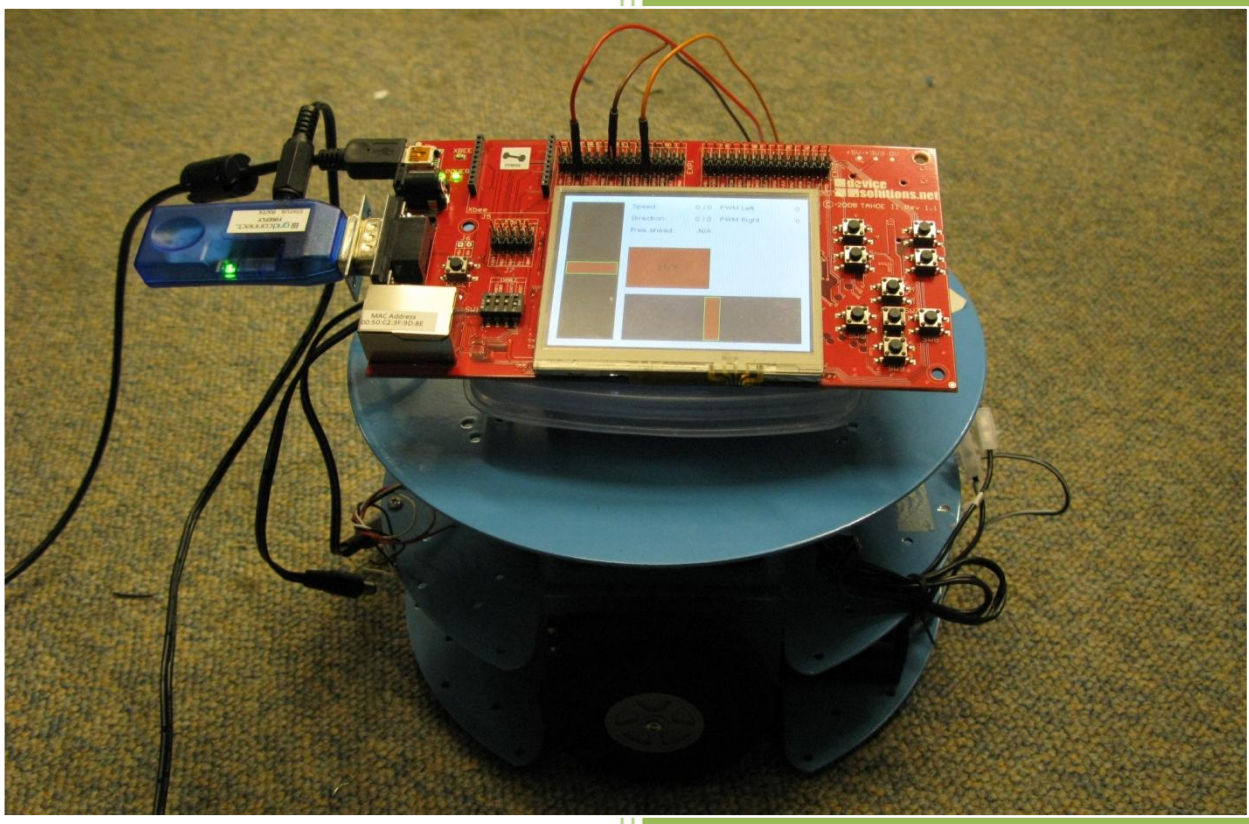
Blue Rogue Remote



Daniel Christoffersen og Asbjørn Baagø

Ingeniørhøjskolen i Århus, ITWEM1

04-06-2009

Indholdsfortegnelse

Projektet er udført af:

_____
06523 – Daniel Christoffersen

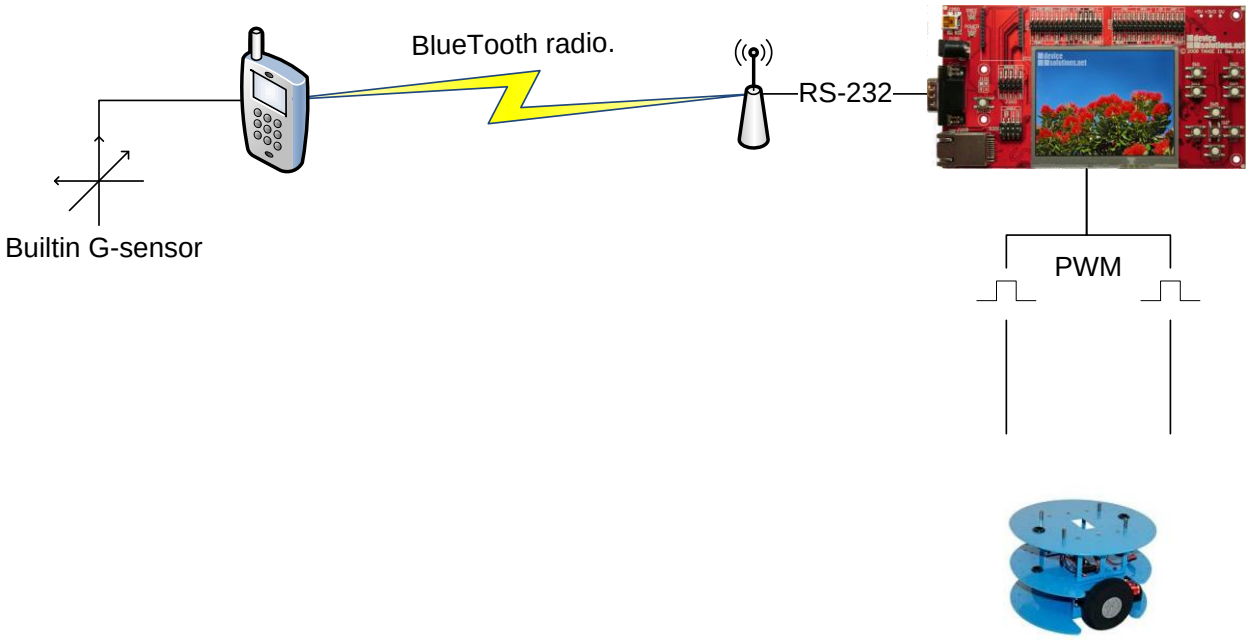
_____
05264 – Asbjørn Baagø

Indledning og systemoversigt

I dette projektforløb er udviklet et system, der kan styre en simpel robot (Blue Rogue fra Rogue Robotics) trådløst via en Windows Mobile-baseret mobiltelefon eller pda. Selve robotten styres af to PWM-signaler (Pulse Width Modulation – pulsbreddemodulering), der bliver genereret af et Tahoe II-kort fra device solutions.net. Dog, da kortet kun har én PWM-enhed i hardware, bliver signalerne genereret i software, hvilket selvsagt ikke giver samme præcision.

På den mobile enhed anvendes et indbygget accelerometer til at styre, hvordan robotten skal køre. Vippes enheden frem eller tilbage, styres hastigheden, og vippes der til siden, styres retningen.

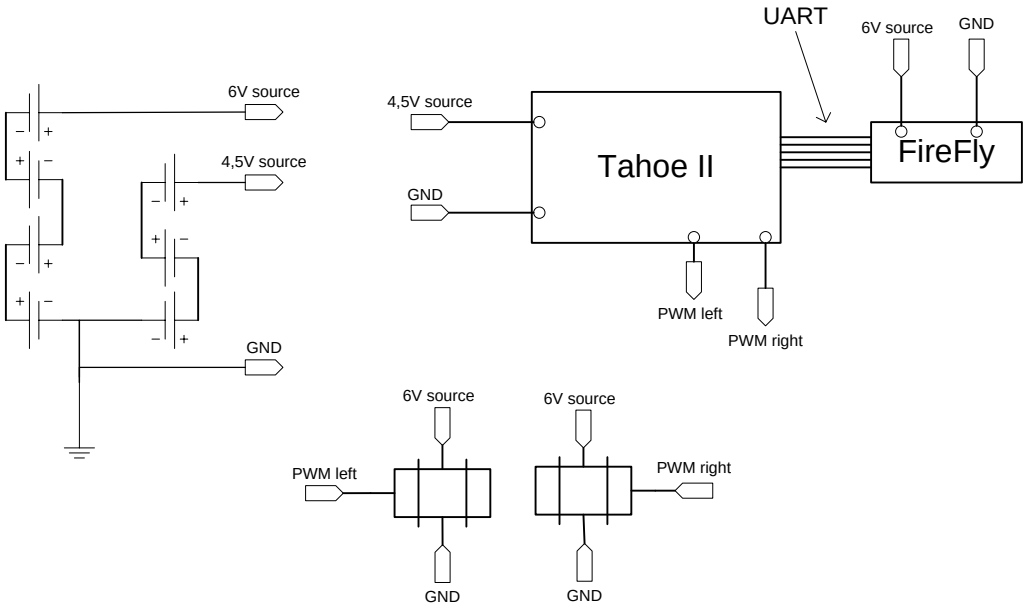
Kommunikationen mellem de to enheder foregår via Bluetooth, og data sendes i det mest simple format, man kan forestille sig: Én byte, hvor den ene nibble angiver hastighed, og den anden angiver retning. Da Tahoe II-kortet ikke har Bluetooth indbygget, anvendes en FireFly serieladapter fra GridConnect.



Figur 1 Systemoversigt

Hardware-design

Ledningsnet



Figur 2 Ledningsnet

Batterierne, der benyttes, er C-celler på 1,5 V. Grunden til at der er to batteribanker er, at Tahoe II kun kan forsynes med 4,5-6 V. Men da batterierne har en spænding på lidt over 1,5 V pr. stk., når de er friske, kan Tahoe II ikke bruge batteribank med 6 V. Derfor er der monteret en ekstra bank på 4,5 V. Her er problemet så igen. Det skal være forholdsvise friske batterier for at holde en spænding på 4,5 V eller derover. Når spændingen falder til under 4,5 V, vil Tahoe II slukke. Motorer og FireFly holdes på 6 V batteribanken, så deres dræn ikke har indflydelse på Tahoe IIs forsyning.

Software-design

Kommunikation

Robotten kan fjernstyres via serielporten, hvori der i dette projekt er placeret en FireFly-enhed, som giver mulighed for at anvende Bluetooth som bærer af de serielle data.

Der er kun implementeret envejs-kommunikation, så selve robotten har ingen mulighed for at sende status tilbage til fjernstyringsenheden.

Der anvendes en meget simpel protokol til kommunikationen. Den består af i alt én byte, som kan skitseres på denne måde:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Sign	Value			Sign	Value		
Direction				Speed			

Tabel 1 Kommunikationsprotokol

Hvis Sign er 1, ses værdien som værende negativ. Ved negativ Direction, vil robotten dreje til venstre, og ved negativ Speed, kører den baglæns.

Denne protokol giver selvsagt kun adgang til 7 værdier i hver retning, men det er mere end rigeligt til den anvendte robot.

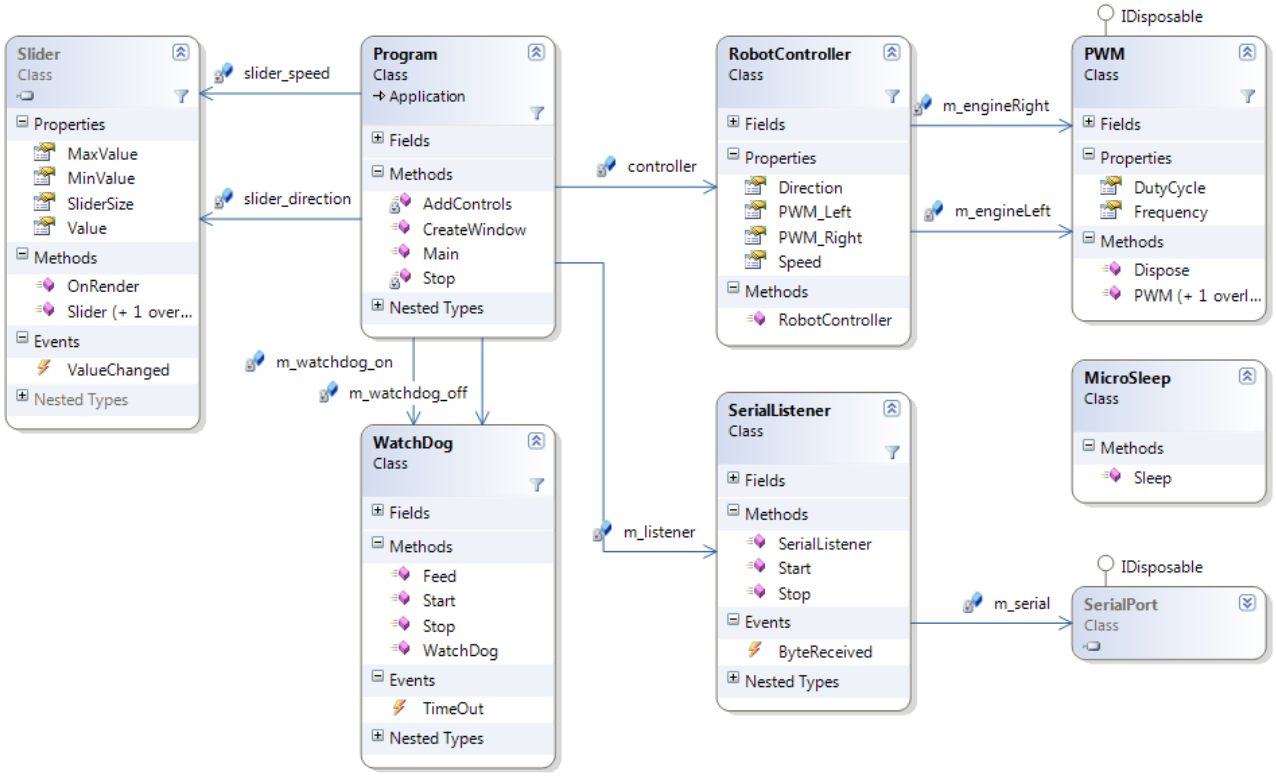
Robot

Softwaren til robotten er udviklet i Microsofts .NET Micro Framework og afvikles på et Tahoe II evaluation board fra device solutions.net.

Softwaren er udviklet til både at kunne styres lokalt og via fjernkontrol. Da Tahoe II-kortet har et trykfølsomt display er det blevet udnyttet til at lave to slidere, som kan anvendes til at styre robottens hastighed og retning. Disse to slidere vil til enhver tid følge robottens aktuelle hastighed og retning.

Figur 3 Robotdisplay under kørsel

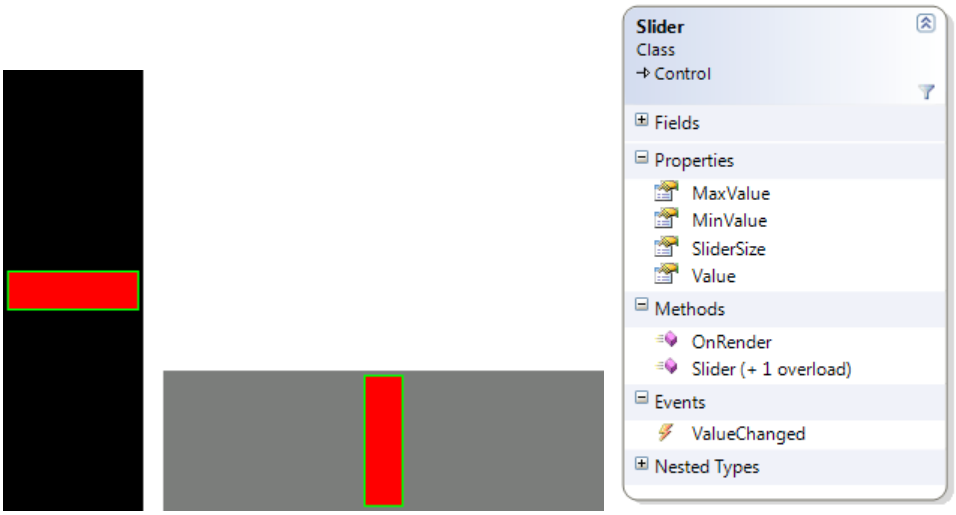
Herunder ses en oversigt over de klasser, der er anvendt.



Figur 4 Klasseoversigt, Robot

Slider

Denne klasse implementerer en Slider Control, som vist nedenfor.



Figur 5 User Control, Slider

Slideren er implementeret således, at den senere kan anvendes i andre projekter med ingen eller meget begrænset modifikation. Følgende værdier kan sættes dynamisk:

- Størrelse (bredde og højde)
- Retning (horisontal, vertikal)

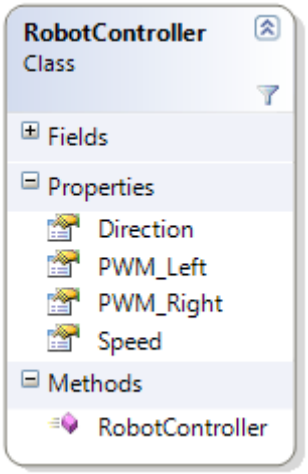
- Værdigrænser (minimum og maksimum)
- Størrelse på selve slideren
- Baggrundsfarve

Slideren kan selvfølgelig styres via touch, men man kan også sætte dens værdi ved at skrive til Value.

Hver gang værdien ændres, uanset hvordan det sker, trigges sliderens ValueChanged event.

RobotController

RobotController giver et interface til robotten.

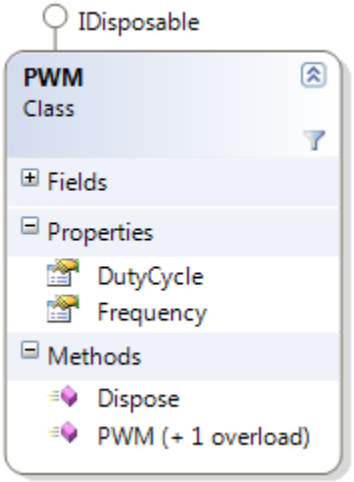


Figur 6 Klasse RobotController

RobotController anvender to instanser af PWM-klassen (se nedenfor) for at styre robotten; én til hver motor. Speed og Direction kan både sættes og læses, mens PWM_Left og PWM_Right kun kan læses.

PWM

PWM-klassen laver et PWM-signal på den udgangspin, som angives til constructoren.



Figur 7 Klasse PWM

PWM-signalet er software-genereret og derfor ikke 100 % stabilt. Den software-baserede løsning er valgt på baggrund af, at der på Tahoe II-kortet kun er én PWM-generator i hardware. En anden løsning kunne have været at anvende den hardware-baserede generator sammen med én software-baseret generator. Denne løsning er dog fravalgt med begrundelsen, at det er nemmere at arbejde med to ensfungerende (lige ustabile) generatorer i modsætning til at skulle forsøge at holde to vidt forskellige generatorer synkroniserede. Man kunne forestille sig en række mere kreative løsningsforslag, men da denne software-baserede løsning har vist sig at fungere fint i praksis, er den blevet valgt.

Ved tests har det vist sig, at robotten arbejder bedst, når PWM-frekvensen sættes til 400 Hz. Herved opnås den effekt, at hjulene står tilnærmelsesvist stille ved en duty cycle på 50 %. Jo længere man bevæger sig fra 50 %, desto hurtigere drejer hjulet så rundt. Retningen er afhængig af, om man bevæger sig over eller under 50 %. Det har ikke været muligt at finde nogen omfattende dokumentation til robotten. Derfor er værdierne fundet ved egne tests.

Da signalet er software-genereret, kan man som nævnt ikke regne 100 % med det resulterende signal. Måling med oscilloskop viser, at de 400 teoretiske Hz i virkeligheden ligger nærmere 350 Hz. Dette skyldes dels trådprioriteringer og dels overhead i funktionskaldene. Der er trods alt ikke tale om et real-time system.

MicroSleep

MicroSleep er en lille hjælpeklasse, der indeholder én statisk funktion.

Figur 8 Klasse MicroSleep

Sleep() anvender system ticks til at opnå en mere præcis tidsangivelse. Der er testet to udgaver af den løkke, der udgør selve vente-funktionen. Den ene laver en Thread.Sleep(0) (dvs. frigiver cpu'en kortvarigt) mellem hver sammenligning med stoptiden, mens den anden forsøger at holde fast for at kunne kigge oftere. Det er tydeligt på oscilloskopet, at den sidste mulighed er mere stabil, men den dræner også systemet for ressourcer og gør det dermed svært at opretholde en jævn opdatering af displayet. I dette projekt er derfor valgt den mindre ressourcekrævende løsning, da unøjagtighederne i PWM-signalet vurderes at have minimal indflydelse på den samlede oplevelse, mens en mere flydende grafik er tydelig for enhver.

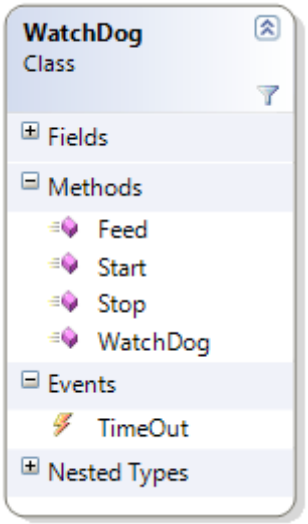
SerialListener

Figur 9 Klasse SerialListener

SerialListener implementerer en tråd, der lytter på serielporten på Tahoe II-kortet. Hver gang den modtager en byte, sender den et event med den modtagne byte.

WatchDog

Denne klasse stiller en WatchDog timer til rådighed.



Figur 10 Klasse WatchDog

Denne giver mulighed for at sætte en timeout for en handling, som så kan skubbes ved at ”fodre hunden”. Klassen anvendes til at styre bagbelysningen i displayet, således at der kan spares strøm.

Free ahead?

Den opmærksomme læser vil have opdaget, at der i Figur 3 vises teksten ”Free ahead” med værdien ”N/A”. Dette henviser til muligheden for at se, om der er forhindringer foran robotten. Robotten har påmonteret en optisk afstandsmåler, som kan aflæses ved hjælp af en analog indgang til Tahoe-kortet. Denne

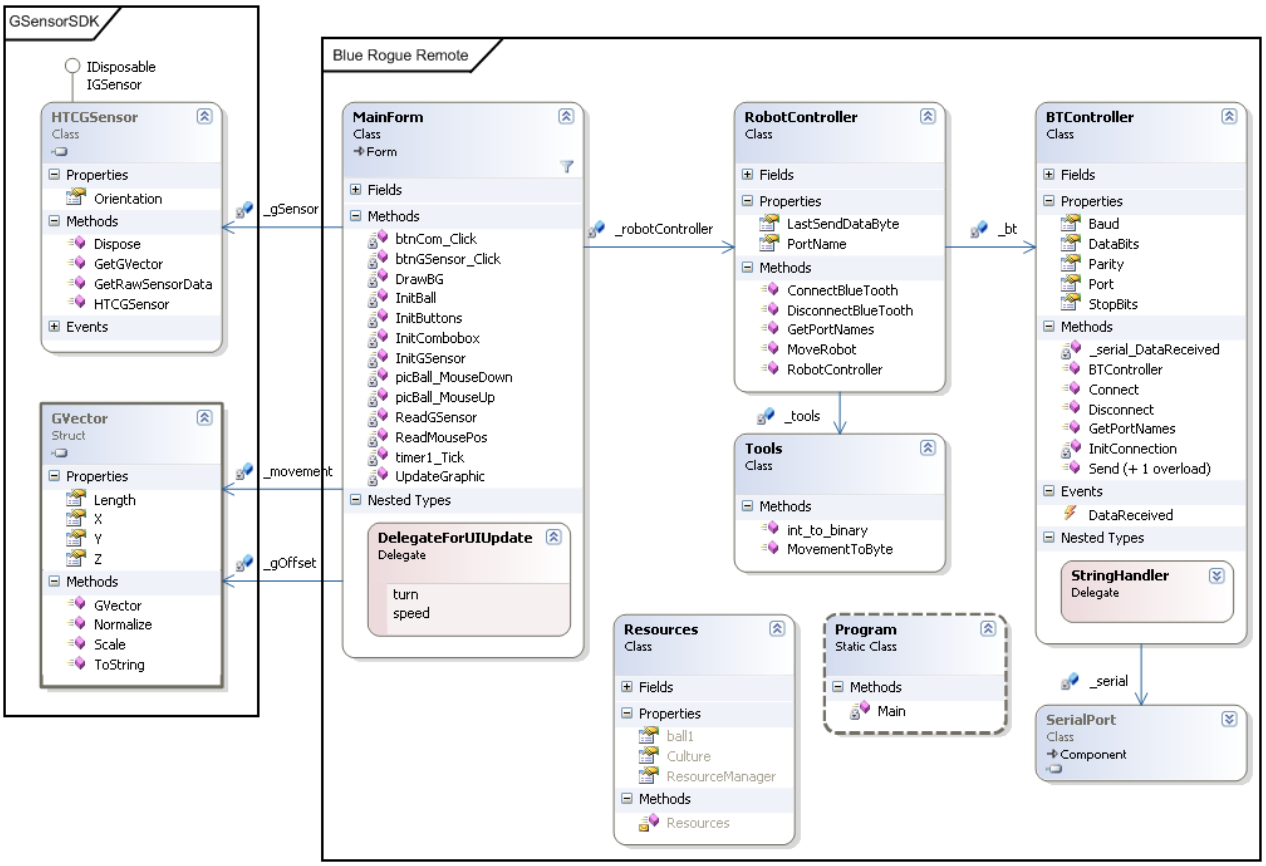
funktionalitet har været undersøgt og afprøvet, men desværre gav det visse ulemper i forhold til den almindelige drift af systemet. F.eks. var det kun ganske sjældent, at det trykfølsomme display registrerede input, når afstandsmåleren var i brug. Dette henføres umiddelbart til det faktum, at den analoge indgang er en del af display-controlleren, og at der derfor er en eller anden form for konflikt.

Endvidere lider afstandsmåleren af et par sjove uhensigtsmæssigheder. Blandt andet kan den ikke korrekt registrere afstanden, når emner er nærmere end ca. 2 – 3 cm. I det tilfælde, vil afstanden se ud til at være længere væk.

Alt i alt blev vurderes, at afstandsmåleren vil bidrage med så lidt værdi til det endelige produkt, at det ikke opvejer de betydelige mangler og problemer, som følger med anvendelsen. Derfor blev der ikke lagt den store energi i at finde en løsning på problemet med det trykfølsomme display, selvom der ganske givet findes en. Funktionaliteten er derfor implementeret som test men udkommenteret.

Det kan dog bemærkes, at der er foretaget målinger og beregninger i Excel med henblik på at kunne omsætte en målt spænding til en fysisk afstand. På baggrund heraf er klassen DistanceSensor implementeret.

Mobile



Figur 11 Windows Mobile klassediagram

Beskrivelse

Mobilprogrammet er selve fjernbetjeningen til robotten. Det er herfra, der sendes kommandoer til robotten med retning og hastighed. Der er implementeret to måder at styre robotten på. Enten via touchskærmen eller vha. mobilens indbyggede G-sensor. Disse to udelukker hinanden. Det vil sige, at hvis G-sensoren er aktiveret, kan man ikke bruge touch. Og hvis G-sensoren er deaktiveret, kan man benytte touch. For at læse på G-sensoren har vi på internettet fundet et GSensorSDK, der kan give os adgang til de informationer, vi ønsker fra mobilens accelerometer.

Klassebeskrivelser

GSensorSDK

Denne SDK er fundet på nettet. Den er lavet ud fra en god portion reverse engineering, da HTC ikke ville frigive et SDK til G-sensoren på mobilen.

Selve reverse engineering-delen er lavet af Scott Mitchelle og kan findes på:

http://scottandmichelle.net/scott/comments.html?entry=784

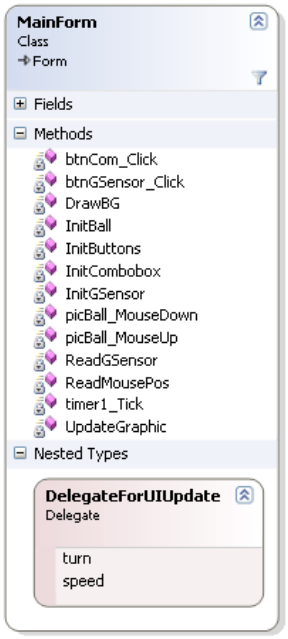
GSensorSDK'en er lavet af Koushik Dutta, men er stadig baseret på Scott Michelles forarbejde. SDK'en kan hentes her:

http://blog.enterprisemobile.com/wp-content/uploads/2008/07/htcsensorsdk.zip

GSensorSDK'en er lavet specifikt til HTC Diamond mobilen, og det er også på denne platform, vi har brugt den.

Gennem GSensorSDK'en kan der læses på alle tre akser; x, y og z. Vi bruger dog kun x og y.

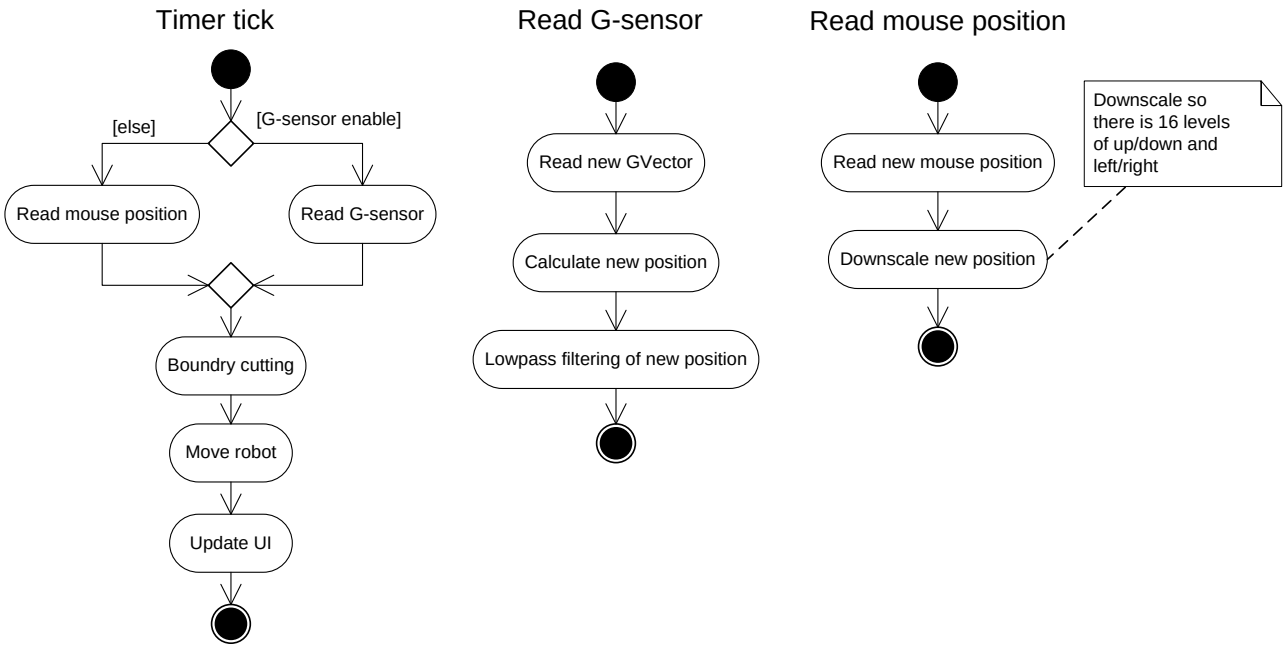
MainForm



Figur 12 Klasse MainForm

Hovedansvaret for denne klasse er at opdatere grafikken og reagere på inputs. Det er denne klasse som læser på både touch og G-Sensor.

Ved hjælp af en timer læser vi på G-Sensoren med 100 ms mellemrum. Herefter omformes de fundne informationer om hældningen på mobilen til en kommando-byte. Hvis G-sensoren ikke er aktiveret, kan man bruge touchscreen'en i stedet. Dette gøres ved at tage fat i "kuglen" og flytte den derhen, hvor man gerne vil have den. Det er den samme timer, der bruges til dette. Se Figur 13 Timer tick.



Figur 13 Timer tick

Grunden til at det er den samme timer der benyttes til både G-sensor og touch, er for at holde ændringerne i grafikken og den serielle kommunikation på det samme niveau, både når der bruges G-sensor, og når der bruges touch.

For at mindske rystelserne fra G-Sensoren er der implementeret et simpelt lavpasfilter. Dette giver en noget bedre fornemmelse, når man bruger G-Sensoren til at styre med. Filteret er implementeret som følgende:

$$position = 0{,}75 \cdot position + 0{,}25 \cdot new\ reading$$

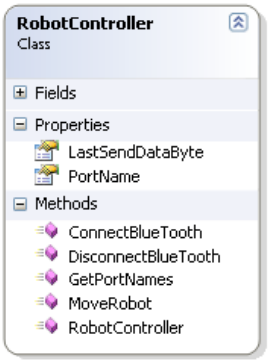
Tidskonstanten τ for dette filter er på

$$\tau = \frac{0{,}75 \cdot dt}{0{,}25} = \frac{0{,}75 \cdot 0{,}01}{0{,}25} = 0{,}3s$$

Variablen dt er tiden mellem målingerne. I vores tilfælde læser vi hver 100 ms, altså en dt på 0,01 s.

En tidskonstant på 0,3s betyder, at der går op til 0,3 sekunder, før position er trukket helt op til den nye værdi. Dette giver et lille delay i målingerne, men det er ikke mærkbart. Og gevinsten med den blødere styring opvejer dette lille delay. Alt i alt fungerer dette godt, og styringen føles nem og kontrolleret

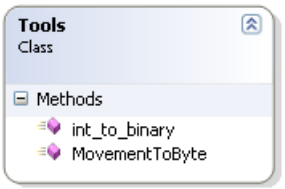
RobotController



Figur 14 Klasse RobotController

Gennem denne klasse er det muligt at sende kommandoer til robotten. Når MoveRobot-funktionen kaldes, omsættes de to input speed og turn, til en samlet kommando-byte vha. Tools-objektet. Når denne kommando-byte er lavet, sendes den via BTControlleren til robotten.

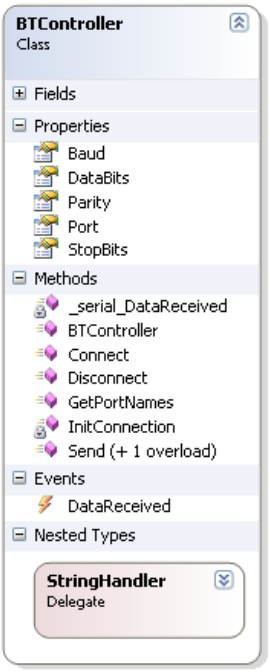
Tools



Figur 15 Klasse Tools

Tools bruges i forbindelse med konverteringen fra de to input, speed og turn, til den kommando-byte, der skal sendes til robotten.

BTController



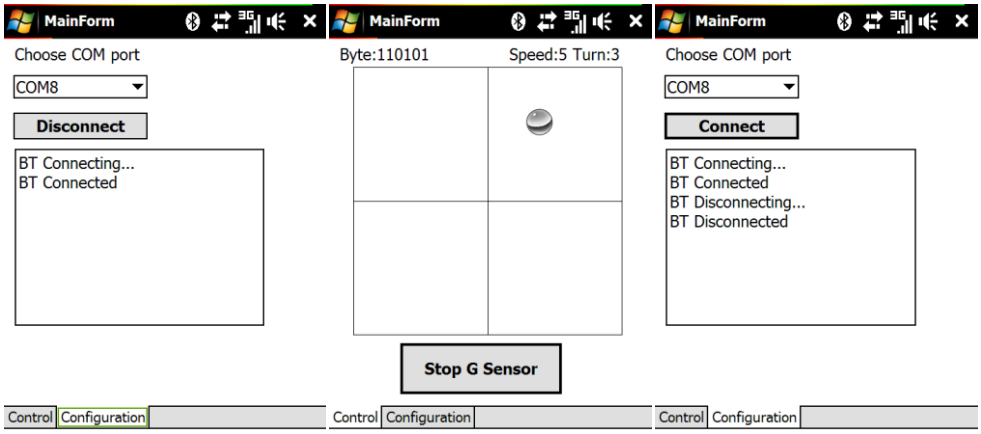
Figur 16 Klasse BTController

BTController styrer den serielle kommunikation til robotten. Det er muligt at ændre på forbindelsesindstillingerne. BTController er hardcoded til følgende indstillinger:

COM port	COM8
Baud rate	9600
Paritet	Ingen
Data bits	8
Stop bits	1

Tabel 2 Serielle indstillinger

Brugerfladen



Figur 17 Brugerflade

Overstående billeder viserer brugerfladen på mobilprogrammet. På configuration-fanebladet kan man vælge den COM port, der skal benyttes samt se log-informationer. Den COM port, der skal benyttes, er den, man har knyttet til FireFly'en i Windows Mobiles BlueTooth interface. Koden til FireFly'en er 1234. Denne skal bruges i forbindelse med parringen af mobilen og FireFly'en. På CD'en, under Bilag, er der en video, der viser opsætningen af BlueTooth og gennemgang i brugen af Blue Rogue Remote.

På Control-fanebladet kan man styre robotten, hvis man er forbundet til den. Hvis man trykker på 'Start G Sensor', styrer man ved at bevæge mobilen. Bevægelserne vises vha. kuglen i det markerede felt og tallene over feltet. I overstående eksempel kører robotten fremad med en hastighed på 5 og drejer til højre med 3. Den kommando-byte, der sendes, står i toppen til venstre. I dette tilfælde er den 00110101.

Hvis G-sensoren ikke er startet, kan man tage fat i kuglen, enten med en stylus eller en finger, og på den måde styre robotten.

Resultater

Der er i dette projektforløb udviklet et system til lokal og trådløs styring af en simpel robot. Systemet er baseret på en Blue Rogue-robot, som skal drives ved hjælp af to PWM-signaler; ét til hvert hjul. Dette håndteres af et Tahoe II-kort, hvortil der er udviklet software baseret på Microsoft .NET Micro Framework. PWM-signalerne genereres i software, da der kun er hardware-understøttelse af én PWM-udgang på Tahoe II-kortet. Det resulterer i, at signalet kører en anelse langsommere og mere ustabilt, end hvis det havde været hardware-genereret. Problemet er dog ikke større, end at det sagtens kan lade sig gøre at styre robotten.

Der er også via Tahoe-kortets trykfølsomme display implementeret en simpel brugergrænseflade til den lokale styring. Den består af to egenudviklede slidere, der styrer henholdsvis hastighed og retning. Herudover er der en STOP-knap og nogle status-felter. Disse slidere vil også vise status for robotten, når den er styret fra serielporten. Dette dog med den undtagelse, at der for at spare strøm er implementeret et system, der slukker for displayet fem sekunder efter, at serielporten har overtaget kontrollen. Når der i et sekund ikke er modtaget data fra serielporten, stoppes robotten, og displayet tændes igen.

Fjernstyring er implementeret på robotsiden ved hjælp af Tahoe II-kortets serielport, hvorpå der konstant lyttes efter data. Hver modtaget byte omsættes til en værdi for hastighed og retning som beskrevet i afsnittet Kommunikation ovenfor. For at opnå trådløs styring er der monteret en FireFly-enhed, som omsætter Bluetooth til almindelig RS232.

Der er også udviklet en mobilklient til styring af robotten. Denne klient er udviklet i Microsoft .NET Compact Framework og er tilpasset en HTC Touch Diamond mobiltelefon. Klienten giver mulighed for at styre robotten både ved hjælp af mobiltelefonens trykfølsomme display og ved hjælp af telefonens indbyggede G-sensor.

Konklusion

Det er altså muligt at styre Blue Rogue robotten via Tahoe II-kortet ved anvendelse af software-baserede PWM-signaler. Det er også muligt at lave et simpelt interface på Tahoe II-kortet, der anvender det trykfølsomme lcd-display til nem og intuitiv betjening af robotten.

Samtidigt et det muligt at lave en simpel og hurtig, trådløs kommunikationskanal mellem Tahoe II-kortet og en Windows Mobile-enhed ved anvendelse af Bluetooth og seriel kommunikation, og det er også lykkedes at lave et enkelt og brugervenligt kontrol-interface på mobiltelefonen, dels ved anvendelse af trykfølsomt display og ved anvendelse af G-sensor.

Der er udviklet et velfungerende system, der virker stabilt og hurtigt.

Bilag

Bilag forefindes elektronisk i mappen Bilag på rapportens vedlagte cd-rom.

- Tahoe II Development Kit Technical Reference Manual (Tahoe-II Technical Reference Manual.pdf)
- Blue Rogue Remote opsætningsvideo (Blue Rogue Remote.wmv)