

# **LIBRAR – CONCEITOS BÁSICOS DE LIBRAS USANDO REALIDADE AUMENTADA E REALIDADE VIRTUAL**

Aluno(a): Luan Ribeiro da Silva

Orientador: Dalton Solano dos Reis

# Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Implementação
- Resultados e discussões
- Conclusões e sugestões
- Extensões

# Introdução

- Ensino da Libras nas escolas
- Tornar mais divertido
- Inclusão da tecnologia

# Objetivos

Disponibilizar um sistema na plataforma móvel para o aprendizado dos conceitos básicos da Libras.

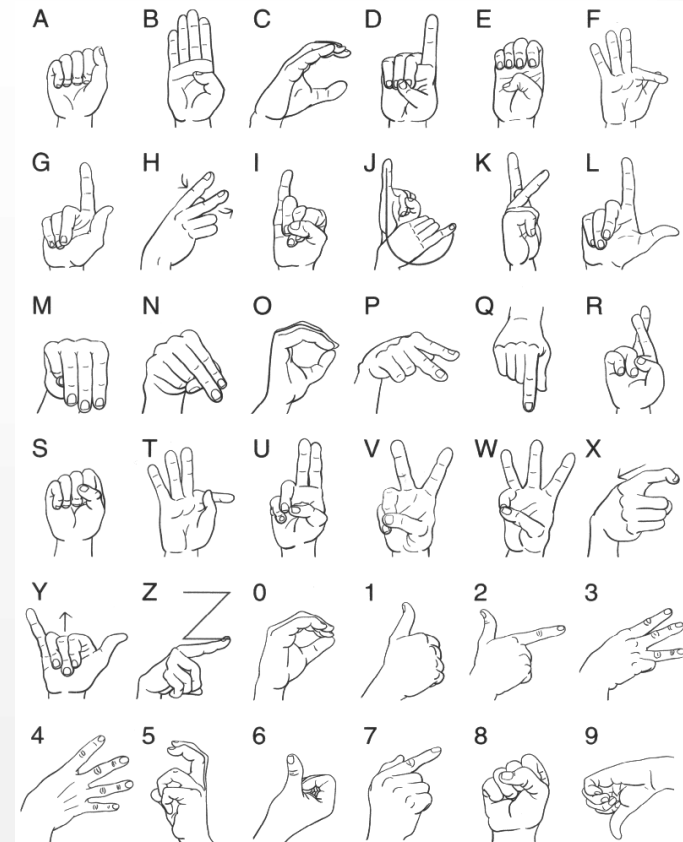
# Objetivos

- Disponibilizar ao usuário a possibilidade de visualizar todas as letras do alfabeto e algarismos numéricos em uma mão virtual 3D
- Disponibilizar ao usuário um jogo usando RA para poder entender os conceitos básicos da Libras
- Disponibilizar ao usuário um jogo usando RV para o mesmo ter um ponto de vista diferente do sinal na Libras enquanto entende os conceitos básicos

# Fundamentação Teórica

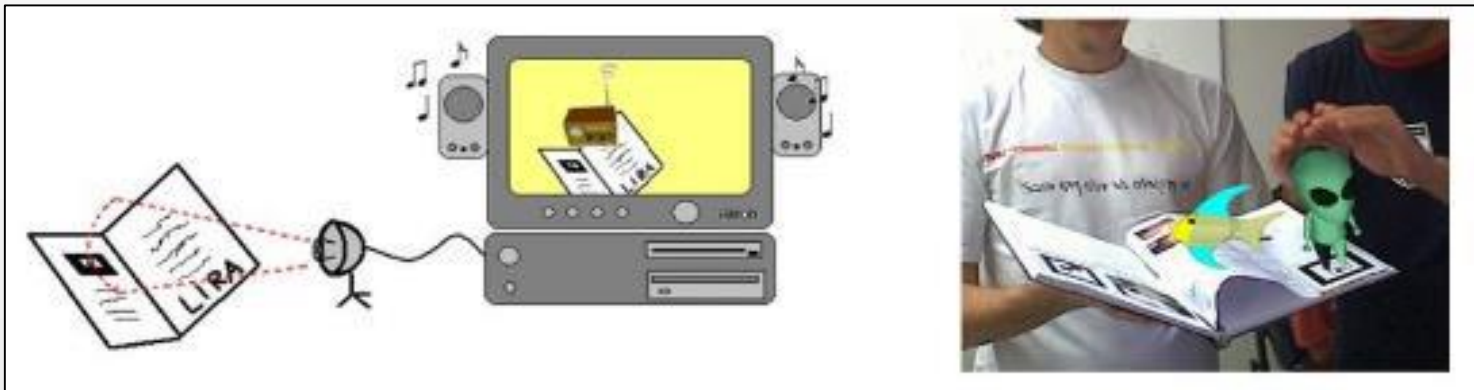
- Libras

- Língua Brasileira de Sinais
- Língua oficial dos deficientes auditivos
- É uma língua gestual-visual
- Português: sujeito → verbo → objeto
- Libras: Objeto → verbo → sujeito
- Português: eu vou para casa
- Libras: casa vou eu



# Fundamentação Teórica

- Realidade Aumentada
  - É uma interação entre o mundo real e virtual



# Fundamentação Teórica

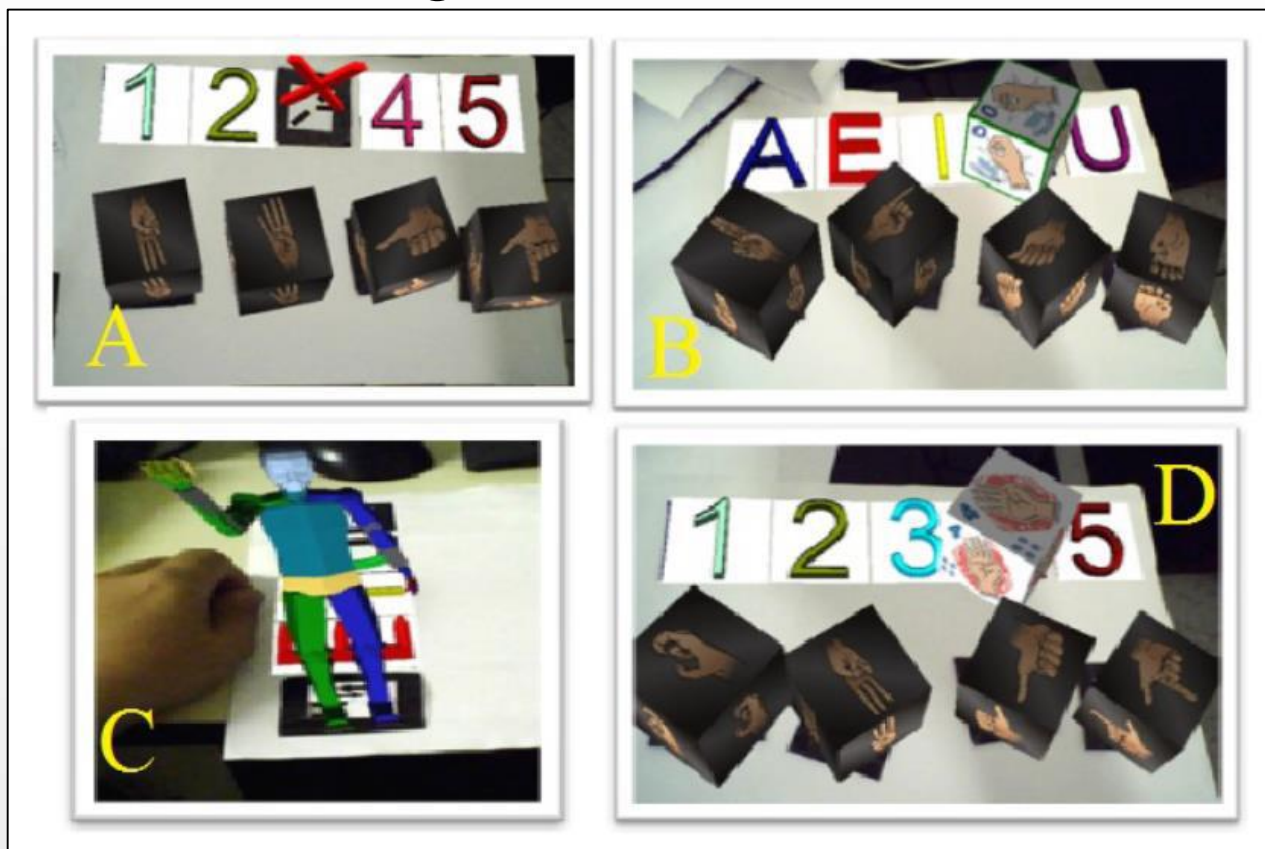
- Realidade Virtual
  - Interação com o usuário em um ambiente tridimensional
  - Sentido predominante é a visão





# Trabalhos Correlatos

- Realidade aumentada como ferramenta de apoio na alfabetização de crianças com surdez usuárias da língua brasileira de sinais



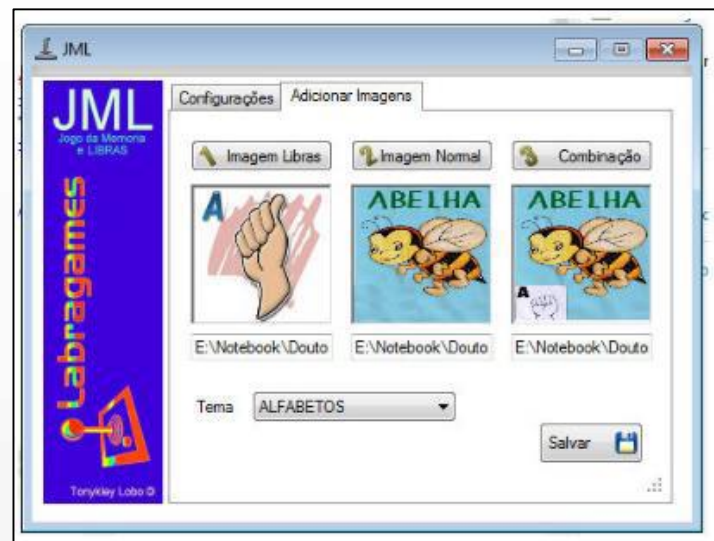
# Trabalhos Correlatos

- Aprendendo números em libras com a tecnologia da realidade aumentada



# Trabalhos Correlatos

- Jogando com a realidade aumentada e aprendendo libras



# Requisitos

- Requisitos funcionais
  - ter uma mão 3D para exibir o sinal em Libras
  - módulo para visualizar cada letra ou algarismo numérico em Libras
  - ter a possibilidade de rotacionar a mão 3D
  - módulo para treinar Libras com um jogo associativo usando realidade aumentada
  - utilizar oito marcadores de RA para o jogo associativo
  - exibir no lado esquerdo da tela os resultados do jogo associativo
  - módulo para treinar Libras com um jogo de raciocínio rápido usando realidade virtual e HMD

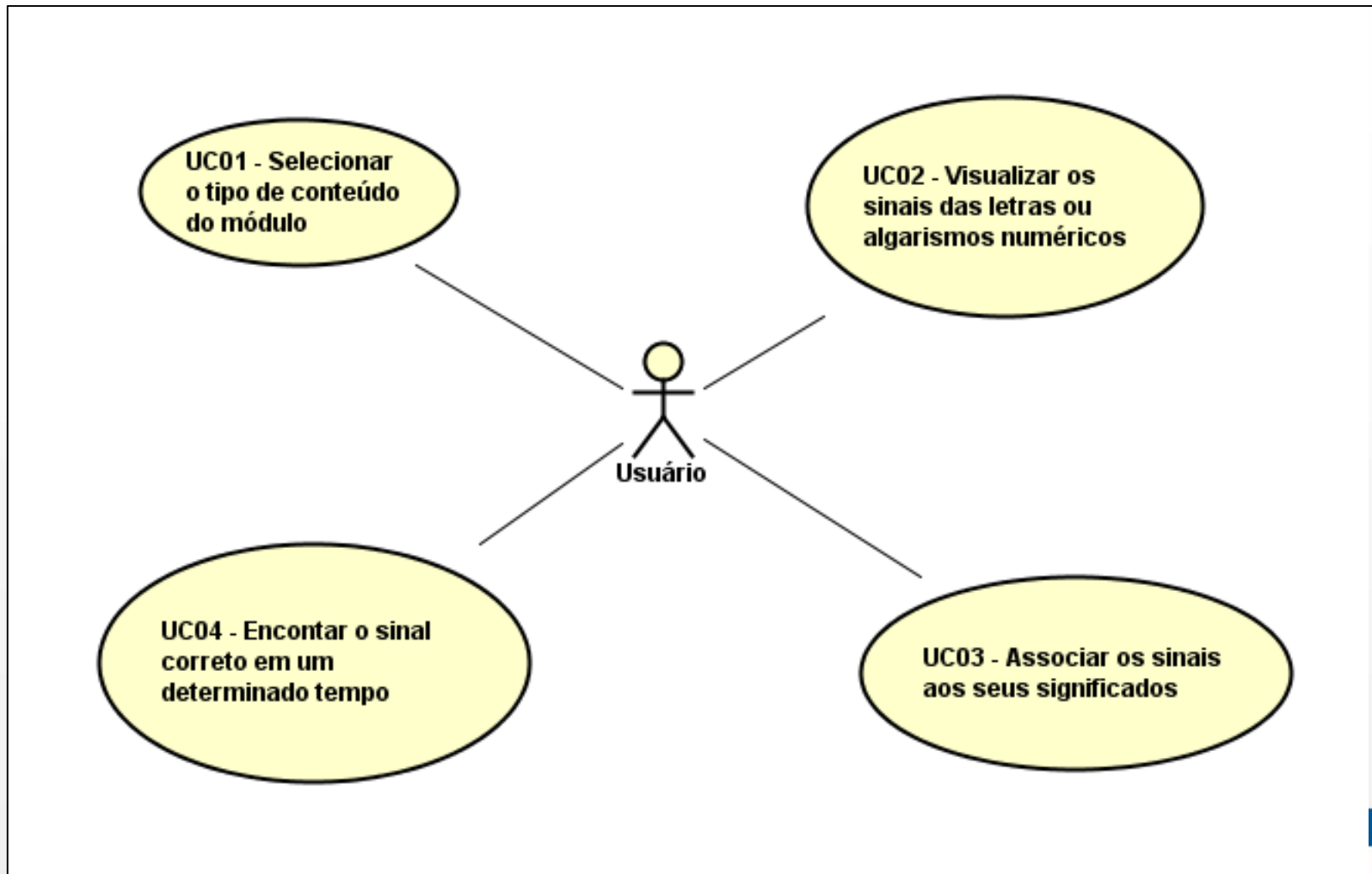
# Requisitos

- Requisitos funcionais
  - exibir um contador regressivo de vinte e cinco segundos durante o jogo de raciocínio rápido
  - exibir a quantidade de erros cometidas durante o jogo de raciocínio rápido, podendo ser no máximo duas vezes
  - tela inicial para escolher se o conteúdo será em cima de letras ou algarismos numéricos
  - botões para o usuário voltar a tela inicial, voltar a tela para escolher o conteúdo do módulo e executar novamente as animações dos sinais

# Requisitos

- Requisitos não funcionais
  - executar o sistema em dispositivos móveis com sistema operacional Android e iOS
  - utilizar o ambiente de desenvolvimento Unity para o desenvolvimento do sistema
  - utilizar o Vuforia junto com o Unity para o desenvolvimento da RA no sistema
  - utilizar o Google VR junto com o Unity para o desenvolvimento da RV no sistema
  - executar o sistema no modo offline

# Especificação





# Especificação

<<MonoBehaviour>>

## GameVRScript

- MaxTimer : float
- MaxErrors : int
- + progress : GameObject
- + menuObject : GameObject
- + gameVRObject : GameObject
- + aleatorySignalText : Text
- + secondText : Text
- + numberErrorsText : Text
- + finishText : Text
- + gameOverText : Text
- objectClick : GameObject
- scriptProgress : ProgressRadialBehaviour
- hands : GameObject[]
- letters : string[]
- numbers : string[]
- currentSignals : string[]
- aleatorySignals : string[]
- aleatorySignal : string
- timer : float
- beginTimer : boolean
- errors : int

- Start() : void
- Update() : void
- LoadVR() : IEnumerator
- CloseVR() : IEnumerator
- + BackButton() : void
- + StartProgress(objectClick : GameObject) : void
- + CancelProgress() : void
- + ExecuteAction() : void
- Init() : void
- + LearnNumbersButton() : void
- + LearnLettersButton() : void
- CreateSignals() : void
- + CheckChooseSignal() : void
- ShowHideHands(show : boolean) : void
- FinishGame() : void
- GameOver() : void
- + RestartGame() : void
- + ReloadSignalButton() : void

<<MonoBehaviour>>

## LearnSignalScript

- + uppercaseLetterText : Text
- + lowercaseLetterText : Text
- + numberText : Text
- + signalsObject : GameObject
- + menuObject : GameObject
- transformHand : Transform
- animatorHand : Animator
- letters : string[]
- numbers : string[]
- currentSignals : string[]
- positionCurrentSignal : int

- Start() : void
- Update() : void
- Init() : void
- PlayAnimationSignal() : void
- + BackButton() : void
- + MenuButton() : void
- + LearnNumbersButton() : void
- + LearnLettersButton() : void
- + NextSignalButton() : void
- + PreviousSignalButton() : void
- + ReloadSignalButton() : void

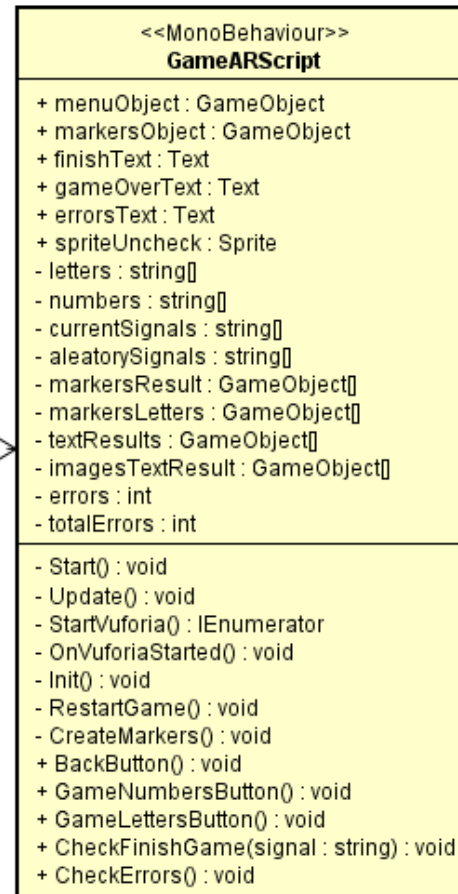
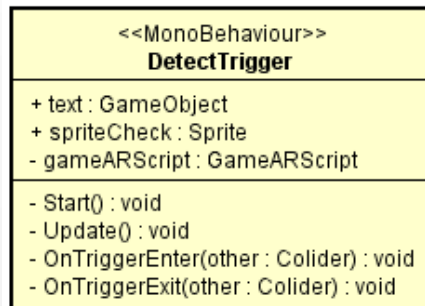
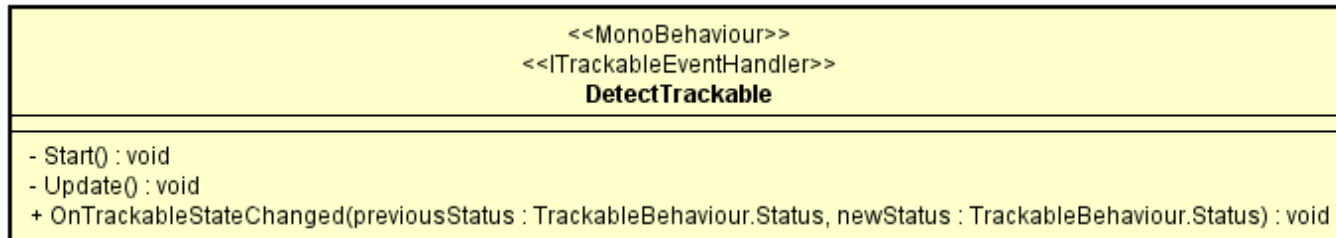
<<MonoBehaviour>>

## MainMenuScript

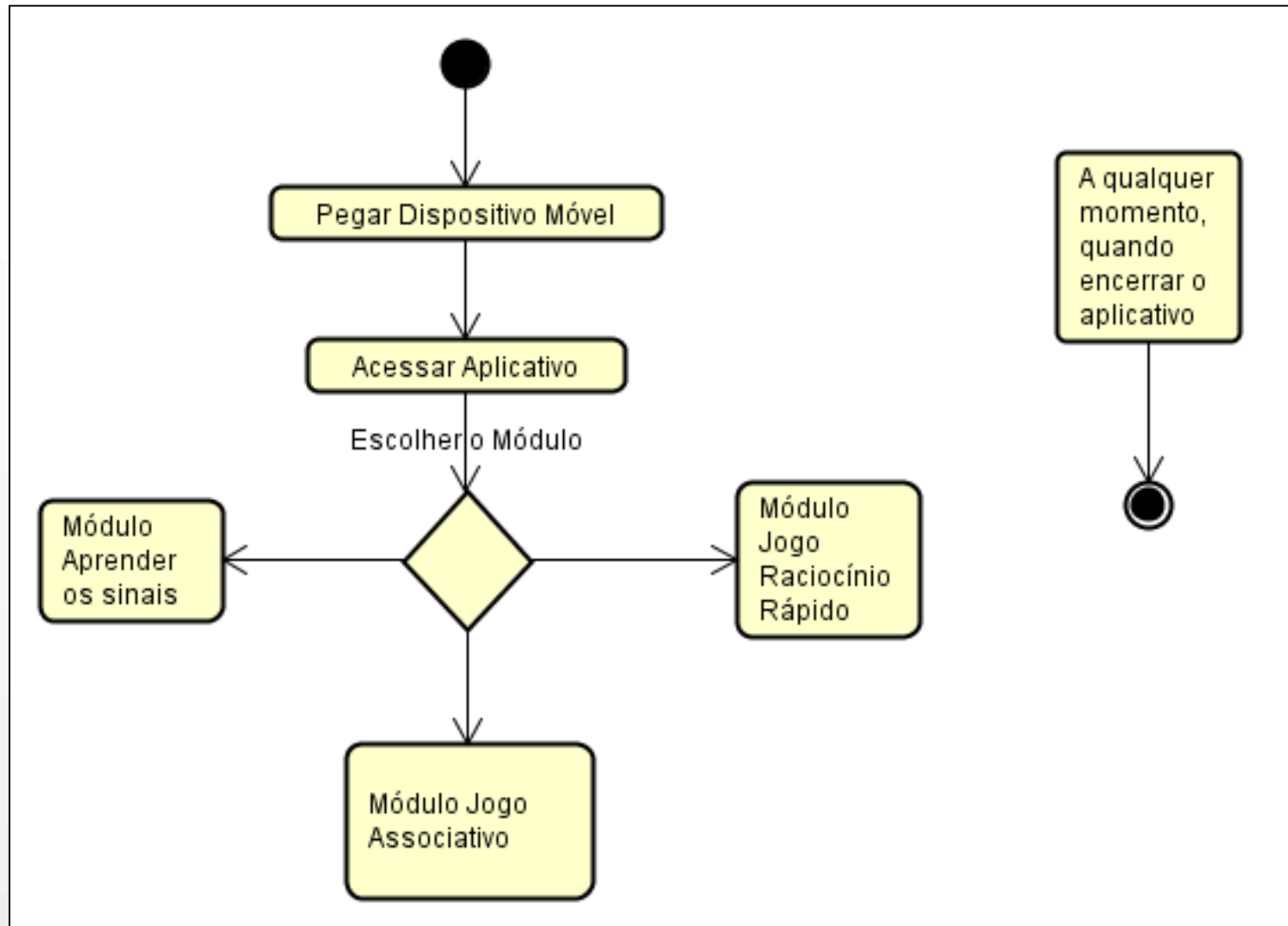
- Start() : void
- Update() : void
- + BackButton() : void
- + LoadGameARScene() : void
- + LoadLearnSignalScene() : void
- + LoadGameVRScene() : void
- LoadScene(scene : string) : IEnumerator



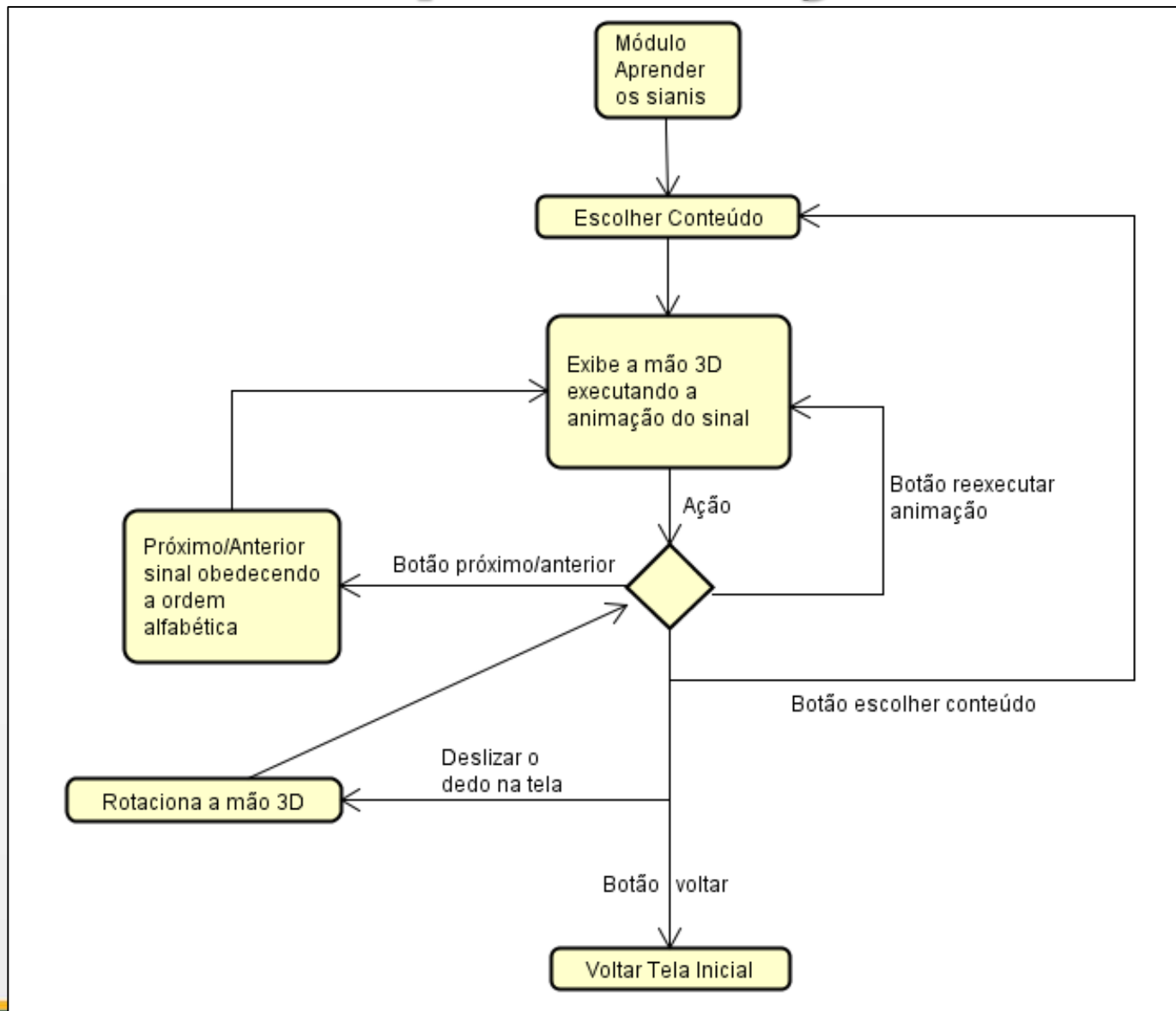
# Especificação



# Especificação



# Especificação

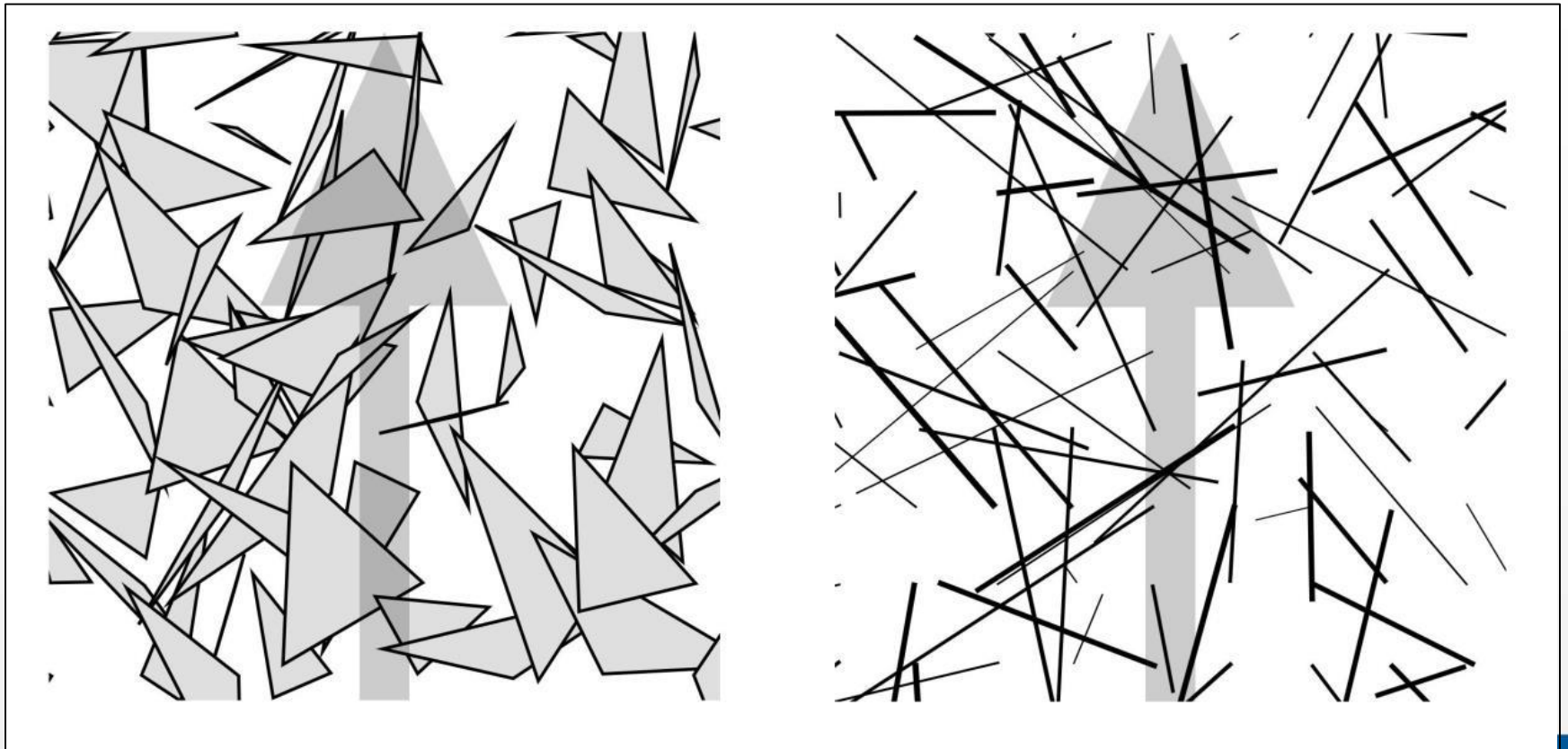


# Implementação

- Ferramentas utilizadas
  - Unity 3D
  - SDK Vuforia
  - SDK Google VR
  - Augmented reality marker generator
  - Photoshop
  - CadNav
  - Blender

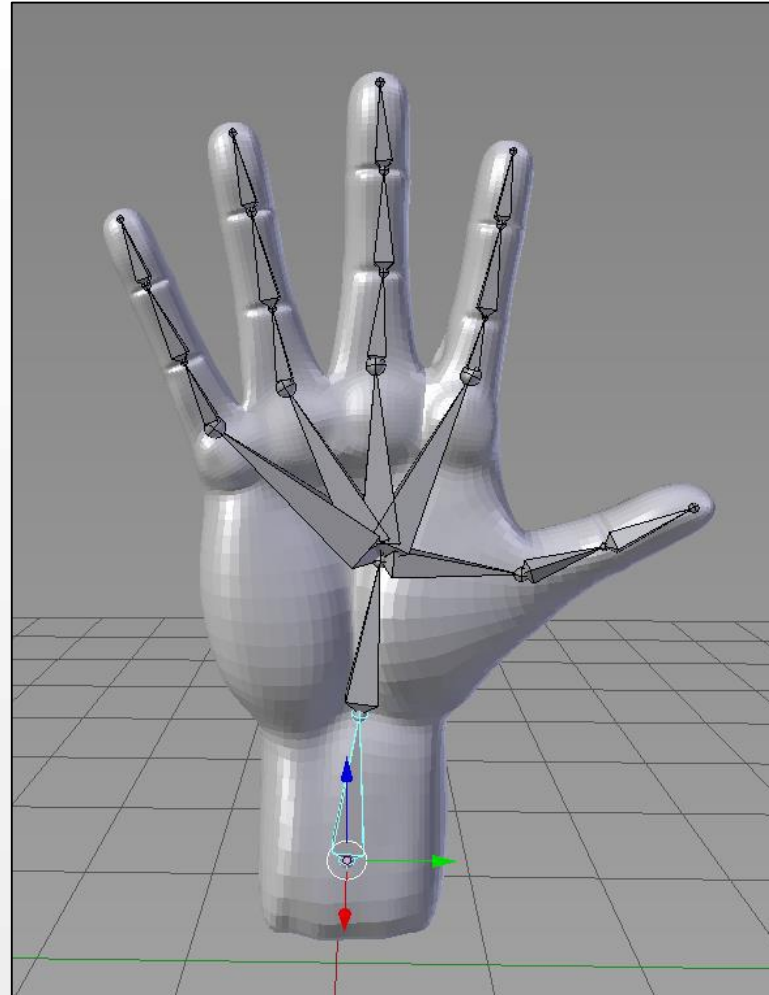
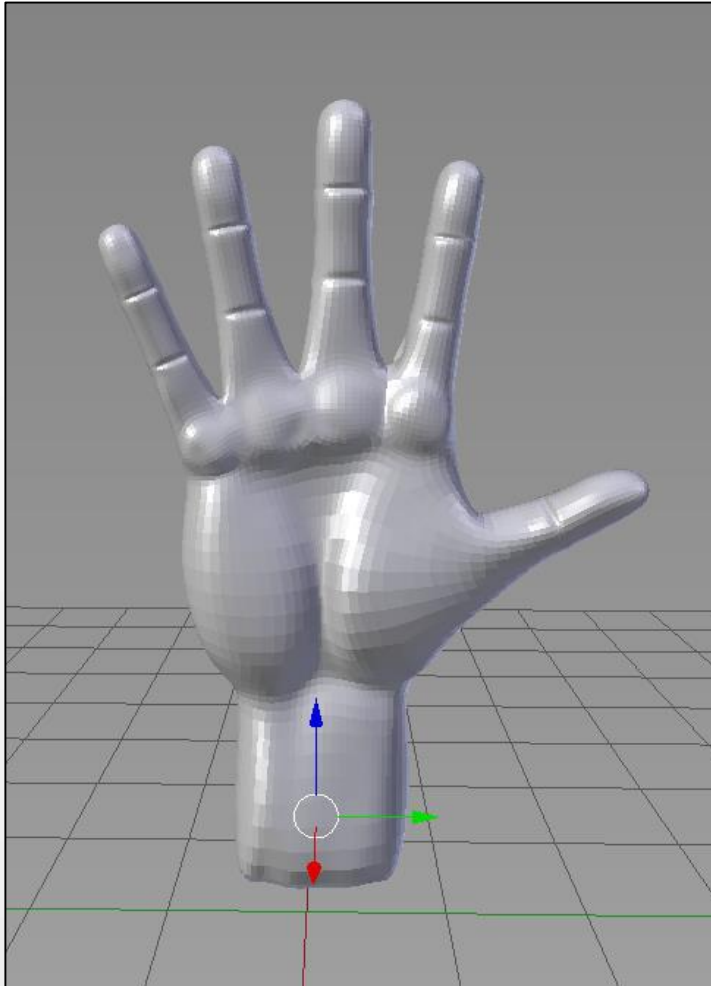
# Implementação

- Marcadores



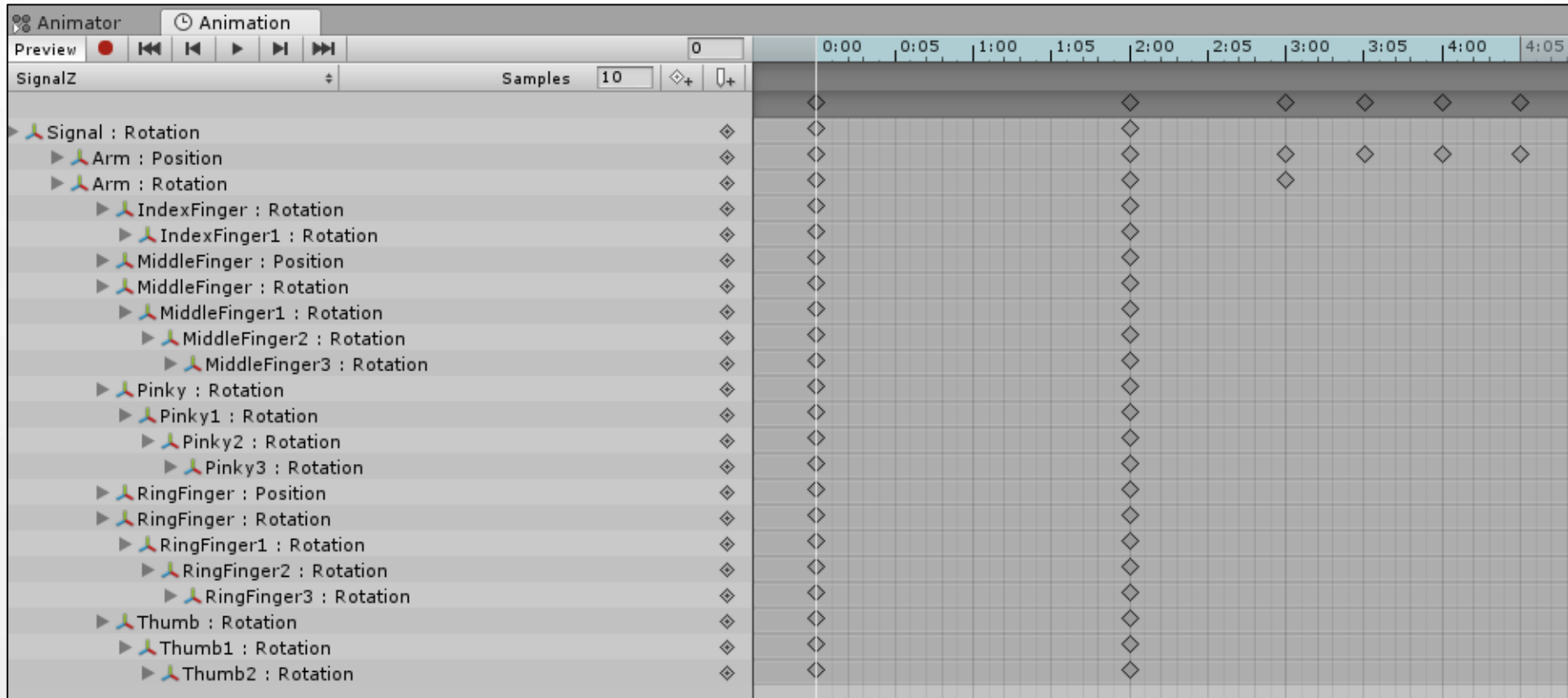
# Implementação

- Mão 3D



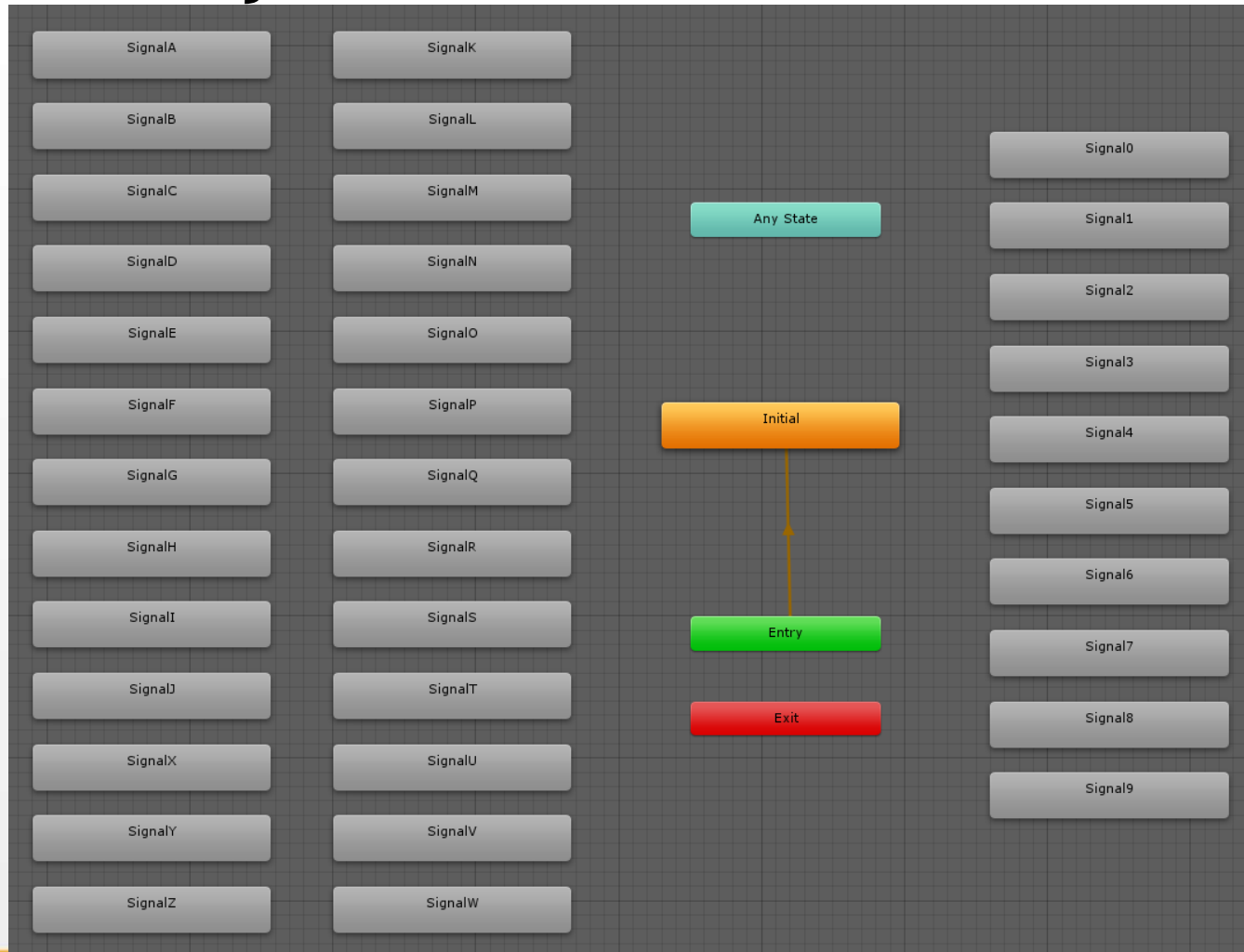
# Implementação

- Animações dos sinais



# Implementação

- Animações dos sinais





# Implementação

## ***Aprendendo Libras***



**Aprender os Sinais**

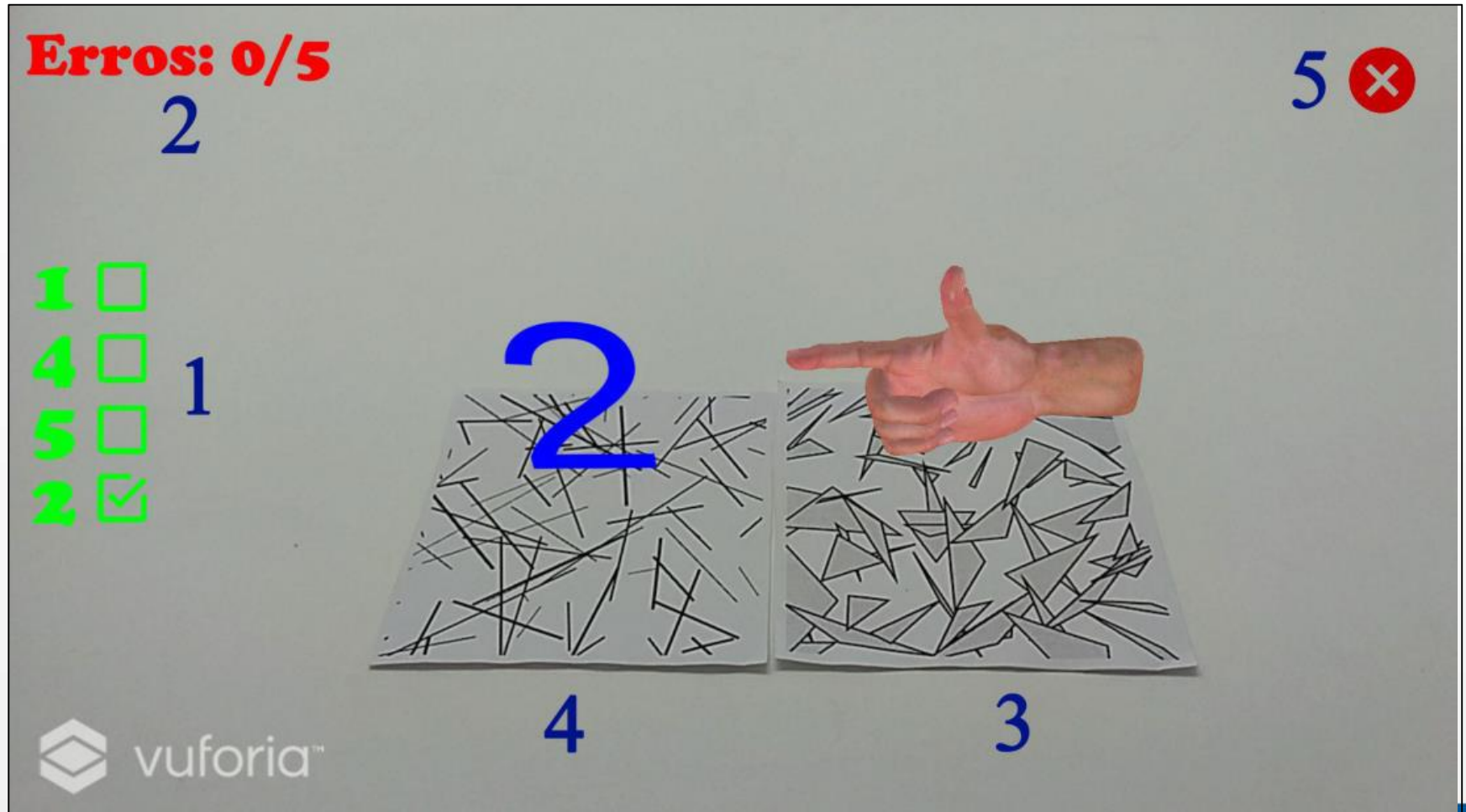


**Jogo Associativo**

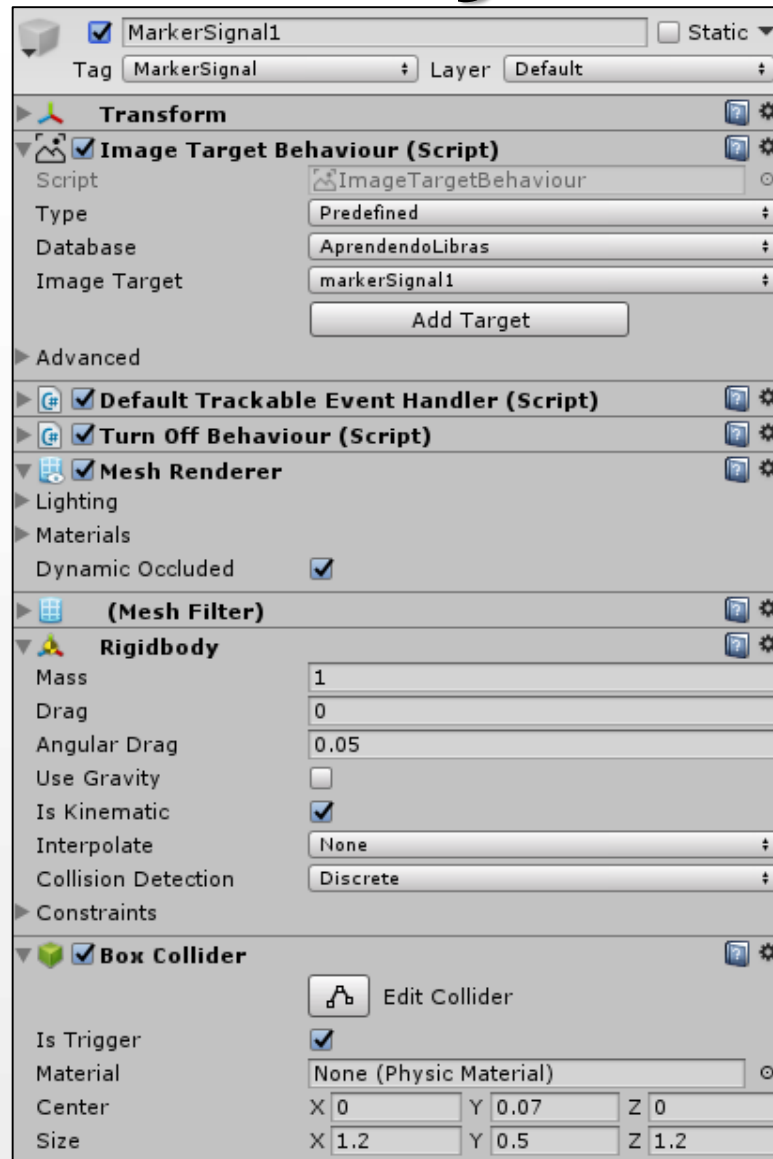
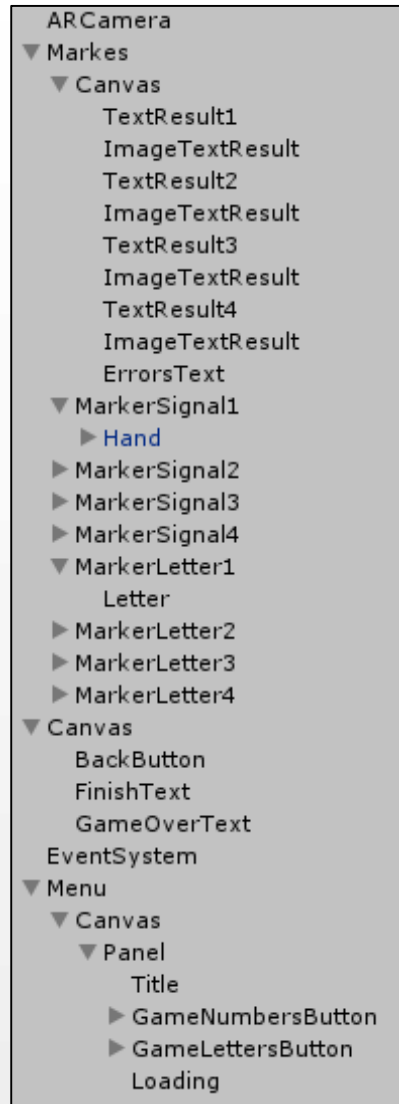


**Jogo de Raciocínio  
Rápido**

# Implementação



# Implementação



# Implementação

```
...  
void CreateMarkers()  
{  
    aleatorySignals = new string[] { "", "", "", "" };  
    int[] positionMarkers = { 0, 1, 2, 3 };  
  
    for (int i = 0; i < 4; i++)  
    {  
        string signal = currentSignals[Random  
            .Range(0, currentSignals.Length)];  
  
        while (System.Array.IndexOf(aleatorySignals, signal) != -1)  
        {  
            signal = currentSignals[Random  
                .Range(0, currentSignals.Length)];  
        }  
  
        aleatorySignals[i] = signal;  
        markersSignals[i].name += signal;  
        textResults[i].GetComponent<Text>().text = signal;  
        imagesTextResult[i].name += signal;  
  
        int position = positionMarkers[Random  
            .Range(0, positionMarkers.Length)];  
        positionMarkers = positionMarkers.ToList()  
            .Where(x => x != position).ToArray();  
        markersLetters[position].name += signal;  
        markersLetters[position]  
            .GetComponentInChildren<TextMesh>().text = signal;  
    }  
}  
...
```

# Implementação

```
void OnTriggerEnter(Collider other)
{
    if (gameObject.tag.Equals(other.tag))
    {
        return;
    }

    char signal = gameObject.name[gameObject.name.Length - 1];

    if (signal.Equals(other.gameObject
        .name[other.gameObject.name.Length - 1]))
    {
        text.GetComponent<TextMesh>().color = Color.blue;
        GameObject.Find("ImageTextResult" + signal)
            .GetComponent<Image>().sprite = spriteCheck;

        gameARScript.CheckFinishGame(signal.ToString());
    }
    else
    {
        text.GetComponent<TextMesh>().color = Color.red;

        gameARScript.CheckErrors();
    }
}
```

# Implementação

**1 Encontre o sinal: 1**



3 ○

**4 Tempo: 21 Erros: 0/2 5**



7



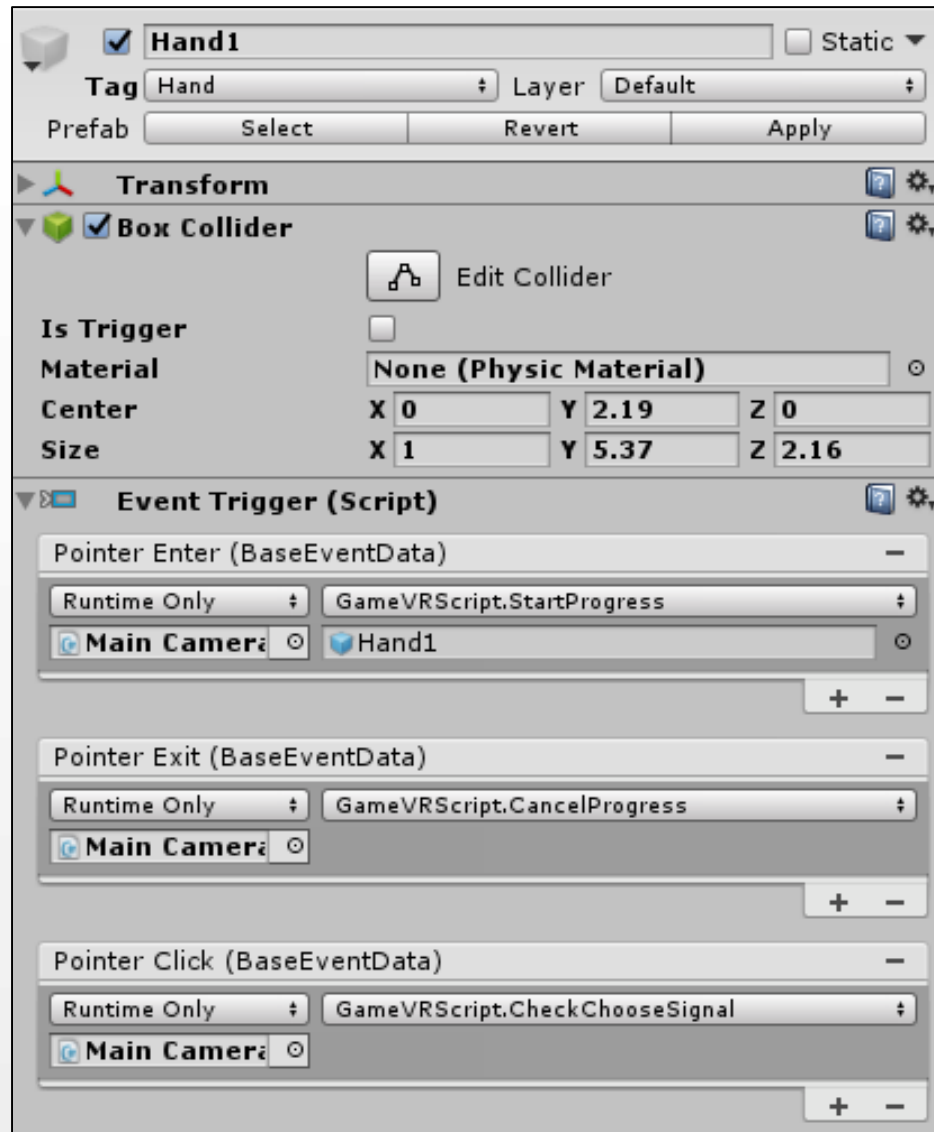
6



8

# Implementação

- ▼ Main Camera
  - ▶ ProgressRadialPlain
  - [GvrEventSystem](#)
  - [GvrReticlePointer](#)
- ▼ Menu
  - ▼ Canvas
    - ▼ Panel
      - Title
        - ▶ LearnNumbersButton
        - ▶ LearnLettersButton
- ▼ Canvas
  - BackButton
- ▼ GameVR
  - ▼ Canvas
    - FindSignalText
    - AleatorySignalText
    - TimerText
    - SecondText
    - ErrorText
    - NumberErrorsText
    - FinishText
    - GameOverText
    - ReloadSignalButton
    - MenuButton
  - ▶ [Hand1](#)
  - ▶ [Hand2](#)
  - ▶ [Hand3](#)
  - ▶ [Hand4](#)



# Implementação

```
public void StartProgress(GameObject objectClick)
{
    this.objectClick = objectClick;
    scriptProgress.IncrementValue(100);
}

public void CancelProgress()
{
    scriptProgress.Value = 0.01f;
    scriptProgress.TransitoryValue = 0;
}

public void ExecuteAction()
{
    CancelProgress();
    ExecuteEvents.Execute<IPointerClickHandler>(
        objectClick,
        new PointerEventData(EventSystem.current),
        ExecuteEvents.pointerClickHandler);
}
```



# Resultados e Discussões

- Experimento
  - ABADA
  - Turma com 5 alunos
  - Todos deficientes auditivos e alguns com deficiências psicológicas
  - Realizado uma demonstração do sistema
  - Testes acompanhados

# Resultados e Discussões

- Resultado do experimento
  - Um pouco de dificuldade no primeiro módulo
  - Um pouco de dificuldade no início do segundo módulo para associar os marcadores
  - Um pouco de dificuldade no início do terceiro módulo com o HMD
  - Alunos que usavam óculos
  - Padrões de botões
  - No final, todos ficaram usando sozinho o sistema

Características / Trabalhos correlatos	Ferramenta desenvolvida	Freire et al. (2015)	Santos e Souza et al. (2013)	Santos e Lobo et al. (2013)
Plataforma	Smartphone	Computador	Computador	Computador
Exibir os sinais em 3D	X	Sim	Parcialmente	Parcialmente
Realidade Aumentada	X	X	X	X
Realidade Virtual	X			
Associar os objetos virtuais	X	X	X	X
Consultar o sinal de um algarismo numérico ou letra	X			
Forma de associação dos marcadores	Sistema de colisão	Marcador fiducial	Marcador fiducial	Não informado
Ferramentas utilizadas	Unity, SDK Vuforia e Google VR	ARToolKit	ARToolKit	ARToolKit

# Conclusões e Sugestões

- Objetivos do sistema alcançados
- Ferramentas utilizadas foram adequadas
- Contribuição social
- Contribuição científica

# Extensões

- implementar no segundo módulo a possibilidade de o usuário usar o HMD para poder ficar com as duas mãos livres para manusear os marcadores
- criar um sistema de pontuação por usuário para o segundo e terceiro módulo do sistema
- implementar uma interface em Libras para o sistema
- melhorar o desempenho do segundo módulo de RA, para funcionar melhor em smartphones e tablets com hardware mais limitado

# **Apresentação Prática**