

Teknisk rapport

Examinationsprojekt VT 2015



Viktor Rusnak

Michael Nilsson

Innehåll

[Inledning 0](#_Toc428347253)

[Teori 0](#_Toc428347254)

[Metod 1](#_Toc428347255)

[Sensorn (M) 1](#_Toc428347256)

[PWM (V) 2](#_Toc428347257)

[Regleringen (V+M) 3](#_Toc428347258)

[Kommunikation mellan Arduino och Matlab(M) 4](#_Toc428347259)

[Sammanfattning och diskussion (V+M) 4](#_Toc428347260)

[Källförteckning 5](#_Toc428347261)

# Inledning

Uppgiften i examinationsprojektet var att skapa ett program i C som kan reglera en pingisboll på en lutande arm. Detta skulle vidare regleras och plottas med hjälp av Matlab.

C-programmet består av två delar. Den ena delen behandlar regleringen och avläsningen från sensorn. Den andra delen tar hand om kommunikationen mellan Matlab och Arduinokortet. Ett realtidsoperativsystem hanterar vidare de olika delarna, för att de båda delarna ska kunna köras parallellt. Högst prioritet har regleringen och avläsningen från sensorn, medan kommunikationen har lägst.

Kommunikationen mellan arduino och matlab sköts seriellt via USB-sladd. Kommunikationen har två syften. Det första är att skicka ett börvärde från matlab till arduino. Det andra syftet är att skicka fläktens styrsignal och avståndsmätarens värden från arduino till matlab, där de sedan ”plottas” i en graf i realtid.

Uppgiften har delats upp mellan de båda laboranterna på följande sätt. Mjukvarumässigt har Viktor arbetet självständigt med PWM och haft huvudansvaret för regulatordelen. Michael har arbetat med sensorn och kommunikationen. Då regulatordelen är en så pass stor del har båda haft att göra med denna. De olika delarna i rapporten skrevs så långt som möjligt av den som arbetat med motsvarande del i mjukvaran. För de delar där det blev svårt att dra tydliga gränser, till exempel regulatordelen, hjälptes laboranterna åt där det behövdes. Slutligen har en sammanfattning tillsammans med ett diskussions avsnitt skrivits tillsammans av båda.

# Teori

Pingismodellen innehåller en sensor som har till uppgift att mäta var bollen befinner sig på armen. Informationen skickas vidare till en regulator, vilken genererar en styrsignal. Signalen skickas som en digital signal i intervallet [0…1023], eftersom en tio bitars D/A-omvandlare används. Fläkten styrs med en spänning mellan 0 - 3,3V, något som ger upplösningen:

Ett stabilt system är ett system som har ett förutsägbart beteende samt behåller detta vid eventuella störningar [8]. Felaktigt val av regulator såväl som felaktig inställning kan leda till att vårt reglersystem blir instabilt. Detta är något som man vill undvika men risken att systemet blir ostabilt finns alltid hos ett återkopplat system. Ett sådant system är nästintill oanvändbart.

Snabbhet är en annan viktig egenskap hos återkopplade systemet. Hur snabbt systemet ställer in sig i förhållande till börvärdet är vad som definierar snabbheten. Man brukar prata om stigtiden vilket är ett mått som berättar hur lång tid systemet går från 10% upp till 90% av börvärdet. Då stigtiden är ett relativt mått, kommer den att variera beroende på i vilken situation samt regulatortyp man använder. Nackdelen är att en ökad snabbhet oftast leder till sämre stabilitet hos regulatorn. Ett annat mått är insvängningstiden vilket talar om hur lång tid ett system tar på sig att ställas in av börvärdet (oftast +-5%).

I ett statiskt tillstånd betraktas endast sluttillståndet. Valet av regulator påverkar således den statiska noggrannheten. Ett bra val av regulator kan leda till att det kvarstående felet nästan helt elimineras medan ett dåligt val av regulatortyp kan leda till ett större kvarstående fel vilket resulterar i ett dåligt reglersystem.

Regulatorer existerar av olika typer. Beroende på användningsområde krävs ett genomtänkt val av regulator för att få ut en så effektiv reglering som möjligt. För stabilisering av första ordningens processer är det oftast mest praktiskt att använda en P-regulator. Detta då regulatorns huvudsakliga uppgift är att minska det kvarstående felet i systemet. System som är mest lämpliga för denna typ av reglering är de som kan tåla ett konstant kvarstående fel.

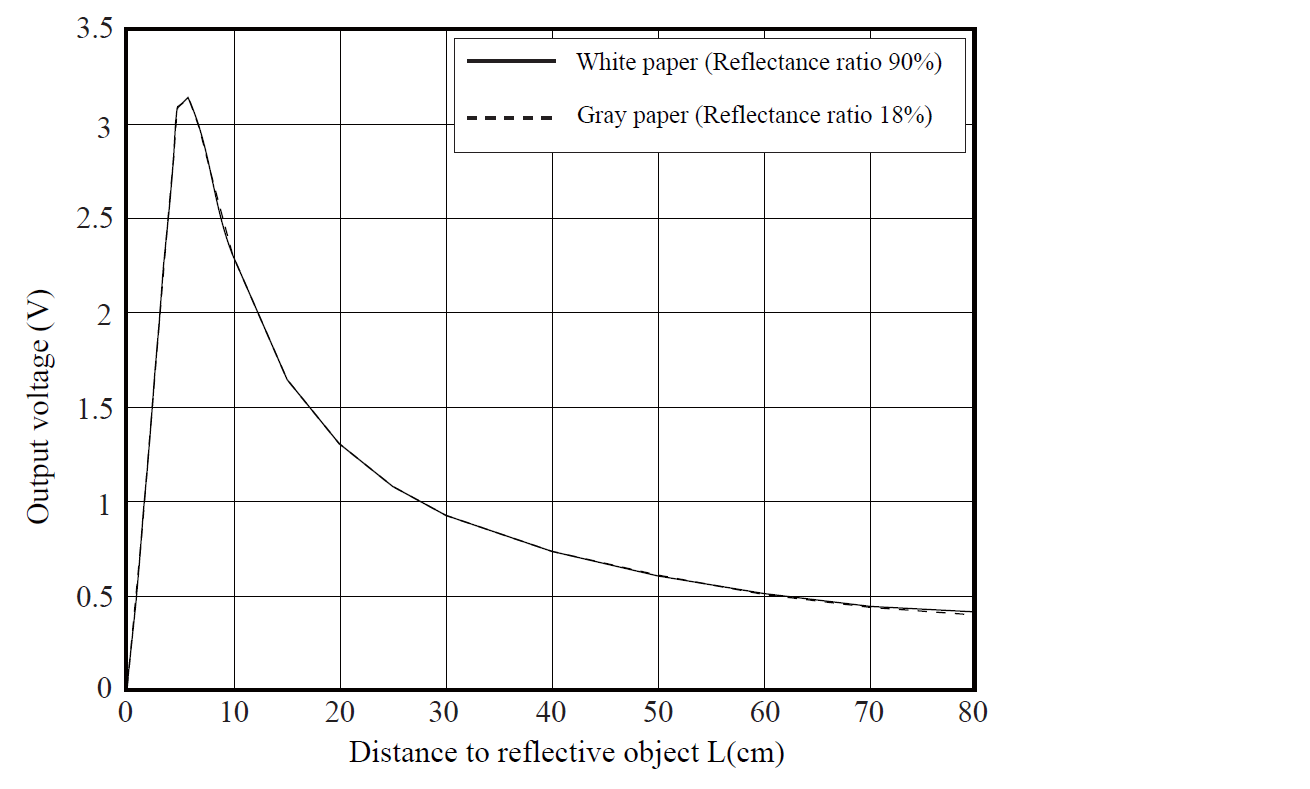
En integrerande del bidrar till att regulatorn omöjligen kan förutse ett eventuellt kvarstående fel och därmed garanteras alltid en överskjutning av sitt börvärde.

En PID-regulator fås genom att tillföra PI-regulatorn ett derivat- eller ”D”-andel. PID-reglering är mycket vanligt förekommande inom industrin. Då felet ökar ger den deriverande delen (D-delen) ett bidrag till styrsignalen med samma tecken som den proportionella delen (P-delen). När felet minskar ger D-delen ett bidrag till styrsignalen med motsatt tecken jämfört med P-delen. Fördelen med detta är att det motverkar översvängningar och tendenser till instabilitet. PID-regulatorn har en snabb svarstid och bidrar till ett mer stabilt system. PID-regulatorn är populär då den även kan användes hos processer av högre ordning.

# Metod

## Sensorn

I projektet används sensorn GP2Y0A21YK0F, tillverkad av SHARP (Givet datablad för sensorn). Figur 1 nedan visar hur förhållandet mellan avstånd till ett objekt och utspänning för sensorn ser ut. Sensorn använder infrarött ljus för att mäta avståndet.I figuren kan man se att det förekommer en topp mellan noll och tio centimeter. Ett objekt som befinner sig på ett avstånd på upp till tio centimeter från sensorn kan inte uppfattas ordentligt. Därför sker ingen reglering inom detta område. Den påbörjas istället vid 20 centimeter och sträcker sig till 40 centimeter. Anledningen till att det inte regleras över 40 centimeter beror på att dessa värden är lika varandra och därmed svåra att särskilja. Detta leder till att det blir svårt att få till en ordentlig reglering.



Figur 1: Graf som visar förhållandet mellan avstånd till ett objekt och utspänning för sensorn.

Sensorn genererar värden för bollens position och skickar dessa till en av de analoga ingångarna på arduinokortet. Metoden för detta återfinns i c-filen FuncADC.c och ser ut på följande sätt:

uint32\_t ReadFanValue(void)

{

uint32\_tcurrSensorValue;

adc\_start(ADC);

currSensorValue = adc\_get\_channel\_value(ADC, ADC\_CHANNEL\_10);

returncurrSensorValue;

}

Vidare hämtar regulatorn värden från sensorn med hjälp av metoden ovan. Hur fläkten regleras blir beroende på hur stor avvikelsen är från börvärdet. Om sensorn ger ut ett värde som är mindre respektive större än börvärdet påverkas regulatorn.

## PWM

Generering av en pwm signal sker på pinne 40 på detArduinoDue kort som användes.I metoden InitPWM görs divers inställningar innan pinne 40 kan initieras. Nedan följer metoden där initieringen av PWM sker.

voidInitPWM(void){

pmc\_enable\_periph\_clk(ID\_PWM);

pwm\_channel\_disable(PWM,PWM\_CHANNEL\_3);

pwm\_clock\_tclock\_setting={

.ul\_clka=1000\*1000,

.ul\_clkb=0,

.ul\_mck=sysclk\_get\_cpu\_hz()

};

pwm\_init(PWM,&clock\_setting);

InitPIN40();

}

Vidare sker en initiering av pinne 40 i InitPIN40:

voidInitPIN40(void)

{

pwmPIN40.channel=PWM\_CHANNEL\_3;

pwmPIN40.ul\_prescaler=PWM\_CMR\_CPRE\_CLKA;

pwmPIN40.ul\_duty=0;

pwmPIN40.ul\_period=100;

pwm\_channel\_init(PWM,&pwmPIN40);

pio\_set\_peripheral(PIOC,PIO\_PERIPH\_B,PIO\_PC8B\_PWML3);

pwm\_channel\_enable(PWM,PWM\_CHANNEL\_3);

}

Signalen som skickas ut kan variera mellan noll och 100, där 100 är maximalt varvtal för fläkten.

## Regleringen

Regleringen utförs av en PID-regulator. Variablerna k\_p, k\_i och k\_d bestämdes genom att vi valde olika värden och sedan tittade hur bra regleringen fungerade i praktiken. Värdena vi kom fram till var bra skickar vi sedan in via matlab, tillsammans med en sampeltid och börvärdet. Koden för själva regulatorn visas nedan.

int32\_t CalcSignal(double sampTime, double k\_p, double k\_i, double k\_d, int32\_t currErr, int32\_t prevErr, int32\_t sumErr)

{

double proportionalPart;

double integralPart;

double derivingPart;

int32\_t signal;

proportionalPart = k\_p \* (double)currErr;

integralPart = (double)sumErr \* (sampTime/k\_i);

derivingPart = ((double)currErr - (double)prevErr) \* (k\_d / sampTime);

signal = proportionalPart + integralPart + derivingPart;

return signal;

}

## Kommunikation mellan Arduino och Matlab

Kommunikation från matlab till arduino upprättas genom att man skapar ett serial object enligt följande rad kod: s = serial('COM10', 'BAUDRATE', 115200). Därefter måste porten öppnas med kodsnutten ”fopen(s)”, innan man kan skicka exempelvis ett börvärde med kodsnutten ”fwrite(s, uint32(30))”.

Kommunikation från arduino till matlab sker genom metoden printf(). Exempel på detta i koden är printf("%d\n", circBuffertActualValue[0]);.

# Sammanfattning och diskussion

Programmet som utvecklats klarar av uppgiften att reglera en pingisboll. Regleringen blir dock aldrig helt exakt, vilket anses bero dels på utrustningens utformning men även att det var svårt att integrera så många aspekter. Programmet klarar även av att plotta en graf i realtid med värden från Arduino. Här finns dock vissa brister, då programmet inte kan hantera decimaltal vid kommunikationen. Gruppen känner att det funnits för lite tid efter sommaren att göra nödvändiga justeringar och felsökningar. I stort anses dock projektet ha varit en framgång, då det varit relativt nytt att hantera ett större programmeringsprojekt.

# 

# Källförteckning

Datasheet distance sensor for GP2Y0A21YK0F, SHARP