KTH

Följa en linje

Lego mindstorm

Hampus Hammersberg
2014-08-28
hamham@kth.se
II1310 Introduktionskurs i datateknik

Sammanfattning

Bakgrunden till den här laborationen är att man ska få se hur det fungerar att jobba med programmering nära hårdvaran och att introducera ett sätt att öva på att felsöka kod.

Målet med uppgiften är att få roboten att röra sig efter den svarta linjen.

Medan syftet är att ge studenten en introduktion hur det är att plugga som ICT student och vilken arbetsmetodik som används.

För att genomföra laborationen behövs ett program och en drivrutin för att datorn ska kunna kommunicera med roboten. Dessa behövs laddas ned innan man börjar med uppgiften. Resten av genomförandet går ut på att felsöka ett färdigt program och göra korrigeringar av det så att roboten följer linjen.

Resultatet analyseras och det undersöks varför det gick som det gjorde och hur resultatet kunde förändrats om en annan metod använts under genomförandet.

Efter detta diskuteras arbetsmetoden som användes för att lösa problemet. Är det bäst att testa sig fram eller är det bra att läsa vad syntaxen betyder innan man börjar testa och korrigera?

Innehållsförteckning

1. Inledning

Den här laborationen går ut på att man ska felsöka en annan persons kod till en legorobot som ska följa en svart linje på marken. Det här gör att studenten får en känsla för programmering och för hur en riktig ingenjör arbetar och hur man jobbar i grupp och kan behöva hjälpa en annan person med att felsöka deras kod.

1.1 Bakgrund

Laborationen utförs på ett sätt som liknar det som används när man ska programmera mot annan mer relevant hårdvara som en ingenjör skulle kunna använda. Labben innehåller även felsökning av en annan persons kod. Det här kan vara användbart då man ofta jobbar i par eller i grupp på ett företag och då kan hjälpa andra personer i gruppen. Det är även introduktion till det arbetsätt och den noggrannhet som behövs när man programmerar.

1.2 Syfte och målsättning

Målet är att man ska få roboten att åka längs med en linje och korrigera sin färdriktning genom att köra långsammare eller snabbare på de olika hjulen. Eftersom koden är färdigskriven är syftet att göra små korrigeringar i koden så att roboten gör det den ska.

Syftet är att på ett intressant sätt ge en introduktion till programmering genom att arbeta i par. Men även att få studenten att tänka enligt det arbetsätt och med den precisionen som en ICT student behöver.

2. Genomförande

Inledningsvi gick vi igenom Lab-PM för att få en bild av vad uppgiften var och vilken programvara som behövdes för att genomföra laborationen. Den utvecklingsmiljö och drivrutin som krävdes laddades ner. Sedan läste vi in koden i programmet. Det första steget av felsökningen är att kompilera koden för att se om det är några fel på syntaxen. Alltså om det är något som programmet inte förstår hur det ska tolka. När man inte längre har några syntaxfel kan man läsa in programmet i roboten för att se vad som händer när roboten kör koden och skapa sig en uppfattning för vad som är fel med hur roboten beter sig. Efter detta försökte vi förstå oss på vad de olika delarna av programmet gjord och vilka komandon för att köra motorerna osv. När vi hade identifierat de olika felen ändrade vi så att roboten sa till rätt utgångar att köra och tolka informationen från sensorerna på rätt sätt. Slutligen såg vi till att roboten skrev ut den texten som den skulle skriva ut och att texten såg ut som vi ville att den skulle se ut.

3. Resultat

Radnummer	Ny kod	Kommentar
34	String	Vi ändrade från int till string
92	OUT_C	Vi ändrade utgång som kördes från A till C
94	OUT_C	Vi ändrade utgång som kördes från A till C
76	IN_3	Vi ändrade ingången den hämtade informationen om ljust och mörkt till den optikska censorn istället för den som satt uppe på.
94	SpeedFast	Vi ändrade så att när det blev för ljust så körde man fortare på det ena hjulet än det ändra så att den korrigerar sin riktning lite.
98	SpeedSlow	Vi ändrade så att när det blev för ljust så körde den snabbare på det andra hjulet och korrigerade sin riktning lite.
86	OUT_BC	Vi ändrade så att den stannade på båda hjulen. Innan stod det OUT_AB
2	#define SpeedSlow 30	Vi ändrade den långsamma av farterna till 30 för att den optiska censorn skulle hinna med och mäta om det var ljust eller mörkt.
3	#define SpeedSlow 50	Vi ändrade den snabba hastigheten till 50 så att den optiska censorn skulle hinna med att mäta om det var ljust eller mörkt.
46	TextOut(0, (LCD_LINE2 - (8*i)), names[i]);	Vi tog bort -16. Eftersom det gjorde så att den skrev ut våra namn på samma rad som den skrev ut gruppmedlemmar. Det gjorde att bara hälften av det som skulle skrivas ut skrevs ut.
35	"Hampus",	Vi ändrade så den skulle skriva ut mitt namn.
36	"Oskar"	Vi ändrade så den skulle skriva ut min kompis namn.

4. Analys

Resultatet var att vi hade ändrat mindre delar av koden och lyckades få roboten att följa linjen. Det mestadels bra med genomförandet kunde dock vissa saker genomförts bättre om vi gjorde det på ett annat sätt. Om vi hade kopplat om sensorerna istället för att skriva om koden hade det gått lite fortare och vi hade inte behövt ändra på så många olika ställen. Det hade förmodligen gått fortare om vi hade ett mer strukturerat angripsätt när vi började med felsökningen av koden. Om vi hade tänkt efter mer noggrant vad som hände på de olika raderna hade det förmodligen gått fortare att hitta alla problemen med koden. Eller om vi använde oss mer av API:n och sökt upp vad de olika komandona gjorde hade det varit mer effektivt.

Om man analyserar koden märker man vilka delar av koden som spelade roll för hur roboten betedde sig. Main Tasken börjar med att man sätter två funktioner som ska köras helatiden medan programmet kör. Den första av dessa två är en task som kollar om den sensorn som sitter längs fram blir påverkad av att roboten kör in i något. Den här tasken består av en while loop som kör hela tiden. I den finns det en if-sats som kollar om sensor 1 eller 4 blir påverkad. Om någon av de här

sensorerna blir påverkade kommer en lite melodi kommer spelas upp, gruppensmedlemmar kommer skrivas upp på skärmen, den variabeln finished sätts till true och efter 20 sekunder avslutas programmet. Den andra av de två tasks som körs hela tiden under programmets körning börjar med en loop som körs helatiden. Sedan kommer det en if-sats som kollar om variabeln finished är true. Om den är det kommer båda hjulen att stannas och man kommer hoppa ur loopen genom att använda break. Om finished inte är true kommer programmet gå vidare. Sedan anropas funktionen readLightSensor(). Den här funktionen kollar vad den optiska sensorn läser av och sparar det i variabeln lightIntensity. Eftersom variabeln är global och gäller i hela programmet behöver den inte returneras. Sedan kommer det en variabel en if-sats som kollar om variabel är minder än makrot TopThreshold. Om den är det kommer ett av hjulen att köras i normalfart. Om variabeln är större än TopThreshold kommer hjulet att köras med en snabbare hastighet så att roboten svänger. Efter detta går programmet in i en ny if-sats som kollar om variabeln är mindre än större än makrot BotThreshold om det är det kommer det andra hjulet att köra med samma fart som innan. Om variabeln är mindre än BotThreshold kommer det andra hjulet att köras fortare så att roboten svänger åt andra hållet.

5. Diskussion

Labbens mål var att vi skulle få en mjuk start till programmering nära hårdvara där det är ganska enkelt. Det var även att vi skulle få roboten att följa en linje medhjälp av en ljuskänslig sensor. Dessa mål är ganska lätta att uppnå med tanke på att koden redan var klar och det bara var en enkel tankeövning att hitta felen och korrigera dem. Dock gick inte allt felfritt. Det största problemet vi stötte på var det som var angående displayen eftersom det dels krävdes en förståelse av vad funktionen som matade ut text på displayen gjorde och dels få en förståelse för vad det var som gjorde att det skrev ut de olika strängarna på varandra. Det gjorde att jag lärde mig lite mer om hur det går till under en labb när man pluggar på högskolan och att men inte får allting serverat för en utan att man måste söka upp informationen själv. Vilket kommer vara viktigt när man pluggar på en högskola. Programvaran NXC är bra ur de synpunkter som vi använde den. För att lösa problemet testade vi oss fram genom att ändra på kod för att se vad som hände. Detta var kanske inte det mest effektiva tillvägagångssätten. Det hade förmodligen gått fortare att leta reda på vad al syntax betydde och sedan ändra på delar som behövdes istället för att ändra på kod där vi trodde att felet kunde vara.

Referenser

https://bilda.kth.se/courseId/11430/content.do?id=22224147
http://bricxcc.sourceforge.net/nbc/nxcdoc/nxcapi/

Bilagor

Egen anteckning
28 augusti 10:13

Korrigera inlägg Mer

Hampus Hammersberg
Labben i introdata var intressant. Det kändes dock som att vi inte förstod så mycket utan mest provade oss fram istället för att sätta oss ner och tänka över vad som hände i programmet. Sedan var koden i programmet skriven på ett sätt som jag är ovan vid. Måsvingarna satt på konstiga ställen och liknande. Jag gillar i övrigt inte hårdvarunära programmering utan objektorientering. Det är för mycket olika kommandon och olika in/utgångar att hålla reda på och att man ska hålla reda på var texten visas på displayen. Men uppgiften var bra och man fick var tvungen att tänka lite över hur man skulle ändra koden.