Three Wheel Drive Control

Heet Sakaria

Team KJSCE Robocon

06-11-2018

Author Note

The code is still premature and optimizations are required. Interfacing the joystick to control a three wheel drive was a stepping stone towards our next project i.e Fence Following drive using Ultrasound module.

Task Description

Use joystick module to control a three wheel drive. Joystick should be interfaced using Atmega 2560 or Atmega 640.

Joystick

Pin Configuration

Pin No.	Pin Name	Description
1	Gnd	Ground terminal of Module
2	+5v	Positive supply terminal of Module
3	VRx	Voltage Proportional to X axis
4	VRy	Voltage Proportional to Y axis
5	SW	Switch

Features

Two independent Potentiometer: one for each axis (X and Y)

Auto return to center position

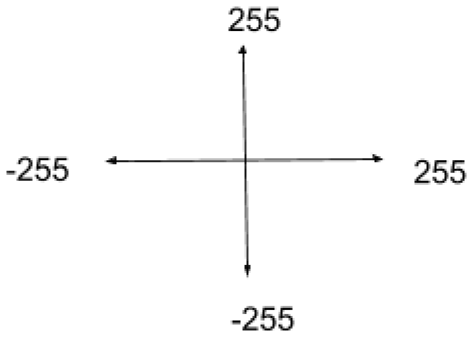
Compatible to interface with Arduino or with most microcontrollers

Method

Logic

Analog To Digital feature is used of the avr to read the joystick.

The readings are mapped to -255 to 255.



Since x and y is known, r and theta can be calculated using

$$r^2 = x^2 + y^2 \text{ and } \theta = tan{-}1(y/x)$$

Now, r is used to determine the magnitude (pwm) and theta is used to determine the direction.

After r and theta are determined, the function motor() appropriately sets the direction of the motors and returns the required pwm value.

Source Code

```
/*
 * ThreeWheelDrive.c
 *
 * Created: 22-08-2018 20:53:51
 * Author : Heet Sakaria
 */

#define F_CPU 14745600UL
#define pi 3.14159265
#include <avr/io.h>
#include <math.h>
#include <stdlib.h>
#include <util/delay.h>

void config()
{
	DDRF = 0x00; //input ADC
	DDRB =  0xFF; //direction
	DDRE = 0xFF; // pwm
	PORTB = 0x00; PORTE= 0x00;
	DDRH = 0xFF;
}

void pwm()
```

```
{
        TCCR3A |= (1 << WGM31)|(1 << COM3A1)|(1 << COM3B1)|(1 << COM3C1);//FAST
PWM-ICRn & NON-INV
        TCCR3B |= (1 << WGM32)|(1 << WGM33)|(1 << CS30)|(1 << CS32);//FAST
PWM-ICRn & PRESCALAR - 1024
        ICR3 = 255;
        PORTE = 0x00;
        OCR3A = 0; OCR3B = 0; OCR3C = 0;
}

uint8_t adc(uint8_t channel)
{
        ADMUX &= 0x00;//cleaning channel
        channel = channel & (0b00000111);//setting channel into binary
        ADMUX |= (ADMUX | channel);
        ADMUX |= (1<<REFS0) | (1<<ADLAR); // AVCC AND LEFT SHIFT
        ADCSRA |= (1 << ADEN)| (1<<ADSC)
|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADIF); //ENABLE//SINGLE
CONVERSION//PRESCALE - 128
        uint8_t a = ADCH;
        while(!(ADCSRA & (1 << ADIF)));
        return a;
```

```
}


int motor(double v, int n) //sets the direction and returns the pwm of the given motor
{
        int c = (2*n)- 1; //clockwise : 10
        int ac = 2*(n-1); //anticlockwise : 01
        double error=3.0;


        if(v>error)
        {
                PORTB |= (1 << c);
                PORTB &= ~(1 << ac);
        }
        else if(v<-error)
        {
                PORTB |= (1 << ac);
                PORTB &= ~(1 << c);
                v =-v;
        }
        else
        {
                PORTB |= (1 << c)|(1 << ac);
```

```
	}

	return (int)v;
}

int main(void)
{
	config();
	pwm();

	uint8_t x = 0; uint8_t y = 0;
	double xt = 0; double yt = 0;
	double ref = 129; //x center = 129//y center = 130
	int angle = 0;

	while(1)
	{
		y = adc(0); //Y-axis at ADC0
		x = adc(1); //X-axis at ADC1

		yt = y - ref - 1; // converting (0 to 255) to approx(-129 to 129) Y-AXIS
		xt = x - ref; // converting (0 to 255) to approx(-129 to 129) X-AXIS
```

```
        double theta = round(atan2(yt,xt)*(180/pi)); // calculating theta : deg(-180 to 180 )
        double r = sqrt(pow(xt,2)+pow(yt,2)); //calculating r :(0 to 129)
        double speed = r *(255/ref); //mapping


        double va = round(speed*(cos((angle - theta)*(pi/180))));
        double vb = round(speed*(cos(((angle + 120) - theta)*(pi/180))));
        double vc = round(speed*(cos(((angle + 240) - theta)*(pi/180))));


        OCR3A = motor(va,1);
        OCR3B = motor(vb,2);
        OCR3C = motor(vc,3);
    }
}
```

Results

The program works well and gives the drive, mobility in every direction in ideal conditions.

However, due to old motors, friction and other physical factors, the desired output may be difficult to achieve. In such case, one should find appropriate pwm to be given to the motor by trial and error.