Faculty of Aerospace Engineering

Vision-based Autonomous Drone Racing in GPS-denied Environments

M.M.O.I Ozo

March 24, 2018

TU Delft Delft University of Technology

Challenge the future

Vision-based Autonomous Drone Racing in GPS-denied Environments

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

M.M.O.I Ozo

March 24, 2018

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Vision-based Autonomous Drone Racing in GPS-denied Environments"** by **M.M.O.I Ozo** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: March 24, 2018

Readers:

Dr. G. C. H. E. de Croon

Dr.ir. C. C. de Visser

Dr.ir. J. Alonso-Mora

Acronyms

EKF	Extended Kalman Filter
EKF VIO	EKF Visual Inertial Odometry
FOE	Focus of expansion
FPV	First Person View
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INDI	Incremental Nonlinear Dynamic Inversion
IROS	International Conference on Intelligent Robots and Systems
LIDAR	LIght Detection And Ranging
MAV	Micro Air Vehicle
MEMS	Micro Electrical Mechanical Sensor
MSCKF	Multi State Constraint Kalman Filter
NDI	Nonlinear Dynamic Inversion
P3p	Perspective-3-point
P4p	Perspective-4-point
PID	Proportional Integral Derivative
Pnp	Perspective-n-point
PRM	Probabilistic Road Maps
PTAM	Parralel Tracking And Mapping
RMSE	Root Mean Squared Error
RRT	Rapidly exploring Random Tree
SLAM	Simultinous Localization and Mapping
TTC	Time-To-Contact
UKF	Uncented Kalman Filter

Introduction

Micro Air Vehicles or drones are generating increased interest in recent years. It is expected that these small flying robots will become ever more important in a wide variety of applications. Current MAVs are however still lacking in autonomous flight capabilities. This task is challenging due to the limited sensor quality and processing power on-board of such a small platform. Especially in indoor and urban environments, when a reliable GPS signal is unavailable. Another major limitation of current MAVs is the endurance. Therefore in general it can be said that to increase range in autonomous flight, the drone has to fly faster.

A recent trend is so called First Person View (FPV) drone racing. Human pilots have to control their drone through a track of gates, while viewing a live stream of the on-board camera. The maneuverability and speed of these human pilots still outperforms the best autonomous drones. Drone racing has much in common with autonomous high speed drone flight. The races are often held in an indoor environment, blocking the GPS signal and requiring alternative navigation methods. Also the planning and executing of a time-optimal trajectory is part of the challenge. Therefore to promote the developments in the area of autonomous high speed MAV flight the International Conference on Intelligent Robots and Systems (IROS) is organizing a yearly autonomous drone race since 2016. The goal is to fly a known track of gates in the right order and as fast as possible. The race is indoor, therefore no GPS can be used and other external position systems are not allowed. The drone has to navigate purely based on onboard sensors.

The current thesis describes a system which is capable of autonomous drone racing, while only using on-board sensors and processing power. The system has to operate in an indoor environment and on a small computationally constrained MAV. Therefore the system uses a computationally lightweight visual-inertial navigation method. The computer-vision, state estimation and control methods, together with a performance analysis of the system are described in detail in a scientific article. After that the preliminary thesis report is also included as a reference.

Vision-based Autonomous Drone Racing in GPS-denied Environments

Student: M.M.O.I. Ozo
Supervisors: ir S. Li, ir C. de Wagter, Dr G.C.H.E. de Croon

***Abstract*— High-speed autonomous flight of Micro Air Vehicles has gained much attention in recent years. However, flight in complex GPS-denied environments still poses a serious challenge. One scenario which contains these elements is drone racing, where pilots have to fly complex tracks at high speed, often in an indoor environment. In this work we therefore present an MAV capable of autonomously flying such a drone race track. The system has to operate in a GPS-denied environment, hence a visual navigation method is employed. Position is recovered from gate detections based on a novel least-squares method, while heading is estimated using an optimization based method. Experiments show that both methods have a higher accuracy than the standard P3P pose estimation method. Furthermore, a state estimation filter is designed to fuse the visual measurements with IMU measurements, by using an EKF with drag based prediction model. For high-level control different motion primitives are linked, which allow the MAV to fly the track without having a detailed on-board map. The overall approach does not rely on SLAM or Visual odometry, which results in low computational complexity. Also, it does not rely on downward optical flow velocity measurements, which enables it to work even in low texture environments.**

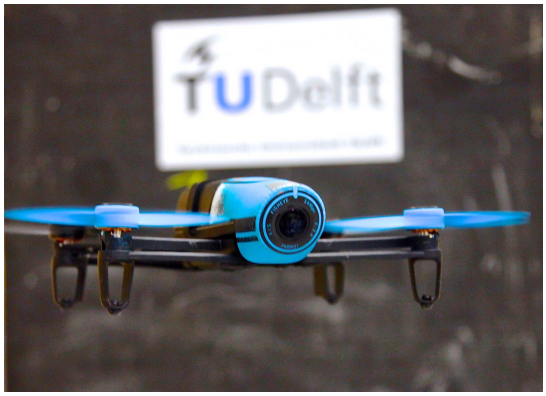


Fig. 1: Parrot Bebob™ Consumer MAV which is used as a test platform

NOMENCLATURE

$j_{x,y}$	Pixel-coordinate of point in image frame
$f_{x,y}$	Camera focal length in pixels
$c_{x,y}$	Camera principal point
$\mathbf{v_i}$	Bearing vector of 3D point in body frame
R_c^b	Camera to body rotation matrix
R_b^w	Body to world rotation matrix
$\mathbf{n_i}$	Bearing vector of 3D point in world frame
$\mathbf{x_i}$	3D light ray from gate corner point
$\mathbf{p_i}$	Gate corner point position
λ	Parameter of 3D line equation
$\mathbf{t}$	Translation vector of body frame w.r.t. the world frame, expressed in world frame
D	Point to line distance
g_s	Gate size
x_h	Histogram gate distance
y_h	Histogram lateral position
d	Distance from gate to vehicle
h	Vehicle altitude above ground
a	Gate approach angle
ϕ	Roll Euler angle
ψ	Yaw Euler angle
θ	Pitch Euler angle
R_w^b	World to body rotation matrix
R_b^c	Body to camera rotation matrix
s_i	gate corner point detection in image frame
B	Back-projection error
$a_{x,y,z}$	Accelerometer specific force measurements
$F_{dx,y}$	External force in body x- and y-axis
$v_{bx,y,z}$	Velocity in body frame
$k_{x,y}$	Drag coefficients in body x- and y-axis
$b_{x,y,z}$	Accelerometer sensor bias
m	Vehicle mass
$\mathbf{x}$	State vector
$\mathbf{u}$	Input vector
$\mathbf{y}$	Output vector
$\mathbf{w}$	Process noise vector
$\mathbf{m}$	Measurement noise vector
p, q, r	Rotational rates of body frame w.r.t. the inertial frame
$\mathbf{v}$	Velocity vector
g	gravity vector
k	Heading filter gain
$\mathbf{F_{ext}}$	External force vector
Ω	Rotational rates vector
T	Thrust force in body frame
$\mathbf{D}$	Acceleration due to drag
R_b^f	Body to f-frame rotation matrix

I. INTRODUCTION

Micro Air Vehicles (MAVs), especially from the quadrotor type have applications in various fields, including aerial photography, industrial inspection, search and rescue[16], agriculture and law enforcement. Currently most MAVs are piloted manually. However, to improve the efficiency and scalability of MAV operations it is necessary that MAVs can operate autonomously[13]. This task can be challenging due to the absence of a reliable GPS signal in indoor environments and urban canyons. Also, the quality of on-

board sensors such as the Inertial Measurement Unit (IMU) is often low. Additionally, a typical mission of an MAV has a time constraint, such as in a search and rescue mission or due to limited endurance of the MAV in general. Therefore, these factors require an MAV to be capable of high speed autonomous flight.

A recent trend in manual MAV flight is so called FPV drone racing. Human pilots race each other by controlling the drone via a video stream. A typical drone race has all components of high speed MAV flight, such as a time constraint, a race track which requires aggressive maneuvers and the absence of a GPS signal when flying indoor. To promote the developments in high speed autonomous MAV flight, now also autonomous drone races are being organized. An example of this is the annual drone race that is part of the International Conference on Intelligent Robots and Systems(IROS) since 2016. The current work therefore describes an MAV system for autonomous drone racing. The system uses visual-inertial navigation to allow for flights in GPS-denied environments.

In the literature various examples of high speed autonomous MAV flight were performed by [11], [6], [2], [14] and [10]. In[11] impressive results were obtained of quadrotor MAVs flying aggressive trajectories. They exploit the differential flatness property of the quadrotor model to design control laws and to iteratively design trajectories. It has to be noted however that these trajectories were generated off-board and a high-precision indoor positioning system was used for navigation. In[2] a solution is proposed with both on-board sensing and processing. LIDAR depth information is fused with IMU data for state estimation. Trajectories are flown in a known obstacle dense environment. Way-points are generated by RRT after which polynomial spline trajectories through these way-points are optimized for time. The disadvantage of LIDAR is that the sensors tend to be relatively heavy and power consuming, compared to other sensors such as a camera and IMU. In[14] a small quadrotor is equipped with a stereo camera for navigation purposes. Their approach consists of a high-rate monocular pose estimation algorithm, combined with low-rate scale recovery step from the second camera. The camera pose information is fused in a Kalman filter framework with IMU data. A major drawback of this approach is that the scale recovery is dependent from depth information. Therefore, if the depth in a scene is outside of the maximum stereo vision range, scale drift will not be corrected for. Next to stereo vision also monocular vision methods can be used for indoor navigation. This has as an advantage that it can be used on-board of even the smallest MAVs. An example of this is shown in[4] where a small quadrotor performs aggressive high-speed flight through a narrow gap. Visual data from the gap detection is fused with IMU data for pose estimation. It has to be realized though that only a single gap flight is performed at a time, after which the drone is stabilized to a hovering state. This does not test the system performance over longer periods of time.

One example of an autonomous drone racing system is given in [8]. This is the winning system of the 2016 IROS drone race. Here a gate-based visual servoing approach is used together with optical flow velocity estimation. Race gates are detected using a color based segmentation method and stereo-depth vision. Global navigation is performed by means of leg-by-leg planning, taking into account the known map of the track. The use of optical flow means that an abundantly textured floor surface is required for feature tracking. However, in many situations the floor surface is very smooth causing this method to fail. Also, the gate detection algorithms that are used require a high-end embedded GPU board to run on. This limits the implementation on smaller and more lightweight MAVs.

The current work aims to solve the drone racing problem as computationally efficient as possible. By limiting the required processing power, the methods can potentially be implemented on even the smallest, computationally constrained MAVs. Therefore computationally expensive SLAM[2], visual odometry[14] and gate detection methods cannot be used. Instead, visual navigation will only rely on the detection of gates in the drone race track, combined with IMU and sonar data in a lightweight filtering approach. The system is therefore not relying on optical flow velocity measurements. Also in the current drone racing scenario longer flights will be performed without time for stabilization. Compared to the relative short experiments of [2] , [14] and [4] we investigate the performance of the system on a longer flight of several minutes. We do this in order to verify if the state estimation can cope with long exposure to aggressive maneuvers.

The remainder of this work is structured as follows. First in section II a high-level overview is given of the drone racing problem and the strategy which is employed for solving it. Section III gives a description of the gate detection method and how these detections are used for pose estimation. Section IV shows how visual pose estimates are fused with inertial and sonar measurements. Furthermore, the control system design is discussed in section V. Then in section VI the performance of the previously described methods is evaluated, while section VII finally holds the conclusions and recommendations.

II. Task and system overview

In this work two different tracks are considered. One short two-gate track as described in figure 2 and a longer track, which represents a more realistic racing scenario. The short track is equipped with an Optitrack motion-capture system, which provides ground-truth position, speed and pose data. For both tracks the MAV has to pass the gates in the correct sequence and as fast as possible. In the current drone racing scenario the flight takes place in a GPS-denied environment. Therefore, computer vision is used for detecting gates in the race track. The shape of a gate in image frame together with the known gate geometry will then be used to determine the position of the MAV with respect to the current gate in view. Global navigation is performed without building a detailed map of the environment, but rather by linking multiple standard maneuvers. On-board of the MAV a sequence is

stored which describes how far to fly after passing a gate, how much to turn and where the next gate will be located approximately. One advantage of the overall approach is that it is computationally lightweight. Also, by only using gate detections as visual feedback the method will work even in low texture environments.

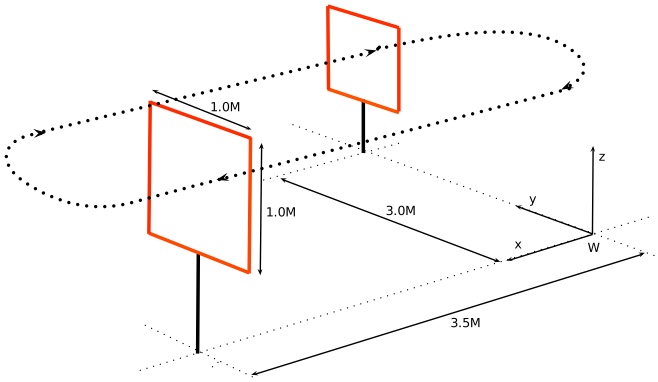


Fig. 2: Two-gate oval race track

In figure 3 a description of the drone racing autopilot system is given. Images from the on-board camera are used to detect the gates in the track. Gate detections are then processed to recover position and heading data with respect to the current gate in view. Pitch and roll attitude is estimated using IMU data in a state observer approach similar to[15]. These pitch and roll estimates are used in the visual pose estimation block as well as in the state estimation block. An Extended Kalman Filter(EKF) estimates position and accelerometer biases, based on raw IMU measurements, attitude and visual pose estimates. Position estimates are then fed into the controller which can switch from closed loop mode, when there is a gate detected, to an open loop mode when no gate is detected.

The platform that is used for this research is a 2014 Bebop MAV from Parrot. The Bebop is a small 400 gram computationally constrained consumer quadrotor. The platform is equipped with a dual-core ARM-processor capable of basic computer vision. Sensors include a front-facing camera with 180 degrees fish-eye lens, as well as a MEMS inertial measurement unit and sonar altimeter. Note that the camera image available for processing is only 160x315 pixels and has a rate of 15 Hz. The MAV is further only modified by replacing the standard autopilot firmware with the open source Paparazzi autopilot. The Bebop was chosen because of its low cost and high durability. However, the drone racing system would function on any quadrotor MAV capable of basic computer vision and equipped with front-facing camera, IMU and sonar.

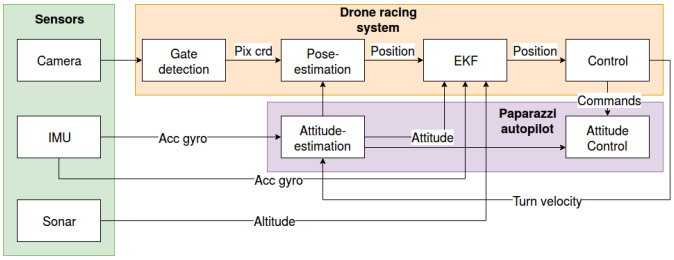


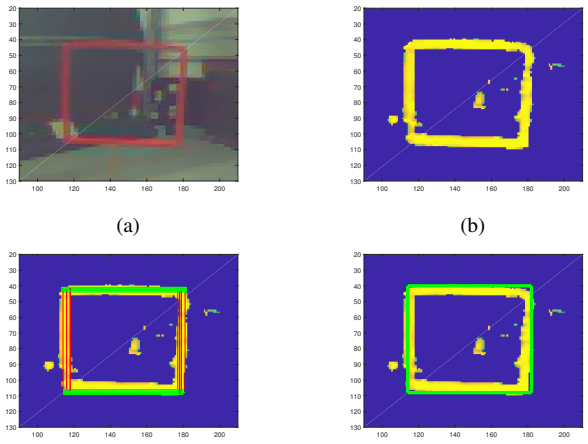
Fig. 3: System architecture

III. Visual Sensing

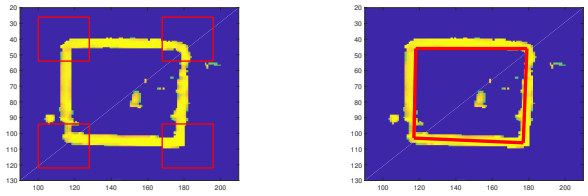
A. Gate detection

Gate detections are used as a visual reference for position and heading determination. The task of detecting these gates in the current scenario amounts to detecting orange square shaped objects, which are open in the middle. This may be accomplished by multiple different computer vision methods, such as Viola and Jones[17], Hough transform[7] and deep learning[12]. However, in order to limit processing power to a minimum, an alternative method is used instead. The current method for detection of square gates in the image consists of a two-step approach where an initial rough detection is further refined. The rough estimate is made by random sampling the original image(4a) for the target color(4b), after which a pixel search is performed, first in vertical direction and then in horizontal direction(4c). If two or more gate segments are found, the rest of the segments are calculated to form a square(4d). To further improve this approximation the corner point locations are refined by finding the centroid of the patch around each rough corner (3e). This results in a polygon(3f), which is later used for pose estimation. When a previous image frame contained a valid detection, this detection is used as an initial rough estimate in the next frame.

Examples of gate detections are given in figure 4. The detections vary in accuracy from a high quality detection in 4a, to a low quality detection with a significant outlier in 4d.

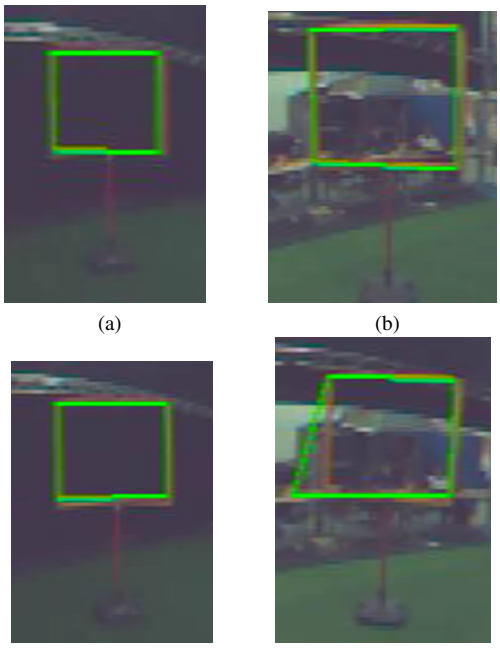


(a) (b) (c) (d)



(e) (f)

Fig. 3: Gate detection process



(a) (b)

(c) (d)

Fig. 4: Gate detection examples

When the MAV is close to the gate and only a part of the gate is in view, a second detection method is employed. This method is able to detect the sides of the gate by searching the histogram for peeks in the color value of the gate as in (5). Here the histogram represents the accumulated color value in the image columns. The side bar positions in pixel coordinates can then be converted to a position estimate, enabling position feedback in the final approach to a gate.

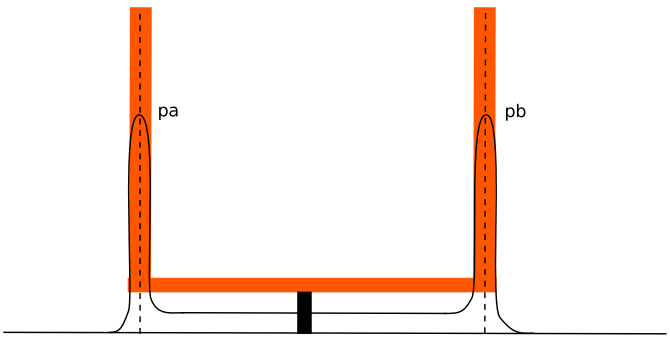


Fig. 5: Histogram gate side detection

B. Pose estimation

The gate detection together with the known geometry of the gate, can be used to estimate the position of the MAV. The problem of determining the camera pose from a set of image points, given the known 3D locations of these points is known as the Perspective-n-Point problem(PnP). A solution exists for 3,4,8 and more points, where for the square gate detection problem only the 3- and 4-point methods are possible. Both 3- and 4-point methods have multiple solutions, such as[5]and[9]. Small errors in the gate detection process can amplify into large errors in pose estimate. Therefore, these methods are generally implemented in a RANSAC scheme to reject noise and outliers. However, the fact that only four corner points are available on one gate limits the effectiveness of such a scheme.

Therefore, alternative methods are used in our system to improve performance in the presence of noise and outliers. The approach consists of separating the pose estimation problem in a position estimation problem and a heading estimation problem. The heading method then employs the initial estimate of the position method.

For both methods first a distortion correction step is applied as in[3]. The camera is therefore further treated as a pinhole camera.

1) Position estimation least squares: The position method makes use of the IMU based attitude to reduce the pose estimation problem to a position estimation problem. It will be shown that even in the case of large errors in the attitude estimation process, the method is more accurate than the P3P method. The position estimation method of figure 6 can be described as follows:

The projection of a 3D point onto the image is given by the following formula, assuming a pinhole camera model. Metric position x, y and z is normalized and multiplied with the intrinsic matrix to obtain pixel coordinate j. Here focal lengths f_x, f_y and principal point $c_{x,y}$ are both in pixel units.

$$\begin{bmatrix} j \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \\ 1 \end{bmatrix} \tag{III.1}$$

A detected gate corner points in the image frame can be converted to a bearing vector using the camera model.

$$\mathbf{v} = \begin{bmatrix} (j_x - c_x)/f_x \\ (j_y - c_y)/f_y \\ 1 \end{bmatrix} \tag{III.2}$$

The vectors $\mathbf{v_1}$ to $\mathbf{v_4}$ for each corner point are described in the camera frame.

These bearing vectors are rotated into the world frame with the camera to body rotation matrix R_c^b and body to world matrix R_b^w. The rotation matrices are based on the current attitude and heading, together with the camera to body transform.

$$\mathbf{n_i} = R_b^w R_c^b \mathbf{v_i} \tag{III.3}$$

The vectors $\mathbf{v_1}$ to $\mathbf{v_4}$ and the known 3D gate corner point locations $\mathbf{p_1}$ to $\mathbf{p_4}$ are then used to parameterize the light rays originating from the corner points $\mathbf{x_1}$ to $\mathbf{x_4}$.

$$\mathbf{x_i} = \mathbf{p_i} + \lambda \mathbf{n_i} \tag{III.4}$$

The intersection of these light rays then determines the camera position. Errors in corner point location or attitude estimation can however cause the rays to not intersect perfectly. The problem is solved by writing the point to line distance and optimizing the translation $\mathbf{t}$ of the body frame, such that the distance to all rays is minimized.

$$D(\mathbf{t}; \mathbf{p_i}, \mathbf{n_i}) = \|(\mathbf{p_i} - \mathbf{t}) - ((\mathbf{p_i} - \mathbf{t})^T \mathbf{n_i})\mathbf{n_i}\| \tag{III.5}$$

$$\underset{\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{i=4} D(\mathbf{t}; \mathbf{p_i}, \mathbf{n_i}) \tag{III.6}$$

To find the best approximation the problem is formulated as a least squares problem. Therefore this method will further be referred to as the LS method.

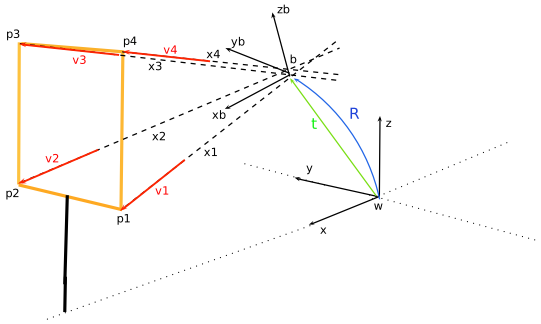
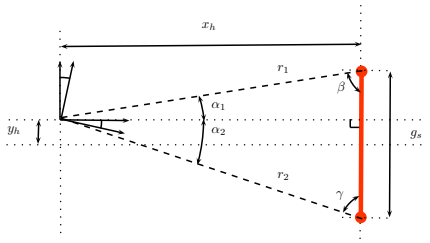


Fig. 6: Gate based pose estimation problem

2) Position estimation histogram: The histogram gate detection method can find the vertical segments of a gate. The positions of the segments in image frame are converted to angles α_1 and α_2. These angles determine the vehicles position by using the simple trigonometric equations of III.7-III.10. Note that only the position in the horizontal plane can be estimated.



$$\gamma = \pi/2 - \alpha_2 \tag{III.7}$$

$$r_1 = (sin\gamma\ g_s)/sin(\alpha_1 - \alpha_2) \tag{III.8}$$

$$x_h = cos\alpha_1\ r_1 \tag{III.9}$$

$$y_h = g_s/2 - sin\alpha_1\ r_1 \tag{III.10}$$

3) Heading estimation: Heading estimation on-board small MAVs is generally performed by using a magnetometer sensor. However, in an indoor environment the earth magnetic field is often distorted, preventing the MAV of obtaining a reliable heading estimate. Alternatively heading can be estimated with the IMU by integrating the gyroscope rates, however small sensor bias errors will cause this estimate to drift over time. Therefore, a visual heading estimation method is needed. The method estimates the heading angle with respect to the gates by minimizing back-projection errors. When the heading of each gate is known in the track, the global heading of the MAV can be estimated.

The visual heading estimation can be described as an optimization problem where orientation R and translation $\mathbf{t}$ have to be found to minimize the error between the gate detection and its back-projection. During the optimization the altitude h as well as the distance d to the gate are kept constant in order to reduce the computational complexity of the problem. This is allowed because the altitude is already measured by the sonar and known with relative high certainty. Also, the distance to the gate, estimated by the LS method is known with a relative high certainty, even under large initial attitude and heading errors. This can be observed from the results in the experiments section. The translation $\mathbf{t}$ of the body frame is therefore only depending on constant distance d, altitude h and a varying gate approach angle a, as shown in figure 7.

$$\mathbf{t}(d, h, a) \tag{III.11}$$

The position of a corner point in camera reference frame is obtained by multiplying the corner point position $\mathbf{p_i}$ with the world to body and body to camera rotation matrices.

$$R_w^b(\phi, \theta, \psi) \tag{III.12}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_b^c R_w^b(\mathbf{p_i} - \mathbf{t}) \tag{III.13}$$

Back-projection error per corner point is given as follows,

$$B(a, \phi, \theta, \psi; d, h, \mathbf{p_i}, s_i) = \|s_i - j_i(x, y, z)\| \tag{III.14}$$

with s_i the actual corner point detection i in the image. Point j_i is the image point projected using the pinhole model. Here j is a function of x,y,z position in camera frame.

For all four points the optimization problem can be described as in.

$$\operatorname{argmin} \sum_{i=1}^{i=4} B(a, \phi, \theta, \psi; d, h, \mathbf{p_i}, s_i) \tag{III.15}$$

This problem is solved with gradient descent and yields the heading angle ψ.

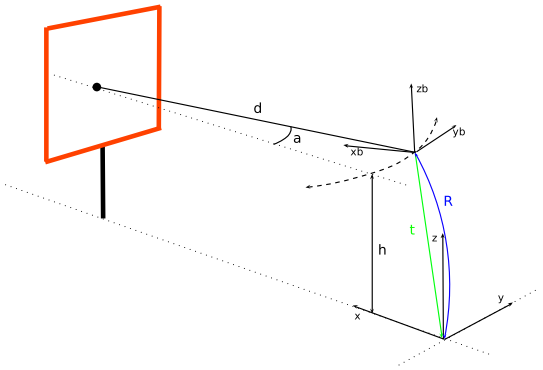


Fig. 7: Gate heading estimation

IV. State estimation

The vision-based position estimates described in the previous chapter are of low rate and contain noise, which is insufficient for control. Also, these estimates are only available when a gate is in view of the camera. The estimation performance can be improved by fusing the vision-based estimates with data from other on-board sensors. A common approach of such filtering is performed by combining IMU data with position data in an Extended Kalman Filter(EKF). EKF based position filters consist of a prediction and a measurement update step. The prediction model used is generally a kinematic model, which integrates IMU angular rates into attitude angles. Attitude is then used for rotating the accelerometer measurements in a global frame, after which they are integrated twice to predict the position. In a kinematic model however errors in the accelerometer bias will be integrated over time, which causes the velocity error to increase unbounded between measurement updates. Therefore, the prediction model of the current filter instead makes use of the aerodynamic forces measured by the on-board accelerometer. This approach makes the velocity prediction error linearly dependent of the accelerometer bias error. The prediction error therefore only varies at the same rate as the slowly changing bias drift. Also, the quality of these velocity measurements is not dependent on the floor texture, as with common optical flow velocity estimation methods. A similar method can be found in [1], however their approach still relies partly on optical flow.

A. Drag based velocity

One of the properties of an accelerometer is that it measures specific force(a_x, a_y, a_z), rather than vehicle acceleration. This can be interpreted as the resultant external force applied, divided by the vehicle mass. In x- and y-direction this is described as:

$$a_x = \frac{F_{dx}}{m} \ , \ a_y = \frac{F_{dy}}{m} \tag{IV.1}$$

Furthermore the accelerometer z-axis is assumed to be aligned with the body z-axis z_b of the vehicle, which is the same axis in which the resultant thrust vector T acts. In figure 8 only the two-dimensional case is given, therefore the force F_{dx} measured in the body x-axis x_b is purely caused by drag. For quadrotor MAVs the drag is mostly caused by the blade flapping effect [1], which is linearly dependent from airspeed v_{bx} with drag-coefficient k_x.

$$F_{dx} = v_{bx} k_x \ , \ F_{dy} = v_{by} k_y \tag{IV.2}$$

Once the drag forces are known, they can be converted to airspeed by the inverse drag model, while accounting for sensor biases b_x, b_y.

$$v_{bx} = (a_x - b_x)\frac{m}{k_x} \ , \ v_{by} = (a_y - b_y)\frac{m}{k_y} \tag{IV.3}$$

This theory therefore allows to measure airspeed in body frame in the x_b and y_b body axis, which are perpendicular to the thrust axis z_b. Note that this theory is able to measure airspeed only and not ground speed. However in an indoor scenario these quantities can be assumed the same.

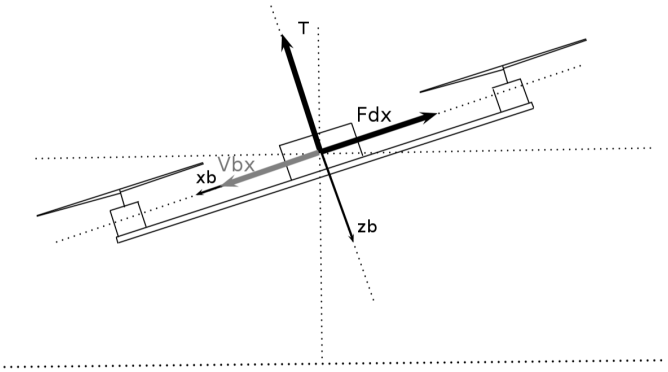


Fig. 8: Drag model

B. Extended Kalman Filter

The dynamics of the MAV can be described by a non-linear state-space system with state vector $\mathbf{x}$, input vector $\mathbf{u}$, additive process noise $\mathbf{w}$ and measurement noise $\mathbf{m}$.

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) + \mathbf{w} \tag{IV.4}$$

$$\mathbf{y} = h(\mathbf{x}) + \mathbf{m} \tag{IV.5}$$

The state vector of the Extended Kalman Filter estimates the following states. x, y and z position in world frame, body velocity v_{bz} in z-axis, as well as accelerometer biases b_x, b_y, b_z. The input vector contains raw accelerometer measurements a_x, a_y, a_z, gyroscope body-rates p and q and IMU based attitude angles ϕ, θ, ψ. Output vector $\mathbf{y}$ contains the x, y and z position estimates.

$$\mathbf{x} = (x, y, z, v_{bz}, b_x, b_y, b_z)^T \tag{IV.6}$$

$$\mathbf{u} = (a_x, a_y, a_z, \phi, \theta, \psi, p, q)^T \tag{IV.7}$$

$$\mathbf{y} = (x, y, z)^T \tag{IV.8}$$

The prediction model uses the accelerometer measurements a_x and a_y for estimating the body velocity in x- and y-direction. These are then rotated into the world frame with

rotation R_b^w according to the attitude inputs ϕ, θ and ψ from the attitude filter.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R_b^w(\phi, \theta, \psi) \begin{bmatrix} (a_x - b_x)\frac{m}{k_x} \\ (a_y - b_y)\frac{m}{k_y} \\ v_{bz} \end{bmatrix} \tag{IV.9}$$

The vehicle acceleration in z body-frame $\dot{v_{bz}}$ is obtained using the relation of the vector time derivative in rotating frame. Inertial acceleration expressed in the body-frame $\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\Big|_I^b$, can be measured by the accelerometer after subtracting the gravity vector and sensor biases. Rotational rates p,q and r between inertial- and body-frame are measured by the gyroscopes.

$$\left.\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\right|_I^b = \begin{bmatrix} \dot{v_{bx}} \\ \dot{v_{by}} \\ \dot{v_{bz}} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} v_{bx} \\ v_{by} \\ v_{bz} \end{bmatrix} \tag{IV.10}$$

Isolating the z-component, working out the crossproduct and substituting the drag based velocity yields the following result.

$$\dot{v_{bz}} = a_z - b_z + gcos\theta cos\phi + q(a_x - b_x)\frac{m}{k_x} - p(a_y - b_y)\frac{m}{k_y} \tag{IV.11}$$

Furthermore the accelerometer biases are assumed to be constant during the prediction step.

$$\dot{b_x} = \dot{b_y} = \dot{b_z} = 0 \tag{IV.12}$$

The EKF prediction step is performed at a constant rate while the measurement step is only performed when a new visual measurement becomes available. Now velocity and position are also estimated when visual measurements are temporarily unavailable. EKF based state estimation is performed in all closed-loop parts of the track. During the open-loop turning maneuver no feedback is needed, therefore the EKF is inactive and reinitializes when starting a new leg of the track. The discrete time EKF can be described by the following equations.

$$\mathbf{x_{k+1,k}} = \mathbf{x_{k,k}} + \int_{t_k}^{t_{k+1}} f(\mathbf{x}, \mathbf{u})dt$$

$$P_{k+1,k} = F_{k+1,k}P_{k,k}F_{k+1,k}^T + Q$$

$$K_{k+1} = P_{k+1,k}H_x^T(H_xP_{k+1,k}H_x^T + R)^{-1}$$

$$\mathbf{x_{k+1,k+1}} = \mathbf{x_{k+1,k}} + K_{k+1}(\mathbf{z_{k+1}} - h(\mathbf{x_{k+1,k}}))$$

$$P_{k+1,k+1} = (I - K_{k+1}H_x)P_{k+1,k}(I - K_{k+1}H_x)^T + K_{k+1}RK_{k+1}$$

C. Heading filter

Heading can be estimated by gyroscope integration alone, however small bias errors in the angular rate will result in large angular errors over time. Therefore, heading is estimated by fusing gyroscope rate measurements with vision-based heading measurements in a complementary filter.

$$\psi_k = k(\psi_{k-1} + \dot{\psi}\ dt) + (k-1)\psi_v \tag{IV.13}$$

The gain k determines the weight given to the gyroscope measurements $\dot{\psi}$ with respect to the vision measurements ψ_v. Gyroscope measurements are available at a rate of 512 Hz, while the vision measurements are only available when a detection is made at a maximum rate of 15Hz. Therefore, when no vision measurements are available the gain will be set to 1 which results the angular rate simply being integrated. Filter results over a multi-lap flight are discussed in the results section.

V. Control

The control strategy in the drone racing scenario consists of a mode switching controller which can change between open loop and closed loop control. Closed loop control is performed when approaching a gate, which provides visual feedback from the gate detections. For turning, an arc maneuver is performed using an open loop control method. In both control modes forward velocity is maintained by commanding a fixed pitch angle.

For the open loop arc maneuver roll and yaw angles over time are determined based on the velocity of the MAV. Speed along the trajectory is then predicted using the initial speed, when entering the arc and by numerically integrating the dynamical MAV model.

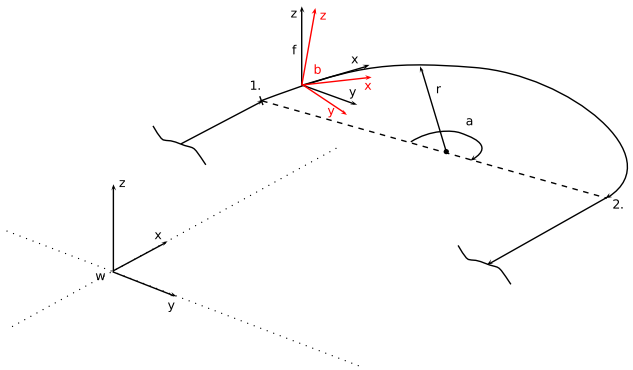


Fig. 9: Arc maneuver

For the arc maneuver velocity is described in f frame which moves along the arc with x-axis tangent to the arc and z-axis pointing up as in figure 9. The dynamical model is based on newton's second law in a rotating frame.

$$m\left.\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\right|_W^f = \mathbf{F_{ext}^f} \tag{V.1}$$

$$\left.\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\right|_W^f = \left.\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t}\right|_f^f + \Omega_{fW}^f \times \mathbf{v_f} \tag{V.2}$$

With external forces in the f frame consisting of gravity, thrust and drag.

$$\mathbf{F_{ext}^f} = m\begin{bmatrix}0\\0\\-g\end{bmatrix} - mR_b^f\begin{bmatrix}0\\0\\T\end{bmatrix} - m\mathbf{D_f} \tag{V.3}$$

After simplifying for the arc maneuver, the model can be described as follows, assuming a constant pitch angle and a negligible small roll rate.

$$\dot{\mathbf{v}}_\mathbf{f} = \begin{bmatrix}0\\0\\g\end{bmatrix} + R_b^f\begin{bmatrix}0\\0\\T\end{bmatrix} + \mathbf{D_f} - \begin{bmatrix}0\\0\\\dot{\psi}\end{bmatrix} \times \mathbf{v_f} \tag{V.4}$$

Acceleration due to drag $\mathbf{D_f}$, in f-frame is obtained by rotating the velocity $\mathbf{v_f}$ in body frame, calculating the drag and rotating back in f frame.

$$\mathbf{v_b} = R_f^b\mathbf{v_f} \tag{V.5}$$

$$\mathbf{D_b} = diag(k_x, k_y, k_z)\mathbf{v_b}/m \tag{V.6}$$

$$\mathbf{D_f} = R_b^f\mathbf{D_b} \tag{V.7}$$

Thrust is calculated by assuming zero vertical acceleration.

$$T = (\frac{-g - \mathbf{D_{f_z}}}{cos\phi})/cos\theta \tag{V.8}$$

The desired roll command can then be determined by writing the y-component of V.4 and enforcing $\dot{v_{f_y}}$ zero, to maintain a zero side slip turn.

$$\dot{v_{f_y}} = -sin\phi\ T + \mathbf{D_{f_y}} - v_{f_x}\dot{\psi} \tag{V.9}$$

When using

$$\dot{\psi} = \frac{v_{f_x}}{r} \tag{V.10}$$

and substituting for T , the roll-command is as follows.

$$\phi_{cmd} = \tan^{-1}((\frac{v_{f_x}^2}{r} - \mathbf{D_{f_y}})\frac{cos\theta}{g + \mathbf{D_{f_z}}}) \tag{V.11}$$

During the straight parts of the drone race track simple lateral PID position control is used for passing gates. Altitude is maintained with vertical PID control. High-level control is based on switching between the straight flying mode and the arc maneuver. After each gate pass an on-board stored sequence determines where and how far to turn, as well as where the next gate is approximately located.

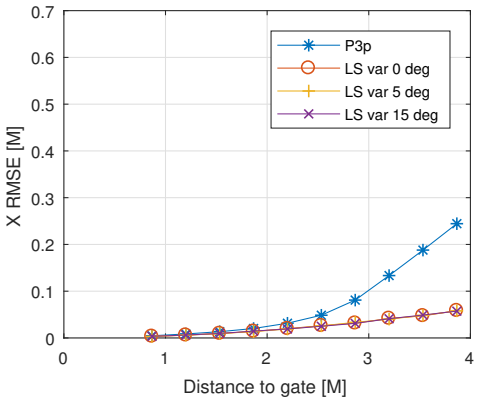
VI. Experimental results

In the following section the performance characteristics of the vision, state estimation and control methods are evaluated separately. Also, the performance of the complete system is evaluated in a simple racing scenario with two gates, as well as on a more complex five-gate track.

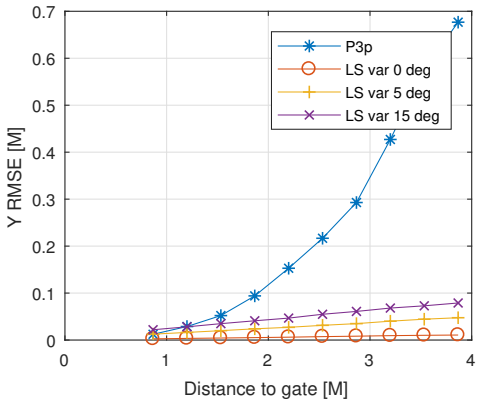
A. Vision-based position and heading

Position is estimated using the least squares method(LS), which is complemented by the histogram method when close to the gate. Heading is estimated with the optimization based method(OP). The accuracy of the LS method as well as the OP method is compared with the standard P3P method in a simulation environment. The simulation approach allows the camera location to be chosen precisely and a high number of experiments can be performed repeatedly. For simulation an artificial gate is created, which is projected onto a virtual pinhole camera image. Because gate detections contain image noise and outliers, a set of real gate detections was compared with ground truth data. Based on this test the vision method experiments will therefore contain image noise with a standard deviation of 3.5 pixels.

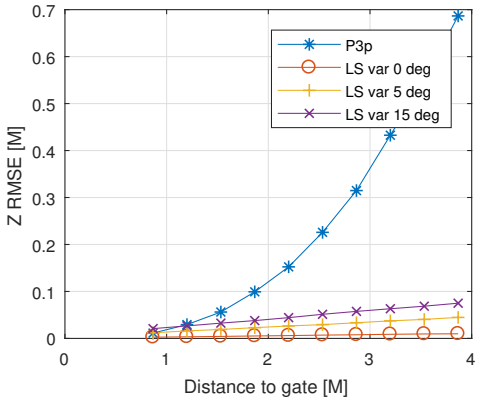
The LS position accuracy is evaluated with the Root Mean Squared Error(RMSE). The error varies mainly as a function of distance to the gate. Therefore, the RMSE is given per axis and as a function of distance. Each data point represents a thousand trails of the position estimation algorithm in the presence of pixel noise. In figure 9 the position RMSE of the LS method is compared with the P3P method from[9] at various distances. The LS method uses prior knowledge of the attitude and heading of the vehicle to obtain a more accurate position estimate. To study the effect of attitude error, noise with a variance of 0, 5 and 15 degrees is added to the attitude and heading estimates. It is clear from the figure that the LS method has far higher accuracy in RMSE compared to the P3P method, even in the presence of relative large noise in the attitude estimate.



(a) Position RMSE in X-direction



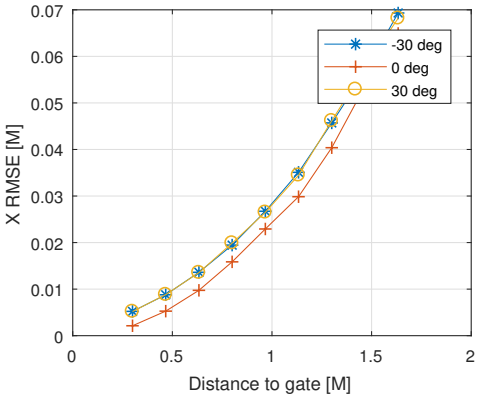
(b) Position RMSE in Y-direction



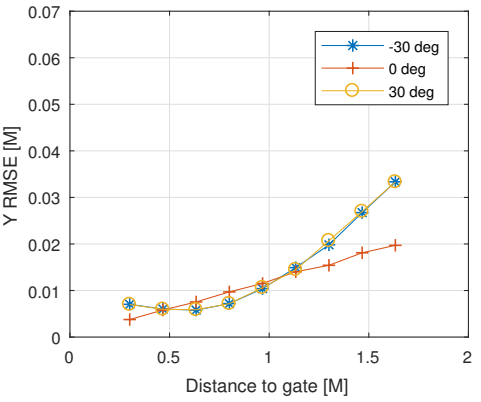
(c) Position RMSE in Z-direction

Fig. 9: X, Y and Z Least squares position RMSE as function of distance to the gate

Also, the histogram position estimation method is evaluated in simulation. Similar to the LS method, pixel noise with a standard deviation of 3.5 is introduced. Figure 9 shows the results of the position RMSE in the horizontal plane in x- and y-direction. The experiment is performed with a heading angle of -30, 0 and 30 degrees. From the figure it can be observed that the position error of this method is relatively low. However, in reality the method is only effective up to a maximum distance of 1.5 meters, due to possible background color.



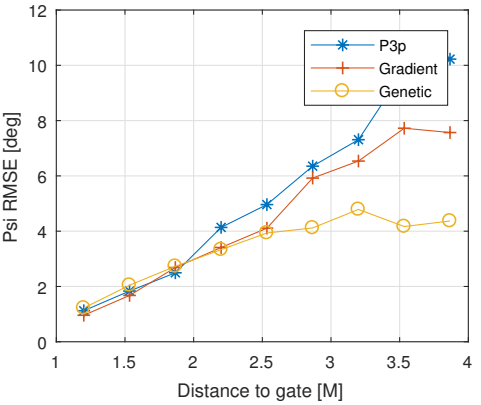
(a) Position histogram RMSE in X-direction



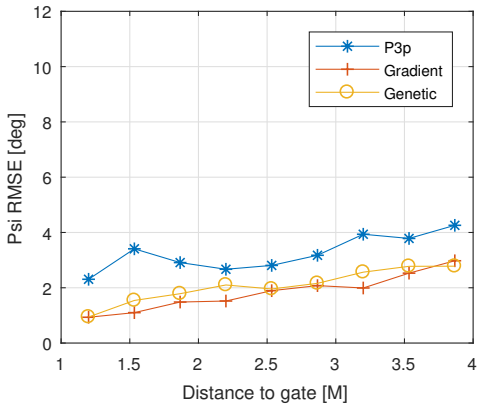
(b) Position histogram RMSE in Y-direction

Fig. 9: X and Y histogram position RMSE as function of distance to the gate

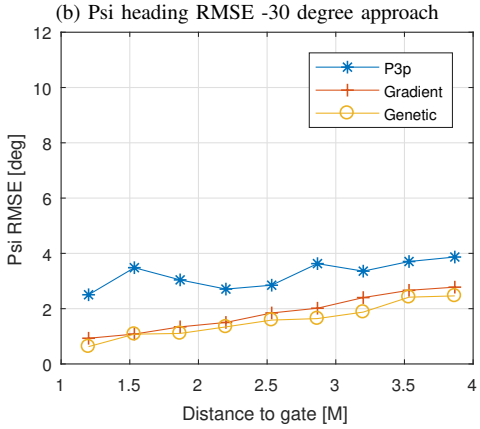
The heading as estimated with the OP method is compared with P3P heading in simulation. The experiments are performed at various distances to the gate, as well as with different heading angles to the gate. In figure 9 RMSE heading error is shown for P3P as well as for the gradient descent and genetic algorithm version of the OP method. The heading is varied from -30 degrees to 30 degrees. It can be seen that the OP method has a higher accuracy than the P3P method for both solution methods. Especially at large heading angles the difference is significant. The P3P method in this case evaluates groups of 3 points on a gate and tries to minimize the back-projection error with the fourth point in a RANSAC approach. P3P likely performs worse, because there are only a few points available for RANSAC.



(a) Psi heading RMSE 0 degree approach



(b) Psi heading RMSE -30 degree approach



(c) Psi heading RMSE 30 degree approach

Fig. 9: Psi heading RMSE at various gate approach angles

B. State estimation

The performance of the Extended Kalman Filter as described in the state estimation chapter is now evaluated using data from on-board the MAV. The MAV repeatedly flies a simple oval track with two gates. In figure 10,11 and 12 x, y and z position estimates of the filter are compared with the ground truth position values. Also, the estimated accelerometer sensor biases are shown in figure 15,16 and 17. In figure 10 the x position estimate is shown for one lap on the track. Note that the vision measurements are only available at the straight parts of the track, when a gate is in sight. After the MAV passes a gate and starts with the turn, the state is predicted. Note that there are only small errors in x position with respect to the ground truth. On the y position graph in figure 11 it can also be noted that only small errors occur, even when vision measurements are temporarily unavailable. In figure 12 also the z position estimate is compared to the ground truth. Finally, the body velocity estimate in z-direction is given in figure 13. The z-body velocity contains the most noise of all state variables, with errors up to 0.2 m/s. However, this internal state variable is not directly used for control and has a limited effect on the z position estimate.

Error distributions of the x, y and z position estimates are given in figure 14. All histograms are centered around zero error. Both x and y distributions have a small tail, which can be contributed to the fact that one of the turns in the data set was too wide. For turn accuracy also see figure 21.

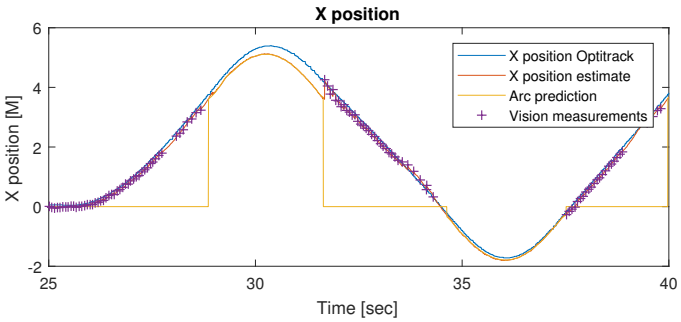


Fig. 10: Estimating X position

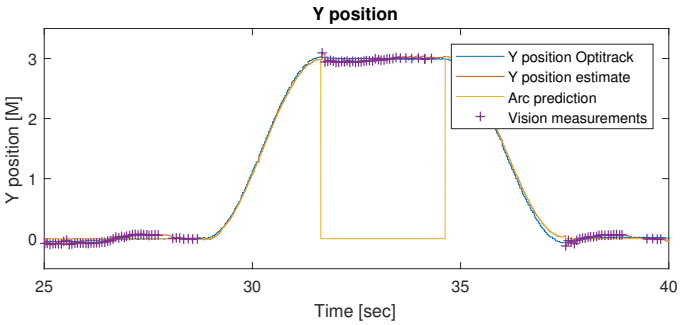


Fig. 11: Estimating Y position

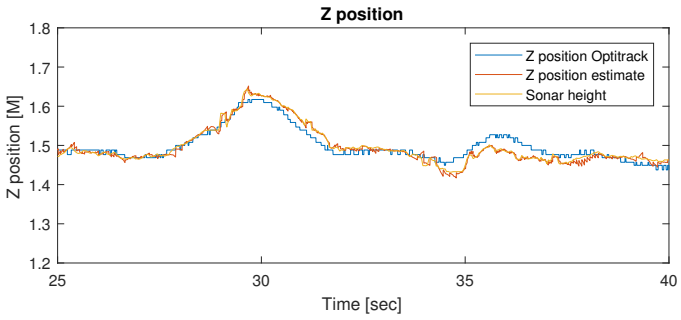


Fig. 12: Estimating Z position

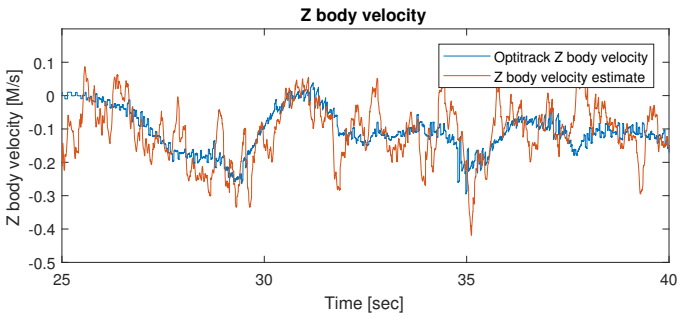


Fig. 13: Estimating Z body velocity

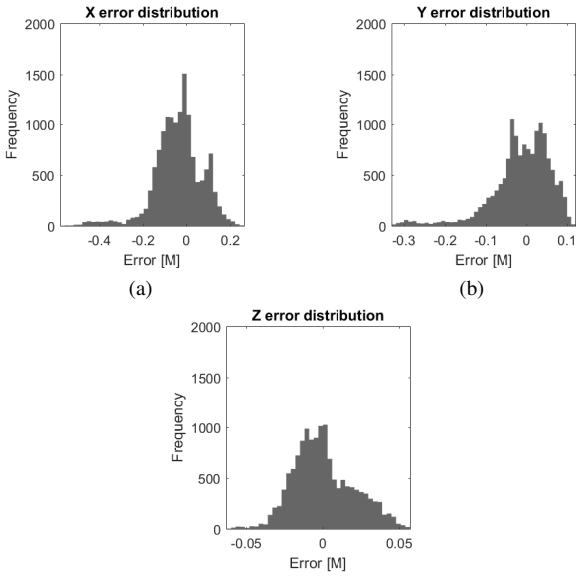


Fig. 14: X-Y and Z error distributions

The accelerometer bias estimation has a direct effect on the predictive performance of the EKF. Therefore, it is important to study the convergence of these biases. In figure 15 and 16 it can be observed that the X and Y accelerometer biases converge fast to the final value. This happens once the first vision measurements become available to the filter. In figure 17 the z-axis bias also converges fast, however the value first briefly overshoots. This short overshoot has no major effect on the estimation of the other state variables.

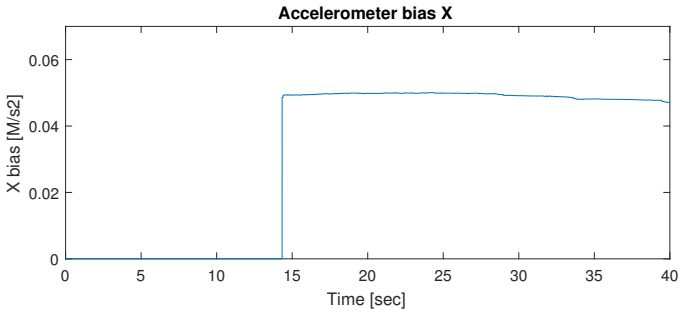


Fig. 15: X accelerometer bias estimation

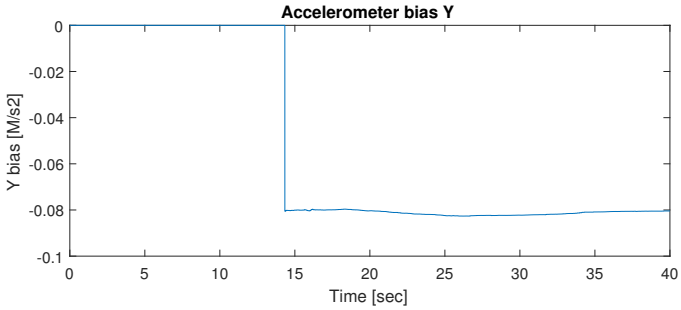


Fig. 16: Y accelerometer bias estimation

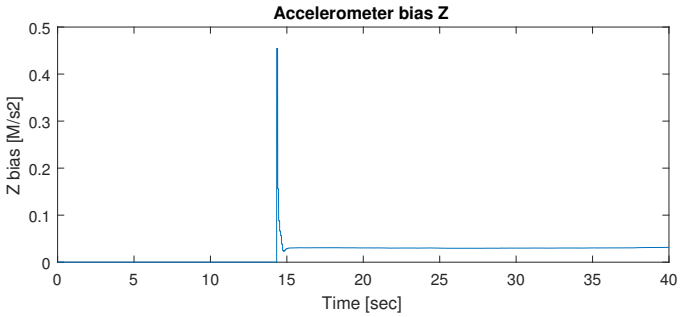


Fig. 17: Z accelerometer bias estimation

The estimated trajectory on the oval drone race track is shown in figure 18.

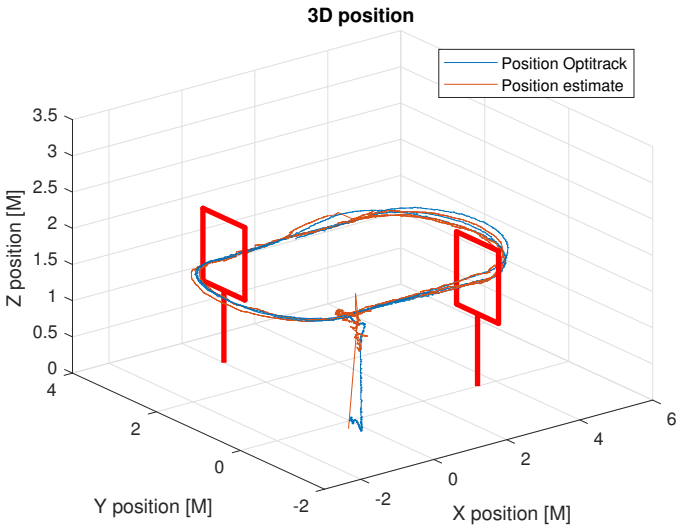


Fig. 18: Trajectory estimate during test

Heading is estimated using a complementary filter which fuses gyroscope rate measurements with visual heading estimates. In figure 19 the filtered heading and the integrated heading estimates are compared with the heading ground-truth data from Optitrack after 28 gate passes. From figure 20 It can be observed that the integrated heading has a drift of more than twenty degrees at the end of the flight. The filtered heading still shows a maximum error of approximately eight degrees, however the long term drift is sufficiently bounded. Note that the vision-based heading estimation unfortunately could not be implemented on-board of the drone in time. Therefore, for long flights on the two-gate track, Optitrack heading is used as a reference. Vision-based heading was post-processed in Matlab.

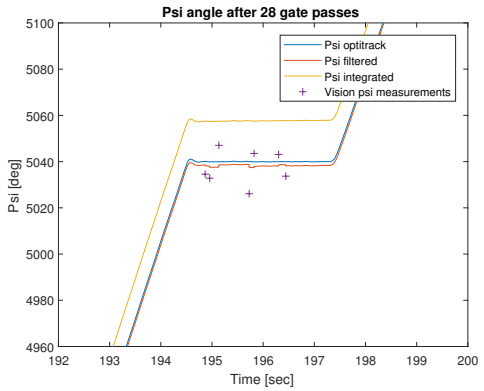


Fig. 19: Heading filter results after 28 laps

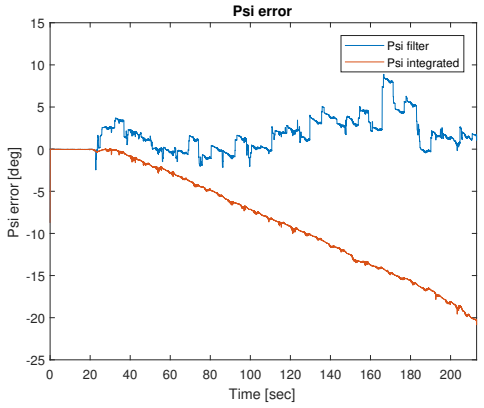


Fig. 20: Heading error

C. Control

The MAV performs turns with an open loop arc maneuver. The accuracy of this predictive method is naturally highly dependent on state estimation accuracy at the start of the maneuver. Therefore, a high number of arc turns are performed during a test flight, as can be seen in figure 21. This results in a pattern with an even spread in position error with a few outliers. The velocity estimation errors at the start of a turn where logged together with the resulting position errors at the end of each turn. These error variances in x- and y-direction are summarized in table I. The table shows that the position variance is the highest in y-direction. This also corresponds to the relative high entry velocity error in the same direction. The velocity error in y-direction is higher than the x error, because in x-direction a moving average filter with a longer window is chosen. This is because x velocity is more stable, due to the constant pitch angle.

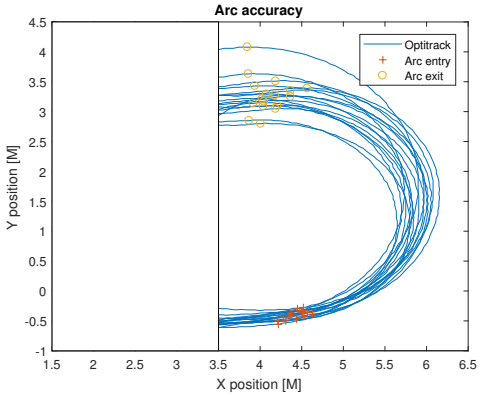


Fig. 21: Open loop arc maneuver

Axis	Entry speed variance M/S	Position error variance
X	0.0043 M/S	0.0296 M
Y	0.0106 M/S	0.8087 M

TABLE I: Open loop arc accuracy

D. Full track

The drone racing system is now tested in an obstacle dense drone racing track. The indoor environment provides no useful GPS-signal for navigation. Also, the floor offers no significant texture for optical flow velocity measurements. The environment was not equipped with a ground truth position system, therefore only estimated data is available. However, analyzing the estimated trajectory does give an insight of the flight and estimation performance in general.

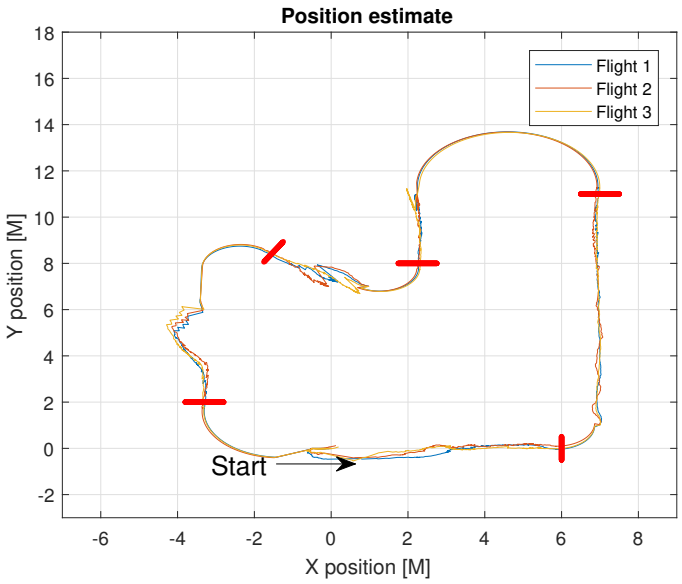


Fig. 22: Drone race track top view

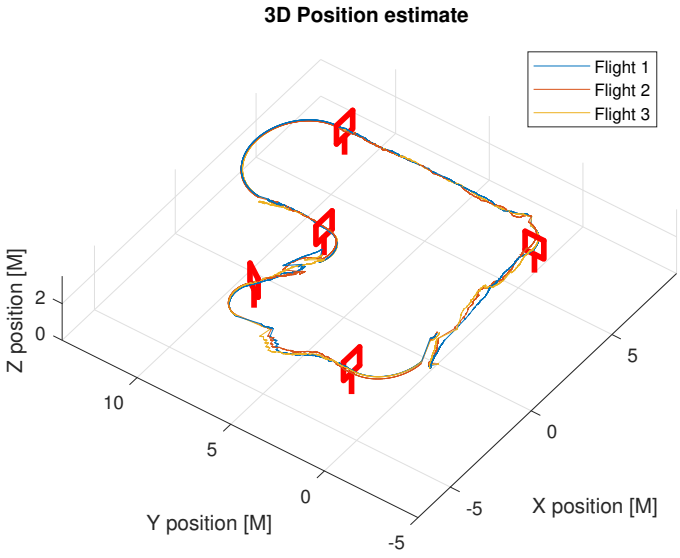


Fig. 23: Long race track

The track consists of five gates which have to be passed in the correct sequence. In figure 22 and 23 the estimated trajectories of three different one-lap flights are plotted. It can be observed that during some parts of the track some rapid changes in position occur. These jumps in position estimate occur once the next gate is first detected after a longer period without seeing a gate. During this period the position estimation only relies on integration of the drag based velocity. Errors in this prediction introduce an accumulating drift in the position estimate, which is corrected when a gate detection is available again. After the correction, the lateral position controller has enough time to steer the drone through the gate. Note that with these tests the visual heading estimation algorithm is not yet implemented on-board of the drone. The gyroscope-only heading therefore limits the flights to about three consecutive laps. After this, the heading estimate has already drifted excessively with more than 10 degrees.

VII. Conclusion and Future work

In this work a system overview is given of an autonomous racing drone. The system runs on-board a computationally constrained consumer MAV, while only making use of a monocular camera, IMU and sonar altimeter. The approach does not rely on computational complex SLAM or visual odometry methods, or optical flow based velocity measurements. But it rather detects the gates in the track and exploits their known geometry. Long sections without visual feedback are traversed using a combination of drag based odometry and open loop turning maneuvers. The MAV executes the track by linking different motion primitives based on the known track layout. Tests on the race track showed speeds over 2 m/s and yaw rates of more than 2 rad/s. The success rate of gate passes is high, with a record flight of 62 consecutive gate-passes on the small track and 3 laps on the five-gate track. It is shown that the complex task of autonomously flying a drone race track can be solved by only using sparse vision data and by taking into account the vehicles drag model, in a computationally lightweight sensor fusion approach.

However, there is still room for improvement on a number of subjects. The current system uses gate detections as the only visual input. Errors in the detection process thus have a large influence on the state estimation process, making it beneficial to improve the detection accuracy. Future work will therefore focus on using a computationally lightweight learning method for gate detection, to better cope with uncertain conditions in the environment. Additionally, the front facing camera can be used to provide optical flow measurements, which can be integrated in the state estimation filter. In this way the state estimation accuracy can be improved, without having the texture problem of downward facing optical flow. Also, the current leg-by-leg path planning can be improved, by implementing a planning method, which globally optimizes the trajectory for minimum time.

APPENDIX

The experiments take place in the basement of the Aerospace engineering faculty at Delft university of technology. The basement is filled with aircraft parts, serving as a reference for students. This location therefore provides a perfect obstacle dense and GPS-denied environment for demonstrating the capabilities of the system. Images from the drone race track are included in figure 24 to 27. On-board square gate and histogram detections are given in figure 28 and 29.

Fig. 24: First gate in the track

Fig. 25: Second gate

Fig. 26: Third and fourth gate

Fig. 27: Final fifth gate

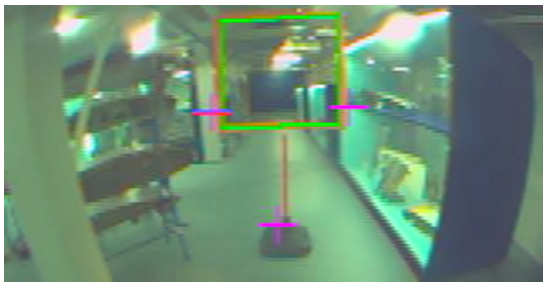


Fig. 28: On-board square gate detection

Fig. 29: On-board histogram detection

REFERENCES

[1] P.-j. Bristeau and N. Petit. The Navigation and Control technology inside the AR.Drone micro UAV. pages 1477–1484, 2011.

[2] A. Bry, C. Richter, A. Bachrach, and N. Roy. Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. *The International Journal of Robotics Research*, 34(7):969–1002, 2015.

[3] P. Dhane, K. Kutty, and S. Bangadkar. A Generic Non-Linear Method for Fisheye Correction. 51(10):58–65, 2012.

[4] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza. Aggressive Quadrotor Flight through Narrow Gaps with Onboard Sensing and Computing. 2016.

[5] X. S. Gao, X. R. Hou, J. Tang, and H. F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003.

[6] M. Hehn and R. D'Andrea. Quadrocopter Trajectory Generation and Control. *IFAC Proceedings Volumes*, 44(1):1485–1491, 2011.

[7] J. Illingworth and J. Kittler. A survey of the hough transform. *Computer Vision, Graphics and Image Processing*, 44(1):87–116, 1988.

[8] S. Jung, S. Cho, D. Lee, H. Lee, and D. H. Shim. A direct visual servoing-based framework for the 2016 IROS Autonomous Drone Racing Challenge. *Journal of Field Robotics*, 35(1):146–166, 2018.

[9] L. Kneip, D. Scaramuzza, and R. Siegwart. A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation. 1991.

[10] G. Loianno, C. Brunner, G. McGrath, and V. Kumar. Estimation, Control and Planning for Aggressive Flight with a Small Quadrotor with a Single Camera and IMU. *IEEE Robotics and Automation Letters*, 3766(c):1–1, 2016.

[11] D. Mellinger, N. Michael, and V. Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *Springer Tracts in Advanced Robotics*, 79:361–373, 2014.

[12] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Nips*, pages 91–99, 2015.

[13] J. Scherer, S. Yahyanejad, S. Hayat, E. Yanmaz, V. Vukadinovic, T. Andre, C. Bettstetter, B. Rinner, A. Khan, and H. Hellwagner. An autonomous multi-UAV system for search and rescue. *DroNet 2015 - Proceedings of the 2015 Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, pages 33–38, 2015.

[14] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. Vision-Based State Estimation and Trajectory Control Towards High-Speed Flight with a.

[15] Y. Tae. Gain-Scheduled Complementary Filter Design for a MEMS Based Attitude and Heading Reference System. pages 3816–3830, 2011.

[16] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. Grixa, F. Ruess, M. Suppa, and D. Burschka. Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue. *IEEE Robotics and Automation Magazine*, 19(3):46–56, 2012.

[17] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1:I–511–I–518, 2001.

Contents

Chapter 1

Introduction

Micro Air Vehicles or drones are gaining much interest over the past years. It is expected that these kind of small flying robots will become very important in a wide variety of applications. Possible applications include aerial photography, industrial inspections, search and rescue missions(Tomic et al., 2012), agriculture and many more. Currently most MAV operations are piloted manually which poses limitations in efficiency due to how many MAV's can be controlled by an individual pilot(Scherer et al., 2015). Also in most scenarios a reliable high-speed data link can not always be guaranteed. To overcome these issues MAV's should be capable of full autonomous flight.

Current MAV's are still lacking in autonomous flight capabilities(Floreano & Wood, 2015). This task is challenging due to the limited sensor quality and processing power on-board of such a small platform. Also in indoor and urban environments, GPS position is not available, or from low quality. Next to that current drones also have major limitations in range and endurance. The quadrotor drone type which is most commonly used, has a severely limited flight time. However the dynamics of the quadrotor type allow the drone to fly higher speeds with only a minor increase in energy consumption with respect to the hovering state. Therefore to increase range in autonomous flight, the drone has to fly faster.

A recent drone related trend is so called FPV drone racing. In this new sport multiple human pilots have to fly a small drone through a track of gates, while monitoring a live stream of the on-board camera. The speeds and level of maneuverability which is attained by these human pilots is stunning and still far from the capabilities of current autonomous drones. The drone racing scenario has much in common with autonomous high speed drone flight. Often these races are held in an indoor environment, blocking the GPS signal, which requires alternative navigation methods. Also the planning and executing of a time-optimal trajectory is part of the challenges. Therefore to promote the developments in the area of autonomous high speed MAV flight the IROS has started organizing an autonomous drone race since 2016. The goal is to fly a known track of gates, which have to be passed in the right order and as fast as possible. The race is indoor, therefore no GPS position can be used and other external position systems are not allowed. The drone has to navigate purely based on onboard sensors.

1-1 Research Objective

The Micro Air vehicle Lab, at which the current research will take place, was amongst the teams participating in the 2016 IROS autonomous drone race. Although their drone became second overall, the speed at which the drone flew the race was far from satisfying and not even close to the speeds which human pilots can attain. Therefore it was decided that the current drone racing system needs to be improved.

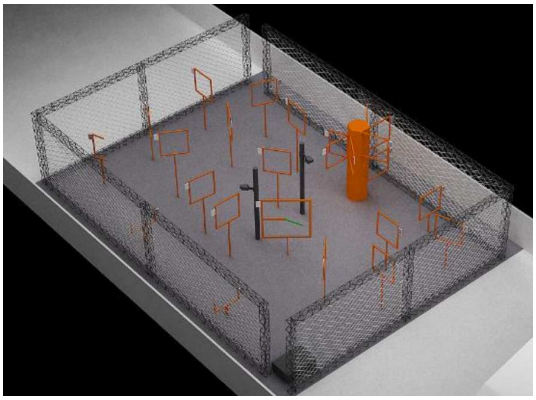
The task of autonomous drone racing can be divided in multiple different sub problems. First the state of the vehicle, with variables such as position, speed and attitude has to be estimated in a GPS denied environment. The state is estimated by fusing data from onboard sensors, such as the camera and Inertial Measurement Unit (IMU). Subsequently the estimated state is used to plan and follow the desired trajectory through the race track.

The current thesis continues on the MAV Lab's research towards an autonomous racing drone, capable of near human performance. First an evaluation will be made of methods in the area of computer vision, state estimation and control to find the best performing combination of methods in the drone racing scenario. The methods and possible improvements thereof will then be tested in a simulation environment after which the most promising methods will be implemented on-board a real drone for validation.

The objective of this thesis can be summarised as:

A solution for the autonomous drone racing problem has to be found by designing autopilot software which will turn an existing drone into an autonomous racing drone, capable of flying the track, without external positioning systems and only relying on on-board available sensors and processing power.

The MAV platform which will be used for this research is the Parrot Bebob drone. This small 250mm and 400gram commercial off the shelve (COTS) quadrotor MAV is equipped with one front facing and one downward looking monocular camera, multiple other sensors and a dual core processor.



(a) 2016 IROS autonomous drone race track

(b) Parrot Bebob commercial off the shelf drone

Figure 1-1: Autonomous drone racing

1-2 Main research question

How can a MAV perform high speed autonomous flight through a track of known obstacles?

Sub-questions:

1. What are the limitations and advantages of each sensor on-board of the MAV?
2. What state estimation technique should be implemented to optimize performance, while taking into account the limitations and advantages of each on-board sensor, without exceeding on-board computational resources?
3. What computer vision approach is most suitable for this task, given the available on-board computational resources?
4. How to generate trajectories such that they are feasible and are optimized on completing the track as fast as possible?
5. How does the currently designed system compares to the state-of-the-art in speed and agility?

1-3 Content and structure

This report contains an extensive literature study of the state-of-the-art in the various sub domains of high speed MAV flight. chapter 2 describes previous work on high speed autonomous drone flight. In chapter 3 the characteristics of the sensors on-board of the drone are given, including multiple methods for pre-processing this data. Chapter 4 evaluates different visual inertial state estimation methods which can be used with the on-board cameras and other sensors. Trajectory planning and control is discussed in chapter 5. The results of the literature study are discussed in chapter 6. As preliminary results a performance analysis is made of candidate state estimation and computer vision methods, which can potentially be used for the drone race system. Conclusion and future work are given in chapter 8 and 9.

Chapter 2

High speed flight in general

In this chapter an evaluation is made of previous work in the area of autonomous high speed MAV flight. Although the examples do not cover autonomous drone racing particularly, still most of the work is very much related to the problems of state estimation, trajectory planning and control. The previous work is divided by the type of positioning system that is employed for navigation. In the second part of this chapter an analysis is made of the work so far and what possible contribution the current project can potentially provide.

2-1 Previous work

The variety of previous work at high speed autonomous MAV flight is numerous. In recent years a wide range of different approaches to the problem of high speed flight have been tried. The approaches differ in what type of sensors are used, what state estimation methods were employed and what type of trajectory planning and control is performed. In the current section it is chosen to classify the work based on which type of main sensor is used for state estimation.

2-1-1 Motion capture systems

First major work in aggressive high speed flight of quadrotor MAV's was performed by using an external motion capture system. This system consists of an array of infrared illuminators and cameras, to capture infrared markers on the MAV. The system enables high accuracy position and attitude information, such that the research can focus more on the trajectory generation and control problem associated with high speed aggressive flight. (Mellinger, Michael, & Kumar, 2014) Focus on the design of dynamically feasible trajectories and controllers. Maneuvers in this paper include flying through narrow gaps and perching on vertical surfaces. They use the dynamical quadrotor model for generating the trajectories and designing PD and PID controllers. With this approach impressive results were achieved. However it has to be emphasized that both navigation and control where completely performed off-board. The

trajectories were designed offline in an iterative model based approach. Other work based around a external motion capture system was performed by (Hehn & D'Andrea, 2011). Here an iterative trajectory generation method is presented, capable of generating time-optimal trajectories. The trajectory is optimized as an optimal control problem separately on each degree of freedom. at each iteration it is evaluated if the control inputs remain within their boundaries and the trajectory is feasible. If the trajectory is not feasible the optimization is repeated with different boundary values.

2-1-2 LIDAR

In real world scenarios a reliable high speed data-link might not be available during an indoor flight, also GPS position is not available or has severe limitations on accuracy. Therefore the MAV can not rely on accurate external positioning systems or a high performance computer to perform off-board calculations. Position information in these cases can be acquired from other sources such as a LIght Detection And Ranging (LIDAR) sensor carried by the MAV. (Bry, Richter, Bachrach, & Roy, 2015) used this approach on fixed wing and quadrotor platforms. Both trajectory planning and the state estimation are discussed in detail. The experiments take place in known and obstacle dense environments. Trajectories are generated by first starting a straight line Rapidly exploring Random Tree (RRT) search and then using the waypoints to fit a polynomial spline trajectory. This polynomial trajectory is optimized for time. State estimation is performed by means of an extended Kalman filter with a particle filter on the LIDAR data as measurement update. Trajectory generation is still performed off board, however they claim that the algorithm is efficient enough to potentially run on-board an MAV.

2-1-3 Stereo vision

Instead of LIDAR technology also vision based state estimation can be employed. Monocular or stereo cameras are than the main sources of position information. An example of a stereo vision approach is described by(Shen, Mulgaonkar, Michael, & Kumar, n.d.). A small quadrotor platform is presented that is able to fly at high speed through an indoor environment. The main sensors are a stereo camera pair and an inertial measurement unit. The stereo pair consists of a main camera, which is used for estimating the camera's orientation and position up to a scale. The absolute scale of movement is recovered using the second camera to measure absolute scene depth. Here the monocular vision runs at a frequency of 20Hz, while the scale recovery with the second camera is only performed at 1Hz rate. The camera pose is then fused with data from the inertial measurement unit to provide a state estimate for control.

2-1-4 Monocular vision

One of the difficulties in autonomous indoor flight, are the limited size restrictions on the MAV platform. The MAV for example has to be small enough to be able to fit through open doors or windows. Therefore recent studies are focusing on developing smaller autonomous drones. In (Loianno, Brunner, McGrath, & Kumar, 2016), a small 250g quadrotor is able to fly

fast trajectories through narrow gaps. Planning state estimation and control is all performed on-board, while only using a single monocular camera and IMU as sensors. Camera and IMU data is combined in a Visual inertial Odometry approach, where feature points are tracked between frames at 30Hz, to estimate the current camera pose of the vehicle. An Unscented Kalman filter estimated the vehicle states at a higher rate of 500 Hz for control purposes. Trajectories are then generated by using the differential flatness approach. Here position and yaw angles are chosen as flat outputs. The fourth order position and second order yaw angle are then minimized as a quadratic programming problem. Both slalom and gap traversing trajectories were flown in a small indoor environment at speeds up to 4.5 m/s.

Another example where all sensing planning state estimation and control tasks are performed on-board is described by (Falanga, Mueggler, Faessler, & Scaramuzza, 2016). They focus on the traversing of a narrow tilted gap, to simulate a part of a search and rescue mission in a collapsed building. The gap size and tilt is such that an aggressive maneuver is required for a successful passage. Interesting is that in contrary to other approaches, the state estimation in this paper only makes use of the detection of the gap with a monocular camera. A so called Perspective-8-point algorithm is used to determine the position and orientation of the drone, given the known corner positions of the gap. The position and orientation are than fused with inertial sensor data in an Extended Kalman Filter. Furthermore the trajectories are designed as a ballistic trajectory in a tilted plane through the center of the gap. The traversing time is optimized given the relevant constraints using quadratic programming. The yaw angle is optimized separately to make the camera point in the direction of the gap for as long as possible. It is also discussed how the approach trajectory is generated which brings the drone to the start of the traverse trajectory.

2-2 Analysis and possible contribution

According to the previous overview of key papers in the area of high speed autonomous MAV flight can be summarized as follows. Initial work focused more on the trajectory generation and control part of the problem by using a high accuracy external positioning system for state estimation. Therefore the sensing and state estimation part of the problem was largely neglected. Also most of the processing was done off-board. These limitations only allow the drone to fly in a specially prepared room and not in a general area. Later work uses a scanning laser LIDAR and IMU as primary sensors. This approach allows all calculations to be performed onboard without relying on external positioning systems or off-board computing. This approach however has the drawback that LIDAR sensors are relatively heavy, requiring a larger drone and limiting flight time. Vision based state estimation methods prove useful, since camera sensors are weighing less than typical LIDAR units. Both stereo vision as well as monocular vision approaches were investigated. The most lightweight approach currently known uses a monocular vision system, fused with inertial sensor data.

One thing that can be noticed in the described methods is that all test maneuvers only last for a very small amount of time, typically only a few seconds. After a few seconds of fast and aggressive flight the MAV brakes and stabilizes itself again into a stable hover. It is however to our knowledge never been investigated what will happen if such an MAV would perform aggressive high speed flight for longer periods at a time. This could potentially deliver insights

on particular issues in state estimation, such as issues with bias drift or slowly diverging error behaviour. The drone racing scenario forms a perfect test case for such a drone.

Another aspect of the drone racing scenario which is not touched by the previous work, is the higher level decision making process which has to take place during a race. Typical drone race tracks consist of a complex course of gates, which have to be passed in the right order. The complexity of a course can make it unclear what next gate to pass, when consecutive gates are close to each other. This requires fast decision making and robustness to false detections.

Chapter 3

Sensing and vision

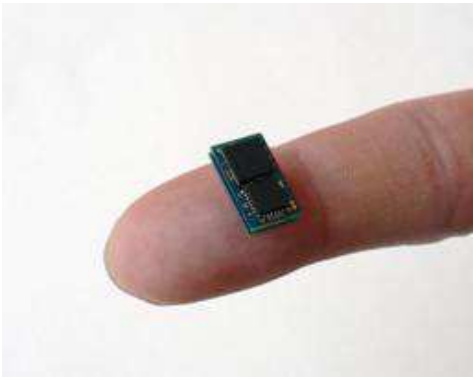
Sensing can be performed with a wide variety of sensors which all have their specific strong and weak points. Sensors can include cameras, inertial sensors, pressure or magnetic sensors. Due to the strict weight limitations which are inherent to small MAV's, only very small and lightweight sensors can be used. However these types of sensors tend to have much lower accuracy and higher noise compared to sensors used in other applications of aerospace or robotics. In this chapter the variety of sensors found on-board the Bebob MAV are described and evaluated for performance and their strong and weak points. In particularly the inertial and camera sensors are discussed in detail, since they will be used in the next chapter in a state estimation approach. For the inertial sensors the error characteristics are described. For the camera sensors multiple methods of preprocessing camera data are described and compared for their performance characteristics.

3-1 On-board sensors

One of the key technologies which enabled small drones is the invention of small scale lightweight inertial sensors. Micro Electrical Mechanical Sensor (MEMS) sensors were originally developed for the smartphone industry to determine the movements and attitude of a device. These sensors have the advantage that they are low cost, have a small form factor of several millimeters and weighing less than a gram. However these benefits come at a price, since MEMS sensors offer highly reduced performance in both accuracy and noise levels, compared to regular tactical or navigation grade sensors.

3-1-1 Inertial sensors

The most important on-board sensor is the Inertial Measurement Unit, consisting of a MEMS gyroscope and accelerometer. Both sensors have multiple causes of errors in the measured signal. For each sensor a description of the measured signal and an analysis of the various errors are given. MEMS gyroscopes are measuring angular rate but are sensitive to bias error,



(a) MEMS IMU

Figure 3-1: MEMS IMU data can be used to estimate attitude angles

scale error, alignment error and noise. Angular rate bias error changes steadily therefore rate integrated attitude can drift by as much as 200deg/hour. Bias drift is mainly influenced by temperature, which can be monitored by means of an onboard temperature sensor(TNO mpu6050). Scale error varies per sensor, but can be estimated using a calibration table. Alignment error is caused by the mounting of the MEMS chip on the PCB. Also the alignment error can be estimated using a calibration table. the noise can be assumed to be white gaussian with a limited bandwidth. MEMS accelerometers measure specific force and suffers from bias, scale and alignment errors as with the gyroscope.

Another option for estimation of the gyroscope and accelerometer biases is by means of a sensor fusion filter which takes into account other sensor data, such as acceleration and position data. Sensor fusing is studied in more detail in chapter 4.

3-1-2 Additional sensors

Apart from the main IMU sensor also a number of other sensors are available on-board the Bebop drone. These sensors are a MEMS magnetometer, a MEMS barometer and a sonar altimeter.

The magnetometer measures the direction of the earth's magnetic field lines. If an attitude estimation is available this direction vector can be used to estimate heading. The magnetometer is also very sensitive to ferromagnetic metals and magnetic fields other than the earth magnetic field. Therefore the magnetometer is mostly suitable for use in outdoor environments with limited metal and other potential disturbances.

The MEMS barometer is a highly sensitive air pressure meter which can be used for altitude measurements, when the pressure on the ground is known. However even with this high accuracy altitude can only be measured to about half a meter accuracy(Sensortec, 2015). Also the noise of such a pressure meter is very high. In an indoor scenario a barometer is also sensitive to sudden pressure changes, such as caused by the slamming of a door.

For measuring absolute altitude with respect to the ground surface a sonar altimeter is used. This sensor sends and receives high frequency sound waves and estimates distance by timing

the received signal. The sonar sensor is known to be more accurate, with a typical accuracy of 2.5cm (Maxbotix, 2015). However the distance is limited to five meters from the ground surface.

3-2 Monocular camera

The Parrot Bebob MAV used in this research has two Monocular cameras on-board. One high resolution front facing fisheye camera and one standard resolution downward facing camera. Monocular cameras are a rich source of information for navigation and collision avoidance purposes. The resolution of monocular cameras is generally higher than that of stereo camera systems. The size and weight of camera sensors make them suitable to be carried on board of small drones. However monocular cameras can only sense relative depth and motions, where for navigation absolute measurements are needed. Therefore the vision system has to be complemented with other sensors in order to estimate absolute motions and navigate properly. In the following paragraphs the characteristics of the monocular camera are described. Also a number of methods for extracting useful information from the monocular camera are described and compared in a qualitative analysis.

3-2-1 Calibration and pinhole model

Most computer vision techniques assume a perfect pinhole camera model, where the incoming light rays exit the lens the same angle as they enter the lens. The basic pinhole camera model without lens distortion is given in fig(). The transformation from world coordinates to camera coordinates is performed by subsequently multiplying the world point vector with the extrinsic matrix and the intrinsic matrix. The extrinsic matrix contains the rotation matrix and translation vector. Describing the camera pose with respect to the world frame. The intrinsic matrix contains the camera focal length and principal point. Parameter s is chosen such that the system is transformed to homographic coordinate system. Hence s = 1/Z

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

The world coordinates are given in an arbitrary unit such as meters. The camera focal length in millimeters is generally known which then given the dimensions and resolution of the image sensor the length and principal point can be transferred to pixel units.

However in reality almost all cameras have a lens which adds a certain amount of distortion to the light rays. Various types of distortion exist, with the most important ones being radial distortion and tangential distortion. In fig() is described how this model is extended to also include radial and tangential distortion. The image coordinates are transformed to homography coordinates and distorted with a radial and tangential distortion model. Radial distortion is given by the equation with parameters k1 to k6 and tangential distortion is modeled with p1 and p2.

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$x_h = x_c/z_c, y_h = y_c/z_c, r^2 = x_h^2 + y_h^2$$
$$x_u = x_c(1 + k_1 r^2 + k_2 r^4) + 2p_1 x_c y_c + p_2(r^2 + 2x_c^2)$$
$$y_u = y_c(1 + k_1 r^2 + k_2 r^4) + 2p_2 x_c y_c + p_1(r^2 + 2y_c^2)$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix}$$

3-2-2 Optic flow

One of the most important information that can be extracted from a sequence of images is optical flow. Early research ha be done by gibson, studying what techniques human pilots employed for manual aircraft landings. He concluded that an important aspect of their approach uses optical flow. Optical flow is described by Gibson as the deformation of the retinal image as a result from motion(Gibson, 1950). Possible information that can be retrieved from the optical flow field is direction of travel, time to contact with obstacles and relative speed.

Flow information

The equations describing the relation between camera movement and optical flow can be described as in (eq..)(Society, Society, & Sciences, n.d.). It has to be noted that these equations can be split in a rotation dependent part and a translation dependent part. This property can be simplify the problem when rotational information is available from other sources, such as gyroscopes.

$$u = -\frac{U}{Z} + x\frac{W}{Z} + Axy - Bx^2 - B + Cy = u_T + u_R,$$

$$v = -\frac{V}{Z} + y\frac{W}{Z} - Cx + A + Ay^2 - Bxy = v_T + v_R.$$

Figure 3-2: In Longett, Higgins are defined the optical flow vectors in image plane u and v. These relate to camera translational and rotational rates.

When for the feature z coordinate is known by means of another sensor such as an ultrasonic distance sensor, the camera speed can also be calculated. This approach of optical flow as

$$u = -\frac{U}{Z} + x\frac{W}{Z}$$

$$v = -\frac{V}{Z} + y\frac{W}{Z}$$

Figure 3-3: The flow can be broken down in a rotational and translational part. This can be utilized by measuring the rotational rates using the onboard gyroscopes. This leaves the following expression for optical flow.

speed measurement is commonly used in drones to assist in stabilizing the drone in lateral direction. When integrating the velocity measurements a crude form of position measurement can be obtained. This approach is susceptible to measurement noise which will cause a drift in position over time. There are however there are a number of drawbacks to this method. It is assumed that there is a flat underground, such that the measured z is predictable for the whole surface. If the surface is not flat errors in the solution will become more apparent.

The direction of camera motion can derived from the optical flow field by searching for the Focus of expansion (FOE). This is the point in the image where the optical flow due to translational motion equals zero. This can be applied to (eq..) which results in (eq..).

$$u_T = 0 = -\frac{U}{Z} + x_{\text{FoE}}\frac{W}{Z}$$

$$x_{\text{FoE}} = \frac{U}{W}$$

$$v_T = 0 = -\frac{V}{Z} + y_{\text{FoE}}\frac{W}{Z}$$

$$y_{\text{FoE}} = \frac{V}{W}$$

$$\frac{x_{\text{FoE}}}{y_{\text{FoE}}} = \frac{U}{V}$$

Figure 3-4: Focus of expansion

During research on the human braking behavior in car driving tasks(Lee, 1976) discovered that humans make use of the expansion rate of objects, to judge if a collision is imminent. the ratio of distance Z and forward speed W is known as the Time-To-Contact (TTC). Using the TTC it is possible to judge distances, without knowing absolute depth. TTC can be derived as follows from the optical flow equations(eq..).

$$u_T = (-U + xW)/Z = (-\frac{U}{W} + x)\frac{W}{Z} = (x - x_{\text{FoE}})\frac{W}{Z}$$
$$v_T = (-\frac{V}{W} + y)\frac{W}{Z} = (y - y_{\text{FoE}})\frac{W}{Z}$$

Figure 3-5: Time-Tc-Contact

$$\frac{W}{Z} = \frac{u_T}{(x - x_{\text{FoE}})} = \frac{v_T}{(y - y_{\text{FoE}})}$$

Figure 3-6: Time-To-Contact

Optic flow calculation methods

For optical flow measurements it is necessary to track visual features over multiple image frames. Various methods were developed to solve this problem. Commonly used feature types are corner points. Corner points are suitable because they have a gradient in two directions. This avoids the aperture problem of having only a gradient on one direction, such as a line segment, making it impossible to observe motion in parallel to the gradient direction. Corner based optical flow generally follows a two-step approach, consisting of a combination of a corner detection method and a corner tracking method. Well known detection methods are Harris (Harris & Stephens, 1988) and FAST(Trajkovii, Hedley, Trajkovic, & Hedley, 1998). A corner tracker which is widely used is Lukas Kanade(Lucas & Kanade, 1981). This method uses a numerical method to solve the shift of a block of pixels.

An example of another feature descriptor is the Scale Invariant Feature Transform or SIFT(Lowe, 1999). This feature descriptor looks at the various gradient directions in a pixel block and is independent of rotation, translation or scaling. Features are matched by comparing the collection of all sift features and matching the most similar feature descriptions with each other. The computational complexity of the algorithm makes it unsuitable for real-time embedded applications. This is illustrated by(Mori & Scherer, 2013) where sift features are used for monocular collision avoidance onboard an MAV. They have to run the sift algorithm offboard due to computational constraints. An adaptation to the sift algorithm is SURF, which is a faster version of the original sift algorithm(Xu & Namit, 2008). Although improving the performance surf is still very processor intensive compared to Lucas kanade and harris corner detection.

A recently developed method for optical flow is described in(McGuire et al., 2016). This edge Flow method performs a simple sobel edge filter, to convert the image to an edge image. The edge image is then converted to a histogram in x direction and a histogram in y direction. Histogram binning in x and y direction is defined as the total amount of edge pixels in each column or row respectively. The edge histogram is characterized by a certain pattern of peaks, depending from the pattern in view. If the scene viewed by the camera shifts in x or y direction, the corresponding peak pattern in the histogram also shifts. This approach gives the global optical flow. This optical flow measure still has to be de-rotated using the on-board

gyroscopes. The main advantage of this method is that it has very limited computational requirements; also the method performs better in regions with a low number of corner points, if compared to Lukas Kanade. An additional functionality is the measurement of diversion, which can also be derived by observing the change in peak pattern.

3-2-3 Perspective n point

For the drone racing scenario there is also an alternative option for computer vision based navigation. The drone race track will consist of a number of brightly colored gates, of constant shape and size. Also as with every race, the track layout is known in advance. This information can be exploited to be able to make a more efficient computer vision and navigation system. The gate size and shape are especially useful for relative position estimation. In case of a circular gate the eccentricity can provide information about rotation of the camera and the gate size provides slant range. Also for a gate with corners such as a square or hexagonal, a specific type of methods exist to determine the camera pose by viewing a set of points, of which the world coordinates are known. This problem is known as the Perspective-n-point (Pnp)(S. Li, Xu, & Xie, 2009).

P3P

Multiple solutions exist for this problem, both analytical as well as numerical. Also the methods vary for how many points are needed to solve the problem. The minimum number of points needed to solve the Pnp problem is three (P3p). The number of points available can of course be higher than the minimum needed for the respective method. In this case a RANSAC approach can be used, by back projecting the remaining points onto the image plane and comparing them with the viewed image points. A well known Pnp method is that of (Gao, Hou, Tang, & Cheng, 2003) in which three points are used to find four closed form solutions for the camera pose. An example of an analytical solution for the p3p problem is given in(Kneip, Scaramuzza, & Siegwart, 1991). The author compares its performance with the standard solution of Gao. In the current work performance of the analytical P3p is evaluated, in comparison to other methods.

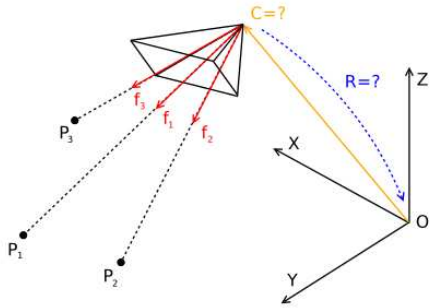


Figure 3-7: Perspective 3 point problem description

P4P

The Perspective-n-point problem also exist for 4 points, which is also known as P4p. Multiple solutions for this problem exist. The majority of these methods assume that the points involved are non-coplanar. However in the problem of pose estimation based on gate corners, the gate corner points clearly lie on a single plane. Methods which also function with coplanar points do exist, such as described in(Horaud, Conio, Leboulleux, & Lacolle, 2011).

Chapter 4

State estimation

Small unmanned aerial vehicles usually have a large number of on-board sensors. Typical sensors include inertial sensors, a camera, and a magnetometer for sensing the direction of the earth magnetic field. Furthermore a barometric pressure sensor for altitude estimation and a sonar sensor for low altitude height sensing. The challenge of optimally fusing all these sensors is complex due to the relative low quality of the sensor data and the very different data rates of the sensors.

In the first part of this chapter an elaborate description is made of sensor fusion by means of a Kalman filter. Different types of Kalman filters are discussed together with their specific characteristics. In the second part an overview is given about the most important monocular visual inertial navigation methods.

4-1 Kalman filter sensor fusing

4-1-1 Original Kalman filter

One of the most frequently used sensor fusion method is the Kalman filter. The Kalman filter is an optimal estimator for linear problems, capable of giving an optimal state estimate provided with multiple noisy sensor readings(Welch & Bishop, 2006). The filter chooses the optimal weight between the model based state prediction and the measurements, given the process and sensor noise characteristics. Five steps can describe the propagation of the filter as follows. In step one the current state is predicted based on the previous state given the process model. step two predicts the covariance given the previous covariance the process noise and the process model. The covariance prediction is then used for computing the kalman gain, which is the dynamic weight factor describing the importance of measurements with respect to the model prediction. Step four gives the optimal estimate of the current state by correcting the state prediction using the measurement and the Kalman gain. The fifth step computes the final covariance of the current state estimate. The original Kalman filter assumes a linear sensor model, which is mostly not the case. Still the filter has shown to work on systems which are locally linear(Lefebvre *, Bruyninckx, & De Schutter, 2004).

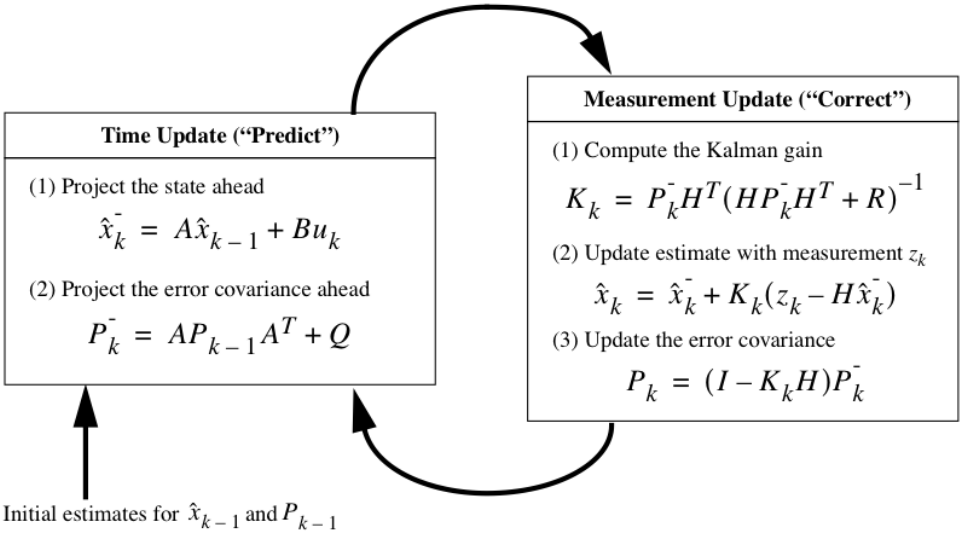


Figure 4-1: Original kalman filter equations

4-1-2 Extended Kalman filter

The original Kalman filter theory is only applicable to linear or approximately linear problems. To be able to apply the Kalman filter theory to nonlinear systems an extension to the linear Kalman filter was proposed, known as the Extended Kalman Filter (EKF). The EKF extends the regular kalman filter theory to nonlinear systems. This is accomplished by locally linearizing the nonlinear state transition and output matrices around the current state estimate. Hence calculating the first order partial derivatives or jacobians of the nonlinear state transition and output matrices. Also the state prediction step for the nonlinear model is performed by numerically integrating the state. One has to note that contrary to the linear kalman filter, the EKF is not guaranteed to converge, due to the linearization approximation (Boutayeb, Rafaralahy, & Darouach, 1995). The extended Kalman filter is widely used in SLAM and Visual Inertial Odometry methods.

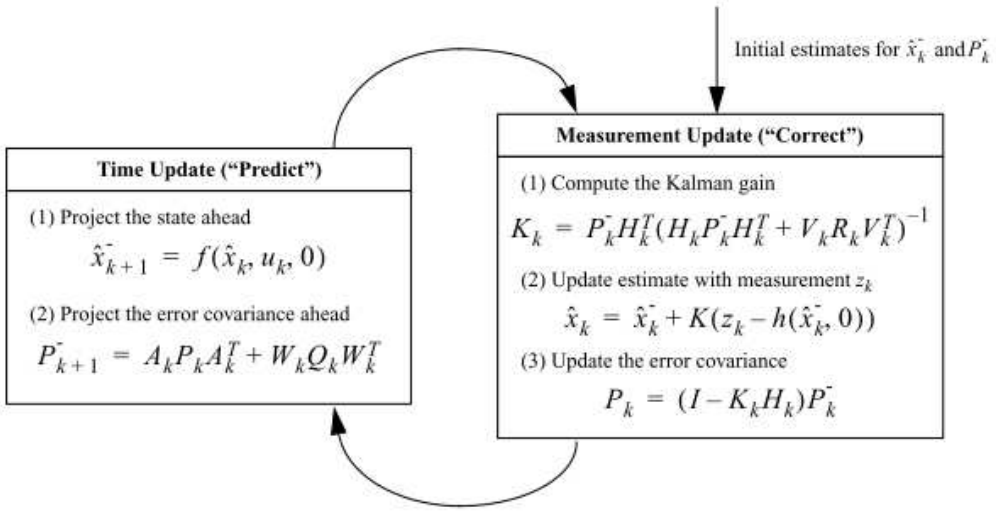


Figure 4-2: Extended kalman filter equations

4-1-3 Unscented Kalman filter

A different form of the Kalman filter for nonlinear systems is the Uncented Kalman Filter (UKF), first proposed by (Julier & Uhlmann, 1997). The UKF differs from the EKF that the state and error covariance prediction are directly propagated to the next time step by using the nonlinear prediction model. This is accomplished by selecting a small number of points around the current state estimate called sigma points, which are then propagated through the nonlinear prediction model. The resulting spread of the sigma points determines the new mean and covariance. The mean is then used as the state prediction. It has been proven that the UKF has a better convergence behavior than the EKF(Wan & Van Der Merwe, 2000). Also the results are more accurate, because the UKF uses a 3rd order nonlinear approximation instead of a 1st order approximation as with the EKF.

$$\begin{aligned} \boldsymbol{\mathcal{X}}_0 &= \bar{\mathbf{x}} & W_0 &= \kappa/(n+\kappa) \\ \boldsymbol{\mathcal{X}}_i &= \bar{\mathbf{x}} + \left(\sqrt{(n+\kappa)\mathbf{P}_{xx}}\right)_i & W_i &= 1/2(n+\kappa) \\ \boldsymbol{\mathcal{X}}_{i+n} &= \bar{\mathbf{x}} - \left(\sqrt{(n+\kappa)\mathbf{P}_{xx}}\right)_i & W_{i+n} &= 1/2(n+\kappa) \end{aligned}$$

Figure 4-3: Sigma point vectors and corresponding weights.

$$\boldsymbol{\mathcal{Y}}_i = \mathbf{f}\left[\boldsymbol{\mathcal{X}}_i\right].$$

Figure 4-4: Sigma point are propagated through the nonlinear mapping.

$$\bar{\mathbf{y}} = \sum_{i=0}^{2n} W_i \boldsymbol{\mathcal{Y}}_i.$$

Figure 4-5: Mean is calculated based on nonlinear mapped sigma points and corresponding weights.

$$\mathbf{P}_{yy} = \sum_{i=0}^{2n} W_i \left\{\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}}\right\} \left\{\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}}\right\}^T.$$

Figure 4-6: Covariance is calculated based on nonlinear mapped sigma points and corresponding weights.

4-1-4 Loose and tight coupling

One important distinction in the types of sensor fusing methods is the difference between loosely coupled systems and tightly coupled systems. This difference has to do with how the sensor data from each sensor is integrated in the state estimation filter. In a loosely coupled filter the sensor data is first separately preprocessed to some higher level of information, after which this preprocessed data from multiple sources is fused together to form the final state estimate. In a tightly coupled approach the raw sensor data is fused directly in the state estimation filter to provide the final state estimate. In general the tightly coupled methods give a more accurate result, however depending on the situation the loosely coupled design might be more favorable.

4-2 Monocular Visual inertial navigation

Navigation and state estimation are very important but challenging tasks for small scale unmanned aerial vehicles, both from a perspective of sensor limitations as well as processing power. In a GPS denied environment the vehicle can only rely on laser or vision based navigation. However if the goal is navigation on even the smallest drones possible, even LIDAR sensors exceed weight limitations.

4-2-1 Method classification

Visual state estimation can be divided in so called odometry methods, as well as Simultinous Localization and Mapping (SLAM) methods. In SLAM approaches the map and state are estimated concurrently(Barea, Bergasa, & Molinos, n.d.). In an odometry approach only the pose is calculated in an incremental fashion. Only a small subset of features might be used for refining the results, no attempt is made in reconstructing a globally consistent map of the environment(Scaramuzza, 2012).

Since it is decided that only the front facing camera should be used, visual inertial state estimation is limited to monocular methods only. One of the major difficulties with these

methods is that a monocular camera is not able to estimate absolute visual scale. Hence additional sensor data is needed to do this. Often inertial data is used for performing this scale estimate, but in theory also other sensors can be used. In both SLAM and Odometry methods there are differences in how visual data is fused with inertial data. This can be either a loosely coupled or a tightly coupled approach. In a loosely coupled approach the visual data is preprocessed into pose estimates or pose and map estimates. However these vision only estimates are still missing an global scale factor, which converts them to absolute estimates. In a tightly coupled approach raw visual data is used directly in the filter, instead of preprocessing it first. In general a tightly coupled approach his better performing in accuracy and robustness, However also other factors can play a role, such as computational complexity.

One other means of classifying the various different visual navigation methods is the way that they handle the detection of features of the environment. A distinction can be made between so called direct methods, semi-direct methods and feature based methods. Direct methods work by estimating camera motion and scene structure by directly evaluating pixel intensity values(Engel, Sch, & Cremers, 2014). Generally direct methods are more accurate because they include all information available from the image. Also direct methods are much more robust in scenes with little texture. However this comes at the cost of requiring more processing power.

Feature based methods on the other hand work by tracking a limited number of salient features between different camera poses(Mur-Artal, Montiel, & Tardos, 2015). The tracked features are then used to estimate both scene structure and camera motion. Back projection can then be used to further refine the pose and structure. Feature based methods tend to be faster than direct methods, but use only a limited amount of the information in the image which reduces their accuracy. Also environments with a low amount of detectable features form a problem when using such a method.

A relatively new development are the so called semi-direct methods(Forster, Pizzoli, & Scaramuzza, 2014). These methods are similar to the direct methods in the sense that they operate directly on pixel intensity values. However the pixel intensity optimization operations are performed on sparse set of small image patches, instead of on the full image. This approach increases the computational performance, while still benefiting from the accuracy and robustness of direct methods.

4-2-2 SLAM methods

Given the background information in the previous paragraph, the following paragraphs will describe the most important monocular SLAM methods used in MAV navigation. A brief description of the main working principles, together with its specific performance characteristics, including strong and weak points is given.

PTAM

One method which is frequently used for state estimation in MAV's is Parralel Tracking And Mapping (PTAM)(Jama & Schinstock, 2011). PTAM is a form of simultaneously estimating the camera pose and building a map of the environment. In this approach the tracking

and mapping are split, enabling them to work on two different processor cores to increase performance. The method, first proposed by (Klein & Murray, 2007). A typical sequence of the PTAM algorithm is follows: The pose estimation part of PTAM works by projecting previously known map points onto the current camera image, using an initial estimate of the new camera pose. The reprojected points are compared to the corresponding feature points in the image. Reprojection error is then minimized using a least-squares approach. Mapping is performed by using multiple camera poses to triangulate feature points. The map is further refined with local and global bundle adjustment.

PTAM was developed for use in small augmented reality workspaces, but also received attention from the autonomous MAV community. Multiple examples exist of PTAM implementations on MAV's(Jama & Schinstock, 2011). The original PTAM algorithm does not estimate absolute position. Therefore for use in MAV's the algorithm has to be adapted in order to estimate the global scale of the map. Most examples use a loosely coupled approach which fuses the PTAM based position with inertial data from the IMU by means of an EKF.

The inherent parallel processing structure of this method makes it especially suitable for use on modern multi core embedded devices. in(Weiss, Achtelik, Lynen, Chli, & Siegwart, 2012a) the PTAM algorithm is able to run on a 1.6 GHz atom computer, at a rate of 20Hz. Also the authors in (Klein & Murray, 2007) compared PTAM to EKF-SLAM on the same image sequence. They found that their method was more accurate by testing it in a office space scene. However the algorithm is also prone to a number of failure modes. The stereo matching initialisation can fail, feature tracking can fail, which results in wrong information being added to the map. Finally because the detection and matching of points make PTAM a feature based method, typically a large number of features are tracked to increase accuracy. This however makes the method sensitive to environments which lack of salient visual features.

ORB SLAM

One other important monocular SLAM methods is ORB-SLAM. This method estimates both the camera pose, as well as a globally consistent map. Also if a previously visited area is visited the algorithm performs a loop closure procedure, which globally refines the map. ORB-SLAM first proposed by (Mur-Artal et al., 2015) is a feature based method which employs ORB feature descriptors. These scale and rotation invariant(ref) features are used in both tracking, mapping and loop closure parts of the algorithm. During the tracking stage, the camera pose is estimated by matching point features from previous image frames to features in the current frame. The pose is optimized to minimize back projection error of the tracked points. One of the features of ORB-SLAM is that if tracking is lost due to camera occlusion an automated relocation process can recover the current position by looking for matches in a database of stored feature descriptions. Mapping is performed by adding new features to by means of triangulation. Also a test is performed to verify that the points are tracked properly, with minimal outliers. Local map consistency is then optimized using local bundle adjustment. The final step in the method is able to detect previously visited locations. These detections are than used to globally refine the map.

Multiple examples exist for ORB-SLAM implementations in MAV's, such as (Marquez, Garcia, & Mayol-cuevas, 2015),(Barea et al., n.d.),(ref). These approaches use a loosely coupled EKF framework for fusing the scaled position and speed measurements with the onboard

IMU. The first two papers also use speed information provided by the OEM optic flow sensor fusing methods pre-installed on their respective drone. Real-time performance is obtained, however it has to be noted that neither of these papers implement the full ORB-SLAM approach on-board of the MAV. They rather use a powerful ground station computer for running the ORB-SLAM algorithm, based on a video stream and sensor data from the MAV.

LSD SLAM

A recent development in direct monocular SLAM methods is Large Scale Direct SLAM. This method is able to build a large-scale map of the environment together with the current estimate of the camera pose. The method was first proposed by(Engel et al., 2014) and is able to run at a CPU. The method consists of three components, a tracking components, a depth map estimation components and a map optimization component. The tracking component estimates new camera poses by trying to minimize the variance normalised re projection error with respect to the previous frame. the optimization is performed by the iteratively re-weighted Gauss-Newton method. By normalizing by the depth noise, this methods takes into account the varying noise values. Also it has to be noted that the method only works on areas of the image with sufficient gradient.

4-2-3 Odometry methods

In the following paragraphs the most important monocular odometry methods used in MAV navigation are described. A brief description of the main working principles, together with its specific performance characteristics, including strong and weak points is given.

Multi State Constraint Kalman Filter

An example of such an approach of fusing visual and inertial data is the Multi State Constraint Kalman Filter (MSCKF) (Mourikis & Roumeliotis, 2007). Here the well-known Extended Kalman Filter is formulated in such a way that the tracking of visual features with the camera greatly improves the inertial navigation solution. Contrary to other approaches (Mingyang Li and Anastasios I. Mouriki), the features do not form a part of the state vector, but are rather used for posing geometric constraints on a sliding window of estimated camera poses. This approach of not including the tracked features in the state is less computational intensive if a high number of features has to be tracked over a short period of time.

EKF Visual Inertial Odometry

Contrary to the MSCKF approach EKF Visual Inertial Odometry (EKF VIO) does hold the feature positions inside the state vector. This approach has a computational advantage when a small number of features need to be tracked for a longer time(M. Li & Mourikis, 2012). An example of the EKF VIO formulation is described in (Bloesch, Omari, Hutter, & Siegwart, 2015). The EKF state propagation of the position velocity and attitude states is based on the IMU measurements inserted into a kinematic model. Also the feature states are propagated based on the current speed predictions and angular rate measurements. Features are described

in the inverse depth representation by a bearing vector and a distance. This representation has an advantage during the feature initialization process, when far away features can be used immediately for improving the camera motion estimate(Civera, Davison, & Montiel, 2008). Features used are rectangular patch features instead of corner features. Each feature consists of a small block of pixel intensity values. Pixel intensity errors are directly used in the measurement update step of the EKF. This so called semi-direct approach has as an advantage that it is not relying on the availability of corner like features. The feature tracking will also work in scenes that are dominated by straight lines, common for most man made environments.

An example of this can be shown in(Bloesch et al., 2015). Here a semi-direct visual inertial odometry algorithm is implemented on a MAV and is able to run at 20Hz update rate. in (Loianno et al., 2016) the VIO algorithm is implemented on a 250g micro drone. state estimation is accurate enough to aggressively fly through tilted gaps.

EKF Optical flow speed estimation

The previous approaches mentioned that fuse visual and inertial data provide the vehicle state, by either keeping track of a window of previous states, or by including the features inside the state vector. A more computationally efficient approach is proposed by(Weiss et al., 2012a). They propose a framework for metric speed estimation by fusing optical flow information with inertial measurements. The approach makes use of the fact that a camera can estimate scaled speed based on optical flow and the IMU can estimates metric speed, corrupted by sensor noise and bias drift. As with the some of the previously discussed visual inertial state estimation approaches the complementary nature of both sensors make them suitable for sensor fusion with an EKF.

In the chapter of sensing and vision was described how optical flow information can be extracted and processed. One of the things which can be extracted from the optical flow field is the location of the FOE. The FOE points to where the camera is traveling, expressed in the camera reference frame. Hence when the FOE is obtained one also obtained an arbitrarily scaled speed vector in camera frame. Using epipolar geometry the scaled speed in camera frame can be obtained by solving the following system, with flow, point direction vectors and unknown speed vector.

$$\begin{pmatrix} (p_1 \times \dot{p}_1)^T \\ p_2 \times \dot{p}_2)^T \\ \dots \\ p_n \times \dot{p}_n)^T \end{pmatrix} V = 0$$

Figure 4-7: Optical flow speed calculation by solving this system

The scaled optical flow based speed data is combined with IMU data inside the EKF framework. Scaled speed treated as a measurement in the EKF, by rotating the speed in global frame and multiplying it with the speed scale factor. This scale factor dependent from average scene depth and is a state which is estimated by the EKF.

The authors of (Weiss et al., 2012a) evaluate their method on speed and position accuracy. The velocity RMSE is less than 5 CM/Sec. The position was estimated during a 80 second

flight. At the end of the flight position was drifted for 20 centimeters. These results are considerably less than EKF VIO approaches, however the method is much less computationally intensive, with the most demanding operation in the EKF being a 3 by 3 matrix inversion. The authors therefore evaluate the algorithm as a backup mechanism to rely on when a more complex method such as PTAM fails. The computational efficiency of this method makes it also suitable for implementation on computationally constrained platforms.

4-2-4 Qualitative evaluation

Given the description in the previous paragraphs about the visual inertial state estimation methods now a summary can be made of each method's performance characteristics. The methods are rated on accuracy, computational efficiency and the reliability. This evaluation can then help with the selection for a visual inertial state estimation approach for the drone racing system. It has to be noted tough that full objective performance indicators can only be determined by implementing all methods in a test framework, where they are subjected to the same test datasets. To limit the comparison task to a more comprehensible scale it is chosen to perform a qualitative rating, based on each method's characteristics as reported by literature.

From the discussed SLAM methods is considered the most accurate method is ORB SLAM, since this method has an integrated loop closure mechanism. This refines the global map when e previously visited place is visited. The accuracy and computational performance of the odometry methods of MSCKF and EKF VIO is complementary(M. Li & Mourikis, 2012),where the best performing method depends on the number of tracked features. The optic flow speed estimation method is the least accurate, because it does not use local or global refinement methods and is semi-tightly coupled.

Computational efficiency is evaluated based on what hardware the algorithms have proved to work. From all methods ORB SLAM has proved to be the most computational intensive and only works on high performance desktop computers(Martinez-Carranza, Loewen, Mirquez, Garcia, & Mayol-Cuevas, 2016). All other methods are capable of running on-board a ARM processor. The most computationally efficient method is optic flow speed estimation. This approach is lightweight because it does not perform global optimizations and does not estimate feature positions.

Visual inertial navigation methods can have different types of failure modes, which affects the reliability of such a method. Here feature based methods such as PTAM and MSCKF can lose tracking when in environments which lack corner points. Although ORB SLAM is also a feature based method, it is very tolerant to tracking loss. When tracking is lost, due to camera occlusion for example, than the method can use its map to quickly relocalize itself when images are available again.

Table 4-1: Qualitative analysis of the most important monocular visual inertial navigation methods

	Accuracy	Computational efficiency	Reliability
PTAM	++	++	+
ORB SLAM	+++	+	+++
LSD SLAM	++	++	++
MSCKF	++	++	++
EKF VIO	++	++	++
Flow speed	+	+++	++

Chapter 5

Trajectory generation and control

5-1 Trajectory generation

If the environment is partly or fully known by the drones internal state estimation and navigation algorithms, then this knowledge can be used for generation trajectories throughout that part of the environment. The planning of a dynamical feasible trajectory of a UAV in a high speed flying scenario can be based on multiple different goals. Trajectories can be optimized for least energy consumption, least amount of time and other constraints. In the next paragraphs a division is made between polynomial based planning using differential flatness, sample based planning where the search space is explored in a probablistic way and motion primitive planning.

5-1-1 Polynomial trajectory generation

The quadrotor type of MAV used in this research has a certain characteristic which are relevant for trajectory generation. The quadrotor is proved to be a differentially flat system(Mellinger, 2011). A system is called differentially flat if the inputs can be written as function of one or more flat output variables and their derivatives. Trajectories can than be designed in the flat output space. Also the formulation allows to calculate the required control inputs for a given set of flat outputs, which can be used for controller design. In (Mellinger, 2011) the position and yaw angle are selected as flat outputs. In this work a polynomial minimum snap trajectory is planned through multiple waypoints. It is chosen to minimize the fourth order position derivative or snap to generate a smooth trajectory. Other Approaches which make use of differential flatness to optimize polynomial trajectories are(Loianno et al., 2016),(Jamieson & Biggs, n.d.) and(Richter, Bry, & Roy, 2013).

5-1-2 Sample based

One way of solving the trajectory generation problem is the sampling based approach. These approaches include RRT (SM Lavalle, 1999) and Probabilistic Road Maps (PRM)(Kavraki,

Vestka, Latombe, & Overmars, 1996). These methods are useful for planning in high dimensional state spaces.

PRM consists of a learning phase and a query phase. In the learning phase a graph is made containing nodes and possible paths between them in the collision free part of the search space. During this process care is taken that the nodes are also covering difficult parts of the search space. The possible connections between nodes are determined by a local planner, which calculates paths at a high rate. Next in the query phase the graph that was created in the learning phase is used to get a close to optimal trajectory.

In RRT the trajectory is found by growing a tree of feasible solutions in the search space. The tree is created by the following iterative provess: First a random point is generated in the state space, then the nearest vertex in the current tree is selected. After that a control input u is generated to minimize the distance between the random point and the new point. Finally a new vertex and edge are added to the tree. In contrary to PRM, RRT gives the globally optimal result, however this is at the cost of higher computational complexity. An example of RRT in MAV flight is given in(Richter et al., 2013). To limit the computational resources RRT is used only for a straight line search. The straight line trajectory is then further optimized in a quadratic programming.

5-1-3 Motion primitive based

Another approach for trajectory generation is the use of so called motion primitives. In this method a number of small and feasible trajectory segments are used to design a larger feasible trajectory. The motion primitives can be generated offline and stored on-board of the drone in a library. In flight the drone only has to choose between the trajectory segments in the library to come closer to the goal and evade potential obstacles. This has as an advantage that the trajectories do not have to be calculated in realtime, which would otherwise be very computational intensive. However a drawback is that not all possible trajectories are considered during the planning and there may is a more optimal choise outside the library(Paranjape, Meier, Shi, Chung, & Hutchinson, 2013).

Multiple examples can be given of the use of motion primitives int autonomous drone flight. In (Frazzoli, 2002) a framework for uav motion planning is proposed which is based on the interconnection of a finite number of motion primitives. The motion primitives in this work concist of so called trim trajectories and maneuver trajectories. During a trim trajectory the state of the vehicle is steered to the trim position with constant body velocity and control inputs. A maneuver trajectory is a trajectory that connects two trim trajectories. Trim and maneuver trajectories are used sequentially.

An extension to motion primitive based planning algorithms is the use of so called funnel libraries(Tedrake, Manchester, Tobenkin, & Roberts, 2010). The funnel approach compared to the general motion library approach also takes into account potential disturbances or sensor noise. In (Majumdar & Tedrake, 2013) an online implementation is discussed. After generating the motion primitives, the library is augmented whith feedback controllers that locally stabilize the trajectory. Then a funnel around the trajectory is calculated which garantees stability of the closed loop system, taking into acount uncertainties caused by sensor noise, initial conditions and model mismatch. A global trajectory is generated by interconnecting multiple funnels.

5-2 control

The control problems that have to be solved for the autonomous racing drone project are numerous. First of all the drone has only limited visual feedback throughout the course, because there are scenarios where the next gate is not yet in view, after passing the current gate. These periods of vision blackout have to be traversed by either performing open loop feedforward control, or employing an inertial navigation dead reckoning approach. Which method works optimal is dependent of the difference in accuracy of model based trajectory prediction on the one hand and the quality of the ins solution. Therefore the control consists of a feedback part and if model prediction is used, a feedforward part is also included. In the next paragraph an analysis of feedback and feedforward control techniques is given.

5-2-1 Feedback control

PID

A common way to solve linear or local linear control problems is Proportional Integral Derivative (PID) control. This method however assumes that the system is linear, hence the performance is dependent of the level of nonlinearity in the system model. However in practice it turns out that most systems even with some level of non-linear behavior can still be controlled by a PID controller. A quadrotor drone is a highly nonlinear system but still PID is commonly used as attitude or trajectory controller on quadrotor drones, such as(Salih, Moghavvemi, Mohamed, & Gaeid, 2010),(Sadeghzadeh & Mehta, 2011) and(Zul Azfar & Hazry, 2011). This is because at small attitude angles and low speeds the system can be approximated by a linear model(Bolandi, Rezaei, Mohsenipour, Nemati, & Smailzadeh, 2013). The racing scenario is not characterized by low speed and small attitude angles. Therefore it has to be investigated if this local non-linear assumption is still valid at race speed.

LQR

A linear Quadratic regulator is part of optimal control theory. The LQR controller tries to minimize a quadratic cost function which is based on the desired performance characteristics of the system. LQR control can be used for different control tasks ralated to MAV flight. Examples of LQR implemntations can be found in(Panomrattanarug, Higuchi, & Mora-Camino, 2013), where attitude control is performed. In(Jafari, Zareh, Roshanian, & Nikkhah, 2010)and(Cowling, Yakimenko, Whidborne, & Cooke, 2007) an LQR trajectory controller shows good performance.

NDI

Nonlinear Dynamic Inversion (NDI) uses the nonlinear model to linearize the control problem(Enns, Jski, Hendrick, & Stein, 2017). This method has a better performance than the common gain scheduling approach for controlling nonlinear systems. NDI is however sensitive to model mismatch, which requires an accurate aerodynamic model and kinematic model. Such an aerodynamic modle can be derived by interpolating data points from a large

aerodynamic database. For the kinematic model center of gravity position and moment of inertia are important. To make NDI more robust, a novel nonlinear control method is introduced called Incremental Nonlinear Dynamic Inversion (INDI)(Sieberling, Chu, & Mulder, 2010). INDI is based on NDI but instead of calculating the control signal every time step, INDI only calculates the increment in control signal needed. The rotational equations of motion are rewritten in an incremental form, which has as a result that the time scale separation principle can be applied. The slow dynamics of the system, such as position change or yaw can be separated from the fast dynamics of angular rotations and attitude changes. The incremental form of NDI is much more tolerant to model mismatches as wel as actuator faults.

5-2-2 Feed forward

If the inertial navigation solution proves to be not accurate enough in the presence of large discontinuities in the vision based gate tracking feedback, than feedforward control can be employed to bridge the gap between two gate detection zones. The problem amounts then to find a sequence of control commands to execute a planned trajectory to the next gate detection zone, given the vehicle state at the start of the trajectory. Errors in the initial state and errors during the execution of the maneuver build up over time. Therefore this type of control can only be used for a short period.

Chapter 6

Literature discussion

In this chapter the main findings of the literature search are discussed. The literature search first evaluated general approaches to the problem of high speed autonomous drone flight. After that the main methods in the topics of sensing, state estimation, trajectory generation and control were reviewed.

6-1 General high speed flight

The general problem of high speed autonomous drone flight is an area of research since more than a decade. The overall complexity of the problem has resulted in a large number of different approaches to the problem. Early approaches made use of off-board sensing, state estimation and processing. However these approaches are far from fully autonomous, therefore later work aims at performing all sensing and processing task on-board of the drone. A possible approach for on-board navigation is by employing a LIDAR sensor. Although the LIDAR based navigation approach proved a vailable solution to the autonomous flight problem, the relative large size and weight of typical LIDAR sensors pose a severe limitation for use on small drones. Also LIDAR methods are generally limited to well structured environments(Shen, Michael, & Kumar, 2011). As an alternative for LIDAR also vision can be used as a basis for state estimation, since camera sensors are more lightweight. Both stereo vision based approaches as well as monocular based approaches were tried. Stereo and monocular methods provided similar results, however monocular cameras have the advantage that they can even be mounted on even the smallest drones(Loianno et al., 2016).

What all these approaches have in common is that they only consider short high speed maneuvers lasting only a few seconds. It is not investigated what will happen if the high speed flight lasts for longer periods of time. For example if sensor bias errors are being estimated properly. The drone racing scenario forms a perfect setting for testing this. Also in the drone racing scenario higher level decision making is needed to successfully complete the course. At some parts of the track there might be multiple gates in sight, and the drone needs to quickly choose where to go. Also the drone racing system should be robust to false detections through distractors in the background.

6-2 Sensing

In this literature review all on-board sensors of the drone used in the current research were evaluated on their specific characteristics. The most important sensors on-board the drone are the inertial measurement unit and two monocular cameras. The low cost lightweight IMU sensor type on-board typical consumer drones has low accuracy, high noise and drifting bias errors. Results can be improved by fusing the sensor data with other sensors such as cameras. Monocular cameras on their own can not sense absolute depth or motion. However still much important data can be obtained by studying the optical flow field in the moving camera image. The optical flow field can be used for determining things like the direction of travel, or the time to collision with an object. Various methods exist for extracting optical flow from a sequence of images, the most frequently used method being subsequent corner detection and tracking with the Lukas Kanade algorithm. Most methods are dependent on the availability of corner features, which might not be available in certain scenes. Alternatively a new method for estimating the global optical flow was developed in(McGuire et al., 2016). This method uses an edge image and a histogram peak approach, which consumes exceptionally low computational resources.

Other means of employing the camera for state estimation purposes is in the Perspective-n-point problems(Pnp). The problem amounts to estimating the camera pose from viewing a number of world points with known or unknown locations. When the point locations are known, than the minimum number of points necessary for a solution is 3 (P3p). In the drone race scenario square gates of known size can thus be used as a computational efficient method for localization and attitude estimation. Therefore in the next chapter a preliminary analysis will be conducted about the performance characteristics of P3p.

6-3 State estimation

State estimation in general as well as multiple monocular visual inertial state estimation techniques were discussed, including a qualitative evaluation of the latter. Particularly the Kalman filter is explained, including the most important variations, being the Extended and Unscented Kalman filter types. These filter types are capable of performing sensor fusing tasks in nonlinear and highly non-linear problems.

The visual inertial state estimation problem can be approached with two different fundamental methods, being SLAM and odometry. SLAM methods simultaneously locate the robot and try to build a map of the environment, while odometry methods try to incrementally reconstruct the path of the robot in a locally consistent way. Odometry method are more sensitive to the incremental buildup of small errors over time, however this could be compensated by other visual methods. One other way to classify the SLAM and odometry methods is through how visual data is integrated into the filter. A distinction can be made between feature based methods, direct methods and semi-direct methods. Feature based methods rely on the detection and tracking of salient features. Direct methods work by directly optimizing pixel intensity values. Generally feature based methods are more computationally efficient, while direct methods work better in environments with limited salient features to track. A recent development are semi-direct methods, which have the computational efficiency of feature based methods, while also working in feature sparse environments.

Both SLAM and odometry methods were evaluated. Commonly used SLAM methods are PTAM, ORB-SLAM and LSD-SLAM. However only PTAM proved to be computationally efficient enough to run on-board a drone. The odometry methods MSCKF and EKF VIO have complementary characteristics in accuracy and computational efficiency(M. Li & Mourikis, 2012). From the odometry methods the optical flow speed estimation approach(Weiss, Achtelik, Lynen, Chli, & Siegwart, 2012b) is the most computational efficient. This makes it an interesting method suitable for integration in the current drone or even smaller drone types.

As a minimalistic approach to estimating the position of the drone it is chosen to fuse 3D position based on the P3p algorithm with data from the IMU inside an Extended Kalman Filter framework. The results of this analysis will be presented in the next chapter. The estimated state can then later be used for trajectory planning and control tasks.

6-4 Trajectory generation and control

When an estimation of the MAV's state is known and the race track is known, than still the question remains how to generate and track the optimal trajectory. Multiple approaches to trajectory generation were investigated. A common class of trajectory planning methods are sample based methods such as RRT and PRM, where the state space is searched in a probabilistic way. These methods are good at exploring the state space and generally are able to find a close to optimal solution(Kavraki et al., 1996). The downside of these methods if that they are computationally intensive(Boeuf, Cortes, Alami, & Simeon, 2015), while autonomous MAV flight is bounded by the onboard computational constraints. Therefore implementations on sample based MAV trajectory generation are generally performed off board.

An alternative method to the trajectory generation problem is the use of motion primitives. The trajectory is then assembled from a finite number of feasible trajectory segments from a library. This method is less computational intensive since the library of motion primitives can be calculated offboard in advance, while the actual trajectory generation can be performed onboard the MAV. However this comes at the cost considering only part of the search space(Paranjape et al., 2013). A recent extension to motion primitives is the funnel based approach(Tedrake et al., 2010) this method also takes into account the various uncertainties by using funnels where the robot is guaranteed to stay inside.

One other trajectory generation approach which is often seen in autonomous MAV flight is a polynomial trajectory generation method(Loianno et al., 2016),(Jamieson & Biggs, n.d.) and(Richter et al., 2013). These methods use the differentially flatness theory to optimize the trajectory in the flat output space. As a additional criteria the trajectories are often minimized to minimize snap, in order to guarantee smooth trajectories. This method relies on existing waypoints, however if these are not available a simple straight-line RRT can be used first to generate them(Richter et al., 2013).

Once the trajectory is established a controller is needed to track the trajectory. PID and LQR control are commonly used for both inner loop attitude control as well as trajectory tracking control. These methods rely on a linear system model. If aggressive maneuvers are to be performed, as is the case in the drone racing scenario, then the linear model might not be valid anymore. This can be solved by PID gain scheduling(Sadeghzadeh & Mehta, 2011)

or other nonlinear control theory. Recent work in Nonlinear Dynamic Inversion as well as its incremental form INDI have shown good results as controllers onboard MAV's(Xie, Xia, & Fu, 2011).

Chapter 7

Preliminary results

7-1 Experimental Setup

As a preliminary analysis a minimal approach to visual inertial state estimation is considered. The P3p algorithm is used as a main source of position. Pixel noise from the camera propagates through the algorithm which results in noise in the position measurement. Additionally the camera runs at a lower frame of 30Hz. That combined with possible missed or false detections explains the need for improving the state estimation performance by fusing it with inertial data. The sensor fusing part will than be performed in an Extended Kalman Filter, due to the nonlinear characteristics of the problem. The performance of the Perspective-3-point method is first evaluated separately, after which these results are processed in the EKF state estimation filter. The analysis will be conducted in Matlab and involves both artificial data, as well as real data from onboard of the drone. The data from the drone is obtained by performing a number of test flights in a dedicated testing area of the MAV Lab, called the Cyberzoo. The Cyberzoo is a 10X10X10m space which is protected by safety nets. It is equipped with a high performance infrared camera tracking system called Optitrack, which is able to determine the position of the MAV at sub millimeter accuracy at 30HZ data rate. The data from the Optitrack system is used as groundtruth during the experiments.

The flight tests will be performed with the Bebop drone as described earlier. The drone is able to log sensor data such as IMU readings at 500HZ rate and camera images at 30HZ in its onboard memory. After a flight the logged data is downloaded and analysed in Matlab.

7-2 Vision

7-2-1 Gate corner detection

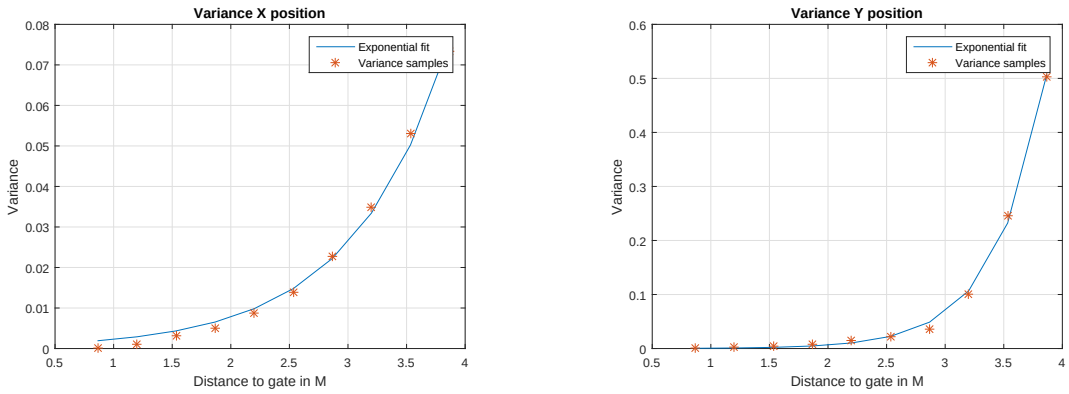
The perspective-3-point method is based on viewing the known locations of the gate corner points. Therefore first the four corner points of the gate have to be found. Because the gate

is coloured, a colour filter is applied followed by a coarse search for a square like shape. The square is identified by randomly sampling the image for coloured pixels and performing a snake like pattern of up, down and left, right motions along stretches of coloured pixels in the image. However the initially estimated gate corner positions still have to be refined to compensate for the orientation of the camera and lens distortion. This is performed by a histogram based corner refinement. After that the corner point locations in image frame are undistorted for radial lens distortion and passed to the P3p algorithm.

7-2-2 P3p

A perspective-3-point algorithm will be used to determine position and attitude of the MAV. When the 3D world coordinates of the gate corner points are known, then position and attitude can be determined based on the projection of these points onto the image frame. The P3p algorithm as described in (Kneip et al., 1991) is implemented in a ransac approach by back projecting the fourth gate point of different 3 corner point sets. The method is tested on simulated data of a gate which is viewed at different angles and distances. Statistical metrics include the Root Mean Squared Error (RMSE), as well as the variances of position and attitude errors.

Results as function of distance:



(a) Position variance in x direction

(b) Position variance in y direction

Figure 7-1: X and Y position variances as function of distance to the gate

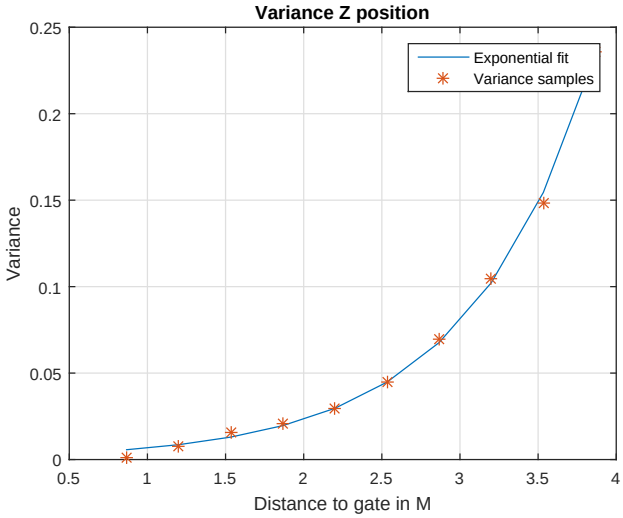


Figure 7-2: Position variance in z direction as function of distance to the gate

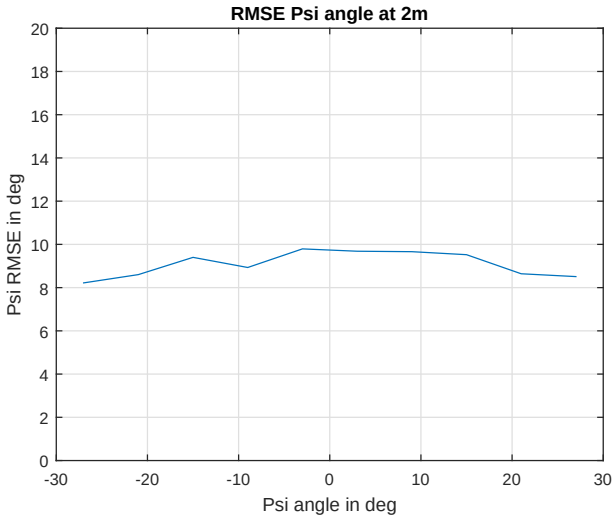


Figure 7-3: RMSE of Psi angle as function of Psi heading angle

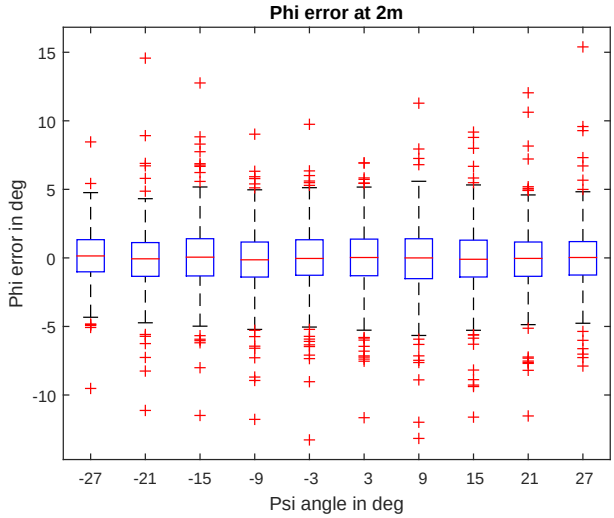


Figure 7-4: Phi error as function of Psi heading angle at 2m distance

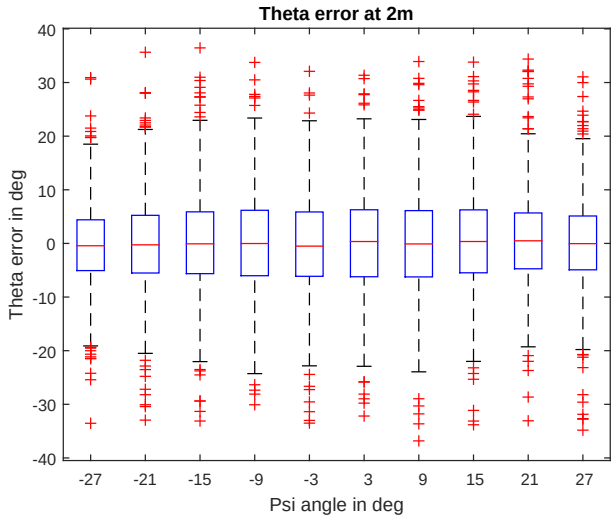


Figure 7-5: Theta error as function of Psi heading angle at 2m distance

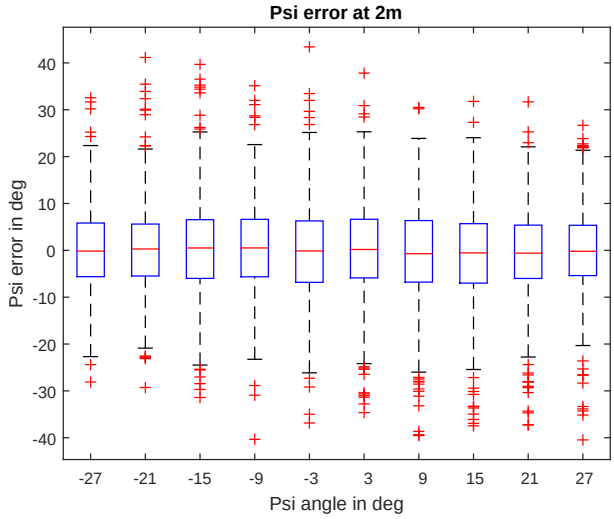


Figure 7-6: Psi error as function of Psi heading angle at 2m distance

The position estimation based on P3p only is subject to pixel noise from the camera. It is expected that a larger distance between the camera and gate will also result in a larger error. In figure 7-1 and 7-2 the error variances of x y and z position are plotted as function over distance. It has been observed that the error variance characteristics are well fitting an exponential function. This knowledge may be later used to integrate into the state estimation filter to improve performance. Additionally also the possibility for cross coupling between the psi heading angle and the attitude angle error is investigated, to verify the method's consistency. In 7-3 the RMS error in heading is shown as a function of the heading angle between positive and negative 30 degrees. This shows that there is only a minor sensitivity in heading, with the error remaining between 10 and 8 degrees. Also the figures 7-4, 7-5 and 7-6 supports this notion.

7-3 State estimation

In the previous section position and attitude data is estimated based on the known corner point locations of a gate, using the P3p algorithm. Now to improve accuracy the P3p position will be fused with inertial data from the IMU. During the test the MAV was manually flown through a gate while logging the on-board sensor data. In Matlab an Extended Kalman filter is used for state estimation. On-board inertial data is fused with simulated noisy vision data to estimate position, attitude and sensor biases.

The test flight consists of a take-off, forward flight and a landing part. During the take-off, the filter is initialised with accurate Optitrack position data. In the forward flight the filter uses the P3p position data and the drone flies through the gate. After passing the gate the P3p position is turned off and the filter continues estimation, based on inertial data only. After that the drone lands again. A side view of the drone's trajectory during the test flight can be seen in 7-7

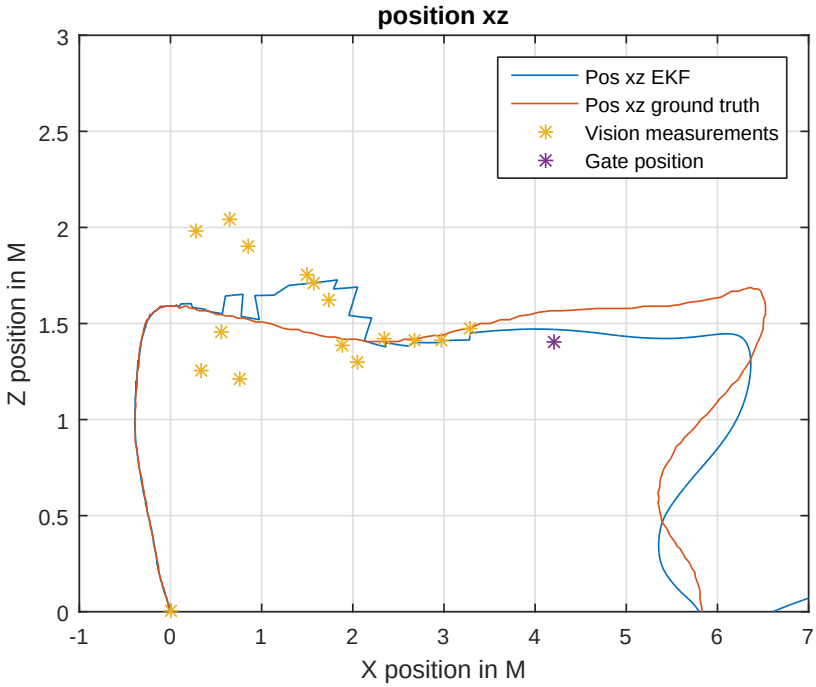


Figure 7-7: XZ position EKF estimate

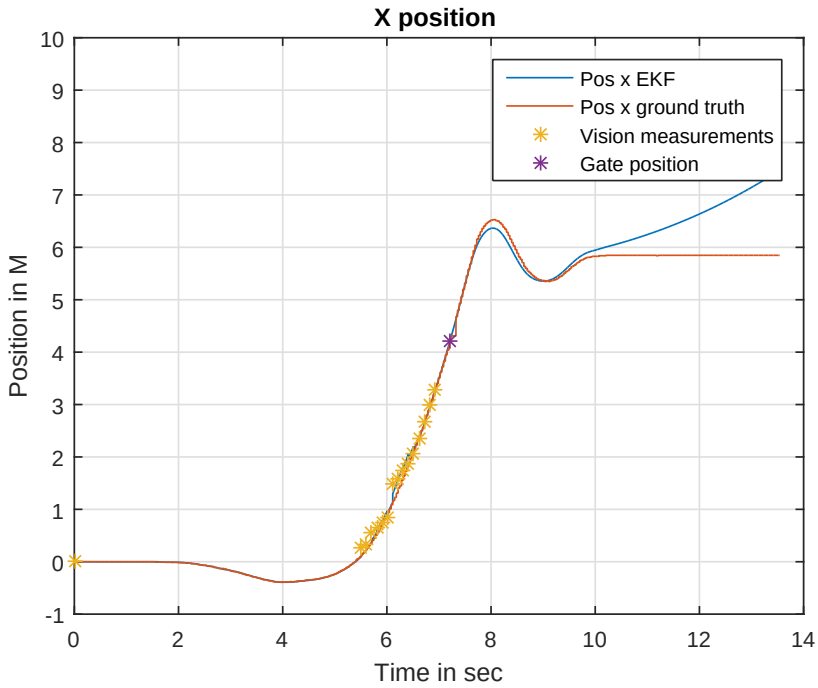


Figure 7-8: X position EKF estimate

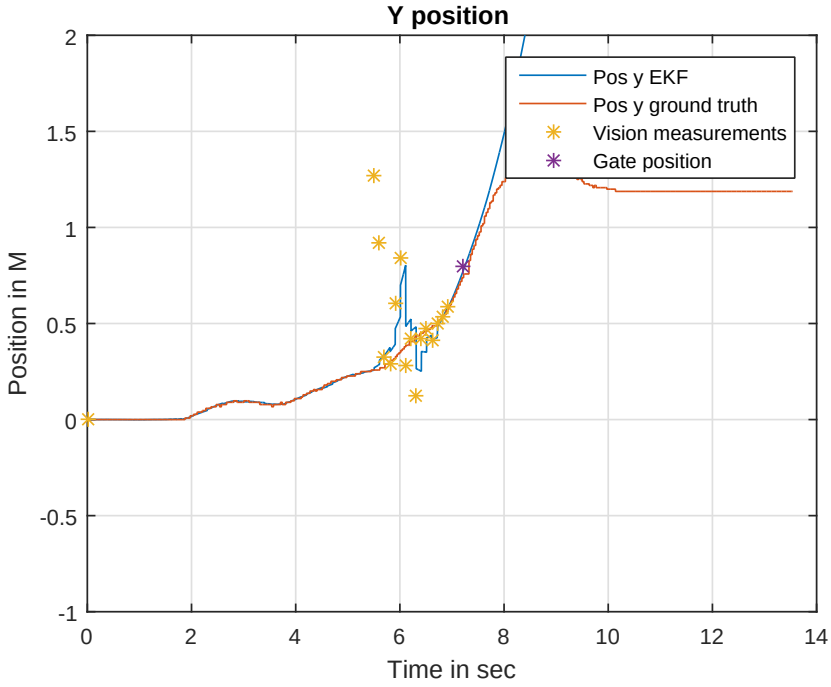


Figure 7-9: Y position EKF estimate

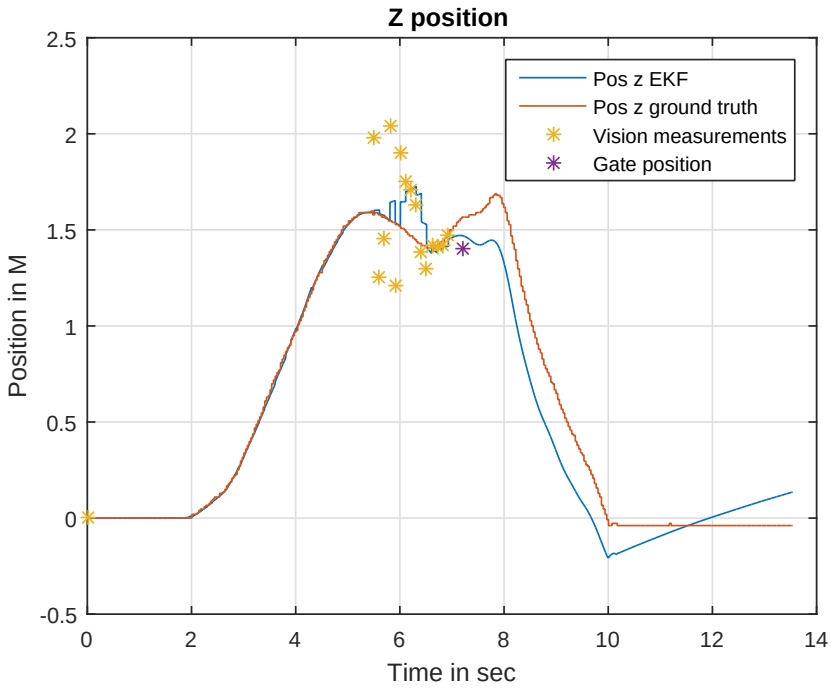


Figure 7-10: Z position EKF estimate

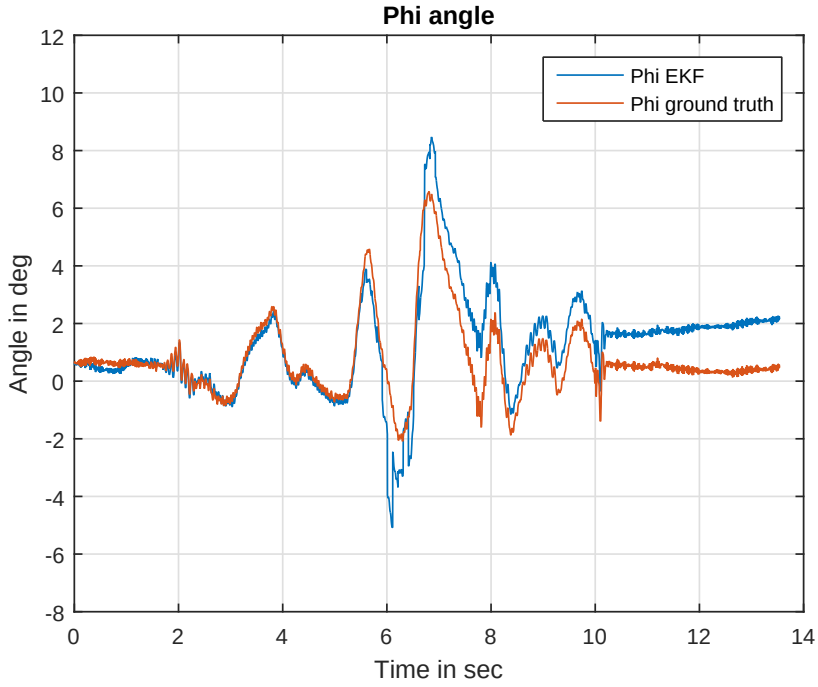


Figure 7-11: Phi angle EKF estimate

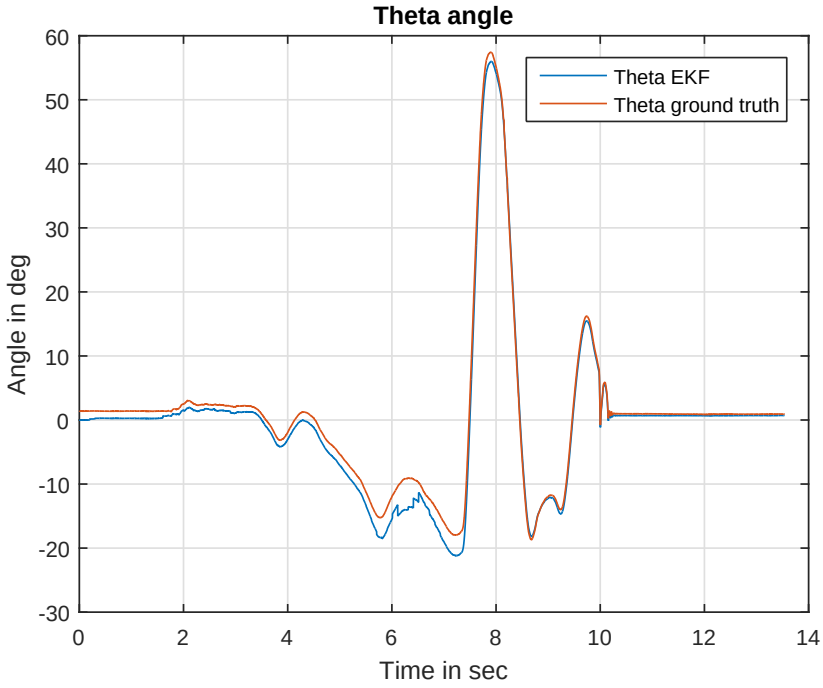


Figure 7-12: Theta angle EKF estimate

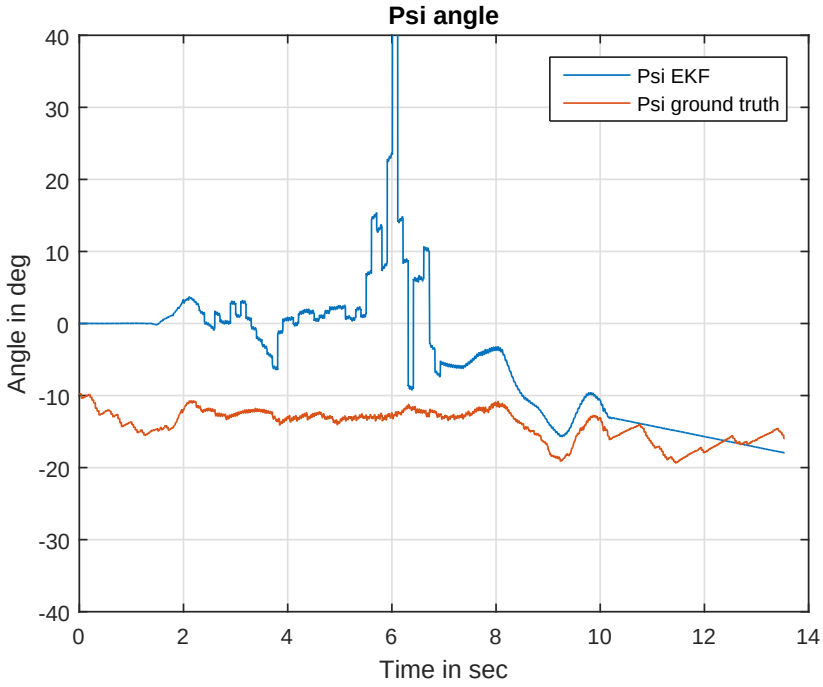


Figure 7-13: Psi angle EKF estimate

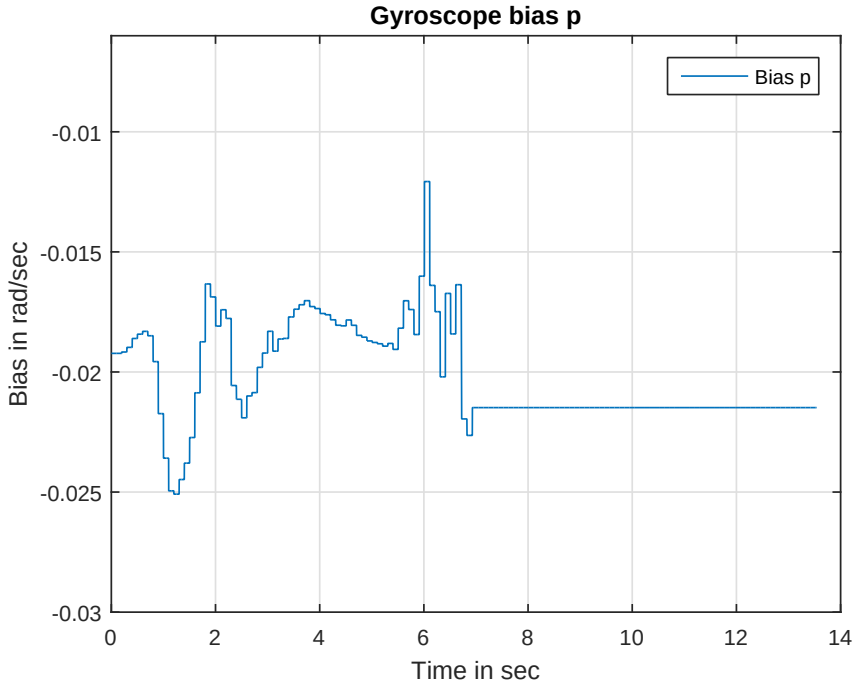


Figure 7-14: Gyroscope bias p EKF estimate

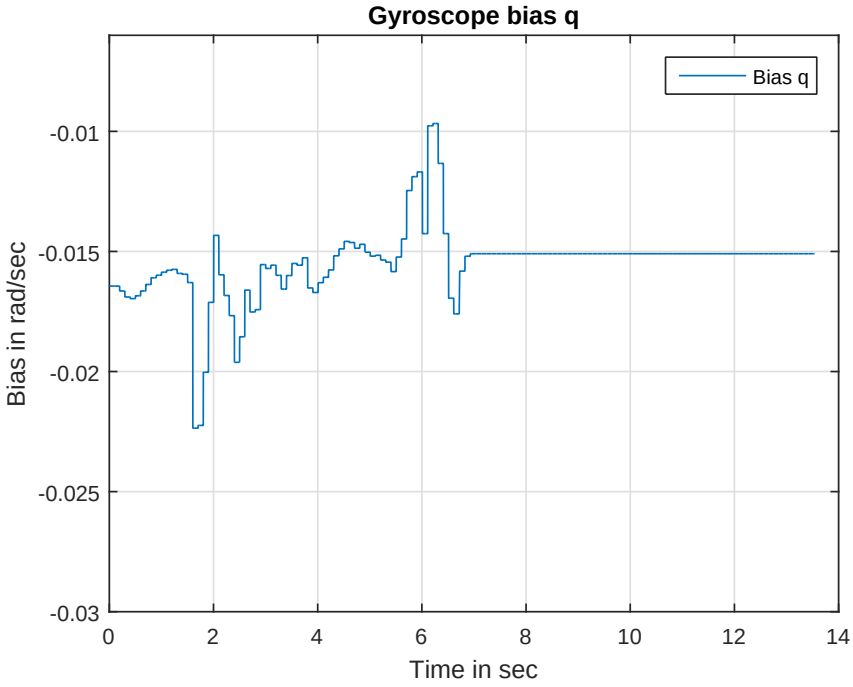


Figure 7-15: Gyroscope bias q EKF estimate

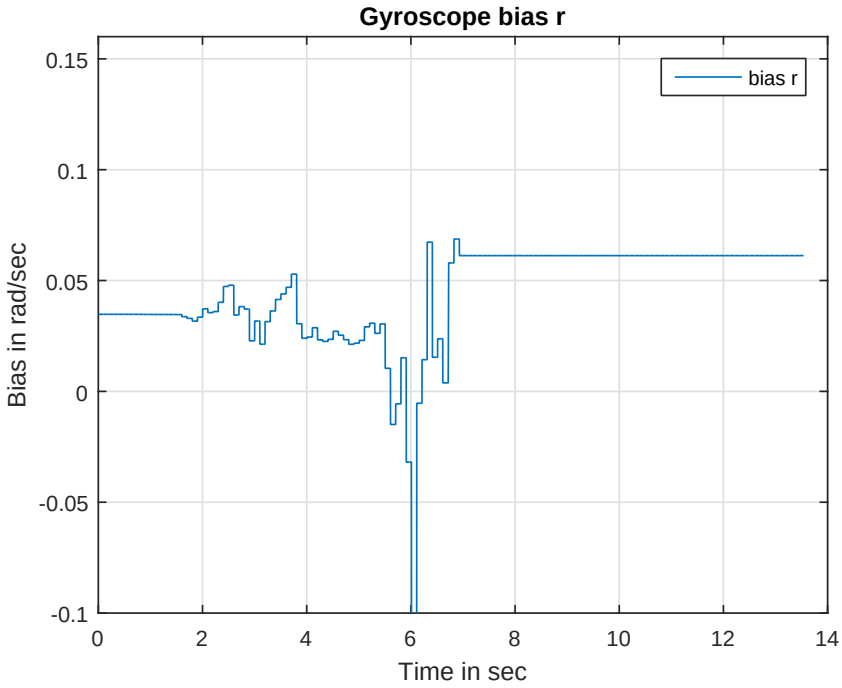


Figure 7-16: Gyroscope bias r EKF estimate

The results from the EKF state estimation with real flight test data look promising. In 7-7 the first part shows almost perfect correspondence between the estimated position and the

ground truth position, as would be expected because high accuracy position information is used to initialize the filter. When the forward flight starts, position measurements switches to the P3p method. The vision based position measurements are denoted by the yellow asterisks. It can be observed that after a short time the position converges towards the ground truth value. The step like behaviour of the predicted signal is caused by the high noise and low data rate of the P3p gate detection and the measurement updates of the filter only performed at this low rate.

While flying towards the gate also sensor biases and attitude are estimated. In 7-14 to 7-16 the rate gyroscope sensor biases are plotted. The p, q and r biases seem to converge back to their original value. It can also be observed that the psi angle shows a large error, before converging to towards the groundtruth value. This can be explained by noting that the yaw angle 7-13 is only indirectly observable when the vehicle is moving. Also the high noise levels in the vision data pose an extra challenge to the estimation.

After passing the gate the detection stops and the filter is essentially an inertial navigation system which integrates bias compensated inertial data. It can be seen in 7-8 that the error after a few seconds does not grow larger than 20Cm. However in 7-9 the y direction position estimate error quickly grows. It is assumed that this is the result of a noise spike during the violent braking maneuver performed after passing the gate.

7-4 Discussion

In the current chapter the Perspective-3-point algorithm was evaluated and integrated in an EKF framework where position measurements are fused with IMU data. One important finding is that there is an exponential relation between the distance to the gate and the noise level of the P3p method. This result can be used to improve the state estimation method, for example by making the error variance in the Kalman Filter dependent from estimated distance. Experiments with the Extended Kalman Filter show that it is possible to perform inertial only navigation for short periods of time.

Chapter 8

Conclusion

In this preliminary thesis report a literature review and early experimental results are presented towards the development of an autonomous drone racing system. In the literature review previous work in the area of autonomous high speed drone flight is discussed. Also a review is given of the on-board sensors, as well as methods for visual inertial state estimation. Finally trajectory generation and control are described.

Previous work has first focussed on trajectory planning and control by using an external positioning system and off-board processing. Later approaches did use onboard sensing and processing. However the LIDAR sensors initially used were too heavy to be carried by small drones. Alternatively on-board cameras were used for navigation purposes which reduces weight and form factor. One thing that can be noticed in the previous work is that current examples of high speed autonomous drone flight only consider short maneuvers. Also they lack in a fast high level decision making process. These subjects can be investigated in the autonomous drone racing scenario.

The primary sensors on-board the MAV are the inertial measurement unit and the camera. The low cost lightweight IMU is highly susceptible to noise and bias errors, therefore adequate filtering is especially important. The monocular camera sensor is the primary means of navigation of the drone. Optical flow data from the camera can estimate useful information such as direction of travel and time to contact. However to estimate absolute depth and motions, the camera data has to be combined with other sensor data.

The monocular camera and the IMU data can be combined in a state estimation framework to provide speed, position and attitude. Both SLAM and odometry methods can be used to solve this problem. SLAM methods aim to create a globally consistent map of the environment, while localizing the drone in the environment. Odometry methods only try to reconstruct the robots path in an iterative way, which makes them more computationally efficient. A method which is an interesting candidate for integration in the drone race system is an odometry, method were optical flow measurements are fused with IMU measurements in an EKF framework.

Trajectory generation can be performed by computationally intensive sample based methods such as Rapidly Randomized Tree(RRT) and Probabilistic roadmaps. However if an

onboard implementation on the MAV is required, more computationally approaches are required. These include motion primitive methods as well as minimum snap polynomial based trajectory planning. Control can be performed by classical linear methods such as PID and LQR, but in a high speed aggressive flight nonlinear control methods such as Nonlinear Dynamic Inversion as well as its incremental form INDI is preferable.

The preliminary results reviewed a minimal viable approach to the drone racing scenario, where the position is obtained by a Perspective-3-point algorithm. The position measurements are then fused with IMU measurements in a Extended Kalman Filter. Experiments have shown that the filter is able to estimate position attitude and sensor biases based on visual and inertial data. When visual data is lost, bias compensated IMU readings can still be used for short term inertial navigation. There is however much room for improvements, such as better state estimation when between two gate detections. Also a global path planning and control method should be implemented.

The results from the literature search and experiments in this preliminary thesis will be used to further develop the autonomous drone racing system. In the next chapter a more detailed explanation is given about the future work that will be conducted to develop this system.

Chapter 9

Planning

The preliminary results in this report only concidered a minimalistic solution to the autonomous drone racing system. Improvements can be made in the areas of sensing, state estimation and trajectory planning.

9-0-1 Sensing

In the preliminary results the method for determining position and attitude with respect to a gate was a Perspective-3-point method. Although the method showed prommising results when combined with inertial data, the measurements are far from perfect. At a larger distance small pixel noise is amplified to relative high position and attitude errors. To improve the method prior knowledge about the state of the vehicle has to be taken into acount. The attitude and heading angle are generally known within few degrees, also the distance to the gate can be estimated roughly by looking at the overall size. This information will be used to improve the accuracy of the gate based position measurements.

Annother improvement that will be made is to the detection process of the gate. The estimated previous state of the drone can be used to predict the new location of the gate in image frame. This prediction will increase the reliability of the gate detection.

9-0-2 State estimation

The current approach which was evaluated in the preliminary results relied on gate detections for position measurements. When no gate detections are available, the system is essentially navigating using deadreckoning. This approach is only accurate for a very short period of time. Therefore also a navigation method is needed that works in between gate detections. Therefore it is chosen to implement a computationally lightweight odometry method which fits the computational constraints on-board the drone. This method fuses de-rotated optical flow vectors in an EKF framework. The method also has to be adapted to include the position and heading estimates based on the gate detection.

9-0-3 Trajectory planning and control

When the current position speed and attitude of the drone are known, still the question of how to fly the required trajectory remains. In the drone racing scenario the track is previously known. The track can be seen as a number of waipoints tht have to be passed, with each waypoint positioned at the center of the gate. The problem then amounts to finding a time optimal trajectory through these waypoints. For this an arc based motion primitive trajectory planning method will be investigated.

Bibliography

Barea, R., Bergasa, L. M., & Molinos, E. J. (n.d.). Indoor SLAM for Micro Aerial Vehicles Control using Monocular Camera and Sensor Fusion.

Bloesch, M., Omari, S., Hutter, M., & Siegwart, R. (2015). Robust visual inertial odometry using a direct EKF-based approach. *IEEE International Conference on Intelligent Robots and Systems*, *2015-Decem*, 298–304.

Boeuf, A., Cortes, J., Alami, R., & Simeon, T. (2015). Enhancing sampling-based kinodynamic motion planning for quadrotors. *IEEE International Conference on Intelligent Robots and Systems*, *2015-Decem*, 2447–2452.

Bolandi, H., Rezaei, M., Mohsenipour, R., Nemati, H., & Smailzadeh, S. M. (2013). Attitude Control of a Quadrotor with Optimized PID Controller. *Intelligent Control and Automation*, *4*(August), 335–342. Available from `http://dx.doi.org/10.4236/ica.2013.43039%5Cnhttp://www.scirp.org/journal/ica`

Boutayeb, M., Rafaralahy, H., & Darouach, M. (1995). Convergence analysis of the extended Kalman filter as an observer\nfor nonlinear discrete-time systems. *Proceedings of 1995 34th IEEE Conference on Decision and Control*, *2*(December), 1555–1560.

Bry, A., Richter, C., Bachrach, A., & Roy, N. (2015). Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. *The International Journal of Robotics Research*, *34*(7), 969–1002. Available from `http://ijr.sagepub.com/content/early/2015/03/19/0278364914558129.abstract`

Civera, J., Davison, A. J., & Montiel, J. M. M. (2008). Inverse Depth Parameterization for Monocular {SLAM}. *IEEE Transactions on Robotics*, *24*(5), 932–945.

Cowling, I. D., Yakimenko, O. a., Whidborne, J. F., & Cooke, A. K. (2007). A Prototype of an Autonomous Controller for a Quadrotor UAV. *European Control Conference*, 1–8.

Engel, J., Sch, T., & Cremers, D. (2014). LSD-SLAM: Direct Monocular SLAM. *European Conference on Computer Vision (ECCV)*, *8690 LNCS*(PART 2), 834–849.

Enns, D., Jski, D. A. N. B., Hendrick, R., & Stein, G. (2017). Dynamic inversion: an evolving methodology for flight control design. , *7179*(May).

Falanga, D., Mueggler, E., Faessler, M., & Scaramuzza, D. (2016). Aggressive Quadrotor Flight through Narrow Gaps with Onboard Sensing and Computing.

Floreano, D., & Wood, R. J. (2015). Science, technology and the future of small autonomous drones. *Nature*, *521*(7553), 460–466. Available from `http://www.nature.com/doifinder/10.1038/nature14542%5Cnhttp://www.nature.com/nature/j`

Forster, C., Pizzoli, M., & Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. *Proceedings - IEEE International Conference on Robotics and Automation*, 15–22.

Frazzoli, E. (2002). Maneuver-based motion planning and coordination for multiple UAVs. *Proceedings. The 21st Digital Avionics Systems Conference*, *2*(May), 8D3–1–8D3–12. Available from `http://ieeexplore.ieee.org/document/1052947/`

Gao, X. S., Hou, X. R., Tang, J., & Cheng, H. F. (2003). Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(8), 930–943.

Gibson, J. J. (1950). The perception of the visual world. *Psychological Bulletin*, *48*(4), 1–259.

Harris, C., & Stephens, M. (1988). A Combined Corner and Edge Detector. *Procedings of the Alvey Vision Conference 1988*, 147–151. Available from `http://www.bmva.org/bmvc/1988/avc-88-023.html`

Hehn, M., & D'Andrea, R. (2011). Quadrocopter Trajectory Generation and Control. *IFAC Proceedings Volumes*, *44*(1), 1485–1491. Available from `http://linkinghub.elsevier.com/retrieve/pii/S147466701643819X`

Horaud, R., Conio, B., Leboulleux, O., & Lacolle, B. (2011). An analytic solution for the perspective 4-point problem - Computer Vision and Pattern Recognition, 1989. Proceedings CVPR '89., IEEE Computer Society Confer.

Jafari, H., Zareh, M., Roshanian, J., & Nikkhah, A. (2010). An Optimal Guidance Law Applied to Quadrotor Using LQR Method. *Transactions of the Japan Society for Aeronautical and Space Sciences*, *53*(179), 32–39. Available from `http://joi.jlc.jst.go.jp/JST.JSTAGE/tjsass/53.32?from=CrossRef`

Jama, M., & Schinstock, D. (2011). Parallel tracking and mapping for controlling VTOL airframe. *Journal of Control Science and Engineering*, *2011*.

Jamieson, J., & Biggs, J. (n.d.). Near minimum-time trajectories for quadrotor UAVs in complex environments.

Julier, S. J., & Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. *Proc. of AeroSense: The 11the Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls*. Available from `http://reviews.spiedigitallibrary.org/data/Conferences/SPIEP/39058/182_1.pdf`

Kavraki, L. E., Vestka, P., Latombe, J. C., & Overmars, M. H. (1996). *Probabilistic roadmaps for path planning in high-dimensional configuration spaces* (Vol. 12) (No. 4).

Klein, G., & Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*.

Kneip, L., Scaramuzza, D., & Siegwart, R. (1991). A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation.

Lee, D. (1976). A theory of visual control of braking based on information about time-to-collision. *Perception*, *5*(4), 437–459.

Lefebvre *, T., Bruyninckx, H., & De Schutter, J. (2004). Kalman filters for non-linear systems: a comparison of performance. *International Journal of Control*, *77*(7), 639–653. Available from http://www.tandfonline.com/doi/abs/10.1080/00207170410001704998

Li, M., & Mourikis, A. I. (2012). Optimization-Based Estimator Design for Vision-Aided Inertial Navigation. *Robotics: Science and Systems*(6), 1–8. Available from http://roboticsproceedings.org/rss08/p31.pdf

Li, S., Xu, C., & Xie, M. (2009). O (n). , *57*(5), 3826.

Loianno, G., Brunner, C., McGrath, G., & Kumar, V. (2016). Estimation, Control and Planning for Aggressive Flight with a Small Quadrotor with a Single Camera and IMU. *IEEE Robotics and Automation Letters*, *3766*(c), 1. Available from http://ieeexplore.ieee.org/document/7762111/

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, *2*([8), 1150–1157. Available from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=790410

Lucas, B. D., & Kanade, T. (1981). *An iterative image registration technique with an application to stereo vision.* Available from http://www.ic.unicamp.br/ rocha/teaching/2013s1/mc851/aulas/additional-material-lucas

Majumdar, A., & Tedrake, R. (2013). Robust Online Motion Planning with Regions of Finite Time Invariance. *Algorithmic Foundations of Robotics ...*, *86*(Section 4), 543–558. Available from http://link.springer.com/10.1007/978-3-642-36279-8%0Ahttp://link.springer.com/10.1007

Marquez, F., Garcia, E. O., & Mayol-cuevas, W. (2015). On Combining Wearable Sensors and Visual SLAM for Remote Controlling of Low-cost Micro Aerial Vehicles.

Martinez-Carranza, J., Loewen, N., Mirquez, F., Garcia, E. O., & Mayol-Cuevas, W. (2016). Towards autonomous flight of micro aerial vehicles using ORB-SLAM. *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems, RED-UAS 2015*, 241–248.

Maxbotix. (2015). *LV - MaxSonar ® - EZ Series.*

McGuire, K., Croon, G. C. H. E. de, Wagter, C. D., Remes, B., Tuyls, K., & Kappen, H. J. (2016). Local Histogram Matching for Efficient Optical Flow Computation Applied to Velocity Estimation on Pocket Drones. *CoRR*, *abs/1603.0*, 3255–3260. Available from http://arxiv.org/abs/1603.07644

Mellinger, D. (2011). Trajectory Generation and Control for Quadrotors. *2011 IEEE International Conference on Robotics and Automation*, 2520–2525.

Mellinger, D., Michael, N., & Kumar, V. (2014). Trajectory generation and control for precise aggressive maneuvers with quadrotors. *Springer Tracts in Advanced Robotics*, *79*, 361–373.

Mori, T., & Scherer, S. (2013). First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. *Proceedings - IEEE International Conference on Robotics and Automation*, 1750–1757.

Mourikis, A. I., & Roumeliotis, S. I. (2007). A multi-state constraint Kalman filter for vision-

aided inertial navigation. *Proceedings - IEEE International Conference on Robotics and Automation*, 3565–3572.

Mur-Artal, R., Montiel, J. M. M., & Tardos, J. D. (2015). ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, *31*(5), 1147–1163.

Panomrattanarug, B., Higuchi, K., & Mora-Camino, F. (2013). Attitude Control of a Quadrotor Aircraft Using LQR State Feedback Controller with Full Order State Observer.

Paranjape, A. A., Meier, K. C., Shi, X., Chung, S. J., & Hutchinson, S. (2013). Motion primitives and 3-D path planning for fast flight through a forest. *IEEE International Conference on Intelligent Robots and Systems*, 2940–2947.

Richter, C., Bry, A., & Roy, N. (2013). Polynomial trajectory planning for quadrotor flight. *International Conference on Robotics and Automation*(Isrr), 1–16. Available from `http://www.michigancmes.org/papers/roy7.pdf`

Sadeghzadeh, I., & Mehta, A. (2011). Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain-scheduled PID and model reference adaptive control. *Annual Conference of the Prognostics and Health Management Society*, 1–10.

Salih, A. L., Moghavvemi, M., Mohamed, H. A. F., & Gaeid, K. S. (2010). Modelling and PID controller design for a quadrotor unmanned air vehicle. *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 1–5. Available from `http://ieeexplore.ieee.org/document/5520914/`

Scaramuzza, D. (2012). A Tutorial on visual Odometry.

Scherer, J., Yahyanejad, S., Hayat, S., Yanmaz, E., Vukadinovic, V., Andre, T., et al. (2015). An autonomous multi-UAV system for search and rescue. *DroNet 2015 - Proceedings of the 2015 Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, 33–38. Available from `http://www.scopus.com/inward/record.url?eid=2-s2.0-84959503077&partnerID=40&md5=b6b0b`

Sensortec, B. (2015). Digital pressure sensor BMP280. Available from `http://www.bosch-sensortec.com/en/bst/products/all_products/bmp180`

Shen, S., Michael, N., & Kumar, V. (2011). Autonomous multi-floor indoor navigation with a computationally constrained MAV. *Proceedings - IEEE International Conference on Robotics and Automation*, 20–25.

Shen, S., Mulgaonkar, Y., Michael, N., & Kumar, V. (n.d.). Vision-Based State Estimation and Trajectory Control Towards High-Speed Flight with a.

Sieberling, S., Chu, Q. P., & Mulder, J. A. (2010). Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction. *Journal of Guidance, Control, and Dynamics*, *33*(6), 1732–1742. Available from `http://arc.aiaa.org/doi/10.2514/1.49978`

SM Lavalle. (1999). Rapidly-Exploring Random Trees: A New Tool for Path Planning. *Algorithmic & Computational Robotics New Direc.*. Available from `http://xueshu.baidu.com/s?wd=Rapidly-exploring+random+trees%3A+A+new+tool+for+path+pl`

Society, T. R., Society, R., & Sciences, B. (n.d.). Downloaded from http://rspb.royalsocietypublishing.org/ on January 3 , 2017.

Tedrake, R., Manchester, I. R., Tobenkin, M., & Roberts, J. W. (2010). LQR-trees: Feedback Motion Planning via Sums-of-Squares Verification. *The In-*

ternational Journal of Robotics Research, *29*(8), 1038–1052. Available from http://journals.sagepub.com/doi/10.1177/0278364910369189

Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., et al. (2012). Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue. *IEEE Robotics and Automation Magazine*, *19*(3), 46–56.

Trajkovii, M., Hedley, M., Trajkovic, M., & Hedley, M. (1998). Fast corner detection. *Image and Vision Computing*, *16*(2), 75–87. Available from http://dx.doi.org/10.1016/s0262-8856(97)00056-5

Wan, E. a., & Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. *Technology*, *v*, 153–158. Available from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=882463

Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., & Siegwart, R. (2012a). Real-time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments. , *231855*, 957–964.

Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., & Siegwart, R. (2012b). Real-time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments X (t) X (t). ilT l v (t) Jx (lil (t) Jxfv X (t). *(Icra), 2012 Ieee* {...}, *231855*, 957–964. Available from http://ieeexplore.ieee.org/xpls/abs%7B_%7Dall.jsp?arnumber=6225147

Welch, G., & Bishop, G. (2006). An Introduction to the Kalman Filter. *In Practice*, *7*(1), 1–16. Available from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.79.6578&rep=rep1&type=pdf

Xie, Z., Xia, Y., & Fu, M. (2011). Robust trajectory-tracking method for {U}{A}{V} using nonlinear dynamic inversion. *5th International Conference on Cybernetics and Intelligent Systems (CIS)*, 93–98.

Xu, A., & Namit, G. (2008). SURF : Speeded Up Robust Features. *European Conference on Computer Vision*, 1–30. Available from papers2://publication/uuid/32095B14-5C8B-49A1-B157-ED2737F06BE1

Zul Azfar, A., & Hazry, D. (2011). A simple approach on implementing IMU sensor fusion in PID controller for stabilizing quadrotor flight control. *Proceedings - 2011 IEEE 7th International Colloquium on Signal Processing and Its Applications, CSPA 2011*, 28–32.