, e ChangeObject permite trocar o objeto por outro novo.

Opções com ChangeObjectProperties selecionada em TypeOfChange:

- ChangeScale: Quando esta opção estiver ativada, permite mudar a escala atual do objeto por uma nova escala definida.
  Campos com ChangeScale ativada:
  - NewScale: Permite definir a nova escala do objeto em cada eixo.
- ChangeRotation: Quando esta opção estiver ativada, permite mudar a rotação atual do objeto por uma nova rotação definida.
  Campos com ChangeRotation ativada:
  - NewRotation: Permite definir a nova rotação do objeto em cada eixo.
- ChangeMaterials: Quando esta opção estiver ativada, permite mudar os Materiais do objeto por novos Materiais criados. Nota: A mudança de Materiais de um objeto se faz através dos seus sub-objetos com Material, que são os objetos filhos (na hierarquia do Unity) que possuem diretamente um componente MeshRenderer com Materiais do objeto principal, assim o usuário deverá indicar aqueles sub-objetos (ou filhos) nos que quer mudar materiais. No caso de usar objetos padrões do Unity (p.ex. cubos, esferas etc., que possuem todos seus Materiais em seu componente MeshRenderer diretamente e não em sub-

objetos), então o mesmo objeto deve ser usado como o único sub-objeto com Material. As instruções para ambos os casos comentados serão detalhadas a seguir.

Campos com ChangeMaterials ativada:

- SubObjectsWithMaterial: Através de Size permite definir a quantidade de sub-objetos com Material, dos quais se quer mudar materiais. Como falado, no caso de usar um objeto padrão do Unity, Size deve ficar com tamanho um.
  Para cada sub-objeto com Material, fica disponível um campo Element com os seguintes sub-campos:
- SubObjectWithMaterial: Neste campo arrasta-se o sub-objeto com Material do qual se quer mudar Materiais. No caso de usar um objeto padrão do Unity, então deve ser arrastado neste campo o próprio objeto, usando ele como o único sub-objeto com Material.
- MaterialsVector: Nos campos Element disponíveis dentro de MaterialsVector, arrastam-se novos Materiais criados pelo usuário que vão substituir os Materiais originais do sub-objeto com Material. É importante se notar que no caso de não querer mudar todos os Materiais disponíveis de um sub-objeto, então Materiais que não se querem mudar deverão ficar com seus campos Element vazios.

Opções com ChangeObject selecionada em TypeOfChange:

- NewObject: Neste campo arrasta-se o novo objeto que vai substituir o objeto original arrastado em MyObject. Este objeto deve ser posicionado no lugar desejado onde irá aparecer (normalmente no mesmo lugar do objeto original), e ser colocado como filho do marcador em Marker1.

Instruções Do Uso Da Classe ChangeElementE: A classe deve ser arrastada em um objeto do Unity que já possui a classe **`ChangeElement`**, mantendo esta última classe com todos seus campos por *default*. Finalmente após ser feita a inicialização, recomenda-se remover a classe **`ChangeElementE`**.

Finalmente, um tutorial de exemplo, usando a classe **`ChangeElement`** e sua inicialização *default* com **`ChangeElementE`**, é fornecido para maior compreensão em "Tutorial Classe **`ChangeElement`**" em Apêndice D. Nota-se que o tutorial apresenta como usar **`ChangeElement`** de um jeito isolado, já um tutorial para usar **`ChangeElement`** junto com perguntas criadas com o *framework* (usando **`QuestionController1`** ou **`QuestionController2`**), é fornecido em "Tutorial Uso Classe **`ChangeElement`** Em Perguntas".

Instruções Do Uso Da Classe ActivateInformation: A classe deve ser arrastada sobre um objeto do Unity. Os campos da classe que ficam visíveis para o usuário no inspetor, permitem definir a operação a ser realizada e configurar as preferências do usuário, e em vários deles, devem ser arrastados elementos necessários (marcadores, elementos 3D etc.), que deverão ser criados e posicionados na cena do Unity pelo usuário.

A seguir, são explicados em detalhe os campos da classe e seu uso, nota-se que no caso de não fazer a inicialização *default* (disponível através de **`ActivateInformationE`**), todos os elementos que for necessários deverão ser criados e posicionados pelo usuário:

- Marker1: Neste campo arrasta-se o marcador principal vinculado ao objeto, este é o marcador que vai ter como filho (na hierarquia do Unity) o objeto sobre o qual vai ser ativada informação adicional.
- Marker2: Neste campo arrasta-se o marcador de controle, que vai permitir ativar a informação adicional sobre o objeto quando estiver junto do outro marcador.
- MyObject: Neste campo arrasta-se o objeto sobre o qual vai ser ativada a informação adicional. Este objeto deve ser colocado como filho do marcador arrastado em Marker1.
- InformationFollowARCamera: Quando esta opção estiver ativada, permite que a informação adicional a ser ativada sobre o objeto, possa

seguir automaticamente a orientação da câmera ficando sempre legível.

Opções com InformationFollowARCamera ativada:

- MyARCamera: Neste campo arrasta-se a **`ARCamera`** do Unity que está sendo usada na cena atual.
- InformationSpeed: Este campo permite definir a velocidade da informação adicional para seguir a orientação da Camera.

- TypeOfInformation: Permite escolher o tipo de informação adicional que vai ser ativada quando os dois marcadores estiverem juntos, entre três opções possíveis. Texts3D suporta o uso de textos em 3D através dos elementos 3D Text do Unity (**`GameObject/3D Object/3D Text`**). Sprites suporta o uso de imagens em RA através dos elementos Sprites do Unity (**`GameObject/2D Object/Sprite`**) e finalmente CanvasOfImages é uma opção adicional para suportar o uso de imagens em RA, porém através de um elemento **`Canvas`** do Unity (**`GameObject/UI/Canvas`**), configurado como *World Space* na sua opção *Render Mode*, e que tem como filhos elementos Image do Unity (**`GameObject /UI/Image`**) usados como as imagens em 3D.

  Opções com Texts3D selecionada em TypeOfInformation:

  - Texts3D: Através de Size, deve ser definida a quantidade de elementos 3D Text do Unity a serem usados, os quais devem ser arrastados nos campos Element disponíveis.

  Opções com Sprites selecionada em TypeOfInformation:

  - Sprites: Através de Size, deve ser definida a quantidade de elementos Sprite do Unity a serem usados, os quais devem ser arrastados nos campos Element disponíveis.

  Opções com CanvasOfImages selecionada em TypeOfInformation:

  - MyCanvas: Neste campo deve ser arrastado o elemento **`Canvas`**, já configurado como *World Space* na sua opção *Render Mode*, e tendo como filhos unicamente os elementos Image do Unity a ser usados como informação adicional.
- UseLine: Quando esta opção estiver ativada, permite usar linhas que irão desde um ponto de origem no objeto, definido pelo usuário, até os

elementos usados como informação adicional (3D Texts, Sprites ou Images), sendo disponível uma linha para cada um desses elementos. Os pontos de origem respectivos deverão ser definido nos campos Element do atributo **`Points`**, arrastando GameObjects vazios do Unity (**`GameObject/Create Empty`**) que representarão eles.

Opções com UseLine selecionada:

- Points: Permite definir o ponto de origem para cada uma das linhas. Assim, nos campos Element disponíveis, devem ser arrastados GameObjects vazios do Unity que representarão esses pontos, e deverão ser posicionados no lugar desejado do objeto principal.
- UseIconForPoints: Quando esta opção estiver ativada, permite mostrar ícones nos GameObjects vazios que representam os pontos, visando fazer eles visível para o usuário e facilitar sua localização enquanto cria a cena. Sugere-se desativar a opção para ícones 3D no projeto (**`Gizmos/3D Icons`**) para melhor visibilidade. Nota-se que estes ícones só serão visíveis na visão da cena do Unity como ajuda visual, e não na execução. Ícones próprios do Unity também podem ser usados ao invés destes.
- StartWidthLine: Permite definir a largura inicial das linhas.
- EndWidthLine: Permite definir a largura final das linhas.
- StartColorLine: Permite definir a cor inicial das linhas.
- EndColorLine: Permite definir a cor final das linhas.

Instruções Do Uso Da Classe ActivateInformationE: A classe deve ser arrastada em um objeto do Unity que já possui a classe **`ActivateInformation`**, mantendo esta última classe com todos seus campos por *default*. Finalmente após ser feita a inicialização, recomenda-se remover a classe **`ActivateInformationE`**.

Finalmente, um tutorial de exemplo, usando a classe **`ActivateInformation`** e sua inicialização *default* com **`ActivateInformationE`**, é fornecido para maior compreensão em "Tutorial Classe **`ActivateInformation`**" em Apêndice D. Nota-se que o tutorial apresenta

como usar **`ActivateInformation`** de um jeito isolado, já um tutorial para usar **`ActivateInformation`** junto com perguntas criadas com o *framework* (usando **`QuestionController1`** ou **`QuestionController2`**), é fornecido em "Tutorial Uso Classe **`ActivateInformation`** Em Perguntas".

Instruções Do Uso Da Classe QuestionController1:

Para cada pergunta necessária, a classe deve ser arrastada sobre um objeto do Unity independente (cada instância da classe deve ficar em um GameObject independente), assim recomenda-se para cada pergunta, criar um novo GameObject vazio do Unity (**`GameObject/Create Empty`**) e arrastar nele a classe. Os campos da classe que ficam visíveis para o usuário no inspetor, permitem configurar as preferências do usuário para cada pergunta, e em vários deles, devem ser arrastados elementos necessários (marcadores, botões, textos etc.), que deverão ser criados e posicionados na cena do Unity pelo usuário, e a classe será encarregada do seu gerenciamento.

A seguir, são explicados em detalhe os campos da classe e seu uso, nota-se que no caso de não fazer a inicialização *default* (disponível através de **`QuestionController1E`**), todos os elementos que for necessários deverão ser criados e posicionados pelo usuário:

- Active: Permite ativar ou desativar a pergunta com todos seus elementos relacionados. Sempre que for usadas várias perguntas numa mesma cena do Unity, esta opção será usada para indicar qual será a primeira delas, deixando seu valor em **`True`**, e colocando o valor das outras perguntas em **`False`**.
- UseMarker: Quando esta opção estiver ativada, permite disponibilizar as opções da classe orientadas à utilização de um marcador na pergunta, o qual pode ser de utilidade principalmente para mostrar elementos em RA sobre os quais fazer a pergunta. Nota-se que estes objetos em RA poderiam ser usados como objetos pergunta ou elementos adicionais, arrastando eles nos campos QuestionObjects ou

AdditionalElements respetivamente, ver mais detalhes nos campos QuestionObjects e AdditionalElements.

Opções com UseMarker ativada:

- Marker: Neste campo arrasta-se o marcador usado quando for ativada a opção UseMarker.
- UseQuestionObjects: Quando esta opção estiver ativada, permite usar objetos nomeados “objetos pergunta”, que são objetos em RA vinculados ao marcador e sobre os quais vai ser realizada a pergunta. Estes objetos tem a característica de serem ativos nos estados quando a pergunta está sendo respondida, e quando está sendo apresentada a resposta certa, por outro lado, eles serão restabelecidos a seu estado original quando a pergunta for reiniciada, e ainda quando se passa ao estado de ver a resposta certa.

  Opções com UseQuestionObjects ativada:

  - QuestionObjects: Através de Size, deve ser definida a quantidade de objetos pergunta (comumente precisa-se só de um), os quais devem ser arrastados neste campo, e colocados como filhos (na hierarquia do Unity) do marcador. Nota: Objetos pergunta já tem seus estados pré-definidos pelo *framework* onde aparecerão na pergunta, portanto no caso de precisar definir o estado dos objetos a usar, sugere-se utilizar esses objetos como elementos adicionais ao invés de como objetos pergunta, ver mais detalhes nos campos AdditionalElements.
- AlternativesDependingOfMarker: Quando esta opção estiver ativada, permite que as alternativas de resposta dependam da visibilidade do marcador, sendo ativadas sempre que o marcador estiver visível pela câmera, e desativadas no caso contrário.

- TypeOfAnswerAlternatives: Permite escolher o tipo de alternativas de resposta a usar entre duas opções possíveis. Texts suporta o uso de textos em 2D através dos elementos Text do Unity

(**GameObject/UI/Text**), e Images suporta o uso de imagens em 2D através dos elementos Image do Unity (**GameObject/UI/Image**).

Opções com Texts selecionada em TypeOfAnswerAlternatives:

- Texts: Através de Size, deve ser definida a quantidade de elementos Text do Unity que vão ser usados como alternativas de resposta, e eles devem ser arrastados nos campos Element disponíveis.
- UseTextBackGround: Quando esta opção estiver ativada, permite usar uma imagem como fundo dos textos de alternativas para melhorar sua legibilidade quando for necessário.

  Opções com UseTextBackGround ativada:

  - TextBackGroundImage: Neste campo deve ser arrastado o elemento Image do Unity que vai ser usado como fundo dos textos.

Opções com Images selecionada em TypeOfAnswerAlternatives:

- Images: Através de Size, deve ser definida a quantidade de elementos Image do Unity que vão ser usados como alternativas de resposta, e eles devem ser arrastados nos campos Element disponíveis.

Nota: Elementos Text ou Image usados como alternativas de resposta devem ser independentes para cada pergunta, não sendo suportado seu reuso em várias delas.

- CorrectAnswerIndex: Permite definir o índice do elemento que representa a alternativa certa.
- ResetButton: Neste campo arrasta-se um botão da interface do Unity (**GameObject/UI/Button**), que permitirá repetir a pergunta após ter sido respondida. O processo de fazer reset numa pergunta inclui restabelecer ao estado original a *transform* (rotação e escala) e os Materiais dos objetos associados com a pergunta (ainda elementos adicionais que estiverem sendo utilizados, ver campos AdditionalElements).
- UseNextQuestionButton: Quando esta opção estiver ativada, permite usar um botão para passar a uma próxima pergunta.

  Opções com UseNextQuestionButton ativada:

- NextQuestionButton: Neste campo arrasta-se o botão que permitirá passar à próxima pergunta.
- NextQuestion: Neste campo arrasta-se o GameObject que tem acrescentado o **`QuestionController1`** (ou **`QuestionController2`**) da próxima pergunta.

- UsePreviousQuestionButton: Quando esta opção estiver ativada, permite usar um botão para passar a uma pergunta anterior.

  Opções com UsePreviousQuestionButton ativada:

  - PreviousQuestionButton: Neste campo arrasta-se o botão que permitirá passar à pergunta anterior.
  - PreviousQuestion: Neste campo arrasta-se o GameObject que tem acrescentado o **`QuestionController1`** (ou **`QuestionController2`**) da pergunta anterior.
- RandomOptions: Quando esta opção estiver ativada, permite que as alternativas sejam apresentadas em jeito aleatório.
- CorrectAnswerImage: Neste campo arrasta-se um elemento Image do Unity (**`GameObject/UI/Image`**), que vai ser usado como a imagem de *feedback* de resposta certa.
- WrongAnswerImage: Neste campo arrasta-se um elemento Image do Unity, que vai ser usado como a imagem de *feedback* de resposta errada.
- UseTime: Quando esta opção estiver ativada, permite definir um tempo limite para responder a pergunta. Uma vez que for usado, fica também relacionado com a resolução da pergunta, assim, finalizar o tempo faz com que a pergunta seja errada.

  Opções com UseTime e UseMarker ativada:

  - TimeDependingOfMarker: Quando esta opção estiver ativada, permite que o tempo dependa da visibilidade do marcador, correndo normalmente sempre que o marcador estiver visível pela câmera, e ficando pausado no caso contrário.

  Opções com UseTime ativada:

  - TotalTime: Permite definir o tempo total ou máximo (em segundos) para responder a pergunta.

- TimeText: Neste campo arrasta-se o elemento Text (**`GameObject/UI/Text`**), que será usado como o texto do tempo.
- UseAlertTime: Quando esta opção estiver ativada, permite que a partir de certo tempo, mude a cor do texto do tempo para indicar que está esgotando.

  Opções com UseAlertTime ativada:

  - AlertTime: Permite definir o tempo a partir do qual vai ser mudada a cor do texto do tempo.
  - NewTextColor: Permite definir a nova cor do texto do tempo pela qual vai ser mudada a cor original.

Campos AdditionalElements: Os campos AdditionalElements, são usados para suportar elementos adicionais, que são qualquer outros tipos de elementos (p.ex. botões, imagens, textos, elementos em RA etc.), que o usuário precisar utilizar em algum dos estados da pergunta e que devem ser gerenciados independentemente pelo usuário, excetuando o processo de ativação e desativação que já será feito pela classe. Estes elementos serão restabelecidos a seu estado original quando a pergunta for reiniciada. São permitidos diferentes tipos de elementos adicionais dependendo do seu estado de ativação no fluxo da pergunta, eles são: AdditionalElements0, AdditionalElements1, AdditionalElements2 ou AdditionalElements3, e serão explicados mas para frente.

Usar objetos como Elementos Adicionais (AdditionalElements): Para usar objetos como elementos adicionais (ou AdditionalElements), simplesmente é necessário arrastar eles em um dos campos AdditionalElements disponíveis, explicados a seguir.

- AdditionalElements0: Neste campo arrastam-se elementos adicionais que precisem estarem ativos durante a pergunta toda (nomeado Estado 0). Através de Size se permite definir a quantidade destes elementos.
- AdditionalElements1: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente quando está sendo respondida a pergunta (Estado 1).

- AdditionalElements2: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente após responder a pergunta (Estado 2). Nota-se que este é o mesmo estado da pergunta onde são ativadas as imagens de *feedback* de resposta correta e errada.
- UseAnswerButton: Quando esta opção estiver ativada, permite usar um botão para ver a resposta correta da pergunta após ela ter sido respondida.

  Opções com UseAnswerButton ativada:

  - AnswerButton: Neste campo arrasta-se o botão de ver a resposta certa.
  - UseDefaultAnswer: Quando esta opção estiver ativada, permite usar o jeito padrão de ver a resposta correta quando for pressionado o botão, que consiste em ativar os textos ou imagens alternativas indicando a alternativa correta através de uma imagem sobreposta nela, e ativar os objetos pergunta arrastados em QuestionObjects (no caso de existir).

    Opções com UseDefaultAnswer ativada:

    - CorrectAnswerImage2: Neste campo arrasta-se o elemento Image do Unity que permitirá indicar a alternativa certa.
  - AdditionalElements3: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente quando está sendo apresentada a resposta correta da pergunta (Estado 3).

Usar Múltiplas Perguntas Com **`QuestionController1`**: Para utilizar várias perguntas desde tipo numa mesma cena do Unity, o *framework* permite o reuso de todos os elementos necessários (botões, imagens etc.), excetuando os elementos usados como alternativas, que devem ser independentes para cada pergunta, sendo possível assim criar uma nova pergunta simplesmente duplicando o objeto original que contém a classe QuestionCotroller1, criando seus próprios elementos de alternativa, personalizando os campos da nova pergunta, e finalmente utilizando as opções para mudar de pergunta com os campos UseNextQuestionButton e UsePreviousQuestionButton.

Instruções Do Uso Da Classe QuestionController1E: A classe deve ser arrastada em um objeto do Unity que já possui a classe **`QuestionController1`**, mantendo esta última classe com todos seus campos por *default*. Finalmente após ser feita a inicialização, recomenda-se remover a classe **`QuestionController1E`**.

Finalmente, um tutorial de exemplo, usando a classe **`QuestionController1`** e sua inicialização *default* com **`QuestionController1E`**, é fornecido para maior compreensão em "Tutorial Classe **`QuestionController1`**" em Apêndice D. O tutorial também inclui os passos para usar múltiplas perguntas numa mesma cena.

Instruções Do Uso Da Classe QuestionController2:

Para cada pergunta necessária, a classe deve ser arrastada sobre um objeto do Unity independente (cada instância da classe deve ficar em um GameObject independente), assim recomenda-se para cada pergunta, criar um novo GameObject vazio do Unity (**`GameObject/Create Empty`**) e arrastar nele a classe. Os campos da classe que ficam visíveis para o usuário no inspetor, permitem configurar as preferências do usuário para cada pergunta, e em vários deles, devem ser arrastados elementos necessários (marcadores, elementos 3D, botões, etc.), que deverão ser criados e posicionados na cena do Unity pelo usuário, e a classe será encarregada do seu gerenciamento.

A seguir, são explicados em detalhe os campos da classe e seu uso, nota-se que no caso de não fazer a inicialização *default* (disponível através de **`QuestionController2E`**), todos os elementos que for necessários deverão ser criados e posicionados pelo usuário:

- Active: Permite ativar ou desativar a pergunta com todos seus elementos relacionados. Sempre que for usadas várias perguntas numa mesma cena do Unity, esta opção será usada para indicar qual será a primeira delas, deixando seu valor em **`True`**, e colocando o valor das outras perguntas em **`False`**.

- Marker: Neste campo arrasta-se o marcador vinculado aos elementos 3D que representam as alternativas da pergunta, ou seja o marcador através do qual serão apresentadas as alternativas.
- AnswerAlternatives: Neste campo arrastam-se os elementos 3D que representam as alternativas da pergunta, os quais devem estar numa mesma posição, e ser colocados como filhos (na hierarquia do Unity) do marcador arrastado em Marker. Estes elementos tem a características de serem restabelecidos a seu estado original quando a pergunta for reiniciada, e ainda quando se passa ao estado de ver a resposta certa.
- CorrectAnswerIndex: Permite definir o índice do elemento que representa a alternativa certa.
- LeftButton: Neste campo arrasta-se um botão da interface do Unity (**`GameObject/UI/Button`**), que permitirá passar à alternativa anterior da pergunta, desativando o elemento atual e ativando o anterior.
- RightButton: Neste campo arrasta-se um botão, que permitirá passar à alternativa seguinte da pergunta, desativando o elemento atual e ativando o próximo.
- CheckButton: Neste campo arrasta-se um botão, que permitirá confirmar a alternativa atual.
- ResetButton: Neste campo arrasta-se um botão, que permitirá repetir a pergunta após ter sido respondida. O processo de fazer reset numa pergunta inclui restabelecer ao estado original a *transform* (rotação e escala) e os Materiais dos objetos associados com a pergunta (ainda elementos adicionais que estiverem sendo utilizados, ver campos AdditionalElements).
- UseNextQuestionButton: Quando esta opção estiver ativada, permite usar um botão para passar a uma próxima pergunta.

  Opções com UseNextQuestionButton ativada:

  - NextQuestionButton: Neste campo arrasta-se o botão que permitirá passar à próxima pergunta.
  - NextQuestion: Neste campo arrasta-se o GameObject que tem acrescentado o **`QuestionController2`** (ou **`QuestionController1`**) da próxima pergunta.

- UsePreviousQuestionButton: Quando esta opção estiver ativada, permite usar um botão para passar a uma pergunta anterior.

  Opções com UsePreviousQuestionButton ativada:

  - PreviousQuestionButton: Neste campo arrasta-se o botão que permitirá passar à pergunta anterior.
  - PreviousQuestion: Neste campo arrasta-se o GameObject que tem acrescentado o **`QuestionController2`** (ou **`QuestionController1`**) da pergunta anterior.
- RandomOptions: Quando esta opção estiver ativada, permite que as alternativas sejam apresentadas em jeito aleatório.
- CorrectAnswerImage: Neste campo arrasta-se um elemento Image do Unity (**`GameObject/UI/Image`**), que vai ser usado como a imagem de *feedback* de resposta certa.
- WrongAnswerImage: Neste campo arrasta-se um elemento Image do Unity, que vai ser usado como a imagem de *feedback* de resposta errada.
- UseTime: Quando esta opção estiver ativada, permite definir um tempo limite para responder a pergunta. Uma vez que for usado, fica também relacionado com a resolução da pergunta, assim, finalizar o tempo faz com que a pergunta seja errada.

  Opções com UseTime ativada:

  - TimeDependingOfMarker: Quando esta opção estiver ativada, permite que o tempo dependa da visibilidade do marcador, correndo normalmente sempre que o marcador estiver visível pela câmera, e ficando pausado no caso contrário.
  - TotalTime: Permite definir o tempo total ou máximo (em segundos) para responder a pergunta.
  - TimeText: Neste campo arrasta-se o elemento Text (**`GameObject/UI/Text`**), que será usado como o texto do tempo.
  - UseAlertTime: Quando esta opção estiver ativada, permite que a partir de certo tempo, mude a cor do texto do tempo para indicar que está esgotando.

Opções com UseAlertTime ativada:

- AlertTime: Permite definir o tempo a partir do qual vai ser mudada a cor do texto do tempo.
- NewTextColor: Permite definir a nova cor do texto do tempo pela qual vai ser mudada a cor original.

Campos AdditionalElements: Os campos AdditionalElements, são usados para suportar elementos adicionais, e têm a mesma funcionalidade dos "Campos AdditionalElements" já descritos em "Instruções Do Uso Da Classe **`QuestionController1`**".

Usar objetos como Elementos Adicionais (AdditionalElements): Para usar objetos como elementos adicionais (ou AdditionalElements), simplesmente é necessário arrastar eles em um dos campos AdditionalElements disponíveis, explicados a seguir.

- AdditionalElements0: Neste campo arrastam-se elementos adicionais que precisem estarem ativos durante a pergunta toda (nomeado Estado 0). Através de Size se permite definir a quantidade destes elementos.
- AdditionalElements1: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente quando está sendo respondida a pergunta (Estado 1).
- AdditionalElements2: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente após responder a pergunta (Estado 2).
- UseAnswerButton: Quando esta opção estiver ativada, permite usar um botão para ver a resposta correta da pergunta após ela ter sido respondida.

  Opções com UseAnswerButton ativada:

  - AnswerButton: Neste campo arrasta-se o botão de ver a resposta certa.
  - UseDefaultAnswer: Quando esta opção estiver ativada, permite usar o jeito padrão de ver a resposta correta quando for

pressionado o botão, que consiste em ativar o elemento 3D que representa a alternativa certa.

- AdditionalElements3: Neste campo arrastam-se elementos adicionais que precisem estarem ativos unicamente quando está sendo apresentada a resposta correta da pergunta (Estado 3).

Usar Múltiplas Perguntas Com **`QuestionController2`**: Para utilizar várias perguntas deste tipo numa mesma cena do Unity, o *framework* permite o reuso de todos os elementos necessários (botões, imagens etc.), sendo possível criar uma nova pergunta simplesmente duplicando o objeto original que contém a classe QuestionCotroller2, personalizando os campos da nova pergunta, e utilizando as opções para mudar de pergunta com os campos UseNextQuestionButton e UsePreviousQuestionButton.

Instruções Do Uso Da Classe QuestionController2E: A classe deve ser arrastada em um objeto do Unity que já possui a classe **`QuestionController2`**, mantendo esta última classe com todos seus campos por *default*. Finalmente após ser feita a inicialização, recomenda-se remover a classe **`QuestionController2E`**.

Finalmente, um tutorial de exemplo, usando a classe **`QuestionController2`** e sua inicialização *default* com **`QuestionController2E`**, é fornecido para maior compreensão em “Tutorial Classe **`QuestionController2`**” em Apêndice D.

Apêndice D

TUTORIAIS CLASSES DO AR EDUCATIONAL FRAMEWORK

A seguir são apresentados tutoriais que incluem imagens de capturas de tela, usando as classes das funcionalidades do *framework* com suas respetivas inicializações *default*. O tutorial apresenta como usar cada classe de um jeito isolado, e também como usar várias delas juntas, aplicando-as sobre os mesmos elementos. Cada elemento do Unity que for necessário criar, terá indicado entre parêntesis os passos para sua criação.

Nota-se que os marcadores padrão utilizados em cada funcionalidade, são marcadores *default* da Vuforia, e suas imagens podem ser acessados no caminho: **`Editor/Vuforia/ImageTargetTextures/VuforiaMars_Images`**. Para fácil acesso, uma pasta chamada: Imagens Marcadores Padrão, fica disponível no link: https://www.dropbox.com/sh/urve0vbtegm2q3p/AACXArDmeMPuRKtsxpxVaEuda?dl=0

Tutorial Classe ButtonsPack

Inicialização Default Da Classe ButtonsPack

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`ButtonsPack`** com sua inicialização padrão, que vai permitir criar automaticamente um pacote de botões padrões (com todos os botões necessários

para rotacionar e escalar um objeto), um objeto cubo de exemplo (que poderá ser rotacionado e escalado pelos botões), e um marcador de RA para o objeto. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALBUTTONSPACK, disponível no link:

https://www.dropbox.com/s/cjr1g3j5jzn6azp/PROJETOUNITYTUTORIALBUTTONSPACK.rar?dl=0[15]

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura D 1, que será usado como container para as classes que permitirão criar os elementos comentados (classes **`ButtonsPack`** e **`ButtonsPackE`**).

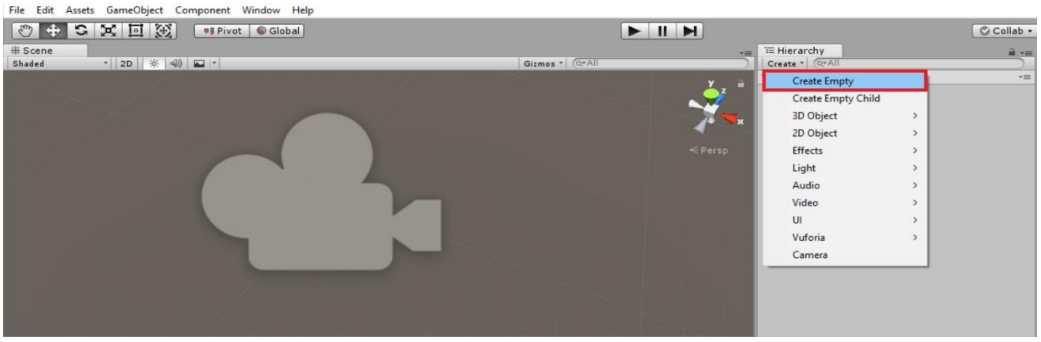

Figura D 1: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`ButtonsPack`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura D 2, como o pacote de botões já criado dentro de um elemento **`Canvas`** do Unity.

[15] Todas as referências ao projeto "PROJETOUNITYTUTORIALBUTTONSPACK" neste documento, se referem a este link.

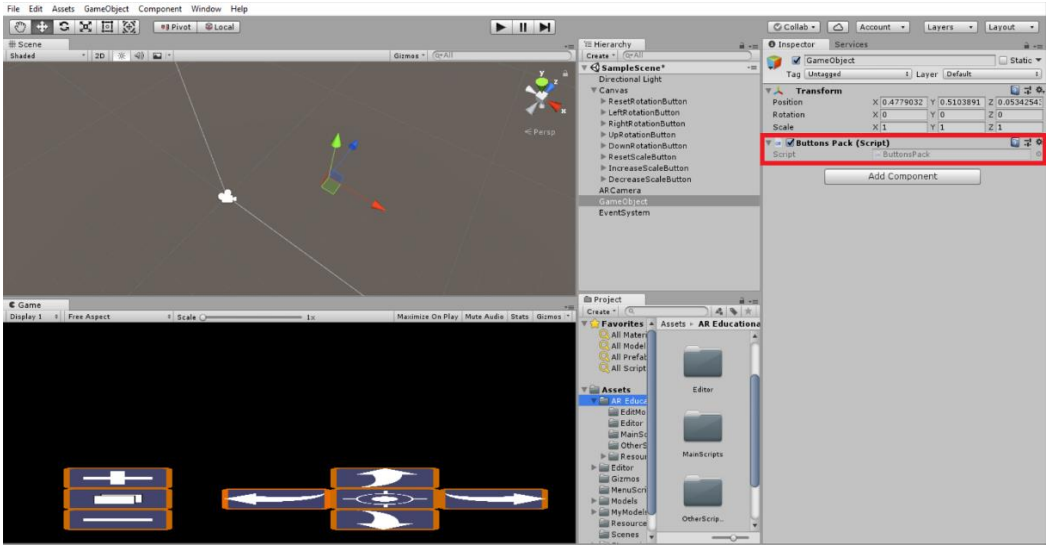

Figura D 2: Classe `ButtonsPack` arrastada no objeto vazio, e pacote de botões já criados.

3) Selecionar o objeto vazio, e arrastar nele a classe **`ButtonsPackE`** (contida na pasta **`EditMode`**) para a criação do cubo exemplo e seu marcador (recomenda-se fazer click no marcador **`ImageTarget`** quando for criado para fazer ele visível na cena). A cena deveria ficar como na Figura D 3, já pronta para ser executada, com o cubo sobre o marcador, e os botões respectivos dentro do **`Canvas`**.

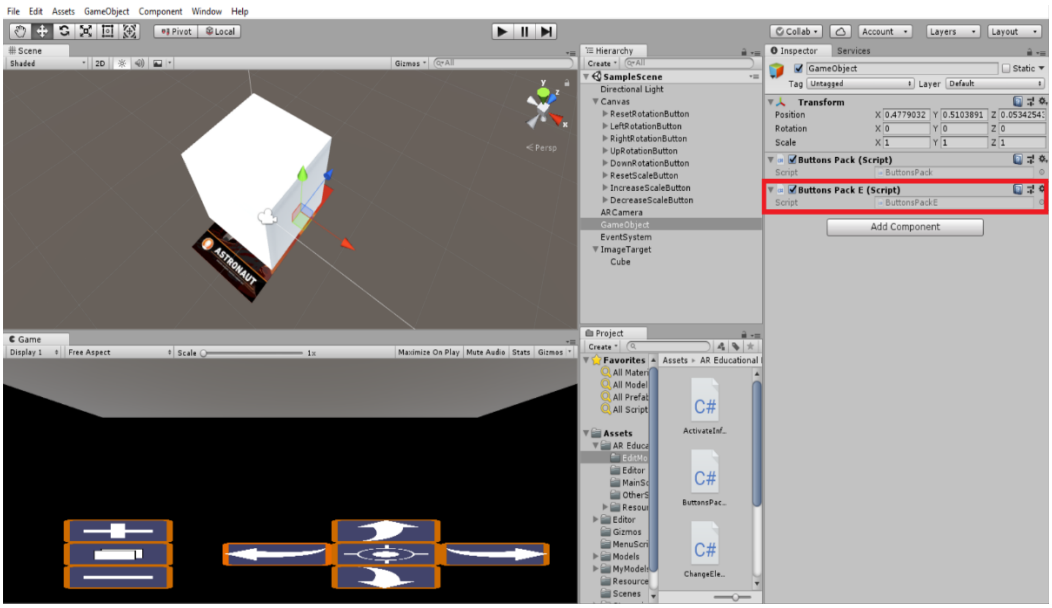

Figura D 3: Classe `ButtonsPackE` arrastada no objeto vazio, e cena já pronta para ser executada.

4) Finalmente recomenda-se remover o objeto vazio container que não será mais necessário. Nota: Para configurar preferencias nas operações de rotação e escala, ao invés da classe **`ButtonsPack`** será usada a classe **`ButtonOperations`** contida em cada botão.

Configurações Para Usar Modelos 3D Próprios Com Classe ButtonsPack

A seguir, serão apresentados os passos para usar um novo modelo 3D ao invés do cubo padrão, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALBUTTONSPACK.

1) Importar o pacote Heart de modelos no projeto, o qual possui o modelo necessário Heart. O pacote Heart está disponível no link: https://www.dropbox.com/s/aqu5qxbtqsech3j/Heart.rar?dl=0[16]
2) Substituir o novo modelo 3D Heart (caminho **`MyModels/Heart`**) pelo cubo, para isso: Colocar o novo modelo Heart na mesma posição do cubo, e fazer ele filho do mesmo marcador. A cena deveria ficar como na Figura D 4.

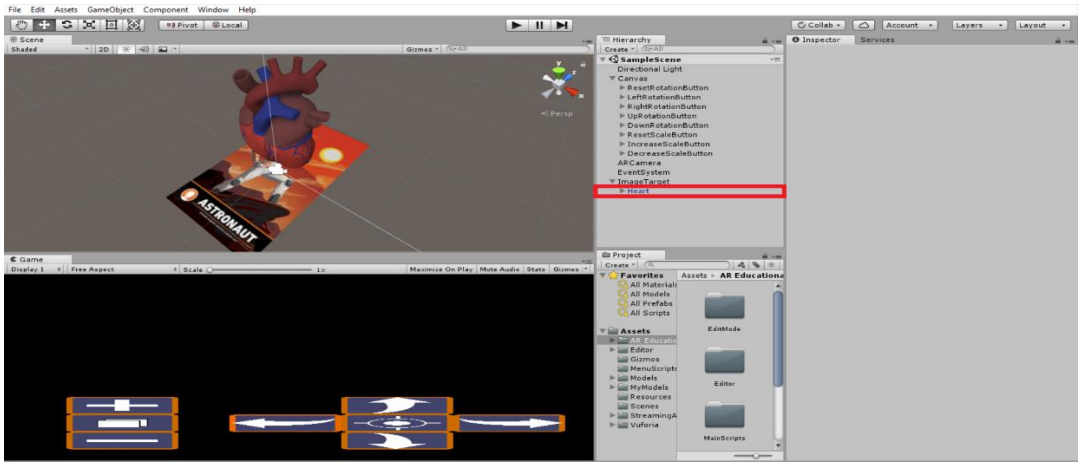

Figura D 4: Novo modelo 3D Heart, já vinculado ao marcador `ImageTarget`.

[16] Todas as referências ao pacote “Heart” neste documento, se referem a este link.

3) Finalmente para cada botão, deve se arrastar o novo modelo 3D no seu campo Objects da classe **`ButtonOperations`**. Para fazer isso rapidamente, são selecionados todos os botões no mesmo tempo mantendo a tecla CTRL pressionada, e finalmente se arrasta o modelo 3D Heart no campo Objects, ver o processo na Figura D 5. Após isso, a cena deveria ficar pronta para ser executada.

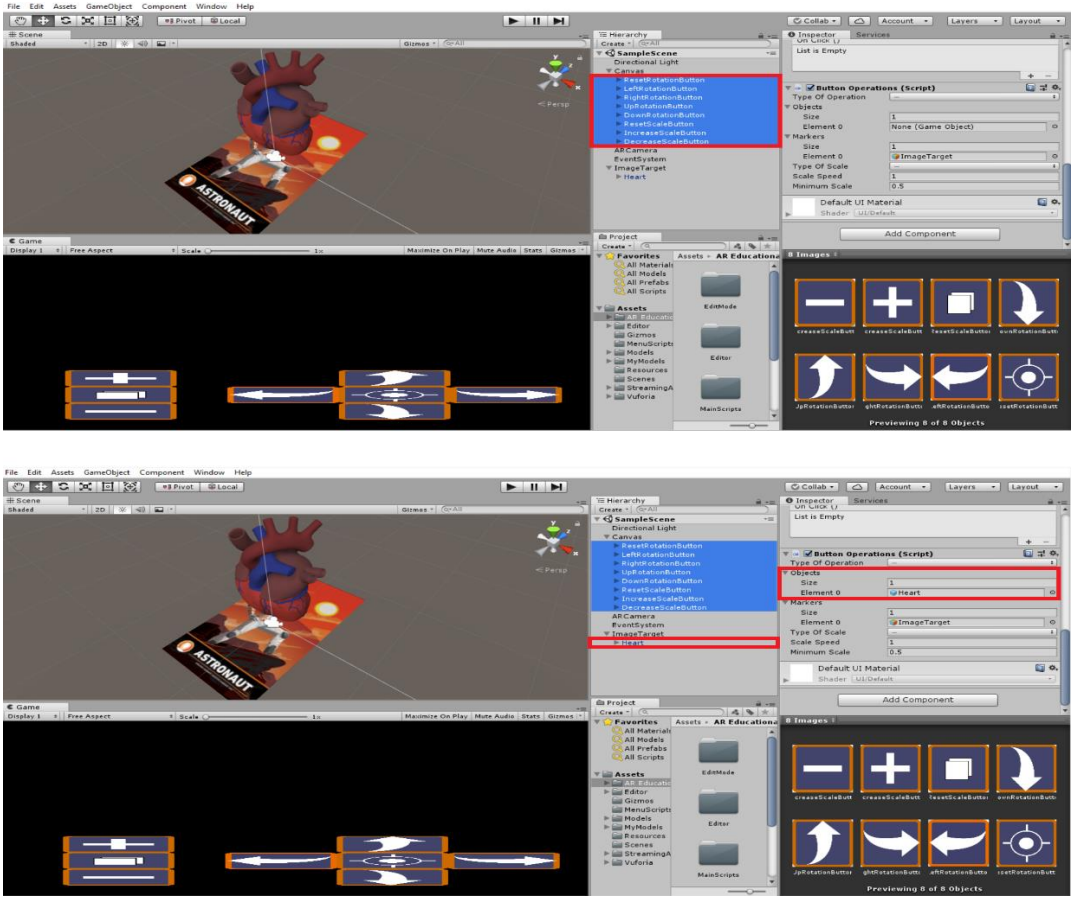

Figura D 5: Inicialmente todos os botões são selecionados no mesmo tempo com a tecla CTRL, e finalmente o modelo Heart é arrastado no campo Objects

Tutorial Classe ChangeElement

Inicialização Default Da Classe ChangeElement

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`ChangeElement`** com sua inicialização padrão, que vai permitir criar automaticamente um objeto cubo vinculado a um marcador de RA, onde o objeto vai

ser mudado para outra cor (trocando seu Material) sempre que seu marcador estiver junto de um marcador de controle. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALCHANGEELEMENT, disponível no link:

https://www.dropbox.com/s/t2cdb4ghb53ixyb/PROJETOUNITYTUTORIALCHANGEELEMENT.rar?dl=0[17]

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura D 6, que será usado como container para as classes que permitirão fazer o evento de mudar cor (classes **`ChangeElement`** e **`ChangeElementE`**).

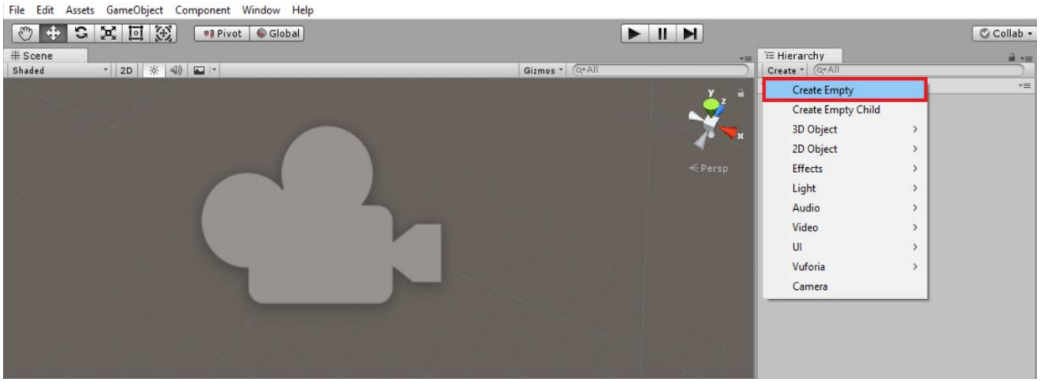

Figura D 6: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`ChangeElement`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura D 7.

[17] Todas as referências ao projeto "PROJETOUNITYTUTORIALCHANGEELEMENT" neste documento, se referem a este link.

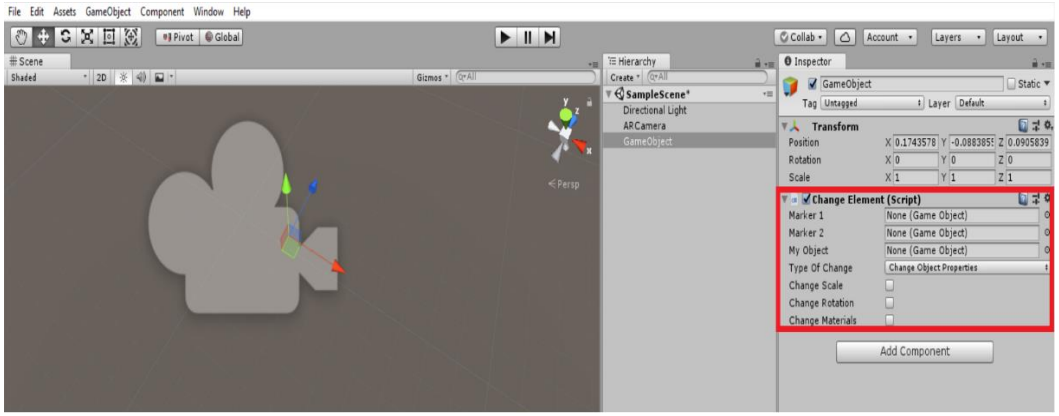

Figura D 7: Classe `ChangeElement` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`ChangeElementE`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default* (recomenda-se fazer click nos marcadores **`ImageTarget`** e **`ImageTarget2`** quando for criados para fazer eles visíveis na cena). A cena deveria ficar como na Figura D 8, já pronta para ser executada, com o cubo sobre o marcador principal, e ao lado dele o marcador de controle que ativa o evento de mudar cor.

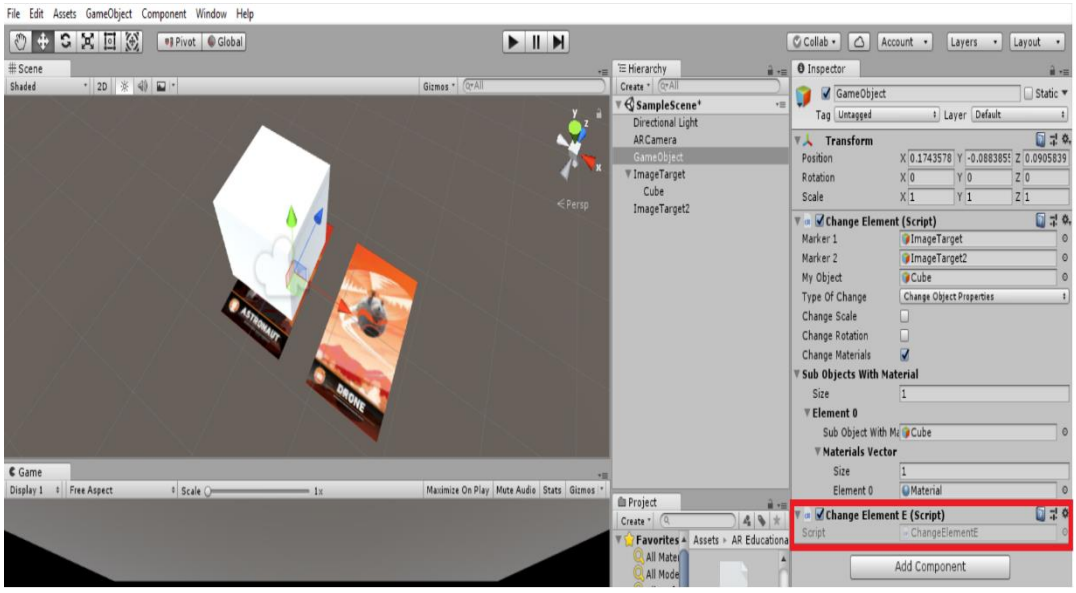

Figura D 8: Classe `ChangeElementE` arrastada no objeto vazio, e cena pronta para ser executada.

4) Finalmente, após ser feita a inicialização padrão recomenda-se remover a classe **`ChangeElementE`**.

Configurar Classe ChangeElement Para Usar Modelos 3D Próprios

A seguir, serão apresentados os passos para mudar a cor de um novo modelo 3D, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALCHANGEELEMENT.

1) Importar o pacote Heart de modelos no projeto, o qual possui o modelo necessário Heart.
2) Substituir o novo modelo 3D Heart (caminho **`MyModels/Heart`**) pelo cubo, para isso: Colocar o novo modelo Heart na mesma posição do cubo, e fazer ele filho do mesmo marcador. A cena deveria ficar como na Figura D 9.

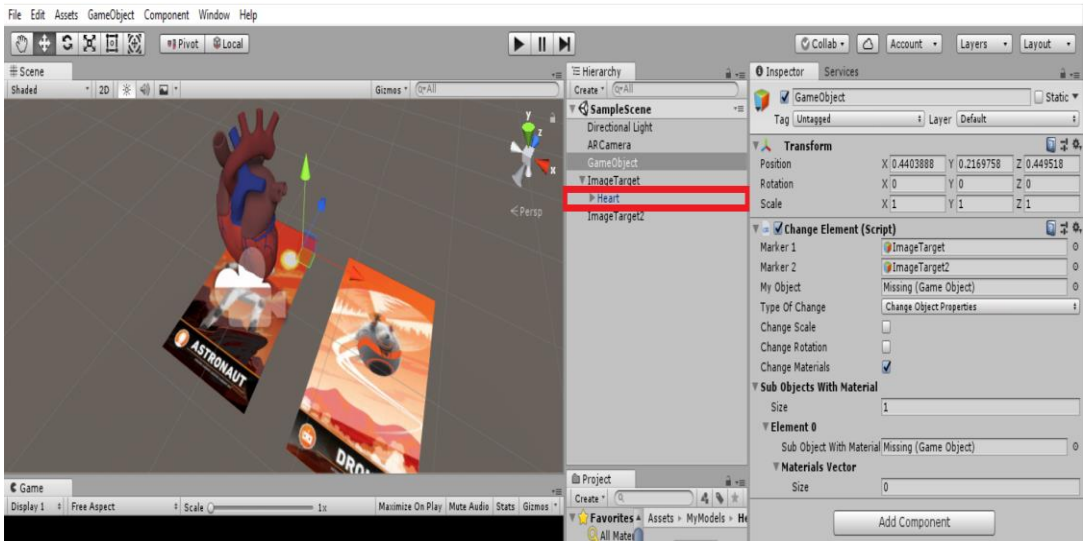

Figura D 9: Novo modelo 3D Heart, já vinculado ao marcador principal `ImageTarget`.

3) Arrastar o novo modelo 3D Heart no campo MyObject da classe. A cena deve ficar como na Figura D 10.

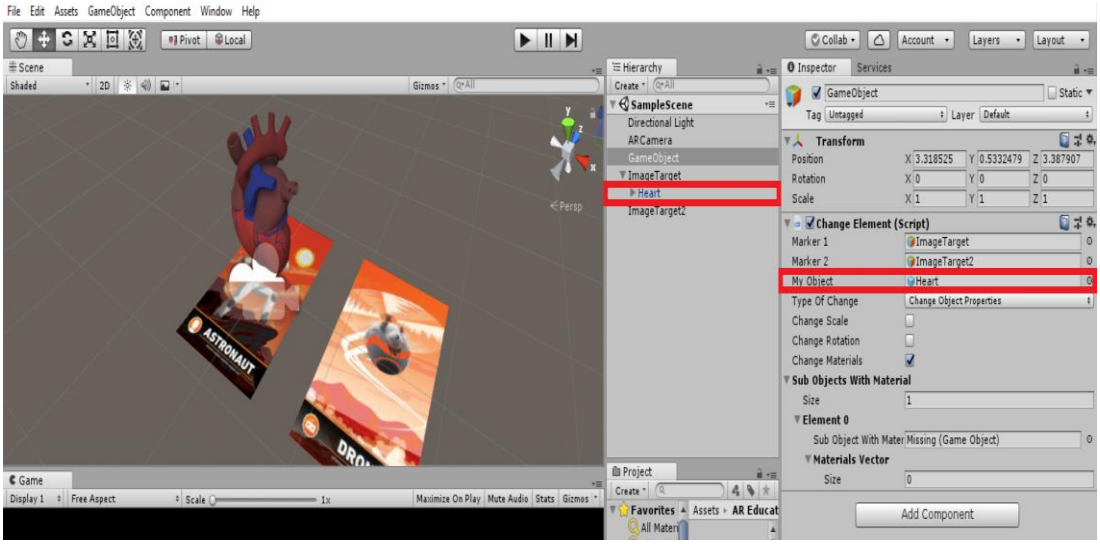

Figura D 10: Novo modelo 3D Heart, arrastado no campo MyObject da classe.

4) O modelo 3D Heart, tem seus materiais (que definem sua cor) distribuídos nos seus filhos, portanto neste passo, deve-se definir a qual dos filhos do modelo 3D é ao que vamos muda-lhe o material. Para este exemplo é escolhido o objeto filho chamado Ventriculos, do qual será mudado unicamente seu segundo material chamado Corazon, que define a cor da parte inferior do modelo. Na Figura D 11 pode ser vista a lista dos materiais que poderíamos mudar para o objeto filho escolhido.

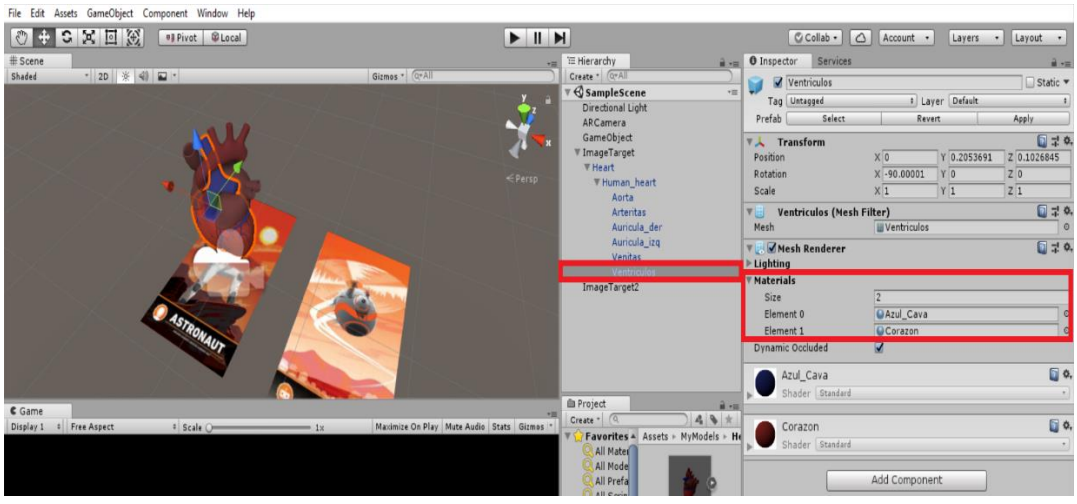

Figura D 11: Materiais disponíveis para novo modelo 3D Heart, no seu filho Ventriculos.

5) Arrastar o filho do modelo escolhido no passo anterior (chamado Ventriculos), no campo SubObjectWithMaterial. O *framework* detectará automaticamente a quantidade de materiais existentes, e deixará

disponíveis campos para mudar cada um deles respetivamente em MaterialsVector. A cena deveria ficar como na Figura D 12.

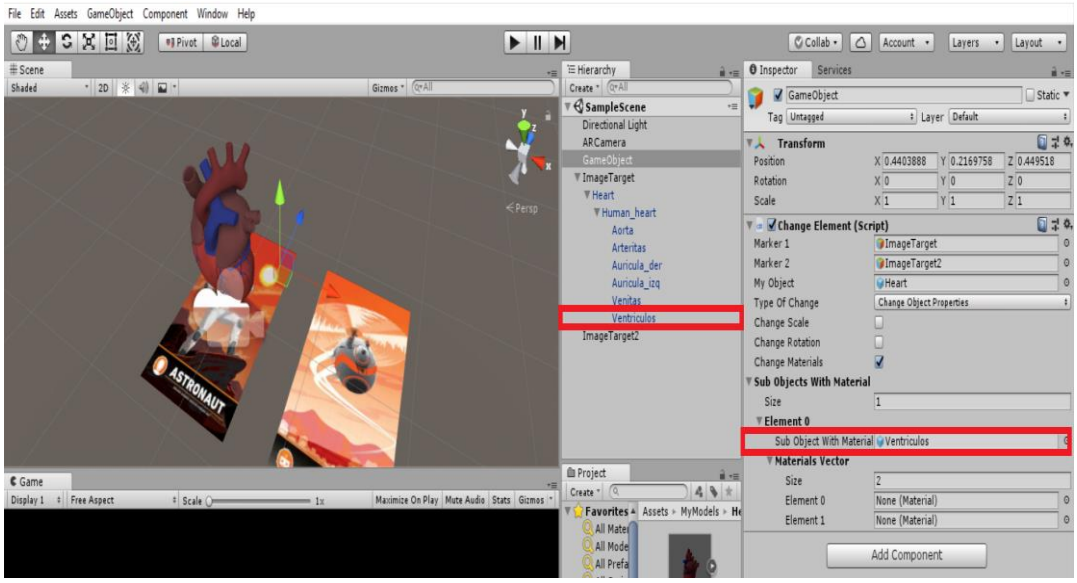

Figura D 12: Filho do modelo 3D com materiais (Ventriculos), já arrastado no campo respetivo.

6) Finalmente arrasta-se o novo material com a nova cor, no segundo campo disponível de MaterialsVector que se refere ao material que queremos mudar (neste caso, se usa como novo material, o de cor azul do *framework*, chamado Material e localizado na pasta com o caminho: **`AR Educational Framework/Resources`**). A cena deveria ficar como na Figura D 13, já pronta para ser executada. Nota-se que deixar campos vazios em MaterialsVector, faz que os materiais originais respectivos fiquem sem mudanças.

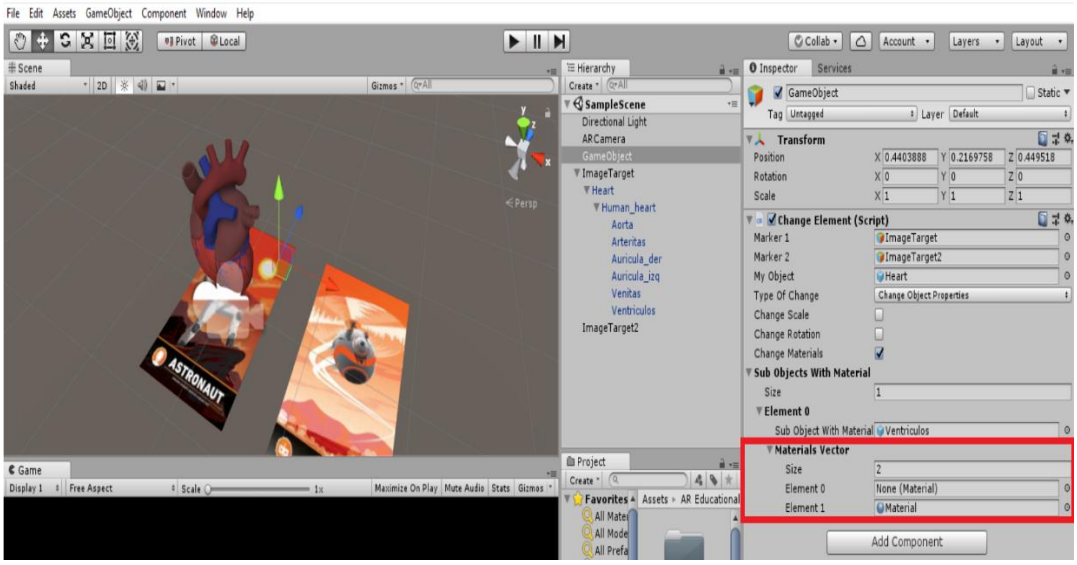

Figura D 13: Material novo arrastado no segundo campo disponível de MaterialsVector, e cena já pronta para ser executada.

Tutorial Classe ActivateInformation

Inicialização Default Da Classe ActivateInformation

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`ActivateInformation`** com sua inicialização padrão, que vai permitir criar automaticamente um objeto cubo vinculado a um marcador de RA, onde vai ser ativado um texto 3D sobre o objeto sempre que seu marcador estiver junto de um marcador de controle. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALACTIVATEINFORMATION, disponível no link:

https://www.dropbox.com/s/85zo6ovucx0q119/PROJETOUNITYTUTORIALACTIVATEINFORMATION.rar?dl=0[18]

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura D 14, que será usado como container para as classes que permitirão fazer

[18] Todas as referências ao projeto "PROJETOUNITYTUTORIALACTIVATEINFORMATION" neste documento, se referem a este link.

o evento de ativar o texto 3D sobre o objeto (classes **`ActivateInformation`** e **`ActivateInformationE`**).

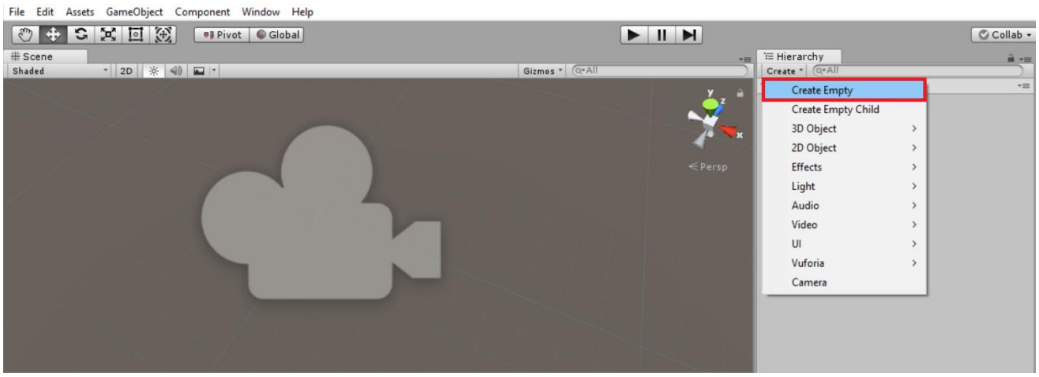

Figura D 14: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`ActivateInformation`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura D 15.

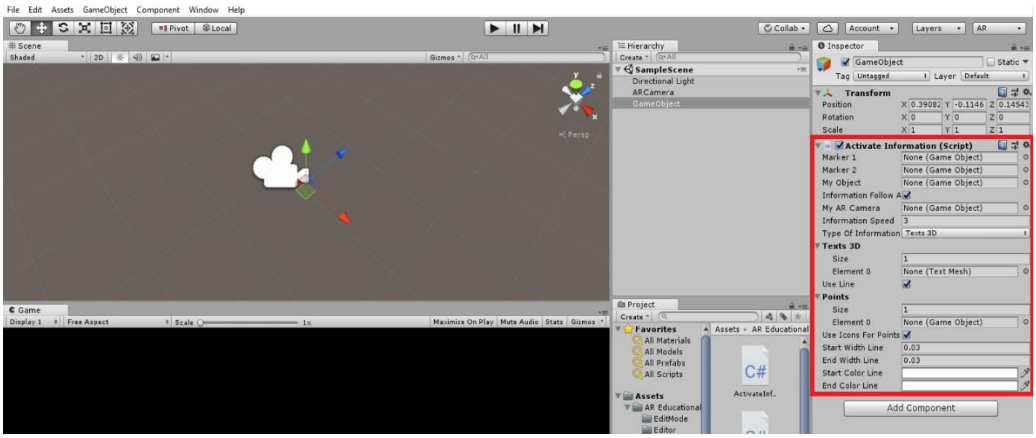

Figura D 15: Classe `ActivateInformation` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`ActivateInformationE`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default* (recomenda-se fazer click nos marcadores **`Imagetarget`** e **`ImageTarget2`** quando for criados para fazer eles visíveis na cena). A cena deveria ficar como na Figura D 16, já pronta para ser executada, com o cubo sobre o marcador principal, acima do cubo o texto 3D (que vai ser ativado através do marcador de controle), e ao lado o marcador de controle (que ativará o texto 3D).

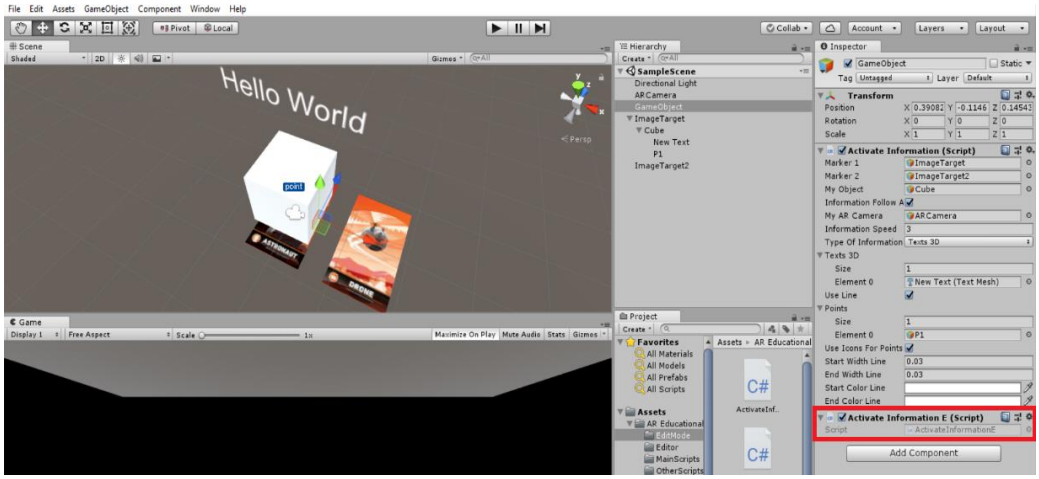

Figura D 16: Classe `ActivateInformationE` arrastada no objeto vazio, e cena pronta para ser executada.

4) Finalmente, após ser feita a inicialização padrão, recomenda-se remover a classe **`ActivateInformationE`**.

Configurar Classe ActivateInformation Para Usar Modelos 3D Próprios

A seguir, serão apresentados os passos para aplicar a mesma funcionalidade em um novo modelo 3D, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALACTIVATEINFORMATION.

1) Importar o pacote Heart de modelos no projeto, que possui o modelo necessário Heart.
2) Substituir o novo modelo 3D Heart (caminho **`MyModels/Heart`**) pelo cubo, para isso: Colocar o modelo Heart na mesma posição do cubo, faze-lo filho do mesmo marcador, e finalmente fazer filhos dele aos elementos filhos do objeto cubo (elementos newText e P1, que são o texto 3D e um ponto que indica a origem da linha do texto 3D, ambos usados no evento da classe **`ActivateInformation`**). A cena deveria ficar como na Figura D 17.

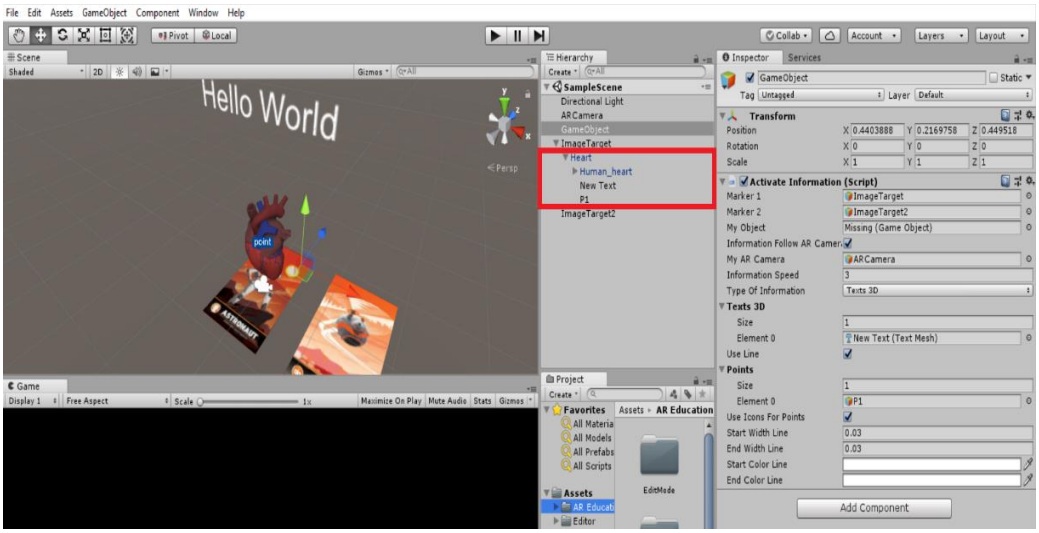

Figura D 17: Novo modelo 3D já vinculado ao marcador principal, os elementos newText e P1 que eram filhos do objeto cubo original, agora são filhos do novo modelo.

3) Finalmente arrastar o novo modelo 3D Heart, no campo MyObject da classe **`ActivateInformation`**. A cena deveria ficar como na Figura D 18, já pronta para ser executada.

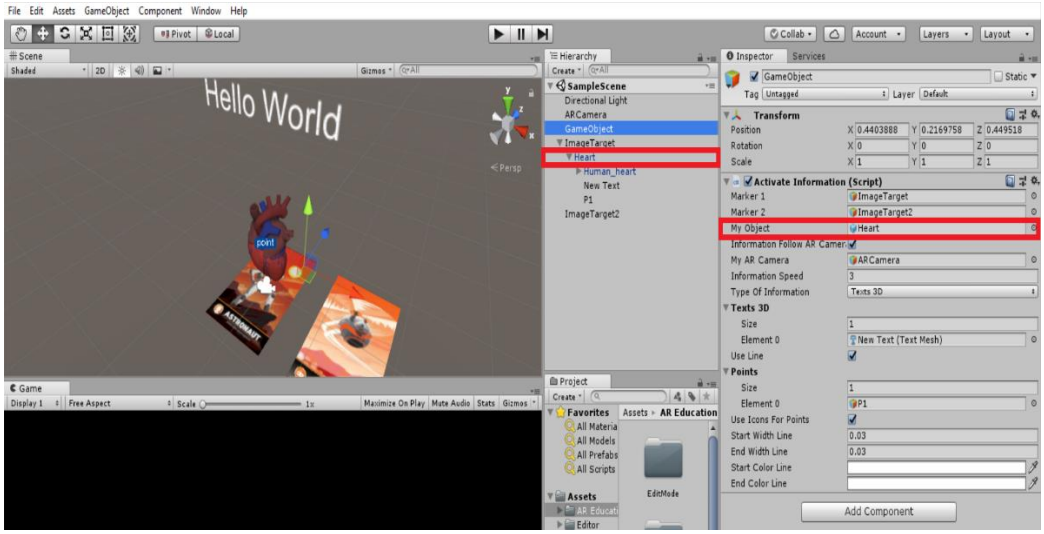

Figura D 18: Novo modelo 3D arrastado no campo MyObject, e cena pronta para ser executada.

Tutorial Classe QuestionController1

Inicialização Default Da Classe QuestionController1

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController1`** com sua inicialização padrão, que vai permitir criar automaticamente uma pergunta (com elementos padrões de exemplo), na qual as alternativas são textos 2D tipo UI tradicional. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALQUESTIONCONTROLLER1, disponível no link:

https://www.dropbox.com/s/p10ke3oqo0994zg/PROJETOUNITYTUTORIALQUESTIONCONTROLLER1.rar?dl=0[19]

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura D 19, que será usado como container para as classes que permitirão criar a pergunta (classes **`QuestionController1`** e **`QuestionController1E`**).

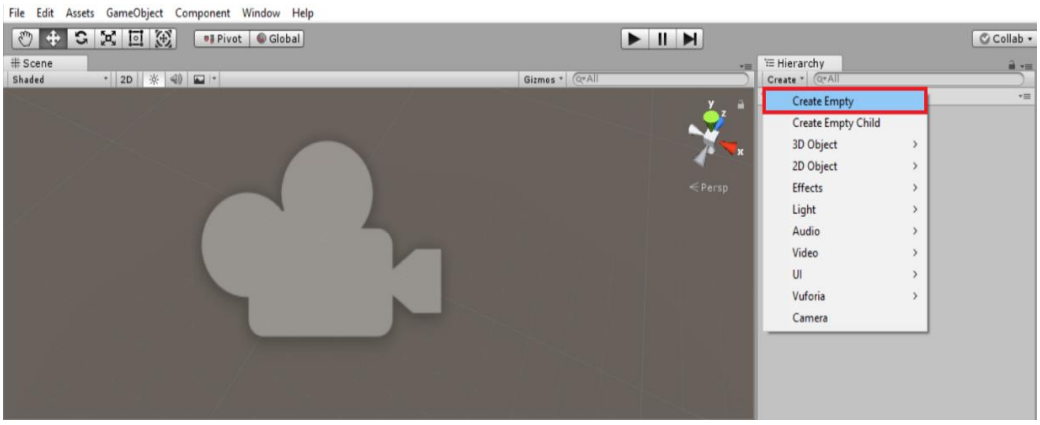

Figura D 19: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController1`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura D 20.

[19] Todas as referências ao projeto "PROJETOUNITYTUTORIALQUESTIONCONTROLLER1" neste documento, se referem a este link.

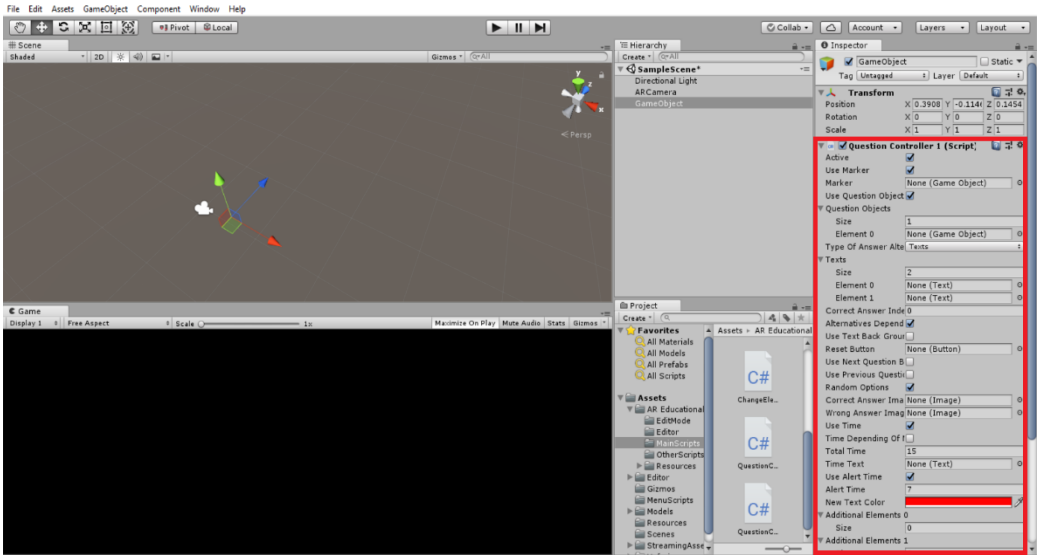

Figura D 20: Classe `QuestionController1` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController1E`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default*, após isso recomenda-se fazer click no marcador criado **`ImageTarget`** para fazer ele visível na cena.

   A cena deveria ficar como na Figura D 21, já pronta para ser executada, com um cubo sobre um marcador (que representa um objeto exemplo sobre o que se faz a pergunta), na parte direita da tela dois textos usados como alternativas de resposta, no centro duas imagens de *feedback* de resposta correta e errada com botões para ver resposta correta e repetir a pergunta, no corner superior direito um texto para mostrar o tempo, e finalmente na parte superior da tela dois textos adicionais para pergunta e resposta respetivamente.

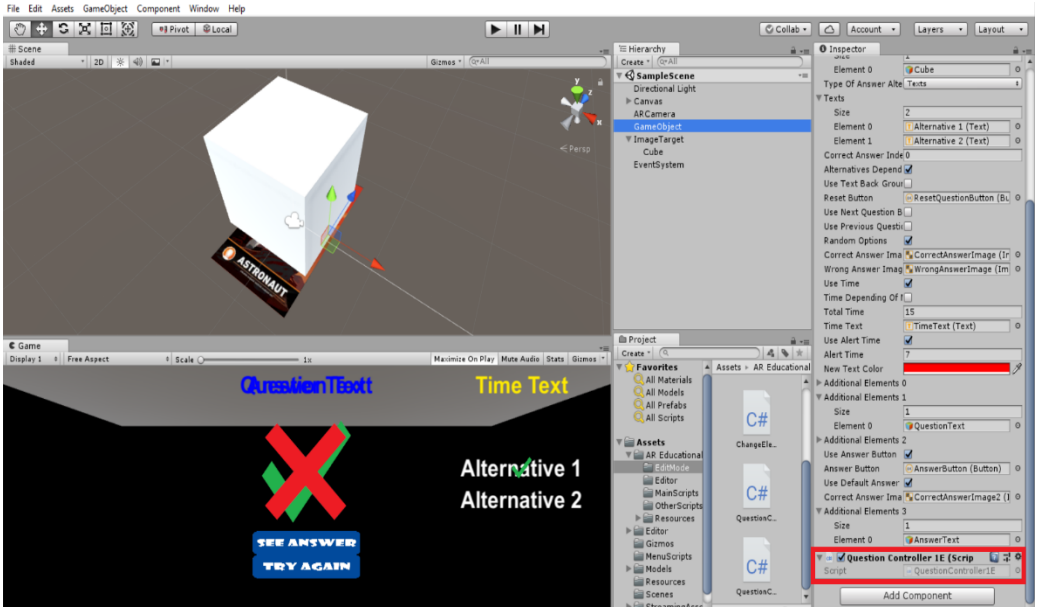

Figura D 21: Classe `QuestionController1E` arrastada no objeto vazio, e cena já pronta para ser executada.

4) Finalmente, após ser feita a inicialização padrão recomenda-se remover a classe **`QuestionController1E`**.

Configurar Classe QuestionController1 Para Usar Modelos 3D Próprios

A seguir, serão apresentados os passos para aplicar a mesma funcionalidade com um novo modelo 3D, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALQUESTIONCONTROLLER1.

1) Importar o pacote Heart de modelos no projeto, o qual possui o modelo necessário Heart.
2) Substituir o novo modelo 3D Heart (caminho **`MyModels/Heart`**) pelo cubo, para isso: Colocar o novo modelo Heart na mesma posição do cubo, e fazer ele filho do mesmo marcador. A cena deveria ficar como na Figura D 22.

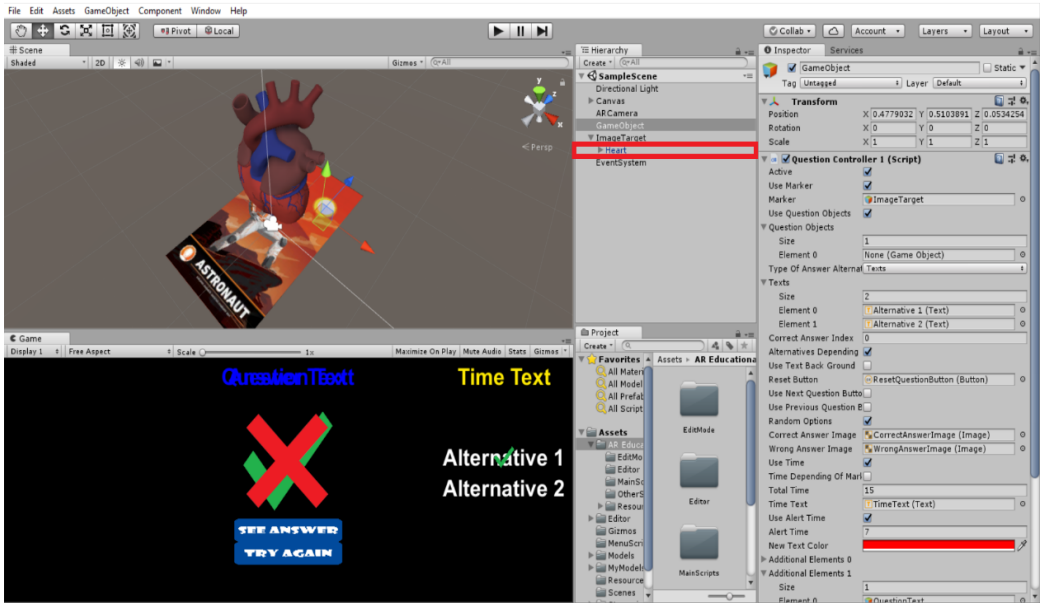

Figura D 22: Novo modelo 3D Heart, vinculado ao marcador `ImageTarget`.

3) Finalmente arrastar o novo modelo 3D Heart, no campo QuestionObjects da classe **`QuestionController1`**. A cena deveria ficar como na Figura D 23, já pronta para ser executada.

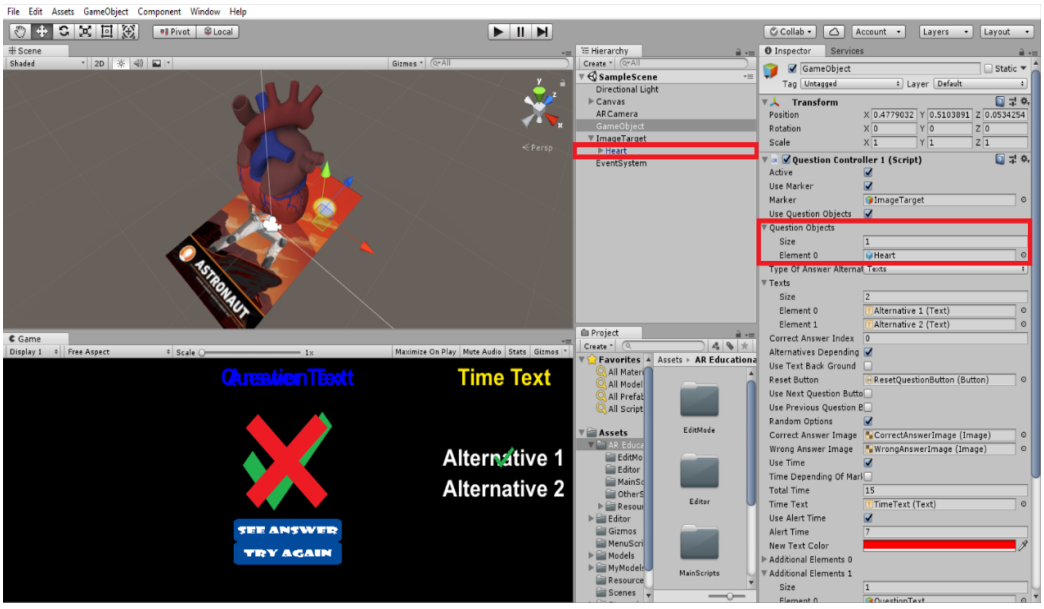

Figura D 23: Novo modelo 3D Heart arrastado no campo QuestionObjects, e cena pronta para ser executada.

Configurar Classe QuestionController1 Para Usar Múltiplas Perguntas

A seguir será explicado o jeito de usar múltiplas perguntas numa cena, para o exemplo, a pergunta atual ficará igual, e será criado uma nova pergunta onde o segundo texto alternativa será configurado como o certo, assim uma vez responder a primeira pergunta será possível ir para a nova. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene3, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALQUESTIONCONTROLLER1.

1) Duplicar o objeto vazio original que possui a classe **`QuestionController1`** fazendo click direito nele e na opção Duplicate. Para melhor compreensão os objetos vazios são renomeados como Question1 e Question2 respetivamente, a cena deveria ficar como na Figura D 24.

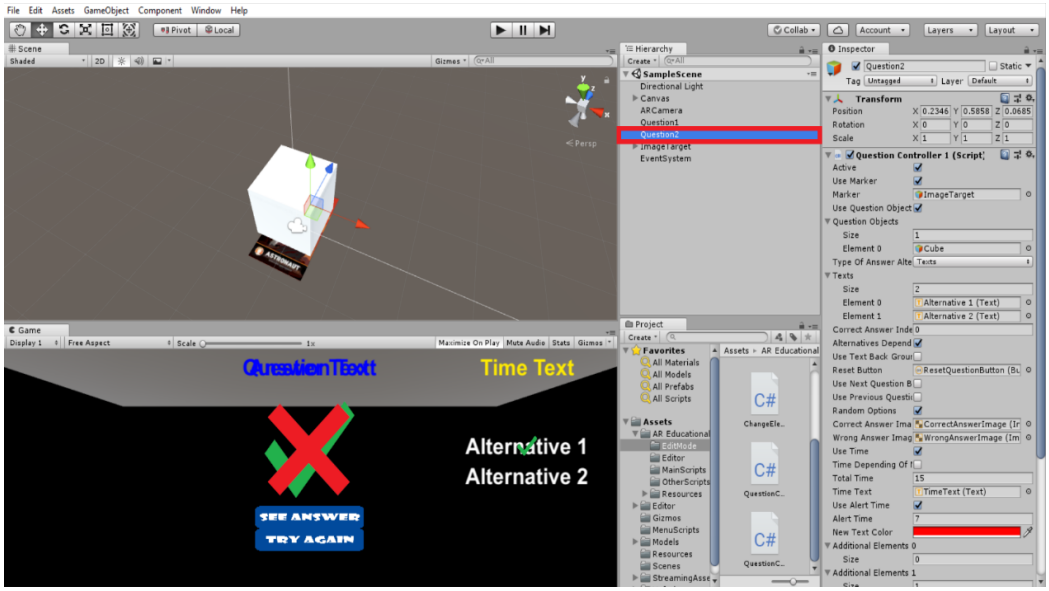

Figura D 24: Novo objeto vazio criado duplicando o original. Os objetos vazios tem sido renomeados como Question1 e Question2 respetivamente.

2) Indicar qual será a primeira das perguntas em aparecer, para isso deve-se deixar marcado o campo Active da primeira pergunta e desmarcar o das outras, para este exemplo a pergunta de Question1 vai ser a primeira em aparecer, portanto unicamente precisa-se desmarcar o campo Active de Question2, ver Figura D 25.

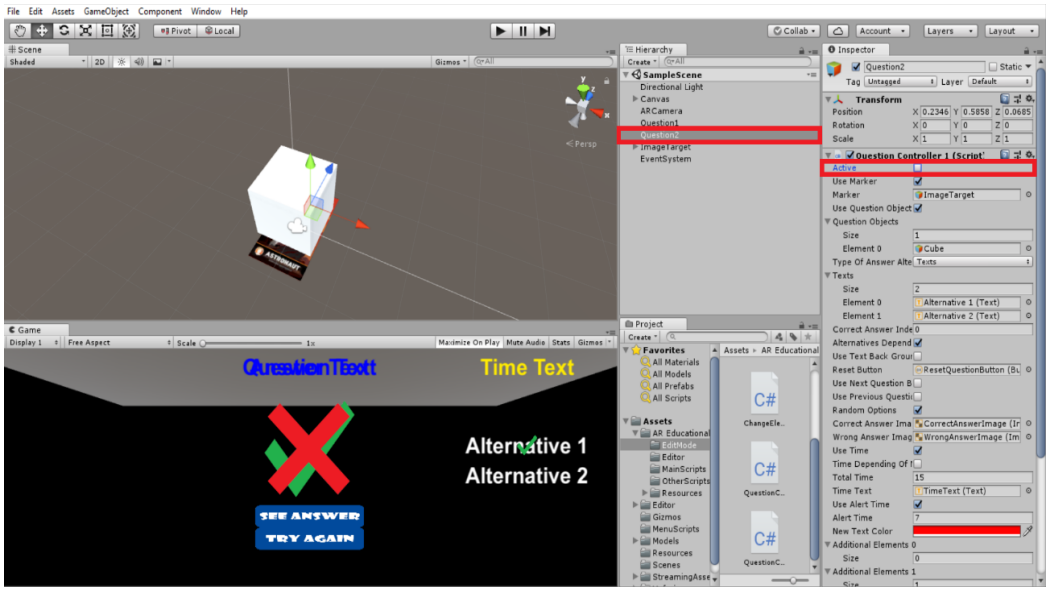

Figura D 25: Campo Active do objeto vazio Question2 desmarcado.

3) Criar textos alternativas da nova pergunta, para isso, neste exemplo são duplicados os textos alternativa existentes localizados no **`Canvas`** (Alternative1 e Alternativa2), renomeado os textos novos como Alternative1Question2 e Alternative2Question2 respetivamente. Após isso, os textos novos são arrastados nos campos Texts do objeto vazio Question2 como na Figura D 26. Nota: Com a classe **`QuestionController1`** não é suportado o reuso dos elementos usados como alternativas, e portanto deve ser feito este passo a mais criando novos textos alternativa para a nova pergunta.

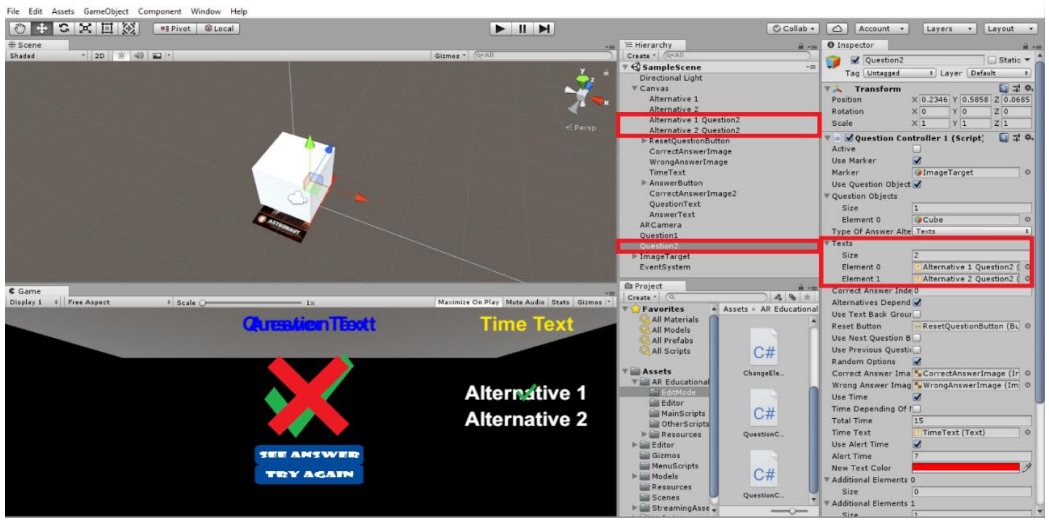

Figura D 26: Novos textos de alternativa criados e arrastados nos campos Texts de Question2.

4) Criar o botão de próxima pergunta (**`GameObject/UI/Button`**) e posicionar ele abaixo do botão TryAgain. Para isso, neste exemplo se duplica o botão ResetQuestionButton, se posiciona o novo abaixo do original na visão da cena renomeando ló como NextQuestionButton, e se arrasta a imagem Next (localizada no caminho: **`AR Educational Framework/Resources/ButtonTextures`**), no campo SourceImage do botão, ficando como na Figura D 27.

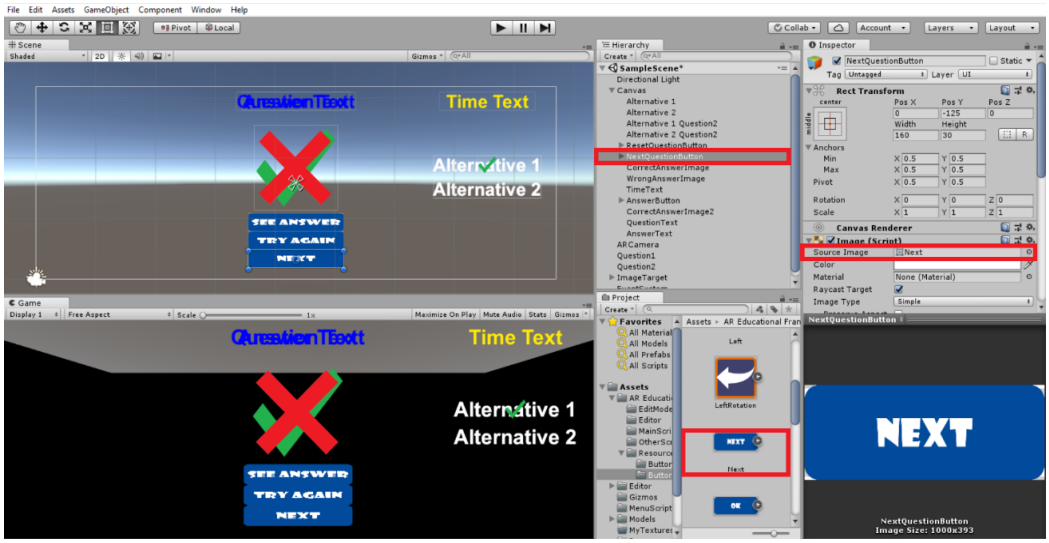

Figura D 27: Novo botão NextQuestionButton da próxima pergunta já criado, e com a imagem Next arrastada no seu campo SourceImage.

5) Selecionar o objeto vazio Question1, ativar o campo UseNextQuestionButton para usar o botão de ir à próxima pergunta, e finalmente arrastar no campo NextQuestionButton o botão criado no passo anterior com o mesmo nome, e no campo NextQuestion o objeto vazio Question2 que se refere à próxima pergunta, a cena deve ficar como na Figura D 28.

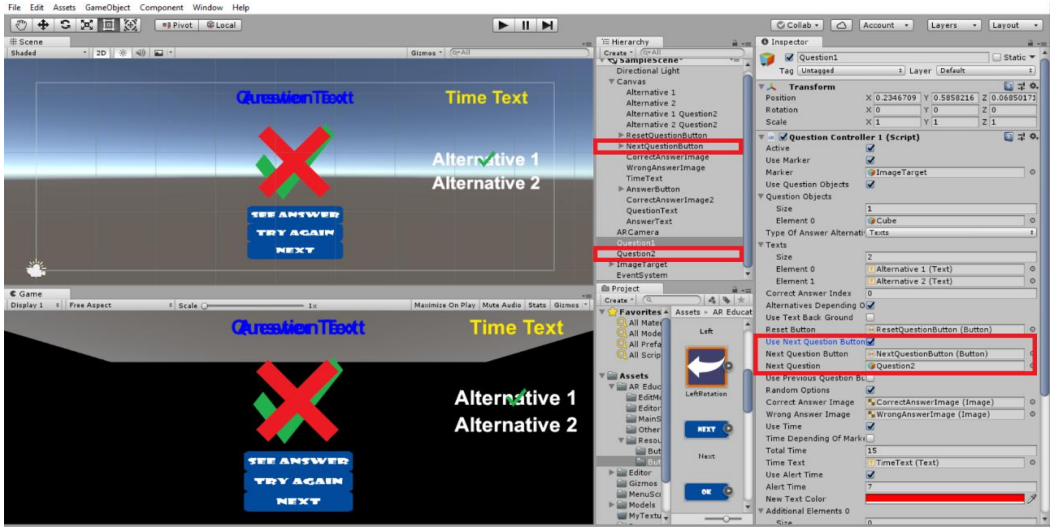

Figura D 28: Campo UseNextQuestionButton ativado no objeto vazio Question1, para usar botão de ir à próxima pergunta.

6) Finalmente, selecionar o objeto vazio Question2, e no campo CorrectAnswerIndex colocar 1 para definir o segundo texto de alternativa como a alternativa correta nesta pergunta, ver Figura D 29. A cena já deveria ficar pronta para ser executada com as duas perguntas.

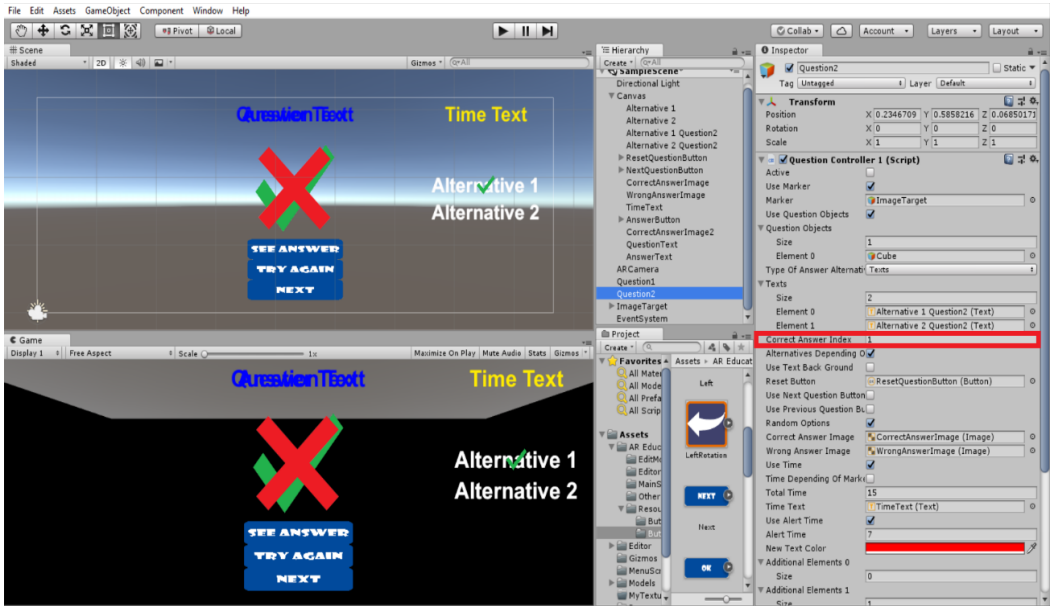

Figura D 29: Campo CorrectAnswerIndex do objeto vazio Question2 colocado em 1, e cena pronta para ser executada.

Tutorial Classe QuestionController2

Inicialização Default Da Classe QuestionController2

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController2`** com sua inicialização padrão, que vai permitir criar automaticamente uma pergunta (com elementos padrões de exemplo), na qual as alternativas são elementos 3D em RA vinculados a um marcador, e apresentados por turno. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALQUESTIONCONTROLLER2, disponível no link:

https://www.dropbox.com/s/ti2ekjvf98agcaq/PROJETOUNITYTUTORIALQUESTIONCONTROLLER2.rar?dl=0[20]

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura D 30, que será usado como container para as classes que permitirão criar a pergunta (classes **`QuestionController2`** e **`QuestionController2E`**).

Figura D 30: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController2`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura D 31.

[20] Todas as referências ao projeto "PROJETOUNITYTUTORIALQUESTIONCONTROLLER2" neste documento, se referem a este link.

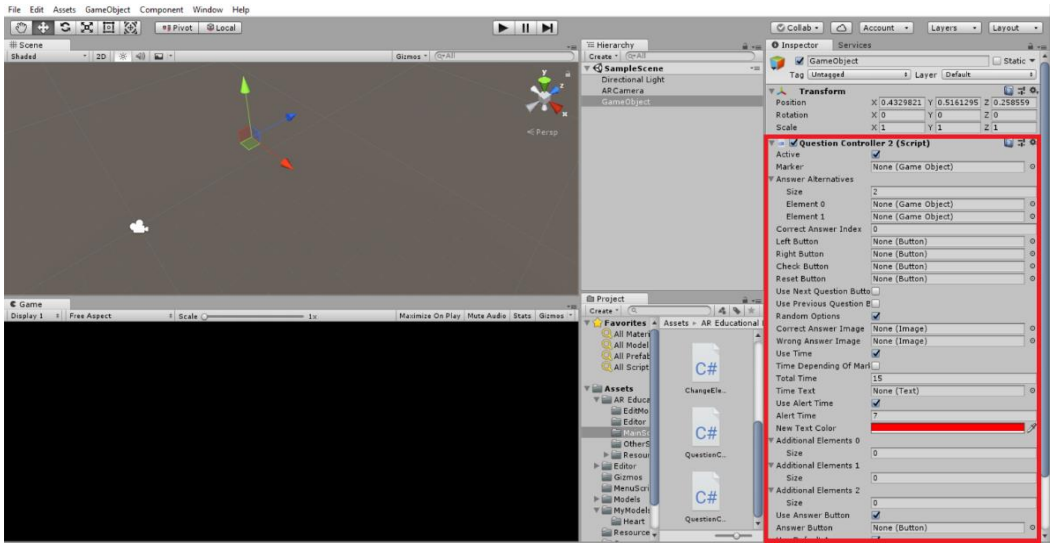

Figura D 31: Classe `QuestionController2` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController2E`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default*, após isso recomenda-se fazer click no marcador criado **`ImageTarget`** para fazer ele visível na cena.

   A cena deveria ficar como na Figura D 32, já pronta para ser executada, com um cubo e uma esfera (que representam as alternativas de resposta) sobre um marcador, no centro duas imagens de *feedback* de resposta correta e errada, com botões para ver resposta correta e repetir a pergunta, assim como botões de passar à próxima alternativa, à alternativa anterior, e escolher a alternativa atual (lembrando que as alternativas são por turno), finalmente no corner superior direito está localizado um texto para mostrar o tempo, e na parte superior da tela dois textos adicionais para pergunta e resposta respetivamente.

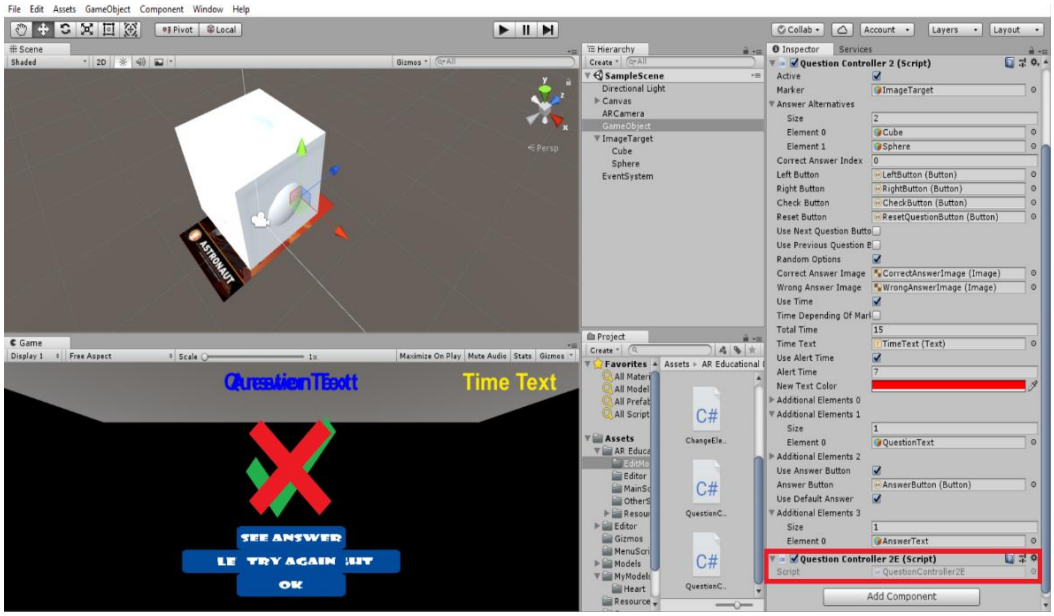

Figura D 32: Classe `QuestionController2E` arrastada no objeto vazio, e cena já pronta para ser executada.

4) Após ser feita a inicialização padrão recomenda-se remover a classe **`QuestionController2E`**.

Configurar Classe QuestionController2 Para Usar Modelos 3D Próprios

A seguir, serão apresentados os passos para aplicar a mesma funcionalidade com dois novos modelo 3D usados para as alternativas de resposta, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALQUESTIONCONTROLLER2.

1) Importar os pacotes Heart e MODEL2 no projeto, que possuem os modelo necessários Heart e MODEL2 respetivamente. O pacote MODEL2 está disponível no link: https://www.dropbox.com/s/o65m4c7fm4xzter/MODEL2.unitypackage?dl=0
2) Substituir os novos modelos 3D Heart (caminho **`MyModels/Heart`**) e MODEL2 (caminho **`MyModels/model2`**), pelo cubo e esfera

respetivamente, para isso: Colocar cada novo modelo na mesma posição do original, e fazer eles filhos do mesmo marcador. A cena deveria ficar como na Figura D 33.

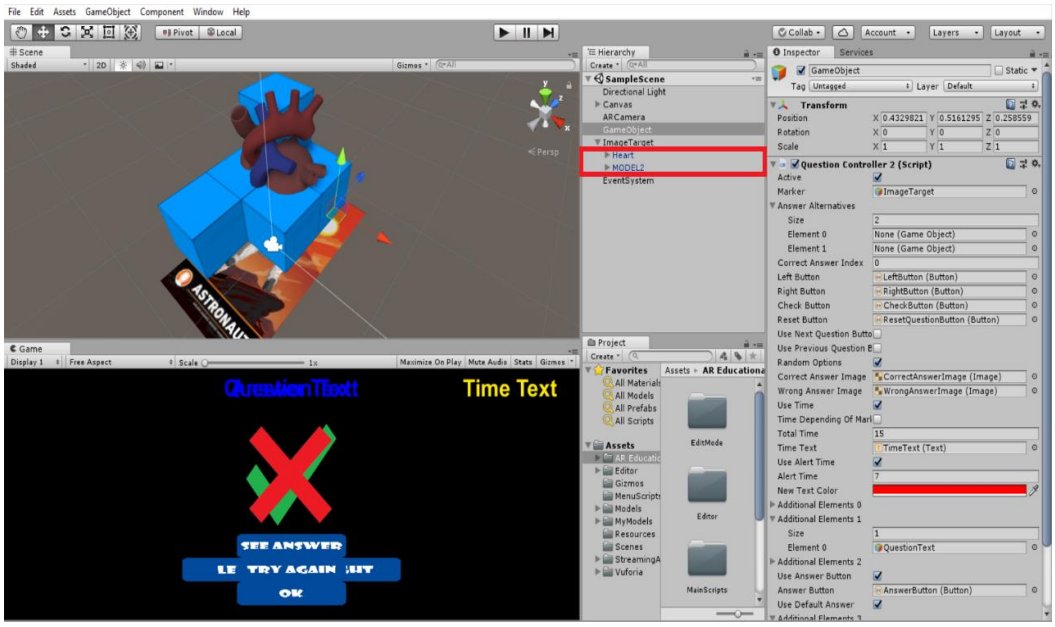

Figura D 33: Novos modelos 3D (Heart e MODEL2), vinculados ao marcador `ImageTarget`.

3) Finalmente arrastar os novos modelos 3D, nos campos AnswerAlternatives da classe **`QuestionController2`**. A cena deveria ficar como na Figura D 34, já pronta para ser executada.

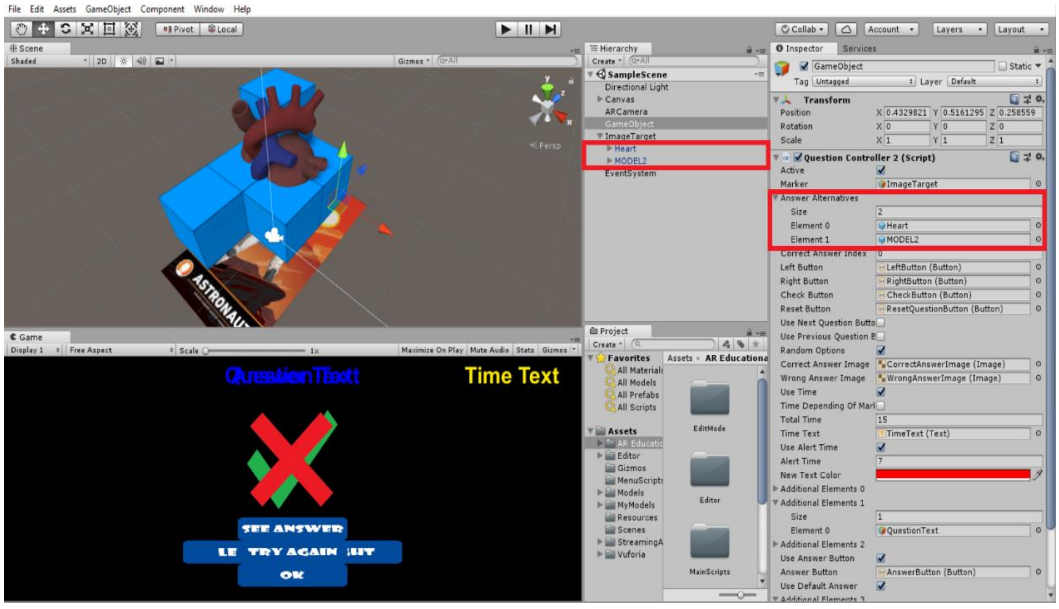

Figura D 34: Novos modelos 3D arrastados no campos AnswerAlternatives, e cena já pronta para ser executada.

Nota: Para suportar múltiplas perguntas com a classe **`QuestionController2`**, seguir os mesmo passos descritos no “Tutorial Classe **`QuestionController1`**”, em: “Configurar Classe **`QuestionController1`** Para Usar Múltiplas Perguntas”.

Tutorial Uso Classe ButtonsPack Em Perguntas

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController1`** com sua inicialização padrão para criar automaticamente uma pergunta, e vai ser explicado o jeito de usar a classe **`ButtonsPack`** para criar um pacote de botões para rotacionar e escalar o objeto 3D usado na pergunta. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALBUTTONSPACKEMPERGUNTAS, disponível no link:

https://www.dropbox.com/s/h7gx2da4buuvdcg/PROJETOUNITYTUTORIALBUTTONSPACKEMPERGUNTAS.rar?dl=0[21]

Criar Pergunta Com Classe QuestionController1

1) Criar uma pergunta com a classe **`QuestionController1`** e sua inicialização padrão, como explicado no “Tutorial Classe **`QuestionController1`**”, em: “Inicialização Default Da Classe **`QuestionController1`**”. A cena deveria ficar como na Figura D 35.

[21] Todas as referências ao projeto “PROJETOUNITYTUTORIALBUTTONSPACKEMPERGUNTAS” neste documento, se referem a este link.

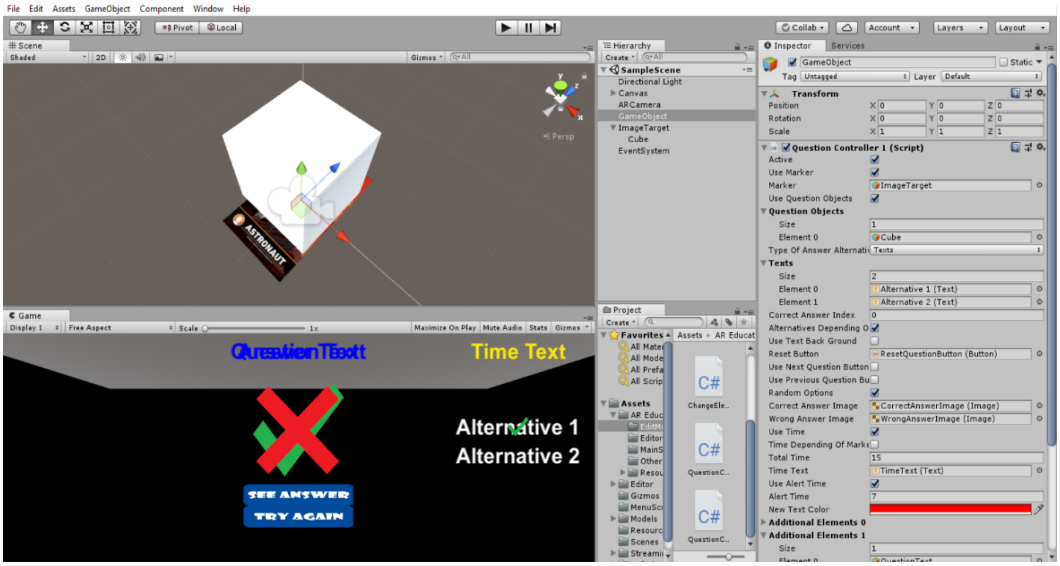

Figura D 35: Pergunta criada com `QuestionController1`.

Criar Pacote De Botões Com Classe ButtonsPack

1) Usar a classe **`ButtonsPack`** e sua inicialização padrão para criar um pacote de botões, objeto cubo e seu marcador *default*, como explicado no "Tutorial Classe **`ButtonsPack`**", em: "Inicialização Default Da Classe **`ButtonsPack`**". Lembre-se que o objeto vazio com a classe **`ButtonsPack`** deveria ser removido, já o objeto vazio com **`QuestionController1`** será renomeado como Question1. A cena deveria ficar como na Figura D 36.

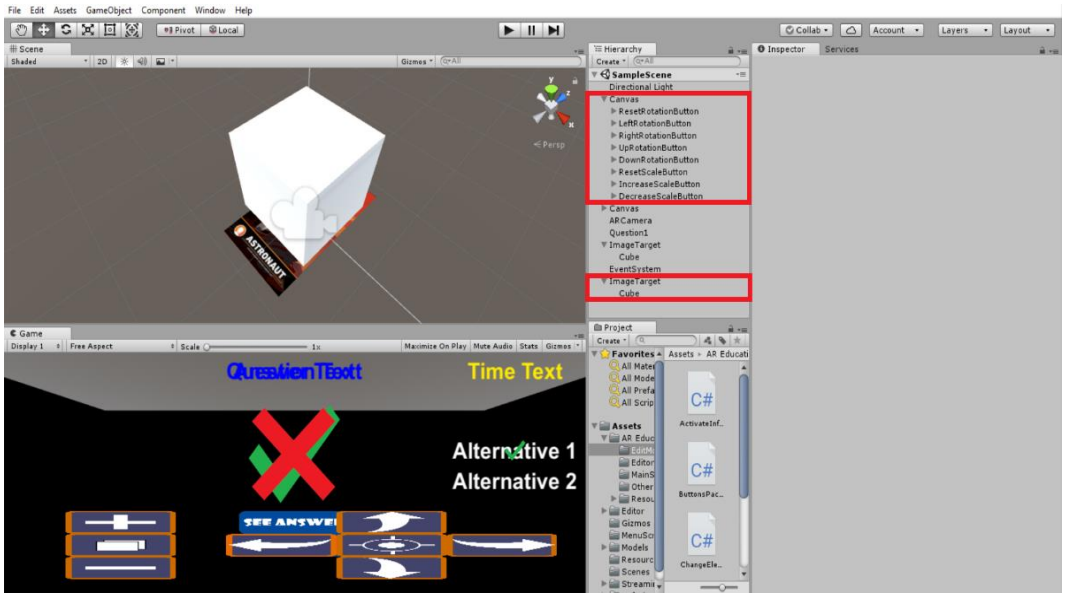

Figura D 36: Pacote de botões, novo objeto cubo e marcador criados com `ButtonsPack`. Objeto vazio com `ButtonsPack` removido e objeto vazio com `QuestionController1` renomeado para Question1.

Atualizar Botões Para Ficar Associados Com Elementos Da Pergunta

Nota-se que a classe **`ButtonsPack`** com sua inicialização padrão, criou elementos próprios para os botões (novos marcador **`ImageTarget`** e objeto Cube, indicados na Figura D 36), ainda ficando na cena alguns elementos duplicados, portanto os passos a seguir consistem em fazer os botões ficar associados com os elementos da pergunta (marcador **`ImageTarget`** original e modelo Cube original), e não com seus próprios elementos padrões, após isso, os elementos desnecessários serão removidos.

1) Atualizar os botões para usar o mesmo marcador da pergunta, para fazer isso, para cada botão deve-se arrastar o marcador original **`ImageTarget`** (da pergunta), no seu campo Markers da classe **`ButtonOperations`**. Para fazer isso rapidamente, são selecionados todos os botões no mesmo tempo mantendo a tecla CTRL pressionada, e finalmente se arrasta o **`ImageTarget`** da pergunta no campo Markers da classe **`ButtonOperations`**, ver processo na Figura D 37.

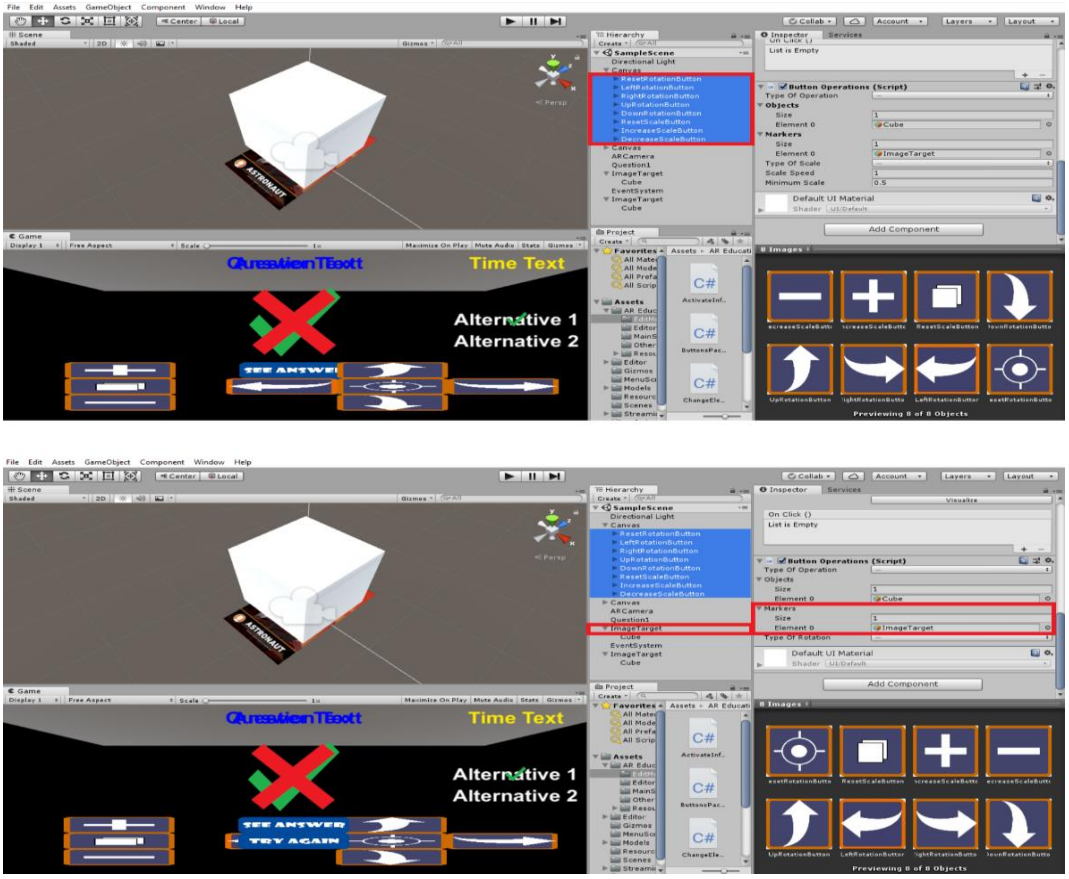

Figura D 37: Inicialmente todos os botões são selecionados no mesmo tempo com a tecla CTRL, e finalmente o marcador da pergunta é arrastado no campo Markers.

2) Atualizar os botões para usar o mesmo modelo 3D da pergunta, para fazer isso, para cada botão deve-se arrastar o modelo Cube original (da pergunta) no seu campo Objects da classe **`ButtonOperations`**, de jeito parecido como explicado no passo anterior, ver processo Figura D 38.

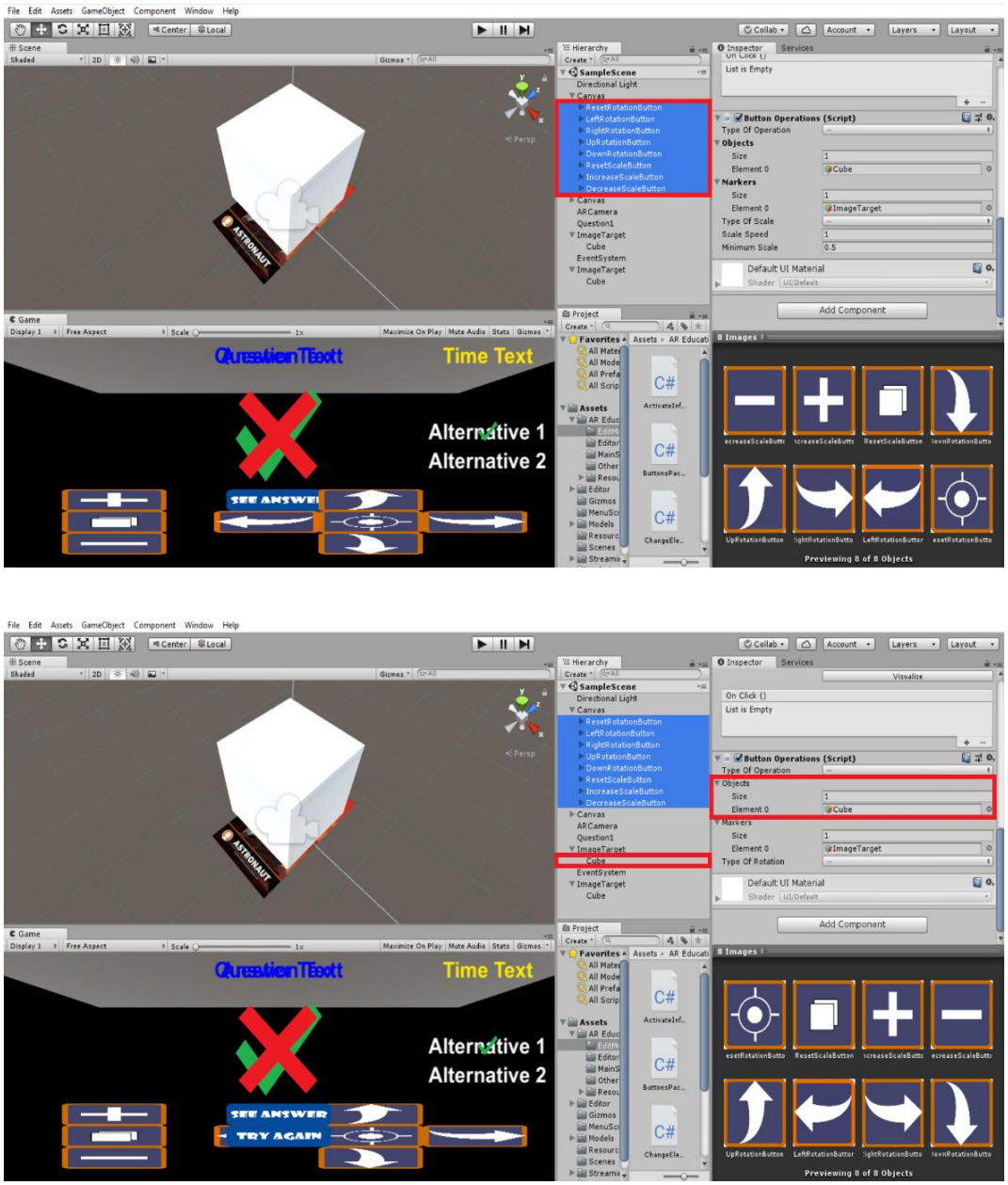

Figura D 38: Inicialmente todos os botões são selecionados no mesmo tempo com a tecla CTRL, e finalmente o Cube da pergunta é arrastado no campo Objects.

3) Remover o objeto Cube novo e seu marcador **`ImageTarget`** que não serão mais necessários. Finalmente a cena já deveria ficar pronta para ser executada, como na Figura D 39.

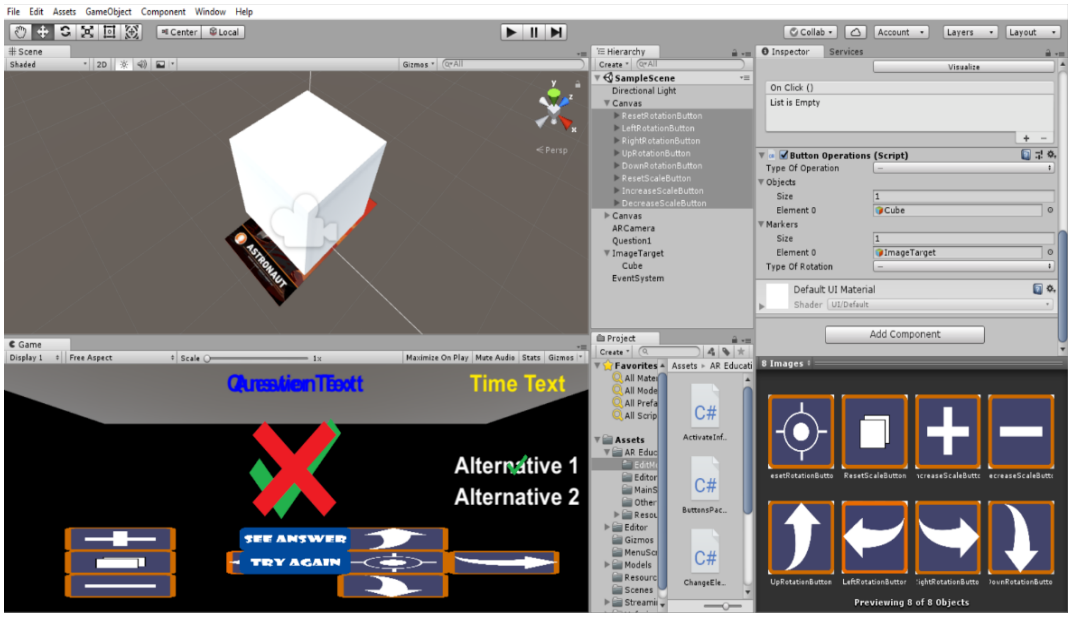

Figura D 39: Objeto Cube novo com seu marcador já removidos, e cena pronta para ser executada.

Vincular Botões A Estados Da Pergunta

No caso de ser necessário, é possível fazer que os botões apareçam unicamente em alguns dos estados da pergunta, isso é feito arrastando eles nos campos AdditionalElements da classe **`QuestionController1`** da pergunta, as instruções serão apresentadas as seguir:

Nota: Uma cena final chamada Scene2, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALBUTTONSPACKEMPERGUNTAS.

1) Neste passo deve-se definir em qual dos estados da pergunta queremos que os botões apareçam, nota-se que existem 4 estados disponíveis para uma pergunta, chamados: Estado0, Estado1, Estado2 e Estado3, suportados através dos campos: AdditionalElements0, AdditionalElements1, AdditionalElements2, e AdditionalElements3, da classe **`QuestionController1`**. A seguir são detalhados estes campos:
    - AdditionalElements0: Usado para elementos ativos durante a pergunta toda (Estado 0).

- AdditionalElements1: Usado para elementos ativos unicamente quando está sendo respondida a pergunta (Estado 1).
- AdditionalElements2: Usado para elementos ativos unicamente após responder a pergunta (Estado 2).
- AdditionalElements3: Usado para elementos ativos unicamente quando está sendo apresentada a resposta correta da pergunta (Estado 3).

Na Figura D 40, são indicados os campos AdditionalElements0, AdditionalElements1, AdditionalElements2, AdditionalElements3 para a pergunta do objeto vazio Question1.

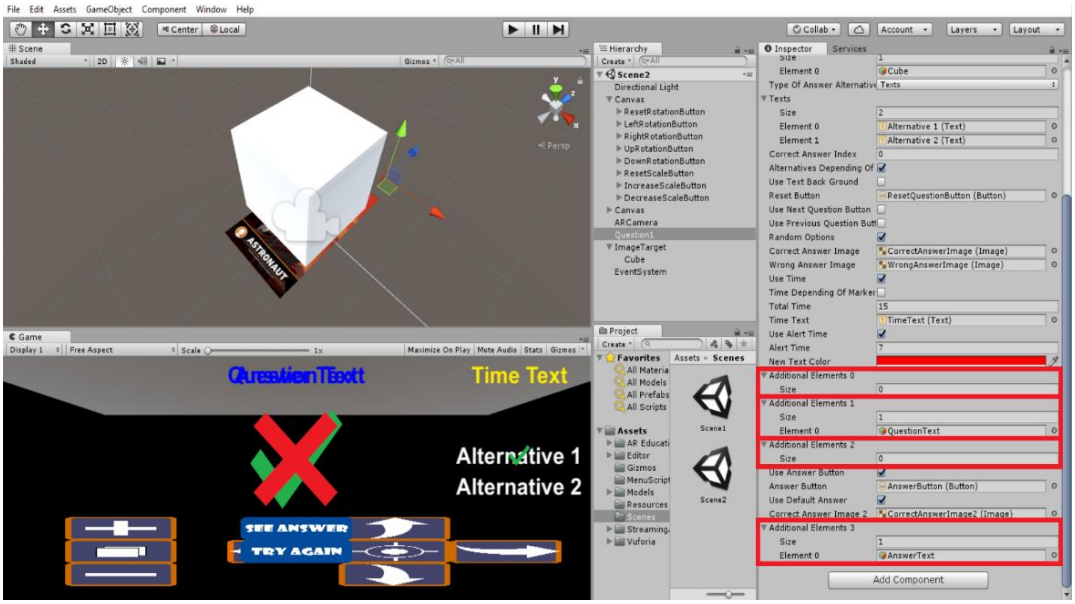

Figura D 40: Campos AdditionalElements para a pergunta do objeto vazio Question1.

Para este exemplo os botões são definidos no Estado1, e portanto eles serão arrastados em campos AdditionalElements1 do objeto vazio Question1.

2) O campo Size permite definir a quantidade de elementos em AdditionalElements1, neste caso já existe um elemento em AdditionalElements1 e precisa-se de oito elementos a mais (referentes à quantidade dos botões que vamos arrastar), portanto coloca-se o valor de 9 em Size, ver Figura D 41.

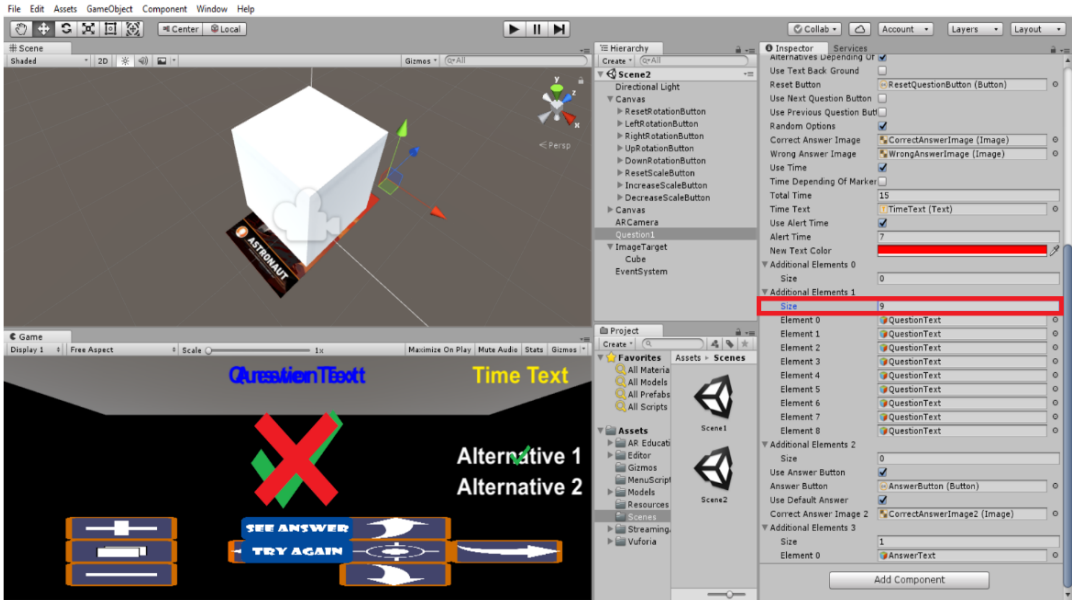

Figura D 41: Campo Size de AdditionalElements1 ajustado a 9 para suportar todos os botões.

3) Finalmente, deve-se arrastar cada botão um por um, nos campos criados no passo anterior. A cena deveria ficar como na Figura D 42, já pronta para ser executada, e com os botões aparecendo unicamente no Estado1 da pergunta (ou seja, enquanto ela está sendo respondida).

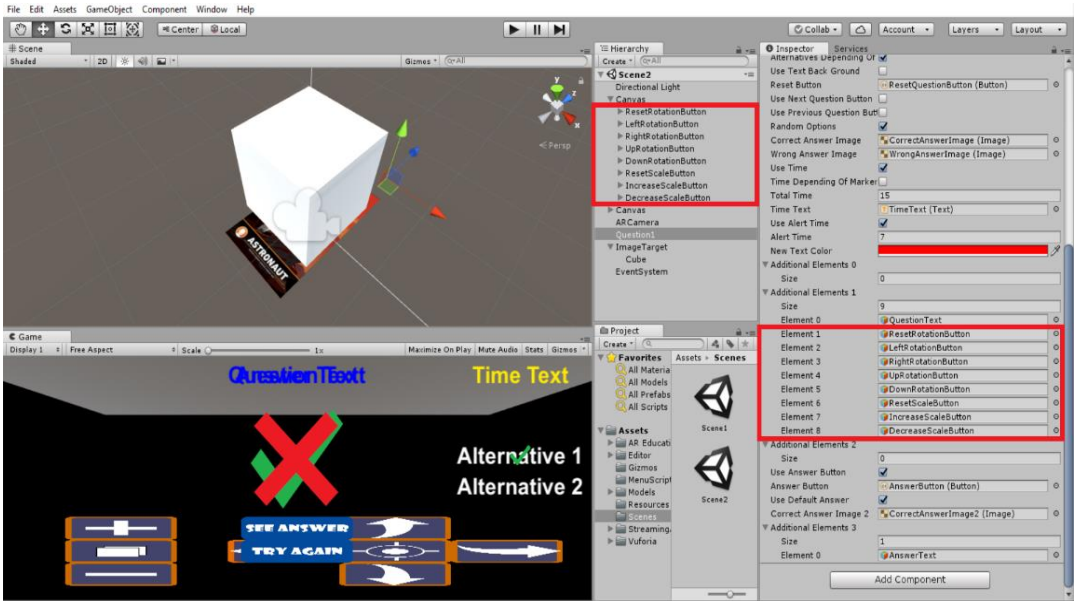

Figura D 42: Botões já arrastados nos campos AdditionalElements1, e cena pronta para ser executada.

Tutorial Uso Classe ChangeElement Em Perguntas

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController1`** com sua inicialização padrão para criar automaticamente uma pergunta, e vai ser explicado o jeito de usar a classe **`ChangeElement`** para criar um evento de mudar cor no objeto 3D da pergunta. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALCHANGEELEMENTEMPERGUNTAS, disponível no link:

https://www.dropbox.com/s/2zzb26itncru0z9/PROJETOUNITYTUTORIALCHANGEELEMENTEMPERGUNTAS.rar?dl=0

Criar Pergunta Com Classe QuestionController1

1) Criar uma pergunta com a classe **`QuestionController1`** e sua inicialização padrão, como explicado no "Tutorial Classe **`QuestionController1`**", em: "Inicialização Default Da Classe **`QuestionController1`**". A cena deveria ficar como na Figura D 43.

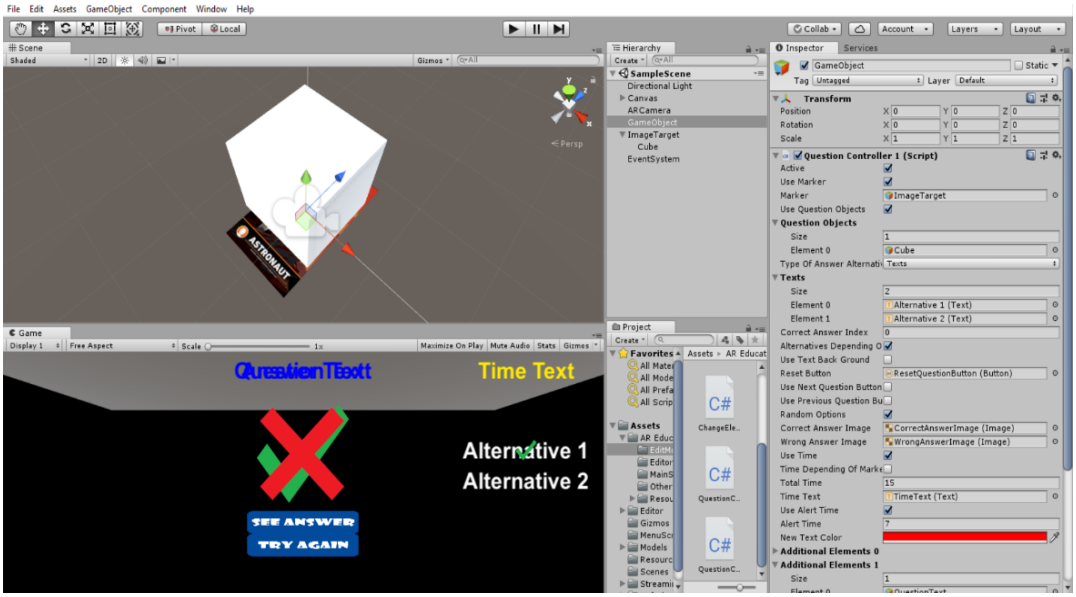

Figura D 43: Pergunta criada com `QuestionController1`.

Criar Evento Com Classe ChangeElement

1) Criar um evento com a classe **`ChangeElement`** e sua inicialização padrão, como explicado no “Tutorial Classe **`ChangeElement`**”, em: “Inicialização Default Da Classe **`ChangeElement`**”. Para melhor compreensão, os objetos vazios container que foram criados são renomeados como Question1 e ChangeElement respetivamente, a cena deveria ficar como na Figura D 44.

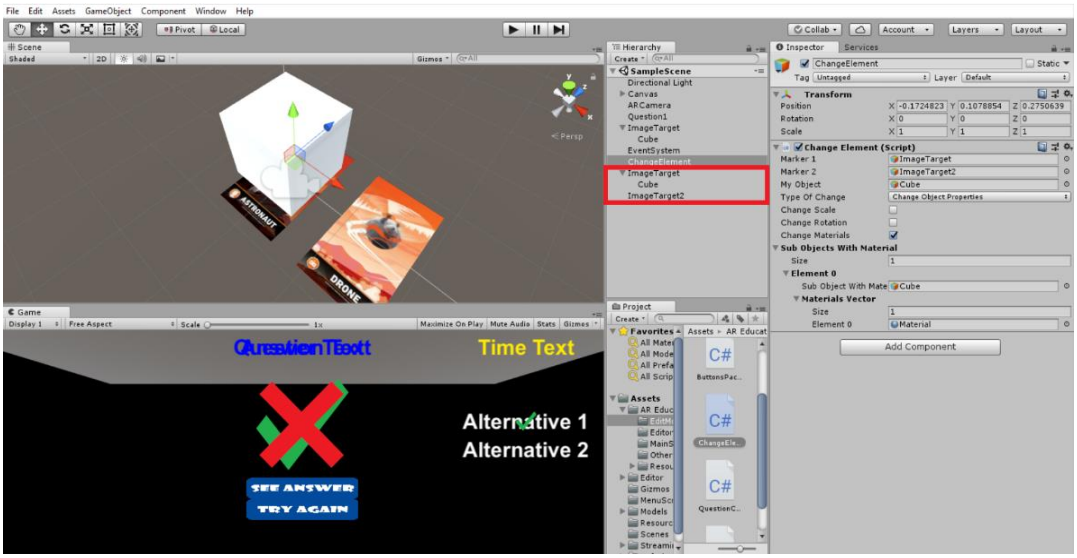

Figura D 44: Novo objeto Cube e marcadores criados com `ChangeElement`. Os dois objetos vazios container foram renomeados para Question1 e ChangeElement.

Atualizar Classe ChangeElement Para Ficar Associada Com Elementos Da Pergunta

Nota-se que a classe **`ChangeElement`** com sua inicialização padrão, criou seus próprios elementos para seu evento (marcadores **`ImageTarget`** novo, **`ImageTarget2`** e objeto Cube novo, indicados na Figura D 44), ainda ficando na cena alguns elementos duplicados, portanto os passos a seguir consistem em fazer a classe **`ChangeElement`** ficar associada com os elementos da pergunta (marcador **`ImageTarget`** original e modelo Cube original), e não com seus próprios elementos padrões, após isso, os elementos desnecessários serão removidos.

1) Atualizar a classe **`ChangeElement`** para usar o mesmo marcador da pergunta, para isso arrasta-se o marcador original **`ImageTarget`** da

pergunta no campo Marker1 da classe **`ChangeElement`**, como na Figura D 45.

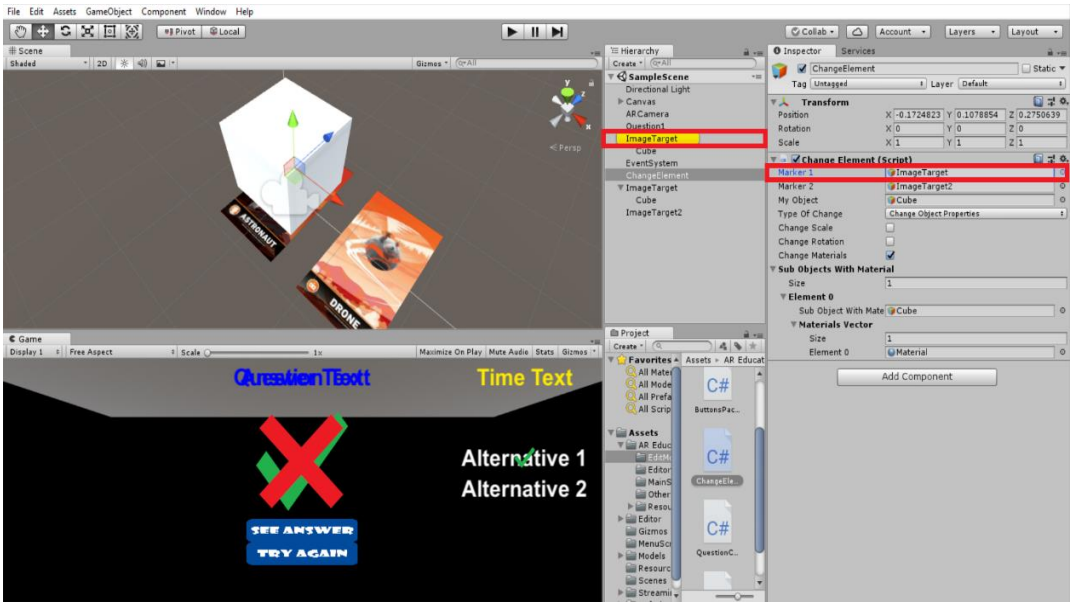

Figura D 45: Marcador `ImageTarget` da pergunta arrastado no campo Marker1 da classe `ChangeElement`.

2) Atualizar a classe **`ChangeElement`** para usar o mesmo modelo 3D da pergunta. Para isso, inicialmente arrasta-se o modelo Cube original no campo MyObject da classe, ver Figura D 46.

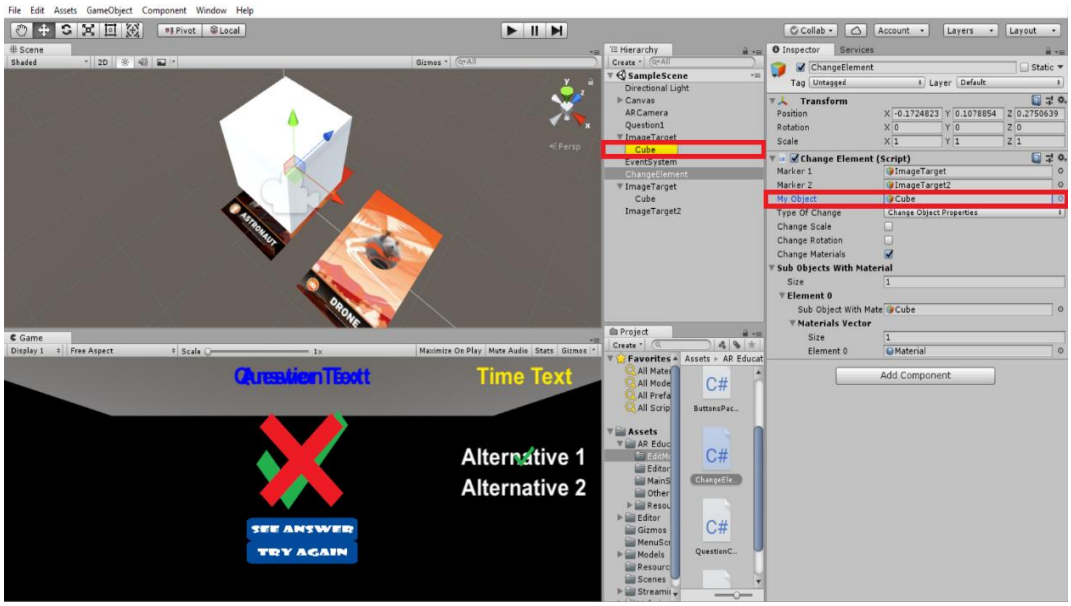

Figura D 46: Modelo Cube original arrastado no campo MyObject da classe `ChangeElement`.

Após isso, arrasta-se também o modelo Cube original no campo SubObjectWithMaterial da classe, ver Figura D 47.

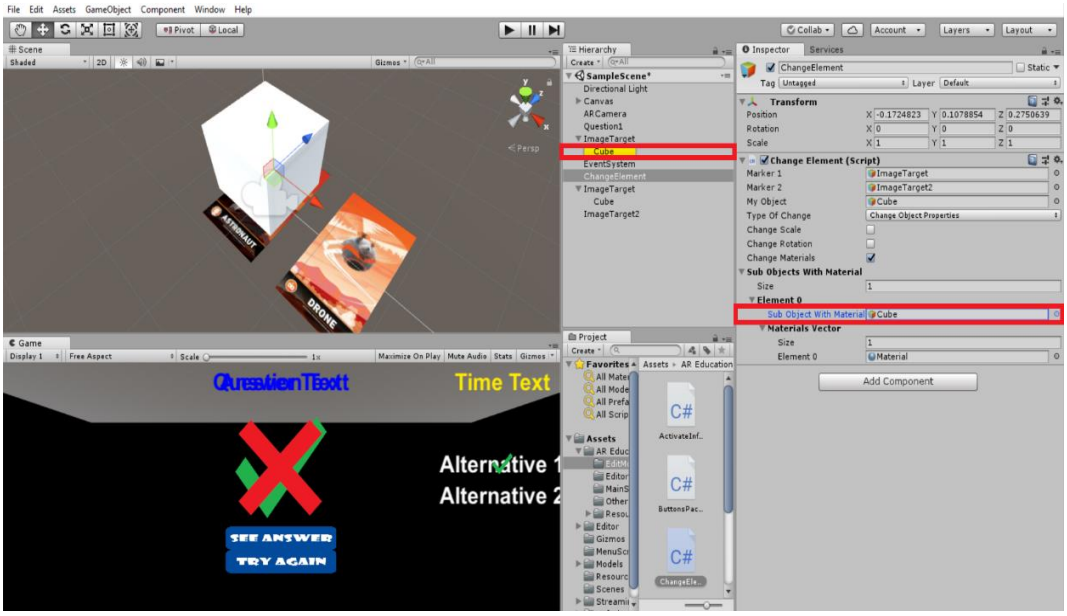

Figura D 47: Modelo Cube original arrastado no campo SubObjectWithMaterial da classe `ChangeElement`.

3) Finalmente, remover o objeto Cube novo e seu marcador **`ImageTarget`** que não serão mais necessários. Nota-se que o marcador **`ImageTarget2`** continuará sendo necessário e não deve ser removido. A cena deveria ficar como na Figura D 48, pronta para ser executada.

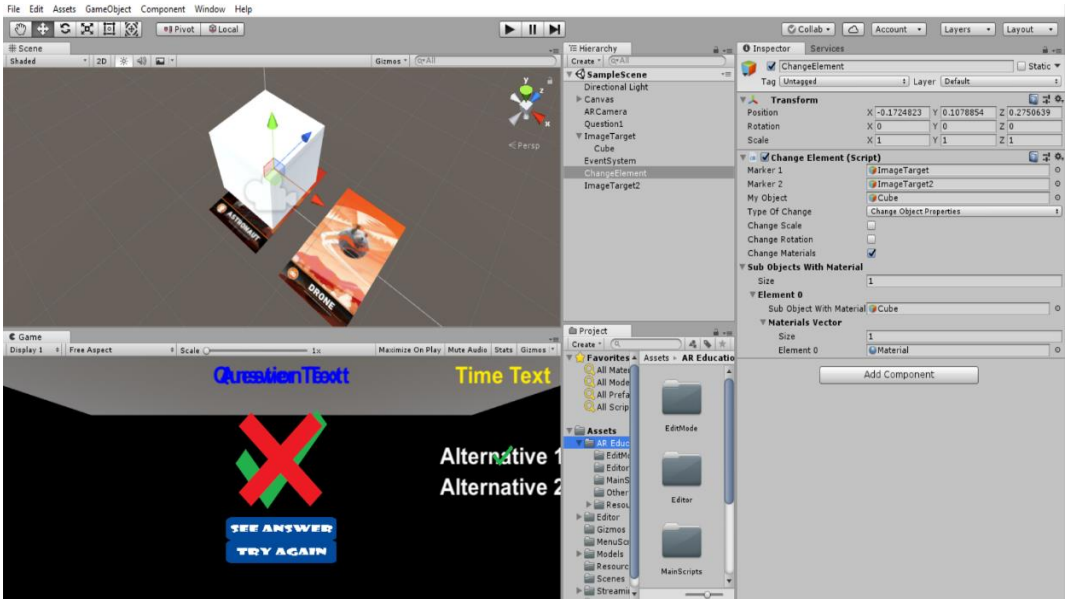

Figura D 48: Modelo Cube novo e seu marcador `ImageTarget` removidos, e cena pronta para ser executada.

Tutorial Uso Classe ActivateInformation Em Perguntas

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController1`** com sua inicialização padrão para criar automaticamente uma pergunta, e vai ser explicado o jeito de usar a classe **`ActivateInformation`** para criar um evento de ativar um texto sobre o objeto 3D da pergunta. Para isso, devem-se seguir os seguintes passos:

Nota: Uma cena final chamada Scene1, obtida aplicando os passos a seguir, está disponível no projeto Unity chamado PROJETOUNITYTUTORIALACTIVATEINFORMATIONEMPERGUNTAS, disponível no link:

https://www.dropbox.com/s/24ta8tw6szeat8x/PROJETOUNITYTUTORIALACTIVATEINFORMATIONEMPERGUNTAS.rar?dl=0

Criar Pergunta Com Classe QuestionController1

1) Criar uma pergunta com a classe **`QuestionController1`** e sua inicialização padrão, como explicado no "Tutorial Classe **`QuestionController1`**", em: "Inicialização Default Da Classe **`QuestionController1`**". A cena deveria ficar como na Figura D 49.

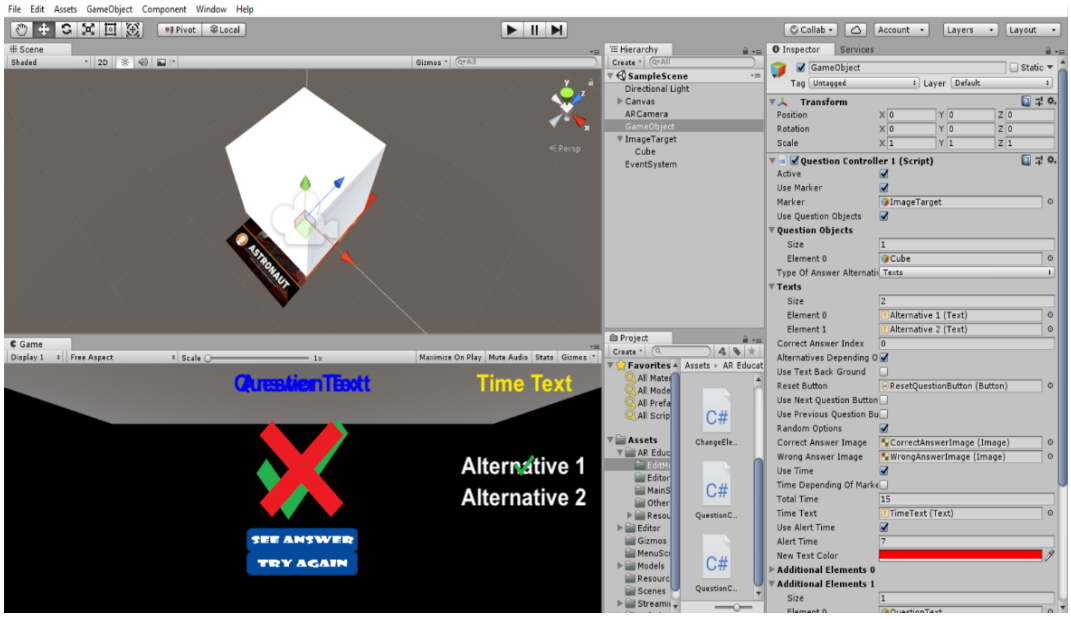

Figura D 49: Pergunta padrão criada com `QuestionController1`.

Criar Evento Com Classe ActivateInformation

1) Criar um evento com a classe **`ActivateInformation`** e sua inicialização padrão, como explicado no "Tutorial Classe **`ActivateInformation`**" em: "Inicialização Default Da Classe **`ActivateInformation`**". Para melhor compreensão, os objetos vazios container que foram criados são renomeados como Question1 e ActivateInformation respetivamente, a cena deveria ficar como na Figura D 50.

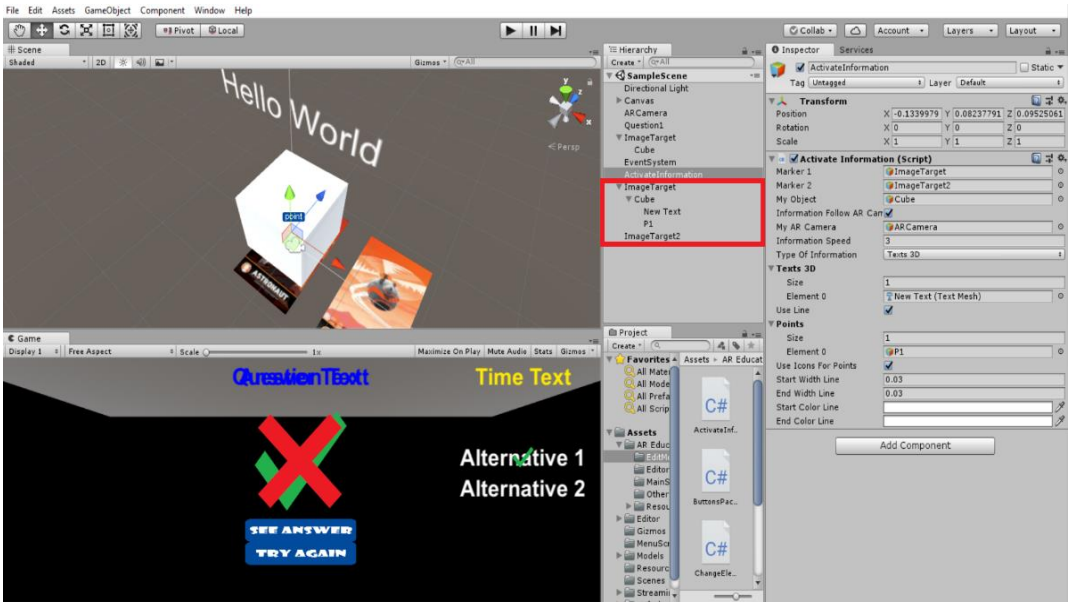

Figura D 50: Novo objeto cubo, texto 3D e marcadores criados com `ActivateInformation`. Os dois objetos vazios container foram renomeados para Question1 e ActivateInformation.

Atualizar Classe ActivateInformation Para Ficar Associada Com Elementos Da Pergunta

Nota-se que a classe **`ActivateInformation`** com sua inicialização padrão, criou seus próprios elementos para seu evento (marcadores **`ImageTarget`** novo, **`ImageTarget2`** e objeto Cube novo, indicados na Figura D 50), ainda ficando na cena alguns elementos duplicados, portanto os passos a seguir consistem em fazer a classe **`ActivateInformation`** ficar associada com os elementos da pergunta

(marcador **`ImageTarget`** original e objeto Cube original), e não com seus próprios elementos padrões, após isso, os elementos desnecessários serão removidos.

1) Atualizar a classe **`ActivateInformation`** para usar o mesmo marcador da pergunta, para isso arrasta-se o marcador original **`ImageTarget`** da pergunta no campo Marker1 da classe **`ActivateInformation`**, como na Figura D 51.

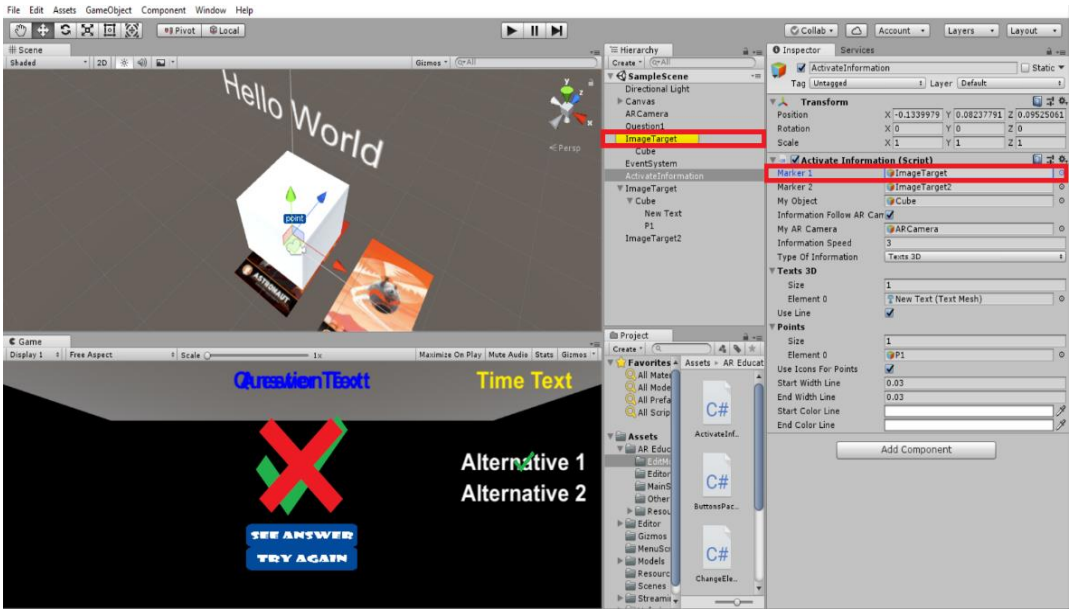

Figura D 51: Marcador `ImageTarget` da pergunta arrastado no campo Marker1 da classe `ActivateInformation`.

2) Atualizar a classe **`ActivateInformation`** para usar o mesmo modelo 3D da pergunta, para isso arrasta-se o modelo Cube original no campo MyObject da classe, ver Figura D 52.

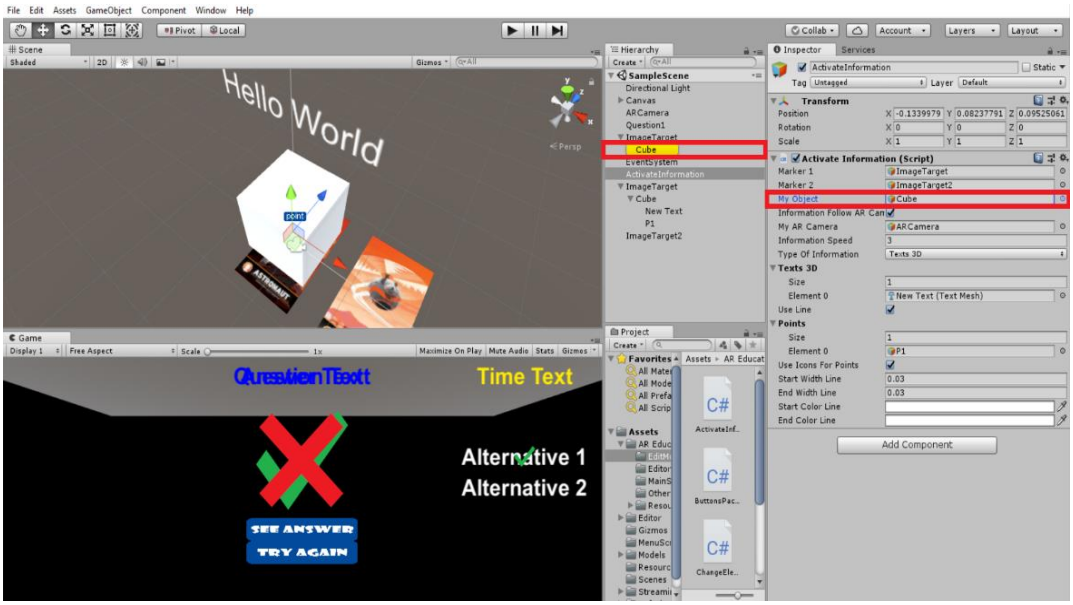

Figura D 52: Modelo Cube original arrastado no campo MyObject da classe `ActivateInformation`.

3) Fazer filhos do modelo Cube da pergunta, aos elementos filhos do Cube novo criado com a classe **`ActivateInformation`** (elementos newText e P1). Ver Figura D 53.

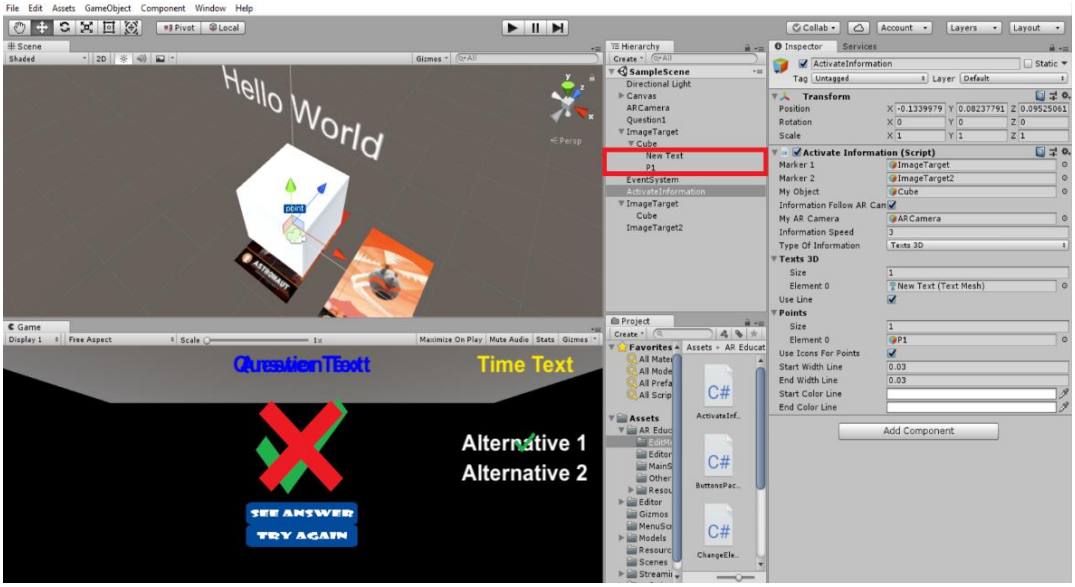

Figura D 53: Elementos newText e P1 que eram filhos do modelo Cube novo, agora são filhos do modelo Cube da pergunta.

4) Remover o objeto Cube novo e seu marcador **`ImageTarget`** que não serão mais necessários. Nota-se que o marcador **`ImageTarget2`**

continuará sendo necessário e não deve ser removido. Finalmente a cena já deveria ficar pronta para ser executada, como na Figura D 54.

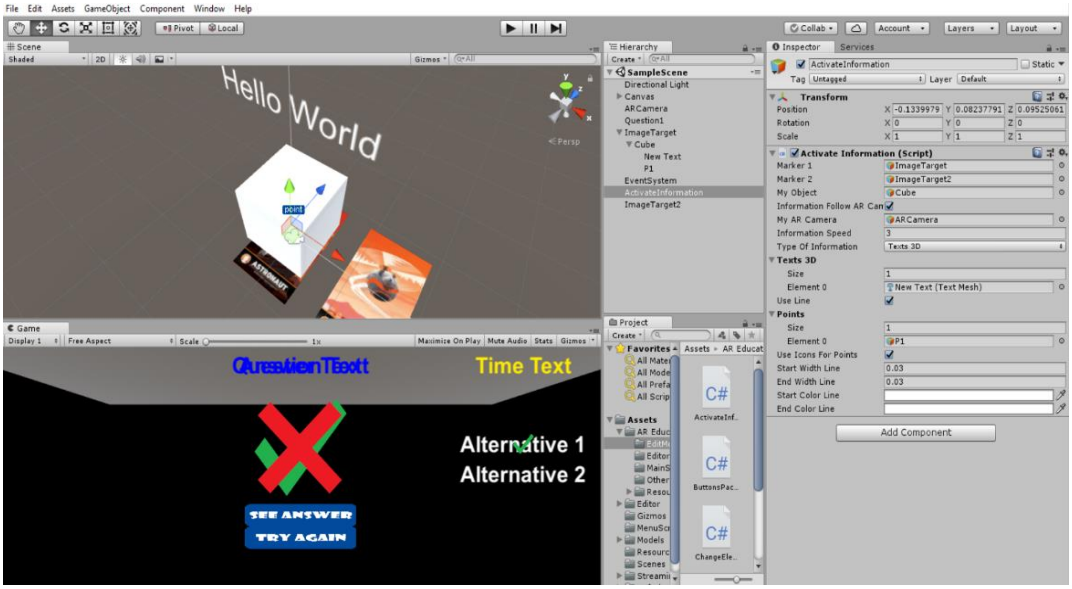

Figura D 54: Objeto Cube novo com seu marcador já removidos, e cena pronta para ser executada.

Apêndice E

INSTRUÇÕES APLICATIVO ESTUDO DE CASO

São apresentadas a seguir as instruções do uso para o aplicativo do estudo de caso realizado, com seus dois cenários.

Uma vez o usuário acessar no aplicativo, será apresentado o menu de cenários inicial, o qual será detalhado a seguir.

Menu de Cenários: Este menu consiste no menu principal do aplicativo e permitirá ao usuário, escolher o cenário desejado pressionando seu botão respetivo. O botão “1” (da esquerda) permite ir ao Cenário 1 (do área de Geometria, e pensamento espacial), e o botão “2” ir ao Cenário 2 (do área de Biologia, e do estudo do coração humano), nota-se que existe um botão para sair do aplicativo (botão “SAIR”), o qual estará presente em ambos os cenários. Ver Figura E 1.

Figura E 1: Menu de canários do aplicativo.

Inicialmente serão apresentadas as instruções do Cenário 1, e após isso, as instruções do Cenário 2.

Cenário 1

Se for selecionado o Cenário 1 no menu de cenários, o usuário ficará no menu de perguntas do Cenário 1 detalhado a seguir.

Menu De Perguntas Do Cenário 1: Este menu permite escolher uma das quatro perguntas disponíveis no Cenário, pressionando seu botão respetivo, ver Figura E 2.

Figura E 2: Menu de perguntas do Cenário 1.

Após pressionar algum dos botões das perguntas, o usuário ficará na fase de perguntas do Cenário 1, que será detalhada a seguir.

Fase De Perguntas (Sem RA) Do Cenário 1: Nesta fase, são apresentadas as perguntas que devem ser respondidas pelo usuário neste cenário, em cada pergunta é apresentada uma imagem 2D de uma figura geométrica sobre a qual vai ser realizada a pergunta, e três imagens das alternativas representando possíveis visões da figura (visões da frente, topo, e lado), então o usuário deverá escolher a única imagem alternativa que representa uma visão certa da figura em questão. Na Figura E 3 se apresenta esta fase com a primeira pergunta. Os elementos presentes nesta fase são detalhados a seguir, e são numerados também na Figura E 3.

Texto de pergunta: Consiste no texto que informa qual é a pergunta a realizar, ver Figura E 3 elemento número 1.

Figura geométrica da pergunta: Imagem 2D da figura geométrica sobre a qual vai ser realizada a pergunta, e que deve ser analisada pelo usuário em todas as suas visões (visões da frente, topo, e lado) antes de responder, ver Figura E 3 elemento número 2.

Imagens das alternativas: Imagens representando possíveis visões (visões da frente, topo, e lado) da figura geométrica da pergunta, e das quais o usuário deverá escolher a única alternativa correta pressionando sobre ela (alternativa que representa uma visão certa da figura). As imagens alternativas são apresentadas na Figura E 3 elementos número 3. Nota-se que as imagens das alternativas irão aparecer sempre em ordem aleatória.

Texto do Tempo: Consiste no texto que apresenta o tempo disponível para realizar a pergunta, ver Figura E 3 elemento número 4. O tempo máximo das perguntas deste cenário é de 25 segundos.

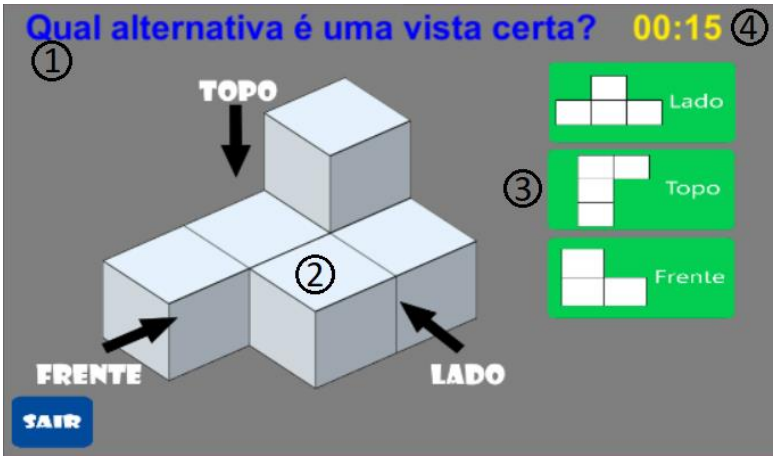

Figura E 3: Fase de Perguntas do Cenário 1 com a primeira pergunta, os elementos desta fase foram numerados para maior detalhe.

Após responder a pergunta (pressionando alguma das imagens das alternativas), uma imagem de *feedback* de resposta correta ou errada, será mostrada para o usuário informando-lhe se a sua resposta está correta ou errada, e ficando como na Figura E 4. Nota-se que finalizar o tempo também será considerado pelo aplicativo como resposta errada.

Figura E 4: Imagens de *feedback* de resposta correta (figura esquerda), e resposta errada (figura direita).

Vários botões de navegação ficam também disponíveis juntos com as imagens, eles são: botão “VER RESPOSTA”, que permite ir à fase de exploração e ver a resposta certa, botão “REPETIR” que permite repetir a pergunta, e o botão “SEGUINTE”, para ir à próxima pergunta diretamente (sem precisar voltar ao menu de perguntas para selecionar a próxima).

Após pressionar o botão “VER RESPOSTA”, o usuário ficará na fase de exploração em RA, detalhada a seguir.

Fase De Exploração Em RA Do Cenário 1: Nesta fase, se permite tanto explorar a figura geométrica da pergunta através de um modelo 3D em RA, quanto ver a resposta certa. Para ver o modelo em RA, deve se apontar para o marcador mostrado na Figura E 5, ficando como mostrado na Figura E 6, os elementos presentes nesta fase são detalhados a seguir e são numerados também na Figura E 6.

Figura E 5: Marcador usado para ver modelo 3D em RA da fase de exploração. (PTC, n.d.)

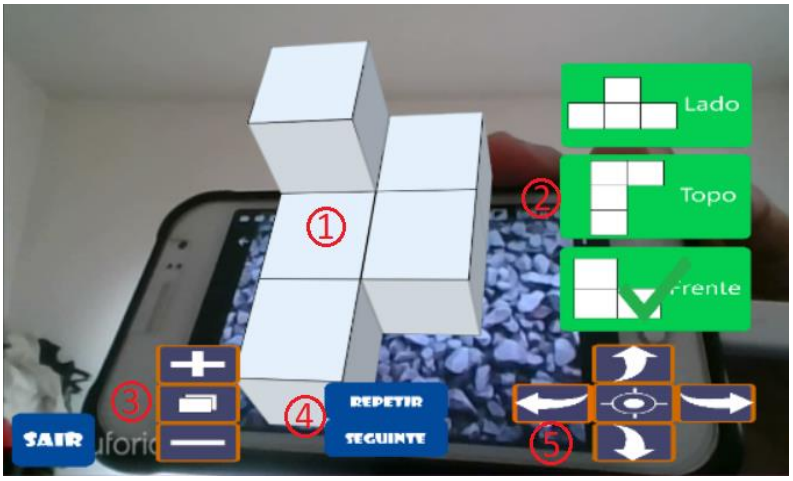

Figura E 6: Fase de exploração do Cenário 1 com a primeira pergunta, os elementos desta fase foram numerados para maior detalhe.

Modelo 3D: Modelo 3D da figura geométrica da pergunta, o qual vai permitir uma maior exploração ao usuário enquanto verifica a resposta certa. Ver Figura E 6 elemento número 1.

Imagens das alternativas: Imagens das alternativas tal como foram apresentadas na fase de perguntas (na Figura E 3 elementos número 3), porém, nesta fase acrescenta-se uma nova imagem com o símbolo “Check” sobreposta em uma das imagens das alternativas para indicar a resposta certa. As imagens alternativas nesta fase são apresentadas na Figura E 6 elementos número 2.

Botões de escalar: Conjunto de botões que permitem escalar o tamanho do modelo 3D e restabelecer ele a seu tamanho original. Estes botões podem ser vistos na Figura E 6 elementos número 3.

Botões de rotacionar: Conjunto de botões que permitem rotacionar o modelo 3D e restabelecer ele a seu rotação original. Estes botões podem ser vistos na Figura E 6 elementos número 5.

Botões de navegação: Botões que permitem repetir a pergunta (botão “REPETIR”), ou ir à próxima (botão “SEGUINTE”). Estes botões podem ser vistos na Figura E 6 elementos número 4.

Uso de Marcador de controle: Nesta fase, alguns eventos podem ser ativados colocando junto do marcador original da Figura E 5, ao marcador adicional apresentado na Figura E 7 (marcador de controle).

Figura E 7: Marcador de controle, usado para ativar eventos sobre o modelo 3D. (ROBERTSON, 2011)

Estes eventos permitem tanto mostrar um texto 3D indicando as faces da visão certa da figura, quanto mudar essas mesmas faces a cor vermelha para melhorar sua visibilidade. O modelo 3D da figura, já tendo colocado o marcador de controle junto ao marcador original, fica como na Figura E 8.

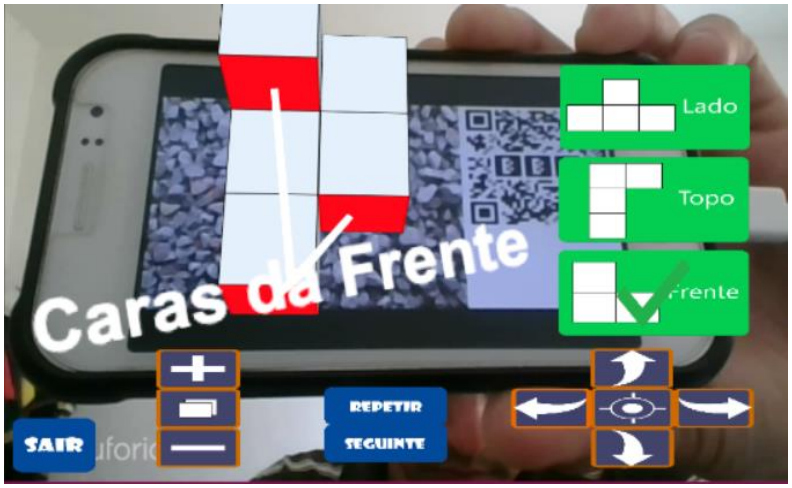

Figura E 8: Fase de exploração do Cenário 1 com a primeira pergunta, e com os eventos ativados através do marcador de controle.

Cenário 2

Se for selecionado o Cenário 2 no menu de cenários, o usuário ficará no menu de fases do Cenário 2, detalhado a seguir.

Menu De Fases Do Cenário 2: Este menu permite escolher a fase do Cenário 2 que o usuário quiser pressionando seu botão respetivo (botão

“TREINAMENTO” para a fase de treinamento, e botão “PERGUNTAS” para a fase de perguntas). Ver Figura E 9.

Figura E 9: Menu de Fases do Cenário 2.

Inicialmente serão apresentadas as instruções tendo selecionado a fase de treinamento, e após isso, as instruções tendo selecionado a fase de perguntas.

Se for selecionada a fase de treinamento no menu de fases, o usuário ficará diretamente na fase de treinamento, que será detalhada a seguir.

Fase De Treinamento do Cenário 2: Na fase de treinamento é apresentado um modelo 3D de um coração em RA, sendo possível mudar seu tamanho e rotação com botões. Para ver o modelo em RA, deve se apontar para o marcador da Figura E 5 (o mesmo usado no Cenário 1), ficando como mostrado na Figura E 10. Nota-se que que em ambos os cenários são usados os mesmos marcadores.

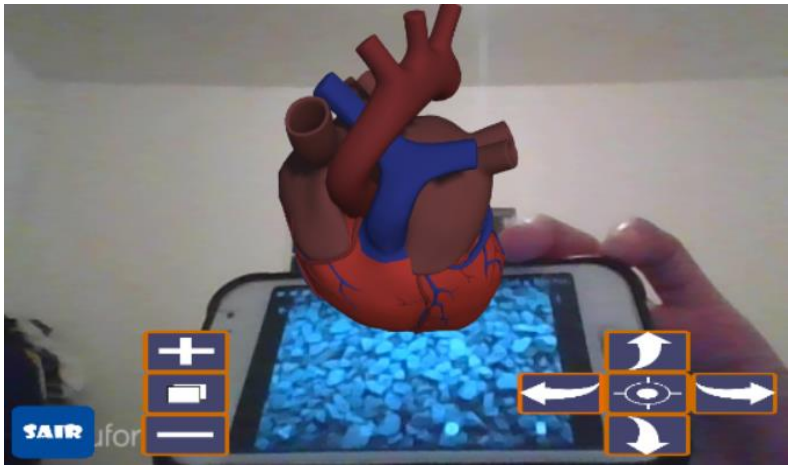

Figura E 10: Fase de Treinamento do cenário 2.

Uso de Marcador de controle: Nesta fase, um evento para mostrar informação adicional poder ser ativado colocando junto do marcador da Figura E 5, ao marcador adicional da Figura E 7 (marcador de controle). Este evento permite mostrar textos 3D indicando os componentes principais do coração, ficando como na Figura E 11.

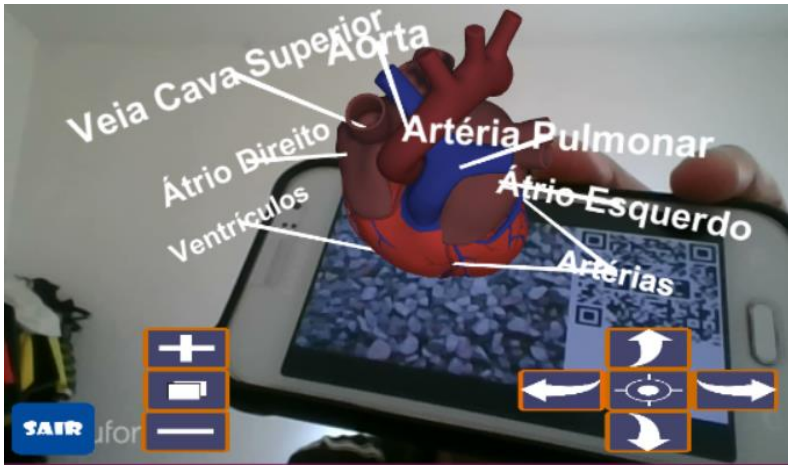

Figura E 11: Fase de Treinamento do Cenário 2, com o evento ativado através do marcador de controle.

Finalmente, se for selecionada a fase de perguntas (ao invés da fase de treinamento) no menu de fases, o usuário ficará no menu de perguntas do Cenário 2, detalhado a seguir.

Menu De Perguntas Do Cenário 2: Este menu permite escolher uma das quatro perguntas disponíveis no Cenário, pressionando seu botão respetivo, ver Figura E 12.

Figura E 12: Menu de perguntas do Cenário 2.

Após pressionar algum dos botões das perguntas, o usuário já ficará na fase de perguntas do Cenário 2, que será detalhada a seguir.

Fase De Perguntas Do Cenário 2: Nesta fase, são apresentadas as perguntas que devem ser respondidas pelo usuário neste cenário, em cada pergunta apresentam-se como alternativas, três modelos 3D em RA (apresentados por turnos) e indicando componentes específicos do coração, e o usuário deverá escolher o modelo com o componente do coração solicitado. O marcador usado será o mesmo da Figura E 5 (o mesmo usado na fase de treinamento, e no Cenário 1), na Figura E 13 se apresenta esta fase com a primeira pergunta, os elementos presentes nesta fase são detalhados a seguir, e são numerados também na Figura E 13.

Texto de pergunta: Consiste no texto que informa qual é a pergunta a realizar, ver Figura E 13 elemento número 1.

Modelo 3D de alternativa: Modelo 3D em RA indicando um componente específico do coração, e representando uma das alternativas da pergunta (alternativa atual), ver Figura E 13 elemento número 2. Lembre-se que nas perguntas deste cenário, são apresentados por turno três modelos 3D como alternativas, os modelos para a primeira pergunta podem ser vistos na Figura E 14.

Botões de escalar: Conjunto de botões que permitem escalar o tamanho dos modelos 3D e restabelecer eles a seu tamanho original. Estes botões podem ser vistos na Figura E 13 elementos número 3.

Botões de rotacionar: Conjunto de botões que permitem rotacionar os modelos 3D e restabelecer eles a seu rotação original. Estes botões podem ser vistos na Figura E 13 elementos número 5.

Botões de explorar alternativas e confirmar alternativa: Botões que permitem ver o próximo modelo alternativa da pergunta (botão “DIREITA”), ver o anterior (botão “ESQUERDA”), e finalmente confirmar ou escolher a alternativa atual (botão “OK”). Estes botões podem ser vistos na Figura E 13 elementos número 4.

Texto do Tempo: Consiste no texto que apresenta o tempo disponível para realizar a pergunta, ver Figura E 13 elemento número 6. O tempo máximo das perguntas deste cenário é de 20 segundos.

Figura E 13: Fase de Perguntas do Cenário 2 com a primeira pergunta, os elementos desta fase foram numerados para maior detalhe.

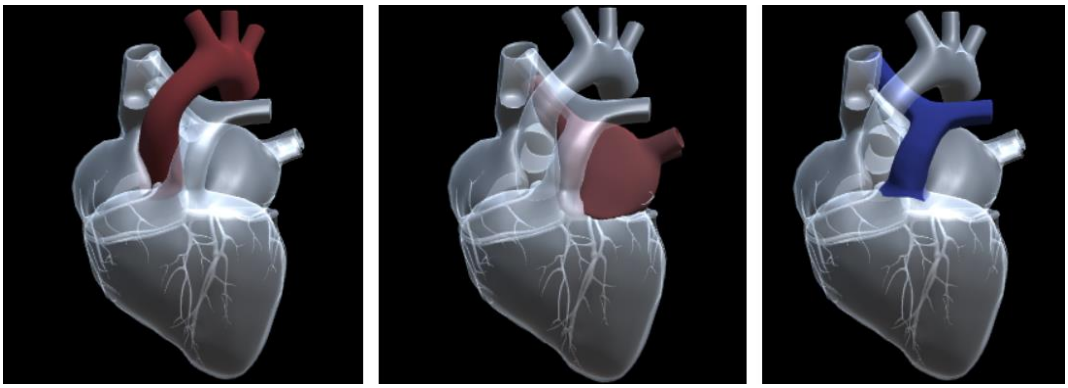

Figura E 14: Modelos 3D indicando componentes do coração, e usados como alternativas para uma pergunta. O primeiro modelo indica a Aorta (figura esquerda), o segundo o Átrio Esquerdo (figura centro), e o último a Artéria Pulmonar (figura direita).

Após responder a pergunta (pressionando o botão de confirmar “OK”), uma imagem de *feedback* de resposta, será mostrada para o usuário informando-lhe se a sua resposta está correta ou errada, e ficando como na Figura E 15. Nota-se que finalizar o tempo também será considerado pelo aplicativo como resposta errada.

Figura E 15: Imagens de *feedback* de resposta correta (figura esquerda), e resposta errada (figura direita).

Vários botões de navegação ficam também disponíveis juntos com as imagens, eles são: botão “VER RESPOSTA”, que permite ver a resposta correta da pergunta, botão “REPETIR” que permite repetir a pergunta, e o botão “SEGUINTE”, para ir à próxima pergunta diretamente (sem precisar voltar ao menu de perguntas para selecionar a próxima).

Após pressionar o botão “VER RESPOSTA”, a fase ficará como na Figura E 16, unicamente com o modelo 3D da alternativa certa.

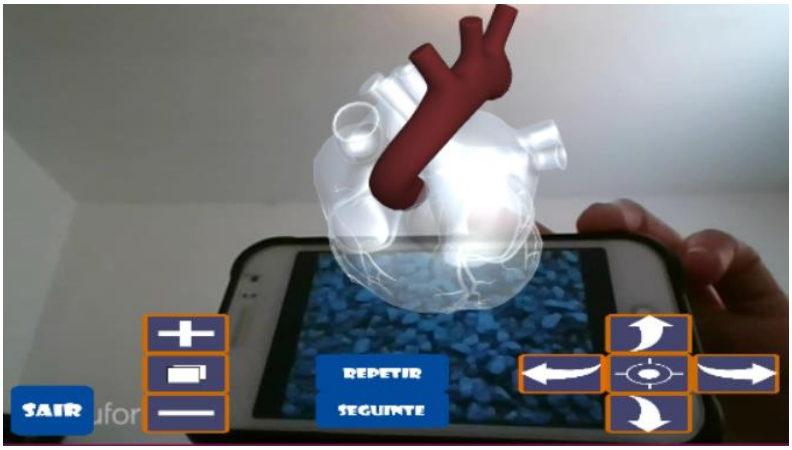

Figura E 16: Fase de Perguntas do Cenário 2 com a primeira pergunta, e enquanto está sendo apresentada a resposta certa.

Uso de Marcador de controle: Nesta fase, enquanto está sendo apresentada a resposta certa, vários eventos podem ser ativados colocando junto do marcador principal (da Figura E 5), ao marcador de controle (da Figura E 7). Estes eventos permitem tanto mostrar um texto 3D indicando o componente do coração em questão, quanto mudar a cor do modelo para fazer ele ficar sem nenhuma

transparência (ficando assim, o modelo todo com cor), como mostrado na Figura E 17.

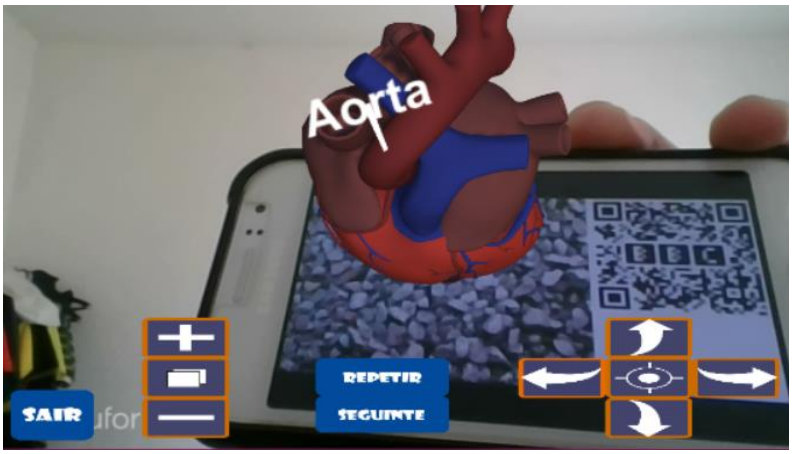

Figura E 17: Fase de Perguntas do Cenário 2 com a primeira pergunta, enquanto está sendo apresentada a resposta certa, e com os eventos ativados através do marcador de controle.

Apêndice F

INSTRUÇÕES PARA CRIAÇÃO DE PERGUNTAS DO APLICATIVO DO ESTUDO DE CASO

São apresentadas a seguir as instruções para a criação das primeiras perguntas dos dois cenários do aplicativo realizado no estudo de caso. Nota-se que as perguntas de cada cenário são semelhantes umas às outras, portanto uma vez que for criada a primeira pergunta de um cenário, essa pergunta pode ser usada como base para criar as outras perguntas facilmente.

Instruções Criação Primeira Pergunta Cenário 1 Aplicativo

A seguir serão apresentados os passos necessários para criar a primeira pergunta do cenário 1 do aplicativo, fazendo uso do AR Educacional Framework.

1) Fazer as configurações necessárias para instalar o AR Educacional Framework em um projeto do Unity. Nota-se que para este caso deve-se importar o pacote de marcadores necessários chamado imagem1marker3.unitypackage, e copiar a sua licença respetiva. O pacote imagem1marker3.unitypackage, sua licença, e as imagens dos marcadores estão disponíveis na pasta chamada: Marcadores Estudo de Caso, no link:

https://www.dropbox.com/sh/n6mecqur8hi7qtf/AADtwUUWhyfDarNbOit9_JESa?dl=0[22]

2) Importar o pacote MODELS no projeto, o qual possui o modelo necessário: MODEL2, no caminho (**`MyModels/Models/model2`**). O pacote MODELS está disponível no link: https://www.dropbox.com/s/cv54s0s2l9p7aml/MODELS.unitypackage?dl=0 [23]

   Importar também a pasta Textures com as imagens necessárias para a pergunta, disponível no link: https://www.dropbox.com/sh/733zghqed1x12re/AAB7sh6zld1ZGSLsiQiJw7uSa?dl=0[24]

3) Criar uma pergunta com a classe **`QuestionController1`** e sua inicialização padrão, como explicado no "Tutorial Classe **`QuestionController1`**", em: "Inicialização Default Da Classe **`QuestionController1`**", ver Apêndice D. Além disso, neste caso também será removido o objeto Cube padrão, e desmarcado o campo UseMarker para desabilitar as opções orientadas à utilização de um marcador na pergunta (lembrando que a fase de perguntas deste cenário não precisa marcadores, e portanto essas opções não são de muito interesse). A cena deveria ficar como na Figura F 1.

[22] Todas as referências à pasta "Marcadores Estudo de Caso" neste documento, se referem a este link.

[23] Todas as referências ao pacote "MODELS" neste documento, se referem a este link.

[24] Todas as referências à pasta "Textures" neste documento, se referem a este link.

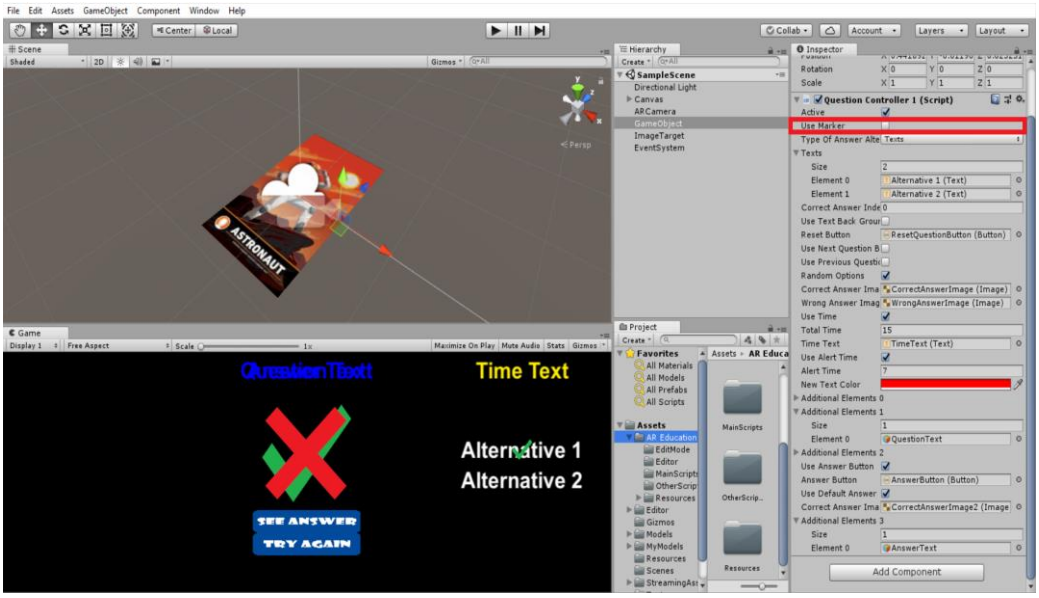

Figura F 1: Pergunta criada com objeto Cube removido, e campo UseMarker desmarcado.

4) Utilizar três imagens como alternativas ao invés dos textos padrão, para isso é necessário mudar o tipo de alternativas da pergunta no campo *TypeOfAnswerAlternatives* para usar “Images”, aumentar a 3 o valor Size de Images, e finalmente criar e associar na pergunta as imagens necessárias usando as texturas importadas (P1alternativa1, P1alternativa2 e P1alternativa3), através da pasta Textures. Nota-se que os textos padrão tem sido removidos nesse processo. A cena deveria ficar como na Figura F 2.

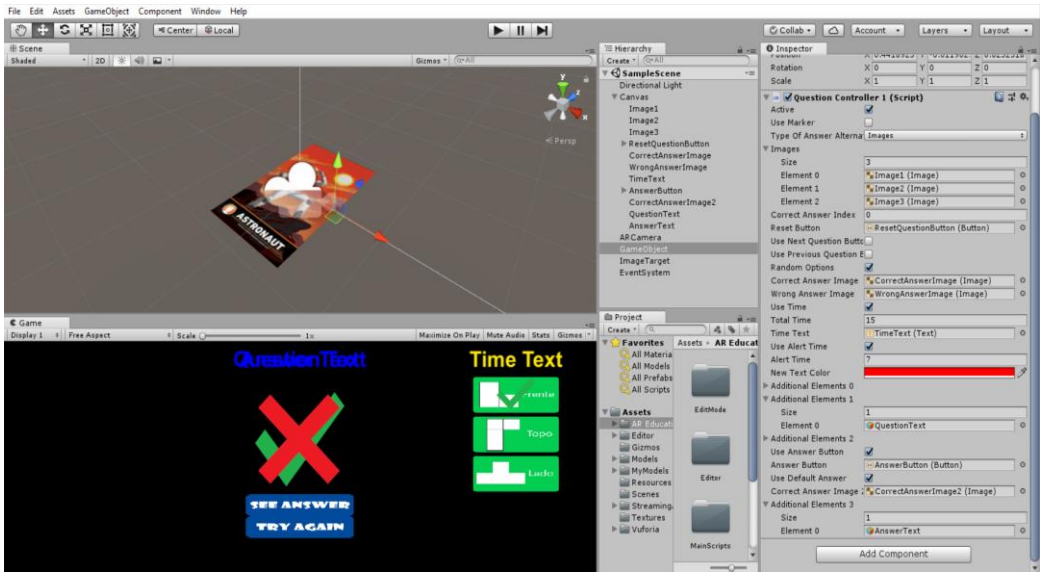

Figura F 2: Novas imagens usadas como alternativas da pergunta.

5) Criar e utilizar para a pergunta, a imagem da pergunta (que aparecerá quando está sendo respondida a pergunta), e a imagem de fundo (que aparecerá após responder ela). Para isso duas imagens são criadas em um novo **`Canvas`** (configurado com seu campo *Sort Order* em -1) usando as texturas importadas (backgroundimage1 e backgroundimage0) da pasta Textures, e finalmente essas imagens são arrastadas em novos campos criados em AdditionalElements1 e AdditionalElements2 respetivamente. A cena deveria ficar como na Figura F 3. Nota: A imagem de fundo foi desativa para deixar visível só a imagem de pergunta.

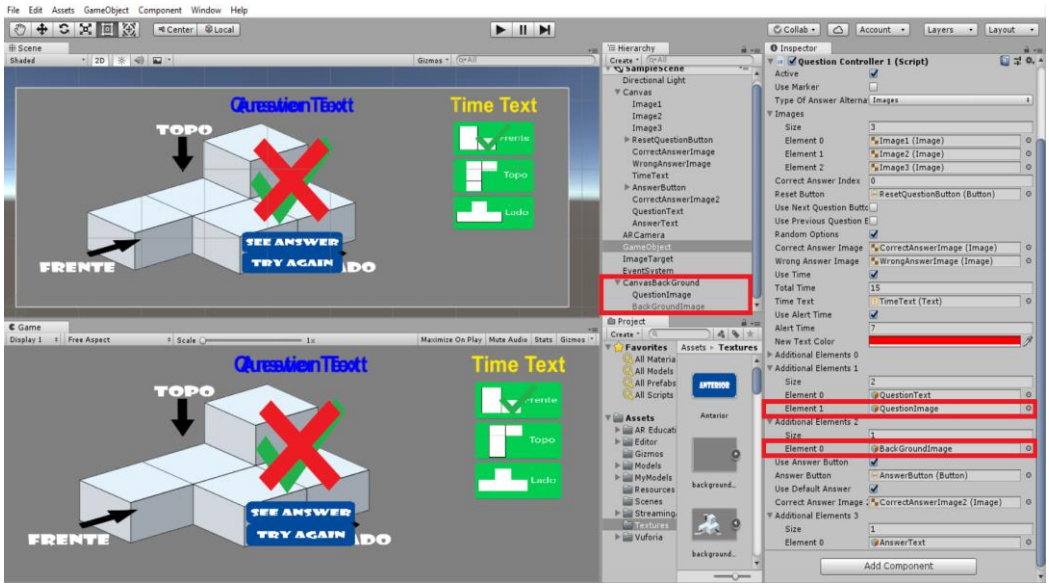

Figura F 3: Novas imagens QuestionImage e BackGroundImage (em um novo `Canvas`), já arrastadas nos campos criados AdditionalElements1 e AdditionalElements2 respetivamente.

6) Renomear o texto da pergunta padrão (QuestionText) para o texto real usado na pergunta, que será o seguinte: Qual alternativa é uma vista certa?, além disso, remover e deixar de usar na pergunta o texto da resposta padrão (AnswerText), que não será necessário.
7) Colocar como filho do marcador, ao modelo MODEL2 importado (que será usado para a fase de exploração após responder a pergunta), e arrastar ele no campo AdditionalElements3, para usá-lo como elemento adicional. Nota-se que isso permitirá ao modelo ser ativado unicamente quando está sendo apresentada a resposta certa, após ser respondida a pergunta. A Cena deveria ficar como na Figura F 4.

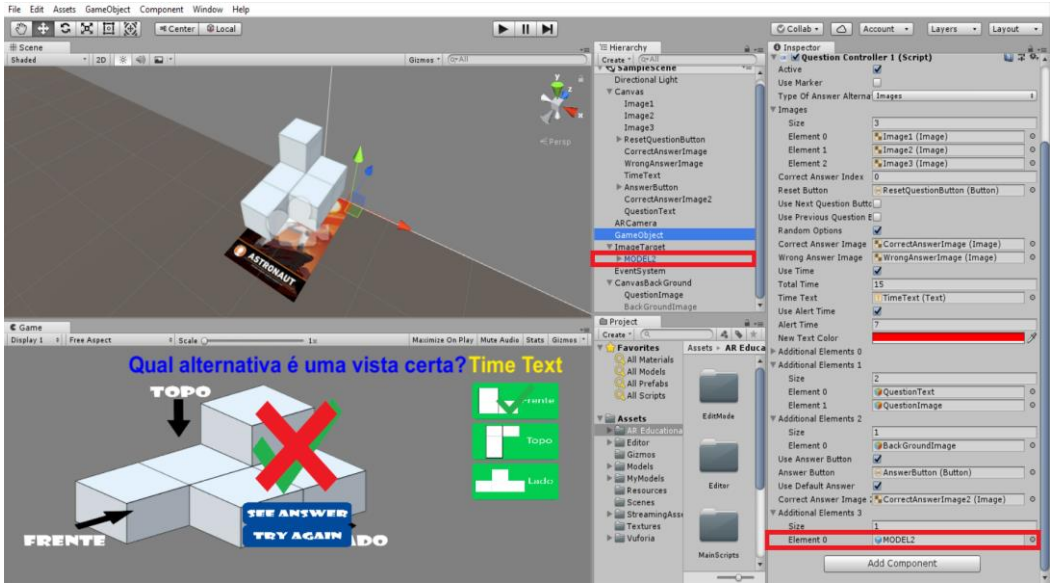

Figura F 4: Modelo MODEL2 importado, já colocado como filho do marcador e arrastado no campo AdditionalElements3.

8) Atualizar o marcador *default*, para usar o marcador chamado unnamed disponível no banco do dados dos novos marcadores importados, como explicado em no passo “Configurar Marcadores Do Framework Na Cena” da fase “Configuração Do Framework”, em Apêndice A. Adicionalmente, a largura do marcador foi colocada em 1.5 no seu campo Width. A cena deveria ficar como na Figura F 5.

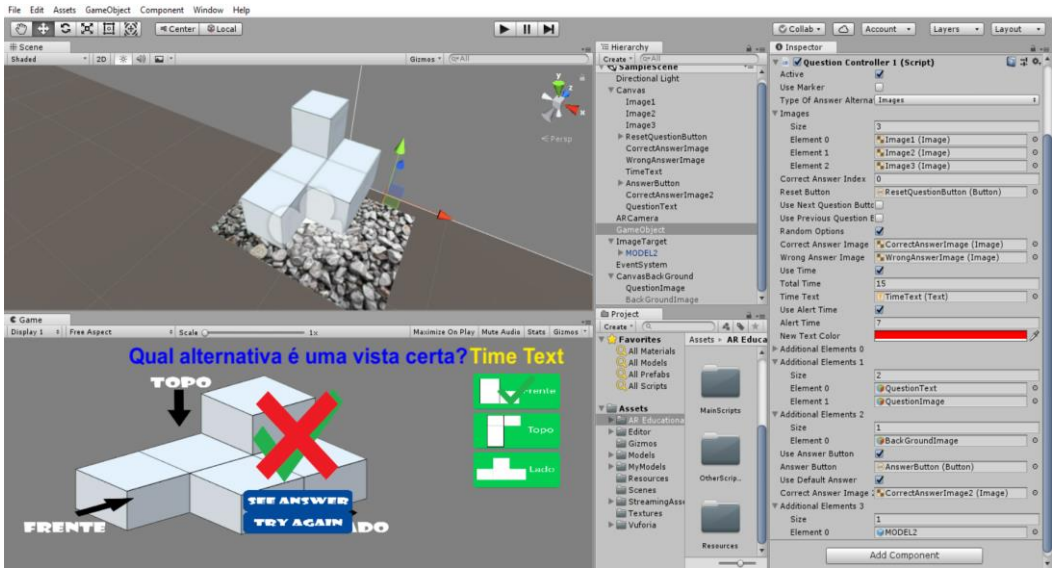

Figura F 5: Marcador atualizado para usar novo importado.

9) Criar um pacote de botões para rotacionar e escalar com a classe **`ButtonsPack`** e sua inicialização padrão, como explicado no "Tutorial Classe **`ButtonsPack`**" em: "Inicialização Default Da Classe **`ButtonsPack`**", ver Apêndice D. Lembre-se que o objeto vazio com a classe **`ButtonsPack`** deveria ser removido, já o objeto vazio com **`QuestionController1`** criado anteriormente, sugere-se renomeá-lo como Question1. A cena deveria ficar como na Figura F 6.

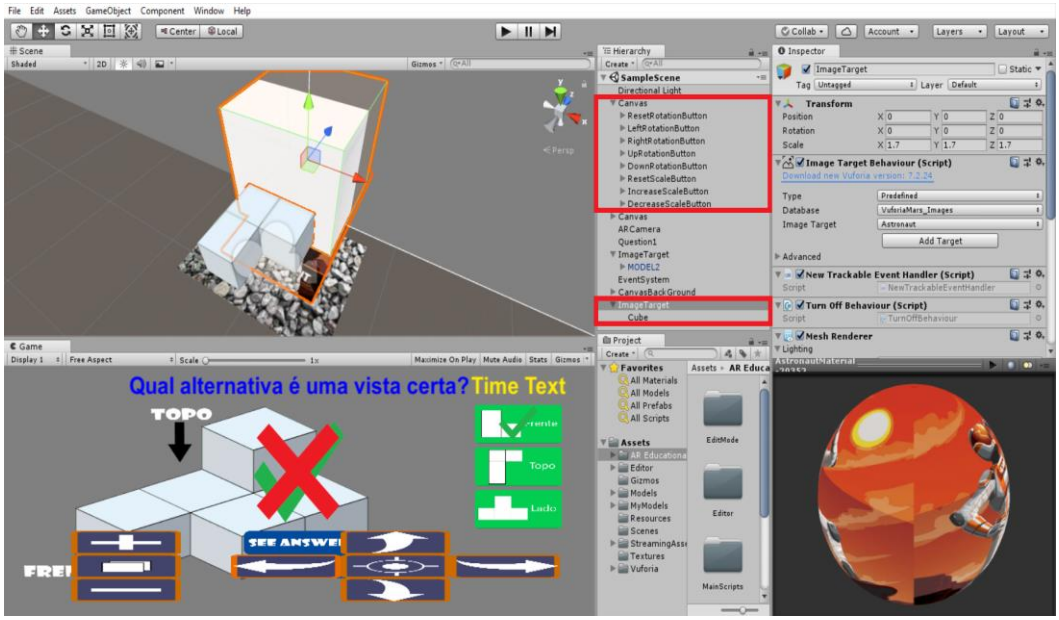

Figura F 6: Pacote de botões, objeto cubo e marcador criados com `ButtonsPack`. Objeto vazio com `ButtonsPack` removido e objeto vazio com `QuestionController1` renomeado para Question1.

10) Fazer os botões de rotacionar e escalar, ficar associados com o modelo MODEL2 para permitir rotacionar e escalar ele (ao invés do elemento padrão Cube), para isso, em cada um dos botões deve-se atualizar os campos da sua classe **`ButtonOperations`** para usar MODEL2 e seu marcador, ao invés dos elementos padrão, de jeito similar como foi explicado no "Tutorial Uso Classe **`ButtonsPack`** Em Perguntas", em: "Atualizar Botões Para Ficar Associados Com Elementos Da Pergunta", ver Apêndice D. Após isso, lembre-se que os elementos desnecessários deverão ser removidos (objeto padrão Cube e seu marcador), e a cena deveria ficar como na Figura F 7.

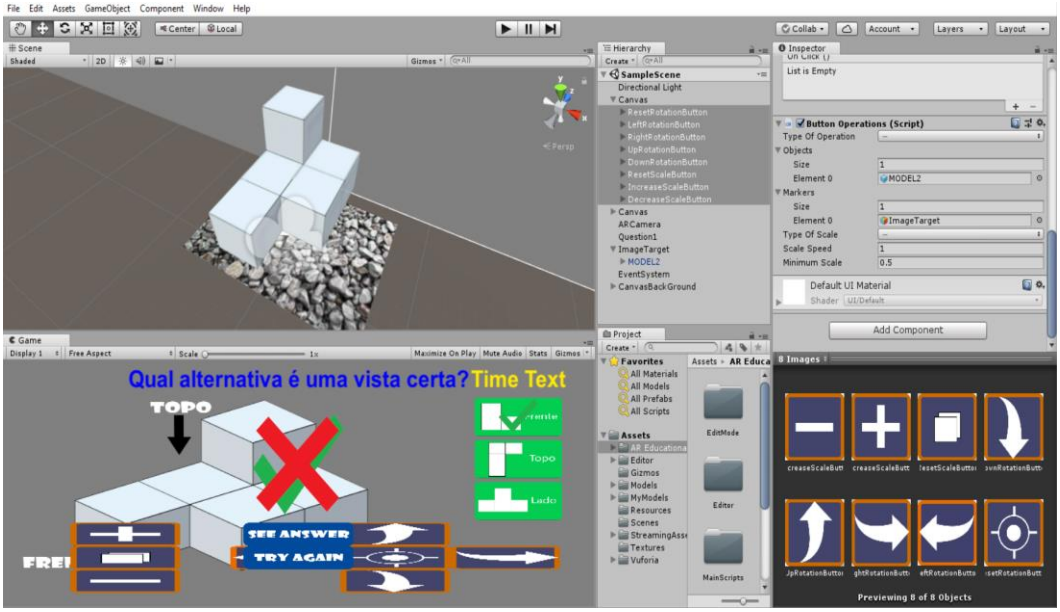

Figura F 7: Classe `ButtonOperations` dos botões atualizada para usar MODEL2.

11) Os botões para rotacionar e escalar são diminuídos de tamanho e reposicionados para não se sobrepor com os outros botões existentes.
12) Mudar as imagens *default* dos botões que têm textos, para usar imagens com textos em português (usar novas texturas chamadas TryAgain e SeeAnswer da pasta Textures importada). A cena deveria ficar como na Figura F 8.

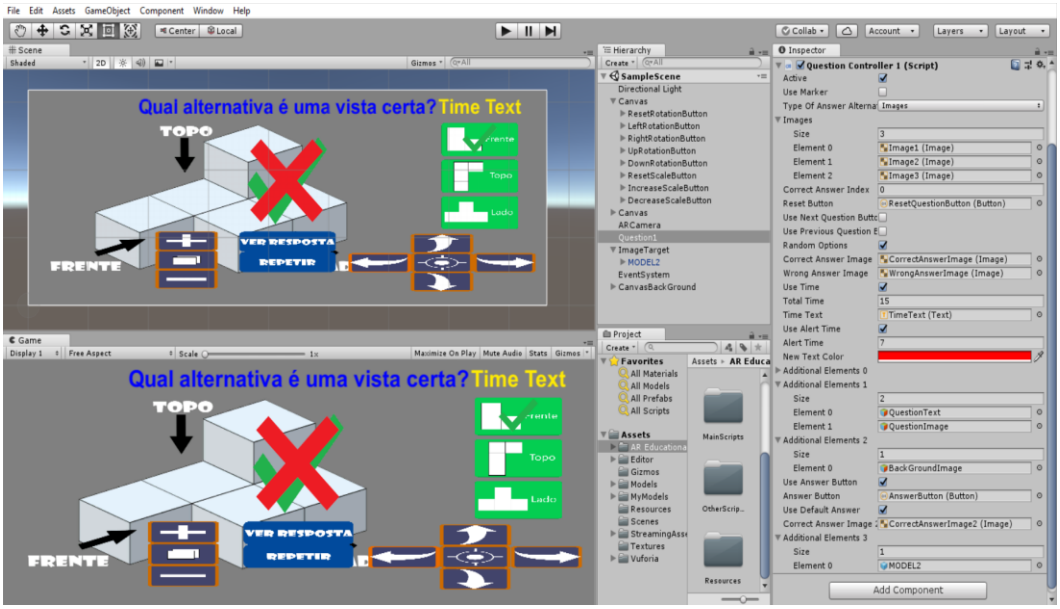

Figura F 8: Botões atualizados com textos em português.

13) Fazer os botões de escalar e rotacionar, ficar visíveis unicamente quando está sendo apresentada a resposta correta da pergunta (fase de exploração). Isso, será feito, como explicado no “Tutorial Uso Classe **`ButtonsPack`** Em Perguntas”, em: “Vincular Botões A Estados Da Pergunta” (ver Apêndice D), notando-se que para este caso os botões serão definidos no estado 3 (e portanto devem ser usados campos AdditionalElements3). A cena deveria ficar como na Figura F 9.

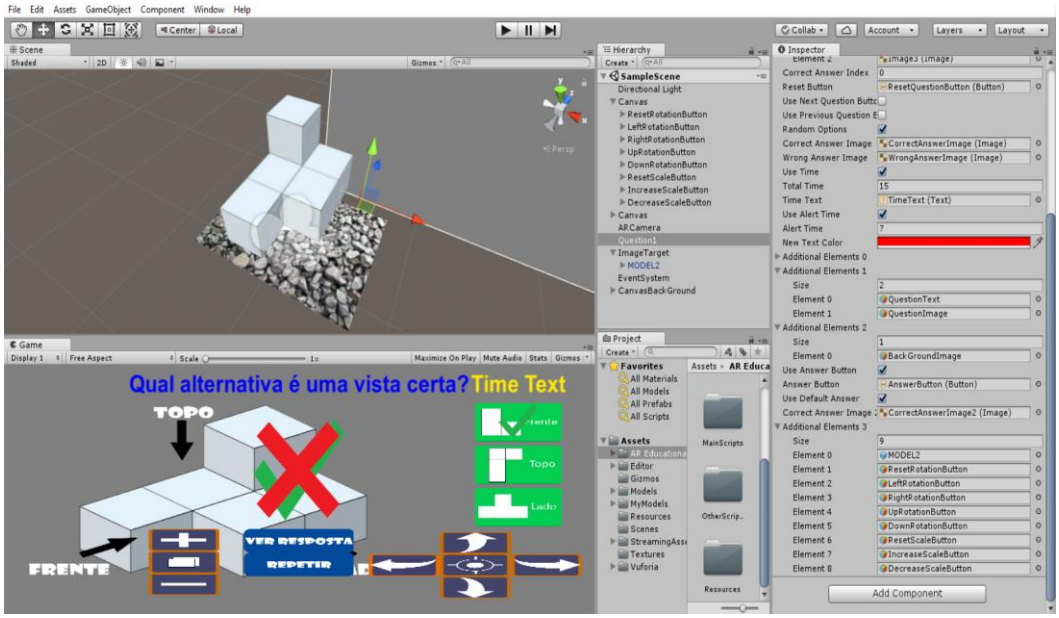

Figura F 9: Botões visíveis só quando está sendo apresentada a resposta correta da pergunta.

14) Criar um evento com a classe **`ActivateInformation`** e sua inicialização padrão, como explicado no “Tutorial Classe **`ActivateInformation`**” em: “Inicialização Default Da Classe **`ActivateInformation`**”, ver Apêndice D. Sugere-se renomear o novo objeto vazio container para ActivateInformation, a cena deveria ficar como na Figura F 10. Nota: O CanvasBackGround com as imagens de pergunta e fundo, foi desativado temporariamente para algumas capturas de tela visando melhorar a visualização dos novos elementos criados. Lembrar ativar ele sempre que for a executar a cena.

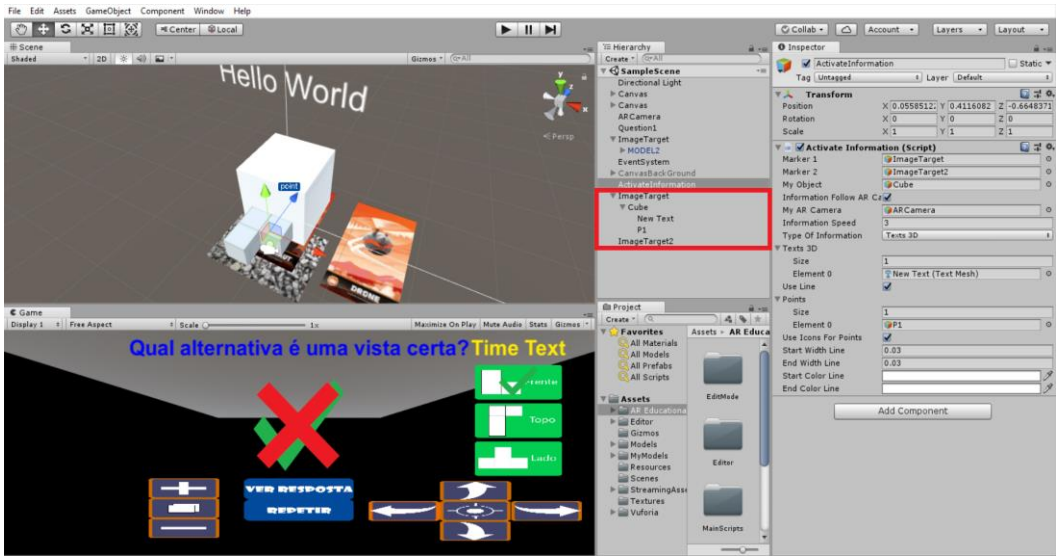

Figura F 10: Objeto cubo, texto 3D e marcadores criados com `ActivateInformation`. O objeto vazio criado foi renomeado para ActivateInformation.

15) Atualizar os campos da Classe **`ActivateInformation`** para usar o modelo MODEL2 com seu marcador, ao invés dos elementos padrão, de jeito similar como foi explicado no "Tutorial Uso Classe **`ActivateInformation`** Em Perguntas", em: "Atualizar Classe **`ActivateInformation`** Para Ficar Associada Com Elementos Da Pergunta", ver Apêndice D. Após isso, lembre-se que os elementos desnecessários deverão ser removidos, e a cena deveria ficar como na Figura F 11. Nota-se também, que o texto 3D NewText, e o elemento P1 têm sido reposicionados.

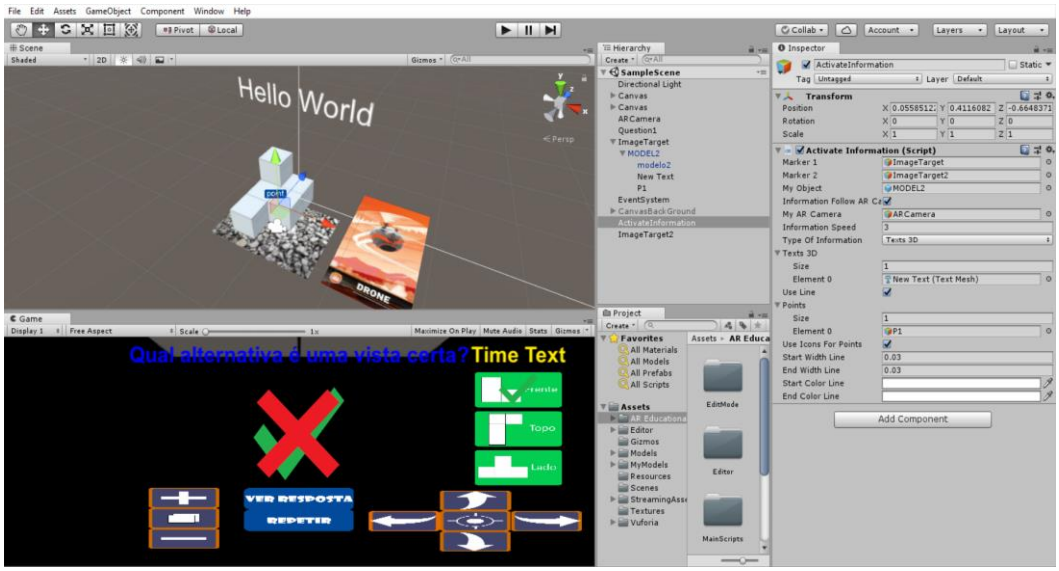

Figura F 11: Classe `ActivateInformation` atualizada para usar modelo MODEL2 e seu marcador.

Adicionalmente, atualizar o marcador *default* **`imageTarget2`**, para usar o novo marcador (chamado marker2) disponível no banco do dados dos novos marcadores importados, tal como foi feito no passo 8. A cena deveria ficar como na Figura F 12.

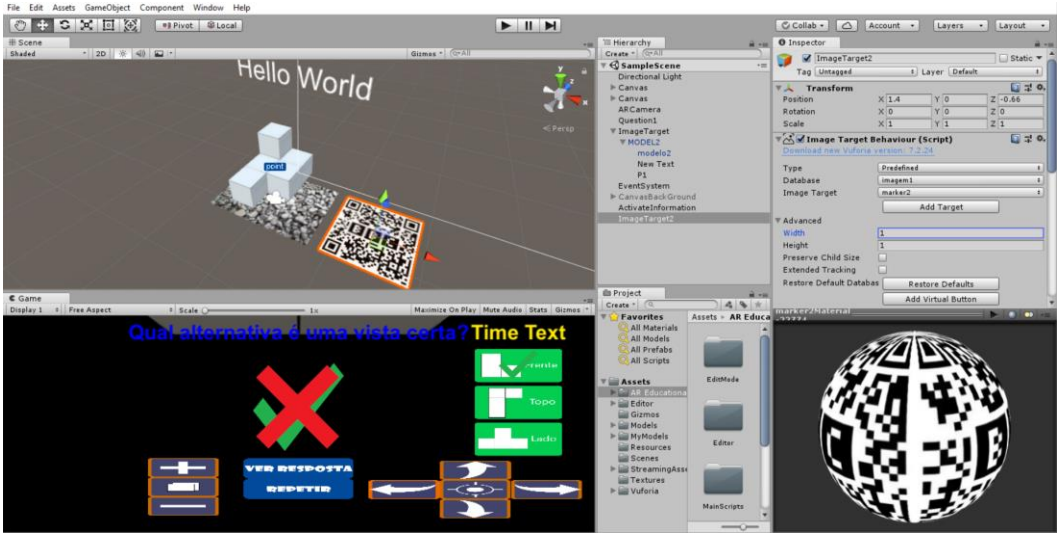

Figura F 12: Marcador `imageTarget2` atualizado para usar novo importado.

16) Posicionar o ponto P1 (que indica a origem da linha do texto 3D), para ficar numa das faces da frente do modelo, como mostrado na Figura F 13. Além disso, posicionar o texto 3D newText para ficar como na mesma figura, e renomear ele para: “Caras da Frente”.

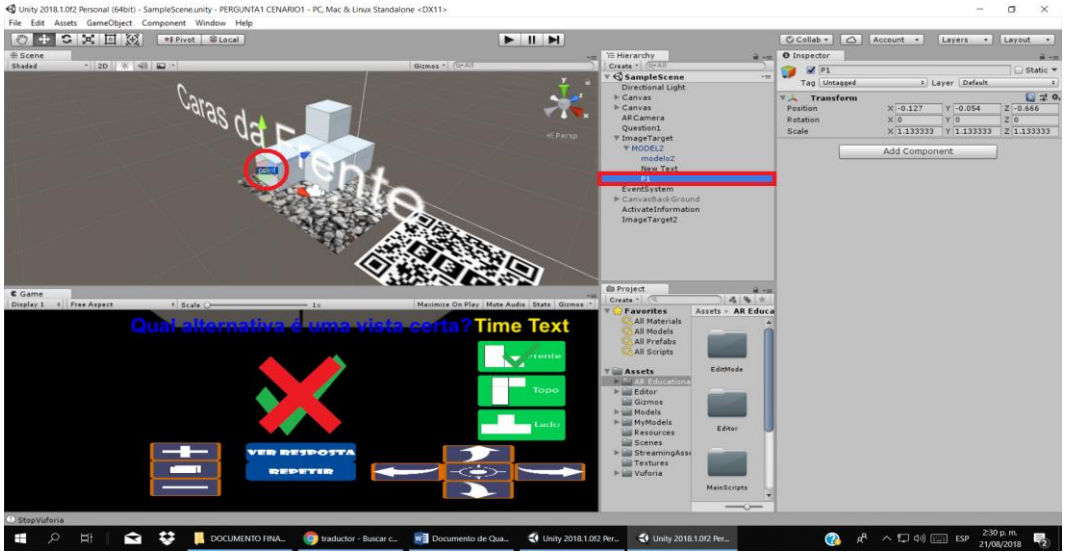

Figura F 13: Ponto P1, posicionado numa das faces da frente do modelo. Texto 3D renomeado para Caras da Frente.

17) Uma vez que neste ponto existem dois marcadores simultâneos na cena, este passo consiste em fazer a configuração sugerida com vários marcadores, falada no “Tutorial Configuração AR Educational Framework”, em “Configuração Adicional Sugerida Usando Vários Marcadores”, ver Apêndice A.
18) Criar dois novos textos 3D (posicionando eles para ficar na mesma posição e com o mesmo nome que o texto 3D anterior), e seus dois pontos respectivos, e utilizar eles para indicar as outras faces da frente do modelo, ficando como na Figura F 14.

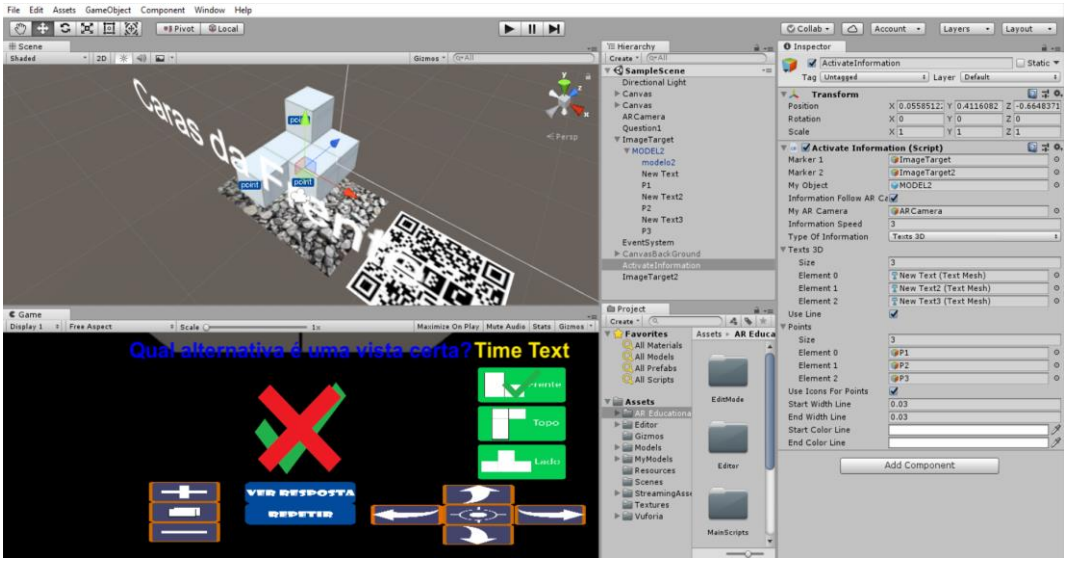

Figura F 14: Novos textos 3D (newText2 e newText3) e seus pontos respectivos (P2 e P3) criados e usados para indicar as outras faces da frente do modelo.

19) Criar um evento com a classe **`ChangeElement`** e sua inicialização padrão, como explicado no “Tutorial Classe **`ChangeElement`**”, em: “Inicialização Default Da Classe **`ChangeElement`**”, ver Apêndice D. Sugere-se renomear o novo objeto vazio container para ChangeElement, a cena deveria ficar como na Figura F 15.

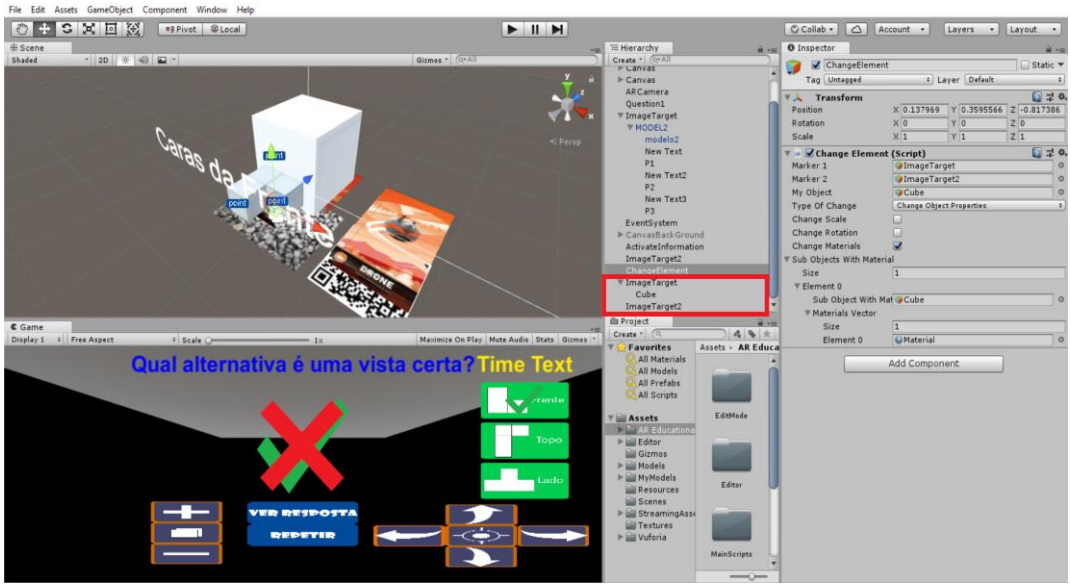

Figura F 15: Novo objeto Cube e marcadores criados com `ChangeElement`. O novo objeto vazio container foi renomeado para ChangeElement.

20) Atualizar os campos da classe **`ChangeElement`** para usar o modelo MODEL2 com os marcadores originais, ao invés dos elementos padrão, e neste caso para permitir trocar o material das faces da frente do modelo (o segundo Material disponível no seu filho model2), por um novo material com cor vermelho, como explicado em Apêndice D, no “Tutorial Uso Classe **`ChangeElement`** Em Perguntas”, em: “Atualizar Classe **`ChangeElement`** Para Ficar Associada Com Elementos Da Pergunta”, (para o processo de atualizar os campos), e no “Tutorial Classe **`ChangeElement`**”, em: “Configurar Classe **`ChangeElement`** Para Usar Modelos 3D Próprios”, (para o processo de definir os materiais a serem trocados). Após isso, lembre-se que os elementos desnecessários deverão ser removidos, e a cena deveria ficar como na Figura F 16.

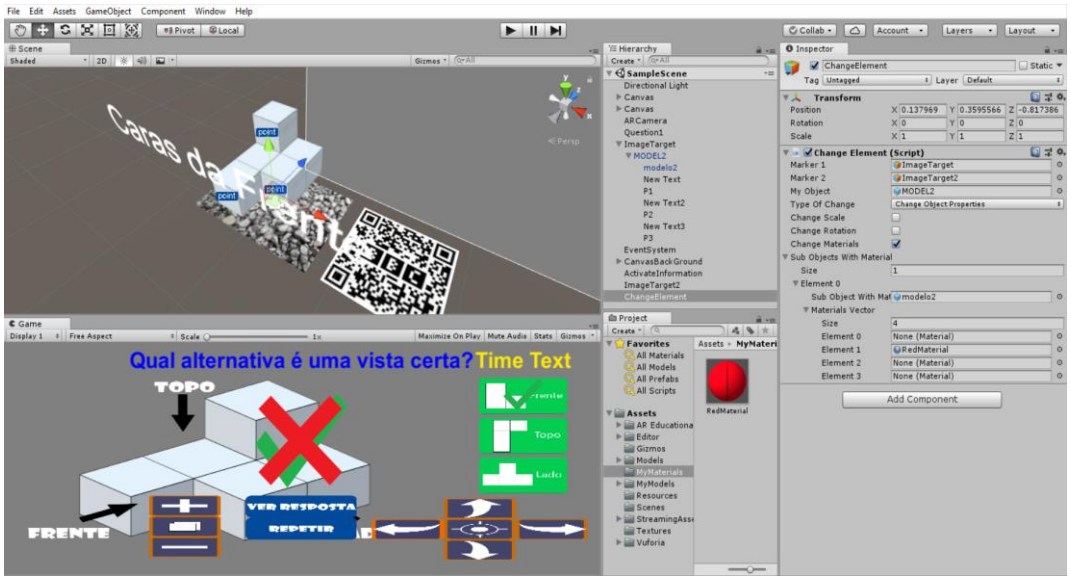

Figura F 16: Classe `ChangeElement` atualizada para usar o modelo MODEL2 e os marcadores originais.

Nota: O Material vermelho utilizado, foi um novo material criado (nomeado RedMaterial), configurado como ***`Legacy Shaders/Diffuse`*** na opção **`Shader`**, e que usa a textura "rojo" disponível no caminho: **`MyModels/Models/modelstextures`** (do pacote MODELS dos modelos importado). Lembrar também ativar o CanvasBackGround que tinha sido desativado para as capturas de tela.

21) Como passo final, o tempo se configura a 25 segundos em TotalTime, ficando a cena pronta para ser executada.

Instruções Criação Primeira Pergunta Cenário 2 Aplicativo

A seguir serão apresentados os passos necessários para criar a primeira pergunta do cenário 2 do aplicativo, fazendo uso do AR Educacional Framework.

1) Fazer as configurações necessárias para instalar o AR Educacional Framework em um projeto do Unity. Nota-se que para este caso deve-se importar o pacote de marcadores necessários chamado imagem1marker3.unitypackage, e copiar a sua licença respetiva, os quais estão disponíveis na pasta: Marcadores Estudo de Caso.

2) Importar o pacote Heart de modelos no projeto, o qual possui os modelos necessários: Aorta, Átrio Direito, e Artéria Pulmonar, no caminho **`MyModels/Heart`**.
3) Criar uma pergunta com a classe **`QuestionController2`** e sua inicialização padrão, e atualizar ela para usar os modelos 3D: Aorta, e Átrio Direito, como explicado no "Tutorial Classe **`QuestionController2`**", em: "Inicialização Default Da Classe **`QuestionController2`**" e "Configurar Classe **`QuestionController2`** Para Usar Modelos 3D Próprios", ver Apêndice D. Além disso, utilizar também, como uma terceira alternativa, o modelo 3D chamado Artéria Pulmonar. A cena deveria ficar como na Figura F 17.

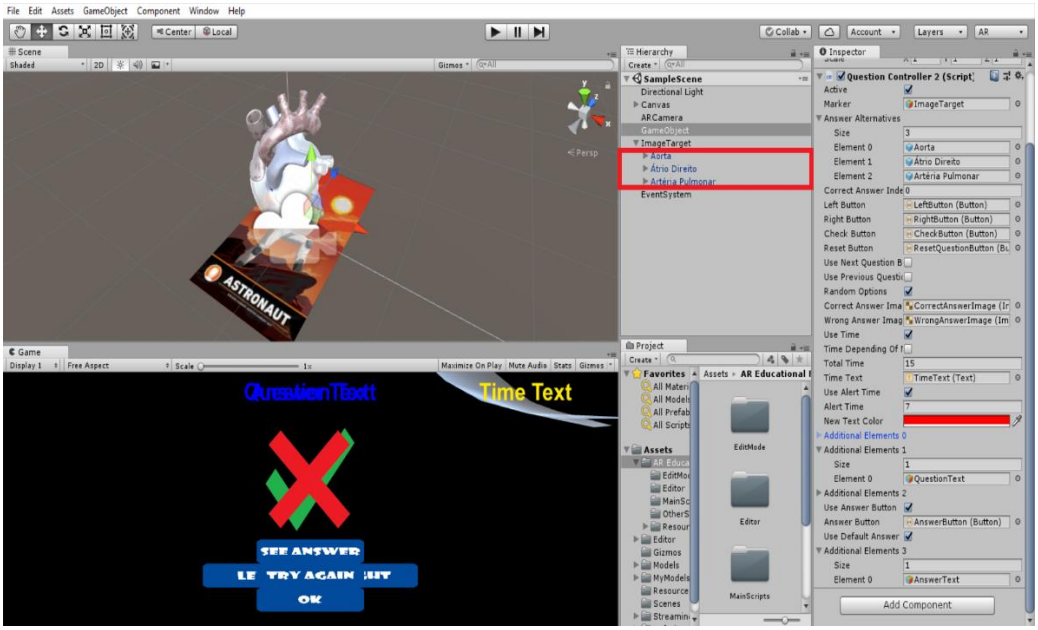

Figura F 17: Pergunta criada, e utilizando os modelos 3D: Aorta, Átrio Direito, e Artéria Pulmonar.

4) Atualizar o marcador *default*, para usar o marcador chamado unnamed disponível no banco do dados dos novos marcadores importados, como explicado no passo "Configurar Marcadores Do Framework Na Cena" da fase "Configuração do Framework", ver Apêndice A. Adicionalmente, a largura do marcador foi colocada em 1 no seu campo Width. A cena deveria ficar como na Figura F 18.

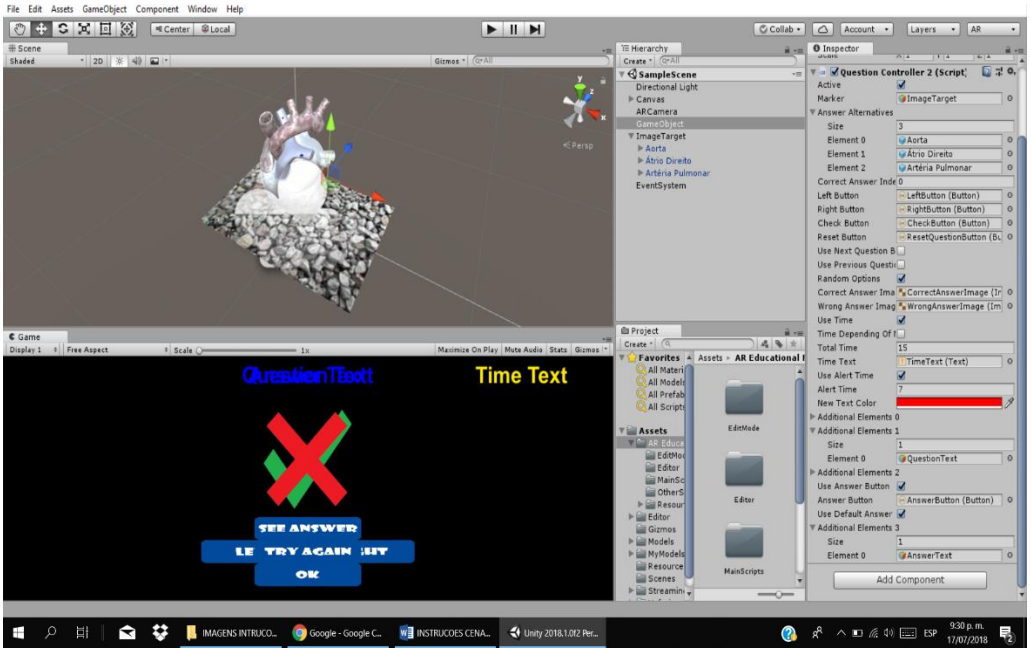

Figura F 18: Marcador atualizado para usar novo importado.

5) Criar um pacote de botões para rotacionar e escalar com a classe **`ButtonsPack`** e sua inicialização padrão, como explicado no "Tutorial Classe **`ButtonsPack`**" em: "Inicialização Default Da Classe **`ButtonsPack`**", ver Apêndice D. Lembre-se que o objeto vazio com a classe **`ButtonsPack`** deveria ser removido, já o objeto vazio com **`QuestionController2`** criado anteriormente, sugere-se renomeá-lo como Question1. A cena deveria ficar como na Figura F 19.

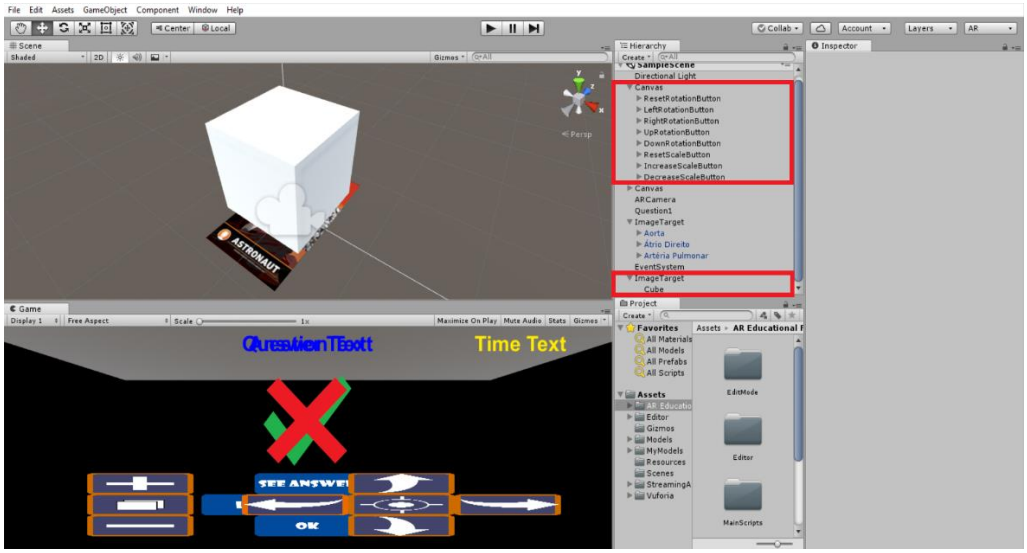

Figura F 19: Pacote de botões, objeto cubo e marcador criados com `ButtonsPack`. Objeto vazio com `ButtonsPack` removido e objeto vazio com `QuestionController2` renomeado para Question1.

6) Fazer os botões de rotacionar e escalar, ficar associados com os elementos da pergunta (modelos Aorta, Átrio Direito, e Artéria Pulmonar) para permitir rotacionar e escalar eles, para isso, atualizar os campos da sua classe **`ButtonOperations`** para usar os elementos da pergunta ao invés dos seus próprios elementos padrão, como foi explicado no "Tutorial Uso Classe **`ButtonsPack`** Em Perguntas", em: "Atualizar Botões Para Ficar Associados Com Elementos Da Pergunta", ver Apêndice D. Após isso, lembre-se que os elementos desnecessários deverão ser removidos (objeto padrão Cube e seu marcador), e a cena deveria ficar como na Figura F 20.

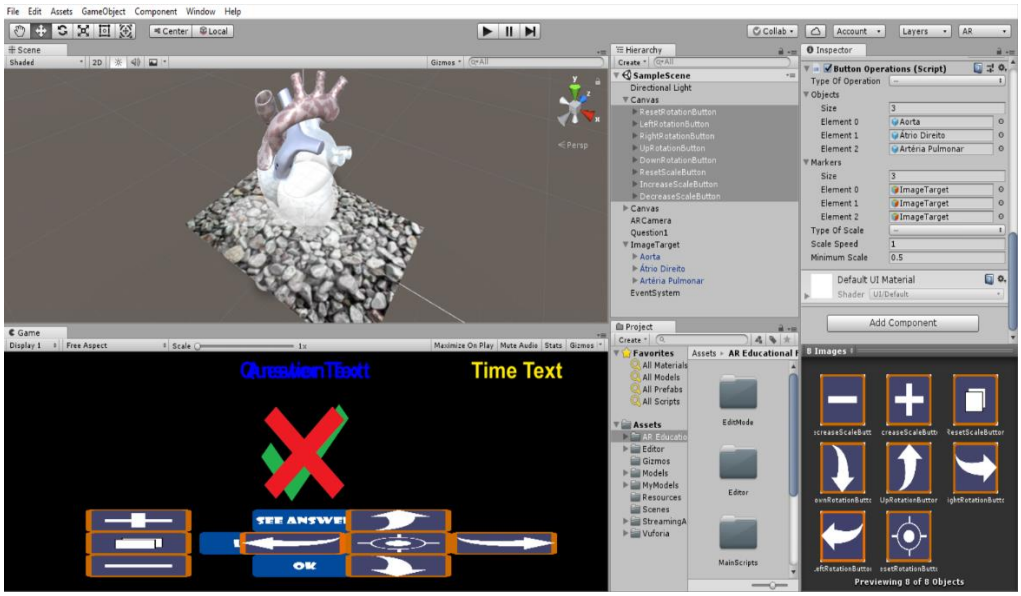

Figura F 20: Classe `ButtonOperations` dos botões atualizada para usar os elementos da pergunta.

7) Os botões para rotacionar e escalar são diminuídos de tamanho e reposicionados para não se sobrepor com os outros botões existentes, ficando a cena como na Figura F 21.

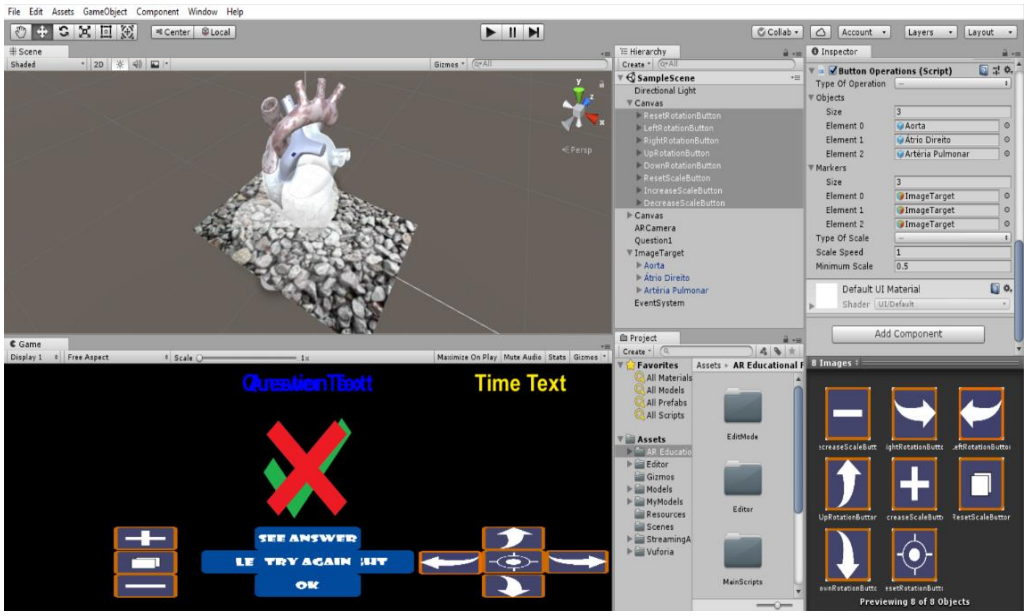

Figura F 21: Botões de rotacionar e escalar diminuídos de tamanho e reposicionados.

8) Renomear o texto da pergunta padrão (QuestionText) para o texto real usado na pergunta, que será o seguinte: Em qual modelo se indica a Aorta?, além disso, remover e deixar de usar na pergunta o texto da resposta padrão (AnswerText), que não será necessário.
9) Importar a pasta MyTextures, disponível no link: https://www.dropbox.com/sh/gn3uort0t6n7r3f/AACDAo8mGBwzyWO4XnibDEdya?dl=0

   E após isso, mudar as imagens *default* dos botões que têm textos, para usar imagens com textos em português (usar novas texturas chamadas Repetir, VerResposta, Esquerda e Direita da pasta MyTextures).
10) Fazer os botões de escalar e rotacionar, ficar visíveis unicamente quando está sendo respondida a pergunta e quando está sendo apresentada a resposta correta da pergunta. Isso, será feito, como explicado no “Tutorial Uso Classe **`ButtonsPack`** Em Perguntas”, em: “Vincular Botões A Estados Da Pergunta”, (ver Apêndice D), notando-se que para este caso os botões serão definidos nos estados 1 e 3 (e portanto devem ser usados os campos AdditionalElements1 e AdditionalElements3). A cena deveria ficar como na Figura F 22.

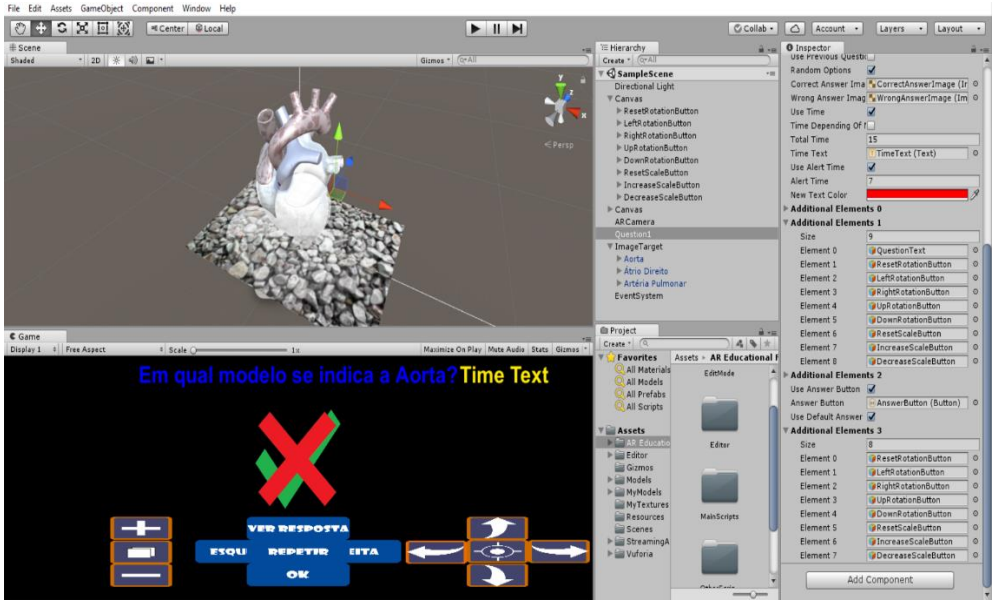

Figura F 22: Botões visíveis só quando está sendo respondida a pergunta e quando está sendo apresentada a resposta correta da pergunta.

11) Desativar o campo UseDefaultAnswer da classe **`QuestionController2`** para não usar o jeito padrão de ver reposta certa, e ao invés disso, duplicar o modelo 3D da alternativa correta chamado Aorta e arrastar o novo no campo AdditionalElements3 para usá-lo como um elemento adicional que apareça unicamente quando está sendo apresentada a resposta certa, ver “Usar objetos como Elementos Adicionais (AdditionalElements)” em “Instruções Do Uso Da Classe **`QuestionController2`**” em Apêndice C. Sugere-se renomear o novo modelo como AortaAnswer. A Figura F 23 apresenta o processo descrito.

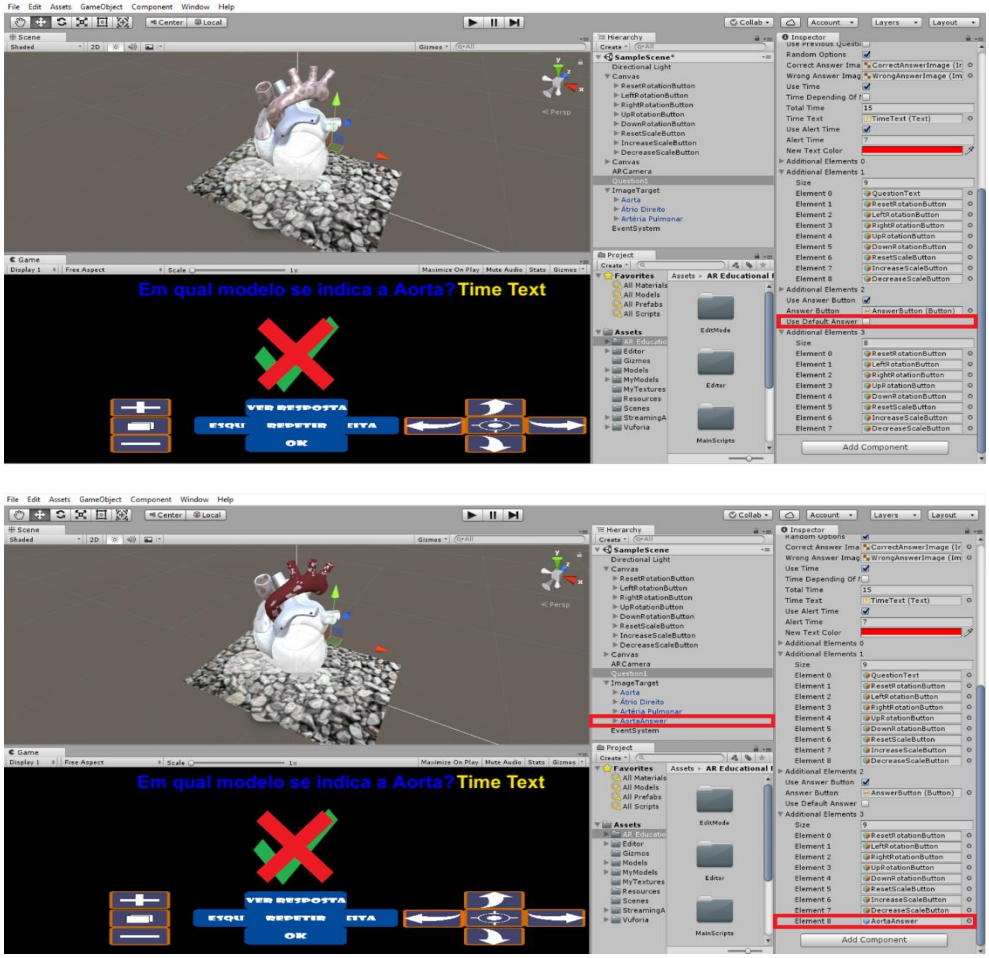

Figura F 23: Inicialmente é desativo o campo UseDefaultAnswer, e finalmente o modelo AortaAnswer (criado duplicando Aorta), é arrastado no campo AdditionalElements3.

12) Fazer os botões de rotacionar e escalar ficar associados também com o novo modelo 3D AortaAnswer (de jeito similar como mostrado no passo 6), permitindo rotacionar e escalar ele igual do que os outros modelos.

13) Criar um evento com a classe **`ActivateInformation`** e sua inicialização padrão, como explicado no "Tutorial Classe **`ActivateInformation`**" em: "Inicialização Default Da Classe **`ActivateInformation`**", ver Apêndice D. Sugere-se renomear o novo objeto vazio container para ActivateInformation, a cena deveria ficar como na Figura F 24.

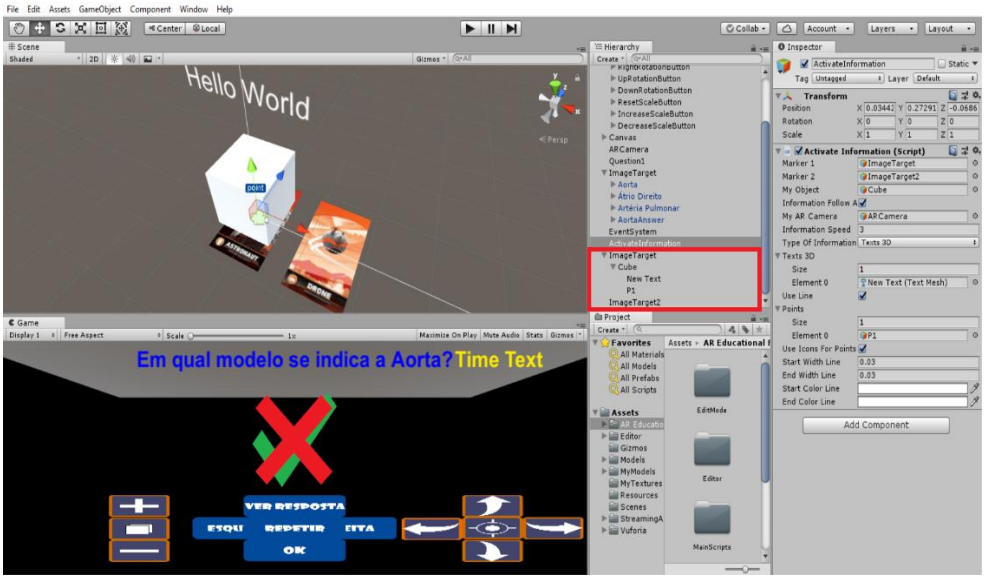

Figura F 24: Objeto cubo, texto 3D e marcadores criados com `ActivateInformation`. O objeto vazio criado foi renomeado para ActivateInformation.

14) Atualizar os campos da Classe **`ActivateInformation`** para usar o modelo 3D AortaAnswer com seu marcador, ao invés dos elementos padrão, como foi explicado no "Tutorial Uso Classe **`ActivateInformation`** Em Perguntas", em: "Atualizar Classe **`ActivateInformation`** Para Ficar Associada Com Elementos Da Pergunta", ver Apêndice D. Após isso, lembre-se que os elementos desnecessários deverão ser removidos, e a cena deveria ficar como na Figura F 25. Nota-se também, que o texto 3D NewText, e o elemento P1 têm sido reposicionados.

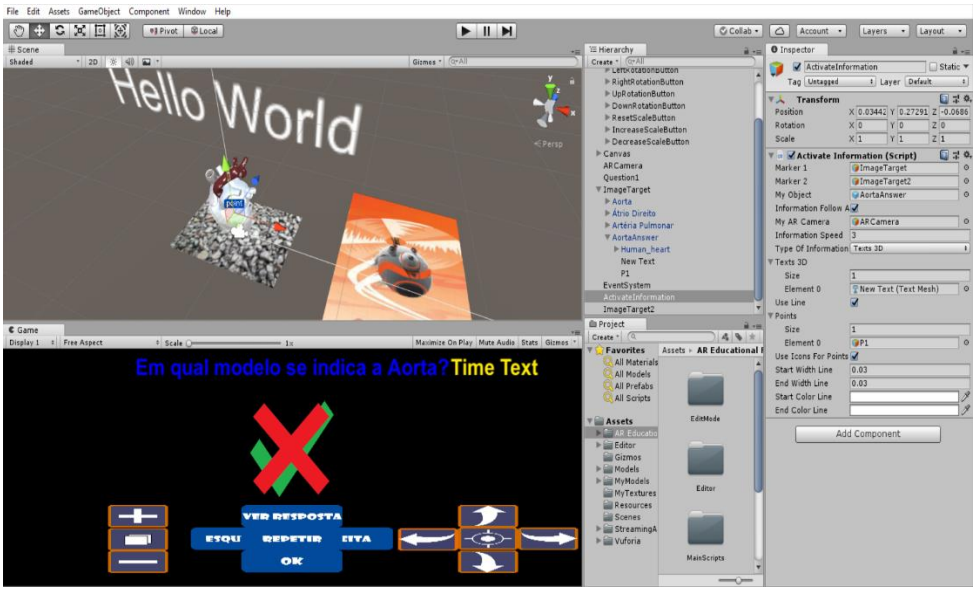

Figura F 25: Classe `ActivateInformation` atualizada para usar modelo 3D AortaAnswer e seu marcador.

Adicionalmente, atualizar o marcador *default* **`imageTarget2`**, para usar o novo marcador (chamado marker2) disponível no banco do dados dos novos marcadores importados, tal como foi feito no passo 4. A cena deveria ficar como na Figura F 26.

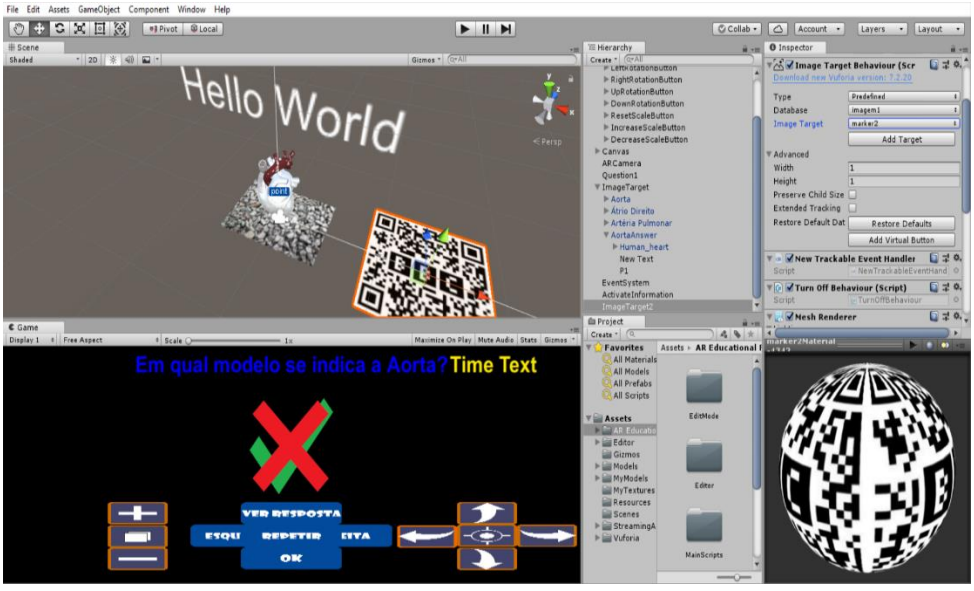

Figura F 26: Marcador `imageTarget2` atualizado para usar novo importado.

15) Posicionar o ponto P1 (que indica a origem da linha do texto 3D), para ficar na posição da Aorta, como mostrado na Figura F 27. Além disso, renomear o texto 3D newText para: “Aorta”.

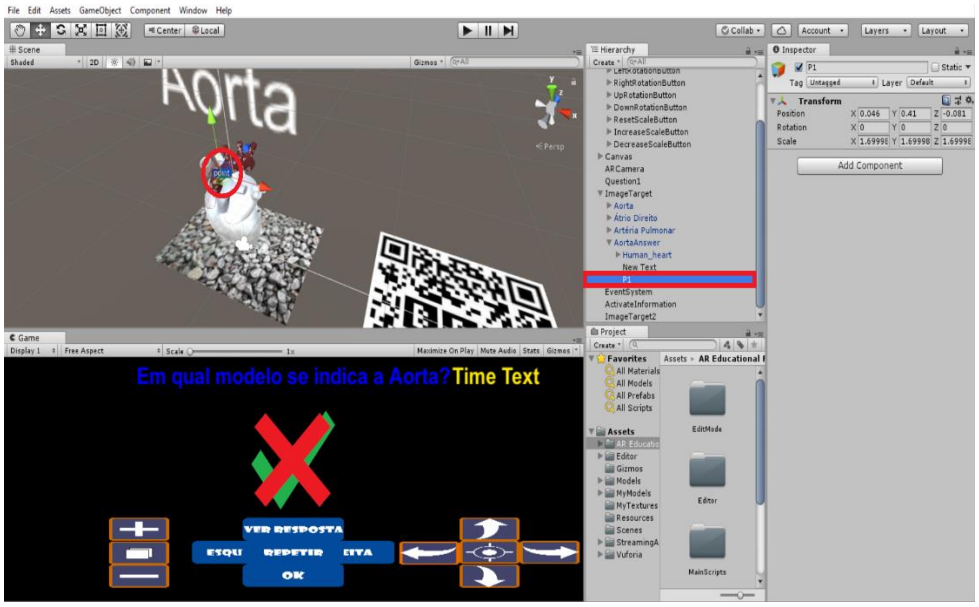

Figura F 27: Ponto P1, posicionado no componente do coração chamado Aorta. Texto 3D renomeado para Aorta.

16) Uma vez que neste ponto existem dois marcadores simultâneos na cena, este passo consiste em fazer a configuração sugerida com vários marcadores, falada no “Tutorial Configuração AR Educational Framework”, em “Configuração Adicional Sugerida Usando Vários Marcadores”, ver Apêndice A.
17) Criar um evento com a classe **`ChangeElement`** e sua inicialização padrão, como explicado no “Tutorial Classe **`ChangeElement`**”, em: “Inicialização Default Da Classe **`ChangeElement`**”, ver Apêndice D. Sugere-se renomear o novo objeto vazio container para ChangeElement, a cena deveria ficar como na Figura F 28.

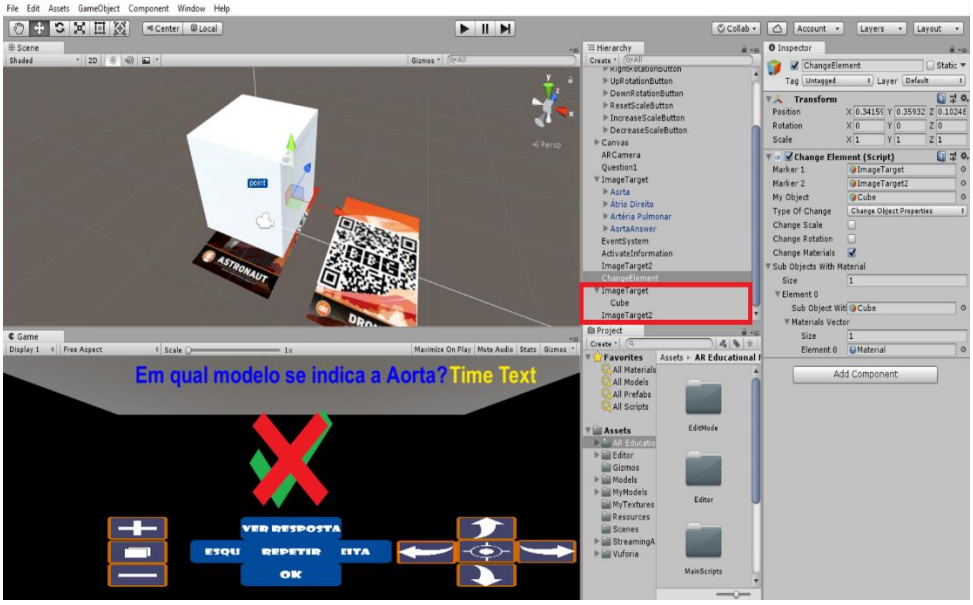

Figura F 28: Novo objeto Cube e marcadores criados com `ChangeElement`. O novo objeto vazio container foi renomeado para ChangeElement.

18) Atualizar os campos da Classe **`ChangeElement`** para usar o modelo 3D AortaAnswer com os marcadores originais, ao invés dos elementos padrão, e neste caso para permitir trocar os materiais dos componentes do coração transparentes, por materiais com cores adequados para ditos componentes, como explicado em Apêndice D, no "Tutorial Uso Classe **`ChangeElement`** Em Perguntas", em: "Atualizar Classe **`ChangeElement`** Para Ficar Associada Com Elementos Da Pergunta", (para o processo de atualizar os campos), e no "Tutorial Classe **`ChangeElement`**", em: "Configurar Classe **`ChangeElement`** Para Usar Modelos 3D Próprios", (para o processo de definir os materiais a serem trocados). Após isso, lembre-se que os elementos desnecessários deverão ser removidos, e a cena deveria ficar como na Figura F 29.

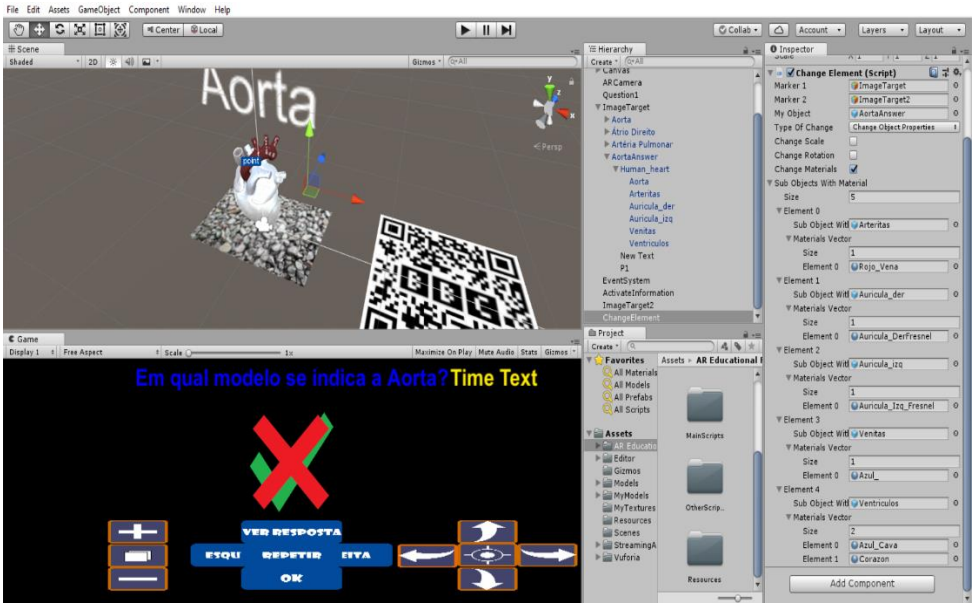

Figura F 29: Classe `ChangeElement` atualizada para usar o modelo 3D AortaAnswer e os marcadores originais.

Nota: Usar como guia o modelo do coração original Heart (disponível no pacote Heart importado), para saber quais são os materiais de cada componente do coração específico.

19) Como passo final, o tempo se configura a 20 segundos em TotalTime, ficando a cena pronta para ser executada.

Apêndice G

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO - PARTICIPANTES DO EXPERIMENTO

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

1. Você está sendo convidado para participar da pesquisa “UM FRAMEWORK PARA APOIO AO DESENVOLVIMENTO DE APLICAÇÕES EDUCACIONAIS DE REALIDADE AUMENTADA MÓVEL BASEADA EM MARCADORES”.
2. Propõe-se um *Framework* chamado AR Educational *Framework*, para apoiar o desenvolvimento de aplicativos educativos com a tecnologia de Realidade Aumentada Móvel (RAM) baseada em marcadores. Espera-se com este *Framework*, que tanto desenvolvedores experientes quanto iniciantes no Unity, possam criar aplicações educativas de Realidade Aumentada com facilidade, implementando funcionalidades sem necessidade de programação (arrastando e soltando ao invés disso).
    a) Você foi selecionado e sua participação não é obrigatória.
    b) Os objetivos deste estudo são validar a utilização do *Framework*.
    c) Sua participação nesta pesquisa consistirá em utilizar o *Framework* proposto seguindo as orientações, e responder a um questionário final.
3. Os participantes são escolhidos de forma aleatória, não é requisito ter conhecimentos de programação para participar no experimento, nem para

utilizar o *Framework*, embora é recomendado ter conhecimentos básicos na interface do Unity, e noções da Vuforia.

4. Este experimento baseia-se no teste do *Framework* com estudantes de cursos de computação com nível iniciante ou intermediário no Unity, visando medir usabilidade e dificuldade.
5. Este experimento será realizado no laboratório de objetos de aprendizagem (LOA) da Universidade Federal De São Carlos, no qual você está com grande frequência, a fim de diminuir os riscos de desconfortos.
6. Os dados que serão publicados desse estudo não te identificarão devido ao uso de siglas e números para referir a você, quando necessário.
7. Há risco de o *Framework* não permitir criar com facilidade aplicações educativas de Realidade Aumentada.
8. Não se aplica nenhum risco ao participante por usar o *Framework* ou de participar nestes experimentos.
9. Será ministrado uma primeira sessão em formato treinamento para conhecer conceitos básicos do Unity e da Vuforia, e após isso duas sessões consistindo em treinamentos de duas funcionalidades do *Framework*, e a aplicação de um exercício proposto usando cada funcionalidade. Será solicitado informações sobre utilização do *Framework* e sugestões.
10. A qualquer momento você pode desistir de participar e retirar seu consentimento. Sua recusa não trará nenhum prejuízo a sua relação com o pesquisador ou instituição.
11. As informações obtidas através dessa pesquisa serão confidenciais, e asseguram o sigilo sobre sua participação. Os dados não serão divulgados de forma a possibilitar sua identificação.
12. Caso necessite de uma cópia deste termo, ele pode ser solicitado junto ao Pesquisador Marcos Tulio Soto De La Vega pelo e-mail: mt-soto@hotmail.com.

_____

Marcos Tulio Soto De La Vega
mt-soto@hotmail.com / (55) 11 983207221

São Carlos, ____ de _____ de 2018.

_____

Sujeito de Pesquisa

Apêndice H

QUESTIONÁRIO APLICADO PARA VALIDAR O EXPERIMENTO

QUESTIONÁRIO

Nome: _____

Levantamento de seu conhecimento:

Legenda:
1- Nenhum
2- Básico
3- Intermediário
4- Alto
5- Avançado

Conhecimento em Unity
[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Conhecimento em Vuforia
[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Conhecimento em Realidade Aumentada
[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Sobre o experimento com o AR Educational Framework

Legenda alternativas: 1- Não concordo totalmente, 2- Não concordo parcialmente, 3-Indiferente, 4- Concordo parcialmente, 5- Concordo totalmente.

Realizei com sucesso as atividades previstas no experimento 1
[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Realizei com sucesso as atividades previstas no experimento 2
[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Treinamento - Foi essencial para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Material de Apoio (Tutoriais) - Foi essencial para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Só o treinamento seria suficiente para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Só o Material de Apoio seria suficiente para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

É necessário que o nível de conhecimento em Unity seja avançado para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

É necessário que o nível de conhecimento em Vuforia seja avançado para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

É necessário que o nível de conhecimento em Realidade Aumentada seja avançado para o desenvolvimento dos experimentos

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

De forma geral, a utilização do Framework pode ser considerada de nível avançado

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

O Framework fornece um pacote de funcionalidades altamente customizáveis de acordo às necessidades do usuário

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

O Framework fornece um pacote de funcionalidades uteis em qualquer área da educação (p.ex. Matemática, Química etc.)

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

O Framework permite criar cenários educativos simples de Realidade Aumentada com facilidade

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

O Framework fornece um pacote de funcionalidades suficientes para criar aplicações educativas com Realidade Aumentada

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

O Framework permitiria o desenvolvimento de aplicações educativas de Realidade Aumentada com facilidade

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Utilizaria o Framework para criar aplicações educativas de Realidade Aumentada

[ ] 1 [ ] 2 [ ] 3 [ ] 4 [ ] 5

Comentários e opiniões sobre o AR Educational Framework

1) Como você avalia sua experiência de utilizar o Framework para desenvolver cenários educativos de Realidade Aumentada?

2) Quais são suas sugestões e críticas em relação ao Framework proposto?

3) Quanto você acha que o Framework iria apoiar o desenvolvimento de aplicações educativas de Realidade Aumentada?

Apêndice I

TUTORIAIS DOS EXPERIMENTOS (MATERIAL DE APOIO)

Tutorial Classe QuestionController2

Inicialização Default Da Classe QuestionController2

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController2`** com sua inicialização padrão, que vai permitir criar automaticamente uma pergunta (com elementos padrões de exemplo), na qual as alternativas são elementos 3D em RA vinculados a um marcador, e apresentados por turno. Para isso, devem-se seguir os seguintes passos:

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura I 1, que será usado como container para as classes que permitirão criar a pergunta (classes **`QuestionController2`** e **`QuestionController2E`**).

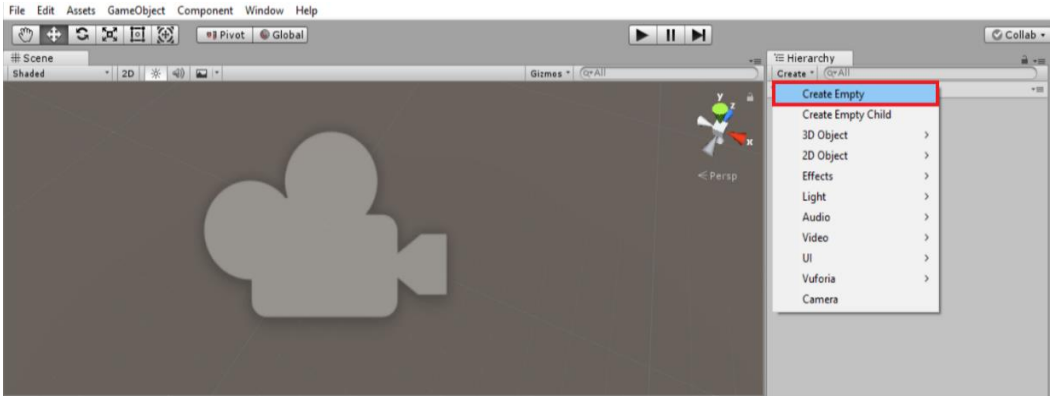

Figura I 1: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController2`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura I 2.

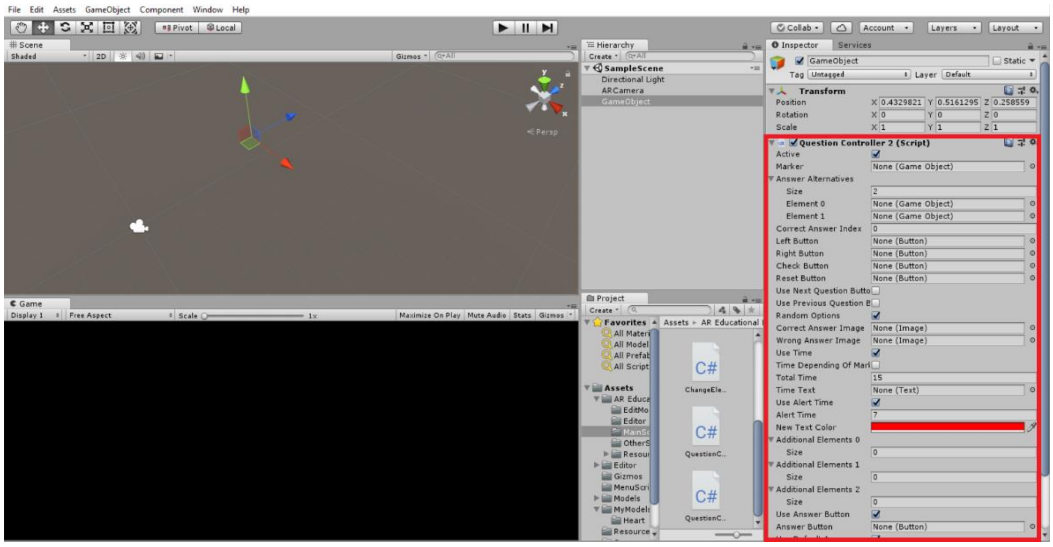

Figura I 2: Classe `QuestionController2` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`QuestionController2E`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default*, após isso recomenda-se fazer click no marcador criado **`ImageTarget`** para fazer ele visível na cena.

   A cena deveria ficar como na Figura I 3, já pronta para ser executada, com um cubo e uma esfera (que representam as alternativas de resposta) sobre um marcador, no centro duas imagens de *feedback* de resposta correta e errada, com botões para ver resposta correta e repetir a pergunta, assim como botões de passar à próxima alternativa, à

alternativa anterior, e escolher a alternativa atual (lembrando que as alternativas são por turno), finalmente no corner superior direito está localizado um texto para mostrar o tempo, e na parte superior da tela dois textos adicionais para pergunta e resposta respetivamente.

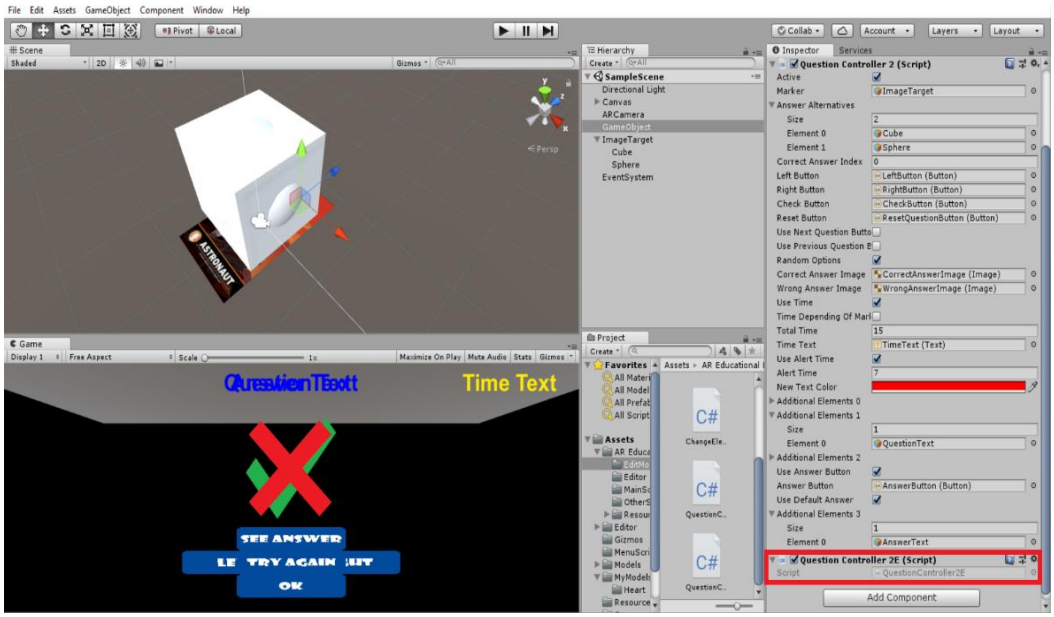

Figura I 3: Classe `QuestionController2E` arrastada no objeto vazio, e cena já pronta para ser executada.

4) Após ser feita a inicialização padrão recomenda-se remover a classe **`QuestionController2E`**.

Configurar Classe QuestionController2 Para Usar Outros Modelos 3D

A seguir, serão apresentados os passos para aplicar a mesma funcionalidade usando uma cápsula do Unity como a primeira alternativa, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado, após isso, será utilizado outro novo modelo como uma terceira alternativa.

1) Criar o novo modelo 3D, no caso um elemento Capsule do Unity (**`GameObject/3D Object/Capsule`**).
2) Substituir o novo modelo 3D (Capsule) pelo cubo, para isso: Colocar o modelo Capsule na mesma posição do cubo, e faze-lo filho do mesmo marcador. A cena deveria ficar como na Figura I 4. Nota: O elemento Capsule também foi diminuído de tamanho.

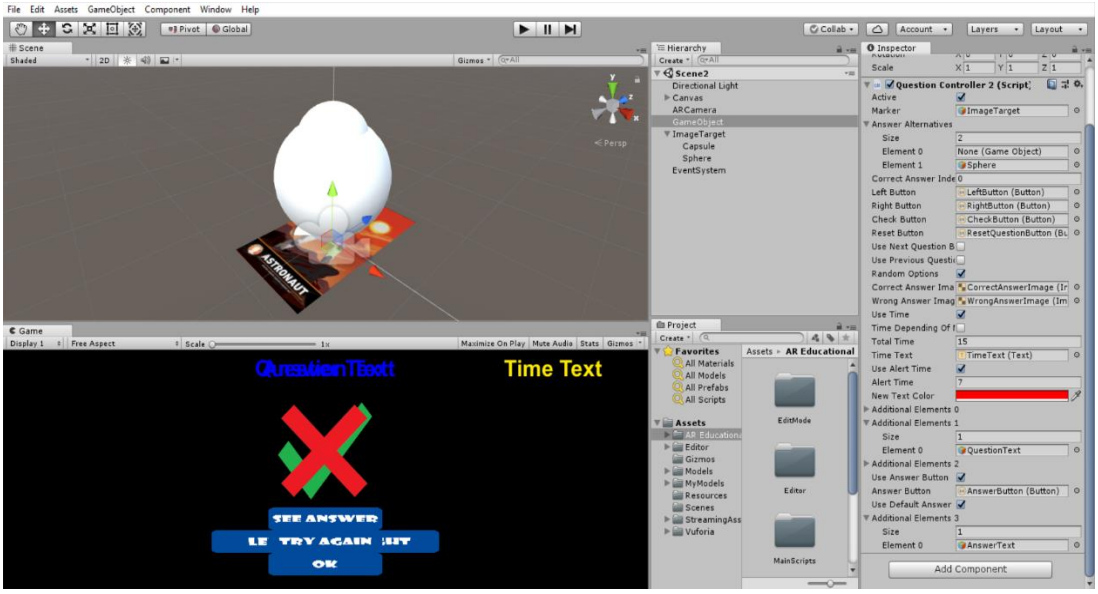

Figura I 4: Novo modelo 3D Capsule, vinculado ao marcador `ImageTarget`.

3) Finalmente arrastar o novo modelo 3D Capsule, no campo AnswerAlternatives da classe **`QuestionController2`**. A cena deveria ficar como na Figura I 5.

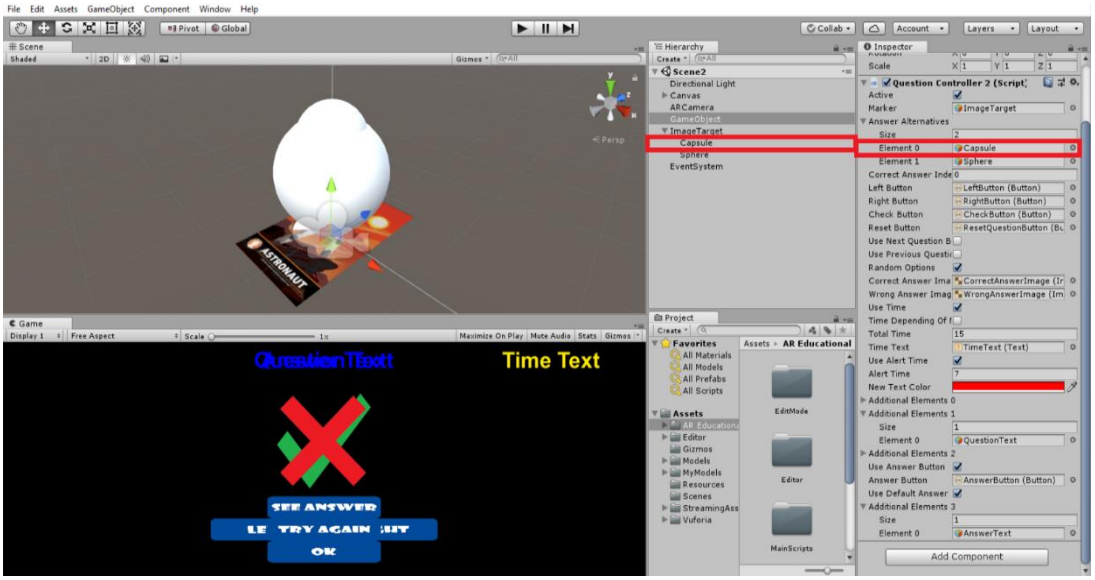

Figura I 5: Novo modelo 3D Capsule arrastado no campo AnswerAlternatives.

Neste ponto, a cena fica pronto para ser executada. Para substituir o segundo elemento da pergunta por outro, devem-se seguir os mesmos passos já descritos.

Adicionalmente, será utilizado um novo modelo como a terceira alternativa da pergunta. Para isso:

4) Criar o novo modelo 3D, no caso um elemento Cylinder do Unity (**`GameObject/3D Object/Cylinder`**).
5) Colocar o modelo Cylinder na mesma posição dos outros modelos, e faze-lo filho do mesmo marcador. A cena deveria ficar como na Figura I 6. Nota: O elemento Cylinder também foi diminuído de tamanho.

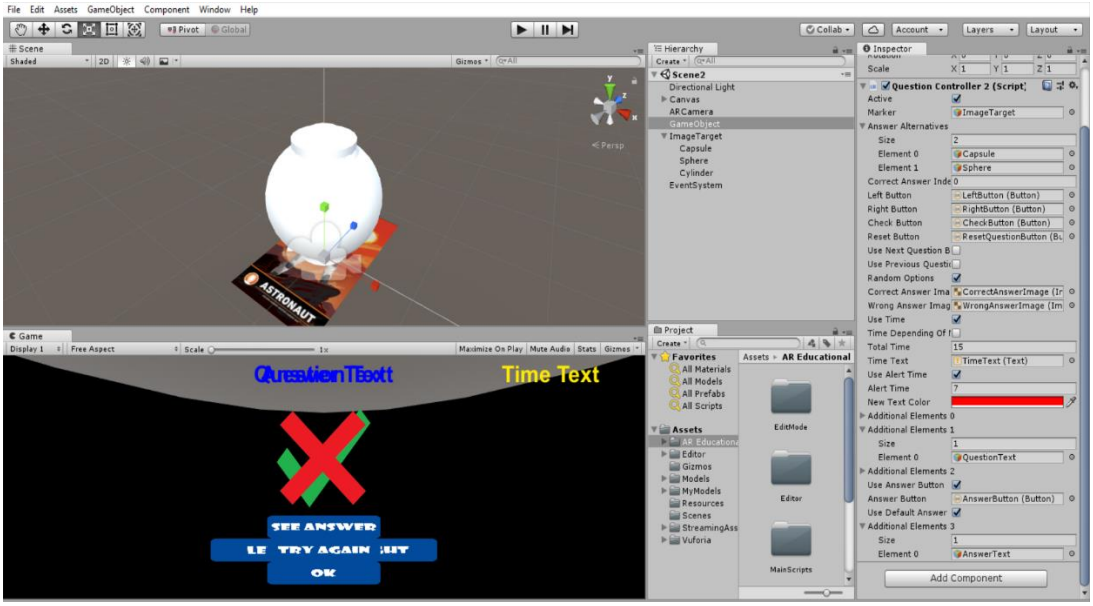

Figura I 6: Novo modelo 3D Cylinder, vinculado ao marcador `ImageTarget`.

6) Criar uma nova alternativa na pergunta, mudando para 3 o campo Size de AnswerAlternatives, e arrastando o novo modelo 3D Cylinder no novo campo criado. O processo anterior é mostrado na Figura I 7.

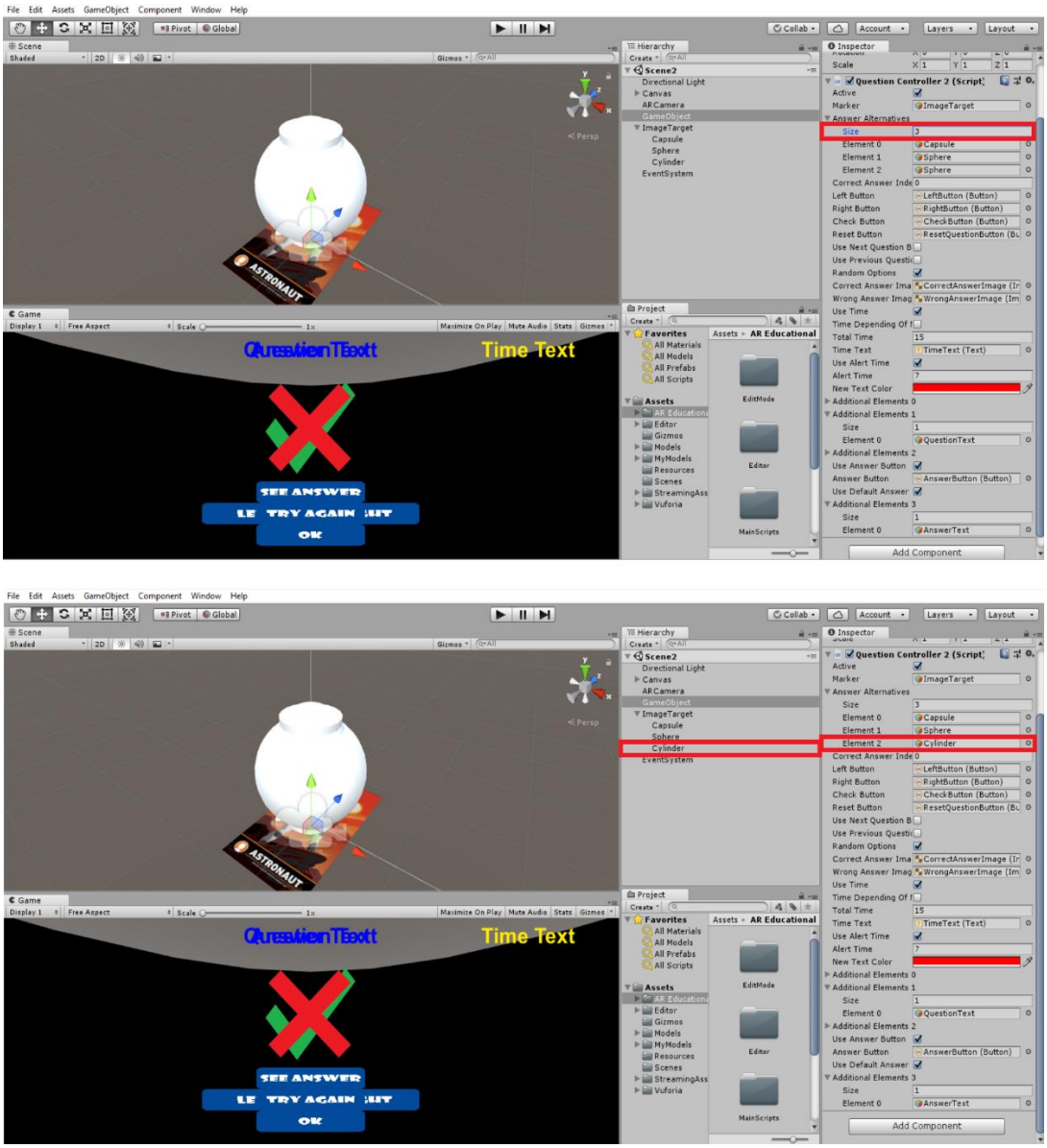

Figura I 7: Novo modelo 3D Cylinder usado como a terceira alternativa da pergunta. Cena pronta para ser executada.

Tutorial Classe ActivateInformation

Inicialização Default Da Classe ActivateInformation

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`ActivateInformation`** com sua inicialização padrão, que vai permitir criar automaticamente um objeto cubo vinculado a um marcador de RA, onde vai ser ativado um texto 3D sobre o objeto sempre que seu marcador estiver junto de um

outro marcador adicional (chamado marcador de controle). Para isso, devem-se seguir os seguintes passos:

1) Criar um objeto vazio do Unity (**`GameObject/Create Empty`**), ver Figura I 8, que será usado como container para as classes que permitirão fazer o evento de ativar o texto 3D sobre o objeto (classes **`ActivateInformation`** e **`ActivateInformationE`**).

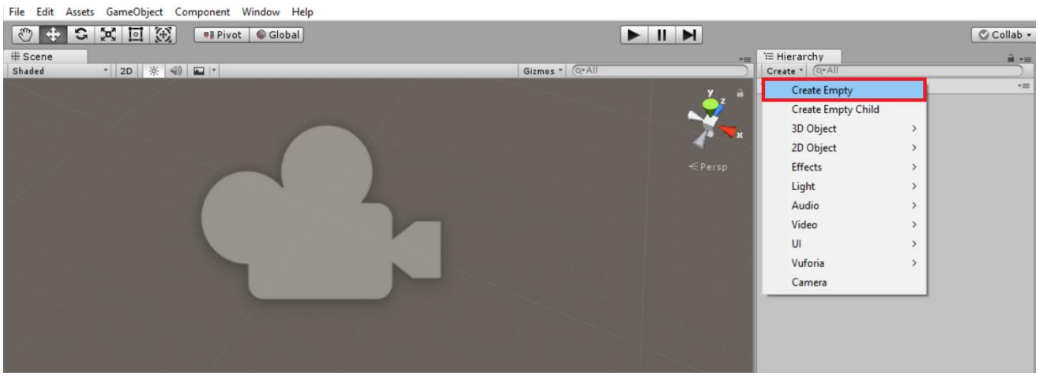

Figura I 8: Criação de um objeto vazio do Unity.

2) Selecionar o objeto vazio, e arrastar nele a classe **`ActivateInformation`** (contida na pasta **`MainScripts`**). A cena deveria ficar como na Figura I 9.

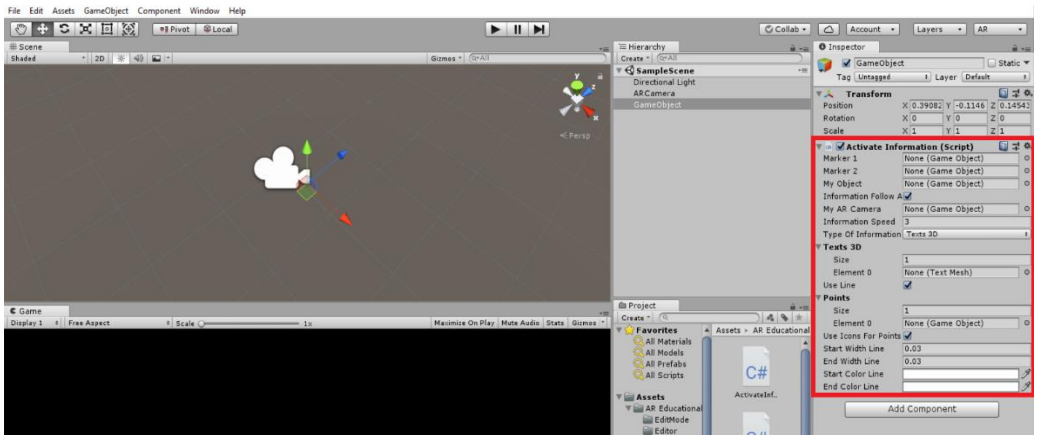

Figura I 9: Classe `ActivateInformation` arrastada no objeto vazio.

3) Selecionar o objeto vazio, e arrastar nele a classe **`ActivateInformationE`** (contida na pasta **`EditMode`**) para a criação automática dos elementos *default* (recomenda-se fazer click nos marcadores **`ImageTarget`** e **`ImageTarget2`** quando for criados para

fazer eles visíveis na cena). A cena deveria ficar como na Figura I 10, já pronta para ser executada, com o cubo sobre o marcador principal, acima do cubo o texto 3D (que vai ser ativado através do marcador de controle), e ao lado o marcador de controle (que ativará o texto 3D).

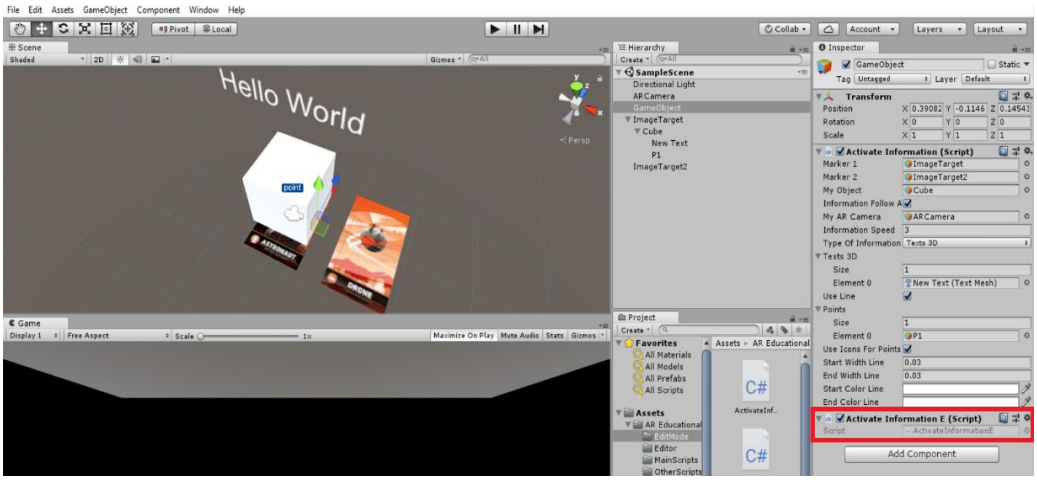

Figura I 10: Classe `ActivateInformationE` arrastada no objeto vazio, e cena pronta para ser executada.

4) Finalmente, após ser feita a inicialização padrão, recomenda-se remover a classe **`ActivateInformationE`**.

Configurar Classe ActivateInformation Para Usar Outros Modelos 3D

A seguir, serão apresentados os passos para aplicar a mesma funcionalidade com uma esfera do Unity, nota-se que deste mesmo jeito pode-se aplicar esta funcionalidade com qualquer modelo importado.

1) Criar o novo modelo 3D, no caso um elemento Sphere do Unity (**`GameObject/3D Object/Sphere`**).
2) Substituir o novo modelo 3D (Sphere) pelo cubo, para isso: Colocar o modelo Sphere na mesma posição do cubo, faze-lo filho do mesmo marcador, e finalmente fazer filhos dele aos elementos filhos do objeto cubo (elementos newText e P1). A cena deveria ficar como na Figura I 11.

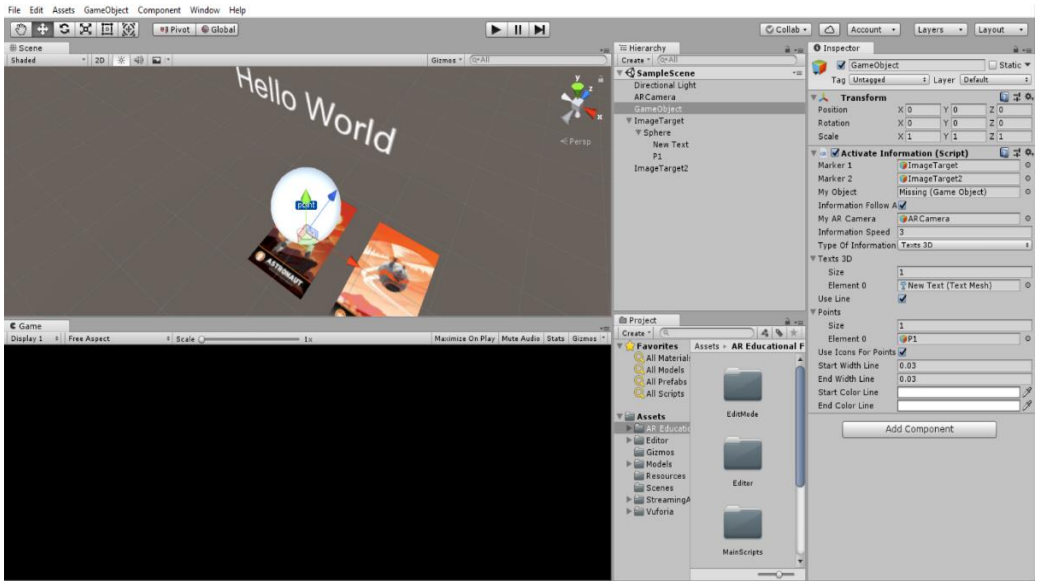

Figura I 11: Novo modelo 3D já vinculado ao marcador principal, os elementos newText e P1 que eram filhos do objeto cubo original, agora são filhos do novo modelo.

3) Finalmente arrastar o novo modelo 3D Sphere, no campo MyObject da classe **`ActivateInformation`**. A cena deveria ficar como na Figura I 12, já pronta para ser executada com Sphere como o novo objeto 3D em RA do evento **`ActivateInformation`**.

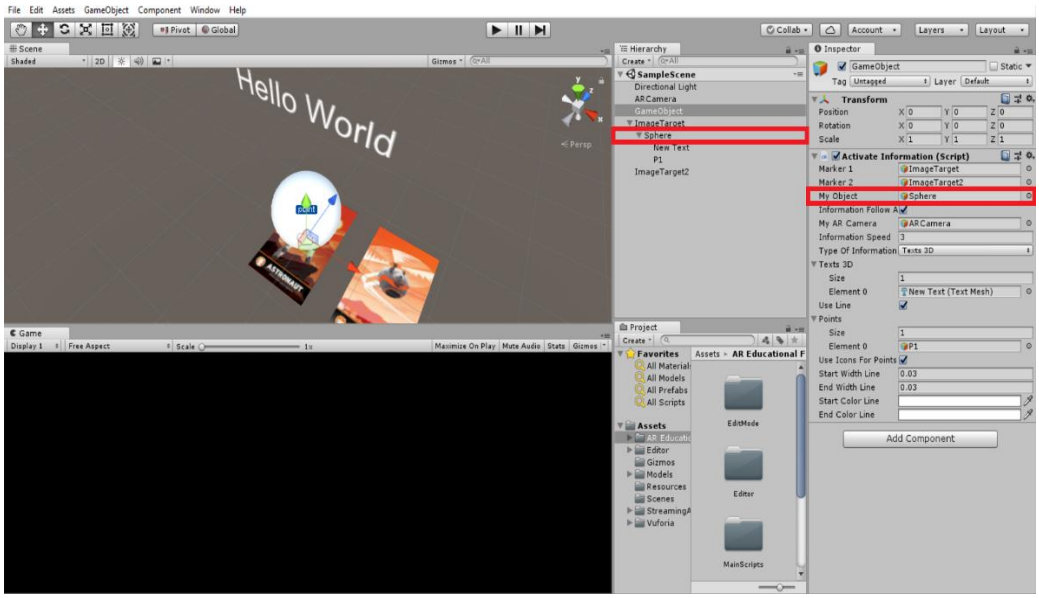

Figura I 12: Novo modelo 3D Sphere arrastado no campo MyObject, e cena pronta para ser executada.

Apêndice J

ATIVIDADES PROPOSTAS DOS EXPERIMENTOS

EXPERIMENTO 1 AR EDUCATIONAL FRAMEWORK (Classe QuestionController2)

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`QuestionController2`** para criar uma pergunta usando três modelos 3D indicando componentes específicos de um coração em RA. Para isso, devem-se seguir os seguintes passos:

1) Criar uma pergunta com a classe **`QuestionController2`** e sua inicialização padrão, como explicado no "Tutorial Classe **`QuestionController2`**", em: "Inicialização Default Da Classe **`QuestionController2`**". A cena deveria ficar como na Figura J 1.

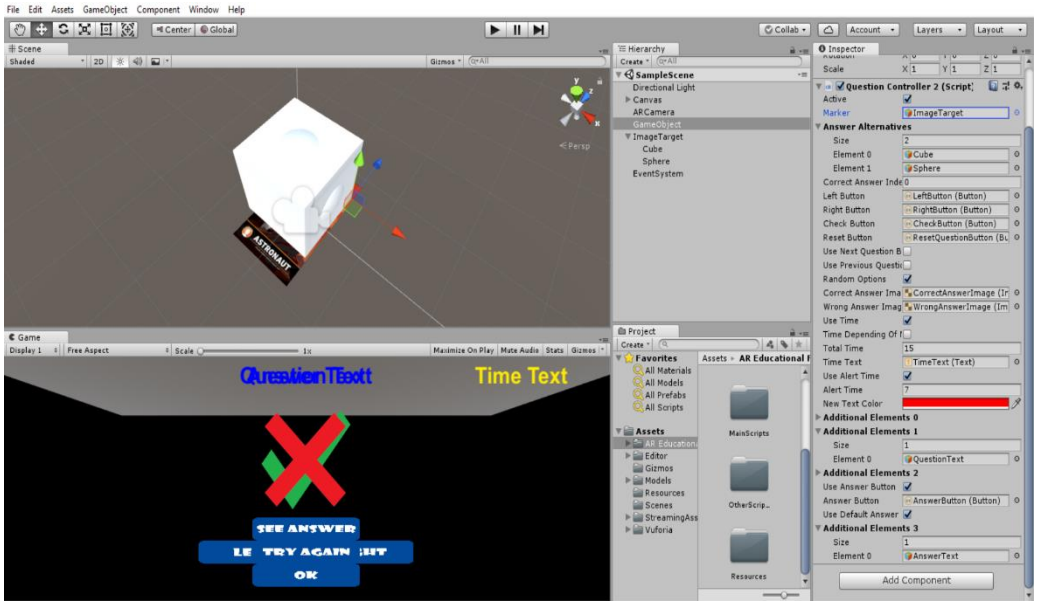

Figura J 1: Pergunta criada com `QuestionController2` e sua inicialização *default*.

2) Importar o pacote Heart de modelos no projeto, o qual possui os modelos necessários chamados Ventrículos, Aorta, Átrio Esquerdo. Nota: Para importar pacotes do Unity lembrar fazer **`Assets/ImportPackage/CustomPackage`**.
3) Realizar as configurações necessárias na pergunta criada com **`QuestionController2`**, para usar os novos modelos 3D Ventrículos e Aorta (caminho **`MyModels/Heart`**) ao invés do cubo e esfera respetivamente. A cena deveria ficar como na Figura J 2. Este processo foi explicado no "Tutorial Classe **`QuestionController2`**" em: "Configurar Classe **`QuestionController2`** Para Usar Outros Modelos 3D".

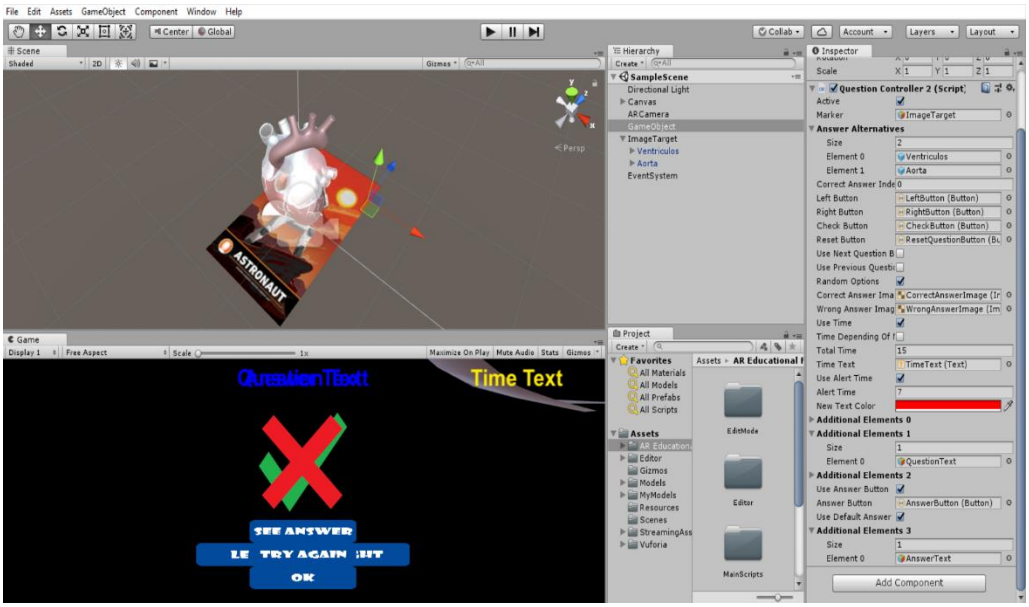

Figura J 2: Novos modelos 3D Ventrículos e Aorta, já associados com a pergunta.

4) Criar para a pergunta uma terceira alternativa usando o novo modelo 3D chamado Átrio Esquerdo. Este processo foi explicado no "Tutorial Classe **`QuestionController2`**" em: "Configurar Classe **`QuestionController2`** Para Usar Outros Modelos 3D". A cena deve ficar como na Figura J 3.

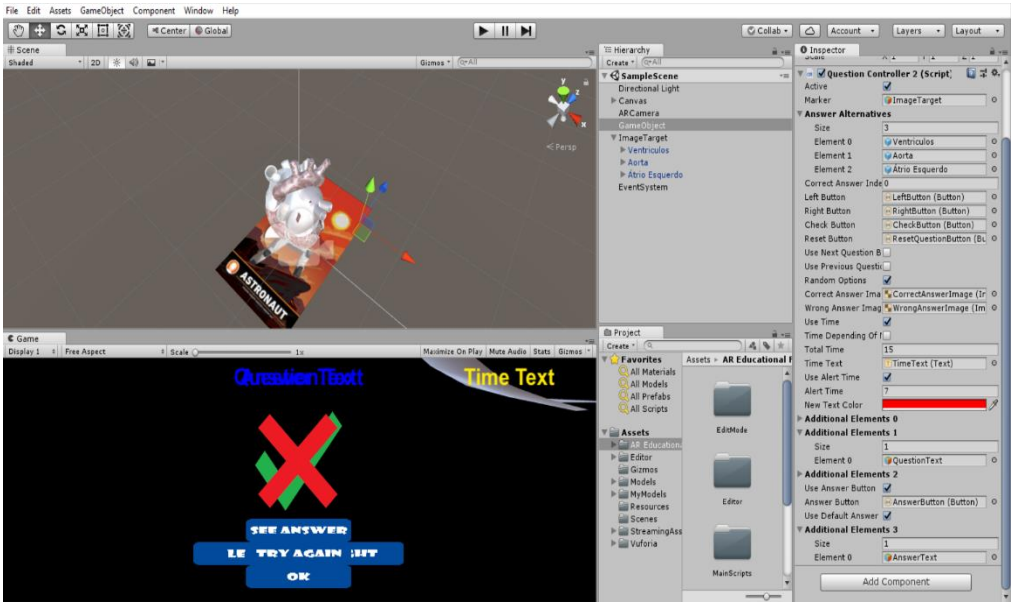

Figura J 3: Novo modelo 3D Átrio Esquerdo usado como a terceira alternativa da pergunta.

5) Renomear o texto de pergunta (QuestionText) para o texto real necessário, que será o seguinte: Em qual modelo se indicam os Ventrículos?, e renomear também o texto de resposta (AnswerText), que deverá ser o seguinte: Ventrículos. A cena deveria ficar como na Figura J 4.

   Nota 1: Se for necessário, movimentar os textos para não sobrepor o texto do tempo.

   Nota 2: Lembrar que para renomear o conteúdo dos textos (elementos Text), simplesmente deve-se selecionar o elemento dentro do **`Canvas`**, e mudar o texto no seu campo Text do inspetor do Unity.

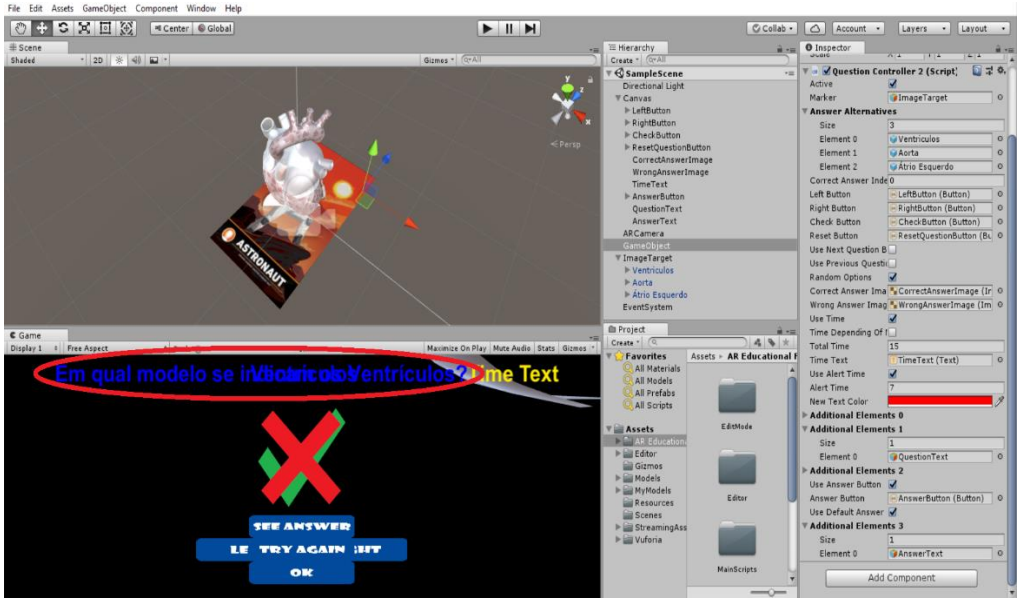

Figura J 4: Textos padrão de pergunta e resposta renomeados para a os textos necessários.

6) Mudar para 20 segundos o tempo total para resolver a pergunta.

EXPERIMENTO 2 AR EDUCATIONAL FRAMEWORK (Classe ActivateInformation)

Através do *framework*, e tendo ele já instalado no Unity, vai ser utilizada a classe **`ActivateInformation`** para criar um evento que permita ativar vários texto 3D com pontos respectivos, e indicar componentes específicos de um modelo 3D de um coração em RA. Para isso, devem-se seguir os seguintes passos:

1) Criar um evento com a classe **`ActivateInformation`** e sua inicialização padrão, como explicado no "Tutorial Classe **`ActivateInformation`**" em: "Inicialização Default Da Classe **`ActivateInformation`**". A cena deveria ficar como na Figura J 5.

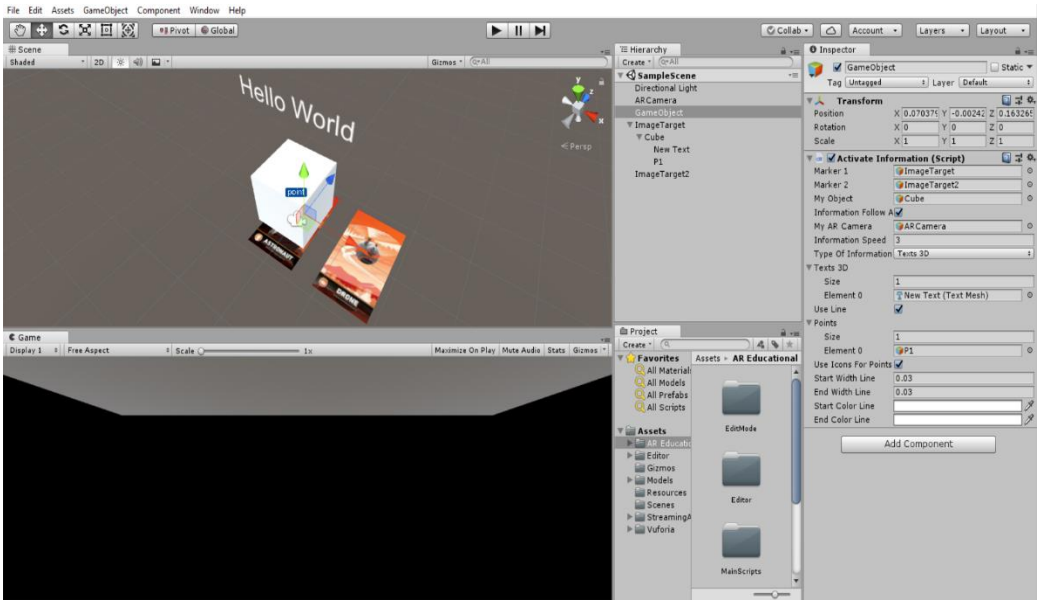

Figura J 5: Novo objeto cubo, texto 3D e marcadores criados com `ActivateInformation`.

2) Importar o pacote Heart de modelos no projeto, o qual possui o modelo necessário chamado Heart. Nota: Para importar pacotes do Unity lembrar fazer **`Assets/ImportPackage/CustomPackage`**.
3) Realizar as configurações necessárias para usar o novo modelo 3D Heart (caminho **`MyModels/Heart`**) ao invés do cubo, no evento da classe **`ActivateInformation`** que foi criado. A cena deveria ficar como na figura J 6. Este processo foi explicado no "Tutorial Classe **`ActivateInformation`**" em: "Configurar Classe **`ActivateInformation`** Para Usar Outros Modelos 3D".

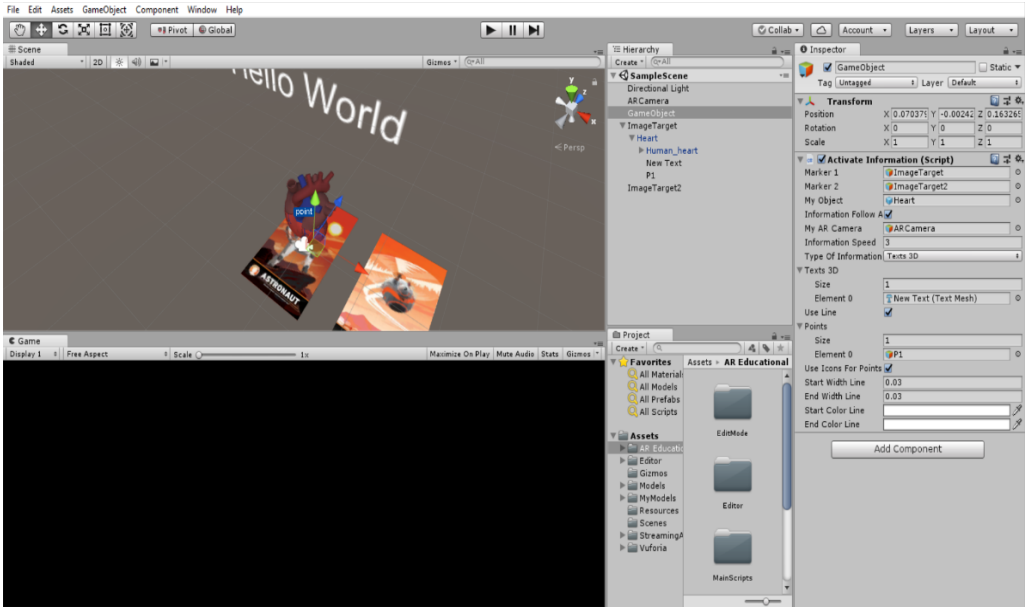

Figura J 6: Novo modelo 3D Heart, Já associado ao evento `ActivateInformation`.

4) Posicionar o ponto que indica a origem da linha (do texto 3D), para ficar no componente do coração chamado Ventrículos, como mostrado na Figura J 7. Além disso, posicionar o texto 3D (usado como informação adicional), para ficar mais próximo do objeto de jeito como se mostra na mesma Figura J 7.

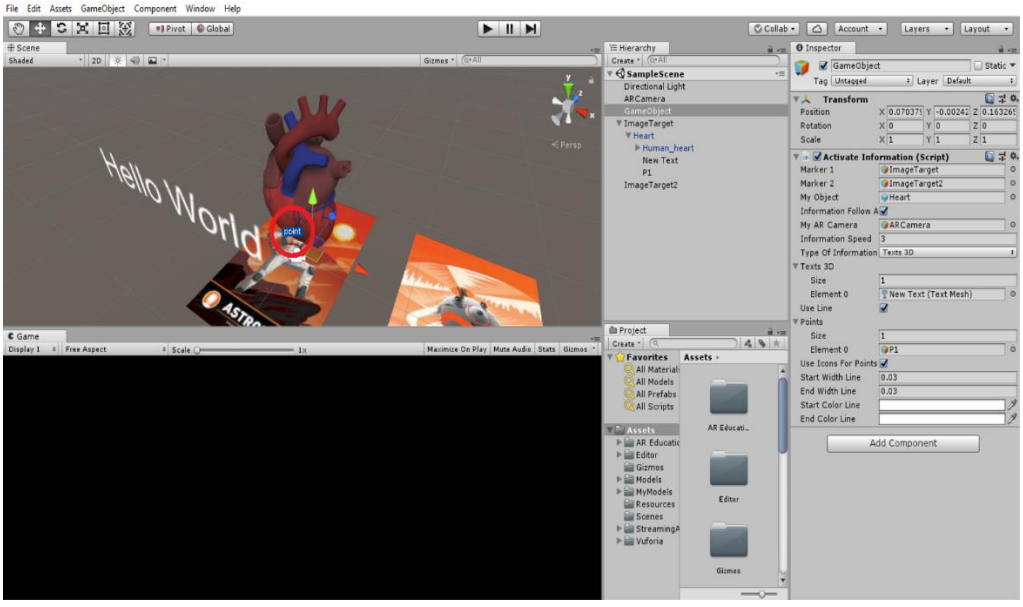

Figura J 7: Ponto (em círculo vermelho) posicionado no componente do coração chamado Ventrículos, e texto 3D posicionado mais para frente.

5) Renomear o conteúdo do texto 3D, para o texto: Ventrículos. A cena deveria ficar como na Figura J 8.

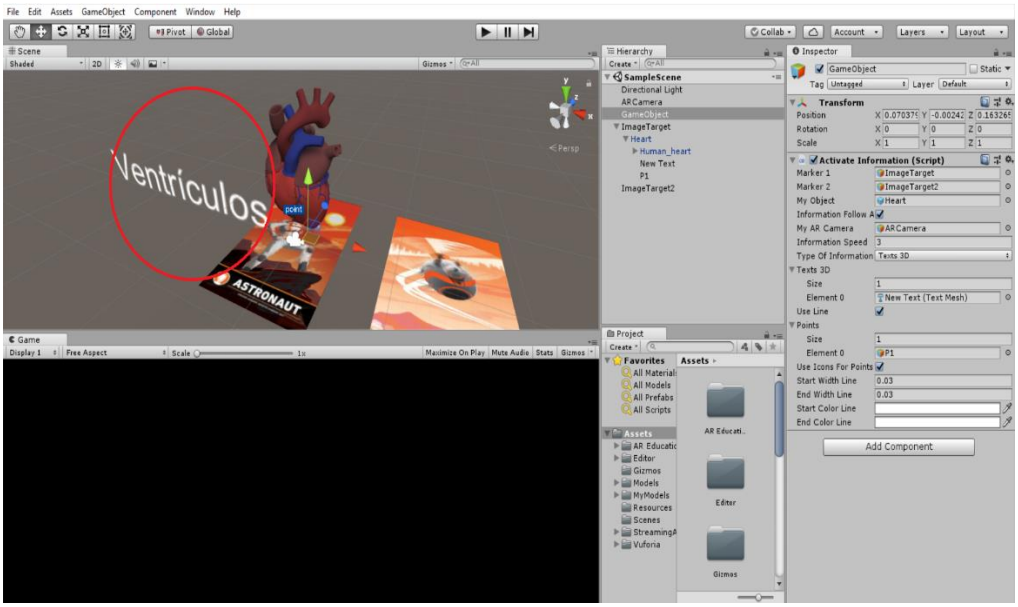

Figura J 8: Texto 3D já renomeado para: Ventrículos.

6) Criar dois novos textos e dois pontos respectivos (pode ser duplicando os elementos existentes), e utilizar eles para indicar os componentes do coração chamados: Átrio Esquerdo, e Aorta, para ficar de jeito similar como no passo anterior. A Figura J 9 mostra como deveria ficar a cena final com os dois novos textos e seus pontos respectivos.

Figura J 9: Cena final com dois novos textos e seus pontos respectivos.