TECHNISCHE UNIVERSITÄT DRESDEN

IFA LS-AT

Fakultät Elektrotechnik und Informationstechnik Institut für Automatisierungstechnik

Bachelor Thesis

zum Thema

Image Based Visual Servoing for Aerial Robot

vorgelegt von Pablo Rodríguez Robles
im Studiengang Aerospace Engineering, Jg. 2014
geboren am 28.02.1996 in León, Spain

Betreuer: Dipl.-Ing. Chao Yao
Verantwortlicher Hochschullehrer: Prof. Dr. techn. Klaus Janschek
Tag der Einreichung: 27.03.2018

Aufgabenstellung

Test der PDF-Integration

Achtung

Auch wenn die Möglichkeit besteht, die eingescannte Aufgabenstellung als PDF zu integrieren, muss in **einem einzureichendem Exemplar** die Aufgabenstellung **im Original** eingebunden werden.

TECHNISCHE UNIVERSITÄT DRESDEN

IFA LS-AT

Fakultät Elektrotechnik und Informationstechnik Institut für Automatisierungstechnik

Image Based Visual Servoing for Aerial Robot

Hier muss der Text für die deutsche Kurzfassung inklusive eines aussagekräftigen Bildes eingefügt werden.

Betreuer:	Dipl.-Ing. Chao Yao
Hochschullehrer:	Prof. Dr. techn. Klaus Janschek
Tag der Einreichung:	27.03.2018

Bearbeiter: Pablo Rodríguez Robles

TECHNISCHE UNIVERSITÄT DRESDEN

IFA LS-AT

Fakultät Elektrotechnik und Informationstechnik Institut für Automatisierungstechnik

Image Based Visual Servoing for Aerial Robot

Here an English abstract including one significant image must be inserted.

Tutor:	Dipl.-Ing. Chao Yao
Supervisor:	Prof. Dr. techn. Klaus Janschek
Day of Submission:	27.03.2018

STUDENT RESEARCH THESIS Author: Pablo Rodríguez Robles

Contents

List of Figures

List of Tables

List of Listings

1 Introduction

1.1 Motivation and Background

During the last decade, the use of Unmanned Aerial Vehicles (UAVs) has spread among very different applications. The use of flying robots can be very helpful to improve the way some tasks are already achieved by terrestrial robots. For example, object transportation, environment mapping or surveillance. At the Institute of Automation Engineering[1] of the Technical University of Dresden, a drone is being developed in cooperation with the Institute of Solid Mechanics[2] to investigate the use of flying robots in aerial manipulation.

When dealing with manipulation of objects, it is desired that the aerial robot adopts a certain pose with respect to the target before the manipulation process really starts. The present work deals with the development of a Visual Servoing (VS) control system that helps a quadrotor robot to acquire the desired pose by means of image data.

A monocular monochrome camera as well as an Inertial Measurement Unit (IMU) are planed to be the only available on board sensors. For the controller proposed the feedback is directly computed from image features rather than estimating the robot's pose and using the pose errors as control input.

In order to integrate the visual servoing algorithm into the future modular robot system, the algorithm has been designed and tested on a underactuated conventional quadrotor. The aerial robot is implemented within the ROS[3] framework, where the visual servoing controller developed for this thesis is also integrated. Instead of using real hardware the complete system is simulated using Gazebo[4].

[1] Technische Universität Dresden. Institut für Automatisierungstechnik. 01062 Dresden, Germany

[2] Technische Universität Dresden. Institut für Festkörpermechanik. 01062 Dresden, Germany

[3] www.ros.org

[4] www.gazebosim.org

1.2 Aims and Objectives

The aim of this work is to implement and test a VS control algorithm for a quadrotor, which could be later used by the Flypulator (TODO: Add reference) project. This includes the review of the state of the art with regard to Visual Servoing, the design of a solution and a prototypical implementation with in the ROS framework and simulation with Gazebo of a test case.

The present thesis documents comprehensively the theoretical background, implementation details and results of the conducted work through the following structure. In Chapter 2 the theoretical background and state of the art of Visual Servoing is presented. Chapter 3 gives a description of the system requirements as well as the system decomposition by Structure Analysis (SA) **SA_Braune** Chapter 4 describes the solution developed and the algorithms to be tested. Chapter 5 deals with the implementation, testing and validation. Finally, Chapter 6 contains the final results and conclusions and Chapter 7 suggests future improvement and research paths.

2 Theoretical Background and State of the Art

2.1 Visual Servoing Theoretical Basics

In this section the theoretical basic background of visual servo controllers is briefly discussed. It is usual in the literature to take **chaumette_visual_2006** and **chaumette_visual_2007** as the main reference when it comes to the theoretical setup of the discipline. As a result, the following description is completely based on these popular sources[1].

Visual Servoing is defined in the literature as the use of computer vision data to control the motion of a robot. The image data comes from a camera, which can observe the robot fixed in the space or moving with the robot. The latter approach is know as eye-in-hand Visual Servoing and is the selected one for the case of this work.

Visual servo controllers accomplish their task of reaching a certain pose by trying to minimize the following error $\boldsymbol{e}(t)$

$$\boldsymbol{e}(t) = \boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a}) - \boldsymbol{s}^* \tag{2.1}$$

Here, $\boldsymbol{m}(t)$ is a set of image measurements (e.g. the image coordinates of the interest points or the image centroid of an object), that is, information computed from the image data. With the help of these measurements a vector of k visual features, $\boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a})$ is obtained, in which $\boldsymbol{a}$ is a vector containing different camera parameters. In contrast, $\boldsymbol{s}^*$ defines a set of desired features.

For the present case, where the target is not moving, $\boldsymbol{s}^*$ and the changes in $\boldsymbol{s}$ depend only on the camera motion.

There exist two main variants of Visual Servoing depending on how the features vector $\boldsymbol{s}$ is defined. On the one hand, Image Based Visual Servoing (IBVS) takes as $\boldsymbol{s}$ a set of features already available within the image data (TODO it can be seen as a control of the features in the image plan such

[1] The interested reader should visit the Lagadic research group home page (http://www.irisa.fr/lagadic), pioneers in the area.

that moving the features to a goal configuration implicitly results in the task being accomplished **espiau_1992**). On the other hand, Position Based Visual Servoing (PBVS) considers for $\boldsymbol{s}$ a set of 3D parameters that must be estimated from the image data (TODO: Add that it is a cartesian motion planing problem).

Using the PBVS approach leads to the necessity of camera calibration and estimation of the flying robot pose (TODO: Add reference or a bit more of information), these are two big disadvantages for the application intended in this work. On the other side, IBVS needs no camera calibration and allows the robot to achieve the pose desired without any pose estimation process.

A simple velocity controller can be arranged in the following way. Let $\boldsymbol{v}_c = (v_c, \boldsymbol{\omega}_c)$ be the spatial velocity of the camera, with v_c the instantaneous linear velocity of the origin of the camera frame and $\boldsymbol{\omega}_c$ the instantaneous angular velocity of the camera frame, as a result we can express the temporal variation of the features vector as

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_{\boldsymbol{s}} \boldsymbol{v}_c \tag{2.2}$$

Where $\boldsymbol{L}_{\boldsymbol{s}} \in \mathbb{R}^{k \times 6}$, the feature Jacobian, acts as iteration matrix relating the camera velocity and the change in the visual features.

The time variation of the error to be minimized can be obtained by combining 2.1 and 2.2

$$\dot{\boldsymbol{e}} = \boldsymbol{L}_{\boldsymbol{e}} \boldsymbol{v}_c \tag{2.3}$$

with $\boldsymbol{L}_{\boldsymbol{e}} = \boldsymbol{L}_{\boldsymbol{s}}$. The input for such a controller is the camera velocity $\boldsymbol{v}_c$, which, using 2.3, we can set in such a way that an exponential decrease of the error is imposed (i.e. $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$)

$$\boldsymbol{v}_c = -\lambda \boldsymbol{L}_{\boldsymbol{e}}^{+} \boldsymbol{e} \tag{2.4}$$

Here, $\boldsymbol{L}_{\boldsymbol{e}}^{+} \in \mathbb{R}^{k \times 6}$ is the Moore-Penrose pseudoinverse of $\boldsymbol{L}_{\boldsymbol{e}}$. It is computed as $\boldsymbol{L}_{\boldsymbol{e}}^{+} = (\boldsymbol{L}_{\boldsymbol{e}}^T \boldsymbol{L}_{\boldsymbol{e}})^{-1} \boldsymbol{L}_{\boldsymbol{e}}^T$, provided that $\boldsymbol{L}_{\boldsymbol{e}}$ is of full rank 6. Imposing this condition leads to $\|\dot{\boldsymbol{e}} - \lambda \boldsymbol{L}_{\boldsymbol{e}}^T \boldsymbol{L}_{\boldsymbol{e}} \boldsymbol{e}\|$ and $\|\boldsymbol{v}_c\|$ being minimal. Note that for the special case of $k = 6$, if $\boldsymbol{L}_{\boldsymbol{e}}$ is nonsingular, it is possible to obtain a simpler expression using the matrix inversion $\boldsymbol{v}_c = -\lambda \boldsymbol{L}_{\boldsymbol{e}}^{-1} \boldsymbol{e}$.

When implementing real systems it is not possible to know perfectly either $\boldsymbol{L_e}$ or $\boldsymbol{L_e^+}$. Thus, an approximation of these two matrices is introduced, noted with the symbol $\widehat{\boldsymbol{L_e}}$ for the approximation of the error interaction matrix and $\widehat{\boldsymbol{L_e^+}}$ for the approximation of the pseudoinverse of the interaction matrix. Inserting this notation in the control law we obtain

$$\boldsymbol{v}_c = -\lambda \widehat{\boldsymbol{L_e^+}} \boldsymbol{e} \tag{2.5}$$

Once the basic appearance of a visual servo controller has being presented, the goal is to ask the following questions: How should $\boldsymbol{s}$ be chosen? What is the form of $\boldsymbol{L_s}$? How should we estimate $\widehat{\boldsymbol{L_e^+}}$?

In the simplest approach, the vector $\boldsymbol{s}$ is selected as a set of image-plane points, where $\boldsymbol{m}$ are the set of coordinates of these image points and $\boldsymbol{a}$ the camera intrinsic parameters. Later in this work, a more complex definition for the image features vector $\boldsymbol{s}$ will be chosen.

The Interaction Matrix

The camera image capture is a procedure which projects a 3D point from its coordinates in the camera frame, $\boldsymbol{X} = (X, Y, Z)$, to a 2D image point with coordinates $\boldsymbol{x} = (x, y)$. From this geometry we have

$$\begin{cases} x = X/Z = (u - c_u)/f\alpha \\ y = Y/Z = (v - c_v)/f \end{cases} \tag{2.6}$$

where $\boldsymbol{m} = (u, v)$ gives the coordinates of the image point in pixel units, and $\boldsymbol{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters: c_u and c_v are the coordinates of the principal point, f is the focal length, and α is the ratio of the pixel dimensions. In this case, we take as feature the image point, thus $\boldsymbol{s} = \boldsymbol{x} = (x, y)$.

Taking the time derivative of the projection equations 2.6, we obtain

$$\begin{cases} \dot{x} = \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z \\ \dot{y} = \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{X} - y\dot{Z})/Z \end{cases} \tag{2.7}$$

The velocity of the 3D point can be related to the spatial velocity of the camera using the equation for the velocity in a non-inertial reference frame

$$\dot{\boldsymbol{X}} = -\boldsymbol{v}_c - \omega_c \times \boldsymbol{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases} \tag{2.8}$$

Introducing 2.8 in 2.7, and grouping terms we can write

$$\begin{cases} \dot{x} = -v_x/Z + xv_z/Z + xy\omega_z - (1+x^2)\omega_y + y\omega_z \\ \dot{y} = -v_y/Z + yv_z/Z + xy\omega_z - (1+y^2)\omega_x + x\omega_z \end{cases} \tag{2.9}$$

using matrix notation

$$\dot{\boldsymbol{x}} = \boldsymbol{L_x}\boldsymbol{v}_c \tag{2.10}$$

where the interaction matrix that relates the camera velocity $\boldsymbol{v}_c$ to the velocity of the image point $\dot{\boldsymbol{x}}$ is

$$\boldsymbol{L_x} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix} \tag{2.11}$$

In Equation 2.11, the value Z corresponds to the depth of the point relative to the camera frame. As a result, any Visual Servoing scheme using this form of the interaction matrix must provide an estimation of this value. Furthermore, the camera intrinsic parameters are necessary to compute x and y. Therefore, it is not possible to use directly $\boldsymbol{L_x}$, but an approximation $\widehat{\boldsymbol{L_x}}$ is to be used.

Approximation of the Interaction Matrix

When the current depth Z of each point is known, there is no need of approximation and $\widehat{\boldsymbol{L_e^+}} = \boldsymbol{L_e^+}$ for $\boldsymbol{L_e} = \boldsymbol{L_x}$ can be used. However, this approach requires the estimation of Z for all iterations of the scheme control (see **hutchinson_1996**), which may be conducted by means of pose estimation

methods.

A second alternative is to use $\widehat{\boldsymbol{L}_{\boldsymbol{e}}^{+}} = \boldsymbol{L}_{\boldsymbol{e}^*}^{+}$, where $\boldsymbol{L}_{\boldsymbol{e}^*}$ is the value of $\boldsymbol{L}_{\boldsymbol{e}}$ for the desired position ($\boldsymbol{e} = \boldsymbol{e}^* = 0$) (see **espiau_1992**). Here, the depth parameter only needs to be estimated once for every point.

2.2 State of the Art

2.2.1 Visual Servoing for Aerial Robots

In this section the literature on the use of Visual Servoing to control de translation of aerial robots is analyzed. The focus of this review is on the IBVS approach, where image feature errors between the current and target image are mapped to actuator inputs through the inverse of the Jacobian matrix. The different approaches discus the alternatives on feature selection, Jacobian matrix construction and different control strategies to make the error converge to zero, and with it, the relative pose between UAV and target object will converge to the desired one.

One of the problems of the classical formulation of IBVS is the necessity to estimate the depth of each of the features. In the past several approaches have been taken to cope with this problem, such as: partial pose estimation (see **malis_2_1999**), adaptive control (see **Papanikolopoulos**) or estimation of the Jacobian using quasi-Newton methods (see **Piepmeier**). An alternative is to use a hybrid control approach by treating rotational and translational control separately. However, it is important to bear in mind that due to the under-actuated dynamics of an usual 4 DOF quadrotor, it is only possible to control the three linear velocities and the yaw (rotational) speed. While the other two rotational speeds, roll and pith, are used to move the thrust vectors of the propellers and thus generate movement on the horizontal plane.

Most of the Visual Servoing methods have been developed for serial manipulators. For this kind of robot, a low-level actuator is usually employed to compensate the dynamic behavior of the system, making possible to control it with velocity commands as a first order system. For a full-dynamics quadrotor model accounting for aggressive maneuvers this is not at all the case, since it is a fourth order system. Thus, other strategies are usually considered to control such a robot (see**Mellinger**).

2.2.2 Visual Servoing for Aerial Manipulators

In this section a perspective of Visual Servoing applied to aerial manipulators is presented. The main literature is analyzed and some of the common characteristics of the approaches followed by them are highlighted.

Some publications related to the University of Pennsylvania GRASP Laboratory[2] (see **thomas_toward_2014** and **thomas_visual_2016**) have studied the vision-based localization and servoing of quadrotors in grasping and perching tasks. However, the emphasis of these publications lays on the generation of dynamically-feasible trajectories in the image space, thus second order system control is performed instead of the most common kinematic control strategy. These vehicles are not manipulators in the sense of the rest of the approaches presented in this section, since the main task here is hanging from structures and grasping targets by means of aggressive, thus dynamical, maneuvers. In addition to that, the actuator used is not a high-DOF actuator, but a 1 DOF gripper.

In the last years, a concrete aerial manipulator architecture has been popularized, for example in the context of the European projects ARCAS[3] and AEROARMS[4]. These aerial manipulators have usually the task of collecting an structural element from its initial position and fly it to a final position, where the element is used to assemble a strut structure. The main configuration of such a robot is an under-actuated rotary-wing aircraft (usually a quadrotor) and a robotic manipulator arm. Different degrees of freedom (DOF) are used for the arm and different camera placements are considered.

The usual implementation considers the simultaneous control (at the velocity level) of the mobile platform (i.e. quadrotor) and the manipulator for such a grasping task. Since the sum of the 4 DOF of a quadrotor plus the multiple-DOF of a serial manipulator leads to a redundant system, the possibility of choosing degrees of freedom is used to realize different subtasks (e.g. joint reaching prevention). The Visual Servoing controller chosen generates velocity inputs both for the manipulator joints (i.e. $\dot{\boldsymbol{q}}$) and for the quadrotor (i.e. translational velocity $\boldsymbol{v}$ and rotational velocity ω_z). The use of a weighted pseudo-inverse allows to favor the control of the mobile platform when the distances to the target are bigger and increase the manipulator use when it is

[2] https://www.grasp.upenn.edu
[3] http://www.arcas-project.eu
[4] http://www.arcas-project.eu

close to the target.

mebarki_image-based_2014 propose a quadrotor equipped with a 5 DOF arm. Traditional Visual Servoing distinguishes between two different classes of camera configuration: eye-to-hand (fixed in the workspace) and eye-on-hand (mounted on the mobile platform). In this paper a new configuration for the camera is presented, called onboard-eye-to-hand, i.e. the camera is placed on-board of the robot while it observes the manipulator. In this way, the manipulator can accomplish large rotations while the target is not left out of the camera field of view, as happens in the eye-in-hand configuration. Furthermore, for the case of eye-in-hand configuration, during assembly tasks the manipulator end-effector can contact or impact with objects and damage or obstruct the camera. Thanks to the onboard-eye-to-hand camera configuration the paper is able to introduce a variation of the IBVS approach, called Self Visual Servoing (SVS). Where the error nullified comes directly form the image itself (hence the adjective self) an there is no need for a target image. The servo controller implemented has two different tasks. The main task is position the feature points at a target position on the target object and the second one the end-effector motion. Error formulation decouples both tasks and a wighted pseudo-inverse is used to provide a different gain for the arm joints rates $\dot{\boldsymbol{q}}$ and for the UAV velocities $\boldsymbol{v}$ and ω_z.

The image moments as features for the Visual Servoing are proposed in **mebarki_exploiting_2013** for the previous system. Furthermore, aerial manipulators have to cope with the change of the center of mass during flight due to the effect of suspended loads **palunko_2012** To achieve this behavior, low-level attitude controllers are usually designed to compensate this effects using Cartesian impedance control **lippiello_impedance_2012** or adaptive control. To this end, the system includes a controller to reduce dynamic effects by vertically aligning the arm center of gravity to the multirotor gravitational vector, along with one that keeps the arm close to a desired configuration of high manipulability and avoiding arm joint limits. In **mebarki_cross-coupled_2014** the author completes the robot with a nonlinear low-level controller that thanks to a integral approach allows the inclusion of the dynamic coupling of the UAV and the robotic arm, while the control of the system through the velocities provided by the IBVS high-level is maintained.

In this case the source (see **danko_evaluation_2014**) includes as host a gantry used to emulate an UAV and a 6 DOF manipulator with an end-effector mounted camera (i.e. eye-in-hand). Coordination of redundant degrees of

freedom by means of partitioned control. Visual servoing is used to drive the end-effector pose relative to a target thanks to the use of feature points and their desired positions.

lippiello_hybrid_2016 uses a hybrid-control framework the main benefits of both IBVS and PBVS schemes to control a octorotor with a 6 DOF arm. Kinematic redundancy of the end-effector is used to accomplish secondary tasks and lead by a hierarchical task-composition algorithm, in conjunction with a smooth activation mechanism for the tasks.

DLR's work within ARCAS project **laiacker_high_2016** uses of an helicopter and a 7 DOF manipulator. The helicopter is a bigger robot when compared to the rest of the systems, with more than 1 m from manipulator to center of gravity. Influence of the arm movement is significant to the helicopter flight and is actively compensated by the robot controller by means of a coordinated control of both elements. The paper discusses performance and accuracy in aerial manipulation, where the time it takes between the measurement of a position difference and its compensation using the manipulator or the flying platform is the main factor. Additionally, the work presents a multi-marker approach to compensate occlusion of the target marker by the manipulator derived of the onboard-eye-to-hand camera configuration.

A combination of kinematic and dynamic models to develop a passivity-based adaptive controller which can be applied on both position and velocity control is proposed in **kim_vision-guided_2016** Position control is used for waypoint tracking and landing, while velocity control is triggered for target servoing. The robot is a quadrotor with a 3 DOF arm and an eye-in-hand camera. The work uses IBVS with image moments as visual features and tries to solve two problems of the method when applied to aerial manipulators (under-actuated system). Firstly, movements of the manipulator produce movement of the camera, thus making probable that the target object is taken out of its field of view. For this reason a fisheye camera is used. Secondly, the under-actuation of the robot is corrected introducing a image modification method. Velocity weighting of the Jacobian matrix in accordance of the situation is used for the simultaneous control of UAV and manipulator.

In **santamaria-navarro_uncalibrated_2017** redundant manipulation and hierarchical control law are combined with a new variation of the IBVS that does not need the camera parameters. The system establishes as primary task the avoidance of obstacles, as well as several secondary tasks. The visual servo strategy is used to drive the arm end-effector to a desired position and

Table 2.1: My caption

Reference	Vehicle	Manipulator's DOF	Cam
thomas_toward_2014 and **thomas_visual_2016**	quadrotor	1	eye-
mebarki_image-based_2014	quadrotor	5	onbo
mebarki_exploiting_2013	quadrotor	5	onbo
mebarki_cross-coupled_2014	quadrotor	5	onbo
danko_evaluation_2014	grantry	6	eye-
lippiello_hybrid_2016	octortor	6	onbo
laiacker_high_2016	helicopter	7	onbo
kim_vision-guided_2016	quadrotor	3	eye-
santamaria-navarro_uncalibrated_2017	quadrotor	6	eye-

orientation by using a camera attached to it. The configuration used is a 4 DOF quadrotor and equipped with a 6 DOF robotic arm. In the common IBVS approaches, Jacobian or interaction matrix, which relates the camera velocity with the image feature velocities, depends on a priori knowledge of the intrinsic camera parameters. The paper presents a variation of IBVS called Uncalibrated IBVS, the approach uses the barycenter of the features as control points. The method recovers the coordinates of these control points and also the camera focal length, with this data a new formulation of the Jacobian is constructed. The system also compensates by means of a hierarchical algorithm the position of the manipulator.

3 Software Requirements Specification and Structured Analysis

This chapter deals with the Software Requirement Specification (SRS) **IEEE8301998** and the Structured Analysis **SA_Braune** of the system developed in this work. Thanks to these two procedures, the objectives that the system must fulfil and a decomposition of it into different functions are stated. This leads to a complete definition of the system.

The purpose of this work is to design a Visual Servoing controller to provide an underactuated aerial robot the commands necessary to reach a desired pose with respect to a target object.

The VS controller developed is to be integrated into the hector_quadrotor **2012simpar_meyer** an underactuated aerial robot equipped with a monocular monochrome camera pointing downwards.

3.1 Software Requiremet Specification

In this section the Software Requirement Specification **IEEE8301998** for the Visual Servoing controller developed in this thesis is presented. Using SRS helps to define the system that is being designed, tracking continuously that the product developed satisfies the needs of the user. Only when every requirement stated therein is fulfilled the implementation would be completed.

3.1.1 Product Perspective

The VS controller is to be used with an aerial robotic system based on the ROS framework. From the perspective of the robotic system, the VS controller subsystem will appear as a ROS node which publishes control commands through a ROS topic to the rest of the system.

The used aerial robotic system is the hector_quadrotor[1] model **2012simpar_meyer** (TODO: Add diagram).

[1] wiki.ros.org/hector_quadrotor

The subsystem developed here is to interact with the camera hardware of the robot, a monocular monochrome camera pointing downwards (TODO: Add hardware). The output of the subsystem are the control inputs of the aerial robot dynamics (TODO: Add which are the quadrotor control inputs), this inputs interact with the inner control loop for the attitude and outer control loop for the position already implemented in the robotic system (TODO: Position loop is not related to vs system, but can be useful for benchmark).

3.1.2 User Characteristics

The product developed in this thesis will be used as part of a ROS-based system, thus the expected user is a designer willing to implement a VS control strategy for his robotic system. The user should be familiarized with the ROS framework and the system will need the structure and interfaces of any standard ROS product.

3.1.3 Assumptions and Dependencies

The software has been tested on the following platforms, forward or backward support is not guaranteed on a different set-up.

- ROS version: ROS Indigo[2]
- Operating System: Ubuntu 14.04[3] Trusty Tahr, 64 bit

3.1.4 Functional Requirements

The functional requirements describe what the system must do to complete the overall task:

- F1: *Give visual servoing control input.* Control input based on image data so that the aerial robot comes closer to the target pose. Control as a difference on the image features, no pose estimation.
- F2: *Tell user when the target pose is achieved.* The system must be able of telling the user whether the target pose has been already achieved or not.

[2] http://wiki.ros.org/indigo
[3] http://releases.ubuntu.com/14.04/

3.1.5 Other Requirements

- A1: All components are working reliably.
- A2: The software is sufficiently fast, modular and modifiable.
- A3: The implementation is transparent and comprehensible.
- A4: Control inputs must provide stable and smooth flight manoeuvres.
- A5: Robot must be able to start from different initial positions.
- A6: Algorithm must be fast enough to allow real time control of the aerial robot.
- A7: The implementation should follow the style guide of ROS**ROS_Style**

3.1.6 General Constraints

- The environment must be sufficiently illuminated for the camera to work.
- The target pose must be provided by a sufficient number of features (TODO: How many?) in form of a 2D code (TODO: At least in the first version).
- The target must be always in the filed of view of the camera, so features can be extracted an control input computed.
- Testing computer is a MacBook Pro[4] (Early 2015) with 2.7 GHz Intel Core i5 processor, 8 GB 1867 MHz DDR3 memory and Intel Iris Graphics 6100 1536 MB graphics. Linux OS is run using Oracle VM VirtualBox[5] (Version 5.1.14 r112924) with 5 GB base memory and two processors.

3.2 Structured Analysis

[4] https://everymac.com/systems/apple/macbook_pro/specs/macbook-pro-core-i5-2.7-13-early-2015-retina-display-specs.html
[5] www.virtualbox.org

4 Visual Servoing Algorithm Description

5 Implementation of the Visual Servoing Controller

6 Final Results and Conclusions

7 Future Work

Selbstständigkeitserklärung

Hiermit versichere ich, Pablo Rodríguez Robles, geboren am 28.02.1996 in León, Spain, dass ich die vorliegende Bachelor Thesis zum Thema

Image Based Visual Servoing for Aerial Robot

ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Bei der Auswahl und Auswertung des Materials sowie bei der Herstellung des Manuskripts habe ich Unterstützungsleistungen von folgenden Personen erhalten:

Dipl.-Ing. Chao Yao

Weitere Personen waren an der geistigen Herstellung der vorliegenden Bachelor Thesis nicht beteiligt. Mir ist bekannt, dass die Nichteinhaltung dieser Erklärung zum nachträglichen Entzug des Diplomabschlusses (Masterabschlusses) führen kann.

Dresden, den 27.03.2018 .....

Unterschrift