
Multiagent Robot systemer

Specialkursus forår 2007

Christian Agerbeck, s021772

Kongens Lyngby 2007

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Forord

Denne rapport er skrevet i perioden mellem d. 1. februar og d. 29. juni 2007. Rapporten er et polyteknisk special kursus på IMM, DTU.

Jeg vil gerne takke Assisterende Professor Thomas Bolander for vejledning og hjælp igennem forløbet.

Lyngby, Juni 2007

Christian Agerbeck

Indhold

Forord	i
1 Indledning	1
1.1 Problembeskrivelse	1
2 Jagtdomænet	3
2.1 Design af jagtdomæne-instansen	4
3 LEGO NXT miljøet	7
3.1 Krav	7
3.2 Design af robotterne og verdenen	8
3.3 Konklusion	14
4 Agenterne	15
4.1 Enkeltagent system versus Multiagent system	15
4.2 Homogene versus heterogene agenter	16
4.3 Agent arkitektur	16
4.4 Analyse	17

4.5	Design	19
4.6	Implementering	22
4.7	Test	25
4.8	Diskussion	27
4.9	Konklusion	27
5	Konklusion	29
A	Bevægelseslaget	33
B	Mindstorm NXT-G Program	35
B.1	Kode	35
C	Agentimplentering	37
C.1	GUI	37
C.2	Kode	38

Indledning

Robotter er et område som igennem de seneste årtier har fået mere og mere opmærksomhed. En fremtidsvision er, at vi med tiden vil opleve robotter, som i højere grad vil kunne operere, som om de tænker selv. For at opnå dette, er et af skridtene på vejen, at kunne ræsonnere sig frem til en plan som udfører et ønsket mål. Agent teorien er netop et område inden for Kunstig Intelligens der især beskæftiger sig med denne abstraktion.

Igennem rapporten antages det at læseren har et kendskab til generel agent teori. Derudover vil være en fordel at have læst Laidlaw et al.'s bachelor projekt [Laidlaw et al., 2007], som omhandler multiagent systemer i DTU's iMARS laboratorium.

1.1 Problembeskrivelse

Formålet med dette projekt er at benytte et jagt domæne til at illustrere kommunikations og koordinationsprincipper i et multiagent-system. I praksis skal dette opnås ved at lade en række identiske robotter i DTU's iMARS laboratorium fungere som jægere, hvis mål er at fange en bytte robot som bliver kontrolleret af en menneskelig agent.

Jagtdomænet

Jagtdomænet er også internationalt kendt som Predator/Prey eller Pursuit domænet [Stone & Veloso, 1997]. I denne rapport vil det dog blive benævnt som jagtdomænet. Dette domæne er velegnet til at illustrere samarbejde og andre koordinations principper i et kunstigt intelligent system. Princippet bag jagtdomænet er velkendt: Et antal jægere (predators) skal fange et bytte (prey). Dette relativt komplicerede domæne kan herefter simplificeres ved at definere en række parametre og ”spilleregler” som skal gælde for en instans af domænet. En velkendt instans af jagtdomænet er f.eks. computerspillet PacMan ¹, hvor et antal spøgelser skal forsøge at fange PacMan.

Stone foreslår følgende variable parametre i jagtdomænet:

- Størrelse og udformning af verdenen
- Lovlige træk
- Definition af tilfangetagelse
- Antal af jægere og bytte
- Simultan eller sekventiel bevægelse
- Synlighed og rækkevidde
- Kommunikation imellem jægere
- Bevægelse af bytte

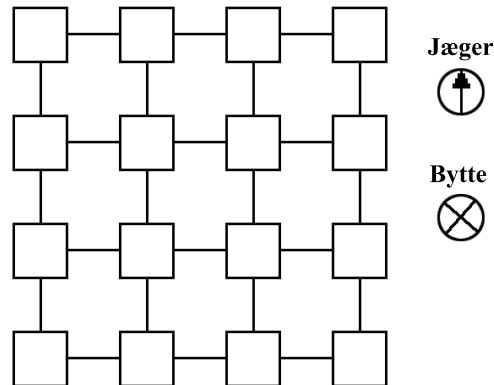
Tabel 2.1: Parametre i jagtdomænet

Definition af jagtdomæne instansen benyttet igennem denne rapport, vil bygge ovenpå den LEGO verden og funktionalitet, som er udviklet af Laidlaw et al., og vil blive uddybet i Kapitel 3.

¹<http://en.wikipedia.org/wiki/Pac-Man>

2.1 Design af jagtdomæne-instansen

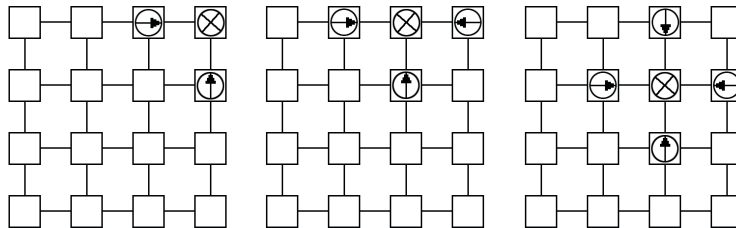
LEGO verdenen benyttes til at anskueliggøre domænet. Denne består af et 4x4 kvadratisk gitter, hvor hvert gitterknudepunkt er forbundet som vist på figuren.



Figur 2.1: Plade

Eftersom LEGO NXT robotterne kun kan bevæge sig langs gitterlinjerne defineres et lovligt træk, som en bevægelse vertikalt eller horisontalt langs en gitterlinje.

En tilfangetagelse defineres som en situation hvor byttet er omringet af jægere og ikke længere kan bevæge sig. Denne situation kan i den viste verden opstå med 2, 3 eller flere jægere.



Figur 2.2: Tilfangetagelse med hhv. 2, 3 og 4 jægere

Normalt benyttes der kun ét bytte i jagtdomænet. Flere bytter komplicerer samarbejdsproblemerne og koordineringen, da det herefter imellem jægerne skal besluttes, hvilket bytte man samarbejder om at fange. Derfor benyttes blot ét bytte i denne instans af domænet. Antallet af jægere bliver bestemt af de praktiske muligheder. Da der i realiteten er 5 LEGO NXT robotter til rådighed, kunne 4 jægere benyttes, men da gitter-verdenen er forholdsvis begrænset, er det tilstrækkeligt med 3 jægere for derved ikke at give byttet for få flugtmuligheder.

Problemet i domænet antages at skulle løses kontinuert, det er derfor besluttet at parterne bevæger sig simultant.

Jægerne skal vha. deres sensorer kunne observere verdenen foran dem. Rækkevidden er bestemt af de fysiske egenskaber på sensorerne og vil blive behandlet senere i rapporten. I flere jagtdomæne instanser bevæger byttet sig tilfældigt, i dette vil byttet dog blive kontrolleret af en menneskelig agent. Denne vil fysisk have indblik i verdenens tilstand og derfor vil byttet aktivt kunne forsøge at undgå jægerne.

For at opveje byttets åbenlyse fordel får jægerne kendskab til hinandens position i gitterverdenen, derudover kommunikerer de deres sensor observationer med hinanden for at få en mere nuanceret opfattelse af verdenen.

LEGO NXT miljøet

Multiagent systemet tænkes vist i et fysisk miljø. I iMARS laboratoriet på DTU eksisterer allerede en fungerende rammeløsning udviklet af Laidlaw et al.. Denne rammeløsning indeholder følgende:

1. En gitterverden hvorpå robotterne kan bevæge sig
2. Et færdigbygget antal robotter
3. En række klasser og hjælpebiblioteker m.m. til kommunikation imellem computer og robotter
4. LEGO NXT-G program der indeholder de basale bevægelsesmønstre og linjefølgning til robotterne
5. En fungerende implementering af et multiagent system beregnet til at flytte objekter.

Tabel 3.1: Egenskaber for iMARS robotverdenen

3.1 Krav

Denne rammeløsning benyttes derfor til at skabe et multiagent system der kan løse problemstillingen beskrevet i kapitel 1 og 2. Først defineres et antal krav som skal opfyldes. Der er to typer af agenter som skal operere i miljøet, hhv. jægere og bytte, disse skal begge kunne repræsenteres vha. de robotter som miljøet

stiller til rådighed. Kravene til hardwareopsætning er inspireret af dem beskrevet i [Laidlaw et al., 2007].

Navigation Robotterne skal kunne bevæge sig rundt i verdenen. Derudover skal verdenen være så simpel at robotten har kendskab til sin egen position ud fra de bevægelser (træk) den har foretaget sig.

Kommunikation Robotterne skal kunne kommunikere med hinanden således at de i fællesskab kan løse problemer.

Opfattelse af miljøet Robotterne skal have en opfattelse af miljøet. Denne opfattelse skal række, udover kun at kende til sin egen position, de skal ligeledes have en mulighed for at kunne lokalisere et bytte som befinder sig på en anden position. Systemet behøver ikke at fuldt observerbart, men skal som minimum være partielt observerbart. Dette vil sige at dele af hele systemets tilstand skal kunne bestemmes ud fra sensor data [Russel & Norvig, 2002].

Reaktivitet Robotterne skal inden for et kort tidsrum kunne reagere på ændringer, således at de opfylder deres mål [Wooldridge, 2002].

Stabilitet Systemet skal være stabilt. Da flere robotter skal operere sammen for at løse et problem i et dynamisk miljø er det nødvendigt at robotten udfører sine aktioner på en stabil vis.

Eftersom byttet styres af en menneskelig agent, skal denne robot kun opfylde kravene om **navigation** og **stabilitet**. Derudover skal det være muligt at for den menneskelige agent at kommunikere med byttet, for at kunne styre dens bevægelser. Det tænkes at byttet får de samme bevægelser til rådighed som jægeragenten. I princippet er miljøet fuldt observerbart for den menneskelige agent, og derfor er yderligere sensor informationer unødvendige.

3.2 Design af robotterne og verdenen

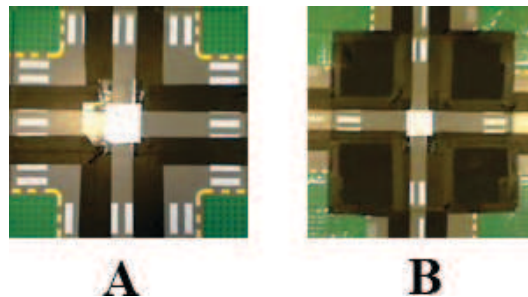
Selvom den eksisterende løsning tilbyder en færdigbygget verden og et antal færdigbyggede robotter, har det igennem projektet vist sig at justeringer og udvidelser til disse er nødvendige.

Verdenen

Verdenen består af et antal LEGO vejplader samlet i et 4x4 gitter. På disse er der markeret et antal kørelinjer, som robotten er i stand til at følge. Princippet er at robotten følger vejpladen indenfor to afmærkede sorte striber, og finder et knudepunkt markeret med sølv [Laidlaw et al., 2007]. Ved at benytte verdenen uændret viste det sig igennem projektet at en robot engang imellem foretog et forkert

retningskift, f.eks. drejede 180 grader i stedet for 90 og omvendt. Derved fik robotten et forkert kendskab til hvor den befandt sig, og destabiliserede systemet. Årsagen til dette skyldes at robotten drejede ”blindt” i den tidligere opsætning, dvs. at den blot drejede et vist antal hjulomdrejninger svarende til en vinkel på ca. 90 eller 180 grader.

Et problem i jagt domænet kræver adskillige retningskift, det var derfor nødvendigt at forbedre robottens retningskift. Løsningen blev at markere den umiddelbare omegn af alle knudepunkter med sort. Herefter kan robotten benytte de sorte områder til at navigere sig rundt i retningskiftet. F.eks. er tankegangen bag en 90 graders drejning til venstre at robotten drejer til venstre indtil den møder en sort overflade, herefter fortsætter den med at dreje indtil den møder linjen igen. For en uddybning af de resterende retningskift se appendiks A over de tilgængelige robotbevægelser.



Figur 3.1: Knudepunkt A viser den oprindelige konfiguration. B viser den nye konfiguration

Ved at benytte en simpel gitterverden sikres det at **navigations** kravet bliver opfyldt. En agent vil ligeledes være i stand til at bestemme sin position (ved hvilket gitterpunkt den befinder sig), ud fra de bevægelser (træk) den har foretaget i denne simple verden.

Robotterne

Det undersøges om det robotdesign, som rammeløsningen stiller til rådighed (NXT Mover), kan opfylde kravene til dette multiagent system. Mht. **navigation** er det af [Laidlaw et al., 2007] vist at robotten kan bevæge sig rundt i verdenen.

NXT Mover benytter sig af adskillige sensorer til at danne sig en opfattelse af miljøet. Alle sensorerne giver dog kun information om robottens pågældende position. Miljøet er fortsat partielt observerbart, men robotten kan ikke selv observere systemets tilstand udenfor sin egen position. Kravet om en **opfattelse af miljøet** er derfor ikke opfyldt, da robotten ikke selv har en opfattelse af de andre positioner.

Derudover er NXT Mover robotten bygget op med det formål at flytte objekter,

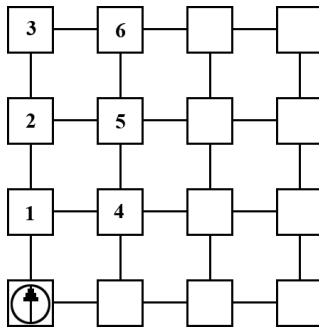
hvilket er overflødig i dette system. Det står derfor klart, at for at opfylde kravet om en **opfattelse af miljøet** skal sensoropsætning på robotten ændres.

Alle overflødige sensorer m.m. fjernes, og i stedet tilføjes en ultralydssensor til robotten. I første omgang undersøges egenskaberne ved ultralydssensoren.

Test af ultralydssensoren

Ultralydssensoren er i stand til at måle afstande mellem 0-255 cm med en præcision på ± 3 cm. Princippet i sensoren, er at der udsendes et ultralydssignal, hvorefter det måles hvor lang tid et ekko er om at nå tilbage. På dette vis kan afstanden til det nærmeste objekt bestemmes nogenlunde præcist.

For at teste sensorens egenskaber bygges en mur af LEGO-klodser som kan anbringes forskellige steder i gitterverdenen. Årsagen til dette er, at sensoren er utrolig følsom overfor det reflekterende objekts udformning. Derfor antages det også at byttet skal have en udformning der tager højde for dette. I dette forsøg skal muren repræsentere én side af byttet. Forsøget udføres ved at placere robotten i et gitterknudepunkt og måle afstanden til de andre knudepunkter ved at placere muren vinkelret på ultralydssignalet i de anførte positioner.



Figur 3.2: Målepositioner

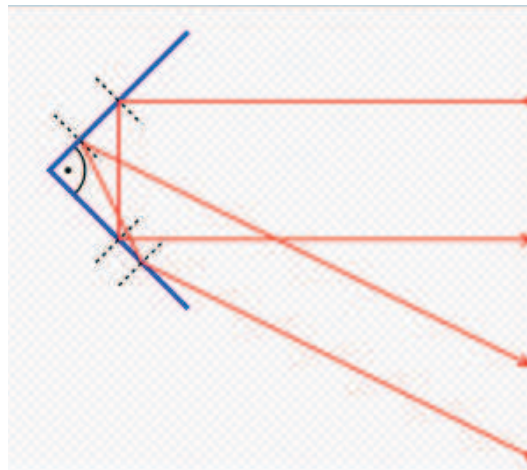
Det er valgt ikke blot at undersøge afstanden til position 1,2 og 3 men desuden afstanden til 4,5 og 6 for at få et billede af hvor stor spredning der er på det udsendte signal. De korrekte og målte værdier er anført i tabel 3.2.

Af målingerne kan det ses at ultralydssensoren måler den korrekte afstand med en afvigelse på 1-4%. Ligeledes ses det at signalet ikke reflekteres ved nogen af positionerne 4,5 eller 6. Ved en tom måling returneres værdien 255, hvilket betyder at sensoren har ikke opfanget nogen objekter indenfor dens øvre grænse. Under forsøget er det dog afprøvet at placere en robot midt imellem position 1 og 4, og her opfanges et ekko. For at afprøve hvor følsom sensoren er overfor den relative placeringen af muren, afprøves de samme positioner med muren anbragt i en vinkel på 45 grader i forhold til signalet. Ved disse målinger kunne signalet ikke opfanges.

Position	Afstand	Måling 1	Måling 2	Måling 3	Gns. afvigelse
1	60	57	56	61	4%
2	111	112	111	112	1%
3	161	163	162	255	1%
4	-	255	255	255	-
5	-	255	255	255	-
6	-	255	255	255	-

Tabel 3.2: Ultralydsmålinger

Dette skyldes måden hvorpå signalet reflekteres. Ved en vinkel på f.eks. 45 grader bliver signalet ikke sendt tilbage i retning af robotten, men derimod ud i rummet. Sensoren har derfor ingen mulighed for at opfange det direkte reflekterede signal, men vil i stedet måle eventuelle ekkoe. Dette er et problem, eftersom placeringen af byttet ikke er konstant vinkelret på signalretningen. For at løse dette benyttes et princip kendt fra hav-bøjer kaldet corner cube reflector¹. Princippet kan ses på figur 3.3.



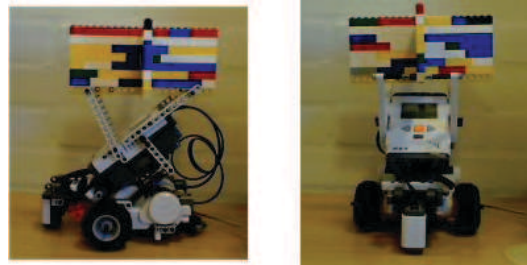
Figur 3.3: Princippet i en corner cube reflector

Der bygges derfor en corner cube reflector i legoklodser og denne monteres oven på en af LEGO NXT robotterne, som vist på billedet 3.4. Denne skal herefter fungere som bytte-robot igennem projektet.

Herefter gentages forsøget ved en vinkel på 45 grader, således at cornercube princippet kan afprøves. Resultatet kan ses i 3.3.

Det ses ud fra måleresultaterne at afstanden til knudepunkterne foran robotten kan bestemmes nogenlunde præcist. Ydermere kan sensoren, p.g.a. spredningen af signalet, opfange objekter tæt ved positionerne 1, 2 og 3. Det antages derfor at

¹http://en.wikipedia.org/wiki/Corner_cube



Figur 3.4: Bytterobotten

Position	Afstand	Måling 1	Måling 2	Måling 3	Gns. afvigelse
1	60	60	60	60	0%
2	111	112	112	112	1%
3	161	163	161	255	20%

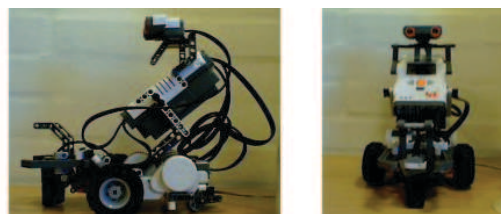
Tabel 3.3: Ultralydsmålinger med corner cube reflector

ultralydssensoren kan benyttes til at bestemme afstanden til positioner 1, 2 og 3 felter foran robotten. Hvilket samtidig vil være robotens synsvidde.

Afslutningsvis skal det bemærkes at ultralydssensoren desværre kun kan operere alene. Dette bevirker at hvis flere robotter forsøger at benytte deres sensor, vil det medføre forkerte målinger. Dette giver nogen begrænsninger som senere vil blive løst.

Jæggerrobotten

Det endelige fysiske design af jæggerrobotten, bliver således en modificeret version af NXT Mover robotten. Den grundlæggende opbygning bibeholdes, men fangarme og overflødige sensorer fjernes. I stedet monteres en ultralydssensor således at den udsender ultralydssignaler i en højde tilsvarende den højde reflektionsmuren er monteret på bytterobotten. Resultatet kan ses på billede 3.5.



Figur 3.5: Jæggerrobotten

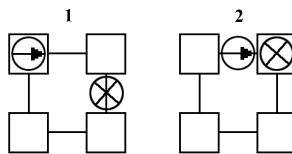
Kontrol af robotterne

Ved at benytte den eksisterende rammeløsning kan der opnås en avanceret kontrol af robotterne. Princippet er, at der på NXT robotterne kører et program der varetager en række basis operationer, såsom at følge en linje, dreje ved et knudepunkt osv.. Dette program kan opfattes som et lavniveau bevægelseslag, som reagerer på en række simple kommandoer. Dette bevægelseslag tænkes styret af et adfærdslag på et højere niveau. Dette adfærdslag vil i denne rapport blive implementeret på en pc. Kommunikationen imellem de to lag foregår igennem et kommunikationslag, som bliver varetaget af NXTRemoteControl [Laidlaw et al., 2007]. Dermed sikres det samtidig at **kommunikations** kravet er opfyldt. Agentarkitekturen, og hvorledes denne opfylder kravene til miljøet, vil blive implementeret i adfærdslaget, hvilket behandles i næste kapitel.

Inden bevægelseslaget behandles, overvejes det i hvilket lag kontrollen af ultralydssensoren skal ligge. Hvis kontrollen placeres i bevægelseslaget, vil robotten være i stand til at reagere direkte på sensor inputtene. Dette kræver dog en form for protokol der kan sørge for, at kun én robot forsøger at benytte sensoren ad gangen. En sådan protokol vil indebære en nødvendig kommunikation enten direkte imellem robotterne, eller imellem de forskellige programlag. Eftersom det er muligt at læse sensordata fra robotten direkte igennem NXTRemoteControl [Laidlaw et al., 2007] biblioteket, forekommer det velegnet at ligge kontrollen i adfærdslaget i stedet.

Et færdigt bevægelseslag for NXT Mover robotten er til rådighed, dog er dette ikke længere tilstrækkeligt idet sensoropsætningen på robotten er ændret. Derudover er verdenen ligeledes ændret således at et retnings skift skulle blive mere stabilt. Et nyt bevægelseslag implementeres efter samme princip som det eksisterende, dvs. at programmet kontinuerligt modtager en kommando igennem kommunikationslaget, som udføres hvorefter robotens status sendes tilbage igennem kommunikationslaget. I appendiks kan ses en beskrivelse af de tilgængelige kommandoer og i appendiks ses den egentlige NXT-G implementering.

For at sikre at robotten er **reaktiv** er linjefølge-rutinen ændret. Dette skyldes at det skal være muligt at stoppe robotten imellem 2 knudepunkter, f.eks. hvis en jæger er ved at kolliderer med byttet. Dette kunne f.eks. ske, hvis jægeren forsøger at rykke til et tomt knudepunkt, hvorefter byttet ankommer til knudepunktet først (se figur 3.6).



Figur 3.6: Kollision

Denne situation vil imidlertid ikke opstå imellem jægerne da det antages at de altid

ved hvor hinanden befinder sig. Dette aspekt vil blive håndteret af adfærdslaget.

3.3 Konklusion

Det er i kapitlet beskrevet hvordan det eksisterende iMARS LEGO miljø udvides og ændres således at det opfylder kravene til et multiagent system i jagtdomænet. Ved at benytte en simpel verden og den eksisterende rammeløsning er kravene om **navigation** og **kommunikation** opfyldt. Ved at benytte en ultralydssensor som ”øjne” får robotten en **opfattelse af miljøet**. Til sidst ved at omskrive det eksisterende bevægelseslag sikres det at robotten opnår en vis **reaktivitet**. Rammerne er desuden defineret for et **stabilt** system, dog har det igennem projektet vist sig at sensorer og hardware ikke altid opfører sig som forventet.

Agenterne

Med den fysiske verden, NXT bevægelseslaget, og kommunikationslaget på plads kan det egentlige agentsystem blive udformet. Agentsystemet berører kun jægeragenterne, da bytteagenten som før nævnt kontrolleres af en menneskelig agent, og kan derfor opfattes som et separat system. I det følgende vil jægeragenterne derfor blot benævnes som agenter. Agentsystemet vil derfor repræsentere det adfærdslag som skal styre robotterne.

4.1 Enkeltagent system versus Multiagent system

Før det egentlige multiagent system udformes, er det relevant at se på dets åbenlyse modstykke, et enkeltagent system. Et enkeltagent system, er f.eks. et system hvor én agent styrer en række robotter, således at agentens mål kan opfyldes [Stone & Veloso, 1997]. I denne rapport foregår styringen og kommunikationen med de forskellige robotter igennem én pc. Dette giver åbenlys anledning til at opfatte enhver løsning med denne opbygning som et enkeltagent system. Argumentationen imod denne påstand er, at selvom agent systemet kører fysisk på én pc, er det opsplittet, således at forskellige tråde har kontrol over agenterne. Formålet er at skabe en abstraktion, der simulerer et fysisk multiagent system. I princippet er målet, at de implementerede agenter i multiagent systemet kan køres direkte på robotten, f.eks. vha. en lommecomputer koblet til robotten, således at robotterne bliver autonome.

Det er givet at et problem i jagtdomænet også vil kunne løses ved et enkeltagent system, dette vil dog ikke blive behandlet yderligere, og i det følgende vil udformningen af et multiagent system til jagtdomænet blive behandlet.

4.2 Homogene versus heterogene agenter

Den første overvejelse i multiagent systemet, er om det skal bestå af homogene agenter, eller heterogene agenter. Fordelene ved et homogent system er, at alle agenter er ens, dette giver et simpelt og overskueligt system. I et heterogent system vil der være agenter, der varetager forskellige roller. Dette giver et mere komplekst system, men samtidig også bedre mulighed for avanceret koordination. I dette projekt er det dog pga. tidsbegrænsningen valgt at fokusere på et homogent system, da et heterogent system blev for tidskrævende. Samtidig vil det være lettere at forstå, hvilke forbedringer m.m. der kan foretages med et heterogent system, ved at observere et fungerende homogent system.

4.3 Agent arkitektur

Agenterne i systemet skal være intelligente. Wooldridge beskriver tre egenskaber man kan forvente en intelligent agent besidder [Wooldridge, 2002].

Reaktivitet Intelligente agenter er i stand til at opfatte deres miljø, og inden for et kort tidsrum reagere, så de opfylder deres målsætning.

Proaktivitet Intelligente agenter er i stand til at udvise målbevidst opførsel ved at *tage initiativ*, med henblik på at opfylde deres målsætning.

Sociale evner Intelligente agenter kan interagere med andre agenter (og eventuelt mennesker) med henblik på at opfylde deres målsætning

Disse egenskaber skal opfyldes af systemet. Den overordnede målsætning for systemet vil være at fange byttet, se kapitel 2 for definition af en tilfangetagelse. Det overvejes derfor hvordan agenterne kan opfylde denne målsætning. I [Stone & Veloso, 1997] foreslås et simpelt scenarie, hvor et antal homogene agenter opererer uden indbyrdes kommunikation. I dette scenarie vil agenterne rykke hen til den nærmeste position der kan omringe byttet (herefter benævnt fangeposition). Denne simple fremgangsmåde vil gøre det muligt at fange byttet, men vil samtidig medføre problemer idet agenterne ikke kommunikerer indbyrdes. F.eks. kan der opstå en situation hvor to agenter forsøger at rykke til samme position for at omringe byttet. For at undgå disse problemer udvides scenariet ved at lade agenterne kommunikere indbyrdes. Kommunikationen giver samtidig mulighed for at koordinere sensorinput imellem agenterne for at give et bedre billede af miljøet. For at kunne opfylde målsætningen om at fange byttet skal følgende derfor gælde.

- Hvis en agent ser et bytte rykkes til den nærmest mulige fangeposition.
- Agenterne kommunikerer indbyrdes således at,

1. to agenter aldrig forsøger at rykke til den samme position.
2. sensorinput kombineres. Hvis én agent ser et bytte, skal de agenter der ikke kan se byttet kunne resonere sig frem til en vej hen til byttet.

Dette beskriver en simpel fremgangsmåde, der forsøger at opfylde målsætningen: at fange byttet. Denne fremgangsmåde vil ligeledes give anledning til en koordinering imellem agenterne, som bliver betegnet som koordination igennem fælles intentioner [Wooldridge, 2002]. Princippet er at et hold af agenter kan udvise samarbejdende opførsel, dog uden egentlig at samarbejde aktivt om at nå målet.

Der findes flere modeller til at beskrive agenterne. Modellerne er generelt en beskrivelse af hvordan agenten ræsonnerer. F.eks. er to hyppigt anvendte modeller, reaktive agenter kontra deliberative agenter. Kort sammenfattet gælder følgende for de to forskellige agentmodeller.

Reaktive agenter En udelukkende reaktiv agent handler i nuet uden reference til dens historie og tidligere handlinger. Den baserer sine handlinger direkte på miljøets nuværende tilstand [Wooldridge, 2002].

Deliberative agenter En deliberativ agent opretholder en intern tilstand. Den opfører sig som om den tænker, ved at gennemsøge et opførselsrum, opretholde en intern tilstand, og forudsige effekten af dens handlinger [Stone & Veloso, 1997].

Det kan ofte være svært at adskille de to modeller, da en egentlig implementering ofte indeholder aspekter fra begge. Et højere abstraktions niveau tilbydes dog med den såkaldte Belief-Desire-Intention (BDI) model [Wooldridge, 2000]. Her defineres Beliefs (Opfattelser) som agentens opfattelse af miljøet. Opfattelsen behøver dog ikke nødvendigvis at stemme overens med den faktiske virkelighed. Desires (Ønsker) er de ting som agenten ønsker opfyldt. Intentions (Intentioner) er de intentioner om agenten har valgt at udføre, hvilket den vil gøre, indtil intentionen enten er opfyldt eller umulig at gennemføre [Laidlaw et al., 2007].

Ved at benytte denne model kan agentens ræsonnering beskrives forholdsvis simpelt udfra dens opfattelser, ønsker og intentioner.

4.4 Analyse

I dette afsnit vil sammenkoblingen med LEGO miljøet, agentens ræsonnering i jagt-domænet blive vha. BDI modellen blive analyseret. Derudover blev det i kapitel 3 argumenteret for at sensorikontrollen skulle ligge hos agenten, derfor vil sensorikontrollen også blive analyseret.

Sammenkobling med LEGO miljøet

For at kunne agere og kontrollere robotter i LEGO verdenen, skal agentstrukturen have en repræsentation af miljøet. Eftersom verdenen er et 4x4 gitter, er det oplagt at repræsentere den i programmet som en graf. Grafen defineres således at alle sidelængder er ens. Dette simplificerer beregningen af den korteste vej fra knudepunkt A til knudepunkt B [Laidlaw et al., 2007].

Opfattelser

Hver agent har sin egen opfattelse af verdenen. Denne opfattelse behøver ikke at være ens fra agent til agent, dog skal vise opfattelser gælde for alle agenter. Eftersom alle agenterne skal have kendskab til hinandens position kræver det en vis konsistens imellem opfattelserne. Dette håndteres ved at lade agenterne informere de andre agenter om sin position. Her er det vigtigt at kun én agent kommunikerer af gangen. På dette vis sikres gensidig udelukkelse mht. agentposition.

Ønsker

Alle agenter har de samme ønsker eftersom det er et homogent system, disse kan beskrives ved følgende.

Fang bytte Dette betyder at agenten har en opfattelse af hvor byttet er, og derfor ønsker den fange det.

Udforsk verdenen Dette betyder at agenten ikke har nogen opfattelse af hvor byttet er, og derfor ønsker den at bevæge sig rundt i verdenen for at finde byttet.

Intentioner

Ud fra de ovenstående ønsker, har agenten en række intentioner som vil forsøge at opfylde ønskerne.

Hvis ønsket er at **fange byttet** kan agenten enten forfølge byttet, eller blive stående, hvis den allerede står i en fangeposition.

Hvis ønsket er at **udforske verdenen**, vil agenten blot have intentionen at lede efter byttet.

Dette resulterer i tre intentioner *forfølg byttet, bliv stående og led efter byttet*.

Sensorkontrol

En egenskab ved ultralydssensoren bevirker at det skal sikres at kun én ultralydssensor opererer af gangen. Dette betyder at agenterne skal skiftes til at foretage en observation med sensoren. For at sikre at alle agenter får ret til at se benyttes en synkroniserings barriere [Andrews, 2000]. Denne sørger for at en agent der lige har benyttet sin sensor, først kan benytte den igen efter de andre agenter har benyttet deres. På denne måde sikres det at synkroniseringen er fair.

4.5 Design

I det følgende vil design overvejelser og valg under implementering af systemet blive gennemgået.

BDI agenter

Rammeløsningen i iMARS laboratoriet stiller allerede en eksisterende BDI implementering til rådighed. Det er derfor oplagt at benytte samme opbygning i dette projekt. Implementeringen følger grundlæggende Wooldridge's forslag til en agentløkke i [Wooldridge, 2000] side 38, som kan ses i figur 4.1.

Agentens ræsonnering og opførsel bestemmes af denne løkke. De enkelte funktioner vil blive beskrevet kort i det følgende, og hovedparten af dem er magen til dem beskrevet af Laidlaw et al.. For en mere uddybende beskrivelse se derfor [Wooldridge, 2000] og [Laidlaw et al., 2007].

$brf(B, \rho)$ Denne funktion reviderer agentens opfattelser. I dette system, kontrollerer agenten sin indbakke for beskeder fra andre agenter. Disse beskeder kan enten indeholde information om de andre agents placering, eller sensor information fra de andre agenter. Derudover benytter den de tilgængelige sensor informationer fra den selv, samt andre agenter til at forsøge at bestemme en position til byttet.

$options(B, I)$ Denne funktion bestemmer hvilket ønske agenten skal have. Hvis agenten har opfattet et bytte vil ønsket være at fange det, ellers ønsker agenten at udforske verdenen.

$filter(B, D, I)$ Formålet med denne funktion er at opdatere agentens intentioner og samtidig sætte en ønsket position for agenten. Denne ønskede position knytter sig til intentionen, og skal forstås som om at agenten udfører intentionen ved at bevæge sig til denne position. F.eks. hvis agenten har opfattet et bytte, vil den ønskede position for agenten være en fangeposition. Agentens intention vil derfor være at bevæge sig hen til den nærmeste, og hvis den allerede står i den ønskede position

```
1.
2.   $B := B_0$           /*  $B_0$  are initial beliefs */
3.   $I := I_0$           /*  $I_0$  are initial intentions */
4.  while true do
5.      get next percept  $\rho$ ;
6.       $B := brf(B, \rho)$ ;
7.       $D := options(B, I)$ ;
8.       $I := filter(B, D, I)$ ;
9.       $\pi := plan(B, I)$ ;
10.     while not ( $empty(\pi)$  or  $succeded(I, B)$  or  $impossible(I, B)$ ) do
11.          $\alpha := hd(\pi)$ ;
12.          $execute(\alpha)$ ;
13.          $\pi := tail(\pi)$ ;
14.         get next percept  $\rho$ ;
15.          $B := brf(B, \rho)$ ;
16.         if  $reconsider(I, B)$  then
17.              $D := options(B, I)$ ;
18.              $I := filter(B, D, I)$ ;
19.         end-if
20.         if not  $sount(\pi, I, B)$  then
21.              $\pi := plan(B, I)$ ;
22.         end-if
23.     end-while
24. end-while
```

Figur 4.1: Agentløkke pseudokode

vil dens intention være at blive stående. Har agenten derimod ikke opfattet et bytte vil agentens intention være at bevæge sig til den nærmest udforskede position.

plan(B, I) Funktionen udfærdiger en plan for at fuldføre ens intention. I princippet er dette en liste af positioner som repræsenterer en sti hen til den ønskede position.

empty(π) Returner sandt hvis planen er tom.

succeeded(I, B) Afgør om intentionerne er fuldført.

impossible(I, B) Denne funktion afgør om en intention er umulig. I dette system er en intention umulig hvis agenten ikke kan bevæge sig til den næste position i sin plan. F.eks. hvis en anden agent befinder sig på positionen.

execute(π) Formålet er at udføre planen et skridt af gangen. Dvs., at for hvert skridt styrer funktionen robotten således, at den rykker fra en position til den næste.

Kommunikation i mellem agenterne

For at sikre en konsistent informations udveksling imellem agenterne, er det valgt at agenterne kun kan kommunikere én agent af gangen. Dette princip er i forvejen implementeret i rammeløsningen, og benyttes derfor videre igennem denne rapport. Laidlaw et al. beskriver hvordan en semafor tilgang til problemet, sikrer at kun én agent kun for lov til at kommunikere af gangen [Laidlaw et al., 2007].

Ved kommunikationen imellem agenterne benyttes kommunikations sproget FIPA [Wooldridge, 2002]. I FIPA sproget, sendes beskeder imellem agenterne. Disse beskeder omkapsler et performativ som beskriver beskedens indhold. Eksempler på performativer er

inform, der benyttes til at videregive information.

propose, der benyttes til forhandling.

agree, der benyttes til at udføre en handling.

failure, der benyttes til fejlhåndtering.

Ud af disse benyttes kun **inform** i dette system, da kommunikationen kun har til opgave at videregive information. I et mere avanceret system, kunne alle disse performativer blive brugt.

I praksis foregår kommunikationen imellem agenterne igennem en delt adgang til verdenen.

Sensorkontrol

Agentens sensorinput bliver kontrolleret af funktionen *see*. Her sørges det for, at en agent først ber om retten til at se. Herefter venter agenten indtil den har fået retten, hvorefter ultralydssensoren foretager fem målinger (for at sikre mod enkelte fejlmålinger). Gennemsnitsafstanden beregnes og observationen kommunikerer ud til de andre agenter. For at give de andre agenter mulighed for selv at ræsonnere over observationen videresendes tilstrækkeligt med oplysninger.

- Tidspunkt
- Relativ placering i forhold til sidste position
- Bevægelsesretning
- Agent ID
- Målt afstand

Tabel 4.1: Information til andre agenter om observation

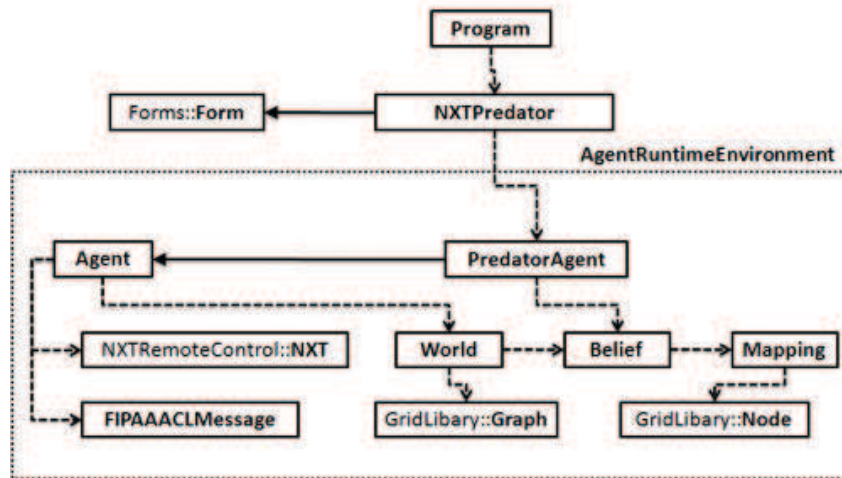
4.6 Implementering

Dette afsnit vil behandle implementeringen af multiagent systemet. Som tidligere nævnt benyttes der en eksisterende rammeløsning der tilbyder en to færdige hjælpebiblioteker, *NXTRemoteControl* og *GridLibary*. Formålet med *NXTRemoteControl* er at håndtere kommunikationen imellem computeren og robotterne. *GridLibary* indeholder klasser og funktioner der kan benyttes til at instantiere en gitterverden som beskrevet. Ligeledes findes der i dette bibliotek en funktion til at beregne den korteste vej i verdenen vha. A* algoritmen. For en dyberegående gennemgang af hjælpebibliotekerne henvises til [Laidlaw et al., 2007].

Multiagent systemet

Strukturen af det implementerede multiagent system kan ses i figur 4.2. De stiplede pile fra en klasse A til en klasse B, viser at en klasse A indeholder en instans af klasse B. En solid pile fra en klasse A til en klasse B, indikerer at klasse A udvider klasse B.

NXTPredator klassen er den grafiske brugerflade til systemet. Denne opretter en tråd for hver jægeragent (*PredatorAgent*), og viser en logning af agenternes ræsonnering. Den opretter samtidig en instans af klassen *World*, som deles imellem agenterne. Der er defineret en basis klasse for en agent. Denne indeholder funktioner til at kommunikere med robotten, samt referencen til det delte *World* objekt. *World* objektet repræsenterer verdenen og benyttes af agenterne til at kommunikere indbyrdes. Klassen *World* tilbyder derfor metoden *informWorld* som sender en multicast besked ud til de alle agenter i systemet. Derudover er det denne klasse der håndterer rettigheden til at kommunikere, samt synkroniseringen i forbindelse med



Figur 4.2: UML diagram der viser strukturen af NXTPredator multiagent systemet.

ultralydsmålinger. Det kan sikres at kun én agent taler af gangen, ved at kalde metoden **speak** før en besked sendes og **endspeak** efter endt kommunikation. Disse to metoder implementer en semafor således at kun en agent kan fuldføre **speak** metoden af gangen. Ligeledes er der to metoder **see** og **endsee** der sørger for at kun én ultralydssensor måler af gangen. For at sikre at alle agenter for lige ret til at benytte ultralydssensoren benyttes metoden **hasPermissionToSee**. Denne metode er implementeret ved en monitor, der opretholder en liste over de agenter der har benyttet sensoren. Hvis en agent ikke er på listen, giver metoden agenten ret til at se. Er agenten derimod på listen, venter agenten indtil den igen har ret til at se. Når alle agenter har set, nulstilles listen og alle ventende agenter vækkes.

Belief, repræsenterer robottens opfattelser. Derved også hvor robotten er placeret i verdenen, samt dens ønskede position. **Mapping** instansen benyttes til at give enhver position i agentens egen opfattelse af verdenen, en særlig status. Denne status kan være én af følgende.

Unknown Agenten har ikke nogen kendskab til positionens status.

Clear Agenten har en opfattelse af at den pågældende position er tom.

Object Agenten kan se et objekt, højst sandsynligt et bytte, ved den pågældende position.

NXT Positionen indeholder en robot. Denne status skal være konsistent imellem alle agenterne. Dette sikres ved det gennemgåede kommunikationsprincip.

Capture Positionen er en fangeposition ifht. byttets nuværende placering.

Prey Byttets placering.

Visited Positionen har været besøgt af agenten for nylig.

Disse statusmarkeringer benyttes herefter i agentens BDI implementering til at ræsonnere sig frem til en plan.

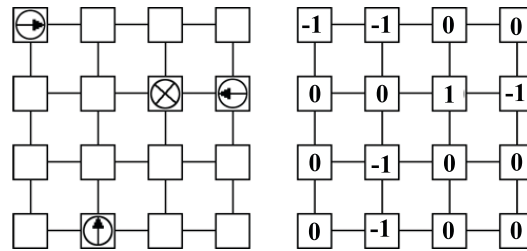
PredatorAgent indeholder implementeringen af BDI modellen. Denne følger fuldstændigt overvejelserne beskrevet i forrige afsnit og vil derfor ikke blive uddybet yderligere. Metoden **see** udfører målingen med sensoren og meddeler de andre agenter om dens observation ved at sende en FIPA meddelelse hvor et **Observation** objekt indeholdende de relevante informationer er indeholdt. I metoden **updateBeliefsFromInbox** opdateres agentens opfattelser ud fra dens modtagne beskeder. Her opfanges ligeledes de andre agenter observationer og gemmes lokalt. Når agentens opfattelser derefter skal revurderes, benyttes metoden **locateTarget** til at bestemme en sandsynlig bytte position. Fremgangsmåden er at der oprettes en matrice hvor hver position i verden er repræsenteret. Enhver position får som udgangspunkt værdien 0. Herefter benyttes observationerne til at vægte hver position positivt, hvis det er sandsynligt at byttet befinder sig på positionen, eller negativt, hvis det er usandsynligt at byttet befinder sig på positionen. Agentens opfattelser og observationer vægtes således.

- Hvis en agent selv kan se et bytte foran sig, lægges 1 til positionens vægtning.
- Hvis agenten selv har en opfattelse af at positionen foran sig er tom, trækkes 1 fra positionens vægtning.
- Hvis en anden agent kan se et bytte foran sig, lægges 1 til positionens vægtning.
- Hvis en anden agent har en opfattelse af at positionen foran sig er tom, trækkes 1 fra positionens vægtning.
- Hvis et objekt er observeret af en agent, lægges 2 til positionens vægtning.
- Hvis en anden agent er på positionen trækkes 1 fra positionens vægtning.

Til sidst bestemmer agenten den position med den højeste værdi, og hvis der findes en position med en maksimum værdi højere end nul, tror agenten på at byttet befinder sig ved denne position. Et eksempel på denne fremgangsmåde kan ses i figur 4.3, hvor der til venstre er vist en situation i jagtdomænet. Til højre ses hvordan positionerne vægtes og hvordan byttet kan findes i positionen med maksimums vægten 1.

4.6.1 Bytteagent

Der er i denne rapport ikke blevet gået i detaljer med bytte agenten. Dette skyldes at denne skal styres af et menneske. Den fysiske bytterobot bygger dog på en



Figur 4.3: Vægtning af en situation i jagtdomænet

modificeret jægerrobot. Dvs. at den har det samme bevægelseslag, men mangler ultralydssensoren, da denne ikke er nødvendig. For at kontrollere robotten implementeres derfor en simpel brugergrænseflade, hvor der kan sendes beskeder til robotten, således at den kan udføre én af de tilgængelige kommandoer i bevægelseslaget.

4.7 Test

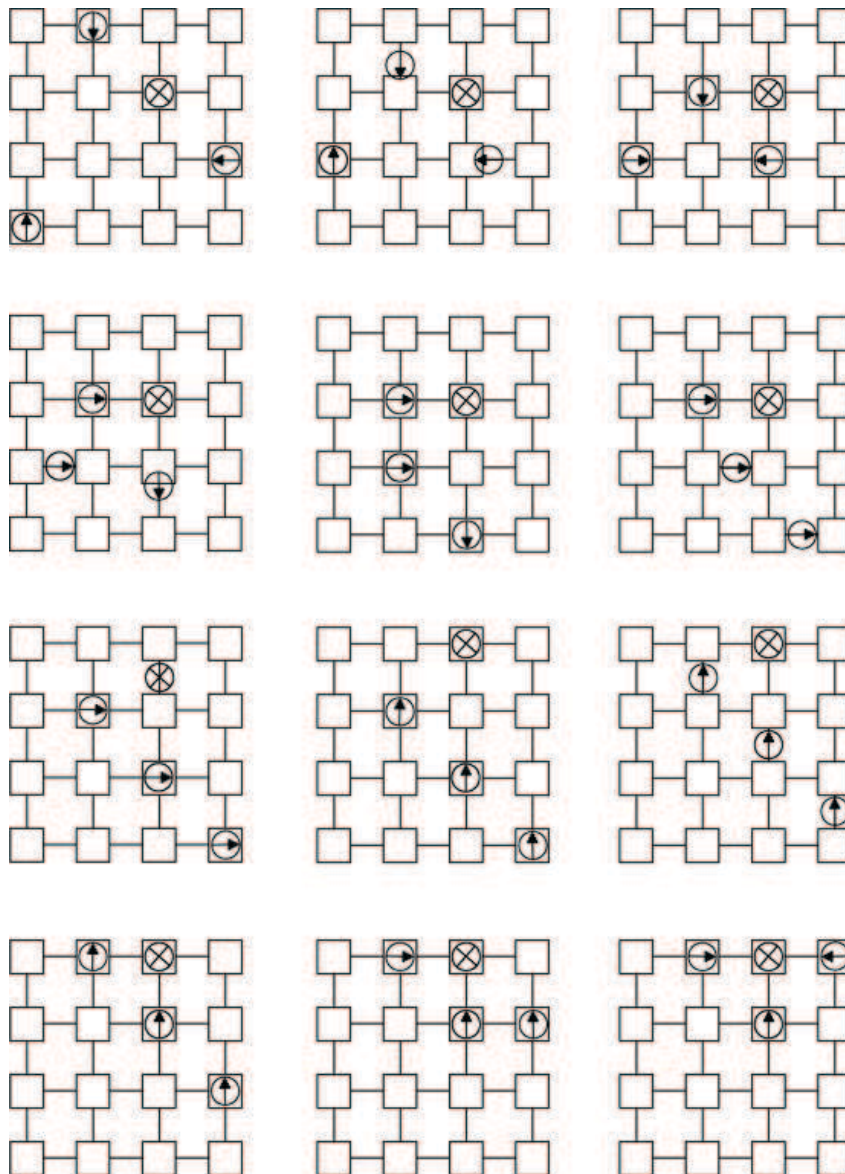
Systemet er blevet testet ved at udføre en funktionel test. Her er det blevet testet om systemet er i stand til at fange et bytte.

Til dette benyttes tre jægeragenter og et bytte. Jægeragenterne placeres på vilkårlige positioner, hvor de som udgangspunkt ikke kan se byttet. Forløbet er forsøgt beskrevet i figur .

Det ses, hvordan jægeragenterne begynder at lede efter byttet. I begyndelsen, er byttet udenfor synsvidde, så agenterne forsøger derfor at finde byttet ved at udforske verdenen. Dernæst får én agent øje på byttet, og ved at kombinere observationerne, får de resterende agenter en opfattelse af, hvor byttet befinder sig. Det ses herefter at de ændrer deres intention om at finde byttet, til i stedet at rykke hen til deres nærmeste fange position. Hernæst forsøger byttet at flygte til en anden position. Agenternes placering resulterer i, at en anden agent får øje på byttet, og igen koordineres observationerne. Ved at rykke til de nærmeste fange positioner kan byttet nu ikke længere flygte, og er derfor fanget.

Andre test er blevet udført og det er tydeligt at jægeragenterne kun har succes hvis byttet ikke er alt for aktiv i sin flugt. Dette vil sige at byttet ofte skal blive på sin position, så jægerne har en mulighed for at opfatte det. Samtidig er det ikke altid sikkert at jægerne kan se byttet efter det er rykket til en ny flugt position. Dette bevirker, at jægerne i realiteten kan begynde at lede ved uhensigtsmæssige positioner i relation til den ønskede tilfangetagelse.

Igennem alle testene ses det at systemet ikke er specielt reaktivt. Agenterne bevæger sig langsomt, og det ses tydeligt når agenterne venter på ret til at bruge ultralydssensoren.



Figur 4.4: Trinvis gennemgang af et jagtforløb. Figuren læses fra venstre mod højre, dernæst successt fra top til bund.

4.8 Diskussion

Testen viser at systemet virker som forventet. Det er muligt for jægerne at fange et bytte. Dog viser testen samtidig at systemet kan forbedres. Problemet med at jægerne ikke altid finder byttet efter det er flygtet, skyldes at de forsøger at finde byttet ved én af de fire flugtpositioner. Hvis jægerne ikke har noget bytte, husker de kun byttets tidligere position i et ryk. Dvs. at hvis de rykker hen til en af de fire flugtpositioner og stadig ikke kan se byttet, begynder de at lede efter byttet i hele verdenen. På denne vis kan byttet være heldig at igen befinde sig uden for jægerens synsvinkel. En åbenlys forbedring vil være at jægerne i fællesskab ræsonnerer sig frem til byttets flugt position, og på dette vis bestemmer den næste bedste fangeposition.

Et andet problem er at jægerne ikke tager højde for de andres afstand til fangepositionerne. Dette vil sige at i realiteten kan to jægere ønske at fange byttet ved den samme position. Herefter vil den først ankomende jæger få retten til at placere sig på positionen, og den anden vil da skulle revurdere sin plan til en ny fangeposition. Dette kan medføre nogen unødvendige ryk, men kan også simpelt forbedres ved at lade jægerne forhandle om fangepositionerne. Et forhandlingsprincip kunne være at den jæger der er tættest på en fange position får retten til det.

Med hensyn til reaktiviteten af systemet, skyldes problemet især de fysiske egenskaber ved ultralydssensoren. For det første kræver ultralydssensoren en vis pause i mellem målingerne for at sikre at eventuelle ekkoer forsvinder. Denne pause er dog relativt lille, dvs. i omegnen af 60 ms. Men da der foretages 5 målinger hver gang en agent "ser", giver dette en lille forsinkelse. Ydermere har agenterne behov for ofte at benytte sensoren, men da kun én agent kan benytte den af gangen, betyder det at en agent kan risikere at skulle vente ca. 1/2 sekund, hver gang den skal benytte sensoren. En forbedring af systemet vil derfor være enten, at benytte en ultralydssensor, der kan fungerer samtidig med andre ultralydssensorer, eller evt. at kigge nærmere på infrarøde sensorer.

4.9 Konklusion

I dette afsnit er et multiagent system blevet beskrevet og implementeret ved B-DI modellen. Derudover er det vist systemet er i stand til at opfylde jagtdomæne målsætningen: at fange et bytte. Igennem test er det vist hvilke umiddelbare forbedringer der vil kunne foretages på systemet.

Konklusion

Det kan konkluderes at det har været muligt at designe og implementere et multi-agent system, der i realiteten er opbygget af flere lag. En jagtdomæne instans der er velegnet til DTU's iMARS miljø er defineret. Der er argumenteret for hvordan iMARS miljøet skal udvides og ændres for at opfylde kravene til systemet, ved bl.a. at ændre på verden og ændre på sensor opsætningen af robotterne. Ligeledes er der udarbejdet og implementeret en agentarkitektur efter BDI modellen, der er ved hjælp af den eksisterende rammeløsning udarbejdet af Laidlaw et al., er i stand til at løse jagtdomæne målsætningen om at: fange et bytte.

Afslutningsvis har en test af systemet vist at på trods af at målsætningen er opfyldt er der mulighed for forbedringer.

- Koordinering imellem agenterne for at forfølge byttet bedre. Dette kan løses ved at sørge for at agenterne kommunikerer sammen om hvilke positioner de overvejer at fange byttet i.
- Implementering af en forhandlingsmekanisme der sørger for at hver agent ønsker en unik fangeposition. Dette kunne gøres vha. en auktion f.eks. hvor agenten med den korteste afstand vinder auktionen.
- Forbedring af ultralydssensoren. Enten ved at benytte en anden type ultralydssensoren, eller f.eks. erstatte den med en infrarød sensor. Ved en infrarød sensor vil det dog kræve at bytte agenten kan sende et infrarødt signal ud i alle retninger.

Udover de nævnte forbedringer, kunne det ligeledes være interessant at undersøge hvorledes et heterogent system vil kunne føre til en bedre løsning af målsætningen om at: fange et bytte.

Litteratur

- [Andrews, 2000] Andrews, G. R. (2000). Foundations of Multithreaded Parallel and Distributed Programming. Addison Wesley Longman Inc.
- [Laidlaw et al., 2007] Laidlaw, J., Zilmer-Pedersen, L. & Lemke, A. (2007). Developing Multi-Agent Lego Robotics. IMM, DTU.
- [Russel & Norvig, 2002] Russel, S. & Norvig, P. (2002). Artificial Intelligence: A Modern Approach. Prentice Hall.
- [Stone & Veloso, 1997] Stone, P. & Veloso, M. (1997). Multiagent Systems: A Survey from a Machine Learning Perspective. Carnegie Mellon University.
- [Wooldridge, 2000] Wooldridge, M. (2000). Reasoning About Rational Agents. The MIT Press.
- [Wooldridge, 2002] Wooldridge, M. (2002). An Introduction to MultiAgent Systems. Wiley.

Bevægelseslaget

I det følgende vil kommandoerne implementeret i bevægelseslaget blive gennemgået. Bevægelseslaget er i praksis et NXT-G program kaldet NXTPredator. Kildekoden kan ses i appendiks B.

followLine Kommandoen benyttes når en NXT skal køre ud fra et knudepunkt. Den vil starte med at køre frem indtil lyssen[soren er fri af knudepunktet, herefter forsøger NXT'en at holde sig inde imellem de 2 sorte kørestriber. Kommandoen har en timer på 1 sek., hvilket betyder at status = 3 returneres, med mindre NXT'en opfanger et knudepunkt hvorefter status = 2 returneres.

followLineNoClear Svarer til ovenstående kommando, dog uden først at rykke fri af knudepunktet.

followlineBackwards Kommandoen benyttes når en NXT skal bakke tilbage til et knudepunkt. Princippet er det samme som i followLine, dog foregår bevægelsen baglæns, og der er ikke nogen timer på bevægelsen. Status = 2 returneres derfor når knudepunktet er nået.

right NXT robotten rykker først fri af knudepunktet og drejer herefter et fastlagt antal grader til højere. Dernæst skulle det være sikret at agenten befinder sig på et sort underlag. Hvorefter agenten fortsætter med at dreje til højere indtil linjen opfanges af lyssensoren.

left Samme som ovenstående dog til venstre i stedet.

rightPrey Denne kommando benyttes når en robot skal dreje til højre, men blive ved knudepunktet. Den tidligere rutine for at dreje til højre benyttes, men

derefter følger robotten linjen baglæns indtil lyssensoren er placeret over knudepunktet. På dette vis sikres det at agenten herefter kan foretage et korrekt retningsskift.

leftPrey Samme som ovenstående dog til venstre i stedet.

turnAroundRight Kommando til at lave en drejning på 180 grader. Dette gøres ved at lave to på hinanden følgende højresving.

turnAroundLeft Samme som ovenstående dog til venstre i stedet.

BILAG B

Mindstorm NXT-G Program

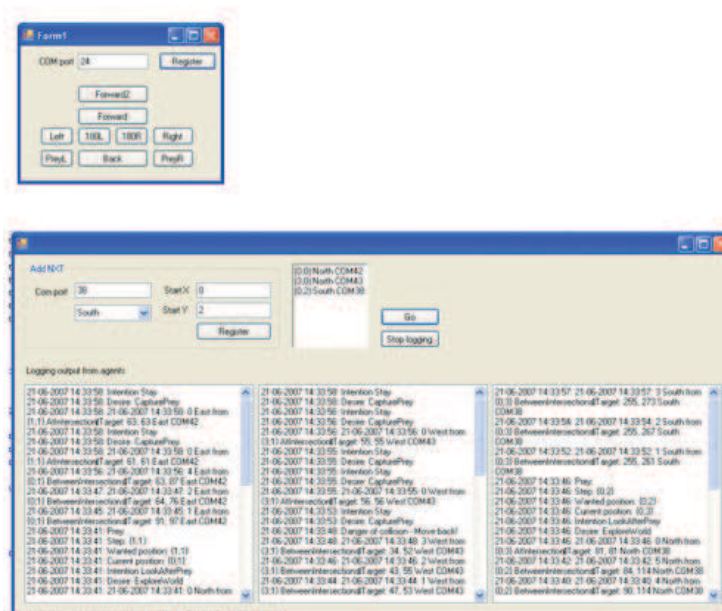
B.1 Kode

Koden er blevet printet direkte fra LEGO udviklingsmiløet og er vist på de følgende sider.

Agentimplentering

C.1 GUI

Der er udviklet to grafiske brugergrænseflader til kontrol af både multiagent system samt bytte agent. Et skærmdump af disse kan ses i figur C.1



Figur C.1: De grafiske brugergrænseflader

C.2 Kode

Kildekoden er blevet printet direkte fra Microsoft Visual Studio og vises på de følgende sider.