Software Engineering en Gedistribueerde Applicaties

Verslag

Team F.A.C.H.T.

Chris Bovenschen, Harm Dermois,
Alexandra Moraga Pizarro, Tamara Ockhuijsen en Fredo Tan

6104096, 0527963, 6129544, 6060374, 6132421

Universiteit van Amsterdam

22 juni 2011

Inhoudsopgave

1 Inleiding

Dit verslag beschrijft het ontwerpen, implementeren en testen van een gedistribueerde robotapplicatie. De robot die hiervoor wordt gebruikt is de gesimuleerde P2DX, deze heeft twee aangedreven wielen en een zwenkwiel. Het platform voor de robotsimulatie is USARSim. Het doel is om met behulp van de sensoren op de robot in een vrij volle ruimte een pad langs een wand te rijden, botsingen te vermijden, zonodig een wand op te zoeken, uit doodlopende stukken te ontsnappen, bijhouden waar hij is en welk pad hij heeft gevolgd en de omgeving in kaart te brengen.

Het project is op te delen in verschillende fases. Aan het begin is er nagedacht over een ontwerp met een model, een planning en een specificatie van het eindproduct. Hieruit is een voorlopig werkplan ontstaan. In het definitieve werkplan is een uitgebreidere beschrijving van de realisatie van verschillende componenten met bijbehorende tests te vinden samen met de eerste inleverbare versie van de code. Het grootste deel van de tijd heeft gezeten in de implementatie en tests. Als laatst zijn de componenten gentegreerd tot een geheel, getest en geoptimaliseerd.

Dit verslag behandelt eerst de implementatie van de componenten met de bijbehorende tests en de opzet van het experiment. Daarna worden de resultaten besproken en hieruit volgt een conclusie.

2 Implementatie

De modules zijn verdeeld in low-level, sensor, mid-level en high-level componenten. De listener is low-level, de sonar, laser en odometry zijn sensormodules, de wall search, wall follow en collision avoider zijn mid-level en de map maker en path finding zijn high-level. De movements en communicator zijn bibliotheken die door andere modules worden gebruikt.

De listener is verbonden met de server via een TCP-verbinding en stuurt geldige data door naar de sensormodules. De sensormodules halen de juiste waarden uit de data en sturen deze door naar de mid-level modules en op aanvraag naar de map maker en path finding. Berekende kaarten van de map maker kunnen worden gebruikt voor de path finding. Path finding gebruikt de movements bibliotheek om het pad te kunnen volgen.

2.1 Low-level

De low-level component verkrijgt rechtstreeks data van de robot en stuurt de data door naar de juiste module.

2.1.1 Listener

De listener zit direct aan de server vast. Hij haalt informatie binnen die de robot verstuurt en stuurt ook commando's naar de robot toe. De binnenkomende data worden gecontroleerd op geldigheid en in het juiste formaat gestuurd naar de bijbehorende sensormodules.

2.2 Sensormodules

De sensormodules ontvangen data van de listener. Ze verkrijgen de sensorwaarden uit de data en sturen ze door naar de mid-level modules. De sensormodules kunnen door de high-level modules worden aangeroepen voor de meest recente data. Zodra er ongeldige data binnenkomen, wordt de robot gestopt.

2.2.1 Odometry

Odometry haalt de drie waarden op die nodig zijn voor de bepaling van de positie van de robot. Dit zijn de x, y positie en de theta hoek in verhouding tot de startpositie en de richting van de robot. Deze waarden worden doorgestuurd naar de mid-level modules.

2.2.2 Range scanner

De range scanner leest alle 181 laserwaarden uit en stuurt ze door naar de mid-level modules.

2.2.3 Sonar

De sonar leest alle 8 sonarwaarden uit en stuurt ze door naar de mid-level modules.

2.3 Mid-level

De mid-level componenten verzorgen het vermijden van botsingen en het zoeken en volgen van een muur.

2.3.1 Collision avoider

De collision avoider kijkt alleen naar de waarden van de sonar, omdat de sonar wel naar de grond staat gericht en de laser niet. Als de kleinste waarde van de sonar kleiner is dan 0,315 stopt de robot met rijden. Tevens wordt er met behulp van de snelheid uitgerekend na hoeveel seconden de botsing plaats zal vinden als de robot door blijft rijden.

2.3.2 Wall follow

De muur wordt gevolgd die door wall search is gevonden. **???Als de muur zich aan de linkerkant van de robot bevindt, draait hij naar rechtss en als de muur zich aan de rechterkant van de robot bevindt, draait hij naar links.???** De kleinste laserwaarde bevindt zich nu aan de rechter- of linkerkant. Hij blijft rechtdoor rijden als de kleinste laserwaarde niet kleiner wordt dan een ingestelde waarde. Als hij te ver van de muur afwijkt, stuurt hij bij richting de muur. Als hij te dicht bij de muur komt, stuurt hij bij van de muur af. Als er geen muur meer wordt waargenomen, zoekt wall search de nieuwe dichtstbijzijnde muur.

2.3.3 Wall search

Bij het zoeken naar een muur draait de robot eerst 360 graden om te kijken waar de dichtstbijzijnde muur is. Zodra de kleinste sensorwaarde is gevonden, roteert de robot nog een keer totdat hij in de juiste positie staat. Hij rijdt recht op de muur af totdat hij op een bepaalde afstand van de muur af staat en dan gaat hij over op wall follow.

2.4 High-level

De high-level componenten houden bij waar de robot is geweest en kunnen hem naar een ander punt laten rijden.

2.4.1 Map maker

De map maker maakt een kaart van de constante stroom van laser en odometry waarden die hij binnenkrijgt. Voor het maken van een kaart wordt de Simultaneous Localization And Mapping (SLAM) techniek gebruikt. De kaart heeft de vorm van een matrix.

2.4.2 Path finding

Path finding wordt alleen gebruikt als er al een kaart is gemaakt en kan dus alleen een route plannen binnen een bekend domein. Hierin probeert hij een zo kort mogelijk pad te vinden naar de bestemming. Er wordt gebruik gemaakt van het A ster zoekalgoritme om het efficintste pad te vinden tussen twee punten.

2.5 Libraries

Bibliotheken

2.5.1 Communicator

De communicator zorgt voor de communicatie tussen de modules.

2.5.2 Movements

Dit is een bibliotheek waarin alle bewegingen zijn gedefinieerd. Deze bibliotheek wordt gebruikt voor de modules die de robot willen besturen.

3 Tests

Voor elke module is er een aparte test geschreven om er zeker van te zijn dat elke module op zichzelf goed werkt. Elke module wordt direct aan de simulator vastgezet en getest met de directe data die binnenkomen.

3.1 Wall combo

De wall combo is geschreven om te testen of de wall search goed werkt in samenwerking met de wall follow.

4 Experiment

De robot wordt genitialiseerd op een bepaalde plaats. Hij zoekt een muur en blijft deze muur volgen. Zodra hij de opdracht krijgt om naar een punt toe te rijden, wordt path finding ingeschakeld. De map maker werkt continu de kaart bij en de collision avoider zorgt ervoor dat er geen botsingen plaatsvinden. Als hij geen nieuwe opdracht krijgt om naar een ander punt toe te rijden, gaat hij weer een muur zoeken en deze volgen.