TAMPERE UNIVERSITY OF TECHNOLOGY

PETTERI KYRÖNLAHTI

CAMERA POSE ESTIMATION FROM AERIAL IMAGES

Master of Science Thesis

Examiners: Professor Ari Visa & Laboratory Engineer Heimo Ihalainen
Examiners and topic approved by the Faculty Council of the Faculty of Computing and Electrical Engineering on 9 May 2012.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Signal Processing and Communications Engineering
PETTERI KYRÖNLAHTI : Camera Pose Estimation from Aerial Images
Master of Science Thesis, 60 pages
September 2012
Major: Multimedia
Examiners: Professor Ari Visa & Laboratory Engineer Heimo Ihalainen
Keywords: aerial images, image registration, visual odometry, camera pose estimation

This thesis demonstrates the applicability of the digital camera as an aerial positioning device. The necessary theory behind digital, optical imaging systems and geometrical image formation is presented. In addition, basic image distortions and camera calibration are introduced. However, the main emphasis is on the correspondence problem between two images and on camera pose estimation. The position and orientation of the camera can be estimated relatively to previous known coordinates or absolutely to some reference coordinate system.

In relative camera pose estimation, the correspondences between two consecutive images can be recognized from image derivatives. In general, differential methods are used for low resolution images with high frame rates. For high resolution images, feature-based methods are generally more appropriate. Image features are often detected with subpixel accuracy, and their surroundings are described with feature vectors. These feature vectors are matched between two images to find the pointwise correspondences. The relative translation and orientation of the camera can be estimated from the correspondences. However, the major problem in all relative positioning methods is the error accumulation, where errors from previous estimations are accumulated to further estimations.

The error accumulation can be avoided by registering sensed aerial images to previously captured georeferenced images, which coordinates are known for every pixel. In this thesis, image registration between the reference image and an aerial image is implemented manually. Position and orientation of a camera are estimated absolutely to the reference coordinate system.

This thesis presents algorithms to solve the correspondence problem and to estimate the relative and absolute position and orientation of an aerial camera. The presented algorithms are verified with virtual Google Earth images and real-life aerial images from the test flight. In addition, the performance of the algorithms is also analyzed in terms of noise resistance.

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Signaalinkäsittelyn ja Tietoliikkennetekniikan koulutusohjelma
PETTERI KYRÖNLAHTI: Kameran Paikannus ja Orientointi Ilmakuvista
Diplomityö, 60 sivua
Syyskuu 2012
Pääaine: Multimedia
Tarkastajat: Professori Ari Visa & Laboratorioinsinööri Heimo Ihalainen
Avainsanat: ilmakuvat, kuvien rekisteröinti, visuaalinen odometria, kameran paikannus ja orientointi

Työssä perehdytään kameran sijainnin ja orientaation estimointiin ilmakuvista. Ensin esitellään työn kannalta tarvittava yleinen teoria optisesti kuvantavista digitaalisista kameroista ja geometrisesta kuvanmuodostuksesta. Lisäksi tarkastellaan keskeisiä geometrisia vääristymiä ja menetelmiä niiden poistoon, ts. kameran kalibrointia. Työn pääsisältö on kuitenkin vastinpisteiden tunnistaminen kahdesta kuvasta sekä kameran sijainnin ja orientaation estimointi löydetyistä vastinpisteistä. Paikannus ja orientointi voidaan tehdä suhteessa edelliseen tunnettuun estimaattiin tai absoluuttisesti suoraan tunnettuun referenssikoordinaatistoon.

Suhteellisessa paikantamisessa peräkkäisten kuvien välillä vastinpisteet voidaan tunnistaa kuvien muutoskohdista eli differentiaaleista. Differentiaaliset menetelmät soveltuvat matalan resoluution kuviin ja kuvasarjoihin, joiden päivitystaajuus on nopea. Sen sijaan korkean resoluution kuviin soveltuvat paremmin kuvan piirteisiin perustuvat menetelmät, jotka etsivät kuvista haluttuja piirteitä, kuten kulmia tai läiskiä. Piirteille lasketaan sijainti tyypillisesti alipikselitarkkuudella ja piirteen ympäristö kuvataan piirrevektorilla. Piirrevektoreita vertaamalla peräkkäisistä kuvista voidaan tunnistaa samoja vastinpisteitä. Samojen vastinpisteiden avulla kameran suhteellinen sijainti- ja orientaatio voidaan selvittää. Suhteellisen paikannuksen ongelmana on virheen kumuloituminen seuraaviin sijainti- ja orientaatioestimaatteihin, mikä heikentää menetelmien paikannustarkkuutta pitkällä aikavälillä.

Virheen kumuloituminen voidaan kuitenkin estää rekisteröimällä ilmakuva aikaisemmin otettuun georeferöityyn ilmakuvaan tai kartta-aineistoon. Tässä työssä referenssikuvan ja ilmakuvan rekisteröinti suoritetaan manuaalisesti, mutta varsinainen sijainti- ja orientaatioestimointi suoritetaan absoluuttisesti tunnettuun koordinaatistoon.

Työssä esitellään tunnetuimmat menetelmät sekä vastinpisteiden löytämiseen että niiden pohjalta tehtävään kameran paikannukseen. Työssä esitellyt menetelmät todennettaan käyttämällä virtuaalisia ilmakuvia Google Earth:stä ja oikeita ilmakuvia testilennolta. Lisäksi menetelmien tarkkuutta arvioidaan kohina-analyysillä.

PREFACE

I started my studies at the *Tampere University of Technology* (TUT) six years ago. At the end of the first academic year, I also started to work in the *Multimedia and Data Mining Research Group* (MMDM). To begin with, I would like to thank Ari Visa and the whole MMDM group for giving me this opportunity. All these years have been tremendously rewarding, both professionally and personally.

I have always been attracted by the beauty of photography and the technical possibilities of digital cameras. My adventurous mind led me to study in Singapore for a while. As an exchange student, I took courses on computer vision and image processing. Back in TUT, I continued my studies related to signal processing and multimedia. As if on cue, my work in MMDM included assignments related to cameras, especially to camera pose estimation. Altogether, I realized that it would be possible to write my thesis on a topic that I am truly interested in.

Writing this thesis has been both troublesome and satisfactory. The amount of information related to this field is vast. I still think that I have just scratched the surface. Fortunately, ever-growing understanding has increased the urge and passion to learn more.

For inspiration and meaningful conversations related to this thesis, I would like to give special thanks to Heimo Ihalainen. I also want to thank my colleagues and Matti Jukola for valuable feedback and suggestions. Moreover, I am grateful to Jarkko Tikka and Patria for providing useful material for real-life scenarios.

Last but definitely not least, I address my deepest gratitude to my tender and loving wife, Saila. She is the best thing that ever happened to me.

Petteri Kyrönlahti, Tampere, 13th of September 2012

Vehkakatu 1 B 8
33580 Tampere FINLAND
+358408437080
petkyron@gmail.com

CONTENTS

ABBREVIATIONS

APS	Active-Pixel Sensor
A/D	Analog-to-Digital Converter
CCD	Charge-Coupled Device
CFA	Color Filter Array
CMOS	Complementary Metal-Oxide-Semiconductor
DLT	Direct Linear Transformation
DoF	Degrees of Freedom
DoG	Difference of Gaussians
DoH	Determinant of Hessian
DSLR	Digital Single-lens Reflex Camera
EKF	Extended Kalman Filter
FPA	Focal Plane Array
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
JPEG	Joint Photographic Expert Group
INS	Inertial Navigation System
KLT	Kanade-Lucas-Tomasi feature tracker
KML	Keyhole Markup Language
LIBVISO	Library for Visual Odometry
LMA	Levenberg-Marquardt algorithm
LoG	Laplacian of Gaussian
MAE	Mean Absolute Error
MATLAB	Matrix Laboratory
MEMS	Micro-electromechanical Systems
MMDM	Multimedia and Data Mining
MP	Megapixel
OpenCV	Open Source Computer Vision Library
PT	Photon Transfer
PTC	Photon Transfer Curve
QE	Quantum Efficiency
SAD	Sum of Absolute Differences
SIFT	Scale-Invariant Features Transform
SNR	Signal-to-noise ratio
SSD	Sum of Squared Difference
STD	Standard Deviation

SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
TIFF	Tagged Image File Format
TUT	Tampere University of Technology
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter
WGS	World Geodetic System

SYMBOLS

Chapter 2

Δ_x	Horizontal length of a pixel
Δ_y	Vertical length of a pixel
δ_z	Distance difference between z-coordinates
Δ_t	Exposure time
θ	Rotation around z-axis
φ	Rotation around x-axis
ψ	Rotation around y-axis
e	Epipole
E	Essential matrix
f	Focal length
F	Fundamental matrix
k_n	Radial distortion coefficient
K	Camera calibration matrix
$\mathbf{l}$	Homogenous representation of a line
n	Bit depth
o	Optical center of the camera
P	Camera projection matrix
P_a	Affine camera projection matrix
p_n	Tangential distortion coefficient
p_x	Principal point offset in x-dimension
p_y	Principal point offset in y-dimension
r	Radial distance from the optical center
R	Rotation matrix
t	Time
T	Translation vector
t_0	Initial time moment
u	Column index of image sensor
v	Row index of image sensor
W	Object point in a coordinate frame
x	Horizontal coordinate of image sensor
$\mathbf{x}$	Homogenous representation of image coordinates
$\tilde{x}$	Homogenous representation of normalized image coordinates
X	x-coordinate of the object in the world coordinate frame

y	Vertical coordinate of image sensor
Y	y-coordinate of the object in the world coordinate frame
Z	z-coordinate of the object in the world coordinate frame

Chapter 3

λ	Eigenvalue of the structure tensor
σ	Size of the Gaussian kernel
∇I	Image gradient
A	System of linear equations
C	Structure tensor
I	Image brightness
k	Constant between two consecutive Gaussian kernels
L	Feature detection function
$\mathbf{v}$	Motion field
w	Windowing function

Chapter 4

ω	Scale factor
D	Diagonal matrix of singular values in SVD
d_1	Euclidean distance from the plane to the first camera
h	Length of the line segment inside the circular field of view
H	Homography
H_c	Calibrated homography
k	Tangent of the slope
n_1	Unitary plane normal vector
S	Skew symmetric matrix
U	Left-hand unitary matrix in SVD
V	Right-hand unitary matrix in SVD
v_h	Horizontal vanishing point
v_v	Vertical vanishing point

1. INTRODUCTION

Location awareness is a term which refers to devices that can determine their location locally, regionally, or globally. Knowing the exact location enables various positioning, navigation, mapping, and homing applications. Positioning is also an important step in making ground or aerial vehicles autonomous or even unmanned. Moreover, navigation generally requires absolute location information of a vehicle in a known coordinate system. However, positioning can be made relatively to previously known coordinates. Obvious weakness for all relative positioning methods is the error propagation because errors from the previous estimations are accumulated to further estimations. In aerial vehicles, increased instability makes positioning and navigation even more difficult and inaccurate than in ground vehicles. In aerial positioning and navigation, all the possible additional information from sensory data is exploited.

Currently, probably the most widely used methods for absolute aerial positioning are *global navigation satellite systems* (GNSS). GNSS receivers can be found in everything from mobile phones to spacecrafts, but they also have weaknesses. Like all radio systems, they need a proper signal reception to function. This is a problem in shaded environments, such as urban and mountainous areas. Another crucial problem is that the low power GNSS signal can be interfered with a high power noise jammer or a spoofer which degrades the accuracy of the GNSS receiver. However, there exist positioning methods which are not so fragile for jamming.

An *inertial navigation system* (INS) is a passive positioning method incorporating accelerometers and gyroscopes for determining the position, orientation, and velocity of the device. INS can only be used for relative positioning when no external information is available. In addition, accurate inertial systems are usually very expensive and heavy for small devices and vehicles. However, relatively cheap and lightweight *micro-electromechanical systems* (MEMS) are developing rapidly and can be used in applications where accuracy is not the most crucial component. INS provides continuous and high rate pose information. In many cases, there are also other sensors, such as GPS, barometric altitude sensor, or magnetometer, to provide position and orientation information. The actual pose of the vehicle is estimated as a combination of various sensors. Furthermore, recent developments of digital cameras and computing power have extended the use of digital images, and they can be

used as another source of position and orientation information for aerial navigation purposes.

Cameras are small, affordable, and quite easy to implement, even in to small aerial vehicles. They offer an interesting failsafe or a complementary source of information in addition to GNSS, INS, or other sensors used in flight control and aerial positioning. Also, cameras may have other critical functions in aerial applications, such as object detection and recognition, for which reason a camera may already be a payload of an aerial vehicle. Cameras have been used in aerial imaging, photogrammetry, and cartography for a long time but not for long in positioning and navigation.

Cameras can be used for both relative and absolute pose estimation. Visual odometry is a process for determining the position and orientation of a camera from consecutive digital images. It is generally used in robotics and computer vision applications. Visual odometry, like INS, can only be used for relative positioning. General idea in visual odometry is to find correspondences between consecutive images and to calculate translation and rotation based on those pointwise correspondences. Image registration is the process of determining these correspondences and calculating the transformation between two different sets of data. From digital imagery itself, it is impossible to determine the location of a camera globally without a priori information.

However, if a captured image is registered to some georeferenced image, whose global coordinates are known, it is then possible to estimate the coordinates of the captured image and even the global coordinates of the camera from a single digital image. Unfortunately, absolute pose estimation is rather cumbersome problem as reference images and current environment may have very little mutual information. This is mainly due to different natural conditions. That is the main challenge for image registration algorithms. Generally, higher level or multimodal image registration is very application dependent and very hard to make robust enough for outdoor applications and aerial positioning purposes. In this thesis, images in absolute pose estimation are registered manually. After the manual correspondence search, the absolute pose is estimated from pointwise correspondences between a sensed image and a reference image.

The main objective of this thesis is to study how to utilize a camera as a positioning device, especially in the context of aerial applications. The main emphasis is on registration of two consecutive images, and how to estimate a pose from pointwise correspondences. Algorithms for both relative and absolute pose estimation are introduced. Some of the algorithms are chosen to be verified and analyzed with virtual Google Earth data and real-world flight data. In addition, the performance of these algorithms are also analyzed in terms of noise resistance. The fundamental

theory behind digital cameras and geometrical image formation is also covered. This includes a brief examination of digital cameras, the mathematical model of a mono and stereo camera, image transformations and distortions, and camera calibration.

Chapter 2 explains the fundamental theory behind digital imaging systems and geometrical image formation, and depicts how a point in a 3D world is projected onto 2D image plane. Chapter 3 introduces the correspondence problem and algorithms, which try to find these correspondences from two different images. Chapter 4 explores algorithms for relative and absolute pose estimation from single or consecutive views. Chapter 5 analyzes the performance of the presented algorithms in virtual and real-life pose estimation experiments and in more controlled noise analysis. Chapter 6 concludes the work and discusses about future aspects and research possibilities.

2. DIGITAL IMAGING SYSTEM

Usual purpose of imaging is to make a visual representation of a real world object, scene, or phenomenon. Imaging can be done throughout the electromagnetic spectrum and can be everything from molecular to radar imaging. Various applications represent retrieved information in a digital format, in many cases in two dimensions. That is because vision and images play the single most important role in human perception [16, p. 24]. Digital photography in the visual band of the electromagnetic spectrum is the most common form of digital imaging, and this chapter covers the fundamentals of digital cameras and geometrical image formation.

2.1 Digital Image Fundamentals

Photographic images are generated as a combination of an illumination source and an absorptive material. Irradiated energy from the scene is transformed into voltage by the combination of input electrical power and sensor material responsive to the particular type of energy being detected [16, p. 69]. Mainly three types of sensor arrangements are used: single light-sensitive element, line sensor, and array sensor.

Digital camera sensor is typically composed of a *focal plane array* (FPA) of solid state detectors which start to capture photons once the shutter is released. Photons are further converted into electrons through the photoelectric effect, and electrons are accumulated in the well during the exposure time [23]. A single detector, i.e. a pixel (picture element), performs sampling as integration over the spatial and time domain:

$$\Psi(x,y,t) = \int_{-\Delta_x/2}^{\Delta_x/2} \int_{-\Delta_y/2}^{\Delta_y/2} \int_{t_0}^{t_0+\Delta_t} \psi(x,y,t), \mathrm{d}x\, \mathrm{d}y\, \mathrm{d}t \ , \tag{2.1}$$

where Δ_x and Δ_y are the horizontal and vertical length of a pixel, t_0 is the initial moment of time, and Δ_t is the exposure time.

Sampling digitizes all coordinate values, but the amplitude values of the electric charge are also digitized, i.e. quantized. Quantization is made with an *analog-to-digital converter* (A/D) to 2^n separate digital values, where n is the bit depth. Typical values for bit depths are 8-bit, 12-bit, and 14-bit depending on the sensor type and the image file format. The basic idea behind sampling and quantization is illustrated in Figure 2.1. A scan line from A to B in the continuous image is sampled and quantized to digital values.

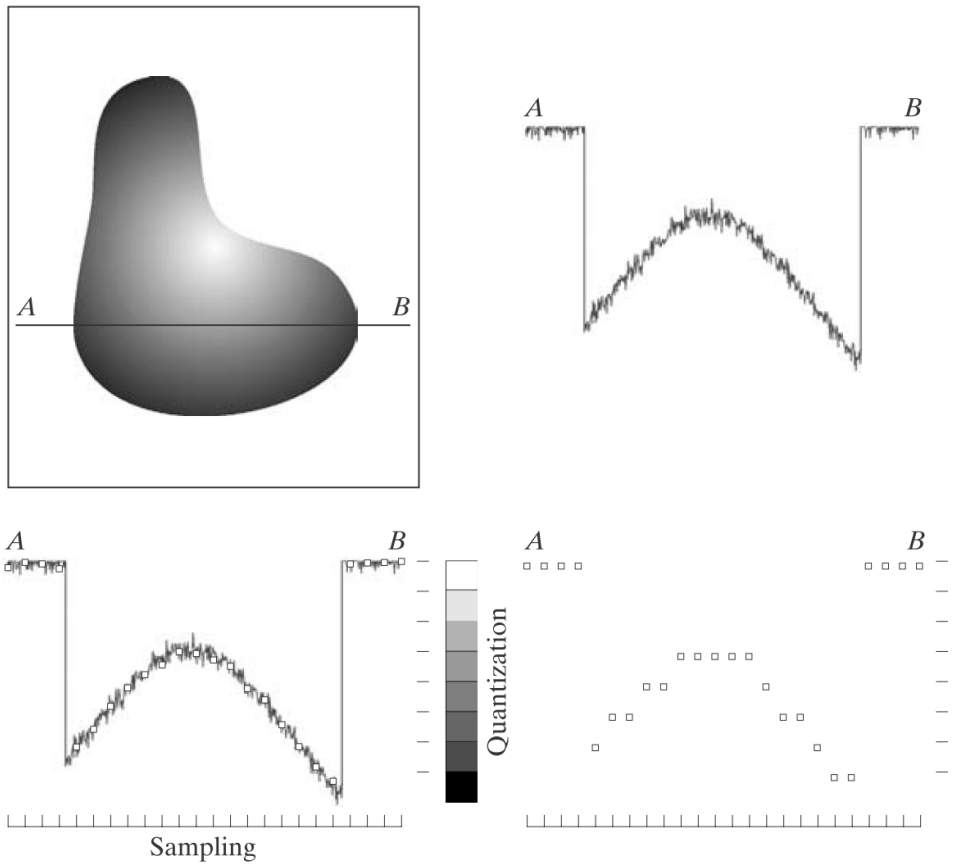

Figure 2.1: *Generating a digital image [16, p. 75].*

Sensors

Currently, the most used sensor types are *charge-coupled device* (CCD) and *complementary metal-oxide-semiconductor* (CMOS). Actually, CMOS is merely a manufacturing technique for an *active-pixel sensor* (APS). The idea behind CCD and CMOS sensor is somewhat similar, but the main difference is where the electrical charge of a single detector is converted to voltage.

In CCD sensor, the charge from each individual detector is shifted to the end of the row where it is converted to voltage and read as a digital value. CMOS (APS) converts the charge to voltage at each pixel using transistors and amplifiers. Both sensor types are used in various cameras, but they have had some advantages and disadvantages over each other. Traditionally, CMOS image sensors consume less power, have less image lag, require less specialized manufacturing facilities, and can combine image processing functions within the same integrated circuit. CCD image sensors generally have higher fill factor, better system noise performance, and lower sensor complexity [35]. However, camera technology evolves rapidly, and both type of sensors are used in areas which were previously dominated by the other sensor type.

In terms of picture quality and *signal-to-noise ratio* (SNR), one of the most important factor is the size of the photosensitive area: more photons a sensor can capture, higher the SNR and better the picture quality. High-end or professional *digital single-lens reflex* (DSLR) cameras use a full-frame 35 mm sensor format (area 864 mm^2) whereas compact consumer cameras generally use a 1/2.3" sensor format (area 28.5 mm^2) [21]. Depending on the resolution of the sensor, this yields that typical DSLRs have 1–5 MP/cm^2, and typical compact cameras have 20–60 MP/cm^2. Due to other circuitry, not all areas of the sensor can be used to collect light.

Fill factor depicts the percentage of the photosensitive area compared to the whole area of the sensor. Higher fill factor allows more photons to the photosensitive area of the sensor improving the noise performance of the camera. One way to increase the optical fill factor is to use microlens arrays in front of the imaging sensor. These tiny lenses try to focus and concentrate the incoming light onto the photosensitive area of the sensor.

Another quantity, which measures the photosensitivity of the sensor, is *quantum efficiency* (QE). QE is the percentage of the photons hitting the photosensitive area that will produce an electron-hole pair in the well [23]. On some wavelengths modern back-illuminated CMOS can have a QE over 90% while photographic film typically has a QE less than 10%. Fill factor and QE are both quantities which measure the sensor's ability to transform the desired signal to image, but there are also various sources of undesirable signal, i.e. noise.

Noise in imaging systems is random variations associated with detection and reproduction systems [40, p. 507]. The most valuable testing methodology for designing, characterization, optimization, calibration, and specification of solid state imagers and camera systemss is *photon transfer* (PT) [23]. The most basic form of PT includes a plot of noise versus signal, *a photon transfer curve* (PTC). In basic PT, there are three primary sources of noise: photon noise, fixed pattern noise and read noise.

Photon noise is related to photon interaction and the natural variation of the incident photon flux. The photoelectrons, collected by a detector, exhibits a Poisson distribution. Photon noise has a square root relationship between signal and noise and cannot be reduced via camera design.

Fixed pattern noise is a result of sensitivity differences in charge collection process between individual pixels. It is not random because the noise pattern stays the same from image to image, though, it may vary with integration time, imager temperature, imager gain, or incident illumination.

Read noise is a combination of system noise components, inherent to the process of converting charge carriers into a voltage signal, the subsequent processing, and

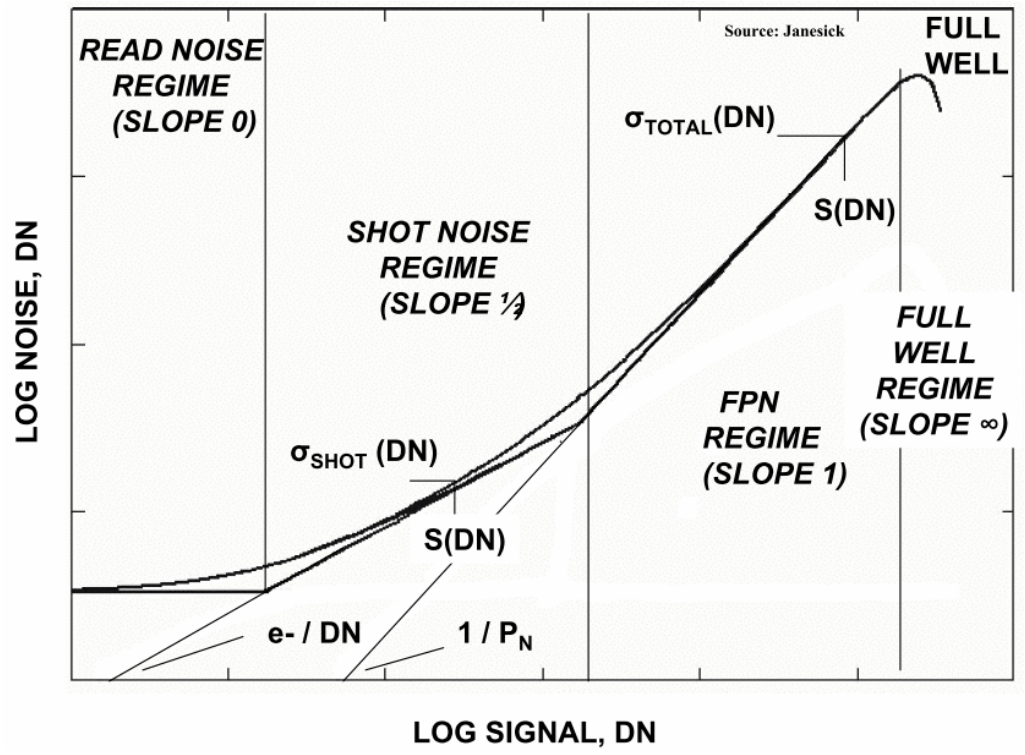

Figure 2.2: *Photon transfer curve [23].*

A/D conversion. In general, read noise may be defined as any noise that is not a function of signal originating mainly from on-chip preamplifiers. Read noise is added uniformly to every image pixel. High-performance camera systems utilize design enhancements that greatly reduce the significance of read noise.

Figure 2.2 shows an example of a PTC. It is divided into four regions, which are corresponding to a distinct regime dominated by one type of noise. Read noise is invariant of signal level, e.g. noise floor, meaning that very low signals are dominated by read noise. As the signal level increases, photon noise becomes more dominant. However, fixed pattern noise is directly proportional to signal level becoming dominant before the full well. Fixed pattern noise can be reduced with flat fielding techniques and camera design. In terms of noise, camera systems are limited to read noise and photon noise.

Another design enhancement for camera manufacturers is a demosaicing algorithm. In order to have limited color information with one imaging sensor, almost all cameras have a *color filter array* (CFA) in front of the sensor: typically a Bayer filter on which each two-by-two submosaic contains two green, one blue and one red filters [29]. The raw color filtered image data (RAW) is converted to a full color image by a demosaicing algorithm, which interpolates two thirds of the necessary information from neighboring pixels. The aim for a demosaicing algorithm is to avoid false colour artefacts, such as purple fringing and zippering, to preserve max-

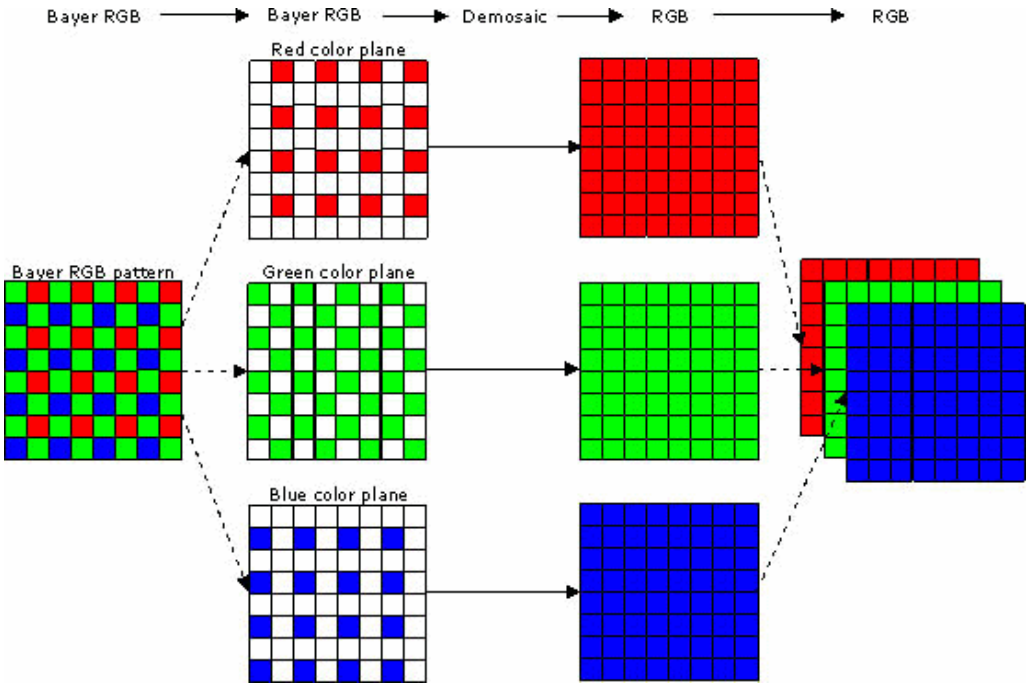

Figure 2.3: *Demosaicing process: raw color filtered image data is converted to a full RGB image [29].*

imum resolution of the image, and to minimize the computational cost. Figure 2.3 illustrates a demosaicing process of a color filtered raw image data.

Raw image data is converted either to a lossless image file, such as *Tagged Image File Format* (TIFF), or to a lossy image file, such as *Joint Photographic Expert Group* (JPEG). However, professional photographers generally use RAW image file format, such as Canon's .cr2. RAW image files offer higher image quality and freedom to use various settings and algorithms on a personal computer. In RAW images, original image data is available for demosaicing, sharpening, noise reduction, gamma correction, or white balance adjusting. In professional photography, RAW images are usually the starting point of the workflow.

2.2 Geometry of Image Formation

Geometry of image formation has been studied from the early days of photography. Traditionally, photogrammetry has been defined as the process of deriving metric information about an object through measurements made on photographs [32]. Initial applications were motivated by military considerations, but photogrammetry is now applied across a diverse set of commercial applications as well [34]. One of the most important application of photogrammetry is mapping, where the goal is to minimize the error between projected image feature positions and 3D ground control points. Furthermore, photogrammetry also has applications for closer range, such as anthropometrics, industrial metrology, or archeological surveying [34]. Nev-

ertheless, photogrammetry has experienced significant changes during the last two decades caused by advances in optics, electronics, imaging, and computer technologies. Fundamentals of photogrammetry were developed at the age of traditional film imaging while the modern imaging systems are purely digital.

Another field has evolved under the central theme of achieving human-level capability in the extraction of information from image data [34]. Computer vision is a field emerging from analyzing and understanding images and high-dimensional data from the real world in order to retrieve numerical or symbolic information. There are many computer vision applications, such as navigation, object recognition, and object modeling, since much of human experience is associated with images and visual information processing. 3D computer vision and photogrammetry share some similar goals, such as camera calibration, pose determination, model projection or model construction. Despite some similar goals, generally speaking, computer vision is more view-centered, and photogrammetry is more world-centered. Most of the references in this thesis are from the field of computer vision, though, the same principles can be found from the field of photogrammetry. This thesis uses same notation found from [18] and [13].

The Pinhole Camera Model

The pinhole camera model, in figure 2.4, represents how a point from a 3D object is projected through the optical center to an image point, $I(u, v)$, onto the 2D image plane. u and v are the column and row index of the image sensor. The camera projection is based on the collinearity condition, where an object point, an image point, and the optical center lie on the same line. The focal length, f, of the camera defines the distance between the image plane and the optical center. The line from the optical center perpendicular to the image plane is called the optical axis, and the point where the optical axis meets the image plane is called the principal point.

By similar triangles the relation between an object point and an image point is

$$I(u, v) = (fX/Z, fY/Z), \tag{2.2}$$

where the focal length, f, is expressed in $\frac{pixels}{meter}$. If the object and image points are represented by homogenous vectors, then the camera projection is expressed as a linear mapping between their homogenous coordinates [18, p. 154]. Homogenous coordinates are commonly used in computer vision because this ensures that the translation can also be expressed with a matrix multiplication as will be seen in

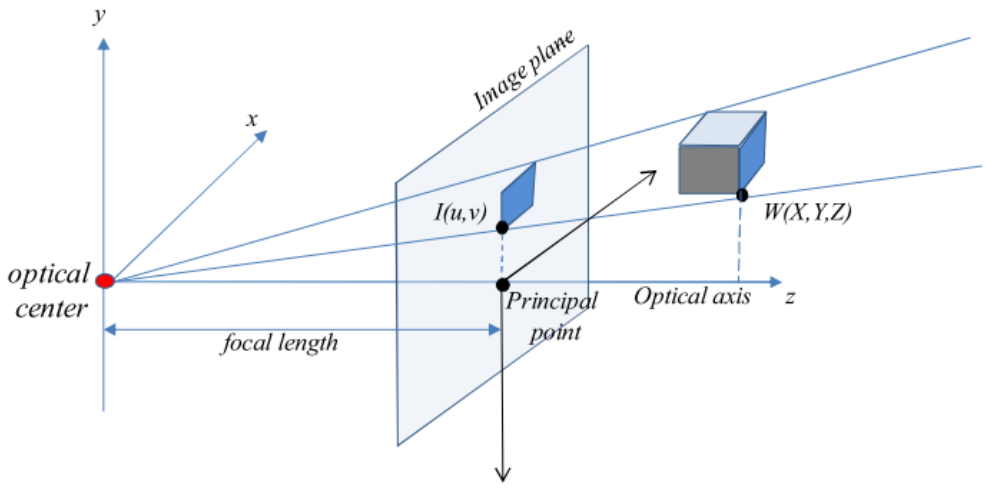

Figure 2.4: *Pinhole camera model [15].*

further equations. Equation (2.2) may be rewritten as a matrix multiplication

$$\mathbf{x} = \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \tag{2.3}$$

where $\mathbf{x}$ is a vector presentation of image coordinates. Actual image coordinates can be retrieved as follows: $u = x_1/x_3$ and $v = x_2/x_3$.

Above equation assumes that the optical axis passes trough the image plane at the corner of the image rather than at the principal point. In general, this is not the case. The principal point offset can be expressed conveniently in homogenous coordinates as

$$\mathbf{x} = \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \tag{2.4}$$

where p_x and p_y are the principal point offset in pixels in x- and y-dimensions. Above equation may be rewritten as

$$\mathbf{x} = K\,[I|0]\,W = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \tag{2.5}$$

where K is called the camera calibration matrix. Parameters in K are called the internal camera parameters.

World Coordinate Frame

In equation (2.5), the camera is located at the origin of the Euclidean coordinate system. When object points can be expressed in this coordinate system, such a coordinate system may be called the camera coordinate frame. In many cases, object points in 3D space are expressed in a different Euclidean coordinate frame, known as the world coordinate frame. Object points in the world coordinate frame need to be transformed to the camera coordinate frame. Rotation and translation between two Euclidean coordinate frames can be expressed as

$$W_c = [R|T]\, W_w = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}, \tag{2.6}$$

where R is a 3×3 rotation matrix, T is a 3×1 translation vector, W_w is a 4×1 homogenous vector for the object point in the world coordinate frame, and W_c is a 3×1 vector for the object point in the camera coordinate frame. As mentioned before, homogenous coordinates allow translation to be expressed as a simple matrix multiplication. The parameters of R and T are called the external camera parameters. Now, combining equations 2.5 and 2.6 yields to:

$$\mathbf{x} = K\,[R|T]\, W_w = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}, \tag{2.7}$$

where $K[R|T]$ is called the camera projection matrix P.

Camera pose, i.e. position and orientation, estimation is mainly estimation of these external parameters. When the camera calibration matrix is known, then it is possibly to apply its inverse to the image point $\tilde{x} = K^{-1}x$. Then $\tilde{x} = [R|T]\, W$, where $\tilde{x}$ is the image point expressed in normalized coordinates, from which the effect of the known calibration matrix is removed.

The matrix R contains rotations around all three axes (x - φ, y - ψ, z - θ), and the total rotation matrix is the multiplication of those components. Order of multiplication yields to a different rotation matrix, so it should be carefully decided which convention is used. Order of multiplication below first rotates coordinates

about x-axis, second about y-axis, and finally about z-axis.

$$R = R_z(\theta)R_y(\psi)R_x(\varphi), \tag{2.8}$$

where

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{2.9}$$

$$R_y(\psi) = \begin{bmatrix} \cos(\psi) & 0 & \sin(\psi) \\ 0 & 1 & 0 \\ -\sin(\psi) & 0 & \cos(\psi) \end{bmatrix} \tag{2.10}$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.11}$$

A rotation matrix is an orthogonal matrix, which transpose is equal to its inverse, i.e. $R^{\mathrm{T}} = R^{-1}$.

Affine Cameras

If the focal length or the distance between the camera and the object increases it is possible to use the weak-perspective assumption:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} fX/Z \\ fY/Z \end{pmatrix} \approx \begin{pmatrix} fX/Z_{ave} \\ fY/Z_{ave} \end{pmatrix}, \tag{2.12}$$

where Z_{ave} is the average distance of the points from the scene to the camera. The weak-perspective assumption becomes viable when the distance difference, δ_z, between all the Z-coordinates $\delta_z < Z_{ave}/20$. [43, p. 27]

A mathematical generalization of the weak-perspective camera is the affine camera model. In full generality, affine camera matrix has the form:

$$P_a = \begin{bmatrix} m_{11} & m_{12} & m_{13} & T_x \\ m_{21} & m_{22} & m_{23} & T_y \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.13}$$

where M_{2x3} is a rank 2 matrix. The full affine camera model is the abstraction of affine cameras, such as weak-perspective or orthographic camera, which satisfy additional constraints. For example, the rows of matrix M are scalings of rows of a full rotation matrix. The weak-perspective assumption and the affine camera model

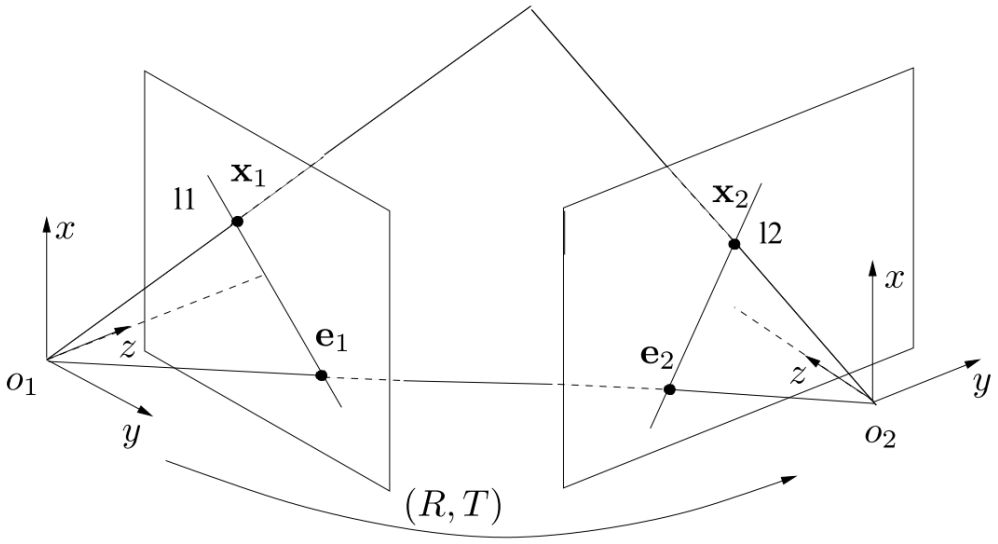

Figure 2.5: *Characterization of the epipolar geometry [28].*

can be used in aerial imaging applications when camera is close to nadir view even though that requires careful consideration and analysis.

2.3 Epipolar Geometry

Two views of the same scene may be acquired simultaneously, as in stereo rig, or sequentially, e.g. when the camera is moving relatively to the scene. Geometrically those situations are equivalent, and the same principles are valid in both cases. The intrinsic projective geometry of two views is called the epipolar geometry. [18, p. 239]

The epipolar geometry is illustrated in figure 2.5. A line from one camera center to another, o_1 and o_2 , is called the baseline. The point at the baseline, which intersects the image plane, is called the epipole, $\mathbf{e}_1$ and $\mathbf{e}_2$. Camera centers and a world point define the epipolar plane. The epipolar line is the intersection of the epipolar plane with the image plane. All epipolar lines intersect at the epipole.

Fundamental Matrix

Any point, $\mathbf{x}_1$, in the first image must lie on the epipolar line $\mathbf{l}_2$ in the second image. Conversely, any point ,$\mathbf{x}_2$, in the second image must lie on the epipolar line $\mathbf{l}_1$ in the first image. $\mathbf{l}_1$ and $\mathbf{l}_2$ is a homogenous representation of a line, $[a\ b\ c]$, where $a * x + b * y + c * z = 0$. The algebraic representation of this projective mapping from points to lines is represented by the fundamental matrix, F:

$$\mathbf{l}_2 = \mathrm{F}\mathbf{x}_1 \tag{2.14}$$

$$\mathbf{l}_1 = \mathrm{F}^{\mathrm{T}}\mathbf{x}_2. \tag{2.15}$$

In addition, the fundamental matrix links any pair of corresponding points in the two images

$$\mathbf{x}_2^{\mathrm{T}}\mathrm{F}\mathbf{x}_1 = 0. \tag{2.16}$$

This is known as the epipolar constraint.

Fundamental matrix can be derived in many ways. One algebraic derivation can be found from [18, p. 244]. If camera matrices are chosen such that the first camera is at the world origin, $P_1 = K[I|0]$, and another camera is rotated and translated, $P_2 = K'[R|T]$, the fundamental matrix may be derived as

$$\mathrm{F} = K'^{-\mathrm{T}}RK^{\mathrm{T}}[\mathrm{e}]_x, \tag{2.17}$$

where $[\mathbf{e}_1]_x$ is the epipole of the first camera written in a skew-symmetric form:

$$[\mathbf{e}_1]_x = \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix}. \tag{2.18}$$

As the epipolar constraint suggest, the fundamental matrix can be computed from image correspondences alone, and there exist several methods for computing it. Hartley and Zisserman, [18, Chap. 11] , give several methods for solving the fundamental matrix. For a quick method, which gives adequate results, they suggest the normalized 8-point algorithm. A simple normalization, including translation and scaling, before formulating the linear equations leads to an enormous improvement in the conditioning of the problem and in the stability of the result [18, p. 282]. After normalization a linear solution is obtained through *Singular Value Decomposition* (SVD). The epipolar constraint can also be enforced with the aid of SVD. For more accurate results, it is possible to minimize various geometric cost functions, such as the algebraic error or first-order geometric error.

General motion in rigid environment ensures that the fundamental matrix can be estimated uniquely, up to a scale factor. From noisy correspondences, the closest fundamental matrix can be enforced with SVD. However, a set of correspondences may be geometrically degenerate if they fail to uniquely define the epipolar geometry; i.e. if there exist more than one linearly independent fundamental matrices which fulfill the epipolar constraint. These cases arise when the motion is degenerate: only rotation about the camera center or all world points lie on a plane. Depending of the degeneracy, there may exist more than one fundamental matrix.

In camera pose estimation, it is not necessary to compute the fundamental matrix. Especially in case of calibrated cameras, it is possible to solve the specialization of the fundamental matrix from normalized image coordinates.

Essential Matrix

As in equation (2.7), the camera matrix is decomposed as $P = K[R|T]$, and the world point is projected onto the image plane as $\mathbf{x} = PW$. If the camera calibration K is known, it is possible to apply its inverse to the image point $\mathbf{x}$ to obtain the point $\tilde{\mathbf{x}} = K^{-1}\mathbf{x} = [R|T]W$. $\tilde{\mathbf{x}}$ is the image point expressed in normalized coordinates.

The fundamental matrix corresponding to the pair of normalized coordinates is customarily called the essential matrix [18, p. 257]. Similarly the epipolar constraint can be expressed

$$\tilde{\mathbf{x}}_2^{\mathrm{T}}\mathrm{E}\tilde{\mathbf{x}}_1 = 0. \tag{2.19}$$

Comparing this to the epipolar constraint, equation (2.16), the relationship between the essential and the fundamental matrix is

$$\mathrm{E} = \mathrm{K}'^{\mathrm{T}}\mathrm{F}\mathrm{K}. \tag{2.20}$$

If camera matrices are chosen as in equation (2.17), the essential matrix has the form

$$\mathrm{E} = [\mathrm{T}]_x R = R[R^{\mathrm{T}}\mathrm{T}]_x. \tag{2.21}$$

The essential matrix has five degrees of freedom: three for the rotation, three for the translation, and an overall scale ambiguity.

As with the fundamental matrix, the essential matrix can be computed using linear techniques from 8 points or more [18]. From noisy normalized correspondences, the closest essential matrix can be recovered with SVD. However, if the goal is to solve camera matrices, i.e. rotation and translation, it is not necessary to enforce the epipolar constraint because camera matrices can be recovered directly from the SVD, e.g. with the algorithm introduced by Tsai et al. [44]. This solution is given in chapter 4.

2.4 Image Transformations

Linear image transformations can be divided depending on how many elements or quantities they preserve, i.e. how many *degrees of freedom* (DoF) they have. The most common division of image transformations is: isometric (3 DoF), similarity (4 DoF), affine (6 DoF) and projective (8 DoF) [18].

Isometric transformation preserves the Euclidean distance and is represented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \epsilon\cos(\theta) & -\sin(\theta) & T_x \\ \epsilon\sin(\theta) & \cos(\theta) & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{2.22}$$

where θ is the rotation angle and $\epsilon = \pm 1$ defines that does the transformation preserve orientation, $(\epsilon = 1)$, or not, $(\epsilon = -1)$. Isometric transformation has three degrees of freedom: one for rotation and two for translation.

Similarity transformation is isometric transformation composed with isotropic scaling. Similarity transformation is represented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} s\cos(\theta) & -s\sin(\theta) & T_x \\ s\sin(\theta) & s\cos(\theta) & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{2.23}$$

where s is the isotropic scaling. Similarity transformation has four degrees of freedom: one for rotation, two for translation, and one for isotropic scaling. Similarity transformation can be computed from two corresponding points.

Affine transformation is a non-singular linear transformation followed by translation and is represented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & T_x \\ a_{21} & a_{22} & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{2.24}$$

where $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ is a 2×2 non-singular matrix. 2D affine transformation has six degrees of freedom corresponding to six matrix elements. It can be computed from three corresponding points.

Projective transformation is a general non-singular linear transformation of homogeneous coordinates and is represented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}. \tag{2.25}$$

Projective transformation between two planes has eight degrees of freedom and can be computed from four corresponding points. It is a pair of perspective projections and is often called homography.

All real-life cases include noise, so it is appropriate to solve transformations with minimization of some error function. Equation 4.13 shows one way to solve linear equations in a least-square sense. Naturally there exist various types of error functions and solutions, which are introduced in more detail in [18][chapter 4].

2.5 Image Distortions

In optical systems, cameras do not use pinholes but rather complex lens systems. Manufacturing process, mechanical mounting, zooming, and temperature variations affect the focal length of the camera and introduce geometrical distortions. In general, geometrical distortions can be classified into three different categories: radial, tangential, and linear distortion. Linear distortion is due to non-orthogonal displacement of pixels in the camera sensor. However, linear distortion is negligible in modern digital cameras.

Radial distortion arises mainly from the geometry and material of the lens. It is the most important and noticeable distortion. In radial distortion, coordinates in the observed image are displaced away from (barrel distortion) or towards (pincushion distortion) the image center by an amount proportional to their radial distance. Radial distortion is more severe in the periphery of the lens and is usually represented by the means of polynomial approximation [4]. For normalized coordinates, $[\ \tilde{u}\ \tilde{v}\ 1\ ]$, radial distortion is

$$\begin{pmatrix} \tilde{u}_r \\ \tilde{v}_r \end{pmatrix} = \begin{pmatrix} \tilde{u}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + ...) \\ \tilde{v}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + ...) \end{pmatrix}, \tag{2.26}$$

where k_n is a radial distortion coefficient and

$$r = \sqrt{\tilde{u}^2 + \tilde{v}^2}. \tag{2.27}$$

Tangential distortion is produced when the lens is not parallel to the image plane or the shape of the optical component is not symmetric. It is also called as decentering distortion. The model for tangential distortion is

$$\begin{pmatrix} \tilde{u}_t \\ \tilde{v}_t \end{pmatrix} = \begin{pmatrix} 2p_1\tilde{u}\tilde{v} + p_2(r^2 + 2\tilde{u}) \\ 2p_2\tilde{v}\tilde{u} + p_1(r^2 + 2\tilde{v}) \end{pmatrix}, \tag{2.28}$$

where p_n is a tangential distortion coefficient [4].

Total normalized distortion is merely a summation of radial and tangential distortion

$$\begin{pmatrix} \tilde{u}_d \\ \tilde{v}_d \end{pmatrix} = \begin{pmatrix} \tilde{u}_t + \tilde{u}_r \\ \tilde{v}_t + \tilde{v}_r \end{pmatrix}. \tag{2.29}$$

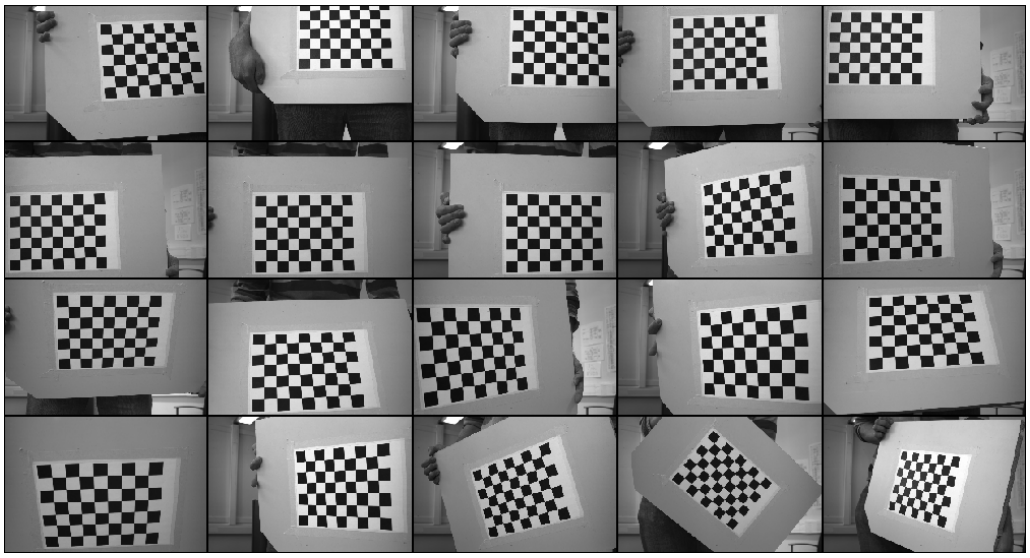

Figure 2.6: *Camera calibration images of a checkerboard pattern.*

True pixel coordinates and normalized, distorted coordinates are related to each other through the camera calibration matrix, K, as already mentioned in the section 2.2. In many machine vision applications, the inverse mapping is more useful. The goal is to remove distortions and to get undistorted pixel coordinates. Because of the high degree of the distortion model, there does not exist any general algebraic expression for this inverse mapping. However, there exist many numerical and iterative implementations for that problem. After the undistortion process, the pinhole camera model is valid.

2.6 Camera Calibration

Purpose of the camera calibration is to acquire the intrinsic parameters and the distortion coefficients of the camera. This is usually achieved by taking multiple photographs from known object, e.g. a checkerboard pattern in figure 2.6. From correspondences between multiple images, a mathematical distortion model is fitted and minimized. As a result, the extrinsic and intrinsic parameters of the camera are solved.

There are many commercial and non-commercial softwares for camera calibration, but one common software is the *Camera Calibration Toolbox for MATLAB* ® developed by Bouguet. It is freely available online to MATLAB [3]. Similar camera calibration tool for C can be found from [36]. Camera calibration result in figures 2.6 – 2.8 is retrieved with Bouquet's software. Images were captured with the same high-resolution industrial camera used in the test flight. In each image, a checkerboard pattern is semi-automatically detected at a subpixel accuracy. The extrinsic and intrinsic parameters of the camera are iteratively estimated by minimiz-

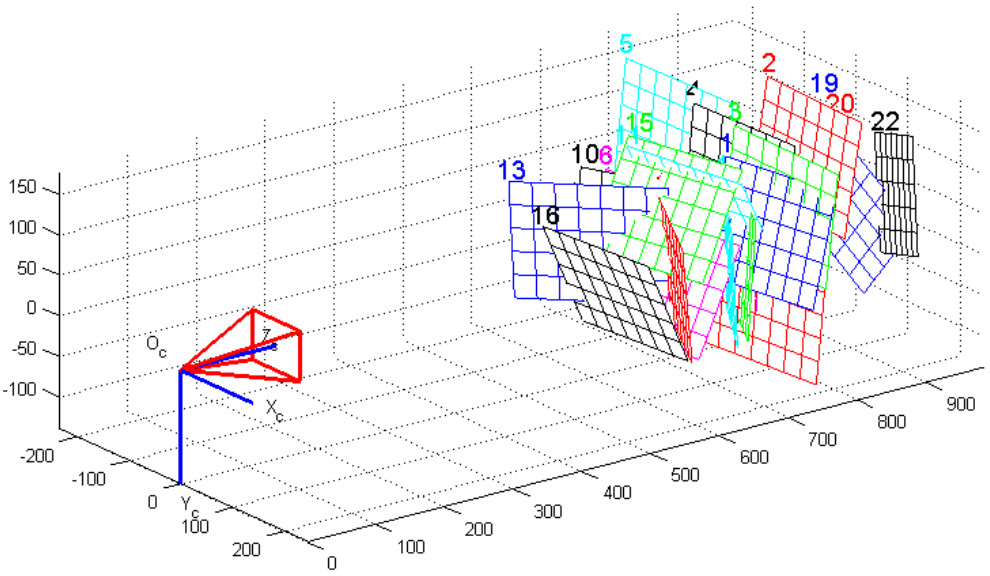

Figure 2.7: *Checkerboard patterns of several calibration images illustrated in a camera-centric coordinate frame.*

ing the back-projection error, which approximates the geometric error. Geometric error is the distance between a measured point and a reprojected point. Other error metrics can also be used. In figure 2.8, the radial distortion of the camera is very noticeable in the periphery of the lens.

Traditionally in photogrammetry and in many computer vision applications, cameras are first calibrated to get the intrinsic parameters of the camera. Now, those parameters can be used for a computer vision application. Nevertheless, that is rather cumbersome process because those parameters vary if a camera is zoomed or focused on a different distance. Therefore, the intrinsic parameters are different depending on the situation. Camera calibration should cover all the possible scenarios. However, calibrated cameras are not necessary because there exist solutions to auto-calibrate cameras on the fly. Unfortunately, those solutions are not completely trustworthy; they can work well in the right circumstances, but used recklessly they will fail. Hartley and Zisserman give several specific recommendations for auto-calibration which can be further read from [18, p. 498].

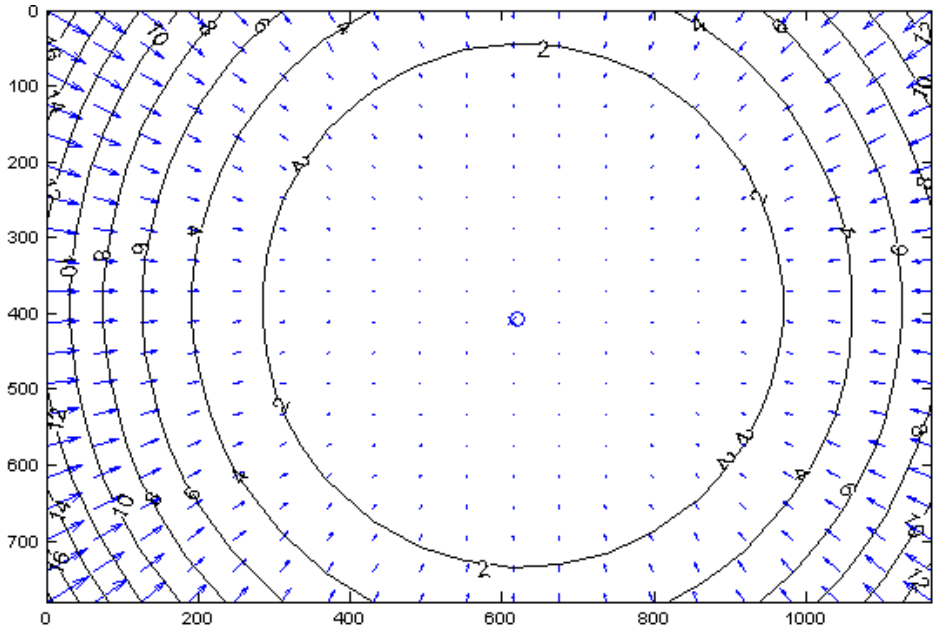

Figure 2.8: *Intrinsic parameters and distortion model of the calibrated camera.*

3. CORRESPONDENCE PROBLEM

Problem in many computer vision applications, such as camera calibration, image registration, or 3D scene reconstruction, is to find correspondences between two or more images of the same scene after the camera has moved or time has elapsed. When dealing with image sequences, these correspondences are usually solved as a pointwise movement of pixels or features from one image plane to another.

Estimation of correspondences can be roughly divided into two major categories: differential and feature-based techniques. Differential techniques are concentrating on first or higher-order partial derivatives of the 2D image signal whereas feature-based techniques usually try to describe the surrounding of an interesting 2D point with a feature vector. Furthermore, these feature vectors of two distinct images are compared, and matches between corresponding points may be found.

There also exist other methods, such as phase correlation, block-based methods, and discrete optimization. Essentially in phase correlation, the relative translation between two similar images are found by using a fast frequency-domain approach. Block-based methods usually minimize the sum of squared differences (SSD) or the sum of absolute differences (SAD). In discrete optimization methods, the optimal solution is often found through min-cut max-flow algorithms, linear programming, or belief propagation. All in all, the division between different methods is sometimes rather artificial because algorithms can exploit the principles of many techniques. When correspondences are needed to be found between two multimodal or dissimilar images, e.g. a real-life aerial image and a basic map, traditional techniques do not usually provide adequate solutions.

3.1 Differential Techniques

An often used approximation of the reflectivity of an unknown surface is the Lambertian reflectance model, which assumes that the apparent brightness of the surface to an observer, e.g. camera, is the same regardless of the angle of view. In other words, the surface luminance is isotropic. It is also a common experience that the apparent brightness of moving objects remains constant [43, p. 192]. The constancy of the image brightness, I, can be expressed as a function of both spatial coordinates and time:

$$\frac{dI(u(t), v(t), t)}{dt} = \frac{\partial I}{\partial u}\frac{du}{dt} + \frac{\partial I}{\partial v}\frac{dv}{dt} + \frac{\partial I}{\partial t} = 0. \tag{3.1}$$

The partial derivatives of the image brightness are essentially the components of the spatial image gradient, ∇I. The temporal derivatives, du/dt and dv/dt, are the components of the motion field, $\mathbf{v}$. Equation (3.1) may be rewritten as the image brightness constancy equation

$$(\nabla I)^{\mathrm{T}}\mathbf{v} + I_t = 0, \tag{3.2}$$

where the subscript t denotes partial differentiation with respect to time.

The apparent motion of the image brightness is almost always different than the true motion field. The approximation of the true motion field is called the optical flow. The motion field is well approximated by a constant vector field, $\mathbf{v}_c$, within any small patch, Q_{NxN}, of the image plane [43]. This assumption holds well if the displacement is rather small, i.e. few pixels. The optical flow can be estimated by minimizing the image brightness constancy equation

$$\Psi[\mathbf{v}_c] = \sum_{\mathbf{p}_i \in Q} [(\nabla I)^{\mathrm{T}}\mathbf{v} + I_t]^2. \tag{3.3}$$

The least-squares solution, provided in [43, p. 196], to the overconstrained system can be obtained as

$$\mathbf{v}_c = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}\mathbf{b}, \tag{3.4}$$

where

$$A = \begin{bmatrix} \nabla I(\mathbf{p}_1) \\ \nabla I(\mathbf{p}_2) \\ \ldots \\ \nabla I(\mathbf{p}_{NxN}) \end{bmatrix} \tag{3.5}$$

and

$$\mathbf{b} = -[\mathrm{I_t}(\mathbf{p}_1), \mathrm{I_t}(\mathbf{p}_2), ..., \mathrm{I_t}(\mathbf{p}_{\mathrm{NxN}})]^{\mathrm{T}}. \tag{3.6}$$

A is a $N^2 \times 2$ matrix of the spatial image gradients evaluated at point $\mathbf{p}_i$, and $\mathbf{b}$ is the N^2-dimensional vector of the partial temporal derivatives of the image brightness evaluated at $\mathbf{p}_i$ after a sign change. A slightly improved solution is to use a weighted least squares algorithm, which gives more importance to the pixels near the center of the patch:

$$\mathbf{v}_w = (A^{\mathrm{T}}w^2A)^{-1}A^{\mathrm{T}}w^2\mathbf{b}, \tag{3.7}$$

where w is a windowing function [43].

A similar use of image gradients and the assumption of a constant displacement for the local patch is the core idea in the famous optical flow method: *Kanade-Lucas-Tomasi* (KLT) feature tracker. KLT trackers are mostly based on two papers originally presented by Lucas-Kanade [26] and Tomasi-Kanade [42]. Naturally, there

have been many improvements compared to the original papers, but in many implementations, the essence of KLT is still used. One implementation can be found from the *Open Source Computer Vision Library* (OpenCV), which is a BSD-licensed library that includes several hundreds of computer vision algorithms (C/C++) originally developed by Intel and supported by Willow Garage [36] [37].

Main idea in KLT feature tracker is to find the displacement of the local patch using *Newton-Raphson method* to minimize the image brightness constancy equation, like in equation (3.3). Overall range of the algorithm may be expanded using smoothing and pyramidical implementation.

Displacement is impossible to find from image areas, where the image gradient is close to zero. This means that the brightness values are almost constant. By the definition, points with high spatial image gradient are the locations at which the motion field can be best estimated by the image brightness constancy equation [43, p. 194]. There exist numerous methods for finding these interesting points, mainly edges and corners. Many of them are utilizing image gradients: [19], [39] and [42].

Structure tensor, C, is a 2×2 matrix derived from gradients of a function. For image intensity function, the structure tensor is

$$C = \sum_u \sum_v w(u,v) \begin{bmatrix} (\nabla I_x)^2 & \nabla I_x \nabla I_y \\ \nabla I_x \nabla I_y & (\nabla I_y)^2 \end{bmatrix}, \tag{3.8}$$

where w is a window function, such as the Gaussian, ∇I_x is the image gradient in x-direction, and ∇I_y is the image gradient in y-direction.

By analyzing the eigenvalues and eigenvectors of the matrix C, the intensity and direction of a feature can be estimated. Following interpretations can be made from the eigenvalues:

1. If $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$, the patch has no feature of interest.

2. If $\lambda_1 \approx 0$ and λ_2 has some large positive value, the patch has an edge-like feature.

3. If both λ_1 and λ_2 has some large positive value, the patch has a corner-like feature.

For a differential algorithm, such as a KLT tracker, it is quite common to use good feature points or patches. However, the information of an interesting point and especially its surrounding is not fully exploited. Moreover, if the displacement between two consecutive images is large or the resolution of the image is large, there can be found more robust algorithms.

3.2 Feature-based Techniques

In computer vision, features have clear, mathematically well-founded definition and location in image space. Features may be whatever interesting information from an image which is relevant for a computational task. The most common features are edges, corners, blobs, T-junctions, and ridges. The search for feature correspondences can be divided into three main steps: feature detection, feature description and feature matching.

such as edges, corners, blobs, T-junctions or ridges,

First, feature points are selected at distinctive locations with an appropriate algorithm, i.e. feature detection. The most valuable property of a feature detection algorithm is its repeatability. It express the reliability of a detector for finding the same physical interest point under different viewing conditions, such as rotation and scaling [1].

Next, the surrounding of every feature point is represented by a feature vector, i.e. feature description. Descriptors have to be distinctive and robust to noise, detection displacements, and geometric deformation [1]. Typically, feature descriptors take advantage of local histograms and orientation and magnitude of a feature point.

Finally, feature vectors are matched between different images, i.e. feature matching. The similarity of feature vectors are often measured by the Euclidean or Mahalanobis distance. The dimension of the feature vector has a direct impact on the computation time. Higher dimensions generally offer more robustness and distinctiveness to feature matching. However, dimensioning is more or less balancing between accuracy and fast performance. All in all, there have been a variety of feature detection and description algorithms in the literature, and some of them offer more robustness or speed over each other.

Feature point detection algorithms can exploit the eigenvalues of the structure tensor as discussed in the previous section, and probably one of the most used feature point detector is the Harris corner detector [19]. Instead of using corner detection algorithms, it is also common to use larger regions to retrieve features as blob-like structures, i.e. blob detection. Blob detection algorithms usually exploit expressions of image derivatives. Three main type of differential blob detection methods are *Laplacian of Gaussian* (LoG), *Difference of Gaussians* (DoG), and *Determinant of Hessian* (DoH).

In LoG, input image, $I(u, v)$, is convolved by a Gaussian kernel

$$L(u, v, \sigma) = \frac{1}{2\pi\sigma} \exp^{-(u^2+v^2)/(2\sigma)} * I(u, v) \tag{3.9}$$

whereafter the Laplacian operator is computed

$$\nabla^2 L(u, v, \sigma) = L_{uu} + L_{vv}, \tag{3.10}$$

where L_{uu} is the convolution of the Gaussian second order derivative with the image in point (u, v) in xx-direction and L_{vv} is similarly in yy-direction [24]. This is a single-scale representation, and adjusting the size of the Gaussian kernel, σ, a blob may be detected with its own characteristic scale. However, a multi-scale blob detection with automatic scale selection is more useful, and it can be achieved with the scale-normalized Laplacian operator

$$\nabla^2_{norm} L(u, v, \sigma) = \sigma(L_{uu} + L_{vv}). \tag{3.11}$$

Detection of feature points and scales can be achieved when the scale-normalized Laplacian operator is simultaneously local extremum with respect to both scale and space [24].

DoG is essentially an approximation of the Laplacian operator

$$\nabla^2_{norm} L(u, v, \sigma) \approx \frac{[g(u, v, k\sigma) - g(u, v, \sigma)] * I(u, v)}{k - 1}, \tag{3.12}$$

where k is an appropriate constant between two consecutive Gaussian kernels. It has been shown that $k = 1.6$ is a balance between bandwidth and sensitivity [30]. Similarly, as in LoG, blobs can be detected from local scale-space extremum of multi-scale representation of Difference of Gaussians.

Another widely used method for blob detection is to use the determinant of the normalized Hessian matrix

$$det(H_{norm} L(u, v, \sigma)) = \sigma^2(L_{uu} L_{vv} - L^2_{uv}). \tag{3.13}$$

Blob is detected when the determinant of the normalized Hessian matrix is maximized [24]. The trace of the normalized Hessian matrix is same as LoG, but DoH generally offers better scale selection properties and fires less on elongated, ill-localized structures [1].

Once feature points are detected with an appropriate feature detection algorithm, they need to be described that they can be distinguished from each other. There has been a numerous amount of feature descriptors, such as Gaussian derivatives, moment invariants, complex features, steerable filters, or phase-based local features. However, *Scale-Invariant Features Transform* (SIFT), introduced by Lowe [25], has proved to outperform others [33], although, more recent algorithms are exploiting and further developing powerful local descriptors, which are inspired by SIFT. It is

also shown that robust region-based descriptors perform slightly, but systematically, better than point-wise descriptors [33].

In general, SIFT is a feature recognition algorithm for feature detection, description, and matching. It is invariant to translation, rotation, and scaling, and partially invariant to illumination changes and projective transformations. SIFT uses DoG for fast scale-space detection of blob-like structures. For feature description, SIFT utilizes a histogram of local oriented gradients around the feature point, originally a 16×16 window resampled with the scale obtained from DoG. To achieve rotation invariance, the main direction of the feature point is first calculated. Gradients, in smaller 4×4 windows, are compared to the main direction, weighted with a Gaussian window, and sorted to 8 different direction bins. In the end, there is a 128 long feature vector for each detected feature point. Finally, feature vectors between two different images are matched to each other using nearest-neighbor approach and an estimate of affine transformation model. [25]

Partly inspired by SIFT, Bay et al. presented a novel scale- and rotation invariant detector and descriptor called *Speeded Up Robust Features* (SURF) [1]. Originally, it is faster than any previously proposed algorithms approximating and even outperforming them in repeatability, distinctiveness, and robustness. The main idea in SURF is to use integral images for image convolutions in both feature detection and description. This drastically reduces the computation time. For feature detection, SUFR uses a Hessian matrix based blob detection for both scale and space selection. For feature description SURF utilizes the first order Haar wavelet responses rather than image gradients, like SIFT. Although, feature vectors are quite similarly constructed from locally oriented spatial distribution of Haar wavelet responses. However, SURF integrates the gradient information within a subpatch whereas SIFT depends on orientations of the individual gradients. This makes SURF less sensitive to noise. Furthermore, the matching step is also speeded up with the contrast information retrieved from the trace of the Hessian matrix, so that blobs which have the same contrast are matched to each other.

The original implementation of the SURF algorithm is downloadable from the author's website [2]. In addition, there exist numerous implementations in all major programming languages, including C, C++, Python, Java and MATLAB. SURF is also a part of OpenCV 2.0. SURF algorithm is also used in this thesis to find correspondences between consecutive aerial images.

Another similar algorithm can be found from the *Library for Visual Odometry* 2 (LIBVISO). It it a fast cross-platform C++ Library with MATLAB wrappers for computing the 6DoF motion of a moving mono and stereo camera. In this thesis, LIBVISO2 algorithms are only used for correspondence search because it was originally developed for vehicular motion estimation assuming that the height of the

vehicle from the ground is known and fixed. For correspondence search, LIBVISO2 uses similar approach than SURF, but some simplifications are made to establish computationally more efficient algorithm for real-time use. In LIBVISO2, features are detected with a 5×5 blob and corner detection filter. Furthermore, features are described with quantized 11×11 Sobel filter responses, which are matched to each other using SAD error metric. LIBVISO2 also offers a set of parameters to tune the algorithm, such as minimum distance between features, maximum radius between corresponding points, and subpixel accuracy. The correspondence search algorithm of the LIBVISO2 library is also used in this thesis for a comparison to SURF. Pose estimation performance and analysis of both of these algorithms are shown in chapter 5.

3.3 Higher Level Correspondences

Image registration may need to be done between images, which only contain minutely mutual information. This may happen when an aerial image needs to be registered to some reference images, which may be georeferenced to provide additional information from the scene. Georeferencing means that image row and column coordinates are mapped to some local or global coordinates. This enables absolute pose estimation in known environment because registered image pixels would now have coordinates in known reference coordinate system.

In some cases, it is almost impossible to find a direct pointwise correspondence between two images with neither differential nor feature-based techniques if images are multimodal or texturally very different. Then, it is feasible to use higher level features or object recognition. Generally, higher level image registration requires careful preprocessing and feature selection in order to extract the available mutual information from two dissimilar images. These methods can be very application dependent and are not discussed here in detail. For example, some higher level image registration methods can be found from [11], [17] and [31].

In this thesis, higher level image registration and therefore absolute pose estimation are performed semi-automatically. Aerial and reference images are registered manually, but the position and orientation are estimated with the algorithms presented in the next chapter.

4. CAMERA POSE ESTIMATION

Once correspondences for 2D image pixels are found, camera pose estimation is used to solve the translation and rotation of a camera. Generally, pose can be estimated relatively between two frames or absolutely from a single image frame. Relative pose estimation means that there is no local or global 3D information available from detected 2D image points, i.e. unknown environment, whereas absolute pose estimation needs that 3D information about the environment, i.e. known environment.

In aerial applications, when flying at medium or high altitude, it is possible to use the planarity assumption for the surface of the earth because the relative height of the 3D points diminishes proportionally to the distance of the camera from the surface. This assumption may reduce one dimension from calculations, but it can cause ambiguities, which may be only resolved by introducing additional information about the scene.

One of the most fundamental sources of theory and computational methods related to geometrical image formation of a mono, stereo, and multiple cameras are found in [18]. This chapter covers essential methods for camera pose estimation in the context of aerial applications.

4.1 Relative Pose Estimation

Relative pose is calculated from correspondences between two consecutive image frames. Methods for solving the correspondence problem are introduced in chapter 3.

It has been stated that in general motion the fundamental matrix can be uniquely determined from point correspondences, and that also the two camera matrices uniquely determine the fundamental matrix, as in equation (2.17). However, this mapping is not injective since pairs of camera matrices that differ by a projective transformation give rise to the same fundamental matrix [18, p. 254]. This means that camera matrices can be retrieved from the fundamental matrix up to a projective ambiguity in uncalibrated cases.

In case of calibrated cameras and normalized image coordinates, the essential matrix is the specialization of the fundamental matrix. From the essential matrix camera matrices can be retrieved up to a four-fold ambiguity and an overall scale ambiguity, which cannot be determined without extra information. In general motion, the essential matrix can have four possible solutions because of the two possible

choices of the rotation and two possible signs of the translation [18, p. 259]. Suppose that the SVD of $E = Udiag(1,1,0)V^{\mathrm{T}}$ and the first camera matrix $P = [I|0]$, possible solutions for the second camera matrix are

$$P' = [USV^{\mathrm{T}}|+u_3] \; or \; [USV^{\mathrm{T}}|-u_3] \; or \; [US^{\mathrm{T}}V^{\mathrm{T}}|+u_3] \; or \; [US^{\mathrm{T}}V^{\mathrm{T}}|-u_3], \quad (4.1)$$

where S is a skew symmetric matrix $\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and u_3 is the last column of U [18, p. 259]. However, only in one solution a reconstructed point will be in front of both cameras. Thus, it is appropriate to check the right solution with only a single point.

Relative Pose from Planar Homography

For medium and high altitude aerial vehicles, small differences in topography compared to the height of the vehicle can be neglected with the planarity assumption. Planar homography is illustrated in the picture 4.1. Suppose that the first view has a camera matrix $P = K[I|0]$ and a second view has a camera matrix $P' = K[R|t]$. Thus, the mapping between two consecutive image frames and their corresponding points is a planar homography between two views of the same scene as follows:

$$\mathbf{x}_2 = H\mathbf{x}_1 = KR(I - Tn_1^{\mathrm{T}}/d_1)K^{-1}\mathbf{x}_1, \quad (4.2)$$

where n_1 is the unitary plane normal vector expressed in the coordinate frame of the first camera and d_1 is the euclidean distance from the plane to the first camera [6]. A solution for homography estimation is given in section4.2. For calibrated cameras, the calibrated homography is defined as

$$H_c = K^{-1}HK = R(I - Tn_1^{\mathrm{T}}/d_1). \quad (4.3)$$

Rotation and translation, up to a scale factor, can be retrieved using SVD. SVD of the calibrated homography is

$$H_c = UDV^{\mathrm{T}} = Udiag(\lambda_1, \lambda_2, \lambda_3)V^{\mathrm{T}}, \quad (4.4)$$

where $\lambda_1 > \lambda_2 > \lambda_3$. Rotation, translation and the plane normal is then

$$R = U \begin{bmatrix} \alpha & 0 & \beta \\ 0 & 1 & 0 \\ -s\beta & 0 & s\alpha \end{bmatrix} V^{\mathrm{T}} \quad (4.5)$$

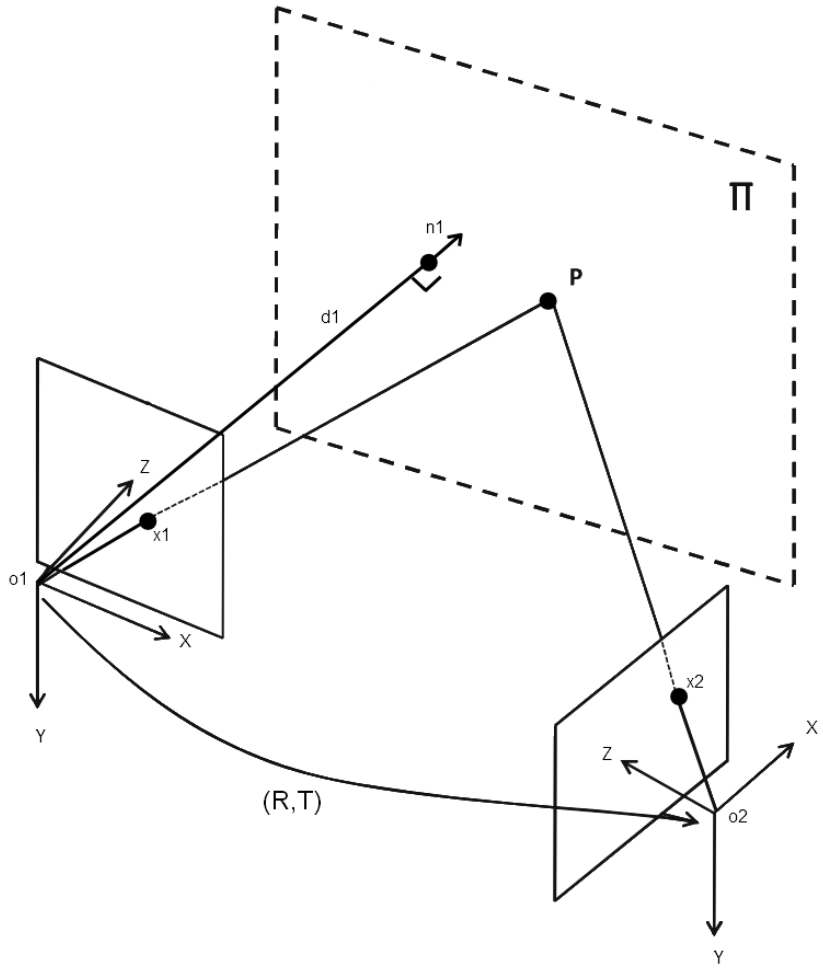

Figure 4.1: *Planar homography between two consecutive views of the same scene point P. Image is retrieved and edited from [6].*

$$T = \frac{1}{\omega}(-\beta u_1 + (\frac{\lambda_3}{\lambda_2} - s\alpha)u_3) \tag{4.6}$$

$$n_1 = \omega(\delta v_1 + v_3) \tag{4.7}$$

where

$$\delta = \pm\sqrt{\frac{\lambda_1^2 - \lambda_2^2}{\lambda_2^2 - \lambda_3^2}} \tag{4.8}$$

$$\alpha = \frac{\lambda_1 + s\lambda_3\delta^2}{\lambda_2(1+\delta^2)} \tag{4.9}$$

$$\beta = \pm\sqrt{1-\alpha^2} \tag{4.10}$$

$$s = det(U)det(V) \tag{4.11}$$

and ω is a scale factor [44]. In general, there are two possible solutions for rotation, translation, and plane normal, and each solution must accomplish that $sgn(\beta) = -sgn(\delta)$.

The correct solution can be disambiguated using a third frame compared to the first frame because the plane normal, n_1, should be the same in both cases. It is common to set the scale factor so that $\|n_1\| = 1$. Nevertheless, only the product $\frac{\|t\|}{d_1}$ can be recovered. The overall scale factor for translation is solved only when the distance to the ground plane, d_1, is known. In aerial applications, this information can be retrieved from a height sensor, such as a barometric sensor or a laser altimeter. The translation scale or the distance to the ground plane can also be estimated from an airspeed sensor.

4.2 Absolute Pose Estimation

In order to avoid drift, which continuously degrades orientation and position estimation, there is a need for absolute pose estimation, which is not dependent on GNSS. Generally in flight control, orientation estimation is more important than position estimation because it assures the stability of the vehicle. Position and trajectory estimation is needed for navigational purposes.

Absolute pose estimation can be done, when 2D image pixels are registered to georeferenced images. Furthermore, if camera is calibrated and image is undistorted with known distortion parameters, the camera pinhole model is valid between image pixels and 3D-coordinates. From these correspondences, with appropriate methods, camera pose to known coordinate system can be retrieved. Absolute pose can only be estimated in known environment, but absolute orientation can be estimated from other visual cues, which are closely related to the direction of gravity.

Absolute Pose from Planar Homography

Equation (2.6) links 3D world coordinates to image pixels through the camera pinhole model. In case of medium or high altitude flying vehicles, it is possible to use the planarity assumption, i.e. $Z = 0$, for the surface of the earth. In this case the camera pinhole model from the equation 2.7 reduces to

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = H \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}, \tag{4.12}$$

where H is perspective projection or homography between the image plane and the ground plane.

Homography between two planes can be retrieved using the *Direct Linear Transformation* (DLT) algorithm. A linear set of equations for multiple correspondences can be written as

$$Ah = \begin{bmatrix} 0 & 0 & 0 & -X_1 & -Y_1 & -1 & X_1y_1 & Y_1y_1 & y_1 \\ X_1 & Y_1 & 1 & 0 & 0 & 0 & -X_1x_1 & -Y_1y_1 & -x_1 \\ -X_1y_1 & -Y_1y_1 & -y_1 & X_1x_1 & Y_1x_1 & x_1 & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & -X_n & -Y_n & -1 & X_ny_n & Y_ny_n & y_n \\ X_n & Y_n & 1 & 0 & 0 & 0 & -X_nx_n & -Y_ny_n & -x_n \\ -X_ny_n & -Y_ny_n & -y_n & X_nx_n & Y_nx_n & x_n & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} h_1 \\ ... \\ h_9 \end{pmatrix} = 0. \tag{4.13}$$

Four correspondences are enough to solve this system of equations uniquely. As mentioned earlier, in case of noisy correspondences, homography is estimated by minimizing the error in a least-square sense. This can be done with (SVD). Solution for h_i is the unitary singular vector corresponding to the smallest singular values of A, i.e. last column of matrix V in SVD.

When a camera is calibrated, estimated homography can be multiplied with the inverse of the calibration matrix as follows

$$K^{-1}H = K^{-1}K\omega[R|T] = \lambda \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} = H_c, \tag{4.14}$$

where ω is the overall scale ambiguity and H_c is known as the calibrated homography. Rotation, translation and the overall scale ambiguity can be estimated by minimizing the Frobenius matrix norm $\|\cdot\|_F$. The problem may be formulated as:

$$min_{R,T,\omega} \left\| 1/\omega H_c - R \begin{pmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & T_z \end{pmatrix} \right\|_F^2 \tag{4.15}$$

subject to $R^TR = I$. The optimal solution for the rotation R can be obtained independently by solving the the following subproblem:

$$min_{\bar{R}} \left\| \bar{H}_c - \bar{R} \right\|_F^2, \tag{4.16}$$

subject to $\bar{R}^T\bar{R} = I_2$, where $\bar{R}$ and $\bar{H}_c$ are the two first columns of R and H_c. Above problem may be solved using SVD. Let $\bar{H}_c = USV^T$ be the SVD of $\bar{H}_c$. The optimal

solution for amputated rotation $\bar{R}$ is then:

$$\bar{R} = UV^T. \tag{4.17}$$

The third column of the rotation matrix R can be calculated from the cross product of the first and the second column. [41]

The optimal scale factor ω is obtained as:

$$\omega = \frac{trace(\bar{R}^T \bar{H}_c)}{trace(\bar{H}_c{}^T \bar{H}_c)}. \tag{4.18}$$

Finally the translation can be retrieved from the third column of the calibrated homography divided by the scale factor: $T = \frac{H_{c3}}{\omega}$. [41]

It is also very common to use these solutions as an initial guess for some iterative optimization algorithm, such as the *Newton-Raphson method*. Rotation matrix and translation vector are usually parametrized as six variables for minimization the back-projection error. In this thesis, the *Levenberg-Marquardt algorithm* (LMA) is used for optimization. LMA can be found from the *Optimization Toolbox* ^{TM} for MATLAB.

Attitude from Horizon

In aerospace engineering, attitude is a vehicle's orientation about its center of mass. Attitude is defined with three angles: yaw, pitch and roll. Pitch is the angle which increases or decreases the lift generated by the wings. Roll is the angle which change the horizontal direction of flight. Yaw is the rotation about vehicle's vertical axis. For vehicle's stability in flight, roll and pitch angle play very important role, whereas yaw angle is more useful for overall trajectory estimation, i.e. navigation.

An infinite scene line is imaged as a line terminating in a vanishing point [18]. All parallel lines meet at the same vanishing point. Similarly an infinite scene plane is a line at infinity, i.e. vanishing line. The planarity or local flat disk assumption for the surface of the earth is a close and effective approximation [12]. It also yields that the curve of the horizon is approximated with a secant. Thus, the visible horizon line uniquely determines the camera attitude, especially roll and pitch angles. Roll angle is simply the inverse tangent of the slope, k,

$$\phi = \arctan(k). \tag{4.19}$$

Pitch angle is dependent on the roll angle, height of the vehicle and the position of the horizon on the image plane [12]. Height of the vehicle diminishes when the

distance to the horizon is significantly greater. Thus, pitch angle is

$$\theta = \arctan(\pm\frac{u\sin(\phi) + v\cos(\phi)}{f}), \tag{4.20}$$

where u and v are coordinates from the principal point in metric scale [12]. Furthermore, pitch angle can be decoupled from roll angle by using circular field of view.

Usually the horizon line is recovered by separating the sky and the ground based on context differences between both regions [22]. For a circular field of view, the line joining the centroids bisects the horizon at a right angle as long as the the horizon makes a straight line in the view [8]. The pitch angle relative to the horizon line is

$$\theta = \arctan(\frac{h - r)}{f}), \tag{4.21}$$

where h is the length of the line segment inside the circular field of view and above the horizon, r is the radius of the circular field of view, and f is the focal length of the camera in meters. The circular field of view is illustrated in the figure 4.2. S is the centroid of the sky, G is the centroid of the ground, and C is the camera center. Unfortunately, horizon may be very difficult to retrieve in dense urban environment or flying low. However, there exist also other gravity-related properties, which have a direct relationship with the attitude.

Attitude from Vanishing Points

Urban environment contains many lines, which are parallel or orthogonal to the direction of gravity. Intersections of parallel lines, i.e. vanishing points, have a direct relationship with the pitch and roll angle of the camera [22]. Scene lines, which are on a same plane, define the vanishing line. If scene lines are on the ground plane and orthogonal to the direction of gravity, the vanishing line is the same as the horizon line. The horizon line can be retrieved from the cross product of two horizontal vanishing points.

On the contrary, vertical lines, such as building edges, are parallel to the direction of gravity, and they uniquely determine the vertical vanishing point. In case of calibrated cameras, the vertical vanishing point, v_v , and possibly multiple horizontal vanishing points, v_h, have the following geometric constraint:

$$v_v^{\mathrm{T}} v_h^i = 0, \tag{4.22}$$

where $i = 1, ..., n - 1$. The horizon is equivalent to the vertical vanishing point in an image because the horizon is the projection of the ground plane. The roll and

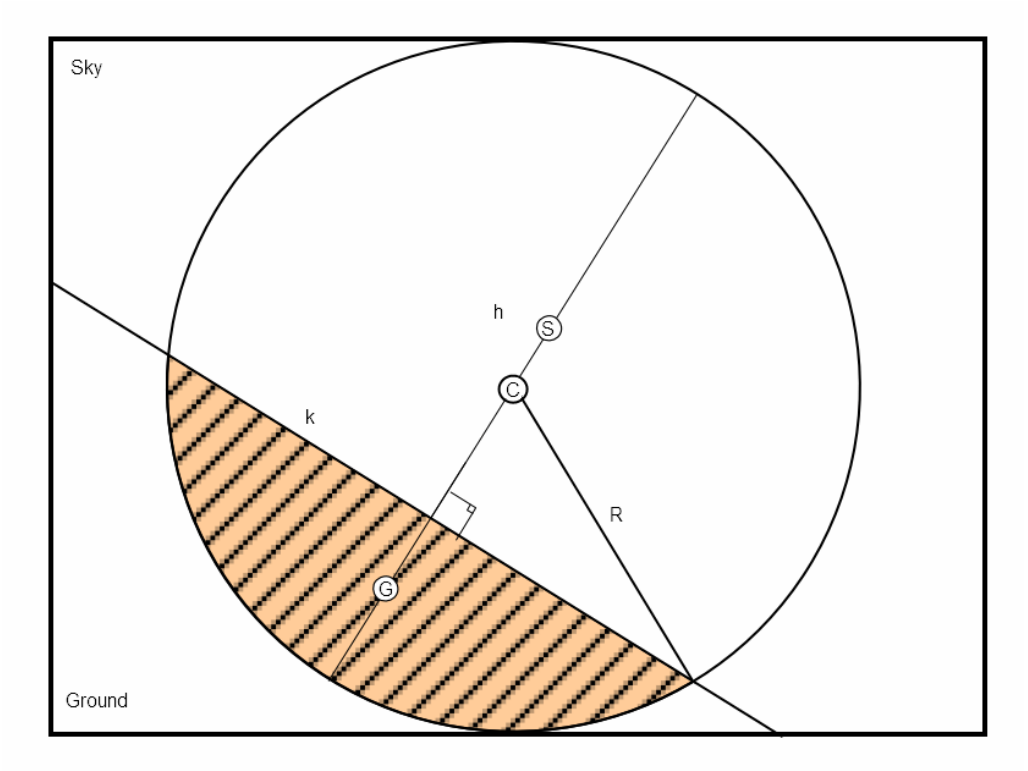

Figure 4.2: *Visualization of the circular field of view of the camera.*

the pitch angle can be derived directly from the vertical vanishing point

$$\phi = \arctan 2(v_x, v_y), \theta = \arctan(\frac{1}{\sqrt{v_x^2 + v_y^2}}) \tag{4.23}$$

where $v_v = [v_x \ v_y]$ and $\arctan 2$ is the two-argument variation of the arctangent function. [22]

4.3 Sensor Fusion

When additional sensors are provided, such as barometric sensor, laser altimeter, or INS, it is feasible to gather all the possible estimates of vehicle's position and orientation to a state estimation algorithm, such as the Kalman filter, to estimate the best possible output from noisy inputs. Kalman-based filters are common in variety of technologies and widely used for guidance, navigation, and control of vehicles, particularly in aircrafts and spacecrafts.

In theory, the Kalman filter is a linear quadratic estimator, which models a linear dynamical system with a Bayesian model and assumes that all the error terms have a Gaussian distribution [46]. The Kalman filter projects the current state and error covariance estimates to obtain a priori estimates for the next time step. New measurement is incorporated into a priori estimate to obtain an improved a posteriori

estimate. The Kalman filter resembles a predictor-corrector algorithm for solving numerical problems. However, in many cases, the vision-based navigation problem is non-linear and non-Gaussian. Therefore, solutions based only on a Kalman filter may not be applied. There are also variety of extensions to non-linear systems, such as the *Extended Kalman filter* (EKF) or the *Unscented Kalman filter* (UKF). Sensor fusion is not applied in this thesis, and further readings about some sensor fusion algorithms and applications to image registration and pose estimation can be found from [6], [7] and [22].

5. POSE ESTIMATION EXPERIMENTS

Camera pose estimation experiments are executed in a typical finnish environment with algorithms described earlier in this thesis. The main emphasis is on relative pose estimation, but absolute pose estimation is also demonstrated. Information from other sensors are utilized for solving the overall scale ambiguities and for reference information. Furthermore, error analysis for both relative and absolute pose estimation is also implemented.

5.1 Test Data

The test flight was performed with a small fixed wing aircraft. Global coordinates of the aircraft were measured with a GPS sensor. The *World Geodetic System* (WGS) defines the reference frame for the Earth and is highly used in geodesy and navigation. The latest revision is WGS 84, and it is also the reference coordinate system used in GPS.

Aerial images were gathered during the flight with a high resolution industrial camera using frame rate of 1 Hz. Figure 5.1 represents a sample image from the flight. As can be seen, the weather was sunny and clear during the flight, although, during the winter sun casts long shadows from forest.

Data was recorded from the whole flight, but for the experiment demonstrated in this thesis, only 100 consecutive images and appropriate GPS waypoints are used. The total trajectory of the test sequence is about 4500 meters and 1 minute 40 seconds long. Unfortunately, the instrumentation of the test flight does not offer reference orientation information for pitch and roll angles. Heading angle, calculated from the GPS waypoints, is proportional to yaw angle. However, the GPS waypoints allow the trajectory of the vehicle to be used in another application, where the camera can be situated and orientated arbitrarily.

Virtual Images

Google Earth is a geographical information program offering a virtual globe map. It is possible to input trajectories with *Keyhole Markup Language* (KML). KML is a file format used in Google Earth and Google Maps to express geographical annotation and visualization.

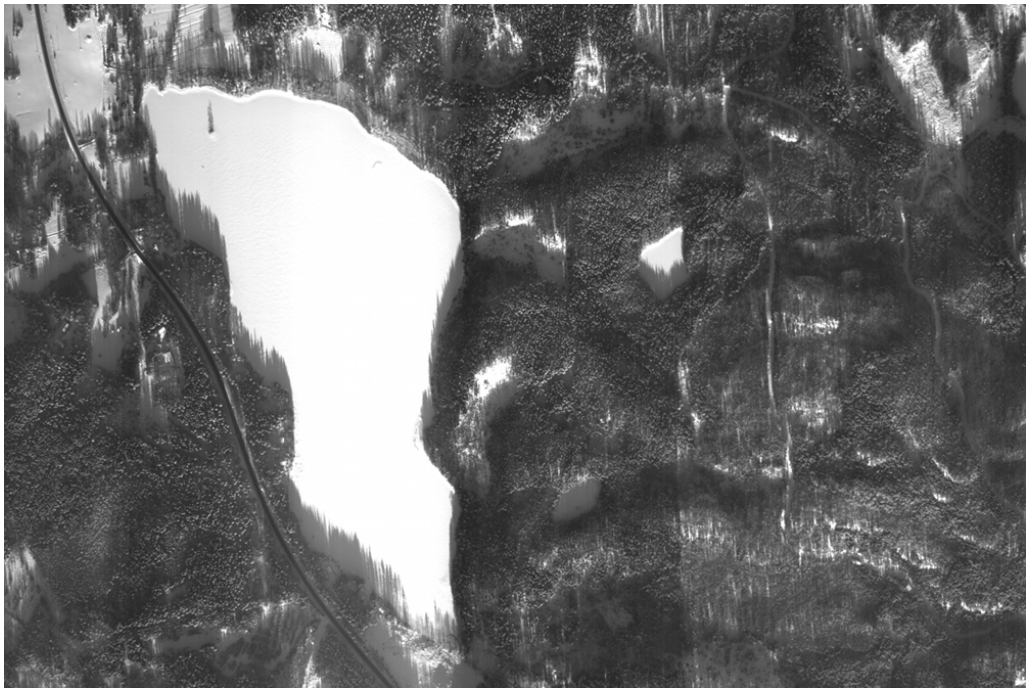

Figure 5.1: *An aerial image retrieved from the test flight.*

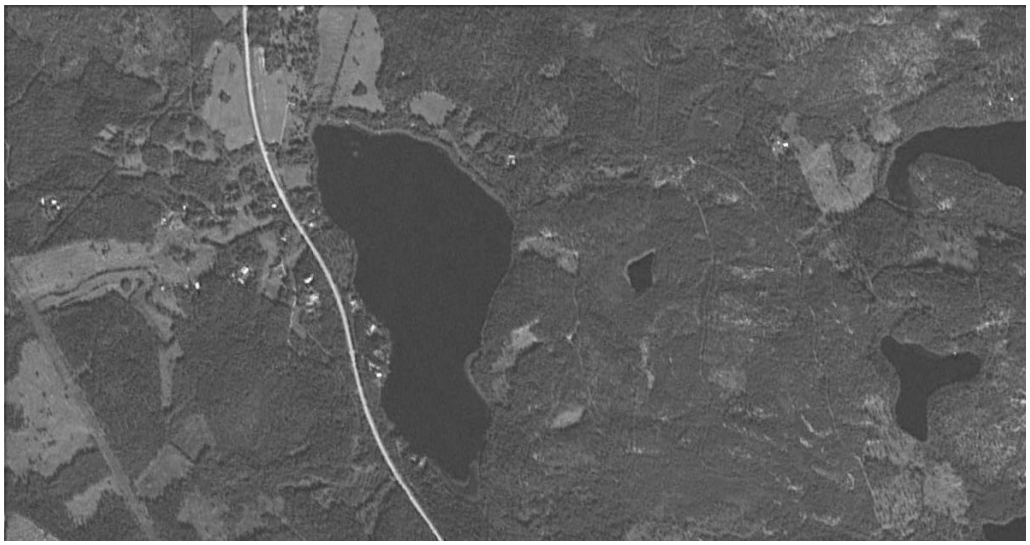

Figure 5.2: *A virtual aerial image retrieved from Google Earth.*

WGS 84 data can be translated to .kml file for Google Earth, and camera can be rotated to desired orientation with respect to world coordinates. It is also possible to capture the virtual image from Google Earth with known reference coordinate and orientation. Furthermore, position and orientation information can now be used to analyze the performance of the pose estimation algorithms. Figure 5.2 represents a sample image retrieved from Google Earth.

In this experiment, the GPS waypoints of the test fligt are used as a trajectory for virtual Google Earth images. Heading information is used as a yaw angle for the virtual camera. Pitch and roll angle are set to zero responding to nadir view, although, that is not assumed in pose estimation algorithms.

5.2 Correspondence Search

Correspondences between two consecutive images are estimated with two algorithms, which were introduced in the section 3.2: SURF and LIBVISO2. Both of the algorithms are robust feature detectors, which mainly rely on blob or corner detection and efficient local feature description and matching. In addition, both algorithms offer feature detection with subpixel accuracy.

There are four different scenarios for the relative pose estimation experiments: virtual images with SURF, virtual images with LIBVISO2, real images with SURF and real images with LIBVISO2.

Figure 5.3 shows a corresponding result of the SURF algorithm between two consecutive Google Earth images from the trajectory of the test flight. Vectors represent the movement of a single feature between two frames. The overall matching performance is around 90% with the following parameters: number of octave layers = 1, number of octaves = 1, hessian threshold = 500 and matching threshold = 0.7. An octave represents a series of filter response maps obtained by convolving the same input image with a filter of increasing size [1]. Hessian threshold is a value for SURF's Hessian-Laplace detector, and matching thresholf is a value for SURF's feature point matching between correspondence points. More detailed description can be found from the original paper [1]. All matched features are verified with a simple outlier rejection. Only 60% of all the features, which have the smallest back-projection error in homography estimation, are selected. Verified features are represented in red, and mismatched or disgarded features are represented in blue. Only verified features are used in further pose estimation.

Figure 5.4 shows a corresponding result with the LIBVISO2 algorithm between two consecutive Google Earth images from the trajectory of the test flight. In a similar manner, vectors represent the movement of a single feature between two frames. Following parameters were used in this scenario: minimum distance between maxima = 30 pixels, peakiness threshold = 50, matching bin height/width = 100,

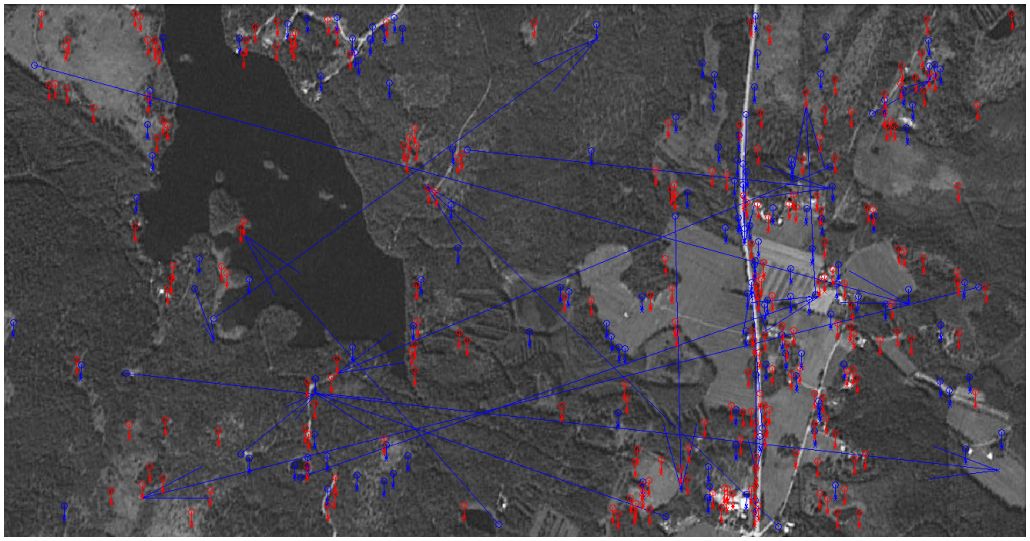

Figure 5.3: *Corresponding points between two virtual images using SURF.*

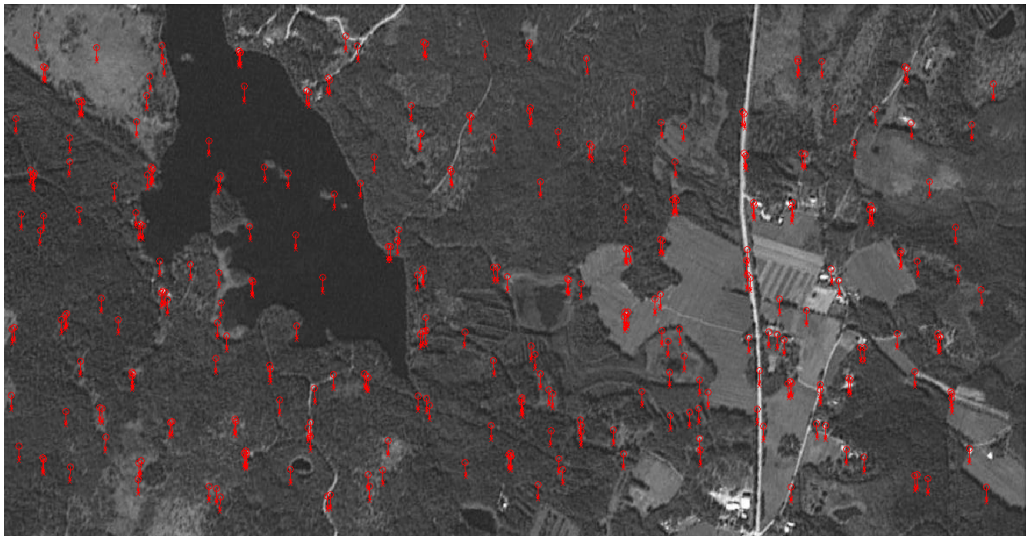

Figure 5.4: *Corresponding points between two virtual images with LIBVISO2.*

matching radius = 80, disparity tolerance = 5 pixels, flow tolerance = 5 pixels, multistage matching = 1 and subpixel accuracy = 1. Parameters for LIBVISO2 are quite self-explanatory, but more detailed description can be found from the original paper [14]. There is a built-in outlier rejection in LIBVISO2. Compared to the SURF algorithm, features are more uniformly distributed over the image plane. This is explained by the different nature of the algorithms. SURF doesn't try to search correspondences uniformly whereas LIBVISO2 does.

Figure 5.5 shows a corresponding result with the SURF algorithm between two consecutive real aerial images from the test flight. Similarly, vectors represent the movement of a single feature between two frames. This time the overall matching performance is close to 100% with the following parameters: number of octave layers = 1, number of octaves = 1, hessian threshold = 1000 and matching threshold = 0.4. All matched features are verified with a simple outlier rejection in homography estimation. Red vectors represent verified features, which will be further used in planar homography composition and pose estimation.

Figure 5.6 shows a corresponding result with the LIBVISO2 algorithm between two consecutive real aerial images from the test flight. Following parameters were used in this scenario: minimum distance between maxima = 30 pixels, peakiness threshold = 50, matching bin height/width = 150, matching radius = 100, disparity tolerance = 5 pixels, flow tolerance = 5 pixels, multistage matching = 1 and subpixel accuracy = 1. Correspondingly, features are more uniformly distributed over the image plane.

Correspondence search in the absolute pose estimation experiment is implemented manually. A sensed aerial image and the reference image are registered to each other by choosing 10 correspondence points by hand.

5.3 Relative Pose from Virtual Images

Virtual Google Earth images from the trajectory of the test flight are used in the relative pose estimation experiment. Relative pose is estimated from correspondences demonstrated in the previous section and with methods described in section 4.1. The first frame of the sequence of 100 frames is set to origo, and all the positions and orientations are calculated in proportion to it. Coordinates of the first frame are assumed to be known in order to solve the scale ambiguity. Furthermore, only information that is known or presumed is the internal camera parameters and the planarity assumption.

Figure 5.7 shows the translation error compared to the GPS reference. Translation is calculated straight from the verified SURF correspondences. X-axis describes the frame number of the whole sequence, and Y-axis is the Euclidean translation error in meters. The translation error is calculated for all the axes X,Y and Z, but

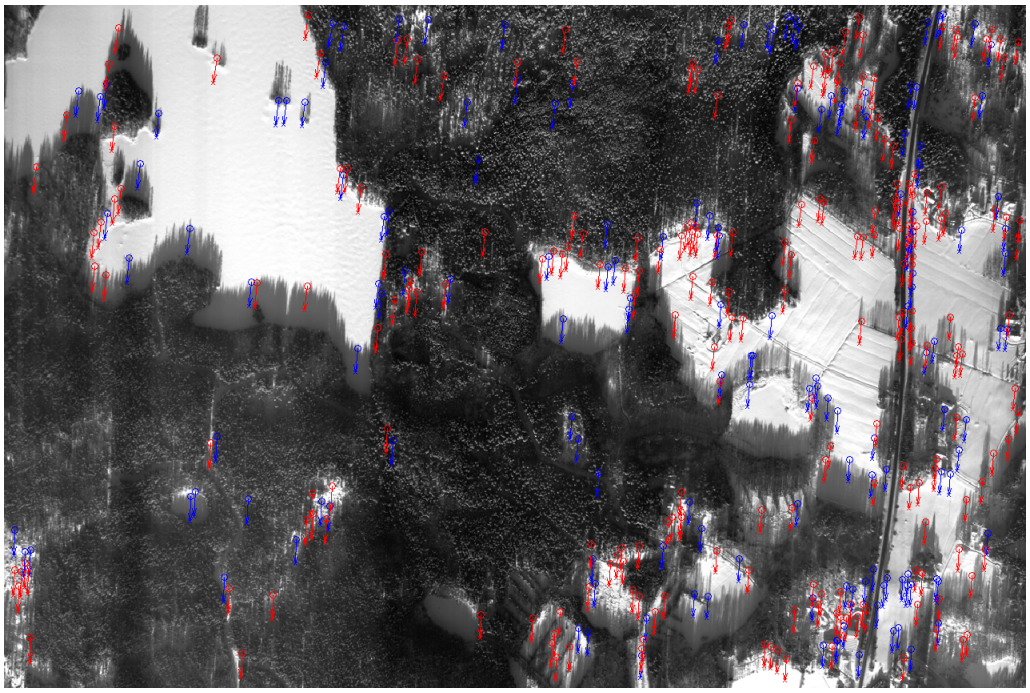

Figure 5.5: *Corresponding points between two real images using SURF.*

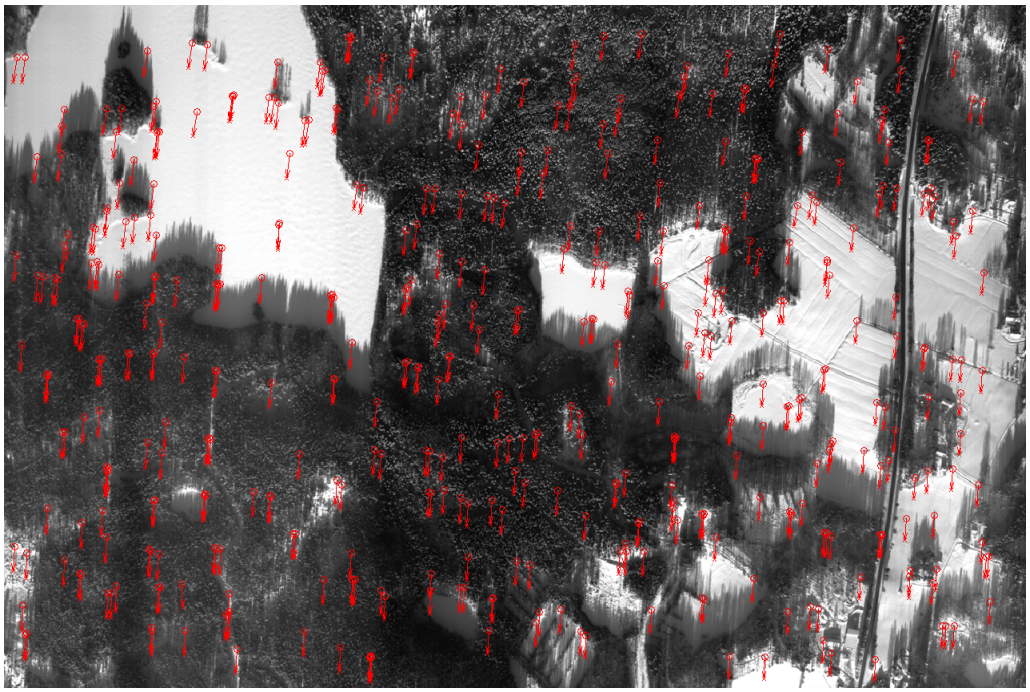

Figure 5.6: *Corresponding points between two real images using LIBVISO2.*

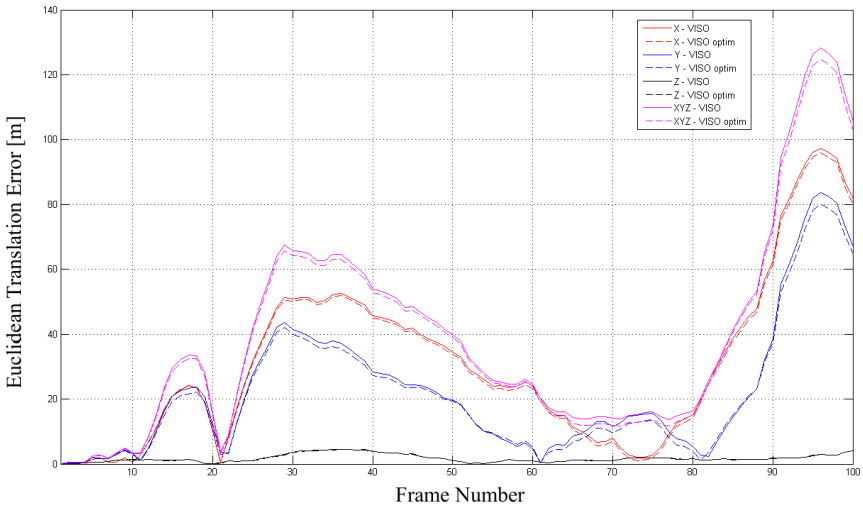

Figure 5.7: *Translation error of relative pose estimation from virtual images using SURF.*

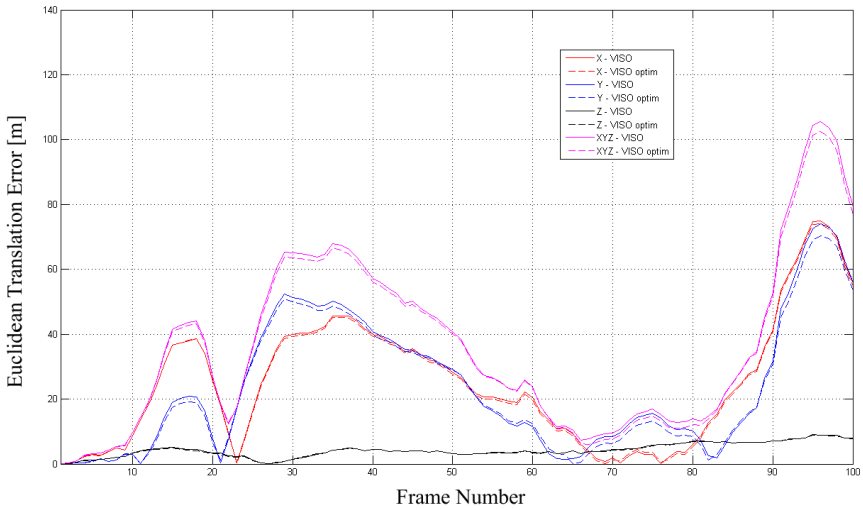

Figure 5.8: *Translation error of relative pose estimation from virtual images using LIBVISO2.*

the total trajectory error is also presented. Dashed curve depicts optimized translation error. Optimization is calculated by minimizing the back-projection error with the LMA.

Figure 5.8 shows the translation error compared to the GPS reference. Translation is calculated straight from LIBVISO2 correspondences. Same notations are used in this figure as in previous figure.

The overall performance in relative translation estimation from corresponding points retrieved either with SURF or LIBVISO2 is close to each other. Correspondences from LIBVISO2 are more uniformly distributed which may cause slight advantage for it. Optimization reduces the overall error only minutely.

Figure 5.9 shows the rotation error compared to the reference roll, pitch and yaw angles. Similarly, rotations are calculated straight from verified SURF correspondences. Horizontal axis describes the frame number of the whole sequence, and vertical axis is the rotation error in degrees. The rotation error is calculated for all the axes X,Y and Z. Dashed curve depicts optimized rotation error. In a similar manner optimization is calculated by minimizing the back-projection error.

Figure 5.10 shows the rotation error compared to reference angles. Rotation is calculated straight from LIBVISO2 correspondences. Same notations are used in this figure as in previous figure.

It can be seen that in rotation estimation correspondences from SURF produces smaller error. This may originate from more accurate feature detection even though both of the algorithms exploit the subpixel accuracy, but in LIBVISO2 there have been made some simplifications to maximize the efficiency of the algorithm.

5.4 Relative Pose from Real Images

Real aerial images from the test flight are utilized in relative pose estimation experiment. Similarly to the previous experiment, relative pose is estimated from feature correspondences between consecutive aerial images. At this time, there is no information for the reference orientation, only GPS heading, which is a quite rough estimation from the GPS waypoints. Coordinates of the first frame are assumed to be known in order to solve the scale ambiguity. Further on, only information that is known or presumed is the internal camera parameters and the planarity assumption.

Figure 5.11 shows the translation error compared to the GPS reference. Translation is calculated straight from the verified SURF correspondences. Horizontal axis describes the frame number of the whole sequence, and vertical axis is the absolute Euclidean translation error in meters. Translation error is calculated for all the axes X,Y and Z, but the total trajectory error is also presented. Dashed curve depicts the optimized translation error. Optimization is calculated by minimizing the back-projection error with the LMA.

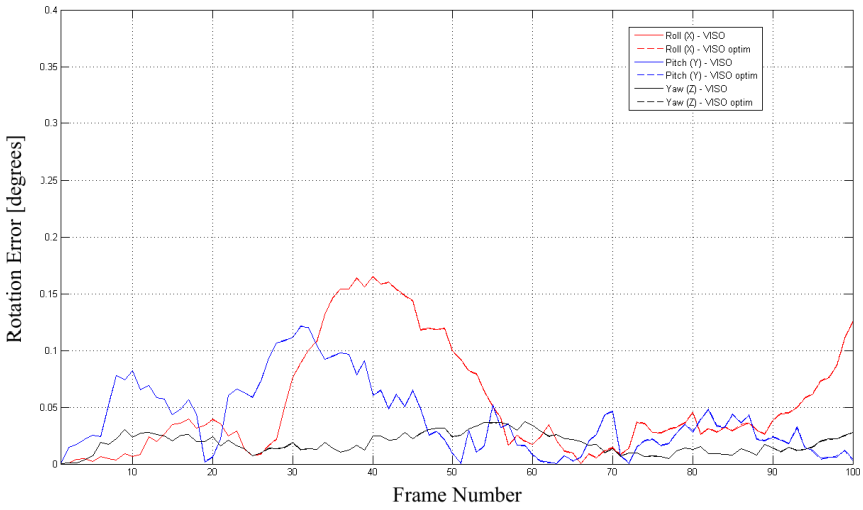

Figure 5.9: *Rotation error of relative pose estimation from virtual images using SURF.*

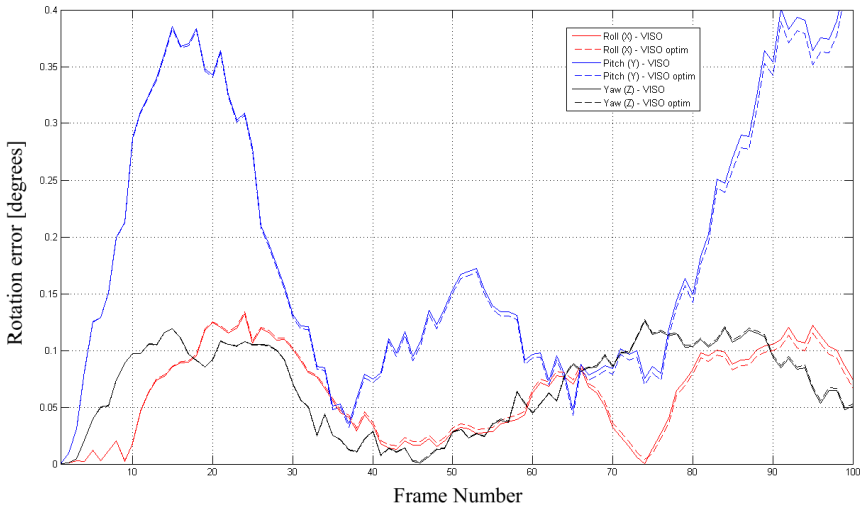

Figure 5.10: *Rotation error of relative pose estimation from virtual images using LIBVISO2.*

Figure 5.12 shows the translation error compared to the GPS reference. Translation is calculated straight from LIBVISO2 correspondences. Same notations are used in this figure as in previous figure.

The overall performance in relative translation estimation is slightly better using corresponding points retrieved from LIBVISO2. Optimization reduces error only minutely. It should be noted that in trajectory estimation the initial orientation is assumed to be nadir view which in reality causes an unknown bias to orientation estimation and therefore the trajectory is a little off already in the beginning.

Figure 5.13 shows calculated rotation angles and GPS heading. Similarly rotations are calculated straight from verified SURF correspondences. Horizontal axis describes the frame number of the whole sequence and vertical axis is the absolute rotation in degrees. Rotation error is calculated for all the axes X,Y and Z. Dashed curve depicts optimized rotation error. In a similar manner optimization is calculated by minimizing the back-projection error.

Figure 5.14 shows calculated rotations from LIBVISO2 correspondences. Same notions than in previous image are used in this figure. The trends of the curves between two different methods are very similar, and it is hard to make any thorough conclusions without proper reference information. In addition, estimated yaw angle in both figures has a very similar trend compared to GPS heading which validates the overall orientation estimation.

5.5 Absolute Pose from Real Images

Absolute pose is estimated from real aerial images using 10 manual correspondences to virtual Google Earth images, which are georeferenced. Georeferencing is done using original GPS position information and the known orientation of a virtual image. In practice, every pixel of the reference image has a coordinate estimate. This is achieved by back-projecting the image plane to the surface of the earth. Again, the planarity assumption becomes viable because the distance to the surface is far greater compared to the relative fluctuation of the topography.

Absolute pose is calculated at the rate of 0.1 Hz, i.e. every tenth frame, with algorithm presented in section 4.2. In between absolute pose estimations, trajectory is calculated with relative pose estimation algorithm. This time only LIBVISO2 correspondences are used.

Figure 5.15 shows the translation error in absolute pose estimation experiment. Horizontal axis describes the frame number of the whole sequence, and vertical axis is the translation error in meters. The red curve depicts the relative translation error calculated from LIBVISO2 correspondences. In the black curve, the absolute pose is calculated only for the first frame. From thereon, trajectory is estimated relatively. While the initial absolute estimate is nearly 100 meters off, the overall

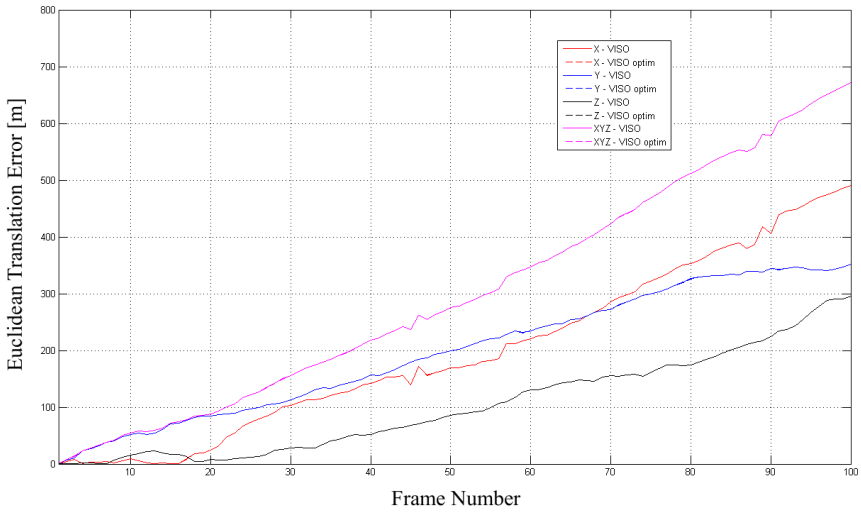

Figure 5.11: *Translation error of relative pose estimation from real images using SURF.*

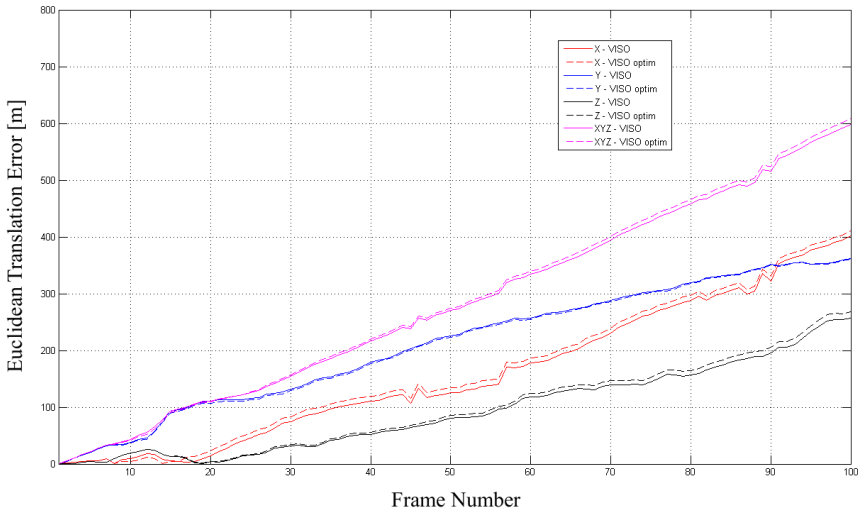

Figure 5.12: *Translation error of relative pose estimation from real images using LIBVISO2.*

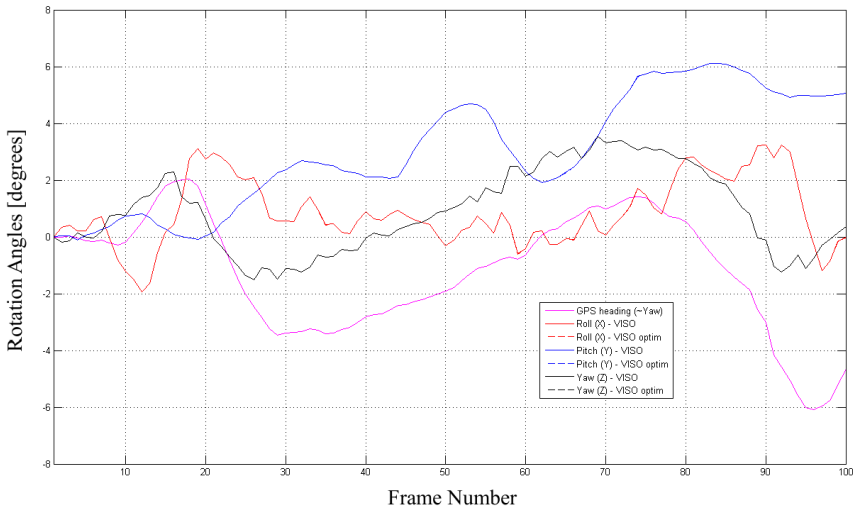

Figure 5.13: *Rotation angles of relative pose estimation from real images using SURF.*

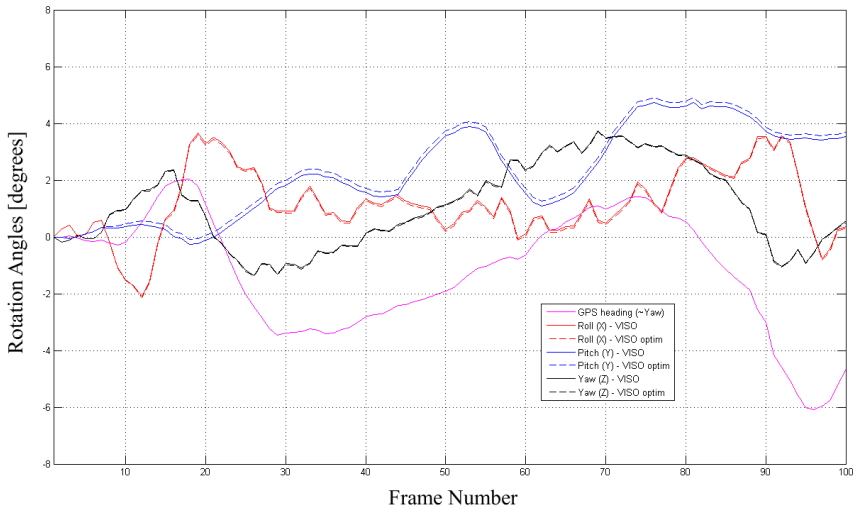

Figure 5.14: *Rotation angles of relative pose estimation from real images using LIBVISO2.*

translation error is smaller than in the red curve. This is due to the unknown initial bias in the orientation of the first frame. Dotted curve is the trajectory error of the combined absolute, 0.1 Hz, and relative pose estimation. Blue dot depicts the moment of absolute pose estimation. This graphs clearly summarizes the problem in relative positioning which was already mentioned in the beginning of the thesis. As the pose is estimated relatively, all the previous errors are accumulated to the following estimations. However, as the graph presents this drift can be compensated off with absolute pose estimation, even though, this experiment uses only 10 feature correspondences. The mean of the absolute pose estimation error in this experiment is about 80 meters, which is around 3% percent from the flying altitude, 2.8 km.

Rotation is also estimated with the same absolute pose estimation algorithm. Figure 5.16 shows the result for absolute rotation estimation. Horizontal axis describes the frame number of the whole sequence and vertical axis is the angle in degrees. The blue curve is the reference GPS heading and the red curve is the relative yaw angle calculated from LIBVISO2 correspondences. Dotted curve is the combined absolute and relative pose estimation result. Blue dots represents a point where pose is estimated absolutely. Even with 10 correspondences absolute orientation estimates settle precisely to the overall trend of the dotted curve. This clearly indicates that there is a bias between the true yaw-angle and GPS heading.

5.6 Error Analysis

The performance of the relative and absolute pose estimation algorithm is analyzed by adding noise to corresponding points, which are randomly and uniformly generated to the image plane. In total 200 corresponding points are generated and pose estimation is repeated 300 times in both cases. Parameters are chosen to correspond to the previous experiments. Following parameters are used: focal length = 50 mm, height of the camera from the ground 2800 m, resolution of the camera = 1280 x 1280, yaw, pitch, and roll angles = 4 degrees, and translation in X,Y, and Z axes = 50 meters. Error analysis is performed with a *standard deviation* (STD) from 0.1 to 6.4 pixels. All cases are summarized in the tables in the end of this section.

In the relative case, camera is first located at the origo and then translated and rotated to the second frame. The true displacements of corresponding points are first calculated and noise is added. From noisy correspondences, translation and rotation are estimated with the relative pose estimation algorithm. Figure 5.17 shows a relative pose estimation result for a relative translation with STD of 1 pixel. X-axis describes the frame number, and Y-axis is the translation.

In the absolute case, camera is translated and rotated with the same parameters. Correspondences are calculated between the 2D surface of the ground plane and the image plane of the camera. Noise is added to the correspondences in the image

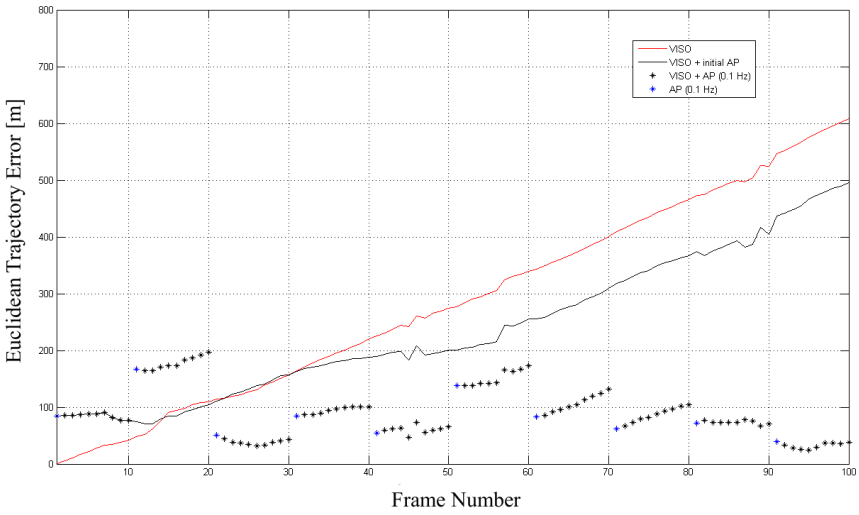

Figure 5.15: *Translation error for relative and absolute pose estimation from real images using LIBVISO2 and manual correspondence search.*

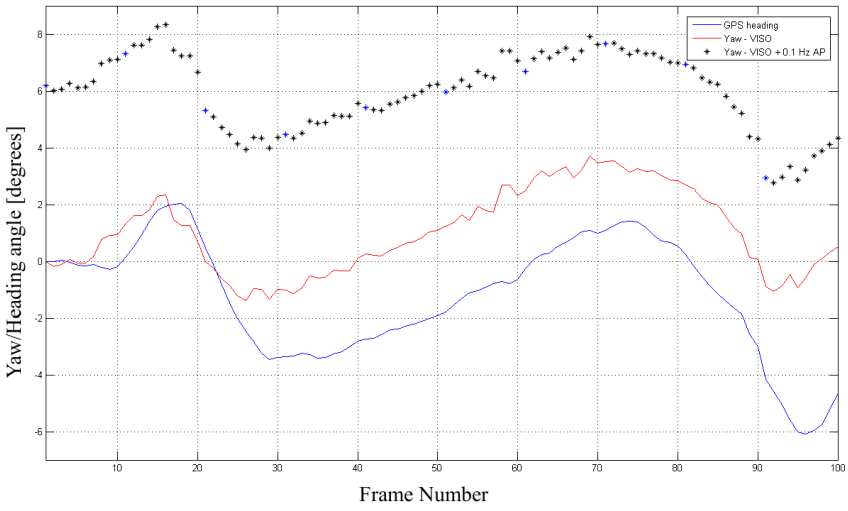

Figure 5.16: *Rotation angles for relative and absolute pose estimation from real images using LIBVISO2 and manual correspondence search.*

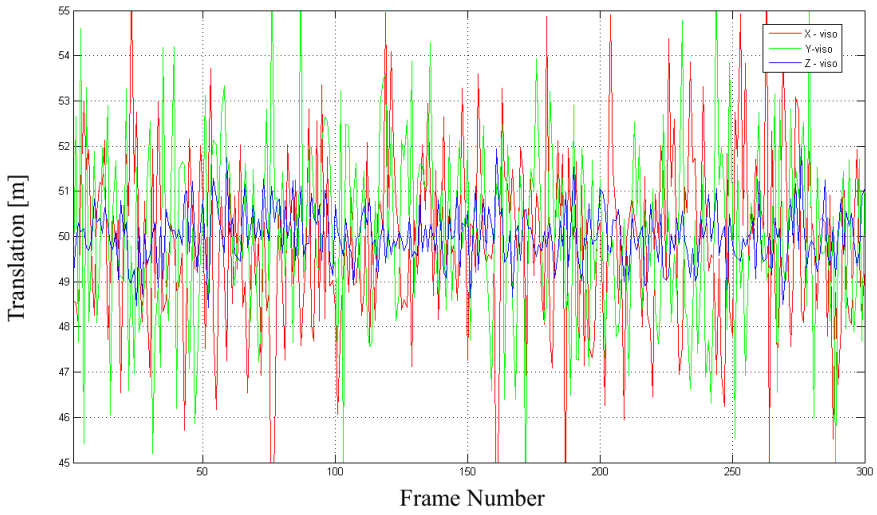

Figure 5.17: *Translation estimations of relative pose estimation with STD of 1 pixel.*

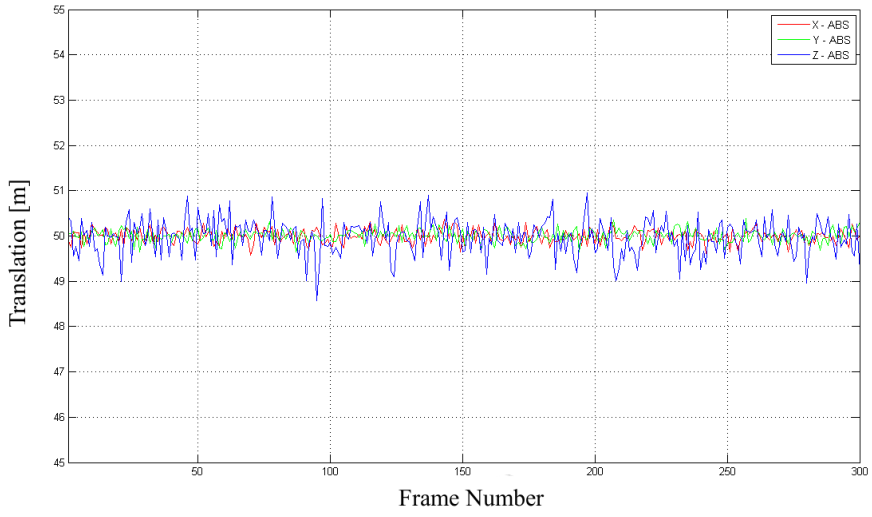

Figure 5.18: *Translation estimations of absolute pose estimation with STD of 1 pixel.*

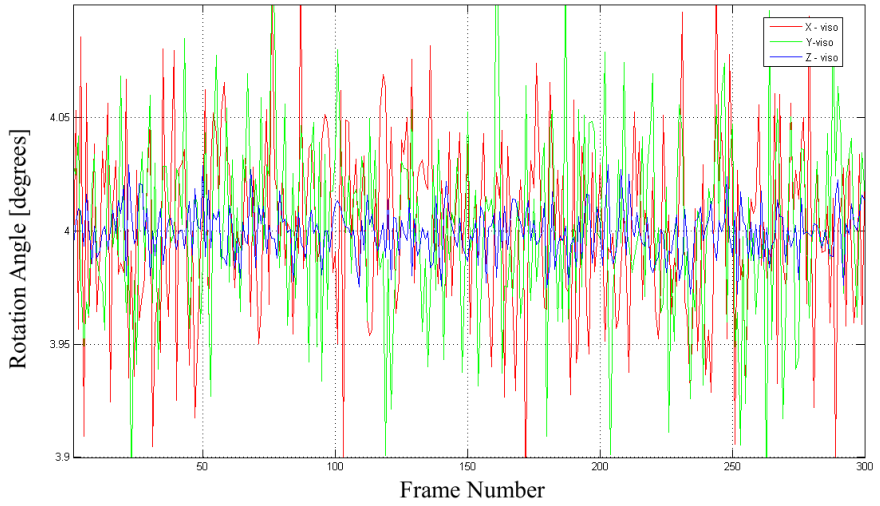

Figure 5.19: *Rotation estimations of relative pose estimation with STD of 1 pixel.*

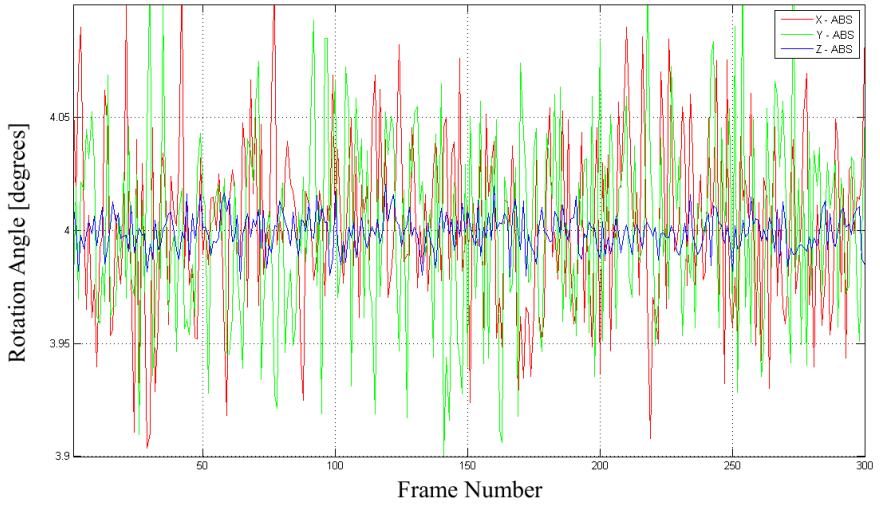

Figure 5.20: *Rotation estimations of absolute pose estimation with STD of 1 pixel.*

plane. From noisy correspondences, translation and rotation are estimated with the absolute pose estimation algorithm. Figure 5.18 shows a pose estimation result for an absolute translation with STD of 1 pixel. Correspondingly results for relative and absolute orientation estimation are shown in figures 5.19 and 5.20.

All the results are presented in tables 5.1 and 5.2 by calculating the *mean absolute error* (MAE) for each STD. In addition, for better comparison with the absolute pose estimation experiment in previous section, table 5.3 presents MAE for only 10 corresponding points, instead of 300 points, and with STD ranging from 0.5 to 10 pixels.

From the table 5.1, it is obvious to note that the subpixel accuracy of the SURF and LIBVISO2 algorithm improves the relative pose estimation results. It can be confirmed from the figures 5.7 and 5.8 that the accuracy of those algorithms are comparable to noise levels less than 1 pixel. This analysis also shows that the relative translation estimation in Z-axis is more accurate compared to the X and Y axis. Similar effect can be noticed from the figures 5.7 and 5.8.

By comparing the results between tables 5.1 and 5.2 it is clear that rotation estimation is on the same decade whereas there is a significant difference in translation estimation along X and Y axes. Peculiarly, translation along Z-axis is estimated almost on equal accuracy even though in the absolute pose estimation, the translation scale is calculated purely from the surface correspondences. Relative pose estimation needs the height of the camera from the ground in order to solve the scale ambiguity. In the relative pose estimation experiment, the only error source is the performance of the correspondence algorithm. Nevertheless, there are definitely some errors in the mosaicing process of making virtual images in Google Earth. However, these errors are not estimated in this thesis.

Table 5.3 on the other hand represents the magnitude of the error in the absolute pose estimation experiment. Corresponding points are manually selected and they are definitely not at the subpixel accuracy, but closer to several pixel accuracy. This alone explains the error of tens of meters in the absolute pose estimation. Moreover, there are errors due to the camera calibration. The camera, which was used in the test flight, was calibrated properly to the focus of 1 meter and 3 meters, but in the actual imaging process, camera focus was set to infinite which cause error to camera calibration matrix and geometric error correction. In addition, the GPS waypoints have some degree of error. All in all, the accuracy of the absolute pose estimation experiment, about 80 meters, can be more or less understand with this error analysis.

Table 5.1: *Translation error in meters and rotation error in degrees for different STD of noise for relative pose estimation.*

$\sigma_n(pixels)$	$T_x(m)$	$T_y(m)$	$T_z(m)$	$R_x(^\circ)$	$R_y(^\circ)$	$R_z(^\circ)$
0.1	0.18	0.17	0.05	0.003	0.003	0.001
0.2	0.33	0.35	0.09	0.007	0.007	0.002
0.4	0.67	0.73	0.20	0.014	0.013	0.004
0.8	1.31	1.37	0.38	0.027	0.026	0.008
1.6	2.69	2.44	0.80	0.048	0.053	0.014
3.2	5.01	5.25	1.67	0.104	0.100	0.028
6.4	10.7	10.7	3.17	0.210	0.212	0.057

Table 5.2: *Translation error in meters and rotation error in degrees for different STD of noise for absolute pose estimation.*

$\sigma_n(pixels)$	$T_x(m)$	$T_y(m)$	$T_z(m)$	$R_x(^\circ)$	$R_y(^\circ)$	$R_z(^\circ)$
0.1	0.01	0.01	0.03	0.003	0.003	0.001
0.2	0.02	0.02	0.07	0.006	0.007	0.001
0.4	0.05	0.05	0.13	0.012	0.013	0.002
0.8	0.09	0.09	0.24	0.026	0.026	0.005
1.6	0.18	0.17	0.49	0.050	0.053	0.010
3.2	0.40	0.34	1.08	0.106	0.104	0.020
6.4	0.74	0.66	2.24	0.224	0.222	0.036

Table 5.3: *Translation error in meters and rotation error in degrees for different STD of noise for absolute pose estimation. (only 10 correspondences)*

$\sigma_n(pixels)$	$T_x(m)$	$T_y(m)$	$T_z(m)$	$R_x(^\circ)$	$R_y(^\circ)$	$R_z(^\circ$
0.5	0.31	0.32	1.19	0.121	0.127	0.019
1.0	0.61	0.65	2.33	0.233	0.226	0.031
2.0	1.24	1.16	4.42	0.460	0.462	0.071
4.0	2.51	2.57	9.16	0.894	0.943	0.154
6.0	3.70	3.86	14.1	1.430	1.453	0.230
8.0	4.91	4.77	17.9	1.792	1.784	0.287
10.0	6.37	6.10	21.1	2.387	2.240	0.375

6. DISCUSSION & CONCLUSIONS

The main objective of this thesis was to study the applicability of the camera as a positioning device in aerial environment. Thesis first covered the fundamental theory behind imaging systems and geometrical image formation. That included a brief examination of camera principles and the mathematical model of a mono and stereo camera. Moreover, camera distortions and camera calibration were introduced and demonstrated with sample images of a checkerboard pattern. However, the main focus in this thesis was on correspondence search between images and on camera pose estimation algorithms.

There are many algorithms for solving correspondences. In general differential algorithms are used for low resolution images but their usage may be expanded with smoothing and multiresolution implementation. There are already various implementations in ground robotics or low flying aerial vehicles of KLT-based feature trackers with low resolution images and high frame rate. Differential algorithms can rarely be exploited for high resolution images because displacements between correspondences increase and assumptions for a constant motion field and the image brightness equation may not be valid in the time domain.

For high resolution images and medium or high flying aerial vehicles, feature-based algorithms offer more accurate and robust estimation of pointwise correspondences. Their performance can be improved by limiting the maximum distance between features both in the feature detection and feature matching stage. The pose estimation performance of the optimized LIBVISO2 was proved to be as accurate as SURF, and LIBVISO2 has already been used successfully for real-time applications by the author. LIBVISO2 or a similar library or algorithm would also be appropriate for aerial use. Furthermore, it would be interesting to combine inertial measurements and correspondence search as the technology is already used in some compact cameras, where the movement of a camera is compensated with digital image stabilization methods. Inertial sensors would offer an initial estimate for a correspondence algorithm so that the overall search range would be minimized. This would lead to an interdependent relationship of two sensors which may not be desirable. However, in flight control it is well-known practice to combine pose estimations from different sensory data.

Camera pose estimation algorithms and their theoretical background were also presented. It was shown that the subpixel accuracy of the correspondence search improves the performance in relative pose estimation. However, that conclusion was not totally clear in the case of the real-life aerial images, due to the equipment installation. There was not appropriate orientation information, and the relative pose estimation experiment for real-life aerial images did not fully demonstrated the utility of the presented algorithms. The unknown orientation bias remained in the results and it was not compensated off afterwards.

For the absolute pose estimation, the presented algorithm was shown to be sufficient to remove the error propagation originated from relative pose estimation. Nevertheless, the challenge of making a robust higher level image registration algorithm for aerial application and navigation remains still unsolved. That would be a natural step for further research and demonstration of cameras as a positioning device. Although, that would require a comprehensive and diverse test material.

Another clear future topic would be to combine pose estimations from aerial images with other sensory data. This would require deeper understanding of computational methods for non-linear and non-Gaussian estimation, such as EKF and UKF, because that is the nature of many vision-based problems. However, a camera could offer independent pose information for positioning and navigation algorithms which would improve the performance and reliability of a flying vehicle.

The presented algorithms were shown to be applicable for both relative and absolute positioning in virtual and real-life scenarios. Virtual Google Earth images offered a controlled environment and reference information for relative pose estimation experiment and analysis. Also, the application with the real-life aerial images proved the feasibility and determined the order of accuracy of the discussed techniques.

REFERENCES

[1] Bay H., Ess A., Tuytelaars T. & Gool L.V., *SURF: Speeded-Up Robust Features.*, Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346–359, 2008.

[2] Bay H., Ess A., Tuytelaars T. & Gool L.V., *SURF: Speeded-Up Robust Features.*, 19.3.2012, `http://www.vision.ee.ethz.ch/~surf/`

[3] Bouquet J.-Y., *a Camera Calibration Toolbox for MATLAB*, 22.1.2012, `http://www.vision.caltech.edu/bouguetj/calib_doc/`

[4] Brown D.C., *Decentering Distortion of Lenses*, Photometric Engineering, Vol. 32, No. 3, pp. 444–462, 1966.

[5] Brown L. G., *A survey of image registration techniques*, ACM Computing Surveys archive, Vol. 24, Issue 4, U.S.A. December 1992.

[6] Caballero F., Merino L., Ferruz J. & Ollero A., *Vision-Based Odometry and SLAM for Medium and High Altitude Flying UAVs*, Journal of Intelligent and Robotic Systems, 2008.

[7] Conte G., *Vison-Based Localization and Guidance for Unmanned Aerial Vehicles*, Ph.D. Dissertation, Linköping University, 2009.

[8] Cornall T. & Egan G., *Calculating Attitude from Horizon Vision*, Eleventh Australian International Aerospace Congress, 2005.

[9] Takamatsu J., Matsuhita Y., Ogasawara T. & Ikeuchi K., *Estimating Demosaicing Algorithms Using Image Noise Variance*, Proceedings of Computer Vision and Pattern Recognition (CVPR), 2010.

[10] Dakin J. & Brown R.G.W., *Handbook of Optoelectronics (Two-Volume Set)*, Taylor & Francis, 1680 p., U.S.A. 2006.

[11] Dawn S., Saxena V. & Sharma B., *Remote Sensing Image Registration Survey*, The International Conference on Image and Signal Processing, Lecture Notes on Computer Science, pp. 103-112, 2010.

[12] Dusha D., Boles W. & Walker R., *Attitude Estimation for a Fixed-Wing Aircraft Using Horizon Detection and Optical Flow*, Proceedings of the 9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications, pp. 485–492, 2007.

[13] Faugeras O., *Three-Dimensional Computer Vision: A Geometric Viewpoint*, MIT Press, 663 p., U.S.A. 1993.

[14] Geiger A., Ziegler J. & Stiller C., *StereoScan: Dense 3D Reconstruction in Real-time*, IEEE Intelligent Vehicles Symposium, Germany, June 2011. `http://www.cvlibs.net/software/libviso2.html`

[15] Georgiev M., *Laboratory Exercise: Stereo Camera Calibration and Rectification*, SGN-5456 3D Media Technology, TUT, February 2011.

[16] Gonzales R.C. & Woods R.E., *Digital Image Processing (3rd Edition)*, Prentice Hall, 954 p., U.S.A. 2008.

[17] Goshtasby A., *2-D and 3-D Image Registration for Medical, Remote Sensing, and Industrial Applications*, John Wiley & Sons, 258 p., U.S.A. 2005.

[18] Hartley R. & Zisserman A., *Multiple View Geometry in Computer Vision (2nd Edition)*, Cambridge University Press, 655 p., UK 2004.

[19] Harris C. & Stephens M., *A combined edge and corner detector*, Proceedings of the 4th Alvey Vision Conference, pp. 147–151, 1998.

[20] Hawkins J. K., *Textural Properties for Pattern Recognition*, Picture Processing and Psychopictorics, Academic Press, U.S.A. 1969.

[21] Hoddinott R., *Digital Macro Photography*, Institute Press, UK 2006.

[22] Hwangbo M., *Robust Monocular Vision-Based Navigation for a Miniature Fixed-Wing Aircraft*, Ph.D. Proposal, Robotics Institute, Carnegio Mellon University, 2009.

[23] Janesick J. R., *Photon Transfer*, SPIE Press, 256 p., U.S.A. 2009.

[24] Lindeberg T., *Feature Detection with Automatic Scale Selection*, International Journal of Computer Vision, Vol. 30, No. 2, 1998.

[25] Lowe D., *Object Recognition from Local Scale-Invariant Features*, Proceedings of the International Conference on Computer Vision, pp. 1150–1157, September 1999.

[26] Lucas B.D & Kanade T., *An Iterative Image Registration Technique with an Application to Stereo Vision*, Proceedings of Imaging Understanding Workshop, pp. 121–131, 1981.

[27] Luhmann T., Robson S., Kyle S. & Harley I., *Close Range Photogrammetry: Principles, Techniques and Applications*, Jowh Wiley & Sons, 528 p., UK 2007

[28] Ma Y., Soatto S., Kosecka J. & Sastry S.S., *An Invitation to 3-D Vision*, Springer, 546 p., U.S.A. 2003.

[29] Maccaferri A., *DCRaw v.s. Camera Raw*, 2.12.2011, `http://www.photoactivity.com/Pagine/Articoli/005DCRaw/Bayer3.jpg`

[30] Marr D. & Hildreth E., *Theory of Edge Detection*, Proceedings of the Royal Society of London, Vol. 207, No. 1167, pp. 187–217., February 1980.

[31] Mei X. & Porikli F., *Fast Image Registration via Joint Gradient Maximization: Application to Multi-Modal Data*, SPIE Conference on Electro-Optical and Infrared Systems, 2006.

[32] Mikhail E., Bethel J. & McGlone J., *Introduction to Modern Photogrammetry*, John Wiley & Sons, 479 p., U.S.A. 2001.

[33] Mikolajczyk K. & Schmid C., *A Performance Evaluation of Local Descriptors*, IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, pp. 257–263, June 2003.

[34] Mundy J.L., *The Relationship Between Phogrammetry and Computer Vision*, Int. Society for Optical Engineers Conference (SPIE), Vol. 1994, pp. 92–105, 1993.

[35] Nixon O., *Applications Set Imager Choices*, Advanced Imaging, July 2008.

[36] *Open Source Computer Vision Library (OpenCV)*, 15.3.2012, `http://opencv.willowgarage.com/wiki/`

[37] *OpenCV Reference Manual (rel. 2.3)*, 15.3.2012, `http://opencv.itseez.com/opencv2refman.pdf`

[38] Sakoe H. & Chiba S., *Dynamic Programming Algorithm Optimization for Spoken Word Recognition*, IEEE Transactions on Acoustics, Speech and Signal Processing, 26(1), pp. 43–49, 1978.

[39] Shi J. & Tomasi C., *Good Features to Track*, 9th IEEE Conference on Computer Vision and Pattern Recognition, June 1994.

[40] Stroebel L. & Zakia D., *The Focal Encyclopedia of Photography*, Focal Press, 914 p., U.S.A. 1995.

[41] Sturm P., *Algorithms for Plane-Based Pose Estimation*, IEEE Computer Vision and Pattern Recognition, Vol. 1, pp. 706–711, June 2000.

[42] Tomasi C. & Kanade T., *Detection and Tracking of Point Features*, Carnegio Mellon University Technical Report, CMU-CS-91-132, April 1991.

[43] Trucco E. & Verri A., *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, 343 p., U.S.A. 1998.

[44] Tsai R.Y., Huang T.S. & Zhu W.-L., *Estimating three-dimensional motion parameters of a rigid planar patch, ii: singular value decomposition.* IEEE Transactions on Acoustics Speech Signal Process, 1982.

[45] Waynant R.W. & Ediger M.N., *Electro-Optics Handbook (2nd Edition)*, McGraw-Hill, 992 p., U.S.A. 2000.

[46] Welch G. & Bishop G., *An Introduction to the Kalman Filter*, SIGGRAPH 2001, Annual Conference on Computer Graphics & Interactive Techniques, August 2001.