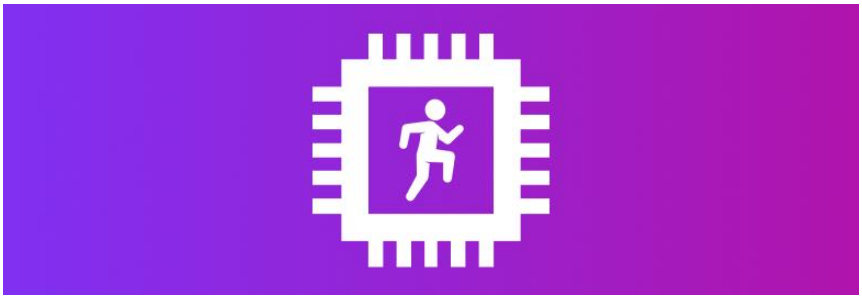
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA (ISEL)

DEPARTAMENTO DE ENGENHARIA ELETRÓNICA E DE TELECOMUNICAÇÕES E COMPUTADORES (DEETC)

LEIM

LICENCIATURA EM ENGENHARIA INFORMÁTICA E MULTIMÉDIA
UNIDADE CURRICULAR DE PROJETO

Aplicação Móvel para Aquisição de Dados Inerciais de Atletas de Alta Competição



Miguel Luís Girão Ginga (A46292)

Pedro Guilherme Goulão Alves (A46359)

Orientadores

Professor Carlos Gonçalves – Instituto Superior de Engenharia de Lisboa (ISEL)

Professor Pedro Teodoro – Escola Superior Náutica Infante D. Henrique (ENIDH)

Lisboa, setembro 2023

Resumo

Este documento consiste na descrição do processo de desenvolvimento do Projeto "**Aplicação Móvel para Aquisição de Dados Inerciais de Atletas de Alta Competição**" no âmbito da Unidade Curricular Projeto da Licenciatura em Engenharia Informática e Multimédia lecionada no Instituto Superior de Engenharia de Lisboa.

O projeto consiste na captura de movimentos através de sensores que incluem Unidades de Medição Inercial que estarão sob controlo de uma aplicação móvel e por fim reproduzir estes mesmos movimentos num ambiente de desenvolvimento 3D. Este projeto aborda uma vertente exploratória derivada do implementado pelo estudante de mestrado Gabriel Diaz que no seu caso desenvolveu os sensores com a finalidade principal de serem aplicados na natação.

Com este projeto visamos testar a viabilidade de um sistema mais completo, prático e acessível para todo o tipo de atletas em alternativa aos já conhecidos e dispendiosos métodos de captura de movimento ótico. Embora a captura de movimento inercial não seja em si um projeto inédito, sendo já desenvolvido por várias empresas, o intuito do trabalho desenvolvido é comprovar até que ponto podemos fazer uso de tecnologias mais acessíveis e flexíveis para obter os mesmos resultados.

A recolha de dados inerciais é algo já abordada e desenvolvida no projeto original sendo agora necessário recolher e aplicar estes dados num ambiente de desenvolvimento 3D. Para tal os sensores serão equipados com cartões de memória que irão guardar os dados dos treinos efetuados para que possam mais tarde ser utilizados. A aplicação móvel em si, fará a comunicação com os sensores para que estes saibam quando estão conectados e quando devem iniciar a escrita dos dados. De forma a generalizar um futuro uso deste projeto e dado o seu extenso nome, foi dada à aplicação um nome chave de acordo com o contexto da mesma: `SWEAT`.

Palavras-Chave: Aplicação Móvel, Atletas, Alta Competição, Código Aberto, Baixo Custo, Inertial Measurement Unit (IMU), Unity, Captura de Movimento Inercial, Bluetooth, React Native.

Índice

Índice de Figuras

Índice de Tabelas

Índice de Listagens

Lista de Acrónimos

BLE Bluetooth Low Energy

CSV Comma Separated Values

GATT Generic Attribute Value

IMU Inerial Measurement Unit

ISEL Instituto Superior de Engenharia de Lisboa

LED Light Emitting Diode

SD Secure Digital

1. Introdução

As tecnologias de captura de movimento continuam nos dias de hoje a ser desenvolvidas para que se possa tirar o máximo proveito das mesmas. Quando falamos deste tipo de tecnologias várias ideias vêm à mente como algo simples e comum que encontramos nas câmaras de vigilâncias para detetar se existe alguém ou algo a passar na cena que se está a capturar ou os fatos e equipamentos para capturar movimento que atualmente são usados com mais frequência nomeadamente no contexto de filmes e videojogos. Esta última vertente é a que mais próxima está do projeto que foi desenvolvido e cujo desenvolvimento será relatado ao longo deste documento. Para compreendermos melhor a captura de movimento foi realizado um breve estudo às técnicas mais prevalentes e usadas, do qual obtemos os seguintes resultados:

- Captura Ótica passiva: a técnica mais comum e flexível, usa marcadores retroreflectivos (refletem radiação, neste caso luz, de volta para o dispositivo de origem) e câmaras infravermelhas.
- Captura Ótica ativa: são usadas câmaras especificas para o rastreamento de movimento através de marcadores `LED` (*Light-Emiting Diode*).
- Captura Inércia: captura de movimentos que envolve sensores de unidade de medição inercial que por norma transmitem os dados para um dispositivo recetor. Não necessita do uso de câmaras ou marcadores.

Existem duas grandes diferenças relativas á captura de movimento Ótica e Inercial. A primeira consiste no custo, sendo que as soluções de captura ótica exigem material mais caro como ainda necessitam de ambientes próprios para que se possa tirar o maior proveito da tecnologia. A segunda diferença consiste na fiabilidade e qualidade dos resultados obtidos onde neste caso a tecnologia Ótica acaba por se destacar relativamente à inercial. Isto, no entanto, não implica que a captura de movimento inercial não seja adequada para a captura de movimentos de atletas, pelo contrário, o estudo efetuado apontado para que seja algo que está em crescimento a nível de desenvolvimento e mercado.

Tal como pode ser deduzido neste ponto do documento, o projeto desenvolvido fará uso da tecnologia de captura de movimento inercial. Este tipo de captura já conta com empresas dedicadas que oferecem produtos para a mesma, como iremos ver mais adiante, mas relativamente ao nível de equipamento e software que serão utilizados, encontram-se num patamar muito mais complexo e desenvolvido o que por sua vez resulta em valores de acesso mais elevados para um utilizador geral e fornecendo material que não pode ser readaptado para qualquer outra função. Teremos então como

um dos principais objetivos verificar a viabilidade deste tipo de sistema recorrendo a produtos mais acessíveis tanto a nível de disponibilidade e flexibilidade como ainda desenvolver software mais intuitivo e de fácil uso para o público-alvo em questão, que são atletas de alta competição.

Os microcontroladores utilizados já foram previamente trabalhados por um estudante de mestrado, Gabriel Diaz, num projeto denominado como "*Machine Learning of Swimming Styles*" (Diaz, 2022), também referido como `SWIMU`. O trabalho desenvolvido no "`SWIMU`", tal como podemos concluir pelo nome, teria como finalidade principal ser aplicado no contexto da natação, no entanto as bases utilizadas nestes, nomeadamente a recolha de dados inerciais através dos microcontroladores servem como raiz do nosso projeto.

O projeto atual, no entanto, possui novas vertentes que necessitam de ser exploradas, como o funcionamento e transmissão de dados para vários microcontroladores em simultâneo, uma aplicação móvel que possa fazer a gestão destes e o tratamento dos dados recolhidos para que possam ser usados de forma genérica por vários tipos de atletas.

A sincronização dos dados recolhidos será essencial, pois de forma a reproduzir os movimentos da forma mais viável e próxima do que foi efetuado na realidade por um atleta, não poderemos ter diferenças na quantidade de dados recolhidos e no espaço de tempo a que cada entrada destes é recolhida.

1.1 Contribuições

Como já referido anteriormente e ainda a ser abordado mais em profundidade no capítulo 2, existem já empresas que oferecem produtos que utilizam tecnologias de captura de movimento inercial. No entanto estas ofertas são de acesso difícil em diversos campos, não existindo uma alternativa mais básica para o público geral.

Aqui é onde entra a elaboração deste projeto como sendo a alternativa acessível, sendo feito usando software *open-source* e hardware, não só de valores mais acessíveis, facilmente readaptável a diversas situações. Como tal, de seguida apresentamos as contribuições desenvolvidas ao longo do projeto:

- Desenvolvimento de um software para o microcontrolador *Seeed Studio* `XIAO nRF52840`, onde no qual está incluído um cartão de memória SD *Card* (*Secure Digital Card*) para gravar os dados inerciais relativos ao treino efetuado.
- Desenvolvimento de uma aplicação móvel que pode ser utilizada em sistemas Android e IOS na qual foi implementado um protocolo que permite a comunicação com os microcontroladores de modo a poder iniciar e terminar treinos.

- Desenvolvimento de scripts que permitem ler dados inerciais do microcontrolador diretamente de uma porta série e aplicar os mesmos num modelo 3D em UNITY para reproduzir os movimentos em tempo real.
- Desenvolvimento de scripts que permitem ler dados inerciais em formato CSV de ficheiros obtidos após a realização de treinos e aplicar os mesmos num modelo 3D em UNITY para reproduzir os movimentos realizados.

De referir ainda que no início do projeto foram fornecidos ficheiros pertinentes à recolha de dados inerciais através dos microcontroladores utilizados referentes ao trabalho realizado pelo Gabriel Diaz.

1.2 Estrutura do Relatório

A estrutura do relatório, não tendo em conta o resumo inicial, encontra-se organizada da seguinte forma.

O capítulo 1 é uma secção introdutória ao projeto desenvolvido, fala de forma pouco aprofundada sobre o objetivo e alguns conceitos bases do projeto.

O capítulo 2 incide sobre as tecnologias e produtos semelhantes a este projeto, falando de forma leve sobre algumas especificações técnicas do mesmo assim como custo de venda.

No capítulo 3 encontra-se especificado os requisitos do projeto, assim como casos de utilização e a estrutura geral da arquitetura do mesmo.

A fase de implementação encontra-se no capítulo 4, onde se aprofunda sobre a arquitetura estabelecida e a implementação técnica dos vários módulos e componentes do projeto.

O capítulo 5 mira descrever os testes e resultados obtidos do funcionamento geral do projeto, tendo em conta os objetivos estabelecidos.

Por fim no capítulo 6 é feita uma conclusão do trabalho desenvolvido durante o projeto em adição com possível trabalho futuro de continuação do mesmo.

2. Trabalho relacionado

Este capítulo aborda trabalhos e tecnologias semelhantes já existentes no mercado para o público geral. Como referido anteriormente no Capítulo 1, quando efetuado o estudo relativo ao tipo de tecnologias existentes para a captura de movimento foram também recolhidos dados relativos a produtos já disponíveis com tecnologias de captura de movimento inercial.

Embora existissem várias companhias a produzir fatos de captura de movimento inercial, nem todas disponibilizavam os valores dos seus produtos, pelo que iremos analisar os que o faziam, servindo desde já com uma boa base para posteriormente compararmos o sistema desenvolvido no âmbito deste documento.

Tabela 2.2.1: Produtos Relacionados

Produto	Custo	Pontos Chave
Rokoko Smartsuit Pro II (Rokoko, 2023)	1.995€	• Fato com sensores instalados • Tecnologia para reduzir a latência entre dispositivos • Configuração rápida com transmissão em tempo real
Nansense Motion Capture Suit (Nansense, 2023)	6.299€	• Fato com a inclusão de 50 sensores, incluindo captura de movimento das mãos • Diferentes configurações para diferentes casos de uso • Aplicação móvel para controlar um ambiente de captura de movimento
VDsuit Full motion capture suit (Virdynmocap, 2023)	2.191€	• Equipado com 27 sensores, incluindo captura de movimento das mãos • Comunicação dos dados através de um dongle dedicado ao invés de Bluetooth • Software próprio fornecido para capturar dados com a possibilidade de fazer transmissões para plataformas de streaming
Perception Neuron Studio Inertial System (NeuronMocap, 2023)	6.850€	• 18 Sensores incluidos • Inclusão de uma hub para rececionar os dados transmitidos pelos sensores • Utilização em tempo real • Fitas para o corpo onde se conectam os sensores

Como podemos observar nos produtos presentes na Tabela 2.2.1 , estamos na presença de produtos com um nível de complexidade elevada, onde muitos possuem software próprio, como fazem uso de sistemas de transmissão que ignoram por completamente o Bluetooth. Também de referir que ao contrário do que é pretendido para este projeto, todos estes produtos possuem a capacidade de reproduzir em tempo real os movimentos realizados para um ambiente de modelação 3D. Outro fator interessante, e aqui no âmbito do projeto é debatível para já a utilidade desta implementação, alguns dos produtos incluíam já rastreamento para os movimentos dos dedos enquanto outros vendiam luvas próprias com valores a rondar 1.500€. No contexto to projeto podemos ter em conta desportos como o lançamento do dardo, do peso e do disco, por exemplo, como desportos de alta competição onde o rastreamento dos dedos poderá eventualmente ser importante.

Esta análise foi feita por vários motivos, sendo um deles contextualizar que tipo de produtos atualmente existem que desempenham funções semelhantes ao sistema desenvolvido no projeto. Outro motivo passa por descrever a complexidade destes sistemas para que posteriormente possa ser feita uma comparação com a solução alcançada. E por fim, os custos e encargos que estes produtos apresentam, novamente para que adiante possamos tirar conclusões da viabilidade da nossa solução relativamente às soluções já desenvolvidas.

Não ficando em falta, há que referir como já previamente mencionado o trabalho de mestrado desenvolvido por Gabriel Diaz, “Machine Learning of Swimming Styles - SWIMU” **(Diaz, 2022)**, que embora tenha sido desenvolvido concretamente com o intuito de detetar estilos de natação, apresenta em si uma solução para a captura de dados de inércia provenientes dos mesmos microcontroladores utilizados no nosso projeto.

3. Modelo Proposto

Este capítulo oferece uma examinação do modelo visionado, elucidando a abordagem e solução escolhida. Começando, na secção 3.1, com uma visão geral dos requisitos necessários, aprofundando sobre os requisitos funcionais na secção 3.2 e não funcionais na secção 3.3, seguido do estabelecimento dos casos de utilização na secção 3.4. A arquitetura do projeto está especificada e explicada na secção 3.5. Já na secção 3.6 está explicado um cenário de utilização. Este capítulo termina com uma referência ao custo do desenvolvimento do projeto na secção 3.7.

3.1 Requisitos

De forma projetar uma solução para o desenvolvimento do projeto é necessário estabelecer os requisitos necessários para construir um sistema capaz de cumprir o objetivo. Nesta secção serão estabelecidos esses requisitos de forma a perceber a necessidade das escolhas tomadas durante a fase de implementação.

3.2 Requisitos Funcionais

Os requisitos funcionais do projeto indicam quais as funcionalidades que devem ser implementadas no sistema para que este consiga fornecer as funcionalidades pretendidas. Ao nível deste projeto estas podem dividir-se em três módulos principais: o **Módulo do Sensor** (Secção 3.2.1), o **Módulo da Aplicação Móvel** (Secção 3.2.2) e por último o **Módulo do `Unity`**(Secção 3.2.3).

3.2.1 Módulo do Sensor

O sensor é responsável por obter os dados inerciais do atleta e gravá-los, para isso deve ser capaz de:

1. **Calcular e obter dados inerciais:** O sensor deve obter os valores de *pitch*, *roll* e *yaw* de forma consistente, sem falha e uma frequência suficientemente elevada.
2. **Gravar dados em cartão de memória**: De forma que os dados possam ser utilizados posteriormente, o sensor deve conseguir guardá-los num cartão SD a comando da aplicação móvel.

3.2.2 Módulo da Aplicação Móvel

A aplicação móvel é responsável informar aos sensores quando começa e acaba o treino, de forma a cumprir essa função é necessário:

1. **Estabelecer ligação Bluetooth com os sensores:** De forma a enviar informação e comandos para os sensores a aplicação deve ser capaz de iniciar ligação e comunicação via BLE com os sensores.
2. **Notificar o utilizador:** O sistema deve apresentar notificações ao utilizador quando este conecta os seus sensores com sucesso.
3. **Enviar comandos de início e paragem de treino:** A aplicação é responsável por enviar comandos de início de treino e fim de treino. De forma a organizar a gravação das pastas e ficheiros no cartão de memória o comando deve incluir a data e hora do início do treino.

3.2.3 Módulo do Unity

Utilizando o `Unity` como o nosso ambiente de desenvolvimento 3D, o sistema deve ser capaz de reproduzir os movimentos utilizando os dados obtidos durante as sessões de treino. Para tornar isso possível tem que:

1. **Ler informação de ficheiros extraídos do cartão de memória:** De forma a animar o modelo 3D e reproduzir os treinos, o sistema deve ler a informação contida nos ficheiros guardados no cartão de memória.
2. **Reproduzir os movimentos do treino:** Utilizar os dados recolhidos dos ficheiros de forma a animar o modelo 3D e ser possível visualizar o treino praticado pelo atleta.

3.3 Requisitos Não Funcionais

Os requisitos não funcionais indicam qualidades e características que o sistema desenvolvido deve apresentar de modo a cumprir critérios de performance, usabilidade, estabilidade, entre outros. Como tal foram estabelecidos os seguintes requisitos:

Tabela 3.1: Requisitos Não Funcionais

Produto	Pontos Chave
Estabilidade	O sistema não deve apresentar erros quando o mesmo estiver em execução.
Compatibilidade	O sistema deve ser compatível com várias dispositivos e sistemas operativos.
Usabilidade	As interfaces desenvolvidas devem ser de fácil utilização para o utilizador.

Modularidade	O sistema deve estar composto de forma modular de modo que seja mais acessível realizar atualizações e correções ao mesmo.

3.4 Casos de Utilização

Nesta secção iremos abordar os casos de utilização, que descrevem uma sequência de eventos originada de um agente externo (utilizador) ao usar o sistema para completar uma tarefa. Como abordado na secção 3.2, existem vários módulos pelo qual o sistema é composto, sendo que para os casos de utilização apenas são relevantes o módulo da aplicação móvel e o módulo do `Unity`, que serão os únicos que o utilizador irá interagir diretamente.

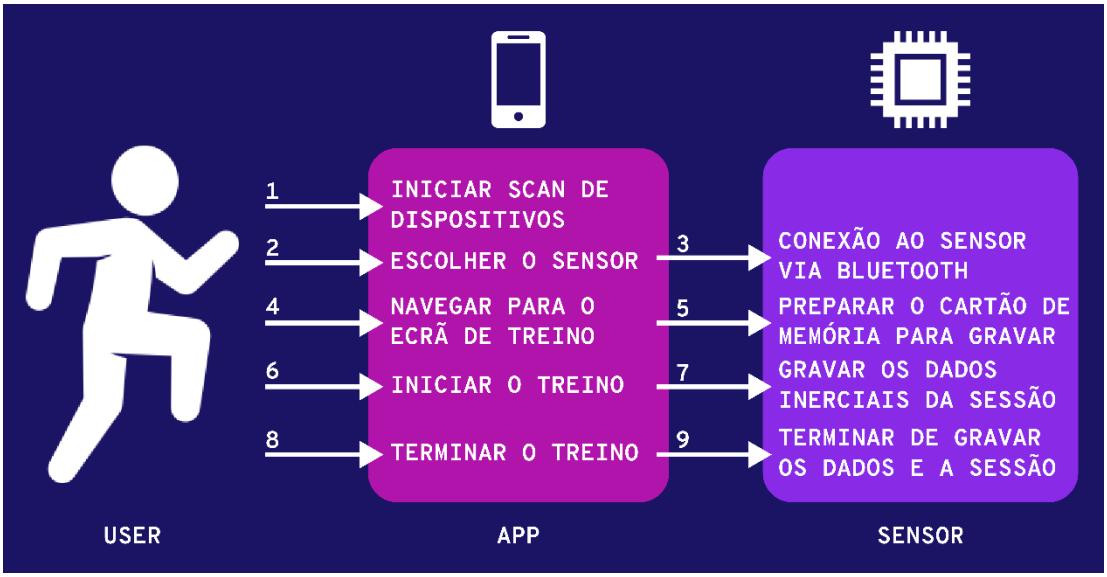


Figura 3.1: Caso de Utilização Aplicação-Sensor

A Figura 3.1 apresenta o caso de utilização referente ao módulo da aplicação móvel que estará em contato permanente com os sensores. O caso inicia-se quando o utilizador, neste caso um atleta, inicia a aplicação e seleciona a opção para efetuar scan dos dispositivos. Após efetuar o scan, este deve-se conectar ao sensor ou sensores que pretender utilizar.

Feita a conexão o utilizador pode navegar para o ecrã de treino, onde em paralelo o sensor prepara o cartão de memória para gravar os dados da sessão. O utilizador depois controla o início e fim do treino que consequentemente dita a escrita e término da sessão do lado do sensor.

O segundo caso de utilização faz referência à reprodução do treino efetuado em `Unity` usando os ficheiros gravados, com os dados inerciais, no cartão de memória associado aos sensores equipados.

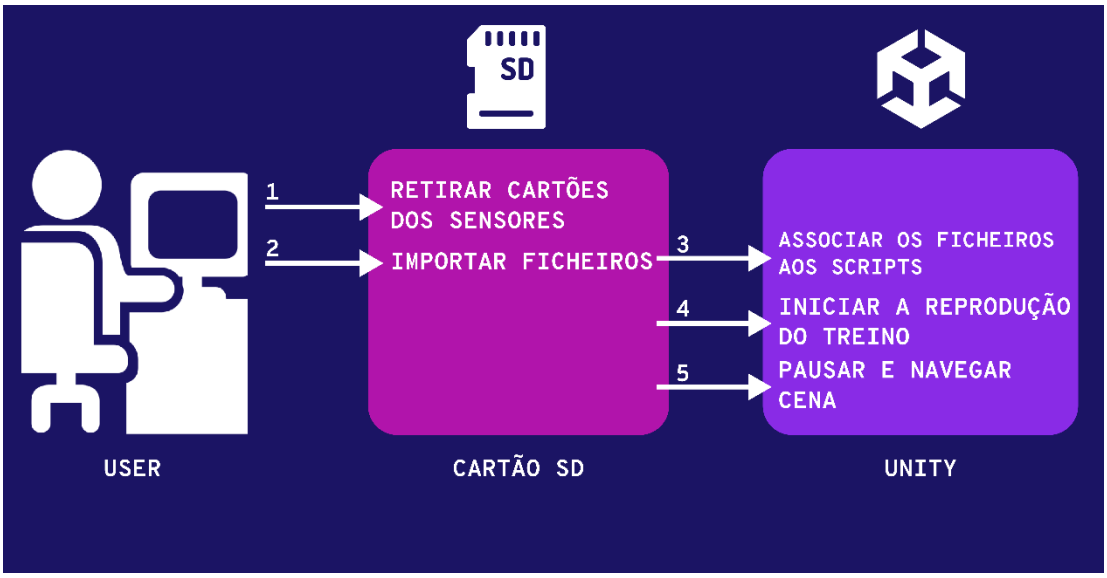


Figura 3.2: Caso de Utilização `CartãoSD-Unity`

Este caso, como possível observar na Figura 3.2, inicia-se quando o utilizador, novamente o atleta ou possivelmente um treinador, retira os cartões de memória dos sensores. De seguida devem ser importados os ficheiros referentes ao treino efetuado e associados os mesmos aos scripts desenvolvidos em `Unity`. Com estes passos efetuados é depois possível iniciar a reprodução do treino tal como pausar o mesmo e navegar a cena.

3.5 Arquitetura

Este capítulo incide sobre a arquitetura escolhida e tem como objetivo dar uma visão geral do projeto assim como justificar a implementação e escolhas tomadas.

A arquitetura envolve uma integração de sensores de medição de unidades inerciais, gravação de dados em cartão `SD`, transferência da informação para computador e subsequente utilização da informação recolhida para replicar a os movimentos do atleta num modelo 3D em `Unity`. Envolve ainda uma aplicação móvel que tem como objetivo comunicar com os sensores, através de *Bluetooth* de forma a iniciar e terminar a gravação dos treinos do atleta. Um esquema simplificado da arquitetura implementada pode ser observado na Figura 3.3.

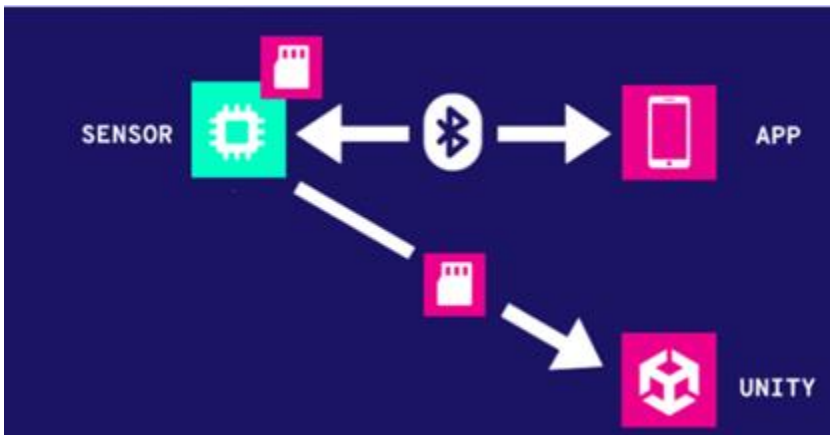


Figura 3.3: Arquitetura do Projeto

1. **Sensor:** O fundamento do projeto reside na utilização de sensores de medição de unidades inerciais. Estes tipos de sensores permitem captar várias formas de movimentos, incluindo *roll*, *pitch* e *yaw*. Estes sensores serão colocados em partes do corpo estratégicas do atleta de forma a capturar informação relativa aos movimentos.
2. **Bluetooth Low Energy**: Outro fator fundamental é a capacidade de ter uma aplicação móvel capaz de comunicar com os vários sensores através de *Bluetooth Low Energy*, esta comunicação permitirá o atleta iniciar os treinos e terminá-los de forma simples através uma aplicação móvel.
3. **Aquisição de informação e armazenamento:** Assim que o atleta inicia o treino através da aplicação, cada sensor distribuído no seu corpo deverá iniciar o armazenamento de dados em cartão de memória, este armazenamento é feito em cartão SD. Cada sensor obtém informação sobre *roll*, *pitch* e *yaw* a cada determinado momento e armazena-o em ficheiro no cartão de memória.
4. **Aplicação Móvel**: A aplicação móvel desenvolvida, capaz de funcionar em Android e IOS, permite ao atleta ligar-se a cada sensor, iniciar o treino e terminá-lo. A aplicação dispõe também de um tutorial de como equipar os sensores.
5. **Transferência de dados**: A transferência de dados dos cartões de memória para computador, onde mais tarde serão utilizados em `Unity`, é feita manualmente, isto significa que o atleta terá de remover o cartão do sensor, inseri-lo no computador e transferi-los manualmente para uma pasta dentro do ambiente de `Unity` onde serão utilizados para reproduzir o treino num ambiente virtual.
6. **Integração em `Unity`**: A informação adquirida por cada sensor é crucial, permitindo a reprodução do treino num modelo 3D em `Unity`. Após os ficheiros estarem inseridos no ambiente, os dados neles contidos são lidos linha a linha e processados de forma a aplicar rotações nos vários componentes do modelo 3D de forma a reproduzir os movimentos gravados durante a sessão de treino.

3.6 Cenário de Utilização

De forma a melhor entender os requisitos, a estrutura e implementação do sistema, complementando a informação apresentada na secção 3.4, de seguida apresentamos um cenário de utilização resumido do mesmo disponível na Figura 3.4.

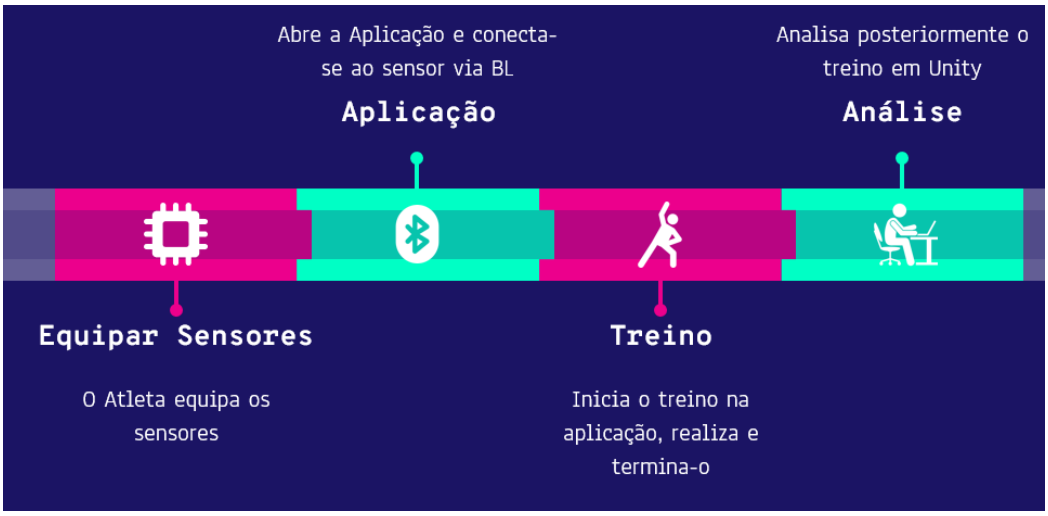


Figura 3.4: Cenário de Utilização

O atleta que deseja rever o seu treino começa por equipar os sensores no seu corpo, pode fazer uso do tutorial incluído na aplicação que demonstra os locais onde colocar os sensores de forma a obter os melhores dados possíveis. De seguida utiliza a aplicação para estabelecer ligação com os vários sensores distribuídos pelo seu corpo, ao finalizar este passo o atleta fica pronto para iniciar o seu treino.

Assim que deseja iniciar o treino, o atleta carrega no botão de iniciar na aplicação e efetua o treino, é importante que o atleta se encontre numa posição a direito com os braços estendidos ao seu lado, assim que o treino é iniciado os sensores começam a gravar dados no cartão de memória, antes de começar o treino o atleta deve esperar cerca de 3 segundos antes de se começar a movimentar. Quando o atleta deseja terminar o treino, utiliza a aplicação para esse efeito.

Por fim, o atleta extrai os cartões de memória do sensor, insere-os no computador onde passa os ficheiros correspondentes aos treinos que deseja analisar para o ambiente `Unity`, estando os ficheiros no ambiente, basta carregar no botão de *Play* e o treino irá ser reproduzido no ambiente virtual.

3.7 Custo do Projeto

Como já referido anteriormente a elaboração deste projeto visa a apresentar uma alternativa de baixo custo relativamente aos produtos já disponíveis e descritos no

capítulo 2. Para tal devemos complementar esta informação com o custo esperado deste projeto, apresentando de seguida um valor aproximado de um dispositivo completo:

Tabela 3.2: Custo de um dispositivo

Produto	Custo	Fonte
Seed Xiao nRF52840	14.59€	**(SeedStudio, 2023)**
3.7V 450mAh lithium battery	7.70€	**(Mauser, Bateria 3.7V 400mAh, 2023)**
Cartão de Memória SD Kingston 64GB	4.40€	**(PCDiga, 2023)**
Painel de Circuito PCB	2.52€	**(JLCPCB, 2023)**
Módulo Cartão SD	0.54€	**(AliExpress, 2023)**
Caixa do Sensor	0.19€	**(Mauser, Rolo de Filamento de Impressão 3D, 2023)**
Custo Final	29.94€	
Custo Final 12 Sensores	359.28€	

Os valores apresentados na Tabela 3.2 são aproximados pois podem existir alterações no valor final quando pedindo material em avulso, no entanto este encontra-se longe dos valores praticados no mercado atual observados no capítulo 2, deixando a porta aberta para que sejam implementados mais sensores e possivelmente componentes que complementem uma melhor funcionalidade do sistema, sem comprometer o objetivo que este seja uma alternativa de baixo custo relativamente ao restante mercado como ao mesmo tempo fazendo uso de materiais que podem ser facilmente readaptados para diferentes cenários.

4 Implementação

Este capítulo é dedicado à análise da implementação do projeto e descreve as opções tomadas no processo assim como os procedimentos técnicos.

A implementação do projeto divide-se em três componentes principais, a configuração do sensor utilizado para obter e registar os dados inerciais (Secção 4.1), e a criação de um programa em `Unity` que permita reproduzir os movimentos registados e testar a viabilidade do sistema (Secção 4.2) e o desenvolvimento de uma aplicação móvel de forma a comunicar com o sensor (Secção 4.3)).

4.1 Sensor

Um dos requisitos do projeto é ter sensores que possam ser utilizados no corpo dos atletas sem causar impacto no seu desempenho, para isso a escolha do sensor a ser utilizado tem de ter em conta o tamanho que deve ser o mais reduzido possível. Assim sendo foi escolhido o microcontrolador **Seeed Studio XIAO nRF52840** [Seeed Studio, 2023], incorporado com uma Unidade de Medição Inercial (IMU) de 6 eixos e capacidades de conexão por Bluetooth. O microcontrolador também permite a adição de um módulo de leitura de cartões de memória pelo que cumpre todos os requisitos do projeto.

4.1.1 Unidade de Medição Inercial

Neste capítulo será falado sobre o papel da IMU no processo de obter dados que possibilitem reproduzir os movimentos dos atletas em `Unity`.

Uma Unidade de Medição Inercial é um componente crucial que representa um papel vital na medição da orientação, velocidade e força gravitacional que agem sobre um objeto num espaço tridimensional. `IMUs` são projetados para providenciar informação precisa em tempo real sobre o movimento de um certo objeto.

Um IMU consiste de múltiplos sensores, tipicamente incluindo acelerómetros, giroscópios e por vezes magnetómetros, trabalhando todos em conjunto para dar informação relacionada com movimento.

Este tipo de sensores é geralmente utilizado em aviação, navegação, robótica, realidade virtual, entre outros. Neste conjunto de aplicações destaca-se para o nosso projeto as funcionalidades no mundo do desporto e fitness onde já existem aplicações que permitem registar números de passos, distância percorrida e mesmo exercícios realizados.

A informação obtida através destes sensores irá ser utilizada para calcular os valores de *roll*, *pitch* e *yaw*. Estes três valores representam a rotação de um determinado objeto nos três eixos. Associando sensores às articulações do corpo do atleta é possível obter os valores dos ângulos de cada articulação e, em teoria, será possível reproduzir os movimentos num ambiente virtual, permitindo o atleta analisar em detalhe e em 360 graus o seu treino.

Para melhor entender o conceito a Figura 4.1 dá-nos uma perspetiva de como aplicar o *roll*, *pitch e yaw* para reproduzir os movimentos do atleta. Caso o atleta rode a sua cabeça para a esquerda ou direita, essa rotação seria detetada pelo *yaw*, se o atleta inclinar a cabeça para a esquerda ou direita, isto é, tentar tocar com a orelha num dos ombros por exemplo, este movimento é detetado pelo *roll* e por fim o movimento de inclinar a cabeça para a frente e para trás, movimento característico de alguém que responde que sim ou que concorda utilizando a cabeça, a rotação é detetada pelo *pitch*. A teoria é que utilizando estas três medida de forma combinada é possível descrever todos os movimentos da cabeça do atleta. Esta lógica seria então implementada às restantes partes do corpo.

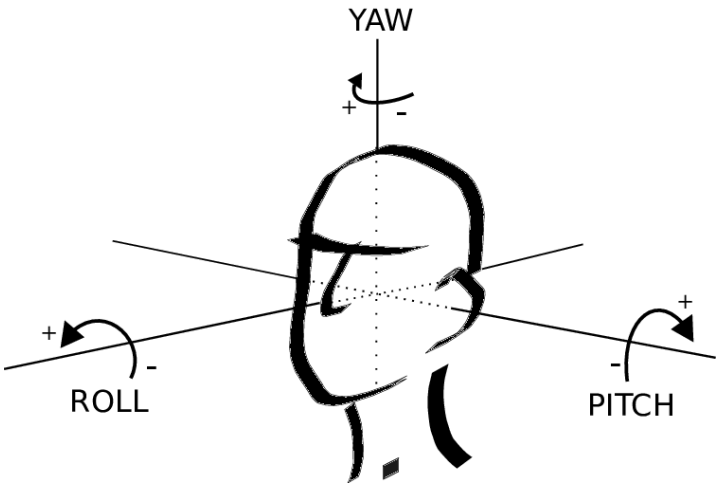


Figura 4.1: Aplicação *yaw*, *pitch* e *roll* na cabeça do atleta

Os ângulos utilizados para estas três medidas estarão compreendidos entre os 0 e 360 graus.

4.1.2 Bluetooth Low Energy

Bluetooth Low Energy (`BLE`) é uma tecnologia de comunicação sem fios desenhada para permitir troca de informação de forma eficiente entre aparelhos a distâncias curtas. É uma variante do Bluetooth clássico, otimizado para aplicações que requerem pouca energia, tornando-o particularmente útil em aparelhos que trabalhem através de baterias e aplicações que envolvam trocas esporádicas de informação.

Neste projeto as trocas de informação entre o telemóvel e os sensores cumprem estes requisitos pelo esta é a tecnologia mais adequada.

Outra característica importante desta tecnologia é que está bem preparado para transmitir comandos com baixa latência e com um modelo de comunicação simples. Esta comunicação revolve à volta de *Generic Attribute Profile* (`GATT`). `GATT` define a estrutura da comunicação de dados entre aparelhos através do uso de atributos. Atributos podem representar informação como leitura de sensores, definições de configuração e outros comandos.

`BLE` utiliza características para representar elementos de informação que podem ser escritas, lidas ou notificadas. Características podem ser utilizadas para enviar comandos de um aparelho para outro, habilitando controlo e interação entre aparelhos. As características utilizadas serão mencionadas mais à frente.

4.1.3 Gravação dos Dados de Treino

Para mais tarde reproduzir os movimentos do atleta, os valores de *pitch, roll e yaw* têm de ser guardados à medida que são medidos pelo sensor. Para cumprir este requisito foram propostas duas soluções, transferir os dados por Bluetooth para a aplicação do telemóvel em tempo real, guardar os dados em cartão de memória ligado diretamente ao sensor ou ainda tirar partido da memória interna dos sensores para guardar os dados.

De forma a tomar uma decisão é necessário ter em consideração diversos aspetos:

- O tamanho da informação gerada pelo sensor na fase de gravação.
- A frequência de amostragem dos dados.
- A velocidade e estabilidade da comunicação *Bluetooth Low Energy*.
- A sincronização entre os vários sensores a enviar informação para a aplicação.

A frequência de gravação dos dados é o principal fator para a escolha dos cartões de memória como forma de armazenar dados relativos aos treinos ao invés da solução por Bluetooth. Para que seja possível reproduzir de forma clara os movimentos dos atletas é necessário gravar dados com uma frequência alta, infelizmente a transferência de dados em tempo real por Bluetooth iria impor uma séria limitação na frequência a que os dados poderiam ser guardados. Isto poderia não ser problemático para desportos menos velozes ou explosivos, mas acabaria por não ser uma solução que pudesse ser utilizada e adaptada a todos os tipos de desportos e movimentos, que é um dos principais objetivos deste projeto.

Adicionalmente iria surgir um problema de sincronização de todos os sensores a enviar informação a uma alta frequência para o telemóvel, em que todos os dados teriam de ser guardados de forma consistente e sem falhas. Este é um processo extremamente complexo.

Por fim e fazendo referência ao projeto, ***Swimmer's Training Log Application***, desenvolvido por Daniela Levezinho e Letícia Lucas (Levezinho & Lucas, 2023)

também no contexto desta Unidade Curricular e que partilha várias semelhanças com o projeto descrito neste documento, foi concluído que o tamanho da memória interna do sensor simplesmente não é suficiente para armazenar treinos longos e/ou múltiplos treinos.

Tendo em contas estes fatores foi escolhido utilizar o cartão de memória ligado diretamente ao sensor, utilizando o cartão é possível gravar os dados a uma frequência alta e como cada sensor seria responsável apenas por guardar os seus próprios dados não existem problemas de sincronização na comunicação entre os sensores e a aplicação móvel. Assim basta assegurar que todos os sensores recebem o comando de começar a gravar de forma síncrona e igualmente na altura de parar a gravação.

Esta escolhe representa mais à frente uma maior dificuldade na transferência dos dados do cartão de memória para o computador para que possam ser utilizados no `Unity`.

4.1.4 Implementação do sensor

Estabelecidos os principais conceitos relacionados com a arquitetura do sensor podemos passar à fase de implementação do código. Esta secção incide sobre o modo de funcionamento do sensor e os aspetos técnicos da sua implementação.

De forma a controlar o comportamento do sensor foi definida e implementada uma **máquina de estados,** capaz de alterar o comportamento e funcionalidade do sensor conforme os comandos recebidos da aplicação móvel, de acordo com a Figura 4.2.

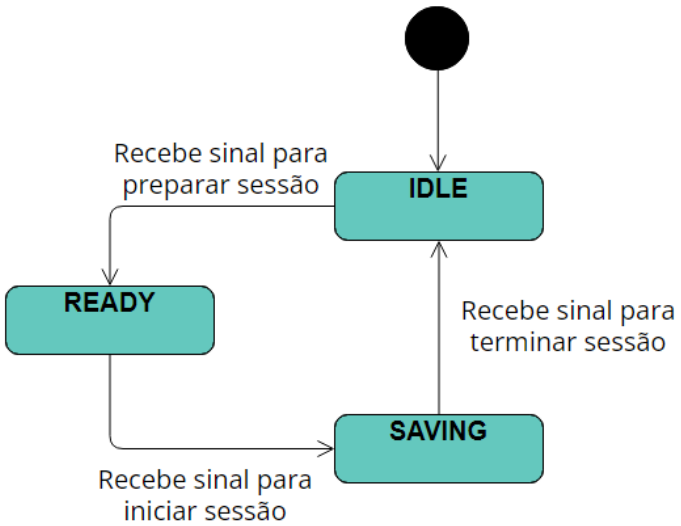


Figura 4.2: Diagram Máquina Estados Sensor

IDLE Este é o estado inicial do sensor, após ser ligado, realiza a calibração e inicializações necessárias como cartão de memória e Bluetooth e passa para este estado.

No estado `IDLE` o sensor limita-se a esperar por comandos vindos a aplicação móvel para alterar para o estado seguinte. Caso surja um comando correspondente à característica **`Current Time`** o sensor passa para o estado `READY`. Esta característica é utilizada no estado seguinte para criar as diretorias necessárias para guardar o ficheiro com a informação.

READY O estado `READY` representa o estado de preparação da sessão de treino, neste estado são criadas as diretorias necessárias para guardar o ficheiro e aberto o ficheiro para que possa começar-se a gravar no estado seguinte. Cada ficheiro de treino irá ser armazenado dentro de duas diretorias armazenados na diretoria principal `SWEAT/WORKOUTS`. A primeira diretoria corresponde à data no formato `DD/MM/YYYY`, todos os treinos realizados neste dia serão guardados nesta diretoria, a diretoria seguinte é a hora no formato HH/MM/SS, por fim o nome do ficheiro corresponde ao nome do sensor em questão. O tipo de ficheiro é txt.

SAVING O último estado corresponde ao comportamento de gravar os dados no ficheiro e permanecerá neste estado até que o utilizador envie o comando de terminar o treino. Enquanto se encontra neste estado o sensor vai escrevendo no ficheiro as medidas de *roll*, *pitch*, *yaw*, que vai obtendo, calculadas de forma a estarem compreendidas entre 0 e 360. Em cada instância de medição o sensor escreve numa linha única dos ficheiros os valores no formato "time;roll;pitch;yaw" como se pode observar na Figura 4.3, este processo procede até que o atleta deseje finalizar o treino. Ao finalizar passa para o estado IDLE.

```
149527;180;180;0
149606;161;149;342
149613;160;152;342
149622;159;154;342
149627;158;156;342
149635;157;158;342
149643;157;160;342
149652;156;162;342
149661;155;164;342
149669;154;166;342
149678;153;168;342
149682;153;170;342
149690;152;172;342
149699;151;173;342
149706;150;175;342
149716;150;177;342
149720;149;179;342
```

Figura 4.3: Exemplo Gravação Dados

4.2 Aplicação em Unity

Esta secção incide sobre a aplicação desenvolvida em Unity para testar e tentar reproduzir os movimentos do atleta num ambiente virtual.

Antes de passar à implementação e aspetos técnicos da mesma é importante explicar a escolha da plataforma Unity como ferramenta para tentar reproduzir os movimentos guardados pelo atleta. A escolha da plataforma apropriada é crucial para o sucesso de qualquer projeto, no contexto de recrear movimentos através de dados de IMU, o Unity destaca-se como uma escolha ideal por várias razões. A flexibilidade inerente ao Unity permite a integração perfeita de dados do mundo real com ambientes virtuais dinâmicos. O amplo suporte a gráficos 3D, simulação de física e renderização em tempo real garante uma replicação de movimentos precisa e visualmente cativamente. Além disso, o Unity possui uma comunidade grande e ativa, proporcionando acesso a uma vasta quantidade de recursos, plugins e bibliotecas que aceleram o processo de desenvolvimento.

Por fim e provavelmente o fator mais importante é a experiência e conhecimentos adquiridos na utilização de Unity durante o curso, facilitando o processo de desenvolvimento e poupando tempo que teria de ser utilizado a aprender uma plataforma nova.

4.2.1 Modelo 3D

De forma a reproduzir movimentos do corpo humano é necessário um modelo do corpo humano que funcione de acordo com as suas características biomecânicas. Felizmente, a *Asset Store* do Unity contém diversos modelos 3D que cumprem com esses requisitos, não sendo necessário desenvolver um de raiz.

Foi escolhido o modelo **3D Character Dummy** desenvolvido por Kevin Iglesias (Inglesias, 2023) por ser um modelo que se adequa às necessidades deste projeto e não ter custo monetário associado ao mesmo. O modelo pode ser observado na Figura 4.4, composto pelos principais ossos e articulações do corpo humano onde serão aplicados os valores das rotações obtidos na fase de gravação do treino.

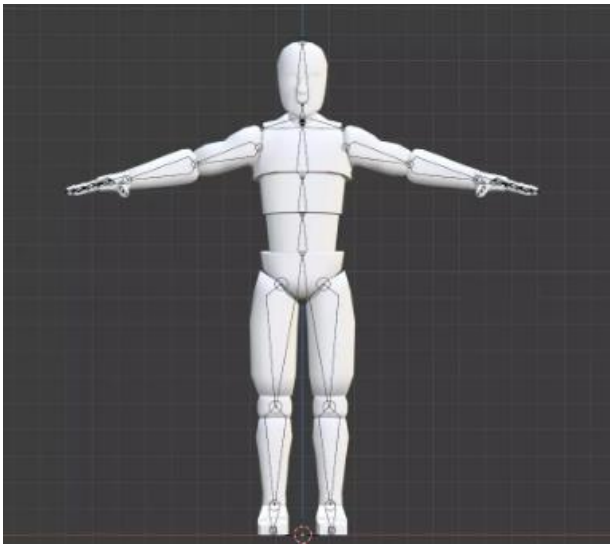


Figura 4.4: Modelo Humano 3D

4.2.2Implementação `Unity`

O desenvolvimento da aplicação `Unity` necessária para cumprir o objetivo do projeto foi realizado através de várias etapas.

No `Unity` cada item inserido no ambiente chama-se `game object`, estes objetos têm propriedades como posição, rotação e escala. Destas três a rotação é a que define os ângulos a que cada objeto se encontra no ambiente, assim sendo, é esta propriedade que permitirá usar os dados obtidos durante a gravação e aplicá-los de forma a reproduzir os movimentos.

Cada membro do modelo é considerado um `game object` no `Unity`, a cada objeto é possível associar um script, este script pode ser usado para modificar ou definir o comportamento de um `game object`, isto inclui afetar as suas propriedades de rotação. Assim cada membro e parte do corpo terá um script associado a si que irá ler do ficheiro onde foram guardados os dados registados pelo sensor correspondente. Lendo destes ficheiros em simultâneo com todas as partes do corpo torna possível ir alterando a rotação das várias partes do corpo do modelo à medida que os valores vão sendo lidos, e, se tudo correr bem reproduzir os movimentos do atleta.

Inicialmente começou-se por explorar o modo de funcionamento das rotações do motor de `Unity`, o objetivo inicial era aplicar diretamente os valores de *roll*, *pitch* e *yaw*, às várias partes do modelo 3D, isto é colocar no valor da rotação x o valor do *roll*, no valor do y o valor de *pitch* e no valor de z o valor de *yaw*. Esta solução acabou por demonstrar alguma inviabilidade devido à forma como o motor de jogo calcula e processa rotações, acabando por não dar uma representação correta das rotações no

mundo real. Isto acontece devido às rotações no `Unity` não sere calculados de forma independente nos 3 eixos.

Felizmente o `Unity` possui uma função que permite aplicar rotações de forma independente nos 3 eixos, a função `rotate(x,y,z)` permite efetuar rotações de forma a que seja possível usar os dados recolhidos para movimentar o modelo. Ao usar esta função é necessário calcular a diferença dos ângulos registados num certo momento e o seguinte e aplicar esta diferença na função. Assim os valore x, y e z utilizados na função são calculados da seguinte forma:

$$x = roll(n) - roll(n-1)$$

$$y = pitch(n) - pitch(n-1)$$

$$z = yaw(n) - yaw(n-1)$$

Onde n representa o valor registado num certo momento e n-1 no momento diretamente antes. Na figura 5 é observável a lógica utilizada, onde os ângulos θ correspondem à diferença calculada e x, y e z.

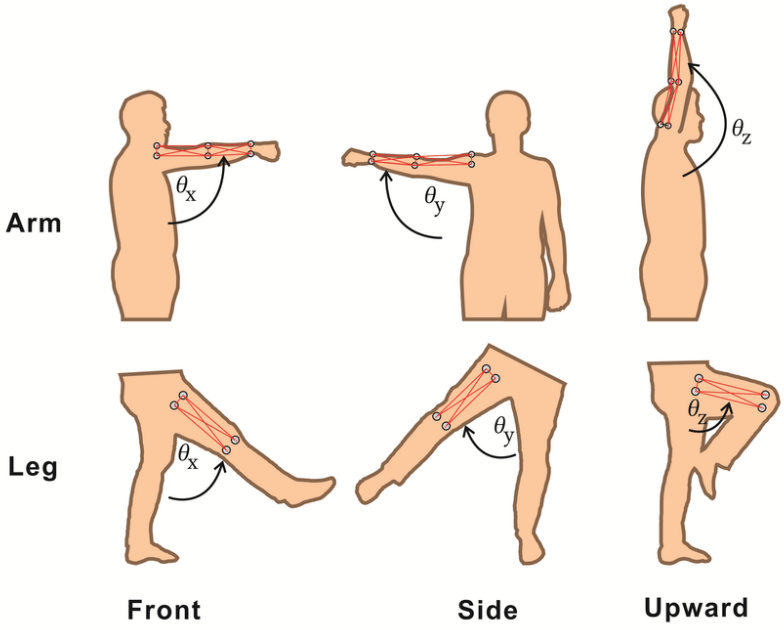


Figura 4.5: Cálculo dos ângulos de rotação

Aplicando esta lógica a todos as articulações é possível descrever os movimentos do atleta com bastante precisão. Explicado a lógica utilizada na implementação pode avançar-se para o código dos scripts implementados.

O script desenvolvido é relativamente simples e semelhante para cada parte do corpo, na função de `start()`, que corre assim que o programa inicia é aberto o ficheiro da parte do corpo correspondente e são guardadas todas as linhas num `array`. A Listagem 4.1 apresenta este comportamento.

Listagem 4.1: Função `start()`

```
 1. void Start(){
 2. fullPath = Path.Combine(Application.dataPath, fileName);
 3.
 4. if(File.exists(fullPath){
 5.       lines = File.ReadAllLines(fullPath);}
 6. else{
 7.       Debug.LogError("File not found: " + fullPath)}
 8. currentLineIndex = 0;
 9. isReading = true;
10. }
```

De forma a garantir que as rotações aplicadas a todas as partes do corpo acontecem ao mesmo ritmo, é utilizada a função `loop()` que é chamada pelo motor do jogo sempre ao mesmo ritmo independentemente do script ou objeto que a utiliza. Nesta função é onde é obtido os valores de *roll*, *pitch* e *yaw*, são efetuados os cálculos referidos anteriormente e é aplicada a transformação através da função `rotate(x,y,z)`. Assim é garantido que todos os objetos leem do ficheiro e são aplicadas as suas rotações com a mesma frequência, de forma a reproduzir o movimento preciso do atleta. Este processo é observável na Listagem 4.2.

Listagem 4.2: Função `update()`

```
 1. flaot velX = adjustedRoll - adjustedOldRoll;
 2. float velY = adjustedPitch - adjustedOldPitch;
 3. float velZ = adjustedYaw - adjustedOldYaw;
 4.
 5.
 6. if(Mathf.Abs(velX)>180f){
 7.     velX-=Mathf.Sign(VelX) *360}
 8. if(Mathf.Abs(velY)>180f){
 9.     velY-=Mathf.Sign(velY) *360}
10. if(Mathf.Abs(velZ)>180f){
11.     velZ-=Mathf.Sign(velZ) *360}
12.
13. transform.Rotate(velX, velY, velZ);
14.
15. oldYaw = yaw;
16. oldPitch = pitch;
17. oldRoll = roll;
```

4.3 Aplicação Móvel

Esta secção incide sob o desenvolvimento efetuado para a produção da aplicação móvel. A aplicação irá permitir a comunicação com os sensores através de `BLE`, implementando um protocolo com essa finalidade. As funções da aplicação serão a conexão de dispositivos via `BLE` e o envio de comandos para os mesmos. Contará ainda

com um breve menu referente à colocação dos sensores no corpo para facilitar o processo de utilização ao utilizador.

4.3.1Ambiente de Desenvolvimento

Na licenciatura na qual se enquadra esta Unidade Curricular, abordamos a criação de aplicações móveis via o ambiente de desenvolvimento `Android Studio`. No entanto, este ambiente limita a aplicação a sistemas `Android`, não proporcionando soluções para `IOS`. Aqui entra o ambiente de desenvolvimento escolhido para este projeto, `React Native` (ReactNative, 2021) , que proporciona a possibilidade de produzir aplicações que simultaneamente são compatíveis com ambos os sistemas.

Este ambiente suporta várias plataformas de desenvolvimento, cada uma com as suas nuances e especificações e inicialmente optou-se por desenvolver a aplicação na plataforma `Expo`, sendo algo mais acessível a nível de aprendizagem e bibliotecas disponíveis. No entanto, esta plataforma apresenta uma incompatibilidade com as bibliotecas de `BLE` disponíveis, inviabilizando a mesma. Consequentemente optou-se por utilizar a plataforma tradicional de `React Native`, denominada de `React Native CLI`.

O ambiente `React Native`, suporta então JavaScript (com algumas novas inclusões e adaptações) e uma linguagem própria `TextScript`, sendo que no desenvolvimento do projeto optou-se pela primeira opção com a qual já estamos familiarizados no contexto do conteúdo lecionado na licenciatura. Para efeitos de produção, foi utilizado o editor de texto `Visual Studio Code` devido à sua excelente organização e disponibilidade de *plugins* que facilitam a escrita de código correta.

Relativamente ao grafismo e design apresentado, recorreu-se à plataforma `Figma` para produzir todo este conteúdo.

4.3.2Implementação da Aplicação Móvel

A aplicação móvel passou por várias iterações até chegar ao seu estado final, sendo estas ao nível de funcionalidades, interface de utilizador (`UI`) e design da mesma. Na Figura 4.6 podemos observar um *mockup* do conceito inicial da aplicação que serviu como base para o restante desenvolvimento. Este conceito era composto por quatro ecrãs:

- *Home*: ecrã onde se conecta os sensores e se prepara o início do treino.
- *Start Workout*: ecrã onde se inicia o treino.
- *Workout In Progress*: ecrã onde mostra o tempo de progresso e se pode terminar o treino.

- *Workout Finished*: ecrã onde mostra o tempo final do treino e permite dar início a um novo.

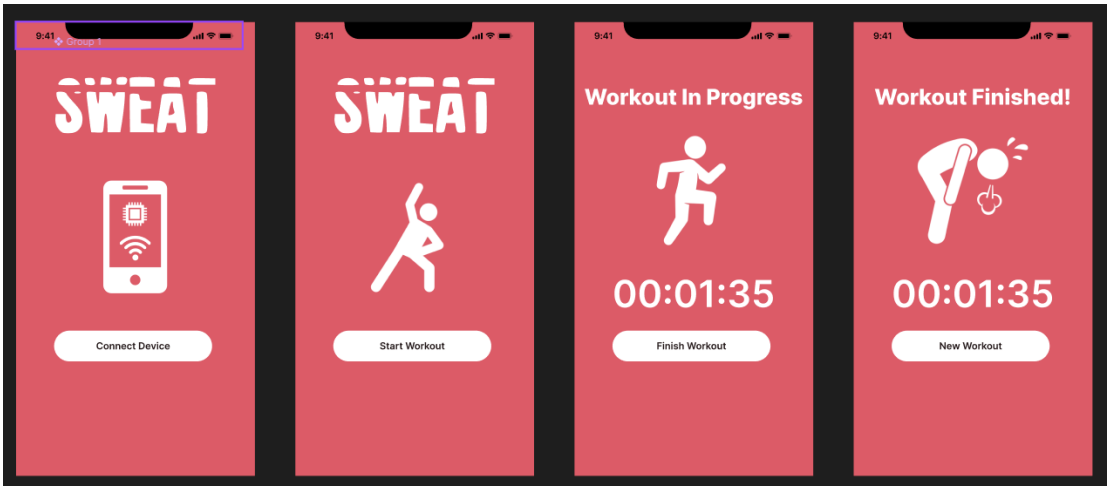


Figura 4.6: Mockup do Conceito Inicial da Aplicação

Como conceito inicial, este satisfazia as necessidades a nível funcional presentes no momento correspondente. No entanto à medida que se desenvolveu o projeto e se elevou o nível de complexidade foi necessário rever as funcionalidades presentes neste para se ajustarem à realidade atual. Não só isso, mas o número de ecrãs para a função que cada um exercia era desnecessária, sendo possível compilar esta informação de uma forma mais compacta e fácil de navegar. Para tal foi feito uso dos elementos e bibliotecas disponibilizadas em `React Native`.

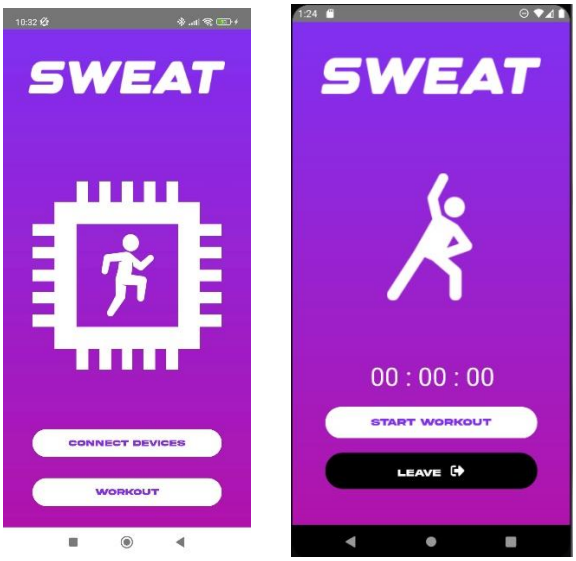
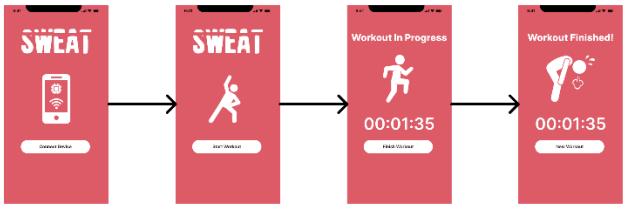


Figura 4.7: Capturas de Ecrã da Aplicação Final

Na Figura 4.7 temos agora os dois únicos ecrãs pelos quais é composta a aplicação: *Home* e *Workout*. Desde já foi diminuído o número de ecrãs, consequentemente apresentando ao utilizador uma navegação mais sucinta e interativa como iremos ver

adiante quando descrevermos cada um destes. No entanto, de forma a complementar esta mudança e para se entender concretamente as mudanças implementadas a nível de conteúdo e navegação, foram produzidos os seguintes diagramas.

Navegação Conceito Inicial



Navegação Aplicação Final

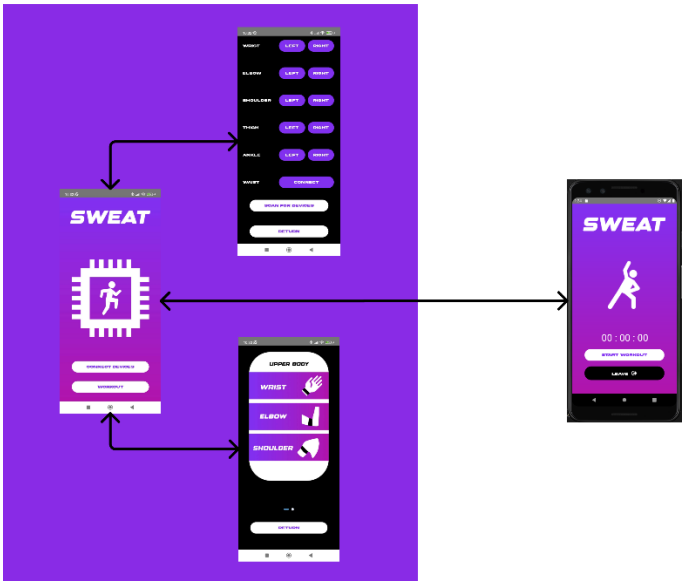


Figura 4.8: Diagrama de Navegação da Aplicação

Nos diagramas presentes na Figura 4.8 desde já é possível observar como foi removida a linearidade presente no conceito inicial, permitindo ao utilizador uma navegação mais interativa. De seguida, o número de ecrãs, não só foi reduzido para metade, como foram implementadas mais funcionalidades em cada ecrã, tornando também estes mais interativos do que simplesmente cada um possuir um botão para efetuar uma transição para o próximo.

Relativamente ao ecrã *Home*, este é composto por mais elementos do que aparenta à primeira vista. Os componentes que constituem este ecrã são do tipo `Modal` (de certa forma podemos assumir este componente como um compartimento que podemos abrir e fechar) onde a sua visibilidade é controlada pelos botões existentes. Portanto embora a Figura 4.8 apresente a ilusão que estamos na presença de quatro ecrãs, toda a zona delineada com o fundo roxo é o ecrã *Home* e os seus componentes.

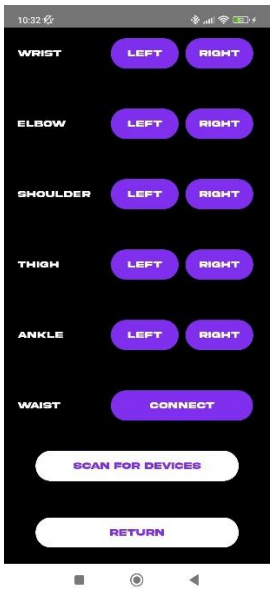


Figura 4.9: Componente BLEconnection

O primeiro componente, e este o mais complexo e crucial para o funcionamento da aplicação é o `BLEconnection`. Este componente, como apresentado na Figura 4.9, apresenta uma lista com botões para todos os sensores disponíveis, onde o utilizador pode escolher a quais destes se conectar, carregando no botão correspondente após efetuar um scan inicial. Ao efetuar o scan o utilizador recebe uma notificação no ecrã indicando que o scan foi realizado com sucesso, podendo depois conectar-se aos sensores que tiver disponíveis, novamente recebendo uma notificação se a conexão for estabelecida. No entanto para estabelecermos conexões via `BLE` é necessário importar e usar a biblioteca `react-native-ble-manager` (marcosinigaglia, 2023), que fornece todos os métodos necessários para as funcionalidades `BLE`, desde conexão de dispositivos a escrita e leitura dos mesmos.

Quando são conectados sensores, os `ID's` (endereço `MAC`) dos mesmos, são colocados numa lista que será passada posteriormente para o ecrã *Workout* de modo que quando o treino seja iniciado seja possível passar essa informação para todos os dispositivos.

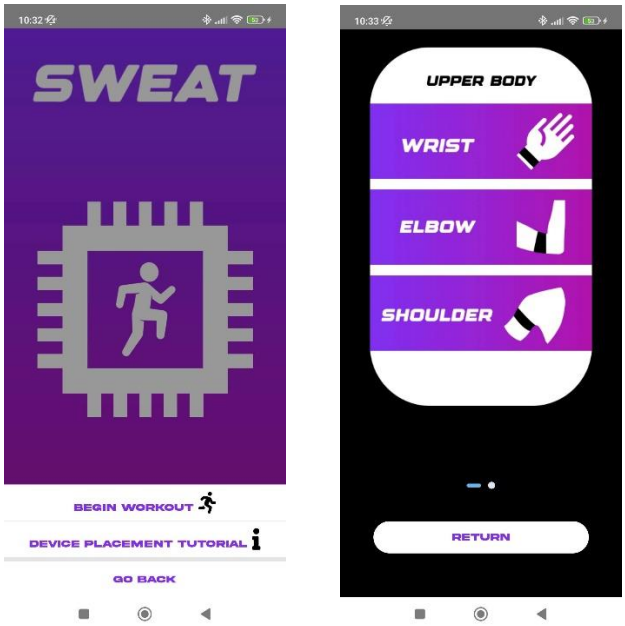


Figura 4.10: Exemplo do Tutorial de Colocação de Sensores

Quando pressionado o botão *Workout* no ecrã *Home*, é disponibilizado um pequeno menu que tem duas opções como se pode observar na primeira captura de ecrã presenta na Figura 4.10. A primeira opção irá enviar o utilizador para o ecrã *Workout* onde poderá dar início ao treino. Quando pressionada esta opção, é recolhida a lista de dispositivos conectados e são efetuadas várias operações com esta:

1. Verificar se existe pelo menos um dispositivo conectado para iniciar o treino. Caso esta condição não se verifique é mostrada uma notificação de erro.
2. A cada dispositivo conectado é escrito na característica `CurrentTime`, mencionada no capítulo 4.1.4, a data atual e o tempo corrente, sendo que isto irá provocar no sensor que este crie as diretorias necessárias no cartão de memória e prepare o ficheiro para escrever os dados relativo ao treino. O objetivo desta preparação é para que todos os sensores estejam prontos na mesma janela de tempo para escrever quando for dada a ordem para tal.
3. É enviada para o ecrã *Workout* a lista de dispositivos conectados.

A segunda opção deste menu abre um compartimento que contém imagens ilustrativas que guiam o utilizador para a colocação dos sensores no corpo, tendo como exemplo destas ilustrações a segunda captura de ecrã na Figura 4.10.

Por fim temos o ecrã *Workout*, o qual ilustrado na segunda captura de ecrã Figura 4.7, que possui dois botões e um cronómetro. O primeiro botão permite iniciar e terminar treinos, enquanto o segundo regressa o utilizador ao ecrã *Home*. A logística para iniciar um treino é a mesma já referida anteriormente, onde é escrito na característica `CurrentTime` dos sensores, mas neste caso a informação enviada não é pertinente sendo que apenas é pretendido dar um sinal ao sensor para que entre na próxima fase da sua máquina de estados. Aqui é essencial garantir que todos os

sensores começam a escrever no cartão de memória ao mesmo tempo de forma a termos dados consistentes que possam ser utilizados em `Unity` posteriormente. De modo a certificar isto, são usadas as funcionalidades `async`/`await` de JavaScript.

Estas funcionalidades permitem que um método seja declarado como `async`, o que significa que este irá retornar um objeto denominado por `promisse`. A palavra-chave `await`, que só pode ser usada dentro de uma função declarada como `async`, por sua vez indica a um bloco de código que este só deve ser executado quando o objeto `promisse` for devolvido. A implementação na aplicação passou exatamente por apenas dar ordem de escrita aos sensores, quando for devolvido o objeto `promisse` de todos os conectados, resultando em ficheiros de escrita finais com um número de dados registados consistentes.

5 Validação e Testes

Este capítulo incide sobre os testes e validação da solução implementada. Na secção 5.1 estão especificados os testes realizados no módulo do sensor. Os resultados obtidos na recolha e gravação dedos estão especificados na secção 5.2. De seguida, na secção 5.3, são apresentados os resultados e o funcionamento da aplicação móvel. Por fim, na secção 5.4 estão especificados os resultados obtidos na reprodução de movimentos em `Unity`.

5.1 Módulo Sensor

No módulo do sensor é importante testar alguns aspetos. Em primeiro lugar é necessário perceber se os dados de *roll*, *pitch* e *yaw* calculados são consistentes e representativos dos ângulos e movimentos detetados no sensor, para testar este aspeto foi elaborado um teste. Este teste consiste em rodar o sensor 180 graus (meia-volta) em cada um dos três eixos e analisar os dados obtidos de forma a averiguar se são consistentes.

Observando os gráficos abaixo (Figura 5.1, Figura 5.2, Figura 5.3) é possível ver que os valores de *roll*, *pitch* e *yaw* são consistentes com uma rotação de 180 graus no seu respetivo eixo, isto deixa-nos confiantes que os dados calculados pelos sensores são uma representação precisa dos movimentos de rotação dos mesmos.

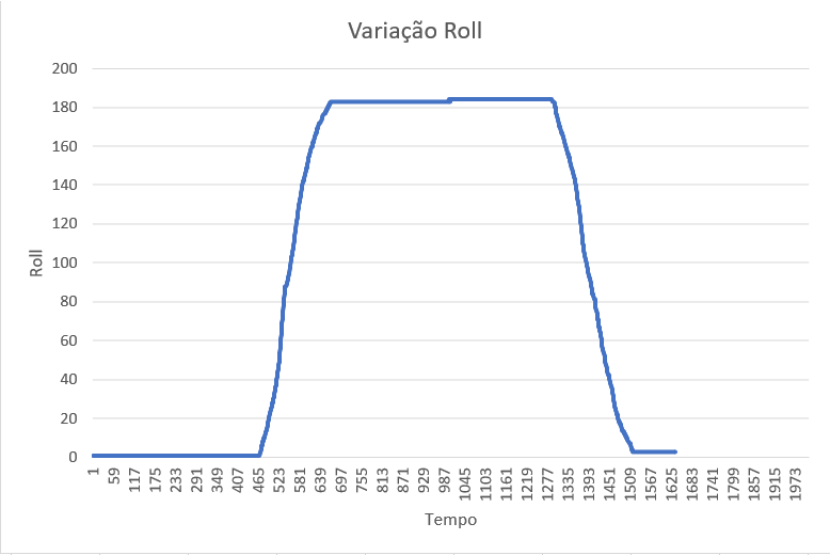


Figura 5.1: Gráfico Variação Roll

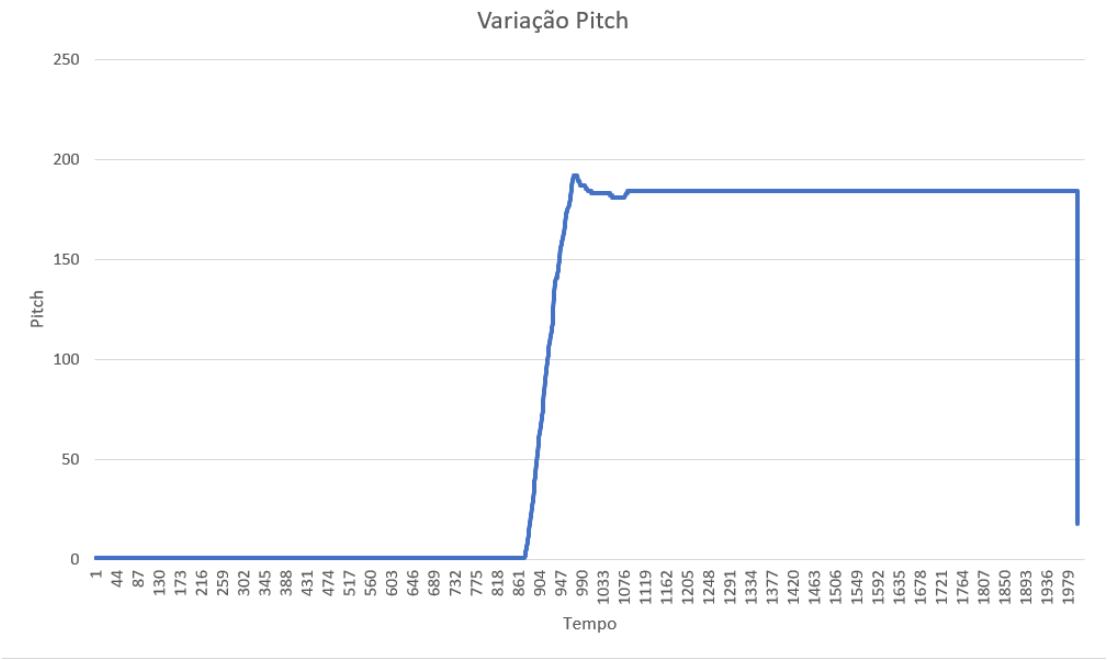


Figura 5.2: Gráfico Variação Pitch

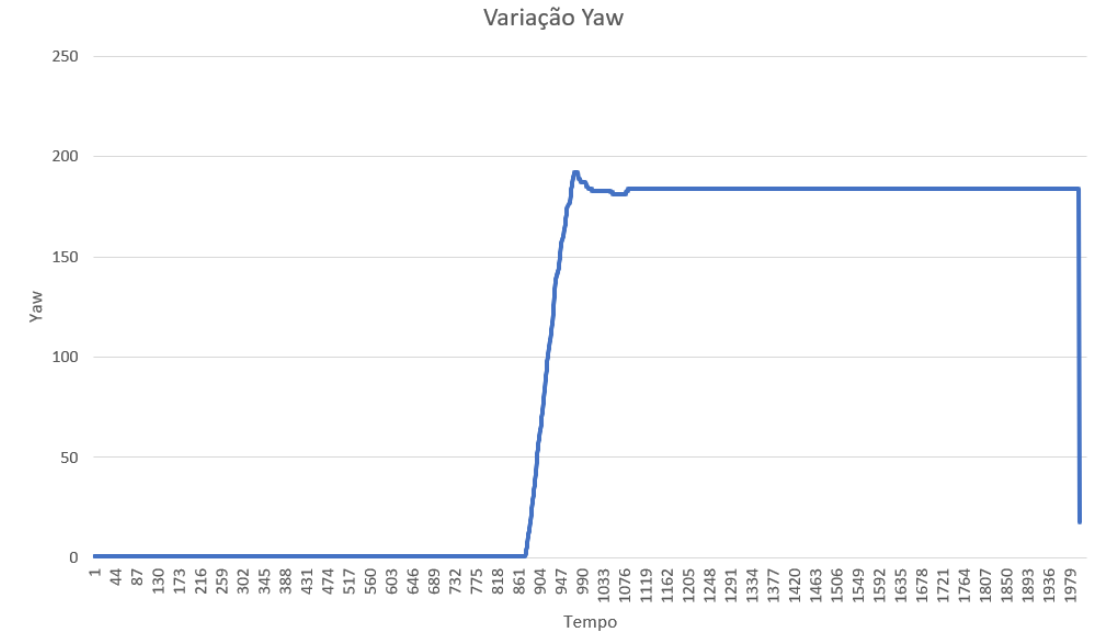


Figura 5.3: Gráfico Variação Yaw

5.2 Dados Recolhidos

Devido à natureza do sistema onde são efetuadas comunicações via BLE entre a aplicação e vários sensores em simultâneo, existe uma janela de período indeterminado no início da gravação dos ficheiros, onde os valores passam por uma fase de ajuste até que estabilizem dentro do que seria esperado. No entanto, como se pode observar na Listagem 5.1, onde o primeiro campo desta é o tempo em milissegundos de cada entrada apresentada em relação ao início da aplicação, verificamos que estamos na presença de períodos relativamente pequenos de ajuste.

De notar que estes períodos de ajuste apenas são verificados quando fazendo a ligação por BLE e com mais do que um sensor ligado, indicando que possivelmente estas são causadas devido a questões de interferência causada pela comunicação entre o sensor e a aplicação, picos de energia na bateria do sensor e latência e/ou sincronismo.

Listagem 5.1 Listagem Com Valores Recolhidos

```
145301;180;177;0
145379;-86;93;114
145387;-81;97;114
145396;-75;100;114
145400;-70;103;114
145408;-64;106;114
145417;-59;109;114
145425;-54;112;114
145434;-49;115;114
145443;-44;118;114
145450;-39;121;114
145454;-34;123;114
145462;-30;126;114
145471;-25;129;114
145479;-20;131;114
145487;-16;134;114
145495;-11;136;114
145500;-7;139;114
145509;-3;141;114
145517;1;143;114
145525;5;146;114
145533;9;148;114
145542;13;150;114
```

Se voltarmos à secção 4.1.1, é indicado que os dados escritos devem estar compreendidos entre 0 e 360 graus, mas observando os valores obtidos na Listagem 5.1 verificamos que as primeiras entradas obtidas se encontram completamente fora desta gama, ainda verificando que estes acabam por se ajustar e estabilizar para o que era inicialmente esperado.

Na secção 4.3.2, é referido que são estabelecidas medidas na aplicação móvel de modo que todos os sensores tenham o ficheiro escrito antes de proceder à gravação e que a mesma só terá início após ser recebida uma notificação que todos estão prontos.

De forma a complementar estas medidas foi definido que o utilizador quando inicia o treino, deve permanecer na posição inicial durante um período de 3 segundos antes de se começar a movimentar. Em `Unity` foram também implementadas alterações aos scripts para que estes apenas iniciem a leitura dos ficheiros 3 segundos após a primeira entrada registada no mesmo fazendo uso do primeiro campo da Listagem 5.1.

Quando estabilizados os valores, estes vão de encontro aos movimentos que foram executados no treino que os originou.

5.3 Aplicação Móvel

De forma a testar a aplicação móvel é importante mencionar novamente as suas funções. A aplicação é responsável por estabelecer comunicação Bluetooth com os sensores e enviar comandos de início e fim de treino conforme a vontade do utilizador. No código implementado no sensor, é estabelecido que quando o sensor estiver conectado via `BLE` a luz do mesmo deve alterar para azul. Para complementar este caso e visto que os sensores vão estar contidos nas pulseiras correspondentes não tendo um feedback visual, a própria aplicação terá uma notificação quando estabelecida a ligação com um sensor.

Figura 5.4: Exemplo de Sensor Conectado

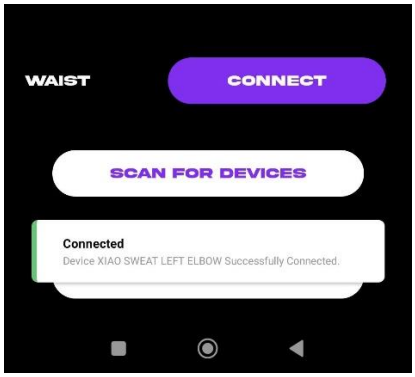


Figura 5.5: Exemplo de Notificação de Sucesso de Conexão

Após realizados os testes podemos concluir que a aplicação aplica de forma correta os protocolos de comunicação, estabelecendo ligações com os sensores.

5.4 Reprodução dos movimentos de treino

Devido à limitação do número de sensores apenas foi possível testar o sistema usando o braço, para isso foram utilizados 3 sensores, um colocado abaixo do ombro, outro no antebraço e o último no pulso. Cada um destes sensores é responsável por guardar informação relativa aos ângulos de cada parte do braço e mão.

Apesar de não ser possível realizar um teste com o corpo inteiro, a realização deste teste permite-nos comprovar o conceito que é possível reproduzir movimentos do corpo humano utilizando `IMUs` estrategicamente posicionados.

Neste teste de um "treino" de cerca de meio minuto foram testados vários movimentos com o braço, colocando-o em várias posições, de forma a verificar se a informação recolhida pelos sensores seria capaz de reproduzir os mesmos. Os resultados destes testes podem ser observados no repositório do trabalho (Alves, PROJ28, 2023) onde se encontram vídeos que demonstram o funcionamento do sistema.

Analisando os vídeos é possível ver que o movimento gravado em tempo real em forma de vídeo é reproduzido em `Unity`, provando assim que a solução proposta funciona de forma correta.

6 Conclusões e Trabalho Futuro

De modo a concluir este documento é necessário refletir se os objetivos propostos foram cumpridos com o culminar do desenvolvimento do projeto.

Fazendo uso da tecnologia existente é possível desenvolver novos tipos de apoios para atletas de alta competição, nomeadamente providenciar estes com ferramentas onde possam analisar os movimentos que executam no contexto das suas atividades. Embora existam vários produtos no mercado que fornecem este tipo de ferramentas, estes não são facilmente acessíveis tanto a nível de disponibilidade como a nível da versatilidade, isto é, são equipamentos que só possuem uma vertente e não podem ser readaptados para outras funções. Como tal foi abordada uma vertente exploratória no projeto fazendo uso de material com um custo mais acessível e facilmente adaptável a diferentes cenários para criar uma ferramenta que possa ser disponibilizada para o público-geral. Usando os sensores **XIAO nRF52840**, conseguimos implementar um sistema capaz de recolher e gravar dados inerciais, dados que se mostraram positivos em vários testes provando que os sensores podem ser usados numa vertente de treino mais exigente e competitiva.

A aplicação móvel criada em conjunto com os protocolos de comunicação desenvolvidos permite a comunicação entre esta e múltiplos sensores com sucesso, sendo ainda esta aplicação é multiplataforma graças ao uso do *framework* `React-Native`. Esta ainda permite ao utilizador ter um guia de como colocar corretamente os sensores para tirar o máximo partido destes.

Em UNITY foi possível implementar scripts que leem com sucesso os ficheiros em formato CSV com os dados inerciais gravados pelos sensores no cartão de memória correspondente. Foi tirado o máximo partido das funcionalidades do UNITY para corretamente reproduzir os movimentos efetuados num modelo 3D humano.

6.1 Trabalho Futuro

Sendo que inicialmente este projeto foi apresentado numa vertente experimental, ao longo do desenvolvimento do mesmo foram deduzidas várias ferramentas que podem e devem ser implementas em trabalhos futuros de modo a não só melhorar o trabalho desenvolvido a nível funcional como oferecer mais ferramentas para que este se torne mais acessível para qualquer tipo de utilizador. Neste contexto, apresentamos as seguintes propostas:

- **Transferência dos ficheiros via porta-série**: O desenvolvimento de uma web app que possa transferir os ficheiros através de uma porta-série irá

facilitar o processo pós-treino aos utilizadores para que possam de forma mais acessível ter acesso aos ficheiros resultantes do mesmo.

- **Criação de Perfis Personalizados**: Esta mesma aplicação app poderá contar com perfis de utilizadores geridos por um servidor onde os utilizadores tenham acesso a áreas pessoais com os seus treinos transferidos, podendo categorizar estes como entenderem.

- **Implementação de mais sensores**: como analisado nos produtos já existentes no mercado, mais sensores consequentemente resulta em melhores resultados. Será interessante implementar mais sensores, por exemplo ao nível das mãos para que possam ser analisados movimentos mais específicos e relativos a desportos de alta competição onde estes têm mais impacto.

- **Novas Vertentes**: embora este trabalho seja direcionado para atletas de alta competição, as bases desenvolvidas permitem que se explorem novas vertentes como a implementação deste tipo de ferramenta para a criação de videojogos ou aplicações que usem os movimentos recordados em tempo real.

- **Executável UNITY**: a criação de um executável em UNITY onde o utilizador tenha uma interface para importar os ficheiros associados a cada articulação disponibilizada relativamente aos sensores utilizados, e onde seja ainda possível ter controlo sobre a reprodução do treino (ex: pausar, reproduzir, avançar, recuar). Também deve ser implementada uma funcionalidade que possa em tempo real comparar os vídeos de treino se disponíveis com o *output* obtido no modelo 3D.

- **Gravar e Exportar Excertos do Treino**: algo que será também interessante a nível do executável UNITY, é incorporar uma função onde excertos considerados importantes do treino possam ser exportados num formato vídeo, para que posam ser facilmente acedidos sem ter de ser necessário todo o processo de configurar a aplicação.

Grande parte destas propostas visam a melhorar a interatividade com o utilizador e todas podem ser desenvolvidas fazendo uso do projeto implementado e documentado neste relatório, visto que este foi desenvolvido usando ferramentas *open-source*.

Referências

AliExpress. (2023). *Módulo Cartão de Memória SD*. Obtido de https://pt.aliexpress.com/item/1865616455.html?gatewayAdapt=glo2bra

Alves, P. (2023). *PROJ28*. Obtido de https://github.com/pedrogalvs/PROJ28/tree/main/03_Teste

Alves, P., & Ginga, M. (2023). *PROJ28*. Obtido de https://github.com/pedrogalvs/PROJ28

Diaz, G. (Dezembro de 2022). Aprendizagem Automática de Estilos de Natação - SWIMU. ISEL.

Inglesias, K. (2023). *3D Character Dummy*. Obtido de https://assetstore.unity.com/packages/3d/characters/humanoids/humans/3d-character-dummy-178395

JLCPCB. (2023). Obtido de https://jlcpcb.com

marcosinigaglia. (2023). *react-native-ble-manager*. Obtido de https://github.com/innoveit/react-native-ble-manager

Mauser. (2023). *Bateria 3.7V 400mAh*. Obtido de https://mauser.pt/catalog/product_info.php?products_id=035-9004

Mauser. (2023). *Rolo de Filamento de Impressão 3D*. Obtido de https://mauser.pt/catalog/product_info.php?cPath=2207_2402_2308&products_id=096-6911

Nansense. (2023). *Suits*. Obtido de https://www.nansense.com/suits/

NeuronMocap. (2023). *Perception Neuro Studio Inertial*. Obtido de https://neuronmocap.com/products/perception-neuron-studio-inertial

PCDiga. (2023). *Cartão de Memória Kingstonc Canvas Select Plus 64GB*. Obtido de https://www.pcdiga.com/cart-o-memoria-kingston-canvas-select-plus-c10-a1-uhs-i-microsdxc-64gb-adaptador-sd-sdcs2-64gb-740617298697

ReactNative. (2021). *React Native*. Obtido de React native - getting started. https://reactnative.dev/docs/getting-started

Rokoko. (2023). *SmartSuit Pro*. Obtido de https://eu.store.rokoko.com/products/smartsuit-pro-

ii?_gl=1*wl5rpo*_ga*MTYzMTg5NzE5My4xNjkzMzA5Mjcx*_ga_2TG4CJCP6E*MTY5MzQ0MjI3Ny4zLjEuMTY5MzQ0MjM5Ni41OS4wLjA.

SeedStudio. (2023). *XIAO-BLE-Sense-nRF52840*. Obtido de https://www.seeedstudio.com/Seeed-XIAO-BLE-Sense-nRF52840-p-5253.html

Virdynmocap. (2023). *Vd Suit Full Motion Capture*. Obtido de https://virdynmocap.jp/en/products/vdsuit-full-motion-capture