Mosaicagem automática e em tempo real de imagens obtidas a partir de VANTs de asa rotativa

Guilherme Muzzi da Rocha, *Universidade de São Paulo - ICMC/USP*; Luiz Henrique Castelo Branco, *Instituto Federal de São Paulo - IFSP*; Roberto Santos Inoue, *Universidade Federal de São Carlos - UFSCar*; Kalinka Regina Lucas Jaquie Castelo Branco, *Universidade de São Paulo - ICMC/USP*

***Resumo* -- O uso de Veículos Aéreos Não Tripulados (VANTs) vem crescendo não só no cenário militar, mas também em aplicações civis. Tanto em ambientes militares quanto em ambientes civis, uma das grandes utilidades desses veículos é permitir que decisões possam ser tomadas. Quando se fala em tomada de decisões, imagens e informações não meramente textuais permitem que essas decisões possam ser efetuadas de modo mais fácil e até mesmo que outros elementos presentes nessas imagens possam ser avaliados e novas decisões ou estratégias possam ser alavancadas. Uma forma de viabilizar o uso dessas imagens para tomada de decisões, e também para permitir uma melhor localização visual dessas aeronaves é a partir da geração de mosaicos em tempo real, permitindo que uma visão geral de onde e como se encontra a região sobrevoada possa ser conseguida. Esse artigo apresenta uma nova abordagem para a geração automática de mosaicos em tempo real a partir de imagens obtidas por um VANT de asa rotativa a partir do sobrevoo de regiões. O VANT utilizado é um quadrotor cuja autonomia é de 12 minutos em condições normais de voo. Os resultados obtidos a partir da comparação dos descritores SIFT, SURF e ORB demonstram que o uso de descritores comuns na literatura podem ser utilizados para obtenção de mosaicos em tempo real. A utilização do SIFT com a *feature matching* Força Bruta apresenta resultados satisfatórios em termos de tempo e acurácia do mosaico gerado.**

***Palavras-chave* -- Mosaicamento Rápido de Imagens, Mosaico, Sensoriamento Remoto, Veículos Aéreos Não Tripulados, VANTs**

***Abstract* -- The use of Unmanned Aerial Vehicles (UAVs) has been growing not only in the military but also in civilian applications. In both environments one of the great importance of these vehicles is to allow decisions to be taken. Images and textual information are not enough to take a decision. Images are better instead of usual information, thus these decisions can be made more easily and even other elements present in these images can be evaluated allowing new decisions or strategies. One way to facilitate the use of these images to decision-making, and also to enable better visual aircraft location is provide a real time mosaic generation. A real time mosaic can allow an overview of where and how is the overflown region. This paper presents a new approach for an automatic real time mosaic generation from images obtained by a rotary-wing UAV. The UAV used in the experiments is a quad-rotor whose autonomy is 12 minutes under normal flight conditions. Results obtained from comparing SIFT, SURF and ORB descriptors demonstrated that the use of common descriptors in the literature can be used to obtain tiles in real time. The use of the SIFT with brute force feature matching presents satisfactory results in terms of time and accuracy.**

***Index Terms* -- Fast Image Mosaic, Mosaic, Remote Sensing, Unmanned Aerial Vehicles, UAVs**

I. Introdução

Um Veículo Aéreo Não Tripulado (VANT) é uma aeronave que não possui um piloto humano dentro da mesma, e atualmente a utilização dessas aeronaves constitui uma área que vem crescendo tanto em número de projetos quanto em aplicações. O avanço da microeletrônica proporciona um melhor gerenciamento desses sistemas por meio de algoritmos de controle e de navegação mais sofisticados. Porém, um sistema embarcado desse tipo é também considerado crítico, e tanto a construção do seu software quanto do seu hardware deve se preocupar com limitações e garantir uma taxa de falhas da ordem de uma falha grave a cada 10^5 até 10^9 horas de operação.

Essas aeronaves podem ser classificadas de acordo com o seu tamanho e seu modo de operação. VANTs de pequeno porte possuem os benefícios de serem versáteis, portáveis, com fácil manutenção, sendo utilizados nas mesmas aplicações que aeronaves grandes em escalas menores e sendo muito mais baratos [1]. Quanto a operação, estas aeronaves podem voar de forma autônoma, seguindo uma trajetória de voo pré-programada (baseada em um *grid* ou em uma sequência de *waypoints*) [2], ou podem voar recebendo comandos a partir de estações terrestres operadas por pilotos (teleguiadas).

Uma vez que ele pode realizar voos autônomos tanto utilizando *grids* ou meramente *waypoints*, a obtenção de imagens e a geração mosaicos devem ser efetuadas de modo a garantir que os mapas gerados possuam qualidade. As técnicas atuais de ortorretificação destes mosaicos adotam, normalmente, pontos de controle coletados em campo. Entretanto, tal tarefa é geralmente dispendiosa e exige levantamento em solo que demandam maior tempo para que o produto final seja disponibilizado. O uso da ortorretificação com base em MDEs (Modelos Digitais de Elevação) derivados de estereoscopia digital podem então ser utilizados com intuito de diminuir tanto o tempo de disponibilização do produto final quanto o custo de se alocar os pontos de controle em solo. Diante disso, além da possibilidade de ortorretificação precisa dos mosaicos, tem-se ainda a oportunidade de geração de mapas topográficos refinados, por meio de técnicas automáticas implementadas em softwares baseados em algoritmos SQM (*Structure from Motion Procedures*).

O que se espera é que seja obtido um resultado final análogo ao obtido por meio de imageamentos por *laser* aerotransportado, resguardando é claro o fato da incapacidade de penetração na superfície do alvo. Mesmo considerando-se tal limitação, estudos recentes têm

mostrado que a aplicação dessa técnica pode resultar em produtos de grande interesse para mapeamentos topográficos de detalhe, beneficiando-se dos atuais avanços nos métodos de coleta de imagens por VANT e nos softwares de tratamento de imagens [3], [4]. Além disso, trata-se de uma técnica menos dispendiosa que levantamentos por *laser* aerotransportado e possibilita a geração de MDEs com resolução espacial submétrica e, ainda, a mosaicagem e a ortorretificação das imagens.

Por exemplo, em [5] foram realizados experimentos com com o Parrot para navegação autônoma visual em ambientes estruturados. Já em [6], abordagens de aprendizado de máquinas foram aplicadas para predizer erros de posição de acompanhamento de trajetória de voo de um Parrot. Outros trabalhos tem utilizado o Parrot como plataforma experimental para tarefas de vigilância autônoma [7], interação homem-máquina [8], e até como assistente de esporte [9].

Outros trabalhos se preocupam com o desenvolvimento de mosaicos para VANTs e com as formas de validar os mapas gerados, bem como de averiguar se as informações obtidas são suficientes para a tomada de decisões, uma vez que, apesar dos VANTs fornecerem imagens de alta resolução, essa imagens cobrem apenas pequenas partes da área de interesse. Sendo assim, faz-se necessário o uso de algoritmos que permitam a junção de forma organizada das diversas imagens obtidas.

Aparentemente parece fácil simplesmente efetuar a junção das imagens. Entretanto, devido a distância das cenas obtidas com as câmeras dos VANTs ser maior que o movimento da câmera entre os pontos de interesse, a relação entre as imagens vizinhas é efetuada por meio de uma matriz homográfica, que permite o ajuste entre as imagens a serem unidas. No entanto, quando são utilizadas imagens de seqüência para a construção do mosaico, a precisão homográfica é afetada pelos erros acumulados. Sendo assim, em [10], foi desenvolvida uma abordagem para mosaicagem rápida de imagens em série de um VANT. Resultados experimentais mostraram que a abordagem atende aos requisitos de precisão e desempenho computacional. E que o método possui boa adaptabilidade em relação à ruídos, distúrbios de intensidade e pequenas distorções de geometria. Entretanto, essa proposta não é validada para diferentes tipos de aeronaves, nem mesmo é realizado um experimento real, apenas simulado.

Já em [11], foi apresentado um modelo matemático para ortorretificação e mosaicagem em tempo real de um fluxo de vídeo adquirido por um VANT pequeno de baixo custo. Testes de campo demonstrarão que a velocidade de processamento e a precisão alcançada atendem aos requisitos de tempo real da aeronave. Neste caso, diferentemente do trabalho aqui proposto, foi utilizado um veículo de asa fixa. Desse modo, a implementação e a validação do modelo matemático apresentado, de modo averiguar a sua viabilidade.

Em [12], uma abordagem de estimação de posição e atitude utilizando um receptor GPS, sensores inerciais e a correlação de imagens de características SIFT (*Self-Invariant Feature Transformation*) entre imagens sobrepostas foi apresentada. Além disso, a abordagem utilizada para mosaicagem possibilita um sensoriamento de baixo custo utilizando um VANT de pouco peso. De modo análogo as referencias apresentadas anteriormente, a aeronave utilizada é de asa fixa.

E em [13], foi proposto um método adaptativo para determinar a distância limiar do algoritmo SIFT utilizando a proporção de pontos otimizados de RANSAC. Esta abordagem tornou o algoritmo baseado em SIFT mais preciso e robusto. As imagens utilizadas também foram obtidas a partir de um VANT de asa fixa.

O diagnóstico rápido e preciso do problema pode ser a chave para a proposição de métodos eficazes de solução, e o mosaicamento e a ortorretificação são fortes aliados nesse sentido. Sendo assim, as principais contribuições desta pesquisa são:

1) apresentar uma abordagem de mosaicagem de tempo real;

2) permitir a efetiva comparação entre os diferentes tipos de descritores existentes na literatura utilizados nos algoritmos de mosaicagem;

3) possibilitar uma mosaicagem automática a partir de imagens de veículos aéreos não tripulados de asa rotativa.

Este artigo está organizado da seguinte forma: a seção II apresenta a plataforma experimental utilizada para obtenção do vídeo descrevendo seu funcionamento e os sensores embarcados que permitem a obtenção das informações necessárias para a confecção do mosaico; O tratamento das imagens é apresentado na seção III. São detalhados as técnicas utilizadas para que sejam feiras as correspondências das imagens (SIFT (*Scale Invariant Feature Transform*), SURF (*Speeded-Up Robust Features*) e ORB (*Oriented FAST and Rotated BRIEF*)), bem como o operador RANSAC (*Random Sample Consensus*) utilizado para realização da melhoria na definição dos dados de correspondências das imagens; A metodologia proposta para a geração de mosaicos em tempo real a partir de vídeos obtidos por meio de VANTs de asa rotativa é apresentada na seção IV; Os resultados obtidos e as discussões são apresentados na seção V; Por fim as conclusões e os trabalhos futuros são apresentados na seção VI.

II. Plataforma Utilizada - AR.Drone 2.0

A plataforma experimental escolhida para a desenvolvimento deste trabalho é o quadrotor da fabricante Parrot, modelo Ar.Drone 2.0[1]. O Parrot AR.Drone 2.0, Figura 1, é um veículo aéreo autônomo, acionado eletricamente, comercializado como brinquedo e destinado a jogos de realidade aumentada, sendo composto por uma estrutura de suporte de fibra de carbono, corpo de plástico, quatro motores *brushless* de alta eficiência, placa de controle, sensores e duas câmeras.

[1] http://ardrone2.parrot.com/

Figure 1. Quadrotor Parrot AR.Drone 2.0.

Originalmente o Parrot foi projeto para ser controlado por *tablets* e *smartphones* a partir de uma rede *Wi-Fi* disponível no equipamento. Dessa forma o operador pode definir os ângulos de guinada, arfagem, rolagem e as velocidades vertical e angular que o quadrotor deve desempenhar e a placa de controle ajustará automaticamente as potências dos motores para reposicionar a aeronave. O quadrotor pode alcançar velocidades superiores a 5 ms^{-1} e sua bateria fornece energia suficiente para um voo contínuo de 8 à 12 minutos, dependendo das condições em que o voo é realizado.

O computador da placa de controle da aeronave é baseado em um processador ARM9 de 468MHz com 128 MB de DDR RAM de 200 MHz. Entretanto, a aeronave vem sendo utilizada em pesquisas com VANTs[5]. Em [6], abordagens de aprendizado de máquinas foram aplicadas para predizer erros de posição de acompanhamento de trajetória de voo de um VANT. Outros trabalhos têm utilizado a aeronave como plataforma experimental para tarefas de vigilância autônoma [7], interação homem-máquina [8], e até como assistente de esporte [9].

O fabricante do quadrotor fornece aplicativo de interface com o usuário, aplicativo de simulação da aeronave, e a API (*Application Programming Interface*). O aplicativo de interface com o usuário que pode ser instalado em um *smartphone* ou *tablet* permite a comunicação com a aeronave em uma rede *Wi-fi* e a API permite definir o estado desejado da aeronave e fornece acesso as medições de sensores e às imagens das câmeras a bordo. Uma vez que esses *software* são fornecidos gratuitamente, o desenvolvimento de algoritmos e estratégias para o Ar.Drone 2.0 é facilitado.

A. Dados Sensoriais

O sensoriamento da aeronave consiste de uma unidade de medida inercial (IMU - *Inertial Measurement Unit*) composta por um giroscópio de três eixos, um acelerômetro de três eixos e um magnetômetro, um altímetro baseado em ultrassom e sensor de pressão do ar, duas câmeras (a primeira é montada para frente e fornece uma imagem colorida HD de 720 *pixels* (1280x720) a 30 fps, e a segunda é montada na parte inferior e fornece uma imagem colorida de 320 x 240 *pixels*, ambas com baixa latência de transmissão), um receptor GPS (*Global Position System*) que vem com uma unidade de armazenamento de 4GB, um computador de bordo, além da rede de comunicação sem fio. O receptor GPS permiti o rastreamento da aeronave e ao piloto definir uma trajetória de voo selecionando uma série de *waypoints* de interesse que a aeronave deverá seguir.

A placa de controle utiliza os sensores para auxiliar o usuário com manobras difíceis, como a decolagem e a aterrissagem, estabilizar o voo e bloquear imediatamente as hélices em caso de contato com um corpo estranho (para garantir a segurança do operador). O *firmware* desta placa também recebe e responde a comandos de movimento externos tais como altitude do veículo em relação ao solo, velocidades lineares desenvolvidas nos referencias x e y, bem como os ângulos de inclinação da aeronave (referenciados no sistema de coordenadas global) ϕ, θ *e* ψ.

B. Comunicação

A comunicação com a aeronave é feita por meio de uma conexão sem fio (padrão *Wi-fi*), providenciada pelo próprio dispositivo. Quatro canais de comunicação são utilizados, sendo que três deles operam via protocolo UDP (*User Datagram Protocol*), para agilizar a transmissão. Estes são responsáveis pela obtenção de dados navegacionais, tais como posição, velocidade e situação; pela transmissão de comandos de controle e pela obtenção de vídeo em tempo real (em uma taxa de até 30 quadros por segundo).

O quarto canal utiliza uma conexão TCP (*Transmission Control Protocol*) e é utilizado para transmissão de dados críticos, que necessitem de confirmação (como obtenção e transmissão de dados de configuração)[2].

Os canais de interesse no escopo deste trabalho são os canais responsáveis pela obtenção de vídeo em tempo real e de dados navegacionais. Para que as informações e vídeos possam ser obtidas o VANT foi conectado ao computador por meio da rede sem fio e do MATLAB®, mais especificamente foi utilizada e atualizada a biblioteca *Controller for AR.Drone*[3]. Os atributos que podem ser alterados e modificados são acessados a partir da porta de dados 5554 via UDP. A conexão é tratada como um arquivo, com isso, os dados podem ser lidos do mesmo. As informações mais relevantes são:

- Velocidade na horizontal ($X_Velocity$) – Indica a velocidade horizontal da aeronave em tempo real do voo;
- Velocidade na vertical ($Y_Velocity$) – Indica a velocidade vertical da aeronave em tempo real do voo:
- Altitude (*Altitude*) – Indica a altitude da aeronave em tempo real do voo.

A conexão com o Parrot se dá por meio de duas portas UDP (5556 - porta de configuração e controle e 5554 - porta que permite acesso aos dados do voo como altitude, velocidade, nível de bateria entre outros) por meio do IP (*Internet Protocol*) 192.168.1.1. Por meio dessa conexão é possível enviar comandos para

[2]De acordo com o Guia do Desenvolvedor AR.Drone, disponível no SDK 2.0.

[3]http://www.mathworks.com/matlabcentral/fileexchange/42532-controller-for-ar-drone

o VANT de modo que ele sobrevoe uma região de interesse para a obtenção da mosaicagem, e receber as imagens obtidas de suas câmeras a bordo, criando assim uma mosaicagem automática e em tempo real. Dentre os métodos utilizados para enviar os comandos de controle para o VANT estão: `takeoff()`; `land()`; `moveUp(speed)`; `moveDown(speed)`; `moveLeft(speed)`; `moveRight(speed)`; `moveForward(speed)`; `stop()`; `moveReverse(speed)`; `rotateLeft(omega)` e `rotateRight(omega)`.

Todos esses comandos usam como referência a câmera frontal da aeronave, por exemplo, o método `moveLeft(speed)` faz com que a aeronave se mova para a esquerda (com relação à câmera frontal) fazendo uso da velocidade informada.

Outras informações sobre o funcionamento, operação e outras características do Parrot AR.Drone 2.0 podem ser encontradas em [14], [15] e no site do fabricante já informado anteriormente.

III. Imagens - Retificação e Correspondência

Mosaico é um produto gerado a partir de técnicas de registro de imagens, sejam elas obtidas por meio de fotografias ou por meio de vídeos. Um mosaico consiste na composição de imagens adquiridas de diferentes pontos de vista com objetivo de obter uma área de cobertura maior, permitindo assim uma visão geral da cena.

A geração de mosaicos de imagens obtidas a partir de sensores remotos consiste em um conjunto de procedimentos como a retificação das imagens, a correspondência entre imagens (inicialmente o registro da imagem de referência com a imagem de ajuste é obtido e assim uma imagem que implica na combinação das duas é gerada) e a suavização das imagens. Um dos desafios está em se combinar com precisão, rapidez e robustez essas imagens.

Abordagens atuais para a correspondência de imagens podem divididas em duas classes: baseadas em características da imagem (como borda, canto, contorno, etc) e as baseadas em intensidade. A classe de algoritmos baseado em características requer baixa quantidade de computação, mas uma maior qualidade de imagens e capacidade de extração de características, sendo assim sensíveis a distúrbios como distorções geométricas e ruídos. A segunda classe faz uso da comparação de intensidade da região de exemplo, sendo muito precisa, mas por outro lado necessitando de um alto poder computacional, o que não é ideal para o cômputo em tempo real [10]. Quando o foco é a obtenção de mosaicos em tempo real, principalmente fazendo uso de plataformas com baixo poder computacional, baixa memória, como é o caso do VANT proposto neste trabalho, a segunda classe de técnica passa a ser inaceitável.

Desse modo, como obter correspondência de imagens assegurando boa precisão? Existem algumas técnicas que buscam melhorar esse desempenho como abordagens baseadas em transformadas *wavelets*, em projeções, SSDA (*Sequential Similarity Detection Algorithm*), em alterações de exemplos, e em correlação aos domínios de frequência [16] [17].

A. Retificação de Imagens

A retificação geométrica é aplicada em imagens de sensoriamento remoto para corrigir a distorção geométrica da imagem original e permitir que as imagens sejam corrigidas de acordo com um quadro de referência uniforme gerando uma nova imagem que satisfaça a exigência de um mapa de projeção. Quando um VANT está voando, alguns fenômenos como a inclinação e rotação podem ocorrer, principalmente pela existência de ventos e outros fatores, o que fará com que as imagens obtidas pela câmera apresentem distorção geométrica que afetam a qualidade da imagem. Para garantir que o mosaico seja realizado com sucesso, as imagens distorcidas devem ser corrigida geometricamente antes da geração do mosaico.

Formas de retificação mais eficazes são apresentadas na literatura [11], [4], entretanto, não serão abordadas nem discutidas neste trabalho uma vez que, considerando-se as condições reais de voo de um VANT de asa rotativa como o Parrot, a exigência de qualidade para a imagem mosaicada e o requisito de desempenho pelo fato da mosaicagem ser executada em tempo real, a retificação de imagem é executada apenas para a rotação horizontal do VANT, caracterizada pelo ângulo de *yaw*.As distorções geométricas causadas pelos ângulos *pitch* e *roll* não serão consideradas. Sendo assim, adotando-se a interpolação com o vizinho mais próximo é possível efetuar a retificação da imagem.

B. Correspondência de Imagens

Encontram-se na literatura diversas técnicas que permitem que a correspondência entre pontos de imagens possa ser realizada. Neste trabalho são utilizadas e comparadas três das técnicas existentes na literatura.

A transformada SIFT (*Scale Invariant Feature Transform*) é a técnica mais amplamente utilizada para a extração de características de imagens [18] [19] apesar do SURF (*Speeded-Up Robust Features*) ter se destacado em alguns trabalhos por apresentar menor custo computacional [20]. Em [21] é apresentada uma comparação entre as técnicas SIFT, SURF e PCA-SIFT (*Principal Components Analysis - Scale Invariant Feature Transform*), demonstrando que apesar de maior custo computacional, a transformada SIFT ainda apresenta melhor estabilidade quanto à invariância, à escala e rotação quando comparada às demais.

Entretanto em [22] uma nova abordagem, denominada ORB (*Oriented FAST and Rotated BRIEF*) é apresentada como alternativa mais eficiente que SIFT e SURF, apresentando esse novo descritor como uma boa abordagem principalmente para ambientes embarcados. Uma vez que a proposta desse paper, é a geração de mosaicos em tempo real, optou-se por avaliar essas três técnicas, permitindo assim que seja escolhida a melhor frente a análise dos resultados.

1) SIFT: O uso do SIFT permite que um conjunto de descritores de características seja extraído, e então possam ser combinadas. Dessa forma esse método permite que imagens sejam convertidas em uma coleção de vetores de

características, sendo esses invariantes à escala, rotação e mudanças de iluminação e ponto de vista. O algoritmo SIFT consiste de quatro estágios: Detecção de extremos no espaço-escala; Localização de pontos chaves; Definição da orientação e Descritor dos pontos chave.

O primeiro estágio faz uso da função de Diferenças de Gaussianas (DoG) para a detecção dos pontos de interesse, que são invariantes em escala e orientação. A DoG é utilizada em substituição a Gaussiana para melhorar o tempo de computação. Uma pirâmide de imagens filtradas pela DoG é formada e organizadas em oitavas. A DoG é a convolução da imagem em diferentes escalas de espaço as quais são separadas por um fator constante k, como ilustrado na Equação 1

$$D(x, y\sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \quad (1)$$

onde G é a função Gaussiana, σ é a variância da Gaussiana, I é a imagem de entrada e $*$ é a operação de convolução em x e y.

A oitava seguinte é formada por sub-amostradas da imagem original pelo fator de 2 e executada a função DoG. Os pontos de interesse detectados são selecionados como pontos candidatos somente se forem máximos ou mínimos após comparação com a vizinhança de 3x3 na escala atual e duas escalas adjacentes.

No próximo estágio, os pontos chaves candidatos são localizados e refinados pela eliminação dos pontos com baixo contraste. Eles são detectados e eliminados a partir da definição de um *threshold* da expansão de Taylor de $D(x, y\sigma)$. A matriz hessiana é usada para calcular as curvaturas principais e eliminar os pontos chave que têm uma razão entre a curvaturas principais superior ao *threshold*.

O terceiro estágio consiste na atribuição de orientações para cada ponto chave localizado, baseado em direções do gradiente. Um histograma de orientação é formado a partir das orientações de gradiente dos pontos amostrados dentro de uma região ao redor do ponto-chave a fim de obter uma atribuição de orientação. Um histograma de orientações contento 36 *bins* permite cobrir 360 graus de orientação. O pico em um histograma de orientações corresponde a direção dominando do gradiente local. Isso garante a invariância na rotação da imagem.

O quarto e último estágio do SIFT consiste em calcular o descritor para cada ponto baseado no gradiente de magnitude e orientação da imagem. Eles são calculados em um vetor de localização de 4×4 e 8 *bins* de orientação. Com isso tem-se um total de 128 elementos no vetor de características para cada ponto chave, que será então invariante com relação a escala, rotação, mudanças de iluminação, robusto quando a ruídos e diferenças de pontos de vista.

2) SURF: Inspirado em parte pelo descritor SIFT, SURF [23] foi apresentada pela primeira vez em 2006. Ele usa a imagem original para calcular o determinante aproximado do detector de Hessian eficientemente. Ele também adopta a imagem original para calcular a soma da resposta da *wavelet* de Haar em torno do ponto de interesse como descritor. A versão padrão do SURF segundo os autores é várias vezes mais rápida do que o SIFT.

O detector SURF é baseado no na matriz de Hesse, mas utiliza uma aproximação muito básica, tal como o DoG é um detector muito básico baseado em Laplace. Ele se baseia em imagens integrais para reduzir o tempo de computação e, portanto, é chamado de detector *Fast-Hessian*. O descritor do SURF, por outro lado, descreve uma distribuição de respostas *Haar-wavelet* dentro da vizinhança do ponto de interesse. Mais uma vez, as imagens integrais são exploradas para obtenção de velocidade no processamento. Além disso, apenas 64 dimensões são utilizadas, reduzindo o tempo de computação para a caracterização e correspondência dos pontos de interesse das imagens e, simultaneamente, aumentando a robustez.

O conceito de imagem original [24] utilizado pelo SURF permite a rápida implementação dos filtros de convolução do tipo *box*. A entrada de uma imagem original $I_{\sum}(\mathbf{x})$ em uma localização $\mathbf{x} = (x, y)$ representa a soma de todos os *pixels* em uma imagem I de entrada de uma região retangular formada pelos pontos $\mathbf{x}$ e a origem, $I_{\sum}(\mathbf{x}) = \sum_{i=0}^{i<x} \sum_{j=0}^{i<y} I(i, j))$. Com $I_{\sum}$ calculado, são simplesmente realizadas quatro adições para calcular a soma de todas as intensidades sobre qualquer área retangular vertical, independentemente do seu tamanho.

O detector do SURF é baseado na matriz Hessiana devido a seu bom desempenho computacional e a sua acurária. Entretanto, ao invés de usar uma medida diferente para selecionar a localização e a escala (como feito pelo detector Hessiano-Laplaciano), foi utilizado o determinante Hessiano para ambos. Dado o ponto $\mathbf{x} = (x, y)$ em uma imagem I, a matriz Hessiana $H(x, \sigma)$ em $\mathbf{x}$ com escala σ é definida por:

$$H(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \quad (2)$$

onde $L_{xx}(x, \sigma)$ é a convolução da derivada de segunda ordem da Gaussiana $\frac{\partial^2}{\partial x^2} g(\sigma)$ com a imagem I no ponto $\mathbf{x}$, e similaridade $L_{xy}(x, \sigma)$ e $L_{yy}(x, \sigma)$

Gaussianas são adequadas para a análise do espaço-escala, entretanto, na prática Gaussianas precisam ser discretizadas e cortadas, e mesmo com filtros de Gauss ainda assim as imagens resultantes são sub-amostradas. Além disso, a propriedade que permite que estruturas não novas podem aparecer a medida que se baixa a resolução da imagem (como é o caso de imagens de 1 dimensão), mas que não é relevante para o caso de imagens com resolução maior (caso de imagens de 2 dimensões), parecem ter sobrestimado a importância da Gaussiana. Como os filtros de Gauss são não-ideal, em qualquer caso, e dado o sucesso de Lowe com aproximações LoG, o SURF emprega aproximação ainda mais com filtros *box*. Essas aproximam as derivadas de segunda ordem da Gaussiana e podem ser avaliadas mais rapidamente fazendo uso de imagens originais, independentemente do seu tamanh. Os resultados em termo de desempenho do seu uso é comparado ao que faz uso da Gaussina discretizada e cortada.

Desse modo, o SURF apresenta um melhor desempenho quando comparado ao SIFT.

3) ORB: Com foco em sistemas que necessitam de descritores e detectores de baixo custo e bom desempenho foi proposto o ORB [22]. O ORB foi construído com base no detector FAST [25] e no descritor BRIEF [26].

FAST e suas variantes são famosos por sua eficiência em encontrar pontos chaves razoáveis, e tem sido popular em muitos sistemas de tempo real. O BRIEF, por outro lado, usa apenas um parâmetro, o limiar de intensidade entre o pixel central e seus pixels vizinhos em um anel circular. Uma das principais limitações da FAST é que ele não inclui um operador de orientação - por exemplo, os histogramas de gradiente presente no SIFT e no SURF. Outra limitação é que a FAST dá grandes respostas ao longo das bordas.

BRIEF é um descritor que faz uso de testes binários simples entre *pixels* em uma imagem plana. Seu desempenho é similar ao SIFT em vários aspectos, incluindo robustez quando a luminosidade, manhas e distorções de perspectivas. Entretanto, ele é sensível a rotação no plano.

Para lidar com o problema de rotação o ORB faz uso do centróide de intensidade para calcular uma orientação para cada canto do FAST e então adota a resposta do canto de Harris para remover os pontos ao longo das bordas. Além disso, uma vez que os cantos do FAST não produzem recursos multi-escala, são extraídos características do ORB em cinco imagens escaladas separadamente, com um fator de escala de $\sqrt{2}$.

O centróide de intensidade proposto no ORB assume que a intensidade do canto está deslocada do seu centro, e este vetor pode ser utilizado para imputar uma orientação. Sendo assim, em [27] os momentos um recorte como:

$$m_{pq} = \sum_{x,y} x_p y_q I(x,y), \tag{3}$$

e com esses momentos é possível encontrar o centróide:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{4}$$

Então é possível construir o vetor a partir do centro do canto, O, para o centróide $\overrightarrow{OC}$. A orientação para o recorte é simplesmente:

$$\theta = atan2(m_{01}, m_{10}), \tag{5}$$

onde $atan2$ é a versão do quadrante-consciente do $arctan$ de R.

Para melhorar a invariância desta medida foi certificado que os momentos são calculados com x e y restante dentro de uma região circular de raio r. O tamanho de r foi escolhido aleatoriamente para definir o tamanho do recorte de modo que x e y sejam parte de $[-r, r]$. Como $|C|$ tende a 0, a medida torna-se instável; com cantos oriundo do FAST, isso acontece raramente. Desse modo o centróide fornece uma orientação uniforme, mesmo que em imagens com muitos ruídos.

Sendo assim, a extração das características com o ORB é mais rápida que o SURF e o SIFT segundo os autores, sendo que memos pode ser utilizado em dispositivos embarcados, como por exemplo *smartphones*.

C. Operador RANSAC

Após a aplicação da abordagem de correspondência de imagens, há ainda um certo número de pontos encontrados indicarem falsas correspondências, de modo que faz-se necessário identificá-las e removê-las. Por meio da homografia [28] e da geometria epipolar [29] é possível remover essas falsas correspondências e assim juntar duas imagens sequenciais. Nesse artigo é utilizado o operador RANSAC (*Random Sample Consensus*) [30] para resolver esse problema, ou seja, filtrar os falsos pontos correspondentes entre pares de imagens.

O RANSAC é um método de estimativa de parâmetros robusto, e a idéia básica de funcionamento é: em primeiro lugar, criar uma determinada função alvo de acordo com problemas específicos; segundo, extrair pontos mínimos repetidamente para estimar o valor inicial dos parâmetros da função alvo e, em seguida, usa-se o valor inicial para dividir todos os dados em *inlier* e *outlier* (ou falsas correspondências); por fim, recalcula-se e re-estima-se os parâmetros da função de todos os dados *inlier*.

Sendo assim, o RANSAC tenta encontrar os *inliers* estimando a matriz h e ignorando os *outliers*. Ele aleatoriamente amostras pontos correspondentes e tenta encaixar a matrix homográfica H nas amostras de dados correspondentes. Em seguida, um erro é calculado entre o resto de amostras de dados e o modelo. Os pontos de dados são classificados como *inlier* ou *outliers* com base em um *threshold* T. O processo é continuado até que o número de *outliers* seja suficientemente pequeno.

O RANSAC método deve ser capaz de distinguir o quão bem a matriz H estimada encaixa todas as amostras de dados. Para que isso seja realizado de forma correta, o erro de transferência simétrica é utilizado como a medida de distância (Equação 6).

$$E(z_i, z_i^{'}) = d(z_i, H^{-1} z_i^{'})^2 + d(z_i^{'}, H z_i)^2 \tag{6}$$

onde H é a mtriz homográfica estimada, $d(x_1, y_2)$ é a Distância Euclidiana entre as amostras x_1 e x_2, z_i e $z_i^{'}$ as coordenadas dos pontos correspondentes. Se $E(z_i, z_i^{'}) < T$, o par correspondente z_i e $z_i^{'}$ são considerados *inliers*.

Considerando $z = [x_i, y_i]^T$ as coordenadas dos pontos do quadro a, e $z^{'} = [x_i^{'}, y_i^{'}]^T$ as coordenados do ponto correspondentes no próximo quadro b, a matriz de transformação H entre os dois quadros é dada por $z \approx Hz^{'}$. Deve-se observar que para que H possa ser estimada são necessários pelo menos quatro pontos de correspondência.

Algoritmos de Transformação Linear Direta são métodos bem conhecidos para se encontrar a matriz homográfica H. Sendo assim, uma vez que tem-se quatro pontos

de correspondência entre duas imagens $z = [x_i, y_i]^T$ e $z^{'} = [x_i^{'}, y_i^{'}]^T$, e $i \in \{1, ..., 4\}$, o objetivo é encontrar a matriz H de modo que $z_i^{'} \approx Hz_i$ para toros os pontos. Hz_i pode ser obtido a partir da Equação 7.

$$Hz_i = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ v_i \end{bmatrix} \tag{7}$$

$$= \begin{bmatrix} h_{11}x_i + h_{12}y_i + h_{13}v_i \\ h_{21}x_i + h_{22}y_i + h_{23}v_i \\ h_{31}x_i + h_{32}y_i + h_{33}v_i \end{bmatrix} = \begin{bmatrix} h_1^T z_i \\ h_2^T z_i \\ h_3^T z_i \end{bmatrix}$$

Em adição, $z_i^{'} \times Hz_i = 0$ pode ser calculado como:

$$z_i^{'} \times Hz_i = \begin{bmatrix} y_i^{'} h_3^T z_i - v_i^{'} h_2^T z_i \\ v_i^{'} h_1^T z_i - x_i^{'} h_3^T z_i \\ x_i^{'} h_2^T z_i - y_i^{'} h_1^T z_i \end{bmatrix} = 0 \tag{8}$$

$$\begin{bmatrix} y_i^{'} z_i^T h_3 - v_i^{'} z_i^T h_2 \\ v_i^{'} z_i^T h_1 - x_i^{'} z_i^T h_3 \\ x_i^{'} z_i^T h_2 - y_i^{'} z_i^T h_1 \end{bmatrix} = 0$$

Desse modo, a Equação 8 pode ser vista como um sistema linear (Equação 9 e 9.1).

$$\begin{bmatrix} 0 & -v_i^{'} z_i^T & y_i^{'} z_i^T \\ v_i^{'} z_i^T & 0 & -x_i^{'} z_i^T \\ -y_i^{'} z_i^T & x_i^{'} z_i^T & 0 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \tag{9}$$

Na maioria dos casos, mais de quatro correspondentes podem ser obtidos. As posições desses pontos não são absolutamente precisas, então, $h = 0$ é a única solução para a equação 8. O problema é então reformulado com a minimização de $||Ah||$ para $||h|| = 1$. Assim, o menor vetor de *eigenvector* de $A^T A$ é a solução para esse problema de otimização. As coordenadas de transformação são usadas para normalizar um conjunto de pontos correspondentes em um novo quadro, denominado como T e $T^{'}$. Adicionalmente, os pontos coordenados normalizados podem ser expressos como $\tilde{z_i} = Tz_i$ e $\tilde{z_i^{'}} = T^{'} z_i^{'}$.

A matriz homográfica H para os pontos não normalizados é então definida como $H = T^{'-1} \tilde{H} T$, onde $\tilde{H}$ é a matriz homográfica normalizada que diz respeito aos pontos normalizados $\tilde{z_i^{'}} = \tilde{H} \tilde{x_i}$.

Para calcular T e $T^{'}$, cada conjunto de coordenadas é traduzido de modo que o centróide de cada conjunto esteja localizado na origem. Assim, cada conjunto é escalado de modo que a distância média dos pontos de origem seja $\sqrt{2}$. A matriz normalizada T para z_i é vista na Equação 10:

$$T = \begin{bmatrix} \frac{\sqrt{2}}{\mu_d} & 0 & -\frac{\sqrt{2}}{\mu_d}\mu_x \\ 0 & \frac{\sqrt{2}}{\mu_d} & -\frac{\sqrt{2}}{\mu_d}\mu_y \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

onde μ_x e μ_y são as coordenadas x e y da média, e μ_d é a distância média das coordenadas de origem. De modo similar, a matriz normalizada $T^{'}$ pode ser calculada.

Homografia é uma transformação geométrica entre duas imagens expressa em uma matrix H 3 × 3. Essa matriz é calculada matematicamente a partir do conjunto de pontos de correspondência obtidos do algoritmo SIFT, conforme Equação 11.

$$X^{'} = H \times X \tag{11}$$

onde X e $X^{'}$ são as coordenadas dos pontos de correspondência nas duas imagens e a matriz H representa a transformação homográfica com oito graus de liberdade.

Desse modo, o RANSAC, diferentemente da maioria dos algoritmos clássicos, utiliza um conjunto mínimo de correspondências amostradas para estimar os parâmetros de transformação da imagem e achar a solução que tem o melhor consenso com os dados.

Uma vez estimada a matriz homográfica H, um novo quadro de imagem é alinhado ao quadro de fundo para a construção do mosaico.

IV. Mosaicagem de Imagens

Esse artigo apresenta um processo automático e em tempo real de geração de mosaicos a partir de vídeos obtidos de VANTs de asa rotativa. O processo de mosaicagem a partir de um *stream* de vídeo é baseado em 2 etapas (ilustrados na Figura 2:

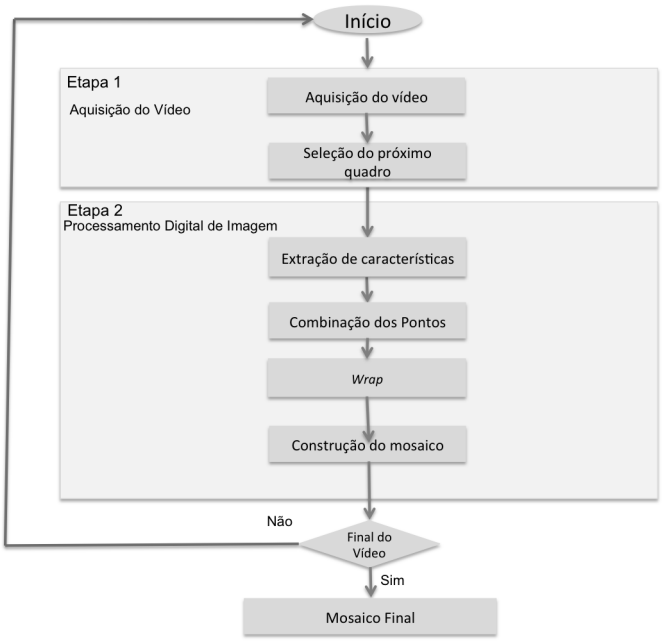

Figure 2. Metodologia utilizada para a mosaicagem em tempo real.

lembrando que todo o processo ocorre durante o voo, em tempo real.

$$\begin{bmatrix} 0 & 0 & 0 & -v_i^{'}x_i & -v_i^{'}y_i & -v_i^{'}v_i & y_i^{'}x_i & y_i^{'}y_i & y_i^{'}v_i \\ v_i^{'}x_i & v_i^{'}y_i & v_i^{'}v_i & 0 & 0 & 0 & -x_i^{'}x_i & -x_i^{'}y_i & -x_i^{'}v_i \\ -y_i^{'}x_i & -y_i^{'}y_i & -y_i^{'}v_i & x_i^{'}x_i & x_i^{'}y_i & x_i^{'}v_i & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = A_i h = 0 \qquad (9.1)$$

A. Aquisição do vídeo

A primeira etapa consiste na obtenção do vídeo (a partir da câmera a bordo da aeronave).

Normalmente o uso de imagens de vídeo para geração de imagens é realizado por meio de um pré-processamento que efetua amostragem dos quadros do vídeo a partir de um determinado intervalo de tempo fixo, reduzindo assim o desperdício de dados.

Para VANTs de pequeno porte, principalmente os de asa fixa, é muito difícil conseguir a estabilidade tanto da atitude quanto da velocidade de voo, e por isso algumas técnicas fazem uso de intervalos de tempos variados para a obtenção dos quadros e a geração de mosaicos pós-processados, permitindo assim uma melhor sobreposição das imagens [31]. Entretanto, para esse trabalho, fez-se uso da obtenção de quadros em tempos fixos uma vez que a aeronave utilizada (VANT de asa rotativa) é considerada estável, permitindo que tanto a altitude quanto a velocidade sejam preservadas durante o voo. Considerou-se também condições apropriadas de voo sem a presença de ventos ou condições adversas do tempo.

A taxa de amostragem de quadros do vídeo foi fixada em 1 quadro a cada 4 segundos levando-se em consideração que a taxa de captura do vídeo de entrada é de aproximadamente 30 quadros por segundo (*frames per second* - fps), sendo que a sobrecarga de pré-processamento para a obtenção de quadros a partir de valores variáveis é eliminada.

O tamanho do vídeo é determinado pelo tamanho da missão executada. O tempo máximo de voo da aeronave é de 12 minutos. Para o processo de mosaicagem foi definido que o início do vídeo se dá após a decolagem e início efetivo da missão, o que geralmente ocorre um minuto após a sua decolagem. De modo análogo, o final da mosaicagem quando da finalização da missão, o que não inclui o pouso da aeronave. Desse modo o processo de mosaicagem ocorre em no máximo 8 minutos de *stream* de vídeo.

B. Processamento digital de imagem

A segunda etapa consiste do processamento digital de imagem, que permite que sejam realizadas: Extração de Características (utilização de detecção de *features* para extração dos pontos chaves da imagem); Combinação dos pontos (combinação dos pontos de interesse (RANSAC)); *Wrap* (a matriz de homografia H é estimada para transformar novos quadros de coordenadas de fundo ajustando perspectiva/distorção); e Composição do Mosaico.

Como as imagens são obtidas sequencialmente, o primeiro quadro é obtido a partir do início da captação do vídeo. O quadros são combinados aos pares, sendo que a primeira imagem e a segunda são comparadas e a junção é efetuada em um terceira imagem de forma sucessiva até o final do vídeo.

Uma vez que os quadros do vídeo são sequenciais, na extração de características é possível fazer uso dos algoritmos SIFT, SURF e ORB (já discutidos na seção III-B) para que sejam extraídas as características do par de imagens, criados os descritores e criadas as correspondências dessas características.

De posse dos vetores de características e dos correspondentes, é então efetuada a Combinação dos pontos a partir da aplicação do RANSAC de modo a estimar a matriz homográfica e então remover os falsos pontos correspondentes.

O *Wrap* então é realizando permitindo que as imagens seja unidas de modo que a perspectiva e distorções sejam ajustadas gerando assim as fases intermediárias para a composição do mosaico final (que podem ser observados durante todo o processo).

Esse processo se repete enquanto não ocorrer o final do vídeo. Uma vez terminado o vídeo o mosaico final fica disponível.

V. Resultados e Discussões

A obtenção dos mosaicos foram efetuadas em tempo real e os resultados do vídeo captura foi processado em um *Notebook Inspiron* 15R-5537-A10 com Intel Core 4 i7 8 GB 1 TB de disco e LED 15,6 W8 da fabricante Dell®.

Na Figura 3 são ilustrados os mosaicos obtidos a partir da execução do algoritmo fazendo uso dos descritores SIFT, SURF e ORB. Em todas as execuções foi utilizado o RANSAC. Para que a análise e comparação dos resultados pudesse ser justa foram definidos como número máximo de *feature points* 1000. Desse modo, todos os métodos rodam com a detecção de até 1000 *feature points*.

Os descritores SIFT, SURF e ORB utilizaram, mais uma vez para que a análise comparativa pudesse ser justa, como *feature matching* o algoritmo de força bruta (*Brute-Force*) diferentemente do FLANN (*Fast Library for Approximate Nearest Neighbors*) normalmente utilizado no SIFT, mas que consome muito tempo uma vez que faz uso de algoritmos de aprendizagem [32].

Sendo assim, nesse artigo ao invés de usar os recursos SIFT-like, o algoritmo de força bruta que faz uso de

recursos binários para correspondência característica e que apresenta alta velocidade e propõe um filtro coerente e eficaz permitindo manter o equilíbrio entre robustez e tempo real, capacidades necessárias para o sistema aqui proposto de mosaicagem de vídeo em tempo real.

É possível observar que os mosaicos gerados com o uso dos diferentes descritores apresentam resultados visualmente similares. Todos os mosaicos gerados atendem as especificações de qualidade, gerando imagens sem ruídos e com distorções aceitáveis. O fator mais importante é o tempo para a geração do mosaico, uma vez que a proposta é a de obtenção de mosaicos em tempo real para que se possa realizar observações quando da realização do voo do VANT.

Fica evidente, por meio da observação da Tabela I, que o SIFT fazendo uso do algoritmo de *feature matching* apresenta resultados bastante interessantes e apropriados para a realização de mosaicos em tempo real com uma qualidade acima das apresentada pelos outros descritores.

Na Figura 4 são ilustrados os pontos relevantes que constituem os *matching points* em cada um dos descritores avaliados. É possível observar que o SIFT, como já definido na literatura, apresenta uma maior quantidade desses pontos, permitindo que as correspondências sejam mais acuradas e com isso sejam obtidos mosaicos cuja qualidade é superior a apresentada pelos demais descritores.

Por meio dos testes aqui realizados é possível observar que o SIFT, fazendo uso do força bruta como *feature matching* é um candidato apropriado para a realização de mosaicos em tempo real a partir de vídeos obtidos por plataformas aéreas de asa rotativa.

Na Tabela I são apresentados: tempo de processamento, porcentagem de *inliers* por *features* válidas, tempo médio para cômputo de cada imagem, e quantidade de imagens descartadas para cada um dos descritores em um total de 118 imagens obtidas a partir de um voo de 8 minutos realizado com o AR.Drone Parrot. A taxa de amostragem das imagens utilizadas para a realização da mosaicagem em tempo real foi de 1 quadro a cada 4 segundos.

É possível observar que o número de correspondentes no SIFT foi igual do do SURF, provendo uma boa identificação dos pontos chaves. Já no ORB essa porcentagem foi menor, reafirmando o que pode ser observado na Figura 4. Por outro lado, o SURF não apresenta nenhum descarte de imagens, o que faz com que o mosaico gerado seja muito bom. Entretanto, o seu tempo de execução, quando comparado ao SIFT (neste caso especificamente ambos rodando com o algoritmo força bruta como mencionado anteriormente.

O ORB apresentou tempo intermediário de execução da mosaicagem e um número intermediário também de imagens descartadas.

A relação entre tempo e qualidade de mosaicamente permite afirmar que o SIFT fazendo uso do algoritmo de força bruta constitui alternativa interessante para a obtenção de vídeo em tempo real. Permite também que a taxa de amostragem de imagens seja aumentada, o que certamente permitirá que a taxa de imagens descartadas tenha menor implicação no mosaico final resultante.

VI. Conclusões

Para imagens aéreas captadas por uma câmera apontando para baixo, as alterações predominantes entre dois quadros consecutivos são translação e rotação, e as mudanças na escala e perspectiva são geralmente pequenas. Como resultado, não é necessário selecionar o SIFT fazendo uso do FLANN, que envolve custo significativo de computação em lidar com as transformações de imagem grande seja em escala e/ou perspectiva. Desse modo, foi utilizado como algoritmos para *features matching* o força bruta, que é computacionalmente mais leve e permite que sejam exploradas as características importantes do SIFT.

O SURF, por sua vez, apresentou resultados bastante interessantes também, apesar de qualidade do mosaico ser inferir ao gerado pelo SIFT. O SURF pode ser também uma alternativa para o uso em sistemas embarcados uma vez que o tempo de computação o mesmo é inferior ao SIFT. Entretanto, se o que se deseja é uma acurácia melhor a relação custo/desempenho do SIFT é melhor.

O ORB apresentou o pior resultado dentre os descritores apresentados. Novos estudos fazendo uso de diferentes *features matching* faz-se importante para todos os descritores, mas é de maior importância para o ORB, de modo a averiguar se é possível melhorar esse algoritmo sem que se perca tanto na qualidade dos mosaicos gerados.

Sendo assim, esse artigo apresentou uma mosaicagem em tempo real para um VANT de asa rotativa onde foram avaliados três métodos para extração de características das imagens, sendo que todas elas apresentam como resultado final um mosaico visualmente aceitável. Entretanto, a mosaicagem realizada com o uso de SIFT com o algoritmo de força bruta é o que apresentou um tempo de processamento mais realista frente a qualidade do mosaico apresentado, permitindo assim a realização da mosaicagem em tempo real.

Como trabalhos futuros pretende-se ampliar a avaliação aqui iniciada, permitindo que sejam avaliados outros descritores como FERNS e FAST, além dos algoritmos de *feature matching* como KD Tree, BBF, FLANN e BOW, e os algoritmos de estimação robusta como BaySAC, MSAC, LMedS, entre outros em substituição aos que foram apresentados aqui neste artigo.

Com essa avaliação completa será possível indicar quais são as melhores combinações a serem utilizados nos diferentes propósitos de geração de mosaicos. De posse da avaliação completa será possível extender a proposta para VANTs de asa fixa, que apresentam um grau de instabilidade de obtenção de imagens maior que o dos VANTs de asa rotativas.

Agradecimentos

Os autores gostariam de agradecer à CAPEs, CNPq e à FAPESP (processos número 2014/00694-5 e 2014/12134-4) pelo suporte financeiro dado a este projeto.

Figure 3. a. Mosaico gerado com SIFT; b. Mosaico gerado com SURF; c. Mosaico gerado com ORB

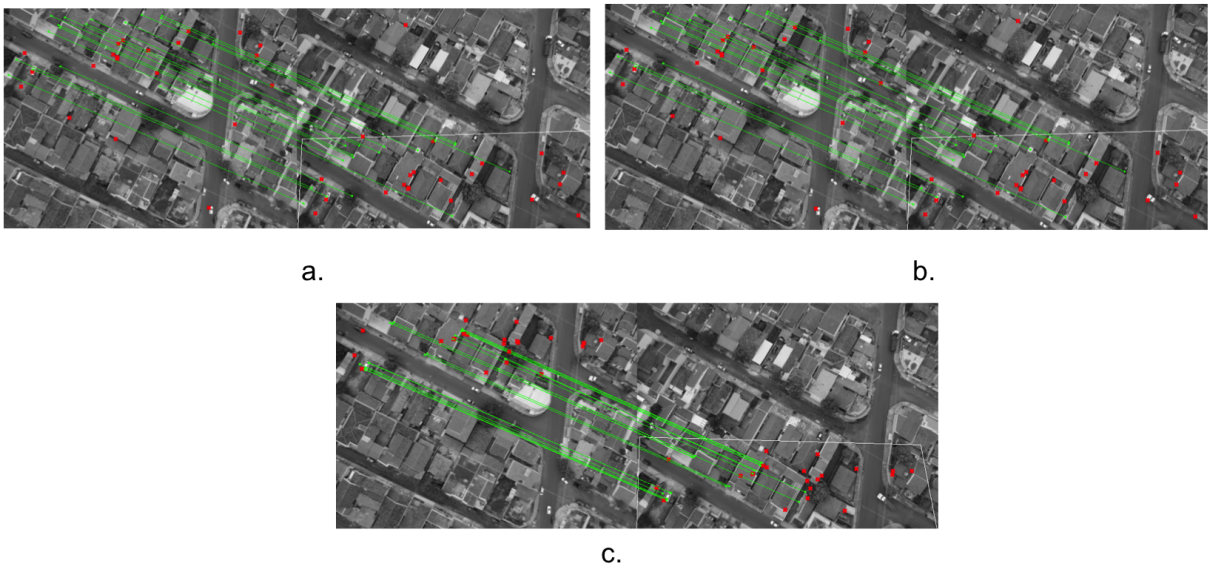

Figure 4. a. *Matching points* no SIFT; b. *Matching points* no SURF; c. *Matching points* no ORB

Table I. Comparação entre os descritores SIFT, SURF e ORB.

Descritor	**Tempo total (s)**	***Inliers/Features* Válidas (%)**	**Tempo Médio por Imagem (s)**	**Erros de *Matching* (qtd de imagens descartadas)**
SIFT	125	53	1s	10
SURF	427	53	3,6	0
ORB	334	47	2,8	4

References

[1] K. Valavanis and M. Kontitsis, "A historical perspective on unmanned aerial vehicles," in *Advances in Unmanned Aerial Vehicles*, ser. Intelligent Systems, Control and Automation: Science and Engineering, K. Valavanis, Ed. Springer Netherlands, 2007, vol. 33, pp. 15–46.

[2] O. Trindade, L. O. Neris., L. P. Barbosa, and K. R. L. J. C. Branco, "A layered approach to design autopilots," in *IEEE International Conference on Industrial Technology (ICIT)*, Vina del Mar - Valparaiso, Chile, 2010.

[3] L. Valentine, B. Agnes, and M. Jean-François, "Sensitivity of airborne-derived crop stress indices to the agricultural pratices," in *International Conference on Agricultural Engineering*, Crete, Greece, 2008.

[4] F. Neitzel and J. Klonowski, "Mobile 3d mapping with a low-cost UAV system," in *International Conference on Unmanned Aerial Vehicle in Geomatics*, Zurich, Switzerland, 2011.

[5] C. Bills, J. Chen, and A. Saxena, "Autonomous MAV flight in indoor environments using single image perspective cues," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, Shanghai, China, 2011, pp. 5776–5783.

[6] C. Bills and J. Yosinski, "MAV stabilization using machine learning and onboard sensors," Cornell University, Tech. Rep., 2010.

[7] J. Faigl, T. Krajník, V. Vonásek, and L. Přeučil, "Surveillance planning with localization uncertainty for UAVs," in *3rd Israeli Conference on Robotics*. Ariel: Ariel University Center, 2010, pp. –. [Online]. Available: http://www.icr2010.org.il/

[8] W. S. Ng and E. Sharlin, "Collocated interaction with flying robots," in *IEEE International Symposium on Robot and Human Interactive Communication*, Atlanta, GA, USA, 2011.

[9] K. Higuchi, T. Shimada, and J. Rekimoto, "Flying sports assistant: External visual imagery representation for sports training," in *Proceedings of the 2Nd Augmented Human International Conference*, New York, USA, 2011. [Online]. Available: http://doi.acm.org/10.1145/1959826.1959833

[10] H. Xiaowei, Z. Hongying, L. Yan, and Y. Shaowen, "An approach of fast mosaic for serial remote sensing images from uav," in *International Conference on Fuzzy Systems and Knowledge Discovery*, Washington, DC, USA, 2007. [Online]. Available: http://dx.doi.org/10.1109/FSKD.2007.144

[11] G. Zhou, "Near real-time orthorectification and mosaic of small UAV video flow for time-critical event response," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 3, pp. 739–747, 2009.

[12] T. Suzuki, Y. Amano, and T. Hashizume, "Vision based localization of a small UAV for generating a large mosaic image," in *SICE Annual Conference*, Taipei, Taiwan, 2010.

[13] S. Sun and Z. Zeng, "UAV image mosaic based on adaptive SIFT algorithm," in *International Conference on Geoinformatics*, Kaifeng, China, 2013.

[14] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "Ar-drone as a platform for robotic research and education," in *Research and Education in Robotics: EUROBOT 2011*. Heidelberg: Springer, 2011.

[15] P.-J. Bristeau, F. Callou, D. Vissire, and N. Petit, "The navigation and control technology inside the AR.Drone micro UAV," in *18th IFAC World Congress*, Milano, Italy, 2011.

[16] L. Tao and G. Xu, "Color in machine vision and its application," *Chinese Science Bulletin*, vol. 46, no. 17, pp. 1411–1421, 2001.

[17] R. Machuca and K. Phillips, "Applications of vector fields to image processing," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 3, pp. 316–329, 1983.

[18] L. Bei and Z. Haizhen, "An algorithm of fabric image mosaic based on sift feature matching," in *2009 International Conference on Artificial Intelligence and Computational Intelligence*. IEEE, 2009, pp. 435–438.

[19] N. Geng *et al.*, "Algorithm for sequence image automatic mosaic based on sift feature," in *Information Engineering (ICIE), 2010 WASE International Conference on*, vol. 1. IEEE, 2010, pp. 203–206.

[20] J. Hong, W. Lin, H. Zhang, and L. Li, "Image mosaic based on surf feature matching," in *Information Science and Engineering (ICISE), 2009 1st International Conference on*. IEEE, 2009, pp. 1287–1290.

[21] L. Juan and O. Gwun, "A comparison of sift, pca-sift and surf," *International Journal of Image Processing (IJIP)*, vol. 3, no. 4, pp. 143–152, 2009.

[22] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2564–2571.

[23] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*. Springer, 2006, pp. 404–417.

[24] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

[25] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 430–443.

[26] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," *Computer Vision–ECCV 2010*, pp. 778–792, 2010.

[27] P. L. Rosin, "Measuring corner properties," *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 291–307, 1999.

[28] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[29] D. Oram, "Rectification for any epipolar geometry." in *BMVC*, vol. 1, 2001, pp. 653–662.

[30] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[31] D. Wei and G. Zhou, "Real-time uav ortho video generation," in *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International*, vol. 5. IEEE, 2008, pp. V–510.

[32] J. Li, T. Yang, J. Yu, Z. Lu, P. Lu, X. Jia, and W. Chen, "Fast aerial video stitching," *International Journal of Advanced Robotic Systems*, vol. 11, 2014.