# Sistema de Aquisição de Dados de Vôo

Arthur Evangelista dos Santos
Universidade de Brasília,
Faculdade do Gama — UnB, FGA
Matrícula: 14/0016686
Email: arthuevangelista@hotmail.com

Fábio Barbosa Pinto
Universidade de Brasília,
Faculdade do Gama — UnB, FGA
Matrícula: 11/0116356
Email: fabio_bbarbosa@hotmail.com

***Abstract*—Este projeto visa criar um sistema de aquisição de dados, com o uso de uma Raspberry Pi 3 Model B, para uma aeronave não tripulada, radiocontrolada e de pequeno porte. Os dados a serem adquiridos e processados serão relacionados à aeroelasticidade e referência-atitude da aeronave. Serão utilizadas três unidades de medição inercial (IMU), sendo uma em cada meia asa e uma no centro da aeronave, um módulo GPS e um *display* para apresentação dos dados obtidos por meio de uma Interface Gráfica para o Usuário (GUI).**

***Keywords*—*Sistemas Operacionais Embarcados, Raspberry Pi, VANT, IMU, AHRS, Aeroelasticidade.***

## I. Introdução

A indústria aeronáutica é uma das mais vastas do mundo. A quantia de inovações tecnológicas que visam a melhoria de desempenho de uma aeronave, bem como a segurança de sua tripulação, agregam alto valor de mercado a este segmento. Para concepção de um projeto aeronáutico, se faz necessária a validação de modelos teóricos de engenharia aeronáutica e aeroespacial. Entretanto, a construção de um protótipo de uma aeronave requer grande investimento. Caso o modelo teórico não seja coerente ou condizente com a realidade de operação desta aeronave danos com insumos, recursos humanos e recursos financeiros podem acarretar na falência de uma empresa deste ramo.

Portanto, uma opção viável é a realização de um modelo em pequena escala deste projeto aeronáutico e extrapolar alguns dados obtidos nos testes em túnel de vento e em voo. Por motivos de segurança, é comum que este modelo seja radiocontrolado. A este modelo é dado o nome de *Aerodesign*, *Unmanned Aerial Vehicle* (UAV) ou Veículo Aéreo Não Tripulado (VANT).

A aquisição de dados pode ser realizada por um simples microcontrolador (MCU). Entretanto, separar os dados e processá-los deve ser realizado com um computador numa *groundstation* e consome tempo e recursos humanos. A automatização destes processos pode ser realizada com um *System-on-Chip* (SoC) durante a operação de voo. Com o auxílio de um GPS e um IMU pode ser traçada a trajetória de um VANT. Para automatização de voo deste veículo, os dados adquiridos pelo conjunto proposto (GPS, IMU e SoC) podem ser úteis para os algoritmos utilizados no sistema de controle da aeronave.

Um sistema utilizado na indústria é o *Attitude-Heading Reference System* (AHRS) que consiste em sensores, em conjunto com um MCU, para aquisição de dados de ângulo de atitude e referencial inercial da aeronave (conceitos definidos apropriadamente na Seção **??**).

Este trabalho visa projetar e implementar um sistema de aquisição de dados de voo para um VANT que consiga adquirir os dados de um AHRS, torção, flexão e vibração da asa e velocidade e altitude da aeronave. Alguns dos dados a serem adquiridos podem ser derivados, por meio de manipulações matemáticas (Seção **??** e **??**), de leituras de sensores implementados para outros propósitos no sistema. Para rápida referência, é proposta a apresentação destes dados para um usuário ao final do procedimento de voo por meio de uma GUI. A aeronave a ser utilizada para testes do sistema é radiocontrolada por um piloto em *groundstation*. Este piloto seguirá o procedimento de avaliação proposto por Cooper e Harper (Seção V-B). Um trabalho a ser realizado no futuro é a integração do sistema proposto com um sistema de controle autônomo de uma aeronave.

## II. Objetivos

- Adquirir dados de flexão e torção da asa;
- Adquirir dados de vibração da asa;
- Adquirir velocidade da aeronave;
- Adquirir altitude, ângulo de atitude e ângulo de ataque;
- Fusão dos dados dos acelerômetros e dos giroscópios;
- Processamento dos dados adquiridos;
- *Plot* da FFT, PDS e espectrograma (FFT/tempo);
- Fusão dos dados do IMU e do GPS com o Filtro de Kalman;
- Organizar dados de acordo com o procedimento de voo realizado;
- Apresentar resultados em uma GUI para o usuário.

## III. Requisitos

- Velocidade, aceleração e posição (linear e angular) da aeronave e de cada meia asa;
- Ângulo de atitude e ângulo de ataque da aeronave;
- Altitude, posição e trajetória da aeronave;
- Operações matemáticas (FFT, arctg, plot de gráficos);
- Implementação do Filtro de Kalman;
- GUI apresentando trajetória e dados adquiridos;

## IV. Benefícios

O benefício de se utilizar o conjunto proposto em relação ao uso de um microcontrolador é a possibilidade de se automatizar a etapa de processamento de dados durante o voo de um VANT. Isto reduz os custos com recursos humanos alocados no trabalho com estes dados. Também é viabilizado o envio destes dados para um sistema de controle afim de se automatizar o procedimento de voo. Ademais, a implementação aqui proposta possui baixo custo, quando comparado a implementações encontradas no mercado, tornando viável sua reprodução em protótipos funcionais ou outras pesquisas.

## V. Revisão Bibliográfica

### A. Veículo aéreo não tripulado (VANT)

Não há consenso quanto ao surgimento de VANT [1] por parte dos historiadores. Considera-se que o primeiro veículo aéreo não tripulado tenha surgido na Áustria em 1849 para transportar explosivos [1]. Pesquisas relacionadas a VANT para o segmento militar foram intensificadas com o advento da Primeira e Segunda Guerra Mundial. VANT foram utilizados para validar modelos de aerodinâmica e de aeroelasticidade como no projeto DAST (sigla, do inglês, *Drones for Aerodynamic and Structural Testing*) da NASA que ocorreu de 1977 a 1983 [2]. O uso de VANT com propósito militar se popularizou com a repercussão do programa classificado dos Estados Unidos desmascarado pelo governo Chinês por volta de 1982 [3].

Fig. 1. Técnicos instalam um drone BQM-43 Firebee II no pilone da asa de uma aeronave B-52B para testes no projeto DAST [2].

Atualmente, VANT são utilizados para fins militares, como monitoramento de divisas, reconhecimento territorial ou transporte de suprimentos, ou para fins civis, como em projetos de pesquisa, agricultura, transporte de bens ou simples recreação. Sua introdução no espaço aéreo não segregado ainda não é regulamentada [4] sendo uma tecnologia recentemente em ascensão quando comparada a aviação tripulada. A regulamentação sobre a operação de VANT autônomos ainda está em discussão por órgãos responsáveis como a ANAC, FAA, entre outros. Em um futuro próximo, a existência de um AHRS e uma caixa preta podem ser condições mínimas e necessárias para operação de VANT.

[1]No presente trabalho, a sigla VANT será utilizada para veículo aéreo não tripulado no singular ou no plural

### B. Avaliação Cooper-Harper

A Escala de Avaliação de Qualidade do Manuseio de Aeronaves Cooper Harper (Cooper Harper Handling Qualities Rating Scheme), comumente designada Escala Cooper-Harper, é uma escala de avaliação quantitativa e qualitativa de uma aeronave quanto à sua controlabilidade e manobrabilidade. A avaliação segue o algoritmo apresentado no Apêndice A, retirado do relatório original disponível em [5]. O piloto, após executar procedimentos de voo pré-estabelecidos, deve avaliar se houve carga de trabalho para manter a aeronave estável e em bom estado de desempenho. Seguindo o algoritmo, o avaliador deve escolher as opções que melhor representam sua experiência de manuseio. Caso tenha sido necessária excessiva interferência, o veículo necessita de alterações de projeto para que haja menor carga de trabalho e redução do risco de acidentes. Do contrário, a aeronave está em condições excelentes e não necessita de melhorias. O piloto tem ainda a possibilidade de incluir notas e observações quanto à missão executada para melhor levantamento de ameaças à qualidade do produto em estudo.

Esta escala é utilizada em VANT para verificar o grau de controlabilidade requerida pelo sistema autônomo ou semi-autônomo [**?**]. É como uma avaliação preliminar para levantamento de requisitos para o projeto do sistema de controle. Os níveis, a escolha da característica e a nota da avaliação podem dizer muito quanto a um sistema de controle implementado ou quanto ao grau de interferência que o projeto deste sistema deve realizar na aeronave para que haja menor, ou nenhuma, carga de trabalho por parte do condutor. Com o sistema de aquisição de dados proposto será possível avaliar as condições de voo em conjunto com a Escala Cooper-Harper. A aquisição de dados torna a avaliação menos subjetiva e otimiza tempo no momento de implementar um sistema de controle. Com os dados em mãos, a equipe de trabalho e o piloto avaliador podem buscar a melhor maneira de estudar os algoritmos a serem desenhados para determinadas trajetórias, circunstâncias de voo, procedimentos e manobras de teste realizadas durante a operação conduzida.

### C. Aeroelasticidade

**Torção** é um tipo deformação que ocorre quando um *torque*, também chamado de momento torsor, é aplicado no componente estrutural de modo a torcê-lo [6]. Quando da ocorrência deste torque, linhas longitudinais no componente são distorcidas. Se o componente tiver geometria circular, seções transversais ao longo do componente continuarão as mesmas após a deformação por torção, o que não ocorre com componentes de geometrais distintas como retangular, prismática e entre outras. A Fig. 2 representa um componente estrutural de geometria retangular sofrendo torção. Podem ser observadas as linhas longitudinais e as seções transversais se deformando devido ao torque aplicado.

O **ângulo de torção** pode ser definido como sendo o ângulo que um elemento do material em uma dada posição será rotacionado em relação a outro elemento do componente

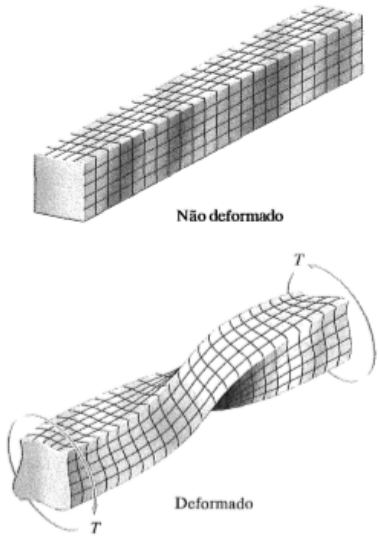



Fig. 2. Deformação por cisalhamento de um componente estrutural de geometria retangular [6].

estrutural. A expressão para o cálculo do ângulo de torção encontra-se na Equação 1 a seguir.

$$\phi(x) = \int_0^L \frac{T(x)}{J(x)G} dx \tag{1}$$

Em que $\phi(x)$ é o ângulo de torção, $L$ é o comprimento do componente estrutural, $T(x)$ é o torque interno que age na seção transversal, $J(x)$ é o momento polar de inércia da área da seção transversal, $G$ é o módulo de rigidez ou módulo de cisalhamento do material e $x$ é uma posição arbitrária.

A Equação 2 retirada de [6] apresenta o resultado da análise de torção para o cálculo do ângulo de torção de um componente estrutural de seção transversal quadrada de lado $l$ em uma posição arbitrária $x$.

$$\phi(x) = \frac{7.10T(x)}{l^4 G} \tag{2}$$

O termo $T(x)$ pode ser isolado nesta expressão para obter a Equação 3.

$$T(x) = \frac{\phi(x) l^4 G}{7.10} \tag{3}$$

Com esta equação é possível calcular o torque interno que age na seção transversal em um ponto arbitrário $x$ à partir do ângulo de torção. Na prática, o cálculo do ângulo de torção e torque interno é realizado por uma simulação numérica utilizando o Método de Elementos Finitos (MEF). Os dados resultantes da simulação são posteriormente comparados com os valores adquiridos pelos ensaios em laboratório.

No sistema proposto, os sensores IMU medirão o ângulo de torção na longarina da asa do VANT. A diferença de posição angular entre um sensor IMU posicionado num ponto $x_o$ da envergadura da asa e um sensor IMU posicionado no centro da aeronave (ponto de apoio central da longarina) resultará neste ângulo de torção. Com este dado, pode ser calculado, de maneira aproximado com a Equação 3, o torque interno no ponto $x_o$.

O VANT MMT003, no qual serão conduzidos os testes deste sistema, possui longarina de seção transversal retangular de fibra de carbono composta com alma de divinycell® de acordo com a Fig. 3. Observa-se que a seção transversal da longarina varia na extremidade. Sendo assim, os sensores serão instalados em um ponto anterior a esta mudança de geometria. A justificativa é reduzir a complexidade computacional para este protótipo do sistema.

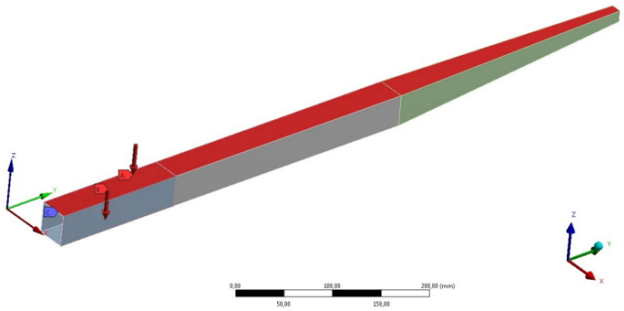


Fig. 3. Longarina do VANT MMT003. [**?**]

**Flexão**, assim como a torção, é um tipo de deformação que ocorre quando um momento fletor é aplicado a um componente estrutural. Sendo assim, as linhas longitudinais do objeto ficam curvadas e as linhas transversais se deformam de modo que um lado comprime e o outro alonga. Na Fig. 4 encontra-se uma ilustração deste fenômeno.

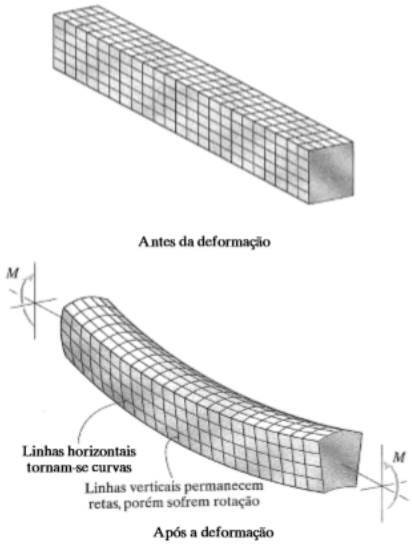



Fig. 4. Deformação por momento fletor de um componente estrutural de geometria retangular [6].

### *D. Sensores IMU*

Os sensores utilizados neste trabalho são o MPU-6050 e o MPU-9250 da *InvenSense*. O MPU-6050, Fig. 5, é um conjunto de giroscópio de 3 eixos, acelerômetro de 3 eixos e um micro-processador integrados em um único Circuito Integrado (CI).

O MPU-9250, componente primo do MPU-6050, Fig. 6, conta com um módulo magnetômetro na segunda camada do chip que permite a realização de medidas de altitude do componente para combinar com os dados adquiridos pelo acelerômetro e giroscópio. Ambos módulos possuem a tecnologia *Digital Motion Processor* ™ (DMP) inclusa que permite uma etapa de pré-processamento dos dados colhidos pelos sensores antes de enviá-los pelo protocolo $I^2C$.

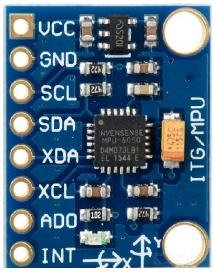



Fig. 5. Sensor MPU-6050 da *InvenSense*

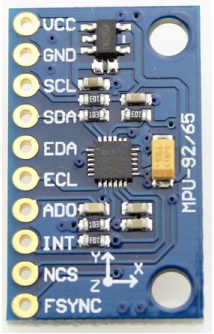



Fig. 6. Sensor MPU-9250 da *InvenSense*

**Giroscópios** são unidades de medição inercial (IMU) que respondem a uma mudança de posição angular em relação ao tempo. Portanto, os dados adquiridos por um giroscópio correspondem a uma derivada da posição angular em relação ao tempo (velocidade angular). Nos sensores utilizados, a escala do giroscópio pode ser ajustada para $\pm$ 250, $\pm$ 500, $\pm$ 1000, ou $\pm$ 2000 $^\circ$ por segundo.

Para adquirir a posição angular do giroscópio deve ser realizada uma integral do dado obtido pelo tempo de acordo com a Equação 4 onde $\theta(t)$ é a posição angular, $\omega(t)$ é a velocidade angular medida pelo giroscópio, $T$ é um intervalo de tempo e $dt$ é um valor infinitesimal de tempo.

$$\theta(t) = \int_T \omega(t) * dt \tag{4}$$

Por se tratar de um sistema digital, a integração realizada deve ser numérica com valores discretos. O resultado será aproximado do valor da integral contínua cuja estimativa do erro será avaliada mais adiante. A equação utilizada na Raspberry Pi deverá ser a Equação 5 abaixo.

$$\theta(t) \approx \sum_{n=0}^{N} \frac{\omega[n]}{fs} + \varepsilon[n] \tag{5}$$

Em que $n$ é o número de amostras, $fs$ é a frequência de amostragem utilizada para adquirir os dados do sensor, $\varepsilon[n]$ é a estimativa do erro e $N$ é o período em que foram adquiridas as amostras.

Em geral, recomenda-se o uso de $fs$ na ordem de 100 Hz a 200 Hz devido a lenta resposta de um sistema mecânico [7]. Entretanto, dado o propósito deste trabalho, será atualizado o valor de $fs$ de acordo com a medição a ser efetuada. Isto é, caso a medida efetuada tenha como objetivo a **torção** ou **flexão** da asa, a frequência de amostragem pode estar entre 100 Hz e 200 Hz; Caso a medida efetuada tenha como objetivo a **vibração**, a frequência de amostragem utilizada deve estar de acordo com o teorema de amostragem de Nyquist-Shannon para a captura das frequências naturais da vibração na aeronave.

Choques e vibrações no sistema irão influenciar na medição de torção e flexão da asa. Na ocorrência de perturbações no sistema de forma que $fs$ não respeite o teorema de Nyquist-Shannon ocorrerá o fenômeno de *aliasing* e a medição apresentará erro por deriva, também chamado de *drift* [7]. Uma maneira de lidar com o erro por *drift* é utilizar uma espécie de filtro que seja capaz de aplicar correções à deriva do sinal sem que seja alterada a informação de interesse. Uma proposta a ser avaliada é a implementação do filtro de Kalman [8] ou um filtro complementar como sugerido em [**?**], [9] e [10].

**Acelerômetros**, assim como Giroscópios, são IMU capazes de medir a aceleração a que estão submetidos. Consistem de uma massa de prova e sensores capacitivos. Quando a massa de prova é deslocada a diferença de capacitância é detectada e convertida. A saída deste sensor é a aceleração e são necessárias duas etapas de integração numérica para adquirir o deslocamento ao qual o sensor foi submetido. Por ser suscetível a acelerações, é comum que os dados de um acelerômetro sejam expressos em função da aceleração da gravidade $g$. Para testes em Brasília, será utilizada a aceleração da gravidade com valor de $9,7808\ m/s^2$. Nos sensores utilizados, a escala do acelerômetros pode ser ajustada para $\pm 2g$, $\pm 4g$, $\pm 8g$ ou $\pm 16g$.

### *E. Global Positioning System (GPS)*

O Sistema de Posicionamento Global, do inglês *Global Positioning System* (GPS), é um sistema de geoposicionamento desenvolvido pelo Departamento de Defesa (DoD) dos Estados Unidos. Este sistema foi desenvolvido para determinar, de maneira acurada, a posição, velocidade e tempo em um sistema comum de referências [11]. Este sistema utiliza ao menos quatro satélites em órbita e um receptor dos sinais vindos destes satélites. O dispositivo receptor calcula a distância que os satélites estão e estima sua posição utilizando um algoritmo chamado trilateração [**?**]. Os dados recebidos pelo GPS seguem o protocolo NMEA 0183 [**?**] e dizem respeito ao tempo (os satélites possuem um relógio atômico interno para ajustes devido à relatividade do tempo na órbita da Terra) e sua posição a uma constante taxa de amostragem.

Trilateração é um método de estimar a localização de um vetor usando a geometria de circunferências, para o caso bidimensional, ou esferas, para o caso tridimensional [**?**]. Quando os dados são recebidos, calcula-se as esferas aproximadas nas quais é possível que o receptor se encontre. A interseção

entre estas esferas, a grosso modo, será a posição na Terra aproximada do receptor.

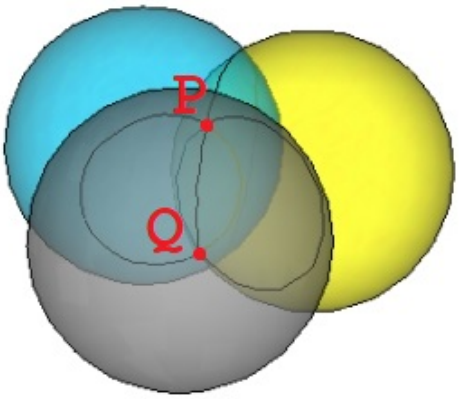



Fig. 7. Exemplo gráfico tridimensional da localização por trilateração. As interseções das esferas (pontos P e Q) são possíveis localizações do receptor GPS.

São utilizados, no mínimo, quatro satélites para calcular a posição do receptor. Desta forma, a posição calculada é muito próxima da localização exata. Quanto maior o número de satélites identificados, mais informações serão receptadas e com maior acurácia ocorrerá a estimativa da localização do dispositivo. Outros dados são derivados destes cálculos como a velocidade, altitude e direcionamento. Uma falha no sistema de GPS é sua indisponibilidade em ambientes fechados como em prédio e túneis, ou obstáculos como árvores e pontes. Sendo assim, sistemas que necessitam estimar sua localização utilizam a fusão dos dados de um GPS e de acelerômetros e giroscópios [8].

O Módulo GPS utilizado neste trabalho será o módulo da u-blox NEO-6M apresentado na Fig. 8. De acordo com [**?**], a alimentação deste módulo é 3.3V e possui capacidade para interfaceamento com a Raspberry Pi por meio dos protocolos UART, SPI, USB e $I^2C$. Para o projeto foi escolhida a comunicação por meio de UART por não depender do endereçamento da $I^2C$, consumido com os três sensores IMU, não utilizar muitos fios para conexão e por não estar sujeito ao congestionamento da linha de comunicação entre os dispositivos conectados na $I^2C$.

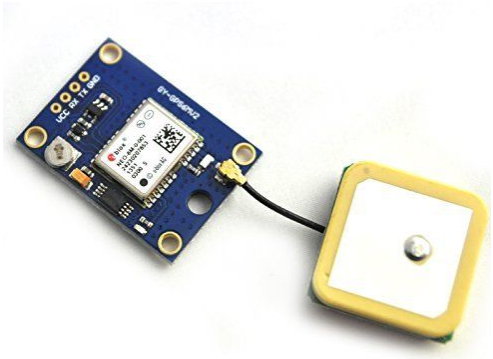


Fig. 8. Módulo GPS NEO-6M da u-blox.

| Quantidade | Componente |
|---|---|
| 1 | Raspberry Pi 3 Model B+ |
| 2 | MPU-6050 |
| 1 | MPU-9250 |
| 1 | GPS ublox NEO-6M |
| 1 | Buzzer Ativo |
| 1 | Switch |
| 2 | Resistor 1K $\omega$ |
| 9 | Jumper fêmea-fêmea |
| 16 | Jumper macho-fêmea |
| 3 | Jumper macho-macho |
| 1 | Cabo USB |
| 1 | Bateria Externa |

## VI. Hardware

Os componentes a serem utilizados estão listados na Tabela VI-A a seguir. O GPS está conectado à SoC pela porta serial UART ttyS0. Os sensores MPU-6050 conectam-se à Raspberry Pi por meio do protocolo IC. O sensor posicionado na asa direita possui endereço IC 0x68 e o sensor posicionado na asa esquerda possui endereço IC 0x69. O sensor MPU-9250 posicionado no centro da aeronave possui endereço padrão SPI. O buzzer está na porta GPIO 7 e a saída para modificação do endereço do sensor MPU-6050 da asa esquerda está na porta GPIO 18.

### A. *BOM*

### B. *Esquemático*

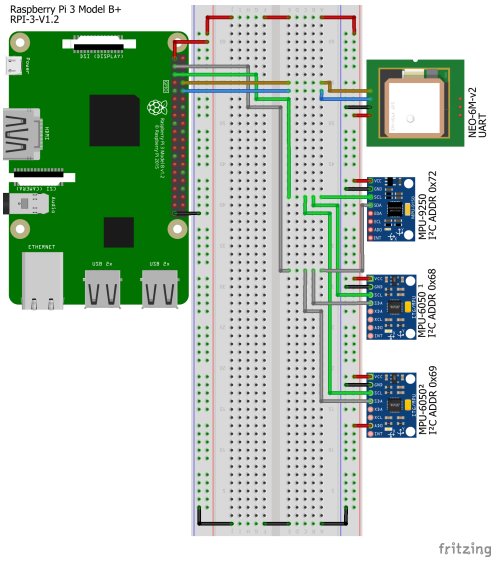



Fig. 9. Diagrama do sistema realizado no software *fritzing*.

## VII. Software

O código foi construído na linguagem C, por exceção dos módulos de interface com os sensores IMU, que utiliza recursos do C++. A compilação deste código é realizada por meio de um makefile que cria os objetos para cada biblioteca e módulo utilizado, realiza o link destes objetos e cria um arquivo executável.

### A. *Algoritmo*

- Sensores IMU Para interfaceamento dos sensores IMU com a Raspberry Pi foi utilizada a biblioteca RTIMULib, do usuário do github Richard Barnett, que

realiza a calibração e leitura dos sensores e, posteriormente, o processamento dos dados de cada sensor individual de acordo com um arquivo de configuração. Este arquivo de configuração é criado no momento da calibração dos sensores ou é gerado automaticamente caso não sejam encontrados arquivos de configuração. Foi criada uma pasta para cada IMU com seu respectivo arquivo de configuração, uma vez que esta biblioteca apenas reconhece "RTIMULib.ini" como arquivo de configuração. Este arquivo possui os dados da porta a ser acessada para aquisição dos dados do sensor e os parâmetros de configuração, calibração e processamento. Como referido na Seção VI, um MPU-6050 é posicionado na asa direita com endereço IC 0x68, outro é posicionado na asa esquerda com endereço IC 0x69 e o MPU-9250, posicionado no centro da aeronave, possui endereço padrão SPI.

No módulo principal são criados três ponteiros para objetos da classe RTIMU e um ponteiro para uma struct local do tipo "imuDataAngulo" que receberá os dados processados pela biblioteca RTIMULib. Um contador também é declarado para identificação de qual sensor IMU estará sendo lido no momento. Estes objetos, a struct e o contador serão passados para as funções do módulo de implementação dos sensores IMU que funciona como um *wrapper* para as funções da biblioteca utilizada.

Um teste foi realizado utilizado o aplicativo padrão RTIMULibDemo para o sensor MPU-6050, Fig. 10, e para o sensor MPU-9250, Fig. 11. Posteriormente a estes testes fora realizada a calibração de cada sensor e suas configurações de inicialização salvas em uma pasta separada, como citado anteriormente.

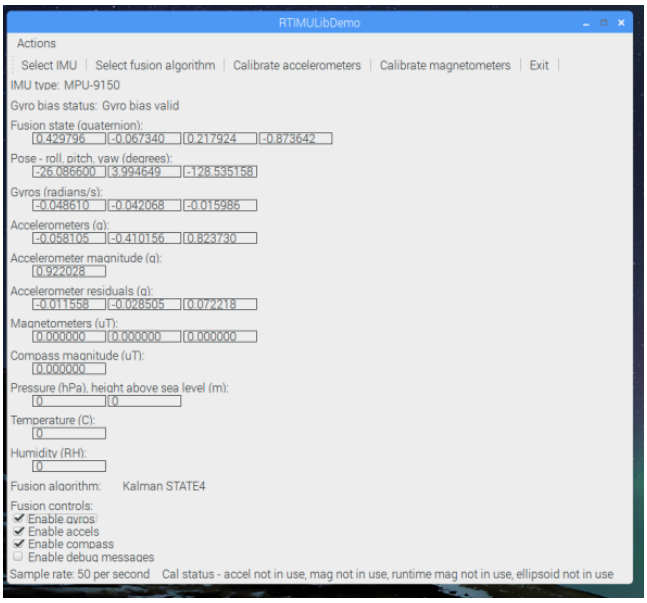



Fig. 10. Teste do sensor MPU-6050 utilizando aplicativo padrão.

- Receptor GPS Foi utilizado o *daemon* GPSD para interfaceamento da Raspberry Pi com o GPS. Este serviço monitora os receptores GPS e traduz suas informações de padrão NMEA para variáveis, utilizando uma biblioteca em C para uso em aplicações externas, como a proposta por este projeto. A biblioteca possui a opção de se realizar o *polling* do GPS por meio do protocolo TCP ou por meio de variáveis locais no sistema operacional. Prezando pela redução de *overhead* no sistema, foi utilizada a opção de *polling* com a variável local. Esta opção é escolhida pela macro $GPSD_SHARED_MEMORY$ enviada como primeiro argumento da função $gps_open()$ da biblioteca.

  Executou-se um teste utilizando um aplicativo chamado "cgps", padrão da distribuição do serviço GPSD. O resultado deste teste pode ser observado na Fig. 12. As informações de tempo, latitude e longitude foram desfocadas para evitar rastreamento da localização em que foi realizado este teste. Foi passado "-s" como argumento para suprimir as informações das strings no padrão NMEA recebidas pelo gps e concatenadas para a saída do console do terminal. Nesta e demais aplicações o tipo de FIX é importante para escolher quais informações possuem 95% de confiabilidade.

  - FIX 0 - Não existem informações suficientes dos satélites para compor alguma variável;
  - FIX 2 - Apenas informações de tempo, velocidade horizontal, latitude, longitude e seus respectivos erros possuem confiabilidade suficiente; e
  - FIX 3 - Todas as informações repassadas pelo receptor GPS são confiáveis e podem ser utilizadas.

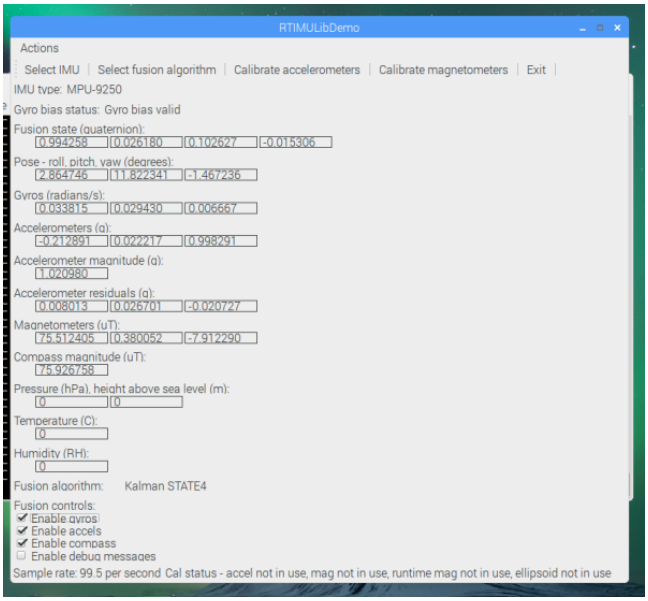



Fig. 11. Teste do sensor MPU-9250 utilizando aplicativo padrão.

## REFERENCES

[1] B. Custers, *The Future of Drone Use: Opportunities and Threats from Ethical and Legal Perspectives*, ser. Information Technology and Law Series. T.M.C. Asser Press, 2016. [Online]. Available: https://books.google.co.il/books?id=WytEDQAAQBAJ

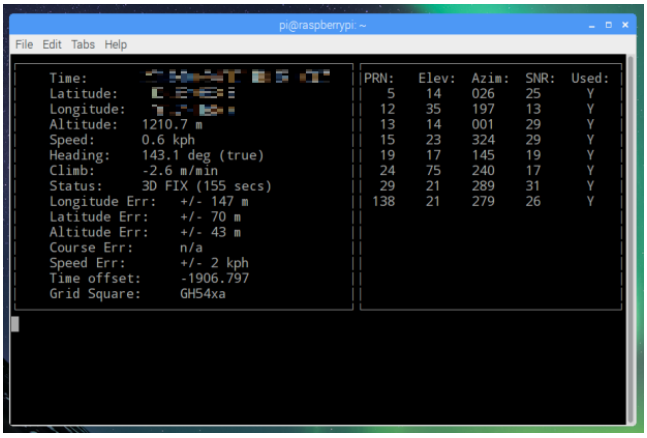



Fig. 12. Resultado do teste do GPS com o *daemon* GPSD.


[2] H. Murrow and C. Eckstrom, "Drones for aerodynamic and structural testing," in *Aircraft Systems and Technology Conference*. NASA Langley Research Center: National Aeronautics and Space Administration, Aug 1978.

[3] W. Wagner, *Lightning Bugs and Other Reconnaissance Drones*. Armed Forces Journal, 1982. [Online]. Available: https://books.google.com.br/books?id=L-xzQgAACAAJ

[4] R. A. V. Gimenes, "Método de avaliação de segurança crítica para a integração de veículos aéreos não tripulados no espaço aéreo controlado e não segregado," Escola Politécnica, 2015.

[5] G. E. Cooper and R. P. Harper, "The use of pilot rating in the evaluation of aircraft handling qualities," National Aeronautics and Space Administration, Washington D.C., Tech. Rep. NASA-TN-D-5153, April 1969.

[6] R. C. Hibbeler, *Resistência dos Materiais*, 7th ed. Pearson Prentice Hall, 2010.

[7] P.-J. V. de Maele, "Getting the angular position from gyroscope data," Disponível em: https://www.pieter-jan.com/node/7, Sept 2012, acessado em: 24/07/2018.

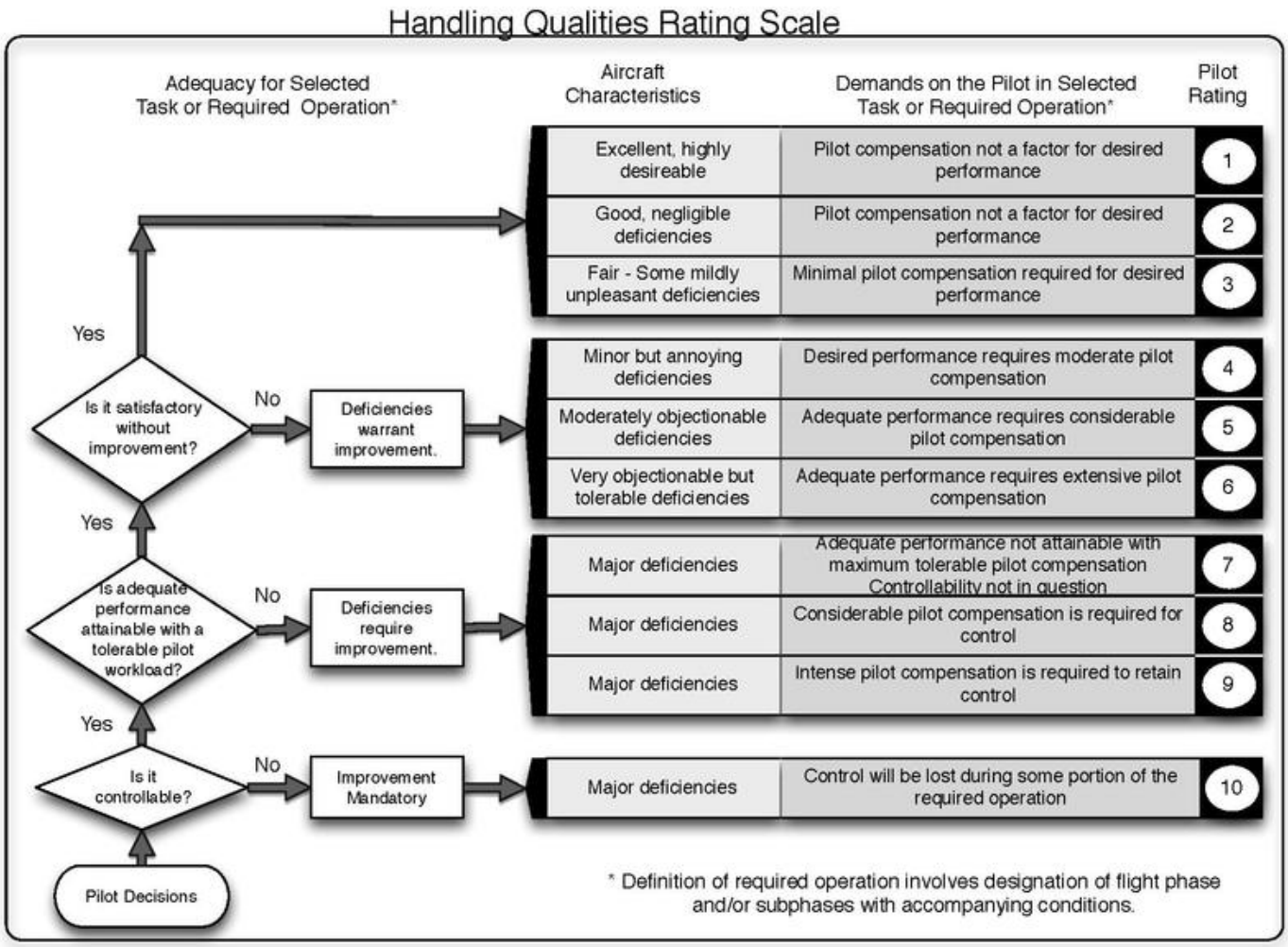
[8] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB*, 2nd ed., ser. Wiley-Interscience. John Wiley & Sons, Inc., 2001.

[9] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, "A complementary filter for attitude estimation of a fixed-wing uav," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2008, pp. 340–345.

[10] P.-J. V. de Maele, "Reading a imu without kalman: The complementary filter," Disponível em: https://www.pieter-jan.com/node/11, April 2013, acessado em: 24/07/2018.

[11] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System: Theory and Practice*. Springer Vienna, 2012. [Online]. Available: https://books.google.com.br/books?id=F7jrCAAAQBAJ

APPENDIX A
ESCALA DE AVALIAÇÃO COOPER-HARPER



# APPENDIX B
## QUADRO PLANEJAMENTO

| **#Hashtag** | Sistema de Aquisição de Dados | | | |
|---|---|---|---|---|
| **Equipe**<br><br>Arthur Evangelista dos Santos (14/0016686)<br><br>Fábio Barbosa Pinto (11/0116356) | **Datas**<br>PC1 - 29/03<br>PC2 - 29/04<br>PC3 - 27/05<br>PC4 - 10/06<br>FINAL - 05/07 | **Por que?**<br><br>A aquisição de dados pode ser realizada por um simples MCU. Entretanto, separar os dados e processá-los deve ser realizado com um computador numa *groundstation* e consome tempo e recursos humanos. A automatização destes processos pode ser realizada com um SoC (no presente trabalho uma *raspberry pi*) durante o voo. Com o auxílio de um GPS e um IMU pode ser traçada a trajetória de um VANT. Para automatização de voo deste VANT, os dados adquiridos pelo AHRS (conjunto do GPS, IMU e SoC para aquisição de dados de voo) podem ser utilizados para o algoritmo utilizado no sistema de controle da aeronave.<br><br>**Por que não?**<br><br>Talvez não seja possível implementar todos as características propostas para o projeto até a data limite. Ademais, a ideia deste sistema de aquisição de dados não é original, sendo algo implementado e estudado pela indústria aeroespacial desde o lançamento da missão Apolo à Lua. Ou seja, já existem soluções na indústria para o problema apresentado. Outro impecilho são os custos elevados para obtenção dos sensores e componentes para o projeto. Talvez o barateamento dos custos afete a precisão dos dados adquiridos. | **Objetivos**<br>- Adquirir dados de flexão e torção da asa;<br>- Adquirir velocidade da aeronave;<br>- Adquirir dados de vibração da asa;<br>- Adquirir altitude, ângulo de atitude e ângulo de ataque;<br>- Fusão dos dados dos acelerômetros e dos giroscópios;<br>- Processamento dos dados adquiridos;<br>- *Plot* da FFT, PDS e espectrograma (FFT/tempo);<br>- Fusão dos dados do IMU e do GPS com o Filtro de Kalman;<br>- Organizar dados de acordo com o procedimento de voo realizado;<br>- Apresentar resultados em uma GUI para o usuário; | **Ameaças ao Projeto**<br>- Não implementação de todas as características a tempo;<br>- Propagação do erro de medida e do erro por deriva;<br>- Ruído nos componentes e na PCB e incompatibilidade EM;<br>- Matemática avançada necessária para implementação do filtro de Kalman;<br>- Aliasing devido à taxa de amostragem X modos de vibração;<br>- Portas I2C insuficientes;<br>- GPIO insuficiente;<br>- Temporização falha devido ao OS (possível solução seria o uso da abordagem RTOS);<br>- CPU insuficiente para processamento de todos os dados + GUI;<br>- Custo X Precisão; |

**Custos**

- Raspberry pi 3 Model B+ [R$ 190,00]
- Case impresso 3D [Indefinido]
- 2 x MPU-6050 [R$ 19,90]
- Tela para raspi (touch ou não) [Indefinido]
- Módulo GPS [R$ 119,90]
- Jumpers [Indefinido]
- MPU-9250 [R$ 99,90]

**Requisitos**

- Velocidade, aceleração e posição (linear e angular) da aeronave e de cada meia asa [accel]
- Ângulo de atitude e ângulo de ataque da aeronave [gyro]
- Altitude, posição e trajetória da aeronave (fusão dos dois últimos requisitos com módulo gps) [accel + gyro + gps]
- Operações matemáticas (FFT, arctg, plot de gráficos) [octave, matlab, scilab]
- Implementação do Filtro de Kalman [octave, matlab, scilab]
- GUI apresentando trajetória e dados adquiridos [GTK+, Visual Studio, Processing]

## APPENDIX C
## DIAGRAMA LÓGICO DO SISTEMA

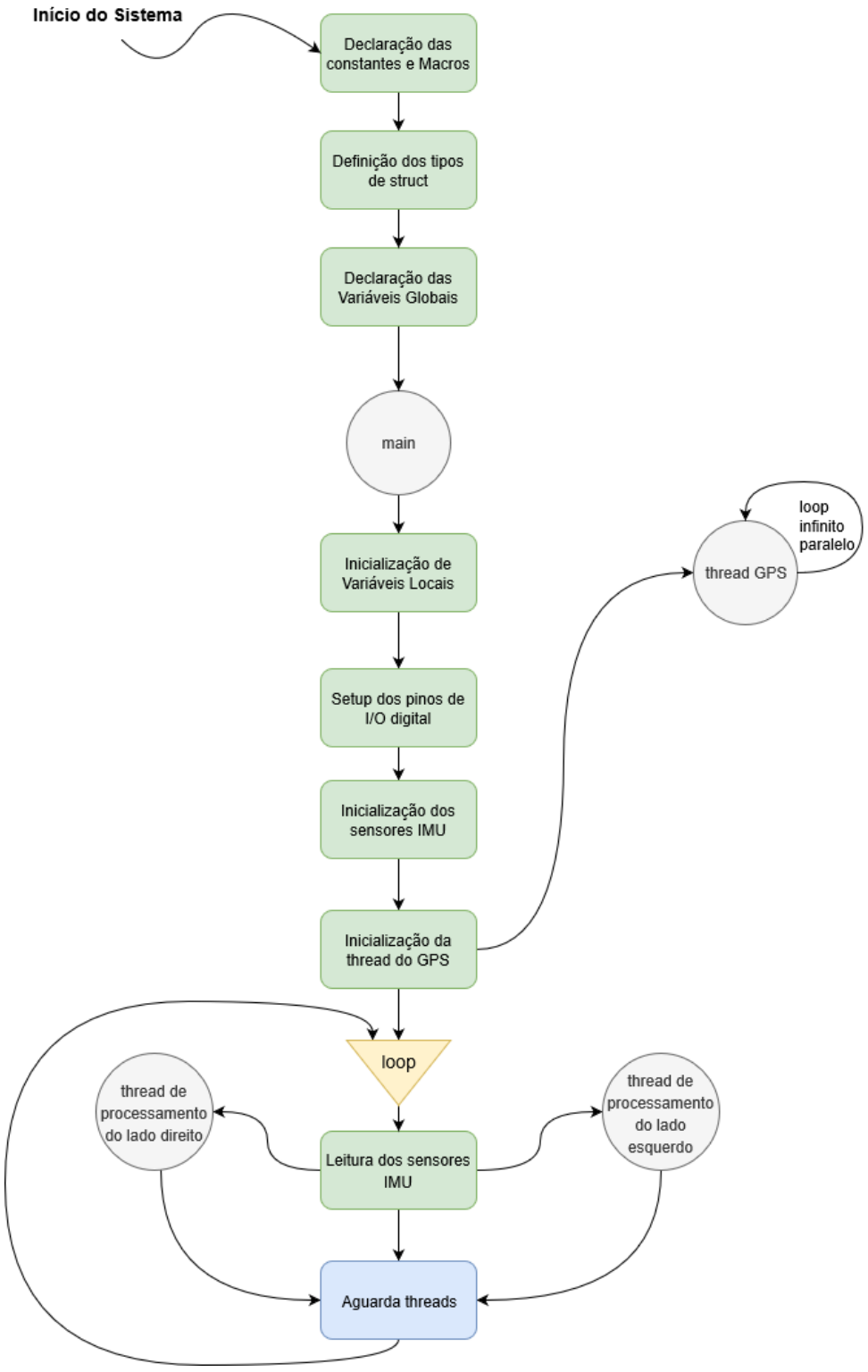



Fig. 13. Diagrama lógico do sistema.