DISSERTAÇÃO DE MESTRADO

UM ESTUDO SOBRE ARQUITETURAS DE HARDWARE PARA TÉCNICAS DE FUSÃO SENSORIAL ATRAVÉS DO EKF E DA ESTIMAÇÃO DE ESTADOS BASEADA EM FILTROS HÍBRIDOS OTIMIZADOS

FABIÁN BARRERA PRIETO

Brasília, Abril de 2018

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

UM ESTUDO SOBRE ARQUITETURAS DE HARDWARE PARA TÉCNICAS DE FUSÃO SENSORIAL ATRAVÉS DO EKF E DA ESTIMAÇÃO DE ESTADOS BASEADA EM FILTROS HÍBRIDOS OTIMIZADOS

FABIÁN BARRERA PRIETO

Orientador: Prof. Dr. Carlos Humberto Llanos Quintero, PPMEC/UnB
Coorientador: Prof. Dr. Daniel Maurício Muñoz Arboleda,
FGA/UnB,PPMEC/UnB

DISSERTAÇÃO DE MESTRADO EM SISTEMAS MECATRÔNICOS

BRASÍLIA/DF, 27 ABRIL DE 2018

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

UM ESTUDO SOBRE ARQUITETURAS DE HARDWARE PARA TÉCNICAS DE FUSÃO SENSORIAL ATRAVÉS DO EKF E DA ESTIMAÇÃO DE ESTADOS BASEADA EM FILTROS HÍBRIDOS OTIMIZADOS

FABIÁN BARRERA PRIETO

DISSERTAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA MECÂNICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM SISTEMAS MECATRÔNICOS

APROVADA POR:

Prof. Dr. Carlos Humberto Llanos Quintero, PPMEC/UnB
Orientador

Prof. Dr. Ricardo Pezzuol Jacobi, CIC/UnB
Membro Interno

Prof. Dr. Helon Vicente Hultmann Ayala, ENM/PUC-RJ
Membro Externo

BRASÍLIA/DF, 27 ABRIL DE 2018

Barrera Prieto, Fabián
Um estudo sobre arquiteturas de hardware para técnicas de fusão sensorial através do EKF e da estimação de estados baseada em filtros híbridos otimizados / FABIÁN BARRERA PRIETO. –Brasil, 2018.
92 p.

Orientador: Carlos Humberto Llanos Quintero

Dissertação (Mestrado) – Universidade de Brasília – UnB
Faculdade de Tecnologia – FT
Programa de Pós-Graduação em Sistemas Mecatrônicos – PPMEC, 2018.

1. Unidade de Medição Inercial. 2. *Attitude* e *heading* 3. Fusão sensorial. 4. FPGA. 5. Filtros híbridos otimizados. I. Carlos H. Llanos, orientador. II. Universidade de Brasília. III. Faculdade de Tecnologia.

Dedicatória

Este trabalho é dedicado a Deus e a aquelas pessoas que sempre acreditaram em mim. Tanto as que hoje me acompanham em vida, especialmente aos meus pais (Wilson e Maria) e ao meu irmão (Wilson Jr.); quanto as que já estão descansando em paz, à minha avó Carmen, à minha madrinha Rosa e à Sra. Adela (QEPD).

FABIÁN BARRERA PRIETO

“La vida es como un rompecabezas, en la que hay que saber acomodar cada pieza en su lugar. A veces cometemos diversos errores (fracasos) posicionando las piezas donde no deben ir. En ocasiones hace falta quien te mueva las piezas de sitio, porque solo así logras comprender que cada movimiento (decisión) y cada pieza (persona) tienen su razón de ser. Sin embargo, sólo al completar el acertijo (sueño) conseguirás realmente entender que todos tus movimientos (vivencias e experiencias) eran necesarios e indispensables en tu rompecabezas (vida).”

FABIÁN BARRERA PRIETO

Agradecimentos

No primeiro lugar, a Deus por ter-me permitido terminar felizmente mais uma etapa na minha vida e por todas as bênçãos durante os dois anos do Mestrado.

Aos meus pais Wilson e Maria, e ao meu irmão Wilson Jr. por serem os que sempre me motivaram para continuar neste desafio, além disso pelos conselhos, ensinamentos e por toda a força que me deram; e a toda a minha família pela compreensão e pelo apoio.

Ao meu primo e padrinho Alejandro, assim como à sua esposa Naty e ao seu filho Isaac, pela amizade e ajuda que me brindaram desde o primeiro dia que cheguei ao Brasil.

Ao meu orientador, o professor Carlos pela paciência, ajuda e orientação no desenvolvimento do meu projeto, assim como pelos conselhos dados durante o Mestrado.

Ao meu coorientador Daniel Muñoz e ao Renato pela paciência, assim como pelas explicações e tutorias durante o processo de desenvolvimento do meu projeto de Mestrado.

Aos meus colegas do LEIA, Reurison, Carlos, Oscar, Sergio Pertuz, Sergio Cruz, João Carlos e Jessé pelas ideias compartilhadas e pela ajuda e amizade que me brindaram.

À FAP-DF (Fundação de Apoio à Pesquisa do Distrito Federal) pelo apoio financeiro.

FABIÁN BARRERA PRIETO

RESUMO

Este trabalho apresenta um estudo e uma implementação em *hardware* de uma técnica de fusão sensorial de Unidades de Medição Inercial (IMUs) de 9 graus de liberdade (DOF) baseada no Filtro de Kalman Estendido (EKF) como mecanismo de fusão; com o objetivo de estimar a posição (*attitude*) e a orientação (*heading*) em tempo real. A IMU usada está composta por sensores (giroscópio, acelerômetro e magnetômetro) nos três eixos ortogonais (X, Y e Z). A técnica de fusão de sensores foi baseada em dois Filtros de Kalman Estendidos (EKFs), um para estimar a posição (rolagem e arfagem) e o outro para estimar a orientação (guinada). Portanto, o giroscópio foi usado como modelo do sistema nos dois casos, enquanto o acelerômetro e o magnetômetro foram usados independentemente como modelos de medição. Esses sensores foram calibrados previamente à filtragem (processamento). O estudo dessa abordagem de fusão sensorial foi realizado através de uma bancada (plataforma) fabricada em impressão 3D, afim de facilitar os movimentos durante a calibração e os testes das IMUs. Nesse estudo, as leituras dos sensores e o processamento dos dados foram feitos através do Arduino e do Matlab R2015a, respectivamente. A arquitetura foi embarcada em FPGA (*Field Programmable Gate Array*) e desenvolvida usando a linguagem de descrição de *hardware* VHDL (*Very High Description Language*). A arquitetura opera a uma frequência de 50MHz e faz uso de operadores em ponto flutuante de 27 bits. Adicionalmente, a comunicação entre a IMU e o FPGA foi realizada através do protocolo I2C implementado em *software*, através do processador Nios II. Portanto, a leitura das medições fornecidas pelos sensores foi feita pelo Nios II, enquanto o processamento de dados foi realizado na arquitetura embarcada no FPGA. Com respeito ao estudo do desempenho de diferentes filtros híbridos otimizados para problemas de estimação de estados, tiveram-se em conta quatro tipos de *resamplings* diferentes (*multinomial*, *systematic*, *stratified* e *residual*). Isto foi feito com o intuito de avaliar diferentes abordagens que poderiam ser utilizadas em aplicações onde seja fundamental ter uma boa qualidade de estimação das variáveis de estado (e.g. rede multi-sensor em um exoesqueleto). Essas abordagens combinam o Filtro de Partículas (PF), o Filtro de Kalman (KF) e o algoritmo de Otimização por Enxame de Partículas (PSO). Onde o KF e o PSO são implementados na etapa de amostragem e após do *resampling* do PF, respectivamente. Além disso, dois métodos de adição de diversidade foram utilizados (Aprendizagem Baseada em Oposição (OBL) e Atrativo e Repulsivo (AR)). Com o intuito de avaliar essas abordagens híbridas foram levados em consideração dois sistemas não lineares (*benchmarks*). Finalmente, os resultados em *hardware* apresentados neste trabalho mostraram um ganho significativo na velocidade de processamento da fusão sensorial, pois o processamento na arquitetura é 6023 vezes mais rápido do que o mesmo algoritmo implementado no Nios II (operando a 100 MHz). Com respeito aos resultados dos filtros híbridos otimizados, os que atingiram menor erro (RMSE) na estimação de estados foram o UKF PF OPSO para o *benchmark* 1 e o UKF PF PSO para o *benchmark* 2.

ABSTRACT

This work presents a study and hardware implementation of a sensor fusion technique of 9 Degrees Of Freedom (DOF) Inertial Measurement Units (IMUs) based on the Extended Kalman Filter (EKF) as a fusion mechanism; with the objective of estimating the position (attitude) and orientation (heading) in real time. The IMU used is composed of sensors (gyroscope, accelerometer and magnetometer) in the three orthogonal axes (X, Y and Z). The sensor fusion technique was based on two Extended Kalman Filters (EKFs), one for attitude estimation (roll and pitch) and the other for estimating the heading (yaw). Therefore, the gyroscope was used as the model of the system in both cases, while the accelerometer and the magnetometer were used independently as measurement models. These sensors were calibrated prior to filtration (processing). The study of this approach of sensorial fusion was carried out through a bench (platform) manufactured in 3D printing to facilitate movement during calibration and tests of IMUs. In this study, sensor readings and data processing were done through Arduino and Matlab R2015a, respectively. The architecture was embedded in Field Programmable Gate Array (FPGA) and developed using the hardware description language VHDL (Very High Description Language). The architecture operates at a frequency of 50MHz and makes use of 27-bits floating-point operators. Additionally, the communication between the IMU and the FPGA was performed through the I2C protocol implemented in software, through the processor Nios II. Therefore, the readings of the measurements provided by the sensors were made by Nios II, while the data processing was performed in the FPGA-embedded architecture. Regarding the study of the performance of different optimized hybrid filters for state estimation problems, four different types of resamplings (multinomial, systematic, stratified and residual) were considered. This was done with the aim of evaluating different approaches that could be used in applications where it is essential to have a good quality of estimation of state variables (eg multi-sensor network in an exoskeleton). These approaches combine the Particle Filter (PF), the Kalman Filter (KF), and the Particle Swarm Optimization (PSO) algorithm. Where the KF and the PSO were implemented in the sampling step and after the resampling of the PF, respectively. Furthermore, two methods of adding diversity were used (Opposition Based Learning (OBL) and Attractive and Repulsive (AR)). In order to evaluate these hybrid approaches two nonlinear systems (benchmarks) were considered. Finally, the hardware results presented in this work showed a significant gain in the processing speed of the sensor fusion, since the processing in the architecture is 6023 times faster than the same algorithm implemented in Nios II (operating at 100 MHz). With respect to the results of the optimized hybrid filters, the ones that reached the lowest error (RMSE) in the state estimation were the UKF PF OPSO for the benchmark 1 and the UKF PF PSO for the benchmark 2.

SUMÁRIO

RESUMO ..... **i**

ABSTRACT ..... **ii**

LISTA DE FIGURAS ..... **v**

LISTA DE TABELAS ..... **viii**

LISTA DE ALGORITMOS ..... **ix**

LISTA DE ABREVIATURAS E ACROGRAMAS ..... **xi**

1 Introdução **1**
1.1 Contextualização 1
1.2 Definição do problema e motivações 2
1.3 Objetivos da dissertação 4
1.3.1 Objetivo geral 4
1.3.2 Objetivos específicos 4
1.4 Contribuições 5
1.5 Organização do documento 5

2 Estudo da fusão sensorial para as IMUs com o EKF **7**
2.1 Orientação e posição 7
2.2 Unidades de Medicão Inercial 9
2.2.1 Comunicação I2C 11
2.2.2 Giroscópio 13
2.2.3 Acelerômetro 14
2.2.4 Magnetômetro 16
2.3 Fusão sensorial 17
2.4 Filtros para fusão sensorial de IMUs 18
2.4.1 Filtro complementar 18
2.4.2 Filtro de Kalman 19
2.5 Bancada para calibrações e testes 22
2.6 Métricas de avaliação 23
2.7 Trabalhos correlatos sobre técnicas de fusão sensorial IMUs 24

3 Plataformas para implementar a fusão sensorial usando IMUs 28
3.1 Arquiteturas programáveis via *software* 29
3.1.1 Processador de Propósito Geral (GPP) 29
3.1.2 Processador digital de sinal (DSP) 29
3.1.3 Unidade de processamento gráfico (GPU) 30
3.2 Circuito integrado de aplicação específica (ASIC) 31
3.3 Arquiteturas reconfiguráveis 31
3.3.1 FPGA 32
3.4 Trabalhos correlatos em implementação de algoritmos de fusão sensorial de IMUs em FPGAs 34

4 Arquitetura em FPGA para a fusão sensorial de IMUs 38
4.1 Arquitetura geral de fusão sensorial 38
4.2 Módulos para operações em ponto flutuante 40
4.2.1 *Adder/Subtractor* 41
4.2.2 *Multiplier* 41
4.2.3 *Divider* 42
4.2.4 *sqrt* (raiz quadrada) 42
4.2.5 *Cordic1 e Cordic2* (*seno*/*cosseno* e *atan*) 42
4.2.6 *atan2* 43
4.3 Módulos de fusão sensorial 44
4.3.1 *Master* 44
4.3.2 *PreCordic* 46
4.3.3 *EKF1* 46
4.3.4 *EKF2* 51
4.3.5 *Accelerometer* 54
4.3.6 *Magnetometer* 56
4.3.7 *Conversion* 57

5 Estimadores Não Lineares 58
5.1 Filtro de partículas 58
5.1.1 *Sequential Importance Sampling* (SIS) 59
5.1.2 *Sequential Importance Resampling* (SIR) 60
5.2 Filtros híbridos 66
5.2.1 Filtro de Partículas Estendido 67
5.2.2 Filtro de Partículas Unscented 68
5.3 Algoritmos bioinspirados 68
5.3.1 Otimização por enxame de partículas (PSO) 69
5.3.2 Métodos de adição de diversidade 70
5.4 Filtros híbridos otimizados 71
5.4.1 Filtro de Partículas com PSO 71
5.4.2 Filtro de Partículas Estendido com PSO 72
5.4.3 Filtro de Partículas Unscented com PSO 73

5.5 Trabalhos correlatos sobre filtros híbridos baseados no PF 73

6 Resultados e análise 77
6.1 Calibração dos sensores da IMU 77
6.2 Técnica de fusão sensorial 81
6.3 Arquitetura embarcada em FPGA 84
6.4 Filtros híbridos otimizados 87

7 Conclusões e trabalhos futuros 97
7.1 Conclusões 97
7.2 Propostas de trabalhos futuros 98

REFERÊNCIAS BIBLIOGRÁFICAS 100

ANEXOS 106

LISTA DE FIGURAS

1.1 Centros de gravidade das extremidades inferiores (adaptado de (VAUGHAN et al., 1999))... 3
1.2 Abordagem mapeada em FPGA para uma rede multi-sensor baseada em IMUs. A transmissão de dados no barramento *Avalon* é de 32 bits. 4

2.1 Representação dos ângulos de Euler (adaptado de (STEPHEN, 2014)). 9
2.2 Eixos ortogonais e ângulos de referência das IMUs de 6 e 9 DOF (adaptado de (INVENSENSE, 2013) e (INVENSENSE, 2016)). 10
2.3 MPU6050 (6 DOF) e MPU9250 (9 DOF) (adaptado de (ADDICORE, 2017)). 11
2.4 Diagrama de blocos da MPU9250 (adaptado de (INVENSENSE, 2016)). 12
2.5 Sincronização da comunicação I2C (adaptado de (SEMICONDUCTORS, 2000)). 12
2.6 Processo de escrita e leitura na comunicação I2C (adaptado de (CHANDRASEKARAN, 2013)). 13
2.7 Aplicações do giroscópio (adaptado de (KRAFT, 2000)). 13
2.8 Aplicações do acelerômetro (adaptado de (KRAFT, 2000)). 15
2.9 Estrutura da fusão sensorial. 17
2.10 Funcionamento do filtro Complementar para o caso do giroscópio e acelerômetro. 18
2.11 Inicialização, predição e correção do filtro de Kalman. Modelo de transição de estados (A), modelo de observação (H), covariância do ruído do processo (Q), covariância do ruído da medição (R), variável de medida (z_k). 19
2.12 Transformação *Unscented* para a propagação da média e da covariância. (a) Estado atual (método do Monte Carlo), (b) estado linearizado (EKF), (c) transformação *Unscented* (UKF) (adaptado de (WAN; MERWE, 2000)). 21
2.13 Bancada para calibração e testes com IMUs. As medidas correspondentes são 15x15x12.4cm 22

3.1 Comparação qualitativa entre a GPU e o FPGA (adaptado de (BERTEN, 2016)). 29
3.2 Processador de propósito geral (adaptado de (VAHID; GIVARGIS, 2002)). 30
3.3 Comparação entre a CPU e a GPU (adaptado de (DUTCH, 2017). 31
3.4 Classificação dos ASICs (adaptado de (KAESLIN, 2008)). 32
3.5 Arquitetura básica do FPGA (adaptado de (FAROOQ; MARRAKCHI; MEHREZ, 2012)).. 33
3.6 Fluxo do projeto no FPGA (adaptado de (PEDRONI, 2004)). 33
3.7 Diagrama de blocos do Kit DE2-115 (adaptado de (ALTERA, 2010)). 34

4.1 Diagrama do projeto em FPGA. A transmissão de dados no barramento *Avalon* é de 32 bits. 39

4.2 Arquitetura de fusão sensorial de uma IMU de 9 DOF com os EKFs. Está composta por (a) interface de dados (*input/output*), (b) módulos de fusão sensorial, (c) multiplexadores (MUX1 e MUX2), (d) bibliotecas em ponto flutuante de 27 bits (*cordic1*, *multipliers*, *adders*, *subtractors*, *dividers*, SQRT, *cordic2*)..... 40
4.3 Scheduling da arquitetura. Para dois casos: (a) primeira rodada ($k = 1$) e (b) seguintes rodadas ($k = 2, 3, ..., n$). * *Accelerometer*. ** *Magnetometer*. 41
4.4 Representação em ponto flutuante (adaptado de (RAFAEL, 2014)). 41
4.5 Operadores e FSM do módulo *atan2* 43
4.6 FSM do módulo *master*. 45
4.7 FSM do módulo *preCordic*. 46
4.8 Módulo *EKF1*. 47
4.9 Operadores e FSM do *módulo A* do *EKF1*. 47
4.10 Operadores e FSM do *módulo B* do *EKF1*. 48
4.11 Operadores e FSM do *módulo 1* do *EKF1*..... 48
4.12 Operadores e FSM do *módulo 2* do *EKF1*..... 49
4.13 Operadores e FSM do *módulo 3* do *EKF1*..... 49
4.14 Operadores e FSM do *módulo 4* do *EKF1*..... 50
4.15 Operadores e FSM do *módulo 5* do *EKF1*..... 50
4.16 Operadores e FSM do *módulo 6* do *EKF1*..... 51
4.17 Módulo *EKF2*. 52
4.18 Operadores e FSM do *módulo 1* do *EKF2*..... 52
4.19 Operadores e FSM do *módulo 2* do *EKF2*..... 53
4.20 Operadores e FSM do *módulo 3* do *EKF2*..... 53
4.21 Operadores e FSM do *módulo 4* do *EKF2*..... 53
4.22 Operadores e FSM do *módulo 5* do *EKF2*..... 54
4.23 Operadores e FSM do *módulo A* do *EKF2*. 54
4.24 Módulo *accelerometer*. 55
4.25 FSM do módulo *master accel*. S0 (*waiting*), S1 (*roll*) e S2 (*pitch*) 55
4.26 Operadores e FSM do módulo *Pitch*. 55
4.27 Módulo *magnetometer*. 56
4.28 Operadores e FSM do módulo *Bfy*. 56
4.29 Operadores e FSM do módulo *Bfx* 57
4.30 Operadores e FSM do módulo *Conversion*..... 57

5.1 Comparação dos filtros para estimação de estados (adaptado de (SIMON, 2006)). 58
5.2 Saída de estimação do filtro de partículas (adaptado de (LLANOS, 2017))..... 59
5.3 Lógica do *resampling* no filtro de partículas (adaptado de (SIDIBÉ, 2011)). 61
5.4 Esquema do resampling multinomial (adaptado de (BLANCO, 2017))..... 62
5.5 Esquema do resampling systematic (adaptado de (BLANCO, 2017)). 63
5.6 Esquema do resampling stratified (adaptado de (BLANCO, 2017)). 64
5.7 Esquema do resampling residual (adaptado de (BLANCO, 2017)). 66
5.8 Combinação do filtro de partículas com o filtro de Kalman..... 67

5.9 Aprendizagem Baseada em Oposição (adaptado de (MUÑOZ, 2016)). 70
5.10 Atração e repulsão (adaptado de (MUÑOZ, 2016)). 71

6.1 Conexões entre a IMU, o Arduino e o Matlab 77
6.2 Plataforma com o Arduino 78
6.3 Resultados de antes e após da calibração do acelerômetro da MPU9250 79
6.4 Resultados de antes e após da calibração do giroscópio da MPU9250 80
6.5 Resultados de antes e após da calibração do magnetômetro da MPU9250 81
6.6 Fusão sensorial para as IMUs 82
6.7 Estimativa dos ângulos de Euler da MPU6050 83
6.8 Estimativa dos ângulos de Euler da MPU9250 84
6.9 Conexões entre a IMU, o FPGA e o Eclipse 84
6.10 Plataforma com o FPGA 85
6.11 Erros da arquitetura no FPGA para os ângulos de rolagem, arfagem e guinada 86
6.12 Funcionamento dos filtros híbridos otimizados 89
6.13 Resultados da estimação de estado dos filtros híbridos otimizados para os benhcmarks 1 e 2 90

LISTA DE TABELAS

2.1 Comparação entre os métodos para determinar a rotação 3D de um sistema 8
2.2 Comparação das IMUs de baixo custo. 9
2.3 Principais características da MPU6050 e MPU9250..... 11
2.4 Características del Giroscópio da MPU6050 e MPU9250..... 14
2.5 Características do acelerômetro da MPU6050 e MPU9250..... 15
2.6 Características do magnetômetro da MPU9250..... 16
2.7 Quantidades das peças da bancada 22
2.8 Medidas estatísticas e métricas de avaliação de desempenho..... 23
2.9 Revisão bibliográfica dos trabalhos correlatos de fusão sensorial de IMUs para determinar a posição e orientação 25

3.1 Revisão bibliográfica dos trabalhos correlatos para determinar *attitude* e *heading* através da fusão sensorial de IMUs embarcados em FPGAs..... 35

4.1 Condições para a transições de estados do módulo *atan2*. * Sem condição 44
4.2 Lógica da FSM do módulo *master*. S0 (*Waiting*), S1 (*preCordic*), S2 (*accelerometer*), S3 (*cordic1*(*sin/cos*)), S4 (*EKF1*), S5 (*sqrt*), S6 (*cordic2*(*atan*)), S7 (*magnetometer*), S8 (*EKF2*), S9 (*conversion*). 45

5.1 Revisão bibliográfica dos trabalhos correlatos dos filtros híbridos para estimação de estados. 74

6.1 Parâmetros utilizados nos dois módulos baseados no EKF. No EKF1 os parâmetros estão definidos para duas variáveis de estado em uma matriz 2x2, enquanto o EKF2 trabalha com escalares para só uma variável de estado. 83
6.2 Erros NRMSE para a MPU6050 e MPU9250 84
6.3 Erros NRMSE dos ângulos de Euler (arquitetura FPGA e Matlab) 85
6.4 Recursos utilizados para a arquitetura embarcada no FPGA..... 87
6.5 Resultados de processamento para fusão sensorial com os EKFs 87
6.6 Resultados da média, mediana e desvio padrão dos 32 erros RMSE de cada filtro não linear para o *benchmark* 1 92
6.7 Resultados da média, mediana e desvio padrão dos 32 erros RMSE de cada filtro não linear para o *benchmark* 2 93
6.8 Ranking dos filtros não lineares para o *benchmark* 1 e 2 96

LISTA DE ALGORITMOS

1 EKF 20
2 UKF 21
3 Filtro de partículas SIS 60
4 Filtro de partículas SIR 61
5 Resampling multinomial 62
6 Resampling systematic 64
7 Resampling stratified 65
8 Resampling residual 66
9 Filtro de partículas estendido 67
10 Filtro de partículas *unscented* 68
11 PSO 69
12 Filtro de partículas com PSO 72
13 Filtro de partículas estendido com PSO 72
14 Filtro de partículas unscented com PSO 73

LISTA DE ABREVIATURAS E ACROGRAMAS

ABNT	Associação Brasileira de Normas Técnicas
ACC	Acelerômetro
ADC	Conversor Análogo Digital
AR	Atrativo e Repulsivo
ASIC	Circuito Integrado de Aplicação Específica
BLE	Elemento Lógico Básico
CF	Filtro Complementar
CLB	Blocos Lógicos Configuráveis
COBEM	ABCM International Congress of Mechanical Engineering
CORDIC	Coordinate Rotation Digital Computer
CPLD	Dispositivos Lógicos Programáveis Complexos
CPU	Unidade Central de Processamento
DCM	Matriz de Cossenos Diretores
DLT	Transformação Linear Direta
DMP	Processador de Movimento Digital
DOF	Graus de Liberdade
DSP	Processadores Digitais de Sinal
EKF	Filtro de Kalman Estendido
EPF	Filtro de Partículas Estendido
EEPROM	Electrically-Erasable Programmable Read-Only Memory
FAP-DF	Fundação de Apoio à Pesquisa do Distrito Federal
FF	Flip Flops

FIFO	First In, First Out
FPGA	Field Programmable Gate Arrays
FSM	Máquina de Estados Finita
GPP	Processador de Propósito Geral
GPU	Unidade de Processamento Gráfico
GRACO	Grupo de Automação e Controle
HLS	Sínteses de Alto Nível
I2C	Inter-Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
IMU	Unidade de Medição Inercial
INS	Sistema de Navegação Inercial
ISO	Internacional Organization for Standarization
KF	Filtro de Kalman
LEIA	Laboratory of Embedded Systems and Integrated Circuits Applications
LSB	Less Significant Bit
LUT	Lookup Table
MAC	multiplicação e acumulação
MATLAB	Matrix Laboratory
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MEMS	Sistemas Microelectromecânicos
MSB	Most Significant Bit
OBL	Aprendizagem Baseada em Oposição
PDF	Função de densidade de probabilidade
PLD	Dispositivo Lógico Programável
PLL	Phase Locked Loop
PSO	Otimização por Enxame de Partículas
RAM	Memória de acesso aleatório
RPF	Filtro de Partículas Regularizado
RTL	Register transfer level
SCL	Serial Clock

SDA	Serial Data
SoC	System on Chip
SPLD	Dispositivo Lógico Programável Simples
SIS	Sequential Importance Sampling
SIR	Sequential Importance Resampling
SMC	Sequential Monte Carlo
UAV	Veículo aéreo não tripulado
UKF	Filtro de Kalman Unscented
ULA	Unidade Lógica e Aritmética
UNB	Universidade de Brasília
UNGM	Modelo de crescimento não estacionário univariado
UPF	Filtro de Partículas Unscented
USB	Universal Serial Bus
UT	Transformação *Unscented*
VANT	Veículo Aéreo Não Tripulado
VHDL	Hardware Description Language

Capítulo 1

Introdução

Neste capítulo apresenta-se a contextualização e a definição do problema, assim como a principal motivação desta dissertação. Adicionalmente são apresentados os objetivos, assim como as contribuições do desenvolvimento do projeto. Finalmente, se faz uma breve descrição sobre a organização deste documento.

1.1 Contextualização

Atualmente, os sistemas de navegação inercial têm diversas aplicações industriais desde a orientação dos aviões, navios e veículos terrestres até na área de robótica (WESTON; TITTERTON, 2000). Portanto, nos últimos anos criou-se uma nova área de estudo baseada na combinação da medicina com a robótica, a qual é conhecida atualmente como engenharia biomédica. Nessa área têm-se desenvolvido projetos complexos, sendo um desses os exoesqueletos. Esses equipamentos são basicamente estruturas externas ao corpo, tendo uma função principal, já seja ajudando na recuperação da mobilidade das extremidades inferiores das pessoas (uso médico) ou aumentando a resistência das pessoas (uso militar). Portanto, esses robôs vestíveis precisam ser instrumentados a fim de se obter informações confiáveis e precisas, em tempo real, referentes às posições e orientações dos membros dos mesmos. Nesse contexto, um requisito importante na instrumentação de um exoesqueleto é a incorporação de diferentes sensores (acelerômetro, giroscópio, magnetômetro, sensor de forca, etc) na estrutura do robô, a fim de estimar os movimentos do exoesqueleto. Em alguns casos são utilizados alguns componentes externos que ajudam na estimativa dos movimentos das pessoas, tal como o uso de uma bengala instrumentada com sensores, a qual pode ser utilizada para identificação e estimações dos movimentos do usuário no momento da caminhada (iniciar, parar e caminhar continuamente) (HASSAN et al., 2014).

A maioria dos dispositivos eletrónicos de medida usados em exoesqueletos são baseados na tecnologia de Sistemas Microelectromecânicos (*MicroElectroMechanical Systems* - MEMS), os quais variam entre $1\mu m$ (um milionésimo de metro) a $1mm$ (NAWRAT et al., 2012). No ano 2016, os principais fabricantes de sensores MEMS foram (1) Robert Bosch (US$ 1160 milhões), (2) Broadcom (US$ 910 milhões), (3) Texas Instruments (US$ 787 milhões), (4) STMicroelectronics (US$ 630 milhões) e (5) Qorvo (US$ 585 milhões) (YOLE, 2017). Tais sensores são utilizados frequentemente para estimar a posição e orientação nos sistemas de navegação, junto com técnicas modernas baseadas em fusão sensorial.

No caso de sistemas dinâmicos com comportamento não linear, uma escolha importante de projeto é a seleção e ajuste dos estimadores de dados. Entre os estimadores mais usados em robótica podem-se destacar: (a) Filtro de Kalman Estendido (*Extended Kalman Filter* - EKF); (b) Filtro de Kalman Unscented (*Unscented Kalman Filter* - UKF) e (c) Filtro de Partículas (*Particle Filter* - PF). O EKF é o filtro sobre o qual têm-se desenvolvido mais pesquisas nos últimos anos, destacando-se principalmente o fato de ser de fácil implementação (BAVDEKAR; DESHPANDE; PATWARDHAN, 2011). Adicionalmente, há abordagens de combinação de filtros não lineares com algoritmos de otimização, com o objetivo principal de melhorar a qualidade da estimação de estados (CHENG, 2012). Porém, esses *filtros híbridos otimizados* geram um alto custo computacional.

Levando em consideração as arquiteturas utilizadas para implementar estimadores não lineares, se destacam as arquiteturas de *hardware* devido a suas vantagens em termos de desempenho e eficiência. Além disso, esse tipo de arquiteturas tem a característica principal que permitem paralelizar algoritmos que envolvem funções complexas (MOSQUERA, 2016). Portanto, qualquer dos filtros mencionados anteriormente (EKF, UKF e PF) podem ser relativamente de fácil implementação em uma arquitetura de *hardware*.

No contexto do projeto de sistemas de medição envolvendo Unidades de Medição Inercial (*Inertial Measurement Units* - IMUs), existe a carência de uma abordagem integrada que inclua arquiteturas de *hardware*, direcionadas para métodos apropriados de implementação de fusão de sensores; especialmente para IMUs de 9 DOF de baixo custo. Isto tem-se tornado em um desafio, dado que o processamento deve ser feito a altas velocidades e a sua vez a solução deve ser facilmente embarcada em um único chip de silício, tendo em conta que um dos requisitos deste tipo de sistemas consiste em realizar a fusão sensorial (processamento dos dados) em tempo real.

1.2 Definição do problema e motivações

O alto custo computacional está relacionado com o tempo de processamento e a quantidade de recursos utilizados, se constituindo como os principais problemas que se apresentam no momento de realizar a estimação da '*pose*' (posição e orientação) de um sistema de navegação. Neste sentido, os algoritmos utilizados (como mecanismos de fusão sensorial) são altamente complexos. Diferentes abordagens de fusão sensorial (técnicas e plataformas) têm sido utilizadas a fim de estimar a posição e a orientação em tempo real e com a menor quantidade de recursos possíveis. Outro problema é a estimação de estados com alta incerteza (GALLAS, 1998); processo que depende principalmente do desempenho dos algoritmos utilizados para dita estimação, sendo afetado pelo nível da não linearidade do sistema, assim como do filtro utilizado. Portanto, diferentes abordagens de filtros híbridos otimizados têm sido propostas a fim de estimar as variáveis de estado com menores incertezas (ZHANG et al., 2008).

Levando em consideração o caso da fusão sensorial com Unidades de Medição Inercial (IMUs), aplicada em sistemas altamente não lineares, destaca-se o Filtro de Kalman Estendido (EKF) como o mecanismo de fusão sensorial, pois este estimador apresenta baixo esforço computacional se comparado com outros estimadores (SIMON, 2006). Adicionalmente, este filtro é muito utilizado para diferentes aplicações focadas à estimação estados, devido ao seu bom desempenho e a sua fácil implementação (BAVDEKAR; DESHPANDE; PATWARDHAN, 2011). Nesse contexto, as arquiteturas reconfiguráveis são uma solução

ótima em termos de desempenho para esse tipo de aplicações, já que permitem a aceleração dos algoritmos através da paralelização dos mesmos. Adicionalmente, outras vantagens significativas do uso de FPGAs (*Field Programmable Gates Arrays*) são: (a) menor consumo de potência se comparado com uma solução baseada em *desktop* (*software*), (b) miniaturização da solução, (c) possibilidade de atingir um processamento em tempo real e (d) em alguns casos, melhoria da flexibilidade quando se usa um SoC. Isto é, aliando a flexibilidade do *software* com a robustez do *hardware*, sendo essa uma boa solução para aplicações embarcadas com restrições de portabilidade (tamanho e peso). Portanto, resulta importante desenvolver uma solução embarcada para esse caso, onde possam-se obter informações da '*pose*' em tempo real. Além do EKF, têm-se outros estimadores não lineares com melhor desempenho, sendo muito utilizados para as mesmas aplicações, porém com maiores custos computacionais. Cabe ressaltar que a combinação entre esses estimadores com algoritmos de otimização podem dar uma estimativa mais próxima ao valor verdadeiro. Nesse sentido, podem ser visadas implementações futuras em *hardware* (baseadas em plataformas com FPGAs) de tais filtros.

No LEIA (*Laboratory of Embedded Systems and Integrated Circuits Applications*) que faz parte do GRACO (Grupo de Automação e Controle) da UNB (Universidade de Brasília) está sendo desenvolvido um projeto relacionado com um exoesqueleto autónomo para a reabilitação de pessoas com deficiências físicas (e.g. hemiplegia). Sendo esse um caso específico onde o estudo feito neste trabalho poderia ser aplicado, pois a estimação em *hardware* da posição e orientação com IMUs foi feita visando a implementação de uma rede de múltiplos sensores (IMUs). Nesse sentido, este trabalho sugere implementar uma IMU em cada centro de gravidade (ponto ao redor do qual a massa está igualmente distribuída) das extremidades inferiores do exoesqueleto de acordo com a Figura 1.1, sendo neste caso sete IMUs implementadas em total. Isto com o intuito de estimar tanto a posição quanto a orientação de cada um desses pontos onde se equilibram as forças (VAUGHAN et al., 1999). Portanto, diferentes soluções não lineares baseadas em filtros híbridos otimizados poderiam ser avaliadas, testadas e embarcadas em arquiteturas reconfiguráveis, a fim de obter uma estimação mais confiável (menor incerteza) dos mensurandos, além de um ganho na aceleração do processamento nas sete fusões sensoriais.

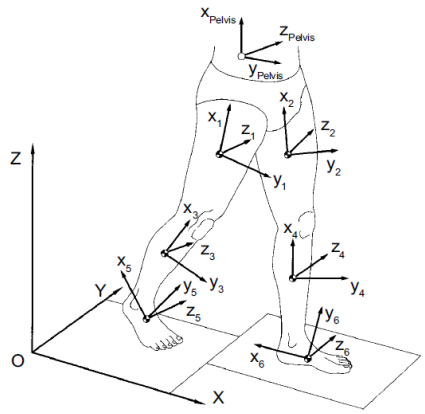


Figura 1.1: Centros de gravidade das extremidades inferiores (adaptado de (VAUGHAN et al., 1999)).

A Figura 1.2 apresenta uma abordagem mapeada em FPGA para uma rede multi-sensor baseada em IMUs, na qual observa-se que a arquitetura desenvolvida neste trabalho para fusão sensorial poderia replicar-se o número de vezes iguais ao número de IMUs. Com isto, os processamentos para as 'n'

IMUs seriam completamente paralelos e independentes, com o qual seria possível garantir a aceleração do processamento para a rede multi-sensor. No entanto, a única limitação para isto é a capacidade de recursos do FPGA a ser utilizado. Como já foi referido, um caso de aplicação de múltiplos sensores poderia ser o exoesqueleto, sendo que possivelmente sete IMUs seriam utilizadas.

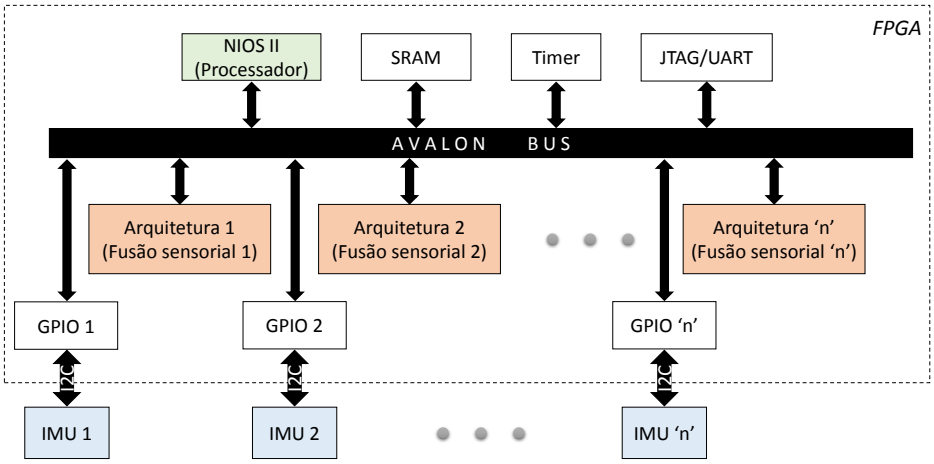


Figura 1.2: Abordagem mapeada em FPGA para uma rede multi-sensor baseada em IMUs. A transmissão de dados no barramento *Avalon* é de 32 bits.

1.3 Objetivos da dissertação

1.3.1 Objetivo geral

Esta dissertação tem como objetivo principal desenvolver e implementar uma arquitetura em *hardware* com o intuito de realizar a fusão sensorial de IMUs baseada no EKF, a fim de estimar a posição (*attitude*) e orientação (*heading*). Adicionalmente, estudar o desempenho de diferentes filtros híbridos otimizados para problemas de estimação de estados, prevendo uma futura implementação em *hardware*.

1.3.2 Objetivos específicos

Para atingir o objetivo geral deste trabalho, definiram-se os seguintes objetivos específicos:

- Estimar a posição (*attitude*) e a orientação (*heading*) através de IMUs.
- Desenvolver uma arquitetura da técnica de fusão sensorial com IMUs baseada no EKF.
- Implementar e validar a arquitetura no FPGA da família Altera DE2-115.
- Combinar diferentes filtros não lineares com algoritmos de otimização.
- Avaliar o desempenho dos filtros híbridos otimizados prevendo uma futura implementação em *hardware*.

1.4 Contribuições

Neste trabalho são apresentadas três principais contribuições. Em primeiro lugar, o desenvolvimento de uma plataforma (bancada) feita em impressão 3D para facilitar as calibrações e os testes de IMUs de baixo custo. A segunda contribuição é a implementação em FPGA de uma nova arquitetura para uma técnica de fusão sensorial baseada no EKF, utilizando uma IMU de 9 DOF (MPU9250) a fim de estimar a posição (rolagem e arfagem) e a orientação (guinada). A arquitetura foi projetada para alto desempenho, alta precisão e baixo consumo de recursos de *hardware*, permitindo compartilhar esses recursos de acordo com a função a ser realizada. Além disso, essa contribuição foi feita prevendo aplicações de rede multi-sensor. A terceira contribuição deste trabalho é a proposta dos filtros híbridos otimizados, pois essa abordagem envolve a implementação do Filtro de Kalman (EKF ou UKF) e do algoritmo de Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO) na etapa de amostragem e após da etapa do *resampling* do Filtro de Partículas (PF), respectivamente; com o intuito de melhorar a qualidade de estimação de estados. Cabe ressaltar que essa proposta não foi encontrada anteriormente na literatura.

1.5 Organização do documento

Este documento está composto por seis capítulos e dois anexos. No primeiro anexo mostram-se as medidas da bancada para calibração e testes de IMUs, enquanto o segundo anexo é o artigo baseado neste trabalho e publicado em 2017 no COBEM.

No capítulo 2 se faz uma análise de todos os aspectos relacionados com a estimação da posição e orientação através de fusão sensorial para Unidades de Medição Inercial (IMUs) baseada no Filtro de Kalman Estendido (EKF). Portanto, levam-se em consideração as calibrações dos diferentes sensores que compõem as IMUs, assim como as técnicas para realizar a fusão sensorial desse tipo de sensores. Além disso, apresenta-se uma breve descrição dos trabalhos correlatos nesta área.

Posteriormente, o capítulo 3 apresenta as diferentes plataformas utilizadas para implementar a fusão sensorial descrita no capítulo 2. Neste trabalho foram analisadas tanto as arquiteturas programáveis via software, tais como (a) Processadores de Propósito Geral (GPPs), (b) Processadores Digitais de Sinais (DSPs) e (c) Unidades de Processamento Gráfico (GPUs)), assim como os Circuitos Integrados de aplicação especifica (ASICs), quanto as arquiteturas reconfiguráveis, mais especificamente as FPGAs. Adicionalmente, no final deste capítulo se faz uma revisão da literatura sobre os trabalhos relacionados com a fusão sensorial de IMUs em FPGAs.

Em seguida, o capítulo 4 mostra a arquitetura proposta como solução para realizar a fusão sensorial de IMUs com EKFs em tempo real. Portanto, nesse capítulo são apresentados os módulos de bibliotecas em representação em ponto flutuante de 27 bits, para operações aritméticas (*soma*, *subtração*, *multiplicação*, *divisão* e *raiz quadrada*) e trigonométricas (*seno*, *cosseno*, *arcotangente* e *arcotangente2*). Além disso, apresentam-se os módulos desenvolvidos para fazer o processamento da fusão sensorial (*master*, *preCordic*, *EKF1*, *EKF2*, *accelerometer*, *magnetometer* e *conversion*). Finalmente, neste capítulo descreve-se a comunicação feita entre a IMU, o Processador (Nios II) e a arquitetura embarcada no FPGA.

No capítulo 5 apresenta-se uma abordagem de filtros híbridos otimizados baseados no Filtro de Partí-

culas (PF), com o intuito de realizar a estimação de estados de sistemas dinâmicos altamente não lineares. Neste caso, foram combinados o Filtro de Partículas (PF) com o Filtro de Kalman nas suas duas versões não lineares (EKF e UKF), com o algoritmo de Otimização por Enxame de Partículas (PSO). Além disso, nessas combinações foram levados em consideração dois métodos de adição de diversidade, a Aprendizagem Baseada em Oposição (OBL) e o Atrativo e Repulsivo (AR), a fim de melhorar a qualidade de estimação dos filtros híbridos otimizados. No final do capítulo apresenta-se uma comparação dos trabalhos correlatos sobre filtros combinados para estimação de estados.

No capítulo 6 mostram-se os resultados obtidos e a análise feita para cada um desses. Onde inicialmente descrevem-se e comparam-se algumas métricas de avaliação (MSE, RMSE e NRMSE). Seguidamente são apresentados os resultados correspondentes à técnica de fusão sensorial utilizada para as IMUs com EKFs, junto com os resultados da arquitetura dessa técnica de fusão sensorial embarcada no FPGA. Assim como no final deste capítulo apresentam-se também os resultados dos filtros híbridos otimizados. Finalmente, no capítulo 7 deste trabalho apresentam-se as conclusões e as respetivas propostas de trabalhos futuros.

Capítulo 2

Estudo da fusão sensorial para as IMUs com o EKF

Neste capitulo apresenta-se uma abordagem de estimação da posição (*attitude*) e orientação (*heading*), através de dois módulos (EKF1 e EKF2), sendo que em cada um desses é realizada uma fusão sensorial diferente. No primeiro módulo (EKF1) as medições do giroscópio e do acelerômetro são usadas para estimar os ângulos de rolagem e arfagem (posição), e no outro módulo (EKF2) o ângulo de guinada (orientação) é estimado a partir da combinação entre o giroscópio e o magnetômetro. Para testar essa abordagem, neste trabalho foram utilizadas dois tipos diferentes de IMUs: de 6 e 9 DOF, a MPU6050 e a MPU9250, respectivamente. Neste capítulo descrevem-se os algoritmos (filtro complementar e filtro de Kalman) mais utilizados para realizar a combinação de sensores. Além disso, apresenta-se uma bancada (plataforma) desenvolvida no LEIA através de impressão 3D afim de fazer calibração e testes com as IMUs. Finalmente, se faz uma contextualização dos trabalhos correlatos com respeito à fusão sensorial em IMUs. Cabe ressaltar que o estudo apresentado neste capítulo já foi publicado no congresso internacional do Cobem 2017 (vide anexo).

2.1 Orientação e posição

No contexto de IMUs existem dois métodos bastante utilizados para determinar a orientação e a posição de qualquer sistema ou objeto, os quais são os quaternions e os ângulos de Euler. O primeiro é um número complexo, que tem uma parte real e três partes imaginárias i, j e k (vide Equação 2.1), as quais representam as rotações nos eixos *X*, *Y* e *Z*, respetivamente (BIASI; GATTASS, 2007). Esse método é uma solução prática para representar as rotações, já que não tem restrição das mesmas; no entanto os processamentos dos quaternions têm um custo computacional alto.

$$q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \tag{2.1}$$

Por outro lado os ângulos de Euler estão representados por um ângulo de rotação para cada um dos três eixos ortogonais (X, Y e Z), também conhecidos como três graus de liberdade. Esses ângulos são rolagem,

arfagem e guinada, respetivamente. Embora este método é mais intuitivo e fácil de simbolizar em um gráfico, a sua principal dificuldade é o *gimbal lock*, o qual consiste na perda de um grau de liberdade. Nos Sistemas de Navegação Inercial (*Inertial Navegation Systems* - INS) o *gimbal lock* acontece no momento em que o eixo Y (ângulo de arfagem) é rotado $\pm 90^o$, fazendo que o eixo Z (ângulo de guinada) se torne paralelo ao eixo X (ângulo de rolagem), com o qual se perde um grau de liberdade (ângulo de rotação). Isto é conhecido como uma singularidade de eixos já que os dois eixos (X e Z) coincidem no mesmo plano fornecendo as mesmas medições (BIASI; GATTASS, 2007).

Os ângulos de Euler são definidos a partir das matrizes de rotação (vide Equações 2.2, 2.3 e 2.4) para cada eixo do plano (X, Y e Z). Para isto, é preciso ter em conta a ordem em que as rotações são realizadas.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{2.2}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.3}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

Na Tabela 2.1 apresentam-se as vantagens e desvantagens desses métodos, onde a principal diferença está no custo computacional e no *gimbal lock*. Neste trabalho optou-se por trabalhar com os ângulos de Euler, devido à fácil representação no espaço 3D e ao baixo custo computacional desse método se comparado com os quaternions. Além disso, devido a que os movimentos rotacionais da possível aplicação de estudo (exoesqueleto), na qual seria implementado o sistema de posição e orientação desenvolvido neste trabalho, não serão iguais ou maiores a 90^o. Portanto, para dita aplicação não se teria o problema do *gimbal lock*, o qual acontece somente em específicas aplicações que precisam de grandes rotações como são os sistemas de navegação inercial em aviões.

Tabela 2.1: Comparação entre os métodos para determinar a rotação 3D de um sistema

	Ângulos de Euler	Quaternions
Vantagens	Fácil representação no espaço 3D e baixo custo computacional	Liberdade de rotação
Desvantagens	*Gimbal lock*	Representação visual mais complexa e alto custo computacional

Tanto a posição (*attitude*) quanto a orientação (*heading*) são o sistema de referência utilizado nas aplicações de navegação, que especifica os movimentos de um objeto ou corpo no plano horizontal e no plano vertical, através dos ângulos de Euler (rolagem, arfagem e guinada). A rolagem é o ângulo correspondente ao eixo *X*, a arfagem no eixo *Y* e a guinada no eixo *Z*.

A Figura 2.1 mostra a representação dos ângulos de Euler em um sistema de navegação (STEPHEN, 2014). Para estimar esses três ângulos com alta precisão, os sensores MEMS são amplamente associados às técnicas de fusão de sensores. Este tipo de sensor tem várias vantagens, tais como: (a) baixo custo, (b) altamente sensitivo, (c) compacto (chip de silício único), (d) tamanho e (e) baixo consumo de energia.

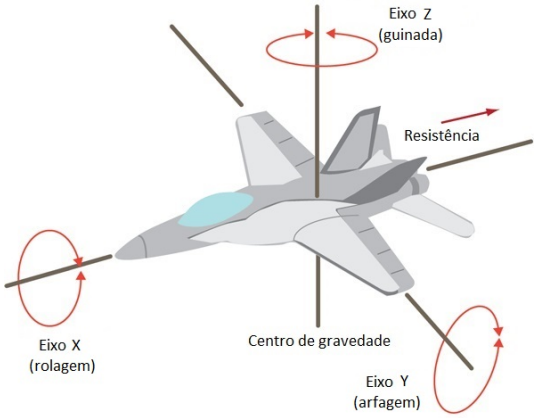


Figura 2.1: Representação dos ângulos de Euler (adaptado de (STEPHEN, 2014)).

2.2 Unidades de Medicão Inercial

As Unidades de Medição Inercial (IMUs) são Sistemas Microelectromecânicos (MEMS) desenvolvidos para uma ampla gama de aplicações, tais como robótica, visão computacional, inteligência artificial, entre outros. Os principais componentes dessas unidades inerciais são: (a) fonte de alimentação (3.3V), (b) processador, (c) sensores e (d) interface de comunicação (I2C ou SPI). Os sensores essenciais que compõem uma IMU são o acelerômetro, o giroscópio e o magnetômetro, sendo que com esses sensores são obtidas medidas de variações de velocidade e rotação, assim como as forças gravitacionais. Atualmente é comum encontrar diferentes IMUs com mais sensores, por exemplo, barómetro, sensor de temperatura, entre outros.

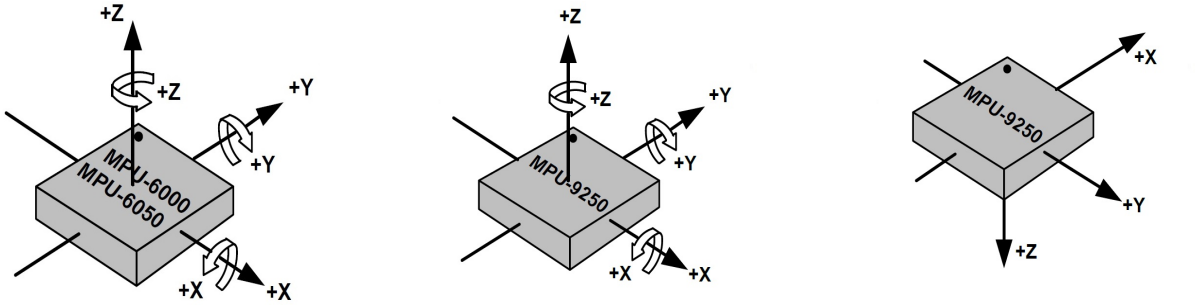
Cada sensor incorporado em uma IMU pode medir nos três eixos de referência (X, Y e Z). Na Tabela 2.2 apresenta-se uma comparação entre algumas IMUs de baixo custo, que são as mais utilizadas no desenvolvimento de projetos correlatos na área de robótica. Nessa Tabela têm-se em conta aspectos relevantes para cada IMU, tais como: (a) sensores embarcados no mesmo chip de silício, (c) graus de liberdade (DOF), (d) custo e (e) nível comercial no mercado (WINER, 2016).

Tabela 2.2: Comparação das IMUs de baixo custo.

CARACTERÍSTICAS	MPU6050	MPU9250	GY80	LSM9DS0	MPU6050 + HMC5883L
Único chip de silício	Sim	Sim	Sim	Sim	**Não**
Acelerômetro	MPU6050	MPU6500	ADXL345	LSM300D	MPU6050
Giroscópio	MPU6050	MPU6500	L3G4200D	L3G4200D	MPU6050

Magnetômetro	-	AK8963	HMC5883L	LIS3MDL	HMC5883L
Barômetro	-	-	**BMP180**	-	-
DOF	6	9	10	9	9
Custo	**R$13**	**R$45**	**R$88**	**R$110**	R$13 + R$18
Nível comercial	Alto	Alto	Médio	Baixo	Alto

Levando em consideração a Tabela 2.2, foram selecionadas as duas primeiras IMUs (MPU6050 (6 DOF) e MPU9250 (9 DOF)) a fim de realizar o estudo da estimação de posição e orientação. Isto devido que tanto a GY80 quanto a LSM9DS0 são IMUs com altos custos no mercado. Enquanto a quinta opção é uma combinação entre uma IMU de 6 DOF (MPU6050) com um magnetômetro externo (HMC5883L), portanto, o principal problema nesse caso é a sincronização de dois dispositivos diferentes. A Figura 2.2 mostra os eixos ortogonais e os ângulos de referência da MPU6050 e da MPU9250. A Figura 2.2(a) é correspondente ao acelerômetro e giroscópio da MPU6050 (INVENSENSE, 2013). A Figura 2.2(b) é referente ao acelerômetro e giroscópio da MPU9250, enquanto a Figura 2.2(c) representa o magnetômetro da MPU9250 (INVENSENSE, 2016).



(a) Acelerômetro e Giróscopio da MPU6050

(b) Acelerômetro e Giróscopio da MPU9250

(c) Magnetômetro da MPU9250

Figura 2.2: Eixos ortogonais e ângulos de referência das IMUs de 6 e 9 DOF (adaptado de (INVENSENSE, 2013) e (INVENSENSE, 2016)).

Na Tabela 2.3 são apresentadas as principais características desses dois dispositivos, na qual pode ser observado que a principal diferença entre as duas IMUs é a implementação do magnetômetro, o qual faz que a MPU9250 tenha três graus de liberdade a mais do que a MPU6050. No entanto, as outras características dessas IMUs são bastante similares, nos quesitos de: (a) precisão dos sensores embarcados (a) tensão de alimentação, (b) tamanho do pacote, (c) interface de comunicação, (d) corrente e frequência de operação, e (e) outros componentes (DMP e sensor de temperatura).

As duas IMUs utilizadas neste trabalho são fabricadas pela Invensense (ADDICORE, 2017). A MPU6050 de 6 DOF é composta de acelerômetro e giroscópio, enquanto que a MPU9250 é de 9 DOF, sendo composta de acelerômetro, giroscópio e magnetômetro. Para estimar tanto a posição quanto a orientação de cada IMU, utilizou-se uma técnica de fusão de sensores baseada no uso de um giroscópio como o modelo do sistema, enquanto o acelerômetro e o magnetômetro como modelos de medição. Essas duas IMUs mostram-se na Figura 2.3.

Tabela 2.3: Principais características da MPU6050 e MPU9250

CARACTERÍSTICAS	MPU6050	MPU9250
Sensores	Giroscópio e Acelerômetro	Giroscópio, Acelerômetro e Magnetômetro
Fonte de alimentação	2.375V-3.46V	2.4V-3.6V
Tamanho do pacote	16 x 20 mm	15 x 25 mm
Graus de liberdade	6	9
Interface de comunicação	I2C	I2C
Corrente de operação normal*	3.9mA	3.5mA
Frequência de operação	400kHz	400kHz
Outros	**DMP Sensor de Temperatura (-40 to +85 ^{o}C)	**DMP Sensor de Temperatura (-40 to +85 ^{o}C)

$^{(*)}$ Corrente de operação quando todos os sensores e o DMP estão habilitados

$^{(**)}$ DMP é o processador interno de cada IMU

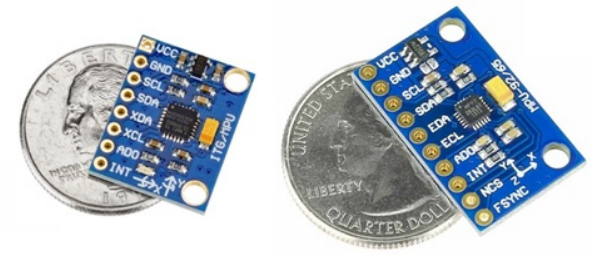


Figura 2.3: MPU6050 (6 DOF) e MPU9250 (9 DOF) (adaptado de (ADDICORE, 2017)).

Na Figura 2.4 detalha-se a configuração interna dos componentes da IMU de 9 DOF (MPU9250). Na mesma podem ser observados os três sensores (acelerômetro, giroscópio e magnetômetro) nos três eixos (X, Y e Z), além do sensor de temperatura. Cada um desses sensores tem um Conversor Análogo Digital (ADC) com resolução de 16 bits, em conjunto com um condicionador de sinal dedicado a unificar as medições digitais. O processador (DMP) embarcado na IMU realiza os cálculos da 'pose' a partir dos sensores da IMU, onde esses resultados são armazenados em registradores ou em uma fila de dados de tipo FIFO. Porém, os detalhes do algoritmo utilizado pelo DMP para a fusão sensorial são desconhecidos; portanto, esses valores são pouco confiáveis. Adicionalmente, esta IMU tem internamente blocos dedicados a fim de realizar a comunicação com dispositivos externos, através das interfaces I2C e SPI (INVENSENSE, 2016).

2.2.1 Comunicação I2C

O protocolo I2C foi desenvolvido em 1982 por *Philips Semiconductors* (PHILIPS, 2003). Este protocolo realiza a comunicação de escrita (W) e leitura (R) entre dois o mais dispositivos eletrónicos, a qual consiste na comunicação de um mestre e de um ou mais escravos; sendo que cada um dos escravos tem um endereço (7 bits) de identificação diferente. Além disso, o I2C utiliza dois pinos para realizar dita comunicação, o SCL (sinal de relógio) e o SDA (sinal de dados). Portanto, na Figura 2.5 mostra-se a sincronização do processo de comunicação, onde as sequências de inicio (*start*) e parada (*stop*) da comunicação I2C são geradas pelo *master* no momento que o SCL estiver em '1' (*high*). Porém o inicio é definido pela transição

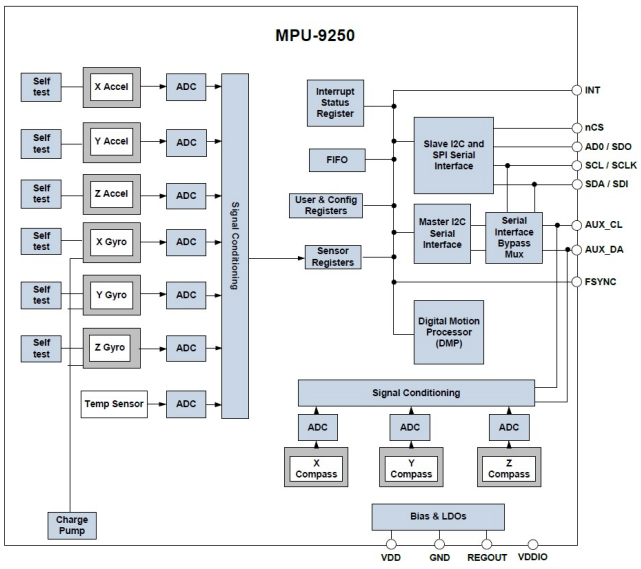


Figura 2.4: Diagrama de blocos da MPU9250 (adaptado de (INVENSENSE, 2016)).

do SDA de ‘1’ (*high*) para ‘0’ (*low*), enquanto a parada é definida pela transição do SDA de *low* para *high*. A partir do sinal de inicio, começa a transferência de dados entre o mestre e o escravo; sendo que cada bit deve ser enviado pelo SDA quando o SCL estiver em *low* (SEMICONDUCTORS, 2000).

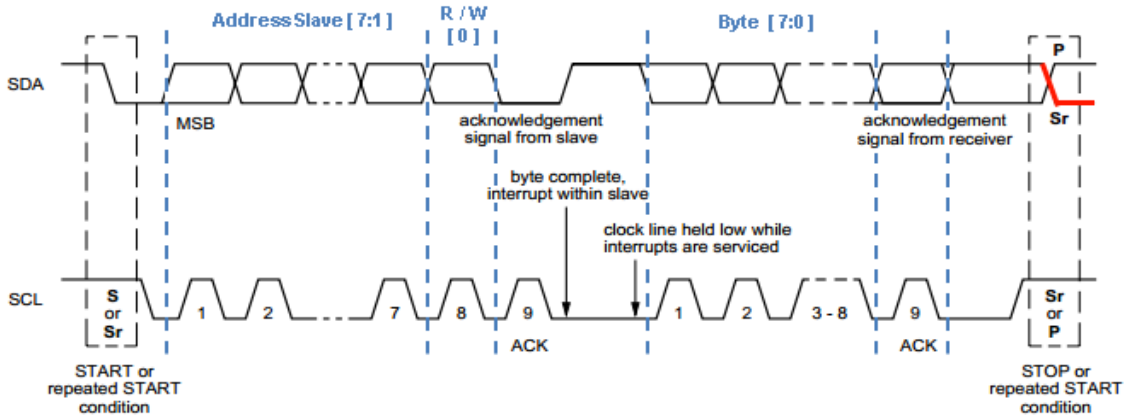
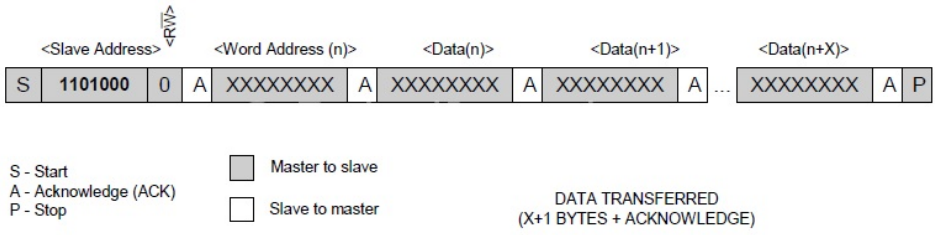


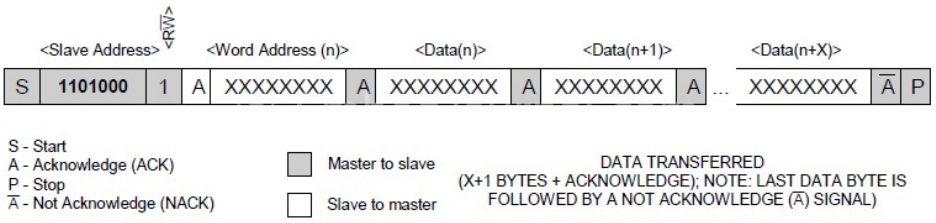
Figura 2.5: Sincronização da comunicação I2C (adaptado de (SEMICONDUCTORS, 2000)).

Nessa comunicação, inicialmente o mestre (transmissor) envia 1 *Byte* (8 bits), dos quais os 7 Bits Mais Significativos (*Most Significant Bit* - MSB) descrevem o endereço do escravo, enquanto o Bit Menos Significativo (*Less Significant Bit* - LSB) representa a opção a ser realizada (escrita (‘0’) ou leitura (‘1’)). Seguidamente, o escravo (receptor) retorna um sinal de reconhecimento (*Acknowledge* - ACK), o qual corresponde à confirmação de cada *Byte* transmitido. Onde ‘0’ significa que o receptor está pronto para aceitar mais dados; porém ‘1’ corresponde a uma interrupção, na qual não são transmitidos mais dados, nesse caso o sinal é de não reconhecimento (*Not Acknowledge* - NACK). Adicionalmente, o sinal de relógio mantém-se em *low*. Após disso, o mestre envia ou recebe um ou ‘n’ *Bytes*, segundo a operação a ser realizada (escrita ou leitura). Portanto, o receptor (mestre ou escravo segundo o caso) envia outro sinal de reconhecimento (ACK) (SEMICONDUCTORS, 2000). Isto pode ser visto na Figura 2.6, na qual

observam-se os dois casos de transmissão de dados, a escrita (vide Figura 2.6(a)) e a leitura (vide Figura 2.6(b)).



(a) Escrita simultânea em I2C



(b) Leitura simultânea em I2C

Figura 2.6: Processo de escrita e leitura na comunicação I2C (adaptado de (CHANDRASEKARAN, 2013)).

2.2.2 Giroscópio

Este sensor fornece informações referentes às variações de rotação (em unidades ^{o}/seg) exercidas sobre ele. No caso do giroscópio do tipo MEMS sua operação é baseada no efeito Coriolis, que consiste em gerar uma força externa através de um movimento rotativo de uma superfície circular. Além disso, as medições podem ser feitas em 1, 2 ou 3 eixos ortogonais. Algumas das principais aplicações e casos de uso do giroscópio do tipo MEMS são: (a) sistemas de navegação, (b) biomedicina, (c) telefones celulares e dispositivos inteligentes, (d) consoles de videogames, (e) automotivo, entre outros. A Figura 2.7 mostra a classificação dessas aplicações em relação à largura de banda de trabalho e ao alcance dinâmico da velocidade angular (KRAFT, 2000).

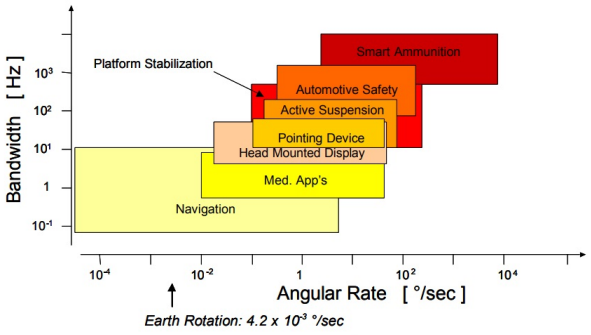


Figura 2.7: Aplicações do giroscópio (adaptado de (KRAFT, 2000)).

O giroscópio mede a velocidade angular de um corpo ou objeto, a qual é integrada em relação ao tempo para determinar a posição angular do mesmo. Esse processo de integração gera um erro cumulativo significativo no sinal de saída, aumentando em cada iteração. Portanto, as medições do sensor em um determinado tempo podem estar longe do valor correto. Esta é a principal desvantagem do giroscópio, também conhecido como erro *drift*. A solução para este problema é filtrar o sinal de saída. Por outro lado, a vantagem deste sensor é a alta precisão nas medições, já que a presença de ruído em suas medições é quase nula. A Tabela 2.4 apresenta as características do giroscópio que está integrado nas MPU6050 e MPU9250. Pode ser observado que o giroscópio é o mesmo nas duas IMUs estudadas.

Tabela 2.4: Características del Giroscópio da MPU6050 e MPU9250

CARACTERÍSTICAS	MPU6050	MPU9250
Eixos de medição	Eixo triplo (*X*, *Y* and *Z*)	Eixo triplo (*X*, *Y* and *Z*)
Alcance dinâmico	±250, ±500, ±1000 e ±2000 ^{o}/seg.	±250, ±500, ±1000 e ±2000 ^{o}/seg.
Precisão	0.01^o/s	0.01^o/s
Fator de escala de sensibilidade	131, 65.6, 32.8, 16.4 LSB/(^{o}/s)	131, 65.6, 32.8, 16.4 LSB/(^{o}/s)
Saída	Digital (16 bits ADCs)	Digital (16 bits ADCs)
Corrente de operação normal	3.6mA	3.2mA

Outra característica relevante deste sensor é que a partir das suas medições (w_x, w_y e w_z), tanto a posição (rolagem e arfagem) quanto a orientação (guinada), podem ser calculadas. Portanto, nas Equações 2.5, 2.6 e 2.7 os modelos matemáticos não lineares são apresentados, a fim de determinar os ângulos de Euler, levando em consideração as medições do giroscópio (KIM, 2011).

$$\dot{\phi} = w_x + w_y \sin\phi \tan\theta + w_z \cos\phi \tan\theta \tag{2.5}$$

$$\dot{\theta} = w_y \cos\phi - w_z \sin\phi \tag{2.6}$$

$$\dot{\psi} = w_y \cos\phi \tag{2.7}$$

2.2.3 Acelerômetro

O acelerômetro é um sensor que detecta a variação da velocidade linear que é exercida sobre ele, determinando assim a aceleração (em unidades "g", sendo equivalente 1g a 9,8 m/s^2) do corpo. A tecnologia MEMS que envolve aos acelerômetros tem a capacidade de realizar medições de aceleração em 1, 2 ou 3 eixos. Além disso, o funcionamento desses sensores está baseado em princípios físicos tais como piezoelétrico, piezoresistivo e capacitivo. A principal vantagem do acelerômetro é que as medições não têm um erro significativo. No entanto, sua principal desvantagem é a presença de ruído nas medições devido às acelerações externas (aceleração da gravidade), de modo que o sinal de saída desse sensor deve ser filtrado. As características do acelerômetro (que está integrado com a MPU6050 e a MPU9250) estão descritas na Tabela 2.5. Neste caso, o acelerômetro para as duas IMUs também é o mesmo.

A partir da aceleração fornecida por este sensor diferentes movimentos de um corpo ou objeto podem

Tabela 2.5: Características do acelerômetro da MPU6050 e MPU9250

CARACTERÍSTICAS	MPU6050	MPU9250
Eixos de medição	Eixo triplo (*X*, *Y* and *Z*)	Eixo triplo (*X*, *Y* and *Z*)
Alcance dinâmico	±2g, ±4g, ±8g and ±16g	±2g, ±4g, ±8g and ±16g
Precisão	60μg	60μg
Fator de escala de sensibilidade	16.384, 8.192, 4.096, 2.048 LSB/g	16.384, 8.192, 4.096, 2.048 LSB/g
Resolução (Saída)	Digital (16 bits)	Digital (16 bits)
Corrente de operação normal	500μA	450μA

ser estimados, tais como: (a) aceleração (b) vibração, (b) choque, (c) inclinação e (d) rotação. Esses movimentos são gerados por acelerações em diferentes períodos de tempo (SACCO, 2011). A Figura 2.8 mostra as principais aplicações dos acelerômetros em relação à largura de banda de trabalho e ao alcance dinâmico de aceleração (KRAFT, 2000). Além disso, a partir desta Figura, pode-se dizer que um acelerômetro com menor alcance de medição tem uma sensibilidade maior.

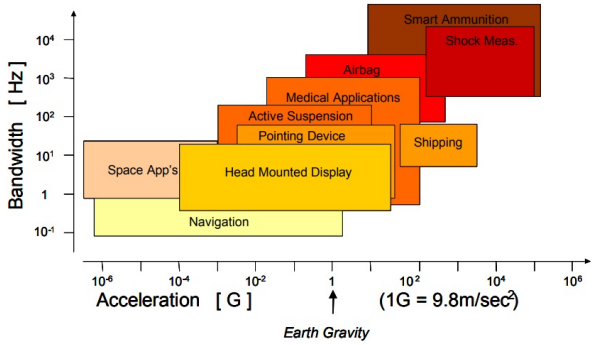


Figura 2.8: Aplicações do acelerômetro (adaptado de (KRAFT, 2000)).

Levando em consideração as Equações 2.8 e 2.9 e as medições correspondentes do acelerômetro nos três eixos (a_x, a_y e a_z); dois ângulos de Euler podem ser estimados, especificamente rolagem e arfagem (OZYAGCILAR, 2015). O modelo de acelerômetro na Equação 2.9 geralmente é usado em aplicações relacionadas à navegação espacial, pois o ângulo está limitado a -90^o e +90^o. Portanto, seu cálculo é determinado com a função ATAN. Caso contrário, a Equação 2.8 tem soluções no intervalo -180^o e +180^o, portanto, a função a ser usada (em uma aplicação de *software*) é a ATAN2 (OZYAGCILAR, 2015). Com as informações fornecidas pelo acelerômetro é possível determinar unicamente a posição (rolagem e arfagem), devido que com as medições de aceleração nos três eixos (X, Y e Z) não pode ser estimada a rotação no eixo perpendicular ao plano horizontal da terra.

$$\phi = \arctan\left(\frac{a_y}{a_z}\right) \tag{2.8}$$

$$\theta = \arctan\left(\frac{-a_x}{\sqrt{{a_y}^2 + {a_z}^2}}\right) \tag{2.9}$$

2.2.4 Magnetômetro

Este sensor mede a magnitude do campo magnético em uma determinada direção, sendo sua unidade de medição os Teslas "T"ou os Gauss "G". O Magnetômetro do tipo MEMS é um dispositivo eletrônico que possui a capacidade de medir até 3 eixos ortogonais (*X*, *Y* e *Z*). Sua principal aplicação é estimar a orientação de um corpo ou objeto, através de técnicas de fusão de sensores com o giroscópio, sendo sua função igual à de uma bússola eletrônica. Esta abordagem de aplicação é amplamente utilizada em sistemas de navegação. Embora também seja comum encontrar este sensor em telefones inteligentes. A Tabela 2.6 mostra as principais características do magnetômetro na MPU9250. Neste caso, o magnetômetro só está embarcado no MPU9250, já que o outro IMU de 6 DOF possui somente o acelerômetro e o giroscópio.

Tabela 2.6: Características do magnetômetro da MPU9250

CARACTERÍSTICAS	MPU9250
Eixos de medição	Triple-axis (*X*, *Y* and *Z*)
Alcance dinâmico	$\pm 4800\mu$T
Fator de escala de sensibilidade	1.46 mG/LSB
Resolução (Saída)	Digital (16 bits)
Corrente de operação normal	280μA

Semelhante ao acelerômetro MEMS, o magnetômetro MEMS é essencial para corrigir o erro *drift* do giroscópio através do procedimento de fusão sensorial. A principal desvantagem deste sensor é que o sinal de saída não é confiável, já que as medições possuem valores com pouca precisão (alto nível de incerteza) em relação ao valor real, devido às interferências de campos magnéticos externos (por exemplo, de dispositivos elétricos e eletrônicos). Essa desvantagem é gerada porque os fabricantes economizam nos custos de fabricação, no quesito de calibração do sensor. Então, eles determinam que cada usuário deve calibrar o magnetômetro a fim de obter resultados confiáveis de medição.

A calibração do magnetômetro do tipo MEMS deve ser feita para cada eixo do mesmo. Este tipo de calibração é dividido em duas partes: (a) *hard iron* e (b) *soft iron*. A *hard iron* refere-se a determinar o valor de *offset* ou *bias* do magnetômetro (levando em consideração uma amostra de medições), enquanto o *soft iron* consiste em dimensionar os resultados da calibração anterior (*hard iron*). Isto é, nesta segunda parte tenta-se melhorar a qualidade dos dados do magnetômetro, de modo que eles sejam mais parecidos com o valor real (WINER, 2017).

As Equações 2.10, 2.11 e 2.12 apresentam os modelos matemáticos correspondentes ao cálculo do terceiro ângulo de Euler (guinada), levando em consideração as medições do magnetômetro (m_x, m_y e m_z (OZYAGCILAR, 2015). As Equações 2.10 e 2.11 são os componentes nos eixos *X* e *Y* do campo magnético, respectivamente. A Equação 2.12 está restrita apenas em uma faixa de soluções entre -180^o e +180^o. Por esta razão, a função a ser utilizada é a ATAN2 (para obter os respetivos cálculos em *software*) (OZYAGCILAR, 2015). Neste caso, com o magnetômetro pode ser calculada somente a orientação (guinada), pois através das medições do campo magnético da terra (fornecidas por este sensor) é possível determinar a direção desse campo magnético. Portanto, com essas informações calcula-se com boa precisão a rotação sobre o eixo perpendicular ao plano horizontal terrestre (ABYARJOO et al., 2015).

$$B_{fy} = m_z \sin\phi - m_y \cos\phi \tag{2.10}$$

$$B_{fx} = m_x \cos\theta + m_y \sin\theta \sin\phi + m_z \sin\theta \cos\phi \tag{2.11}$$

$$\psi = \arctan\left(\frac{-B_{fy}}{B_{fx}}\right) \tag{2.12}$$

2.3 Fusão sensorial

A fusão sensorial é um processo que consiste em combinar informações (medições) de diferentes fontes (sensores), através de um mecanismo de fusão (e.g. filtro estimador), a fim de obter um sinal de saída com melhor qualidade daquela que poderia ser obtida a partir de um único sensor, reduzindo assim a incerteza da estimativa através das medições fornecidas pelos diferentes sensores (NXP, 2016). Na Figura 2.9 a estrutura de fusão sensorial é apresentada através de um filtro estimador, o qual pode ser do tipo: (a) filtro de Kalman (KF, EKF ou UKF), (b) filtro complementar, (c) filtro de partículas, (d) filtros híbridos otimizados (vide seção 5.4), entre outros. Esses filtros são ajustados através de diferentes parâmetros e dependendo dessa configuração os resultados da fusão sensorial podem ser os esperados.

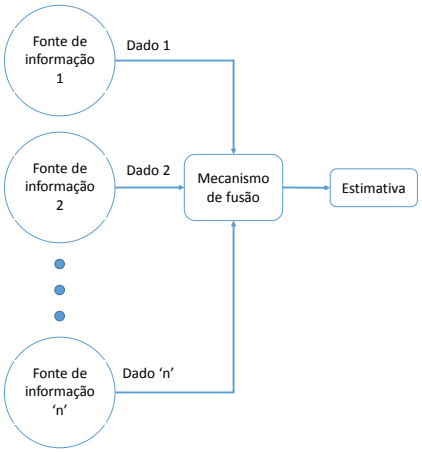


Figura 2.9: Estrutura da fusão sensorial.

Essa técnica de combinação de sensores requer que os sensores disponíveis sejam complementares uns aos outros, de modo que as desvantagens de cada um dos sensores sejam compensadas pelas vantagens dos outros. Portanto, se a combinação de sensores não estiver correta, será difícil obter os resultados esperados. Com respeito à fusão sensorial com IMUs, é comumente usada a combinação entre as informações fornecidas pelo giroscópio com as provenientes do acelerômetro, assim como as medições entre o giroscópio e o magnetômetro; com o intuito de determinar a posição (rolagem e arfagem) e a orientação (guinada), respectivamente. Essas duas combinações são levadas a cabo a fim de corrigir o problema do giroscópio (erro *drift*). Adicionalmente, essas fusões de sensores são muito utilizadas para aplicações relacionadas a sistemas de navegação (*attitude* e *heading*).

2.4 Filtros para fusão sensorial de IMUs

Atualmente, existem vários mecanismos de fusão sensorial para IMUs, entre os quais destacam-se o filtro complementar e o filtro de Kalman. Esses estimadores têm sido amplamente considerados nos sistemas de navegação a fim de determinar a posição e orientação através da representação dos ângulos de Euler (rolagem, arfagem e guinada).

2.4.1 Filtro complementar

Este filtro tem como objetivo a fusão de dados redundantes ou similares de diferentes sensores (HIGGINS, 1975). Além disso, é denominado complementar devido a que se comporta como uma combinação entre os filtros passa alto e o passa baixo, tornando-se em um filtro passa banda. A frequência de corte dos dois filtros para a estimação dos ângulos é a mesma (BROWN; HWANG, 1997). No caso dos sensores da IMU, nas medições do giroscópio (para períodos longos) o erro incrementa-se pelo processo de integração (erro *drift*), o qual pode ser visto como um ruído de baixa frequência (LLANOS, 2016), para o qual se utiliza um filtro passa-altas. Entretanto, o acelerômetro não apresenta ruídos acumulativos ao longo do tempo, mas o ruído ligado a este sensor é visto como um ruído de alta frequência, já que as medições dos movimentos do sensor são afetadas por acelerações externas, tal como a aceleração da gravidade. Para isto é utilizado um filtro passa-baixas (LLANOS, 2016). Portanto, neste caso, o filtro complementar elimina os ruídos de baixa frequência do giroscópio e de alta frequência do acelerômetro. O funcionamento deste filtro complementar para os sensores de exemplo apresenta-se na Figura 2.10

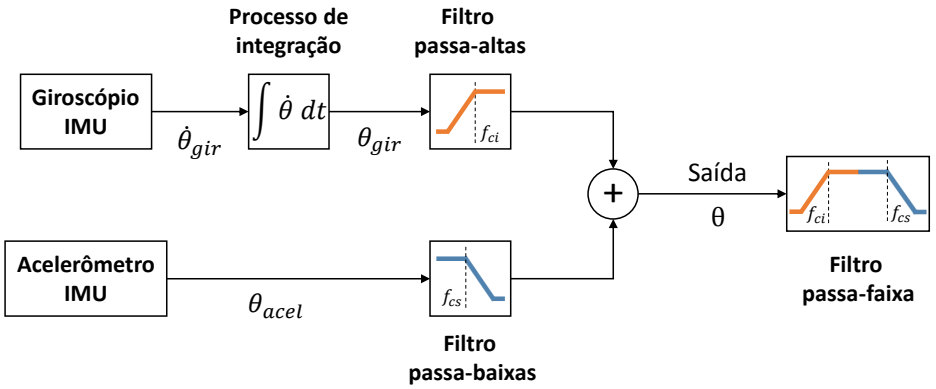


Figura 2.10: Funcionamento do filtro Complementar para o caso do giroscópio e acelerômetro.

O filtro complementar pode ser expressado através da Equação 2.13, a qual é utilizada a fim de determinar cada ângulo que deseja-se estimar. Os parâmetros A e B são duas constantes, sendo que a soma entre elas tem que ser igual a 1; geralmente para A é dado o maior valor (0.98) e para B o menor valor (0.02). No entanto, esses parâmetros podem ser ajustados a fim de calibrar o filtro, e assim obter um melhor desempenho deste.

$$\theta^k = A * (\theta^{k-1} + (\theta^k_{giro} * dt)) + B * \theta^k_{acel} \tag{2.13}$$

2.4.2 Filtro de Kalman

O filtro de Kalman (*Kalman Filter* - KF) é um estimador Bayesiano recursivo que estima para cada passo de tempo (k) o estado atual (x_k) de um sistema linear ou não linear (nos casos do Filtro de Kalman Estendido 'EKF', e do Filtro de Kalman Unscented 'UKF'), a partir de um estado anterior (x_{k-1}). Se baseando no modelo do sistema em espaço de estado, representado pelas Equações 2.14 e 2.15, onde, w_k é o ruído do processo e v_k é o ruído da medição, os quais são independentes e seguem uma Função de Distribuição de Probabilidade (*Probability Density Function* - PDF) Gaussiana. Além disso, são levados em consideração: (1) modelo de transição de estados (A), (2) modelo de observação (H), (3) covariância das variáveis de estado (P_k), (4) covariância do ruído do processo (Q), (5) covariância do ruído da medição (R) e (6) variável de medida (z_k).

$$x_k = Ax_{k-1} + w_k \tag{2.14}$$

$$z_k = Hx_k + v_k \tag{2.15}$$

No caso linear, as etapas (inicialização, predição e correção) do Filtro de Kalman podem ser observadas na Figura 2.11. A primeira etapa é inicializar as variáveis de estado ($\hat{x}_0^-$) e o erro de covariância (P_0^-). A segunda é a etapa de predição, que consiste em estimar essas duas variáveis no estado *a priori* (($\hat{x}_k^-$) e (P_k^-)). A última (terceira) etapa é de correção, na qual calcula-se o ganho de Kalman (K_k). Além disso, atualiza-se o estimado (*a posteriori*) das variáveis de estado ($\hat{x}_k$) e do erro de covariância (P_k) (WELCH; BISHOP, 2004). A principal desvantagem da versão linear do KF está relacionada a sua inaplicabilidade em sistemas não lineares, nos quais a estimação tende a não ser confiável (alta incerteza) (BROWN; HWANG, 1997).

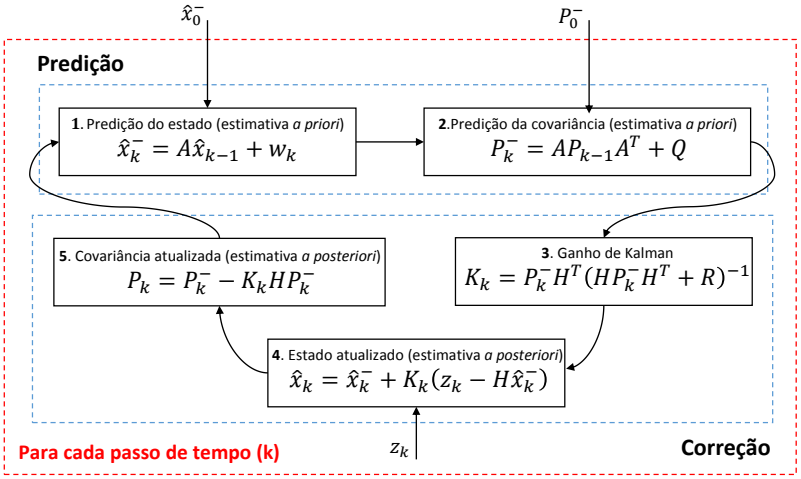


Figura 2.11: Inicialização, predição e correção do filtro de Kalman. Modelo de transição de estados (A), modelo de observação (H), covariância do ruído do processo (Q), covariância do ruído da medição (R), variável de medida (z_k).

2.4.2.1 Filtro de Kalman Estendido

O Filtro de Kalman Estendido (*Extended Kalman Filter* - EKF) é um dos filtros estocásticos mais conhecidos na área de filtragem e estimação de estados (x_k) para sistemas não lineares, porém é baseado sobre a versão linear do KF. Neste caso, levam-se em consideração os modelos matemáticos do processo (f_k) e da medição (h_k), descritos pelas Equações 2.16 e 2.17, respectivamente. Onde w_k e v_k são ruídos Gaussianos independentes (TURNER; SHERLOCK, 2013).

$$x_k = f_k(x_{k-1}, w_k) \tag{2.16}$$

$$y_k = h_k(x_k, v_k) \tag{2.17}$$

O EKF baseia-se na linearização do sistema através de duas matrizes Jacobianas A e H, as quais são as derivadas parciais da funções $f(x)$ e $h(x)$ em relação a cada variável de estado, respectivamente (KIM, 2011). O algoritmo 1 apresenta o pseudo-código do EKF. Da linha 1 à 7 está o ciclo repetitivo com medições atualizadas para cada passo de tempo (k), no qual a linha 2 e 3 representam a etapa de predição, nas quais realiza-se a estimação das variáveis de estado e o cálculo do erro de covariância (tendo em conta o Jacobiano A e a covariância de ruído do processo Q), respectivamente. A etapa de correção leva-se a cabo da linha 4 à 6, sendo na linha 4 o cálculo do ganho de Kalman (levando a consideração o Jacobiano H e a covariância de ruído da medição R), na linha 5 realiza-se a estimação das variáveis de estado com a medição atualizada (z_k) e na linha 6 calcula-se o erro de covariância.

Algoritmo 1 EKF

Entrada: Valores iniciais, $\hat{x}_0$, P_0. Medição, z_k.

Saída: Estimativa e erro de covariância, $\hat{x}_k$, P_k.

1: **para** k = 1 : N **faça**
2: $\quad x_k^- = f(x_{k-1})$
3: $\quad P_k^- = AP_{k-1}A^T + Q$
4: $\quad K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
5: $\quad x_k = x_k^- + K_k(z_k - h(x_k^-))$
6: $\quad P_k = P_k^- - K_k H P_k^-$
7: **fim para**

2.4.2.2 Filtro de Kalman Unscented

O Filtro de Kalman Unscented (UKF) foi proposto em 1997 como uma nova extensão do KF para sistemas dinâmicos com alto nivel de não linearidade. Sendo de todas as versões do KF, o UKF é a mais complexa e custosa computacionalmente; no entanto é a mais eficiente em termos de estimação de estados (JULIER; UHLMANN, 1997). A ideia de funcionamento do UKF baseia-se na transformação *Unscented* (UT), a qual consiste em propagar a média e a coviariância de x através da representação de pontos *sigma* x_i, os quais são transformados através da função $f(x)$ para assim calcular a média e a covariância dos novos pontos *sigma* $f(x_i)$ (KIM, 2011). Isso pode ser observado na Figura 2.12, na qual o gráfico da esquerda mostra a propagação da média e da covariância usando a amostragem de Monte Carlo. O gráfico do centro

representa a abordagem da linearização do EKF, enquanto o gráfico da direita mostra a propagação com a transformação *Unscented* (WAN; MERWE, 2000). Os pontos *sigma* não são escolhidos aleatoriamente (no caso dos métodos de amostragem de MC), ao contrário são selecionados de acordo a uma lógica determinista, com a qual é possível ter importantes informações da PDF com um pequeno número de amostras (pontos *sigma*) (PASCUAL, 2004).

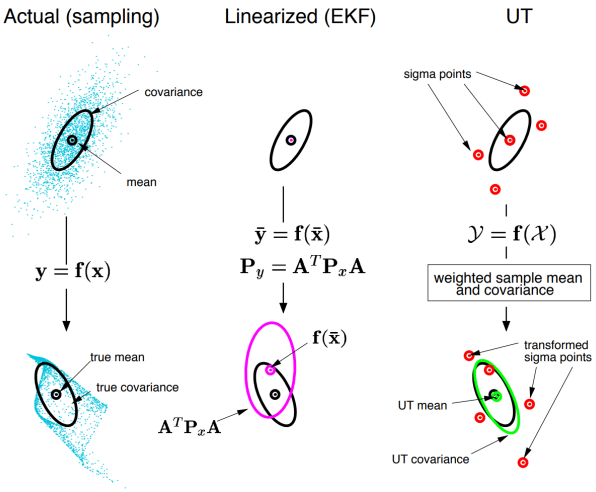


Figura 2.12: Transformação *Unscented* para a propagação da média e da covariância. (a) Estado atual (método do Monte Carlo), (b) estado linearizado (EKF), (c) transformação *Unscented* (UKF) (adaptado de (WAN; MERWE, 2000)).

O algoritmo 2 mostra o pseudo-código do UKF; Da linha 1 à 9 está o ciclo repetitivo com medições atualizadas para cada passo de tempo (k). Nas linhas 2, 3 e 4 é feita a etapa de predição, onde na linha 2 calculam-se os pontos *sigma* e os seus "pesos", tendo em conta o estado anterior (x_{k-1}) e a fatoração de Cholesky, a qual é muito utilizada para resolver sistemas de equações matriciais. Nas linhas 3 e 4 se faz a predição das variáveis de estado e do erro de covariância, e da medição e a sua covariância tendo em conta a transformação *unscented*, respetivamente. Nas linhas 5, 6, 7 e 8 a etapa de correção é realizada. Portanto, na linha 5 determina-se a covariância de x_k e z_k, onde 'n' representa o número de variáveis no vetor de estados. Na linha 6 calcula-se o ganho de Kalman. Na linha 7 realiza-se a estimação das variáveis de estado e finalmente, na linha 8 atualiza-se a covariância.

Algoritmo 2 UKF

Entrada: Valores iniciais, $\hat{x}_0$, P_0. Medição, z_k.

Saída: Estimado e erro de covariância, $\hat{x}_k$, P_k.

1: **para** k = 1 : N **faça**

2: $(x_i, W_i) = (\hat{x}_{k-1}, P_{k-1}, k)$

3: $(\hat{x}_k^-, P_k^-) = UT(f(x_i), W_i, Q)$

4: $(\hat{z}_k, P_z) = UT(h(x_i), W_i, R)$

5: $P_{xz} = \sum_{i=1}^{2n+1} W_i * \left\{f(x_i) - \hat{x}_k^-\right\} \left\{h(x_i) - \hat{z}_k\right\}$

6: $K_k = P_{xz} * P_z^{-1}$

7: $\hat{x}_k^+ = \hat{x}_k^- + K_k(z_k - \hat{z}_k)$

8: $P_k^+ = P_k^- - K_k P_z K_k^T$

2.5 Bancada para calibrações e testes

Na Figura 2.13 mostra-se a bancada (plataforma) para calibração e testes com IMUs, a qual tem 13 peças diferentes. Essa plataforma foi desenhada no SolidWorks 2016 e fabricada na impressora 3D do LEIA. Além disso, foi desenvolvida a fim de medir o valor verdadeiro (real) dos movimentos exercidos sobre a IMU, os quais são medidos com um transferidor (um para cada eixo) permitindo assim ao usuário ter uma referência da magnitude dos ângulos rotados. Adicionalmente, a bancada facilita os procedimentos para calibração e testes desses dispositivos eletrónicos (IMUs). Nesta bancada são três peças moveis as que permitem fazer as rotações nos 360^o em cada eixo X, Y e Z, através de rolamentos e parafusos. Além disso, têm-se umas guias que servem para visualizar e acompanhar os ângulos girados (valores reais) em cada um dos três transferidores (um para cada eixo).

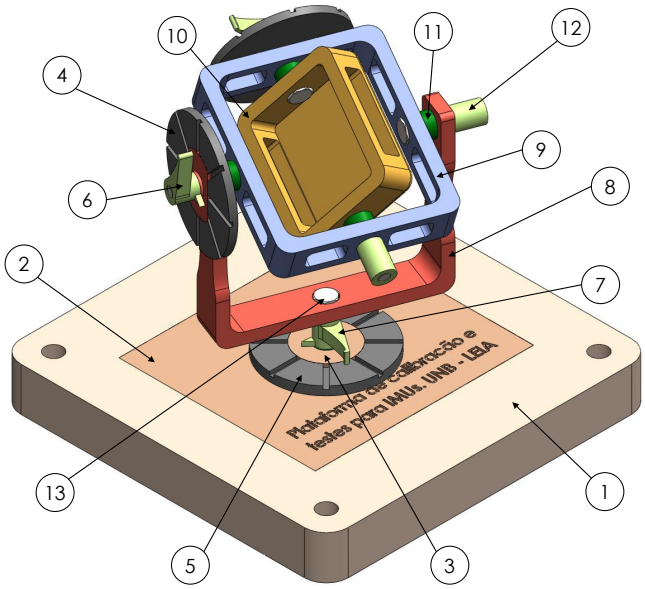


Figura 2.13: Bancada para calibração e testes com IMUs. As medidas correspondentes são 15x15x12.4cm

As peças da bancada são descritas na Tabela 2.7, na qual mostram-se as 13 peças que compõem a bancada e suas respectivas quantidades, sendo no total 23 componentes. As duas principais vantagens da plataforma é que o desenho é muito simples (vantajoso durante a impressão 3D), além de ser funcional. Pois os movimentos para as calibrações de cada um dos sensores e para os testes dos ângulos (rolagem, arfagem e guinada) estimados na fusão sensorial são realizados correctamente. No Anexo mostram-se as medidas de cada uma das peças que compõem a plataforma. Para as duas IMUs escolhidas na seção 2.2 (MPU6050 e MPU9250), utilizou-se este equipamento no estudo da fusão sensorial (PRIETO et al., 2017).

Tabela 2.7: Quantidades das peças da bancada

Número	Peça	Quantidade
1	Base externa	1
2	Base interna	1

3	Conector base	1
4	Bases verticais transferidor	2
5	Base horizontal transferidor	1
6	Guias verticais medição	2
7	Guia horizontal medição	1
8	Suporte no eixo Z	1
9	Suporte no eixo X	1
10	Suporte no eixo Y	1
11	Rolamentos	4
12	Casquilhos	2
13	Parafusos	5
	TOTAL	**23**

2.6 Métricas de avaliação

Na Tabela 2.8 apresentam-se diferentes medidas estatísticas (Média, Mediana e Desvio Padrão) e algumas métricas (MSE, RMSE e RMSE) para avaliar o desempenho de problemas de estimação de estados. As medidas estatísticas são aplicadas para só um conjunto de dados (amostra), enquanto as métricas de avaliação de desempenho são levadas em consideração para casos onde têm-se dois conjuntos de dados diferentes, um estimado e outro verdadeiro. Adicionalmente, nessa tabela se faz uma breve descrição, apresenta-se o modelo matemático e mostram-se as unidades em que são obtidos os resultados, para cada uma das medidas estatísticas e das métricas de avaliação de desempenho.

Tabela 2.8: Medidas estatísticas e métricas de avaliação de desempenho

Métrica	Descrição	Cálculo	Unidades
Média	É a soma dos valores de uma amostra dividido pelo tamanho da mesma, sendo o resultado obtido um **valor 'médio' dos valores**. A média é utilizada para avaliar **amostras com baixa dispersão**.	$\frac{\sum_{i=1}^{n} x^i}{n}$ (2.18)	Mesmas unidades das variáveis que são estimadas
Mediana	Representa o valor que divide os valores de uma amostra em dos subconjuntos de igual tamanho. A mediana é utilizada para **amostras com alta dispersão**. Essa métrica pode ser determinada tanto para **amostras ímpares** (vide Equação 2.19) quanto para **amostras pares** (vide Equação 2.20).	$x_{(n+1)/2}$ (2.19) ou $\frac{x_{(n/2)} + x_{(n/2)+1}}{2}$ (2.20)	Mesmas unidades das variáveis que são estimadas

Desvio padrão	É uma medida utilizada para avaliar a **dispersão** dos dados de uma amostra, com respeito à media dos mesmos. Portanto, o desvio padrão indica a **regularidade de um conjunto de dados**.	$\frac{\sum_{i=1}^{n}(x_i - \sum_{i=1}^{n} x^i/n)^2}{n-1}$ (2.21)	Mesmas unidades das variáveis que são estimadas
Erro quadrático médio (MSE)	Este tipo de erro mede a média dos erros ao quadrado, isto é, cada erro é determinado entre a **diferença entre o valor estimado e o valor observado (verdadeiro)**; levando em consideração as medições tanto positivas quanto negativas. O MSE faz referencia ao valor da covariância entre dois conjuntos de dados diferentes.	$\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}$ (2.22)	Mesmas unidades das variáveis que são estimadas
Raiz do erro quadrático médio (RMSE)	Este erro está baseado no MSE. Portanto, o RMSE determina a raiz do MSE, o qual é igual a determinar o **desvio padrão** entre dois conjuntos de dados diferentes. Esta métrica é uma das mais utilizadas devido a sua **facilidade de analisar os resultados**.	$\sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}$ (2.23)	Mesmas unidades das variáveis que são estimadas
Raíz do erro quadrático médio normalizado (NRMSE)	O NRMSE é a versão normalizada do RMSE, pelo qual, neste caso o resultado é normalizado a fim de obter um erro dentro da faixa 0 a 1, pois isto permite ter um **entendimento mais claro do erro calculado**. Além de ter a possibilidade de **comparar com outros resultados**, sem levar a consideração **as unidades das variáveis**.	$\frac{\sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}}{\bar{y}}$ (2.24) ou $\frac{\sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}}{y_{max} - y_{min}}$ (2.25)	Percentagem (%)

2.7 Trabalhos correlatos sobre técnicas de fusão sensorial IMUs

Na Tabela 2.9 apresentam-se alguns trabalhos correlatos à fusão sensorial com IMUs de baixo custo. Nesta Tabela detalham-se diferentes caraterísticas, tais como: (a) descrição, (b) sensores, (c) taxa de amostragem, (d) algoritmo utilizado e (e) métrica de avaliação. Cabe ressaltar que em todos os trabalhos os sensores das IMUs foram calibrados prévio a processamento (fusão sensorial).

Tabela 2.9: Revisão bibliográfica dos trabalhos correlatos de fusão sensorial de IMUs para determinar a posição e orientação

Ref.	Descrição	Sensores	Taxa de amostragem (IMUs)	Algoritmo	Métrica de avaliação
(CARON et al., 2006)	Determinação da posição e velocidade de um veículo terrestre com uma abordagem de multisensores (giroscópios e acelerômetros)	IMU de 6 DOF e GPS	13 ms / 75 Hz	KF	Metros
(SCHOPP et al., 2009)	Cálculo da posição linear de um corpo, através de multiplicações de matrizes **utilizando só acelerômetros**	4 IMUs de 3 DOF	**4 ms / 250 Hz**	UKF	RMSE
(LOU et al., 2011)	Estimação da *attitude* para um sistema de navegação de um robô móvel, a fim de minimizar os erros sistemáticos dos **sensores (giroscópios, acelerômetros e magnetômetros)** e otimizar a fusão sensorial	IMU de 9 DOF	**10 ms / 100 Hz**	DCM	Graus
(ZHAO; WANG, 2012)	Método de compensação dos erros dos sensores inerciais (giroscópios e acelerômetros) para localização de movimento através da fusão sensorial	IMU de 6 DOF, magnetômetro e ultrassom	40 ms / 25 Hz	EKF	Graus
(LIGORIO; SABATINI, 2013)	Comparação de duas abordagens com EKFs para estimação de orientação de um sistema câmera-IMU **(giroscópios, acelerômetros e magnetômetros)**, o primeiro baseado em DLT e o segundo nos erros de projeção	IMU de 9 DOF e camêra	**10 ms / 100 Hz**	EKF	RMSE

(BENINI; MANCINI; LONGHI, 2013)	Localização em ambiente interior de um UAV usando rede de sensores (giroscópios e acelerômetros) sem fio	IMU de 6 DOF e odômetro	100 ms / 10 Hz	EKF	Metros
(LIU; NOGUCHI; ISHII, 2014)	Estimação da *attitude* para um robô agrícola, utilizando um filtro de média para o giroscópio com o intuito de apagar o ruído gerado ao ar livre, além do uso do acelerômetro	IMU de 6 DOF	-	KF	Graus
(BERGAMINI et al., 2014)	Estimação da orientação para diferentes abordagens de fusão sensorial, sendo avaliados em tarefas manuais e locomoção de pessoas através de **giroscópios, acelerômetros e magnetômetros**	IMU de 9 DOF	**10 ms / 100 Hz**	KF e CF	Graus
(ABYARJOO et al., 2015)	Deteção da orientação em três dimensões, através da fusão sensorial para determinar a *attitude* com o giroscópio e acelerômetro, enquanto para a *heading* só é usado o magnetômetro	IMU de 9 DOF	112 ms / 9 Hz	KF	Graus
Este trabalho	Fusão sensorial flexível para estimação da posição (rolagem e arfagem) e orientação (guinada) através de modelos não lineares e **(giroscópios, acelerômetros e magnetômetros)**	IMU de 9 DOF	**7 ms / 143** Hz	EKF	NRMSE

Da Tabela 2.9 pode ser observado que a maior parte dos trabalhos de fusão sensorial para estimação da posição e/ou orientação utilizam IMUs de 6 DOF (acelerômetros e giroscópios); sendo que a taxa de amostragem depende do número de sensores que compõem as IMUs. Todos os trabalhos referenciados fizeram uma calibração dos sensores prévia ao processamento, devido que basearam-se em IMUs de baixo custo. No entanto, diversos algoritmos são utilizados para fusão sensorial, entre esses o mais comum é o KF (vide referências da Tabela 2.9), tendo em conta que o mesmo gera uma boa precisão, a um custo computacional moderado. Cabe destacar que nenhum dos trabalhos citados utilizou uma técnica flexível de fusão sensorial para IMUs de 9 DOF. Este quesito é muito importante porque permite projetar sistemas independentes a fim de determinar a posição e a orientação. Isto é, no caso que se quiser estimar só a posição, utilizaria-se uma IMU de 6 DOF e mesmo assim a técnica projetada para 9 DOF funcionaria. Além disso, alguns trabalhos não utilizaram métricas para avaliar a estimação da fusão sensorial, devido que nesses trabalhos os resultados obtidos foram avaliados e comparados de acordo às unidades utilizadas (graus ou metros).

Capítulo 3

Plataformas para implementar a fusão sensorial usando IMUs

Neste capítulo apresentam-se diversas propostas de arquiteturas que têm sido desenvolvidas com o intuito de embarcar algoritmos específicos de fução sensorial. Cada arquitetura tem diferentes vantagens e desvantagens, sendo que a escolha destas depende principalmente: (a) complexidade do algoritmo a embarcar, (b) eficiência de processamento do algoritmo embarcado, (c) consumo energético e (d) espaço ocupado na plataforma (MOSQUERA, 2016). Estas propostas de arquiteturas são classificadas aqui de: (a) arquiteturas programáveis via *software*, as quais são aquelas de propósito geral (*General Purpose Processors* - GPPs), (b) arquiteturas de propósito específico, tais como as Unidades de Processamento Gráfico (*Graphics Processing Unit* - GPU) e (c) as arquiteturas reconfiguráveis baseadas em Arranjos de Portas Programáveis em Campo (Field Programmable Gate Arrays - FPGAs).

Para aplicações em tempo real, diferentes etapas devem ser feitas antes da leitura do resultado; isto é, durante o processamento a informação pode passar por uma ou mais operações, tais como comparações, deslocamentos, operações aritméticas (multiplicações, divisões, somas e subtrações), etc. Portanto, o tempo de execução no processamento é uma das variáveis mais relevantes neste tipo de aplicações (LENART, 2008).

No caso dos algoritmos de fusão sensorial, os mesmos têm sérias restrições computacionais devido ao: (a) processamento em tempo real, (b) cálculos e operações de alta complexidade matemática e (c) sincronização de diferentes instrumentos de medição (sensores). As arquiteturas que suportam essas exigências, são aquelas que têm alta capacidade de processamento a fim de acelerar ou paralelizar esses algoritmos. Tanto as GPUs quanto o FPGAs são tecnologias candidatas para realizar essas duas funções: aceleração e paralelismo. A principal diferença está em que os FPGAs são projetados para permitir a reconfiguração do *hardware*, a fim de criar circuitos que permitam aproveitar o paralelismo intrínseco dos algoritmos. Por outro lado, as GPUs têm uma configuração de *arquitetura de hardware* pré-definida (BERTEN, 2016).

Na Figura 3.1 apresenta-se uma comparação qualitativa entre GPUs e FPGAs, na qual detalham-se as vantagens e as desvantagens de cada uma dessas plataformas. As GPUs são melhores em termos da capacidade total de processamento de ponto flutuante, esforço de desenvolvimento e custo do dispositivo. No entanto, os FPGAs têm bons recursos de processamento, além da alta eficiência térmica e energética, assim

como a flexibilidade na integração com outros dispositivos através das interfaces de comunicação. Adicionalmente, os FPGAs têm alta flexibilidade através da integração de lógica programável e de periféricos padronizados, enquanto as GPUs estão limitadas ao protocolo PCIe (BERTEN, 2016).

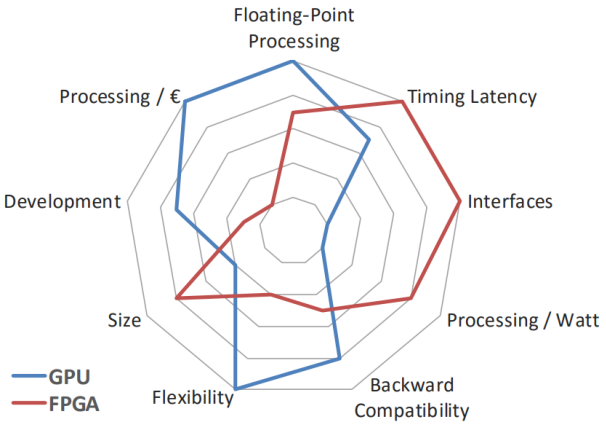


Figura 3.1: Comparação qualitativa entre a GPU e o FPGA (adaptado de (BERTEN, 2016)).

3.1 Arquiteturas programáveis via *software*

As arquiteturas programáveis via *software* são caracterizadas por serem fáceis de programar e verificar. Além disso, as instruções programadas nestas plataformas são executadas sequencialmente, devido que os algoritmos são descritos em linguagem de *software* e o modelo de computação é de Von Neumann (NEUMANN, 2012).

3.1.1 Processador de Propósito Geral (GPP)

Os processadores de propósito geral (GPPs) são projetados com o intuito de realizar diferentes atividades, isto é, esse tipo de processadores não são projetados para algum propósito especial. Algumas das principais vantagens de utilizar os GPPs são: (a) custo e tempo de prototipagem baixo, (b) alto desempenho no processamento de operações, (c) alta flexibilidade a nível de *software* (RANGANATHAN; ADVE; JOUPPI, 1999). Alguns GPPs utilizados são: (a) os processadores *soft-processor* (NIOS e Microblaze), (b) os processadores *hard processors* (ARM, PowerPC), e (a) todos os microprocessadores dos computadores (Intel e AMD) (MOSQUERA, 2016). A arquitetura básica desses processadores está baseada em uma unidade de controle, uma unidade *datapath* e uma unidade de memória; sendo que as duas primeiras unidades compõem a CPU (vide Figura 3.2) (VAHID; GIVARGIS, 2002).

3.1.2 Processador digital de sinal (DSP)

Os processadores digitais de sinal (DSPs) são microprocessadores que baseiam-se em arquiteturas MAC (multiplicação e acumulação), as quais combinam as operações de multiplicação e acumulação em uma única instrução de processamento de sinais. Uma vantagem dos DSPs é que são projetados para tarefas

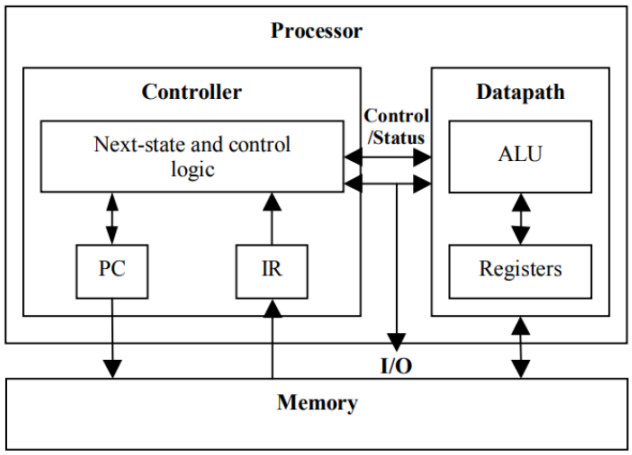


Figura 3.2: Processador de propósito geral (adaptado de (VAHID; GIVARGIS, 2002)).

que requerem alta complexidade de cálculo. Hoje em dia há DSPs que têm sido desenvolvidos com taxas de *clock* iguais a 1 GHz (JR, 2009). Os DSPs também podem estar embarcados em alguns FPGAs, o qual é conhecido como lógica *hardwired*, pois realizam funções de acordo às conexões configuradas internamente nesse dispositivo reconfigurável (nível de *hardware*) (ALTERA, 2017) (XILINX, 2017).

As aplicações dos DSPs dependem dos requisitos de solução a um problema específico. Neste caso, existem duas categorias tendo em conta a aplicação. Por exemplo, no caso dos sistemas comerciais (e.g. dispositivos de teléfonia móvel) e no caso de processamento de algoritmos complexos (e.g. processamento de imagens). A largura da palavra mais comum dos DSPs para representação fixa é de 16 bits, enquanto para o ponto flutuante é de 32 bits. Porém, o tamanho do barramento de dados influi nos recursos utilizados; portanto a melhor opção a fim de poupar recursos é determinar a menor largura possível da palavra, para que a aplicação possa funcionar otimamente (SALAZAR, 2000).

3.1.3 Unidade de processamento gráfico (GPU)

As GPUs são processadores desenhados para a renderização de imagens e vídeos em 3D de alta resolução em tempo real. Suas principais aplicabilidades estão focadas em sistemas que integram essas tarefas, tais como em computadores e consoles de videogame. Esse tipo de processadores têm evoluído nos últimos anos. Na Figura 3.3 podem ser observadas arquiteturas do tipo CPUs e GPUs, onde a arquitetura das unidades de processamento gráfico está baseada em um conjunto de multiprocessadores, sendo que cada microprocessador contém internamente um conjunto de núcleos (também conhecidos como processadores escalares). Por outro lado, as CPUs têm poucos núcleos. Portanto, se fosse comparado o desempenho desses dois tipos de processadores em relação à potência e velocidade de processamento, as GPUs têm vantagem com respeito às CPUs (GREGG; HAZELWOOD, 2011). Levando em consideração esses fatos, as GPUs têm-se utilizado para aceleração de algoritmos complexos, através da paralelização dos mesmos.

As GPUs apresentam um alto consumo energético devido ao elevado número de recursos (processadores e núcleos) que estão embarcados nestas plataformas. A pesar das diferenças, a combinação entre as CPUs e as GPUs é comummente utilizada, sendo um exemplo de multiprocessamento heterogéneo (PAT-

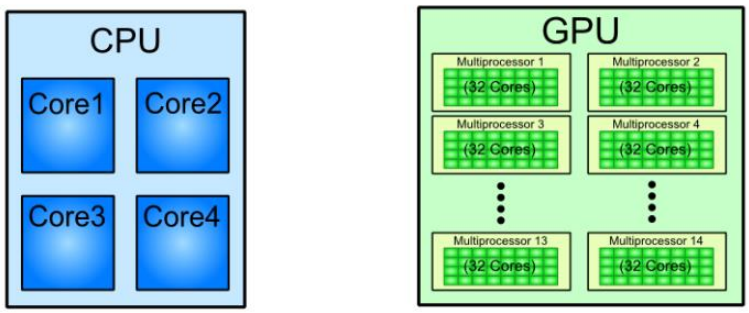


Figura 3.3: Comparação entre a CPU e a GPU (adaptado de (DUTCH, 2017).

TERSON; HENNESSY, 1994); onde as GPUs realizam cálculos múltiplos, enquanto as CPUs realizam cálculos únicos. Além disso, as instruções de máquina para as GPUs são geradas através de interfaces de programação de aplicativos de alto nível (e.g. OpenGL e DirectX da Microsoft) e linguagens orientados a gráficos (e.g. C para gráficos (Cg) da NVIDIA e o Linguagem de programação em alto nível para criar *shaders* (HLSL) da Microsoft) (PATTERSON; HENNESSY, 1994).

3.2 Circuito integrado de aplicação específica (ASIC)

Os circuitos integrados de aplicação específica (ASICs) são sistemas em chip, que são projetados com o intuito de realizar tarefas específicas. Para descrever as arquiteturas são usadas linguagens de descrição de *hardware* (e.g. Verilog e VHDL). Para esta tecnologia a arquitetura não pode ser reconfigurada depois da fabricação, pelo qual são chamadas de arquiteturas desenvolvidas para um só propósito. Uma das características deste tipo de plataformas é que o cliente é o mesmo desenhador da arquitetura, sendo umas das principais diferenças com os outros tipos de circuitos integrados (EINSPRUCH, 2012).

Os circuitos integrados são classificados de acordo ao número de portas lógicas (vide Figura 3.4) embarcadas no chip: (a) pequena escala (SSI), onde têm-se de 1 a 10 portas/chip (e.g. portas AND, OR, NOT, XOR), (b) escala média (MSI) que inclui de 10 a 100 portas/chip (e.g. *Flip Flops*, somadores, contadores, multiplexadores), (c) grande escala (LSI), na qual há de 100 a 10.000 portas/chip (e.g. memórias de pequeno espaço), (d) escala muito alta (VLSI) que tem de 10.000 a 1.000.000 portas/chip (e.g. memórias de grande espaço) e (e) escala ultra alta (ULSI), composta por mais de 1.000.000 portas/chip (e.g. microprocessadores) (KAESLIN, 2008).

3.3 Arquiteturas reconfiguráveis

As arquiteturas reconfiguráveis permitem que o *hardware* possa ser programável via *software*, isto é, um tipo de arquitetura que tem a flexibilidade de ser programada e configurada para tarefas específicas, definidas pelo usuário. Alguns dispositivos baseados em lógica programável são: (a) Dispositivos Lógicos Programáveis Complexos (CPLDs), (b) Dispositivos Lógicos Programáveis Simples (SPLDs) e (c) matrizes de portas programáveis em campo (FPGA). A principal diferença entre os CPLDs e SPLDs com os FPGAs é que os dispositivos lógicos (SPLDs e CPLDs) têm um número limitado de portas lógicas

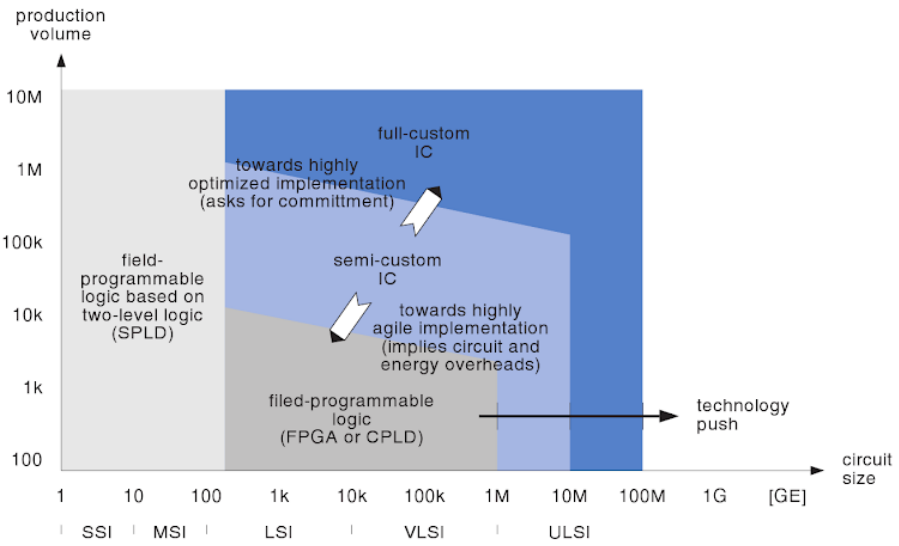


Figura 3.4: Classificação dos ASICs (adaptado de (KAESLIN, 2008)).

(MAXFIELD, 2004). Adicionalmente, ao comparar os FPGAs com os ASICs, o custo e a facilidade no desenvolvimento dos projetos nos FPGAs é menor, além de ser reconfigurável. Porém, os ASICs têm consumo de potência menor (AMARA; AMIEL; EA, 2006). Neste trabalho utilizou-se a FPGA para embarcar a técnica de fusão sensorial para IMUs, tendo em conta as vantagens desse tipo de arquitetura e devido a que no laboratório LEIA da UNB têm-se os equipamentos e os conhecimentos sobre projetos em FPGA.

3.3.1 FPGA

FPGAs são chips de silício inventadas em 1985 pelo cofundador da Xilinx, Ross Freeman (INSTRUMENTS, 2013). Estas arquiteturas consistem em um arranjo matricial, composto principalmente de três elementos básicos: (a) blocos lógicos configuráveis (*Configurable Logic Blocks* - CLBs), (b) portas de entradas e saídas (I/O) configuráveis, e (c) interconexões programáveis. Os CLBs fornecem a lógica básica de um circuito que pode ser reconfigurada de acordo às instruções do programa; além disso, um CLB está composto por um conjunto de elementos lógicos básicos (*Basic Logic Elements* - BLE), os quais baseiam-se em *Look-Up Tables* (LUTs), *flip flops* (FF) e multiplexadores, sendo que as LUTs são tabelas de verdade das funções booleanas (NOT, AND, OR , XOR) e das combinações entre essas. Portanto, os CLBs são ligados entre si para criar funções maiores, através das interconexões direcionadas pelos blocos de conexão (*Connection Blocks* - CBs) e os blocos de chaves (*Switch Blocks* - SB), os quais são compostos por transístores. A arquitetura básicas da FPGA com todos os componentes detalha-se na Figura 3.5 (FAROOQ; MARRAKCHI; MEHREZ, 2012).

No entanto, os FPGAs modernos envolvem internamente DSPs, RAM, EEPROM, PLLs, microprocessadores (ARM, μBlaze, Nios, etc.), ADCs, *clock's*, entre outros componentes. Os DSPs são circuitos pré-embarcados para operações de multiplicação e acumulação de bits, com os quais consegue-se poupar elementos lógicos (LUTs), sendo esse último o método tradicional de implementação de multiplicadores nos FPGAs; portanto, os processamentos com uma maior quantidade de bits irão precisar mais DSPs ou multiplicadores embarcados (segundo a tecnología do FPGA utilizado), ocupando assim mais espaço e

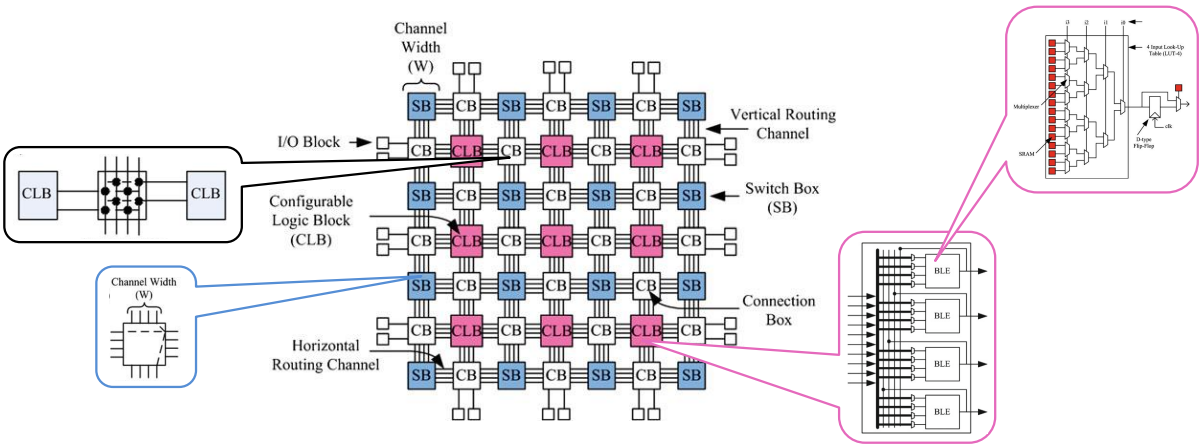


Figura 3.5: Arquitetura básica do FPGA (adaptado de (FAROOQ; MARRAKCHI; MEHREZ, 2012)).

recursos nessas arquiteturas (HAZANCHUK, 2003).

O fluxo de desenho de um projeto no FPGA é apresentado na Figura 3.6. Inicialmente o projeto deve estar na linguagem VHDL a nível RTL (*Register Transfer Level*), isto é, a descrição do comportamento do circuito através de registradores. A primeira etapa é a *synthesis*, na qual realiza-se a compilação e otimização do projeto a nível de portas, onde gera-se um *netlist* e otimiza-se a velocidade e a área do FPGA, respectivamente. No final de esta etapa o projeto é simulado ao igual que na segunda etapa do *place and route*, que consiste em gerar um circuito para um dispositivo programável ou para a fabricação de um ASIC (PEDRONI, 2004).

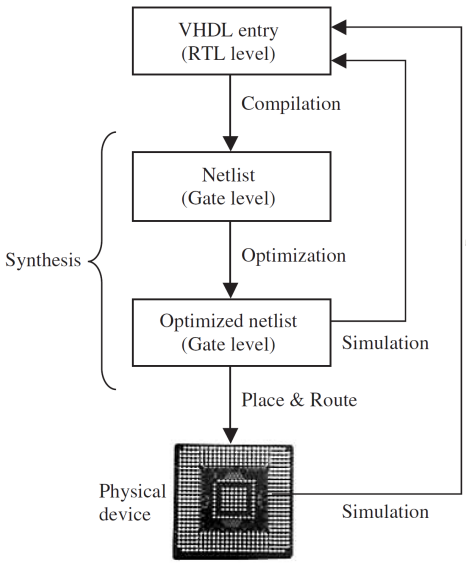


Figura 3.6: Fluxo do projeto no FPGA (adaptado de (PEDRONI, 2004)).

3.3.1.1 Cyclone IV DE2-115

Neste projeto utilizou-se o dispositivo lógico programável (FPGA) Cyclone IV DE2-115 da Altera, para embarcar a arquitetura desenvolvida e apresentada no capítulo 4. Esse FPGA conta principalmente com: (a) 114.480 elementos lógicos (LEs), (b) 3.888 Kbits memória embarcada, (c) 266 multiplicadores embarcados de 18x18 bits, (d) 4 PLLs de propósito geral, (e) 528 I/Os e (d) 1 microprocessador 100MHz (Nios II). O kit DE2-115 conta com dispositivos de memoria: (a) SDRAM (2x64MB), (b) SRAM (2MB), (c) Flash (8MB) e (d) EEPROM (32Kbit); além de outros componentes (*clock* 50MHz, periféricos, etc). Na Figura 3.7 detalham-se as especificações do Kit DE2-115 (ALTERA, 2010).

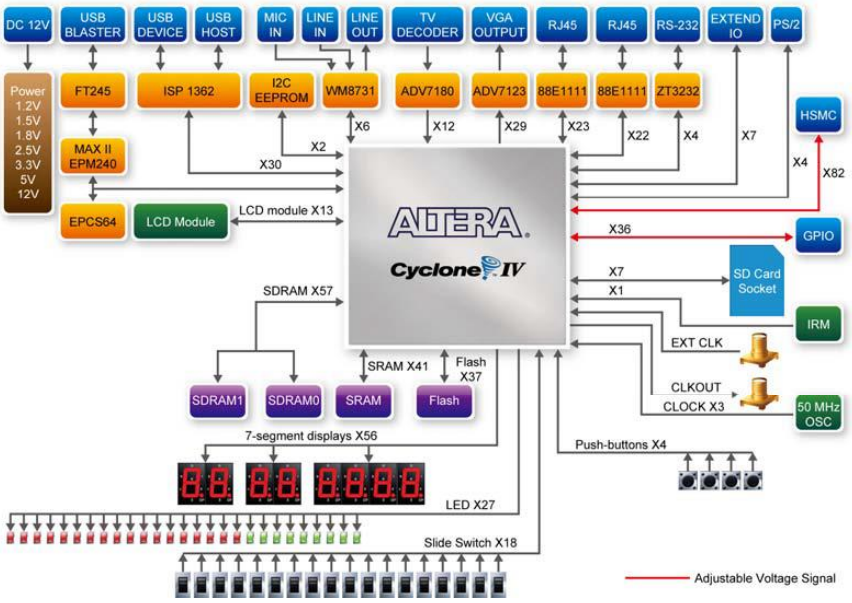


Figura 3.7: Diagrama de blocos do Kit DE2-115 (adaptado de (ALTERA, 2010)).

3.4 Trabalhos correlatos em implementação de algoritmos de fusão sensorial de IMUs em FPGAs

Na Tabela 3.1 comparam-se alguns dos aspectos mais relevantes dos trabalhos correlatos em implementação de algoritmos de fusão sensorial de IMUs em FPGAs, tais como (a) descrição, (b) FPGA utilizado, (c) algoritmos utilizados, (d) interface de comunicação para o sensor (IMU, entre outros), (e) representação de bits do processamento da fusão sensorial e (f) métricas de avaliação.

Tabela 3.1: Revisão bibliográfica dos trabalhos correlatos para determinar *attitude* e *heading* através da fusão sensorial de IMUs embarcados em FPGAs

Ref.	Descrição	FPGA	Algoritmos utilizados	Comunicação	Representação	Métrica de avaliação
(CHAPPELL et al., 2006)	Fusão sensorial para estimação de rolagem, arfagem e guinada de um sistema de segurança automotivo adaptável. Através de IMUs (6 DOF), ACCs e uma câmera ajustados ao veículo para **determinar o desalinhamento deste sistema automotivo**	Xilinx Virtex2 XC2V1000	KF	RS232 e RCA	Ponto fixo de 16 bits	Graus
(ABDELFA-TAH et al., 2011)	Implementação em *hardware* de um sistema em tempo real para **localização da trajetória de um robô** móvel através da fusão sensorial da IMU de 6 DOF e o GPS; enquanto o odômetro é utilizado para medir a distância percorrida	Xilinx Virtex-4	KF	UART	Ponto flutuante de 32 bits	RMSE
(SABATELLI et al., 2012)	Fusão sensorial para localização da orientação, **levando a consideração duas etapas**, uma para *attitude* e outra para *heading* com **uma técnica de quaternios** para uma IMU de 9 DOF	Xilinx Spartan 3A	KF	**I2C (*hardware*)**	-	Graus
(BRUCKNER; SPINDEL-DREIER; BLUME, 2013)	**Comparação de eficiência em *hardware* e em *software* (ARM) de fusão sensorial** para estimação dos ângulos rolagem, arfagem e guinada; com IMU de 9 DOF e **através da ferramenta HLS**	Xilinx Zynq 7020 SoC	KF	RS232	Ponto flutuante	RMSE

(BASHA, 2016)	Estimação da localização usando IMU de 9 DOF e um GPS para sistemas de assistência de mobilidade para deficiência visual em pedestres, **avaliando diferentes representações de bits** na arquitetura projetada	Zynq 7000 SoC	KF	UART	**Ponto fixo de 32 bits, ponto flutuante de 32 e 64 bits**	RMSE
(HAJDU; BRASSAI; SZEKELY, 2017)	Fusão sensorial para estimação da posição angular (rolagem e arfagem), com a implementação de **um filtro passa baixa aplicado ao acelerômetro e um filtro passa alta aplicado ao giroscópio** para uma IMU de 6 DOF	Xilinx Zynq-XC7Z010	**CF**	**I2C (*hardware*)**	Ponto fixo de 24 bits (16.8 bits)	Graus
Este trabalho	Implementação em *hardware* de **uma técnica flexível de fusão sensorial para estimação da posição e orientação de IMUs de 9 DOF** em tempo real	Altera DE2-115	EKF	**I2C (*software*)**	Ponto flutuante de 27 bits	RMSE

Levando a consideração a Tabela 3.1 é valido afirmar que a maioria dos trabalhos correlatos apresentados para a estimação da posição e orientação em FPGA utilizaram o KF como algoritmo de fusão sensorial, devido a suas vantagens em termos de boa precisão e baixo custo de processamento. Além disso, as interfaces de comunicação I2C, RS232 e UART foram as mais utilizadas para a transferência de informação entre os sensores e a arquitetura em FPGA, o qual é devido à flexibilidade de comunicação que têm esses dispositivos (sensores). Em alguns casos o I2C foi implementado como um módulo em *hardware*, externo à arquitetura. Cada trabalho utilizou uma representação em bits (ponto flutuante e ponto fixo) diferente, pois isto depende principalmente da precisão requerida em cada aplicação e dos recursos do FPGA a utilizar. Em relação à métrica de avaliação, a mais utilizada foi o RMSE, no entanto houve trabalhos que fizeram a avaliação da arquitetura em *hardware* através das unidades de medição (graus). Desta Tabela é importante notar que nenhum desses trabalhos embarcou uma técnica de fusão sensorial flexível baseada em IMUs de 9 DOF através do EKF.

Capítulo 4

Arquitetura em FPGA para a fusão sensorial de IMUs

Neste capitulo apresenta-se uma arquitetura de *hardware* implementada em FPGA a fim de realizar a fusão sensorial em tempo real, para Unidades de Medição Inercial (IMUs) de 9 graus de liberdade (DOF) através do algoritmo de Filtro de Kalman Estendido (EKF). O estudo dessa técnica de fusão de sensores é apresentada no capítulo 2. Portanto, para o desenvolvimento desta arquitetura foram utilizados módulos para operações aritméticas (*soma*, *subtração*, *multiplicação*, *divisão* e *raiz quadrada*) e trigonométricas (seno, cosseno, *arcotangente* e *arcotangente2*) em ponto flutuante, assim como módulos para o processamento da fusão sensorial (*master*, *preCordic*, *EKF1*, *EKF2*, *accelerometer*, *magnetometer* e *conversion*) dos sensores (giroscópio, acelerômetro e magnetômetro) que compõem a IMU. Esses dois tipos de módulos são descritos neste capítulo.

4.1 Arquitetura geral de fusão sensorial

Na Figura 4.1 mostra-se o projeto em *hardware* para a fusão sensorial de IMUs. Onde os módulos: (a) processador (Nios II), (b) arquitetura para fusão sensorial, (c) SRAM, (d) JTAG/UART, (e) Timer e (f) GPIO são conectados através do barramento Avalon; no qual os dados são transmitidos com um tamanho de palavra de 32 bits. O Nios II se comunica tanto com a IMU quanto com a arquitetura em *hardware* através do protocolo I2C, o qual foi programado em *software* e conectado aos pinos do GPIO da placa, sendo que o Nios II atua como mestre e a IMU de 9 DOF (MPU9250) como escravo. Enquanto o SRAM realiza a função de armazenamento de dados, o timer faz a temporização para a leitura da IMU e o JTAG/UART leva a cabo comunicação da placa DE2-115 com o computador (a fim de programar o FPGA e o Nios II).

O processamento inicial da arquitetura consiste em realizar a leitura de cada um dos três sensores (giroscópio, acelerômetro e magnetômetro) da IMU de 9 DOF, através do Nios II. Posteriormente esse processador envia ditas medições à arquitetura dedicada (mapeada no FPGA), com o intuito de realizar a estimação dos ângulos de rolagem, arfagem e guinada em tempo real. Depois disso, esses ângulos de Euler estimados são requeridos pelo processador, a fim de fazer a leitura dos mesmos através do *software* Eclipse.

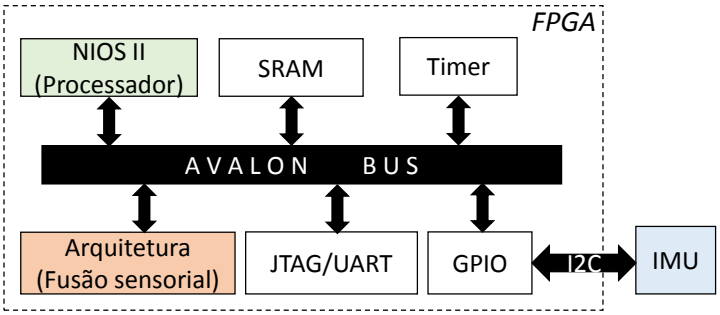


Figura 4.1: Diagrama do projeto em FPGA. A transmissão de dados no barramento *Avalon* é de 32 bits.

A arquitetura de fusão sensorial tem como objetivo principal estimar a posição (rolagem e arfagem) e a orientação (guinada) em tempo real. Isto é, o tempo de processamento na arquitetura deve atingir um tempo menor ao tempo de amostragem da IMU de 9 DOF ($7ms$). A arquitetura desenvolvida (vide Figura 4.2) opera a 50MHz e faz uso de operadores em representação em ponto flutuante de 27 bits. A comunicação e sincronização entre a IMU e o FPGA foi realizada no Nios II. A interface de entrada e saída permite escolher a função a ser realizada, já seja a escrita ou a leitura através dos sinais 'avs_write' e 'avs_read', respectivamente. Da mesma maneira, levam-se em consideração os dados de entrada ($avs\_writedata$), os dados de saída ($avs\_readdata$) e o endereço dos dados ($avs\_address$).

Os módulos operacionais nessa arquitetura são aqueles que realizam os cálculos de: (a) *multiplicação*, (b) *soma*, (c) *subtração*, (d) *divisão*, (e) *seno* e *cosseno*, (f) *arcotangente* e (g) *raiz quadrada*. Na arquitetura desenvolvida para fusão sensorial, o módulo *Multipliers* contem internamente 12 multiplicadores, o módulo *Adders* e o módulo *Subtractors* têm 6 componentes cada um, enquanto o módulo *Dividers* contem somente 2 divisores. No entanto, os outros módulos (*cordic1*, *cordic2* e *sqrt*) têm um único componente interno. O módulo *cordic1* (*seno* e *cosseno*) é usado somente pelo módulo *preCordic*, enquanto os que acessam aos módulos das operações aritméticas são: (a) *EKF1*, (b) *EKF2*, (c) *accelerometer*, (d) *magnetometer* e (e) *conversion* através do multiplexador 1 (Mux 1). Adicionalmente, o multiplexador 2 (Mux 2) comuta o uso do *cordic2* (arcotangente) para os módulos do *accelerometer* e *magnetometer*. Cabe ressaltar que o módulo *sqrt* é usado apenas pelo módulo *accelerometer*.

A Figura 4.3 mostra o *scheduling* da arquitetura em *hardware* para a fusão sensorial de IMUs, onde a sequência de estados para os módulos é observada. Tal sequência é feita através do gerenciamento do módulo *master*, o qual é o controlador dos sinais de cada um dos módulos instanciados na arquitetura. O processamento dos ângulos de Euler inicia a partir do sinal '*start*' e conclui com o sinal '*ready*'. Portanto, depois do sinal de inicio a primeira rodada ($k = 1$) começa a sua execução, onde o primeiro módulo a ser executado é o *preCordic*. Isto devido a que inicialmente devem ser calculados e armazenados em registradores os valores trigonométricos (*seno* e *cosseno*) dos valores iniciais para os ângulos de Euler. Seguidamente, o módulo *accelerometer* é ativado a fim de calcular os ângulos de rolagem e arfagem tendo em conta as Equações 2.8 e 2.9, as quais relacionam as medições do acelerômetro. Depois disso, o módulo *EKF1* é o seguinte em ser ativado, no qual é realizada a primeira fusão sensorial para estimar a posição (rolagem e arfagem), através do giroscópio e acelerômetro (vide seção 2.3). Em seguida, o módulo *preCordic* executa-se novamente com o intuito de calcular os valores de *seno* e *cosseno* dos ângulos estimados pelo *EKF1*.

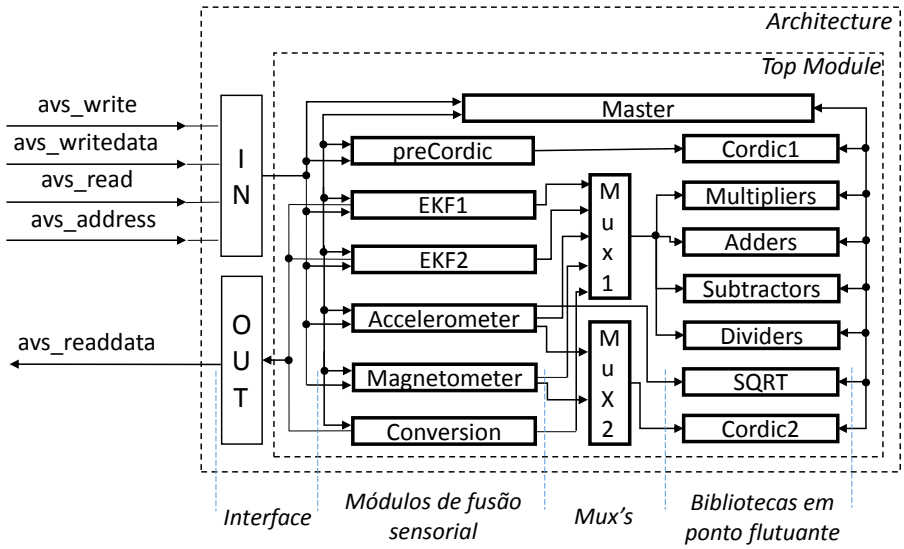


Figura 4.2: Arquitetura de fusão sensorial de uma IMU de 9 DOF com os EKFs. Está composta por (a) interface de dados (*input/output*), (b) módulos de fusão sensorial, (c) multiplexadores (MUX1 e MUX2), (d) bibliotecas em ponto flutuante de 27 bits (*cordic1*, *multipliers*, *adders*, *subtractors*, *dividers*, SQRT, *cordic2*).

Posteriormente, o módulo *master* ativa o módulo *magnetometer* com o objetivo de determinar o ângulo de guinada a partir das medições do magnetômetro e das Equações 2.10, 2.11 e 2.12. Seguidamente, é ativado o módulo *EKF2*, onde o processamento da segunda fusão sensorial entre o giroscópio e acelerômetro é feito, a fim de estimar a orientação (guinada) (vide seção 2.3). Finalmente, o ultimo módulo a ser ativado é o *conversion*, no qual se faz a conversão de unidades de radianos a graus, para cada um dos três ângulos de Euler estimados tanto pelo módulo *EKF1* quanto pelo módulo *EKF2*. Depois disso, o módulo *master* envia um sinal de encerramento, indicando o fim da execução para dita rodada ($k = 1$). As seguintes rodadas começam com informações atualizadas de cada um dos sensores (giroscópio, acelerômetro e magnetômetro), sendo que desde a segunda rodada ($k = 2$) até a enésima rodada ($k = n$), a sequência da arquitetura é similar à primeira rodada. Porém, com a principal diferença que nesses casos a arquitetura ignora o passo inicial da execução do módulo *preCordic* após do sinal '*start*', devido que os cálculos do *seno* e *cosseno* dos últimos valores estimados dos ângulos de rolagem e arfagem foram calculados pelo módulo *preCordic* na iteração anterior $(k - 1)$.

4.2 Módulos para operações em ponto flutuante

Tendo em conta que os cálculos em aplicações tanto em *software* quanto em *hardware* requerem de alta precisão, as operações devem ser feitas em representação de ponto flutuante, definida pela norma IEEE 754 para diferentes tamanhos de palavras (32 bits (tipo float) e 64 bits (tipo double)) (IEEE, 2008). Essa representação divide-se em três partes: (a) sinal, (b) exponente e (c) mantissa (parte fracionaria), assim como mostrado na Figura 4.4. Um fator importante a ter em conta no momento de utilizar essa representação em *hardware* é o custo de consumo de recursos no FPGA. Neste caso, para o processamento desta arquitetura utilizou-se uma biblioteca parametrizável de 27 bits tanto para as operações aritméticas

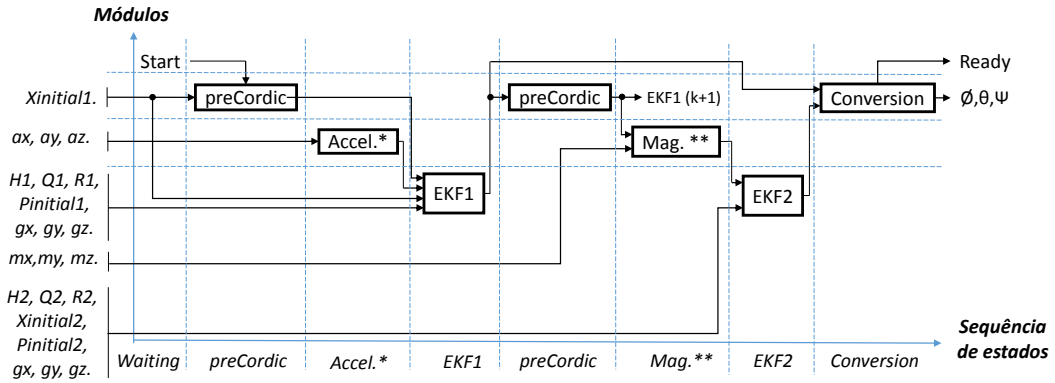


Figura 4.3: Scheduling da arquitetura. Para dois casos: (a) primeira rodada ($k = 1$) e (b) seguintes rodadas ($k = 2, 3, ..., n$). * *Accelerometer*. ** *Magnetometer*.

(SÁNCHEZ et al., 2009) quanto para as trigonométricas (MUÑOZ et al., 2010), isto com o intuito de utilizar a menor quantidade de recursos (multiplicadores embarcados e elementos lógicos) possíveis do FPGA, atingindo uma precisão adequada para este tipo de aplicação (MUÑOZ, 2012).

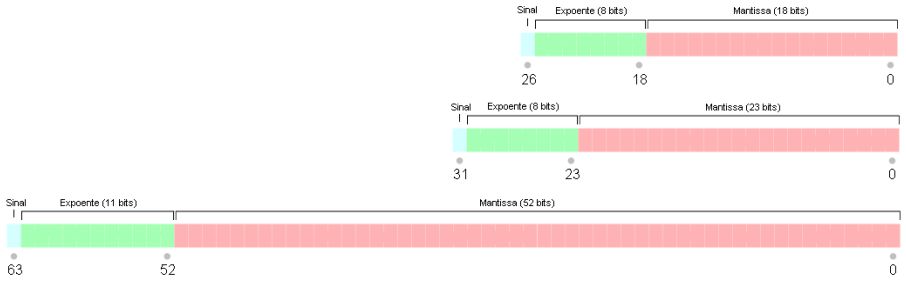


Figura 4.4: Representação em ponto flutuante (adaptado de (RAFAEL, 2014)).

4.2.1 *Adder/Subtractor*

As operações em ponto flutuante de soma e subtração (MUÑOZ, 2012) são obtidas através das seguintes etapas: (1) Separar o sinal, o exponente e a mantissa de cada uma das duas entradas, assim como verificar se as entradas são zero, infinitas ou têm uma representação inválida ao padrão IEEE 754, e posteriormente adicionar o bit oculto às mantissas. (2) Comparar as duas entradas, tendo em conta que o número de bits da mantissa do menor número deve-se deslocar à direita, dependente da diferença do expoente, para assim somar ou subtrair as mantissas atuais. (3) Deslocar à esquerda a mantissa obtida até que o MSB seja 1, onde para cada deslocamento o expoente diminui em 1. (4) Concatenar o sinal, o exponente e a mantissa dos resultados finais (SÁNCHEZ et al., 2009).

4.2.2 *Multiplier*

No caso da multiplicação em representação em ponto flutuante (MUÑOZ, 2012), os passos para realizar esta operação aritmética são os seguintes: (1) Separar o sinal, o exponente e a mantissa de cada uma das

duas entradas, assim como verificar se as entradas são zero, infinitas ou têm uma representação inválida ao padrão IEEE 754, e posteriormente adicionar o bit oculto às mantissas. (2) Multiplicar as mantissas, assim como determinar o sinal do resultado dessa multiplicação, somar os expoentes e subtrair o "*bias*". (3) No caso que o MSB fosse 1 no resultado da multiplicação das mantissas, não é necessário fazer a normalização, isto é, a mantissa é deslocada à esquerda até que o MSB seja 1, porém para cada deslocamento o expoente atual diminui em 1. (4) Concatenar o sinal, o exponente e a mantissa dos resultados finais (SÁNCHEZ et al., 2009).

4.2.3 *Divider*

Para executar a operação em ponto flutuante de divisão (MUÑOZ, 2012) é necessário: (1) X e Y devem ser números reais representados com o padrão IEEE 754, onde X representa o dividendo e Y o divisor. (2) Separar o sinal, o expoente e a mantissa tanto de X quanto de Y, adicionando o bit oculto à mantissa, e verificar se as entradas são zero ou inválidas. (3) Calcular os resultados da mantissa usando o algoritmo *Newton-Raphson* para a divisão; além de calcular o valor do expoente $(expoente = expoente(X) - expoente(Y) + bias)$ e determinar o sinal desse resultado. (4) Concatenar o sinal, o exponente e a mantissa dos resultados finais (SÁNCHEZ et al., 2009).

4.2.4 *sqrt* (raiz quadrada)

No cálculo da raiz quadrada em ponto flutuante (MUÑOZ, 2012) deve ser levado em consideração: (1) X deve ser um número real representado com o padrão IEEE 754. (2) Separar o sinal, o expoente e a mantissa de X, adicionando o bit oculto à mantissa e verificar se a entrada é negativa, zero ou inválida, além disso, se o expoente de X é par, então deve-se multiplicar a mantissa por 2. (3) Calcular a mantissa usando o algoritmo *Goldschmidt* para a raiz quadrada, da mesma forma deve-se calcular o resultado do expoente $(expoente = expoente((X + Bias)/2))$. (4) Concatenar o sinal, o exponente e a mantissa dos resultados finais e remover o bit oculto da mantissa resultante (SÁNCHEZ et al., 2009).

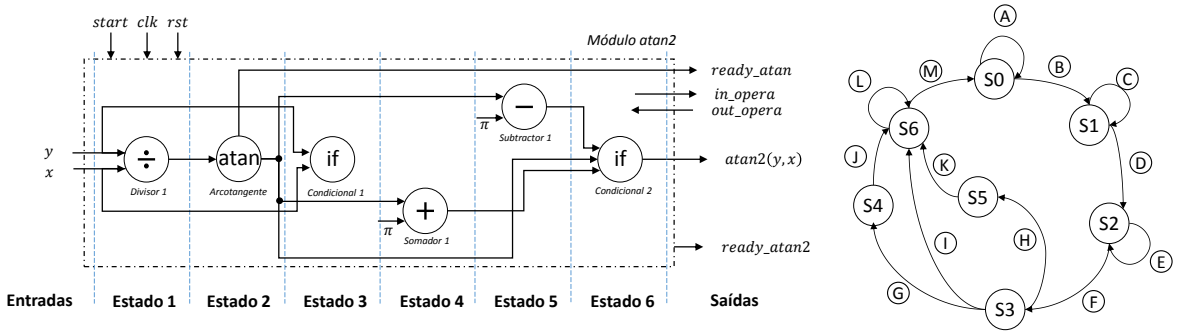
4.2.5 *Cordic1 e Cordic2* (*seno*/*cosseno* e *atan*)

O cálculo das funções do seno/cosseno e arcotangente é realizado através dos módulos *cordic1* e *cordic2*, respectivamente. Esses dois módulos estão baseados na mesma arquitetura, somente com mudanças nas condições iniciais. A arquitetura é composta por quatro unidades: (a) unidade de redução de argumentos, (b) unidade de FSM, (c) unidade de micro-rotação e (d) unidade de normalização. A unidade de redução de argumentos e a unidade de normalização consideram um quadrante para os cálculos do seno e cosseno. Enquanto a unidade de FSM seleciona as condições iniciais para as variáveis X, Y e Z, as quais dependem da operação desejada (*seno*/*cosseno* ou *atan*). A unidade de micro-rotação toma conta das operações e concatenações para o cálculo respectivo das funcoes *seno*/*cosseno* e *arcotangente* (MUÑOZ et al., 2010).

4.2.6 *atan2*

O projeto em *hardware* da função *atan2* foi totalmente desenvolvido neste trabalho. Esta função retorna valores no intervalo $(0, 2\pi)$, ao contrario da função *atan* que retorna valores só de $(-\pi/2, \pi/2)$. Esta função é utilizada para calcular os ângulos de rolagem e guinada. A Figura 4.5 apresenta a FSM deste módulo com sete estados. A lógica da FSM é representada na Figura 4.5(b), sendo que as condições para as transições de estados são definidas pela Equação 4.1. Além disso, as operações envolvidas neste módulo são: (a) uma divisão, (b) uma arcotangente, (c) uma soma e (d) uma subtração (vide Figura 4.5(a)).

$$atan2(y,x) = \begin{cases} atan(y/x) & se\ x > 0 \\ atan(y/x) + \pi & se\ x < 0\ e\ y \geq 0 \\ atan(y/x) - \pi & se\ x < 0\ e\ y < 0 \\ \pi/2 & se\ x = 0\ e\ y > 0 \\ -\pi/2 & se\ x = 0\ e\ y < 0 \\ indefinido & se\ x = 0\ e\ y = 0 \end{cases} \tag{4.1}$$



(a) Operadores do módulo *atan2*.

(b) FSM do módulo *atan2*.

Figura 4.5: Operadores e FSM do módulo *atan2*

O módulo *atan2* é ativado através de um sinal de *start* enviado desde o módulo master. Portanto, o estado inicial $S0$ abrange as condições iniciais, enquanto nos estados $S1$ e $S2$ são realizados os cálculos de divisão e arcotangente, respectivamente. No estado $S3$ é feita a seleção do quadrante onde as entradas x e y estão localizadas no plano cartesiano, o qual determina a seguinte operação a realizar; isto é, se ditas entradas estão localizadas em um dos quadrantes 1 e 4, a saída é igual à obtida no estado $S2$, devido que o valor da função *arcotangente2* nesses casos é igual à função *arcotangente*. No entanto, se as entradas x e y localizam-se no quadrante 2 ou 3, ao valor prévio de *arcotangente* (estado $S2$) deve-se somar (estado $S4$) ou subtrair (estado $S5$) o valor de π, respectivamente. Finalmente, no estado $S6$ a saída valida da tangente inversa para os quatro quadrantes (*atan2*) é definida e indicada pelo sinal *ready_atan2*. As condições de transição de estados são apresentadas na Tabela 4.1.

Tabela 4.1: Condições para a transições de estados do módulo *atan2*. * Sem condição

Abreviação	Desde	Até	Condição	Saída
A	S0	S0	(a) start = '1' e ay = 0 (b) start = '1' e az = 0 e ay >0 (c) start = '1' e az = 0 e ay <0 (d) start = '0'	atan2 = 0 atan2 = $\pi/2$ atan2 = $-\pi/2$ atan2 = 0
B	S0	S1	start = '1' e az $\neq$ 0 e ay $\neq$ 0	atan2 = 0
C	S1	S1	ready_div1 = '0'	atan2 = 0
D	S1	S2	ready_div1 = '1'	atan2 = 0
E	S2	S2	ready_atan = '0'	atan2 = 0
F	S2	S3	ready_atan = '1'	atan2 = 0
G	S3	S4	az < 0 e ay $\geq$ 0	atan2 = 0
H	S3	S5	az < 0 e ay < 0	atan2 = 0
I	S3	S6	az >0	atan2 = 0
J	S4	S6	*	atan2 = 0
K	S5	S6	*	atan2 = 0
L	S6	S6	s_ready_state3 = '0' e ready_add1 = '0' e ready_sub1 = '0'	atan2 = 0
M	S6	S0	(a) s_ready_state3 = '1' (b) ready_add1 = '1' (c) ready_sub1 = '1'	atan2 = out_atan atan2 = out_add1 atan2 = out_sub1

4.3 Módulos de fusão sensorial

Todos os módulos de fusão sensorial foram desenvolvidos através de Máquinas de Estado Finito (FSM) do tipo *Mealy*, devido que o valor de saída de cada estado depende tanto do valor de entrada quanto do estado atual. Além disso, para o cálculo de cada ângulo de Euler é necessário ter a realimentação deles no passo anterior de tempo $(k-1)$. Esses módulos estão conectados aos módulos operacionais (módulos aritméticos e trigonométricos) através do *mux1* e *mux2*, os quais comutam as entradas dependendo do estado e de um sinal de bandeira (*flag*) enviado pelo módulo *master*.

4.3.1 *Master*

Na Figura 4.6 apresenta-se a lógica da FSM do módulo *master*, no qual estão envolvidos dez estados; sendo que as condições para as transições de estados observam-se na Tabela 4.2. Este é o módulo principal da arquitetura apresentada no capítulo 4, já que toma conta do controle de processamento da mesma. Isto é feito por meio dos sinais *starts* e *readys* de cada um dos módulos instanciados dentro do *Top_module*, assim como do sinal *flag* (para o caso dos multiplexadores *mux1* e *mux2*) a fim de comutar as entradas dos módulos operacionais. Da mesma forma, os estados do módulo *preCordic* são controlados por um sinal *flag_preCordic* enviado pelo *master*. Adicionalmente, o módulo *master* tem conhecimento sobre o número de rodadas da arquitetura, através do sinal *cont* enviado pelo módulo *EKF1*.

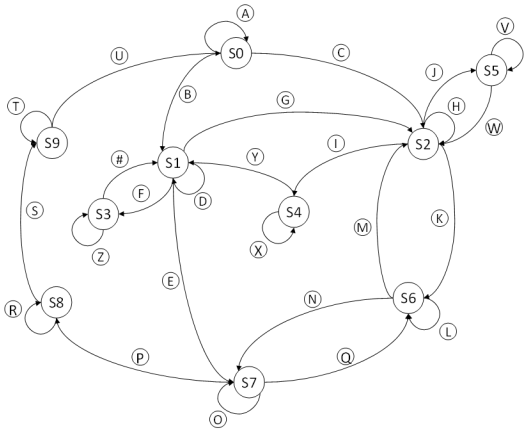


Figura 4.6: FSM do módulo *master*.

Tabela 4.2: Lógica da FSM do módulo *master*. S0 (*Waiting*), S1 (*preCordic*), S2 (*accelerometer*), S3 (*cordic1*(*sin*/*cos*)), S4 (*EKF1*), S5 (*sqrt*), S6 (*cordic2*(*atan*)), S7 (*magnetometer*), S8 (*EKF2*), S9 (*conversion*).

Abreviação	Desde	Até	Condição
A	S0	S0	start = ’0’
B	S0	S1	start = ’1’ e cont <1
C	S0	S2	start = ’1’ e cont $\geq$ 1
D	S1	S1	ready_preCordic = ’0’ e flag_preCordic $\geq$ 2
E	S1	S7	ready_preCordic = ’1’ e cont $\geq$ 1
F	S1	S3	ready_preCordic = ’0’ e flag_preCordic < 2
G	S1	S2	ready_preCordic = ’1’ e cont < 1
H	S2	S2	ready_acelerometro = ’0’ e start_atan_accel = ’0’ e start_sqrt_accel = ’0’
I	S2	S4	ready_acelerometro = ’1’ e start_atan_accel = ’0’ e start_sqrt_accel = ’0’
J	S2	S5	ready_acelerometro = ’0’ e start_atan_accel = ’0’ e start_sqrt_accel = ’1’
K	S2	S6	ready_acelerometro = ’0’ e start_atan_accel = ’1’ e start_sqrt_accel = ’0’
L	S6	S6	(a) s_flag_state_prev = 1 e ready_cordic_atan = ’0’ (b) s_flag_state_prev = 2 e ready_cordic_atan = ’0’ (c) s_flag_state_prev = 0
M	S6	S2	s_flag_state_prev = 1 e ready_cordic_atan = ’1’
N	S6	S7	s_flag_state_prev = 2 e ready_cordic_atan = ’1’
O	S7	S7	ready_magnetometro = ’0’ e start_atan_magn = ’0’
P	S7	S8	ready_magnetometro = ’1’ e start_atan_magn = ’0’
Q	S7	S6	ready_magnetometro = ’0’ e start_atan_magn = ’1’
R	S8	S8	ready_EKF2 = ’0’

S	S8	S9	ready_EKF2 = ’1’
T	S9	S9	ready_conversion = ’0’
U	S9	S0	ready_conversion = ’1’
V	S5	S5	ready_sqrt = ’0’
W	S5	S2	ready_sqrt = ’1’
X	S4	S4	ready_EKF1 = ’0’
Y	S4	S1	ready_EKF1 = ’1’
Z	S3	S3	ready_cordic_sin_cos = ’0’
#	S3	S1	ready_cordic_sin_cos = ’1’

4.3.2 *PreCordic*

O módulo *preCordic* foi desenvolvido com o intuito de armazenar em registradores os valores de *seno* e *cosseno* dos ângulos de rolagem e arfagem, objetivando poupar espaço no FPGA; pois neste caso, a arquitetura tem só um módulo *cordic1* (*seno*/*cosseno*). Os dados armazenados nos registradores são enviados posteriormente ao módulo *EKF1*. A FSM deste módulo é simples e contem três estados (vide Figura 4.7). As condições de transição de estados são iguais, devido que o *master* envia um sinal de *start* quando cada cálculo do *cordic1* é feito. Porém o direcionamento na FSM está definido pelo sinal *flag_preCordic*, o qual é a referência do estado atual deste módulo para o *master*.

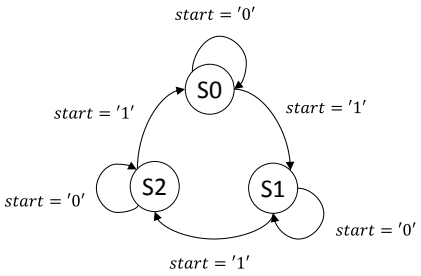


Figura 4.7: FSM do módulo *preCordic*.

4.3.3 *EKF1*

O módulo *EKF1* é mostrado na Figura 4.8, sendo o módulo que leva a cabo a fusão sensorial para estimar a posição (rolagem e arfagem) com as informações do giroscópio e do acelerômetro da IMU de 9 DOF (MPU9250). Este módulo tem internamente 12 sub-módulos, entre os quais encontram-se: (1) *módulo A*, (2) *módulo B*, (3) *módulo 1*, (4) *módulo 2*, (5) *módulo 3*, (6) *módulo 4*, (7) *módulo 5*, (8) *módulo 6*, (9) *mux1*, (10) *mux2*, (11) *cont1* e (12) *cont2*. Os *módulos A* e *B* são executados em paralelo, além disso os mesmos realizam cálculos prévios de multiplicações e divisões. Enquanto cada um dos *módulos 1*, *2*, *3*, *4*, *5* e *6* realiza um passo específico do EKF, onde os *módulos 1* e *2*, *5* e *6* executam-se paralelamente. Os multiplexadores *mux1* e *mux2*, definem as entradas do vetor de estados (x_k) e da covariância (P_k), respetivamente, através do número de rodadas determinadas pelos outros dois módulos restantes (*cont1* e

cont2). Similarmente aos outros módulos instanciados no *Top_module*, o módulo *master* envia o sinal *start* e deteta o sinal *ready_EKF1* deste módulo.

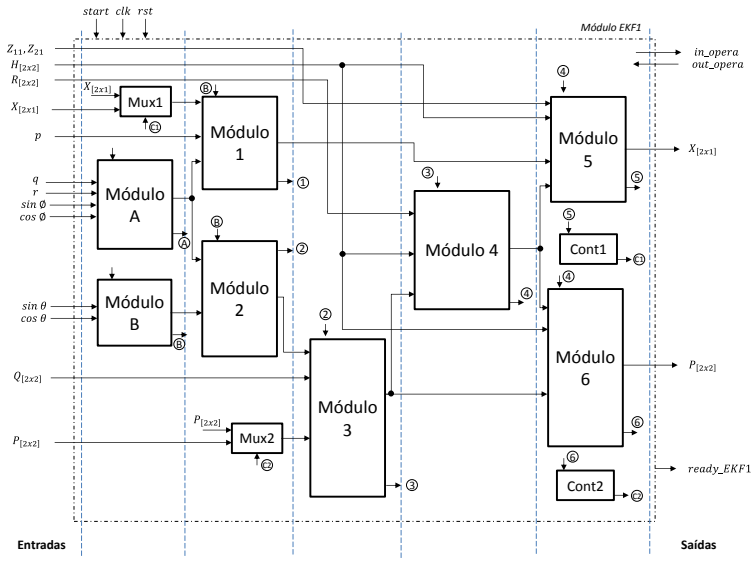
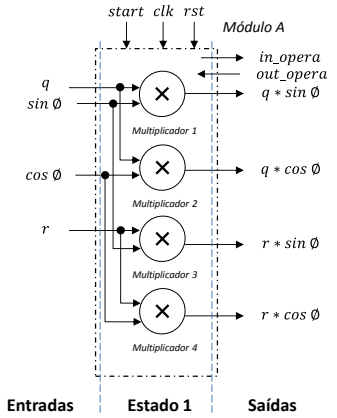
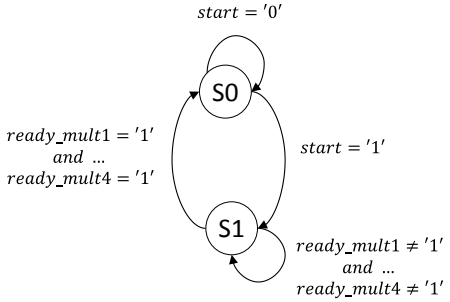


Figura 4.8: Módulo *EKF1*.

O *módulo A* tem quatro operações de multiplicação envolvidas internamente, como pode ser observado na Figura 4.9. A partir das entradas q, r, $\sin\phi$ e $\cos\phi$ são determinados os produtos $q \cdot \sin\phi$, $q \cdot \cos\phi$, $r \cdot \sin\phi$ e $r \cdot \cos\phi$ (vide Figura 4.9(a)), sendo que cada uma dessas saídas são enviadas ao *módulo 1*. A lógica de processamento da FSM deste módulo é mostrada na Figura 4.9(b), a qual tem unicamente dois estados.



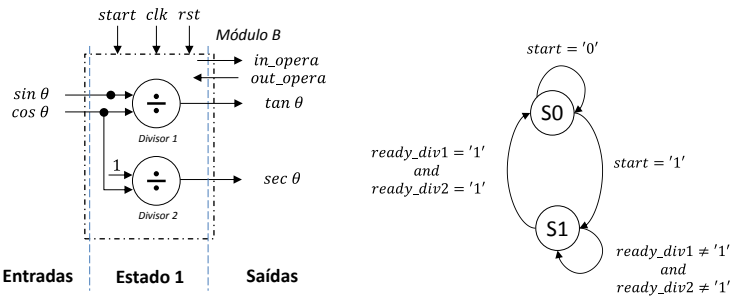
(a) Operadores do *módulo A* do *EKF1*.



(b) FSM do *módulo A* do *EKF1*.

Figura 4.9: Operadores e FSM do *módulo A* do *EKF1*.

O *módulo B* é mostrado na Figura 4.10. A FSM junto com as condições de transição de estados deste módulo são detalhadas na Figura 4.10(b). Este módulo realiza os cálculos de $\tan\theta$ e $\sec\theta$ com respeito às entradas de $\sin\theta$ e $\cos\theta$, através de duas operações de divisão (vide Figura 4.10(a)). Adicionalmente, as saídas deste módulo são aproveitadas no *módulo 2*.

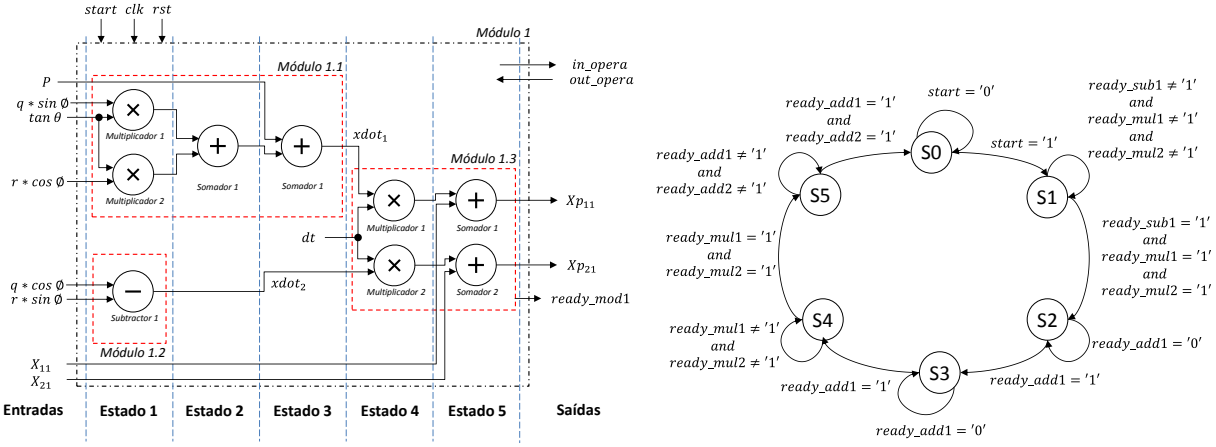


(a) Operadores do *módulo B* do *EKF1*.

(b) FSM do *módulo B* do *EKF1*.

Figura 4.10: Operadores e FSM do *módulo B* do *EKF1*.

O *módulo 1* do *EKF1* (vide Figura 4.11) realiza a estimativa das variáveis de estado (rolagem e arfagem) na etapa a priori ($x_p = x + xdot \cdot dt$), sendo esse o primeiro passo do EKF (vide Algoritmo 1). Este módulo está composto de uma FSM de seis estados detalhada na Figura 4.11(b), onde também observam-se as condições para as transições de estado. Além disso, o *módulo 1* tem três sub-módulos como pode ser visto na Figura 4.11(a). O *sub-módulo 1.1* está composto de dois multiplicadores e dois somadores, sendo que este módulo determina o valor de $xdot_1 = p + q \cdot \sin\phi \cdot \tan\theta + r \cdot \cos\phi \cdot \tan\theta$, assim como o *módulo 1.2* tem só um módulo de subtração e calcula o valor de $xdot_2 = q \cdot \cos\theta - r \cdot \sin\phi$. Enquanto o *módulo 1.3* realiza as operações de multiplicação ($xdot \cdot dt$) e soma ($x + xdot \cdot dt$), através de dois multiplicadores e dois somadores.



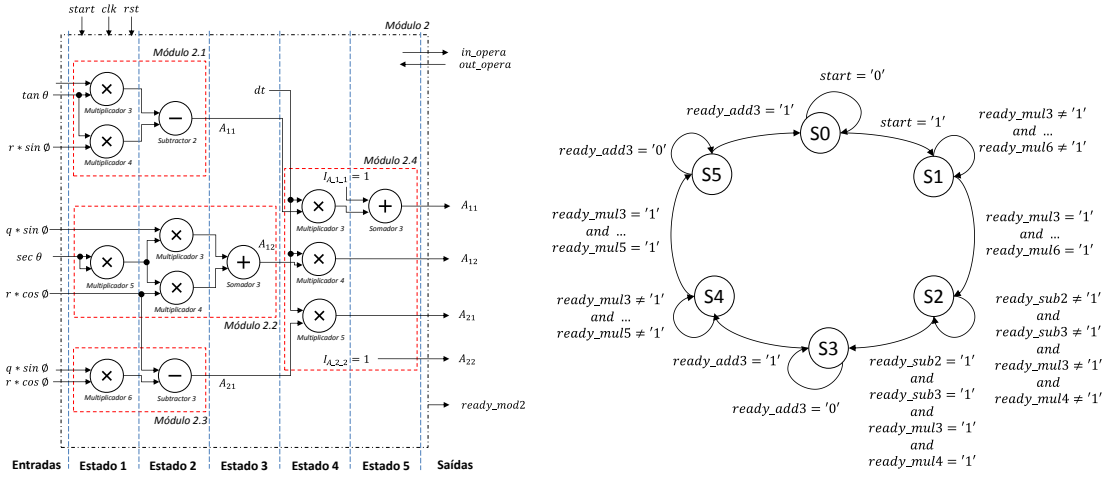
(a) Operadores do *módulo 1* do *EKF1*.

(b) FSM do *módulo 1* do *EKF1*.

Figura 4.11: Operadores e FSM do *módulo 1* do *EKF1*.

Na Figura 4.12 apresenta-se o *módulo 2* do *EKF1*, o qual determina o valor da matriz Jacobiana descrita por $A = I_A + A \cdot dt$, através de quatro sub-módulos (vide Figura 4.12(a)) e de uma FSM de seis estados. A Figura 4.12(b) mostra as condições de transição de estados. No *sub-módulo 2.1* determina-se o cálculo do Jacobiano $A_{11} = q \cdot \cos\phi \cdot \tan\theta - r \cdot \sin\phi \cdot \tan\theta$, utilizando dois multiplicadores e um somador. O *módulo 2.2* realiza o cálculo do Jacobiano $A_{12} = q \cdot \sin\phi \cdot \sec^2\theta + r \cdot \cos\phi \sec^2\theta$ tendo em conta três multiplicadores e um somador. Adicionalmente, o *módulo 2.3* determina o valor do Jacobiano $A_{21} = -q \cdot \sin\phi - r \cdot \cos\phi$ com um multiplicador e um operador de subtração. Enquanto o Jacobiano $A_{22} = 0$. Portanto, o módulo

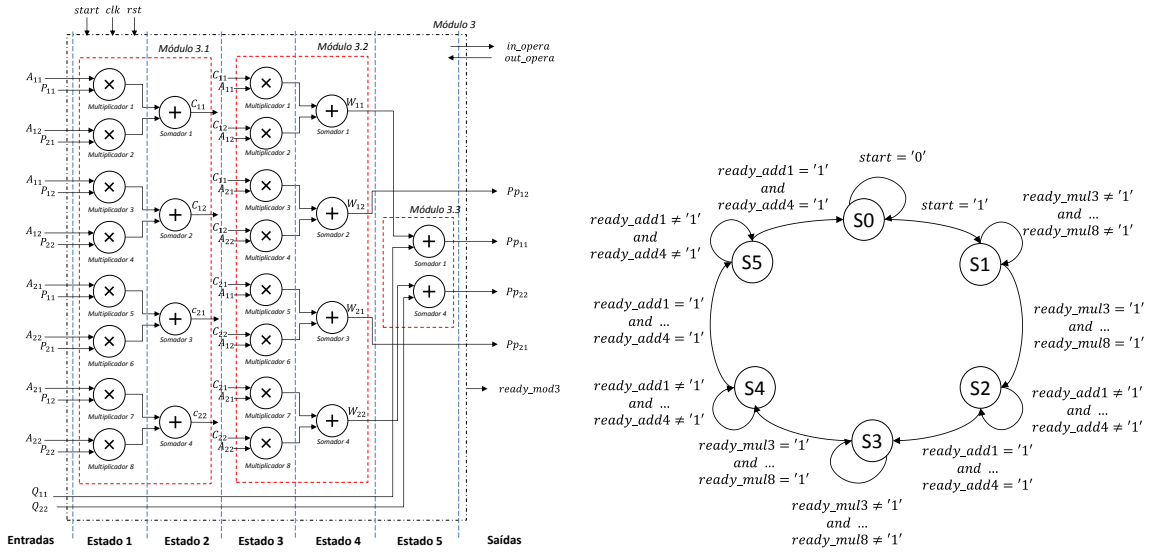
2.4 realiza o cálculo de cada Jacobiano levando em consideração três multiplicadores ($A \cdot dt$) e um somador ($I_A + A \cdot dt$).



(a) Operadores do *módulo 2* do *EKF1*. (b) FSM do *módulo 2* do *EKF1*.

Figura 4.12: Operadores e FSM do *módulo 2* do *EKF1*.

No *módulo 3* do *EKF1* (vide Figura 4.13) calcula-se o valor predito do erro da covariância ($P_p = A \cdot P \cdot A^T + Q$) na etapa a priori do EKF; sendo esse o segundo passo do EKF (vide Algoritmo 1). Este módulo está baseado em uma FSM mostrada na Figura 4.13(b) de seis estados com 16 operações de multiplicação e 10 de soma (vide Figura 4.13(a)). Além disso, o *módulo 3* compõe-se de três sub-módulos, onde os *sub-módulos 3.1* e *3.2* realizam a multiplicação de matrizes de tamanho 2x2, calculando $C = A \cdot P$ e $W = C \cdot A^T$ respectivamente. Seguidamente, o *módulo 3.3* realiza a operação de soma ($W + Q$).

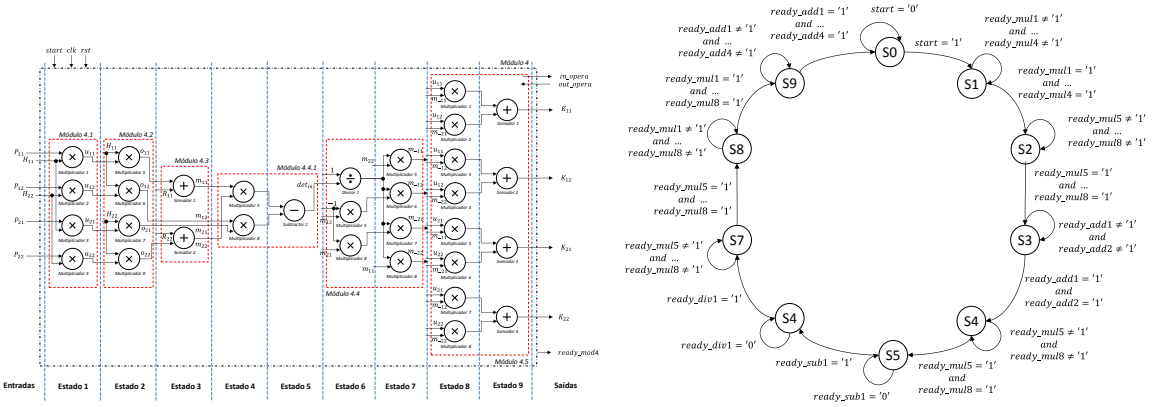


(a) Operadores do *módulo 3* do *EKF1*. (b) FSM do *módulo 3* do *EKF1*.

Figura 4.13: Operadores e FSM do *módulo 3* do *EKF1*.

O *módulo 4* (vide Figura 4.14) do *EKF1* foi desenvolvido para calcular o terceiro passo do EKF (vide Algoritmo 1), o qual é o ganho de Kalman ($K = P_p \cdot H^T \cdot (H \cdot P_p \cdot H^T + R)^{-1}$). A lógica de processamento

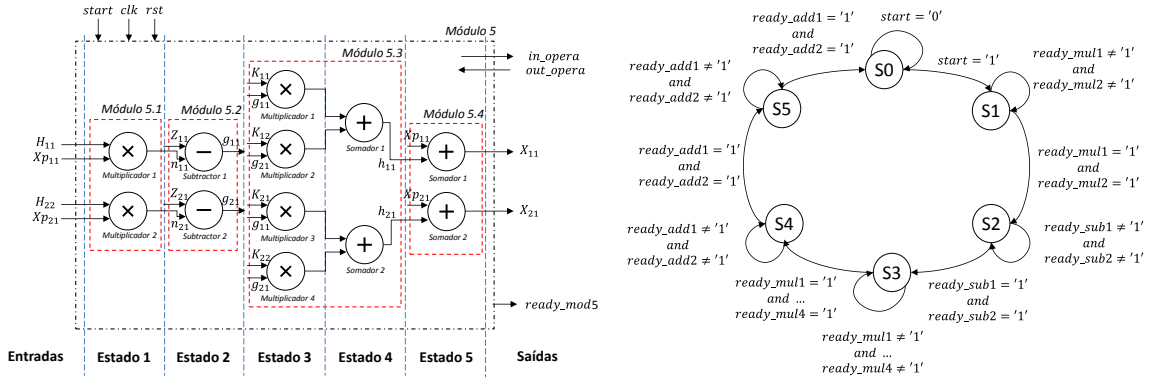
deste módulo está baseada em uma FSM de dez estados (vide Figura 4.14(b)). Na figura 4.14(a) observam-se os seis sub-módulos, onde os *módulos 4.1* e *4.2* têm quatro operações de multiplicação para determinar $u = P_p \cdot H^T$ e $o = H \cdot u$, respectivamente. O *módulo 4.3* realiza duas somas ($m = o + R$). O *módulo 4.4* (com seis multiplicações e uma divisão) determina a matriz inversa da matriz m. Porém para isto é necessário calcular o determinante de m através de dois operações de multiplicação e uma de subtração, assim como é mostrado no *módulo 4.4.1*. Adicionalmente, o *módulo 4.5* realiza a multiplicação de matrizes de tamanho 2x2 ($u \cdot m$), levando em consideração oito multiplicações e quatro somas.



(a) Operadores do *módulo 4* do *EKF1*. (b) FSM do *módulo 4* do *EKF1*.

Figura 4.14: Operadores e FSM do *módulo 4* do *EKF1*.

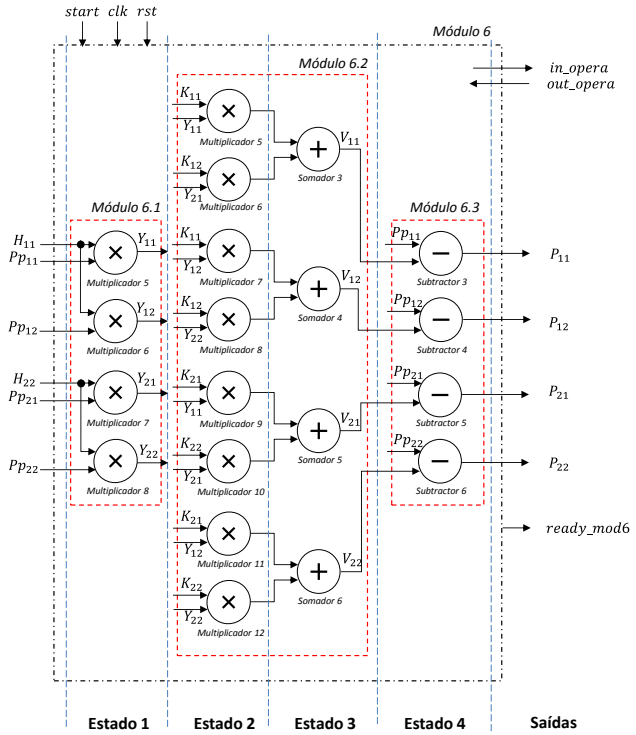
O *módulo 5* do *EKF1* calcula a estimativa das variáveis de estado ($x = x_p + K \cdot (Z - H \cdot x_p)$) na etapa de correção do EKF, sendo o quarto passo do filtro EKF (vide Algoritmo 1). Este módulo é mostrado na Figura 4.15. Similarmente aos outros módulos do *EKF1*, o *módulo 5* está baseado em uma FSM detalhada com as transições de estado na Figura 4.15(b). Além disso, o mesmo está composto por quatro sub-módulos (vide Figura 4.15(a)), onde o *módulo 5.1* realiza duas operações de multiplicação para calcular $n = H \cdot x_p$. Posteriormente são realizadas duas subtrações no *módulo 5.2* ($g = Z - n$). Enquanto o *módulo 5.3* realiza quatro operações de multiplicação e dois de soma, a fim de calcular $h = K \cdot (g)$. Adicionalmente, no *módulo 5.4* são feitas duas somas ($x_p + h$).



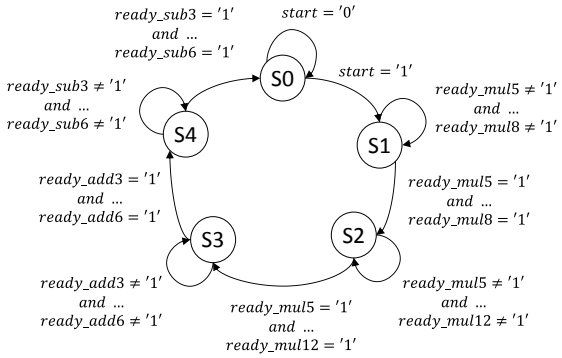
(a) Operadores do *módulo 5* do *EKF1*. (b) FSM do *módulo 5* do *EKF1*.

Figura 4.15: Operadores e FSM do *módulo 5* do *EKF1*.

A Figura 4.16 apresenta o *módulo 6* do *EKF1*. As Figuras 4.16(a) e 4.16(b) mostram as operações e a FSM deste módulo, respectivamente. Além disso, neste módulo determina-se o erro de covariância ($P = P_p - K \cdot H \cdot P_p$) da etapa de correção do EKF, o qual é o último (quinto) passo do EKF (vide Algoritmo 1). Sendo que o *módulo 6* contem internamente 3 sub-módulos. O *módulo 6.1* realiza quatro operações de multiplicação ($Y = H \cdot P_p$). O *módulo 6.2* leva a cabo quatro multiplicações de matrizes 2x2 para calcular $V = K \cdot Y$. Finalmente, no *módulo 6.3* determina-se $P_p - V$ através de quatro subtrações.



(a) Operadores do *módulo 6* do *EKF1*.



(b) FSM do *módulo 6* do *EKF1*.

Figura 4.16: Operadores e FSM do *módulo 6* do *EKF1*.

4.3.4 *EKF2*

No que diz respeito ao *módulo EKF2*, este realiza a fusão sensorial para a estimativa da orientação (guinada) levando em consideração as medições do giroscópio e do magnetômetro da MPU9250 (IMU de 9 DOF). Este módulo é detalhado na Figura 4.17, o qual tem 10 módulos instanciados internamente: (1) *módulo A*, (2) *módulo 1*, (3) *módulo 2*, (4) *módulo 3*, (5) *módulo 4*, (6) *módulo 5*, (7) *mux1*, (8)*mux2*, (9) *cont1* e (10) *cont2*. O *módulo A* realiza um cálculo prévio de divisão, enquanto os contadores (*cont1* e *cont2*) foram desenvolvidos para ter conhecimento sobre o número de rodadas do *EKF2*, e com os quais controlam-se os multiplexadores (*mux1* e *mux2*) que são os que definem as entradas do vetor de estados (x_k) e do erro de covariância (P_k).

Similarmente ao *EKF1*, cada um dos *módulos 1*, *2*, *3*, *4* e *5* do *EKF2* calcula um passo específico do EKF, com a diferença que o Jacobiano neste caso é $A = 1$ e que as operações realizadas dentro deste módulo são escalares; isso devido que o *módulo EKF2* trabalha com uma variável de estado (guinada).

Portanto, têm-se só cinco módulos principais. O *master* ativa o sinal de *start* e da mesma forma deteta o sinal *ready_EKF2*.

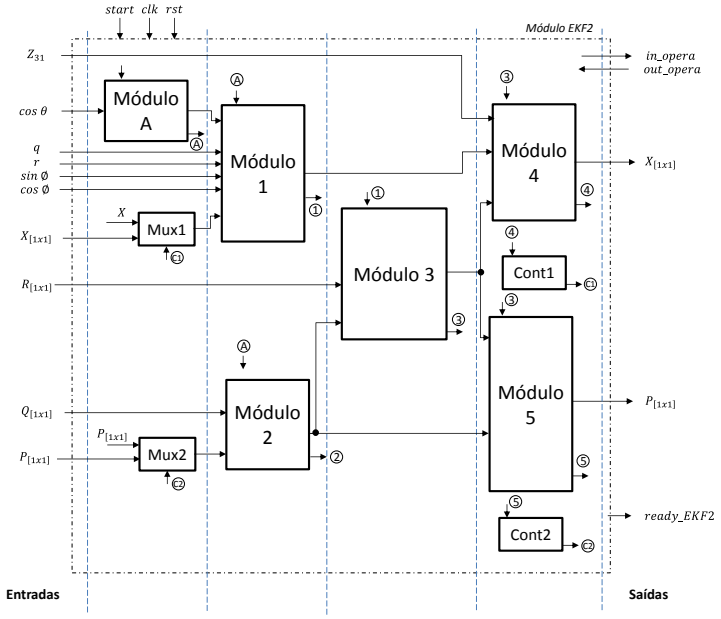
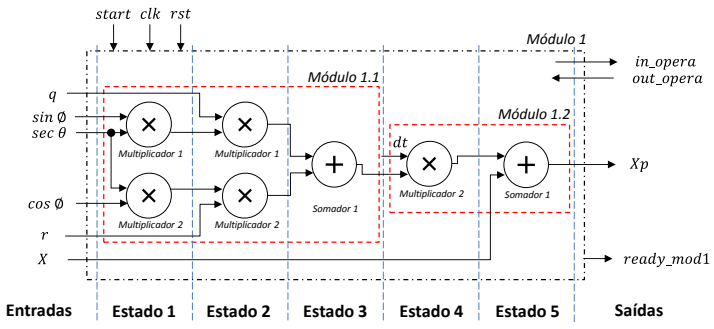
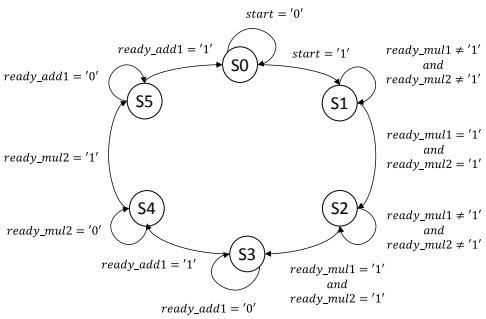


Figura 4.17: Módulo *EKF2*.

Na Figura 4.18 mostra-se o *módulo 1* do *EKF2*, o qual calcula a estimativa do vetor de estado $x_p = x + xdot \cdot dt$ na etapa de predição, o qual é o primeiro passo do EKF (vide Algoritmo 1). Para isto, o módulo tem dois sub-módulos e além disso, está baseado em uma FSM de cinco estados (vide Figuras 4.18(a) e 4.18(b)). No *sub-módulo 1.1* é calculado $xdot = q \cdot \sin\phi \cdot \sec\theta + r \cdot \cos\phi \cdot \sec\theta$, através de quatro operações de multiplicação e uma operação de soma. Enquanto o *módulo 1.2* tem somente uma multiplicação ($xdot \cdot dt$) e uma soma ($x + xdot \cdot dt$).



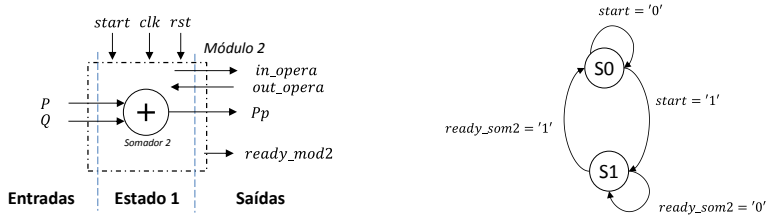
(a) Operadores do *módulo 1* do *EKF2*.



(b) FSM do *módulo 1* do *EKF2*.

Figura 4.18: Operadores e FSM do *módulo 1* do *EKF2*.

O *módulo 2* tem envolvida uma operação de soma, a fim de calcular o segundo passo do EKF (vide Algoritmo 1), o qual é o erro de covariância ($P_p = P + Q$). Isto é feito a partir da covariância a priori (P_p) e do ruído do processo (Q) que apresenta-se na Figura 4.19(a). A FSM deste módulo tem dois estados, a qual detalha-se na Figura 4.19(b) junto às condições de transição de estados. Adicionalmente, este segundo módulo do *EKF2* observa-se na Figura 4.19.

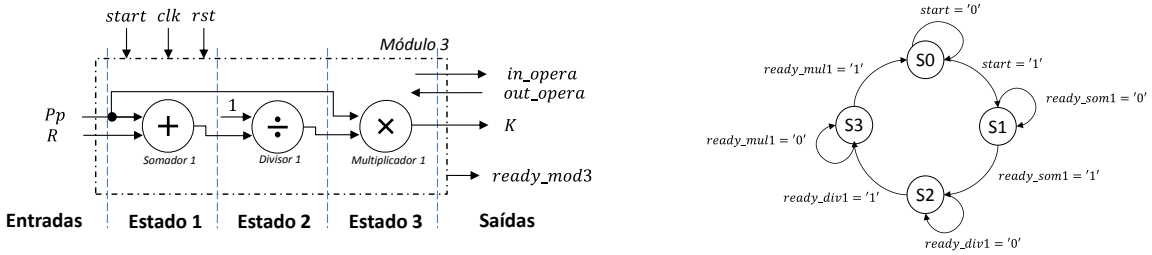


(a) Operadores do *módulo 2* do *EKF2*.

(b) FSM do *módulo 2* do *EKF2*.

Figura 4.19: Operadores e FSM do *módulo 2* do *EKF2*.

Na Figura 4.20 observa-se o *módulo 3* do *EKF2*, onde é calculado o ganho de Kalman ($K = P_p \cdot (P_p + R)^{-1}$), o qual é o terceiro passo do EKF (vide Algoritmo 1). Esse módulo baseia-se em uma FSM de quatro estados (vide Figura 4.20(b)). Portanto, as operações para calcular o ganho de Kalman são uma soma ($P_p + R$), uma divisão ($(P_p + R)^{-1}$) e uma multiplicação ($P_p \cdot (P_p + R)^{-1}$), assim como detalha-se na Figura 4.20(a). Neste caso, o valor inverso foi determinado para um escalar, feito no estado 2.

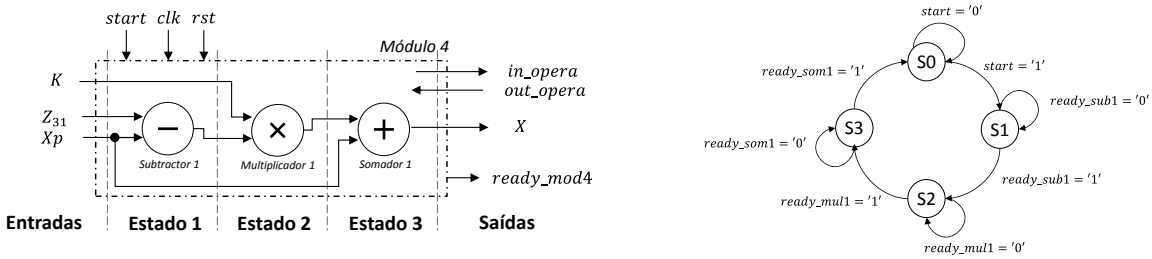


(a) Operadores do *módulo 3* do *EKF2*.

(b) FSM do *módulo 3* do *EKF2*.

Figura 4.20: Operadores e FSM do *módulo 3* do *EKF2*.

O *módulo 4* do *EKF2* mostrado na Figura 4.21, foi desenvolvido com o intuito de determinar o quarto passo do EKF (vide Algoritmo 1), isto é a estimativa das variáveis de estado ($x = x_p + K \cdot (Z - x_p)$) na etapa de correção. Para isto foram necessárias uma operação de subtração ($Z - x_p$), uma de multiplicação ($K \cdot (Z - x_p)$) e uma de soma ($x_p + K \cdot (Z - x_p)$) (vide Figura 4.21(a)). Este módulo está composto por uma FSM de quatro estados, a qual é apresentada na Figura 4.21(b).

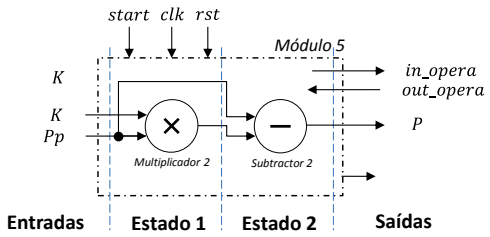


(a) Operadores do *módulo 4* do *EKF2*.

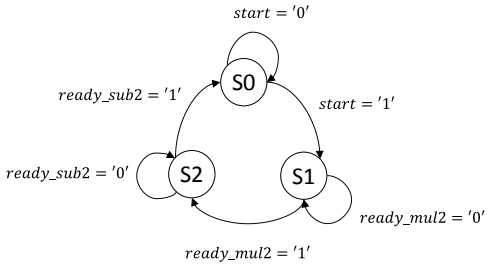
(b) FSM do *módulo 4* do *EKF2*.

Figura 4.21: Operadores e FSM do *módulo 4* do *EKF2*.

No *módulo 5* (vide Figura 4.22) calcula-se o erro de covariância $(P = P_p - K \cdot P_p)$ na etapa de correção, ou seja o quinto passo do EKF (vide Algoritmo 1); levando em consideração uma operação de multiplicação $(K \cdot P_p)$ e uma subtração $(P_p - K \cdot P_p)$, as quais são detalhadas na Figura 4.22(a). A FSM deste módulo tem dois estados e mostra-se na Figura 4.22(b) junto às condições de transição de estado.



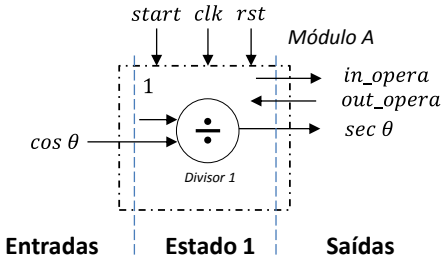
(a) Operadores do *módulo 5* do *EKF2*.



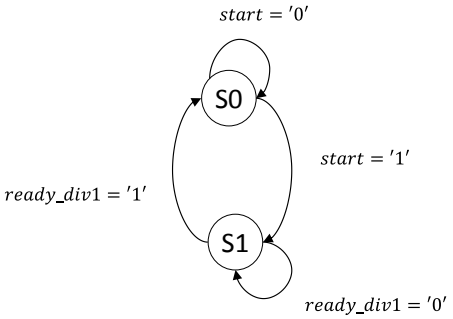
(b) FSM do *módulo 5* do *EKF2*.

Figura 4.22: Operadores e FSM do *módulo 5* do *EKF2*.

O *módulo A* do *EKF2* apresenta-se na Figura 4.23. Este módulo tem uma operação de divisão na que calcula-se o $\sec\theta$, a partir da entrada de $\cos\theta$ (vide Figura 4.23(a)). A FSM de dois estados com as respectivas condições de transição de estados são mostradas na Figura 4.23(b).



(a) Operadores do *módulo A* do *EKF2*.



(b) FSM do *módulo A* do *EKF2*.

Figura 4.23: Operadores e FSM do *módulo A* do *EKF2*.

4.3.5 *Accelerometer*

O módulo *accelerometer* executa os cálculos de rolagem e arfagem, em relação aos modelos de medição apresentados nas Equações 2.8 e 2.9, respectivamente; apresentados na subseção 2.2.3. Na Figura 4.24 mostra-se este módulo, o qual tem quatro módulos internos: (1) *master accel*, (2) *Roll*, (3) *Pitch* e (4) *mux1*. Este último comuta o acesso ao *cordic2* (*atan*) para os módulos *Roll* e *Pitch*.

O módulo *master accel* está baseado em uma FSM (vide Figura 4.25), composta por três estados: (1) *waiting*, (2) *roll* e (3) *pitch*. Este módulo é o "cérebro"do processamento para o cálculo da *attitude* (rolagem e arfagem). Similarmente ao *master* principal instanciado no *Top_module* da arquitetura (descrito na seção 4.3.1), este controla os sinais de *start* e *ready* em cada um dos outros dois módulos *Roll* e *Pitch*.

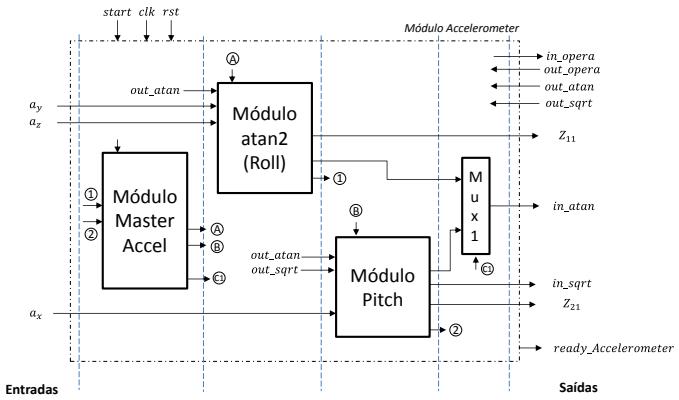


Figura 4.24: Módulo *accelerometer*.

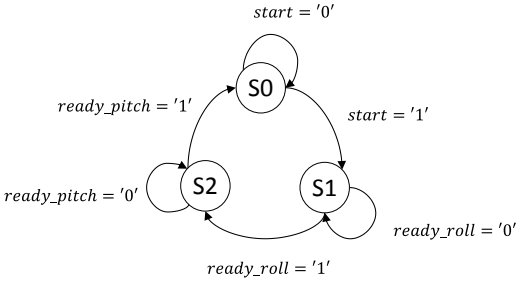
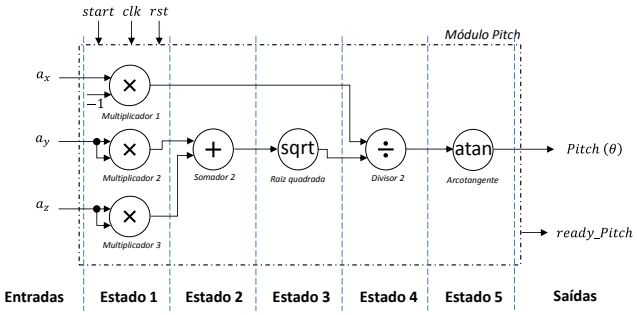
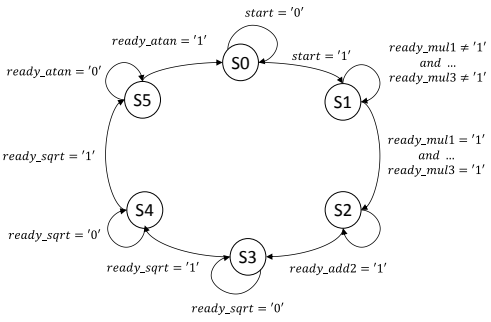


Figura 4.25: FSM do módulo *master accel*. S0 (*waiting*), S1 (*roll*) e S2 (*pitch*)

O módulo *Roll* determina o valor da Equação 2.8 (levando em consideração as informações do acelerômetro nos três eixos) através do módulo da arcotangente para os quatro quadrantes (*atan2*), apresentado na seção 4.2.6. A Figura 4.26 descreve o módulo *Pitch*, o qual se compõe de três multiplicações, uma soma, uma raiz quadrada, uma divisão e uma arcotangente (vide Figura 4.26(a) a fim de avaliar a Equação 2.9. Este módulo baseia o seu funcionamento em uma FSM de seis estados (vide Figura 4.26(b)).



(a) Operadores do módulo *Pitch*.



(b) FSM do módulo *Pitch*.

Figura 4.26: Operadores e FSM do módulo *Pitch*.

4.3.6 *Magnetometer*

O módulo *Magnetometer* calcula o terceiro ângulo de Euler (guinada), levando em consideração as medições do magnetômetro e os modelos descritos nas Equações 2.10, 2.11 e 2.12 (vide seção 2.2.4). Este módulo detalha-se na Figura 4.27, o qual tem três módulos instanciados internamente: (1) *Bfy*, (2) *Bfx* e (3) *Yaw*. Adicionalmente, o módulo *Magnetometer* é ativado através do sinal *start_magn*, assim como envia o sinal *ready_magn* quando há uma saída valida. Ao igual que o sub-módulo *Roll* do módulo *accelerometer* descrito na seção 4.3.5, o sub-módulo *Yaw* realiza o cálculo da Equação 2.12 através do módulo da função *arcotangente* para quatro quadrantes (*atan2*) apresentado na subseção 4.2.6.

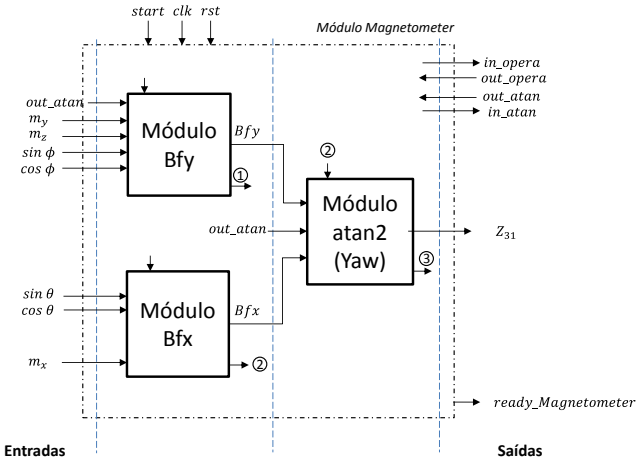
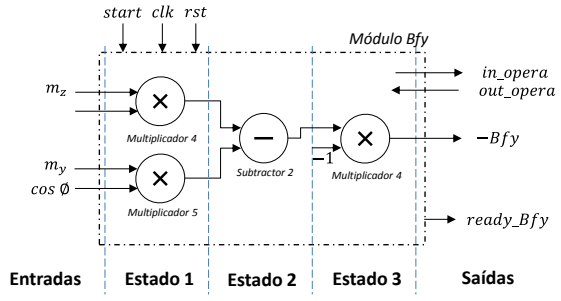
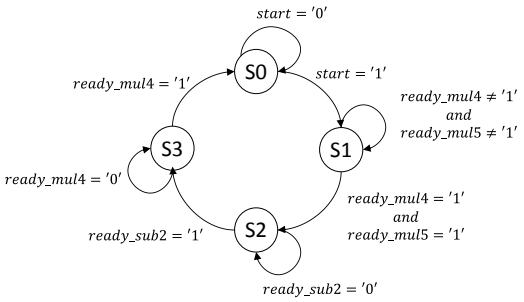


Figura 4.27: Módulo *magnetometer*.

O módulo *Bfy* é mostrado na Figura 4.28, no qual determina-se o valor do campo magnético na componente Y, através da Equação 2.10. Para isto, foi necessário implementar três operações de multiplicação e uma de subtração, assim como detalha-se na Figura 4.28(a). Além disso, a FSM deste módulo (quatro estados) junto com as restrições de transição de estados, apresentam-se na Figura 4.28(b).



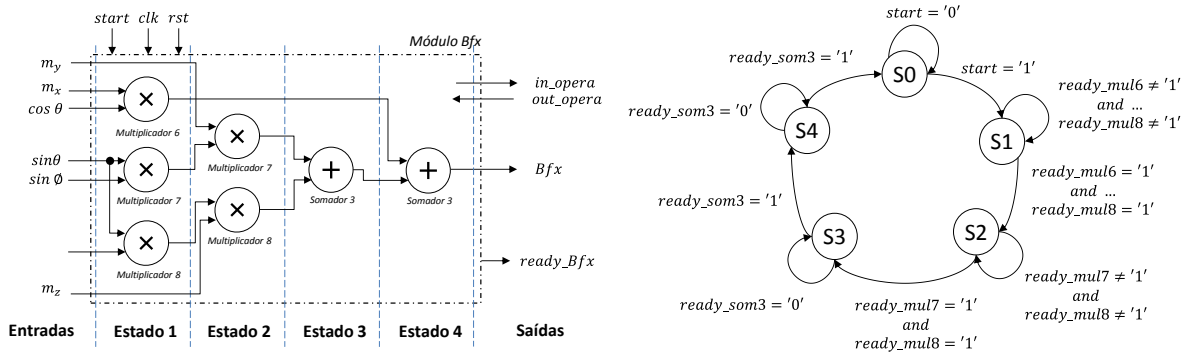
(a) Operadores do módulo *Bfy*.



(b) FSM do módulo *Bfy*.

Figura 4.28: Operadores e FSM do módulo *Bfy*.

A Figura 4.29 apresenta o módulo *Bfx*, o qual foi desenvolvido com o intuito de calcular em *hardware* a componente do campo magnético no eixo X, levando em consideração a Equação 2.11. Para isto, foram utilizadas cinco operações de multiplicação e duas de soma (vide Figura 4.29(a)), sendo assim, uma sequência de cinco estados em uma FSM detalhada na Figura 4.29(b).



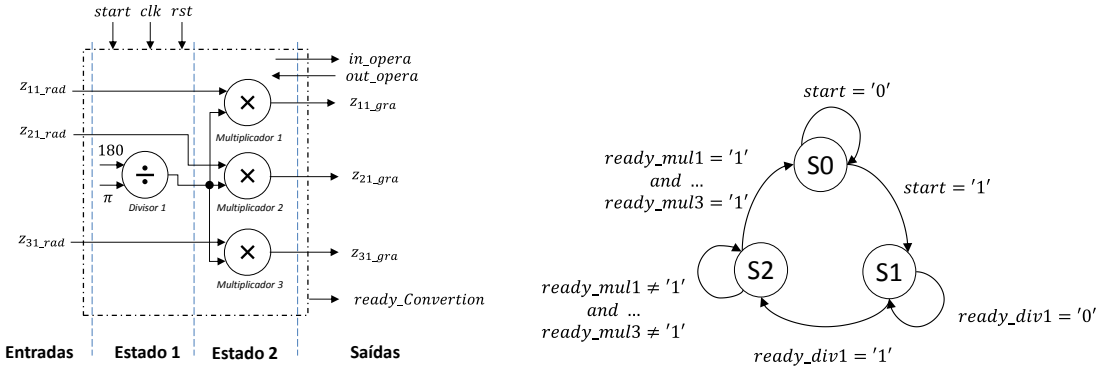
(a) Operadores do módulo *Bfx*.

(b) FSM do módulo *Bfx*.

Figura 4.29: Operadores e FSM do módulo *Bfx*

4.3.7 *Conversion*

O módulo *conversion* é mostrado na Figura 4.30, sendo o mesmo desenvolvido nesta arquitetura a fim de converter de radianos para graus as estimativas dos ângulos de Euler (rolagem, arfagem e guinada), para assim obter uma representação visualmente mais compreensível. Para isto, levou-se a cabo uma operação de divisão e três multiplicações, tal como detalha-se na Figura 4.30(a). Adicionalmente, o funcionamento deste módulo de conversão de unidades está baseado na lógica de uma FSM, a qual mostra-se na Figura 4.30(b).



(a) Operadores do módulo *Conversion*.

(b) FSM do módulo *Conversion*.

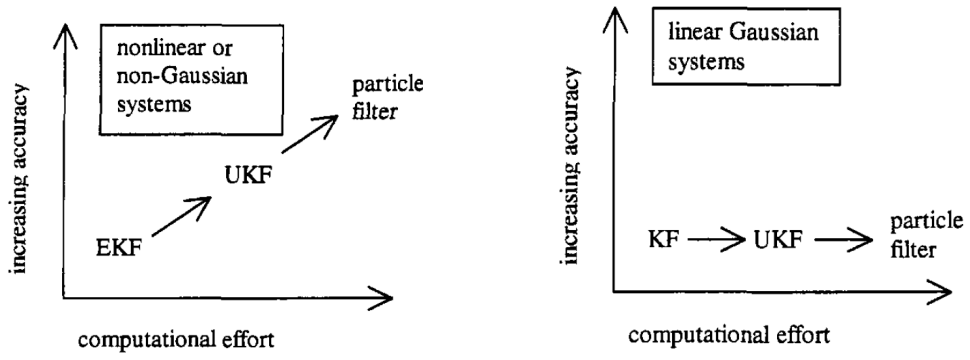
Figura 4.30: Operadores e FSM do módulo *Conversion*

Entre os aspectos mais relevantes a serem destacados da arquitetura para uma técnica de fusão sensorial de IMUs, a fim de estimar a posição e orientação em tempo real estão: (a) controle do processamento da arquitetura através de um módulo *master* baseado em uma FSM; (b) uso e compartilhamento dos módulos operacionais (aritméticos e trigonométricos) em ponto flutuante de 27 bit;, (c) comunicação da IMU com o FPGA através do protocolo I2C implementado em *software* (Nios II); (d) desenvolvimento de um módulo para o cálculo da função *arcotangente2* e (e) desenvolvimento de módulos independentes (*EKF1* e *EKF2*) para fusão sensorial, com o qual dita arquitetura pode ser utilizada para aplicações com IMUs de 9 DOF ou de 6 DOF.

Capítulo 5

Estimadores Não Lineares

Neste capítulo são apresentadas diferentes abordagens para estimação de estados de sistemas altamente não lineares, as quais são baseadas no Filtro de Partículas (PF). Essas abordagens são: (a) filtro de partículas estendido (EPF), (b) filtro de partículas unscented (UPF), (c) filtro de partículas com PSO (PF-PSO), (d) filtro de partículas estendido com PSO (EPF-PSO) e (e) filtro de partículas *unscented* com PSO (UPF-PSO). Além disso, apresentam-se as quatro técnicas de re-amostragem (*resampling*) mais estudadas e conhecidas na área do PF: (a) *multinomial*, (b) *systematic*, (c) *stratified* e (d) *residual*. Na Figura 5.1 mostram-se duas comparações para os filtros estimadores de estado, a primeira (vide Figura 5.1(a)) compara os filtros para sistemas não lineares, enquanto a segunda é a comparação dos mesmos filtros para sistemas lineares (vide Figura 5.1(b)). Nas duas comparacoes detalha-se que o UKF fornece um equilíbrio entre o baixo custo computacional do KF e o alto desempenho do PF (SIMON, 2006).



(a) Sistemas não lineares (b) Sistemas lineares

Figura 5.1: Comparação dos filtros para estimação de estados (adaptado de (SIMON, 2006)).

5.1 Filtro de partículas

O filtro de partículas é um método Sequencial Monte Carlo (SMC), utilizado para estimar os estados de sistemas dinâmicos que apresentam uma alta não linearidade (DOUCET; JOHANSEN, 2009). A primeira versão deste filtro foi o *Sequential Importance Sampling* (SIS), que baseia seu algoritmo em realizar a

estimação do vetor de estados representando a PDF em elementos denominados de *partículas*, onde cada partícula é uma possível solução do sistema, relacionando: (a) o modelo dinâmico do sistema, (b) as medições dos sensores e (c) os ruídos dos sinais do sistema e das medições (BROWN; HWANG, 1997). Além disso, este estimador tem outra versão que relaciona uma etapa de re-amostragem, mais conhecido como *Sequential Importance Resampling* (SIR); o qual melhora a estimação de estados através de uma seleção e multiplicação das "melhores"partículas, e apagando as "piores"(BROWN; HWANG, 1997). Uma das principais diferencias com o KF é que no PF podem-se utilizar ruídos que sigam PDFs não Gaussianas (RISTIC; ARULAMPALAM; GORDON, 2003). Na Figura 5.2 detalha-se a saída para cada passo de tempo do PF, a qual está dada pela média das partículas, sendo que podem ser utilizadas outras métricas tais como o desvio padrão e a mediana para a avaliação da estimação de estados (LLANOS, 2017).

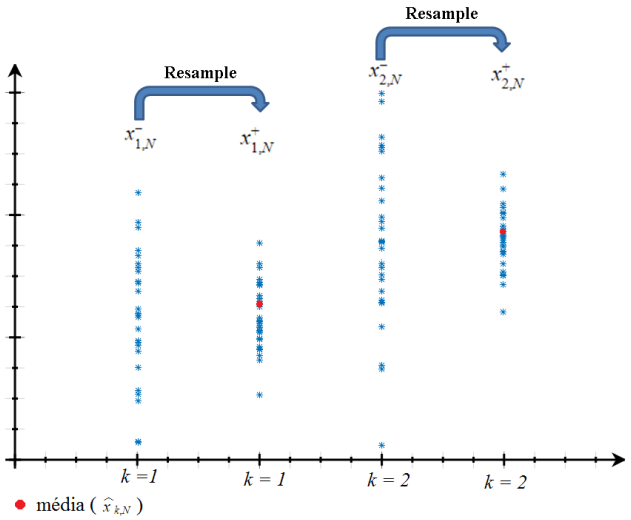


Figura 5.2: Saída de estimação do filtro de partículas (adaptado de (LLANOS, 2017)).

5.1.1 *Sequential Importance Sampling* (SIS)

O *Sequential Importance Sampling* foi a primeira versão do PF, sendo baseada no método de integração de Monte Carlo. O SIS está definido para sistemas não lineares, levando em consideração para a estimação de estados x_k os modelos matemáticos do processo f_k e da medição h_k, descritos pelas Equações 5.1 e 5.2, respectivamente. Onde w_k e v_k são os ruídos independentes (TURNER; SHERLOCK, 2013).

$$x_k = f_k(x_{k-1}, w_k) \tag{5.1}$$

$$y_k = h_k(x_k, v_k) \tag{5.2}$$

Na etapa de amostragem, as partículas são atualizadas através da Equação 5.3, sendo que cada partícula do passo anterior $(k-1)$ é utilizada para representar um novo conjunto de partículas no estado atual (k), através do modelo não linear do processo (vide Equação 5.1) (TURNER; SHERLOCK, 2013).

$$x_k^{i=1:N} \sim p(x_k | x_{k-1}^{i=1:N}) \tag{5.3}$$

Além disso, nesta versão genérica do PF calculam-se as probabilidades ou pesos das partículas a partir da função de distribuição utilizada (TURNER; SHERLOCK, 2013). Isto pode ser visto na Equação 5.4, onde apresenta-se também o exemplo para uma função de distribuição Gaussiana, sendo que dita função depende da média (μ) e do desvio padrão (σ) da amostra.

$$w_k^i \propto w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, zk)} \sim \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5.4}$$

O pseudocódigo 3 apresenta as etapas (inicialização, predição e correção) do PF. A inicialização das partículas N é feita da linha 1 à 3, a qual baseia-se na PDF da covariância P. Nas linhas 4 à 13 executa-se o algoritmo para cada passo de tempo k, sendo que da linha 5 à 8 realiza-se a etapa de predição para cada partícula. Adicionalmente, na linha 6 se faz a amostragem (*sampling*) das partículas tendo em conta o modelo dinâmico não linear do sistema, enquanto que na linha 7 determinam-se os pesos das partículas. Na linha 9 são somados todos os pesos das partículas, e finalmente da linha 10 à 12 normalizam-se os pesos para cada partícula. A principal desvantagem desse filtro é a degeneração das partículas, a qual consiste em que após de algumas iterações todas as partículas exceto uma, convergirão a probabilidades (pesos) desprezíveis (RISTIC; ARULAMPALAM; GORDON, 2003). Para resolver esse problema, é necessário implementar o passo do *resampling* no PF, pelo qual esta versão do filtro obtém assim o nome de SIR.

Algoritmo 3 Filtro de partículas SIS

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.

Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: **para** k = 1 : tf **faça**
3: $x_k^{i=1:N} \sim p(x_k|x_{k-1}^{i=1:N})$
4: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
5: $w_k^{sum} = \sum_{i=1}^{N} w_i$
6: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
7: $out_k = \sum_{i=1}^{N} x_k^i/N$
8: **fim para**

5.1.2 *Sequential Importance Resampling* (SIR)

Esta versão do PF está baseada no SIS, sendo que o SIR implementa a etapa do *resampling*. No Algoritmo 4 mostra-se o pseudocódigo do SIR, onde na linha 7 mostra-se essa etapa implementada, a qual consiste em fazer uma amostragem nova para selecionar as melhores partículas e apagar as partículas mais ruins. A desvantagem da implementação do *resampling* é o empobrecimento da amostra, onde as partículas com maiores pesos multiplicam-se muitas vezes (com o passo das iterações), gerando pouca diversidade das partículas (RISTIC; ARULAMPALAM; GORDON, 2003). Como solução proposta na literatura para esta problemática, encontram-se algumas versões do PF, tais como o RPF e o MCMC *move step* (SIMON, 2006). Porém, outro enfoque tem sido proposto para melhorar a qualidade das partículas, o qual baseia-se na implementação do Filtro de Kalman não linear (EKF ou UKF) no passo de amostragem do PF (*sampling*), obtendo como resultado umas novas versões do PF, mais conhecidas como o EPF e o UPF

(SIMON, 2006). Além disso, há uma solução que consiste em implementar um algoritmo bioinspirado após da etapa do *resampling* no PF.

Algoritmo 4 Filtro de partículas SIR

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.

Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: **para** k = 1 : tf **faça**
3: $x_k^{i=1:N} \sim p(x_k|x_{k-1}^{i=1:N})$
4: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
5: $w_k^{sum} = \sum_{i=1}^{N} w_i$
6: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
7: $[x_k^{j=1:N}, w_k^{j=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
8: $x_k^{i=1:N} = x_k^{j=1:N}$
9: $out_k = \sum_{i=1}^{N} x_k^i/N$
10: **fim para**

5.1.2.1 Resampling

O *resampling* é uma etapa do PF que amostra um novo conjunto de partículas, partindo de uma população original de partículas a fim de representar melhor a distribuição de filtragem. Isto é feito reproduzindo as partículas com ponderações altas, a fim de substituir aquelas partículas com ponderações baixas, reduzindo assim a incerteza de estimação do filtro (vide Figura 5.3) (DOUCET, 2001). Esse passo é um componente essencial para o PF já que evita a degeneração das partículas (GORDON; SALMOND; SMITH, 1993); isto é, a discrepância de peso (probabilidade) entre as partículas aumenta com o tempo. Neste caso, todas as partículas terão pesos desprezíveis exceto uma; fazendo que o resultado da estimação não seja confiável (RISTIC; ARULAMPALAM; GORDON, 2003).

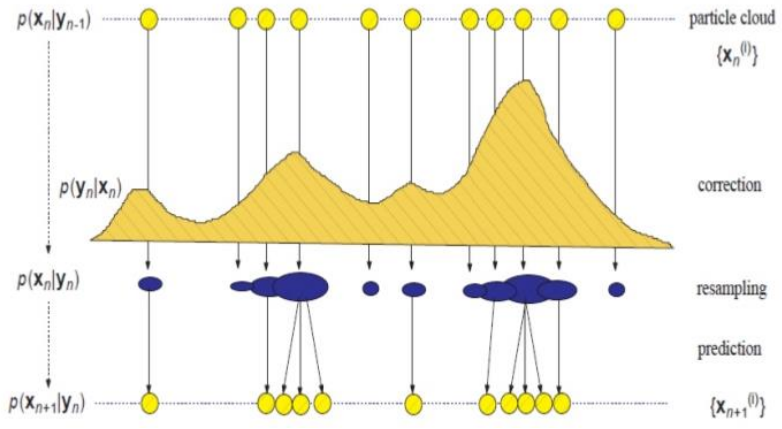


Figura 5.3: Lógica do *resampling* no filtro de partículas (adaptado de (SIDIBÉ, 2011)).

Porém, com a implementação do *resampling* gera-se um outro problema: o empobrecimento da amostra. Isto consiste em que as partículas que têm pesos grandes são estatisticamente selecionadas mais vezes;

portanto, a amostra resultante terá muitas partículas repetidas (RISTIC; ARULAMPALAM; GORDON, 2003). Porém, há técnicas para minimizar esse problema (a baixa diversidade das partículas), tais como as que foram tidas em conta neste trabalho (filtros híbridos otimizados, vide seção 5.4).

Neste contexto, há 4 abordagens de *resampling* que são as mais comuns: (a) *multinomial*, (b) *systematic*, (c) *residual* e (d) *stratified* (TURNER; SHERLOCK, 2013). Neste trabalho utilizaram-se esses 4 métodos de *resampling* com o intuito de fazer um análise do PF.

Resampling multinomial: Este *resampling* foi a primeira abordagem implementada no PF, sendo a mais simples e baseada na ideia do método *bootstrap* (GORDON; SALMOND; SMITH, 1993). Sua lógica consiste em gerar 'n' números independentes com uma PDF uniforme no intervalo (0,1] (DOUC; CAPPÉ, 2005); os quais são utilizados para escolher 'n' direções aleatórias independentes, assim como mostra-se na Figura 5.4 (BLANCO, 2017). O nome *"multinomial"* foi dado devido a que as contagens de duplicação $N^1, ..., N^i$ estão definidas pela distribuição multinomial $mult(n; w^1, ..., w^i)$, tendo em conta os pesos como parâmetros (DOUC; CAPPÉ, 2005).

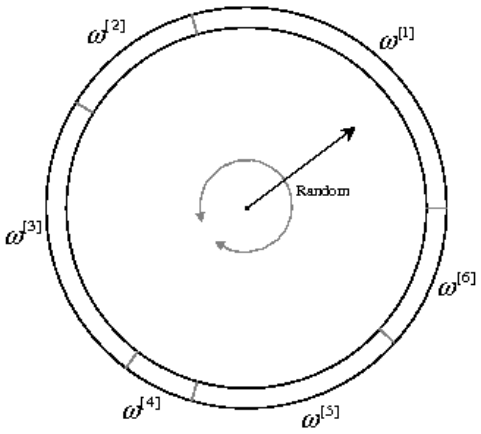


Figura 5.4: Esquema do resampling multinomial (adaptado de (BLANCO, 2017)).

O pseudocódigo deste método de *resampling* é mostrado no Algoritmo 5 (LI; BOLIC; DJURIC, 2015). Nas linhas 1 à 4 a soma acumulada dos pesos das partículas é calculada; na linha 5 inicializa-se o *"index"* j. Da linha 6 à 14 encontra-se o ciclo repetitivo para cada partícula $j \leq N$. Nas linhas 7 e 8 gera-se o número aleatório (que segue uma PDF uniforme em uma faixa (0,1] e inicializa-se o *"index"* i, respetivamente. Além disso, da linha 9 à 11 há outro ciclo repetitivo com condição $q_k^i < u$, onde internamente (na linha 10) se faz um incremento de '1' no *"index"* i. Porém, enquanto não seja cumprida essa condição, o algoritmo executa as linhas 12 e 13, nas quais incrementa-se em '1' no *"index"* j; e a partícula com o maior peso é reproduzida em cada rodada, até cumprir a condição da linha 8, respetivamente.

Algoritmo 5 Resampling multinomial

Entrada: Partículas e pesos, $x_k^{i=1:N}$, $w_k^{i=1:N}$].

Saída: Partículas e pesos a posteriori, $x_k^{j=1:N}$, $w_k^{j=1:N}$].

1: $q_k^0 = 0$

2: **para** i = 1 : N **faça**

3: $q_k^i = q_k^{i-1} + w_k^i$
4: **fim para**
5: $j = 0$
6: **enquanto** $j \leq N$ **faça**
7: $u \sim (0, 1]$
8: $i = 1$
9: **enquanto** $q_k^i < u$ **faça**
10: $i = i + 1$
11: **fim enquanto**
12: $j = j + 1$
13: $x_k^j = x_k^i$
14: **fim enquanto**

Resampling systematic: O *Resampling systematic* é também conhecido como *"universal sampling"*, sendo a única técnica de *resampling* em que as posições das partículas não são independentes (DOUC; CAPPÉ, 2005). Isto é porque este método gera só um número aleatório uniforme, o qual é o ponto de partida desta técnica para todas as N partículas; onde para cada partícula leva-se em conta um incremento fixo de $1/N$. Na Figura 5.5 detalha-se o funcionamento do *resampling systematic* ao redor de uma roda (BLANCO, 2017).

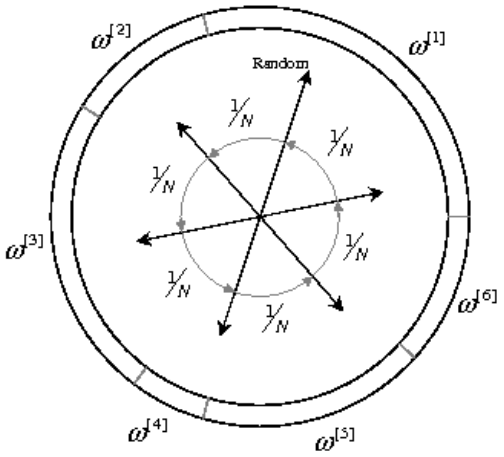


Figura 5.5: Esquema do resampling systematic (adaptado de (BLANCO, 2017)).

O pseudocódigo deste *resampling* é apresentado no Algoritmo 6 (LI; BOLIC; DJURIC, 2015), no qual observa-se da linha 1 à 4 o cálculo da soma acumulada dos pesos das partículas; nas linhas 5 e 6 inicializam-se os *"indexs"* j e i, respectivamente. Enquanto na linha 7 gera-se o único número aleatório que segue uma PDF uniforme em uma faixa (0,1/N]. No entanto, da linha 8 à 15 está o ciclo repetitivo para cada partícula $j \leq N$, sendo que na linha 9 determina-se o incremento j/n do número aleatório. Além disso, da linha 10 à 12 há outro ciclo repetitivo com condição $q_k^i < u$, onde internamente, na linha 11 se faz um incremento de '1' no *"index"* i. Portanto, quando não seja satisfeita essa condição ($q_k^i < u$), o algoritmo passa às linhas 13 e 14, nas quais incrementa-se em '1' o *"index"* j e a partícula com o maior

peso é reproduzida em cada rodada, respetivamente. Isto é feito até cumprir a condição da linha 8.

Algoritmo 6 Resampling systematic

Entrada: Partículas e pesos, $x_k^{i=1:N}$, $w_k^{i=1:N}$].
Saída: Partículas e pesos a posteriori, $x_k^{j=1:N}$, $w_k^{j=1:N}$].

1: $q_k^0 = 0$
2: **para** i = 1 : N **faça**
3: $\quad q_k^i = q_k^{i-1} + w_k^i$
4: **fim para**
5: $j = 0$
6: $i = 1$
7: $u_0 \sim U(0, 1/N]$
8: **enquanto** $j \leq N$ **faça**
9: $\quad u = u_0 + j/N$
10: $\quad$ **enquanto** $q_k^i < u$ **faça**
11: $\quad\quad i = i + 1$
12: $\quad$ **fim enquanto**
13: $\quad j = j + 1$
14: $\quad x_k^j = x_k^i$
15: **fim enquanto**

Resampling stratified: Na Figura 5.6 representa-se o funcionamento do *resampling stratified*, na qual o espaço de amostragem (roda) é dividido em segmentos proporcionais a $1/N$, sendo iguais ao número de partículas (N). Portanto, N números aleatórios são gerados independentemente. A diferença deste tipo de *resampling* com o multinomial é que cada partícula selecionada (reproduzida) é escolhida de um segmento diferente, ao contrário acontece no multinomial, onde as partículas são escolhidas tendo em conta todo o espaço de amostragem, portanto uma partícula pode-se reproduzir muitas vezes (BLANCO, 2017).

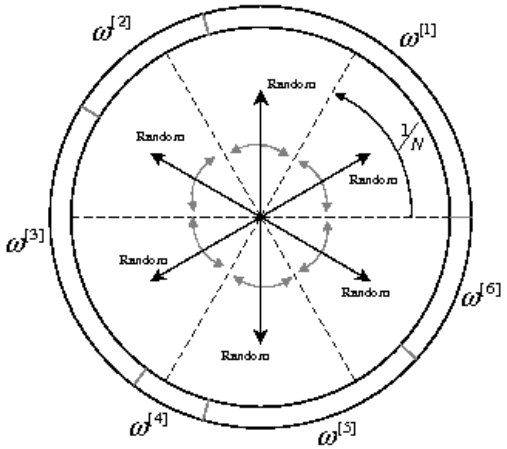


Figura 5.6: Esquema do resampling stratified (adaptado de (BLANCO, 2017)).

No Algoritmo 7 mostra-se o pseudocódigo do *resampling stratified* (LI; BOLIC; DJURIC, 2015).

Neste algoritmo, da linha 1 à 4 a soma acumulada dos pesos das partículas é realizada, enquanto, na linha 5 inicializa-se o *"index"* j. Na linha 6 inicializa-se o *"index"* i. Da linha 7 à 15 está o ciclo repetitivo para cada partícula $j \leq N$, onde internamente na linha 8 geram-se os números aleatórios, que seguem uma PDF uniforme em uma faixa (0,1/N]. Porém, na linha 9 determina-se o incremento j/N do número aleatório gerado anteriormente. Além disso, da linha 10 à 12 há outro ciclo repetitivo com condição $q_k^i < u$. Na linha 11 se faz um incremento de '1' no *"index"* i. Enquanto não seja satisfeita a condição $q_k^i < u$, o algoritmo vai para às linhas 13 e 14, nas quais incrementa-se em '1' o *"index"* j e a partícula com o maior peso em cada rodada é reproduzida, respectivamente; até cumprir a condição da linha 8.

Algoritmo 7 Resampling stratified

Entrada: Partículas e pesos, $x_k^{i=1:N}$, $w_k^{i=1:N}$].
Saída: Partículas e pesos a posteriori, $x_k^{j=1:N}$, $w_k^{j=1:N}$].

1: $q_k^0 = 0$
2: **para** i = 1 : N **faça**
3: $\quad q_k^i = q_k^{i-1} + w_k^i$
4: **fim para**
5: $j = 0$
6: $i = 1$
7: **enquanto** $j \leq N$ **faça**
8: $\quad u_0 \sim U(0, 1/N]$
9: $\quad u = u_0 + j/N$
10: $\quad$ **enquanto** $q_k^i < u$ **faça**
11: $\quad\quad i = i + 1$
12: $\quad$ **fim enquanto**
13: $\quad j = j + 1$
14: $\quad x_k^j = x_k^i$
15: **fim enquanto**

Resampling residual: O *resampling residual* ou também conhecido como *resampling remainder* (DOUC; CAPPÉ, 2005) é composto por duas etapas. A primeira consiste em realizar uma copia das partículas originais deterministicamente, de acordo a $N_i = [N * w^i]$ e determinando os pesos de essas novas partículas através de $\tilde{w}^i = w^i - N_i/N$. Posteriormente, a segunda etapa baseia-se basicamente em executar o *resampling multinomial* (visto na seção 5.1.2.1) com as partículas e os pesos obtidos na etapa anterior. A Figura 5.7 mostra esta técnica de *resampling* (BLANCO, 2017) onde detalha-se a sua lógica.

O Algoritmo 8 apresenta o pseudocódigo do *resampling residual* (LI; BOLIC; DJURIC, 2015). O qual consiste de duas etapas, a primeira baseia-se em uma replicação determinística e a segunda etapa faz uma amostragem aleatória (resampling multinomial) utilizando os pesos residuais. Da linha 1 à 4 determinam-se os pesos residuais. Na linha 5 inicializa-se o *"index"* n, enquanto da linha 6 à 11 realiza-se o processo de replicação das partículas. Na linha 12 inicializa-se o *"index"* Tk. Na linha 13 à 15 calculam-se novamente os pesos residuais com as variáveis atualizadas. Finalmente, o *resampling* multinomial (vide seção 5.1.2.1) é feito na linha 16.

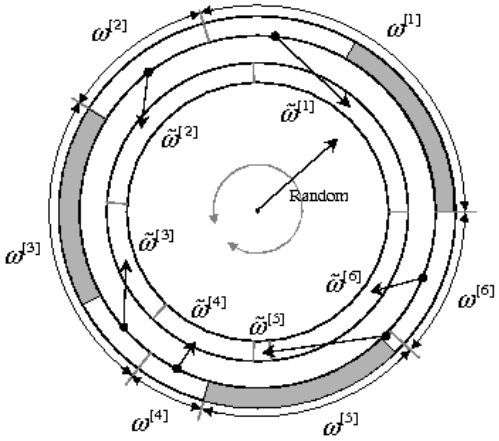


Figura 5.7: Esquema do resampling residual (adaptado de (BLANCO, 2017)).

Algoritmo 8 Resampling residual

Entrada: Partículas e pesos, $x_k^{i=1:N}$, $w_k^{i=1:N}]$.
Saída: Partículas e pesos a posteriori, $x_k^{j=1:N}$, $w_k^{j=1:N}]$.

para $i = 1 : N$ **faça**
    $M_k^i = floor(N\dot{w}_k^i)$
    $\hat{w}_k^i = w_k^i - M_k^i/N$
fim para
$n = 0$
para $i = 1 : N$ **faça**
    para $h = 1 : M_k^i$ **faça**
        $n = n + 1$
        $\tilde{x}_k^j = x_k^i$
    fim para
fim para
$T_k = n$
para $i = 1 : N$ **faça**
    $\hat{w}_k^i = \hat{w}_k^i \dot{N}/(N - T_k^i)$
fim para
$[x_k^j] = Resampling\ multinomial[x_k^i, \hat{w}_k^i, N - T_k]$

5.2 Filtros híbridos

Os filtros híbridos consistem principalmente na combinação de diferentes estimadores de estados. Neste caso, as combinações são feitas com o filtro de partículas e as versões não lineares do Filtro de Kalman (já seja o EKF ou o UKF). Essas abordagens têm sido propostas a fim de prevenir o empobrecimento das partículas (o qual gera-se com o *resampling*), procurando assim melhorar o desempenho do PF. Nesta abordagem as partículas são atualizadas a través de um banco de filtros de Kalman; isto é, para cada partícula é rodado um KF com o intuito de melhorar a qualidade (probabilidade ou peso) de cada uma de-

las. Este tipo de filtro híbrido é mostrado detalhadamente na Figura 5.8, onde observa-se a implementação do KF na etapa de amostragem (*sampling*) do PF.

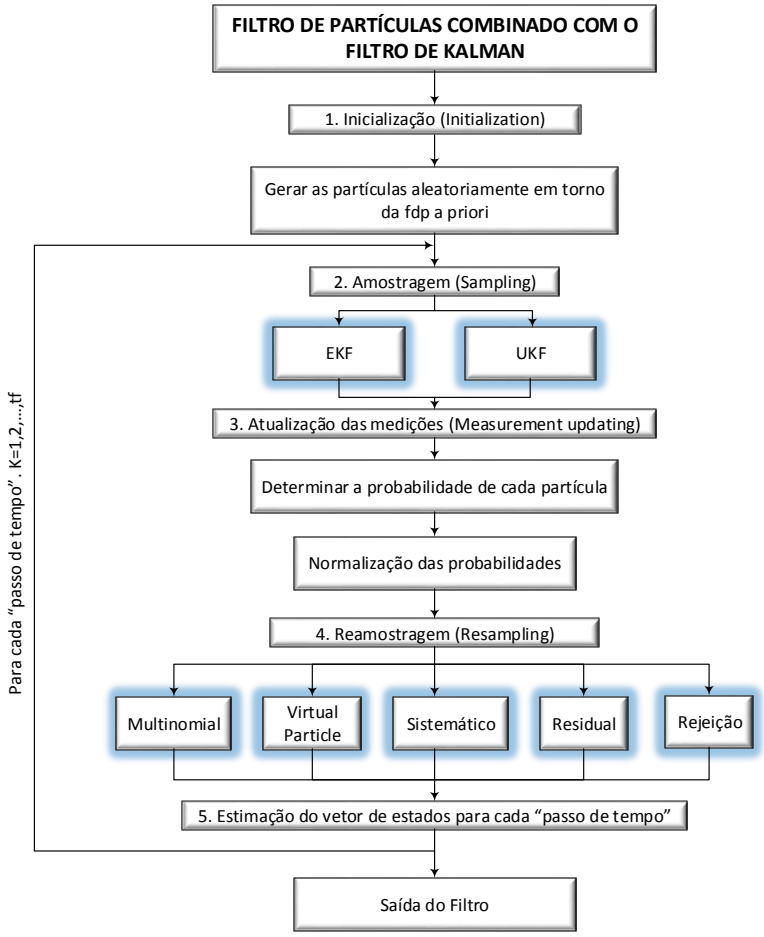


Figura 5.8: Combinação do filtro de partículas com o filtro de Kalman

5.2.1 Filtro de Partículas Estendido

O filtro de partículas estendido combina as vantagens entre o PF e o EKF (vide subseções 5.1.2 e 2.4.2.1), sendo que o EKF é implementado na etapa de amostragem do PF. Neste caso, cada partícula é atualizada através de um filtro EKF, sendo essa abordagem apresentada no Algoritmo 9, onde na linha 4 observa-se que para cada partícula é rodado um EKF. Esta estrutura é chamada de *banco de EKFs* (SIMON, 2006).

Algoritmo 9 Filtro de partículas estendido

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.

Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$

2: $P_0^{i=1:N} = P$

3: **para** k = 1 : tf **faça**

4: $[\hat{x}_k^{i=1:N}, \hat{P}_k^{i=1:N}]$ = EKF $[x_{k-1}^{i=1:N}, P_{k-1}^{i=1:N}]$
5: $x_k^{i=1:N} \sim \hat{x}_k^{i=1:N} + PDF(\hat{P}_k^{i=1:N})$
6: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
7: $w_k^{sum} = \sum_{i=1}^{N} w_i$
8: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
9: $[x_k^{j=1:N}, w_k^{j=1:N}, P_k^{j=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
10: $x_k^{i=1:N} = x_k^{j=1:N}$
11: $P_k^{i=1:N} = \hat{P}_k^{i=1:N}$
12: $out_k = \sum_{i=1}^{N} x_k^i/N$
13: **fim para**

5.2.2 Filtro de Partículas Unscented

O *Filtro de Partículas Unscented* (UPF) foi proposto em (MERWE et al., 2001). Esta abordagem é igual ao EPF (vide seção 5.2.1), sendo que neste caso utiliza-se um banco de N estimadores de UKFs (vide seção 2.4.2.2) a fim de atualizar as partículas. Isto pode ser visto no Algoritmo 10, especificamente na linha 4 desse pseudocódigo do UPF (SIMON, 2006).

Algoritmo 10 Filtro de partículas *unscented*

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.
Saída: Média das partículas, out_k.
1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: $P_0^{i=1:N} = P$
3: **para** k = 1 : tf **faça**
4: $[\hat{x}_k^{i=1:N}, \hat{P}_k^{i=1:N}]$ = UKF $[x_{k-1}^{i=1:N}, P_{k-1}^{i=1:N}]$
5: $x_k^{i=1:N} \sim \hat{x}_k^{i=1:N} + PDF(\hat{P}_k^{i=1:N})$
6: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
7: $w_k^{sum} = \sum_{i=1}^{N} w_i$
8: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
9: $[x_k^{j=1:N}, w_k^{j=1:N}, P_k^{j=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
10: $x_k^{i=1:N} = x_k^{j=1:N}$
11: $P_k^{i=1:N} = \hat{P}_k^{i=1:N}$
12: $out_k = \sum_{i=1}^{N} x_k^i/N$
13: **fim para**

5.3 Algoritmos bioinspirados

Os algoritmos bioinspirados são metaheurísticas e estão inspirados no comportamento de enxames, cardumes, alcateias e outros; levando em consideração a inteligência coletiva como base fundamental a fim de encontrar boas soluções. Por isto, este tipo de algoritmos caracterizam-se por utilizar "partículas"ou "indivíduos"de uma população, como possíveis soluções na busca do ponto ótimo (KENNEDY, 2006). A otimização de problemas altamente complexos consiste em maximizar ou minimizar uma função $f(x)$

(denominada de função custo ou *fitness*) com o intuito de encontrar a melhor solução do problema. Alguns algoritmos de otimização são: (a) Otimização por Enxame de Partículas (PSO), (b) Otimização por Colónia de Formigas (ACO), Otimização por Colônia de Abelhas (ABC), Otimização por Salto Aleatório de Sapos (SFLA), Otimização por Colônia de Vagalumes (FA), entre outros. Neste trabalho utilizou-se unicamente o PSO devido que ao invés de fazer o estudo com outros algoritmos bioinspirados, optou-se por estudar métodos de adição de diversidade para o PSO, a fim de melhorar ainda mais o desempenho desse algoritmo.

5.3.1 Otimização por enxame de partículas (PSO)

O algoritmo de otimização por enxame de partículas (PSO) foi proposto em (EBERHART; KENNEDY, 1995). É um dos algoritmos bioinspirados mais utilizados para resolver problemas de otimização, o qual foi baseado especificamente no comportamento dos cardumes de peixes e dos enxames de aves na procura de alimento. No PSO, os indivíduos de uma população são simulados como partículas (pontos), onde cada partícula tem uma velocidade através da qual movimenta-se no espaço de busca. Além disso, o PSO utiliza os conhecimentos tanto de cada individuo quanto de toda a população em geral (EBERHART; KENNEDY, 1995).

No Algoritmo 11 apresenta-se o pseudocódigo deste algoritmo bioinspirado, onde na linha 1 são inicializadas as partículas de acordo a uma PDF Gaussiana, enquanto na linha 2 inicializam-se os *fitness* de cada partícula segundo o caso de otimização (minimização\maximização). Da linha 3 à 19 executa-se o algoritmo até atingir um limite definido. Da linha 4 à 9 avalia-se cada partícula (S) na função *fitness*, assim como deteta-se o melhor *fitness* de cada partícula, sendo que nas linhas 11 e 12 é definido a melhor partícula global y_s. Finalmente da linha 13 à 18 é atualizada tanto a velocidade quanto a posição das partículas. Finalmente, a linha 20 mostra a saída do PSO.

Algoritmo 11 PSO

Entrada: $S, n, c_1, c_2, x_{min}, x_{max}, v_{max}, iter_{max}$.

Saída: Melhores partículas out_y.

1: $x_{i=1:N} \sim x_{min} + PDF$
2: $f(y_{i=1:S}) = 1e^{10} \setminus 1e^{-10}$
3: **enquanto** $k \leq max_{iter}$ *ou* $f(y_s) < threshold$ **faça**
4: **para** i = 1 : S **faça**
5: $f(x_i) \sim f_{fitness}(x_i)$
6: **se** $f(x_i) < f(y_i) \setminus f(x_i) > f(y_i)$ **então**
7: $y_i = x_i$
8: $f(y_i) = f(x_i)$
9: **fim se**
10: **fim para**
11: $y_p = min(y_{i=1:S}) \setminus max(y_{i=1:S})$
12: $y_s = y_p$
13: **para** j = 1 : N **faça**
14: **para** i = 1 : S **faça**
15: $v_{kj} = v_{kj} + c_1 * U_1[0,1] * (y_{ikj} - x_{kj}) + c_2 * U_2[0,1] * (y_{sj} - x_{kj})$

16: $x_{kj} = x_{kj} + v_{kj}$
17: **fim para**
18: **fim para**
19: **fim enquanto**
20: $out_y = y_{i=1:S}$

5.3.2 Métodos de adição de diversidade

Os métodos de adição de diversidade são utilizados para melhorar o desempenho dos algoritmos bio-oinspirados, através de técnicas caracterizadas por explorar melhor o espaço de busca a fim de encontrar a solução ótima. Neste trabalho foram tidos em conta dois métodos de adição de diversidade para o PSO: (a) Aprendizagem Baseada em Oposição (OBL) e (b) Atrativo e Repulsivo (AR).

5.3.2.1 Aprendizagem Baseada em Oposição (OBL)

A Aprendizagem Baseada em Oposição (OBL) foi proposta em (TIZHOOSH, 2005), a qual é uma técnica que baseia-se na exploração e busca do ponto ótimo na direção oposta à original, com a qual têm-se maiores chances de encontrar a melhor solução (vide Figura 5.9) (MUÑOZ, 2016). Uma das principais características deste método é que a diversidade das partículas não é afetada. Através da Equação 5.5 geram-se as partículas opostas, onde $\tilde{x}$ é a partícula oposta de x, assim como x_{min} e x_{max} são os limites do espaço de busca (JABEEN; JALIL; BAIG, 2009).

$$\tilde{x} = x_{min} + x_{max} - x \tag{5.5}$$

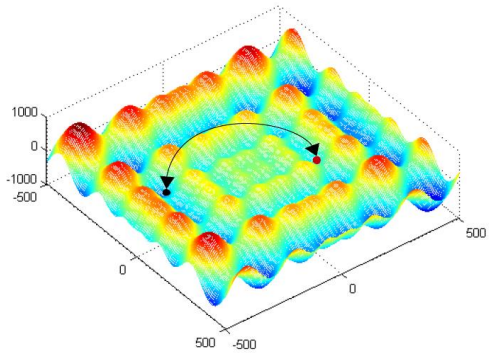


Figura 5.9: Aprendizagem Baseada em Oposição (adaptado de (MUÑOZ, 2016)).

5.3.2.2 AR

Na Figura 5.10 detalha-se o método Atrativo e Repulsivo (AR) que consiste em realizar um equilíbrio entre a busca global e a busca local no espaço de busca, através das fases de atração (vide Figura 5.10(a)) e repulsão (vide Figura 5.10(b)) (MUÑOZ, 2016). Sendo que essas fases são comutadas levando em consideração uma medida de diversidade das partículas (vide Equação 5.6). Portanto, se a diversidade for

menor do que uma diversidade mínima $(div(S) < div_{min})$, as partículas são repulsadas, tendo uma direção negativa $(dir = -1)$; enquanto a fase atrativa $(dir = 1)$ aconteceria no caso contrario, isto é quando a diversidade for maior do que uma diversidade máxima $(div(S) > div_{max})$. O valor da direção é levado em consideração na Equação 5.7 de atualização de velocidade das partículas (RIGET; VESTERSTRØM, 2002).

$$div(S) = \frac{1}{|S||L|} \sum_{i=1}^{|S|} \sqrt{\sum_{j=1}^{N} (x_{i,j} - \frac{1}{|S|} \sum_{i=1}^{|S|} x_{i,j})^2} \tag{5.6}$$

$$v_{kj} = v_{kj} + dir(c_1 * U_1[0,1] * (y_{ikj} - x_{kj}) + c_2 * U_2[0,1] * (y_{sj} - x_{kj})) \tag{5.7}$$

(a) Fase atrativa (b) Fase repulsiva

Figura 5.10: Atração e repulsão (adaptado de (MUÑOZ, 2016)).

5.4 Filtros híbridos otimizados

Neste trabalho são propostos como *Filtros Híbridos Otimizados* aquelas combinações entre os filtros híbridos (EPF e UPF) descritos na seção 5.2 com o PSO e os métodos de adição de diversidade. Obtendo assim três abordagens altamente robustas, tais como: (a) Filtro de Partículas com PSO (PF-PSO), (b) Filtro de Partículas Estendido com PSO (EPF-PSO) e (c) Filtro de Partículas *Unscented* com PSO (UPF-PSO). Nessas abordagens, o PSO foi implementado após do passo do *resampling* a fim de melhorar ainda mais a qualidade das partículas e assim obter uma estimativa dos estados com menor incerteza, sendo essa abordagem apresentada em (ZHANG et al., 2008).

Tendo em conta que o PSO é uma metaheurística, a população das partículas não pode ser muito grande na procura do ponto ótimo, pois isto faria que vira-se um algoritmo de força bruta. Porém em (CHENG, 2012), os autores fizeram uma comparação entre o PF e PF-PSO para um sistema de deteção e localização de faixas com diferentes números de partículas (20:20:200), obtendo como resultado que as estimações do PF-PSO não vêm-se afetadas pelo número de partículas. No entanto, o custo computacional é alto para um conjunto de partículas maior.

5.4.1 Filtro de Partículas com PSO

O Filtro de Partículas com PSO (PF-PSO) foi proposto em (ZHANG et al., 2008). Este filtro implementa o PSO após do resampling do PF apresentado na seção 5.1.2. Além disso, as abordagens com os

métodos de adição de diversidade (OBL-PSO e AR-PSO) são implementadas da mesma forma que o PSO básico. Na linha 9 do pseudocódigo do PF-PSO apresentado no Algoritmo 12 mostra-se a implementação do PSO após do *resampling*.

Algoritmo 12 Filtro de partículas com PSO

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.

Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: $P_0^{i=1:N} = P$
3: **para** k = 1 : tf **faça**
4: $x_k^{i=1:N} \sim p(x_k|x_{k-1}^{i=1:N})$
5: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
6: $w_k^{sum} = \sum_{i=1}^{N} w_i$
7: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
8: $[x_k^{m=1:N}, w_k^{j=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
9: $[x_k^{j=1:N}]$ = PSO \ OBL-PSO \ AR-PSO $[x_k^{m=1:N}, x_{min}, x_{max}, z_k, R]$
10: $x_k^{i=1:N} = x_k^{j=1:N}$
11: $out_k = \sum_{i=1}^{N} x_k^i/N$
12: **fim para**

5.4.2 Filtro de Partículas Estendido com PSO

O Filtro de Partículas Estendido com PSO (EPF-PSO) é implementado a partir do EPF (vide seção 5.2.1) e da abordagem do PF-PSO apresentada em (ZHANG et al., 2008). Por outro lado, as abordagens com OBL-PSO e AR-PSO são implementadas da mesma maneira que o PSO básico, sendo este fato apresentado na linha 10 do pseudocódigo do Algoritmo 13.

Algoritmo 13 Filtro de partículas estendido com PSO

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.

Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: $P_0^{i=1:N} = P$
3: **para** k = 1 : tf **faça**
4: $[\hat{x}_k^{i=1:N}, \hat{P}_k^{i=1:N}]$ = EKF $[x_{k-1}^{i=1:N}, P_{k-1}^{i=1:N}]$
5: $x_k^{i=1:N} \sim \hat{x}_k^{i=1:N} + PDF(\hat{P}_k^{i=1:N})$
6: $w_k^{i=1:N} = p(z_k|x_k^{i=1:N})$
7: $w_k^{sum} = \sum_{i=1}^{N} w_i$
8: $w_k^{i=1:N} = w_k^{i=1:N}/w_k^{sum}$
9: $[x_k^{m=1:N}, w_k^{j=1:N}, P_k^{j=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
10: $[x_k^{j=1:N}]$ = PSO \ OBL-PSO \ AR-PSO $[x_k^{m=1:N}, x_{min}, x_{max}, z_k, R]$
11: $x_k^{i=1:N} = x_k^{j=1:N}$
12: $P_k^{i=1:N} = \hat{P}_k^{i=1:N}$

13: $out_k = \sum_{i=1}^{N} x_k^i / N$
14: **fim para**

5.4.3 Filtro de Partículas Unscented com PSO

O Filtro de Partículas *Unscented* com PSO (UPF-PSO) está baseado no UPF (vide seção 5.2.2). Ao igual, que no PF-PSO e no EPF-PSO, neste caso os métodos de adição de diversidade (OBL-PSO e AR-PSO) são levados em consideração com a implementação do PSO no PF, proposta em (ZHANG et al., 2008). Na linha 10 do pseudocódigo do UPF-PSO apresentado no Algoritmo 14 é detalhada essa implementação.

Algoritmo 14 Filtro de partículas unscented com PSO

Entrada: Estado inicial e número de partículas, x_0, N. Medição, z_k.
Saída: Média das partículas, out_k.

1: $x_0^{i=1:N} \sim x_0 + PDF(P)$
2: $P_0^{i=1:N} = P$
3: **para** k = 1 : tf **faça**
4: $[\hat{x}_k^{i=1:N}, \hat{P}_k^{i=1:N}]$ = UKF $[x_{k-1}^{i=1:N}, P_{k-1}^{i=1:N}]$
5: $x_k^{i=1:N} \sim \hat{x}_k^{i=1:N} + PDF(\hat{P}_k^{i=1:N})$
6: $w_k^{i=1:N} = p(z_k | x_k^{i=1:N})$
7: $w_k^{sum} = \sum_{i=1}^{N} w_i$
8: $w_k^{i=1:N} = w_k^{i=1:N} / w_k^{sum}$
9: $[x_k^{m=1:N}, w_k^{m=1:N}]$ = *Resampling* $[x_k^{i=1:N}, w_k^{i=1:N}]$
10: $[x_k^{j=1:N}]$ = PSO $\setminus$ OBL-PSO $\setminus$ AR-PSO $[x_k^{m=1:N}, x_{min}, x_{max}, z_k, R]$
11: $x_k^{i=1:N} = x_k^{j=1:N}$
12: $P_k^{i=1:N} = \hat{P}_k^{i=1:N}$
13: $out_k = \sum_{i=1}^{N} x_k^i / N$
14: **fim para**

5.5 Trabalhos correlatos sobre filtros híbridos baseados no PF

Na Tabela 5.1 apresentam-se alguns trabalhos correlatos dos filtros híbridos, baseados no PF para estimação de estados. Sendo que esses trabalhos são comparados levando em consideração alguns aspectos, tais como: (a) descrição, (b) algoritmos utilizados, (c) número de partículas tanto para o PF quanto para o PSO (se for o caso), (d) outros parâmetros (ruídos e covariâncias), (e) algoritmo que apresentou o melhor desempenho e (f) métrica de avaliação.

Tabela 5.1: Revisão bibliográfica dos trabalhos correlatos dos filtros híbridos para estimação de estados.

Ref.	Descrição	Algoritmos	Partículas (PF / PSO)	Outros parâmetros	Melhor desempenho	Métrica de avaliação
(MERWE et al., 2001)	Avaliação e comparação de um novo filtro de partículas proposto e baseado **no UKF na etapa de amostragem** para as partículas com um benchmark com PDF Gaussiana	EKF, UKF PF, EPF, UPF	200	α = 3 λ = 2	UPF	MSE
(ZHANG et al., 2008)	Estudo e avaliação de um novo algoritmo, no qual baseia-se na implementação **do PSO após do *resampling* no PF** em um benchmark com PDF Gaussiana.	PF PF-PSO	500 / -	P = 5 Q = 1 R = 10	PF-PSO	RMSE
(JING et al., 2009)	Avaliação de um novo filtro (UPF-PSO) através de dois *benchmarks* não lineares com duas PDFs (Gaussiana e Gama), onde **o PSO é modificado e implementado após do UKF.**	EKF, UKF EPF, UPF UPF-PSO	200 / -	P = 2 Q = 0.75 R = 0.01 α = 3 λ = 1	UPF-PSO	MSE
(LIN; SHENG, 2010)	Proposta e estudo de um algoritmo baseado na combinação entre o **PF e o EKF para multisensores** através de um benchmark com ruídos Gaussianos	PF EPF	50	P = 2 Q = 10 R1 = 1 R2 = 2	EPF	RMSE
(ZHAO; LI, 2010)	Estudo e proposta de um método de ***resampling* baseado no PSO** para o PF, aplicado a um problema de rastreamento de objetos por visão	PF PF-PSO	- / 10	-	PF-PSO	-

(CHENG, 2012)	Método de deteção e localização de faixas através câmeras e de uma combinação do PF e PSO, sendo **o PSO implementado após do *resampling* no PF.**	PF PF-PSO	50 / 50	-	PF-PSO	centímetros
(SOUIBGUI; HMIDA; CHAARI, 2013)	Estudo de filtros de partículas híbridos para a estimação de estados através de um benchmark com ruído Gaussiano, levando em consideração **o KF na etapa de amostragem do PF**.	PF EPF, UPF	200	Q = 0.01 R = 0.0001	UPF	RMSE
(LI et al., 2013)	Proposta e avaliação de um novo filtro, baseado na implementação **do PSO na etapa de amostragem e do algoritmo genético na etapa do *resampling***, tendo em conta um benchmark com ruídos Gaussianos	PF PF-PSO-GA	100	Q = 10 R = 1	PF-PSO-GA	RMSE
Este trabalho	Estudo de filtros híbridos entre o PF, o KF e o PSO para dois *benchmarks* não lineares com diferentes PDFs (Gaussiana e Gama), tendo em conta **métodos de adição de diversidade para o PSO**. Além **do KF e PSO serem implementados na amostragem e após do *resampling*, respectivamente.**	EKF, UKF PF, EPF, UPF PF-PSO PF-OBL-PSO PF-AR-PSO EPF-PSO EPF-OBL-PSO EPF-AR-PSO UPF-PSO UPF-OBL-PSO UPF-AR-PSO	50 / 50	P = 2 Q1 = 1 R1 = 0.1 Q2 = 0.01 R2 = 0.0001 α = 3 λ = 2	UPF-OBL-PSO	RMSE

Na Tabela 5.1 pode-se concluir que a maioria dos trabalhos correlatos, basearam-se em diferentes *benchmarks* a fim de avaliar o desempenho dos algoritmos propostos. Levando em consideração que tanto o PF quanto o PSO baseiam o seu funcionamento em populações de partículas; em alguns dos trabalhos referenciados foram definidas a mesma quantidade de partículas para esses dois algoritmos (PF e PSO). Outros parâmetros a ter em conta são os valores de covâriancia e os ruídos (Gaussianos, Gama, etc), pois uma pequena mudança neles afeta consideravelmente o desempenho dos filtros. Além disso, na mesma Tabela mostra-se o estimador que apresentou o melhor desempenho para cada caso, e a métrica que foi utilizada para avaliar os filtros de cada trabalho. É valido ressaltar que poucos trabalhos fizeram um estudo com a abordagem dos filtros híbridos otimizados, pois somente um trabalho combinou o PSO com o UPF. Nesse caso, o PSO foi implementado após do UKF, sendo essa abordagem totalmente diferente à apresentada neste trabalho (JING et al., 2009).

Capítulo 6

Resultados e análise

Neste capítulo inicialmente apresentam-se os processos de calibração para cada um dos sensores (giroscópio, acelerômetro e magnetômetro). Adicionalmente, mostram-se os resultados tanto da técnica de fusão sensorial com IMUs quanto da implementação em FPGA dessa técnica de fusão sensorial (vide seções 2 e 4). Finalmente, são apresentados os resultados obtidos pelos filtros híbridos otimizados, descritos no capítulo 5.

6.1 Calibração dos sensores da IMU

Para o estudo da estimação da posição (rolagem e arfagem) e orientação (guinada) das IMUs (MPU6050 e MPU9250), os testes foram realizados através da conexão entre o Arduino Nano e o Matlab R2015, a fim de realizar a leitura dos valores desses sensores e o processamento dos dados (filtragem), respectivamente. Essa conexão representa-se na Figura 6.1.

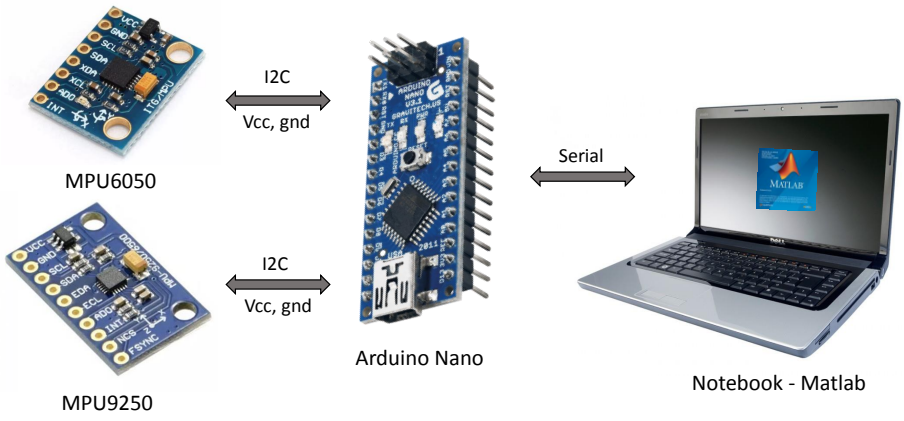


Figura 6.1: Conexões entre a IMU, o Arduino e o Matlab

Adicionalmente, nesta etapa do trabalho utilizou-se o equipamento projetado para esta fase do projeto, que consiste principalmente em: (a) bancada para calibrações e testes, (b) Arduino Nano, (c) *protoboard*, (d) transferidores, (e) nível de bolha (aplicação de *Android*) e (f) IMU de 6 ou 9 DOF (MPU6050 ou MPU9250). A Figura 6 representa esta plataforma de IMUs em conjunto com os outros componentes.

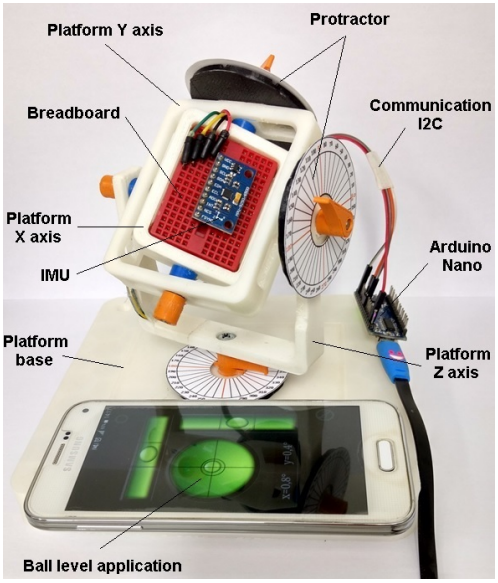


Figura 6.2: Plataforma com o Arduino

A resolução de medição da bancada (plataforma) com os transferidores análogos é de 10^o. Portanto, através da Equação 6.1 definida pela norma técnica ISO/TAG4/WG3 (ISO, 1992) é determinada a incerteza dessa bancada, sendo de $\pm 4^o$.

$$I_c = \sqrt{\frac{(2Re)^2}{24}} \tag{6.1}$$

Para as calibrações e os testes das IMUs, foi levado em consideração que a bancada móvel estivesse bem posicionada. Para conseguir isso, foi utilizado um aplicativo em *Android* de nível de bolha, no qual a inclinação 2D pode ser observada, tendo assim o controle sobre a posição da plataforma. Além disso, os transferidores foram usados para medir os ângulos de rotação das IMUs, onde esses ângulos foram considerados como valores verdadeiros. Portanto, essas medições da plataforma utilizaram-se para avaliar as estimativas dos ângulos de Euler fornecidas pela técnica de fusão sensorial de IMUs baseada no EKF.

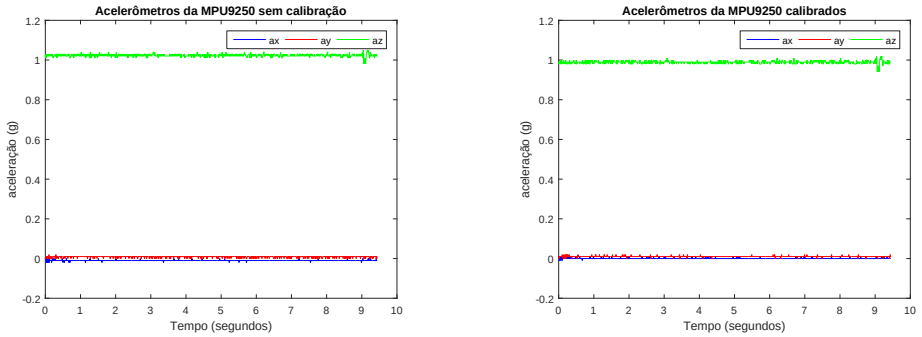
Cabe ressaltar que a plataforma não foi calibrada devido a que foi projetada com o objetivo principal de facilitar os movimentos da IMU durante os processos de calibração e testes dos sensores (giroscópio, acelerômetro e magnetômetro), e não para calibrar os sensores a partir da referencia da plataforma. Adicionalmente, com essa plataforma acontece o mesmo que em algumas aplicações onde são utilizadas IMUs (drones (Veículos Aéreos Não Tripulados) e os exoesqueletos instrumentados), onde a calibração dos sensores é feita manualmente; isto é, sem nenhum tipo de equipamento de calibração. Portanto, o efeito do erro propagado ao se usar a bancada, junto com um aplicativo de nível de bolha e um transferidor analógico, é mínimo e controlável, sendo o mais importante a técnica usada no processo de calibração. Além disso, é necessário realçar que o foco deste trabalho não consistiu em realizar a calibração da bancada (para servir de referencia na calibração e testes das IMUs). Ao contrário, o foco foi o desenvolvimento de uma arquitetura para o processamento da fusão sensorial de IMUs em tempo real.

Para os procedimentos de calibração, os intervalos dinâmicos de $\pm 250^o/seg$, $\pm$2g e $\pm 4800\mu$T foram utilizados para o giroscópio, o acelerômetro e o magnetômetro, respectivamente. Por outro lado, a cali-

bração foi realizada usando uma metodologia diferente para cada sensor. Para o giroscópio e acelerômetro foi realizada uma calibração que consistiu em manter os sensores estáticos, enquanto o magnetômetro do MPU9250 foi girado 360^o em cada um dos seus eixos durante o processo de calibração. No caso do giroscópio e acelerômetro, as IMUs foram colocadas em posições específicas com o eixo Z perpendicular ao plano terrestre horizontal. Neste processo de calibração, foram adquiridas 100 amostras em cada um dos 3 eixos (X, Y e Z), a partir das quais foram obtidos um valor máximo e mínimo para determinar o valor de deslocamento dos sensores, conforme indicado pela Equação 6.2.

$$offset_{x,y,z} = \frac{max_{x,y,z} + min_{x,y,z}}{2} \tag{6.2}$$

Na Figura 6.3 observam-se os resultados obtidos das medições no três eixos (X, Y e Z) do acelerômetro, antes e após do processo de calibração. Na mesma pode ser observado que a calibração faz um ajuste nas medições deste sensor. Isto pode ser visto na Figura 6.3(a) que mostra as medições do acelerômetro sem calibração, onde o eixo Z e os eixos X e Y medem valores um pouco distantes aos valores reais ($1g$ e $0g$). Neste caso, a Figura 6.3(b) apresenta as mesmas medições do acelerômetro (porém calibradas), onde detalha-se o "ajuste"feito pelo processo de calibração. Neste caso, os três eixos medem valores próximos aos valores reais.



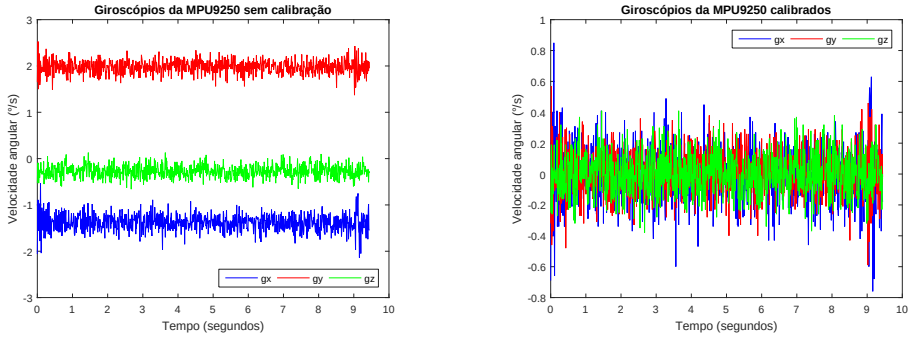
(a) Acelerômetro não calibrado

(b) Acelerômetro calibrado

Figura 6.3: Resultados de antes e após da calibração do acelerômetro da MPU9250

A Figura 6.4 apresenta os resultados da calibração nos três eixos (X, Y e Z) do giroscópio da IMU de 9 DOF (MPU9250). Neste caso, na Figura 6.4(a) mostram-se as medições desse sensor antes da calibração, na qual as medições têm um alto nível de incerteza (devido ao processo de fabricação de sensores de baixo custo). Apesar do giroscópio estar estático, os três eixos X, Y e Z medem valores distantes ao valor real ($0^o/s$); isto é corrigido através do processo de calibração. Na Figura 6.4(b) são apresentadas as medições calibradas do giroscópio, na qual tanto o eixo X quanto os eixos Y e Z do sensor possuem medições próximas ao valor real.

No caso do magnetômetro, a calibração consistiu primeiro na calibração dos eixos X e Y seguidamente da calibração do eixo Z, devido que cada eixo desse sensor deve ser calibrado tendo em conta o mesmo campo magnético; isto é, com respeito ao mesmo campo magnético de referência. Essa a razão pela qual os três eixos não podem ser calibrados ao mesmo tempo. Antes de realizar a calibração, os eixos do magnetômetro da IMU de 9 DOF foram ajustados de acordo com a Figura 2.2(c), pois o sentido e direção



(a) Giroscópio não calibrado

(b) Giroscópio calibrado

Figura 6.4: Resultados de antes e após da calibração do giroscópio da MPU9250

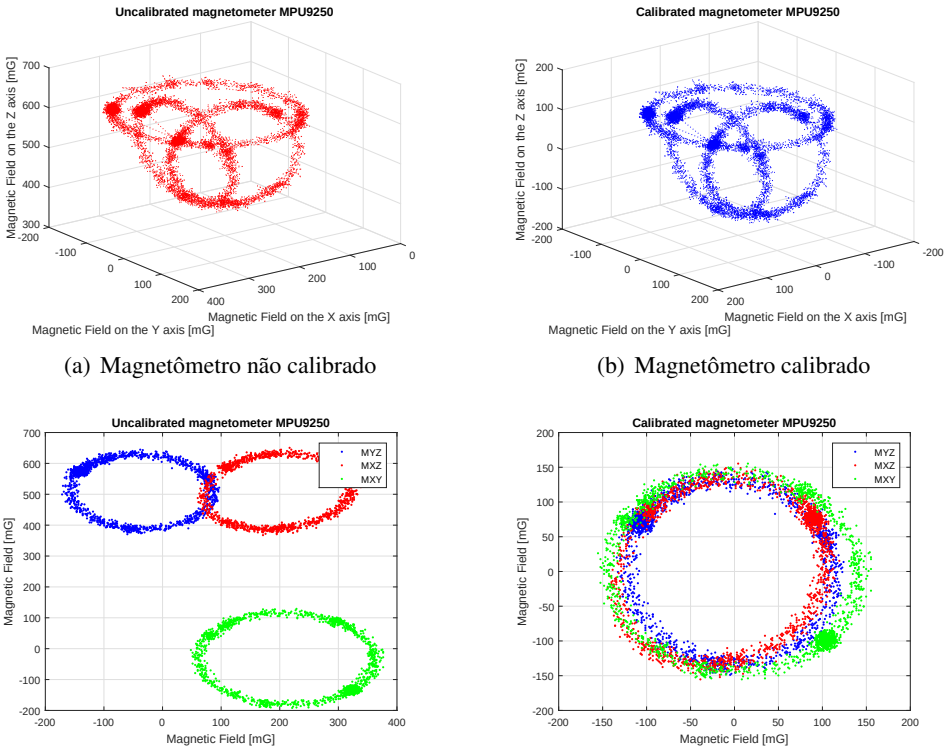
dos 3 eixos (X, Y e Z) desse sensor são diferentes em relação aos 3 eixos do giroscópio e do acelerômetro. Por isto, ao realizar a leitura das medições do magnetômetro se fez a mudança de atribuição de sentido e direção nos eixos X, Y e Z, o qual foi feita para que os 3 sensores tenham a mesma configuração em relação aos eixos do giroscópio e acelerômetro.

As calibrações para o magnetômetro basearam-se primeiro no posicionamento da IMU com o eixo Z perpendicular ao plano horizontal, girando a IMU em torno desse eixo 360^o a fim de medir o campo magnético terrestre; sendo feita assim a calibração dos eixos X e Y. Enquanto para a calibração do eixo Z, a IMU posicionou-se com o eixo Y perpendicular ao plano horizontal e em seguida foi girada 360^o em torno desse eixo. A duração de cada um dos dois processos de calibração foi de 20 segundos. Os valores máximos e mínimos das medições foram obtidos para determinar o *offset* (calibração *hard iron*) usando a Equação 6.2 e o fator de escala (calibração *soft iron*) usando a Equação 6.3.

$$scale_{x,y,z} = \frac{max_{x,y,z} - min_{x,y,z}}{2} \tag{6.3}$$

A Figura 6.5 mostra os resultados antes e após do procedimento de calibração do magnetômetro da MPU9250. Nas Figuras 6.5(a) e 6.5(b) detalham-se as visualizações tridimensionais de ambas as medições (não calibradas e calibradas), respectivamente. Ao comparar essas figuras, observa-se que as duas têm a mesma magnitude (forma), porém com diferentes proporções de posição e comprimento; sendo que as medições na Figura 6.5(a) são dados "inúteis"a serem aplicados, devido que as medições não estão centradas no ponto de referência (0,0,0) dos 3 eixos, pois as circunferências possuem medições diferentes. Em contraste, essas duas desvantagens não são observadas na Figura 6.5(b), já que durante o processo de calibração as medições ficaram melhor posicionadas e dimensionadas de acordo com o campo magnético medido. A Figura 6.5(c) e a Figura 6.5(d) mostram as mesmas medições do caso anterior em uma perspectiva 2D, portanto ao compará-las, a diferença do magnetômetro não calibrado e calibrado é mais fácil de observar. Na Figura 6.5(c) as medidas de cada eixo do magnetômetro estão muito distantes dos outros e são completamente desproporcionais, enquanto que na Figura 6.5(d) as medidas dos 3 eixos foram corretamente calibradas, centralizadas e dimensionadas.

Os valores de *offset* e escalas obtidos nos processos de calibração para o giróscopio, acelerômetro e magnetômetro devem ser levados em consideração para ajustar as medições de cada um desses sensores.



(a) Magnetômetro não calibrado

(b) Magnetômetro calibrado

(c) Magnetômetro não calibrado

(d) Magnetômetro calibrado

Figura 6.5: Resultados de antes e após da calibração do magnetômetro da MPU9250

Para isso, os valores de *offset* e escalas devem ser subtraídos das medidas "cruas"de cada sensor, sendo que esses valores referem-se às medições padrão fornecidas por cada sensor. Portanto, para as medições de saída do giroscópio nos três eixos e para as medições nos eixos X e Y do acelerômetro, os valores de saída estão definidos pela Equação 6.4. Enquanto ao eixo Z do acelerômetro, o valor de saída é determinado pela Equação 6.5. No caso do magnetômetro, através da Equação 6.6 determina-se o valor de saída para cada um dos três eixos desse sensor.

$$Out\ Value = \frac{Raw\ Value - Offset}{Sensitivity\ Scale} \tag{6.4}$$

$$Out\ Value = 1 + \frac{Raw\ Value - Offset}{Sensitivity\ Scale} \tag{6.5}$$

$$Out\ Value = \frac{(Raw\ Value - Offset) * Scale}{Sensitivity\ Scale} \tag{6.6}$$

6.2 Técnica de fusão sensorial

A Figura 6.6 apresenta a técnica de fusão sensorial que foi utilizada neste trabalho para as IMUs; onde pode ser visto que os ângulos de rolagem e arfagem são determinados através da combinação entre o

giróscopio e acelerômetro, sendo que o giroscópio foi considerado como o modelo do sistema e o acelerômetro como modelo de sensor (no algoritmo EKF). No caso do cálculo do ângulo de guinada, o giroscópio também foi usado como modelo do sistema, enquanto o magnetômetro foi usado como modelo de sensor. Portanto, para cada estimativa da posição e orientação um EKF foi desenvolvido e ajustado.

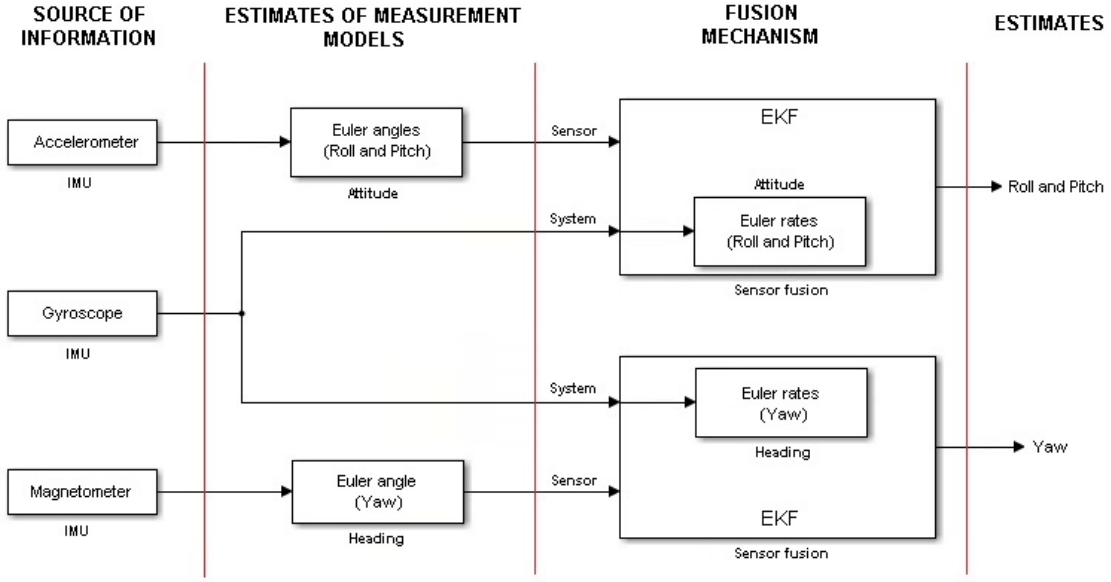


Figura 6.6: Fusão sensorial para as IMUs

A Tabela 6.1 apresenta os parâmetros selecionados para os dois módulos baseados no EKF. A seleção desses parâmetros consiste em estimar um valor inicial das variáveis de estado x_0 e da matriz de covariância P_0. Outro parâmetro a configurar é a matriz Jacobiana H, que está relacionada aos sensores disponíveis para cada variável de estado. Para obter um desempenho ótimo do EKF, os parâmetros Q e R devem ser ajustados corretamente; para o qual existem duas formas de ajuste. A primeira é o processo de prova e erro; isto é, mudar os valores dessas covariâncias até obter uma boa estimativa das variáveis de estado. Enquanto o segundo método é ajustar Q de acordo com a confiabilidade do modelo do processo, de modo que se for altamente confiável, um pequeno valor pode ser selecionado para este parâmetro. Para o parâmetro R, um conjunto de medidas do sensor deve ser analisado estatisticamente a fim de determinar a covariância do sensor. Outra opção, para determinar esses parâmetros é tomar uns valores já usados em casos semelhantes de fusão sensorial para tais sensores.

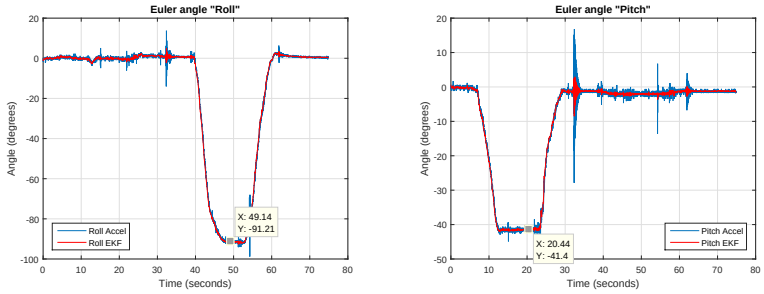
Este trabalho levou em consideração para o ruído do processo Q e para o ruído do sensor R valores que foram utilizados em um trabalho anterior de fusão sensorial, no qual determinou-se a posição com o filtro linear de Kalman (KF) (LAUSZUS, 2015). Tendo em conta que o EKF1 tem duas variáveis de estado (rolagem e arfagem), os parâmetros estão definidos em uma matriz de tamanho 2x2; enquanto no caso do EKF2, os parâmetros são escalares devido que está-se trabalhando com só uma variável de estado (guinada).

A Figura 6.7 ilustra os resultados obtidos para os ângulos de rolagem (vide Figura 6.7(a)) e de arfagem (vide Figura 6.7(b)) para a MPU6050. Enquanto a Figura 6.8 mostra os resultados obtidos da estimativa dos três ângulos de Euler para a IMU de 9 DOF (MPU9250), onde os ângulos de rolagem, arfagem e

Tabela 6.1: Parâmetros utilizados nos dois módulos baseados no EKF. No EKF1 os parâmetros estão definidos para duas variáveis de estado em uma matriz 2x2, enquanto o EKF2 trabalha com escalares para só uma variável de estado.

Parâmetro	Valores iniciais do EKF1	Valores iniciais do EKF2
x	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	0
P	$\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$	10
Q	$\begin{bmatrix} 0.001 & 0 \\ 0 & 0.001 \end{bmatrix}$	0.001
R	$\begin{bmatrix} 0.03 & 0 \\ 0 & 0.03 \end{bmatrix}$	0.03
H	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	1

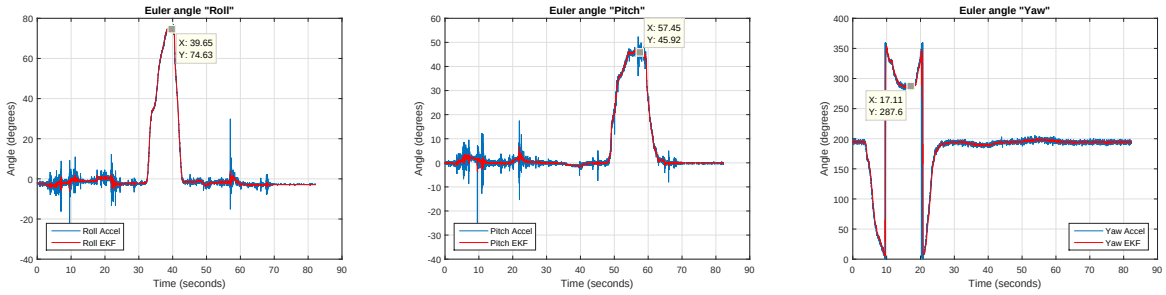
guinada detalham-se nas Figuras 6.8(a), 6.8(b) e 6.8(c), respectivamente. Levando em consideração esses resultados observa-se que o ruído foi eliminado no sinal filtrado (linha vermelha), o qual é comparado com os sinais dos sensores (acelerômetro ou magnetômetro) vistos na linha azul. Cabe ressaltar que esses resultados foram publicados em (PRIETO et al., 2017).



(a) Ângulo de rolagem (b) Ângulo de arfagem

Figura 6.7: Estimativa dos ângulos de Euler da MPU6050

A métrica utilizada para avaliar o desempenho dessa técnica de fusão de sensores foi o NRMSE (vide Equações 2.24 e 2.25) devido a sua vantagem para realizar comparações diretas com resultados de outros conjuntos de dados (com diferentes unidades de medida). Portanto, a Tabela 6.2 mostra os resultados NRMSE dos ângulos de Euler tanto para a MPU6050 quanto para a MPU9250; sendo que os erros foram calculados através da diferença entre os valores estimados pelos EKFs e os valores medidos na plataforma (bancada) para calibrações e testes para IMUs (vide Figura 6.2). Nessa tabela detalha-se a boa qualidade da estimação da posição e orientação. O ângulo de guinada para a MPU6050 não foi calculado devido que essa IMU não tem o magnetômetro, sendo assim possível determinar somente os ângulos de rolagem e arfagem.



(a) Ângulo de rolagem (b) Ângulo de arfagem (c) Ângulo de guinada

Figura 6.8: Estimativa dos ângulos de Euler da MPU9250

Tabela 6.2: Erros NRMSE para a MPU6050 e MPU9250

Ângulos de Euler	MPU6050	MPU9250
Rolagem	0.0163	0.07
Arfagem	0.0375	0.0882
Guinada	-	0.0095

6.3 Arquitetura embarcada em FPGA

Neste caso, a leitura das medições dos sensores da IMU de 9 DOF (MPU9250) e o processamento de fusão sensorial a fim de estimar os ângulos de Euler (filtragem), foram feitas através do Nios II (I2C em *software*) e da arquitetura embarcada no FPGA DE2-112 da Altera, respectivamente. Dita conexão pode ser vista na Figura 6.9, sendo que os resultados fornecidos pela arquitetura são visualizados através do Eclipse.

Figura 6.9: Conexões entre a IMU, o FPGA e o Eclipse

Na Figura 6.10 observa-se a montagem feita para realizar os testes a fim de estimar os ângulos de Euler com a arquitetura, apresentada no capítulo 4. Nesta montagem utilizou-se: (a) plataforma e calibração, (b) FPGA Altera DE2-115, (c) *protoboard*, (d) transferidores, (e) nível de bolha (aplicação de *Android*) e (f) IMU de 9 DOF (MPU9250).

Com o intuito de verificar a funcionalidade adequada da arquitetura, os resultados obtidos da mesma

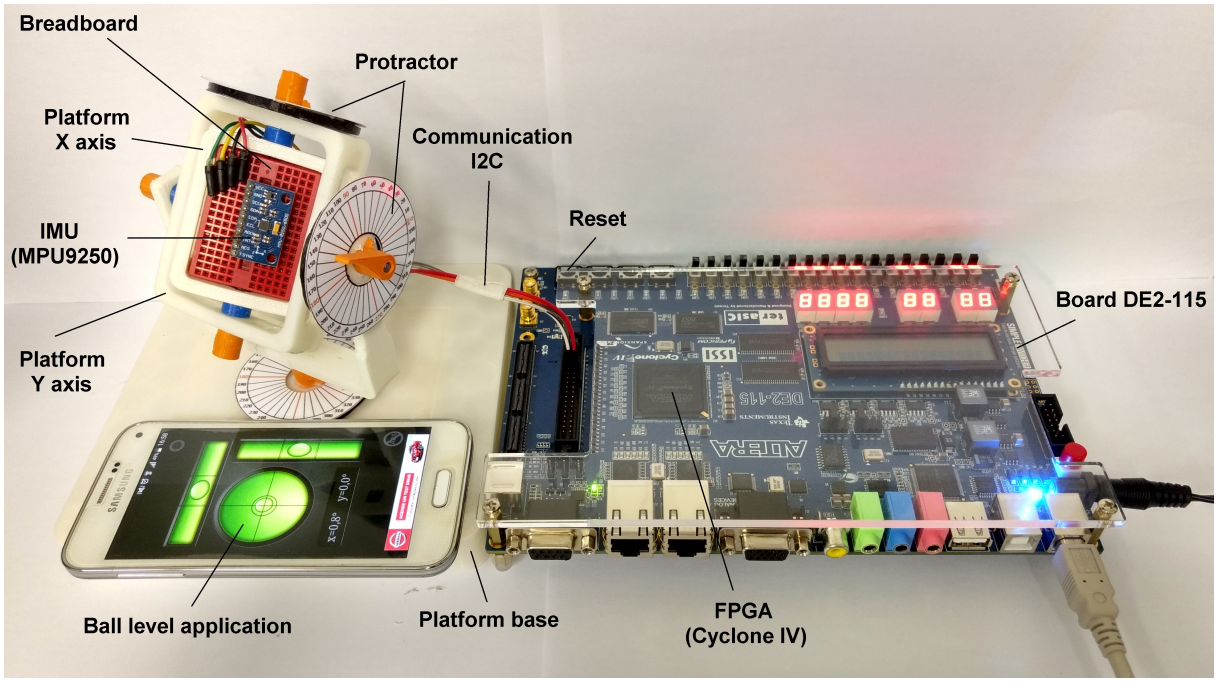


Figura 6.10: Plataforma com o FPGA

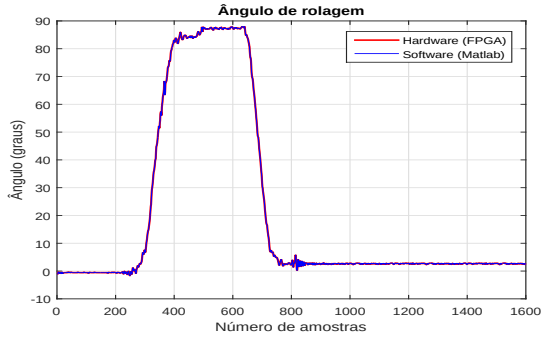
com uma representação de dados em "ponto flutuante"de 27 bits foram comparados com os resultados em Matlab do tipo "*double*"de 64 bits (vide Figura 6.11). Nas Figuras 6.11(a), 6.11(c) e 6.11(e) detalham-se as comparações dos ângulos girados em graus, levando em consideração a solução em *hardware* (FPGA) e em *software* (Matlab). Além disso, as Figuras 6.11(b), 6.11(d) e 6.11(f) mostram o erro entre essas duas abordagens (*hardware* e *software*). Cabe ressaltar que os valores dos parâmetros dos dois módulos baseados no EKF utilizados na arquitetura, foram iguais aos utilizados no estudo prévio (vide Tabela 6.1).

A Tabela 6.3 mostra os erros NRMSE para cada um dos ângulos de Euler levando em consideração os resultados da arquitetura e do Matlab, sendo esses da ordem de 10^{-5} e 10^{-6}. Portanto, pode-se concluir que os valores obtidos pela arquitetura têm alta precisão para a estimativa da posição e orientação, utilizando uma representação de ponto flutuante de 27 bits.

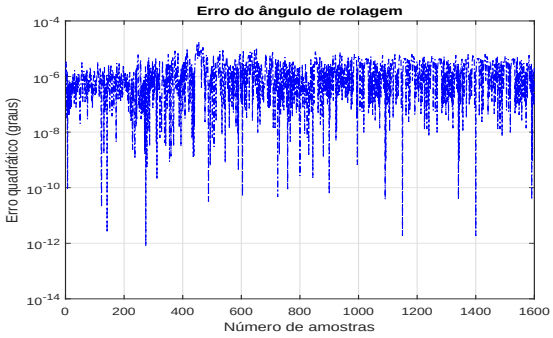
Tabela 6.3: Erros NRMSE dos ângulos de Euler (arquitetura FPGA e Matlab)

	Rolagem	Arfagem	Guinada
NRMSE	1.499E-5	8.241E-5	7.089E-6

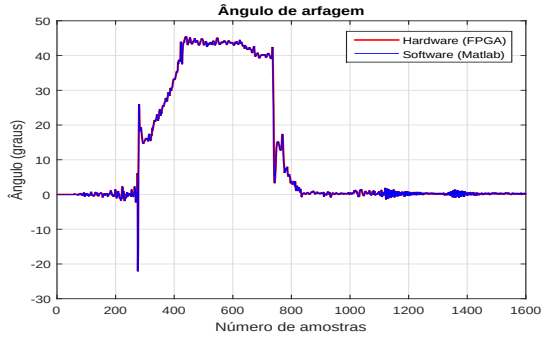
Os recursos utilizados para a arquitetura proposta no FPGA DE2-115 da Altera são apresentados na Tabela 6.4, na qual mostram-se diferentes características para cada módulo desenvolvido e implementado na arquitetura, tais como: (a) número de elementos lógicos, (b) multiplicadores embarcados (18x18 bits) e (c) frequência máxima alcançada. Além disso, mostra-se a equivalência em percentagem dos recursos utilizados com respeito ao total de recursos do FPGA, sendo que a arquitetura utiliza aproximadamente um total de 21.620% e 42.105% dos elementos lógicos e dos multiplicadores embarcados, respectivamente. Levando em consideração que cada módulo atinge uma frequência máxima diferente, a frequência de operação total da arquitetura (para fusão sensorial) é definida pela frequência mais baixa (atrelada ao conceito de *caminho crítico*, entre dois registradores). Portanto, a frequência máxima total é de 56,68



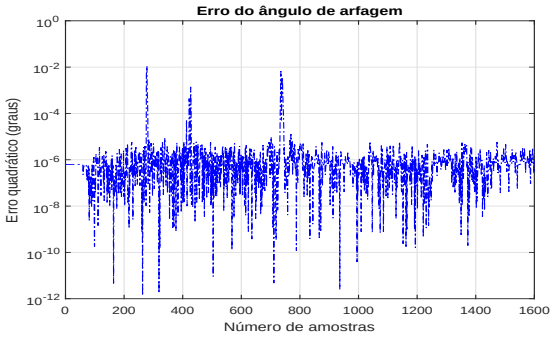
(a) Ângulo de rolagem no FPGA



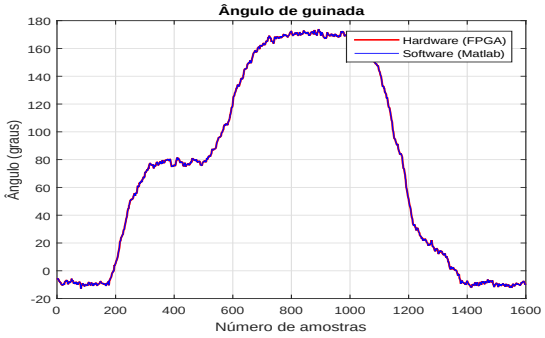
(b) Erro do ângulo de rolagem da arquitetura



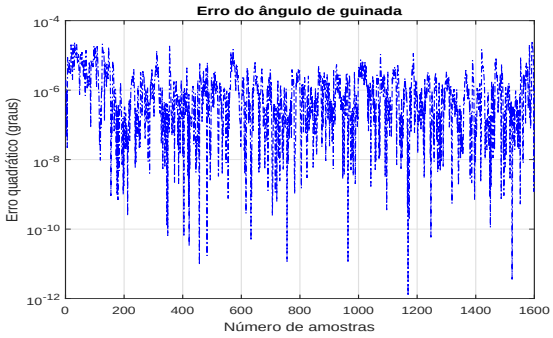
(c) Ângulo de arfagem no FPGA



(d) Erro do ângulo de arfagem da arquitetura



(e) Ângulo de guinada no FPGA



(f) Erro do ângulo de guinada da arquitetura

Figura 6.11: Erros da arquitetura no FPGA para os ângulos de rolagem, arfagem e guinada

MHz.

Tabela 6.4: Recursos utilizados para a arquitetura embarcada no FPGA

Módulos da arquitetura		Elementos lógicos	Multiplicadores embarcados	Frequência máxima (MHz)
Master		64 (0.056%)	0	133.01
preCordic		145 (0.126%)	0	81.75
EKF1		9.166 (8.006%)	0	61.14
EKF2		1.888 (1.649%)	0	79.76
Accelerometer		648 (0.566%)	0	81.61
Magnetometer		728 (0.636%)	0	71.68
Conversion (rad a deg)		253 (0.221%)	0	84.1
MUX1		1.934 (1.689%)	0	66.05
MUX2		28 (0.024%)	0	93.09
Bibliotecas em ponto flutuante (27 bits)	Cordic1	2.326 (2.032%)	7 (2.631%)	65.82
	Multipliers	1.369 (1.196%)	84 (31.579%)	75.0
	Adders	3.320 (2.900%)	0	62.79
	Subtractors	3.320 (2.900%)	0	60.83
	Dividers	718 (0.627%)	14 (5.263%)	71.31
	SQRT	330 (0.288%)	7 (2.631%)	78.86
	Cordic2	1.833 (1.601%)	0	73.42
TOTAL		24.750 (21.620%)	112 (42.105%)	56.68

A Tabela 6.5 compara o tempo de processamento (tempo e ciclos de latência) da arquitetura em FPGA baseada em EKF com um algoritmo equivalente executado em *software* no processador Nios II. Embora esse processador esteja funcionando duas vezes à frequência do *clock* do FPGA, o módulo de *hardware* ainda é 6023 vezes mais rápido. Os resultados obtidos mostram que a arquitetura proposta apresenta melhores resultados de desempenho (em termos de tempo de processamento) se comparado ao processador Nios II.

Tabela 6.5: Resultados de processamento para fusão sensorial com os EKFs

Dispositivo	Frequência de operação (MHz)	Tempo de latência (ms)	Ciclos de latência (clk)
DE2-115 (*hardware*)	50	0.01506	753
Nios II (*software*)	100	91.3097	9130970

6.4 Filtros híbridos otimizados

A fim de avaliar e comparar o desempenho dos diferentes filtros híbridos otimizados (apresentados no capítulo 5), entre os quais encontram-se: (1) EKF, (2) UKF, (3) PF, (4) EKF PF, (5) UKF PF, (6)

PF PSO, (7) EKF PF PSO, (8) UKF PF PSO, (9) PF OBLPSO (10) EKF PF OBLPSO, (11) UKF PF OBLPSO (12) PF ARPSO, (13) EKF PF ARPSO, (14) UKF PF ARPSO. Utilizaram-se dois sistemas não lineares (*benchmarks*) com diferentes características, os quais mostram-se nas Equações 6.7 e 6.9, com seus respectivos modelos de medição descritos nas Equações 6.8 e 6.10.

O *benchmark* 1 foi estudado em (GORDON; SALMOND; SMITH, 1993) para o caso do PF (vide seção 5.1.2), onde $w_k \sim N(0,1)$ e $v_k \sim N(0,0.1)$ são ruídos Gaussianos e independentes do processo e de medição, respectivamente.

$$x_k = 0.5x_{k-1} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 8\cos(1.2(k-1)) + w_k \tag{6.7}$$

$$y_k = \frac{x_k^2}{20} + v_k \tag{6.8}$$

O *benchmark* 2 foi analisado em (DOUCETH; WAN, 2000), o qual tem características diferentes ao *benchmark* 1, pois o ruído do processo segue uma PDF Gama $n_k \sim \Gamma(3,2)$, enquanto o ruído de medição segue uma PDF Gaussiana $m_k \sim N(0,0.0001)$. Neste caso, o modelo de medição está restrito para uma quantidade determinada de iterações (k).

$$x_k = 1 + \sin(0.04\pi k) + 0.5x_{k-1} + n_k) \tag{6.9}$$

$$y_k = \begin{cases} 0.2x_k^2 + m_k & se\ k \leq 30 \\ 0.5x_k - 2 + m_k & se\ k > 30 \end{cases} \tag{6.10}$$

Os quatro tipos de *resamplings* (*multinomial*, *residual*, *stratified* e *systematic*) foram tidos em conta para fazer a análise dos filtros híbridos otimizados. Os mesmos implementaram-se a partir dos códigos propostos em Matlab por (LI, 2014). A Figura 6.12 representa o funcionamento dos filtros híbridos otimizados. Inicialmente é definido um conjunto de partículas com características semelhantes à PDF. Posteriormente, para cada passo de tempo (k) executam-se as etapas de forma sequencial, sendo que a primeira etapa dentro desse ciclo repetitivo é a amostragem das partículas. A mesma é feita levando em consideração o modelo do processo e o filtro de Kalman a implementar (EKF ou UKF). Seguidamente, a segunda etapa consiste em determinar as probabilidades ou pesos de cada uma das partículas, a partir da função de distribuição. A terceira etapa é o *resampling* (vide seção 5.1.2.1), a qual resolve o problema da degeneração das partículas, no entanto gerando outro problema mais conhecido como o *empobrecimento da amostra*; pois o *resampling* baseia-se basicamente em multiplicar as partículas com grandes pesos, eliminando as de baixos pesos.

Além disso, na Figura 6.12 pode-se observar a abordagem da implementação do PSO após do *resampling*, a qual é realizada com o intuito de resolver o problema do empobrecimento da amostra, aumentando assim a diversidade das partículas. Essa é a quarta etapa e foi proposta em (ZHANG et al., 2008), onde as partículas de entrada do PSO são as partículas resultantes do *resampling*, assim como as partículas de saída são aquelas com os melhores *fitness*. Neste caso, a função da distribuição utilizada para calcular os pesos

das partículas é a função objetivo (unidimensional), a fim de selecionar as partículas com os pesos maiores. Portanto, trata-se de um problema de otimização (maximização) combinatória mono-objetivo, pois a ideia principal do PSO no PF é encontrar um conjunto de boas soluções (partículas com grandes pesos) dentro de uma variedade de possibilidades (combinações); levando em consideração a diversidade delas.

A quinta etapa baseia-se em definir o conjunto de partículas a posteriori, a partir das partículas fornecidas pelo PSO. A sexta e última etapa é o cálculo do valor estimado para cada passo de tempo, o qual neste caso é determinado através da média das partículas a posteriori.

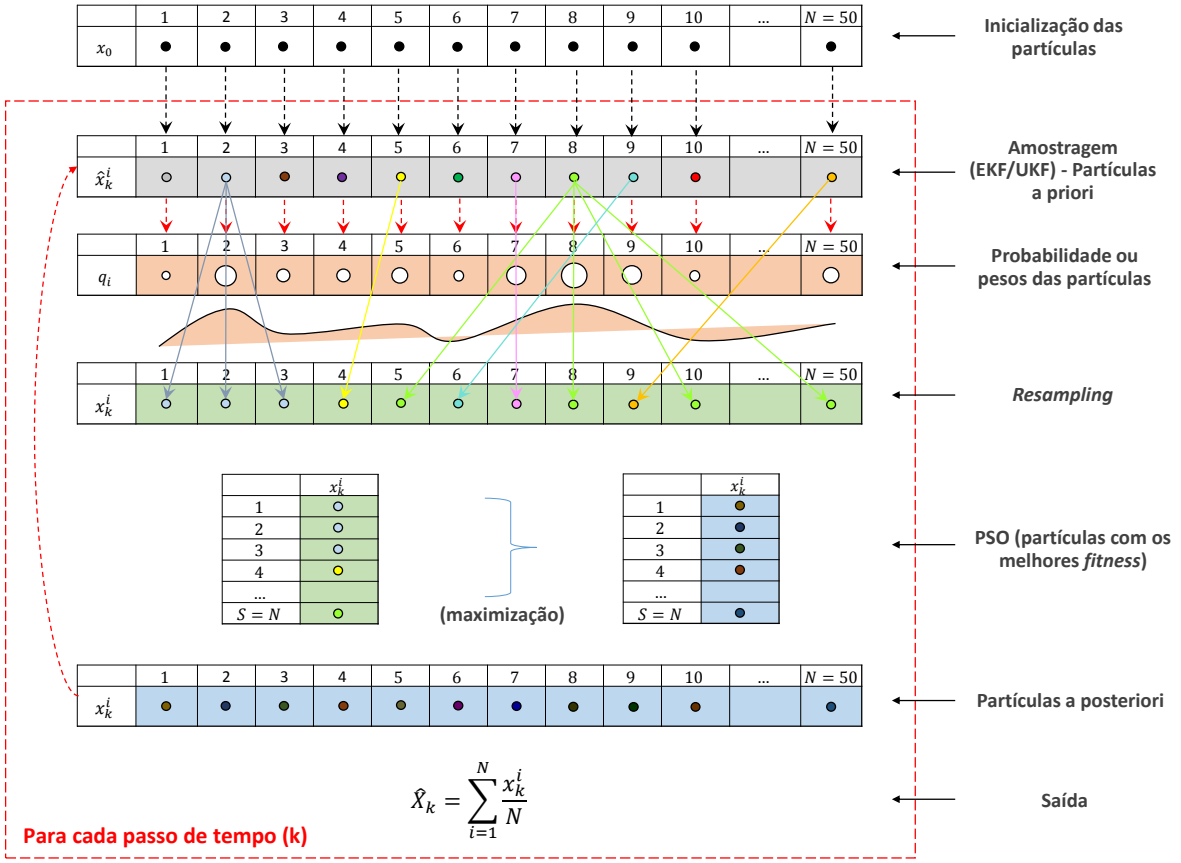
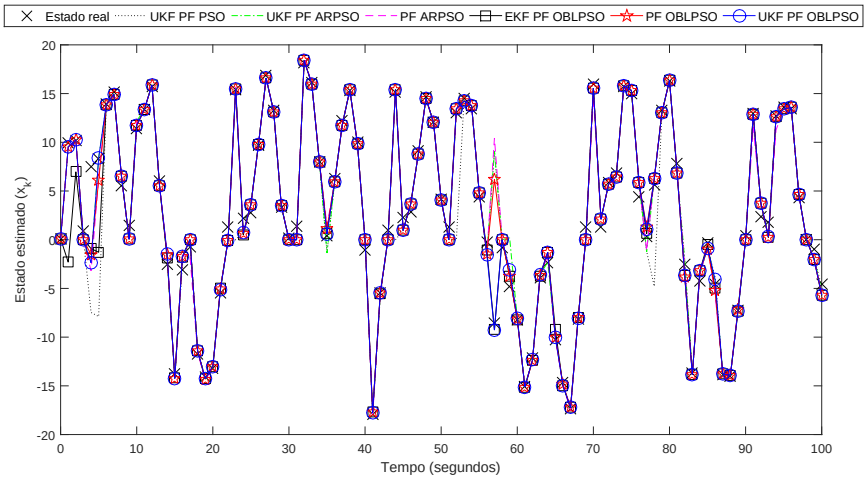


Figura 6.12: Funcionamento dos filtros híbridos otimizados

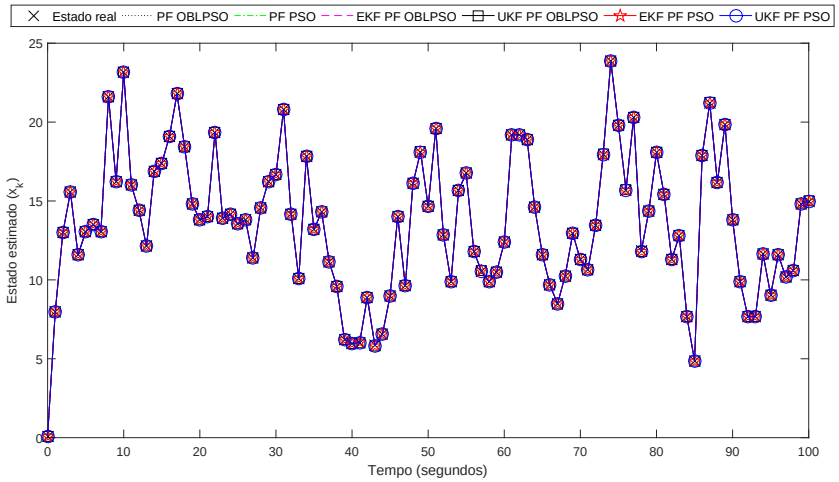
Os resultados da simulação feita para os filtros híbridos otimizados (vide Figura 6.13) tanto para o *benchmark* 1, quanto para o *benchmark* 2 mostram-se nas Figuras 6.13(a) e 6.13(b), respectivamente. Nas Figuras supracitadas se faz a comparação da estimação de estados dos filtros com respeito ao valor real (definido pelo modelo do processo mais um ruído). Para os dois casos de estudo (*benchmarks)* foram tidos em conta os seguintes parâmetros: (a) a simulação foi rodada 100 vezes, (b) todos os filtros de partículas utilizaram 50 partículas tanto no PF quanto no PSO, (c) o estado inicial foi $x_0 = 0.1$ e a covariância de estimativa do estado inicial foi $p(x_0) \sim N(0, 2)$. Adicionalmente, tiveram-se em conta os ruídos definidos anteriormente para o modelo do processo e do modelo de medição em cada um dos dois *benchmarks*.

Para os experimentos foram levados em consideração os seguintes parâmetros do PSO: (a) o conjunto de partículas tem a mesma quantidade que o PF $S = 50$, (b) a dimensionalidade dos problemas é $n = 1$

devido que tem só uma variável a ser estimada, (c) os limites do espaço de busca foram definidos a partir do estado real mais a PDF das partículas no PF ($x_{min} \sim x_k - PDF$ e $x_{max} \sim x_k + PDF$), (d) 250 iterações foram rodadas, (e) os coeficientes cognitivo e social foram $c_1 = 2.2$ e $c_2 = 1.9$, respectivamente e (f) a velocidade máxima das partículas definiu-se como o 10% do limite máximo $(0.1 \cdot x_{max})$.



(a) Estimação de estados para o benchmark 1 com o *resampling multinomial*



(b) Estimação de estados para o benchmark 2 com o *resampling multinomial*

Figura 6.13: Resultados da estimação de estado dos filtros híbridos otimizados para os benhcmarks 1 e 2

Com o intuito de fazer uma boa análise da estimação de estados dos filtros para cada um dos dois *benchmarks*, foram realizados 32 testes independentes, sendo que para cada teste determinou-se o erro RMSE (vide Equação 2.23). Portanto, como resultados finais calcularam-se a média, a mediana e o desvio padrão desses 32 RMSEs, os quais são apresentados nas Tabelas 6.6 e 6.7 para o *benchmark* 1 e *benchmark* 2, respectivamente. Neste caso, a mediana foi a métrica utilizada para determinar o filtro com a melhor estimativa de estados, devido a que a mediana é menos sensível (mais robusta) para avaliar amostras com dados heterogéneos (diferentes, desiguais ou desuniformes) do que a média. Isto é, o resultado da mediana é mais preciso para os dados que apresentam uma alta dispersão na distribuição, pois essa métrica não é afetada por todos os dados da amostra, assim como acontece no cálculo da média. Essa avaliação dos

filtros é representada através do *ranking*, no qual definem-se as posições de cada um dos catorze filtros estudados. Esses testes foram feitos levando em consideração cada um dos quatro algoritmos de *resampling* (*multinomial*, *residual*, *stratified* e *systematic*) tanto para o *benchmark* 1 quanto para o *benchmark* 2. Cabe ressaltar que as abordagens destacadas em azul correspondem aos filtros híbridos otimizados (PF OPSO, PS ARPSO, EKF PF PSO, EKF PF OPSO, EKF PF ARPSO, UKF PF PSO, UKF PF OPSO e UKF PF ARPSO) propostos neste trabalho.

Tabela 6.6: Resultados da média, mediana e desvio padrão dos 32 erros RMSE de cada filtro não linear para o *benchmark* 1

Resamples	Multinomial		Residual		Stratified		Systematic	
	Ranking	Média	Ranking	Média	Ranking	Média	Ranking	Média
		Mediana		Mediana		Mediana		Mediana
		Desvio padrão		Desvio padrão		Desvio padrão		Desvio padrão
EKF	14	26,4134	14	28,4283	14	26,121	14	23,563
		24,2605		**25,4429**		**22,119**		**22,2037**
		8,8856		20,8162		13,5889		10,2036
UKF	13	4,9358	13	4,6149	13	4,6589	13	4,4557
		4,9202		**4,4829**		**4,7012**		**4,4453**
		0,9017		0,9172		1,0248		0,9083
PF	12	4,8767	12	4,2895	12	4,449	12	3,6674
		4,7686		**4,3287**		**4,3492**		**3,4152**
		2,0441		1,478		1,3735		1,2226
PF PSO	7	2,7069	7	2,6125	7	2,4996	8	2,5609
		2,9462		**2,6357**		**2,7163**		**2,7241**
		0,8922		1,0781		0,9986		0,8432
PF OPSO	2	1,0733	2	1,1947	2	1,046	2	1,022
		0,8163		**0,9549**		**0,8526**		**0,8253**
		0,5535		0,6456		0,4934		0,4805
PF ARPSO	4	1,7195	4	1,4914	4	1,5481	4	1,429
		1,601		**1,5373**		**1,6207**		**1,4075**
		0,7945		0,6468		0,5791		0,6289
EKF PF	11	4,2909	11	3,9163	11	4,2017	11	3,6302
		3,7631		**3,7516**		**3,5653**		**3,3993**
		1,7606		1,0053		1,6478		0,7799
EKF PF PSO	9	3,7209	8	3,6178	8	3,1367	9	3,4131

		3,5163		**2,8234**		**2,7461**		**3,2446**
		1,3252		1,7075		1,4391		1,3738
EKF PF OPSO	3	1,5814	3	1,5402	3	1,3767	3	1,5982
		1,3263		**1,2984**		**1,2012**		**1,245**
		0,7801		0,7808		0,591		0,8327
EKF PF ARPSO	8	2,9533	9	2,8613	9	2,9073	7	2,9629
		2,9926		**2,8828**		**2,8144**		**2,6423**
		1,0757		0,9985		1,1595		1,2518
UKF PF	10	3,7496	10	3,7949	10	3,4039	10	3,6242
		3,7562		**3,7053**		**3,3246**		**3,396**
		0,7803		1,0042		0,8225		1,4594
UKF PF PSO	6	2,609	6	2,464	6	2,3115	6	2,3259
		2,5603		**2,4882**		**2,347**		**2,3893**
		0,9626		0,9108		0,8629		1,0258
UKF PF OPSO	1	0,8392	1	0,8256	1	1,0282	1	0,8828
		0,781		**0,7989**		**0,8511**		**0,7818**
		0,2673		0,1998		0,4651		0,3171
UKF PF ARPSO	5	2,0617	5	1,815	5	1,833	5	1,8476
		2,0295		**1,7205**		**1,8655**		**1,8034**
		1,0188		0,8338		0,7725		0,979

Tabela 6.7: Resultados da média, mediana e desvio padrão dos 32 erros RMSE de cada filtro não linear para o *benchmark* 2

Resamples	Multinomial		Residual		Stratified		Systematic	
	Ranking	Média	Ranking	Média	Ranking	Média	Ranking	Média
		Mediana		Mediana		Mediana		Mediana
		Desvio padrão		Desvio padrão		Desvio padrão		Desvio padrão

EKF	14	3,3208	14	2,5698	14	2,4979	14	3,2661
		2,6033		**2,5294**		**2,5161**		**2,7748**
		1,9177		0,5253		0,2863		1,499
UKF	13	2,799	13	2,9247	13	3,2318	13	2,5267
		2,5996		**2,4262**		**2,4583**		**2,4791**
		0,6066		1,285		3,0824		0,4857
PF	12	2,6041	12	2,6116	12	2,533	12	2,5418
		2,5238		**2,2996**		**2,3926**		**2,3133**
		0,483		0,8248		0,824		0,6048
PF PSO	4	0,0114	4	0,012	3	0,0115	3	0,0111
		0,0112		**0,0122**		**0,0114**		**0,011**
		0,0024		0,0017		0,002		0,0018
PF OPSO	6	0,0127	6	0,0135	6	0,0134	6	0,0129
		0,0122		**0,0136**		**0,0131**		**0,013**
		0,0022		0,0015		0,0021		0,0018
PF ARPSO	9	0,0455	9	0,0445	9	0,045	9	0,0458
		0,0456		**0,044**		**0,0455**		**0,0462**
		0,005		0,0052		0,0051		0,0047
EKF PF	11	2,4187	11	2,0826	11	1,9515	11	2,1675
		2,3542		**1,9886**		**1,9273**		**2,084**
		0,6615		0,4894		0,4008		0,5614
EKF PF PSO	2	0,0074	2	0,0091	2	0,0079	2	0,0088
		0,0074		**0,009**		**0,0078**		**0,0087**
		0,0019		0,0023		0,002		0,0022
EKF PF OPSO	5	0,0113	5	0,0125	5	0,0119	5	0,0119
		0,0112		**0,0125**		**0,012**		**0,0118**

		0,0016		0,0016		0,0014		0,0019
EKF PF ARPSO	8	2,1484	8	1,7785	8	1,372	7	2,1207
		0,0331		**0,0243**		**0,0266**		**0,0275**
		2,765		3,0419		2,6237		3,0092
UKF PF	10	2,2554	10	2,2214	10	1,9891	10	2,1872
		2,1149		**1,984**		**1,908**		**2,0318**
		0,6085		0,8497		0,4934		0,6689
UKF PF PSO	1	0,0044	1	0,0053	1	0,0051	1	0,0049
		0,004		**0,0057**		**0,0051**		**0,0048**
		0,0019		0,002		0,0022		0,0019
UKF PF OPSO	3	0,011	3	0,0122	4	0,0115	4	0,0115
		0,0109		**0,0121**		**0,0115**		**0,0116**
		0,0015		0,0016		0,0016		0,0019
UKF PF ARPSO	7	2,0611	7	1,3172	7	0,7556	8	2,0437
		0,0246		**0,0224**		**0,0225**		**0,0276**
		2,6332		2,5148		1,9329		3,0315

Tendo em conta os resultados apresentados nas Tabelas 6.6 e 6.7 dos *benchmarks* 1 e 2, respectivamente, observa-se que o melhor filtro para o primeiro caso de estudo (*benchmark* 1) foi o UKF PF OPSO. Sendo os *resamplings multinomial* e *systematic* os que tiveram uma melhor estimação de estados. No caso do *benchmark* 2, o UKF PF PSO com o *resampling multinomial* foi o filtro híbrido otimizado que teve uma melhor estimativa, se comparado ao estado real. Esses dois filtros foram os que apresentaram a menor mediana dos 32 erros RMSE obtidos nos testes feitos. Neste sentido, pode ser observado que cada *benchmark* foi representado por um filtro híbrido otimizado diferente como a melhor solução. Além disso, cabe ressaltar que os ruídos do processo e de medição são parâmetros que têm uma grande influência no desempenho dos filtros, portanto, em cada caso de estudo foram tidos em conta diferentes parâmetros para esses ruídos. Na Tabela 6.8 mostram-se as posições nas que ficaram cada um dos catorze filtros estudados na estimação de estados para os dois sistemas não lineares.

Tabela 6.8: Ranking dos filtros não lineares para o *benchmark* 1 e 2

Ranking	Benchmark 1	Benchmark 2
1	UKF PF OPSO	UKF PF PSO
2	PF OPSO	EKF PF PSO
3	EKF PF OPSO	UKF PF OPSO
4	PF ARPSO	PF PSO
5	UKF PF ARPSO	EKF PF OPSO
6	UKF PF PSO	PF OPSO
7	PF PSO	UKF PF ARPSO
8	EKF PF ARPSO	EKF PF ARPSO
9	EKF PF PSO	PF ARPSO
10	UKF PF	UKF PF
11	EKF PF	EKF PF
12	PF	PF
13	UKF	UKF
14	EKF	EKF

Da Tabela 6.8, pode-se concluir que as abordagens dos filtros híbridos otimizados com o método de Aprendizagem Baseada em Oposição (OBL) foram os que melhor desempenho (menor mediana dos 32 erros RMSE) tiveram para o *benchmark* 1. No entanto, no *benchmark* 2 nenhum dos métodos de adição de diversidade (OBL e AR) conseguiram melhorar a estimação de estados dos filtros. Adicionalmente, cabe ressaltar que no primeiro caso (*benchmark* 1) o PF OPSO teve uma melhor estimação do que o EKF PF OPSO. Porém, no segundo caso de estudo o EKF PF PSO teve melhor desempenho do que o PF PSO. Isto é devido a que os exemplos tratados são sistemas dinâmicos não lineares com diferentes características, pois o *benchmark* 1 tem maior grau de não linearidade do que o *benchmark* 2. Portanto, o EKF devido ao seu principio de linearidade através do Jacobiano é afetado para problemas altamente não lineares, o qual pode ser visto na Figura 5.1. Com respeito aos estimadores tradicionais (EKF, UKF, PF, EKF PF e UKF PF), para os dos sistemas não lineares tiveram desempenho similares. Outro fator relevante a ser ressaltado é que nos dois casos de estudo, os filtros híbridos otimizados propostos neste trabalho conseguiram atingir melhores resultados do que os filtros tradicionais para problemas de estimação de estados.

Capítulo 7

Conclusões e trabalhos futuros

Neste capítulo apresentam-se as conclusões e as propostas dos trabalhos futuros, levando em consideração a fusão sensorial para IMUs e os filtros híbridos otimizados.

7.1 Conclusões

Tendo em conta tanto as abordagens quanto as técnicas e os resultados apresentados nos capítulos desta dissertação, as respectivas conclusões são definidas a continuação:

- A calibração dos sensores MEMS de baixo custo é um quesito relevante no momento de utilizar as medições fornecidas por eles; pois um bom resultado do processamento da fusão sensorial depende de uma boa precisão das fontes de informação (sensores). Portanto, para as duas IMUs estudadas (MPU6050 e MPU9250) foram realizadas duas calibrações diferentes, uma calibração para o giroscópio e o acelerômetro e outra calibração para o magnetômetro, sendo que para esse sensor foram realizados dois tipos de calibrações (*hard iron* e *soft iron*). Isto é devido a que esse sensor é altamente sensitivo aos campos magnéticos externos. Portanto, os resultados obtidos das medições após das calibrações dos três sensores mostraram que os dados são confiáveis para o processamento da fusão sensorial, pois ditas medições ficaram mais perto do valor que cada sensor tinha que medir realmente. Isto levou a resultados satisfatórios na estimação da 'pose' tanto com a IMU de 6 DOF quanto com a IMU de 9 DOF.

- O EKF foi utilizado como mecanismo de fusão de sensores a fim de determinar a posição (rolagem e arfagem) e orientação (guinada) para duas IMUs (MPU6050 de 6 DOF e MPU9250 de 9 DOF). Neste caso, uma técnica flexível de fusão sensorial baseada em dois módulos EKFs teve-se em conta, a fim de estimar os ângulos de rolagem e arfagem através da combinação entre o giroscópio (sistema) e o acelerômetro (sensor) e o ângulo de guinada com o giroscópio (sistema) e o magnetômetro (sensor). A métrica NRMSE foi utilizada para avaliar o desempenho da 'pose' para cada uma das duas IMUs, levando em consideração tanto as estimações dos ângulos de Euler fornecidas pela fusão sensorial quanto os valores medidos (valores reais) em uma bancada (plataforma) móvel projetada para calibrações e testes desses tipos de sensores. Tais resultados NRMSE mostraram que a fusão

sensorial baseada no EKF tem boa qualidade para problemas de estimação da posição (*attitude*) e da orientação (*heading*) para as duas IMUs estudadas.

- Uma arquitetura reconfigurável em FPGA apresentou-se com o intuito de realizar uma estimativa em tempo real da posição (rolagem e arfagem) e orientação (guinada), através de uma técnica de fusão sensorial aplicada a uma IMU de 9 DOF com dois EKFs (o primeiro para os ângulos de rolagem e arfagem e o segundo para o ângulo de guinada). Os resultados mostraram que a arquitetura é uma proposta eficiente em relação à reutilização de recursos, já que compartilha os módulos operacionais para os cálculos, poupando assim recursos do FPGA. Portanto, os resultados NRMSE detalharam a alta precisão dos ângulos de Euler estimados através da arquitetura proposta ao serem comparados com as estimações no Matlab; sendo que a arquitetura em *hardware* opera com a representação de ponto flutuante de 27 bits enquanto que o Matlab usa tipo "*double*"de 64 bits. Além disso, a aceleração em *hardware* do processamento da fusão sensorial foi 6063 vezes mais rápida do que a execução no *software* (processador Nios II).

- A abordagem dos filtros híbridos otimizados baseados na combinação dos filtros não lineares (KF e PF) e do algoritmo bioinspirado (PSO), com dois dos métodos de adição de diversidade (OBL e AR) foram propostos com o intuito de melhorar a estimação de estados de sistemas não lineares. Para avaliar esses filtros levaram-se em conta dois sistemas dinâmicos não lineares (*benchmarks*) como casos de estudo. Tal avaliação foi feita através de 32 testes, nos quais calculou-se o RMSE correspondente para cada teste, sendo no final a mediana desses 32 erros a métrica com a qual determinou-se o desempenho de cada um dos catorze filtros híbridos otimizados. Portanto, os resultados mostraram que o UKF PF OPSO e o UKF PF PSO foram os que tiveram menor erro se comparado ao valor real, para o *benchmark* 1 e para o *benchmark* 2, respectivamente; levando em consideração diferentes parâmetros dos ruídos do processo e de medição para cada um dos casos de estudo.

- Levando em consideração os filtros híbridos otimizados, no primeiro caso de estudo (*benchmark* 1) o método OBL melhorou o desempenho dos filtros híbridos otimizados; no entanto no segundo caso (*benchmark* 2), nenhum dos métodos de adição de diversidade conseguiram melhorar a estimação de estados dos filtros. Isto é devido a que cada *benchmark* tem características não lineares diferentes. Portanto, pode-se concluir que os métodos de adição de diversidade são afetados dependendo do grau de não linearidade do sistema dinâmico estudado. Porém, cabe ressaltar que nos dois casos de estudo os filtros híbridos otimizados apresentaram melhor desempenho (estimativa mais perto do valor real) do que os filtros convencionais.

7.2 Propostas de trabalhos futuros

A partir do trabalho desenvolvido nesta dissertação, sugerem-se algumas propostas de trabalhos futuros que podem ser feitas com o intuito de avaliar e melhorar o desempenho da técnica de fusão sensorial e dos filtros híbridos otimizados.

- A bancada (plataforma) para as calibrações e testes de IMUs poderia ser calibrada com o intuito de desenvolver um equipamento com menor incerteza para medições da posição e da orientação.

- Com respeito à técnica de fusão sensorial apresentada (dois módulos EKF), testes com outras IMUs de 9 DOF poderiam ser realizados. Adicionalmente, poderia-se utilizar outro tipo de mecanismo de fusão sensorial (UKF, PF, entre outros), a fim de realizar uma análise e uma comparação dos resultados da estimativa de rolagem, arfagem e guinada obtidos neste trabalho.

- A solução de fusão sensorial poderia ser implementada em uma placa SoC ou em uma GPU, com o objetivo principal de comparar o tempo de latência obtido com a arquitetura apresentada.

- A arquitetura proposta poderia ser implementada e avaliada em aplicações que exigem uma abordagem multi-sensor onde é necessário ter conhecimento da posição e orientação, como é o caso de um exoesqueleto.

- Um estudo com diferentes números de partículas poderia ser feito tendo em conta os filtros híbridos otimizados apresentados neste trabalho, assim como um estudo mais aprofundado com um modelo não linear de um sistema real.

- O resultados obtidos na avaliação dos filtros abrem caminhos para as suas respectivas implementações em *hardware*, tal como foi feito no problema da fusão sensorial.

REFERÊNCIAS BIBLIOGRÁFICAS

ABDELFATAH, W. F. et al. FPGA-based real-time embedded system for RISS/GPS integrated navigation. *Sensors*, Molecular Diversity Preservation International, v. 12, n. 1, p. 115–147, 2011.

ABYARJOO, F. et al. Implementing a sensor fusion algorithm for 3D orientation detection with inertial/magnetic sensors. In: *Innovations and advances in computing, informatics, systems sciences, networking and engineering*. [S.l.]: Springer, 2015. p. 305–310.

ADDICORE. *Whats the Difference Between Pitch, Roll, and Yaw?* 2017. 18 Oct. 2017 <https://www.addicore.com/GY-521-MPU6050-p/170.htmehttps://www.addicore.com/mpu-9250-p/ad280.htm>.

ALTERA. *DE2-115 User Manual*. [S.l.]: Terasic Technologies Inc., 2010.

ALTERA. *Stratix Series FPGAs and SoCs*. 2017. 23 Fev. 2018 <https://www.altera.com/products/fpga/stratix-series.html>.

AMARA, A.; AMIEL, F.; EA, T. FPGA vs. ASIC for low power applications. *Microelectronics Journal*, Elsevier, v. 37, n. 8, p. 669–677, 2006.

BASHA, K. H. *Localization using IMU/GPS Sensors For Mobility Assistant for Visually Imparied System (MAVI)*. Tese (Doutorado) — Department of Electrical Engineering, Indian Institute of Technology Delhi, 2016.

BAVDEKAR, V. A.; DESHPANDE, A. P.; PATWARDHAN, S. C. Identification of process and measurement noise covariance for state and parameter estimation using extended kalman filter. *Journal of Process control*, Elsevier, v. 21, n. 4, p. 585–601, 2011.

BENINI, A.; MANCINI, A.; LONGHI, S. An IMU/UWB/vision-based extended Kalman filter for mini-UAV localization in indoor environment using 802.15. 4a wireless sensor network. *Journal of Intelligent & Robotic Systems*, Springer, p. 1–16, 2013.

BERGAMINI, E. et al. Estimating orientation using magnetic and inertial sensors and different sensor fusion approaches: Accuracy assessment in manual and locomotion tasks. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 14, n. 10, p. 18625–18649, 2014.

BERTEN. GPU vs FPGA performance comparison. *Digital Signal Processing*, v. 1, p. 1–4, 2016.

BIASI, S. C. de; GATTASS, M. Utilização de quatérnios para representação de rotações em 3-D. *Relatório técnico, TecGraf–Pontifícia Universidade Católica do Rio de Janeiro, PUCRIO. Disponível em <http://www.tecgraf.pucrio.br/~{}mgattass>*, 2007.

BLANCO, J. L. *Resampling Schemes*. 2017. 11 Oct. 2013 <https://www.mrpt.org/tutorials/programming/statistics-and-bayes-filtering/resampling_schemes/>.

BROWN, R. G.; HWANG, P. Y. *Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions*. [S.l.: s.n.], 1997.

BRUCKNER, H.-P.; SPINDELDREIER, C.; BLUME, H. Energy-efficient inertial sensor fusion on heterogeneous FPGA-fabric/RISC System on Chip. In: IEEE. *Sensing Technology (ICST), 2013 Seventh International Conference on*. [S.l.], 2013. p. 506–511.

CARON, F. et al. GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects. *Information fusion*, Elsevier, v. 7, n. 2, p. 221–230, 2006.

CHANDRASEKARAN, S. *I2C Protocol (2-Wire Interface) in a nut shell*. 2013. 30 Mar. 2018 <https://embedjournal.com/two-wire-interface-i2c-protocol-in-a-nut-shell/>.

CHAPPELL, S. et al. Exploiting real-time FPGA based adaptive systems technology for real-time sensor fusion in next generation automotive safety systems. IET, 2006.

CHENG, W.-C. PSO algorithm particle filters for improving the performance of lane detection and tracking systems in difficult roads. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 12, n. 12, p. 17168–17185, 2012.

DOUC, R.; CAPPÉ, O. Comparison of resampling schemes for particle filtering. In: IEEE. *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*. [S.l.], 2005. p. 64–69.

DOUCET, A. *Sequential Monte Carlo methods in practice/Arnaud Doucet, Nando de Freitas, Neil Gordon, editors; foreword by Adrian Smith*. [S.l.]: Springer, New York London:, 2001.

DOUCET, A.; JOHANSEN, A. M. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, v. 12, n. 656-704, p. 3, 2009.

DOUCETH, R. v. d. M. A.; WAN, N. d. F. E. THE UNSCENTED PARTICLE FILTER. 2000.

DUTCH. *Hardware and Project Selection Part 1 - CPU vs GPU*. 2017. 7 Fev. 2018 <https://steemit.com/gridcoin/@dutch/hardware-and-project-selection-part-1-cpu-vs-gpu>.

EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. In: IEEE. *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. [S.l.], 1995. p. 39–43.

EINSPRUCH, N. *Application specific integrated circuit (ASIC) technology*. [S.l.]: Academic Press, 2012. v. 23.

FAROOQ, U.; MARRAKCHI, Z.; MEHREZ, H. FPGA architectures: An overview. In: *Tree-based Heterogeneous FPGA Architectures*. [S.l.]: Springer, 2012. p. 7–48.

GALLAS, M. R. Incerteza de medição. *Texto baseado no Guia Para a Expressão da Incerteza de Medição, 2a edição, ABNT, INMETRO*, 1998.

GORDON, N. J.; SALMOND, D. J.; SMITH, A. F. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: IET. *IEE Proceedings F (Radar and Signal Processing)*. [S.l.], 1993. v. 140, n. 2, p. 107–113.

GREGG, C.; HAZELWOOD, K. Where is the data? why you cannot debate CPU vs. GPU performance without the answer. In: IEEE. *Performance Analysis of Systems and Software (ISPASS), 2011 IEEE International Symposium on*. [S.l.], 2011. p. 134–144.

HAJDU, S.; BRASSAI, S. T.; SZEKELY, I. Complementary filter based sensor fusion on FPGA platforms. In: IEEE. *Optimization of Electrical and Electronic Equipment (OPTIM) & 2017 Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP), 2017 International Conference on*. [S.l.], 2017. p. 851–856.

HASSAN, M. et al. Wearable gait measurement system with an instrumented cane for exoskeleton control. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 14, n. 1, p. 1705–1722, 2014.

HAZANCHUK, A. Soft multipliers for DSP applications. In: *GSPx Conference, April*. [S.l.: s.n.], 2003.

HIGGINS, W. T. A comparison of complementary and kalman filtering. *IEEE Transactions on Aerospace and Electronic Systems*, IEEE, n. 3, p. 321–325, 1975.

IEEE. Ieee standard for floating-point arithmetic. *IEEE Std 754-2008*, p. 1–70, Aug 2008.

INSTRUMENTS, N. *Fundamentos da tecnologia FPGA*. 2013. 19 Nov. 2017 <http://www.ni.com/white-paper/6983/pt/>.

INVENSENSE. *MPU-6000 and MPU-6050 Product Specification*. [S.l.]: Technology Drive, 2013.

INVENSENSE. *MPU-9250 Product Specification*. [S.l.]: Technology Drive, 2016.

ISO. *Guide to the Expression of Uncertainty in Measurement*. [S.l.], 1992.

JABEEN, H.; JALIL, Z.; BAIG, A. R. Opposition based initialization in particle swarm optimization (O-PSO). In: ACM. *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*. [S.l.], 2009. p. 2047–2052.

JING, W. et al. A optimized particle filter based on PSO algorithm. In: IEEE. *BioMedical Information Engineering, 2009. FBIE 2009. International Conference on Future*. [S.l.], 2009. p. 122–125.

JR, L. E. F. *Fundamentos de Comunicação Eletrônica-Volume 1*. [S.l.]: AMGH Editora, 2009.

JULIER, S. J.; UHLMANN, J. K. New extension of the Kalman filter to nonlinear systems. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Signal processing, sensor fusion, and target recognition VI*. [S.l.], 1997. v. 3068, p. 182–194.

KAESLIN, H. *Digital integrated circuit design: from VLSI architectures to CMOS fabrication*. [S.l.]: Cambridge University Press, 2008.

KENNEDY, J. Swarm intelligence. In: *Handbook of nature-inspired and innovative computing*. [S.l.]: Springer, 2006. p. 187–219.

KIM, P. *Kalman filter for beginners: with MATLAB examples*. [S.l.]: CreateSpace, 2011.

KRAFT, M. Micromachined inertial sensors: The state-of-the-art and a look into the future. *Measurement and Control*, SAGE Publications Sage UK: London, England, v. 33, n. 6, p. 164–168, 2000.

LAUSZUS, K. *KalmanFilter*. 2015. 13 Apr. 2017 <https://github.com/TKJElectronics/KalmanFilter/blob/master/Kalman.cpp>.

LENART, T. *Design of reconfigurable hardware architectures for real-time applications*. [S.l.]: Thomas Lenart, 2008. v. 5.

LI, M. et al. Particle Filter Improved by Genetic Algorithm and Particle Swarm Optimization Algorithm. *JSW*, v. 8, n. 3, p. 666–672, 2013.

LI, T. *Resampling codes for PF*. 2014. 20 Nov. 2017 <https://sites.google.com/site/tianchengli85/matlab-codes/resampling-methods>.

LI, T.; BOLIC, M.; DJURIC, P. M. Resampling methods for particle filtering: classification, implementation, and strategies. *IEEE Signal Processing Magazine*, IEEE, v. 32, n. 3, p. 70–86, 2015.

LIGORIO, G.; SABATINI, A. M. Extended Kalman filter-based methods for pose estimation using visual, inertial and magnetic sensors: Comparative analysis and performance evaluation. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 13, n. 2, p. 1919–1941, 2013.

LIN, M.; SHENG, L. Multi-sensor information fusion extended Kalman particle filter. In: IEEE. *Advanced Computer Control (ICACC), 2010 2nd International Conference on*. [S.l.], 2010. v. 4, p. 417–419.

LIU, Y.; NOGUCHI, N.; ISHII, K. Development of a low-cost IMU by using sensor fusion for attitude angle estimation. *IFAC Proceedings Volumes*, Elsevier, v. 47, n. 3, p. 4435–4440, 2014.

LLANOS, C. H. Filtros complementares. Unpublished. 2016.

LLANOS, C. H. Filtro de particulas. Unpublished. 2017.

LOU, L. et al. Sensor fusion-based attitude estimation using low-cost MEMS-IMU for mobile robot navigation. In: IEEE. *Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International*. [S.l.], 2011. v. 2, p. 465–468.

MAXFIELD, C. *The design warrior's guide to FPGAs: devices, tools and flows*. [S.l.]: Elsevier, 2004.

MERWE, R. V. D. et al. The unscented particle filter. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2001. p. 584–590.

MOSQUERA, O. E. A. *Implementação do algoritmo de Richardson-Lucy em arquiteturas reconfiguráveis aplicado ao problema de borramento de imagens*. Dissertação (Mestrado) — Universidade de Brasília, 2016.

MUÑOZ, D. *Otimização por Inteligência de Enxames Usando Arquiteturas Paralelas para Aplicações Embarcadas*. Tese (Doutorado) — Universidade de Brasília, 2012.

MUÑOZ, D. M. Aula 9 - Métodos de Adição de Diversidade Artificial. Unpublished. 2016.

MUÑOZ, D. M. et al. FPGA based floating-point library for CORDIC algorithms. In: IEEE. *Programmable Logic Conference (SPL), 2010 VI Southern*. [S.l.], 2010. p. 55–60.

NAWRAT, A. et al. Inertial navigation systems and its practical applications. In: *New approach of indoor and outdoor localization systems*. [S.l.]: InTech, 2012.

NEUMANN, J. V. *The computer and the brain*. [S.l.]: Yale University Press, 2012.

NXP. NXP Sensor Fusion Library for Kinetis MCUs. *DATA SHEET: PRODUCT PREVIEW, rev*, v. 8, 2016.

OZYAGCILAR, T. Implementing a Tilt-Compensated eCompass using Accelerometer and Magnetometer Sensors. *Freescale Semiconductor Application Note, rev*, v. 3, 2015.

PASCUAL, A. EKF y UKF: dos extensiones del filtro de Kalman para sistemas no lineales aplicadas al control de un péndulo invertido. *Monografía para el curso: Tratamiento Estadístico de Señales*, p. 35, 2004.

PATTERSON, D. A.; HENNESSY, J. L. *Computer Organization and Design*. [S.l.: s.n.], 1994.

PEDRONI, V. A. *Circuit design with VHDL*. [S.l.]: MIT press, 2004.

PHILIPS, N. S. *I2C Manual, AN10216-01*. 2003.

PRIETO, F. B. et al. A Study of Attitude and Heading Determination through an EKF-based Sensor Fusion for Inertial Measurement Units (IMUS). *COBEM*, ABCM International Congress of Mechanical Engineering. Curitiba, Brasil, n. 24, 2017.

RAFAEL, F. d. C. C. *Padrão IEEE 754 para números de Ponto Flutuante*. 2014. 12 Mar. 2018 <https://www.slideserve.com/vita/padr-o-ieee-754-para-n-meros-de-ponto-flutuante>.

RANGANATHAN, P.; ADVE, S.; JOUPPI, N. P. Performance of image and video processing with general-purpose processors and media ISA extensions. In: IEEE COMPUTER SOCIETY. *ACM SIGARCH Computer Architecture News*. [S.l.], 1999. v. 27, n. 2, p. 124–135.

RIGET, J.; VESTERSTRØM, J. S. A diversity-guided particle swarm optimizer-the ARPSO. *Dept. Comput. Sci., Univ. of Aarhus, Aarhus, Denmark, Tech. Rep*, v. 2, p. 2002, 2002.

RISTIC, B.; ARULAMPALAM, S.; GORDON, N. *Beyond the Kalman filter: Particle filters for tracking applications*. [S.l.]: Artech house, 2003.

SABATELLI, S. et al. A double stage Kalman filter for sensor fusion and orientation tracking in 9D IMU. In: IEEE. *Sensors Applications Symposium (SAS), 2012 IEEE*. [S.l.], 2012. p. 1–5.

SACCO, M. *Sensores Inerciales: El mundo en movimiento*. 2011. 1 Aug. 2017 <http://www.neoteo.com/21690-sensores-inerciales-el-mundo-en-movimiento>.

SALAZAR, J. Procesadores digitales de señal (DSP). *Mundo electrónico*, CETISA, n. 314, p. 46–57, 2000.

SÁNCHEZ, D. F. et al. Parameterizable floating-point library for arithmetic operations in FPGAs. In: ACM. *Proceedings of the 22nd Annual Symposium on Integrated Circuits and System Design: Chip on the Dunes*. [S.l.], 2009. p. 40.

SCHOPP, P. et al. Sensor fusion algorithm and calibration for a gyroscope-free IMU. *Procedia Chemistry*, Elsevier, v. 1, n. 1, p. 1323–1326, 2009.

SEMICONDUCTORS, P. The I2C-bus specification. *Philips Semiconductors*, v. 9397, n. 750, p. 00954, 2000.

SIDIBÉ, D. *Particle Filters and Applications in Computer Vision*. 2011. 24 Nov. 2017 <https://www.slideshare.net/zukun/particle-filters-and-applications-in-computer-vision>.

SIMON, D. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. [S.l.]: John Wiley & Sons, 2006.

SOUIBGUI, F.; HMIDA, F. B.; CHAARI, A. Bayesian estimation via extended and unscented Kalman particle filtering for non linear stochastic systems. In: IEEE. *Sciences and Techniques of Automatic Control and Computer Engineering (STA), 2013 14th International Conference on*. [S.l.], 2013. p. 89–95.

STEPHEN, M. *Whats the Difference Between Pitch, Roll, and Yaw?* 2014. 6 Sep. 2017 <http://www.machinedesign.com/engineering-essentials/what-s-difference-between-pitch-roll-and-yaw>.

TIZHOOSH, H. R. Opposition-based learning: a new scheme for machine intelligence. In: IEEE. *Computational intelligence for modelling, control and automation, 2005 and international conference on intelligent agents, web technologies and internet commerce, international conference on*. [S.l.], 2005. v. 1, p. 695–701.

TURNER, L.; SHERLOCK, C. *An introduction to particle filtering*. [S.l.]: Mayıs, 2013.

VAHID, F.; GIVARGIS, T. *Embedded System Design: A Unified Hardward Software Introdution*. [S.l.]: John Wiley, 2002.

VAUGHAN, C. L. et al. *Dynamics of human gait*. [S.l.]: Kiboho Publishers, 1999.

WAN, E. A.; MERWE, R. V. D. The unscented Kalman filter for nonlinear estimation. In: IEEE. *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*. [S.l.], 2000. p. 153–158.

WELCH, G.; BISHOP, G. *The Discrete Kalman Filter*. 2004. 22 Mar. 2017 <http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/WELCH/kalman.1.html#pgfId=11839>.

WESTON, J.; TITTERTON, D. Modern inertial navigation technology and its application. *Electronics & Communication Engineering Journal*, IET, v. 12, n. 2, p. 49–64, 2000.

WINER, K. *Affordable 9 DoF Sensor Fusion*. [S.l.]: GitHub, 2016. <https://github.com/kriswiner/MPU6050/wiki/affordable-9-dof-sensor-fusion>.

WINER, K. *Simple and Effective Magnetometer Calibration*. [S.l.]: GitHub, 2017. <https://github.com/kriswiner/MPU6050/wiki/Simple-and-Effective-Magnetometer-Calibration>.

XILINX. *Virtex UltraScale and Product Table*. 2017. 23 Fev. 2018 <https://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale-plus.html#productTable>.

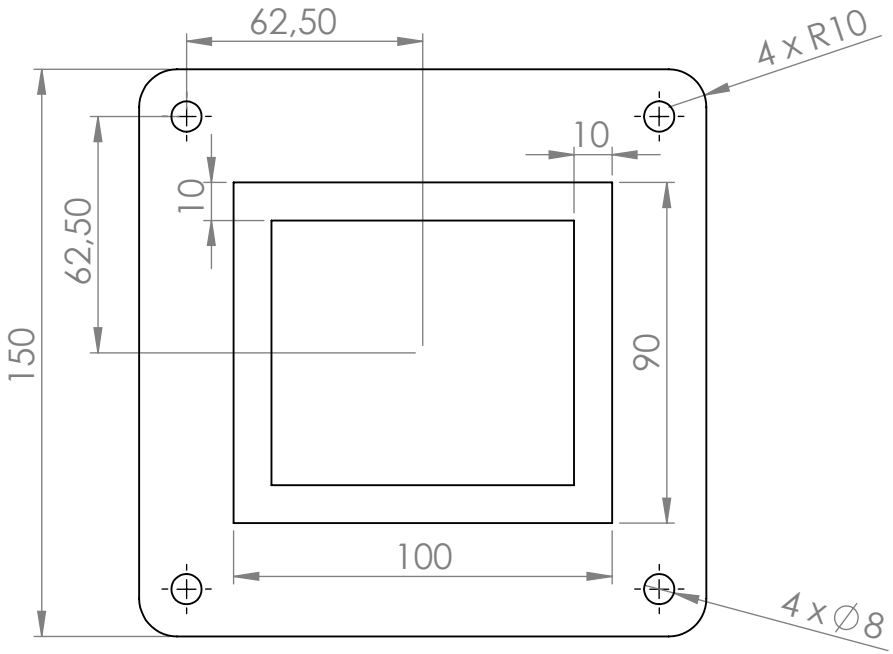
YOLE, D. *2016 Top MEMS manufacturers - In US$ million*. 2017. 6 Aug. 2017 <http://www.yole.fr/iso_album/illus_memsmanufacturers_2016memsranking_yole_may2017.png>.

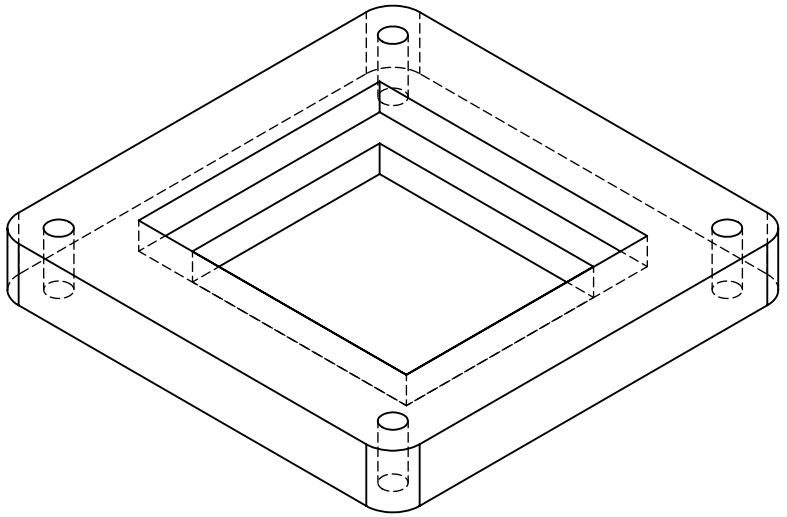
ZHANG, G. et al. Particle filter based on PSO. In: IEEE. *Intelligent Computation Technology and Automation (ICICTA), 2008 International Conference on*. [S.l.], 2008. v. 1, p. 121–124.

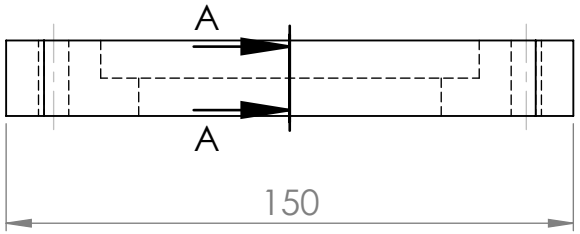
ZHAO, H.; WANG, Z. Motion measurement using inertial sensors, ultrasonic sensors, and magnetometers with extended Kalman filter for data fusion. *IEEE Sensors Journal*, IEEE, v. 12, n. 5, p. 943–953, 2012.

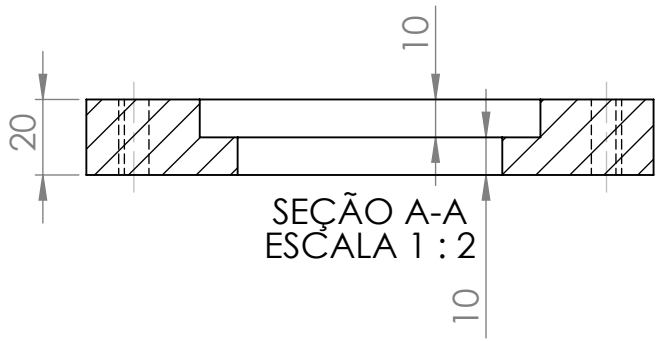
ZHAO, J.; LI, Z. Particle filter based on Particle Swarm Optimization resampling for vision tracking. *Expert Systems with Applications*, Elsevier, v. 37, n. 12, p. 8910–8914, 2010.

ANEXOS



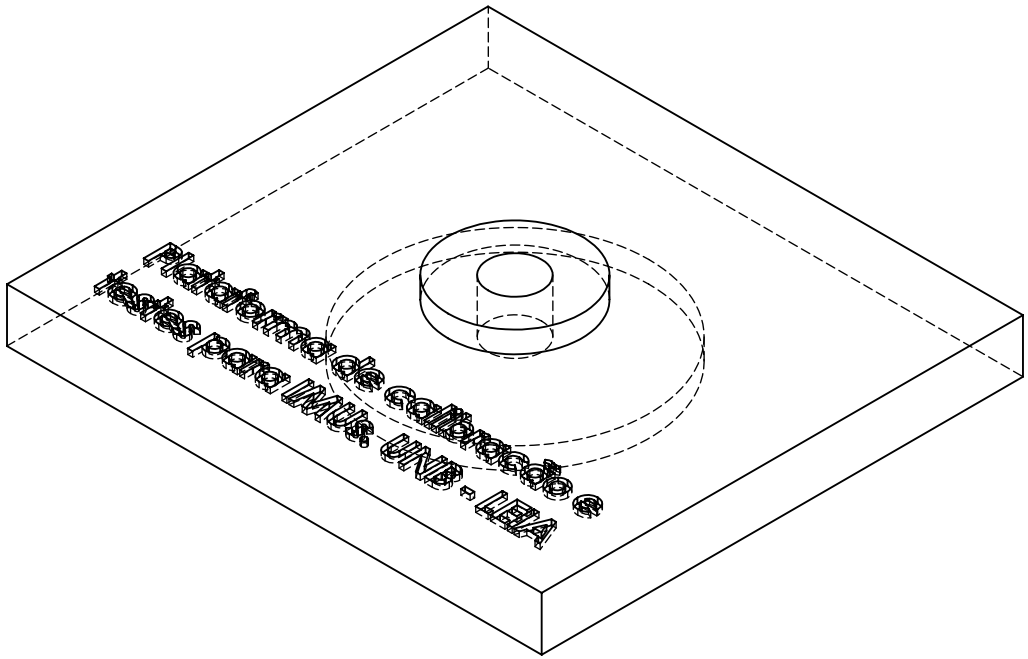


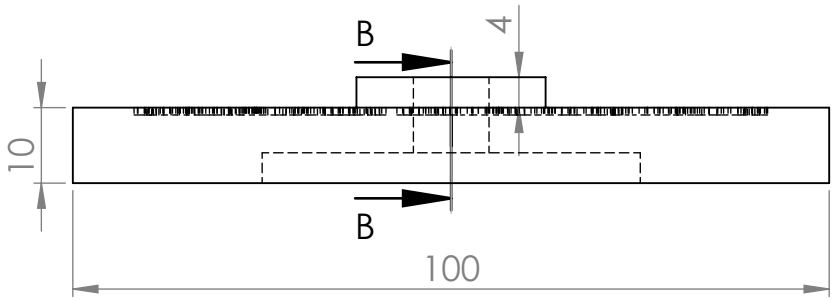


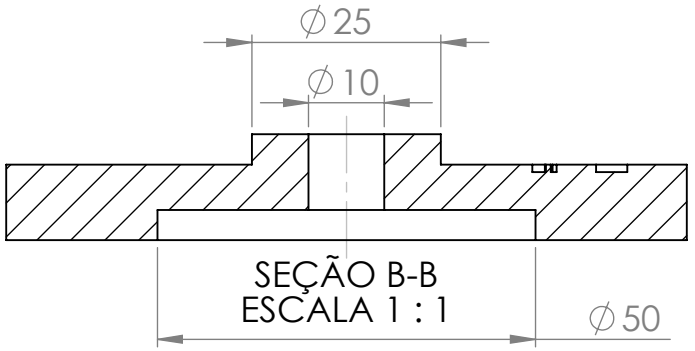


SEÇÃO A-A
ESCALA 1 : 2

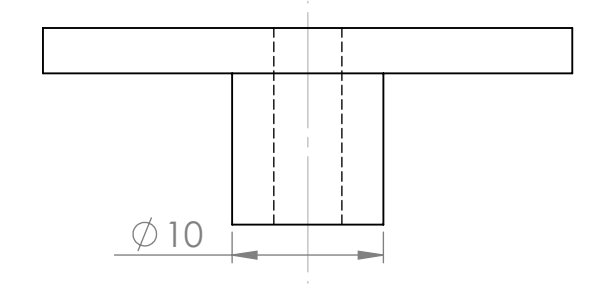
SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 22/05/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Base externa	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 1	FORMATO: A4
		PESO:	ESCALA: 1:2	FOLHA 1 DE 12

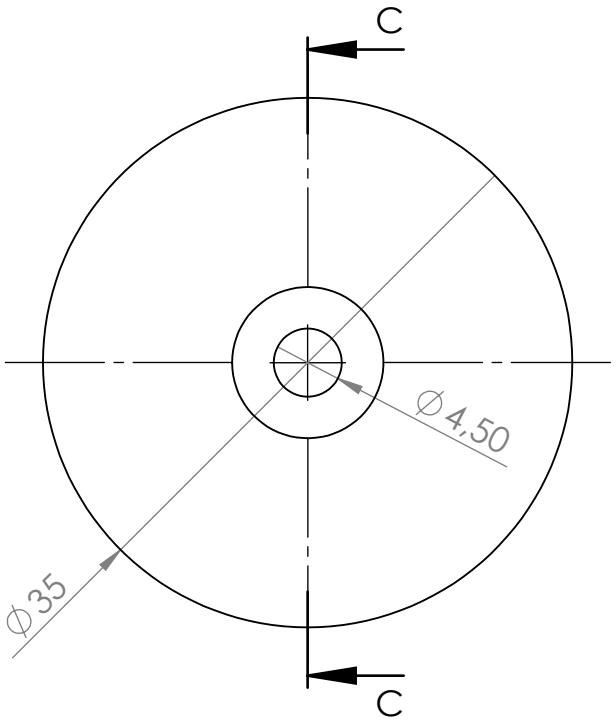


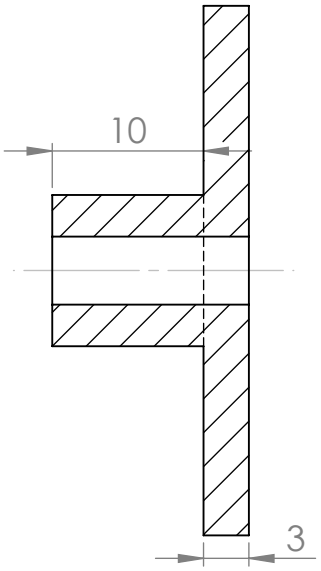




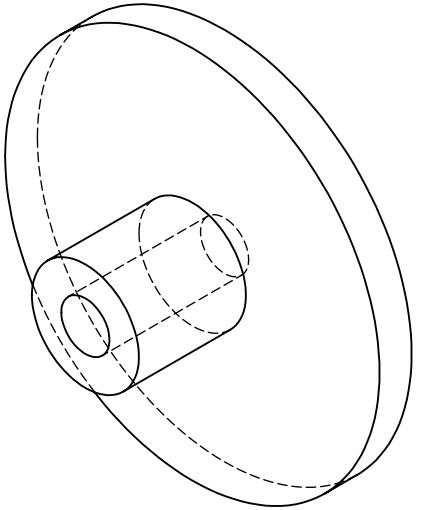
SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 27/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Base interna	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 2	FORMATO: A4
		PESO:	ESCALA: 1:1	FOLHA 2 DE 12



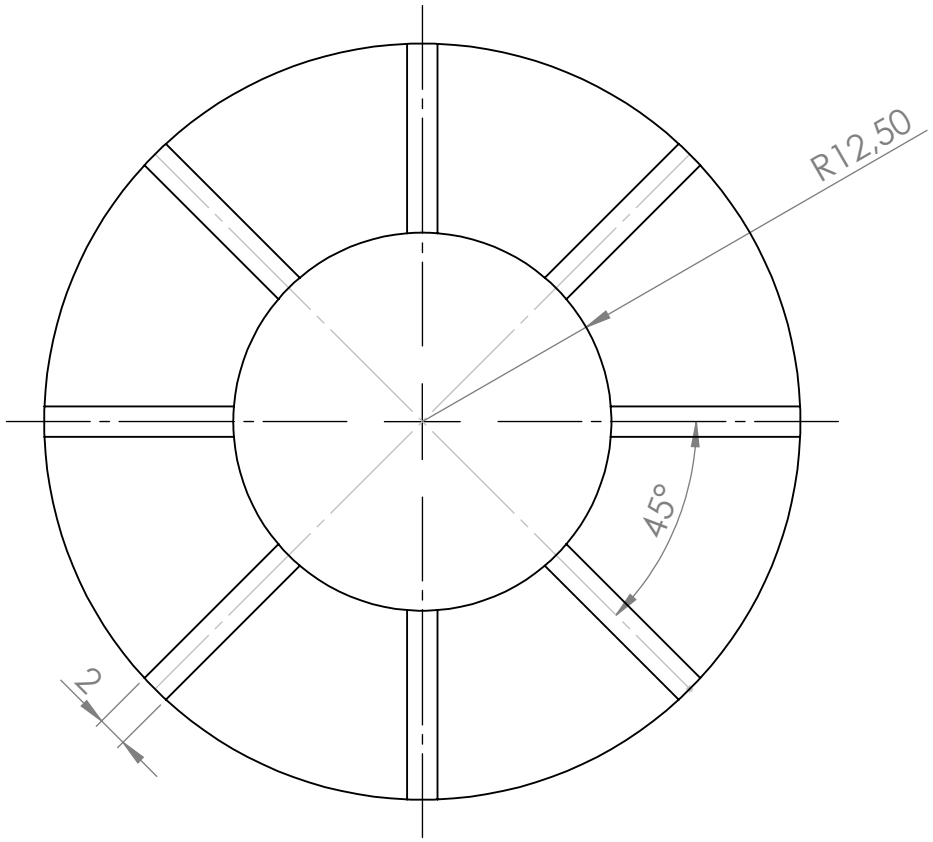


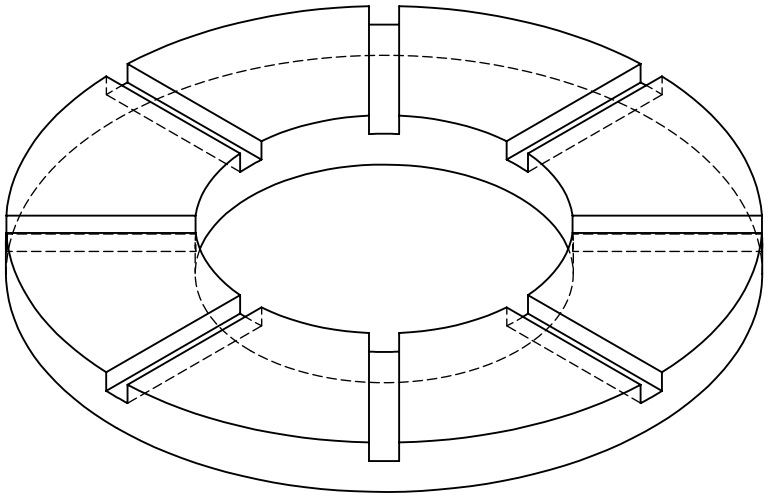


SEÇÃO C-C
ESCALA 2 : 1



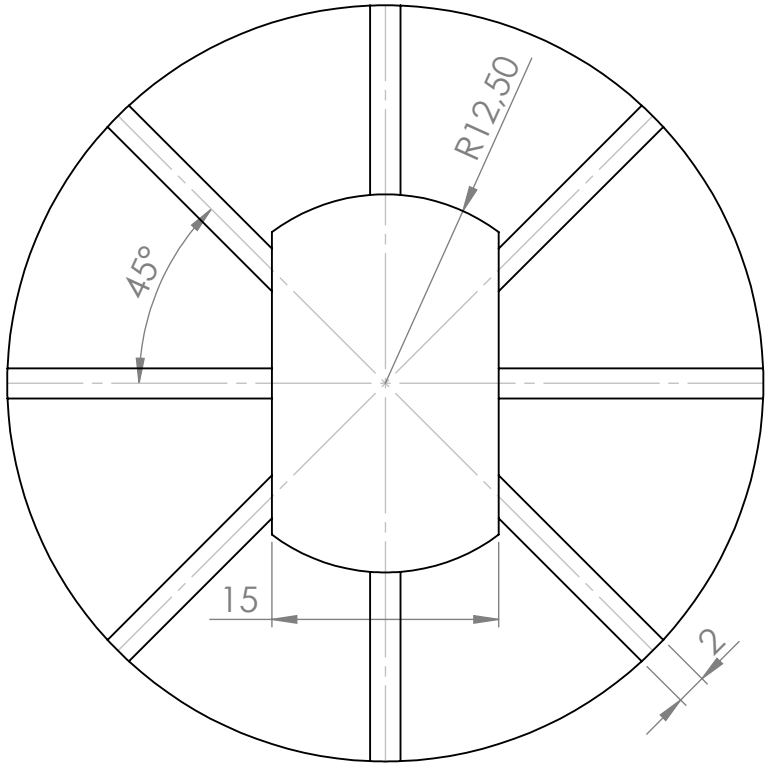
SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 02/04/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Conector base	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 3	FORMATO: A4
		PESO:	ESCALA: 2:1	FOLHA 3 DE 12

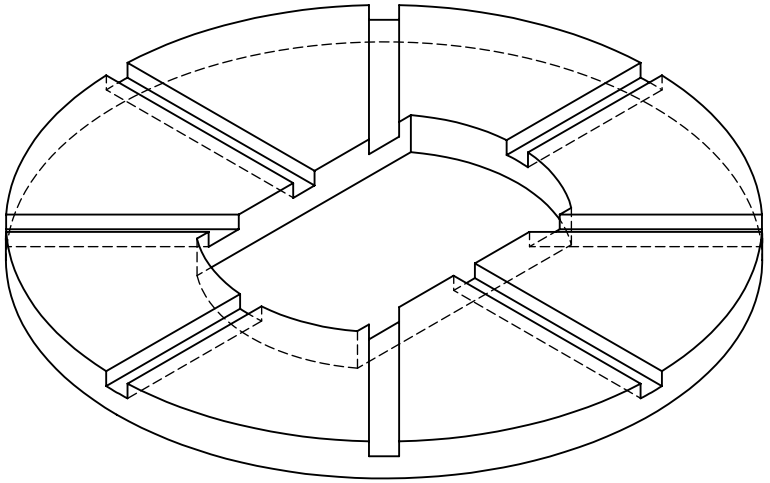


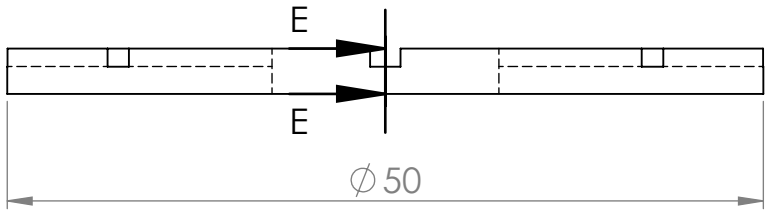


SEÇÃO D-D
ESCALA 2 : 1

SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 12/07/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Base horizontal transferidor	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 4	FORMATO: A4
		PESO:	ESCALA: 2:1	FOLHA 4 DE 12

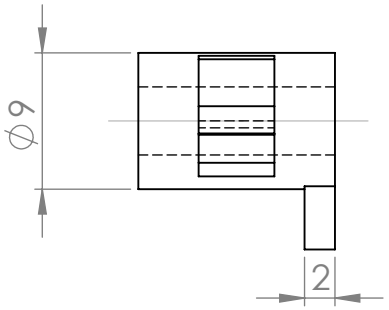


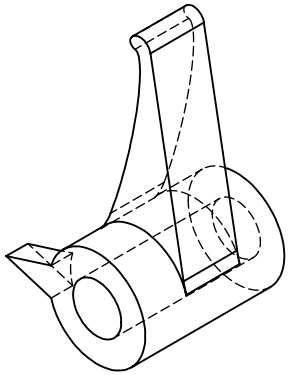


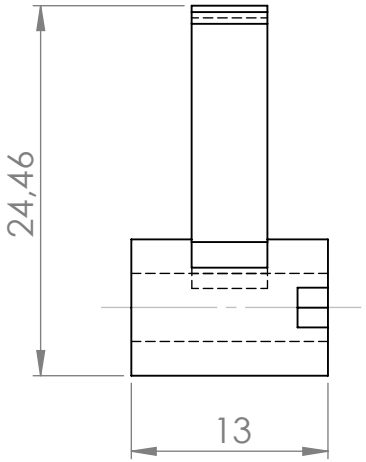


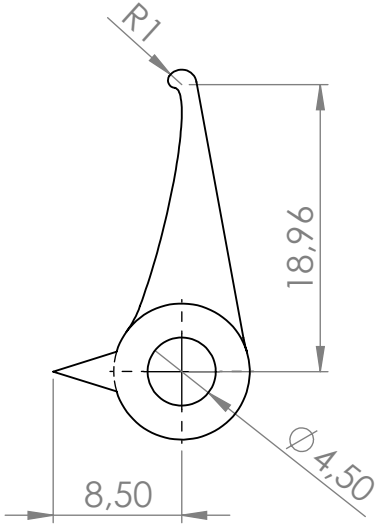
SEÇÃO E-E
ESCALA 2 : 1

SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 12/07/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 2	TÍTULO: Bases verticais transferidores	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 5	FORMATO: A4
		PESO:	ESCALA: 2:1	FOLHA 5 DE 12

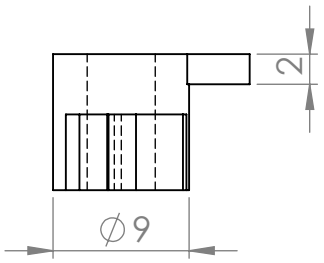


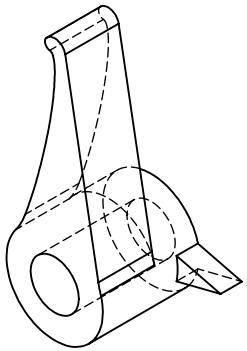


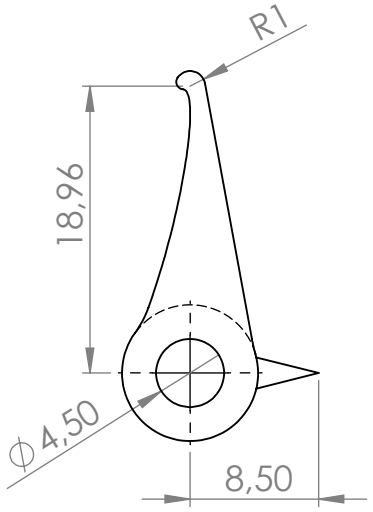


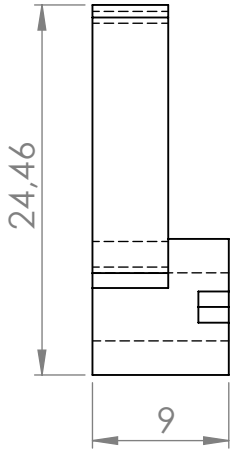


SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 23/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Guia horizontal medição	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 6	FORMATO: A4
		PESO:	ESCALA: 2:1	FOLHA 6 DE 12

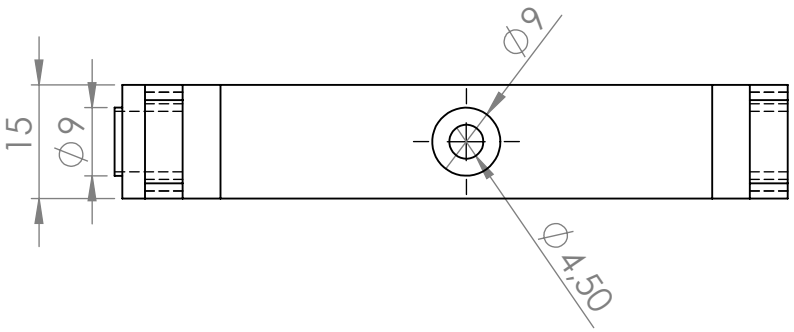


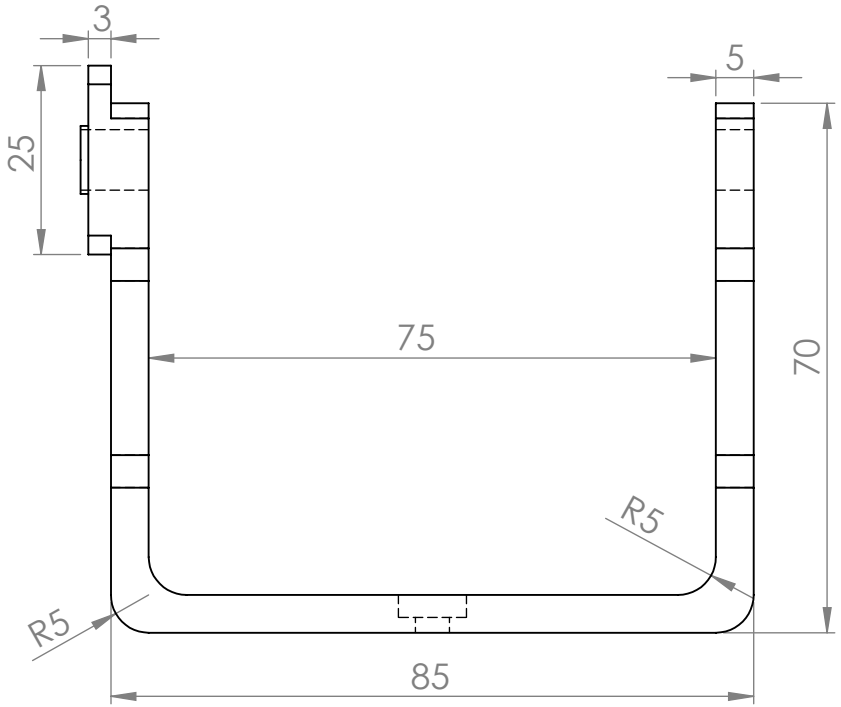


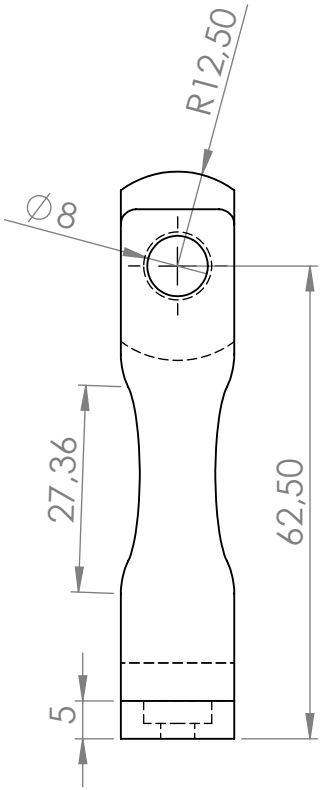


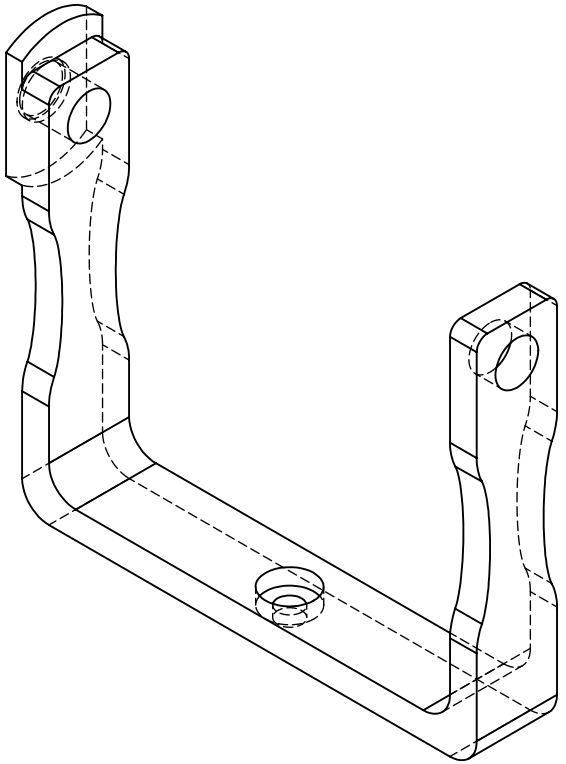


SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 29/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 2	TÍTULO: Guias verticais medição	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 7	FORMATO: A4
		PESO:	ESCALA: 2:1	FOLHA 7 DE 12

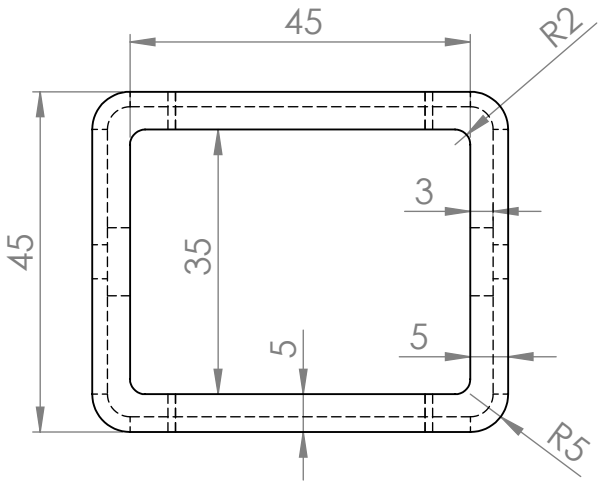


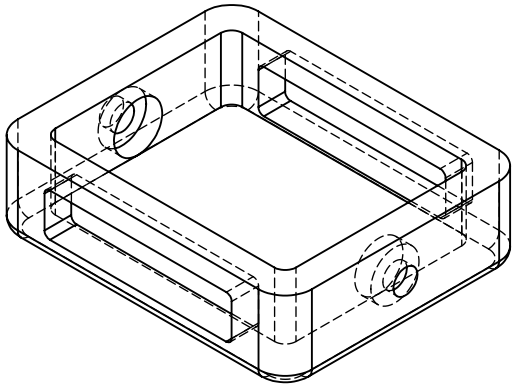


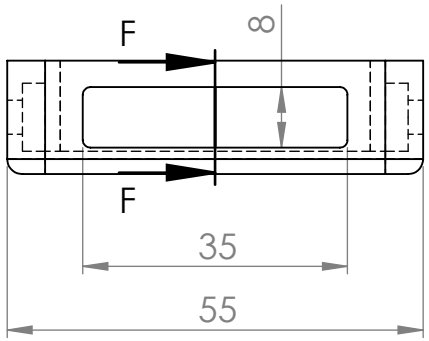


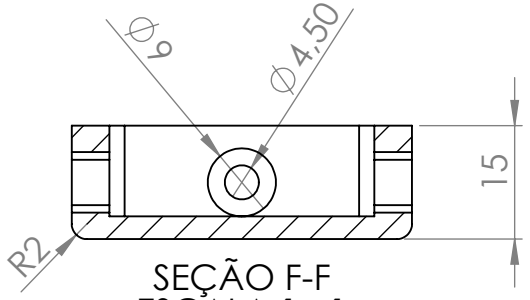


SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 21/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1	TÍTULO: Suporte no eixo Z	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 8	FORMATO: A4
		PESO:	ESCALA: 1:1	FOLHA 8 DE 12



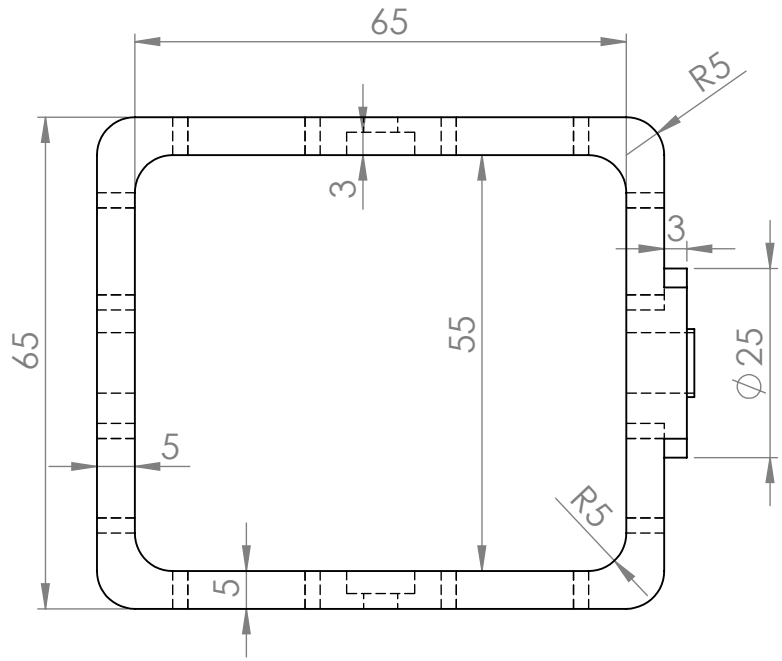


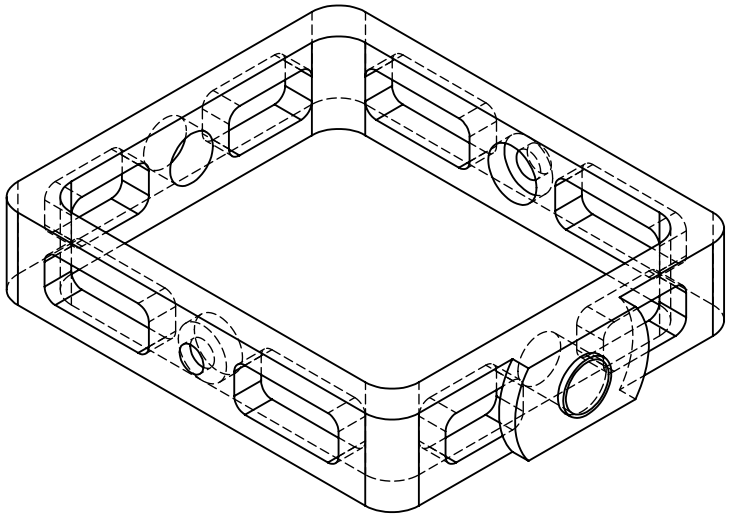


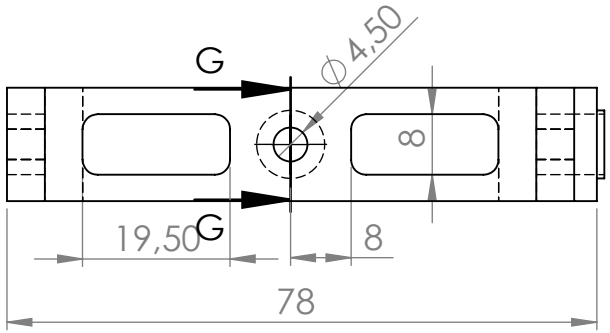


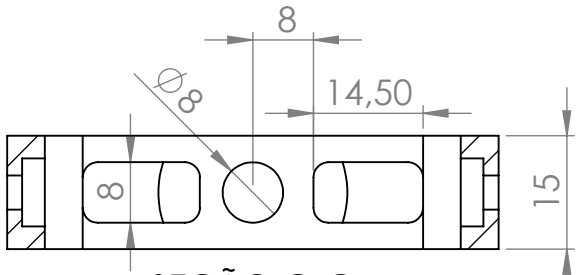
SEÇÃO F-F
ESCALA 1 : 1

SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 21/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
			TÍTULO: Suporte no eixo X	
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1		
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 9	FORMATO: A4
		PESO:	ESCALA: 1:1	FOLHA 9 DE 12



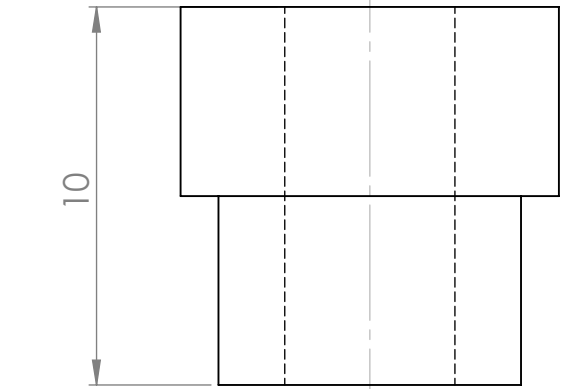


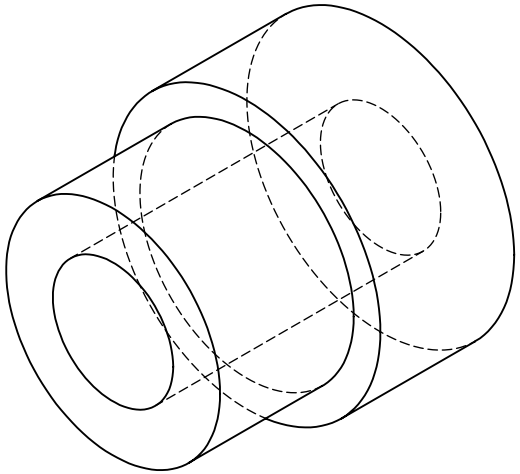


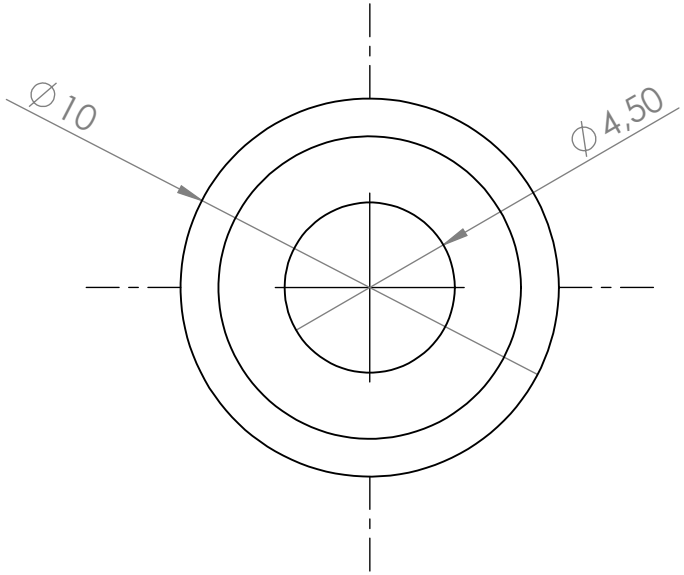


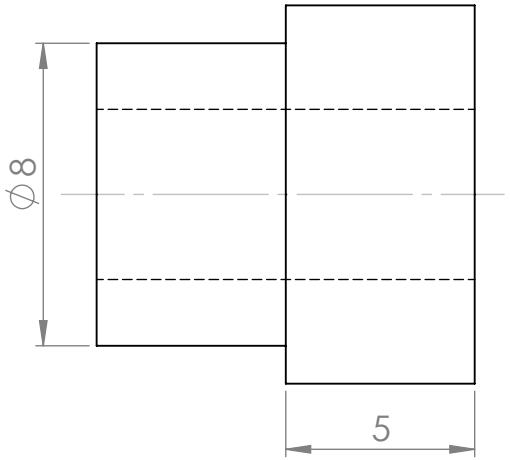
SEÇÃO G-G
ESCALA 1 : 1

SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 21/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
			TÍTULO: Suporte no eixo Y	
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 1		
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 10	FORMATO: A4
		PESO:	ESCALA: 1:1	FOLHA 10 DE 12

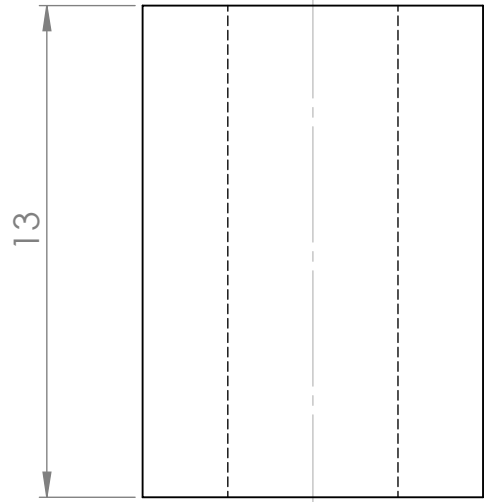


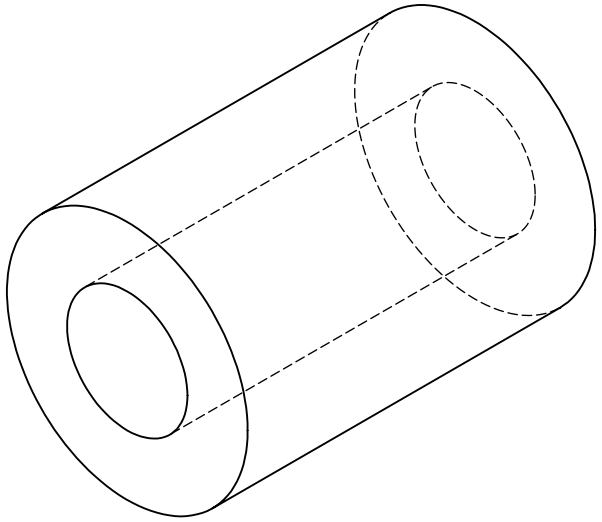


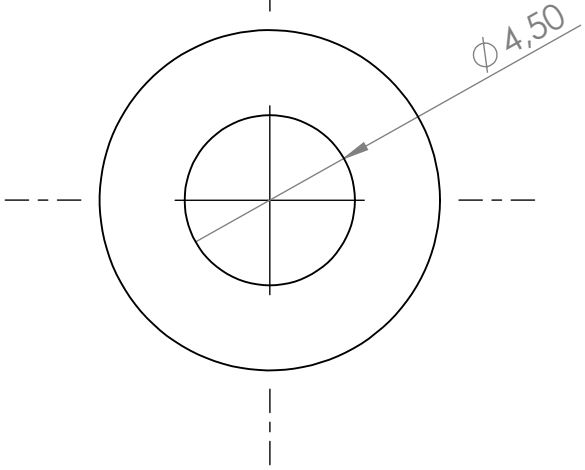


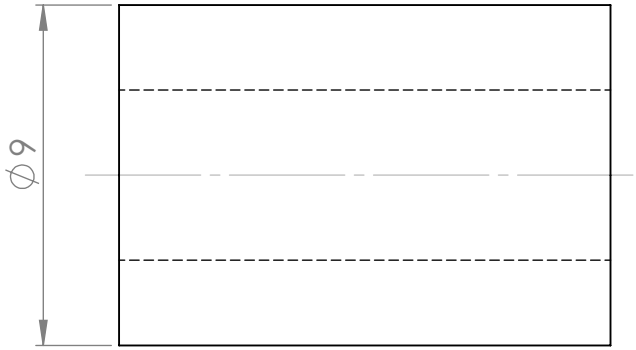


SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 21/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 4	TÍTULO: Rolamentos	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 11	FORMATO: A4
		PESO:	ESCALA: 5:1	FOLHA 11 DE 12









SE NÃO ESPECIFICADO: DIMENSÕES EM MILÍMETROS E TOLERÂNCIAS +-0.1MM	REBARBAR E QUEBRAR ARESTAS AGUDAS	DATA: 21/03/2017	NÃO MUDAR ESCALA DO DESENHO	PLANO DE REVISÃO
NOME: Fabián Barrera Prieto		QUANTIDADE DE PEÇAS: 2	TÍTULO: Casquilhos	
INSTITUIÇÃO OU EMPRESA: Universidade de Brasília		MATERIAL: ABS ou PLA	DES. Nº Peça 12	FORMATO: A4
		PESO:	ESCALA: 5:1	FOLHA 12 DE 12

24th COBEM - 2017

24th ABCM International Congress of Mechanical Engineering
December 3-8, 2017, Curitiba, PR, Brazil

COBEM-2017-1859
A STUDY OF ATTITUDE AND HEADING DETERMINATION THROUGH AN EKF-BASED SENSOR FUSION FOR INERTIAL MEASUREMENT UNITS (IMUs)

Fabián Barrera Prieto
Renato C. Sampaio
Daniel M. Muñoz
Carlos H. Llanos
Universidade de Brasilia, Departament of Mechanical Engineering, Brasilia, Brazil
fbarrera6@gmail.com, llanos@unb.br, renatocoral@unb.br, damuz23@gmail.com

Abstract. *This paper presents the study and design of a sensor fusion technique for Inertial Measurement Units (IMUs) of 6 and 9 degrees of freedom (DOF), in order to determine the attitude (roll and pitch) and heading (yaw) using the Extended Kalman Filter (EKF). The IMUs chosen are the MPU6050 (6 DOF, comprising accelerometer and gyroscope) and the MPU9250 (9 DOF, comprising accelerometer, gyroscope and magnetometer). In order to estimate both the attitude and heading of each IMU a sensor fusion technique based on the use of a gyroscope as the system model and both the accelerometer and the magnetometer as measurement models were used. For the sensor fusion design the EKF has been chosen due to the fact that the system model is highly non-linear. The calibration and tests of the IMUs was performed in a platform, manufactured in a 3D printer. Additionally, simulation results show that the technique used for the sensor fusion presents good quality for both attitude and heading estimation for the two IMUs.*

Keywords: *Sensor Fusion, Extend Kalman Filter, Inertial Measurement Units, MPU6050, MPU9250, Attitude, Heading.*

1. INTRODUCTION

The IMUs are Microelectromechanical Systems (MEMS) developed for a wide range of applications such as robotics, computational vision and artificial intelligence, among others. The main components of these inertial units are: (a) a power supply (3.3V), (b) a processor, (c) sensors and (d) a communication interface (I2C, SPI). Commonly, the sensors that compose the IMU are the accelerometer, the gyroscope and the magnetometer, providing gravitational forces, speed and rotation measurements. Nowadays, it is also common to find IMUs with barometer, altimeter, temperature sensors, among others (Ahmad *et al.*, 2013). Each embedded sensor in an IMU can measure in the three reference axes (namely, X, Y and Z). Sensor fusion techniques aim to combine the information provided by two or more sensors, in order to obtain a better estimative of the state variable(s) with the best possible quality, thus reducing the uncertainty of the estimation. In the case of IMUs, the sensor fusion consists of combining the information of the gyroscope with the accelerometer and magnetometer, allowing the variable of interest (such as inclination, orientation, shock, vibration, among others) to be estimated close to the real value.

The Extended Kalman Filter (EKF) is one of the nonlinear versions of the Kalman Filter, which linearises the state vector through the current mean and covariance. This estimator has been widely regarded in navigation systems (Li and Xu, 2010), through the representation of Euler angles, which are coordinates (*roll*, *pitch* and *yaw*) that serve to specify the orientation and position of a mobile system, with respect to another fixed system. The EKF can be configured for sensor fusion by using information from the gyroscope as the system model, and both data from the accelerometer and magnetometer as the measurement models. This work presents an approach for attitude and heading determination using two EKFs modules for the IMUs, MPU6050 and MPU9250. In the first module (EKF1) the gyroscope and accelerometer data are used to estimate attitude (*roll* and *pitch*). In the second module (EKF2), the heading (*yaw*) is estimated from the fusion between the gyroscope and the magnetometer.

One of the main contributions of this work is the development of a calibration and test platform of IMUs, as well as presenting good attitude and heading estimation results for low cost IMUs, commonly used in commercial and industrial applications; in addition to facilitating the understanding of important topics such as sensors calibration and sensor fusion.

2. ATTITUDE AND HEADING WITH EKF

Both the attitude and the heading is the reference system used in navigation applications, which specifies the movements of an object or body (rotation and orientation) in the horizontal plane as well as in the vertical plane, through the Euler angles (*roll*, *pitch* and *yaw*). *Roll* is the angle corresponding to the X axis, *pitch* on the Y axis and *yaw* on the Z axis. Figure 1 depicts the representation of the Euler angles in a navigation system.

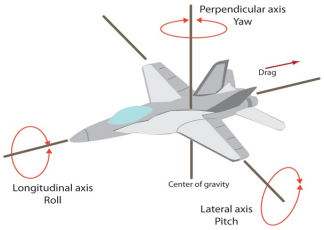
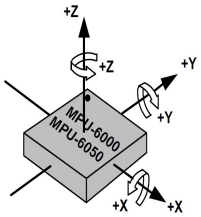
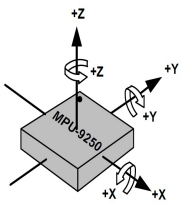


Figure 1. Representation of Euler angles

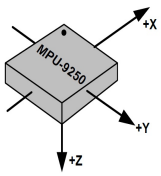
To estimate these three angles with high precision, MEMS sensors are widely used associated with sensor fusion techniques. This type of sensor has several advantages such as: (a) low cost, (b) high sensitivity, (c) compact (single silicon chip), (d) practical (size), and (e) low power consumption. The size of MEMS ranges from $1\mu m$ to $1mm$.



(a) Accelerometer and Gyroscope of MPU6050. InvenSense (2011)



(b) Accelerometer and Gyroscope of MPU9250. InvenSense (2016)



(c) Magnetometer of MPU9250. InvenSense (2016)

Figure 2. Orthogonal axes and reference angles of IMUs.

Figure 2 shows the respective orthogonal axes and reference angles of IMUs utilized in this work, the MPU6050 and MPU9250 with 6 and 9 degrees of freedom (DOF), respectively; both manufactured by Invensense. In Table 1 the main features of these two devices are presented. It can be observed that the main difference between both IMUs is the implementation of the magnetometer in the case of the IMU of 9DOF.

Table 1. Main features of MPU6050 and MPU9250

FEATURES	MPU6050	MPU9250
Sensors	Gyroscope, Accelerometer	Gyroscope, Accelerometer, Magnetometer
Power supply	2.375V-3.46V	2.4V-3.6V
Package size	16 x 20 mm	15 x 25 mm
Degrees of freedom	6	9
Communication interface	I2C	I2C
Normal operating current*	3.9mA	3.5mA
Operation frequency	400kHz	400kHz
Price	$5.00	$18.00
Others	**DMP (Digital Motion Processor) Temperature sensor (-40 to +85^{o}C)	**DMP (Digital Motion Processor) Temperature sensor (-40 to +85^{o}C)

$^{(*)}$ Operating current when all sensors and the DMP are enabled

$^{(**)}$ DMP is the internal processor of each IMU

2.1 Previous works

Several previous works regarding sensor fusion of IMUs based on the EKF estimator were found in the scientific literature. Motion measurement (orientation and position) using a method to compensate the drift error of the inertial sensors (IMU) with the assist of ultrasonic sensors for sensor fusion using EKF. The position measured by the ultrasonic sensor and the orientation measured by the digital compass (magnetometer combined with accelerometer) are defined as

the observation values, and the position, velocity, and orientation are included in the state vector (Zhao and Wang, 2012). A camera pose estimation based on sensor fusion with monocular vision system and IMUs was proposed by (Ligorio and Sabatini, 2013). In this work, the authors developed two EKFs filters, each with different methodology, an EKF using a variant of the Direct Linear Transformation (DLT) method, and the other EKF using the projection errors. Indoor localization of a mini-Unmanned Aerial Vehicle (UAV) with IMUs and vision sensors, in which a EKF like technique to improve the localization was used. The proposed approach allows the designer to use a low-cost Inertial Measurement Unit (IMU) in the prediction step, and the integration of vision odometry for the detection of markers nearness the touchdown area (Benini *et al.*, 2013).

2.2 Gyroscope

This sensor measures the rotational variations (^{o}/sec) exerted on it. MEMs based gyroscopes operate as a result of the Coriolis effect, which consists of generating an external force through a rotating movement on a circular surface. Furthermore, measurements can be made on 1, 2 or 3 orthogonal axes.

The gyroscope measures the angular velocity which is integrated with respect to time, determining the angular position of a body or object. This integration process generates a significant cumulative error in the output signal, being increasing at each iteration; therefore, the measurements of the sensor in a certain time may be far from the correct value. This is the main disadvantage of the gyroscope, also known as *error drift*. The solution to this problem is to filter the output signal. In contrast, the advantage of this sensor is the high precision in the measurements, since the presence of noise in its measurements is almost null. Table 2 presents the features of the gyroscope that is integrated in the MPU6050 and MPU9250. It can be observed that the gyroscope is the same in both IMUs.

Table 2. Features of Gyroscope of the MPU6050 and MPU9250

FEATURES	MPU6050	MPU9250
Measuring axis	Triple-axis (X, Y and Z)	Triple-axis (X, Y and Z)
Dynamic range	$\pm$250, $\pm$500, $\pm$1000 and $\pm$2000 ^{o}/seg.	$\pm$250, $\pm$500, $\pm$1000 and $\pm$2000 ^{o}/seg.
Sensitivity Scale Factor	131, 65.6, 32.8, 16.4 LSB/(^{o}/s)	131, 65.6, 32.8, 16.4 LSB/(^{o}/s)
Output	Digital (16 bits ADCs)	Digital (16 bits ADCs)
Normal operating current	3.6mA	3.2mA

Some of the main applications and use cases of the MEMS gyroscope are: (a) navigation systems, (b) remote control, (c) biomedicine, (d) mobile phones and smart devices, (e) video game consoles, (f) automotive, among others. Figure 3 shows the classification of these applications in relation to the working bandwidth and the dynamic range of the angular velocity (Kraft, 2000).

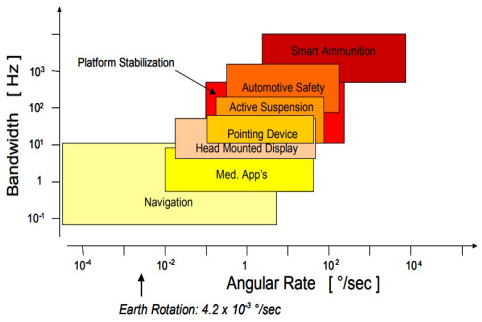


Figure 3. Gyroscope applications according on bandwidth and dynamic angular rate range. (Kraft, 2000)

In the Equations (1), (2) and (3) (Sacco, 2011), the mathematical models to determine the angular position and orientation (Euler angles), using the gyroscope measurements are presented.

$$\dot{\phi} = w_x + w_y \sin\phi \tan\theta + w_z \cos\phi \tan\theta \tag{1}$$

$$\dot{\theta} = w_y \cos\phi - w_z \sin\phi \tag{2}$$

$$\dot{\psi} = w_y \cos\phi \tag{3}$$

2.3 Accelerometer

The accelerometer sensor detects the variation of linear velocity that is exerted on it, thus determining the acceleration (in units "g", where 1g is equivalent to 9.8 m/s^2) of the body. The MEMS technology involving accelerometers has

the ability to perform acceleration measurements on 1, 2 or 3 axes; and can be manufactured based on the piezoelectric, piezoresistive and capacitive physical principles. The main advantage of the accelerometer is that the measurements do not have a significant error. Although its major disadvantage is the presence of noise in the measurements so that the output signal of this sensor must be filtered.

The measurements provided by the sensor can be used to estimate different movements of a body or object, such as: (a) acceleration (b) vibration, (b) shock, (c) inclination and (d) rotation. These movements are generated by accelerations in different periods of time (Sacco, 2011). Figure 4 shows the main applications of the accelerometers with respect to the working bandwidth and the dynamic acceleration ranges (Kraft, 2000). Furthermore, from this figure it can be said that an accelerometer with a lower measurement range has a higher sensitivity.

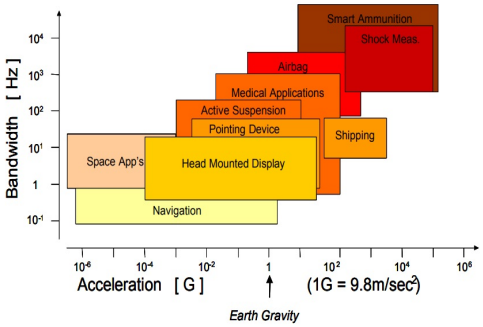


Figure 4. Accelerometer applications according on bandwidth and dynamic acceleration range. (Kraft, 2000)

The features of the accelerometer that is integrated with the MPU6050 and MPU9250 are described in Table 3. In this case, the accelerometer is the same in both the MPU6050 and MPU9250.

Table 3. Features of Accelerometer of the MPU6050 and MPU9250

FEATURES	MPU6050	MPU9250
Measuring axis	Triple-axis (X, Y and Z)	Triple-axis (X, Y and Z)
Dynamic range	±2g, ±4g, ±8g and ±16g	±2g, ±4g, ±8g and ±16g
Sensitivity Scale Factor	16.384, 8.192, 4.096, 2.048 LSB/g	16.384, 8.192, 4.096, 2.048 LSB/g
Output	Digital (16 bits ADCs)	Digital (16 bits ADCs)
Normal operating current	500μA	450μA

Taking into account equations (4) and (5), and the corresponding accelerometer measurements on the three axes (a_x, a_y y a_z), two Euler angles can be estimated, specifically *roll* and *pitch* (Ozyagcilar, 2015). The model of accelerometer in the Equation (5) is usually used in applications related to space navigation. Since the *pitch* angle is restricted to -90^o and +90^o its calculation is achieved with the *atan* function. Otherwise, Equation (4) have solutions in the range -180^o and +180^o; therefore, the *atan2* function can be used in a software application.

$$\phi = \arctan\left(\frac{a_y}{a_z}\right) \tag{4}$$

$$\theta = \arctan\left(\frac{-a_x}{\sqrt{{a_y}^2 + {a_z}^2}}\right) \tag{5}$$

2.4 Magnetometer

This sensor measures the magnitude of the magnetic field in a given direction and its units of measurement are Teslas "T" or Gauss "G". The MEMS magnetometer is an electronic device that has the ability to measure up to 3 orthogonal axes (X, Y and Z). Its main application is to estimate the orientation of a body or object through sensor fusion techniques with the gyroscope, being its function equal to that of an electronic compass. This application approach is widely used in navigation systems. Table 4 shows the main features of the magnetometers MPU9250 device. In this case, the magnetometer is just implemented in the MPU9250.

The Equations (6), (7) and (8) present the mathematical models corresponding to the calculation of the third Euler angle (*yaw*) (Ozyagcilar, 2015). Equations (6) and (7) are the components in the axes X and Y of the magnetic field. Equation (8) is restricted only in a range of solutions between -180^o and +180^o; therefore, it is suitable to use the *atan2* function for this calculation.

$$B_{fy} = m_z \sin\phi - m_y \cos\phi \tag{6}$$

Table 4. Features of Compass of the MPU6050 and MPU9250

FEATURES	MPU6050	MPU9250
Measuring axis	N/A	Triple-axis (X, Y and Z)
Dynamic range	N/A	$\pm 4800 \mu$T
Sensitivity Scale Factor	N/A	1.46 mG/LSB
Output	N/A	Digital (16 bits ADCs)
Normal operating current	N/A	280μA

$$B_{fx} = m_x \cos\theta + m_y \sin\theta \sin\phi + m_z \sin\theta \cos\phi \tag{7}$$

$$\psi = \arctan\left(\frac{-B_{fy}}{B_{fx}}\right) \tag{8}$$

Similar to the MEMS accelerometer the MEMS magnetometer is essential to correct the *drift error* of the Gyroscope for sensor fusion procedure. The main disadvantage of this sensor is that the output signal is unreliable since the measurements have values with little precision (high level of uncertainty) with respect to the real value due to the disturbances of external magnetic fields (for instance from electrical and electronic devices). For which it is necessary to perform a calibration of this sensor before using them in later processes, with the aim of eliminating those perturbations. This drawback (uncalibrated by default) is generated since the manufacturers save manufacturing costs in the sensor calibration. Then, they predict that each user must calibrate the magnetometer to expect good measurement results.

The MEMS magnetometer calibration must be done for each axis. This type of calibration is divided into two parts, called: (a) hard iron and (b) soft iron. The hard iron refers to determine the offset value or bias of the magnetometer, while the soft iron consists of scaling the results of the previous calibration (hard iron). Thus, the soft iron part allows the quality of the magnetometer data to be improved (Winer, 2015).

2.5 Sensor fusion technique

Sensor fusion is a process that consists of combining information (measurements) from different sources (sensors), through a fusion mechanism (e.g., a filter estimator), with the main objective of obtaining an output signal with better quality than that could be obtained from a single sensor (NXP, 2016). In Figure 5 the sensor fusion structure is observed through a fusion structure, which can be based on: (a) Kalman filter (KF, EKF or UKF), (b) complementary filter, (c) particle filter, among others. The parameters of the filters must be adjusted in order to improve the sensor fusion results.

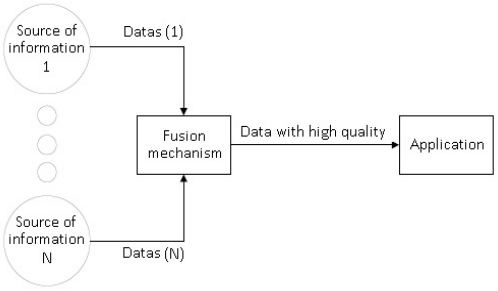


Figure 5. Structure of sensor fusion

This sensor combination technique requires that the available sensors to be complementary to each other, so that the disadvantages of each of the sensors are outweighed by the advantages of the others. Sensor fusion with IMUs is commonly used to find the suitable combination between the gyroscope and the accelerometer (also between the gyroscope and magnetometer) for applications related to navigation systems (for instance, attitude and heading).

2.6 Extended Kalman Filter

The Extended Kalman Filter (EKF) is one of the best known stochastic filters in the area of filtering and state estimation for nonlinear systems. The EKF is an algorithm derived from the basic Kalman filter, which is based on linearising the nonlinear system and performing the estimation of the state variables on such linearised system (Kim, 2011). Linearisation is reflected in a Jacobian matrix (A) of partial derivatives of the function $f(x)$ with regard to each state variable.

Algorithm 1 presents the pseudo-code of the EKF, where it can be observed the 5 main steps of the Kalman filter: (1) predict the state with initial values, (2) compute the error covariance, (3) compute the Kalman gain, (4) update the state estimate and (5) update the error covariance. After step (5) the algorithm returns to step (1) with updated measurements, until a certain number of time steps (N) are reached.

Algorithm 1 Extended Kalman Filter

Require: Initial values: x_0, P_0. Measurements: z_k.
Ensure: Estimate and error covariance: x_k, P_k.

1: **for** $k = 1 : N$ **do**
2: $x_k^- = f(x_{k-1})$
3: $P_k^- = AP_{k-1}A^T + Q$
4: $K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
5: $x_k^+ = x_k^- + K_k(z_k - h(x_k^-))$
6: $P_k^+ = P_k^- - K_k H P_k^-$
7: **end for**

Where, $f(x_{k-1})$ and $h(x_k^-)$ are the predict *state* and *measurements* functions, respectively, k is the *step time*, x_k^- is the predict *state vector*, P_k^- is the predict *covariance matrix*, K_k is the *Kalman gain*, x_k^+ is the estimate *state vector*, P_k^+ is the estimate *covariance matrix*, A and H are the *Jacobian matrices* of partial derivatives of f and h with respect to state variables, z is the *measurement vector*, Q is the *process noise* covariance matrix and R is the *measurement noise* covariance matrix.

2.7 Results and analysis

For the estimation of attitude and heading (Euler angles), two procedures were conceived: (a) experimental and (b) computational. In the experimental procedure both the calibration and tests of IMUs were achieved through of a mobile platform in the three axes (X, Y and Z), which was manufactured in a 3D printer. The main objective of the platform is to measure the real value of the movement exercised over the IMU. These movements are yielded with a protractor (one for each axis) allowing the user to have a reference of such value. Additionally, the mobile platform facilitates both the calibration and testing procedures. For both IMUs the same equipment was used, which consists mainly of: (a) the platform, (b) an Arduino Nano, (c) a breadboard, (d) the protractors and (e) a ball level (android application). The Figure 6 depicts these platform of IMUs together with the other components.

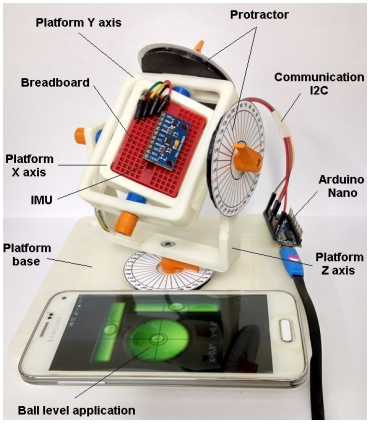


Figure 6. Calibration and tests platform of IMUs

For both calibrations and tests of the IMUs it was taken into account that the mobile platform was well positioned. To achieve that, an Android ball level application was used, in which the 2D tilt can be observed, and thus having control over the position of the platform. Furthermore, the protractors were used to measure the angles at which the IMUs are rotated, these measurements will be used to compare them with regard to the estimates provided by the EKF based sensor fusion technique.

For the calibration procedure the dynamic ranges of $\pm 250^o/sec$, $\pm$2g and $\pm 4800\mu$T were used for the gyroscope, the accelerometer and the magnetometer, respectively. On the other hand, the calibration was performed by using a different methodology for each sensor. For the gyroscope and accelerometer a static calibration was performed, whereas a dynamic calibration was executed for the magnetometer of the MPU9250. In the case of gyroscope and accelerometer, the IMUs were initially placed on specific positions with the Z axis perpendicular to the horizontal terrestrial plane and keeping them static. In this calibration process, 100 samples of the 3 axes (X, Y and Z) were acquired, from which a maximum and minimum value were obtained in order to determine the *offset value* of the sensors, as stated by Equation 9.

$$offset_{x,y,z} = \frac{max_{x,y,z} + min_{x,y,z}}{2} \tag{9}$$

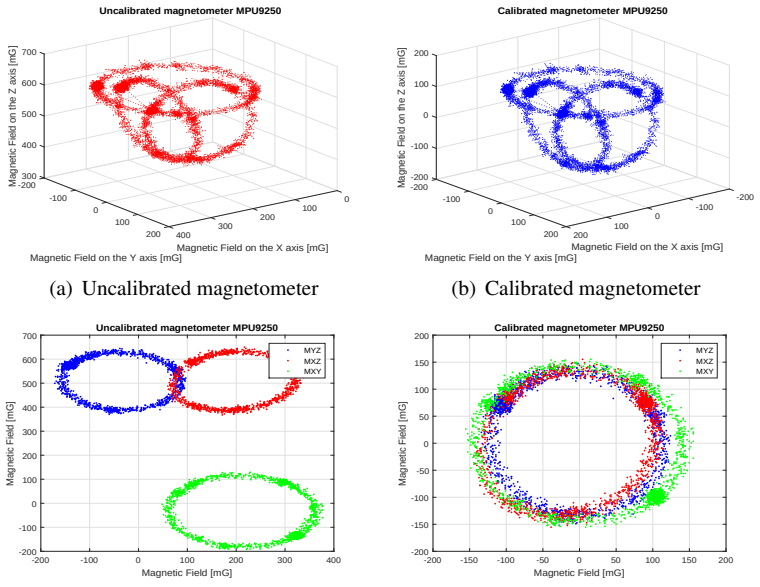
For the magnetometer the calibration firstly consisted on calibrating the X and Y axis and then calibrating the Z axis, given that the calibration of this sensor must be dynamic for each of the axes, with respect to the same reference magnetic field. This is the reason why the three axes can not be calibrated at the same time. Before performing the calibration, the axis of the magnetometer was adjusted according to Figure 2c. In this case, the sense and direction of the 3 axes (X, Y and Z) of this sensor are different with respect to the 3 axes of the gyroscope and accelerometer. For this reason, in the code developed in Arduino the change of assignments of both sense and direction was made so that the 3 sensors have the same configuration with respect to the axes of the gyroscope and accelerometer.

Calibrations for the magnetometer were based firstly on positioning the IMU with the Z axis perpendicular to the horizontal plane, and then rotating the IMU 360^o around this axis, in order to measure the Earth's magnetic field. In this way the X and Y axes were calibrated. For calibration of the Z axis the IMU should be positioned with the Y axis perpendicular to the horizontal plane, and then rotated the IMU 360^o around this axis. The duration of each of the two calibration processes was 20s. The maximum and minimum values were obtained, determining the offset (hard iron part) by using Equation 9 and the scale factor (soft iron part) using Equation 10.

$$scale_{x,y,z} = \frac{max_{x,y,z} - min_{x,y,z}}{2} \tag{10}$$

Figure 7 shows the results before and after calibration procedure of the magnetometer of the MPU9250. In Figure 7a and Figure 7b, the three-dimensional views of both uncalibrated and calibrated measurements are presented respectively, when comparing these figures it can be observed that both have the same shape, but different proportions of both position and length. The measurements in Figure 7a are "useless" data to be applied, since the measurements are not centered on the reference point (0,0,0) of the 3 axes and furthermore, the circumferences have different measurement ratio. In contrast, these two disadvantages are not observed in Figure 7b, since during the calibration process the measurements were positioned and scaled according to the measured magnetic field.

Figure 7c and Figure 7d show the same measurements of the previous case from a 2D perspective. When compared, the difference between uncalibrated and calibrated magnetometer data is better observed. In Figure 7c the measurements of each axis of the magnetometer are very distant from the other axes and completely disproportionate, whereas in Figure 7d the measurements of the 3 axes were correctly calibrated, centralized and scaled.



(a) Uncalibrated magnetometer (b) Calibrated magnetometer

(c) Uncalibrated magnetometer (d) Calibrated magnetometer

Figure 7. Results of uncalibrated and calibrated magnetometer for MPU6050

The offset and scales values obtained in the previous calibration processes for the 3 sensors, must be taken into account in order to adjust the measurements of each of the sensors. For this, the values of offset and scales must be subtracted from the "raw" measurements of each of the sensors in the Arduino code, where these "raw" values refer to the default measured data provided by each sensor. Therefore, for the output measurements of the gyroscope and for the measurements of the axis X and Y of the accelerometer, the output values are given by Equation 11, whereas for the measures corresponding to the axis Z of the accelerometer the output value is determined by Equation 12. In the case of the magnetometer, Equation 13 represents the model for calculating the output value for the 3 axes of this sensor.

$$Out\ Value = \frac{Raw\ Value - Offset}{Sensitivity\ Scale} \tag{11}$$

$$Out\,Value = 1 + \frac{Raw\,Value - Offset}{Sensitivity\,Scale} \tag{12}$$

$$Out\,Value = \frac{(Raw\,Value - Offset) * Scale}{Sensitivity\,Scale} \tag{13}$$

With regard to the sensor fusion computational procedure two techniques have been used. The first one for *roll* and *pitch* angles and second one for *yaw*. In the case of *roll* and *pitch*, the gyroscope was considered as the system model and the accelerometer as the sensor model in the EKF algorithm. Meanwhile the gyroscope was also used as the system and the magnetometer as sensor for the estimation of the *yaw* angle. For each combination technique, for attitude and heading estimation an EKF was developed and tuned. Figure 8 depicts the configuration of the EKFs sensor fusion modules.

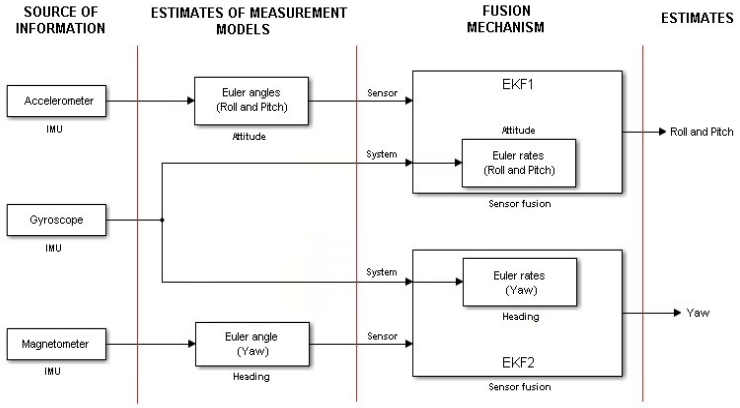


Figure 8. Sensor fusion tecnique for IMUs

Table 5 presents the parameters selected for the EKF, in order to perform the study of the sensor fusion technique. The selection of these parameters consists of estimating an initial value of the state variables x_0 and the covariance matrix P_0. The other parameter to configure is the Jacobian matrix H, which is related to the available sensors for each state variable.

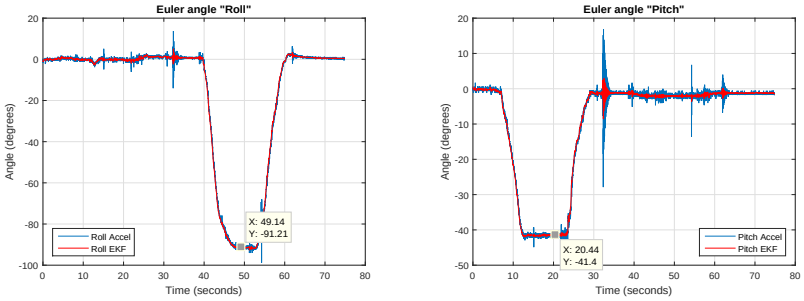
In order to obtain an optimal performance of the EKF, the parameters Q and R must be adjusted correctly. To do that, there are two ways of tuning them. The first one is trial and error; that is, to change the values of these covariances until obtaining a good estimate of the state variables. The other method is to adjust Q according to the reliability of the process model, so that if it is highly reliable a small value can be selected for this parameter. For the R parameter, a set of sensor measurements must be obtained and then perform a statistical analysis to determine the covariance of the sensor. Another option to determine this parameter is to take a value already used in similar cases for this sensor.

This work took into account that the process model is highly reliable; therefore, Q is small. Whereas for the sensor (R) a value used in a previous work of sensor fusion was taken, to determine the attitude with the linear Kalman filter (KF) (Lauszus, 2015). In EKF1 are two state variables (*roll* and *pitch*) so that the parameters are set in an array of 2x2 size, whereas in the case of EKF2, the parameters are scalar because this second fusion mechanism is working with only one state variable (*yaw*).

Table 5. EKF parameters for sensor fusion of the MPU6050 and MPU9250

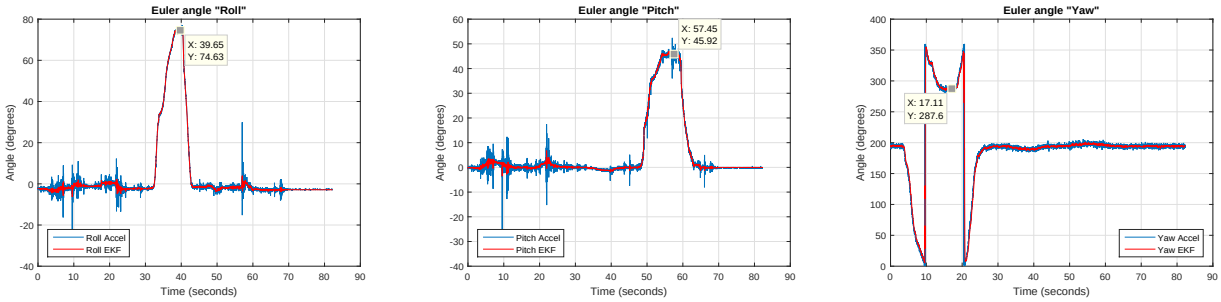
PARAMETER	Initial values EKF1	Initial values EKF2
x	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	0
P	$\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$	10
Q	$\begin{bmatrix} 0.001 & 0 \\ 0 & 0.001 \end{bmatrix}$	0.001
R	$\begin{bmatrix} 0.03 & 0 \\ 0 & 0.03 \end{bmatrix}$	0.03
H	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	1

The tests were achieved through the interaction between the Arduino-Nano and the Matlab R2015 software for reading (the sensors values) and data processing (the filter), respectively. Figure 9 illustrates the results obtained for both *roll* (Figura 9a) and *pitch* angles (Figura 9b) for MPU6050 and Figure 10 shows the results obtained from the estimation of three Euler angles, *roll* (Figure 10a), *pitch* (Figura 10b) and *yaw* (see Figure 10c) for MPU9250. Based on these figures it can be concluded that the noise has been eliminated from the filtered signal (red line) when compared to the accelerometer or magnetometer signal (blue line).



(a) Results for *roll* angle (b) Results for *pitch* angle

Figure 9. Euler angles estimation for MPU6050



(a) Results for *roll* angle (b) Results for *pitch* angle (c) Results for *yaw* angle

Figure 10. Euler angles estimation for MPU9250

The metric that was used to evaluate the performance of this sensor fusion technique is the Normalized Root Mean Squared Error (NRMSE), which is based on a normalized version of the Root Mean Squared Error (RMSE). Equation 14 presents the model for determining the NRMSE, which is given in units of percentage. Therefore, it can be used for direct comparisons with results from other datasets (with different units of measure) (Spüler *et al.*, 2015).

$$NRMSE = \frac{\sqrt{\frac{\sum_{i=0}^{n}(\hat{y}_i - y_i)^2}{n}}}{\bar{y}} \tag{14}$$

Table 6 shows the NRMSE results of the Euler angles, for both MPU6050 and MPU9250, which details the good quality of the attitude and heading estimation. The values estimated by the EKFs were compared with the measured values (real) in the calibration and tests platform for IMUs.

Table 6. NRMSE for the MPU6050 and MPU9250

EULER ANGLES	MPU6050	MPU9250
roll	0.0163	0.07
pitch	0.0375	0.0882
yaw	N/A	0.0095

3. CONCLUSIONS

In this paper the EKF was used for attitude and heading determination for two IMUs (MPU6050 of 6DOF e MPU9250 of 9DOF) through a sensor fusion technique using two EKF modules, one to estimate *roll* and *pitch* angles, and the other one to estimate *yaw* angle. Two different calibrations were performed, a static calibration for the gyroscope and the accelerometer, and a dynamic calibration for the magnetometer. For the latter sensor two types of calibrations were performed (hard iron and soft iron) with the same sample. Several tests have been made in order to evaluate the performance of this multi-sensor approach. The simulation results obtained have been compared with the real value measurements using a mobile platform designed for calibration and tests of this type of sensors. These platform allows the movement of IMUs in the three axes. The NRMSE performance metric was used to compare the achieved results. The results of NRMSE shows that the EKF based on sensor fusion has good quality for both attitude and heading estimation.

As a proposal of future works taking into account the presented sensor fusion technique (two EKF modules), tests with other IMUs of 9 DOF could be performed, moreover, the UKF or another type of sensor fusion mechanism could be implemented, with the main objective of analyzing and comparing the results of the estimation of *roll*, *pitch* and *yaw* obtained in this work.

4. ACKNOWLEDGEMENTS

This work was supported by FAPDF.

5. REFERENCES

Ahmad, N., Ghazilla, R.A.R., Khairi, N.M. and Kasi, V., 2013. "Reviews on various inertial measurement unit (imu) sensor applications". *International Journal of Signal Processing Systems*, Vol. 1, No. 2, pp. 256–262.

Benini, A., Mancini, A. and Longhi, S., 2013. "An imu/uwb/vision-based extended kalman filter for mini-uav localization in indoor environment using 802.15. 4a wireless sensor network". *Journal of Intelligent & Robotic Systems*, pp. 1–16.

InvenSense, 2011. "Mpu-6000 and mpu-6050 product specification". 25 Sep. 2016.

InvenSense, 2016. "Mpu-9250 product specification". 8 Mar. 2017.

Kim, P., 2011. *Kalman filter for beginners: with MATLAB examples*. CreateSpace.

Kraft, M., 2000. "Micromachined inertial sensors: The state-of-the-art and a look into the future". *Measurement and Control*, Vol. 33, No. 6, pp. 164–168.

Lauszus, K., 2015. "Kalmanfilter". 13 Apr. 2017 <https://github.com/TKJElectronics/KalmanFilter/blob/master/Kalman.cpp>.

Li, Y. and Xu, X., 2010. "The application of ekf and ukf to the sins/gps integrated navigation systems". In *Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference on*. IEEE, pp. 1–5.

Ligorio, G. and Sabatini, A.M., 2013. "Extended kalman filter-based methods for pose estimation using visual, inertial and magnetic sensors: Comparative analysis and performance evaluation". *Sensors*, Vol. 13, No. 2, pp. 1919–1941.

NXP, 2016. "Nxp sensor fusion library for kinetics mcus". *DATA SHEET: PRODUCT PREVIEW, rev*, Vol. 8.

Ozyagcilar, T., 2015. "Implementing a tilt-compensated ecompass using accelerometer and magnetometer sensors". *Freescale Semiconductor Application Note, rev*, Vol. 3.

Sacco, M., 2011. "Sensores inerciales: El mundo en movimiento". 1 Aug. 2017 <http://www.neoteo.com/21690-sensores-inerciales-el-mundo-en-movimiento>.

Spüler, M., Sarasola-Sanz, A., Birbaumer, N., Rosenstiel, W. and Ramos-Murguialday, A., 2015. "Comparing metrics to evaluate performance of regression methods for decoding of neural signals". In *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*. IEEE, pp. 1083–1086.

Winer, K., 2015. "Simple and effective magnetometer calibration". 22 Mai. 2017 <https://github.com/kriswiner/MPU6050/wiki/Simple-and-Effective-Magnetometer-Calibration>.

Zhao, H. and Wang, Z., 2012. "Motion measurement using inertial sensors, ultrasonic sensors, and magnetometers with extended kalman filter for data fusion". *IEEE Sensors Journal*, Vol. 12, No. 5, pp. 943–953.

6. RESPONSIBILITY NOTICE

The following text, properly adapted to the number of authors, must be included in the last section of the paper:
The author(s) is (are) the only responsible for the printed material included in this paper.