EXPLOITING ATTITUDE SENSING IN VISION-BASED NAVIGATION, MAPPING AND TRACKING INCLUDING RESULTS FROM AN AIRSHIP.

PHD THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING OF THE UNIVERSITY OF COIMBRA

By
Luiz Gustavo Bizarro Mirisola

September 2008

Abstract

In this thesis, an attitude sensor is used to compensate for rotational motion, and to generate virtual images to simulate a pure translation movement. The compensation of rotational motion is made possible by the combination of inertial and earth field magnetic sensors within a package which provides an absolute orientation measurement, and by a calibration routine to determine the rotation between the camera and inertial sensor frames. The objective is to facilitate vision-based tasks such as navigation over smooth terrain, 3D point cloud registration, generation of image mosaics and digital elevation maps, and plane segmentation. In the rotation-compensated, pure translation case, homographies are reduced to planar homologies, which are used to recover the relative pose between two views. In the particular case where the ground plane is horizontal, the relative pose between two views can also be recovered by directly finding a rigid transformation to register corresponding scene coordinates. These pure translation models perform fairly accurately especially regarding the estimation of the vertical motion component. The results include recovering trajectories of hundreds of meters for an airship UAV and comparison with GPS data. The accuracy of the height estimation was evaluated against ground truth through experiments in controlled environments. The rotational motion and the vertical and horizontal components of translational motion are measured or estimated separately, which should facilitate the integration of other sensors in the future, although only orientation measurements and image pixel correspondences have been used in this thesis. The rotation-compensated images and the pixel correspondences are further exploited to perform plane segmentation and to generate sparse Digital Elevation Maps and image mosaics. The visual odometry is also fused with GPS position fixes, improving the recovered trajectory locally. These gains are further exploited to improve the tracking accuracy of a target moving on the ground which is tracked in a 2D frame of reference in actual metric units, in the airship and urban people surveillance scenarios. Orientation estimates are also used to rotate 3D point clouds obtained by ranging devices such as stereo cameras or Laser Range Finders to a stabilized frame of reference, and the problem of registration of these point clouds is reduced to the determination of the remaining translation vector, taking into account the characteristics of the ranging device which generated the point clouds. The appendix presents

the technical characteristics of the UAV platform utilized for dataset recording, a remotely piloted airship, as well as requirements towards future extension for automatic flight.

Resumo

Nesta tese, um sensor de orientação é usado para compensar o movimento rotacional, e para gerar imagens virtuais, simulando um movimento de translação pura. A compensação do movimento rotacional se torna possível devido à combinação de sensores inerciais e de campo magnético, de forma a fornecer uma medida de orientação absoluta, e à uma rotina de calibração para determinar a rotação entre os sistemas de cordenadas da câmera e do sensor inercial. O objetivo é facilitar tarefas baseadas em visão como navegação sobre superfícies planas, registro de nuvens de pontos 3D, geração de mosaicos de imagens e mapas digitais de elevação, e segmentação de planos. Com a rotação compensada, supondo-se um movimento de translação pura, homografias são reduzidas à homologias planares, que são usadas para recuperar a pose relativa entre duas vistas. No caso particular onde o plano do chão é horizontal, a pose relativa entre duas vistas também pode ser recuperada pela estimação de uma transformação rígida que registre coordenadas correspondentes na cena. Estes modelos considerando translação pura obtêm resultados mais precisos especialmente ao estimar a componente vertical da trajetória. Os resultados incluem a recuperação de trajetórias de centenas de metros para um UAV dirigível com comparação com dados de GPS. A estimação da altura foi avaliada também em ambientes controlados, onde a altura real é conhecida com precisão. O movimento rotacional e os componentes horizontais e verticais do movimento translacional são medidos ou estimados separadamente, o que deve facilitar a integração de outros sensores no futuro, embora apenas medidas de orientação e correspondências entre os pixels das imagens foram usadas nesta tese. As imagens com rotação compensada e as correspondências entre os pixels das imagens são exploradas ainda para realizar segmentação de planos e para gerar mapas de elevação digital esparsos e mosaicos de imagens. A odometria visual é também fundida com leituras de posição do GPS, melhorando localmente a trajetória recuperada. Estas melhorias são ainda exploradas para aumentar a acurácia do seguimento de um alvo que se move no plano da terra, que é seguido em coordenadas 2D considerando as unidades métricas reais, nos cenários do dirigível e de vigilância urbana de pessoas. As estimativas de orientação também são usadas para rodar nuvens de pontos 3D, obtidas por sensores de mapeamento por medida de distância como cameras estéreo e Laser Range Finders, em um sistema de coordenadas estabilizado. Então o problema de registro destas nuvens de pontos é reduzido à determinação de um vetor de translação, levando em consideração as características do sensor que gerou as nuvens de pontos. O apêndice apresenta as características técnicas da plataforma UAV, um dirígivel pilotado remotamente, usada para gravar os datasets utilizados, bem como requisitos para extender a plataforma para a realização futura de vôos automáticos.

Acknowledgments

As I started to think about who deserves a mention here, I quickly realized the list would be too large to fit in this space, and any "selection of the most important people" would be necessarily unfair. The process of my coming to this PhD defense did not start in Coimbra, neither is restricted to the four years of work here. I have to thank not just the people who made life more enjoyable - or more bearable? - with their friendship and companionship, but also the ones that subtly - or not so subtly - made me go forward in the worse downturns.

I also realized that this must be written mainly in English, because of my many foreign friends from many countries. It was an interesting experience to be together, away from our countries and families. I hope that I have helped to receive and support some of you, but it is not enough to pay back the kindness of the ones who received me when I was in other country struggling with another language. E também aos que me receberam depois aqui em Portugal. Me despeço de Portugal, já com saudades do vinho, azeite, e bacalhau. As I will leave Europe I will miss the frequent opportunity to see "live" the history, a história, la historia, la storia, die geschichte, et la histoire. Back into Brazil where 200 years is very very amazingly old...

Um abraço para todos os outros brazucas perdidos, para os que voltam e para os que ficam. Espero que tenha lhes trazido doces do Brasil o suficiente para matar a saudade, mas não o suficiente para engordar muito. Saludo a los amigos españoles y de latinoamerica. ¡Viva la Sangria! Je salue aussi mes amis français et leurs familles. Je promets que je vais améliorer mon français ! Finally an english salute to all the others, at least you understand this language.

To my advisor, Professor Jorge Dias, and to all colleagues in the Institute of Systems and Robotics and in the Department of Electrical and Computer Engineering of the University of Coimbra, especially of the Mobile Robotics Laboratory, where I realized all my work, and to the other partners of the DIVA project, also in Lisbon and Minho, I thank for the advice and support, which allowed me to learn much during these years. My work depended on yours.

Finalmente aos meus pais obrigado por todo o apoio constante e incondicional. Agora já os dispenso do hábito de ir todo sabádo conectar no computador e conversar. Estive longe mas nunca sem notícias. E vamos lembrar da viagem por aqui.

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 State of the Art 2

1.2.1 Visual navigation & 3D mapping 3

1.2.2 Image Mosaicing 4

1.2.3 3D Depth map registration 5

1.2.4 Surveillance and Tracking 6

1.3 Objectives 6

1.4 Experimental Platform 9

1.5 Thesis summary 10

1.6 Publications 11

2 Sensor Modeling **13**

2.1 Projective Camera model 13

2.1.1 Stereo Cameras 14

2.2 Image Features & Homographies 15

2.2.1 Interest Point Matching 15

2.2.2 Planar surfaces and homographies 16

2.2.3 Pure translation case 18

2.3 Attitude Heading Reference System 23

2.4 Laser Range Finder 24

2.5 Frames of reference 25

2.6 Calibration of Camera - Inertial System 27

2.7 The virtual horizontal plane concept or an inertial stabilized plane . . 29

3 Inertial Aided Visual Trajectory Recovery **31**

3.1 Introduction 31

3.2 Visual Navigation 31

3.2.1 Reprojecting the images on the virtual horizontal plane 31

3.2.2 Procrustes solution to register scene points. 35

3.2.3 The homology model for a horizontal plane 39
3.2.4 From the FOE and μ to the translation vector 42
3.2.5 Filtering the translation measurements 44
3.2.6 Fusing visual odometry and GPS 44
3.3 The case of planes in general position 45
3.3.1 The homology model for planes in general position 45
3.4 Trajectory recovery results 49
3.4.1 Relative height experiments with ground truth 49
3.4.2 Overpass Experiment 52
3.4.3 Screwdriver Table Experiment 54
3.4.4 Trajectory recovery for aerial vehicles with the MTB-9 AHRS 55
3.4.5 Trajectory recovery for aerial vehicles with the MTi AHRS . . 66
3.4.6 Combining GPS and Visual Odometry 68

4 Inertial Aided Visual Mapping 73
4.1 Introduction 73
4.2 Image Mosaicing 73
4.3 Plane Segmentation and DEM 79
4.3.1 Plane Segmentation Experiments 79
4.3.2 Calculating Height For Each Pixel Correspondence 81
4.3.3 Constructing a DEM 83
4.3.4 Applications and Future Improvements 84
4.4 3D Depth Map Registration 86
4.4.1 Delimiting an image sequence 86
4.4.2 Rotating to a stabilized frame of reference 87
4.4.3 Translation to a common frame. 89
4.4.4 Subsuming other geometric constraints for outlier removal . . 92
4.4.5 Filtering out redundant points 94
4.4.6 Results 97

5 Aerial Surveillance by Visual Tracking 103
5.1 Introduction 103
5.2 Tracking of moving targets 104
5.2.1 Target Pose measurement: projecting from image to world frame 104
5.2.2 Filtering of target pose 105
5.3 Tracking of a Moving Target from Airship Observations 106
5.3.1 Tracking after fusing GPS and Visual Odometry 107
5.4 Tracking people with a moving surveillance camera 109

6 Conclusions 113

Bibliography **117**

A Experimental Platform **125**
A.1 Technical Characteristics of the DIVA Airship 125
A.1.1 Embedded System 125
A.1.2 Ground Station 126
A.2 Requisites for autonomy 127
A.2.1 Embedded System 127
A.2.2 Ground Station 128
A.3 Overview of architecture 129
A.3.1 Sensors and hardware components 129
A.3.2 Embedded & Ground Station Software 135

B Mathematical Proofs and formulae **139**
B.1 Recovering the scale of the recovered homography matrix 139
B.2 Calculating sigma points for the Unscented Transform 142
B.3 Jacobian formulae for the homology transformation. 143
B.4 Jacobian formulae for the optimization in $\mathbf{t}$. 145
B.5 The discrete Wiener process acceleration model for the Kalman Filter 146
B.6 The Procrustes Procedure 147

List of Figures

1.1 The DIVA airship, during taking off (a) and approaching to land (b). 2
1.2 The DIVA airship, with details showing the vision-AHRS system and GPS receiver mounted on the gondola. 9

2.1 The pinhole camera model. 14
2.2 The ideal stereo model of disparity for binocular cameras. 15
2.3 An example of interest point matching. Each line connects a pair of corresponding pixels in both images. 16
2.4 A 3D plane imaged by a moving camera induces a homography. . . . 17
2.5 An example of a homology planar transformation with two corresponding pairs of points in the plane. 19
2.6 A camera images a pair of 3D points. The depth ratio is related with a ratio of distances on the image (equation (2.8)). 21
2.7 Relating absolute scene depth with ratios of distances on the image for a pair of cameras and a pair of points. 21
2.8 The module of the magnetic vector in different environments. 24
2.9 Sensors utilized for mapping: AHRS, LRF, stereo and single cameras. 25
2.10 Definitions of frames of reference. 26
2.11 The calibration recovers the rotation between the camera and inertial sensor frames. The camera and AHRS measure the gravity direction. Two example images with vertically placed chessboards used in the calibration are shown in the right. 27
2.12 The virtual horizontal plane concept. 30

3.1 Example of reprojection in the virtual horizontal plane 32
3.2 An example of a reprojected image before and after cutting the low latitude regions. The limit was set at 45°. 34
3.3 Deleting pixels projected on the virtual sphere in low latitude regions. 35
3.4 The relation between pixel coordinates and the angle of the projection ray with the optical axis. 35

3.5 Finding the translation between successive camera poses by 3D scene registration. 37
3.6 A pair of virtual cameras under pure translation imaging the same 3D point. 40
3.7 Finding the homology transformation between two images reprojected on the stabilized plane. 43
3.8 Finding the scale of translation from the difference in height. 44
3.9 The ratios Z/Z', d/d', b/b' are the same. 47
3.10 Finding the homology transformation between two images. 49
3.11 A tripod with a camera rigidly mounted with the AHRS, and an image example. 50
3.12 Relative heights to the ground for the tripod experiment, with two example images. Each unit on the abscissa corresponds to one image. Images were taken from 3 different heights. 50
3.13 The camera mounted on a base, to be manually aligned with the walls, with an image example. 51
3.14 Relative heights to the ground plane, with two image examples. Each unit on the abscissa corresponds to one image. Images were taken from 5 different heights. 52
3.15 Trajectories recovered by the homology and homography models for the overpass dataset. 53
3.16 Trajectories recovered by the Procrustes procedure for the overpass dataset, and by optimizing in $\mathbf{t}$, with two distance metrics. 54
3.17 The screwdriver with its moving table and one example image (a) and the recovered 3D camera trajectories (b). 56
3.18 Two views of the recovered trajectories with highlighted displacements. 56
3.19 Visual odometry based on homology compared with GPS altitude measurements. 57
3.20 The μ values for individual pixel correspondences. 58
3.21 3D (a) and 2D (b) plots comparing the trajectory recovered by the homology with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation). 60
3.22 3D (a) and 2D (b) plots comparing the trajectory recovered by optimizing in the translation vector with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation). . . 61
3.23 Data from GPS and visual odometry fused by a Kalman filter. . . . 62
3.24 2D plot of the trajectories recovered by the homography model with different height measurements to scale it. Compare with figure 3.21. 63
3.25 2D plots comparing of the trajectory recovered by the homography model and scene registration approaches. 64

3.26 Comparison of average computation time per frame for some of methods tested. All experiments in this chapter were executed in a Pentium D 3.2 [GHz]. 65
3.27 3D (a) and 2D (b) plots comparing visual odometry with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation). 69
3.28 Maps formed by reprojecting the images on the ground plane, with trajectories superimposed. The circles indicate some of the GPS eph values (i.e. one standard deviation). 70
3.29 A satellite image of the flight area, the Coimbra City Airfield. 70
3.30 Data from GPS and visual odometry fused by a Kalman filter. The images are projected on the ground plane, forming a map. 71

4.1 Example of mosaic constructed from successive images. 74
4.2 Choosing the next base frame for the next mosaic. 75
4.3 On the right, two mosaics, with their pixel correspondences; left, the same mosaics registered and drawn together. 76
4.4 A mosaic from 61 registered images. 78
4.5 The relative depths from two cameras used to order different planes by they height. 80
4.6 The ground plane, and other parallel planes, segmented and ordered by height. 81
4.7 Calculating the height of a 3D point from an image pair under pure translation. 82
4.8 Image ratios before (a) and after (b) compensation by fitting a plane. The points on the building are the only ones above the ground plane. 83
4.9 2D and 3D plots of the DEM generated from the 10 frames mosaiced on the left. The red stars indicate the vehicle trajectory. 85
4.10 The data flow for the registration of each 3D point cloud. 87
4.11 The process for the registration of 3D point clouds. The box “process point cloud” corresponds to figure 4.10. 87
4.12 Compensating the rotation: point clouds aligned on inertial stabilized frame. 88
4.13 The concept of a corresponding pair of 3D points with a translation vector. 90
4.14 An example of a translation vector set with outliers. 90
4.15 An example of the usage of RANSAC to detect outliers. 91
4.16 An example of an aggregated point cloud 95
4.17 From the garden dataset: (a) one point cloud; (b) two registered point clouds. 98

4.18 The result of registering point clouds for 26 successive frames. 98
4.19 From the jeep dataset: (a) one point cloud; (b) two registered point clouds. 99
4.20 The result of registering point clouds for 27 successive frames. 99
4.21 The sidewalk dataset 100
4.22 The result of registering point clouds for 27 successive frames. 100
4.23 Registration of three point clouds taken by the LRF. 102

5.1 Tracking an independently moving target with observations from a moving camera. 104
5.2 A block diagram of the tracking process. 105
5.3 A 3D view of the recovered trajectories from Visual Odometry and from GPS, for the target and airship. 107
5.4 Tracking a car from the airship with the pure translation and the image only methods. The green circles are the tracked target trajectory with one standard deviation ellipses drawn in red. 108
5.5 The airship trajectory from GPS and from the fusion of GPS and visual odometry, with the target observations shown in the ground plane. . 108
5.6 The target trajectory over a satellite image of the flight area. The car followed the dirty roads. In (a), the airship trajectory was taken from GPS, in (b) a Kalman Filter estimated the airship trajectory by fusing GPS and visual odometry. 109
5.7 A photo with highlighted trajectories of camera and target person (a). A 3D view of the recovered trajectories, using Procrustes and optimizing in $\mathbf{t}$ to recover the camera trajectory (b). 110
5.8 A low pass filter is applied to the observed target trajectory. 111
5.9 A closer view of the target person tracked trajectory. 112

A.1 The embedded system of the DIVA project. 129
A.2 Hardware components of the DIVA prototype. 130
A.3 The Ground Station. 132
A.4 The MiniITX VIA EN 1500G CPU used in the DIVA on board computer. 133
A.5 Ground Station Components 136
A.6 Monitoring GUI at the Ground Station laptop with a instruments panel and a map of the flight area. 137

B.1 A 3D plane imaged by a moving camera induces a homography. . . . 140

List of Tables

3.1 Results for relative depth of 3D plane parallel to virtual image planes. 51
3.2 Results for relative depth of 3D plane in general position. 52
3.3 Comparison of trajectories recovered using the homography and the pure translation models for the overpass dataset. 53
3.4 Comparison of displacements recovered using the homography and homology models for the screwdriver table dataset. All displacements are given in [cm]. 55
3.5 Comparison of all trajectories presented with the GPS as a reference. Red and green cells represent the two worst and the two best values in each column. The grayed lines represent the results of Kalman Filters including the GPS position fixes which can not be fairly compared. All error values are given in meters. 66
3.6 Comparison of visual odometry with the GPS reference. All error values are given in meters. 67
3.7 Comparison of the execution time of visual odometry techniques. . . 67

A.1 Comparison between the PC104 and MiniITX standards. '+' indicates an advantage, '-' a disadvantage. 133
A.2 The weight budget of a flight experiment. 134

List of Algorithms

1 Finding pixels to delete from the image. 36
2 Filtering redundant 3D points from sequences of point clouds. 96

Nomenclature

Acronyms (in alphabetical order):

AHRS	Attitude Heading Reference System
CCD	Charge-Coupled Device
DEM	Digital Elevation Map
DIVA	Dirigível Instrumentado para Vigilância Aérea[1]
DOF	Degree of Freedom
eph	expected horizontal error (GPS)
epv	expected vertical error (GPS)
FOE	Focus of Expansion
GPS	Global Positioning System
ICP	Iterative Closest Point
LLA	Latitude Longitude Altitude
LRF	Laser Range Finder
NED	North East Down
PWM	Pulse Width Modulation
RANSAC	Random Sample Consensus
RGB	Red, Green, Blue
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
SURF	Speed Up Robust Features
UAV	Unmanned Aerial Vehicle
UUV	Unmanned Undersea Vehicle

[1]Portuguese for “Instrumented Dirigeable for Aerial Surveillance”

Notation: general rules

Type of variable	**Notation**	**Example**
scalar values	italic	x
symbols of SI units	italic between []	$1[kg]$
vectors, inclusive image coordinates	bold, lower case	$\mathbf{x}$
matrices	bold, upper case	$\mathbf{X}$
3D points (which are also vectors)	bold, italic, upper case	$\boldsymbol{P}$
3D points coordinates (scalars)	italic, upper case	$\boldsymbol{P} = (X, Y, Z)$
Sets of points (e.g. 3D point clouds)	blackboard font, upper case	$\mathbb{P}$
Frames of reference	calligraphic font w/ {}	$\{\mathcal{C}\}$
Frames & variables at a specific time	time index shown at the right	$\{\mathcal{C}\}\|_i$
quaternions (which are four vectors)	a $\circ$ above the vector	$\mathring{\mathbf{q}}$

Notation: variable list

Symbol	**Unit**	**Definition**
$\mathbf{a}$		homology axis; vanishing line of the 3D plane
${}^{\mathcal{W}}\mathbb{A}$		aggregated point cloud in world frame
B		baseline: distance between camera centers
$\square_B$		B: time index of the base frame used as the $\{\mathcal{W}\}$ frame.
d	m	distance from camera center to 3D plane
$dist()$	pixels	distance metric (usually Euclidean) measured on the image
dpx, dpy	mm	Pixel dimensions in the CCD sensor array
f	pixels	camera focal length
$\mathbf{F}$		Fundamental matrix
$\mathbf{G}$		3×3 matrix representing a homology transformation
$\mathbf{g}$		gravity vector
h	m	height above the ground plane
hp	m	height of a point above the ground plane
$\mathbf{H}$		3×3 matrix representing a homography transformation
$\mathbf{H}_\infty$		the infinite homography
i		index; time index for reference frames
$\mathbf{i}$		intersection of the homology axis with the line $\overline{\mathbf{x}\mathbf{x}'}$
$\mathbf{I}$		identity matrix of appropriate size
$\mathbf{I}_i$		image acquired at time index i
$\mathbf{I}_i^{\mathcal{D}}$		The virtual image of the $\{\mathcal{D}\}\|_i$ camera at time index i.
$\mathbf{K}$		pinhole camera model intrinsic calibration matrix
$\mathbf{l}$		a line, usually in the image.

Symbol	**Unit**	**Definition**
$\mathbf{n}$		vector normal to a 3D plane
$\mathbf{n}$	pixels	the nadir point on the image, principal point of the $\{\mathcal{D}\}\|_i$ camera
n		number of measurements or elements in a set
$\mathbf{P}$		a camera projection matrix
${}^{\mathcal{X}}\mathbb{P}\|_i$		point cloud (set of 3D points) in frame of reference $\{\mathcal{X}\}$, time index i
$\mathbf{R}$		3×3 rotation matrix
${}^{\mathcal{Y}}\mathbf{R}_{\mathcal{X}}$		3×3 rotation matrix changing frames of reference from $\{\mathcal{X}\}$ to $\{\mathcal{Y}\}$
$\mathbf{T}$		3×3 translation matrix
$\mathbf{t}$		a translation vector
$\mathbf{v}$	pixels	image coordinates of the homology vertex; the FOE
$\mathbf{x}$	pixels	an image pixel, in homogeneous coordinates unless otherwise specified
${}^{\mathcal{W}}\boldsymbol{X}_C(i)$	m	the camera position in the $\{\mathcal{W}\}$ frame at time index i
${}^{\mathcal{W}}\boldsymbol{X}_O(i)$	m	the tracked object position in the $\{\mathcal{W}\}$ frame at time index i
$\mathbb{X}$		set of points in the ground plane, projections of pixel correspondences
$\boldsymbol{X}, \boldsymbol{P}, \boldsymbol{Q}$		3D points
x_0, y_0	pixels	Camera principal point coordinates
Z		depth: coordinate of a point along the optical axis.
δ		distance between image planes; baseline in the depth direction.
θ	rad	angle of a laser beam in the 2D scan plane.
λ		scale of the recovered homography matrix
μ		relative depth; homology cross ratio parameter
ρ	cm	a range measurement
φ	rad	pantilt's tilt angle
$\mathbf{0}$		a zero vector or matrix of appropriate size

Chapter 1

Introduction

1.1 Motivation

This thesis combines orientation measurements from an Attitude Heading Reference System (AHRS) and computer vision techniques, exploiting this combination in problems related with the determination of the relative pose of successive views taken by a mobile observer, such as an Unmanned Aerial Vehicle (UAV). It includes results obtained from datasets acquired by the remotely piloted unmanned airship of figure 1.1, which carries both a camera and an AHRS. The covered problems are trajectory recovery for navigation above leveled terrain, registration of 3D point clouds acquired by the mobile observer, image mosaicing and 3D mapping with monocular cameras, and tracking of moving objects in the ground plane. This thesis explores the benefits obtained when the orientation estimates allow the rotational motion to be compensated, and imagery or 3D point clouds to be reprojected into a stabilized reference frame.

The limits of computer vision or sensorial data fusion alone have already been explored, and it is already known that some limits may be overcome by combining them. This combination has been made possible by recent technological advances, such as the development of small scale sensor packages (AHRS) which output absolute orientation estimates. A modern AHRS outputs geo-referenced orientation measurements, using accelerometers which measure the direction of gravity and magnetometers which measure the earth magnetic field. The AHRS firmware fuses and filters information from its internal sensors, freeing the main CPU for higher level tasks and generating outputs at a larger frequency than the frame rate of typical digital cameras.

The experiments reported in is thesis utilize a small, micro-machined AHRS which is fixed rigidly to a digital camera, providing a geo-referenced estimate of the camera orientation. To generate this estimate the rigid transformation between the camera

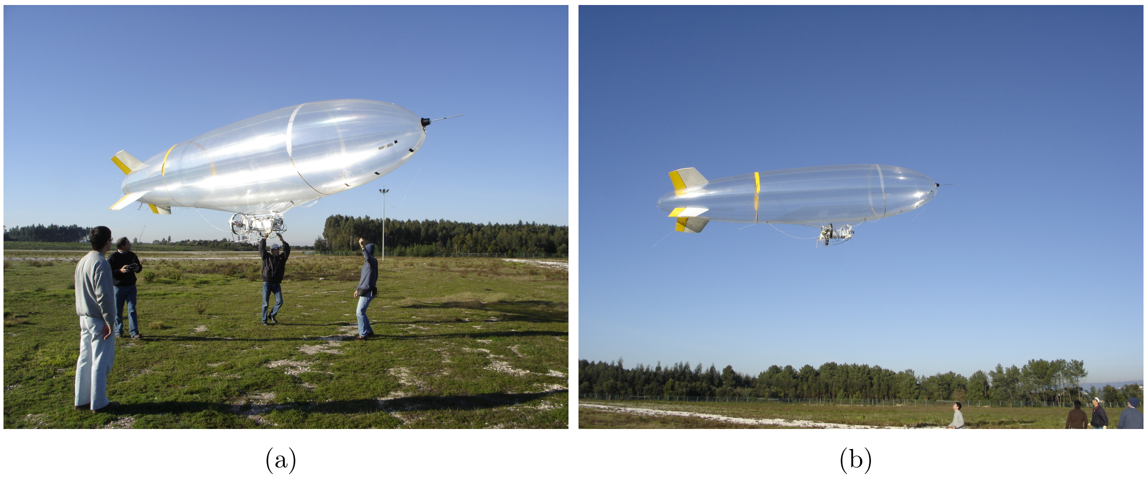


Figure 1.1: The DIVA airship, during taking off (a) and approaching to land (b).

and inertial sensor frames must be known. This transformation is estimated by a recently developed calibration procedure [Lobo and Dias, 2007], eliminating the need for precise mechanical assembly. This has already been used to improve robustness on image segmentation and 3D structure recovery from stereo [Lobo and Dias, 2003, Mirisola et al., 2006] or independent motion segmentation [Lobo et al., 2006].

The development of vision-only or inertial-only algorithms are certainly worthy research goals by themselves and are suited to applications where only one of them is available. Nevertheless, the technological development, producing higher quality digital cameras and inertial and magnetic sensors at smaller sizes, weights and costs increase the range of applications where both are available together. For these latter applications it is already becoming clear that many important problems can be solved more efficiently, accurately, or with better scalability, with the combination of both sensor modalities.

1.2 State of the Art

This section reviews recently reported research on the area of relative pose recovery and navigation, using computer vision, sensorial data, and especially the fusion of both. The related problems of image mosaicing, 3D map registration and tracking are also covered.

1.2.1 Visual navigation & 3D mapping

In [Hygounenc et al., 2004], a stereovision-only approach is used to build a 3D map of the environment from stereo images taken by a remotely controlled airship, keeping estimates of the camera pose and the position of automatically detected landmarks on the ground. The landmarks are found by interest point algorithms applied on the aerial images. It was not their aim to integrate inertial measurements.

Again utilizing stereo images taken by a UAV, the trajectory can be recovered by registering successive sets of triangulated 3D points calculated for each stereo frame. Trajectories of hundred of meters have been recovered [Kelly et al., 2007], although the UAV height is limited to a few meters due to stereo baseline size. An earlier work focused on estimating the height above the ground plane using efficient sparse stereo techniques [Corke et al., 2001].

Images from a moving camera can also be used to track a sparse set of 3D points in the environment using triangulation and interest points matched across the image sequence and considered as natural landmarks. These estimates of observed landmarks, combined with measurements from inertial sensors, were used to estimate the landmark poses and the camera trajectory for a ground vehicle carrying an omnidirectional camera through a large trajectory [Strelow and Singh, 2004]. Matched interesting points in images with broad baseline were used to generate stochastic epipolar constraints which were used to minimize drift in the position estimate from inertial-based navigation [Diel, 2005].

Image mosaicing was performed for an unmanned submarine navigating above flat sea-bottom, using only images from a monocular camera as input for the calculation of relative poses [Gracias, 2002]. The most recent vehicle orientation estimate was used to reproject the images onto a stabilized plane, avoiding using direct measures from inertial sensors. The vehicle pose is estimated, and a mosaic of the sea-bottom is generated, which in turn is used for navigation. Although it involved elaborate optimization steps, the registration converges only if the vehicle has restricted movement and shows minimal variation in roll and pitch angles. These results indicate a limitation for vision-only approaches.

A UAV trajectory can also be estimated by fusing GPS and on-board inertial data and considering a dynamic vehicle model. Given accurate vehicle poses, images taken from a high-flying airplane are reprojected onto the ground plane thus achieving one-pixel accuracy with no need for image-based registration techniques [Brown and Sullivan, 2002]. At high altitudes a relatively flat area can be safely assumed to be planar.

Combined inertial and vision data were used to keep pose estimates in an underwater environment, navigating a robot submarine above a large area [Eustice, 2005], with no access to a beacon-based localization system. Relative pose measurements from the images were used to avoid divergence of the tracked vehicle pose, and an image mosaic was generated as a byproduct.

In the context of an aerial vehicle, even without utilizing the available GPS data, inertial measurements and observations of artificial landmarks on images can be fused together to provide a full 6-DOF pose estimate, performing localization and mapping, and incorporating recent advances on filtering. Inertial sensors and barometric altitude sensors can also compensate for inaccurate GPS altitude measurements or satellite drop-outs [Kim, 2004].

The trajectory of a mobile observer can be recovered from images of a planar surface using interest point matching and the well known planar homography model. Various geometric constraints have been proposed to recover the right motion among the four solutions of the homography matrix decomposition [Caballero et al., 2006]. This has been already performed for an airship by using clustering and blob-based interest point matching algorithms, building an image mosaic which in turn is used as a map for navigation. The relative pose estimation involved homographies calculated from images of a planar area taken by a UAV [Caballero et al., 2006] and image sequences taken by various UAVs [Merino et al., 2006].

The trajectory of a UAV can also be recovered by tracking known fixed targets on the ground, which requires modifying the environment [Saripalli et al., 2003].

Another example of usage of inertial data to aid vision tasks are ego-motion algorithms without using pixel correspondences [Makadia and Daniilidis, 2005], where gravity vector measurements compensate for two rotational degrees of freedom, decreasing the dimensionality of the unknown parameter space.

Visual servoing schemes based on the relative pose obtained by decomposing homographies could also be improved with the homology model [Suter et al., 2002], especially when depth ratios calculated from the homography matrix are directly utilized.

Aerial vehicles have been utilized to produce 3D maps of the ground using a variety of different sensors. Stereo images were used to build a dense 3D map of the ground surface, in the form of a DEM (Digital Elevation Map), in the work already cited above [Hygounenc et al., 2004]. Stereo imagery has also been combined with other vision techniques such as color segmentation [Huguet et al., 2003]. Airborne range sensing devices such as laser range finders or radars have also been extensively used to build 3D maps actually exploited in domains such as geology [Cunningham et al., 2006].

1.2.2 Image Mosaicing

Image mosaicing consists on registering a set of images of a surface in the world, producing a single, larger image of this surface. The mosaic ideally images the union of the areas imaged by the individual images.

Many of the image mosaic results existing so far register images taken at the same position, but with the camera looking at different directions, i.e., a pure rotational movement. In this case, the mosaicing surface is the plane at infinity, or the sphere of directions. Disregarding differences on the camera intrinsic parameters, the transformation needed to register any pair of images is an infinite homography, as there is not translation. For the specific pure-rotation case, there are commercial software products which take as input a set of images, and output a single mosaiced image, which is called a *panorama* [Brown, 2006]. The solutions available take into account differences on the camera intrinsic parameters [Brown and Lowe, 2003], illumination [Eden et al., 2006], and include optimization steps to reach a final registration, plus processes to produce a visually clean image by eliminating misregistered images of the same object which appear repeated in the mosaic, called "ghosts" [Szeliski, 2005, Szeliski, 2004, Brown and Lowe, 2003].

Image mosaicing also can be done above a planar surface, allowing the camera to freely translate, and there are also solutions available for this case, including bundle adjustment, N-view interest point matching, and automatically reprojecting the images with an infinite homography in a direction that minimizes the projective distortion [Capel, 2001]. The number of images registered on such mosaics range from a handful to less than a hundred images.

1.2.3 3D Depth map registration

3D mapping with color images and a rotating LRF was already performed [Ohno and Tadokoro, 2005], but without any calibration process to calibrate the rotation between the sensor frames, and using only ICP to register the point clouds. Rotating LRFs were also used to recover 3D range scans, that were matched to build maps, in room scale with small mobile robots [Kleiner et al., 2005].

Large terrestrial 3D mapping [Triebel et al., 2006] was performed using laser scanners mounted on a car, and registered on the world frame with the aid of fused information from on-board GPS, inertial systems and odometry. Before being integrated into a new representation for the global map, local laser scans were registered together with ICP algorithms, with no help from image or inertial data.

The ICP (Iterative Closest Point) algorithm [Besl and McKay, 1992], and its variations, has been widely used for the registration of pairs of 3D point clouds. It is based on picking a subset of 3D points from each point cloud, and associating each such point to the closest point in the other point cloud. Then, a transformation is found to register these corresponding pairs of points. The process is repeated until no significant progress is made. This process is vulnerable to convergence into local minima, particularly when the initial position is far from the correct one. Applying ICP after the processes defined in this thesis remain a possibility, depending on the application.

Results on visual odometry from stereo images associate 3D points obtained from stereo images with interest points detected on the images. By matching the interest points on images taken successively, the sets of 3D points derived from these images are registered, recovering the full 6-DOF relative pose between two views [Cheng et al., 2006, Sünderhauf and Protzel, 2007]. The error on stereo triangulation is considered, leading to different weights to each constraint depending on the position of the 3D points relative to the cameras. The case of translation-only movement the registration reduces to solve a weighted average [Matthies and Shafer, 1987].

The processes of interest point matching and stereo imaging are used in the registration of point clouds, but they usually output wrong estimates for some pixels, which are called outliers. To detect and eliminate these outliers, robust estimation techniques such as RANSAC [Fischler and Bolles, 1981] may be used, or geometric constraints can be exploited, avoiding iterative techniques. An example are the two constraints proposed by [Hirschmüller et al., 2002], which exploit the fact that the transformation is rigid. Given a pair of 3D points which have correspondences on the other point cloud, the first constraint checks if the distance between the two 3D points remains the same after the transformation. The second constraint checks if the pair of 3D points is rotated by an angle too large, given a maximum possible rotation between both point clouds.

A comprehensive comparison of different approaches for the registration of stereo point clouds is still missing in the literature, as it was noted in a recent survey paper [Sünderhauf and Protzel, 2007].

1.2.4 Surveillance and Tracking

The Unscented Transform [Julier and Uhlmann, 1997] has been used to propagate uncertainty in the moving camera position and orientation, and from the target detected position in the image, to the target position on the ground plane. Then from a series of successive observations of the same static target, a final estimate of its position was obtained, taking into account the anisotropic uncertainty of each observation [Merino et al., 2005].

Uncertainty in the camera orientation estimate is often the most important source of error in tracking of ground objects imaged by an airborne camera [Redding et al., 2006], and its projection in the 2D ground plane is usually anisotropic even if the original distribution is isotropic.

1.3 Objectives

This thesis aims to perform trajectory recovery from a sequence of images reprojected in a stabilized frame where the rotation is compensated with the aid of AHRS orientation estimates. In the particular case where the ground plane is horizontal, the relative pose between two views can also be recovered by directly finding a rigid transformation to register corresponding scene coordinates. Additionally, for a sequence of images of a planar area, the transformation that relates corresponding pixel coordinates in two images is a planar homography, which is reduced to a planar homology in the pure translation case. Some preliminary results using the homology model have already been obtained [Michaelsen et al., 2004], but a comprehensive evaluation of these pure translation models with aerial images is still missing in the literature.

In our experiments, the camera trajectory is recovered without recourse to artificial landmarks and using only a monocular camera instead of depending on stereo imaging. The method depends on orientation measurements, which are obtained only from the AHRS sensor package, without considering any model of the vehicle dynamics. GPS is utilized only for comparison, and not on the trajectory recovery process itself, except in the experiments where both GPS and visual odometry are fused.

As it is shown in some of the works reviewed, incorporating GPS measurements or considering a dynamic vehicle model in the estimation of the camera orientation could improve the orientation estimate. Moreover, the models shown here has performed reasonably or at least better than image only approaches even with a relatively low-cost and inaccurate AHRS. More expensive and accurate AHRS models should provide more accurate estimates.

Differently of the UUV utilized to perform navigation and mosaicing [Gracias, 2002], the airship used in this thesis has large variations on roll and pitch during its flight, which is typical behavior for airships [de Paiva et al., 2006]. The orientation estimates compensate for these variations, and overcome that limitation of the vision only approach.

Some of the works reviewed also utilize the reprojection of images into a virtual stabilized plane. Nevertheless, the homography model is still used. It may still be necessary to recover the residual rotation for some applications, but the pure translation models presented here are certainly an option, and appear to be especially suitable to estimate the vertical motion component. Moreover, with the pure translation models the extraction of the translation vector up to scale has a unique solution. In contrast, with the homography model, the recovered matrix must be algebraically decomposed into rotational and translational components, yielding four possible solutions, of which only one is the real relative pose [Ma et al., 2004].

Additionally, the homology model explicity separates the vertical and the horizontal components of the translation, facilitating the fusion of other measurements of these specific motion components in the future. For example, GPS north-east velocity or optical flow could be used to measure the direction of horizontal translation.

This work deals mainly with the calculation of relative pose between successive camera poses, and the trajectory recovered is only a concatenation of the relative poses, with Kalman Filters employed to reduce errors. It also includes results from the fusion of visual trajectory recovery and GPS position fixes. But this relative pose measurement could be incorporated in localization and mapping schemes such as the ones reviewed before, using artificial landmarks placed on the environment or natural landmarks found by interest point matching.

This thesis also provides a method for obtaining 3D maps from the stabilized imagery, using the images and a FOE estimate to estimate the height or the 3D points imaged by some image pixels. This map is less accurate and much coarser than the ones obtained by other sensors such as airborne LRFs and radars or stereo imagery, but it is very fast to obtain, and it requires only images from a monocular camera.

While this thesis do not develop new algorithms in the image mosaicing domain, it quantifies the gains obtained with the usage of camera orientation estimates to reproject images on a virtual horizontal plane, and present an image mosaic as an example. These gains are related with the gains of affine invariant interest point algorithms, when compared with similar algorithms invariant only to rotation, translation, and illumination [Mikolajczyk and Schmid, 2004, Mikolajczyk et al., 2005]. Other navigation works have already used inertial data or image based measures in the same way [Gracias, 2002, Brown and Sullivan, 2002, Eustice, 2005], but to our knowledge it was not clearly quantified what is gained with the reprojection of images into a stabilized plane.

In the 3D point cloud registration domain, it is necessary to find the relative pose between successive camera poses. The rotational components of these relative poses may be directly compensated by exploiting the AHRS orientation estimates. Further, the translational components may be more efficiently estimated when dealing with rotation-compensated point clouds. This thesis leverage results in visual odometry and interest point matching and applies them into the registration of 3D maps obtained by stereo cameras and laser range finders (LRF), always exploiting AHRS orientation estimates, and detailing the specific coordinate frames and sensor models utilized in each case. A comparison with ICP based algorithms is also provided for some results, aiming to determine under which conditions it is advisable to employ the the method described here instead or together with ICP.

This thesis also showns that the improvements in trajectory recovery for the camera also translate into improvements in the accuracy of the tracking of a moving object in the ground. The recovered target trajectory becomes smoother or more accurate, due to the improvements in the camera pose estimation and because the parameters of the filters involved in the tracking are defined in the actual metric units related to the target motion, which is also tracked in a metric frame. In contrast, when the tracking is performed directly on the image, in pixel coordinates, the scene suffers projective distortion and the relation between pixel units and metric distances in the ground changes with the camera height and these distortions are often not taken into account. Two scenarios are covered: aerial surveillance with the airship, and urban people surveillance with a moving camera.

1.4 Experimental Platform

The trajectory recovery algorithms were tested on data obtained from a remotely controlled blimp, the DIVA non-rigid airship. Figure 1.2 shows an image of the airship, which has a helium filled envelope with $18[m^3]$ volume. It is $9.4[m]$ long, with $1.9[m]$ diameter and the maximum speed attained is around $70[{}^{km}/{}_{h}]$. Typically, the maximum height reached during our remotely piloted flights reached $200[m]$ above the ground, corresponding to an altitude of approximatelly $300[m]$.

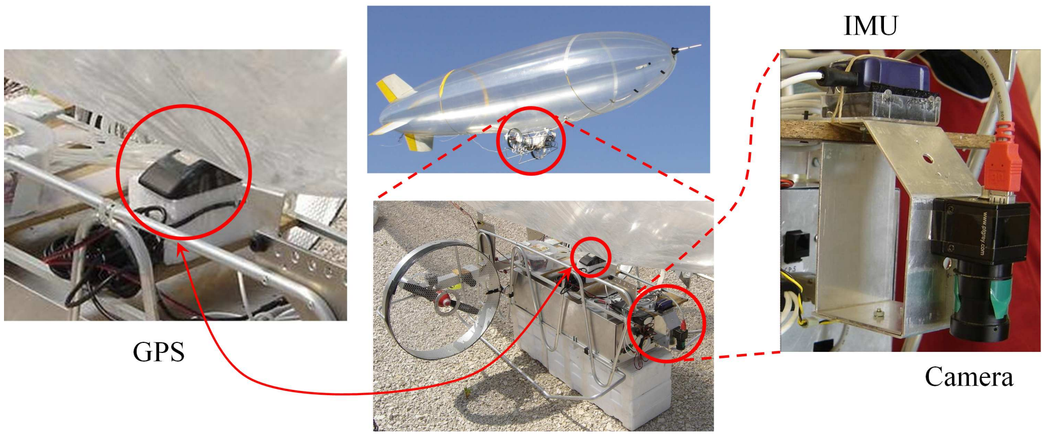


Figure 1.2: The DIVA airship, with details showing the vision-AHRS system and GPS receiver mounted on the gondola.

It is propelled by two internal combustion motors with propellers, and controlled by standard aeromodel equipment. Servo motors are used as actuators, to control the motors acceleration and steer the airship with the flaps mounted in the back of the envelope. A human pilot with a radio control unit sends commands to an on-board radio control receiver to control the servo motors.

The combustion motors and electronic equipment are mounted on the gondola, below the envelope, which may have a maximum weight of $13.5[kg]$. Thus discounting the weight of the mechanical structure of the gondola, motors and aeromodel equipment, the payload available for electronic equipment and sensors is about $7[kg]$.

The blimp carries a digital camera, rigidly mounted with an AHRS, and a GPS receiver, among other sensors, plus an embedded CPU with flash memory storage to read and store sensor data and images, and radio Ethernet equipment to communicate with the ground. Figure 1.2 also shows details of the camera inertial system and the GPS receiver.

A ground station, consisting of a laptop connected to a wireless Ethernet access point with a $7[dB]$ antenna, receives telemetry data during the flight, and displays it to the operator, who is able to check the operation of all sensors before taking off and monitor the telemetry data during the flight.

The datasets used in the experiments of chapters 3 and 5 were acquired with this platform, remotely piloted. Appendix A provides a more detailed description of the onboard platform and the ground station, considering both software and hardware components.

The sensor specifications most relevant for the experiments of chapters 3 and 5 are:

Digital Camera: Model Point Gray Flea [Point Grey Inc., 2007], a digital camera with a bayer color CDD array with resolution of 1024x768 pixels and Firewire IEEE 1394 interface, weighting $40[g]$ without lens.

GPS receiver: Model GARMIN GPS35 [Garmin Int. Inc., 2007], a 12 channel GPS receiver unit, which outputs 3D position and velocity at $1[Hz]$ rate, and weights $130[g]$. DGPS correction was not utilized in this thesis.

Inertial System: The last experiments used a XSens MTi, and the first ones a XSens MTB-9 [XSens Tech., 2007]. Both AHRS sensor suites have 3-axis accelerometer, inclinometer and gyroscope. They contain also a thermometer, and output calibrated and temperature-compensated sensor readings, plus filtered absolute orientation at maximum $100[Hz]$ rate. The weights are $35[g]$ (MTB-9) and $50[g]$ (MTi). If the sensor is static, then the manufactures states that the error in its orientation estimate has standard deviation of $3[°]$ for the MTB-9 and for the MTi of $0.5[°]$ in the pitch and roll angles and of $1[°]$ in the heading direction. In dynamic conditions this error should be larger.

1.5 Thesis summary

The next chapter reviews concepts, sensor models and frames of reference which will be needed in the rest of the thesis. It also reviews the calibration camera inertial utilized plus some important results in computer vision including the case of translation-only movement.

Chapter 3 presents models and results for trajectory recovery with rotation-compensated imagery, including experiments using the DIVA blimp. The chapter opens with the reprojection of the images on the virtual horizontal plane, a process that will be used in the next chapters too.

Then, chapter 4 covers Image Mosaicing and Mapping. The mapping experiments include point cloud registration with stereo cameras and LRFs, and mapping from monocular cameras. The latter includes mapping from rotation-compensated imagery taken by the DIVA UAV, and plane segmentation experiments.

The trajectory recovery results are extended to the tracking of an object independently moving on the ground plane which is observed by the moving camera, in chapter 5.

The conclusions and discussion of results are presented in chapter 6. After the conclusions, the appendix A present a more detailed view of the DIVA airship, including system characteristics and pre-requisites for future expansion, and hardware and software architecture. Finally some mathematical proofs and formulae are left to appendix B.

1.6 Publications

The following publications resulted of the work leading to this thesis:

Book chapters

- Mirisola, L. G. B., Dias, J. "Tracking a Moving Target from a Moving Camera with Rotation-Compensated Imagery" In "Intelligent Aerial Vehicles", I-Tech Publishing, Vienna, Austria, 2008
  A more complete version of the results in tracking of a moving object on the ground after the camera trajectory is recovered are published in this paper, including the airship and people surveillance scenarios (chapter 5) and including the fusion of GPS and visual odometry.

- Moutinho, A, Mirisola, L. G. B., Azinheira, J., Dias, J. "Project DIVA: Guidance and Vision Surveillance Techniques for an Autonomous Airship" In *"Research Trends in Robotics"* (preliminary book title), NOVA Publishers, 2008.
  This book chapter covers part of the trajectory recovery and mapping results using imagery taken by the DIVA airship, which appear in chapter 3 and section 5.2.

Conference Papers

- Mirisola, L. G. B., Lobo, J., and Dias, J. "Stereo vision 3D map registration for airships using vision-inertial sensing." *The 12th IASTED Int. Conf. on Robotics and Applications (RA 2006)*, Honolulu, HI, USA, August 2006.
  This paper presents the 3D depth map registration using a stereo camera, as presented in section 4.4.

- Mirisola, L. G. B. and Dias, J. M. M. "Exploiting inertial sensing in mosaicing and visual navigation." *6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV07)*, Toulouse, France, September 2007.
  In this paper were published, with less detail, the mosaicing results of section 4.2, and the earliest experiments with the homology model: the tripod height determination of section 3.4.1.1 and the recovery of altitude for the blimp flight (part of section 3.4.4).

- Mirisola, L. G. B., Lobo, J., and Dias, J. "3D Map Registration using Vision/Laser and Inertial Sensing" *European Conference on Mobile Robots (ECMR07)*, Freiburg, Germany, September 2007.
  The 3D map registration algorithm (section 4.4) was adapted and applied for the LRF setup, with the introduction of the weighted averaging and comparison with ICP.

- Mirisola, L. G. B., Dias, J. M. M. and Traça de Almeida, A. "Trajectory Recovery and 3D Mapping from Rotation-Compensated Imagery for an Airship." *In International Conference on Intelligent Robots and Systems (IROS 07)*, San Diego, CA, USA, November 2007.
  The first version of the full trajectory recovery results for the DIVA UAV (section 3.4.4) was published here. This paper also includes the mapping results of section 4.3.3.

- Mirisola, L. G. B. and Dias, J. "Tracking a Moving Target from a Moving Camera with Rotation-Compensated Imagery" *Israeli Conference in Robotics*, Herzlia, Israel, November 2008
  A shorter version of the results in tracking of a moving object on the ground after the camera trajectory is recovered are published in this paper, including the airship and people surveillance scenarios (chapter 5).

Chapter 2

Sensor Modeling

2.1 Projective Camera model

The pinhole camera model, shown in figure 2.1 represents the central projection of points in space onto a plane. The center of projection, called *camera center*, is the origin of an Euclidean coordinate system called the *camera frame* and denoted as the $\{\mathcal{C}\}$ frame. The plane $z = f$ is called the *image plane*, and f is called the *focal length.* A point in space with homogeneous coordinates $\boldsymbol{X} = (X, Y, Z, 1)^T$ in the camera frame is mapped to the homogeneous image point $\mathbf{x} = (x, y, 1)$. Considering a line joining the point $\boldsymbol{X}$ to the camera center, the image point $\mathbf{x}$ is where this line intersects the image plane. The line that passes through the camera center and is perpendicular to the image plane is the *optical axis*, and the intersection of the optical axis and the image plane is called the *principal point.*

The mapping is defined by the following equation:

$$\mathbf{x} = \mathbf{K}[\mathbf{I}|\mathbf{0}]\boldsymbol{X} \tag{2.1}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{0}$ the zero vector, and $\mathbf{K}$ is the *camera calibration matrix*, or *intrinsic parameter matrix*, that is defined by:

$$\mathbf{K} = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

where f_x and f_y represent the focal length of the camera in terms of pixel dimensions in the x and y directions respectively. As f_x and f_y are often very close, in this thesis

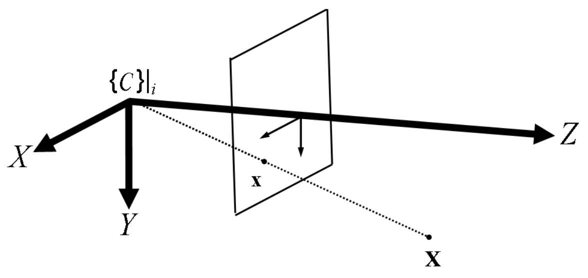


Figure 2.1: The pinhole camera model.

f is considered as the average of its two components, although this approximation is not strictly necessary. The variables x_0 and y_0 are the principal point coordinates in terms of pixel dimensions, and s is the *skew parameter*, very small for most cameras. In this thesis digital cameras with CCD sensors are used, where the image plane is considered as the plane of the CCD surface with its array of light sensors.

Each camera have its lens fixed and is calibrated, i.e., its intrinsic parameter matrix $\mathbf{K}$ is determined. Besides that, the calibration process finds the parameters which define lens camera distortion - deviations from the camera projective model caused by the optical properties of the lens geometry. Through all this thesis, all cameras are calibrated, and all images are corrected for lens distortion. Single cameras were calibrated by the Camera Calibration Toolkit [Bouguet, 2006], and stereo cameras by the SVS Calibration Software [Konolige, 1997].

In this thesis all camera intrinsic parameters and the matrix $\mathbf{K}$ are given in pixel units unless it is explicity defined otherwise. The dimension of each sensor on the CDD sensor array, i.e., the pixel size in millimeters, is given in the camera manual. In this thesis the pixel dimension is denoted as (dpx, dpy) and used to transform distances in the image plane from pixels to a metric scale when needed. Alternatively, a calibration target with known dimensions can be used to calibrate the camera to find the components of $\mathbf{K}$ in metric units.

2.1.1 Stereo Cameras

The stereo camera consists on two identical cameras rigidly mounted with near parallel optical axes. The distance between the optical centers of both cameras is called *baseline*, denoted by B. When both cameras image the same 3D point $\boldsymbol{P}$, this point will be projected on different positions on the two image planes, $\mathbf{x}_L$ and $\mathbf{x}_R$, referring to the projections on the left and right camera, respectively, as shown in figure 2.2. The difference $\mathbf{x}_L - \mathbf{x}_R$ is called *disparity*, and the depth of the point $\boldsymbol{P}$ is related to the disparity by:

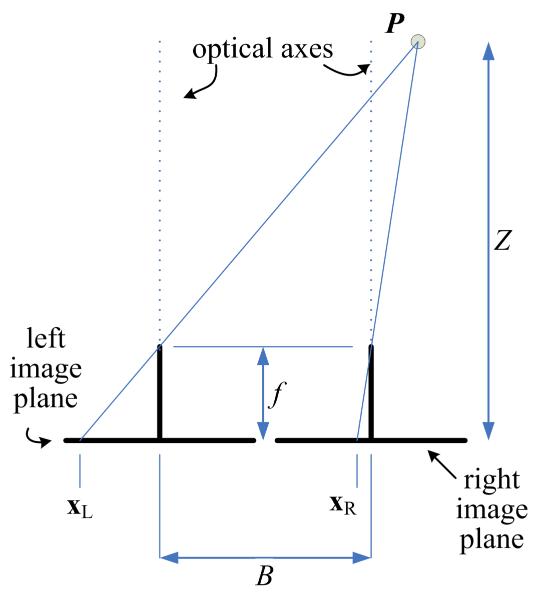


Figure 2.2: The ideal stereo model of disparity for binocular cameras.

$$Z = \frac{B \cdot f}{\mathbf{x}_L - \mathbf{x}_R} \tag{2.3}$$

The SVS (Small Vision System) [Konolige, 1997] is utilized to generate a disparity image (a disparity value is calculated for every pixel which can be associated with a corresponding pixel in the other camera), and to calculate a set of 3D points ${}^{\mathcal{C}}\mathbb{P}|_i$, defined in the left camera frame $\{\mathcal{C}\}$. This set is called a *point cloud*. Each 3D point has a color c given by the corresponding pixel in the image, i.e $c = \mathbf{I}_l(\mathbf{x}_L)|_i$, where the color may be a gray level for monochromatic cameras or a RGB color for color cameras.

2.2 Image Features & Homographies

2.2.1 Interest Point Matching

Given a pair of images of the same area, many 3D points in the world are imaged in both images, i.e., the 3D points are projected into both image planes and thus each 3D point correspond to one pixel on each image. The interest point matching problem consists on finding pairs of pixels, one from each image, which are projections of the same 3D point in the world.

Figure 2.3 shows an example of corresponding pixel pairs found on two partially

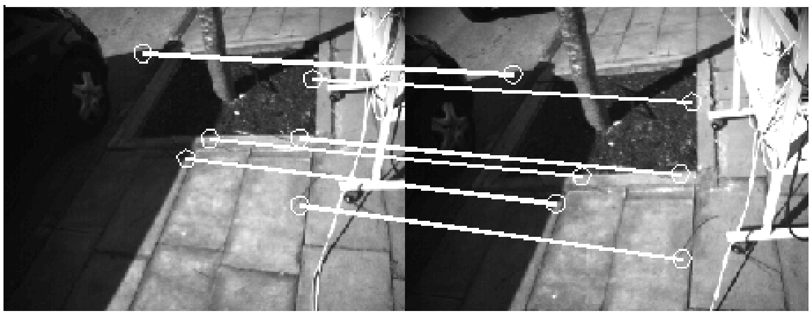


Figure 2.3: An example of interest point matching. Each line connects a pair of corresponding pixels in both images.

overlapping images. Often hundreds of corresponding pixel pairs can be found on an image pair (only a handful are shown in the figure, for the sake of clarity).

More formally, the interest point matching problem is defined as: given two images $\mathbf{I}$ and $\mathbf{I}'$, find a set of pairs of corresponding points on the images, such that, for each pair of image points $(\mathbf{x}, \mathbf{x}')$, $\mathbf{x}$ and $\mathbf{x}'$ correspond to the projection of the same 3D point on the images $\mathbf{I}$ and $\mathbf{I}'$, respectively.

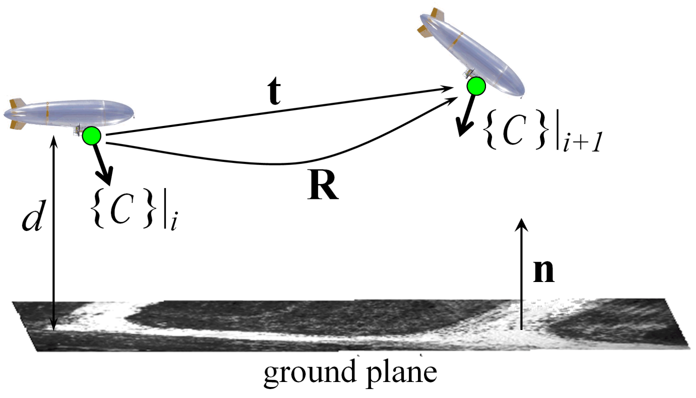
Over the years, this problem has received a lot of attention, and it is still challenging, as there are many variable conditions which make the recognition of the same point as seen in the two images be difficult. A good algorithm should be *invariant*, i.e., able to return the same results, detecting and matching the same points, in face to environmental changes such as illumination and blur, and to differences on the image caused by the difference on the camera position in the world: scaling, translation, rotation, and even changes of viewpoint and affine transformations in general.

For the experiments reported in this thesis, initially the *SIFT* [Lowe, 2004] algorithm was used. After, the newer *SURF* [Bay et al., 2006] algorithm was used, as it offers very similar performance in our experiments, and executes much faster.

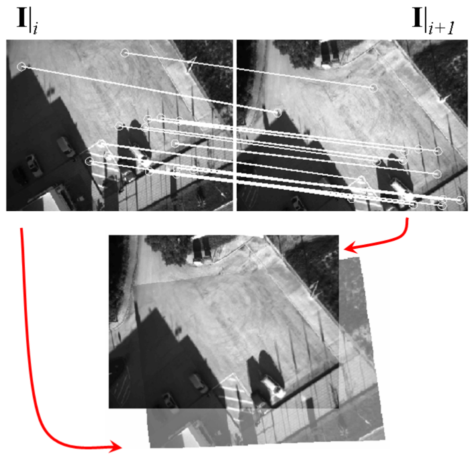
All interest point algorithms find wrong matches, among the set of matched pixel pairs. Robust techniques such as RANSAC [Fischler and Bolles, 1981] are often used to filter out these wrongly paired points, which are called *outliers*, leaving a set of consistently matched pixel pairs, called *inliers*. Some kind of geometric model is necessary to provide a criterion to discriminate the outliers - the robust estimation algorithm selects and considers only the largest possible set of inliers, i.e., of points which are consistent with the geometric model.

2.2.2 Planar surfaces and homographies

Consider a 3D plane imaged by two identical cameras placed in different positions. Consider also a set of pixel correspondences belonging to that plane in the form of



(a) A 3D plane is imaged by a moving camera.



(b) Above, two aerial images with some pixel correspondences. Below, the same images registered with a homography.

Figure 2.4: A 3D plane imaged by a moving camera induces a homography.

pairs of pixel coordinates $(\mathbf{x}, \mathbf{x}')$, where each pair corresponds to the projection of the same 3D point into each view. A *homography* represented by a 3×3 matrix $\mathbf{H}$ relates these two sets of homogeneous pixel coordinates such that $\mathbf{x}' = \mathbf{H}\mathbf{x}$, and the homography is said to be *induced* by the 3D plane [Hartley and Zisserman, 2000]. The homography can be recovered from pixel correspondences, and it is related to the 3D plane normal $\mathbf{n}$, the distance from the camera center to the plane d, and to the relative camera poses represented by the two camera projection matrices $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$ and $\mathbf{P}' = [\mathbf{R}|\mathbf{t}]$, as shown in figure 2.4, by:

$$\mathbf{H}_\lambda = \lambda \mathbf{H} = \lambda \left(\mathbf{R} - \mathbf{t}\mathbf{n}^T / d \right) \tag{2.4}$$

where $\mathbf{R}$ is rotation matrix and $\mathbf{t}$ a translation vector. The arbitrarily scaled matrix $\mathbf{H}_\lambda = \lambda \mathbf{H}$ is recovered first, and then the scale factor λ must be recovered. The scale λ is equal to the second largest singular value of $\lambda \mathbf{H}$, up to sign, as shown in appendix B.1. The correct sign of λ is recovered by imposing a positive depth constraint [Ma et al., 2004].

Defining $\mathbf{H} = \mathbf{H}_\lambda / \lambda$, the normalized homography matrix $\mathbf{H}$ is then decomposed into $\mathbf{R}$, $\mathbf{n}$, and ${}^{\mathbf{t}}/_{d}$ [Ma et al., 2004]. The relative pose recovered has an inherent scale ambiguity, as the translation magnitude is not recovered, only the ratio ${}^{\mathbf{t}}/_{d}$. The recovered homography can be used to register the image pair by applying the recovered transformation $\mathbf{H}$ to the first image. Figure 2.4(b) shows, above, an example of a pair of aerial images with some corresponding pixel pairs, and below, the same images registered.

2.2.2.1 Pure rotation case

The *infinite homography* $\mathbf{H}_\infty$ is a special case: it is the homography induced by the plane at infinity. It is also the homography between two images taken from two cameras *at the same position* (i.e., no translation, $\mathbf{t} = \mathbf{0}$), but rotated by a rotation represented by the matrix $\mathbf{R}$. The infinite homography can also be used to synthesize a virtual view from a non-existent virtual camera, at a desired orientation, given the appropriate rotation matrix.

The infinite homography is calculated by a limiting process where d approaches infinity, or the translation $\mathbf{t}$ tends to zero. In both cases the ratio ${}^{\mathbf{t}}/_{d}$ tends to zero in equation (2.4):

$$\mathbf{H}_\infty = \lim_{{}^{\mathbf{t}}/_{d} \to \mathbf{0}} \mathbf{H} = \lim_{{}^{\mathbf{t}}/_{d} \to \mathbf{0}} \mathbf{K}(\mathbf{R} + \frac{\mathbf{t}}{d}\mathbf{n}^T)\mathbf{K}^{-1} = \mathbf{K}\mathbf{R}\mathbf{K}^{-1} \tag{2.5}$$

2.2.3 Pure translation case

This section reviews computer vision results for the special case of pure translation movement.

2.2.3.1 Planar Homologies

Equation (2.4) of section 2.2.2 is valid for general camera motion, involving the relative camera rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$. In the translation-only case, $\mathbf{R} = \mathbf{I}$ and the homography becomes a *planar homology.*

A planar homology $\mathbf{G}$ is a planar perspective transformation with the property that there is a line, called the *axis*, such that every point on this line is a fixed point[1], and there is another fixed point not on the axis (the *vertex*) . A 3D plane imaged under pure translation induces a planar homology, where the axis is the image of the vanishing line of the plane (the intersection of the 3D plane and the plane at infinity), and the vertex is the epipole, or the FOE. Among the properties of homologies [Hartley and Zisserman, 2000, van Gool et al., 1998], we recall:

- Lines joining corresponding points intersect at the vertex, and corresponding lines (lines joining two pairs of corresponding points) intersect at the axis.

[1]A fixed point is a point $\mathbf{x}$ such that $\mathbf{G}\mathbf{x} = \mathbf{x}$, i.e., a point that is not changed by the transformation

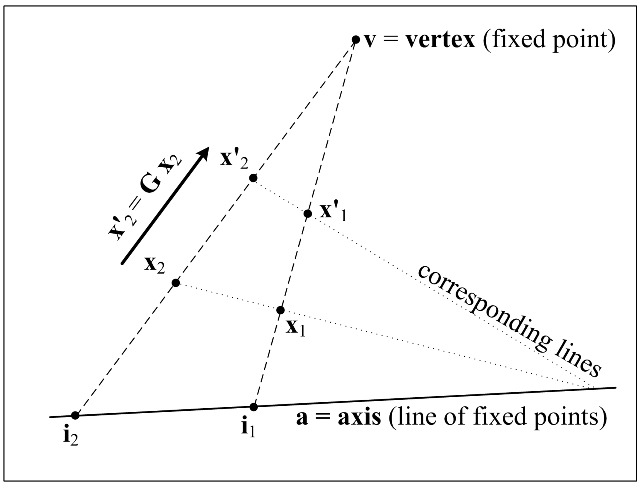


Figure 2.5: An example of a homology planar transformation with two corresponding pairs of points in the plane.

- The cross ratios defined by the vertex, a pair of corresponding points $(\mathbf{x}, \mathbf{x}')$, and the intersection of the line joining this pair with the axis (named as $\mathbf{i}$), have the same value for all points. Define a homology parameter μ as $\mu = crossratio(\mathbf{i}, \mathbf{x}, \mathbf{x}', \mathbf{v}) + 1$ such that:

$$(\mu - 1) = crossratio(\mathbf{i}, \mathbf{x}, \mathbf{x}', \mathbf{v}) = \frac{|\mathbf{x}\mathbf{x}'| \cdot |\mathbf{i}\mathbf{v}|}{|\mathbf{i}\mathbf{x}| \cdot |\mathbf{x}'\mathbf{v}|} \tag{2.6}$$

- The homology matrix $\mathbf{G}$ may be defined from its axis, vertex, and cross-ratio parameters:

$$\mathbf{G} = \mathbf{I} + (\mu - 1)\frac{\mathbf{v}\mathbf{a}^T}{\mathbf{v}^T\mathbf{a}} \tag{2.7}$$

where $\mathbf{v}$ is the vertex coordinates, and $\mathbf{a}$ the axis line. Figure 2.5 illustrate these entities and properties and shows a pair of points transformed by a homology in general configuration.

Under pure translation, each plane in 3D space induces a homology, where the vertex depends only on the camera movement and the axis depends on the relative orientation of the camera and the 3D plane. For the homologies utilized in section 3.2.3, the inducing plane is parallel to the image plane and the axis is the infinite line, and therefore corresponding lines are parallel (i.e., intersect in the infinite line).

2.2.3.2 Calculating the depth of a 3D point from ratios of distances on the image

The scene depth of individual 3D points is related with ratios of distances measured on the image [Arnspang et al., 1999]. Consider the images of two scene points and the vanishing point of the line defined by the two points, viewed by a single camera in any position, as shown in figure 2.6.

Denoting $\mathbf{x}$ and $\mathbf{x}'$ as the images of the two scene points $\boldsymbol{X}$ and $\boldsymbol{X}'$, $\mathbf{v}$ as the vanishing point of their connecting line, and Z and Z' as the scene depth of $\boldsymbol{X}$ and $\boldsymbol{X}'$, and $(b+c)$ and c as the *image* distances between $\mathbf{x}$ and $\mathbf{x}'$ to the vanishing point, the following equation holds:

$$\frac{Z'}{Z} = \frac{c}{b+c} = \frac{dist(\mathbf{x}, \mathbf{v})}{dist(\mathbf{x}', \mathbf{v})} \tag{2.8}$$

where $dist()$ means distance measured in the image. These entities are shown in figure 2.6, and equation (2.8) is written down as:

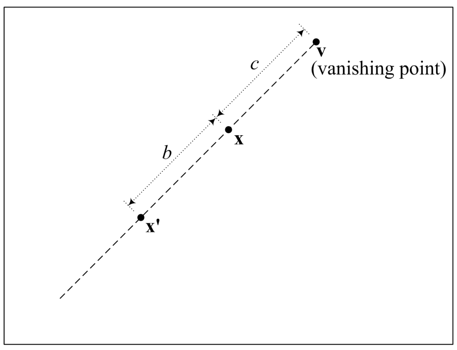
The relative scene depth of two points equals the reciprocal ratio of the image plane distances to the vanishing point of their connecting line [Arnspang et al., 1999].

This fundamental result will be applied in section 3.2.3 for a different configuration where two cameras with the same orientation image a single 3D point. The image projections are the same, but instead of having two scene points as in this case, in section 3.2 it is the camera which is moved under pure translation and images a single point twice. The equivalent configuration is shown in figure 3.6, where virtual cameras under pure translational motion image points in a 3D plane. Note that the problem of determining the vanishing point is not addressed in this section. Nevertheless, equation (2.8) is used in section 3.2 where the vanishing point is the FOE, which is estimated as presented in section 2.2.3.3.

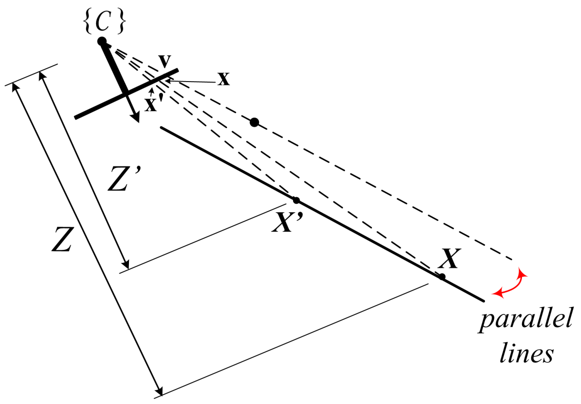
Absolute depth can be calculated for the configuration shown in figure 2.7(a): two cameras with the same orientation but located at different positions (the two image planes are parallel but not coincident, and they are separated by a distance δ) image the pair of scene points $\boldsymbol{X}$ and $\boldsymbol{X}'$. The previous notation is extended with f and r suffixes denoting the "front" and "rear" cameras to differentiate between the images of the points in the two views and the scene depths of the points relative to the two cameras, as seen on figure 2.6(b). With two views epipolar lines can be drawn to determine the epipole $\mathbf{e}$.

For this configuration the depths Z_l and Z_r, for any given scene point, are related by the following equation, :

$$Z_{rl} = Z_f - \delta \tag{2.9}$$

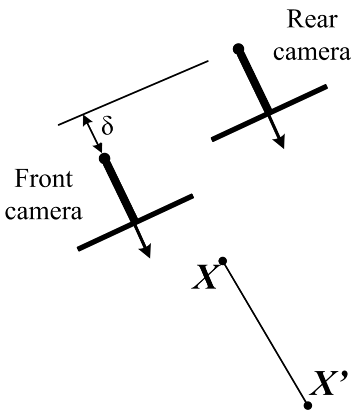


(a) The image of two 3D points and the vanishing point.

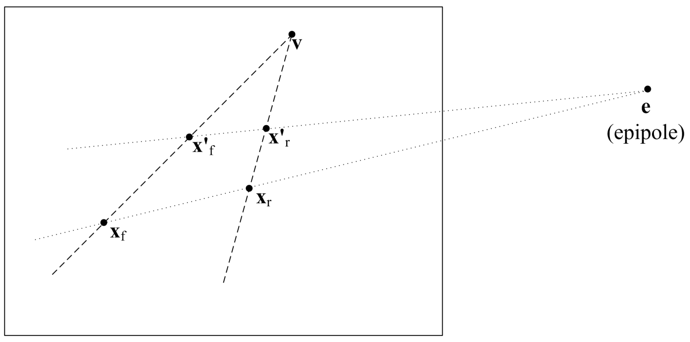


(b) Diagram with the 3D points and their image projections.

Figure 2.6: A camera images a pair of 3D points. The depth ratio is related with a ratio of distances on the image (equation (2.8)).



(a) Two parallel cameras image a pair of points.



(b) The two images superimposed.

Figure 2.7: Relating absolute scene depth with ratios of distances on the image for a pair of cameras and a pair of points.

Now, applying the anterior result on both views, we get the following non-singular system of equations:

$$\begin{pmatrix} (b_r + c_r) & -c_r \\ (b_f + c_f) & -c_f \end{pmatrix} \begin{pmatrix} Z_{1r} \\ Z_{2f} \end{pmatrix} = \begin{pmatrix} 0 \\ -\delta b_f \end{pmatrix} \tag{2.10}$$

Therefore, if the image plane separation δ is known, the image distances from the two views (b_r, c_r, b_f, c_f) may be used to determine the absolute depth. Note that for frontal binocular views (i.e., two cameras with coincident image planes), this system of equations is not regular.

After solving the above system of equations the following solution is obtained:

$$Z_{1r} = \frac{-\delta b_f c_r}{b_f c_r - b_r c_f} \tag{2.11}$$

$$Z_{2r} = \frac{-\delta b_f (b_r + c_r)}{b_f c_r - b_r c_f} \tag{2.12}$$

These equations offer an alternative to classical depth from stereo for computer vision, and they do not depend on focal length, coordinate disparities nor coordinate scale. They depend on an accurate determination of the distance between the image planes (the "baseline"), of the corresponding points image coordinates on the two views, and of their vanishing point. It is supposed that the line joining the points $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ is not parallel to the image planes, otherwise the epipolar lines would be parallel, the epipole would be on infinity, and the two lines joining their images $\overline{\mathbf{p}_{1l}\mathbf{p}_{2l}}$ and $\overline{\mathbf{p}_{1r}\mathbf{p}_{2r}}$ would not share a vanishing point.

2.2.3.3 Recovering the FOE from pixel correspondences.

Under pure translation, all lines connecting corresponding pixel pairs intersect in the FOE. This is a property of the homologies (section 2.2.3.1), but it is true for all corresponding pixels independently of the presence of 3D planes. An initial FOE estimate may be obtained from the pixel correspondences by solving a linear system.

To form the linear system, one constraint is defined for each corresponding pixel pair $(\mathbf{x}, \mathbf{x}')$: as the FOE must lie in the line $\mathbf{l} = \mathbf{x} \times \mathbf{x}'$, where $\times$ represents vector cross product, the following must be satisfied:

$$(\mathbf{x} \times \mathbf{x}') \cdot \mathbf{v} = \mathbf{0} \tag{2.13}$$

Stacking these constraints for n corresponding pixel pairs, we have:

$$\begin{cases} (\mathbf{x}_1 \times \mathbf{x'}_1) \cdot \mathbf{v} = \mathbf{0} \\ (\mathbf{x}_2 \times \mathbf{x'}_2) \cdot \mathbf{v} = \mathbf{0} \\ \vdots \\ (\mathbf{x}_n \times \mathbf{x'}_n) \cdot \mathbf{v} = \mathbf{0} \end{cases} \tag{2.14}$$

Then, given two or more corresponding pixel pairs, this linear system is solved to find a first estimate for the FOE $\mathbf{v}$. Then an optimization routine improves this estimate by considering that the fundamental matrix is $\mathbf{F} = [\mathbf{v}]_\times$ ($[\mathbf{v}]_\times$ indicates the skew symmetric matrix built from the entries of $\mathbf{v}$), and minimizing $\mathbf{x'}^T\mathbf{F}\mathbf{x}$ for all corresponding pixel pairs. If there were no error, the linear system would have an exact solution and $\mathbf{x'}^T\mathbf{F}\mathbf{x}$ would be zero for all corresponding pixel pairs. This process, including the usage of RANSAC [Fischler and Bolles, 1981] to exclude outliers on the pixel correspondences, and normalization terms for the optimization, is detailed in [Chen et al., 2003], with results using real images.

The FOE estimation is used in the experiments of chapter 3 and to exclude outliers from the set of corresponding pixel pairs.

2.3 Attitude Heading Reference System

The Attitude Heading Reference Systems (AHRS) used in this thesis are the models MTi and MTB-9 manufactured by the XSens company [XSens Tech., 2007]. The MTB-9 is shown in figure 1.2. Both consist on a set of sensors mounted together, namely three-axis accelerometers, gyroscopes, and magnetometers, plus a thermometer. It outputs direct readings from all its internal sensors, and also filtered, drift-less absolute orientation, obtained by fusing the readings of all internal sensors, considering variations with temperature. The internal sensors are calibrated in factory.

When the AHRS is not moving, the gravity acceleration is directly measured by the accelerometers, in the form of a 3D vector with $9.81[m/s^2]$ magnitude. If the magnitude is different, then the AHRS is moving and the acceleration measured has an additional component due to the AHRS´s own motion.

The three-axis magnetometers measure the direction and magnitude of the magnetic field around the AHRS. The results are normalized to the earth magnetic field (which is constant for a given location on earth), therefore if the measured magnitude is close to 1, then the earth magnetic field is being measured.

In indoor environments, the earth magnetic field is often distorted due to ferromagnetic objects, electric equipment or power lines, and thus the magnetic field readings may be heavily distorted. This distortion can be detected as a variation on magnetic field magnitude, and if the magnetic field can not be measured, then the AHRS heading angle can not be trusted. In outdoor environments this is a less important issue, as when the AHRS is mounted on a free flying UAV, provided that it is isolated from electromagnetic fields generated by electronic equipment or large currents, and that the gondola structure is made of non-ferromagnetic materials such as aluminum.

Figure 2.8 shows the value of the magnetic vector module with the AHRS moving in three different environments: inside a laboratory room, over the roof of a car driving on campus, and finally in an outdoor environment. The magnetic field measurements are normalized, therefore the earth magnetic field magnitude is represented with the value 1. In the most polluted environment, the magnetic vector module reaches 50% of the expected nominal value.

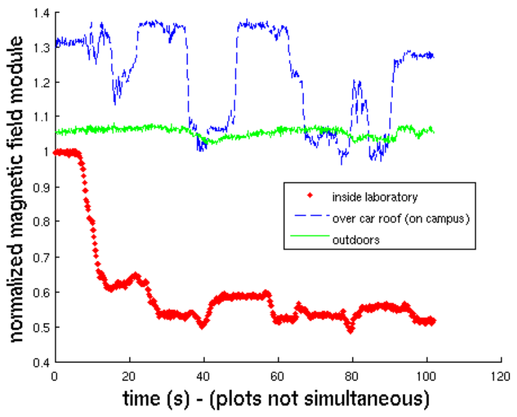


Figure 2.8: The module of the magnetic vector in different environments.

2.4 Laser Range Finder

The *Laser Range Finder* (LRF) sensor measures the time of return for a light signal emitted and reflected by the environment. The laser beam mounted on a rotating axis inside the sensor, performing a series of range readings separated by a constant angular interval. This correspond to a set of 3D points called a "2D scan", because all the 3D points are on the plane defined by the coplanar laser beams.

The LRF itself is mounted on a pantilt, as shown in figure 2.9, and by moving the tilt axis between each LRF "2D scan", a set of 3D points is obtained covering the space around the LRF device, and this set is called a "3D scan".

The 3D scan is a set of 3D points ${}^{\mathcal{L}}\mathbb{P}|_i$, with coordinates expressed in the laser frame of reference $\{\mathcal{L}\}|_i$, which has origin on the tilt rotation axis of the pantilt, directly below the center of projection of the laser beams. The 3D positions of each point are given by a laser projection model as defined in [Scaramuzza et al., 2007]:

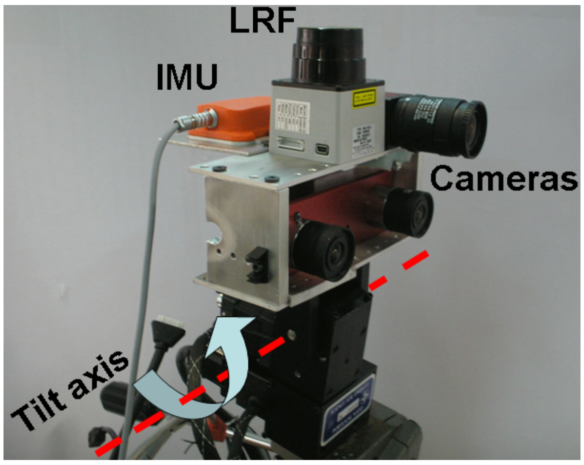


Figure 2.9: Sensors utilized for mapping: AHRS, LRF, stereo and single cameras.

for each point the pantilt position is defined by the tilt angle φ, as the pan axis is not moved, and the LRF supplies a distance measurement ρ and a beam angle θ in the 2D scan plane for each range reading. Thus the projection of a range reading into a 3D point ${}^{\mathcal{L}}\boldsymbol{X} = (X, Y, Z)^T$ in the $\{\mathcal{L}\}|_i$ frame is given by:

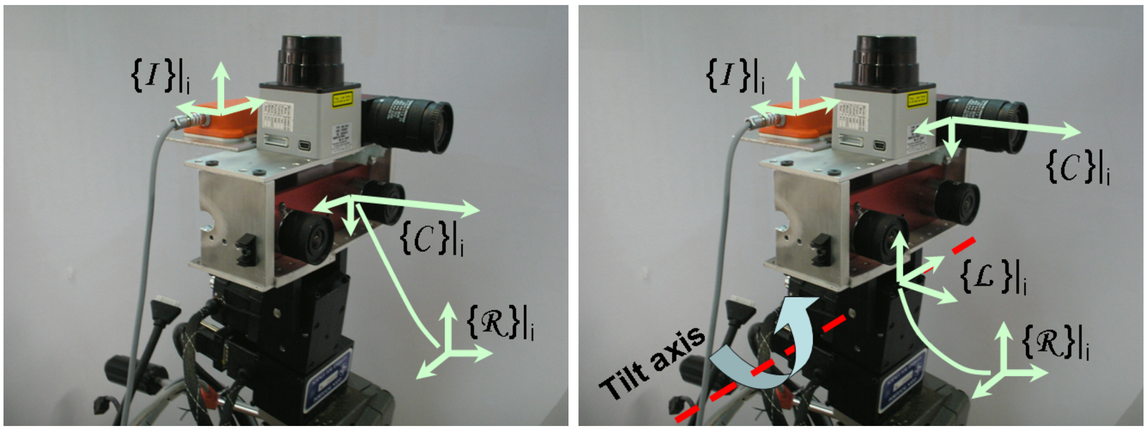
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} c_\varphi c_\theta & -c_\varphi s_\theta & s_\varphi & c_\varphi d_x + s_\varphi d_z \\ s_\theta & c_\theta & 0 & 0 \\ -s_\varphi c_\theta & s_\varphi s_\theta & c_\varphi & -s_\varphi d_x + c_\varphi d_z \end{bmatrix} \begin{bmatrix} \rho \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.15}$$

where c and s represent the cosine and sine functions. This projection takes into account the distance between the pantilt rotation axis and the emission/sensing point of the laser beams.

2.5 Frames of reference

An AHRS is rigidly mounted together with a camera (single camera or stereo pair). The AHRS estimates the orientation of the rigid body containing itself and the camera(s). For the experiments using the stereo camera (figure 2.9(a)) the LRF and monocular camera are not used. For the experiments using a LRF the AHRS is rigidly mounted with a monocular camera and the LRF as in figure 2.9(b). For the experiments using datasets from the airship, the camera-AHRS system is shown in section 1.4 and figures 1.2 and 2.11.

The stereo cameras provide intensity images $\mathbf{I}_l(x, y)|_i$ and $\mathbf{I}_r(x, y)|_i$, where x and y are pixel coordinates, r and l represent the right and left cameras, and i the frame



(a) The stereo camera scenario (b) The LRF scenario

Figure 2.10: Definitions of frames of reference.

time index. A single camera provides only one intensity image $\mathbf{I}(x, y)|_i$. All cameras are calibrated [Bouguet, 2006], and their intrinsic parameter matrix $\mathbf{K}$ is known. We henceforth define the following reference frames, as shown in figure 2.10:

- **Camera Frame** $\{\mathcal{C}\}|_i$: This is the reference frame used in the common pinhole camera projection model. The origin is placed at the *camera center*, the axis z points to the front (the *depth* axis), and the axes x and y form the *image plane*, where, on the image, x points right and y points down. In the stereo camera case, $\{\mathcal{C}\}|_i$ is defined as referring only to the left camera.

- **Inertial Sensor Frame** $\{\mathcal{I}\}|_i$: It is defined by the internal sensor axes of the AHRS. The AHRS orientation estimate is a rotation which aligns the axes of $\{\mathcal{I}\}|_i$ to the $\{\mathcal{W}\}$ world frame axes. As the AHRS is mounted rigidly together with the camera, the transformation between the frames $\{\mathcal{I}\}|_i$ and $\{\mathcal{C}\}|_i$ is constant for all time indexes i.

- **World Frame** $\{\mathcal{W}\}$: This is a LLA (Latitude Longitude Altitude) frame, where its axes correspond to the local north, east and up directions. Its origin is an arbitrarily set point, often the origin of the first camera frame.

- **Laser Frame** $\{\mathcal{L}\}|_i$ Its origin is the tilt rotation axis of the pantilt, directly below the center of projection of the laser beams. Its axes are aligned with the laser beams as shown in figure 2.10.

- **Rotated Camera Frame** $\{\mathcal{R}\}|_i$: This frame shares its origin with the $\{\mathcal{C}\}|_i$ frame, but its axes are aligned with the world frame $\{\mathcal{W}\}$. Figure 2.10 shows

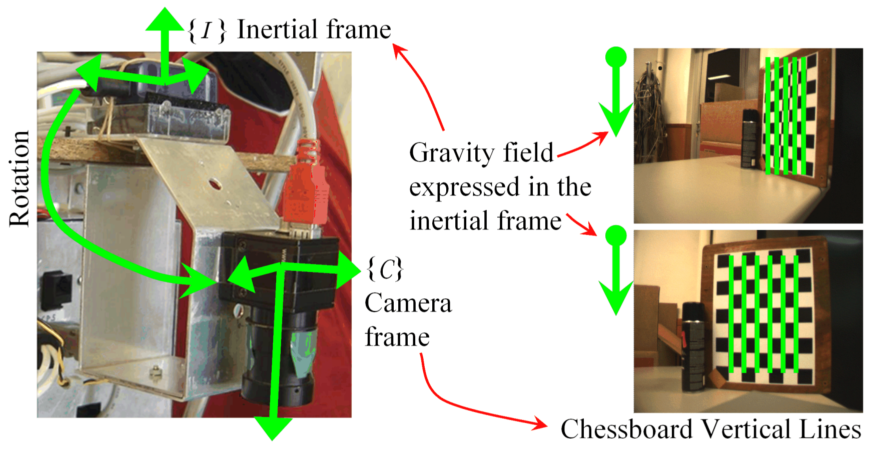

{I} Inertial frame
Rotation
Gravity field expressed in the inertial frame
{C} Camera frame
Chessboard Vertical Lines

Figure 2.11: The calibration recovers the rotation between the camera and inertial sensor frames. The camera and AHRS measure the gravity direction. Two example images with vertically placed chessboards used in the calibration are shown in the right.

this frame on a different position than the $\{\mathcal{C}\}|_i$ frame only to increase the clarity of the drawing.

- **Virtual Downwards Camera** $\{\mathcal{D}\}|_i$: This is a camera frame, which shares its origin with the $\{\mathcal{C}\}|_i$ frame, but its optical axis points down, in the direction of gravity, and its other axes (i.e., the image plane) are aligned with the north and east directions.

2.6 Calibration of Camera - Inertial System

The camera is first calibrated to determine its intrinsic parameter matrix and lens distortion parameters [Bouguet, 2006]. A subsequent calibration routine [Lobo and Dias, 2007, Lobo and Dias, 2005, Alves et al., 2003] finds the rigid body rotation between the $\{\mathcal{I}\}$ and $\{\mathcal{C}\}$ frames and is performed by having both sensors observing the gravity vector, as shown in figure 2.11. The camera observes vertical vanishing points from a vertically placed chessboard target, and the AHRS measures the acceleration vector from its accelerometers. If the AHRS is not moving, the sensed acceleration represents the gravity.

The output of this calibration is a rotation matrix ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$ that brings a point from the $\{\mathcal{C}\}|_i$ frame into the $\{\mathcal{I}\}|_i$ frame for all i. There is also a process to find the translational component of the transformation, but in this thesis the translation is considered as zero, as the distance between the camera and AHRS is negligible.

Two examples of calibration images with a vertically placed chessboard are shown in figure 2.11. Each image must be taken from a different point of view and the camera must be immobile while taking each shot. The assembly of camera and AHRS must be kept rigid, but the camera orientation is not fixed or restricted during this off-line calibration process. The same set of images may have been used previously for the calibration of the camera's intrinsic parameters.

To obtain the rotation matrix ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$, two sets of gravity vector measurements ${}^{\mathcal{I}}\mathbf{v}_i$ and ${}^{\mathcal{C}}\mathbf{v}_i$ on the $\{\mathcal{I}\}|_i$ and $\{\mathcal{C}\}|_i$ frames, respectively, are measured. Then, the unit quaternion $\mathring{\mathbf{q}}$ that best rotates one set of measurements to be registered with the other is determined by maximizing the following equation [Alves et al., 2003]:

$$\sum_{i=1}^{n}(\mathring{\mathbf{q}} \cdot {}^{\mathcal{I}}\mathbf{v}_i \cdot \mathring{\mathbf{q}}^*) \cdot {}^{\mathcal{C}}\mathbf{v}_i \tag{2.16}$$

where n is the number of measurements, which can be rewritten as:

$$\sum_{i=1}^{n}(\mathring{\mathbf{q}}\, {}^{\mathcal{I}}\mathbf{v}_i) \cdot ({}^{\mathcal{C}}\mathbf{v}_i\, \mathring{\mathbf{q}}) \tag{2.17}$$

The quaternion product can be expressed as a matrix. Defining ${}^{\mathcal{I}}\mathbf{v}_i = ({}^{\mathcal{I}}x_i, {}^{\mathcal{I}}y_i, {}^{\mathcal{I}}z_i)^T$ and ${}^{\mathcal{C}}\mathbf{v}_i = ({}^{\mathcal{C}}x_i, {}^{\mathcal{C}}y_i, {}^{\mathcal{C}}z_i)^T$, we have:

$$\mathring{\mathbf{q}} \cdot {}^{\mathcal{I}}\mathbf{v}_i = \begin{bmatrix} 0 & -{}^{\mathcal{I}}x_i & -{}^{\mathcal{I}}y_i & -{}^{\mathcal{I}}z_i \\ {}^{\mathcal{I}}x_i & 0 & {}^{\mathcal{I}}z_i & -{}^{\mathcal{I}}y_i \\ {}^{\mathcal{I}}y_i & -{}^{\mathcal{I}}z_i & 0 & {}^{\mathcal{I}}x_i \\ {}^{\mathcal{I}}z_i & {}^{\mathcal{I}}y_i & -{}^{\mathcal{I}}x_i & 0 \end{bmatrix} \mathring{\mathbf{q}} = {}^{\mathcal{I}}\mathbf{V}_i \cdot \mathring{\mathbf{q}} \tag{2.18}$$

and

$${}^{\mathcal{C}}\mathbf{v}_i \cdot \mathring{\mathbf{q}} = \begin{bmatrix} 0 & -{}^{\mathcal{C}}x_i & -{}^{\mathcal{C}}y_i & -{}^{\mathcal{C}}z_i \\ {}^{\mathcal{C}}x_i & 0 & -{}^{\mathcal{C}}z_i & {}^{\mathcal{C}}y_i \\ {}^{\mathcal{C}}y_i & {}^{\mathcal{C}}z_i & 0 & -{}^{\mathcal{C}}x_i \\ {}^{\mathcal{C}}z_i & -{}^{\mathcal{C}}y_i & {}^{\mathcal{C}}x_i & 0 \end{bmatrix} \mathring{\mathbf{q}} = {}^{\mathcal{C}}\mathbf{V}_i \cdot \mathring{\mathbf{q}} \tag{2.19}$$

Using the matrix form of the quaternion product and substituting in (2.17):

$$\sum_{i=1}^{n}({}^{\mathcal{I}}\mathbf{V}_i\, \mathring{\mathbf{q}}) \cdot ({}^{\mathcal{C}}\mathbf{V}_i\, \mathring{\mathbf{q}}) \tag{2.20}$$

or

$$\sum_{i=1}^{n} \mathring{\mathbf{q}}^T\, {}^{\mathcal{I}}\mathbf{V}_i {}^{\mathcal{C}}\mathbf{V}_i\, \mathring{\mathbf{q}} \tag{2.21}$$

Now factoring out $\overset{\circ}{\mathbf{q}}$ we get:

$$\overset{\circ}{\mathbf{q}}^T \left(\sum_{i=1}^{n} {}^{\mathcal{I}}\mathbf{V}_i {}^{\mathcal{C}}\mathbf{V}_i \right) \overset{\circ}{\mathbf{q}} \tag{2.22}$$

So we must find the unit quaternion $\overset{\circ}{\mathbf{q}}$ that maximizes:

$$\max \overset{\circ}{\mathbf{q}}^T \ \mathbf{N} \ \overset{\circ}{\mathbf{q}} \tag{2.23}$$

where $\mathbf{N} = \sum_{i=1}^{n} {}^{\mathcal{I}}\mathbf{V}_i {}^{\mathcal{C}}\mathbf{V}_i$. To express $\mathbf{N}$ concisely, we define:

$$S_{xx} = \sum_{i=1}^{n} {}^{\mathcal{I}}x_i {}^{\mathcal{C}}x_i \, , \, S_{xy} = \sum_{i=1}^{n} {}^{\mathcal{I}}x_i {}^{\mathcal{C}}y_i \tag{2.24}$$

and analogously for all 9 pairings of components of the two vectors. Therefore $\mathbf{N}$ can be expressed by equation (2.25). These sums contain all information needed to find the solution. Since $\mathbf{N}$ is a symmetric matrix, the solution is the four vector $\overset{\circ}{\mathbf{q}}_{max}$ corresponding to the largest eigenvalue of $\mathbf{N}$ [Horn, 1987].

$$\mathbf{N} = \begin{bmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & (-S_{xx}+S_{yy}-S_{zz}) & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & (-S_{xx}-S_{yy}+S_{zz}) \end{bmatrix} \tag{2.25}$$

Note that the camera-inertial calibration deals with a monocular camera. For the stereo camera only data from the left camera are used in this calibration.

The rotation from the camera frame to the inertial sensor frame for the camera-inertial system of figures 1.2 and 2.11 is:

$${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}} = \begin{bmatrix} -0.9998 & 0.0080 & 0.0206 \\ 0.0084 & 0.9998 & 0.0141 \\ -0.0206 & 0.0143 & -0.9998 \end{bmatrix} \tag{2.26}$$

2.7 The virtual horizontal plane concept or an inertial stabilized plane

The knowledge of the camera orientation provided by the AHRS orientation estimates allows the image to be reprojected on entities defined on an absolute geo-referenced

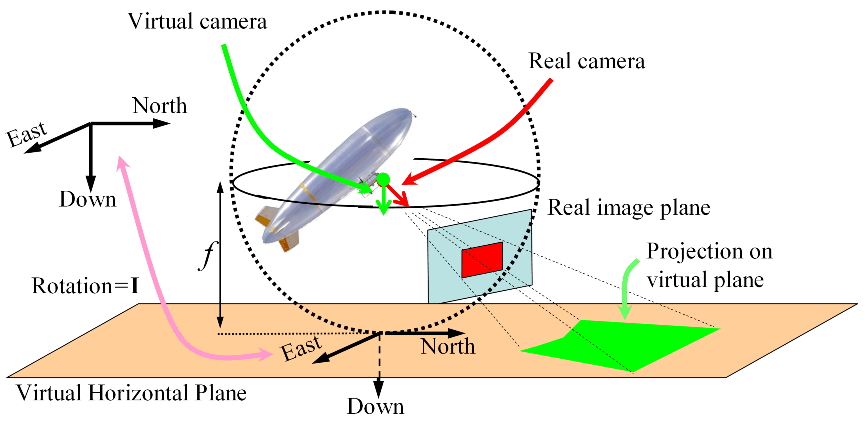


Figure 2.12: The virtual horizontal plane concept.

frame, such as a *virtual horizontal plane* (with normal parallel to gravity), at a distance f below the camera center, as shown in figure 2.12. Projection rays from 3D points to the camera center intersect this plane, reprojecting the 3D point into the plane. This reprojection corresponds to the image of a virtual camera with the same optical center as the original camera, but with optical axis coincident with the gravity vector. Section 3.2.1 details how to perform this reprojection.

Chapter 3

Inertial Aided Visual Trajectory Recovery

3.1 Introduction

In this chapter, the trajectory of a mobile observer is recovered from a sequence of images of a planar area. Orientation estimates from an AHRS compensate the rotational degrees of freedom and allow the usage of a translation-only model.

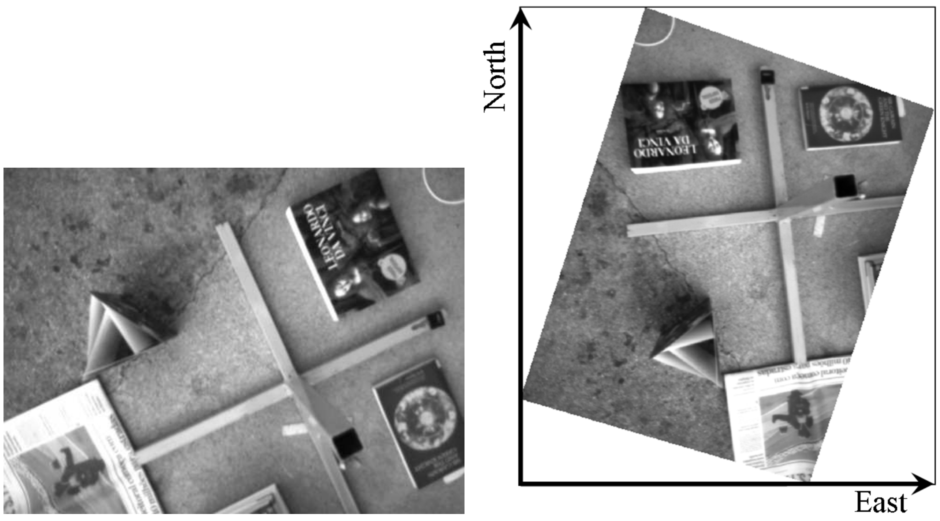
Firstly, the knowledge of camera orientation allows the images to be reprojected on a virtual horizontal plane, compensating the rotation, as presented in section 3.2.1. Then, the pure rotation models are introduced in sections 3.2.2 and 3.2.3. The case of non-horizontal planes is covered in section 3.3.

The results are presented in section 3.4. As the vertical component is of special importance for aerial vehicles and is recovered separately by the homology algorithm, experiments with images taken from a tripod are presented to evaluate the algorithms against hand measured ground truth. The chapter finishes by presenting trajectory recovery for the DIVA airship, and evaluating it by comparison with the common homography model and GPS. Visual odometry is also fused with GPS position fixes, and the maps generated when images are projected on the ground plane are also compared.

3.2 Visual Navigation

3.2.1 Reprojecting the images on the virtual horizontal plane

The first step of the processes of this chapter and of sections 4.2 and 5.2 is to reproject each image onto the virtual horizontal plane. In other words, a virtual view is generated, to compensate differences due to heading and viewpoint. Each image



(a) Original image (b) Warped image

Figure 3.1: Example of reprojection in the virtual horizontal plane

is transformed in order to appear as if it were taken from a virtual camera with the same camera center and intrinsic parameters, but looking down (i.e., having its optical axis coincident with the gravity vector), and heading north (i.e., having the image y axis parallel to the North-South direction). The camera height variation is not compensated, resulting in scale differences in the virtual images.

In figure 2.12, the image, shown in red into the real image plane, is reprojected into the image plane of the virtual camera, that is perpendicular to the gravity (shown in green). Figure 3.1 shows the reprojection of one image as an example. The cross in the ground is nearly aligned with the north and east axes, as it is seen on figure 3.1(b).

In this way, the camera rotation is compensated, and the relative pose between any pair of camera poses is reduced to a pure translation, as required by the models used in this thesis.

Another objective is to make interest point matching algorithms more robust, by relaxing the demands on the interest point detection, encoding and matching algorithms. Otherwise, it would be necessary that the feature encoding be invariant to heading and viewpoint differences. The usage of this reprojection during interest point matching in the image mosaicing context is evaluated in section 4.2 [Mirisola and Dias, 2007].

For the low altitude aerial dataset used in section 3.4.4, the time spent matching interest point descriptors using the reprojected images was 41% of the time required to perform the matching with the original, non-reprojected images. The number of correctly matched interest points was 10% smaller with the reprojected images. The time spent to generate the reprojected images must be discounted, but this on average it was four times smaller than the speedup obtained in descriptor matching. Besides that, reprojecting the images may be a necessary task itself, for example to generate the images drawn in figure 3.21(a).

This reprojection - sometimes called *pre-warping* - is already widely used to pre-process images taken from moving vehicles like submarines [Gracias, 2002]. The techniques presented in this thesis use only the reprojected coordinates of the matched interest points, and the actual generation of reprojected images is not strictly necessary. Nevertheless, most experiments in this thesis used reprojected images, because of the improvements shown in this section and section 4.2 and because it is easier to visualize the results and errors in the reprojected images.

3.2.1.1 The infinite homography.

The image is reprojected on the virtual horizontal plane by an infinite homography, using the rotation that transforms the $\{\mathcal{C}\}|_i$ frame into the $\{\mathcal{D}\}|_i$ frame. Thus the infinite homography is used to synthesize a virtual view from a non-existent virtual camera.

For each image $\mathbf{I}_i$, a simultaneous AHRS orientation estimate and the rotation matrix ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$ from the camera-inertial calibration are used to calculate the camera orientation in relation to the world frame, expressed by the rotation matrix ${}^{\mathcal{W}}\mathbf{R}_{\mathcal{C}}|_i$. This is shown in §4.4.2.

The virtual camera frame $\{\mathcal{D}\}|_i$ has its axes parallel to axes of the $\{\mathcal{W}\}$ and $\{\mathcal{R}\}|_i$ frames, but its optical axis points down. The rotation matrix ${}^{\mathcal{D}}\mathbf{R}_{\mathcal{W}}$ rotates from the $\{\mathcal{W}\}$ to the $\{\mathcal{D}\}|_i$ frame:

$$ {}^{\mathcal{D}}\mathbf{R}_{\mathcal{W}} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{3.1} $$

Then the rotation between the $\{\mathcal{D}\}|_i$ frame and the $\{\mathcal{C}\}|_i$ frame is calculated as ${}^{\mathcal{D}}\mathbf{R}_{\mathcal{C}}|_i = {}^{\mathcal{D}}\mathbf{R}_{\mathcal{W}} \cdot {}^{\mathcal{W}}\mathbf{R}_{\mathcal{C}}|_i$

Therefore the transformation used to reproject images into the virtual stabilized frame is:

$$ {}^{\mathcal{D}}\mathbf{H}_{\mathcal{C}}|_i \triangleq \mathbf{K} \cdot {}^{\mathcal{D}}\mathbf{R}_{\mathcal{C}}|_i \cdot \mathbf{K}^{-1} \tag{3.2} $$

3.2.1.2 Deleting image pixels from close to horizontal 3D rays.

In the previous section, images were reprojected into an image plane perpendicular to the vertical direction of gravity. But there is not any guarantee that the camera

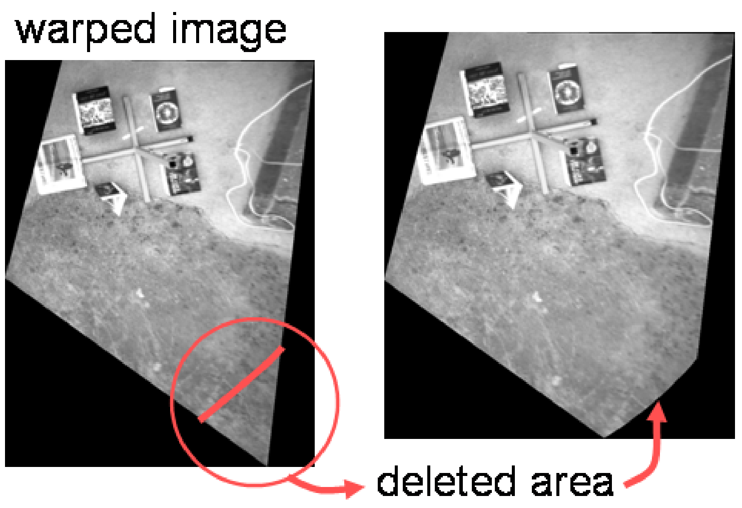


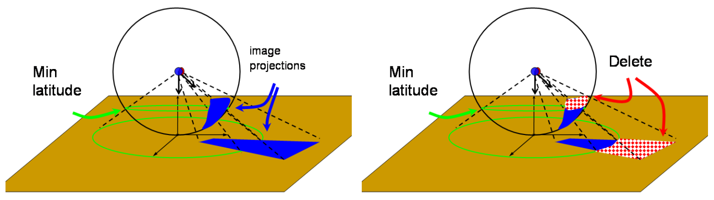
Figure 3.2: An example of a reprojected image before and after cutting the low latitude regions. The limit was set at 45°.

principal axis will be pointing close to the gravity direction. If it is not, then some areas on the reprojected images will be heavily distorted by the reprojection and have small resolution. This leads to low quality images and large memory consumption and computational time. Moreover, if the camera optical axis is close to horizontal, some pixel rays will be horizontal, rendering the reprojection infeasible.

As the camera orientation is known, it is possible to delete from the original image beforehand the pixels on these undesirable areas. It is also possible to delete these regions from an already reprojected image. Suppose an imaginary sphere, centered into the camera center, with radius equal to the camera focal length. Any image ray connecting a 3D point to the camera center, passing by the image plane, also intersects the surface of this sphere.

Supposing that the gravity direction is the south pole of this sphere, image pixels reprojected into the low latitude regions of this sphere (i.e., close to the equator) should be deleted, as it is shown in figure 3.3. An example of an image before and after the deletion of the low latitude regions is shown in figure 3.2.

The geometric process to find the pixels to be deleted from the image is given in algorithm 1. Note that the pixels can be found both on the original or in the reprojected image, and both cases are shown, although it is obviously preferable to eliminate the pixels already the original image, to avoid wasting computational resources. The key step is to find the *nadir point* on the image (the nadir point is the image projection of the 3D point that is exactly below the camera center, in a straight vertical line, in the world frame). As the $\{\mathcal{D}\}|_i$ frame corresponds to a virtual camera with a principal axis coincident with gravity, its nadir point is also its principal point.



(a) An image projected on the virtual sphere and image plane (b) Delete the low latitude regions (checkered)

Figure 3.3: Deleting pixels projected on the virtual sphere in low latitude regions.

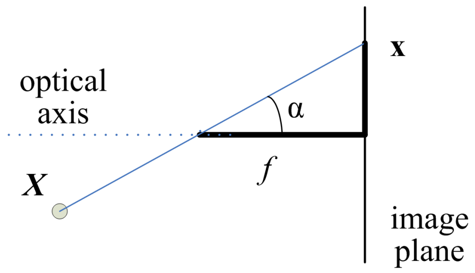


Figure 3.4: The relation between pixel coordinates and the angle of the projection ray with the optical axis.

Equation (3.3) in algorithm 1 is derived from the relation between pixel coordinates and the angle of the projection ray with the optical axis on a pinhole camera projection: supposing a 3D point $\boldsymbol{X}$ which is projected in the image plane in the 2D image point $\mathbf{x}$. The camera center, the camera principal point and $\mathbf{x}$ form a triangle as seen in figure 3.4. From this triangle, as the focal length in pixels f is known, and the distance in pixels from $\mathbf{x}$ to the principal point is also known, we extract $\tan\alpha = dist(\mathbf{x}, principal\, point)/f$, where α is the angle between the projection ray and the optical axis and $dist$ is the Euclidean distance on the image.

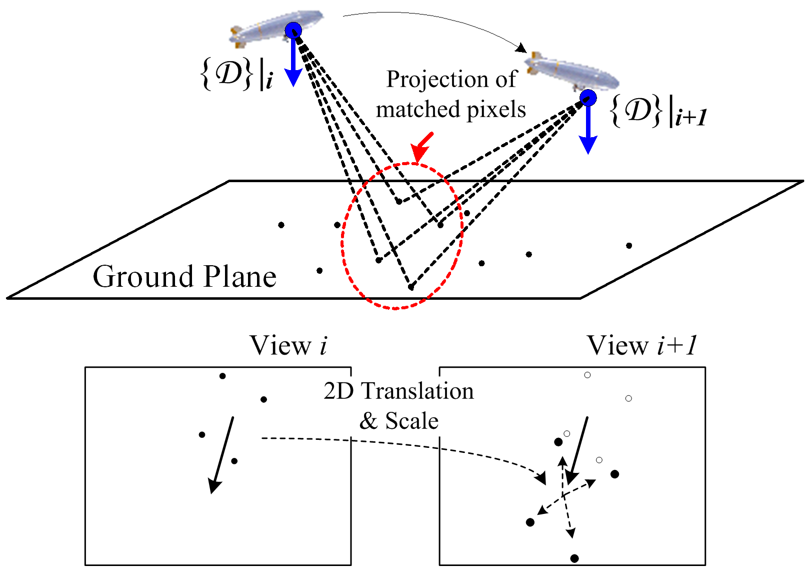
3.2.2 Procrustes solution to register scene points.

Suppose a sequence of aerial images of a horizontal ground patch, and that these images are reprojected on the virtual horizontal plane as presented in section 3.2.1.1. The virtual cameras have horizontal image planes parallel to the ground plane. Then, each corresponding pixel is projected into the ground plane, generating a 3D point, as shown in figure 3.5(a). Two sets of 3D points are generated for two successive views, and these sets are directly registered in scene coordinates. Indeed, as all points belong

Algorithm 1 Finding pixels to delete from the image.

```
// ${}^{\mathcal{C}}\mathbf{g} \leftarrow$ direction of gravity in the $\{\mathcal{C}\}|_i$ frame
${}^{\mathcal{C}}\mathbf{g} \leftarrow {}^{\mathcal{C}}\mathbf{R}_{\mathcal{W}}|_i \cdot \begin{pmatrix} 0 & 0 & -1 \end{pmatrix}^T$

// project ${}^{\mathcal{C}}\mathbf{g}$ on the image of the $\{\mathcal{C}\}|_i$ camera
${}^{\mathcal{C}}\mathbf{n} \leftarrow \mathbf{K} \cdot {}^{\mathcal{C}}\mathbf{g}$

// transform the nadir image point ${}^{\mathcal{C}}\mathbf{n}$ into the virtual projected image
${}^{\mathcal{D}}\mathbf{n} \leftarrow {}^{\mathcal{D}}H_{\mathcal{C}}|_i \cdot {}^{\mathcal{C}}\mathbf{n}$ // not needed if pixels are deleted from original $\{\mathcal{C}\}|_i$ image

// given the image $\mathbf{I}$ and the nadir point $\mathbf{n}$, which are either ${}^{\mathcal{C}}\mathbf{I}$, ${}^{\mathcal{C}}\mathbf{n}$ or ${}^{\mathcal{D}}\mathbf{I}$, ${}^{\mathcal{D}}\mathbf{n}$.

// define the maximum angle with the gravity direction
$\gamma \leftarrow MAX\_ANGLE\_GRAVITY$

// calculate maximum image distance with $\mathbf{n}$
$$max\_dist = f \cdot \tan\gamma \qquad (3.3)$$

Delete all pixels $\mathbf{p}$ on the image such that $dist(\mathbf{n}, \mathbf{p}) > max\_dist$
```

to the same ground plane, the registration is solved in 2D coordinates. Figure 3.5(b) shows a diagram of this process.



(a)

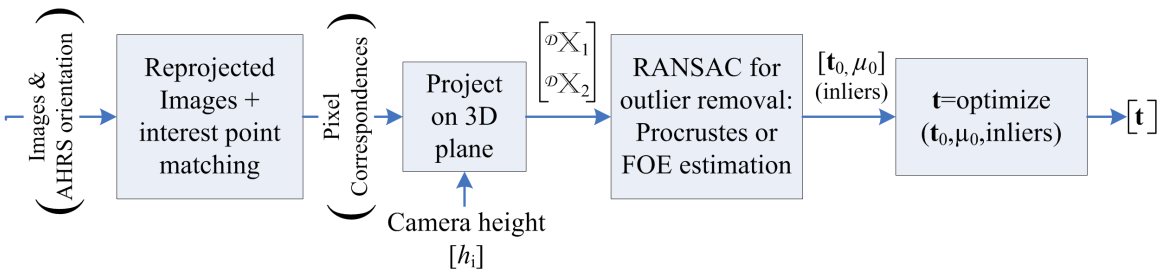

Images & AHRS orientation
Reprojected Images + interest point matching
Pixel Correspondences
Project on 3D plane
${}^{\mathcal{D}}X_1$ ${}^{\mathcal{D}}X_2$
Camera height $[h_i]$
RANSAC for outlier removal: Procrustes or FOE estimation
$[\mathbf{t}_0, \mu_0]$ (inliers)
$\mathbf{t}$=optimize ($\mathbf{t}_0$,μ_0,inliers)
$[\mathbf{t}]$

Figure 3.5: Finding the translation between successive camera poses by 3D scene registration.

Each corresponding pixel pair $(\mathbf{x}, \mathbf{x}')$ is projected by equation (3.4) yielding a pair of 3D points $(\boldsymbol{X}, \boldsymbol{X}')$, defined in the $\{\mathcal{D}\}|_i$ frame:

$$\boldsymbol{X} = \begin{bmatrix} \frac{(x_x - n_x)\cdot h_i}{f} \\ \frac{(x_y - n_y)\cdot h_i}{f} \\ h_i \end{bmatrix}, \quad \boldsymbol{X}'(\mathbf{t}) = \begin{bmatrix} \frac{(x'_x - n_x)\cdot(h_i - t_z/t_w)}{f} + \frac{t_x}{t_w} \\ \frac{(x'_y - n_y)\cdot(h_i - t_z/t_w)}{f} + \frac{t_y}{t_w} \\ h_i - t_z \end{bmatrix} \qquad (3.4)$$

where $\mathbf{x} = [x_x, x_y, 1]^T$, $\mathbf{x}' = [x'_x, x'_y, 1]^T$, again in inhomogeneous form, h is the camera height above the ground plane, $\mathbf{t}$ is defined as a four element homogenous vector $\mathbf{t} = [t_x, t_y, t_z, t_w]^T$. The $\mathbf{t}$ value which turns $\boldsymbol{X}'(\mathbf{t}) = \boldsymbol{X}$ is the translation which registers the $\{\mathcal{D}\}|_i$ and $\{\mathcal{D}\}|_{i+1}$ frames, and which must be determined. If there are n corresponding pixel pairs, this projection yields two sets of 3D points, $\mathbb{X} = \{\boldsymbol{X}_k | k = 1 \ldots n\}$ and $\mathbb{X}' = \{\boldsymbol{X}'_k | k = 1 \ldots n\}$

An initial, inhomogeneous, value for $\mathbf{t}_0$ is calculated by the *Procrustes* registration routine [Borg and Groenen, 1997, Gower and Dijksterhuis, 2004], which is described in appendix B.6. It finds the 2D translation and scale factor which register the two point sets taken as 2D points, yielding estimates the x and y components of $\mathbf{t}_0$ and of the scale factor μ_0. The inputs for the Procrustes routine are the configurations $\mathbb{X}$ and $\mathbb{X}'(\mathbf{0})$.

From μ_0 and the current estimate of the camera height an initial estimate the vertical component of $\mathbf{t}_0$ can be calculated, as $\mu_0 = (h_i - t_z)/h_i$. Outliers in the pixel correspondences are removed by embedding the Procrustes routine in a RANSAC procedure. Then $\mathbf{t}_0$ is used as an initial estimate for an optimization routine which minimizes the registration error between $\mathbb{X}$ and $\mathbb{X}'(\mathbf{t})$, estimating an updated and final value for $\mathbf{t}$.

This optimization variables are the four elements of $\mathbf{t}$, with equation (3.4) used to update $\mathbb{X}'(\mathbf{t})$. The function to minimize is:

$$\min_{(t_x, t_y, t_z, t_w)} \sum_{k=1\ldots n} dist\left(\boldsymbol{X}'_k\left(\mathbf{t}\right), \boldsymbol{X}_k\right) \tag{3.5}$$

where *dist* is a distance metric. Appendix B.4 shows the partial derivatives of this function with Euclidean distance as *dist*.

The same process could be performed with an inhomogeneous, three element $\mathbf{t}$. But, as it is the case with homography and homology estimation, the over-parameterization improves the accuracy of the final estimate and sometimes even the speed of convergence. In this case the extra dimension allows the length of the translation to change without changing its direction.

The optimization variables could also be (t_x, t_y, t_w, μ), with the translation being a 2D translation vector in homogeneous form, with μ as a scale factor and with the optimization performed in the virtual plane 2D coordinates. For the aerial image dataset, this second parameterization yielded worse results and was abandoned. The same remarks about overparameterization are again valid.

Nevertheless, for datasets such as the overpass dataset of section 3.4.2, where the actual camera orientation is almost constant, the error in the orientation estimate is less significant, and interest point matching is easier, not only both parameterizations obtain good results even without over-parameterization or homogeneous variables, but also the algebraic Procrustes procedure obtains good results alone, with no optimization at all, as it is shown in figure 3.16. Indeed, if the assumptions of having both image and ground planes parallel and horizontal are really true, with outliers removed, and considering isotropic error in the corresponding pixel coordinates, then it can be proved that the Procrustes solution is the best solution in a least squares sense. But the other methods should be more robust and resilient to errors, outliers and deviations from the model, and still exploit the available orientation estimate to recover the relative pose more accurately than an image-only method.

The FOE estimation of section 2.2.3.3 can also be used to find the initial estimate $\mathbf{t}_0$ instead of the Procrustes routine. But $\mathbf{t}_0$ can be calculated from the FOE only up to scale (section 3.2.4). To find the correct scale, an estimate of the cross-ratio parameter μ is obtained by measuring and averaging, for all corresponding pixel pairs, the ratios of image distances to the FOE as in equation (3.13). Then, given the estimate of the height of the first view h, t_z is calculated as $t_z = (\mu - 1)h$, and the horizontal components are calculated by equation (3.14).

3.2.3 The homology model for a horizontal plane

Suppose again a sequence of aerial images of a horizontal ground area, and that these images are reprojected on the virtual horizontal plane as presented in section 3.2.1. This section shows another solution for the relative pose determination problem. As the virtual image planes and the ground plane are both horizontal, the camera height above the plane is equal to the plane depth in the virtual camera frame.

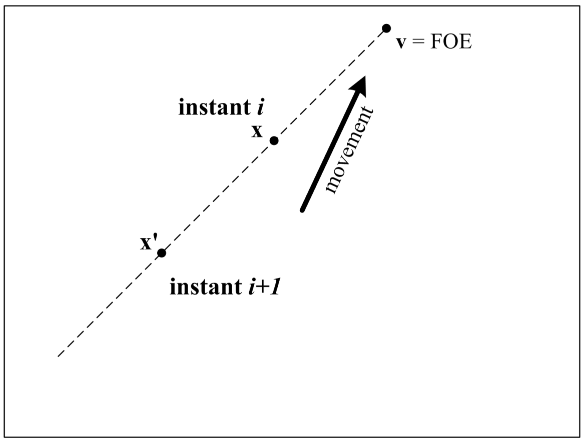
Although the homology (equation (2.7)) considers an inducing 3D plane in general position, in our experiments the relative pose estimation becomes less accurate if the plane axis also must be estimated, therefore it was necessary to restrict the inducing plane to be parallel to the virtual image planes. If a 3D plane is imaged by two cameras with image planes parallel to it, the image of vanishing line of the 3D plane, i.e., the homology axis, is the infinite line $\mathbf{a} = (0, 0, 1)^T$, and equation (2.7) becomes:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & (\mu - 1) \cdot v_x \\ 0 & 1 & (\mu - 1) \cdot v_y \\ 0 & 0 & \mu \end{bmatrix} \tag{3.6}$$

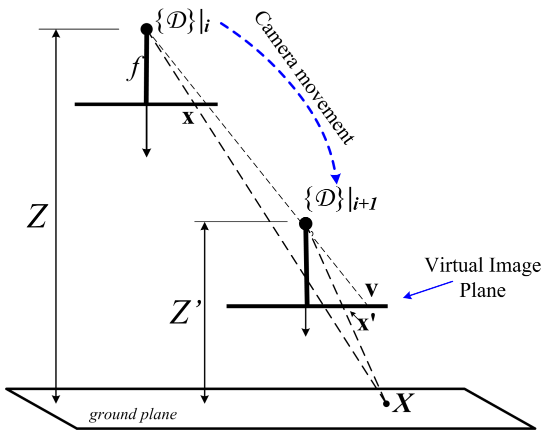
where v_x, v_y are the *inhomogeneous*[1] image coordinates of the vertex $\mathbf{v} = (v_x, v_y, 1)$. The cross ratio parameter μ depends only of the depths of the 3D plane in the two views. To analyze the latter relation, we recall that the relative scene depth of two points equals the reciprocal ratio of the image plane distances to the vanishing point of their connecting line [Arnspang et al., 1999], as reviewed in section 2.2.3.2. Here, instead of having two parallel cameras as in section 2.2.3.2, the same principles are applied to two images taken by a single moving camera, which keeps the same orientation during its movement.

Take two images of the same 3D point $\boldsymbol{X}$ taken under pure translation, as in figure 3.6(a), where the vanishing point is the FOE and the image is the superimposed image

[1]Without loss of generality as if $\mathbf{v}$ is known in homogeneous coordinates, it can be scaled into its inhomogeneous form.



(a) A superimposed image with the FOE and the image projections.



(b) 3D diagram with virtual cameras and image projections.

Figure 3.6: A pair of virtual cameras under pure translation imaging the same 3D point.

of two views. Defining Z and Z' as the depth of $\boldsymbol{X}$ in the first and second views, and $\mathbf{x}$ and $\mathbf{x}'$ as the image coordinates of their respective projections, as in figure 3.6(b), we have:

$$\frac{Z'}{Z} = \frac{dist(\mathbf{x}, \mathbf{v})}{dist(\mathbf{x}', \mathbf{v})} \tag{3.7}$$

where $dist$ means Euclidean distance in the superimposed image. Therefore the relative depth of the same point in two views is calculated from image measurements.

The relation between scene depths and image distances is valid for every single point, and it only requires an image of the same point in two views, and the FOE. But, if a 3D plane is parallel to the image planes, all points in the plane have the same depth, and are transferred between the two views by the same homology.

The homology calculation involves many pairs of corresponding pixels, and is therefore potentially more stable than an image measurement involving just one pair. Returning to the relation between the homology parameter μ and the depth of the 3D plane, applying equation (3.6) allow us to find:

$$\mathbf{x}' \;=\; \mathbf{G}\mathbf{x} = \begin{bmatrix} \frac{x_x}{\mu} + v_x - \frac{v_x}{\mu} \\ \frac{x_y}{\mu} + v_y - \frac{v_y}{\mu} \\ 1 \end{bmatrix} \tag{3.8}$$

where $\mathbf{x} = (x_x, x_y, 1)^T$. Now by calculating $|\mathbf{x} - \mathbf{x}'| = |\mathbf{x} - \mathbf{G}\mathbf{x}|$, we relate this difference with $|\mathbf{x} - \mathbf{v}|$, in image coordinates:

$$|\mathbf{x}-\mathbf{x}'| = \left[\left(x_x - \left(\frac{x_x}{\mu} + v_x - \frac{v_x}{\mu}\right)\right)^2 + \left(x_y - \left(\frac{x_y}{\mu} + v_y - \frac{v_y}{\mu}\right)\right)^2\right]^{1/2} \tag{3.9}$$

$$= \left[\left((x_x - v_x)\left(1-\frac{1}{\mu}\right)\right)^2 + \left((x_y - v_y)\left(1-\frac{1}{\mu}\right)\right)^2\right]^{1/2} \tag{3.10}$$

$$= \left[\left(((x_x - v_x))^2 + ((x_y - v_y))^2\right)\left(1-\frac{1}{\mu}\right)^2\right]^{1/2} \tag{3.11}$$

and then:

$$|\mathbf{x}-\mathbf{x}'| = |\mathbf{x}-\mathbf{v}|\left(1-\frac{1}{\mu}\right) \tag{3.12}$$

as $\mathbf{x}, \mathbf{x}', \mathbf{v}$ are collinear $|\mathbf{x}-\mathbf{x}'| + |\mathbf{v}-\mathbf{x}'| = |\mathbf{x}-\mathbf{v}|$, and therefore from 3.12 we find the image distances of equation (3.7), which is updated as:

$$\frac{Z'}{Z} = \frac{dist(\mathbf{x},\mathbf{v})}{dist(\mathbf{x}',\mathbf{v})} = \mu \tag{3.13}$$

We have therefore shown that the relative depth of the plane is equal to μ, a parameter of the homology matrix. This relation is in accordance with the known fact that given the homography matrix induced by a 3D plane in two views, the relative distance between the camera centers and the plane is equal to the determinant of the homography [Malis et al., 1999, Malis, 1998].

This is valid for general homographies (correctly scaled), thus also for planar homologies. From equation (3.6), we note that $det(\mathbf{G}) = \mu$, and as the distance between the camera center and the plane is the depth of the plane, equation (3.13) is again demonstrated in another way.

3.2.3.1 Calculating Relative Depth For Horizontal Planes

This section describes the process to calculate the depth ratio μ of a 3D plane imaged by two cameras with image planes parallel to it. This process exploits the results of section 3.2.3, especially equation (3.13), in a practical implementation. The subsequent calculation of the translation vector given the FOE and μ is dealt with in section 3.2.4.

Firstly, the images are reprojected on the virtual horizontal plane and pixel correspondences are established. Then an initial FOE estimate $\mathbf{v}_0$ is obtained from the pixel correspondences using a robust linear estimation with RANSAC [Fischler and Bolles, 1981] followed by an optimization step [Chen et al., 2003], as presented in section 2.2.3.3. The FOE estimation also excludes outliers from the set of corresponding pixel pairs.

From the pixel correspondences and the FOE estimate, an initial estimate μ_0 of the cross-ratio parameter μ is obtained by measuring and averaging for all pairs of corresponding pixel the ratios of image distances to the FOE as in equation (3.13).

Given the initial estimates $\mathbf{v}_0$ and μ_0, an optimization routine minimizes the projection error of the pixel correspondences when projected by the homology $\mathbf{G}(\mathbf{v}, \mu, \mathbf{a} = [0, 0, 1]^T)$, finding improved estimates for $\mathbf{v}$ and μ. The optimization is performed by the Levenberg-Marquardt algorithm, using the same implementation used to estimate homographies [Capel et al., 2006], but parameterized by the homology parameters $\mathbf{v}$ and $\mathbf{G}$. As it is advisable to overparameterise the optimization [Hartley and Zisserman, 2000], $\mathbf{v}$ is considered as a 3-element homogeneous vector and equation (2.7) is used instead of equation (3.6). The point $\mathbf{v}$ is normalized once its final value is known. The relative depth is the determinant of $\mathbf{G}$, i.e., μ. Two error metric were tested: Sampson distance, which is commonly used to estimate full homographies [Hartley and Zisserman, 2000], and Euclidean distance. For the latter case, the partial derivatives necessary to calculate the Jacobian of the optimization function are shown in Appendix B.3.

Figure 3.7 summarizes this process. Notice that there is no need to reproject the whole image onto the virtual plane, but rather only the coordinates of the pixel correspondences, if interest point matching with the original images achieves good enough performance. Sensor data could provide an initial FOE estimate directly, and the initial μ_0 estimate is trivial. The final optimization takes roughly as much time as the optimization necessary to calculate a homography when using Sampson distance, and with Euclidean distance it is faster. Therefore, this process can be potentially fast enough for robotic applications.

3.2.4 From the FOE and μ to the translation vector

With the homology model, the FOE and μ are directly estimated (section 3.2.3.1). Additionally, the FOE estimation described in section (2.2.3.3) may be used instead of a Procrustes routine to remove outliers and to find an initial estimate for the translation vector, before the optimization step described by equation (3.5) in section 3.2.2. In both cases, the translation vector must be calculated from the FOE, and this is described in this section.

As the rotation is compensated, the virtual cameras relative pose may be represented by a translation vector $\mathbf{t}$. The relative height corresponds to the z component of $\mathbf{t}$, although its scale depends on the height of the first camera. The FOE, already calculated in the process above, is the direction of the other two components of $\mathbf{t}$, although the FOE does not indicate the scale of $\mathbf{t}$. From images alone it is not possible to find the translation magnitude, thus this limitation is expected. Nevertheless,

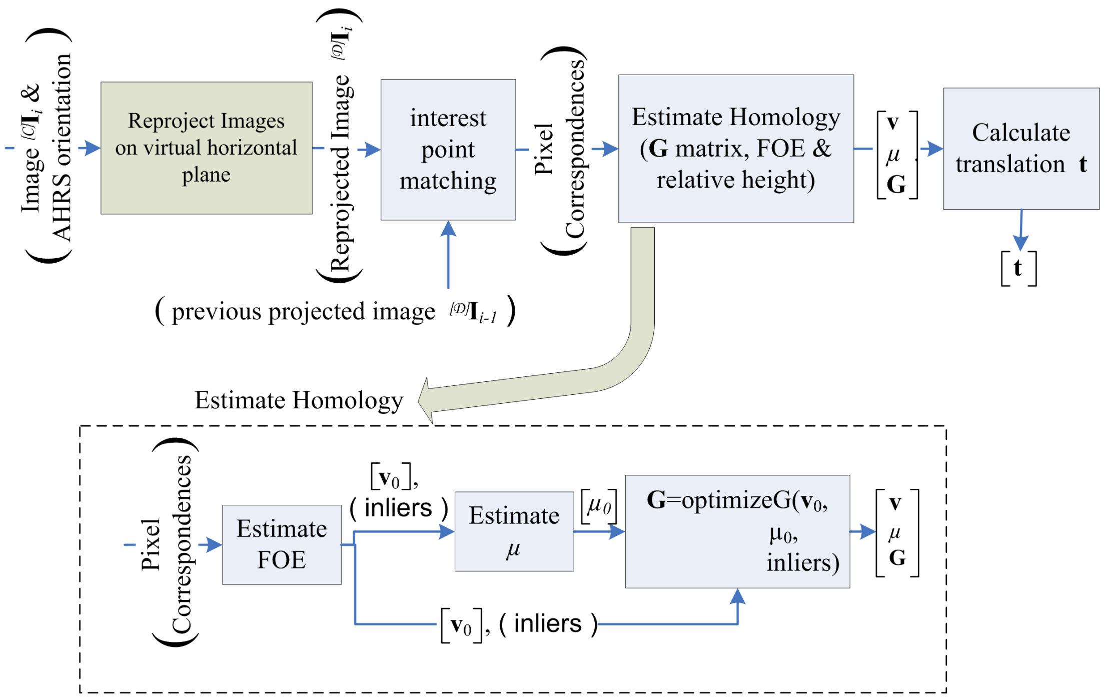


Image ${}^{\{C\}}\mathbf{I}_i$ & AHRS orientation
Reproject Images on virtual horizontal plane
Reprojected Image ${}^{\{D\}}\mathbf{I}_i$
interest point matching
(previous projected image ${}^{\{D\}}\mathbf{I}_{i-1}$)
Pixel Correspondences
Estimate Homology (G matrix, FOE & relative height)
v μ G
Calculate translation t
t
Estimate Homology
Pixel Correspondences
Estimate FOE
[v0], (inliers)
Estimate μ
μ0
G=optimizeG(v0, μ0, inliers)
v μ G
[v0], (inliers)

Figure 3.7: Finding the homology transformation between two images reprojected on the stabilized plane.

the scale of the horizontal x and y components can be calculated as a function of the vertical component.

Given the FOE $\mathbf{v} = (v_x, v_y, 1)^T$, the camera intrinsic parameters focal length f and pixel size dpx and dpy in the x and y directions, and the nadir point $\mathbf{n} = (n_x, n_y, 1)^T$, which is the image of the direction of gravity and the principal point of the $\{\mathcal{D}\}|_i$ camera, the vector $\mathbf{t} = (t_x, t_y, t_z)^T$ is calculated as a function of its vertical component t_z, as:

$$\mathbf{t} = \begin{bmatrix} \frac{(v_x - n_x)\cdot dpx \cdot t_z}{f} \\ \frac{(v_y - n_y)\cdot dpy \cdot t_z}{f} \\ t_z \end{bmatrix} \tag{3.14}$$

This relation is derived from the similar triangles shown in figure 3.8 for the x component, and a similar relation exists for the y component. The figure omits the change of coordinates (n_x) and units (dpx) that must be applied to v_x.

To find the correct scale, given the height of the first view h, t_z is calculated as $t_z = (\mu - 1)h$. Therefore the trajectory of a mobile observer may be reconstructed by summing the relative pose vectors over an image sequence. If the homology matrix is estimated after the FOE, μ will be known. If the estimated FOE is used only to initialize the process of section 3.2.2, μ is estimated by measuring and averaging, for all corresponding pixel pairs, the ratios of image distances to the FOE as in equation (3.13).

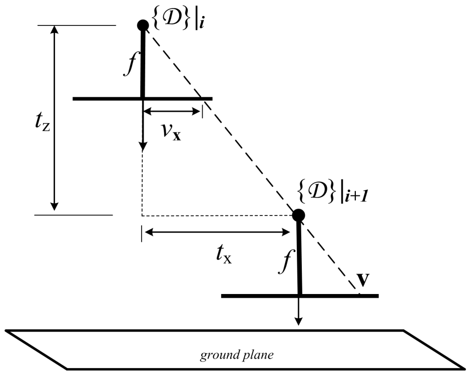


Figure 3.8: Finding the scale of translation from the difference in height.

3.2.5 Filtering the translation measurements

The visual odometry yields a sequence of translation vectors $\mathbf{t}$, which is filtered with a Kalman Filter, with the acceleration modeled as a Wiener process [Bar-Shalom et al., 2001], as detailed in B.5. This filtering should reduce the influence of spurious measurements and generate a smoother trajectory. The filter state contains the camera position, velocity and acceleration. The process error considers a maximum acceleration increment of 0.35 $[{}^{m}/_{s^2}]$, and the sequence of translation vectors is considered as a measurement of the airship velocity, adjusted by the image sampling period (0.5$[s]$ or 0.2$[s]$). The measurement error is considered as a zero mean Gaussian variable with standard deviation of 4 $[{}^{m}/_{s}]$ in the horizontal axes and 1 $[{}^{m}/_{s}]$ in the vertical axis. The sequence of 3D camera poses is taken from the filter state after the filtering.

3.2.6 Fusing visual odometry and GPS

The Kalman Filter described in section 3.2.5 has the 3D camera position in its state vector. GPS supplies a measurement of the camera position, with expected errors in the horizontal and vertical axes. Then the same Kalman Filter with a Wiener process acceleration model fuses the GPS position fixes (considered as measurements of the camera position), and from the visual odometry (the translation vectors are considered as measurements of velocity, as before). It is expected that the incorporation of GPS measurements avoids drifting of the estimated vehicle pose, while the visual odometry corrects the trajectory locally, particularly in the vertical axis.

3.3 The case of planes in general position

3.3.1 The homology model for planes in general position

This section relates Arnspang's relative depth theorem (section 2.2.3.2) with a planar homology induced by a 3D plane in general position, i.e., the plane does not need to be parallel to the camera image planes. In this case, the depth of the 3D points belonging to the plane is not always constant, so their relative depth will also not be a constant as in equation (3.13), but it must vary with the image coordinates of the pixel correspondences, or equivalently, with the corresponding 3D point coordinates on the 3D plane.

Define a translation matrix $\mathbf{T}$ that brings the FOE (i.e., the vanishing point) to the origin of the coordinate system:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & -v_x \\ 0 & 1 & -v_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.15}$$

This translation is applied to the image correspondences and to the original homology $\mathbf{G}$, yielding a transformation on the translated coordinates: ${}^{tr}\mathbf{x}' =^{tr} \mathbf{G} \cdot {}^{tr}\mathbf{x}$, where ${}^{tr}\mathbf{x} = \mathbf{T}\mathbf{x}$, ${}^{tr}\mathbf{x}' = \mathbf{T}\mathbf{x}'$, and ${}^{tr}\mathbf{G} = \mathbf{T}\mathbf{G}\mathbf{T}^{-1}$. The homology axis is also translated as ${}^{tr}\mathbf{a} = \mathbf{T}^{-T}\mathbf{a} = (a_x, a_y, \mathbf{v} \cdot \mathbf{a})^T$, if the original vertex coordinates are given in non-homogeneous form. As the vertex for the translated transformation is in the origin ${}^{tr}\mathbf{v} = (0, 0, 1)^T$, substituting in equation (2.7) yields the form:

$${}^{tr}\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{(\mu-1)a_x}{\mathbf{v}\cdot\mathbf{a}} & \frac{(\mu-1)a_y}{\mathbf{v}\cdot\mathbf{a}} & \mu \end{bmatrix} \tag{3.16}$$

where $\mathbf{v}\cdot\mathbf{a}$ is the scalar product of the *original, non-translated*, vertex and axis vectors.

Note that the μ parameter is the same for ${}^{tr}\mathbf{G}$ and $\mathbf{G}$, and $det({}^{tr}\mathbf{G}) = det(\mathbf{G}) = \mu$. This can be understood intuitively as a translation should not change the cross-ratios of a transformation.

As done in section 3.2.3, we will calculate the relation between the differences $|{}^{tr}\mathbf{x}' - {}^{tr}\mathbf{v}|$ and $|{}^{tr}\mathbf{x} - {}^{tr}\mathbf{v}|$, expressed in image coordinates. Initially we follow the steps shown in [Liang et al., 2004]:

$${}^{tr}\mathbf{x}' = {}^{tr}\mathbf{G} \cdot {}^{tr}\mathbf{x} = \tag{3.17}$$

$$= \begin{bmatrix} {}^{tr}x_x \\ {}^{tr}x_y \\ \frac{(\mu-1)a_x{}^{tr}x_x}{\mathbf{v}\cdot\mathbf{a}} + \frac{(\mu-1)a_y{}^{tr}x_y}{\mathbf{v}\cdot\mathbf{a}} + \mu \end{bmatrix} = \begin{bmatrix} {}^{tr}x_x \\ {}^{tr}x_y \\ \frac{(\mu-1)(a_x{}^{tr}x_x + a_y{}^{tr}x_y)}{\mathbf{v}\cdot\mathbf{a}} + \mu \end{bmatrix} \tag{3.18}$$

, and separating this equation into two equations for the x and y components:

$$ {}^{tr}x'_x \left(\frac{(\mu-1)(a_x {}^{tr}x_x + a_y {}^{tr}x_y)}{\mathbf{v}\cdot\mathbf{a}} + \mu \right) = {}^{tr}x_x \tag{3.19} $$

$$ {}^{tr}x'_y \left(\frac{(\mu-1)(a_x {}^{tr}x_x + a_y {}^{tr}x_y)}{\mathbf{v}\cdot\mathbf{a}} + \mu \right) = {}^{tr}x_y \tag{3.20} $$

and, supposing ${}^{tr}\mathbf{x}$ given in homogeneous coordinates, squaring and summing 3.19 and 3.20 yields:

$$ ({}^{tr}x'^2_x + {}^{tr}x'^2_y)^2 \left(\frac{(\mu-1)(a_x {}^{tr}x_x + a_y {}^{tr}x_y)}{\mathbf{v}\cdot\mathbf{a}} + \mu \right)^2 = ({}^{tr}x^2_x + {}^{tr}x^2_y)^2 $$

remembering that ${}^{tr}\mathbf{v} = \mathbf{0}$, we have:

$$ \left(\frac{(\mu-1)(a_x {}^{tr}x_x + a_y {}^{tr}x_y)}{\mathbf{v}\cdot\mathbf{a}} + \mu \right) \left| {}^{tr}\mathbf{x}' - {}^{tr}\mathbf{v} \right| = \left| {}^{tr}\mathbf{x} - {}^{tr}\mathbf{v} \right| \tag{3.21} $$

Then, substitute ${}^{tr}x_x$, ${}^{tr}x_y$ by their non-translated versions, and noticing that the scalar product is $\mathbf{v}\cdot\mathbf{a} = v_x a_x + v_y a_y + 1$, if both vertex and axis are in non-homogeneous form, we have:

$$ \begin{aligned} \frac{|{}^{tr}\mathbf{x} - {}^{tr}\mathbf{v}|}{|{}^{tr}\mathbf{x}' - {}^{tr}\mathbf{v}|} &= \\ \left(\frac{(\mu-1)(a_x(x_x - v_x) + a_y(x_y - v_y))}{\mathbf{v}\cdot\mathbf{a}} + \mu \right) &= \\ \frac{(\mu-1)(a_x x_x + a_y x_y) + (\mu-1)(-a_x v_x - a_y v_y) + \mathbf{v}\cdot\mathbf{a}\cdot\mu}{\mathbf{v}\cdot\mathbf{a}} &= \\ \frac{(\mu-1)(a_x x_x + a_y x_y) - \mu a_x v_x - \mu a_y v_y + a_x v_x + a_y v_y + \mu a_x v_x + \mu a_y v_y + \mu}{\mathbf{v}\cdot\mathbf{a}} &= \\ \frac{(\mu-1)(a_x x_x + a_y x_y) + a_x v_x + a_y v_y + 1 - 1 + \mu}{\mathbf{v}\cdot\mathbf{a}} &= \\ \frac{(\mu-1)(a_x x_x + a_y x_y) + (\mu-1)}{\mathbf{v}\cdot\mathbf{a}} + 1 &= \\ \frac{(\mu-1)(a_x x_x + a_y x_y + 1)}{\mathbf{v}\cdot\mathbf{a}} + 1 &= \\ \frac{(\mu-1)(\mathbf{a}\cdot\mathbf{x})}{\mathbf{v}\cdot\mathbf{a}} + 1 &= \end{aligned} $$

$$ \frac{(\mu-1)(\mathbf{a}\cdot\mathbf{x})}{\mathbf{v}\cdot\mathbf{a}} + 1 = \frac{|{}^{tr}\mathbf{x} - {}^{tr}\mathbf{v}|}{|{}^{tr}\mathbf{x}' - {}^{tr}\mathbf{v}|} = \frac{Z}{Z'} \tag{3.22} $$

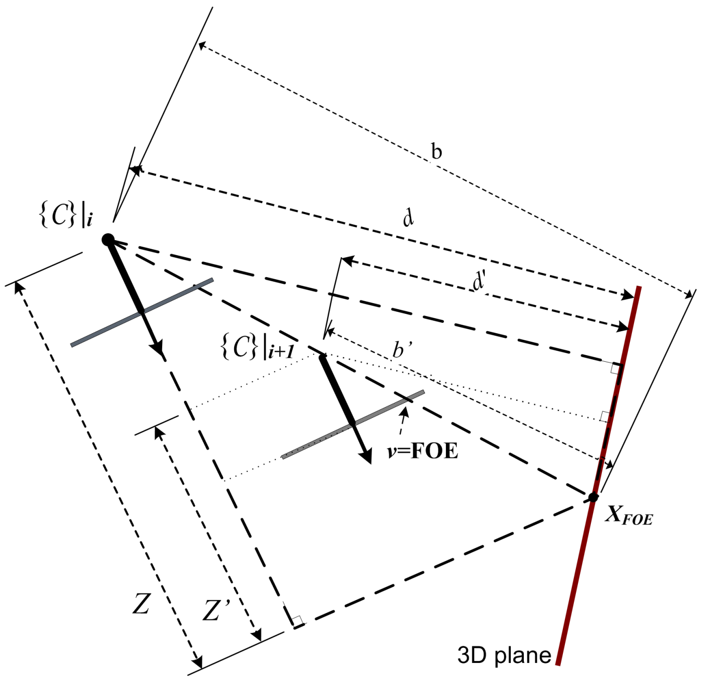


Figure 3.9: The ratios Z/Z', d/d', b/b' are the same.

Note that if $\mathbf{x} = \mathbf{v} = (v_x, v_y, 1)^T$, then the term dependent on $\mathbf{x}$ disappears and equation (3.22) is reduced to equation (3.13). Therefore μ is the relative depth of the plane in the direction of the FOE.

The relation between the relative depth and the determinant of ${}^{tr}\mathbf{G}$ yields:

$$det(\mathbf{G}) = det({}^{tr}\mathbf{G}) = \mu = \frac{Z}{Z'} \tag{3.23}$$

It is stated here that the determinant of $\mathbf{G}$ is the relative depth of the plane in the direction of the FOE, while the original theorem stated that the determinant is the ratio of the distances between the camera centers and the plane. These two ratios are indeed the same: figure 3.9 shows that they are the projections of another ratio in different directions. This other ratio is the ratio of the distances between the camera centers and the plane along the translation direction. In figure 3.9, b is the hypotenuse for the two dashed triangles, relating the ratio b/b' with the ratios d/d' and Z/Z', where d and d' are the distances between the camera centers and the plane along lines perpendicular to the plane (as in the determinant theorem), $B = b - b'$ is the baseline, so b and b' are the distances between the camera centers and the plane along the direction of translation. Z/Z' are the depths in the direction of the FOE.

3.3.1.1 Calculating Relative Depth for planes in general position

As section 3.2.3.1, this section describes a process to calculate the ratio of the depths of a 3D plane in two views related by a pure translation. But now the plane may be in a general position, not parallel to the image planes, and the relevant result is equation (3.23) of section 3.3.1.

The FOE $\mathbf{v}$ and the cross-ratio μ must still be estimated, but now the axis $\mathbf{a}$ is also unknown, and is estimated first. An usual homography H, calculated from pixel correspondences, allow us to estimate the axis $\mathbf{a}$ by solving the following linear system [Hartley and Zisserman, 2000]:

$$\lambda \mathbf{H} = \mathbf{I}_{3\times 3} + \mathbf{t}\mathbf{n}^T \tag{3.24}$$

, where $\mathbf{H}$ is the homography matrix, $\mathbf{t} = [v_x, v_y, focal\, length]^T$ is the translation between both cameras (defined up to scale), $\mathbf{n}$ is the vector normal to the 3D plane, the unknown to be determined in the linear system, and the scale λ is found as in section 2.2.2. Once the normal vector is calculated, it is related with the axis $\mathbf{a}$, which is the image of the vanishing line of the 3D plane, by the following formula [Hartley and Zisserman, 2000]:

$$\mathbf{a} = \mathbf{K}^{-T} \cdot \mathbf{n} \tag{3.25}$$

where $\mathbf{K}$ is the camera intrinsic calibration matrix.

As the calculation of the homography $\mathbf{H}$ already excludes outliers from the pixel correspondences, the first estimate of the FOE is given by a simpler linear estimation with the inliers, which is the first step of the process described in [Chen et al., 2003], without need of RANSAC or optimization.

Then an optimization is performed as in the previous section, but over $\mathbf{v}$, μ, and also $\mathbf{a}$, yielding an estimated $\mathbf{G}(\mathbf{v}, \mu, \mathbf{a})$. Note that, as discussed in section 3.3.1, μ is the determinant of $\mathbf{G}$ and the relative depth in the direction of the FOE as well the relative distance to the 3D plane.

Figure 3.10 summarizes this process, showing additional steps in relation to the algorithm in section 3.2.3.1.

Therefore, for 3D planes in unknown position, the additional cost after the homography estimation consists in solving two linear system, (to calculate the FOE and the axis) and estimate $\mathbf{G}$ as in the last section but having one more tree dimensional parameter to estimate. The determinant of $\mathbf{H}$ is also the relative depth as well as the determinant of $\mathbf{G}$, and the question is if the latter is a better approximation to the truth, in order to justify its additional cost.

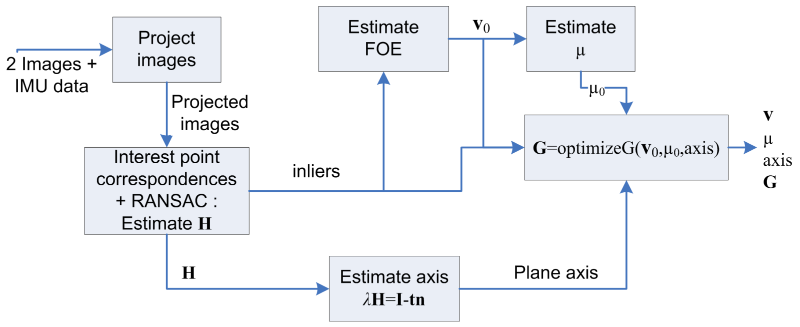


Figure 3.10: Finding the homology transformation between two images.

3.4 Trajectory recovery results

3.4.1 Relative height experiments with ground truth

On these experiments, the camera inertial system of figure 1.2 was mounted in a tripod or a fixed base, as shown in figures 3.11 and 3.13, taking images of objects on the ground floor. The camera height was manually set, with the purpose of providing experiments with measurable ground truth. These experiments used the MTB-9 AHRS.

3.4.1.1 Tripod experiment with 3D plane parallel to image plane

In this experiment, the rigidly coupled AHRS-camera system of figure 1.2 was mounted on a tripod as shown in figure 3.11 and 50 images of the ground floor were taken from different viewpoints at three different heights. The tripod was moved manually but kept still while each image was being taken. Image examples are shown in figure 3.12and 3.11. The objective is to calculate the height of the camera for each view as a ratio of the height of the first view, and to compare the results obtained by the homography and the pure translation models models against hand-measured ground truth.

Figure 3.12 shows the relative height for all 50 images. Two arrows connect two highlighted points to their respective images. The tripod was set to 3 different heights, therefore the 3 horizontal lines are the ground truth. The circles are the relative heights found by the Procrustes routine, with no need of any optimization. The stars are the result of the process described in section 3.2.3.1. The crosses are the relative depths taken as the determinant of a homography **H**, estimated with RANSAC, optimized to minimize the projection error on pixel correspondences, and scaled as in equation (2.4).

The results are summarized in table 3.1, where the unit is the reference height

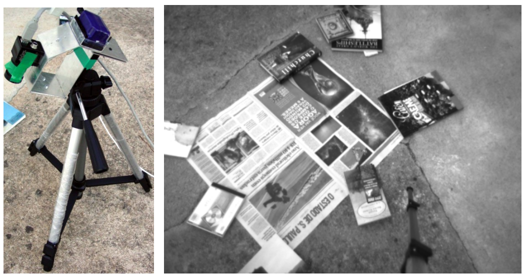


Figure 3.11: A tripod with a camera rigidly mounted with the AHRS, and an image example.

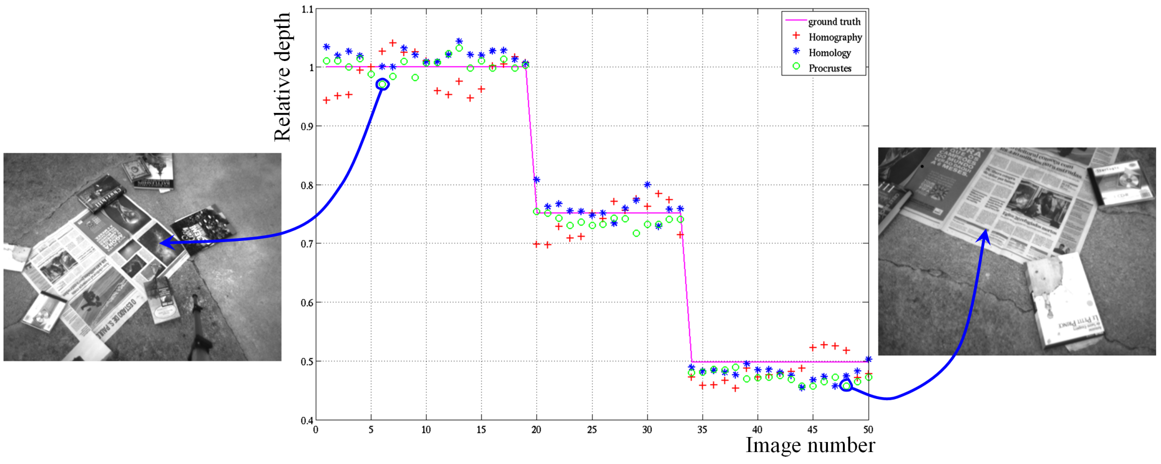


Figure 3.12: Relative heights to the ground for the tripod experiment, with two example images. Each unit on the abscissa corresponds to one image. Images were taken from 3 different heights.

	Rel. height error		**Average Time [s]**		
	RMS	**std**	**ransac**	**Optimiz.**	**Total**
homography	0.033	0.030	0.40	0.29	0.69
homology	0.020	0.020	0.57	0.19	0.76
Procrustes	0.019	0.017	0.74	-	0.74

Table 3.1: Results for relative depth of 3D plane parallel to virtual image planes.

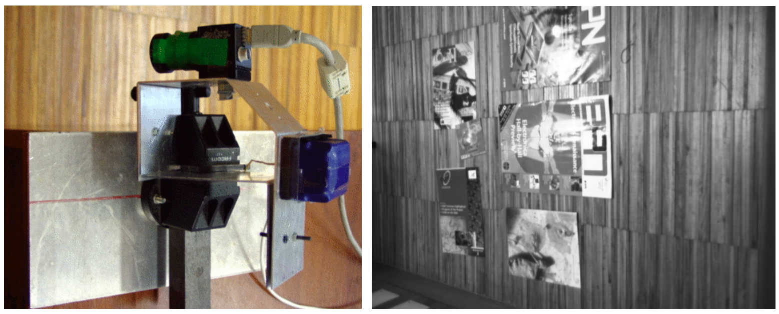


Figure 3.13: The camera mounted on a base, to be manually aligned with the walls, with an image example.

(104.5[cm]), and show that the pure translation models yield better accuracy than the homography model. The closed form Procrustes routine yielded the best results. Optimizing in **t** (equation (3.5)) did not improve, or even worsened the results. While the average computational time spent with optimization decreased with the simpler models, the time spent with outlier removal increased - and using the FOE estimation to detect outliers was faster than using the Procrustes routine.

3.4.1.2 Tripod experiment with general 3D plane

In the next experiment the monocular camera, fixed to its base in order to have always the same orientation, was manually placed at 5 different heights above the same portion of the ground plane, and a total of 25 images were taken from different positions. The images are not reprojected on the virtual horizontal plane, the original images are directly used.

The experimental setup with an image example is shown in figure 3.13. The objective, as in the first experiment of section 3.4.1.1, is to use the homology process to estimate the relative heights, comparing with the usual homography model, but now considering the ground plane in general position, and thus applying the process of section 3.3.1.1.

The inertial system was not used in this experiment. The FOE was estimated from the pixel correspondences, and from it the cross-ratio parameter μ was estimated. A

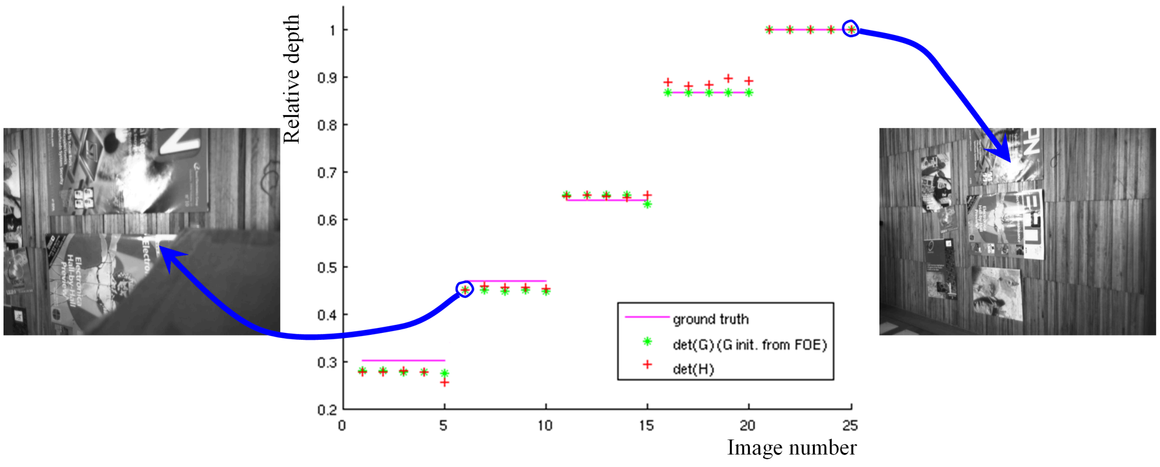


Figure 3.14: Relative heights to the ground plane, with two image examples. Each unit on the abscissa corresponds to one image. Images were taken from 5 different heights.

	RMS error	std of error
homography	0.018	0.011
homology	0.015	0.010

Table 3.2: Results for relative depth of 3D plane in general position.

homography **H** was also estimated from the pixel correspondences, and an initial estimate for the axis was obtained from **H** using equations (3.24) and (3.25). Then the homology **G** was estimated from these parameters.

The results, shown in figure 3.14 and summarized in table 3.2, indicate that the homology model slightly improved the relative depth estimates from the full homography model. The maximum height, with relative depth equal to one, was 83$[cm]$

3.4.2 Overpass Experiment

In the next experiment, the camera and MTB-9 AHRS were manually moved down an overpass access where 39 images were taken. The recovered camera trajectory is similar to the trajectory of an UAV approaching to land. Figure 3.15 shows the trajectories recovered by the homology (trajectory with blue lozenges) and homography models, superimposed onto a photograph of the overpass. Only the homographies were used to recover the trajectory indicated by the red squares. The green circles indicate a trajectory recovered by the homographies where the camera orientation was given by the AHRS to avoid drifting of the camera orientation estimate. The

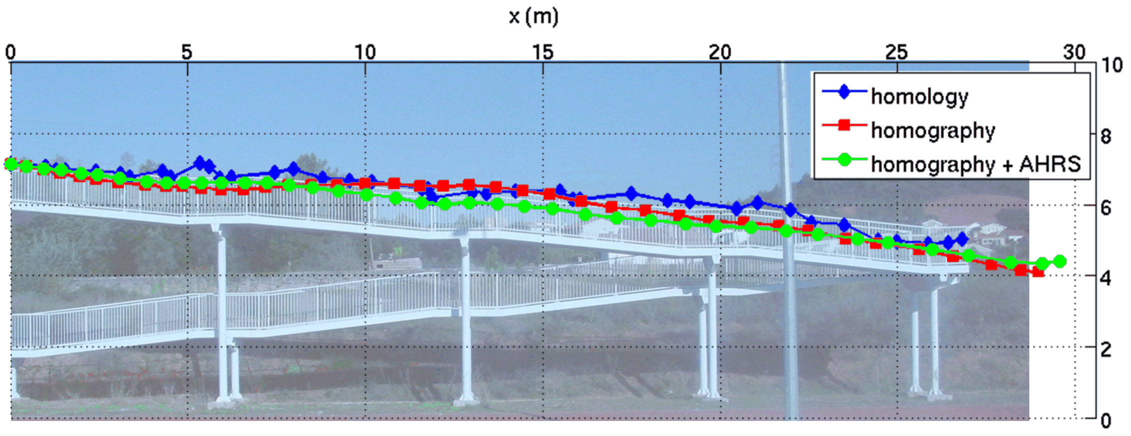


Figure 3.15: Trajectories recovered by the homology and homography models for the overpass dataset.

	Final Error [m]		Average Time [s]		
	Vertical	**3D**	**ransac**	**Opt.**	**Total**
Procrustes	0.14 (06%)	3.57 (13%)	0.09	-	0.09
Proc. + opt(t) Euclidean	0.14 (06%)	3.66 (14%)	0.10	0.04	0.14
Proc. + opt(t) Sampson	0.08 (04%)	3.71 (14%)	0.10	0.06	0.16
Homology	0.04 (02%)	3.78 (14%)	0.22	0.18	0.40
Homography	0.89 (40%)	4.17 (16%)	0.04	0.35	0.39
Homography + ahrs	0.60 (27%)	5.19 (19%)	0.04	0.34	0.38

Table 3.3: Comparison of trajectories recovered using the homography and the pure translation models for the overpass dataset.

metric scale was measured in this image from a hand-measured reference, and the trajectory estimation presupposes that the height of the first image is known. The AHRS was used to estimate only the roll and pitch angles, while the yaw angle could not be estimated due to the metallic structure of the overpass which interferes with the magnetometer readings. Therefore the camera was kept in the same orientation as far as it is possible when carrying it by hand - small errors on the orientation were still visible in the images. Table 3.3 shows the final error in the vertical component as well as final 3D pose error, indicating also the percentage relative to the total displacement. The vertical displacement was 2.18$[m]$ and the horizontal displacement 26.5$[m]$.

Figure 3.16 shows the trajectory recovered by the Procrustes procedure alone, using RANSAC to eliminate outliers, and two trajectories recovered with the optimization in $\mathbf{t}$ described by equation (3.5), initialized with the result of the Procrustes

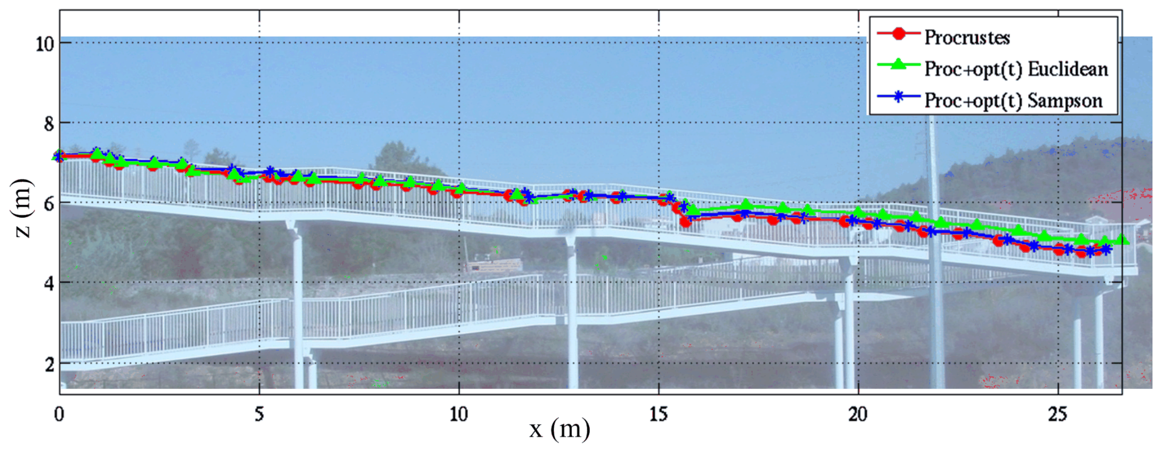


Figure 3.16: Trajectories recovered by the Procrustes procedure for the overpass dataset, and by optimizing in **t**, with two distance metrics.

procedure, and using two distance metrics. For this dataset the projective models are not necessary, as the simpler Procrustes registration finds a good, non-iterative solution, and even is able to remove outliers. The 10$[cm]$ difference between the pure translation models in the vertical displacement is not very significative.

3.4.3 Screwdriver Table Experiment

In this experiment, the camera was mounted on a moving table below the screwdriver of figure 3.17(a). The camera was fixed with its optical axis pointing approximately down - bubble levels were utilized to fix the camera enclosure at the right orientation. Therefore in this experiment reprojected images were not utilized, and the original images were always used. Although the AHRS is present, it was not utilized. The screwdriver moving table was moved by manually turning three mechanical screws, which allow independent movement for each axis. The objective of this experiment is to perform trajectory recovery with small errors in the camera orientation with motion in a single axis, vertical or horizontal. Both cases represent extreme situations: for horizontal displacements, the direction of translation is almost parallel to the image plane, and the FOE is estimated as a large number (from 10^4 to 10^6 pixels in this dataset), nevertheless the trajectory recovery proceeds normally. Large FOE coordinate values also can be found in the flight experiment of section 3.4.4, as the airship motion is often almost horizontal. On the other hand, during vertical displacement the FOE is close to the camera principal point in the center of the image.

The screwdriver table was moved and the images used to recover its trajectory as shown in figure 3.17(b). The horizontal (forward and lateral) displacements in-

Unit: $[cm]$ **ground truth**	**forward1** 15.9	**forward2** 15.9	**lateral** 8.5	**height2** 99.6	**height3** 109.6
homology	16.4	15.9	7.9	100.1	110.5
homology filtered	16.4	15.8	8.0	100.1	110.5
homography	15.4	13.6	9.1	98.8	108.1
homography scaled by homology height	15.4	13.5	9.4	99.2	108.8

Table 3.4: Comparison of displacements recovered using the homography and homology models for the screwdriver table dataset. All displacements are given in $[cm]$.

dicated in figures 3.18(a) and 3.18(b) are the maximum displacements allowed by the mechanical devices, and were measured with metric tape. After each vertical displacement of the camera, its height in relation to the ground plane was also hand measured. Therefore the distances indicated in figures 3.18(a) and 3.18(b) can be compared against ground truth as it is shown in table 3.4. One image was taken every $0.6[s]$, including the moments when the camera was not moving, e.g. when its height was being measured. The initial height was also measured and it was $85.6[cm]$. Figure 3.17(a) also presents an example image.

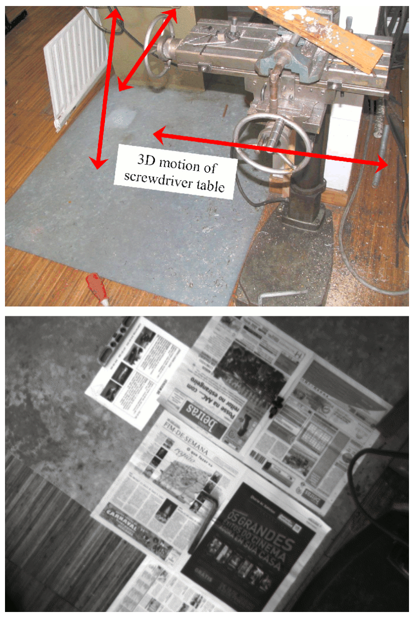
3.4.4 Trajectory recovery for aerial vehicles with the MTB-9 AHRS

This experiment uses images taken by the remotely controlled airship of figure 1.2, carrying the AHRS-camera system with the MTB-9 AHRS and a GPS receiver, flying above a planar area next to the Coimbra Aerodrome. The image frame rate is $2fps$.

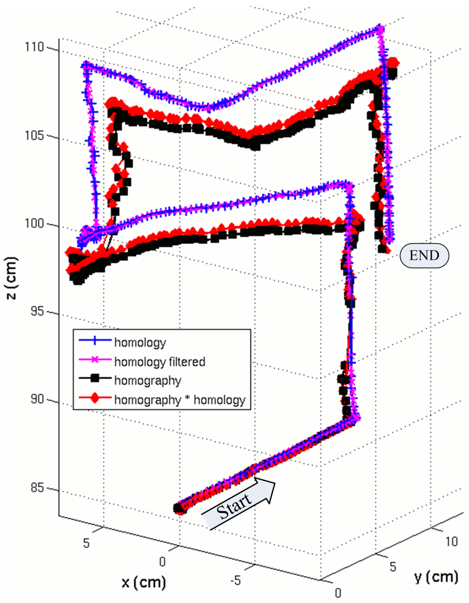
3.4.4.1 Visual Odometry: Heights For a UAV

The GPS measured height is shown in figure 3.19 compared against visual odometry based on the μ value of homologies calculated for the sequence of images using the process described in section 3.2.3.1.

For the first image the height is visually guessed and manually set as $h_1 = 4[m]$, and for the ith image the height is $h_i = \left(\prod_{j=1}^{i-1} {}^{j+1}\mu_j\right) \cdot h_1$, where ${}^{j+1}\mu_j$ represents the cross-ratio parameter for the homology that transforms the jth image into the image $j+1$. For a few image pairs the homology could not be calculated due to corrupted or missing images, or because too few interest points were matched. In this case the last valid μ value is assumed to be the current one. Except by that, no other attempt was

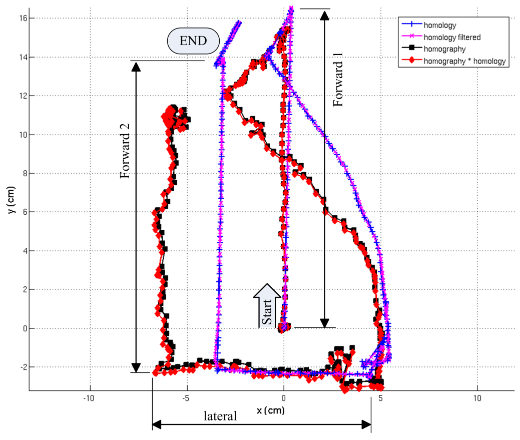


(a)

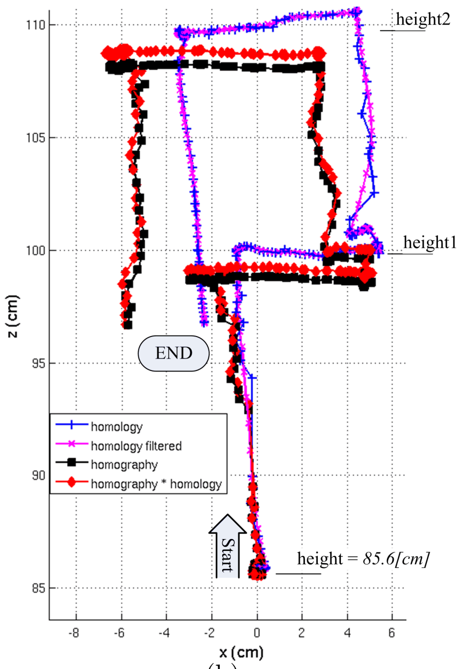


(b)

Figure 3.17: The screwdriver with its moving table and one example image (a) and the recovered 3D camera trajectories (b).



(a)



(b)

Figure 3.18: Two views of the recovered trajectories with highlighted displacements.

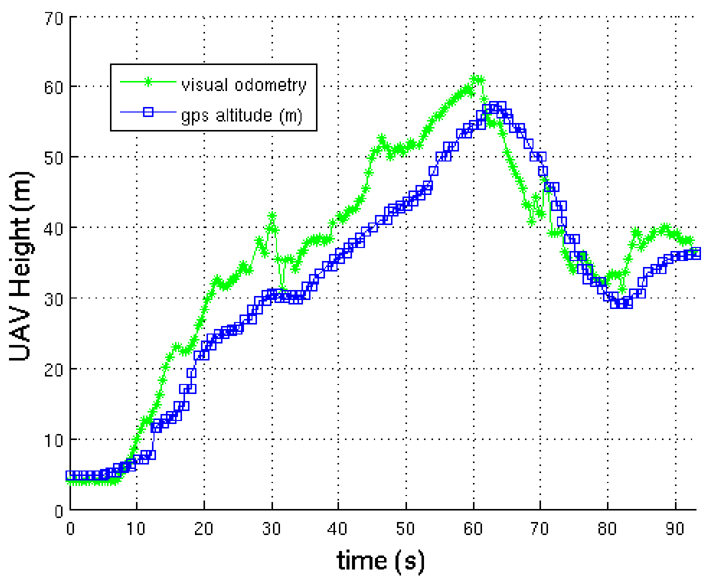


Figure 3.19: Visual odometry based on homology compared with GPS altitude measurements.

made to filter the data to avoid the drift from successive multiplications of relative heights.

The scale of the visual odometry depends on the manually set height of the first image. GPS altitude measurements are not very accurate; in this case their expected vertical error (*epv* in GPS terminology, or one standard deviation), was always around $30[m]$ - so large that it covers most of the scale, therefore it was omitted in the graph. It is evident that the visual odometry is close to the GPS data, even having peaks a few seconds ahead of it.

3.4.4.2 Visualizing the effect of orientation error with an example image

Within the projective homology model, equation (3.13) allows to calculate individual μ values for each corresponding pixel pair, which should be equal for all points in the ground plane as they represent the same relative height. Figure 3.20 shows, for one example image of the Coimbra Aerodrome, the individual μ values for each matched pixel (correspondences classified as outliers in the FOE estimation are *not* shown). On the right, the same data is shown as a 3D plot with a larger scale for ease of visualization. The color scale and the z axis both indicate the same μ values. This example has noticeable orientation error, although the translation vector still could be estimated.

In figure 3.20(a) all corresponding pixels are in a relatively narrow band close to

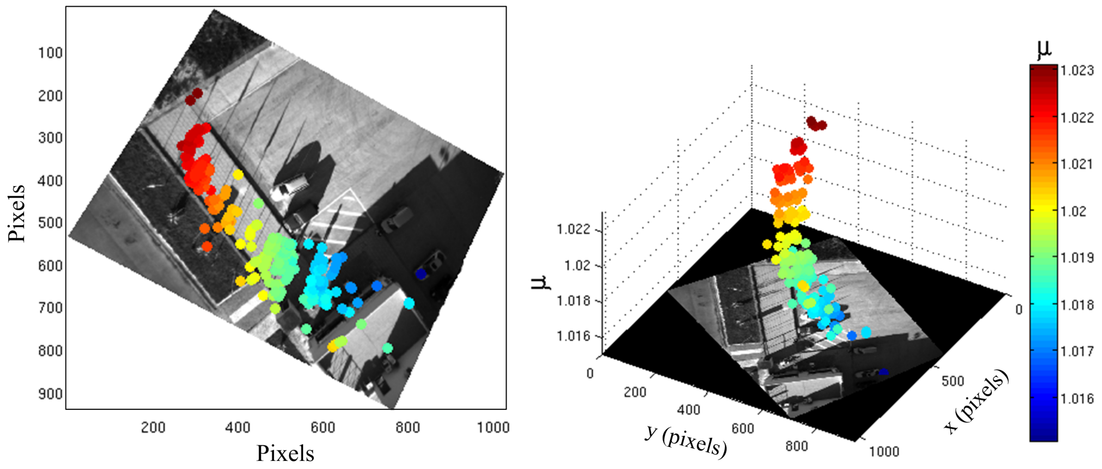


Figure 3.20: The μ values for individual pixel correspondences.

the center of the image. There were other correctly matched pixels in other sections of the image but, as the motion model does not cover the non-compensated rotation, they were classified as outliers due to errors on the orientation estimation. This effect is more significant in image areas far from the nadir point. Errors in the calibration of the rotation between the camera and inertial sensor frames should increase this effect. If the orientation error is too large, too many pixel correspondences are discarded and it is not possible to estimate the homology reliably. The AHRS orientation output standard deviation for a static setup (as on the tripod experiments) is 3°. The error is larger for the moving UAV.

Even among the inliers the measured height change varies between 2.3% and 1.6% of the first image height. The building in the bottom of the image is out of the horizontal plane and thus there are a few points that do not follow the general tendency.

3.4.4.3 Recovering the complete UAV Trajectory

As presented in section 3.2.4, the UAV trajectory is recovered by adding the translation vectors recovered for each image pair in the image sequence. The trajectory is thus reconstructed for the same UAV dataset and shown in figure 3.21. A Kalman Filter as described in (3.2.5) filtered the translation vectors for all methods, and predicted the translation for the few image pairs for which the translation estimation was not successful and a measurement was missing.

Both 2D and 3D plots of the same data are provided. The GPS trajectory is indicated by red circles, with the larger circles indicating the expected horizontal error (*eph* in GPS terminology), which was around 15[m] most of the time. The GPS indicates that the trajectory length was $660[m]$, and the average speed was $7.5[^m/_s]$.

The squares in figure 3.21(b) show the trajectory reconstructed by adding up all translation vectors, estimated by the homology model. The trajectory recovered by the homography model is also shown, and as only the ratio $^{\mathbf{t}}/_{d}$ is recovered, the recovered vector is multiplied by the current airship height found by the homology model.

Figure 3.22 show as squares the same trajectory estimated by the optimization approach of equation 3.5 initialized by the FOE estimation. The trajectory recovered by the homography model is also shown, scaled by the current recovered airship height.

The most important source of error in our experience, inclusive in this estimated UAV trajectory, is the FOE estimation, which was found to be less accurate and slower if the orientation estimate is less accurate. With an inaccurate orientation estimate, the lines connecting corresponding pixels fail to converge to a single point, and the FOE is more difficult to estimate. Therefore better orientation estimates, or another measurement of the direction of movement (which is equivalent to the FOE) to improve the initial FOE estimate, would improve the accuracy of the estimated trajectory.

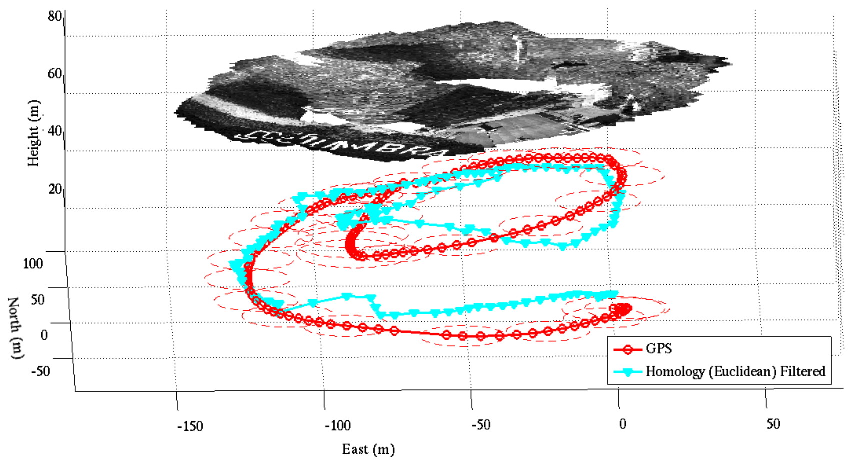
3.4.4.4 Combining GPS and Visual Odometry

The translation recovered by the visual odometry was fused with GPS position fixes in a Kalman Filter with a Wiener process acceleration model as described in section 3.2.5. The GPS measures the airship position and translation vectors from the visual odometry are interpreted as a velocity measurement between two successive poses, with a manually set covariance smaller in the vertical axis than in the horizontal ones. The fused trajectory is shown in figure 3.23(a) using the translation recovered by the homology model and in figure 3.23(b) using the translation vectors recovered the optimizing in $\mathbf{t}$ after obtaining an initial estimate from the FOE.

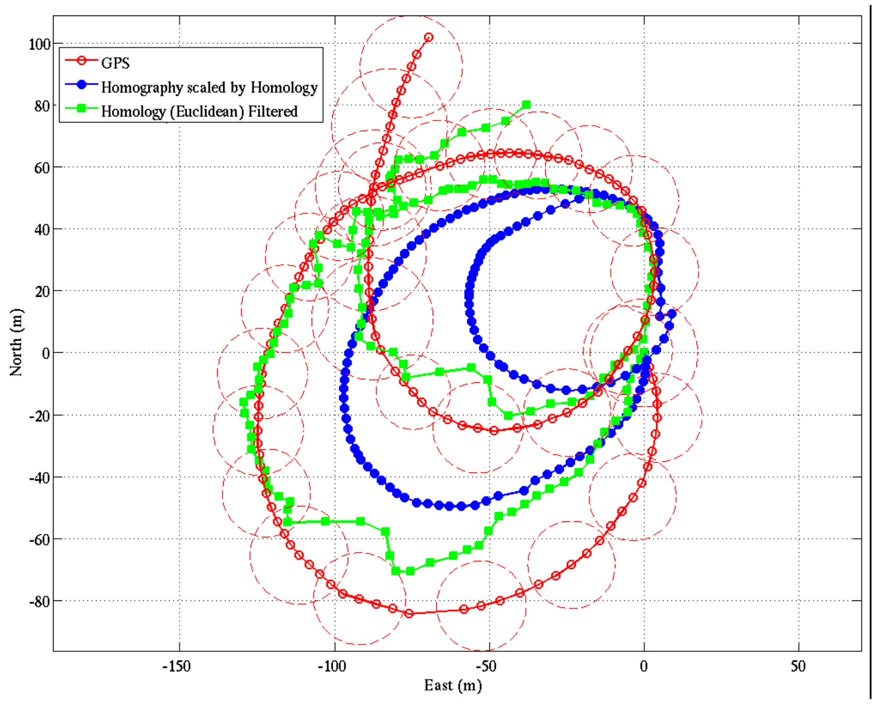
3.4.4.5 Comparison with GPS as scale to relative pose recovered from homography

The the GPS measured trajectory and the trajectory recovered from rotation-compensated images were shown. It is hard to obtain accurate ground truth to evaluate these results for a free flying aerial vehicle but this section offers another piece of evidence that the altitude recovered is accurate.

The homography model was applied on the reprojected images, also recovering the trajectory from a sequence of relative poses. There is an inherent scale ambiguity on equation (2.4), as only the ratio $^{\mathbf{t}}/_{d}$ is recovered, and not the actual translation vector $\mathbf{t}$. Therefore, to compare the trajectory recovered by the homography model



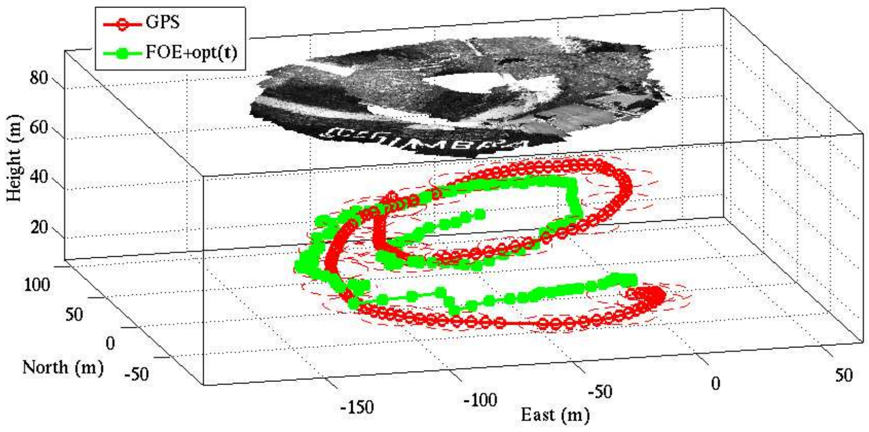
(a)



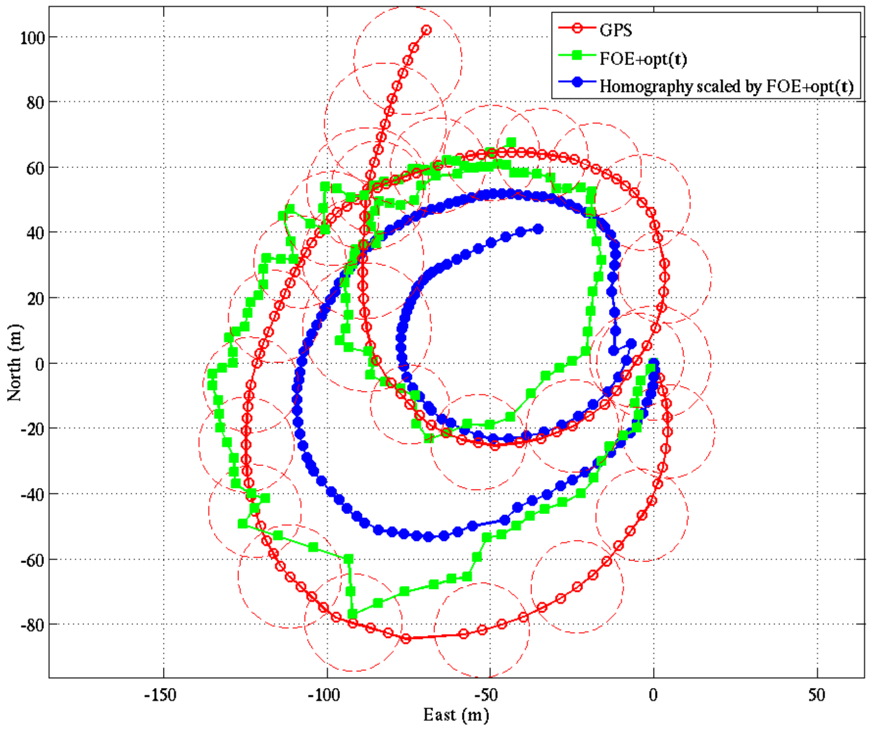
(b)

Figure 3.21: 3D (a) and 2D (b) plots comparing the trajectory recovered by the homology with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation).

with the trajectory measured by GPS, it is necessary to have an independent measure of d, to recover the scale.



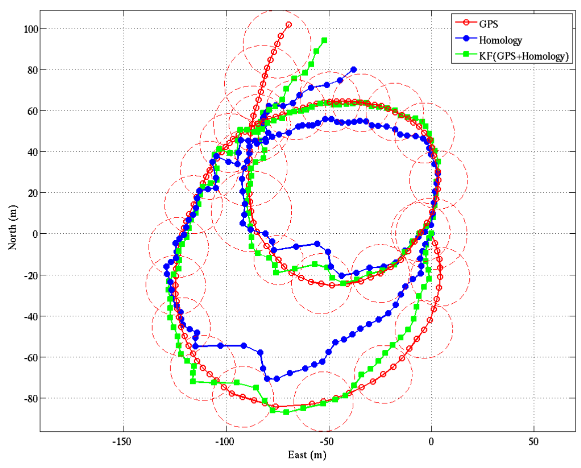
(a)



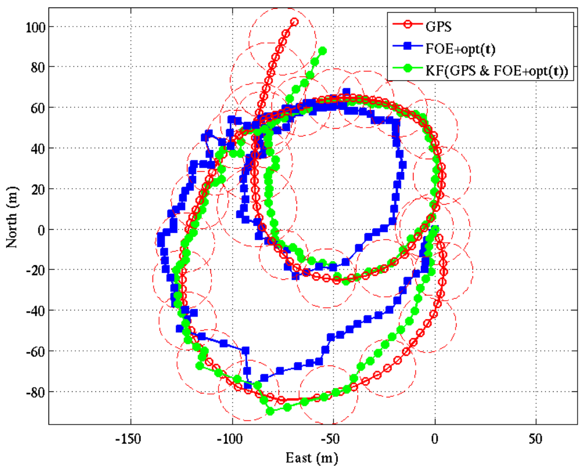
(b)

Figure 3.22: 3D (a) and 2D (b) plots comparing the trajectory recovered by optimizing in the translation vector with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation).

Firstly, the GPS is used to obtain these measurements. For each image pair, the recovered ratio $\mathbf{t}/d$ is multiplied by the current GPS measured height to obtain the



(a) With the homology model.



(b) Optimizing in **t** after FOE estimation.

Figure 3.23: Data from GPS and visual odometry fused by a Kalman filter.

actual translation **t**, and the resulting trajectory is shown in figure 3.24. The GPS height measurement is simply the current GPS altitude minus the altitude measured before taking off.

Secondly, another trajectory is generated by taking as d the height estimated by fusing GPS and visual odometry data as in section 3.4.4.4. It is also shown in figure 3.24. And finally, the height recovered by the visual odometry was used to measure d in the trajectory shown in figure 3.21(b).

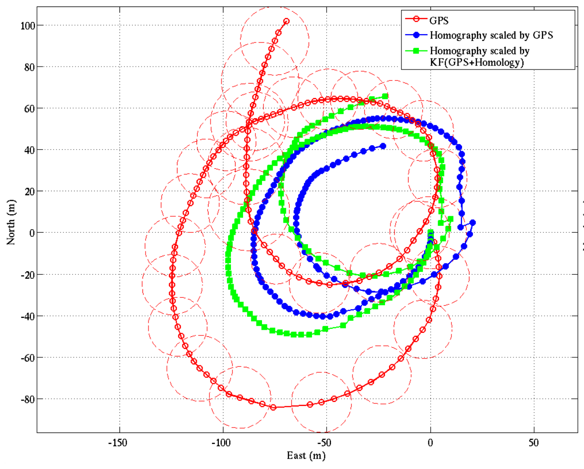
Note that for the first, larger curve, the trajectory scaled by the GPS height is too short, and the other trajectories are closer to the GPS 2D trajectory. This indicates that the height measured by the GPS was too low.

The trajectory scaled by the fused data exploits the GPS data to avoid drifting, and the visual odometry data offers some degree of improvement - with the homology model it is even able to close the second loop as in figure 3.24(a). In the second, smaller loop of figure 3.24(b), the trajectory scaled by the GPS height has a larger loop, closer to the size of the loop in the reference trajectory, indicating that height recovered by the visual odometry has already drifted .

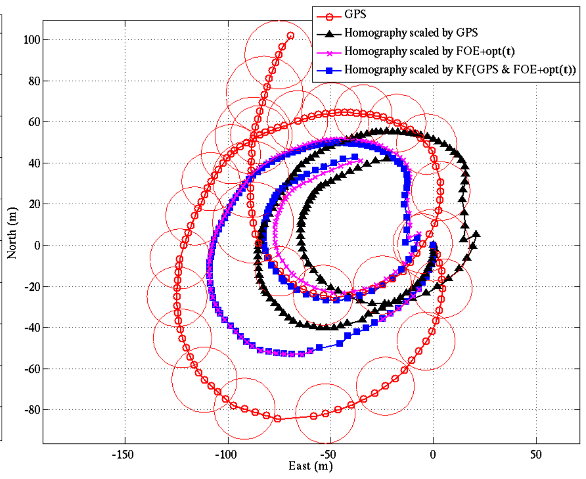
Table 3.5 also presents error values for the trajectory recovered by the homography model alone, with no extra height measurement: the height of the first image is manually set and each translation vector is scaled by the current estimated height.

3.4.4.6 Comparison of methods including the Procrustes solution.

For the flight dataset, the trajectory recovered by the Procrustes procedure is too small, even if optimization is performed. Some translation vectors are wrongly esti-



(a) Using the homology model



(b) Optmizing in **t** after initialization from the FOE.

Figure 3.24: 2D plot of the trajectories recovered by the homography model with different height measurements to scale it. Compare with figure 3.21.

mated, and this error propagates in the subsequent estimated trajectory. To address this problem, the FOE was estimated as shown in section 2.2.3.3, and an initial estimate of **t** was derived from it as shown in section 3.2.4, obtaining the results of section 3.4.4.3.

Figure 3.25 shows the following recovered trajectories: the triangles indicate the trajectory recovered by optimization in **t** initialized by a Procrustes procedure. The stars and filled circles are the result of applying the same optimization to an initial estimate generated by estimating the FOE, with respectively Sampson and Euclidean distance metrics. The trajectories recovered by the homology model (squares) and GPS position fixes (empty circles) are also shown. This thesis shows results only for the best variations of these methods, with some additional results available online in [Mirisola and Dias, 2008].

3.4.4.7 Computational requirements and error statistics

Both Procrustes and the first linear step of FOE estimation are very fast routines. But, due to the presence of outliers, they are embedded in a RANSAC procedure. For the aerial dataset, figure 3.26 shows the time spent with outlier removal, with the final optimization step, and the total of both steps. With any of the translation-only models, outlier removal is the most computationally expensive part of the process, while with the homography model the final optimization is more costly. Errors in the orientation estimate may hamper outlier removal with a pure translation model,

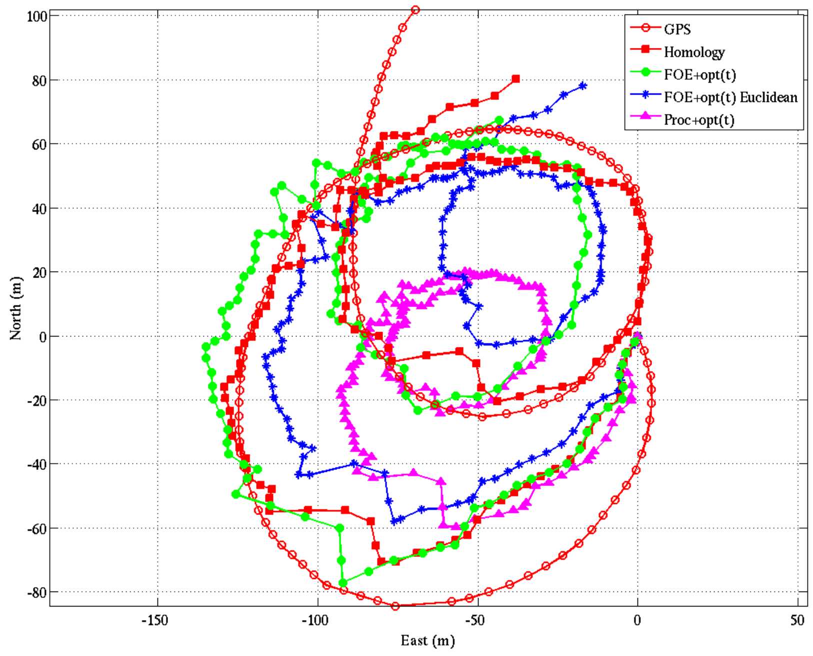


Figure 3.25: 2D plots comparing of the trajectory recovered by the homography model and scene registration approaches.

as they are not modeled. These losses are partially compensated by the faster final optimization step, as there are less variables to optimize. The data shown in figure 3.26 also include the optimization step which is part of the FOE estimation, and is applied after the outliers are removed. It takes on average $0.05[s]$, and it is a reason why outlier removal with the FOE estimation is slower than with Procrustes.

For the homology model with Sampson distance metric, the average computational time for the final optimization step was 4% larger than for the homography model, with standard deviations of 20% for homologies and 40% for homographies related to their respective average values. But, if the Euclidean distance metric is used, the average computational time is halved, with a standard deviation of 26% of the mean. The simpler distance metric also yielded slight better, or not significantly different results, therefore it is the one used in this thesis.

Table 3.5 compares numerically the error, using the GPS as a reference, of the trajectories presented so far and includes some results not shown in the figures of this thesis. The fused trajectory from the Kalman Filter of section 3.4.4.4 is the closest to the GPS, which is expected as it utilizes GPS data directly. The comparison of the methods tested show that the homology model achieved the results closest to GPS, very closely followed by the optimization with the translation vector (equation 3.5) after obtaining an initial estimate from the FOE estimation. The FOE estimation

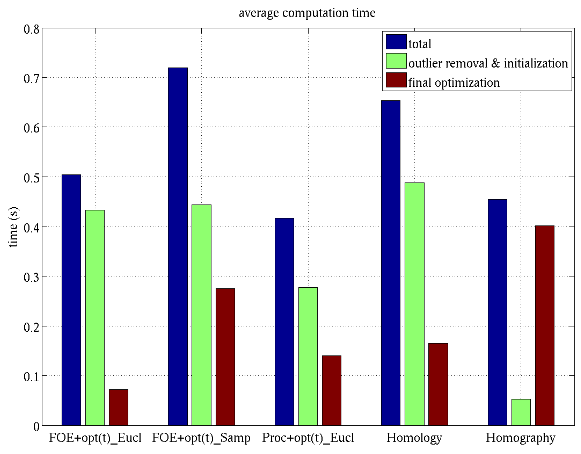


Figure 3.26: Comparison of average computation time per frame for some of methods tested. All experiments in this chapter were executed in a Pentium D 3.2 [GHz].

was better than the Procrustes routine to find an initial estimate of **t**.

Different height measurements were used to scale the translation recovered by the homography, including GPS data, the height component of the trajectory recovered by the pure translation models, and the result of the Kalman Filter fusing both visual odometry and GPS. The visual odometry achieved better results than GPS, indicating a more accurate height estimate in the fused trajectory. With the homology model, the result of the Kalman filter was better than either GPS or visual odometry alone to scale the trajectory recovered by the homography model.

Unit: meters	3D Position Error			2D Position Error			Error in the length of **t**		
	Avg.	Max.	Final	Avg.	Max.	Final	RMS	Avg.	Max.
Homography	48.5	100	100	44.3	98.6	98.6	3.00	-2.68	14.6
Homography scaled by GPS	38.5	79.3	79.3	33.0	76.2	76.2	2.07	-1.19	14.3
Homography scaled by FOE+opt(**t**)	27.7	72.4	72.4	20.0	69.7	69.7	1.88	-1.14	13.2
Homography scaled by the filtered output KF(GPS & FOE+opt(**t**))	27.9	70.2	70.2	20.1	67.0	67.0	1.84	-1.07	13.2
KF(GPS & FOE+opt(**t**))	7.5	21.6	20.0	6.0	21.5	20.0	1.62	0.1	8.06
optimization in **t** initialization: FOE	16.4	43.3	43.3	15.8	43.0	43.0	2.03	0.16	8.71
optimization in **t** initialization: Procrustes	38.9	87.0	87.0	35.1	85.7	85.7	2.56	-1.73	10.5
Homology	12.1	39.8	38.9	10.6	39.0	38.2	2.11	0.09	10.3
Homography scaled by Homology	35.2	62.5	49.5	31.5	58.2	45.8	1.67	-0.66	12.2
Homography scaled by the filtered output KF(GPS & Homology)	30.3	63.7	63.7	24.1	60.1	60.1	1.83	-1.00	13.8
KF(GPS & Homology)	7.0	20.3	19.6	5.0	19.5	19.0	1.71	-0.36	8.04

Table 3.5: Comparison of all trajectories presented with the GPS as a reference. Red and green cells represent the two worst and the two best values in each column. The grayed lines represent the results of Kalman Filters including the GPS position fixes which can not be fairly compared. All error values are given in meters.

3.4.5 Trajectory recovery for aerial vehicles with the MTi AHRS

This experiment uses images taken by the remotely controlled airship of figure 1.2, carrying the AHRS-camera system with the MTi AHRS and a GPS receiver, flying above a planar area next to the Coimbra Aerodrome. The camera captured images at 5 fps.

The letters of the city name written on the ground were visible on the first image of the sequence. The letters were measured on the image, and then the first airship height was calculated from the camera intrinsic parameters and the actual, hand-measured, size of the letters. The first height was found to be $h_1 = 25[m]$.

The recovered trajectory is shown in figure 3.27. Both 2D and 3D plots of the same data are provided. The GPS trajectory is indicated by small circles, with larger circles indicating the GPS *eph*, which was around 8[m] in the horizontal axes and 15[m] in the vertical axis. The GPS indicates a trajectory length of 543[m], and average speed of 9[$^m/_s$]. A Kalman Filter with a constant acceleration model [Bar-Shalom et al., 2001] filtered the translation vectors for all methods, and predicted the translation for the few image pairs for which the translation estimation was not successful and a measurement was missing.

The squares in figure 3.27 show the trajectory reconstructed by adding up all translation vectors, estimated by the optimization approach of equation 3.5 initialized by Procrustes. The recovered trajectory is very close to the one recovered by the Procrustes procedure alone, indicated by crosses. The trajectory recovered by the homography model using only the images is also indicated with stars. As only the ratio $^{\mathbf{t}}/_{d}$ is recovered by the homographies, the scale is recovered by multiplying the recovered vector by the current recovered airship height. Tables 3.6 and 3.7 show the errors in the visual odometry, taking the GPS as a reference, and the execution times.

Unit: meters	3D Position Error			2D Position Error			Error in the length of $\mathbf{t}$		
	Avg.	Max.	Final	Avg.	Max.	Final	RMS	Avg.	Max.
Homography	57.5	98.2	52.3	43.1	67.6	52.0	0.88	-0.63	1.24
Visual Odometry: Procrustes only	16.5	43.6	43.6	15.3	43.3	43.3	0.73	-0.46	0.59
Visual Odometry: optimization in $\mathbf{t}$ initialization: Procrustes	16.0	41.6	41.6	14.9	41.3	41.3	0.70	-0.45	0.59

Table 3.6: Comparison of visual odometry with the GPS reference. All error values are given in meters.

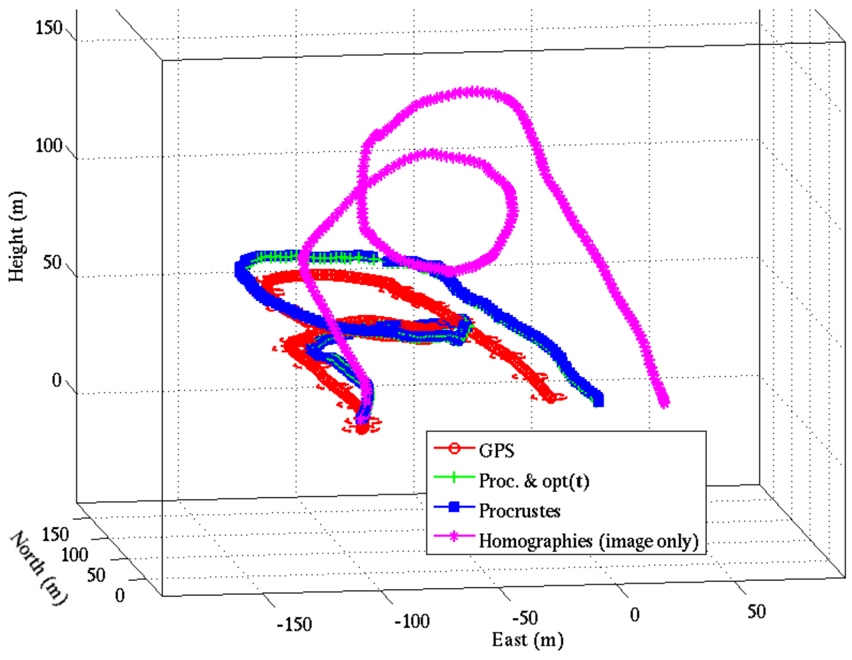
Unit: seconds	RANSAC		Optimization		Total
	Avg.	std.	Avg.	std.	
Homography	0.05	0.02	0.43	0.19	0.48
Visual Odometry: Procrustes only	0.18	0.08	-	-	0.18
Visual Odometry: optimization in $\mathbf{t}$ initialization: Procrustes	0.18	0.08	0.18	0.10	0.36

Table 3.7: Comparison of the execution time of visual odometry techniques.

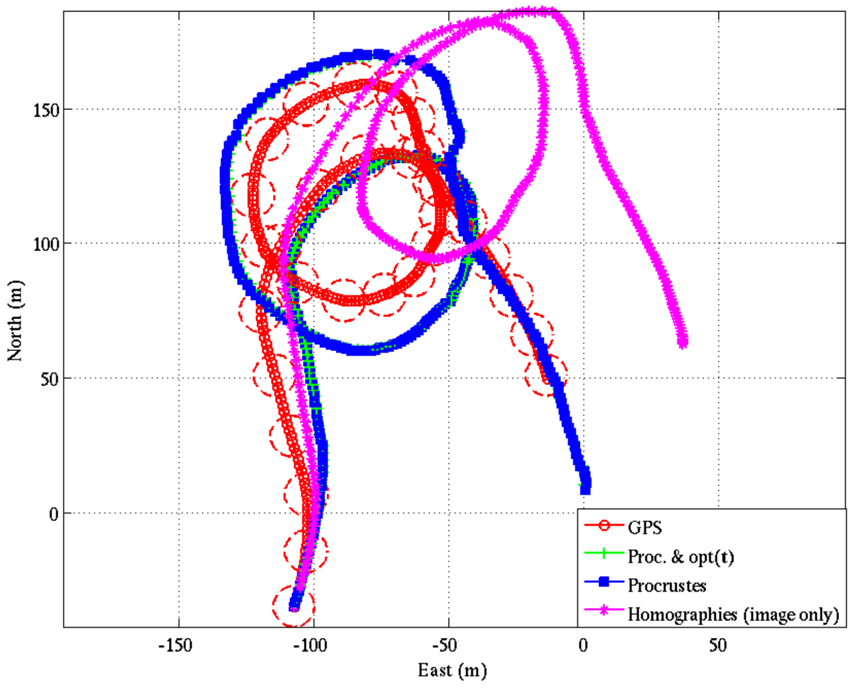
Another image sequence with a recovered trajectory is shown in figure 3.28. The images are reprojected on the ground plane forming a map by using equation (3.4) to find the metric coordinates of the projection of its four corners in the ground plane and drawing the image on the canvas accordingly. The better alignment of the road and lines in the grass field indicates that the map of figure 3.28(a), which utilizes the vehicle poses recovered by the visual odometry, is better registered than the map of figure 3.28(b), which utilizes the GPS position fixes. Both figures utilize the same images orientation data.

3.4.6 Combining GPS and Visual Odometry

This experiment also utilized the MTi AHRS unit. The translation recovered by the visual odometry was fused with GPS position fixes in a Kalman Filter with a constant acceleration model [Bar-Shalom et al., 2001]. The GPS measures the airship position with standard deviation given by the *eph* and *epv* values supplied for each position fix, and translation vectors from the visual odometry are interpreted as a velocity measurement between two successive poses, with a manually set covariance smaller in the vertical axis than in the horizontal ones. The GPS absolute position fixes keep the estimated airship position from diverging, while the visual odometry measurements improve the trajectory locally. The fused trajectory is shown in figure 3.30(a) next to the original GPS trajectory. The latter is shown alone in figure 3.30(b). The maps in figures 3.30(a) and 3.30(b) were generated from the same images and camera orientation values, the only difference being in the camera poses. The fused trajectory is still heavily dependent on the GPS, and its corresponding map is far from perfect, but some details such as the letters and the airfield runway are better registered with the fused trajectory (a) than with GPS alone (b). This dataset contains 1000 images (only 68 are drawn). This trajectory is too large to be recovered by visual odometry alone, therefore the map corresponding to this is not shown.



(a)



(b)

Figure 3.27: 3D (a) and 2D (b) plots comparing visual odometry with GPS position fixes. The circles indicate some of the GPS eph values (i.e. one standard deviation).

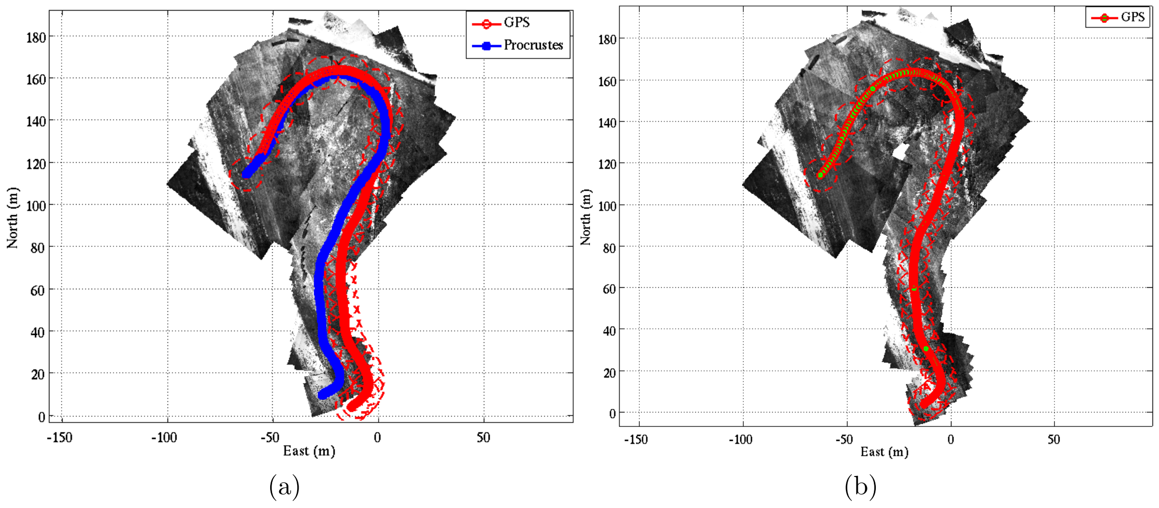


Figure 3.28: Maps formed by reprojecting the images on the ground plane, with trajectories superimposed. The circles indicate some of the GPS eph values (i.e. one standard deviation).

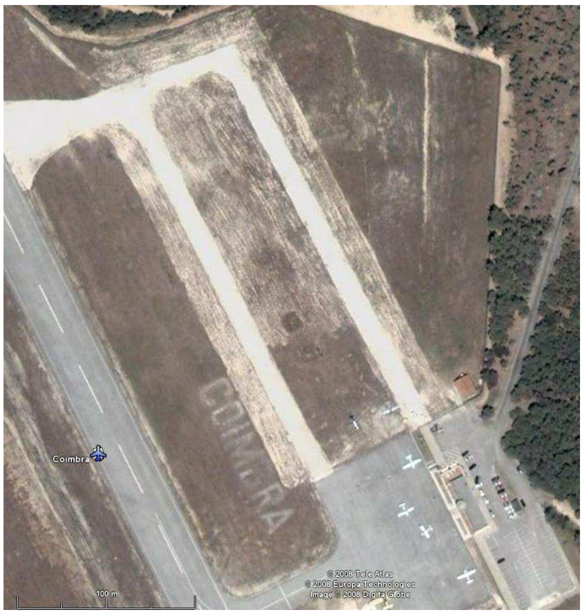
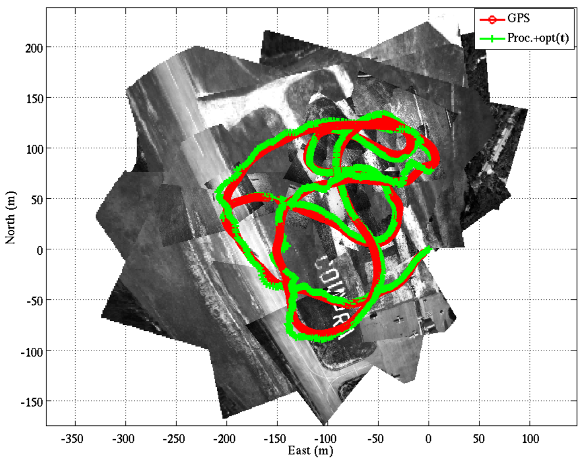
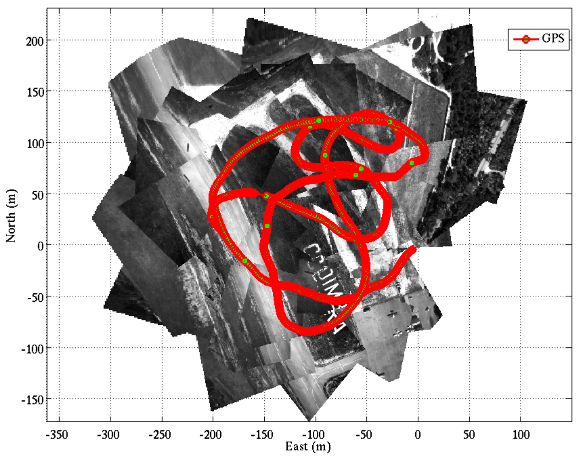


Figure 3.29: A satellite image of the flight area, the Coimbra City Airfield.



(a) Images projected in the ground using the fused trajectory.



(b) Images projected in the ground using the GPS trajectory.

Figure 3.30: Data from GPS and visual odometry fused by a Kalman filter. The images are projected on the ground plane, forming a map.

Chapter 4

Inertial Aided Visual Mapping

4.1 Introduction

The reprojection on the virtual horizontal plane introduced in section 2.7 and detailed in section 3.2.1 can be applied in other domains besides trajectory recovery. This chapter considers other problems such as Image Mosaicing, Plane Segmentation and 3D Mapping from a monocular camera image sequence and registration of point clouds taken from a stereo camera or LRF.

4.2 Image Mosaicing

Building small mosaics

All images are reprojected on the virtual horizontal plane as presented in sections 2.7 and 3.2.1. These images of the virtual downwards camera are denoted by $\mathbf{I}_i^{\mathcal{D}}$. Interest point matching is used to establish point correspondences between each image $\mathbf{I}_i^{\mathcal{D}}$ and an arbitrarily chosen reference image $\mathbf{I}_B^{\mathcal{D}}$. Wrong correspondences will appear as outliers, that are detected by RANSAC, using the homography model.

In the sequence of images, the interest point matching algorithm tries to match the reference image $\mathbf{I}_B^{\mathcal{D}}$ with the images following and preceding it until the image match is not good enough, therefore delimiting an image sequence denoted by $\{\mathbf{I}_1^{\mathcal{D}} \ldots \mathbf{I}_B^{\mathcal{D}} \ldots \mathbf{I}_n^{\mathcal{D}}\}$.

For an image match to be considered "good enough", two conditions must be satisfied:

1. There must be a minimum number of corresponding pixel pairs, after excluding outliers. (A minimum 20 pixel correspondences in this work);

2. The area covered by the corresponding pixels in the image must be large. This is relevant because the homography calculation is better conditioned when dif-

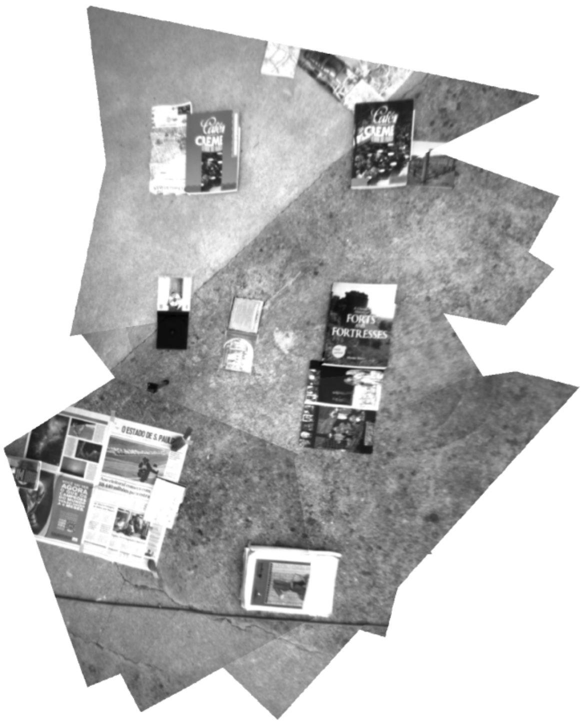


Figure 4.1: Example of mosaic constructed from successive images.

ferent portions of the image are constrained [Hartley and Zisserman, 2000]. In this work the area of the convex hull of all corresponding pixels must be larger than 20% of the image area, in both images.

If any of these two conditions do not hold, the pair of images is considered badly matched, the image sequence also stops.

Then for each image $\mathbf{I}_i^{\mathcal{D}}$, a homography ${}^{B}\mathbf{H}_i$ is calculated to register the image $\mathbf{I}_i^{\mathcal{D}}$ to the reference image $\mathbf{I}_B^{\mathcal{D}}$, and then the images are drawn together. Figure 4.1 shows an example of a mosaic built from a sequence of successive image frames.

It might be asked why the reprojected images were registered using homographies instead of homologies as used in chapter 3. Certainly homologies can be estimated, as it is done with the same kind of dataset in the tripod experiments of section 3.4.1, but the remaining uncompensated rotation is still significant enough to cause visible misalignments on an image pair when drawn together. It is still necessary to estimate homographies with the reprojected images to obtain visually clean small mosaics.

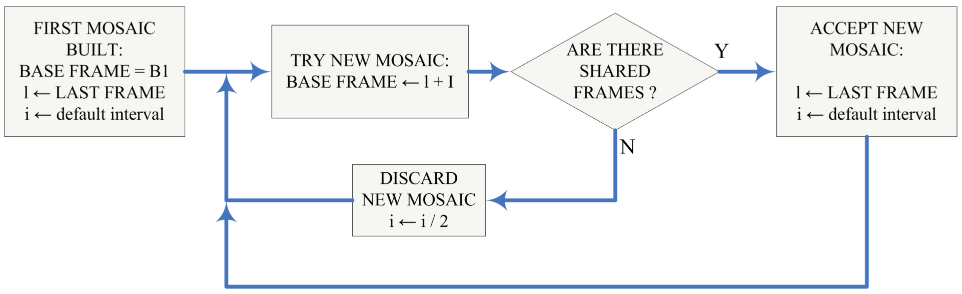


Figure 4.2: Choosing the next base frame for the next mosaic.

Choosing the next base frame.

The purpose of building mosaics from successive frames is to register themselves into a larger mosaic. In order to do that, the choice of the reference frame of the next mosaic in the image sequence must guarantee that consecutive mosaics overlap.

To guarantee this overlapping, a small number of image frames must be shared between two consecutive mosaics. Therefore after setting a default interval between consecutive reference frames, and arbitrarily choosing the base frame of the first mosaic, the process shown in figure 4.2 generates a sequence of overlapping mosaics.

Basically the new mosaic base frame is chosen closer and closer to the last frame used by the last mosaic until both mosaics share enough frames. A few stop conditions were not shown in the figure: if the interval reaches 1 and the new mosaic still does not share frames with the previous one, then the new mosaic is accepted anyway, but the mosaic sequence may be not continuous. And when the last existing frame is used, the algorithm stops.

Mosaicing the mosaics

By construction, most mosaics will overlap with the next mosaic in the temporal sequence, especially when they share image frames. It is possible to apply recursively the same algorithm that registered sequences of successive images, registering sequences of successive mosaics. This should further reduce the number of images, and make the number of chained transformations smaller than if one was chaining all the homographies between every pair of original images. A similar concept was used to generate a panorama from video in the pure rotation case [Steedly et al., 2005], where images in the video are matched to a small number of "key frames", which are in turn matched themselves. Figure 4.3 shows an example of two matched mosaics.

This data shows that the mosaics themselves may be registered by interest point matching. The mosaics shown in this section originate from a dataset of 61 images, taken from a tripod moved manually above a planar patch, set at 2 different heights.

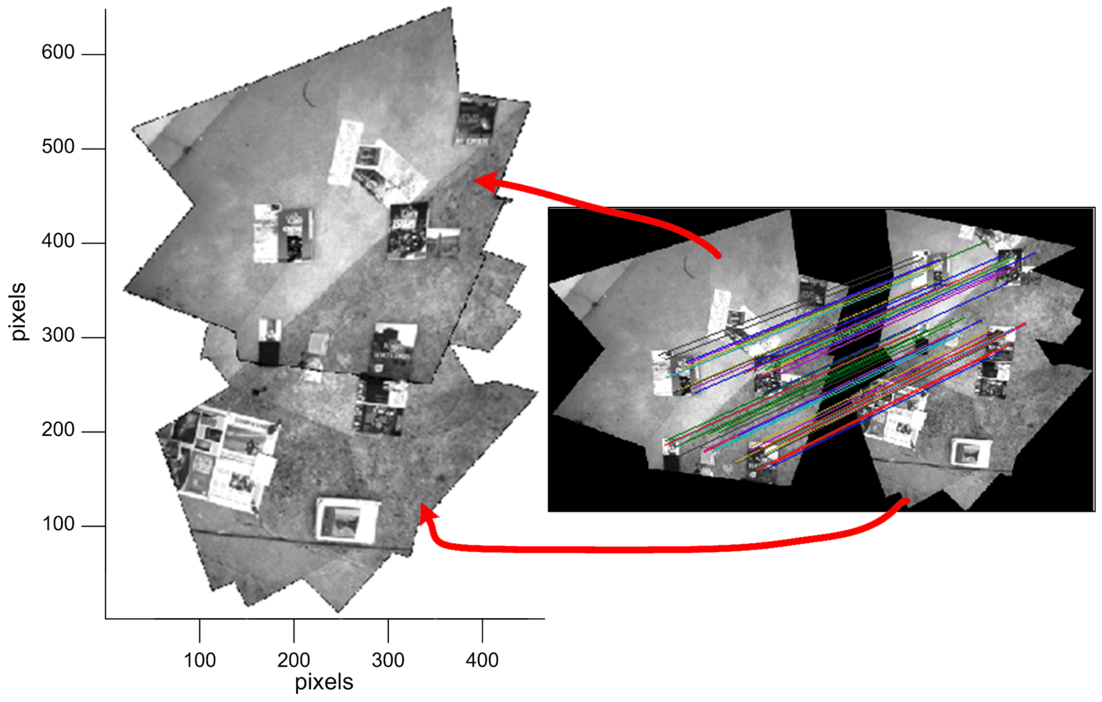


Figure 4.3: On the right, two mosaics, with their pixel correspondences; left, the same mosaics registered and drawn together.

This sequence generated 20 mosaics built from successive pairs of images. Then, a second run over them generated 4 larger mosaics. A final run generated the mosaic of fig. 4.4. A feathering process was applied to smooth image transitions on the last step, and this implied on some ghosting in the books in the middle.

The mosaic of figure 4.4 was obtained without deghosting and bundle adjustment, which usually must be applied to image-only mosaicing of this size [Szeliski, 2004, Capel, 2001]. These techniques still could be used to register larger datasets. Also, some recent results in mosaicing suppose a rotation-only model (e.g [Brown and Lowe, 2003, Szeliski, 2004]), where the camera center is the same for all images. But here the camera is freely translating and rotating.

For comparison, the same interest point matching algorithms were applied to successive frames in the original image sequence as well as in the sequence reprojected into the virtual horizontal plane. The reprojection error (root mean square) on pixel correspondences was 20% less on the reprojected images.

The ratio of correct matches versus total number of interest points detected is better with images taken from the same viewpoint, therefore the algorithm parameters may be tuned to detect less features, while keeping the same number of correct pixel correspondences. Therefore, interest point matching is faster (less features means less descriptor computation and faster matching) or more robust (more correct matches with the same number of detected interest points means greater probability of successful registration), which is a trade-off. For this dataset, the matching of interest point descriptors with the reprojected images was 50% faster, while still yielding a 2% larger number of correct correspondences.

Otherwise, it would be necessary that the feature encoding be invariant to heading and viewpoint differences of the original images [Mikolajczyk and Schmid, 2004].

Note that the small mosaics are built from a small temporal sequence of overlapping images, where the camera pose and viewpoint could not change too much. But during a larger movement, such as when the mosaics themselves are registered, the moving observer may revisit the same area from other points of view, including changes in scale, rotation, and luminosity, and thus interest point matching with the mosaics is more difficult.

The gains obtained with the reprojection to the virtual horizontal plane should be related to the gains of *affine invariant* interest point matching algorithms [Mikolajczyk et al., 2005], i.e., interest point algorithms invariant to affine transformations of planar images - the transformation that a small patch of image suffers when seen under different viewpoints.

Some affine invariant interest point matching algorithms improve the matching reliability under viewpoint change by reprojecting the region around each detected interest point to a viewpoint uniquely determined by the statistical characteristics of that image region [Mikolajczyk and Schmid, 2004]. In our case, the reprojection to

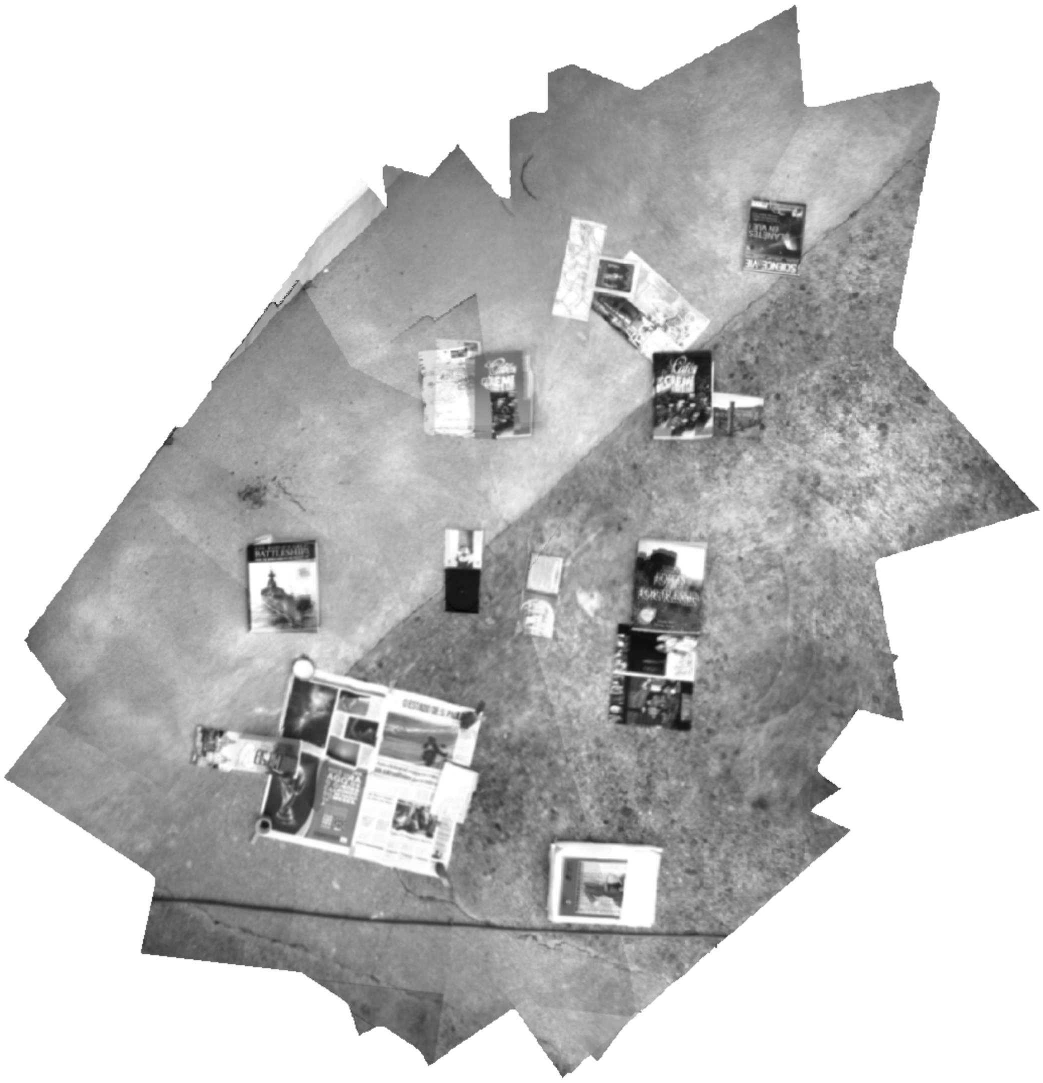


Figure 4.4: A mosaic from 61 registered images.

the virtual horizontal plane essentially transforms each image to generate a virtual image taken from a constant, downwards point of view, and the transformation is estimated from AHRS orientation measurements.

It is not clear in the algorithms comparisons found in the literature [Mikolajczyk et al., 2005], which would be the advantages of affine invariant algorithms over the more usual algorithms invariant only to rotation and scaling in this context. But reprojecting the images into the virtual horizontal plane has been shown here to improve execution times by allowing the point detection thresholds to be more restrictive. Therefore the detected interest points have better distinctiveness, improving the ratio of inliers versus outliers, which facilitates and accelerates the matching and outlier removal steps.

4.3 Plane Segmentation and a Digital Elevation Map From Pixel Correspondences

Recall that equation (3.7), the relation between scene depths and image ratios with the FOE for pixel correspondences, is valid for each individual corresponding pixel pair. Therefore, if an image contains regions above the ground plane, the relative height can be directly recovered for each corresponding pixel. This suffices to order these pixels by their height, but it is not an absolute measurement. Again, additional information is needed to recover scale from imagery.

First, in section 4.3.1, plane segmentation experiments are shown, where 3D planes are segmented on the image and ordered by their depth, with no concern for absolute heights. Then, in the following sections, given the height of the first view, the absolute height is calculated for each pixel correspondence and a coarse Digital Elevation Map (DEM) grid is built, performing 3D mapping from monocular aerial images.

4.3.1 Plane Segmentation Experiments

Planes parallel to the image plane In the following experiment, two images were taken from a staircase scene containing various horizontal planes. As there are various planes on the scene, the process of section 3.2.3.1 can not be applied using all pixel correspondences. Instead, the relative depths for every pixel correspondence were calculated and used to perform plane segmentation by clustering the pixel correspondences belonging to each plane, ordering the planes by their relative height.

To accomplish this, the image pair was reprojected into the virtual horizontal plane, and pixel correspondences were established. The FOE was calculated as in section 2.2.3.3.

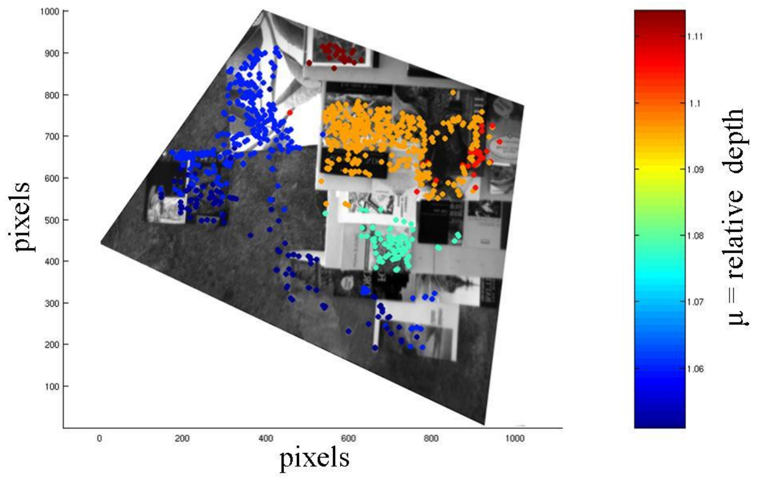


Figure 4.5: The relative depths from two cameras used to order different planes by they height.

The relative depths, i.e., image cross ratios with the FOE, were calculated via equation (3.7) for all corresponding pixel pairs, and these values were arranged into a histogram with 50 bins. Groups of corresponding pixel pairs with close relative depth values were found by picking the peaks the histogram. Figure 4.5 shows each group - corresponding to the ground level and three steps of a staircase - with a different color, and the scale relating colors to relative depths is found on the right of the image.

These points are very fast to obtain, and they can be seeds for other plane segmentation algorithms.

Planes in General Position but known orientation The aim of this experiment is to perform plane segmentation, as above, by clustering corresponding pixels belonging to each plane. The ground plane is still supposed to be horizontal, but now the images are not reprojected on the virtual horizontal plane. Camera orientation measurements from the inertial system were used to compensate for the difference on rotation by calculating the infinite homography between both poses, and applying it to one of the images.

From the camera orientation measurement the vanishing line for horizontal planes is directly calculated on image coordinates, allowing us to look for a plane corresponding to it. The same process also finds other horizontal planes, as the vanishing line is the same, and orders them by their height.

Therefore, two images were taken from a tripod, showing objects on the ground plane and books forming another parallel but higher plane.

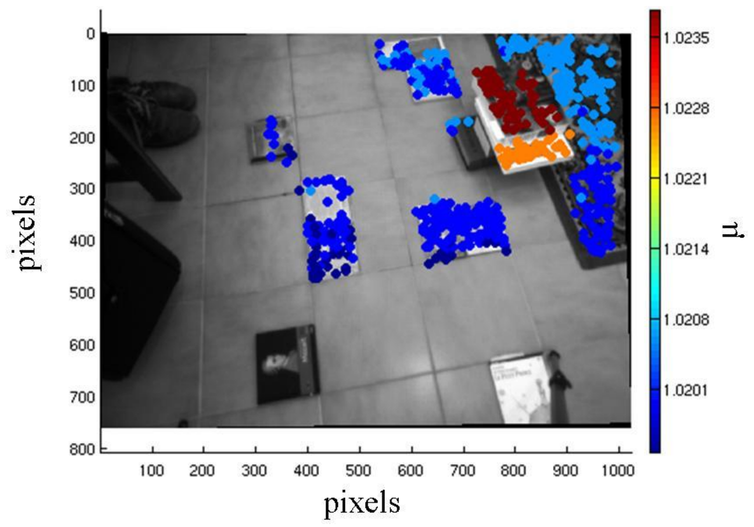


Figure 4.6: The ground plane, and other parallel planes, segmented and ordered by height.

The FOE was robustly estimated from all pixel correspondences [Chen et al., 2003], and outliers were discarded in this step. The horizon line $\mathbf{a}$, i.e., the vanishing line of the ground plane, is obtained via equation (3.25), with $\mathbf{n}$ being the vector normal to the ground plane in the camera frame.

For each corresponding pixel pair $(\mathbf{x}, \mathbf{x}')$, the cross-ratio μ is calculated by:

$$\mu = \frac{\|\mathbf{v}, \mathbf{x}\| \, \|\mathbf{x}', \mathbf{i}\|}{\|\mathbf{v}, \mathbf{x}'\| \, \|\mathbf{x}, \mathbf{i}\|} \tag{4.1}$$

where $\mathbf{i}$ is the intersection of the vanishing line $\mathbf{a}$ and the line $|\mathbf{x}, \mathbf{x}'|$. Again, a histogram is calculated on the values of μ, and by picking the peaks of the histogram we find groups of pixel correspondences corresponding to the same plane.

Figure 4.6 shows the value of the cross-ratio μ (corresponding to relative height) for groups of pixel correspondences.

4.3.2 Calculating Height For Each Pixel Correspondence

In the previous section, the relative height is directly recovered for each corresponding pixel. This sufficed to order these pixels by their height, but it is not an absolute measurement. Again, additional information is needed to recover scale from imagery. The absolute height of these points may be recovered if the absolute height on both views is known - case (a), or, equivalently, if the height of one view and relative height corresponding to the ground plane is known - case (b).

In case (a), define $\mu_i = \frac{dist(\mathbf{x}_i,\mathbf{v})}{dist(\mathbf{x}'_i,\mathbf{v})} = \frac{Z'_i}{Z_i} = \frac{h'-hp_i}{h-hp_i}$ as the relative height for the corresponding pixel pair $(\mathbf{x}_i, \mathbf{x}'_i)$, where hp_i is the height of the 3D point $\boldsymbol{X}_i$ imaged as $\mathbf{x}_i$, and h and h' are the known camera heights, as shown in Fig. 4.7. Then solving for hp_i, we have:

$$hp_i = \frac{\mu_i h - h'}{\mu_i - 1} \tag{4.2}$$

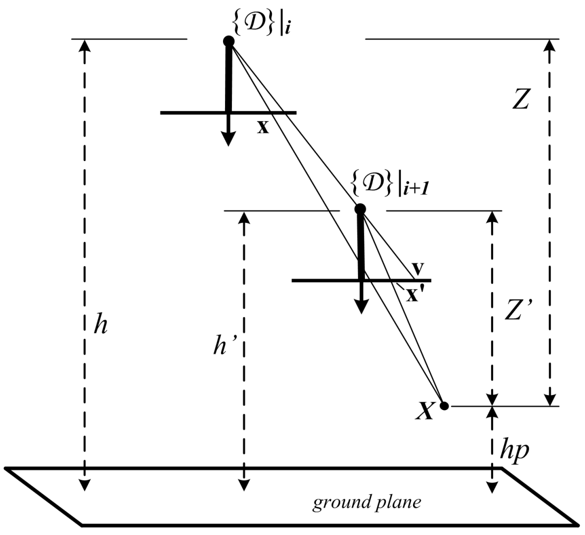


Figure 4.7: Calculating the height of a 3D point from an image pair under pure translation.

In case (b), we substitute $h' = \mu h$, where μ is the relative camera height above the ground plane, into (4.2) to obtain:

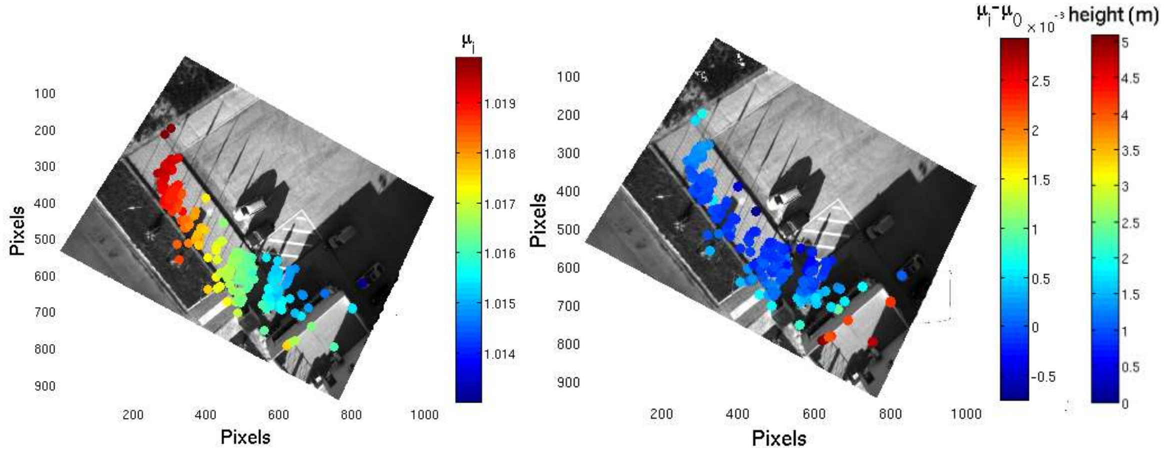
$$hp_i = \frac{(\mu_i - \mu)h}{\mu_i - 1} \tag{4.3}$$

Supposing that the ground plane is visible in the majority of the image, the term $(\mu_i - \mu)$ will be used to compensate for errors in the AHRS orientation measurements. An image wrongly reprojected on the virtual horizontal plane due to errors in the orientation estimate results on a deviation on the image ratios calculated for the pixel correspondences, as shown in section 3.4.4.2. Therefore, given all image ratios for all pixel correspondences, a 3D plane is fitted to the (x_i, y_i, μ_i) triplets, where $\mathbf{x}_i = (x_i, y_i, 1)$ in inhomogeneous image coordinates, using RANSAC to look for the dominant plane on the scene, that should be the ground plane.

This fitted 3D plane is used to compensate for a linear deviation induced by orientation errors, by calculating, for each $\mathbf{x}_i$, the value $(\mu_i - \mu_0(x_i, y_i))$, where $\mu_0(x_i, y_i)$

is the μ value corresponding to the (x_i, y_i) coordinates on the fitted plane, and taking this value as $(\mu_i - \mu)$ on (4.3).

Figure 4.8 shows the image ratios calculated before and after correction with a fitting 3D plane. These points are very fast to obtain, as no homography or homology is recovered, and just image ratios for individual points are needed. The plane fitting is a simple model for RANSAC, and potentially it can be skipped if better measurements are available.



(a) (b)

Figure 4.8: Image ratios before (a) and after (b) compensation by fitting a plane. The points on the building are the only ones above the ground plane.

4.3.3 Constructing a DEM

A DEM is a 2D grid dividing the ground plane into equal square regions called cells, where the height of each cell is represented as a probability density. A DEM can be constructed from punctual height measurements if the 3D coordinates of each measured point are known. The camera geometry allows us to find the remaining two horizontal coordinates if the height from the ground plane is known as presented in section 4.3.2. The distance r_i between the principal axis of the virtual downward camera and the 3D point $\boldsymbol{X}_i = (X_i, Y_i, hp_i)$ may be calculated by similarity of triangles (see Fig. 4.7) as:

$$r_i = \frac{dist(\mathbf{x}_i, \mathbf{n})}{f}(h - hp_i) \tag{4.4}$$

where $\mathbf{x}_i$ is the image projection of $\boldsymbol{X}_i$ and $\mathbf{n}$ the principal point of the virtual downward camera. The angle θ_i between the line $\overline{\mathbf{x}_i\mathbf{n}}$ and the east axis is directly calculated from their coordinates. Then, transforming (r_i, θ_i) from polar to rectangular coordinates yields (X_i, Y_i), the remaining two components of $\boldsymbol{X}_i = (X_i, Y_i, hp_i)$, in the $\{\mathcal{D}\}|_i$ frame. If the camera pose in the world frame is known, these coordinates may be registered into the $\{\mathcal{W}\}|_i$ frame, incorporating the height measurements in a global DEM.

Each point $\boldsymbol{X}_i$ represents a height measurement for an individual DEM cell. The height of each cell and the heights measurements are represented by random variables with Gaussian distribution, and the cell update follows the Kalman Filter update rule. There is not an initialization and the cell takes its first value when incorporating its first measurement.

The position of each point $\boldsymbol{X}_i$ has also an uncertainty on the xy axes, i.e., it may be uncertain which cell the measurement belongs to. Currently the influence of measurements on neighboring cells is approximated by considering all measurements exact on the North-East axes, and then smoothing each local DEM with a Gaussian kernel with variance similar to the measurements.

Figure 4.9 shows a DEM constructed from a 10 frame sequence (a $5[s]$ portion of the flight), with the GPS-measured vehicle localization shown as red stars. The cell size is $3[m]$, and the Gaussian convolution kernel has standard deviation of $2[m]$. The highest cells correspond to the building, and the smaller blue peaks correspond to the airplanes and vehicles. Points more than $1[m]$ below the ground plane are discarded as obvious outliers. The mosaic on the left - included just to allow the reader to compare visually the covered area and the DEM grid - is built from homographies calculated between each successive image pair as in section 4.2.

4.3.4 Applications and Future Improvements

In section 3.4 the complete UAV trajectory was reconstructed from rotation compensated imagery. In this section, a 3D map of the environment, in the form of a sparse DEM, is built from punctual height measurements generated from a sequence of images and the vehicle pose. These two techniques assume mutually exclusive conditions: the trajectory recovery requires imaging a planar area, and the mapping procedure is meaningful only if there are obstacles above the ground plane. Therefore, they could not make a complete SLAM (Simultaneous Localization and Mapping) scheme on their own, but they may be complimentary methods to other SLAM approaches. A SLAM scheme also needs to obtain, from the maps built at each step, another displacement measurement to constrain the vehicle pose estimation, by matching local maps with the previous local map or with the global map built so far. A comparison of different approaches for matching 2D occupancy grids built from sonar data indicates that both options are feasible [Schiele and Crowley, 1994], and the DEM grids could be correlated this way.

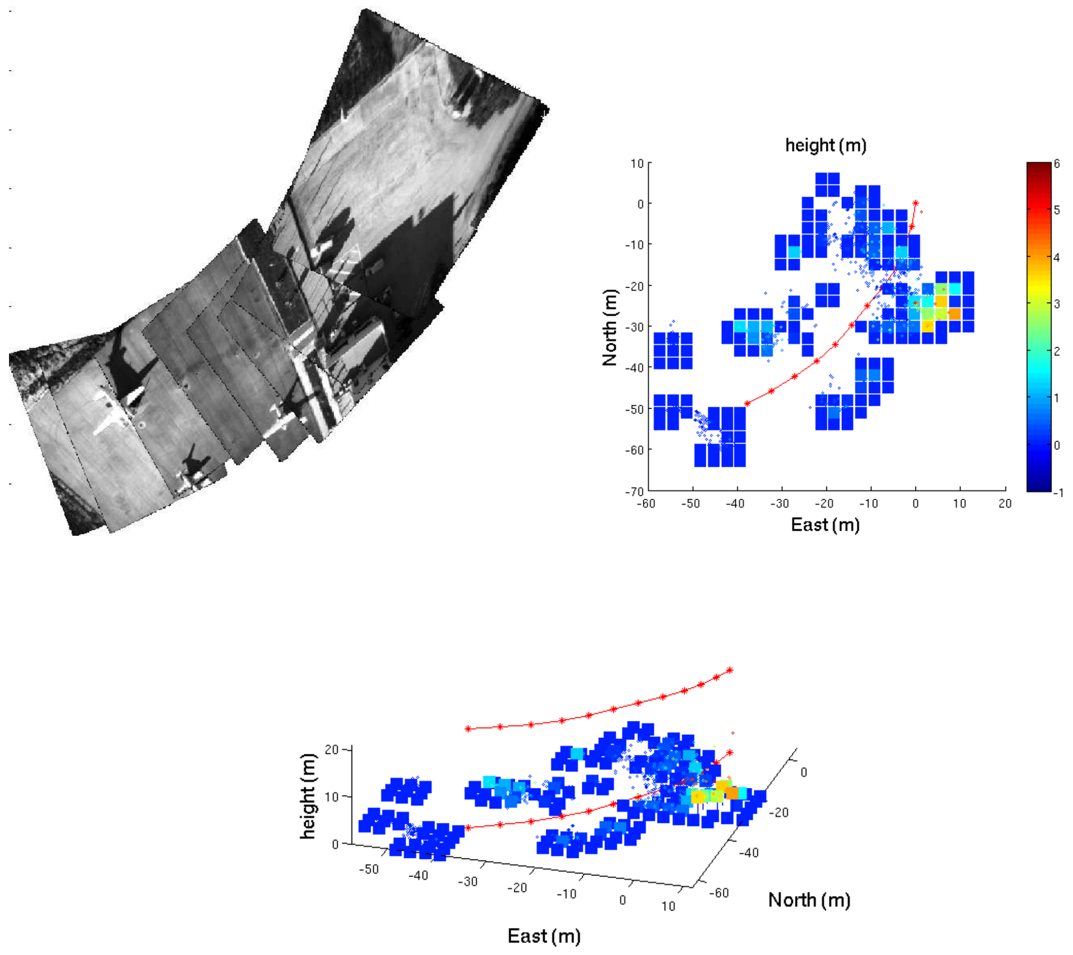


Figure 4.9: 2D and 3D plots of the DEM generated from the 10 frames mosaiced on the left. The red stars indicate the vehicle trajectory.

Figure 4.8 shows that all pixel correspondences are concentrated into one small portion of the image. Outside this region, there are other correspondences found by the interest point matching algorithm, but they are considered outliers during the FOE estimation and rejected, probably due to errors on the orientation measurement. For the estimation of the homology, and particularly of its cross ratio used on the height component of the trajectory estimation, usually there are enough correspondences for a reliable estimation. But for the mapping problem, the area covered by height measurements is quite smaller than the total area imaged (see Fig. 4.9), and thus the map becomes too sparse for a reliable correlation between the DEM grid maps. This problem could be addressed by improving the accuracy of the orientation measurements, by increasing the image frame rate, or by developing models that take into account small uncompensated rotations.

4.4 3D Depth Map Registration

This section describes the registration of 3D maps taken from a moving ranging device, which may be a calibrated stereo camera, or a LRF coupled with a single camera. There is also an AHRS rigidly coupled with the ranging device as before. The maps consist of sets of 3D points, or point clouds, defined in the frame corresponding to the ranging device used, i.e., the $\{\mathcal{C}\}|_i$ or $\{\mathcal{L}\}|_i$ frames. These point clouds must be registered in a common world frame $\{\mathcal{W}\}$. The models to calculate these point clouds from the raw sensor readings at each frame index i are given in sections 2.1.1 and 2.4, and they are denoted by ${}^{\mathcal{C}}\mathbb{P}|_i$ or ${}^{\mathcal{L}}\mathbb{P}|_i$, where the left superscript represent the frame of reference and the right subscript represent the time index.

Each point cloud is associated to an image $\mathbf{I}_i$ captured at the same instant by a digital camera. This image is the left camera image in the stereo camera case. In the LRF case, the acquisition of the point cloud is not instantaneous, therefore the pantilt position at the moment of image acquisition must be known. For these results the image was always acquired with the tilt and pan angles zeroed.

First, an image sequence is delimited, consisting of images that can be successfully matched with an arbitrarily chosen reference image $\mathbf{I}_B$ by interest point matching algorithms (section 4.4.1). Then for each point cloud, the rotational components are compensated using measurements of the camera orientation in section 4.4.2, and image correspondences find the remaining translational component in section 4.4.3.

The output for each frame is a point cloud registered into a common frame, the frame $\{\mathcal{W}\} = \{\mathcal{R}\}|_B$. The registered point clouds are not kept separately as in figure 4.22(a) but are combined into an aggregated point cloud (figure 4.22(b)), discarding redundant points to save memory.

Figure 4.10 shows the data flow for the registration of each point cloud. On the left of the figure, the inputs are shown: for each time index i, the stereo point cloud, the image from the left camera, and the inertial orientation measurement. Two other inputs are constant for all frames: the image from the reference frame, and the rotation matrix ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$, output of the camera-inertial calibration.

4.4.1 Delimiting an image sequence

To delimit the subsequence to be registered, interest point matching is applied to match the reference image $\mathbf{I}_B$ with the next images on the sequence until an image is found that can not be successfully matched with the reference frame, as shown in figure 4.11 (actually, for the sake of clarity the figure omits that the loop is executed twice, first matching frames in the forward direction and then backwards). An image

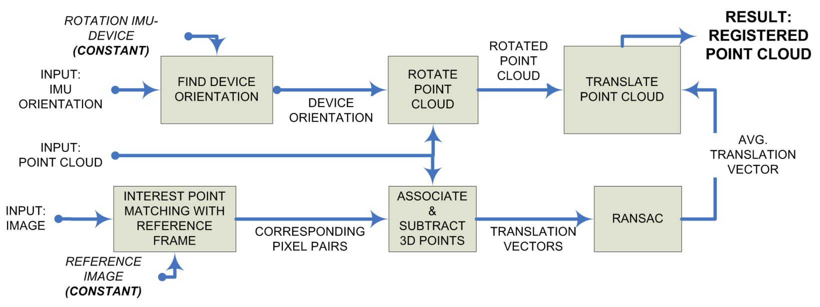


Figure 4.10: The data flow for the registration of each 3D point cloud.

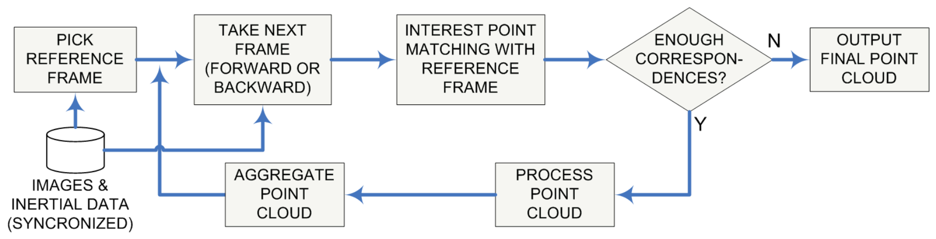

PICK REFERENCE FRAME
TAKE NEXT FRAME (FORWARD OR BACKWARD)
INTEREST POINT MATCHING WITH REFERENCE FRAME
ENOUGH CORRESPON-DENCES?
N
OUTPUT FINAL POINT CLOUD
Y
IMAGES & INERTIAL DATA (SYNCRONIZED)
AGGREGATE POINT CLOUD
PROCESS POINT CLOUD

Figure 4.11: The process for the registration of 3D point clouds. The box “process point cloud” corresponds to figure 4.10.

match is deemed successful if the number of corresponding pixel pairs is bigger than a threshold. The frames on the delimited image sequence are numbered with a time index $i = 1 \dots B \dots n$. The inertial orientation measurement corresponding to each time index i is also available.

4.4.2 Rotating to a stabilized frame of reference

The measured inertial orientation for the time index i is expressed as a rotation matrix ${}^{\mathcal{R}}\mathbf{R}_{\mathcal{I}}|_i$ which rotates the inertial sensor frame $\{\mathcal{I}\}|_i$ into the stabilized frame $\{\mathcal{R}\}|_i$.

If stereo cameras are used as the ranging device, the point cloud ${}^{\mathcal{C}}\mathbb{P}|_i$ is rotated from the camera frame to the stabilized frame by the rotation ${}^{\mathcal{R}}\mathbf{R}_{\mathcal{C}}|_i = {}^{\mathcal{R}}\mathbf{R}_{\mathcal{I}}|_i \cdot {}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$. As figure 4.12 shows for two point clouds, the purpose of this rotation is to align all point clouds to the earth-referenced $\{\mathcal{R}\}|_i$ frame, i.e., North, East and vertical directions, as indicated by the inertial and magnetic orientation measurements.

For the LRF, a similar step is taken, but it is necessary to compensate for the pantilt position at the moment the inertial measurement was taken. The pantilt rotation is represented by $\mathbf{R}(\varphi, 0, 0)$, the rotation matrix equivalent to the Euler angles φ, 0, 0 considering φ as the tilt angle and zero as the other two Euler angles. For each time index $i = 1 \dots n$, the rotation matrix ${}^{\mathcal{W}}\mathbf{R}_{\mathcal{L}}|_i = {}^{\mathcal{W}}\mathbf{R}_{\mathcal{I}}|_i \cdot {}^{\mathcal{I}}\mathbf{R}_{\mathcal{L}} \cdot \mathbf{R}(\varphi, 0, 0)$ brings a point from the laser frame $\{\mathcal{L}\}|_i$ into the $\{\mathcal{R}\}|_i$ frame, this rotation is applied

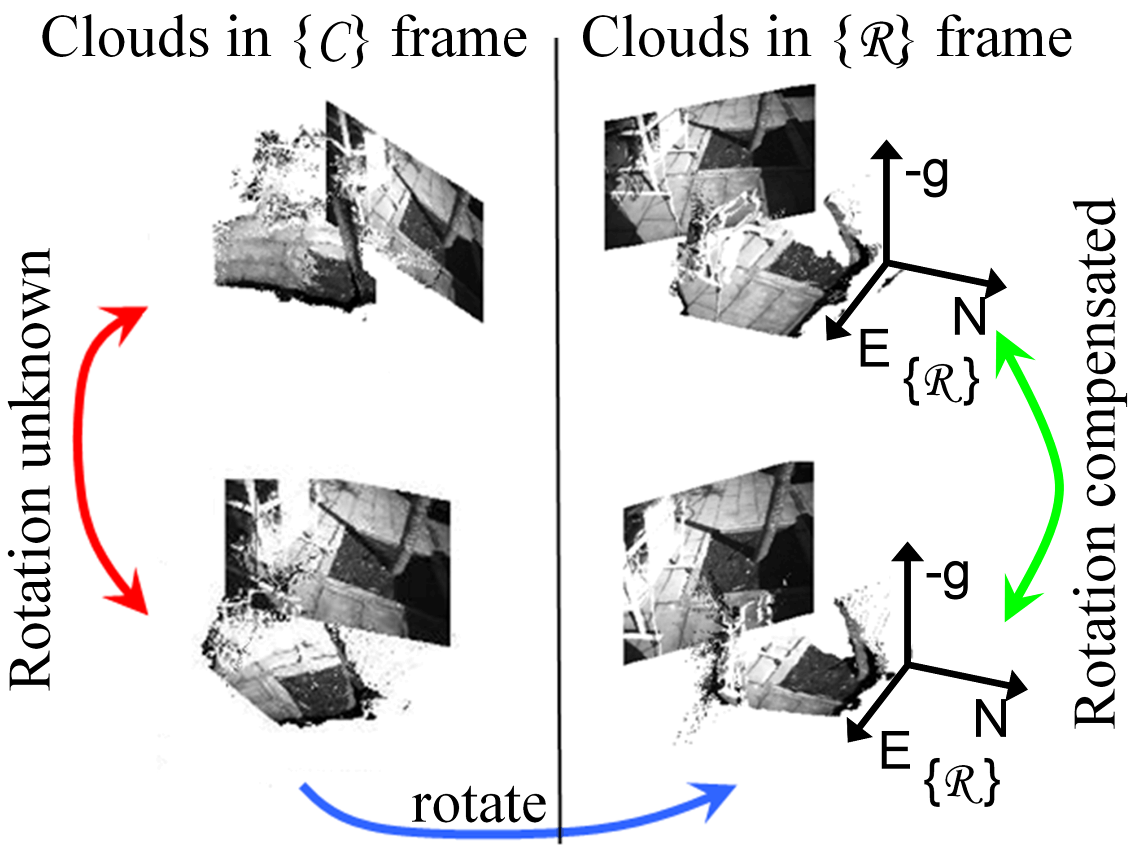


Figure 4.12: Compensating the rotation: point clouds aligned on inertial stabilized frame.

to all 3D points on the point cloud ${}^{\mathcal{L}}\mathbb{P}|_i$, generating a point cloud ${}^{\mathcal{R}}\mathbb{P}|_i$ in the rotated frame of reference.

This calculation depends on the rotations between the inertial and the ranging devices, which are constant in time as the devices are rigidly mounted together. The fixed rotation ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{C}}$ is found by the inertial camera calibration of section 2.6. In this thesis, the fixed rotation ${}^{\mathcal{I}}\mathbf{R}_{\mathcal{L}}$ was found by using the camera image to paint the point cloud, and adjusting the rotation between the camera and the laser until the point cloud appeared to be correctly painted.

After this step only a translation is missing to register the point cloud into the $\{\mathcal{W}\}$ frame.

4.4.3 Translation to a common frame.

The objective now is to register the a set of point clouds $\{{}^{\mathcal{R}}\mathbb{P}|_1 \cdots {}^{\mathcal{R}}\mathbb{P}|_B \cdots {}^{\mathcal{R}}\mathbb{P}|_n\}$, by finding the translation between each $\{\mathcal{R}\}|_i$ frame and the base $\{\mathcal{W}\} \triangleq \{\mathcal{R}\}|_B$ frame, for $i = 1 \ldots B \ldots n$.

Every point cloud ${}^{\mathcal{R}}\mathbb{P}|_i$ must overlap with the reference point cloud ${}^{\mathcal{R}}\mathbb{P}|_B$. A pair of point clouds overlaps when, after correct registration, the 3D volumes covered by each point cloud have significant intersection. To register a pair of point clouds, the algorithm searches pairs of 3D points, one from each point cloud, that correspond to the same point on the $\{\mathcal{W}\}$ world frame.

The stereo processing associates each 3D point in the point cloud with a pixel on its corresponding image. Thus, if corresponding pixel pairs are found in the images $\mathbf{I}_l|_i$ and $\mathbf{I}_l|_B$, then their associated stereo 3D points $\boldsymbol{P}|_i$ and $\boldsymbol{P}|_B$ should be a pair of corresponding 3D points, with $\boldsymbol{P}|_i \in {}^{\mathcal{R}}\mathbb{P}|_i$ and $\boldsymbol{P}|_B \in {}^{R}\mathbb{P}|_B$. This pair of 3D points indicates the translation $\mathbf{t}$ between the $\{\mathcal{R}\}|_i$ and $\{\mathcal{R}\}|_B$, in the form:

$$\mathbf{t} = \boldsymbol{P}|_i - \boldsymbol{P}|_B \tag{4.5}$$

Figure 4.13 illustrates the concept of two corresponding 3D points in two overlapping point clouds, obtained from a corresponding pixel pair on the respective images. The translation vector is the dashed blue arrow, and the corresponding 3D points and image pixels are connected by full green lines.

If the images $\mathbf{I}_l|_i$ and $\mathbf{I}_l|_B$ have enough overlap for interest point matching, then a set of corresponding pixel pairs can be found, and therefore a set of translation vectors. Note that outliers must be removed from this set.

Besides the outliers in interest point matching, for some pixels, the stereo processing algorithm may not be able to find disparities, or it may find wrong disparities, due to lack of texture, depth discontinuities, etc. In this case, a corresponding pixel

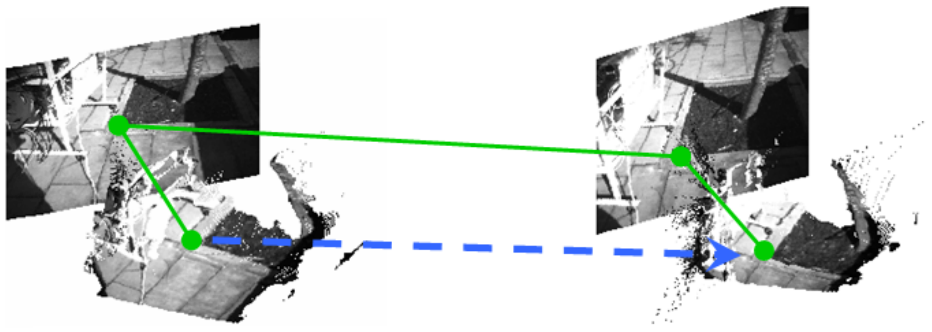


Figure 4.13: The concept of a corresponding pair of 3D points with a translation vector.

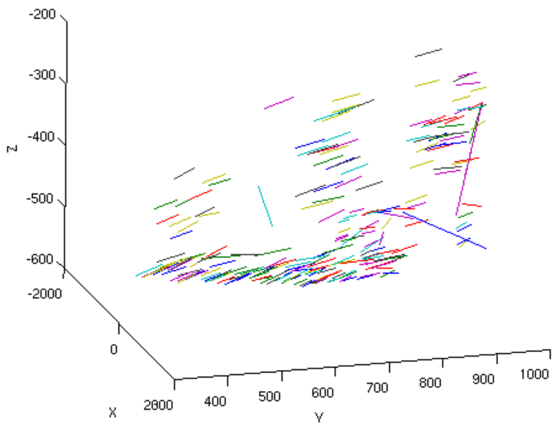
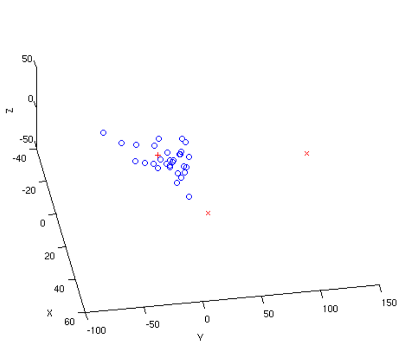


Figure 4.14: An example of a translation vector set with outliers.

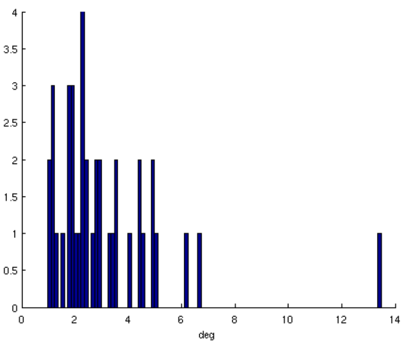
pair may not have any corresponding 3D point pair at all, or it may correspond to the wrong 3D point, and this implies that on incorrect translation vectors, which appear as outliers as well.

An example of a set of translation vectors between corresponding 3D points in two rotated point clouds is shown in figure 4.14, including outliers.

Assuming that the majority of corresponding pixel pairs are from the static background, a robust estimation algorithm such as RANSAC [Fischler and Bolles, 1981], utilizes directly the set of translation vectors to discriminate *inliers* and *outliers*, utilizing a simple Euclidean translation vector model (a simpler model implies that the estimation is faster), and detecting and eliminating outliers caused by errors on both the stereo and interest point matching processes *in a single* robust estimation process. The inlier translation vectors are averaged to obtain the final translation vector to translate the point cloud ${}^{\mathcal{R}}\mathbb{P}|_i$ from the $\{\mathcal{R}\}|_i$ frame into the $\{\mathcal{W}\}$ frame, obtaining a point cloud ${}^{\mathcal{W}}\mathbb{P}|_i$.



(a) Difference between translations vectors and the mean vector (in mm).



(b) [Histogram of the angle between the translation vectors and the mean vector.

Figure 4.15: An example of the usage of RANSAC to detect outliers.

Figure 4.15(a) is an example of the usage of RANSAC to detect outliers in one set of translation vectors from one image pair. The plotted circles are the differences between each vector and the mean vector - i.e., if all vectors were the same, all circles would appear on the origin. The crosses ($\times$) are outliers. The '+' sign is the mean of the inliers. Figure 4.15(b) is a histogram of the angle between the translation vectors and the mean vector, showing that most of them point approximately to the same direction, except a few outliers corresponding to the crosses on the left figure.

By repeating this process for every frame $i = 1 \ldots n$ (skipping $i = B$) the algorithm calculates a set of point clouds $\{{}^{\mathcal{W}}\mathbb{P}|_1 \cdots {}^{\mathcal{W}}\mathbb{P}|_B \cdots {}^{\mathcal{W}}\mathbb{P}|_n\}$, where all point clouds have been registered into the same frame of reference $\{\mathcal{W}\}$.

In the LRF scenario, as the LRF angular resolution is typically less than the image pixel angular resolution, most pixels do not have an associated 3D point. When one pixel $\mathbf{x}$, belonging to a matched interest point pair, do not have an associated 3D point, the closest pixel $\overset{*}{\mathbf{x}}$ with an associated 3D point is found. If the image distance $\left|\mathbf{x} - \overset{*}{\mathbf{x}}\right|$ is less than a small threshold (2 pixels in our experiments) that 3D point substitutes the missing one, otherwise the corresponding pixel pair is discarded.

This approximation increases the error of the measured translation, dominating errors on the individual range measurements, but may be necessary to obtain enough pairs of 3D points. Note that there is not any guarantee that the 3D point corresponding to $\overset{*}{\mathbf{x}}$ will be close to the right place, as the line $\overline{\mathbf{x}\overset{*}{\mathbf{x}}}$ in the image may correspond to a depth discontinuity in the 3D scene. Therefore this association step must be performed before RANSAC, in order that any badly associated 3D point may appear as an outlier for RANSAC.

Therefore, after using RANSAC to remove outliers, for each remaining corresponding pixel pair $(\mathbf{x}_k, \mathbf{x'}_k)$ in the images I_i and I_j, the value $w_k = \left(\left|\mathbf{x}_k - \overset{*}{\mathbf{x}}_k\right| + \left|\mathbf{x'}_k - \overset{*\prime}{\mathbf{x}}_k\right|\right)^{-1}$ is defined as a weight in the averaging of the resulting translation vector, as suggested in [Matthies and Shafer, 1987], and therefore the translation vector is calculated by:

$$\mathbf{t} = \left(\sum_{for\,all\,k} w_k\right)^{-1} \left(\sum_{for\,all\,k} w_k \mathbf{t}_k\right) \tag{4.6}$$

where $\mathbf{t}_k = \boldsymbol{P}_k|_i - \boldsymbol{P'}_k|_j$, with $\boldsymbol{P}_k|_i$ and $\boldsymbol{P'}_k|_j$ being the pair of 3D points associated with the corresponding pixel pair $(\mathbf{x}_k, \mathbf{x'}_k)$. For the corresponding pixel pairs where this approximation is not necessary, their weights should be set at some larger value, as equation (4.6) would yield $w_k = (0+0)^{-1} = \infty$. In the experiments in this thesis they were set at 2.

In the stereo scenario, this approximation is not usually necessary, as many corresponding pixel pairs have associated 3D points. A covariance for each 3D point can be calculated using stereo triangulation error models, and used as weights in the averaging of translation as presented in [Matthies and Shafer, 1987].

4.4.4 Subsuming other geometric constraints for outlier removal

Other geometric constraints have been proposed to remove outliers from the set of corresponding 3D points. In this section, the two constraints proposed in [Hirschmüller et al., 2002] are reviewed and it is shown that in the rotation compensated case they may be enforced more tightly and subsumed by the translation vector constraint proposed in section 4.4.3.

Consider two pairs of corresponding 3D points on the i and j frames, $(\boldsymbol{P}|_i, \boldsymbol{P}|_j)$ and $(\boldsymbol{Q}|_i, \boldsymbol{Q}|_j)$. The first constraint concerns the invariance of the length of the $\boldsymbol{P}|_i - \boldsymbol{Q}|_i$ vector under a rigid transformation, i.e. $\|\boldsymbol{P}|_i - \boldsymbol{Q}|_i\| = \|\boldsymbol{P}|_i - \boldsymbol{Q}|_i\|$ must be true.

The second constraint limits the angle between a vector formed by two 3D points before and after the motion. With the orientation measured, it can be more tightly enforced with a statistically significant boundary if the error on orientation measurements is known.

Represent the rotation between the poses i and j by a rotation axis r and a rotation angle ϕ. The angle between the vectors $\boldsymbol{P}|_j - \boldsymbol{Q}|_j$ and $\boldsymbol{P}|_i - \boldsymbol{Q}|_i$ must be between 0 and ϕ. It is zero if $\boldsymbol{P}|_i - \boldsymbol{Q}|_i$ is parallel to r, and it is ϕ if $\boldsymbol{P}|_i - \boldsymbol{Q}|_i$ is orthogonal to r.

Imposing the restriction $\phi < \theta$ results in the inequality:

$$\left(\frac{\boldsymbol{P}|_i - \boldsymbol{Q}|_i}{\|\boldsymbol{P}|_i - \boldsymbol{Q}|_i\|}\right) \cdot \left(\frac{\boldsymbol{P}|_j - \boldsymbol{Q}|_j}{\|\boldsymbol{P}|_j - \boldsymbol{Q}|_j\|}\right) > \cos\theta$$

If for such two pairs of corresponding 3D points this inequality do not hold, at least one 3D point is an outlier. In [Hirschmüller et al., 2002], with large and unknown camera rotation, θ was set at 45°, a loose constraint, but if it were tighter, the camera rotation would be limited. If the point clouds are already rotated to a stabilized frame, ϕ should be zero, but in practice it has a small value due to errors on the orientation measurements. But, if the orientation measurement error is bounded, ϕ is bounded too.

Suppose the AHRS orientation measurement errors are normal variables $\mathcal{N}(0, \sigma^2_{IMU})$. ϕ is the sum of the uncompensated rotation on the two frames, and thus ϕ is normally distributed with $\phi = \mathcal{N}(0, 2 \cdot \sigma^2_{IMU})$. Therefore θ can be set as a function of σ^2_{IMU} and the percentage of discarded outliers can be predicted by standard statistical techniques. What is missing in this scheme is to consider the positioning error of the 3D points, that will be relevant as σ^2_{IMU} is small.

With the rotation compensated, both constraints can be subsumed by one constraint, the difference of translation vectors (section 4.4.3). To take into account the errors on 3D points positioning the error in orientation measurements is approximated as a random perturbation on the position of the 3D point $\boldsymbol{Q}|_j$, with magnitude $\mathcal{N}(0, \sigma^2_{rot})$, where $\sigma^2_{rot} = \|\boldsymbol{Q}|_j - \boldsymbol{Q}|_i\| \sin(2 \cdot \sigma^2_{IMU})$. Supposing that each 3D point $\boldsymbol{P}$ has a covariance matrix $\Sigma(\boldsymbol{P})$, define $\Sigma(\boldsymbol{Q}|_j)' = \Sigma(\boldsymbol{Q}|_j) + (\sigma^2_{rot} I)$ as an updated $\Sigma(\boldsymbol{Q}|_j)$ matrix. An approximation of the covariance of the difference $\Delta\mathbf{t} = \mathbf{t}_{\boldsymbol{P}} - \mathbf{t}_{\boldsymbol{Q}}$ is calculated as:

$$\Sigma(\Delta\mathbf{t}) = \Sigma(\boldsymbol{P}|_j) + \Sigma(\boldsymbol{P}|_i) + \Sigma(\boldsymbol{Q}|_j)' + \Sigma(\boldsymbol{Q}|_i) \tag{4.7}$$

If both translation vectors $\mathbf{t}_{\boldsymbol{P}}, \mathbf{t}_{\boldsymbol{Q}}$ are consistent, the Mahalanobis distance between then, $(\mathbf{t}_{\boldsymbol{P}} - \mathbf{t}_{\boldsymbol{Q}})^T \Sigma(\Delta\mathbf{t})(\mathbf{t}_{\boldsymbol{P}} - \mathbf{t}_{\boldsymbol{Q}})$, will be small. Otherwise one of the four 3D points is an outlier. The threshold should be set as a function of the desired outlier rejection ratio. The difference between translation vectors is a single constraint that is not satisfied if any of the two other constraints is not satisfied, and therefore it subsumes both.

A non-iterative outlier rejection scheme was implemented as presented by [Hirschmüller et al., 2002], using this constraint instead of the two constraints originally proposed. The constraint is checked against all possible pairs of corresponding 3D points, and the maximum subset of consistent translation vectors is selected by searching for a maximal clique in a matrix graph were all consistent translations are connected.

Although this method is not interactive, it takes longer to execute than the RANSAC method presented in section 4.4.3, because the translation vector model is simple, allowing RANSAC to be executed very fast, without need to test every possible pair of translation vectors. Therefore, RANSAC was used on the experiments of this thesis to exclude outliers, and the non-iterative method was abandoned.

Nevertheless, using equation (4.7) with typical values for the covariance of the 3D points and the AHRS orientation error, yields an expected covariance for the difference between translation vectors which may be used to define the threshold for inlier/outlier discrimination in the RANSAC outlier detection method.

4.4.5 Filtering out redundant points

The algorithm described on the last sections registers a set of overlapping point clouds $\{{}^{\mathcal{W}}\mathbb{P}|_1 \cdots {}^{\mathcal{W}}\mathbb{P}|_B \cdots {}^{\mathcal{W}}\mathbb{P}|_n\}$ into the frame of reference $\{\mathcal{W}\}$. The simple union of all point clouds in this set is a registered point cloud itself, but, due to overlapping, there may be a many 3D points from different point clouds which are very close and are considered redundant. They should be eliminated to save memory and processing time.

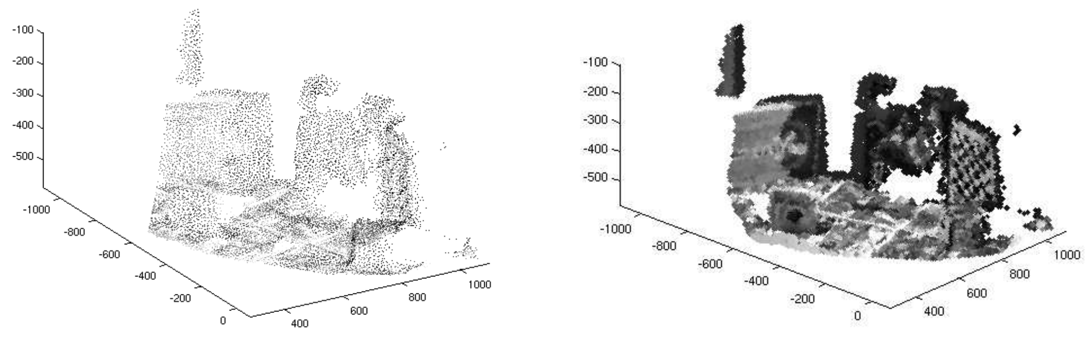
Additionally, the point clouds must be filtered, deleting points wrongly positioned due to errors on stereo processing. Isolated, "floating" 3D points must be eliminated, generating a smoother point cloud. The outlier detection process of section 4.4.3 deals only with the relatively small number of 3D points corresponding to the pixels matched by the interest point matching algorithm, and do not change the point clouds themselves.

The registered point clouds in the set $\{{}^{\mathcal{W}}\mathbb{P}|_1 \cdots {}^{\mathcal{W}}\mathbb{P}|_B \cdots {}^{\mathcal{W}}\mathbb{P}|_n\}$ do not need to actually exist at the same time; Instead, they are calculated sequentially by the process described in the last sections, and aggregated into a final aggregated point cloud ${}^{\mathcal{W}}\mathbb{A}$. ${}^{\mathcal{W}}\mathbb{A}$ is initialized with ${}^{\mathcal{W}}\mathbb{A} = \emptyset$, and as each point cloud ${}^{\mathcal{W}}\mathbb{P}|_i$ is calculated, a subset of the points on ${}^{\mathcal{W}}\mathbb{P}|_i$ is added into ${}^{\mathcal{W}}\mathbb{A}$. Namely, the points of ${}^{\mathcal{W}}\mathbb{P}|_i$ that are closer than a threshold value d_{min} to any point already in ${}^{\mathcal{W}}\mathbb{A}$ are discarded.

The value of d_{min} determines the final resolution of ${}^{\mathcal{W}}\mathbb{A}$. It may be close to the resolution of the original point clouds, or d_{min} may be set to a higher value, leading to a coarser point cloud which occupies less memory.

As the number of points is typically large, it would be too slow to check linearly all the stored points in ${}^{\mathcal{W}}\mathbb{A}$ to test if each new point in ${}^{\mathcal{W}}\mathbb{P}|_i$ is redundant. We have chosen the well-known approach of keeping only the point cloud ${}^{\mathcal{W}}\mathbb{A}$ and a hash table, that indexes all its points by their x-y coordinates. The hash table is used to retrieve a list of potentially close points for each point in ${}^{\mathcal{W}}\mathbb{P}|_i$, and the closest point is quickly searched for.

The cloud ${}^{\mathcal{W}}\mathbb{A}$ could also be built by dividing the 3D space in voxels, marking each voxel as occupied or free as new points are added. But as the number of voxels increases with the volume of the covered space, it is difficult to cover a large volume.



(a) (b)

Figure 4.16: An example of an aggregated point cloud

Besides that, many voxels are empty, outside and inside the 3D surface visible on the scene.

With a moving camera with a fast frame rate, the same 3D point in the world may be imaged by a relatively large number of frames, being represented repeatedly in various point clouds in the set $\{{}^{\mathcal{W}}\mathbb{P}|_1 \cdots {}^{\mathcal{W}}\mathbb{P}|_B \cdots {}^{\mathcal{W}}\mathbb{P}|_n\}$. Therefore, for each point $P \in {}^{\mathcal{W}}\mathbb{P}|_i$ the hash table bin corresponding to $\boldsymbol{P}$ is searched for a point $\boldsymbol{Q} \in {}^{W}\mathbb{A}$ such that $\boldsymbol{Q}$ is closer than d_{min} to $\boldsymbol{P}$. If such $\boldsymbol{Q}$ is found, $\boldsymbol{P}$ is discarded, but a vote counter associated with $\boldsymbol{Q}$ is incremented.

In such a way, the 3D points in ${}^{\mathcal{W}}\mathbb{A}$ receive "*votes*" from each individual point cloud. For every 3D point in ${}^{\mathcal{W}}\mathbb{A}$, the software keeps a vote counter, and the time index i of the point cloud that last voted for it. A 3D point in ${}^{W}\mathbb{A}$ is only allowed to receive one "vote" at each time step i.

The most voted 3D points in ${}^{\mathcal{W}}\mathbb{A}$ correspond to 3D positions that were imaged and considered occupied by various successive stereo frames, and therefore may be considered as occupied with more confidence. The 3D points in ${}^{\mathcal{W}}\mathbb{A}$ with less "votes" correspond to random stereo errors to be eliminated, or to moving objects, which can also be eliminated in this way [Lobo et al., 2006], although this is not the objective here.

Algorithm 2 summarizes in pseudo-code the process of filtering redundant 3D points, where $dist()$ calculates the Euclidean distance between two points and d_{min}, $vote_{min}$ are constant parameters.

Figure 4.16 shows an example of an aggregated point cloud, created from 200 highly overlapping point clouds. Only 3D points with more than 35 votes were allowed to remain in the aggregated point cloud, and the hash table bins had $5[cm] \times 5[cm]$

Algorithm 2 Filtering redundant 3D points from sequences of point clouds.

```
    ${}^{\mathcal{W}}\mathbb{A} \leftarrow \emptyset$
    $cloudcount \leftarrow 0$
    define constants $d_{min}, vote_{min}$

    for all ${}^{\mathcal{W}}\mathbb{P}|_i \in \{{}^{\mathcal{W}}\mathbb{P}|_1 \cdots {}^{\mathcal{W}}\mathbb{P}|_B \cdots {}^{\mathcal{W}}\mathbb{P}|_n\}$
        $cloudcount \leftarrow cloudcount + 1$
        for all $\boldsymbol{P} \in {}^{W}\mathbb{P}|_i$
            if $\exists \boldsymbol{Q} \in {}^{\mathcal{W}}\mathbb{A} \mid dist(\boldsymbol{Q}.xyz, \boldsymbol{P}) < d_{min}$
                if $\boldsymbol{Q}.lastvote < cloudcount$
                    $\boldsymbol{Q}.vote \leftarrow \boldsymbol{Q}.vote + 1$
                    $\boldsymbol{Q}.lastvote \leftarrow cloudcount$
                end
            else
                $\boldsymbol{newP}.xyz \leftarrow [P.x\ P.y\ P.z]$
                $\boldsymbol{newP}.vote \leftarrow 1$
                $\boldsymbol{newP}.lastvote \leftarrow cloudcount$
                ${}^{\mathcal{W}}\mathbb{A} \leftarrow {}^{\mathcal{W}}\mathbb{A} \cup \{\boldsymbol{newP}\}$
            end
        end
    end
    delete $\forall \boldsymbol{P} \in {}^{\mathcal{W}}\mathbb{A}\ s.t.\ \boldsymbol{P}.vote < vote_{min}$
```

in a $1[m] \times 1[m] \times 1[m]$ covered area - therefore each hash table bin contained about 1/400 of the points in the point cloud. The threshold d_{min} was set to $1[cm]$. The point cloud looks sparse when plotted with a small point size (figure 4.16(a)), the gaps are filled when plotting with a larger point size, as in figure 4.16(b).

4.4.6 Results

Results with Stereo Cameras

Outdoor image sequences were used in the next experiments. Given one reference frame $\{\mathcal{R}\}|_B$, corresponding to an image arbitrarily chosen as the reference image, *SIFT* correspondences were used to match each image with the reference image, finding a set of translation vectors to translate the point clouds into the $\{\mathcal{R}\}|_B$ frame.

This set of translation vectors contains outliers, due to mismatched *SIFTs* and wrong stereo disparities, that are detected by the same RANSAC procedure.

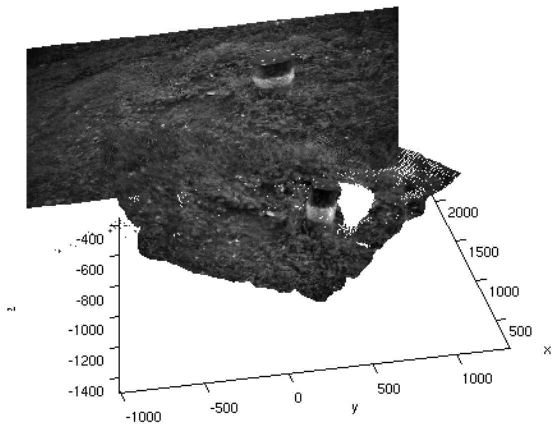
As the model utilized on the RANSAC calculations is very simple, involving averaging and calculating differences between Euclidean 3D vectors, the RANSAC procedure runs very fast, on the order of a few tenths of seconds per frame in MATLAB®.

As many backward and forward frames are matched as it is possible, by keeping trying to match frames in the sequence until the number of inliers correspondences found by RANSAC falls below a threshold - probably because the camera has moved and the overlap region is too small.

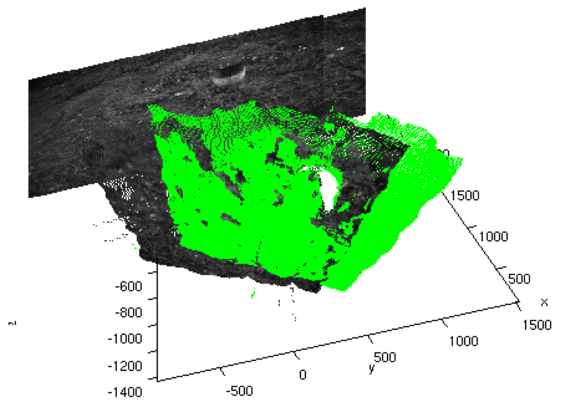
The garden sequence was taken on an outdoor grass field, under indirect sunlight. There is a small light post. Figure 4.17(a) shows one point cloud from this data set, with its corresponding left camera image displayed on the back. Figure 4.17(b) shows two registered point clouds, one in green, the other in the original gray level color. Figure 4.18 shows the resulting point cloud after registering together a sequence of 26 successive frames, with points seen in less than 4 frames discarded.

The jeep sequence shows a jeep parked on the street under sunlight. Analogously, figures 4.19(a) and 4.21(a) show one point cloud from this data set, with its corresponding left camera image displayed on the back. Figures 4.19(b) and 4.21(b) show two registered point clouds, one in green, the other in the original gray level color, with their respective images displayed behind them. Figures 4.20 and 4.22 show, in the left, a set of registered point clouds, and on the right, the resulting point cloud after registering together a sequence of 27 (in both figures) successive point clouds, and filtering out points with less than four votes. In the left figure, only one every four point clouds is shown, to ease visualization.

The pyramids in figures 4.18, 4.20 and 4.22 represent the camera trajectory and orientation: there is one pyramid for every four camera poses, and camera points towards the base of the pyramid. The images on the graph sides are an approximate visual reference, being stretched up to the extreme coordinates of their point clouds.

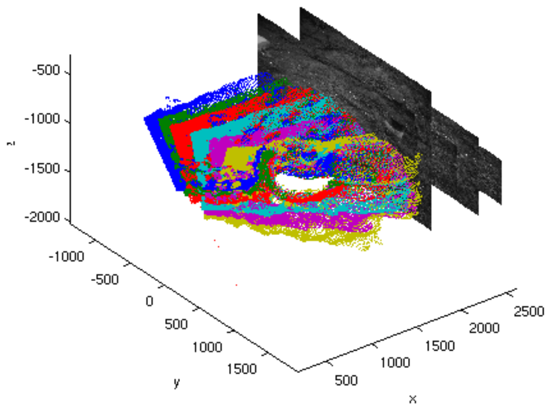


(a)

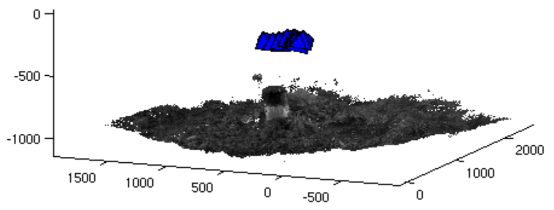


(b)

Figure 4.17: From the garden dataset: (a) one point cloud; (b) two registered point clouds.

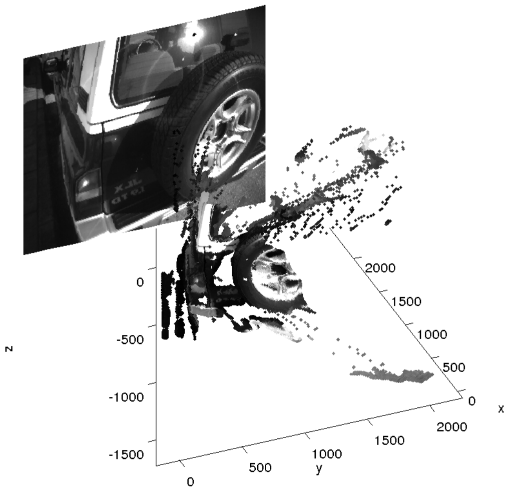


(a) Registered point clouds (only one every four).

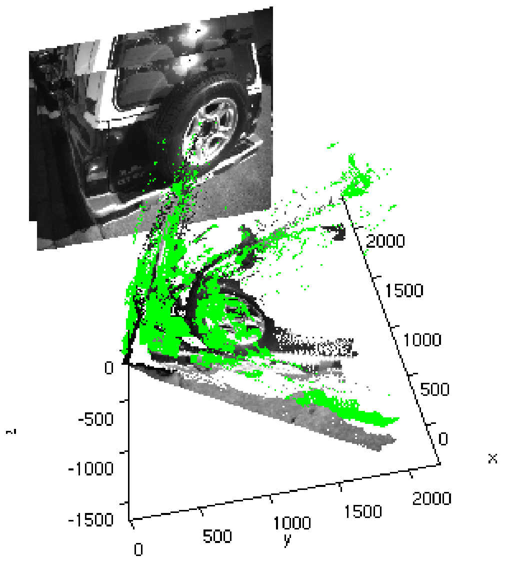


(b) the final, filtered point cloud. The camera trajectory is shown in blue.

Figure 4.18: The result of registering point clouds for 26 successive frames.

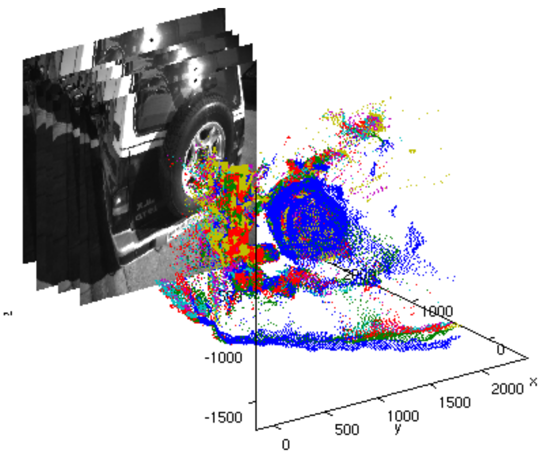


(a)

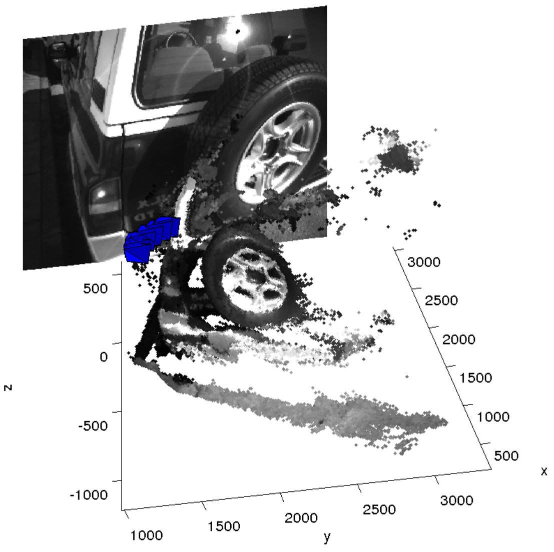


(b)

Figure 4.19: From the jeep dataset: (a) one point cloud; (b) two registered point clouds.

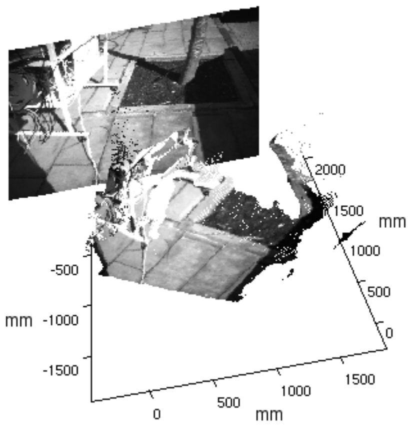


(a) Registered point clouds (only one every four)

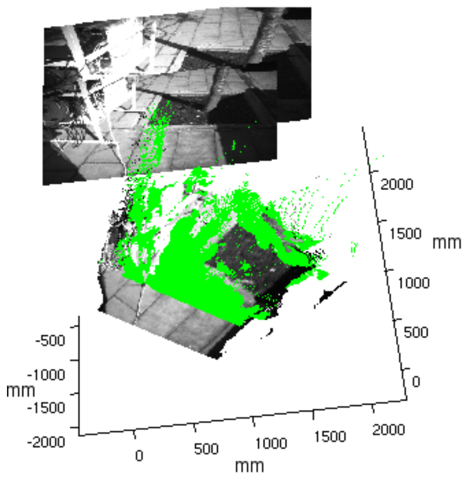


(b) The resulting, filtered point cloud. The camera trajectory is shown in blue.

Figure 4.20: The result of registering point clouds for 27 successive frames.

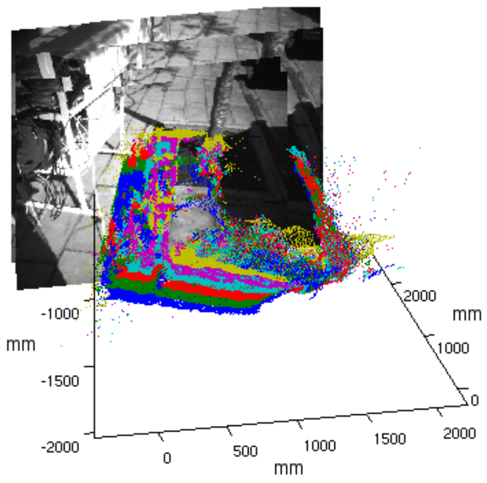


(a) One point cloud.

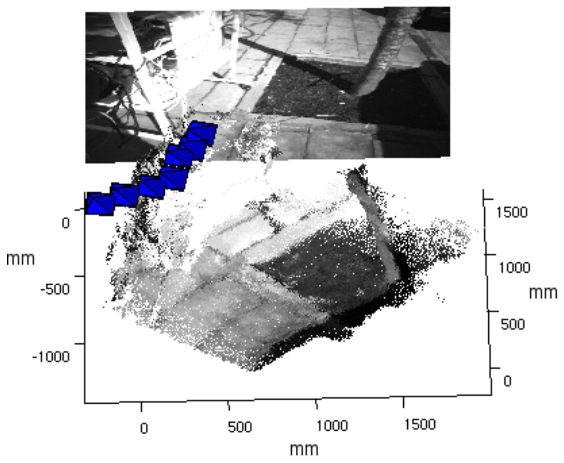


(b) Two registered point clouds.

Figure 4.21: The sidewalk dataset



(a) Registered point clouds (only one every four)



(b) The resulting, filtered point cloud. The camera trajectory is shown in blue.

Figure 4.22: The result of registering point clouds for 27 successive frames.

The RANSAC threshold for membership in the inlier set was $5[cm]$ for both datasets. The minimum acceptable number of inliers was 20.

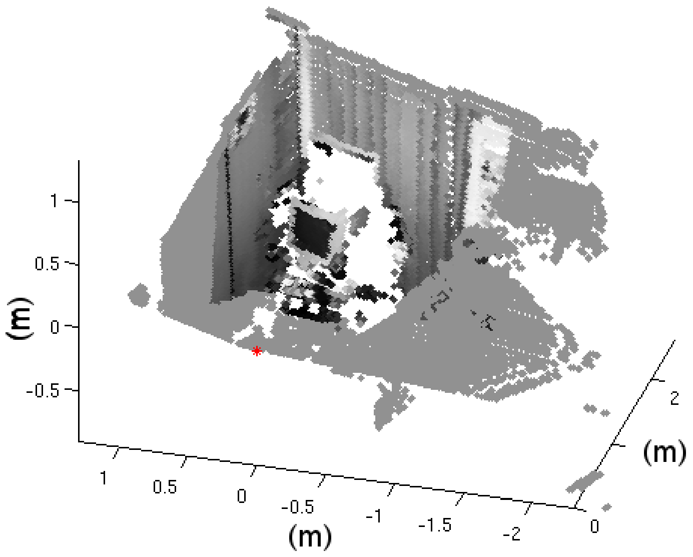
For each frame chosen as a reference frame, between 20 and 50 frames could be registered, equivalent to approximately between 1.5 and 3 seconds at $15fps$.

Results with LRF and monocular cameras

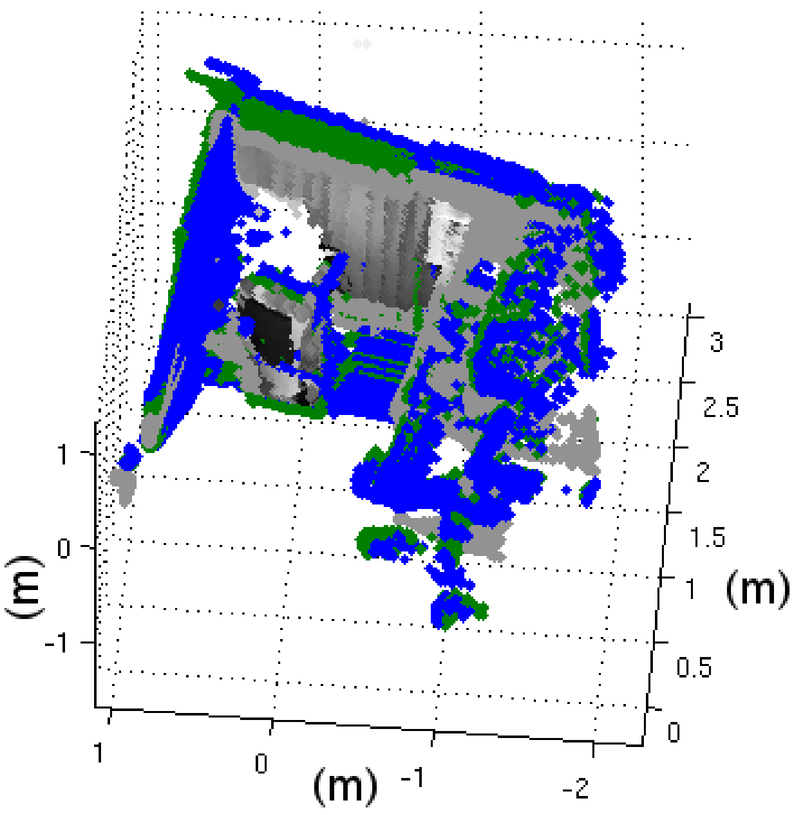
To generate a 3D point cloud, the LRF was mounted on a pantilt, and its tilt axis was moved from $-30°$ to $30°$, taking a scan every $1°$. One example is shown in figure 4.23(a). Points in the area imaged by the camera are painted with the gray color of their corresponding pixel, while points not imaged are in a uniform gray. Three such point clouds are shown registered in figure 4.23(b), with the other point clouds highlighted in green and blue.

After the process shown in this section is completed, ICP can be used to fine tune the registration. This was unnecessary with the outdoor sequence of figure 4.22. But in indoor environments, often metallic structures or electric equipment interferes on the magnetic field, and thus the AHRS compass output has larger errors. In such conditions our method can only be used as a first approximation for other techniques.

Averaging the processing times for 10 different pairs of point clouds, after the data is acquired, it takes $1.5[s]$ to generate interest points on both images, plus $1.5[s]$ to run the process described here, against $7[s]$ to execute ICP.



(a) A point cloud taken by a LRF



(b) Three registered point clouds.

Figure 4.23: Registration of three point clouds taken by the LRF.

Chapter 5

Aerial Surveillance by Visual Tracking

5.1 Introduction

As reported in chapter 3, orientation measurements from an AHRS compensated the rotational degrees of freedom of the motion of the moving camera, including in the case of the remotely controlled airship of figure 1.2. Firstly, the images were reprojected in a geo-referenced virtual horizontal plane. Pure translation models were then used to recover the camera trajectory from images of a horizontal planar area, and they were found to be especially suitable for the estimation of the height component of the trajectory.

In this chapter, the models which achieved the best results are used to recover the trajectory of the camera while it images a target independently moving in the ground plane. The target position is marked in the reprojected images. The target trajectory is then recovered and tracked using only the observations made from a moving camera, including the airship on-board camera, as it is shown in figure 5.1, and results in a urban people surveillance context with known ground truth.

GPS also can be utilized to recover the airship trajectory, but GPS position fixes are notoriously less accurate in the altitude than in the latitude and longitude axes, and this uncertainty is very significant for the low altitude dataset used in this chapter.

Uncertainty in the camera orientation estimate is the most important source of error in tracking of ground objects imaged by an airborne camera [Redding et al., 2006], and its projection in the 2D ground plane is usually anisotropic even if the original distribution is isotropic. The Unscented Transform [Julier and Uhlmann, 1997], which has been used to localize static targets on the ground [Merino et al., 2005], is thus used to project the uncertainty on the camera orientation estimate to the 2D ground plane, taking into account its anisotropic projection.

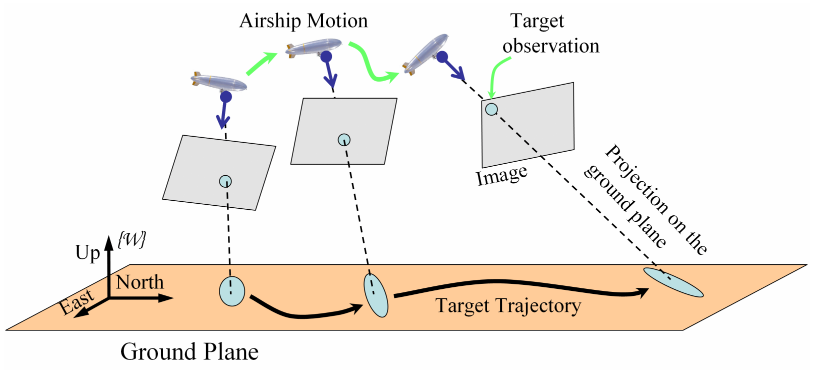


Figure 5.1: Tracking an independently moving target with observations from a moving camera.

Kalman Filters are utilized to filter the recovered trajectories of both camera and the tracked target. In the airship case, visual odometry and GPS position fixes are also fused to recover the camera trajectory.

The target trajectory is represented, tracked, and filtered in 2D coordinates. In this way the full geometry of the camera and target motion is considered and the filters involved may utilize covariances and constants set to the physical limits of the camera and target motion in actual metric units and coordinate systems. This should allow for more accurate tracking than when only pixel coordinates in the images are utilized.

5.2 Tracking of moving targets

Once the camera pose is known, a moving target is selected on each reprojected image. Problems such as image segmentation or object detection are out of the scope of this thesis. Nevertheless, to track its position on the plane, the target coordinates on the virtual image must be projected on the reference $\{\mathcal{W}\}$ frame, considering the error in the camera position and orientation. Figure 5.2 summarizes this process which is detailed in the next sections.

5.2.1 Target Pose measurement: projecting from image to world frame

The target coordinates in the image are projected into the ground plane by equation (3.4), and then these coordinates are transformed into the $\{\mathcal{W}\}$ frame by the appropriate change of axes - equation (3.1) - and translation (the origin of the $\{\mathcal{D}\}$ frame is ${}^{\mathcal{W}}\mathbf{x}_C$ in the $\{\mathcal{W}\}$ frame).

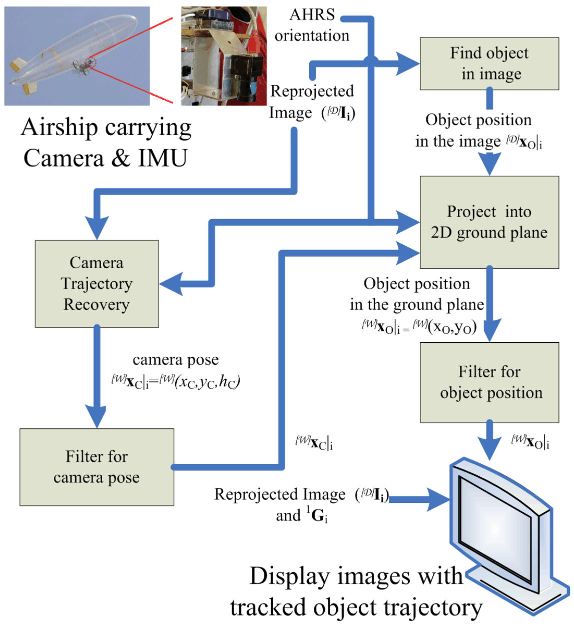

AHRS orientation
Find object in image
Reprojected Image $({}^{\{D\}}\mathbf{I}_i)$
Airship carrying Camera & IMU
Object position in the image ${}^{\{D\}}\mathbf{x}_O|_i$
Project into 2D ground plane
Camera Trajectory Recovery
Object position in the ground plane
${}^{\{W\}}\mathbf{x}_O|_i = {}^{\{W\}}(x_O,y_O)$
Filter for object position
camera pose
${}^{\{W\}}\mathbf{x}_C|_i={}^{\{W\}}(x_C,y_C,h_C)$
${}^{\{W\}}\mathbf{x}_C|_i$
${}^{\{W\}}\mathbf{x}_O|_i$
Filter for camera pose
Reprojected Image $({}^{\{D\}}\mathbf{I}_i)$ and ${}^{1}\mathbf{G}_i$
Display images with tracked object trajectory

Figure 5.2: A block diagram of the tracking process.

The projection of the images in the virtual horizontal plane does not by itself improves the measurement on the target's position on the ground, although it facilitates interest point matching (Sections 3.2.1and 4.2). Moreover, the measurement of the position of a target imaged on the ground is known to be very sensitive to errors in the camera orientation [Redding et al., 2006]. Therefore the uncertainty on camera 6D pose is propagated with the Unscented Transform, which has already been used to measure the positions of static targets from a low altitude UAV [Merino et al., 2005, Redding et al., 2006]. The actual errors in the camera position and orientation are unknown. The covariances found by the Kalman Filter of section 3.2.5 are taken as the camera pose covariance, and the camera orientation estimate is supposed to have Gaussian error with standard variation of 5[°].

Therefore, given a sequence of camera poses with the respective images and an object detected on these images, this projection generates a sequence of observations (2D coordinates and covariance) of the target pose in the ground plane .

5.2.2 Filtering of target pose

The target pose is tracked in the 2D reference frame, and filtered by a Kalman Filter to generate a smoother trajectory and filter out wrong measurements. The filter state contains the target 2D position, velocity and acceleration, with the acceleration modeled as a Wiener process [Bar-Shalom et al., 2001], as detailed in the appendix

B.5. The process error considers a maximum acceleration increment of 0.35 $[{}^{m}/{}_{s^2}]$, and the Unscented Transform supplies observations of the target position with covariance matrices which are considered as the measurement error.

The target observations projected in the ground plane have high frequency noise, due to errors in the camera position and orientation estimate, and in the target detection in each image. This is clearly seen in the trajectories of figure 5.9 where the ground truth trajectory is a sequence of straight lines. These errors are accounted for by the Unscented Transform to estimate a covariance for the target observation, but nevertheless, the original target trajectory is filtered by a low pass filter before the input of the Kalman Filter. Analyzing the spectrum of the trajectory of the walking person, most of the energy is concentrated below $1[Hz]$. As the frequencies involved are too small, a low pass filter with too large attenuation or too small cut frequency would filter out true signal features such as going from zero velocity to motion in the beginning of the movement, and introduce delays in the filtered signal after curves. Therefore after empirical testing, the low pass filter parameters were set to a cut frequency of $1[Hz]$ and attenuation of $-40[dB]$. Thus the input of the Kalman Filter is a better conditioned signal, and the final trajectory is smoother.

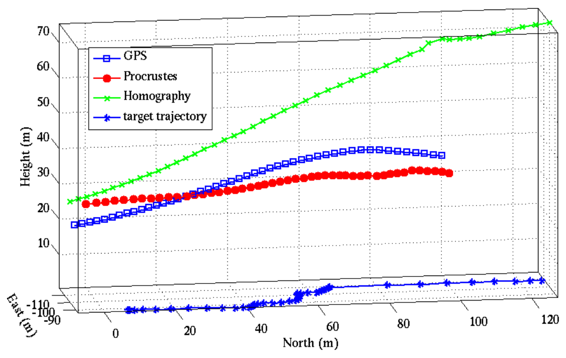
5.3 Tracking of a Moving Target from Airship Observations

Firstly, an object of known dimensions in the ground was observed by the airship, and the height of the camera estimated from its image dimensions, eliminating the scale ambiguity inherent to relative pose recovery from images alone. This was done a few seconds in the image sequence before the images shown. Then the airship trajectory was recovered by the model of section 3.2.2. Only the Procrustes procedure was necessary as the optimization did not improve the results. Finally the target (a moving car) trajectory was recovered from image observations.

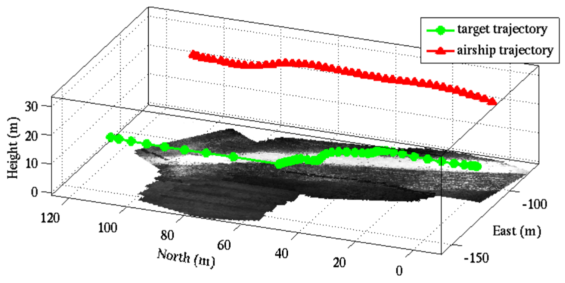
Fig. 5.3(a) shows the recovered airship trajectories using the Procrustes method of section 3.2.2 (red circles) and by the standard homography estimation and decomposition method (green crosses). The blue squares show the GPS measured trajectory. In the ground the target trajectory derived from the airship trajectory recovered by our method is shown as blue stars.

The trajectory recovered by the pure translation method is shown again in Figure 5.3(b) with the corresponding target trajectory drawn in the ground plane. The images projected in the ground plane by using equation (3.4) to find the coordinates of their corners in the ground plane and drawing the image in the canvas accordingly. One every three images is drawn.

Figure 5.4 shows the a 2D view of the target trajectory on the ground plane over



(a) Airship trajectories from GPS, pure translation and image-only method. Target trajectory derived from pure translation airship trajectory.



(b) Trajectories recovered by the pure translation method, with registered images.

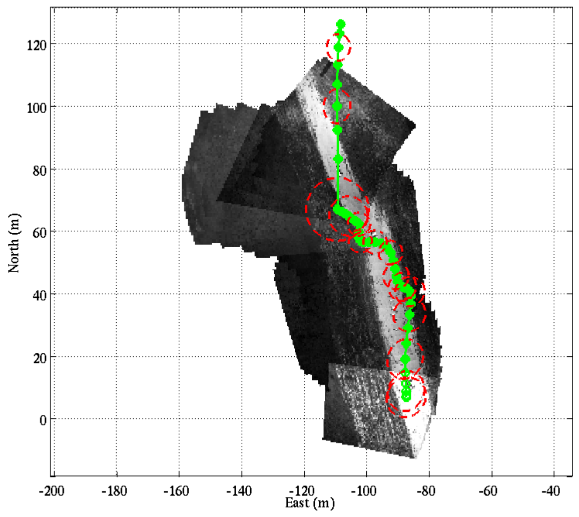
Figure 5.3: A 3D view of the recovered trajectories from Visual Odometry and from GPS, for the target and airship.

the corresponding images for the pure translation (a) and image-only (b) methods. The error in height estimation for the image only method is apparent in figure 5.4(b) as an exaggeration in the size of the last images. The same low pass and Kalman filters were used with both methods.

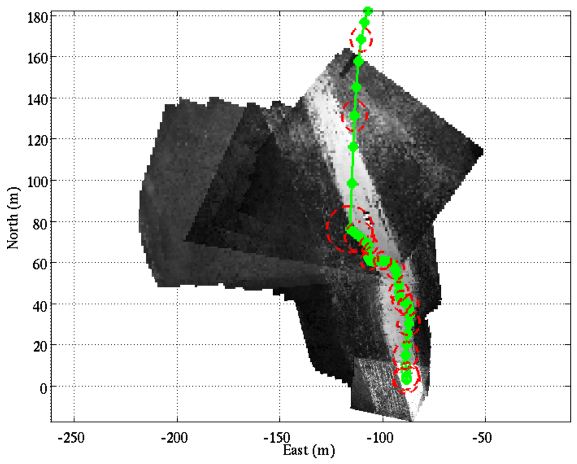
5.3.1 Tracking after fusing GPS and Visual Odometry

In this experiment the car has been driven in a closed loop in the ground while the airship was flying above it. The translation recovered by the visual odometry was fused with GPS position fixes by a Kalman Filter with a constant acceleration model [Bar-Shalom et al., 2001]. The GPS outputs standard deviation values for its position fixes (shown as the red ellipses and red points in Fig. 5.5), and the translation vectors from the visual odometry are interpreted as a velocity measurement, with a manually set covariance smaller in the vertical axis than in the horizontal ones. The GPS absolute position fixes keep the estimated airship position from diverging, while the visual odometry measurements improve the trajectory locally. The fused airship trajectory is shown as green crosses in Fig. 5.5, while the target observations are shown as blue points in the ground plane. The target could not be continously observed, therefore the straight lines (for example the straight lines crossing the path) indicate where observations were missing and resumed at some other point of the path.

Figure 5.6(a) shows the target trajectory drawn over a satellite image of the flight area. The airship trajectory was taken directly from GPS. Figure 5.6(b) shows the same target trajectory obtained when the airship trajectory is recovered by a Kalman Filter fusing both visual odometry and GPS. In both figures, the squares show the

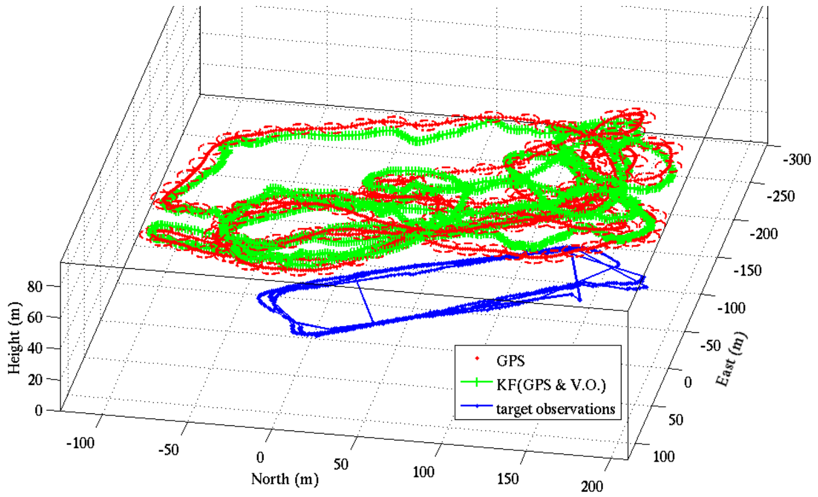


(a) Pure translation method.



(b) Image only method.

Figure 5.4: Tracking a car from the airship with the pure translation and the image only methods. The green circles are the tracked target trajectory with one standard deviation ellipses drawn in red.



(a)

Figure 5.5: The airship trajectory from GPS and from the fusion of GPS and visual odometry, with the target observations shown in the ground plane.

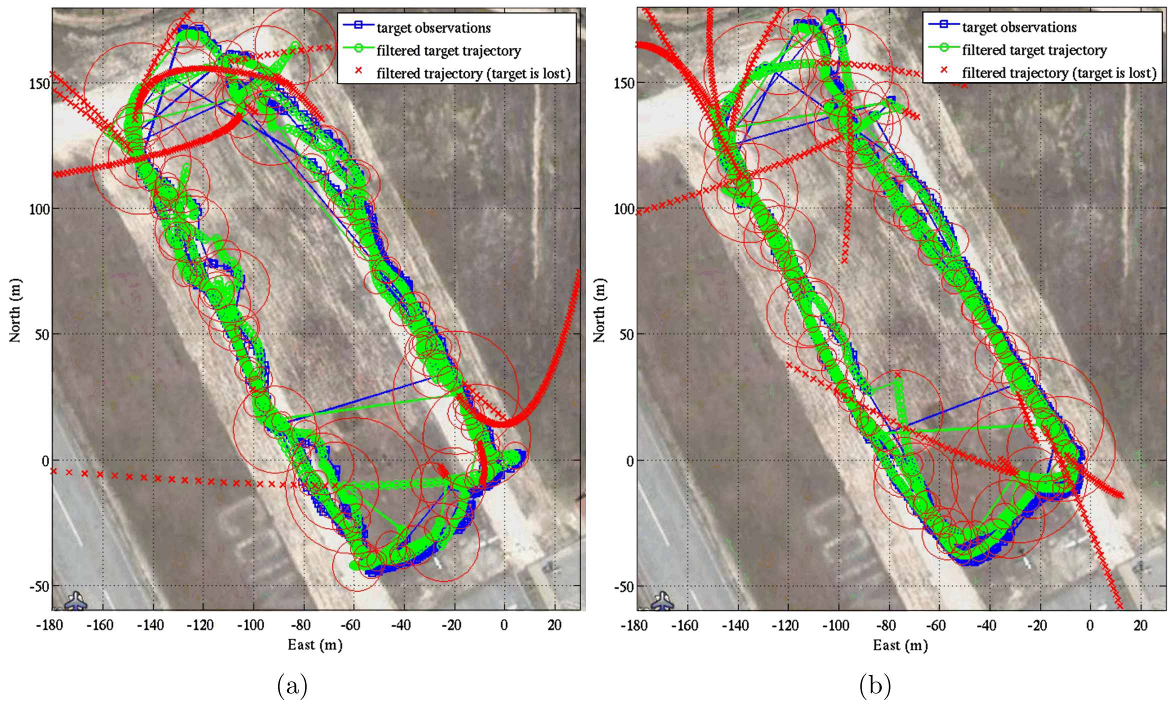
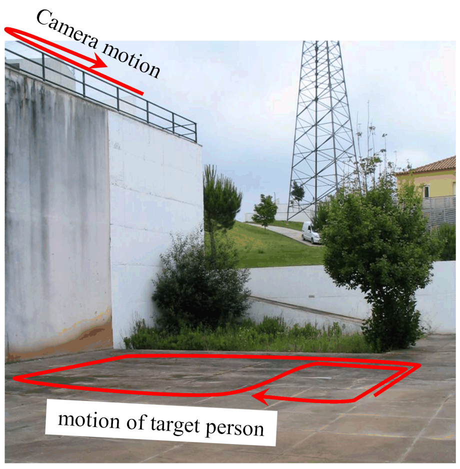


Figure 5.6: The target trajectory over a satellite image of the flight area. The car followed the dirty roads. In (a), the airship trajectory was taken from GPS, in (b) a Kalman Filter estimated the airship trajectory by fusing GPS and visual odometry.

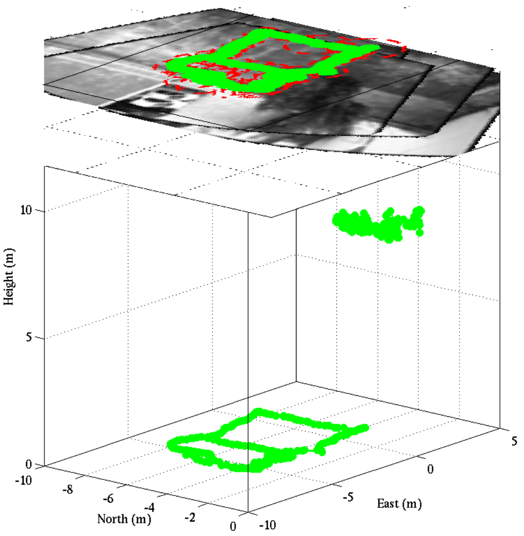
coordinates of the target observations in the ground plane, the circles show the target trajectory filtered by its own Kalman Filter, and the crosses indicate that the target is "lost". The blimp can not keep observing the target continuously, thus when there is not observations for an extended period of time the tracked trajectory diverges. If the target position standard deviation becomes larger than 30 $[m]$ then the target is declared "lost" and the filter is reinitialized at the next valid observation. Fusing the visual odometry with GPS resulted in a smoother trajectory for the tracked target.

5.4 Tracking people with a moving surveillance camera

The method described in this chapter was applied to track a person moving on a planar yard, imaged by a highly placed camera which is moved by hand. The large squares in the floor provide a ground truth measure, as the person was asked to walk only on the lines between squares. The ground truth trajectories of the camera and the target person are highlighted in the photograph of figure 5.7(a), and figure 5.7(b)



(a)



(b)

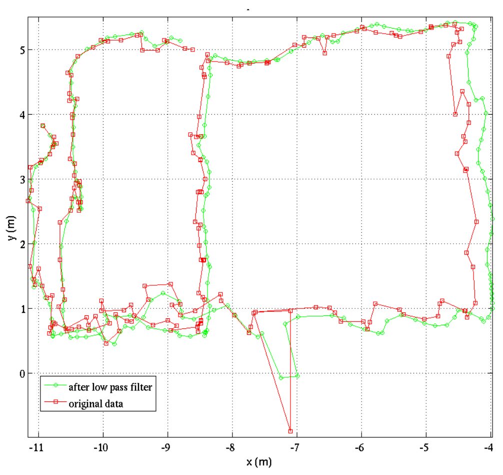
Figure 5.7: A photo with highlighted trajectories of camera and target person (a). A 3D view of the recovered trajectories, using Procrustes and optimizing in $\mathbf{t}$ to recover the camera trajectory (b).

shows a 3D view of the recovered trajectories with the registered images above it. The camera height above the ground was around $8.6[m]$, and each floor square measures $1.2[m]$.

This experiment used the MTB-9 AHRS. The method used to recover the camera trajectory shown in figure 5.7(b) was Procrustes registration followed by the optimization in the translation vector which improved the results in this case. Figure 5.8 shows the target observations projected in the ground plane before (squares) and after (circles) applying a low pass filter to the data.

Figure 5.9(a) shows a closer 2D view of the target trajectory to be compared with figure 5.9(b). In the latter case, the camera trajectory was recovered by the homography model. The red ellipses are 1 standard deviation ellipses for the covariance of the target position as estimated by the Kalman Filter. In both figures, the large covariances in the bottom right of the image appear because the target was out of the camera field of view for a number of frames, and therefore its estimated position covariance grew with the Kalman Filter prediction stage. When the target went back in the camera field of view the tracking resumed.

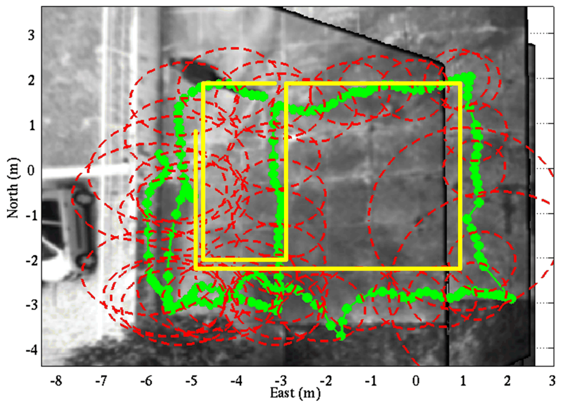
The solid yellow lines are the known ground truth, marked directly over the floor square tiles in the image. Comparing the shape of the tracked trajectories is more significant than just the absolute difference to the ground truth, as the images themselves have also some error in registration. The tracked trajectory after recovering



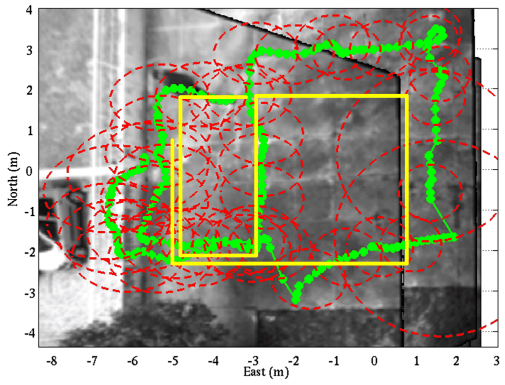
(a) Camera trajectory recovered by the homography model.

Figure 5.8: A low pass filter is applied to the observed target trajectory.

the camera trajectory with the pure translation model appears more accurate than when the homography model is used.



(a) Camera trajectory recovered by Procrustes and opt(**t**)



(b) Camera trajectory recovered by the homography model.

Figure 5.9: A closer view of the target person tracked trajectory.

Chapter 6

Conclusions

In this thesis, pure translation models are used to recover an airship UAV trajectory using AHRS orientation estimates and aerial images of the horizontal ground plane. The sets of points projected in the ground plane may be directly registered. Theoretically, this is an instance of the well-known Procrustes problem [Gower and Dijksterhuis, 2004], with closed-form solution, but with real datasets sometimes optimization approaches may improve the results, for example taking the translation vector itself as the optimization variable. The other alternative, the homology model, is derived from projective geometry, being the special case of the homography transformation under pure translation.

These models were also compared with the usual homography model against ground truth, by determining relative heights in the tripod experiments and recovering the camera trajectory in the overpass experiment. The screwdriver table experiment showed that horizontal and vertical displacements can also be recovered with the homology model, even if the FOE is very close to the camera principal point or very large, tending to infinite as the trajectory is parallel to the image plane.

The experiments with aerial image datasets show that the optimization proposed appears to be unnecessary with the more accurate AHRS, as faster non-iterative solutions are obtained by directly registering the correspondences, and the optimization provides very small or no improvement in the results. Optimization is also not necessary in the tripod and overpass experiment where the camera is static or moves only slowly.

Nevertheless, with the less accurate AHRS the best visual odometry results for the airship were obtained by a projective model (FOE estimation and planar homologies) which required an optimization step. Future work may explore the limit conditions where the Procrustes registration fails and other approaches are necessary.

The vehicle poses recovered by the visual odometry were compared with the vehicle poses estimated by GPS by generating maps of the ground plane, and the visual odometry results in locally more coherent image alignment. GPS and visual odometry data were also fused by a Kalman Filter, and the map generated with the fused trajectory is more accurate than the map generated with the GPS alone, even if the trajectory is too large to be recovered by visual odometry alone.

Over-parameterization is usually recommended in the case of homography estimation (the nine elements of the matrix are estimated although it has only eight degrees of freedom [Hartley and Zisserman, 2000]). In the pure translation case, over-parameterization with homogeneous variables was also found to be necessary.

The pure translation models have performed better than the image-only approach in the recovery of the vertical motion component. Vertical motion is more critical because errors in the height estimation propagate not only in the vertical component, but also as an error in the scale of the horizontal components. The effect of error accumulation in vertical motion is visible in figure 3.21: the recovered trajectories have a relatively correct shape but a wrong scale.

Altitude estimation is important for aerial vehicles, and GPS altitude measurements are less accurate in the altitude axis than in the horizontal axes. During landing and taking off, GPS uncertainty may be very significant due to the small height, and the restriction to horizontal ground planes is likely to be valid. Also in high altitudes the ground plane can often be safely assumed as horizontal. Moreover, reasonable pose estimates were obtained even with the orientation estimates directly output by the relatively inexpensive and inaccurate AHRS utilized, and under relatively large roll and pitch variations.

Certainly, even under pure translation motion, image-based measurements are not sufficient to find the correct scale of the translation. Scale ambiguity is inherent to relative pose retrieval from images, even when using homographies [Merino et al., 2006]. However, as image-based visual odometry may be integrated with other measurements [Kelly et al., 2007], the fact that in the pure translation model rotational, vertical, and horizontal motion are explicitly separated should facilitate the data fusion needed to recover the actual scale and improve the overall relative pose estimate. For example, other sensors which measure the direction of movement, even if relatively inaccurate, could contribute to the initial estimate before the final optimization.

The computer vision community has sporadically used planar homologies for at least ten years, but not as widely as homographies. To our knowledge the pure translation models have never been used successfully for trajectory recovery from a real aerial image sequence. Possible reasons include the relative difficulty and expense of obtaining good orientation estimates until some years ago, or even lack of communication between the robotics and computer vision communities.

Another reason may be that the estimation of the homology with unknown axis, vector and cross ratio, empirically appears to be less reliable and more vulnerable to convergence into local minima than the homography estimation and decomposition process which has already become usual in the computer vision community. The further simplification of supposing the ground plane horizontal and therefore using the virtual camera with an image plane parallel to it, improves much the reliability of the estimation.

Future work can explore more elaborate error models, e.g., estimating an uncertainty for each pixel coordinate by propagating the uncertainty in the camera 6D pose. Procrustes problem variants with diverse uncertainty models have been solved, although for some cases there are only non-iterative solutions [Gower and Dijksterhuis, 2004]. The FOE estimation algorithm could be adapted by taking into account these uncertainties when calculating epipolar distances.

Other constraints between the recovered vehicle poses could be extracted from the map formed by the reprojected images by matching other image pairs, thus a SLAM method could improve both the map and the vehicle poses. Such method must consider that when one vehicle pose is updated by a new constraint, the scale of all subsequent translation vectors from the visual odometry must be also updated.

The tracking experiments further exploited the improvements in camera trajectory recovery, with the same rotation compensated images being used to track an independently moving target in the ground plane. Using the pure translation model to recover the camera trajectory resulted in a more accurate trajectory for the tracked object than using the homography model. In a low altitude setting recovering the trajectory from the images represents a significant improvement, because GPS uncertainty is very significant, particularly as uncertainty in its altitude estimate is projected as uncertainty in the tracked object position. The fusion of Visual Odometry and GPS position fixes in the airship scenario improved the recovered airship trajectory, and these improvements translate in a smoother recovered trajectory for the moving target in the ground. The tracking can also be performed over an extended period of time.

In the context of urban people surveillance, the camera could be carried by a mobile robot to extend the coverage of a network of static cameras, being able to focus the surveillance by moving towards a detected occurrence. In such context visual odometry can be fused with wheel odometry or beacon-based localization systems, and the surveillance of an horizontal plane at a lower level is a likely scenario.

In the domain of 3D point clouds registration, the inertial orientation measurements also permitted to deal with the rotational and translational components separately. The compensation of the rotational component is very fast (it just applies a rotation to the points) and may accelerate considerably a posterior ICP registration, besides improving its reliability.

The second part of the method, the approximation of the translational component using the image correspondences, does not represent a large time saving if its is necessary to apply ICP afterwards to fine-tune the registration. ICP quickly approximates the remaining translation when its input are rotation-compensated point clouds. Besides that, the time necessary to calculate the translational components is around 30% to 50% of the time needed to run ICP in this case.

To decide if this technique should be applied, alone or as a first approximation to ICP, the developer should consider the characteristics of its particular sensor setup and the application requisites in terms of accuracy and reliability. We highlight the following points to consider:

1. The computational time spent in the calculation of the translational component is dominated by the interest point detection and matching algorithm. If interest point matching is already being done for other purposes, then the additional cost of RANSAC is very small.

2. Even with interest point matching, the total time spent by this method is considerably less than with ICP. Depending of the application, the approximation of the translational components may be good enough and the final ICP may be dismissed. For example in the stereo results of section 4.4.6, the point clouds registered using this method are visually indistinguishable from the same point clouds registered using ICP. On the other hand, when using the LRF, the point cloud is more sparse, the association of image pixels and 3D points is less accurate, and thus the results are not so accurate.

3. The approximation of the translational component represents a better starting point for ICP. In some cases, such as registration of point clouds with only partial overlap, this may allow ICP to avoid a wrong registration due to local minima. Therefore the larger computational cost may be paid off by increased reliability. In the LRF case, as the time needed to tilt the LRF to collect a 3D scan is significant, it is reasonable to think that in any robotic application the distance traveled between two consecutive 3D scans will also be relatively large. Therefore in the LRF case it is more likely that this method would be useful to approximate the translation before applying ICP.

Bibliography

[Alves et al., 2003] Alves, J., Lobo, J., and Dias, J. (2003). Camera-Inertial Sensor modelling and alignment for Visual Navigation. In *Proceedings of the 11th International Conference on Advanced Robotics*, pages 1693–1698, Coimbra, Portugal.

[Arnspang et al., 1999] Arnspang, J., Henriksen, K., and Bergholm, F. (1999). Relating scene depth to image ratios. In Solina, F. and Leonardis, A., editors, *Proc. of the 8th Int. Conf. on Computer Analysis of Images and Patterns (CAIP'99)*, volume 1689 of *Lecture Notes in Comp. Science*, pages 516–525, Ljubljana, Slovenia. Springer.

[Bar-Shalom et al., 2001] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation.* John Willey & Sons, Inc.

[Bay et al., 2006] Bay, H., Tuytelaars, T., and van Gool, L. (2006). SURF: Speeded Up Robust Features. In *the Ninth European Conference on Computer Vision*, Graz, Austria.

[Besl and McKay, 1992] Besl, P. and McKay, N. (1992). A method for registration of 3-d shapes [ICP]. *IEEE Pattern Analysis and Machine Intelligence*, 14(2):239–256.

[Borg and Groenen, 1997] Borg and Groenen (1997). *Modern Multidimensional Scaling, Theory and Application.* Springer Verlag.

[Bouguet, 2006] Bouguet, J. (2006). Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

[Bronshtein and Semendyayev, 1997] Bronshtein, I. N. and Semendyayev, K. A. (1997). *Handbook of mathematics (3rd ed.).* Springer-Verlag, London, UK.

[Brown and Sullivan, 2002] Brown, A. and Sullivan, D. (2002). Precision kinematic alignment using a low-cost GPS/INS system. In *ION GPS*, Portland, OR, USA.

[Brown, 2006] Brown, M. (2006). Autostitch: Automated panorama creation. http://www.cs.ubc.ca/ mbrown/autostitch/autostitch.html.

[Brown and Lowe, 2003] Brown, M. and Lowe, D. G. (2003). Recognising panoramas. In *10th Int. Conf. on Computer Vision (ICCV)*, Nice, France.

[Caballero et al., 2006] Caballero, F., Merino, L., Ferruz, J., and Ollero, A. (2006). Improving vision-based planar motion estimation for unmanned aerial vehicles through online mosaicing. In *IEEE Int. Conf. on Robotics and Automation (ICRA06)*, pages 2860–2865, Orlando, FL, USA.

[Capel et al., 2006] Capel, D., Fitzgibbon, A., Kovesi, P., Werner, T., Wexler, Y., and Zisserman, A. (2006). Matlab functions for multiple view geometry. http://www.robots.ox.ac.uk/∼vgg/hzbook/code/.

[Capel, 2001] Capel, D. P. (2001). *Image Mosaicing and Super-resolution*. PhD thesis, University of Oxford, Oxford, UK.

[Chen et al., 2003] Chen, Z., Pears, N., McDermid, J., and Heseltine, T. (2003). Epipole estimation under pure camera translation. In Sun, C., Talbot, H., Ourselin, S., and Adriaansen, T., editors, *Proc. of the 7th Int. Conf. on Digital Image Computing: Techniques and Applications, DICTA 2003*, pages 849–858, Sydney, Australia. CSIRO Publishing.

[Cheng et al., 2006] Cheng, Y., Maimone, M. W., and Matthies, L. (2006). Visual odometry on the mars exploration rovers. *IEEE Robotics and Automation Magazine*, 13(2):54–62.

[Corke et al., 2001] Corke, P., Sikka, P., and Roberts, J. M. (2001). Height estimation for an autonomous helicopter. In *ISER '00: Experimental Robotics VII*, pages 101–110, London, UK. Springer-Verlag.

[Cunningham et al., 2006] Cunningham, D., Grebby, S., Tansey, K., and Gosar, A. (2006). Application of airborne LiDAR to mapping seismogenic faults in forested mountainous terrain, southeastern Alps, Slovenia. *Geophysical Research Letters*, 33.

[de Paiva et al., 2006] de Paiva, E. C., Azinheira, J. R., Ramos, J. J. G., Moutinho, A., and Bueno, S. S. (2006). Project AURORA: Infrastructure and flight control experiments for a robotic airship. *Journal of Field Robotics*, 23(3-4):201–222.

[Diel, 2005] Diel, D. D. (2005). Stochastic constraints for vision-aided inertial navigation. Master's thesis, Dept. of Mechanical Engineering - Massachussetts Institute of Technology, Cambridge, MA, USA.

[Eden et al., 2006] Eden, A., Uyttendaele, M., and Szeliski, R. (2006). Seamless image stitching of scenes with large motions and exposure differences. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2498–2505, Washington, DC, USA. IEEE Computer Society.

[Eustice, 2005] Eustice, R. M. (2005). *Large-Area Visually Augmented Navigation for Autonomous Underwater Vehicles.* PhD thesis, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution - Joint Program in Applied Ocean Science & Engineering.

[Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24:381–395.

[Garmin Int. Inc., 2007] Garmin Int. Inc. (2007). www.garmin.com.

[Gower and Dijksterhuis, 2004] Gower, J. C. and Dijksterhuis, G. B. (2004). *Procrustes Problems.* Oxford Statistical Science Series. Oxford University Press.

[Gracias, 2002] Gracias, N. R. E. (2002). *Mosaic-based Visual Navigation for Autonomous Underwater Vehicles.* PhD thesis, Universidade Técnica de Lisboa - Instituto Superior Técnico, Lisbon, Portugal.

[Hartley and Zisserman, 2000] Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision.* Cambridge University Press, Cambridge, UK.

[Hirschmüller et al., 2002] Hirschmüller, H., Innocent, P. R., and Garibaldi, J. (2002). Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics. In *7th International Conference on Control, Automation, Robotics and Vision*, Singapore.

[Horn, 1987] Horn, B. (1987). Closed-Form Solution of Absolute Orientation Using Unit Quaternions. *Journal of the Optical Society of America*, 4(4):629–462.

[Huguet et al., 2003] Huguet, A. B., Carceroni, R. L., and de A. Araújo, A. (2003). Towards automatic 3d reconstruction of urban scenes from low altitude aerial images. In *IEEE 12th Int. Conf. on Image Analysis and Processing (ICIAP'03)*, Mantova, Italy.

[Hygounenc et al., 2004] Hygounenc, E., Jung, I.-K., Soueres, P., and Lacroix, S. (2004). The Autonomous Blimp Project at LAAS/CNRS: Achievements in Flight Control and Terrain Mapping. *Int. J. of Robotics Research*, 23(4/5):473–512.

[Julier and Uhlmann, 1997] Julier, S. J. and Uhlmann, J. K. (1997). A new extension of the kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, Orlando, FL, USA.

[Kelly et al., 2007] Kelly, J., Saripalli, S., and Sukhatme, G. S. (2007). Combined visual and inertial navigation for an unmanned aerial vehicle. In *Proc. 6th Int'l Conf. Field and Service Robotics (FSR'07)*, Chamonix, France.

[Kim, 2004] Kim, J. (2004). *Autonomous Navigation for Airborne Applications*. PhD thesis, The University of Sydney, Sydney, NSW, Australia.

[Kleiner et al., 2005] Kleiner, A., Steder, B., Dornhege, C., Hoefler, D., Meyer-Delius, D., Prediger, J., Stueckler, J., Glogowski, K., Thurner, M., Luber, M., Schnell, M., Kuemmerle, R., Burk, T., Braeuer, T., and Nebel, B. (2005). RoboCupRescue - RescueRobots Freiburg (Germany), Team Description Paper. In *Rescue Robot League*, Osaka, Japan.

[Konolige, 1997] Konolige, K. (1997). Small vision systems: Hardware and implementation. In *Eight International Symposium on Robotics Research*, Hayama, Japan.

[Liang et al., 2004] Liang, B., Pears, N., and Chen, Z. (2004). Affine height landscapes for monocular mobile robot obstacle avoidance. In Groen, F., Amato, N., Bonarini, A., Yoshida, E., and Kröse, B., editors, *Intelligent Autonomous Systems 8*, pages 863–872, Amsterdam, The Netherlands. IOS Press.

[Lobo and Dias, 2003] Lobo, J. and Dias, J. (2003). Vision and inertial sensor cooperation using gravity as a vertical reference. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(12):1597–1608.

[Lobo and Dias, 2005] Lobo, J. and Dias, J. (2005). Relative pose calibration between visual and inertial sensors. In *ICRA Workshop on Integration of Vision and Inertial Sensors - 2nd InerVis*, Barcelona, Spain.

[Lobo and Dias, 2007] Lobo, J. and Dias, J. (2007). Relative pose calibration between visual and inertial sensors. *International Journal of Robotics Research*, 26(6):561–575.

[Lobo et al., 2006] Lobo, J., Ferreira, J. F., and Dias, J. (2006). Bioinspired visuo-vestibular artificial perception system for independent motion segmentation. In *ICVW06 (2nd Int. Cognitive Vision Workshop)*, Graz, Austria.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[Ma et al., 2004] Ma, Y., Soatto, S., Kosecka, J., and Sastry, S. (2004). *An Invitation to 3D Vision*. Springer.

[Makadia and Daniilidis, 2005] Makadia, A. and Daniilidis, K. (2005). Correspondenceless ego-motion estimation using an imu. In *Proceedings of the IEEE International Conferenece on Robotics and Automation*.

[Malis, 1998] Malis, E. (1998). *Contributions à la modélisation et à la commande en asservissement visuel*. PhD thesis, L'Université de Rennes, École de Informatique, Traitement du Signal et Telecommunications, Rennes, France.

[Malis et al., 1999] Malis, E., Chaumette, F., and Boudet, S. (1999). 2-$\frac{1}{2}$-D Visual Servoing . *IEEE Trans. on Robotics and Automation*, 15(2):238–250.

[Matthies and Shafer, 1987] Matthies, L. and Shafer, S. (1987). Error modeling in stereo navigation. *IEEE J. of Robotics and Automation*, RA-3(3).

[Merino et al., 2005] Merino, L., Caballero, F., de Dios, J., and Ollero, A. (2005). Cooperative fire detection using unmanned aerial vehicles. In *IEEE International Conference on Robotics and Automation*, pages 1896–1901, Barcelona, Spain.

[Merino et al., 2006] Merino, L., Wiklund, J., Caballero, F., Moe, A., de Dios, J. R. M., Forssen, P.-E., Nordberg, K., and Ollero, A. (2006). Vision-based multi-UAV position estimation. *Robotics & Automation Magazine, IEEE*, 13(3):53–62.

[Michaelsen et al., 2004] Michaelsen, E., Kirchhof, M., and Stilla, U. (2004). Sensor pose inference from airborne videos by decomposing homography estimates. In *XXth ISPRS Congress*, Istambul, Turkey. The International Society for Photogrammetry and Remote Sensing.

[Mikolajczyk and Schmid, 2004] Mikolajczyk, K. and Schmid, C. (2004). Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1).

[Mikolajczyk et al., 2005] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and van Gool, L. (2005). A comparison of affine region detectors. *International Journal of Computer Vision*, 65(7):43 – 72.

[Mirisola, 2001] Mirisola, L. G. B. (2001). Desenvolvimento da estação de terra do projeto AURORA. Master's thesis, Universidade Estadual de Campinas, Instituto de Computação, Campinas, SP, Brazil.

[Mirisola and Dias, 2008] Mirisola, L. G. B. and Dias, J. (2008). Complementary results and data to Exploiting inertial sensing in vision based navigation. http://paloma.isr.uc.pt/∼lgm/.

[Mirisola and Dias, 2007] Mirisola, L. G. B. and Dias, J. M. M. (2007). Exploiting inertial sensing in mosaicing and visual navigation. In *6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV07)*, Toulouse, France.

[Mirisola et al., 2006] Mirisola, L. G. B., Lobo, J., and Dias, J. (2006). Stereo vision 3D map registration for airships using vision-inertial sensing. In *The 12th IASTED Int. Conf. on Robotics and Applications (RA 2006)*, Honolulu, HI, USA.

[Ohno and Tadokoro, 2005] Ohno, K. and Tadokoro, S. (2005). Dense 3D map building based on LRF data and color image fusion. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS 2005)*, pages 2792– 2797.

[OMG, 2007] OMG, O. M. G. (2007). OMG's CORBA page. http://www.corba.org.

[Point Grey Inc., 2007] Point Grey Inc. (2007). www.ptgrey.com.

[Redding et al., 2006] Redding, J., McLain, T., Beard, R., and Taylor, C. (2006). Vision-based target localization from a fixed-wing miniature air vehicle. In *American Control Conference*, Minneapolis, MN, USA.

[Saripalli et al., 2003] Saripalli, S., Montgomery, J., and Sukhatme, G. (2003). Visually-guided landing of an unmanned aerial vehicle. *IEEE Transactions on Robotics and Automation*, 19(3):371–381.

[Scaramuzza et al., 2007] Scaramuzza, D., Harati, A., and Siegwart, R. (2007). Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. In *IEEE Int. Conf.on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA.

[Schiele and Crowley, 1994] Schiele, B. and Crowley, J. L. (1994). A comparison of position estimation techniques using occupancy grids. In *Int. Conf. on Robotics and Automation (ICRA94)*, pages 1628– 1634, San Diego,CA, USA.

[Schmidt, 2007] Schmidt, D. C. (2007). TAO - The ACE ORB. http://www.cs.wustl.edu/∼schmidt/TAO.html.

[Steedly et al., 2005] Steedly, D., Pal, C., and Szeliski, R. (2005). Efficiently registering video into panoramic mosaics. In *IEEE Int. Conf. on Computer Vision (ICCV05)*, Beijing, China.

[Strelow and Singh, 2004] Strelow, D. and Singh, S. (2004). Motion estimation from image and inertial measurements. *The International Journal of Robotics Research*, 23(12):1157 – 1195.

[Suter et al., 2002] Suter, D., Hamel, T., and Mahony, R. (2002). Visual servo control using homography estimation for the stabilization of an x4-flyer. In *41st IEEE Conf. on Decision and Control*, pages 2872–2877, Las Vegas, NV, USA.

[Szeliski, 2004] Szeliski, R. (2004). Image alignment and stitching: A tutorial. Technical Report MSR-TR-2004-92, Microsoft Research.

[Szeliski, 2005] Szeliski, R. (2005). *Image alignment and stitching*, chapter 17, pages 273–292. Handbook of Mathematical Models in Computer Vision. Springer.

[Sünderhauf and Protzel, 2007] Sünderhauf, N. and Protzel, P. (2007). Stereo odometry - a review of approaches. In *IASTED Conference on Robotics and Applications*, Würzburg, Germany.

[Triebel et al., 2006] Triebel, R., Pfaff, P., and Burgard, W. (2006). Multi-level surface maps for outdoor terrain mapping and loop closing. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China.

[van Gool et al., 1998] van Gool, L., Proesmans, M., and Zisserman, A. (1998). Planar homologies for grouping and recognition. *Image and Vision Computing*, 16(21-26).

[XSens Tech., 2007] XSens Tech. (2007). www.xsens.com.

Appendix A

Experimental Platform

This chapter provides a more detailed description of the DIVA airship, complementing the overview of section 1.4. A high level view of the characteristics and capabilities of the current system is presented in section A.1. Then, section A.2 presents the requisites for an extension of this system towards automatic flight. Finally, the hardware components, sensors, and software architecture are shown in section A.3.

A.1 Technical Characteristics of the DIVA Airship

This section details the technical characteristics of the system which collected the data used on the experiments shown in chapter 3. They can be considered as the requisites to build an equivalent system. The system is composed by an embedded system and a ground station, which communicate through a wireless link.

A.1.1 Embedded System

Operational Requisites

- Periodically, the embedded system must read data from all sensors and transmit the data to the ground station, with a period compatible with the vehicle dynamics.
- The embedded system must integrate at least one camera, capturing images with sufficient resolution and frame rate. A modern camera with automatic gain adjustment is necessary as illumination conditions change often during the flight.
- The embedded system must store all telemetry data, besides transmitting them to the ground station, to avoid losing data if the data link is lost. Camera images also are stored (the heaviest burden to the CPU and storage system).

- If DGPS (Differential GPS) corrections are needed, the embedded system must receive DGPS data from the ground station via a reliable data link. GPS position fixes when DGPS data is intermittent often "jump" between corrected and uncorrected states, which may be worse than having no DGPS correction at all. A separate low speed link for DGPS data may be advisable.

- Payload weight is a severe limitation for airships. Thus, energy consumption must be minimized to decrease battery weight.

Safety Characteristics

- The embedded system must have a Remote Control (RC) receiver to allow a human pilot to manually pilot the airship. Figure A.3 shows a Radio Control Unit (standard aero-modeling equipment), used to pilot the airship.

- There must be a device electrically independent of the CPU to read the servo commands from the RC receiver, able to continue working in case of CPU malfunctioning. It receives commands from the RC receiver, and relays them to the servo motors and to the CPU to be read and stored.

- The embedded system must be sufficiently resistant to vibration and tilting to resist the flight and motor induced vibration. Vibration isolation (lightweight) may be necessary. Vibration-resistant data storage such as a flash disk is needed to store images.

A.1.2 Ground Station

This section presents the technical characteristics of the ground station.

Data Collection And Storage

- Although the embedded system stores all state variables, the data may not be recoverable due to accidents or malfunctioning. Therefore, the ground station should receive and store data from the embedded system.

- The ground station must be easily reconfigurable if there is a change in the data format sent by the embedded system (e.g., if a new sensor is added).

- The stored data must be easily converted to a format readable by commercial mathematical software such as MATLAB® (e.g., an ASCII format).

User Interface: Vehicle Safety And Monitoring The ground station must monitor the vehicle state not only to detect hazardous situations (safety issues), but also to avoid useless flights and waste of time in case of malfunctioning. The human pilot and the algorithm developers should determine which variables should be monitored.

- Monitor all critical state variables, i.e., the ones which are essential to the flight safety or data recording, like tachometers, GPS, camera status, etc. The user interface must show the state of the most critical variables clearly and continuously (but not necessarily the actual numeric value of all of them), indicating critical failures with alarms (red lights and/or sounds).
- This monitoring must be active before take-off, so that the ground station operator can abort the flight if a critical system is not operating.

Differential GPS

- If differential GPS correction is needed, the ground station should be connected to a DGPS (Differential GPS) station and transmit its signals to the embedded system.

A.2 Requisites for extension towards an autonomous airship prototype

This section details the requisites necessary to extend the current system towards automatic flight, in addition to the requisites already presented in section A.1.1. An early version of these requisites, without an on-board camera, is found on [Mirisola, 2001].

A.2.1 Embedded System

Operational Requirements

- Periodically, the embedded system must also receive from the ground station commands and parameters related to the algorithms to be executed, transmit feedback about their execution, and send commands to the actuators.
- To develop control algorithms based on inputs generated from images, the embedded system must also transmit camera images to the ground.
- If more than one camera is utilized to perform some of the tasks described in this thesis, it is desirable to have synchronized cameras, i.e., cameras which, when connected in the same data bus, can take frames at the same time.

Safety Requirements

- The human pilot must be able to take over the control at his will using a channel on his RC unit. The on-board electronics must detect this command and change immediately to manual mode.

- There must be a device electrically independent from the CPU to select between manual and automatic control, able to continue working in case of CPU malfunctioning. It should receive commands from both the embedded CPU and the RC receiver, and relay one of them to the servo motors. The commands are also relayed to the CPU to be stored.

A.2.2 Ground Station

Ground Station User Interface: Operational These requirements aim to increase the usefulness of each flight, reducing the number of flights needed to achieve the intended results.

- For image-based algorithms, the user interface must show the images being received from the embedded system.

- The embedded system may be able to execute different types of mission, with different parameters, e.g., controller gains, or camera parameters as brightness, saturation, etc. Parameter adjustments must be done online via the ground station, during the flight. It is not desirable to have to land the airship and log on the embedded system to change the a parameter value.

- The operator must be informed about the execution of the mission. Mission related information must be displayed over a map of the vehicle trajectory.

Ground Station User Interface: Vehicle Safety And Monitoring

- The user interface must have an Instruments Panel displaying the value of the most important variables (e.g. tachometers, altitude, attitude). Under automatic flight a sensor failure may represent a safety risk.

- The user interface must be able to show graphs of all state variables. They may be important to evaluate the algorithms being tested and choose new values for parameters.

- To visualize the vehicle in a virtual 3D world is not useful if the virtual world is not sufficiently detailed to provide visually recognizable references to the user. If the airship is always within visual range a virtual world visualization is also less important.

A.3 Overview of architecture

This section provides an overview of the system architecture developed and utilized to obtain the datasets used in this thesis including the aerial dataset used in chapter 3. Remotely piloted flights were performed with telemetry and image recording with the on-board hardware architecture shown in Fig. A.1.

A.3.1 Sensors and hardware components

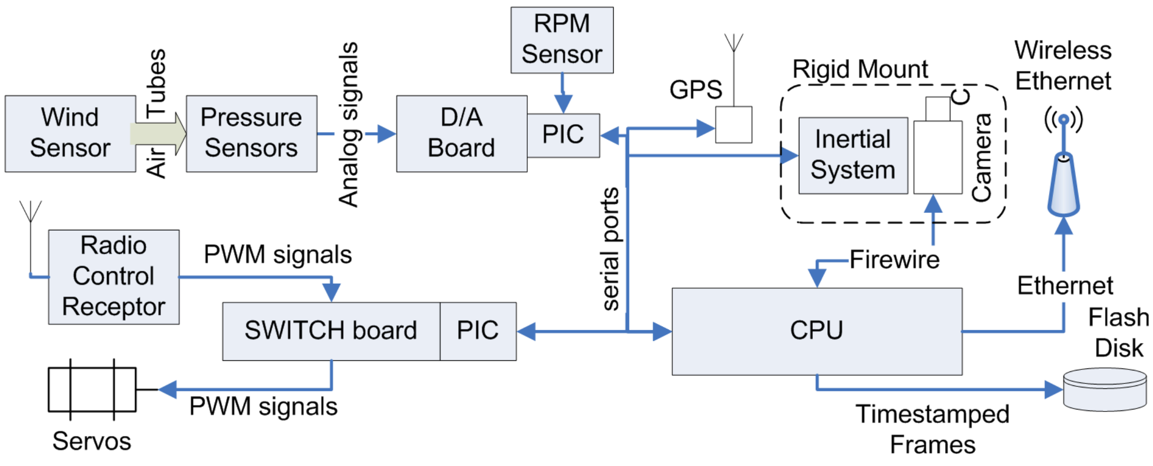
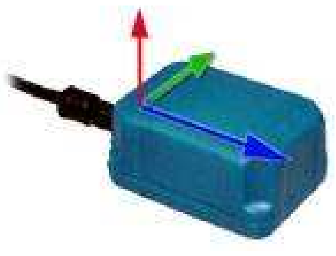


Figure A.1: The embedded system of the DIVA project.

Figure A.2 shows hardware components used in the DIVA airship, some of which are described below:

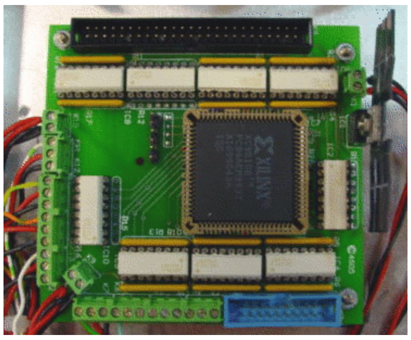
CPU The aerial dataset used in section 3.4.4 and other datasets used in chapter 4 were collected with a M570-BAB Board with a VIA EDEN 600$[MHz]$ processor. The board has the PC/104+ format, and incudes 512$[MB]$ RAM. Nevertheless, an upgrade was necessary to be able to store images at a higher frame rate. The embedded CPU was upgraded to a MiniITX CPU model VIA EN 1500G, with a 1.5$[GHz]$ processor, 6x USB, 2x RS232, Ethernet, and Firewire camera ports. It was not necessary to add any expansion to the MiniITX board. Figure A.4(a) shows a picture of the CPU board, and figure A.4(b) shows it mounted on the DIVA on-board computer.
The reasons for the upgrade are detailed in section A.3.1.1. The CPU was used to obtain the aerial dataset used in section 3.4.5 and chapter 5, with a image frame rate of 5fps.

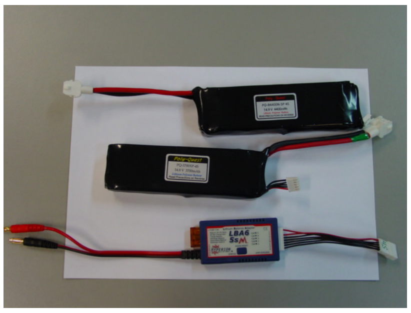

(a) Xsens MTB-9 AHRS

(b) Xsens MTi AHRS

(c) Garmin GPS35

(d) Wind sensor on the airship nose


(e) The switch board is the interface between the servos, RCU and CPU


(f) Li-Po batteries for the embedded system.

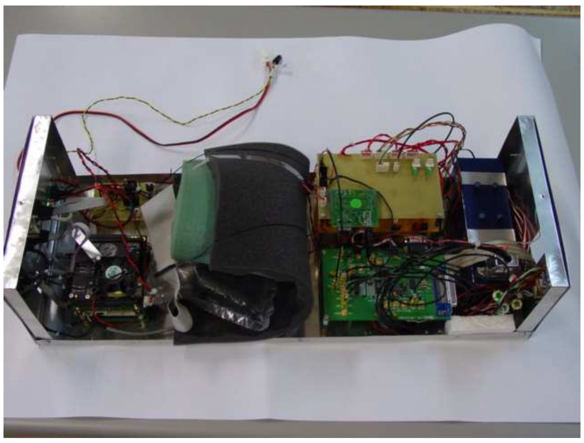

(g) The embedded system.

Figure A.2: Hardware components of the DIVA prototype.

Inertial System (Xsens MTi and MTB-9) [XSens Tech., 2007] See sections 2.3 and 1.4for details.

Digital Camera (Point Grey Flea) [Point Grey Inc., 2007]. See section 1.4 for details. This camera automatically controls parameters such as gain to adapt to illumination changes. Images were captured during flight, with no need of any parameter control external to the camera. Bayer images were converted into grayscale images by simple linear interpolation with OpenCV functions.

GPS Receptor (Garmin GPS35 12 channel GPS unit) Low weight GPS receptor and antenna integrated in a single mouse-sized package, with RS-232 communication.

Wind Sensor It is mounted on the airship nose and measures barometric altitude, the angle of attack, sideslip angle and wind speed. It is not yet calibrated.

RPM Sensor Measures the rotation speed of the motors, by counting the number of interruptions on an Infra Red light signal that is cut by the propeller movement.

Switch Board The servo-motors on the flaps and motors accelerators are commanded by Pulse Width Modulation (PWM) signals sent by the human pilot via the Radio Control Unit. The Switch Board reads the signals from the Radio Control Receiver, transmits their values in digital form (serial port) to the CPU, for recording, and retransmits PWM signals to the servo motors. In future automatic flights, the same board may switch to transmit commands sent from the CPU to the servo motors.
As the switch board is responsible to send commands to the servos in both manual and automatic modes, a failure on it would render the airship unresponsive to any command. For safety, it must be able to continue functioning even in the case of CPU failure. Therefore, it is electrically isolated from the on-board CPU and servo motors, with a separate battery for power.

Video Storage Vibration resistant storage for images is provided by a fast USB flash disk. It must be tested by saving a file large enough to fill all buffers involved in data writing (in the order of hundreds of megabytes), and then observing the continuous data transfer speed. Actual continuous writing speed is often much less than the advertised value (respectively $5^{MB}/_{s}$ and $21^{MB}/_{s}$ with our flashdisk).

Besides the on-board system, there is also support equipment on the ground (see Fig. A.3). The human pilot commands the airship with a standard aeromodel Remote Control Unit, sending PWM signals to command the servo-motors on the flaps and motor accelerators. Also, a laptop connected to a Wireless Access Point receives, stores and displays telemetry data from the airship.

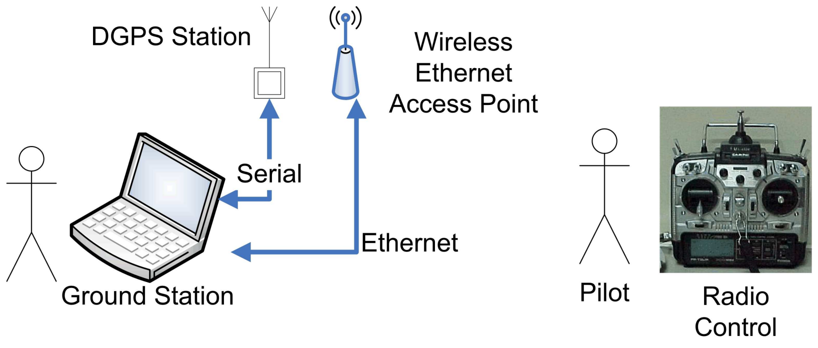


Figure A.3: The Ground Station.

Batteries and energy budget Separate batteries are needed for the on-board radio receiver, to avoid interference, and for the switch board, to keep it electrically isolated from other components. As their current consumption is quite small, these are small Ni-MH or Li-Po batteries. A separate, larger Ni-MH battery, supplies power to the servo motors.

The main concern is the battery for the on-board CPU and the sensors connected to it. The current system needs around $35[W]$ power, supplied by a Li-Po (Lithium-Polymer) battery with $3700[mAh]$ at $14.9[V]$, shown in figure A.2, which can safely supply power for more than an hour. Therefore the tightest limitation of flight time is now fuel consumption, and not battery power.

Actuators The motors accelerators are actuated by servo motors, to control the speed of the airship. In the tail, there are four aerodynamical surfaces (flaps), mounted on 'X' shape, with one servo motor deflecting each one, which are used to steer the airship. There is also a servo motor to open and close a valve to release helium from the envelope, to land quickly the airship in the case of emergency.

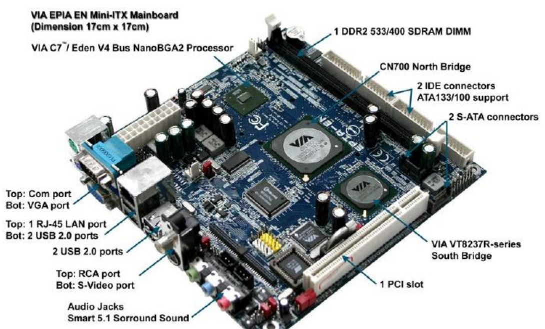
A.3.1.1 Comparison of standards PC104 and MiniITX

The DIVA project switched from PC104 to MiniITX for the following reasons, which are also summarized in table A.1:

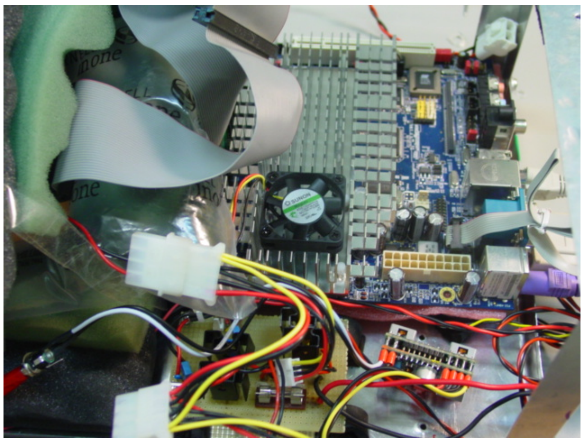
1. The PC104 standard was extended in the last years with the addition of PCI-104 and PC104+ standards, to incorporate different buses (ISA, PCI, or both). Although in theory these different boards can be stacked and connected together, this resulted in common incompatibility problems between boards of different suppliers. These problems are avoided with MiniITX.

	PC104			**MiniITX**
Board Dimensions	$11[cm] \times 10[cm]$	+	-	$17[cm] \times 17[cm]$
Board Area	$110[cm^2]$	+	-	$289[cm^2]$
availability of new tech	larger delay	-	+	more up to date
support Linux	uncertain	-	+	more common
expansibility	yes, stacked bus	+	-	very limited (1 slot)
incompatibility risks	yes	-	+	all-in-one-board
price range ($1[GHz]$ **CPU**)	€ 400-800	-	+	€ 150-280

Table A.1: Comparison between the PC104 and MiniITX standards. '+' indicates an advantage, '-' a disadvantage.



(a)



(b)

Figure A.4: The MiniITX VIA EN 1500G CPU used in the DIVA on board computer.

	weight [g]	**total weight [g]**
Gondola (mechanical structure)	6800	6800
Battery box (for airmodel equipment)	850	7650
Embedded System Box	3050	10700
GPS, CAM-AHRS, Anemometer	1300	12000
Li-Po Battery for Embedded System	450	12450
Wireless AP	150	12600
Fuel	800	13400
Envelope Safety Limit		13500

Table A.2: The weight budget of a flight experiment.

2. The larger size of MiniITX means that it is possible to fit more functionality into the board, therefore it is more likely that one single board will have all needed functionality. For DIVA, the CPU must have at least a Firewire port and enough serial or USB ports to connect the other sensors.

3. The larger weight of a MiniITX board is compensated by eliminating the extra Firewire board, and by exchanging the PC104 power source board by the smaller MiniITX DC-DC converter. Also, the weight of vertical bus connectors and screws is eliminated by having all functionality on one board.

4. A MiniITX system usually has few expansion slots available to connect expansion boards, and it can not expand by stacking new boards like a PC104 system. But future expansion needs may be covered by USB or Ethernet connections. Newer MiniITX boards have connectors for Firewire, 4x or 6x USB2.0, IDE and/or SATA, 2x RS232, 1x or 2x Ethernet. Many have developed multimedia capabilities.

5. MiniITX motherboards generally are cheaper than PC104 motherboards.

A.3.1.2 A weight budget example

Table A.2 represents the weights of all hardware equipment aboard the gondola for the last flight experiment of August 2008. The weights in this table include the corresponding cables to connect each item into the system.

A.3.2 Embedded & Ground Station Software

The C++ embedded software includes a main loop to read, store and transmit sensor data, and threads to capture and store images. The period is $100[ms]$.

CORBA middleware [OMG, 2007] is used for the transmission of a struct containing all sensor data. More specifically the free software TAO ORB [Schmidt, 2007] is used in both the embedded system and ground station. TAO is aimed at real time and embedded systems, and has been one of the main references for CORBA in these areas because it was one of the first free ORBs to implement a significant part of the standard specifications. The implementation is optimized to avoid data copying and context switching, and their results show good latency times. Although some services are called real-time, this is valid only for specific services, and TAO is not really real-time without a supported real time operational system.

The subset of CORBA called minimumCORBA was evaluated but found to be too limited. Anyway, as the embedded system do not use sophisticated services, the overhead of the normal implementation is negligible.

The usage of CORBA middleware avoids manual data marshaling and allows fast reconfiguration when the telemetry format is changed. For example, when a new sensor is added, new fields are added in the struct and the communication routines remain the same.

Images and sensor data are time-stamped immediately after reading with the same CPU clock, and each image is saved with the timestamp embedded on its filename.

The telemetry data is saved both in the embedded system and on the ground station, for redundancy. It is saved in a comma-separated text file, which may be open directly in MATLAB® or other processing software.

The main task of the ground station software is to receive a stream of data from the embedded computer, store it and show it to the user with a GUI. Additionally the ground station must receive commands from the user and send it to the embedded computer.

Therefore, there may be various GUI clients to display/store telemetry data and to send commands to the embedded system, but only one piece of software to centralize communication with the embedded computer. Here we use a CORBA server to receive telemetry data and the Event Service to distribute telemetry data among the various clients.

In this way, any new GUI client just needs to connect to the Event Service, to receive all telemetry data coming from the embedded system. The GUI client must check the data types to discriminate between telemetry and other kinds of data from the embedded system, such as acknowledgments of camera settings.

Figure A.5 shows a diagram with the main components of the Ground Station. A GUI client developed as a undergrad student project displays the 2D trajectory of the blimp over a satellite image of the area, and displays the value of some important

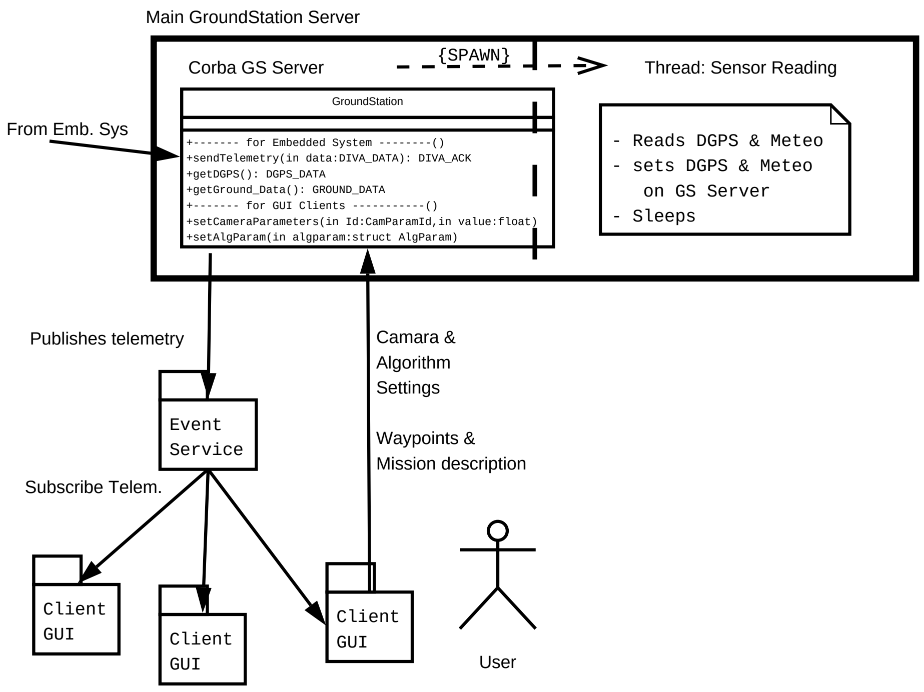

Main GroundStation Server
Corba GS Server
{SPAWN}
Thread: Sensor Reading
GroundStation
+------- for Embedded System --------()
+sendTelemetry(in data:DIVA_DATA): DIVA_ACK
+getDGPS(): DGPS_DATA
+getGround_Data(): GROUND_DATA
+------- for GUI Clients -----------()
+setCameraParameters(in Id:CamParamId,in value:float)
+setAlgParam(in algparam:struct AlgParam)
- Reads DGPS & Meteo
- sets DGPS & Meteo on GS Server
- Sleeps
From Emb. Sys
Publishes telemetry
Camara & Algorithm Settings
Waypoints & Mission description
Event Service
Subscribe Telem.
Client GUI
Client GUI
Client GUI
User

Figure A.5: Ground Station Components

state variables in real-time including a graphical instruments panel. It can "replay" a previous flight by reading a previously stored telemetry file and sending the data to the Ground Station Server via a CORBA remote method. Figure A.6 shows an image of one of the windows of this GUI with the instrument panel that was used in the flight tests.

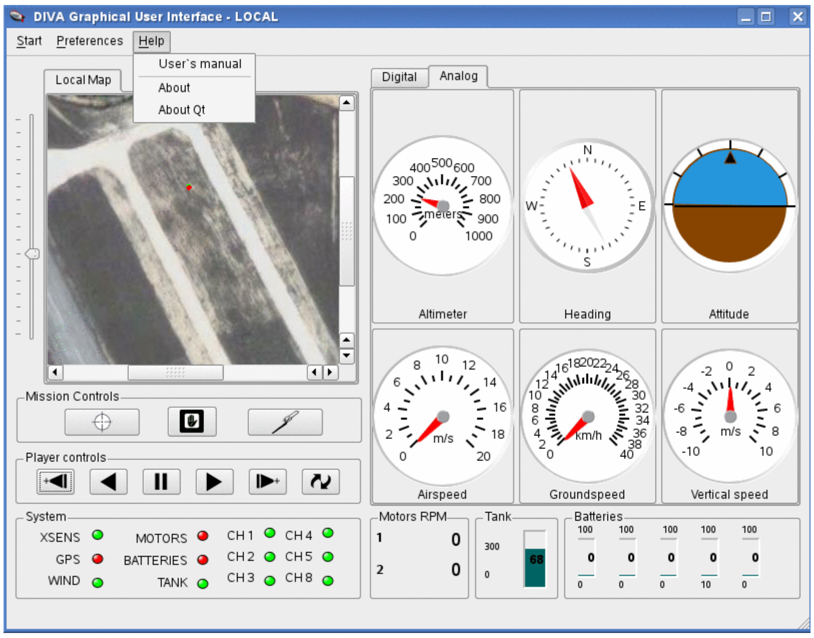


Figure A.6: Monitoring GUI at the Ground Station laptop with a instruments panel and a map of the flight area.

Appendix B

Mathematical Proofs and formulae

B.1 Recovering the scale of the recovered homography matrix

Given two images of the same 3D plane, a homography recovered from point correspondences is recovered only up to a scale factor λ (section 2.2.2), in the form $\mathbf{H}_\lambda = \lambda\mathbf{H} = \lambda\left(\mathbf{R} - \mathbf{t}\mathbf{n}^T/d\right)$. The matrix $\mathbf{R}$ is a 3×3 rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is a translation vector, $\mathbf{n} \in \mathbb{R}^3$ is the plane normal vector, and $d \in \mathbb{R}$ the distance from the camera center to the plane, as shown in figure B.1.

The matrix $\mathbf{H}_\lambda = \lambda\mathbf{H}$ is recovered only up to the scale factor $\lambda \in \mathbb{R}$ which must be recovered first. After λ is recovered, the translational and rotational components embedded in the matrix will be recovered. Therefore, the following proposition must be proved:

Proposition: For a matrix on the form $\mathbf{H}_\lambda = \lambda\left(\mathbf{R} - \mathbf{t}\mathbf{n}^T/d\right)$, $|\lambda|$ is the second largest singular value of $\mathbf{H}_\lambda$.

Proof: This proof was taken from [Ma et al., 2004], and it is more explicitly detailed here. For definitions in algebra we relied on [Bronshtein and Semendyayev, 1997]. The notation $\mathbf{x} \sim \mathbf{y}$ denotes that the vectors $\mathbf{x}$ and $\mathbf{y}$ are proportional or parallel.

Let $\mathbf{u} = \mathbf{R}^T\mathbf{t}/d \in \mathbb{R}^3$. Then it can be shown that:

$$\mathbf{H}_\lambda^T\mathbf{H}_\lambda = \lambda^2\left(\mathbf{I} + \mathbf{u}\mathbf{n}^T + \mathbf{n}\mathbf{u}^T + \|\mathbf{u}\|^2\,\mathbf{n}\mathbf{n}^T\right) \tag{B.1}$$

Denote $\mathbf{u} = (u_1, u_2, u_3) \in \mathbb{R}^3$, and

$$\tilde{\mathbf{u}} = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{B.2}$$

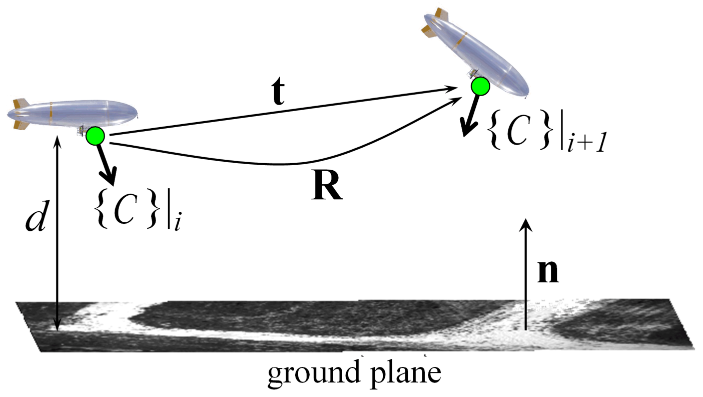


Figure B.1: A 3D plane imaged by a moving camera induces a homography.

where the matrix $\tilde{\mathbf{u}}$ is a skew-symmetric matrix. From the properties of skew symmetric matrices, we known that $\tilde{\mathbf{u}}^T = -\tilde{\mathbf{u}}$, and for any vector $\mathbf{x} \in \mathbb{R}^3$, $\mathbf{u} \times \mathbf{x} = \tilde{\mathbf{u}}\mathbf{x} \in \mathbb{R}^3$

Take the vector $\mathbf{u} \times \mathbf{n} = \tilde{\mathbf{u}}\mathbf{n} \in \mathbb{R}^3$, which is orthogonal to both $\mathbf{u}$ and $\mathbf{n}$. Now calculate the scalar product of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$ by $\tilde{\mathbf{u}}\mathbf{n}$:

$$\mathbf{H}_\lambda^T\mathbf{H}_\lambda \cdot \tilde{\mathbf{u}}\mathbf{n} \;=\; \lambda^2 \left(\mathbf{I} + \mathbf{u}\mathbf{n}^T + \mathbf{n}\mathbf{u}^T + \|\mathbf{u}\|^2 \mathbf{n}\mathbf{n}^T\right) \cdot \tilde{\mathbf{u}}\mathbf{n} = \lambda^2 \left(\tilde{\mathbf{u}}\mathbf{n}\right) \tag{B.3}$$

as, due to the orthogonality, all terms except $\lambda^2(\mathbf{I} \cdot \tilde{\mathbf{u}}\mathbf{n})$ are zero.

As $\mathbf{H}_\lambda^T\mathbf{H}_\lambda \cdot \tilde{\mathbf{u}}\mathbf{n} = \lambda^2(\tilde{\mathbf{u}}\mathbf{n})$, we know that λ^2 is an eigenvector of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$. Therefore λ^2 is an eigenvalue of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$, and, by definition, $|\lambda|$ is a singular value of $\mathbf{H}_\lambda$.

Now it must be shown that this singular value is the second largest. Let a new matrix $\mathbf{Q}$ be defined as:

$$\mathbf{Q} = \mathbf{u}\mathbf{n}^T + \mathbf{n}\mathbf{u}^T + \|\mathbf{u}\|^2 \mathbf{n}\mathbf{n}^T \tag{B.4}$$

such that:

$$\mathbf{H}_\lambda^T\mathbf{H}_\lambda = \lambda^2(\mathbf{I} + \mathbf{Q}) = \lambda^2\mathbf{I} + \lambda^2\mathbf{Q} \tag{B.5}$$

Then let $\mathbf{v} = \|\mathbf{u}\|\,\mathbf{n}$, $\mathbf{w} = \mathbf{u}/\,\|\mathbf{u}\| \in \mathbb{R}^3$. It can be shown that:

$$\mathbf{Q} = \mathbf{u}\mathbf{n}^T + \mathbf{n}\mathbf{u}^T + \|\mathbf{u}\|^2 \mathbf{n}\mathbf{n}^T = (\mathbf{w} + \mathbf{v})(\mathbf{w} + \mathbf{v})^T - \mathbf{w}\mathbf{w}^T \tag{B.6}$$

Now we analyze the eigenvalues of the matrix $\mathbf{Q}$.

For any column vector $\mathbf{x} \in \mathbb{R}^3$, a matrix $\mathbf{A}$ in the form $\mathbf{A} = \mathbf{x}\mathbf{x}^T$ is symmetric matrix, which has only real eigenvalues. Therefore, as $\mathbf{Q}$ is the sum of two terms of the form $\mathbf{x}\mathbf{x}^T$, $\mathbf{Q}$ is the sum of two symmetric matrices. Hence, $\mathbf{Q}$ itself is symmetric and all its eigenvalues are real.

Furthermore, any matrix $\mathbf{A} = \mathbf{x}\mathbf{x}^T - \mathbf{y}\mathbf{y}^T$ with linearly independent $\mathbf{x}$ and $\mathbf{y}$ has one positive and one negative eigenvalue. Indeed, if a vector $\mathbf{z}$ is a linear combination of $\mathbf{x}$ and $\mathbf{y}$ and is orthogonal to $\mathbf{y}$, then $\mathbf{z}^T\mathbf{A}\mathbf{z} = (\mathbf{z}^T\mathbf{x})^2 > 0$, therefore $\mathbf{A}$ is not negative definite or semi-negative definite and then $\mathbf{A}$ must have a positive eigenvalue. Given another vector $\mathbf{z}$ which is a linear combination of $\mathbf{x}$ and $\mathbf{y}$ and is orthogonal to $\mathbf{x}$, then $\mathbf{z}^T\mathbf{A}\mathbf{z} = -(\mathbf{z}^T\mathbf{y})^2 < 0$, therefore $\mathbf{A}$ is not positive definite or semi-positive definite and then $\mathbf{A}$ must have a negative eigenvalue.

Therefore if $\mathbf{w}$ and $\mathbf{v} + \mathbf{w}$ are linearly independent, then $\mathbf{Q}$ has a positive and a negative eigenvalue.

Now the following facts will be used: let $e \in \mathbb{R}$ be an eigenvalue of a matrix $\mathbf{A}$, then, for any constant $c \in \mathbb{R}$:

1. ce is an eigenvalue of the matrix $c\mathbf{A}$.

2. $c + e$ is an eigenvalue of the matrix $c\mathbf{I} + \mathbf{A}$.

As $\mathbf{H}_\lambda^T\mathbf{H}_\lambda = \lambda^2(\mathbf{I} + \mathbf{Q}) = \lambda^2\mathbf{I} + \lambda^2\mathbf{Q}$, the eigenvalues of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$ can be calculated as a function of the eigenvalues of $\mathbf{Q}$ and vice versa. Applying these two facts, we know that if one the eigenvalues of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$ is λ^2, than zero is one of the eigenvalues of $\mathbf{Q}$, as $\lambda^2 + \lambda^2 \cdot 0 = \lambda^2$.

Therefore, $\mathbf{Q}$ has one positive and one negative eigenvalue and a zero eigenvalue, except when $\mathbf{u} \sim \mathbf{n}$. In this case $\mathbf{w}$ and $\mathbf{v} + \mathbf{w}$ are linearly dependent and the rank of $\mathbf{Q}$ is 1, and therefore $\mathbf{Q}$ have two repeated zero eigenvalues. In any case, the second largest eigenvalue of $\mathbf{Q}$ is zero. This special case happens when the direction of the translation $\mathbf{t}$ is proportional to the plane normal $\mathbf{n}$, i.e., when the direction of movement is perpendicular to the 3D plane.

As the second largest eigenvalue of $\mathbf{Q}$ is zero, λ^2 is the second largest eigenvalue of $\mathbf{H}_\lambda^T\mathbf{H}_\lambda$ and therefore $|\lambda|$ is the second largest singular value of $\mathbf{H}_\lambda$.

■

Once $|\lambda|$ is known, a new matrix is set as:

$$\mathbf{H} = \mathbf{H}_\lambda / |\lambda| \tag{B.7}$$

and thus $\mathbf{H}$ is recovered up to the form $\mathbf{H} = \pm\left(\mathbf{R} - \mathbf{t}\mathbf{n}^T/d\right)$. The correct sign is recovered by imposing a positive depth constraint using the pixel correspondences, which are given in the form of pixel pairs $(\mathbf{x}_i, \mathbf{x}'_i)$, for $i = 1 \ldots n$. Remembering that $\mathbf{x}'_i \sim \mathbf{H}\mathbf{x}_i$, we have, for all i:

$$\mathbf{x}'^T_i\mathbf{H}\mathbf{x}_i > 0 \tag{B.8}$$

for the correct sign of $\mathbf{H}$. Thus if the pixel correspondences are on general configuration on the plane, the matrix $\mathbf{H} = \left(\mathbf{R} - \mathbf{t}\mathbf{n}^T/d\right)$ can be uniquely determined.

B.2 Calculating sigma points for the Unscented Transform

The Unscented Transform is applied to the transformation from the virtual camera image plane to the ground plane, expressed by equation (3.4), to allow uncertainty propagation from the camera pose and orientation to the object position in the ground plane. Sigma points are necessary for the six dimensions corresponding to three rotational and three translational DOF.

Developing and inverting equation (3.4), and changing the origin of the horizontal coordinates to the world frame, we have:

$$
{}^{\mathcal{W}}\boldsymbol{X}_O = \begin{bmatrix} \frac{({}^{\mathcal{D}}o_x - n_x)h}{f} + {}^{\mathcal{W}}C_x \\ \frac{-({}^{\mathcal{D}}o_y - n_y)h}{f} + {}^{\mathcal{W}}C_y \\ 0 \end{bmatrix} \tag{B.9}
$$

where the observation of the object in the image coordinates, in the virtual camera frame $\{\mathcal{D}\}$, is ${}^{\mathcal{D}}\mathbf{x}_O = [{}^{\mathcal{D}}o_x, {}^{\mathcal{D}}o_y, 1]^T$, and the nadir point $\mathbf{n}$ is also measured in the image.

The sigma points for the three translational DOF are trivially calculated: change the values of ${}^{\mathcal{W}}\boldsymbol{X}_C = [{}^{\mathcal{W}}C_x, {}^{\mathcal{W}}C_y, h]^T$ into equation (B.9) and recalculate ${}^{\mathcal{W}}\boldsymbol{X}_O$.

Sigma points for the rotational DOF The orientation of the camera do not appear in equations (3.4) and (B.9) because the virtual camera with the rotation compensated is being used. Nevertheless, it is possible to calculate the orientation of the observation in the virtual camera frame and generate the sigma points related to errors in this orientation. This is necessary because errors on the orientation estimate are the most significant in this scenario.

First we deal with the roll and pitch angles. The sigma points for the roll angles and the x axis are calculated here but there is a similar development to calculate the sigma points for the pitch angle and the y axis.

Define θ as the angle between the vertical and the projection ray from the camera center to the object position in the ground plane ${}^{\mathcal{W}}\boldsymbol{X}_O = [{}^{\mathcal{W}}O_x, {}^{\mathcal{W}}O_y, 0]^T$. Looking to the triangle formed by the camera center, its nadir point in the ground plane and ${}^{\mathcal{W}}\boldsymbol{X}_O$, we have $\tan\theta = ({}^{\mathcal{W}}O_x - {}^{\mathcal{W}}O_x)/h$

Lets calculate the new value of ${}^{\mathcal{W}}O_x$ if θ is changed by an amount $+\Delta\theta$:

$$\begin{aligned}\chi^{+}_{pitch} &= h\tan(\theta+\Delta\theta) + {}^{\mathcal{W}}C_x \\ &= h\left(\frac{\tan(\theta)}{1-\tan(\theta)\tan(\Delta\theta)} + \frac{\tan(\Delta\theta)}{1-\tan(\theta)\tan(\Delta\theta)}\right) + {}^{\mathcal{W}}C_x\end{aligned} \tag{B.10}$$

and similarly for the negative sigma point:

$$\begin{aligned}\chi^{-}_{pitch} &= h\tan(\theta-\Delta\theta) + {}^{\mathcal{W}}C_y \\ &= h\left(\frac{\tan(\theta)}{1+\tan(\theta)\tan(\Delta\theta)} - \frac{\tan(\Delta\theta)}{1+\tan(\theta)\tan(\Delta\theta)}\right) + {}^{\mathcal{W}}c_y\end{aligned} \tag{B.11}$$

The sigma points for the yaw angle can be calculated by transforming the object coordinates from Cartesian to polar coordinates, changing the yaw angle, and transforming it back into Cartesian. A translation must be performed to place the origin in the camera center, as in the following algorithm:

$$\begin{bmatrix}\theta\\ \rho\end{bmatrix} \leftarrow cart2pol(({}^{\mathcal{W}}O_x - {}^{\mathcal{W}}C_x), ({}^{\mathcal{W}}O_y - {}^{\mathcal{W}}C_y)) \tag{B.12}$$

$$\chi^{+}_{yaw} \leftarrow pol2cart((\theta+\Delta\theta), \rho) + [\ {}^{\mathcal{W}}C_x \quad {}^{\mathcal{W}}C_y\ ]^T \tag{B.13}$$

$$\chi^{-}_{yaw} \leftarrow pol2cart((\theta-\Delta\theta), \rho) + [\ {}^{\mathcal{W}}C_x \quad {}^{\mathcal{W}}C_y\ ]^T \tag{B.14}$$

$$\chi^{+}_{yaw} \leftarrow \begin{bmatrix}\chi^{+}_{yaw}\\ 0\end{bmatrix} \tag{B.15}$$

$$\chi^{-}_{yaw} \leftarrow \begin{bmatrix}\chi^{-}_{yaw}\\ 0\end{bmatrix} \tag{B.16}$$

where the last two lines just set the z coordinate to zero as the object is on the ground plane, and *cart2pol* and *pol2cart* are the functions which transform Cartesian coordinates into polar coordinates and vice-versa.

B.3 Jacobian formulae for the homology transformation.

In section 3.2.3 the following equation is minimized to find the homology transformation which register two images:

$$\min_{v_x,v_y,v_w,\mu}\sum_{i=1}^{n} dist(\mathbf{x}'_i, \mathbf{G}\mathbf{x}_i) = \sum_{i=1}^{n} dist(\mathbf{x}'_i, \begin{bmatrix} 1 & 0 & (\mu-1)\cdot\frac{v_x}{v_w} \\ 0 & 1 & (\mu-1)\cdot\frac{v_y}{v_w} \\ 0 & 0 & \mu \end{bmatrix} \mathbf{x}_i) \tag{B.17}$$

where the FOE $\mathbf{v} = [v_x, v_y, v_w]^T$ is given in homogenous form, and n is the number of corresponding pixels between the image pair. Thus, for each corresponding pixel pair i, the function being minimized is, if *dist* is defined as Euclidean distance:

$$f(v_x, v_y, v_w, \mu) = \sqrt{\left(x'_x - \left(\frac{v_x}{v_w} + \frac{x_x}{\mu} - \frac{v_x}{v_w\mu}\right)\right)^2 + \left(x'_y - \left(\frac{v_y}{v_w} + \frac{x_y}{\mu} - \frac{v_y}{v_w\mu}\right)\right)^2} \tag{B.18}$$

where $\mathbf{x} = [x_x, x_y, 1]^T$ is given in unhomogeneous form, and $\mathbf{x}'$ is defined similarly. The partial derivatives of f in function of each of its four free variables, which are necessary to compose the Jacobian of the optimization function, are:

$$\frac{\partial f}{\partial v_x} = \frac{\left(\frac{1}{\mu}-1\right)}{dist(\mathbf{x}'_i, \mathbf{G}\mathbf{x}_i)\cdot v_w}\left(\left(x'_x - \frac{v_x}{v_w}\right) - \left(\frac{x_x - \frac{v_x}{v_w}}{\mu}\right)\right) \tag{B.19}$$

$$\frac{\partial f}{\partial v_y} = \frac{\left(\frac{1}{\mu}-1\right)}{dist(\mathbf{x}'_i, \mathbf{G}\mathbf{x}_i)\cdot v_w}\left(\left(x'_y - \frac{v_y}{v_w}\right) - \left(\frac{x_y - \frac{v_y}{v_w}}{\mu}\right)\right) \tag{B.20}$$

$$\begin{aligned}\frac{\partial f}{\partial v_w} = \frac{1}{dist(\mathbf{x}'_i, \mathbf{G}\mathbf{x}_i)\cdot v_w^2}\Bigg[&\left(\left(x'_x - \frac{v_x}{v_w}\right) - \left(\frac{x_x - \frac{v_x}{v_w}}{\mu}\right)\right)\left(v_x - \frac{v_x}{\mu}\right) + \\ &+\left(\left(x'_y - \frac{v_y}{v_w}\right) - \left(\frac{x_y - \frac{v_y}{v_w}}{\mu}\right)\right)\left(v_y - \frac{v_y}{\mu}\right)\Bigg]\end{aligned} \tag{B.21}$$

$$\begin{aligned}\frac{\partial f}{\partial \mu} = \frac{1}{dist(\mathbf{x}'_i, \mathbf{G}\mathbf{x}_i)\cdot \mu^2}\Bigg[&\left(\left(x'_x - \frac{v_x}{v_w}\right) - \left(\frac{x_x - \frac{v_x}{v_w}}{\mu}\right)\right)\left(x_x - \frac{v_x}{v_w}\right) + \\ &+\left(\left(x'_y - \frac{v_y}{v_w}\right) - \left(\frac{x_y - \frac{v_y}{v_w}}{\mu}\right)\right)\left(x_y - \frac{v_y}{v_w}\right)\Bigg]\end{aligned} \tag{B.22}$$

note that for faster computation, $\left(x'_x - \dfrac{v_x}{v_w}\right)$, $\left(x'_y - \dfrac{v_y}{v_w}\right)$, $\left(x_x - \dfrac{v_x}{v_w}\right)$, and $\left(x_y - \dfrac{v_y}{v_w}\right)$ are computed first for all corresponding points.

B.4 Jacobian formulae for the optimization in t.

In section 3.2.2, a set of corresponding points in two images are projected into the virtual horizontal plane, generating two sets of 3D points which are registered to find the translation between the two camera poses. The process involves optimization in the translation vector $\mathbf{t}$, and this section shows the partial derivatives of the following optimization function (equation (3.5)):

$$f\left(t_x, t_y, t_z, t_w\right) = \sum_{k=1\ldots n} dist\left(\boldsymbol{X}'_k\left(\mathbf{t}\right), \boldsymbol{X}_k\right) \tag{B.23}$$

which, if $dist()$ represents Euclidean distance, becomes, for each corresponding point pair:

$$\begin{aligned} f\left(t_x, t_y, t_z, t_w\right) \;=\; & \left[\left(\left(x'_x \cdot \frac{h}{f}\right) - \left(\frac{x_x \cdot \left(h + \frac{t_z}{t_w}\right)}{f} + \frac{t_x}{t_w}\right)\right)^2 + \right. \\ & \left. + \left(\left(-x'_y \cdot \frac{h}{f}\right) - \left(\frac{-x_y \cdot \left(h + \frac{t_z}{t_w}\right)}{f} + \frac{t_y}{t_w}\right)\right)^2\right]^{1/2} \end{aligned} \tag{B.24}$$

and then the partial derivatives are:

$$\frac{\partial f}{\partial t_x} = \frac{1}{dist(\boldsymbol{X}'_k(\mathbf{t}),\boldsymbol{X}_k)\cdot t_w}\cdot\left[+\left(x'_x - x_x\right)\cdot\frac{h}{f} - \frac{x_x\cdot t_z}{f\cdot t_w} - \frac{t_x}{t_w}\right] \tag{B.25}$$

$$\frac{\partial f}{\partial t_y} = \frac{1}{dist(\boldsymbol{X}'_k(\mathbf{t}),\boldsymbol{X}_k)\cdot t_w}\cdot\left[-\left(x'_y - x_y\right)\cdot\frac{h}{f} + \frac{x_y\cdot t_z}{f\cdot t_w} - \frac{t_y}{t_w}\right] \tag{B.26}$$

$$\begin{aligned}\frac{\partial f}{\partial t_z} &= \frac{1}{dist(\boldsymbol{X}'_k(\mathbf{t}),\boldsymbol{X}_k)\cdot t_w}\cdot\left[\left[+\left(x'_x - x_x\right)\cdot\frac{h}{f} - \frac{x_x\cdot t_z}{f\cdot t_w} - \frac{t_x}{t_w}\right]\left(\frac{-x_x}{f\cdot t_w}\right)+\right.\\ &\quad\left.+\left[-\left(x'_y - x_y\right)\cdot\frac{h}{f} + \frac{x_y\cdot t_z}{f\cdot t_w} - \frac{t_y}{t_w}\right]\left(\frac{x_y}{f\cdot t_w}\right)\right]\end{aligned} \tag{B.27}$$

$$\begin{aligned}\frac{\partial f}{\partial t_w} &= \frac{1}{dist(\boldsymbol{X}'_k(\mathbf{t}),\boldsymbol{X}_k)\cdot t_w^2}\cdot\left[\left[+\left(x'_x - x_x\right)\cdot\frac{h}{f} - \frac{x_x\cdot t_z}{f\cdot t_w} - \frac{t_x}{t_w}\right]\left(\frac{x_x\cdot t_z}{f}+t_x\right)+\right.\\ &\quad\left.+\left[-\left(x'_y - x_y\right)\cdot\frac{h}{f} + \frac{x_y\cdot t_z}{f\cdot t_w} - \frac{t_y}{t_w}\right]\left(\frac{-x_y\cdot t_z}{f}+t_y\right)\right]\end{aligned} \tag{B.28}$$

where, to speed up the calculation, the following quantities which appear repeatedly can be calculated first:

$$\begin{gathered}\frac{1}{dist(\boldsymbol{X}'_k(\mathbf{t}),\boldsymbol{X}_k)\cdot t_w}\\ \left[+\left(x'_x - x_x\right)\cdot\frac{h}{f} - \frac{x_x\cdot t_z}{f\cdot t_w} - \frac{t_x}{t_w}\right]\\ \left[-\left(x'_y - x_y\right)\cdot\frac{h}{f} + \frac{x_y\cdot t_z}{f\cdot t_w} - \frac{t_y}{t_w}\right]\end{gathered} \tag{B.29}$$

B.5 The discrete Wiener process acceleration model for the Kalman Filter

The Kalman Filters used in chapter 3 to filter the airship trajectory and in chapter 5 to filter the target trajectory utilize the discrete Wiener process acceleration model [Bar-Shalom et al., 2001], where the process noise is the acceleration increment during the sample period k and it is assumed to be a zero-mean white noise sequence. The filter state $\mathbf{X}$ consist in pose, velocity and acceleration:

$$(filter\,state)\ \mathbf{X} = \begin{bmatrix}\mathrm{x}\\ \dot{\mathrm{x}}\\ \ddot{\mathrm{x}}\end{bmatrix} \tag{B.30}$$

where $\mathbf{x}$, in this thesis, may be the airship position (3D) or target position (2D). The state equation is:

$$(state\,equation)\ \mathbf{X}(k+1) = \mathbf{F}\mathbf{X}(k) + \mathbf{\Gamma}v(k) \tag{B.31}$$

where, with $\mathbf{I}$ and $\mathbf{0}$ representing the identity and zero matrix of appropriate size, and T representing the length of the sample period:

$$\mathbf{F} = \begin{bmatrix} \mathbf{I} & T\mathbf{I} & \frac{T^2}{2}\mathbf{I} \\ \mathbf{0} & \mathbf{I} & T\mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{B.32}$$

$$\mathbf{\Gamma} = \begin{bmatrix} \frac{T^2}{2}\mathbf{I} \\ T\mathbf{I} \\ \mathbf{I} \end{bmatrix} \tag{B.33}$$

Therefore the prediction step is performed by the following equations:

$$(prediction\,of\,state)\ \mathbf{X}(k+1|k) = \mathbf{F}\mathbf{X}(k|k) \tag{B.34}$$

$$(prediction\,of\,covariance)\ \mathbf{P}(k+1|k) = \mathbf{F}\mathbf{P}(k|k)\mathbf{F}' + \mathbf{Q} \tag{B.35}$$

$$\mathbf{Q} = \begin{bmatrix} \frac{T^4}{4}\mathbf{I} & \frac{T^3}{2}\mathbf{I} & \frac{T^2}{2}\mathbf{I} \\ \frac{T^3}{2}\mathbf{I} & T^2\mathbf{I} & T\mathbf{I} \\ \frac{T^2}{2}\mathbf{I} & T\mathbf{I} & \mathbf{I} \end{bmatrix} \cdot \sigma_v^2 \tag{B.36}$$

The value of σ_v should be of the order of magnitude of the maximum acceleration increment over the sample period.

B.6 The Procrustes Procedure

The similarity Procrustes[1] problem consists in finding a transformation to register two sets of points in an Euclidean space, with known point correspondences. More formally, the problem variant which is considered in this thesis is: given two sets of n points in $\mathbb{R}^d$, in the form of $n \times d$ matrices $\mathbf{X}$ and $\mathbf{Y}$, where the ith line in both matrices correspond to the same point $\boldsymbol{P}_i$, the transformation parameters s, $\mathbf{t}$ and $\mathbf{R}$ such that $\mathbf{Y} = s\mathbf{X}\mathbf{R} + \mathbf{1}\mathbf{t}^T$ must be determined. The notation $\mathbf{1}$ represents a vector of ones.

[1]In Greek mythology, Procrustes ($\Pi\rho o k\rho o\acute{\upsilon}\sigma\tau\eta\varsigma$), son of Poseidon, was an evil innkeeper whose inn offered a wonderful all-fitting bed. The innkeeper “fitted” his unlucky guests to his bed by forcefully stretching their bodies or chopping off their legs. If a guest luckily fitted the first bed perfectly, then there were a second, differently sized bed. The young Theseus killed the innkeeper by fitting him to his own bed.

The derivation and proof of the solution can be found in many places including [Borg and Groenen, 1997], from which it was taken, and [Gower and Dijksterhuis, 2004]. The latter offers an extensive treatment of many variations of the problem. The steps to calculate the transformation are:

1. Compute $\mathbf{C} = \mathbf{X}^T\mathbf{J}\mathbf{Y}$, where $\mathbf{J}$ is the matrix $\mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^T$.
2. Compute the SVD of $\mathbf{C}$, and obtain the following matrices in the form $\mathbf{C} = \mathbf{P}\mathbf{\Phi}\mathbf{Q}^T$.
3. The optimal reflection/rotation matrix is $\mathbf{R} = \mathbf{Q}\mathbf{P}^T$.
4. The optimal scaling factor is $s = tr\left(\mathbf{X}^T\mathbf{J}\mathbf{Y}\mathbf{R}\right)/tr\left(\mathbf{Y}^T\mathbf{J}\mathbf{Y}\right)$.
5. The optimal translation vector is $\mathbf{t} = n^{-1}\left(\mathbf{X} - s\mathbf{Y}\mathbf{R}\right)^T\mathbf{1}$.

The scaling factor s is the inverse of the relative depth μ in the registration problem of section 3.2.2.