<Undergraduate Thesis: EEE-2015-02-08>

Build and control a rolling robot

Wonchul Kim, Chan Sagong

School of Electrical and Electronic Engineering

College of Engineering

Yonsei University

<Undergraduate Thesis: EEE-2015-02-08>

Build and control a rolling robot

Thesis Advisor: Daeun Kim

A thesis submitted in a partial fulfillment
for the Capstone Design for Electrical & Electronic
Engineers' requirements

June 2015

Wonchul Kim, Chan Sagong
School of Electrical and Electronic Engineering
College of Engineering
Yonsei University

감사의 글

대학생활의 막바지에, 그 동안 배운 것 들을 총 동원하여 종합설계를 진행하였습니다. 시작할 때에 가졌던 목표들이 시간이 지나며 점점 바뀌어 나갔고, 진행에 어려움을 겪기도 했습니다. 알았던 내용을 더 공부하고 새로운 것들을 배우며 우리 스스로 배운 것을 연구에 접목시키는 것이 참으로 힘들다는 걸 깨달았습니다.

우리가 연구과제로 삼은 주제는 구형 로봇에 관한 것이었고 하드웨어 제작부터 소프트웨어 처리까지 모두 진행하는 연구였습니다. 예상과는 달리 초기 하드웨어 제작에서부터 큰 어려움을 겪었습니다. 아크릴 소재의 재료를 통해 전체 베이스 부분을 만드는데 있어서, 가장 먼저 3D CAD 프로그램을 이용해서 모델링 하고, 발주를 맡기기 전 손으로 미리 제작을 해야 했습니다. 구성한 도면을 통해 일일이 손으로 아크릴을 꺾아 나가면서 깨지기도 하고, 사이즈가 맞지 않아 다시 만들기도 하며 시행착오를 겪었습니다. 소프트웨어적인 부분도 새로운 프로그램을 사용하느라 환경설정에서부터 우리의 모델에 적용하기까지 시간이 많이 걸렸습니다. 그러나 연구를 진행하는 동안 많은 분들의 도움으로 연구를 완료할 수 있었습니다.

먼저 연구 진행에 아낌없는 지원을 해주시고 연구 목표를 함께 세워주셨으며, 진행 상황에 맞게 방향을 설정해주신 김대은 교수님께 감사의 말을 드립니다. 그리고 초기부터 마무리까지 함께 의견을 나누고, 조언을 주신 이창민, 손승배 조교님께도 감사의 말을 전합니다.

Contents

Figure index

Table index

ABSTRACT

Build and control a Rolling Robot

In this paper, a miniature spherical rolling robot capable of navigating over two dimensional surfaces, applying the mechanism of cheetah's tail, and controlled entirely by two methods is described. This 18cm spherical shell driven by an internal two wheeled differential drive cart. A gravity powered pendulum effect is produced as the internal device climbs up the internal surface of the shell, propelling the robot forward up to a speed of 0.2m/s. We derived its dynamic model using Lagrangian and studied its dynamics and performance under various applied torques. The spherical robot is built and its overall mechanical, hardware and control architecture are elaborated. Experiments are conducted to evaluate the robot open and closed loop performance on a linear trajectory, controlled by two methods, which are bluetooth controller offered by 'google store' and the number of finger captured using a webcam. Either way, the robot basically follow the desired orientation we command with our implemented PID controller. With a Bluetooth controller, we can command forth, back, left, and right to the robot and we can also control the speed of the robot faster and slower by the application keys. Through the webcam, it track the hand and count the finger of the hand and this count number is the command like when it is one, it means forth as we already assigned.

Key words : spherical, pendulum effect, torque, PID, bluetooth, finger-count

1. Introduction

Entering the 20th century, the spread of computer and mobile system has been a revolutionary change in human life. Recently support for national technologies like IT(Information Technology), BT(Biotechnology Technology), NT(Nanotechnology Technology) was the emergence of new electronic products, fused with existing technologies, such as mechanical robots. Since the robot is one of the country's 10 new growth engine industry is still in formative stages, but the target market, a very large market, such as home or public facilities, universities, research institutions have the power pitching and the development and commercialization of technology-based companies in many developed as well. Therefore, by 2020s There will be one robot per household and we look forward to life that the robot live with human and has many roles instead of human or rather give human to enjoy new life by surpassing the capability of human[1].

Since the word 'robot' is used for the first time in the 1920s "Russum's Universal Robots" a wide variety of robots has been made as products around us, or a virtual image in a movie or a novel. Because of this variety of the type it is not easy to define only one. As we classify robots based on a most general method, its own purpose, they can be divided into for industrial robots and for service robots and robots for service is again divided into for professional services and for personal service. Industrial robots are mainly used in manufacturing welding, poor conditions, such as construction sites that requires a mastery of high levels of labor, semiconductor manufacturing plant to improve the productivity, and deep-sea exploration, which all are places where it is not available for people to work[2].

Robots for professional services are designed for a specific purpose. For example, medical robots capable of minute surgery, military robots to complete missions, and fire fighter robots

[1] 김병수, 지능형 서비스 로봇 – 산업 활성화, 한국정보통신기술협회, 2005.

[2] 최병태,노근창,개인 서비스용 로봇 산업,HMC 투자증권,200

are representatives. Personal service robots are made for serving people in daily life such as for lyrics, nursing, entertaining, leisure, and education. Figure 1.1 shows that industrial robots are utilized in modern society for professional services, personal services, medical purpose and so on.

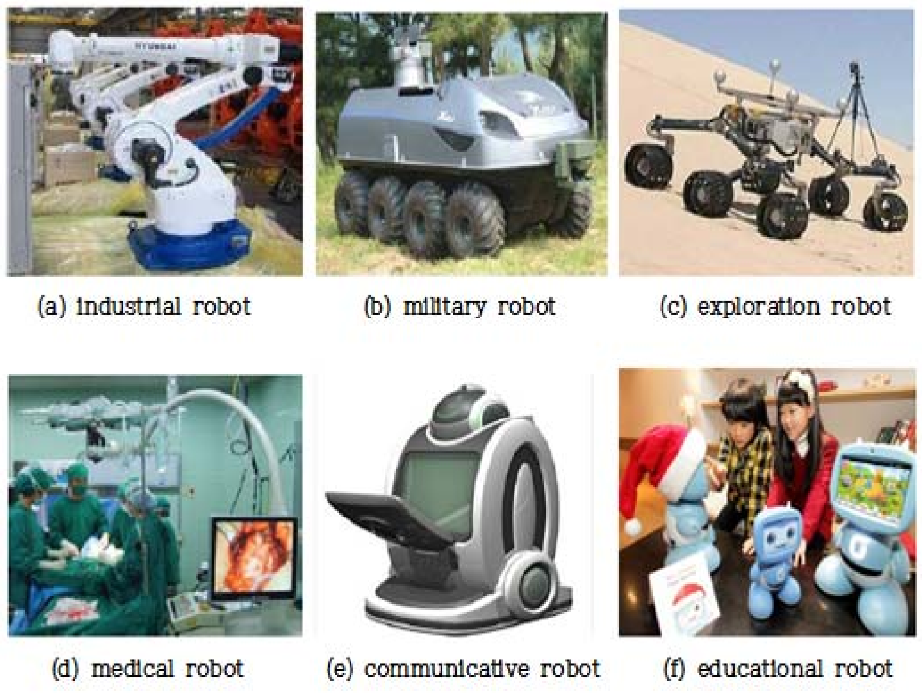


Fig. 1.1. Application for robot of various fields

The era of service robots related to our life very close will come in the future. Service robot are being customized as products capable of doing work instead of human or helping them more effectively by allowing the robots have more roles based on the core technology of robot. Personal service robot give information to human, robot which can have emotional conversation is being developed for educating children, or nursing the old, and daily life service robot is doing house chores like cleaning, doing laundry, or guarding the house. All these robots are now of development. Especially, personal service robot can be designed in a variety of types based on the demand of customers. In this regard, service robot has larger growth potential than professional service robot does and it will stimulate the growth of domestic robot industry which is now a little bit small scale. One-person household gradually increased is also expected to drive the demand for service robot. One-person households tend to find emotional counterpart and it seems to be welcome to entertainment robot or pet robot. So it is expected that expansion of one-person household and diversification of the play have effect to widen the purpose of service robot from just supporting our life to even giving us pleasure and fun[3].

The most of service robots we can look around are still cleaning robots. Figure 1.2 shows the cleaning robots actually used at home and it has greatly contributed to the popularization of the robot occupying 60% of personal service robots.

Fig. 1.2. commercailly available cleaning robots

It would be essential to have smooth communication bewteen the user and the robot because of the lack of ability for robot to respond independently. So robot will be expected to place in every home when combined with a mobile communication in the future and it will develop more. Currently, industrial robot has accounted for a large proportion of the robot market and it can't be said that personal robot is in firm state though, it is going to be the core of the robot technology and will be used in all areas in the future[4].

Most of robots move and perform their own mission. Figure 1.3 shows various methods how to move: wheeling, walking, crawling, rolling, swimming, and flying. These are applied in several fields.

[3] 최병태,노근창,개인 서비스용 로봇 산업,HMC 투자증권,200

[4] 이홍석, 개인 서비스 로봇의 잠재력, 디지털 타임스,2012.10.2

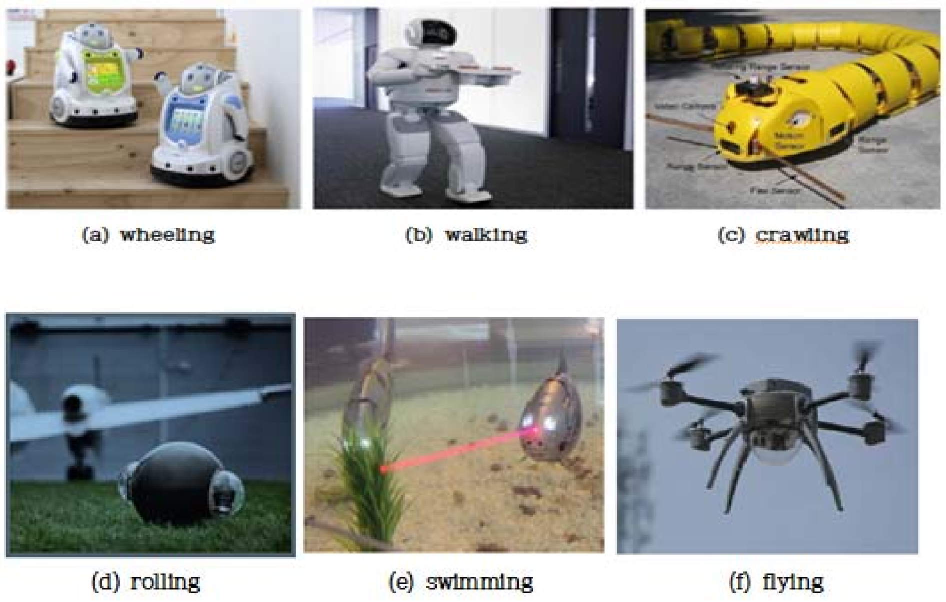


Fig. 1.3. The mobile robots having various driving mechanism

Rolling robot is the one of them which is the most studied. The wheeling robot uses method of producing the power to rotate wheels connected to the driving source like motors. Advantage is the stable high-speed movement on the flat ground and it can load things heavier than other can. In contrast, when it faced obstacle and size of the wheel is bigger than the obstacle, it is hard to avoid it and each wheel's friction is different, so it can slip on the rugged ground.

Walking robot is imitated by the human being in the view of the legs' movement of biological organism to implement the mobile of robot. The good thing is that it is able to be used in almost every environment compared to the wheeling robot. However, its structure would be complex and the control would be also very difficult and slow, so it has a limit in terms of pracitcal aspect when it comes to consider efficiency.

Flying robot is flying around the air and it is usually used for military missions like reconnoitering the area of operations or controlling the disaster. It can be separated into two types: plane with a fixed wings and hellicopter with rotating wings, and most of them are dronea pilotless planes. Swimming robot moves under the sea and it usually detects or explore the deep sea[5].

So far, the development of the mobile robot has been another sphere of mobile robot research as one way. The spherical robot exhibits the behavior of the omnidirectional and reduce the risk of rollover of a sphere having a symmetrical structure and the mobility by using the small actuator has a distint advantage. This could be used for entertainment, military, exploration, and education, especially for the infants, it is highly preferred to entertain them as a personal service robot.

2. Background

2.1 position and attitude of a rolling robot equipped with a Gyro

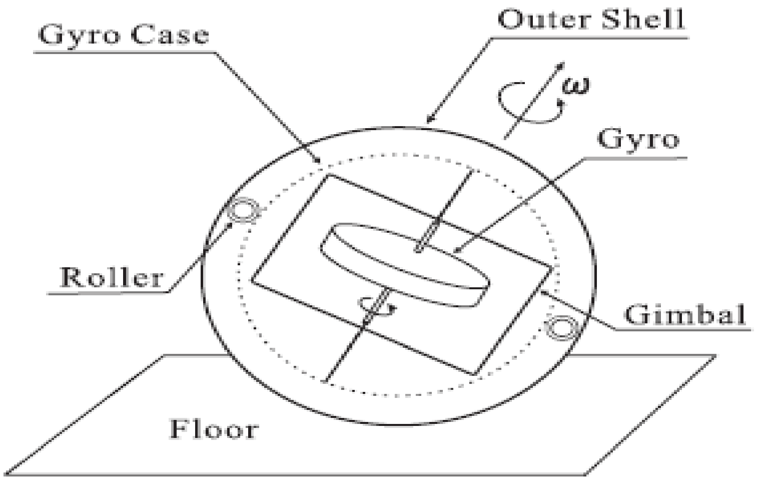


Fig. 2.1.1. Schematic model of a spherical robot

Figure 2.1.1 shows the driving mechanism of the spherical robot that we are now developing. The spherical robot is composed of four links, gyro, gimbal, gyro case and outer spherical shell. The gyro rotates on an axis inside the gimbal at a high speed. Three DC motors

5 서현덕,풍력 구동 메커니즘 기반의 구형로봇,석사학위논문,경북대학교,전자전기컴퓨터학부,2012

are installed as actuators to change the orientation of the gimbal arbitrarily. Two motors located at the rollers in the gyro case generate the driving torques about the axes perpendicular to the rotation axis of the gyro, and one motor located in the gimbal generates the driving torque about the rotation axis of the gyro. When the rollers installed between the outer shell and the gyro case rotates, most of the power supplied by the motors is transmitted to the outer shell, because the change of attitude of the gyro case is small due to the gyro rigidity produced by the angular momentum of the gyro. Therefore, the driving mechanism of the spherical robot is expected to provide efficiently the driving torque to rotate the outer shell. Moreover, the robot is designed so that the center of mass of the spherical robot lies exactly at the geometric center of the sphere. And so, when the robot moves on the horizontal plane, gravitational force acting on the robot does not rotate the robot, because the center of mass of the robot always lies above its contact point

2.1.1 Gyro sensor : myahrs+

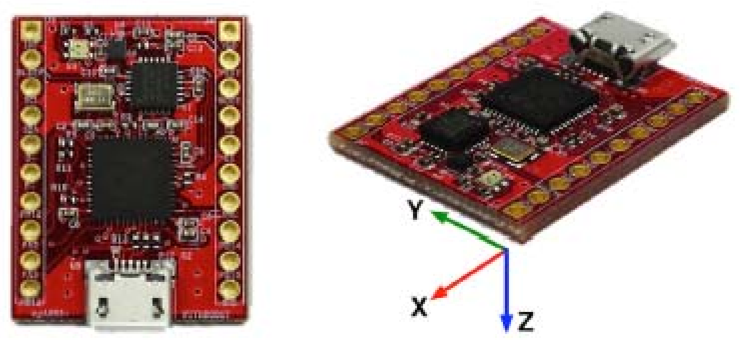


Fig. 2.1.1.1. The definition of myahrs+ coordination

Myahrs+ is a sensor module 3-dimensional producing attitude and heading through the UART/I2C/USB interface by merging outputs from the inertial sensor and geomagnetic sensor respectively.

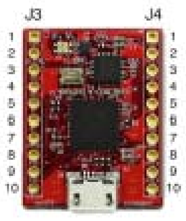


Fig. 2.1.1.2. Pin map

pin number	name	port	definition
J3 - 1	INT	O	Data ready interrupt output
J3 - 2	SLEEP	I	for sleep mode
J3 - 3	I2C_SCL	I	input for I2C clock
J3 - 4	I2C_SDA	I/O	I/O for I2C data
J3 - 5	USB_DM	I/O	USB D-
J3 - 6	USB_DP	I/O	USB D+
J4 - 1	VDD	PWR	+5V power input
J4 - 2	nRST	I	for reset
J4 - 3	NCI		-
J4 - 4	UART_TX	O	UART transmitter
J4 - 5	UART_RX	I	UART Receiver
J4 - 6	NC		-
J4 - 7	NC		-
J4 - 10	GND	PWR	power ground

Table 2.1.1.1. Pin information

Figure 2.1.1.2 and table 2.1.1.1 shows the informaion of the myahrs+ on each pin in detail.

RPY	Format	$RPY, sequence number, roll, pitch, yaw
	Example	$RPY,68,0.04,1.56,34.22

Table 2.1.1.2. Data format by ASCII

Table 2.1.1.2 shows how the data transfer in terms of format.

2.1.2 Euler angles

For balancing, Gyro sensor is needed to know pitch, roll, and yaw. Pitch, roll, and yaw are aircraft principal axes. The yaw axis is defined to be perpendicular to the wings with its origin at the center of gravity. The pitch axis passes through the plane from wingtip to wingtip. Pitch moves the aircraft's nose up and down. The roll passes through the plane from nose to tail. A positive roll angle lifts the left wing and lowers the right wing. It can easily understand by Figure 2.1.2.1

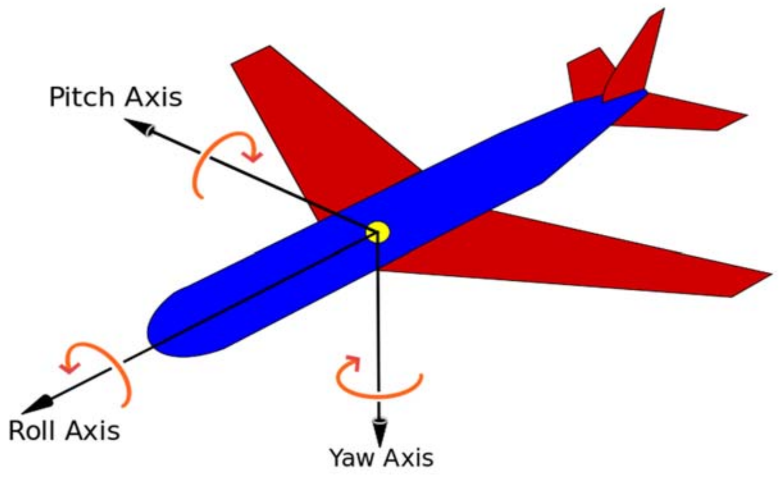


Fig. 2.1.2.1 euler angle

2.2 Motor control

2.2.1 DC motor : RA-12WGM + Encoder 02TYPE(6V)

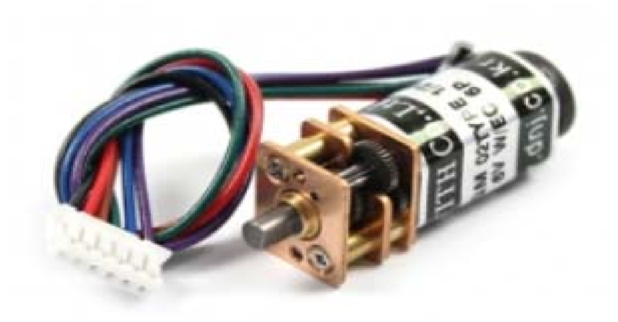
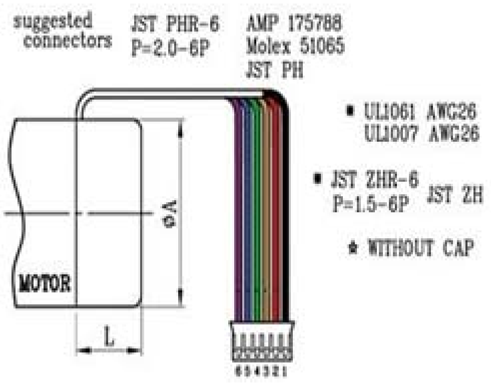
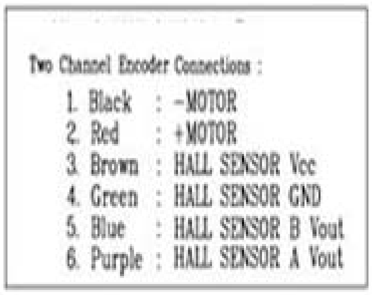


Fig. 2.2.1.1. RA-12WGM + Encoder 02TYPE

Figure 2.2.1.1 is the actual picture of the motor we used.


suggested connectors
JST PHR-6 P=2.0-6P
AMP 175788 Molex 51065 JST PH
■ UL1061 AWG26 UL1007 AWG26
■ JST ZHR-6 P=1.5-6P JST ZH
★ WITHOUT CAP
MOTOR
φA
L
654321



Two Channel Encoder Connections :

1. Black	:	-MOTOR
2. Red	:	+MOTOR
3. Brown	:	HALL SENSOR Vcc
4. Green	:	HALL SENSOR GND
5. Blue	:	HALL SENSOR B Vout
6. Purple	:	HALL SENSOR A Vout

Fig. 2.2.1.2. Motor pin information

Figure 2.2.1.2 shows the information of each pin of the motor. Pin 1 and 2 are power and each one is + and – respectivley with 6V power.

2.2.2 DC motor driver : DCM-A1

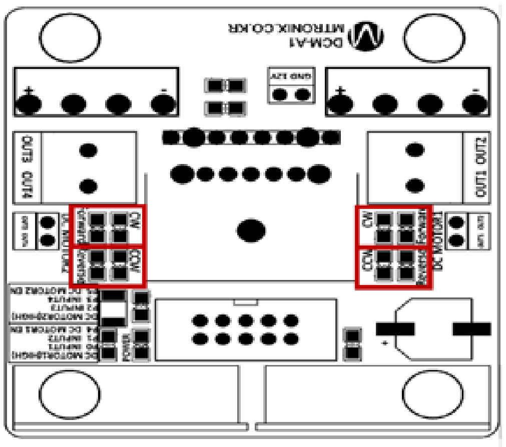


Fig. 2.2.2.1. DCM-A1 pin information

PORT1	PORT2	PORT3	PORT4	PORT5	PORT6	PORT7	PORT8	PORT9	PORT10
FORTH1	BACK2	FORTH1	BACK2	EN1	EN2	-	-	VCC	GND

Table 2.2.2.1 pin information

Figure 2.2.2.1 and table 2.2.2.1 shows the laction of each pin on the motor driver and pin information. Seeing table 2.2.2.1 in detail, it explains how it works. To drive DC motors connected to this motor driver, EN1 needs to be HIGH(1), when going forth PORT1 needs to be HIGH(1) and PORT2 be LOW(0), and the other way is same: EN1 is to be HIGH(1), PORT1 be LOW(0), and PORT2 be HIGH(1). In this fashion, motors are drived forth or back.

2.2.3 Servo motor

For control of angle we can use servo motor. This is DC motor which is controlled for specific angular rotation with help of additional servomechanism (a typical closed loop feedback control system). The servo motor is very useful because a servo is that it provides angular precision, i.e. it will only rotate as much we want and then stop and wait for next signal. This is a difference aspect of servo motor. A normal electrical motor keeps rotating as long as it supplied power. If power is supplied, normal motor does not stop. So servo motor is a type of motor and it is moved to certain limit with help of error-sensing feedback. In servo system, there are three components and these are controlled device, output sensor and feedback system. This output is controlled by feedback principle by comparing between output and input. From the input signal, the difference between input and output is calculated and this value is used to drive motor to eliminate the error.

Inside the servo motor, it has dc motor, potentiometer and gear arrangement. The potentiometer helps the servo to rotate according to input. Gear arrangement is needed to make enough torque from the dc motor that has fast speed and weak torque. To rotate servo motor, some electric pulse is needed. Pulse-width modulation (PWM) signals are transmitted to the servo. and this signals are translated to the width of the pulse into a position inside the servo. As we discussed in above, we use PWM signal. The rotation angle depends on pulse width.

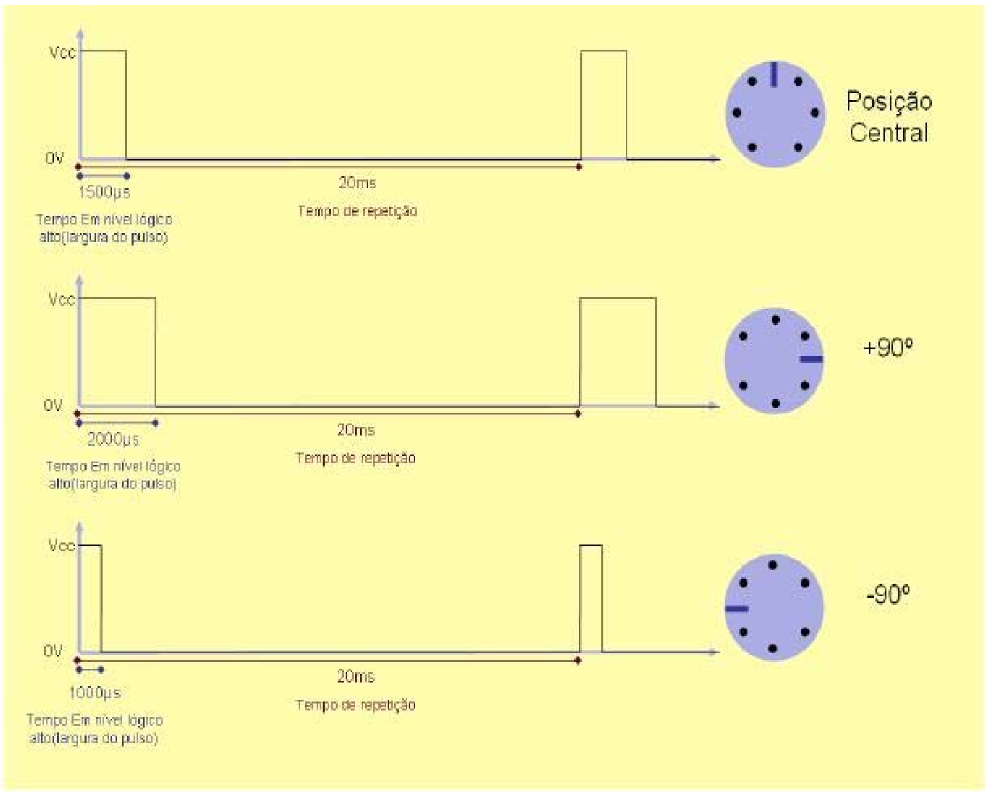

Vcc
0V
1500µs
Tempo Em nivel lógico alto(largura do pulso)
20ms
Tempo de repetição
Posição Central
Vcc
0V
2000µs
Tempo Em nivel lógico alto(largura do pulso)
20ms
Tempo de repetição
+90º
Vcc
0V
1000µs
Tempo Em nivel lógico alto(largura do pulso)
20ms
Tempo de repetição
-90º

Fig. 2.2.3.1 PWM of servo motor

According to pulse width that is applied to the servo motor, the angle of mechanical rotation is determined. This is a form of pulse-width modulation, however servo position is not defined by the PWM duty cycle but only by the width of the pulse.

For example, a servo motor can have specification – require 3-5V peak to peak square wave pulse and pulse duration which is formed from 0.9ms to 2.1ms, the pulse refreshes after 20ms. From these characteristics, we make pulse width from 0.9ms to 2.1ms and these values can be converted to degree of mechanical part of servo motor.

In C language, this can be made by timer of MCU(like atmega128) and interrupt service routine.

2.3 PID Control

PID control is widely used control method. To get desirable output, this control is very important. If we just use on/off control, output will be fluctuating. In PID control, there are three term. P, I, and D. Each of them means proportional, integral, and derivative. In control engineering, we can check these terms in frequency domain. Before we look PID a control, let's check the specification for controller. To make appropriate controller, controller is satisfied transient response, steady state error, and stability. In transient response, there are three component, peak time, % overshoot, and settling time. The peak time is the time when signal reaches to first peak of signal. This represents how fast system respond. % overshoot has relation like next equation.

$$\%oversoot = \frac{overshoot - steadystate}{steadystate} \times 100(\%)$$

In this equation, overshoot is first peak value of signal. Settling time is the time required for the transient's damped oscillations to reach and stay within $\pm$ 2% of the steady-state value.

These values can easily understand from below graph.

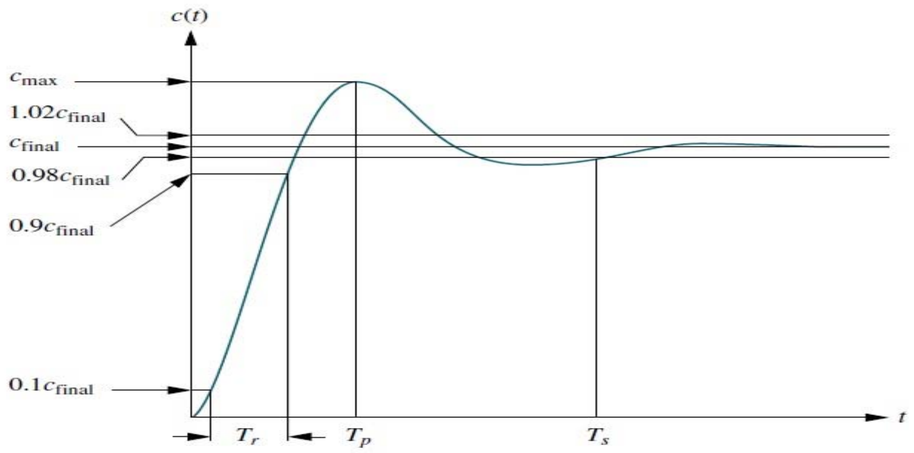


Fig. 2.3.1 response of system

Cmax is the first peak value and this time is peak time. In above equation, overshoot is a value of Cmax and steady state is Cfinal. Settling time is Ts in graph, the first value within 2% of final value.

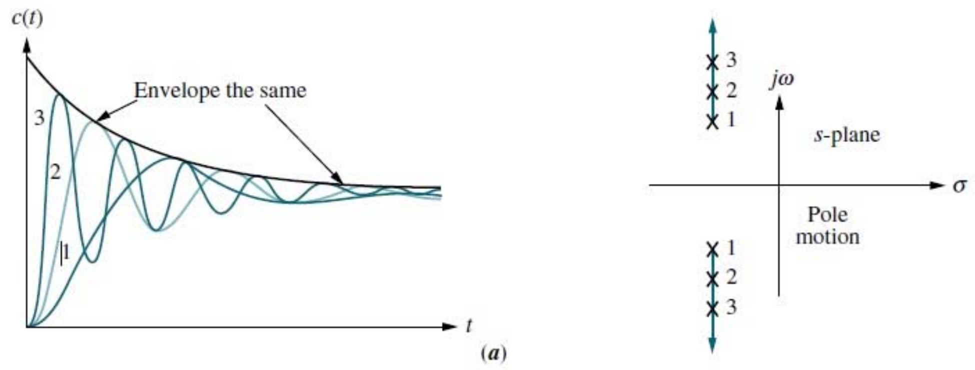


Fig. 2.3.2 change of imaginary part in root locus

In above figure, pole is moved to up or down and real part is same. This movement affects the response like left graph. From left graph, the peak time decreases and overshoot increases. Actually, real part of pole is same, so the envelope of graph is same.

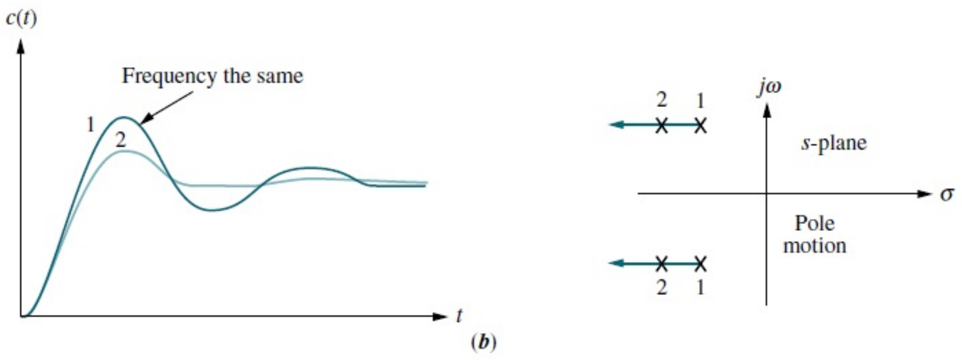


Fig. 2.3.3 change of real part in root locus

Next, pole is moved to left or right and imaginary part of pole is unchanged. Because of the same imaginary part, peak time is same. Overall oscillation is lowered, so the settling time decreases.

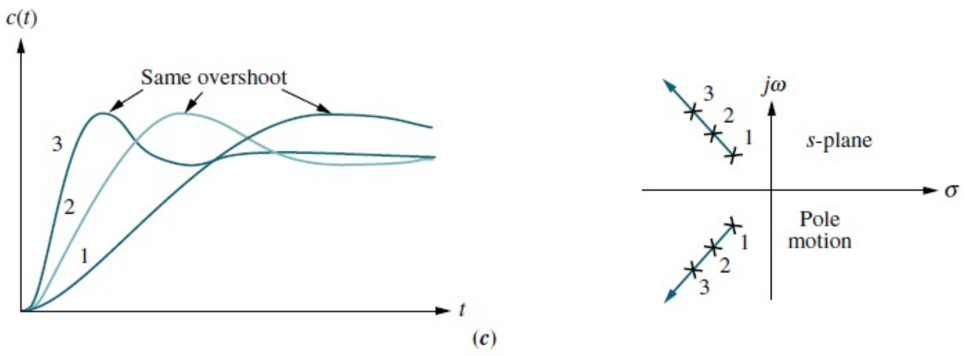


Fig. 2.3.4 change of imaginary and real part in root locus

Now we change both of imaginary and real part. The pole is moving along a constant radial line. From the left graph, the peak value and final value is same so % overshoot doesn't change. Although % overshoot is same, the speed of response is changed. The farther the poles are from the origin, the more rapid the response. From above discussion, we consider root locus for system design.

For stability, we can check if some systems are stable of unstable by drawing root locus. The root locus give us insight of stability. If root locus is in right side in complex plane, then system is unstable. And also, to satisfy design condition, we can know how we select the real value or imaginary value. In systems, if we want to improve steady-state error, transient response or something like that, just changing gain is insufficient. For improvement stability, sometimes we must change the root locus. To change root locus, we have to add zero of pole in system. Adding a pole at origin can improve steady-state error. Adding a zero can improve transient response. Adding a pole in origin means system's transfer function is divided by 's' term. 's' term in denominator means integral in time domain. This is why we use Integral terminology. Similarly, adding a zero means 's' term is multiplied to transfer function and it means derivative in time domain. Then, why we add additional zero and pole? This is because root locus. In control engineering, root locus is used to check system characteristics. Root locus is a graphical presentation of the closed loop poles as a parameter is varied. If root locus is in the right side, then system is unstable. And also we can know how we select root locus value to find desirable response. In some cases, we can control the system by changing gain, and it is P control. But just adjusting proportional gain is insufficient. For better system response, we have to change root locus. To do this, we add zero or pole. If we add zero of pole, then root locus' shape is changed and characteristics of system also change. These method are I and D control. Actually, we can improve steady state error from PI control and improve transient response from PD control.

If we just use on/off control, the output is fluctuating and we can't control subtly. So first, we consider Proportional term. We chose reference value and calculate error, a difference between reference value and current value. After that, we select adequate gain and multiply to error. According to error, output is controlled. In P control, we can go smoothly to reference value but there is residual steady-state error. In means that current value goes to reference closely, but still has some difference. So we need Integral term. The integral term accelerates the movement of the process towards reference value and eliminates the residual steady-state error that occurs with a pure proportional controller. This term is calculated by multiplying some gain to instantaneous sum of error. Lastly, we can improve speed of response and need a derivative term. This term is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain and improves settling time and stability of the system.

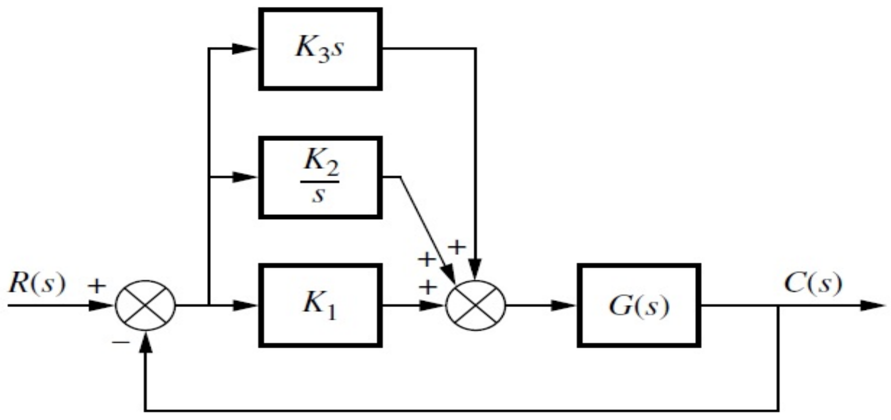


Fig. 2.3.5 block diagram of pid control

In Fig2.3.5, K1 is proportional part, K2/s is integral part, and K3s is derivative part(G(s) is plant.).

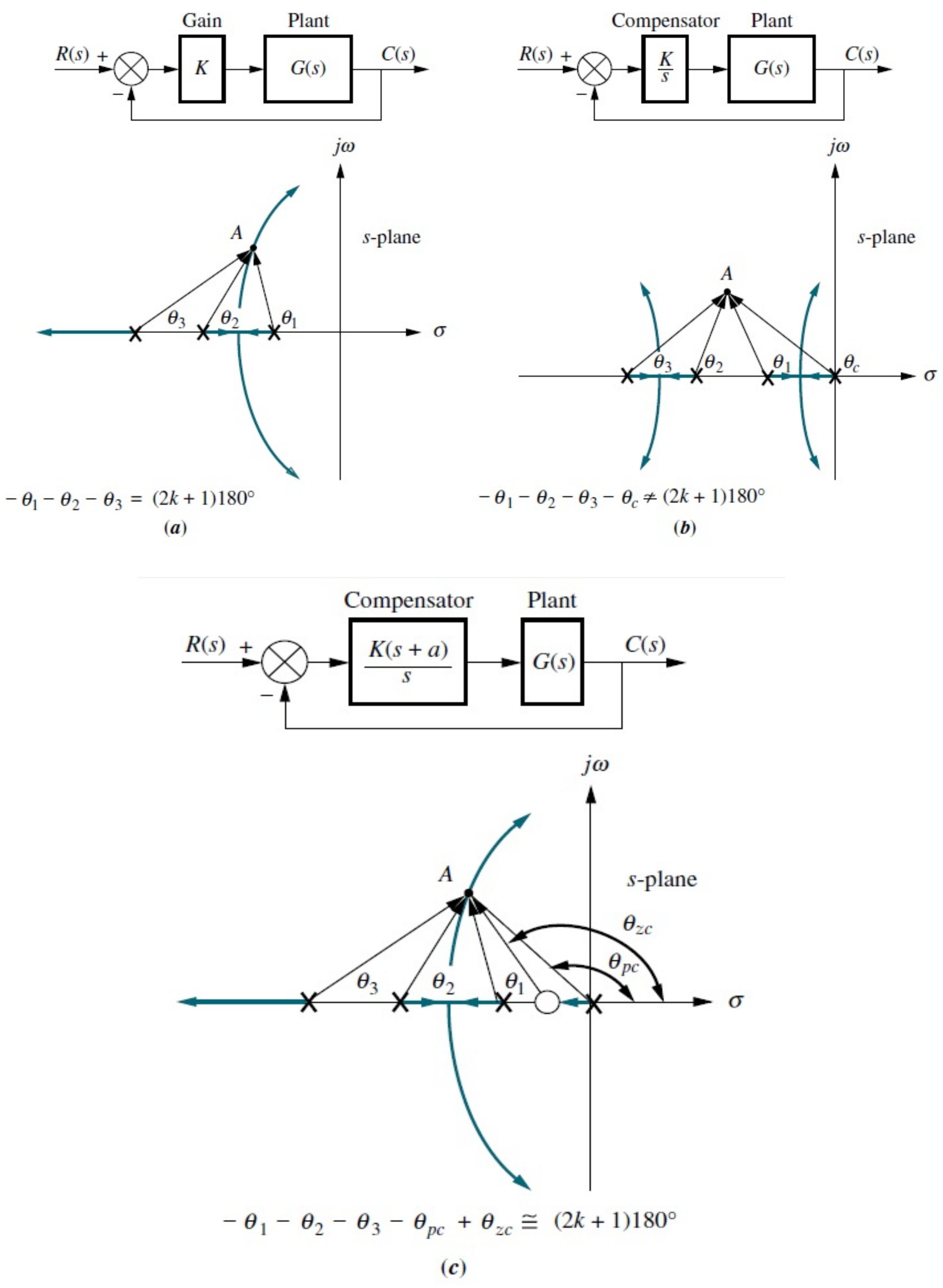


Fig. 2.3.6 Adding a zero and pole

In PI control, it places an open loop pole at the origin and system type increases by one. But if we place the pole at origin, it causes bad transient response. So to improve the steady-state error without affecting the transient response, we also add a zero close to the pole at the origin. This situation is represented at Fig.2. In (a), just proportional gain, it has pole at A. In (b), we add integral compensator to improve steady state error, but the root locus no longer goes through point A. In (c), point A is still on the root locus, and we can improve steady state error.

For improvement of transient response, we can use PD controller(derivative compensation). One way to speed up the original system is to add a single zero to the forward path. Judicious choice of the position of the compensator zero can quicken the response over the uncompensated system.

For improvement of steady state error and transient response, PID controller design is needed. PID controller, it has both of integral compensator and derivative compensator. There are some way to gain parameter P, I, and D, but we use trial and error way.

This is background of PID control and we will look at the PID control of motor in computer by using C language.

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant Kp. If the proportional gain is too high, the system can become unstable. The accumulated error is then multiplied by the integral gain Ki. The integral term accelerates the movement of the process and eliminates the residual steady-state error that occurs with a pure proportional controller. The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain. Derivative action predicts system behavior and thus improves settling time and stability of the system.

2.4 Atmega 128

2.4.1 Overview of atmega128

The Atmel AVR ATmega128 is high performance but low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. It has 133 instruction set, and usually it operates in one cycle. And also has 32 general purpose working registers, 4Kbytes EEPROM, 4Kbytes SRAM, 53 general purpose I/O lines, and so on.

For user, it contains two 8bits timer/counter and two 16bits timer/counter. Using these timer, we can make some clock that has various intervals. To communicate, USART port can be used and also there is I2C communication. If we add Bluetooth module, we can Bluetooth communication. It contains the interrupt vectors and provide interrupt service routine. This can be used with timer and timer overflow scheme for a action which is planned by users. Atmega128's pin configuration and internal block diagram are like below.

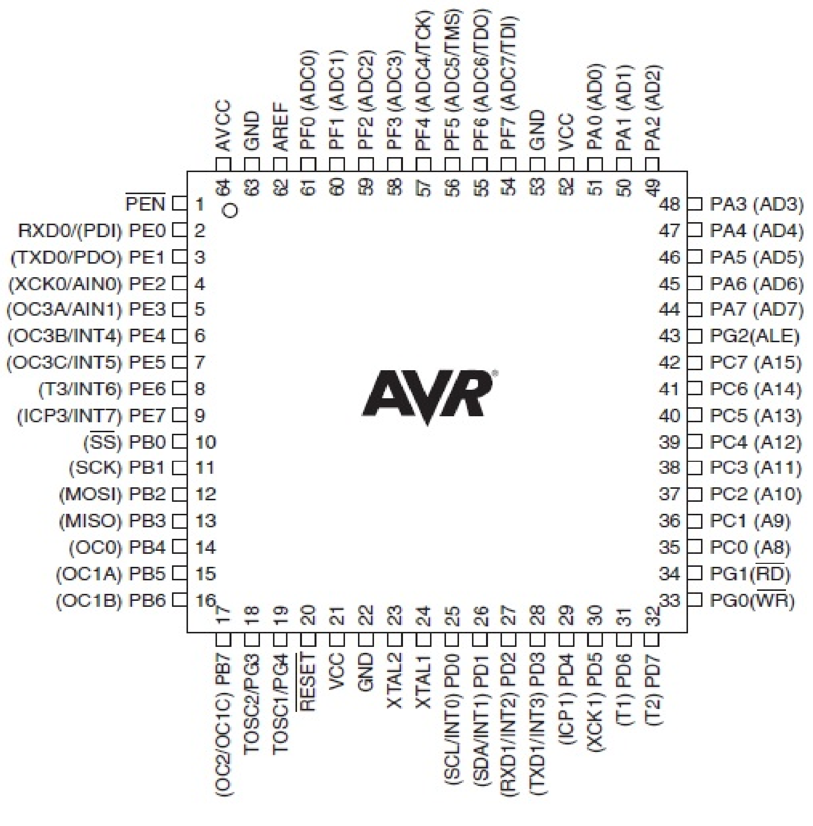


Fig. 2.4.1.1 pin information of Atmega128

Each pin has different role, for example, the pin PD0(SCL) and PD1(SDA) can be used in I2C communication.

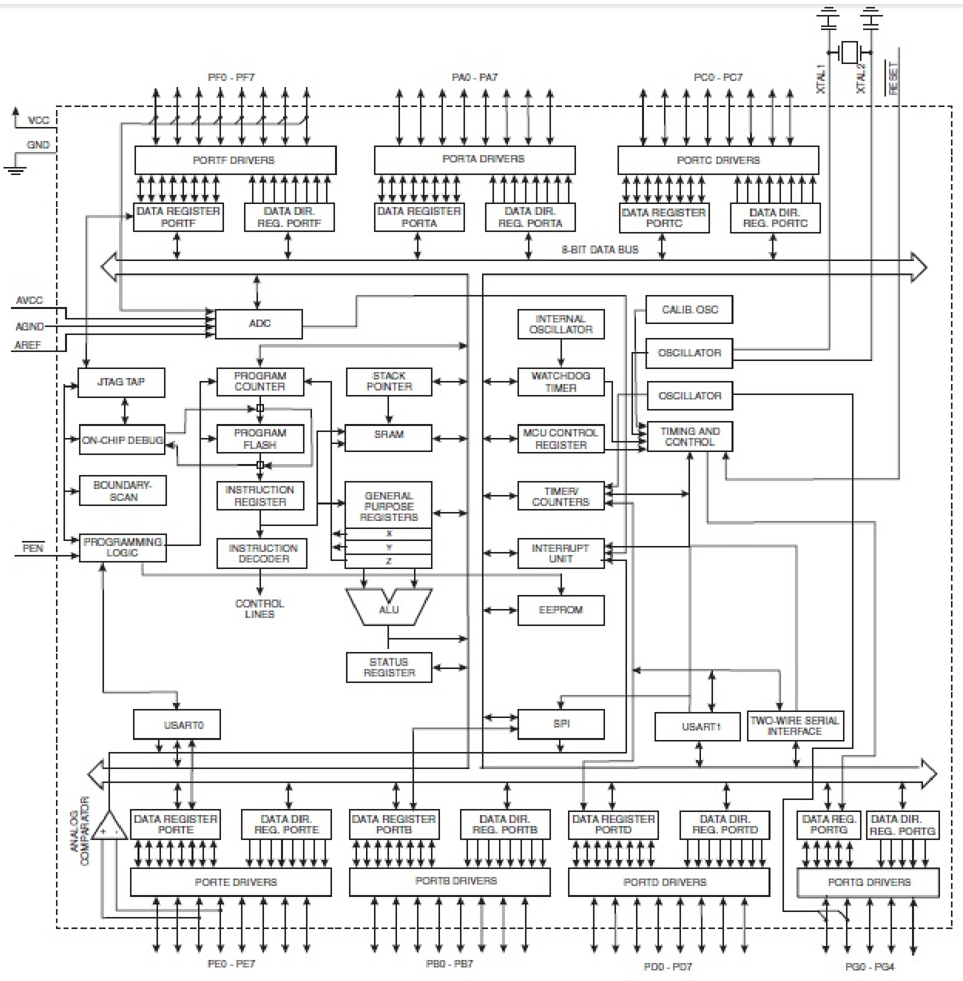


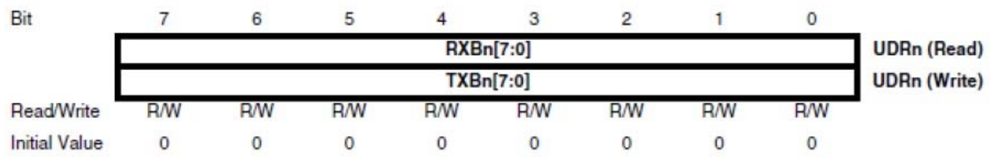
Fig. 2.4.1.2 internal block diagram of Atmega128

2.4.2 Memory of Atmega128

It has two type memory, one is program memory and the data memory. The program memory has internal 128KB flash memory, and address of memory is 16bits. This can be divided two section – Application Flash section, boot flash section. By using SPI ISP, we can easily write program. Data memory has SRAM as internal data memory, external data memory, internal EEPROM. The EEPROM can be accessed through register.

2.4.3 Memory of Atmega128

Atmega128 has two USART circuits. USART means Universal Synchronous and Asynchronous serial Receiver and Transmitter. This is a program that handle serial communication by using micro chip. It has specifications – Synchronous and Asynchronous serial mode, Full duplex, high accurate baud rate, noise filtering and so on. The USART date frame format has bits from 7 to 13 which is consist of start bit, data bit, parity bit, and stop bit. If we want to use USART, we must know the usage of register of USART.



Bit	7	6	5	4	3	2	1	0	
	RXBn[7:0]								UDRn (Read)
	TXBn[7:0]								UDRn (Write)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Table. 2.4.3.1 register UDRn

Above is UDRn(Usart I/O Data Register) and it contains data. According to mode of Rx or Tx, data is transmitted from TxBn or stored to RxBn. There are many other register like USCRnA, USCRnB, USCRnC, UBRRnH/L and we can select operation mode and set the parity bit, stop bit, baud rate, data length of transmission according to register value.

3. Modeling Methodology

3.1 Modeling scheme

3.1.1 1st model

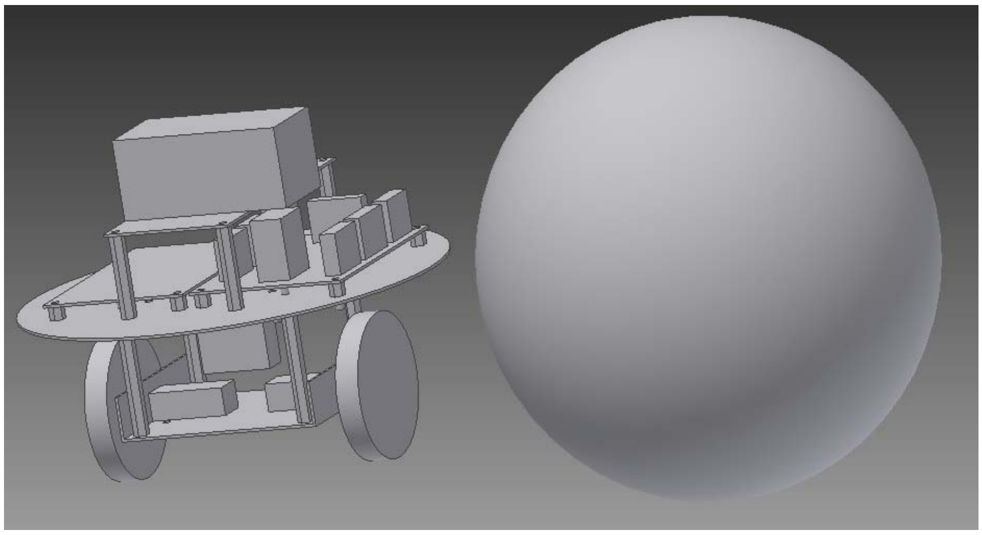
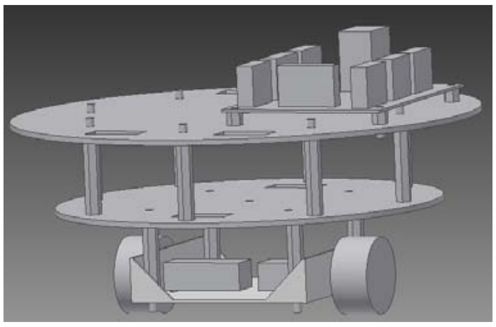


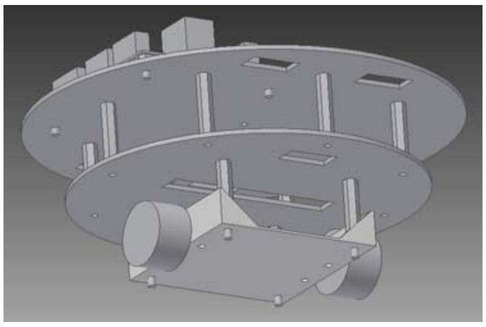
Fig. 3.1.1.1 first model in CAD

We first designed hardware like above(Fig.3.1.1.1) using the CAD program. In the middle, there is a large circle acrylic plate with MCU, Motor driver, gyro sensor and any other device locate on this plate. In the below, another plate shaped clamp is connected large circle plate and motor.

3.1.2 2nd model

For stability, we added some small but heavy things in low position and also changed a wheel as a small one. It makes the center of mass lower. The second model is in below images.





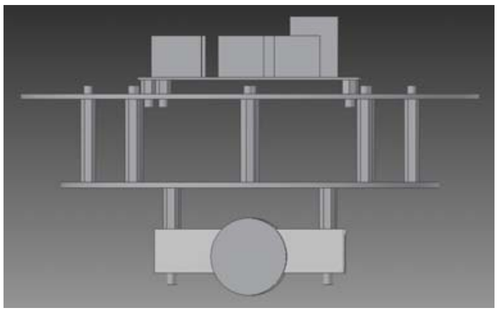
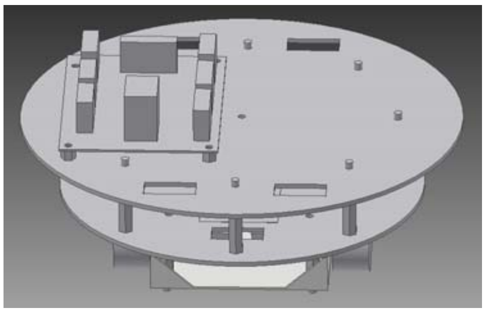
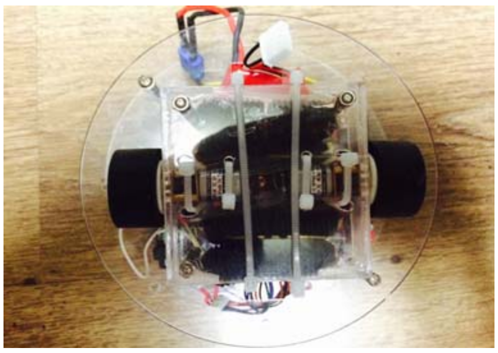
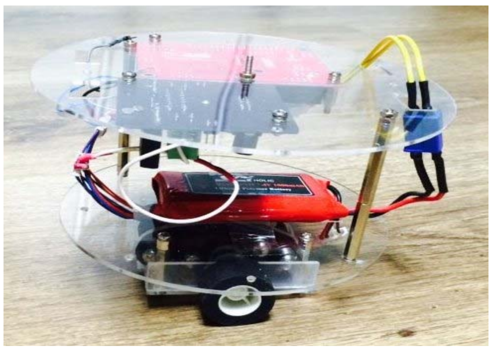
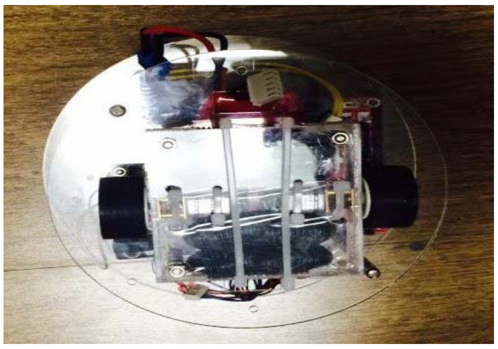



Fig. 3.1.2.1 second model in CAD

In this design, we tried to locate almost all weights of hardwares below the hemisphere.

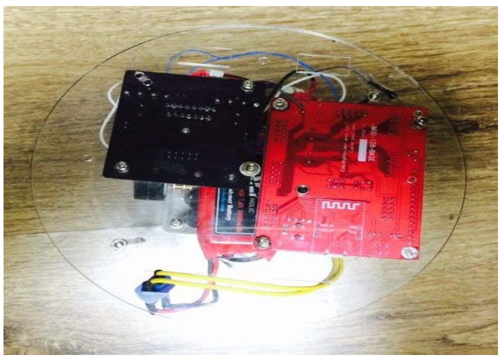
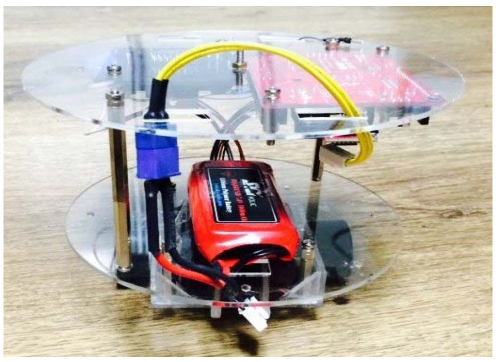



Fig. 3.1.2.2 second model's pictures

We had two circle plates and an above one is matched in inner surface of sphere, it gives stability to our model. And we weighted heavy matter for stable motion. You can see this at Fig3.1.2.3.

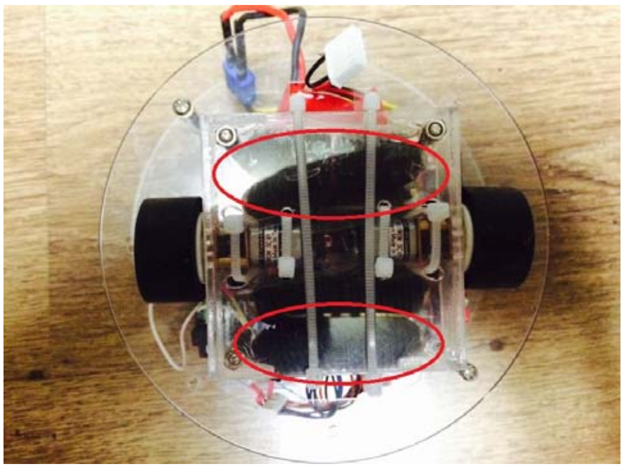


Fig. 3.1.2.3 heavy materials for stability

With the 2nd model, we didn't consider the gyro sensor. Because we realized that the most important thing in the rolling robot is to balance itself with the weight. So we added something heavy on the motor, tried to make it heavy so that the central point of the weight may be below the hemisphere. We also coded the program using the theory of the torque control of DC motor using velocity profile based acceleration/deceleration control and we can control it with application of Bluetooth controller using android phone. The reason we used this theory is that when we just controlled it with simple torque control and we commanded it to stop, it was trying to fall down forward due to the effect of the inertia. So what we came up with is to use the acceleration and deceleration control. With this theory, we could control it stably with minimizing the force of the inertia.

3.1.3 3rd model

Finally, we added small and heavy metal sphere like pendulum. It imitates a cheetah's tail to balance robot. According to how robot inclines to left or right, the sphere is moving opposite side to recover balance. This mechanism is controlled by servo motor and we have to know the angle value by using gyro sensor. The changed model which is add servo motor, gyro sensor, and metal sphere is like below.

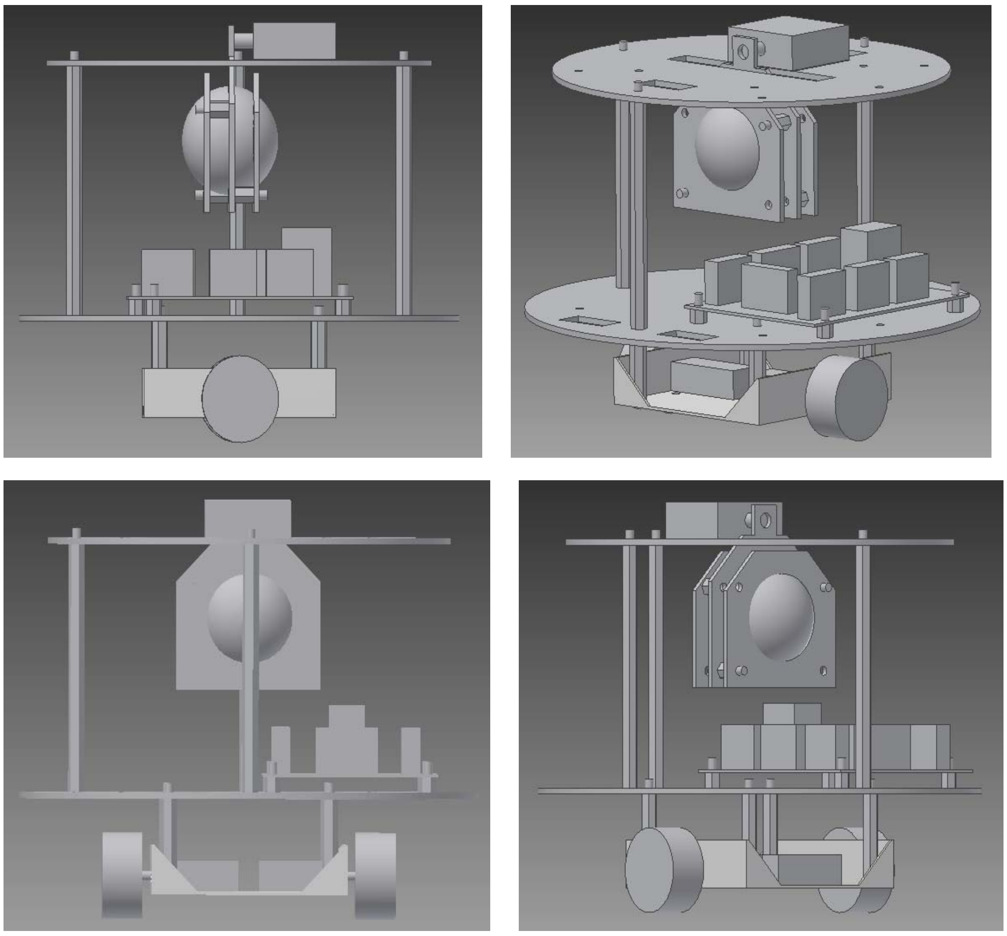


Fig. 3.1.3.1 third model in CAD

The next figure shows the balance mechanism in our model.

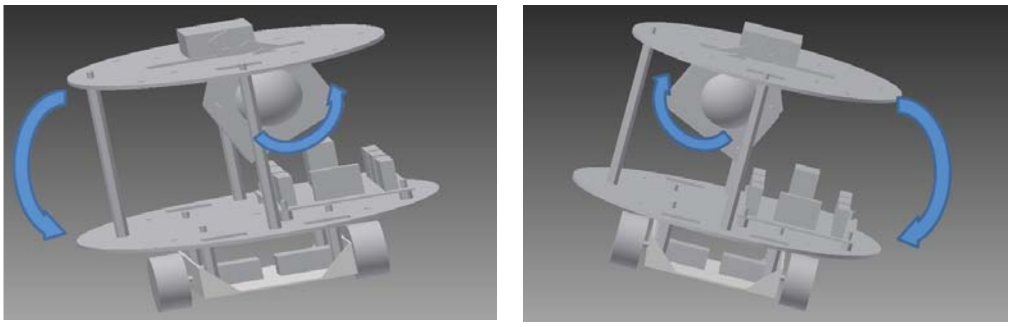


Fig. 3.1.3.2 left-right balancing process

From above modeling, we made final model.

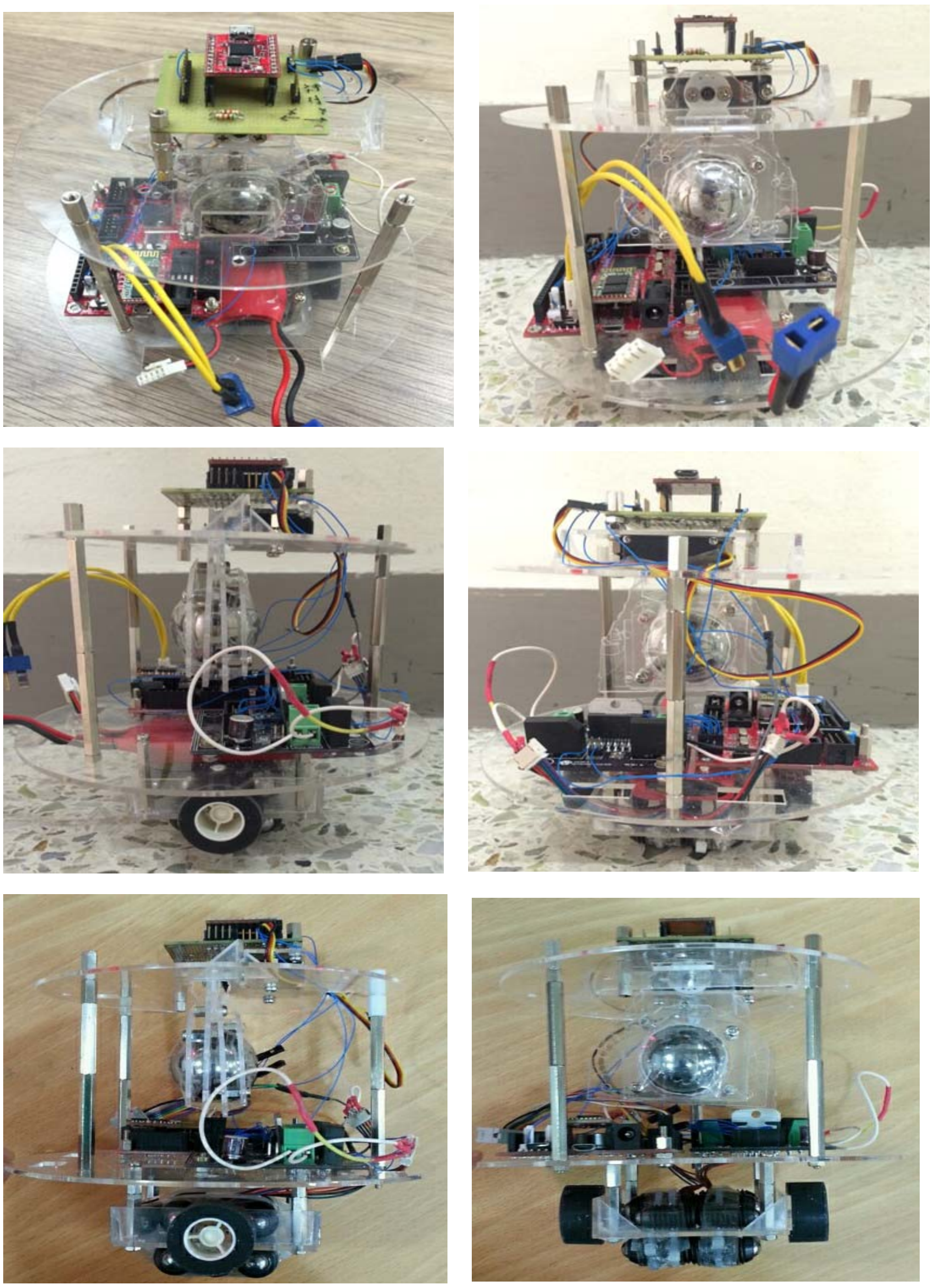


Fig. 3.1.3.3 third model's pictures

3.2 Components of model

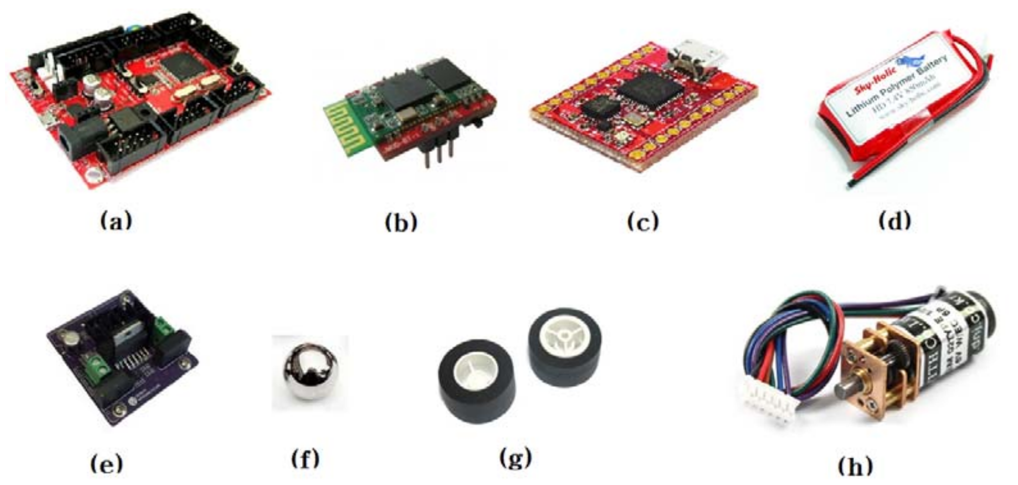


Fig. 3.2.1 Components

The Figure 3.2.1 shows the components that are used in our model

(a) is MCU, JMOD-128-BASE

(b) is Bluetooth module, JMOD-BT-1

(c) is gyro sensor, myAHRS+

(d) is battery, Sky-Holic Lithium Polymer Battery HD 7.4V 850mAh

(e) is DC motor driver, DCM-A1

(f) is metallic sphere, diameter 35mm

(g) is wheel, Line Tracer wheel

(h) is DC motor, RA-12WGM + Encoder 02TYPE (6V)   / reduction ratio=1/298

4. Balancing Methodology

4.1 Back-Forth

4.1.1 Acceleration / Deceleration

For balancing Back-Forth, we take Acceleration and Deceleration method. When robot goes to back or forth from stop state, robot's speed increases or decreases gradually. PWM is effective. In C language, we set the motor-on-variable that sends a on signal to motor. The value of this variable, the operation time of motor can change. We made 10 modes to handle several situations that could happen while it is moving and the below shows those happenings. Each mode motor-on-variable has different value at different situation and it can make speed of motor change gradually

When the command is like this:

Forward : stop → Forward / Forward → Forward / backward → Forward

Backward : stop → Backward / Forward → Backward / backward → Backward

Left : Forward→decrease only left motor / Backward→decrease only right motor

Right : Forward→decrease only right motor / Backward→decrease only left motor

4.1.2 PID control by using Gyro sensor

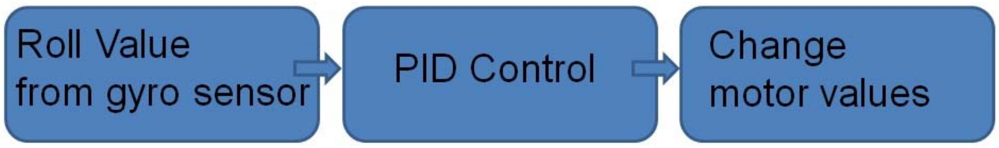


Fig. 4.1.2.1 block diagram of back-forth balancing

To balance Back-Forth, we can also use gyro sensor. From gyro sensor, we get the Roll value and we make this roll value zero. To make our wanted value, PID control can be used. From the Roll value, we know models inclined towards back or forth, then we give back or forth order to motor.

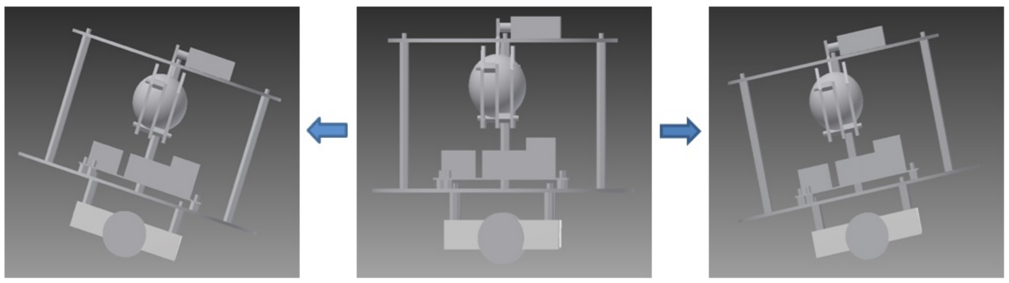


Fig. 4.1.2.2 back-forth balancing process

The PID control handles a continuous value(analog) but computer's input and output is not analog, so we take a sampling method. From this method, the added value is like below.

$$K_p \times error + K_i \times accumulative\ \ error + K_d \times error\ difference$$

Error is easily calculated from reference and some value from sensor or motor encoder value. In C language, PID controller's Pseudocode likes below.

```
 error = desired value - present value;
            P = Kp*error;
            I = Iterm_old + Ki * (dt/2) * ((error + previous_error) - windup);
            D = Kd*(error – previous_error);/dt
            PID = P +I + D;

        if (PID > lim)    PID_limit = limit;
            else if (PID < -lim)    PID_limit = -limit;
            else PID_limit= PID;

            //anti wind up
            windup = 1/Kp *(PID - PID_limit);
            previous_I = I;
            error_old = error;

return PID;
```

In code, P is proportional, so Perm is multiplication of Kp and error. I is integral, so error is added with some parameter. D is derivative, difference of error and old error is scaled with parameter. In I term, the wind up is existed. Integral windup refers to the situation in a PID controller where a large change occurs and the integral terms accumulates a significant error during the rise. So we reduce the I term with some value, 'windup'. The below picture shows the initial model and we succeeded balancing on the ground .. However there were a little bit of fluctuation in the move, so we tried to figure it out.

To make it balance more stable, first we needed to balance the weight of itself without any other forces. So what we have done next is to design it in CAD, the thing in that time is to add each devices' weight so that we could find the central point of the all weight and this must be on the center of the robot. Modified models is in 3.1.2 and 3.1.3.

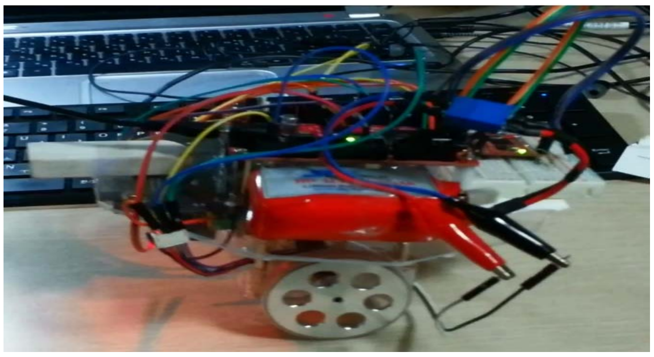


Fig. 4.1.2.3 self-balance picture

4.2 Left-Right Balance

Since it was our first time to use myahrs+ module as gyro sensor, we had to know how it sense the change of the angle. First, we used the test program, 'mymotoioncc' offered by Withrobot company. Second, we did experiment of receiving the data from myahrs+ to ATmega128.

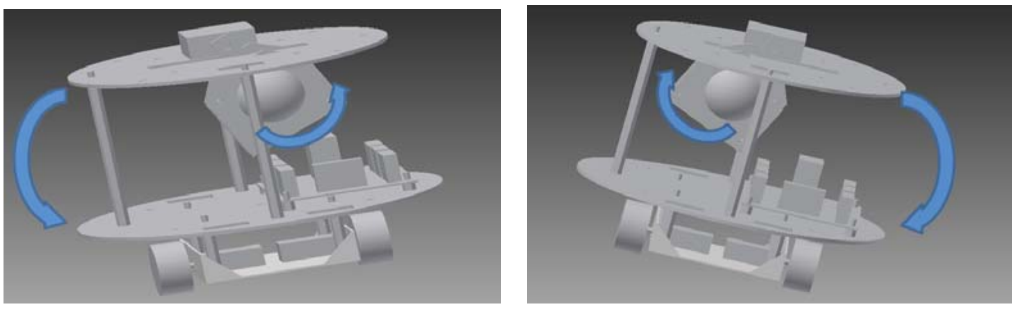


Fig. 4.2.1 left-right balancing process

As we discussed in figure 4.2.1, To balance left-right, we imitated cheetah's tail. According to gyro value, the metal sphere that is connected to servo motor is moving left or right. In this time, PID control is also needed to control the angle of servo motor.

4.3 Myahrs+ simulation

4.3.1 Test program : mymotioncc

The program is offered by withrobot producing myahrs+ and other sensors. With this program, we just needed to connect the computer and myahrs through UART-USB converter and this connection is accomplished by the below method figure4.3.1.1 and table 4.3.1.2 shows.

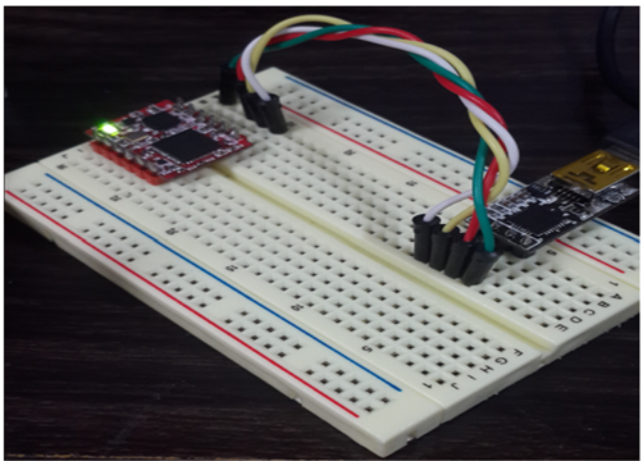
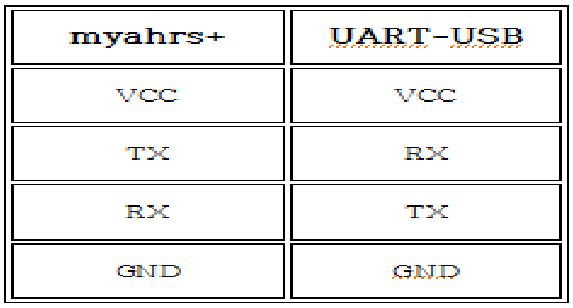


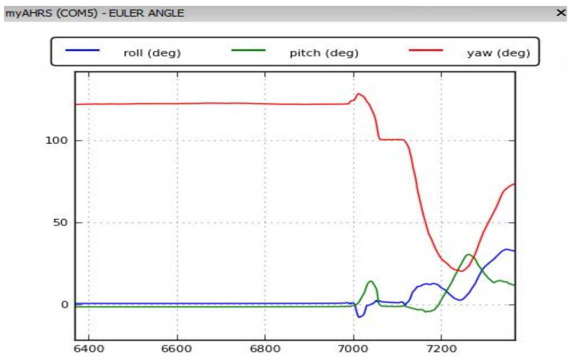
Fig. 4.3.1.1 Myahrs and UART-USB connection

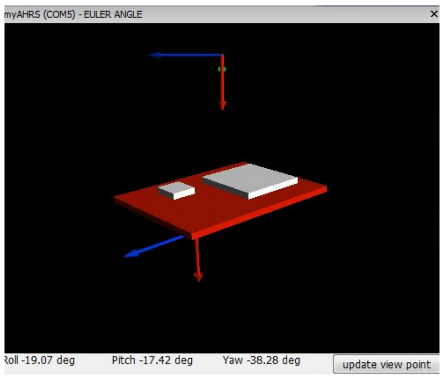


myahrs+	UART-USB
VCC	VCC
TX	RX
RX	TX
GND	GND

Fig. 4.3.1.2. Method of connection between myahrs+ and UART-USB

In this fashion, we could find out the data from the sensor, myahrs+ about pitch, roll, and yaw in degree. Figure 4.3.1.3 shows how it works





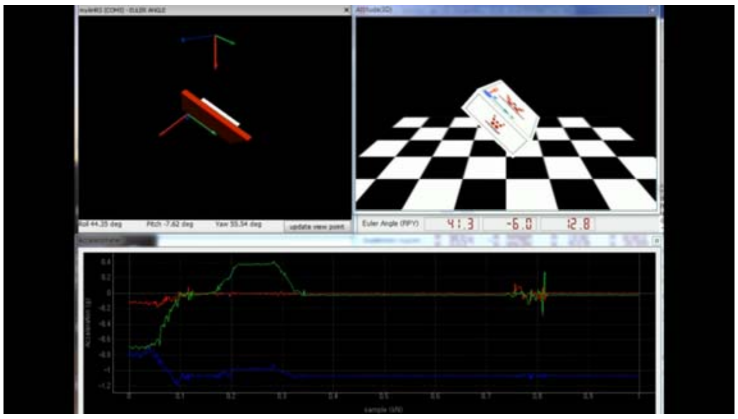


Fig. 4.3.1.3　3 dimensional data from myahrs+

As it shows, myahrs+ offer 3 dimensional data which is composed of roll, pitch, and yaw. The left figure shows the graph of data and the right one is actual myahrs+ which demonstrate its movement and angle change based on 3 dimension. The bottom one is the entire program.

4.3.2 Test myahrs+ with ATmega128

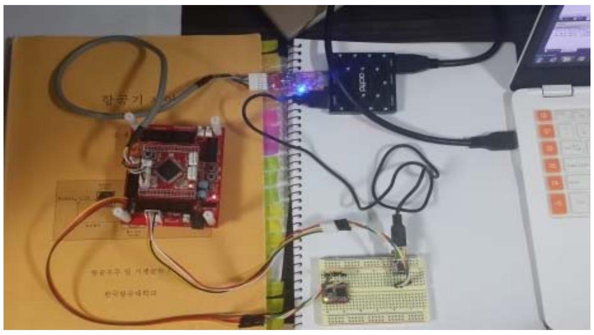
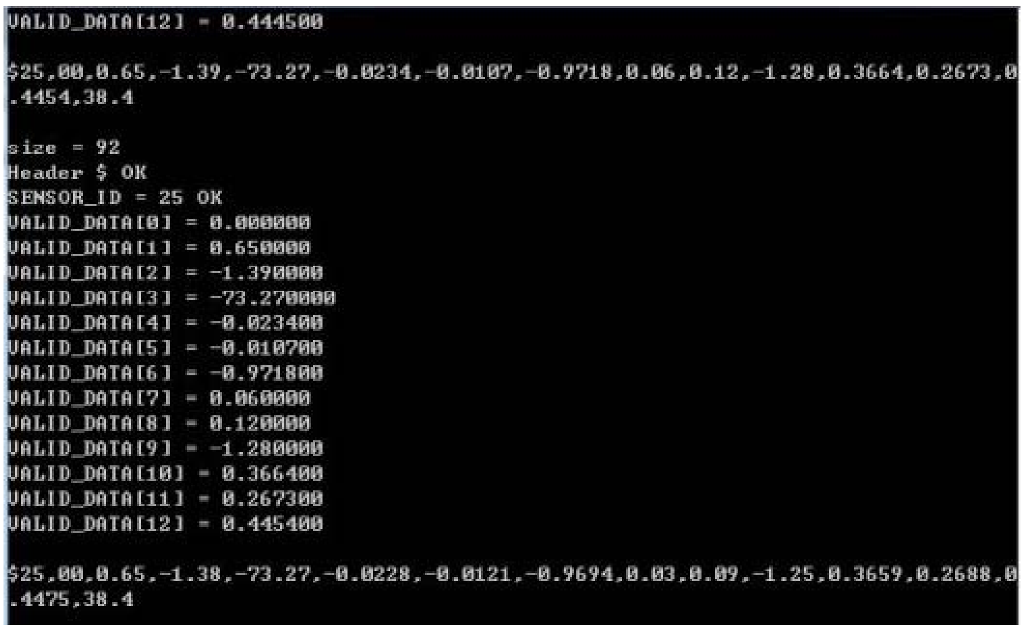


Fig. 4.3.2.1 Connection with ATmega128 on AVR studio

Figure 4.3.2.1 shows how myahrs+ is connected with ATmega128. Since our main MCU uses ATmega128, gyro sensor should be connected with it and communicate with it to control the movement and balance of a rolling robot. To do this, we needed to test how it communicate each other and especially, the principle of how myahrs+ senses the change and sends that to MCU is very important, so we did this experiment by coding C.



```
VALID_DATA[12] = 0.444500

$25,00,0.65,-1.39,-73.27,-0.0234,-0.0107,-0.9718,0.06,0.12,-1.28,0.3664,0.2673,0
.4454,38.4

size = 92
Header $ OK
SENSOR_ID = 25 OK
VALID_DATA[0] = 0.000000
VALID_DATA[1] = 0.650000
VALID_DATA[2] = -1.390000
VALID_DATA[3] = -73.270000
VALID_DATA[4] = -0.023400
VALID_DATA[5] = -0.010700
VALID_DATA[6] = -0.971800
VALID_DATA[7] = 0.060000
VALID_DATA[8] = 0.120000
VALID_DATA[9] = -1.280000
VALID_DATA[10] = 0.366400
VALID_DATA[11] = 0.267300
VALID_DATA[12] = 0.445400

$25,00,0.65,-1.38,-73.27,-0.0228,-0.0121,-0.9694,0.03,0.09,-1.25,0.3659,0.2688,0
.4475,38.4
```

Fig. 4.3.2.2. The result of communication between ATmega128 and myahrs+

The above Figure is the result of the test and we used UART0 as communication path. The data from the sensor are saved into the buffer we assigned in sequence. From having done this experiment, the data is composed of header and PRY data. Each one is separated by the specific sign, in this case it is null or enter.

5. Controlling Methodology

5.1 Bluetooth controller application

5.1.1 Application

Bluetooth controller we were using is an android application offered at google store for free. Figure 4.1.1 shows that the screen of the bluetooth controller showing several keys: forth, back, left, right, speed-up, speed-down, and stop.

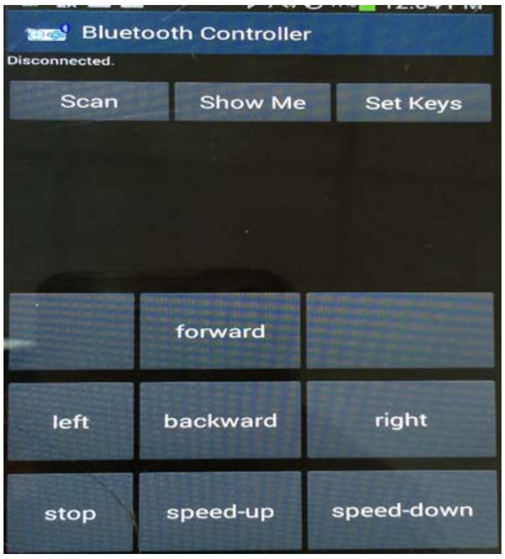
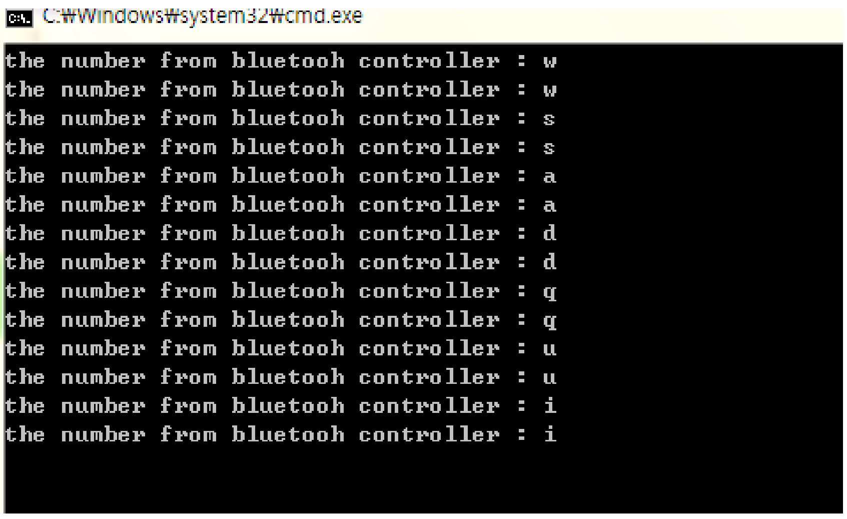


Fig. 5.1.1. bluetooth controller screen shot on the smart phone

Seeing figure 5.1.1, ‘scan’ key is to connect the application with the bluetooth module on the robot.

5.1.2 Test and result



```
C:₩Windows₩system32₩cmd.exe
the number from bluetooh controller : w
the number from bluetooh controller : w
the number from bluetooh controller : s
the number from bluetooh controller : s
the number from bluetooh controller : a
the number from bluetooh controller : a
the number from bluetooh controller : d
the number from bluetooh controller : d
the number from bluetooh controller : q
the number from bluetooh controller : q
the number from bluetooh controller : u
the number from bluetooh controller : u
the number from bluetooh controller : i
the number from bluetooh controller : i
```

Fig. 5.1.2. The result of bluetooth controller communication

Seeing the above figure, when we clicked each command, it sends the data to the compter by bluetooth, the computer displayed the data exactly. The actual communication path is like this: bluetooth controller application → bluetooth module on MCU → MCU → Computer.

5.2 The number of fingers

5.2.1 Theory of how to count the fingers

Computer vision is in many ways the ultimate sensor, and has endless applications to robotics. This is why we chose it as one of controlling methods. The basic idea of this is that we assigned the command in advance like this: when the number of finger is 1, it means forth, 2 is for back, 3 is for left, 4 is right, and 5 is stop. Figure 5.2.1.1 shows the flowchart of the entire process, how this program works.

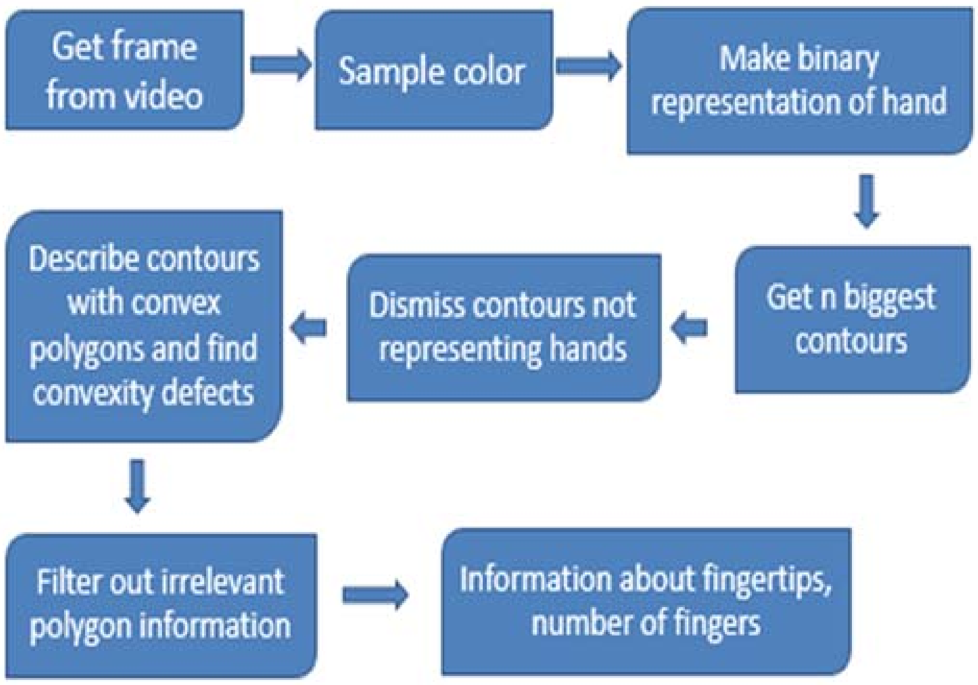

Get frame from video
Sample color
Make binary representation of hand
Get n biggest contours
Dismiss contours not representing hands
Describe contours with convex polygons and find convexity defects
Filter out irrelevant polygon information
Information about fingertips, number of fingers

Fig. 5.2.1.1 Flowchart of the program counting the fingers of hand

As it shows, the basic principle of this program is using the color recognition. In other words, it detect the hand's color telling this from the background and track the hand. The program is therefore initialized by sampling color from the hand as figure 4.2.2 shows

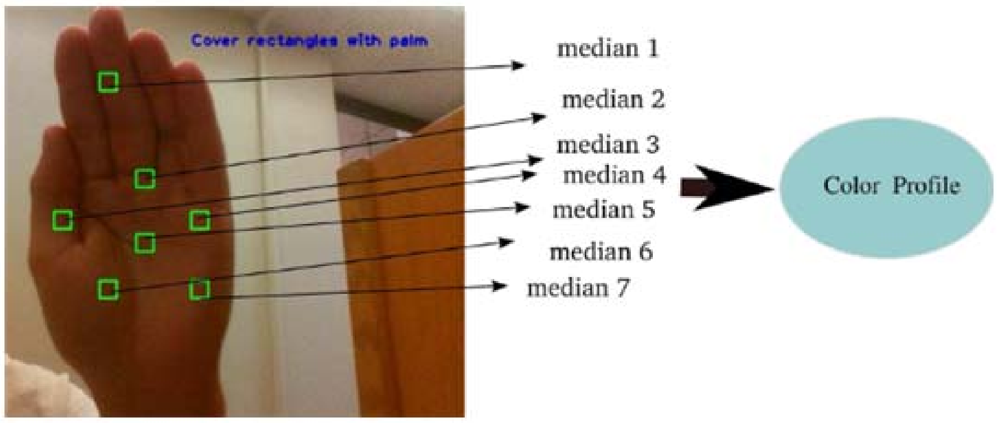


Fig. 5.2.1.2. Getting sample colors from hand

Each green rectangular thing is for getting sample color. When we put our hand covering the all 7 green things, then the hand is extracted from the background by using a threshold using the sampled color profile. Each color in the profile produces a binary image like figure 5.2.1.3 shows.

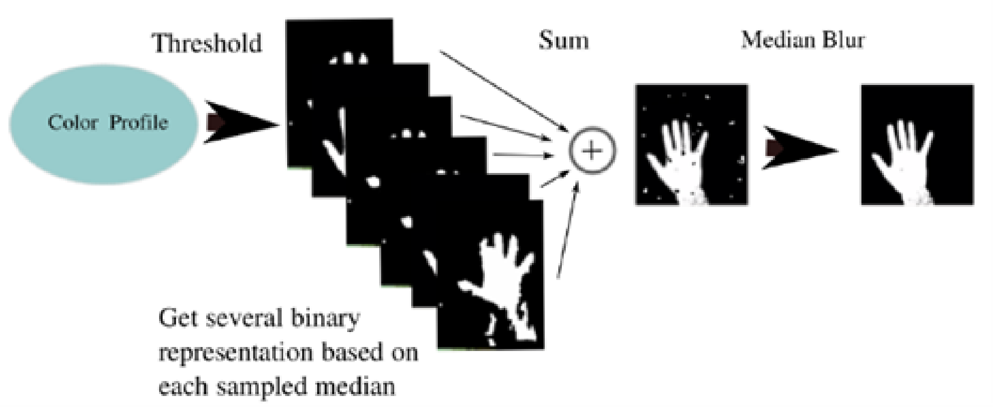

Threshold
Sum
Median Blur
Color Profile
Get several binary representation based on each sampled median

Fig. 5.2.1.3. Making a binary representation of hand

Each binary representation are all summed together and a nonlinear median filter is then applied to get a smooth and noise free binary representation of the hand. When the binary representation is generated, the hand is processed in the following way.

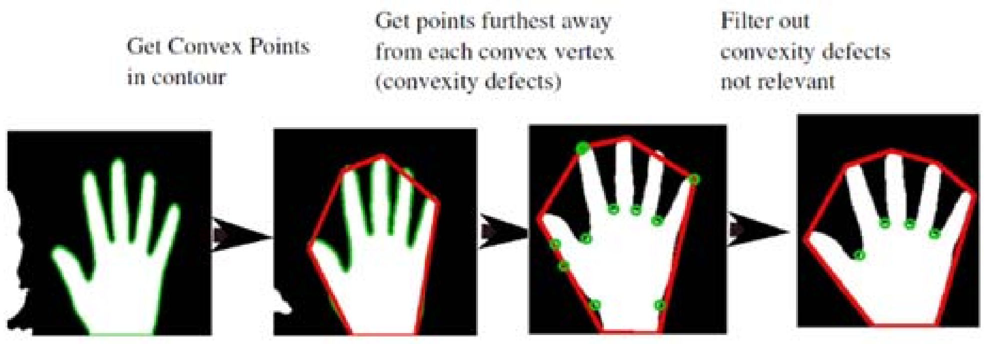

Get Convex Points in contour
Get points furthest away from each convex vertex (convexity defects)
Filter out convexity defects not relevant

Fig. 5.2.1.4. Dismissing contours not representing hands

Figure 5.2.1.4 explain the process: It gets convex points in contour of hand and get points furthest away from each convex vertex. Finally it is deleting convexity defects not relevant. The properties determining whether a convexity defect is to be dismissed is the angle between the lines going from the defect to the neighboring convex polyson vertices like figure 5.2.1.5

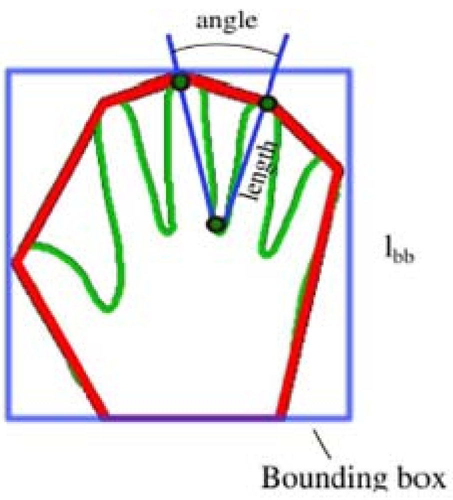
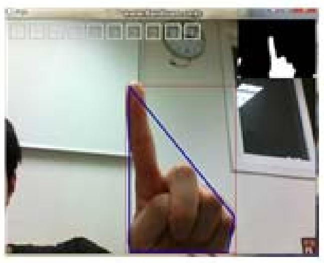
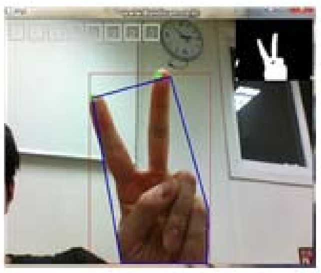


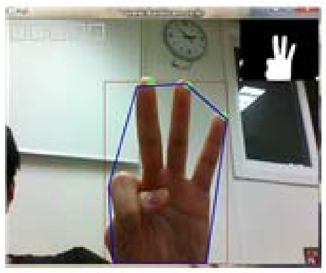
Fig. 5.2.1.5. The properties determining a convexity defect

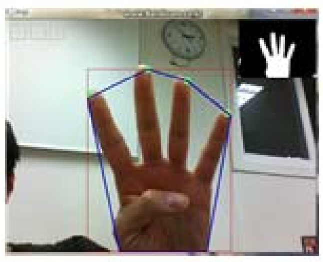
The defect is determined in this way: if the length is smaller than 0.4lbb and angle is bigger than 8°, it is defect.

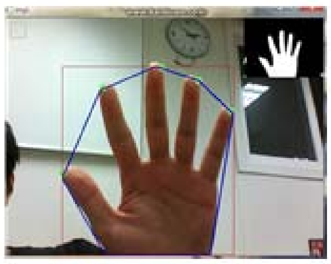
5.2.2 Test and result


1 : Forward


2 : Backward


3 : Left


4 : Right


5 : Stop

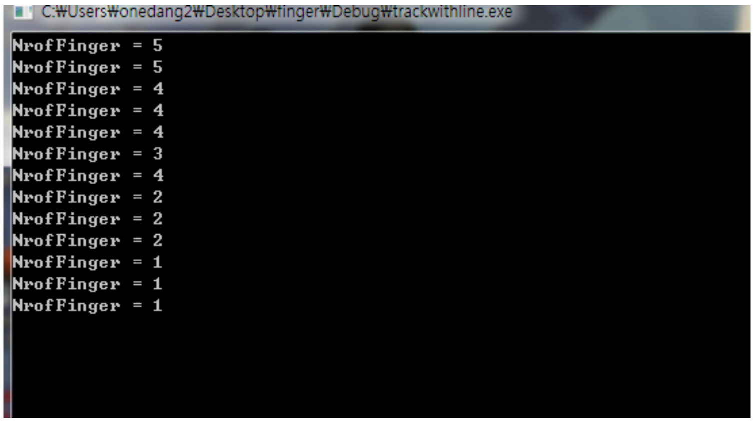

```
C:\Users\onedang2\Desktop\finger\Debug\trackwithline.exe
NrofFinger = 5
NrofFinger = 5
NrofFinger = 4
NrofFinger = 4
NrofFinger = 4
NrofFinger = 3
NrofFinger = 4
NrofFinger = 2
NrofFinger = 2
NrofFinger = 2
NrofFinger = 1
NrofFinger = 1
NrofFinger = 1
```

Fig. 5.2.2.1 Assign command on each counted number

With the above figure, we already assigned each count to five kinds of command and executing the program, we could get the counted number of the hand. Actually, the counted numbers are displayed on the screen or video that webcam shows on the monitor. The picture is a little bit small though, it is possible to notice that there is something like white little square boxes, in each square box there is a counted number.

6. Control a rolling robot

6.1 Using Bluetooth controller

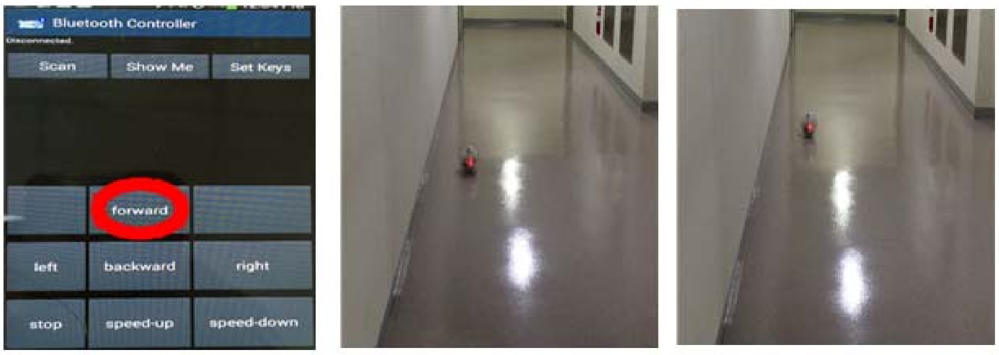


Fig. 6.1.1 Forward from Bluetooth controller

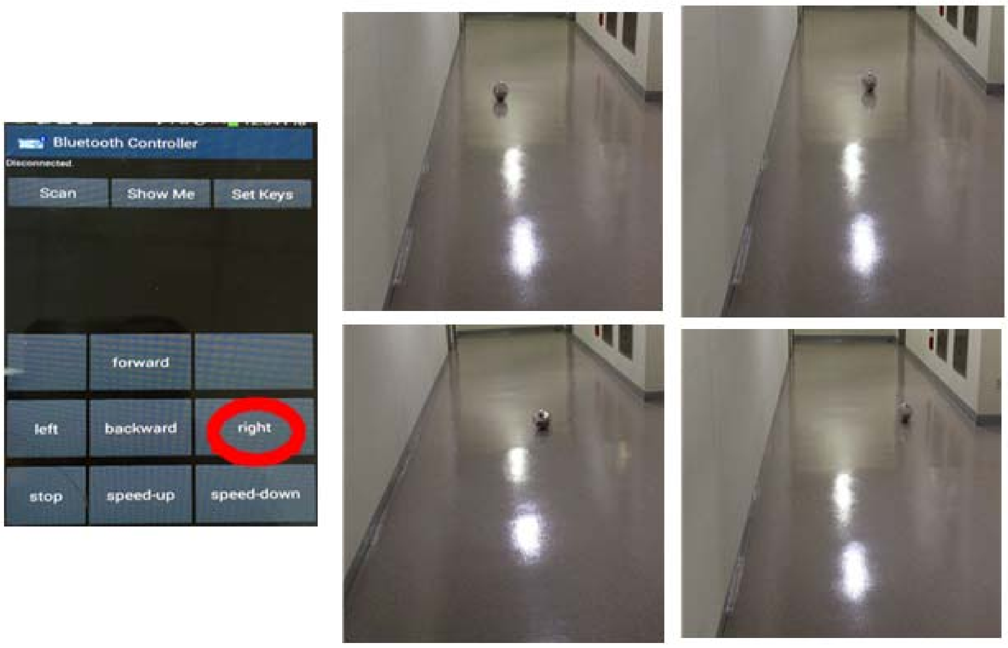


Fig. 6.1.2 Right from Bluetooth controller

6.2 Using the counted number of fingers

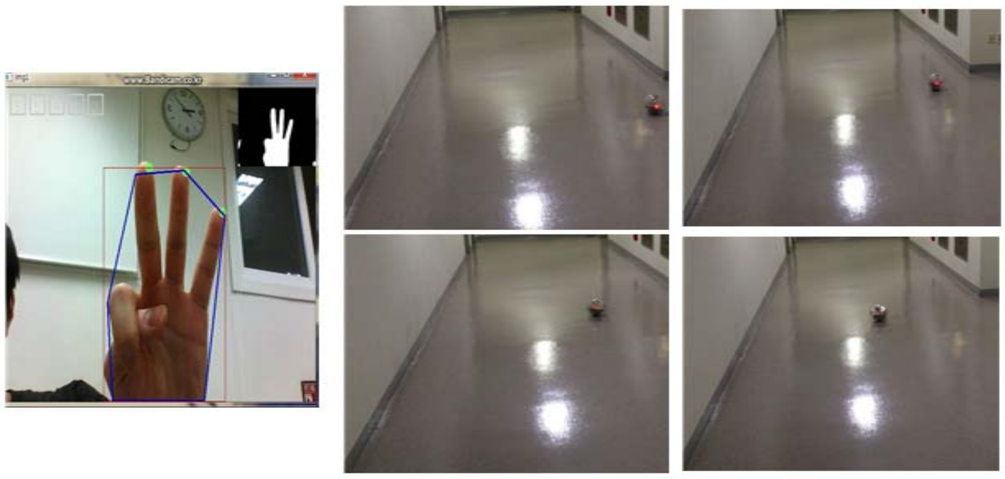


Fig. 6.2.1 Left from program counting the number of fingers

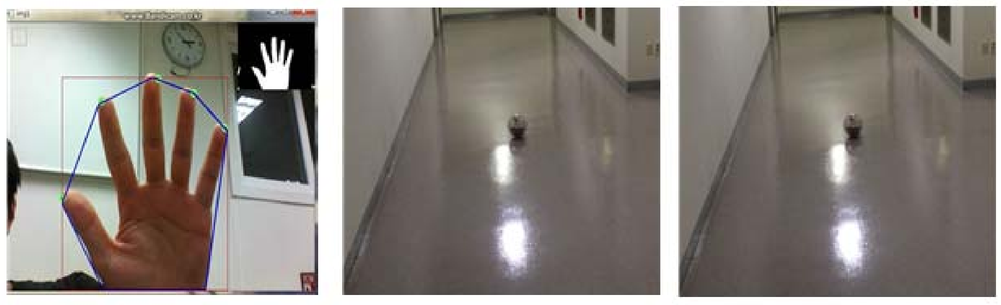


Fig. 6.2.2 Stop from program counting the number of fingers

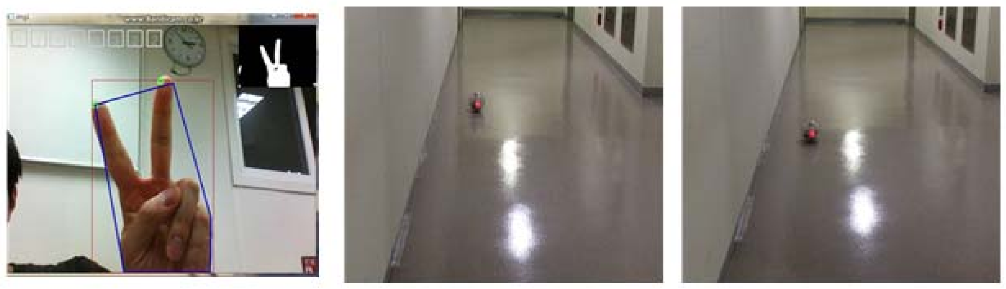


Fig. 6.2.3 Backward from program counting the number of fingers

7. Conclusion

Having started from the bottom, we designed the hardware and all the way have proceeded to code the software controlling the hardware. From our experience, it was also very hard to come up with how to control it and which algorithm suits perfectly to balance and control it, but the most difficult and stressful part is to make its own hardware. Not only it was our first time to design something like a structure of robot, but it also must be considered with everything including how it is going to be driven and balanced. Especially, the driving part is composed of two wheels, so the balance would have been considered. That's why we have spent almost 2 months for making the final structure.

As we mentioned earlier, it was also hard to come up with the idea of how to operate it more safely. After many experiments, and trial and error, the solution we used was two things: use acceleration and deceleration of DC motors and the mechanism of cheetah's tail. The first one is actually basic theory of driving any mobile robots with motors, but to be more efficient movement, we made new algorithm. Also, the second one is very famous theory, but it took us spend a long time to think how to apply this theory to our robot. In particular we had a limitation of size, because our robot is covered with an external sphere and it has 18cm diameter, and all the mass concentration should be in the bottom of the center.

After designing all the hardware and software to operate a rolling robot safely, we wanted to study more related to this robot, so that what we thought was the control method which were Bluetooth controller which offer us to control it wireless and the counted number of the hand. The first one is a little bit easy to accomplish, because the application of the smart phone was offered on google store for free. So we just used it. However, the second one is more difficult than we thought. Since the recongnization of the hand is based on the color detecting, separation the hand out of the background and tracking it is hard. To solve this problem, we extracted 7 sample colors of several points and create representative sample capable of tracking hand. Then, to count the fingers, we used the angle of how each finger is away using special calculation method.

All things considered, we have built and controlled a rolling robot. A rolling robot can move safely, in detail when it change its original direction to any one, there is very less instability like vibration or wave. Furthermore, in addition to build a rolling robot, we have made two method to control it. Even if using the application was a little bit simple, using the counted number of fingers was required a little bit skilled capability and idea, we think.

Reference

[1] 김병수, 지능형 서비스 로봇 - 산업 활성화, 한국정보통신기술협회, 2005

[2] 최병태, 노근창, 개인 서비스용 로봇 산업, HMC 투자증권, 2009

[3] 이홍석, 개인 서비스 로봇의 잠재력, 디지털 타임스, 2012. 10. 2

[4] 서현덕, 풍력 구동 메커니즘 기반의 구형로봇, 석사학위논문, 경북대학교, 전자전기컴퓨터학부, 2012

[5] myAHRS+ user manual – WITHROBOT (http://www.withrobot.com/data/)

[6] Norman S. Nise, "Control Systems Engineering",6th ed, JohnWiley&Sons , 2011.

[7] Atmega128 datasheet.

[8] 최한호, 서계원, 김재경 지음, (한번에 이해되는) AVR ATmega128 마이크로컨트롤러, 인피니티북스, 2011.

[9] Joong-Cheol Yoon, Sung-Su Ahn, Yun-Jung Lee, KisBot II : New Spherical Robot with Curved Two-pendulum Driving Mechanism (The Journal of Korea Robotics Society vol.6. 4), 2011.

[10] Jung Won Lee, Multi media spherical robot controlled by smart mobile device (Major in Measurement and Control Engineering Graduate School of Industry, Kyungpook National University), 2013.

[11] Aircraft principal axes http://en.wikipedia.org/wiki/Aircraft_principal_axes

[12] Torque Control of DC Motor Using Velocity Profile Based Acceleration/Deceleration Control, Jong-Yeon Lee and Chang-Ho Hyun, 2012

국 문 요 약

구형 로봇의 제작 및 제어

본 논문은 로봇의 안정성을 위한 메커니즘을 고려하여 구형 로봇을 구성하고 해당 로봇을 움직이는 방식 및 통신 방법에 대한 논문이다.

로봇을 구성하기 위하여 3D CAD 프로그램을 이용하여 하드웨어 자체적으로 구의 내부에서 뛰어난 안정성을 가질 수 있는 모델을 제시하고, 추가적으로 소프트웨어적인 처리를 통하여 균형을 잡는 로봇을 제작하였다. 좌우의 균형을 잡기 위하여 치타 꼬리의 메커니즘을 모방하여 비슷하게 동작하도록 하는 추로 구성된 부분이 존재하며 해당 부분은 서보 모터와 자이로 센서에서 받은 값을 토대로 각도를 조절하여 움직인다. 즉, 전체 로봇이 기울어진 방향의 반대로 추가 움직여 균형을 잡도록 한다. 여기서 원하는 각도를 맞추기 위하여 PID 제어 방식을 이용하였다.

완성된 로봇은 기본적으로 블루투스 무선 통신 방식을 이용하요 조종한다. 가장 먼저 스마트폰의 어플리케이션을 통하여 해당 로봇을 조종하였다. 추가적으로 영상인식을 통해 캠을 이용하여 손가락의 개수를 인식한 다음 인식한 손가락 개수를 블루투스 통신을 통해 로봇에 넘겨주는 것으로 해당 로봇의 조종이 가능하였다.

핵심되는 말 : 구형 로봇, 자이로 센서, PID제어, 영상인식, 블루투스 통신