Projektrapport - Smörjd blixt

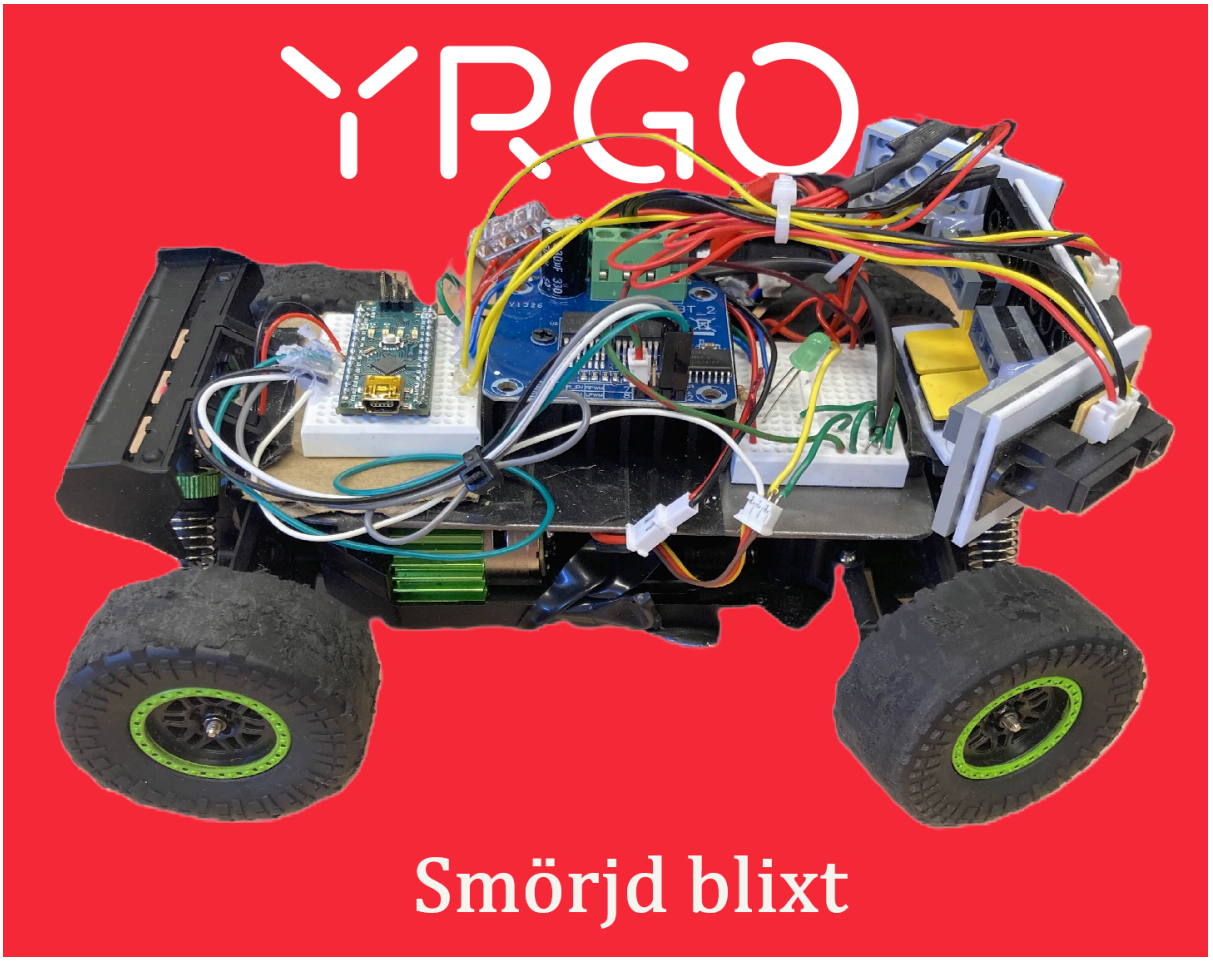
JoBuZz

Rapportförfattare:

- Sebastian Larsson
- Joel Ryberg
- Bobo Bäck Engström

Examinator: Erik Jagre

Datum: 2025-05-26



Sammanfattning

Projektets syfte var att ge studenterna praktisk erfarenhet av grupparbete och agilt arbetsmetod genom att konstruera en självkörande bil. Bilen skulle navigera en förutbestämd bana med hinder och delta i tävlingen Yrgo Grand Prix. Arbetet bedrevs enligt scrum-metodik med sprintar om 2 veckor och definierade roller i gruppen. Under projektet byggdes ett fordon styrt av en mikrodator och infraröda sensorer, och gruppen fördjupade sina kunskaper i avståndsmätning och sensor integration. Projektet gav värdefull erfarenhet i både teknisk utveckling och samarbete.

Innehållsförteckning

Inledning

Självkörande fordon är ett växande område inom teknikvärlden, särskilt för oss som studerar med inriktning autonoma system i en stad med stor koppling till bilindustrin. Som en del av utbildningen genomfördes ett projekt där målet var att i grupper konstruera ett självkörande fordon enligt specifikationerna från Robot-SM: regelverk, under en tidsram uppdelad i sju stycken två-veckors sprintar med en fastställd budget. Projektet avslutades med en tävling, Yrgo Grand Prix, där det färdiga fordonet skulle ta sig runt en bana med hinder i form av tre mindre gupp. Utöver det tekniska målet låg fokus på samarbete och arbete enligt den agila arbetsmetoden scrum. Den här rapporten redogör för projektets genomförande, tekniska lösningar, motgångar och reflektioner kring arbetsprocessen.

Syfte

Syftet med projektet var att ge studenterna erfarenhet av att arbeta i grupp med ett tekniskt utvecklingsprojekt där de skulle använda sig av den agila arbetsmetoden Scrum.
Projektet syftade till att utveckla ett självkörande fordon och samtidigt ge praktisk erfarenhet att planera, samarbeta och problem lösa i ett tekniskt projekt genom att tillämpa den agila arbetsmetoden Scrum.

Metod och Arbetsprocess

Scrum

Med denna projektmetod så jobbar man ut efter olika sprintar som startas med jämna mellanrum. Sprintarna är designade för att man innan sprint-start ska veta vad man ska göra, vilket man gör för att hålla ett bra arbetsflöde.

För att kunna fylla på arbeten och se vad man har för arbeten framför sig så använder man något som kallas för backlog, här fyller oftast projektledaren i olika arbetsuppgifter men även ibland andra medlemmar. De personer som arbetar med sprintarna sätter gemensamt poäng på respektive uppgifter utifrån hur svårt eller tidskrävande uppgiften kommer att bli. Under en sprint vill man inte att de totala poängen är för höga utan hellre lite för låga.
Under sprinten har man även olika möten såsom:

- **Daily stand-up** *(daglig genomgång av vad man gjort, både lyckade lösningar som misslyckade)*
- **Backlog grooming** *(fylla på backlog)*
- **Sprint review** *(redovisar sprintens gång för produktägare)*
- **Sprint planning** *(planera vilka "tasks" som ska utföras under sprinten)*
- **Sprint retrospective** *(direkt efter sprint review för att se vad som kan förbättras till nästa sprint)*

För att ha allt vårt material och våra arbetsuppgifter på samma ställe har vi använt oss av Jira. Jira är en mjukvara skapad av det australienska mjukvaruföretaget Atlassian. Bakgrunden till att vi använder oss av Jira och Scrum är på grund av att Jira är det globalt dominerande verktyget på arbetsplatser som arbetar med scrum.

I Jira fyller man backloggen med issues, problem/arbetsuppgifter, som sedan läggs in i lämplig sprint. Dessa olika issues blir sedan tilldelade med olika story points beroende på hur stor uppgift det är. Sedan genomgås en sprintplanering där man tillsammans sitter och planerar vad som ska göras i sprinten beroende på hur många story points som är lämpliga och vilka issues som är lämpliga beroende på vilken fas projektet är i. I sprinten kan sedan olika issues tilldelas olika personer, men det kan även vara issues där hela gruppen ska vara involverad. Dessa issues hamnar sedan på en scrum board och ligger i en to do-lista. Påbörjas en issue ska den placeras i in progress-kolumnen och om en issue är klar ska den läggas i done-kolumnen. När en issue är avklarad ska även dokumentation göras i detta issue så att andra medlemmar kan se vad som gjorts.

Ekonomi

Projektet hade en fast budget på 7000 kr. Vi använde oss av 6 657 kr, exklusive frakt. Vissa beställningar betalas ur egen ficka för att få påskyndad leverans. Kostnaden för det som bilen består av i slutändan är cirka 2000 kr. De största beställningarna var nummer 1 och 4, vilket omfattade huvuddelen av komponenterna för projektet. Övriga beställningar var mindre och innehöll enstaka komponenter såsom batterier, och sensorer som behövdes för mindre justeringar och kompletteringar under projektet. Med lärarens godkännande lades ett privat inköp på 1900kr för de studerade privata projekt efter att bilen ansågs vara färdig.

Beställningslista 1				
Företag	Beskrivning	Antal	Pris, tot	Kommentar
electrokit	Avståndsgivare 1m 3-zon OPT3101	1	419,20	
electrokit	Nano I/O shield med skruvterminaler	1	23,20	
electrokit	Avståndssensor IR 4-30cm GP2Y0A41SK0F	2	254,40	
electrokit	Arduino Nano (with headers)	1	199,20	
Amazon	ELEGOO 5 HC-SR04 ultraljudsmodul avståndssensor			Skickades inte
electrokit	Avståndsmätare ultraljud HC-SR04 2 - 400cm	3	141,60	
Hobbymagasinet	Service 30 minuter			Skulle vart våran bill, men dom kunde inte leverera

Beställningslista 4				
Företag	Beskrivning	Antal	Pris, tot	
rcsweden	Maverick MV150703 Doha 1/20 4WD Electric Truck - Green	1	695,00	
Adafruit	Adafruit ESP32-S3 Reverse TFT Feather - 4MB Flash, 2MB PSR	3	797,76	
Adafruit	STEMMA QT / Qwiic JST SH 4-pin Cable - 100mm Long	20	202,60	
Adafruit	Adafruit PCA9546 4-Channel STEMMA QT / Qwiic I2C Multiplex	4	168,00	
Adafruit	SparkFun STEMMA QT / Qwiic Breadboard Breakout Adapter	6	124,62	
electrokit	Jordningsplugg ESD 10mm	1	149,00	
Amazon	5 Pcs 0.96 Inch OLED I2C IIC Display Module 12864			Skickades inte
Adafruit	NeoKey 1×4 QT I2C - Four Mechanical Key Switches with NeoPi	1	105,58	
Adafruit	Kailh Mechanical Key Switches - Linear Red - 10 pack - Cherry MX Re	1	73,74	
Adafruit	Light Blue DSA Keycaps for MX Compatible Switches - 10 pack	1	63,14	

Samarbete

Under projektets gång har vi lärt oss att dela upp arbetet, men tack vare ett gott samarbete har vi oftast arbetat tillsammans för att kombinera gruppens kunskaper. Vi har delat upp projektets arbete till respektive sprint för att aldrig ha för lite eller för mycket att göra.
Vi har också haft olika fokusområden när det gäller att fördjupa oss i projektets olika delar.

Joel Ryberg, *Scrummaster,* har främst bidragit med dokumentation i veckodagboken som stöd för rapporten. Han har även ritat flödesscheman och deltagit i övriga moment under projektets gång

Bobo Bäck Engström, *teknikansvarig*, har huvudsakligen ansvarat för mjukvaran och skapat en gemensam kodmall för projektet. Han har också varit delaktig i problemlösning och andra delar av arbetet.

Sebastian Larsson, *ekonomiansvarig*, har tagit fram de flesta hårdvaru lösningarna och varit ansvarig för inköp samt budgetuppföljning. Även han har bidragit i flera andra moment.

Riskanalys

Komponentfel: att sensorer, mikrokontroller eller andra komponenter går sönder.
Åtgärd: Beställa extra komponenter för att ha reservdelar tillgängligt.
Risk: Relativt låg, den större risken är om komponenterna på bilen gått sönder såsom motorn och servot.

Leverans fel: Försenade leveranser eller leveranser som inte kan levereras.
Åtgärd: Beställa i god tid och ha fler alternativ redo ifall fel uppstår. Beställa varorna på egen bekostnad.
Risk: Medelhöga, leveransproblem kan skapa förseningar i arbetet och även att man inte får de delar som är kritiska mot slutet.

Kod Buggar: Koden fungerar inte som förväntat.
Åtgärd: Regelbunden testning, användning av UART för felsökning och GIT för versionshantering.
Risk: Låg, med tanke på de åtgärder som kan göras.

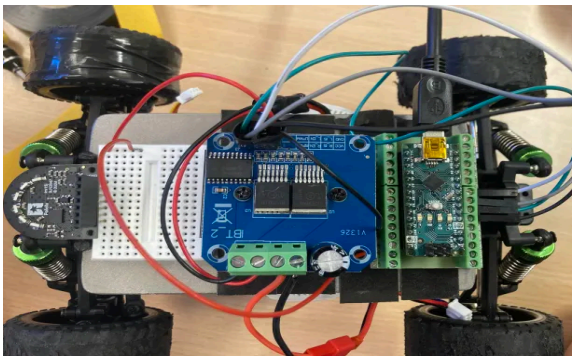
Sensorproblem i tävlings miljö: Att fler bilar på banan skapar oförutsedda problem.
Åtgärd: Testkörning med andra bilar på banan.
Risk: Medelhög, Svårt att få till flertal tester med andra bilar.

Ej klara i tid.
Åtgärd: Jobba i ett jämnt tempo med tydliga delmål för att undvika stress i slutet av projektet.
Risk: Låg, eftersom projektet följer scrum-metoden bör det vara möjligt att behålla god översikt över projektet.

Genomförande

Konstruktion

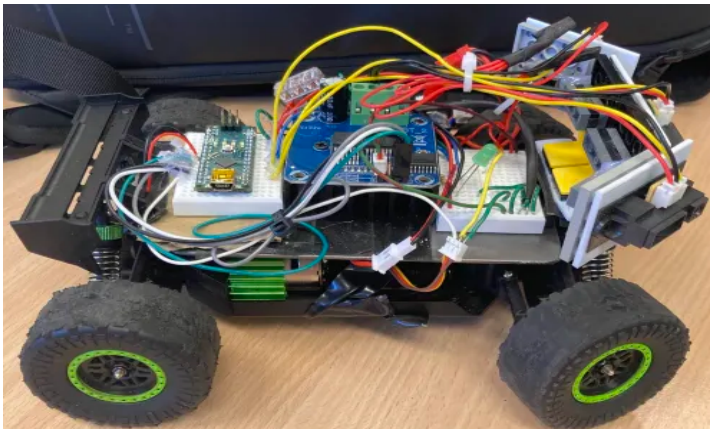
Det första steget i bilens konstruktion var att korta ner den. Bilen var ursprungligen 240 mm lång men maxlängden bilen fick vara för projektet var 200 mm. Genom att ta bort karossen, stötfångarna och reservdäcket på baksidan minskade vi längden till 210 mm. Däcken modifierades sedan med en Dremel där gängorna på däcken slipades bort vilket gjorde att bilen nådde den tillåtna maxlängden.



En platta konstruerades sedan för att ha en solid bas att montera komponenter på plattan, skruvas fast i hål som tidigare användes för att fästa karossen. Arduinon, nano shield, H-bryggan och en breadboard fästes på plattan med hjälp av dubbelhäftande tejp.

Den första sensorn som användes var OPT3101 och den monterades på breadboarden. Men när beslutet gjordes att använda IR sensorer konstruerades ett justerbart stativ för sensorerna med LEGO. Stativet använde sig av LEGO:s friktionsgångjärn för att kunna justera vinklarna på sensorerna under testfasen.

Under den sista sprinten togs beslutet att göra om kabeldragningen för att underlätta felsökning inför tävlingsdagen. Detta gjordes genom att ha färgkodning av kablar, användning av kopplingsklämmor och korta ner överflödigt långa kablar. Detta ledde till problem med glapp i vissa anslutningar och slutade med att kablar löddes istället för att använda kopplingsklämmor och att nano shield byttes bort mot ytterligare ett breadboard.



Material och komponenter

Maverick MV150703 Doha

Efter att vår första tanke, som var att specialbeställa en bil utefter våra kriterier, inte gick i lås så var vi tvungna att hitta en ny bil. Att hitta en produktions RC som mötte alla våra kriterier, såsom mått, borstad motor och ett PWM-styrt servo, skulle visa sig vara svårt då vi inte heller ville ha en crawler eftersom de omvandlar energi till kraft medans vi ville omvandla energi till fart. Efter en tids research hittade vi Maverick MV150703 Doha [1]. Denna bil mötte alla våra kriterier förutom ett. Den var 240 mm lång medan maxlängden för tävlande bilar endast var 200 mm. Eftersom den mötte alla kriterier förutom längden och endast kostade 695 kr bestämde vi oss för att beställa den ändå med vetskapen att vi skulle behöva korta ner den.

När bilen sedan blev levererad märkte vi fort att detta var en RC-bil av hög kvalitet. Den bestod av ett underrede som var utrustat med en metallplatta samtidigt som den var låg och hade bra hjulbredd, vilket gjorde bilen stabil. Bilen var även lättarbetad då kablar till batteri, motor och servo var lättillgängliga. Det saknades information om servot på RC Sweden´s hemsida, men efter ett telefonsamtal med deras kundservice kunde vi enkelt få den information vi letade efter.

Sensorer

Under projektet testades flera sensorer för att hitta den optimala för våran bil. OPT3101 [2] är en infraröd sensor med tre stycken sensorer monterade på ett kort för att täcka en zon på 160 grader. Denna fungerade relativt bra men var märkbart sämre på att köra vänstervarv, då högersidan på sensorn inte visade lika stabila värden som vänster. Därför togs valet att inte använda OPT3101 i slutändan.

Ultraljudssensorn HC-SR04-2[3] testades under en kort period enbart på breadboard. Det var strul att få ut användbara värden och därför avslutades testerna med denna sensorn.

Den infraröda sensorn IR 4-30cm[4] gav oss bäst resultat under testningen, dock betedde den sig sämre i vissa lägen i kurvorna till slut valde vi att prova IR 10-80cm[5] som hade längre räckvidd och det gav oss bättre resultat i kurvorna, troligen för att sensorerna ny lyckades mäta till båda sargkanterna oavsett läget bilen kom in i kurvan. Vi har tre av dessa sensorer monterade på vår bil, den i mitten styr enbart vår backfunktion. Våra IR- sensorer använder sig av analog kommunikation till våran mikrokontroller och på så sätt får vi värden från 0-1023 genom nanos 10-bitars analog omvandlare. Det var enkelt att använda råvärden och vi behövde inte formatera om till några metriska värden.

Arduino Nano

Vi valde en Arduino Nano[6] på grund av att den är billig och kompakt men besitter ändå alla funktioner som finns på en Arduino Uno. Vi valde här att sätta nanon på en shield för att lätt kunna koppla in och ur kablar. Nanon använder sig av en Atmega 328p processor.

H-brygga BTS7960

Vi valde denna BTS7960[7] H-brygga för att vi ville ha en kraftfull och hållbar konstruktion som är lätthanterlig. Då vi hade hört talas om tidigare års grupper som haft lite billigare H-bryggor som hade sämre verkningsgrad valde vi att lägga ut mer pengar på en mer robust.

H-bryggan fungerar med två stycken PWM-pins, RPWM och LPWM. När RPWM är hög och LPWM är låg körs motorn framåt och tvärtom, när RPWM är låg och LPWM är hög körs motorn bakåt. Dessa kopplades på digitala outputs på vår Arduino Nano. Två andra inputs på H-bryggan som även de kopplades till digitala outputs på vår Arduino är L_EN och R_EN som är två enable pins som måste vara höga för att motor och servo ska kunna styras genom H-bryggan. Den även inputsen VCC och GND för bilens batteri.

Outputs från H-bryggan som ger ström till motorn är M+ och M- och B+ och B- för servot. Dessa outputs strömförsörjer endast motorn och servot medan de även styrs via PWM-signaler från arduinon.

Programmering

Språk & Program

För programmeringen i detta projekt så har vi att jobba med C++ i VSCode. För att vi enkelt skulle kunna jobba gemensamt i koden så har vi använt oss av Git och Github[8]

Kompilator & Hårdvara

Vi använder en Arduino Nano med en Atmega328 som vi kompilerar med hjälp av ett verktyg som heter PlatformIO. Koden har vi tagit fram stegvis genom att få en fungerande kod för de olika komponenterna vi använder och sendan har vi successivt fört ihop koder för att till exempel få servot att fungera med en sensor osv.

Felsökning

För att felsöka möjliga problem i koden så har vi använt oss av UART-kommunikation som vi skrivit ut på serial monitor. Vi har printat de flesta variabelvärdena för att kunna se sensorernas värden, detta har gjort att vi lätt kunnat lösa konstiga problem eller filtrera bort dåliga värden.

Funktioner

För att kunna ha dynamisk styrning så har vi använt oss av en inbyggd funktion i C++ *(rad 101-104)*. Koden styr hur mycket bilen ska svänga, baserat på skillnaden mellan vänster och höger IR-sensor. Ju större skillnad, desto mer svänger bilen, detta skapar en dynamisk, gradvis styrning. Graderna som vi ska svängas sparas i variabeln steerAmount som används i koden i raderna (*106-111*)

För att få bilen att kunna ta sig ur alla situationer så använde vi oss av en backfunktion som kallades på när vi IR-Mitten inte hade ändrat på sig på en sekund*(rad 35-48)*. Bilen backade ut lite och svängde åt ena hållet sen fortsätta att köra, om backfunktionen körde åt fel håll så fick man hjälp gratis från funktionärer. Utan denna funktion hade vi behövt be om hjälp men på grund av den så behövde vi aldrig be om hjälp.

Diskussion

Problem

När projektet drog igång strax innan jul och det var dags att bestämma oss för vilken bil vi skulle beställa i beställning #1, bestämde vi oss för att inte beställa en komplett bil. Istället vände vi oss till Hobbymagasinet och förklarade för dom om vårt projekt och våra kriterier, som t.ex att vi ville ha en borstlös motor som styrs med PWM och att måtten inte fick överstiga 200 mm i längd, 150 mm i bredd och 150 mm i höjd.

Kommunikationen mellan oss och Hobbymagasinet var rak och tydlig och fungerade till en början bra över både mail och telefon. Men efter ett tag slutade dom att svara och första beställningen kom och vi hade inte fått något svar. Vi kontaktade dem sedan igen inför den andra beställningen och förklarade för dem att vi behövde ha vår bil så snart som möjligt för att kunna arbeta med den så att vi inte skulle hamna efter i planeringen. Trots att vi flertalet gånger hade kontakt med dem via mail och telefon så fick vi aldrig vår bil. Under denna tiden beställde vi inte någon annan bil då vi hade fått till oss av Hobbymagasinet att de skulle leverera och att det inte var några problem, vilket ledde till att vi under denna period endast kunde sitta och labba med sensorer och h-bryggor samt fixa administrativa grejer som Github och rapportmall medans de andra grupperna arbetade på sina bilar och vissa redan hade fått bilarna att köra flera varv runt banan.

Inför den fjärde beställningen fick vi ett mail från Hobbymagasinet där de skrev att de inte kunde leverera till oss på grund av att vissa komponenter var slut i lager hos dom och att köpes skulle makuleras. Detta skapade frustration från vårt håll i gruppen och vi blev besvikna på Hobbymagasinet då vi kände att detta hade varit till en stor nackdel för oss eftersom vi hade sett många av de andra grupperna köra med sina bilar. Vid denna tidpunkt var vi tvungna att hitta en ny bil att beställa, vilket vi gjorde ganska snabbt. Denna motgång stärkte oss som grupp då vi hanterade det bra och löste snabbt en ny bil som fungerade riktigt bra. Man kan även dra paralleller från detta problem till det riktiga arbetslivet där vi fått höra att det faktiskt brukar vara ett vanligt problem att man inte alltid får det man behöver eller att en underleverantör inte kan leverera i tid. Därför blev detta trots allt lärorikt för oss.

Under projektets gång så har vi stött på ett flertal problem, ett av de första problemen vi hade var när vi använde oss av OPT 3101.[2] Denna sensor använder sig av UART och då krockade kompileringen av Nanon med sensorn som gjorde att vi fick väldigt konstiga fenomen, detta insåg vi till slut och lösningen var att koppla ur sensorn medan vi kompilerade koden. Ett annat problem som medföljde med sensorn var att vi kunde köra mycket bättre åt ett håll än de andra.

När vi bytte till IR 4-30cm [4] och skrev en ny kod för Analog kommunikation så hade vi betydligt mycket mindre problem men vi märkte ett litet problem som skulle märkas vara större än vi trodde. På grund av att sensorn bara kunde ungefär 30 cm så blev värdena konstiga emellanåt och bilen kunde köra rakt in i en vägg. Felsökningen här var svår att hitta då vi trodde de var ett mjukvaruproblem och inte ett hårdvaruproblem. Till slut beställde vi samma sensor fast med 80 cm räckvidd.

Ett stort problem vi upptäckte relativt sent var att vårt servo inte kunde svänga hela vägen åt ett av hållen fast vi hade ställt in max-steer på 30 grader. Vi trodde att detta problem var omöjligt att fixa så för att lösa det så använde vi en backfunktion som funkade för det “dåliga hållet”. I ett av våra race så krockade vi nästan två gånger varje varv in i väggen på grund av den dåliga svängradien, men på

grund av vår backfunktion så lyckades vi ändå vinna racet. Lösningen för detta problem kom till oss 5 minuter innan final racet. Eftersom bilen svängde fullt åt höger vid 30 graders max-steer men inte fullt åt vänster, beslöt vi oss för att uppdatera koden och ändra variabeln till 35 grader istället. Detta var en chansning då vi inte visste om servot skulle hålla men det visade sig istället vara lösningen på vårt problem, eftersom bilen nu även kunde svänga fullt åt vänster och under finalen inte fastnade mot väggen på det föregående racet.

Lärdomar

Under projektets gång har motgångarna och problemen som dykt upp lett till olika lärdomar. Problemet med Hobbymagasinet har gett oss insikten i att man alltid bör ha en reservplan ifall leverantörer inte kan leverera som utlovat. När vårt batteri gick sönder i slutet mot projektet och det blev leveransförseningar på ett nytt, fick vi anpassa oss och konstruera ett eget batteri.

Vi har även utvecklat vår förmåga i felsökning både i mjukvara och hårdvara. Speciellt mot slutet när vi fick stora problem på grund av glapp som tog tid att identifiera var felet låg. Och fått en utökad förståelse för att jobba med git.

Utöver de tekniska lärdomarna har det som möjligen har störst betydelse i arbetslivet varit att lära sig jobba i team och i en förenklad variant av scrum. Detta har hjälpt oss hålla en tydlig riktning och kunnat hantera oväntade hinder mer effektivt och med mindre stress.

Resultat

Under testfasen lyckades bilen köra fem varv runt banan utan hinder på 53 sekunder. Det visade att både sensorerna och styralgoritmer fungerade stabilt under hög hastighet. På tävlingsdagen presterade bilen bra nog för att vinna YRGO Grand Prix. Vi behövde aldrig be om hjälp om bilen fastnade, tack vare vår backfunktion som gjorde att bilen lyckades ta sig ut från dåliga situationer varje gång. Vår bil genomförde över tio varv varje race med ett varvrekord på 15. Överlag har gruppen lyckats leverera en bil av hög kvalitet med god precision, stabilitet, hög fart och robust konstruktion.

Referenser

[1] DOHA, Maverick quick start guide, parts list. [online]. tillgänglig:
https://www.hpiracing.com/assets/documents/maverick/instructions/maverick_doha_instruction_manual_25072024_web.pdf
https://www.hpiracing.com/assets/documents/maverick/exploded_views/maverick_doha_instruction_manual_25072024_e.pdf
[åtkomstdatum: 26-maj-2025]

[2] Texas Instruments, OPT3101 datasheet. [online]. tillgänglig:
https://www.electrokit.com/upload/product/41017/41017838/opt3101.pdf
[åtkomstdatum: 26-maj-2025]

[3] Elecfreaks, HC-SR04 2 datasheet. [online] tilläggning:
https://www.electrokit.com/upload/product/41013/41013207/HC-SR04.pdf
[åtkomstdatum: 26-maj-2025]

[4] Sharp, IR 4-30cm data sheet. [online] tillgänglig:
https://www.electrokit.com/upload/product/41013/41013009/GP2Y0A41SK0F_ED05G101A.pdf
[åtkomstdatum: 26-maj-2025]

[5] Sharp, IR 10-80cm data sheet. [online] tillgänglig:
https://www.electrokit.com/upload/product/41004/41004830/GP2Y0A21YK.pdf
[åtkomstdatum: 26-maj-2025]

[6] Arduino, Arduino Nano datasheet. [online] tillgänglig:
https://www.electrokit.com/upload/quick/a0/8b/d467_41010033-tds.pdf
[åtkomstdatum: 26-maj-2025]

[7] Handson Technology, BTS7960 motor driver module data sheet. [online] tillgänglig:
https://www.handsontec.com/dataspecs/module/BTS7960%20Motor%20Driver.pdf
[åtkomstdatum: 26-maj-2025]

[8] Github, Winning_car_2025… [online] tillgänglig:
https://github.com/Dexter9532/Winning_car_2025_yrgo_grand_prix/blob/main/src/main.cpp
[åtkomstdatum: 26-maj-2025]

Appendix

https://github.com/Dexter9532/Winning_car_2025_yrgo_grand_prix/blob/main/src/main.cpp

```cpp
#include <Servo.h>

#include <Arduino.h>



// === PIN-KONFIGURATION ===

#define L_EN 8

#define R_EN 7

#define L_PWM 5

#define R_PWM 6

#define SERVO_PIN 9



// === SENSOR-PINS ===

#define IR_LEFT A2

#define IR_MIDDLE A3

#define IR_RIGHT A0



// === OBJEKT ===

Servo steeringServo;



// === KONSTANTER ===

const int maxSpeed = 50;

const int servoCenter = 85;
```

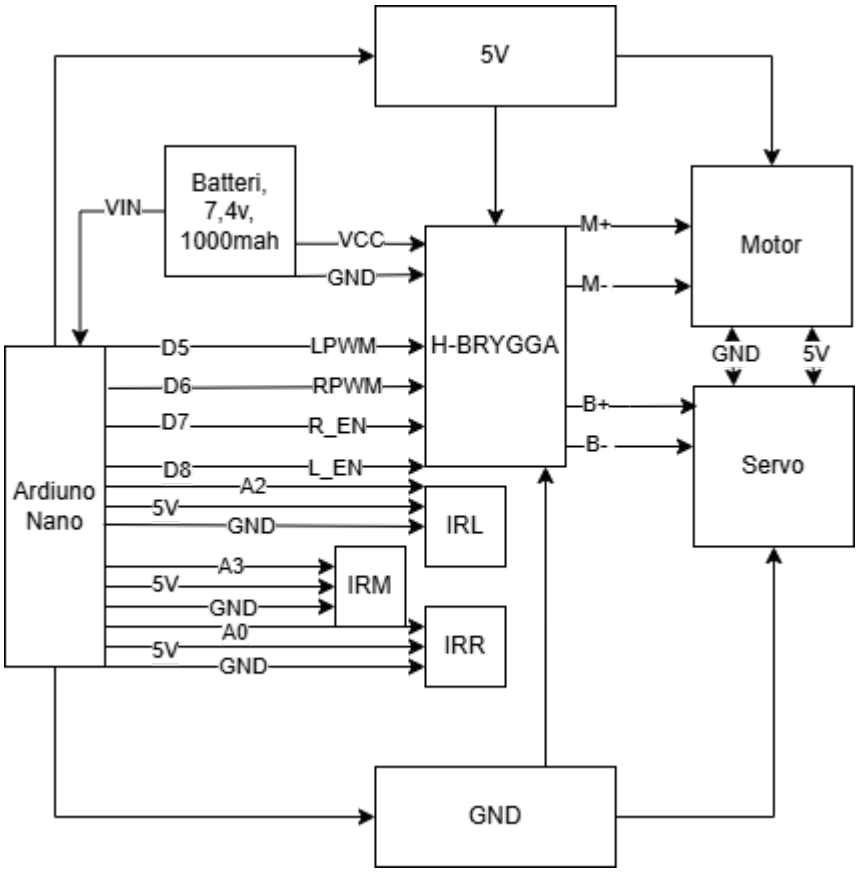
```
const int stuckThreshold = 20;
const unsigned long stuckTime = 1000;

const int detectThreshold = 100;

const int maxSteer = 35;
const int minSteer = 5;

// === STUCK-DETEKTION ===
int lastIR = 0;
unsigned long lastMoveTime = 0;

void recoverFromStuck() {
    Serial.println("⚠️ Fastnat! Backar...");

    analogWrite(L_PWM, 30);
    analogWrite(R_PWM, 0);
    steeringServo.write(servoCenter + 25);
    delay(700);

    analogWrite(L_PWM, 0);
    analogWrite(R_PWM, 0);
    delay(200);
```

```
    steeringServo.write(servoCenter);
    delay(200);
}

void setup() {
    Serial.begin(9600);
    analogReference(DEFAULT);

    steeringServo.attach(SERVO_PIN);
    steeringServo.write(servoCenter);

    pinMode(A4, INPUT); // Killpin
    pinMode(A5, INPUT); // Killpin
    pinMode(4, OUTPUT); // LED

    pinMode(L_EN, OUTPUT);
    pinMode(R_EN, OUTPUT);
    pinMode(L_PWM, OUTPUT);
    pinMode(R_PWM, OUTPUT);
    digitalWrite(L_EN, HIGH);
    digitalWrite(R_EN, HIGH);

    pinMode(IR_LEFT, INPUT);
    pinMode(IR_MIDDLE, INPUT);
    pinMode(IR_RIGHT, INPUT);
```

```
    lastIR = analogRead(IR_MIDDLE);
    lastMoveTime = millis();
}

void loop() {
    if (digitalRead(A4) == HIGH && digitalRead(A5) == HIGH) {
        digitalWrite(4, HIGH); // LED på

        int irLeft = analogRead(IR_LEFT);
        int irMiddle = analogRead(IR_MIDDLE);
        int irRight = analogRead(IR_RIGHT);

        Serial.print("Vänster: "); Serial.print(irLeft);
        Serial.print(" | Mitten: "); Serial.print(irMiddle);
        Serial.print(" | Höger: "); Serial.println(irRight);

        // === STUCK-DETEKTION ===
        if (abs(irMiddle - lastIR) > stuckThreshold) {
            lastMoveTime = millis();
            lastIR = irMiddle;
        }

        if (millis() - lastMoveTime > stuckTime && irMiddle >
detectThreshold) {
```

```
            recoverFromStuck();
            lastMoveTime = millis();
            return;
        }

        // === DYNAMISK STYRNING ===
        int diff = abs(irLeft - irRight);
        int steerAmount = map(diff, 0, 300, minSteer, maxSteer);
        steerAmount = constrain(steerAmount, minSteer, maxSteer);

        int angle = servoCenter;
        if (irLeft > irRight) {
            angle = servoCenter + steerAmount; // hinder vänster → sväng höger
        } else if (irRight > irLeft) {
            angle = servoCenter - steerAmount; // hinder höger → sväng vänster
        }

        steeringServo.write(constrain(angle, 50, 120));

        // === MOTOR KÖR FRAMÅT ===
        analogWrite(R_PWM, maxSpeed);
        analogWrite(L_PWM, 0);
    } else {
        digitalWrite(4, LOW); // LED av
```

```
        analogWrite(L_PWM, 0);

        analogWrite(R_PWM, 0);

        steeringServo.write(servoCenter);

    }



    delay(50);

}
```



Vinnande hårdvarulösning

```
Psuedo koden
Funktion StyrAlgoritm(vänsterAvstånd, högerAvstånd)
    Tröskel = 10  // Avstånd i cm som definierar när bilen ska reagera

    Om vänsterAvstånd < Tröskel och högerAvstånd < Tröskel
        Returnera "Stanna"  // För nära båda sidor, stanna för säkerhet

    Om vänsterAvstånd < Tröskel
        Returnera "Sväng höger"  // För nära vänster, sväng höger

    Om högerAvstånd < Tröskel
        Returnera "Sväng vänster"  // För nära höger, sväng vänster

    Om vänsterAvstånd > högerAvstånd
        Returnera "Sväng vänster"  // Mer utrymme till vänster, sväng åt det hållet

    Om högerAvstånd > vänsterAvstånd
        Returnera "Sväng höger"  // Mer utrymme till höger, sväng åt det hållet

    Annars
        Returnera "Kör rakt fram"  // Jämnt avstånd, kör framåt
Slutfunktion
```

Psudeo kod

Beställningslista					
Företag	Beskrivning	Artikelnummer	Antal	Pris/st	Pris, tot
electrokit	Avståndsgivare 1m 3-zon OPT3101	41017838 - Pololu	1	419,2	419,20
electrokit	Nano I/O shield med skruvterminaler	41016067	1	23,2	23,20
electrokit	Avståndssensor IR 4-30cm GP2Y0A41SK0F	41013009 - Sharp	2	127,2	254,40
electrokit	Arduino Nano (with headers)	41010033 - Arduino	1	199,2	199,20
Amazon	Motordrivmodul, kretskortmodul, dubbel BTS7960-högström-H-backventil	N/A	1	168,76	168,76
electrokit	Avståndsmätare ultraljud HC-SR04 2 - 400cm	41013207	3	47,2	141,60
rcsweden	Maverick MV150703 Doha 1/20 4WD Electric Truck - Green	150703	1	695,00	695,00
Adafruit	Adafruit ESP32-S3 Reverse TFT Feather - 4MB Flash, 2MB PSRAM, STEMMA	5691	3	265,92	797,76
Adafruit	STEMMA QT / Qwiic JST SH 4-pin Cable - 100mm Long	4210	20	10,13	202,60
Adafruit	Adafruit PCA9546 4-Channel STEMMA QT / Qwiic I2C Multiplexer - TCA954	5664	4	42,00	168,00
Adafruit	SparkFun STEMMA QT / Qwiic Breadboard Breakout Adapter	4527	6	20,77	124,62
electrokit	Jordningsplugg ESD 10mm	41017013	1	149,00	149,00
Adafruit	NeoKey 1x4 QT I2C - Four Mechanical Key Switches with NeoPixels - STEMI	4980	1	105,58	105,58
Adafruit	Kailh Mechanical Key Switches - Linear Red - 10 pack - Cherry MX Red Com	4952	1	73,74	73,74
Adafruit	Light Blue DSA Keycaps for MX Compatible Switches - 10 pack	5006	1	63,14	63,14
electrokit	Avståndssensor IR 4-30cm GP2Y0A41SK0F	41013009 - Sharp	3	127,20	381,60
electrokit	Minneskort SDHC 32GB med Raspberry Pi OS	41021732	3	129,00	387,00
electrokit	Raspberry Pi 4 inbyggnadslåda röd/vit	41017111	1	99,00	99,00
electrokit	Strömförsörjning 15W USB-C Raspberry Pi 4 vit	41017113	2	149,00	298,00
electrokit	Raspberry Pi 4 Model B 1GB	41017108	1	499,00	499,00
electrokit	ESP32-S3 utcecklingskort med 1.28" rund LCD - touch	41022529	2	329,00	658,00
electrokit	Raspberry Pi 4 inbyggnadslåda mörkgrå	41017369	1	99,00	99,00
electrokit	Fläkt och kylfläns för Raspberry Pi 4	41018040	1	75,00	75,00
electrokit	Raspberry Pi Zero 2 W	41018730	1	219,00	219,00
electrokit	Power Supply 12.5W Micro-USB white Raspberry Pi	41020554	1	149,00	149,00
electrokit	Case for Raspberry Pi Zero and Zero 2 W	41015525	1	69,00	69,00
Amazon	ELEGOO 5 HC-SR04 ultraljudsmodul avståndssensor för Arduino UNO Nano	saknas			0,00
Hobbymagasinet	Service 30 minuter	Referens 99.0300			0,00
Amazon	5 Pcs 0.96 Inch OLED I2C IIC Display Module 12864	N/A	0	159,41	0,00
Amazon	Yangers 2-pack 7,4 V 2 000 mAh Li-ion-batteri RC uppladdningsbara batteri	N/A			0,00
radiostyrda-modeller	Batteri 7.4V 1300mAh Li-Ion - Atom - Maverick	150544		229,00	0,00

Beställningslista, grönt använt i slutändan, rött levererades inte.

Github Repo

https://github.com/Dexter9532/Winning_car_2025_yrgo_grand_prix