Control System Design for a General Blimp

Undergraduate Honor Research Thesis

In partial fulfillment of the requirements for graduation with distinction in the Department of Mechanical and Aerospace Engineering of The Ohio State University

By
Zhiyi Jiang

The Ohio State University

April 2019

Project Advisor: Dr. Ran Dai

Acknowledgments

I would like to express my great appreciation to the following people for their help in this research.

Dr. Ran Dai, my research advisor, for providing me research and study opportunity in Automation and Optimization Laboratory (AOL). I would like to thank for her patient guidance, enthusiastic encouragement, and constructive suggestions during this research.

Myungjin Jung for helping me to learn robotics and to give me advice on the flight test. He agreed with me to study in his Solar-powered Blimp Project when I was new to AOL.

Sean You for helping me in the design of controller and providing study resources about Model Predictive Control.

Other people in the AOL for giving me advice during the weekly meeting.

Abstract

A blimp is an airship without internal structure and has been used to explore unknown areas and advertisements. The blimp can be used to do a long-time flight with less energy consumption. What's more, the blimp can work with other robots to explore unknown areas. Automation and Optimization Lab has built a blimp with the utilization of Proportional–Integral–Derivative (PID) controller. This blimp has experienced an indoor manual control test and an outdoor trajectory tracking test. During the tests, the PID controller helped the blimp finish all tasks during the indoor flight test. However, in the outdoor flight test, the blimp was unstable because the PID controller could not control the altitude and y-axis position of the blimp while the blimp tried to go forward. In addition, wind disturbance can easily influence the blimp's motion. What's more, the energy cost of the blimp could not be managed during the test. To solve the instability and energy cost problem when facing unknown disturbance, new control algorithms will be implemented for the system. Sliding Mode Control (SMC) and Model Predictive Control (MPC) are two candidates for controlling the blimp because SMC has a strong disturbance rejection ability and MPC can add constraints to energy cost. By testing two controllers' ability in controlling simulated blimp model, the performances of two controllers are summarized. For the further test, new controllers will be tested in indoor trajectory tracking test.

Contents

List of Figures

Chapter 1: Introduction

1.1 Blimp

A blimp is a kind of non-rigid airship whose development is limited by the influence of Hindenburg disaster and the development of jet planes and helicopters [1]. Although people are using helium instead of hydrogen to fill the blimp today, the flight endurance and safety of today's blimp cannot achieve a very high level which can lead people to ignore the blimp's disadvantages. For example, Goodyear blimp *Wingfoot* which is a relatively high-technology blimp in today's world can only have a flight endurance up to 40 hours with the maximum speed up to 73 miles per hour and maximum load up to 1,9780 pounds [2]. Compared to some planes such as Airbus A380, this blimp cannot be used in transportation because of its low speed and small capacity. Therefore, this kind of blimp can only be used for advertising and getting the view for live events. To explore the utilization of the blimp, people are trying to maximize the features of blimps

In the recent year, unmanned aerial vehicles (UAV) are becoming popular because of the advancement of electronics such as cameras, the inertial measurement unit (IMU), and flight controllers. By combing the traditional blimp and UAV technology, the blimp is developed to work without human working on it. In this situation, the blimp can work with other robots to explore the unknown area. For the exploration of a new environment, generating maps is one of the fundamental tasks of exploration. To do this task, the blimp can provide a wide view of the unknown area and other robots can explore the blind spots within the view. Therefore, this combination will be an advancement in exploration. To test the blimp performance on exploring an unknown area, a prototype of the blimp has already been built by Automation and Optimization Laboratory.

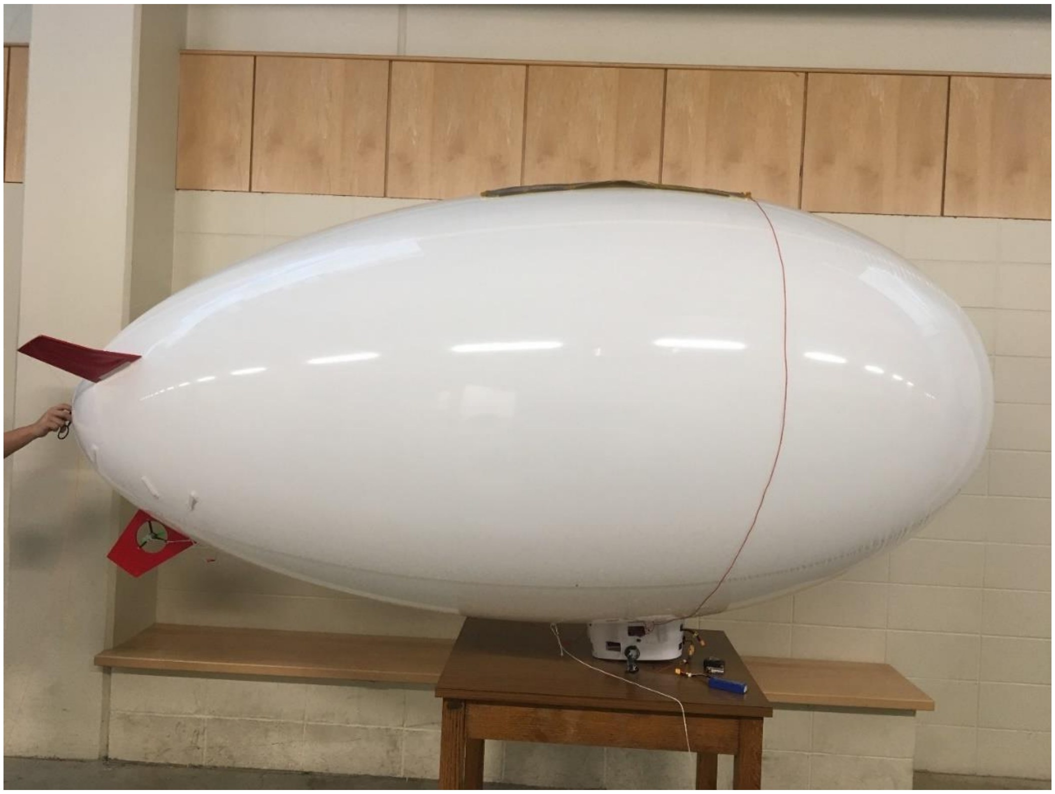

Figure 1: Blimp prototype

The Ph.D. student in the lab has done two flight tests. For the first flight test, the blimp was tested inside the building by manually sending commands to the blimp. The commands decided to go which direction and utilize how much energy. This test was mainly about checking the performance of actuator and feedback sensors such as motors and IMU. For the second flight test, the place was set to be outdoors. During this test, the blimp was asked to follow a specific trajectory from one point to another point. This test provided the performance of the blimp in sending feedback information to the laptop and autonomous flight. According to these two tests, the actuators such as motors were proved to work well during the flight. However, during the outdoor flight test, the blimp could not maintain its altitude while going forward and the controller's performance in following the trajectory did not achieve the desired goal because the blimp was hard to maintain its y position and yaw angle under wind disturbance. To solve these two problems, the control system of the blimp is required to be changed.

1.2 Control system

A control system is usually composed of sensors, control algorithms, and actuators. A feedback controller was built for the blimp because it was designed to work autonomously in unknown areas. Firstly, sensors are tools which can send the current status of the plant to the controller and laptop. For the blimp, sensors will provide the specific position and Euler angles of the blimp. Secondly, control algorithms or control theory is the utilization of mathematics in the dynamic systems [3]. Control theories are divided into classical control theories such as Proportional Integral Derivative (PID) Control and modern control theories such as Sliding Mode Control (SMC). Classical control theory is limited to a single input and single output which leads to multiple controllers for multiple-input and multiple-output (MIMO) system. By contrast, modern control theory can deal with MIMO system because it is carried out in state space form. For the existing blimp, the PID controller was utilized in both indoor and outdoor flight test. Thirdly, the actuators of the blimp are motors at the bottom and on the side of the blimp, which can provide required forces in all three axes.

1.3 Current Controller of the blimp

For the current controller of the blimp, sensors utilized to get the feedback information of the blimp were different in different situations. For outdoor flight test, GPS, ultrasonic sensor, and

IMU were used to get the accurate position of the blimp. For indoor flight test, VICON camera system was accurate enough to capture the motion and status of the blimp. This blimp was designed to work autonomously and had been tested for an indoor manual control flight test and an outdoor autonomous trajectory tracking flight. For this blimp, the PID controller was utilized for the first generation of the blimp because there were some researches about using PID theory to control the blimp or airship to finish high altitude working mission. During the indoor flight test, the blimp was controlled by the command to go forward, backward, make turns, take off, and land. For the outdoor flight test, the blimp was asked to follow a designed curve autonomously. During the test, the blimp was easily influenced by the surrounding disturbance from the environment such as strong winds which could achieve 6 m/s and the blimp couldn't keep its altitude while going forward. For the first problem of outdoor flight test, the disturbance easily influenced the blimp's performance because the PID controller has low robustness which means a weak ability in disturbance rejection. For the second problem of outdoor flight test, the limitation of side motor power and the function of the PID controller caused the unstable flight status. In addition, the PID controller cannot manage energy cost or do energy optimization of the blimp. The only way for PID to control the energy cost was to set a max input which only allowed input less than the maximum value and skipped the input larger than the maximum value. These problems will lead to increasing uncertainty in exploring unknown areas. To advance the performance of the blimp during outdoor autonomous flight, different control methods will be applied and compared on both simulation and real flight test. In this thesis, except for the PID controller, Model Predictive Control (MPC) and Sliding Mode Control (SMC) will be discussed based on their performance in simulation and real flight tests. With the implementation of the new controller, the blimp can have a strong disturbance rejection ability while flying in the unknown area. In addition, the energy cost of the blimp will also be constrained to a certain value. Therefore, the blimp can achieve the desired flight time without losing too much energy.

For MPC control, this is a model-based control method and optimization is set as the theory behind this method [4]. The accuracy of the model will be an important factor in the control result. In addition, in this thesis, linear model predictive control method will be utilized because nonlinear MPC has a slow converge time which is even larger than the sample time. To use linear MPC, linearization of dynamic model and finding equilibrium point are required. By using a dynamic model to estimate the motion of the blimp, the error between the reference and estimated status is minimized through optimization. At the same time, the constraint of energy cost can also be added to the optimization process. For this problem, the constraint will be the push force of the blimp. According to the optimization, the input can be calculated and will be sent the blimp. This method has its own advantage in adding constraints to the control system. Therefore, the blimp does not need to sacrifice the energy cost to reject the disturbances. At the same time, MPC controller also has its own disadvantage because MPC is firstly designed to control linear system and its dependence on model accuracy even though the designer can add some disturbance prediction to the system. For the blimp whose dynamic model is nonlinear, the linearized model may not be accurate enough in predicting real blimp performance under the influence of disturbances.

For SMC control, this is a model-free control theory like PID control. SMC has been developed to control nonlinear system for a long time [5]. This method is known to have a strong disturbance rejection ability and fast convergence time. This method does not require dynamic model and model accuracy will not influence the control performance. Sliding mode control method needs to set a sliding surface so that the model can slide toward it. By using Lyapunov function to determine the stability of the system, the input of the blimp can be defined. For convergence time, SMC does not require a complicated calculation like MPC, which largely reduce the time cost of convergences. Therefore, it is a good choice for controlling the blimp. SMC also has its own disadvantage about its fuzzy output result and the chattering problem. For the blimp, the influence of the chattering resulted from SMC controlling is unknown. If the chattering influence won't lead an obvious vibration of the blimp, this concern can be ignored.

1.4 Thesis Objective

This research mainly focuses on the implementation and comparison of MPC and SMC controller. After comparison in simulation, two controllers will be compared in controlling a small blimp model inside the building so that their performance can be checked and the advanced control method can be used to control the blimp. This small blimp will finish the indoor target following test.

With the new controller, the blimp can maintain its regular stable status with the smallest energy cost and can quickly reject disturbance in an unknown situation. With these two features, the blimp can work better in co-operating with other robots because it can be regarded as a foundation for the user to send collected data, send command, and provide the vision of blind spot of other robots in exploring unknown areas.

1.5 Thesis Summary

In this thesis, there will be four chapters. The first chapter will introduce the background of the research topic and the objective of the research. The second chapter which is composed of three parts will talk about the methodology used in the design of the control system. The first part will introduce the dynamic model of the blimp. The second part will introduce the methods used in the design of model predictive controller, the design of sliding mode controller, and indoor flight test. In the third chapter, the content will focus on the discussion of the simulation result. For the last chapter, there will be a summary of the research project. Also, there will be some recommendations and ideas for future work because the result of the project is limited to the time cost.

Chapter 2 Methodology: Design of Control System

2.1 Dynamic Model

The blimp was compared with some existing blimp to find a proper dynamic model for it. After comparison, the blimp in the paper [6] has a similar frame and functionality to the blimp, which can provide a dynamic model frame for analysis. By deleting some useless parameters and variables in this dynamic model, the dynamic model for blimp is derived.

State variables:

The velocity in all three axes are defined as $\mathrm{UA} = [\mathrm{u\ v\ w}]$

The position in all three axes are defined as $\mathrm{Pos} = [\mathrm{px\ py\ pz}]$

The angle velocity in all three axes are defined as $\omega_{\mathrm{B_A}} = [\mathrm{p\ q\ r}]$

The Euler angles in all three axes are defined as $\text{Euler angle} = [\mathrm{u\ v\ w}]$

The wind velocity in all three axes are defined as $\mathrm{W} = [\mathrm{wx\ wy\ wz}]$

The wind acceleration in all three axes are defined as $\dot{W} = [\dot{w}x\ \dot{w}y\ \dot{w}z]$

The inputs of the system are assumed to be propulsive force $\mathrm{P_A} = [PAx\ PAy\ PAz]$

Rotation Matrix:

$$R_{B_AI} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -sin\theta \\ sin\theta cos\psi sin\phi - cos\phi sin\psi & sin\theta sin\psi sin\phi + cos\phi cos\psi & cos\theta sin\phi \\ sin\theta cos\psi cos\phi + sin\phi sin\psi & sin\theta sin\psi cos\phi - sin\phi cos\psi & cos\theta cos\phi \end{bmatrix} \tag{1}$$

To simplify the expression of model, $\mathrm{f}_1, c_1, f_2, c_2$ are defined:

$$f_1 = -[(M+m)I_{3\times3} - A_1]^{-1}[-A_2 + MS(\rho_{CM})] \tag{2}$$

$$c_1 = [(M+m)I_{3\times3} - A_1]^{-1}\{(M+m)[S(\omega_{B_A})U_A - R_{B_AI}\dot{W}] + R_{B_AI}F_G + A_0 + P_A - MS^2(\omega_{B_A})\rho_{CM}\} \tag{3}$$

$$f_2 = (I_t - M_1)^{-1}[M_2 + MS(\rho_{CM})] \tag{4}$$

$$c_2 = (I_t - M_1)^{-1}\{M_0 + M_G + M_p + S(\omega_{B_A})I_M(\omega_{B_A}) + MS(\rho_{CM})[-S(\omega_{B_A})U_A + R_{B_AI}\dot{W}]\} \tag{5}$$

In the equation above, F_G is the sum of the gravity force factor. I_t is the total inertia matrix and I_M is the inertia matrix of the rigid portion with respect to B_A frame. M_G *and* M_P are momentum caused by gravity and propulsion. $A_0, A_1,$ *and* A_2 are aerodynamic forces representation. $M_0, M_1,$ *and* M_2 are matrixes used to take virtual mass and inertia effect into the rotational dynamics.

Therefore, the dynamic model can be defined as:

$$\dot{UA} = (I_{3\times3} - f_1 f_2)^{-1}(f_1 c_2 + c_1) \tag{6}$$

2.2 Design of Model Predictive Control

2.2.2 Linearization for MPC

Model Predictive Control is designed to work with a linear system. To use it control the blimp, linearization is required. For the blimp, nonlinear model predictive control is not fast enough to control it because of its time cost during convergence. The dynamic model derived from the existing paper is a nonlinear system which cannot be controlled by linear model predictive control (LMPC). For the implementation of the LMPC, the dynamic model needs the form of:

$$\dot{x} = Ax + Bu$$
$$\mathrm{y} = \mathrm{Cx} + \mathrm{Du} \tag{7}$$

The linearization is required to get A, B, C, D matrixes. However, in the nonlinear model, state variables are defined to be 6 variables: [u v w p q r]. The LMPC is not going to control all of these variables because the design purpose of LMPC is to get control of the blimp velocity in x-axis while keeping its altitude and the position on y-axis stable. So, the control variables are [u py pz]. Although using x y z position for linearization will be easier in trajectory tracking, the linearized system with these three variables could not be used in MPC to control the blimp and more details will be shown in simulation result. In addition, before linearization of the model, equilibrium points are required because equilibrium points can prove the system staying at a steady state. The dynamic system can only be linearized at a steady status.

After finding the equilibrium points of After linearization, u is close to 0 m/s, py is close to 0 m, and pz is close to 1 m. By doing linearization around these values, the model can be calculated to be:

Matrix A (1,2)

$$\frac{\frac{M*w-M*p*rhocm2+2*M*q*rhocm1}{A_1-I_{3\times3}*M}+\frac{M*srhocm*w*(A_2-M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}M)}}{I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}} \tag{8}$$

Matrix A (1,3)

$$-\frac{\frac{M*v+M*p*rhocm3-2*M*r*rhocm1}{A_1-I_{3\times3}*M}+\frac{M*srhocm*v*A_2-M*srhocm}{(I_t-M_1)*(A_1-I_{3\times3}*M)}}{(I_{3\times3}+\frac{(A_2-M*rhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)})} \tag{9}$$

Matrix A (2,1)

$$\frac{\frac{M*r}{A_1-I_{3\times3}M}+\frac{M*r*srhocm*(A_2-M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}}{I_{3\times3}+\frac{(A_2-M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}M)}} \tag{10}$$

Matrix A (2,2)

$$-\frac{(M*p*rhocm1+M*r*rhocm3)}{(A_1-I_{3\times3}*M)*\left(I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{\left((I_t-M_1)*(A_1-I_{3\times3}M)\right)}\right)} \tag{11}$$

Matrix A (2,3)

$$\frac{\frac{M*u-M*p*rhocm3+2*M*r*rhocm2}{A_1-I_{3\times3}*M}+\frac{M*srhocm*u*A_2-M*srhocm}{(I_t-M_1)*(A_1-I_{3\times3}*M)}}{(I_{3\times3}+\frac{(A_2-M*rhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)})} \tag{12}$$

Matrix A (3,1)

$$\frac{\frac{-M*q}{A_1-I_{3\times3}M}+\frac{M*q*srhocm*(A_2-M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}}{I_{3\times3}+\frac{(A_2-M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}M)}} \tag{13}$$

Matrix A (3,2)

$$-\frac{\frac{M*u-2*M*p*rhocm3+M*r*rhocm2}{A_1-I_{3\times3}*M}+\frac{M*srhocm*u*(A_2-M*srhocm}{(I_t-M_1)*(A_1-I_{3\times3}M)}}{I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}} \quad (14)$$

Matrix A (3,3)

$$-\frac{(M*p*rhocm1+M*q*rhocm2)}{(A_1-I_{3\times3}*M)*\left(I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{\left((I_t-M_1)*(A_1-I_{3\times3}M)\right)}\right)} \quad (15)$$

$$Matrix\ A=\begin{bmatrix} 0 & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix} \quad (16)$$

Matrix B (1,1)

$$\frac{1}{(A_1-I_{3\times3}*M)*\left(I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}\right)} \quad (17)$$

Matrix B (2,2)

$$\frac{1}{(A_1-I_{3\times3}*M)*\left(I_{3\times3}+\frac{(A_2-M*srhocm)*(M_2+M*srhocm)}{(I_t-M_1)*(A_1-I_{3\times3}*M)}\right)} \quad (18)$$

$$Matrix\ B = \begin{bmatrix} B & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & B \end{bmatrix} \tag{19}$$

State variables:

$$x = \begin{bmatrix} u \\ pos_y \\ pos_z \end{bmatrix} \tag{20}$$

Inputs:

$$U = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} \tag{21}$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{22}$$

$$D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{23}$$

Continuous Motion of equation:

$$\dot{\Delta x} = A * \Delta x + B * \Delta U$$

$$\Delta y = C * \Delta x + D * \Delta U \tag{24}$$

2.2.2 Design Process

Model predictive control requires a discrete form of motion equation because prediction function is required to be derived from discrete form. According to the book written by Hong Chen[2], Discrete Motion of equation with the assumption of sample time T:

$$
\begin{aligned}
\mathrm{A_d} &= e^{AT} \\
\mathrm{B_d} &= A^{-1}(A_d - I)B = A^{-1}(e^{AT} - I)B \\
\mathrm{C_c} &= C \\
\mathrm{D_c} &= D
\end{aligned}
\tag{25}
$$

In addition, there are some noises for the input of the system. Therefore, the measured input variation was assumed to be:

$$\mathrm{y_m}(k) = C_m x(k) \tag{26}$$

The estimation equation was assumed to be:

$$
\begin{aligned}
\Delta \mathrm{x}(\mathrm{k}+1) &= \mathrm{A_d}\Delta x(k) + B_d \Delta U(k) + L(y_m(k) - C_m \Delta x(k)) \\
\Delta \mathrm{y_c}(k) &= C_c \Delta x(k)
\end{aligned}
\tag{27}
$$

If the matrix $\mathrm{A_d}$ and $\mathrm{C_m}$ are observable, pole placement method can be utilized to get the stable state of the system by tuning the value of parameter L.

Prediction Function:

$$p: prediction\ horizon;$$

$$m: control\ horizon$$

$$\mathrm{n_c}: number\ of\ control\ variable = 3$$

$$\Delta U = \begin{bmatrix} Forwar\ Force \\ Side\ Force \\ Upward\ Force \end{bmatrix}$$

$$\Delta x = \begin{bmatrix} \Delta x_{velocity} \\ \Delta y_{position} \\ \Delta z_{position} \end{bmatrix}$$

$$\mathrm{Y_p}(k+1|k) = S_x \Delta x(k) + I y_c(k) + S_u \Delta U(k) \tag{28}$$

$$\mathrm{S_x} = \begin{bmatrix} C_c A_d \\ \sum_{i=1}^{2} C_c A_d^i \\ \vdots \\ \sum_{i=1}^{50} C_c A_d^i \end{bmatrix}_{p\times 1} \quad I = \begin{bmatrix} I_{n_c \times n_c} \\ I_{n_c \times n_c} \\ \vdots \\ I_{n_c \times n_c} \end{bmatrix}_{p\times 1} \tag{29}$$

$$\mathrm{S_u} = \begin{bmatrix} C_c B_d & 0 & 0 & \dots & 0 \\ \sum_{i=1}^{2} C_c A_d^{i-1} B_d & C_c B_d & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{m} C_c A_d^{i-1} B_d & \sum_{i=1}^{m-1} C_c A_d^{i-1} B_d & \dots & \dots & C_c B_d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{p} C_c A_d^{i-1} B_d & \sum_{i=1}^{p-1} C_c A_d^{i-1} B_d & \dots & \dots & \sum_{i=1}^{p-m+1} C_c A_d^{i-1} B_d \end{bmatrix}_{\mathrm{p\times m}} \tag{30}$$

Objective Function:

$$\mathrm{J} = \sum_{i=1}^{p} \sum_{j=1}^{n_c} \left(\Gamma_{\mathrm{p}} \left(Y_p(k+i|k) - r_j(k+i) \right) \right)^2 + \sum_{i=1}^{p} \sum_{j=1}^{n_c} (\Gamma_{\mathrm{U}}(\Delta U(k))^2 \tag{31}$$

Γ_p is the weight factor to control the error between reference values and system feedback values. If Γ_p is getting larger, the controller hopes to get an output value which is close to the reference. What’s more, Γ_U is the weight factor of the input which is utilized to control the system input. Larger weight factor of input means larger expectation for input to achieve the desired value. Therefore, these two factors require tuning during the controller design process.

2.2.3 Optimization Problem in MPC

After getting the objective function and constraints, the MPC controller problem becomes an optimization problem. The purpose of optimization is to minimize the objective function with the constraint of inputs. For the blimp, the linear MPC will be implemented on the linearized model because the nonlinear MPC is time cost for the complicated blimp model. If the calculation time is larger than the sample time because of slow matrix calculation speed of MATLAB, the controller cannot draw out its capacity.

For optimization, there are hard constraints and soft constraints. The hard constraint is strictly limited to the constraint range. For soft constraints, the system will keep the violation of the constraints as small as possible by doing optimization. For the design of MPC, no hard constraints MPC is firstly introduced because in the final design there will be both soft design and hard design. The hard constraint will supervise the normal activity of input when the blimp stays in relatively stable status. The soft constraints will work when there's some situation required large input such as taking off and rejecting disturbances.

Firstly, unconstrained optimization provides some inspiration for the range of soft constraints and hard constraints. When there are no hard constraints added to the system, the optimization problem utilized the following method. To do the optimization problem of MPC, variable ρ and $E_p(k+1|k)$ was defined [4]:

$$\rho \stackrel{\text{def}}{=} \begin{bmatrix} \Gamma_p \left(Y_p(k+1|k) - R(k+1) \right) \\ \Gamma_u \Delta U(k) \end{bmatrix} \tag{32}$$

$$E_p(k+1|k) = R(k+1) - S_x \Delta x(k) - I y_c(k) \tag{33}$$

$$\Gamma_u = diag(\Gamma_{u,1}, \Gamma_{u,2}, \dots, \Gamma_{u,m}) \tag{34}$$

$$\Gamma_y = diag(\Gamma_{y,1}, \Gamma_{y,2}, \dots, \Gamma_{y,p}) \tag{35}$$

The reference value can also be summed to the form:

$$R(k+1) = \begin{bmatrix} r(k+1) \\ r(k+2) \\ \vdots \\ r(k+p) \end{bmatrix}_{p\times 1} \tag{36}$$

Because the reference value itself is a 3×1matrix for each $r(k+i)$, R(k+1) actually contains $(3p) \times 1$matrix.

The objective function can be changed to:

$$\mathrm{J}(\mathrm{x}(\mathrm{k}), \Delta\mathrm{U}(\mathrm{k}), \mathrm{m}, \mathrm{p}) = \mathrm{\rho}^{\mathrm{T}}\rho \tag{37}$$

And the optimization problem was changed to minimize $\mathrm{\rho}^{\mathrm{T}}\rho$. By taking estimation function into the defined variable ρ:

$$\mathrm{\rho} = \underbrace{\begin{bmatrix} \Gamma_y S_u \\ \Gamma_u \end{bmatrix}}_{A} \underbrace{\Delta\mathrm{U}(\mathrm{k})}_{z} - \underbrace{\begin{bmatrix} \Gamma_y E_p(k+1|k) \\ 0 \end{bmatrix}}_{b} \tag{38}$$

By setting the derivation of the objective function about variable z to be zero and the second derivation of the objective function to be larger than zero, the output of the system can be calculated to be:

$$\Delta\mathrm{U}(\mathrm{k}) = \left(\mathrm{S}_{\mathrm{u}}^{T}\Gamma_y^T\Gamma_y S_u + \Gamma_u^T\Gamma_u\right)^{-1} S_u^T\Gamma_y^T\Gamma_y E_p(k+1|k) \tag{39}$$

The output $\Delta\mathrm{U}(\mathrm{k})$ is the estimated result for the control range. According to MPC rule, the first element of $\Delta\mathrm{U}(\mathrm{k})$ will be applied to the system.

After unconstrained optimization, the constrained optimization is required to design the MPC controller because the input matrix needs to be limited within $[\pm 5\ \ \pm 5\ \ \pm 5]'$. For constrained optimization, Quadratic Programming is used to get the optimization result.

2.2.4 Parameters of the MPC controller

During the design process of MPC, there are many variables available for tuning. Firstly, the sample time of the MPC controller is required to be defined. This value is determined by the output of the system. When there is a step reference sent to the system, the rise time can be calculated. With the calculated rise time $\mathrm{T_r}$, sampling time $\mathrm{T_s}$ is set to the range $\frac{\mathrm{T_r}}{20} \leq T_s \leq \frac{T_r}{10}$. Secondly, the prediction horizon p should also be determined according to the step input model. Its value cannot be too long or too short. For long prediction horizon, some unexpected situation will influence the control result. For a short prediction horizon, the dynamic system does not have enough time to react to the situation. To determine the value of p, settling time should also be figured out through the step input figure. With the settling time, p is defined to be larger than $\frac{\mathrm{T_{settling}}}{T_s}$. Thirdly, for the value of the control horizon, it can be the same to p. However, m which is equal to p will lead to a large calculation load to the system. What's more, the only front part of input will make an influence on the output. The control horizon was decided to be $0.1\mathrm{p} \leq \mathrm{m} \leq 0.2\mathrm{p}$.

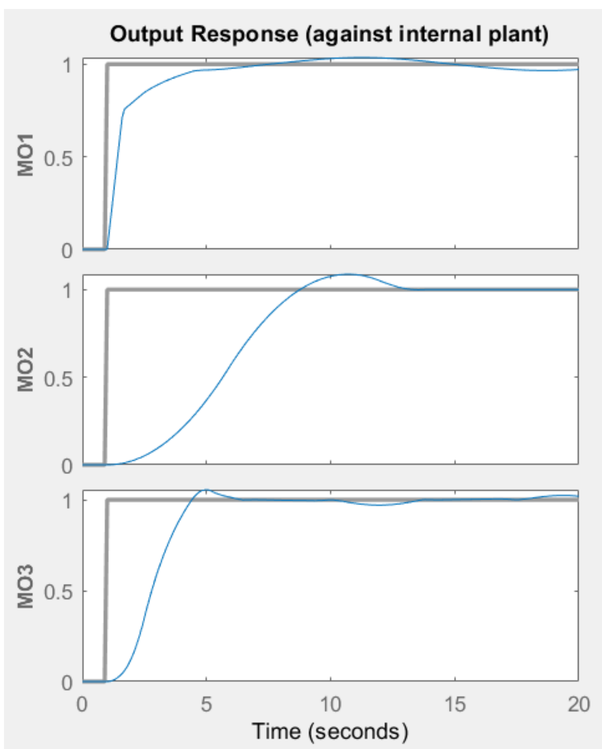

Figure 2: Step reference

2.4 Design of Sliding Mode Control

Sliding Mode Control is also introduced in this thesis to control the blimp. Firstly, the error functions were defined separately because velocity need integration once and position need integration twice. The control variables of the blimp are set to be velocity in the x-axis, position on the y-axis, and position on the z-axis.

Error Function: output result-reference

$$\mathrm{e}_1 = u - ud$$
$$e_2 = \begin{bmatrix} Posy \\ Posz \end{bmatrix} - \begin{bmatrix} Posyd \\ Poszd \end{bmatrix} \tag{40}$$

According to the integration times of the control variables, the sliding surfaces are different for velocity and position. The form of the sliding surface can be defined by the relationship between $\dot{s}$ and dynamic model because of equation $\dot{s} = -\dot{U}A$. This equation means that the change of the current state is opposite to the sign of sliding surface[7]. Therefore, the sliding surfaces are determined to be:

$$\mathrm{s}_1 = Ce_1$$
$$\mathrm{s}_2 = \dot{e_2} + \lambda e_2 \tag{41}$$

According to stability theory Lyapunov function:

$$\mathrm{V(x)} = \frac{1}{2} s(x)^T s(x) \tag{42}$$

$$\dot{\mathrm{V}} = \mathrm{s}^{\mathrm{T}}\dot{s} < 0$$

$$\dot{\mathrm{V}} = -s^T\zeta s \tag{43}$$

In the equation above, ζ must be a positive value to achieve function stability.

$$s^T(\zeta s + \dot{s}) = 0 \tag{44}$$

$$\zeta s + \dot{s} = 0 \tag{45}$$

$$\dot{s} = \begin{bmatrix} C\dot{e_1} \\ \ddot{e_2} + \lambda\dot{e_2} \end{bmatrix} = \begin{bmatrix} C\dot{U}_A \\ \dot{U}_A + \lambda U_A \end{bmatrix} = -\zeta s \tag{46}$$

For the following analysis, $C\dot{U}_A$ was assumed to be $\dot{U}_A$ because C and λ are positive constants which can be moved to the right of the expression. In the following equations, s will only be a variable instead of vector because its content will not influence the derivation and $\dot{s}$ was assumed to be $\dot{U}_A$ for both sliding surfaces.

To get the expression of input, the expression of the dynamic model was changed:

$$D_1 = [(M)I_{3\times3} - A_1]^{-1}\{M(S(\omega_{BA})U_A) - MS^2(\omega_{BA})\rho_{cm}\} \tag{47}$$

$$D_2 = (I_t - M_1)^{-1}\{S(\omega_{BA})I_M(\omega_{BA}) + MS(\rho_{cm})[-S(\omega_{BA})U_A]\} \tag{48}$$

$$G_1 = (I_{3\times3} - f_1f_2)^{-1}[(M)I_{3\times3} - A_1]^{-1} \tag{49}$$

$$G_2 = (I_t - M_1)^{-1} f_1 (I_t - M_1)^{-1} \tag{50}$$

$$\dot{U}_A = (I_{3\times3} - f_1 f_2)^{-1}(f_1 D_2 + D_1) + (G_1 + G_2 G_m)u \tag{51}$$

By taking this equation back the expression of derivative of sliding surface.

Therefore, the input of the system based on Sliding Mode Control is:

For position input:

$$u = -(G_1 + G_2 G_m)^{-1}(\zeta_1 s + \lambda U_A) - (G_1 + G_2 G_m)^{-1}\left((I_{3\times3} - f_1 f_2)^{-1}(f_1 D_1 + D_1)\right) \tag{52}$$

For velocity input:

$$u = (G_1 + G_2 G_m)^{-1}\left(\frac{\zeta_2 s}{C}\right) - (G_1 + G_2 G_m)^{-1}\left((I_{3\times3} - f_1 f_2)^{-1}(f_1 D_1 + D_1)\right) \tag{53}$$

Sliding surface s_2 was shown in the figure below.

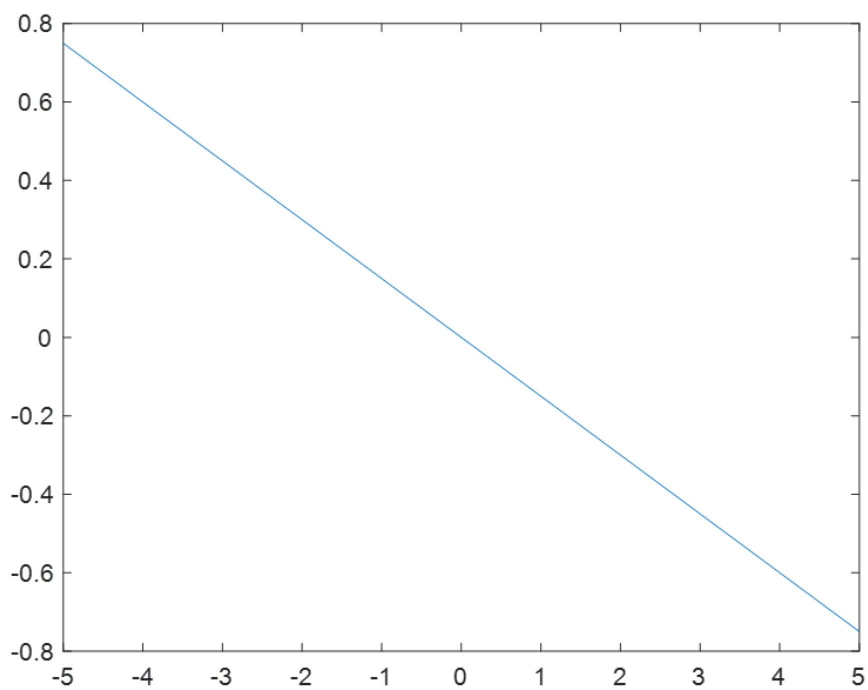

Figure 3: s_2 sliding surface (x is error and y is error derivative)

2.5 Indoor flight test of the blimp

Except for the large blimp built by AOL, a smaller blimp was also built. This blimp was much smaller than the existing large blimp and could be used to test indoor for trajectory tracking. Before the indoor flight test, a simulation had done to check the controller performance.

Figure 4: The small blimp

The main purpose of this test is to check the target following or trajectory tracking ability of the blimp with MPC and SMC controller. In the test, the Vicon system, small blimp, and the target markers were used in the test. For this blimp, Vicon was used for position tracking and there will be three markers placed on the blimp. Vicon system can provide the position of three markers and their positions can provide the status of the blimp. Also, the Vicon system can provide the target position through markers, which will be used as reference values.

Chapter 3 Results: Simulation Result

3.1 MPC controller & Linearization

To test the performances of two controllers, the dynamic model of the blimp below was modified based on existing Simulink Model to simulate the controller's performance in indoor autonomous flight. To use the MPC controller, the dynamic system required linearization about $[u\ py\ pz]$.

Firstly, the linearized system was set to be time-invariant because the main result was to compare the output of linear and nonlinear systems under steady-state input. The linearized system and nonlinear system were compared by sending the equal status input to both systems. The simulation results were shown below.

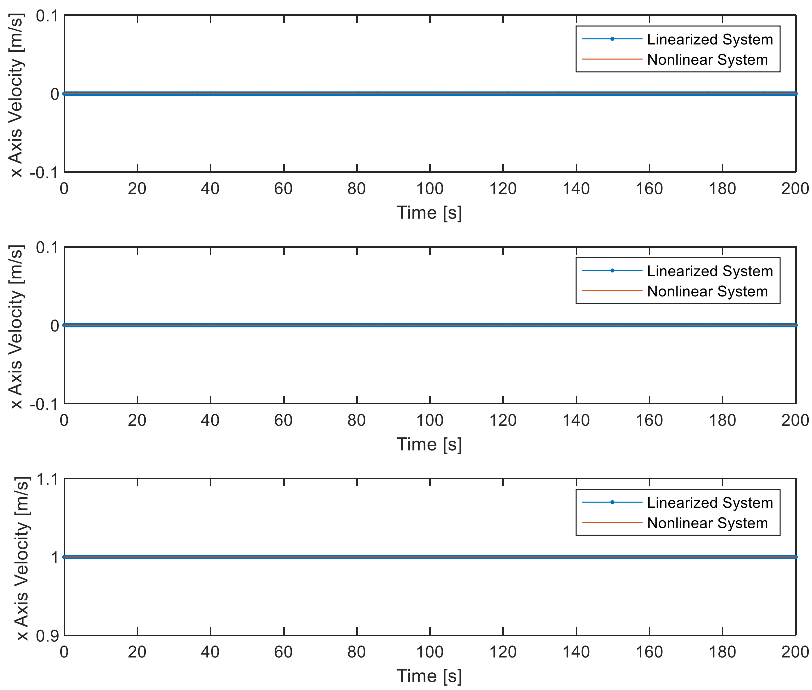

Figure 5: Linearized and nonlinear system at steady-state

In the figure above, the linearized system and nonlinear system both achieve steady status with the same input. Although the equilibrium points found are only local equilibrium point, the linearized system can still be used to predict the performance of the blimp in the situation without too many changes on its status.

Secondly, the time-vary linearized system was derived to test its performance on steady state. For time-vary linearization, the nonlinear system was linearized when the dynamic system status changed. After building the Simulink Model and test its performance on a steady state, the result showed that the time-vary linearization was very time cost which led to at least two seconds for each linearization. When this linearization was combined with the MPC controller, the controller did not work because the sample time of the MPC controller could not be set as 2 seconds. When the sample time was 2 seconds, the input could not make any change within 2 seconds. After testing with time-vary linearization, a time-invariant linearized system was used for further design.

After getting the linearized system of the blimp, the Model Predictive Controller was also built in the Simulink to its performance in controlling the blimp. For the blimp, the sample time was set to 0.1 seconds, the prediction horizon was 80, and the control horizon was 10. Except for these parameters, the constraints added to the system was set to be larger than -5 N and smaller than 5 N. According to these settings, the MPC controller was built inside the Simulink Model. For the output of MPC controller, values were added by steady-state input because the MPC controller sent out ΔU. By adding steady-state inputs, the results were the correct input which can make the blimp stable. For reference values $[u\ py\ pz] =$ $[desired\ x\ speed, desired\ y\ position, desired\ z\ position]$, they would change under different situations.

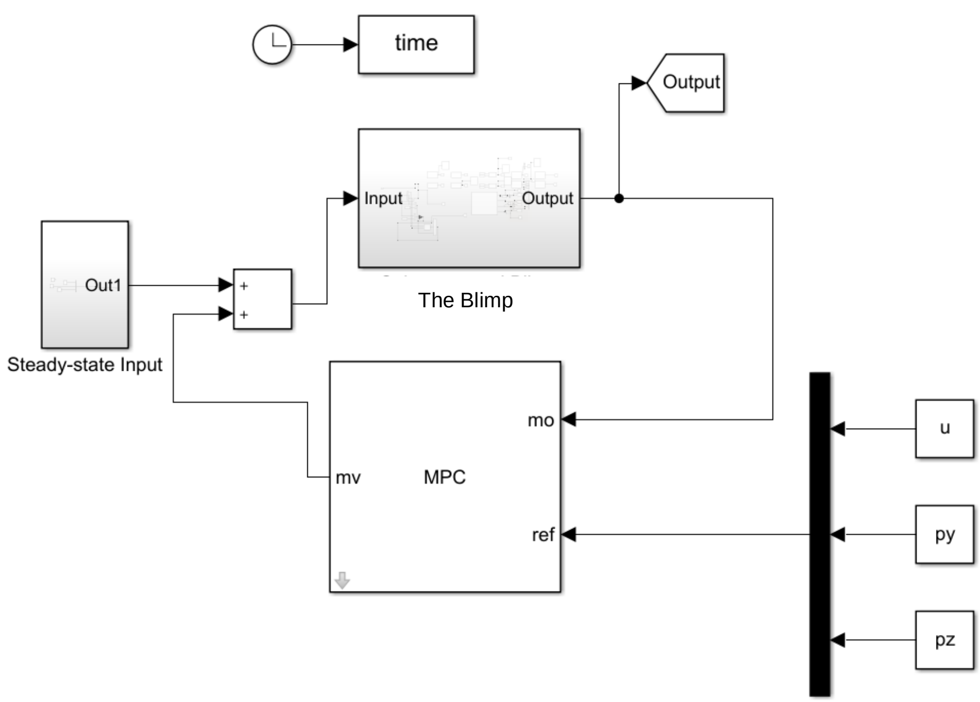

Figure 6: Simulink block of MPC controller and the simulated blimp

Before testing, the designed MPC controller's stabilities and performance were also checked by MATLAB internal function. The result showed that the controller passed all tests.

Test	Status
MPC Object Creation	Pass
QP Hessian Matrix Validity	Pass
Closed-Loop Internal Stability	Pass
Closed-Loop Nominal Stability	Pass
Closed-Loop Steady-State Gains	Pass
Hard MV Constraints	Pass
Other Hard Constraints	Pass
Soft Constraints	Pass
Memory Size for MPC Data	Pass

Figure 7: Review of the designed MPC controller

Firstly, by setting desired x-axis speed to 1 m/s, desired y position to be 0 m, and desired z position to be 1 m. The simulation result within 200 seconds as shown below. In the figure, the x-axis speed did not achieve exactly 1 m/s and z-axis position had a vibration at the start which was caused by the velocity change in the x-axis. For the built blimp, there were only two motors which provide all forces. When the blimp wanted to go forward, it leaned forward. Therefore, there were some changes to the blimp altitude (z-axis position). In this process, y-axis was very stable because the forward movement did not influence the blimp on the y-axis. For x-axis speed, 0 m/s was the steady-state speed. When the speed changed from 0 m/s to 1 m/s, the previous linearized model was hard to predict the performance of the blimp. Therefore, the blimp did not achieve the desired value.

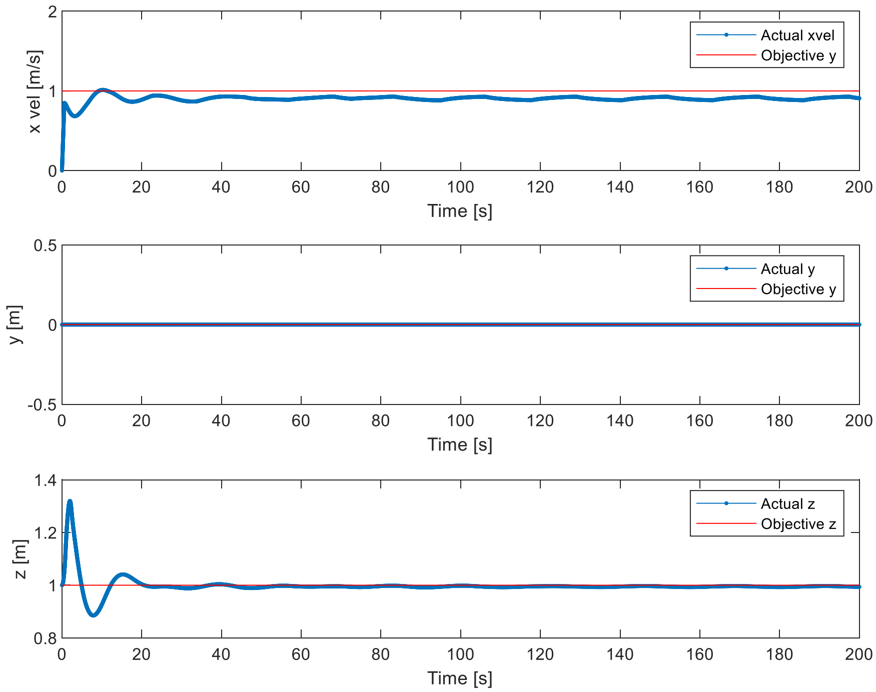

Figure 8: Using MPC to the blimp going forward with x_vel=1m/s

On the other hand, the energy cost was hard to estimate because the simulated model did not include the electrical energy and the input of the system was assumed to be the forces in three directions. To estimate the energy cost, the force could be considered as motor power at that time. The areas encircled by force curves and x-axis could be considered as energy cost. In the figure below, the energy cost to push the blimp forward was large when the blimp started from 0 m/s to 1 m/s. After taking off, the force/power kept at a steady level. On the z axis, the same situation happened which was caused by taking off. For side force, the energy cost was smaller than 2×10^{-5} N all the time and could be ignored.

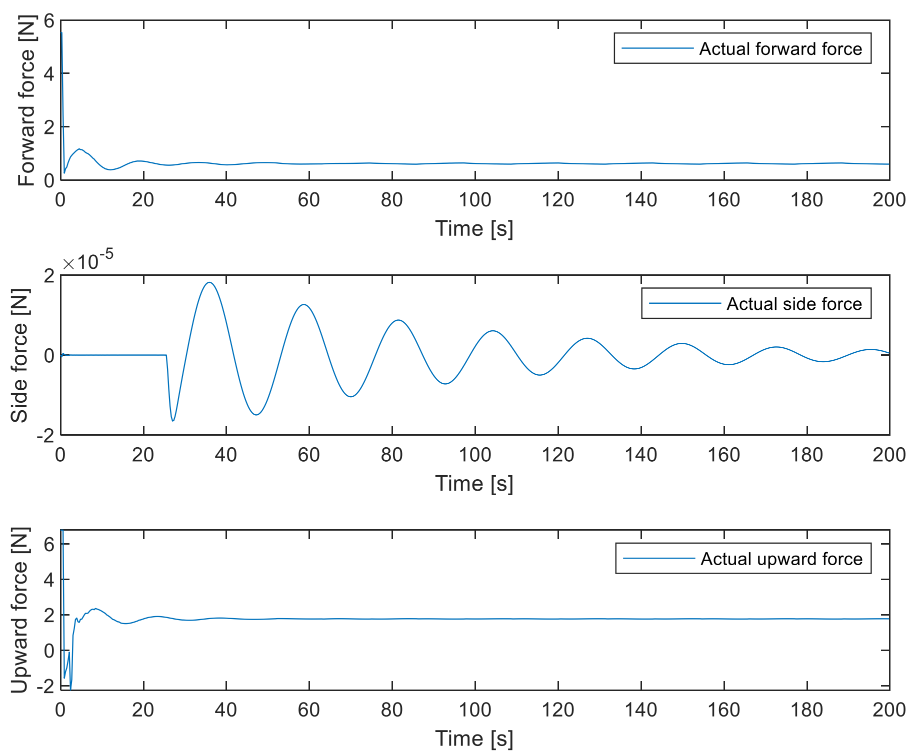

Figure 9: Input force of the blimp while going forward

Secondly, the objective x-axis velocity and the blimp altitude was set to be steady state. The objective y-axis was a combination of step inputs. The simulation results were shown in the figure below. At steady state, all three variables are constant. After the first step input, all three variables changed because of limitation of two motors. What's more, the y-axis position went to a short stable status after a relatively long settling time. The disadvantage of long settling time was also shown in the next two step inputs whose amplitudes were larger. This situation was not caused by the accuracy of the linearized model because the time-invariant model cannot estimate accurately when the actual model changed to different states. The inaccuracy of prediction increased with the increasing change of model states.

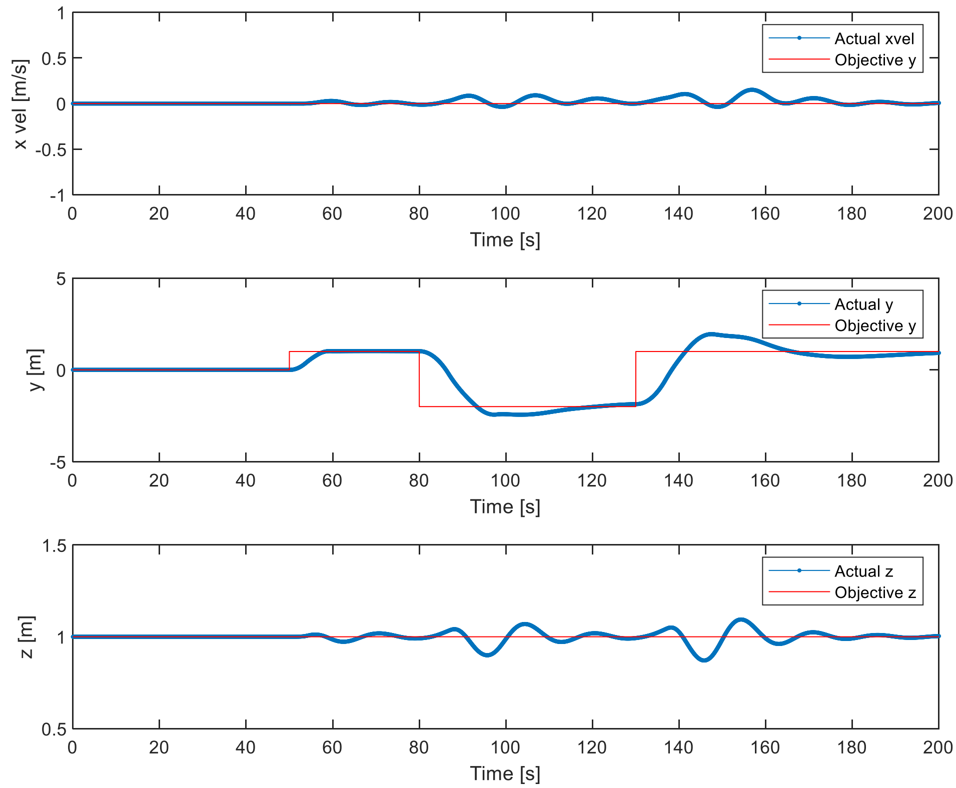

Figure 10: MPC in controlling y position

In the figure below, the energy cost for the simulation was shown. For side force, the requires force achieved the constraints several times when the desired y position changed. During the rest time, the side force was close to zero. For forward and upward force, they were both controlled within 5 N all the time without touching the constraints.

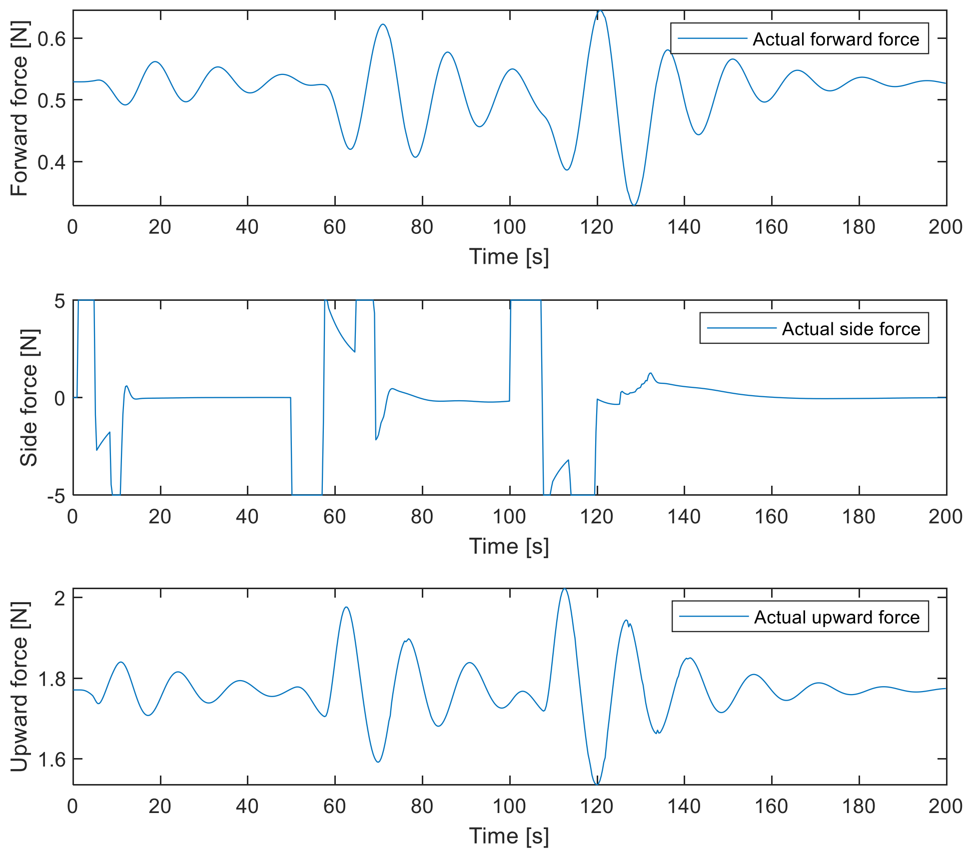

Figure 11: Input force in controlling y position

The above simulation result was done when there's no wind disturbance. After adding wind disturbances to the simulation, the MPC cannot keep steady state when the input was constrained to $\pm$ 5 N. In figure 11, x position should be around zero because the controlled variable was x velocity which was zero. In addition, y and z positions were far away from the zero axis which was the objective line.

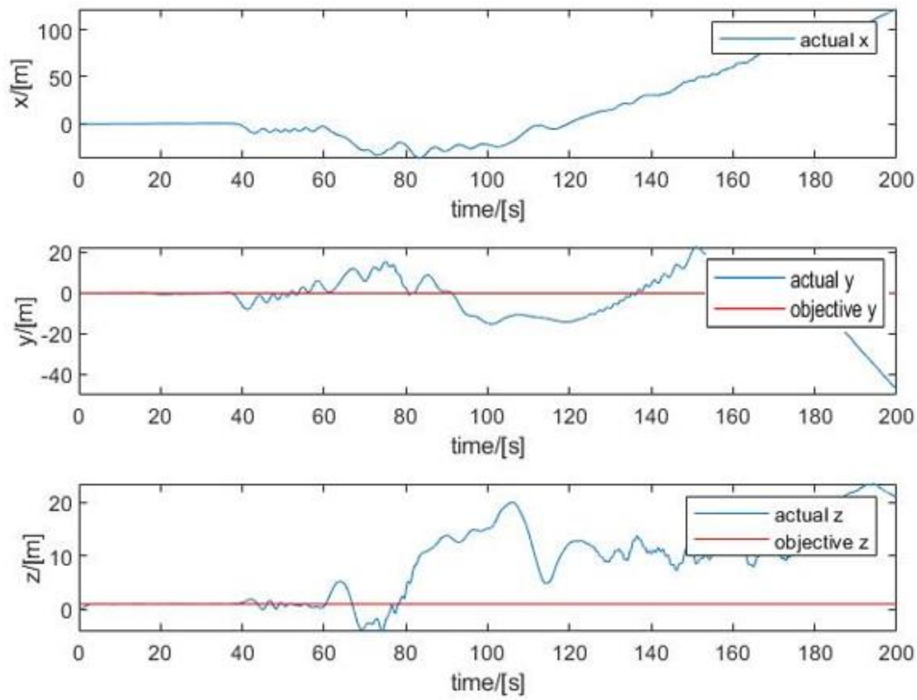

Figure 12: MPC with wind disturbance

Although MPC controller can add some measured disturbance or unknown disturbance to the controller design, these additions did not work when there was a large wind disturbance and there was a very influenced constraint to the input. Therefore, MPC will only be used for indoor flight test.

3.2 SMC controller

A sliding mode controller was also designed for comparison. The sliding surface was designed as follows. The parameters were set for the test:

$$C = 0.6$$

$$\lambda = 0.15$$

$$\zeta \, for \, velocity = 8$$

$$\zeta \, for \, position = 150$$

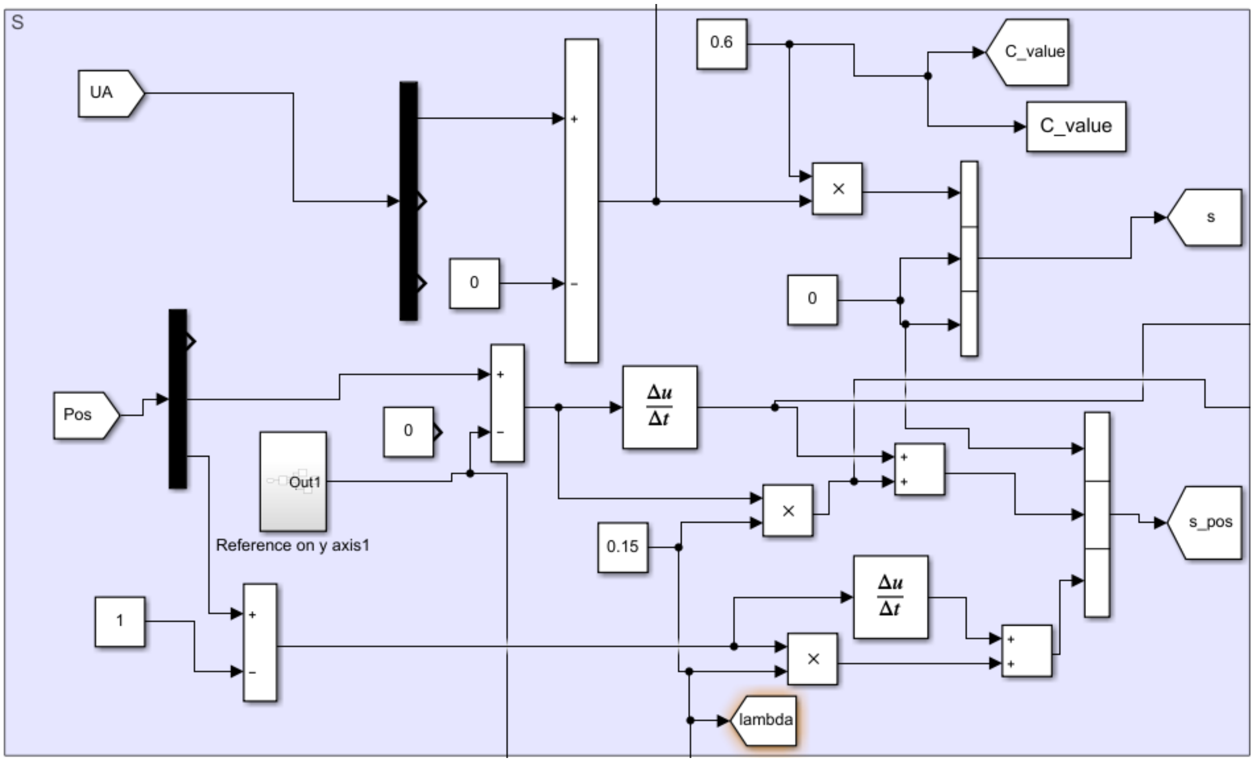

Figure 13: Sliding surfaces

The sliding mode controller was tested to do the same thing as the model predictive controller. To control the input, the saturation block was used in the Simulink so that the input did not get a value larger than 5 N or smaller than 5 N. The inputs which were not in the range of $[-5,5]$ would not be used. In the figure below, the control result showed that SMC had a better performance in outdoor simulation test where the disturbance was very large. In addition, the wind disturbance influence can also be found in the input figure because the input will change with the wind speed change. Although the SMC controller had a relatively large energy cost in the y position tracking test without wind disturbance, it showed a strong ability in controlling this blimp under large wind disturbance which was up to 6 m/s. In addition, the force of rejecting wind disturbances was controlled within $\pm$ 5 N.

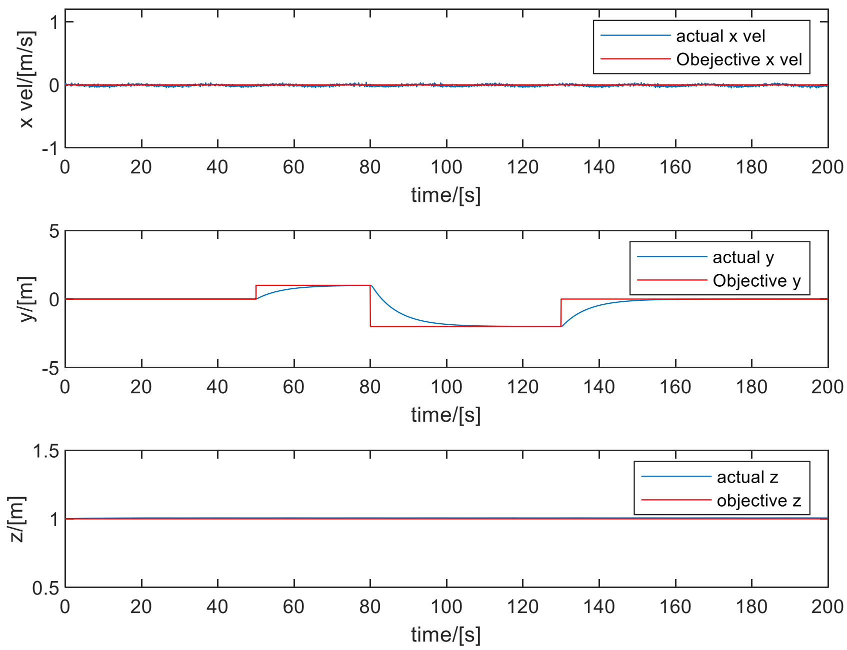

Figure 14: SMC controller in controlling y position

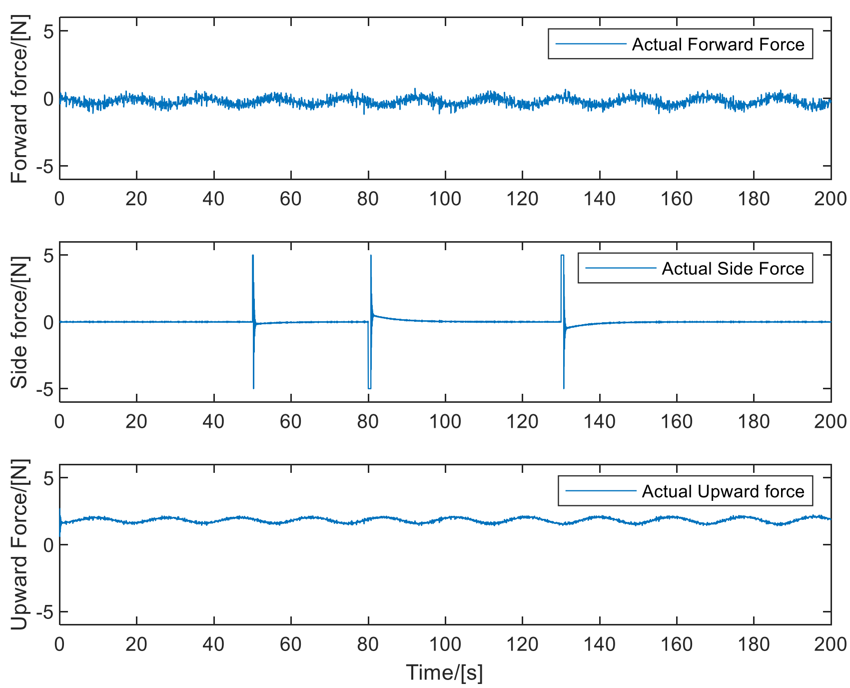

Figure 15: Input of SMC controller in controlling y position

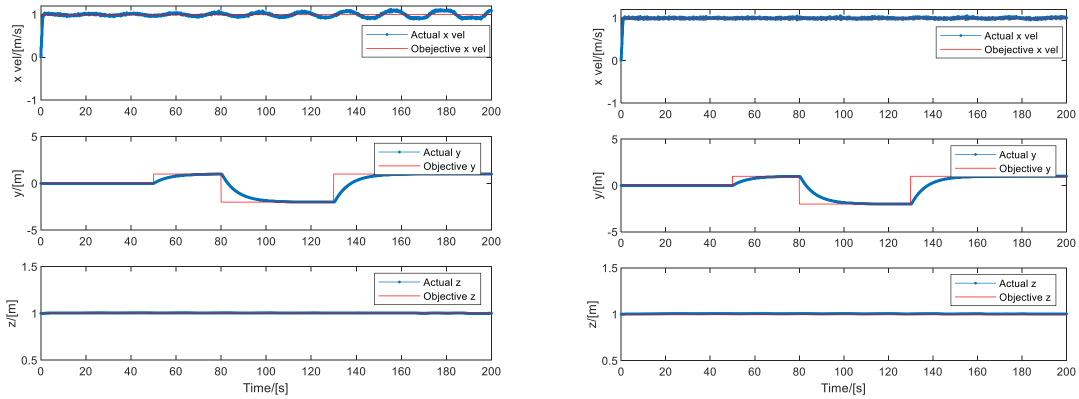

Figure 16: SMC controller in controlling y position with wind disturbance

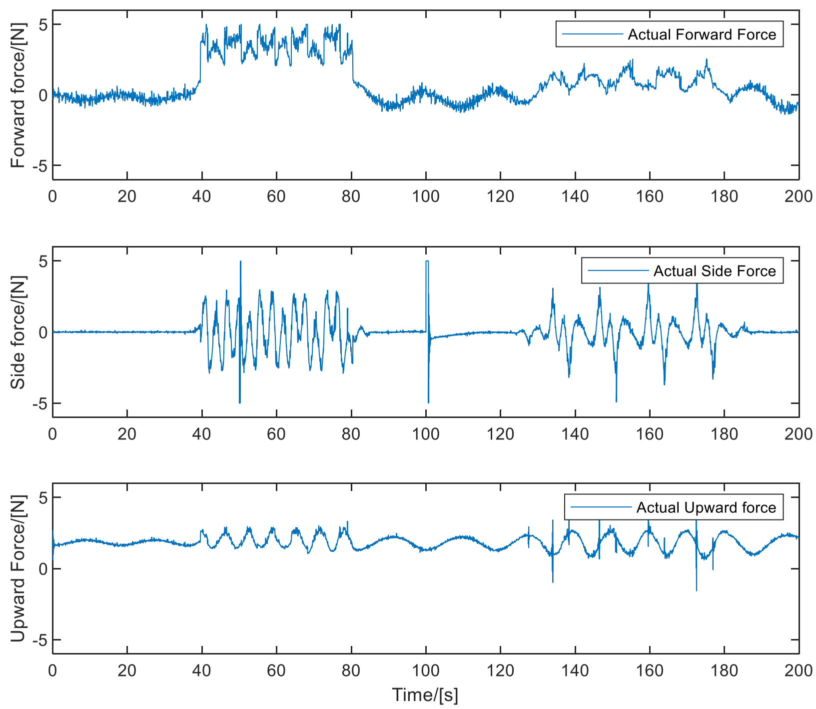

Figure 17: Input of SMC controller in controlling y position with wind disturbance

According to the figures above, the SMC controller had a better performance in controlling the blimp in an area with large disturbance than the MPC controller. However, this result came from changing the C value inside the sliding surface to 4. Therefore, this led to an increased input even though it is still controlled within constraints

3.3 Explanation of Linearization Parameters

The reason for using $[u\ py\ pz]$ for linearization can also be proved. For this linearization, $[px\ py\ pz]$ could not be set as control variables even though they were easier to be used in trajectory following. The following simulation results showed the difference in tracking the same trajectory with the different linearized system. Among these two figures, objectives were set to be the same except for the first plot because one objective was position and another one was velocity. In the first figure, the x position was assumed to be $time \times 1\ m/s$, which is a straight line. In the second figure, the velocity was set to be $1\ m/s$ and the blimp traveling distance was the same to the previous one. Therefore, these two tests had the same trajectory.

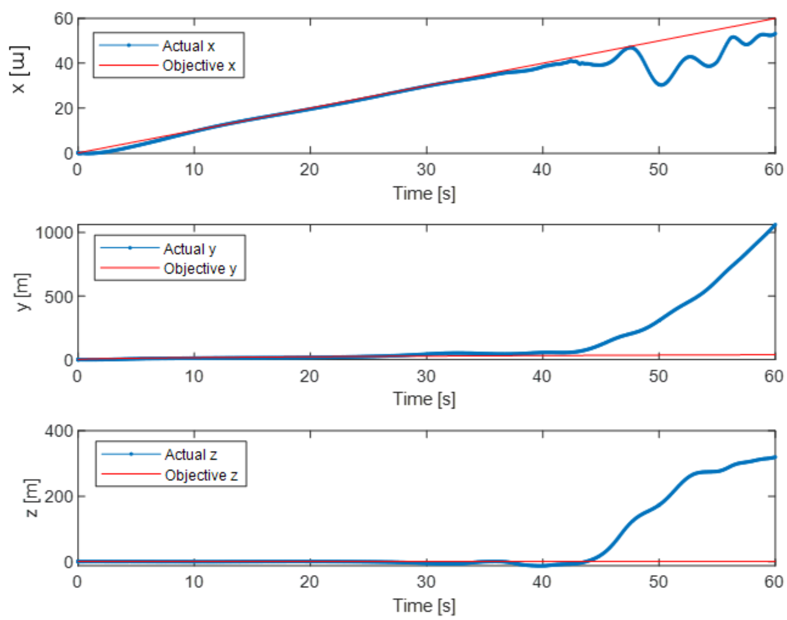

Figure 18: Choosing $[px\ py\ pz]$

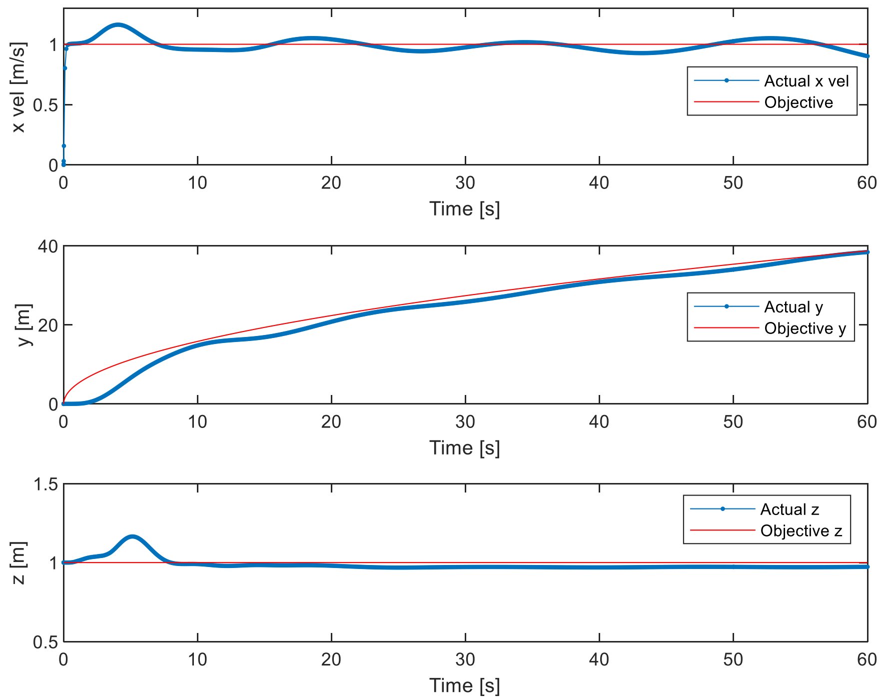

Figure 19: Choosing $[u\ py\ pz]$

By comparing the figures above, the disadvantage of using three position variables was found that the model accuracy decreased with the change of blimp status. When the traveling distance was larger than 45 m on the x-axis, the controller cannot control the blimp toward the reference values. By contrast, using $[u\ py\ pz]$ could provide a much better result in achieving all three desired values.

Chapter 4 Conclusion

4.1 Future Work

Future work will focus on Indoor flight test with the Vicon system. By now the codes can provide the position but the lab computer with Vicon system cannot run MPC controller designed because of lack of toolboxes. After solving this problem, indoor flight test will be able to start.

Figure 20: Cameras for VICON system

4.2 Summary

To design the required control system for the blimp, model predictive control and sliding mode control was studied and compared in this thesis. In the MPC controller design process, the linearization process cost a bunch of time of this project because the original purpose was to get a time -vary linearization system. However, in the linearization process, there were many matrix calculations which decrease the calculation speed and led to a difficulty in using an MPC controller. After doing some tests with time-invariant linearization system, the control performance was stable but there were some vibrations, which led to the utilization of time-invariant linearization. The linear MPC was not good at controlling the nonlinear system by using linearization. The advantage of SMC in following the trajectory and strong robustness leads to vibrations on all three axes. Although this controller will lead to vibration, its robustness can help to solve the disturbance during the outdoor flight test and SMC has the best performance in the simulation. The advantage of MPC in energy cost can bring some benefits in the steady state. Finally, SMC and MPC controllers will be designed to be tested in a smaller blimp for indoor trajectory test with wind created by fans. Further work will finish this flight test.

Appendices

MATLAB Code for Sliding Mode Control

```
close all
clear
clc

%% parameter for blimp

rho_air=1.225; % density of air

rho_h=0.1624; %density of helium

V=3.553; %volume of blimp

M=3.964; %Mass of blimp

g=9.8; %the accelerate of gravity

A1=[-0.359794831666667,0,0;0,-4.96845790000000,0;0,0,-
4.80976731666667];

A2=[0,0,0;0.355178378333333,0,16.4143768833333;0,-
15.9469609833333,0];

rhocm=[0.05,0,0.2]';%0.2

% FG=[0 0 0]';
FG=-[0,0,M*g-(rho_air-rho_h)*g*V]';

It=[0.232221359481588,0,0;
0,0.473303439663614,0;
0,0,0.804876866745891];

Im=It;

M1=[-1.48481119230769,  0,  -0.405937170512821;
0,  -34.6123027564103,  0;
-0.405937170512821, 0,  -35.1560676923077];
% M1=zeros(3,3);
M2=[0,  0.136607068589744,  0;
0,  0,  -6.13344653205128;
0,  6.31322187820513,   0];
```

```
G=[0 0 g]'; %G vector in the inertial frame

L=3; %characteristic length of blimp

Wind=[1 1 1]'; % initial speed of wind

L_fp=[0.03936 0 -0.81836]; %arm of main propeller
L_bp=[-1.4795 0.012 -0.56128];
sysHz = 100;
% L_bp=[-1.4795 0.012 0];

%% run model

sim('blimp_model8_SMC.slx')

% plot figure
%
figure(1)
subplot(3,1,1)
plot(time,UA(:,1));

hold on
plot(time,1*ones(length(time),1),'r');
xlabel('time/[s]');
ylabel('dx/dt/[m/s]');
%ylim([0,2]);
legend('actual speed','objective speed');
filename=['Wind
Speed',num2str(Wind(1)),num2str(Wind(2)),num2str(Wind(3))];
savefig(filename)
subplot(3,1,2)
plot(time,UA(:,2));
hold on
plot(time,zeros(length(time),1),'r');
xlabel('time/[s]');
ylabel('dy/dt/[m/s]');
%ylim([-1,1])
legend('actual speed','objective speed');
subplot(3,1,3)
plot(time,UA(:,3));
hold on
plot(time,zeros(length(time),1),'r');
xlabel('time/[s]');
ylabel('dz/dt/[m/s]');
```

```
%ylim([-1,1])
legend('actual speed','objective speed');

%position
figure(2)
subplot(3,1,1)
plot(time,UA(:,1),'.-');
hold on
plot(time,1*ones(length(time),1),'r');

ylabel('x vel/[m/s]');
ylim([-1,1.2])
legend('Actual x vel','Obejective x vel');
filename=['Wind
Speed',num2str(Wind(1)),num2str(Wind(2)),num2str(Wind(3))];
savefig(filename)
subplot(3,1,2)
plot(time,Pos(:,2),'.-');
hold on
plot(time,Reference,'r');

ylabel('y/[m]');
legend('Actual y','Objective y');
ylim([-5,5])
subplot(3,1,3)
plot(time,Pos(:,3),'.-');
hold on
plot(time,ones(length(time),1),'r');
xlabel('Time/[s]');
ylabel('z/[m]');
ylim([0.5,1.5]);
legend('Actual z','Objective z');




figure(3)
subplot(3,1,1)
plot(time,wind(:,1));

ylabel('Wind speed/[m/s]');
legend('Wind speed in x direction');
subplot(3,1,2)
plot(time,wind(:,2));
```

```
ylabel('Wind speed/[m/s]');
legend('Wind speed in y direction');
subplot(3,1,3)
plot(time,wind(:,3));
xlabel('Time/[s]');
ylabel('Wind speed/[m/s]');
legend('Wind speed in z direction');
% % Force
figure(4)
subplot(3,1,1)
plot(time,PA(:,1));
ylabel('Forward force/[N]');
legend('Actual Forward Force');
ylim([-6,6])
subplot(3,1,2)
plot(time,PA(:,2));

ylabel('Side force/[N]');
legend('Actual Side Force');
ylim([-6,6])
subplot(3,1,3)
plot(time,PA(:,3));
xlabel('Time/[s]');
ylabel('Upward Force/[N]');
legend('Actual Upward force');
ylim([-6,6])
```

MATLAB Code for Model Predictive Control

```
%% parameter for blimp
clc
clear
close all

rho_air=1.225; % density of air
input=[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0];
rho_h=0.1624; %density of helium

V=3.553; %volume of blimp
Wind=[1 1 1]';
M=3.964; %Mass of blimp

g=9.8; %the accelerate of gravity
```

```
G=[0 0 g]'; %G vector in the inertial frame
A1=[-0.359794831666667,0,0;0,-4.96845790000000,0;0,0,-
4.80976731666667];

A2=[0,0,0;0.355178378333333,0,16.4143768833333;0,-
15.9469609833333,0];

rhocm=[0.05,0,0.2]';%0.2

% FG=[0 0 0]';
FG=-[0,0,M*g-(rho_air-rho_h)*g*V]';

It=[0.232221359481588,0,0;
0,0.473303439663614,0;
0,0,0.804876866745891];

Im=It;

M1=[-1.48481119230769,  0,  -0.405937170512821;
0,  -34.6123027564103,  0;
-0.405937170512821, 0,  -35.1560676923077];
% M1=zeros(3,3);
M2=[0,  0.136607068589744,  0;
0,  0,  -6.13344653205128;
0,  6.31322187820513,   0];

G=[0 0 g]'; %G vector in the inertial frame

L=3; %characteristic length of blimp


L_fp=[0.03936 0 -0.81836]; %arm of main propeller
L_bp=[-1.4795 0.012 -0.56128];

%% Linearization

u=1;
v=0;
w=0;
p=0;
q=0;
r=0;
wx=input(7);
wy=input(8);
```

```
wz=input(9);
PAx=2;
PAy=0;
PAz=3;
phi=0;
theta=0;
psi=0;
px=0;
py=0;
pz=1;

% find equil point
st=[u v w px py pz phi theta psi p q r]';
in=[PAx PAy PAz]';
[state,inp,out,dx]=trim('model_nonli',st,in,[u py pz]')
u=state(1);
v=state(2);
w=state(3);
px=state(4);
py=state(5);
pz=state(6);

phi=state(7);
theta=state(8);
psi=state(9);

p=state(10);
q=state(11);
r=state(12);

PAx=inp(1);
PAy=inp(2);
PAz=inp(3);
%% code

%% model
mod=linmod('model_nonli_3_26',state,inp);
A=mod.a;
B=mod.b;
C=mod.c;
D=mod.d;

% mod1=linearize('model_nonli');
% Am=mod1.A;
```

```
% Bm=mod1.B;
% Cm=mod1.C;
% Dm=mod1.D;
% sim('linear_model_b')
% plot(time,result(:,1))
%
plant_linear=minreal(ss(A,B,C,D));

% Tis=0.5;
% pt=25;
% mt=5;
% mpco=mpc(plant_linear,Tis,pt,mt);
%% create MPC controller object with sample time
mpco_Copy = mpc(plant_linear, 0.1);
%% specify prediction horizon
mpco_Copy.PredictionHorizon = 80;
%% specify control horizon
mpco_Copy.ControlHorizon = 10;
%% specify nominal values for inputs and outputs
mpco_Copy.Model.Nominal.U = [0;0;0];
mpco_Copy.Model.Nominal.Y = [u;py;1];
%% specify constraints for MV and MV Rate
mpco_Copy.MV(1).Min = -(-PAx+5);
mpco_Copy.MV(1).Max = (-PAx+5);
mpco_Copy.MV(2).Min = -(-PAy+5);
mpco_Copy.MV(2).Max = (-PAy+5);
mpco_Copy.MV(3).Min = -(-PAz+5);
mpco_Copy.MV(3).Max = (-PAz+5);
%% specify constraints for OV
mpco_Copy.OV(1).Max = 5;
mpco_Copy.OV(2).Max = 5;
mpco_Copy.OV(3).Max = 5;
%% specify overall adjustment factor applied to weights
beta = 4.953;
%% specify weights
mpco_Copy.Weights.MV = [0 0 0]*beta;
mpco_Copy.Weights.MVRate = [0.00242339473838763
0.00242339473838763 0.00242339473838763]/beta;
mpco_Copy.Weights.OV = [41.26442895 41.26442895
41.26442895]*beta;
mpco_Copy.Weights.ECR = 100000;
%% specify overall adjustment factor applied to estimation
model gains
alpha = 6.3096;
```

```
%% adjust custom output disturbance model gains
%setoutdist(mpco_Copy, 'model', mpco_Copy_ModelOD*alpha);
%% adjust default measurement noise model gains
mpco_Copy.Model.Noise = mpco_Copy.Model.Noise/alpha;
%% specify simulation options
options = mpcsimopt();
options.RefLookAhead = 'off';
options.MDLookAhead = 'off';
options.Constraints = 'on';
options.OpenLoop = 'off';




mdl_b = 'nonlinear_model_3_26';
open_system(mdl_b)     % Open Simulink Model
sim(mdl_b);            % Start Simulation
figure(1)
subplot(3,1,1)
plot(time,UA(:,1));

hold on
plot(time,0*ones(length(time),1),'r');
xlabel('time/[s]');
ylabel('dx/dt/[m/s]');
%ylim([0,2]);
legend('actual speed','objective speed');
filename=['Wind
Speed',num2str(Wind(1)),num2str(Wind(2)),num2str(Wind(3))];
savefig(filename)
subplot(3,1,2)
plot(time,UA(:,2));
hold on
plot(time,zeros(length(time),1),'r');
xlabel('time/[s]');
ylabel('dy/dt/[m/s]');
%ylim([-1,1])
legend('actual speed','objective speed');
subplot(3,1,3)
plot(time,UA(:,3));
hold on
plot(time,zeros(length(time),1),'r');
xlabel('time/[s]');
ylabel('dz/dt/[m/s]');
```

```
%ylim([-1,1])
legend('actual speed','objective speed');

%position
figure(2)
subplot(3,1,1)
plot(time,UA(:,1),'.-');
hold on
plot(time,u*ones(length(time),1),'r');
ylim([-1 1])

ylabel('x vel [m/s]');
legend('Actual x vel','Objective y');
filename=['Wind
Speed',num2str(Wind(1)),num2str(Wind(2)),num2str(Wind(3))];
savefig(filename)
subplot(3,1,2)
plot(time,Pos(:,2),'.-');

hold on
plot(time,Ref,'r');

ylabel('y [m]');
legend('Actual y','Objective y');
ylim([-5,5])
subplot(3,1,3)
plot(time,Pos(:,3),'.-');
hold on
plot(time,pz*ones(length(time),1),'r');
xlabel('Time [s]');
ylabel('z [m]');
ylim([0.5,1.5]);
legend('Actual z','Objective z');



figure(3)
subplot(3,1,1)
plot(time,wind(:,1));
xlabel('time/[s]');
ylabel('Wind speed/[m/s]');
legend('Wind speed in x direction');
subplot(3,1,2)
plot(time,wind(:,2));
```

```
xlabel('time/[s]');
ylabel('Wind speed/[m/s]');
legend('Wind speed in y direction');
subplot(3,1,3)
plot(time,wind(:,3));
xlabel('time/[s]');
ylabel('Wind speed/[m/s]');
legend('Wind speed in z direction');
```

References

[1] “The Hindenburg Disaster.” *Airships.net*, www.airships.net/hindenburg/disaster/.

[2] Goodyearblimp.com. (2019). *Current Blimps | Goodyear Blimp*. [online] Available at: https://www.goodyearblimp.com/behind-the-scenes/current-blimps.html [Accessed 25 Mar. 2019].

[3] S. Simrock. Control theory. In Proceedings of the CERN Accelerator School on Digital Signal Processing, 2007.

[4] Chen, H. (2013). *Systems and Control Series: Model Predictive Control (Chinese Edition)*. 1st ed. Beijing: Science Press.

[5] Chen, Z., Wang, Z. and Zhang, J. (2012). *Sliding Mode Variable Structure Control Theory and Application*. 1st ed. Beijing: Publishing House of Electronic Industry.

[6] Waishek, Jayme & Dogan, Atilla & Bestaoui, Yasmina. (2009). Investigation into the Time Varying Mass Effect on Airship Dynamics Response. 47th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition. 10.2514/6.2009-735.

[7] K. Erbatur, O. Kaynak, and A. Sabanovic, “A study on robustness property of sliding mode controllers: A novel design and experimental investigations,” IEEE Trans. Ind. Electron., vol. 46, no. 5, pp. 1012–1018, Oct. 1999.