Lúcio André Amorim

Montagem de mosaico com imagens aéreas para auxílio a equipes em operações de combate, prevenção ou perícias em incêndios florestais utilizando *drone*

Vitória - ES

2016

Lúcio André Amorim

Montagem de mosaico com imagens aéreas para auxílio a equipes em operações de combate, prevenção ou perícias em incêndios florestais utilizando *drone*

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Universidade Federal do Espírito Santo – UFES

Programa de Pós-Graduação em Engenharia Elétrica

Orientador: Dr. Mário Sarcinelli Filho

Coorientador: Dra. Raquel Frizera Vassallo

Vitória - ES

2016

Lúcio André Amorim

Montagem de mosaico com imagens aéreas para auxílio a equipes em operações de combate, prevenção ou perícias em incêndios florestais utilizando *drone*/ Lúcio André Amorim. – Vitória - ES, 2016-

64 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Mário Sarcinelli Filho

Disertação (Mestrado) – Universidade Federal do Espírito Santo – UFES
Programa de Pós-Graduação em Engenharia Elétrica , 2016.

1. Palavra-chave1. 2. Palavra-chave2. 2. Palavra-chave3. I. Orientador. II. Universidade xxx. III. Faculdade de xxx. IV. Título

Lúcio André Amorim

Montagem de mosaico com imagens aéreas para auxílio a equipes em operações de combate, prevenção ou perícias em incêndios florestais utilizando *drone*

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Aprovada em 20/10/2016.

Dr. Mário Sarcinelli Filho - Orientador
Universidade Federal do Espírito Santo

Dra. Raquel Frizera Vassallo - Coorientadora
Universidade Federal do Espírito Santo

Dra. Eliete Maria de Oliveira Caldeira
Universidade Federal do Espírito Santo

Dr. Lucas Vago Santana
Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo - Campus Linhares

Vitória - ES
2016

Dedico este trabalho a todos que estiveram ao meu lado nesse percurso em especial à minha família e amigos.

Agradecimentos

À minha esposa Nalva dos Santos Amorim e aos meus filhos André Felipe Candeas Amorim e Lúcio André Amorim Júnior por estarem sempre ao meu lado e servirem de incentivo para minha dedicação.

Aos meus pais Valci Amorim e Luzia Pereira Amorim, por serem exemplos de vida e nunca deixarem de acreditar em mim.

À Profa. Dra. Raquel Frizera Vassallo, por ser a peça chave para minha aceitação no Programa de Pós Graduação de Engenharia Elétrica - PPGEE/UFES onde, além de me recomendar ao programa, apadrinhou-me na situação de aluno especial. Ainda, confiando a mim, equipamentos de valores relativamente altos, com grande risco de quebra durante os experimentos, sem os quais, seria impossível a realização deste trabalho. Agradeço ainda pelas orientações na área de visão computacional, do direcionamento inicial, do auxílio durante todo o projeto, e é claro, de proporcionar um excelente ambiente de trabalho no laboratório.

A meu orientador, Prof. Dr. Mário Sarcinelli-Filho, por me aceitar no grupo de pesquisa após o cumprimento dos créditos necessários ao programa, auxiliar-me com as dúvidas sobre o projeto, e principalmente por motivar na obtenção dos resultados, publicações dos artigos e conclusão deste trabalho.

À Profa. Dra. Eliete Maria de Oliveira Caldeira, que acreditou no meu potencial, explicitando isto claramente em sua carta de recomendação ao PPGEE.

Ao Prof. M.Sc. Milton Cesar Paes Santos, por me apresentar à professora Raquel e auxiliar em seu convencimento para me aceitar no programa. Por, juntamente comigo, realizar a publicação do meu primeiro artigo da pesquisa.

Ao Prof. M.Sc. Fabricio Bortolini de Sá, por aprimorar e configurar o quadrimotor utilizado nos experimentos iniciais, deixando-o praticamente a prova de quebra.

Ao colega de mestrado Leonardo de Assis Silva e aos alunos de iniciação científica Felippe Mendonça de Queiroz e Ricardo da Mota Salvador, por auxiliarem de forma ímpar nos experimentos. Por reprojetarem e reconstruírem a segunda versão do quadrimotor pois sem isto a obtenção dos resultados seria bem mais difícil. Pelo excelente auxílio do Felippe na configuração dos computadores Linux, Raspberry e dos ambientes de desenvolvimento.

A todos os colegas de LAI, VIROS que, de alguma forma incentivaram, auxiliaram ou até mesmo se fizeram presentes para que o ambiente ficasse mais agradável.

Ao Corpo de Bombeiros Militar do Espírito Santo - CBMES, seus Oficiais e Praças

que de alguma forma contribuíram para este resultado.

Em especial aos seguintes Oficiais do CBMES: Ten Cel Hekssandro Vassoler, meu então chefe imediato no início do mestrado e que me apoiou inteiramente, não colocando obstáculo que dificultasse meus trabalhos, inclusive auxiliando a transpor alguns já existentes; Ten Cel André Có Silva, que foi a peça chave para flexibilização do meu horário de trabalho junto ao Comando do CBMES por contribuir de forma significativa para viabilizar a publicação dos meus artigos.

"Por vezes sentimos que aquilo que fazemos não é senão uma gota de água no mar. Mas o mar seria menor se lhe faltasse uma gota."

Anjezë Gonxhe Bojaxhiu
(Madre Teresa de Calcutá)

Resumo

Este projeto tem como objetivo desenvolver um sistema de coleta, processamento e disponibilização de informações a partir de imagens aéreas capturadas por um robô aéreo, conhecido como VANT (Veículo Aéreo Não Tripulado), para auxiliar uma equipe em operações de combate, prevenção ou realizando perícia, em casos de incêndios florestais. O objetivo principal é desenvolver uma estação de processamento em terra e programar o computador de bordo do robô aéreo. A estação de terra é conectada ao VANT através de um link de telemetria, o qual armazena os dados de GPS (*Global Positioning System*) quando necessário. De posse da posição e orientação iniciais do VANT, além de alguns parâmetros informados pelo operador (tais como altitude de voo e área a ser coberta), a estação em terra gera um *grid* de *waypoints*, os quais são encaminhados ao computador de bordo do robô. O VANT, após decolar e atingir a altitude de voo definida, sobrevoa todos os *waypoints*, ao mesmo tempo que envia seus dados de posicionamento tridimensional a seu computador de bordo. O sistema embarcado supervisionará e armazenará os dados recebidos do robô, e quando identificar que está sobre o ponto desejado fará uso de sua câmera, realizando assim a aquisição de uma fotografia e armazenando-a. Ao final do voo, os dados e imagens serão repassados para a estação de terra, a qual os processará utilizando algoritmos robustos. O resultado final do processamento será a disponibilização, para o operador, de um mosaico da região do voo, com os pontos de aquisição das imagens georreferenciados. Nesta dissertação serão mostrados os passos do desenvolvimento dos sistemas de comunicação, de aquisição de imagens, de geração de *waypoints* e montagem do mosaico, bem como os algoritmos utilizados para tal. Por fim, serão mostrados alguns experimentos onde o robô realizou sobrevoo de uma região, com a posterior geração do mosaico, validando assim o projeto proposto.

Palavras-chave: Robótica Móvel, Robótica Aérea, Quadrimotor, Mosaico, Imagens Aéreas

Abstract

In this work we developed a system to collect, process and provide information extracted from aerial images captured by an aerial robot, known as UAV (Unmanned Aerial Vehicle). The focus is to assist a team of firefighters in operations of fire extinguishing, prevention or investigation. The project consisted on programming a ground processing station and a computer onboard the aerial robot. The ground station is connected to the UAV through a telemetry link, and collects and stores GPS data supplied by a GPS unit onboard the vehicle. From the starting position and orientation of the UAV, as well as some parameters defined by the operator (such as altitude of flight and area to be supervised), the ground station defines a grid of 3D waypoints to be covered by the aerial robot during the flight. Such grid of waypoints is then delivered to the UAV. After that, the UAV goes to each waypoint, always sending its 3D position to its onboard computer, which controls the vehicle to reach every next planned point. When the aerial robot gets to a waypoint, the onboard computer keeps the vehicle hovering for a while in order to take a picture with a camera attached to the UAV. All the images are stored in the memory of the onboard computer. Upon finishing the flight, the waypoint images and their corresponding GPS data are downloaded to the ground station, which processes the data using robust algorithms. The result of such processing is a mosaic of the region over which the UAV flew, with the GPS coordinates of the waypoints included. The obtained mosaic is made available for the operator. The steps involved in the development of the systems responsible for the communication between the ground station and the UAV, the image acquisition, the waypoint generation and mosaic building, including the algorithms adopted to build the mosaic, are all discribed in this dissertation. Finally, experiments are shown where the UAV flies over a region and the corresponding mosaic is generated, thus validating the functionality of the project.

Keywords:Mobile Robotics, robotics Air, Mosaico, Aerial images

Lista de ilustrações

Figura 1 – *Drone* utilizado no início dos experimentos. 20
Figura 2 – APM 2.6 2(a) e APM + case 2(b). 21
Figura 3 – GPS da 3DR. 21
Figura 4 – Motor 4(a) e hélice 4(b) utilizada no VANT. 22
Figura 5 – ESC de 20A. 22
Figura 6 – Bateria de três células com capacidade 5200mAh e 30C. 22
Figura 7 – Rádio Turngy 9XR de nove canais. 23
Figura 8 – Módulos FRSKY de transmissão 8(a) e recepção 8(b). 23
Figura 9 – Link de telemetria 915 MHz. 24
Figura 10 – RaspberryPi B+ e seu módulo de câmera. 25
Figura 11 – Segunda vesão do *drone* utilizado. 25
Figura 12 – Capa para acoplar a câmera à *gimbal* 12(a) e o estabilizador de câmera 12(b). 26
Figura 13 – Padrão de quadrados de 20 cm x 20 cm utilizado nos experimentos externos. 27
Figura 14 – Os resultados da estimação do $[\xi\ \eta]$ pelo DKF, onde $\xi = [x\ y\ z] \in \Re^3$ representa os deslocamentos longitudinal (x), lateral (y) e normal (z), e $\eta = [\phi\ \theta\ \psi] \in \Re^3$ indica os ângulos de *roll* (ϕ), *pitch* (θ) e *yaw* (ψ). . . 28
Figura 15 – Estrutura das mensagens do Protocolo Mavlink. 29
Figura 16 – Diagrama do Subsistema de comunicação. 34
Figura 17 – Software da Estação de Terra com os parâmetros para missão. 36
Figura 18 – Software da Estação de Terra com os *waypoints* enviados. 37
Figura 19 – *Waypoints* gerados pela estação de terra. 37
Figura 20 – Protocolo Mavlink utilizado para escrita de *waypoints*. 38
Figura 21 – Protocolo Mavlink utilizado para leitura de *waypoints*. 39
Figura 22 – Pontos selecionados pelo SURF. 41
Figura 23 – *Matching* realizado pelo SURF. 42
Figura 24 – *Matching* realizado pelo SURF, aplicada a restrição de 3x a menor distância euclidiana encontrada no conjunto. 43
Figura 25 – Região a ser fotografada com o *drone* (Fonte Google Maps). 45
Figura 26 – A Figura 26(a) possui uma sobreposição de aproximadamente 50% e a Figura 26(b) possui uma sobreposição um pouco maior 65%. 46
Figura 27 – Montagem do mosaico com processamento de regiões de interesses. . . 47
Figura 28 – Erro de orientação na colagem da primeira imagem. 47
Figura 29 – Erro de orientação na colagem da primeira imagem. 48
Figura 30 – Cálculo do erro de orientação da primeira imagem. 49

Figura 31 – Colagem da primeira imagem da segunda linha. 50
Figura 32 – Colagem da segunda imagem da segunda linha. 51
Figura 33 – *Drone* sobrevoando o ponto central de uma região a ser fotografada. . . 51
Figura 34 – Montagem do mosaico sem a correção da primeira imagem. 52
Figura 35 – Montagem do mosaico com seis fotografias. 52
Figura 36 – Experimento com captura de 18 imagens a 65 metros de altura. 54
Figura 37 – Mosaico gerado utilizando 24 imagens capturadas a 100 metros de altura. 56
Figura 38 – Mosaico gerado utilizando 30 imagens capturadas a 100 metros de altura. 57
Figura 39 – Região de sobrevoo na ótica do Google Maps. 58
Figura 40 – Mosaico de 30 imagens da região de sobrevoo gerado pelo Autostich64. 59
Figura 41 – Mosaico de 30 imagens da região de sobrevoo gerado pelo Autostich64. 60

Lista de abreviaturas e siglas

CAN	*Controller Area Network*
DKF	*Differential Kalman Filter*
ESC	*Electronic Speed Control*
GPS	*Global Positioning System*
IMU	*Inertial Measurement Unit*
LAI	*Laboratório de Automação Inteligente*
LGPL	*Library General Public License*
LIPO	*Lithium-Polymer*
Mavlink	*Micro Air Vehicle Communication Protocol*
PPM	*Pulse Position Modulation*
RANSAC	*RANdom SAmple Consensus*
SIFT	*Scale Invariant Feature Transform*
SURF	*Speeded Up Robust Features*
VANT	*Veículo Aéreo Não Tripulado*

Sumário

1 INTRODUÇÃO 15
1.1 Motivação 15
1.2 Contexto 16
1.3 Objetivos 18
1.4 Publicações 18
1.5 Organização da Dissertação 19

2 EQUIPAMENTOS UTILIZADOS 20
2.1 *Drone*, sensores e equipamentos 20
2.1.1 Especificações do *Drone* 20
2.1.2 Rádio Controle 23
2.1.3 Link de rádio para telemetria 24
2.1.4 Computador de bordo 24
2.1.5 *Gimbal* e câmera 24
2.2 Validação do protótipo e dados coletados 26

3 SOFTWARES DESENVOLVIDOS 29
3.1 Protocolo Mavlink 29
3.2 Software de bordo 32
3.2.1 Rotina de comunicação com o *drone* 33
3.2.2 Rotina de armazenamento de dados 33
3.2.3 Rotina de controle de alto nível 33
3.3 Estação de terra 34
3.4 Construção do Mosaico 38

4 CONSTRUÇÃO DO MOSAICO 40
4.1 SURF 40
4.2 HOMOGRAFIA 41
4.3 RANSAC 43
4.4 Metodologia para geração dos mosaicos 44
4.5 Georreferenciamento 49
4.6 Exemplos 49

5 RESULTADOS 53
5.1 Experimento com 3x6 imagens 53
5.2 Experimento com 5x6 imagens 55

5.3 Comparação com o Google Maps 56
5.4 Comparação com a aplicação Autostich64 58

6 CONSIDERAÇÕES FINAIS 61
6.1 Conclusão 61
6.2 Trabalhos Futuros 62

REFERÊNCIAS 63

1 Introdução

Esta Dissertação de Mestrado está orientada no desenvolvimento de um sistema de coleta, processamento e disponibilização de informações a partir de imagens aéreas adquiridas por um *drone* (VANT - Veículo Aéreo Não Tripulado). As imagens serão montadas em forma de um mosaico, em um tempo suficiente para disponibilizá-lo em campo, para equipes em combate, perícia ou prevenção de incêndios florestais. Desta maneira, pode-se melhorar as tomadas de decisões, com o objetivo de minimizar o emprego de equipes de trabalho, sem com isto, diminuir sua eficácia. Este capítulo apresenta a motivação para realização deste trabalho, seus objetivos e trabalhos publicados. Ao final deste capítulo, será apresentada a informação em como esta Dissertação está organizada.

1.1 Motivação

Nos últimos anos, o número de incêndios em áreas de vegetação, no Brasil e no mundo vem aumentando, devido a diversos motivos ligados principalmente ao aumento da temperatura global. Observa-se uma média anual no período compreendido entre 1998 à 2015 de 175.870 focos de incêndios ativos, tendo seu pico em 2010 de 249.291 focos e nos anos de 2012, 2014 e 2015 tem-se a média de 193.838, 183.693, 236.371 focos, respectivamente (INPE, 2016). Fazendo uma análise geral dos dados, é nítido o aumento no número de focos com o passar dos anos.

O combate a incêndios em áreas de vegetação requer, em muitos casos, a utilização de equipamentos e máquinas de difícil logística. A tomada de decisão requer, entre outras coisas, informações referente ao contexto atual do evento, como a extensão da frente, tamanho, velocidade de propagação do fogo, que nem sempre está disponível devido à extensão da região ou limitação do campo visual. Uma boa prática é a utilização de aeronaves com o objetivo de aumentar o campo visual do responsável pela ação de combate, dando-o uma visão global do cenário (FIRELAB, 2016).

O uso de aeronaves, tais como aviões e helicópteros nem sempre é viável devido à disponibilidade e também ao alto custo de aquisição e operação. Um outro problema relacionado à disponibilidade destes é a sua exigência sazonal, onde em alguns meses do ano a demanda costuma ser maior do que a quantidade de aeronaves disponíveis. Levando em conta tais problemáticas, surge a possibilidade do uso de equipamentos menores buscando resultados satisfatórios, como por exemplo, aeromodelos ou multirotores, mais conhecidos como *drones*.

Desde uns poucos anos atrás o número de pesquisas em aplicações de veículos aéreos

não tripulados (VANTs) tem aumentado de forma significativa. As aplicações envolvem desde grandes aeronaves até pequenos quadrimotores, de centímetros de comprimento. Seu emprego abrange diversos campos, como o entretenimento, a área militar, o monitoramento de ambientes, além do uso por profissionais autônomos para aquisição de fotos e vídeos. Devido às grandes extensões das áreas agrícolas e ao baixo custo do emprego de pequenas aeronaves não tripuladas, pode-se observar um aumento do uso destes equipamentos no monitoramento de grandes plantações, auxiliando na prevenção e no combate de incêndios e pragas, dentre outras tarefas (TRIPICCHIO et al., 2015; POBKRUT; EAMSA-ARD; KERDCHAROEN, 2016; TRIPICCHIO et al., 2015).

Os multirotores utilizados em aplicações de monitoramento em geral, além de serem classificados como robôs aéreos, também podem ser classificados como robôs de serviço, uma vez que eles auxiliam as pessoas em ambientes de difícil acesso, como florestas (BERNI et al., 2009), plantações agrícolas (ZAINUDDIN et al., 2014; NONAMI et al., 2010), áreas em que seria inviável a utilização de aeronaves convencionais (ambientes internos como armazéns, por exemplo), isto é, pouco espaço de operação e acesso limitado ou mesmo devido ao alto custo de operação.

1.2 Contexto

Atualmente, o Laboratório de Automação Inteligente (LAI) do Departamento de Engenharia Elétrica da Universidade Federal do Espírito Santo está realizando vários estudos utilizando veículos aéreos não tripulados, tanto em ambientes internos (SANTOS et al., 2015) quanto externos (SÁ, 2014; SANTANA; BRANDÃO; SARCINELLI-FILHO, 2015; SANTANA; BRANDÃO; SARCINELLI-FILHO, 2015).

Em cooperação com o Corpo de Bombeiros Militar do Estado do Espírito Santo, o LAI está desenvolvendo pesquisas para realizar aquisições de imagens aéreas a serem utilizadas na construção de mosaicos georreferenciados, além de processamento de imagens a fim de detectar possíveis focos de incêndio. Entretanto, o uso de um sistema de montagem de imagens convencional geralmente demanda muito tempo para obter os resultados do mosaico ou exigem conexões com a Internet. Desta forma, serão experimentadas formas alternativas de geração do mosaico, buscando sobretudo a diminuição do tempo gasto em sua geração, seu georreferenciamento, possibilitando assim seu uso durante a ação das equipes do Corpo de Bombeiros.

Na literatura atual encontram-se alguns trabalhos voltados para o tema desta dissertação. Em (TARALLO et al., 2010) o autor utiliza imagens aéreas, obtidas com o uso de um avião de pequeno porte, para geração de mosaico. No trabalho é utilizado o algoritmo SURF (*Speeded Up Robust Features*) para obtenção dos pontos-chave e seus respectivos descritores. Para o casamento dos pontos leva-se em conta a menor distância euclidiana no

domínio das características, utilizando em seguida a técnica RANSAC (*RANdom SAmple Consensus*) para obtenção dos *inliers*. Depois disso estima-se a homografia e realiza-se a colagem em perspectiva das imagens.

Em Brown e Lowe (2007) há a geração de panoramas multilinhas com excelentes resultados. O método é robusto no que tange a ordem das imagens de entrada, orientação, iluminação, escala e remove imagens que não fazem parte do contexto. Além de extrair os pontos de interesses utilizando o SIFT (*Scale Invariant Feature Transform*), realizar o casamento dos pontos e utilizar o RANSAC para localizar os *inliers*, o autor faz uso de um método denominado *Multi-Band Bending*, obtendo resultados muito interessantes, porém o resultado não é georreferenciado.

Um mapeamento 3D em tempo real é realizado em Heng et al. (2011). Um algoritmo de odometria visual é executado a bordo do VANT e usa-se uma IMU (Unidade de Medida Inercial, do inglês, Inertial Measurement Unit) para melhorar o desempenho deste algoritmo. Isso melhora a estimativa de movimento entre um *frame* e outro de imagem capturada durante o movimento do veículo. A informação sensorial é usada para construir uma grade de ocupação 3D, a qual juntamente com imagens e dados inerciais, é usada pela estação em terra para montar um mapeamento 3D com texturas.

Trabalho semelhante é feito em Rosnell e Honkavaara (2012), no qual um VANT equipado com uma câmera e sensores de telemetria é programado para executar uma determinada rota e capturar imagens que apresentem sobreposição. Tais imagens mais os dados sensoriais de telemetria são usados para a construção de um mapa e obtenção de uma nuvem de pontos com informação topográfica.

Já no trabalho TARALLO et al. (2012) é apresentado uma metodologia semelhante a utilizada nesta dissertação, comparando algumas formas de obtenção de mosaicos. Para tal as imagens são capturadas por uma câmera acoplada a um avião com posterior processamento utilizando o algoritmo SURF e RANSAC.

Os trabalhos citados anteriormente ressaltam alguns pontos interessantes em se trabalhar com os VANTs em tarefas de mapeamento e monitoramento. Pelo fato de voarem, os robôs aéreos possuem um grande campo de visão, podendo capturar imagens com diferentes pontos de vista e alcances visuais, à medida que variam sua posição e altura. Isso permite inspecionar mais rapidamente grandes áreas e visualizar regiões que normalmente não são alcançadas por equipes de terra ou por unidades móveis terrestres.

Apesar dos trabalhos já citados, fazerem uso de imagens aéreas para processamento ou geração de mosaicos, não foi encontrada, na etapa de pesquisa do estado da arte feita durante este trabalho, uma interface única para configuração da missão de voo e geração do mosaico. Um outro ponto a ser considerado, é a possibilidade de geração do mosaico ainda em campo, mostrando-se na própria imagem gerada alguns pontos georreferenciados.

Estas funcionalidades foram abordadas e desenvolvidas nesta dissertação de mestrado.

1.3 Objetivos

O objetivo principal deste trabalho é o desenvolvimento de um software que viabilize a comunicação entre a estação de terra e o *drone*, um sistema de coleta de imagens e dados, com seus respectivos processamentos.

Pode-se enumerar como objetivos específicos pertinentes ao trabalho:

- Estudar trabalhos que abordam o tema em questão que possam contribuir na implementação deste trabalho.
- Desenvolver um sistema de comunicação entre a estação de terra e o *drone*.
- Implementar algoritmos para execução em um computador de bordo como o objetivo de armazenar dados de odometria e imagens.
- Desenvolver um sistema de colagem de imagens, com o intuito de criar o mosaico em campo.
- Georreferenciar o mosaico com dados de GPS obtidos no momento da captura das imagens aéreas.
- Comparar o método desenvolvido com aplicações comerciais.

1.4 Publicações

O trabalho apresentado nesta Dissertação já originou as seguintes publicações:

- Amorim, L.; Silva, L.; Queiroz, F.; Salvador, R.; Vassalo, R. and M. Sarcinelli-Filho. "Construção de Mosaico Utilizando Imagens Aéreas Adquiridas de Forma Autônoma." Anais Do XII Simpósio Brasileiro de Automação Inteligente - SBAI 2015, 987-992. Natal, Brasil, 2015.
- Amorim, L.; Santos, M.; Queiroz, F.; Silva, L.; Vassalo, R. and M. Sarcinelli-Filho. "Estimação de Posição e Atitude de Um VANT Baseada em GPS, IMU e Dados Visuais". Anais Do XII Simpósio Brasileiro de Automação Inteligente - SBAI 2015, 1660-1665. Natal, Brasil, 2015.

1.5 Organização da Dissertação

Este documento está organizado da seguinte forma.

O Capítulo 2 apresenta os equipamentos utilizados nos experimentos e o experimento realizado com o objetivo de validar os dados adquiridos através da estimação do posicionamento do *drone* com seus sensores, inclusive visão.

No Capítulo 3, são detalhados os softwares desenvolvidos para obtenção dos resultados desejados, bem como o protocolo utilizado na comunicação entre o *drone*, o computador de bordo e a estação de terra.

Já no Capítulo 4 será mostrado o método utilizado para geração do mosaico com seus algoritmos e escolha de parâmetros utilizados, exemplos de geração de mosaicos utilizando imagens adquiridas de forma distintas, além de algumas restrições encontradas para obtenção de resultados satisfatórios, com a execução do algoritmo.

Já no Capítulo 5 tem-se resultados mais concretos e refinados. Será possível notar que nos experimentos houve um maior número de aquisições de imagens. Para melhor julgamento do sistema, o software desenvolvido será comparado com outros métodos.

Por fim, no Capítulo 6, tem-se as conclusões dos resultados alcançados conforme os objetivos iniciais. Baseado nestes resultados serão apontados possíveis trabalhos para desenvolvimentos futuros.

2 Equipamentos Utilizados

Avanços recentes nas áreas das tecnologias, principalmente na computação, associadas ao uso cada vez mais comum de sistemas globais de navegação, tornam possível a utilização de uma diversidade de equipamentos que poderiam ser utilizados nos experimentos desta dissertação. Neste capítulo serão apresentados os equipamentos adotados no desenvolvimento deste trabalho, bem como os acessórios e configurações utilizadas.

2.1 *Drone*, sensores e equipamentos

A escolha do *drone* (Figura 1) deu-se pela disponibilidade do *drone* aprimorado por SÁ (2014) (Projeto Ômega), além da facilidade para realizar modificações necessárias com objetivo de suportar outros equipamentos durante o voo. O conjunto possuía uma autonomia inicial de aproximadamente doze minutos de voo, diminuindo para pouco menos de dez minutos após inclusão de mais alguns itens, como a RaspberryPi B+ (<http://www.raspberrypi.org>) e um módulo de câmera proprietário (<https://www.raspberrypi.org/products/camera-module/>).

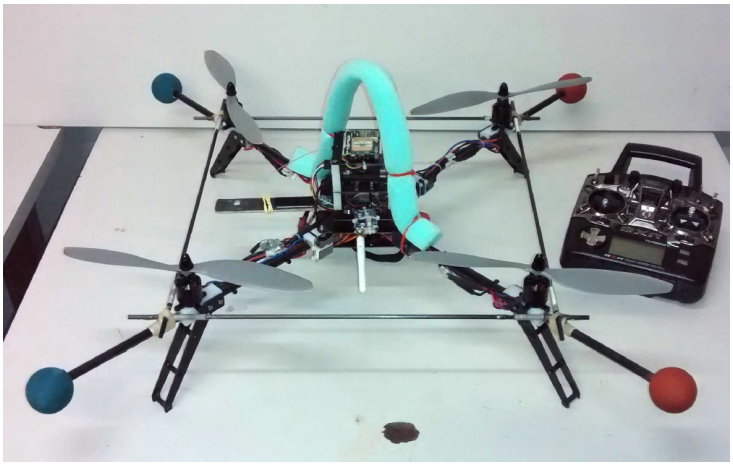
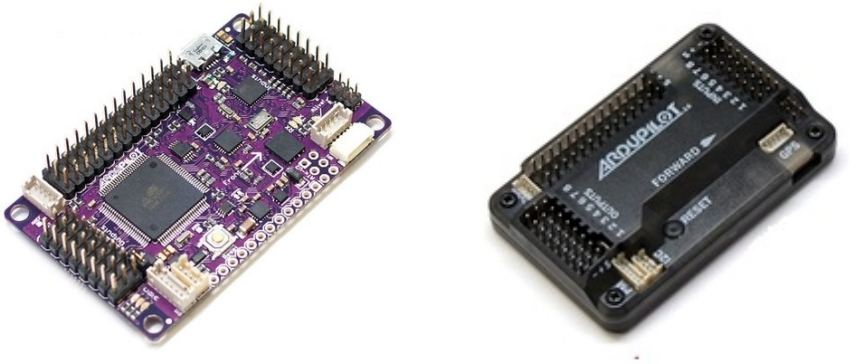


Figura 1 – *Drone* utilizado no início dos experimentos.

2.1.1 Especificações do *Drone*

O drone do Projeto Omega possui como controlador o Ardupilot ATMega 2560 (APM 2.6) muito utilizado para este fim, principalmente em projetos científicos por se tratar de código aberto. Na Figura 2(a) tem-se o Ardupilot ATMega e em 2(b) o Ardupilot ATMega em seu *case*. Com este controlador, é necessário focar apenas no controle de alto nível e a estratégia escolhida foi a geração de uma grade de *waypoints* fazendo com que o

drone navegasse por cada um destes pontos obtendo as fotografias. Para tal, o controlador (APM) é conectado a um sensor GPS da 3DR conforme pode-se visualizar na Figura 3.



(a) (b)

Figura 2 – APM 2.6 2(a) e APM + case 2(b).

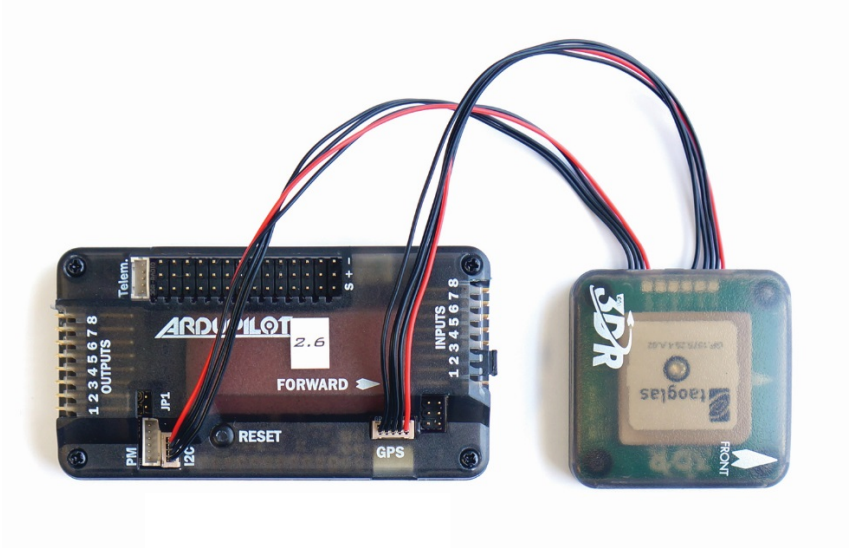


Figura 3 – GPS da 3DR.

Como atuadores, o *drone* utiliza quatro motores DC *Brushless* 880Kv (880 rpm por volt). Um dos motores está mostrado na Figura 4(a), onde cada um aciona uma hélice APC 11X47 (onze polegadas de comprimento e passo de 4,7 polegadas) (Figura 4(b)). Este tipo de motor, por possuir ímãs permanentes fixos, dispensa a utilização de escovas, o que melhora sua eficiência energética e ruídos de utilização, quando comparados a outros modelos de motores DC.

O acionamento e o controle de cada motor são realizado por quatro ESCs (*Electronic Speed Control*) de 20A, sendo um para cada motor (Figura 5). A escolha da corrente máxima do ESC está diretamente relacionada com a corrente máxima exigida por cada motor, que no caso atual é de 16A. Este tipo de equipamento recebe os sinais PPM (*Pulse*

(a) (b)

Figura 4 – Motor 4(a) e hélice 4(b) utilizada no VANT.

Position Modulation) do controlador, controlando assim a tensão aplicada a cada atuador, que por sua vez, desenvolve a respectiva velocidade angular.

Figura 5 – ESC de 20A.

A alimentação do *drone* é realizada por uma bateria de LIPO (*Lithium-Polymer*) de três células (11.1V de tensão nominal), capacidade nominal de 5.200mAh, podendo realizar uma descarga trinta vezes maior que a sua corrente nominal (Figura 6), suprindo com facilidade a necessidade energética do controlador, instrumentos, sensores e atuadores. Os equipamentos que necessitam de alimentação de 5V fazem uso de um regulador de tensão.

Figura 6 – Bateria de três células com capacidade 5200mAh e 30C.

2.1.2 Rádio Controle

Com o objetivo de auxiliar no controle do *drone* nos pousos, decolagens, enviar comandos durante o voo e também controlá-lo de forma manual durante eventuais falhas, fez-se uso de um rádio transmissor (tipo aeromodelismo) Turngy 9XR de nove canais (Figura 7). Este rádio tem a função básica de misturar os sinais dos canais de comando e enviá-los ao módulo transmissor. Para esta função foi utilizado o módulo transmissor da FrSky DJT (Figura 8(a)).

Figura 7 – Rádio Turngy 9XR de nove canais.

Os sinais enviados pelo rádio transmissor são recebidos no VANT pelo módulo receptor, DR4-II também da FrSky (Figura 8(b)), que subsequentemente os encaminha para o controlador APM para interpretação e execução dos comandos.

(a) (b)

Figura 8 – Módulos FRSKY de transmissão 8(a) e recepção 8(b).

2.1.3 Link de rádio para telemetria

Além do rádio de oito canais utilizado para envio de comandos ao *drone* também é utilizado um link de rádio, com operação na frequência de 915 MHz da 3DR (Figura 9) com o objetivo de realizar a comunicação entre a controladora e a estação de terra ou seu computador de bordo, dependendo da necessidade. Esta comunicação é feita sobre protocolo Mavlink (*Micro Air Vehicle Communication Protocol*) amplamente utilizado neste tipo de aplicação.

Figura 9 – Link de telemetria 915 MHz.

2.1.4 Computador de bordo

Para servir de computador de bordo, com as tarefas de supervisionar o deslocamento do *drone*, armazenar os dados de voo (ângulos de orientação, coordenadas de GPS e altitude) além de acionar a câmera e armazenar as imagens capturadas, foi acoplado ao mesmo uma placa de processamento RaspberryPi B+ (<http://www.raspberrypi.org>) e um módulo de câmera proprietário (<https://www.raspberrypi.org/products/camera-module/>), conforme observa-se na Figura 10.

2.1.5 *Gimbal* e câmera

Com o objetivo de manter o eixo óptico da câmera perpendicular ao plano do terreno mapeado é utilizado um estabilizador de câmera (*gimbal*) conforme se observa na Figura 11. Para tal, fez-se necessária a modificação do *drone* para uma segunda versão, possibilitando a decolagem e o pouso com a câmera acoplada à *gimbal* (Figura 11) sem que a câmera tocasse o solo.

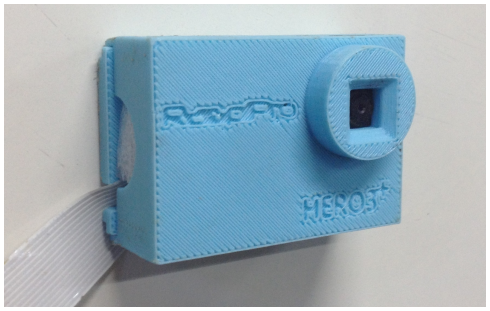
Na Figura 12(a) pode-se observar que a câmera (Raspcam) foi acomodada dentro dentro de um *case* para seu correto acoplamento na *gimbal* (Figura 12(b)). Este artifício, além de ajudar no balanceamento do braço do estabilizador, uma vez que no

Figura 10 – RaspberryPi B+ e seu módulo de câmera.

Figura 11 – Segunda vesão do *drone* utilizado.

seu funcionamento leva em conta o peso de uma câmera, auxilia sua correta fixação e estabilização.

A Raspcam, com apenas 3g de peso, possibilita a aquisição de imagens com a resolução de até 5 Megapixels, podendo ser configurada a critério de sua aplicação.



(a)

(b)

Figura 12 – Capa para acoplar a câmera à *gimbal* 12(a) e o estabilizador de câmera 12(b).

2.2 Validação do protótipo e dados coletados

Na tentativa de obter o melhor caminho no desenvolvimento deste trabalho, foram realizados inicialmente alguns experimentos com o objetivo de validar, além da plataforma aérea (*drone*), os dados colhidos através da mesma, principalmente quanto ao posicionamento global fornecido pelo GPS. Para tal, fez-se a captura dos dados de localização do GPS e de imagens aéreas focando em um padrão (Figura 13). Para isto, realizou-se um voo manual do *drone*, onde o operador o posicionava sobre um local desejado, acionando a câmera por uma chave no controle remoto. A cada fotografia obtida, o operador alterava a posição e orientação do veículo e capturava uma outra imagem.

De posse das imagens, para obter o posicionamento do *drone* através de técnicas de visão computacional, toma-se como referência o padrão da Figura 13, e calcula-se a posição da câmera em relação a este (parâmetros extrínsecos da câmera). Considerando que a posição da câmera no mundo coincide com a posição do *drone* e de posse desta posição no momento da captura das fotos, pode-se então comparar estes dados com os dados de GPS obtidos junto à sua controladora, a fim de validar e aferir os sensores de bordo do *drone*. Além disso, pode-se aplicar a metodologia de fusão apresentada em (AMORIM et al., 2015) a qual faz uso de Filtro de Kalman aprimorando assim a estimativa de posição e orientação do *drone*. Os sistemas de aquisição e armazenamento dos dados durante os voos serão explicados em momento oportuno nesta dissertação.

Os dados provenientes do módulo GPS estão em coordenadas geográficas (latitude e longitude). Já os dados obtidos por visão computacional estão mensurados em metros, tendo a origem do padrão visual utilizado como referência. Para realizar a comparação entre estes, e os dados do GPS, os dados em coordenadas globais foram convertidos para a referência local com a origem e a orientação, também, do padrão visual. Tal transformação do sistema de referências global para o sistema local (em metros) também considera a curvatura da Terra. Em especial, na conversão de longitude, foi utilizado como referência o raio da Terra na latitude 20 16′18.2”S, devido à proximidade com a localização geográfica dos experimentos.

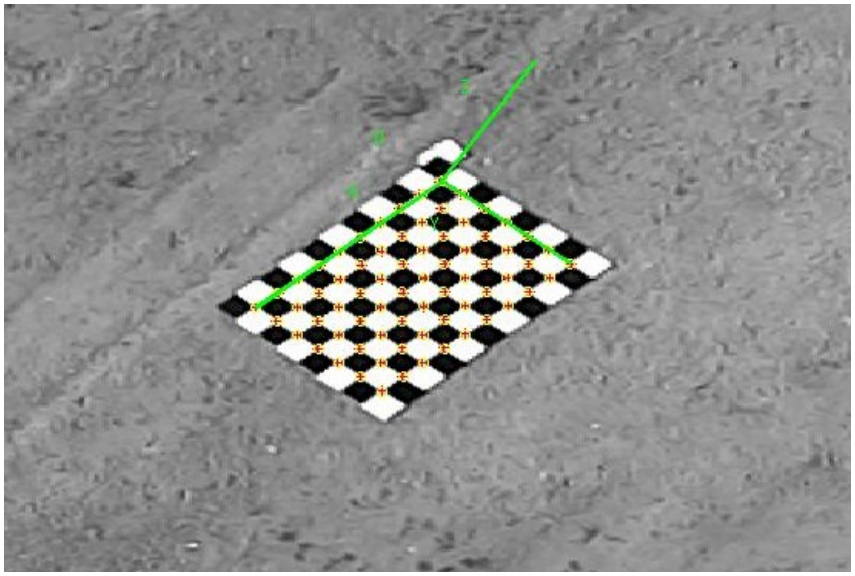


Figura 13 – Padrão de quadrados de 20 *cm* *x* 20 *cm* utilizado nos experimentos externos.

Para extrair informações métricas utilizando uma câmera, no que diz respeito ao posicionamento do *drone*, é fundamental realizar sua calibração (câmera). Na literatura, existem duas categorias de método de calibração. Uma delas é conhecida como *self-calibration*, que estima os parâmetros intrínsecos e extrínsecos (posição da câmera) a partir de imagens do ambiente, com pouco ou nenhum conhecimento sobre a cena; e outra, que é mais simples e amplamente utilizada, emprega imagens de objetos com dimensões conhecidas (MA et al., 2004).

O método de Zhang (2000) é amplamente utilizado, onde a partir de uma sequência de imagens de um padrão planar com dimensões conhecidas, é possível estimar os parâmetros intrínsecos. Com estes pode-se calcular os parâmetros extrínsecos, que representam uma transformação de corpo rígido composta por uma rotação e uma translação, utilizada para trocar o referencial do sistema de coordenadas do padrão para o referencial da câmera.

No caso em questão, com uma sequência de imagens capturadas, utilizando um padrão quadriculado de dimensões 1.60 *m* x 2.20 *m* com cada quadrado medindo 20 *cm*, realizou-se a calibração utilizando uma *toolbox* do MATLAB® (*Camera Calibration Toolbox for Matlab*). Ao fim do processo (que utiliza o método Zhang (2000)) foi possível obter, no referencial do padrão de calibração, as coordenadas 3-D do VANT, além dos ângulos de rotação em torno dos três eixos.

Na Figura 14 pode-se visualizar os resultados obtidos pelo processamento de imagens e o resultado obtido pelo GPS. Observa-se um erro grande entre a câmera e o GPS nos eixos x e y, explicado pela baixa resolução na qual foi feita a aquisição destes dados (GPS) no momento do experimento. Porém, como é observado, nos demais gráficos tem-se uma certa coincidência dos demais dados de altura (eixo z) e orientação, salientando o quão confiável é o uso do GPS para a aplicação em que os dados serão utilizados.

Ainda na Figura 14 tem-se os dois dados fundidos no Filtro de Kalman Descentralizado (DKF) dando uma segunda opção para melhorar a estimativa do posicionamento do *drone*, o que não será abordado neste trabalho. Demais dados, metodologia, entre outros, podem ser encontrados em (AMORIM et al., 2015).

Observa-se ainda na Figura 14, que o resultado da fusão dos dados GPS para posicionamento, com os resultados provenientes do processamento das imagens adquiridas com a câmera acoplada ao veículo aéreo e com os sensores inerciais do *drone*, é possível obter um resultado mais coerente e com erros menores do que 10m (obtidos com baixa resolução nos eixos x e y). Adicionalmente, estes resultados demonstram a viabilização da plataforma na aquisição dos dados dos sensores internos do *drone* (GPS e sensores inerciais, por exemplo) e imagens através de comandos enviados pelo operador em terra, através do rádio controle. Futuramente, essa fusão poderá ser feita em tempo real para melhorar a estimativa da postura do veículo aéreo, visto que a fusão dos dados apresentados da fusão na Figura 14 apresenta uma melhora significativa no posicionamento do veículo, mesmo utilizando uma baixa resolução na aquisição dos dados do GPS.

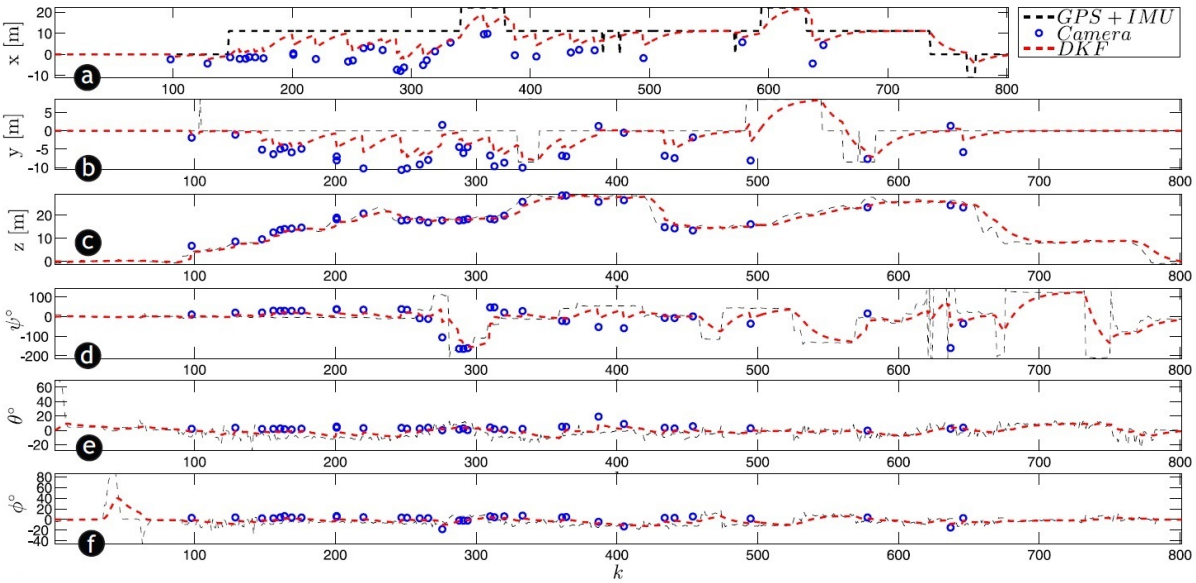


Figura 14 – Os resultados da estimação do $[\xi\ \eta]$ pelo DKF, onde $\xi = [x\ y\ z] \in \Re^3$ representa os deslocamentos longitudinal (x), lateral (y) e normal (z), e $\eta = [\phi\ \theta\ \psi] \in \Re^3$ indica os ângulos de *roll* (ϕ), *pitch* (θ) e *yaw* (ψ).

3 Softwares Desenvolvidos

Neste capítulo serão descritos os softwares criados para automatizar algumas funcionalidades, além de facilitar a comunicação entre a estação de terra e o *drone*, e deste com seu computador de bordo durante os voos. Será descrita também a metodologia utilizada para obtenção de um mosaico, montado com as imagens aéreas capturadas pelo *drone* durante seu deslocamento.

3.1 Protocolo Mavlink

Para o melhor entendimento deste capítulo, será abordado as principais características do protocolo Mavlink (*Micro Air Vehicle Communication Protocol*) (<http://qgroundcontrol.org/mavlink/start/>), base da comunicação entre *drone* - estação de terra e entre *drone* - computador de bordo. Este protocolo foi lançado no início de 2009 por Lorenz Meier sob licença LGPL (*Library General Public License*), baseia-se em um cabeçalho de mensagem muito leve, amplamente utilizado em mini VANTS devido à sua versatilidade, tendo sua estrutura de pacotes das mensagens inspirada no protocolo CAN (*Controller Area Network*).

O Mavlink foi implementado sobre dois pilares: Segurança e Velocidade. Sua arquitetura permite verificar a correta entrega das mensagens, e ainda identificar se houve mensagem perdida, com uma sobrecarga de apenas seis *bytes* para cada pacote. A transmissão pode ser realizada a uma taxa de 9600 a 115200 *baud*, dependendo do *hardware* utilizado, com uma carga útil de 42 a 224 *bytes* respectivamente. Na Figura 15 pode-se observar a estrutura das mensagens.

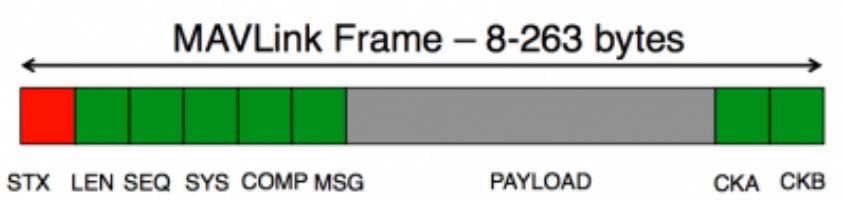


Figura 15 – Estrutura das mensagens do Protocolo Mavlink.

Na Tabela 1 pode ser verificada a descrição da cada *byte* utilizado no protocolo. Verifica-se ainda uma diferença no posicionamento dos *bytes*, de acordo com o total de carga útil transportado na mensagem. Como pode ser visto, uma mensagem pode ter de 8 a 263 *bytes*, no caso do uso de sua capacidade total de carga. Os dois últimos *bytes*, além de identificar qualquer erro na mensagem, pode informar se a mensagem está sendo decodificada em versão do protocolo diferente da versão de sua codificação.

Tabela 1 – *Bytes* do Mavlink

Byte Index	**Conteúdo**	**Valor**	**Descrição**
0	Sinal início do Pacote	0xFE	Sinaliza o início de um pacote.
1	Tamanho dos dados	0 - 255	Indica o tamanho do dado útil enviado na mensagem.
2	Número de sequência do pacote	0 - 255	Indica o número de sequência do pacote atual, para poder detectar perda de pacotes.
3	ID do sistema	1 - 255	Identifica o sistema que está enviando, facilitando a diferenciação entre veículos no caso de mais de um.
4	ID do Componente	0 - 255	Identifica o componente do veículo que está enviando a mensagem (IMU, GPS, entre outros).
5	ID da Mensagem	0 - 255	Identifica o tipo de mensagem enviada, possibilitando sua correta decodificação.
6 até (n+6)	Dados	(0 - 255) bytes	Conteúdo da mensagem (carga útil).
(n+7) até (n+8)	*Checksum* (*low byte*, *high byte*)	-	Dois *byte*s utilizados na verificação ITU X.25/SAE AS-4 *hash*. No calculo dos *bytes* leva-se em conta a versão do protocolo utilizado, evitando uma decodificação em versão diferente da utilizada na codificação.

Para cada tipo de mensagem (ID) tem-se um codificador e um decodificador, que devem ser utilizados antes da transmissão e após a recepção de cada mensagem respectivamente, assim tem-se os dados de cada mensagem para aplicação onde for conveniente. Abaixo serão descritas as mensagens utilizadas nos *softwares* implementados para os experimentos desta dissertação.

- MISSION_REQUEST_LIST: Esta mensagem possui o ID 43 e deve ser enviada quando se deseja ler os *waypoints* gravados na controladora do *drone*. Após seu envio é necessário aguardar uma mensagem de retorno MISSION_COUNT. Ao receber esta mensagem, deverá ser solicitado o envio dos *waypoints*, um por vez.

- MISSION_COUNT: Esta mensagem possui o ID 44 e é a resposta da mensagem MISSION_REQUEST_LIST. Após sua decodificação pode-se ler a quantidade de *waypoints* gravados na controladora do *drone*. Esta mensagem também é enviada ao *drone* quando se deseja informar o envio de uma certa quantidade de *waypoints*.

- MISSION_REQUEST_INT: Com o ID 51, esta mensagem é enviada para solicitar um item específico da missão (*waypoint*). Após o envio desta mensagem, deve-se esperar uma mensagem do tipo MISSION_ITEM_INT, em seguida solicitar o item seguinte da missão. Pode ser enviada tanto pela estação de terra quanto pelo *drone*.

- MISSION_ACK: Possui o ID 47, e deve ser enviada ao finalizar o recebimento de todos os *waypoints*. Esta mensagem é envida pela estação de terra quando esta recebe o último *waypoint*, depois da solicitação da lista completa de pontos. É enviada também pelo *drone* (controladora) quando este está recebendo uma missão (conjunto de *waypoints*) e recebe o último ponto.

- MISSION_ITEM: Esta mensagem possui o ID 39 e trata-se de um *waypoint*, contendo todas as informações de um ponto 3D na atmosfera terrestre (ou em relação a um referencial), a orientação do veículo ao atingir o ponto, além do tempo em que permanecerá no ponto e uma *flag* indicando se o *drone* vai continuar automaticamente para o ponto seguinte.

- MISSION_CLEAR_ALL: Com o ID 45, esta mensagem deve ser enviada ao *drone* quando se deseja apagar toda a missão armazenada na controladora.

- MISSION_SET_CURRENT: Possui ID 41, e deve ser enviada quando se deseja modificar o *waypoint* corrente (destino atual do veículo).

- COMMAND_INT: Possui ID 75, e é utilizada quando há necessidade de enviar comandos para o *drone*. Nos experimentos foi utilizada para enviar o comando MISSION_START, enviado ao veículo para que seja iniciada a missão. Como parâmetro, deve-se enviar o *waypoint* inicial e o final da missão.

- SYS_STATUS: Possui o ID 1, e é utilizada para monitorar o *status* do *drone*. Nos experimentos esta mensagem foi utilizada para monitorar a tensão atual da bateria do *drone*, mas entre outros, pode-se monitorar também a corrente consumida e a estimativa de tempo restante de uso da bateria.

- HEARTBEAT: Possui o ID 0, sendo utilizada também para monitorar o *status* do *drone* no que diz respeito ao seu modo de voo (guiado armado, manual armado, automático armado entre outros) e ao estado da controladora (inicializando, ativo, crítico entre outros) podendo assim realizar, se necessária uma ação correspondente a cada situação.

- GPS_RAW: Com o ID 24 esta mensagem é utilizada nos experimentos para monitorar a quantidades de satélites conectados ao GPS do *drone*, podendo garantir, se necessário, uma maior precisão durante os experimentos.

- GLOBAL_POSITION: Possui o ID 33 e contém as coordenadas geográficas do *drone*, além de sua altura. Esta mensagem já contém os dados de GPS fundidos com o acelerômetro do veículo.
- ATTITUDE: Com o ID 30, esta mensagem contém os ângulos *roll*, *pitch* e *yaw* do *drone*, além de suas respectivas velocidades angulares.
- MISSION_CURRENT: Com o ID 42, esta mensagem é utilizada para monitorar o item da missão corrente, isto é, o *waypoint* para onde o *drone* está seguindo no momento do envio da mensagem.
- RC_CHANNELS_RAW: Possui ID 35, mensagem utilizada para monitorar o sinal de cada canal recebido no receptor do rádio controle. Com isto pode-se enviar um comando pelo rádio (tipo aeromodelismo) e de acordo com este comando, pode-se tomar uma decisão no computador de bordo do *drone*.
- NAV_CONTROLLER_OUTPUT: Com o ID 62, esta mensagem foi usada nos experimentos para monitorar a distância atual entre o *drone* e o *waypoint* corrente.

3.2 Software de bordo

Com o objetivo de monitorar *online* os voos durante os experimentos, armazenar as imagens capturadas, os dados relativos ao posicionamento e à postura do *drone*, foi acoplado ao mesmo uma placa de processamento RaspberryPi B+ juntamente com um módulo de câmera (Raspcam). A escolha deste conjunto deu-se pela facilidade de acoplar a câmera ao computador, realizando seu acionamento no momento desejado. A placa executa uma versão de sistema operacional Linux (Debian PI) customizado pelo fabricante.

A comunicação, durante o voo, entre a RaspberryPi B+ e o *drone* é feita através de um *link* de rádio de 915 MHz (2.1.3). Optou-se por este *link*, pois o uso da conexão direta via porta USB além de impossibilitar o uso do *link* acima mencionado, na comunicação com a estação de terra quando necessária, força a alimentação da RaspberryPI B+ por esta ligação (porta USB) sobrecarregando a controladora (Ardupilot). Com isto, foi necessário inserir um novo regulador de tensão para suprir as necessidades energéticas da RaspberryPI B+, através da bateria principal do *drone*.

O *software* implementando para execução nesta placa foi desenvolvido em C++, utilizando o protocolo Mavlink na comunicação. O *software* possui como base as seguintes funcionalidades: comunicação serial com o *drone*, armazenamento dos dados coletados relativos à postura do *drone*, controle de alto nível e captura de imagem com seu devido armazenamento.

3.2.1 Rotina de comunicação com o *drone*

Esta rotina é a principal executada no computador de bordo do *drone*. Devido a limitações de *software*, a comunicação é feita a um *baud rate* de 57600, com oito *bits* de dados e sem *bit* de paridade. O *loop* principal dessa rotina aciona, quando necessário, as demais rotinas tomando como parâmetros para isto os dados obtidos na comunicação com o *drone*. No momento em que o *drone* (por consequente a RaspberryPi B+) é ligado, essa rotina é executada, iniciando o monitoramento das mensagens enviadas pela Ardupilot através do protocolo Mavlink. A rotina principal aguarda até que o sistema de GPS do *drone* sincronize com o número mínimo de satélites (três) para depois, dar continuidade à sua execução.

De posse das mensagens, o sistema analisa a RC_CHANNELS_RAW obtendo o valor do canal 7 (em PPM) recebido na controladora do *drone*, através do módulo receptor DR4-II. Este canal representa o estado da chave GEA SW do rádio transmissor (Figura 7). Caso a chave esteja acionada (valor do PPM maior que 1500) a missão é considerada como automática, isto é, o *drone* navegará de *waypoint* a *waypoint* fazendo a aquisição das imagens aéreas de forma automática, sem intervenção humana. Caso a chave não esteja acionada, o veículo ao sobrevoar um dado ponto, aguardará o acionamento da chave GEA SW (canal 8) e só após este acionamento, realizará a aquisição da imagem e prosseguirá para o próximo *waypoint*. Este processo se repetirá até o último *waypoint*, que coincide com o ponto inicial da missão. Todo o processo acima pode ser observado nos diagramas apresentados na Figura 16.

3.2.2 Rotina de armazenamento de dados

A partir do momento em que a controladora do *drone* informa ao computador de bordo, através da rotina de comunicação, que o GPS está sincronizado com o mínimo necessário de satélites, a rotina de armazenamento de dados começa a armazenar os dados de posicionamento global do veículo (latitude, longitude, altura) além da postura do mesmo (*roll*, *pitch* e *yaw*), a uma taxa de 2 Hz. Estes dados são armazenados em um arquivo de texto para posterior processamento juntamente com as imagens capturadas.

3.2.3 Rotina de controle de alto nível

Nas missões automáticas, esta rotina monitora a distância atual até o próximo *waypoint* através da mensagem NAV_CONTROLLER_OUTPUT. No momento em que a distância atinge um valor menor que 3 metros (constante estimada de forma empírica) a rotina aciona a câmera, capturando a imagem aérea, armazenando-a no cartão de memória da RaspberryPi B+.

Já nas missões manuais o operador, ao visualizar que o *drone* estacionou sobre

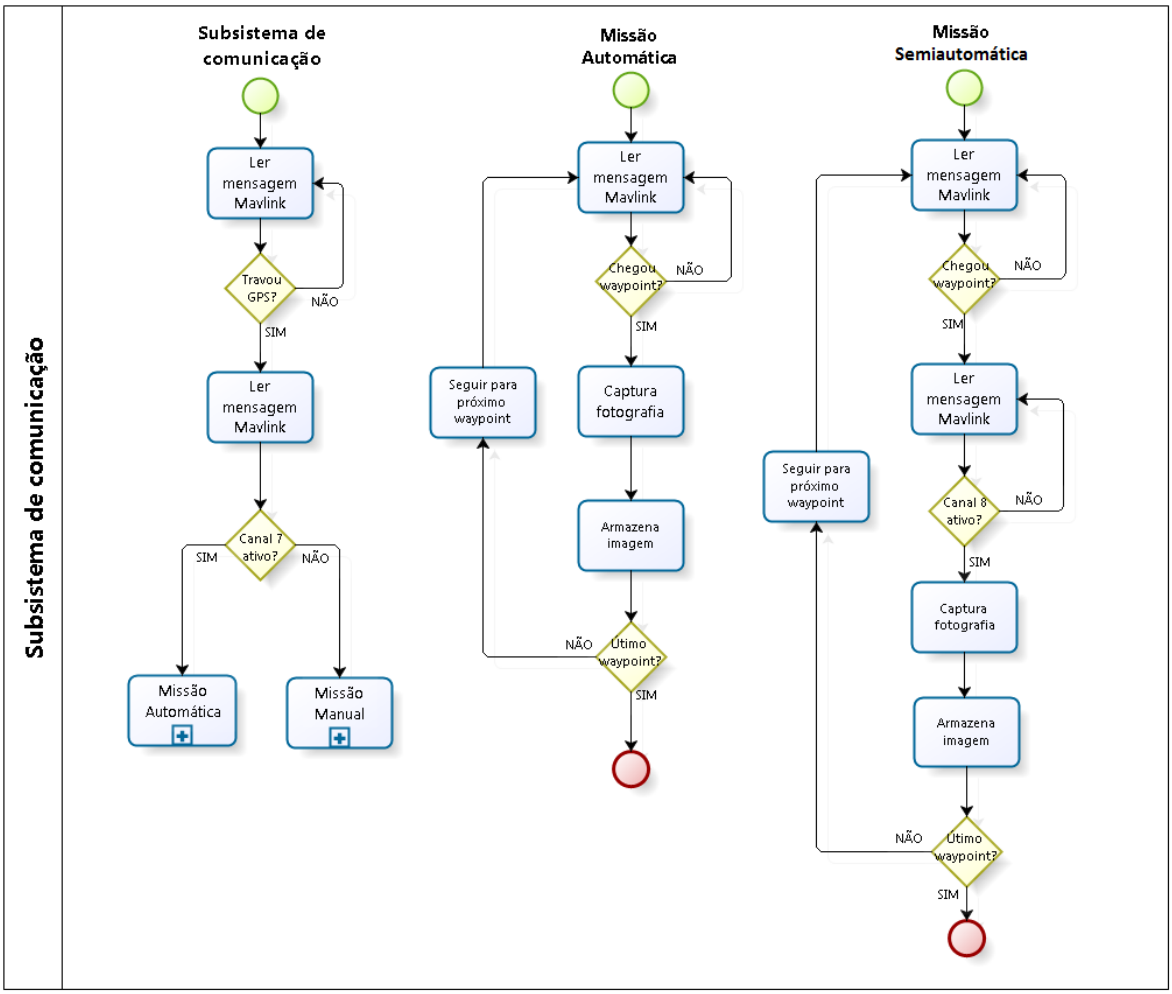


Figura 16 – Diagrama do Subsistema de comunicação.

um dado *waypoint*, aciona a chave TRN SW (canal 8) o que causa a mudança do *status* do canal na mensagem NAV_CONTROLLER_OUTPUT. Neste momento, a rotina de controle realiza a aquisição da imagem e a armazena de forma semelhante à da missão automática, porém após este armazenamento, é enviada uma mensagem (comando) à Ardupilot mandando-a seguir para o próximo *waypoint*.

Nas duas formas de missão, no momento da captura da imagem, é sinalizada uma *flag* informando à rotina de armazenamento de dados que neste momento foi capturada uma imagem. Assim, aquela rotina ao armazenar o dado subsequente, também o sinaliza, tornando possível o sincronismo da imagem com as coordenadas atuais, em momento posterior.

3.3 Estação de terra

Foi criado um *software* com interface gráfica para ser executado em um *notebook*, a fim de realizar as configurações do *drone* para realização das missões, além de realizar

o processamento das imagens obtidas e assim obter o mosaico (Capítulo 4) em campo. No que diz respeito às configurações para realização da missão, este *software* realiza a comunicação com o *drone* através do protocolo Mavlink utilizando para isto o *link* de telemetria.

Esta comunicação em alto nível (utilizando o protocolo Mavlink) é realizada de forma assíncrona, com o uso de *threads*, de modo que seja possível enviar um comando e aguardar por respostas sem a necessidade de pausar o *loop* principal, ou utilizar um outro artifício que não condiz com as boas práticas de programação.

Para um perfeito funcionamento, ao iniciar o programa, três *threads* são iniciadas, sendo: uma para tratar a tela do programa e interceptar os acionamentos dos botões, executando suas devidas ações; outra para ler o *link* de comunicação, decodificar as mensagens recebidas e atualizar as demais *threads* quando necessário; e a terceira *thread* é utilizada para enviar os comandos ao *drone*, uma vez que, a maioria destes comandos, necessita do envio de várias mensagens intercaladas com confirmações de recebimento.

Para realizar toda a programação foi utilizado o Qt Framework (<http://doc.qt.io/qt-5>). Este *framework* além de fornecer um ambiente de programação de alta qualidade, disponibiliza uma série de classes para facilitar o desenvolvimento de *software*. Uma delas é a QMutex, utilizada neste projeto para sincronizar as *threads* evitando concorrências durante os acessos a determinados atributos de classe. A comunicação entre as *threads* é realizada através de *signals* e *slots*. Os *signals* são avisos emitidos por uma *thread* em determinado momento do código, já os *slots*, são métodos de uma determinada *thread* acionada no momento em que um determinado *signal* é emitido. No momento em que as diversas *threads* são criadas, deve-se realizar a ligação entre os *signals* e os *slots*, pois só assim o programa saberá qual *slot* deve ser acionado na emissão de um determinado *signal*.

Para configuração da missão, o operador posiciona o *drone* voltado para a área onde será realizado o mapeamento, e só então, deverá ligá-lo. Neste momento, após iniciar o *software* da estação de terra, o operador deverá conectá-lo ao *drone*. Assim que o veículo travar o GPS (sincronizar com o mínimo de três satélites), a estação de terra marcará suas coordenadas (latitude e longitude), altura em relação ao nível do mar e sua postura (*roll*, *pitch* e *yaw*). Neste momento o operador deverá informar a que altura o voo será realizado, além das dimensões da grade de *waypoints* a ser criada (quantidade de linhas e colunas), conforme visualizado na Figura 17.

De posse destes dados o operador aciona o comando de gerar *waypoints*. A estação gerará as coordenadas de cada ponto já na sequência do voo e automaticamente os envia ao *drone*. Cada *waypoint* contém a sua latitude, longitude, altura, orientação que deve ter ao atingir cada ponto, tempo que permanecerá em cada ponto e se o veículo seguirá para o ponto seguinte automaticamente. Para o cálculo das coordenadas geográficas onde

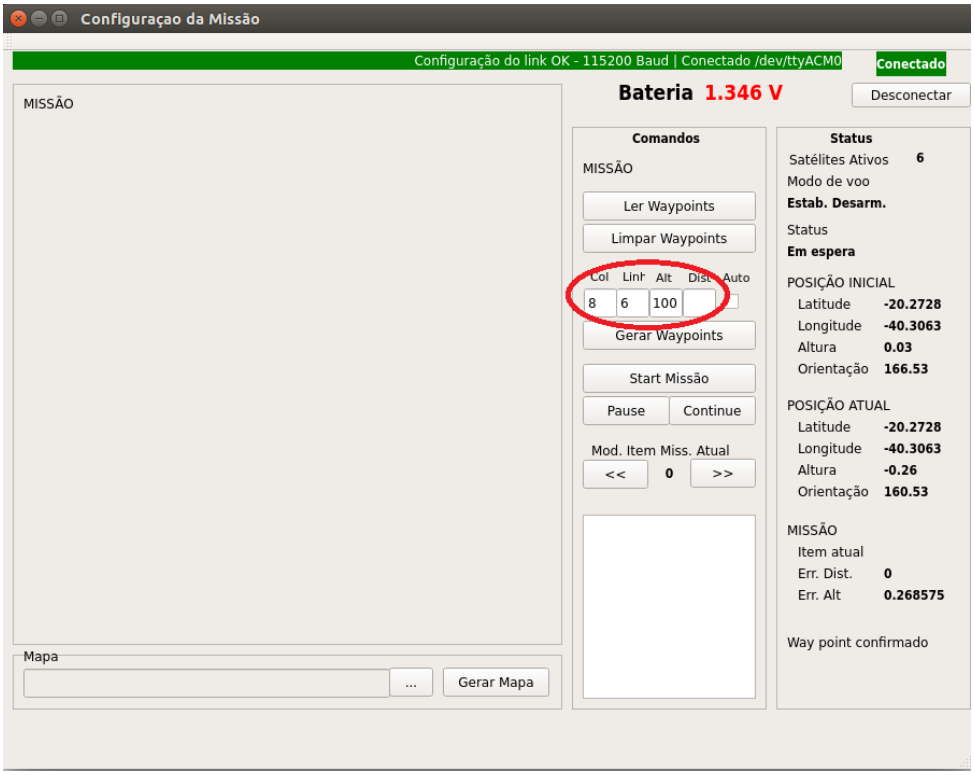


Figura 17 – Software da Estação de Terra com os parâmetros para missão.

as imagens deverão ser capturadas, são levadas em conta a abertura angular da câmera ($41.41^0 + / - 0.11^0$ em X e $53.50^0 + / - 0.13^0$ em Y) e a altura de voo. Assim obtêm-se o mínimo de 60% de sobreposição das imagens nos dois eixos. Ao final do envio dos pontos ao *drone* é solicitado ao mesmo que envie os *waypoints* armazenados em sua memória para comparação com os pontos enviados. Tal procedimento garante que a missão a ser executada corresponde corretamente àquela que foi enviada (Figura 18).

Na Figura 19 temos um mapa com uma sobreposição de uma grade de *waypoints*, contendo 5 linhas e 4 colunas. Observa-se que a missão do *drone* é um percurso fechado, isto é, após sobrevoar o último ponto, o veículo retorna ao local de saída, facilitando a execução do pouso, por parte do operador.

Na Figura 20 temos o protocolo Mavlink sendo utilizado para enviar uma missão de 50 *waypoints* para a controladora (Ardupilot). A seguir será descrito este procedimento.

Como primeiro passo, deve-se informar à controladora a quantidade de pontos a serem enviados e aguardar por uma resposta. A controladora, em resposta, envia a mensagem WAYPOINT_REQUEST(WP 0) informando que está aguardando *waypoint* 0 (passo 2). A seguir (passo 3), envia-se o WAYPOINT(WP 0) e aguarda novamente. Assim que a controladora receber esta mensagem, ela solicitará os demais pontos de forma sequencial (passo 4), até receber o último *waypoint* quando ela finalizará com uma mensagem do tipo WAYPOINT_ACK.

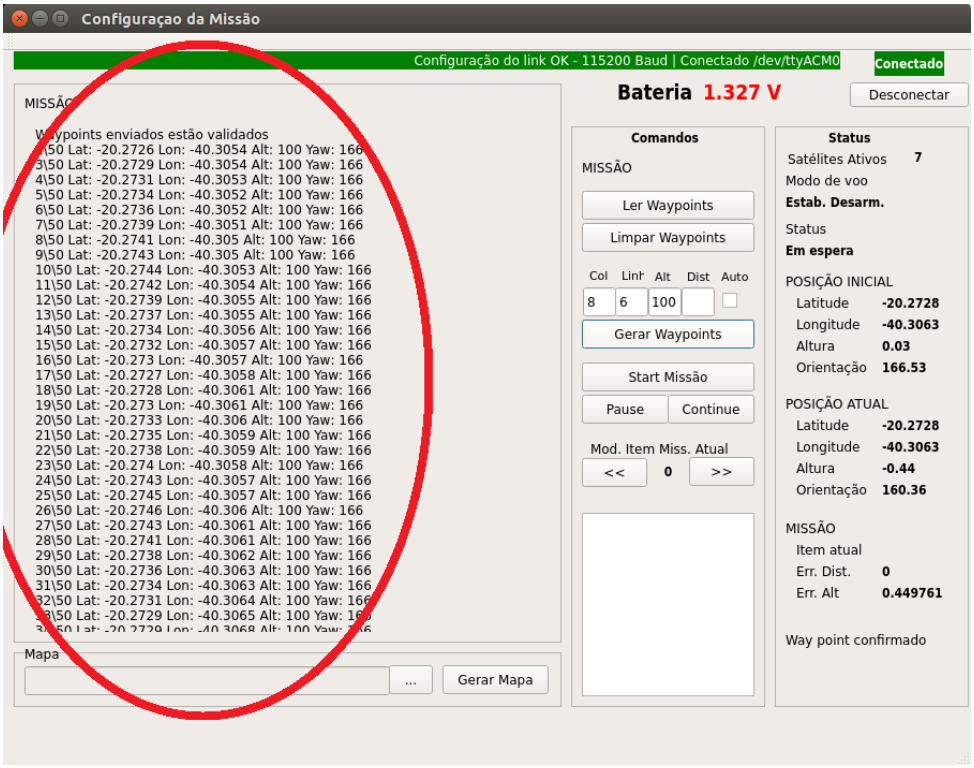


Figura 18 – Software da Estação de Terra com os *waypoints* enviados.

Figura 19 – *Waypoints* gerados pela estação de terra.

De forma bem semelhante, na Figura 21 utilizou-se protocolo Mavlink para ler todos os *waypoints* armazenados na missão atual da controladora. No passo 1 informa-se a intenção de ler os pontos com o envio da mensagem WAYPOINT_REQUEST_LIST aguarda-se a resposta da controladora. No passo 2, a controladora informa a quantidade

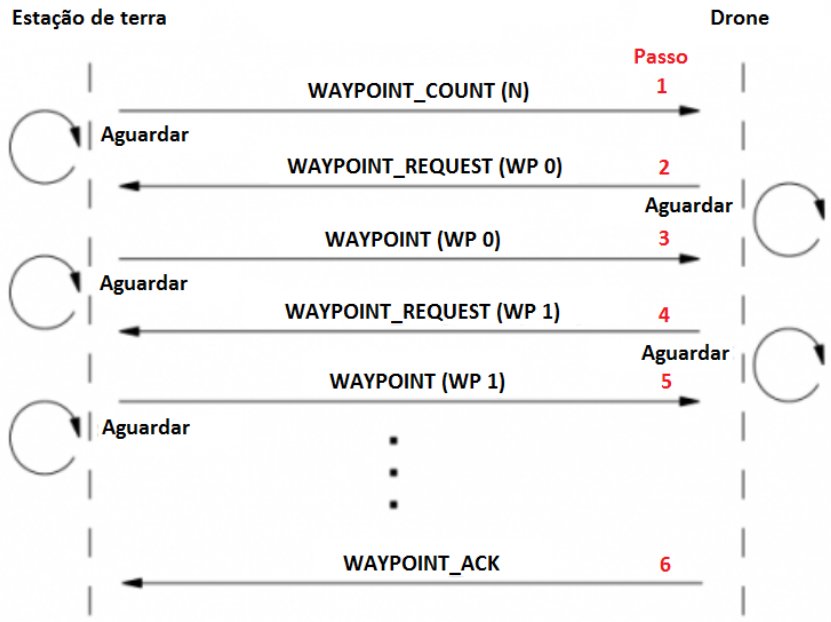


Figura 20 – Protocolo Mavlink utilizado para escrita de *waypoints*.

de pontos da missão, através da mensagem WAYPOINT_COUNT(N). A partir deste momento solicita-se à controladora *waypoint* a *waypoint* até atingir o enésimo ponto, através da mensagem WAYPOINT_REQUEST(WP i) (passo 3), recebendo a cada requisição os respectivos pontos através da mensagem WAYPOINT(WP i), como pode-se visualizar no passo 4. Após receber o último ponto devemos enviar um WAYPOINT_ACK informando que todos os pontos foram recebidos (passo 7).

3.4 Construção do Mosaico

Também com *software* desenvolvido, foi implementado um módulo para construção do mosaico, o qual foi anexado à interface da estação de terra. Devido à quantidade de detalhes e ao tempo significativo empregado neste módulo, ele será detalhado no Capítulo 4.

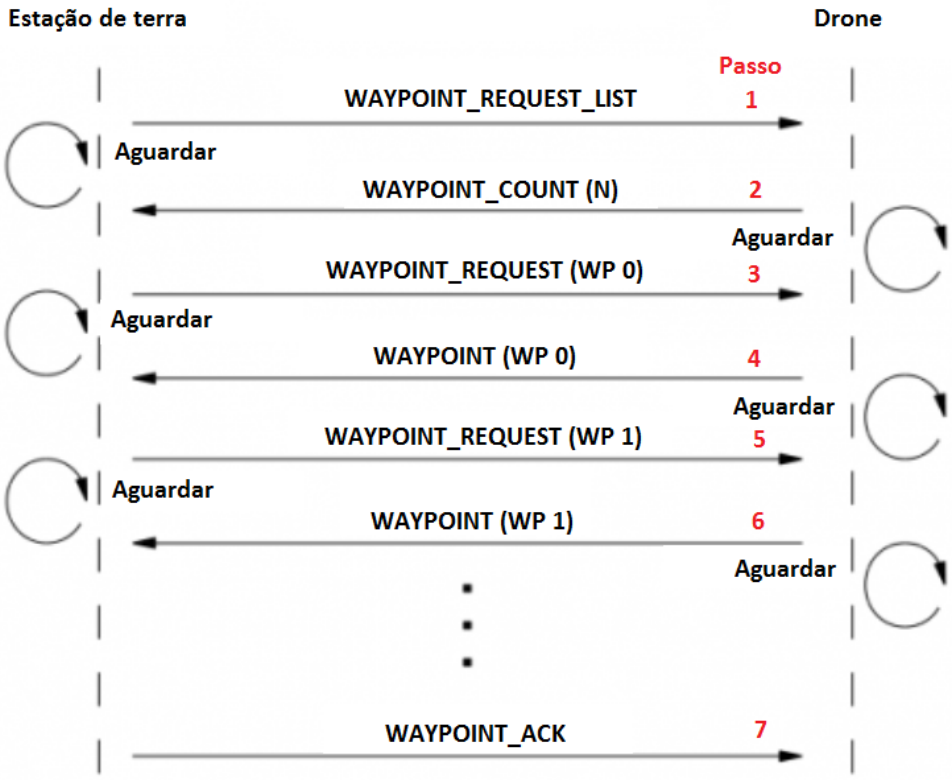


Figura 21 – Protocolo Mavlink utilizado para leitura de *waypoints*.

4 Construção do Mosaico

O *software* criado para gerar os mosaicos a partir das imagens obtidas durante o voo do *drone* foi desenvolvido também na linguagem de programação C++ e incorporado à interface criada para a estação de terra. Esta interface é utilizada para executar todos os procedimentos, desde a configuração da missão até a obtenção final do mosaico. Neste capítulo serão mostradas as estratégias utilizadas no desenvolvimento do *software*, bem como as configurações dos métodos de terceiros utilizados.

4.1 SURF

O algoritmo SURF (*Speeded Up Robust Features*), proposto por Bay, Tuytelaars e Gool (2006), foi utilizado como base para extração dos pontos de interesse (*keypoints*) das imagens e cálculo de seus respectivos descritores para posterior obtenção da homografia (transformação projetiva 2D entre duas imagens). Além de possuir um algoritmo rápido e robusto, o SURF busca uma implementação invariante à rotação e escala, o que é essencial para este trabalho.

Para encontrar os pontos de interesse e sua orientação o SURF utiliza, respectivamente, um detector Hessiano (BAY; TUYTELAARS; GOOL, 2006; LINDEBERG, 1998) e a transformada de Haar Wavelets (LEE; , 1999). Os vetores dos descritores dos pontos possuem dimensão de 64 ou 128, tendo em seu cálculo as características dos pontos e seus vizinhos.

Os autores do SURF o subdivide em três partes: encontrar pontos-chave na imagem; calcular os descritores dos pontos, garantindo sua distinção e robustez a ruído, deformações geométricas e a erros; comparar os descritores de duas imagens na tentativa de achar uma correlação entre eles, baseando-se em distância euclidiana entre os vetores do descritor.

Neste trabalho, o SURF foi utilizado em uma versão disponível na biblioteca OPENCV (BRADSKI, 2000) na versão 2.4.4. Para obtenção de melhores resultados foram modificados os parâmetros que indicam o tamanho da matriz Hessiana e a distância euclidiana máxima entre uma correlação de descritores. Na Figura 22 temos uma imagem com seus descritores marcados. Já na Figura 23 temos os *keyspoints* de duas imagens onde o SURF realizou o *matching* apenas pelo casamento entre os descritores obtendo um erro entre eles (distancia euclidiana) e casando os pares com menor erro. Na Figura 24 temos o mesmo *matching* mostrado na figura anterior, porém foram descartados os *matching* que apresentavam distância superior a 3x a menor distância encontrada.

Figura 22 – Pontos selecionados pelo SURF.

4.2 HOMOGRAFIA

Após obter a correspondência entre os pontos utilizando o SURF, tem-se a necessidade de estimar a transformação 2D entre a primeira e a segunda imagem, a homografia **H**. Neste caso em específico, onde a região a ser mapeada é considerada planar, ou seja, todos os pontos de interesse estão em um mesmo plano, pode-se escrever um ponto P deste plano conforme a Equação 4.1 onde temos a coordenada z igual a zero.

$$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} \tag{4.1}$$

A Equação 4.2 descreve a transformação dos pontos no sistema de coordenadas da imagem 1 para o plano de projeção da imagem 2, de forma simplificada. Onde s é um fator de escala, (u1,v1) são as coordenadas 2D de um ponto no referencial da imagem 1 e

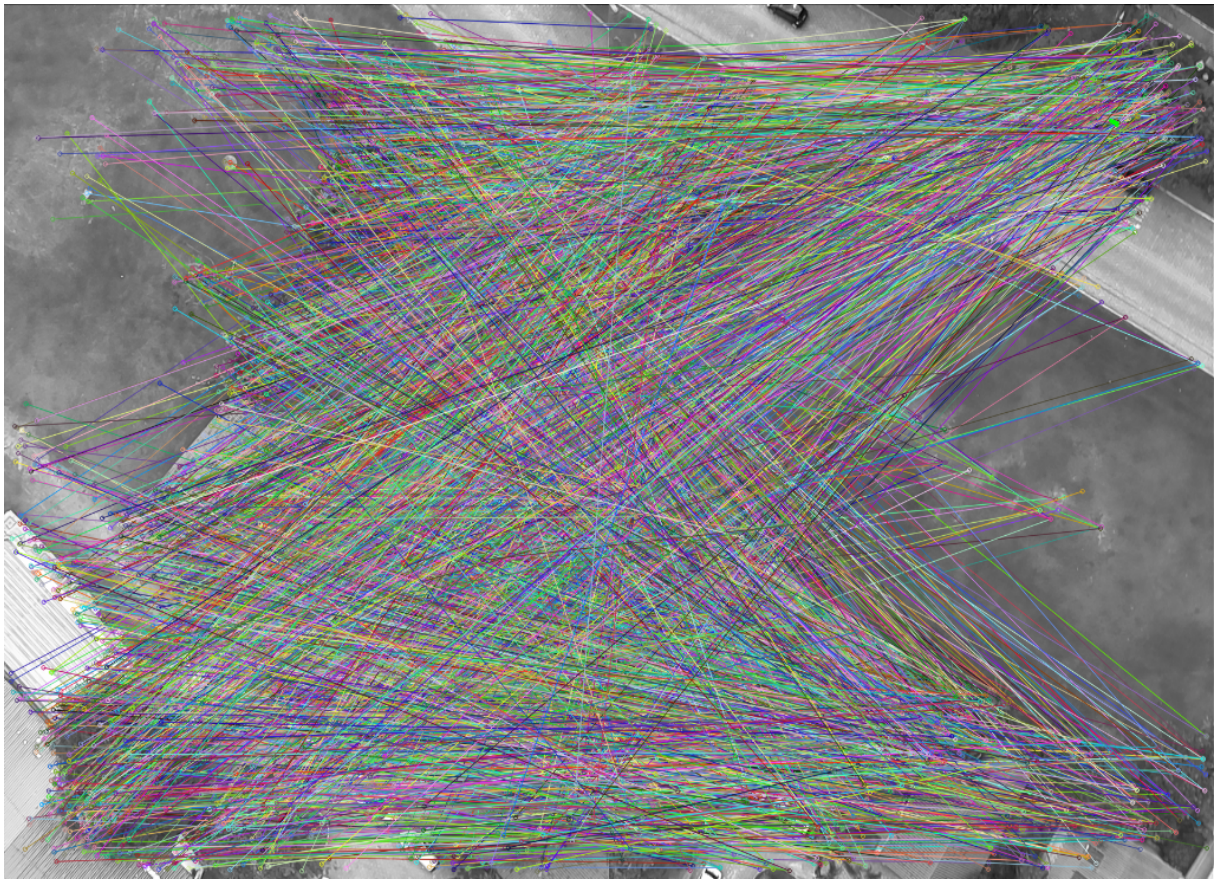


Figura 23 – *Matching* realizado pelo SURF.

(u2,v2) no referencial da imagem 2.

$$s \begin{bmatrix} u1 \\ v1 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u2 \\ v2 \\ 1 \end{bmatrix} \tag{4.2}$$

onde,

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{4.3}$$

A matriz **H** pode ser decomposta conforme a equação 4.4.

$$\mathbf{H} = \mathbf{A}\,[\mathbf{R}\,\mathbf{T}] \tag{4.4}$$

Onde **A** é a matriz dos parâmetros intrínsecos da câmera, **R** a matriz de rotação e **T** a matriz de translação entre os respectivos referenciais de cada imagem.

Figura 24 – *Matching* realizado pelo SURF, aplicada a restrição de 3x a menor distância euclidiana encontrada no conjunto.

Neste trabalho, a homografia é calculada utilizando o função findHomography da biblioteca OPENCV, que necessita de pelo menos quatro correspondências. Caso haja mais correspondências, o resultado será ainda melhor. Para tal, é passado como parâmetros as correlações entre os pontos das duas imagens, o método a ser utilizado no cálculo e o *threshold* (parâmetro este relacionado com erro máximo na reconstrução). O método escolhido para esta dissertação foi o CV_RANSAC baseado no algoritmo RANSAC (FISCHLER; BOLLES, 2000).

4.3 RANSAC

O RANSAC (*RANdom SAmple Consensus*) é classificado como uma ferramenta estatística, iterativa e robusta especializada em resolver problemas de estimativa, mesmo com a presença de ruídos nos dados amostrados. Com esta ferramenta pode-se alcançar resultados confiáveis em comparação com métodos mais custosos no que diz respeito ao processamento e ao tempo.

O RANSAC depende de um valor inicial para dar seguimento em suas estimativas. Para isto, são escolhidos alguns elementos do conjunto de amostras, de forma aleatória, e

assim é calculada a primeira hipótese. Dando seguimento, utilizando métricas parametrizadas pelo usuário, realiza-se a avaliação dos cenários valorando os resultados em busca do erro mínimo (desejado em cada situação) para cada estimativa calculada. As amostras que se enquadram neste parâmetro mínimo são classificadas como *inliers* e as que não se enquadram são classificadas como *outliers*. O processo se repete até que o conjunto de *inliers* gere um resultado satisfatório que atenda à parametrização definida.

4.4 Metodologia para geração dos mosaicos

A estratégia para criação do mosaico parte do pré-suposto que o *drone* sobrevoará uma região retangular, subdividida em retângulos menores, onde sobre o centro geométrico destes será capturada uma fotografia aérea. Na Figura 25 há um exemplo de uma região onde se deseja criar um mosaico. Na imagem são observados retângulos delimitados em azul, sendo que os *waypoints* da missão de voo são gerados sobre o centro de cada um destes. As dimensões das delimitações são diretamente proporcionais à altura de sobrevoo e à abertura angular da câmera utilizada.

Ainda na Figura 25 pode-se observar um ponto marcado com a letra "A"onde supostamente será capturada uma imagem. A fim de garantir sobreposição de imagens, pré-requisito para geração do mosaico como explicado mais à frente, observa-se uma marcação em amarelo indicando a real área a ser fotografada quando da captura no ponto "A".

Com a finalidade de simplificar, porém sem perder qualidade no mosaico, é considerado que a região a ser mapeada possui um relevo plano. Tal consideração é justificada pelas pequenas dimensões da região e objetos em comparação com a altura de voo e o raio de curvatura da Terra. Com esta consideração e a utilização da *gimbal* para estabilizar a câmera sempre voltada para o solo, procura-se garantir que o eixo óptico da câmera esteja sempre perpendicular ao plano de sobrevoo. O exemplo mostrado na Figura 25 é um caso que tangencia a situação ideal, isto é, o *drone* sempre estará sobrevoando o ponto desejado, sem nenhuma rotação (a *gimbal* utilizada não corrige erros de orientação), o que nem sempre é possível. O algoritmo gerou resultados satisfatórios com voos em alturas superiores a 10 vezes o tamanho médio dos prédios e árvores das regiões mapeadas.

Inicialmente foi desenvolvido uma versão do algoritmo especializado para geração de mosaico nestas circunstâncias, obtendo-se bons resultados. Tal algoritmo calculava a possível localização da área a ser sobreposta pela imagem seguinte, e pequenos erros de posição e orientação não causavam maiores problemas para seu processamento. Porém, em experimentos onde as imagens eram capturadas em voos reais, tal algoritmo mostrou-se ineficiente, principalmente devido aos erros de posicionamento e orientação do robô no momento da captura da imagem. Estes erros causavam grandes sobreposições de imagens em algumas regiões, e em outras, uma sobreposição insuficiente para um cálculo correto

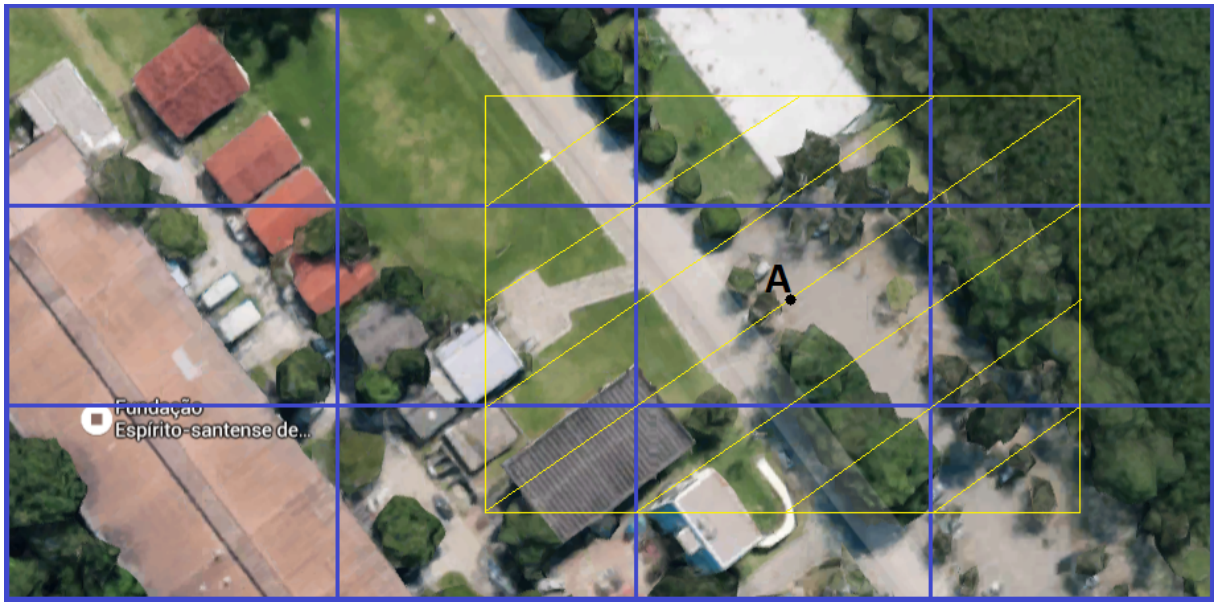


Figura 25 – Região a ser fotografada com o *drone* (Fonte Google Maps).

da homografia entre duas imagens vizinhas, interrompendo a geração do mosaico.

Para corrigir tais problemas tornou-se necessário aumentar a área de possível sobreposição, tanto na aquisição da imagem pelo veículo aéreo, quanto para montagem do mosaico pelo algoritmo. Esta mudança, ocasionou um aumento do número de aquisições de imagens para uma mesma área a ser mapeada, em comparação com o método descrito anteriormente, além de exigir maior processamento e com isto maior custo de tempo. Para exemplificar este caso, pode-se observar na Figura 26(a), com uma sobreposição de aproximadamente 50% das imagens, que foram necessárias três fotografias para cobrir uma determinada região. Já na Figura 26(b) observa-se um número maior de fotografias (mais que quatro), para garantir uma sobreposição maior que 50% das imagens.

Apesar de existir erro de posicionamento do *drone* no momento das aquisições das imagens, este não se acumula, fornecendo uma estimativa válida de onde cada imagem deve se encaixar no mosaico. Desta forma, cabe ao algoritmo de montagem, o ajuste fino durante a colagem. Esta consideração torna possível uma diminuição considerável no processamento. Pode-se observar na Figura 27 que a última imagem será colada ao mosaico, com a área A' se sobrepondo a uma parte da área A. Como dito anteriormente, uma vez que já se possui uma estimativa de onde cada imagem será colada, ao invés de se tentar localizar a sobreposição da imagem A' sobre todo o mosaico já montado, pode-se procurar apenas na região A e suas adjacências, que correspondem a regiões próximas à estimativa de localização de A'.

Um outro desafio a ser vencido para utilização da estratégia descrita acima foi o problema da orientação na captura da primeira imagem pelo *drone*. Um pequeno erro na orientação desta imagem (imagem base do mosaico) ocasionava grandes erros no momento

(a)

(b)

Figura 26 – A Figura 26(a) possui uma sobreposição de aproximadamente 50% e a Figura 26(b) possui uma sobreposição um pouco maior 65%.

da estimação de área de sobreposição, quando na colagem das demais imagens. A seguir será explicado melhor este problema.

Na Figura 28 pode-se observar uma região, onde, hipoteticamente, após ser criada a missão de sobrevoo, foram gerados quatro *waypoints* e nestes foram capturadas as quatro fotografias, numeradas de 1 a 4. Observa-se que as imagens possuem uma certa rotação em torno do seu centro geométrico.

Na Figura 29 observa-se o problema descrito acima. Após a colagem da primeira imagem, o algoritmo realiza a colagem da segunda, mantendo a orientação imposta pelo erro da primeira. Porém para colagem das demais imagens, observa-se o aumento do erro de posição de todo o mosaico, em relação ao esperado pelo algoritmo, dificultando assim a estimação de sobreposição. Na colagem da quarta imagem, o algoritmo tentará sobrepor a região delimitada em azul A' sobre a região A. Neste caso, tem-se em vermelho uma região que teoricamente deveria conter características (pontos de interesse), os quais seriam utilizados para cálculo da homografia entre a terceira e a quarta imagem, o que não ocorre, causando a interrupção do processamento do mosaico. Para resolver este problema, optou-se por corrigir o erro de orientação da primeira imagem, antes de colocá-la no

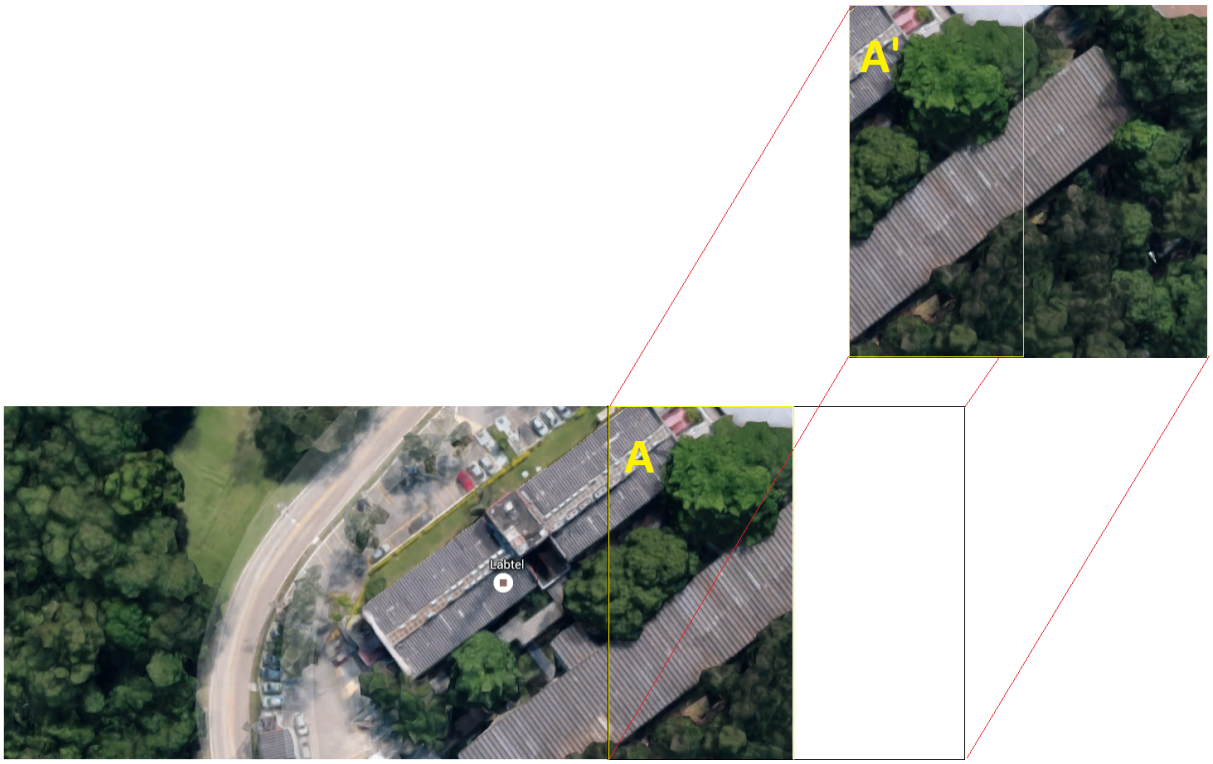


Figura 27 – Montagem do mosaico com processamento de regiões de interesses.

Figura 28 – Erro de orientação na colagem da primeira imagem.

mosaico.

Para obter o ângulo relativo ao erro de orientação, foi calculada a homografia da segunda imagem em relação à primeira. De posse desta homografia, a mesma foi aplicada ao ponto central da segunda imagem C2, obtendo-se uma nova coordenada para estes

pontos C2'. Na Figura 30 tem-se um triângulo com seus vértices em C1 (centro da primeira image), C2 (centro da segunda imagem) e C2' (centro da segunda imagem após aplicação da homografia). Supondo apenas o erro de orientação, uma vez que o erro de posição restringe-se apenas ao erro do GPS (erro aproximado de 1m), basta obter o angulo a, que se terá o erro de orientação da primeira imagem. De posse deste ângulo, aplica-se à primeira imagem uma rotação de $-a$, e na sequência, realiza-se a montagem do mosaico.

Figura 29 – Erro de orientação na colagem da primeira imagem.

Vencidas as etapas citadas anteriormente, após inserir a primeira imagem no mosaico no canto superior esquerdo, o algoritmo adiciona as demais imagens de forma semelhante à mostrada na Figura 27. Após inserir a última imagem da primeira linha, o algoritmo adiciona a primeira imagem da linha seguinte, conforme se observa na Figura 31. Na Figura 32 tem-se a colagem da segunda imagem da segunda linha, de forma semelhante ao que ocorre na primeira linha. O processo se repete até inserir a última imagem da última linha.

Ainda na Figura 31 pode-se observar a região B delimitada em azul que será sobreposta pela região B'. Isto ocorre a partir da segunda linha, pois além das informações obtidas da região A tem-se as informações da região B para extração dos pontos-chaves. Esta nova informação, obtida das linhas superiores já coladas, ajuda melhorar a colagem no que se refere ao posicionamento das novas imagem coladas. Por fim, o mosaico final é cortado e nele inserido o georreferenciamento.

Observa-se que durante todo o processo é estimada a região A em que a região A' irá sobrepor no mosaico montado até o momento. Com isto, como dito anteriormente, limita-se todo o cálculo na região A e suas adjacências.

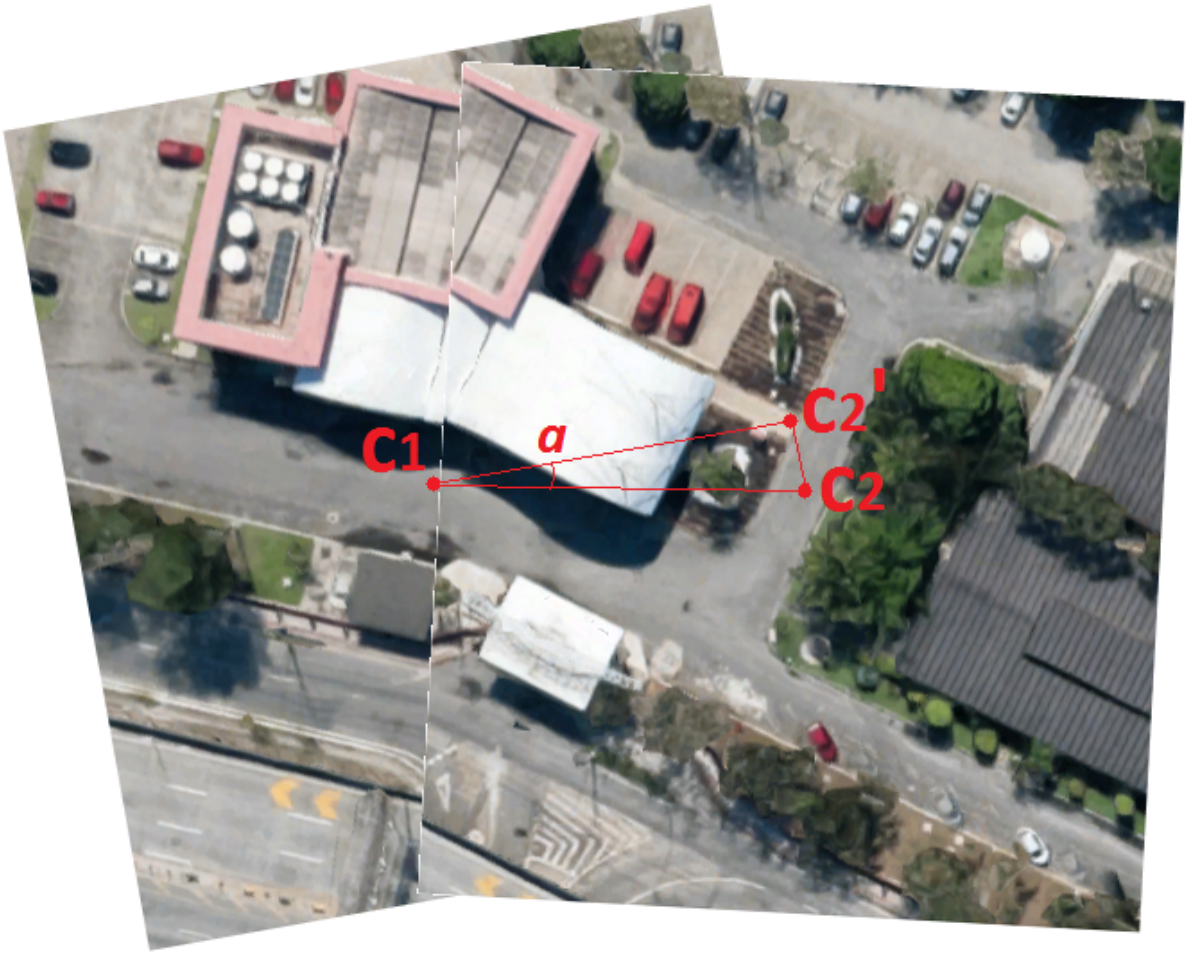


Figura 30 – Cálculo do erro de orientação da primeira imagem.

4.5 Georreferenciamento

Para georreferenciar as imagens são utilizadas as coordenadas geográficas armazenadas no momento da aquisição de cada imagem. A Figura 33 simula o momento em que o *drone* sobrevoa uma região e captura uma imagem aérea da região marcada em verde. Neste momento o robô está sobre o ponto C, que coincide como o centro geométrico desta região. Este posicionamento é mandido pelo uso do GPS e auxiliado pelo estabilizador de câmera (*gimbal*) que a mantém sempre paralela ao plano a ser fotografado.

Para plotar as coordenadas exatamente onde fora registrada a imagem, pega-se o ponto central da imagem (x,y), onde x é a metade de sua largura e y a metade de sua altura, e aplica-se a homografia acumulada neste ponto plotando-o no mosaico finalizado.

4.6 Exemplos

A fim de obter bons resultados, foram testados alguns parâmetros na criação do mosaico, entre eles a variação do tamanho do vetor do descritor SURF, o tamanho da

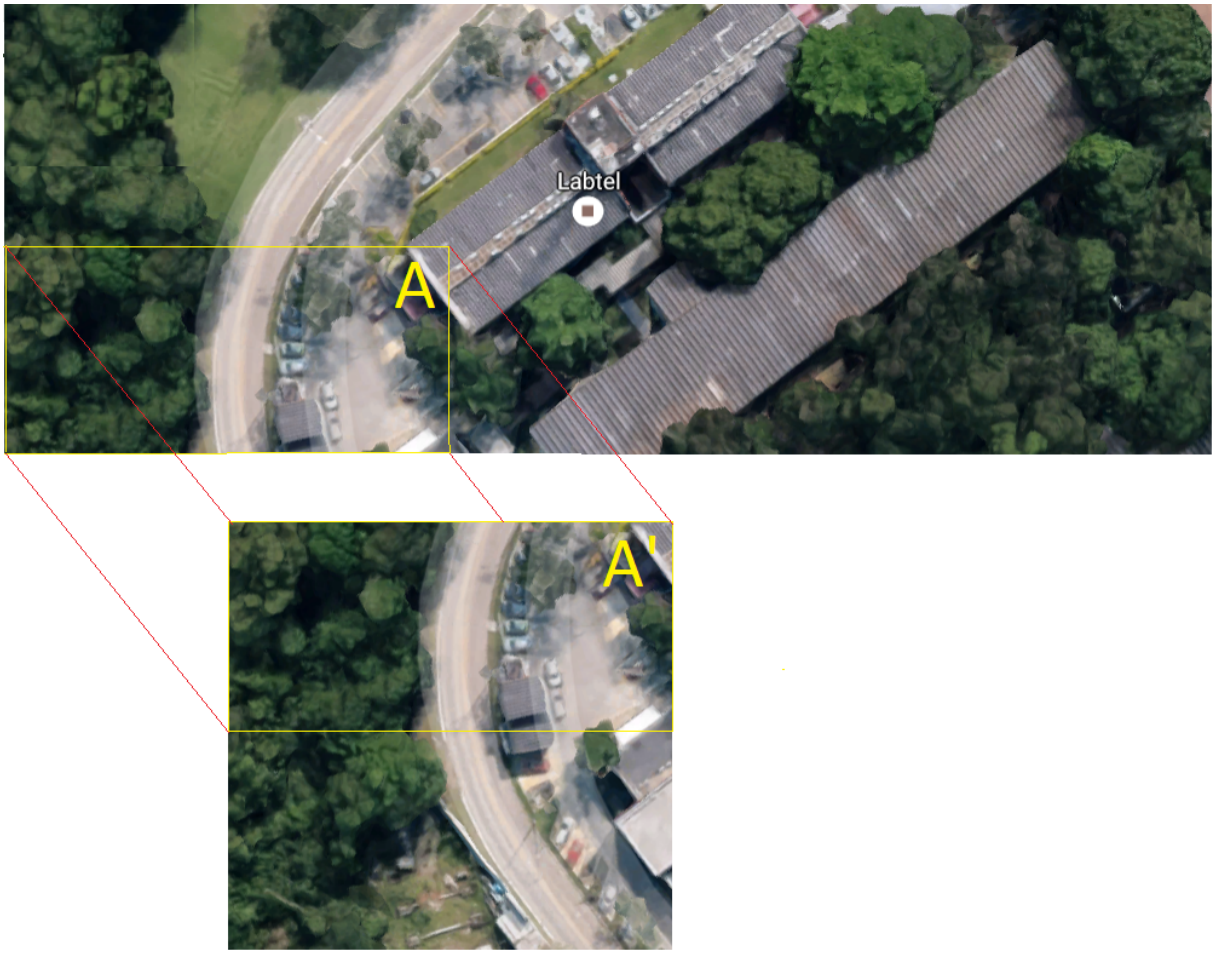


Figura 31 – Colagem da primeira imagem da segunda linha.

matriz Hessiana utilizada na interpolação da imagem, um filtro inicial onde se excluíam o *matches* com uma distância maior que n vezes a menor distância euclidiana obtida e o erro de reprojeção para o RANSAC (*threshold*).

A Figura 34 mostra uma montagem do mosaico contendo quatro imagens em uma única linha. Neste caso, não foi realizada a correção da primeira imagem e observa-se que a região escura na parte superior do mosaico aumenta à medida que mais fotos são adicionadas. Este voo foi realizado a 65 metros de altura e sem a utilização da *(*gimbal) para estabilização da câmera. Observa-se ainda que, mesmo sem a estabilização da câmera o algoritmo realizou de forma satisfatória a montagem.

Na Figura 35 temos um mosaico com seis colunas e uma linha de fotografias com o voo a 65 metros de altura, realizado de forma totalmente autônoma, com os *waypoints* gerados pelo *software* desenvolvido. É possível notar a variação de brilho entre as imagens, uma vez que a câmera realizava de forma automática a compensação de brilho, porém a montagem ficou satisfatória.

Para as duas montagens mostradas anteriormente, utilizaram-se como parâmetros: dimensão do descritor do SURF: 128; dimensão da Matriz Hessiana: 400; erro máximo nos

Figura 32 – Colagem da segunda imagem da segunda linha.

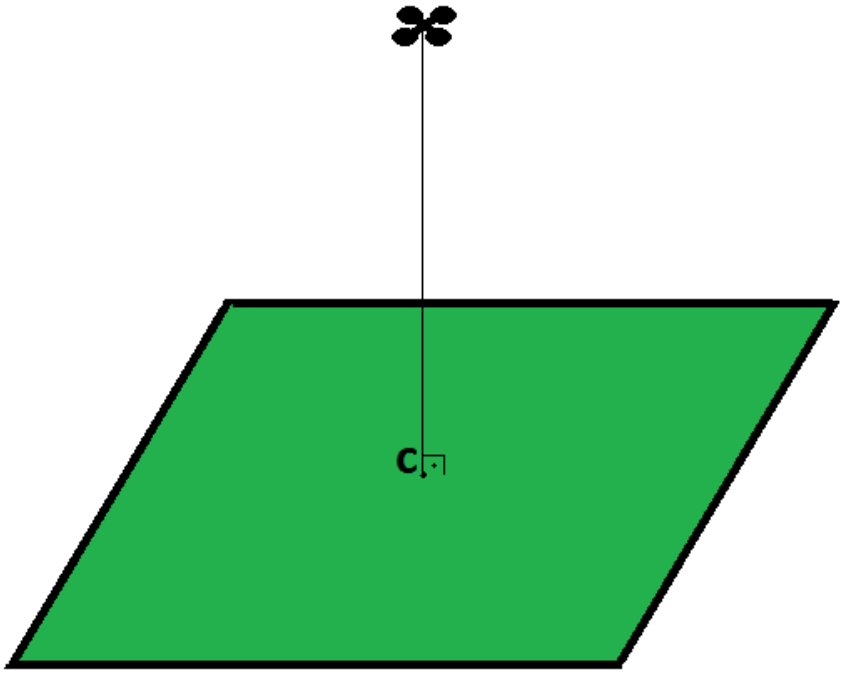


Figura 33 – *Drone* sobrevoando o ponto central de uma região a ser fotografada.

matches: três vezes a distância mínima; e erro de reprojeção para o RANSAC: 3.

Figura 34 – Montagem do mosaico sem a correção da primeira imagem.

Figura 35 – Montagem do mosaico com seis fotografias.

5 Resultados

Para validar os resultados foram realizados mais dois experimentos, com a aquisição de 3x6 e 5x6 fotografias. Em seguida foram obtidos os respectivos mosaicos georreferenciados. A fim de dar mais substância aos resultados obtidos, o mosaico gerado pelas 5x6 imagens será comparado com outros métodos de obtenção de mosaicos.

5.1 Experimento com 3x6 imagens

Para este experimento, foi realizado um sobrevoo em uma região desabitada a 65 metros de altura obtendo-se 18 imagens (6x3) que foram coladas e georreferenciadas. A escolha desta região deu-se pelo fato de realizarmos um voo autônomo com maior duração, sem por em risco nenhuma pessoa em uma eventual falha e também, por a região possuir uma vegetação extensa, de forma que possa se assemelhar com a região de uma possível aplicação destes equipamentos em uma situação real. Realizou-se o voo sob a ação de vento moderado o que contribuiu apenas para um consumo maior de bateria, uma vez que o *drone* realizou a missão sem maiores problemas, obtendo todas as imagens e dados necessários ao experimento em campo. Para este experimento foi necessário levar para campo, o *drone* e um *notebook* para a execução dos *softwares* desenvolvidos. Em momento algum, houve necessidade de conexão com a internet, pois não é necessário o mapa da região para geração dos *waypoints*.

O *software* de geração de *waypoints* foi configurado para gerar pontos a uma distância tal que garantiria uma sobreposição mínima da imagem de 60%, mais especificamente 62,5%, assim cada ponto no eixo x estaria a 18,42 metros de distância um do outro, já no eixo y a distância definida entre os pontos foi de 24,32 metros

Após o pouso foi retirado o cartão micro SD da RaspberryPi B+, embarcada no *drone* e inserido no *notebook*. Em seguida, acionou-se a funcionalidade de gerar o mosaico. Pode-se visualizar na Figura 36 o mosaico obtido. Note que nas duas primeiras linhas houve uma montagem correta do mosaico, porém na terceira linha da montagem houve uma diminuição na escala das fotografias criando um efeito gravata no mosaico. Este efeito é uma distorção projetiva que, aumenta na direção das bordas de mosaicos renderizados em superfícies de projeção planar.

Em todos os experimentos utilizou-se um *notebook* com processador Intel Core I5 M620 de 2,67 GHz da 1ª Geração, com 4 GB de memória e sem placa de vídeo dedicada. Com isto, neste experimento, o tempo total de processamento das dezoito imagens foi de 15,44 segundos, dando uma média de 858 milissegundos por imagem adicionada.

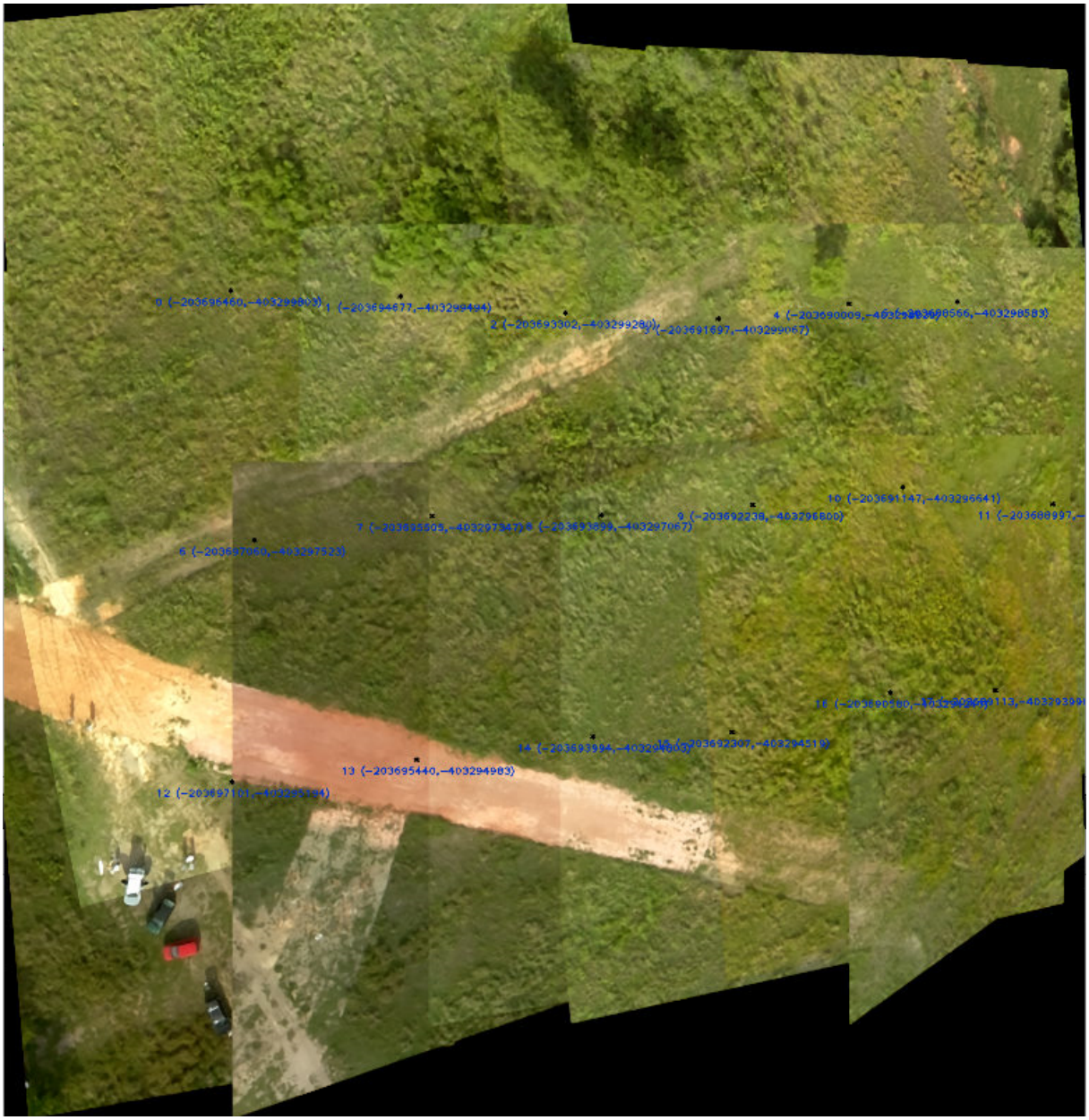


Figura 36 – Experimento com captura de 18 imagens a 65 metros de altura.

Pode-se observar na imagem gerada os pontos marcados com (*) identificando o centro geométrico da imagem original, aplicando a este as devidas transformações sofridas pela imagem. Como citado anteriormente, observa-se na última linha montada uma redução na escala das imagens no sentido da esquerda para a direita, com isto os pontos marcados sofrem também esta deformação, retirando-os do alinhamento original. Em cada ponto marcado, é inserida uma legenda com as coordenadas geográficas em formato decimal multiplicado de 10^7. Notou-se ao executar a montagem do mesmo mosaico por outras vezes, um tempo bem próximo ao apresentado, quando executado no mesmo computador.

5.2 Experimento com 5x6 imagens

Neste experimento, o voo foi realizado na mesma região do experimento descrito na Seção 5.1, porém com a geração de novos pontos para captura das imagens. O sobrevoo deu-se a uma altura de 100 metros obtendo-se 30 imagens (5x6) a serem coladas e também georreferenciadas. Realizou-se o voo também sob a ação de vento moderado, nas mesmas condições conforme o experimento anterior, inclusive utilizando os mesmos equipamentos, substituindo apenas a bateria por outra carregada em sua capacidade total. O tempo total de voo foi de oito minutos e trinta segundos.

O *software* de geração de *waypoints* também foi configurado para gerar pontos a uma distância tal que garantiria uma sobreposição da imagem de 60%, mais especificamente 62,5%. Assim cada ponto no eixo x estaria 28,34 metros separados um do outro. Já no eixo y a distância entre os pontos ficou definida em 37,42 metros.

Na Figura 37 tem-se o mosaico obtido onde foi solicitado inicialmente a geração de um mosaico apenas com 24 imagens (4x6). É observado também neste mosaico o efeito gravata nas imagens, porém em menores proporções, não prejudicando a montagem. Observa-se nesta montagem uma geometria final em forma de um retângulo, como se esperava. O tempo de geração do mosaico, foi de 20,45 segundos contabilizando uma média de 852 milissegundos por imagem adicionada, utilizando para isto o mesmo computador do experimento descrito na seção anterior.

Por fim, foi gerado um mosaico com 30 imagens (5x6), conforme observa-se na Figura 38. Neste mosaico também é observada uma mudança na escala das imagens da segunda linha, gerando uma colagem ligeiramente equivocada das fotografias. Observa-se ainda na última linha da montagem um ligeiro aumento das imagens, correspondendo apenas ao aumento da escala, o que deforma o aspecto final da montagem. Para este mosaico o tempo de geração, foi de 24,71 segundos com uma média de 824 milissegundos por fotografia adicionada, utilizando também o mesmo computador do experimento descrito na Seção 5.1.

Na parte inferior da Figura 38 pode-se visualizar uma barra de escala e a estimativa da área total mapeada. Para calcular a escala, toma-se a distância em metros entre a primeira e a última coordenada da primeira linha do mosaico, isto é, entre a coordenada mais a esquerda e a mais a direita da primeira linha. Após obter esta distância em metros, calcula-se o número de pixeis entre estes dois pontos plotados no mosaico. De posse destas duas grandezas, tem-se a relação pixeis/metros do mosaico em questão. Para calcular a área total, aplica-se a mesma relação calculada em relação a quantidade de pixeis do mosaico. Neste experimento obteve-se uma área mapeada de aproximadamente 76.889 m^2, o que equivale a um pouco mais de dez campos de futebol.

Figura 37 – Mosaico gerado utilizando 24 imagens capturadas a 100 metros de altura.

5.3 Comparação com o Google Maps

A fim de realizar uma comparação dos resultados obtidos com outros métodos, optou-se por compará-la com uma imagem extraída do Google Maps da mesma região onde fora realizado o voo do experimento descrito na Seção 5.2. Nota-se uma imagem perfeita da região, porém, com menor resolução e em um momento anterior ao voo. A utilização do Google Maps pode ser aconselhada em uma situação onde deseja-se um prévio planejamento de uma possível intervenção em uma região, pois a resolução não é de má qualidade, e ainda pode-se extrair as coordenadas geográficas do ponto que desejar. Porém, como é observado na Figura 39 e já mencionado acima, a aquisição da imagem disponibilizada no Google Maps foi em momento anterior à data dos experimentos

Figura 38 – Mosaico gerado utilizando 30 imagens capturadas a 100 metros de altura.

realizados, sendo assim inviável o seu uso, caso o desejado fosse a obtenção do cenário atual da região no momento de uma situação de emergência.

Figura 39 – Região de sobrevoo na ótica do Google Maps.

5.4 Comparação com a aplicação Autostich64

A fim de realizar uma segunda comparação, optou-se também em comparar os resultados com a montagem utilizando o *software* Autostich64 (BROWN; LOWE, 2007). Conforme se observar na Figura 40, a aplicação gera um mosaico com as mesmas trinta imagens utilizadas no experimento 5.2 visualmente perfeita. Esta aplicação é de livre uso, para qualquer fim, desde que as imagens geradas referenciem a aplicação utilizada. No momento da concepção deste trabalho, não foi localizada um versão em Linux para integração, uma vez que esta aplicação, por si só, não englobaria todo o trabalho desenvolvido para esta dissertação, principalmente no que diz respeito ao georreferenciamento.

Apenas a título de comparação, utilizando o mesmo computador dos experimentos citados anteriormente, o Autostich64 em seu processamento, descartou uma fotografia na

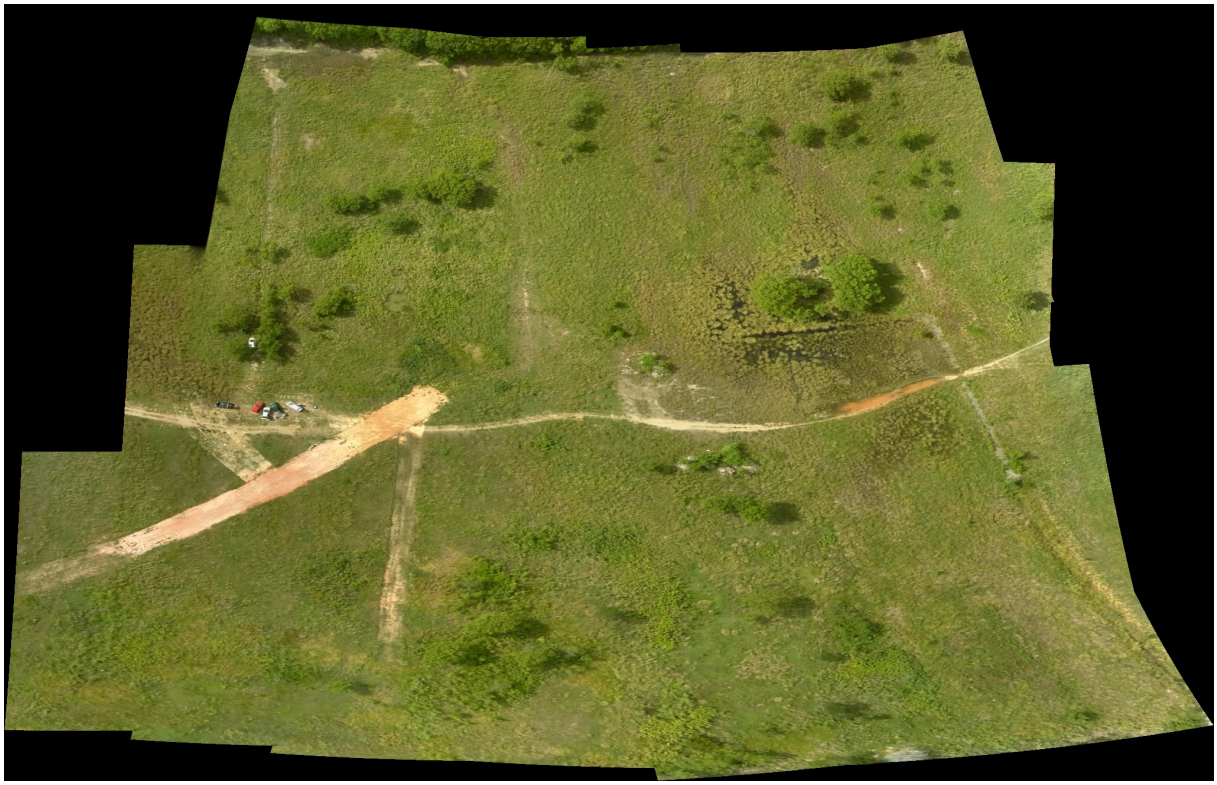


Figura 40 – Mosaico de 30 imagens da região de sobrevoo gerado pelo Autostich64.

montagem, e realizou todo o processamento em 22,72 segundos após selecionar todas as imagens, contabilizando uma média de 757 milissegundos por fotografia adicionada. Um outro fator interessante é que, o Autostich64 também gerou distorção na imagem como pode-se observar, na linha inferior do mosaico, um aumento de escala das imagens dando um efeito gravata na imagem. As mesmas imagens foram utilizadas em novas tentativas de montagem com o Autostich64, obtendo como melhor resultado o mosaico mostrado na Figura 41.

Figura 41 – Mosaico de 30 imagens da região de sobrevoo gerado pelo Autostich64.

6 Considerações Finais

6.1 Conclusão

Nessa Dissertação de Mestrado foi abordado o tema de geração de mosaicos georreferenciados utilizando robôs aéreos. Para realizar a geração dos mosaicos foram desenvolvidos *softwares* para comunicação, geração de rotas, aquisição de dados e imagens aéreas além de um controle de alto nível e uma interface para operação do usuário. Para tal foi acoplado a um *drone* um microcomputador RaspberryPi B+ com os *softwares* de obtenção e armazenamento dos dados e imagens, além do controlador de alto nível.

Para geração do mosaico foi proposto um método que busca a sobreposição da imagem a ser adicionada ao mosaico em uma região estimada, diminuindo assim o custo computacional na obtenção da homografia entre esta imagem e o mosaico parcial já montado. Observa-se que as médias para cada imagem adicionada nos experimentos das Seções 5.1 e 5.2 foram de 858, 852 e 824 milissegundos. A média manteve-se praticamente constante, o que tornará o tempo final proporcional à quantidade de imagens adicionadas, viabilizando assim, futuramente, mosaicos de maiores dimensões.

Pode-se observar no Capítulo 5 que os experimentos foram satisfatórios, já que o objetivo final seria fornecer um mosaico de uma área onde fora realizado um sobrevoo com um *drone*, imediatamente após o sobrevoo. Foram gerados mosaicos com até 30 imagens a uma altura de 100m. Neste caso, a limitação foi sensivelmente impactada pelos equipamentos utilizados, principalmente no que diz respeito a autonomia de voo do *drone*. Observa-se ainda que apesar de algumas distorções e a falta de equalização das imagens, devido à exposição solar diferenciada no momento da aquisição de cada imagem, ou devido às compensações automáticas da câmera utilizada, não houve problemas para geração do mosaico.

Em uma situação real, onde ocorra um emergência em um ponto da região visualizada na Figura 37, poder-se-ia facilmente, direcionar as equipes de emergência para realizar seus trabalhos em locais específicos, utilizando-se um aparelho de GPS para chegar às coordenadas visualizadas na imagem. Qualquer outro ponto desta imagem em que não se tenha marcado as coordenadas, estas podem ser obtidas através de uma simples interpolação. Além disto, os informações gerais, como estradas, riachos, vegetação rasteira, entre outros, ajudarão no gerenciamento das operações, podendo inclusive contribuir para o emprego mínimo de recursos e para maximizar os resultados positivos da operação.

6.2 Trabalhos Futuros

Para melhorar o resultado final dos mosaicos a serem gerados, há a necessidade de implementar métodos de colagem mais eficientes, a fim de evitar os efeitos de bordas no mosaico final. Um outro ponto a ser melhorado é a equalização das imagens a fim de suavizar as descontinuidades das imagens coladas, ou até mesmo utilizar os métodos propostos por Brown e Lowe (2007) para isto.

Neste trabalho, no momento do voo, tem-se apenas a comunicação do *drone* com seu computador de bordo, ficando a estação de terra sem nenhum tipo de comunicação durante a missão. A fim de aumentar o controle e até mesmo o monitoramento durante o voo, deve-se estudar um pouco mais o sistema de comunicação da controladora Ardupilot a fim de realizar sua conexão não apenas com o computador de bordo, mas também com a estação de terra, de forma simultânea.

Após cada voo é retirado o cartão de memória da RaspberryPi B+ e o mesmo é inserido na estação de terra, a fim de se obter todas as imagens, uma vez que não é aconselhável a utilização de um *link wireless* de 2.4GHz, pois pode ocasionar interferência nos controles de voo, já que o mesmo utiliza esta frequência. Assim, deve-se estudar uma forma de comunicação eficiente para transferência das imagens através de uma comunicação sem fio, possibilitando assim o envio das imagens durante o voo.

Outra possibilidade de aplicação do modelo construído nesta Dissertação seria o seu uso em regiões agrícolas, a fim de viabilizar o controle de pragas e da irrigação através do processamento de imagem, utilizando como entrada o mosaico gerado.

Referências

AMORIM, L. et al. Estimação de posição e atitude de um vant baseada em gps, imu e dados visuais. *Anais Do XII Simpósio Brasileiro de Automação Inteligente - SBAI 2015*, p. 1660–16650, 2015. Citado 2 vezes nas páginas 26 e 28.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. In: SPRINGER. *European conference on computer vision*. [S.l.], 2006. p. 404–417. Citado na página 40.

BERNI, J. et al. Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle. *Geoscience and Remote Sensing, IEEE Transactions on*, v. 47, n. 3, p. 722–738, March 2009. ISSN 0196-2892. Citado na página 16.

BRADSKI, G. The opencv library. *Dr. Dobb's Journal of Software Tools*, 2000. Citado na página 40.

BROWN, M.; LOWE, D. G. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, v. 74, n. 1, p. 59–73, 2007. Citado 3 vezes nas páginas 17, 58 e 62.

FIRELAB. *Técnicas de combate*. 2016. Disponível em: <http://www.floresta.ufpr.br/firelab/tecnicas-de-combate>. Citado na página 15.

FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, v. 24, n. 6, p. 381–395, 2000. Citado na página 43.

HENG, L. et al. Real-time photo-realistic 3d mapping for micro aerial vehicles. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, p. 4012–4019, 2011. Citado na página 17.

INPE. *Monitoramento dos Focos Ativos no Brasil*. 2016. Disponível em: <http://www.inpe.br/queimadas/estatisticas.php>. Citado na página 15.

LEE, Y. B.; , S. Y. . Application of the discrete wavelet transform to the monitoring of tool failure in end milling using the spindle motor current. *The International Journal of Advanced Manufacturing Technology*, v. 15, n. 4, p. 238–243, 1999. ISSN 1433-3015. Disponível em: <http://dx.doi.org/10.1007/s001700050062>. Citado na página 40.

LINDEBERG, T. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 30, n. 2, p. 79–116, nov. 1998. ISSN 0920-5691. Disponível em: <http://dx.doi.org/10.1023/A:1008045108935>. Citado na página 40.

MA, Y. et al. *An Invitation to 3-D Vision: From Images to Geometric Models*. [S.l.]: Springer, 2004. ISBN 978-1-4419-1846-8. Citado na página 27.

NONAMI, K. et al. *Autonomous Flying Robots*. 1. ed. [S.l.: s.n.], 2010. Citado na página 16.

POBKRUT, T.; EAMSA-ARD, T.; KERDCHAROEN, T. Sensor drone for aerial odor mapping for agriculture and security services. In: *2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. [S.l.: s.n.], 2016. p. 1–5. Citado na página 16.

ROSNELL, T.; HONKAVAARA, E. Point cloud generation from aerial image data acquired by a quadrocopter type micro unmanned aerial vehicle and a digital still camera. *Sensors 2012*, p. 453–480, 2012. Citado na página 17.

SÁ, F. B. de. *Cooperação entre um Robô Aéreo e um Robô Terrestre para Identificação de Rotas Livres em Solo*. Dissertação (Dissertação de Mestrado) — Universidade Federal do Espírito Santo – UFES, dez 2014. Citado 2 vezes nas páginas 16 e 20.

SANTANA, L. V.; BRANDÃO, A. S.; SARCINELLI-FILHO, M. An automatic flight control system for the ar . drone quadrotor in outdoor environments. In: *3rd Workshop on Research, Education and Develpment of Unmanned Aerial Systems - REDUAS 2015*. [S.l.: s.n.], 2015. p. 10. Citado na página 16.

SANTANA, L. V.; BRANDÃO, A. S.; SARCINELLI-FILHO, M. Outdoor waypoint navigation with the ar.drone quadrotor. In: *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*. [S.l.: s.n.], 2015. p. 303–311. Citado na página 16.

SANTOS, M. C. P. et al. Estimating and controlling uav position using rgb-d/imu data fusion with decentralized information/kalman filter. In: *Industrial Technology (ICIT), 2015 IEEE International Conference on*. [S.l.: s.n.], 2015. p. 232–239. Citado na página 16.

TARALLO, A. D. S. et al. Mosaico Automático de Imagens Agrícolas através da Transformada SIFT. *VI Workshop de Visão Computacional*, 2010. Citado na página 16.

TARALLO, A. d. S. et al. Construção de mosaicos de imagens aéreas agrícolas e comparação com outras metodologias. *WORKSHOP DE VISÃO COMPUTACIONAL - WVC CENTRO-OESTE, 8.,: Universidade Federal de Goiás - UFG, 2012.*, p. 1660–16650, 2012. Citado na página 17.

TRIPICCHIO, P. et al. Towards smart farming and sustainable agriculture with drones. In: *Intelligent Environments (IE), 2015 International Conference on*. [S.l.: s.n.], 2015. p. 140–143. Citado na página 16.

ZAINUDDIN, K. et al. Utilizing quadcopter as lars image platform to determine the paddy spectral and growth parameter. In: *Aerospace Electronics and Remote Sensing Technology (ICARES), 2014 IEEE International Conference on*. [S.l.: s.n.], 2014. p. 210–215. Citado na página 16.

ZHANG, Z. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 22, n. 11, p. 1330–1334, Nov 2000. ISSN 0162-8828. Citado na página 27.