Vision-Based Navigation for a Small Fixed-Wing Airplane in Urban Environment

Myung Hwangbo

CMU-RI-TR-12-11

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Robotics.

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

May 2012

Thesis Committee:
Takeo Kanade (chair)
Omead Amidi
Sanjiv Singh
Randy Beard, Brigham Young University

Keywords: Unmanned aerial vehicle, fixed-wing airplane, autonomous navigation, vision-based attitude estimation, sensor calibration, sampling-based motion planning, air-slalom task

This dissertation is dedicated to

my lovely wife, Hyojin,

who has been my greatest support, encouragement and shelter
during my time at Carnegie Mellon.

Abstract

An urban operation of unmanned aerial vehicles (UAVs) demands a high level of autonomy for tasks presented in a cluttered environment. While fixed-wing UAVs are well suited for long-endurance missions at a high altitude, enabling them to navigate inside an urban area brings another level of challenges. Their inability to hover and low agility in motion cause more difficulties on finding a feasible path to move safely in a compact region, and the limited payload allows only low-grade sensors for state estimation and control.

We address the problem of achieving vision-based autonomous navigation for a small fixed-wing in an urban area with contributions to the following several key topics. Firstly, for robust attitude estimation during dynamic maneuvering, we take advantage of the line regularity in an urban scene, which features vertical and horizontal edges of man-made structures. The sensor fusion with gravity-related line segments and gyroscopes in a Kalman filter can provide driftless and realtime attitude for flight stabilization. Secondly, as a prerequisite to sensor fusion, we present a convenient self-calibration scheme based on the factorization method. Natural references such as gravity, vertical edges, and distant scene points, available in urban fields, are sufficient to find intrinsic and extrinsic parameters of inertial and vision sensors. Lastly, to generate a dynamically feasible motion plan, we propose a discrete planning method that encodes a path into interconnections of finite trim states, which allow a significant dimension reduction of a search space and result in naturally implementable paths integrated with flight controllers. The most probable path to reach a target is computed by the Markov Decision Process with motion uncertainty due to wind, and a minimum target observation time is imposed on the final motion plan to consider a camera's limited field-of-view.

In this thesis, the effectiveness of our vision-based navigation system is demonstrated by what we call an "air slalom" task in which the UAV must autonomously search and localize multiple gates, and pass through them sequentially. Experiment results with a 1m wing-span airplane show essential navigation capabilities demanded in urban operations such as maneuvering passageways between buildings.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Unmanned Aerial Vehicles (UAVs) have been under development since the beginning of flight because they can eliminate the risk to a pilot's life and they are generally inexpensive to procure. Most UAVs are controlled remotely from a ground control station but automation systems are being progressively integrated for autonomous operation. GPS (Global Positioning System) and other sophisticated navigation systems make it possible for UAVs to take over high-risk aerial missions that required manned aircraft.

The main role of UAVs thus far has been information gathering. UAVs' aerial viewpoint provides a better perspective; they also have the ability to cover a large area and with higher speeds than ground vehicles, and explore a viewpoint that is not constrained to a road network. While intelligence, surveillance and reconnaissance missions still remain their predominant tasks, their roles are expanding to diverse civilian, federal and commercial areas including law enforcement, environment monitoring, network connection and communications relay [1].

One active research direction for UAVs is a development of miniature systems such as man-portable UAVs or MAVs (Micro Air Vehicles) whose wingspan is less than 6 inches [2]. As hardware continues to shrink in size, weight and cost, there is a dramatic rise in both interest and application for miniaturized UAVs. Small size and improved endurance can provide a significant advantage in urban and other high-density environments.

1.1 Autonomy in Urban Applications

In contrast to the recognized value of military applications, it is rare to encounter successful UAV deployment in civil applications. This is because safety remains an important issue in civil airspace and UAVs can be a hazard to other aircraft, ground vehicles, people and facilities. Near-human capabilities, which are quite challenging to design, are required to ensure safety as the Federal Aviation Administration (FAA) has a regulation that UAVs must have onboard *detect,*

Figure 1.1: Prospective urban operations of a fixed-wing UAV: air-to-ground target tracking, surveillance in urban canyons, and UAV & UGV collaborative mission.

see and avoid capabilities to prevent in-air collisions [1].

Autonomous flight refers to the UAVs' capability to carry out an entire task independently without input from a remote operator. Prospective UAV urban operations in Figure 1.1 show that the number and complexity of obstacles and the density of operations increase the number of decisions that must be made, and compress the time required for the decisions. The importance and difficulty of this problem motivates the development of increased levels of autonomy to support urban UAV operations.

1.2 Vision-based Navigation

One of the critical requirements to make a UAV autonomous and practical is a reliable estimation of its position and orientation as well as a 3D mapping of its surrounding environment. Among many possible sensors for these purposes, a camera has been acclaimed for its unique advantage to deliver multi-layered information in the format of images. In contrast to navigational sensors such as GPS and INS (Inertial Navigation System), which provide information only about the vehicle's own motion with respect to the inertial frame, vision can provide additional information relative to the environment, for example, how close the vehicle is to an obstacle, where targets are located in an area, or how the vehicle is aligned with the horizon.

In addition to GPS, which may stop working in the shadow of satellite visibility, vision always works in a cluttered urban environment and even indoors as long as the captured images have sufficient texture and illumination. The drawback is that a substantial computational burden is required to extract useful information from images, and the information is scene-dependent.

1.3 Fixed-wing UAV

There are three common types of miniaturized UAVs, as shown in Figure 1.2. Each type has its own advantages and disadvantages, and different task scenarios call for different types of UAVs.

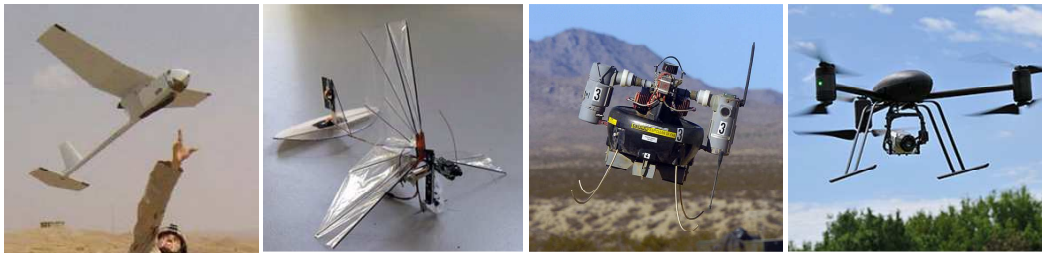


Figure 1.2: Typical types of small and micro UAVs: an airplane-like fixed-wing model, bird or insect-like ornithopter (flapping wing) model, and helicopter-like rotary wing models (ducted fan and quad rotor).

For instance, fixed wings (drones) are well suited for tasks that require high endurance and extended loitering times, but generally they do not work indoors as they cannot hover or make tight turns. Rotary wings, on the other hand, allow hovering and movement in any direction at the cost of short flight time and complex design. Flapping wings may offer the most potential for miniaturization and maneuverability, but in terms of real implementation, they continue to be far inferior to the other UAV types.

Fixed wings have unique benefits such as readiness to fly and glide, lower running cost and a wide working area. Lacking the ability to hover, however, causes challenges for motion planning and obstacle avoidance in complex environments due to minimum forward velocity and limited agility in motion.

1.4 Problem Statement

In this thesis, we will investigate autonomous algorithms that enable a small fixed-wing vehicle to navigate in an urban environment at low altitudes and achieve a given task in the following operational situations:

Commodity-level fixed-wing airplane Our UAV system development is targeted to automate an ordinary remote-controlled fixed-wing airplane, which is readily available in market for hobbyists. No special design of a wing's airfoil shape or control surface for a specific task is considered. A high budget-performance ratio is expected from nearly the simplest platform that can offer aerial navigation.

Low-grade sensors Due to the small payload allowed for a small UAV, the sensing capability is subject to simple and unsophisticated lightweight onboard sensors. A low-cost avionics system equipped with on-chip GPS, IMU and pressure sensors is considered. The camera is restricted to one with a low-resolution image and limited field-of-view lens due to the payload constraint and subsequent small onboard computational power.

Sensor fusion and calibration Vehicle and target states required in a given navigation task are estimated from low-grade sensors. Sensor fusion is therefore necessary to combine sensors' complementary measurements and yield a reliable and realtime state estimation. Prior to the fusion, sensor calibration for an IMU and camera is a necessary step. It would be more favorable if the calibration can be done in field because the sensors are subject to replacement and frequent setting change in an expendable system.

Urban environment The environment of interest is mostly composed of man-made structures such as buildings, towers and traffic roads. Rather than a series of steady flights between way-points, an urban operation requires continuous dynamic maneuvering at a low altitude, and deliberate motion plans to reach a goal in a complex environment where close interactions often occur with the surroundings. A demonstrative task asks the UAV to pass through multiple gates in different orientations, which emulates common situations in an urban area such as maneuvering through passageways between buildings.

1.5 Thesis Statement

In this thesis, we will present a vision-based navigation framework for a small, unmanned fixed-wing airplane operating in an urban environment. The core problems we address here are the development and integration of new algorithms on attitude estimation, sensor calibration, and motion planning. The effectiveness of our framework is evaluated and demonstrated through an air-slalom task. This thesis makes the following key statements:

- It is possible to produce a driftless and realtime attitude estimation for flight control when gravity-related line segments in urban scene are combined with inertial sensing.
- Convenient and on-site sensor calibration is available for inertial and visual sensor fusion from natural references such as gravity, vertical edges and distant scene points.
- The use of trim states of a fixed-wing is efficient to build discrete primitive motions and generate a feasible and practically implementable motion plan.

The framework and methods developed in this thesis can be readily transferrable to other fixed-wing platforms and other urban task scenarios. See Figure 2.1 for the overview of the thesis organization.

Chapter 2

Air Slalom Task

One of the most common maneuvers demanded for a fixed-wing UAV in an urban environment is to approach a target or region of interest in a specified direction. This kind of maneuver is necessary, for example, when the UAV is in the middle of entering a gate or passageway between buildings, following a target moving on a road network, or landing on a runway or driveway. Unlike rotor-type air vehicles that can make a standing rotation, the change of an approaching direction raise a challenge to fixed-wings because they require not only space and time to make a turn but also a deliberate motion plan to meet the new direction.

This fact leads us to design a benchmark mission in what we call an *air-slalom task*, which is focused on the evaluation and demonstration of vision-based autonomous navigation algorithms for a fixed-wing UAV by presenting a challenging course. As illustrated in Figure 2.1, an air vehicle needs to complete tight turns through a slalom course consisting of multiple gates. In order to precisely enter multiple gates in a row, it is essential to integrate advanced realtime motion planning as well as reliable target and vehicle state estimation into the navigation system. Like the Red Bull Air Race for human pilots [3], this benchmark task can be used to compete with other intelligent UAV systems for completing the task in the fastest time.

2.1 Task Description

Prior to flight, the location and orientation of gates are completely unknown to a UAV, but their geometric shapes and colors are indicated so that their poses are visually identifiable. The UAV needs to search, localize and pass through this series of gates in a designated order with no human intervention. These multiple gates are placed with different entrance orientations in a twisting and potentially difficult manner. Flying too low or high with respect to each gate disqualifies the successful gate entrance.

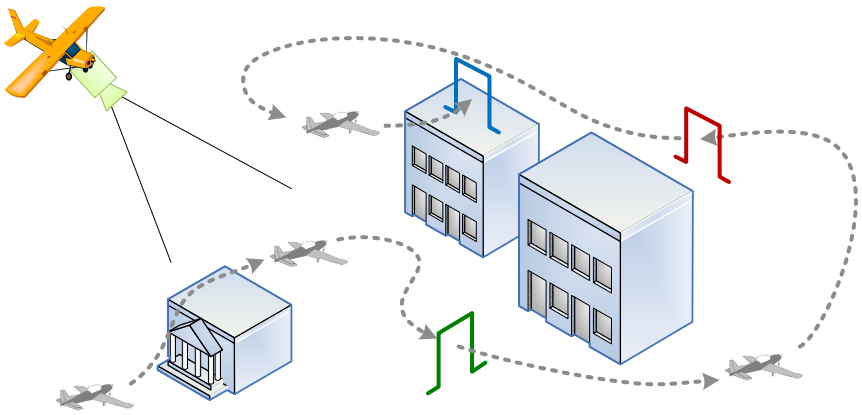


Figure 2.1: An air-slalom task designed as a benchmark test of an autonomous vision-based navigation system of a small fixed-wing UAV in an urban environment

2.2 Navigation System Requirement

A high-level autonomous system for mission activities in aerial navigation involves the proper integration of many intelligent sub-systems. Figure 2.2 shows a diagram of functional blocks generally required to surmount the air-slalom task when conventional avionic sensors such as GPS, IMU and camera are equipped onboard. The navigation system has to minimally include functional blocks for flight control, attitude estimation, vehicle and target state estimation, target detection, motion planning, and sensor calibration.

Among the many requirements for an intelligent navigation system, this thesis is dedicated to the research and development of the following fundamental functions.

Attitude estimation for flight control Reliable and realtime attitude estimation is essential for robust flight control during dynamic maneuvering. The regularity of parallel line segments appearing in an urban scene is utilized to recover an absolute measurement of the vehicle attitude. A Kalman filter framework to combine visual and inertial measurements will be described in Chapter 3.

Sensor self-calbration The calibration of intrinsic and extrinsic parameters of a camera/IMU system is mandatory before the sensor fusion. The use of natural references, such as gravity and distant scene points, provides in-situ calibration capability whenever a sensor setting changes in field. A linear or iterative calibration scheme based on the factorization method will be presented in Chapter 4.

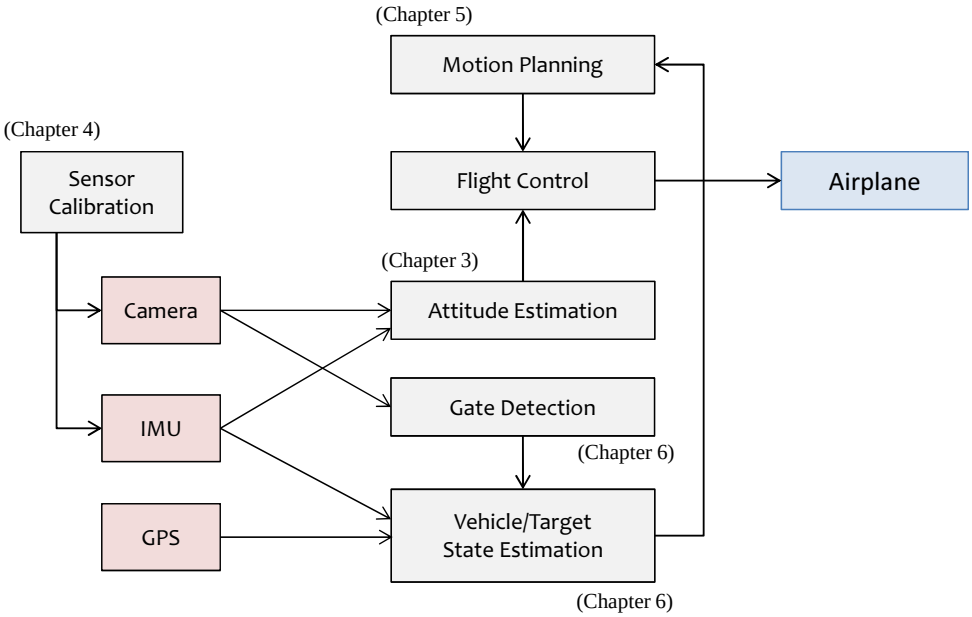


Figure 2.2: Diagram of a vision-based navigation system for the air-slalom task

Motion planning of a fixed-wing UAV A vehicle trajectory that results in success when approaching and entering a gate should be feasible for a fixed-wing airplane. It is also preferred that the trajectory be easily executable and implementable. An efficient discrete motion planning method that interconnects a vehicle's trim states will be described in Chapter 5.

Integration and experiments The navigation system additionally integrates target detection and state estimation of vehicle and target postures for the air-slalom task. The description of the hardware and software architecture of our UAV system and the results of the air-slalom experiment will be presented in Chapter 6.

Chapter 3

Visual/Inertial Attitude Estimation

Attitude is a fundamental vehicle state towards stabilizing an aircraft and achieving autonomous navigation of UAVs. In this chapter, a fusion framework of visual and inertial sensors is presented to provide reliable and robust attitude estimation during highly dynamic maneuvers.

A vision sensor has enabled natural scenes to act as an alternative source for attitude inference in addition to other sensors such as IMU, GPS, magnetometer and thermopile. The merit of a camera is that it is not only self-contained and passive, but also interactive with its environment. Detection of a reliable reference of gravity in an image can recover absolute camera attitude with a relatively constant level of accuracy. Horizon detection, for example, has been a prevalent method for vision-based attitude estimation for mid-altitude missions. Figure 3.1(a) shows that the horizon, equal to a straight line separating the sky and the ground regions, expresses the attitude directly. However, this method is no longer applicable at low altitudes since the horizon loses visibility due to the occlusion with surroundings, as shown in Figure 3.1(b)-(c).

Fortunately, another visual method can provide a way to infer attitude in images that portray urban environments. Since man-made environments generally exhibit strong regularity in structure, a camera will observe a number of line segments which are either parallel or orthogonal to the gravity direction like Figure 3.2. These line segments in an image allow us to construct a set of vanishing points at which each parallel line bundle intersects, and to describe the attitude using an explicit geometric relationship between the vanishing points.

Using this advantage of urban scene regularity, we propose a driftless attitude estimator that combines inertial gyro data with gravity-related line measurements in a Kalman filter; the gyroscope traces instantaneous relative rotation at a high rate and the line measurements reset its inevitable unbounded error accumulation. Note that the line segments become ready for the attitude estimation in the Kalman filter once after they are correctly classified into either vertical or horizontal edges according to vanishing points.

The scene of interest is not limited to a close view of urban structures such as the facade

Figure 3.1: Visual features available in urban scenes for attitude estimation at different altitudes: at a low altitude, the horizon separating the sky and ground is occluded but gravity-related line segments in man-made structures can be exploited for attitude estimation.

of a building or indoor scene. We also consider a distant view of dense structures like that in Figure 3.1(b), which is a more likely setting for fixed-wing UAV operations. These distant images are more challenging because an estimation method must interpret a sparse set of short and noisy line segments and reject outliers effectively.

In the experiment section, our sensor fusion method is compared with other conventional inertial-only and vision-only attitude estimation methods.

3.1 Related Work

Attitude estimation for small UAVs is typically a study of fusion methods that use complementary low-cost sensors. Vast research has been conducted regarding fusion of IMU, GPS and magnetometer in a Kalman filter framework aimed at a full state of the vehicle including position and orientation. Table 3.1 summarizes the pros and cons of sensor fusion methods when gyroscopes are combined with other complementary sensors.

When only an IMU was available, a nonlinear complementary filter [4] and direction cosine matrix [5] was developed for long-term driftless attitude estimation. After dynamic acceleration was removed from accelerometers using simplified vehicle dynamic models, a gravitational direction was combined with integrated gyro angles. However, these simple models could not capture aggressive maneuvers precisely. Infrared thermopile was used to detect the horizon from temperature difference between the sky and ground, but its accuracy and resolution were limited [6].

In vision-only methods, when the horizon is visible with no occlusion, the attitude has been derived from a straight line or a curved line in catadioptric images by separating the sky and ground regions based on context differences [7, 8, 9, 10, 11]. Parallel line segments present in an urban scene have been used for attitude estimation via vanishing point detection [12, 13]. A batch process was developed for recovering the history of camera orientations using a bundle adjustment of vanishing points [14]. Mutually orthogonal vanishing points were searched in a building scene

Table 3.1: Comparison between sensor fusion methods that combine gyroscopes with other complementary sensors for driftless attitude estimation

Fusion with gyro	Reference	Pros and Cons	
GPS	Carrier phase, Antenna array	+ Absolute orientation measure − More noisy in shorter baseline − Fixed integer ambiguity	Gebre-Egziabher [17], Li *et al.* [18]
Accelerometer	Static gravity	+ Compact and cost effective − Big short-term error − Dynamic gravity elimination	Easton *et al.* [4], Premerlani *et al.* [5]
Magnetometer	Earth magnetic field	+ Compact and cost effective − Unknown angle around magnetic vector − Magnetic interference	Metni *et al.* [19], Eldredge [20]
Camera	Horizon	+ Absolute attitude measure − Occlusion at low altitude − Relatively intensive computation	Ettinger *et al.* [7], Cornall *et al.* [9]
Camera	Gravity-related line segments	+ Urban scene at low altitude + Absolute attitude measure − Line classification needed − Intensive computation	Demonceaux *et al.* [15], Hwangbo *et al.* [21]

(+ represents pros and − represents cons)

and a vertical vanishing point was found by sky detection [15] and it was possible to track these points over a catadioptric image stream [16].

Our approach is also based on vanishing point detection but raw line measurements are directly used in a Kalman filter without explicit use of detected vanishing points. In the fusion with the gyroscope, a current best attitude estimate is repeatedly updated by each line measurement.

3.2 Attitude from Parallel Lines

In the perspective projection of a pinhole camera, parallel lines in a 3D scene merge at a single point of the image called a *vanishing point* (VP). Multiple vanishing points may exist from vertical and horizontal edges as shown in Figure 3.2. When a camera calibration matrix $\mathbf{K}$ is known, a geometric relationship between camera orientation, the horizon, and vanishing points along principle world coordinates has been well established in a Gaussian sphere representation [22]. One merit of the Gaussian sphere is that it is possible to describe all vanishing points, including VPs at infinity, in homogeneous coordinates with no exceptional cases.

Suppose the camera is calibrated and line segments are expressed in normalized image coordinates. In Figure 3.3, the Gaussian sphere $\mathcal{G}$ is a unit sphere located at the camera's optical center O_c. On a 2D projective space $\mathcal{P}^2$, a line segment $\ell_i \in R^{3\times 1}$ on the image is represented as a normal vector of its great circle C_i in homogeneous coordinates. The intersection of two parallel line segments, ℓ_i and ℓ_j, constructs a vanishing point, $\mathbf{v}_{ij} = \ell_i \times \ell_j$, located on the sphere. Based on the duality in $\mathcal{P}^2$ between points and lines, $\mathbf{v}_{ij}$ can be viewed as a direction to the corresponding vanishing point at infinity. The vanishing points that lie on the same plane in a scene define a *vanishing line* in the image.

In a view of man-made structures, we can arguably claim that all vertical edges merge at a single vanishing point $\mathbf{v}_v$, which we call a *vertical vanishing point*. On the other hand, horizontal edges may converge at multiple vanishing points, which we call *horizontal vanishing points* $\mathbf{V}_h = (\mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$. The vertical VP is aligned to the gravity direction $\mathbf{z}_w$ in the world coordinate and the horizontal VPs lie on the world ground plane $\mathbf{W}$ of which normal is the gravity direction. Hence, the vanishing points, $\mathbf{v}_v$ and $\mathbf{V}_h$, are subject to the following geometric constraint:

$$\mathbf{v}_v^\top \mathbf{v}_h^i = 0 \quad i = 1, \ldots, m. \tag{3.1}$$

Note that horizontal VPs in $\mathbf{V}_h$ are not necessarily orthogonal to each other unless the scene is strictly subject to the Manhattan world assumption like Figure 3.2. In real-world aerial scenes, no constraints between horizontal VPs are given in this paper.

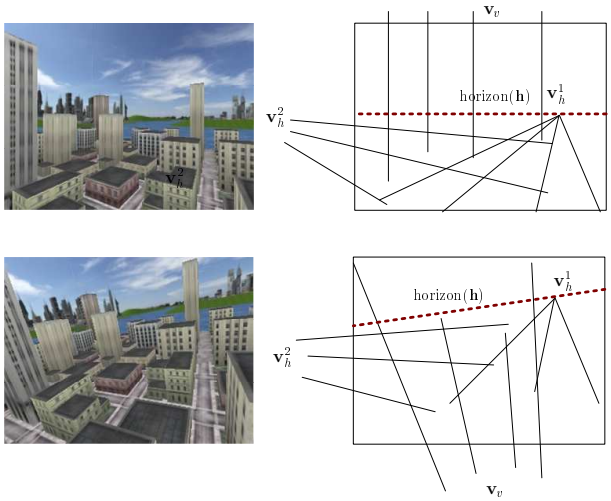
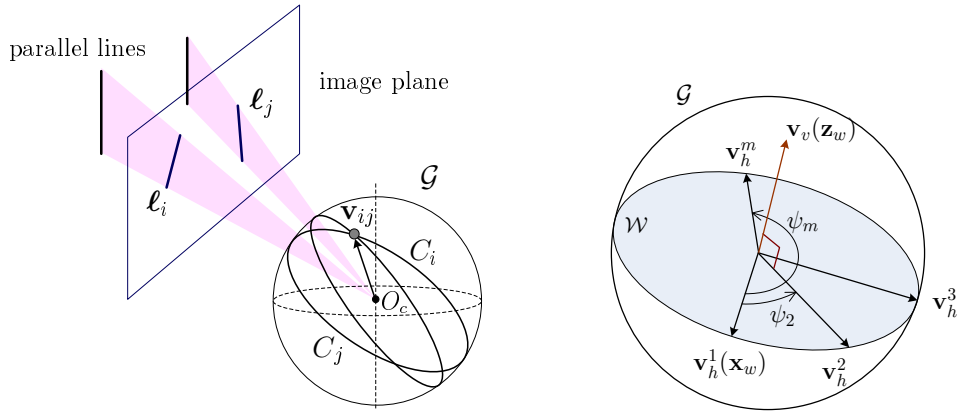


Figure 3.2: Horizon and vanishing points in a Manhattan world: several bundles of parallel lines converge at either a single vertical VP ($\mathbf{v}_v$) or multiple horizontal VPs ($\mathbf{v}_h^1, \mathbf{v}_h^2$) on the image. The horizon $\mathbf{h}$, equivalent to camera attitude, is derived from either $\mathbf{v}_v$ or a vanishing line connecting $\mathbf{v}_h^1$ and $\mathbf{v}_h^2$.



(a) Parallel line intersection (b) Gaussian sphere

Figure 3.3: Representation of vanishing points on a Gaussian sphere $\mathcal{G}$: (a) Parallel lines $\boldsymbol{\ell}_i$ and $\boldsymbol{\ell}_j$ intersect at a point $\mathbf{v}_{ij}$. (b) Camera attitude is equivalent to the vertical VP ($\mathbf{v}_v$, antipodal point), or its great circle $\mathbf{W}$ on which multiple horizontal VPs ($\mathbf{v}_h^1, \ldots, \mathbf{v}_h^k$) lie.

The horizon $\mathbf{h}$ is a vanishing line connecting any two horizontal VPs as shown in Figure 3.2. From the constraint (3.1), it can also be seen that the line $\mathbf{h}$ is *dual* to the point $\mathbf{v}_v$ in $\mathcal{P}^2$.

$$\mathbf{h} = \mathbf{v}_h^i \times \mathbf{v}_h^j, \quad \mathbf{v}_v = \mathbf{v}_h^i \times \mathbf{v}_h^j, \tag{3.2}$$

This duality implies that the horizon $\mathbf{h}$ is a projection of the world ground plane $\mathcal{W}$ and $\mathbf{v}_v$ is a projection of its plane normal $\mathbf{z}_w$ in Figure 3.3.

The camera attitude is equivalent to either the horizon $\mathbf{h}$ or the vertical VP $\mathbf{v}_v$, because an orientation of the world ground plane with respect to the camera is equal to that of the great circle $\mathcal{W}$ in the Gaussian sphere. Given a relative camera pose to the airplane, an airplane attitude can be uniquely determined when either $\mathbf{v}_v$ or at least two $\mathbf{v}_h^i$s are found in an image.

3.2.1 Vertical vanishing point

Suppose that a camera frame c is attached to an airplane frame b so that the camera's principle axis is aligned along the fuselage centerline in Figure 3.4. The orientation of an airplane has a roll ϕ, pitch θ, and yaw angle ψ. Then, by definition, a perspective projection of the world z-axis $\mathbf{z}_w^c$ in the camera coordinate with a normalized camera matrix $\mathbf{P} = [\mathbf{I} \ \mathbf{0}]$ is equal to $\mathbf{v}_v$:

$$\mathbf{v}_v = \mathbf{P} \begin{bmatrix} \mathbf{z}_w^c \\ 0 \end{bmatrix} = \mathbf{R}_b^c \, \mathbf{R}_w^b \, \mathbf{z}_w^w = \begin{bmatrix} -s_\phi c_\theta \\ -c_\phi c_\theta \\ -s_\theta \end{bmatrix} \tag{3.3}$$

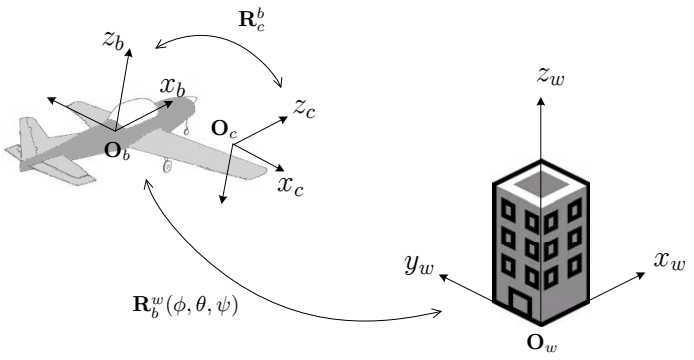


Figure 3.4: Coordinate transformation between the world frame to the camera frame.

where

$$\mathbf{R}_b^w = \begin{bmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}, \quad \mathbf{R}_b^c = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}. \tag{3.4}$$

Note that camera translation has no effect on the projection of a point at infinity such as a vanishing point.

Let $\mathbf{v}^* = (v_x, v_y)^\top$ denote a vanishing point $\mathbf{v}$ on an image which is described in inhomogeneous coordinates. As expected, the position of $\mathbf{v}_v^*$ is determined solely by the attitude (ϕ, θ) regardless of the yaw angle ψ:

$$\mathbf{v}_v^* = \left[\frac{x_w^c}{z_w^c}, \ \frac{x_w^c}{z_w^c} \right]^\top = \left[\frac{s_\phi c_\theta}{s_\theta}, \ \frac{c_\phi c_\theta}{s_\theta} \right]^\top. \tag{3.5}$$

In the opposite way, the attitude can be derived from $\mathbf{v}_v^*$ as follows:

$$\phi = \text{atan2}\,(v_x,\, v_y), \quad \theta = \text{atan}\frac{1}{\sqrt{v_x^2 + v_y^2}} \tag{3.6}$$

The horizon $\mathbf{h}$ on the image, dual to $\mathbf{v}_v$, is a line described as

$$(s_\phi c_\theta)x + (c_\phi c_\theta)y + s_\theta = 0 \tag{3.7}$$

3.2.2 Horizontal vanishing point

A horizontal VP is a point at infinity on the world ground plane $\mathbf{W}$. Without loss of generality, we can assume $\mathbf{v}_h$ is at the direction of the world x-axis. Then the projection of $\mathbf{x}_w^c$ is stated as

$$\begin{aligned} \mathbf{v}_h &= \mathbf{P} \begin{bmatrix} \mathbf{x}_w^c \\ 0 \end{bmatrix} = \mathbf{R}_b^c \, \mathbf{R}_w^b \, \mathbf{x}_w^w \\ &= \Big[c_\phi s_\psi - s_\phi s_\theta c_\psi, \quad -(s_\phi s_\psi + c_\phi s_\theta c_\psi), \quad c_\theta c_\psi \Big]^\top \end{aligned} \tag{3.8}$$

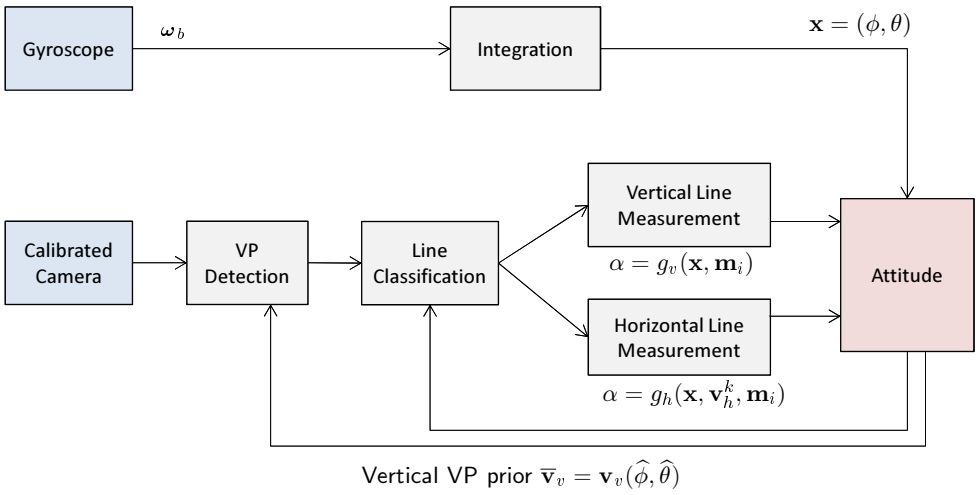


Figure 3.5: Sensor fusion diagram of integrated gyro angle and image line measurements in an extended Kalman filter framework for driftless attitude estimation.

$$\mathbf{v}_h^* = \left[\frac{c_\phi s_\psi - s_\phi s_\theta c_\psi}{c_\theta c_\psi}, \quad \frac{-s_\phi s_\psi - c_\phi s_\theta c_\psi}{c_\theta c_\psi} \right]^\top \tag{3.9}$$

All the horizontal VPs $(\mathbf{v}_1^h, \ldots, \mathbf{v}_m^h)$ in $\mathbf{V}_h$ lie on the horizon $\mathbf{h}$ and their locations are indicated by yaw angles $(\psi_1, \ldots, \psi_m)$, respectively, as shown Figure 3.3(b). In other words, the locus of $\mathbf{v}_h$ along $\psi \in [-\pi, \pi]$ is equal to $\mathbf{h}$.

3.3 Visual/Inertial Sensor Fusion

An extended Kalman filter (EKF) framework [23] is employed to combine inertial and visual sensors for driftless attitude estimation in realtime. Figure 3.5 shows a diagram of the sensor fusion in which a high-rate attitude prediction from a gyroscope is updated by low-rate line measurements from a camera.

At the Kalman update, we choose to use line segments rather than other higher-level measurements such as detected vanishing points or an image-based attitude recoverd by VPs. The first reason is that noise characteristics of line measurements are closer to Gaussian than those of other measurements, and thus more adequate to model a noise variance for the EKF. Edges are lower-level and less-processed features of an image. Another reason is that vanishing points are not needed if all line measurements are known to be vertical. Vertical edges are immediately ready for the Kalman update. In the situation where vertical edges are dominant, no further steps to estimate its VP are necessary.

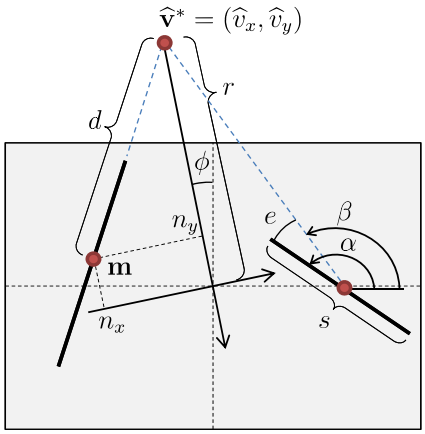


Figure 3.6: Observation model of a line segment for a desired vanishing point $\widehat{\mathbf{v}}^*$, $g(\cdot) = \beta$. The angle β is obtained from a virtual line connecting $\mathbf{v}^*$ and the midpoint m.

3.3.1 Process model

Let $\mathbf{x}$ denote the attitude (ϕ, θ) and $\mathbf{b}$ denote a gyroscope bias. A total state in EKF is $\boldsymbol{\chi} = [\mathbf{x},\ \mathbf{b}]^\top = [\phi, \theta, b_x, b_y, b_z]^\top$. A process model is comprised of the mechanization of a strap-down tri-axial gyroscope and the propagation of a gyroscope bias error model as follows:

$$\dot{\mathbf{x}} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \end{bmatrix} (\boldsymbol{\omega}_g - \mathbf{b}) + \mathbf{n}_x \tag{3.10}$$

$$\dot{\mathbf{b}} = \mathbf{n}_b \tag{3.11}$$

where $\boldsymbol{\omega}_g$ is a gyroscope output, $\mathbf{n}_x$ and $\mathbf{n}_b$ are Gaussian process noise and gyro bias error noise, respectively. Errors in the gyroscope's scale and bias are modeled as a single time-varying bias term $\mathbf{b}$ in (3.11) and the Coriolis effect is neglected in (3.10). The Jacobian $\mathbf{F}$ in the state transition is derived in Appendix-A.1.

3.3.2 Line measurement update

Suppose that line segments are already classified according to vanishing points. From a classification step that will be described in Section 3.4, a set of line segments $\mathbf{L} = (\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_n)$ is grouped into $(m+1)$ line groups with corresponding vanishing points $\mathbf{H} = (\mathbf{v}_v, \mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$, which we call a horizon measurement. Each line segment $\boldsymbol{\ell}_i \in \mathbf{L}$ is now represented as

$$\boldsymbol{\ell}_i = (\alpha_i,\ \mathbf{m}_i,\ s_i;\ \mathbf{v} \in \mathbf{H}) \quad \text{for } i = 1, \ldots, n \tag{3.12}$$

where the parameters are its midpoint $\mathbf{m}_i$, slope α_i, length s_i and associated VP $\mathbf{v}$ in $\mathbf{H}$.

A line observation model $g(\cdot)$ captures how a line segment $\boldsymbol{\ell}_i$ should be aligned when a desired vanishing point $\widehat{\mathbf{v}}^*$ is given on an image in Figure 3.6. This model is represented as the angle β_i

of a virtual line connecting $\widehat{\mathbf{v}}^*$ and the midpoint $\mathbf{m}_i$. The way to set a desired vanishing point $\widehat{\mathbf{v}}^*$ for the line $\boldsymbol{\ell}_i$ is different depending on whether its associated line group is vertical or horizontal. If the line is a vertical edge, $\widehat{\mathbf{v}}^*$ is a function of the attitude $\mathbf{x}$ only. If the line is a horizontal edge, $\widehat{\mathbf{v}}^*$ is a function of the attitude $\mathbf{x}$ and its associated vanishing point $\mathbf{v}_h^k$ in $\mathbf{H}$.

$$\begin{aligned}\beta_i &= \tan^{-1}\left(\frac{\widehat{v}_y^* - m_y}{\widehat{v}_x^* - m_x}\right) \\ &= \begin{cases} g_v(\mathbf{x}, \mathbf{m}_i) & \text{if } \boldsymbol{\ell}_i \text{ is vertical, } \widehat{\mathbf{v}}^* = \mathbf{v}_v^*(\mathbf{x}) \text{ in (3.5)} \\ g_h(\mathbf{x}, \mathbf{v}_h^k, \mathbf{m}_i) & \text{if } \boldsymbol{\ell}_i \text{ is horizontal, } \widehat{\mathbf{v}}^* = \mathbf{v}_h^*(\mathbf{x}, \psi_k) \text{ in (3.9)} \end{cases}\end{aligned} \tag{3.13}$$

Since generally $\mathbf{v}_h^k$ is not on the ground plane $\mathcal{W}$ constructed by the attitude $\mathbf{x}$ in the Gaussian sphere, the desired $\widehat{\mathbf{v}}^*$ for the horizontal edge is obtained from the projection of $\mathbf{v}_h^k$ on $\mathcal{W}$ as seen in Figure 3.7(a). In other words, $\widehat{\mathbf{v}}^*$ on an image plane is a intersection point between the following two lines: the horizon $\mathbf{h}$ equivalent to the attitude $\mathbf{x}$ and the line connecting $\mathbf{v}_h^k$ and the camera center in Figure 3.7(b). Then, ψ_k is computed as

$$\psi_i = \tan^{-1}\big(\cos\theta\sqrt{(v_1^2 + v_2^2)/v_3^2 - \tan^2\theta}\,\big) \tag{3.14}$$

when $\mathbf{v}_h^k = (v_1, v_2, v_3)$ and $\mathbf{x} = (\phi, \theta)$.

The measurement of a line slope α_i is assumed to be perturbed by a Gaussian noise n_ℓ:

$$\alpha_i = \beta_i + n_\ell \qquad \text{for } i = 1, \ldots, n. \tag{3.15}$$

The variance of n_ℓ is set to be inversely proportional to the line length, $n_\ell = \lambda s_i^{-1}$, since the angle of a longer line segment is less likely to be perturbed in an image processing step.

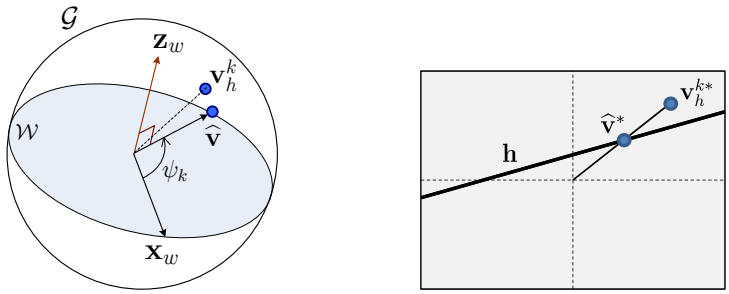
The Jacobian $\mathbf{J}$ in the EKF update is a partial derivative of β_i with respect to $\mathbf{x}$ and $\mathbf{b}$. Since the bias $\mathbf{b}$ is not involved in the update, two different Jacobians with respect to $\mathbf{x}$ are obtained for vertical and horizontal line measurements, respectively, as follows:

$$\mathbf{J}_v = \frac{\partial g_v}{\partial \mathbf{x}} = \left[\frac{-r^2 + r n_y}{d^2}, \quad (\frac{1}{c_\theta^2})\frac{r^2 n_x}{d^2}\right] \tag{3.16}$$

$$\mathbf{J}_h = \frac{\partial g_h}{\partial \mathbf{x}} = \left[\frac{-r^2 + (t_\psi/c_\theta)n_x - t_\theta n_y}{d^2}, \ \frac{n_x + t_\psi(-c_\theta + s_\theta n_y)}{(c_\theta d)^2}\right] \tag{3.17}$$

where r, d, n_x and n_y are defined in Figure 3.6. See Appendix-A.2 for the derivations of $\mathbf{J}_v$ and $\mathbf{J}_h$ in details.

The innovation in the Kalman update is a deviation angle e. For all the line segments in $\mathbf{L}$, the measurement update is sequentially repeated with different noise variance n_ℓ, which is inversely weighted by the line length s.



(a) On a Gaussian sphere (b) On an image plane

Figure 3.7: Determination of a desired vanishing point $\widehat{\mathbf{v}}^*$ in the EKF update for horizontal line measurements associated with $\mathbf{v}_h^k$. The $\widehat{\mathbf{v}}$ ($\widehat{\mathbf{v}}^*$ on the image) is a projection of $\mathbf{v}_h^k$ on the the world ground plane $\mathcal{W}$ ($\mathbf{h}$ on the image) which is equal to a current attitude estimate $\mathbf{x}$.

3.4 Line Classification

This section describes how to detect vanishing points and classify line segments into vertical or horizontal edges. The classified line measurements will proceed to the EKF update in Section 3.3. We present a hierarchical clustering method which recursively splits the lines into vertical or horizontal groups in a greedy manner. At each level of hierarchy, a RANSAC-based vanishing point detection is performed for line grouping and outlier rejection with various accuracy.

In the situation of continuous attitude estimation using an IMU and image stream, one main feature of the proposed method is to use a current attitude estimate as the prior for a vertical vanishing point in order to achieve fast and reliable clustering. Since natural urban images may contain outliers that are neither aligned nor orthogonal to the gravitational direction of the earth, this prior and the VP constraint (3.1) can effectively exclude invalid candidates for a true vanishing point in the RANSAC-based method. Moreover, the hierarchical clustering splits the lines based on this prior at the top level.

3.4.1 RANSAC-based vanishing point detection

Suppose n line segments $\mathbf{L} = (\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_n)$ are given in the presence of outliers. Two additional inputs are provided for an efficient VP hypothesis generation; a coarse estimate of the vertical VP and the type of a vanishing point we seek.

Algorithm 3.1 describes a hypothesis-and-test scheme in RANSAC [24] for the detection of a single major vanishing point. Two lines ($\boldsymbol{\ell}_i$, $\boldsymbol{\ell}_j$) are randomly sampled to produce one VP hypothesis, $\mathbf{v}_c = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j$. This candidate is tested for all the lines in $\mathbf{L}$ to count the number of inliers. An inlier set with respect to $\mathbf{v}_c$ is determined by whether or not the deviation angle e is less than an error threshold e_{th} (see Figure 3.6). After a preset number of trials are repeated, the best VP is selected as one that has a maximum number of inliers. Similar approaches can be

Algorithm 3.1: RANSAC for detecting a major vanishing point

$(\mathbf{v}_{best}, \mathbf{I}_{best})$ = RANSAC_VP($\mathbf{L}$, e_{th}, $\overline{\mathbf{v}}$, vp_type)
$\mathbf{L}$: line segment data $(\boldsymbol{\alpha}, \mathbf{m}, \mathbf{s})$ in (3.12), e_{th}: error threshold
$\overline{\mathbf{v}}$: vertical VP prior, vp_type: vertical or horizontal
begin
  Set N_{trial}, ε_{max}, $n_{max} = 0$
  for $trial = 1$ **to** N_{trial} **do**
    $\mathbf{v}_c = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j \leftarrow (i, j)$ = random_pair()
    $\mathbf{v}_c \leftarrow \mathbf{v}_c / \|\mathbf{v}_c\|$
    if vp_type = *vertical* **then** $\varepsilon = \cos^{-1}(\mathbf{v}_c^\top \overline{\mathbf{v}})$
    else $\varepsilon = \pi/2 - \cos^{-1}(\mathbf{v}_c^\top \overline{\mathbf{v}})$
    if $|\varepsilon| < \varepsilon_{max}$ **then**
      $\mathbf{e}$ = align_error ($\mathbf{v}_c$, $\mathbf{L}$);
      $\mathbf{I}$ = find ($\mathbf{e} < e_{th}$);
      if $|\mathbf{I}| > n_{max}$ **then**
        $n_{max} = |\mathbf{I}|$
        $\mathbf{I}_{best} = \mathbf{I}$, $\mathbf{v}_{best} = \mathbf{v}_c$

(Note: $\mathbf{I}$ is a line index list, $|\mathbf{I}|$ is the number of the index list.)

found in literature [25, 26, 27].

For efficient hypothesis generation using the prior $\overline{\mathbf{v}}$, we discard implausible candidates whose distance from $\overline{\mathbf{v}}$ are greater than ε_{max}. The distance ε between $\mathbf{v}_c$ and $\overline{\mathbf{v}}$ is defined differently depending on the given VP type. For a vertical VP, the distance is the angle between $\mathbf{v}_c$ and $\overline{\mathbf{v}}$, $\varepsilon = \cos^{-1}(\mathbf{v}_c^\top \overline{\mathbf{v}})$. For a horizontal VP, the distance is the angle between $\mathbf{v}_c$ and the plane normal to $\overline{\mathbf{v}}$, $\varepsilon = \pi/2 - \cos^{-1}(\mathbf{v}_c^\top \overline{\mathbf{v}})$. This simple rejection rule significantly reduces invalid VP candidates that are highly likely to be generated by outliers or a combination of horizontal and vertical edges. Note that $\mathbf{v}_c$ and $\overline{\mathbf{v}}$ are normalized in homogeneous coordinates so that $\|\mathbf{v}_c\| = \|\overline{\mathbf{v}}\| = 1$.

3.4.2 Hierarchical line grouping

We use a hierarchical clustering method that partitions the lines $\mathbf{L}$ into vertical and horizontal groups in a greedy manner based on the vertical VP prior ($\overline{\mathbf{v}}$). At the top level, this prior $\overline{\mathbf{v}}$ makes a coarse division of $\mathbf{L}$ into two non-overlapping groups: an uppermost vertical and a horizontal line group in which corresponding VP types (vertical or horizontal) are labeled but their vanishing points are not searched. The accuracy of $\overline{\mathbf{v}}$ is not essential for this division, but generally allows up to 20-degree error from a true vertical VP in a Gaussian sphere.

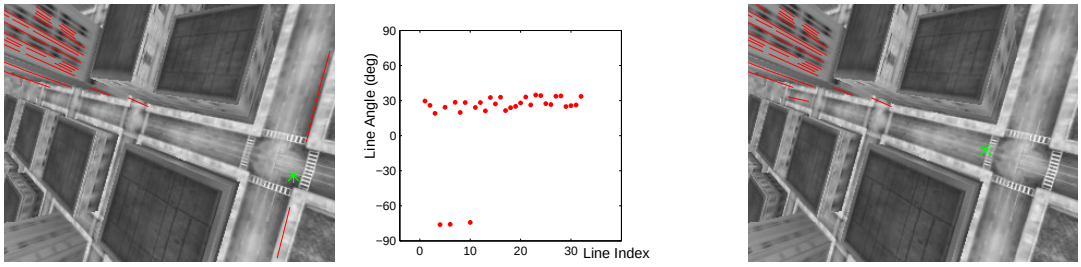
At lower levels, each group is divided into multiple subgroups with VP detection. The

Algorithm 3.2: Hierarchical line clustering with multiple vanishing point detection

$\mathbf{G}$ = Hierarchical_Clustering($\mathbf{L}$, $\overline{\mathbf{v}}$)
$\mathbf{L}$: line segment data $(\mathbf{m}, \boldsymbol{\alpha}, \mathbf{s})$, $\overline{\mathbf{v}}$: vertical VP prior
begin
  Set e_{th}, $n_{\ell,min}$ and α_{max}. $\mathbf{G} = \emptyset$, $\mathbf{I} = \mathbf{U}(\mathbf{L})$, $level = 0$;
  $\mathbf{I}_v$ = find(align_error $(\overline{\mathbf{v}}, \mathbf{L}) < e_{th}$);
  $\mathbf{G}$ = add_line_group($\emptyset$, $\mathbf{I}_v$, vertical, level);
  $\mathbf{G}$ = add_line_group($\emptyset$, $\mathbf{I} - \mathbf{I}_v$, horizontal, level);
  for $level = 1$ **to** L_{max} **do**
    foreach g *in* $\mathbf{G}^{level}$ **do**
      $\mathbf{I} \leftarrow \mathbf{I}(\mathbf{g})$, vp_type = vp_type($\mathbf{g}$);
      repeat
        $(\mathbf{v}_{best}, \mathbf{I}_{best})$ = RANSAC_VP($\mathbf{L}(\mathbf{I})$, $\overline{\mathbf{v}}$, $e_{th}/2^{level}$, vp_type);
        $\mathbf{I}_{best}$ = find(align_error $(\mathbf{v}_{best}, \mathbf{L}) < e_{th}/2^{level}$);
        $(\mathbf{I}_1, \mathbf{I}_2)$ = K_means_clustering $(\boldsymbol{\alpha}, \mathbf{I}_{best}, k = 2)$;
        if $|\text{mean}(\boldsymbol{\alpha}(\mathbf{I}_1)) - \text{mean}(\boldsymbol{\alpha}(\mathbf{I}_2))| > \alpha_{max}$ **then**
          $\mathbf{I}_{best} = \mathbf{I}_1$;
        $\mathbf{v}_{best}$ = SVD_weighted $(\mathbf{L}, \mathbf{I}_{best})$;
        $\mathbf{G}$ = add_line_group($\mathbf{v}_{best}$, $\mathbf{I}_{best}$, type, level);
        $\mathbf{I} \leftarrow (\mathbf{I} - \mathbf{I}_{best})$;
      until $|\mathbf{I}| < n_{\ell,min}$
  foreach g *in* $\mathbf{G}$ **do**
    remove $\mathbf{g}$ from $\mathbf{G}$ if $\mathbf{g}$ has a child or $|\mathbf{I}(\mathbf{g})| < n_{\ell,min}$.
  forall the *group pair* $(\mathbf{g}_i, \mathbf{g}_k)$ *in* $\mathbf{G}$ **do**
    merge if $\cos^{-1}(\mathbf{v}(\mathbf{g}_i)^\top \mathbf{v}(\mathbf{g}_k)) < d_{min}$.

RANSAC in Algorithm 3.1 is repeated until remaining lines in the group are less than a threshold $n_{\boldsymbol{\ell},min}$. As a level moves further down, the inlier threshold e_{th} and the minimum group size $n_{\boldsymbol{\ell},min}$ become smaller in order to achieve a higher-resolution line grouping. The VP type of each group is inherited from its parent group and is thus the same as that of its upper-most group. Therefore, every subgroup tends to be composed exclusively of either vertical or horizontal edges depending on the type of its uppermost group. The RANSAC also tries to find a vanishing point using $\overline{\mathbf{v}}$ and the given VP type.

Algorithm 3.2 shows a procedure that builds a hierarchy of line clusters while the splits are performed recursively by the RANSAC as moving down the hierarchy. The following additional steps take place after the RANSAC at each group division.



(a) (b) (c)

Figure 3.8: Effect of K-means clustering ($k = 2$) in Algorithm 3.2. VPs are marked by green. (a) Inlier size in the RANSAC is maximized by including outliers (horizontal edges) when $e_{th} = 2.5\,\text{deg}$. (b) K-means clustering with respect to α shows that the mean difference is larger than the maximum line angle divergence $\alpha_{max} = 60\,\text{deg}$. (c) The smaller group (horizontal edges) in K-means clustering is removed when α_{max} is violated.

- Group membership $\mathbf{I}_{best}$ is recomputed for the entire line segment with respect to the $\mathbf{v}_{best}$ returned by RANSAC. This allows groups to overlap and provides a chance to include the members missed from its parent node.
- The VP representing the group ($\mathbf{v}_{best}$) is refined as a least squares solution of $\mathbf{A}\mathbf{v} = \mathbf{0}$ where each row of $\mathbf{A}$ is equal to $\mathbf{A}_i = s_i \boldsymbol{\ell}_i^\top, i \in \mathbf{I}_{best}$ in SVD_weighted().
- As exemplified in Figure 3.8, K-means clustering with respect to line angle α is intended to cope with the case that the number of inliers is maximized via including both vertical and horizontal edges. To remedy this case, we assume that a bundle of lines sharing a VP has a bounded range of line angle (α_{max}) when a camera's field-of-view is limited. Hence, the smaller-size group is tested for the possibility to merge with other groups only if the mean difference between two partitions of line angle α is larger than α_{max}.

Once the hierarchical clustering is complete, we first choose all the bottom-level nodes that have no children, and remove small-sized nodes among them since their VPs are supported by insufficient line members. Because a vertical line group should exist uniquely, all the groups labeled as vertical are merged and filtered by the K-means clustering. The horizontal line groups are also merged if the distances between corresponding VPs are closer than a threshold d_{min}, *i.e.*, $\cos^{-1}(\mathbf{v}_h^{i\top}\mathbf{v}_h^j) < d_{min}$. Eventually, the lines are clustered into a single vertical and multiple horizontal line groups, $\mathbf{G} = (\mathbf{g}_v, \mathbf{g}_h^1, \ldots, \mathbf{g}_h^m)$.

Some tree diagram examples and their select line groups are shown in Figure 3.9 and 3.10 for simulated images and real aerial images, respectively.

3.5 Horizon Measurement

The vanishing points in $\mathbf{G}$ ($\mathbf{v}_{best}$ in each line group) that resulted from the line clustering do not necessarily satisfy the VP constraint (3.1) even though this constraint has been used to reject invalid VP candidates. Hence, we seek a set of vanishing points $\mathbf{H} = (\mathbf{v}_v, \mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$ that are strictly subject to $\mathbf{v}_v^\top \mathbf{v}_h^i = 0$ for $i = 1, \ldots, m$, and we call $\mathbf{H}$ a *horizon measurement*. On an image plane, $\mathbf{H}$ is represented by a horizon line $\mathbf{h}$. This $\mathbf{h}$ is equivalent to $\mathbf{v}_v$ in $\mathbf{H}$ and is also the line on which all the horizontal VPs $(\mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$ lie.

To find the $\mathbf{H}_{best}$ that best describes the classified line groups $\mathbf{G}$, we propose another RANSAC-based hypothesis-and-test scheme that efficiently generates horizon measurement hypotheses from $\mathbf{G}$ in a principled way. When a camera is the only available sensor, this scheme serves as a vision-only estimation method that uses line measurements only and yields the best attitude estimate from $\mathbf{v}_v$ in $\mathbf{H}_{best}$.

3.5.1 Horizon hypothesis generation

Suppose a set of line groups $\mathbf{G} = (\mathbf{g}_v, \mathbf{g}_h^1, \ldots, \mathbf{g}_h^m)$ are given by the line clustering method in Section 3.4. There are following three ways to produce a horizon candidate $\mathbf{h}_c$ through sampling classified line segments in $\mathbf{G}$:

- Two vertical edges: $\mathbf{h}_c = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j,\ (i,j) \in \mathbf{g}_v$.
- One vertical edge and one horizontal vanishing point: $\mathbf{h}_c = \boldsymbol{\ell}_i \times \mathbf{v}_h^k,\ i \in \mathbf{g}_v$.
- Two horizontal vanishing points: $\mathbf{h}_c = \mathbf{v}_h^i \times \mathbf{v}_h^j$.

The first case produces a vertical VP from the intersection of two vertical edges. The second case arises from the fact that a vertical line $\boldsymbol{\ell}_i$ is *dual* to a point on the horizon, and thus two points, $\boldsymbol{\ell}_i$ and $\mathbf{v}_h^k$, lie on the horizon h_c. The last case provides the capability to derive $\mathbf{v}_v$ even when no vertical line group exists in $\mathbf{G}$ (*e.g.* a camera is looking down toward the ground). Note that a horizontal VP is also randomly sampled from two horizontal edges in the same group, $\mathbf{v}_h^k = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j,\ (i,j) \in \mathbf{g}_h^k$.

A horizon measurement candidate $\mathbf{H}_c$ is composed of a vertical VP that is equal to $\mathbf{h}_c$, and horizontal VPs that are generated by the intersections between $\mathbf{h}_c$ and line segments randomly sampled from each horizontal line group, *i.e.*,

$$\mathbf{H}_c = (\mathbf{v}_v, \mathbf{v}_h^1, \ldots, \mathbf{v}_h^m) \text{ s.t. } \mathbf{v}_v = \mathbf{h}_c,\ \mathbf{v}_h^k = \mathbf{h}_c \times \boldsymbol{\ell}_i, i \in \text{random}(\mathbf{g}_h^k), \quad \text{for } k = 1, \ldots, m \tag{3.18}$$

which satisfies the VP constraint (3.1). The number of candidates for $\mathbf{H}$ becomes easily intractable due to combinatorial explosion when the three cases for $\mathbf{h}_c$ and the random line sampling for $\mathbf{v}_h^k$ are considered. To create a feasible number of hypotheses, we set the maximum samples of a single line to $n_{v,max}$ and $n_{h,max}$, and line pairs to $n_{vv,max}$ and $n_{hh,max}$, respectively.

In Algorithm 3.3, the horizontal VP candidates $(\mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$ in $\mathbf{H}_c$ are generated and tested at the same time in the hypothesis test function for the computational efficiency described in the following section.

3.5.2 Horizon hypothesis test

A set of horizon measurement candidates $\mathbf{H}_c^*$ is evaluated based on the following score function:

$$\text{Score}(\mathbf{H}_c; \mathbf{G}) = C_v + \sum_{k=1}^{m} C_h^k = \sum_{\boldsymbol{\ell}_j \in \mathbf{g}_v} f(\mathbf{v}_v, \boldsymbol{\ell}_j) \; + \; \sum_{k=1}^{m} \left[\sum_{\boldsymbol{\ell}_j \in \mathbf{g}_h^k} f(\mathbf{v}_h^k, \boldsymbol{\ell}_j) \right] \tag{3.19}$$

$$f(\mathbf{v}_i, \boldsymbol{\ell}_j) = \begin{cases} s_j \left(1 - \dfrac{e(\mathbf{v}_i, \boldsymbol{\ell}_j)}{e_{th}} \right) & : \text{if } e < e_{th} \\ 0 & : \text{otherwise} \end{cases} \tag{3.20}$$

where e is an alignment angle error of the line $\boldsymbol{\ell}_j$ to $\mathbf{v}_i$ and s_j is a line length of $\boldsymbol{\ell}_j$ as shown in Figure 3.6. A similar function is used in Rother's work [25]. The line length in the score reflects that a true VP is more closely positioned in the direction of the longer line segment, and the score is irrelevant to outliers whose alignment error is larger than e_{th}.

The score function for each $\mathbf{v}_i$ in $\mathbf{H}_c$ accounts only for the corresponding line group $\mathbf{g}_i$ in $\mathbf{G}$, but ignores the other groups completely. This is because the line segments are assumed to already be properly clustered into $\mathbf{G}$ and e_{th} rejects outliers in the score, if any. Therefore, as implemented in Algorithm 3.3, the maximum score per horizontal group ($C_{h,best}^k$) can be evaluated simultaneously with the generation of $\mathbf{v}_h^k$ candidate in $\mathbf{H}_c$.

Once all the hypotheses are tested, the hypothesis with the maximum score is selected as $\mathbf{H}_{best}$ which best explains both vertical and horizontal line measurements in terms of the score function. In Figure 3.10, the top five horizon candidates are listed in each image.

Vision-only vs. visual-inertial method:

The procedure to find $\mathbf{H}_{best}$ can be considered as a vision-only method, and $\mathbf{v}_v$ in $\mathbf{H}_{best}$ is an optimal solution for the attitude estimation that uses line segments only.

In the visual-inertial method, $\mathbf{H}_{best}$ and $\mathbf{G}$ are inputs to the Kalman update in Section 3.3.2. Compared with $\mathbf{v}_{best}$ found in the line clustering in Section 3.4, $\mathbf{H}_{best}$ provides a more precise yaw angle ψ_k of each horizontal vanishing point $\mathbf{v}_h^k$ at the Kalman update in (3.14). From the benefits of the sensor fusion, the advantages of the visual-inertial method include a higher estimation rate, smaller estimation error, and the ability to cope with occasional textureless images. Performance comparison between the vision-only and visual-inertial method will be presented in Section 3.7.

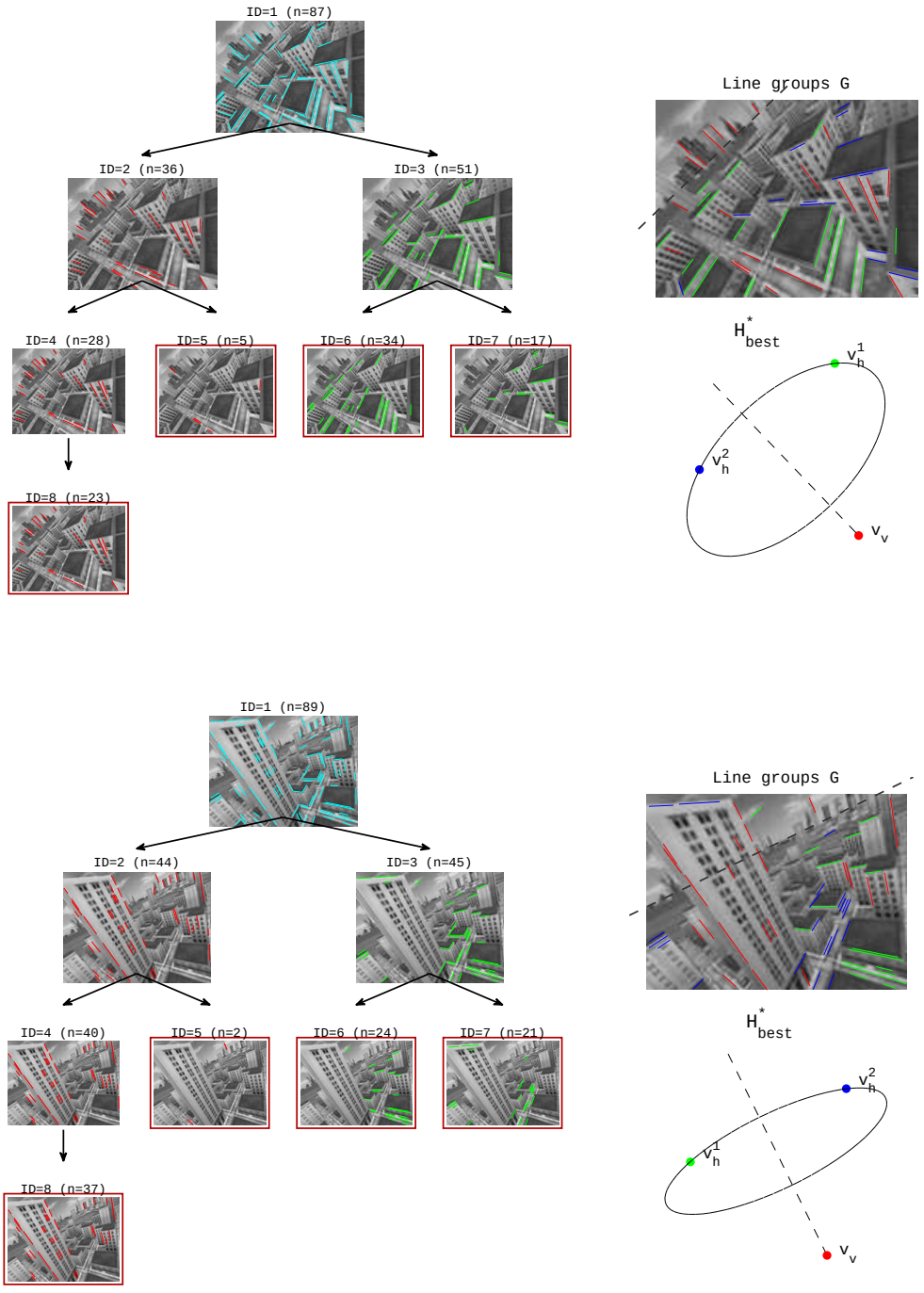


Figure 3.9: Hierarchical line clustering results on two simulated Manhattan-world images: (left) hierarchy of line clusters, (right top) classified line groups $\mathbf{G}$, (right bottom) $\mathbf{H}_{best}$ on Gaussian sphere.

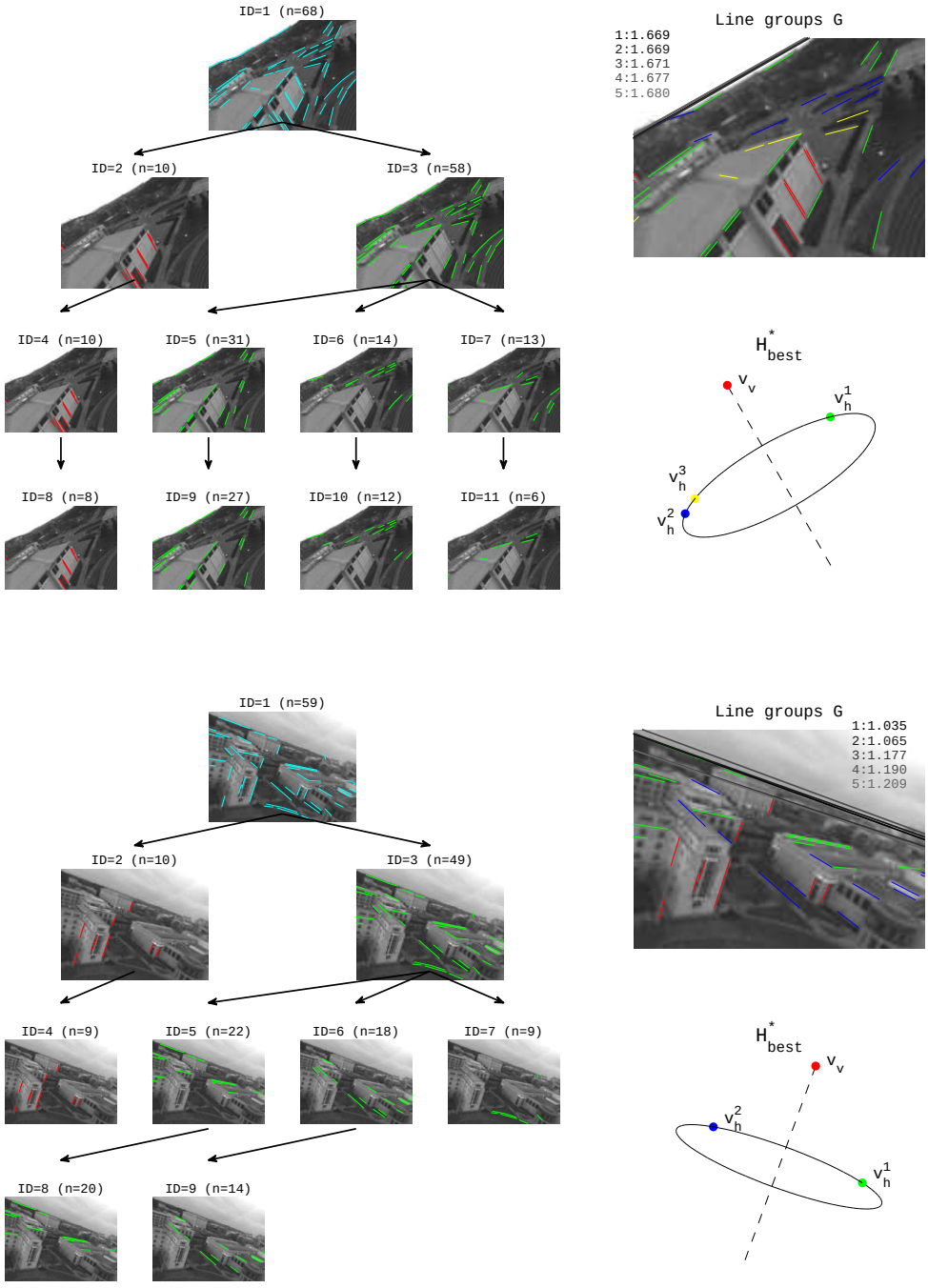


Figure 3.10: Hierarchical line clustering results on two aerial urban images: (left) hierarchy of line clusters, (right top) classified line groups **G** and top-five horizon candidates **H**, (right bottom) $\mathbf{H}_{best}$ on Gaussian sphere.

Algorithm 3.3: Horizon hypothesis generation and test for vision-only attitude estimation

$\{\mathbf{h}_c^1, \ldots, \mathbf{h}_c^{p_{max}}\}$ = Horizon_Hypothesis_Generation($\mathbf{G}$, $\mathbf{L}$)
begin
  Set $n_{v,max}$, $n_{vv,max}$, and $n_{hh,max}$, $p = 0$;
  foreach *vertical line group* $\mathbf{g}_v$ *in* $\mathbf{G}$ **do**
    for $u = 1$ **to** $n_{vv,max}$ **do**
      $\mathbf{h}_c^p = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j \ \leftarrow (i,j) = \text{random\_pair}(\mathbf{g}_v);\ p = p + 1;$
  foreach *horizontal line group* $\mathbf{g}_h$ *in* $\mathbf{G}$ **do**
    for $u = 1$ **to** $n_{hh,max}$ **do**
      $\mathbf{v}_h = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j \ \leftarrow (i,j) = \text{random\_pair}(\mathbf{g}_h);$
      foreach *vertical line group* $\mathbf{g}_v$ *in* $\mathbf{G}$ **do**
        for $v = 1$ **to** $n_{v,max}$ **do**
          $\mathbf{h}_c^p = \mathbf{v}_h \times \boldsymbol{\ell}_p \ \leftarrow p = \text{random}(\mathbf{g}_v);\ p = p + 1;$
  foreach *horizontal line group pair (*$\mathbf{g}_h^1$ *and* $\mathbf{g}_h^2$*) in* $\mathbf{G}$ **do**
    for $u = 1$ **to** $n_{hh,max}$ **do**
      $\mathbf{v}_h^1 = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j \ \leftarrow (i,j) = \text{random\_pair}(\mathbf{g}_h^1);$
      for $v = 1$ **to** $n_{hh,max}$ **do**
        $\mathbf{v}_h^2 = \boldsymbol{\ell}_i \times \boldsymbol{\ell}_j \ \leftarrow (i,j) = \text{random\_pair}(\mathbf{g}_h^2);$
        $\mathbf{h}_c^p = \mathbf{v}_h^1 \times \mathbf{v}_h^2,\ p = p + 1;$

$\mathbf{H}_{best}$ = Horizon_Hypothesis_Test($\{\mathbf{h}_c^1, \ldots, \mathbf{h}_c^{p_{max}}\}$, $\mathbf{G}$, $\mathbf{L}$)
begin
  n_h = number of horizontal line groups in $\mathbf{G}$
  for $p = 1$ **to** p_{max} **do**
    $\mathbf{v}_v^p = \mathbf{h}_c^p$
    $C_v = \sum_i \mathsf{Score}(\mathbf{v}_v^j,\ \boldsymbol{\ell}_i \in \mathbf{g}_v)$
    for $k = 1$ **to** n_h **do**
      $C_{h,best}^k = 0;$
      for $u = 1$ **to** $n_{h,max}$ **do**
        $\mathbf{v}_h = \mathbf{v}_v^p \times \boldsymbol{\ell}_i \ \leftarrow i = \text{random}(\mathbf{g}_h^k);$
        $C_h^k = \sum_i \mathsf{Score}(\mathbf{v}_h,\ \boldsymbol{\ell}_i \in \mathbf{g}_h^k)$
        if $C_h > C_{h,best}^k$ **then**
          $C_{h,best}^k = C_h$
          $\mathbf{v}_{h,best}^k = \mathbf{v}_h$
    $\mathbf{H}_p = (\mathbf{v}_v^k, \mathbf{v}_{h,best}^1, \ldots, \mathbf{v}_{h,best}^k)$
    $C_p = C_v + \sum_k C_{h,best}^k$
  $\mathbf{H}_{best} = \text{argmax}(C_p);$

3.6 Performance Evaluation in Graphical Simulator

The simulation evaluation was performed for the visual/inertial attitude estimation using a graphical flight simulator. This evaluation is focused on the precision and accuracy of our estimation method when an UAV continuously conducts highly dynamic maneuvers in an urban setting.

The airplane was driven manually by a computer-connected RC controller to generate aggressive motions resulting in a large swing of pitch and roll angles up to ± 120 and ± 60 degrees, respectively. The output of a tri-axial gyroscope was sampled at 100Hz. A large gyroscope noise (0.05 rad/s) was intentionally imposed in order to demonstrate the effectiveness of visual attitude sensing in short time intervals (the noise of a MEMS-based gyroscope is typically 0.01 rad/s). A vision-processing pipeline including edge detection and line segments fitting was identical to that in a real experiment, except that rendered images were captured. Figure 3.11 shows some samples of rendered images on which three line groups $(\mathbf{g}_v, \mathbf{g}_h^1, \mathbf{g}_h^2)$ and the true horizon $\mathbf{h}$ are overlaid.

Estimation performance:

Figure 3.12 shows the estimation performance of the visual/inertial estimation method over a 30-second flight. Note that, every 10 seconds, the EKF restarts with an initial 10-degree attitude error and 0.1 rad/s gyroscope bias error. This is intended to investigate how fast the visual Kalman update responds to an initial error. The noises in the EKF are set to $[\mathbf{n}_x^\top \ \mathbf{n}_b^\top] = [0.01 \ 0.01 \ 0.02 \ 0.02 \ 0.02]^\top$ rad/s, and $n_\ell = 10/s_i$ for a line measurement $\boldsymbol{\ell}_i$.

The initial attitude error almost diminishes less than one second corresponding to 30 Kalman updates. The bias error takes about 3 to 5 seconds to reduce to nearly zero. Once the initial error diminishes, estimation errors are always bounded within 3 degrees for both roll and pitch angles and their variances are also less than 2 degrees on average. The higher estimate variance in pitch than in roll demonstrates that the pitch is more sensitive to noises of line measurements and vanishing point location. This is also explained by a high sensitivity of the Jacobian $\mathbf{J}_v$ in (3.16) for a large pitch angle.

The average number of line segments per image is about 40. The sizes of the vertical and first two horizontal line groups in Figure 3.12(f) show that, in most cases, $\mathbf{H}$ consists of both $\mathbf{v}_v$ and $\mathbf{v}_h$s from a result of the hierarchical line clustering.

Fully and partially observable measurements:

A set of line segments holds information about the attitude that can be abstracted from their vanishing points. A horizon measurement $\mathbf{H}$ holds the following three different levels of information about the attitude depending on its components:

- Redundant : $\mathbf{H}_{v+h} = (\mathbf{v}_v, \mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$

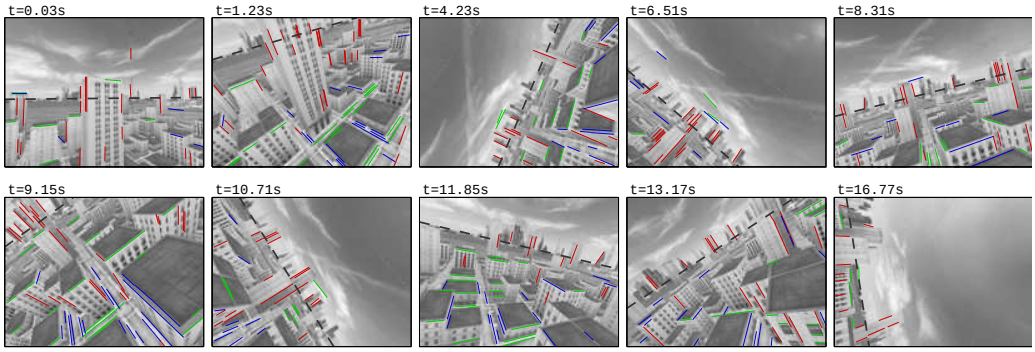


Figure 3.11: (Simulated images) A Manhattan world is rendered in our graphical simulator. The ground-truth horizon is a dotted black line. The lines associated with $\mathbf{v}_v$ are in red and the other lines associated with two $\mathbf{v}_h$s are in green and blue, respectively.

- Full : $\mathbf{H}_v = \mathbf{v}_v$ or $\mathbf{H}_{h+h} = (\mathbf{v}_h^1, \mathbf{v}_h^2)$
- Partial : $\mathbf{H}_h = \mathbf{v}_h^1$

In the redundant case $\mathbf{H}_{v+h}$, the attitude is overdetermined by multiple sources. For example, the attitude can be derived from either $\mathbf{v}_v$ or $(\mathbf{v}_h^1, \ldots, \mathbf{v}_h^m)$. In the full cases, $\mathbf{H}_v$ and $\mathbf{H}_{h+h}$, the attitude is uniquely determined. In the partial case $\mathbf{H}_h$, a single horizontal VP is not sufficient to determine the attitude. Nonetheless, $\mathbf{H}_h$ is still valid for the EKF update because Kalman filter enables the state to be estimated fully even from partial observation of the state.

In Table 3.2, attitude estimation performance is compared when partial, full or redundant information of the horizon measurement $\mathbf{H}$ are respectively used in the Kalman update. The use of line measurements is intentionally restricted to each case of $\mathbf{H}$, and thus all the irrelevant line measurements are discarded. For example, $\mathbf{H}_v$ uses only vertical line segments at the Kalman update. The comparison shows that no significant difference exists between the redundant and full cases in Table 3.2(c)-(e). In the partial case, $\mathbf{H}_h$, where only the first $\mathbf{v}_h$ is used in the Kalman update, no profound degradation of accuracy occurs while the estimation variance increases due to a small number of line measurements.

Table 3.2: Estimation performance comparison when partial, full or redundant information of the horizon measurement $\mathbf{H}$ is respectively used at the Kalman update in the simulation (errors are in degrees)

Measurement	ϕ_{err}	θ_{err}	$\overline{P_{\phi\phi}}$ (°)	$\overline{P_{\theta\theta}}$ (°)
(a) gyro only	27.28	23.71	–	–
(b) gyro + $\mathbf{H}_h$	2.92	2.54	2.61	4.48
(c) gyro + $\mathbf{H}_{h+h}$	1.56	1.53	0.81	1.35
(d) gyro + $\mathbf{H}_v$	1.10	0.96	0.75	1.07
(e) gyro + $\mathbf{H}_{v+h}$	0.90	0.93	0.49	0.73

$\overline{P_{\phi\phi}}$ and $\overline{P_{\theta\theta}}$ represent the means of EKF pitch and roll error variances.

Table 3.3: Computation time to produce classified vertical and horizontal line measurements and the best horizon measurement $\mathbf{H}_{best}$ in the real aerial experiment (tested for 320 × 240 images with a C implementation in Intel Core CPU 1.73 GHz).

	Time (msec)		
Processing step	Average	Deviation	Maximum
Canny edge detection	17.0	4.0	28.4
Recursive straight line fitting	1.7	0.7	5.5
Hierarchical line clustering	4.5	2.3	9.6
Horizon measurement	2.6	3.0	12.9
Total	25.8	7.5	56.4

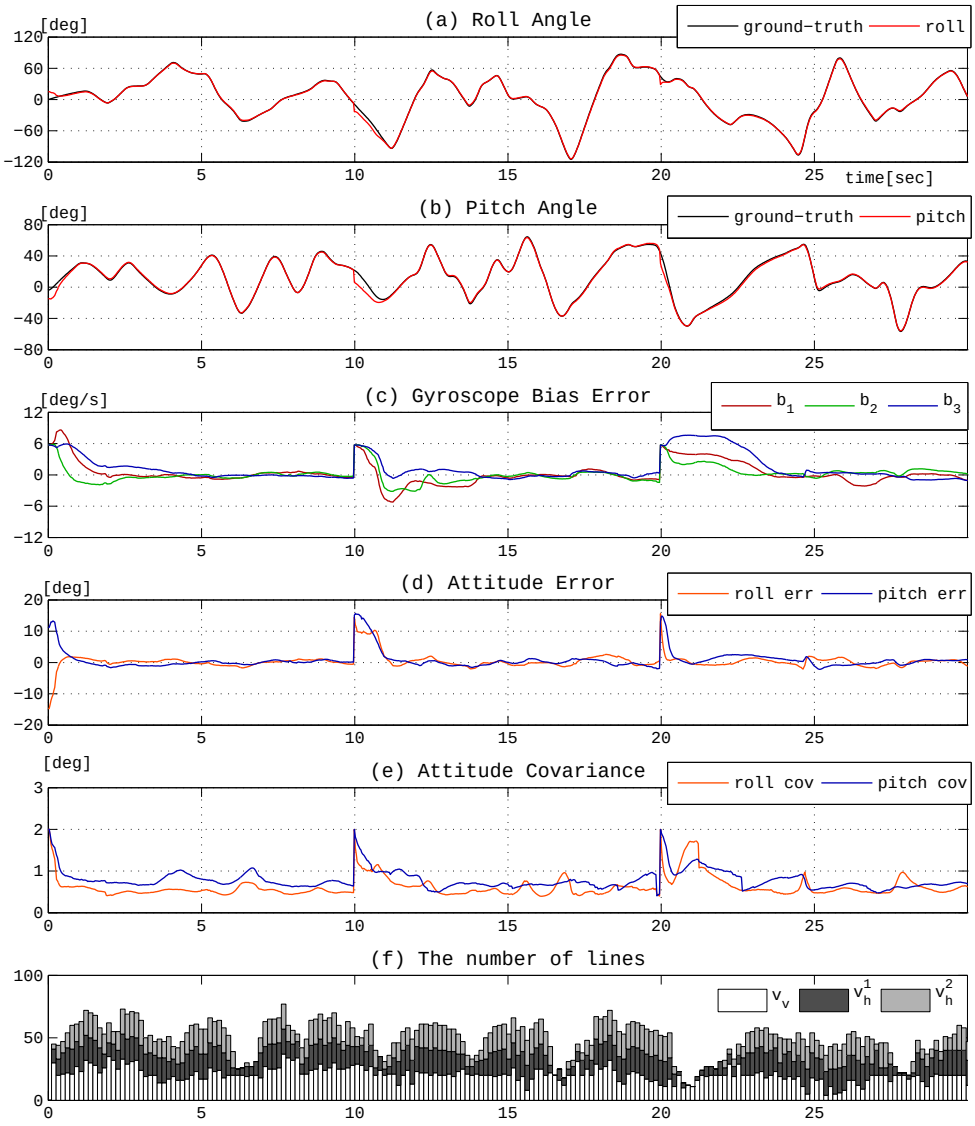


Figure 3.12: Simulation evaluation of the visual-inertial attitude estimation (gyro + $\mathbf{H}_{v+h}$): a small fixed-wing aircraft conducts highly dynamic maneuvers. Every 10 seconds, the Kalman filter restarts with an initial 10-degree attitude error and 0.1 rad/s gyroscope bias error.

3.7 Experiment Results

Firstly, the Canny edge detector was applied to raw images, followed by non-maximal suppression in order to obtain the edges of one-pixel thickness [28]. Prior to line fitting, these edge pixels were rectified and normalized using a camera calibration matrix $\mathbf{K}$ and distortion factors. Finally, each edge chain was recursively divided and fitted to line segments with sub-pixel accuracy based on the thresholds of a line fitting error and minimum line length (we use one pixel error and 8% of the image size, respectively).

3.7.1 Outdoor Urban Scene

An outdoor flight experiment was conducted over a football field surrounded by low-profile buildings and a stand. Some examples of our urban scene are shown in Figure 3.15. Building edges and lane markers on the football field were detected as main line features. Both the camera and IMU were calibrated in advance to obtain their intrinsic parameters and relative orientation. The ground-truth attitude of the experiment was obtained by a manual line classification of inlier line segments and finding a horizon that best explains these lines using (3.19). The boundary between the sky and ground and the perspective view of buildings enabled us to draw an implied true horizon line.

Figure 3.14 shows the estimation performance on a 60-second real aerial scene when the UAV circled around a football field and formed a figure eight. The average attitude errors were 2.63 and 2.24 degrees, and their maximum errors are limited to 8.56 and 5.39 degrees for the roll and pitch angle, respectively. The attitude error variance was always below 1.5 degrees. The gyroscope bias is transient at the beginning due to the initial attitude error, but became steady after 20 seconds. Table 3.3 shows the average and worst computation times spent on individual steps from a raw image until classified line measurements and the best horizon measurement were produced. Almost half the time was spent on the edge detection and the average total time 25.8 msec guaranteed the realtime processing at a video frame rate.

The sizes of classified line groups are shown in Figure 3.14(f). The line segments in each group were sorted by length and the maximum size of each line group was set to 20 discarding the shorter lines. On average, 50 line segments were found and the line length was 35 pixels in 320 $\times$240 images. Compared with the simulation, the experiment had a much smaller number of vertical edges, because the UAV flew high over the buildings and there were no tall buildings nearby. When only horizontal line groups were available, we observed that the attitude error and its variance increased since the Kalman update depends on the accuracy of the vanishing point detection.

Figure 3.13 demonstrates that a single horizontal VP in the EKF is sufficient to restrain the

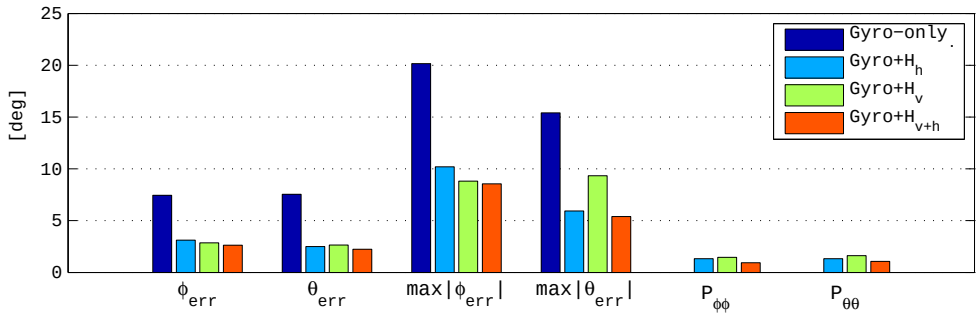


Figure 3.13: Estimation performance comparison between partial, full and redundant information of the horizon measurement $\mathbf{H}$ in the experiment (errors are in degrees).

attitude from gyroscope's unbounded drift (gyro + $\mathbf{H}_h$). Additional line measurements in $\mathbf{H}_{v+h}$ slightly reduce the attitude error and its variance. Figure 3.15 shows several select examples of the experimental result. The classified line segments in the middle column are inputs for the Kalman update. In the last column, the pole axis and great circle of a Gaussian sphere represents the EKF attitude estimate prior to the update step, and the points represent VPs in $\mathbf{H}$ obtained from the hierarchical line clustering. They are plotted together to illustrate the deviation between line measurements and attitude estimate at the prediction step. Their discrepancy indicates an innovation error in the EKF and corrects the gyroscope drift.

3.7.2 Comparison with inertial-only and vision-only methods

The vision-only method refers to the best horizon measurement $\mathbf{H}_{best}$ in Section 3.5. The inertial-only method tested in the paper is a nonlinear complementary filter that combines the direction of an estimated gravity $\mathbf{a}_g$, which elimnates centrifugal acceleration from accelerometer ouput $\mathbf{a}_{imu}$ using vehicle speed v and angular velocity $\boldsymbol{\omega}$ [4, 5]. Briefly speaking, a long-term drift is corrected by the following PI feedback of attitude error:

$$\mathbf{a}_g = \mathbf{a}_{imu} - \boldsymbol{\omega} \times [v\ 0\ 0]^\top \tag{3.21}$$

$$\mathbf{e} = \mathbf{R}_z \times \mathbf{a}_g \tag{3.22}$$

$$\boldsymbol{\omega} = \boldsymbol{\omega}_{imu} + \mathbf{K}_p\mathbf{e} + \mathbf{K}_i \int \mathbf{e} \tag{3.23}$$

In Figure 3.16 and 3.17, the sensor fusion in the visual-inertial method provides a better estimation accuracy and smaller maximum error than those of the single-sensor methods. The vision-only method has error peaks when line measurements are erroneous and insufficient to infer a correct horizon. The inertial-only method has relatively large errors when the airplane undergoes highly dynamic motions because the dynamic acceleration becomes dominant in accelerometer measurements and difficult to correctly be removed.

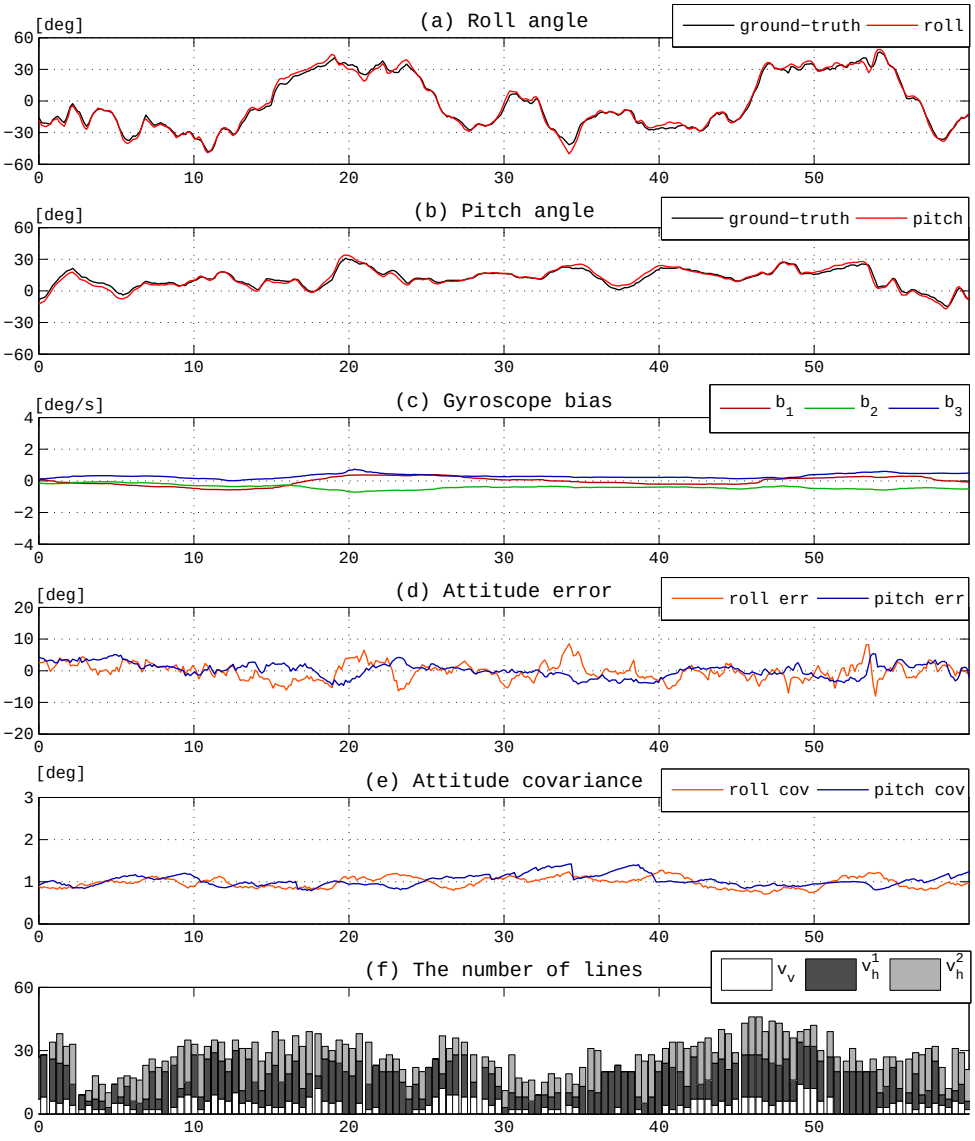


Figure 3.14: The performance of the visual-inertial attitude estimation method (gyro + $\mathbf{H}_{v+h}$) on the urban aerial scene in the experiment. The ground-truth on attitudes is given by manual fit of the horizon line on the images.

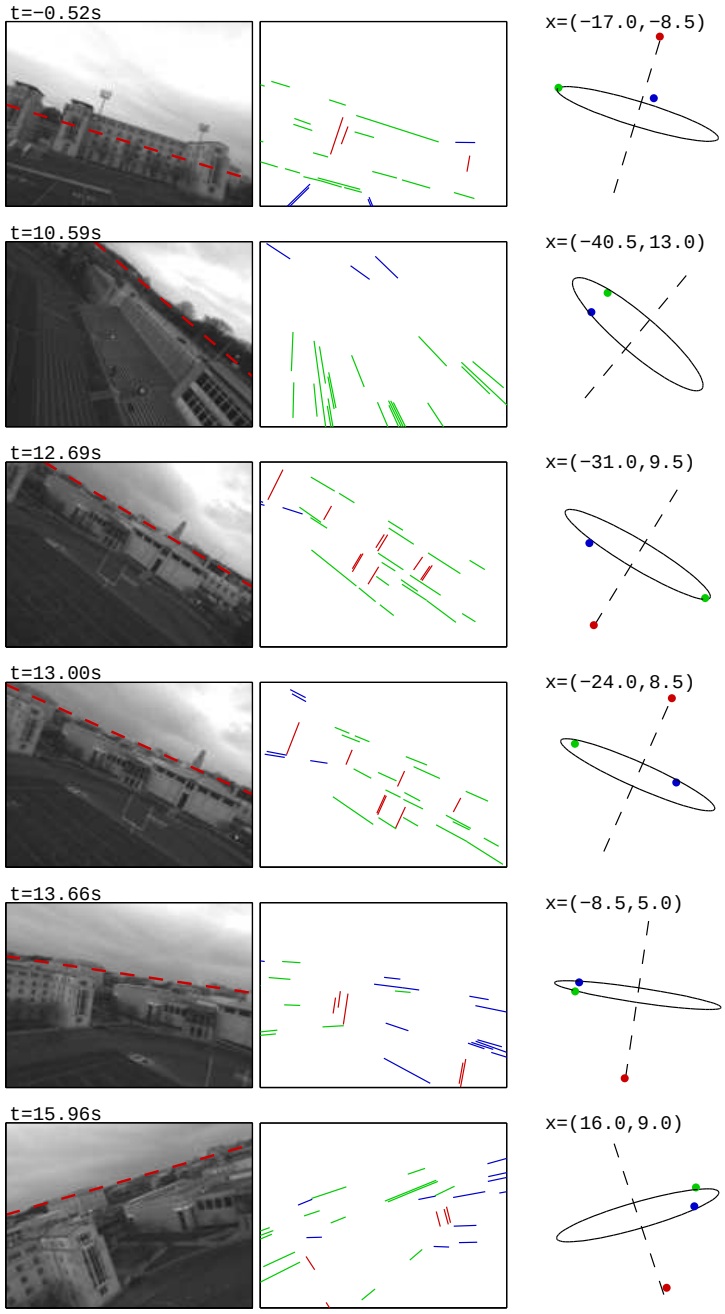


Figure 3.15: (Left) Aerial urban images over a football field and buildings. (Middle) Classified line segments after outlier rejection. The vertical line group is in red and horizontal line groups are in green or blue. (Right) The pole axis and great circle of the Gaussian sphere are the attitude estimate prior to the Kalman update. The points are the vanishing points in **H** estimated during the hierarchical line clustering.

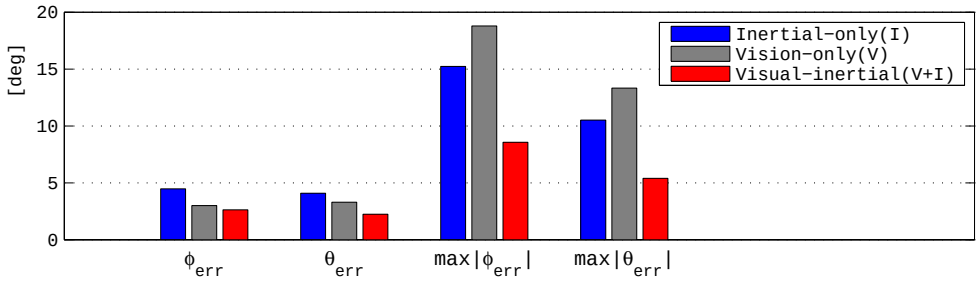


Figure 3.16: Comparison of average and maximum attitude estimation errors between the vision-only, inertial-only, and visual-inertial methods in the outdoor experiment.

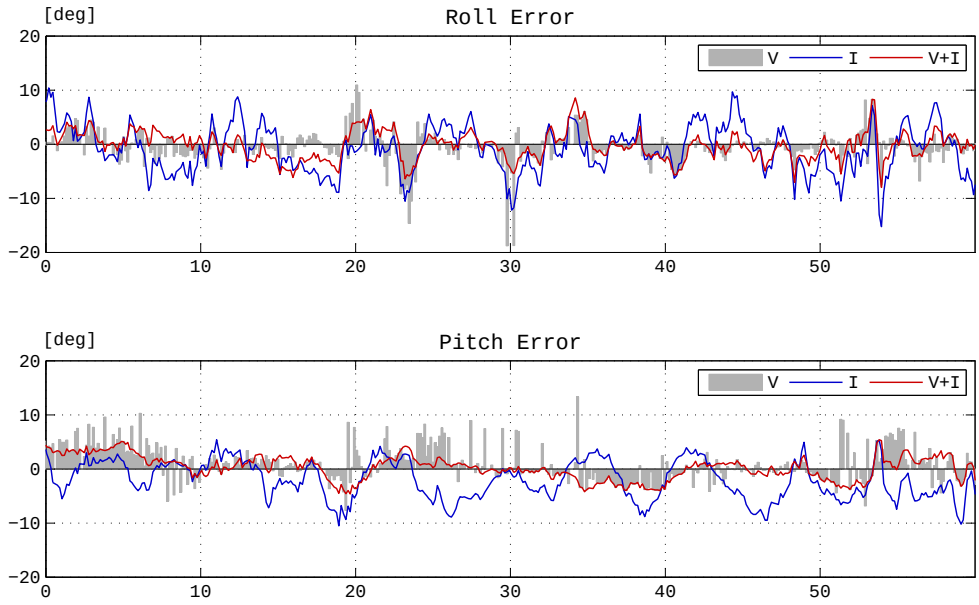


Figure 3.17: Attitude error plots of the vision-only, inertial-only, and visual-inertial attitude estimation methods in the outdoor experiment during 60 seconds.

These comparisons clearly demonstrate the value and power of sensor fusion. In low-cost small UAVs, it is hard to expect that any single sensor will be superior to other sensors in all the aspects of performance and working condition. For instance, high-precision gyroscope angles for a long period of time require enormous effort and time in sensor calibration and compensation of various noise effects like a stocastic bias fluctuation and earth rotation. The sensor fusion we propose makes it possible to effectively circumvent these highly scrutinized processes.

Chapter 4

Visual/Inertial Sensor Calibration

For the fusion of visual and inertial sensors, calibration is an indispensable step to integrate the measurements in a common frame of reference. In this chapter, an easy-to-use camera/IMU calibration procedure is presented for in-field automatic calibration, which uses gravity and natural landmarks as reference measurements.

Our interest is focused on self-calibration which refers to when no artificial calibration object or user interaction is required. Moreover, rather than calibrating each sensor independently, we examine a method to calibrate both sensors simultaneously. The calibration parameter consists of IMU shape and bias $(\mathbf{S}, \mathbf{b})$, camera matrix $\mathbf{K}$, and relative orientation $\mathbf{R}_{ic}$ between the sensors. Relative translation $\mathbf{t}_{ic}$ is excluded here, but can be estimated afterward, when necessary, once the other parameters are found.

The calibration procedure we propose begins with the factorization-based calibration, which recovers an IMU's intrinsic shape from a set of constraints regarding motion magnitude. The factorization method, which originated from shape-and-motion recovery in computer vision, exploits the fact that IMU measurement is the product of intrinsic shape parameters (scale, alignment and bias) and exerted motion (accelerations or angular velocities). We used the following two natural references for the self-calibration. Not only magnitudes of exerted loads on the IMU are known, but also abundant calibration dataset is readily collectable.

Gravity is a natural feature that is commonly involved with accelerometers and cameras. The magnitude of gravity is precisely known in a static condition. For accelerometer calibration, placing an IMU at various arbitrary attitudes produces a large number of calibration datasets from gravity loads in different directions. For a camera, the projection of gravity direction is equal to a vanishing point at which vertical edges in an urban image merges. Since the IMU calibration recovers gravity directions in terms of IMU coordinates, a camera calibration matrix is obtainable from the known Euclidean angles between vanishing points.

The feature track of a distant unknown scene point over images serves as a natural landmark

for both gyroscope and camera calibration. An infinite homography constructed from feature tracks not only contains the information about the magnitude of angular velocity but also provides constraints for a rotating camera calibration.

Compared with other previous methods, the main distinctive features of the proposed camera/IMU calibration are as follows:

- A self-calibration procedure using natural references such as gravity, vertical edge, and distant scene points in an urban environment
- No assumption on the configuration or number of an IMU's inertial components
- A linear solution for a redundant IMU
- An iterative linear solution for a triad IMU with an initial guess only for bias

Mathematical details on the factorization-based IMU calibration are described from Section 4.2 to 4.4 and the collaborative camera/IMU calibration procedure is described in Section 4.6.

4.1 Related Work

Traditionally, the calibration spends expertise, labor, time and external equipment to provide true motions for IMU calibration [29, 30, 31, 32, 33, 34] and known 3D feature points for camera/IMU calibration [35, 36, 37, 38, 39]. For example, precise sensor orientation in a static condition has been realized by a robotic manipulator [29] or a special mechanical platform [30]. IMU's navigational state (position, velocity and orientation) has also been compared with external high-precision devices like an optical tracking system [33, 34] or a speed-controlled turning table [40].

One prevalent approach for the IMU is self-calibration [41, 42, 43, 44]. For instance, a collection of partially known motions constitutes sufficient constraints on a parameter search. From highly abundant loads of known magnitude, a nonlinear least squares solution [42] and a sophisticated iterative method for an enlarged convergence region [44] were proposed. Their drawbacks are that a shape model is not scalable but is instead limited to a tri-axial configuration, and initial guess is also required for all of the parameters. As an alternative, a new perspective on self-calibration was explored by the factorization method [45, 46, 47, 48, 49, 50]. It has inspired force/torque sensor calibration [46, 47, 49] and object color modeling [48] such that a bilinear formulation of measurement is available. Motivated by the same principle of factorization, we propose a comprehensive solution for the IMU calibration in arbitrary configurations.

For a camera/IMU system, a multi-step approach [36] was presented to calibrate the parameters in the following order: IMU's intrinsic parameter using a pendulum, relative orientation using vertical edges and gravity, and then relative translation using a turntable and planer visual target. Although this method is able to obtain the entire intrinsic and extrinsic parameters, a

Table 4.1: Comparison between sensor calibration methods for inertial and visual sensors

	Baseline alg.	Reference	Parameters	Initial	SELF	COMP	CORR
Lang & Pinz [35]	Nonlinear optimization	Gravity, turntable	$\mathbf{S}, \mathbf{b}, \mathbf{K}, \mathbf{R}_{ic}$	Yes	No	Low	No
Lobo & Dias [36]	Two-step method	Vertical planar target, turntable	$\mathbf{S}, \mathbf{b}, \mathbf{K}, \mathbf{R}_{ic}, \mathbf{T}_{ic}$	Yes	No	Low	No
Mirzaei & Roumeliotis [37]	Extended Kalman filter	Planar target	$\mathbf{b}, \mathbf{R}_{ic}, \mathbf{T}_{ic}$	Yes	No	High	Yes
Kelly & Sukhatme [38]	Unscented Kalman filter	Scene points (SFM)	$\mathbf{b}, \mathbf{R}_{ic}, \mathbf{T}_{ic}$	Yes	Yes	High	Yes
Our method [50]	Factorization	Gravity, vertical edges	$\mathbf{S}, \mathbf{b}, \mathbf{K}, \mathbf{R}_{ic}$	No	Yes	Low	No

SELF (self-calibration), COMP (computational load), and CORR (parameter correlation).

significant amount of labor and precision equipment are required. On the other hand, a Kalman filter-based approach has cast calibration as a state estimation problem [37, 38, 39]. A vision-aided inertial navigation system had the augmented states for relative pose and IMU bias. An extended Kalman filter (EKF) [37] or an unscented Kalman filter (UKF) [38] progressively estimated the states from known 3D features. A gray-box system identification technique was also used to estimate the parameters from the nonlinear optimization of prediction errors in EKF [39]. Common drawbacks of these KF-based methods are that a good initial parameter is essential, known feature points are required, and no other intrinsic parameters are included but IMU bias. See Table 4.1 for a more detailed comparison between camera/IMU calibration methods.

4.2 Factorization Method for IMU Self-Calibration

This section provides an overview of the factorization method that is based on the bilinear formulation of IMU measurements. Since no differences exist between accelerometers and gyroscopes as an inertial sensor, motion refers to either force (acceleration) or angular velocity exerted on the IMU. The terms, *motion* and *load*, are identical and used interchangeably.

Affine measurement model:

Suppose that the IMU is an accelerometer (or gyroscope) cluster of n single-axis components and its components are arbitrarily aligned to each other as shown in Fig. 4.1.

A measurement model of each component z is *affine*, $z = af + b$, *i.e.*, linearly proportional

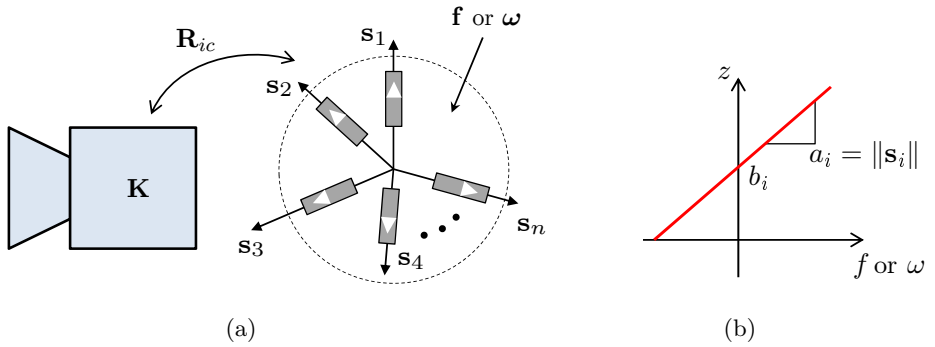


Figure 4.1: Calibration parameters of a camera/IMU system $(\mathbf{K}, \mathbf{S}, \mathbf{b}, \mathbf{R}_{ic})$: (a) A camera calibration matrix $\mathbf{K}$, an accelerometer or gyroscope unit $(\mathbf{S}, \mathbf{b})$ composed of n single-axis components in an arbitrary configuration, and their relative orientation $\mathbf{R}_{ic}$. (b) An affine measurement model $(z = a_i f + b_i)$ of a single-axis inertial sensor component with a scale factor $a_i = \|\mathbf{s}_i\|$ and bias b_i.

to an external motion f and a non-zero for a null motion. In 3D, the measurement z_i of an i-th component in the IMU is expressed as the sum of the projection of a motion $\mathbf{f}$ on the sensitivity axis $\mathbf{s}_i$ and the non-zero bias b_i:

$$z_i = \mathbf{f}^\top \mathbf{s}_i + b_i, \quad i = 1, \ldots, n \tag{4.1}$$

where the scale factor $a_i = \|\mathbf{s}_i\|$.

Bilinear Form:

Once m motions are applied on the n-component IMU, the measurements are collected into a matrix $\mathbf{Z} \in \mathcal{R}^{m \times n}$ as follows:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{f}_1^\top \mathbf{s}_1 + b_1 & \mathbf{f}_1^\top \mathbf{s}_2 + b_2 & \cdots & \mathbf{f}_1^\top \mathbf{s}_n + b_n \\ \mathbf{f}_2^\top \mathbf{s}_1 + b_1 & \mathbf{f}_2^\top \mathbf{s}_2 + b_2 & \cdots & \mathbf{f}_2^\top \mathbf{s}_n + b_n \\ \vdots & \vdots & & \vdots \\ \mathbf{f}_m^\top \mathbf{s}_1 + b_1 & \mathbf{f}_m^\top \mathbf{s}_2 + b_2 & \cdots & \mathbf{f}_m^\top \mathbf{s}_n + b_n \end{bmatrix} \tag{4.2}$$

where z_{ij} is the j-th component's output for the i-th motion $\mathbf{f}_i$ $(i = 1, \ldots, m,\ j = 1, \ldots, n)$.

The measurement matrix $\mathbf{Z}$ can be rewritten as a product of two matrices, $\mathbf{F}_b$ and $\mathbf{S}_b$[1], which have been called an *augmented motion* matrix and a *shape* matrix [45], respectively. Each

[1]The subscript b is intended to clarify that the bias b is explicitly considered in the shape matrix and the motion matrix is augmented with an extra column of ones at the right.

measurement z_{ij} is a product of an augmented motion vector $[\mathbf{f}_i^\top\ 1]$ and a shape parameter $[\mathbf{s}_j^\top\ b_j]$.

$$\mathbf{Z} = \begin{bmatrix} \mathbf{f}_1^\top & 1 \\ \mathbf{f}_2^\top & 1 \\ \vdots & \vdots \\ \mathbf{f}_m^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_n \\ b_1 & b_2 & \cdots & b_n \end{bmatrix} = \begin{bmatrix} \mathbf{F} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{S} \\ \mathbf{b} \end{bmatrix} = \mathbf{F}_b\,\mathbf{S}_b \tag{4.3}$$

If an external calibration device used in a traditional calibration method captures all of the motions $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$ precisely, the calibration parameter could become as simple as a least squares solution, $\mathbf{S}_b = \mathbf{F}_b^+\mathbf{Z} = (\mathbf{F}_b^\top\mathbf{F}_b)^{-1}\mathbf{F}_b^\top\mathbf{Z}$ when $m \geq 4$.

In constrast, our factorization-based calibration aims at estimating $\mathbf{S}_b$ from partially known motions that are readily obtainable without calibration devices.

4.2.1 Matrix reconstruction and load constraints

When $\mathbf{Z}$ is given, constraints are needed to find a true motion and shape matrix since there exist infinitely many factorizations that decompose $\mathbf{Z}$ into a product of two matrices.

The first constraint is a matrix rank condition on $\mathbf{Z}$. Let p denote the dimension of a calibration parameter per component in $\mathbf{S}_b$ ($p = 4$). Since $\mathbf{Z}$ is constructed from $\mathbf{F}_b \in \mathcal{R}^{m\times p}$ and $\mathbf{S}_b \in \mathcal{R}^{p\times n}$, the rank of $\mathbf{Z}$ should be at most p when $\min(m, n) \geq p$ and $\mathbf{Z}$ is noiseless. This has been called the *proper rank constraint* [45]. Singular value decomposition (SVD) [51] can factor $\mathbf{Z}$ into two rank-p matrices, $\widehat{\mathbf{F}_b}$ and $\widehat{\mathbf{S}_b}$, as follows:

$$\mathbf{Z} = \mathbf{U}\,\Sigma\,\mathbf{V}^\top = (\mathbf{U}\boldsymbol{\Sigma}^{1/2})(\boldsymbol{\Sigma}^{1/2}\mathbf{V}^\top) = \widehat{\mathbf{F}_b}\,\widehat{\mathbf{S}_b} \tag{4.4}$$

where $\boldsymbol{\Sigma}$ is a $p \times p$ diagonal, $\mathbf{U}$ and $\mathbf{V}$ are a $m \times p$ and $n \times p$ unitary matrix, respectively. The ambiguity remains because it is possible to insert any invertible matrix $\mathbf{A}_{4\times4}$ inbetween.

$$\mathbf{Z} = \widehat{\mathbf{F}_b}\widehat{\mathbf{S}_b} = (\widehat{\mathbf{F}_b}\,\mathbf{A}_{4\times4})\,(\mathbf{A}_{4\times4}^{-1}\,\widehat{\mathbf{S}_b}) = \mathbf{F}_b\,\mathbf{S}_b \tag{4.5}$$

The second constraint is drawn from partially known calibrating motions $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$. Suppose that each piece of information about the motions is expressed as an inner product:

$$\boxed{\mathbf{f}_i^\top\mathbf{f}_j = c_{ij}} \tag{4.6}$$

which represents the partial knowledge of a pair of motions or a single motion such as a motion magnitude $\|\mathbf{f}_i\|^2 = c_{ii}$ and an orthogonal motion $\mathbf{f}_i^\top\mathbf{f}_j = 0$. We call a set of these motion constraints *load constraint set* $\mathcal{C}$. A procedure on how to find $\mathbf{A}_{4\times4}$ from the constraint set $\mathcal{C}$ is a key step in the factorization and will be explained in detail in Section 4.3.

Finally, once $\mathbf{A}_{4\times4}$ is found from $\mathcal{C}$, the calibration step is completed with the simultaneous recovery of the shape parameters in $\mathbf{S}_b$ and the applied motion in $\mathbf{F}_b$ as follows.

$$\mathbf{S}_b = \mathbf{A}_{4\times4}^{-1}\,\widehat{\mathbf{S}_b}, \quad \mathbf{F}_b = \widehat{\mathbf{F}_b}\,\mathbf{A}_{4\times4} \tag{4.7}$$

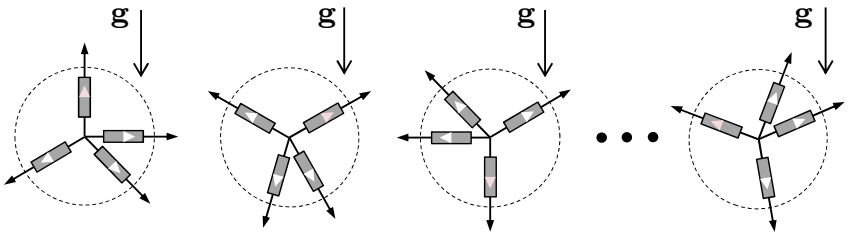


Figure 4.2: Gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$ for the accelerometer self-calibration: an accelerometer unit is oriented at different attitudes with respect to gravity in a static condition.

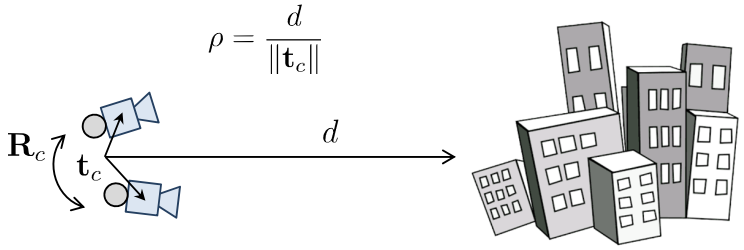


Figure 4.3: Angular speed constraint $\mathcal{C}^*$ for the gyroscope self-calibration: a camera affixed to a gyroscope unit is randomly rotated in front of distant scene objects. The angular speed $\|\boldsymbol{\omega}\|$ is obtained from feature correspondences between two consecutive images and a known time interval ΔT.

4.2.2 Natural reference features for self-calibration

One prevalent constraint type used for self-calibration is magnitude. For example, an accelerometer is exposed to an identical load (gravity) whose magnitude is $\|\mathbf{f}_i\| = g$ for $i = 1, \ldots, m$, as shown in Figure 4.2. The magnitude of an angular velocity $\|\boldsymbol{\omega}\|$ exerted on a gyroscope is readily available from a speed-controlled turntable or a rotating camera tracking distant scene objects as shown in Figure 4.3. The detailed procedure to obtain $\|\boldsymbol{\omega}\|$ from an uncalibrated camera will be described in Section 4.6.2. We define the intra-relation constraint set $\mathcal{C}^*$ as follows.

Definition 1. *$\mathcal{C}^*$ is a intra-relation load constraint set which is composed only of load magnitudes,* i.e., *$\mathcal{C}^* = \{c_{ii} = \mathbf{f}_i^\top \mathbf{f}_i \,|\, i = 1, \ldots, m,\ m \geq 9\}$. Also, $\mathcal{C}^*_\tau$ refers to when all the magnitudes in $\mathcal{C}^*$ are the same as τ,* i.e., *$c_{ii} = \tau$ for all i.*

From Definition 1, the gravity magnitude constraint can be simply denoted by $\mathcal{C}^*_{\tau=1}$ (a known constant magnitude is scaled to 1 without loss of generality).

4.2.3 Redundant vs. triad IMU

The proper rank constraint demands that the component size (n) should be no less than the shape parameter dimension ($p = 4$), *i.e.*, $n \geq p$. This constraint is satisfied by a redundant IMU

($n \geq 4$) but violated by a triad IMU ($n = 3$). The SVD in (4.4) fails to produce a $\widehat{\mathbf{S}_b} \in \mathcal{R}^{p \times n}$ when $n = 3$.

One possible workaround for a triad IMU is to reduce the parameter dimension to $p = 3$ by excluding the bias $\mathbf{b}$ from $\mathbf{S}_b$ and iteratively estimate the bias from factorization error. An iterative factorization solution for a triad IMU will be presented in Section 4.4.

4.3 Redundant IMU Case

The calibration problem is now equivalent to finding a non-singular matrix $\mathbf{A}_{4\times4}$ in (4.5). For a redundant IMU, a linear solution will be derived from a load constraint set $\mathcal{C}$. Note that the linear solution that follows in this section is valid for any combinations of load constraints, not confined to $\mathcal{C}^*$. Instead interesting properties of solution spaces for $\mathcal{C}^*$ and $\mathcal{C}^*_\tau$ will be discussed in Appendix B.

4.3.1 Shape recovery procedure from load constraints

Given a load constraint set $\mathcal{C}$ in (4.6), a set of linear constraints for $\mathbf{A}_{4\times4}$ in (4.5) are constructed. Let $\mathbf{A}_{4\times4}$ be partitioned into the following block matrices.

$$\mathbf{A}_{4\times4} = \left[\begin{array}{c|c} \mathbf{A}_{4\times3} & \mathbf{a}_4 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{A}_{3\times3} & \mathbf{a}_4 \\ \mathbf{u}^\top & \end{array}\right] \tag{4.8}$$

Firstly, the last column $\mathbf{a}_4$ is immediately recoverable without using $\mathcal{C}$. Since the last column of $\mathbf{F}_b = [\mathbf{F}\ \mathbf{1}]$ is constant, the pseudo-inverse of $\widehat{\mathbf{F}_b}$ finds a least squares solution for $\mathbf{a}_4$:

$$\mathbf{a}_4 = \widehat{\mathbf{F}_b}^{+}\,\mathbf{1} \tag{4.9}$$

Secondly, the remaining columns $\mathbf{A}_{4\times3}$ are associated with $\mathcal{C}$. From (4.7), each true load $\mathbf{f}_i \in \mathcal{R}^{3\times1}$ is rewritten as

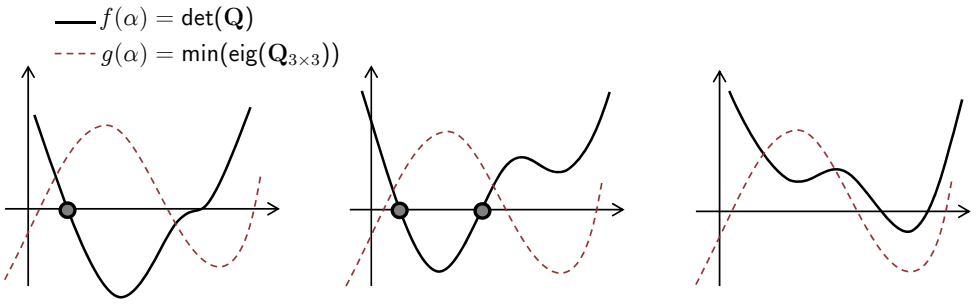
$$\mathbf{f}_i = \mathbf{A}_{4\times3}^\top \widehat{\mathbf{f}}_{b,i} \quad \text{for } i = 1, \ldots, m \tag{4.10}$$

where $\widehat{\mathbf{f}}_{b,i} \in \mathcal{R}^{4\times1}$ is at the i-th row of $\widehat{\mathbf{F}}_b$.

Suppose that k load constraints exist in $\mathcal{C}$. Plugging (4.10) into each load constraint c_{ij} yields one quadratic equation for $\mathbf{A}_{4\times3}$. The Kronecker product rewrites this as a linear equation in terms of $\mathbf{q} \in \mathcal{R}^{10\times1}$, which is a vector of lower triangular elements of a symmetric matrix $\mathbf{Q} = \mathbf{A}_{4\times3}\mathbf{A}_{4\times3}^\top$:

$$c_{ij} = \mathbf{f}_i^\top \mathbf{f}_j = \widehat{\mathbf{f}}_{b,i}^\top (\mathbf{A}_{4\times3}\mathbf{A}_{4\times3}^\top)\, \widehat{\mathbf{f}}_{b,j} = \widehat{\mathbf{f}}_{b,i}^\top \mathbf{Q}\, \widehat{\mathbf{f}}_{b,j} \tag{4.11}$$

$$= [\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^\top \mathtt{Vec}(\mathbf{Q}) = [\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^\top \mathbf{T}_{16\times10}\, \mathbf{q} \tag{4.12}$$



(a) case 1: a single solution (b) case 2: multiple solutions (c) case 3: no solution

Figure 4.4: Three cases on the number of solutions for $\mathbf{q}$ when $\mathrm{rank}(\mathbf{L}) = 9$: Among four roots of $f(\alpha)$, a true α should be real and make $\mathbf{Q}_{3\times3}$ a positive definite. When no noise in $\mathbf{Z}$, a single solution exists (case 1). Multiple or no solutions for $\mathbf{q}$ may exist for noisy $\mathbf{Z}$ (case 2 and 3).

where $\mathbf{T}_{16\times10}$ is a constant matrix that converts a vectorized $\mathbf{Q}$ into a minimal parameter $\mathbf{q}$. Stacking all the constraints in $\mathcal{C}$ produces a linear system equation $\mathbf{L} \in \mathcal{R}^{k\times10}$ for $\mathbf{q}$:

$$\mathbf{c} = \mathbf{L}\,\mathbf{q} \tag{4.13}$$

where $\mathbf{L} = \mathcal{S}[\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^\top \mathbf{T}_{16\times10}$, $\mathbf{c}$ is a vector of $\{c_{ij}\}$ in $\mathcal{C}$, and $\mathcal{S}$ is a row stacking operator that uses all of the constraints in $\mathcal{C}$.

It is noteworthy that $\mathbf{Q} \in \mathcal{R}^{4\times4}$ is not full-rank but a rank-3 matrix because $\mathrm{rank}(\mathbf{Q}) = \mathrm{rank}(\mathbf{A}_{4\times3}) = 3$. Rewriting $\mathbf{Q}$ with the block matrices in (4.8)

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A}_{3\times3}^\top \mathbf{A}_{3\times3} & \mathbf{A}_{3\times3}^\top \mathbf{u} \\ \mathbf{u}^\top \mathbf{A}_{3\times3} & \mathbf{u}^\top \mathbf{u} \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{Q}_{3\times3} & \mathbf{q}_{13} \\ \mathbf{q}_{13}^\top & q_{44} \end{bmatrix} \tag{4.14}$$

reveals that $\mathbf{Q}$ (equivalently $\mathbf{q}$) is internally subject to the following two constraints, which we call a *q-constraint*:

$$\mathtt{det}(\mathbf{Q}) = 0 \tag{4.15a}$$

$$\min\{\mathtt{eig}(\mathbf{Q}_{3\times3})\} > 0 \tag{4.15b}$$

where the first equality constraint is due to a non-invertible rank-3 $\mathbf{Q}$, *i.e.*, $\mathbf{q}_{13}^\top\, \mathbf{Q}_{3\times3}^{-1}\, \mathbf{q}_{13} - q_{44} = 0$, and the second inequality constraint indicates that $\mathbf{Q}_{3\times3} = \mathbf{A}_{3\times3}^\top \mathbf{A}_{3\times3}$ is a positive definite by construction and its eigenvalues should be all positive.

When $\mathbf{L}$ in (4.13) is full-rank with a sufficient number of load constraints ($\mathrm{rank}(\mathbf{L}) = 10$ when $k \geq 10$), a least squares solution $\mathbf{q}$ is obtainable from the pseudo-inverse of $\mathbf{L}$. When no noise is present in $\mathbf{Z}$, the following particular solution $\mathbf{q}_p$ inherently satisfies the q-constraint as well.

$$\mathbf{q} = \mathbf{q}_p \quad \text{s.t.} \quad \mathbf{q}_p = \mathbf{L}^+ \mathbf{c} \tag{4.16}$$

Table 4.2: List of solution schemes for the redundant IMU calibration according to rank($\mathbf{L}$)

rank($\mathbf{L}$)	Solution for $\mathbf{q}$	Examples
< 9	$\mathbf{q}$ is indeterminate	Lack of load constraints: $\lvert\mathcal{C}\rvert < \lvert\mathcal{C}\rvert_{min} = 9$. Degenerate load configuration $\mathbf{F} \in \mathcal{D}_{\mathbf{F}}$ in Figure B.2.
9	$\mathbf{q} = \mathbf{L}^{+}\mathbf{c} + \alpha\mathbf{q}_n$, $\mathbf{q}_n \in \mathcal{N}(\mathbf{L})$ determine α from q-constraint (4.15)	Minimal constraints: $\lvert\mathcal{C}\rvert = 9$ Gravity magnitudes: $\mathcal{C} = \mathcal{C}^{*}_{\tau=1}$ s.t. $\lvert\mathcal{C}^{*}\rvert \geq 9$
10	$\mathbf{q} = \mathbf{L}^{+}\mathbf{c}$ enforce rank($\mathbf{Q}$) = 3 by SVD	Non-minimal mixed constraint: $\mathcal{C} - \mathcal{C}^{*} \neq \emptyset$ Different magnitudes: $\mathcal{C} = (\mathcal{C}^{*} - \mathcal{C}^{*}_{\tau=1})$

On the other hand, even when $\mathbf{L}$ is *not* full-rank, $\mathbf{q}$ is still solvable if rank($\mathbf{L}$) = 9. This is because $\mathbf{q} \in \mathcal{R}^{10\times 1}$ actually has 9 degrees of freedom when the rank q-constraint (4.15a) is considered. The solution for $\mathbf{q}$ spans a one-dimensional null space $\mathcal{N}$ of $\mathbf{L}$ with an unknown scalar α:

$$\mathbf{q} = \mathbf{q}_p + \alpha\mathbf{q}_n \quad \text{s.t.} \quad \mathbf{q}_p = \mathbf{L}^{+},\, \mathbf{q}_n \in \mathcal{N}(\mathbf{L}). \tag{4.17}$$

The rank and positivity condition in the q-constraint (4.15) are used to determine α. Plugging (4.17) into the rank condition (4.15a) yields a fourth-order polynomial $f(\alpha)$ whose coefficients[2] are parameterized by $\mathbf{q}_p$ and $\mathbf{q}_n$. Among up to four possible roots, a true α should be a real-valued one that makes $\mathbf{Q}_{3\times 3}$ a positive definite (4.15b). Fig. 4.4 shows three possible cases for the number of $\mathbf{q}$ solutions depending on the location of α. Only one α meets the q-constraint when no noise in $\mathbf{Z}$. Table 4.2 summarizes the solution schemes to obtain $\mathbf{q}$ according to rank($\mathbf{L}$).

Once $\mathbf{q}$ is obtained, $\mathbf{A}_{4\times 3} = [\mathbf{A}_{3\times 3}^{\top} \;\; \mathbf{u}]^{\top}$ in (4.8) is found as follows via the Cholesky decomposition of a symmetric positive-definite $\mathbf{Q}_{3\times 3}$ in (4.14):

$$\mathbf{A}_{3\times 3} = \texttt{chol}(\mathbf{Q}_{3\times 3}), \quad \mathbf{u} = \mathbf{A}_{3\times 3}^{-\top}\mathbf{q}_{13} \tag{4.18}$$

Finally, the reconstruction of true $\mathbf{S}_b$ and $\mathbf{F}_b$ is completed as (4.7) by the $\mathbf{A}_{4\times 4}$ found from the load constraint set $\mathcal{C}$. Note that the final reconstruction is fundamentally ambiguous up to a three-dimensional rotation $\mathbf{R}$ since $\mathbf{Z} = \mathbf{F}_b\mathbf{S}_b = \mathbf{F}\mathbf{S} + \mathbf{b} = (\mathbf{F}\mathbf{R})(\mathbf{R}^{\top}\mathbf{S}) + \mathbf{b}$. This means that the sensor's internal coordinate system can be chosen freely and is already reflected in the solution during the Cholesky decomposition of $\mathbf{Q}_{3\times 3}$, which determines $\mathbf{A}_{3\times 3}$ up to a rotation matrix $\mathbf{R}$.

4.3.2 Linear solution space

Table 4.2 summarizes the solution schemes according to rank($\mathbf{L}$) and typical examples for each scheme. One key example is that $\mathbf{L}$ is always rank deficient when the gravity magnitude is

[2] A symbolic computation of $\texttt{det}(\mathbf{Q}(\mathbf{q}_p + \alpha\mathbf{q}_n)) = 0$ produces the coefficients of a fourth-order polynomial $f(\alpha)$.

Algorithm 4.1: Factorization-based linear solution for redundant IMU calibration

$\mathbf{S}_b = (\mathbf{S}, \mathbf{b}) = \mathsf{Redundant\_IMU\_Calibration}(\mathbf{Z}, \mathcal{C})$

begin

- $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathtt{svd}(\mathbf{Z})$
- $\widehat{\mathbf{F}_b} = \mathbf{U}_p \mathbf{\Sigma}_p^{1/2}, \ \widehat{\mathbf{S}_b} = \mathbf{\Sigma}_p^{1/2} \mathbf{V}_p^\top, \ p = 4$
- **foreach** c_{ij} *in* $\mathcal{C}$ **do**
  - $\mathbf{L}_k = [\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^\top \mathbf{T}_{16\times 10}$ in (4.12)
- **if** $\mathcal{C} = \mathcal{C}^*_{\tau=1}$ *or* $|\mathcal{C}| = 9$ **then**
  - $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathtt{svd}(\mathbf{L})$
  - $\mathbf{q}_p = \mathbf{L}^+ \mathbf{c}, \quad \mathbf{q}_n = \mathtt{last\_col}(\mathbf{V})$
  - $\alpha = \mathtt{q\_constraint}(\mathbf{q}_p, \mathbf{q}_n)$ in (4.15)
  - $\mathbf{Q} = \mathtt{convert}(\mathbf{q}_p + \alpha \mathbf{q}_n)$
- **else**
  - $\mathbf{q} = \mathbf{L}^+ \mathbf{c}, \quad \mathbf{Q} = \mathtt{convert}(\mathbf{q})$
  - $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathtt{svd}(\mathbf{Q})$
  - $\mathbf{Q} = \mathbf{U}_3 \mathbf{S}_3 \mathbf{V}_3$
- $\mathbf{A}_{3\times 3} = \mathtt{chol}(\mathbf{Q}_{3\times 3}), \ \mathbf{A}_{4\times 4} = \left[\begin{array}{c|c} \mathbf{A}_{3\times 3} & \widehat{\mathbf{F}_b}^+ \mathbf{1} \\ \mathbf{q}_{13}^\top \mathbf{A}_{3\times 3}^{-1} & \end{array}\right]$
- $\mathbf{S}_b = \mathbf{A}_{4\times 4}^{-1} \widehat{\mathbf{S}_b}, \ \mathbf{F}_b = \widehat{\mathbf{F}_b} \mathbf{A}_{4\times 4}$

solely used in the accelerometer calibration, *i.e.*, $\mathrm{rank}(\mathbf{L}) \leq 9$ when $\mathcal{C} = \mathcal{C}^*_{\tau=1}$. For the gravity magnitude, Propositions 1 and 2 in Section B.4 claim that $f(\alpha)$ has one real root for a true α and that the others are triple roots that violate the positivity q-constraint (4.15b).

More analysis on the solution space of $\mathbf{q}$ is described in Section B, including a minimum number of constraints $|\mathcal{C}|_{min}$ and degenerate load condition $\mathcal{D}_{\mathbf{F}}$.

4.4 Triad IMU Case

The factorization method allows the shape parameter dimension (p) up to a component size (n). For a triad IMU ($n = 3$), the exclusion of the bias from $\mathbf{S}_b$ reduces the parameter dimension to three in $\mathbf{S}$. Then, the bias is iteratively estimated at the outside of the factorization loop using motion reconstruction errors compared with a load constraint $\mathcal{C}$. Note that the following iterative algorithm is specific to $\mathcal{C}^*$.

4.4.1 Iterative factorization algorithm

Let $\mathbf{D}$ represent bias-compensated measurements in which the bias is subtracted from $\mathbf{Z}$, *i.e.*, $d_{ij} = z_{ij} - b_j$ for all i, j in (4.2). Since $\mathbf{D} = \mathbf{FS}$, a bilinear formulation of $\mathbf{D}$ is a product of two matrices, $\mathbf{F}$ and $\mathbf{S}$, and $\text{rank}(\mathbf{D}) = 3$. The constraints used for the factorization are identical to those for a redundant IMU, except that we use $\mathbf{D}$ instead of $\mathbf{Z}$.

Suppose that an initial bias $\mathbf{b}^0$ is given. At the k-th iteration, the proper rank constraint ($p = 3$) on $\mathbf{D}^k$ yields

$$\mathbf{D}^k = \mathbf{Z} - \mathbf{1}_{m\times 1}\mathbf{b}^k \tag{4.19}$$

$$= (\widehat{\mathbf{F}}\mathbf{A}_{3\times 3})(\mathbf{A}_{3\times 3}^{-1}\,\widehat{\mathbf{S}}) \tag{4.20}$$

from $\texttt{svd}(\mathbf{D}^k) = \mathbf{U\Sigma V}^\top$, $\widehat{\mathbf{F}} = \mathbf{U\Sigma}^{1/2}$ and $\widehat{\mathbf{S}} = \mathbf{\Sigma}^{1/2}\mathbf{V}^\top$. True shape and motion matrices are now indeterminate up to an invertible matrix $\mathbf{A}_{3\times 3}$. From $\mathbf{f}_i = \mathbf{A}_{3\times 3}^\top \widehat{\mathbf{f}}_i$, each load constraint in $\mathcal{C}^*$ produces one linear equation for $\mathbf{q}'$:

$$c_{ii} = \widehat{\mathbf{f}}_i^\top(\mathbf{A}_{3\times 3}\mathbf{A}_{3\times 3}^\top)\widehat{\mathbf{f}}_i = \widehat{\mathbf{f}}_i^\top \mathbf{Q}_{3\times 3}\,\widehat{\mathbf{f}}_i \tag{4.21}$$

$$= [\widehat{\mathbf{f}}_i \otimes \widehat{\mathbf{f}}_j]^\top \mathbf{T}_{9\times 6}\,\mathbf{q}' \tag{4.22}$$

where $\mathbf{q}' \in \mathcal{R}^{6\times 1}$ is a vector of lower triangular elements of a symmetric matrix $\mathbf{Q}_{3\times 3}$ and $\mathbf{T}_{9\times 6}$ is a constant conversion matrix from a vectorized $\mathbf{Q}_{3\times 3}$ to $\mathbf{q}'$. If $|\mathcal{C}^*| \geq 6$, $\mathbf{q}'$ is solvable from a linear system equation:

$$\mathbf{L}'\mathbf{q}' = \mathbf{c} \tag{4.23}$$

where $\mathbf{L}' = \mathcal{S}[\widehat{\mathbf{f}}_i \otimes \widehat{\mathbf{f}}_i]^\top \mathbf{T}_{9\times 6}$. Note that $\mathbf{q}'$ is solvable only if $\mathbf{L}$ has a full rank because $\mathbf{q}'$ has no internal constraints like (4.15a). The factorization is completed via the Cholesky decomposition of $\mathbf{Q}_{3\times 3}$ into $\mathbf{A}_{3\times 3}$. The best reconstruction of $\mathbf{D}^k$ is given as $\mathbf{F}^k = \widehat{\mathbf{F}}\mathbf{A}_{3\times 3}$ and $\mathbf{S}^k = \mathbf{A}_{3\times 3}^{-1}\,\widehat{\mathbf{S}}$.

Since $\mathbf{D}^k$ is incompletely compensated due to the bias error in $\mathbf{b}^k$, a new bias $\mathbf{b}^{k+1}$ for the next iteration is given as

$$\overline{\mathbf{f}_i} = \frac{\sqrt{c_{ii}}}{\|\mathbf{f}_i^k\|}\,\mathbf{f}_i^k, \quad i = 1, \ldots, m \tag{4.24}$$

$$\overline{\mathbf{S}_b} = \left[\,\overline{\mathbf{F}}\ \mathbf{1}\,\right]^+ \mathbf{Z} \tag{4.25}$$

$$\mathbf{b}^{k+1} = \texttt{last-row}(\overline{\mathbf{S}_b}). \tag{4.26}$$

and the iteration continues until $\|\mathbf{b}^{k+1} - \mathbf{b}^k\| < \varepsilon$ for a small threshold ε.

Since $\mathbf{D}^k$ is contaminated by the bias error in $\mathbf{b}^k$, the reconstructed motion and shape matrices, $\mathbf{F}^k$ and $\mathbf{S}^k$, are also incorrect. The least squares solution $\mathbf{q}'$ in (4.23) has a residual that leaves $\mathbf{F}^k$ not fully constrained by $\mathcal{C}^*$. Hence, each load constraint in $\mathcal{C}^*$ is explicitly re-enforced on

Algorithm 4.2: Iterative factorization-based calibration method for triad IMU

$\mathbf{S}_b = (\mathbf{S}, \mathbf{b}) = \mathsf{Triad\_IMU\_Calibration}(\mathbf{Z},\ \mathbf{b}^0,\ \mathcal{C}^*)$

$k = 0$

repeat

$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathtt{svd}(\mathbf{Z} - \mathbf{b}^k)$

$\widehat{\mathbf{F}} = \mathbf{U}_p \boldsymbol{\Sigma}_p^{1/2},\ \ \widehat{\mathbf{S}} = \boldsymbol{\Sigma}_p^{1/2} \mathbf{V}_p^\top,\ \ p = 3$

foreach c_{ii} *in* $\mathcal{C}$ **do**

$\mathbf{L}'_i = [\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,i}]^\top \mathbf{T}_{9\times 6}$

$\mathbf{q}' = \mathbf{L}'^{+}\mathbf{c},\ \ \mathbf{Q}_{3\times 3} = \mathtt{convert}(\mathbf{q}')$

$\mathbf{A}_{3\times 3} = \mathtt{chol}(\mathbf{Q}_{3\times 3}) \longrightarrow\ \ \mathbf{S} = \mathbf{A}_{3\times 3}^{-1}\, \widehat{\mathbf{S}},\ \ \mathbf{F} = \widehat{\mathbf{F}}\, \mathbf{A}_{3\times 3}$

foreach $\mathbf{f}_i$ *in* $\mathbf{F}$ **do**

$\overline{\mathbf{f}_i} = (\sqrt{c_{ii}}/\|\mathbf{f}_i\|)\, \mathbf{f}_i$

$\mathbf{S}_b = \left[\, \overline{\mathbf{F}}\ \mathbf{1} \,\right]^{+} \mathbf{Z},\ \mathbf{b}^k = \mathtt{last\text{-}row}(\mathbf{S}_b)$

$k \leftarrow k + 1$

until $\|\mathbf{b}^{k+1} - \mathbf{b}^k\| < \varepsilon$

the reconstructed motion $\mathbf{F}^k$ in order to find an improved bias. In other words, the magnitude of each motion $\mathbf{f}_i^k$ is normalized to $\|\mathbf{f}_i^k\|^2 = c_{ii}$. Then, a new bias $\mathbf{b}^{k+1}$ for the next iteration is computed from the updated $\mathbf{F}^k$ and the measurement $\mathbf{Z}$.

4.4.2 Bias convergence region

The advantages of this iterative factorization over the nonlinear methods [42, 43, 44] are that it is based on a linear method and it requires an initial bias only; there is no need to presume initial values for scale factors and alignments.

A convergence of the bias is demonstrated to be insusceptible to initial bias error. The simulation and experiments in Fig. 4.8 and 4.16 show that the convergence region is wide enough to cover $\|\mathbf{b}^0 - \mathbf{b}^*\| < \|\mathbf{s}\|$ (when $\|\mathbf{f}\| = 1$) for a true bias $\mathbf{b}^*$. When $\mathbf{Z}$ is generated by $\mathcal{C}^*_{\tau=1}$, one automatic setting for each initial bias would be $\mathbf{b}^0 = \text{mean}(\mathbf{Z})$ because no measurements are greater than a scale factor from the bias.

4.5 Simulation Evaluation

The performance of factorization-based IMU calibration is evaluated mainly for an accelerometer unit under the gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$ in Figure 4.2. In the following numerical simulations, a true shape $\mathbf{S}_b$ has a unit scale factor and unit bias, $\|\mathbf{s}_i\| = b_i = 1$ for $i = 1, \ldots, n$, and consists of multiple tri-axial units when each unit is skewed to another.

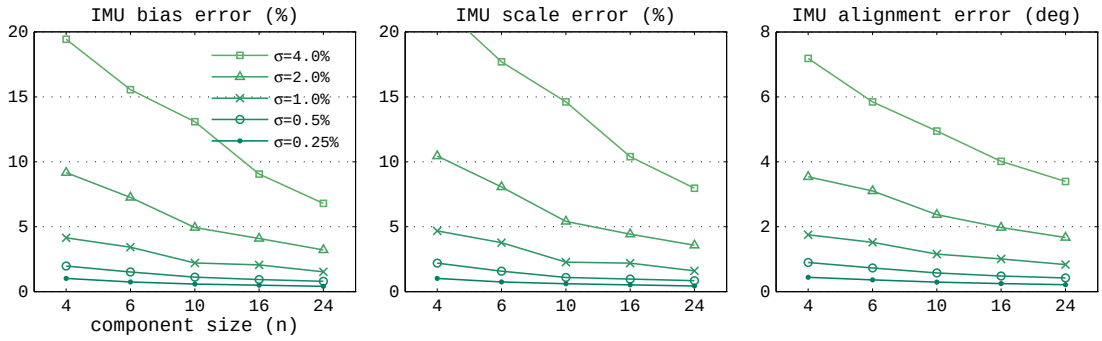


Figure 4.5: IMU calibration performance with respect to component size (n) and noise (σ) when the gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$ is used.

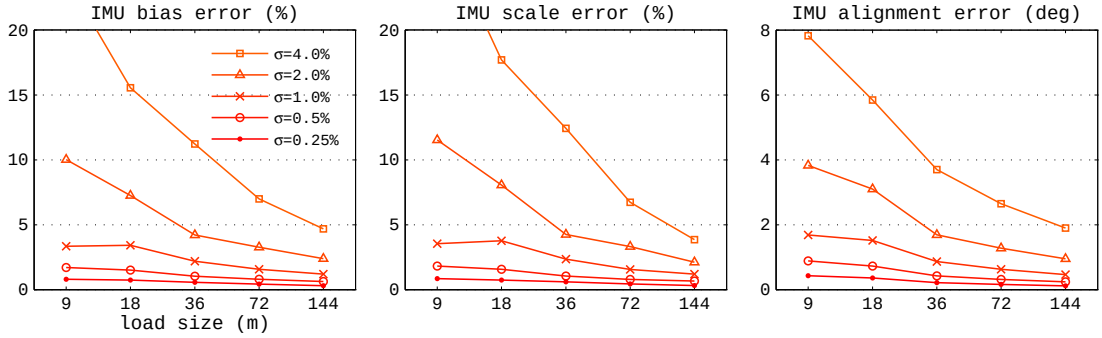


Figure 4.6: IMU calibration performance with respect to load size (m) and noise (σ) when the gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$ is used.

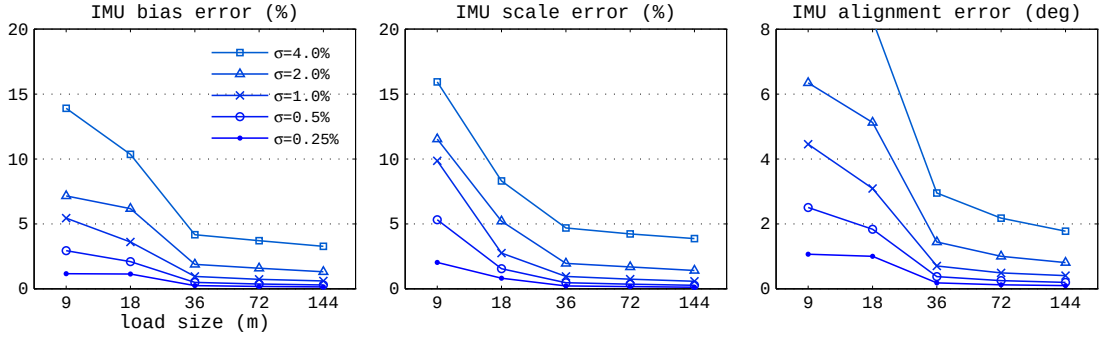


Figure 4.7: IMU calibration performance of mixed constraint types when the gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$ (50%) and the orthogonal motion constraint (50%) are used together.

Performance of gravity magnitude constraint:

The performance of accelerometer calibration in $|\mathcal{C}^*_{\tau=1}|$ was assessed through simulation with respect to the following three factors: Gaussian measurement noise (σ), the number of loads (load

Table 4.3: Simulation evaluation of calibration performance with respect to noise (σ) and component size (n) when $\mathcal{C}^*_{\tau=1}$ is used.

	bias ($\mathbf{b}_{err}$, %)			scale ($\|\mathbf{s}\|_{err}$, %)			alignment ($\angle \mathbf{s}_1 \mathbf{s}_{i,err}$, deg)		
σ (%)	$n=4$	$n=6$	$n=20$	$n=4$	$n=6$	$n=20$	$n=4$	$n=6$	$n=20$
0.25	-0.01 ± 0.82	-0.01 ± 0.56	0.00 ± 0.31	-0.02 ± 0.84	-0.01 ± 0.59	0.00 ± 0.32	0.00 ± 0.42	0.00 ± 0.22	0.00 ± 0.12
0.5	0.02 ± 1.67	0.01 ± 1.11	0.01 ± 0.60	0.09 ± 1.72	0.04 ± 1.16	0.02 ± 0.63	0.00 ± 0.83	0.00 ± 0.44	0.00 ± 0.23
1.0	0.23 ± 3.53	-0.01 ± 2.17	-0.03 ± 1.31	0.66 ± 3.54	-0.04 ± 2.27	-0.07 ± 1.37	0.01 ± 1.75	0.01 ± 0.86	0.00 ± 0.48
4.0	2.24 ± 27.0	-0.03 ± 10.9	-0.58 ± 5.30	7.43 ± 27.0	0.27 ± 11.4	-1.55 ± 5.33	0.03 ± 8.04	-0.01 ± 3.58	-0.04 ± 1.95
8.0	1.58 ± 45.6	-0.10 ± 23.3	-2.27 ± 13.0	11.1 ± 46.3	1.04 ± 24.2	-5.88 ± 12.1	-0.20 ± 14.8	-0.10 ± 7.21	-0.18 ± 4.36

The load size is fixed to 36 ($m = 36$). See Figure 4.5 for the plot.

Table 4.4: Failure rate of the factorization for a redundant IMU when $\mathcal{C}^*_{\tau=1}$ is used (%).

noise (%)	$n=4$	$n=6$	$n=10$	$n=20$
0.5	0.0	0.0	0.0	0.0
1.0	0.0	0.0	0.0	0.0
2.0	0.2	0.0	0.0	0.0
4.0	9.6	6.7	1.4	0.0
8.0	31.3	30.4	18.4	6.2

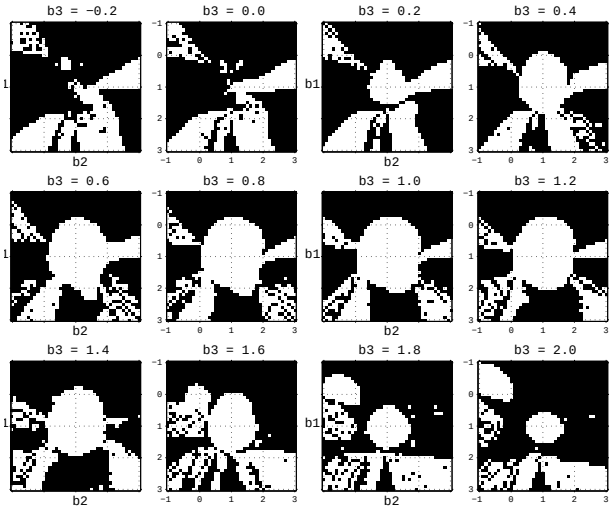


Figure 4.8: Bias convergence region of the iterative factorization for a triad-IMU. Each plot is a cross section along b_3. Initial conditions, $\mathbf{b}^0 = [b_1\ b_2\ b_3]$, in the white region converge to the true $\mathbf{b}^* = [1\ 1\ 1]$ but those in a non-white region are trapped in local minima.

size $m = |\mathcal{C}^*_{\tau=1}|$), and the number of sensor components (component size n). In order to reflect a real scenario in in-field calibration, the IMU's attitude (roll and pitch) was random but bounded within ± 60 degrees. Gaussian noise σ was given as a percentage of the scale factor $\|\mathbf{s}_i\|$.

For every triplet (σ, m, n), a Monte-Carlo simulation was performed with 1000 trials. From Figure 4.5 to 4.7, the bias and scale factor error were evaluated for every sensor component and the alignment error is about vector angles of all possible pairs of sensor components. Figure 4.5 and 4.6 show that the calibration error decreases linearly as the component size (n) increases from 4 to 24, the load size (m) increases from 9 to 144, or the measurement noise (σ) decreases from 8 to 0.25%. Table 4.3 shows the errors in more details and indicates that the factorization method is an unbiased estimator unless the measurement noise is too severe ($\sigma \geq 4.0\%$).

Figure 4.7 shows the case where angular motion constraints (inter-relation constraint) and $\mathcal{C}^*_{\tau=1}$ (intra-relation constraint) are used together. A half of the constraints is given as $\mathbf{f}_i^\top \mathbf{f}_j = 0$ from multiple pairs of orthogonal loads [50]. Compared with Figure 4.6 that uses $\mathcal{C}^*_{\tau=1}$ only, the bias and scale factor error decrease almost by half as the load size increases while the alignment error is found to be greater for a small m. This fact indicates that additional knowledge about angular relation makes the bias and scale factor more constrained when m is large.

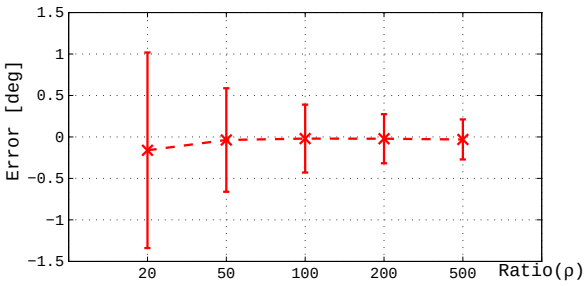
Since a typical MEMS-based accelerometer [52] has noise less than $\sigma = 0.25\%$, the calibration result for an off-the-shelf IMU is expected to be better than 0.01% accuracy and 0.5% precision when $m = 30$ and $n = 6$. The minimal load and component size ($m = 9$, $n = 4$) has 0.02% accuracy and 0.8% precision when $\sigma = 0.25\%$.

The calibration failure occurs when no solution for $\mathbf{q}$ produces a positive definite $\mathbf{Q}_{3\times 3}$ in (4.15), which is usually induced by high measurement noise. Table 4.4 shows the failure rate for 1000 trials in the simulation when $\mathcal{C}^*_{\tau=1}$ is used. The failure begins to appear when the noise is larger than $\sigma = 2.0\%$ and is significantly reduced as the load and component size increase. Practically, this failure would be of little concern because a real accelerometer noise ($\sigma = 0.2\%$) corresponds to a zero failure rate.

Bias convergence region for a triad IMU:

The iterative factorization for the triad IMU in Section 4.4 requires an initial bias $\mathbf{b}^0$. In order to numerically evaluate its convergence region, we tested for every initial point between $\mathbf{b}^0_{min} = -[1\ 1\ 1]$ to $\mathbf{b}^0_{max} = [3\ 3\ 3]$ in 0.05 resolution when a true shape is given as $\mathbf{b}^* = [1\ 1\ 1]$ and $\mathbf{S}^* = \mathbf{I}_{3\times 3}$.

Figure 4.8 shows the bias convergence region $\mathcal{B}$ when $m = 10$, $\sigma = 0.5\%$ and $\mathcal{C} = \mathcal{C}^*_{\tau=1}$. Initial biases in the white region converge to the true bias $\mathbf{b}^*$ but those in the black region are trapped in local minima. Although white regions are divided and scattered, there exists a continuous convergence region that expands spherically around the true bias. Since the radius of this central



ρ	mean(θ_{err})	std(θ_{err})
10	-0.437	2.342
20	-0.161	1.179
50	-0.037	0.624
100	-0.020	0.409
200	-0.021	0.296
500	-0.030	0.342

Figure 4.9: Systematic error in camera rotation angle θ_c = phase(eig($\mathbf{H}$)). The homography $\mathbf{H}$ is estimated by image feature correspondence when the true θ_c is 10 degrees.

volume is at least one, we can conclude that initial biases in $\|\mathbf{b}^0 - \mathbf{b}^*\| < 1$ are all convergent to a global minimum. In other words, the initial error allowed for the bias can be as large as 100% of the scale factor.

Homography angle for gyroscope calibration:

A systematic error exists in $\theta_c(\mathbf{H})$ in cases of non-zero camera translation ($\mathbf{t}_c \neq \mathbf{0}$) and finite scene points ($d \neq \infty$) Figure 4.11. Let $\rho = d/\|\mathbf{t}_c\|$ denote a distance ratio between the distance d to scene points and camera translation amount $\|\mathbf{t}_c\|$. A rotating camera assumption on θ_c is valid only if $\rho = \infty$. For a finite ρ, a systematic error of $\theta_c(\mathbf{H})$ is evaluated by the simulation in Figure 4.9. The error is biased when $\rho < 100$ and its variance decreases logarithmically as ρ increases. When $\rho \geq 200$, the mean squared error of θ_c becomes less than 1%.

4.6 Camera/IMU Calibration for Sensor Fusion

The factorization framework is extended to the calibration of a camera/IMU system. Gravity and natural landmarks such as vertical edges and distant feature points constitute constraints for a collaborative camera/IMU calibration.

4.6.1 Camera vs. accelerometer

Cascaded linear method:

Suppose m images and IMU measurements $\mathbf{Z}$ are collected together in a static condition as shown in Figure 4.10. Once the gravity magnitude $\mathcal{C}^*_{\tau=1}$ calibrates the IMU, gravity directions $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$ are known to the camera. Because a vertical vanishing point $\mathbf{v}_i$ is a projection of the

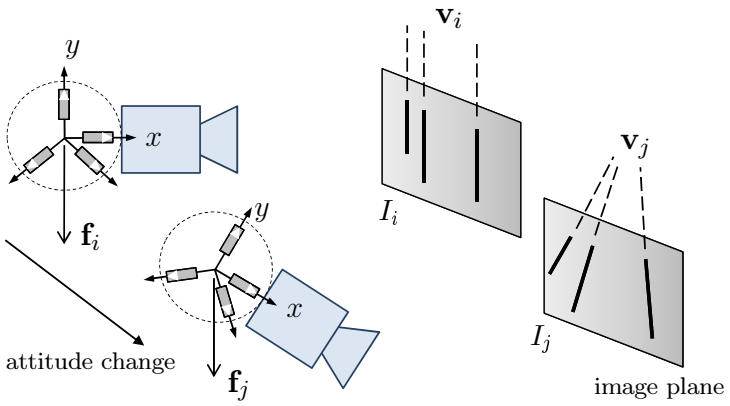


Figure 4.10: Camera/accelerometer calibration: vertical vanishing point $\mathbf{v}_i^v$, which is a common intersection of vertical edges, is a projection of the gravity direction $\mathbf{f}_i$ described in the camera coordinate.

gravity direction $\mathbf{f}_i$, there exists an infinite homography $\mathbf{H}$ between $\mathbf{v}_i$ and $\mathbf{f}_i$ [53]:

$$\mathbf{v}_i \cong \mathbf{K}\begin{bmatrix}\mathbf{R}_c & \mathbf{t}_c\end{bmatrix}\begin{bmatrix}\mathbf{f}_i \\ \mathbf{0}\end{bmatrix} = \mathbf{K}\mathbf{R}_c\mathbf{f}_i = \mathbf{H}\mathbf{f}_i \tag{4.27}$$

where $\mathbf{H} = \mathbf{K}\mathbf{R}_c$, $(\mathbf{R}_c, \mathbf{t}_c)$ is a camera motion and $\cong$ represents an up-to-scale equality in homogeneous coordinates. For each vertical edge $\boldsymbol{\ell}_{ij}$ in i-th image, $\mathbf{v}_i$ and $\boldsymbol{\ell}_{ij}$ are related as follows.

$$\boldsymbol{\ell}_{ij}^\top \mathbf{v}_i = 0 \quad \text{for } j = 1, \ldots, n_i \tag{4.28}$$

A linear system equation for $\mathbf{H}$ is yielded from plugging (4.27) into (4.28) using all of the vertical edge measurements:

$$\boldsymbol{\ell}_{ij}^\top \mathbf{H}\mathbf{f}_i = 0 \quad \xrightarrow[i=1:m,\, j=1:n_i]{\mathcal{S}} \quad \mathbf{M}\mathbf{h} = 0 \tag{4.29}$$

where $\mathbf{M} = \mathcal{S}[\boldsymbol{\ell}_{ij} \otimes \mathbf{f}_i]^\top$ and $\mathbf{h}$ is a vector of $\mathbf{H}$ elements. Since $\mathbf{H}$ is up-to-scale, $\mathbf{h}$ is solvable from at least eight linearly independent constraints. One example is a set of four images containing two vertical edges for each. Finally, $\mathbf{K}$ is obtainable from either QR decomposition of $\mathbf{H}$ or Cholesky decomposition since $\mathbf{H}\mathbf{H}^\top = \mathbf{K}\mathbf{K}^\top$.

Another possible solution is to find $\mathbf{H}$ from a set of constraints, $\mathbf{v}_i \times \mathbf{H}\mathbf{f}_i = 0$. However, this requires an additional step to estimate $\mathbf{v}_i$ from $\{\boldsymbol{\ell}_{ij}\}$ and it cannot deal with the case that a single vertical edge is measured in an image.

Collaborative nonlinear method:

A Euclidean angle between two vertical vanishing points $(\mathbf{v}_i, \mathbf{v}_j)$ should be equal to a vector angle between two gravity loads $(\mathbf{f}_i, \mathbf{f}_j)$, *i.e.*,

$$\cos\theta_{ij} = \mathbf{f}_i^\top \mathbf{f}_j = \frac{\mathbf{v}_i^\top \boldsymbol{\kappa} \mathbf{v}_j}{\sqrt{(\mathbf{v}_i^\top \boldsymbol{\kappa} \mathbf{v}_i)(\mathbf{v}_j^\top \boldsymbol{\kappa} \mathbf{v}_j)}} \tag{4.30}$$

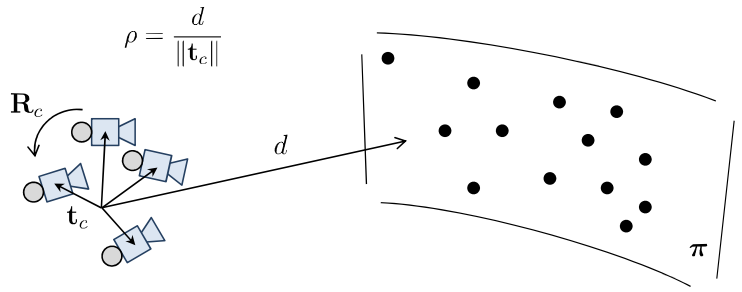


Figure 4.11: Camera/gyroscope calibration: A homography from feature tracks of distant scene points over a known time interval provides the magnitude of an angular rate for the gyroscope calibration.

where the angle between two camera rays is computed by the cosine formula weighted by $\boldsymbol{\kappa} = (\mathbf{K}\mathbf{K}^\top)^{-1}$, which is called the image of the absolute conic [53].

A total cost E_{total} that combines the load constraint $\mathcal{C}^*_{\tau=1}$ and the image angle constraint (4.30) on vanishing point measurements is a function of $\mathbf{q}$ and $\boldsymbol{\kappa}$:

$$E_{total}(\mathbf{q}, \boldsymbol{\kappa}) = E_{imu} + E_{cam/imu} \tag{4.31}$$

$$= \sum_{i=1}^{m} (\mathbf{L}_{ii}\mathbf{q} - 1)^2 + \sum \left(\mathbf{L}_{ij}\mathbf{q} - \frac{\mathbf{v}_i^\top \boldsymbol{\kappa} \mathbf{v}_j}{\sqrt{(\mathbf{v}_i^\top \boldsymbol{\kappa} \mathbf{v}_i)(\mathbf{v}_j^\top \boldsymbol{\kappa} \mathbf{v}_j)}} \right)^2 \tag{4.32}$$

where $\mathbf{f}_i^\top \mathbf{f}_j$ in (4.30) is replaced with $\mathbf{L}_{ij}\mathbf{q} = [\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^\top \mathbf{T}_{16\times 10}\mathbf{q}$ in (4.13).

Due to the nonlinearity with respect to $\boldsymbol{\kappa}$, a nonlinear optimization method numerically seeks a best $(\mathbf{q}, \boldsymbol{\kappa})$ that minimizes the squared sum of the combined constraint error, E_{total}. An initial value for $(\mathbf{q}, \boldsymbol{\kappa})$ can be provided from the cascaded linear solution.

4.6.2 Camera vs. gyroscope

During a pure camera rotation $\mathbf{R}_c$, two images are related by an infinite homography $\mathbf{H}$ such that $\mathbf{x}_i = \mathbf{H}\mathbf{x}'_i$ for every corresponding feature point $(\mathbf{x}_i, \mathbf{x}'_i)$ [53]:

$$\mathbf{H} = \mathbf{K}\mathbf{R}_c\mathbf{K}^{-1} \tag{4.33}$$

Since $\mathbf{H}$ and $\mathbf{R}_c$ are *similar* matrices for a transformation $\mathbf{K}$, they share the same eigenvalues although their eigenvectors are generally different [51]. Since $\mathtt{eig}(\mathbf{H}) = \mathtt{eig}(\mathbf{R}_c) = (1,\, e^{j\theta},\, e^{-j\theta})$ when θ is an angle of $\mathbf{R}_c$, the magnitude of a complex eigenvalue of $\mathbf{H}$ is equal to a rotation angle of the gyroscope affixed to the camera. If a time interval Δt between two images is known, an angular rate $\boldsymbol{\omega}$ exerted on gyroscopes has a magnitude equal to $\|\boldsymbol{\omega}\| = \theta/\Delta t$, although its rotation axis is unknown.

When a set of homographies $(\mathbf{H}_1, \ldots, \mathbf{H}_m)$ is derived from feature tracks over $(\Delta t_1, \ldots, \Delta t_m)$, the factorization method is able to calibrate a gyroscope unit using a load constraint set $\mathcal{C}^* =$

($\|\boldsymbol{\omega}_1\|, \ldots, \|\boldsymbol{\omega}_m\|$). Practically in in-field calibration, the pure rotation assumption of each $\mathbf{H}_i$ is well approximated for distant scene points. Let $\rho = d/\|\mathbf{t}_c\|$ in Figure 4.11 denote a ratio between the distance d to a majority of scene points and camera translation amount $\|\mathbf{t}_c\|$. When $\rho \geq 50$, the mean squared error of θ from $\mathbf{H}$ is less than 1% as evaluated in Figure.

The same homography set $(\mathbf{H}_1, \ldots, \mathbf{H}_m)$ is also used for the rotating camera calibration [54, 55] since the relation $\boldsymbol{\kappa}^{-1} = \mathbf{H}_i \boldsymbol{\kappa}^{-1} \mathbf{H}_i^\top$ yields a set of linear equations for $\boldsymbol{\kappa} = (\mathbf{K}\mathbf{K}^\top)^{-1}$.

Relative orientation:

Once a camera and IMU are both calibrated, the two bundles of motion vectors expressed in each sensor's coordinates are compared to estimate the relative orientation $\mathbf{R}_{ic}$: $\{\mathbf{v}_i, \mathbf{f}_i\}$ for a camera/accelerometer and $\{\boldsymbol{\omega}_i^{cam}, \boldsymbol{\omega}_i^{imu}\}$ for a camera/gyroscope. A motion of the IMU, $\mathbf{f}_i$ and $\boldsymbol{\omega}_i^{imu}$, corresponds to the i-th row of $\mathbf{F}_b$ in the factorization. A camera's angular velocity $\boldsymbol{\omega}_i^{cam}$ is derived from $\mathbf{R}_i = \mathbf{K}^{-1}\mathbf{H}_i\mathbf{K}$ and Δt_i. The best fit of $\mathbf{R}_{ic}$ such that $\mathbf{v}_i = \mathbf{R}_{ic}\mathbf{f}_i$ for $i = 1, \ldots, m$ is computed as a least squares solution $\mathbf{R}_{ic} = \mathbf{U}\mathbf{V}^\top$ from $\mathtt{svd}(\sum_{i=1}^m \mathbf{f}_i \mathbf{v}_i^\top) = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ [56].

4.7 Experiment Results

A series of experiments on the factorization method was conducted using a low-cost small IMU/camera system shown in Figure 4.12 and 4.13. The redundant IMU ($n = 6$) was made of two tri-axial IMUs (O-Navi Gyrocube) which are aligned with a skewed angle. MEMS inertial components are Analog Devices' ADXL-203 and ADXRS-150 whose measuring ranges are $\pm 2g$ and $\pm 200\,^\circ$/sec, respectively. A multi-channel 11-bit A/D converter sampled the output voltage at 100Hz. The Sentech CCD board camera captured 640 $\times$ 480 resolution images at 30 frames/sec.

Accelerometer self-calibration:

To rapidly and reliably collect measurements subject to the gravity magnitude constraint $\mathcal{C}^*_{\tau=1}$, we placed the IMU on a tripod and oriented it within ± 60 degrees. Various attitudes were applied randomly, but we made certain to have at least three different roll or pitch angles so as to prevent a degenerate load configuration $\mathcal{D}_{\mathbf{F}}$ like in Figure B.2 (a-b).

Figure 4.13 shows how the measurement matrix $\mathbf{Z}$ was collected from the output of six accelerometers during the tripod operation. In total, four calibration datasets (E1, ..., E4) were generated with more than 20 loads for each, at different times, several days apart. The noise in $\mathbf{Z}$ was measured as 0.73, which corresponds to $\sigma = 0.2\%$ when scaled by an average scale factor ($\|\mathbf{s}\| = 365.0$).

Note that the redundant calibration treats two tri-axial IMUs as a whole ($n = 6$) and the triad calibration treats each tri-axial IMU individually ($n = 3$, $\mathbf{s}_{123}$ and $\mathbf{s}_{456}$). Table 4.5 com-

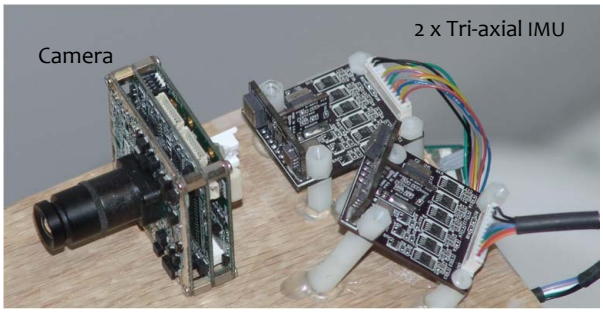


Figure 4.12: An experimental system for the IMU self-calibration. A redundant IMU ($n = 6$) consists of two tri-axial MEMS-based IMUs in a skewed alignment. The camera is used to provide load constraints for the gyroscope calibration.

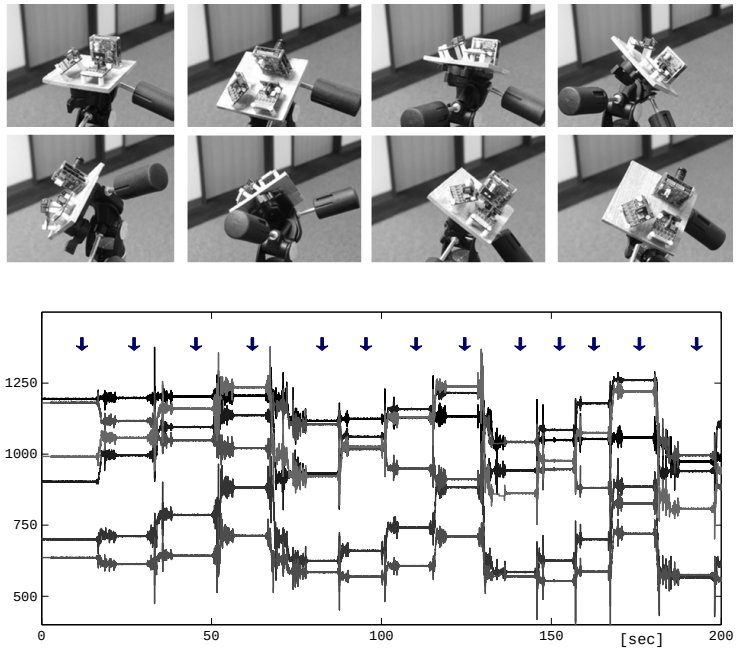


Figure 4.13: The output of six accelerometers generated when the IMU is randomly oriented by a tripod. At each arrow in the plot, the output is collected into the measurement matrix $\mathbf{Z}$ when a static gravity exerts on the accelerometers.

Table 4.5: Accelerometer calibration results from both redundant ($n = 6$) and triad ($n = 3$) calibration methods when $\mathcal{C}^*_{\tau=1}$. (Results are mainly shown for the first three components.)

No.	m	n	b_1	b_2	b_3	b_4	b_5	b_6	$\|\mathbf{s}_1\|$	$\|\mathbf{s}_2\|$	$\|\mathbf{s}_3\|$	$\|\mathbf{s}_4\|$	$\|\mathbf{s}_5\|$	$\|\mathbf{s}_6\|$	$\angle\mathbf{s}_{12}$	$\angle\mathbf{s}_{13}$	$\angle\mathbf{s}_{45}$	$\angle\mathbf{s}_{56}$	$\angle\mathbf{s}_{14}$	$\angle\mathbf{s}_{15}$
E1	36	6	906.7	933.4	927.0	951.4	935.5	918.6	364.5	362.4	366.5	369.1	372.5	367.1	89.89	90.80	89.74	90.17	44.74	51.50
		3	906.5	933.9	926.7	952.6	935.9	920.8	364.2	361.9	366.3	370.3	373.6	369.3	89.84	90.85	89.78	89.94	–	–
E2	25	6	910.1	935.8	928.5	952.7	935.4	918.9	365.7	364.2	366.4	368.8	372.0	366.9	89.90	90.75	89.97	90.18	44.81	51.50
		3	909.6	935.7	928.0	952.8	937.3	921.5	365.7	364.0	365.9	369.6	373.6	369.4	89.90	90.85	90.06	90.11	–	–
E3	24	6	905.9	931.0	923.7	949.9	931.8	915.4	363.7	362.2	364.0	368.0	370.8	364.9	89.84	90.59	89.75	90.03	44.74	51.30
		3	905.5	931.0	922.5	949.3	932.6	916.9	363.5	361.8	362.6	368.9	372.0	366.1	89.82	90.68	89.69	90.22	–	–
E4	20	6	905.3	932.1	927.9	945.0	928.8	915.2	364.7	363.1	367.9	366.1	369.8	367.3	90.11	90.90	90.10	90.31	44.93	51.59
		3	905.3	932.1	922.4	943.8	928.5	916.2	361.8	361.5	362.3	365.7	368.1	368.3	90.08	90.95	90.69	90.39	–	–

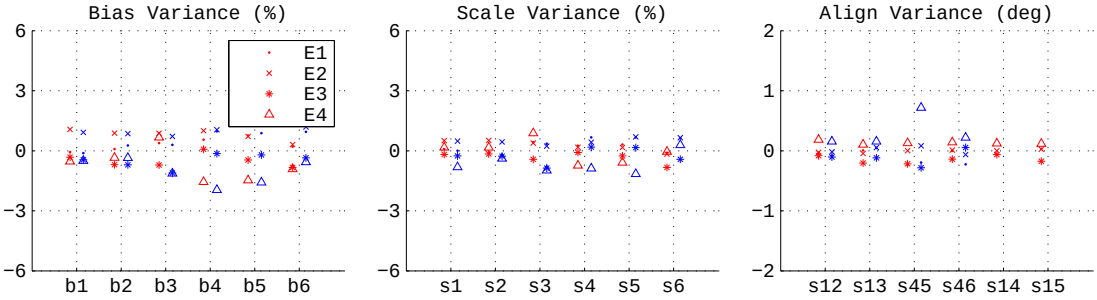


Figure 4.14: Estimation variance of accelerometer parameters: each parameter individually resulting from four experiments is compared using two methods, left (red) from the redundant calibration ($n = 6$) as a whole and right (blue) from the triad calibration ($n = 3$) of each tri-axial IMU in Figure 4.12.

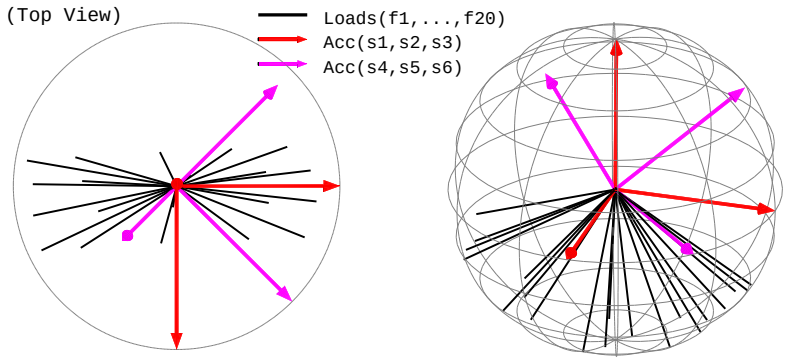


Figure 4.15: The accelerometer's shape and motion matrix ($\mathbf{F}$ and $\mathbf{S}$) are reconstructed from the factorization-based redundant calibration ($n = 6$) in E4 experiment. All of the gravity loads are located on a unit sphere. Two tri-axial IMU are $(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3)$ and $(\mathbf{s}_4, \mathbf{s}_5, \mathbf{s}_6)$.

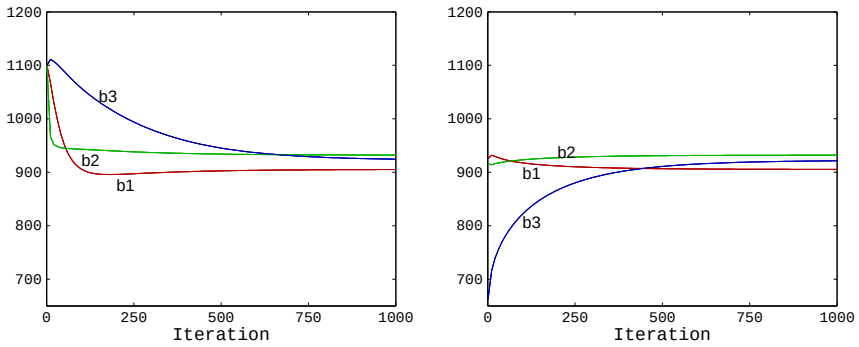


Figure 4.16: Convergence rate of the bias in the iterative linear calibration ($n = 3$) for the first three accelerometer components in E4 experiment. Initial bias $\mathbf{b}^0$ is given as [1100 1100 1100] (left) or mean($\mathbf{Z}$) (right).

pares accelerometer parameters calibrated by these two methods, respectively, and shows little difference between them.

The estimation variance of calibration parameters over four experiments is plotted in Figure 4.14. Deviations of the scale factor $\|\mathbf{s}_i\|$ and alignment $\angle\mathbf{s}_i\mathbf{s}_j$ for each component are less than 0.5% and 0.1°, respectively. They are nearly the same as the simulation evaluation (0.59% and 0.22° when $\sigma = 0.25\%$, $m = 36$ and $n = 6$ in Table 4.3). The bias $\mathbf{b}$ has relatively large deviation of 1% since the operating temperature severely affects the bias of a MEMS component but the temperature was not maintained between the experiments.

An initial bias for the triad calibration was automatically given as the mean of measurements, *i.e.*, $\mathbf{b}^0 = \texttt{mean}(\mathbf{Z})$ and no convergence failure occurred in Table 4.5. The convergence rate in Figure 4.16 illustrates that the bias asymptotically approaches a steady state. For some initial condition, an overshoot is observed at the very beginning of the iteration. The convergence takes about 500 iterations when the distance of $\mathbf{b}^0$ from a true $\mathbf{b}$ is 75% of the gravity magnitude.

Figure 4.15 shows the accelerometer shape and motion matrix, $\mathbf{S}$ and $\mathbf{F}$, reconstructed by the factorization method. Each load in $\mathbf{F}$ located on a unit sphere reveals an applied attitude, and the shape of $\mathbf{S}$ recovers two tri-axial units arranged in a skewed angle.

Gyroscope self-calibration:

Two sets of outdoor images (E5 and E6) in Figure 4.17 were collected at different distances from scene objects while we oscillated the camera on a tripod multiple times with different axes. The camera translation was limited to $\|\mathbf{t}_c\|$ ¡ 0.1m. Since the camera is located far from main buildings (d ¿ 50m for E5 and d ¿ 300m for E6), the pure camera rotation assumption is validated by $\rho \geq 500$.

The measurement matrix $\mathbf{Z}$ was filled with the m peaks of the gyroscope output in Figure 4.17. Although any output could be added in $\mathbf{Z}$, the peaks are practically preferred because they are

expected to have lower variance of angular speed between images, and convenient for automatic measurement selection. The gyroscope noise was measured as 2.0 equal to $\sigma = 0.66\%$ (scaled by $\|\mathbf{s}\| = 300$). Then, two neighboring images (I_i, I_i') closest to each peak $\mathbf{z}_i$ are chosen for the homography computation in Section 4.2.2. More than 200 feature points $\mathbf{x}_i$ are selected in I_i by the Harris corner detector and then tracked to $\mathbf{x}_i'$ in I_i' by KLT feature tracking [57]. Given the feature correspondence $(\mathbf{x}_i, \mathbf{x}_i')$, the homography $\mathbf{H}_i$ such that $\mathbf{x}_i' = \mathbf{H}_i\mathbf{x}_i$ is found based on RANSAC to remove outliers [53]. Finally, the angular speed $\|\boldsymbol{\omega}_i\| = \theta_i/\Delta t_i$ is obtained from $\theta_i = \text{phase}(\text{eig}(\mathbf{H}_i))$ and a fixed frame interval $\Delta t_i = 1/30$s. These m angular speeds $(\|\boldsymbol{\omega}_1\|, \ldots, \|\boldsymbol{\omega}_m\|)$ constitute the load constraint $\mathcal{C}^*$ for the gyroscope calibration ($m = 128$ for E5 and $m = 180$ for E6).

Table 4.6 compares gyroscope parameters found by the redundant ($n = 6$) and triad ($n = 3$) calibration, respectively, in exactly the same way as is done in Table 4.5. Note that an additional comparison is made for the gyroscope bias obtained at a stationary condition ($\boldsymbol{\omega} = 0$). The estimated parameters are almost identical regardless of the calibration method with less than 0.3% deviation. The estimation variance between E5 and E6 is at most 1% at the third and fourth components.

Figure 4.18 shows the reconstructed shape and motion matrix, $\mathbf{S}$ and $\mathbf{F}$, of the gyroscope unit. The distribution of $\mathbf{F}$ reveals that we have rotated the IMU around the same axis multiple times but with different magnitudes.

Camera/accelerometer calibration:

The images and IMU measurements are collected together when the sensor system is placed ahead of office doors to obtain natural landmarks of vertical line segments, as shown in Figure 4.13. Two camera lenses of focal length 3.8mm and 8.0mm are used for the images in Figure 4.19, with different distances to the doors. The ground truth of $\mathbf{K}$ for each lens is precisely calibrated by the DLR camera calibration toolbox [58] using a checkered board. The results in Table 4.5 are used for the ground truth of the IMU calibration.

We use the Canny edge detector to extract straight lines from images. Each edge chain is recursively divided into piecewise segments according to thresholds such as line fitting error and line length. For automatic selection of vertical edges among all those detected, the lines are clustered according to vanishing points in a RANSAC manner [25]. Either a dominant line group or the one consistent with IMU loads are selected as a true vertical line group $\mathcal{L}_i$ in each image. Then, a maximum likelihood estimate of the vanishing point $\mathbf{v}_i$ for $\mathcal{L}_i$ is found through nonlinear minimization of the squared sum of orthogonal image distances when the image noise is Gaussian [59].

Table 4.8 compares camera calibration results between the cascaded linear method and the

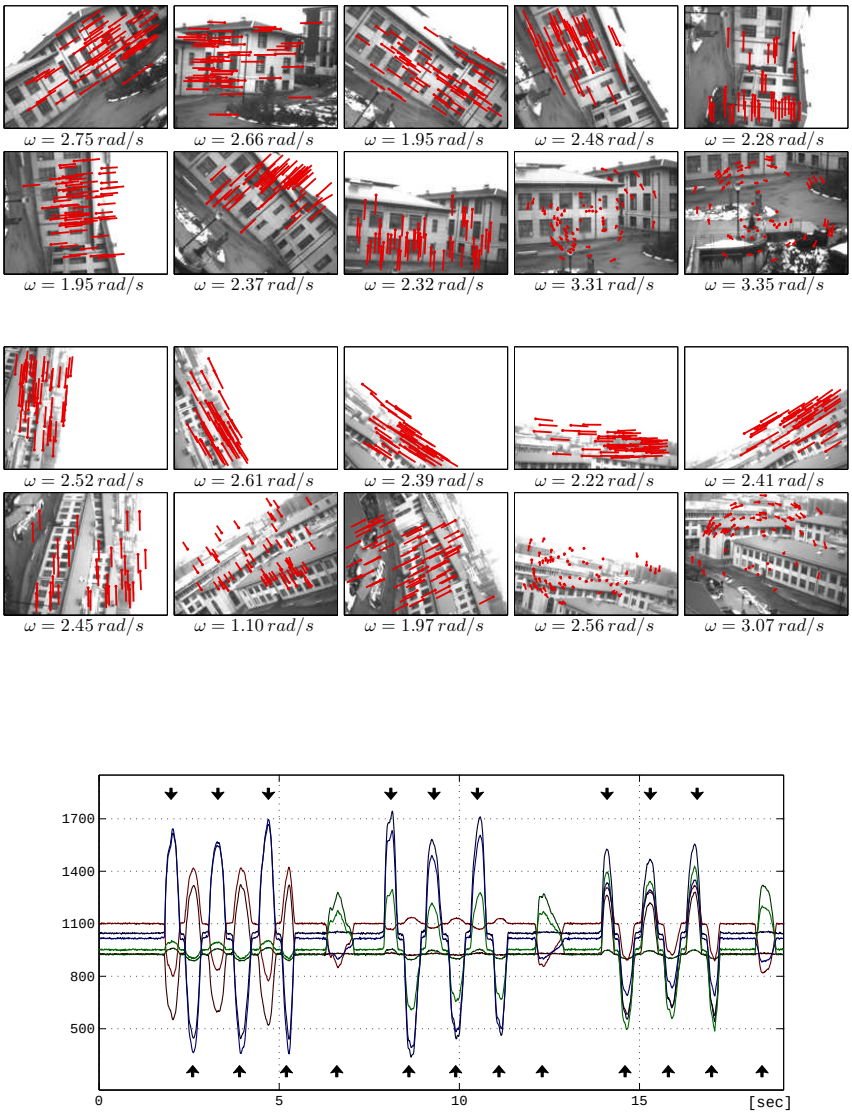


Figure 4.17: (Top) Feature correspondences of distant scene points between two neighboring frames around the peaks of the gyroscope output at bottom. (Bottom) The output of six gyroscopes generated when the IMU is oscillated multiple times in different axes by a tripod.

Table 4.6: Gyroscope calibration results from the redundant ($n = 6$) and triad ($n = 3$) calibration methods when $\mathcal{C}^*$ is given by image feature tracks.

No.	m	n	b_1	b_2	b_3	b_4	b_5	b_6	$\|\mathbf{s}_1\|$	$\|\mathbf{s}_2\|$	$\|\mathbf{s}_3\|$	$\|\mathbf{s}_4\|$	$\|\mathbf{s}_5\|$	$\|\mathbf{s}_6\|$	$\angle\mathbf{s}_{12}$	$\angle\mathbf{s}_{13}$	$\angle\mathbf{s}_{45}$	$\angle\mathbf{s}_{56}$	$\angle\mathbf{s}_{14}$	$\angle\mathbf{s}_{15}$
E5	128	6	925.5	924.3	1047.0	1099.9	952.7	1017.3	291.2	296.3	291.7	299.9	280.3	298.5	92.42	90.78	91.09	89.43	44.99	51.88
		3	924.9	925.0	1047.2	1099.3	953.5	1017.1	290.8	296.2	291.6	299.4	280.0	298.5	92.62	90.69	91.12	89.42	–	–
		$\boldsymbol{\omega}=0$	926.9	923.6	1046.0	1100.0	953.4	1015.7	–	–	–	–	–	–	–	–	–	–	–	–
E6	180	6	923.1	925.2	1041.6	1091.5	948.9	1015.0	290.0	294.2	290.2	292.4	286.3	297.2	90.09	89.70	90.88	89.11	46.25	50.43
		3	922.8	924.6	1041.3	1091.0	948.5	1014.8	289.8	294.3	290.3	293.3	285.3	297.2	90.12	89.71	90.87	89.07	–	–
		$\boldsymbol{\omega}=0$	927.4	925.1	1042.0	1094.7	952.3	1013.8	–	–	–	–	–	–	–	–	–	–	–	–

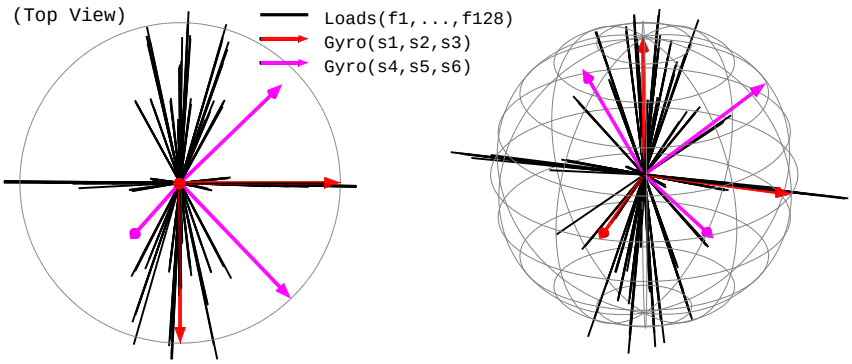


Figure 4.18: The gyroscope's shape and motion matrix ($\mathbf{F}$ and $\mathbf{S}$) reconstructed from the factorization-based redundant calibration ($n = 6$) in experiment E5. Each angular velocity in $\mathbf{F}$ has a different magnitude. Two tri-axial IMU are $(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3)$ and $(\mathbf{s}_4, \mathbf{s}_5, \mathbf{s}_6)$.

Table 4.7: Comparison between the factorization and constrained nonlinear optimization in the accelerometer calibration (E4). E_Z and $E_\mathcal{C}$ are in the RMSE (root mean squared error). $\Delta\mathbf{b}$ is a mean difference of $\mathbf{b}$ and so are $\Delta\|\mathbf{s}\|$ and $\Delta\angle\mathbf{s}_{ij}$.

	Factorization		Nonlinear		Parameter difference		
No.	E_Z	$E_\mathcal{C}$	E_Z	$E_\mathcal{C}$	$\Delta\mathbf{b}$	$\Delta\|\mathbf{s}\|$	$\Delta\angle\mathbf{s}$
E1	0.0934	0.0009	0.1917	0.0000	0.002	0.005	0.002
E2	0.1226	0.0006	0.1763	0.0000	0.010	0.023	0.003
E3	0.1225	0.0007	0.1729	0.0000	0.024	-0.016	0.006
E4	0.1159	0.0009	0.1700	0.0000	-0.017	0.010	-0.009

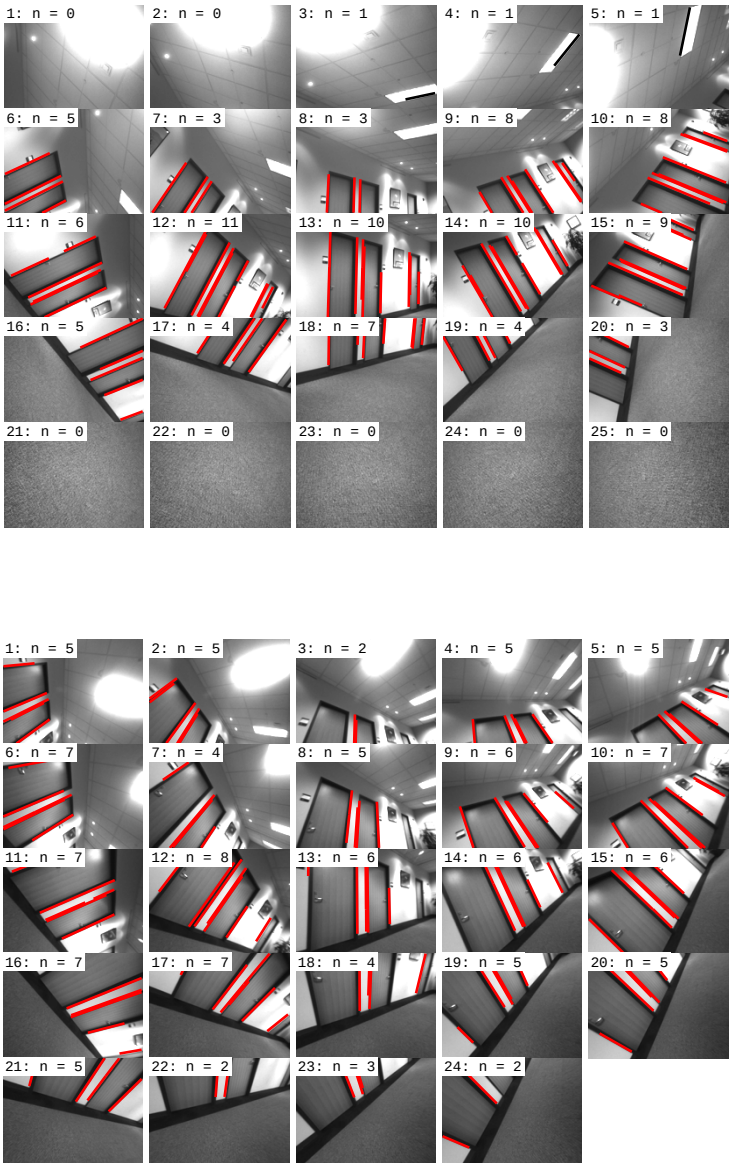


Figure 4.19: (Top) Images taken with an 8.0mm camera lens at 25 different attitudes (E2). The vertical edges along the gravity direction at office doors produce vanishing points in 15 images. (Bottom) Images taken with a 3.2mm camera lens at 24 different attitudes (E3). The vanishing points are produced in all of the images. The bottom images are taken closer to the doors than those images at the top.

Table 4.8: Camera/accelerometer calibration result (use all the measurements, $m = 25$ or 24)

	f_x	f_y	s	c_x	c_y	$\mathbf{b}$	$\|\mathbf{s}\|$	$\angle\mathbf{s}$
E2	1083.29	1083.25	0.00	303.29	247.34	930.24	367.41	–
LIN	30.30	26.33	-6.35	1.20	-10.33	0.00	0.00	0.00
NON	-6.18	-9.94	0.00	0.96	-9.09	-0.70	-0.90	-0.08
E3	361.72	360.39	0.00	301.28	230.65	926.38	365.80	–
LIN	-2.24	-5.62	-1.80	1.73	-1.92	0.00	0.00	0.00
NON	-5.54	-7.76	0.00	0.36	-3.79	-0.28	-0.68	-0.07

E#: ground truth

LIN: cascaded linear method errors, NON: collaborative nonlinear errors

Camera calibration matrix $\mathbf{K} = [f_x \;\; s \;\; c_x; f_y \;\; 0 \;\; c_y; 0 \;\; 0 \;\; 1]$

Table 4.9: Camera/accelerometer calibration result (under the minimal condition, $m = 9, n = 4$)

No.	f_x	f_y	s	c_x	c_y	$\mathbf{b}$	$\|\mathbf{s}\|$	$\angle\mathbf{s}$
E2	1083.29	1083.25	0.00	303.29	247.34	931.76	366.35	–
LIN	49.67	46.72	-2.44	-6.46	-18.00	3.18	-2.07	-0.51
NON	-20.75	-19.16	0.00	-8.96	8.67	-1.85	-1.55	-0.12
E3	361.72	360.39	0.00	301.28	230.65	927.66	364.65	–
LIN	6.93	-23.10	-0.58	-17.23	8.09	1.16	14.43	1.92
NON	-3.41	-8.64	0.00	-6.16	-1.10	-2.61	2.56	0.64

In Figure 4.19, only nine images and their corresponding $\mathbf{z}_i$ are selected.

collaborative nonlinear method in Section 4.6, when all of the available measurements and constraints are used. The linear method reports an acceptable camera calibration performance for both camera lenses. The maximum errors are bounded to a 2.7% focal length and 4.1% principal point. The nonlinear method reduces the focal length error to 0.9% in E2, while producing essentially no changes in IMU calibration parameters. Note that the camera skew s is set to zero in the nonlinear method in order to prevent overfitting.

The advantage of the collaborative nonlinear calibration is more clearly revealed in an unfavorable condition where smaller calibration data sets are available. We intentionally limit the data to the minimal condition for the linear method. Only nine IMU measurements and corresponding images are selected and used for the calibration of four sensor components ($m = 9$ and $n = 4$). The results in Table 4.9 show a significant reduction in both camera and IMU calibration errors in the nonlinear method. The maximum errors are reduced from 6.3% to 2.3% in the focal length, from 7.2% to 2.9% in the principal point, and from 3.8% to 0.5% in the IMU scale; these reductions are caused by additional angle constraints in (4.30) between IMU and camera data.

Camera/gyroscope calibration:

From the desk and aerial scenes, we selected 30 camera motion pieces ($n = 30$) from each scene, either in a steady rotation or at a local maxima of motion speed. Feature correspondences were collected over two frames for the desk scene and four frames for the aerial scene. Some examples of feature tracks are shown in Figure 4.20. The distance ratio ρ is roughly 20 for the desk scene ($\|\mathbf{t}\| = 10cm$, $d = 2m$) and 50 for the aerial scene ($\|\mathbf{t}\| = 2m$, $d = 100m$). The ground truth of $\mathbf{K}$ is obtained using a planar checkered board by the DLR camera calibration toolbox [58], while that for $\mathbf{S}$ is obtained from a product datasheet. The ground truth of $\mathbf{R}_{ic}$ is measured from a mechanical drawing of the sensor plate in Figure 4.12 and corresponds to a zero angle.

Table 4.10 shows the automatic calibration results of the aerial scene obtained from linear methods. As expected from the simulation results, the refinement step significantly increases calibration accuracy compared with initial linear solutions. Note that the bias $\mathbf{v}$ is excluded in the total refinement, since $n = 30$ is not sufficient for preventing the overfitting of increased parameters. In the aerial scene, the linear camera calibration fails in Cholesky decomposition due to the high noise presented in the homographies, which are derived from feature tracks in low-resolution images. The alignment in the gyro shape $\mathbf{S}$ is relatively less accurate than other parameters because the quality of calibration inputs is limited; rotation angles from homographies are slightly biased and unsteady motions during Δt cause the gyro measurements to become more noisy than the feature tracks. Based on the ability of the single-parameter model $\mathbf{S}^*$ to return a smaller calibration error, we can deduce that the calibration dataset is not informative enough to precisely recover the gyro's internal alignment.

It is noteworthy that camera motions in the aerial scenes are not specifically intended for the purpose of calibration, but for typical use of a hand-held device and aerial robot during normal operation. This fact clearly demonstrates that our method works for on-the-fly calibration when it is necessary to run gyro-aided feature tracking with no calibration priors in real scenarios. A calibration dataset collected as principled a way as the simulation input would contain more informative constraints and further improve the calibration accuracy.

Factorization method vs. nonlinear optimization:

It is possible to cast IMU calibration as a constrained nonlinear optimization problem. Using the following cost E_{imu} for a given measurement $\mathbf{Z}$ and a load constraint set $\mathcal{C}$, optimal parameters $(\mathbf{F}, \mathbf{S}, \mathbf{b})$ can be found by minimizing E_{imu}.

$$E_{imu} = E_Z + E_{\mathcal{C}} \tag{4.34}$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{n}(\mathbf{f}_i^\top \mathbf{s}_j + b_j - z_{ij})^2 + \lambda \sum_{(i,j)\in\mathcal{C}}^{k}(\mathbf{f}_i^\top \mathbf{f}_j - c_{ij})^2 \tag{4.35}$$

where λ is a Lagrange multiplier for the constraint $\mathcal{C}$. This cost function is the same as that for the *bundle adjustment* [60] in computer vision, which refines a visual reconstruction to produce

Table 4.10: Auto-calibration results on the aerial scene experiment. A total of 30 homographies were used ($n = 30$). The Cholesky decomposition failed due to high noise in the homographies.

	$\mathbf{K}$ (320×240)	$\mathbf{S}$	Euler$(\mathbf{R}_{ic})$(deg)
Ground-truth	$\begin{bmatrix} 286.34 & 0.01 & 5.02 \\ 0 & 286.53 & -0.72 \\ 0 & 0 & 1.00 \end{bmatrix}$	$\begin{bmatrix} 298.51 & 0 & 0 \\ 0 & 298.51 & 0 \\ 0 & 0 & 298.51 \end{bmatrix}$	$\begin{bmatrix} 0.00 \\ 0.00 \\ 0.00 \end{bmatrix}$
Linear	Cholesky fails	$\begin{bmatrix} 265.10 & -108.67 & -76.82 \\ 0 & 239.63 & -35.81 \\ 0 & 0 & 245.34 \end{bmatrix}$	-
Linear*	$\begin{bmatrix} 208.69 & 0 & 0 \\ 0 & 208.69 & 0 \\ 0 & 0 & 1.00 \end{bmatrix}$	$\begin{bmatrix} 281.83 & 0 & 0 \\ 0 & 281.83 & 0 \\ 0 & 0 & 281.83 \end{bmatrix}$	$\begin{bmatrix} -3.45 \\ 1.94 \\ -8.06 \end{bmatrix}$
Refinement	$\begin{bmatrix} 270.38 & 3.58 & -2.19 \\ 0 & 272.28 & -44.48 \\ 0 & 0 & 1.00 \end{bmatrix}$	$\begin{bmatrix} 259.23 & 13.01 & -78.94 \\ 0 & 274.06 & -56.90 \\ 0 & 0 & 285.88 \end{bmatrix}$	$\begin{bmatrix} 2.50 \\ 0.79 \\ -9.03 \end{bmatrix}$

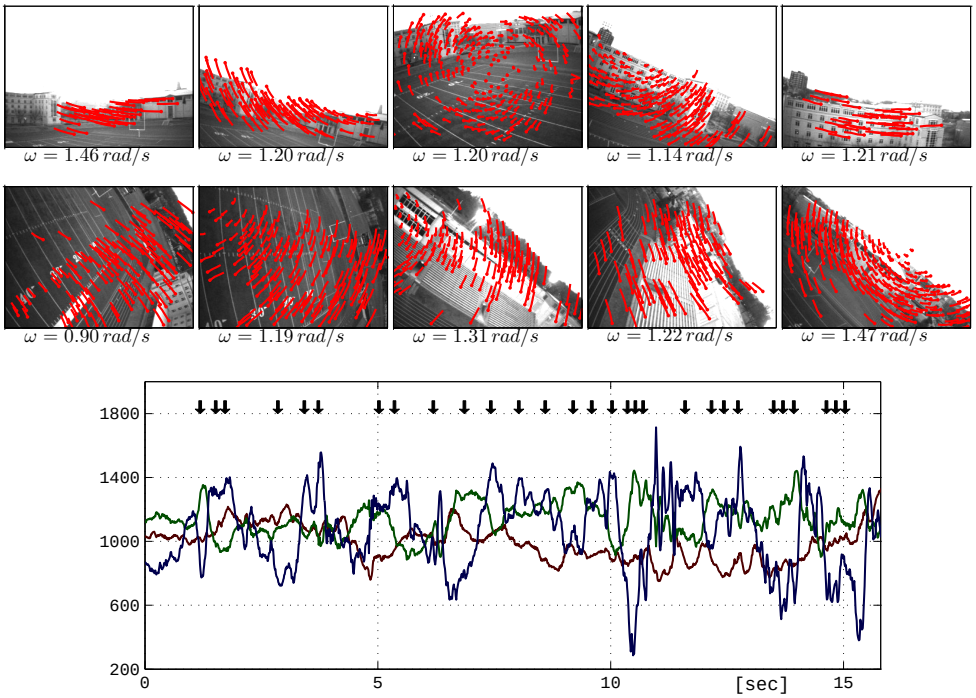


Figure 4.20: Feature correspondences and the output of tri-axial gyroscopes used for online calibration during the aerial maneuvering

jointly optimal structure and camera poses.

Generally, many solutions to this nonlinear optimization are numerically ill-conditioned or result in computationally complex approaches with many convergence hurdles. Moreover, high sensitivity to parameter variations requires a good initial parameter to avoid local minima.

In contrast, the factorization method is not only numerically stable but also globally optimal. The SVD of $\mathbf{Z}$ in (4.5) using the proper rank guarantees that E_Z in (4.34) is always at a global minimum regardless of the subsequent reconstruction step. A minimum E_Z is equal to $\|\mathbf{Z} - \widehat{\mathbf{F}_b}\widehat{\mathbf{S}_b}\|_{\mathbf{F}}$ in (B.1) and a minimum $E_{\mathcal{C}}$ is achieved by the least squares solution $\mathbf{Lq} = \mathbf{c}$ in (4.13).

Table 4.7 shows that the factorization and nonlinear optimization have almost the same minimum cost and the calibration parameter difference between them is negligible. A small difference between E_Z and $E_{\mathcal{C}}$ is caused by the fact that the constrained optimization strictly enforces $E_{\mathcal{C}}$ to zero, while $E_{\mathcal{Z}}$ increases slightly as a trade-off.

Chapter 5

Motion Planning and Control

The problem of motion planning for a fixed-wing aircraft is more difficult than for other types of aircraft. In general non-acrobatic maneuvers, fixed wings cannot hover and climb vertically or make zero radius turns because they require a minimum forward velocity to maintain the lift against gravity. Given the change of a target's location and orientation, the corresponding vehicle motion should be planned deliberately, as turning requires significant time and space.

The goal of this chapter is to develop a real-time motion planning method that allows a fixed-wing UAV to operate toward a target in the presence of obstacles and to reliably accomplish the air-slalom task in Figure 2.1. In order to respond flawlessly to newly identified targets and obstacles in the task, a motion planner needs to arrange a feasible path that respects kinematic and dynamic constraints of fixed-wings. Computational load should also be considered carefully for fast replanning with no significant delays when facing limited computation resources.

In this chapter, we present two approaches for this motion planning problem. Both are based on a set of motion primitives that are sampled from controllable vehicle behaviors and can be interconnected at trim states. The first one is an online search-based planning method that explores an incremental reachability motion tree to find an obstacle-free path. The search is efficiently bounded by a 3D distance metric that is based on 2D Dubins curves. The second is a global planning method that precomputes optimal paths for all of the discretized vehicle states in an obstacle-free space. The motion uncertainty due to external disturbances will be explicitly considered by a probabilistic motion model using a Markov Decision Process.

5.1 Related Work

The problem of UAV motion planning is especially challenging due to several complexities that have not been addressed by traditional planning strategies. Differential constraints become more important to find a feasible trajectory. Wind disturbance makes it impossible for the vehicle

to precisely follow a pre-computed plan. Limited sensor capabilities increase uncertainty of not only vehicle and target states but also the knowledge about the environment. Previous research in robotics has explored a vision-based navigation [61], basic obstacle avoidance during flight [62] and 3D local trajectory planner with a predefined global path [63]. There was a reactive steering for flocking, grouping and obstacle avoidance [64] and an online search and trajectory precomputation [65], both of which are targeted for computer animation.

Many planning algorithms to produce 2D or 3D trajectories for autonomous aircraft guidance in known or unknown environments are based on a decomposition approach; first solve a dynamics-unconstrained planning problem, and then smooth the found path to a feasible trajectory using differential constraints. Traditional and state-of-the-art planning algorithms in textbooks [66, 67, 68, 69] and review papers [70, 71, 72, 73] may be used in this approach.

Our approach to planning with differential constraints is closely related with a finite-state motion model called a *maneuver automation* [74]. Trim states of an air vehicle has been effectively used for motion planning of a complicated nonlinear and high-dimensional dynamic system [74, 75, 76, 77, 78, 79, 80]. The concept of MA is based in part on the fact that human pilots can accomplish nimble control using a mixture of trim trajectories and maneuvers. The general idea of finite state models is to search or optimize a path connected in a discretized finite-dimensional space instead of an infinite one. This significantly reduces the computational complexity of a motion planning problem under differential constraints.

5.2 Motion primitives

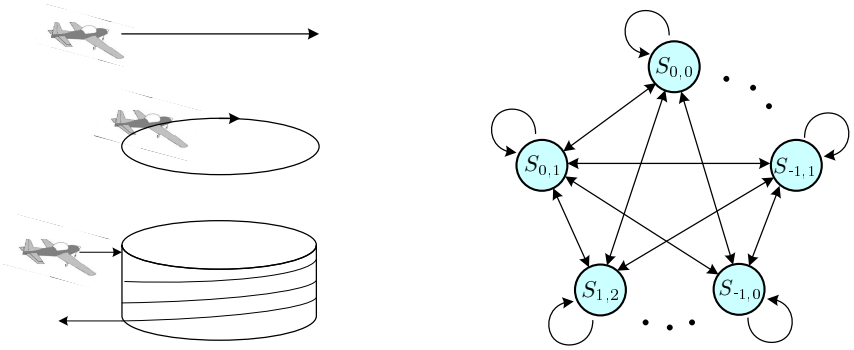
Let a nonlinear differential system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ be the dynamics of a fixed-wing airplane. The vehicle state $\mathbf{x}$ is a high dimensional variable, $\mathbf{x} = (x, y, z, \phi, \theta, \psi, u, v, w, p, q, r, \alpha, \beta)$, and the control inputs $\mathbf{u}$ are a thrust and surface deflections. Using a feedback controller $\mathbf{u}^*$ during time k to $k+1$, this complex dynamic system can be reduced to a discretized dynamic system $\mathbf{f}_d$ that makes the transition from a trim state $\mathbf{s}_k$ to another trim state $\mathbf{s}_{k+1}$.

$$\mathbf{s}_{k+1} = \mathbf{f}_d(\mathbf{s}_k, \mathbf{u}^*(k, k+1)) \tag{5.1}$$

Since the dimension of a trim state is much lower than that of the original state, the motion planning on a trim-state space is more advantageous to run in realtime when a set of motion primitives are built in such a way that they interconnect discretized trim states.

5.2.1 Trim states

In aerodynamics literature [81], a fixed-wing's *trim* state is referred to as one of the following three motions: (1) level flight, (2) constant-altitude turn, and (3) turn with a constant climb



(a) (b)

Figure 5.1: Trim states of a fixed-wing airplane: (a) three motions in a trim state: level flight, constant altitude turning, and turning with a constant climb rate. (b) a finite state machine (FSM) defined by trim states $\mathbf{s}(\phi, \theta, V_a)$.

rate, as illustrated in Figure 5.1(a). Though an equilibrium state $\dot{\mathbf{x}} = \mathbf{0}$ is generally called trim, trim states of a fixed wing include cases where some vehicle states are not constant. In turning with a constant climb rate, the altitude and the yaw angle change at constant rates ($\dot{z} = V_a \sin\gamma$ and $\dot{\psi} = V_a/R$) when a climbing angle γ, turning radius R and airspeed V_a are given.

In the trim conditions above, the position (x, y, z) and the heading ψ in an inertial frame are irrelevant to the description of the motions and the remaining states are constrained by the dynamics $\mathbf{f}$ and three parameters (γ, R, V_a). Since γ is related to the pitch θ by an attack angle α and the roll $\phi = V_a/R$ at the constant speed V_a, all of the trim conditions can be described by the following *trim state* $\mathbf{s}$:

$$\mathbf{s} = (\phi, \theta, V_a) \tag{5.2}$$

The control input $\mathbf{u}^*$ in (5.1) is required to drive the airplane to a given trim state. Suppose that there exist three controls (*e.g.*, thrust, rudder and elevator) that are generally available in a small model airplane. Given a reference profile for desired vehicle states $(\phi_d, \theta_d, V_{a,d})$, the following PID (proportional-integral-derivative) controllers are used to minimize the error between measured and desired states. Each controller is independently dedicated to control the corresponding trim variable and is implemented in autopilot.

$$u_t(t) = K_{p,t}(V_{a,d} - V_a) + K_{i,t}\int (V_{a,d} - V_a) \qquad \text{airspeed hold} \tag{5.3}$$

$$u_r(t) = K_{p,r}(\phi_d - \phi) + K_{d,r}(\dot{\phi}_d - \dot{\phi}) + K_{i,r}\int (\phi_d - \phi) \qquad \text{roll hold} \tag{5.4}$$

$$u_e(t) = K_{p,e}(\theta_d - \theta) + K_{d,e}(\dot{\theta}_d - \dot{\theta}) + K_{i,e}\int (\theta_d - \theta) + \theta_{level} \qquad \text{pitch hold} \tag{5.5}$$

where θ_{level} is an angle of attack that holds an altitude at the air speed V_a. Vehicle behaviors are then governed by the closed-loop dynamic system $\mathbf{f}_d$, in which the past desired trim state $\mathbf{s}_k$ advances to a new desired state $\mathbf{s}_{k+1}$ by the feedback control $\mathbf{u}^*$ during time T_s.

5.2.2 Generation of motion primitives

There exist infinitely many motion primitives, which of each makes a transition between two points in a continuous space of trim states. To select a limited number of transitions in building motion primitives, we confine the trim states to a one-dimensional space $\mathbf{s}_i^* = \mathbf{s}(\phi_i, \theta^*, V_a^*)$ where θ^* and V_a^* are predefined constants. The roll angle ϕ_i is discretized and its change to another trim state creates a *lateral* motion primitive. While the pitch angle θ is always θ_{level} at both ends of a motion (*i.e.*, $\theta^* = \theta_{level}$ at trim), it is not necessarily constant during a transition so that it can create a *longitudinal* motion primitive.

Figure 5.2(a) shows the ramp and trapezoidal motion profiles of desired roll and pitch angles ($\phi_d(t)$ and $\theta_d(t)$), respectively, which are used in the PID controllers during a transition time T_s between the discretized trim states $\mathbf{s}^*$. For the pitch, the peak of $\theta_d(t)$ determines the amount of altitude change. Each profile provides a sufficient settling time so that the error from a final value remains small and the angular rates (p, q, r) become stationary under small ripples. We give a rise time $T_r = 0.5s$ and $T_s = 2s$ for lateral motions and $T_s = 3s$ for longitudinal motions because the pitch angle requires additional time to be regulated.

A set of motion primitives in Figure 5.3(b) is generated using nine discrete roll angles and five different pitch peaks, $\phi_i = \{-60°, -45°, -30°, -15°, 0°, 15°, 30°, 45°, 60°\}$ and $\theta_d(t)_{peak} = \{-30°, -15°, 0°, 15°, 30°\}$, so that each starting $\mathbf{s}_i^*$ has 45 motion branches reaching to another $\mathbf{s}^*$. More complex vehicle behaviors are possible when these motion primitives are interconnected. Given a dynamic model, motion primitives can be computed offline and saved in a look-up table. Figure 5.4 shows a reachability tree when the vehicle begins a level flight and the node expansion stops at the third depth.

Relationships between the motion primitives can be described by a deterministic finite state machine (FSM) between the trim states $(\mathbf{s}_1^*, \ldots, s_9^*)$, as shown in Figure 5.1(b). In the FSM, each motion primitive is equivalent to a transition function from one state $\mathbf{s}_i^*$ to another $\mathbf{s}_k^*$. For each pair of $(\mathbf{s}_i^*, \mathbf{s}_k^*)$, there exist five different longitudinal motion primitives depending on $\theta_d(t)$.

5.3 Search-based Motion Planning

Given the motion library $\mathcal{F}$ in Figure 5.3, a motion planning problem for the air-slalom task is to find a path that connects two points in the configuration space $\mathcal{C} = \{x, y, z, \theta, \phi\}$. In this discrete planning one may use any graph search algorithm such as depth-first or A* to find a path from

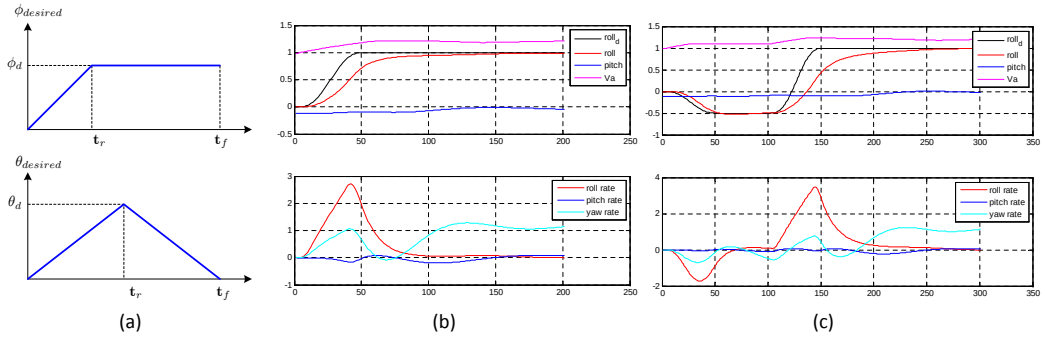


Figure 5.2: PID control response between one-dimensional trim states $\mathbf{s}^*$: (a) roll reference profile $\phi_d(t)$ for a lateral motion primitive and pitch reference profile $\theta_d(t)$ for a longitudinal motion primitive, (b) roll angle control response in simulation, (c) pitch angle control response in simulation.

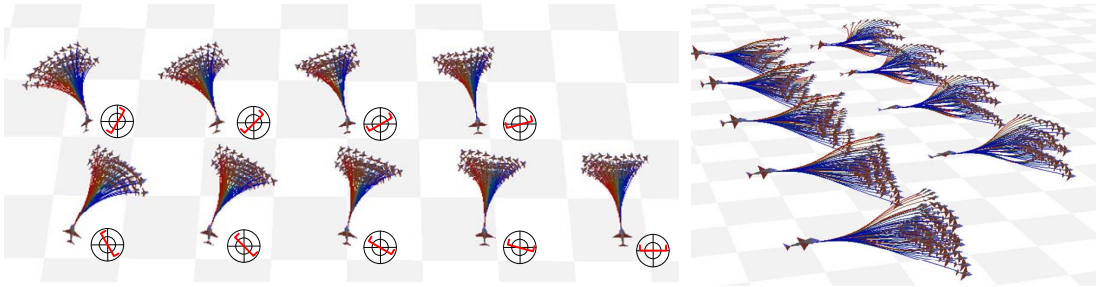


Figure 5.3: A set of motion primitives that make transition between one-dimensional trim states $\mathbf{s}^*$ ($\theta^* = \theta_{level}, V_a^* = V_{cruise}$). Each set consists of nine discretized ϕ_k in a resolution of 15° for lateral motions and five different θ profiles for longitudinal motions.

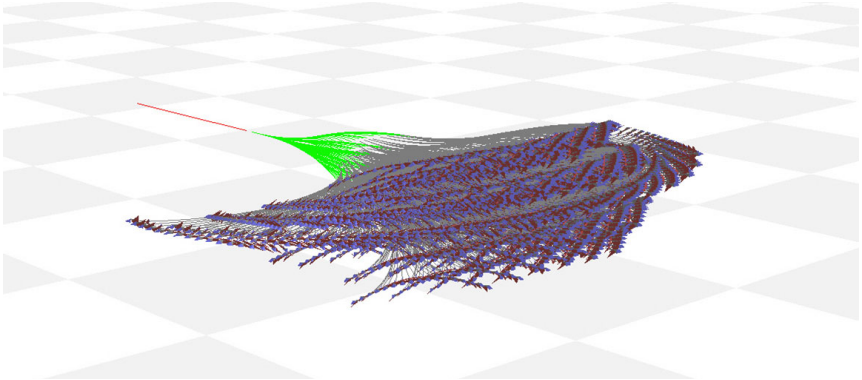
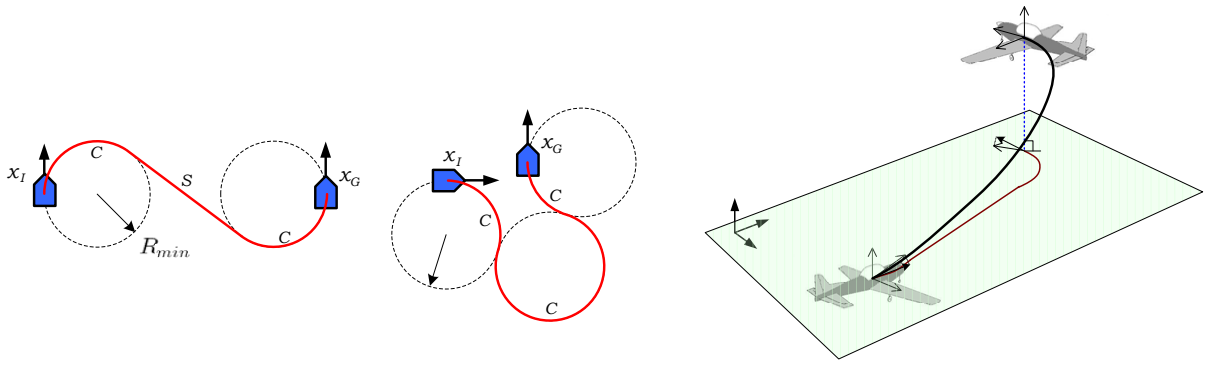


Figure 5.4: A reachability tree of the motion primitives in Figure 5.3: when expanded by nine lateral and five longitudinal actions at each node, the third stage produces 91,125 new nodes.



(a) (b)

Figure 5.5: A heuristic function that estimates the optimal distance to a goal: (a) Two examples of 2D Dubins curves are the path with the shortest length between $\mathbf{x}_I, \mathbf{x}_G \in SE(2)$ (b) The 3D Dubins metric in the configuration space $\mathcal{C} = \{x, y, z, \phi\}$ for the air-slalom task

.

a starting state to a goal state. A search tree like that in Figure 5.4 is incrementally constructed through a state transition in $\mathcal{F}$.

Among many search algorithms, a selection of affordable online planning is restricted by the fact that the branching factor b at each node is quite large in order to cover lateral as well as longitudinal motions ($b = 45$). If a goal is placed d steps away from a starting point ($d = 10 \sim 15$ for sufficient space for obstacle avoidance), the search tree size in an exhaustive search like the breadth-first would grow exponentially with respect to d, *i.e.*, $\mathcal{O}(b^d)$. Hence, the trade-off between path quality and computational load needs to be considered.

An informed search like the best-first is more appropriate for online planning, but it requires accurate heuristic information about the distance to a goal in order to reduce search time. Therefore, we approximate the shortest path to a goal using a Dubins-based heuristic, which closely estimates trim state-based 3D behaviors of a fixed-wing.

5.3.1 3D Dubins heuristic

The Dubins curves have been used as an efficient distance metric for car-like mobile robots [67, 82, 83]. Like the examples in Figure 5.5(a), when a turning radius is lower bounded, the shortest path between two configurations $(\mathbf{x}_I, \mathbf{x}_G) \in SE(2)$ can be expressed as a combination of no more than three motion primitives. The motion primitive for a car is either a straight line or circular path of a minimum radius.

The trim state-based motion primitives in Figure 5.3 can also be considered to be bounded by a minimum horizontal turning radius R_{min} and a maximum climbing rate μ_{max}. Therefore, it is possible to approximate their motions using the following helical kinematic equation [69], in

Algorithm 5.1: Computation of 3D Dubins distance metric

d_{3D} = Dubins_3D_Metric ($\mathbf{c}_i(x_i, y_i, z_i, \phi_i)$, $\mathbf{c}_k(x_k, y_k, z_k, \phi_k)$; R_{min}, μ_{max})
begin
  d_{2D} = Dubins_2D((x_i, y_i, ϕ_i), (x_k, y_k, ϕ_k))
  $\Delta z = |z_i - z_k|$
  while $\Delta z / d_{2D} > \sin(\mu_{max})$ **do**
    $d_{2D} \leftarrow d_{2D} + 2\pi R_{min}$
  Return $d_{3D} = \sqrt{(d_{2D})^2 + (\Delta z)^2}$

which two inputs are a roll angle ϕ and altitude change rate μ:

$$\dot{x} = \cos\theta, \quad \dot{y} = \sin\theta, \quad \dot{\theta} = \tan\phi, \quad \dot{z} = \mu \tag{5.6}$$

where $\tan\phi_{max} = 1/R_{min}$ and the air speed is normalized to 1. A non-zero constant input ϕ and μ to (5.6) generates a helical motion, which is the same as the trim motion that turns with a constant climb rate in Figure 5.1(a).

When a vehicle is located at $\mathbf{c}_i = (x_i, y_i, z_i, \phi_i)$ with respect to a goal point $\mathbf{c}_g = (x_g, y_g, z_g, \phi_g)$, the distance between two points $d(\mathbf{c}_i, \mathbf{c}_g)$ is approximated by the traveling distance of the kinematic equation (5.6) whose inputs are fixed to two parameters R_{min} and μ_{max}. In Figure 5.5(b), the 3D path length can be approximated as follows: (i) compute a 2D Dubins distance metric d_{2D} on the horizontal plane while altitude change is ignored, (ii) add one revolution of a circle with a radius R_{min} until the horizontal traveling distance d_{2D} is long enough to satisfy the maximum climbing angle μ_{max}, (iii) return a combined distance of the horizontal distance d_{2D} and the vertical altitude change. See Algorithm 5.1 for further detail.

Using the phase partitions between the words in Dubins curves [83], the path length d_{2D} can be quickly obtainable without expensive computation. One concern is discontinuity of the distance metric at partition boundaries. A small change in configurations may cause a big change in the estimate to a goal near partition boundaries in sampling-based motion planning. In practice, therefore, we allow a small margin $\pm 10^\circ$ for the goal orientation ϕ_g and find a minimum under the margin to determine the shortest distance for d_{2D}.

5.3.2 Greedy best-first search

A greedy best-first search in Algorithm 5.2 sorts a priority queue according to the heuristic function d_{3D} so that the node estimated closest to a goal is exploited first. The path found in this way is not guaranteed to be optimal. In most cases, however, far fewer nodes than the worst case $\mathcal{O}(b^d)$ are explored until the goal is found, hence, much faster running time is expected. The use

Algorithm 5.2: Greedy best-first search using trim state-based motion primitives

```
p = Greedy_Best_First_Search (x_start, x_goal; F, O)
Set priority queue Q = ∅, k_max
begin
    Q.insert(x_s, Dubins_3D_Metric(x_start, x_goal))
    while Q ≠ ∅ or k < k_max do
        x = Q.top()
        if x ∈ G(x_goal) then
            Return p = Trace_Back(x)
        else
            foreach f_i ∈ F(x) do
                x_new = Motion_Primitive_Transition(x, f_i)
                if x ∉ O of obstacles then
                    Q.insert(x_new, Dubins_3D_Metric(x_new, x_goal))
                k ← k + 1
```

of an efficient heuristic is critical to improve the quality of a path and avoid the worst case in terms of time and space complexity.

The search begins with building a tree and assigning a current vehicle state as its root. By applying all of the motion primitives available at the current vehicle state, a new set of nodes corresponding to unvisited states is generated at the next depth level. If any point on a motion primitive is in any obstacle region, that node is discarded from a tree. The costs to a goal for all new nodes are computed from the 3D Dubins heuristic. The motion node closest to the goal is expanded again until the state in the goal region is found.

Figure 5.6 and 5.7 show the performance of our search-based planner in 2D or 3D when our Dubins metric is used as the measure of an expected cost to the goal. The greedy best-first search keeps exploring the node of the current smallest cost value until the vehicle reaches one of goal states $\mathcal{G}$. Many more node expansions occur near the goal in an effort to meet the goal direction as a result of the discrete nature of the sampled motions. All of the motion primitives queued in the search tree are displayed in order to show which node is a current best estimate towards the goal at each iteration. Although the found plans are not optimal in terms of the distance, the small number of total expanded nodes proves the effectiveness of the heuristic function d_{3D} we propose. Figure 5.6 shows 3D motion plans in the presence of obstacles when the goals and the UAV are located at different altitudes.

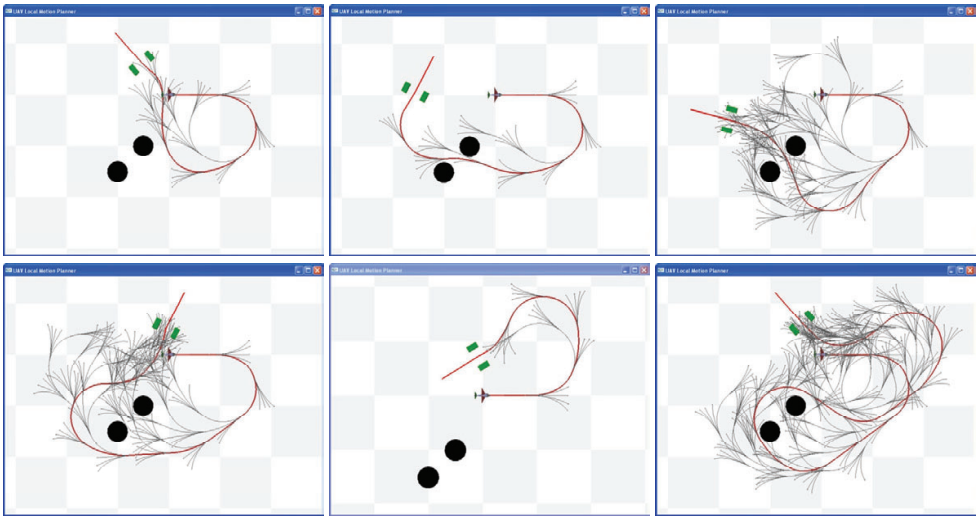


Figure 5.6: 2D motion planning examples of the greedy best-first search using only lateral motion primitives in the presence of obstacles. The goal and the UAV are located at the same altitude. All the node expansions explored in the search tree are also displayed.

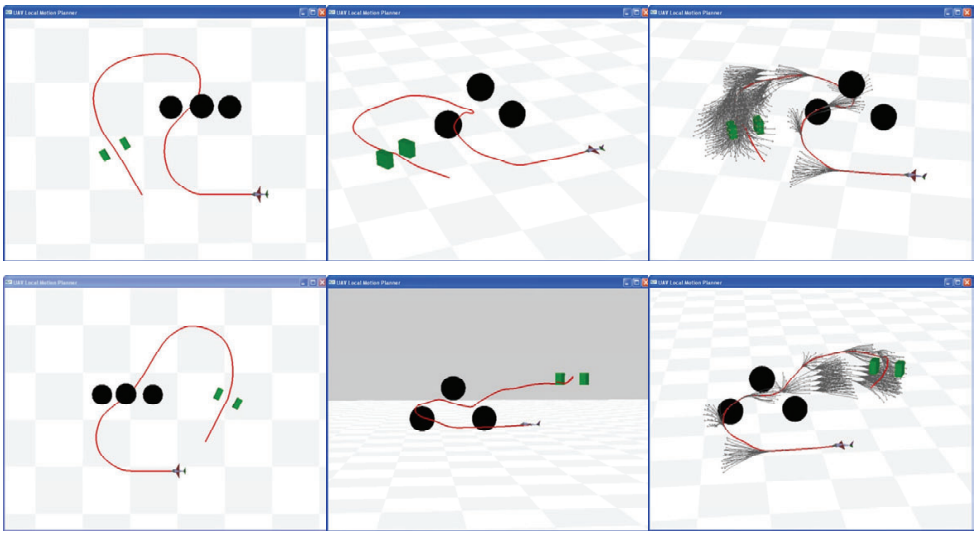


Figure 5.7: 3D motion planning examples of the greedy best-first search using both lateral and longitudinal motion primitives in the presence of obstacles. The goal and the UAV are located at different altitudes.

5.4 MDP-based Motion Planning under Motion Uncertainty

This section describes a global planning method that finds an optimal feasible path to a goal for every vehicle state in a 2D obstacle-free space. In particular, this planning method explicitly considers the uncertainty of a motion primitive due to imperfect flight control and external disturbances such as wind. The goal is therefore to find an optimal motion sequence that maximizes the probability that the vehicle will reach the target at a desired heading angle.

For a goal region located at the origin of a workspace, we formulate the planning problem as a Markov Decision Process (MDP) based on the following three factors; an uncertainty motion model using probability distributions, efficient discretization of the state space, and infinite horizon Dynamic Programming (DP) that computes a sequence of motion transitions that maximizes the probability of success. Since the DP returns an optimal action for every vehicle state in a discretized workspace, the best motion requested at any moments during navigation will be examined in a DP look-up table.

5.4.1 Probabilistic model of a motion primitive

Figure 5.8 shows a set of lateral motion primitives that we use for discrete state transitions in the MDP. A constant cruise speed V_a is maintained in flight and each motion is driven by a change in the roll angle, which is discretized between -30 to 30 degrees in a 10-degree resolution. The motion length is proportional to the amount of the corresponding roll angle change.

Let $\mathbf{q} = (x, y, \psi, \phi)$ be the vehicle state on a 2D horizontal plane. Since only lateral vehicle motions are considered here, the trim state of a fixed-wing UAV is fully characterized by the position (x, y), heading angle ψ and roll angle ϕ. A control input u that causes a transition between states corresponds to the change of a desired roll angle.

Due to an unpredictable error in the feedback control and imprecise vehicle state estimation, the next state $\mathbf{q}_{k+1}$ driven by a control input u from a current state $\mathbf{q}_k$ is *not* deterministic. Additional uncertainty in the state transitions may be caused by external disturbances such as wind gusts and atmospheric turbulence. Actual paths will not always exactly trace the simulated trajectories in Figure 5.8. Figure 5.9 shows lateral motion primitives executed and collected in real flight experiments when the same control input is applied, and reveals non-deterministic responses of state transitions. Since transient and residual errors in the roll controller (5.4) make the most significant impact on the state transition, we model the uncertainty of a vehicle motion using a probability distribution of u as follows:

$$P(\mathbf{q}_{k+1}|\mathbf{q}_k, u) = \mathbf{f}_d(\mathbf{q}_k, u), \quad u \sim \mathcal{N}(\phi_{k+1} - \phi_k,\, \sigma_\phi) \tag{5.7}$$

where $\sigma_\phi = \kappa|\phi_{k+1} - \phi_k|$ reflects the fact that the greater motion uncertainty occurs as the larger control input is given. From the feedback control dynamic system $\mathbf{f}_d$ in (5.1), the control input

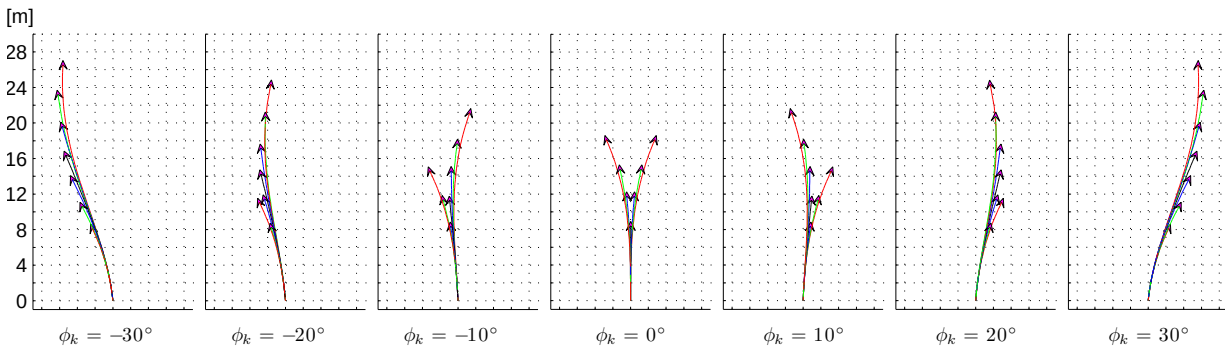


Figure 5.8: A set of lateral motion primitives used in the MDP-based motion planning: each motion primitive makes a state transition from $\mathbf{q}_k = (x, y, \psi, \phi)$ to $\mathbf{q}_{k+1}$ using the control input u in the feedback control dynamic system $\mathbf{q}_{k+1} = \mathbf{f}_d(\mathbf{q}_k, u)$. At each column, the motions start with the same ϕ_k and reach different final ϕ_{k+1} (ϕ_k and ϕ_{k+1} are one of -30, -20, -10, 0, 10, 20 and 30 degrees). The motion length is proportional to the amount of roll angle change, *i.e.*, $t = 0.3\,|\phi_{k+1} - \phi_k| + 0.6$ sec at $V_a = 10.5$ m/sec.

u is modeled as normally distributed with the mean of a roll angle change, $\Delta\phi = \phi_{k+1} - \phi_k$, and the standard deviation σ_ϕ. Figure 5.10 illustrates a probabilistic distribution of $\mathbf{q}_{k+1}$ when finite discrete samples of a normal distribution u are applied.

5.4.2 State space discretization

The discretization of a state space is based on a grid of points with an equal spacing Δ in x and y and angular bins in ψ and ϕ. For a rectangular workspace bounded by x_{max} and y_{max}, $N_s = x_{max}y_{max}/\Delta^2$ position states are aligned at the origin. The heading angle ψ is discreted by dividing $[-\pi, \pi]$ with N_ψ equally spaced bins where N_ψ is a multiple of 4 for purposes of symmetry. The roll angle ϕ is dicretized in the same way as done in the lateral motion primitivies in Figure 5.8, *i.e.*, one of -30, -20, -10, 0, 10, 20 and 30 degrees and $N_\phi = 7$.

Using this discretization, a vehicle state $\mathbf{q}$ can be approximated as a discrete state where the position and heading angle are rounded to a point closest to (x, y, ϕ) in the corresponding grid space and the index of a roll angle changes by the control input. The total number of discrete states is $N = N_s N_\psi N_\phi$.

Since the value iteration for an optimization process in the MDP requires $\mathcal{O}(kN)$ time and memory for each iteration where k is the number of discrete samples of the probabilistic state transition in (5.7), the discretization resolution Δ and N_ψ should be taken into consideration for a reasonable computation time and memory. In our implementation, we use $\Delta = 2$m and $N_\psi = 120$ for a 3-degree resolution in ψ.

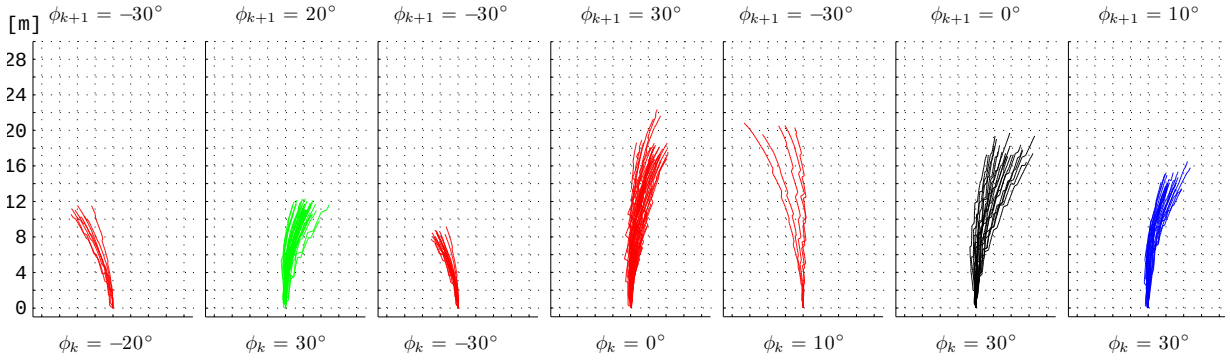


Figure 5.9: Uncertainty of a vehicle motion in the execution of lateral motion primitives in the real world. At each column, the trajectories of a motion primitive are collected from real flight experiments when the same control input u is applied to change from ϕ_k to ϕ_{k+1}. The state transition from $\mathbf{q}_k$ to $\mathbf{q}_{k+1}$ corresponding to both ends of each trajectory is not deterministic due to control error and disturbance.

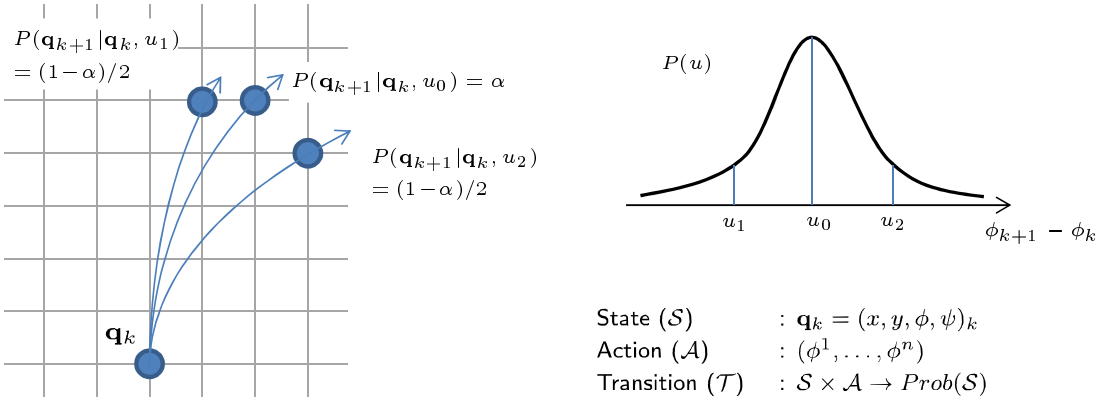


Figure 5.10: Probabilistic model of a state transition in a Markov Decision Process by the corresponding lateral motion primitive in a discretized state space.

5.4.3 Markov Decision Process

The goal of our MDP-based motion planning is to compute an optimal control u^* for every state $\mathbf{q}$ in a workspace to maximize the probability of success P_s where $P_s(\mathbf{q}) = 1$ when $\mathbf{q}$ is inside a goal region $\mathcal{G}$. Given a control u for some state $\mathbf{q}$, P_s depends on the response of the vehicle to the control and the probability of success in the subsequent state. The expected probability of success is described as $P_s(\mathbf{q}) = E[P_s(v)|\mathbf{q}, u]$ over a random variable v for the next state. For N discrete states, the motion planning problem is equal to determining an optimal control u_i for each state $i = 1, \ldots, N$ using the discrete approximation and the expansion of the expected value

to a summation as follows:

$$P_s(\mathbf{q}) = \max_u E[P_s(v)|\mathbf{q}, u] \; = \; \max_{u_i} \sum_{j=1}^{N} P_{ij}(u_i) P_s(\mathbf{q}_j) \tag{5.8}$$

where $P_{ij}(u_i)$ is the probability of entering state $\mathbf{q}_j$ given a control u_i at current state $\mathbf{q}_i$.

A Markov Decision Process (MDP) is a stochastic process on the random variables of state $\mathbf{q}$, control u and reward function R. The control at each state is purely a function of the state without explicit dependence on time and past controls. Since the process is stationary, the expected probability (5.8) becomes a form of the following Bellman equation in the MDP [84]:

$$V^*(\mathbf{q}_i) = \max_{u_i} \big[R(\mathbf{q}_i, u_i, \mathbf{q}_j) + \sum_{j=1}^{N} P_{ij}(\mathbf{q}_j|\mathbf{q}_i, u_i) V^*(\mathbf{q}_j) \big] \tag{5.9}$$

where $R(\mathbf{q}_i, u_i, \mathbf{q}_j)$ is a reward for the state transition from $\mathbf{q}_i$ to $\mathbf{q}_j$ using u_i. We set the reward function R as follows:

$$R(\mathbf{q}_i, u_i, \mathbf{q}_j) = \begin{cases} 1 & : \mathbf{q}_i \text{ in goal} \\ 0 & : \mathbf{q}_i \text{ out of workspace} \\ -0.001 - \alpha_1 |\phi_j - \phi_i| - \alpha_2 |\phi_i| & : \text{Otherwise} \end{cases} \tag{5.10}$$

where α_1 and α_2 are user-defined constants. The reward function is designed to return the higher value as the smaller changes in the roll angle or the motions closer to a level flight ($\phi = 0$) are made. The optimal policy π^* at the state $\mathbf{q}_i$ is given as

$$\pi^*(\mathbf{q}_i) = \operatorname*{argmax}_{u_i} \sum_{j=1}^{N} \big[R(\mathbf{q}_i, u_i, \mathbf{q}_j) + P_{ij}(\mathbf{q}_j|\mathbf{q}_i, u_i) V^*(\mathbf{q}_j) \big] \tag{5.11}$$

We solve this dynamic programming problem defined by the Bellman equation (5.9) by the value iteration, which iteratively updates P_s for each state by evaluating (5.8). This generates a DP look-up table containing the optimal control u_i and the probability of success P_s for $i = 1, \ldots, N$. To improve iteration speed, the sparsity of the matrices $P_{ij}(u) \in \mathcal{R}^{N \times N}$ is exploited since each row of P_{ij} has only k nonzero entries where $k \ll N$ due to the spatial vicinity of a state transition. Hence, only accessing nonzero entries of $P_{ij}(u)$ during computation requires only $\mathcal{O}(kN)$ rather than $\mathcal{O}(N^2)$ time and memory at each iteration [85]. The value iteration terminates when the maximum value change over all states is less than a threshold (we use 10^{-4}).

Figure 5.11 shows several examples of an MDP optimal path using the lateral motions in Figure 5.8 and the probabilistic model in Figure 5.10. Each plot shows three paths that start at the same position and heading angle but different roll angles (0, 30, -30 degrees, respectively).

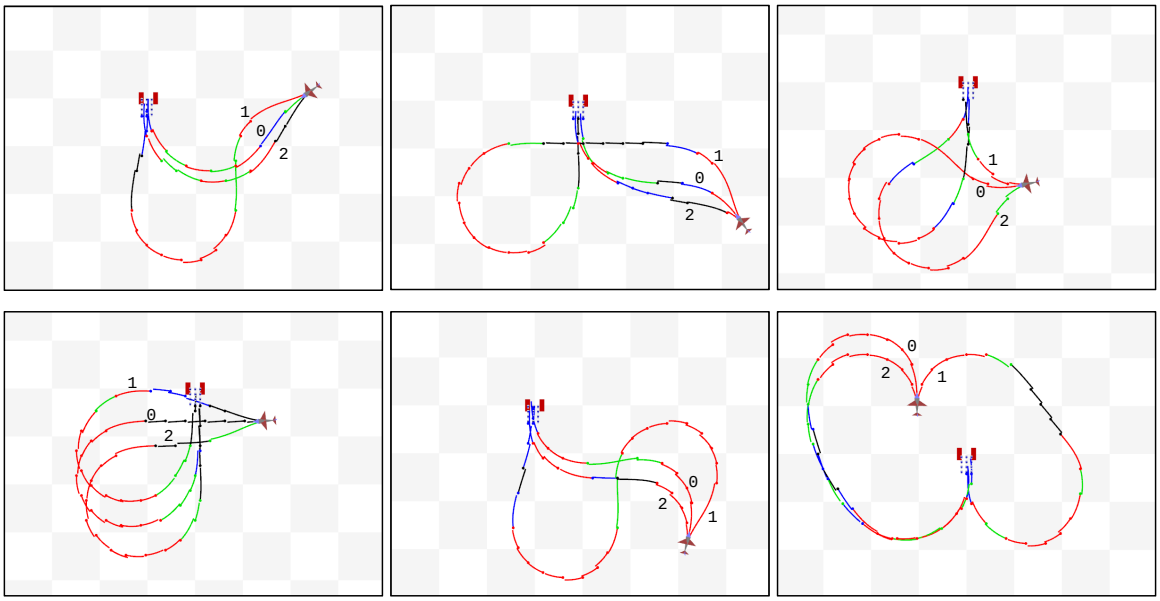


Figure 5.11: MDP-based optimal motion plans in an obstacle-free 2D space: each plot shows MDP paths starting at three different roll angles ($\phi_0 = 0°$, $\phi_1 = 30°$ and $\phi_2 = -30°$). The color of a path segment in the plan represents the magnitude of a final roll angle of the corresponding lateral motion primitive (the red is $|\phi_{k+1}| = 30°$, green is $|\phi_{k+1}| = 20°$, blue is $|\phi_{k+1}| = 10°$, and black is $|\phi_{k+1}| = 0°$).

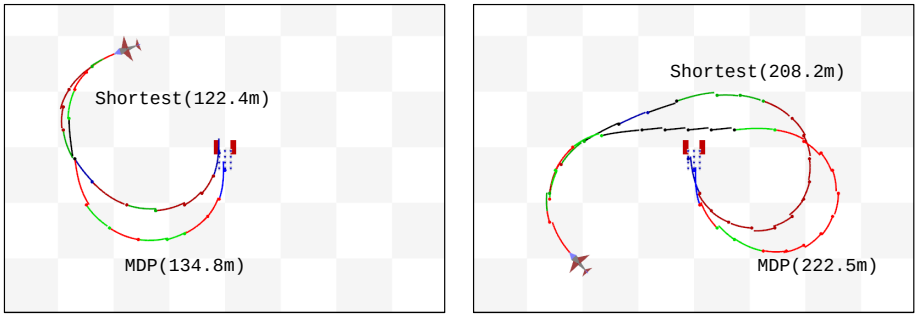


Figure 5.12: Comparison between the stochastic MDP optimal path maximizing the probability of success P_s and the deterministic shortest path ignoring motion uncertainty. The MDP path is slightly longer than the shortest path but more probable to enter the gate.

Due to the vehicle dynamics, distinctive routes to the goal are generated depending on the starting roll angle.

We set a goal region $\mathcal{G}$ to a set of states satisfying $-10 \leq x \leq 0$m, $|y| \leq 3$m, $|\psi| \leq 8°$, and $|\phi| \leq 10°$ for a gate located at the workspace center and oriented toward the positive x-axis. The number of discrete state transitions for each motion primitive is set to $N_p = 3$ such that three control inputs u_0, u_1, and u_2 are sampled at 0, $-1.0\sigma_\phi$ and $1.0\,\sigma_\phi$, respectively when $\kappa = 0.1$ in (5.7). Their probabilities are computed by integrating the corresponding area under the normal curve. We set the workspace by $x_{max} = y_{max} = 100$m and used discretization parameters $\Delta = 2$m and $N_\psi = 120$. The total number of resulting discrete states is $N = 2,100,000$. The DP problem was implemented in C++ and tested on a 2.17GHz Intel PC. The computation time for each value iteration was 55.3 sec and 50 iterations were needed until the maximum value change was less than 10^{-4}. Overall, the total time to compute DP lookup table was 46.1 min.

Figure 5.12 compares an optimal path maximizing the probability of success P_s with the shortest path ignoring motion uncertainty. A deterministic shortest path was computed by replacing the reward function R in (5.10) with the motion length and using a single state transition at the mean of the normal distribution. The MDP path is slightly longer than the corresponding shortest path but is more probable to enter the gate because less aggressive turns are involved near the gate.

5.5 Target visibility constraint

Suppose that a camera in a fixed-wing UAV has a limited field-of-view lens and is aligned downward to detect a target on the ground. The camera's viewing area for a surrounding environment is bounded and subject to the airplane's pose. If a motion plan consists of tight turns in high roll angles near the target, the target is less likely to be exposed to the camera before the vehicle passes through the target. It is not possible to achieve seamless target observation unless specialized units such as a pan-and-tilt gimbal system or omnidirectional lens are additionally attached to the camera.

If this limited target visibility is ignored in motion planning when the confidence on a target state is low, the vehicle is more likely to miss a target because it does not have a sufficient time to observe the target and respond to a refined target pose produced from additional target observation.

5.5.1 Minimum target observation time

To guarantee a continuous target-in-view approach with an increased confidence on the target, we define a visibility goal region $\mathcal{G}_v$ that satisfies $t_{v,min} \leq t_v \leq t_{v,max}$ for a non-discontinuous

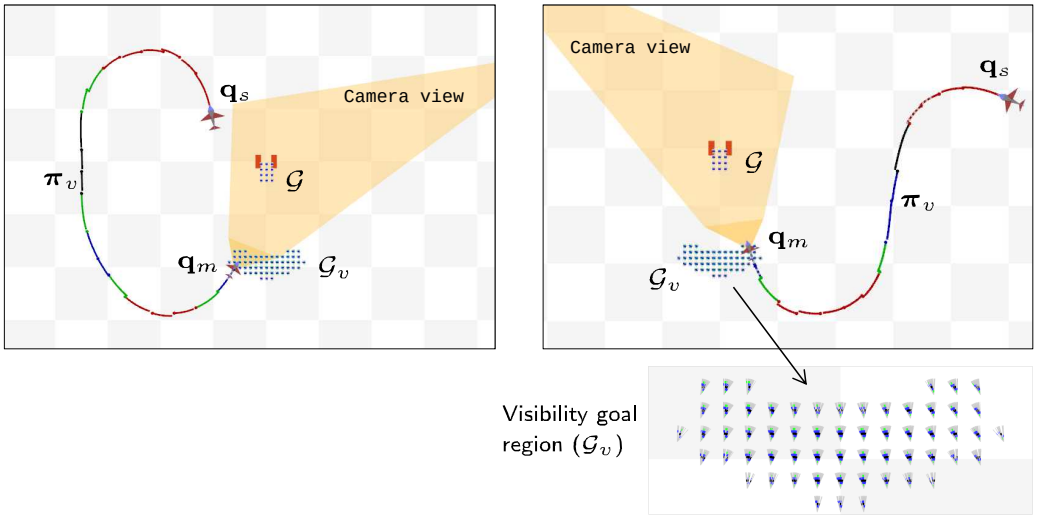


Figure 5.13: MDP motion plans $\boldsymbol{\pi}_v$ to the visibility goal region $\mathcal{G}_v$ that is used as an intermediate goal to ensure a sufficient target observation time t_v. Every state in $\mathcal{G}_v$ satisfies $2.5 \leq t_v(\boldsymbol{\pi}_g(\mathbf{q}_i, \mathcal{G})) \leq 3.2\,\text{sec}$ for the original goal region $\mathcal{G}$ when $V_a = 10.5\,\text{m/sec}$ and $\mathcal{G}$ is located around 40m ahead of the target gate.

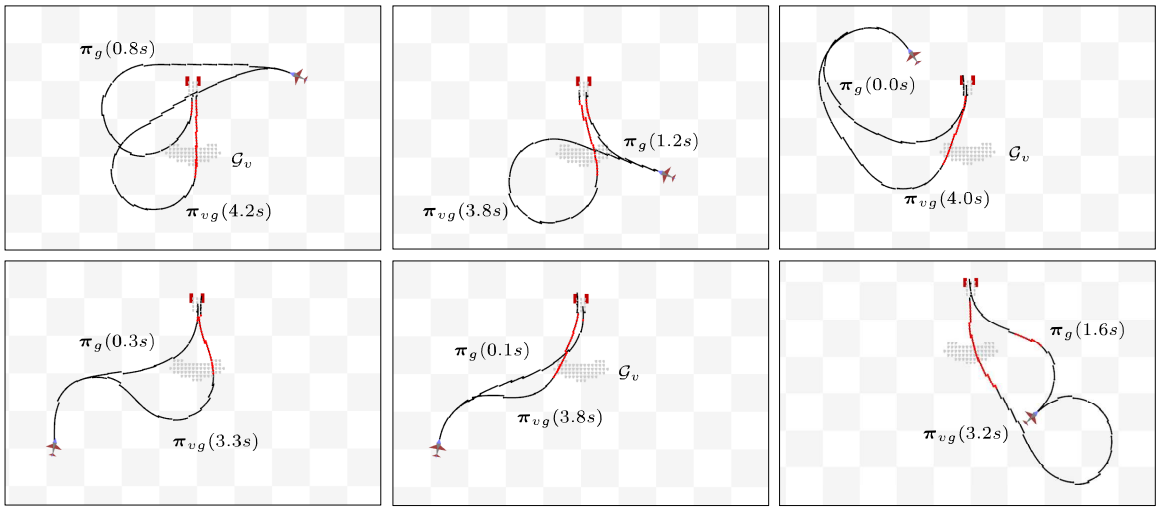


Figure 5.14: Comparison between MDP motion plans when the target visibility constraint is ignored ($\boldsymbol{\pi}_g$) or considered ($\boldsymbol{\pi}_{vg}$). The red segment on each path indicates the states that the target is visible and the number is the target observation time t_v. The plan $\boldsymbol{\pi}_{vg}$ that arrives at the goal by way of $\mathcal{G}_v$ is longer than $\boldsymbol{\pi}_g$ but guarantees the minimum t_v. The target visibility is computed when the vehicle's altitude is 18m and the camera's field of view and downward angle are 70 and 40 degrees, respectively.

target observation time t_v until goal completion. Given a look-up table of the MDP plan $\boldsymbol{\pi}_g$ in Section 5.4.3 which is computed for an original goal region $\mathcal{G}$ around the target object g, we can find the following visibility goal region $\mathcal{G}_v$ by investigating $\boldsymbol{\pi}_g$ for every state:

$$\mathcal{G}_v = \{\mathbf{q}_i \,|\, t_{v,min} \leq t_v(\boldsymbol{\pi}_g(\mathbf{q}_i, \mathcal{G})) \leq t_{v,max},\, V(\mathbf{q}_k, g) = true \text{ for all } \mathbf{q}_k \text{ in } \boldsymbol{\pi}_g(\mathbf{q}_i, \mathcal{G})\} \tag{5.12}$$

where $V(\mathbf{q}_k, g)$ denotes the visibility of the target g at the vehicle state $\mathbf{q}_k$. The path $\boldsymbol{\pi}_g$ starting at any state in $\mathcal{G}_v$ always sees the target for the time longer than $t_{v,min}$.

Providing $\mathcal{G}_v$ as a set of intermediate goals, we design a two-step motion planning scheme in order to impose a target observation time t_v on a resulting motion plan. Firstly, a new MDP optimal plan $\boldsymbol{\pi}_v$ is computed for $\mathcal{G}_v$ using the same MDP framework but the goal region. Then, two individual paths ($\boldsymbol{\pi}_v$ and $\boldsymbol{\pi}_g$) are combined into a final motion plan $\boldsymbol{\pi}_{vg}$ via an intermediate goal $\mathbf{q}_m$ in $\mathcal{G}_v$. In other words, $\boldsymbol{\pi}_{vg}(\mathbf{q}_s, \mathcal{G}) = \boldsymbol{\pi}_v(\mathbf{q}_s,\, \mathbf{q}_m \in \mathcal{G}_v) + \boldsymbol{\pi}_g(\mathbf{q}_m,\, \mathcal{G})$ where $\mathbf{q}_s$ is a starting state and $\mathbf{q}_m$ is a final state of $\boldsymbol{\pi}_v$.

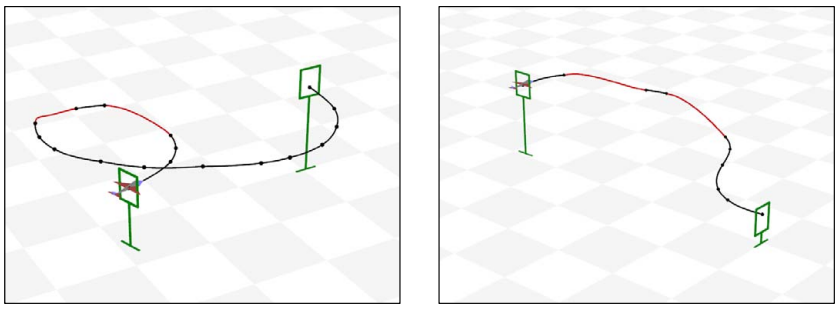
Figure 5.13 shows the visibility goal region $\mathcal{G}_v$ which guarantees the minimum target observation time $t_{v,min} = 2.5$ sec at the cruise speed $V_a = 10.5$ m/sec, and is located around 40m ahead of the goal. In Figure 5.14, we compare the path produced when the visibility constraint is either ignored ($\boldsymbol{\pi}_g$) or considered ($\boldsymbol{\pi}_{vg}$). The path length of $\boldsymbol{\pi}_{vg}$ is typically longer than that of $\boldsymbol{\pi}_g$ in order to provide a minimum target observation time in $\boldsymbol{\pi}_{vg}$.

5.6 Motion Plans in 3D

The MDP framework introduced in the previous sections can be extended to 3D by taking longitudinal motions into account. The state dimension increases by one since the state becomes $\mathbf{q} = (x, y, z, \phi, \psi)$ in 3D. In addition to the lateral motions in Figure 5.8, we use the longitudinal motion primitives presented in Figure 5.3 whose time duration (t_f) of a pitch reference profile is fixed to 2.4 sec for altitude change.

Figure 5.15 shows two examples of 3D optimal motion plans generated by the MDP-based motion planning. The state space is discretized by $\Delta_{x,y} = 2m$, $\Delta_z = 3m$, $\Delta_\phi = 10^\circ$ and $\Delta_\psi = 5^\circ$ in a relatively small 3D workspace of $60 \times 60 \times 24$ meters. The number of discrete states is $N = 18,144,000$ and the total computation time for a DP lookup table was about 12 hours.

Note that the MDP demands significant computation time and memory space when the number of states is large. Thus, in previous sections, we limited the MDP-based planning to lateral maneuvers on a horizontal plane and the altitude was controlled separately. Nonetheless, a deliberate design of state-space discretization (*e.g.*, multi-resolution approach) or an appropriate choice of a workspace range will help reduce the number of states and accelerate the MDP computation for a larger 3D workspace size.



(a) (b)

Figure 5.15: MDP-based optimal motion plans in a 3D space: Contrary to Figure 5.11, the gates are placed at different altitudes from the vehicle's current altitude ($\Delta Z = 10m$ in (a) and $\Delta Z = -15m$ in (b)). The red segments on each motion plan correspond to longitudinal motion primitives which cause altitude changes.

5.7 Entering a gate

When the vehicle is close to a target with an appropriate heading angle, the target completion task requires a fine-level control that can lead the vehicle to enter a gate successfully. Since the resolution of the motion primitives in Section 5.2 are not high enough for control at the desired precision level, the navigation is switched to another algorithm near the gate.

Trajectory tracking requires the vehicle to be in a particular location at a particular time and thus may result in significant problems for small UAVs unless a tracking controller properly accounts for disturbances. Hence, we instead use a straight path following control proposed by Nelson *et al.* [86], which guides the vehicle to remain on the path without the notion of time and is thus proven to be robust to wind disturbances.

5.7.1 Vector field along a straight line

Without the loss of generality, suppose that a gate is located at the origin and directed at the y-axis as shown in Figure 5.16. The goal is to follow the straight line path aligned to the gate orientation and to lead the vehicle to a successful gate passing. This path following method is based on the construction of a vector field surrounding the path to be followed. The vectors of this field provide heading commands to guide the vehicle toward the desired path.

Let x be the lateral distance of the vehicle from the path and let ϕ_d be the desired heading of the vehicle. The objective is to construct the vector field so that the vehicle is directed perpendicularly to the path ($\phi_d = \pi/2$) when x is large, and the vehicle is directed along the

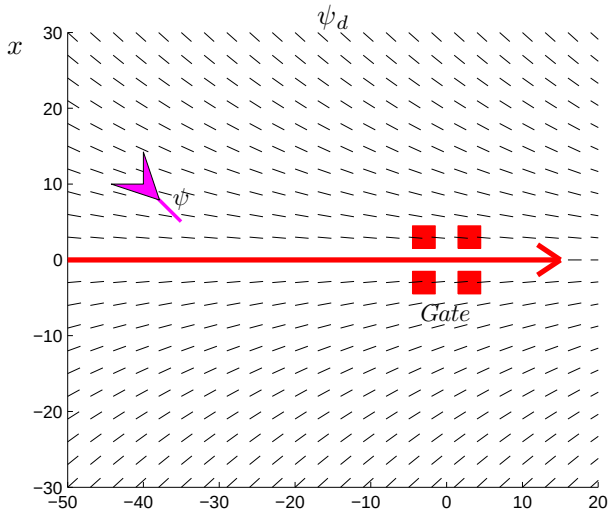


Figure 5.16: Entering a gate: vector field for a desired heading angle ψ_d is generated along the straight line aligned to a gate when the vehicle is sufficiently close to the gate with a proper heading angle ($k = 0.02$ in $\psi_d = -\tan^{-1}(kx)$).

path ($\phi_d = \pi/2$) as x approaches zero as follows:

$$\psi_d = -\tan^{-1}(kx) \tag{5.13}$$

$$\phi_d = -K_p(\psi_d - \psi) \tag{5.14}$$

where k is a positive constant that handles the rate of the transition from $\pi/2$ to zero. A small value of k produces a long and smooth transition in the desired heading. In our autopilot, a heading-hold loop for ψ_d is implemented by the control of the roll angle to follow ϕ_d which is proportional to the heading error in (5.14).

Chapter 6

Air-Slalom Experiment

In this chapter, we will describe the UAV system that has been developed for simulation and experiments, along with additional algorithms that are needed for autonomous navigation, such as target detection and vehicle and target state estimation. Experiment results on the air-slalom task will be presented when a single gate or multiple gates are arranged on the ground.

6.1 UAV system

Our UAV system development is targeted for a small-size and low-cost platform that is deployable and expendable with a high budget-performance ratio. All of the components employed in the system are commodity-level products and readily available in market. It is expected that our system architecture and hardware would be easily transferrable to other airplane platforms.

The UAV system consists of the following two main parts linked by wireless serial communication: (1) an airplane system which is a fixed-wing model airplane equipped with onboard sensors and processors for flight control and vision processing, and (2) a ground station which not only monitors flight status but also provides sufficient computation capabilities for state estimation and motion planning.

6.1.1 Hardware and Software Architecture

Figure 6.1 depicts the architecture of the UAV system hardware and communication system. The fixed-wing airplane is based on a conventional RC model that has a 1.2m wingspan, three control inputs (thrust, rudder and elevation), and is typically sold for hobbyists at the beginner level. The maximum payload is 250g and is almost the same as the total weight of all components including onboard sensors, autopilot, onboard computer and batteries. The maximum flight time is around 10 minutes due to limited battery capacity and high payload. The autopilot combines

Table 6.1: Specifications of low-cost commercial off-the-shelf components in the UAV system

Component	Manufacturer & Model	Specifications
Airplane	Hobbyzone SuperCub	3-channel (thrust, rudder, elevation), 1.2m wingspan
Battery #1	Cellpro CP0950-3S	Lithium polymer, 3-cell, 11.1V, 900mAh
Battery #2	Cellpro CP0300-2S	Lithium polymer, 2-cell, 7.4V, 300mAh
Barometer	BMP085	0.03 hPa accuracy, MEMS
IMU	Analog Devices	±2g, ±200°/sec, tri-axial, MEMS
GPS	U-Blox LEA-5H	4Hz position update, helix antenna
Camera	FireFly MV	captured in 320 × 240 at 15FPS, 70° FOV lens
Autopilot	Ardupilot Mega 1280	16MHz, 8 11-bit A/D, 4 serial, 3 PWMs
Onboard computer	Gumstix Overo Fire	OMAP3530, 720MHz, 512M mem, 8GB flash, Summit
Communication	XBee PRO 900	Serial comm at 38.4kbps, 900 MHz
Ground station	Alienware M15X	1.73GHz, Quad core

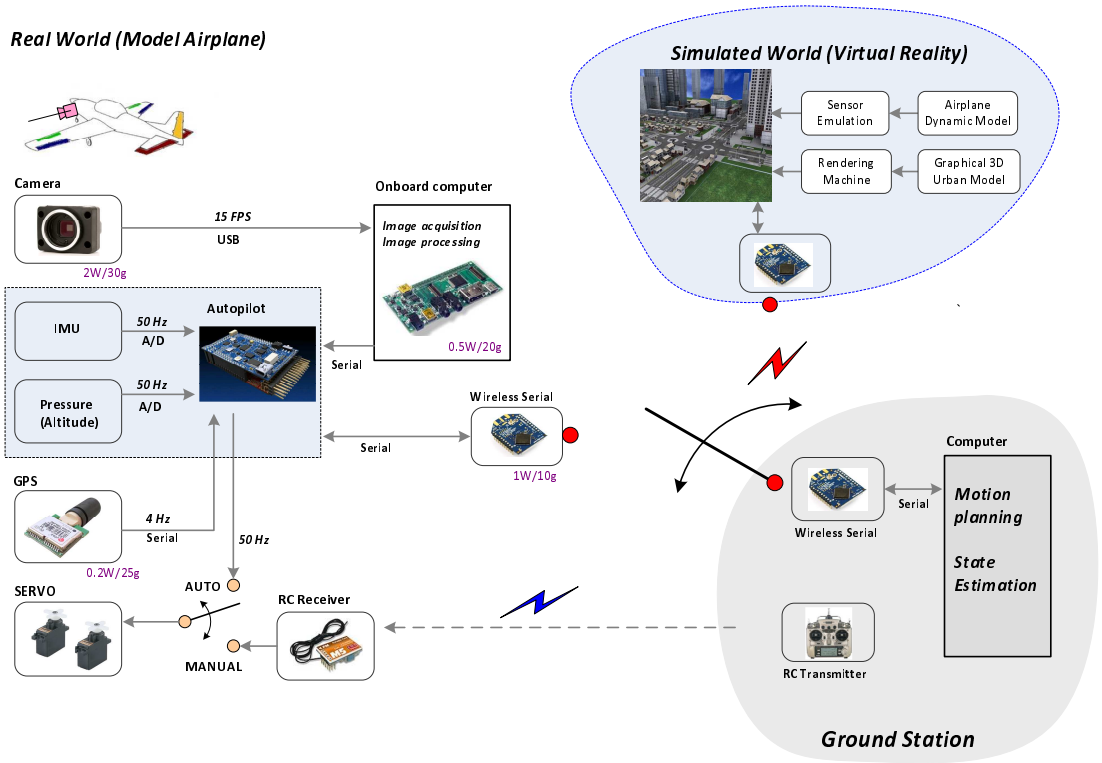


Figure 6.1: The architecture of UAV hardware and software systems. Wireless communication to the ground station can be switched between the real airplane and the simulator.

Figure 6.2: UAV testbed platform and onboard components

GPS, strap-down MEMS IMU, and barometer sensors into PID feedback flight controls. The detail specifications of the airplane platform and onboard sensors are listed in Table 6.1.

The execution of navigation algorithms is distributed over on-body and off-body computation. Since our wireless communication throughput is not sufficient for realtime image delivery, vision processing runs on the onboard computer and its core output is then sent to the ground station in order to be combined with other sensor data for vehicle and target state estimation.

There exist two main switches in the system architecture. One is for the data link connected to the ground station and switched between the real world and the simulated world. The navigation algorithms at the ground station are completely isolated from where the switch is connected. The other is for the airplane control which selects an autonomous mode or manual mode for emergency interception.

6.1.2 UAV Simulator

A graphical UAV simulation environment has been developed in order to evaluate the performance in many different operational scenarios with ground truth. The dynamic model in the simulator was developed at the MAGICC lab at BYU [81] and represents a fixed-wing that has a 1-5 foot wing span and is powered by an electric motor attached to a fixed-pitch propeller. Some model parameters were hand-tuned for our airplane after flight behaviors in outdoor flight were compared with the simulated model. Several 3D graphic models of urban scenery are integrated into the simulation environment for camera emulation. Irrlicht Engine is used for realtime rendering and easy loading of commercially available graphic file formats.

In the simulator, all sensor emulations and vehicle dynamics updates are based on the simulated clock. Since the simulator runs on a computer completely separated from the ground station in Figure 6.1, the clock of the simulated world runs consistently so that periodic sensor emulation take places. Otherwise, unpredictable time delays due to additional heavy computation may occur, which will corrupt the timing of the simulated sensors.

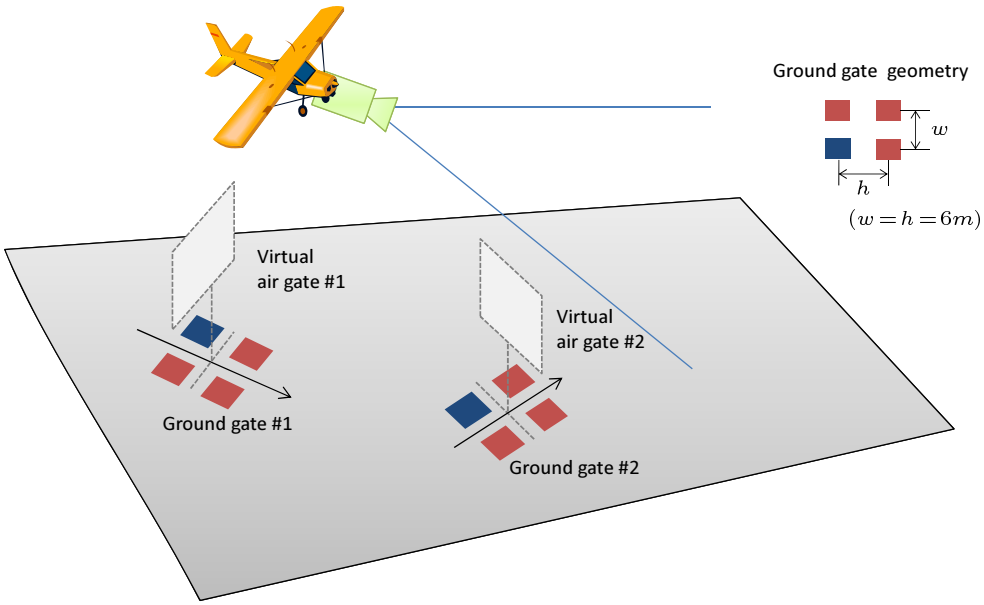


Figure 6.3: The experimental setup for the air slalom task using virtual gates in the air, each of which is conceivable from a set of colored square objects on the ground.

6.2 Virtual gate and target detection

Our experimental setup for the air slalom task is illustrated in Figure 6.3. Since it is difficult to install real gates in the air maintained at a certain altitude, we instead construct virtual gates that can be easily conceivable from real ground targets. In the experiment, two virtual gates are implemented by placing visual targets on the ground.

Virtual gate:

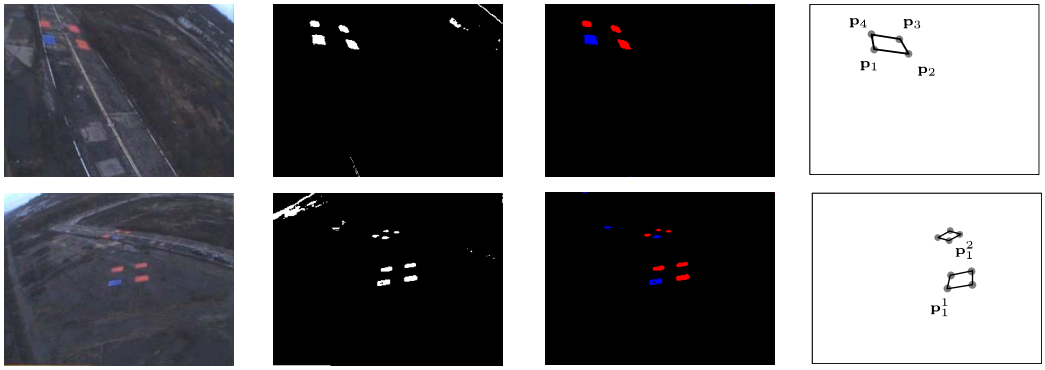
Each virtual gate in Figure 6.3 is constructed using four square-shaped objects placed on the ground. These ground objects are identical in size and consist of one blue square and three red squares. The geometric shape between the four squares is arranged as a rectangle and the distance between the square centers is set to 6m ($w = h = 6m$). The position of the virtual gate is equal to the rectangle center and its zero orientation corresponds to when the blue square is at a bottom-left corner.

Gate detection and association:

The ground target objects are distinctly colored so that simple color matching can easily detect them in images. Firstly, we prescribe color regions for the red and blue targets, respectively, in the HSV color space. The thresholding on each channel can produce a binary image for matched color areas. Secondly, a list of connected regions is extracted from this binary image by an optimal

Algorithm 6.1: Gate detection and association

$\{\mathbf{p}_1^k, \mathbf{p}_2^k, \mathbf{p}_3^k, \mathbf{p}_4^k\}$ = Gate_Detection_And_Association($\mathbf{I}$, $\mathbf{x}_{cam}$)
Set bounded regions R_{hsv} for red and blue targets and A_{min}
$\mathbf{I}_{HSV}$ = convert_color_space($\mathbf{I}_{RGB}$)
$\mathbf{I}_{BIN}$ = thresholding($\mathbf{I}_{HSV}$, R_{HSV})
$\mathbf{I}_{LABEL}$ = two_pass_connected_component_labeling($\mathbf{I}_{BIN}$)
$(\mathbf{p}_b^1, \ldots, \mathbf{p}_b^p, \mathbf{p}_r^1, \ldots, \mathbf{p}_r^q)$ = mass_center($\mathbf{I}_{LABEL}$, A_{min})
foreach *a blue point* $\mathbf{p}_b^i$ **do**
  foreach *gates in* $\mathbf{x}_g^k$ **do**
    foreach *three red points* $(\mathbf{p}_r^1, \mathbf{p}_r^2, \mathbf{p}_r^3)$ **do**
      $\mathbf{r}^j = (\mathbf{p}_1^j, \mathbf{p}_2^j, \mathbf{p}_3^j, \mathbf{p}_4^j)$ = set_anti_clockwise_order($\mathbf{p}_b^i, \mathbf{p}_r^1, \mathbf{p}_r^2, \mathbf{p}_r^3$)
      $\widehat{\mathbf{x}}_{cam}$ = camera_pose_homography($\mathbf{r}^j$, $\mathbf{x}_g^k$, w, h)
      err$(k,j) = \|\mathbf{x}_{cam}^k - \widehat{\mathbf{x}}_{cam}\|$
  $(k^*, j^*) = \text{argmin}_{k,j}(\text{err}(k,j))$



(a) Captured image (b) Thresholding in HSV space (c) Connected blobs (d) Rectangular gate shape

Figure 6.4: A single or multiple-gate detection by color matching and blobing

two-pass connected-component labeling algorithm (Wu *et al.* [87]), which is known as one of the fastest labeling methods. Finally, for the blobs larger than a minimum area A_{min}, mass centers of blue or red blobs $(\mathbf{p}_b^1, \ldots, \mathbf{p}_b^p, \mathbf{p}_r^1, \ldots, \mathbf{p}_r^q)$ become candidates for rectangular corner points of each ground gate.

Suppose that n gates are already initialized and their current pose estimates are $(\mathbf{x}_g^1, \ldots, \mathbf{x}_g^n)$. To correctly associate each mass center with a corresponding rectangular corner in the presence of outliers, we test all possible rectangles that can be drawn by $\{\mathbf{p}_b^i\}$ and $\{\mathbf{p}_r^i\}$. Each rectangle candidate consists of one blue and three red points in an anti-clockwise order on an image. From a known rectangular shape (w, h) and $\mathbf{x}_g^k$, the hypothesis error for the k-th gate is the difference

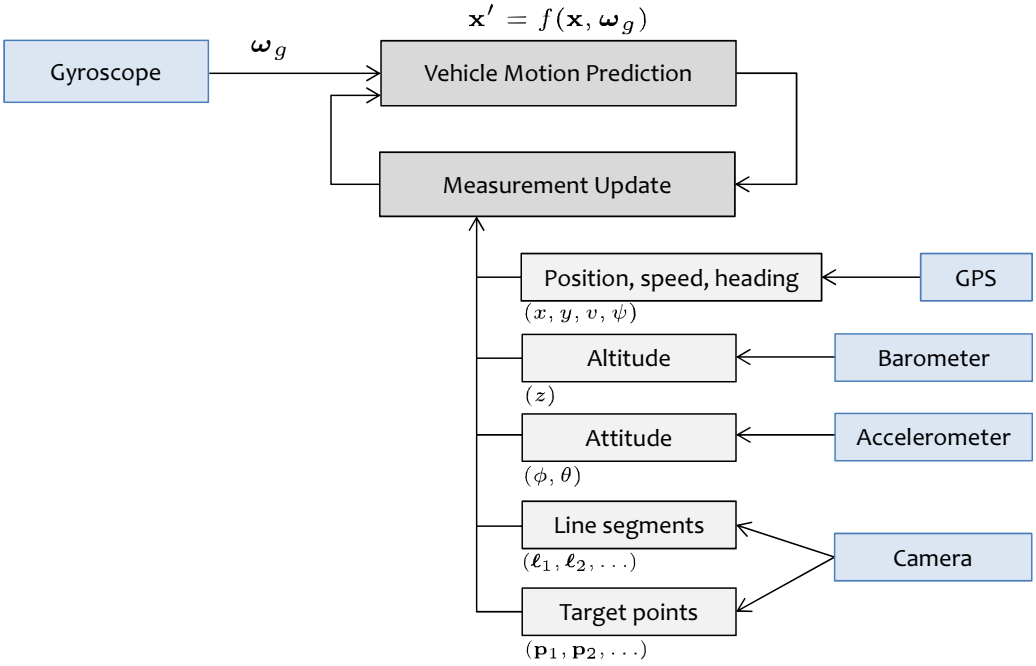


Figure 6.5: The prediction and update step of an Unscented Extended Kalman filter for vehicle and target state estimation based on GPS, IMU, barometer and camera.

between a current camera pose $\mathbf{x}_{cam}^k$ and the pose $\widehat{\mathbf{x}}_{cam}$ estimated by the planar homography, which is computed by 2D-to-2D point correspondences between the four corners on the image plane and those on the ground plane [53]. For each blue point $\mathbf{p}_b^i$, a minimum error reveals which gate is associated with the best hypothesis $(\mathbf{p}_1^k, \mathbf{p}_2^k, \mathbf{p}_3^k, \mathbf{p}_4^k)$ where $\mathbf{p}_1^k = \mathbf{p}_b^i$. Next, this result proceeds to the measurement update in the target state estimation described in the following section. See Algorithm 6.1 and Figure 6.4 for the more detailed procedure.

6.3 Vehicle and target state estimation

The estimation of vehicle and target states for the air-slalom task is based on the sensor fusion of low-grade and lightweight onboard sensors such as GPS, IMU, barometer and camera. We use an Unscented Kalman filter (UKF) [88] to combine a predicted vehicle motion and the sensor measurements in nonlinear equations. For a Gaussian state distribution, the posterior estimate produced by the UKF is accurate to the third order. One main benefit of the UKF is that it is derivative-free and thus the filter does not require Jacobian matrices to be computed at each step. Although the computational load of the UKF increases compared with other Kalman filters, it is amenable on a general desktop computer.

The state estimate $\mathbf{x}$ in the UKF consists of nine states for the vehicle and three states for

each target gate, *i.e.*, $\mathbf{x} = (\mathbf{x}_v, \mathbf{x}_g) = (x,\, y,\, z,\ \ \phi,\, \theta,\, \psi,\ \ v,\, \alpha,\, \beta,\ \ x_g^1,\, y_g^1,\, \psi_g^1, \ldots, x_g^n,\, y_g^n,\, \psi_g^n)$. The vehicle state is denoted by the position $\mathbf{p}_v = (x, y, z)$ and orientation $\mathbf{o}_v = (\phi,\, \theta,\, \psi)$, vehicle speed v, angle of attack α and side slip angle β. The target state $\mathbf{x}_g^i$ for the i-th gate is denoted by the position (x_g^i, y_g^i) and orientation ϕ_g^i on the ground plane. The state space dimension is $(9 + 3n)$ when n is the number of target gates. The UKF in Figure 6.5 alternates the following two distinct phases, *i.e.*, vehicle motion prediction and measurement update:

Vehicle motion prediction:

The prediction phase uses the state estimate from the previous timestep to produce an estimate of the state at the current timestep. The following continuous-time prediction equations in the UKF represent a non-accelerating and non-acrobatic maneuver of a fixed-wing airplane. Two angles (α and β) account for the discrepancy between a true vehicle moving direction and the fuselage orientation which are caused by lift force and wind disturbance. A strap-down gyroscope output $\boldsymbol{\omega}_g$ is the only external input in the prediction step. All the ground gates are assumed to be stationary.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}' = \mathbf{R}(\phi, \theta + \alpha, \psi + \beta) \begin{bmatrix} v \\ 0 \\ 0 \end{bmatrix} \ + \ \mathbf{w}_p \tag{6.1}$$

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}' = \boldsymbol{\Omega}(\phi, \theta)\boldsymbol{\omega}_g \ + \ \mathbf{w}_o \tag{6.2}$$

$$\begin{bmatrix} v \\ \alpha \\ \beta \end{bmatrix}' = \mathbf{0} + \mathbf{w}_v \tag{6.3}$$

$$\begin{bmatrix} x_g^i \\ y_g^i \\ \psi_g^i \end{bmatrix}' = \mathbf{0} + \mathbf{w}_g, \quad \text{for } i = 1, \ldots, n \tag{6.4}$$

where

$$\mathbf{R}(\phi, \theta, \psi) = \begin{bmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}, \ \boldsymbol{\Omega}(\phi, \theta) = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} \tag{6.5}$$

and $\mathbf{w}_p$ $\mathbf{w}_o$ $\mathbf{w}_v$ and $\mathbf{w}_g$ are Gaussian process noises.

Measurement update:

In the update phase, the current *a priori* prediction of the state $\mathbf{x}$ is combined with current measurements of the onboard sensors listed in Table 6.1 in order to refine $\mathbf{x}$ to the *a posteriori* state estimate. The following update equations are used to compute the predicted measurement from the predicted state $\widehat{\mathbf{x}}$ in the UKF.

- GPS (position, speed, heading):
  An up-to-date GPS receiver module provides many features of navigation states from satellites, which commonly includes position, velocity, precision timing and level of data accuracy. Among available GPS data, we select the measurements for a 2D position, ground speed V_{ground}, and ground course from GPS velocity V_{gps}.

$$\mathbf{h}_{gps}(x, y, \psi, \beta, v) = \begin{pmatrix} x \\ y \\ \psi + \beta \\ v \end{pmatrix} = \begin{pmatrix} x_{gps} + x_0 \\ y_{gps} + y_0 \\ \text{atan2}(V_{y,gps}, V_{z,gps}) \\ V_{ground} \end{pmatrix} \tag{6.6}$$

  Since the U-Blox GPS module performs a different satellite signal processing depending on an internal operating mode, it is important to set an appropriate mode for aerial 3D motions. The update rate of the GPS is 4Hz.

- Barometer (altitude):
  The digital pressure sensor reports the pressure and temperature in steps of 0.01hPa and 0.1°C. Once a true pressure p is computed after temperature compensation, the following barometer formula describes how the absolute pressure of the air changes with altitude z.

$$\mathbf{h}_{baro}(z) = p = p_s \left(1 - \frac{z + z_0}{44330}\right)^{5.255} \tag{6.7}$$

  where p_s is the pressure at sea level (*e.g.*, 1013.25hPa) and the altitude change $\Delta z = 1m$ corresponds to $\Delta p = 0.12$hPa at $p = p_s$. The update rate of the barometer is 50Hz.

- Accelerometer (attitude):
  An accelerometer measurement $\mathbf{a}_m$ is the sum of gravity $\mathbf{a}_g$ and aerodynamic forces $\mathbf{a}_{dyn}$ like lift, thrust and drag. Since it is almost impossible to eliminate $\mathbf{a}_{dyn}$ completely from $\mathbf{a}_m$, we estimate $\mathbf{a}_g$ by removing the centrifugal acceleration $\mathbf{a}_{cen}$, which is a main factor in $\mathbf{a}_{dyn}$ during banked turns.

$$\mathbf{a}_g = \mathbf{a}_m - \mathbf{a}_{dyn} \tag{6.8}$$

$$\widehat{\mathbf{a}}_g = \mathbf{a}_m - \mathbf{a}_{cen} \approx \mathbf{a}_m - \boldsymbol{\omega}_{gyro} \times [\, v, 0, 0 \,]^\top \tag{6.9}$$

An estimated gravity $\widehat{\mathbf{a}}_g = (a_x, a_y, a_z)$ provides the vehicle attitude as follows.

$$\mathbf{h}_{acc}(\phi, \theta) = \begin{pmatrix} \phi \\ \theta \end{pmatrix} = \begin{pmatrix} \text{atan2}(a_y, a_z) \\ \text{atan2}(-a_x, \sqrt{a_y^2 + a_z^2}\ ) \end{pmatrix} \tag{6.10}$$

Similar approaches to cancel $\mathbf{a}_{dyn}$ out can be found in the nonlinear complementary filter [4] and direction cosine matrix [5]. The update rate of the IMU is 50Hz.

- Camera (line segments):
  The update equation (3.13) for a gravity-related line segment measurement $\boldsymbol{\ell}_i$ is already presented in the visual/inertial attitude estimation, *i.e.*,

$$\mathbf{h}_{line}(\phi, \theta) = \tan^{-1}\left(\frac{\widehat{v}_y^* - m_y}{\widehat{v}_x^* - m_x}\right) \tag{6.11}$$

  where $\mathbf{m}$ is a line midpoint and $\mathbf{v}^*$ is a predicted vanishing point of $\boldsymbol{\ell}_i$. The prerequisite line classification for vertical and horizontal edges is described in Chapter 3.4.

- Camera (target points):
  As shown in Figure 6.4(d), the measurement of the i-th target gate is composed of four corners $(\mathbf{p}_1^i, \mathbf{p}_2^i, \mathbf{p}_3^i, \mathbf{p}_4^i)$ of a rectangle in a normalized camera coordinate. The blue square corresponds to $\mathbf{p}_1^i$ and the other red squares are listed in an anti-clockwise direction. The following update equation is the perspective projection of these rectangular corners in $\mathbf{P}_r$ which are located on the x-y world plane.

$$\mathbf{h}_{target}(\mathbf{p}_v, \mathbf{o}_v, \mathbf{x}_g^i) = \begin{pmatrix} \mathbf{p}_1^i \\ \mathbf{p}_2^i \\ \mathbf{p}_3^i \\ \mathbf{p}_4^i \end{pmatrix} \equiv [\mathbf{R}_c^\top,\ -\mathbf{R}_c^\top \mathbf{T}_c] \begin{pmatrix} \mathbf{P}_g^1 \\ \mathbf{P}_g^2 \\ \mathbf{P}_g^3 \\ \mathbf{P}_g^4 \end{pmatrix} \tag{6.12}$$

$$\mathbf{P}_r = \frac{1}{2}\begin{bmatrix} -w & -w & w & w \\ h & -h & -h & h \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } \mathbf{P}_g^k = \mathbf{R}(\psi_g^i)\mathbf{P}_r^k + \begin{bmatrix} x_g^i \\ y_g^i \\ 0 \end{bmatrix} \text{ for } k = 1, \ldots, 4. \tag{6.13}$$

  where $\mathbf{p}_v = (x, y, z)$, $\mathbf{o}_v = (\phi, \theta, \psi)$, $\mathbf{x}_g^i = (x_g^i, y_g^i, \psi_g^i)$ and $\equiv$ denotes the equality in homogeneous coordinates. $\mathbf{P}_r^k$ is the $k-$th column of $\mathbf{P}_r$. Currently the lightweight onboard microprocessor limits the update rate of the camera measurement to 7.5Hz.

The orientation drift that occurs in the gyroscope integration (6.2) at the prediction phase is corrected by the GPS ground course, static gravity and line segments in the update phase. For the altitude estimation, we use the barometer rather than the GPS vertical position in (6.6) and

(6.7). This is because the GPS vertical accuracy is relatively poorer than the GPS horizontal accuracy and vulnerable to a systematic error caused by unfavorable satellite locations. While the pressure altitude from the barometer is more noisy than the GPS altitude, the barometer shows better precision and repeatibility than the GPS in measuring the altitude.

6.4 Experimental results

We designed two different air-slalom task scenarios for the demonstration. The first is a *single*-gate case in Figure 6.6 where only one gate is placed in the workspace but a desired gate orientation is repeatedly rotated 90 degrees for the next entrance when the gate entrance is completed. The second is a *multiple*-gate case in Figure 6.11 where two gates are placed and their orientations are unchanged and fixed by the target color.

In both experimental tasks, we used separate motion planning schemes for lateral and longitudinal maneuvers, respectively. For lateral maneuvers moving toward a gate, the MDP planner in Chapter 5.4 was used with taking the target visibility constraint into account. When the vehicle entered the visibility goal region, the straight line following in Chapter 5.7 began to guide the vehicle to the gate. For longitudinal maneuvers, the altitude z_g for each virtual gate was not strictly specified but the flight altitude was maintained between the upper and lower bounds to prevent a crash. At the end of each lateral motion primitive, the vehicle altitude was checked for its desired bound. In the case where the altitude was out of this bound, an appropriate longitudinal motion primitive was immediately executed to push the vehicle back to a safe altitude.

The experiment site at which we performed the slalom task was an obstacle-free and flat environment. Since no architectural structures were present in the workspace, gravity-related line segments were not available for attitude estimation. Hence, the attitude drift in the UKF was mainly reset by an approximate static gravity in (6.10) from the accelerometer after centrifugal accelerations were eliminated. Although this fact may increase the uncertainty of motion primitives due to the degraded accuracy of attitude estimation, the experiments demonstrate that MDP-based motion plans effectively cope with motion uncertainty.

Gate search and initialization:

The UAV first started to navigate around the workspace in a predefined pattern to search and identify all the gates. The first reliable reading of the GPS and barometer set the initial vehicle position (x_0, y_0, z_0) in (6.6) and (6.7) of the UKF. Once four square-shaped ground targets $(\mathbf{p}_0^k, \mathbf{p}_1^k, \mathbf{p}_2^k, \mathbf{p}_3^k)$ were detected for the k-th gate, the gate state $\mathbf{x}_g^k$ was initialized by four points in the world coordinate, each of which is an intersection point of the ground plane and the camera ray computed from a current vehicle state $\mathbf{x}_v$ and target measurement $\mathbf{p}_i^k$. The camera frame

rate was 7.5 Hz. Figure 6.6 displays the trajectories of target states estimated during the gate search until the targets disappeared from the image. The estimation result shows that 10 to 15 times target measurements were sufficient to stabilize the target state (gate posture) in the filter. The initial position deviation was less than 3m, which is relatively small because the targets were detected during level flight where the vehicle attitude would be the most accurate. If the targets were detected during a banked turn, the initial deviation could increase due to a larger attitude error in the filter.

6.4.1 Single-gate air slalom

The task scenario for a single gate is that the UAV repeatedly passes through the same gate in different orientations incremented by 90 degrees each time the gate entrance is completed. Figure 6.7 shows the navigation trajectory and the gate posture estimated in the UKF during the single-gate air slalom. The route shape generated by MDP motion plans produces a four-leaf clover after five gate completions. The images at the top verify the gate orientations which differ by 90 degrees at each entrance. The gate was composed of four identical red squares but one of them (labeled as 1) was rotated 45 degrees with respect to the others.

The jittering of the vehicle position that occurs each time the UAV approaches the gate is caused by a large innovation in the target measurement update in the UKF. When the camera starts to see the ground targets again after turning back, the expected target measurements are less likely to be close to the real ones due to attitude estimation error, which could be large except during a level flight. This error is thus distributed over the vehicle position and heading in the UKF until it is eliminated by a sufficient number of target observations.

Figure 6.8 shows a progressive change of the MDP optimal motion plan as the vehicle approaches the gate. At the end of the execution of each lateral motion primitive, a new optimal plan to the gate was found from a precomputed MDP look-up table that guarantees a sufficient target observation time t_{obs}. The visibility goal region was set to $2.5 \leq t_{obs} \leq 3.5$ sec in Chapter 5.5. In both examples, the first trial to enter the gate failed due to a large deviation from the planned path near the gate and a turning-back motion was then generated to provide a new route. The second trial successfully drove the vehicle to the gate.

Figure 6.9 shows the performance of the straight line following in Chapter 5.7 that enables the vehicle to enter the gate with a desired heading. The vector field for ψ_d is determined by the distance to the straight line in (5.13). In the simplified dynamics of a fixed-wing UAV, the control of a vehicle heading ψ can be considered to be an integrator response of the roll angle ϕ. Hence, the regulation output for the desired heading angle using the simple controller $\phi_d = -K_p(\psi_d - \psi)$ in (5.14) becomes slow and phased off.

Figure 6.10 shows the vehicle speed, altitude, roll and pitch angles regulated by the autopilot

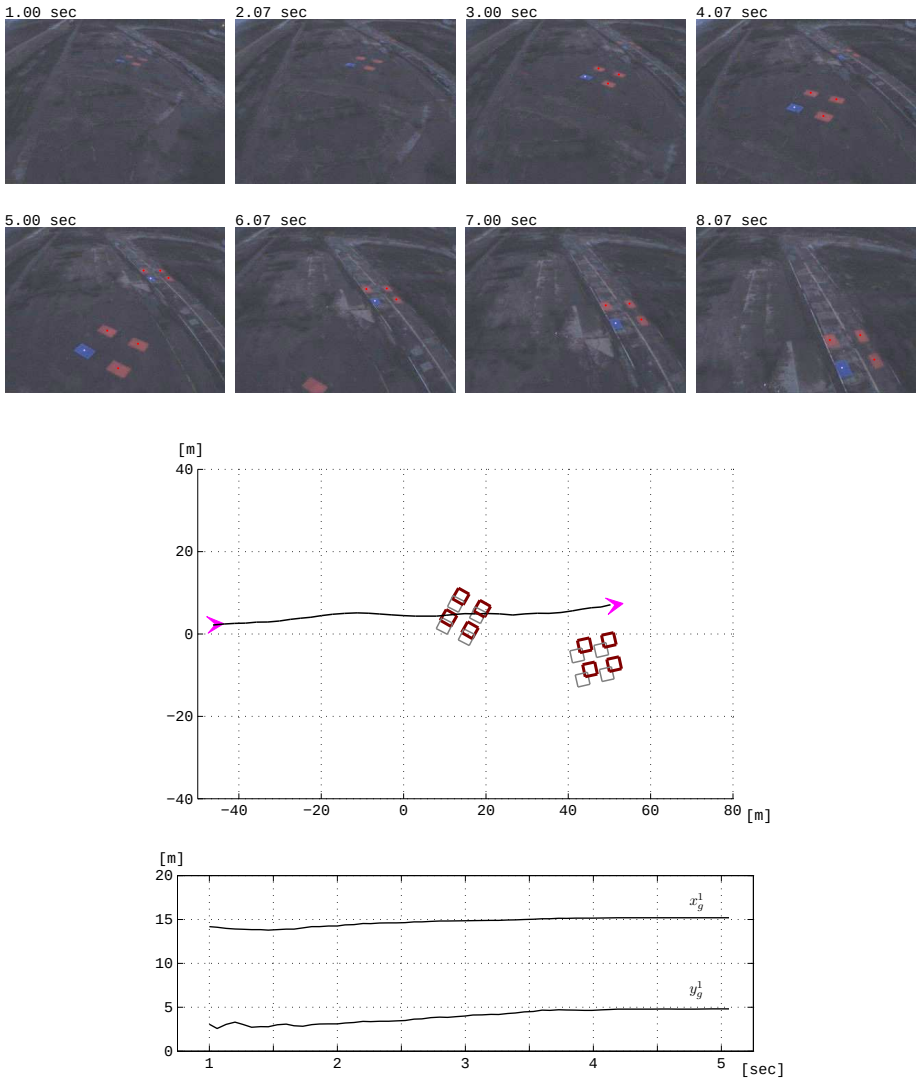


Figure 6.6: Trajectory of the target state estimation from consecutive ground target measurements in the UKF during the gate search. Two gates are detected and each target is observed for about 4 seconds in a 7.5 camera frame rate. (Middle) the gray and red squares for each gate indicate initial and final target poses, respectively.

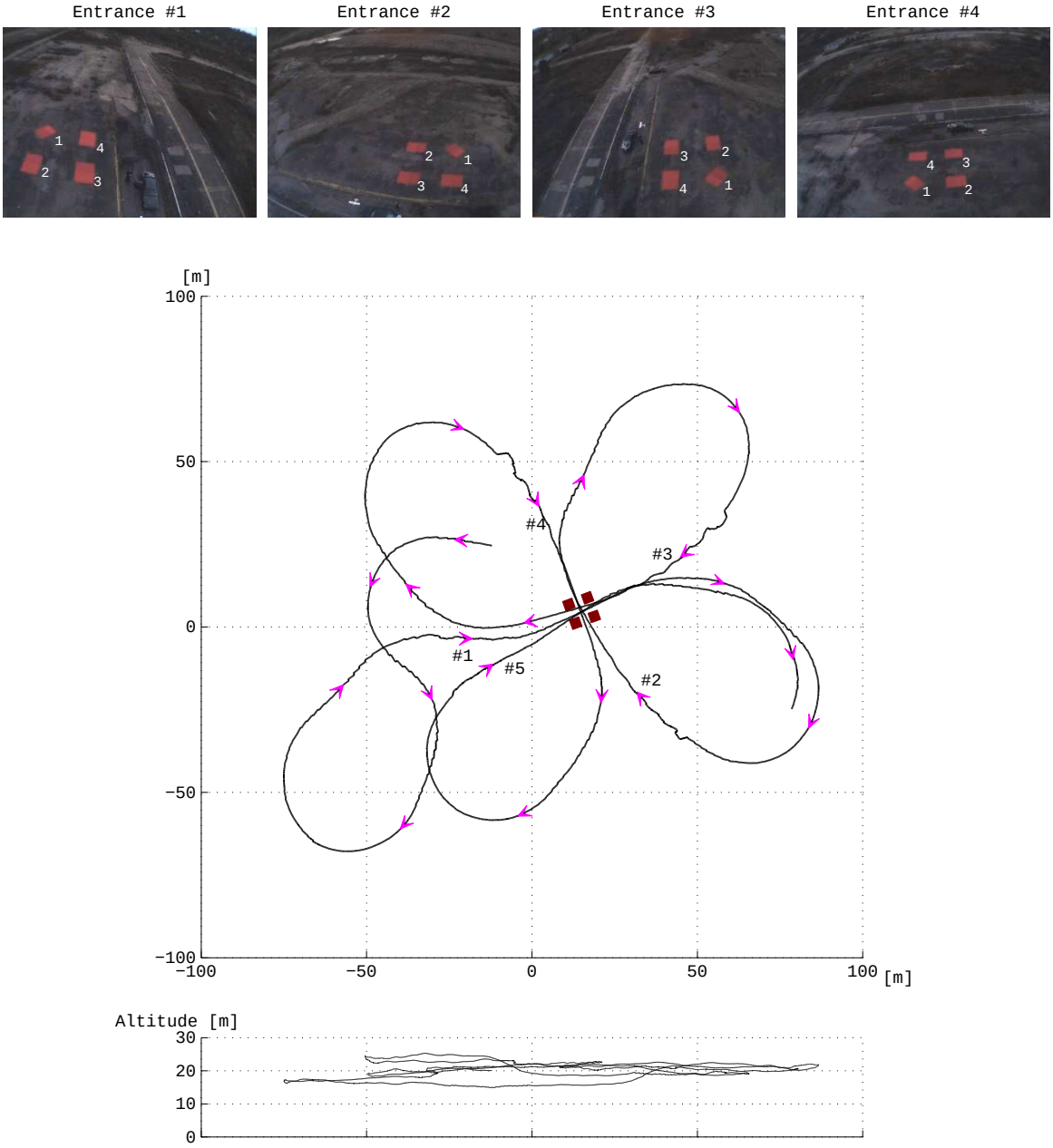
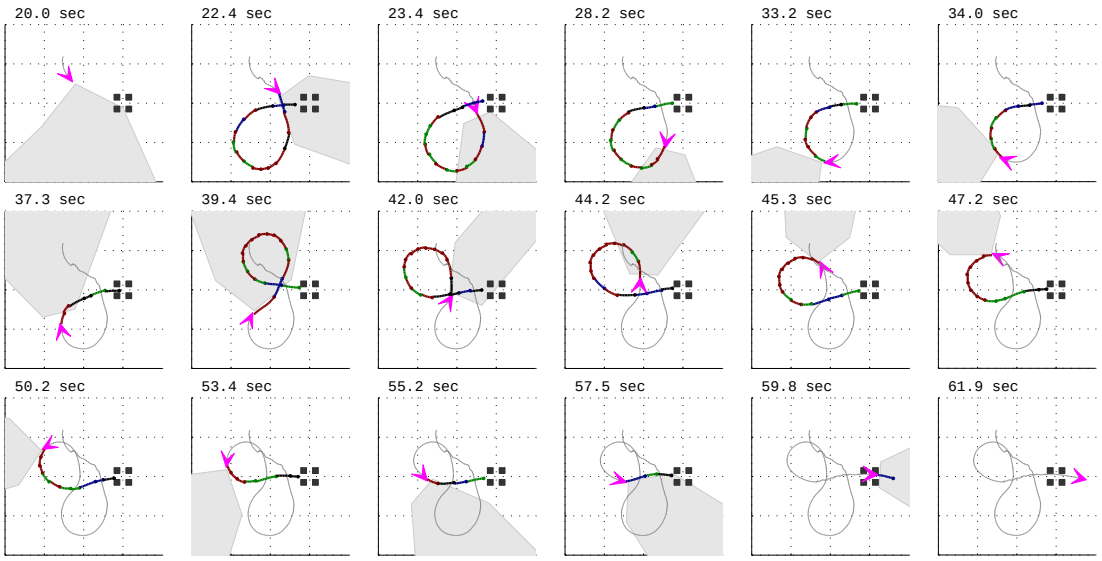
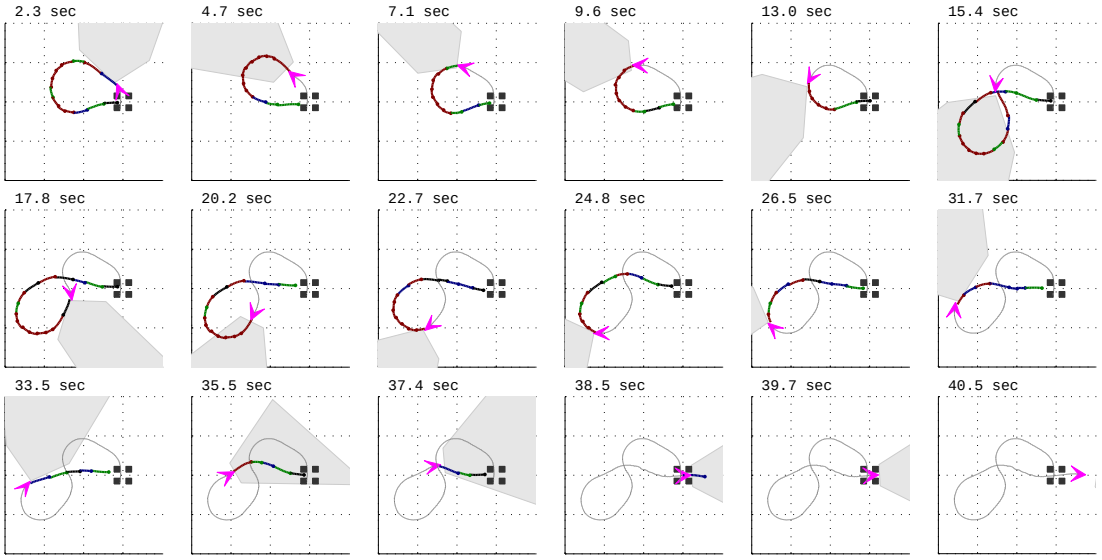


Figure 6.7: Navigation result on the single-gate air slalom task: only one gate is placed in the workspace but a desired gate orientation for the next entrance is repeatedly rotated 90 degrees whenever the gate entrance is completed. The gate entrance is completed five times. The images at the top show the gate at each entrance. All ground targets are red and their first square is rotated by 45 degrees.



(a)



(b)

Figure 6.8: MDP motion planning results in the single-gate air slalom task. At the end of execution of each motion primitive, a new optimal plan to the gate is found from the precomputed MDP look-up table and sent to the autopilot. In both results, the first trial to approach the gate fails due to insufficient target observation time and the entrance is successful at the next approach. The gray area indicates the camera's field-of-view. The color of each motion primitive in the planned path represents its final roll angle (30 degrees in red, 20 in green, 10 in blue, and 0 in black).

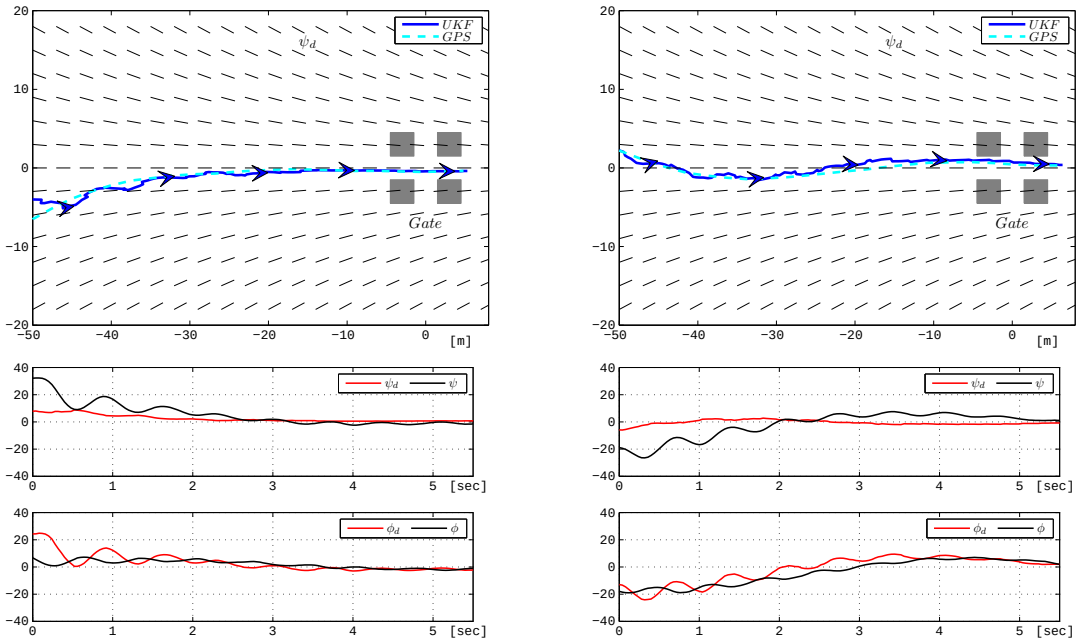


Figure 6.9: Two gate entrance results performed by in the straight line following in the air slalom task. In both results, the vector field for the desired heading angle ψ_d is generated by $\psi_d = -\tan^{-1}(ky)$ in (5.13) and the desired roll angle is given by $\phi_d = -K_p(\psi_d - \psi)$ when $k = 3$ and $K_p = 1.0$.

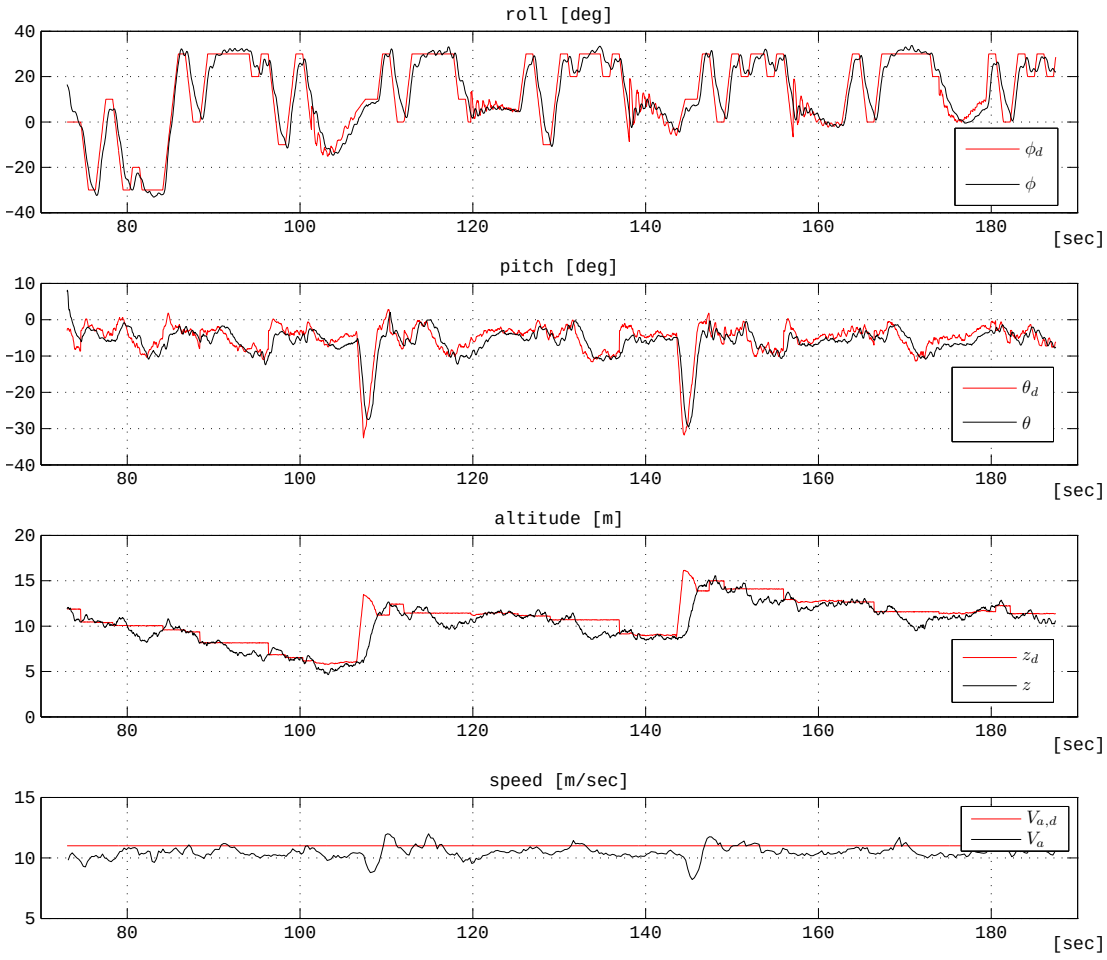


Figure 6.10: Autopilot control outputs to generate lateral and longitudinal motion primitives during the single-gate air slalom task in Figure 6.7. The desired roll angle ϕ_d provided by the MDP planner is switched between -30 to 30 degrees in a 10-degree resolution and most of the lateral motions are right banked turns. The crooked ϕ_d in a high frequency indicates the straight line following mode for the gate entrance. The peaks in the desired pitch angle θ_d occur when altitude recovery is required.

feedback controllers in (5.3) to (5.5). The desired cruise speed $V_{a,d}$ was constantly set to 10.5 m/sec. The desired angles, ϕ_d and θ_d, were provided by a lateral or longitudinal motion primitive chosen in the MDP motion plans. For *lateral* maneuver control, the discrete desired roll ϕ_d switched between -30 to 30 degrees in 10-degree resolution. Dominantly positive ϕ_d indicates that most of lateral motion primitives used for the single-gate slalom were composed of right banked turns as shown in Figure 6.7. The crooked ϕ_d in a high frequency corresponds to the moments when the vehicle was guided by the straight line following in order to enter the gate precisely and ϕ_d was given by a heading error. For *longitudinal* maneuver control, rather than maintaining a constant altitude, a new desired altitude z_d was set as the current altitude at the start of each motion primitive. The desired pitch, $\theta_d = -K_{p,z}(z_d - z)$, was then dedicated to regulate the altitude error. The peaks in the desired pitch angle θ_d occurred when altitude recovery was explicitly required.

6.4.2 Multiple-gate air slalom

The same navigation algorithms used in the single-gate air slalom were also applied for the multiple-gate air slalom task. Two gates were placed in the workspace and their orientations were unchanged and identifiable by a blue ground target. Since both gates were identical in shape, the first found gate was labeled $\mathbf{x}_g^1$ and needed to be passed through first. The search and identification process for both gates have already been shown in Figure 6.6.

In Figure 6.11, an entire navigation trajectory to enter the two gates sequentially was completed twice. The images at the top show the gate observations at the corresponding moments indicated in the plot by the number. The minimum turning radius was about 20m when a lateral motion primitive maintained the 30-degree roll angle. The relative posture between the gates was estimated to be (27.5m, $-$20.1m) and 45.2 degrees. This slalom setting led our optimal motion planning to eventually produce a figure-eight route shape.

The MDP motion planner achieved a successful autonomous guidance to enter the gates multiple times. Figure 6.12 shows the progress of the motion plan during the first slalom completion. Entering the first gate was repeated three times because position and heading deviations from the planned path were larger than the motion uncertainty modeled in the MDP due to an unexpected wind gust. The entrance to the second gate was smoothly accomplished.

Figure 6.13 shows the responses of the autopilot feedback controllers during the multiple-gate air slalom. Since the figure-eight route commanded a full swing of the roll angle, both right and left-turn lateral motion primitives were used almost equally. The maximum overshoot in the regulation of ϕ was about 10 degrees when ϕ_d changed quickly. The altitude gradually decreased since the execution of a lateral motion lost the altitude by up to 5m. When the altitude was below 15m, a longitudinal motion was executed to recover the altitude.

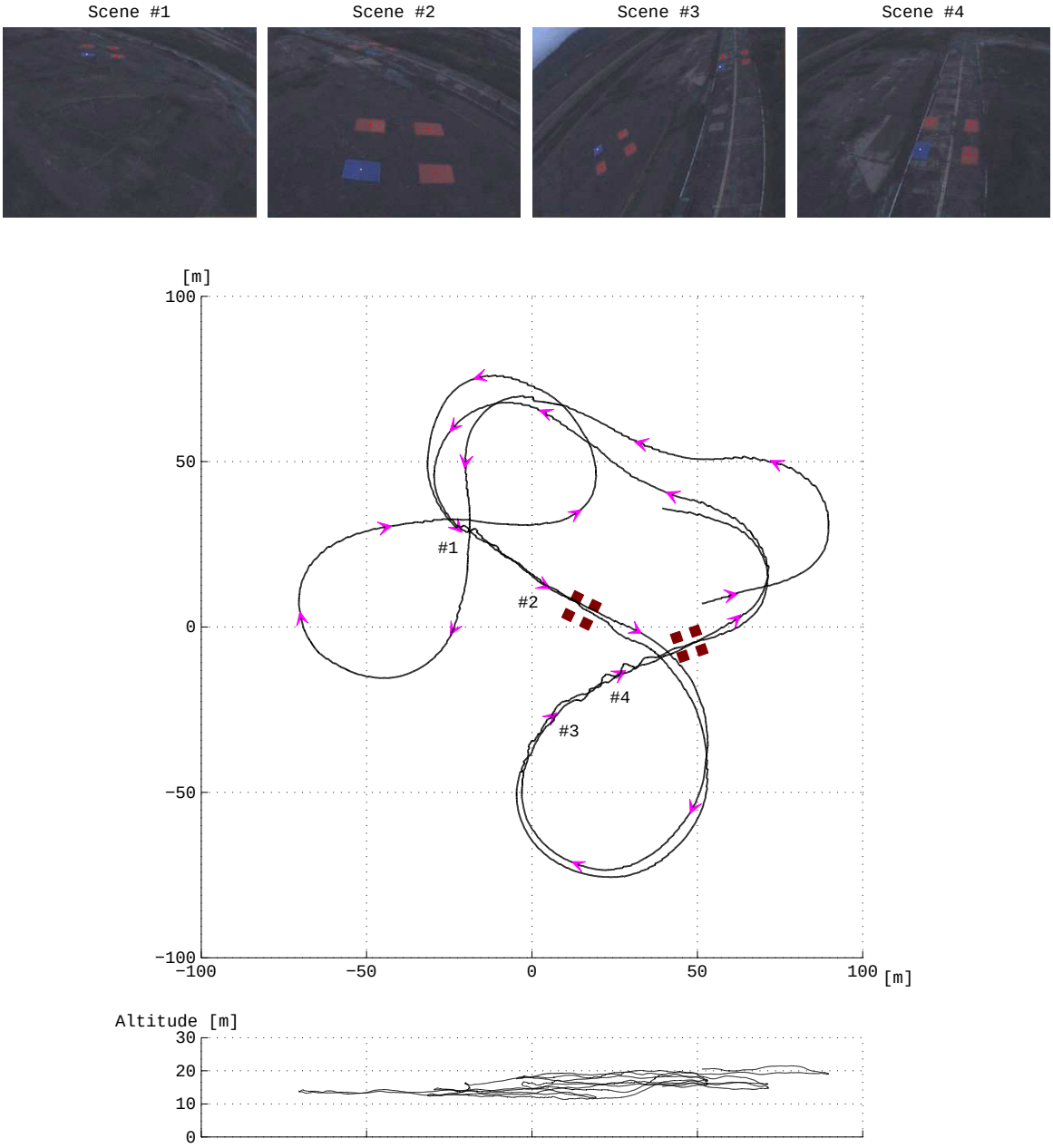


Figure 6.11: Navigation result on the multiple-gate air slalom task: two gates are placed in the workspace and their orientations are unchanged and fixed by the blue square target. The whole route to enter both gates in a row is completed twice. The images at the top show the gate observations at the corresponding moments indicated in the plot.

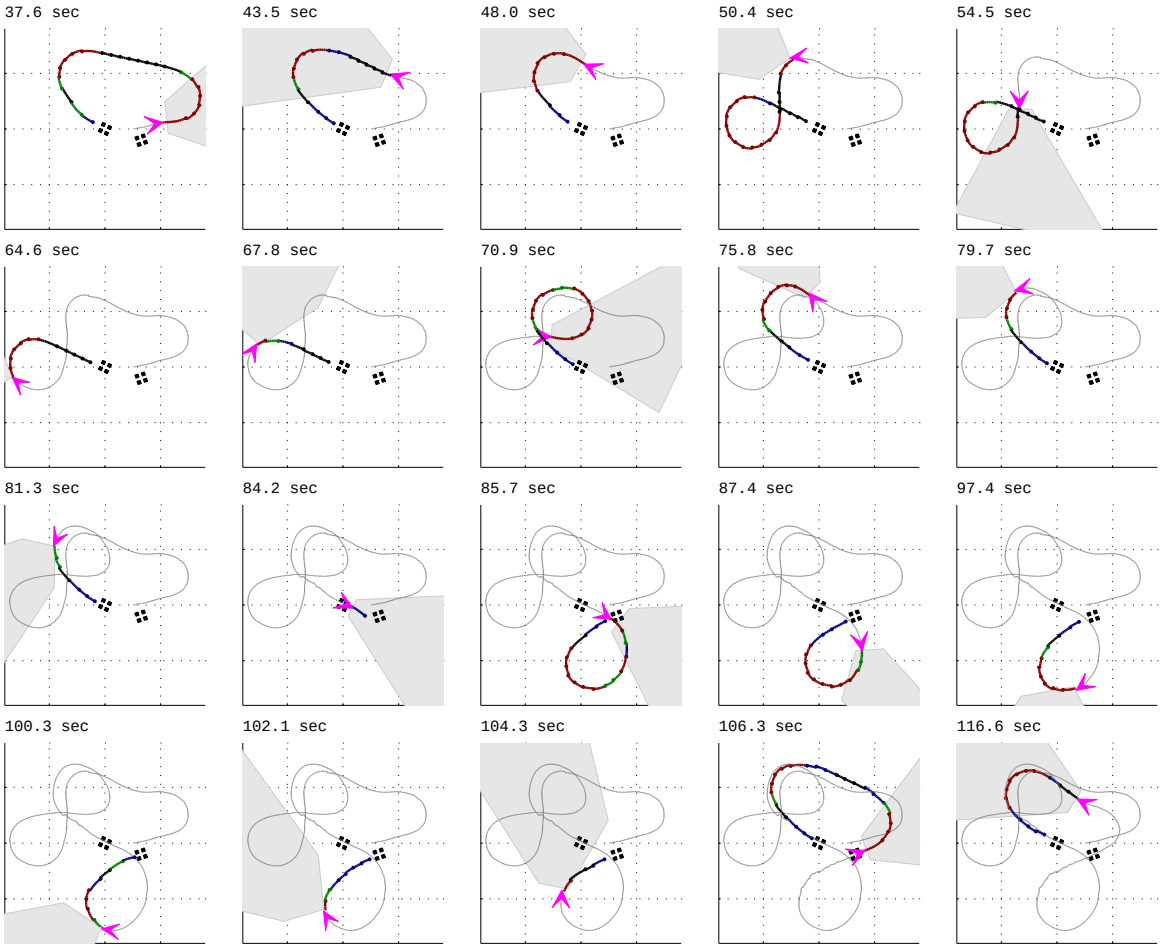


Figure 6.12: MDP motion planning results in the multiple-gate air slalom task. At the end of execution of each motion primitive, a new optimal plan to the gate is found from the precomputed MDP look-up table and sent to the autopilot. The first two trials to approach the first gate fail due to insufficient target observation time, but the entrance is successful at the third approach. The gray area indicates the camera's field-of-view. The color of each motion primitive in the planned path represents its final roll angle (30 degrees in red, 20 in green, 10 in blue, and 0 in black).

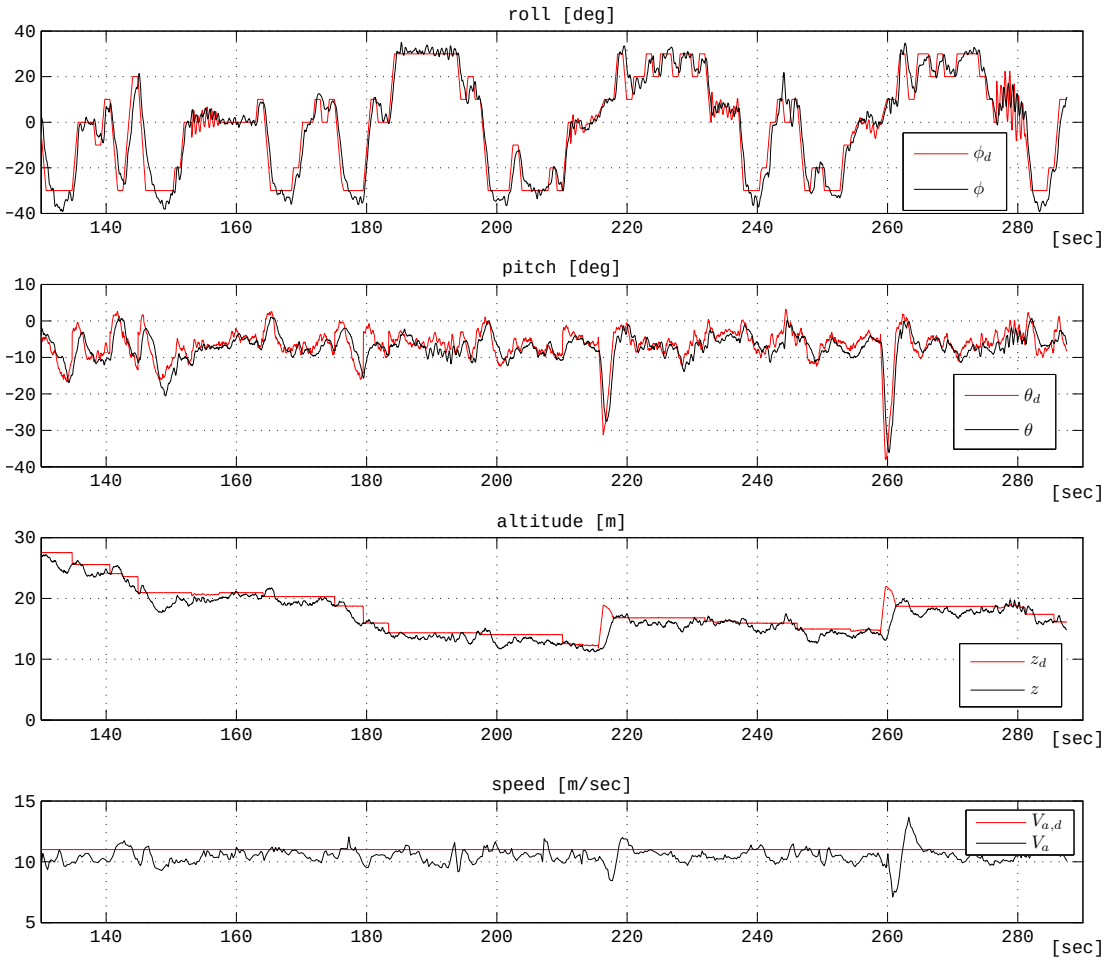


Figure 6.13: Autopilot control outputs to generate lateral and longitudinal motion primitives during the multiple-gate air slalom task in Figure 6.11. The MDP planner provides the desired roll angle ϕ_d which is switched between -30 to 30 degrees in a 10-degree resolution. The crooked ϕ_d in a high frequency indicates the line following mode near the gate entrance. Altitude recovery by the peaks in the desired pitch angle occurs only when the entrance is completed and the altitude is less than 15m.

Chapter 7

Conclusion and Future Work

This thesis presented vision-based algorithms and motion planning methods that are useful and efficient building blocks for the autonomous navigation system for a small unmanned fixed-wing airplane operating in an urban environment. The problems we address are how to provide a driftless attitude estimation for flight control, how to conveniently calibrate visual and inertial sensors in a working field, and how to generate a feasible motion plan that considers limited agility in vehicle motion. The solutions to these problems are successfully integrated and demonstrated in the air slalom task, which contains fundamental utilities required to accomplish typical autonomous navigation tasks presented in an urban area. In this chapter, we summarize these approaches and review the contributions, and propose some ideas for future work.

7.1 Summary

Visual-inertial attitude estimation Our approach to the gyroscope drift problem in the attitude estimation is to use gravity-related line segments available in an urban scene. The regularity existing in vertical and horizontal edges of man-made structures, which is associated with vanishing point detection, can be effectively used to bound the attitude drift. In the Extended Kalman filter which combines inertial and visual measurements, we presented new update equations for vertical and horizontal line segments, respectively. To achieve a fast and efficient line classification in the presence of outliers, the hierarchical clustering uses a current best attitude estimate as the prior in a RANSAC framework. The flight experiment around a football field at 15m high showed that the maximum attitude error during dynamic maneuvers is bounded to 5 degrees, which is two times smaller than that of the inertial-only method.

Sensor calibration for visual and inertial fusion Traditional calibration methods may be meticulous but they demand expertise, labor, time and specialized equipment. Instead we

presented a new self-calibration method that requires no prior knowledge of a sensor setting and easily performs in a field using natural reference features. We proved that the factorization method can provide a linear solution for intrinsic parameters of an IMU using known magnitudes of exerted forces, which are conveniently obtainable from gravity at a static condition for accelerometers, and tracks of distant scene points for gyroscopes, respectively. Once the IMU is calibrated, vertical edges of buildings are used for the calibration of a camera and sensor-to-sensor relative orientation. The simulation and experiments showed that the accuracy of our linear method is fairly comparable to that of other object-based methods or nonlinear solutions.

Trim state-based motion planning The challenge in motion planning for a fixed-wing is to provide a feasible motion plan that can be executable by flight controllers. In addition, not only must task-oriented constraints like a minimum target observation time be taken into account but motion uncertainty due to wind or control error must also be considered for planning.

We addressed these problems through a discrete planning based on the finite trim states of a fixed wing. A set of discrete primitive motions was constructed as transitions between the trim states; the interconnections of these motion primitives were explored to find a feasible motion plan for reaching a goal. Rather than seeking for the shortest path, we used a probabilistic model of motion uncertainty and maximized the probability of success in a goal region in the MDP. A minimum target observation time is guaranteed by setting the visibility region as an intermediate goal. The air-slalom experiments demonstrated that the MDP-based motion planner successfully guides the vehicle to enter a gate.

Air-slalom task experiments We designed the air-slalom task where the UAV's mission is to search and locate several gates on the ground, and pass through them precisely. The purpose of this task is to evaluate the capability of our autonomous navigation system in performing maneuvers commonly required in urban operations. The successful completion of air-slalom courses composed of single or multiple gates demonstrated the effectiveness of our navigation system including the motion planning based on finite trim states, and the vehicle and target states estimated by a low-grade GPS, IMU, barometer and camera.

7.2 Contributions

The following is a list of the contributions presented in this thesis in the areas of attitude estimation, sensor calibration, and motion planning for unmanned fixed-wing airplanes operating in an urban environment:

- A sensor fusion algorithm that combines inertial gyroscope and visual line segment measurements and produces driftless attitude estimation.

- A factorization-based IMU calibration method that provides a linear solution for a redundant IMU and an iterative solution for a triad IMU.
- A thorough analysis of the linear reconstruction space of the factorization and an identification of the rank condition and degenerate load configurations.
- An efficient discrete motion planning method based on trim states that reduces the dimension of a configuration space in planning.
- An extension of Dubins metric to 3D for search-based planning.
- A development of a low-cost deployable and expendable fixed-wing UAV system and an integration of the proposed algorithms into the navigation system.
- A demonstration of the air-slalom task for a single and multiple gates.

7.3 Future Work

The urban operation of a small fixed-wing airplane is an emerging research field that addresses challenging problems both in innovative system integration and the inevitable trade-off between payload feasibility and performance. A fully autonomous and intelligent UAV still requires more advanced perception, control and planning; this thesis contributed to this endeavor in part with the presented methods and results. The following paragraphs summarize future research directions on the topics dealt with in this thesis and possible enhancements of the proposed algorithms.

Line feature tracking In Chapter 3, line measurements used for the attitude estimation are independently extracted in each image. Hence, the proposed line classification method always requires a fresh start for a new set of lines. If line correspondences are available through tracking, and their associations with vanishing points at the previous frame are known, this would significantly reduce the time needed for the classification process. It is also possible to add the heading direction ψ into the Kalman estimator in a situation where one prevalent horizontal vanishing point can be tracked over frames.

More dynamic maneuvers The motion primitives used in the experiments have a maximum of 30 degrees in the roll angle and thus produce mild maneuvers. Since they are only a subset of all possible maneuvers, more dynamic maneuvers can be considered by increasing the maximum roll angle or adding aerobatic motions like looping. These will provide tighter turns and more flexible motions in 3D. The challenge, however, is that dynamic maneuvers also require high-grade sensors for more accurate state estimation, and an advanced control technique to stabilize the flight.

Obstacle detection and avoidance In the air-slalom experiment, one essential but missing component that was not tested or fully demonstrated for the autonomous navigation is obstacle detection and avoidance. Simple floating obstacles like a balloon or virtual no-go regions can be placed for an obstacle avoidance test. To improve the quality of an obstacle-free path generated by the greedy-first search in Chapter 5.3, it would be possible to integrate the search-based planning with the MDP-based optimal path in order to produce a more smooth and near-optimal plan that dodges obstacles.

Experiment in a more realistic urban area For safety reasons, the experiment site where we demonstrated the air-slalom task was a demolished urban area. Based on the proven navigation capability to run a slalom course and enter a gate, the next challenging missions for our UAV system could provide more realistic urban task scenarios, such as traveling through a passageway between buildings or approaching a target on the road in a cluttered area.

Appendix A

Derivation of Jacobian for Line Angle

A.1 Derivation of Jacobian in EKF Prediction

Let $\Delta\omega_i = (\omega_i - b_i)\Delta T$ for $i = x, y$ or z. The process model (3.10) for $\boldsymbol{\chi} = [\mathbf{x}; \mathbf{b}]$ is rewritten in a discrete time as $\boldsymbol{\chi}_{k+1} = \mathbf{f}(\boldsymbol{\chi}_k)$, *i.e.*,

$$\begin{bmatrix} \phi_{k+1} \\ \theta_{k+1} \\ b_{x,k+1} \\ b_{y,k+1} \\ b_{z,k+1} \end{bmatrix} = \begin{bmatrix} \phi_k + (\omega_x - b_x)\Delta T + s_\phi t_\theta(\omega_y - b_y)\Delta T + c_\phi t_\theta(\omega_z - b_z)\Delta T \\ \theta_k + c_\phi(\omega_y - b_y)\Delta T - s_\phi(\omega_z - b_z)\Delta T \\ b_{x,k} \\ b_{y,k} \\ b_{z,k} \end{bmatrix} \tag{A.1}$$

Then, the Jacobian $\mathbf{F} = \dfrac{\partial \mathbf{f}}{\partial \boldsymbol{\chi}}$ in the EKF state transition is given as

$$\mathbf{F} = \begin{bmatrix} 1 + c_\phi t_\theta \Delta\omega_y - s_\phi t_\theta \Delta\omega_z & s_\phi/c_\theta^2 \Delta\omega_y + c_\phi/c_\theta^2 \Delta\omega_z & -\Delta T & -s_\phi t_\theta \Delta T & -c_\phi t_\theta \Delta T \\ -s_\phi \Delta\omega_y - c_\phi \Delta\omega_z & 1 & 0 & -c_\phi \Delta T & s_\phi \Delta T \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

A.2 Derivation of Jacobian in EKF Update

Let $s_x = v_x^* - m_x$, $s_y = v_y^* - m_y$, $s = s_x/s_y$. Then, the line angle β in (3.13) is rewritten as

$$\beta = \tan^{-1}\left(\frac{v_y^* - m_y}{v_x^* - m_x}\right) = \tan^{-1}\left(\frac{s_y}{s_x}\right) = \tan^{-1}(s) \tag{A.3}$$

and its partial derivative using the chain rule is

$$\frac{\partial \beta}{\partial \square} = \frac{\partial \tan^{-1}(s)}{\partial \square} = \left(\frac{1}{1+s^2}\right)\frac{\partial s}{\partial \square} = \frac{s_x^2}{s_x^2+s_y^2}\left(\frac{s_y' s_x - s_y s_x'}{s_x^2}\right) = \frac{v_y' s_x - v_x' s_y}{d^2} \tag{A.4}$$

As denoted in Figure 3.6, let $d = \sqrt{s_x^2 + s_y^2}$, $r = \sqrt{v_x^{*2} + v_y^{*2}}$, $n_x = c_\phi m_x - s_\phi m_y$, and $n_y = s_\phi m_x + c_\phi m_y$. From (3.5) and (3.9), we also can rewrite r as follows: $r = 1/t_\theta$ for $\mathbf{v}_v^*$ and $r = t_\psi^2/c_\theta^2$ for $\mathbf{v}_h^*$, respectively.

For $\mathbf{v}_v^* = \left[\, s_\phi t_\theta^{-1},\ \ c_\phi t_\theta^{-1} \,\right]^\top$ in (3.5), the Jacobian for the line angle associated with a vertical VP is derived as follows.

$$\frac{\partial \mathbf{v}_v^*}{\partial \phi} = \begin{bmatrix} v_x' \\ v_y' \end{bmatrix} = \begin{bmatrix} c_\phi t_\theta^{-1} \\ -s_\phi t_\theta^{-1} \end{bmatrix} \tag{A.5}$$

$$\begin{aligned} v_y' s_x - v_x' s_y &= -(s_\phi t_\theta^{-1})(v_x - m_x) - (c_\phi t_\theta^{-1})(v_y - m_y) \\ &= -t_\theta^{-1}(s_\phi v_x + c_\phi v_y) + t_\theta^{-1}(s_\phi m_x + c_\phi m_y) \\ &= -t_\theta^{-1}(s_\phi s_\phi t_\theta^{-1} + c_\phi c_\phi t_\theta^{-1}) + t_\theta^{-1}(s_\phi m_x + c_\phi m_y) \\ &= -t_\theta^{-2} + t_\theta^{-1} n_y = -r^2 + r n_y \end{aligned} \tag{A.6}$$

$$\frac{\partial \mathbf{v}_v^*}{\partial \theta} = \begin{bmatrix} v_x' \\ v_y' \end{bmatrix} = \begin{bmatrix} -s_\phi s_\theta^{-2} \\ -c_\phi s_\theta^{-2} \end{bmatrix} \tag{A.7}$$

$$\begin{aligned} v_y' s_x - v_x' s_y &= (-c_\phi s_\theta^{-2})(v_x - m_x) - (-s_\phi s_\theta^{-2})(v_y - m_y) \\ &= -s_\theta^{-2}(c_\phi v_x - s_\phi v_y) + s_\theta^{-2}(c_\phi m_x - s_\phi m_y) \\ &= -s_\theta^{-2}(c_\phi s_\phi t_\theta^{-1} - s_\phi c_\phi t_\theta^{-1}) + s_\theta^{-2}(c_\phi m_x - s_\phi m_y) = s_\theta^{-2} n_x \end{aligned} \tag{A.8}$$

For $\mathbf{v}_h^* = \left[\, c_\phi t_\psi / c_\theta - s_\phi t_\theta,\ \ -s_\phi t_\psi / c_\theta - c_\phi t_\theta \,\right]^\top$ in (3.9), the Jacobian for the line angle associated with a horizontal VP is derived as follows.

$$\frac{\partial \mathbf{v}_h^*}{\partial \phi} = \begin{bmatrix} v_x' \\ v_y' \end{bmatrix} = \begin{bmatrix} -s_\phi t_\psi / c_\theta - c_\phi t_\theta \\ -c_\phi t_\psi / c_\theta + s_\phi t_\theta \end{bmatrix} \tag{A.9}$$

$$
\begin{aligned}
v'_y s_x - v'_x s_y &= (-c_\phi t_\psi / c_\theta + s_\phi t_\theta)(v_x - m_x) + (s_\phi t_\psi / c_\theta + c_\phi t_\theta)(v_y - m_y) \\
&= (-c_\phi t_\psi / c_\theta + s_\phi t_\theta)(c_\phi t_\psi / c_\theta - s_\phi t_\theta) + (s_\phi t_\psi / c_\theta + c_\phi t_\theta)(-s_\phi t_\psi / c_\theta - c_\phi t_\theta) \\
&\quad - (-c_\phi t_\psi / c_\theta + s_\phi t_\theta) m_x - (s_\phi t_\psi / c_\theta + c_\phi t_\theta) m_y \\
&= -(c_\phi t_\psi / c_\theta - s_\phi t_\theta)^2 - (s_\phi t_\psi / c_\theta + c_\phi t_\theta)^2 + t_\psi / c_\theta (c_\phi m_x - s_\phi m_y) - t_\theta (s_\phi m_x + c_\phi m_y) \\
&= -t_\psi^2 / c_\theta^2 - t_\theta^2 + (t_\psi / c_\theta) n_x - t_\theta n_y = -r^2 + (t_\psi / c_\theta) n_x - t_\theta n_y
\end{aligned} \tag{A.10}
$$

$$
\frac{\partial \mathbf{v}_h^*}{\partial \theta} = \begin{bmatrix} v'_x \\ v'_y \end{bmatrix} = \begin{bmatrix} c_\phi t_\psi t_\theta / c_\theta - s_\phi / c_\theta^2 \\ -s_\phi t_\psi t_\theta / c_\theta - c_\phi / c_\theta^2 \end{bmatrix} \tag{A.11}
$$

$$
\begin{aligned}
v'_y s_x - v'_x s_y &= (-s_\phi t_\psi t_\theta / c_\theta - c_\phi / c_\theta^2)(v_x - m_x) - (c_\phi t_\psi t_\theta / c_\theta - s_\phi / c_\theta^2)(v_y - m_y) \\
&= (-s_\phi t_\psi t_\theta / c_\theta - c_\phi / c_\theta^2)(c_\phi t_\psi / c_\theta - s_\phi t_\theta) - (c_\phi t_\psi t_\theta / c_\theta - s_\phi / c_\theta^2)(-s_\phi t_\psi / c_\theta - c_\phi t_\theta) \\
&\qquad + 1/c_\theta^2 (c_\phi m_x - s_\phi m_y) + t_\psi t_\theta / c_\theta (s_\phi m_x + c_\phi m_y) \\
&= -(s_\phi t_\psi t_\theta / c_\theta + c_\phi / c_\theta^2)(c_\phi t_\psi / c_\theta - s_\phi t_\theta) + (c_\phi t_\psi t_\theta / c_\theta - s_\phi / c_\theta^2)(s_\phi t_\psi / c_\theta + c_\phi t_\theta) \\
&\qquad + (1/c_\theta^2) n_x + t_\psi t_\theta / c_\theta \, n_y \\
&= t_\psi t_\theta^2 / c_\theta - t_\psi / c_\theta^3 + (1/c_\theta^2) n_x + (t_\psi t_\theta / c_\theta) n_y \\
&= (t_\psi / c_\theta^3)(s_\theta^2 - 1) + (1/c_\theta^2) n_x + (t_\psi t_\theta / c_\theta) n_y \\
&= -(t_\psi / c_\theta) + (1/c_\theta^2) n_x + (t_\psi t_\theta / c_\theta) n_y \\
&= (1/c_\theta^2)(n_x + t_\psi (-c_\theta + s_\theta n_y))
\end{aligned} \tag{A.12}
$$

Appendix B

Discussion on Factorization-based Calibration

In this chapter, we discuss the linear solution space of $\mathbf{L}$ constructed by a load constraint set $\mathcal{C}$ at the redundant calibration described in Section 4.3. Let $|\mathcal{C}|$ denote the number of independent constraints in $\mathcal{C}$.

B.1 Practical Issues

Since the measurement $\mathbf{Z}$ is inevitably corrupted by noise, additional steps are required to meet the assumptions made for the factorization. Some considerations and caveats are:

- In the SVD of $\mathbf{Z}$ in (4.4), a noisy $\mathbf{Z} \in \mathcal{R}^{m \times n}$ may have a rank greater than p. The rank-p enforcement on $\mathbf{Z}$ can be made by

$$\widehat{\mathbf{Z}} = (\mathbf{U}_p \mathbf{\Sigma}_p^{1/2})(\mathbf{\Sigma}_p^{1/2} \mathbf{V}_p^\top) = \widehat{\mathbf{F}_b}\,\widehat{\mathbf{S}_b} \tag{B.1}$$

  where $\mathbf{\Sigma}_p$ is the top left $p \times p$ block of $\mathbf{\Sigma}$, and $\mathbf{U}_p$ and $\mathbf{V}_p$ are the first p columns of $\mathbf{U}$ and $\mathbf{V}$, respectively. $\widehat{\mathbf{Z}}$ is the closest rank-p matrix to $\mathbf{Z}$ in the sense of a Frobenius norm.

- When rank($\mathbf{L}$) = 10, a least squares solution $\mathbf{q}_p$ (4.16) may result in a full-rank $\mathbf{Q}$ that violates the rank condition of q-constraint (4.15a). To enforce the rank-3 $\mathbf{Q}$, the smallest singular value of $\mathbf{Q}$ is set to zero by the SVD of $\mathbf{Q}$.

- When rank($\mathbf{L}$) = 9, multiple solutions for $\mathbf{q}$ may exist (case 2 in Fig. 4.4) but corresponding $\mathbf{S}_b$ are discriminative enough to identify inappropriate ones that have severely distorted and warped shapes. Since the bias is generally uniform, a true solution can be selected as one of the smallest variances of $\mathbf{b}$ in $\mathbf{S}_b$.

- If measurement noise in $\mathbf{Z}$ is too high, the solution for $\mathbf{q}$ does not always produce a positive-definite $\mathbf{Q}_{3\times3}$. Thus, the Cholesky decomposition fails to obtain $\mathbf{A}_{3\times3}$. This sensitivity to noise can be reduced if further loads are made, so that more constraints are used.

B.2 Numerical Example

This is an example of when measurements are collected into $\mathbf{Z}$ with no measurement noise from a four-component accelerometer unit and ten different attitudes in a static condition ($n = 4, m = 10$). From the SVD of $\mathbf{Z}$ with $\text{rank}(\mathbf{Z}) = 4$, $\widehat{\mathbf{F}_b}$ and $\widehat{\mathbf{S}_b}$ are computed as

$$\underset{\mathbf{Z}}{\begin{bmatrix} 0.134 & 1.433 & 0.750 & 1.393 \\ 0.134 & 0.567 & 0.750 & 1.826 \\ 0.412 & 1.000 & 0.191 & 1.866 \\ 0.792 & 1.847 & 0.511 & 1.026 \\ 0.792 & 0.153 & 0.511 & 1.873 \\ 1.208 & 1.847 & 0.511 & 0.818 \\ 1.208 & 1.000 & 0.022 & 1.588 \\ 1.588 & 1.701 & 0.595 & 0.642 \\ 1.588 & 0.299 & 0.595 & 1.342 \\ 1.866 & 1.000 & 0.500 & 0.921 \end{bmatrix}} = \underset{\widehat{\mathbf{F}_b}}{\begin{bmatrix} -0.75 & 0.22 & 0.64 & -0.26 \\ -0.67 & 0.71 & 0.21 & -0.32 \\ -0.78 & 0.49 & 0.24 & 0.40 \\ -0.86 & -0.30 & 0.48 & 0.03 \\ -0.70 & 0.66 & -0.38 & -0.08 \\ -0.88 & -0.53 & 0.26 & 0.01 \\ -0.85 & 0.11 & -0.20 & 0.57 \\ -0.89 & -0.68 & -0.02 & -0.13 \\ -0.76 & 0.11 & -0.73 & -0.23 \\ -0.85 & -0.41 & -0.53 & -0.08 \end{bmatrix}} \underset{\widehat{\mathbf{S}_b}}{\begin{bmatrix} -1.25 & -1.40 & -0.60 & -1.61 \\ -0.68 & -0.72 & 0.08 & 1.13 \\ -1.00 & 0.90 & 0.09 & -0.04 \\ 0.04 & 0.13 & -0.84 & 0.17 \end{bmatrix}}$$

From $\mathcal{C}^*_{\tau=1}$, $\mathbf{L}$ has a one-dimensional null space and the corresponding solutions for $\mathbf{q}$ and α are

$$\begin{aligned} \mathbf{q}_p &= \begin{bmatrix} 1.28 & -0.15 & -0.01 & 0.00 & 0.39 & 0.01 & -0.09 & 0.42 & 0.00 & 0.15 \end{bmatrix}^\top \\ \mathbf{q}_n &= \begin{bmatrix} 0.40 & -0.11 & 0.03 & 0.23 & -0.55 & -0.01 & 0.10 & -0.65 & 0.00 & -0.21 \end{bmatrix}^\top \\ \alpha &= \begin{pmatrix} 0.65 & 0.65 & 0.65 & -2.14 \end{pmatrix} \end{aligned}$$

Since only $\alpha = -2.14$ satisfies the q-constraint (4.15), the true $\mathbf{S}_b$ and $\mathbf{F}_b$ are recovered as

$$\mathbf{A}_{4\times4} = \begin{bmatrix} 0.65 & 0 & 0 & -1.24 \\ 0.13 & 1.24 & 0 & 0.18 \\ -0.11 & 0.03 & 1.34 & -0.01 \\ -0.76 & -0.17 & -0.05 & -0.12 \end{bmatrix}, \quad \mathbf{S}_b = \begin{bmatrix} 1 & 0 & 0 & -0.50 \\ 0 & 1 & 0 & -0.50 \\ 0 & 0 & 1 & -0.71 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad \mathbf{F}_b = \begin{bmatrix} -0.87 & 0.43 & -0.25 & 1 \\ -0.87 & -0.43 & -0.25 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0.59 & -0.70 & -0.40 & 1 \\ 0.87 & 0.00 & -0.50 & 1 \end{bmatrix}$$

B.3 Minimum number of load constraints

The factorization-based calibration is equivalent to finding $\mathbf{q}$ subject to the q-constraint (4.15). The 9 degrees-of-freedom in $\mathbf{q}$ can be verified again from the fact that, for a given $\mathbf{Q}$, $\mathbf{A}_{3\times3}$

in $\mathbf{A}_{4\times3}$ is up to a rotation matrix $\mathbf{R}$ $(12-3=9)$. Hence, nine load constraints $(k=9)$ are sufficient to find $\mathbf{q}$,

$$|\mathcal{C}|_{min} = 9 \tag{B.2}$$

which corresponds when $\text{rank}(\mathbf{L}) = 9$ in (4.17).

The required number of calibrating loads (m) varies depending on how individual loads are associated with each other in $\mathcal{C}$. For example, when $|\mathcal{C}| = |\mathcal{C}|_{min}$, the maximum increases up to $m = 18$ for $\mathcal{C} = \{c_{12},\ c_{34},\ c_{56},\ \ldots\}$, and the minimum is $m = 4$ when $\mathcal{C} = \{c_{11},\ c_{22},\ c_{33},\ c_{44},\ c_{12},\ c_{13},\ c_{14},\ c_{23},\ c_{24}\}$ from which a linearly dependent c_{34} is excluded. The minimal load condition $(m = 4)$ coincides with the case where a matrix $\mathbf{F}_b \in \mathcal{R}^{4\times4}$ is completely known and the calibration is directly solvable by $\mathbf{S}_b = \mathbf{F}_b^{-1}\mathbf{Z}$.

B.4 Gravity magnitude constraint

From Definition 1, a load constraint set solely composed of the gravity magnitude is $\mathcal{C}^*_{\tau=1}$ (a known constant magnitude is scaled to 1 without loss of generality). Firstly, the following lemmas are derived to investigate the rank of $\mathbf{L}$.

Definition 2. $\text{rank}(\mathbf{L}|\widehat{\mathbf{F}_b}, \mathcal{C})$ *is the rank of* $\mathbf{L}$ *when* $\widehat{\mathbf{F}_b}$ *and* $\mathcal{C}$ *are given for the construction of* $\mathbf{L}$ *in (4.13).*

Lemma 1. $\text{rank}(\mathbf{L}|\widehat{\mathbf{F}_b},\ \mathcal{C}) = \text{rank}(\mathbf{L}|\mathbf{F}_b,\ \mathcal{C})$ *for any* $\mathcal{C}$.

Lemma 2. $\text{rank}(\mathbf{L}|\widehat{\mathbf{F}_b},\ \mathcal{C}) = \min\{10,\ \text{rank}(\mathcal{S}[\mathbf{f}_{b,i} \otimes \mathbf{f}_{b,j}])\}$

Proof. (for Lemma 1 and 2) From (4.7), $\mathbf{f}_{b,i} = \mathbf{A}_{4\times4}^{\top}\widehat{\mathbf{f}}_{b,i}$. The mixed-product and transpose property of the Kronecker product [51] rewrite $\mathbf{L}$ in (4.12) as follows.

$$\mathbf{L}(\widehat{\mathbf{F}_b}, \mathcal{C}) = \mathcal{S}[\widehat{\mathbf{f}}_{b,i} \otimes \widehat{\mathbf{f}}_{b,j}]^{\top}\mathbf{T}_{16\times10} \tag{B.3}$$

$$= \mathcal{S}[\mathbf{f}_{b,i} \otimes \mathbf{f}_{b,j}]^{\top}(\mathbf{A}_{4\times4} \otimes \mathbf{A}_{4\times4})^{\top}\mathbf{T}_{16\times10}, \tag{B.4}$$

$$\mathbf{L}(\mathbf{F}_b, \mathcal{C}) = \mathcal{S}[\mathbf{f}_{b,i} \otimes \mathbf{f}_{b,j}]^{\top}\mathbf{T}_{16\times10} \tag{B.5}$$

From $\text{rank}(\mathbf{XY}) = \min\{\text{rank}(\mathbf{X}), \text{rank}(\mathbf{Y})\}$, $\mathbf{L}$ has the same rank for $\mathbf{F}_b$ and $\widehat{\mathbf{F}_b}$ since $\text{rank}(\mathbf{A}_{4\times4} \otimes \mathbf{A}_{4\times4}) = 16$ and $\text{rank}(\mathbf{T}_{16\times10}) = 10$. Lemma 2 is proven by (B.5). □

Lemma 1 and 2 enable us to analyze the rank of the linear system $\mathbf{L}(\widehat{\mathbf{F}}_b, \mathcal{C})$ in (4.13) using a true $\mathbf{F}_b$ instead of $\widehat{\mathbf{F}}_b$. When $\mathcal{C} = \mathcal{C}^*_{\tau=1}$, $\mathbf{L}$ has the following properties.

Proposition 1. *When* $\mathcal{C} = \mathcal{C}^*_{\tau=1}$, $\text{rank}(\mathbf{L}) \leq 9$.

Proof. Let $\mathbf{f}_i = [x_i, y_i, z_i]^\top$. The column expansion of the matrix $\mathcal{S}[\mathbf{f}_{b,i}^\top \otimes \mathbf{f}_{b,i}^\top]$ with $\mathbf{f}_{b,i}^\top = [\mathbf{f}^\top\ 1]$ shows that it has six duplicated columns in (B.6) and thus its maximum rank is bounded to 10 in (B.7).

$$\begin{aligned}
&\operatorname{rank}(\mathcal{S}[\mathbf{f}_{b,i}^\top \otimes \mathbf{f}_{b,i}^\top]) = \operatorname{rank}(\mathcal{S}[(\mathbf{f}_i \otimes \mathbf{f}_i)^\top \ \ \mathbf{f}_i^\top \ \ \mathbf{f}_i^\top \ \ 1]) && \text{(B.6)}\\
&= \operatorname{rank}(\mathcal{S}[x_i^2 \ \ y_i^2 \ \ z_i^2 \ \ x_iy_i \ \ y_iz_i \ \ z_ix_i \ \ x_i \ \ y_i \ \ z_i \ \ 1]) && \text{(B.7)}\\
&= \operatorname{rank}(\mathcal{S}[x_i^2 \ \ y_i^2 \ \ z_i^2 \ \ x_iy_i \ \ y_iz_i \ \ z_ix_i \ \ x_i \ \ y_i \ \ z_i]) && \text{(B.8)}\\
&\leq 9 && \text{(B.9)}
\end{aligned}$$

Since all the loads in $\mathcal{C}_{\tau=1}^*$ are subject to $x_i^2 + y_i^2 + z_i^2 - 1 = 0$, the first three columns and the last column in (B.7) are linearly dependent and thus the maximum rank decreases by one. From Lemma 2, $\operatorname{rank}(\mathbf{L}) \leq 9$ for $\mathcal{C}_{\tau=1}^*$. □

Proposition 2. *When $\mathcal{C} = \mathcal{C}_{\tau=1}^*$, only one root of $f(\alpha)$ satisfies the q-constraint.*

Proof. $\mathbf{Q} = \mathbf{A}_{4\times3}\mathbf{A}_{4\times3}^\top = \mathbf{A}_{4\times4}\operatorname{diag}(1,1,1,0)\,\mathbf{A}_{4\times4}^\top$ from (4.11). Suppose $\mathbf{A}_{4\times4} = \mathbf{I}$. From $\widehat{\mathbf{f}}_{b,i} = \mathbf{f}_{b,i} = [x_i\ y_i\ z_i\ 1]^\top$ and $\mathcal{C}_{\tau=1}^*$ ($x_i^2 + y_i^2 + z_i^2 = 1$ for all i), a linear system $\mathbf{Lq} = \mathbf{1}$ is satisfied by the following particular and null solution:

$$\mathbf{L} = \mathcal{S}[x_i^2\ 2x_iy_i\ 2z_ix_i\ 2x_i\ y_i^2\ 2y_iz_i\ 2y_i\ z_i^2\ 2z_i\ 1] \tag{B.10}$$

$$\mathbf{q}_p = \frac{1}{4}[\,1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 3\,]^\top \tag{B.11}$$

$$\mathbf{q}_n = \frac{1}{2}[\,1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ \text{-1}]^\top \tag{B.12}$$

Then, $\mathbf{q} = \mathbf{q}_p + \alpha\mathbf{q}_n$ is equal to a symmetric matrix $\mathbf{Q}$ s.t.

$$\mathbf{Q} = \begin{bmatrix} q_1 & & & \\ q_2 & q_5 & & \\ q_3 & q_6 & q_8 & \\ q_4 & q_7 & q_9 & q_{10} \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1{+}2\alpha & & & \\ 0 & 1{+}2\alpha & & \\ 0 & 0 & 1{+}2\alpha & \\ 0 & 0 & 0 & 3{-}2\alpha \end{bmatrix} \tag{B.13}$$

From $\det(\mathbf{Q}) = 0$ in (4.15a), this first q-constraint produces $f(\alpha) = (1+2\alpha)^3(3-2\alpha) = 0$ which has two real roots, $\alpha = 1.5$ or -0.5. Only $\alpha = 1.5$ is true because $\alpha = -0.5$ cannot produce a positive definite $\mathbf{Q}_{3\times3}$. This uniqueness of α holds for the multiplication of $\mathbf{Q}$ by any invertible matrix $\mathbf{A}_{4\times4}$. □

Regardless of how many measurements ($m \geq |\mathcal{C}|_{min}$) are collected in $\mathbf{Z}$, Propositions 1 and 2 claim that the accelerometer calibration based on the gravity magnitude should take a one-dimensional null space of $\mathbf{L}$ into account and a unique reconstruction exists for $\mathbf{S}_b$.

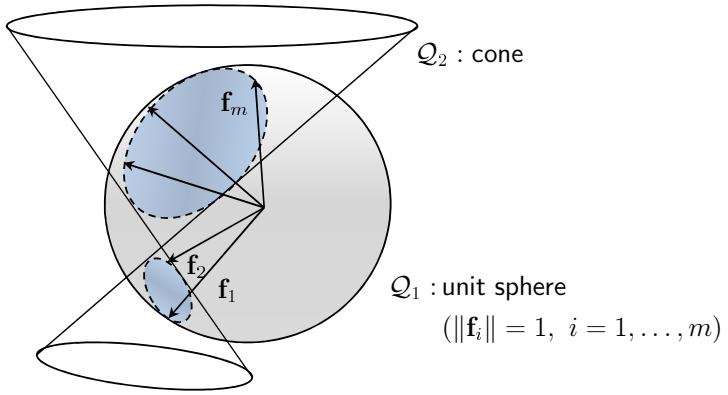
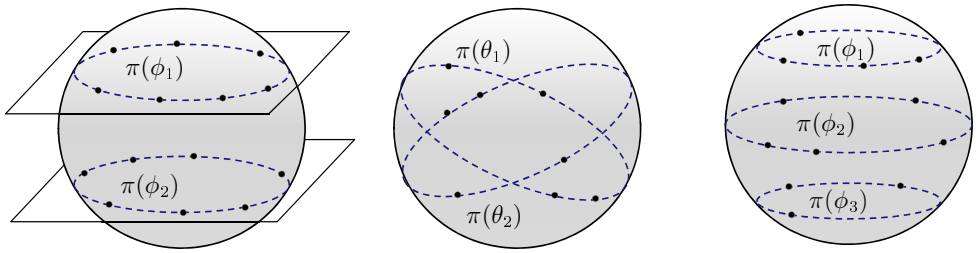


Figure B.1: Degenerate condition for calibrating loads $\mathbf{F}$ in $\mathcal{C}^*_{\tau=1}$: $\mathbf{F}$ is degenerate if $\mathbf{F}$ lies on the interchapter of $\mathcal{Q}_1$ (sphere) and $\mathcal{Q}_2$ (any other quadric, *e.g.*, cone).



(a) $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ (b) $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ (c) $\mathbf{F} \notin \mathcal{D}_\mathbf{F}$

Figure B.2: Examples of degenerate and non-degenerate calibrating load sets $\mathbf{F}$ in $\mathcal{C}^*_{\tau=1}$: (a) $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ when $\mathbf{F}$ on $\mathcal{Q}_2 : \pi(\phi_1)\pi(\phi_2)$ generated by two discrete roll angles ϕ, (b) $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ when $\mathbf{F}$ on $\mathcal{Q}_2 : \pi(\theta_1)\pi(\theta_2)$ generated by two discrete pitch angles θ, (c) $\mathbf{F} \notin \mathcal{D}_F$ when $\mathbf{F}$ on a non-quadric $\pi(\phi_1)\pi(\phi_2)\pi(\phi_3)$ generated by three discrete roll angles.

B.5 Degenerate load configuration

Proposition 1 shows the case where a particular load constraint set produces a null space in $\mathbf{L}$. We also investigate the case where a certain geometric distribution of loads $(\mathbf{f}_1, \ldots, \mathbf{f}_m)$ causes a null space $\mathcal{N}(\mathbf{L})$. If the dimension of a null-space is greater than one, *i.e.*, $\text{rank}(\mathbf{L}) < 9$, a linear system $\mathbf{L}$ is unsolvable and it leads to a failure in the factorization.

Definition 3. *$\mathcal{D}_\mathbf{F}$ is a set of degenerate load distributions $\mathbf{F} = (\mathbf{f}_1, \ldots, \mathbf{f}_m)$ that has* $\text{rank}(\mathbf{L}|\mathbf{F}_b, \mathcal{C}) < 9$.

Proposition 3. *When $\mathcal{C} = \mathcal{C}^*$, $\mathcal{N}(\mathbf{L}) \neq \emptyset$ if and only if a quadric $\mathcal{Q}$ exists for $\mathbf{F}$.*

Proof. Let $\mathbf{L}_\mathcal{S}$ be the stacked matrix in (B.7). $\mathcal{N}(\mathbf{L}) \neq \emptyset$ if a non-zero $\mathbf{a}$ exists for $\mathbf{L}_\mathcal{S}\mathbf{a} = \mathbf{0}$. For a given $\mathcal{C}^*$, $\mathbf{a}$ can be represented by a quadric $\mathcal{Q} \in \mathcal{R}^{4\times4}$: $\mathbf{f}_{b,i}^\top\, \mathcal{Q}(\mathbf{a})\, \mathbf{f}_{b,i} = 0$, *i.e.*, $a_0x_i^2 + a_1y_i^2 + a_2z_i^2 +$

$a_3x_iy_i + a_4y_iz_i + a_5z_ix_i + a_6x_i + a_7y_i + a_8z_i + a_9 = 0$ for all i. Hence, $\mathcal{N}(\mathbf{L}) \neq \emptyset$ is equivalent to the existence of a quadric $\mathcal{Q}$ represented by all the loads in $\mathbf{F}$. □

Proposition 4. *When $\mathcal{C} = \mathcal{C}^*_{\tau=1}$, $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ if and only if a non-sphere quadric $\mathcal{Q}_2$ exists for $\mathbf{F}$.*

Proof. The degeneracy condition, $\text{rank}(\mathbf{L}) < 9$, is the same where at least two linear independent vectors $(\mathbf{a}_1, \mathbf{a}_2, \ldots)$ exist such that $\mathbf{L}_\mathcal{S}\mathbf{a}_1 = \mathbf{L}_\mathcal{S}\mathbf{a}_2 = \cdots = \mathbf{0}$. Proposition 1 shows that $\mathcal{C}^*_{\tau=1}$ already has $\mathbf{a}_1 = [1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \text{-}1]^\top$ which is a unit sphere $\mathcal{Q}_1 : x_i^2 + y_i^2 + z_i^2 - 1 = 0$. Also, $\mathbf{a}_2$ represents another quadric $\mathcal{Q}_2$ which should be not a unit sphere. □

From Proposition 3, geometric properties of $\mathcal{D}_\mathbf{F}$ for an intra-relation constraint $\mathcal{C}^*$ can be analyzed via 3D quadric fitting. The dimension η of $\mathcal{N}(\mathbf{L})$ is equal to how many quadrics $(\mathcal{Q}_1, \ldots, \mathcal{Q}_\eta)$ can represent $\mathbf{F}$ simultaneously. For example, Proposition 4 for $\mathcal{C}^*_{\tau=1}$ is the case $\eta = 2$ when the loads in $\mathbf{F} \in \mathcal{D}^*_\mathbf{F}$ are distributed at the interchapter of a unit sphere $\mathcal{Q}_1$ and any other quadric $\mathcal{Q}_2$ like Fig. B.1. In essence, $\mathcal{D}_\mathbf{F}$ in $\mathcal{C}^*_{\tau=1}$ is determined by whether another possible quadric fitting of $\mathbf{F}$ exists other than the original constraint (geometrically a unit sphere) as the following remarks:

Remark 1. *When $\mathcal{C} = \mathcal{C}^*_{\tau=1}$, $\mathbf{F} \in \mathcal{D}_\mathbf{F}$ if $\mathbf{F}$ is on one or two cutting planes of a unit sphere.*

Remark 2. *When $\mathcal{C} = \mathcal{C}^*_{\tau=1}$, $\mathbf{F} \notin \mathcal{D}_\mathbf{F}$ if $\mathbf{F}$ is on three and more cutting planes of a unit sphere.*

The load set $\mathbf{F}$ distributed on two planes like Fig. B.2(a-b) is represented by $\boldsymbol{\pi} = \boldsymbol{\pi}_1\boldsymbol{\pi}_2 = (a_1x_i + b_1y_i + c_1z_i + d_1)(a_2x_i + b_2y_i + c_2z_i + d_2) = 0$. It is degenerate since this polynomial $\boldsymbol{\pi}$ can be represented by a non-sphere quadric $\mathcal{Q}_2$. Otherwise, if $\mathbf{F}$ is distributed on more than two planes like Fig. B.2(c), it is non-degenerate because a cubic polynomial $(\prod_{i=1}^{n\geq 3} \boldsymbol{\pi}_i = 0)$ cannot be represented by a quadric.

When an IMU is randomly oriented by hand in the calibration, it is far less likely that $\mathbf{F}$ forms a non-sphere quadric. However, attention should be paid when pivotal devices like a tripod are used. One should make sure to generate at least three roll or pitch angles during measurement collection.

Bibliography

[1] A. Puri, "A survey of unmanned aerial vehicles for traffic surveillance," Department of Computer Science and Engineering, University of South Florida, Tech. Rep., 2004.

[2] D. Floreano, J.-C. Zufferey, M. Srinivasan, and C. Ellington, *Flying Insects and Robots*. Springer-Verlag, 2009.

[3] Red Bull Air Race, *International Series of Air Races*, 2011. [Online]. Available: http://www.redbullairrace.com

[4] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, "A complementary filter for attitude estimation of a fixed-wing uav," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, Sept 2008, pp. 340–345.

[5] W. Premerlani and P. Bizard, *Direction Cosine Matrix IMU: Theory*, Mar. 2009. [Online]. Available: http://gentlenav.googlecode.com/files/DCMDraft2.pdf

[6] B. Taylor, C. Bil, and S. Watkins, "Horizon sensing attitude stabilisation: A VMC autopilot," in *18th International UAV Systems Conference*, 2003.

[7] S. M. Ettinger, M. C. Nechyba, P. G. Ifju, and M. Waszak, "Vision-guided flight stability and control for micro air vehicles," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2002, pp. 2134–2140.

[8] S. Todorovic, M. C. Nechyba, and P. G. Ifju, "Sky/ground modeling for autonomous MAV flight," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, May 2003, pp. 1422–1427.

[9] T. Cornall, G. Egan, and A. Price, "Aircraft attitude estimation from horizon video," *Electronic Letters*, vol. 42, no. 12, 2006.

[10] C. Demonceaux, P. Vasseur, and C. Pégard, "Omnidirectional vision on UAV for attitude computation," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2006, pp. 2842–2847.

[11] D. Dusha, W. Boles, and R. Walker, "Attitude estimation for a fixed-wing aircraft using horizon detection and optical flow," in *9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications*, Dec 2007, pp.

485–492.

[12] P. Denis, J. H. Elder, and F. J. Estrada, "Efficient edge-based methods for estimating manhattan frames in urban imagery," in *European Conf. on Computer Vision*, 2008, pp. 197–210.

[13] J. A. Shufelt, "Performance evaluation and analysis of vanishing point detection techniques," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 21, no. 3, pp. 282–288, 1999.

[14] M. E. Antone and S. Teller, "Automatic recovery of relative camera rotations for urban scenes," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Sept 2000.

[15] C. Demonceaux, P. Vasseur, and C. Pégard, "UAV attitude computation by omnidirectional vision in urban environment," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2007.

[16] J.-C. Bazin, I. Kweon, C. Demonceaux, and P. Vasseur, "UAV attitude estimation by vanishing points in catadioptric images," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2008.

[17] D. Gebre-Egziabher, G. Elkaim, J. Powell, and B. Parkinson, "A gyro-free quaternion-based attitude determination system suitable for implementation using low cost sensors," in *IEEE Symposium on Position Location and Navigation*, 2000.

[18] Y. Li, K. Zhang, C. Roberts, and M. Murata, "On-the-fly GPS-based attitude determination using single- and double-differenced carrier phase measurements," *GPS Solutions*, vol. 8, no. 2, pp. 93–102, 2004.

[19] N. Metni, J.-M. Pflimlin, T. Hamel, and P. Soueres, "Attitude and gyro bias estimation for a flying UAV," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2005, pp. 1114–1120.

[20] A. M. Eldredge, "Improved state estimation for miniature air vehicles," Master's thesis, Brigham Young University, Provo, Utah, Tech. Rep., Dec.

[21] M. Hwangbo and T. Kanade, "Visual-inertial UAV attitude estimation using urban scene regularities," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, Sept 2011, pp. 340–345.

[22] S. T. Barnard, "Interpreting perspective images," *Artificial Intelligence*, vol. 21, pp. 435–462, 1984.

[23] G. Welch and G. Bishop, "An introduction to the kalman filter," TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Tech. Rep., 1995.

[24] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. of the ACM*, vol. 24, p. 381395, 1981.

[25] C. Rother, "A new approach for vanishing point detection in architectural environments,"

Journal Image and Vision Computing, Special Issue on BMVC 2000, vol. 20, no. 9–10, pp. 647–656, 2002.

[26] D. G. Aguilera, J. G. Lahoz, and J. F. Codes, "A new method for vanishing points detection in 3D reconstruction from a single view," in *Proceedings of the ISPRS Commission V*, 2005, pp. 197–210.

[27] H. Wildenauer and M. Vincze, "Vanishing point detection in complex man-made worlds," in *IEEE Int'l Conf. on Image Analysis and Processing*, 2007, pp. 615–622.

[28] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[29] R. M. Rogers, *Applied Mathematics in Integrated Navigation Systems (AIAA Education Series)*, 2nd ed. AIAA, 2003.

[30] J. J. Hall and R. L. W. II, "Inertial measurement unit calibration platform," *Journal of Robotic Systems*, vol. 17, no. 11, pp. 623–632, 2000.

[31] J. W. Diesel, "Calibration of a ring laser gyro inertial navigation system," in *13th Biennial Guidance Test Symposium*, Oct. 1987.

[32] E. Nebot and H. Durrant-Whyte, "Initial calibration and alignment of low-cost inertial navigation," *Journal of Robotic Systems*, vol. 16, no. 2, pp. 81–92, 1999.

[33] A. Kim and M. F. Golnaraghi, "Initial calibration of an inertial measurement unit using an optical position tracking system," in *IEEE Position Location and Navigation Symposium (PLANS 2004)*, Apr. 2004.

[34] Z. Dong, G. Zhang, Y. Luo, C. C. Tsang, G. Shi, S. Y. Kwok, W. Li, P. Leong, and M. Y. Wong, "A calibration method for mems inertial sensors based on optical tracking," in *2nd IEEE Int'l Conf. on Nano/Micro Engineered and Molecular Systems*, 2007.

[35] P. Lang and A. Pinz, "Calibration of hybrid vision / inertial tracking system," in *Proc. 2nd Workshop on Integration of Vision and Inertial Sensors (INERVIS'05)*, 2005.

[36] J. Lobo and J. Dias, "Relative pose calibration between visual and inertial sensors," *International Journal of Robotics Research*, vol. 23, no. 12, pp. 1157–1195, 2004.

[37] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. on Robotics*, vol. 24, no. 5, pp. 1143–1156, 2008.

[38] J. Kelly and G. S. Sukhatme, "Fast relative pose calibration for visual and inertial sensors," in *11th Int'l Symp. Experimental Robotics (ISER'08)*, 2008.

[39] J. D. Hol, T. B. Schön, and F. Gustafsson, "Modeling and calibration of inertial and vision

sensors," *International Journal of Robotics Research*, vol. 29, no. 2–3, pp. 231–244, 2010.

[40] S. Y. Cho and C. G. Park, "A calibration technique for a redundant IMU containing low-grade inertial sensors," *ETRI Journal*, vol. 27, no. 4, pp. 418–426, 2005.

[41] A. Lai, D. A. James, J. P. Hayes, and E. C. Harvey, "Semi-automatic calibration technique using six inertial frames of reference," in *Proceedings of the SPIE*, vol. 5274, 2004, pp. 531–542.

[42] I. Skog and P. Handel, "Calibration of a MEMS inertial measurement unit," in *XVII IMEKO World Congress*, Brazil, Sep. 2006.

[43] Z. F. Syed, P. Aggarwal, C. Goodall, X. Niu, and N. El-Sheimy, "A new multi-position calibration method for MEMS inertial navigation systems," *Measurement Science and Technology*, vol. 18, no. 7, pp. 1897–1907, 2007.

[44] S. P. Won and F. Golnaraghi, "A triaxial accelerometer calibration method using a mathematical model," *IEEE Transaction on Instrumentation and Measurement*, vol. 59, no. 8, pp. 2144–2153, 2010.

[45] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[46] R. M. Voyles, J. D. Morrow, and P. K. Khosla, "Shape from motion approach to rapid and precise force/torque sensor calibration," in *Int'l Mechanical Eng Congress and Exposition*, Nov. 1995, pp. 2171–2175.

[47] ——, "Including sensor bias in shape from motion calibration and sensor fusion," in *IEEE Multisensor Fusion and Integration Conference*, Dec. 1996.

[48] T. Kanade and D. Morris, "Factorization methods for structure from motion," *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, vol. 356, pp. 1153–1173, 1998.

[49] K. Kim, Y. Sun, R. M. Voyles, and B. J. Nelson, "Calibration of multi-axis mems force sensors using the shape-from-motion method," *IEEE Sensors Journal*, vol. 7, no. 3, pp. 344–350, 2007.

[50] M. Hwangbo and T. Kanade, "Factorization-based calibration method for mems inertial measurement unit," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, San Hose, USA, Apr. 2008.

[51] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, 1996.

[52] Analog Device, *ADXL203 Precision Dual-Axis iMEMS Accelerometer*, Mar. 2006. [Online]. Available: http://www.analog.com/en/sensors/inertial-sensors/adxl203/products/product.html

[53] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[54] R. Hartley, "Self-calibration from multiple views with a rotating camera," in *European Conf. on Computer Vision*, 1994, pp. 471–478.

[55] ——, "Self-calibration of stationary cameras," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 5–23, 1997.

[56] K. S. Arun, T. S. Huang, and S. D. Bolstein, "Least-squares fitting of two 3-d point sets," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, pp. 698–700, 1987.

[57] M. Hwangbo, J.-S. Kim, and T. Kanade, "Inertial-aided KLT feature tracking for a moving camera," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2009, pp. 1909–1916.

[58] W. Sepp and S. Fuchs, *DLR CalLab and CalDe: The DLR Camera Calibration Toolbox*, 2008. [Online]. Available: http://www.robotic.dlr.de/callab/

[59] D. Liebowitz and A. Zisserman, "Metric rectification for perspective images of planes," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Jun. 1998, pp. 482–488.

[60] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment – a modern synthesis," in *Vision Algorithms: Theory and Practice*, ser. Lecture Notes in Computer Science. Springer Berlin, 2000, vol. 1883, pp. 153–177.

[61] B. Sinopoli, M. Micheli, G. Donata, and T. Koo, "Vision based navigation for an unmanned aerial vehicle," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2001.

[62] R. Zapata and P. Lepinay, "Flying among obstacles," in *European Workshop on Advanced Mobile Robots*, 1999.

[63] J. Sasiadek and I. Duleba, "3d local trajectory planner for uav," *Journal Journal of Intelligent and Robotic Systems*, vol. 29, no. 2, pp. 191–210, Oct. 2000.

[64] C. Reynolds, "Steering behaviors for autonomous characters," in *Game Developers Conference*, 1999.

[65] J. Go, T. D. Vu, and J. Kuffner, "Autonomous behaviors for interactive vehicle animations," *Graphics Models*, vol. 68, no. 2, pp. 90–112, 2006.

[66] J. F. Canny, *The Complexity of Robot Motion Planning*. MIT, Cambridge, 1988.

[67] J.-C. Latombe, *Robot Motion Planning*. Kluwer Academics Publication, 1991.

[68] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms and Implementations*. MIT Press, 2005.

[69] S. M. LaValle, *Planning Algorithms*. Cambridge University Press (also available at http://msl.cs.uiuc.edu/planning/), 2006.

[70] A. G. Barto, S. J. Bradtke1, and S. P. Singh2, "Learning to act using real-time dynamic programming," *Artificial Intelligence*, vol. 72, no. 1-2, pp. 81–138, 1995.

[71] K. Tarabanis, P. Allen, and R. Tsai, "A survey of sensor planning in computer vision," *IEEE Trans. on Robotics and Automation*, vol. 11, no. 1, pp. 86–104, 1995.

[72] P. Svestka and M. Overmars, "Coordinated motion planning for multiple car-like robots using probabilistic roadmaps," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, May 1995, pp. 1631–1636.

[73] D. Ferguson, M. Likhachev, and A. Stentz, "A guide to heuristic-based path planning," in *Proceedings of ICAPS Workshop on Planning under Uncertainty for Autonomous Systems, AAAI*, June 2005.

[74] E. Frazzoli, M. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA Journal of Guidance, Control and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002.

[75] ——, "A hybrid control architecture for aggressive maneuvering of autonomous helicopters," in *IEEE Conf. on Decision and Control*, 1999.

[76] O. Yakimenko, "Direct method for rapid prototyping of near-optimal aircraft trajectories," *AIAA Journal of Guidance, Control and Dynamics*, vol. 23, no. 5, pp. 865–875, 2000.

[77] V. Gavrilets, E. Frazzoli, and B. Mettler, "Aggressive maneuvering of small autonomous helicopters: A human-centered approach," *International Journal of Robotics Research*.

[78] C. Dever, B. Mettler, E. Feron, J. Popovic, and M. McConley, "Trajectory interpolation for parametrized maneuvering and flexible motion planning of autonomous vehicles," *AIAA Journal of Guidance Control and Dynamics*, vol. 29, no. 2, pp. 1–28, 2006.

[79] B. Mettler and Z. Kong, "Receding horizon trajectory optimization with a finite-state value function approximation," in *American Control Conference (ACC)*, 2008, pp. 3810–3816.

[80] M. Hwangbo, J. Kuffner, and T. Kanade, "Efficient two-phase 3d motion planning for small fixed-wing UAVs," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, Apr 2007, pp. 1035–1041.

[81] R. W. Beard, *An Introduction to Autonomous Miniature Air Vehicles*. Unpublished, 2006.

[82] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with

prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, pp. 497–516, 1957.

[83] X.-N. Bui, P. Soueres, J.-D. Boissonnat, and J.-P. Laumond, "Shortest path synthesis for dubins nonholonomic robot," in *Proc. IEEE Int'l Conf. on Robotics and Automation*, 1994.

[84] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed. Athena Scientific, Belmont, MA, 2000.

[85] R. Alterovitz, M. Branicky, and K. Goldberg, "Constant-curvature motion planning under uncertainty with applications in image-guided medical needle steering," in *In Workshop on the Algorithmic Foundations of Robotics*, 2006.

[86] D. Nelson, D. Barber, T. McLain, and R. Beard, "Vector field path following for miniature air vehicles," *IEEE Trans. on Robotics*, vol. 23, no. 3, pp. 519–529, Jun 2007.

[87] K. Wu, E. Otoo, and K. Suzuki, "Optimizing two-pass connected component labeling algorithms," *Pattern Analysis and Application*, vol. 12, no. 2, pp. 117–135, Feb 2009.

[88] E. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.