ALYSON AKIO HARO
LUCAS GUARDIA PALOPOLI

Sistema de mapeamento dos movimentos de atletas de natação em sua saída

São Paulo
2018

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

ALYSON AKIO HARO

LUCAS GUARDIA PALOPOLI

Sistema de mapeamento dos movimentos de atletas de natação em sua saída

Este relatório é apresentado como requisito parcial para obtenção do grau de Engenheiro Mecatrônico na Escola Politécnica da Universidade de São Paulo.

São Paulo

2018

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

ALYSON AKIO HARO

LUCAS GUARDIA PALOPOLI

Sistema de mapeamento dos movimentos de atletas de natação em sua saída

Este relatório é apresentado como requisito parcial para obtenção do grau de Engenheiro Mecatrônico na Escola Politécnica da Universidade de São Paulo.
Orientador: Prof. Dr. Rafael Traldi Moura

São Paulo

2018

Aos meus pais Adriana e Mario.

Lucas Palopoli

Aos meus pais Angela e Pedro e ao meu irmão Emerson.

Alyson Haro

AGRADECIMENTOS

Ao Professor Dr. Rafael Traldi Moura pela orientação e atenção dados durante o desenvolvimento do projeto.

Aos técnicos André Carvalho e Guilherme Giorgi pela disposição e auxílio na escolha de características a serem medidas.

Ao Professor José Carlos pelo auxílio e sugestões no encapsulamento dos módulos.

Ao CEPEUSP por disponibilizar a piscina para que fossem realizados os testes na água.

A todos os amigos e familiares que enriqueceram o projeto por meio de críticas construtivas e sugestões.

RESUMO

A natação é um dos esportes mais completos que existem, promovendo benefícios físicos e psicológicos que ajudam na manutenção do corpo. Uma vez profissionalizada, a natação se tornou alvo de grandes investimentos tanto em equipamentos como em sistemas para a melhora de performance dos atletas. A análise da saída do atleta é importante pois pode ser responsável por até 26% do tempo total da prova. Este trabalho apresenta o projeto e construção de um sistema de medição e análise de parâmetros importantes para a saída de um nadador. O sistema conta com dois módulos em hardware. O primeiro para coletar a orientação da cabeça, pernas, braços e tronco utilizando sensores de medição inerciais (IMU) e sensor de flexão e o segundo consiste no módulo de controle da filmagem. Em software, o sistema possui duas interfaces principais: uma para a coleta de dados e outra para a exibição dos dados.

Palavras-chave: Natação,análise de saída, IMU, mapeamento, instrumentação

ABSTRACT

Swimming is one of the most complete sports there are, promoting physical and psychological benefits that help maintain the body. Once professionalized, swimming became target of large investments in both equipment and systems to improve athlete’s performance. The analysis of the athlete's swimming start is important because it can be responsible for up to 26% of the total time of the race. This work presents the design and construction of a system for measuring and analyzing important parameters for swimming start. The system has two modules in hardware. The first collects the orientation of the head, legs, arms and trunk using inertial measurement units (IMU) and bending sensor and the second consists in a video recording control module. In software, the system has two main interfaces: one for data collection and another for data display.

Keywords: swimming, swimming start analysis, IMU, mapping, instrumentation.

Lista de Ilustrações

Figura 1: Comparação entre modelos de óculos de natação 14
Figura 2: Comparação entre modelos de touca de natação 14
Figura 3: Trajes Tecnológicos. Fonte: a -< https://www.swiminn.com/natacao/speedo-lzr-racer-bodyskin/46825/p> . b - <https://www.arenawaterinstinct.com/en_us/powerskin-carbon-ultra-jammer.html> 15
Figura 4: Relação entre tempos e colocações do 50m livre masculino nas últimas três Olimpíadas .. 16
Figura 5: Relação entre tempos e colocações do 50m livre feminino nas últimas três Olimpíadas 17
Figura 6: Dados da aceleração desenvolvida por um atleta durante o nado de 50m em piscina semiolímpica. (GUIGNARD et al., 2017) 18
Figura 7: Nesse estudo, foram utilizados: 1) um sensor de força Kistler 9253; 2) Computador com conversor AD e software Bioware; 3) Emissor de sinal de partida; 4) Câmera para captar as imagens das fases de bloco e de voo da saída; 5) Câmera subaquática para capturar as imagens dos 5m aos 10m; 6)) Câmera subaquática para capturar as imagens até os 15m; 7) Câmera para medir o tempo até os 25m; 8)Mixador de vídeo; 9) Alternador de vídeo; 10) Cronômetro de vídeo; 11) VHS para armazenamento das filmagens; 12)Monitor. (FUENTE; GARCIA; ARELLANO, 2000) 19
Figura 8: Bloco de Partida OSB11 (“User’s Manual OSB11 SWIMMING STARTING BLOCK”, 2009). 20
Figura 9: Estilos de nado (“natação – Mais um site Sites Blogs @ MIL”, [s.d.]) 24
Figura 10: Correto movimento da cabeça na fase de bloco 29
Figura 11: Análise do uso do braço na saída. 31
Figura 12: Angulação baixa da perna 32
Figura 13: Análise dos movimentos do atleta PcD 33
Figura 14: Análise de imagem pelo Tracker 38
Figura 15: Angulação da cabeça em relação ao tempo 39
Figura 16: Angulação da perna em relação ao tempo 40
Figura 17: Espectro de frequência do movimento da perna 41
Figura 18: Espectro de frequências do movimento da cabeça 41
Figura 19: Componentes de cada módulo e parâmetros passados entre eles 42
Figura 20: Máquina de estados implementada no microcontrolador 46
Figura 21: Encapsulamento das eletrônicas 47
Figura 22: Encapsulamento do Sensor de Flexão 47
Figura 23: Fixação dos flexores. 48
Figura 24: Fixação das caixas aos membros do atleta 48
Figura 25: Fixação das caixas nas braçadeiras 49
Figura 26: Circuito para acionamento da câmera 49
Figura 27: Módulo de disparo da câmera via plug P3. 50
Figura 28: Comunicação entre as placas para sincronização do acionamento 51
Figura 29: Fluxograma do software com a visualização dos resultados em destaque. 52
Figura 30: Criação da planilha 'Banco de Dados.xls' apenas com o cabeçalho. 53
Figura 31: Ambiente de desenvolvimento GUIDE do MATLAB. 53
Figura 32: Opções de atletas analisados anteriormente 54
Figura 33: Opções de datas para o atleta selecionado 54
Figura 34: Aba "Dados" 54
Figura 35: Aba "Resultados" 54
Figura 36: Escolha do sensor a ser calibrado 55

Figura 37: Escolha do ângulo do flexor a ser calibrado 55
Figura 38: Ferramentas de auxílio na visualização das análises. 56
Figura 39: Orientação dos sistemas de coordenadas das IMUs 57
Figura 40: Módulo de medição dos ângulos no teste a seco 57
Figura 41: Posição de stramline no final da fase aérea da saída. 58
Figura 42: Gráficos da medição do sensor de flexão fixado na nuca 58
Figura 43: Variação do ângulo de flexão do braço 59
Figura 44: Gráficos da medição do sensor de flexão fixado no joelho 60
Figura 45: Gráficos das medidas obtidas pelo eixo Z da IMU localizada nas costas 61
Figura 46: Gráficos das medidas obtidas pelo eixo X da IMU localizada no braço 62
Figura 47: Gráficos das medidas obtidas pelo eixo Z da IMU localizada na coxa 63
Figura 48: Gráficos da medição do sensor de flexão fixado na nuca 64
Figura 49: Gráficos da medição do sensor de flexão fixado no cotovelo 66
Figura 50: Gráficos da medição do sensor de flexão fixado no joelho 68
Figura 51: Gráficos das medidas obtidas pelo eixo Z da IMU localizada nas costas 70
Figura 52: Diferença entre as inclinações das partes a e b da Figura 51 70
Figura 53: Gráficos das medidas obtidas pelo eixo X da IMU localizada no braço 71
Figura 54: Gráficos das medidas obtidas pelo eixo Z da IMU localizada na coxa 73
Figura 55: Variação angular no tempo do ângulo de flexão do joelho 74
Figura 56: Diagrama com análise mais robusta do movimento 78

Lista de Tabelas

Tabela 1: São reconhecidos como Recordes Mundiais e Recordes Mundiais Juniores, em piscina de 50 metros, as seguintes distâncias e nados para ambos os sexos (CBDA): 26
Tabela 2: São reconhecidos como Recordes Mundiais, em piscina de 25 metros, as seguintes distâncias e nados para ambos os sexos (site CBDA):..... 27
Tabela 3:Uso da cabeça na condução da saída de um nadador. 29
Tabela 4:Uso da perna traseira no alinhamento do corpo na saída de um nadador. 30
Tabela 5: Compilação entre entrevistas e artigos. 34
Tabela 6: Tempos de cada fase dos movimentos da cabeça de da perna e frequência de amostragem necessária para cada um deles. 40
Tabela 7: Ações executadas no microcontrolador 46
Tabela 8: Comparação dos resultados com os requisitos 76

Sumário

1 INTRODUÇÃO 13

1.1 Tema 13

1.2 Estado da Arte 18

2 OBJETIVOS 22

3 EMBASAMENTO TEÓRICO 23

3.1 Natação 23

3.1.1 Histórico 23

3.1.2 Evolução dos estilos de nado 23

3.1.3 Importância da natação 25

3.1.4 Regras 26

3.1.5 Federação Internacional de Natação (FINA) 27

4 PROJETO BÁSICO 28

4.1 Identificação de Parâmetros 28

4.2 Requisitos do Projeto 38

4.2.1 Requisitos Mecânicos 38

4.2.2 Requisitos Elétricos 38

4.2.3 Requisitos Funcionais 42

4.2.4 Não funcionais 42

4.3 Análise Mecatrônica 42

4.3.1 Medição dos Ângulos de Interesse. 43

4.3.2 Módulo de Filmagem da saída do atleta 44

5 IMPLEMENTAÇÃO 45

5.1 Microcontrolador 45

5.2 Módulo de medição dos ângulos dos membros do atleta 45

5.2.1 Circuito elétrico 45

5.2.2 Programa embarcado 45

5.2.3 Construção Mecânica e Isolamento contra água 46

5.3 Módulo de captura de vídeo 49

5.4.1 ESP-NOW 50

5.4.2 Comunicação entre os módulos 51

5.5 Concepção de software – Interface Gráfica 51

5.5.1 Fluxograma 52

5.5.2 Detalhamento 52

5.5.3 Código fonte do GUIDE 56

6 RESULTADOS 57
6.1 Orientações dos sistemas de coordenadas das IMUs 57
6.2 Teste a seco 57
6.2.1 Sensor de Flexão da Nuca 58
6.2.2 Sensor de Flexão do Braço 59
6.2.3 Sensor de Flexão da Perna 60
6.2.4 IMU – Eixo Z das Costas 61
6.2.5 IMU – Eixo X do Braço 62
6.2.6 IMU – Eixo Z da Perna 63
6.3 Teste na piscina 63
6.3.1 Sensor de Flexão da Nuca 64
6.3.2 Sensor de Flexão do Braço 65
6.3.3 Sensor de Flexão da Perna 67
6.3.4 IMU – Eixo Z das Costas 69
6.3.5 IMU – Eixo X do Braço 71
6.3.6 IMU – Eixo Z da Perna 72
7 DISCUSSÃO 74
7.1 Medição com os sensores de flexão 74
7.2 Medição com as IMUs 75
7.3 Comparação com os requisitos 75
8 CONCLUSÃO 77
8.1 Próximos passos e pontos a melhorar 77
8.1.1 Sensores de flexão. 77
8.1.2 Inclusão de outros módulos de medição 78
8.1.3 Medição da força aplicada pelos braços no bloco 79
8.2 Reflexão 79
9 REFERÊNCIAS BIBLIOGRÁFICAS 80
10 ANEXOS 83
10.1 Código implementado nos módulos de medição de ângulos 83
10.1.1 Para o módulo da nuca – Master 83
10.1.2 Para o módulo do braço e perna – Slaves 104
10.2 Código implementado no controle de disparo da câmera 112
10.3 Código fonte da interface gráfica 114
10.4 Circuito do módulo de medição dos ângulos dos membros do atleta 143
10.5 Termos de autorização de uso de imagem 144

1 INTRODUÇÃO

1.1 Tema

A prática de exercícios físicos traz benefícios físicos e psicológicos à saúde, promovendo mudanças no corpo que ajudam em sua manutenção. Os benefícios físicos podem ser evidenciados pela redução da taxa de ataques cardíacos pois tornam o sangue mais fino, resultando em menor resistência ao fluir pelos vasos sanguíneos. Além disso, aumentam o número de veias que levam sangue ao coração, diminuem a pressão sanguínea, ajudam a prevenir a diabetes, reduzem o risco do desenvolvimento câncer de cólon e de mama, fortalecem ossos e auxiliam na proteção das articulações. Os psicológicos relacionam com um intuito social. Eles aumentam a autoestima pela liberação de hormônios como endorfina, reduzem o estresse e consequentemente reduzem os níveis de depressão e suicídio (MYECHIA MINTER-JORDAN, IRENE S. DAVIS, 2013).

A natação é um dos esportes mais completos, com muitos benefícios ao utilizar grande maioria músculos. De acordo com (LAZAR et al., 2013), é um esporte de baixo impacto, tornando-o ideal para pessoas com problemas nas articulações e que não podem realizar exercícios em solo. (TANNER, 2017), em uma matéria da revista Time®, enumerou vantagens da natação: aumento da força dos músculos respiratórios, uso maior dos músculos superiores e do tronco e ausência de grande impacto na coluna. (ALKATAN et al., 2016) demonstraram que as pessoas tendem a gostar mais da natação, devido à flutuabilidade e à agradável temperatura.

Em decorrência de sua crescente importância no mundo, a natação se tornou um esporte profissional. De acordo com o Comitê Olímpico Internacional, a natação é um esporte Olímpico desde 1896, sendo seu primeiro evento uma competição de nado livre nas águas geladas do Mediterrâneo. De seu começo até hoje, a competição de natação passou por grandes mudanças, como a criação da *Fédération Internationale de Natation* (FINA) e a implantação de outros estilos de nado e distâncias percorridas. Atualmente, a competição olímpica conta com 16 modalidades, sendo 13 individuais: nado livre (50m,100m, 200m, 400m, 800m, 1500m e 10000m), nado borboleta, nado costas e nado peito (100m e 200m); e os revezamentos: 4x100m nado livre, 4x200m nado livre e 4x100 medley.

Para possibilitar a melhora contínua e o aumento de performance dos atletas, grandes investimentos foram realizados, tanto pela FINA quanto por iniciativas privadas. Eles podem ser divididos em três tipos: estruturais, equipamentos utilizados pelos atletas e sistemas de análise para melhoria de performance.

Em relação às estruturais, para a orientação dos atletas durante o percurso, foram introduzidas bandeiras e marcas visuais no fundo da piscina para contagem de braçadas, condução reta do nado e indicações para viradas e chegadas. Pode-se também destacar o desenvolvimento de raias anti-marola, criadas para reduzir a interferência entre a movimentação da água decorrente do nado de diferentes atletas, além das constantes modificações no bloco de saída.

Em relação à evolução dos equipamentos, o atleta passou a utilizar óculos (Figura 1) e toucas com formato hidrodinâmico, as chamadas toucas capacete, para reduzir o arrasto (Figura 2).

a. Comparação entre tamanho de dois modelos diferentes da Speedo. O azul (Flash) é para competição, enquanto o preto (Neon Tek) é de treino.

b. Comparação entre conforto proporcionado pelos modelos. Pelo sistema de ajuste e tamanho das ventosas, o preto é mais confortável que o azul que tem por objetivo ser mais leve e hidrodinâmico para uso em competições.

Figura 1: Comparação entre modelos de óculos de natação

a. Dois tipos de touca diferentes, a preta é o modelo capacete e a azul é a tradicional.

b. Comparação entre as estruturas das toucas. A capacete tem estrutura hidrodinâmica e é mais rígida que a tradicional, portanto é a utilizada em competições.

Figura 2: Comparação entre modelos de touca de natação

Soma-se ainda o uso de trajes tecnológicos como o LZR da Speedo®, desenvolvido em conjunto com a NASA, e os semelhantes lançados posteriormente de poliuretano, que foram banidos por serem um "*doping* tecnológico" pois auxiliavam na flutuabilidade e estímulo muscular do atleta (COMMISSION; TECHNOLOGY, 2009)

a. Traje tecnológico banido pela FINA em 2009.

b. Traje masculino permitido nos dias atuais.

Figura 3: Trajes Tecnológicos. Fonte: a -< https://www.swiminn.com/natacao/speedo-lzr-racer-bodyskin/46825/p> . b - <https://www.arenawaterinstinct.com/en_us/powerskin-carbon-ultra-jammer.html>.

No que tange os sistemas de análise de performance, estes podem ser divididos em dois grupos: um onde os sensores estão fora do atleta e outro com sensores fixos no atleta. O primeiro consiste em sua maioria a sistemas de edição e análise de imagem, como o SwimPro[1], que realiza apenas a edição, e Simi[2], um sistema de análise automática de imagens. O segundo, consiste no uso de unidades de medições inerciais (IMU em inglês) acoplados ao atleta como o K-sens[3].

Dentre as principais vantagens da IMU, pode-se destacar que uma vez que estejam vedadas contra a água, IMU elas podem capturar um grande volume de dados de um treino inteiro de um atleta (GUIGNARD et al., 2017). Além disso, elas não precisam de procedimentos de digitalização(GUIGNARD et al., 2017), não há interferência entre duas quando dois nadadores próximos estiverem utilizando-as, apresentam baixo custo e são portáveis para fácil uso em ambientes externos (PANSIOT; LO; YANG, 2010). Diferentemente, as câmeras são mais caras e ficam sujeitas às intempéries do ambiente, apresentando dificuldades da detecção de pontos devido a água, respingos, variação de iluminação e colorações.

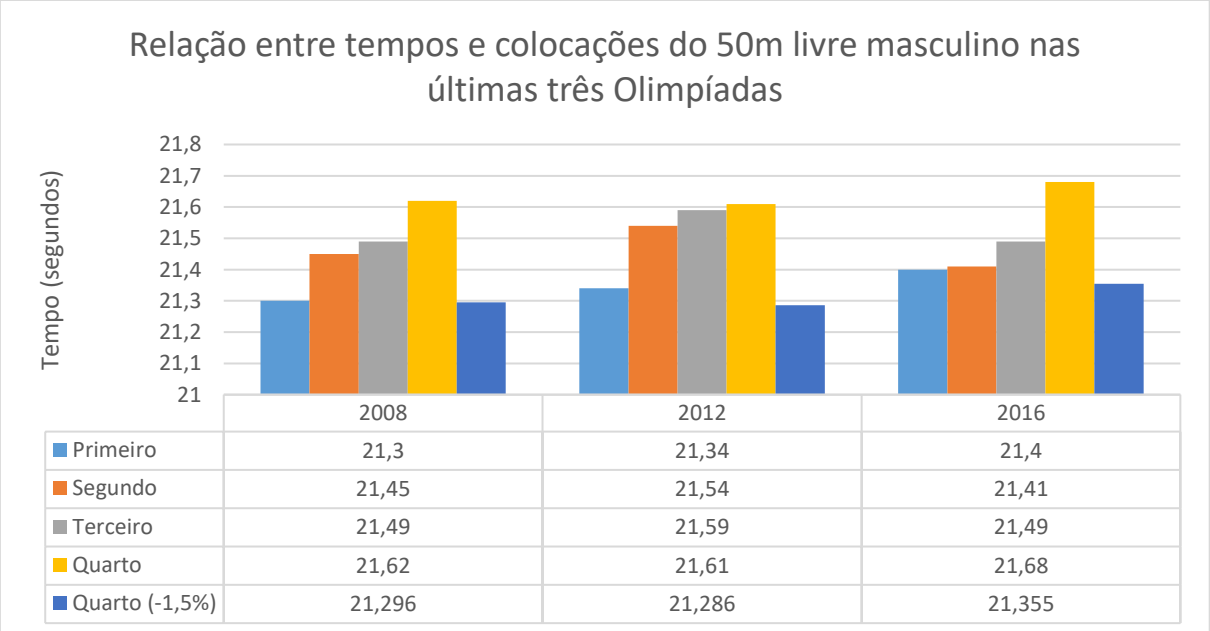
[1] https://www.swimmingcam.com/
[2] http://www.simi.com/en/applications/sport/swimming-analysis.html

[3] https://www.k-invent.com/k-sens/

A análise da saída do atleta é importante pois de acordo com (VANTORRE; CHOLLET; SEIFERT, 2014), a influência do tempo de saída é responsável por 0,8% a 26,1% do tempo total do atleta, considerando que o início do nado é dado pelos primeiros 15m (TOR; PEASE; BALL, 2015). Tendo em vista que o maior impacto ocorre nas provas de curta distância, através de uma breve análise dos resultados olímpicos mais recentes das provas de 50 metros nado livre (Figura 4,Figura 5) obtidos do site da FINA, depreende-se que com uma pequena redução no tempo do quarto colocado, 1,5% no masculino e 2% no feminino, esses atletas, que não subiram ao pódio, passariam para o primeiro lugar. Explicita-se assim a importância do desenvolvimento de instrumentos que auxiliem atletas e técnicos a visualizar erros e possíveis correções para a melhora da performance da saída.

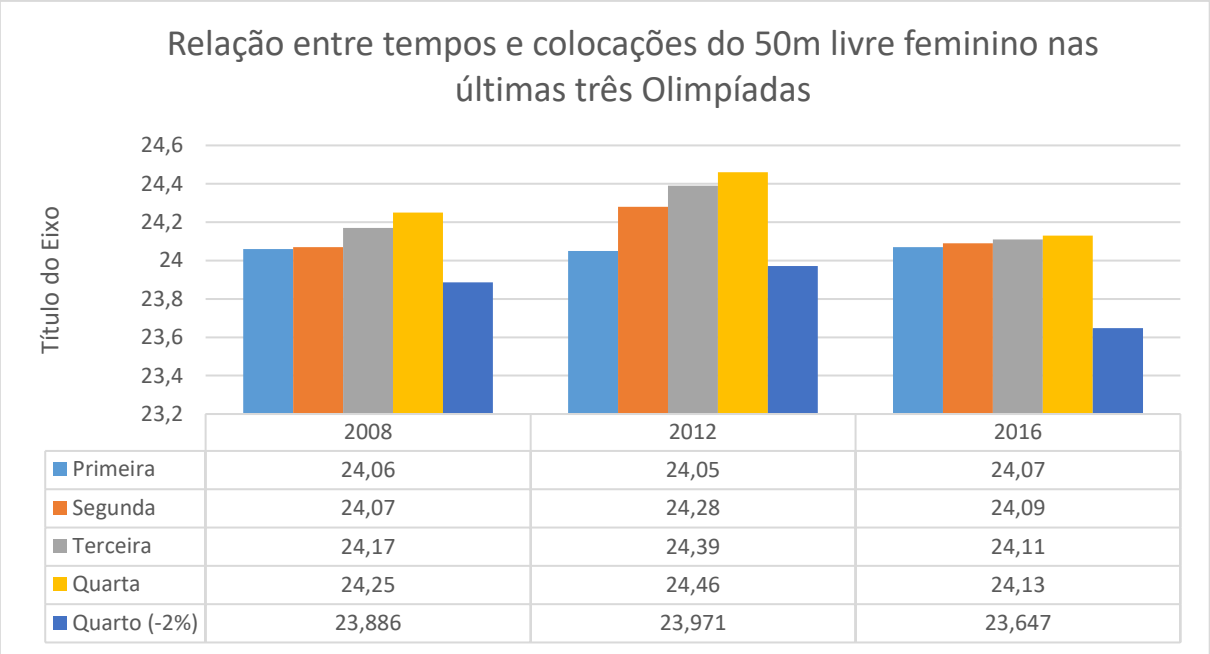
No entanto, nenhum dos sistemas citados anteriormente têm a saída do atleta como foco. O presente trabalho fará uma análise baseada em artigos e opiniões de técnicos de natação para identificar os principais pontos que mais influenciam a saída e, depois, será desenvolvido um sistema para medi-los.

Fica clara a importância da análise das características dos nadadores. Em especial da saída, que pode corresponder porcentagem significativa da prova, apresentando influência direta nos resultados até mesmo de grandes competições.



	2008	2012	2016
Primeiro	21,3	21,34	21,4
Segundo	21,45	21,54	21,41
Terceiro	21,49	21,59	21,49
Quarto	21,62	21,61	21,68
Quarto (-1,5%)	21,296	21,286	21,355

Figura 4: Relação entre tempos e colocações do 50m livre masculino nas últimas três Olimpíadas



	2008	2012	2016
Primeira	24,06	24,05	24,07
Segunda	24,07	24,28	24,09
Terceira	24,17	24,39	24,11
Quarta	24,25	24,46	24,13
Quarto (-2%)	23,886	23,971	23,647

Figura 5: Relação entre tempos e colocações do 50m livre feminino nas últimas três Olimpíadas

1.2 Estado da Arte

Há muitos anos a sociedade começou a medir e quantizar características buscando problemas e soluções. A primeira ferramenta desenvolvida para medições foi o cronômetro para aquisição dos tempos. Atualmente, encontram-se sensores de nado para captar posições, velocidades, acelerações (Figura 6), forças, energia expendida e calorimetria indireta por troca gasosa(DADASHI et al., 2013). Destes sensores, destacam-se câmeras, giroscópios e acelerômetros. Apesar da vasta quantidade de ferramentas de análise, nenhuma se propôs a trabalhar em conjunto sensores e análises automáticas das características das saídas. Por isso, fez-se uma amostra separada dessas aplicações (estudo analítico da saída e sensores para auxiliar na natação).

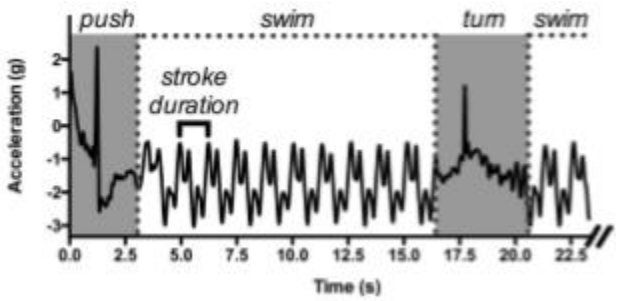


Figura 6: Dados da aceleração desenvolvida por um atleta durante o nado de 50m em piscina semiolímpica. (GUIGNARD et al., 2017)

A análise dos fenômenos envolvidos na natação é um tema largamente estudado, indo desde esforços do atleta durante a saída do bloco à análise da aceleração, velocidade e posição de seus diversos membros durante o nado. Para análise da saída do nadador, existem estudos que investigam a influência do aquecimento de músculos do tronco (IIZUKA, 2016), também existem pesquisas que estudam a correlação entre os músculos inferiores com a performance da saída do nadador (GARCÍA-RAMOS et al., 2016). Em relação ao uso de sensores para a análise dos movimentos, grande parte dos estudos utilizam câmera (Figura 7) para captar velocidades e ângulos de saída e entrada na água ou para capturar o movimento do atleta durante o nado (MOONEY et al., 2015). Há também pesquisas que utilizam unidade de medição inerciais para coletar estes parâmetros(DADASHI; MILLET; AMINIAN, 2013).

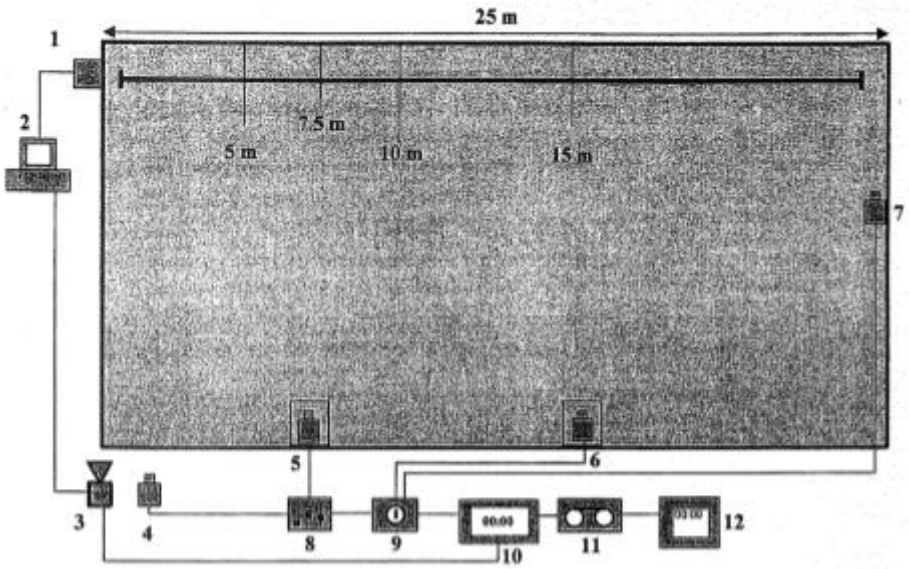


Figura 7: Nesse estudo, foram utilizados: 1) um sensor de força Kistler 9253; 2) Computador com conversor AD e software Bioware; 3) Emissor de sinal de partida; 4) Câmera para captar as imagens das fases de bloco e de voo da saída; 5) Câmera subaquática para capturar as imagens dos 5m aos 10m; 6)) Câmera subaquática para capturar as imagens até os 15m; 7) Câmera para medir o tempo até os 25m; 8)Mixador de vídeo; 9) Alternador de vídeo; 10) Cronômetro de vídeo; 11) VHS para armazenamento das filmagens; 12)Monitor. (FUENTE; GARCIA; ARELLANO, 2000)

Dentre os trabalhos que analisam a influência da musculação com a saída do atleta, há o trabalho de (GARCÍA-RAMOS et al., 2016) que estudou a correlação entre os músculos inferiores com o tempo de saída do nadador. Neste estudo foram coletados o tempo de saída e a velocidade de nadadoras além de testes para avaliar a performance muscular dos membros inferiores das mesmas. Para a coleta do tempo de saída foram utilizadas duas câmeras dentro da piscina e uma câmera fora da piscina e o tempo de saída foi definido como o tempo em que o nadador chega aos 15 metros, de acordo com as regras da FINA. Para a análise da força muscular dos membros inferiores, as atletas foram submetidas a quatro formas de coleta, são eles: salto de agachamento, salto de agachamento com peso variável, salto de contramovimento e a máxima contração isométrica voluntária. Para cada uma dessas formas de coleta foram coletados força de pico, força de pico específica (força de pico dividida pela massa), potência de pico, potência de pico específica, a velocidade inicial e a velocidade de pico. O estudo concluiu que há correlação entre os saltos com peso adicional e o tempo de saída. (FUENTE; GARCIA; ARELLANO, 2000) fizeram um trabalho similar, com o objetivo de encontrar uma correlação com a força vertical envolvida no salto de contra movimento e o tempo de saída. Neste trabalho participaram 44 voluntários masculinos e 21 atletas femininos, a força dos saltos foi medida utilizada uma plataforma de força. Foi concluído que a correlação é pequena entre os parâmetros estudado e confirmou-se que a fase mais importante na saída do nadador é a fase na água. Desta forma, parâmetros como velocidade e angulação de entrada são parâmetros importantes para o desempenho final do atleta (TOR; PEASE; BALL, 2015).

Segundo (HONDA et al., 2012) em 2008 foi desenvolvido o Bloco de partida OSB 11, que conta com a adição de um apoio a 30 graus da superfície do bloco e 5 posições como proposta de aumento de força horizontal e consequentemente da

velocidade horizontal de saída do bloco. Pesquisas como as de (BARLOW et al., 2014), (SLAWSON et al., 2012), (HONDA et al., 2012), estudam como o bloco Omega OBS11 (Figura 8), aprovado em 2009 pelo órgão internacional de natação (DRAGUNAS, 2015), influencia na saída de nadadores, visto que este é o bloco atualmente usado em olimpíadas. O trabalho de (BARLOW et al., 2014) contou com a participação de 7 voluntários masculinos e 3 femininos e objetivava identificar diferenças nos tempos da fase de bloco e da fase de voo, foi utilizado análise de vídeo para coletar posição do quadril no começo, tempo de reação, tempo de movimento, tempo de bloco, ângulo de saída, ângulo de entrada, tempo para as distâncias de 5m e 15m e velocidade média entre as distâncias 4.5~5.5 e 14.5~15.5. No trabalho de (SLAWSON et al., 2012) utilizou-se uma plataforma de força Kistler 9260AA para a parte principal do bloco e transdutores Kistler 9017B para o descanso de pé para a coleta de força, já para a coleta de ângulos foi utilizada uma câmera de alta velocidade. (HONDA et al., 2012) utilizou instrumentações parecidas.

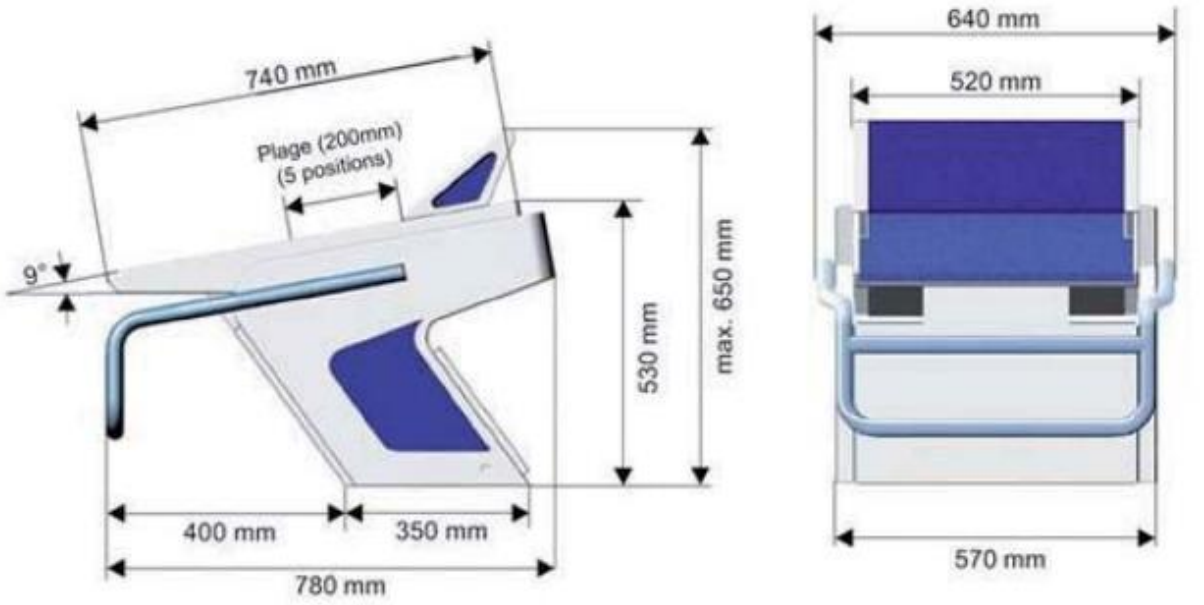


Figura 8: Bloco de Partida OSB11 (“User’s Manual OSB11 SWIMMING STARTING BLOCK”, 2009).

A avaliação técnica do desempenho da natação é um fator essencial da elite atlética e, para isso existem sistemas comerciais que auxiliam por meio do uso de câmeras. No entanto, não possuem potencial analítico, precisão e velocidade operacional suficientes para o uso de atletas de elite. Parâmetros de desempenho importantes, como velocidade e aceleração, requerem ferramentas de análise sofisticadas para seus estudos devido às dificuldades em se obter dados precisos em ambientes aquáticos. Para isso, tem-se investido cada vez mais em instrumentos de sensoriamento inercial. Segundo os apontamentos de (MOONEY et al., 2015), tem-se optado por sensores microeletromecânicos, acelerômetros e giroscópios alternativamente à base de vídeo em relação ao desempenho e aos gastos de energia, bem como *feedback* em tempo real para o atleta (HAGEM et al., 2013), possibilitando correções mais eficientes em nível competitivo. Foi proposto por eles também que é possível saber os diferentes locais de conexão e escolha de sensores únicos e

múltiplos ajudando os profissionais a selecionar os sistemas e métodos mais apropriados para extração dos principais parâmetros relacionados ao desempenho.

Outra aplicação de sensores como as IMUs foi proposta por (DADASHI et al., 2013) e (DADASHI; MILLET; AMINIAN, 2014) que exploraram parâmetros e variabilidade de dinâmica de coordenação do nado no que diz respeito ao gasto calórico, calorimetria indireta e concentração de lactato no sangue. Para isso foram instaladas IMUs em locais estratégicos como antebraços, sacro e perna, concluindo-se que o erro se mantinha semelhante em casos sem a observação dos antebraços. Neste estudo, concluíram que o uso de *Wearable Aquatic Movement Analysis System* (WAMAS) oferece uma ferramenta conveniente para estudos adicionais sobre a biomecânica da natação, estimativa precisa do gasto de energia e ferramenta prática sem necessidade de uso laborioso de medidas de troca de gás e amostragem de lactato sanguíneo. Também em relação aos estudos referentes a IMUs, pode-se salientar os de (GUIGNARD et al., 2017), que estipularam o procedimento da impermeabilização dos sensores, exploraram parâmetros e variabilidade da dinâmica de coordenação monitorando o desempenho do atleta e obtendo recomendações práticas para treinadores de elite.

2 OBJETIVOS

Este trabalho tem como objetivo o projeto e construção de um sistema de análise da saída do atleta, fornecendo resultados em interface amigável para o técnico, auxiliando-o em sua tomada de decisão.

O sistema deve ser capaz de capturar as angulações da cabeça, pernas e braços em relação ao tronco e sincronizar com a filmagem realizada.

3 EMBASAMENTO TEÓRICO

3.1 Natação

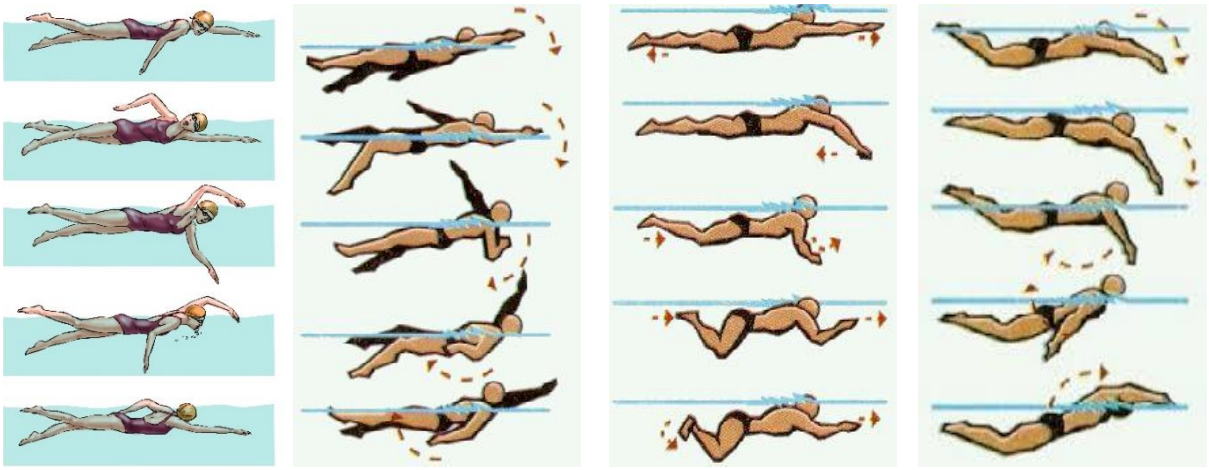
3.1.1 Histórico

A natação é “a habilidade que permite ao ser humano deslocar-se num meio líquido, normalmente a água, graças às forças propulsivas que gera com os movimentos dos membros superiores, inferiores e corpo, que lhe permitem vencer as resistências que se opõem ao avanço" (SAAVEDRA; YOLANDA; FERRAN, 2003).

Desde os primórdios da existência humana, a natação se mostrou importante para sua sobrevivência, seja na busca de alimentos, ou na fuga de perigo em terra. O primeiro registro data de 5.000 a.C. em pinturas rupestres da Rocha de Gilf Kebir no Egito (LEWILLIE, 1983). Na Grécia, essa prática passou a fazer parte da educação, além disso, proporcionava o desenvolvimento harmonioso do corpo. Na Roma antiga, ela esteve mais vinculada à recreação e à preparação militar dos soldados do império (LEWILLIE, 1983). Devido à crença de sua correlação com a disseminação de doenças, houve certa estagnação no período da Idade Média. Novamente, no Renascimento, a modalidade ascendeu com a criação de piscinas públicas na Europa. No ano de 1828, foi construída, em Londres, a primeira piscina coberta. Como esporte competitivo, ou seja, como a natação desportiva, foi introduzida em 1837 na Inglaterra tendo sua primeira competição organizada (SAAVEDRA; YOLANDA; FERRAN, 2003), colocando o objetivo aos atletas de deslocarem-se da forma mais rápida possível. Com isso, houve a necessidade de regulamentar as competições e com esse objetivo, em 1874, nasce a primeira federação de clubes, a *"Association Metropolitan Swimming Clube"*, por meio da qual passou a ser possível estabelecer regulamentos e recordes (SAAVEDRA; YOLANDA; FERRAN, 2003). Prova da consolidação e importância desse esporte foi sua presença já nos primeiros Jogos Olímpicos modernos, em 1894. Através da União de Regatas Fluminense, Rio de Janeiro, esse esporte foi introduzido no Brasil.

3.1.2 Evolução dos estilos de nado

Há na natação 4 estilos, borboleta, crawl, costas e peito. O crawl (Figura 9a) é definido pelo movimento alternado e coordenado da circulação dos braços e batida de pernas com rotação da cabeça para respiração (ARELLANO, 1992). Há indícios de seu surgimento na Inglaterra em meados do século XIX, em 1840, mas sendo apenas subaquático, por volta de 1850, o australiano Wallis usou a recuperação aérea dos membros superiores (REYES, 1998). Depois de algumas alterações, como nadar sobre o abdômen, usar pontapé do nado peito, movimento alternativo dos membros inferiores, batida de seis tempos (REYES, 1998), respiração bilateral, tração descontínua e, por fim, a tração subaquática com flexão de cotovelo na metade do percurso, em 1955, por Jhon Weismüller, o nado atingiu a forma como é usado hoje em dia(SAAVEDRA; YOLANDA; FERRAN, 2003).



a. Nado Crawl b. Nado costas c. Nado Peito d. Nado Borboleta

Figura 9: Estilos de nado ("natação – Mais um site Sites Blogs @ MIL", [s.d.])

O estilo costas usado na atualidade (Figura 9b) pode ser definido como o deslocamento na posição dorsal com movimento alternativo e coordenado da circulação completa das extremidades superiores e batido das inferiores com um giro no eixo longitudinal durante o nado (SAAVEDRA; YOLANDA; FERRAN, 2003). A origem do estilo costas começa quando se nada sobre o dorso do corpo realizando a braçada de forma simultânea e com pontapé de braça. Em 1912, surge o nado com o uso das pernas em forma de pedalado, adiante, em 1930, o joelho está mais estendido, em 1933, o corpo mais horizontalizado e realizando a entrada dos membros superiores mais abertos, em 1948, o é introduzida a flexão de cotovelo para impulso (SAAVEDRA; YOLANDA; FERRAN, 2003). Por fim, a última variação se deu em 1960, com um giro no eixo longitudinal quando entra o membro superior na água, e flexionando o cotovelo 90 graus quando o braço está perpendicular ao ombro (REYES, 1998)

A definição do nado peito atual (Figura 9c) é o deslocamento ventral com movimentos simultâneos, simétricos e coordenado dos membros superiores de forma circular e inferiores com um pontapé, com um movimento de ascensão e descenso de ombros e quadris que, coordenado com os membros superiores permite realizar a inspiração. O peito é o estilo mais antigo dos quatro existentes; seu início se deu com os membros superiores completamente estendidos, e existindo muita separação entre os membros inferiores. Em 1924, a braçada passou a ser realizada em profundidade, um deslizamento horizontal, e uma posição mais baixa dos joelhos, adiante, houve a flexão dos cotovelos durante a tração (SAAVEDRA; YOLANDA; FERRAN, 2003). Posteriormente, em 1946, aparecem duas vertentes, a braçada submersa com três ou quatro ciclos que levavam as mãos até a altura dos quadris e a Braça-borboleta, consistente em realizar o recobro dos membros superiores por fora da água, os estilos se separam provisionalmente em 1949, produzindo-se sua separação definitiva em 1953 (REYES, 1998). No ano 1961, há ação simultânea e simétrica do pontapé, que é a que chega até nossos dias. A partir de 1980 surgem dois modelos de braçada, a Formal, realizada por americanos, com flexão-extensão do pescoço e a Natural, realizada por europeus, com posição oblíqua do corpo que facilita a respiração pela consequente movimentação dos ombros e quadril (MAGLISCHO; MAGLISCHO,

2003). Uma variante proveniente da Hungria é a Braçada Onda que se caracteriza por realizar um movimento ondulatório do corpo para que o atleta possa aproveitar a onda criada por ele mesmo (NAGY; URBANCHEK, 1990).

O estilo borboleta (Figura 9d) é deslocamento ventral com movimento simultâneo e coordenado dom membros superiores com circulação e inferiores com um batido e ondulação de todo o corpo coordenando os demais movimentos inclusive respiração. Proveniente de uma recuperação aérea do nado peito, o primeiro em utilizar esta variante foi o alemão Rademacher. O nado tal como é desenvolvido hoje em dia foi realizado primeiramente pelo húngaro Tempeck, separando peito e borboleta por completo, introduzindo a pernada de borboleta ondulada (SAAVEDRA; YOLANDA; FERRAN, 2003).

O nado Medley compreende as quatro modalidades supracitadas, tendo início com o nado borboleta, seguido do costas, peito e por fim crawl, todos responsáveis por um quarto do percurso da prova, seja ela de 100, 200, ou 400 metros. No caso do revezamento, a ordem é diferente, devido a impossibilidade da saída aérea do nado costas esse é o primeiro, seguido então do peito, borboleta e crawl.

3.1.3 Importância da natação

Devido a correlação entre os benefícios para a reabilitação e esportes, a cada dia, mais pessoas com incapacidade física se envolvem em alguma prática. A inserção desses indivíduos em atividades físicas leva a um aumento da aptidão física, da eficiência dos movimentos e do desenvolvimento de habilidades motoras. A atividade física regular pode trazer novas perspectivas para indivíduos com incapacidades físicas, incluindo novas amizades e até oportunidades de emprego, devido ao aumento da produtividade (SHEPHARD, 1991). A natação é um dos esportes mais apropriados para esse público devido às facilidades proporcionados pela execução de movimentos com o corpo imerso na água. Nela, há o desenvolvimento da coordenação, condicionamento aeróbio, redução da espasticidade e recorre em menos fadiga que outras atividades. “Sabe-se que o uso de exercícios terapêuticos na água é mencionado em obras tão antigas como a de Aureliano, do final do século 5, na qual recomenda natação no mar ou em nascentes quente, e a do médico Jacques Delpech (1777-1838), que escreveu sobre a correção postural com aparelhos e enfatizou o valor da natação para a coluna vertebral. No final do século 19 e início do século 20, os exercícios aquáticos começaram a ser utilizados como meios corretivos eficientes, e as doenças reumáticas e do aparelho locomotor receberam tratamento pioneiro nas estâncias termais europeias. Mais tarde, novos métodos surgem ressaltando o valor do exercício terapêutico dentro da água, acima do valor de suas características quimiotérmicas. Em 1924, Lowman organiza uma hidroginástica terapêutica, dentro de tanques ou piscinas, para portadores de poliomielite paraplégicos e portadores de outros problemas ortopédicos” (TSUTSUMI et al., 2004).

As atividades motoras em meio líquido visam o desenvolvimento cognitivo, afetivo, emocional e social, sendo mencionadas como um excelente meio de execução motora, favorecendo o desenvolvimento global do indivíduo portador de deficiência física. Depreende-se, que o uso da natação no acompanhamento de

quadros de deficiência física traz benefícios físicos, emocionais e, consequentemente, da qualidade de vida (TSUTSUMI et al., 2004).

3.1.4 Regras

No brasil, o órgão responsável por determinar as regras das competições é a Confederação Brasileira de Desportos Aquáticos (CBDA). Dentre as regras que conduzem esses eventos esportivos pode-se destacar as de saída, virada, chegada e infraestruturas. Essas últimas, são essencialmente o tamanho da piscina que pode ser de 50 metros no caso das Olímpicas, ou de 25 metros no caso das Semiolímpicas, profundidade mínima de 3 metros e temperatura entre 25 e 27 graus. Para a saída, pode-se destacar a proibição de queimar a largada, ou seja, saltar antes da largada, sob pena de desclassificação ("CBDA - Natação, Pólo Aquático, Maratonas Aquáticas, Nado Sincronizado, Saltos Ornamentais.", [s.d.]). Os tipos, são três a saída em cima do bloco usada nas largadas de borboleta, crawl, peito e primeiro atleta de revezamento livre, os demais atletas podem saltar de qualquer forma, desde que só saiam da baliza após o nadador que o antecedeu toque na borda e, por fim, a saída de dentro da água segurando no apoio inferior da baliza, que é realizada pelo nado costas e primeiro atleta do Revezamento Medley. As viradas são feitas com cambalhota para os nados crawl e costas e, no caso do peito e borboleta, é necessário primeiro encostar as duas mãos simultaneamente na borda e depois realizar a impulsão com as pernas na mesma. Para a chegada, há a necessidade de tocar a borda, para o costas e crawl com uma mão e para peito e borboleta com as duas.

As provas de 50, 100 e 200 metros apresentam fases eliminatórias, semifinais e final, enquanto as demais têm apenas eliminatória e final. As provas de 800 metros são exclusivamente femininas e as de 1500 metros, masculinas. Todas as provas oficiais estão listadas em Tabela 1 e Tabela 2 ("CBDA - Natação, Pólo Aquático, Maratonas Aquáticas, Nado Sincronizado, Saltos Ornamentais.", [s.d.]).

Tabela 1: São reconhecidos como Recordes Mundiais e Recordes Mundiais Juniores, em piscina de 50 metros, as seguintes distâncias e nados para ambos os sexos (CBDA):

Livre	50, 100, 200, 400, 800 e 1500 metros
Costas	50, 100 e 200 metros
Peito	50, 100 e 200 metros
Borboleta	50, 100 e 200 metros
Medley	200 e 400 metros
Revezamentos Livre	4x100 e 4x200 metros
Revezamento Medley	4x100 metros
Revezamento Misto	4x100 metros Livre e 4x100 metros Medley

Tabela 2: São reconhecidos como Recordes Mundiais, em piscina de 25 metros, as seguintes distâncias e nados para ambos os sexos (site CBDA):

Livre	50, 100, 200, 400, 800 e 1500 metros
Costas	50, 100 e 200 metros
Peito	50, 100 e 200 metros
Borboleta	50, 100 e 200 metros
Medley	200 e 400 metros
Revezamentos Livre	4x50, 4x100 e 4x200 metros
Revezamento Medley	4x50, 4x100 metros
Revezamento Misto	4x50 metros Livre e 4x50 metros Medley

3.1.5 Federação Internacional de Natação (FINA)

A Federação Internacional de Natação (FINA), foi fundada em 19 de julho de 1908 em Londres durante os Jogos Olímpicos de Londres. Alemanha, Bélgica, Dinamarca, Finlândia, França, Grã-Bretanha, Hungria e Suécia foram responsáveis por sua formação. A priori, ela seria responsável por estabelecer regras unificadas para natação, saltos ornamentais e polo aquático, verificar e estabelecer atualizações de Recordes Mundiais e administrar as competições de natações nas Olimpíadas (“FINA | fina.org - Official FINA website”, [s.d.]).

4 PROJETO BÁSICO

4.1 Identificação de Parâmetros

O trabalho visa atender a técnicos de natação de uma forma amigável e rotineira, portanto é necessário entender quais são os parâmetros já observados subjetivamente por eles. Para isso, foram realizadas duas reuniões com profissionais atuantes na área, uma com o ex-técnico da equipe do Hebraica e atual técnico das equipes do SESI Santa Bárbara e dos Wet Rats da Escola Politécnica de São Paulo, Guilherme Giorgi e outra, com o ex-professor da 4fit, ex-técnico dos Wet Rats da Escola Politécnica de São Paulo e da CIEDEF, ex-supervisor da Runner Academias e atual Técnico de natação no Tênis Clube Paulista, André Carvalho.

Como resultado obteve-se os parâmetros do momento inicial da saída, ou seja, enquanto o atleta está parado na baliza esperando pelo sinal do árbitro e do segundo instante, que é o período do voo. Para a fase estática sobre o bloco, ressaltou-se a altura do quadril e a respectiva angulação dos joelhos e a posição da cabeça olhando para baixo para auxiliar no posterior direcionamento da saída. A partir dos primeiros movimentos, mais dados são analisados. Ainda sobre o bloco, é observada a força realizada pelos braços que gera maior impulsão para uma saída mais explosiva, veloz e ampla, movimento dos braços e da cabeça para guiarem o movimento do atleta para frente objetivando o ganho de amplitude da saída e consequente distância da borda. Na parte do voo, observa-se o movimento da perna de trás que, se estiver alta, garantirá um ângulo de entrada na água ideal para que o corpo todo entre em um só ponto. Além disso, a recuperação dos braços e consequente encaixe da cabeça entre eles pretende reduzir arrastos e aprimorar a posição fluidodinâmica.

No Tênis Clube Paulista, pôde-se observar na prática a influência de alguns desses parâmetros, comprovando a direta relação entre o aperfeiçoamento dessas características e a melhora na saída do atleta. A seguir, avaliou-se a influência de parâmetros por meio de filmagens feitas de saídas de três atletas do clube. Nelas deu-se maior enfoque no movimento da cabeça e do pé traseiro. A cabeça olhando para frente conduz o corpo a formar uma reta entre cabeça e quadril paralela com o plano da piscina, de tal forma a aumentar a amplitude devido velocidade horizontal acrescida. O movimento do pé traseiro está relacionado com a posição de entrada na água, se ele estiver alto, o corpo todo irá formar uma linha reta ao entrar na água auxiliando assim na posição fluidodinâmica.

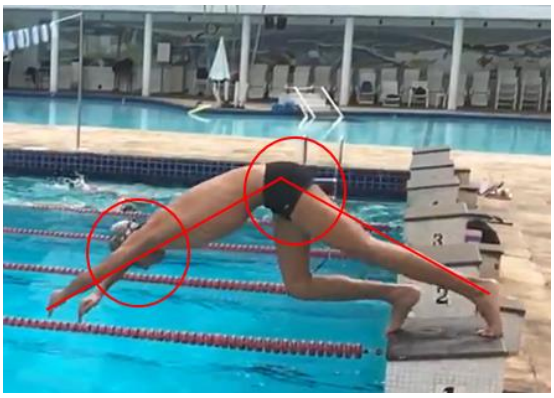
A primeira movimentação no bloco deve ser a da cabeça, que inicia posicionada entre os braços (Figura 10), ela aciona a musculatura das costas e, em conjunto com os braços que garantem explosão, guiarão o movimento da saída do bloco de forma a garantir uma boa condução do movimento para frente e melhorar a amplitude do salto (segunda coluna, Tabela 3).

Figura 10: Correto movimento da cabeça na fase de bloco

No entanto, caso a cabeça não direcione corretamente o corpo (primeira coluna,Tabela 3), não haverá paralelismo entre a reta que liga cabeça e quadril com o plano da água. Dessa forma, a amplitude e distância desejadas não são atingidas. Isso faz com que o atleta entre antes na água.

Tabela 3:Uso da cabeça na condução da saída de um nadador.

Movimento Incorreto	Movimento Correto
	
a - Cabeça olhando para baixo não guia o corpo para a movimentação correta, reduzindo amplitude.	*b – Posicionamento correto da cabeça no início do movimento de saída*

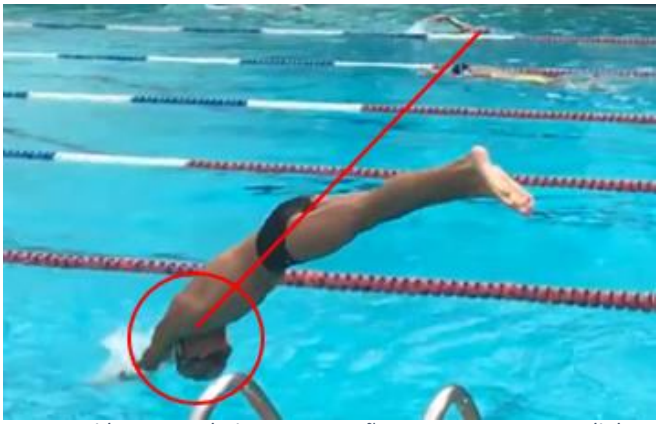


c - Cabeça e quadril não estão alinhados, dessa forma o corpo atingirá antes a água.

d – Posicionamento da cabeça no início do movimento de saída

A angulação da perna de trás para cima, como mostrado na segunda coluna da Tabela 4, garante a entrada retilínea na água em um único ponto. No entanto, é possível analisar o desvio da posição fluido dinâmica ideal na primeira coluna. Com a perna baixa a angulação da entrada na água não foi ideal ficando evidente pela colisão da perna em uma área maior da superfície da água, ou seja, “chapando” as pernas.

Tabela 4:Uso da perna traseira no alinhamento do corpo na saída de um nadador.

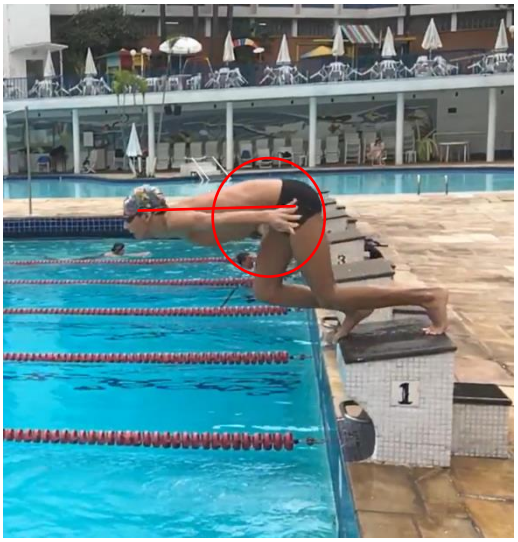
Movimento Incorreto	Movimento Correto
	
a - Perna baixa.	*b – Perna alta.*
	b - . Correto movimento da perna traseira
	
c - Devido a perna baixa, o corpo não entrar em com uma linha reta.	*d – Consequente angul ação do corpo devido ao movimento anterior da perna na saída.*

e - Corpo entra em uma área relativamente grande.

f - Corpo entra sem espalhar muita água, o que evidencia a entrada em único ponto.

Nesse momento, foi solicitado ao atleta para que fizesse a impulsão em conjunto com os braços (Figura 11a) e posterior recuperação dos mesmos pela lateral (Figura 11b). Com esse cenário houve aumento da impulsão do atleta e consequente ganho de amplitude em sua saída.

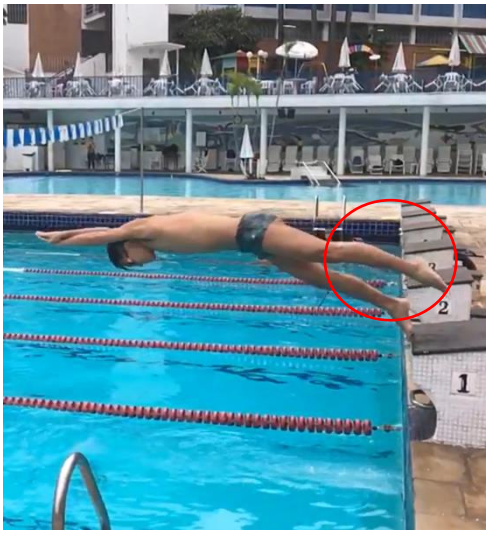
a - Uso dos braços empurrando a baliza.



b - Recuperação pela lateral, cabeça olhando para frente alinhada com o quadril.

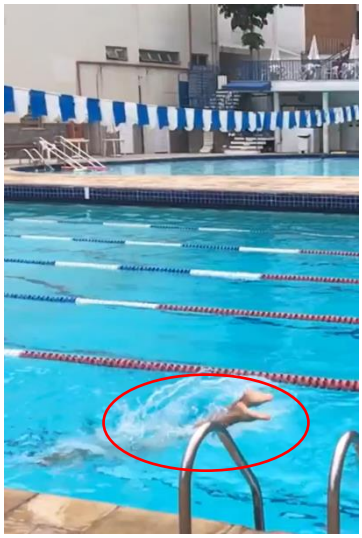
Figura 11: Análise do uso do braço na saída.

O atleta da próxima análise realizou o erro de não movimentar a perna de trás da maneira correta (Figura 12a) e isso implicou em seu desalinhamento corporal (Figura 12b) e consequente entrada com as pernas “chapadas” na água (Figura 12c).



a - Pé traseiro baixo.

b - Como o pé estava baixo, o corpo não está totalmente alinhado com a entrada na água.



c - Devido ao não alinhamento, o atleta "chapou" com a perna.

Figura 12: Angulação baixa da perna

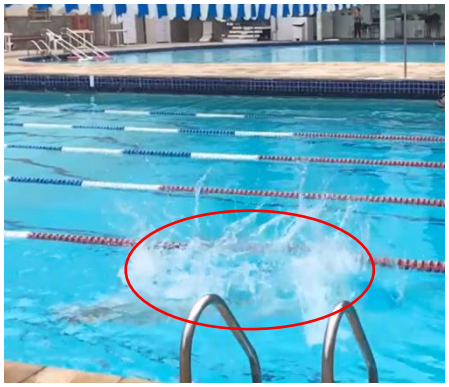
Nessa última análise, o atleta estudado encontra-se na categoria de PcD (Pessoa com Deficiência), ele realizou a correta movimentação da cabeça (Figura 13a) e paralelismo com o plano da piscina, porém, apesar de apresentar membros inferiores mais fortes, não os direciona para cima (Figura 13b), o que evidencia a má aterrissagem (Figura 13c), com corpo “chapando” (Figura 13d) e espalhando muito mais água (Figura 13e) que os casos vistos anteriormente.

a - Condução do corpo correta com o olhar para frente.

b -Devido ao movimento correto da cabeça a linha entre cabeça e quadril está paralela com a linha da água. Pé traseiro baixo.

c -Corpo não forma uma linha única na aterrissagem.

d - Entrada "chapada" na água em diversos pontos.



e - Entrada por diversos pontos evidenciada pela grande quantidade de água espalhada em uma área também maior.

Figura 13: Análise dos movimentos do atleta PcD

Por fim, atrelando as entrevistas ao que foi depreendido com o levantamento bibliográfico pelas leituras do estado da arte (página 18) foi realizada uma compilação das características importantes e que seriam analisadas (Tabela 5).

Tabela 5: Compilação entre entrevistas e artigos.

Artigos	Técnicos	Compilação
Posição da cabeça	Posição da cabeça	Angulação da cabeça
Posição dos Pés na baliza	Altura do quadril	Angulação das pernas
Angulação das pernas	Força exercida pelos braços	Angulação dos braços
Inclinação do corpo	Uso da cabeça na condução	Força exercida pelos braços
Altura do quadril	Altura da perna	
Força exercida pelos braços	Encaixe da cabeça	

Assim, entendeu-se que durante a saída do atleta seria necessário a captura dos parâmetros a seguir:

- Ângulos da cabeça, tronco, pernas e braços
- Força aplicada pelos braços do atleta no bloco

No entanto, neste projeto tem-se como foco a medição dos ângulos da cabeça, tronco, pernas e braços. Além disso, inclui-se como necessidade a filmagem do salto sincronizado com os dados obtidos, para que melhor interpretação dos dados.

Como conclusão da forma como seria apresentada a análise amigavelmente ao técnico, o grupo levantou algumas hipóteses. A primeira delas consiste em demonstrativos pontuais na filmagem com representação conjunta da trajetória do atleta e da ideal para comparações entre eventuais desvios. A outra foi indicar na filmagem o dado do atleta, o desvio e o esperado. Por fim, devido ao sincronismo entre os sensores no mesmo *trigger*. A partir dos movimentos, seriam discriminados, no relatório entregue, os *frames* de interesse, que explicariam as análises apresentadas. Um exemplo de informações contidas em um relatório oferecido aos usuários encontra-se a seguir:

Análise de saída

Atleta: João Pereira

Clube: Poli-Usp Wet Rats

Técnico: Maria da Silva

Data e Horário da Análise: 11/11/2011 10:30:41

1- Angulação da perna

- Se correto:

Posição alta da perna traseira com consequente entrada hidrodinâmica e redução do arrasto.

- Se incorreto:

Uso incorreto da perna traseira, idealmente deveria estar mais alta. Com isso, entrou em uma área maior e gerou mais arrasto.

2- Movimento da cabeça

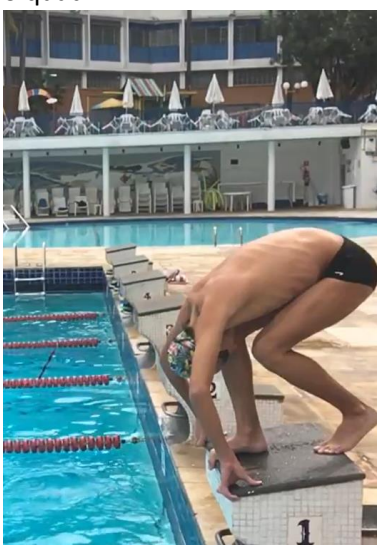
- Se correto:

Cabeça conduz o resto do corpo e projeta o movimento para ganho de amplitude do salto.

- Se incorreto:

Cabeça baixa conduzindo o corpo para uma saída mais curta e entrada adiantada na piscina.

3- Altura do quadril



- Se correto:

Boa altura do quadril que reduz o tempo de reação e aumenta a impulsão.

- Se incorreto:

Quadril baixo torna a saída mais lenta e fraca.

4- Uso dos braços no bloco

- Se correto:

Uso adequado do braço garantindo boa propulsão horizontal

- Se incorreto:

Saída poderia apresentar maior velocidade horizontal ao empurrar o bloco de forma mais acentuada.

- Inclinação:

- Se correto:

Braço para (traz ou centralizado) adequado para saída mais potente e com maior segurança.

- Se incorreto:

Braço para frente, apesar de reduzir o tempo de reação, diminui a potência da saída, bem como torna o movimento instável e, consequentemente, mais suscetível a desequilíbrio e queimar largada.

4.2 Requisitos do Projeto

4.2.1 Requisitos Mecânicos

O dispositivo a ser acoplado no atleta deve ser portátil e confortável para não prejudicar seu desempenho durante as análises, para tanto, controla-se a sua fixação, massa e dimensão. Os valores de referência foram coletados a partir da comparação de produtos semelhantes presentes no mercado, além de dados obtidos na literatura.

A fixação do dispositivo deve ser firme o suficiente de modo a manter o dispositivo no local desejado além de permitir sua fácil alocação e remoção.

Sua massa deve ser pequena para não impactar no rendimento do atleta. Em aplicações semelhantes, as massas das placas variam de 8 a 110g (MAGALHAES et al., 2015). Neste protótipo, o requisito máximo de massa o conjunto eletrônico, bateria e caixa será de 200g.

As dimensões do dispositivo devem pensadas de forma a diminuir sua influência na hidrodinâmica do atleta, para tanto, a principal dimensão a ser controlada deve ser a espessura do dispositivo, em (MAGALHAES et al., 2015), encontra-se projetos semelhantes em que esta dimensão varia de 10mm a 24mm. Assim, para este protótipo, foi considerado um máximo de 30mm.

4.2.2 Requisitos Elétricos

O cálculo da frequência de amostragem remete aos requisitos elétricos. Para tanto, realizou-se um uma análise de como os ângulos da cabeça e das pernas se comportavam no tempo (Figura 14). Para a análise foi utilizado o software de análise de imagens Tracker[4] e o vídeo foi filmado a uma taxa de 250 quadros por segundo.

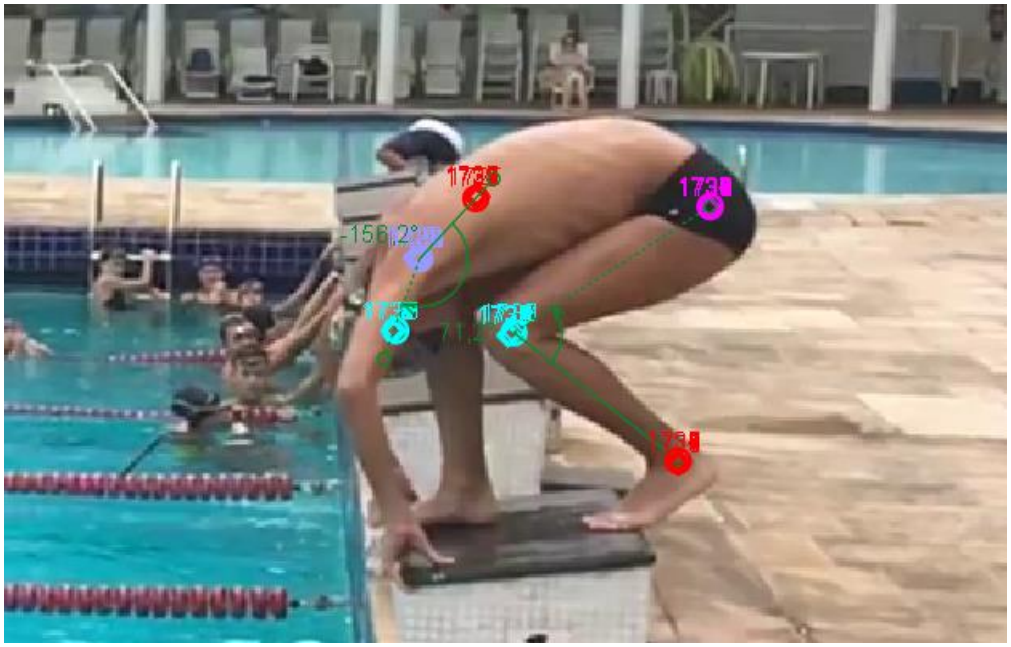


Figura 14: Análise de imagem pelo Tracker

[4] https://physlets.org/tracker/

O gráfico da Figura 15 mostra a angulação da cabeça em relação ao tronco no tempo. O movimento da cabeça pode ser divido em três partes: a primeira é o início do movimento até a flexão máxima da cabeça, o segundo é a recuperação e o terceiro é o encaixe da cabeça para a entrada na água. A duração total da fase de bloco mais a fase de vôo é de 1,263s. Cada parte do movimento e sua duração é mostrado na Tabela 6.

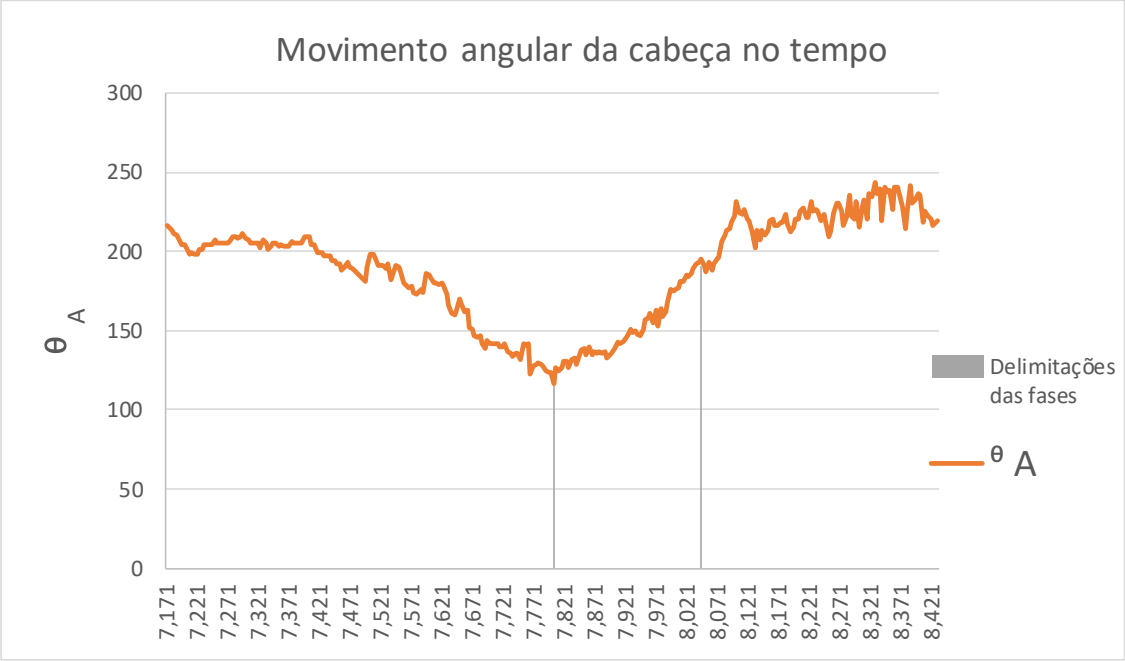


Figura 15: Angulação da cabeça em relação ao tempo

Para a angulação da perna, mostrada na Figura 16, foi possível identificas dois movimentos, um de extensão da perna e outro de manutenção até a entrada na água. O tempo de cada movimento também é mostrado na Tabela 6.

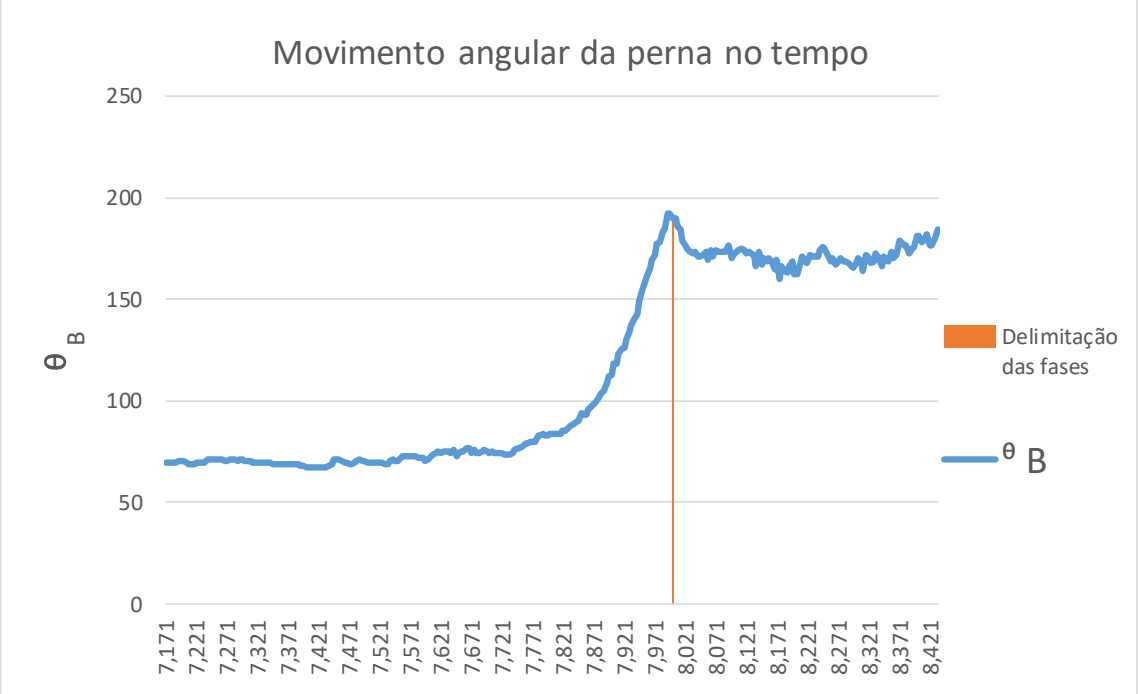


Figura 16: Angulação da perna em relação ao tempo

Aplicando o teorema de Nyquist que diz que a frequência de amostragem deve ser pelo menos duas vezes a maior frequência dos sistemas. Tem-se que o menor movimento a ser captado é o de recuperação que levas 0,242s. Logo, para não perder informação, deve haver um período de amostragem de 0,121s ou menor. Foi escolhido um período 10x menor para que haja ter uma segurança. Logo a frequência de amostragem deve ser pelo menos 83Hz.

Tabela 6: Tempos de cada fase dos movimentos da cabeça de da perna e frequência de amostragem necessária para cada um deles.

Movimentos	Etapas	ti (s)	tf (s)	Duração tf-ti (s)	Período por Nyquist (Evita Aliasing) (s)	Após o Coeficiente de segurança (s)	Frequencia (Hz)
Cabeça	Projeção	7,204	7,804	0,6	0,3	0,030	33,333
	Recuperação	7,804	8,046	0,242	0,121	0,012	82,645
	Encaixe	8,046	8,433	0,387	0,1935	0,019	51,680
Perna	Extensão	7,171	8	0,829	0,4145	0,041	24,125
	Manutenção	7,862	8,433	0,571	0,2855	0,029	35,026

Realizando a análise utilizando o software Matlab, transformando o movimento para o espectro das frequências, chega-se no resultado da Figura 17 e Figura 18.

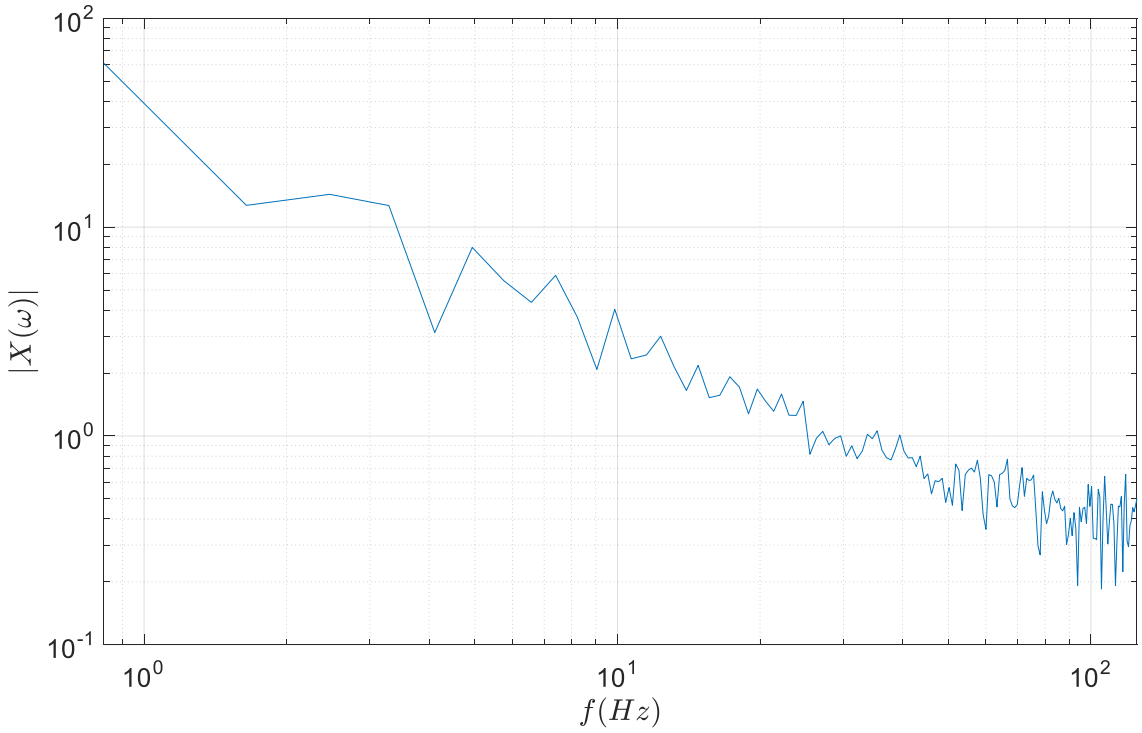


Figura 17: Espectro de frequência do movimento da perna

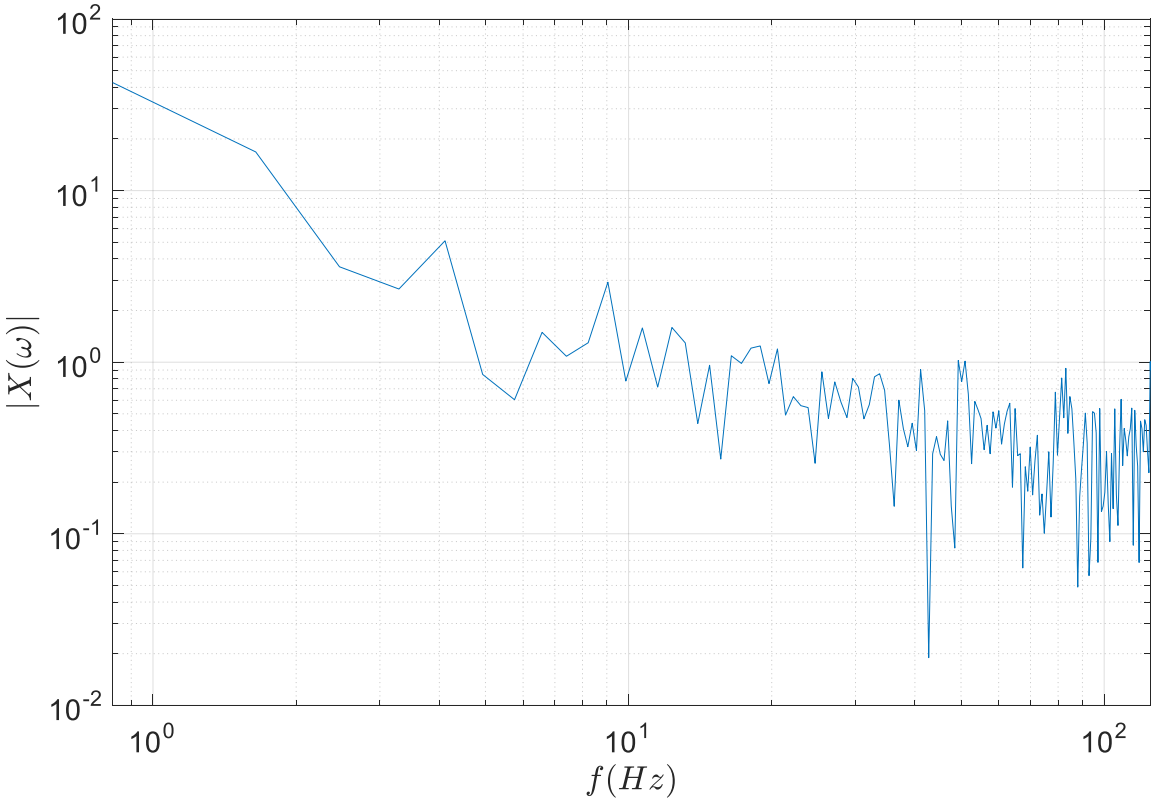


Figura 18: Espectro de frequências do movimento da cabeça

Vê-se, nos dois casos, que frequências a partir de 10Hz possuem amplitudes desprezíveis, desta forma, a frequência de amostragem deve ser no mínimo 20Hz. Aplicando um coeficiente de segurança de 4, a frequência mínima é de 80Hz.

4.2.3 Requisitos Funcionais

Em relação ao seu funcionamento o sistema deve cumprir os seguintes requisitos.

- Da parte da instrumentação:
  - Medir a angulação da cabeça, tronco, pernas e braços
  - Ser pequena a fim de não interferir no movimento do atleta
  - Sincronizar o início da coleta de dados e filmagem
- O programa deve:
  - Apresentar um módulo de captura de dados
  - Apresentar um módulo de apresentação de dados

4.2.4 Não funcionais

Dos requisitos não funcionais, vem:

- O programa deve ser amigável para os técnicos e para os alunos
- O programa deve entregar o relatório pouco tempo depois de o atleta completar seu percurso.

4.3 Análise Mecatrônica

Com base nas medições necessárias obtidas na seção 4.1. Concluiu-se que o sistema deve possuir 2 módulos. O primeiro módulo consiste no módulo de obtenção dos ângulos de interesse dos membros do corpo do atleta e o segundo módulo, consiste em um sistema com câmera para filmagem da saída do atleta

A Figura 19 mostra cada componente de cada módulo, bem como qual parâmetro é passado de componente a componente do sistema.

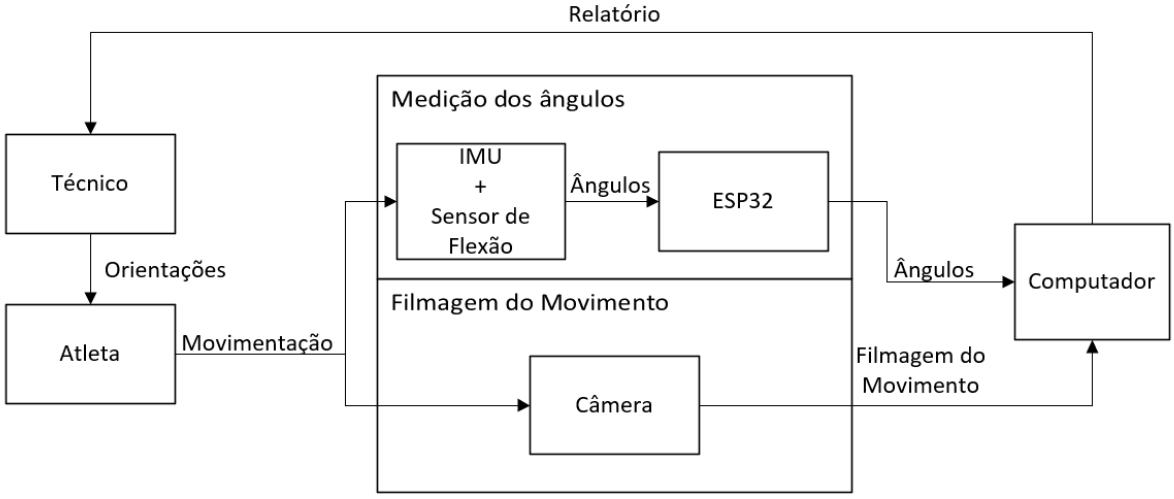


Figura 19: Componentes de cada módulo e parâmetros passados entre eles

A seguir será mostrado as opções consideradas de soluções para cada módulo bem como a escolha dos materiais que o compõe.

4.3.1 Medição dos Ângulos de Interesse.

A medição dos ângulos necessários pode ser feita de várias maneiras. Como mostrado no estado da arte, os meios mais utilizados são câmeras e unidades de medições inerciais. No entanto existem outras maneiras como goniômetros e sensores de flexão.

Sistemas com múltiplas câmeras oferecem alta precisão espacial tridimensional, no entanto, para determinar a variabilidade do comportamento natatório, é necessária a captação de um grande número de ciclos. Isso é muito difícil nesses sistemas, pois registram apenas comportamentos em espaços restritos. Soma-se às essas desvantagens o fato de que digitalizações manuais das dinâmicas do movimento, como durações de saída, virada e chegada, distância percorrida por ciclo, frequência de nado em ciclos por minuto e velocidade média das braçadas são propensas a erros e o processamento de dados é muito longo (GUIGNARD et al., 2017).

As limitações das câmeras incentivaram o surgimento de novas formas de análise como o acelerômetro que capta a dinâmica e o giroscópio que mede velocidades angulares. Logo, um equipamento que conseguisse utilizar-se de ambos teria uma vantagem significativa sobre sistemas de câmeras até então utilizados. As IMUs, que apresentam esses dois sensores, permitem análise dos parâmetros e variabilidade da dinâmica (GUIGNARD et al., 2017).

A análise dinâmica de coordenação e sua variabilidade funcional fornece informações sobre os processos pelos quais nadadores adaptam-se a mudanças imprevisível contínuas no ambiente aquático. Há ainda outras vantagens da IMU, como após vedadas contra a água, elas podem capturar um grande volume de dados de um treino inteiro de um atleta (GUIGNARD et al., 2017). Os resultados são rapidamente disponibilizados para análise pois não precisam de procedimentos de digitalização(GUIGNARD et al., 2017). Não há interferência entre duas quando dois nadadores próximos estiverem utilizando-as, apresentam baixo custo e são portáveis e leves para fácil uso em ambientes externos (PANSIOT; LO; YANG, 2010). Diferentemente, as câmeras são mais caras e ficam sujeitas às intempéries do ambiente, apresentando dificuldades da detecção de pontos devido a água, respingos, variação de iluminação e colorações.

Portanto, a utilização de câmeras, a identificação de pontos de interesse e análise automática é dificultada, pois é dependente de condições ambientais, como iluminação e respingos, não possibilitando uma análise rápida ou em tempo real. Além disso, a utilização de câmeras traz uma limitação espacial, sendo necessário que a câmera acompanhe o atleta ou que a análise seja feita apenas no quadro da câmera.Os sensores inerciais, por outro lado, não apresentam os problemas identificados com câmeras, permitindo seu uso em aplicações em tempo real ou análise rápidas (GUIGNARD et al., 2017). Além disso, são mais baratas que câmeras de alta velocidade.

Os goniômetros elétricos, normalmente utilizados em ambientes fisioterapêuticos, possuem variações quanto ao princípio de funcionamento, podendo

ser por potenciômetros, flexíveis[5] ou por fibra ótica (CAMPIGLIO; MAZZEO; RODRIGUEZ, 2013). Para esta aplicação, apenas os potenciômetros flexíveis e por fibra ótica (DONNO et al., 2008) seriam possíveis de serem utilizados, pois os que utilizam potenciômetros necessitam de barras rígidas, o que pode prejudicar a livre movimentação do atleta.

Outra opção de utilização são sensores de flexão como o Flex Sensor[6] que permite a medição em uma dimensão de ângulos.

Neste trabalho, para o módulo de medição de ângulos, serão utilizados uma IMU e um sensor de flexão. A escolha da IMU se deu por conta de seu pequeno tamanho e massa, permitindo portabilidade. A razão pela escolha dos sensores de flexão se deu por ser uma opção de baixo custo em relação às outras opções e por ser suficiente para a aplicação.

4.3.2 Módulo de Filmagem da saída do atleta

A filmagem da saída do atleta deve estar em sincronia com a coleta de dados dos sensores. Para tanto decidiu-se controlar o acionamento da câmera através de um microcontrolador.

Outras soluções como acionamento via Bluetooth e sincronismo através de análise de imagem utilizando um LED em canto do quadro foram consideradas. No entanto, escolheu-se o microcontrolador por este possuir a funcionalidade de recepção de *broadcasting.* Este tipo de comunicação será explicado na seção 5.3

[5] http://www.biometricsltd.com/goniometer.htm
[6] https://www.sparkfun.com/products/8606

5 IMPLEMENTAÇÃO

5.1 Microcontrolador

Para a leitura e pré-processamento dos dados obtidos pelos sensores foi escolhido o microcontrolador ESP32[7]. Este microcontrolador apresenta Wi-Fi e Bluetooth integrados, sem a necessidade de módulos extras, o que o torna mais leve e compacto, importantes para atenderem aos requisitos de portabilidade e de massa.

5.2 Módulo de medição dos ângulos dos membros do atleta

Este módulo de medição tem como componentes principais uma IMU, um sensor de flexão e um microcontrolador ESP32.

Dentre as várias IMUs existentes no mercado, neste projeto utilizou-se a BNO055 da marca Bosch, que apresenta uma frequência de amostragem de 100Hz, satisfazendo o requisito. Além disso ela possui um filtro integrado, dispensando a necessidade de circuitos externos, tornando-a compacta.

Para a alimentação do módulo, serão utilizadas 3 baterias AA em série, para obter uma tensão nominal de saída de 4.5V, fornecendo uma carga de aproximadamente 2.9mAh. Desta forma, para um circuito de que consome cerca de 100mA, a duração da bateria será de 15h, conforme consta no *datasheet*. Este tipo de bateria foi escolhido por ser de fácil acesso, podendo ser encontrada em supermercados e papelarias por exemplo.

5.2.1 Circuito elétrico

O circuito elétrico do módulo encontra-se em anexo (página 143).

5.2.2 Programa embarcado

O programa utilizado no microcontrolador implementa a máquina de estados mostrada na Figura 20.

Ao ligar o dispositivo, a máquina fica aguardando por um sinal Bluetooth, podendo ser para calibração ou para o início da coleta de dados.

Para a calibração, a IMU utiliza um algoritmo próprio, que checa constantemente se ela está calibrada, assim, é feita a leitura do status de calibração, caso ela não esteja calibrada, a rotina de calibração deve ser realizada. No caso dos sensores de flexão, a rotina de calibração é feita realizando a medição do sensor em ângulos específicos, uma vez que a rotina é realizada, o sensor está pronto para uso.

Na coleta de dados, os dados provenientes da IMU são recebidos através de uma comunicação I2C a 100Hz. A leitura do sensor de flexão é feita através de um conversor AD também a 100Hz, após a coleta dos dados do sensor estes são filtrados por um Filtro de Kalman.

A descrição das ações executadas pelo microcontrolador estão listadas na Tabela 7.

[7] http://esp32.net/

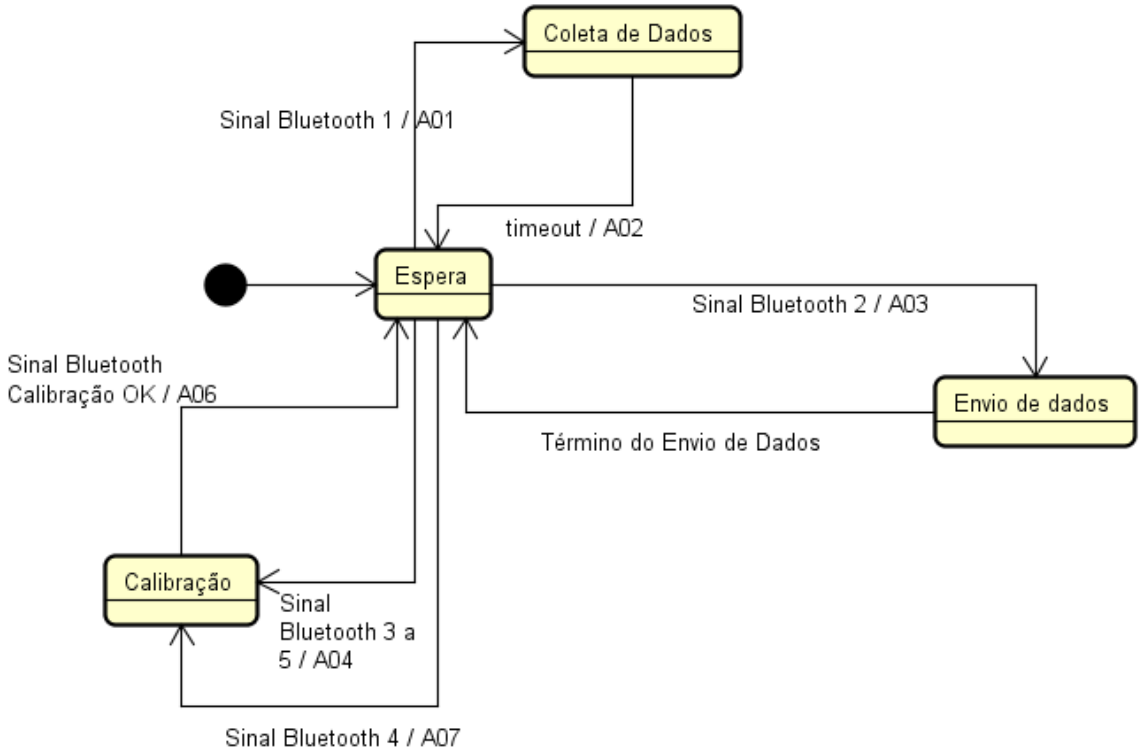


Figura 20: Máquina de estados implementada no microcontrolador

Tabela 7: Ações executadas no microcontrolador

A01	Iniciar coleta de dados / Liga LED amarelo
A02	Terminar coleta de dados / Desliga LED amarelo
A03	Iniciar envio de dados
A04	Calibração do sensor de flexão
A05	Calibração da IMU
A06	Encerramento da calibração/ Acende o LED amarelo por 3 segundos

5.2.3 Construção Mecânica e Isolamento contra água

A construção mecânica tem como objetivo a impermeabilização contra água e fixação dos módulos de medição dos ângulos.

i. Encapsulamento e impermeabilização contra água

Para o encapsulamento das placas utilizou-se caixas da marca Hammond Manufacturing[8]. Foram feitos furos para a passagem dos cabos dos sensores de flexão e do *reed switch* (interruptor magnético) e dos LED's para verificação de status. Realizou-se a impermeabilização com a aplicação de silicone em todos os furos e encaixes. A Figura 21 mostra o encapsulamento das placas.

a. Módulo alocado com seus componentes devidamente passados pelos furos.

b. Caixa fechada e já com o silicone passado em todas as partes necessárias.

Figura 21: Encapsulamento das eletrônicas

Para o sensor de flexão o encapsulamento foi feito com papel adesivo, também utilizou-se silicone para vedação nos conectores do sensor. A Figura 22 mostra o encapsulamento.

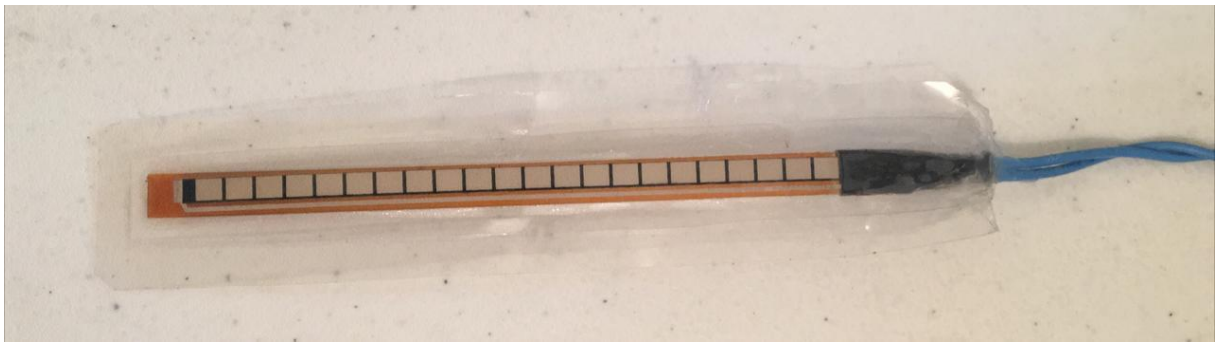


Figura 22: Encapsulamento do Sensor de Flexão

[8] https://www.hammfg.com/part/RL6015

ii. Fixação das caixas e sensor de flexão

A fixação dos sensores de flexão foi feita utilizando fitas adesivas dupla face, junto com um mecanismo que apoia o apoia. O mecanismo é composto de duas tiras plásticas que são unidas apenas pelas laterais (Figura 23a) de forma a deixar o centro livre para a passagem do sensor flexor. Ao esticar e flexionar a articulação a que foi fixado, o sensor pode seguir o movimento sem alteração de sua posição. Note na Figura 23b, Figura 23c que o sensor tem seu contato com o mecanismo em pontos diferentes quando a perna está esticada ou flexionada.

a. Fixadores *b. Joelho esticado* *c. Joelho flexonado*

Figura 23: Fixação dos flexores.

Já as caixas das placas foram fixadas ao corpo do atleta por meio de braçadeiras e colete, como mostra a Figura 24. Para a fixação das caixas às braçadeiras e ao colete foram utilizados fixadores de contato, mostrado na Figura 25.

a. Perna *b. Braco* *c. Nuca*

Figura 24: Fixação das caixas aos membros do atleta

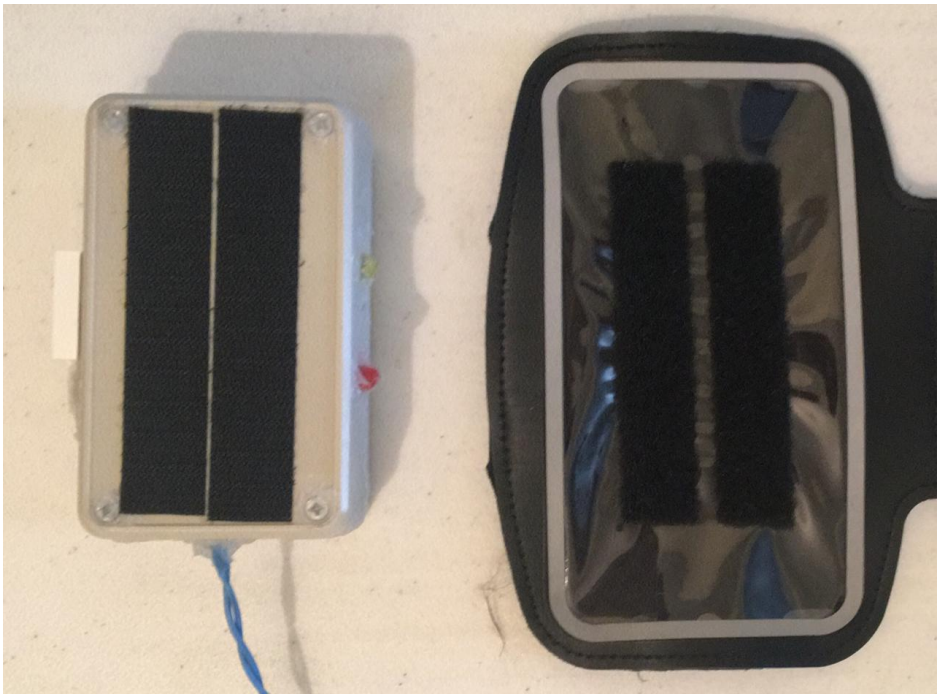


Figura 25: Fixação das caixas nas braçadeiras

5.3 Módulo de captura de vídeo

Para acionar a câmera para a captura do vídeo da saída do atleta foi utilizado também o microcontrolador ESP32. O circuito deste módulo é mostrado na Figura 26.

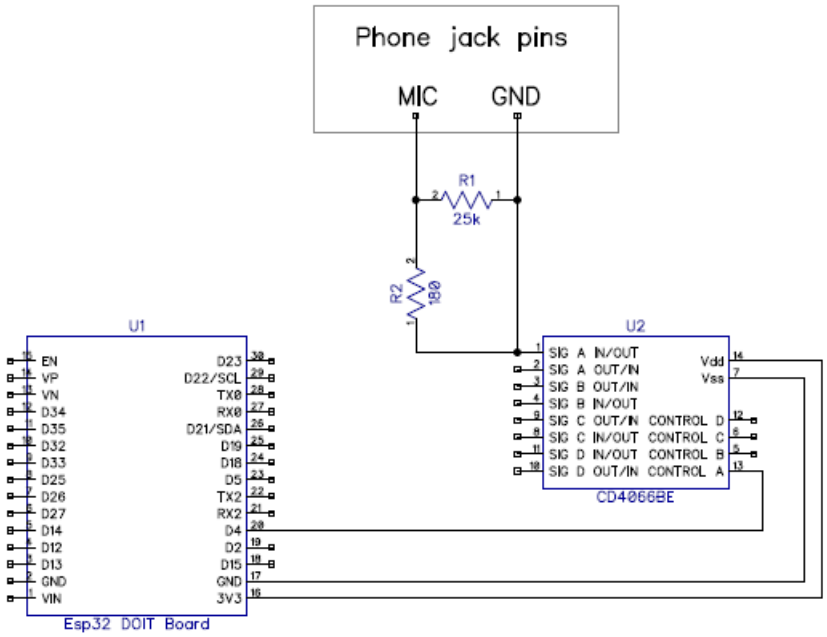


Figura 26: Circuito para acionamento da câmera

Ao conectar o plugue de fone de ouvido no celular, com este circuito foi simulado o clique no botão de Aumentar Volume do fone de ouvido, que quando apertado com o aplicativo de câmera aberto realiza o disparo da câmera.

A Figura 27 mostra este módulo.

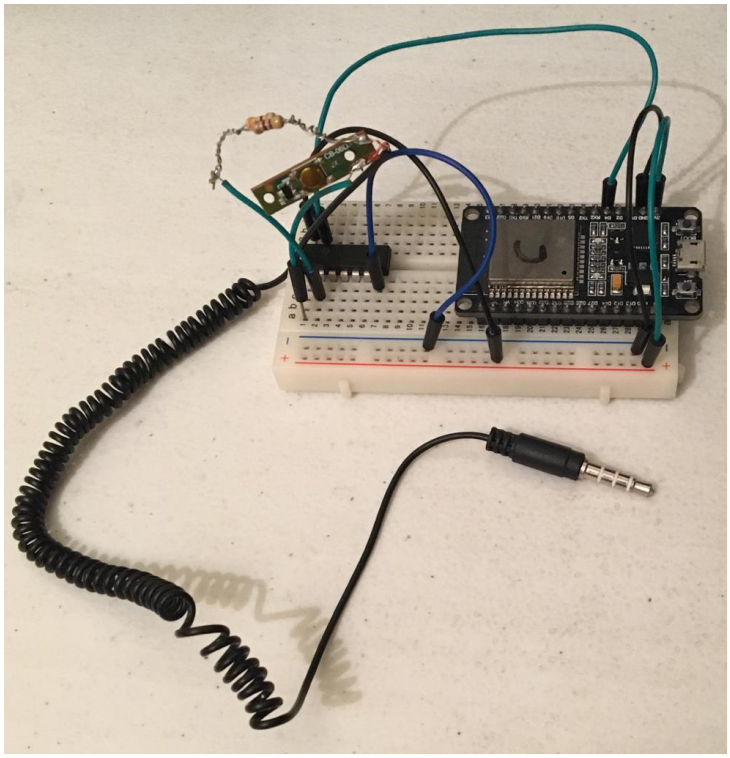


Figura 27: Módulo de disparo da câmera via plug P3.

5.4 Sincronização do acionamento dos módulos

Para realizar a comunicação entre os microcontroladores foi utilizada o protocolo de comunicação ESP-NOW da Espressif, ela foi escolhida por permitir o *broadcasting* do envio de pacotes, fazendo com que todos os integrantes da rede recebam o pacote.

5.4.1 ESP-NOW

O protocolo de comunicação ESP-NOW [9] é um tipo simplificado da conexão sem fio Wi-Fi criado pela Espressif, que trabalha na frequência de 2.4GHz. Ele permite o envio de um pacote de até 250 bytes a cada envio e *broadcasting* quando enviado para o endereço MAC 0xff:0xff:0xff:0xff:0xff:0xff.

A implementação do protocolo consta no código do programa em anexo (página 112).

[9] https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/wifi/esp_now.html

5.4.2 Comunicação entre os módulos

O fluxo de acionamento para a coleta de dados é mostrado na Figura 28. Após receber o sinal de coleta de dados do computador o mestre envia um broadcast (endereço MAC 0xff:0xff:0xff:0xff:0xff:0xff) com o comando para iniciar a coleta dos dados para as outras placas.

Na câmera, ao receber este comando, o microcontrolador aciona o disparo de filmagem. Nos 2 Slaves, a coleta de dados é começada.

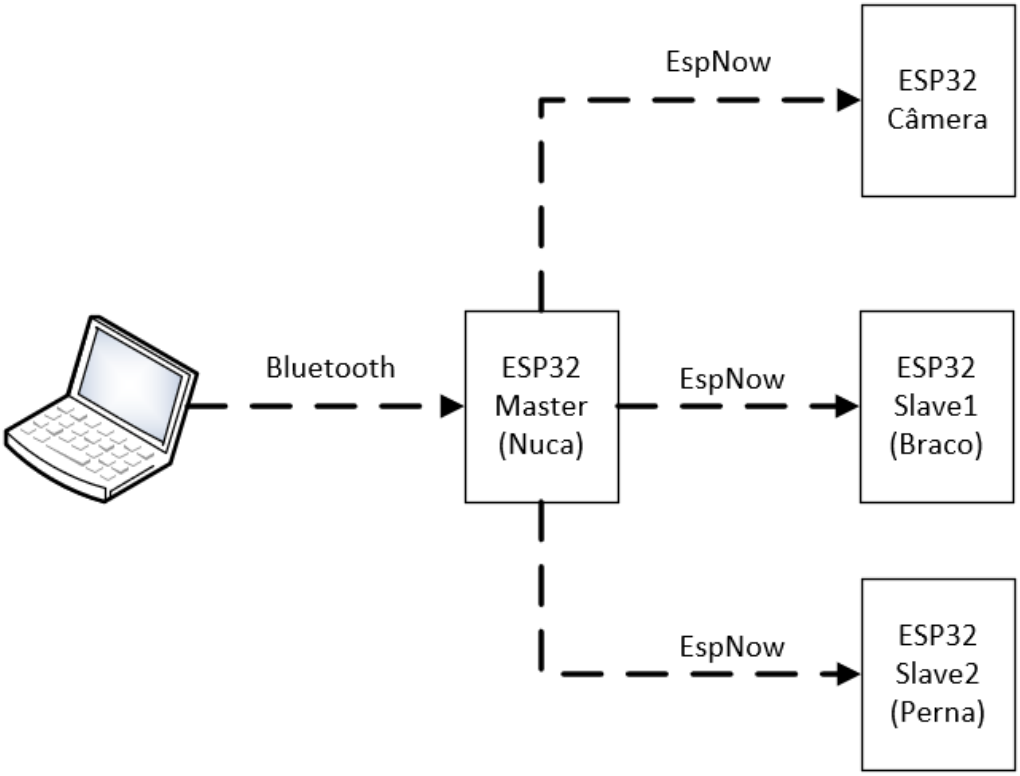


Figura 28: Comunicação entre as placas para sincronização do acionamento

5.5 Concepção de software – Interface Gráfica

O programa a ser instalado no computador do técnico será responsável por mostrar os dados de forma amigável e auxiliar o técnico a fazer uma análise objetiva da saída do atleta.

5.5.1 Fluxograma

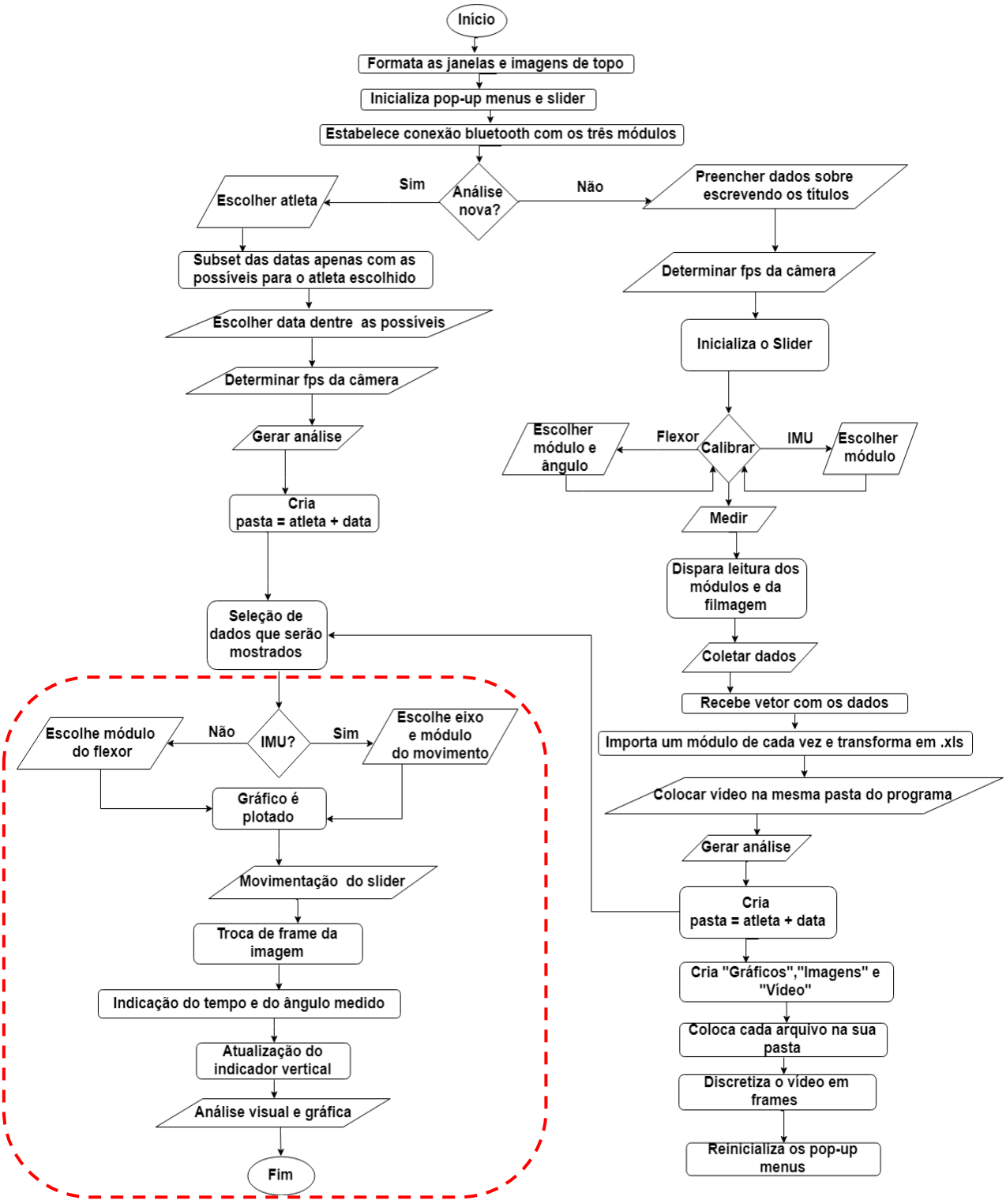


Figura 29: Fluxograma do software com a visualização dos resultados em destaque.

5.5.2 Detalhamento

Foi utilizado o software MATLAB para programar a interface amigável com os usuários: técnico e atleta. Inicialmente, foi necessário criar um banco de dados para armazenar informações de cada medição. Para isso, utilizou-se de um arquivo do tipo

.xls com os cabeçalhos com a descrição de cada informação requerida. Elas compreendem características do atleta: Id (concatenação do nome com o atleta), nome, data, nado, equipe e técnico. Como resultado, obteve-se o arquivo Banco de Dados.xls (Figura 30).

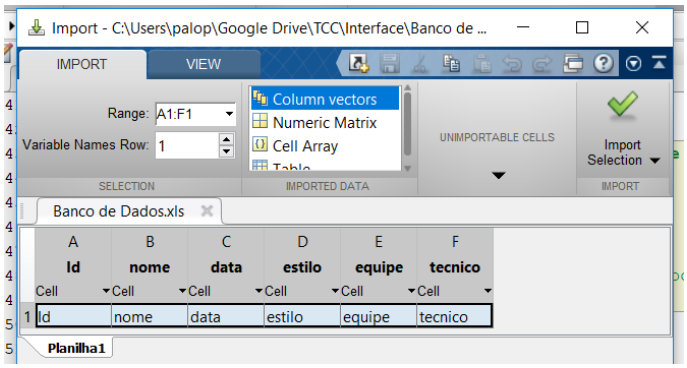


Figura 30: Criação da planilha 'Banco de Dados.xls' apenas com o cabeçalho.

Tendo posse do banco de dados, para acrescentar informações nele, foi criado um Ambiente de Desenvolvimento de Interface Gráfica para Usuário (GUIDE, em inglês) também escrito em MATLAB (Figura 31).

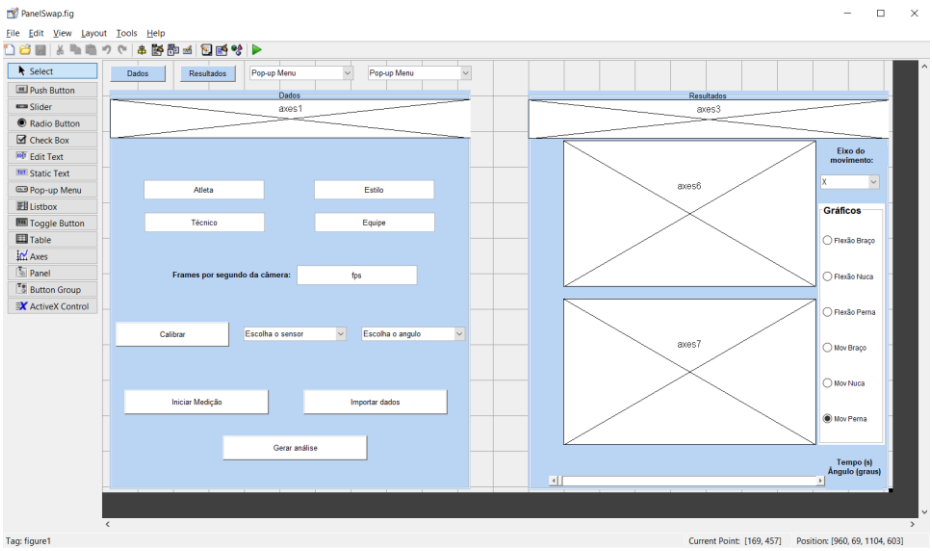


Figura 31: Ambiente de desenvolvimento GUIDE do MATLAB.

Para controlar o hardware pelo software, os comandos são executados via serial do Bluetooth. Para tanto, foi estabelecida a conexão via Bluetooth entre PC e os três módulos logo no início do programa. Na página inicial “Dados” (Figura 34), é possível escolher entre duas formas de proceder na visualização dos dados. A primeira é por meio de análises anteriores escolhendo um atleta que esteja no banco de dados (Figura 32) e em seguida a data, organizada em ‘d_MMM_y hh-mm-ss’ (Figura 33), por fim, ao clicar em “Gerar análise” os dados ficam disponíveis na aba “Resultados” (Figura 35).

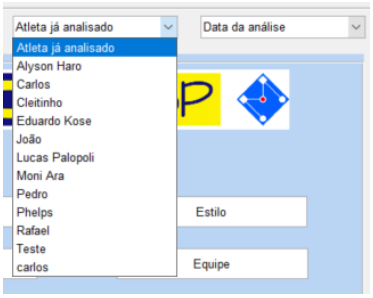


Figura 32: Opções de atletas analisados anteriormente

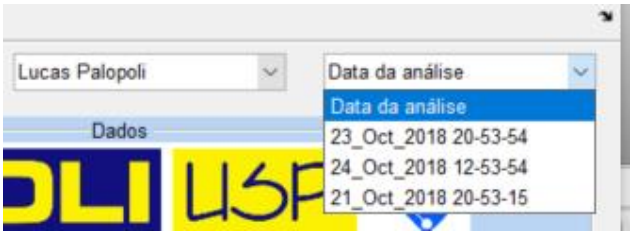


Figura 33: Opções de datas para o atleta selecionado

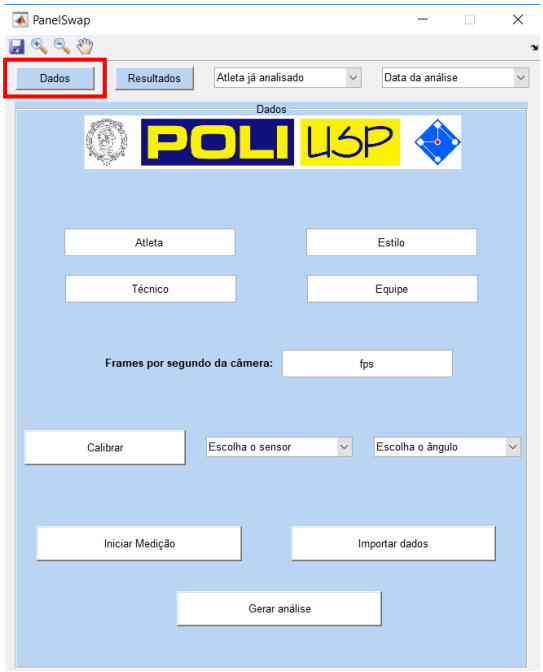


Figura 34: Aba "Dados"

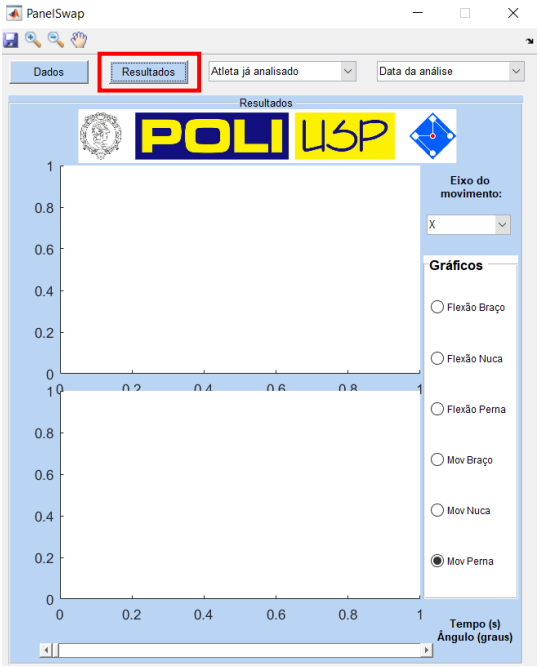


Figura 35: Aba "Resultados"

A segunda é por meio de nova verificação, para tanto, inclui-se as informações que irão ao banco de dados nos espaços indicados. Além de acrescentar essas características, é possível realizar as calibrações escolhendo-se entre IMU ou sensor de flexão (Figura 36), neste escolhe-se também o ângulo a ser calibrado - 45º, 90º, 135º e 180º (Figura 37) .

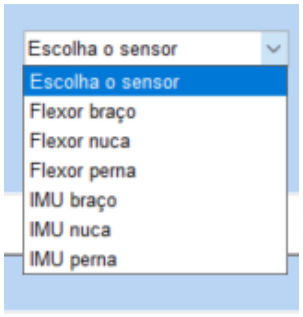


Figura 36: Escolha do sensor a ser calibrado

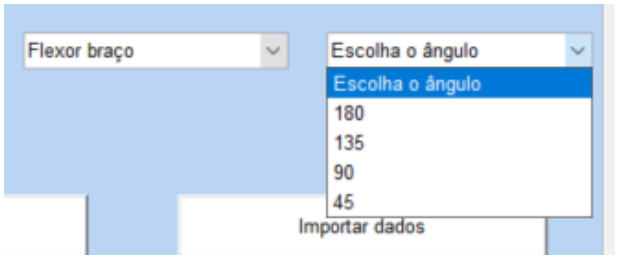


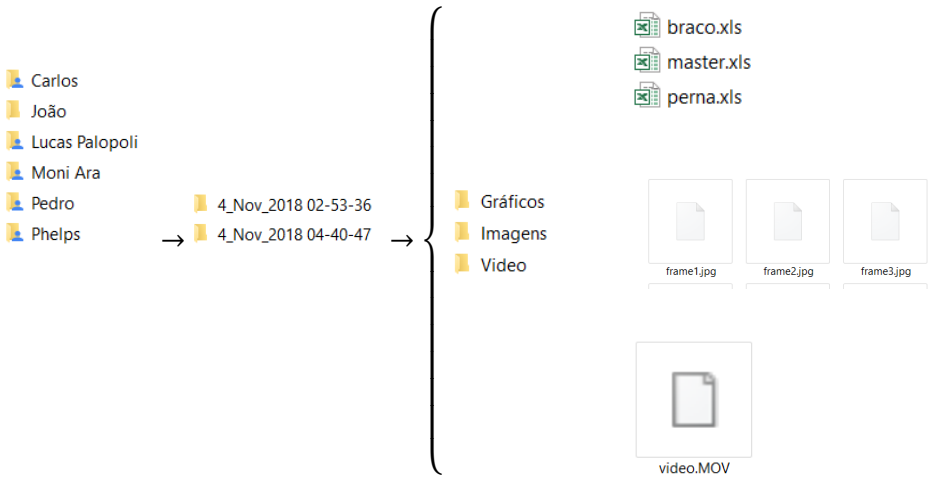
Figura 37: Escolha do ângulo do flexor a ser calibrado

Depois é iniciada a medição, com isso, os três módulos passam a funcionar juntos armazenando suas variações angulares. Após o tempo necessário para isso, é necessário manualmente transferir o vídeo feito pelo celular para o computador e colocá-lo na mesma pasta onde está rodando o programa “PanelSawp”. Os três módulos geram um arquivo que será importado como tipo .xls cada um contendo 5 colunas: tempo, medidas do sensor de flexão, IMU eixo x, IMU eixo y, IMU eixo z. Para transferir os dados coletados pelos módulos é necessário clicar no botão “Importar dados” e esperar até que todas as barras de progresso sejam preenchidas e o aviso de importação com sucesso seja mostrado na tela. A organização das informações se dá por meio de pastas criadas pelo próprio software com a função do botão “Gerar análise”. O vídeo gerado pelo celular deve ser renomeado para “video.MOV” e depois terá seus frames discretizados e alocados na pasta “Imagens”.

A organização da medição ocorre da seguinte forma:

$$Atleta1 \rightarrow Data1 \rightarrow \begin{cases} Gráficos \rightarrow \begin{cases} braco.xls \\ master.xls \\ perna.xls \end{cases} \\ Imagens \rightarrow frameX.jpg \\ Vídeo \rightarrow video.MOV \end{cases}$$

Ou ainda:



Após a saída, os dados são importados para o computador e é possível analisá-la clicando em “Analisar”. Os comandos de calibração, início de medição, importação de dados e geração de análise foram feitos por meio de comando serial via Bluetooth com as ESP32.

Para os dois casos, é necessário selecionar a frequência de aquisição da câmera utilizada para gerar o vídeo em fps.

A fim de auxiliar na compreensão dos passos, barras de progresso foram colocadas para indicar etapas do processo, bem como seu andamento. Caso alguma tarefa seja executada em momento inoportuno, há alertas de bloqueio da ação.

Na segunda janela do programa (Figura 35), é possível visualizar os resultados. Ela é composta por:

- Um “Pop-up Menu” para seleção da coordenada (x,y,z) da IMU a ser analisada;
- Um “Button Group” com seis “Radio Button” para seleção do sensor que será analisado graficamente;
- Um “Slider” para variação do tempo da filmagem;
- Um “Static Text” para mostrar em qual tempo em segundos o slider está indicando e qual o ângulo em graus o gráfico atingiu naquele momento;
- Dois “Axes” que variam de acordo com o “Slider”, o de cima mostra o frame de acordo com o momento e o de baixo plota o gráfico escolhido no “Button Group” bem como apresenta uma linha vertical vermelha para indicação do momento exato que é mostrado na imagem.

Para facilitar as observações e permitir conclusões estruturadas, na barra de ferramentas, foram colocados alguns auxílios para a compreensão dos gráficos e das imagens, como lupa para “zoom in” e “zoom out” e botão “mão” para arrastar (Figura 38).

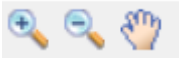


Figura 38: Ferramentas de auxílio na visualização das análises.

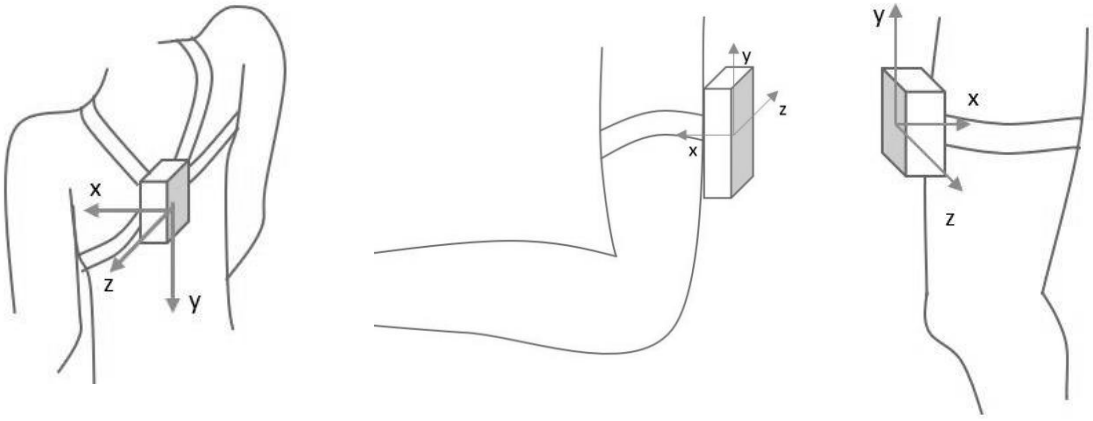
5.5.3 Código fonte do GUIDE

O script com as funções encontra-se em anexo (página 114). Parte dos comentários presentes no início das funções do código são gerados automaticamente pelo MATLAB durante a criação da interface gráfica. Os demais foram realizados com o intuito de explicar comandos e mencionar as fontes de onde foram aprendidos.

6 RESULTADOS

6.1 Orientações dos sistemas de coordenadas das IMUs

A Figura 39 mostra a orientação dos sistemas e coordenadas das IMUs em cada membro. Foram utilizados três conjuntos, um localizados nas costas, na altura da escápula (Figura 39a), outra na parte traseira do braço, próximo ao tríceps (Figura 39b), e última na parte frontal da coxa (Figura 39c).



a. Orientação do sistema de coordenadas da IMU localizada nas costas

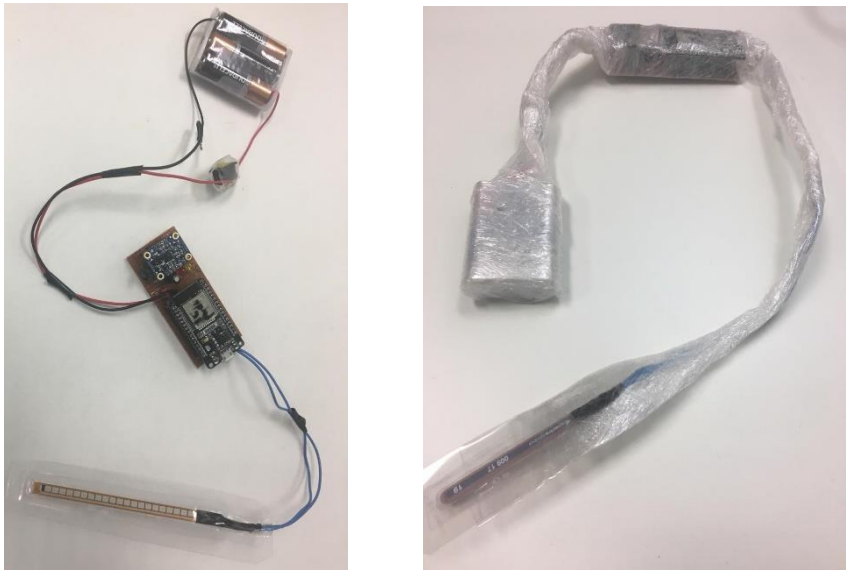
b. Sistema de coordenadas da IMU localizada na parte traseira do braço

c. Orientação do sistema de coordenadas da IMU localizada na parte frontal da coxa

Figura 39: Orientação dos sistemas de coordenadas das IMUs

6.2 Teste a seco

O teste a seco foi realizado ainda com as eletrônicas fora da caixa apenas envoltos em papel filme, como mostra a Figura 40.



a. Módulo esmaltado e devidamente soldado.

b. Módulo envolto no papel filme.

Figura 40: Módulo de medição dos ângulos no teste a seco

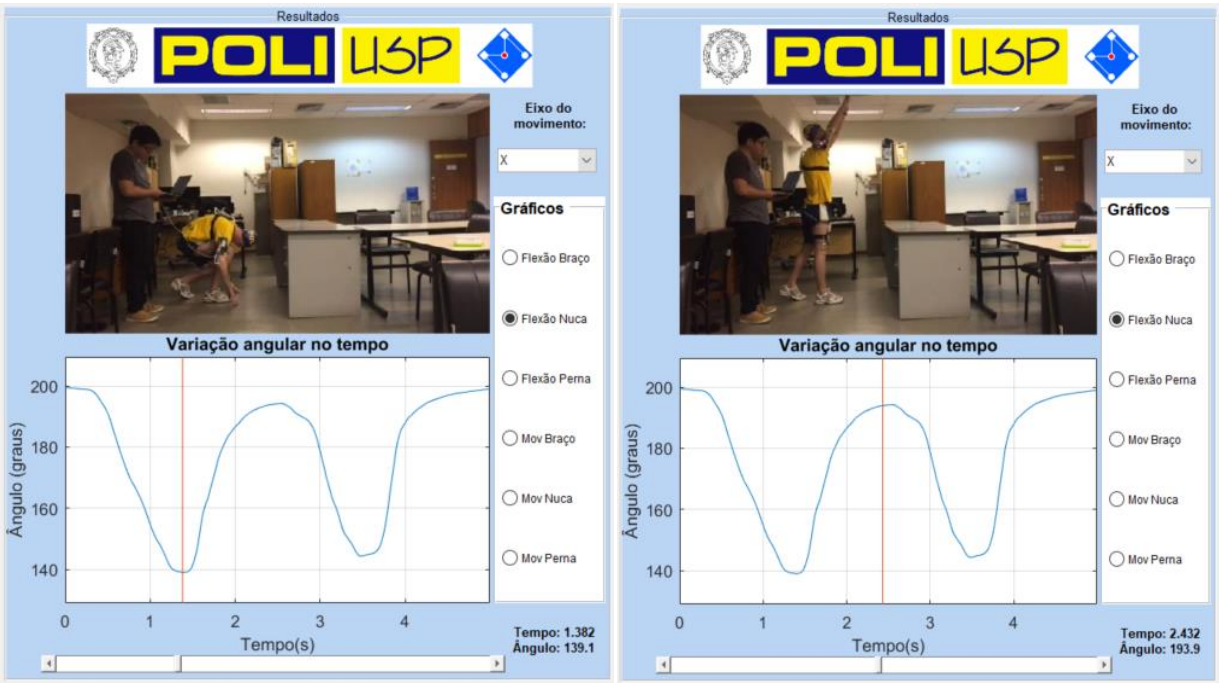
Como o projeto tem por objetivo mapear a saída de um atleta, foi proposto realizar dois ciclos de posição de saída com agachamento e recolhimento dos braços junto aos pés seguido da extensão do corpo em *streamline*, ou seja, com os membros esticados e mão justapostas sobre a cabeça. Essa é a mesma posição que os nadadores usam no final da fase aérea da saída e na submersa para reduzir o arrasto pela diminuição da área que atravessa a água (Figura 41).

Figura 41: Posição de stramline no final da fase aérea da saída.

A seguir estão os resultados obtidos dos sensores de flexão de todos os membros instrumentados e os resultados dos eixos em que houveram maior variação nas IMU's durante o teste a seco.

6.2.1 Sensor de Flexão da Nuca

A Figura 42 mostra os dados coletados pelo sensor de flexão da nuca. Os mínimos locais acontecem quando a flexão é no sentido da cabeça para o peito e os máximos locais para as costas, olhando para cima.



a. Posição de saída *b. Corpo em posição de streamline*

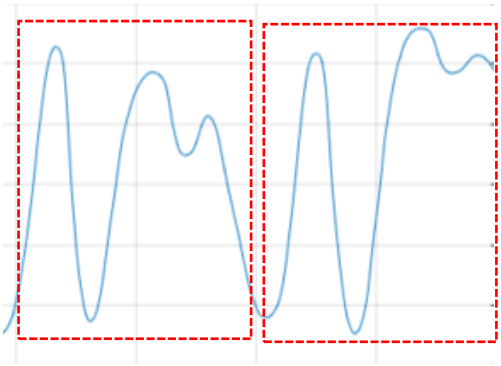
Figura 42: Gráficos da medição do sensor de flexão fixado na nuca

Pode-se ver que os dados acompanham o movimento do atleta, o primeiro mínimo local é atingido quando o atleta prepara a posição de saída (Figura 42a),

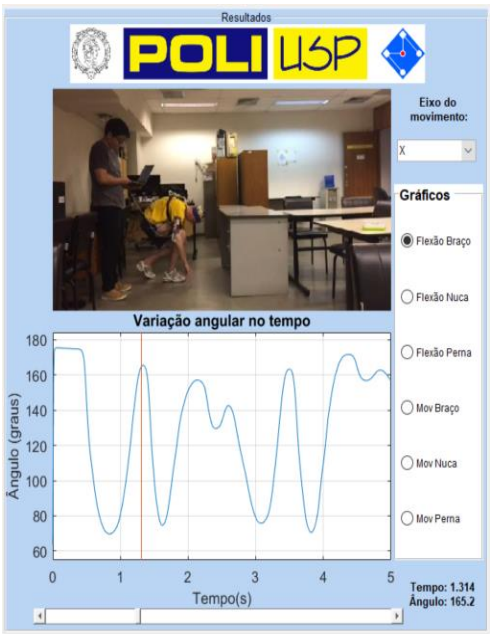
seguido de um máximo local com o corpo esticado e a cabeça virada para as mão (Figura 42b).

6.2.2 Sensor de Flexão do Braço

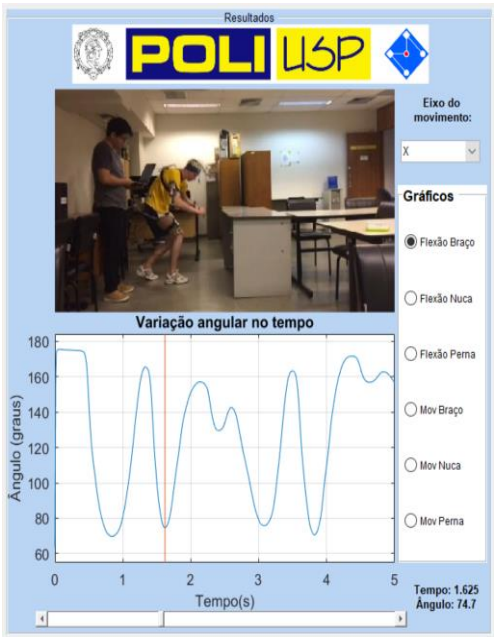
Os dados coletados pelo sensor de flexão localizado no cotovelo são mostrados na Figura 43. Os valores mostram o ângulo entre o bíceps e antebraço, diminuindo no sentido antebraço para o bíceps. Há novamente, dois ciclos no gráfico apresentado (Figura 43a). Um vale (Figura 43c) entre as posições de braço esticados (Figura 43) em cima e embaixo e um caminho semelhante, porém com uma leve movimentação do cotovelo ao abaixar-se (Figura 43).



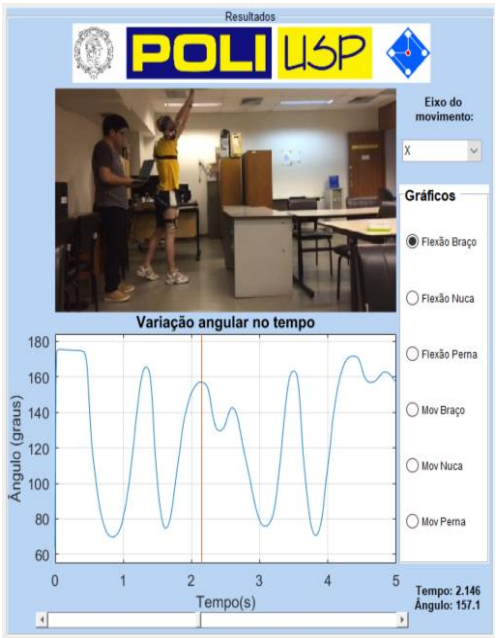
a. Dois ciclos realizados no teste



b. Dois ciclos realizados no teste



c. Dois ciclos realizados no teste

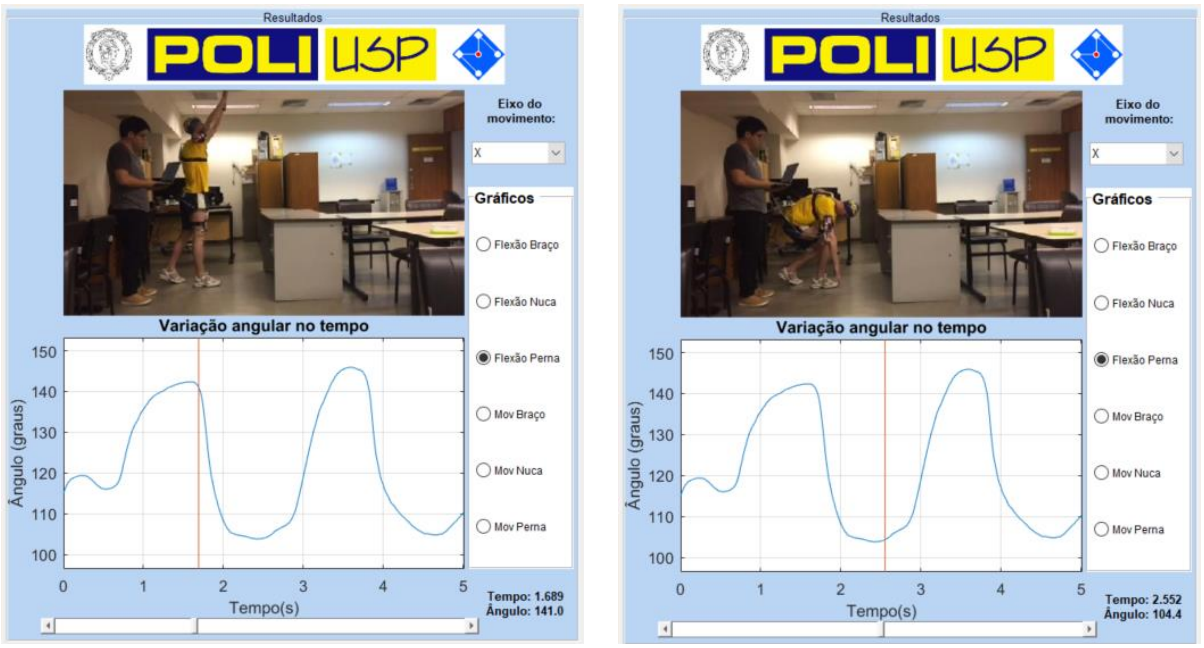


d. Dois ciclos realizados no teste

Figura 43: Variação do ângulo de flexão do braço

6.2.3 Sensor de Flexão da Perna

O sensor de flexão da perna é fixado no joelho e mede o ângulo entre coxa e canela, o ângulo diminui no sentido da canela para a coxa. Os dados registrados pelo sensor são mostrados na Figura 44.



a. Extensão do corpo

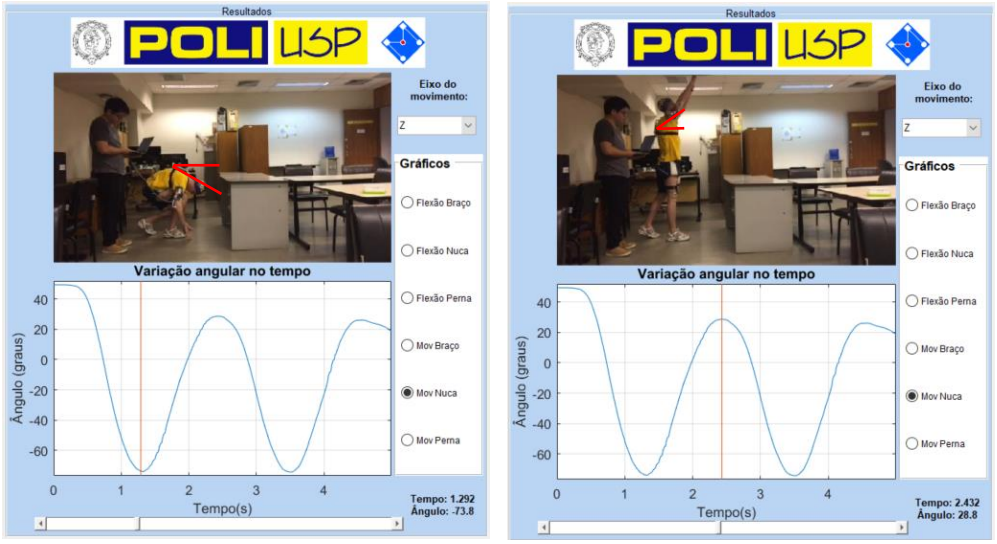
b. Posição de saída

Figura 44: Gráficos da medição do sensor de flexão fixado no joelho

Os dois ciclos são observados pelos dois picos seguidos de extensão (Figura 44a) e pelos vales que dobra do joelho ao realizar a posição de saída(Figura 44b).

6.2.4 IMU – Eixo Z das Costas

A Figura 45 mostra os dados coletados pela IMU localizada nas costas. Os ângulos coletados pela IMU, diferente dos sensores de flexão, não medem o ângulo relativo entre um membro e outro, apenas inclinações do membro em que ela está acoplada.



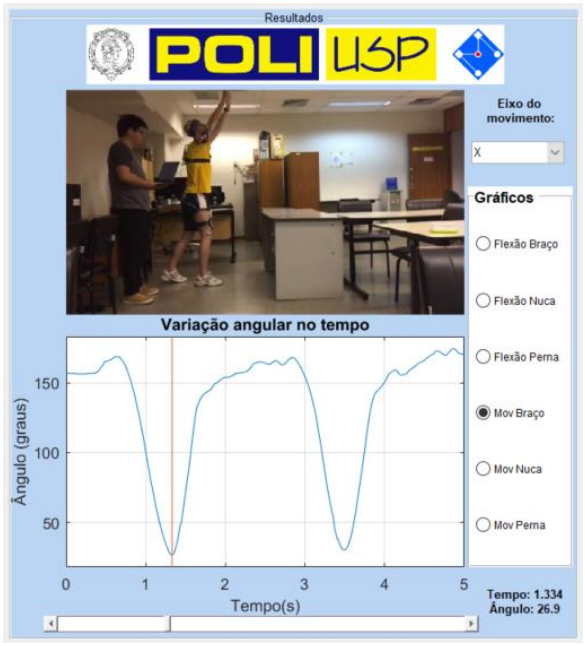
a. Posição de saída *b. Extensão do corpo*

Figura 45: Gráficos das medidas obtidas pelo eixo Z da IMU localizada nas costas

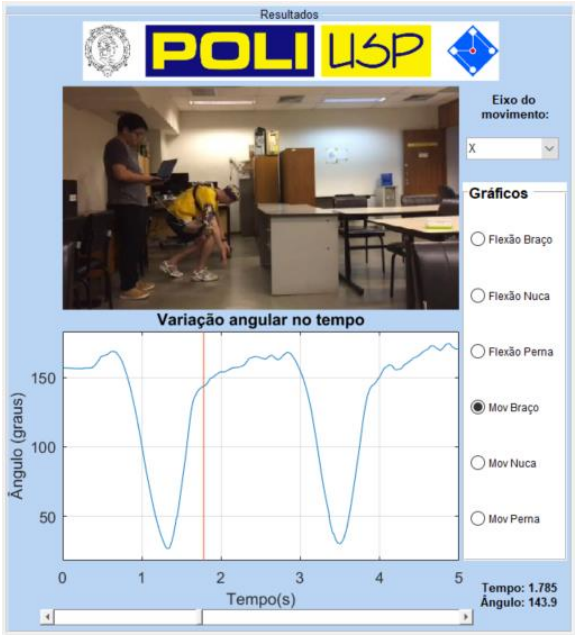
Na posição de saída (Figura 45a) o atleta apresenta uma inclinação negativa de -73,8º, enquanto que em (Figura 45b), apesar de esticado, há uma pequena inclinação das costas como pode ser notada de 20º.

6.2.5 IMU – Eixo X do Braço

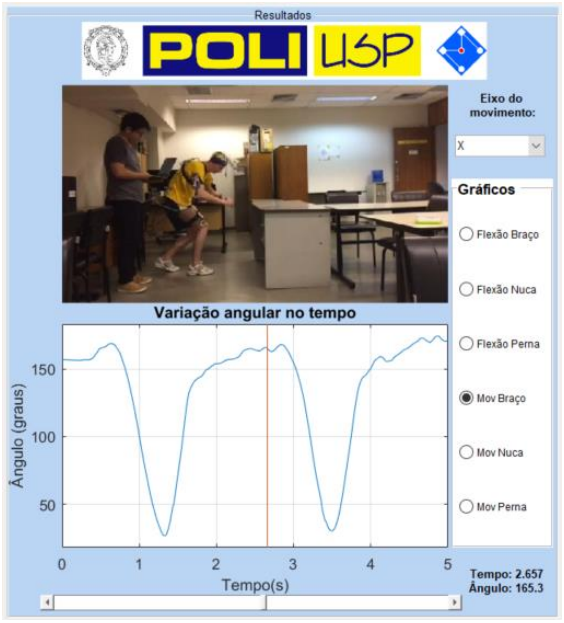
A Figura 46 mostra os dados obtidos pelo eixo X da IMU localiza na parte do tríceps.



a. Extensão



b. Posição de saída



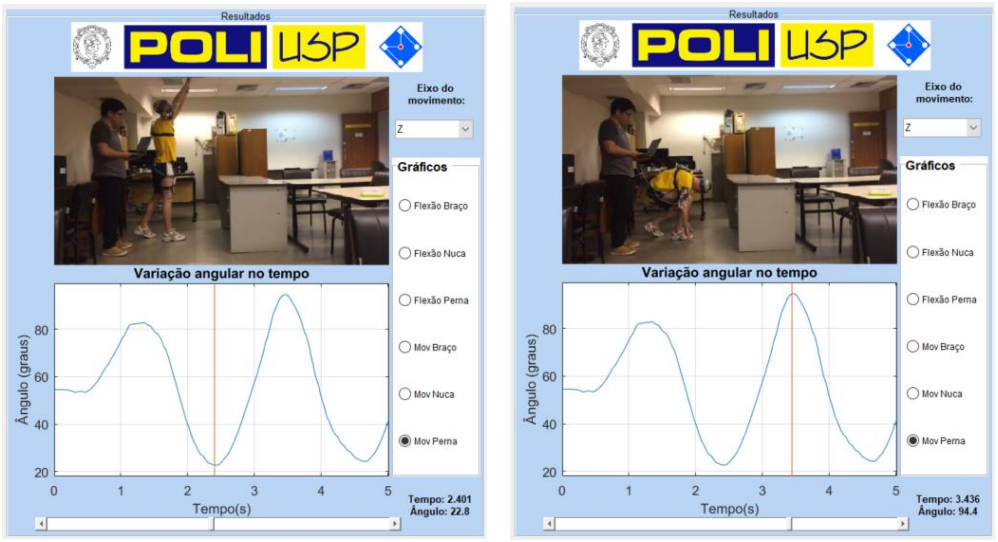
c. Transição entre a posição de saída e extensão

Figura 46: Gráficos das medidas obtidas pelo eixo X da IMU localizada no braço

Nota-se que no caso do braço o duplo ciclo apresenta uma diferença significativa no tempo em que o atleta fica esticado na posição de saída. Na primeira (Figura 46a) a variação de movimento é mais rápida que na segunda que visivelmente manteve o braço na mesma posição por mais tempo, ou seja aproximadamente 1 segundo (Figura 46b e Figura 46c)

6.2.6 IMU – Eixo Z da Perna

A Figura 47 mostra pontos de interesse do gráfico obtido pelas medições do eixo Z da IMU localizada na coxa.



a. Extensão

b. Posição de saída

Figura 47: Gráficos das medidas obtidas pelo eixo Z da IMU localizada na coxa

Seguindo o padrão esperado, a rotação da IMU da coxa também apresentou os dois ciclos, desta vez, bem definidos. No total, a extensão medida foi de 22,8º, enquanto que no encolhimento do corpo foi observado 94,4º.

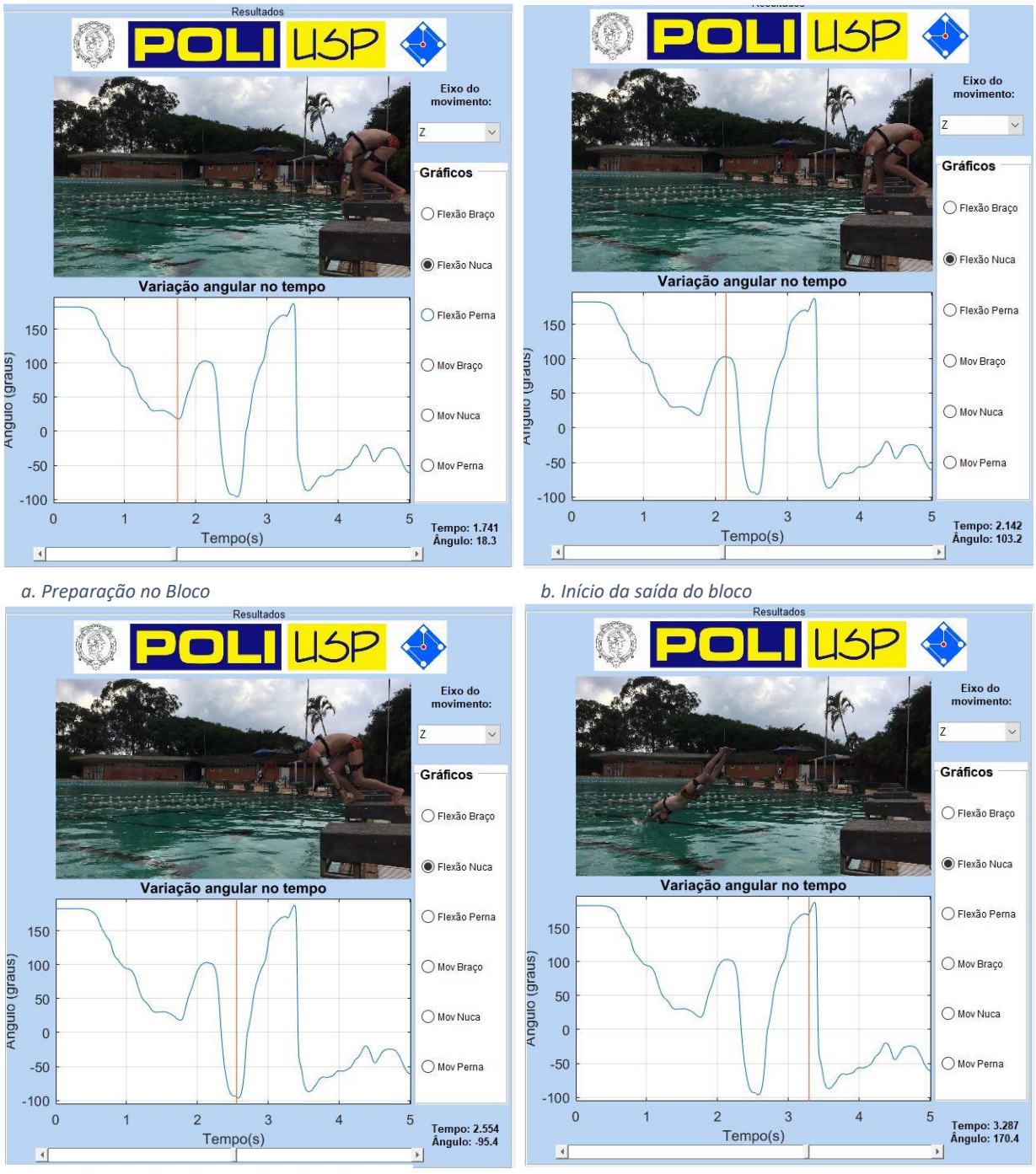
6.3 Teste na piscina

Os testes na piscina foram feitos com o isolamento contra água e encapsulamento mostrados no item 5.2.3.

A seguir estão os resultados obtidos dos sensores de flexão de todos os membros instrumentados e os resultados dos eixos que eram mais significantes para o movimento.

6.3.1 Sensor de Flexão da Nuca

A Figura 48 mostra os dados coletados peos sensor de flexão da nuca. Os mínimos locais acontecem quando a flexão é no sentido da cabeça para as costas e os máximos locais o contrário.



a. Preparação no Bloco

b. Início da saída do bloco

c. Flexão máxima da cabeça para cima

d. Alinhamento da cabeça entre os braços para entrada na água

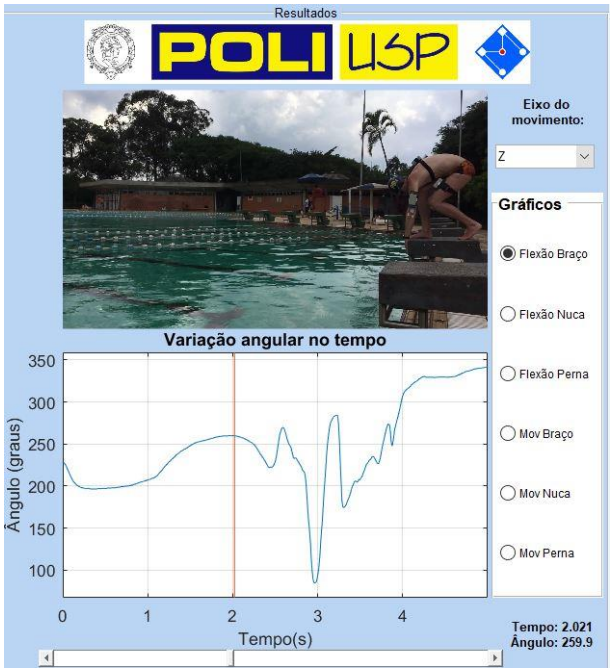
Figura 48: Gráficos da medição do sensor de flexão fixado na nuca

Pode-se ver que os dados acompanham o movimento do atleta, o primeiro mínimo local é atingido quando o atleta prepara a saída no bloco (Figura 48a),

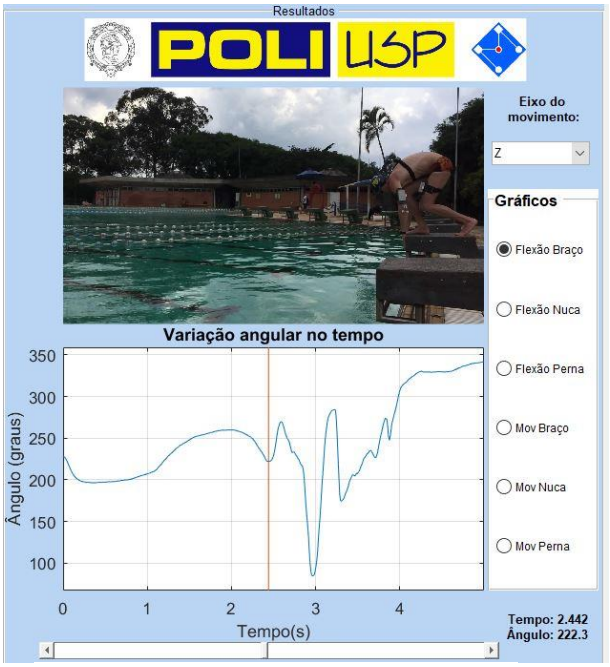
prosseguido de um máximo local no início do salto (Figura 48b). Na Figura 48c é atingido o mínimo absoluto, ou seja o menor ângulo entre a cabeça e as costas.

6.3.2 Sensor de Flexão do Braço

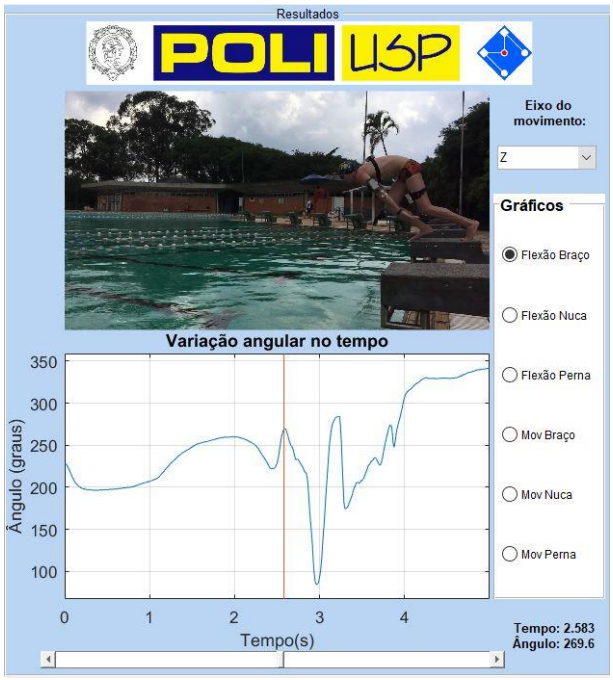
Os dados coletados pelo sensor de flexão localizado no cotovelo são mostrados na Figura 49. Os valores mostram o ângulo entre o bíceps e antebraço, diminuindo no sentido antebraço para o bíceps.



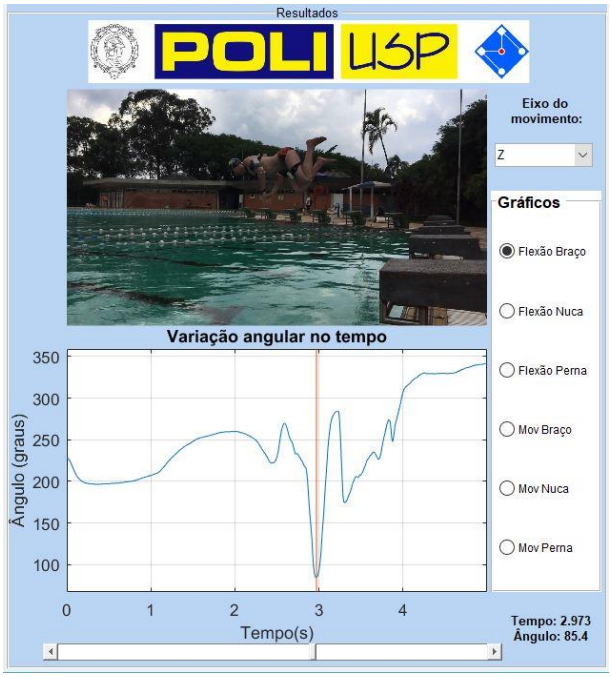
a. Preparação do salto



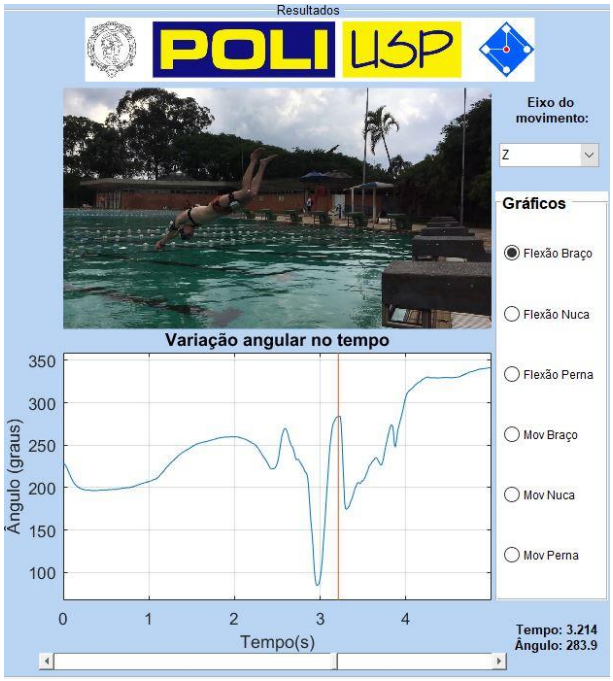
b. Pequena flexão para impulso



c. Extensão do braço empurrando o bloco



d. Flexão do braço ao projetá-los para frente



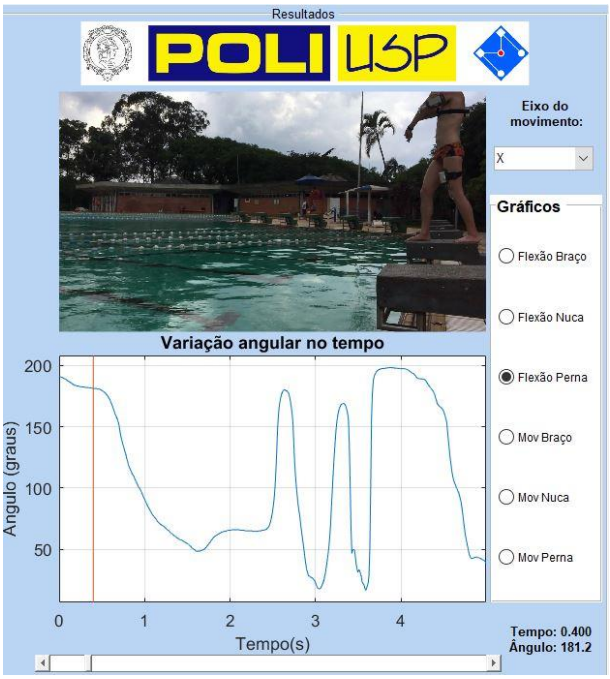
e. Extensão do braço para entrada na água

Figura 49: Gráficos da medição do sensor de flexão fixado no cotovelo

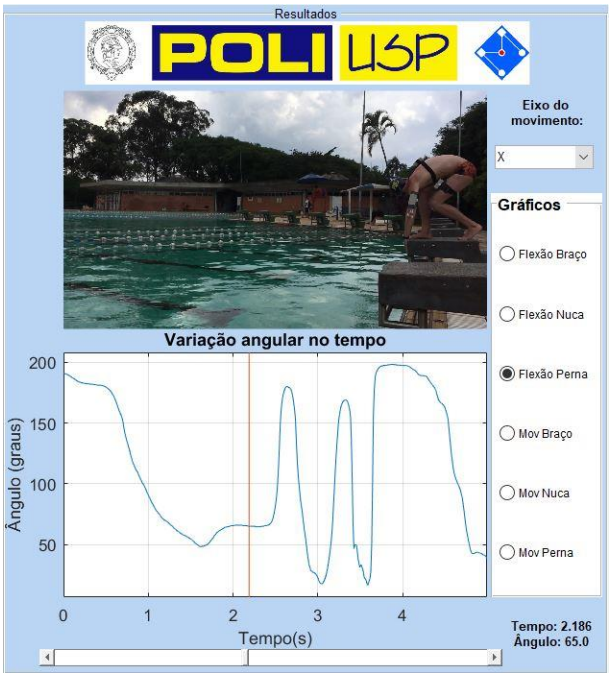
No preparação do salto (Figura 49a) o atleta está com os braços estendidos, ao iniciar o salto o atleta faz uma pequena flexão dos braços, evidenciado na Figura 49b para empurrar o bloco durante a saída, estendendo o braços novamente Figura 49c. Na Figura 49d percebe-se que ao projetar os braços para frente o atleta dobra (ângulo de 85.4º) os braços e depois os estende novamente para entrar na água (Figura 49e).

6.3.3 Sensor de Flexão da Perna

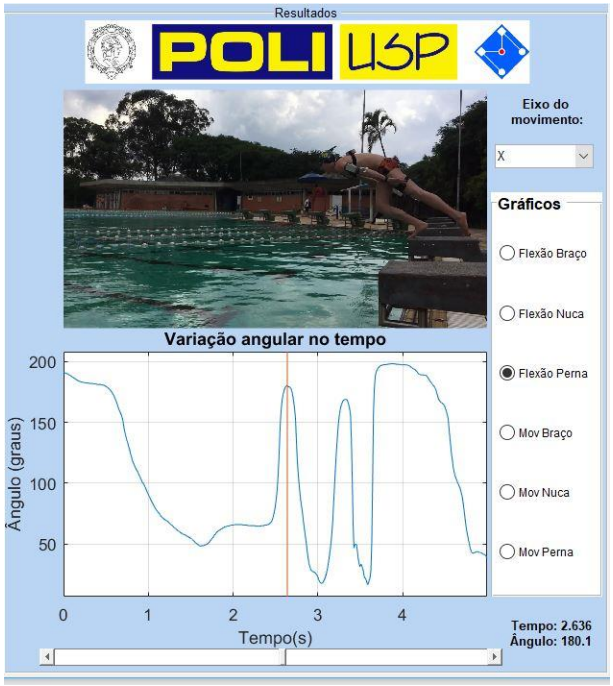
O sensor de flexão da perna é fixado no joelho e mede o ângulo entre coxa e canela, o ângulo diminui no sentido da canela para a coxa. Os dados registrados pelo sensor são mostrados na Figura 50.



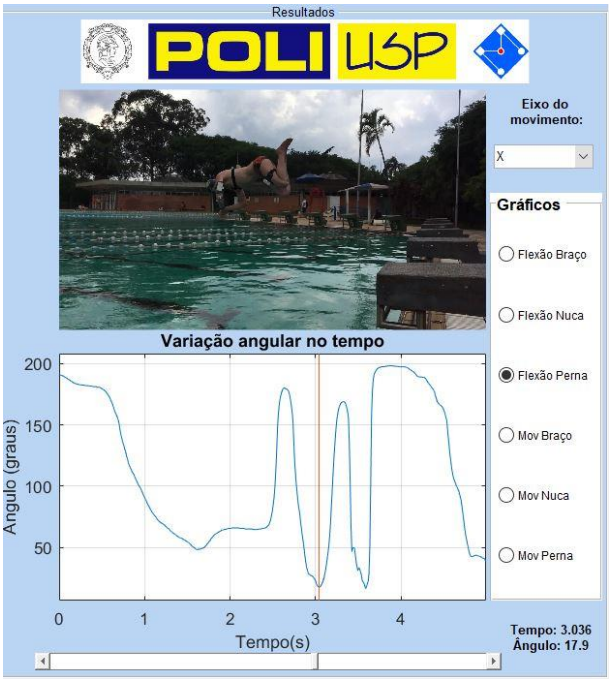
a. Atleta de pé no bloco com a perna estendida



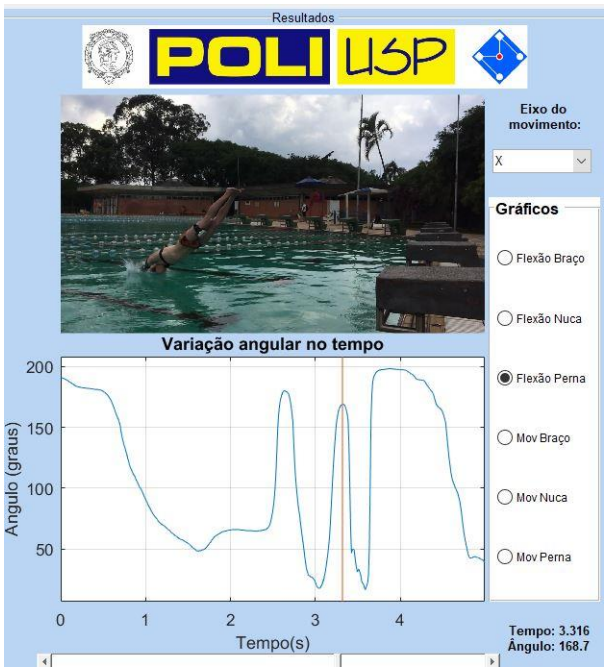
b. Preparação no bloco



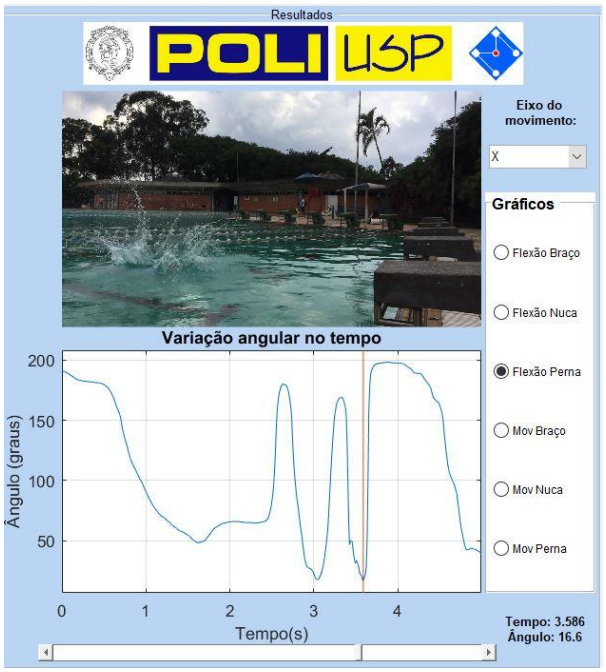
c. Saída do bloco, extensão máxima da perna



d. Flexão da perna na fase de voo



e. Extensão da perna para entrada da água



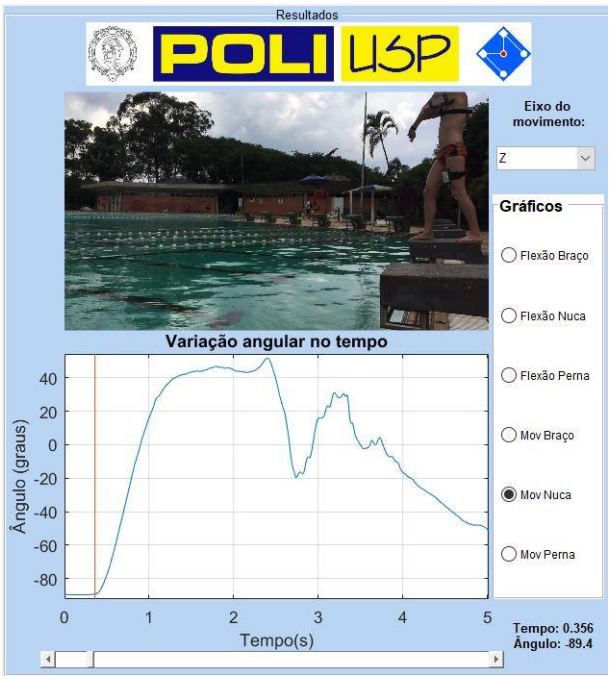
f. Nova flexão ao entrar na água para remada com as pernas

Figura 50: Gráficos da medição do sensor de flexão fixado no joelho

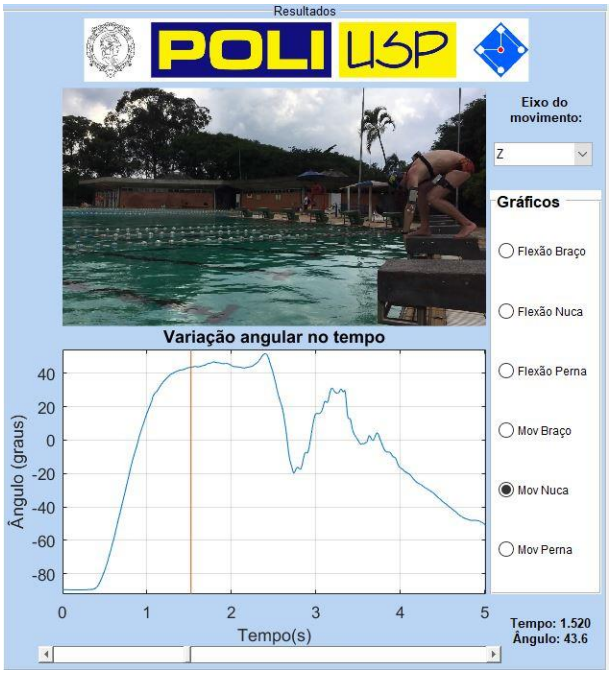
A Figura 50a mostra o atleta de pé no bloco com o registro de ângulo de 181.2º, na preparação para o salto ocorre a flexão da pena levando o ângulo registrado para 65,0º (Figura 50b). Na saída do bloco, o atleta estende totalmente sua perna, registrando o ângulo de 180.1º (Figura 50c). Durante a fase de voo, o atleta faz uma flexão das pernas, com ângulo de 17,9º (Figura 50d) e posterior extensão para entrada na água (Figura 50e). A Figura 50e mostra um registro dentro da água da remada realizada pelas pernas, após isso os sensores se desacoplam do atleta.

6.3.4 IMU – Eixo Z das Costas

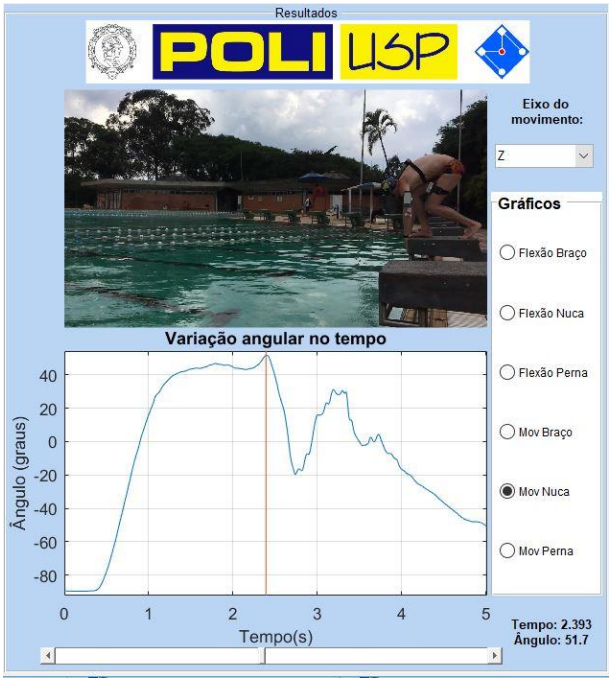
A Figura 51 mostra os dados coletados pela IMU localizada nas costas. Os ângulos coletados pela IMU, diferente dos sensores de flexão, não medem o ângulo relativo entre um membro e outro, apenas inclinações do membro em que ela está acoplada.



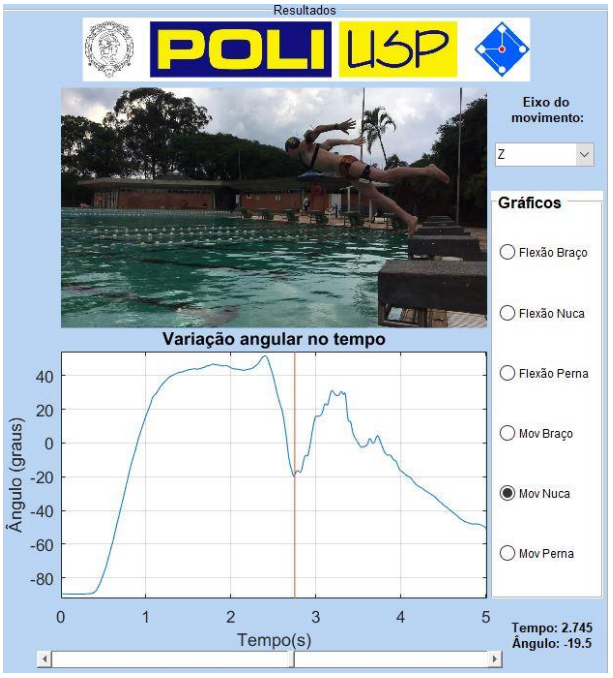
a. Atleta de pé no bloco



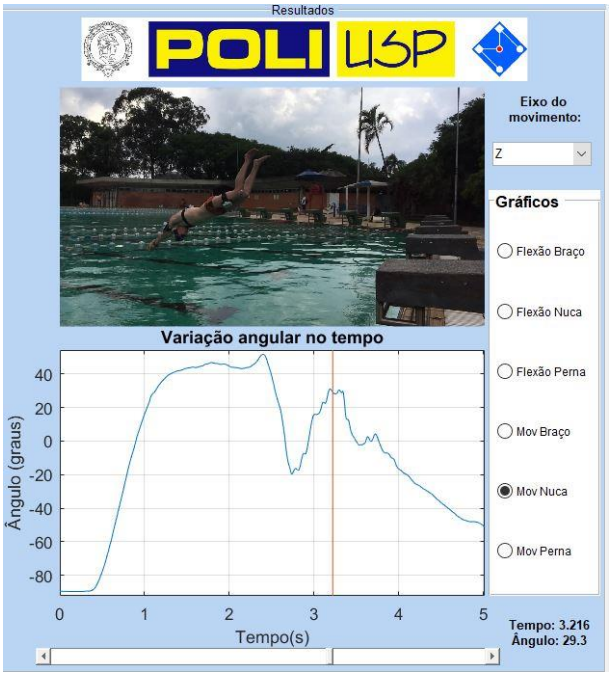
b. Preparação da saída



c. Início da saída



d. Fase de vôo



e. Entrada na água

Figura 51: Gráficos das medidas obtidas pelo eixo Z da IMU localizada nas costas

No início, o atleta passa da configuração de pé (Figura 51a) no bloco para a preparação para a saída (Figura 51b), esta transição gera a uma diferença entra os ângulos de aproximadamente 130º. A (Figura 52) mostra esta diferença analisando a imagem obtida pela câmera.

Na (Figura 51c) nota-se um aumento na inclinação devido à flexão dos braços e posterior redução quando o atleta entra na fase de voo (Figura 51d). Por fim, ao entrar na água ocorre um novo aumento na inclinação (Figura 51e).

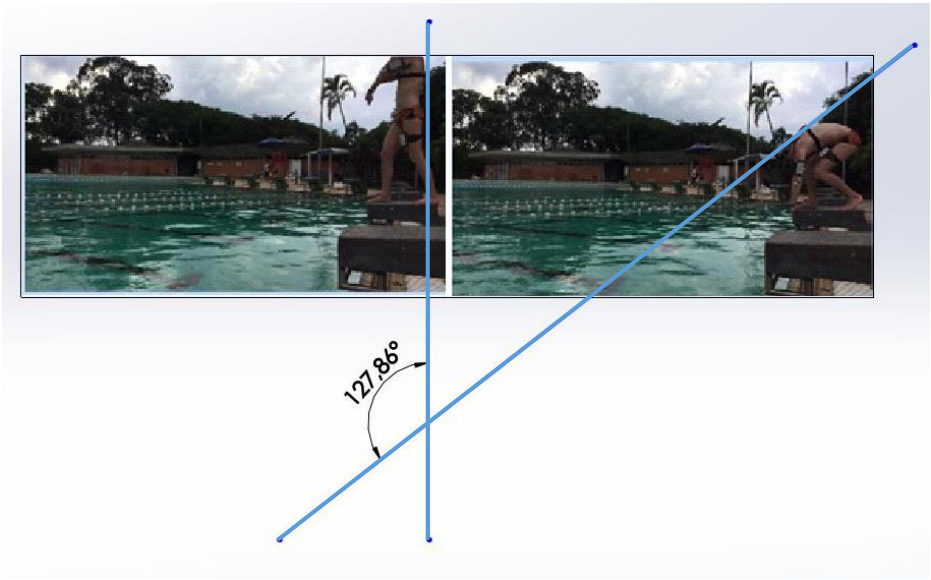
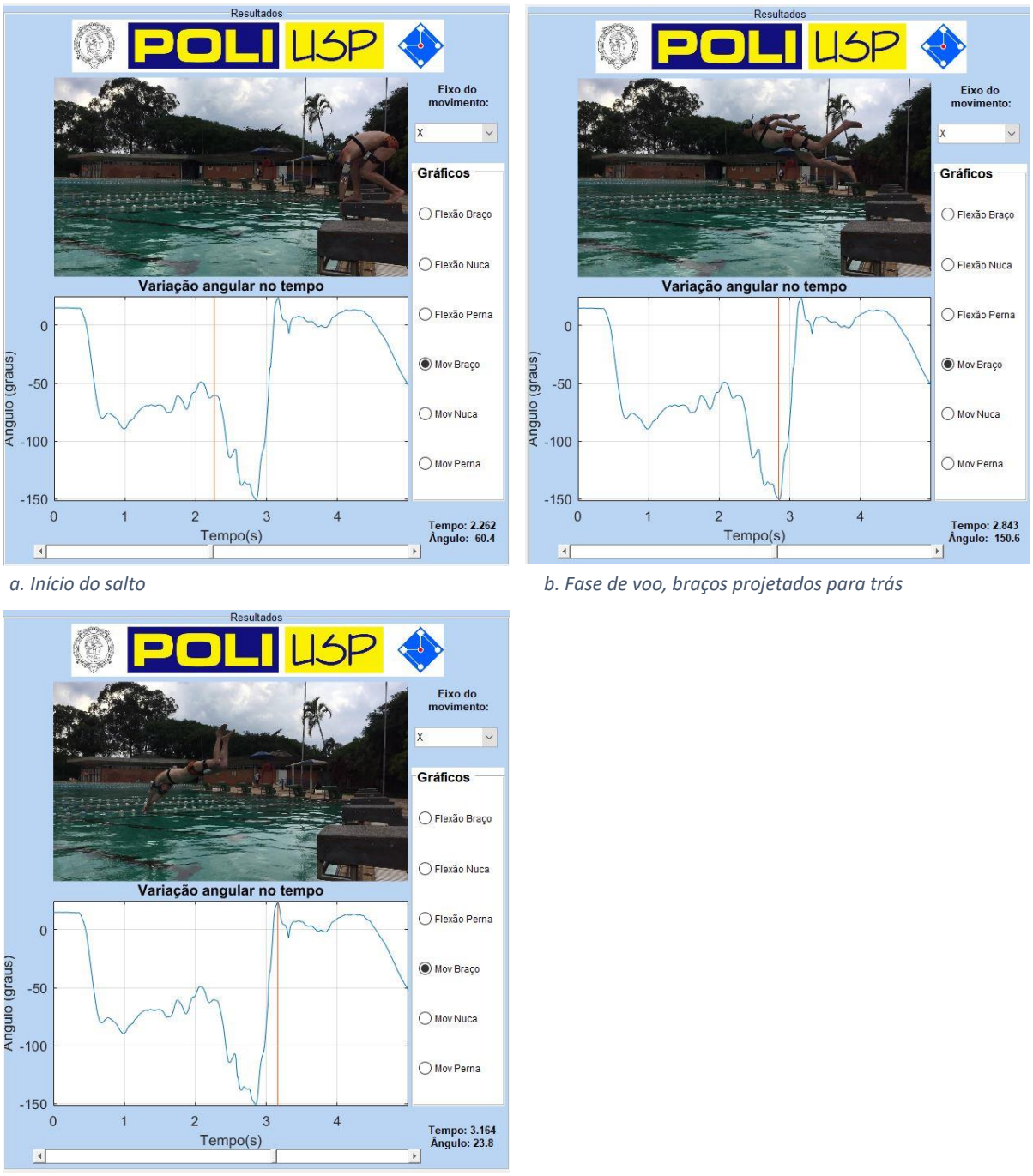


Figura 52: Diferença entre as inclinações das partes a e b da Figura 51

6.3.5 IMU – Eixo X do Braço

A Figura 53 mostra os dados obtidos pelo eixo X da IMU localiza na parte do tríceps.



a. Início do salto

b. Fase de voo, braços projetados para trás

c. Braços projetados para frente, para entrada na água

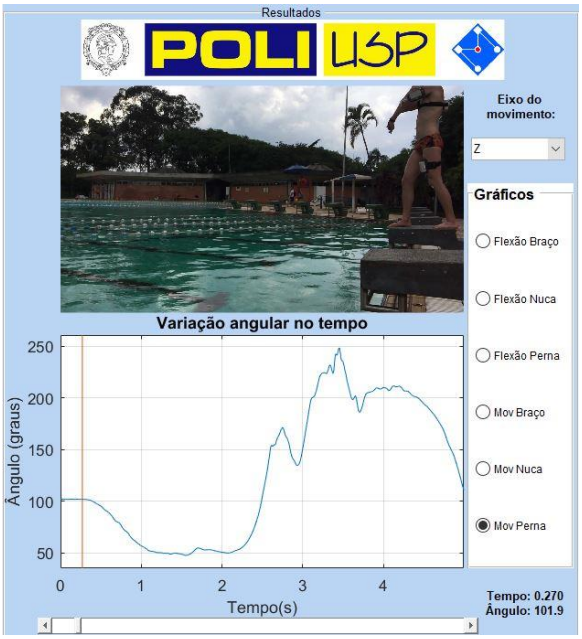
Figura 53: Gráficos das medidas obtidas pelo eixo X da IMU localizada no braço

No início do salto a inclinação marcada é de -60.4º (Figura 53a) passando para -150.6º quando o atleta projeta seus braços para trás (Figura 53b), há uma rotação de aproximadamente 90º em relação ao eixo X. Ao projetar os braços para frente, a

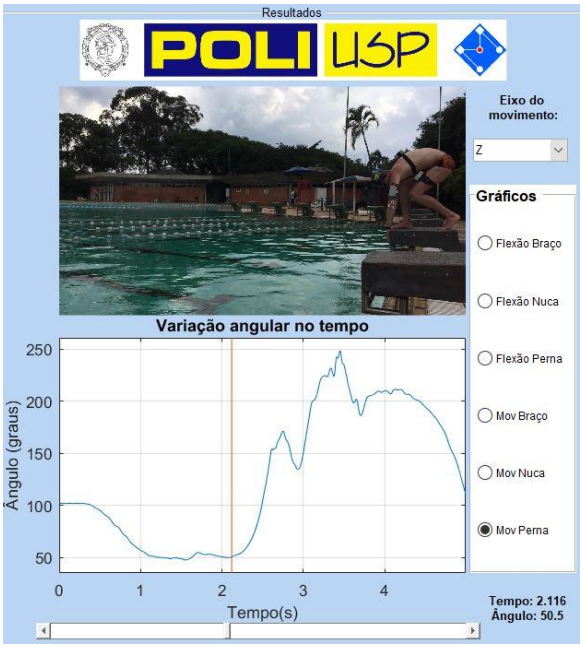
inclinação marcada é de 23,8º (Figura 53c), o que signica que o membro foi rotacionado 174,4º.

6.3.6 IMU – Eixo Z da Perna

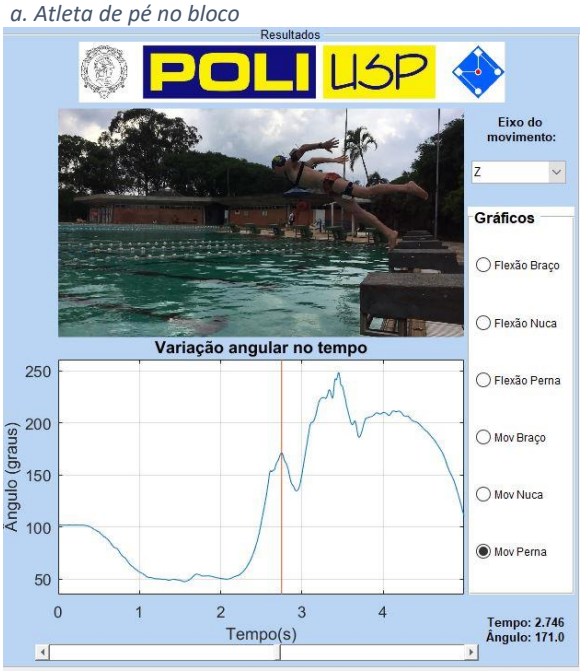
A Figura 54 mostra pontos de interesse do gráfico obtido pelas medições do eixo Z da IMU localizada na coxa.



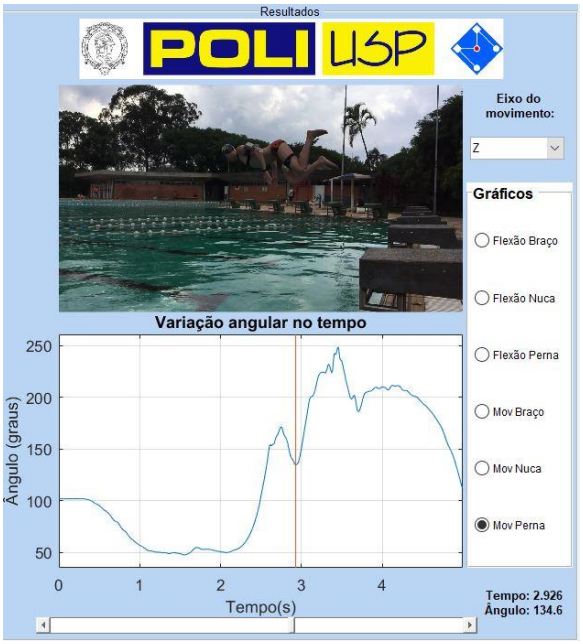
a. Atleta de pé no bloco



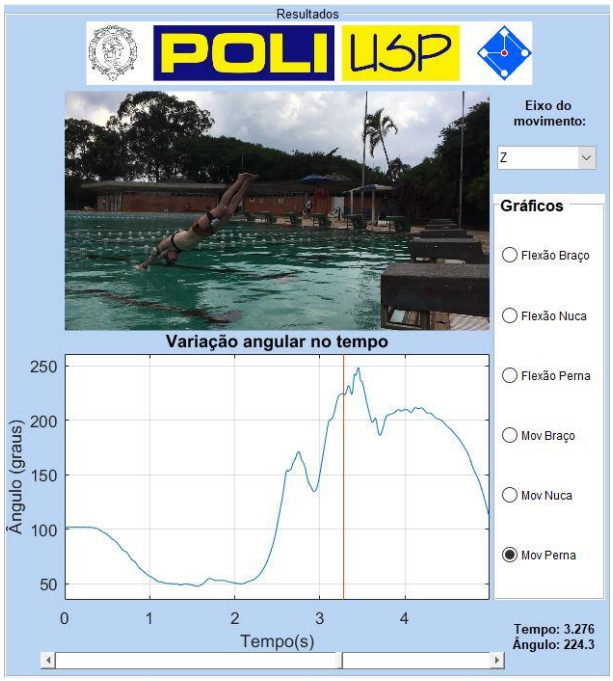
b. Início do salto



c. Início da fase de voo



d. Recolhimento da perna na fase de voo



e. Entrada na água

Figura 54: Gráficos das medidas obtidas pelo eixo Z da IMU localizada na coxa

Com o atleta de pé no bloco há uma inclinação de 101,9º passando para 50,5º no início do salto. No início da fase de voo a IMU marca 171,0º indicando uma rotação de 120º. Durante a fase de voo, o atleta recolhe a perna, indicado pelo pequeno vale da Figura 54d. A Figura 54e mostra a inclinação quando o atleta entra na água.

7 DISCUSSÃO

7.1 Medição com os sensores de flexão

Através dos resultados obtidos vê-se que é possível realizar a medição dos movimentos do atleta durante sua saída.

Os comportamentos das curvas obtidas refletem o movimento filmado. No entanto, nos sensores localizados no braço (6.3.2) e na nuca (6.3.1), em alguns momentos, os valores lidos não representavam o que realmente estava acontecendo. No caso do sensor localizado na perna, tanto o comportamento da curva, quanto os valores reproduziam os movimentos do atleta.

Comparando a análise de vídeo do ângulo de flexão com os dados obtidos pelo sensor de flexão mostrados pelo software criado, obtêm-se o resultado na Figura 55

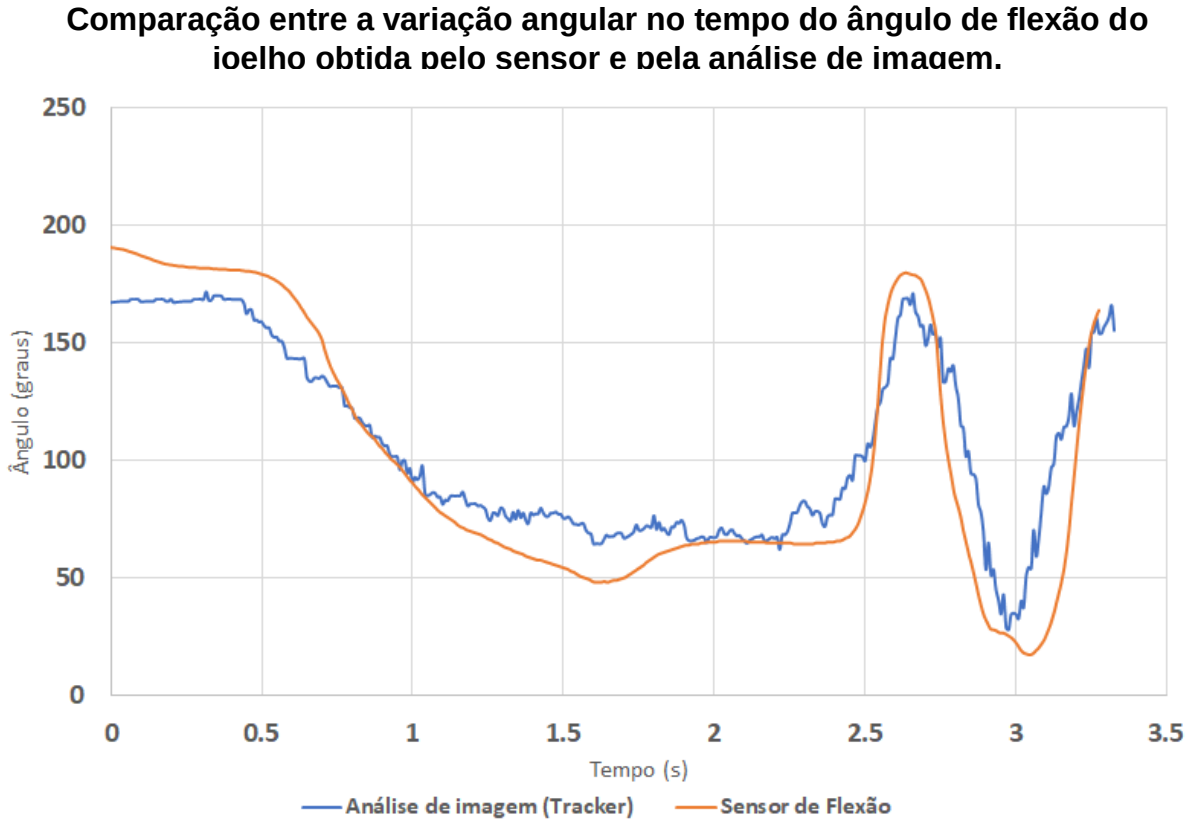


Figura 55: Variação angular no tempo do ângulo de flexão do joelho

Na análise de imagem pelo Tracker, não existem dados a partir do tempo 3,5s pois o atleta está dentro da água.

Pode-se ver semelhança entre as curvas, no ponto de flexão máxima que ocorre na fase de voo, mostrada na Figura 50d, o ângulo medido pelo sensor é de 17,9º, o mesmo momento medido por imagem mostra um ângulo de 28,8º.

O desvio relativo médio entre o valor obtido pelos sensores e os valores medidos por análise de imagem é de 17%. Entende-se que pode ter sido causado pelo modo de fixação utilizado e pela calibração dos sensores.

7.2 Medição com as IMUs

As análises com flexores medem ângulos de junções, ou seja, entre partes, segmentos do corpo. De forma análoga, as IMUs têm seus dados melhor compreendidos quando são utilizadas comparações. No caso dos resultados, foi possível tirar conclusões não só com as imagens, mas também devido a evolução dos ângulos e o referencial de cada IMU.

O que se nota é que ângulos sozinhos podem não trazer informações de forma instantânea e simples. É necessário relacionar suas variações, bem como suas referências, visto que nem todos os eixos são de extrema importância para o movimento.

Além disso, é possível que haja reconstrução do corpo por meio de segmentos relacionando as inclinações das IMUs. Dessa forma, tendo a premissa de que os braços se movimentam de forma espelhada e as pernas também, seria possível observar os ângulos relativo entre tronco e membros de cada ponto no tempo uma vez que os gráficos estão todos sincronizados. Essa informação seria de maior valia por apresentar compreensão mais simples e direta que a anterior.

7.3 Comparação com os requisitos

A comparação com os requisitos é mostrada na Tabela 8.

Em relação aos requisitos mecânicos o requisito de massa foi satisfeito, no entanto a espessura do conjunto ficou aquém do desejado. No entanto, isto pode ser contornado em um reprojeto da placa, substituindo placas de prototipagem por circuitos dedicados à aplicação. Além disso é possível utilizar outro tipo de bateria, que ocupe menos espaço, mas mantenha a carga.

O requisito elétrico de frequência de aquisição foi satisfeito, os sinais são amostrados a uma taxa de 100Hz.

Em relação aos requisitos funcionais o projeto atende a maioria deles. A medição dos ângulos foi satisfeita em partes, pois no caso dos sensores de flexão, os resultados estão sujeitos ao tipo de fixação utilizado e à calibração, mas caso estes sejam melhorados, consegue-se resultados satisfatórios como mostrado na Figura 55.A sincronização do vídeo também foi satisfeita em partes, conseguiu-se acionar a câmera no mesmo momento que os sensores, no entanto, existe um tempo de cerca de 0,2s entre o acionamento da câmera e o início da filmagem.

O software cumpriu seus requisitos ao fornecer uma interface simples e de fácil uso.

Tabela 8: Comparação dos resultados com os requisitos

Descrição do requisito	Especificação	Valor obtido
Requisitos mecânicos		
Massa	Menor que 200g	180g
Espessura	Menor que 30mm	40mm
Requisitos elétricos		
Frequência de Aquisição	Maior que 80Hz	100Hz

Descrição do requisito	Atingido?
Requisitos Funcionais	
Medir a angulação da cabeça, tronco, pernas e braços	Em parte
Sincronizar o início da coleta de dados e filmagem	Em parte
Apresentação de interface de captura de dados	Sim
Requisitos não Funcionais	
O programa deve ser amigável para os técnico e atletas	Sim
O programa deve entregar o relatório pouco tempo depois do atleta completar seu percurso	Sim

8 CONCLUSÃO

Depreende-se, portanto, que os objetivos de mapear a saída de um atleta de natação e transmitir os dados aos interessados de forma simples e amigável é possível. Com isso, os sensores não só foram importantes para as análises, como, no caso das IMUs, podem ser ainda mais benéficos caso o programa seja aprimorado e forneça diretamente ângulos relativos entre os membros e o tronco. Com esse projeto, há uma base significativa de possíveis análises e formas de aquisição de dados que podem ser usadas para esses fins.

Nota-se que há melhorias a serem feitas nos módulos atuais como:

- Redução dos desvios na sincronização entre vídeo e sensores;
- Fixação dos módulos em especial do flexor dos braços e da nuca;
- Garantia da calibração para obtenção de ângulos mais fidedignos com as imagens.

8.1 Próximos passos e pontos a melhorar

Com o objetivo de deixar a análise do atleta mais completa e tornar o projeto mais robusto, deve-se validar as medidas obsvervadas pelo padão ouro.

Além disso, os pontos a seguir sugerem melhorias a serem implementadas no projeto.

8.1.1 Sensores de flexão.

A fixação do sensor de flexão pode ser melhorada, como visto no item 6.3.1, os sensores de flexão podem se soltar do corpo do atleta no momento em que ele entra na água. Além disso, fatores como a transpiração do atleta tornam esta fixação precária. Neste caso a fixação por fitas adesivas pode ser trocada por uma cotoveleira[10]/joelheira elástica[11]. Também pode ser cogitado substituir os sensores de flexão por outras IMUs.

Outra opção ainda melhor seria substituí-los por outras IMUs e trabalhar com o ângulo relativo entre elas.

[10] https://www.netshoes.com.br/cotoveleira-elastica-branca-preto-J96-0251-006?campaign=gglepqpla&gclid=CjwKCAiAz7TfBRAKEiwAz8fKOLnS5ixQqQkYXHa21LqbUslWNzXmTUTgjeOlfpca-54plCGx0tYfGhoCmd8QAvD_BwE

[11] https://www.centauro.com.br/joelheira-elastica-speedo-2-unidades-adulto-894935.html?cor=02&origem=google_kenshoo&utm_source=google_gs&utm_medium=SCH_NOB_PLA_Acess%C3%B3rios&utm_campaign=all\acess%C3%B3rios\academia%20/%20fitness\other\89493502&gclid=CjwKCAiAz7TfBRAKEiwAz8fKOC7GvyKKFtv5hsdXb7AY2aiTcUzyP4i7memeAlvpdIqOjVf7zVryphoCo1gQAvD_BwE

8.1.2 Inclusão de outros módulos de medição

Outros módulos de medição a serem inclusos no projeto são: medição do tempo em que o atleta chega aos 15 metros e medição da força aplicada pelos braços do atleta no bloco.

O diagrama da Figura 56 dá uma sugestão do que haveria em cada módulo e quais parâmetros são recebidos e passados em cada componente.

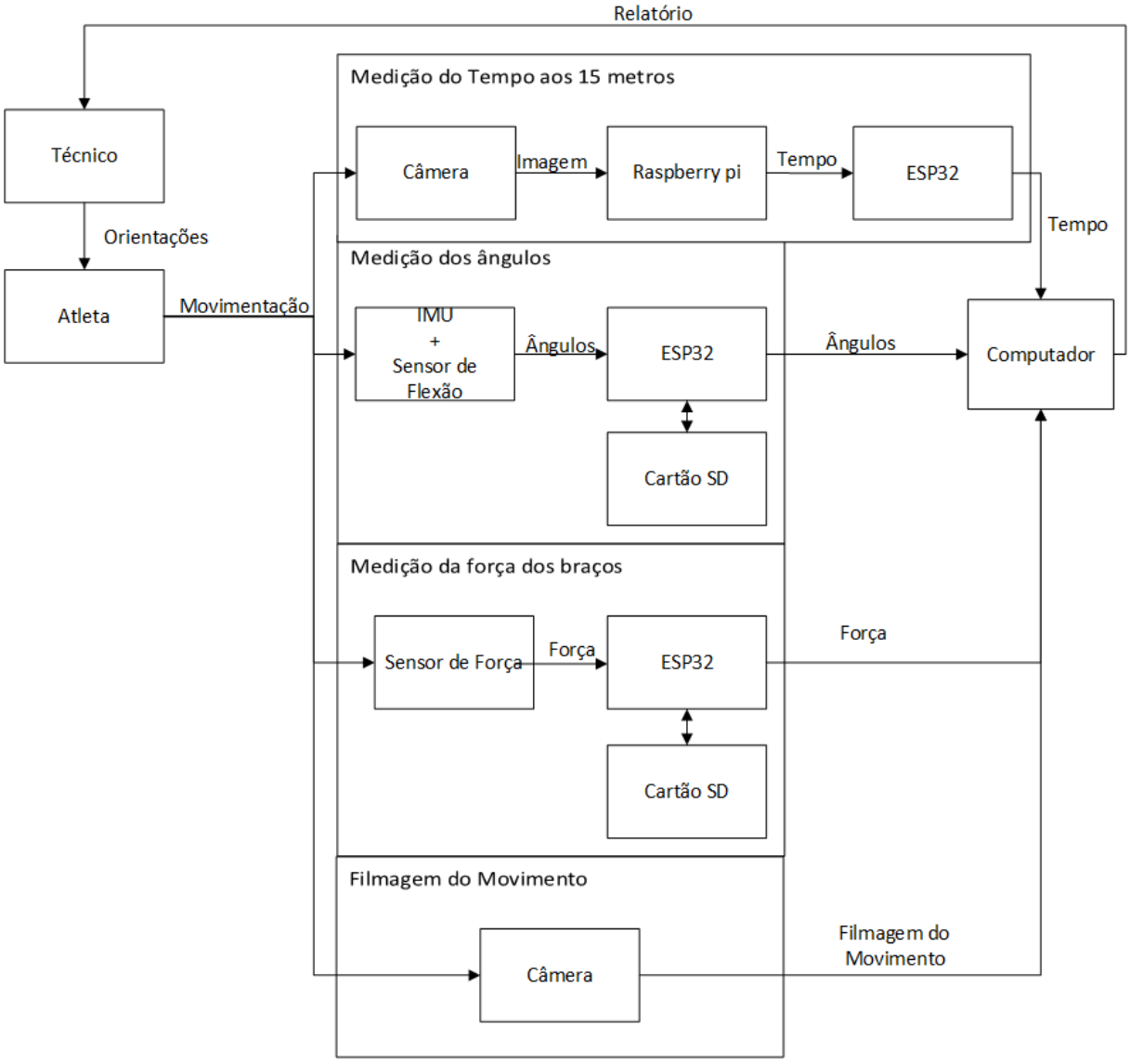


Figura 56: Diagrama com análise mais robusta do movimento

A seguir são sugeridas algumas soluções para a implementação de outros modos.

a) Medição do tempo aos 15 metros

Para a medição do tempo do atleta aos 15 metros, algumas propostas de solução podem ser exploradas.

A primeira delas, é um dispositivo semelhante aos utilizados em competições, que consiste em um *touchpad*[12] em que o atleta bate com as mãos. No entanto, esta solução, ainda que meça o tempo, traz uma dificuldade em sua fixação, pois estes produtos, em seu uso regular, são fixados nas paredes da piscina.

A segunda opção consiste em um cordão de 15m em que uma de suas pontas é fixada no atleta e outra é conectada em um circuito, deste modo, no momento em que o atleta atingisse os 15m o circuito ficaria aberto, sinalizando o sistema. Entretanto, esta solução apresenta outro aparato a ser acoplado no corpo do atleta, o que pode tornar seu uso desconfortável.

A terceira solução pensada foi utilizar-se de câmera, a partir de boias com cores chamativas e específicas, com a análise da imagem é possível traçar uma linha e comparar com outros objetos, neste caso a touca do atleta, tornando possível detectar a passagem do mesmo pelos 15m.

8.1.3 Medição da força aplicada pelos braços no bloco

Para a medição da força aplicada pelos braços no bloco pode ser desenvolvido um sensor simples, utilizando-se de potenciômetros lineares e uma viga de metal. Assim, a força é calculada a partir do deslocamento da viga de metal.

Existem opções no mercado como os sensores de força da Kistler[13], no entanto, nesta aplicação, não se deseja mensurar com grande acurácia e precisão a força, apenas a sua magnitude é suficiente. Portanto, o projeto proposto é suficiente e se torna uma solução de menor custo.

8.2 Reflexão

Com o fim do projeto, melhorias e inclusões de novos módulos como de sensor de força aplicada em um bloco de natação e cronometragem do tempo da travessia de 15 metros em uma piscina, percebeu-se que seriam restritos à requisitos da natação. Portanto, nota-se que as formas como os módulos desse projeto foram desenvolvidos e como o programa relaciona dados captados e movimento podem ser aplicados à movimentos quaisquer. Esse caráter geral de aplicabilidade do projeto sugere que novos estudos podem usá-lo como base para análise de movimentos diferentes da saída de um atleta de natação. Por isso, os mais variados esportes e até mesmo acompanhamentos da área médica como fisioterapias e controles motores de Pessoa com Deficiência (Pcd) podem utilizar-se deste trabalho.

[12] https://www.sportstiming.com.au/shop/Aquatic+Sports/Swimming/OMEGA+OCP5+TouchPad%3Fsku=OCP5.html

[13] https://www.kistler.com/en/product/type-9136b/

9 REFERÊNCIAS BIBLIOGRÁFICAS

ALKATAN, M. et al. Effects of Swimming and Cycling Exercise Intervention on Vascular Function in Patients with Osteoarthritis. **American Journal of Cardiology**, v. 117, n. 1, p. 141–145, 2016.

ARELLANO, R. **Evaluación de la fuerza propulsiva en natación y su relacion con el entrenamiento y la técnica**. [s.l.] Universidad de Granada, 1992.

BARLOW, H. et al. The effect of different kick start positions on OMEGA OSB11 blocks on free swimming time to 15m in developmental level swimmers. **Human Movement Science**, v. 34, n. 1, p. 178–186, 2014.

CAMPIGLIO, G.; MAZZEO, J.; RODRIGUEZ, S. A wireless goniometry system. **Journal of Physics: Conference Series**, v. 477, n. 1, 2013.

CBDA - Natação, Pólo Aquático, Maratonas Aquáticas, Nado Sincronizado, Saltos Ornamentais. Disponível em: <http://www.cbda.org.br/>. Acesso em: 27 maio. 2018.

COMMISSION, T. S.; TECHNOLOGY, C. PR16 - Meeting between FINA and swimwear manufacturers. 2009.

DADASHI, F. et al. **Towards estimation of front-crawl energy expenditure using the wearable aquatic movement analysis system (WAMAS)**. 2013 IEEE International Conference on Body Sensor Networks, BSN 2013. **Anais**...2013

DADASHI, F.; MILLET, G. P.; AMINIAN, K. Inertial measurement unit and biomechanical analysis of swimming: An update. **Schweizerische Zeitschrift fur Sportmedizin und Sporttraumatologie**, v. 61, n. 3, p. 28–33, 2013.

DADASHI, F.; MILLET, G. P.; AMINIAN, K. Estimation of front-crawl energy expenditure using wearable inertial measurement units. **IEEE Sensors Journal**, v. 14, n. 4, p. 1020–1027, 2014.

DONNO, M. et al. A new flexible optical fiber goniometer for dynamic angular, measurements: Application to human joint movement monitoring. **IEEE Transactions on Instrumentation and Measurement**, v. 57, n. 8, p. 1614–1620, 2008.

DRAGUNAS, A. Factors Affecting Block Performance From The Omega OSB11 Starting Platform. n. September, 2015.

FINA | fina.org - Official FINA website. Disponível em: <http://www.fina.org/content/fina>. Acesso em: 27 maio. 2018.

FUENTE, B. D. E. L. A.; GARCIA, F.; ARELLANO, R. Are the Forces Applied in the Vertical Countermovement Jump Related to the Forces Applied During the Swimming Start ? p. 207–212, 2000.

GARCÍA-RAMOS, A. et al. The relationship between the lower-body muscular profile

and swimming start performance. **Journal of Human Kinetics**, v. 50, n. 1, p. 157–165, 2016.

GUIGNARD, B. et al. **Behavioral dynamics in swimming: The appropriate use of inertial measurement unitsFrontiers in Psychology**, 2017.

HAGEM, R. M. et al. Real-time swimmers' feedback based on smart infrared (SSIR) optical wireless sensor. **Electronics Letters**, v. 49, n. 5, p. 340–341, 2013.

HONDA, K. E. et al. **THE EFFECT OF STARTING POSITION ON ELITE SWIM START PERFORMANCE USING AN ANGLED KICK PLATE**. International Society of Biomechanics in Sports. **Anais**...2012Disponível em: <https://www.researchgate.net/publication/235891648_THE_EFFECT_OF_STARTING_POSITION_ON_ELITE_SWIM_START_PERFORMANCE_USING_AN_ANGLED_KICK_PLATE>

IIZUKA, S. Original Research Immediate Effects of Deep Trunk Muscle. v. 11, n. 7, p. 1048–1053, 2016.

LAZAR, J. M. et al. Swimming and the heart. **International Journal of Cardiology**, v. 168, n. 1, p. 19–26, 2013.

LEWILLIE, L. Research in Swimming: Historical and Scientific Aspects. **Biomechanics and Medicine in Swimming (International Series on Sport Sciences - Band 14, Champaign, Illinois: Human Kinetics Publishers, 1983. - S. 7 - 16: Abb., 46 Lit., S. 7-16). Champaign: Human Kinetics.**, 1983.

MAGALHAES, F. A. DE et al. **Wearable inertial sensors in swimming motion analysis: a systematic reviewJournal of Sports Sciences**, 2015.

MAGLISCHO, E. W.; MAGLISCHO, E. W. **Swimming fastest**. [s.l.] Human Kinetics, 2003.

MOONEY, R. et al. Inertial Sensor Technology for Elite Swimming Performance Analysis: A Systematic Review. **Sensors**, v. 16, n. 1, p. 18, 2015.

MYECHIA MINTER-JORDAN, IRENE S. DAVIS, Z. A. Healthy Mind , Healthy Body : Benefits of Exercise. **Bmj**, p. 0–7, 2013.

NAGY, J. URBANCHEK, J. Catching a wave. [Entrevista a Merrel Noden]. **Sports Ilustrated**, p. 52-56, Abr 1990. Disponível em: < https://www.si.com/vault/1990/04/02/121757/catching-a-wave-breaststroker-mike-barrowman-has-become-a-world-beater-thanks-at-least-in-part-to-his-more-efficient-wave-action-technique >. Acesso em: 20 Abr 2018.

natação – Mais um site Sites Blogs @ MIL. Disponível em: <http://labs.mil.up.pt/blogs/marionpereira/>. Acesso em: 18 jun. 2018.

PANSIOT, J.; LO, B.; YANG, G. Z. **Swimming stroke kinematic analysis with BSN**. 2010 International Conference on Body Sensor Networks, BSN 2010. **Anais**...2010

REYES, R. Evolución de la natación española a través de los Campeonatos de España de natación de invierno y de verano desde 1977 a 1996. 1998.

SAAVEDRA, J. M.; YOLANDA, E.; FERRAN, A. R. A evolução da natação. **Revista Digital - Buenos Aires**, v. 9, n. 66, p. 1–14, 2003.

SHEPHARD, R. J. Benefits of sport and physical activity for the disabled: implications for the individual and for society. **Scandinavian journal of rehabilitation medicine**, v. 23, n. 2, p. 51–9, 1991.

SLAWSON, S. E. et al. The effect of knee angle on force production, in swimming starts, using the OSB11 block. **Procedia Engineering**, v. 34, p. 801–806, 2012.

TANNER, D. Why Swimming is So Good For You. [Entrevista a Markham Heid]. Time Health. Mar 2017. Disponível em: < http://time.com/4688623/swimming-pool-health-benefits/ >. Acesso em: 15 jan 2018.

TOR, E.; PEASE, D. L.; BALL, K. A. Key parameters of the swimming start and their relationship to start performance. **Journal of Sports Sciences**, v. 33, n. 13, p. 1313–1321, 2015.

TSUTSUMI, O. et al. Os Benefícios da Natação Adaptada em Indivíduos com Lesões Neurológicas. **Revista Neurociências**, v. 12, n. 2, p. 82–86, 2004.

User's Manual OSB11 SWIMMING STARTING BLOCK. 2009.

VANTORRE, J.; CHOLLET, D.; SEIFERT, L. **Biomechanical analysis of the swim-start: A reviewJournal of Sports Science and Medicine**, 2014.

10 ANEXOS

10.1 Código implementado nos módulos de medição de ângulos

10.1.1 Para o módulo da nuca – Master

i. Aba principal

```
//Libs do espnow e wifi
#include <esp_now.h>
#include <WiFi.h>
#include "BluetoothSerial.h"
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <math.h>
#include <SimpleKalmanFilter.h>


/*Filtro de Kalman*/
SimpleKalmanFilter simpleKalmanFilter(2, 2, 0.01);
/*BLUETOOTH */
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif



BluetoothSerial SerialBT;
/*IMU */


/* Set the delay between fresh samples */
#define BNO055_SAMPLERATE_DELAY_MS (100)


Adafruit_BNO055 bno = Adafruit_BNO055(55);


/*ESPNOW */


//Canal usado para conexão
```

```
#define CHANNEL 1



/*DEFINICOES */



//definitions
#define CAL_VALUES 200
#define TOTAL_DATA_SENSORS 500
#define GRAUS_180   0
#define GRAUS_135   1
#define GRAUS_90    2
#define GRAUS_45    3
#define TRESHOLD    20
#define SAMPLE_TIME 8
float flexSensCal[4]= {1650,1300,1000,700};
const char sig_start = 'p';
const char sig_send = 's';
const char sig_cal_OK = 'k';
const char sig_cal_imu = 'c';
const char sig_cal_180 = 'q';
const char sig_cal_135 = 'w';
const char sig_cal_90 = 'e';
const char sig_cal_45 = 'r';
const char sig_yes = 'y';
const char sig_no = 'n';
char command;

//PIN DEFINITIONS
const int FLEX_PIN = 36;
/* Connect SCL to analog 22
   Connect SDA to analog 21
   Connect VDD to 3.3-5V DC
```

```
   Connect GROUND to common ground
  */


int LED_BUILTIN = 2;


//MAQUINA DE ESTADOS


#define true  1
#define false 0


#define NUM_ESTADOS 4
#define NUM_EVENTOS 7


// ESTADOS
#define ESPERA   0
#define COLETA_DADOS    1
#define ENVIA_DADOS   2
#define CALIBRACAO  3


// EVENTOS
#define NENHUM_EVENTO     -1
#define BT_SIGNAL_1       0
#define TIMEOUT           1
#define BT_SIGNAL_2       2
#define END_SEND_DATA     3
#define BT_SIGNAL_3       4
#define BT_SIGNAL_CAL_OK   5
#define BT_SIGNAL_4       6


// ACOES
#define NENHUMA_ACAO -1
#define A01  0
#define A02  1
```

```
#define A03  2
#define A04  3
#define A05  4
#define A06  5
#define A07  6

boolean ended_sending_data = false;
boolean timeout = false;
boolean cal_ok = false;
/**************************************************************************
 Estaticos
 **************************************************************************/
  int codigoEvento;
  int codigoAcao;
  int estado;
  int sensores;
  int acao_matrizTransicaoEstados[NUM_ESTADOS][NUM_EVENTOS];
  int proximo_estado_matrizTransicaoEstados[NUM_ESTADOS][NUM_EVENTOS];
  int eventoInterno;
  float imuData[TOTAL_DATA_SENSORS][3];
  int flexSens[TOTAL_DATA_SENSORS];
  float flexSens_Kalman_filtered[TOTAL_DATA_SENSORS];
  float flexSens_filtered[TOTAL_DATA_SENSORS];
  int time_millis[TOTAL_DATA_SENSORS];

//Mac Address dos slaves para os quais iremos enviar a leitura
//Se quiser enviar para todos os Slaves utilize apenas o endereço de broadcast {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}.
//Se quiser enviar para ESPs específicos coloque o Mac Address (obtido através da função WiFi.macAddress())
uint8_t macSlaves[][6] = {
  //Se for enviar para ESPs específicos, coloque cada endereço separado por vírgula
  // {0x24, 0x0A, 0xC4, 0x0E, 0x3F, 0xD1}, {0x24, 0x0A, 0xC4, 0x0E, 0x4E, 0xC3}
  //Se for enviar para todos, apenas deixe o endereço de broadcast {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}
```

```
  {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF}
};
//Criamos uma variável que irá guardar as informações do slave
esp_now_peer_info_t slave;

void setup() {
  // put your setup code here, to run once:
   Serial.begin(115200);

    pinMode(LED_BUILTIN, OUTPUT);
  /*Stating BLuetooth*/
    Serial.println("Starting Bluetooth");
    SerialBT.begin("ESP32Master"); //Bluetooth device name
    Serial.println("The device started, now you can pair it with bluetooth!");
//   put your main code here, to run repeatedly:
    Serial.println("Orientation Sensor Test"); Serial.println("");

    /* Initialise BNO055 */
    if(!bno.begin())
    {
        /* There was a problem detecting the BNO055 ... check your connections */
        Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
        while(1);
    }
    delay(1000);

    /* Use external crystal for better accuracy */
    bno.setExtCrystalUse(true);

    /* Display some basic information on this sensor */
    displaySensorDetails();
```

```
iniciaSistema();
estado = ESPERA;
eventoInterno = NENHUM_EVENTO;
delay(1000); // give me time to bring up serial monitor
Serial.println("ESP32 Analog IN Test");
delay(2000); // give me time to bring up serial monitor

//Colocamos o ESP em modo station
WiFi.mode(WIFI_STA);

//Mostramos no Monitor Serial o Mac Address deste ESP quando em modo station
Serial.print("I'm a Master");
Serial.print("Mac Address in Station: ");
Serial.println(WiFi.macAddress());

  //Chama a função que inicializa o ESPNow
InitESPNow();

//Cálculo do tamanho do array com os mac address dos slaves
//sizeof(macSlaves) retorna a quantidade de bytes que o array macSlaves aponta
//Sabemos que cada mac address é um array de 6 posições e
//cada posição possui sizeof(uint8_t) bytes, então
//a quantidade de slaves é a divisão da quantidade de bytes
//total do array pela quantidade de posições e o resultado
//dessa divisão dividimos novamente por quantos bytes cada posição possui
int slavesCount = sizeof(macSlaves)/6/sizeof(uint8_t);

//Para cada slave
for(int i=0; i<slavesCount; i++){

  //Informamos o canal
  slave.channel = CHANNEL;
  //0 para não usar criptografia ou 1 para usar
```

```
    slave.encrypt = 0;
    //Copia o endereço do array para a estrutura
    memcpy(slave.peer_addr, macSlaves[i], sizeof(macSlaves[i]));
    //Adiciona o slave
    esp_now_add_peer(&slave);
  }

  //Registra o callback que nos informará sobre o status do envio
  //A função que será executada é OnDataSent e está declarada mais abaixo
  esp_now_register_send_cb(OnDataSent);
}



void InitESPNow() {
  //Se a inicialização foi bem sucedida
  if (esp_now_init() == ESP_OK) {
    Serial.println("ESPNow Init Success");
  }
  //Se houve erro na inicialização
  else {
    Serial.println("ESPNow Init Failed");
    ESP.restart();
  }
}

//Função que irá fazer a leitura dos pinos
//que estão no array gpios e enviar os valores
//lidos para os outros ESPs
uint8_t data;

void send(){
  //O endereço de broadcast irá enviar as informações para todos os ESPs
  //Se quiser que a informação vá para ESPs específicos você deve chamar a função
```

```
  //esp_now_send para cada Mac Address, passando o Mac Address como primeiro
  //parâmetro no lugar do broadcast
  uint8_t broadcast[] = {0xFF, 0xFF,0xFF,0xFF,0xFF,0xFF};
  esp_err_t result = esp_now_send(broadcast, &data, sizeof(data));
  Serial.print("Send Status: ");
 if (result == ESP_OK) {
     Serial.println("Success");
   } else if (result == ESP_ERR_ESPNOW_NOT_INIT) {
     // How did we get so far!!
     Serial.println("ESPNOW not Init.");
   } else if (result == ESP_ERR_ESPNOW_ARG) {
     Serial.println("Invalid Argument");
   } else if (result == ESP_ERR_ESPNOW_INTERNAL) {
     Serial.println("Internal Error");
   } else if (result == ESP_ERR_ESPNOW_NO_MEM) {
     Serial.println("ESP_ERR_ESPNOW_NO_MEM");
   } else if (result == ESP_ERR_ESPNOW_NOT_FOUND) {
     Serial.println("Peer not found.");
   } else {
     Serial.println("Not sure what happened");
   }
}



// callback when data is sent from Master to Slave
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
//  char macStr[18];
//  snprintf(macStr, sizeof(macStr), "%02x:%02x:%02x:%02x:%02x:%02x",
//           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3], mac_addr[4], mac_addr[5]);
//  Serial.print("Last Packet Sent to: "); Serial.println(macStr);
//  Serial.print("Last Packet Send Status: "); Serial.println(status ==
ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fail");
}
```

```
void loop() {
  if (SerialBT.available()){
    if (eventoInterno == NENHUM_EVENTO) {
          codigoEvento = obterEvento();
      } else {
          codigoEvento = eventoInterno;
      }
      if (codigoEvento != NENHUM_EVENTO)
      {
         codigoAcao = obterAcao(estado, codigoEvento);
         estado = obterProximoEstado(estado, codigoEvento);
         eventoInterno = executarAcao(codigoAcao);
        Serial.print(F("Estado: "));
        Serial.print(estado);
        Serial.print(F("Evento:"));
        Serial.print(codigoEvento);
        Serial.print(F("Acao: "));
        Serial.print(codigoAcao);
        Serial.println(F(""));
      }
     // delay(10);
  }
}
```

ii. Ações da máquina de estados

```
void imuCalibration(){
  /* Also send calibration data for each sensor. */
      boolean cal_done = false;
      boolean m1,m2,m3,m4;
      m1 = false;
      m2 = false;
      m3 = false;
      m4 = false;
      while(cal_done==false){
```

```
        uint8_t sys, gyro, accel, mag = 0;
        bno.getCalibration(&sys, &gyro, &accel, &mag);
        Serial.print(F("Calibration: "));
        Serial.print(sys, DEC);
        Serial.print(F(" "));
        Serial.print(gyro, DEC);
        Serial.print(F(" "));
        Serial.print(accel, DEC);
        Serial.print(F(" "));
        Serial.println(mag, DEC);
        delay(10);
        if (sys==3) m1=true;
        if (gyro==3) m2=true;
        if (accel==3) m3=true;
        if (mag==3) m4=true;
        if ((m1==true) && (m2==true) && (m3==true) && (m4==true))  {
          cal_done = true;
          digitalWrite(LED_BUILTIN, HIGH);
          delay(3000);
          digitalWrite(LED_BUILTIN, LOW);
        }
      }

}


void flexSensorCalibration(char angle){
  boolean cal_done = false;
  int sample_t,cal_pos,sum, sens_Values[CAL_VALUES];
  float n = CAL_VALUES;
  float mean,treshold;
  sample_t=10;
  sum=0;
```

```
mean=0;
int attempt = 0;


Serial.println("Iniciando Calibracao \n");
    switch (angle) {
    case 'q':
      cal_pos = GRAUS_180;
      Serial.println("Calibracao para 180 graus \n");
    break;
    case 'w':
      cal_pos = GRAUS_135;
      Serial.println("Calibracao para 135 graus \n");
    break;
    case 'e':
      cal_pos = GRAUS_90;
      Serial.println("Calibracao para 90 graus \n");
    break;
    case 'r':
      cal_pos = GRAUS_45;
      Serial.println("Calibracao para 90 graus \n");
    break;
  }
    while (cal_done == false && attempt<3){
         //aquisicao dos dados e media
        sum=0;
        for (int i=0; i<n; i++){
          int flexADC = analogRead(FLEX_PIN);
          sens_Values[i]= simpleKalmanFilter.updateEstimate(flexADC);
          sum=sum+sens_Values[i];
          delay(sample_t);
        }
        Serial.println(attempt);
        mean = (float) sum/n;
```

```
    Serial.print("Media: ");
    Serial.print(mean);
     //variancia
    float v_sum=0;
    float desv_pad=0;
    for (int i=0; i<n; i++){
      v_sum = v_sum + (sens_Values[i]-mean)*(sens_Values[i]-mean);
    }
    //verificação se os dados n estao variando muito
      desv_pad = sqrt(v_sum/n);
    Serial.print(F("\n Desvio Padrão: "));
    Serial.println(desv_pad);
    if (desv_pad<TRESHOLD){
      flexSensCal[cal_pos] = mean;
      Serial.println(flexSensCal[cal_pos]);
      cal_done = true;

    }else{
      Serial.println("Calibracao Incorreta \n");

    }
    attempt++;
}
if (cal_done==true){
  Serial.println("Calibracao Completa \n");
  digitalWrite(LED_BUILTIN, HIGH);
  delay(3000);
  digitalWrite(LED_BUILTIN, LOW);
}else{
  for (int i=0;i<6;i++){
    blink_led();
  }
}
```

```
}


void getDataSensors(){
  int flexADC;
  float x,y,z;
  long refresh_time = millis()+SAMPLE_TIME;
  int i=0;
  int t1;
  data = 'p';
  send();
  send();
  digitalWrite(LED_BUILTIN, HIGH);
  while (i<TOTAL_DATA_SENSORS){
    t1 = millis();
    if (t1 > refresh_time) {
      sensors_event_t event;
      bno.getEvent(&event);
      flexADC = analogRead(FLEX_PIN);
      x = (float)event.orientation.x;
      y = (float)event.orientation.y;
      z = (float)event.orientation.z;
      flexSens[i] = flexADC;
      imuData[i][0] = x;
      imuData[i][1] = y;
      imuData[i][2] = z;
      time_millis[i] = t1;
      i++;
      refresh_time = millis() + SAMPLE_TIME;
    }
  }
  digitalWrite(LED_BUILTIN, LOW);
```

```
  delay(100);
  data = 'y';
  send();
  int t0 = time_millis[0];
  for (int i=0; i<TOTAL_DATA_SENSORS; i++){
    flexSens_Kalman_filtered[i] = simpleKalmanFilter.updateEstimate(flexSens[i]);
    time_millis[i] = time_millis[i]-t0;
    flexSens_filtered[i] = get_angle(flexSens_Kalman_filtered[i]);
  }
}
void sendData(){
  delay(1000);
  char mystring[33];

  for (int i=0;i<TOTAL_DATA_SENSORS;i++){
    sprintf(mystring, "%04d;%06.2f;%06.2f;%06.2f;%06.2f",time_millis[i],
flexSens_filtered[i],imuData[i][0],imuData[i][1],imuData[i][2]);
    SerialBT.println(mystring);
    delay(75);
  }
}

float get_angle(int value){
  float  angle;
  if (value<=flexSensCal[GRAUS_90]){
     angle = 45 + (value - flexSensCal[GRAUS_45])*((float)(90-45)/(flexSensCal[GRAUS_90]-
flexSensCal[GRAUS_45]));
  }
  if ((value>flexSensCal[GRAUS_90]) || (value<=flexSensCal[GRAUS_135])){
     angle = 90 + (value - flexSensCal[GRAUS_90])*((float)(135-90)/(flexSensCal[GRAUS_135]-
flexSensCal[GRAUS_90]));
  }
  if (value>flexSensCal[GRAUS_135]){
     angle = 135 + (value - flexSensCal[GRAUS_135])*((float)(180-
135)/(flexSensCal[GRAUS_180]-flexSensCal[GRAUS_135]));
```

```
    }
    return angle;
}
```

iii. Funções da máquina de estado

```
/*****************************************************************************
 executarAcao
 Executa uma acao
 Parametros de entrada:
   (int) codigo da acao a ser executada
 Retorno: (int) codigo do evento interno ou NENHUM_EVENTO
*****************************************************************************/
int executarAcao(int codigoAcao)
{
   int retval;

   retval = NENHUM_EVENTO;
   if (codigoAcao == NENHUMA_ACAO)
      return retval;

   switch(codigoAcao)
   {
   case A01:
      getDataSensors();
      Serial.print("action01");
      timeout = true;
      break;
   case A02:
      Serial.print("action02");
      timeout = false;
      break;
   case A03:
      Serial.print("action03");
```

```
        sendData();
        ended_sending_data = true;
        break;
    case A04:
        Serial.print("action04");
        ended_sending_data = false;
        break;
    case A05:
        Serial.print("action05");
        if ((command == sig_cal_180)|| (command == sig_cal_135)|| (command == sig_cal_90)||
(command == sig_cal_45))
        {
           flexSensorCalibration(command);
        }else{
          if (command == sig_cal_imu){
            imuCalibration();
          }
        }
        cal_ok = true;
        break;

    case A06:
        Serial.print("action06");
        cal_ok = false;
        break;

    } // switch

    return retval;
} // executarAcao

/*****************************************************************************
 iniciaMaquina de Estados
 Carrega a maquina de estados
```

```
 Parametros de entrada: nenhum
 Retorno: nenhum
*************************************************************************/

void iniciaMaquinaEstados()
{
  int i;
  int j;

  for (i=0; i < NUM_ESTADOS; i++) {
    for (j=0; j < NUM_EVENTOS; j++) {
       acao_matrizTransicaoEstados[i][j] = NENHUMA_ACAO;
       proximo_estado_matrizTransicaoEstados[i][j] = i;
    }
  }
  proximo_estado_matrizTransicaoEstados[ESPERA][BT_SIGNAL_1] = COLETA_DADOS;
  acao_matrizTransicaoEstados[ESPERA][BT_SIGNAL_1] = A01;

  proximo_estado_matrizTransicaoEstados[COLETA_DADOS][TIMEOUT] = ESPERA;
  acao_matrizTransicaoEstados[COLETA_DADOS][TIMEOUT] = A02;

  proximo_estado_matrizTransicaoEstados[ESPERA][BT_SIGNAL_2] = ENVIA_DADOS;
  acao_matrizTransicaoEstados[ESPERA][BT_SIGNAL_2] = A03;

  proximo_estado_matrizTransicaoEstados[ENVIA_DADOS][END_SEND_DATA] = ESPERA;
  acao_matrizTransicaoEstados[ENVIA_DADOS][END_SEND_DATA] = A04;

  proximo_estado_matrizTransicaoEstados[ESPERA][BT_SIGNAL_3] = CALIBRACAO;
  acao_matrizTransicaoEstados[ESPERA][BT_SIGNAL_3] = A05;

  proximo_estado_matrizTransicaoEstados[CALIBRACAO][BT_SIGNAL_CAL_OK] = ESPERA;
  acao_matrizTransicaoEstados[CALIBRACAO][BT_SIGNAL_CAL_OK] = A06;
```

```
  proximo_estado_matrizTransicaoEstados[ESPERA][BT_SIGNAL_4] = CALIBRACAO;
  acao_matrizTransicaoEstados[ESPERA][BT_SIGNAL_4] = A07;

}


/*****************************************************************************
 iniciaSistema
 Inicia o sistema ...
 Parametros de entrada: nenhum
 Retorno: nenhum
*****************************************************************************/
void iniciaSistema()
{
   iniciaMaquinaEstados();
} // initSystem

/*****************************************************************************
 obterEvento
 Obtem um evento, que pode ser da IHM ou do alarme
 Parametros de entrada: nenhum
 Retorno: codigo do evento
*****************************************************************************/


int decodificarBT_SIGNAL_1()
{
    if (command  == 'p')
    {
      return true;
    }
```

```
    return false;
}//decodificarBT_SIGNAL_1


int decodificarTIMEOUT()
{
    if (timeout == true)
    {
            return true;
    }
    return false;
}//decodificarTIMEOUT


int decodificarBT_SIGNAL_2()
{
    if (command == 's')
    {
        return true;
    }
    return false;
}//decodificarBT_SIGNAL_2


int decodificarEND_SEND_DATA()
{
    if (ended_sending_data == true)
    {
            return true;
    }
    return false;
}//decodificarEND_SEND_DATA


int decodificarBT_SIGNAL_3()
{
    if ((command == sig_cal_imu) || (command == sig_cal_180)|| (command == sig_cal_135)||
(command == sig_cal_90)|| (command == sig_cal_45))
```

```
    {
        return true;
    }
    return false;
}//decodificarBT_SIGNAL_3



int decodificarBT_SIGNAL_CAL_OK()
{
    if (cal_ok == true)
    {
        return true;
    }
    return false;
}//BT_SIGNAL_CAL_OK


char get_command(){
  //Serial.println("Write Command: \n");
  char aux;
  if(SerialBT.available()){
    aux = (char)SerialBT.read();
    if ((aux == sig_start) || (aux == sig_send) || (aux == sig_cal_imu) || (aux == sig_cal_180) ||
(aux == sig_cal_135) ||  (aux == sig_cal_90) || (aux == sig_cal_45)){
      command = aux;
      Serial.println("Command: ");
      Serial.println((char)command);
      Serial.println("\n ");
      return command;
    }
    else {return '-';}


  //get_command
  }
}
```

```
int obterEvento()
{
  int retval = NENHUM_EVENTO;
  if (decodificarTIMEOUT())
    return TIMEOUT;
  if (decodificarEND_SEND_DATA())
    return END_SEND_DATA;
  if (decodificarBT_SIGNAL_CAL_OK())
    return BT_SIGNAL_CAL_OK;
  command = get_command();
  timeout=false;
  ended_sending_data = false;
  cal_ok = false;
  if (decodificarBT_SIGNAL_1())
    return BT_SIGNAL_1;
  if (decodificarBT_SIGNAL_2())
    return BT_SIGNAL_2;
  if (decodificarBT_SIGNAL_3())
    return BT_SIGNAL_3;

  return retval;

} // obterEvento




/*************************************************************************
 obterAcao
 Obtem uma acao da Matriz de transicao de estados
 Parametros de entrada: estado (int)
```

```
                    evento (int)
 Retorno: codigo da acao
*************************************************************************/
int obterAcao(int estado, int codigoEvento) {
  return acao_matrizTransicaoEstados[estado][codigoEvento];
} // obterAcao




/*************************************************************************
 obterProximoEstado
 Obtem o proximo estado da Matriz de transicao de estados
 Parametros de entrada: estado (int)
                    evento (int)
 Retorno: codigo do estado
*************************************************************************/
int obterProximoEstado(int estado, int codigoEvento) {
  return proximo_estado_matrizTransicaoEstados[estado][codigoEvento];
} // obterAcao
```

10.1.2 Para o módulo do braço e perna – Slaves

Necessário mudar apenas a aba principal

i. Aba principal

```
//Libs do espnow e wifi
#include <esp_now.h>
#include <WiFi.h>
#include "BluetoothSerial.h"
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <math.h>
#include <SimpleKalmanFilter.h>
```

```
/*Filtro de Kalman*/
SimpleKalmanFilter simpleKalmanFilter(2, 2, 0.01);
/*BLUETOOTH */
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif


BluetoothSerial SerialBT;
/*IMU */

/* Set the delay between fresh samples */
#define BNO055_SAMPLERATE_DELAY_MS (100)

Adafruit_BNO055 bno = Adafruit_BNO055(55);

/*ESPNOW */

//Canal usado para conexão
#define CHANNEL 1



/*DEFINICOES */



//definitions
#define CAL_VALUES 200
#define TOTAL_DATA_SENSORS 500
#define GRAUS_180   0
#define GRAUS_135   1
#define GRAUS_90    2
#define GRAUS_45    3
```

```
#define TRESHOLD    20
#define SAMPLE_TIME 8
float flexSensCal[4]= {1650,1300,1000,700};
const char sig_start = 'p';
const char sig_send = 's';
const char sig_cal_OK = 'k';
const char sig_cal_imu = 'c';
const char sig_cal_180 = 'q';
const char sig_cal_135 = 'w';
const char sig_cal_90 = 'e';
const char sig_cal_45 = 'r';
const char sig_yes = 'y';
const char sig_no = 'n';
char command;
char command_esp;
//PIN DEFINITIONS
const int FLEX_PIN = 36;
/* Connect SCL to analog 22
   Connect SDA to analog 21
   Connect VDD to 3.3-5V DC
   Connect GROUND to common ground
  */

int LED_BUILTIN = 2;

//MAQUINA DE ESTADOS

#define true  1
#define false 0

#define NUM_ESTADOS 4
#define NUM_EVENTOS 7
```

```
// ESTADOS
#define ESPERA   0
#define COLETA_DADOS    1
#define ENVIA_DADOS   2
#define CALIBRACAO  3

// EVENTOS
#define NENHUM_EVENTO     -1
#define BT_SIGNAL_1       0
#define TIMEOUT           1
#define BT_SIGNAL_2       2
#define END_SEND_DATA     3
#define BT_SIGNAL_3       4
#define BT_SIGNAL_CAL_OK   5
#define BT_SIGNAL_4       6

// ACOES
#define NENHUMA_ACAO -1
#define A01  0
#define A02  1
#define A03  2
#define A04  3
#define A05  4
#define A06  5
#define A07  6

boolean ended_sending_data = false;
boolean timeout = false;
boolean cal_ok = false;
/*****************************************************************************
 Estaticos
 *****************************************************************************/
  int codigoEvento;
```

```
  int codigoAcao;
  int estado;
  int sensores;
  int acao_matrizTransicaoEstados[NUM_ESTADOS][NUM_EVENTOS];
  int proximo_estado_matrizTransicaoEstados[NUM_ESTADOS][NUM_EVENTOS];
  int eventoInterno;
  float imuData[TOTAL_DATA_SENSORS][3];
  int flexSens[TOTAL_DATA_SENSORS];
  float flexSens_Kalman_filtered[TOTAL_DATA_SENSORS];
  float flexSens_filtered[TOTAL_DATA_SENSORS];
  int time_millis[TOTAL_DATA_SENSORS];



void InitESPNow() {
  //Se a inicialização foi bem sucedida
  if (esp_now_init() == ESP_OK) {
    Serial.println("ESPNow Init Success");
  }
  //Se houve erro na inicialização
  else {
    Serial.println("ESPNow Init Failed");
    ESP.restart();
  }
}
// config AP SSID
void configDeviceAP() {
  char* SSID = "Slave_1";
  bool result = WiFi.softAP(SSID, "Slave_1_Password", CHANNEL, 0);
  if (!result) {
    Serial.println("AP Config failed.");
  } else {
    Serial.println("AP Config Success. Broadcasting with AP: " + String(SSID));
  }
```

```
}
int t1 = millis();
int t2 = t1+10;
void setup() {
  Serial.begin(115200);
  Serial.println("ESPNow/Basic/Slave Example");
  Serial.println("i'M A SLAVE");
  pinMode(LED_BUILTIN, OUTPUT);

  /*Stating BLuetooth*/
    Serial.println("Starting Bluetooth");
//    SerialBT.begin("ESP32Braco"); //Bluetooth device name
    SerialBT.begin("ESP32Perna"); //Bluetooth device name
    Serial.println("The device started, now you can pair it with bluetooth!");
//   put your main code here, to run repeatedly:
    Serial.println("Orientation Sensor Test"); Serial.println("");



    /* Initialise BNO055 */
    if(!bno.begin())
    {
        /* There was a problem detecting the BNO055 ... check your connections */
        Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
        while(1);
    }
    delay(1000);

    /* Use external crystal for better accuracy */
    bno.setExtCrystalUse(true);

    /* Display some basic information on this sensor */
    displaySensorDetails();
```

```
    iniciaSistema();
    estado = ESPERA;
    eventoInterno = NENHUM_EVENTO;
    delay(1000); // give me time to bring up serial monitor
    Serial.println("ESP32 Analog IN Test");
    delay(2000); // give me time to bring up serial monitor

  //Set device in AP mode to begin with
  WiFi.mode(WIFI_STA);
  // configure device AP mode
//  configDeviceAP();
  // This is the mac address of the Slave in AP Mode
  Serial.print("AP MAC: "); Serial.println(WiFi.softAPmacAddress());
  // Init ESPNow with a fallback logic
  InitESPNow();
  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info.
  esp_now_register_recv_cb(OnDataRecv);
}
// callback when data is recv from Master
int trigger_on = 0;
void OnDataRecv(const uint8_t *mac_addr, const uint8_t *data, int data_len) {
 command_esp = (char) *data;
  if (command_esp == 'p'){
    if (trigger_on == 0){
      trigger_on = 1;
    }
  }
//  if (trigger_on == 1){
//    trigger_on = 0;
//  }

}
```

```
void loop(){


if (trigger_on == 1){
  getDataSensors();
  trigger_on = 0;
  delay(1000);
}
if (SerialBT.available() && (trigger_on == 0)){
    if (eventoInterno == NENHUM_EVENTO) {
         codigoEvento = obterEvento();
     } else {
         codigoEvento = eventoInterno;
     }
     if (codigoEvento != NENHUM_EVENTO)
     {
        codigoAcao = obterAcao(estado, codigoEvento);
        estado = obterProximoEstado(estado, codigoEvento);
        eventoInterno = executarAcao(codigoAcao);
       Serial.print(F("Estado: "));
       Serial.print(estado);
       Serial.print(F("Evento:"));
       Serial.print(codigoEvento);
       Serial.print(F("Acao: "));
       Serial.print(codigoAcao);
       Serial.println(F(""));
     }
  }
}
```

10.2 Código implementado no controle de disparo da câmera

```
#include <esp_now.h>
#include <WiFi.h>

#define CHANNEL 1
 int TRIGGER = 4;
 int LED_BUILTIN = 2;
// Init ESP Now with fallback
void InitESPNow() {
  WiFi.disconnect();
  if (esp_now_init() == ESP_OK) {
    Serial.println("ESPNow Init Success");
  }
  else {
    Serial.println("ESPNow Init Failed");
    // Retry InitESPNow, add a counte and then restart?
    // InitESPNow();
    // or Simply Restart
    ESP.restart();
  }
}

void pull_trigger(){
digitalWrite(2, HIGH);
digitalWrite(TRIGGER, HIGH);   // turn the LED on (HIGH is the voltage level)
delay(100);
digitalWrite(TRIGGER, LOW);    // turn the LED off by making the voltage LOW
digitalWrite(2, LOW);
delay(8000);
digitalWrite(2, HIGH);
digitalWrite(TRIGGER, HIGH);   // turn the LED on (HIGH is the voltage level)
delay(100);
digitalWrite(TRIGGER, LOW);    // turn the LED off by making the voltage LOW
```

```
digitalWrite(2, LOW);
}

// config AP SSID
void configDeviceAP() {
  char* SSID = "Slave_1";
  bool result = WiFi.softAP(SSID, "Slave_1_Password", CHANNEL, 0);
  if (!result) {
    Serial.println("AP Config failed.");
  } else {
    Serial.println("AP Config Success. Broadcasting with AP: " + String(SSID));
  }
}
int t0;
void setup() {
  Serial.begin(115200);
  Serial.println("ESPNow/Basic/Slave Example");
  Serial.println("i'M A SLAVE");
  pinMode(TRIGGER, OUTPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  //Set device in AP mode to begin with
  WiFi.mode(WIFI_STA);
  // configure device AP mode
//  configDeviceAP();
  // This is the mac address of the Slave in AP Mode
  Serial.print("AP MAC: "); Serial.println(WiFi.softAPmacAddress());
  // Init ESPNow with a fallback logic
  InitESPNow();
  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info.
  esp_now_register_recv_cb(OnDataRecv);
}
char command;
```

```
// callback when data is recv from Master
boolean trigger_on = false;
void OnDataRecv(const uint8_t *mac_addr, const uint8_t *data, int data_len) {
   command = (char) *data;

  if ((trigger_on == false) && (command == 'p')){
    trigger_on = true;
    pull_trigger();
    trigger_on = false;
    command = '-';
  }
  Serial.print("Last Packet Recv Data: "); Serial.println(command);
  Serial.println("");




}

void loop() {
}
```

10.3 Código fonte da interface gráfica

A seguir encontra-se o código fonte do software apresentado na monografia:

```
function varargout = PanelSwap(varargin)
% PANELSWAP MATLAB code for PanelSwap.fig
%      PANELSWAP, by itself, creates a new PANELSWAP or raises the
existing
%      singleton*.
%
%      PANELSWAP('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in PANELSWAP.M with the given input
arguments.
%
```

```
%      PANELSWAP('Property','Value',...) creates a new PANELSWAP or
raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before PanelSwap_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to PanelSwap_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Last Modified by GUIDE v2.5 28-Oct-2018 20:07:07


% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @PanelSwap_OpeningFcn, ...
                   'gui_OutputFcn',  @PanelSwap_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end


if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before PanelSwap is made visible.
function PanelSwap_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to PanelSwap (see VARARGIN)

% Choose default command line output for PanelSwap
handles.output = hObject;

%inicio dos axes de topo-
https://www.youtube.com/watch?v=w0LiVJZ0NiE&index=17&list=PLtJVdgCW3sr_wP
xMkEtIjUwddFkQgLE3-
axes(handles.axes1)
imshow('topo.PNG')
axes(handles.axes3)
imshow('topo.PNG')

%%% record starting positions and limit the seens area
pos = get(handles.figure1,'position');
set(handles.figure1,'position',[pos(1:2) 5.5 6.25])

handles.panelPos = [get(handles.uipanel1,'position');...
    get(handles.uipanel2,'position');]
handles.currentPanel = 1;

% Update handles structure
guidata(hObject, handles);

% Inicicaliza o slider
global taq
taq = 100; % taxa de aquisicao da IMU
global duracao % tempo de analise em segundos
duracao = 5;
```

```
tamanho = duracao*taq;

step1=1/tamanho; % passo

step2=0.02; % tamanho do marcador do slider

steps = [step1, step2];

set(handles.slider1, 'SliderStep', steps);

set(handles.slider1, 'max', tamanho);


% Inicializa o popup menu com os atletas analisadaos anteriormente

[num,dados,raw] = xlsread('Banco de Dados.xls'); % divide nas 3 partes

dados(1,:)=[]; % elimina a primeira linha que sao os headers

nomes = dados(:,2); % pegar apenas a segunda coluna (nomes)

uni_nomes = unique(nomes.','rows').' ;% vetor com observacoes unicas dos
nomes - https://www.mathworks.com/matlabcentral/answers/340222-remove-
duplicates-from-vector

uni_nomes = sort(uni_nomes); % ordem alfabetica

lista = ['Atleta já analisado'; uni_nomes]; % acrescenta o titulo das
escolhas 'Atleta já analisado'


set(handles.popupmenu1, 'String', lista)

set(handles.popupmenu2, 'String', 'Data da análise')

set(handles.popupmenuAngulo, 'String', 'Escolha o ângulo')


    f = waitbar(0,'0%','Name','Conectando sensores via Bluetooth',...

    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');


    setappdata(f,'canceling',0);


    for step = 1:3

        if step == 1

            btInfo = instrhwinfo('Bluetooth','ESP32Master') %
propriedades da porta em que esta conectado o ESP32Module1 via bluetooth

            channel = str2num(btInfo.Channels{1}) % declaração do canal
da porta como num

            global master

            master = Bluetooth('ESP32Master', channel)

        elseif step == 2
```

```
            btInfo = instrhwinfo('Bluetooth','ESP32Braco'); %
propriedades da porta em que esta conectado o ESP32Module1 via bluetooth
            channel = str2num(btInfo.Channels{1}); % declaração do canal
da porta como num
            global braco
            braco = Bluetooth('ESP32Braco', channel);
        elseif step == 3
            btInfo = instrhwinfo('Bluetooth','ESP32Perna'); %
propriedades da porta em que esta conectado o ESP32Module1 via bluetooth
            channel = str2num(btInfo.Channels{1}); % declaração do canal
da porta como num
            global perna
            perna = Bluetooth('ESP32Perna', channel);
        end
        % Check for clicked Cancel button
        if getappdata(f,'canceling')
            break
        end

        % Update waitbar and message
        valor = step/3*100;
        waitbar(step/3,f,strcat(sprintf('%12.2f',valor),'%'));
    end

    delete(f)

% UIWAIT makes PanelSwap wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = PanelSwap_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in botaoDados.
function botaoDados_Callback(hObject, eventdata, handles)
% hObject    handle to botaoDados (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
swapPanPos(hObject, eventdata, handles,1)

% --- Executes on button press in pushbutton5.
function botaoResultados_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
swapPanPos(hObject, eventdata, handles,2)

function swapPanPos(hObject, eventdata, handles,POI) %
https://www.mathworks.com/matlabcentral/answers/167461-using-a-main-gui-
window-to-open-new-ones

temp= handles.panelPos(POI,:);  %make a temp position location
handles.panelPos(POI,:)=handles.panelPos(handles.currentPanel,:);
%perform position swap
handles.panelPos(handles.currentPanel,:)=temp;

%update panel positions
set(handles.uipanel1,'position',handles.panelPos(1,:))
set(handles.uipanel2,'position',handles.panelPos(2,:))

function variaAtleta_Callback(hObject, eventdata, handles)
% hObject    handle to variaAtleta (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of variaAtleta as text
%        str2double(get(hObject,'String')) returns contents of
variaAtleta as a double


% --- Executes during object creation, after setting all properties.
function variaAtleta_CreateFcn(hObject, eventdata, handles)
% hObject    handle to variaAtleta (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function variaTecnico_Callback(hObject, eventdata, handles)
% hObject    handle to variaTecnico (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of variaTecnico as text
%        str2double(get(hObject,'String')) returns contents of
variaTecnico as a double


% --- Executes during object creation, after setting all properties.
function variaTecnico_CreateFcn(hObject, eventdata, handles)
% hObject    handle to variaTecnico (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


function variaEquipe_Callback(hObject, eventdata, handles)
% hObject    handle to variaEquipe (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of variaEquipe as text
%        str2double(get(hObject,'String')) returns contents of
variaEquipe as a double


% --- Executes during object creation, after setting all properties.
function variaEquipe_CreateFcn(hObject, eventdata, handles)
% hObject    handle to variaEquipe (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function variaEstilo_Callback(hObject, eventdata, handles)
% hObject    handle to variaEstilo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of variaEstilo as text
%        str2double(get(hObject,'String')) returns contents of
variaEstilo as a double


% --- Executes during object creation, after setting all properties.
function variaEstilo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to variaEstilo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


% --- Executes on button press in botaoAnalisar.

function botaoAnalisar_Callback(hObject, eventdata, handles)

% hObject    handle to botaoAnalisar (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


% se algum dos campos não estiverem preenchidos, ou seja, se algum
estiver com o default retorna uma mensagem de erro

if(get(handles.popupmenu2, 'Value') == 1 &
strcmp(get(handles.variaAtleta, 'string'),'Atleta'))

    uiwait(msgbox({'Preencha os campos:',...

                   'Atleta e data acima para análise já realizada',...

                   'ou',...

                   'Todos os campos para nova análise'}));
%https://www.mathworks.com/help/matlab/ref/msgbox.html


% se campos iniciais de atelta E data dos popmenus estiverem escolhidos
serão usados para análise anterior

elseif(get(handles.popupmenu1, 'Value') ~= 1 & get(handles.popupmenu2,
'Value') ~= 1)


    atletas = (get(handles.popupmenu1, 'String'));

    atl = atletas(get(handles.popupmenu1, 'Value'),1);


    datas = get(handles.popupmenu2, 'String');

    data = datas(get(handles.popupmenu2, 'Value'),1);
```

```
    % cria vetor com as duas informacoes necessarias (nome, data) para
percorrer as pastas

    pasta = strcat(atl,'\',data);

    set(handles.botaoAnalisar, 'UserData', pasta) %
https://www.mathworks.com/matlabcentral/answers/38419-get-value-from-
popup-menu-and-use-it-in-different-function


% se ao menos o nome estiver preenchido sera criada uma nova análise

else

    %dados do atleta

    atl = get(handles.variaAtleta, 'string');

    data = char(datetime('now','TimeZone','local','Format','d_MMM_y HH-
mm-ss')); % data da analisec(dia-mês-ano hora:min:seg) -
https://www.mathworks.com/help/matlab/ref/datetime.html

    estilo = get(handles.variaEstilo, 'string');

    equipe = get(handles.variaEquipe, 'string');

    tecnico = get(handles.variaTecnico, 'string');

    id = strcat(atl, data); % o identificador de cada analise sera o nome
concatenado a data realizada

    novo = {id ,atl, data, estilo, equipe, tecnico};


    [num,antigo,raw] = xlsread('Banco de Dados.xls');

    bd = [raw; novo]; % acrescenta os novos dados

    xlswrite('Banco de Dados.xls', bd); % sobreescreve o banco de dados


    %mkdir nome/data - cria 'data' numa pasta 'nome'

    pasta = strcat(atl,'\',data); % cria 'atl' (caso nao exista) e,
dentro dela, 'data'

    set(handles.botaoAnalisar, 'UserData', pasta)


    mkdir (pasta) % cria pasta atleta e dentro dela a pasta data -
https://www.mathworks.com/help/matlab/ref/movefile.html

    mkdir Imagens % cria pasta Imagnes

    mkdir Video % cria pasta Video

         movefile video.MOV Video % Coloca video na pasta 'Video'

    mkdir Gráficos % cria pasta Gráficos

%         % move todos os gráficos gerados para dentro de Gráficos
```

```
        movefile master.xls Gráficos
        movefile braco.xls Gráficos
        movefile perna.xls Gráficos

    cd (atl) % torna a pasta atual a da data da analise
        % move as pastas da análise para a atleta atual
        movefile ../Gráficos
        movefile ../Video
        movefile ../Imagens

        cd (data) % torna a pasta atual a da data da analise
            % move as pastas da análise para a atleta atual
            movefile ../Gráficos
            movefile ../Video
            movefile ../Imagens

    % Quebra do vídeo em frames
    cd ..
    cd ..
    video_caminho = strcat(pasta, '\Video\','video.MOV');
    a = VideoReader (video_caminho);

    imagens_caminho = strcat(pasta, '\', 'Imagens');

    cd (imagens_caminho) %
https://www.mathworks.com/help/matlab/ref/cd.html

    f = waitbar(0,'0%','Name','Convertendo vídeo em frames',...  %
https://www.mathworks.com/help/matlab/ref/waitbar.html
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');

    setappdata(f,'canceling',0);

    nframes = a.NumberOfFrames;
    for step = 1:nframes
```

```
        filename=strcat('frame',num2str(step),'.jpg');
%https://www.youtube.com/watch?v=AI-1ch6CHkI
        b = read(a, step);
        % b = imrotate(b, 180); % rotaciona a imagem 180 graus
        imwrite(b,filename);

        % Check for clicked Cancel button
        if getappdata(f,'canceling')
            break
        end

        % Update waitbar and message
        valor = step/nframes*100;
        waitbar(step/nframes,f,strcat(sprintf('%12.2f',valor),'%'));
    end

    delete(f)

    % atl/data/Imagens
    cd .. % volta para data
    cd .. % volta para atl
    cd .. % Volta para inicial

        uiwait(msgbox({'Análise gerada com sucesso!','Acesse-a na aba
"Resultados".'}));

end

% Inicializa o popup menu com os atletas analisadaos anteriormente
[num,dados,raw] = xlsread('Banco de Dados.xls'); % divide nas 3 partes
dados(1,:)=[]; % elimina a primeira linha que sao os headers
nomes = dados(:,2); % pegar apenas a segunda coluna (nomes)
```

```
uni_nomes = unique(nomes.','rows').' ;% vetor com observacoes unicas dos
nomes - https://www.mathworks.com/matlabcentral/answers/340222-remove-
duplicates-from-vector

uni_nomes = sort(uni_nomes); % ordem alfabetica

lista = ['Atleta já analisado'; uni_nomes]; % acrescenta o titulo das
escolhas 'Atleta já analisado'


set(handles.popupmenu1, 'String', lista)

set(handles.popupmenu1, 'Value', 1)

set(handles.popupmenu2, 'String', 'Data da análise')

set(handles.popupmenu2, 'Value', 1)


% --- Executes on button press in botaoImportar.

function botaoImportar_Callback(hObject, eventdata, handles) % entra o
disparo feito pelo Alyson!!!

% hObject    handle to botaoImportar (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

global duracao % tempo de filmagem

global taq % taxa de aquisição da imu

tot = taq*duracao; % numero de frames (quadros)


% Pegando o locais do blueTooth- propriedades da porta em que esta
conectado o bluetooth


% Graficos do braco

global braco

%fclose(braco)

fopen(braco) % abre a troca de informações

aux2=[];

fprintf(braco,'s')

    f = waitbar(0,'0%','Name','Importando dados braço ',...

    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');

    setappdata(f,'canceling',0);

for cont = 1:tot

            if getappdata(f,'canceling')
```

```
                break
            end
            % Update waitbar and message
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    % ver se é possivel 's' pedir apenas a proxima informacao-entra no
for
    aux1 = fscanf(braco,'%s');
    pause(.025)
    aux2 = [aux2;aux1];
    cont
end
    delete(f)
bracoArray = [];
    f = waitbar(0,'0%','Name','Gerando arquivo braço',...
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
    setappdata(f,'canceling',0);
for cont = 1:tot
            if getappdata(f,'canceling')
                break
            end
            % Update waitbar and message
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    bracoArray=[bracoArray;strsplit(aux2(cont,:),';')]; % quebra cada
linha em 5 vetores
end
    delete(f)
xlswrite('braco.xls',bracoArray);
fclose(braco)


% % Graficos da perna
global perna
fopen(perna) % abre a troca de informações
aux2=[];
```

```
fprintf(perna,'s')
    f = waitbar(0,'0%','Name','Importando dados perna ',...
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
    setappdata(f,'canceling',0);
for cont = 1:tot
            if getappdata(f,'canceling')
                break
            end
            % Update waitbar and message
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    % ver se é possivel 's' pedir apenas a proxima informacao-entra no
for
    aux1 = fscanf(perna,'%s');
    pause(.025)
    aux2 = [aux2;aux1];
    cont;
end
    delete(f)
pernaArray = [];
    f = waitbar(0,'0%','Name','Gerando arquivo perna',...
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
    setappdata(f,'canceling',0);
for cont = 1:tot
            if getappdata(f,'canceling')
                break
            end
            % Update waitbar and message
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    pernaArray=[pernaArray;strsplit(aux2(cont,:),';')]; % quebra cada
linha em 5 vetores
end
    delete(f)
xlswrite('perna.xls',pernaArray);
```

```
fclose(perna)


% Graficos do nuca - master
global master
fopen(master) % abre a troca de informações
aux2=[];
fprintf(master,'s')
    f = waitbar(0,'0%','Name','Importando dados nuca ',...
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
    setappdata(f,'canceling',0);
for cont = 1:tot
            if getappdata(f,'canceling')
                break
            end
            % Update waitbar and message
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    % ver se é possivel 's' pedir apenas a proxima informacao-entra no
for
    aux1 = fscanf(master,'%s');
    pause(.025)
    aux2 = [aux2;aux1];
    cont;
end
    delete(f)
masterArray = []
    f = waitbar(0,'0%','Name','Gerando arquivo nuca',...
    'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
    setappdata(f,'canceling',0);
for cont = 1:tot
            if getappdata(f,'canceling')
                break
            end
            % Update waitbar and message
```

```
            valor = cont/tot*100;
            waitbar(cont/tot,f,strcat(sprintf('%12.2f',valor),'%'));
    masterArray=[masterArray;strsplit(aux2(cont,:),';')]; % quebra cada
linha em 5 vetores
end
    delete(f)
xlswrite('master.xls',masterArray);
fclose(master)

uiwait(msgbox({'Dados importados com sucesso'}));

% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over
botaoAnalisar.
function botaoAnalisar_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to botaoAnalisar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
%https://www.youtube.com/watch?v=rP27Z-2RIWw
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of
slider

% importacao do gráfico selecionado
eixos = get(handles.graficos, 'UserData');
x=eixos(:,1);
y=eixos(:,2);
```

```
% importacao do caminho dos dados

pasta=get(handles.botaoAnalisar, 'UserData');


% Variação da imagem %https://www.youtube.com/watch?v=Ta1uhGEJFBE

global fps

global taq

a=round(get(handles.slider1,'Value')); % posicao do slider

num_img = round((fps/taq)*a)+1; % +1 corrige a = 0 e permite que haja o
primeiro print

axes(handles.axes6)

imshow(char(strcat(pasta,'\Imagens\frame',num2str(num_img-30),'.jpg'))) %
retirada de delay de 30 frames


% Variação do tempo e da medida na caixa de texto "estático"


%obtencao do tempo(index) que mais se aproxima de a/taq = posicao do
slider

[c index] = min(abs(a/taq - x)); %
https://www.mathworks.com/matlabcentral/answers/152301-find-closest-
value-in-array

tempo = sprintf('%.3f',x(index)); % arredonda e transforma em string

t = ['Tempo: ', tempo];

num2str(y(index));

deg = sprintf('%.1f',y(index));

m = ['Ângulo: ',deg];

resp = {t;m};

set(handles.tempo,'string', resp);

guidata(hObject,handles);


% reta vertical : [a/taq a/taq],[-100000 100000], onde a/taq = tempo
atual

axes(handles.axes7);

plot(x, y,[a/taq a/taq],[-100000 100000]);

xlim([min(x) max(x)])

ylim([min(y)-0.05*max(y) max(y)*1.05]) % 5% somado em cada extremidade do
eiy para facilitar visualizacao de limites

grid on
```

```
xlabel('Tempo(s)')

ylabel('Ângulo (graus)')

title('Variação angular no tempo')


% --- Executes during object creation, after setting all properties.

function slider1_CreateFcn(hObject, eventdata, handles)

% hObject    handle to slider1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.

if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor',[.9 .9 .9]);

end


% --- Executes when figure1 is resized.

function figure1_SizeChangedFcn(hObject, eventdata, handles)

% hObject    handle to figure1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


% --- Executes when selected object is changed in graficos.

function graficos_SelectionChangedFcn(hObject, eventdata, handles)

% hObject    handle to the selected object in graficos

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


% imu = 1 = X

%     = 2 = Y

%     = 3 = Z

imu = get(handles.popmenuImu, 'Value') + 2; % soma dois para pular a
coluna tempo e a flexao


% importacao do caminho dos dados
```

```
pasta=get(handles.botaoAnalisar, 'UserData');


escolha = get(handles.graficos, 'SelectedObject');

escolhaStr = get(escolha, 'Tag');


switch escolhaStr


% Graficos de flexão

    case 'radiobutton1' % flexao braco

        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\braco.xls')));

        y = num(:,2);


    case 'radiobutton2' % flexao nuca

        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\master.xls')));

        y = num(:,2);


    case 'radiobutton3' % flexao perna

        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\perna.xls')));

        y = num(:,2);


% Graficos de movimento

    case 'radiobutton4' % movimento braco

        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\braco.xls')));

        y = num(:,imu);


    case 'radiobutton5' % movimento nuca

        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\master.xls')));

        y = num(:,imu);


    case 'radiobutton6' % movimento perna
```

```
        [num,txt,raw] =
xlsread(char(strcat(pasta,'\Gráficos\perna.xls')));

        y = num(:,imu);

end


x = cell2mat(raw(:,1))/1000;


a=round(get(handles.slider1,'Value'));

global taq


axes(handles.axes7);

plot(x, y,[a/taq a/taq],[-100000 100000]);

xlim([min(x) max(x)])

ylim([min(y)-0.05*max(y) max(y)*1.05])

grid on

xlabel('Tempo(s)')

ylabel('Ângulo (graus)')

title('Variação angular no tempo')


% armazena os eixos para poder usá-los a cada alteracao do slider

eixos = [x, y];

set(handles.graficos, 'UserData', eixos)


% --- Executes on selection change in popupmenu1.

function popupmenu1_Callback(hObject, eventdata, handles)

% hObject    handle to popupmenu1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array

%        contents{get(hObject,'Value')} returns selected item from
popupmenu1

[num,dados,row] = xlsread('Banco de Dados.xls'); % divide nas 3 partes

dados(1,:)=[]; % elimina a primeira linha que sao os headers

possiveis = get(handles.popupmenu1, 'String');
```

```
nome = possiveis((get(handles.popupmenu1, 'Value')));

set(handles.popupmenu2, 'Value', 1);

if(get(handles.popupmenu1, 'Value') ~= 1)

    % seleciona o subset de datas possiveis do nome selecionado

    linhas=find(strcmp(dados(:,2),nome));
%https://www.mathworks.com/matlabcentral/answers/84242-find-in-a-cell-
array

    dados_aux = dados(linhas,:); % subset de dados

    datas = dados_aux(:,3); % apenas terceira coluna (datas)

    lista = ['Data da análise'; datas];

    set(handles.popupmenu2, 'String', lista); % seta os valores a serem
apresentados no popup2

else

    set(handles.popupmenu2, 'String', 'Data da análise');

end


% --- Executes during object creation, after setting all properties.

function popupmenu1_CreateFcn(hObject, eventdata, handles)

% hObject    handle to popupmenu1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


% --- Executes on selection change in popupmenu2.

function popupmenu2_Callback(hObject, eventdata, handles)

% hObject    handle to popupmenu2 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
```

```
%        contents{get(hObject,'Value')} returns selected item from
popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenuSensor.
function popupmenuSensor_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenuSensor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenuSensor contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenuSensor
sensor = get(handles.popupmenuSensor, 'Value');
set(handles.popupmenuAngulo, 'Value', 1)
if (sensor == 2 | sensor == 3| sensor == 4)
    lista = {'Escolha o ângulo', '180', '135', '90', '45'};
    set(handles.popupmenuAngulo, 'String', lista)
else
    set(handles.popupmenuAngulo, 'String', 'Escolha o ângulo')
end


% --- Executes during object creation, after setting all properties.
function popupmenuSensor_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to popupmenuSensor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called


% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenuAngulo.
function popupmenuAngulo_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenuAngulo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = cellstr(get(hObject,'String')) returns
popupmenuAngulo contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenuAngulo


% --- Executes during object creation, after setting all properties.
function popupmenuAngulo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenuAngulo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called


% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in pushbuttonCalibrar.
function pushbuttonCalibrar_Callback(hObject, eventdata, handles)
% hObject    handle to pushbuttonCalibrar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
sensor = get(handles.popupmenuSensor, 'Value');
angulo = get(handles.popupmenuAngulo, 'Value');

% se nao selecionou nenhum sensor: Erro
if sensor == 1
        uiwait(msgbox({'Preencha os campos:',...
                    'Sensor',...
                    'e, caso opte por um dos flexores,',...
                    'Angulo'}));
% se nao selecionou nenhum angulo sendo que o sensor é um flexor: Erro
elseif ((sensor == 2 | sensor == 3 | sensor == 4)&(angulo == 1))
            uiwait(msgbox({'Preencha os campos:',...
                    'Sensor',...
                    'e, caso opte por um dos flexores,',...
                    'Angulo'}));
% Se nao tem Erro vai calibrar
else
    % BRACO
    if(sensor == 2 | sensor == 5)
        global braco
        s = braco
        fopen(s) % abre a troca de informações

        if sensor == 5 % IMU Braco
            fprintf(s,'c')
        else % Flexor Braco
            if angulo == 2 % 180
                fprintf(s,'q')
            elseif angulo == 3 % 135
```

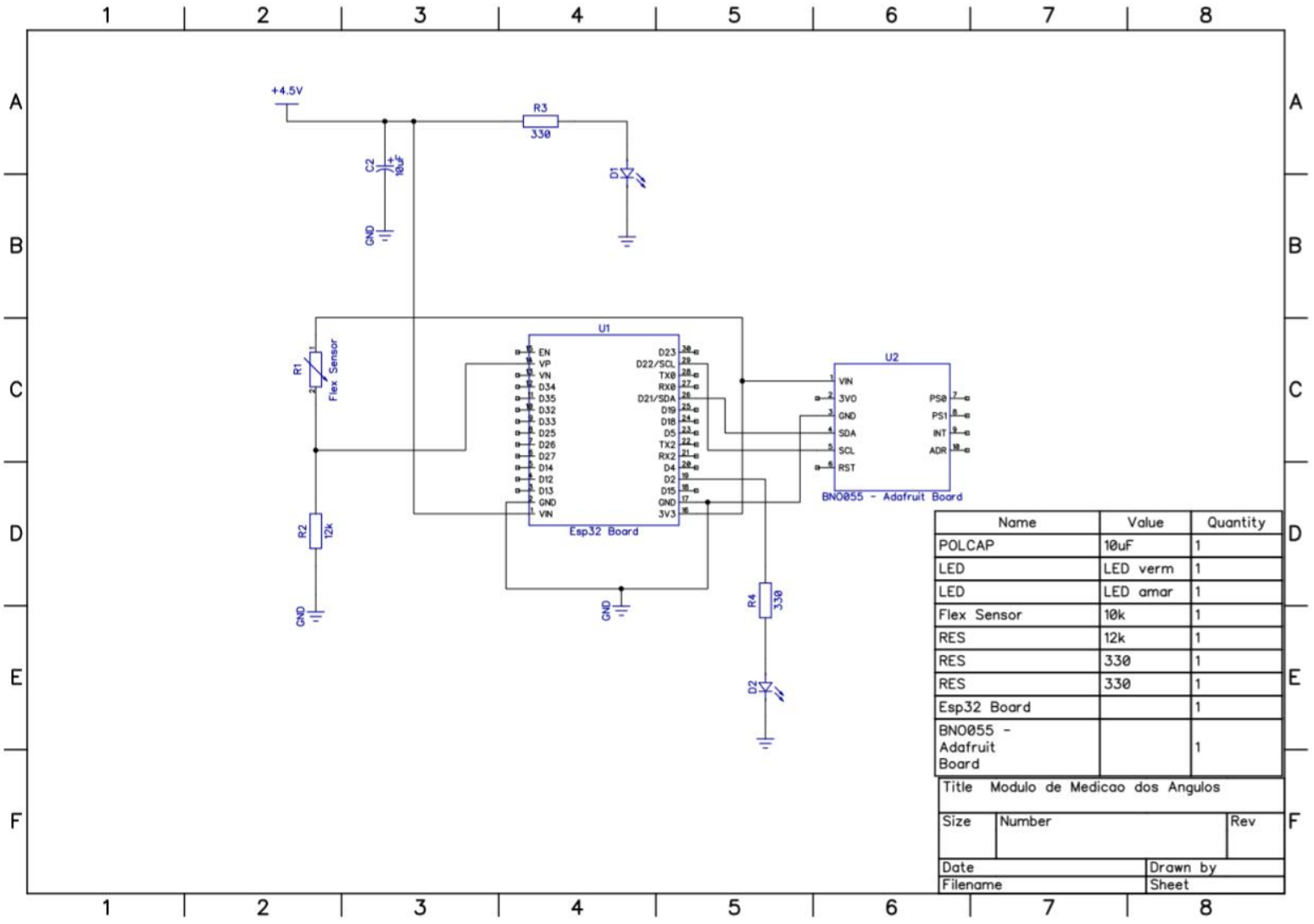
```
                fprintf(s,'w')
            elseif angulo == 4 % 90
                fprintf(s,'e')
            elseif angulo == 5 % 45
                fprintf(s,'r')
            end
        end
    % MASTER - NUCA
    elseif(sensor == 3 | sensor == 6)

        global master
        s = master
        fopen(s) % abre a troca de informações

        if sensor == 6 % IMU Master
            fprintf(s,'c')
        else % Flexor Nuca
            if angulo == 2 % 180
                fprintf(s,'q')
            elseif angulo == 3 % 135
                fprintf(s,'w')
            elseif angulo == 4 % 90
                fprintf(s,'e')
            elseif angulo == 5 % 45
                fprintf(s,'r')
            end
        end
    % PERNA
    elseif(sensor == 4 | sensor == 7)
        global perna
        s = perna
        fopen(s) % abre a troca de informações

        if sensor == 7 % IMU Perna
```

```
            fprintf(s,'c')
        else % Flexor Perno
            if angulo == 2 % 180
                fprintf(s,'q')
            elseif angulo == 3 % 135
                fprintf(s,'w')
            elseif angulo == 4 % 90
                fprintf(s,'e')
            elseif angulo == 5 % 45
                fprintf(s,'r')
            end
        end
    end
    fclose(s) % fecha o acesso a serial
end


% --- Executes on button press in botaoMedir.
function botaoMedir_Callback(hObject, eventdata, handles)
% hObject    handle to botaoMedir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Pegando o local do blueTooth
% btInfo = instrhwinfo('Bluetooth','ESP32Master') % propriedades da porta
em que esta conectado o ESP32Module1 via bluetooth
% channel = str2num(btInfo.Channels{1}) % declaração do canal da porta
como num
% master = Bluetooth('ESP32Master', channel)
global master
fopen(master) % abre a troca de informações
fprintf(master,'p') % dispara os sensores
fclose(master)


% --- Executes on selection change in popmenuImu.
function popmenuImu_Callback(hObject, eventdata, handles)
% hObject    handle to popmenuImu (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popmenuImu
contents as cell array

%        contents{get(hObject,'Value')} returns selected item from
popmenuImu


% --- Executes during object creation, after setting all properties.

function popmenuImu_CreateFcn(hObject, eventdata, handles)

% hObject    handle to popmenuImu (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called


% Hint: popupmenu controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


function editFps_Callback(hObject, eventdata, handles)

% hObject    handle to editFps (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editFps as text

%        str2double(get(hObject,'String')) returns contents of editFps as
a double

global fps

fps = str2num(get(handles.editFps, 'String'));


% --- Executes during object creation, after setting all properties.

function editFps_CreateFcn(hObject, eventdata, handles)

% hObject    handle to editFps (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called


% Hint: edit controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end


% --- Executes during object creation, after setting all properties.

function edit11_CreateFcn(hObject, eventdata, handles)

% hObject    handle to edit11 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called


% Executavel - https://www.youtube.com/watch?v=14FOQqCSGlM
```

10.4 Circuito do módulo de medição dos ângulos dos membros do atleta



Name	Value	Quantity
POLCAP	10uF	1
LED	LED verm	1
LED	LED amar	1
Flex Sensor	10k	1
RES	12k	1
RES	330	1
RES	330	1
Esp32 Board		1
BNO055 - Adafruit Board		1

Title	Modulo de Medicao dos Angulos	
Size	Number	Rev
Date		Drawn by
Filename		Sheet

10.5 Termos de autorização de uso de imagem

TERMO DE AUTORIZAÇÃO DE USO DE IMAGEM

Eu, IBIRATAN RADAMOVIC VIEIRA, portador da Cédula de Identidade nº 16.152.260-9, inscrito no CPF sob nº 115.296.568-93 residente à Rua DOM RAIMUNDO BRITO, nº 214, na cidade de SÃO PAULO, AUTORIZO o uso de minha imagem (ou do menor GABRIEL PIRES VIEIRA sob minha responsabilidade) em fotos ou filme decorrentes da participação no projeto da Escola Politécnica da Universidade de São Paulo, a seguir discriminado:

Programa: Trabalho de Conclusão de Curso da Engenharia Mecatrônica da Escola Politécnica da Universidade de São Paulo.

Pesquisadores: Alyson Akio Haro e Lucas Guardia Palopoli.

Professor Orientador: Rafael Traldi Moura.

Objetivos principais: Mapear a saída de um atleta de natação por meio de sensores e fornecer as respostas de forma amigável à profissionais do esporte.

As imagens e a voz poderão ser exibidas: nos relatórios parcial e final do referido projeto, na apresentação audiovisual do mesmo, em publicações e divulgações acadêmicas, em festivais e premiações nacionais e internacionais, assim como disponibilizadas no banco de imagens resultante da pesquisa e na Internet, fazendo-se constar os devidos créditos.

O aluno fica autorizado a executar a edição e montagem das fotos e filmagens, conduzindo as reproduções que entender necessárias, bem como a produzir os respectivos materiais de comunicação, respeitando sempre os fins aqui estipulados.

Por ser esta a expressão de minha vontade, nada terei a reclamar a título de direitos conexos a minha (ou do menor sob minha responsabilidade) imagem e voz ou qualquer outro.

São Paulo, 15 de OUTUBRO de 2018.

Assinatura

TERMO DE AUTORIZAÇÃO DE USO DE IMAGEM

Eu, Bruno Brambilla, portador da Cédula de Identidade nº 37483743-0, inscrito no CPF sob nº 457649044/62, residente à Rua Dr. Ferreira Lopes, nº 317, na cidade de São Paulo, AUTORIZO o uso de minha imagem (ou do menor _____ sob minha responsabilidade) em fotos ou filme decorrentes da participação no projeto da Escola Politécnica da Universidade de São Paulo, a seguir discriminado:

Programa: Trabalho de Conclusão de Curso da Engenharia Mecatrônica da Escola Politécnica da Universidade de São Paulo.

Pesquisadores: Alyson Akio Haro e Lucas Guardia Palopoli.

Professor Orientador: Rafael Traldi Moura.

Objetivos principais: Mapear a saída de um atleta de natação por meio de sensores e fornecer as respostas de forma amigável à profissionais do esporte.

As imagens e a voz poderão ser exibidas: nos relatórios parcial e final do referido projeto, na apresentação audiovisual do mesmo, em publicações e divulgações acadêmicas, em festivais e premiações nacionais e internacionais, assim como disponibilizadas no banco de imagens resultante da pesquisa e na Internet, fazendo-se constar os devidos créditos.

O aluno fica autorizado a executar a edição e montagem das fotos e filmagens, conduzindo as reproduções que entender necessárias, bem como a produzir os respectivos materiais de comunicação, respeitando sempre os fins aqui estipulados.

Por ser esta a expressão de minha vontade, nada terei a reclamar a título de direitos conexos a minha (ou do menor sob minha responsabilidade) imagem e voz ou qualquer outro.

São Paulo, 17 de Outubro de 2018.

Bruno Brambilla

Assinatura

TERMO DE AUTORIZAÇÃO DE USO DE IMAGEM

Eu, Erika Kazue Yanaba Inoue, portador da Cédula de Identidade nº 24.346.493-9, inscrito no CPF sob nº 153.183.438-80, residente à Rua Júpiter, 97 ap. 91, nº _____, na cidade de São Paulo, AUTORIZO o uso de minha imagem (ou do menor Vinícius Kenzo Inoue sob minha responsabilidade) em fotos ou filme decorrentes da participação no projeto da Escola Politécnica da Universidade de São Paulo, a seguir discriminado:

Programa: Trabalho de Conclusão de Curso da Engenharia Mecatrônica da Escola Politécnica da Universidade de São Paulo.

Pesquisadores: Alyson Akio Haro e Lucas Guardia Palopoli.

Professor Orientador: Rafael Traldi Moura.

Objetivos principais: Mapear a saída de um atleta de natação por meio de sensores e fornecer as respostas de forma amigável à profissionais do esporte.

As imagens e a voz poderão ser exibidas: nos relatórios parcial e final do referido projeto, na apresentação audiovisual do mesmo, em publicações e divulgações acadêmicas, em festivais e premiações nacionais e internacionais, assim como disponibilizadas no banco de imagens resultante da pesquisa e na Internet, fazendo-se constar os devidos créditos.

O aluno fica autorizado a executar a edição e montagem das fotos e filmagens, conduzindo as reproduções que entender necessárias, bem como a produzir os respectivos materiais de comunicação, respeitando sempre os fins aqui estipulados.

Por ser esta a expressão de minha vontade, nada terei a reclamar a título de direitos conexos a minha (ou do menor sob minha responsabilidade) imagem e voz ou qualquer outro.

São Paulo, 15 de outubro de 2018.

Erika K. Inoue

Assinatura