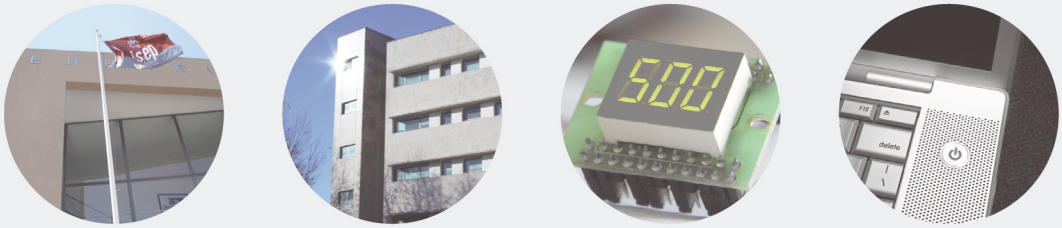
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Sistema de Apoio ao Processo de Aterragem Autónoma de um VTOL

MIGUEL MORIM MOREIRA
outubro de 2017

POLITÉCNICO
DO PORTO

Sistema de Apoio ao Processo de Aterragem Autónoma de um VTOL

Miguel Morim Moreira
Nº 1091286

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Sistemas Autónomos
Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

2017

Dissertação, para satisfação parcial dos requisitos do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Miguel Morim Moreira

Nº 1091286

Orientador: André Miguel Pinheiro Dias

Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização de Sistemas Autónomos

Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

16 de Outubro de 2017

Agradecimentos

A realização desta dissertação não seria possível sem o apoio e contributo de várias pessoas. A todos deixo o meu profundo agradecimento.

Em primeiro lugar, queria agradecer ao meu orientador, Professor André Dias, pelo suporte e todas as oportunidades proporcionadas.

À *Aerial Team*, Amaral, André, Fábio e Miranda. A vossa ajuda e apoio, foram fundamentais.

A todos os membros do Laboratório de Sistemas Autónomos (LSA) por todo o apoio e ajuda.

À Cris, por tudo.

Por fim, aos meus pais, por todo o esforço e sacrifício para que tivesse a formação académica que não lhes foi proporcionada.

Esta página foi intencionalmente deixada em branco.

Resumo

A utilização de veículos aéreos não tripulados é recorrente nos dias de hoje em missões de vigilância, inspeção ou apoio a equipas de socorro em missões de busca e salvamento. De forma a minimizar o erro humano e os custos de operação, estes sistemas são operados em modo autónomo, incluindo a fase da aterragem. A manobra de aterragem necessita de ser desempenhada com o menor erro de posição possível, pelo que as soluções clássicas através de recetores Global Navigation Satellite System (GNSS) em modo *single*, podem não ser suficientemente precisas.

Na dissertação propõe-se endereçar o desenvolvimento de um método de localização relativa que permita a aterragem autónoma de um Vertical Take-Off and Landing (VTOL) numa pista de aterragem móvel. Pretende-se a integração de uma *baseline* móvel Real Time Kinematic (RTK) entre a pista de aterragem e o VTOL, e combinar a *baseline* com a estimação de posição e atitude dada pelo sistema de visão a bordo.

O método desenvolvido de localização através de visão computacional consiste na deteção de um marcador visual ativo modelado para disparar sincronizado com a aquisição de imagens.

A qualidade do posicionamento e atitude fornecidos pela visão computacional em comparação com o sistema RTK e informação inercial, que serviram como *ground-truth*, foi aferida com recurso a testes experimentais.

Palavras-Chave: Aterragem Autónoma, Posicionamento Relativo, UAV, GNSS, RTK, Visão Computacional, IMU, Sincronismo

Esta página foi intencionalmente deixada em branco.

Abstract

The use of unmanned aerial vehicles is recurrent today in missions of surveillance, inspection or support to rescue teams in search and rescue missions. In order to minimize human error and operating costs, these systems are operated in an autonomous mode, including the landing maneuver. The landing maneuver needs to be performed with as little position error as possible, so classical solutions through single-mode GNSS receivers may not be sufficiently accurate.

The present document proposes to address the development of a relative localization method allowing the autonomous landing of a VTOL on a mobile landing strip. The intention is to integrate an RTK mobile baseline between the landing pad and the VTOL, and combine the baseline with the position and attitude estimation given by the on-board vision system.

The developed method of location through computer vision consists of the detection of an active visual marker, which is modelled to trigger synchronized with the acquisition of images.

The quality of the positionion and attitude provided by the computer vision in comparison with the RTK system and inertial information, which served as ground-truth, was measured using experimental tests.

Keywords: Autonomous Landing, Relative Postioning, UAV, GNSS, RTK, Computer Vision, IMU, synchronism

Esta página foi intencionalmente deixada em branco.

Conteúdo

Lista de Figuras

Esta página foi intencionalmente deixada em branco.

Lista de Tabelas

Esta página foi intencionalmente deixada em branco.

Lista de Siglas e Acrónimos

ASV Autonomouns Surface Vehicle

CCD Charge-Coupled Device

EKF Extended Kalman Filter

GNSS Global Navigation Satellite System

GPS Global Positioning System

IMU Inertial Measurement Unit

IIQ Intervalo Interquartil

LED Light Emitting Diode

LiDAR Light Detection And Ranging

LiPo Lithium Polymer

NTP Network Time Protocol

NTRIP Networked Transport of RTCM via Internet Protocol

ORB Oriented FAST and Rotated BRIEF

PPS Pulse-Per-Second

PTAM Parallel Tracking and Mapping

PTP Precision Time Protocol

RANSAC RANdom SAmple Consensus

RTC Real-time clock

RTK Real Time Kinematic

SBAS Statellite Based Augmentation Systems

SLAM Simultaneous Localization and Mapping

TCP Transmission Control Protocol

VTOL Vertical Take-Off and Landing

UAV Unmanned Aerial Vehicle

UGV Unmanned Ground Vehicle

UHF Ultra High Frequency

Capítulo 1

Introdução

Nos últimos anos, os veículos aéreos em particular os VTOL têm assumido um papel predominante em diversas áreas, desde a agricultura[8], vigilância[9], operações de busca e salvamento[10], inspeção de ativos elétricos[11], no apoio às equipas de bombeiros/-proteção civil em situações de incêndios[12] e em catástrofes naturais[13]. O facto de não necessitarem de uma pista de aterragem ou de um sistema externo de disparo [14], levam a que sejam utilizados em vários tipos de missões. Contudo, quando se pretende que as missões sejam desempenhadas de forma totalmente autónoma é fundamental que a fase da aterragem seja também efetuada de forma autónoma.

Durante o processo da aterragem autónoma, o correto posicionamento do UAV na pista de aterragem é fundamental, uma vez que se pretende que o UAV aterre, efetivamente, onde é pretendido. Devido a vários fenómenos de interferência, o erro de posição obtido com recurso a sistemas GNSS, poderá não permitir a utilização exclusiva deste método como fonte de localização para a manobra, sendo necessário um método de localização que garanta menor erro e maior estabilidade.

Nesta dissertação, desenvolvida no âmbito do mestrado de engenharia eletrotécnica e de computadores ramo de especialização de sistemas autónomos é apresentada uma solução para a localização relativa durante o processo da aterragem autónoma de um multi-rotor numa pista de aterragem móvel.

1.1 Motivação

Nesta secção iremos detalhar alguns projetos que se enquadram no tema da dissertação.

1.1.1 Projeto EDP - Inspeção de ativos eléctricos da EDP (Substações, linhas de média e alta tensão e aerogeradores)

Este projeto visa o desenvolvimento de um sistema baseado em drones de asa rotativa, com capacidade operacional para endereçar os requisitos de inspeção e monitorização de ativos elétricos. Nesse sentido, definiu-se objetivos que se centraram no desenvolvimento de um veículo autónomo que garantisse uma otimização do processo de inspeção/operação em ativos EDP como linhas, subestações e aerogeradores. E endereçando aspetos como:

- Redução dos risco humano/equipamento inerentes à inspeção;
- Redução dos custos operacionais, não requerendo equipas especializadas para pilotagem e sistematizando as operações;
- Redução do tempo de operação, e a execução de mais serviços com os mesmos meios.

O projeto numa fase inicial identificou os requisitos do cliente EDP Labelec e dos seus parceiros EDP Distribuição/Renováveis para o processo de inspeção dos ativos: linhas, subestações e aerogeradores, tendo resultado nos seguintes desenvolvimentos já implementados: Plataforma adaptada ao processo de inspeção com um payload sensorial composto por câmara termográfica calibrada, câmara de alta resolução e sistema Light Detection And Ranging (LiDAR); Sistema de navegação com elevada exatidão de posicionamento e atitude, com redundância e tolerância a fortes campos magnéticos; Manobras de controlo do drone, utilizando a informação sensorial a bordo proveniente dos sistemas de navegação e perceção (como LiDAR e câmaras electro-opticas) garantindo assim uma redução clara do risco e tempo de operação (por ex: o processo de inspeção autónoma de um aerogerador apenas requer 5min por cada pá da eólica, no caso das substações/linhas, quando identifica um ponto de interesse (ponto quente num dos ativos), o drone efetua uma manobra de varrimento em diferentes ângulos face ao ponto identificado eliminando o efeito de refração solar); Ferramenta de software para o operador com capacidade de desenho e especificação da missão, parametrização da

operação, supervisão da operação, diagnóstico de problemas, capacidade de exportar/aceder aos dados e ferramenta de geração de relatório preliminar do processo de inspeção; Processamento e fusão sensorial em tempo-real a bordo do drone, permitindo a geração automática de relatórios preliminares com a posição geo-referenciada e imagem de pontos de interesse. (por ex: identificação automática nas substações de pontos quentes com recurso a informação termográfica).

1.1.2 Projeto ICARUS: operações de busca e salvamento

O projeto ICARUS teve como objetivo a integração de sistemas robóticos durante as operações de busca e salvamento em desastres naturais, como é o caso dos terramotos de l'Aquila, Haiti ou do Japão. Com este projeto pretendeu-se utilizar as tecnologias desenvolvidas em laboratório em ambientes reais, reduzindo assim a distância e o tempo entre o desenvolvimento e a aplicação destas tecnologias.

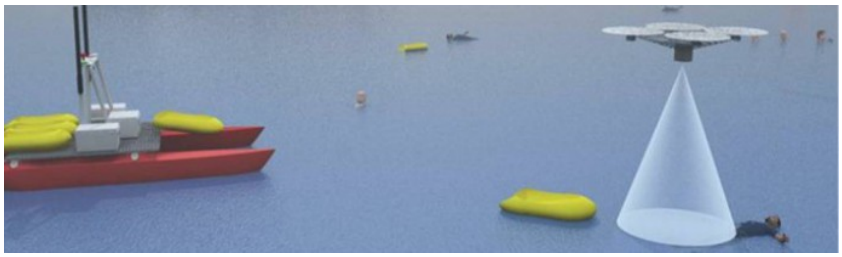


Figura 1.1: Cenário aquático do projeto FP7-ICARUS

Este projeto desenvolveu tecnologia que permite a utilização de robôs em cenários de desastre em conjunto com equipas de resgate. O conjunto de robôs desenvolvido é multidisciplinar, tendo sido criados Autonomouns Surface Vehicle (ASV), UGV e UAV, com o intuito de poder abordar os cenários de desastre por vários pontos e ambiente.

Neste tipo de cenário, o facto de o VTOL ser capaz de efetuar a aterragem e levantar voo do ASV, permite aumentar drasticamente o alcance em missões de busca e salvamento. A cooperação entre estes veículos permite aumentar o alcance do UAV, devido à elevada autonomia do ASV, e reduzir o tempo necessário para efetuar a pesquisa de uma área, uma vez que o UAV tem uma dinâmica bastante superior quando comparado com o ASV.

1.1.3 Projeto Spilless: coordenação multi-robô em operações deteção e mitigação de derrames de petróleo

Projeto europeu que pretende desenvolver uma solução integrada que permita responder a derrames de petróleo utilizando microrganismos nativos com capacidade para biodegradar petróleo (biorremediação) e sua incorporação em veículos autónomos não tripulados que permitam a sua aplicação em zonas afetadas por incidentes de poluição.

Figura 1.2: Spilless - coordenação multi-robô em operações deteção e mitigação de derrames de petróleo

Pretende desenvolver uma abordagem inovadora que possa vir a ser usada como uma primeira linha de resposta a derrames de petróleo associados a acidentes com navios, plataformas *offshore* de petróleo, portos ou outros complexos industriais. Na figura 1.2, é detalhado a metodologia de cooperação multi-robô assumindo a capacidade de um VTOL poder aterrar e levar voo de um ASV em manobras de contenção do derrame de petróleo.

1.1.4 Projeto CoopTrack: Aplicação de UAV em operações de segurança para detecção de intrusão aérea

Nos últimos anos, os drones têm assumido um papel predominante em diversas áreas da nossa sociedade, contudo este rápido desenvolvimento tecnológico também se transformou num grave problema de segurança mundial com a sua utilização em contrabando/-transporte de estupefacientes para dentro das prisões, invasão de espaço aéreo em áreas interditas como aeroportos, em infraestruturas de elevado risco como centrais nucleares, estádios de futebol, e mais recentemente em operações de *hacking* e espionagem. A primeira reação da sociedade a estas ameaças poderá passar pela criação de regras mais restritas para o voo com drones, tendo isso um impacto nocivo para a confiança das empresas em apostar em novas soluções com drones (novas técnicas de inspeção de ativos

elétricos, novas estratégias de apoio à agricultura, etc). No sentido de dar resposta a esta ameaça, o projeto CoopTrack propõe o desenvolvimento de um sistema de detecção de intrusão aérea com recurso a uma equipa de UAVs equipados com sistemas de visão e sensores estáticos de deteção de drones.

1.2 Objetivos

A dissertação endereça o problema da estimação da posição durante o processo de aterragem autónoma de um VTOL. O trabalho tem como objetivo desenvolver um sistemas de localização relativa entre o UAV e a pista de aterragem, que seja robusto e seja capaz de calcular posição mesmo em situações em que recetores GNSS não o consigam efetuar corretamente. Deste modo, o desenvolvimento do projeto implica a concretização dos seguintes objetivos:

- Arquitetura de alto nível do sistema de apoio ao processo de aterragem autónoma;
- Desenvolvimento de uma plataforma de aterragem capaz de ser aplicada aos diferentes cenários de aplicação;
- Implementação de um sistema de perceção a bordo que permita efetuar a deteção da pista para apoio ao processo de aterragem autónoma;
- Desenvolvimento de um estimador de posição de apoio ao processo de aterragem;

1.3 Estrutura

Esta dissertação está organizada em 6 capítulos.

O segundo capítulo diz respeito ao estudo preliminar sobre a temática em questão, onde são descritas algumas abordagens tomadas que estão diretamente relacionadas com o tópico da dissertação.

Os conceitos e fundamentos necessários para a compreensão e desenvolvimento do sistema desenvolvido são apresentados no terceiro capítulo.

A projeção do sistema, bem como a sua arquitetura são apresentado no capítulo quarto.

No quinto capítulo são apresentados o trabalho desenvolvido, assim como os resultados obtidos.

Por último, no sexto capítulo são apresentadas as conclusões e o trabalho a desenvolver no futuro.

Capítulo 2

Estado da Arte

Nesta secção são expostas as escolhas efetuadas por vários autores, para o desenvolvimento do *hardware* e algoritmos para o sistema de aterragem autónoma. As escolhas estão limitadas por vários fatores, como o ambiente de aplicação (i.e.*indoor* ou *outdoor*) ou o tipo de veículo que transporta a pista de aterragem.

F. Cocchioni *et al.* [1], tal como vários autores, assume que a posição obtida por recetores GNSS tem demasiado erro para ser utilizada na manobra de aterragem. Ao descartar este método de localização global, a manobra descrita também pode ser utilizada em ambientes *indoor*. De modo a conseguir efetuar a aterragem sem um sistema GNSS, é utilizado um sistema de visão monocular na parte inferior do multi-rotor, para conseguir detetar o marcador visual da pista de aterragem. O marcador, apresentando na figura 2.1, é composto por duas circunferências concêntricas com raios distintos, permitindo detetar a pista de aterragem a diferentes distâncias. Para além das circunferências, o padrão tem elementos que permitem obter os seis graus de liberdade da posição e orientação da pista face ao veículo (X,Y,Z, *roll*, *pitch* e *yaw*). A plataforma desenvolvida pelo autor tem como segundo propósito, permitir o carregamento das baterias do multi-rotor, que tem contactos de carga nas extremidades do trem de aterragem. Para garantir o preciso alinhamento do UAV para efetuar o processo de carga da bateria, a pista tem cavidades cónicas que permitem assim efetuar o alinhamento passivo do veiculo na pista.

Yang [2] descreve a manobra de aterragem, para um ambiente *indoor*, utilizando como método de localização a técnica Simultaneous Localization and Mapping (SLAM) [15]) Parallel Tracking and Mapping (PTAM) [16]. O local onde o autor pretende que o UAV aterre é definido por uma marca que é descrita por por uma circunferência com um "H", designador de heliporto, no seu interior. Este padrão é definido ao sistema através de imagens prévias sem informação sobre a escala deste, para ultrapassar este falta de

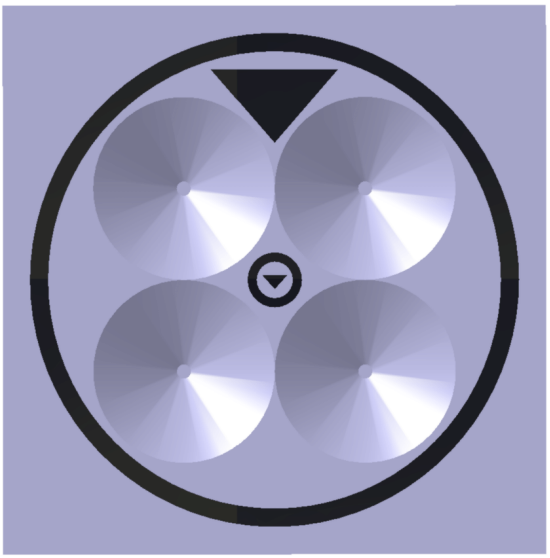


Figura 2.1: Pista de aterragem desenvolvida [1].

informação, o PTAM é inicializado com imagens da pista. Este local está constantemente a ser procurado utilizando Oriented FAST and Rotated BRIEF (ORB) *features* [17].

A estimação de posição da pista é efetuada pelo estimador RANdom SAmple Consensus (RANSAC) [18] através da informação recolhida do SLAM e dos pontos associados à pista de aterragem.

A abordagem típica da manobra de aterragem autónoma tem dois elementos chave: o UAV e a pista de aterragem. É comum, quando é utilizada a abordagem com visão computacional, que exista conhecimento prévio da estrutura e padrão que a descrevem. Com o intuito de detetar a posição do local de aterragem desejado, é habitual utilizar um marcador, que normalmente é visual. O tipo de marcador visual mais comum é o clássico "H"que descreve um heliporto, como o utilizado por S. Yang *et al* [2], exposto na figura 2.2, e Y.Jung *et al* [19].

A utilização de circunferências, como as apresentadas na figura 2.3, como padrão é uma abordagem bastante comum. Para detetar as circunferências na imagem é habitual a utilização técnicas de *ellipse fit* [20]. No caso dos padrões compostos por círculos concêntricos, como é o caso de Cocchioni *et al*, existe a vantagem da redundância no posicionamento no plano horizontal, no entanto a obtenção da atitude (*roll*, *pitch* e *yaw*) não é possível, pois não existe assimetria no patrão. A assimetria pode ser na forma, como C. Hui *et al* [21], ou em outras características, como por exemplo a cor como no padrão utilizado por Cheng-Ming Huang *et al* [22], exposto na figura 2.4.

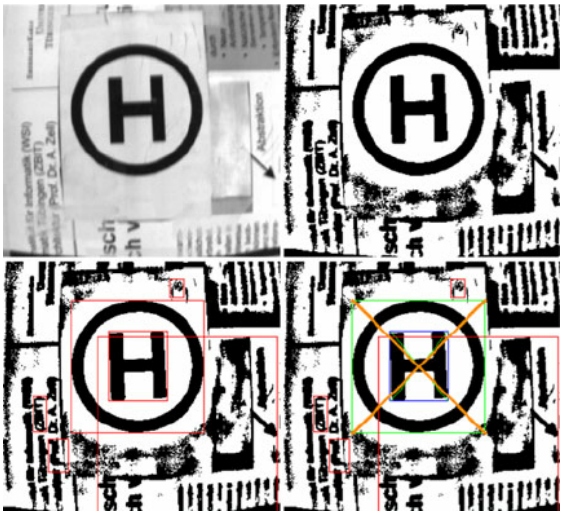


Figura 2.2: Sequencia da deteção do padrão H na imagem capturada. [2]

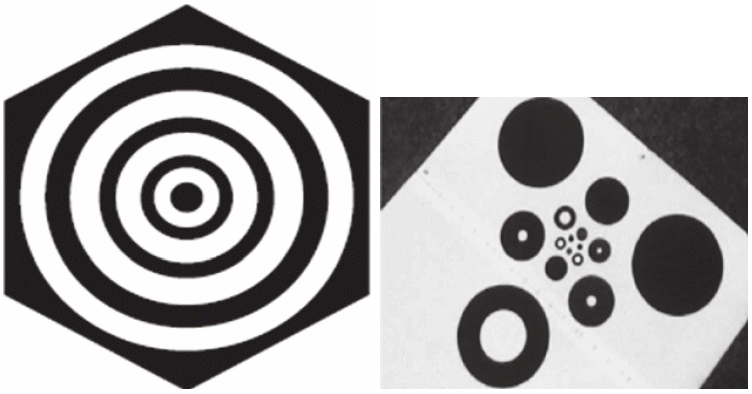


Figura 2.3: Exemplo de padrões utilizados para definir o local da aterragem [3]

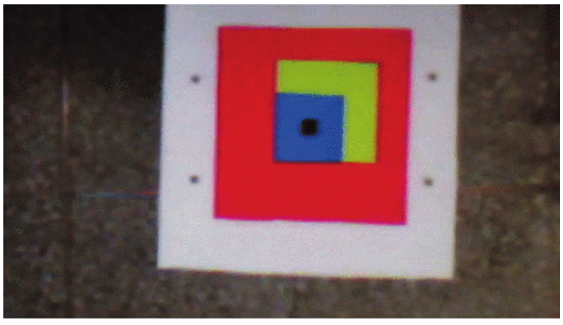


Figura 2.4: Padrão colorido utilizado por [4]

Marcadores ativos são utilizados em várias áreas da robótica móvel. Os padrões definidos nestes marcadores podem ser implementados com vários tipos de emissores, tais como Light Emitting Diode (LED) de luz de espetro visível, como os utilizados por Yutaro Okano *et al* [23] ou Andreas Breitenmoser *et al* [5]. Os marcadores visuais ativos podem ser de outras gamas do espetro de luz, como por exemplo a luz infravermelha, utilizada nos marcadores de Andrea Censi *et al* [24], apresentado na figura 2.5, ou no padrão em 'T' utilizado na aterragem de um UAV por Wang Xiao-Hong *et al* [4]. Wenzel *et al* utilizam um padrão ativo 3D, em que para além de ter emissores LED infravermelhos, tem também emissores num plano perpendicular ao último [25]. A utilização de emissores de luz de espetro infravermelho, tem como principal desvantagem, o facto do Sol também emitir no mesmo espectro, que facilmente pode saturar o sensor utilizado no veiculo, podendo impossibilitar a sua utilização em ambientes exteriores. Os padrões ativos têm a vantagem de ser modelados, como é o caso a implementação de D. KIM *et al* [26], que utiliza um marcador ativo na localização de multi-robôs terrestres. O padrão luminoso é modelado de forma a que seja possível excluir falsos positivos no processamento da imagem.

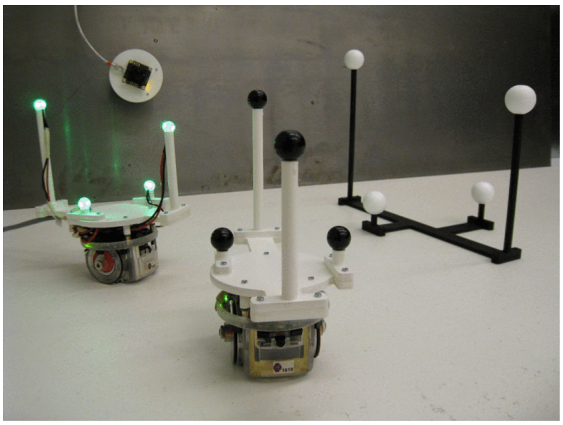


Figura 2.5: Marcadores ativos utilizados na localização de UGV [5]

Padrões monocromáticos como é o caso de *April Tags* [27] e *QR codes* são também usados como marcadores. Mengyin Fu *et al* [28], utiliza um padrão misto com um código QR dentro de uma circunferência, apresentado na figura 2.6. Deste modo, é possível detetar a forma da circunferência quando o veículo está mais afastado, e quando a distância for menor utilizar o código QR, permitindo obter a posição com um erro menor.

A utilização de técnicas de visão para obter a posição de um veículo é uma abordagem comum, no entanto, geralmente os autores optam por descartar a posição sistemas

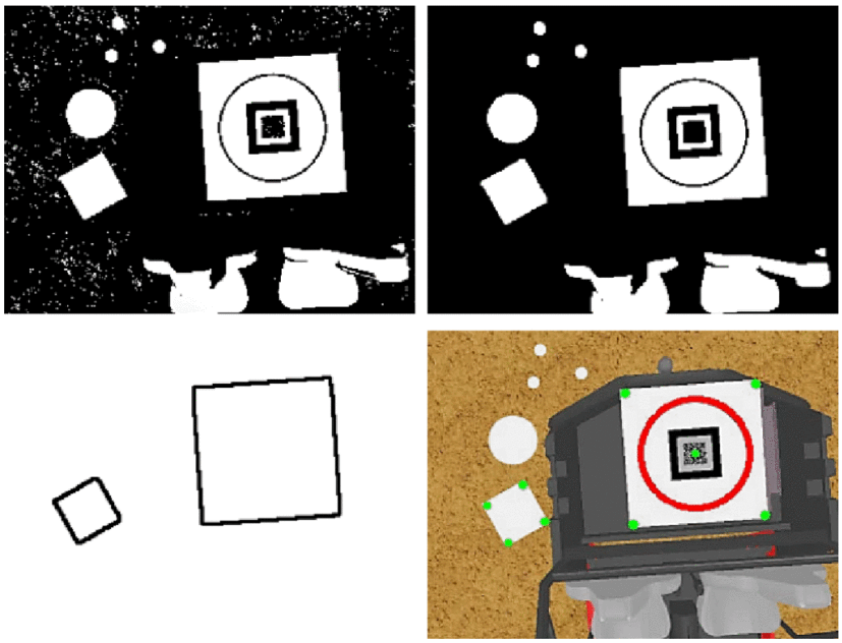


Figura 2.6: Fases do processamento de imagem para detetar o padrão.

globais como recetores GNSS. Este tipo de abordagem apesar de permitir a utilização destas técnicas em ambientes interiores, onde as condições de luminosidade são mais controladas, condiciona a sua robustez, uma vez que não existe um método complementar que possibilite a obtenção de posição caso a visão computacional falhe.

Em cenários de *field robotics*, a utilização das abordagens expostas neste capítulo é condicionada pelas condições de luminosidade variáveis, que podem com facilidade impossibilitar a deteção dos marcadores passivos. Deste modo, a utilização de marcadores ativos apresenta-se como uma abordagem que permite a deteção visual do padrão, mesmo em condições de luminosidade extremas (i.e noite ou zênite). No caso de cenários de *field robotics*, a utilização de sistemas como recetores GNSS em modos RTK é possível, garantindo a redundância e maior robustez na obtenção do posicionamento relativo entre o VTOL e a pista de aterragem.

Esta página foi intencionalmente deixada em branco.

Capítulo 3

Fundamentos Teóricos

Nesta capítulo serão abordados os fundamentos teóricos necessários à compreensão dos capítulos seguintes, nomeadamente, conceitos de visão computacional, sistemas GNSS e métodos de sincronismo multi-robô.

3.1 Visão Computacional

3.1.1 Modelo *Pinhole*

A relação entre as coordenadas de um ponto do mundo com as coordenadas da projeção deste no plano da imagem é descrito pelo modelo de câmera *pinhole* representado na figura 3.1.

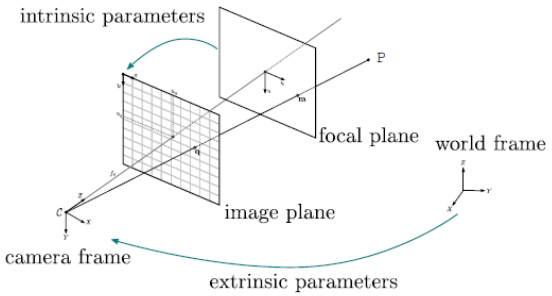


Figura 3.1: Modelo *Pinhole*

A projeção do ponto P no plano da imagem (*image plane*) pode ser expresso da seguinte forma

$$\begin{bmatrix} v \\ u \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.1}$$

Sendo que X, Y e Z são as coordenadas do ponto P no referencial global (*world frame*) e f_x, f_y, c_x, c_y a distância focal e o ponto principal, respetivamente, nas componentes xy. A distância focal e o ponto principal definem as características do conjunto câmera-lente, sendo denominados de parâmetros intrínsecos. A posição e rotação da câmera em relação ao referencial global são denominados de parâmetros extrínsecos e podem ser obtidos da seguinte forma

$$C = -R^T T \tag{3.2}$$

Sendo que R é a matrix de rotação e T a translação 3D da câmera.

3.1.2 Problema *Perspective-N-Points*

Utilizando apenas uma câmera, é possível obter a posição relativa desta em relação a um conjunto de pontos sobre os quais se tem conhecimento no referencial global. Para tal é necessário cumprir dois requisitos, ter os parâmetros intrínsecos da câmera e um conjunto de pontos no referencial da imagem e a respetiva correspondência no referêncial global (mínimo quatro elementos). Existem várias técnicas de resolução do problema, destacando-se as suportadas pela biblioteca *OpenCV* [29]:

- P3P [30]- Método que utiliza quatro pontos para obter a posição no referêncial global. Apesar de referir que utiliza apenas três pontos, é necessário um quarto ponto para eliminar a ambiguidade criada pelo facto deste método retornar quatro possibilidades para R e T;
- EPnP [31] - Assume que cada ponto é o resultado da soma do peso de quatro pontos virtuais de controlo, sendo estes a incógnita;
- Iterativo - A obtenção da posição da câmera pode ser obtida de forma iterativa, procurando a solução em que o erro de re-projeção é menor. No caso da biblioteca *OpenCV* é utilizado o algoritmo Levenberg-Marquardt [32].

3.2 GPS RTK

O GNSS é um método de localização global utilizado como sistema de localização de vários sistemas autónomos, no entanto este método de localização global geralmente apresenta um erro em posição que em alguns cenários de aplicação poderá não ser aceitável. Este erro é proveniente dos seguintes tipos de fontes [33]:

- Sistema GNSS - Precisão de relógio, ou erro da oŕbita dos satélites;
- Atmosfera - Atrasos do sinal GNSS devido às caraterísticas da troposfera ou ionosfera;
- Recetor GNSS - Multicaminho devido à proximidade de estruturas elevadas em que os sinais possam refletir. Ruído eletromagnético proveniente dos circuitos eletrónicos que compõem o recetor GNSS.

É possível mitigar o efeito destas fontes de erro utilizando sistemas de *GNSS Augmentation* como o Statellite Based Augmentation Systems (SBAS) ou sistemas RTK. Estes métodos, têm como objetivo de melhorar a precisão, confiabilidade e oferta dos sinais do sistema GNSS.

O sistema RTK é composto por dois componentes distintos, a estação base e o *rover*, como detalhado na figura 3.2. A estação base é, tipicamente, um recetor GNSS estático do qual se tem conhecimento sobre a sua localização, já o *rover* é o recetor sobre o qual é pretendido obter uma posição com menor erro. Para que seja possível utilizar RTK, é necessário que a distância entre a estação base e o *rover* seja inferior a aproximadamente 20km, garantindo assim duas condições fundamentais, que os dois recetores "observam"os mesmos satélites e os sinais emitidos pelos satélites em orbita a 20km sofrem os mesmos atrasos provocados pela atmosfera. Para além da necessidade da existência de uma estação base, para que esta possa enviar a informação necessária para o *rover* é necessária uma linha de comunicação, que geralmente é um rádio Ultra High Frequency (UHF).

O GPS-RTK é capaz de obter posição milimétrica do *rover*, utilizando para tal a estação base. O processo de obtenção da posição pode ser descrito nos seguintes passos:

- A estação base calcula o *pseudorange* (distância entre o recetor GNSS e o satélite) contando o número de ciclos da onda portadora do sinal enviado pelos satélites GNSS;
- O erro entre o *pseudorange* e a posição conhecida da base é transmitido para o *rover* via uma linha de comunicação.

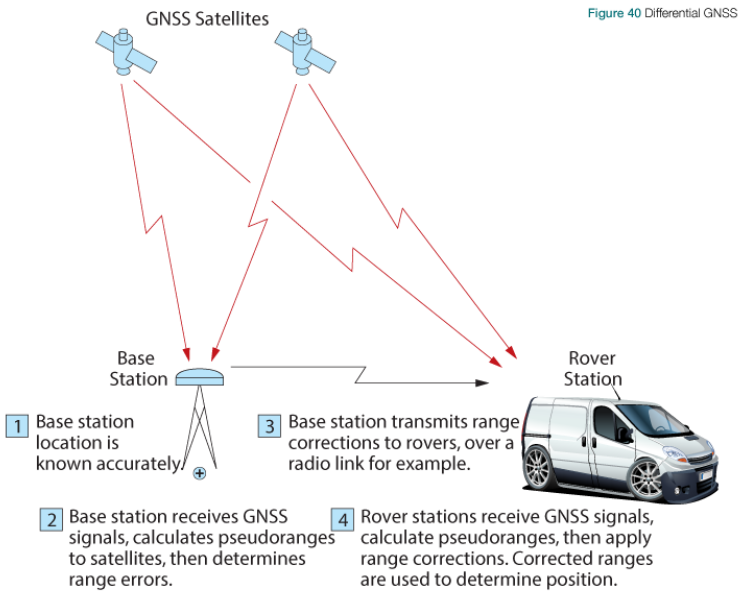


Figura 3.2: Cenário de aplicação de GPS-RTK [6]

- Utilizando esta informação, o *rover* resolve a ambiguidade de fase, para determinar o número de ciclos completos da onda portadora.

Um recetor em modo *standalone/single*, habitualmente para obter a sua posição correlaciona o pseudocódigo recebido com o gerado por ele, para isso o recetor vai atrasando o código recebido para determinar o atraso em que a correlação entre os dois pseudocódigos é maior, sendo este o tempo que o sinal demorou entre a emissão e a sua receção. A pseudo-distância para um instante t pode ser obtida da seguinte forma [34]

$$R_r^s(t) + c\delta^s(t) = \varrho_r^s(t) + c\delta_r(t) \tag{3.3}$$

Sendo que $R_r^s(t)$ é a pseudo-distância entre o recetor r e o satélite s. c é a velocidade da luz e δ^s o *clock bias* do satélite. No lado direito da equação estão expostos o *clock bias* do recetor (δ_r) e a distância geométrica ($\varrho_r^s(t)$) entre o satélite s e o recetor r.

O bias do relógio do satélite s, para o instante t, pode ser obtido utilizando para tal a informação sobre o estado do relógio enviada pelo próprio satélite sob a forma dos coeficientes polinomiais a_0, a_1 e a_2 com um tempo de referência t_c. Para corrigir o efeito do campo gravítico da terra, é utilizado o tempo $\delta^r el$.

$$\delta^s(t) = a_0 + a_1(t - t_c) + a_2(t - t_c)^2 + \delta^r el \tag{3.4}$$

A distância geométrica (ϱ_r^s) entre o recetor r e o satélite s, pode ser obtida através da distância euclidiana entre a posição do satélite (X^s, Y^s e Z^s) e a posição do recetor (X_r, Y_r e Z_r).

$$\varrho_r^s(t) = \sqrt{(X^s(t) - X_r)^2 + (Y^s(t) - Y_r)^2 + (Z^s(t) - Z_r)^2} \tag{3.5}$$

O pseudocódigo tem uma taxa relativamente baixa, levando a que mesmo corretamente correlacionado, possa ocorrer um erro de posição de alguns metros. Recetores mais complexos, utilizam também medições da fase da onda portadora, uma vez que esta tem uma frequência bastante superior, permitindo assim posição com precisão milimétrica. A obtenção da fase da onda portadora é efetuada do seguinte modo:

$$\Phi_r^s(t) + f^s\delta^s(t) = \frac{1}{\lambda^s}\varrho_r^s(t) + N_r^s + f^s\delta_r(t) \tag{3.6}$$

$\Phi_r^s(t)$ é a fase da portadora, no instante t expressa em ciclos, λ^s é o comprimento de onda da onda e N_r^s é um número inteiro que corresponde à ambiguidade de fase, o valor que se pretende estimar. Por fim, f^s corresponde à frequência da onda portadora ($\frac{c}{\lambda^s}$)

No entanto, para obter as medidas utilizando a fase da onda portadora, é necessário efetuar um processo denominado de "resolução da ambiguidade de fase". Para que não seja necessário resolver a ambiguidade de fase para toda a onda, o recetor obtém a posição utilizando o pseudocódigo, sendo assim necessário resolver a ambiguidade para alguns ciclos.

O *rover*, com o número de ciclos da onda portadora obtido, é assim capaz de obter a sua posição com precisão milimétrica.

3.2.1 *Differencing*

Os erros referidos no inicio da secção, podem ser eliminados utilizando uma técnica denominada de *differencing*, representada na figura 3.3, podendo ser de três tipos distintos:

- *Simples* - Neste método existem duas opções, diferença entre satélites ou diferença entre recetores. Através deste método é possível suprimir o erro de relógio dos satélites;
- *Dupla* - Este método utiliza duas diferenças simples para gerar uma diferença dupla, que pode ser entre satélites ou recetores, eliminando assim o erro de relógio do recetor;
- *Tripla* - São utilizadas duas diferenças duplas geradas em dois instantes de tempo distintos, permitindo detetar e corrigir *cycle slips* (descontinuidade na obtenção da fase da portadora).

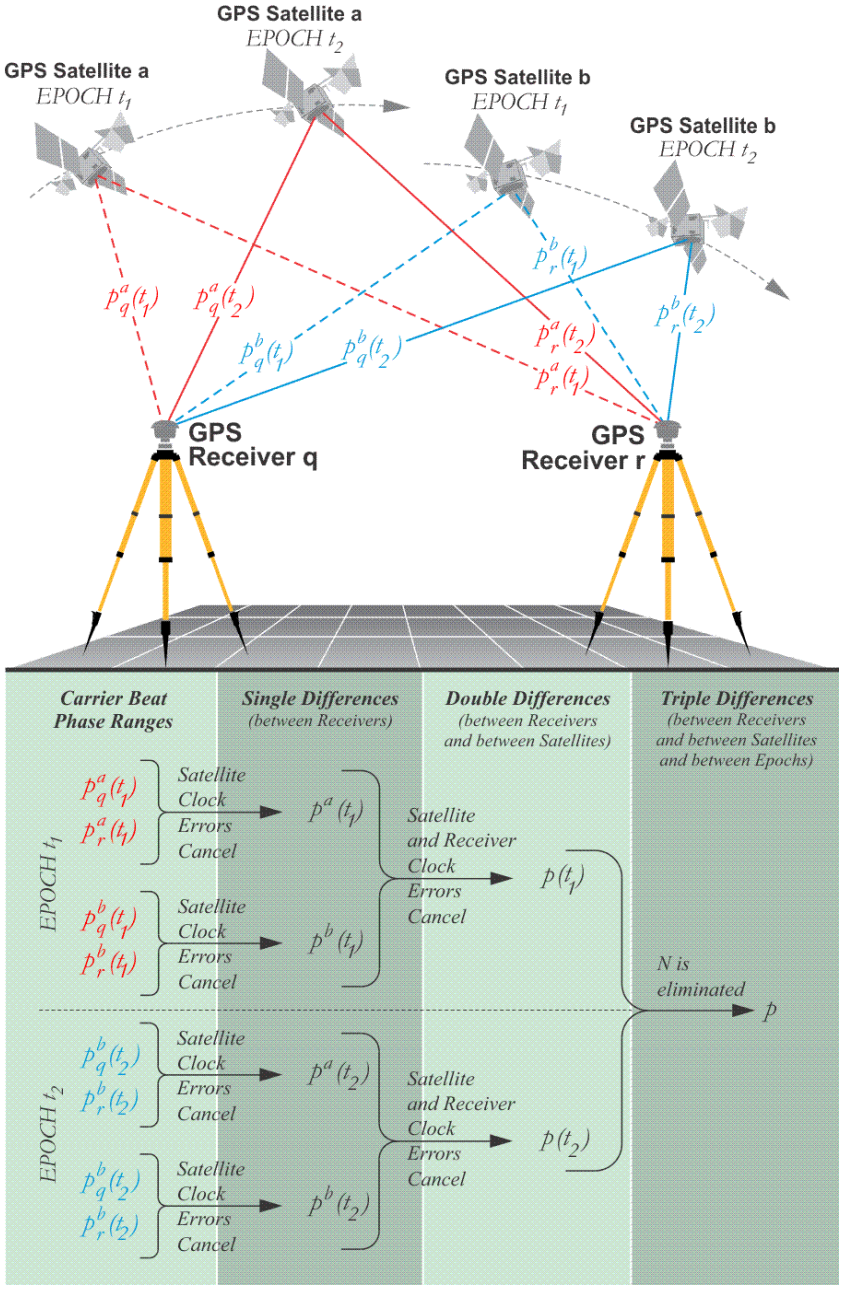


Figura 3.3: Diagrama com os vários tipos de *differencing*

3.3 Sistema de posicionamento relativo RTK em *baseline* móvel

3.3.1 RTKLib

Atualmente existem vários recetores GNSS que têm integradas técnicas de RTK, no entanto estes têm um custo bastante elevado, quando comparado com o baixo custo de um multi-rotor em que este pode ser instalado. Como alternativa, Takasu Tomoji desenvolveu a biblioteca *opensource* RTKLib [35], permitindo assim obter uma solução de posição RTK, sendo compatível com um grande espetro de recetores GNSS, devido às mensagens de recetores GNSS que suporta, destacando-se as mensagens *standard* RINEX, BINEX, RTCM, NMEA e as mensagens proprietárias das marcas NovAtel, u-blox e Hemisphere.

Para além da biblioteca, também são fornecidas várias aplicações, para sistemas operativos Windows ou Linux, que permitem a utilização *out of the box* desta ferramenta. Para linux, destacam-se duas ferramentas distintas que permitem a implementação de RTK: *rtkrcv* e *str2str*. A aplicação *str2str* desempenha a função de estação base do RTK, recebendo a *stream* de mensagens *RAW* do recetor GNSS e enviando a informação de correções para o *rover*. O tipo da *stream* pode ser: porta série, Transmission Control Protocol (TCP) (servidor ou cliente), Networked Transport of RTCM via Internet Protocol (NTRIP) (servidor ou cliente), ou para aplicações de pós-processamento, guardado para ficheiro. A aplicação *rtkrcv* exerce a função de *rover*, tendo como entradas as correções geradas pelo *str2str* e os dados *raw* do recetor GNSS.

Para utilizar o RTKLib é necessário, pelo menos, um computador onde possam ser executado o *rtkrcv* e o *str2str*, no entanto o cenário típico de utilização, consiste em dois computadores, um instalado na estação base, a executar a aplicação *str2str* e um segundo instalado no *rover* a executar a aplicação *rtkrcv*. Num cenário em que a *baseline* seja pequena, o envio de correções pose ser efetuado por TCP, utilizando uma rede WiFi. Caso a *baseline* seja superior, a utilização de rádios UHF conectados aos conetores, também podem ser utilizados, garantindo a receção das correções mesmo a distâncias elevadas.

Moving Baseline

Entre os vários métodos de calculo de posição fornecido pelo RTKLib, existe o modo *moving baseline*, em que a posição da estação base não está fixa. Neste modo de processamento, não é pretendido ter a posição da base e *rover* no referencial global, mas

sim a posição relativa entre ambos. Uma vez que a base é móvel, a sua posição não é definida, mas sim calculada a cada *epoch* em modo *single*. Tendo a posição da base e a posição estimada do *rover*, utilizando para tal Extended Kalman Filter (EKF) é possível obter posição *float*, que utilizando MLAMBDA [36] para resolver a ambiguidade de fase e assim obter posição em estado *fix*.

3.4 Sincronização Multi-Robô

A atualização do tempo de sistema, em relação a uma referência global é uma prática comum nos sistemas autónomos. Deste modo, é garantido que toda a informação guardada tem o *timestamp* correto, permitindo a comparação e correlação destes dados com os de outros sistemas para o mesmo instante/período de tempo. Existem várias soluções que garantem a sincronização do tempo do sistema em relação a uma referência, no entanto o protocolo mais utilizado é o Network Time Protocol (NTP) [37] que tem a arquitetura descrita na figura 3.4. O NTP tem uma hierarquia que pode chegar até 15 niveis, denominados de strata, organizados entre os valores 0 e 16. Quanto menor for o valor do stratum, mais estável e menor erro tem a fonte de sincronismo. Posto isto, no stratum 0 estão incluídos relógios atómicos, relógios dos sistemas GNSS ou relógios rádio, fontes estas com o menor erro possível, sendo considerados fontes de relógio perfeitas. Estes não fazem parte da rede NTP, sendo a referência primária, e são considerados apenas servidores, ao contrário dos restantes stratum, que podem ser clientes e servidores, dependendo do tipo de funcionamento. No NTP é possível efetuar os seguintes tipos de associações:

- Cliente/Servidor - Configuração comum, em que o cliente é o stratum maior e o servidor o stratum menor. O cliente efetua um pedido ao servidor e espera uma resposta.
- Modo Simétrico - Pares com stratum baixo funcionam como *backup* de cada um. Caso um elemento perca a conexão ao servidor, os outros elementos enviam as correções para este.
- *Boardcast/Multicast* - Neste modo, o número de clientes é bastante superior ao número de servidores. Pacotes *broadcast* ou *multicasst* são utilizados para transferir a informação do tempo.

Apesar do NTP ser o método de sincronização mais utilizado, existem soluções que garantem o sincronismo com melhores caraterísticas, chegando a valores inferiores ao

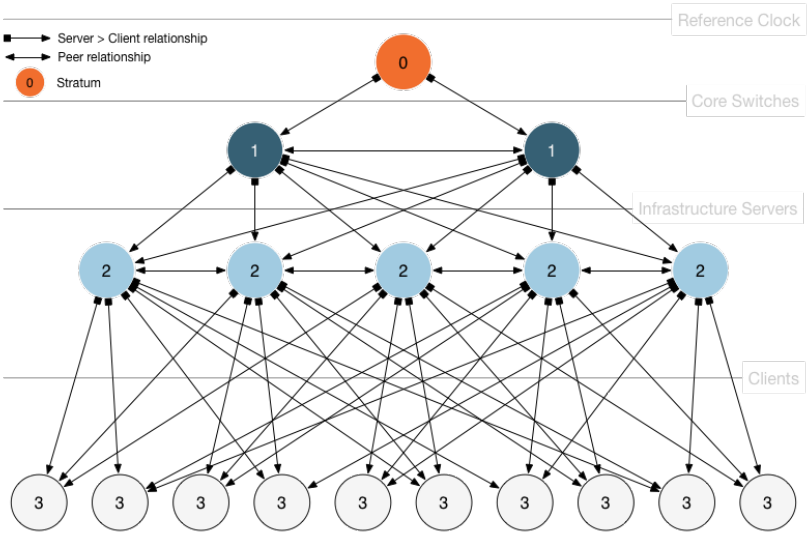


Figura 3.4: Hierarquia cliente-servidor no esquema NTP [7]

micro-segundo, como é o caso do Precision Time Protocol (PTP) [38]. Este método, ao contrário do NTP não é implementada em *software*, mas sim em *hardware*, sendo necessário o suporte ao PTP por parte de toda a interface de rede que suporta a conexão entre os diferentes computadores. No NTP existem dois atores distintos, o *master* e o *slave*, existindo uma sequência de quatro mensagens entre eles, para obter os *timestamps* necessários para sincronizar o tempo do *slave*.

Apesar de existirem soluções de baixo erro, como o PTP, apenas em algumas aplicações é necessário um erro de sincronismo inferior ao microssegundo, pelo que geralmente é utilizado o NTP ou variantes do mesmo, como é o caso do openntpd [39] ou o chrony [40].

3.4.1 Chrony

O chrony é baseado em NTP, no entanto foi desenvolvido para ser utilizado em aplicações onde o computador é reiniciado várias vezes, ou em que existem várias falhas da *stream* por parte do servidor. O facto do chrony conseguir utilizar o tempo guardado no Real-time clock (RTC) do computador, é uma vantagem face ao NTP, permitindo corrigir o tempo do sistema quando o sistema inicia. Tal como o NTP, o chrony suporta recetorees GNSS, como fonte de sincronismo. A robustez do chrony face a falhas da *stream* e a variações do estado do computador (i.e. vários ciclos de *reboot*), levam a que ele seja utilizado em soluções de baixo custo com processador ARM. No entanto, quando

comparado com o NTP, o chrony ainda apresenta algumas desvantagens, destacando-se:

- Número reduzido de sistemas operativos suportados;
- Não suporta os modos *broadcast* e *multicast*, no entanto estes são menos precisos e seguros que a configuração cliente servidor;
- Necessita de outros programas para receber a informação via SHM ou SOCK, como é o caso na utilização de recetores *GNSS* como fonte de *timestamp*.

Esta página foi intencionalmente deixada em branco.

Capítulo 4

Projeto de um Sistema de Aterragem Autónoma

Um dos problemas principais da manobra da aterragem autónoma de um VTOL é a obtenção da posição relativa entre os dois sistemas, a pista de aterragem e o UAV. Este problema prende-se pelo facto de que, geralmente, a pista de aterragem tem dimensões reduzidas, pouco superiores às do veículo aéreo, limitando assim a tolerância de posição. Neste capítulo é detalhada a arquitetura geral do sistema que permitirá endereçar todos os requisitos enumerados previamente, assumindo como base o conteúdo apresentado nos capítulos do estado da arte e fundamentos teóricos.

4.1 Arquitetura do sistema

A manobra de aterragem autónoma será aplicada em cenário exterior em que a pista de aterragem está instalada no ASV ROAZ II [41]. Devido às características do ambiente e do veículo onde a pista estará instalada, esta não será estática, sendo a sua posição no mundo e atitude variáveis. Com o objetivo de obter os seis graus de liberdade da pose (atitude e posição) da pista de aterragem, é necessário desenvolver um sistema com a capacidade de os obter em vários cenários exteriores, em que a luminosidade pode variar ou podem ocorrer oclusão ou outros fenómenos menos favoráveis para um recetor GNSS.

Como referido anteriormente, o erro de posição aceitável na manobra de aterragem autónoma é bastante reduzido, pelo que a utilização de um recetor GNSS em modo *single* não permitiria a obtenção de posição com erro suficientemente baixo. Nesse sentido, a técnica de obtenção de posição global com recurso a recetores GNSS a utilizar, será GPS

RTK, permitindo deste modo erro na escala dos centímetros. Uma vez que é pretendido obter a posição relativa entre a pista de aterragem e o UAV, a configuração que mais se adequa é *moving-baseline*.

Visto que o ASV estará sujeito à ondulação da água, para além da informação sobre a posição da pista é também necessário saber a sua atitude, pelo que iremos utilizar um Inertial Measurement Unit (IMU).

Para que o VTOL possa receber informação sobre a *pose* da pista de aterragem, é necessária a utilização de uma linha de comunicação sem fios. Apesar da utilização deste método de comunicação seja simples e de fácil implementação, está sujeito a falhas (i.e. carga elevada na rede, distância elevada) que não são aceitáveis, quando considerando a necessidade de ter informação sobre a *pose* da pista de aterragem a uma taxa elevada. Os constrangimentos da rede não devem limitar a manobra, pelo que é fundamental também utilizar um método de obtenção da *pose* que não dependa de uma linha de comunicação. A utilização de técnicas de visão computacional como as descritas na secção 3.1 através de uma câmera de espetro visível instalada no VTOL, permitirá o cálculo da posição da pista de aterragem. Esta técnica pressupõe o conhecimento prévio de um objeto sobre as dimensões e posição no mundo de um marcador. Visto que o cenário de aplicação da manobra de aterragem é ao ar livre, as condições de luminosidade podem variar drasticamente, pelo que a implementação de visão computacional com um marcador visual passivo pode ser muito complexa. Posto isto, para garantir a deteção do padrão em todas as condições, o marcador deverá ser ativo.

Uma vez que a câmera instalada no UAV não estará a captar imagens a uma taxa elevada, o marcador não necessita de estar sempre ligado, necessitando apenas de estar ativo quando a câmera captar uma imagem. Esta liberdade permite, no caso da utilização de emissores LED, a utilização de correntes elétricas superiores às nominais para a alimentação dos LED, levando a que a luz emitida por estes seja também superior. A utilização do marcador ativo nesta configuração necessita apenas que o disparo do marcador visual e da câmera de espetro visível aconteça sincronizado, garantindo assim que em cada imagem capturada o marcador esteja acesso.

A arquitetura de alto nível do sistema proposto que garanta a obtenção da posição relativa entre o UAV e a pista de aterragem é representada na figura 4.1.

O sincronismo entre o disparo da câmera e dos LED será desempenhado utilizando o chrony (3.4.1) com as mensagens RAW e Pulse-Per-Second (PPS) provenientes do recetor GNSS como servidores, garantindo deste modo que os tempos de sistema de ambas as unidades de processamento estão sincronizadas com o tempo GPS. Deste modo o disparo efetuado pelos dois sistemas é estanque, não necessitando de qualquer tipo de linha de

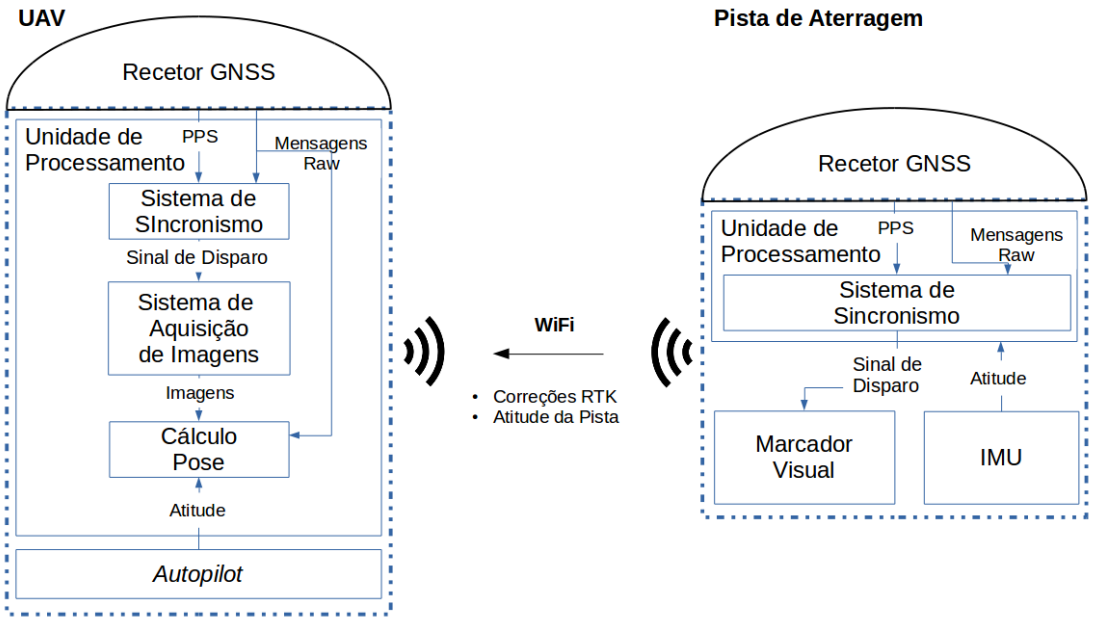


Figura 4.1: Arquitetura de alto nível do sistema desenvolvido (UAV e pista de aterragem)

comunicação entre eles. A posição RTK que se pretende obter pode ser conseguida utilizando a biblioteca RTKLib (3.3.1), possibilitando a utilização de recetores de baixo custo. Esta biblioteca disponibiliza o modo de operação *moving-baseline* que permite obter a posição relativa entre o *rover* e a base móvel com erro bastante reduzido.

O sistema computacional selecionado para desempenhar as funções necessárias é a Odroid XU3. Uma vez que para obter os melhores resultados de sincronismo os dois sistemas devem ser idênticos, tanto a pista de aterragem como o UAV usarão o mesmo modelo. Este computador de baixo custo, de arquitetura ARM, tem as seguintes características:

- Processador Samsung Exynos5422 Cortex™-A15 2.0Ghz quad core e Cortex™-A7 quad core;
- 2Gb de memória ram LPDDR3 933MHz;
- Interfaces: USB 3.0 x 1, USB 3.0 OTG x 1, USB 2.0 x 4

Esta página foi intencionalmente deixada em branco.

Capítulo 5

Implementação e Resultados

Neste capitulo será abordado o desenvolvimento e implementação do sistema de localização relativa a aplicar na aterragem autónoma de um UAV em base móvel.

5.1 GPS RTK

Uma vez que se pretende obter a posição relativa entre a pista de aterragem e o VTOL, a configuração RTK a utilizar que mais se adequa é o *moving-baseline*. Deste modo, como referido no capítulo dos fundamento teóricos (3.3.1), é possível obter posição relativa com baixo erro, utilizando a biblioteca RTKLib e recetores GNSS de baixo custo.

A solução implementada utiliza dois recetores U-BLOX NEO-M8T [42], como base móvel instalado na pista de aterragem e outro no UAV com a função de *rover*. As características dos recetores estão descritas na tabela 5.1.

Tabela 5.1: Especificações do recetor u-blox neo-m8t

Frequência de Navegação Máxima	4Hz (GPS + GLONASS)
Número de Canais	72
Constelações Suportadas	GPS, GLONASS, BEIDOU GALILEO
Tipos de Mensagens	NMEA, RINEX, UBX (proprietário) e RTCM

A biblioteca RTKLib suporta o protocolo de comunicação proprietário ubx da u-blox, o que permite a fácil implementação dos recetores selecionados. Os dois recetores estão configurados para enviar os dados *RAW* a uma taxa de 5HZ que, no caso da aplicação str2str do RKTlib, utiliza para gerar as correções a enviar para o *rover*. As correções são enviadas, via tcp utilizando a ligação sem fios, para o *rover* onde é executada a aplicação

rtkrcv no *middleware* ROS, permitindo publicar a informação relativa à *baseline* em tópicos.

Com o objetivo de averiguar a qualidade da posição relativa entre a base e o *rover* na configuração *moving-baseline* do RTKLib, foram efetuados alguns ensaios. Um dos ensaios efetuados consiste em desempenhar uma trajetória que seguia o perímetro de um quadrado 3x3m, como descrito na figura 5.1, existindo uma *baseline* fixa de 1.8m entre o *rover* e a base móvel.

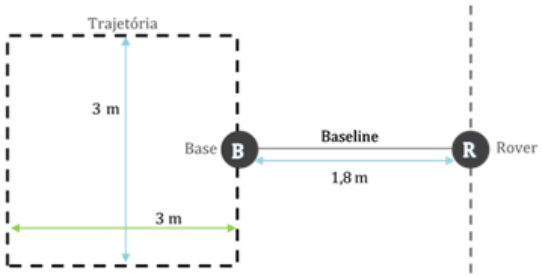


Figura 5.1: Trajetória descrita no teste de qualidade da *baseline* no modo RTK *moving-baseline*

A trajetória apenas foi descrita quando a solução do RTK atingiu o estado *fix*, tendo demorado aproximadamente quarenta segundos até o atingir. Analisando a evolução da *baseline* ao longo do tempo apresentada na figura 5.2, é possível afirmar que enquanto garantia a solução *fix*, o RTKLib apresentou resultados de *baseline* consistentes, tendo perdido este estado apenas por breves instantes, provocados por oclusões durante o ensaio.

Uma vez que a solução em estado *float* não apresenta resultados consistes, este estado não é considerado para a análise de erro do RTKLib apresentada na figura 5.3. Para a solução em estado *fix* o erro médio da *baseline* de 1cm com um desvio padrão de 2cm.

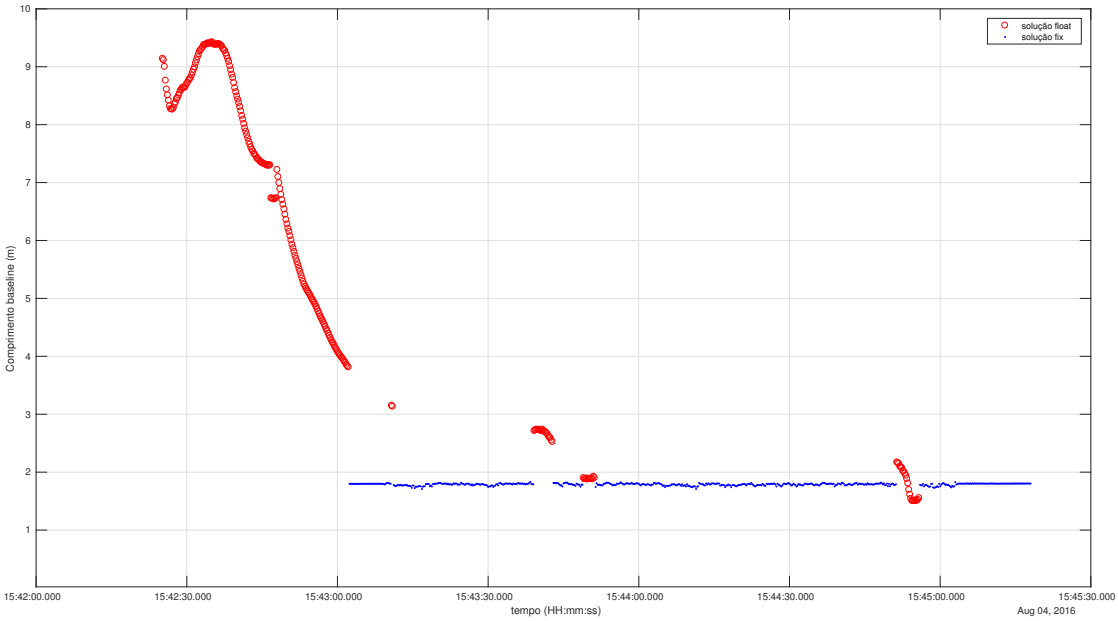


Figura 5.2: *Baseline* entre o *rover* e a base móvel ao longo do tempo

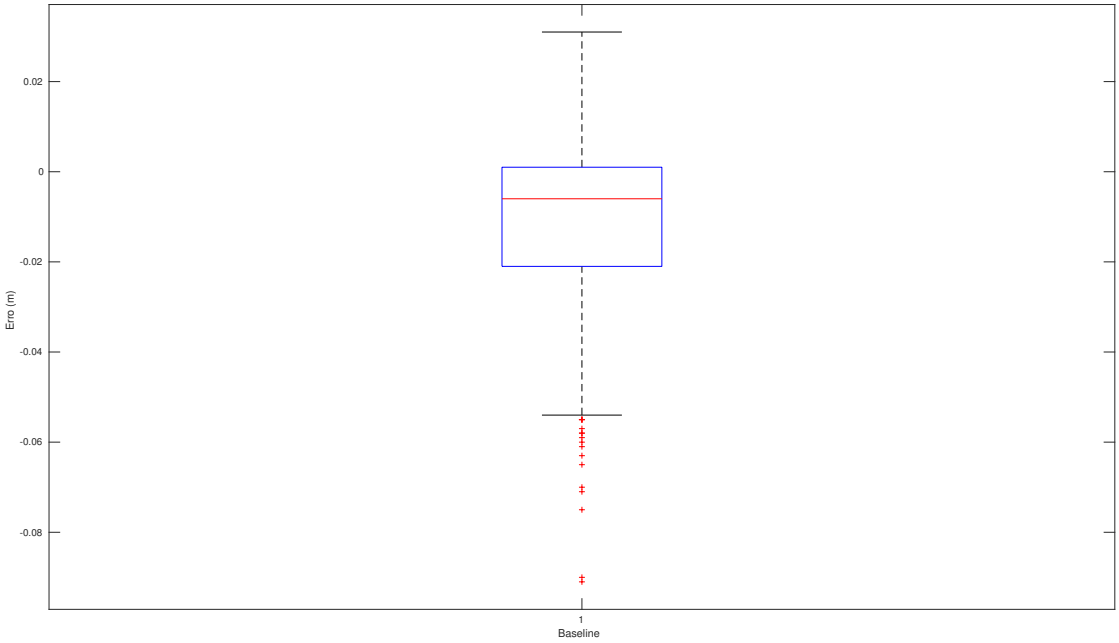


Figura 5.3: Erro da *baseline* no teste efetuado

5.2 Marcador Visual

A pista de aterragem desenvolvida é identificável através de um marcador visual ativo. O padrão desenvolvido, representado na figura 5.4, é composto por onze pontos organizados em três grupos distintos:

- Grupo Central - Composto apenas pelo ponto 1,situado no centro do padrão;
- Grupo Interior - Composto pelos pontos 2,4,5,6 e 7;
- Grupo Exterior - Composto pelos pontos 3,8,9,10 e 11. É a replicação do Grupo Interior, mas com uma escala maior.

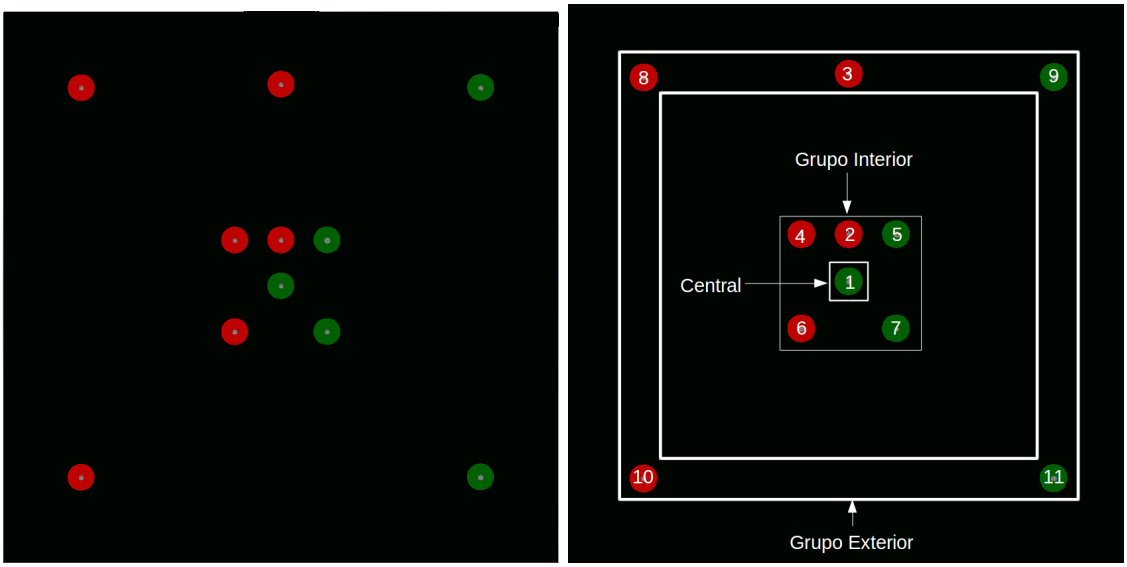


Figura 5.4: Esquerda - Padrão do marcador visual. Direita - Diferentes grupos de pontos que definem o padrão do marcador visual.

O padrão está instalado na face da pista de aterragem, permitindo o seu enquadramento pelo UAV durante toda a manobra de aterragem. O padrão é composto por 6 pontos de cor vermelha e 5 de cor verde. A utilização de duas cores, permite em conjunto com a introdução dos LED número 2 e 3 (figura 5.4 direita), eliminar a simetria do padrão, que impossibilitaria obter os seis graus de liberdade da pose (*roll, pitch, yaw,* x, y e z).

Os grupos interior e exterior, são iguais, mas com escala diferente, para permitir que o padrão seja identificado a diferentes altitudes. O grupo exterior como tem uma

dimensão superior, permite a identificação da pista de aterragem a uma altitude elevada, já o grupo interior, tem uma dimensão que permite que seja totalmente visualizado, mesmo quando o UAV está pousado na pista, momento este em que o padrão exterior já não é enquadrado pela câmara.

O marcador ativo é composto por 11 LED, correspondendo cada ponto do padrão a um LED. Uma vez que os onze LED não são iguais, o brilho emitido por eles para a mesma corrente não é a mesma. Com o objetivo de contornar esta limitação, foi desenvolvido um circuito de controlo de onze canais (figura 5.5), possibilitando ajustar individualmente o brilho de cada um dos onze LED, garantindo deste modo que na imagem capturada os onze pontos brilhantes com a mesma intensidade e dimensão.

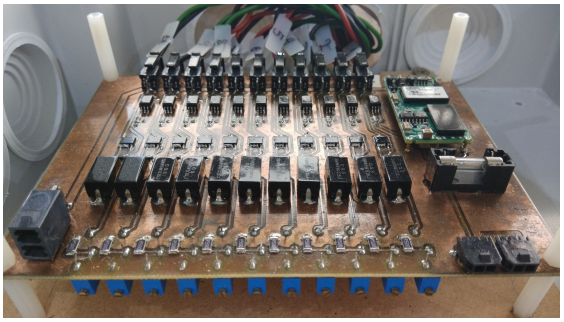


Figura 5.5: Placa desenvolvida para controlar individualmente a intensidade de corrente que percorre cada um dos onze LED do marcador.

De forma a garantir o máximo de brilho possível os LED são percorridos, por breves instantes, por corrente elétrica com intensidade superior aos valores nominais, permitindo assim que a intensidade luminosa emitida seja também superior nestes instantes. Posto isto, é necessário modular e sincronizar o disparo dos LED que compõem o marcador, para que este seja detetado pela câmera instalada no UAV.

No UAV está instalada uma câmera de espetro visível Pointgrey Chameleon de 1.3MP que suporta disparo por *hardware*, o que permite capturar uma imagem quando esta receber um sinal. Deste modo, para que a câmera adquira imagens sempre que o padrão está acesso, apenas é necessário que o sistema computacional do UAV envie o sinal de disparo da câmera ao mesmo tempo que o sistema computacional da pista de aterragem liga os emissores LED.

5.3 Sincronismo do Disparo

Uma vez que não se pretende que exista uma dependência de linha de comunicação entre o UAV e a pista de aterragem na obtenção da pose utilizando visão, o comando de disparo da câmera e dos LED não pode ser enviado pela rede. Como os dois sistemas têm um recetor GNSS instalado, é possível utilizar este como fonte de sincronismo. Utilizando o sinal PPS e a trama enviada pelo recetor GNSS como fontes de sincronismo para o chrony, é possível obter erro temporal entre o disparo dos dois sistemas na escala do micro-segundo (tabela 5.2).

Tabela 5.2: Caracteristicas de sincronismo do sistema desenvolvido

	$\Delta t_{Trigger1Trigger2}$ (μ S)	Frequência Trigger 1 (Hz)	Frequência Trigger 2 (Hz)
μ	2.93	29.99993	29.99967
σ	19.46	0.01633	0.01828
N	12879	8586	8586

A utilização de fontes de sincronismo como PPS ou mensagens RAW por parte do chrony não é estanque, necessitando de outras aplicações para que tal seja possível, nomeadamente o linuxPPS [43] e o gpsd [44].

Uma vez que os dois computadores têm o seu tempo de sistema sincronizado com o tempo GPS, é possível efetuar o disparo da câmara e dos LED utilizando como referência o tempo de sistema. Para tal, foi desenvolvida uma aplicação de disparo cujo seu algoritmo pode ser reduzida à condição exposta no algoritmo 1. A variável $CurrentTime$ corresponde ao tempo atual do sistema, $Time_{PreviousTrigger}$ é o tempo do sistema quando foi efetuado o último disparo, $Trigger_Period$ é o período do sinal de disparo desejado. Para garantir que o erro cumulativo, induzido pelo cálculo do tempo entre períodos, não crie o desfasamento, a cada segundo é forçado um disparo (condição $CurrentTimeisanewSecond$).

Algorithm 1 Disparo sincronizado

if $CurrentTime - Time_{PreviousTrigger} >= Trigger_Period$ **or** $CurrentTimeisanewSecond$ **then**
  TRIGGER
  $Time_{PreviousTrigger} = CurrentTime$
end if

Assumindo as caraterísticas elétricas do díodo e o desvio padrão do erro entre os dois sinais de disparo, é conveniente atrasar ligeiramente o disparo da câmera, como

apresentado na figura 5.6, face ao disparo dos LED, garantindo assim que todos os instantes da imagem captam o LED completamente acesso.

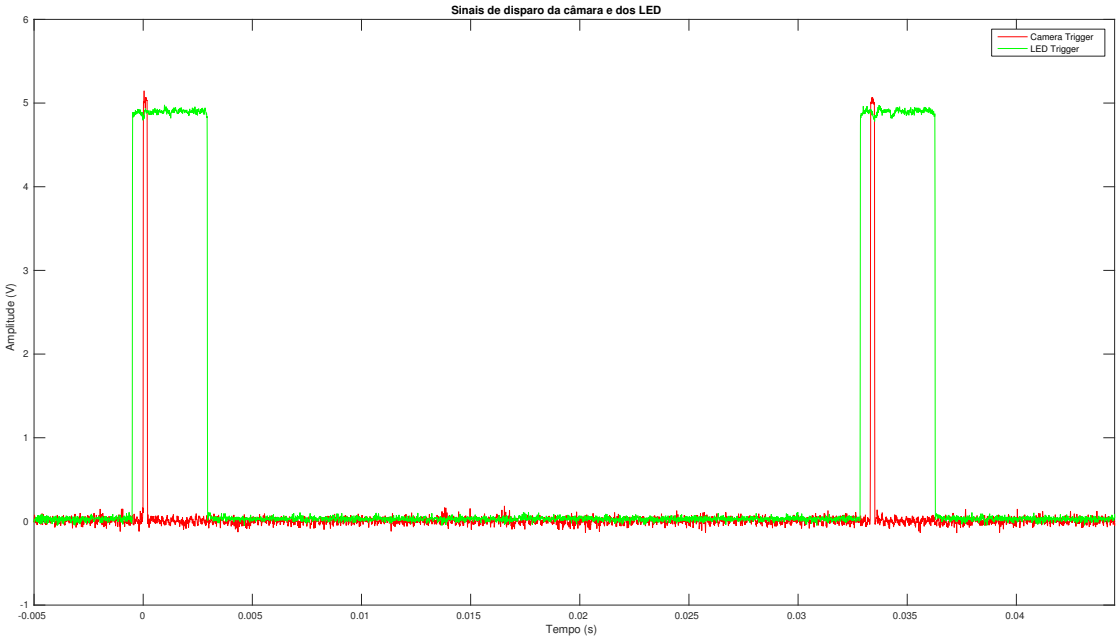


Figura 5.6: Sinais de Trigger durante o normal funcionamento do sistema

5.4 Integração do Recetor GNSS no Sistema Computacional

O recetor GNSS utilizado apenas disponibiliza uma porta série, o que se revela insuficiente para a aplicação desejada, uma vez que as aplicações que garantem o sincronismo e a posição RTK, respetivamente o RTKLib e o chrony com recurso ao gpsd, necessitam de receber as mensagens *RAW*, como representado no diagrama da figura 5.7.

Assumindo a necessidade de enviar a mesma *stream* de dados para duas aplicações distintas, foi implementada uma solução com recurso a um circuito elétrico, garantindo assim que não se aumenta a carga no sistema computacional. O circuito desenvolvido, apresentado na figura 5.8 permite a divisão de uma porta série por três clientes, possibilitando assim que o mesmo recetor GNSS possa ser utilizado pelas duas aplicações e ainda por um terceiro diapositivo, caso seja necessário.

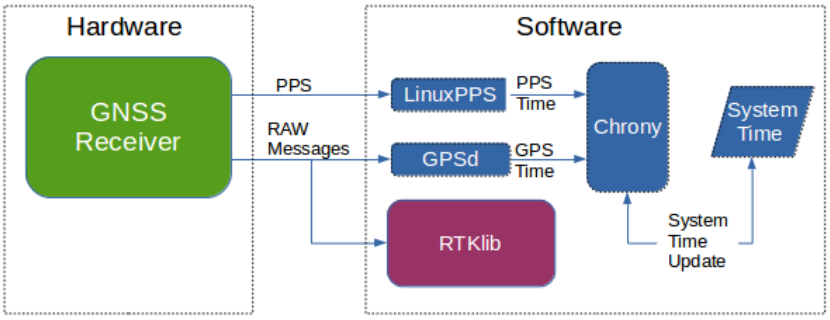


Figura 5.7: Arquitetura das aplicações que utilizam a informação enviada pelo recetor GNSS

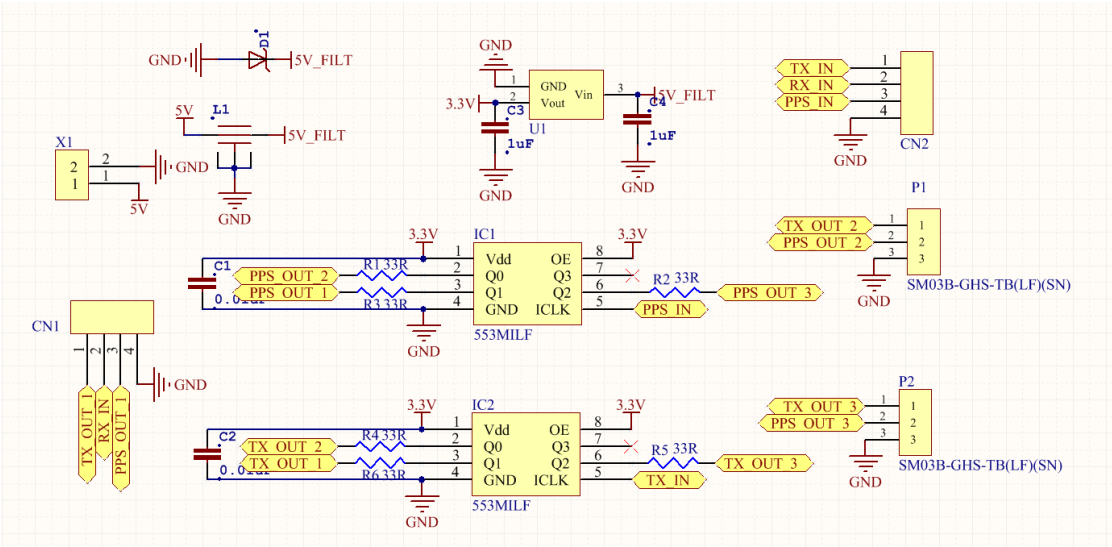


Figura 5.8: Esquema elétrico da placa de replicação da porta série do recetor GNSS

5.5 Pista de Aterragem

A pista de aterragem desenvolvida, apresentada na figura 5.9, é composta por dois elementos:

- Base - Suporta o marcador visual ativo que identifica o local de aterragem para o UAV;
- Unidade Computacional - Responsável por ser base RTK, enviar os dados do sensor inercial para o UAV e disparar os LED.

A base da pista tem as dimensões de 1.2x1.2m, o padrão exterior tem 88x88cm e o padrão interior 20x20cm, permitindo assim ao UAV aterrar sem que o seu trém de

aterragem ocluda algum dos LED que compõem o padrão. A plataforma é composta no seu interior por uma placa de poliestireno extrudido, já o seu exterior é composto por duas placas de poli-carbonato, garantindo assim a rigidez e peso reduzido da estrutura.

Figura 5.9: Pista de aterragem desenvolvida

Na figura 5.10 está apresentada a arquitetura do sistema da pista de aterragem. A unidade de controlo da pista de aterragem é composta por um computador ARM Odroid XU3, um recetor GNSS U-BLOX NEO-M8T, um autopilot Pixhawk servindo de IMU e a placa de controlo dos LED que compõem o padrão.

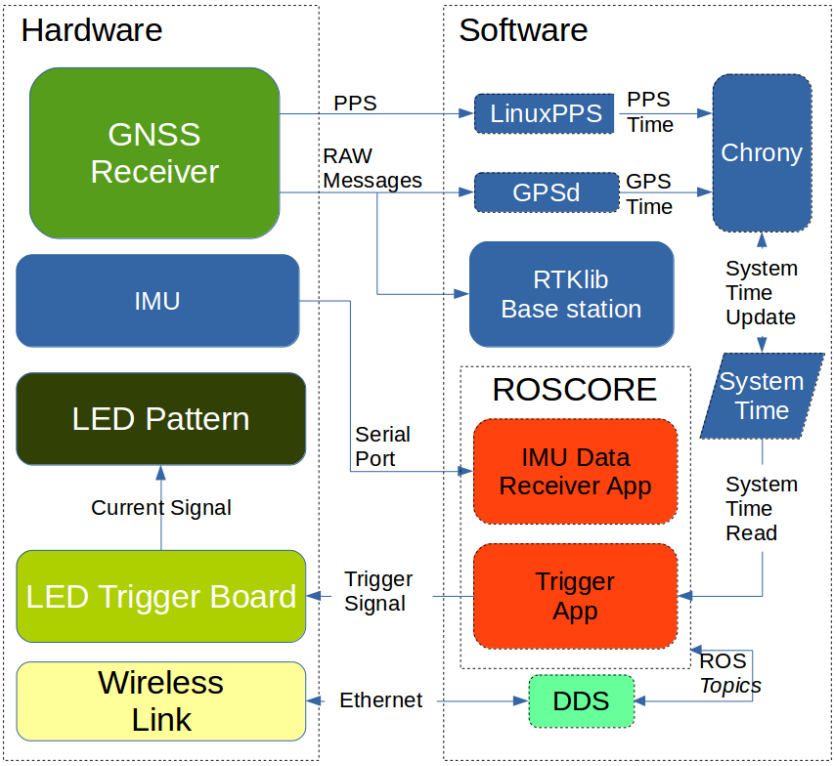


Figura 5.10: Arquitetura da pista de aterragem

5.6 VTOL

O UAV, utilizado na manobra, exposto na figura 5.11, tem como base a plataforma S900 do fabricante DJI. A plataforma é um hexa-rotor com uma capacidade de carga (*payload*) de 3.7Kg e um tempo de voo de 30 minutos com uma bateria LiPo (*Lithium Polymer*) de seis células com 22000mAh de capacidade. Para além do hardware necessário para garantir o controlo de voo, está também equipado com uma Odroid XU3 e uma câmara de espetro visível Pointgrey Chameleon de 1.3MP instalada a apontar para baixo.

Figura 5.11: UAV OTUS

Na figura 5.12 é apresentada a arquitetura de alto nível das componentes de *hardware* e *software* do UAV. Tal como a pista de aterragem, o UAV está equipado com um U-BLOX NEO-M8T que tem como função enviar mensagens *RAW* que são utilizadas pelo RTKlib e pelo chrony.

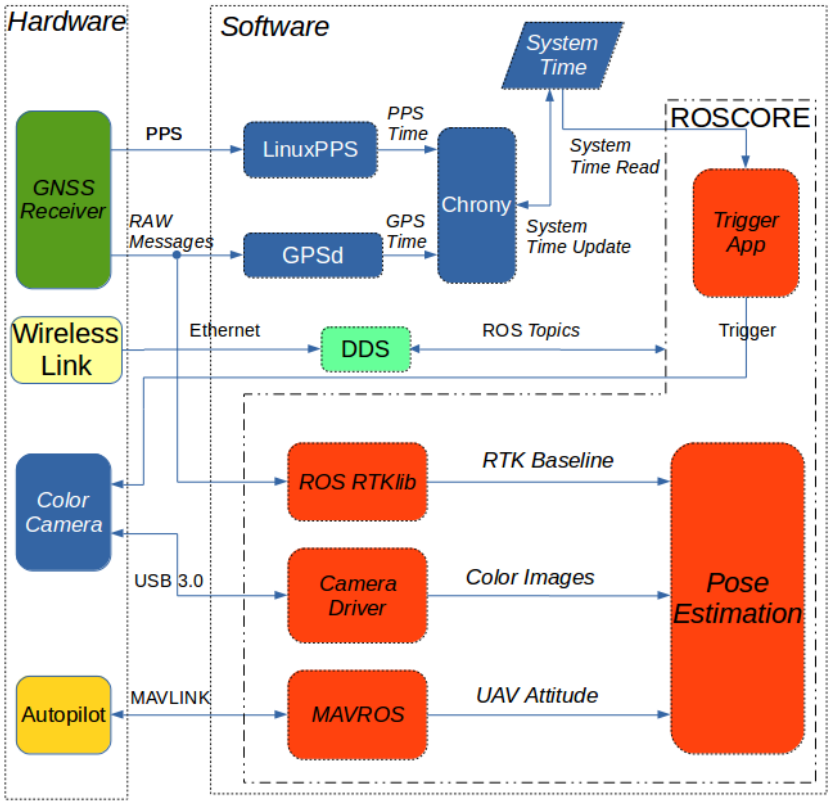


Figura 5.12: Arquitetura de alto nivel do sistema de aterragem autónoma do UAV OTUS

5.7 Visão

A câmara instalada no UAV OTUS, apenas necessita de detetar os pontos brilhantes gerados pelos LED. Para filtrar passivamente os restantes elementos da imagem, a lente instalada na câmera está fechada quase totalmente, garantindo assim que apenas pontos com brilho elevado aparecem na imagem.

A obtenção da posição da pista de aterragem face ao UAV utilizando as imagens obtidas pela câmera pode ser divida em três fases distintas representadas na figura 5.13:

- Deteção dos LED - Consiste na procura por pontos brilhantes na imagem cuja cor corresponde à cor dos LED;
- Associação dos LED - Fase em que os LED são identificados segundo o padrão;
- Calculo da *pose* - Com a informação sobre a posição no plano da imagem dos pontos do padrão, é calculada a posição relativa entre o UAV e a pista de aterragem.

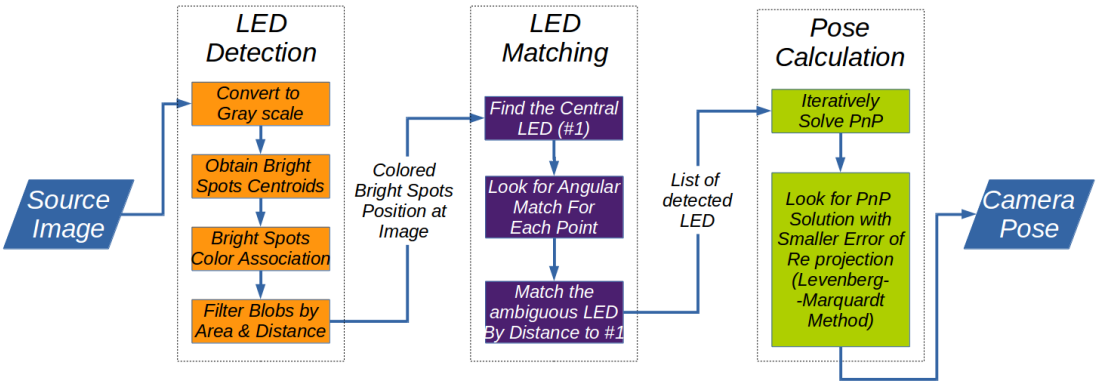


Figura 5.13: Pipeline do sistema de posicionamento utilizando visão

A imagem apresentada na figura 5.14 corresponde à imagem da pista de aterragem capturada a 5 metros de altura às 14 horas de um dia com céu limpo do mês de Junho. Devido às características refletoras da placa de poli-carbonato utilizada existe uma zona brilhante grande que corresponde à luz emitida pelo Sol, que em conjunto com o facto da câmera usar um sensor CCD (*Charge-coupled Device*) levam ao aparecimento de riscas verticais denominadas de *vertical smear*.

Na primeira fase do processamento de imagem, a imagem é filtrada para que apenas os pontos brilhantes sejam considerados, resultando na imagem do lado direito da figura 5.15.

Nos pontos brilhantes identificados são procuradas as cores verde e vermelha que constituem o padrão. A zona brilhante resultante do reflexo do Sol é eliminada nesta fase, uma vez que não tem a cor pretendida, como é possível observar na figura 5.16.

Nesta fase do processamento existem vários falsos positivos provados por vários fatores, como as reflexões criadas pelo poli-carbonato ou a segmentação de um *blob* em vários. Estes *blobs* têm como caraterísticas uma área bastante reduzida em relação aos restantes (figura 5.17 esquerda), ou estão muito próximos de um *blob* grande (5.17 direita).

Utilizando como informação a cor dos *blobs* e a sua posição no plano da imagem é então possível atribuir-lhes o identificador relativo à sua posição no padrão definido, apresentado na figura 5.4. A abordagem utilizada para a identificação do padrão consiste na utilização do ângulo dos vetores formados por pares dos pontos detetados, como representado na figura 5.18.

Pontos distintos do padrão têm relações angulares distintas, pelo que utilizando estas

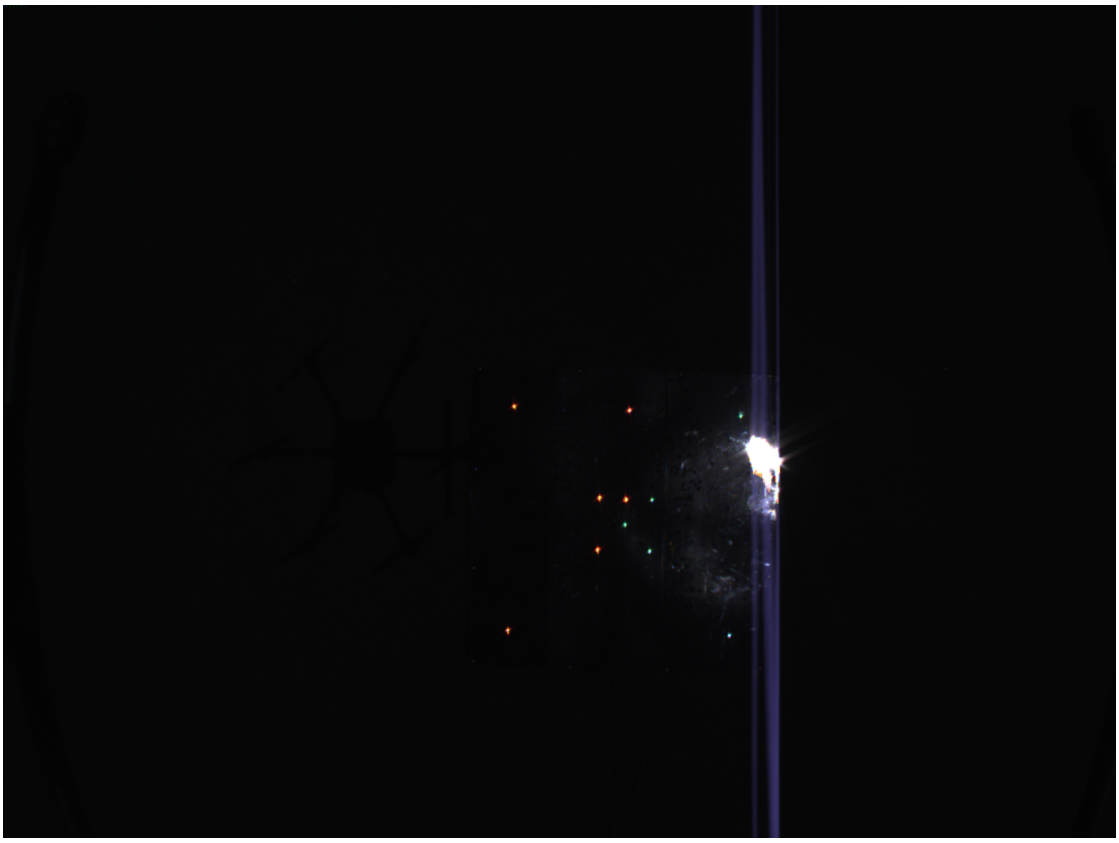


Figura 5.14: Imagem capturada pela câmera instalada no UAV

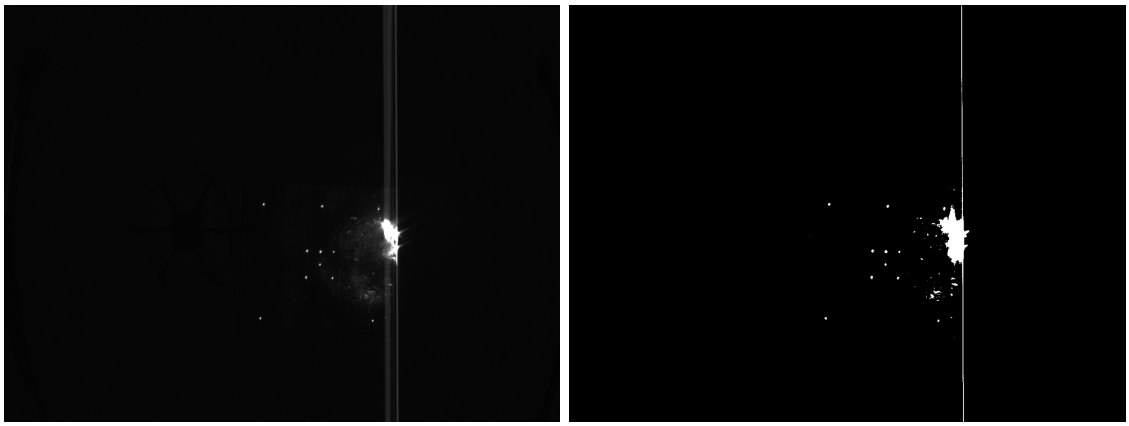


Figura 5.15: Primeira fase do processamento de imagem. Esquerda - Imagem convertida para escala de cinza. Direita - Resultado do *threshold* aplicado para obter os pontos brilhantes da imagem

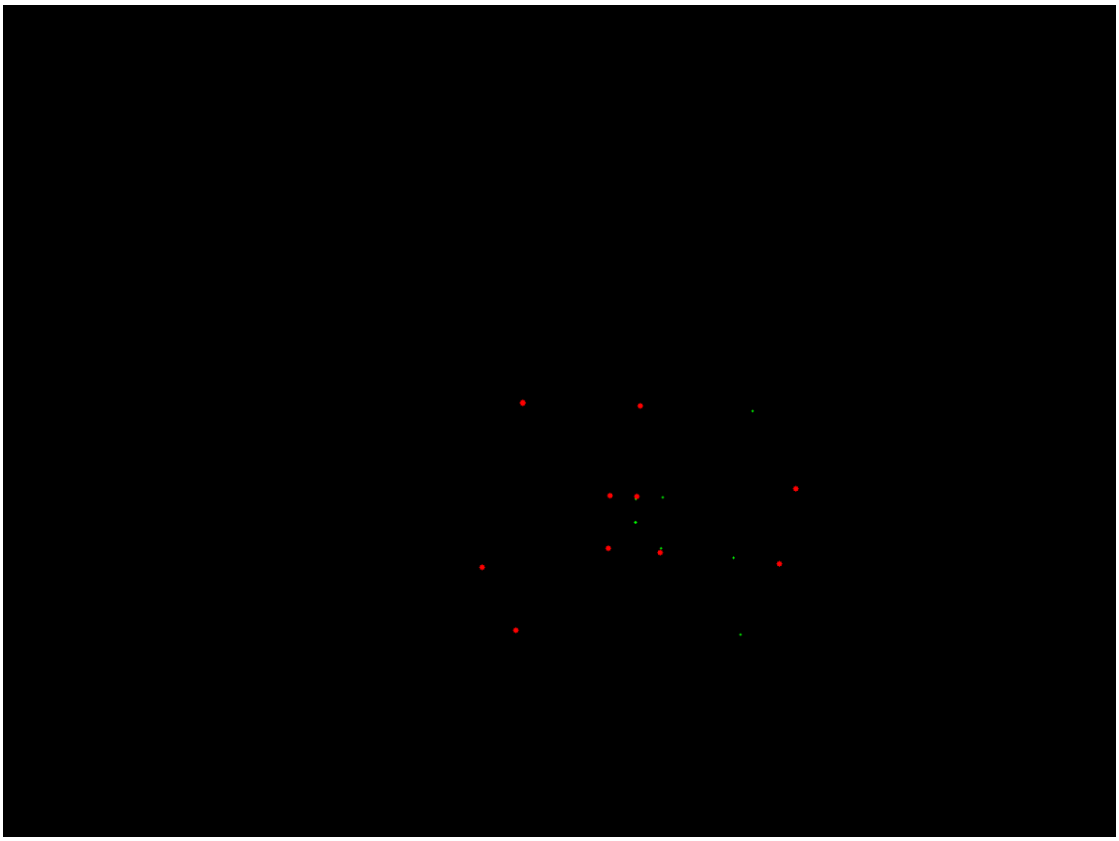


Figura 5.16: Pontos brilhantes com cor verde ou vermelha

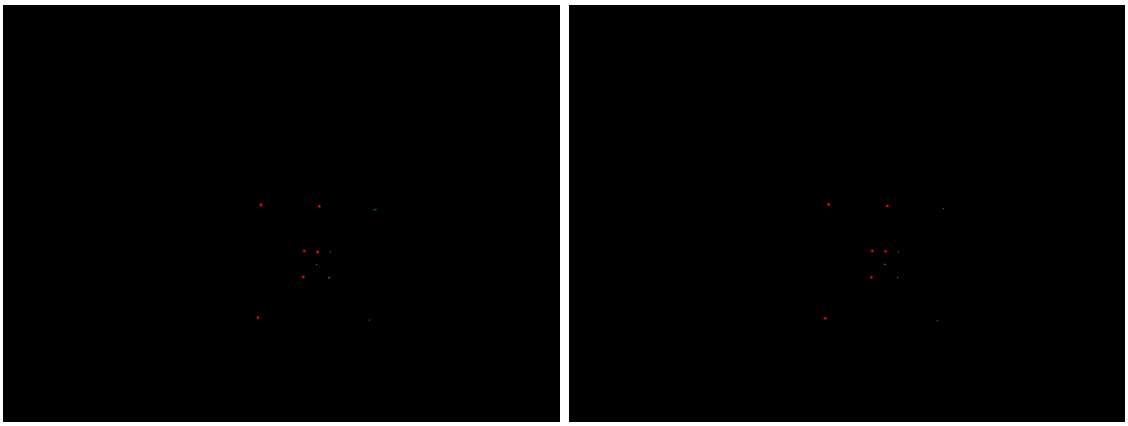


Figura 5.17: Processo de associação de cor aos pontos brilhantes. Os pontos associados são filtrados segundo a sua área (imagem central), e por fim segundo a sua distância relativa (imagem da direita)

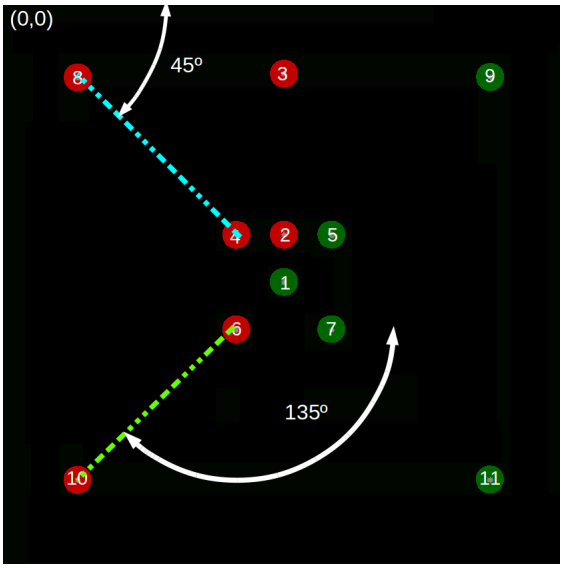


Figura 5.18: Ângulo das linhas geradas pelos pares de pontos 4-8 e 6-10

é possível identificar os pontos do padrão. Para que a identificação do padrão seja possível, a deteção do ponto central, o ponto 1, é fundamental, uma vez que serve de referência para as restantes relações angulares. Na deteção do ponto 1 são procuradas as diferenças entre *blobs* que correspondem às que o caracterizam.

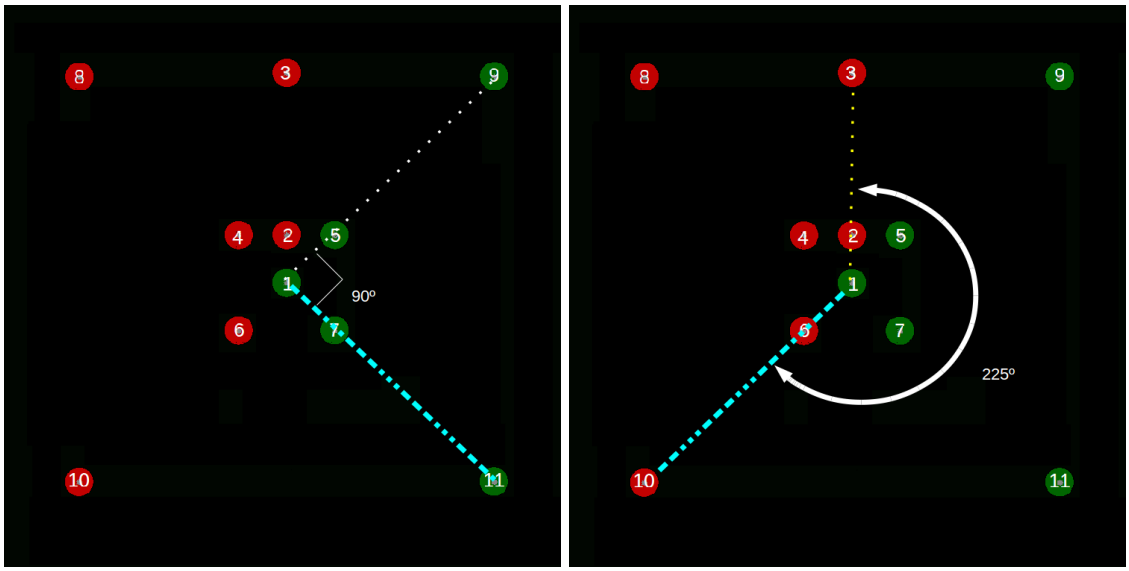


Figura 5.19: Exemplo de alguns ângulos utilizados para a deteção do ponto 1

O ponto com o maior número de correspondências angulares é considerado o ponto 1,

que será utilizado como referência para a identificação dos restantes pontos. Enquanto que na identificação dos ponto 1 são utilizadas linhas formadas por todos os pares de pontos, para identificação dos restantes dez pontos são definidas linhas em que um dos elementos é o ponto 1, como representado no exemplo para os pontos 2 ou 3 na figura 5.20.

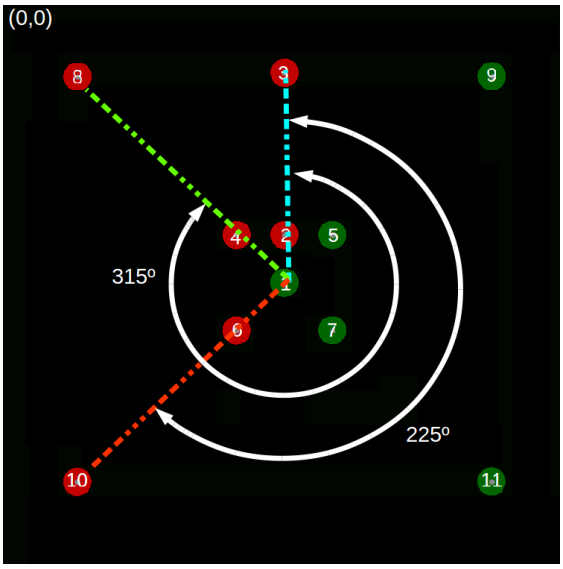


Figura 5.20: Ângulos entre linhas formadas por pontos vermelhos.

O padrão definido, como referido anteriormente, é composto por dois grupos de pontos, o interior e o exterior, sendo que a diferença entre estes é unicamente o seu tamanho. Esta característica leva a que para cada reta definida entre o ponto 1 e um dos restantes, seja colinear com uma outra reta (i.e reta definida pelos pontos 1 e 2 com a reta definida pelos pontos 1 e 3), o que leva a que a abordagem angular não seja suficiente para identificar os pontos. Como solução para a ambiguidade da abordagem angular, é utilizada a distância entre os pontos e o ponto 1.

Utilizando as duas abordagens as relações angulares e a distância ao centro do padrão, é então possível identificar todos os pontos do padrão, desde que o ponto central seja detetado, como exposto na figura 5.21 à direita.

Utilizando as coordenadas dos pontos do padrão no plano da imagem e sabendo a sua posição, é possível obter a posição relativa do UAV face à pista de aterragem, utilizando o método iterativo com recurso à biblioteca Opencv 3.1.2 para resolver o problema *Perspective-N-Points*.

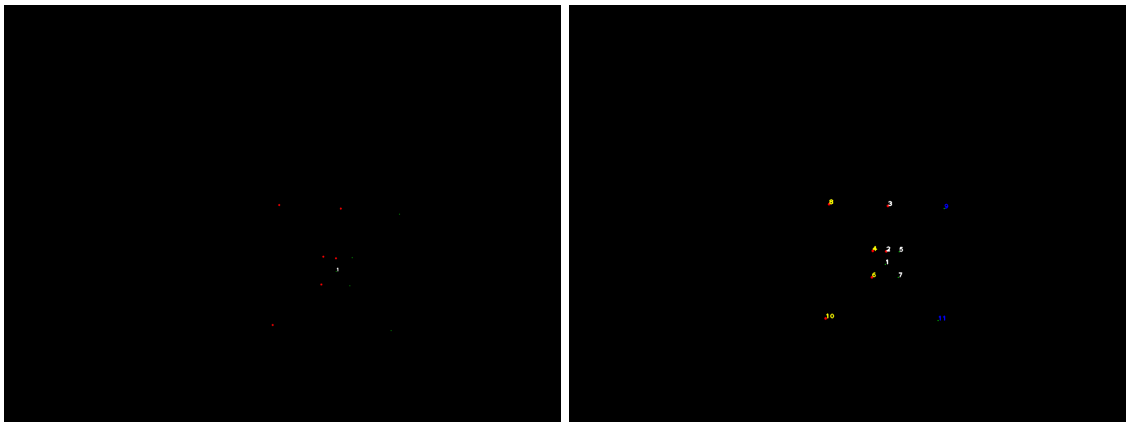


Figura 5.21: Ponto central detetado (esquerda). Utilizando esse ponto, são associados os restantes pontos (direita)

5.8 Ensaio

O algoritmo de posicionamento relativo com recurso a visão, bem como a implementação de GPS RTK foram testados utilizando um *dataset* com os seguintes dados:

- *Pose* do OTUS UAV;
- Posição estimada pelo *autopilot* com recurso aos seus sensores inerciais e GPS;
- *Pose* da pista de aterragem;
- *Baseline* RTK entre o OTUS e a pista de aterragem;
- Imagens capturadas pela câmera instalada no UAV.

O *dataset* tem informação relativa a dois voos efetuados em que o UAV atingiu uma altitude relativa de aproximadamente dez metros. Utilizando a informação armazenada, foi possível comparar os resultados obtidos através da visão computacional com os resultados de referência.

O *autopilot* utilizado, Pixhawk, com o *firmware* PX4 [45], é capaz de calcular uma solução de posição local utilizando a informação da posição GPS e dos sensores inerciais. Posto isto, tal como a posição obtida através da visão computacional, a posição estimada pelo *autopilot* é comparada com a posição obtida através do *RTKLib*, que servirá de referência.

A informação relativa à trajetória efetuada durante os voos está exposta no gráfico da figura 5.22. Nestes gráficos estão representadas as componentes de X,Y e Z da posição

ao longo do tempo. Uma vez que a distância do marcador à câmera influência bastante a qualidade dos dados obtidos através de visão computacional, os períodos de voo em que o OTUS está a altitude superior a quatro metros estão identificados com o fundo vermelho nos gráficos, em contraste com o fundo verde para altitudes inferiores. A posição do UAV calculada com recurso à visão é diferenciada segundo o número de pontos do padrão detetados, cor verde para um número de pontos do padrão detetados igual ou superior a nove e vermelho para menos de nove. Para além da informação relativa à posição obtida com recurso a visão computacional, nos gráficos está também representada a posição obtida através da estimação do *autopilot*, bem como as componentes da *baseline* obtida pelo *RTKLib*, que corresponde também à posição relativa entre a pista de aterragem e o UAV.

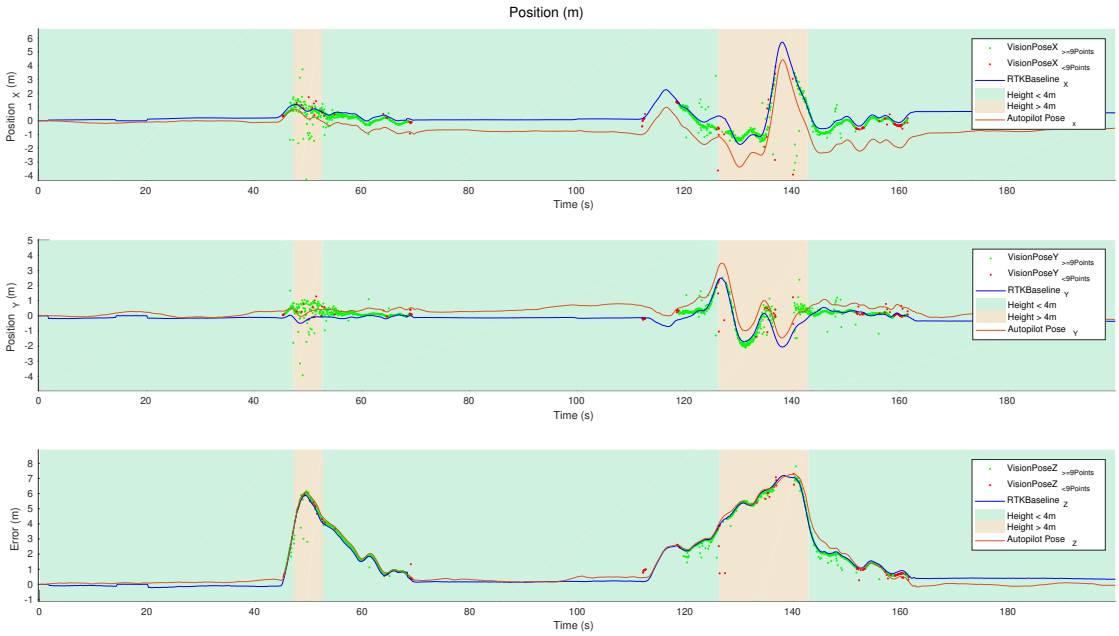


Figura 5.22: Distância entre o UAV e a pista ao longo do tempo

Analisando o erro de posição ao longo do tempo para a posição obtida através da visão computacional, bem como a fornecida pelo *autopilot*, em relação à obtida pelo RTKLib, apresentado na figura 5.23, permite à partida retirar algumas conclusões quanto à qualidade da posição fornecida pelos dois métodos. A posição calculada pelo *autopilot* apresenta um erro superior à calculada com recurso à visão computacional. Este fenómeno deve-se ao facto de o *autopilot* utilizar acelerómetros que devido a erros constantes de medidas levam a que a posição obtida sofra de um erro comulativo, como é possível observar nas curvas das componentes X e Y que após o primeiro voo começam

a divergir. A componente de Z, apresenta também divergência, que é possível verificar comparando a altitude quando o UAV está na pista de aterragem, antes e após o segundo voo.

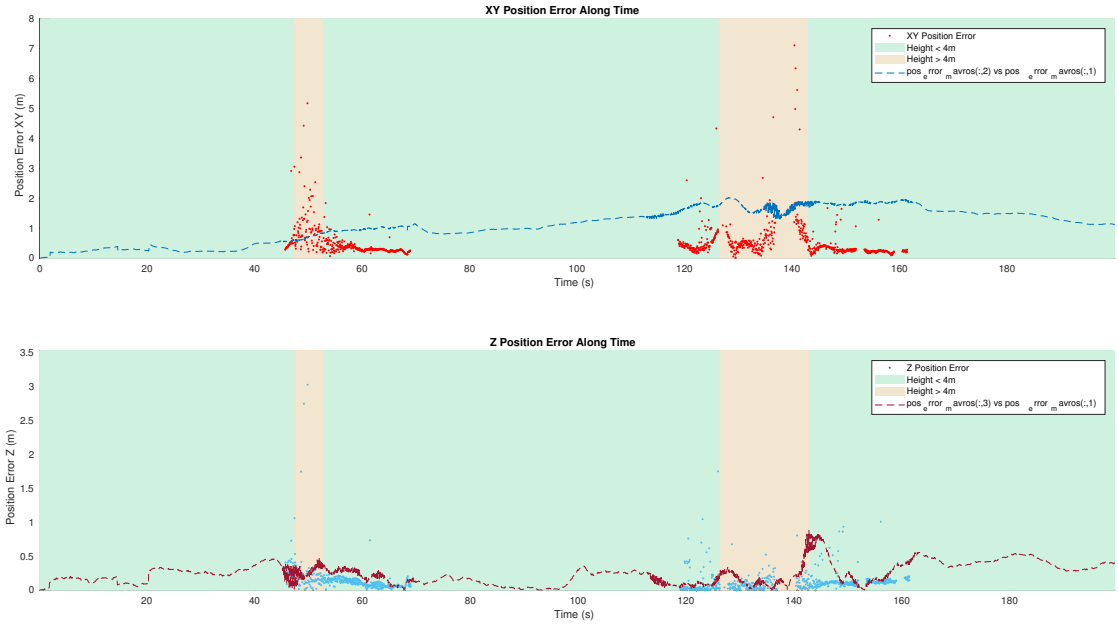


Figura 5.23: Erro da posição calculada utilizando visão

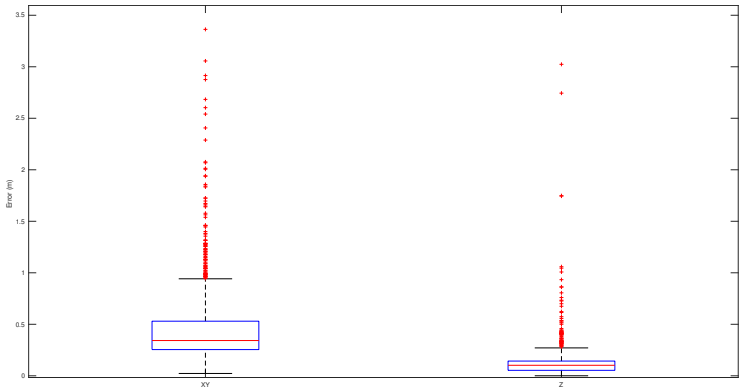
A dispersão do erro obtido através da visão computacional e da posição estimada pelo *autopilot* estão apresentadas na figura 5.24, tem as características apresentadas na tabela 5.3.

Tabela 5.3: Características do erro de posicionamento em relação à posição GPS RTK

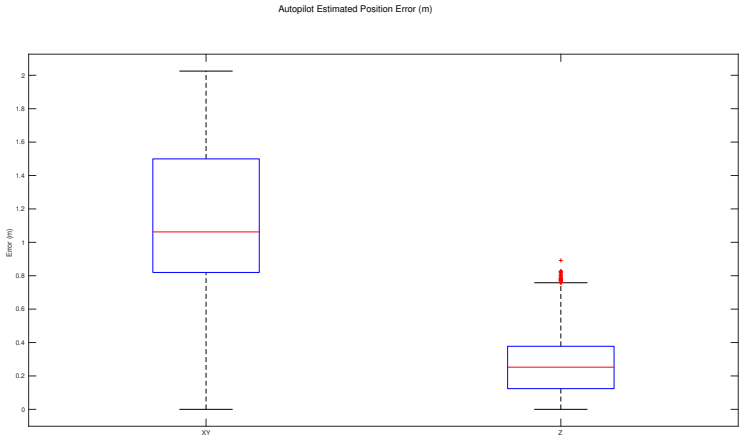
Método	Visao		Autopilot	
Componentes da Posição	XY	Z	XY	Z
Valor Médio (m)	0.34234	0.10206	1.0623	0.2595
Intervalo Interquartil (IIQ) (m)	0.57563	0,089968	0.67988	0.25363
Número de Medidas Descartadas	114 de 1115	74 de 1115	0 de 6119	30 de 6119

Utilizando visão também é possível obter a atitude do UAV em relação à pista de aterragem, tendo sido obtidos os resultados apresentados na figura 5.25, sendo utilizada a atitude obtida pelo *autopilot*, como referência.

O erro das três componentes da atitude, *roll*,*pitch* e *yaw*, apresentado na figura 5.26, tem a dispersão representada no diagrama da figura 5.27 com as características apresen-



(a) Distribuição do erro da posição obtida com recurso à visão computacional



(b) Distribuição do erro da posição fornecida pelo *autopilot*

Figura 5.24: Distribuição do erro de posição.

tadas na tabela 5.4.

Tabela 5.4: Características do erro de atitude em relação à informação inercial

	Roll	*Pitch*	*Yaw*
Valor Médio (°)	1.6384	1.8238	4.2237
IIQ (°)	2.0339	1.4406	2.469
Número de Medidas Descartadas	93 de 1115	115 de 1115	30 de 1115

De forma a quantificar o desempenho do método de deteção do marcador visual, foram selecionados segmentos do voo em que o UAV se encontra com diferentes posições e orientações em relação à pista de aterragem. Os seis segmentos escolhidos têm as

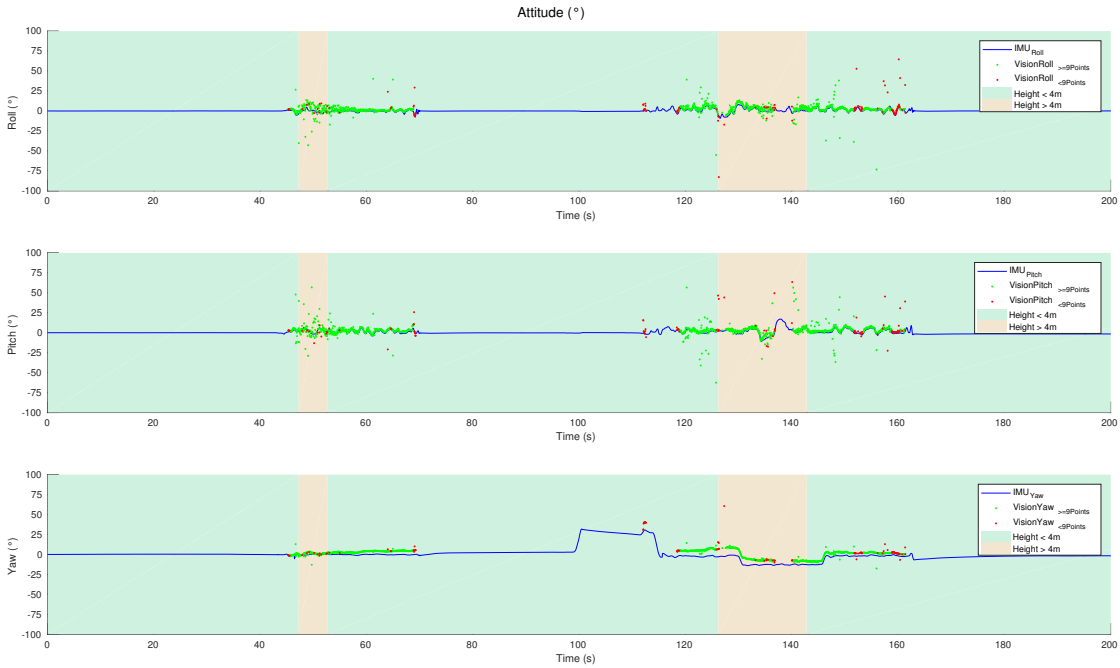


Figura 5.25: Atitude do UAV

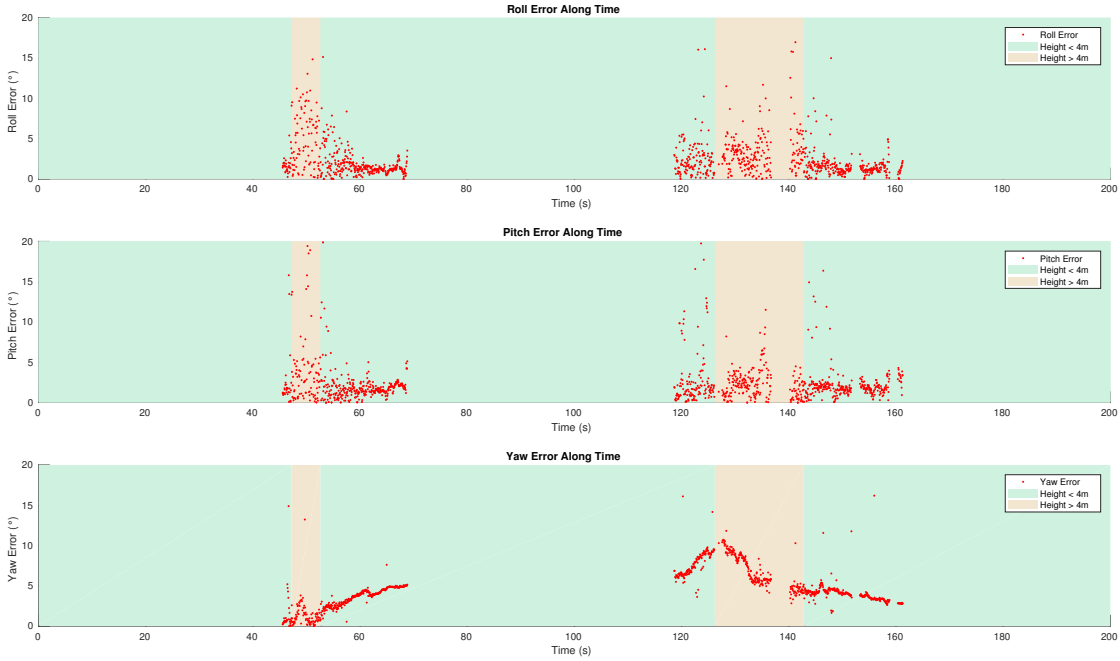


Figura 5.26: Erro da atitude calculada utilizando visão

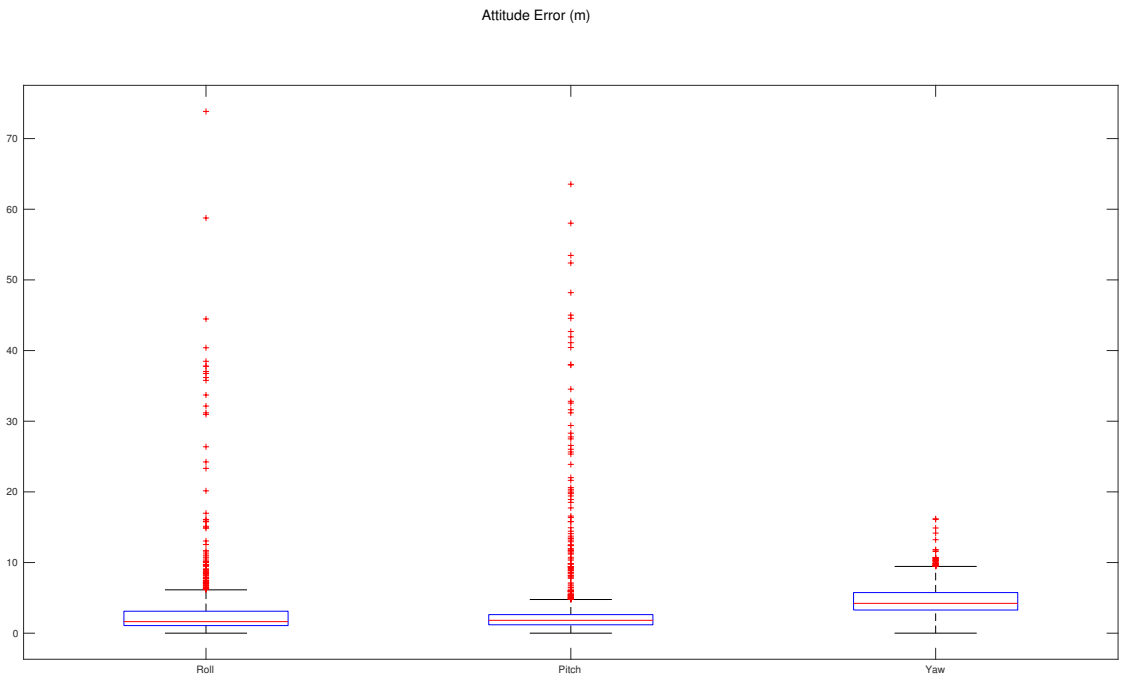
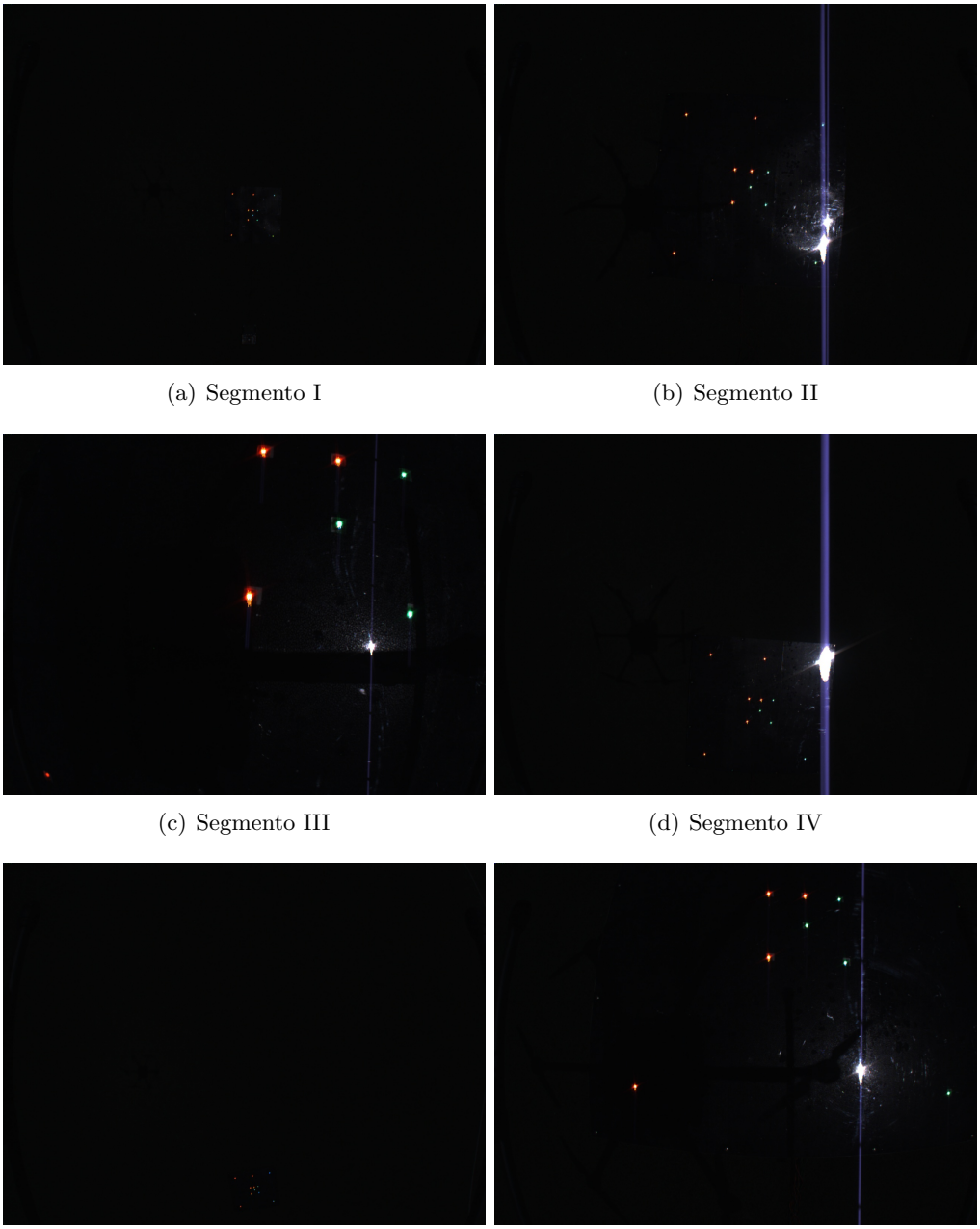


Figura 5.27: Distribuição do erro de atitude utilizando visão

seguintes características:

- Segmento I - Altitude de seis metros, todos os pontos do padrão estão enquadrados na imagem;
- Segmento II - Altitude inferior a quatro metros, todos os pontos do padrão estão enquadrados na imagem, o efeito do Sol e do sensor CCD;
- Segmento III - Altitude reduzida, apenas os pontos do grupo interior são sempre enquadrados;
- Segmento IV - O ponto 9 é saturado pela luz do reflexo do Sol;
- Segmento V - Altitude elevada;
- Segmento VI - Altitude reduzida e alguns pontos dos grupos interior e exterior são enquadrados ;

Na figura 5.28, são apresentados exemplos das imagens detetadas em cada um dos segmentos selecionados.



(a) Segmento I (b) Segmento II

(c) Segmento III (d) Segmento IV

(e) Segmento V (f) Segmento VI

Figura 5.28: Exemplo de imagens obtidas em cada um dos segmentos selecionados

As coordenadas dos pontos do padrão no plano da imagem foram retirados para as imagens dos segmentos selecionados, servindo de referência para quantificar a *performance* do método desenvolvido. O erro da deteção dos pontos do padrão no plano da

imagem é quantificado utilizando a distância euclidiana entre o ponto detetado automaticamente e o ponto detetado manualmente, na figura 5.29 é apresentado o erro na obtenção da posição do ponto do padrão nas imagens dos segmentos selecionados.

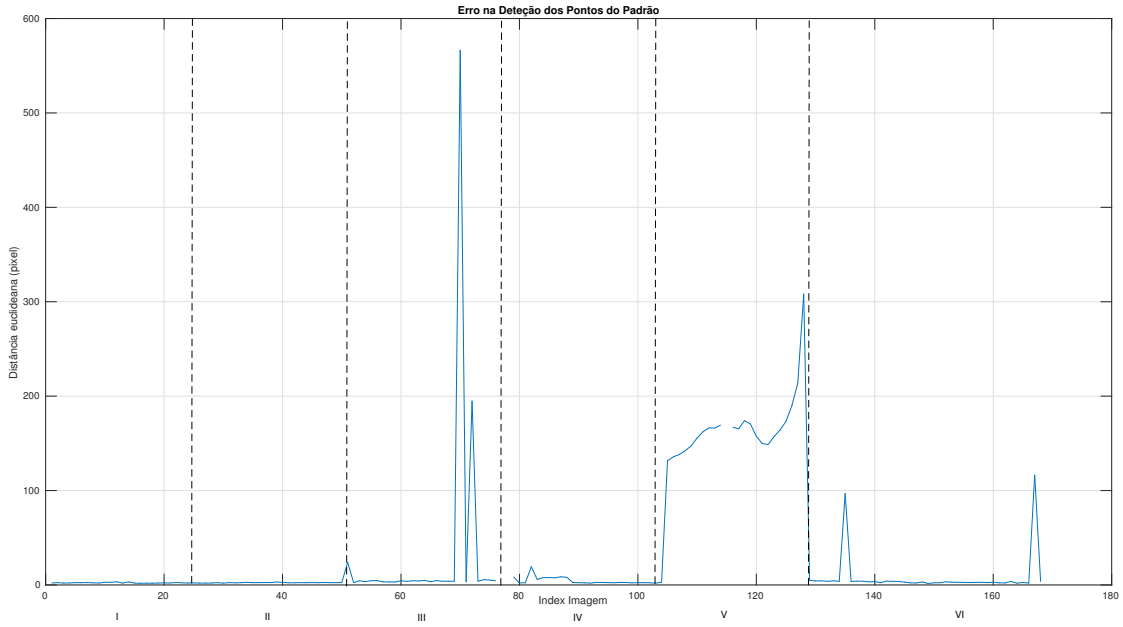


Figura 5.29: Erro médio em pixel da deteção dos pontos do padrão

A distribuição do erro médio da deteção do ponto no plano da imagem, apresentado no diagrama de caixa da figura 5.30, tem as características descritas na tabela 5.5

Tabela 5.5: Características do erro da deteção dos pontos do padrão na imagem

Valor Médio	2.79 pixel
IIQ	2.2 pixel
Número de Medidas Descartadas	32 de 165

O método desenvolvido, para as imagens selecionadas, apenas não detetou corretamente 163 de 1699 pontos, resultando numa taxa de 90%, sendo que o número de pontos detetados, bem como o número de pontos existentes na imagem, são apresentados no gráfico da figura 5.31,

Utilizando a informação relativa à posição dos pontos do padrão na imagem, foi calculada a *pose* da câmera. De forma a confirmar a correta obtenção da *pose*, os pontos do padrão são projetados no plano da imagem, utilizando a *pose* calculada inicialmente. A distância euclidiana entre os pontos do padrão detetados na imagem e os pontos do

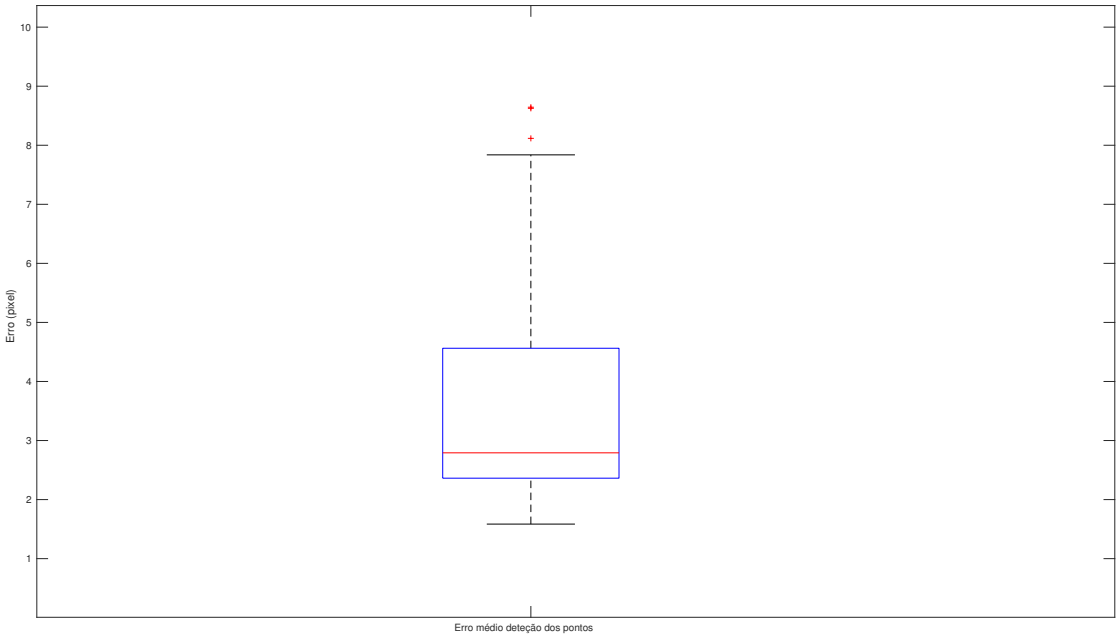


Figura 5.30: Distribuição do erro médio na deteção dos pontos do padrão

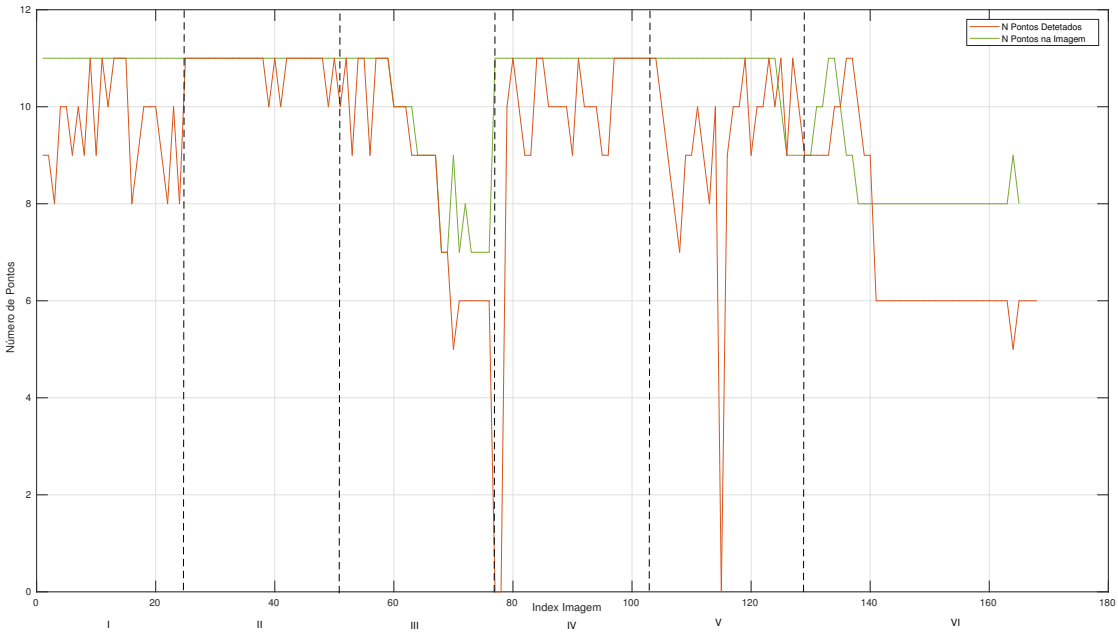


Figura 5.31: Número de pontos do padrão detetados ao longo do tempo

mundo projetados, é denominado de erro de reprojeção. O erro de reprojeção para os pontos obtidos manualmente, tal como para os pontos obtidos automáticamente para

cada imagem, está apresentado na figura 5.32.

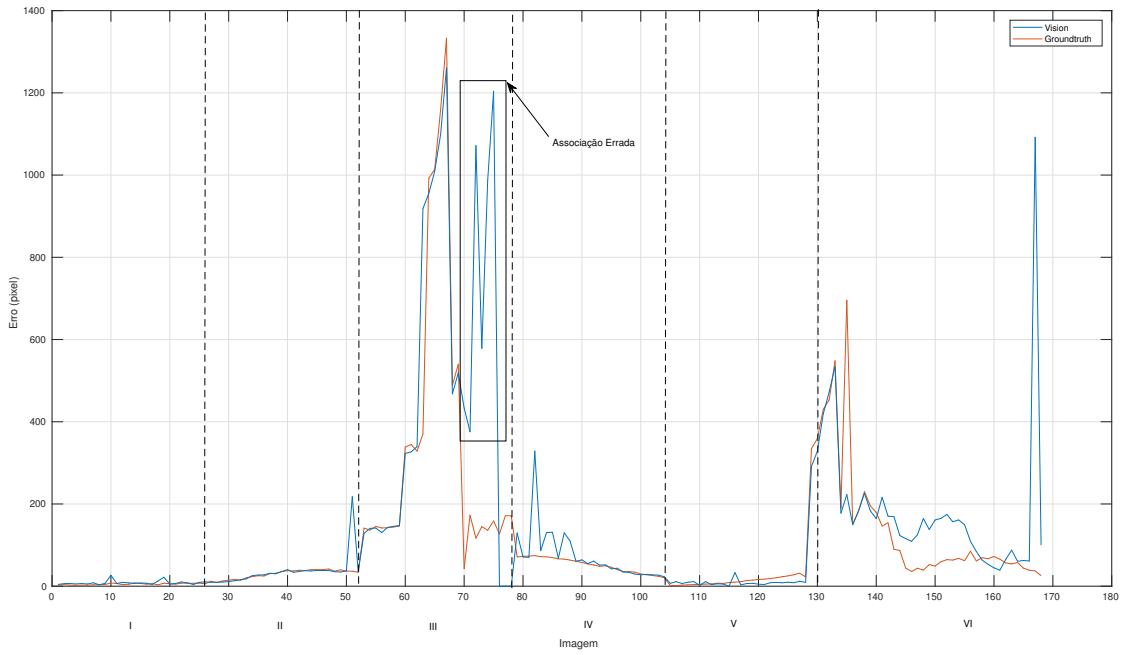


Figura 5.32: Erro da reprojeção dos pontos no plano da imagem

A zona assinalada com o retângulo na figura 5.32, corresponde a uma zona em que o algoritmo de associação dos pontos do padrão falhou. Como é possível observar na figura 5.33, segundo o padrão definido (figura 5.4) o ponto 7 não foi detetado e o ponto 9 foi detetado, erradamente, na posição do ponto 5. Esta zona corresponde também à medida em que existe o maior erro em pixel na deteção dos pontos do padrão (figura 5.29). A comparação da dispersão do erro de reprojeção para os pontos obtidos através dos dois métodos com as caracteristicas apresentadas na tabela 5.6 é representada na figura 5.34.

Tabela 5.6: Características do erro de reprojeção

	Groundtruth	Automático
Valor Médio (px)	39.4127	40.2351
IIQ (px)	66.5171	137.1333
Número de Medidas Descartadas	20 de 168	17 de 168

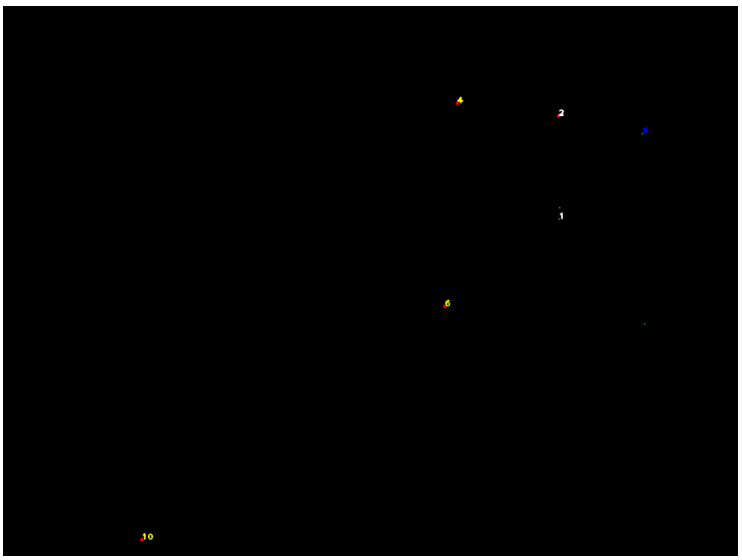


Figura 5.33: Imagem em que ocorreu o erro de associação dos pontos no plano da imagem

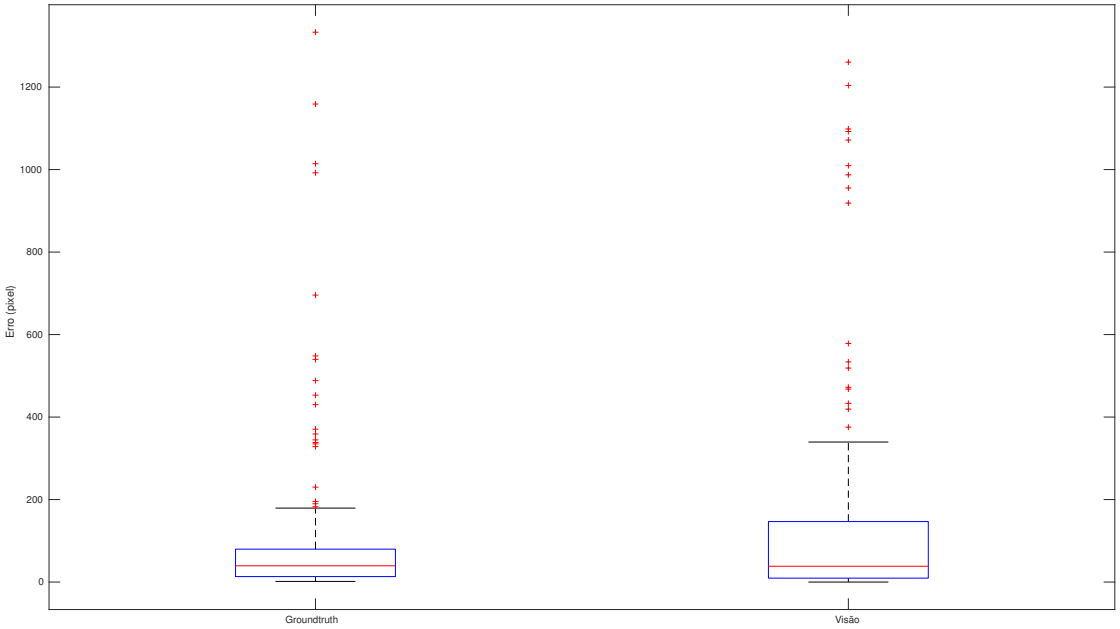


Figura 5.34: Dispersão do erro da reprojeção dos pontos no plano da imagem

Esta página foi intencionalmente deixada em branco.

Capítulo 6

Conclusão e Trabalho Futuro

Esta dissertação aborda o desenvolvimento de um sistema de apoio ao processo de aterragem autónoma de um VTOL com recurso a um sistema de visão e um sistema de localização GPS RTK.

De modo a garantir a robustez da solução às condições de operação, desenvolveu-se uma pista de aterragem com um marcador ativo com disparo sincronizado entre a câmera a bordo do VTOL e os LED da pista. A solução revelou ser uma abordagem inovadora face ao estado da arte e com resultados interessantes no que respeita à robustez face à variação de luminosidade.

Com recurso às imagens do marcador visual, adquiridas pelo sistema de visão monocular instalado no VTOL, é possível obter a posição relativa entre o VTOL e a plataforma de aterragem. Através da implementação de GPS RTK de baixo custo foi possível obter uma solução de *baseline* móvel.

Nos testes apresentados, tanto os dados provenientes do RTK como a informação inercial, serviram apenas de *ground truth* para os resultados obtidos através de visão computacional, pelo que no futuro é fundamental implementar a estimação da posição com recurso aos dados provenientes destas fontes, permitindo obter a *pose* da pista de aterragem com maior robustez.

O algoritmo desenvolvido para a deteção do padrão do marcador, necessita da deteção do ponto central para poder identificar os restantes pontos. Pelo que é fundamental, implementar mecanismos de deteção para confirmar a a deteção do padrão, robustecendo assim o algoritmo.

O método desenvolvido deverá ser testado no ASV em condições adversas (i.e nevoeiro, diferentes ângulos de incidência do sol).

A tecnologia desenvolvida no âmbito da dissertação resultou na publicação de quatro

artigos em conferências internacionais [10] [11] [46] [47] e um artigo em revista [48].

Bibliografia

[1] F. Cocchioni, A. Mancini, and S. Longhi. Autonomous navigation, landing and recharge of a quadrotor using artificial vision. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 418–429, May 2014.

[2] S. Yang, S. A. Scherer, K. Schauwecker, and A. Zell. Onboard monocular vision for landing of an mav on a landing site specified by a single reference image. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 318–325, May 2013.

[3] C. Bu, Y. Ai, and H. Du. Vision-based autonomous landing for rotorcraft unmanned aerial vehicle. In *2016 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pages 1–6, July 2016.

[4] W. Xiao-Hong, X. Gui-Li, T. Yu-Peng, W. Biao, and W. Jing-Dong. Uav's automatic landing in all weather based on the cooperative object and computer vision. In *2012 Second International Conference on Instrumentation, Measurement, Computer, Communication and Control*, pages 1346–1351, Dec 2012.

[5] A. Breitenmoser, L. Kneip, and R. Siegwart. A monocular vision-based system for 6d relative robot localization. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 79–85, Sept 2011.

[6] An introduction to gnss. `https://www.novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/gnss-measurements/`. Accessed: 2017-09-06.

[7] Sddc fundamentals: Ntp infrastructure design. `http://gosddc.com/fundamentals/ntp-infrastructure-design/`. Accessed: 2017-09-06.

[8] P. Katsigiannis, L. Misopolinos, V. Liakopoulos, T. K. Alexandridis, and G. Zalidis. An autonomous multi-sensor uav system for reduced-input precision agriculture

applications. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 60–64, June 2016.

[9] H. Silva, J. M. Almeida, F. Lopes, J. P. Ribeiro, S. Freitas, G. Amaral, C. Almeida, A. Martins, and E. Silva. Uav trials for multi-spectral imaging target detection and recognition in maritime environment. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–6, Sept 2016.

[10] P. Sousa, A. Ferreira, M. Moreira, T. Santos, A. Martins, A. Dias, J. Almeida, and E. Silva. Isep/inesc tec aerial robotics team for search and rescue operations at the eurathlon challenge 2015. In *2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 156–161, May 2016.

[11] J. Fialho F. Moreira J. Almeida A. Dias E. Silva M. Moreira T. Santos J. Formiga, J. Dinis. Field experiments in power line inspection with an unmanned aerial vehicle. In *24th International Conference on Electricity Distribution*, 2017.

[12] C. Yuan, Z. Liu, and Y. Zhang. Fire detection using infrared images for uav-based forest fire surveillance. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 567–572, June 2017.

[13] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis. The sherpa project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments. In *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–4, Nov 2012.

[14] Us department of defense (2005) unmanned aircraft systems roadmap 2005–2030. `https://fas.org/irp/program/collect/uav_roadmap2005.pdf`. Accessed: 2017-09-06.

[15] R. Smith, M. Self, and P. Cheeseman. Autonomous robot vehicles. pages 167–193, 1990.

[16] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '07, pages 1–10, Washington, DC, USA, 2007. IEEE Computer Society.

[17] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2564–2571, Washington, DC, USA, 2011. IEEE Computer Society.

[18] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[19] Y. Jung, H. Bang, and D. Lee. Robust marker tracking algorithm for precise uav vision-based autonomous landing. In *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, pages 443–446, Oct 2015.

[20] Andrew Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least square fitting of ellipses. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(5):476–480, May 1999.

[21] Cheng Hui, Chen Yousheng, Li Xiaokun, and Wong Wing Shing. Autonomous takeoff, tracking and landing of a uav on a moving ugv using onboard monocular vision. In *Proceedings of the 32nd Chinese Control Conference*, pages 5895–5901, July 2013.

[22] C. M. Huang and T. S. Hung. Visual servoing of micro aerial vehicle landing on ground platform. In *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2071–2076, Oct 2014.

[23] Y. Okano and Y. Ito. Led marker position detection using walsh functions. In *2010 International Symposium on Intelligent Signal Processing and Communication Systems*, pages 1–4, Dec 2010.

[24] A. Censi, J. Strubel, C. Brandli, T. Delbruck, and D. Scaramuzza. Low-latency localization by active led markers tracking using a dynamic vision sensor. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 891–898, Nov 2013.

[25] Karl Engelbert Wenzel, Andreas Masselli, and Andreas Zell. Automatic take off, tracking and landing of a miniature uav on a moving carrier vehicle. *J. Intell. Robotics Syst.*, 61(1-4):221–238, January 2011.

[26] D. Kim and J. Choi. Multi-robot team outdoor localization using active marker and high frequency signal sources. In *2011 11th International Conference on Control, Automation and Systems*, pages 192–196, Oct 2011.

[27] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, May 2011.

[28] M. Fu, K. Zhang, Y. Yi, and C. Shi. Autonomous landing of a quadrotor on an ugv. In *2016 IEEE International Conference on Mechatronics and Automation*, pages 988–993, Aug 2016.

[29] Opencv project webpage. `http://opencv.org/`. Accessed: 2017-09-06.

[30] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(8):930–943, August 2003.

[31] V. Lepetit, F.Moreno-Noguer, and P.Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal Computer Vision*, 81(2), 2009.

[32] Ananth Ranganathan Th. The levenberg-marquardt algorithm, 2004.

[33] Esa navipedia. `http://www.navipedia.net/index.php/RTK_Fundamentals`. Accessed: 2017-09-06.

[34] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and Elmar Wasle. *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more*. Springer Science & Business Media, 2007.

[35] Tomoji Takasu and Akio Yasuda. Development of the low-cost rtk-gps receiver with an open source program package rtklib. In *international symposium on GPS/GNSS*, pages 4–6. International Convention Centre Jeju, Korea, 2009.

[36] X. W. Chang, X. Yang, and T. Zhou. Mlambda: a modified lambda method for integer least-squares estimation. *Journal of Geodesy*, 79(9):552–565, Dec 2005.

[37] Cristina D. Murta, Pedro R. Torres, and Prasant Mohapatra. Characterizing quality of time and topology in a time synchronization network. In *GLOBECOM*, 2006.

[38] A. Vallat and D. Schneuwly. Clock synchronization in telecommunications via ptp (ieee 1588). In *2007 IEEE International Frequency Control Symposium Joint with the 21st European Frequency and Time Forum*, pages 334–341, May 2007.

[39] Openntp project webpage. `http://www.openntpd.org/`. Accessed: 2017-09-06.

[40] Chrony project webpage. `https://chrony.tuxfamily.org/`. Accessed: 2017-09-06.

[41] A. Alfredo J. Almeida A. Dias G. Silva H. Ferreira, C. Almeida and E. Silva. Environmental modeling with precision navigation using roaz autonomous surface vehicle. In *IROS – Intelligent Robots and Systems, Workshop in Robotics for Environmental Monitoring*, 2012.

[42] u-blox. *NEO/LEA-M8T u-blox M8 concurrent GNSS timing modules Data Sheet*, 6 2016. R03.

[43] Linuxpps project webpage. `http://linuxpps.org/mediawiki/index.php/Main_Page`. Accessed: 2017-09-06.

[44] Gpsd project webpage. `http://www.catb.org/gpsd/`. Accessed: 2017-09-06.

[45] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Pixhawk: A system for autonomous flight using onboard computer vision. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2992 –2997, may 2011.

[46] J. Almeida, A. Ferreira, B. Matias, A. Dias, A. Martins, F. Silva, J. Oliveira, P. Sousa, M. Moreira, T. Miranda, C. Almeida, and E. Silva. Air and underwater survey of water enclosed spaces for vamos! project. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–5, Sept 2016.

[47] T. Santos, M. Moreira, J. Almeida, A. Dias, A. Martins, J. Dinis, J. Formiga, and E. Silva. Plined: Vision-based power lines detection for unmanned aerial vehicles. In *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 253–259, April 2017.

[48] P. Sousa, A. Ferreira, M. Moreira, T. Santos, A. Martins, A. Dias, J. Almeida, and E. Silva. Isep/inesc tec aerial robotics team for search and rescue operations at the eurathlon challenge 2015. In *Journal of Intelligent Robotic Systems*.