2017

Advanced Line-Follower Robot

Lei Wang
Cleveland State University

ADVANCED LINE-FOLLOWER ROBOT

LEI WANG

Bachelor of Electrical Engineering

Shenyang LiGong University

July 2011

Submitted in partial fulfillment of requirements for the degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

at the

CLEVELAND STATE UNIVERSITY

December 2017

We hereby approve this thesis for

Lei Wang

Candidate for the Master of Science in Electrical Engineering degree for the

Department of Electrical Engineering and Computer Science

And the CLEVELAND STATE UNIVERSITY

College of Graduate Studies

Thesis Chairperson, Dr. Zhiqiang Gao

Electrical Engineering and Computer Science Department & Date

---

Thesis Committee Member, Dr. Lili Dong

Electrical Engineering and Computer Science Department & Date

---

Thesis Committee Member, Dr. Shiqi Zhang

Electrical Engineering and Computer Science Department & Date

Student's Date of Defense (12/04/2017)

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Zhiqiang Gao for his careful guidance and encouragement throughout this project. Dr. Zhiqiang Gao can always point out my mistakes very correctly and teach me how to solve the problems in both studies and life. More importantly, he shared with me his way of thinking.

I would like to thank Han Zhang for all his help in my research work and his great friendship.

I would like to thank the committee members Dr. Lili Dong and Dr. Shiqi Zhang for reviewing my thesis and offering constructive advice.

I would like to thank my wife Tingting Wu for her care and trust that saved me so many times when I struggled with difficulties.

I would like to thank my parents for their love that more than I can say.

ADVANCED LINE-FOLLOWER ROBOT

LEI WANG

ABSTRACT

In this research, an Advanced Line-follower Robot (ALFR) was designed and built. The ALFR mainly consists of the sensor array (QTR-8A), the high-performance microchips (TMS320f28335, TMS320f28069) and two motors (BLY172S-24V-4000). The ALFR keeps the basic function of the Line-follower Robot (LFR) but applies more advanced control theories, such as Proportional Integral Derivative (PID), Active Disturbance Rejection Control (ADRC) and Iterative Learning Control (ILC). PID and ADRC have been tested in the ALFR. The ALFR control problems and the results have been discussed in this thesis. Suggestions are also provided for research on unsolved problems. In particular, the mathematical models of ALFR have been established for both position and speed control. The solutions based on PID, ADRC and ILC are proposed and tested in simulation. The main objective of this thesis is realized in combining methods from control theories with realities in the context of formulating and solving practical problems in a physical process.

Table of Contents

List of Figures

Figure Page

CHAPTER I

Introduction

1.1 BACKGROUND

A line-follower robot (LFR) is a device that can track a pre-designed line [1, 2]. In this research, the robot tracks a black line on the white ground. Normally the LFR only track one line at a time. The Advanced LFR (ALFR) not only can track a line but can also evolve on its own to track the line in a shorter amount of time because of its ability in disturbance rejection and learning. In this research, the core problem is to improve disturbance rejection. This ability will not come from better a mathematical model because to establish a very accurate mathematical model of the controlled system is a passive way of disturbance rejection and not very practical. Details will be discussed in the ADRC review section. Another objective of this research is giving the ALFR self-learning ability. This enhancement was made possible by two elements: one is the ILC

and the other one is an extra position sensor (usually the LFR has only one kind of sensor, but the ALFR has two kinds of sensors).

For the LFR motion control design, the objective is to run fast and accurately. Hardware-wise, there is a couple of solutions, such as setting the robot with more powerful microprocessors or more powerful motors. This is possible, but not recommended. The problem can also be solved using better controls. The robot easily speeds up, but at the same time, the robot is easily out of control. Control is important to engineers. Even if the final results are the same, engineers need to filter out the best way. What is the best way? The answer is the one that is the simplest and the most effective. As mentioned above, advanced control theories can improve the LFR. However, what is the reason to make the robot faster or more stable? Normal logic-based control can make the robot work [1, 3], but the performance of the LFR is often not good enough. It will be shown in this thesis that after adopting the PID [2] or ADRC control methods to the robot, the robot runs faster and smoother because the robot gets much more information of the control process than in the current method. For example, PID is based on the feedback theory, and the feedback is used to calculate output error, and then the error changes the robot's action for the purpose of correcting the error. In addition, the PID's parameters determine the behavior of the robot. Under the same condition, ADRC extracts more information, especially the disturbance information, from the same input and output signals in the physical process. It is demonstrated that the more given to the controller, the better the robot will perform. It is well known that dynamic information plays a significant role in control, but control has to be back to the practice [4].

No matter what controller is used, PID or ADRC, the robot tracks the same line in the same pattern. By itself, the robot will not improve any further in repeated experiments. That is, the robot accomplishes the loop-tracking loop with the same time and the same speed. The ALFR can be more intelligent. Because the robot is always "learning", the ALFR is able to memorize data and analyze it. Therefore, the robot can improve itself and find a better path through repeated experience. For example, the ALFR can be made to adjust its speed depending on the different conditions calculated based on the previous results.

All in all, the ALFR is more intelligent than the LFR. It can use control methods such as PID and ADRC, as well as iterative learning control (ILC). The ILC can be applied to the feed-forward control loop; PID and ADRC can be applied to the feedback control loop with the extra sensors. In general, ALFR has provided many new possibilities to be investigated.

LFR has existed for a long time and there are other products that are similar to the LFR. Due to its simple construction and function (line following), the idea of the LFR can be found everywhere: auto-drive cars, automatic-cruise planes, industrial robots, domestic robots and medical robots [3]. Some factory forklifts that can deliver goods automatically are examples as they have line-following patterns somewhat similar to the LFR. It shows that this mechanism is often applied and used in daily life. In addition, the LFR is a good prototype in learning control theory and embedded systems. The LFR includes two parts: the hardware consists of motors, body constructions, and control boards. The software consists of programming, controller design, filter design, and control method design. These two parts are the core of the LFR because the construction

decides the operating principle of the system and the control properties decide how to design the controller.

The LFR has many types. In this thesis, three types of the LFR will be introduced. They can be classified by the methods of making turns, which are decided by the constructions of the LFR. The first LFR is a tank [5]. This tank LFR equips two DC motors and two tracks on each side. This LFR can run very stable on the rugged ground. It is commonly seen as a moon rover. The picture is shown in Figure 1.1.

Figure 1. 1 ZUMO [5]

While the tank LFR has many advantages, it also contains disadvantages. For example, the radius of gyration of the robot is hard to figure out because the landing place of the track is not in a fixed location. This lack of a fixed location will cause the robot to make turns inaccurately. In order to increase the flexibility of the robot, the control law for a tank LFR is similar to the two-wheeled LFR, which is demonstrated in Figure 1.2.

Figure 1. 2 M3pi [5]

To make the robot stand, the two-wheeled LFR has a ball caster underneath the front, this construction is widely used by engineers [1]. The tank LFR and the two-wheeled LFR have a similar drive method. The speed deviation of the two wheels makes the robot turn. Typically, the two motors have to be controlled respectively. The robot does have two control boards to drive the two motors. Assuming that the control boards drive the motors respectively, the only control variable that can represent the relation between the motors is the position of the robot. Therefore, the controller has to give out two of speed signals. The question is how to calculate out those two signals. In fact, there is an easy way to solve this problem. It only controls the speed difference between the two motors that indirectly control the speeds of the two motors. For example, the robot will go straight before it receives the speed difference signal, and when it needs to make a turn, it means that the speed difference is not zero. So, the speed difference will work on the two motors in opposite ways. For instance, the speed difference will make the right motor slow down and make the left motor speed up at the same time, resulting in the motor turning right. This way, it makes the control operation simple because there is only one control variable that needs to be calculated.

It must be stated that even using the same mechanical system but a different control law, the controller design could be very different. The best one is to find out a way that describes the system and all variables' relationships clearly. Simultaneously, engineers have to decide that, sometimes, the unconsidered elements can be ignored. For example, frictions of motors, energy consumption of circuits, and unknown noises may exist but never can be determined for sure. Also, those never-sure elements can be estimated by using some methods, one of them is ADRC.

There is another kind of LFR that sets up with a steering engine in the front of the robot that can make the robot act like a real car [6]. It is shown in Figure 1.3.

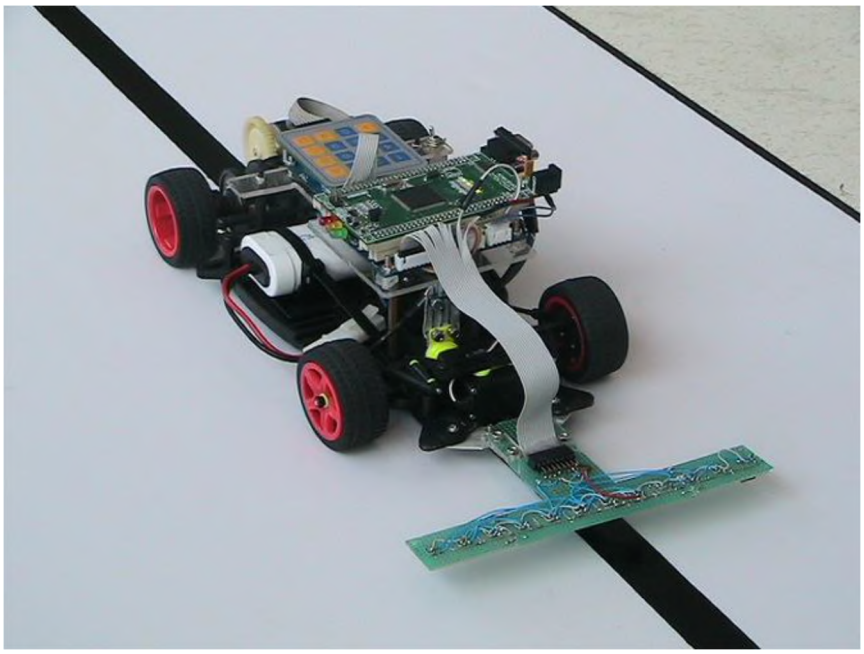

Figure 1. 3 LFR with Steering Engine

For this kind of LFR, the control signal is the angle of the steering engine. It means that usually, this kind of LFR cannot change the speed during the trip. So, the steering engine must react fast to go along with the running speed of the robot. Also, this kind of LFR cannot follow the right-angle line because the steering engine cannot turn 90 degrees. It can be seen from Figure 1.3 that the motor drives the rear two wheels and the sensors are set in front of the robot. Actually, no matter where the sensors are, the robot controller will keep the line in the middle of the sensors.

In addition, the sensor types that robots assemble can also be used to classify the LFRs. No matter what kind of sensors, they are all used to detect lines that are predesigned. For a variety of circumstances, the line could be a black tape, an electromagnetic field, and a GPS signal, or anything that can be detected by the cooperate sensors. Most LFR prefer photoelectric sensors because photoelectric sensors can detect the position of a black line, which is economical and practical. Typically, a sensor array consists of 6 to 8 sensors, and the black line can be drawn, or it can be a black tape. In this paper, we chose the QTR-8A photoelectric sensor array, which will be introduced in the sensor section of this paper.

Often, an LFR only has one PID controller in the position control. PID is the most common solution in motion control but it also has numerous defects: 1) the error computation, 2) noise degradation in the derivative control, 3) oversimplification and the loss of performance in the control law in the form of a linear weighted sum, 4) complications brought by the integral control [7]. All of these disadvantages will bring troubles to engineers. So, it is necessary for this thesis to introduce an advanced control design — Active Disturbance Rejection Control (ADRC). It will be shown later in this

thesis that, based on the experiments that have been done so far when the LFR encounters disturbances, the PID controller cannot reject the disturbances which the ADRC can.

In addition, the ALFR is made to do repetitive line-following operations as commonly seen in industries. In a factory, a production line consists of many automatic machines. Each of them finishes one part of the production, and they repeat their operation. In view of this repetitive operation, there should be a method to optimize its performance. One answer is to learn from experience. Iterative Learning Control (ILC) is a method that deals with the repetitive operations, and it is used in the ALFR in this thesis to make the ALFR better.

1.2 MOTIVATION AND SIGNIFICANCE

An LFR is an unimpressive robot that follows a line. However, it involves a variety of design principles and presents researchers with many possibilities. For example, its operation must be automatic, *i.e.* to track the line by itself. It annotates what is automation and symbolizes what is happening in daily life: transportation from one site to another via a predesigned path. This idea has been used as a Semi-Autonomous Mobile Robot [8]. Secondly, the robot is given a "brain" based on control theory to make self-corrections in following a path. Robots are designed to bring high productivity, safety, and convenience. As seen in the development of the technology for the autonomous car, LFR can play a role and offer a good idea. First, every autonomous car needs a path as a guide to the destination. Second, the running efficiency has to be considered. It contains

the speed control and the position control of the robot. Many LFRs, in the absence of advanced control design, have to run slowly because they cannot correct the deviation smoothly. Most LFRs use PID can work well, but PID has many restrictions. There is a need for the LFR to be more advanced. In the real world, for example, if a car is speeding on the road and a turn is coming, it has to slow down to avoid sliding. However, most LFRs cannot brake or brake well due not only to the limitations of the position control of the robot, but also to that of the speed control. Many robots' designs do not pay much attention to the motor speed control part. For this reason, an ALFR needs to be investigated not only for the ability to slow down but also to brake or even make the wheels run in reverse whenever it is necessary. Thus, ALFR is not simple as it looks, and it is a challenging real-world control system design problem. In this thesis, an ALFR was built for testing control designs intended for industrial productions or articles of daily use. This research on ALFR encompasses controller design, communication design, embedded system design, mathematical modeling, and MATLAB/Simulink validation.

The thesis is organized as follows. The introduction is provided in Chapter I, Chapter II gives out a brief background of the Active Disturbance Rejection Control and Iterative Learning Control. The construction and control problems of ALFR are described in Chapter III, and Chapter IV. Considering the control results and problem, Chapter V and Chapter VI give some solutions based on the mathematical models of ALFR and simulations.

CHAPTER II

CONTROL METHOD BACKGROUND

2.1 THE ADRC DESIGN PRINCIPLE

Before introducing ADRC, the question is, what does ADRC do? The answer is "Control". "Control" is not produced or created; it is a law that exists in nature. Thus, it can be seen that "Control" plays a significant role in our lives. What is control? Control is the behavior over time, and "Control" can be found everywhere, such as in factories, modern medicine, aeronautics, and even in economics, etc. "Control" is not only applied to dynamic systems, but it also operates in many unexpected domains. Because the notion of "Control" extends across different realms, it makes the study of "Control" meaningful and complicated at the same time.

The study of "Control" helps people understand the evolving process of an event. The purpose here is to figure out the relationship between every segment of the systems, which will help engineers obtain the desired results. In order to determine the causal relationship in the real world, mathematical models were used to describe the systems. By using mathematical models, engineers can obtain information that cannot be easily obtained by physical means. The study of control gives engineers a deep insight of the systems' construction.

However, the question is how to achieve the answers or results as desired. For example, before designing a controller, designers have to figure out which inputs directly or indirectly affect the outputs. In other words, what is the input? Is it the current, voltage, or other variables? The input signal may be one or more. The choice of these variables directly affects the controller design.

Open-loop control was found very early life. However, open-loop control cannot satisfy modern control requirements. The relations between inputs and outputs are usually nonlinear. In fact, the outputs are not to be controlled well in open loops because we do not know the exact transitive relation between inputs and outputs. As a result, the desired outputs are hard to obtain. Therefore, automatic control is normally achieved in closed-loops, as shown in various control design methods and theories, among which the proportional-integral-derivative (PID) design is especially prominent [4, 7, 9]. It goes without saying that PID has many advantages: PID has a simple construction, and PID can be inserted in most control systems. In addition, compared to other control methods, PID is error-based and more or less modeless and it is relatively robust. PID tuning has

been a bottleneck but there are many tuning methods available, including auto-tuning and self-tuning.

In order to deeply understand the problems from the practice, "there are three paradigms in control engineering: the industry paradigm, the model paradigm, and the disturbance rejection paradigm" [4]. It is necessary to figure out the nature of the "Control" science.

All in all, the weaknesses of PID are inherent in its error-based design principle. In general, error-based controls start with the error; then the controller generates the correction in the form of the input that goes into the plant. The key point is that the error comes first, and the correction follows. Another point is that PID relies heavily on the mathematical model of the system [10]. However, control is not only the problem of model. It is also the problem of information where we can get from the system [11]. Is there a way to improve this innate disadvantage of PID? A possible answer is ADRC as discussed below:

"This kind of disturbance rejection is deemed active because it doesn't wait for the disturbance to work its way through the physical process and cause significant changes in the state and output. [12]"

"It inherits from proportional-integral-derivative (PID) the quality that makes it such a success: the error driven, rather than model-based, control law; it takes from modern control theory its best offering: the state observer; it embraces the power of nonlinear feedback and puts it to full use; it is a useful digital control technology developed out of an experimental platform rooted in computer simulations. ADRC is

made possible only when control is taken as an experimental science, instead of a mathematical one. [7]"

As Figure 2.1.1 shows, in PID the feedback comes from the output and then subtracted from the reference to generate the error. The error goes through the controller to generate an input signal. Finally, the input drives the plant to produce the output. Apparently, there are three steps in this procedure, the purpose of which is to reduce the output error and guide it towards the reference *r*.

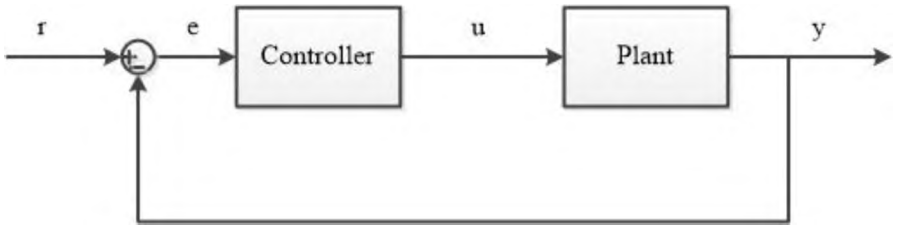

Figure 2.1. 1 PID Configuration

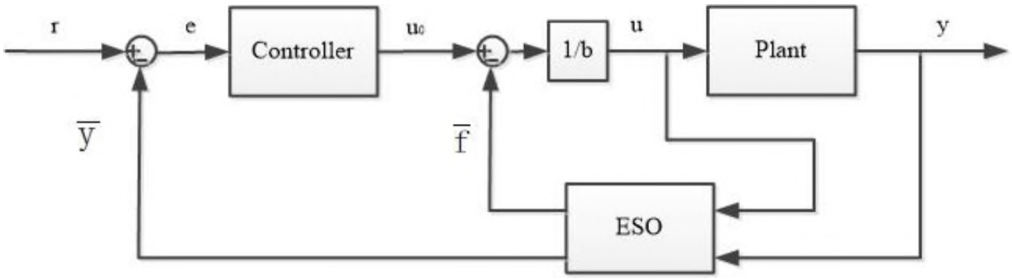

Figure 2.1. 2 ADRC Configuration

Figure 2.1.2 shows the ADRC configuration. As we can see, u_0 subtracts $\bar{f}$ [4, 7, 13], which is the total disturbance, to generate the input signal u. It means that before the new input signal u_0 takes effect, the disturbance has already been estimated and rejected somehow. The controller plays the similar role as in PID. The difference is that the use of

ESO greatly reduces the burden of the controller because the disturbance is estimated and rejected before it makes the output deviate from the setpoint. ADRC is controlling the process of the system [4]. An application that uses this idea is the South-Pointing Chariot (SPC) [12], which was created thousands of years ago. The core concept of SPC is not getting the feedback from the output, but calculating the disturbance and canceling it. Thus, the central problem turns into how to obtain the information of the disturbance, just like the SPC did. It must be pointed out here that $\bar{f}$ is the estimation of a particular disturbance, but also the effect of all disturbances of the system, including the internal and external disturbances of the system.

In a manner of speaking, ADRC is the enhanced version of PID with strong disturbance rejection ability. ADRC compensates for the disadvantages of PID, especially the passiveness, and establishes itself as the next generation industrial control solution. Central to ADRC is the tool that, based on the real-time information, estimates the uncertainties of the entire system. This estimation is obtained from the Extended State Observer (ESO). As demonstrated by Figure 2.1.2, ADRC will not work well if $\bar{y}$ and $\bar{f}$ are not accurately obtained. So, in implementations and tuning, the designer must be sure that the ESO is working correctly.

Here control system design branches out in two directions. On the one hand, we get the most precise system model as we can, analyze the system based on the model, and then proceed to design the controller. This method focuses on getting the precise mathematic model. Also, as we know, the ideal mathematic model cannot be achieved in reality. In industrial applications, "the model is not easily available in many engineering

problems. Even if it is available, the resulting control law could be too dependent on the accuracy of the model parameters and suffers poor robustness." [14]

On the other hand, ADRC provides an alternative to the model-based design in the real world, where there is no ideal mathematical model or controller. Instead, it focuses on the estimation of the total disturbance base on the real-time information. It does not mean ADRC is "model-free". If the model information is somehow given, it is better to apply all the given information of the system into the control design. Thus, we must try to obtain as much information as we can and then build up the mathematical model as precise as possible. We only use ESO to compensate for the uncertainties. This is the basic design principle of ADRC. It is profound because it reveals the commonality of all feedback control systems, where the plant can be transformed into the cascade integral form via one method or the other. Therefore, ADRC can be easily incorporated into other control design methods and used as convenient as the PID.

2.2 ITERATIVE LEARNING CONTROL

Iterative Learning Control (ILC) is a theory about storing the input and output sequence of a repetitive system and extract useful information from it. The ILC system operates repetitively over a fixed time interval and uses the stored signals to generate control actions, known as feed-forward, at the next run, assuming the robot runs under the same condition every time in both the external environment and the internal environment. Because ILC can work with feedback control, the control action will be

corrected each run and gradually improve over time. This correction series is the advantage of ILC. Because of this property, ILC is more or less modeless. In other words, ILC only needs little information about the system. No matter if the system is linear or nonlinear [15, 16].

On the other hand, the ILC system is premised on the fact that a repetitive operation and the external and internal disturbance condition will be the same in every run. The goal is to make the output perfectly track the ideal output after running a certain number of times. The ILC-based control the input signal update is shown in the following (2.2.1) [15, 16, 17, 18], where $u_k(t)$ is the input, γ is the tuning parameter, $e_k(t)$ is the error on trial k, and $u_{k+1}(t)$ is the input for the next trial.

$$u_{k+1}(t) = u_k(t) + d\gamma e_k(t)/dt \tag{2.2.1}$$

Existing LFRs mostly use PID control without ILC, because, generally speaking, most of LFRs do not repeat operations exactly. The LFR's objective is to follow the line, whether or not the line is visible or invisible. The robot itself does not realize it is following a line; its job is just keeping the line in the middle of the sensor array. To repeat an operation exactly, it needs a start and an end point as the reference. In our experiment, the robot is set up with a position sensor. Based on this position sensor, the robot is given the start point and the end point, which makes the repetitive operation possible in the more powerful ALFR. In particular, ALFR is advanced because: 1) the ALFR uses more advanced control methods to improve its performance in the presence of disturbances; 2) ILC gives ALFR memories and learning abilities. After completing a couple of rounds, ALFR will learn from the stored information to make the next run better. However, we must be careful: a poorly designed ILC can make the system

unstable, the stability of the system depends on convergence [15]. If the important signals are stored at each run, it is only reasonable to use them to construct a virtual line, based on which the speed of the robot can be better controlled at the next the run. For example, using the position information to construct a virtual line, smoother control actions and more accurate speed control of the robot can be achieved on the next run. The virtual line is not unique [19, 20]. This virtual line will enable the robot to run faster without excessive deviations. In other words, the robot not only will shorten the time of each run but also learn how to run more efficiently each time.

CHAPTER III

ALFR Construction

3.1 SENSOR

LFR can be assembled with many kinds of sensors, such as electromagnetic sensors and photoelectric sensors, *etc.*; the line can be visible or invisible. An electromagnetic line is invisible and can be detected by electromagnetic sensors; a black line is visible and can be detected by photoelectric sensors. First, to LFRs, line following is the top private. Actually, line following is used quietly in our lives, such as auto–driving forklifts that were invented many years ago, as well as flights auto-driving systems that have matured. Recently, many companies have started to design auto–driving cars. In the recent ten years, the technologies around auto–driving cars have developed very fast. Thus, line following is very common in life when we put attention on it. In line following, the key is how to find the line. Different types of sensors can

detect corresponding lines. Most of the time, the line does not exist; therefore, humans should design the line first. For example, in this paper, ALFR follows a line that is a black tape. Thus, according to the circumstance, engineers have to choose the appropriate sensors and "lines" to solve the problems of the line following in practical terms. Actually, there are many kinds of sensors that can detect the black line, like a camera. In order to reach the aim in an effective way, a photoelectric sensor has been selected and used in this research. Photoelectric sensors have the ability of mapping the position of the black line on the white ground.

This ALFR has two sensors, one is the QTR-8A (a sensor module produced by Pololu), and it is a reflectance sensor array that is shown in Figure 3.1.1.

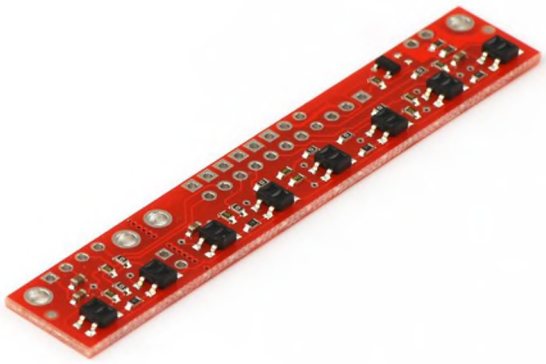

Figure 3.1. 1 QTR-8A [5]

QTR-8A is assembled under the front of the robot, and the interval space to the ground is about 0.125", which is the same as the recommended distance [5]. QTR-8A has eight Infrared Radiation (IR) emitter and receiver (phototransistor) pairs evenly spaced at intervals of 0.375" (9.525 *mm*). The detection range is about 2.75" (6.67 *cm*) wide. The

maximum recommended sensing width is 0.25" (6.35 *mm*). In this experiment, the black line's width is about 0.125"(3.175 *mm*). The QTR-8A is supplied by 5 *V*, and the outputs would be eight analog voltages ranging from 0 *V* to 5 *V*. When the sensor is over the white ground, the output is 0 *V*; otherwise, the output is 5 *V* when the sensor is over the black line. The operating schematic diagram is shown in Figure 3.1.2. As the picture shows, there is a MOSFET that can control the eight IR meters to turn it on and off. The purpose is to consider the power consumption. For this project, the MOSFET is always high voltage. In other words, those eight IR meters are always being turned on. The eight receivers are all independent, which means their outputs will not affect each other and they are all analog signals. The reason to choose this analogy sensor array is that it has accurate and stable output. This is very important to the experiment result because the result from the sensor array directly works on the performance of the robot. IR sensor has been used by LFR for a long time, and it has many practical features [3]. In addition, there are many ways to enhance the sensing quality. For example, the IR sensors connect with color sensors can shorten the time of the detection [21]. It is very important to get clear and accurate signals from the sensors. Thus, the filter for the sensors is necessary.

As shown in Figure 3.1.2, when the eight receivers receive light from the white ground, the receivers' circuits are connected, and then the outputs of voltage will be 0*v*. If the sensors are over the black line, it means there is no light reflection. If circuits are opened, the outputs of voltage will equal VCC.

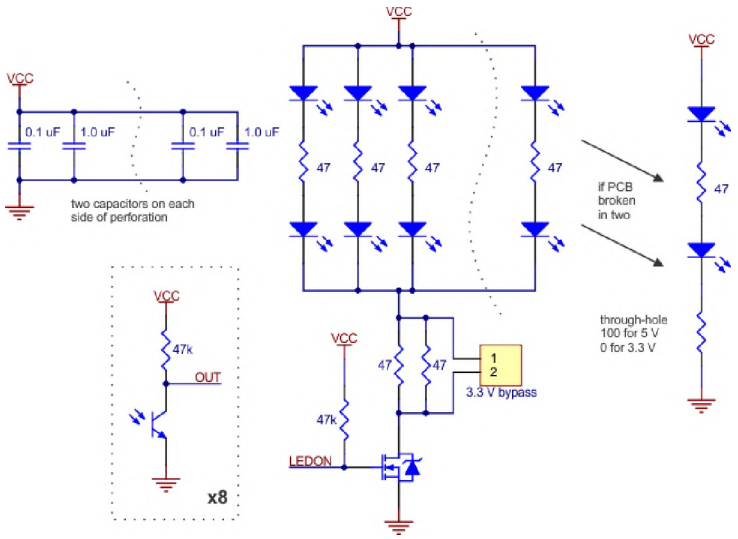

Figure 3.1. 2 Sensor Array Schematic [5]

In Figure 3.1.2, the eight IR emitters turn on when the robot is running, but those eight IR emitters and receivers will not be working perfectly because of the disturbance. From the view of physics, eight IR emitters and receivers must be different from each other. In other words, each voltage of the receivers will be different when the black line is respectively under each of them. Therefore, the eight analog signals need to be calibrated before running.

Pololu has provided a line-read formula that is shown in (3.1.1). It uses weight average to calculate the position of the line, which is a method commonly used by designers. There is a timer to count the time that the voltage goes up for each sensor. So, the timer's value will be 0 to 1000. During the calibration, the robot will scan the line to record the maximum value and the minimum value for each sensor. The maximum value indicates that the line is under the sensor, and the minimum value indicates that the sensor is over the ground. So, there are eight pairs of maximum and minimum values. For

0 to 7, they are to 0 or 1000. It depends on the timer's value. When the timer's value is smaller or equal to the minimum value, the corresponding value is 0. When the timer's value is bigger or equal to the maximum value, the corresponding value is 1000.

$$line\ position = \frac{0\times value0+1000\times value1+2000\times value2+\cdots}{value0+value1+value2+\cdots} \quad (3.1.1)$$

As above, before the robot starts running, the eight pairs of sensors all need to be calibrated because they are independent of on another. In order to make the robot calibrate itself, the robot would rotate to scan the line at the beginning. Then the sensors scan the black line to record the values of the eight sensors, which are from 0 to 1. Those values are precise because the samples are all analog signals. when the noise from the ground is large, this way cannot work well.

Actually, there is another way to calculate the position of the line. For example, the eight sensors are named from 0 to 7, which let the black line go from sensor0 to sensor7, and through ADC, with each writing down both the maximum values and minimum values for all the eight sensors. Then the maximum value minus the minimum value for each of the sensors can be calculated to determine the differential values, which is shown in (3.1.2).

$$differential\ value = Max - Min \quad (3.1.2)$$

When the robot is running on the black line, the eight values of the receivers are subtracted from their corresponding minimum values, respectively. The eight final results should be divided by the eight differential values, respectively, which is represented as (3.1.3). The sensors with the maximum quotient are over the black line. This is the law

that the robot gets the position of the black line. The calibration is done before the robot starts running.

$$calibrate\ value = \frac{sensor\ value - Min}{differential\ value} = \frac{final\ value}{differential\ value} \qquad (3.1.3)$$

In fact, the QTR-8A cannot precisely detect the position of the black line because of the internal and the external disturbances. The outputs from the ADC have a lot of errors. Sometimes, couple sensors will be read as 1, but the black line is not under them. Thus, the data sent from the QTR-8A needs to be processed before it is sent to the controller. The robot has a filter to delete those unreasonable results.

After the calibration, the eight final results are all between 0 and 1. However, in practice, the maximum value may not be the actual position because of the disturbances. Based on the physical propriety of the sensors, the eight sensors' voltages should be on a Para-curve. In order to simplify the problem, the parabola can be plotted through at least two points (values). The two points are not randomly picked (see Figure 3.1.3). The x-axis represents the sensor board length, which is from -3.5 to 3.5. When the x-axis values are fixed, they are$x_0 = -3.5, x_1 = -2.5, x_2 = -1.5, x_3 = -0.5, x_4 = 0.5, x_5 = 1.5, x_6 = 2.5, x_7 = 3.5$; Y-axis values can be collected from the sensors. The last step is how to find the position of the line. To plot out the Para-curve, the maximum value of y-axis values must be found. After that, the two y values, which are just nearby the maximum value, can be known. Then the Para-curve can be plotted out. The Para-curve function should be $y = -ax^2 + bx + c$, where c is 1, a and b can be solved by the given x and y. The top point of the Para-curve is $x = -\frac{b}{2a}$ $y = 1$. The position of the line can be estimated as $x = -\frac{b}{2a}$. This method is more accurate than the common one.

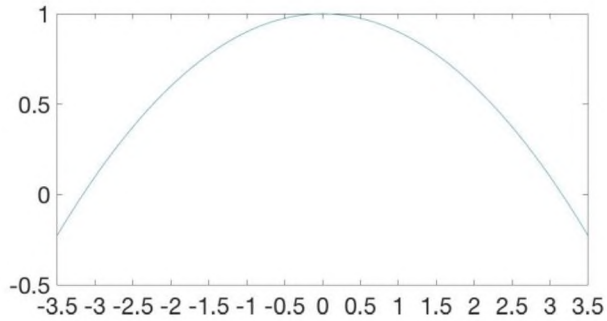

Figure 3.1. 3 Line Calculation

3.2 CONTROL BOARD

TMS320f28335 [22] is the product of the TEXAS INSTRUMENTS (TI). It is a high-performance Digital Signal Processor (DSP). Its control frequency is up to 150 *MHz*, and it includes many common modules, such as Pulse-Width Modulation (PWM), Analog to Digital Converter (ADC), Digital to Analog Converter (DAC), Capture Podule (CAP), Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), and Control Area Network (CAN), etc. which is a high-level tool and can satisfy most control requirements. The control card is shown in Figure 3.2.1.

Figure 3.2. 1 F28335 Control Card Release 1.0 [22]

This control card belongs to C2000™ products of TI. The C2000™ MCU Experimenter Kit was used in the experiment. It has 16 ADC channels inputs, 2 pins of powers 5 *V*, 8 pins of powers 3.3 *V*, 10 GND pins, and 47 commonly used GPIO pins. This experiment kit serves the control cards and allows designers to mount some little components onto the unused space. For instance, CAN and an extra power resource had been used in this control system.

The PWM module had been used as outputs on this experiment kit. It sends a PWM signal to the motor control board to control the speed of the motors. After some experiments, these PWM outputs were removed because of the strong interference and replaced by the CAN.

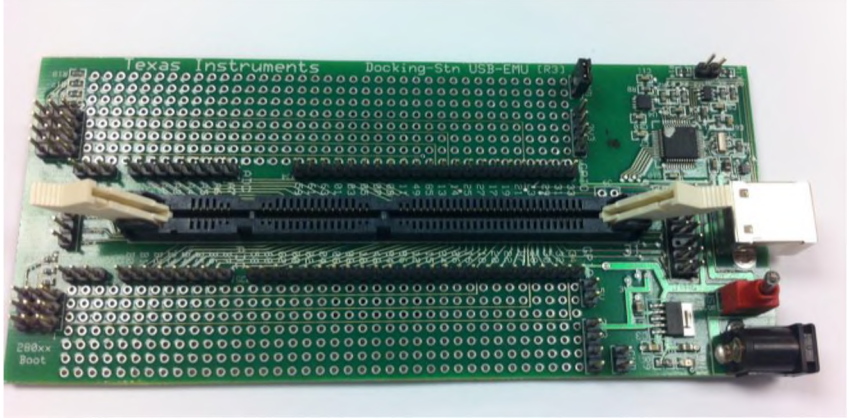

Figure 3.2. 2 C2000TM MCU Experimenter Kit [22]

For the line-follower control system, the ADC module must be used because the sensor array board needs 8 channels to transform 8 analog signals into the ADC module. Then the controller can read the position of the line.

3.3 MOTOR DRIVE

LFR has many types of constructions. Relatively, there are many control laws that are used in LFR. Their hardware constructions decide what kinds of control laws will be used.

The LFR, which was designed for this experiment, has a simple construction. The way of controlling the robot is simple, too. Most people control the two-dimensional placement of the robot [23] by controlling the motors. The main works are on motors and sensors. They are the central parts of this system. Easy build-up and low cost are the benefits of this simple construction robot. The LFR consists of the main body, two

wheels, two motors, sensor array, ball caster, battery, LFR control board, and motor control boards. The robot body is designed with two layers, which is shown in Figure 3.3.1.

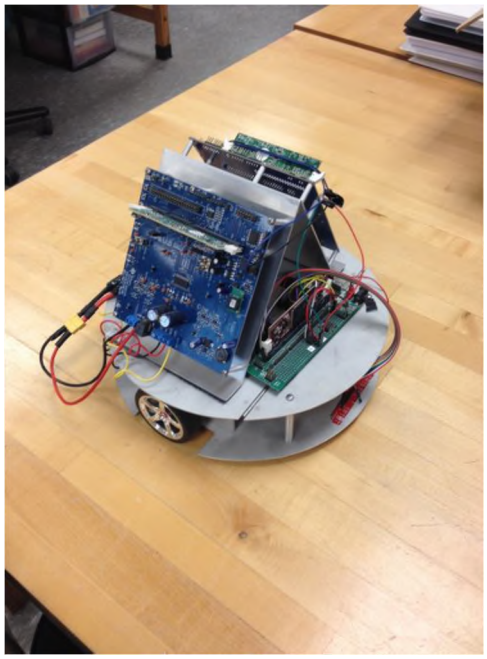

Figure 3.3. 1 ALFR

The substrate has two motors, which directly connect to the two wheels, one sensor array, and one battery. Because of the construction of the robot, the sensor array needs space between the robot and the ground. Thus, the weight of the robot should located mainly be concentrate on the back of the robot, as close to the caster as possible. This is very important because when the robot brakes or slows down quickly, the sensor array will easily touch down the ground because of the unbalance of the robot. It can cause a grievous error from the sensor array feedback to the controller. So far, the only thing done is fastening the battery on the back of the robot. The battery is 769 grams. It comprises about a quarter of the robot. The battery is shown in Figure 3.3.2.

Figure 3.3. 2 Line-follower Battery: Capacity: 5000 *mAh*, Voltage: 6S1P, 6 *Cell*, 22.2 *V*, Discharge: 45 *C* Constant, 55 *C* Burst, Weight: 769 *g* (including wire, plug & case), Dimensions: 144x52x58 *mm*

On the top layer, there are two motor control boards (Tms320f28069 [24]) and one ALFR control board (Tms320f28335). The LFR control board cooperates with the DRV8312-C2-KIT, which is demonstrated in Figure 3.3.4. This kit is used for the three-phase brushless DC motor. It can support up to 50 *V* and 7A. Also, it is supplied by 24 *V* to work. The battery can supply 22.2 *V*, through this experiment, we can see the kit works. The DRV8312-C2-KIT provides 5 *V* power to the control board (Tms320f28335). After some tests, it was found that when the DRV8312-C2-KIT is powering a motor, the voltage of the 5 *V* source would go down, then it cannot satisfy the requirement of the control board. Therefore, an external 5 *V* power resource had to be built for the ALFR.

Figure 3.3. 3 DRV8312-C2-KIT [24]

For normal LFRs, there is no controller for the motors. In other words, the motor control part is an open-loop control. If the LFR is small or light, it means there is not too much inertia to the motors. Then open-loop control can satisfy the requirements of line following; however, when the weight of the LFR is increasing or a lot of disturbance comes from the ground, the motor will work inaccurately. Additionally, the LFR needs to do some special actions, such as rotation, going forward, turning back, and braking immediately. All these actions give the motors challenges. Especially, if the robot is heavier, those actions are harder for the LFR, mainly because of the inertia. How about using advanced control theories in the motor control systems? Can they control the system better? The control theory PID has been used and tested in the experiments, but PID cannot perfectly solve some problems. For example, during the rotation test, PID couldn't make the wheel reverse very well.

The robot uses two brushless DC motors to drive. The motors' mode is BLY172S-24V-4000. The rated voltage, rated power, rated torque, rated speed is 24 V, 53 W, 18 $N{\cdot}m$, and 4000 rpm, respectively, with a torque constant of 5.81 $oz{\cdot}in/A$. The motor is shown in Figure 3.3.4.

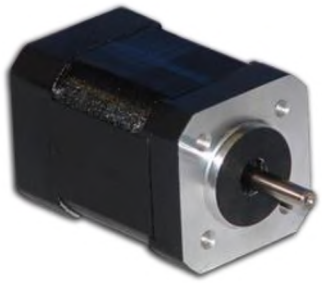

Figure 3.3. 4 BLY172S-24V-4000

As given the radius of the robot wheels, r is 1.25 inches. The motor constant torque is 5.81 $oz{\cdot}in/A$ and the driver board can support a maximum of 7 A current. Then we get

$$\frac{5.81\left(oz\cdot\frac{in}{A}\right)*7(A)}{1.25(in)} = 32.536(oz) \tag{3.3.1}$$

From above we know that the motor can support a maximum output of 32.536 oz, which is 927.3956 g, and the inertia of the robot is 158.09 $g{\cdot}m^2$. Without a doubt, the motors are able to carry this robot.

CHAPTER IV

ALFR CONTROL PROBLEMS

4.1 CONTROL OPERATING PRINCIPLE

Line following can be done in many ways. Based on the features of line following, six steps need to be followed by the LFR: determines the current location of the vehicle, find the path point closest to the vehicle, find the goal point; transform the goal point to vehicle coordinate, calculate the curvature and request the vehicle to set the steering to that curvature, update the vehicle's position [25]. Also, we can define the robot control system with two groups: horizontal decomposition and vertical decomposition [26]. ALFR is designed in horizontal decomposition. As mentioned above, the control law of ALFR is that the speed difference between the two motors controls the position of the robot. The actual speeds of the motors are as follows:

Right Speed = Initial Right Speed + Speed Deviation

Left Speed = Initial Left Speed - Speed Deviation

The initial right speed and initial left speed have the same velocity but opposite direction. The speed difference can be either negative or positive. The controller will calculate the speed difference based on the error of the robot position. By this method, it uses one signal to control the two motors. In the ALFR control process, the whole system is separated into two parts. It is shown in Figure 4.1.1.

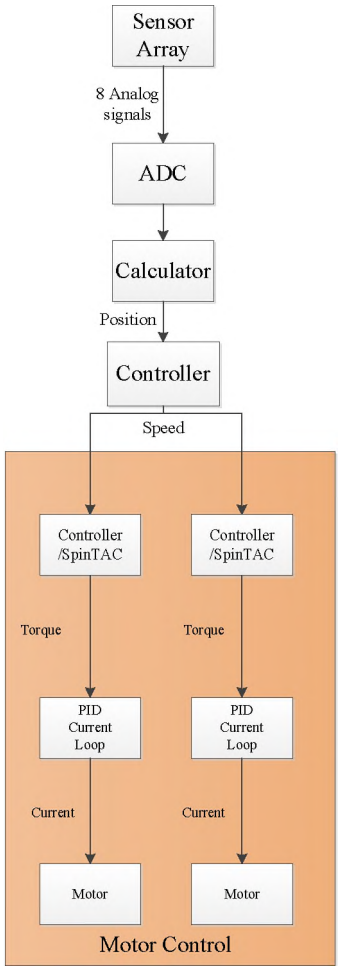

Figure 4.1. 1 Control Diagram of the ALFR

The first part is the ALFR position control; the second one is the ALFR speed control. When the robot receives the 8 analog signals, those signals go through ADC module and then are converted into 8 digital signals. The 8 digital signals must contain noises, but the worst is not the noise, it is the error from the sensor array. If the sensor array is distant from the ground, it will function incorrectly. The sensor array will read a large error. As the robot is designed, the sensor array should offer feedback values that are from 0 to 1. In fact, the sensor array sometimes gives values that are much larger than 1, which will mislead the robot to calculate the wrong position. In order to locate the line, the sensor needs to find out the peak point of the Para-curve. In other words, the robot needs to know the largest value among the 8 digital signals. If the robot takes a wrong peak point, the line position will be wrong. Before those digital signals go into the microchip, the signals that are larger than 1 must be filtered out. Then, it must find out the largest signal in the rest signals. It must make some judgments about whether this signal is larger than the previous one and the later one. Is this signal the edge point? In other words, is this signal the first or the last signal of the sensor array? All works above are done in the programming.

Typically, the LFR uses the PID controller to control the robot. In this thesis, both PID and ADRC will be applied to the ALFR. The entirety of position information passes through the controller, and then a series of control signals will be generated as the speed of the motor. Those outputs from the controller are not the real speed reference for the two motors. They are the speed difference between the two motors. In order to guide the motor to follow a curved line, physically the two motors on each side of the robot must have different speeds. The problem to be answered is guiding the robot to make a correct

turn. This action must be associated with the position of the line, which can be positive and negative. So, the input signal (speed difference) can be positive and negative as well. Ultimately, because the position is positive and negative, the robot can follow the line. For example, the left motor speed plus the speed difference. Meanwhile, the right motor speed minus the speed difference. The speed difference causes the two motors have a speed deviation, which equals two times the speed difference. When the speed difference changes from positive or negative, the left and right motors increase or decrease in speed. From there, the robot only requires an initial speed for both motors.

The next part of controlling the motor is to follow the speed reference with the input signal from the ALFR control board. In order to control the robot well, the control process is separated into two sections. Based on the "bottom-up" vision, obviously, motor control is important in the ALFR control process [27]. In the experiments, two control theories have been used to control the motor: PID and ADRC. Of course, some LFRs do not require a controller for their motors. As we know, that is an open-loop control. Open-loop control has many disadvantages, such as low fidelity and no auto-correct. Due to this reason, open-loop control is very unstable, making it necessary to design a controller for the motor control.

However, controllers cannot solve all problems. Some problems that the controller cannot solve are due to the hardware restrictions, such as the motor property. In the experiments, two motors had been used which have a high speed of 2000 *rpm/v*. Typically, these kinds of motors are used in race games because this motor cannot work well in a low-speed range. The question of why the LFR needs to run in such low speed ranges arises. There are two reasons: at first, the motor control kit can supply the motor

with a voltage from 0 V to 55 V, the max speed of the motor is 110000 *rpm*, which is too high for an LFR. If we need the motor to run 100 *rpm*, the voltage should be 0.05 V. In fact, the control kit does not have the control resolution at 0.05 V. A high speed from the start can cause the robot to easily lose controlled. With the motor loaded, the motor still has a high speed to the ALFR. So, this situation is created by the hardware. The solution should come from the hardware, which is changing the motor or changing the control kit. At the end, both the control kit and motors were changed to the F28035 control card with the DRV8312 control kit, and Nema Size 17 BLDC Motor. All of them are the technical grade, satisfying the low-speed requirement and being controllable.

Motor control is a very common problem throughout the industry. No matter what kind of motors, the controller design is already a mature technique. As discussed above, the ALFR motor part consists of F28335, DRV8312 kits, and a BLDC motor.

4.2 HARMONY

Another problem that arises is determining how to ensure that the whole system has a stable function. In the ALFR control process, the prime problem is having a good performance of the motors, the motor control becomes the priority of the ALFR. For the ALFR, the first step is controlling the motor's speed and then controlling the robot's position. If the motors do not work well or do not follow the speed reference, the ALFR control process will become more complicated and confused. Two variables need to be controlled: one is the position, the other one is the speed. Obviously, those two sections

will affect each other. If the motors do not work precisely, the position control cannot be worked out. Thus, the position control is built on the speed control. For example, in the experiments, the ALFR always running out of the line. One consideration is the sensor array board is relatively short to the ALFR. In result, the rotation of the robot is limited to 30 degrees. Consequently, the ALFR cannot accept a large speed difference between the two motors. Otherwise, the line will be out of the sensors' range. Similar to normal control problems, it readily has an uncertain speed at the beginning because the sensor has a start error. Therefore, the black line easily runs out of the sensor's range, we must be sure that the sensor array can be used for this robot. The diameter of the robot is 28 centimeters, and the radius of the wheel is 3.25 centimeters, which leads to two possibilities: one of the wheels is running, the other one is holding. This way takes the longest time to run out of the sensor's range. Another way is that the robot rotates, which takes the shortest time compared to the former. The longest arc length of the sensor's range is 14.66 centimeters. Assuming that one of the motors is running at 100 *rpm*, the speed of the wheel is 34.0339 centimeter per second. The run-out time should be 0.4308 seconds, meaning that the response time should much less than 0.4308 seconds. In fact, the sampling time is set up as 2000*Hz*. Obviously, to see that the robot can reject the error about 860 times before the ALFR runs out of the black line. On the other hand, the ALFR can respond 430 times with the rotation situation. Thus, the control system should work well if the control frequency is high enough, even though the sensor's range may be short.

4.3 CALIBRATION

In this experiment, another problem that commonly occurred was during the calibration, which is a matter of this control process. The robot calibrates the sensorsc and then the robot gets ready to follow the line. In fact, two requirements must be followed by the robot: sensors' calibration and robot position initialization. In order to achieve these two goals, the robot must move its body and give the sensor enough time to scan the line. In the experiments, the robot scans the line two times in one complete calibration. Theoretically, the robot only needs scan the line one time with the speed at 100 *rpm*. The condition of the ground is changing at all times. Therefore, the robot needs to save an average data as the reference for each sensor. Each sensor has both maximum and minimum outputs of voltages when it is on the white ground and the black line, respectively. In the experiments, the ALFR will scan the line six times by rotation. Secondly, for the initial position, the line should be right in the middle of the robot when it starts. An easy way to do this is to make the line in the middle of the robot when it is placed on the ground. If the robot is programmed and allowed to rotate to the left for a constant amount of time, the robot will also rotate to the right for the same period until it returns to the original position. After all this preparation, the robot is ready to run.

4.4 MOTOR CONTROL PROBLEMS

In the rotation, there are two problems: one is changing the direction of the rotation, and the other one is the launch of the motors.

The change of direction is a challenge to the controller. If there is a large speed difference between the left and right wheels, the robot is easily out of control. Most LFRs do not have controllers for the motors. The LFR imports voltage into the motors if the motor is working stably. Some of the LFRs do not consider this problem probably because they are lightweight. In practice, if the LFRs do not have a large rotational inertia, and the motors can easily take care of the rotational inertia, probably the motor controller can be ignored. In this experiment, the motor control part is an important section since the motor is not strong enough to ignore the robot's weight.

PID is the classical control tools that is widely used by engineers. At the beginning of this experiment, the motors were set up with a PID controller. However, the inertia of the robot is overloaded to the motors. In the end, the PID controller cannot control the motors very well. In most of the experiments, the motors will be out of control. Even the various parameters of a PID controller have been tested and calculated many times, and the motors still didn't work well. So, it is not a tuning problem, but it is the limitation of PID. PID is easy to use and has restrictions. Proportional control can make the output track the reference very fast; however, it can cause the system to be unstable by increasing the gain. Integral control can eliminate the steady-state error, but it also delays the output. Differential control will amplify the disturbances and cause the control system break down. PID cannot work for this system. It depends on the actions of the robot. For the motors, it consists of three steps to change the motor running direction: slowing down, stopping, and speeding up. Speeding up is the only thing that the PID can do well. So, if the proportional gain is small, the control signal will never catch up with the desired control signal. And if the proportional gain is very large, the overshoot will

come, and the noise will be amplified, too. As a result, the PID has a limited control bandwidth, which cannot satisfy the requirement of the speed direction change. In order to solve this problem, PID was replaced by ADRC, which has strong ability of disturbance rejection.

For the motors, if the speed deviation is small, for example, the right motor speed is 100 *rpm*, and the desired speed is -100 *rpm*, then the speed deviation is 200 *rpm*. PID can handle this change. However, increasing the speed deviation to 1000 *rpm* will be an overload to PID. So, it is a test of disturbance rejection ability. In the experiment, PID and ADRC have been used. And in comparison, the results showed that ADRC has a larger range of speed deviation rejection. In PID control, increasing the proportion gain can lightly enhance the disturbance rejection ability, but it will make the system unstable. In ADRC control, the disturbance rejection ability does not depend on the high gain because of the estimation of the disturbance. Therefore, the disturbance is the main issue in the control process. Disturbance rejection control is an effective method to realize the control objective. As a result, ADRC can work well in the rotation that PID cannot.

The last problem is that sometimes the motors cannot start. But through the experiment result analysis, two ways were found and used to solve this problem. The first one is the initial properties setting of the motors' systems. It includes inertia, friction, initial control bandwidth, and current limited, etc. However, in the simulation, these were not considered. In practice, the robot must be set up with those properties to make the robot work better. Therefore, some mathematical work has done to figure out those necessary data. The second one is finding out the right initial position of the motors. In the speed control, Sensorless Trapezoidal Control of BLDC Motor [28] is used. The

sensorless algorithm consists of the torque and the back EMF of the motor. Because of some unknown reasons, this sensorless algorithm does not work well at the beginning. Before starting the motors, the rotors' positions must be correct. This method has to run the motors to find out the rotors' position, and then it switches to ADRC control. The result is that the motors have to run during this motor launch time. If not, the sensorless control will not work correctly. So, the robot is programmed to run for a short time at the beginning to initialize the position of the motors.

CHAPTER V

MODELING AND CONTROL DESIGN

5.1 MATHEMATICAL MODEL OF THE ALFR POSITION CONTROL

Before designing a control system for a robot, we must understand its construction. The more information is given to the controller, the better the performance. In the LFR control system design, the main objective is to command the robot to follow the black line on the ground. For this to happen, the inner speed loop must make the motor speed track its reference. This inner-outer loop design is commonly known in the industry as cascade control system. The inner loop controls the motor speed and the outer position loop keeps the line in the middle of the sensed view window. For the position loop, the input is the speed difference, and such a robot is called by some the drive difference robot. The output is the position that is detected by the sensors. The sensors are in the front of the robot and face the ground. The control design principle is quite clear: motor voltage controls the motor speed on each side, and the speed difference

between the two motors controls the position orientation, or direction, of the robot. Figure 5.1.1 below describes the construction of the LFR:

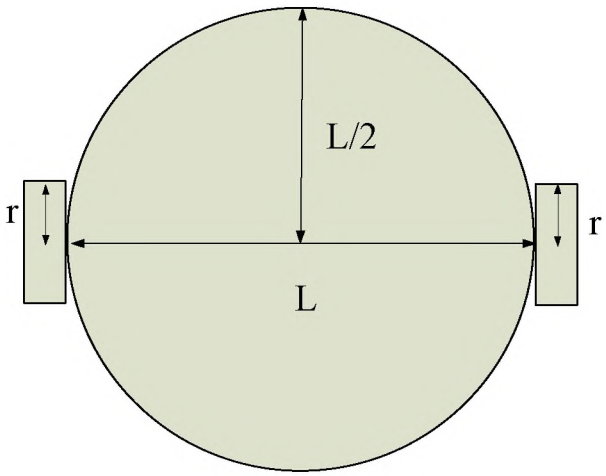

Figure 5.1. 1 Top view of the AFLR

The notions are as follows: r is the radius of the wheels, L the diameter of the robot body, θ_R, θ_L the angle of the wheels, α the rotational angle of the robot, and p the position values, which are read from the sensor array, p_1 the previous position, and p_2 the position of the line. Furthermore, p_2 is considered the future position. The distance from the midpoint of the sensor array to the center of the robot is a. The rotational process is represented in Figure 5.1.2. We use the relative position between the line and the robot, which is easier than the Cartesian frame because the x-axis and y-axis are deleted, and the

controlled variable is p (sensor output value) which is the position of the robot self [29, 30].

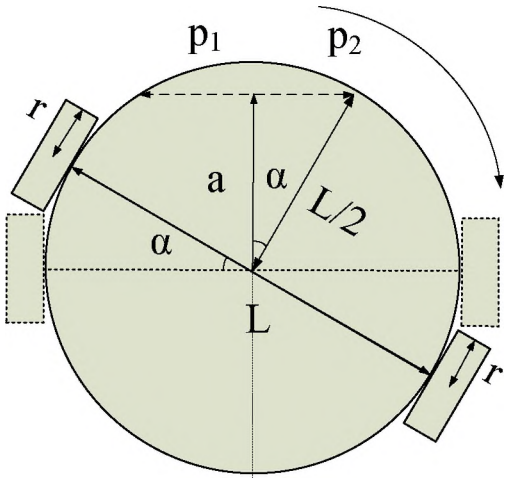

Figure 5.1. 2 The rotational dynamics of the robot

From the Figure 5.1.2, the angular position of the robot and the speed difference between the two wheels can be seen as [30, 31].

$$\frac{(\dot{\theta}_R - \dot{\theta}_L)r}{L} = \dot{\alpha} \tag{5.1.1}$$

where the relationship between the wheel speed $\dot{\theta}_R$ and $\dot{\theta}_L$ in rad/sec and their corresponding values in revolution per minute (*RPM*) are

$$\frac{\dot{\theta}_R}{2\pi} \times 60 = SpeedR\ (RPM) \tag{5.1.2}$$

or

$$\dot{\theta}_R = \frac{\pi}{30} \times SpeedR \tag{5.1.3}$$

and

$$\frac{\dot{\theta}_L}{2\pi} \times 60 = SpeedL\ (RPM) \tag{5.1.4}$$

$$\dot{\theta}_L = \frac{\pi}{30} \times SpeedL \tag{5.1.5}$$

Here SpeedR and SpeedL are the right and left wheel speed in *RPM*, respectively. On the other hand, the rotational angle α can be translated into the position quantity. The translation equation is [31]:

$$\alpha = arctan\frac{p}{a} \tag{5.1.6}$$

Based on (5.1.3), (5.1.5), and (5.1.6), we have

$$\frac{\frac{\pi}{30}(speedR - speedL)r}{L} = \frac{d(\arctan(\frac{p}{a}))}{dt} \tag{5.1.7}$$

The equation (5.1.7) describes a dynamic system where the speed difference e.

$$SpeedR - SpeedL = e \tag{5.1.8}$$

can be seen as the input. The equation (5.1.7) can be rewritten as

$$\frac{\frac{\pi}{30} \cdot e \cdot r}{L} = \frac{d(\arctan(\frac{p}{a}))}{dt} \tag{5.1.9}$$

Integrating both sides of (5.1.9) we have

$$\int \frac{\frac{\pi}{30} \cdot e \cdot r}{L} = arctan(\frac{p}{a}) \tag{5.1.10}$$

or

$$p = a \cdot tan \int \frac{\frac{\pi}{30} \cdot e \cdot r}{L} \tag{5.1.11}$$

Finally, equation (5.1.11) gives us the output of the plant, corresponding to the sensor measurement, and it can be further simplified, assuming $\left(\frac{p}{a}\right)$ is small, as

$$arctan\left(\frac{p}{a}\right) \approx \frac{\frac{p}{0.9}}{\frac{L}{2}} \tag{5.1.12}$$

We do this approximation because the length of the sensor array (*max p*) is about 2.7 inches, and the arc length over the sensor array is about 3.0 inches, thus the scale should be 0.9. The parameters of the robot are all given by (5.1.13), (5.1.14), (5.1.15), and (5.1.16). After adding an uncertain disturbance, the final estimation equation is (5.1.17), where p is the position and e is the speed difference.

$$a = 5.2\ inches\ \approx 0.13208\ m \tag{5.1.13}$$

$$max\ p = 2.7\ inches\ \approx 0.06858\ m \tag{5.1.14}$$

$$r = 1.25\ inches\ \approx 0.03175\ m \tag{5.1.15}$$

$$L = 11\ inches\ \approx 0.2794\ m \tag{5.1.16}$$

$$p = 0.0014962 \cdot \int e \tag{5.1.17}$$

5.2 MATHEMATICAL MODEL OF THE ALFR SPEED CONTROL

The construction of the robot is shown as Figure 5.2.1. Since the robot is moving in the horizontal plane, its kinetic energy function [32, 33] can be shown as

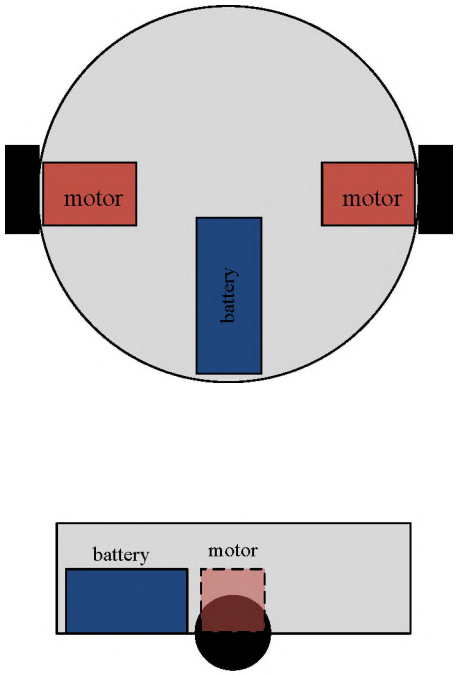

Figure 5.2. 1 The ALFR inside construction

$$K = \frac{1}{2} I \dot{\alpha}^2 + \frac{1}{2} m_t \dot{x}^2 \tag{5.2.1}$$

where I is the total inertia of the robot [33, 34, 35, 36, 37], α is the angle of the robot, m_t is the total mass of the robot, and x is the displacement of the robot. Also, I consists of I_m, I_{Ba}, and I_c, which is shown as below.

$$I = 2I_m + I_{Ba} + I_c \tag{5.2.2}$$

In (5.2.2), I_m is the inertia of the motors, I_{Ba} is the inertia of the battery, I_c is the inertia of the robot. The inertia of the motor is shown as

$$I_m = \frac{1}{12} m_m (a_m^2 + b_m^2) + m_m d^2 \tag{5.2.3}$$

where d is the distance from the center of the motor to the center of the robot, and m_m is the mass of the motor. The a_m and the b_m are the heights and widths of the motor respectively, $m_m = 0.432\ kg$, $a_m = 0.02\ m$, $b_m = 0.03\ m$, and $d = 0.07\ m$. After calculation, $I_m = 0.0022\ kg{\cdot}m^2$. Then the inertia of the battery is

$$I_{Ba} = \frac{1}{12} m_b (a_b^2 + b_b^2) + m_b d^2 \tag{5.2.4}$$

where m_b is the mass of the battery, $m_b = 0.768\ kg$, a_b and b_b are the heights and widths of the battery respectively, $a_b = 0.05$ m, $b_b = 0.05$ m. The distance between the center of the battery and the center of the robot is the same as the motors' which is $d = 0.14\ m$. After calculations, $I_{Ba} = 0.0154\ kg{\cdot}m^2$. Lastly, the inertia of the robot body is

$$I_c = \frac{1}{2} m_c R_b^2 + m_c b^2 \tag{5.2.5}$$

In this equation, m_c is only the mass of the robot and R_b is the radius of the robot. And $m_c = 1.568\ kg$, $R_b = 0.14\ m$, $\mathrm{b} = 0.28\,m$. So $I_c = 0.13829\ kg{\cdot}m^2$. Now the total inertia of the robot is

$$I = 2I_m + I_{Ba} + I_c$$

$$= 0.0044\ kg \cdot m^2 + 0.0154\ kg \cdot m^2 + 0.13829\ kg \cdot m^2$$

$$= 0.15809\ kg \cdot m^2$$

As we discussed above, the kinetic function can describe the direction of the robot, so the direction of the robot can be expressed by the angular difference between the two wheels.

$$\alpha = \frac{r\theta_1 - r\theta_2}{2R_b} \tag{5.2.6}$$

In (5.2.6), θ_1 and θ_2 are the angles of the two wheels respectively and r is the radius of the wheel, which is 0.0325 m. In order to get the angular speed of the robot, we have

$$\dot{\alpha} = \frac{r}{2R_b}(\dot{\theta}_1 - \dot{\theta}_2) \tag{5.2.7}$$

$$x = \frac{r\theta_1 + r\theta_2}{2} \tag{5.2.8}$$

$$\dot{x} = \frac{r}{2}(\dot{\theta}_1 + \dot{\theta}_2) \tag{5.2.9}$$

Combining all equations above and put them into (5.2.1), then the kinetic energy K is obtained as

$$K = \frac{1}{2} I \left[\frac{r}{2R_b}(\dot{\theta}_1 - \dot{\theta}_2)\right]^2 + \frac{1}{2} m_t \left[\frac{r}{2}(\dot{\theta}_1 + \dot{\theta}_2)\right]^2 \tag{5.2.10}$$

In order to establish the mathematic model, equation (5.2.10) has to be transformed into the Euler Lagrange function. Before that, equation (5.2.10) needs to be changed to

$$K = \dot{\theta}_1^2 \left(\frac{Ir^2}{8R_b^2} + \frac{m_t r^2}{8}\right) + 2\dot{\theta}_1\dot{\theta}_2 \left(\frac{m_t r^2}{8} - \frac{Ir^2}{8R_b^2}\right) + \dot{\theta}_2^2 \left(\frac{Ir^2}{8R_b^2} + \frac{m_t r^2}{8}\right) \tag{5.2.11}$$

$$A = \frac{Ir^2}{8R_b^2} + \frac{m_t r^2}{8} \tag{5.2.12}$$

$$B = \frac{m_t r^2}{8} - \frac{Ir^2}{8R_b^2} \tag{5.2.13}$$

The total energy in the system is $L(q, \dot{q}) = T(q, \dot{q}) - V(q, \dot{q})$ [31, 32], which is the difference between the kinetic energy and the potential energy of the robot. Since in this system, the potential energy is 0, the total energy L is the same as the kinetic energy

K. Then the Lagrange equation can be shown as $\frac{d(\frac{dK}{d\dot{\theta}})}{dt} - \frac{dK}{d\theta} = \tau$ [32, 35], where $\frac{dK}{d\theta}$ is 0 and τ is the input torque. Based on the parameters of the robot, it can be shown that A is 0.0014874, B is -0.00064919, and

$$\frac{d(\frac{\partial K}{\partial \dot{\theta}_1})}{dt} = 2A\ddot{\theta}_1 + 2B\ddot{\theta}_2 \tag{5.2.14}$$

$$\frac{d(\frac{\partial K}{\partial \dot{\theta}_2})}{dt} = 2A\ddot{\theta}_2 + 2B\ddot{\theta}_1 \tag{5.2.15}$$

$$\begin{bmatrix} 2A & 2B \\ 2\mathrm{B} & 2\mathrm{A} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{5.2.16}$$

In (5.2.16), τ_1 and τ_2 are the motor torques for the left and the right wheel, respectively. Ignoring the internal friction and heat losses, the state space model of the system can finally be written as:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 2A & 2B \\ 2\mathrm{B} & 2\mathrm{A} \end{bmatrix}^{-1} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{5.2.17}$$

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 415.2635 & 181.2457 \\ 181.2457 & 415.2635 \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{5.2.18}$$

5.3 CONTINUOUS AND DISCRETE EXTENDED STATE OBSERVER IMPLEMENTATION

ESO is a very important part of ADRC and it plays a significant role in the control design process. ESO provides the estimations of disturbance which directly affect the performance of the robot so that the controller can cancel it in a manner. ESO not only

gives out the estimations of the disturbance, but also the filtered outputs and their derivatives. Addition, for practical implementation, the discrete form of the ESO has to be derived. These topics are discussed in this section.

First of all, there are many methods to get the discrete version of ESO from the continuous form. In this paper, the discrete ESO will be obtained by using the zero-order holder (ZOH) method. The Line-follower is a first order system. Therefore, the ESO will be a second order system [13]. The general differential equation of a first order is:

$$\dot{y} = f(y, d, t) + bu \tag{5.3.1}$$

where u is the input signal, $f(y, d, t)$ is the total disturbance, and d is the external disturbance [13]. The next step is to transform (5.3.1) into state-space form:

$$\dot{x} = Ax + Bu + H\dot{f} \tag{5.3.2}$$

$$y = Cx + Du \tag{5.3.3}$$

where $x_1 = y, x_2 = f$, and

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} B = \begin{bmatrix} b \\ 0 \end{bmatrix} H = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$C = [1 \quad 0] D = [0 \quad 0]$$

here $\dot{f}$ represents the unknown. With these matrixes, the next step is to build the ESO in the form of:

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ \hat{y} &= C\hat{x} + Du \end{aligned} \tag{5.3.4}$$

where the estimation of x is $\hat{x}$, and the estimation of y is $\hat{y}$. The output estimate function $\hat{y} = C\hat{x} + Du$ can be incorporated into the input function $\hat{x} = A\hat{x} + Bu + L(y - \hat{y})$ and (5.3.4) can be rewritten as

$$\dot{\hat{x}} = [A - LC]\hat{x} + [B - LD, L]\begin{bmatrix} u \\ y \end{bmatrix} \tag{5.3.5}$$

Let $L = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$, where the elements of L are solved by the following equation where ω_o is the ESO bandwidth

$$|sI - (A - LC)| = (s + \omega_o)^2 \tag{5.3.6}$$

which leads to

$$L = \begin{bmatrix} 2\omega_o \\ {\omega_o}^2 \end{bmatrix} \tag{5.3.7}$$

For example, the ESO of the ALFR position control can be established based on (5.1.17). The differential equation of (5.1.17) is:

$$\dot{p} = 0.0014962 \cdot \text{ e} \tag{5.3.8}$$

where p is the displacement of the sensor array, e is the speed difference between the two wheels. Then p is rewritten as the output y, and e is rewritten as the input u.

$$\dot{y} = 0.0014962 \cdot u \tag{5.3.9}$$

Based on (5.3.4), the extended differential equation is:

$$\begin{aligned} \dot{z}_1 &= z_2 + b \cdot u + l_1(y - z_1) \\ \dot{z}_2 &= l_2(y - z_1) \end{aligned} \tag{5.3.10}$$

here z_1 is the estimation of y, z_2 is the estimation of all disturbances f, and b is 0.0014962. l_1 equals to $2\omega_o$, l_2 equals to ${\omega_o}^2$, and ω_o is the ESO bandwidth. u is the input signal and y is the output signal. Finally, based on (5.3.5), the ESO is:

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \begin{bmatrix}-2\omega_o & 1\\ -{\omega_o}^2 & 0\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} + \begin{bmatrix}0.0014962 & 2\omega_o\\ 0 & {\omega_o}^2\end{bmatrix}\begin{bmatrix}u\\ y\end{bmatrix} \tag{5.3.11}$$

The ESO of the ALFR speed control can be established based on (5.2.18). Each motor has the identical ESO. Therefore, only one ESO is shown below as an example. The differential equation of one motor is:

$$\begin{bmatrix}\ddot{\theta}_1\end{bmatrix} = \begin{bmatrix}415.2635 & 181.2457\end{bmatrix}\begin{bmatrix}\tau_1\\ \tau_2\end{bmatrix} \tag{5.3.12}$$

If τ_1 is from the right motor, and τ_2 is from the left motor. The τ_2 can be regarded as a disturbance of the right motor. Then we can rewrite the equation as:

$$\begin{gathered}\dot{z}_1 = z_2 + b_1 u_1 + l_1(y - z_1)\\ \dot{z}_2 = l_2(y - z_1)\end{gathered} \tag{5.3.13}$$

where z_1 is the estimation of y, z_2 is the estimation of all disturbances f, u_1 is τ_1 and b_1 is 415.2635. l_1 equals to $2\omega_o$, l_2 equals to ${\omega_o}^2$, and ω_o is the ESO gain. τ_2 is estimated by z_2 which is the total disturbance estimation. The extended state space equation is:

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \begin{bmatrix}-2\omega_o & 1\\ -{\omega_o}^2 & 0\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} + \begin{bmatrix}415.2635 & 2\omega_o\\ 0 & {\omega_o}^2\end{bmatrix}\begin{bmatrix}u\\ y\end{bmatrix} \tag{5.3.14}$$

The following step is finding the discrete form of the continuity equations, such as (5.3.4), (5.3.5), and (5.3.6). The corresponding discrete form [38] of (5.3.2) and (5.3.3) is:

$$\begin{aligned} \hat{x}(k+1) &= \Phi\hat{x}(k) + \Gamma u(k) \\ \hat{y}(k) &= H\hat{x}(k) + Ju(k) \end{aligned} \tag{5.3.15}$$

Based on (5.3.14), it is easy to get the corresponding discrete form of (5.3.4), it is:

$$\begin{aligned} \hat{x}(k+1) &= \mathit{\Phi}\hat{x}(k) + \mathit{\Gamma} u(k) + L_p(y(k) - \hat{y}(k)) \\ \hat{y}(k) &= H\hat{x}(k) + Ju(k) \end{aligned} \tag{5.3.16}$$

The corresponding discrete form of (5.3.6) is

$$|zI - (\mathit{\Phi} - \mathit{\Phi} L_c H)| = (z - \beta)^2 \tag{5.3.17}$$

where z is instead of s, $\mathit{\Phi}$ is instead of A, H is instead of C. Specially $\mathit{\Phi} L_c$ is instead of L, and $\beta = e^{-\omega_o T}$. Where ω_o is the gain of the observer, and T is the sampling time.

As mentioned above, ZOH has been used to build the discrete function of the ESO. According to the ZOH method:

$$\mathit{\Phi} = e^{AT} \rightarrow \sum_{k=0}^{\infty} \frac{A^k T^k}{(k)!} \tag{5.3.18}$$

$$\mathit{\Gamma} = \int_0^t e^{AT} d\tau B \rightarrow \sum_{k=0}^{\infty} \frac{A^k T^{k+1}}{(k+1)!} B \tag{5.3.19}$$

$$H = C \tag{5.3.20}$$

$$J = 0 \tag{5.3.21}$$

For instance, the ALFR position control has the extended state space coefficient as:

$$A = \begin{bmatrix} -l_1 & 1 \\ -l_2 & 0 \end{bmatrix} \tag{5.3.22}$$

$$B = \begin{bmatrix} 0.0014962 & l_1 \\ 0 & l_2 \end{bmatrix} \tag{5.3.23}$$

$$C = [1 \quad 0] \tag{5.3.24}$$

According to the result above, we bring the matrix A, B, and C into the matrix Φ, Γ, and for the values of the variables in matrix L_c. Then we can solve it based on (5.3.16).

$$\Phi = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{5.3.25}$$

$$\Gamma = \begin{bmatrix} 0.0014962T \\ 0 \end{bmatrix} \tag{5.3.26}$$

$$L_c = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix} = \begin{bmatrix} 1-\beta^2 \\ \frac{(1+\beta)^2}{T} \end{bmatrix} \tag{5.3.27}$$

Based on the discrete equation [38]:

$$\begin{aligned} \hat{x}(k+1) &= \left[\Phi - L_p H\right]\hat{x}(k) + \left[\Gamma - L_p J, L_p\right]u_d(k) \\ \hat{y}(k) &= [I - L_c H]\hat{x}(k) + [-L_c J, L_c]u_d(k) \end{aligned} \tag{5.3.28}$$

The new discrete state $\bar{A}$, $\bar{B}, \bar{C}, \bar{D}$ can be obtained as follows:

$$\bar{A} = \begin{bmatrix} -1-2\beta & T \\ -\frac{(1+\beta)^2}{T} & 1 \end{bmatrix} \tag{5.3.29}$$

$$\bar{B} = \begin{bmatrix} 0.0014962T & 2+2\beta \\ 0 & \frac{(1+\beta)^2}{T} \end{bmatrix} \tag{5.3.30}$$

$$\bar{C} = \begin{bmatrix} \beta^2 & 0 \\ -\frac{(1+\beta)^2}{T} & 1 \end{bmatrix} \tag{5.3.31}$$

$$\bar{D} = \begin{bmatrix} 0 & 1-\beta^2 \\ 0 & \frac{(1+\beta)^2}{T} \end{bmatrix} \tag{5.3.32}$$

CHAPTER VI

ALFR DESIGN VALIDATION

6.1 PID SIMULATION

The ALFR has been applied to two control theories: PID and ADRC. PID and ADRC were used to control both the ALFR's position and the ALFR's speed. In order to compare the ADRC with the PID, they were separately used in the ALFR. The first one is the PID-PID control mode, which means that the ALFR position control and the ALFR speed control both use the PID controller. Based on the equation (5.1.17), the position control simulation model was designed in Figure 6.1.1.

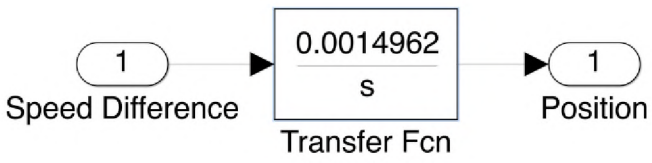

Figure 6.1. 1 Position Control Model

In Figure 6.1.1, the speed difference between the two motors is the input signal of the speed control loop. The output is the position of the ALFR. The PID control simulation was built in Figure 6.1.2.

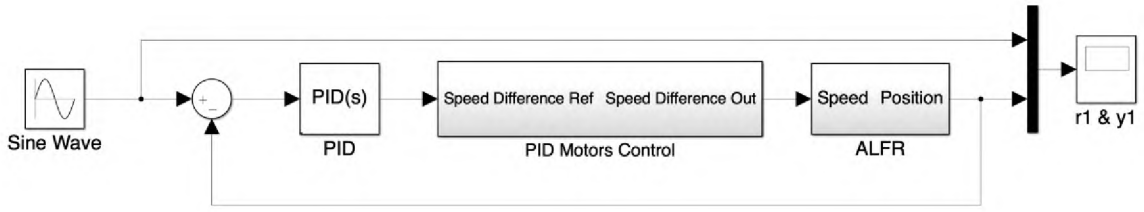

Figure 6.1. 2 PID Position Control

The reference signal is a sine wave. As mention above, this sine wave is designed as a virtual black line on the ground. Actually, the PID controller is a proportional controller because the plant is similar to an integrator without any disturbance. If there exist steady-state error, for example, the robot input is added with a disturbance, and then a PI controller will be used in the system. The output results y_1 and the reference r_1 were shown in a scope.

In practice, the reference signal is 0. When the robot correctly follows the line and then the output y_1 should be 0. That means the line is in the middle of the sensor array. In the simulations, the reference signal cannot be a 0. Fortunately, the position of the line and the position of the sensors are relative. Thus, in the simulations, there is a virtual line, and this virtual line can be designed as a sine wave. Mathematically, the difference between the reference and the sine wave is the input error. Therefore, we can make the input error to be a sine wave to simulate that the ALFR is running on a sine wave black line. Theoretically, they have the same results. In addition, the sine wave can make the

test results cover most running conditions for the ALFR. Obviously, the ALFR mathematic model approximates to an integrator. Thus, the controller is proportion control and the controller gain is 668.35984494 because the gain of the plant is 0.0014962. The result was plotted in Figure 6.1.3.

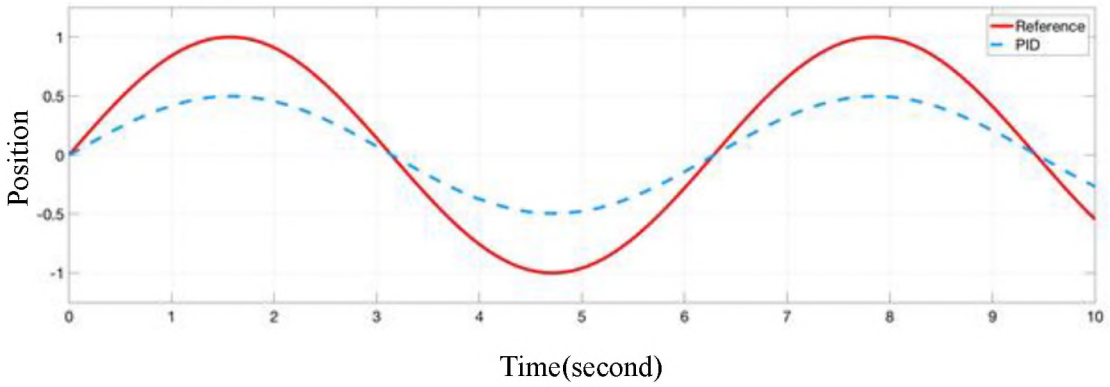

Figure 6.1. 3 PID Position Control Result

From Figure 6.1.3, the problem is the output cannot catch the reference. In order to prove the proportional controller can work well for this plant, the reference was changed to be a step signal. As the Figure 6.1.4 shows, there is no steady state error, and the proportional control is good enough for this position control system. The raising time can be shortened by increasing the value of the controller gain K_p. As increasing the PID controller gain K_p to 66835.984494, the error can be shortened in Figure 6.1.5.

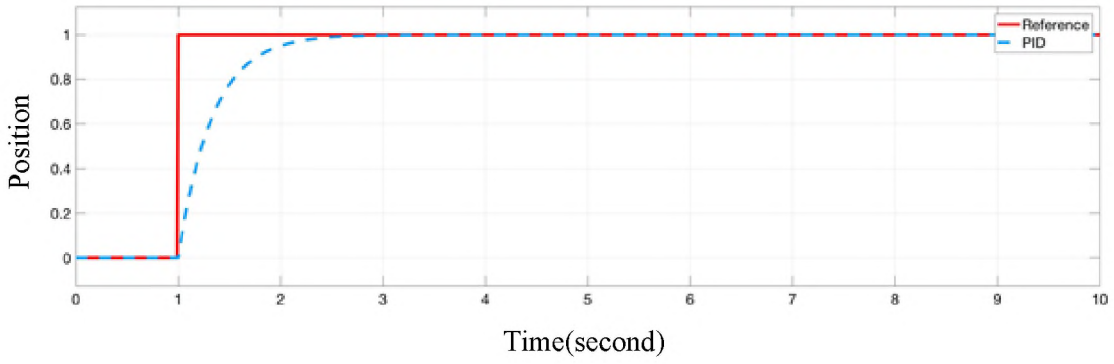

Figure 6.1. 4 PID Position Step Response

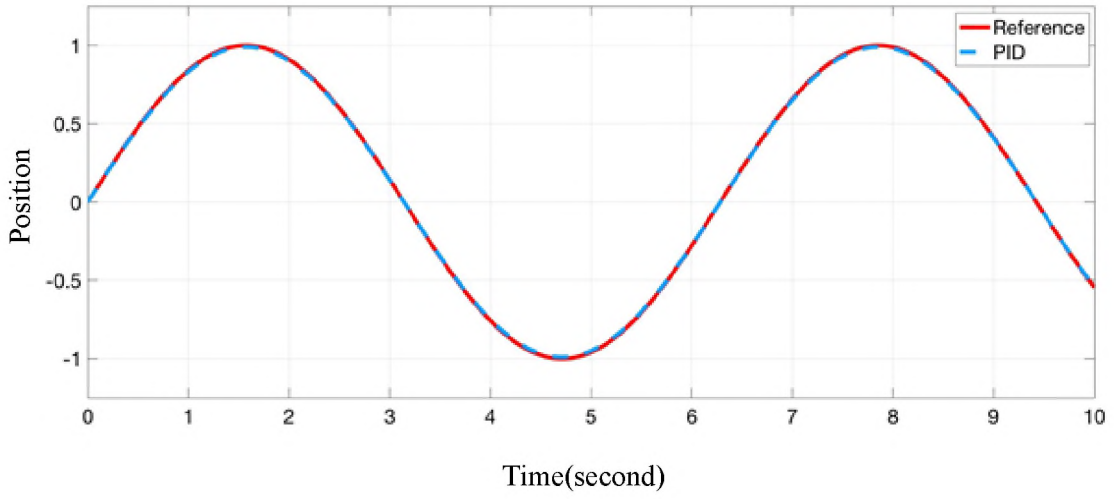

Figure 6.1. 5 PID Position Control Result

Theoretically, the output position can infinitely approach to the reference by increasing the proportion gain K_p. However, in practice, the proportion gain cannot be infinitely increased because it makes the system unstable. One of the reasons is the disturbance since will be amplified as the gain is increased. So, in the experiments, the robot can never have ideal results like those in Figure 6.1.5. For the motor speed loop, the

simulation diagram is represented in Figure 6.1.6, and it also uses the proportional controller.

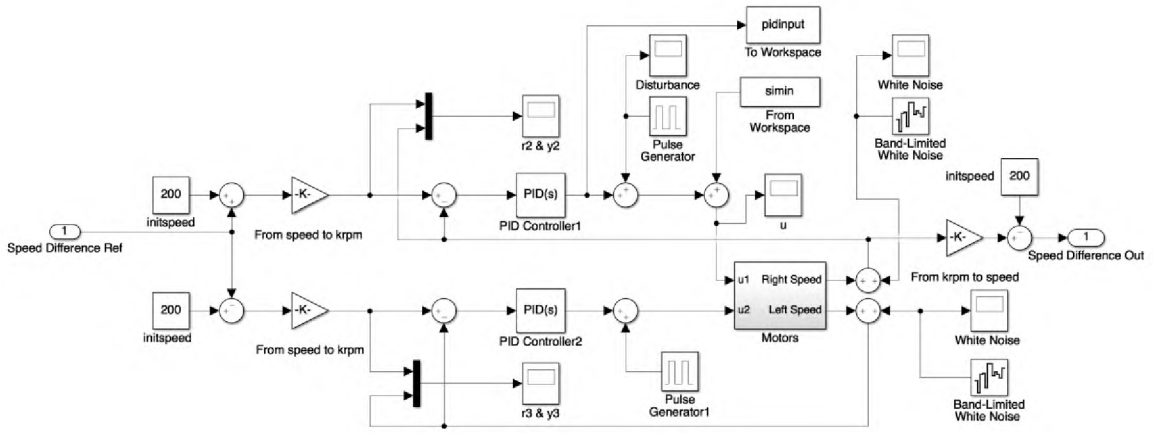

Figure 6.1. 6 PID Speed Control

As the Figure 6.1.6 shows, the reference signal for the speed loop comes from the position loop, which is the input signal of the ALFR position control. This import reference signal represents a half of the speed difference between the two motors with the unit *round per second*. This unit comes from the programming rule. In the programming, the position control board gives out a speed difference signal with the unit *round per second*. The speed difference adds an initial speed as the final speed of the motor. In practice, the initial speed makes the robot run straight. When the robot notices that the line is not in the middle of the robot, then a speed difference will be added to the both motors' initial speed. In Figure 6.1.6, the initial speed equals to 200 revolutions per second, but in the speed control, the unit of the reference signal in the programming is *krpm*. Therefore, the reference has to translate into *krpm*. The translate coefficient is 0.0001047. At the same time, the speed difference is positive to the right motor and

negative to the left motor due to the reason that the speed difference makes the right motor and the left motor have different speeds to make a turn. That is the reason why the reference speed difference is a half of the two motors' speed difference.

One speed loop has two inputs and one output. As mentioned above, the mathematic model of the speed loop can be treated as an integrator. The two inputs add together and go through the integrator. Based on (5.2.19), u_1 is the input signal of the right motor; u_2 is the input signal of the left motor. The $Gain_1$ is 415.2635 and the $Gain_2$ is 181.2457. The left motor speed loop is same as the right motor except the two input signals reversed. The simulation diagram is shown in Figure 6.1.7.

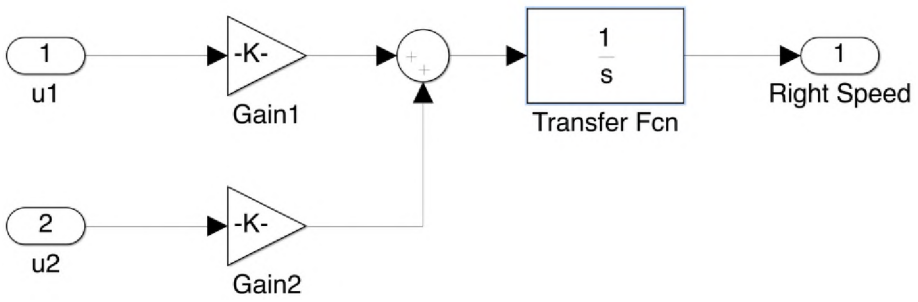

Figure 6.1. 7 ALFR Speed Control

The speed loop also used a proportional controller where the proportional gain is 150 and the result is shown in Figure 6.1.8. In Figure 6.1.8, the outputs follow the reference smoothly since the blue line is very clear. It means the proportional control is good for the speed loop. There is no oscillation and overshoot.

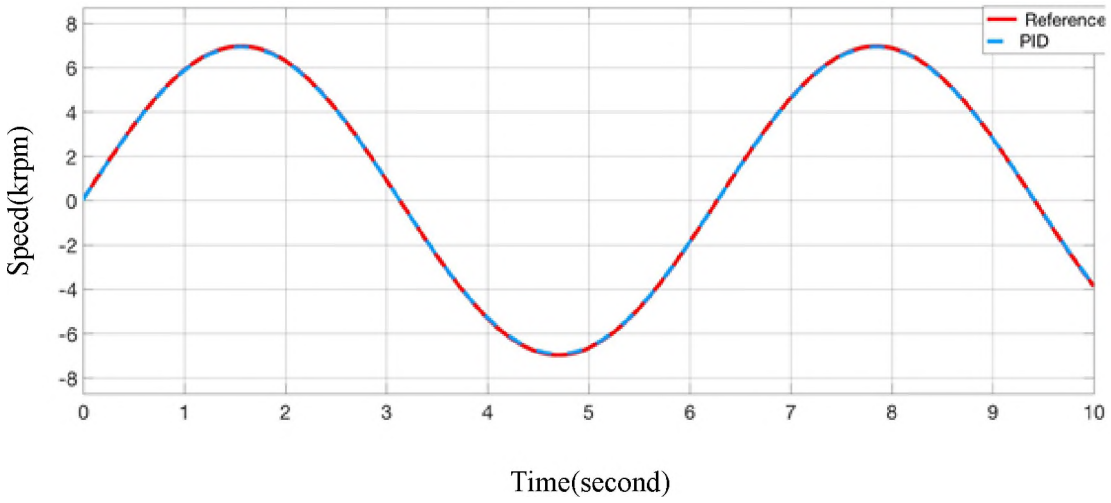

Figure 6.1. 8 PID Speed Control, the proportional gain is 150

In practice, control signals always have noise, and this is important. The simulations should include disturbances, and disturbances come from everywhere. So far, those disturbances can be classified as internal and external disturbances. The internal disturbances mostly come from the internal physical properties of the system. The external disturbances come from the line condition, the ground condition, and unknown force of the environment, etc. The width of the line will affect the sensor detection results. As well, the color of the ground will affect the sensor detection results, too. However, the input signal is the key to the plant. Usually, the input variable will be amplified after passing the plant, so as disturbances. Therefore, the next step is simulating the ALFR working with the disturbances.

6.2 ADRC SIMULATION

Many control theories were found in the past 100 years. The common or the final objective of most control systems is disturbance rejection. A strong disturbance rejection control theory is wanted in industries to replace the PID. ADRC is a control tool that focuses on rejecting all the disturbances of the systems, and it has been applied for tens of years to prove that ADRC is a feasible method. Therefore, ADRC was applied into the ALFR and the simulation diagram is illustrated in Figure 6.2.1.

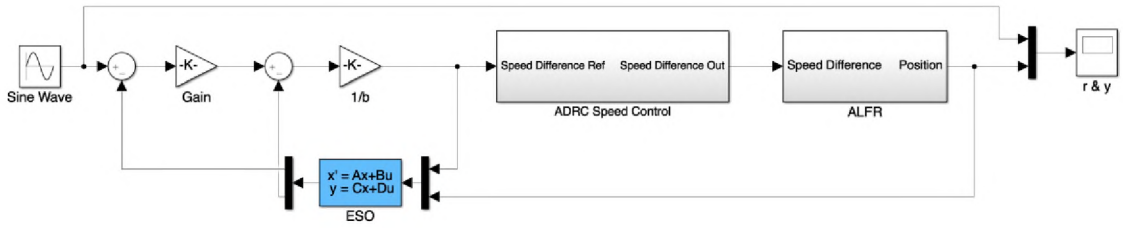

Figure 6.2. 1 ADRC Position Control

At this time, ADRC has been applied into the robot position control and the speed control. Both of the two control systems are first order control systems, and they are similar to an integrator. In Figure 6.2.1, the Proportional Gain is tuned as 100, and the $1/b$ is 1/0.0014962. ω_o is the gain of the ESO, and usually the ESO bandwidth is four or five times of the controller bandwidth. Thus, ω_o is 500 [39], and there is no disturbance. Therefore, ω_o does not need to be very large because ω_o relate to the performance of the disturbance estimation. The result is in Figure 6.2.2. Without the disturbances, PID and ADRC have similar results in this system. Therefore, it is necessary to add the disturbances into the simulations.

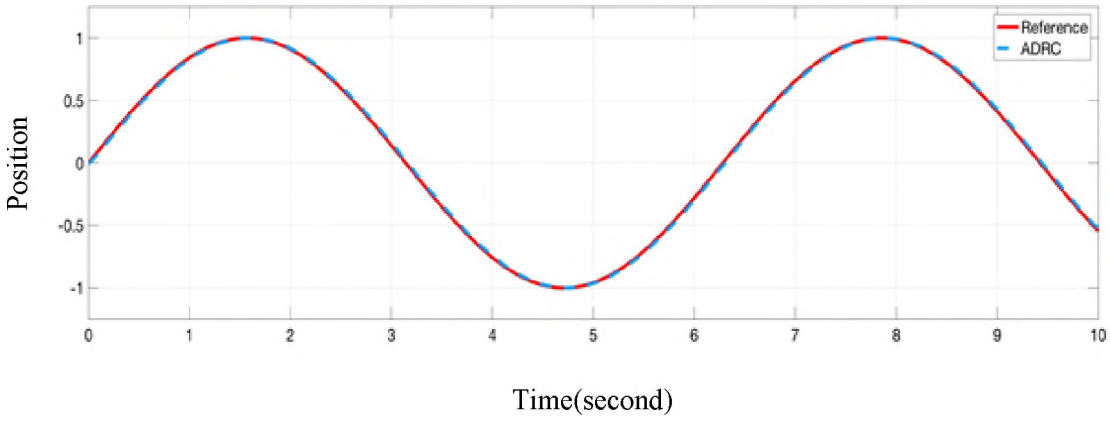

Figure 6.2. 2 ADRC Position Control, the observer gain $\omega_o = 500$, and the proportional gain = 100

In this project, the valuable problem is the robot speed control. The two motors affect each other in the line following process. The relationship between them can be seen as they are helping and disturbing each other. The ADRC speed control simulation diagram is in Figure 6.2.3.

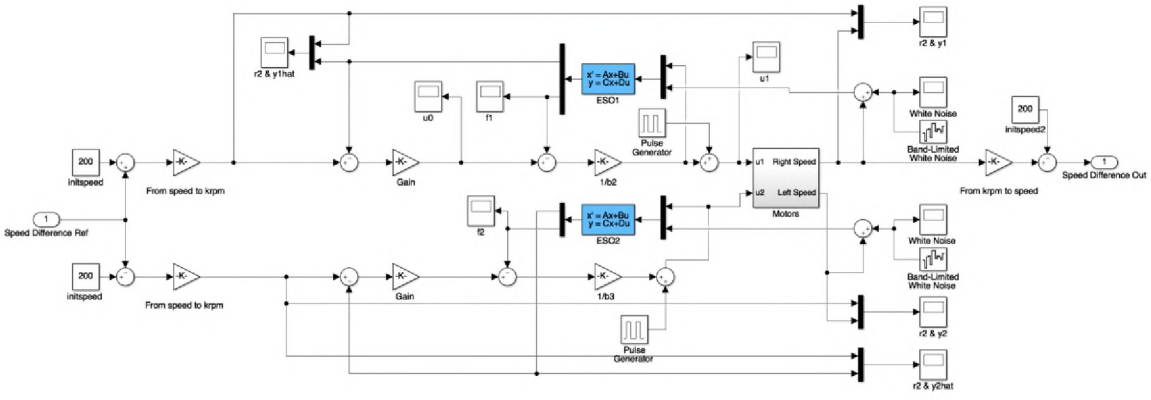

Figure 6.2. 3 ADRC Speed Control

In the ADRC speed control loop, the two of speed loops have the same parameters. The proportional gain is 10000, and the observer gain of the ESO ω_{o2} is 20000. The scope r_2&y_2 shows the result in Figure 6.2.4.

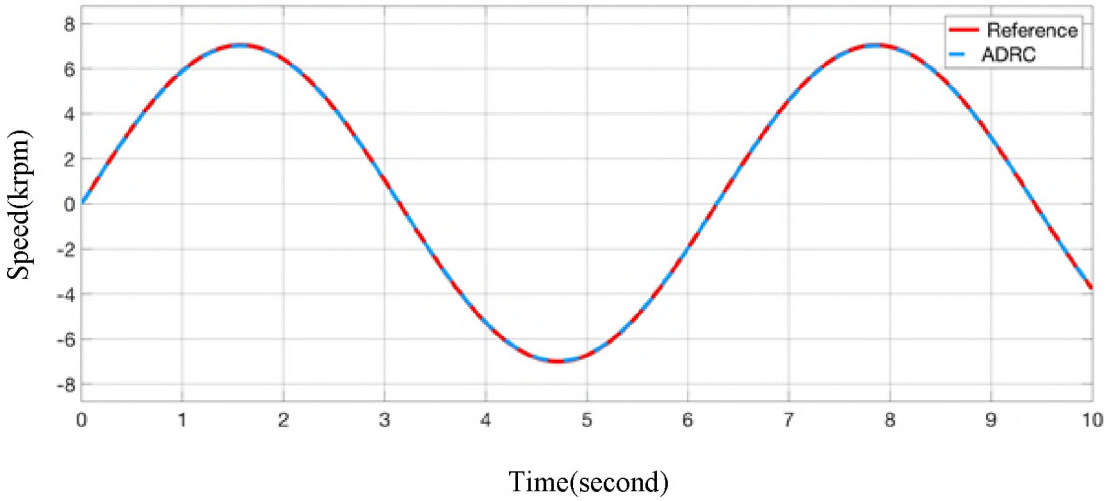

Figure 6.2. 4 ADRC Speed Control, ω_{o2} = 20000, proportional gain = 10000

6.3 WHITE NOISE TESTS

So far, there is no doubt that PID and ADRC controllers can control this robot system very well under no a disturbances condition. However, the final objective of this paper is to apply control theories in practice. It is impossible to have an ideal system without any disturbance. The disturbance test is very important. It can directly demonstrate the properties of the controllers to engineers. White noise was used to test the controllers' performance. Also, this white noise simulates the friction of the robot. The noise power is 10^{-10} and the frequency is 1000 *Hz*. The amplitude of the white noise

is around 0.004, which equals 1/1000 of the output amplitude. This noise has been added to the PID simulation and the ADRC simulation. The results are shown in Figure 6.3.1 and Figure 6.3.2.

When the white noise is added to the outputs, the result of the Proportional control is still stable. One reason is that the noise is much smaller than the output, so the effect of the noise is slight. Another reason is the model of the system is not complicated. It is an integrator, which is a very stable system among control systems. The result of the ADRC control is also good.

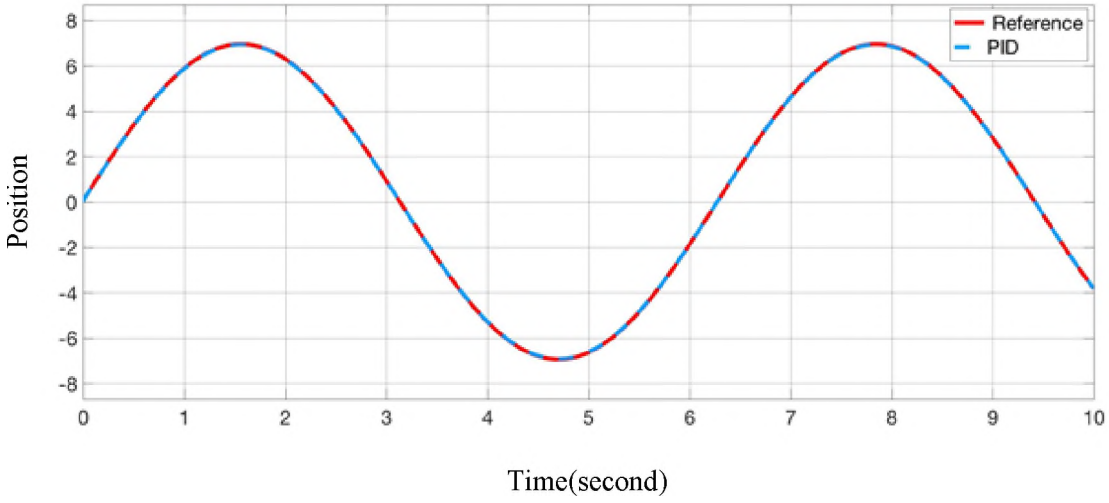

Figure 6.3. 1 PID Speed Control with white noise

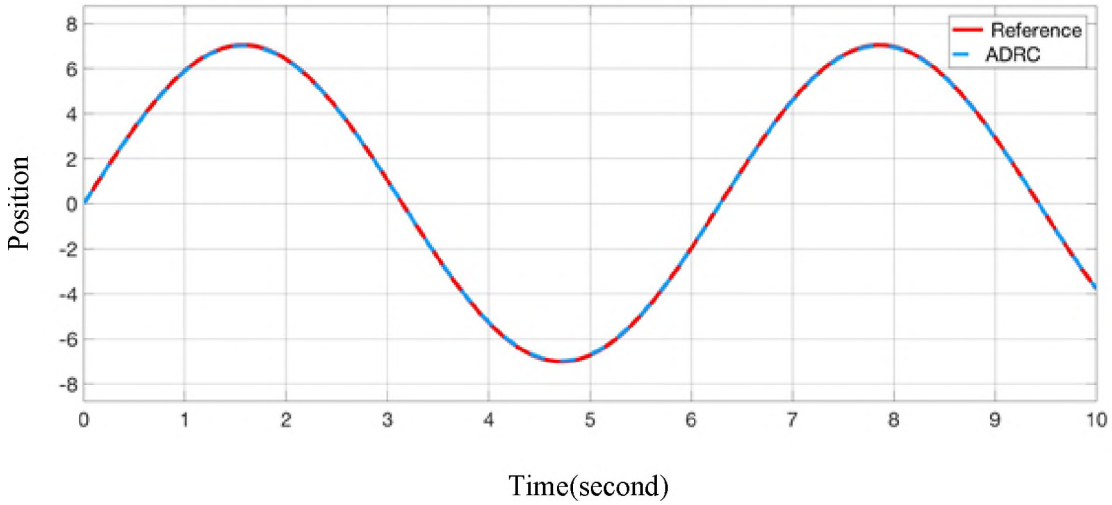

Figure 6.3. 2 ADRC Speed Control with white noise

The output results of PID and ADRC do not have an obvious difference but the input signals of PID and ADRC are very different. The input signals of the PID and ADRC are represented in Figure 6.3.3. The ADRC result is much better than the PID's. The input signal is important to the system. A clean input signal will make the system more "healthy". In other words, a hash of input signal will damage the system physically.

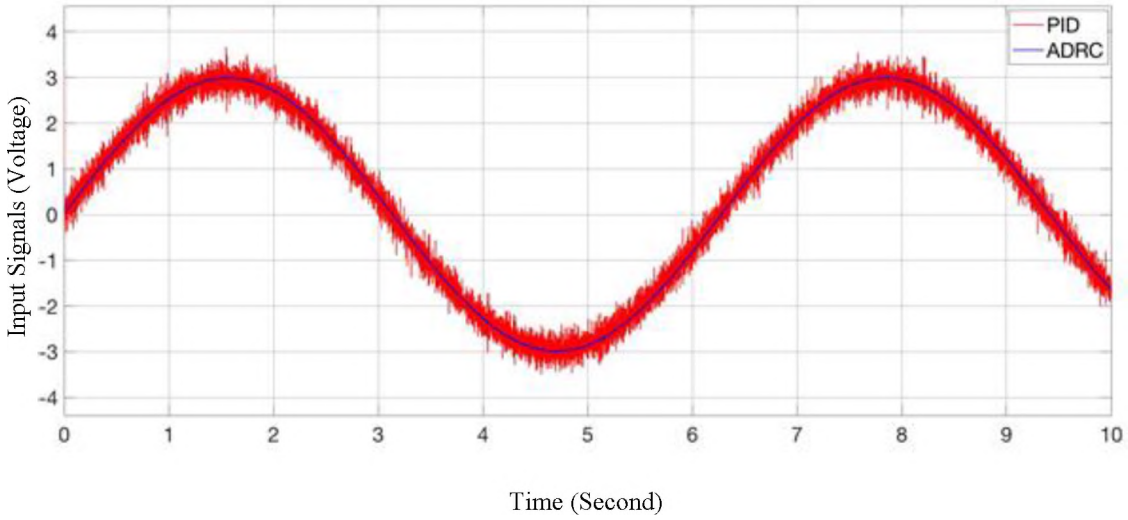

Figure 6.3. 3 PID & ADRC Speed Control input signals with white noise

This better result is the cause of the disturbance estimation item, which is f_1 in the simulation diagram. It proves that ESO estimated the noise well. In this process, the ω_{o2} plays a very important role. If the ω_{o2} is small, the estimation of the noises will have a low response time, and as a result, that not all noises will be rejected. If the ω_{o2} is large, the noises will be amplified, too. Also, the Gain can reduce or increase the noises and that will increase the burden to the ESO.

6.4 DISTURBANCE TESTS

In the disturbance tests, a square wave has been used as a disturbance and added into the input. The input quantity is the torque. So, this square wave simulates the disturbance of the ground, and its amplitude is 3, which equals the max amplitude of the input signal. The input disturbance is shown in Figure 6.4.1.

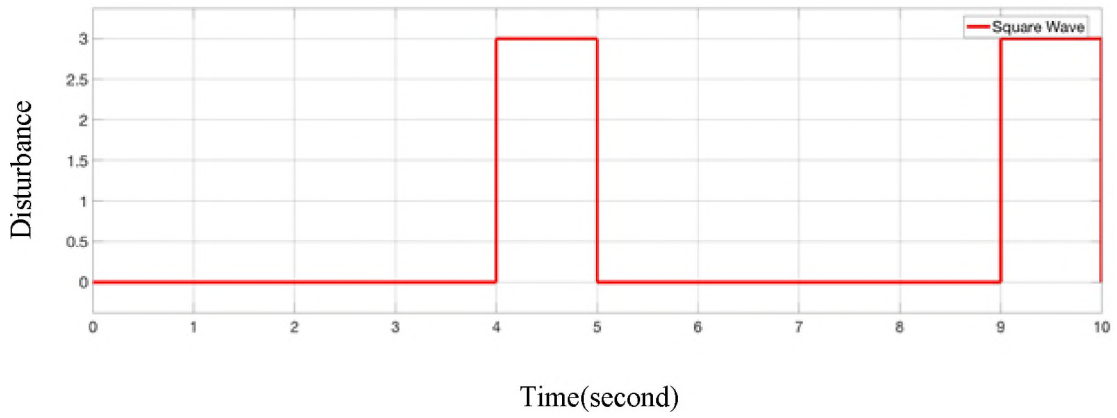

Figure 6.4. 1 Input Disturbance

The first pulse occurs at the 4 seconds of the simulation clock, and it lasts for 1 second occurring every 5 seconds. The output results of PID and ADRC are represented in Figure 6.4.2. For this system, PID and ADRC can delete the disturbance immediately. It is hard to tell the difference between PID and ADRC. In fact, both of the Proportional control and ADRC have steady stead error of the output. The errors are too small to see because of the large gain of the controller. When decreasing the gain of the controller, the steady-state error will increase. In addition, both of the speed loops have a disturbance, and the disturbances will affect each other's loop. This conclusion probably is the reason that the robot shakes during the running. Adding an integrator into the system can solve the steady-state error. As a cascade control system, this speed-loop input disturbance also impacts the speed-loop reference.

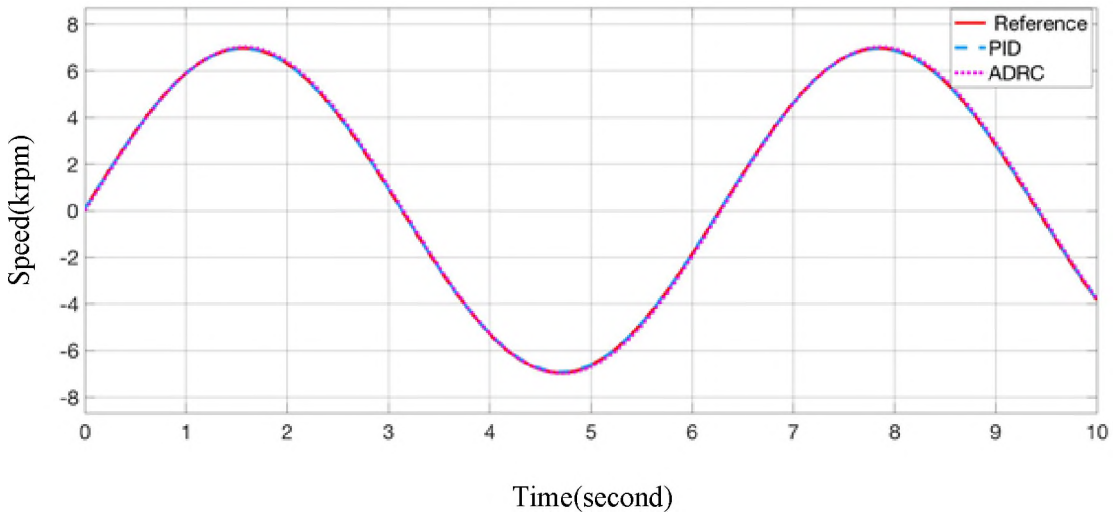

Figure 6.4. 2 PID & ADRC Speed Control output signals with disturbance

In order to test the robustness of PID and ADRC, the input disturbance increases to 100. Now the disturbance is much larger than the input signal. Then the results are represented in Figure 6.4.3. The PID Reference is changed, and there is an obvious steady-state error. However, ADRC still works very well. It proves that the ADRC has a stronger ability of disturbance rejection than PID and it supports the previous solution that PID controller cannot make the robot reverse stably.

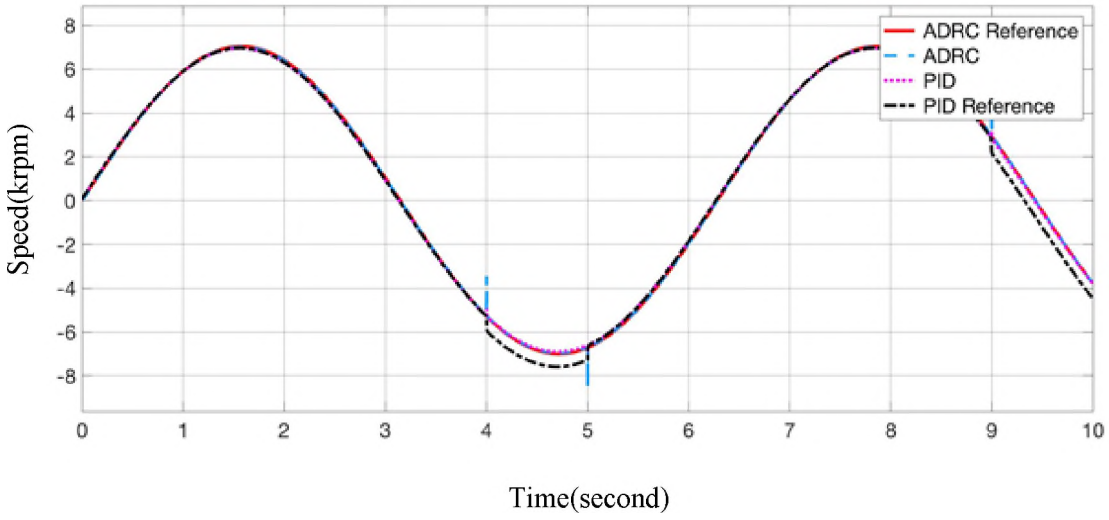

Figure 6.4. 3 PID & ADRC Speed Control output signals and references

6.4 ILC SIMULATION

The results of PID and ADRC controller are good. They can satisfy the most control systems. However, the ALFR control system has a characteristic that it is a repetitive control system. This point can be used to make the robot faster. ILC will help ALFR to perform better. As above, the ILC will cooperate with PID and ADRC, respectively. Based on (2.2.1): $u_{k+1}(t) = u_k(t) + dye_k(t)/dt$, this is parallel ILC [15, 40]. The idea of ALFR ILC simulation is simple: in the first run, the input signal $u_k(t)$ is stored somehow and then used for the next run $u_{k+1}(t)$. In the second run, the correction $dye_k(t)/dt$ is added with $u_k(t)$. The ILC belongs to the feed-forward control. The result of PID-ILC is shown in Figure 6.4.1.

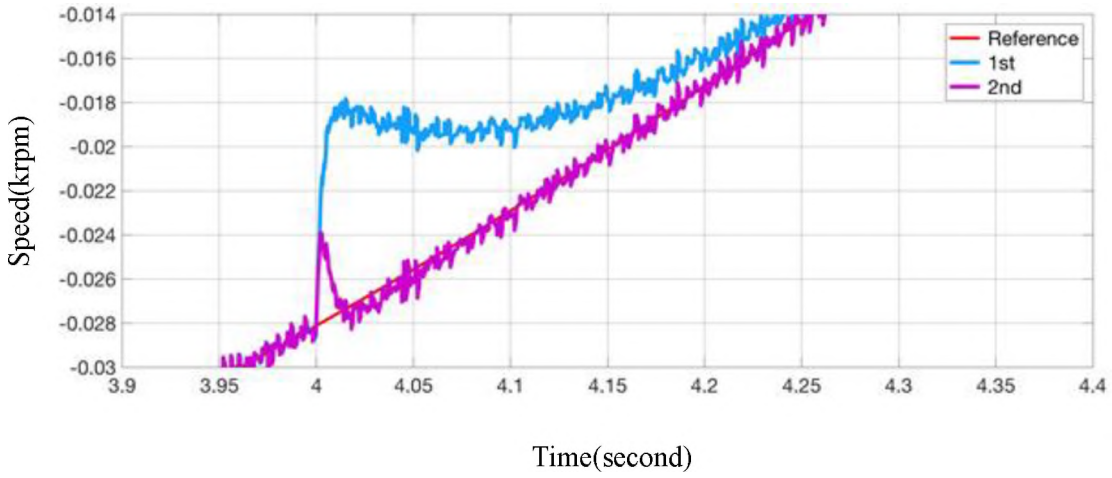

Figure 6.4. 1 Speed Control ILC with PID

It shows that ILC can correct the error, which is a square wave. Along with the repetition, the error will be reduced to zero. Obviously, ILC is a good tool for ALFR. The way of adding ILC is totally different for the systems. For example, iterating the input signal is different from iterating the error. The ADRC-ILC result is represented in Figure 6.4.2.

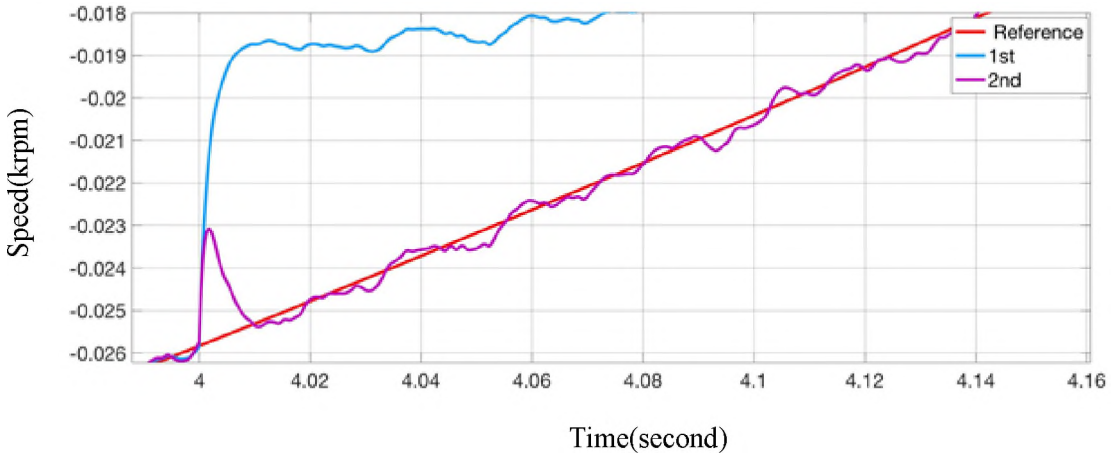

Figure 6.4. 2 Speed Control ILC with ADRC

PID can work with ILC better than ADRC. The deduction is that ADRC estimated the disturbance, so the feed-forward from ILC is treated as disturbances. There should exist a conflict between the feed-forward $u_{k+1}(t)$ and the disturbance estimation f. Furthermore, we change the feed-forward to be f, but the result is worse. Actually, PID oscillates at the 4th run, and ADRC oscillates at the 3rd run. In the future, it is necessary to check the convergence of the ALFR [41]. Actually, ILC also have certain ability to reject the disturbance. ILC works, but it is not as good as anticipation. However, ILC has many types and it evolves day by day. This experiment is entry-level for the ILC application.

6.5 ALFR EXPERIMENT RESULTS

In the experiments, the robot has been tested on the ground. The speed of the motor was set at 0.2 *krpm*. The horizontal axis is time, and each grid is five seconds. There was only one motor running on the ground and the other one is stopped. The motor used PI and ADRC, respectively. The motor speed control using PI is represented in Figure 6.5.1. The control signal (torque) is shown in Figure 6.5.2. For the ADRC, the motor speed control result is represented in Figure 6.5.3, and the control signal (torque) is shown in Figure 6.5.4.

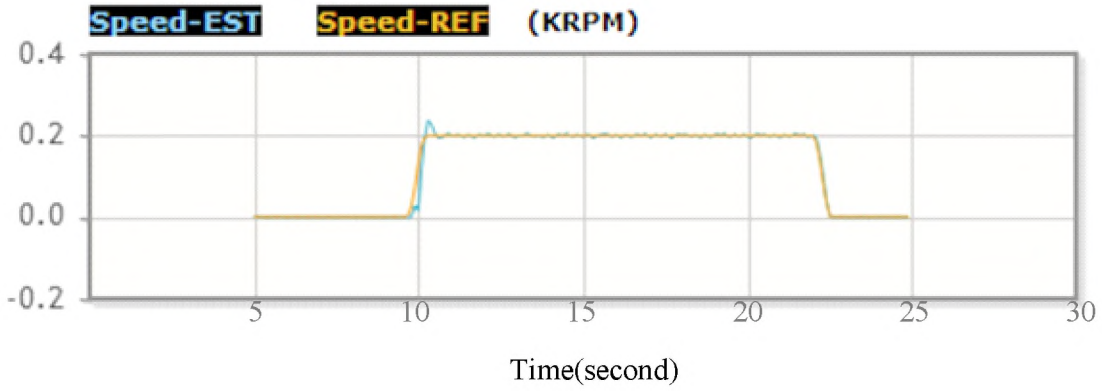

Figure 6.5. 1 PID Motor Speed

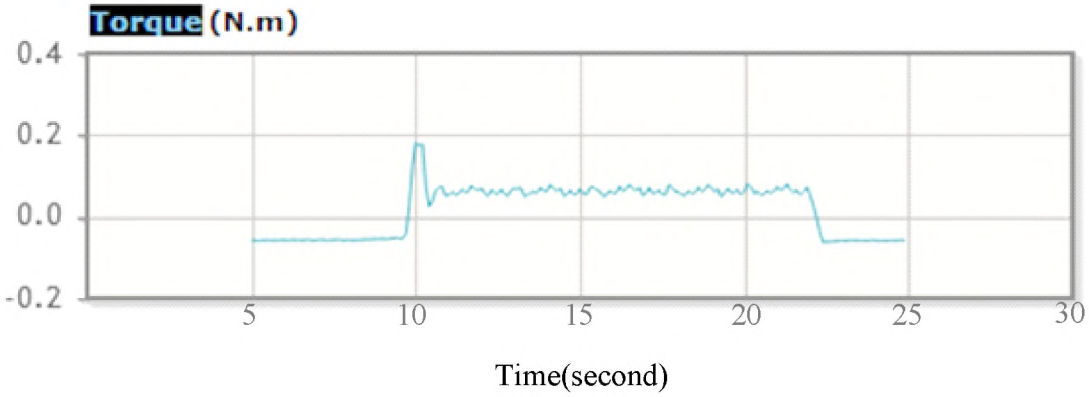

Figure 6.5. 2 PID Control Signal

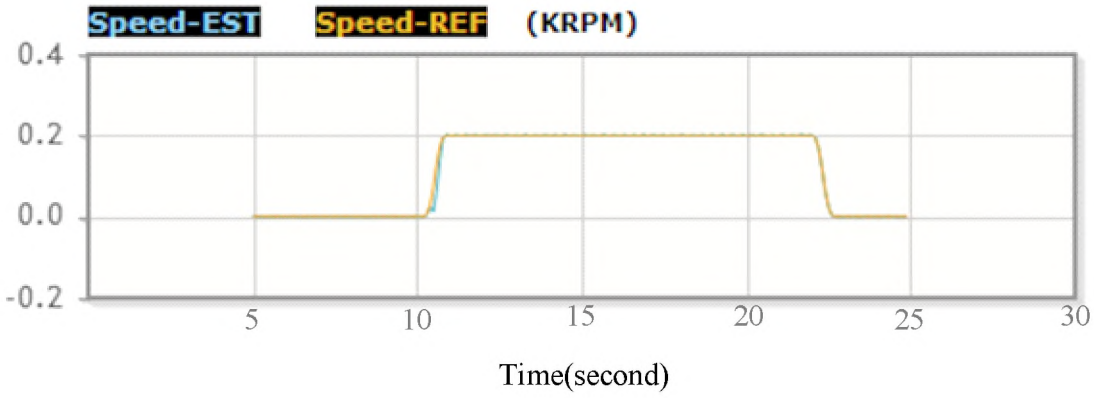

Figure 6.5. 3 ADRC Motor Speed

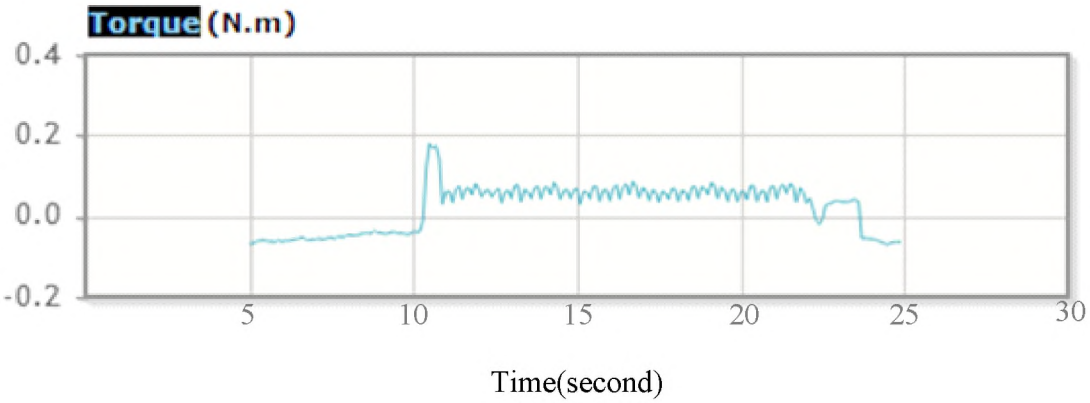

Figure 6.5. 4 ADRC Control Signal

Comparing the results of PI and ADRC, the settling time of both controllers is similar. The robot can stably run on the ground. In Figure 6.5.1, the parameter K_p is 60, and K_I is 0.6. Also, PI control often has an overshoot. If the K_p is increased, all disturbance would be amplified at the same time. As a result, that the ALFR becomes uncontrollable. In Figure 6.5.3, the parameter ω_c, ω_o of the ADRC is 60 and 240, respectively. ADRC has no overshoot, and the settling time is close to the PI's. It means

that ADRC ensures the speed and the stability of the ALFR at the same time. In addition, ADRC controller has stronger robustness. For example, ADRC can reverse the robot to 200 *rpm* when it is run at 200 *rpm,* which PI controller cannot.

CHAPTER VII

CONCLUSIONS

An ALFR design is presented in this thesis with a simple construction and thus a simple mathematical model. It makes it convenient to test different methods from control theory. Many problems have been encountered during in this research, not all of them, but they are worth studying nonetheless. The hardware design dictates how the control theory can be applied because of the constraints, such as the maximum current and speed of the motor and the accuracy of the sensors. We learned that the motor must be sized correctly, or it will not provide the torque necessary for the desired trajectory. We also learned that a dead zone in a motor can have a significant impact on the robot behavior. Overall, this research proved that practice does not always follow theory; not all problems can be solved by control theory alone.

Another problem in practice that cannot be ignored is the controller tuning. No matter what method is used, PID or ADRC, after the controller has been designed and inserted into the system, engineers must find a way to tune it. Most of the time, tuning PID depends on experience. Even PID only has three parameters to tune, which are difficult and complicated. Comparing to PID, ADRC also needs to be tuned, but is easier to tune. ADRC turns all parameters into functions of bandwidth, which greatly decreases the difficulty of tuning. Also, based on the results, ADRC is easier to use than PID, and ADRC has a better performance on disturbance rejection. The simulation results prove that ADRC has a larger bandwidth and it can filter the noise very well. The ADRC's output is still stable when the disturbance is increased.

No matter PID or ADRC, the ultimate objective is to make the robot follow the given line. It is demonstrated in this thesis that PID and ADRC can control this first order system very well. In the experiments, the ALFR cannot track the line as required. This is the reason why we established the mathematical model of this robot system and simulation. The simulation results did show that the robot's two motors easily affect each other, also including the disturbance. On the other hand, simulation is not a real experiment. The high gain cannot be applied in the experiments because high gain will make the system uncontrollable. In this cascade control system, every problem is not caused by only one reason. However, the order of control is clear that the first step is controlling the inner loop well and then the outer loop.

Future work in this area involves improving the performance of the speed control. For example, the friction of the ground should be considered. In the experiments, the ALFR cannot respond quickly. Thus, the correct maximum speed should be determined

and then the gain of the position controller should be increased for a faster response. We can also build a wireless channel into the robot for it to communicate with the computer, making it possible to export the position data, help to debug the ALFR, and improve the controller design. In addition, we can set up the second position sensor to determine the starting point and the end point. Those points will make it possible to implement the ILC in the ALFR system so that the performance can improve over time.

BIBLIOGRAPHY

1. M. S. Islam & M. A. Rahman, "Design and Fabrication of Line Follower Robot," *Asian Journal of Applied Science and Engineering*, Volume 2, No.2, 2013.

2. Sheikh Farhan Jibrail, Rakesh Maharana, PID Control of Line Followers, B.C. thesis, National Institute of Technology, Rourkela, 2013.

3. Tanvir Ahmed, Khandakar Mahbubul Islam, Md. Faiz Ahmed, Rakib Hossain, and Md. Mahbubur Rahman. "Tree Based Localization Model for Line Following Robots-," *International Journal of Multimedia and Ubiquitous Engineering*, Vol.12, No.1 , pp. 17-34, Dec 2017.

4. Zhiqiang Gao, "On Disturbance Rejection Paradigm in Control Engineering," in *Proceedings of the 29th Chinese Control Conference*, July 29-31, 2010, Beijing, China, pp. 6071-6076.

5. Pololu Corporation, Pololu AVR Library Command Reference, 2001–2015 https://www.pololu.com/docs/0J18

6. Frangois G. Pin and Hubert A. Vasseur, "Autonomous Trajectory Generation for Mobile Robots with Non-holonomic and Steering Angle Constraints," in *Proceedings of the IEEE International Workshop on Intelligent Motion Control*, Istanbul, Turkey, August 20-22, 1990.

7. Jingqing Han, "From PID to Active Disturbance Rejection Control," *IEEE Transactions on industrial Electronics*, Vol. 56, No. 3, March 2009

8. Jonathan Connell, Paul Viola. "Cooperative Control of a Semi-Autonomous Mobile Robot," ICRA-90.

9. Z. Gao and R. Rhinehart, "Theory vs. Practice Forum," in *Proc. of the 2004 American Control Conference*, June 30-July 2, 2004, Boston MA, pp. 1341-1349.

10. J. Han, "Control Theory: Is it a Theory of Model or Control?" *Systems Science and Mathematical Sciences*, Vol.9, No.4, pp.328-335, 1989. (In Chinese)

11. Zhiqiang Gao, "On the Problem of Information in Engineering Cybernetics," *J. Sys. Sci. & Math. Scis.* Vol.36 (7), pp. 908–923, July 2016.

12. Zhiqiang Gao, "On the centrality of disturbance rejection in automatic control," *ISA Transactions*, Volume 53, Issue 4, July 2014, Pages 850-857

13. Zhiqiang Gao, "Active Disturbance Rejection Control: A Paradigm Shift in Feedback Control System Design," in *Proceedings of the 2006 American Control Conference Minneapolis*, Minnesota, USA, June 14-16, 2006

14. Zhiqiang Gao, "An Alternative Paradigm for Control System Design," in *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, Florida USA, December 2001

15. Van de Wijdeven, J.J.M. "Iterative Learning Control design for uncertain and time-windowed systems," Ph.D. Dissertation, Technische Universiteit Eindhoven, 2008.

16. Kevin L. Moore, Mohammed Dahleh, and S.P. Bhattacharyya, "Iterative Learning for Trajectory Control," in *Proceedings of 1989 IEEE Conference on Decision and Control*, Tampa, Florida, pp. 860-865, December 1989.

17. Hyo-Sung Ahn, YangQuan Chen, and Kevin L. Moore. "Iterative learning control: brief survey and categorization," *Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*. Volume 37, Issue 6, Pages 1099-112, November. 2007.

18. Kanayama, Yutaka. "A Stable Tracking Control Method for an Autonomous Mobile Robot," Internet: http://hdl.handle.net/10945/40155, 1990.

19. Ji-wung Choi, Renwick E. Curry and Gabriel Hugh Elkaim, "Curvature-Continuous Trajectory Generation with Corridor Constraint for Autonomous Ground Vehicles", *49th IEEE Conference on Decision and Control*, Hilton Atlanta Hotel, Atlanta, GA, USA, December 15-17, 2010.

20. Vivien Delsart, Thierry Fraichard, Luis Martinez-Gomez. "Real-Time Trajectory Generation for Carlike Vehicles Navigating Dynamic Environments." *IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan. May 2009.

21. Ebiesuwa O. O, Adekunle Y. A, Akinyemi L. A, Oyerinde O. D. "Line Follower Robot Using a Sophisticated Sensor Approach," *International Journal of Engineering Research & Technology (IJERT)*, ISSN: 2278-0181 Vol. 2 Issues 7, July 2013.

22. *TMS320F28335, TMS320F28334, TMS320F28332, TMS320F28235, TMS320F28234, TMS320F28232, Digital Signal Controllers (DSCs)*, Literature Number: SPRS439M, TEXAS INSTRUMENTS, June 2007–Revised August 2012

23. J.Borenstein, Y.Koren, "A mobile platform for nursing robots," *IEEE Trans. Ind. Electron*, vol. IE-32, pp. 158-165, June. 1985.

24. *TMS320F2806x PiccoloTMMicrocontrollers*, Literature Number: SPRS698F, TEXAS INSTRUMENTS, NOVEMBER 2010 – REVISED MARCH 2016

25. R. Craig Conlter. "Implementation of the Pure Pursuit Path Tracking Algorithm." *The Robotics Institute Camegie*, Mellon University. January 1992.

26. Huosheng Hu. "Dynamic Planning and Real-Time Control for A Mobile Robot," Ph.D. Dissertation, University of Oxford, 1992.

27. Michael Brady, Huosheng Hu. "Software and Hardware Architecture of a Mobile Robot for Manufacturing," *AAAI Technical Report* SS-95-02. 1995

28. *InstaSPIN™ BLDC Lab*, Application Report, TEXAS INSTRUMENTS, SPRABN7– November 2011

29. C.Samson, K.Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in cartesian space," in *Proc. 1991 Int. Conf. Robot. Auto. Sacramento*, CA, pp. 1136-1141, Apr. 1991.

30. Johann Borenstein. "Control and Kinematic Design of Multidegree-of-Freedom Mobile Robots with Compliant Linkage," *IEEE Transactions on Robotics and Automation*, February 1995, Vol. 11, No. 1, pp. 21-35.

31. J. C. Alexander, J. H. Maddocks. "On the Kinematics of Wheeled Mobile Robots," Department of Mathematics University of Maryland College Park, Maryland 20742. Revised: October. 1988.

32. R. W. Brockett, A. Stokes, and F. Park "A geometrical formulation of the dynamical equations describing kinematic chains," *In IEEE International Conference on Robotics and Automation*, pages 637–642, 1993.

33. J. Angeles, *Rational Kinematics*. Springer-Verlag, 1988.

34. H. Asada and J. J. Slotine, "Robot Analysis and Control," John Wiley, 1986.

35. J. J. Craig "Introduction to Robotics: Mechanics and Control" Addison Wesley, second edition, 1989.

36. RM Murray, Z Li, SS Sastry. "Robot Dynamics and Control," in *A Mathematical Introduction to Robotic Manipulation.* US: CRC Press, 1994, pp.153-160.

37. Alessandro De Luca, Bruno Siciliano. "Closed-Form Dynamic Model of Planar Multilink Lightweight Robots." *IEEE, Transaction on Systems; Man and Cybernetics*, Vol. 21, NO. 4 July/August, 1991.

38. Robert Miklosovic, Aaron Radke, and Zhiqiang Gao, "Discrete Implementation and Generalization of the Extended State Observer," in *Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, USA, June 14-16, 2006

39. Zhiqiang Gao, "Scaling and Bandwidth-Parameterization Based Controller Tuning," in *Proc. of the American Control Conference*, 2006, pp. 4989-4996.

40. Shu-Wen Yu. "Enhanced Iterative Learning Control with Applications to A Wafer Scanner System." Ph.D. Dissertation, University of California, Berkeley. Spring, 2011.

41. Hyo-Sung Ahn, Kevin L. Moore, Yangquan Chen. *Iterative Learning Control.* London, Springer, February 5, 2007.