

Advanced Control Design of an Autonomous Line Painting Robot

Mincan Cao

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

Alexander Leonessa, Chair

Steve Southward

Tomonari Furukawa

April 14, 2017

Blacksburg, Virginia

Keywords: Computer Vision, Adaptive Control, Painting Robot

Copyright 2017, Mincan Cao

Advanced Control Design of an Autonomous Line Painting Robot

Mincan Cao

ABSTRACT

Painting still plays a fundamental role in communication nowadays. For example, the paint on the road, called road surface marking, guides the traffic in order and maintains the high efficiency of the entire modern traffic system. With the development of the Autonomous Ground Vehicle (AGV), the idea of a line Painting Robot emerged. In this thesis, a Painting Robot was designed as a standalone system based on the AGV platform.

In this study, the mechanical and electronic design of a Painting Robot was discussed. The overall design was to fulfill the requirements of the line painting. Computer vision techniques were applied to this thesis since the camera was selected as the major sensor of the robot. Advanced control theory was introduced to this thesis as well. Three different controllers were developed. The Proportional-Integral (PI) controller with an anti-windup feature was designed to overcome the drawbacks of the traditional PI controller. Model Reference Adaptive Control (MRAC) was introduced into this thesis to deal with the uncertainties of the system. At last, the hybrid PI-MRAC controller was implemented to maintain the advantages of both PI and MRAC approaches. Experiments were conducted to evaluate the performance of the entire system, which indicated the successful design of the Painting Robot.

Advanced Control Design of an Autonomous Line Painting Robot

Mincan Cao

GENERAL AUDIENCE ABSTRACT

Painting still plays a fundamental role in communication nowadays. With the development of the Autonomous Ground Vehicle (AGV), the idea of a line Painting Robot emerged. In this thesis, a Painting Robot was designed as a standalone system based on the AGV platform.

In this study, a Painting Robot with a two-camera system was designed. Computer vision techniques and advanced control theory were introduced into this thesis. Three different controllers were developed, including Proportional-Integral (PI) with an anti-windup feature, Model Reference Adaptive Control (MRAC) and the hybrid PI-MRAC. Experiments were conducted to evaluate the performance of the entire system, which indicated the successful design of the Painting Robot.

Acknowledgments

I would like to thank my advisor, Dr. Alexander Leonessa, who dedicated his efforts to helping me build my career. I would also like to thank Dr. Steve Southward for his assistance and helpful suggestions. I would like to thank Dr. Tomonari Furukawa for his guidance of how to be a good researcher. To that end, I appreciate Navneet Singh Nagi who started this project at the beginning and gave me a lot of advice. I am also grateful for Garret Burks, Yingqi Lu, Kaiyuan Peng, Hailin Ren and Tairan Yang, who helped me with the experiments. At last, I am also extremely grateful for the love and support of my friends and family during my graduate study.

Table of Content

CHAPTER 1 INTRODUCTION	1
1.1 MOTIVATION.....	1
1.2 REVIEW OF LITERATURE	3
1.2.1 AGV	3
1.2.2 Computer Vision.....	4
1.2.3 Control System Theory	6
1.3 GOALS	7
1.4 ORGANIZATION OF THESIS.....	7
CHAPTER 2 PAINTING ROBOT OVERVIEW DESIGN	8
2.1 DESIGN OBJECTIVE	8
2.2 MECHANICAL DESIGN.....	9
2.2.1 Design Scheme	9
2.2.2 Selection of Actuators	10
2.2.3 The 3D Model for Paint Mechanism	10
2.2.4 The Design of the Trigger Mechanism	12
2.2.5 Design Verification	12
2.3 ELECTRONICS DESIGN	14
2.3.1 Design Overview	14
2.3.2 Cameras	15
2.3.3 Actuators.....	15

2.3.4 Power System.....	16
2.4 REMARKS.....	17
CHAPTER 3 COMPUTER VISION SYSTEM.....	18
3.1 PROBLEM DEFINITION AND ALGORITHM OVERVIEW	18
3.2 PREPARATION FOR DETECTION	19
3.2.1 Binarization	20
3.2.2 Image Filtering	23
3.2.3 Image Dilation.....	24
3.3 OBJECT DETECTION.....	25
3.3.1 Blob Detection.....	25
3.3.2 Line Detection	27
3.4 REAL WORLD POSITIONING OF THE ROBOT ARM	28
3.4.1 The Position Mapping in the Overview Camera	28
3.4.2 Accuracy Improvement Using a Secondary Camera	30
3.5 SPEEDING UP IMAGE PROCESSING.....	33
3.6 REMARKS.....	36
CHAPTER 4 DESIGN OF THE CONTROL ALGORITHM	37
4.1 PROBLEM DEFINITION	38
4.2 SYSTEM IDENTIFICATION	40
4.3 PI CONTROL WITH ANTI-WINDUP FEATURE.....	42
4.4 MODEL REFERENCE ADAPTIVE CONTROL (MRAC)	45

4.5 PI-MRAC CONTROL.....	48
4.6 REMARKS.....	54
CHAPTER 5 TESTING AND EXPERIMENTS	55
5.1 OVERVIEW OF THE TESTING PLATFORM	55
5.2 LINE TRACKING EXPERIMENTS IN LAB	56
5.3 ON-FIELD LINE PAINTING WITH REFERENCE LINE.....	60
5.4 ON-FIELD LINE PAINTING WITHOUT REFERENCE LINE.....	64
5.5 REMARKS.....	66
CHAPTER 6 CONCLUSION AND FUTURE WORK	67
<i>References.....</i>	<i>69</i>

List of Figures

FIGURE 1.1 ROAD WORKING ZONE FOR PAINT WORK 2

FIGURE 1.2 THE AGV DEVELOPED AT DYSMAC LAB, VIRGINIA TECH..... 4

FIGURE 1.3 A TYPICAL CONTROL SYSTEM BLOCK DIAGRAM..... 6

FIGURE 2.1 COMMERCIAL MARKING SPRAY PAINT 9

FIGURE 2.2 2DOF SCHEME 10

FIGURE 2.3 3D MODEL OF PAINT MECHANISM 11

FIGURE 2.4 THE DESIGN OF THE TRIGGER MECHANISM 12

FIGURE 2.5 AGULAR SPEED FOR WORKING AT AVERAGE RPM..... 13

FIGURE 2.6 ELECTRONIC SYSTEM LAYOUT 14

FIGURE 2.7 THE OVERVIEW OF THE POWER SYSTEM 16

FIGURE 3.1 IMAGE PROCESSING ALGORITHM PSEUDO CODE 19

FIGURE 3.2 RGB AND HSV COLOR SPACE 21

FIGURE 3.3 THE COLOR DETECTION IN HSV COLOR SPACE 22

FIGURE 3.4 THE OUTDOOR ENVIRONMENT IMAGE 23

FIGURE 3.5 THE MEDIAN FILTER 23

FIGURE 3.6 RESULT OF DILATION 24

FIGURE 3.7 THE GEOMETRIC RELATION FOR ROBOT LOCALIZATION 25

FIGURE 3.8 THE RESULT OF BLOB DETECTION..... 26

FIGURE 3.9 LEAST SQUARE LINE FITTING METHOD 28

FIGURE 3.10 THE RELATIONSHIP BETWEEN REAL WORLD AND IMAGE	29
FIGURE 3.11 THE RELATIONSHIP OF THE THREE POSITIONS	30
FIGURE 3.12 THE DEFINITION OF THE OVERVIEW CAMERA SETTING	31
FIGURE 3.13 TWO CAMERA SYSTEM	32
FIGURE 3.14 THE RELATIONSHIP BETWEEN TWO CAMERAS' IMAGE PLANE	33
FIGURE 3.15 THE DETAILED COMPUTER VISION ALGORITHM	34
FIGURE 3.16 CONCURRENT COMPUTER VISION ALGORITHM	35
FIGURE 4.1 SINGLE DEGREE OF FREEDOM SYSTEM	38
FIGURE 4.2 OPEN-LOOP SYSTEM BLOCK DIAGRAM	39
FIGURE 4.3 CLOSE-LOOP SYSTEM BLOCK DIAGRAM.....	40
FIGURE 4.4 H1 ESTIMATE AND INVERSE FREQUENCY RESPONSE.....	41
FIGURE 4.5 COHERENCE OF THE INPUT AND THE OUTPUT SIGNAL.....	41
FIGURE 4.6 BLOCK DIAGRAM OF THE PI CONTROLLED SYSTEM	42
FIGURE 4.7 NON-LINEAR PI WITH GAUSSIAN CONTROL.....	44
FIGURE 4.8 THE TIME RESPONSE OF THE PI WITH ANTI-WINDUP FEATURE AND THE CONSTANT PI.....	44
FIGURE 4.9 THE BLOCK DIAGRAM OF THE MRAC	46
FIGURE 4.10 TIME DOMAIN RESPONSE OF MRAC SYSTEM UNDER SQUARE WAVE	48
FIGURE 4.11 THE BLOCK DIAGRAM OF THE HYBRID PI-MRAC SYSTEM	49
FIGURE 4.12 THE TIME DOMAIN RESPONSE OF THE PI-MARC SYSTEM UNDER THE SQUARE WAVE	54
FIGURE 5.1 3D MODEL OF THE TESTING PLATFORM.....	55
FIGURE 5.2 THE 1 ST GENERATION PAINTING ROBOT	56

FIGURE 5.3 TIME DOMAIN RESPONSE OF PI GAUSSIAN CONTROLLER.....	57
FIGURE 5.4 TIME DOMAIN RESPONSE OF THE MRAC CONTROLLER.....	58
FIGURE 5.5 TIME DOMAIN RESPONSE OF THE PI-MRAC CONTROLLER	59
FIGURE 5.6 THE CONTROL EFFORT IN THE PI-MRAC CONTROLLER.....	60
FIGURE 5.7 THE TWO-CAMERA PAINTING ROBOT	61
FIGURE 5.8 THE SETTING OF THE REFERENCE LINE.....	62
FIGURE 5.9 THE RESULTS OF LINE PAINTING BY PI GAUSSIAN AND PI-MRAC CONTROLLERS.....	62
FIGURE 5.10 RESULTS OF DIFFERENT REFERENCE LINE WITH DIFFERENT WIDTH.....	63
FIGURE 5.11 LINE UPDATING RESULT	63
FIGURE 5.12 DIFFERENT CASES UNDER THE DISTURBANCE	64
FIGURE 5.13 RESULT OF LINE PAINTING WITHOUT THE REFERENCE LINE.....	65

List of Tables

TABLE 2.1 THE PARAMETERS OF COMMON PAINT CAN 9

TABLE 2.2 THE SELECTION OF ACTUATORS 10

TABLE 3.1 BASIC COLORS TABLE 20

Chapter 1 Introduction

Many researchers have dedicated their efforts in developing automatic machines since the 1900s. The attraction of the automatic machine is that it will free the hands of people, save time and increase work efficiency. More and more machines have been invented nowadays since the speed of innovations of technology is exponentially increasing. New technologies like computer vision, artificial intelligence, advanced control theory, etc. will change our opinions about machines. Therefore, the new intelligent machine, which could be defined as a robot, will become more intelligent and more robust. As a result, a world that is feed all around with robots, such as in science fiction books, will become a reality in the future.

1.1 Motivation

Paint still plays a fundamental role in communication nowadays. For example, the paint on the road, called road surface marking, gives brief instructions of traffic rules. It guides the traffic in order and keeps the high efficiency of the entire modern traffic system.

Another example is the paint on the football field or any other sport events field. It gives the limitations of areas to win scores for two participating teams, which makes the competition fair.

Compared to painting (art), paint work is boring and tedious and sometimes dangerous. As mentioned above, road surface marking work would pose a major hazard to the workers who are exposed to moving traffic. According to NHTSA's (National

Highway Traffic Safety Administration) Fatality Analysis, an estimated 35,200 people died in 2015, up from the 32,675 reported fatalities in 2014. There are many reports that indicate around 1-2% of road fatalities occurred at road work zones [1]. Thus, the safety of road workers is a high priority not only for the worker themselves but also for the organizations which employ and represent them.



Figure 1.1 Road Working Zone for Paint Work¹

With the recent improvement in technologies in autonomous ground vehicle (AGV), the idea of automatic Painting Robot comes out. It serves as a subproject to AGVs, a higher level of application based on the AGV platform. The application of automatic Painting Robots is not only to solve the safety problem of highway work zones but also to solve the general painting works in our daily life.

¹ <http://preformedthermoplastic.com/preformed-thermoplastic-road-striping-versus-machine-applied-thermoplastic/>

1.2 Review of Literature

The robot problem could be simply analogous to a decision maker problem. The robot needs to answer three basic questions:

- Where is my target?
- What is my current status?
- How to achieve the target?

The automatic Painting Robot problems could also be casted with the three questions above. Before formally answering all these questions, this section will briefly introduce the techniques involved in the automatic Painting Robot.

1.2.1 AGV

An Autonomous Ground Vehicle (AGV) is a ground vehicle which operates without human intervention. It is capable of making decisions and doing the tasks which people require to be done. The concept of autonomous vehicles began in 1977 with the Tsukuba Mechanical Engineering Lab in Japan [2]. With the recent development of sensors and computer technology, the AGV became a hot topic and many research groups have made contributions to this area within the last decade [3].

DySMAC (Dynamic Systems Modeling and Control) lab at Virginia Tech designed and developed four identical AGVs. DySMAC's AGV could run at speeds up to 10 mph (4.5 m/s) and produce torque up to 322 in-lbs. The AGV is equipped with an on-board computer, 5 GHz radio, and a GPS/INS system [4]. These AGVs have become an efficient scientific and experimental platform. Numerous experiments have been implemented

on these AGVs. For example, a bioinspired tracking control of high speed nonholonomic ground vehicles was developed in 2015 [5] and a radiation search operations using scene understanding was completed in 2016 [4].

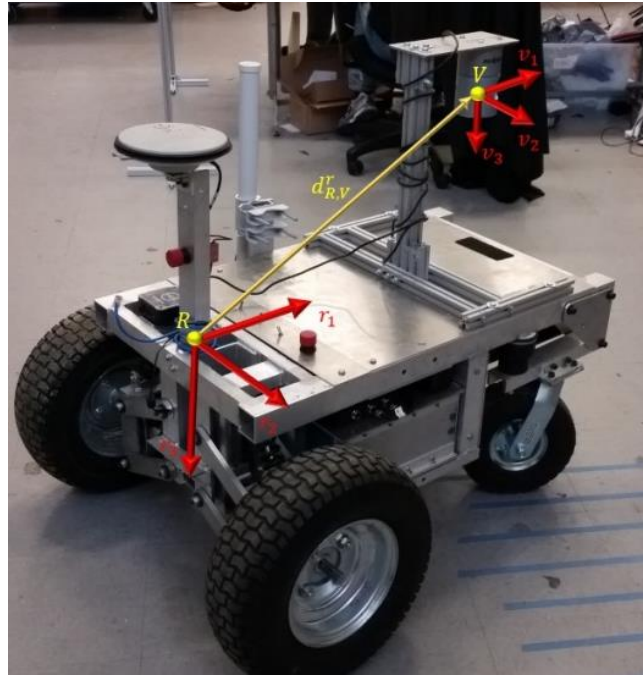


Figure 1.2 The AGV developed at DySMAC lab, Virginia Tech

1.2.2 Computer Vision

Computer vision is widely used in robotics research. It is a humanoid technology to make robots capable of ‘seeing’ the environment. Computer vision is so popular today that it became a common course offered in most universities. Usually, a computer vision system could be divided into three parts:

- Image processing;
- Image analysis;
- Image understanding.

Specifically, computer vision is a technology that lets computers automatically understand images and videos. According to its applications, computer vision system could be utilized to do following things:

- Measurement: computing properties of the 3D world from visual data;
- Perception and interpretation: algorithms and representations to allow a machine to recognize objects, people, scenes and activities;
- Search and organization: algorithms to mine, search, and interact with visual data.

Computer vision technology seems to have a lot of advantages; however, it presents many challenges. For the static environment or simple task, computer vision could solve the problem in an efficient way. For example, the quality monitor of product manufacturing and traffic plate capture. However, the image information usually contains a lot of chaotic contents and it is hard for computers to extract important information from a chaotic background or even to understand the high-level information from the images. Moreover, high level image understanding techniques are computationally expensive for real-time applications.

On the other hand, if the robot is required to do basic identification tasks, then computer vision is a good choice compared to the multi sensors system. Basic image processing requires some common steps:

- Binarization;
- Local feature extraction (filter banks, SIFO, HOG, etc.);

- Feature group and clustering;
- Recognition and understanding.

1.2.3 Control System Theory

Control system theory originated from human observation of the physical world. In 1948, Nobert Wiener defined cybernetics as “the scientific study of control and communication in the animal and the machine” [6]. With the development of computer technology, the new mathematics tools have been introduced into solving control problems in an efficient way. Nowadays, state space methods and frequency analysis dominate the world of control theory. State space representation is a method, which allows to deal with both linear and non-linear systems [7]. The goal of a controller is to ensure the output of a given system (often called the ‘plant’) tracks a desired signal. Advanced mathematics tools are used to study closed-loop properties, such as stability, controllability and observability, etc.

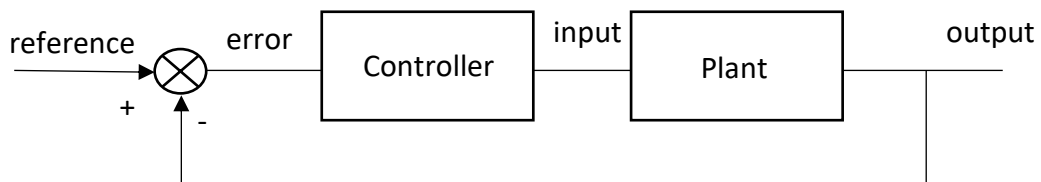


Figure 1.3 A Typical Control System Block Diagram

Based on the type of actuator, the control method could be categorized as:

- Active control;
- Passive control;
- Semi-active control.

This thesis will mainly focus on active control method. PID control is one of the most widely used control methods in industrial areas since it is very simple and effective. The PID controller originated in the 19th century, with the design of a speed governor [8]. However, it was first developed using a theoretical analysis in 1922 by Nicolas Minosky [9]. PID controllers calculate a control input base on proportional, integral, and derivative terms, which is the reason why it was named as PID.

1.3 Goals

Painting Robot is a higher level application based on the AGV platform. The objective of this research is to design, control, and test a good Painting Robot capable to suppress disturbances from the AGV platform. In order to achieve the final goal, this thesis is divided into several parts:

- Painting Robot overview design;
- Target finding solved by computer vision techniques;
- Feedback control design.

1.4 Organization of Thesis

Chapter 2 will discuss the overview design of Painting robot, including the mechanical and electronic parts. Chapter 3 presents the computer vision system algorithms and implementation. Chapter 4 introduces the design of the controller. Chapter 5 describes the testing and experiments. Chapter 6 provides our conclusions and future work.

Chapter 2 Painting Robot Overview Design

This chapter will illustrate the overview design scheme of the Painting Robot. A typical robot system consists of two parts: mechanical and electronic. The following sections will discuss the details of each part step by step.

2.1 Design Objective

Before designing a robot, the very first step is to determine its requirements. What kind of tasks will the robot take? What kind of capabilities should the robot have?

The main task of the Painting Robot is to paint straight lines when it is mounted on a moving platform, while the moving platform is traveling from preselected point A to point B. One would argue that a fixed painter on the moving platform would solve the problem. The reason for this robot design is that when the platform is moving, the fixed painter will be exposed to a very noisy environment. One of the important tasks for a Painting Robot is to stabilize the painting mechanism so that the straightness of a painted line is within an acceptable range. Another reason to make this Painting Robot as a standalone system is to follow the compositional design preference. Every complex robot system could be designed by the component design pattern, which requires one to make sure every submodule works fine before assembling them together.

It can be concluded that the goal of this thesis is to design a standalone Painting Robot to paint a reasonable straight line, while rejecting the disturbance from the moving platform.

2.2 Mechanical Design

2.2.1 Design Scheme

Commercial marking spray paint is used to paint a line. Thus, the design question will be converted to how to design a mechanism to hold a paint can, to adjust the position of the can, and to trigger the spray.

The common commercial marking spray paint is shown in Figure 2.1. The parameters of the paint can are listed in the Table 2.1.



Figure 2.1 Commercial Marking Spray Paint

Table 2.1 The Parameters of Common Paint Can

Item Weight	1.4 pounds
Product Dimensions	9.5×2.6×2.6 in
Weight	18 ounce

Since there is no other special requirements, the paint mechanism is a simple 2 Degree of Freedom system (DOF). One is to trigger the spray and the other is to adjust its position.

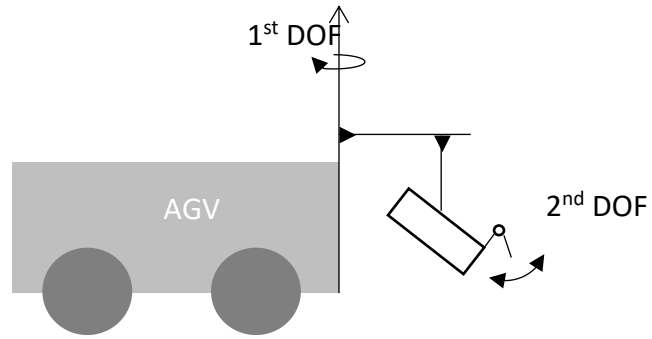


Figure 2.2 2DOF scheme

2.2.2 Selection of Actuators

The selection of actuators for this 2 Degree of Freedom mechanism is in the Table 2.2.

Table 2.2 the Selection of Actuators

DOF	Model	Applied Voltage	Max Drive Torque
1	Dynamixel EX106	14.8 V	5.4 Nm
2	Tower Pro SG92R	5 V	0.25 Nm

2.2.3 The 3D Model for Paint Mechanism

The frame of paint mechanism is constructed by 6061 Aluminum brackets, which are easy to assemble. The 6061 Aluminum brackets already have enough strength to bear the weight under the working conditions. The 3D model of this paint mechanism is shown in Figure 2.3.

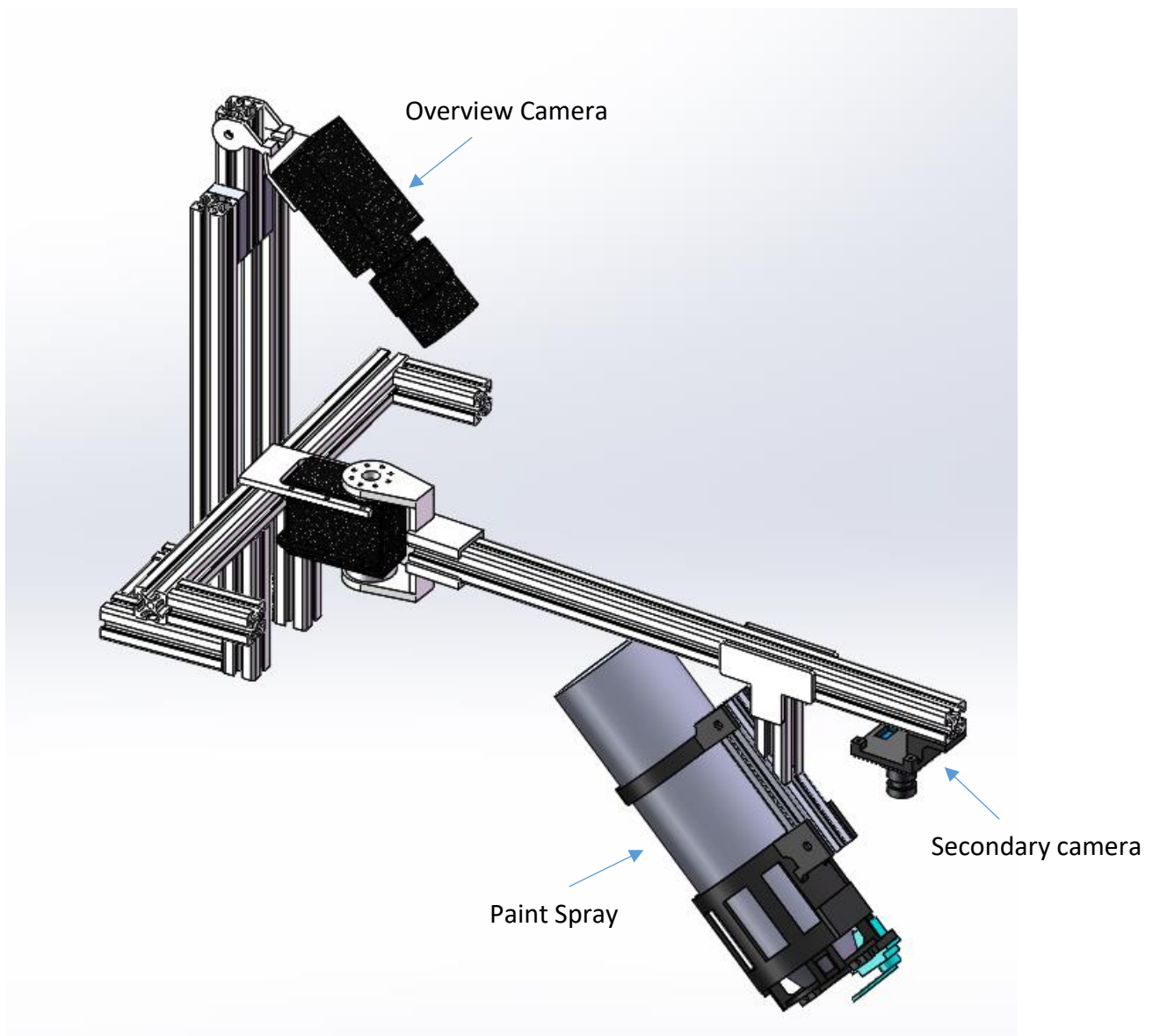


Figure 2.3 3D Model of Paint Mechanism

The setting of the two cameras will influence the design of the computer vision algorithm. The general purpose of the overview camera is to sense the direction of the painted line while the secondary camera helps determining the accurate spray position. The details of the setting of the cameras are discussed in the following chapters.

2.2.4 The Design of the Trigger Mechanism

The design of the trigger mechanism must guarantee that the small servo motor can trigger the spray paint easily. It also makes sure that the replacement of the spray paint is convenient. The design of the holder of the spray paint is shown in Figure 2.4. It takes advantage of the gravity of the spray paint to make sure each replacement of can is at the same position than the previous one. This is important for the design of the control system.

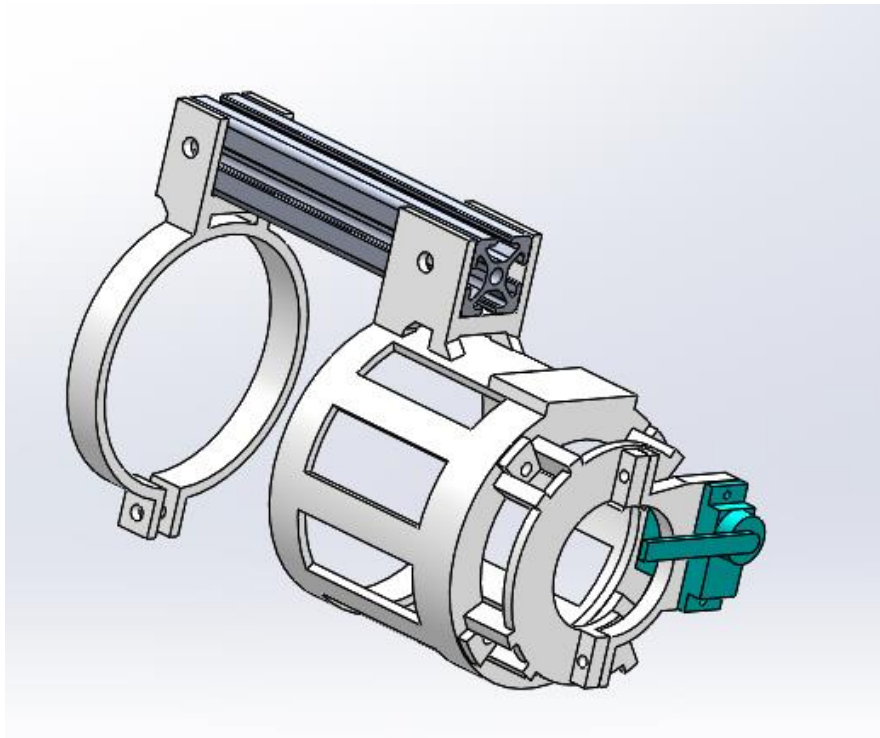


Figure 2.4 The Design of the Trigger Mechanism

2.2.5 Design Verification

The selection of the actuator for the adjustment of the holding arm is straightforward. The servo motor only drives the arm rotation. The maximum angular acceleration of the arm could be calculated by

$$\tau = I\alpha . \tag{1}$$

where I : moment of inertia $I = \frac{1}{2}MR^2$, α : angular acceleration .

The length from the center of rotation to the mass center of the paint spray can is 0.22 m. So, the moment of inertia for this mechanism is 0.0132 Kg*m². Hence, the maximum angular acceleration is 515.15 rad/s² after calculation.

According to the setting of Dynamixel EX106, the average angular speed is 56.8 rpm (5.95 rad/s). Figure 2.5 shows how the angular speed of Dynamixel EX106 changes when it transverses 180 degrees. Thus, the transversal time for 180 degrees is 0.528s. As a result, this typical response time for a robot arm joint is good enough to work at a low frequency disturbance.

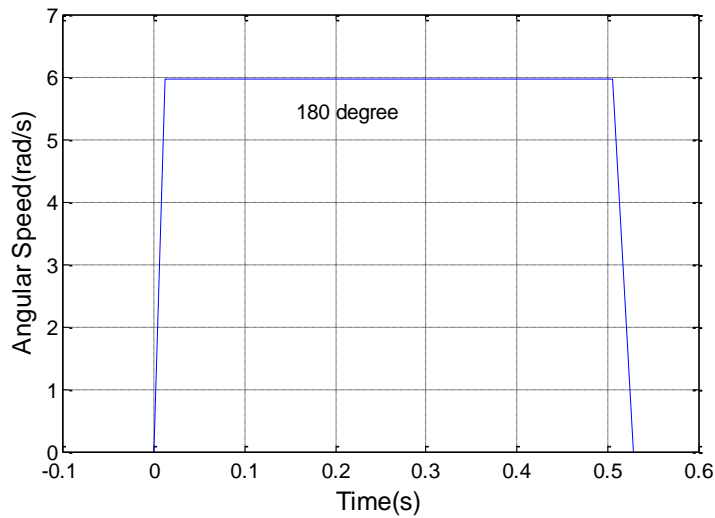


Figure 2.5 Angular speed for working at average RPM

2.3 Electronics Design

2.3.1 Design Overview

The Painting Robot includes two cameras, two actuators (servo motors), and a control unit. The camera is the ‘eye’ of the robot. The robot will utilize the image captured by the camera to figure out its own position and target position. There are two actuators handling the two DOFs of the Painting Robot, which were previously mentioned. The control unit is the ‘brain’ of the robot. It will analyze the image and make the decisions for the robot. The choice of the control unit is a personal computer (PC), which is powerful enough to handle image processing and multi-threads programming. The electronic system layout is illustrated in Figure 2.6.

However, the drawbacks of using a PC are obvious: the CPU cannot run at its full capacity, and it does not have accurate timing for hardware control. These issues will be discussed in the following chapters.

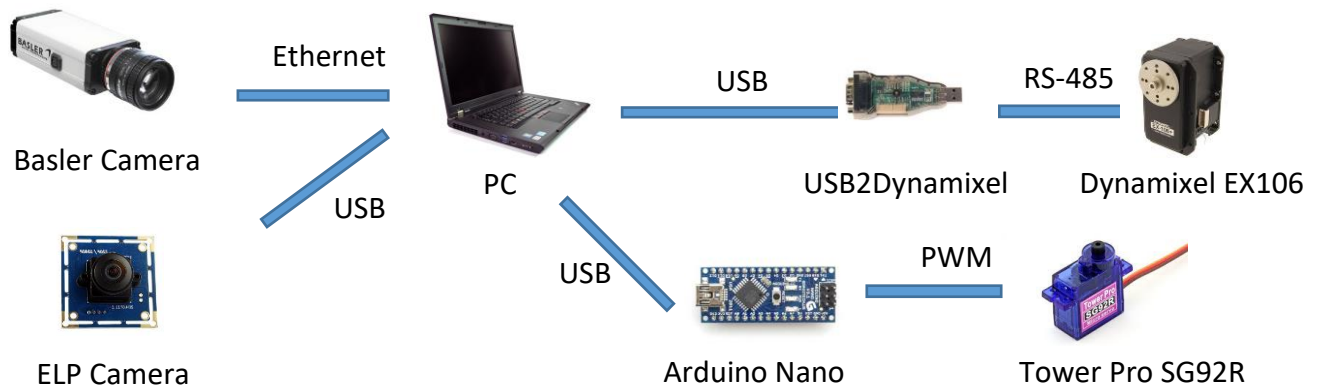


Figure 2.6 Electronic System Layout

2.3.2 Cameras

Two cameras are introduced into the system, which are the main sensors of the Painting Robot system.

The first camera is the Basler Scout scA640-70gc area scan camera. It provides the overview of the robot arm and previously painted line. It will be called overview camera in the following. The Basler camera can provide 658×492 pixels with a maximum frame rate of 70 fps. The communication interface is GigE, which offers high speed data transfer. The Basler Company provides the C++ Camera Software Suite SDK and it is compatible with the OpenCV library.

The second camera is ELP 2 megapixel HD Free USB camera. It is used for the accurate control of the spray position. It can achieve 1280×720 at 60 fps or 1920×1080 at 30 fps. It will be called secondary camera in the following. This camera is programmable and can be easily operated using OpenCV functions.

2.3.3 Actuators

The Painting Robot has two actuators, one is Dynamixel EX106 and the other is Tower Pro SG92R.

The Dynamixel EX106 is a high-performance servo motor, which is typically used as a robot joint. It can provide large torque and high rotation speed. There is an embedded system inside Dynamixel EX106. Thus, the higher level control unit, like PC in our case, only takes care of the serial communication part. The Dynamixel EX106 provides C++ version API, which is used in our system.

The Tower Pro SG92R is a micro servo motor, which is directly powered by 5V and controlled by Pulse-Width Modulation (PWM). However, a PC cannot directly provide a PWM wave. Hence, a microcontroller is introduced into the system. We selected an Arduino Nano because of its convenience.

2.3.4 Power System

There are three operating voltages in this Painting Robot. The Dynamixel EX106 servo motor operates at 14.8V, the Basler camera uses 12V and the ELP camera, Arduino Nano, Tower pro SG92R need 5V. The 5V is supplied by the onboard USB port and the PC is powered by its own battery. Thus, we selected a 14.8V LiPo battery with a 10 Ah and a maximum output current of 7A. It can power the Painting Robot for hours. A DC-DC converter is used to regulate the 14.8 V to 12 V, which is to power the Basler camera. The overview of the power system is like Figure 2.7.

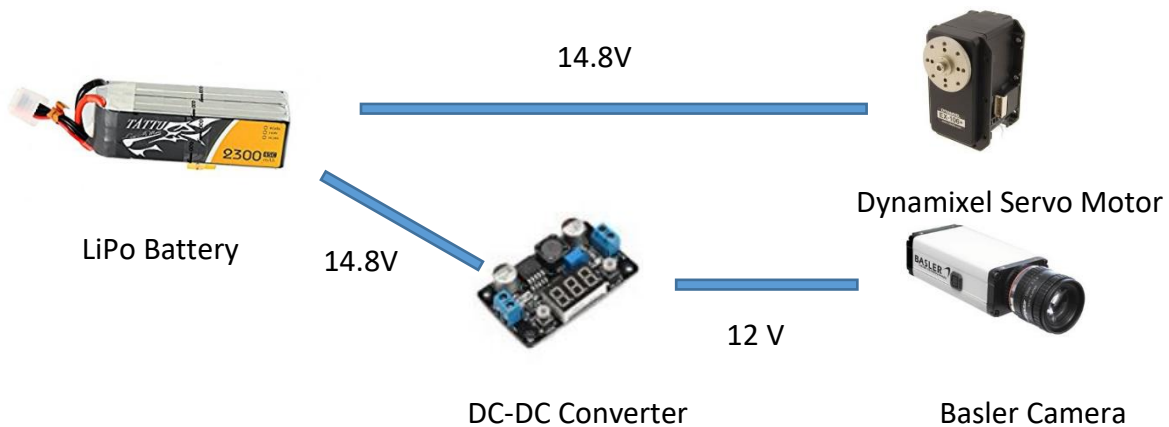


Figure 2.7 The Overview of the Power System

2.4 Remarks

Chapter 2 introduced the overview of the Painting Robot, including the mechanical and electronic components. Section 2.1 converted the real-life problem into engineering language. The following design procedure is reasonable based on the problem definition. Finally, the Painting Robot was designed to fulfill the need of a high-level application consisting of painting straight lines while rejecting disturbances introduced by the moving platform.

Chapter 3 Computer Vision System

Computer vision techniques are applied to solve the proposed problem. The robot analyzes the images which are captured by the camera to make decisions. The advantages of the computer vision system are:

- Easy to implement (two cameras connected to the PC)
- Contains enough information for robot to make decisions

However, the drawbacks are also obvious. For example, it is not easy for higher level perception. The current algorithms are well defined to calculate local features in images, but the perception of images is still a challenging task. Fortunately, the computer vision problem of this robot is at low level recognition of image objects. This chapter will introduce the algorithm to solve the computer vision challenges for the Painting Robot.

3.1 Problem Definition and Algorithm Overview

The target of the Painting Robot is to paint a straight line on the ground. Thus, the problems for the computer vision system to solve are:

- Determining the position of the robot arm;
- Determining the target position for the robot arm to go.

The assumptions for this problem are:

- The image contains both the robot arm and target tracking line;
- There exists a unique pattern for robot to detect

In order to simplify the computer vision problem, two red round markers are placed on the robot arm since one line is determined by two points in the 2D plane. Also, it requires that the color of the line has large contrast with respect to the color of the ground. Thus, the computer vision problem of the Painting Robot is summarized as detecting two markers and a painted line in a single image.

As a result, Figure 3.1 shows the image processing algorithm pseudo code. The algorithm will detect two markers and the target line after the image captured. The target position will be determined after successful detection of the markers and the line.

```
while(true)
    capture_image
    detection_two_markers
    detection_line
    calculate_the_error
    set_target_position
```

Figure 3.1 Image Processing Algorithm Pseudo Code

3.2 Preparation for Detection

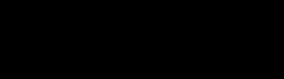




It is important to complete some pre-calculations before applying detection methods. The raw color image usually has three channel values for each pixel, e.g. RGB, HSV. The binarization is one of the most important methods, which extracts the key features from the raw image and will increase the computing speed of the rest of the process.

3.2.1 Binarization

Binarization is highly based on the computer vision problem being solved. For this Painting Robot, the high contrast color is used to separate the objects waiting to be detected, which is also the method that can simplify the computer vision problem. Thus, the raw image will be converted into a binary image based on the color thresholding.

For binarization based on the color, a color space is selected to do the thresholding. The most common color space is RGB. The philosophy behind RGB is that every color can be generated by three basic colors: Red, Green and Blue. The Table 3.1 shows how the basic colors are represented by RGB.

Table 3.1 Basic Colors Table

Color	Name	Hex Code	Decimal Code
	Black	#000000	(0,0,0)
	White	#FFFFFF	(255,255,255)
	Red	#FF0000	(255,0,0)
	Lime	#00FF00	(0,255,0)
	Blue	#0000FF	(0,0,255)

Although the raw data is in the RGB color space, the image is converted into the HSV (hue, saturation and value) color space. The reason that HSV is selected rather than RGB is because RGB value of one certain color varies according to the different environmental factors like luminance and shadows. The HSV representation usually focuses more on the abstract meaning of the color. It means you don't need to know

what percentage of blue or green is required to produce a color [10]. The hue represents the main color, the saturation represents the distance to the vertical axis and the value represents the height in Figure 3.2.

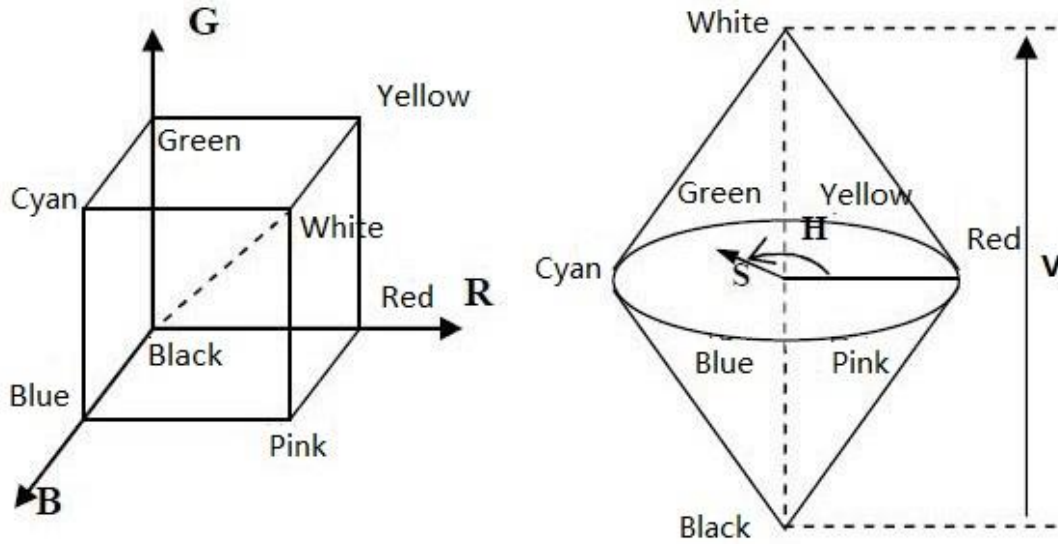


Figure 3.2 RGB and HSV color space

According to OpenCV documentation, RGB can be converted to HSV as follows.

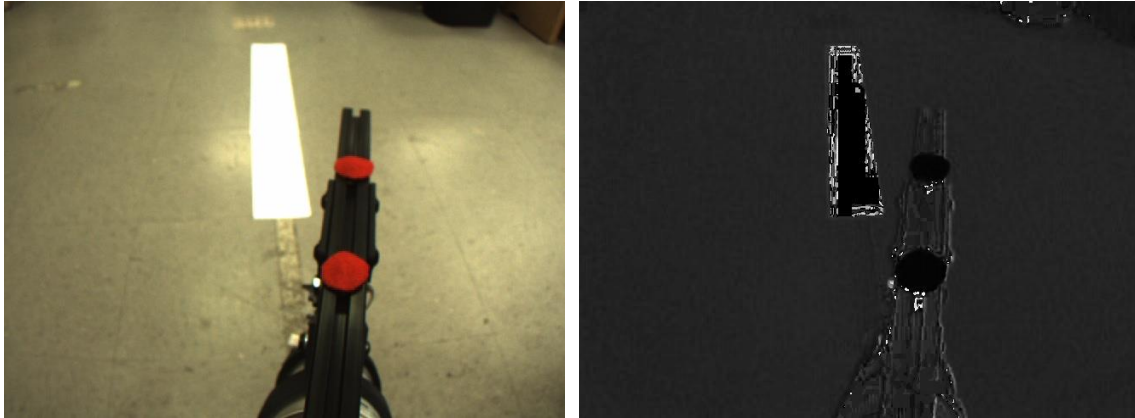
$$V = \max(R, G, B),$$

$$S = \begin{cases} \frac{V - \min(R, G, B)}{V} & \text{if } V \neq 0, \\ 0 & \text{otherwise,} \end{cases} \quad (2)$$

$$H = \begin{cases} 60(G - B)/(V - \min(R, G, B)) & \text{if } V = R, \\ 120 + 60(B - R)/(V - \min(R, G, B)) & \text{if } V = G, \\ 240 + 60(R - B)/(V - \min(R, G, B)) & \text{if } V = B. \end{cases}$$

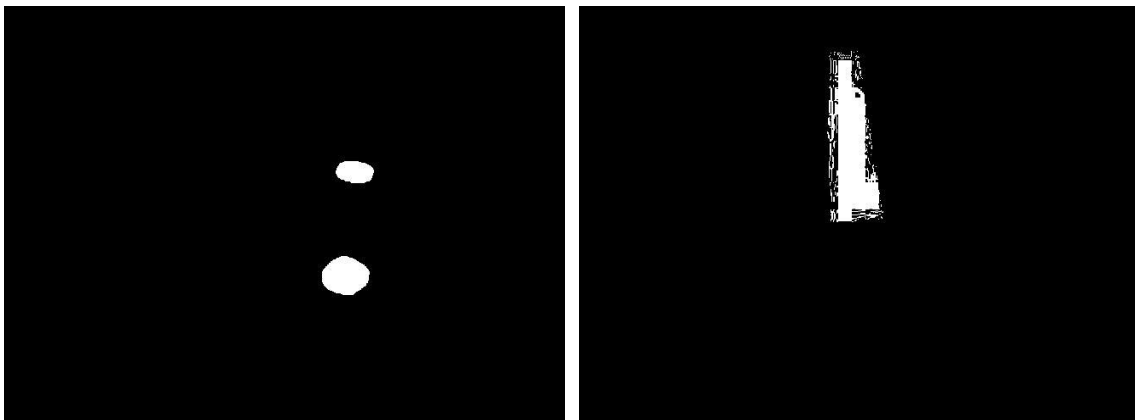
When displaying only hue values, the image is shown in Figure 3.3(b). Similar colors are converted into a similar hue value. Thus, the nice result shown in Figure 3.3(c)(d) are obtained by thresholding on the well-known colors. However, when the images are taken on-field instead of the laboratory environment, we obtain a roughly

good binary image. The Figure 3.4 illustrates that color thresholding provides only a partial line because of the rough surface of the grass field.



(a) Origin Image

(b) Hue Value



(c) Thresholding based on Red

(d) Thresholding based on White

Figure 3.3 The Color Detection in HSV Color Space



Figure 3.4 The Outdoor Environment Image

3.2.2 Image Filtering

Image filters are used to filter out the high frequency noise, such as random noise, or salt and pepper noise. In this case, a Median filter is applied to remove the noise, or the high intensity, small regions which are randomly scattered in the binary image [11]. The Median Filter is a non-linear filter and for small filter sizes, this filter can preserve main objects. The binary image processed after applying this filter is shown in Figure 3.5.

Figure 3.5 clearly shows that the random noise has been removed and the main objects remained.



Figure 3.5 the Median Filter

3.2.3 Image Dilation

The purpose of dilation is to try to connect the line shown in Figure 3.5. However, the drawback of dilation is that the noise is also amplified. Even though the median filter tries its best to remove the noise, there is still noise or unexpected items left in the binary image.

The white line in the binary image after thresholding is not displayed as a connected line, which is because it is captured in the outdoor environment. The reason behind this rough result is because the surface of the grassland is not smooth. As a result, the shadow has a huge effect on the line detection. The image dilation will help to fill the holes where it is supposed to be connected. The simple algorithm for dilation is that if the current pixel is foreground, then take the *OR* operation between the structuring element and the input image, where the structuring element always takes surrounding pixels into consideration. In other words, if the neighborhood of the current pixel is foreground, we could assume that the current pixel is also foreground. Figure 3.6 shows the result image after dilation.

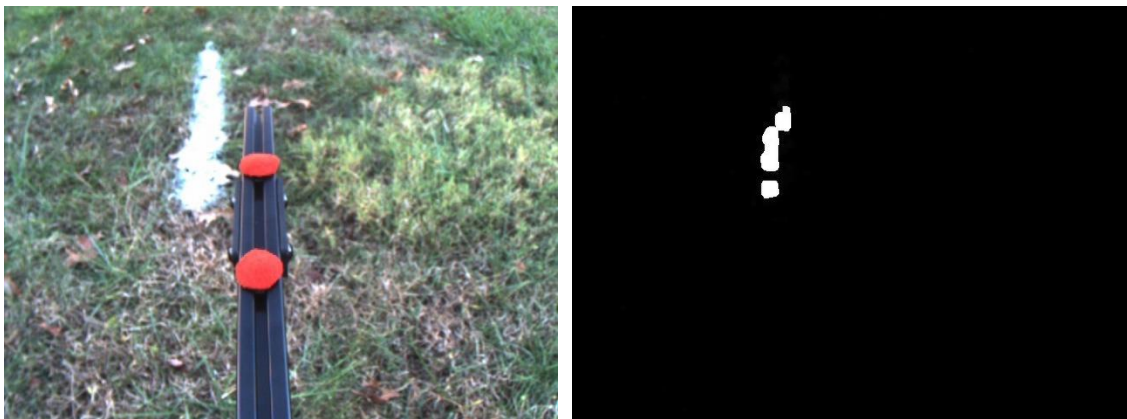


Figure 3.6 Result of Dilation

3.3 Object Detection

3.3.1 Blob Detection

In order to position the robotic arm as needed, we need to determine the angle between the arm itself and the center axis. The first step is to detect two red round markers placed on the arm, which will determine one line in a 2D plane. Based on the setting of the camera and the pinhole camera model, we assume that the plane where the robot arm moves in the real world is linear transformed onto the camera image plane. In Figure 3.7, the point O is the origin of the robot arm rotation, which also overlaps the center point of the camera. The Plane A will be projected to the camera image plane, which looks like in Figure 3.7 (left). The arm will also be projected onto the image plane. Under the assumption that the geometric properties are maintained by camera projection, the robot localization problem is summarized as finding the angle between the center axis and the line determined by two red markers.

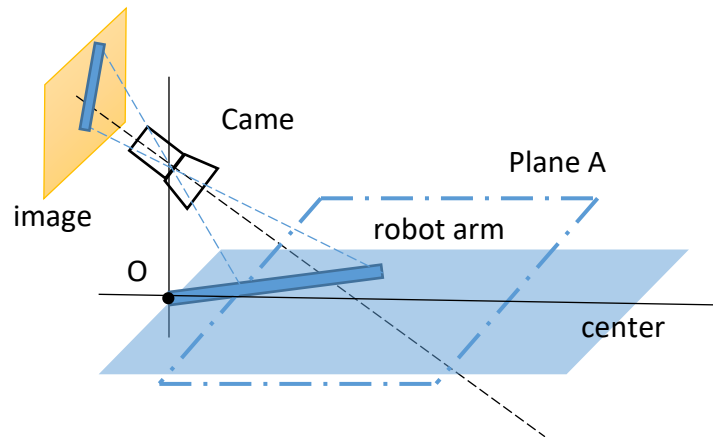
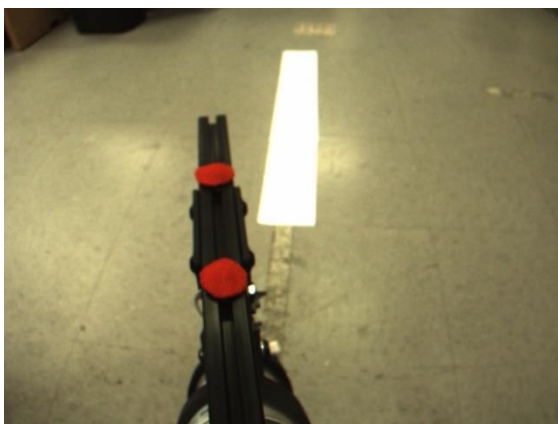


Figure 3.7 The Geometric Relation for Robot Localization

The detection of two red round markers can be cast as a blob detection problem. The common solution to solve such a problem is to use the Laplacian of the Gaussian (LoG).

Given an image $f(x, y)$, first we need to compute the convolution

$$L(x, y, t) = g(x, y, t) * f(x, y) \quad (3)$$

with a Gaussian kernel at a certain scale t , given by

$$g(x, y, t) = \frac{1}{2\pi t^2} e^{-\frac{x^2+y^2}{2t^2}}. \quad (4)$$

Then, the Laplacian operator is computed as follows

$$\nabla^2 L = L_{xx} + L_{yy}. \quad (5)$$

This operator will show strong positive responses for dark blobs of radius $r = t\sqrt{2}$ [12]. As a result, the center of two red markers is easily determined by the blob detector. In Figure 3.8, the position of the robot arm can be found by this method.

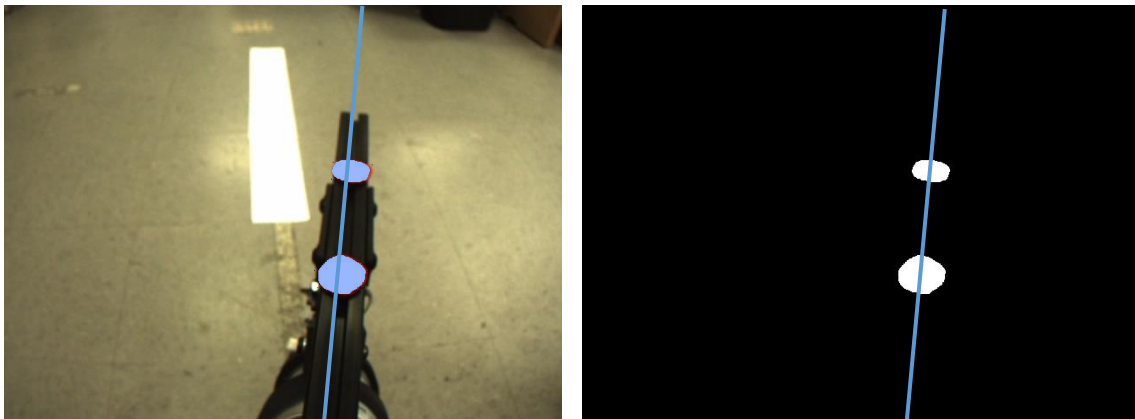


Figure 3.8 The Result of Blob Detection

3.3.2 Line Detection

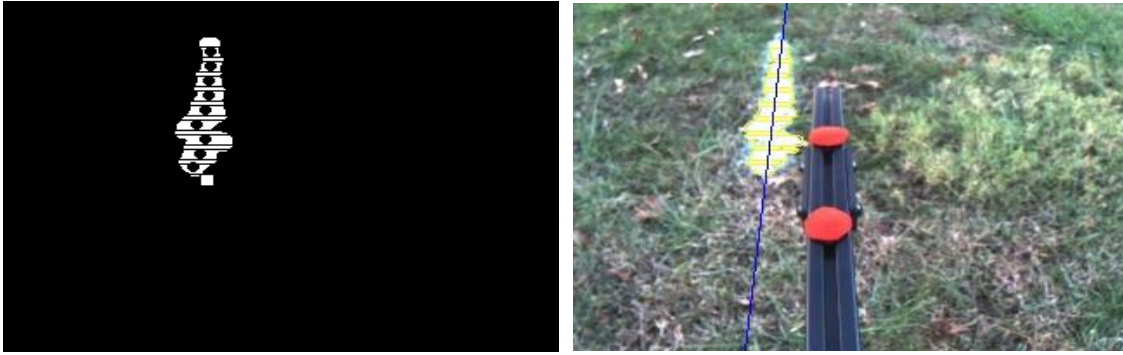
The problem of finding the arm position is still to be solved; however, our field results indicate that there is no nice edge for most edge detectors, such as the Canny Edge Detector [13]. Hence, the problem is simplified to determine the best fit line through a set of points, under the assumption that this set of points is related to the line. In our case, the least square line fitting method is applied. The least squares line fitting method is to minimize the sum of vertical distance between the points to the proposed line. Under our assumption, the points in the set belong to the painted line even though it does not have a well-defined edge. Therefore, the fitted line found by least square method is exactly the previous painted line. To reduce the computational complexity, the proposed algorithm divides the image into slices after image dilation. Each slice of the image is used to detect local maximum object and find its mass center. The collection of the mass centers is used to find the least square fitting line. With this method, the number of points used for line fitting has an upper bound, which guarantees a bounded processing time. Figure 3.9 shows the results obtained from the least square line fitting method.



(a) origin image



(b) image dilation



(c) points set

(d) detection result

Figure 3.9 Least Square Line Fitting Method

3.4 Real World Positioning of the Robot Arm

3.4.1 The Position Mapping in the Overview Camera

The positions of the target line and the robot arm are detected using the method previously described. However, these positions are in the image plane. As mentioned before, the pinhole camera model is applied to retrieve the real-world position from the image. The biggest challenge for mapping image points to real-world points is that the projection does not preserve length or angles. The pinhole camera model is described in equation (6) [14]. The intrinsic matrix is constant because it is related to the properties of the camera itself. The extrinsic matrix consists of the rotation and transformation matrices. The problem to be solve is to find which plane in the real world is projected onto the image plane. The real-world position x_W is in \mathbb{R}^3 , however, the image position x_I is in \mathbb{R}^2 . The pinhole camera model equation reduces the dimension the real-world position. So, retrieving the real-world position from the image is difficult. The missing depth makes it hard to reverse the transformation.

$$x_I = K[R \ T]x_W, \quad (6)$$

where K is the Intrinsic Matrix, R is Rotation Matrix, T is Transformation Matrix, x_I is Image Position, x_W is Real World Position.

Since the overview camera is fixed on the robot frame, there is a constant angle between the optical center of the camera and the plane of the robot arm. Moreover, the ground plane is always parallel to the plane of the robot arm. In Figure 3.10, the robot arm Plane A is parallel to the ground Plane B. The image captured by the camera will see the Plane A' and Plane B'. According to the pinhole camera model, the relationship between the Plane A and Plane A' is linear.

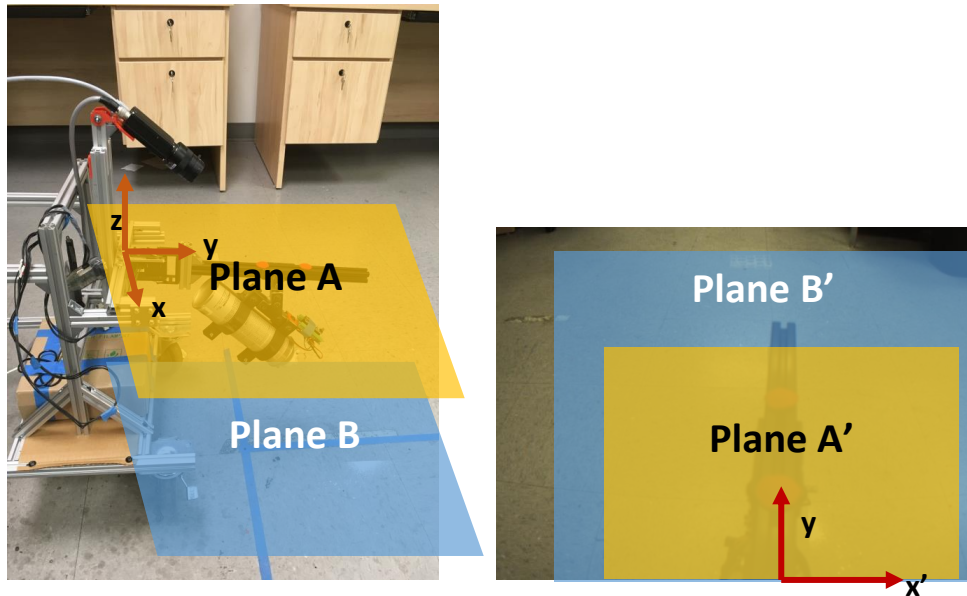


Figure 3.10 The Relationship between Real World and Image

We are more interested with the relative error between the robot arm and the target line. Thus, there is no need to do the 3D reconstruction. The more important thing is to find out the relationship between the arm and spray point positions. Figure 3.11 shows the relationship of the arm position and the spray position.

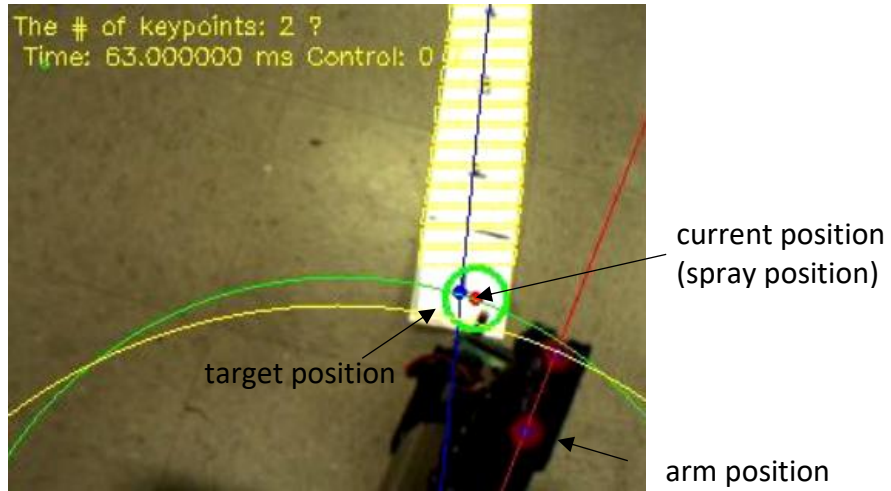


Figure 3.11 The Relationship of the Three Positions

The arm position is determined by the two-red round marker detection, which travels on the yellow circle in Figure 3.11. As mentioned above, there is a linear transformation between the spray position and the arm position. Therefore, the spray position travels following the green circle in Figure 3.11. The red dot is calculated using the arm position, which represents the spray position. The spray position is defined as the current position. The target position is the intersection point of the detected painted line and green circle. Thus, the error is defined as the error between x coordinate of the target position and the current position

$$error = x_{current} - x_{target}. \quad (7)$$

3.4.2 Accuracy Improvement Using a Secondary Camera

Although the target position in the overview camera is calculated based on a mathematical model, high accuracy is required for the Painting Robot. The line painting task is different from the line tracking problem. The line tracking problem is to detect the existing line; however, the line painting problem is to paint the new line while

continuously detecting the previous painted line. Thus, any error while painting the line will be detected and used in painting the next section, hence causing an error accumulation. Even though the mapping between the arm position and spray position is found as described in Section 3.4.1, the overview camera cannot detect a long enough section of the previous painted line to avoid the accumulation of errors.

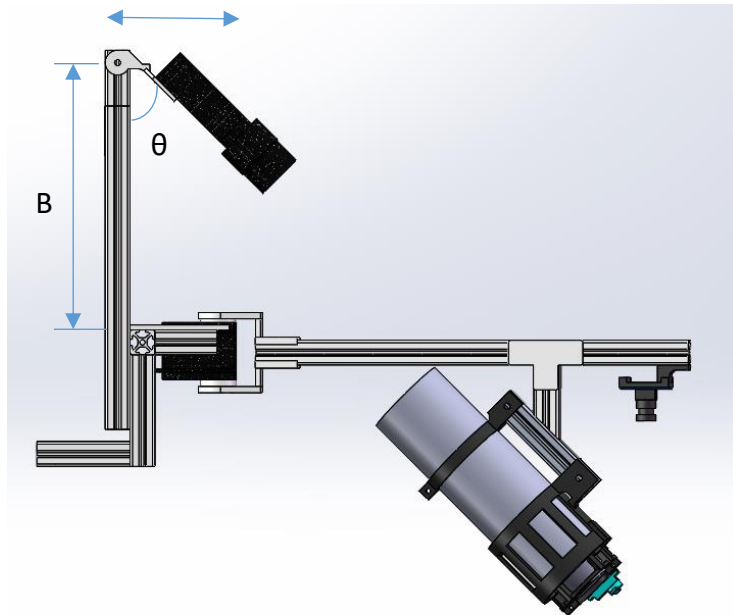


Figure 3.12 The Definition of the Overview Camera Setting

The setting of the camera will influence the visibility of the spray position. In Figure 3.12, the height of the overview camera is B, the horizontal length between the camera and the arm rotation center is A and the pitch angle of the camera is θ . In order to directly detect the spray position, the angle θ needs to be small. However, a larger θ is required to detect longer segments of the previous painted line. Another contradicting fact is that a longer A creates a smaller working area but can detect more

of the previous painted line. Thus, the single camera cannot meet all the requirements at the same time.

Consequently, a secondary camera is introduced into the design of the Painting Robot, placed at the end of the robot arm. The secondary camera moves together with the arm. Thus, the spray position in the image is fixed, which is marked as green circle in Figure 3.13 (b). In Figure 3.13, the overview camera is to sense the direction of the previous painted line and the secondary camera is to find the accurate spray position.

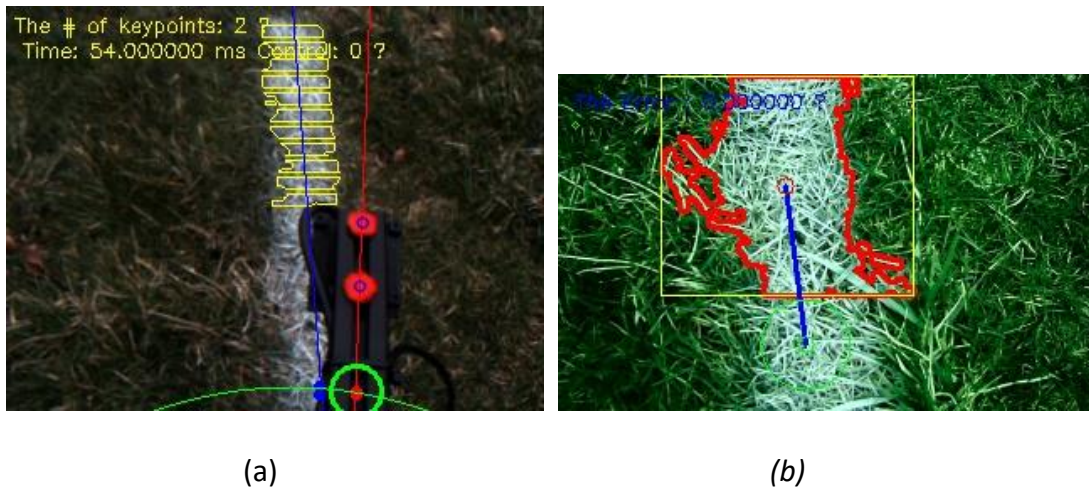


Figure 3.13 Two Camera System

The combination of the two cameras will ensure the correctness of the spray position. The arm position determines the rotation angle between two cameras' image planes. The angle α of the arm position in the overview camera image plane is the same rotation angle of the coordinate reference of the secondary camera. Thus, the painted line detected in the overview image is rotated by α in the secondary camera image plane, which is shown in Figure 3.13.

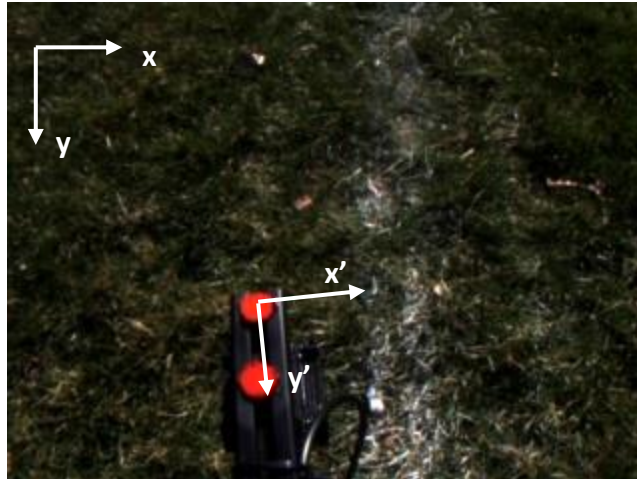


Figure 3.14 The Relationship between Two Cameras' Image Plane

3.5 Speeding up Image Processing

The simple sequential programming will not meet the real-time requirement of the robot control. The processing time of a single image is around 200 ms by sequential execution; however, the desirable speed of control signal should be around 50 ms so that smooth control behavior is achieved. Thus, there is a huge delay for each control signal, which makes it hard to design the control algorithm.

The Amdahl's Law defines the speedup S of a job to be the ratio between the time it takes one processor to complete the job versus the time it takes n concurrent processors to complete the same job [15]

$$S = \frac{1}{1 - p + \frac{p}{n}} \quad (8)$$

where p is the fraction of the job that can be executed in parallel and n is the number of processors.

Thus, if there are 10 threads trying to achieve speedup 4 (200ms/50ms), then it makes a concurrent execution fraction at $p = 5/6$. In other words, the majority part of the computer vision algorithm should be converted into a concurrent execution program.

```
while(true)
|
|   img_RGB = capture_image
|   img_HSV = RGBtoHSV(img_RGB)
|
|   detection two markers
|   img_binaryForRed = Thresholding(img_HSV)
|   red_positions = BlobDetection(img_binaryForRed)
|
|   detection_line
|   img_binaryForWhite = Thresholding(img_HSV)
|   img_clean = MedianFilter(img_binaryForWhite)
|   img_fill = Dilation(img_clean)
|   points_set = PartialContour(img_fill)
|   line = LeastSqaureFit(points_set)
|
end while
```

Figure 3.15 The Detailed Computer Vision Algorithm

The naive way is to create multithreads, then run the algorithm; however, it would not work as the functions called by computer vision algorithm are not always independent from each other. One simple approach is that all the methods work on a single image, whose abstract data structure is a matrix. The details of the methods from this computer vision algorithm is shown in Figure 3.15.

The *detection_two_markers* and *detection_line* are independent from each other because they are calculated based on different color threshold values from the original HSV image. The *RGBtoHSV*, *Thresholding* and *PartialContour* methods can be calculated

concurrently because they calculate the value based on the information of pixel and each pixel calculation is independent to each other; however, the *MedianFilter*, *Dilation* and *BlobDetection* methods are based on the convolution operation, which cannot be simply calculated by each pixel. Therefore, the concurrent program for a single camera will be designed as Figure 3.16.

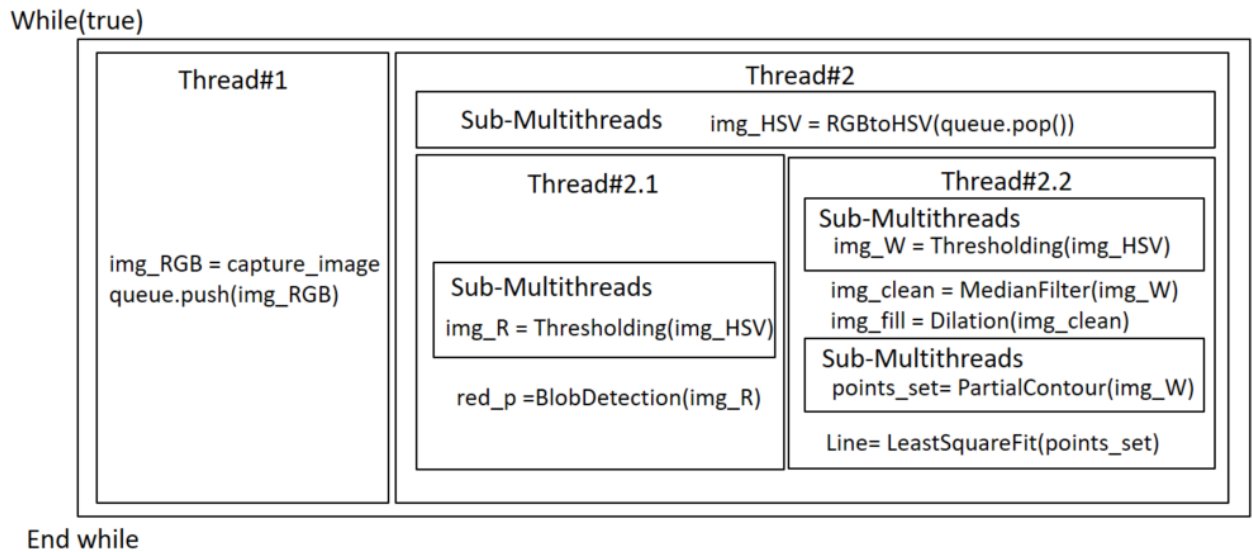


Figure 3.16 Concurrent Computer Vision Algorithm

Since some methods called are based on the results of previously invoked methods, the overall algorithm is partially designed concurrently. In Figure 3.16, the Thread#1 and Thread#2 are parallel in the while loop. The captured image will be stored in a small size queue, and Thread#2 could get latest original image through the queue. Then, the *detection_two_markers* and *detection_line* will require the original image to be converted into HSV. Hence, it takes a high number of threads to do the conversion. After that, inside Thread#2, there are other two threads: Thread#2.1 and Thread#2.2 working on the object detection. Finally, the other multithreads calculation is applied to

Thresholding and *PartialContour*. Also, each camera will execute concurrent versions of the program separately. Thus, the overall image processing time will not increase even if there are multiple cameras.

3.6 Remarks

This chapter introduced the computer vision algorithm applied to the Painting Robot. The raw color image is converted into binary image in the HSV color space based on typical color thresholding. Other image processing techniques, like median filter and image dilation are applied to the binary image before executing the object detection methods. The object detection problem is divided into two categories: blob detection and line detection. Then, common methods are applied to solve these two sub-problems. In order to meet the real-time control requirement of the Painting Robot, the sequential computer vision algorithm is re-designed into a concurrent program.

Chapter 4 Design of the Control Algorithm

Control engineering is a hybrid subject that consists of engineering and mathematics. In the control theory, every system is modeled using a mathematic representation, which reflects the dynamics of the system and the interaction between the environment and the system itself. The control system could be divided into two categories: open-loop and closed-loop [16]. The most common example of open-loop system is the central heating boiler controlled only by a timer [17]. This heating boiler is an open-loop system because the action of switching on or off is based on time without additional measurements. A closed-loop system introduces the feedback of measurements into an open-loop system to achieve control goal. Many standard controllers have been developed by researchers in classical control theory, like the lead-lag compensator and the PID controller [9].

With the development of computer techniques and desire of dealing with high complex system, modern control theory has developed booming during the last few decades. Classical control theory, however, is still efficient when dealing with a linear time-invariant single-input single-output system [18].

The real world is complex and full of uncertainty. Adaptive control techniques can be introduced to make the controller adapt to systems with uncertain parameters. We will discuss the classical PI controller design, Model Reference Adaptive Control (MRAC), as well as the PI-MRAC hybrid controller.

4.1 Problem Definition

The Painting Robot has two degrees of freedom; however, one of the degrees of freedom is related to triggering the paint spray. Thus, there is only one degree of freedom that need to be controlled for positioning the spray.

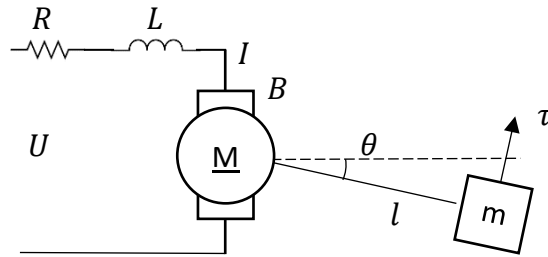


Figure 4.1 Single Degree of Freedom System

This single degree of freedom control problem is shown in Figure 4.1. The mass of spray paint $m(t)$ is continuously but slowly changing during the operation. The terms R and L are the resistance and inductance of the servo motor, respectively. B is the mechanical damping in the motor and l is the current in the circuit. The servo motor provides the torque needed to drive the spray paint to the desired position. Thus, the equation of motion for the open-loop system is given by

$$m(t)l^2\ddot{\theta}(t) + B\dot{\theta}(t) = K_f I(t), \quad (9)$$

$$L\dot{I}(t) + RI(t) + K_e\dot{\theta}(t) = U(t).$$

Then the transfer function relating the input voltage $U(s)$, to the output angle $\theta(s)$ is given by [19]

$$G_1(s) = \frac{\theta(s)}{U(s)} = \frac{K_f}{ml^2Ls^3 + (ml^2R + LB)s^2 + (BR + K_fK_e)s} \quad (10)$$

The Dynamixel servo motor has its internal controller to implement position control. Instead of changing the voltage U , users send a global position command θ_r to control the position θ of the servo motor using a serial communication interface. The open-loop system block diagram is shown in Figure 4.2.

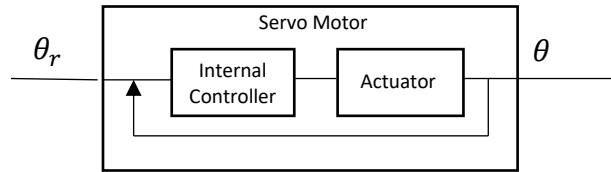


Figure 4.2 Open-Loop System Block Diagram

Under the assumption that the Dynamixel servo motor is in position mode [20], the transfer function $G_2(s)$ between θ_r and U is also given by

$$G_2(s) = \frac{U(s)}{\theta_r(s)} = \frac{ml^2s^2 + Bs}{K_a}, \quad (11)$$

where K_a is proportionality factor between voltage and torque. Thus, the open-loop system transfer function is given by

$$\begin{aligned} G(s) &= \frac{\theta(s)}{\theta_r(s)} = G_1(s)G_2(s) \\ &= \frac{\frac{K_f}{K_a}(ml^2s + B)}{ml^2Ls^2 + (ml^2R + LB)s + BR + K_fK_e'} \end{aligned} \quad (12)$$

where the input is a global position command θ_r and the output is current position θ .

Therefore, the closed-loop system block diagram for the control problem is shown in Figure 4.3. According to the setting of cameras, current position x and desired position r are in a relative coordinate system.

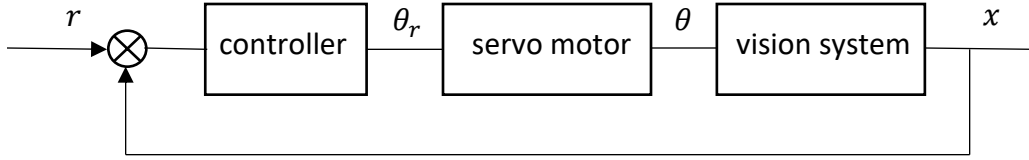


Figure 4.3 Close-Loop System Block Diagram

Thus, the goal of control problem in this thesis is to design an appropriate controller to minimize the error between current position x and desired position r .

4.2 System Identification

The Dynamixel servo motor has internal controller to perform a position control, whose model is unknown. However, it will be treated as a single-input single-output (SISO) system in this thesis. In order to obtain a better controller design, the servo motor model is estimated using system identification techniques. Since the weight of paint spray is continuously changing during the operation, the model is identified when servo motor is without any load.

In this thesis, a white-noise signal is applied to perform system identification. Then, H1 estimation is found by cross-spectrum between input and output and the auto-spectrum of the input signal [21]

$$H_1(mF) = \left(\frac{S_{uy}(mF)}{S_{uu}(mF)} \right). \quad (13)$$

The estimated frequency response and inversed frequency response is shown in Figure 4.4. Thus, the estimated system model for the open-loop system of robotic arm is given by

$$G(s) = \frac{5.327s - 284.1}{s^2 + 33.15s + 322.3} \quad (14)$$

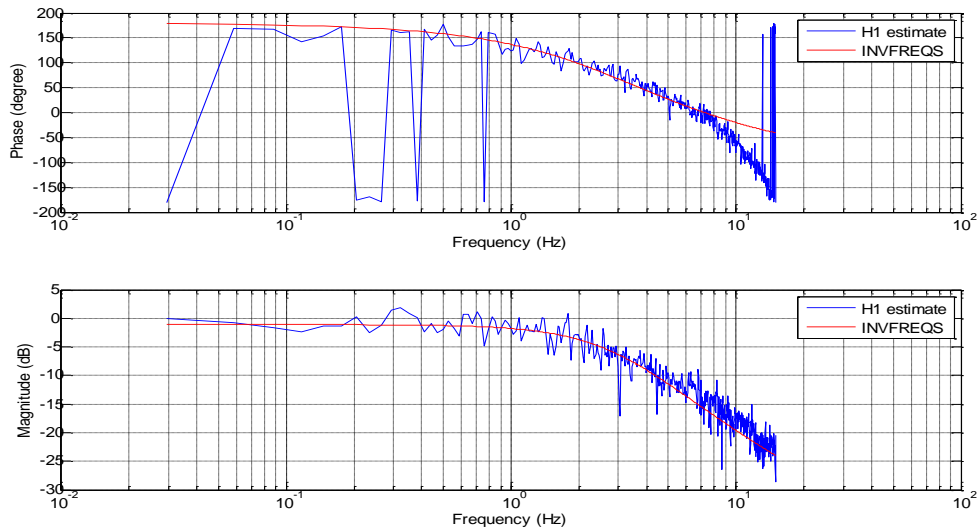


Figure 4.4 H1 Estimate and Inverse Frequency Response

The coherence of the input and the output signal in Figure 4.5 shows a reasonable confidence of the inverse frequency response. The average of the coherence is above 0.7, which indicates the high confidence of the identified model.

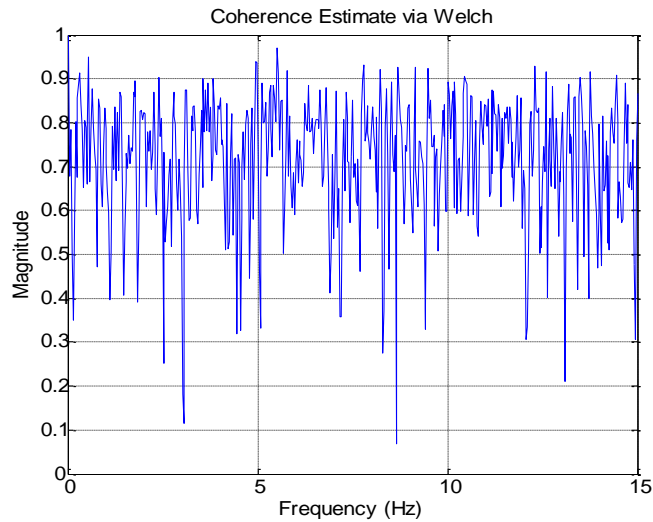


Figure 4.5 Coherence of the input and the output signal

4.3 PI Control with Anti-Windup Feature

PI control refers to Proportional-Integral control. It is a standard controller discussed in classical control theory. It focuses on the error $e(t)$ between the current measurement and the desired output; then applies a control action based on proportional, integral terms

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau. \quad (15)$$

The block diagram of the PI controlled system is shown in Figure 4.6. The plant of servo motor is provided in previous section. The vision system is the measurement procedure, which is treated as a pure delay. The measured x coordinate of the current position x in the image plane is used to calculate the error between the target position and the current position in the image plane. Then the control variable u is determined using the PI algorithm.

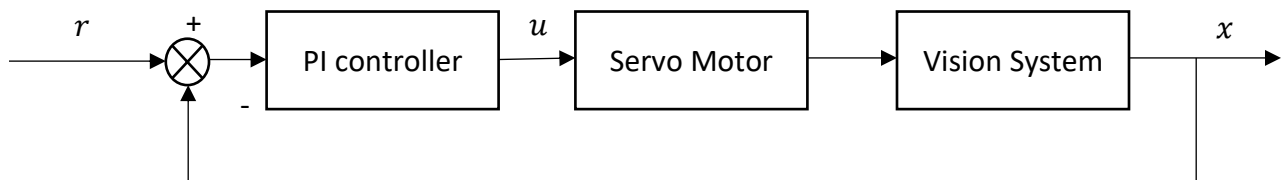


Figure 4.6 Block Diagram of the PI Controlled System

The PI control is robust for most system, but the tuning of the parameters of the PI controller can be challenging. A large gain of the proportional term provides a quick rising time but a large overshoot. A large integral gain eliminates the steady-state error but also increases the overshoot. For a constant PI controlled system, the tuning process is a process to find the tradeoff of every terms so that the overall performance of the

system is optimal. The anti-windup feature is added to traditional PI control to avoid the integral term to blowup. The coefficient of the integral term will be dynamically changed based on the Gaussian equation

$$k_i(t) = K_i \exp^{-\frac{e^2(t)}{2\sigma^2}}. \quad (16)$$

Note that $k_i(t)$ is not a constant coefficient. It is determined by current result of Gaussian equation with respect with current error $e(t)$. This approach also introduces one threshold value to switch between the pure P control and the PI control. Thus, the new control input is computed by following equation:

$$u(t) = \begin{cases} K_p e(t) & |e(t)| > \text{threshold}, \\ K_p e(t) + \int_{t_s}^t k_i(\tau) e(\tau) d\tau & |e(t)| \leq \text{threshold}. \end{cases} \quad (17)$$

where t_s is the time when the error dropped below the threshold. When the absolute value of error $|e(t)|$ becomes smaller and equal to the threshold value, it switches on the integral term. Otherwise, the coefficient of integral term is set to zero when the error is larger than the threshold value. Figure 4.7 shows the idea of this PI control with anti-windup feature. The area above dash line A and below dash line B is when a Proportional control is applied and the area between line A and B is when the PI with anti-windup feature is applied.

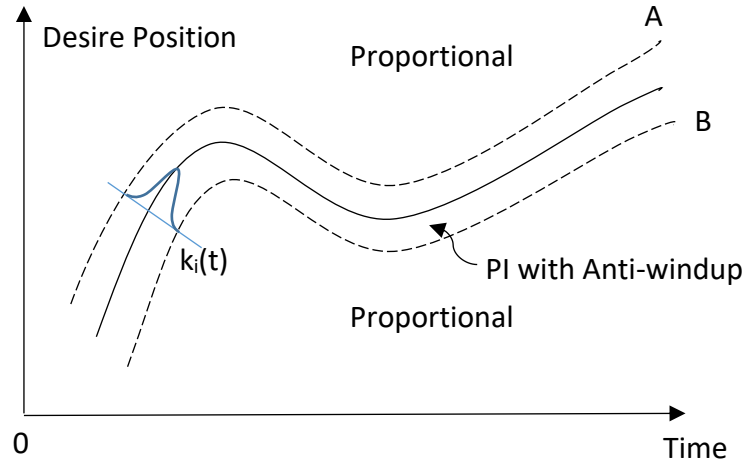


Figure 4.7 Non-linear PI with Gaussian Control

Figure 4.8 compares the constant PI control and the PI control with anti-windup feature for a desired the unit square wave signal. When the coefficients of proportional and integral terms are the same, the anti-windup featured PI control shares the same rising time with the constant PI control and has smaller overshoot. The settling time of the PI control with anti-windup feature increases a little bit, however.

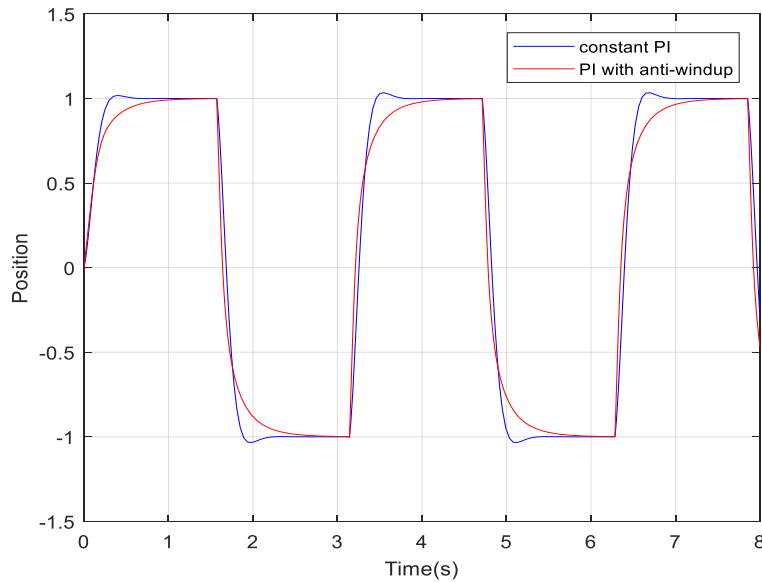


Figure 4.8 The Time Response of the PI with Anti-Windup Feature and the Constant PI

4.4 Model Reference Adaptive Control (MRAC)

A PI controller is already robust; however, there are still some uncertainties and unknown changes in the real system, which could result in poor performance of the constant controller. For example, the vision system is more than a pure delay in reality. The adaptive control approach is an attractive method to deal with these uncertainties. The Model Reference Adaptive Control (MRAC) in [22] provides the design flowchart of the MRAC.

Consider the robot plant as a first order system with unknown parameters

$$\dot{x} = ax + bu, \quad (18)$$

but the sign of b is assumed known given its physical meaning for the system. In this thesis, the sign of b is positive because the increase of the input torque will result in the movement of the robot arm in the defined positive direction.

Now consider an adaptive tracking situation. The reference model will still be a stable first order system. This reference model will make the reference state x_r track the trajectory dictated by the reference input r

$$\dot{x}_r = a_r x_r + b_r r. \quad (19)$$

In the s -domain, the reference model becomes:

$$X_m(s) = \frac{b_r}{s - a_r} R(s). \quad (20)$$

The negative sign of the a_r ensures that the reference model is stable and the tracking error will be eliminated when time goes to infinity.

The aim of the adaptive controller is to make the real system state x track the reference state x_r as well. If there exist ideal parameters θ_x^*, θ_r^* , then the input $u = \theta_x^*x + \theta_r^*r$ is guaranteed to work. Substitute the input into the plant

$$\dot{x} = (a + b\theta_x^*)x + b\theta_r^*r. \quad (21)$$

When $\theta_x^* = \frac{a_r - a}{b}$, $\theta_r^* = \frac{b_r}{b}$, then the closed-loop system will have the same dynamics as the reference model. The error is given $e = x - x_r$, then the error dynamics is

$$\dot{e} = \dot{x} - \dot{x}_r = a_r e, \quad (22)$$

thus, the real system state x tracks the reference position and the error converges to zero. Since the parameters a and b are unknown, the ideal parameters θ_x^*, θ_r^* cannot be directly computed.

The alternative way is to use θ_x, θ_r to estimate the ideal parameters online. The control input becomes $u = \theta_x x + \theta_r r$. This online estimation method is called adaptive law. The block diagram of this adaptive control system is shown in Figure 4.9.

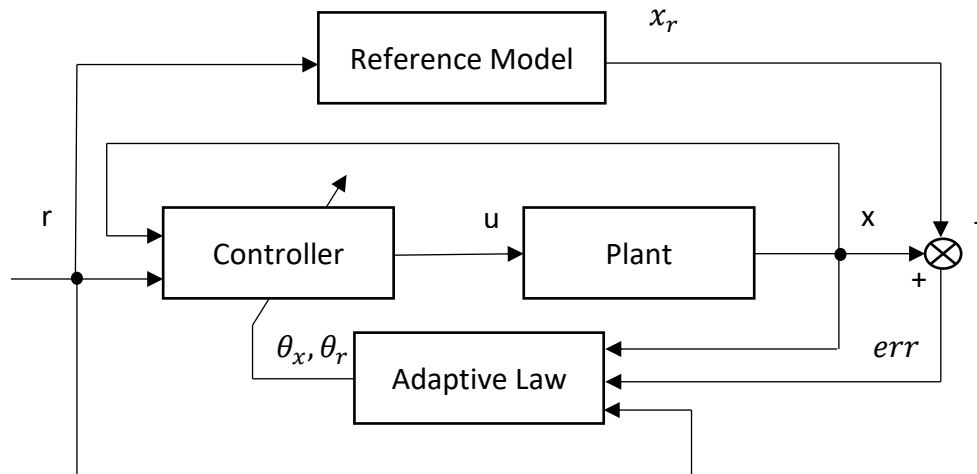


Figure 4.9 The Block Diagram of the MRAC

The next step is to find an appropriate parameter update law. The adaptive law derived from Lyapunov's direct method is called MRAC direct method [23]. The Lyapunov's direct method illustrates that if there exists a positive definite candidate function $V(x)$ such that its time derivative along the trajectories $\dot{V}(x)$ is negative definite, then the tracking error converges to zero.

Thus, after introducing the θ_x, θ_r , the errors between the ideal and estimated parameters is given by $\tilde{\theta}_x = \theta_x - \theta_x^*, \tilde{\theta}_r = \theta_r - \theta_r^*$. Then the closed-loop system becomes:

$$\begin{aligned}\dot{x} &= (a + b\theta_x^*)x + b\theta_r^*r + b\tilde{\theta}_x x + b\tilde{\theta}_r r \\ &= a_r x + b_r r + b\tilde{\theta}_x x + b\tilde{\theta}_r r.\end{aligned}\quad (23)$$

And the error dynamics is

$$\dot{e} = a_r e + b\tilde{\theta}_x x + b\tilde{\theta}_r r. \quad (24)$$

Consider the Lyapunov's candidate function

$$V(e, \tilde{\theta}_x, \tilde{\theta}_r) = \frac{1}{2}e^2 + \frac{1}{2\gamma_x}|b|\tilde{\theta}_x^2 + \frac{1}{2\gamma_r}|b|\tilde{\theta}_r^2, \quad (25)$$

The time derivative along the trajectories $\dot{V}(e, \tilde{\theta}_x, \tilde{\theta}_r)$ is

$$\begin{aligned}\dot{V}(e, \tilde{\theta}_x, \tilde{\theta}_r) &= a_r e^2 + e \cdot |b|\tilde{\theta}_x x \cdot \text{sgn}(b) + e \cdot |b|\tilde{\theta}_r r \\ &\quad \cdot \text{sgn}(b) + \frac{1}{\gamma_x}|b|\tilde{\theta}_x \dot{\tilde{\theta}}_x + \frac{1}{\gamma_r}|b|\tilde{\theta}_r \dot{\tilde{\theta}}_r.\end{aligned}\quad (26)$$

By choosing

$$\dot{\tilde{\theta}}_x = \dot{\theta}_x = -\gamma_x e \cdot x \cdot \text{sgn}(b), \quad (27)$$

$$\dot{\tilde{\theta}}_r = \dot{\theta}_r = -\gamma_r e \cdot r \cdot \text{sgn}(b),$$

we obtain

$$\dot{V}(e, \tilde{\theta}_x, \tilde{\theta}_r) = a_r e^2 \leq 0. \quad (28)$$

It turns out the final system is stable and the tracking error converges to zero.

The parameters $a_r, b_r, \gamma_x, \gamma_r, \theta_x(0), \theta_r(0)$ will influence the performance of the overall system. Figure 4.10 shows the time domain response of an MRAC system for a unit square wave reference input.

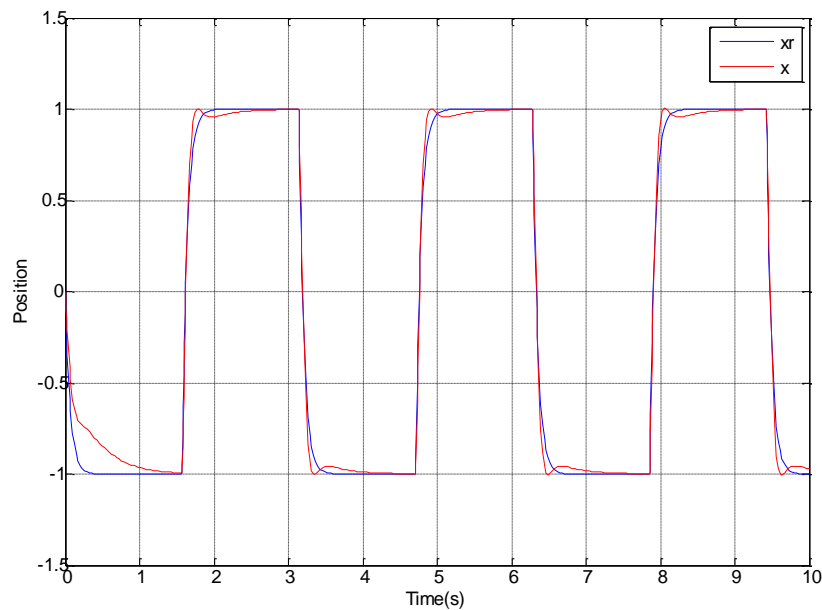


Figure 4.10 Time Domain Response of MRAC System under Square Wave

4.5 PI-MRAC Control

As mentioned above, there are a lot of advantages of PI control; however, the properties of the adaptive control are also attractive. The hybrid design of a PI controller together with a Model Reference Adaptive Controller was proposed in [24]. Thus, the block diagram of the hybrid PI-MRAC system looks like Figure 4.11.

The unknown plant is given by equation (29), where the $A_o \in \mathbb{R}^{2 \times 2}$ and $B_o \in \mathbb{R}^{2 \times 1}$ in canonical observable form, $W^{*T} \phi(x)$ represents the non-linear terms and u is the control input,

$$\dot{x} = A_o x + B_o \lambda^* (u + W^{*T} \phi(x)). \quad (29)$$

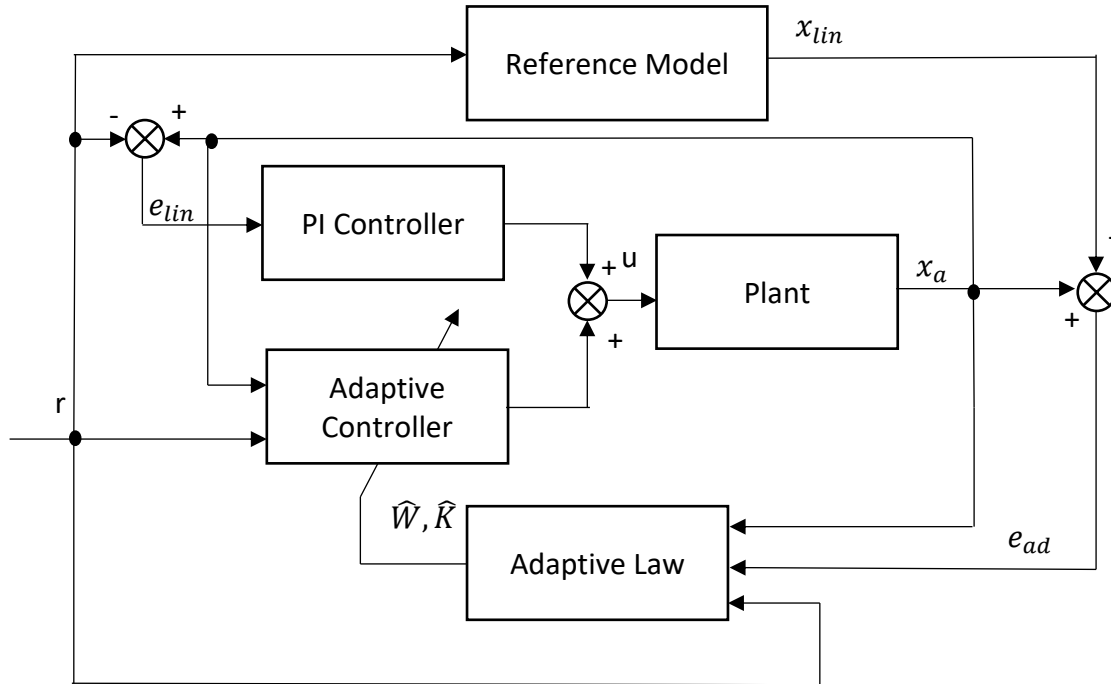


Figure 4.11 The Block Diagram of the Hybrid PI-MRAC System

If the reference model is designed based on the unknown plant with a PI controller, the new state of reference model is $x_{lin} = [x_1 \ x_2 \ x_I]^T$, where the integral term $x_I = \int_0^t (x_1(\tau) - r(\tau)) d\tau$ is introduced and r is the reference input. The x_1 is the position term of the state vector x and $x \in \mathbb{R}^{2 \times 1}$.

The reference model will take the control signal $u_{lin}(t)$ into the system, which is calculated using the PI control. The error between the reference input r and the position term x_1 of the reference model state is used to calculate the PI control effort

$$e_{lin}(t) = x_1(t) - r(t). \quad (30)$$

Thus, the control effort from the PI controller is

$$u_{lin}(t) = -k_p e_{lin}(t) - k_I \int_0^t e_{lin}(\tau) d\tau. \quad (31)$$

The control effort is represented by a reference model state like equation (32),

where $K^* = [k_p \ 0 \ k_I]^T$, $\mathbf{0}$ is 2×1 zero vector, \mathbf{I} is vector $[1 \ 0]$,

$$u_{lin} = -K^{*T} \left(x_{lin} - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right). \quad (32)$$

Then the reference linear model becomes

$$\dot{x}_{lin} = \begin{bmatrix} A_o & \mathbf{0} \\ \mathbf{I} & 0 \end{bmatrix} x_{lin} + \begin{bmatrix} B_o \\ 0 \end{bmatrix} u_{lin} + \begin{bmatrix} \mathbf{0} \\ -1 \end{bmatrix} r. \quad (33)$$

The PI input into the reference model, it can be rewritten as

$$\dot{x}_{lin} = \begin{bmatrix} A_o & \mathbf{0} \\ \mathbf{I} & 0 \end{bmatrix} x_{lin} - \begin{bmatrix} B_o \\ 0 \end{bmatrix} K^{*T} x_{lin} + \begin{bmatrix} k_p B_o \\ -1 \end{bmatrix} r, \quad (34)$$

$$\dot{x}_{lin} = A_r x_{lin} + \begin{bmatrix} k_p B_o \\ -1 \end{bmatrix} r,$$

where

$$A_r = \begin{bmatrix} A_o & \mathbf{0} \\ \mathbf{I} & 0 \end{bmatrix} - \begin{bmatrix} B_o \\ 0 \end{bmatrix} K^{*T}. \quad (35)$$

Similarly, the unknow plant with PI control can be also written into this way

$$u_{pi} = -K^{*T} \left(x - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right). \quad (36)$$

The plant becomes

$$\begin{aligned} \dot{x}_a = & \begin{bmatrix} A_o & \mathbf{0} \\ \mathbf{I} & 0 \end{bmatrix} x_a + \begin{bmatrix} \mathbf{0} \\ -1 \end{bmatrix} r - K^{*T} \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right) \\ & + \begin{bmatrix} B_o \\ 0 \end{bmatrix} \lambda^* \left(u + W^{*T} \phi(x_a) + \frac{1}{\lambda^*} K^{*T} \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right) \right), \end{aligned} \quad (37)$$

$$\begin{aligned}\dot{x}_a &= A_r x_a + \begin{bmatrix} k_p B_o \\ -1 \end{bmatrix} r \\ &+ \begin{bmatrix} B_o \\ \mathbf{0} \end{bmatrix} \lambda^* \left(u + W^{*T} \phi(x_a) + \frac{1}{\lambda^*} K^{*T} (x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix}) \right)\end{aligned}$$

The control effort is the linear combination of the PI control and the MRAC control, where the $u_{pi}(t)$ is determined by the PI controller and the $u_{ad}(t)$ is found by the adaptive controller,

$$u(t) = u_{pi}(t) + u_{ad}(t). \quad (38)$$

Then, the adaptive control effort should be designed to eliminate the error between the plant and reference model. Consider the plant expressed in equation (39)

$$\begin{aligned}\dot{x}_a &= A_r x_a + \begin{bmatrix} k_p B_o \\ -1 \end{bmatrix} r \\ &+ \begin{bmatrix} B_o \\ \mathbf{0} \end{bmatrix} \lambda^* \left(u_{ad} + W^{*T} \phi(x_a) + \frac{1 - \lambda^*}{\lambda^*} K^{*T} (x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix}) \right).\end{aligned} \quad (39)$$

The tracking error between the plant and the reference model is defined as

$$e_{ad} = x_a - x_{lin} \quad (40)$$

The error dynamic is obtained as following

$$\begin{aligned}\dot{e}_{ad} &= \dot{x}_a - \dot{x}_{lin} \\ &= A_r e_{ad} \\ &+ \begin{bmatrix} B_o \\ \mathbf{0} \end{bmatrix} \lambda^* \left(u_{ad} + W^{*T} \phi(x_a) + \frac{1 - \lambda^*}{\lambda^*} K^{*T} (x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix}) \right).\end{aligned} \quad (41)$$

There is an ideal adaptive control effort u_{ad}^* like equation (42) to simplify the error dynamics $\dot{e}_{ad} = A_r e_{ad}$, which guarantees the stability of the overall system,

$$u_{ad}^* = -W^{*T} \phi(x_a) - \frac{1 - \lambda^*}{\lambda^*} K^{*T} \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right). \quad (42)$$

The values of A_o, B_o, λ^* are unknown, but the sign of λ^* is known. W^* is an unknown constant vector and nonlinear functions $\phi(x_a)$ contains the overall uncertainty over the linear coefficient of the system dynamics.

Because the values of W^*, λ^* are unknown, the ideal control effort u_{ad}^* cannot be implemented. The adaptive variables \hat{K}, \hat{W} are introduced into the system instead of the ideal values. Then the ideal values are estimated online by the adaptive laws.

The estimated adaptive control effort becomes

$$u_{ad} = -\hat{W}^T \phi(x_a) - \hat{K}^T \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right). \quad (43)$$

Then, the errors between online adaptive control parameters and ideal parameters are defined as $\tilde{K} = \hat{K} - \frac{1-\lambda^*}{\lambda^*} K^*, \tilde{W} = \hat{W} - W^*$.

Therefore, the error dynamics is rewritten as

$$\dot{e}_{ad} = A_r e_{ad} - \begin{bmatrix} B_o \\ 0 \end{bmatrix} \lambda^* \left(\tilde{W}^T \phi(x) + \tilde{K}^T \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right) \right). \quad (44)$$

According to the design procedure of the adaptive control, the difference between the ideal parameters and the online estimated parameters should be considered into Lyapunov's candidate function.

Now, the Lyapunov candidate function is defined as

$$V(e_{ad}, \tilde{K}, \tilde{W}) = e_{ad}^T P e_{ad} + |\lambda^*| \tilde{K}^T \Upsilon_K^{-1} \tilde{K} + |\lambda^*| \tilde{W}^T \Upsilon_W^{-1} \tilde{W}, \quad (45)$$

where the adaptive gains Υ_K, Υ_W are positive and P is a positive definite matrix. The derivative function of the Lyapunov candidate function is

$$\begin{aligned}
\dot{V}(e_{ad}, \tilde{K}, \tilde{W}) &= e_{ad}^T (A_r^T P + P A_r) e_{ad} \\
&+ 2|\lambda^*| \tilde{K}^T \left(-\text{sign}(\lambda^*) \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right) e_{ad}^T P \begin{bmatrix} B_o \\ 0 \end{bmatrix} + \Upsilon_K^{-1} \dot{\tilde{K}} \right) \\
&+ 2|\lambda^*| \tilde{W}^T \left(-\text{sign}(\lambda^*) \phi(x_a) e_{ad}^T P \begin{bmatrix} B_o \\ 0 \end{bmatrix} + \Upsilon_M^{-1} \dot{\tilde{W}} \right).
\end{aligned} \tag{46}$$

The adaptive law is directly calculated by Lyapunov's direct method,

$$\begin{aligned}
\dot{\tilde{K}} &= \dot{\hat{K}} = \text{sign}(\lambda^*) \Upsilon_K \left(x_a - \begin{bmatrix} r \\ \mathbf{0} \end{bmatrix} \right) e_{ad}^T P \begin{bmatrix} B_o \\ 0 \end{bmatrix}, \\
\dot{\tilde{W}} &= \dot{\hat{W}} = \text{sign}(\lambda^*) \Upsilon_w \phi(x_a) e_{ad}^T P \begin{bmatrix} B_o \\ 0 \end{bmatrix}.
\end{aligned} \tag{47}$$

Substitute the adaptive law equation (47) into derivative function of the Lyapunov candidate function, the derivative of Lyapunov's candidate function will become

$$\begin{aligned}
\dot{V}(e_{ad}, \tilde{K}, \tilde{W}) &= e_{ad}^T (A_r^T P + P A_r) e_{ad} \\
&= -e_{ad}^T Q e_{ad} \leq 0,
\end{aligned} \tag{48}$$

where Q satisfied Lyapunov's algebraic equation $A_r^T P + P A_r + Q = 0$ with P positive definite and A_r is Hurwitz. Hence, LaShalle-Yoshizawa Theorem guarantees e_{ad} will converge to zero as time goes to infinity and all signals stay bounded [23].

This hybrid PI-MRAC system will be determined by $A_r, B_r, k_p, k_I, \Upsilon_K, \Upsilon_W$ and the initial condition of \hat{W}, \hat{K} . Figure 4.12 shows the time domain response of this hybrid PI-MRAC closed-loop system. It provides better tracking ability than what we obtained using only the PI or only the MRAC controls.

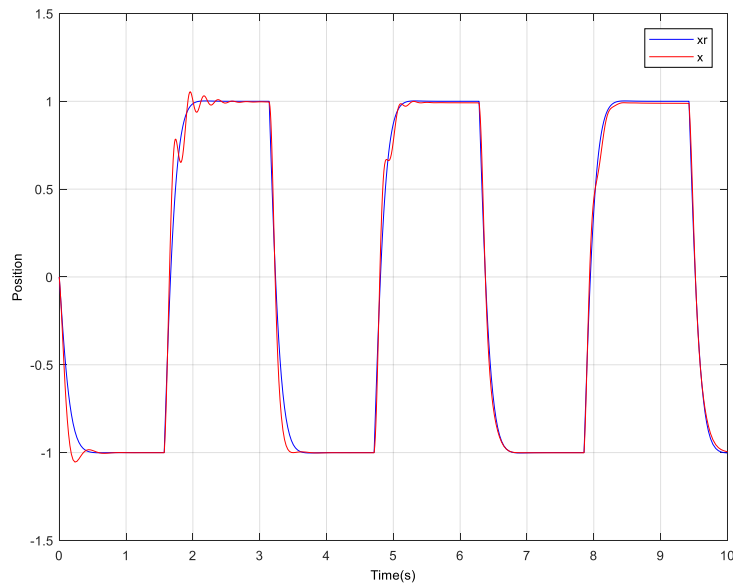


Figure 4.12 The Time Domain Response of the PI-MARC System under the Square Wave

4.6 Remarks

This chapter discussed the design of three different controllers for the Painting Robot. The control problem is summarized at the beginning of this chapter. To analyze the Painting Robot more efficiently, a system identification was conducted. The PI with Anti-windup feature was designed to overcome the drawbacks of the traditional PI controller. In order to deal with the uncertainties of the system, the Model Reference Adaptive Control (MRAC) was introduced into this thesis. Finally, the hybrid PI-MRAC controller was implemented to maintain both the advantages of the PI and MRAC control approaches. The stability of each controller has been studied in each case. The adaptive laws were derived using Lyapunov's Direct Method. The time domain response of each controllers using a square wave reference signal have been provided for validating the proposed approaches.

Chapter 5 Testing and Experiments

The validation of the above design was conducted running a few experiments. Recall from the previous chapters that the Painting Robot is designed as a standalone system, which can be attached to any ground vehicle. To verify the performance of the Painting Robot, a testing platform was constructed. The experimental results were presented using such a testing platform. The results demonstrated the effectiveness of the computer vision algorithm and control algorithm.

5.1 Overview of the Testing Platform

The testing platform was designed separately and assembled using aluminum extrusion brackets. The diameter of the wheel is 5 in. Two wheels are rigid and the other two are swivel with side brakes. The testing platform can travel on grass. Figure 5.1 is the 3D model of the testing platform.

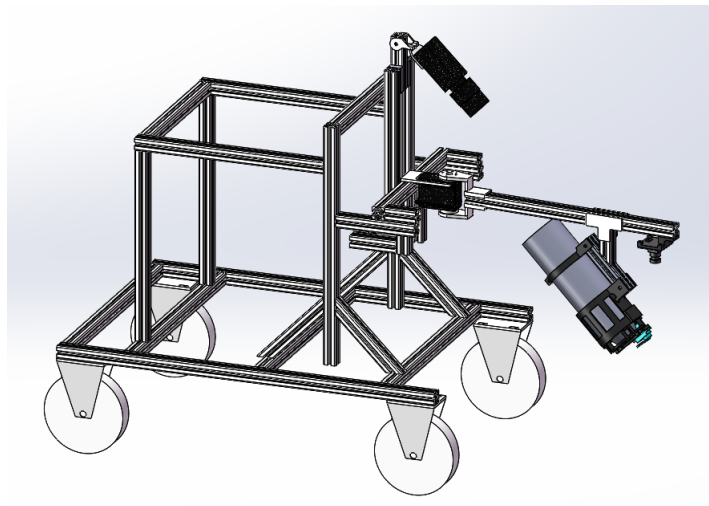


Figure 5.1 3D Model of the Testing Platform

5.2 Line Tracking Experiments in Lab

The ability of line tracking is the fundamental function to the line painting, which we need to test. The first generation of the Painting Robot is shown in Figure 5.2. It is a single camera system. The robot makes decisions by observing the previous painted line using the overview camera. The robot makes decisions by observing the previous painted line using the overview camera. The first-generation system was used to verify the line tracking ability in the lab environment. The experiments were conducted in the lab by manually giving the system an approximate harmonic excitation. The three controllers discussed in Chapter 4 were tested. The output signals of the system were plotted in the following figures.

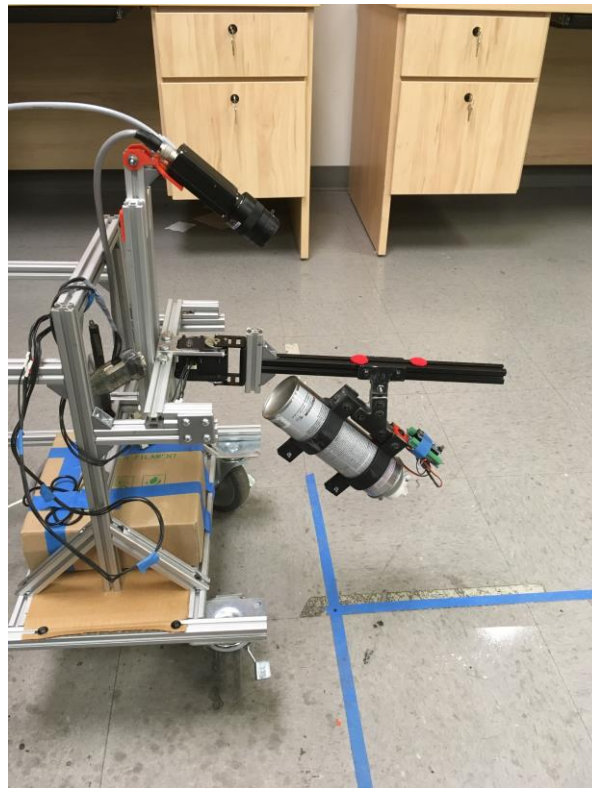


Figure 5.2 The 1st Generation Painting Robot

Figure 5.3 shows the result of the time domain response using a PI controller with anti-windup feature under the manually approximate harmonic excitation. The position error is defined by equation (7) in Chapter 3. The x coordinates of the target position and current position are plotted in Figure 5.3. The result indicates a delay between the target position and current position. Based on the study in Chapter 4, the PI controller with anti-windup feature does not have the overshoot in the step response. Thus, the result in Figure 5.3 shows the correct properties of this controller discussed before. The delay is due to the image processing, which can be treated as a pure delay in the system block diagram.

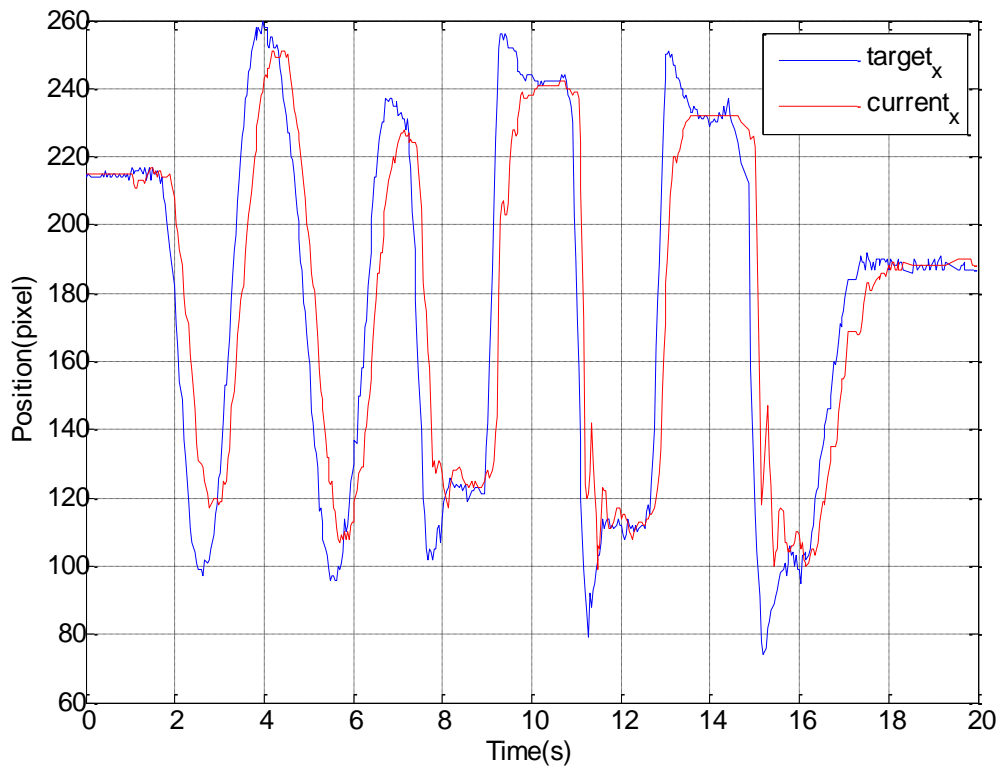


Figure 5.3 Time Domain Response of PI Gaussian Controller

The same experiment was also conducted to study the behavior of the MRAC controller presented in Chapter 4. The main purpose of studying the MRAC controller is to demonstrate the adaptive ability of the classical adaptive tracking example [19]. The reference model was selected as a first order tracking equation, which is similar to a low pass filter. The small delay between target position and reference model was introduced by the first order tracking equation itself. Thus, the black dash line and blue line in Figure 5.4 indicate that theoretic delay. There is another delay between the current position and the target position because of the time of image processing.

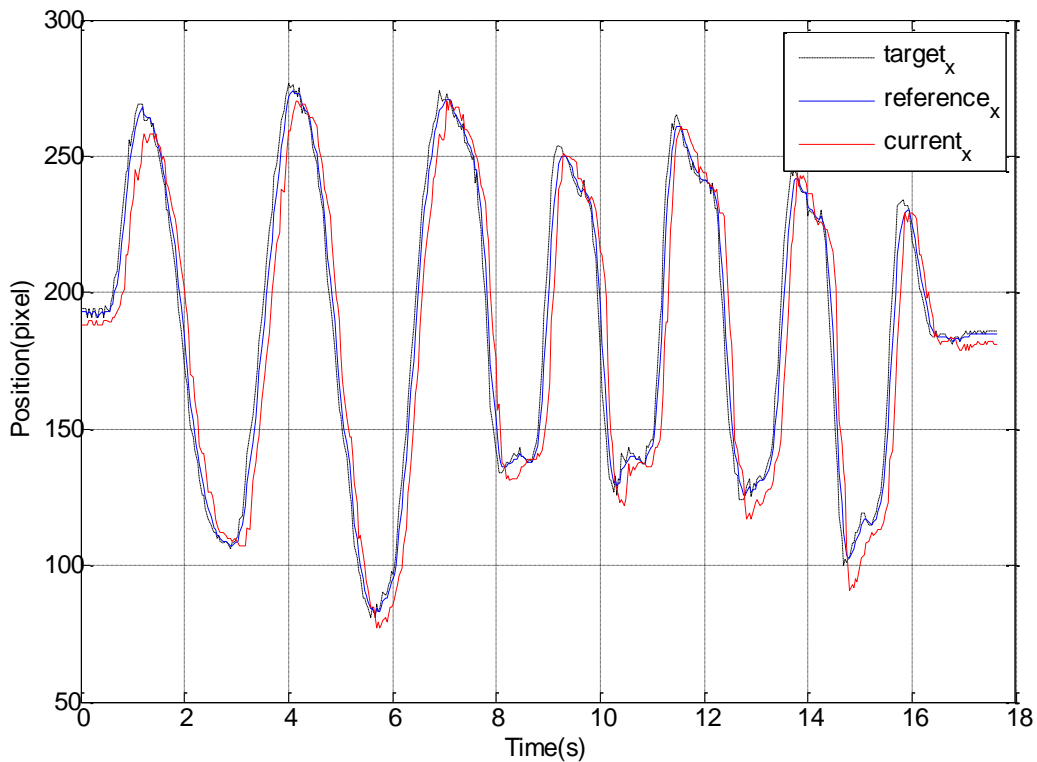


Figure 5.4 Time Domain Response of the MRAC Controller

Comparing to MRAC, the PI-MRAC hybrid controller has a stronger tracking ability, which is discussed in Chapter 4. Figure 5.5 shows the time domain response of

the PI-MRAC hybrid controller by the same experiment method. Since the PI-MRAC is more complicate than MRAC, the program decreases the time step of the PI-MRAC while using the continuous model. It results in the larger delay between the target position (black dash line) and reference position (blue line) than MRAC.

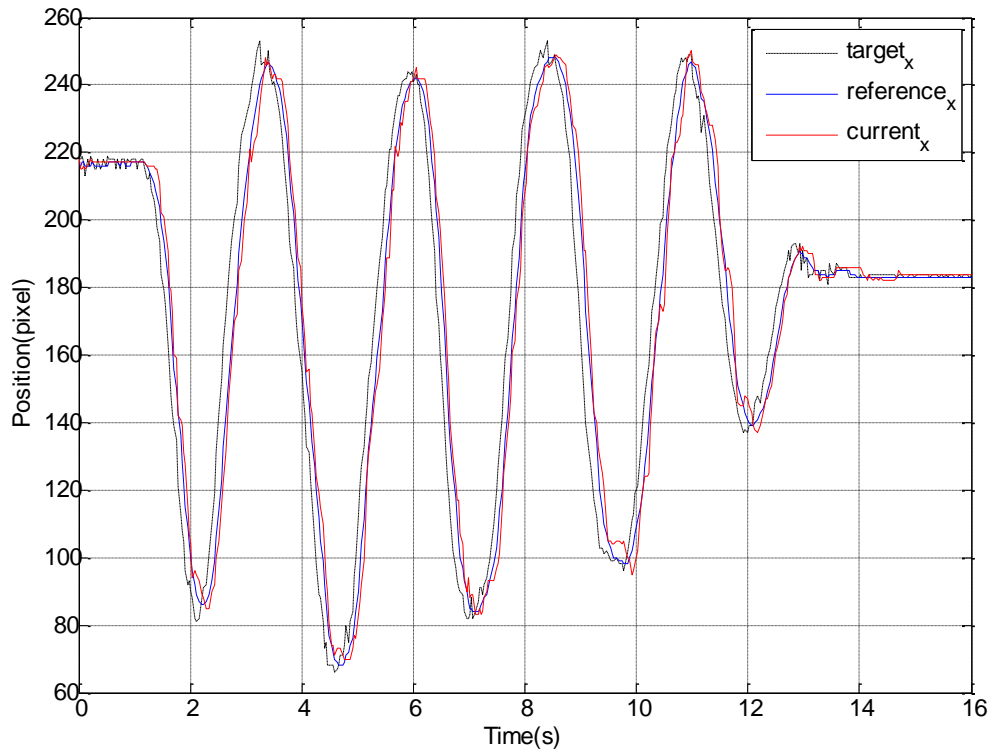


Figure 5.5 Time Domain Response of the PI-MRAC Controller

However, the delay between current position and reference position is almost eliminated thanks to the advantages of PI-MRAC. The control effort of the PI-MRAC controller is calculated by equation (34), which is linear combination of the PI control effort and Adaptive control effort. In Figure 5.6, the PI control effort takes up the majority work, while the Adaptive control effort adaptively eliminates the error between the current position and the reference position.

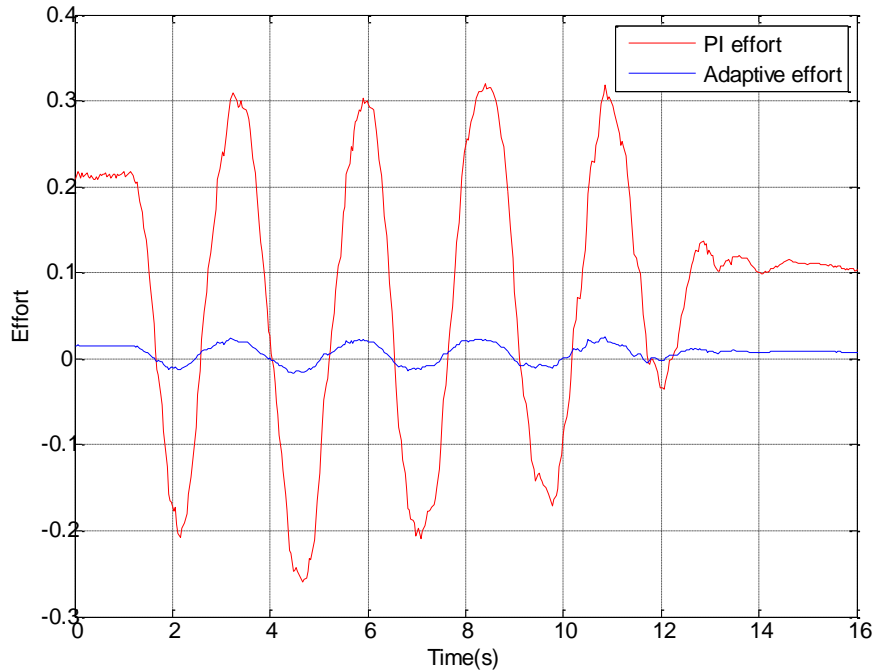


Figure 5.6 The Control Effort in the PI-MRAC controller

Overall, the delay between target position and current position cannot be totally eliminated because of the time of image processing and the speed limit of servo motor. The PI-MRAC hybrid controller keeps the advantages both PI and Adaptive controllers. The experimental results above demonstrate the ability of line tracking, which means the controllers are designed successfully.

5.3 On-field Line Painting with Reference Line

As mentioned in Chapter 2, a two-camera system was introduced in order to increase the accuracy of painting. The two-camera Painting Robot is shown in Figure 5.7. And the line tracking ability was demonstrated in Section 5.3. Thus, the line painting problem can be easily solved by setting a reference line in the real world.



Figure 5.7 The Two-camera Painting Robot

The experiments were conducted on grass fields. A disturbance was introduced to the Painting Robot because of the rough surface of the ground. The goal of the Painting Robot is to paint a straight line following the preset reference line. Figure 5.8 shows the setting of the reference line. Then, Figure 5.9 shows the results of the line painting by PI Gaussian controller and PI-MRAC hybrid controller. The PI-MRAC has a better result than PI Gaussian according to Figure 5.9. Obviously, the drawbacks of this method are:

- It is required to manually set a reference line;
- The blank area is not desired after taking away the reference line.



Figure 5.8 The Setting of the Reference Line



(a) PI Gaussian



(b) PI-MRAC

Figure 5.9 The Results of Line Painting by PI Gaussian and PI-MRAC Controllers

The experiments tried different reference lines with different width, which are shown in Figure 5.10. Although, the result of thinnest reference line looks better than others, the blank area is still not wanted.



(a) width = 25 mm (b) width = 20 mm (c) width = 10 mm

Figure 5.10 Results of Different Reference Line with Different Width

This method can be extended to refresh old lines. The painted line on the grass field fades after several days. To refresh the old line, the Painting Robot can take the old line as a reference line then paint a new color on it. Figure 5.11 shows one line updating result.

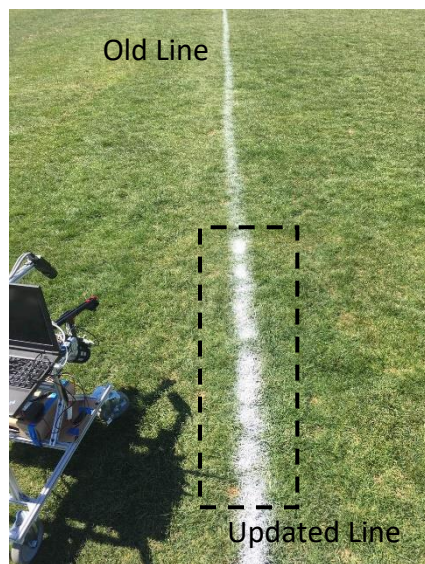


Figure 5.11 Line Updating Result

5.4 On-field Line Painting without Reference Line

As discussed, the Painting Robot can successfully work with the reference line. As mentioned in Section 3.4, the line painting problem is more difficult than line tracking problem. In Figure 5.12, the error painted spot will affect the continuous detection of the next target position. When the Painting Robot is affected by disturbances, it cannot avoid to accidentally spray at the wrong position when the travel speed is high. However, the orange spot in Figure 5.14 (b) will make it difficult for the Painting Robot to understand it sprayed at the wrong position.

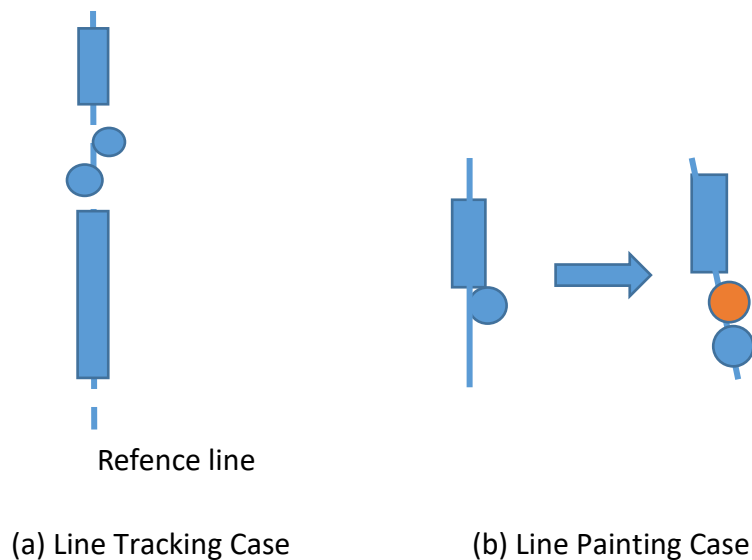


Figure 5.12 Different Cases Under the Disturbance

Because the line painting relies on the previous painted line, the tolerance of mistaken spray is very low. Since the computer vision algorithm is too simple to understand what spray is accidental, the speed of the line painting case is slower than the line tracking case. Figure 5.13 shows the Paint Robot can paint straight line without the reference line.



Figure 5.13 Result of Line Painting without the Reference Line

5.5 Remarks

This chapter discussed the verification of the computer vision algorithm and the controllers. Both simulations and experiments were conducted. The results indicated the correctness of the simulation. The test platform was constructed for the purpose of conducting the field experiments, which showed that the Painting Robot is able to paint straight line with or without reference line. In general, the PI Gaussian controller is robust enough, but the PI-MRAC hybrid controller inherits the advantages of both PI and Adaptive controller.

Chapter 6 Conclusion and Future Work

This thesis presented the design of a Painting Robot to solve the line painting problem. The painting robot consisted of two parts: prediction and control. In Chapter 3, the prediction problem was solved using computer vision techniques. The computer vision algorithm was proven to be concise and effective. In order to speed up the image processing, the multi-threads program was studied in Chapter 3 as well. The multi-threads programming reduced the processing time of the computer vision algorithm, which reduced the pure delay effect in the control problem. In Chapter 4, three types of controllers were studied theoretically and simulated. A PI controller with anti-windup is compared to a traditional PI controller. MRAC introduces the adaptive ability into the system. At last, PI-MRAC hybrid controller inherits the advantages of both PI and Adaptive controllers.

The simulations and experiments were conducted to verify the design of the Painting Robot. The experiments indicated that the line tracking is easier than line painting. The Painting Robot is capable of completely solving the line tracking problem, although the Painting Robot can paint a line without the reference line at a lower speed. Also, the results indicated that the PI-MRAC hybrid controller provides better tracking capabilities than the PI Gaussian controller. Finally, it can be concluded that the design of the Paint Robot in this thesis meets the requirements of the line painting.

The straight-line painting is a relatively simple task for the Painting. Problems left to solve include how to increase the line painting speed and how to deal with painting corners.

References

- [1] Debnath, Ashim Kumar, et al. "Beyond the barriers: Road construction safety issues from the office and the roadside." *Advances in Safety Management and Human Factors* 10 (2014): 185.
- [2] Ahsan, Mubashir, and Khuram Saleem Alimgeer. "Autonomous Ground Vehicle." *International Journal of Technology and Research* 1.3 (2013).
- [3] Durrant-Whyte, Hugh. "A critical review of the state-of-the-art in autonomous land vehicle systems and technology." *Albuquerque (NM) and Livermore (CA), USA: Sandia National Laboratories* 41 (2001).
- [4] Christie, Gordon, et al. "Radiation Search Operations using Scene Understanding with Autonomous UAV and UGV." *arXiv preprint arXiv:1609.00017* (2016).
- [5] Shoemaker, Adam, and Alexander Leonessa. "Bioinspired tracking control of high speed nonholonomic ground vehicles." *Journal of Robotics* 2015 (2015): 11.
- [6] Wiener, Norbert. *Cybernetics or Control and Communication in the Animal and the Machine*. Vol. 25. MIT press, 1961.
- [7] Szederkényi, Gábor, Rozália Lakner, and Miklós Gerzson. *Intelligent control systems: an introduction with examples*. Vol. 60. Springer Science & Business Media, 2006.
- [8] Bennett, Stuart. *A history of control engineering, 1930-1955*. No. 47. IET, 1993.
- [9] Bennett, Stuart. "A brief history of automatic control." *IEEE Control Systems Magazine* 16.3 (1996): 17-25.
- [10] Singh, Sanjay Kr, et al. "A robust skin color based face detection algorithm." *淡江理工學刊* 6.4 (2003): 227-234.
- [11] Schalkoff, Robert J. *Digital image processing and computer vision*. Vol. 286. New York: Wiley, 1989.
- [12] Lindeberg, Tony. "Feature detection with automatic scale selection." *International journal of computer vision* 30.2 (1998): 79-116.

- [13] Canny, John. "A computational approach to edge detection." *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986): 679-698.
- [14] Sturm, Peter. "Pinhole camera model." *Computer Vision*. Springer US, 2014. 610-613..
- [15] Herlihy, Maurice, and Nir Shavit. *The Art of Multiprocessor Programming*, Revised Reprint. Elsevier, 2012.
- [16] Dorf, Richard C., and Robert H. Bishop. "Modern control systems." (1998).
- [17] https://en.wikipedia.org/wiki/Control_theory
- [18] Shim, D. Hyunchul, Hyoun Jin Kim, and Shankar Sastry. "Control system design for rotorcraft-based unmanned aerial vehicles using time-domain system identification." *Control Applications, 2000. Proceedings of the 2000 IEEE International Conference on*. IEEE, 2000.
- [19] Salem, Farhan A. "Modeling, Simulation and Control Issues for a Robot ARM; Education and Research (III)." *International Journal of Intelligent Systems and Applications* 6.4 (2014): 26.
- [20] http://users.encs.concordia.ca/~realtime/elec372/docs/MODELING_SERVO_MOTOR_SYS_TEM2.pdf
- [21] Ljung, Lennart. *System identification*. John Wiley & Sons, Inc., 1999.
- [22] Ioannou, Petros A., and Jing Sun. *Robust adaptive control*. Courier Corporation, 2012.
- [23] Khalil, Hassan K. *Nonlinear Systems*. Prentice-Hall, New Jersey, 1996.
- [24] Jaramillo, Paola, Adam Shoemaker, and Alexander Leonessa. "Skeletal muscle contraction control and tracking using an adaptive augmented pi control approach." *Converging Clinical and Engineering Research on Neurorehabilitation*. Springer Berlin Heidelberg, 2013. 135-139.