

Universitat Politècnica de Catalunya
Facultat d'informàtica de Barcelona

Plataforma per al desenvolupament d'aplicacions en l'àmbit de la robòtica mòbil

GUILLEM SERNA CLARA

PROJECTE DE FINAL DE CARRERA EN ENGINYERIA INFORMÀTICA
DIRIGIT PER JOSEP FERNÁNDEZ RUZAFÀ



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Departament d'Enginyeria de Sistemes,
Automàtica i Informàtica Industrial



OCTUBRE 2016

Informació del projecte

Títol: Plataforma per al desenvolupament d'aplicacions en l'àmbit de la robòtica mòbil

Autor: Guillem Serna Clara

Data: Octubre de 2016

Titulació: Grau en Enginyeria Informàtica

Crèdits: 18

Director del Projecte: Josep Fernández Ruzafa

Departament: Enginyeria de Sistemes, Automàtica i Informàtica Industrial

Tribunal

President: **PERE BARLET ROS**

Firma:

Vocal 1: **JORDI BORONAT MEDICO**

Firma:

Vocal 2: **ALICIA CASALS GELPI**

Firma:

Vocal Suplent: **JOAN MANUEL PARCERISA BUNDO**

Firma:

Qualificació

Qualificació numèrica:

Qualificació descriptiva:

Data:

Resum

CATALÀ

Aquest projecte conté el desenvolupament de la configuració d'un simulador per fer estudi de la robòtica mòbil. Tracta l'utilització de l'API de GazeboSim, un simulador de físiques de robòtica de gran qualitat.

Els alumnes de l'assignatura utilitzen uns robots de neteja reprogramats per tal de rebre comandes des d'un ordinador. Al laboratori hi ha un nombre limitat de robots, i aquests no sempre funcionen o estan ocupats. L'objectiu és replicar l'entorn de les pràctiques de laboratori a un entorn de simulador.

Per tal de millorar la interfície, en el projecte es desenvolupen uns plugins pel simulador, que recreen la interfície del robot a la perfecció. D'aquesta manera es poden provar els programes creats amb l'ordinador al simulador, només canviant el port de connexió del programa. Els alumnes podran realitzar proves dels seus codis sense haver de disposar del robot real.

ESPAÑOL

Este proyecto contiene los desarrollo de la configuración de un simulador para hacer estudio de la robótica móvil. Trata la utilización de la API de GazeboSim, un simulador de físicas de robótica de gran calidad.

Los alumnos de la asignatura utilizan unos robots de limpieza reprogramados para recibir comandos desde un ordenador. En el laboratorio hay un número limitado de robots, y éstos no siempre funcionan o están ocupados. El objetivo es replicar el entorno de las prácticas de laboratorio en un entorno de simulador.

Para mejorar la interfaz, en el proyecto se desarrollan unos plugins para el simulador, que recrean la interfaz del robot a la perfección. De esta manera se pueden probar los programas creados con el ordenador al simulador, sólo cambiando el puerto de conexión del programa. los alumnos podrán realizar pruebas de sus códigos sin tener que disponer del robot real.

ENGLISH

This project contains the development of the configuration of a simulator for study of mobile robotics. It explains the use of the API GazeboSim a quality physical robotic simulator.

Students of this subject are using cleaning robots reprogrammed to receive commands from a computer. In the laboratory there is a limited number of robots, and these do not always work. The aim is to replicate the lab environment to the simulator.

To improve the interface, on the project I developed a plugin for the simulator, recreating perfectly the interface of the robot. So the students can test the created programs with computer simulator, only changing the port connection from program. Students can perform tests of their code without having the real robot.

Agraïments

Índex

1. Gestió del projecte	7
1.1 Abast i contextualització	7
1.1.1 Context	7
1.1.2. Actors implicats	8
1.1.3. Estat de l'art	9
1.1.4. Formulació del problema	12
1.1.5. Abast	13
1.1.6. Metodologia i rigor	14
1.2. PLANIFICACIÓ TEMPORAL	15
1.2.1 Fases	15
Repàs Nocions de Robòtica	15
Primer contacte amb el robot	15
Familiarització amb l'entorn del simulador Gazebo	15
Creació d'un disseny del Neato per el simulador Gazebo	15
Creació pluguin del Socket de connexió per al simulador Gazebo	15
Creació pluguin de les funcions bàsiques per simulador Gazebo	16
Estudi error típic	16
Millorar simulació del neato a Gazebo	16
Disseny 3D d'entorn de treball	16
1.2.2. Diagrama de GANTT	17
1.3. Identificació dels costos	17
1.3.1. Pressupost estimat	17
1.3.1.1. Cost de material	17
1.3.1.2. Costos de personal	18
1.3.1.3. Costos indirectes	18
1.3.1.4. Cost total	19
1.3.2. Cost de replica d'un robot	19
1.3.3. Cost reduït	20
1.3.4. Amortització	20
1.4. Sostenibilitat	21
1.4.1. Econòmica	21
1.4.2. Social	21
1.4.3. Ambiental	22
1.4.4. Matriu de sostenibilitat	24
1.5. Lleis i regulacions	24
2. Objectius	25
2.1. A curt termini	25
2.2. A llarg termini	25

3. Introducció	26
3.1. Neato XV essential	26
3.1.1. Introducció	26
3.1.2. Sensors	26
3.1.3. Equipament i tecnologia	27
3.1.4. Connexió i modificació	28
3.2. Detecció de l'entorn	29
3.3. Estudi error típic	30
4. Desenvolupament	33
4.1. Desenvolupament del Simulador	33
4.1.1. Introducció	33
4.1.2. Creació del model del robot	33
4.1.3. Creació plugin connexió Simulador	35
4.1.4. Creació de funcions bàsiques del Simulador	35
4.1.5. Ús del pluguin	37
4.1.6. Ús de les comandes	37
4.1.6.1. SetMotor	37
4.1.6.2. GetMotors	37
4.1.6.3. GetLDSScan	38
4.1.7 Exemple d'utilització	38
4.2. Desenvolupament de l'entorn 3D	41
5. Conclusions	46
5.1. Objectius aconseguits	46
5.2. Funcionalitats	46
5.3. Desenvolupament de l'entorn	46
5.4. Valoració personal	46
6. Apèndix	47
6.1. Definicions	47
6.2 Users manual simulador	48
6.3. Referències	52

1. Gestió del projecte

1.1 Abast i contextualització

1.1.1 Context

El projecte surt de la proposta del professor Josep Fernández, i la motivació personal de treballar en el camp de la robòtica. En l'assignatura de robòtica, junt amb el professor Antonio Martínez vaig descobrir la meva passió pels robots.

La idea de realitzar un projecte de robòtica mòbil ve del meu interès en els vehicles que es mouen. A més, he descobert una gran passió per la part de maquinària (*hardware*) de la informàtica. Poder unir dos temes per fer el treball de fi de carrera ha estat molt gratificant per part meva.

La motivació d'escollir aquest treball també prové de l'augment en la investigació sobre aquest camp, el qual ha permès que la robòtica s'incorpori poc a poc a la societat. En aquest moment els robots comercials més venuts a espanya¹ són els robots de neteja domèstica. En la indústria de la construcció els robots també són molt útils, ja siguin braços robòtics o robots mòbils. La majoria de fàbriques usen la robòtica mòbil per tal de transportar palers amb vehicles no tripulats per la zona de la indústria.

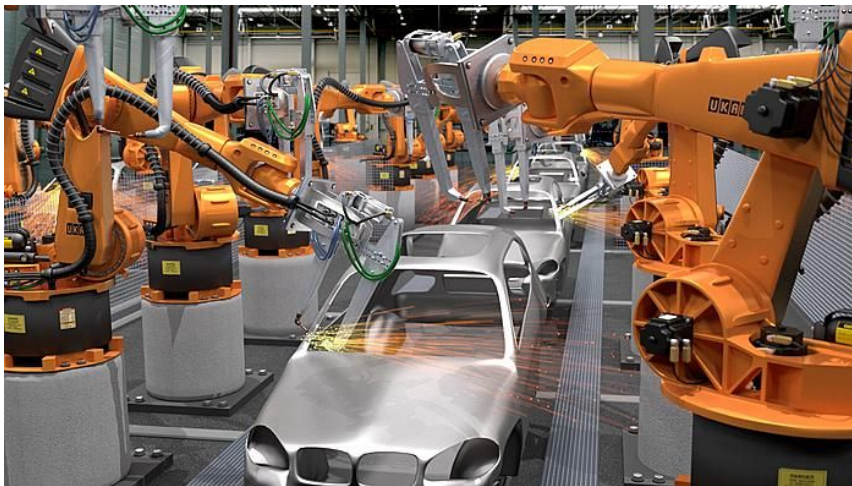


Figura 1: Imatge de robots articulats

Junt amb el professor Josep Fernández vam decidir anomenar el projecte “*Plataforma de desenvolupament en l'àmbit de la robòtica mòbil*”. Vam decidir que faria un estudi per tal de millorar la trajectòria d'un robot mòbil i reduir l'error que pugui obtenir.

Aquest projecte està dedicat principalment a la investigació. Com que està basat en una de les pràctiques en l'assignatura de Robòtica (ROB), els alumnes de l'assignatura podran usar

¹ Patricia Puentes. *eldiario.es*: *Los robots domésticos llegan a casa*. Barcelona, 19/01/2015. [Consulta: 2/03/2016]. Disponible a: http://www.eldiario.es/turing/casa-inteligente_0_347515619.html

els codis realitzats en aquest projecte. A més, per fer més fàcil la realització del projecte, es programarà un simulador per tal de poder treballar a casa, amb el que els alumnes de l'assignatura podran usar-lo sempre que ho necessitin.

1.1.2. Actors implicats

- **Professor Josep Fernández Ruzafa:** Director del projecte, podrà usar alguns dels codis realitzats per tal de preparar exercicis per a les pràctiques de l'assignatura robòtica.
- **Professor Antonio B. Martínez:** Coordinador de l'assignatura robòtica, utilitzarà el simulador Gazebo i els plugins creats del Neato XV-Essential per les seves classes i pràctiques de l'assignatura robòtica a les diferents facultats on treballa.
- **Alumnes de l'assignatura de robòtica de la UPC:** Usaran els codis fets per mi per tal de poder fer un estudi del robot més acurat, també podran utilitzar el simulador per fer les pràctiques a casa i no haver d'anar als laboratoris docents.
- **Alumnes d'altres universitats:** El simulador i la documentació seran de lliure distribució, i per tant altres universitats poden incloure-ho a assignatures de robòtica o cursos sense costos.
- **Usuaris que vulguin fer estudi sobre robòtica mòbil:** Es pretén que el codi sigui *Opensource*, així qualsevol persona a la xarxa podrà investigar i estudiar els moviments del Neato d'una forma gratuïta fent servir el simulador.

1.1.3. Estat de l'art

La robòtica mòbil és objecte d'investigació en l'actualitat a la majoria de universitats. Els robots han esdevingut totalment normals en el àmbit comercial i industrial². Poden proporcionar moltes facilitats a les persones. Els robots mòbils poden desplaçar-se en qualsevol tipus d'entorn i de detectar i estudiar els elements que hi ha dins aquest entorn. Necessiten una planificació per saber a on volen anar i per on han de desplaçar-se. Amb l'ajuda de diferents sensors poden arribar al seu destí o a realitzar la seva tasca.

L'evolució de la robòtica mòbil ha tingut diferents fases:

- **Manualment tele-operats** → Robots que els condueix totalment una persona amb un *joystick* o un control.
- **Tele-operat vigilat** → El robot segueix conduint-lo una persona remotament, però té la capacitat d'evitar xocar amb obstacles.
- **Line-following car** → Són robots que segueixen un circuit pintat per una línia de color. Són dels primers robots autònoms.
- **Robots autònoms aleatoris** → són robots que es mouen autonòment, però que la seva trajectòria va dirigida per els xocs que fan. Porten sensors de xoc (*bumpers*), i al xocar canvien de direcció.
- **Robots guiats de forma autònoma** → Aquests robots saben a quin entorn estan, i el seu objectiu és arribar a algun punt de l'entorn on es troben. Per guiar-se utilitzen diferents tipus de sensors, com làsers, visió, encoders, gps..


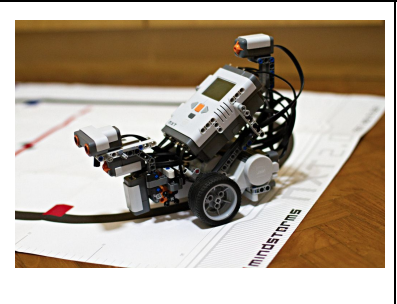
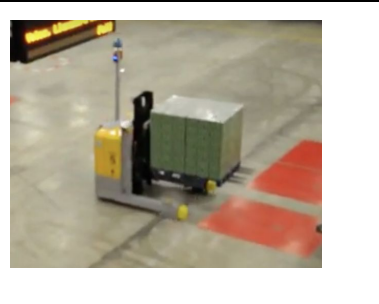
		
Manualment tele-operats	Line-following car	Robot guiat de forma autònoma

Figura 2: Exemples de diferents tipus de robots.

Els robots mòbils es troben desde la indústria a entorns domèstics. Els robots mòbils poden ser inclús joguines infantils o inclús poden realitzar tasques domèstiques senzilles, com els robots de neteja. En els entorns de la indústria, en concret s'utilitzen per fer transport dins de fàbriques automatitzades i en alguns hospitals i farmàcies, des de fa més de 5 anys, usen robots mòbils autònoms per tal de transportar medicaments.

² "Adaptive Manipulation of a Hybrid Mechanism Mobile Robot". P. Moubarak, P. Ben-Tzvi. *IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. Montreal, Canada, 2011, pp. 113 - 118

El robot de neteja Neato XV-Essential conté uns components de molt altes prestacions a un preu assequible. Aquest robot pot ser objecte d'estudi per la robòtica mòbil, ja que el seu sensor làser és molt usat en aquest tipus de màquines. Té unes mides de 31,8cm de llarg per 33cm d'ample. Utilitza dues rodes motrius alimentades per motors diferents, de manera que pot realitzar rotacions sobre si mateix. Aquestes rodes tenen uns encoders de 1mm de resolució. Per tal d'estar estable utilitza dues rodes lliures sense cap tipus de motor.



Figura 3: imatge del robot Neato XV Essential

El robot fa ús de 3 sensors per tal de navegar per la casa mentre neteja. El seu “paraxoc” frontal (*bumper*) és utilitzat per detectar colisions, amb 3 punts diferents (central i laterals) amb el que el robot sap per quina part està xocant. També utilitza un sensor de proximitat, situat únicament a la part dreta del robot, que és usat per saber a quina distància està de la paret mentre neteja i no passar varies vegades pel mateix lloc. L'últim sensor és el sensor làser de 360°, que permet escanejar tot el que tenim al voltant.

A més, el robot Neato consta de un acceleròmetre i un port de comunicació USB. En aquest port és on hem conectat la placa Raspberry Pi2 i per la qual li enviem les instruccions, com es pot observar a la figura 4.

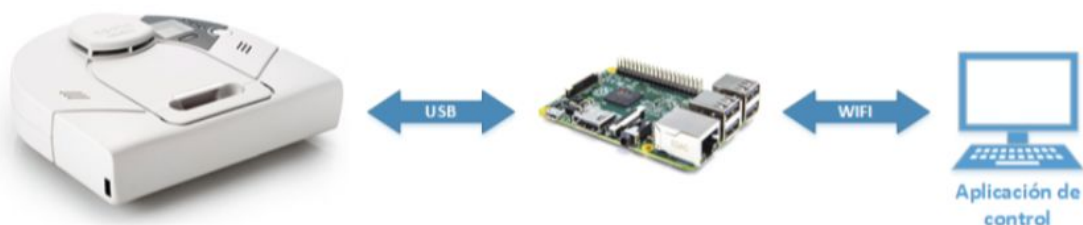


Figura 4: Connexió a través de la Raspberry Pi 2

El projecte es basa en millorar el moviment del robot, amb el que utilitzaré una interfície creada per a un treball de fi de grau, que permet connectar-se al robot. Actualment aquesta

interfície s'utilitza a l'assignatura de robòtica i el fan servir per fer diferents exercicis. Al robot Neato se li han extret totes les peces de neteja (corró, aspirador i bossa de brossa) i se li ha afegit un hardware específic. Consisteix en una Raspberry Pi2 amb un accés point connectat a un router. Mitjançant un script amb Matlab o amb el llenguatge Python es crea un socket de connexió amb el qual ens podem connectar al robot per donar-li instruccions i rebre dades dels sensors.

Per tal de fer aquesta comunicació s'usa un socket creat per un script en python o en Matlab. Mitjançant el socket, enviem les instruccions i rebem els valors demanats al robot. Principalment usem el programa Matlab, ja que permet utilitzar un entorn gràfic intuitiu i fàcil de fer servir. Des de Matlab es creen scripts donant indicacions al robot i la lògica del comportament que realitzarà segons els paràmetres que proporciona el Neato.

1.1.4. Formulació del problema

Els robots mòbils fàcilment obtenen errors de posició. Això és degut a que els moviments no sempre són exactament perfectes. Les rodes poden patinar o fer un mal contacte amb el terra. Una petita pedra pot fer desviar, lleugerament, la seva trajectòria fent que una roda realitzi més trajecte que una altra.

Aquestes dificultats es produeixen en freqüència, no només pel robot, sinó també per la superfície per on es desplaça. Si el robot va variant la seva trajectòria 1º a moviment, el error pot augmentar considerablement. A més aquest error en graus (º) augmenta en fer rotacions, ja que les acceleracions i velocitats de les rodes poden ser lleugerament diferents. En el cas de realitzar la trajectòria d'un quadrat, per exemple, es fa molt complicat que el robot arribi exactament al lloc de sortida.

Cal afegir que el robot creu que està a una posició, quan en realitat pot estar a una altra.. Ell realitza els moviments demanats, sense saber que els està realitzant erronis. Els seus *encoders* li diuen el moviment que ha fet realment, però no té en compte si les rodes patinen.

El làser que porten els robots són dels sensors més fiables del mercat, de manera que els podem usar amb una gran certesa de que no fallen. L'objectiu és fer servir la lectura del làser per poder arribar a fer un *mapping* del lloc on estem. D'aquesta manera podem interpretar *Landmarks* (punts de referència) i veure com evolucionen al llarg del recorregut. Els *landmarks* usables són cantonades, i vèrtex de l'entorn, que amb la lectura del làser permeten ser detectats amb exactitud. Després del moviment del robot, podem fer un càlcul de quina posició hauríem de veure aquests *landmarks*. Fent una nova lectura del làser podem veure si coincideixen i la magnitud d'error que tenim.

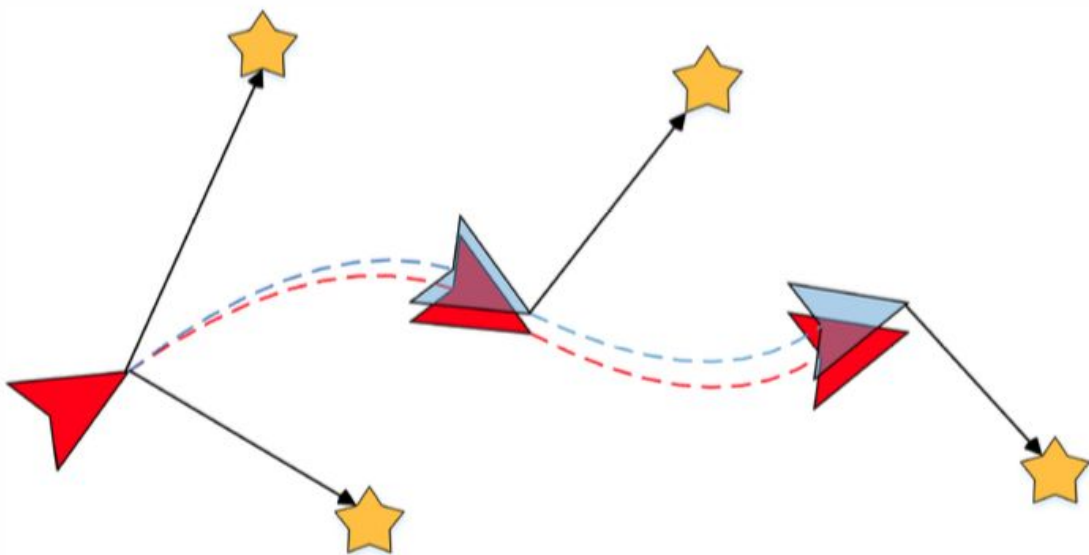


Figura 5: Exemple de detecció de landmarks.

Un robot mòbil ha de ser capaç de detectar els *Landmarks* (punts de referència) de una manera correcta i sense equivocació usant el làser. S'usen 2 tipus de *Landmarks*, els

artificials i els naturals. Els punts de referència artificials seran creats per nosaltres, com ara llaunes i cilindres grans, ubicats al centre de l'habitació. Aquests landmarks artificials són fàcils de detectar. En canvi quan ens referim a landmarks naturals cantonades de paret, columnes... són molt més difícils de detectar.

Un robot ha de saber realitzar un viatge d'un punt A a un punt B, i guiar-se pels landmarks existents i coneguts per tal de no seguir una trajectòria errònea.

L'entorn de pràctiques és una aula de la universitat. El departament consta de 5 robots Neato XV essential idèntics. Tots estan modificats per rebre les comandes d'un ordinador. Aquests robots solen fallar en freqüència, per problemes de bateria o problemes de connexió, sent necessari reiniciar el router. En època de pràctiques els robots estan molt sol·licitats pels alumnes. És possible que per provar un codi, l'alumne hagi d'esperar torn.

Per aquesta raó s'ha decidit desenvolupar un software per tal de fer simulacions amb el robot. Usant una API existent anomenada Gazebo hem creat una sèrie de pluguins per tal de simular el nostre robot Neato XV-Essential i el seu comportament. S'ha realitzat una sèrie de pluguins que permeten la connexió idèntica a la del robot, enviant les comandes desde Matlab i usant el socket de connexió. A més s'ha realitzat un estudi per tal de simular l'error dels Neato reals al model creat per Gazebo.

1.1.5. Abast

Amb aquest projecte pretenc arribar a detectar d'una manera òptima els *landmarks* d'una habitació o d'una ruta del robot mòbil, i ser capaç de reduir l'error de la ruta del robot. La detecció de *landmarks* no és fàcil, i requereix una algoritmia complicada.

Per tal que el robot pugui complir el seu objectiu cal realitzar un estudi de l'error típic que es comet a l'executar instruccions senzilles. Es realitzaran proves de moviment en trajectòria recta, rotacions, i inclús formes primitives com un quadrat per veure amb quina magnitud es desvia el robot.

Els 6 robots Neato cedits per la universitat són completament iguals i estan muntats d'una manera idèntica. Tot i així, estudiant els diferents robots s'ha vist que els errors típics (de distància i rotació) són diferents en la majoria d'ells. Això podria provocar que els algorismes fallassin en algun cas. Per això s'intentarà que l'algoritme pugui aprendre l'error que comet el robot on s'està executant i realitzar altres mecanismes d'adaptació.

Usar Matlab és un inconvenient important, ja que és un Software amb llicència. La universitat té una llicència pels estudiants, però si el projecte es traspasa a gent que no la té, no podran provar si funciona i ho hauran de traduir al llenguatge de Python que actualment és gratuït. També es podria intentar trobar algun software similar al Matlab però de programari lliure.

1.1.6. Metodologia i rigor

En aquest treball, he de realitzar un estudi empíric dels errors típics que té el robot “Neato” al seguir una sèrie d'instruccions senzilles, per tal d'intentar reduir al màxim l'error que el robot realitza de normal.

Després he de fer un estudi dels valors donats pel làser i anar millorant la detecció de *landmarks*, fins a aconseguir la detecció ideal. Per tal de fer el seguiment, es pot anar veient com augmenta la granularitat de la detecció dels punts de referència, i la millora poc a poc de la posició del robot.

Per poder veure els resultats, la prova més senzilla serà la de realitzar un recorregut de forma quadrada, i que el robot acabi arribant al mateix lloc. Per veure més avenços, podrem fer proves on el robot realitzi recorreguts molt més llargs i que per tant l'error tendeixi a augmentar molt més; l'objectiu serà fer passar el robot per llocs estrets sense xocar, ja que no ha obtingut error, a una gran distància de l'inici del seu recorregut.

Usaré la metodologia Agile, fent sprints setmanals i veient resultats a curt termini. He estudiat aquesta manera de fer i crec que m'anirà molt bé per al meu treball.

1.2. PLANIFICACIÓ TEMPORAL

1.2.1 Fases

Repàs Nocions de Robòtica

Temps estimat: 2 setmanes.

Dates: 15/02/2016 - 22/02/2016

Recursos: Apunts de l'assignatura i treball cedit pel professor.

Descripció: Fer un repàs exhaustiu dels apunts personals de l'assignatura robòtica i lectura del anterior treball de fi de grau realitzat amb un Neato.

Primer contacte amb el robot

Temps estimat: 5 setmanes.

Dates: 22/02/2016 - 29/02/2016

Recursos: Robot Neato i ordinadors del laboratori docent.

Descripció: Provar els diferents funcions del robot i els programes ja fets per tal comprendre'ls.

Familiarització amb l'entorn del simulador Gazebo

Temps estimat: 4 setmanes.

Dates: 29/02/2016 - 01/03/2016

Recursos: Programa Gazebo i tutorials oficials.

Descripció: Gazebo permet usar la seva API per crear i simular escenaris amb robots. El seu entorn és senzill, però la programació dels plugins dels robots no és fàcil. La pròpia web presenta uns tutorials bàsics per l'aprenentatge de les funcions de l'API. El simulador Gazebo serà molt útil per tal de provar els codis quan els robots estiguin sense bateria o no tinguin accés a ells.

Creació d'un disseny del Neato per el simulador Gazebo

Temps estimat: 2 setmanes.

Dates: 03/03/2016 - 07/03/2016

Recursos: FreeCad i Gazebo.

Descripció: El model del robot Neato per al simulador Gazebo es pot trobar fàcilment a internet, amb una versió molt ben detallada. Gazebo utilitza el model de "Col·lisió", les mides reals que té i que permeten calcular si el robot xoca o no; i el model "Artístic", que permet veure la forma, disseny i detalls del robot. Per tal que el simulador realitzi els *renders* a amb rapidesa, és necessari reduir el nombre de detalls del model artístic.

Creació plugin del Socket de connexió per al simulador Gazebo

Temps estimat: 4 setmanes.

Dates: 07/03/2016 - 14/03/2016

Recursos: Ordinador, simulador Gazebo i instruccions de l'API de Gazebo.

Descripció: El robot Neato i la Raspberry Pi 2 utilitzen una connexió wifi que transmet les dades via *Socket*. Aquest *Socket* es pot connectar amb el programa Matlab o amb un script amb Python, per tal de donar-li les ordres al robot. Amb Gazebo, volem que funcioni de la

mateixa manera, i que usant matlab i una connexió via *Socket* podem donar instruccions al robot.

Creació pluquin de les funcions bàsiques per simulador Gazebo

Temps estimat: 12 setmanes.

Dates: 14/03/2016 - 28/03/2016

Recursos: Ordinador, simulador Gazebo i instruccions de l'API de Gazebo.

Dependències: El pluquin del *Socket* per Gazebo ha d'estar acabat.

Descripció: Per tal de controlar el robot Neato, utilitzem bàsicament 3 de les funcions implementades pel fabricant: *SetMotors* (dóna l'ordre que els motors s'activin a una certa distància i a una velocitat concreta), *GetMotors* (retorna el valor dels encoders i la velocitat actual de les rodes) i *GetLDSScan* (retorna les 360 lectures dels 360° del làser i la velocitat de rotació). Aquestes funcions cal implementar-les a Gazebo i que retornin els valors igual que el robot.

Estudi error típic

Temps estimat: 1 setmana.

Dates: 14/03/2016 - 21/03/2016

Recursos: Robots Neato i ordinadors del laboratori docent.

Descripció: Els robots mòbils solen tenir un petit error al moure's, com desplaçar-se uns cm més a l'avançar i girar un grau més o menys al fer una rotació. Aquests errors imperceptibles per al robot provoquen que augmenti la diferència de posició real respecte la posició esperada del robot. Estudiant aquest error i veient si compleix patrons semblants podríem arribar a reduir-lo.

Millorar simulació del neato a Gazebo

Temps estimat: 6 setmanes.

Dates: 21/03/2016 - 28/03/2016

Recursos: Ordinador, simulador Gazebo i pluquins del robot Neato.

Dependències: Estudi de l'error típic i implementació dels pluquins per al simulador Gazebo.

Descripció: En el simulador Gazebo, l'error és nul; el robot es mou a la perfecció. Per tal de simular bé l'entorn real de treball és necessari aplicar l'error típic estudiat amb els robots reals. S'ha de trobar la manera de incloure tots els errors.

Disseny 3D d'entorn de treball

Temps estimat: 2 setmanes.

Dates: 28/03/2016 - 30/03/2016

Recursos: Ordinador, simulador Gazebo i FreeCad.

Descripció: Per tal de realitzar simulacions verídiques, és necessari recrear l'entorn de proves per usar-lo a Gazebo.

1.2.2. Diagrama de GANTT

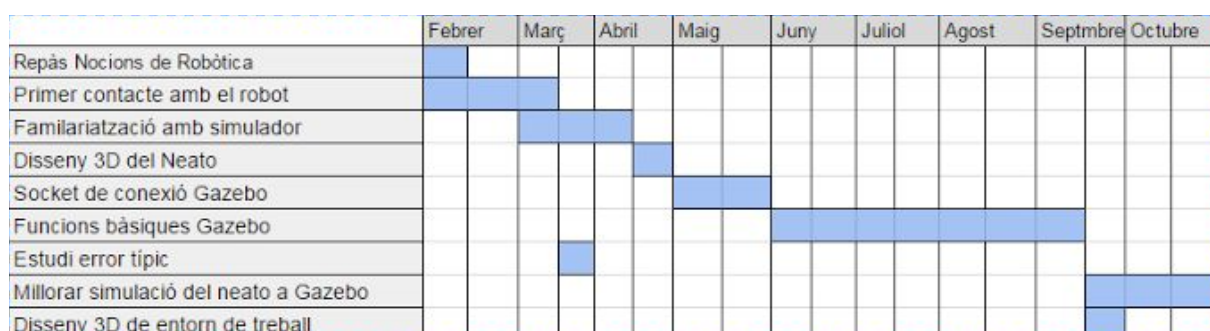


Figura 6: Planificació temporal en format diagrama de Gantt

1.3. Identificació dels costos

1.3.1. Pressupost estimat

1.3.1.1. Cost de material

Element	Cost [€]
Robot Neato XV- Essential	345.1 ³
Raspberry Pi 2 Model B	31.87 ⁴
TP-Link Nano USB Adapter TL-WN725N	10
Matlab*	0
FreeCad	0
Simulador Gazebo	0
Components electrònics addicionals	30
Imprevistos	50
TOTAL [€]	466.97

Figura 7: Taula de costos materials. Realitzada amb equivalència EUR/USD = 0,91055⁵

*Matlab és usat en els ordinadors de la universitat. Per treballar fora del campus, es pot usar scripts amb *Python*.

Costos de material: 466.97€

Cal afegir al pressupost un valor de imprevistos de 50€, per la compra de possibles peces del neato i hardware necessari per a la realització de petites millores al model.

³ "Cnet" [pàgina web], Coline West McDonald.

<https://www.cnet.com/products/neato-xv-essential-robot-vacuum/review/> [Consulta 15 de Maig de 2016]

⁴ "Raspberry Pi 2" [pàgina web], <https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/> [Consulta 15 de Maig de 2016]

⁵ "The Money Converter" [pàgina web]. <http://themoneyconverter.com/ES/USD/EUR.aspx> [Consulta 18 d'Octubre de 2016]

1.3.1.2. Costos de personal

Suposo que els treballadors treballen a una jornada laboral completa, és a dir 8h al dia, 5 dies a la setmana. Una jornada de 40h a la setmana.

Suposant un preu per hora: 20€

FASE	Temps estimat		Cost [€]
	Setmanes	Hores	
Repàs Nocions de Robòtica	2	80	1600
Primer contacte amb el robot	5	200	4000
Familiarització amb simulador	4	160	3200
Disseny 3D del Neato	2	80	1600
Socket de connexió Gazebo	4	160	3200
Funcions bàsiques Gazebo	12	480	9600
Estudi error típic	1	40	800
Millorar simulació del neato a Gazebo	6	240	4800
Disseny 3D de entorn de treball	2	80	1600
TOTAL	38	1520	30400

Figura 8: Taula de costos de personal

Cost total: 30.400€

1.3.1.3. Costos indirectes

Els costos indirectes són aquells que no estan implicats directament en el projecte, però que són necessaris com el lloguer d'una oficina o la factura de la llum.

Suposant:

- Lloguer d'una oficina personal per tal de poder treballar
- Pagament de aigua i llum durant tot el procés
- Compra d'ordinadors per treballar
- Compra de material d'oficina (paper, bolígrafs, taules...)

Raó	Cost per mes [€]	Cost total [€]
-----	------------------	----------------

Lloguer oficina	500 ⁶	4750
Llum i electricitat	100	950
Aigua	40	380
Ordinadors		1500
Material d'oficina		400
	TOTAL [€]	7980

Figura 9: Taula de costos indirectes

Cost: 7980 €

1.3.1.4. Cost total

	Cost [€]
Costos de personal	30400
Costos Indirectes	7980
Costos de material	466.97
TOTAL	38846.97

Figura 10: Taula de costos totals.

Es pot observar que el cost total seria de 38.846.97€.

1.3.2. Cost de replica d'un robot

Per tal de replicar el robot, seria necessari únicament el cost de les peces i del robot.

Element	Cost [€]
Robot Neato XV- Essential	345.1
Rasberrr Pi 2 Model B	31.87
TP-Link Nano USB Adapter TL-WN725N	10
TOTAL [€]	466.97

Figura 11: Costos de replica del robot.

Cost: 466.97€

⁶ "Idealista" [pàgina web], búsqueda de oficinas a Barcelona, mitja de les rellevants.
<https://www.idealista.com/alquiler-oficinas/barcelona-barcelona/> [Consulta 15 de Maig de 2016]

1.3.3. Cost reduït

El projecte es podria arribar a desenvolupar sense necessitat de fer servir el hardware real, usant el simulador Gazebo. L'API de Gazebo és totalment gratuïta, i una vegada fet el projecte es podria cedir a altres universitats. Usant scripts amb *python* el cost total seria totalment nul, assumint que la universitat ja està en posició del material necessari (ordinadors, aules...) i que els costos indirectes no cal contemplar-los .

1.3.4. Amortització

El projecte no té cap ideal econòmic. No es preten guanyar diners amb el producte resultant. El software desenvolupat podrà ser utilitzat pels alumnes de robòtica o futurs projectes de la universitat.

Si es pretén que el projecte arribi a altres universitats i usuaris, es pot usar la plataforma *GitHub*, on es podria penjar el codi i els manuals d'ús per tal que fos accessible per a tothom. Usar aquesta plataforma no generaria cap tipus de sobre cost.

1.4. Sostenibilitat

1.4.1. Econòmica

El projecte té un cost molt reduït, el robot Neato és un dels més barats i amb molt altes prestacions del mercat. La versió del robot utilitzat és un model antic, però amb pocs retocs al codi es podria actualitzar per als nous models.

El preu del projecte és molt competitiu, ja que la plac Raspberry Pi 2 és molt barata. El cost del robot també és molt reduït. EL més complex és tota la interfície i la programació, i per tant ho fa molt valorat en el mercat.

Tot el treball realitzat serà utilitzat per les classes de l'assignatura robòtica de la universitat, i serà molt útil per l'aprenentatge dels alumnes.

Per al treball s'utilitzen diferents codis fets per antics alumnes, que serviran per orientar els inicis del projecte. La majoria de funcions de connexió amb el robot ja estan implementades, i seran de molta ajuda per avançar ràpidament. En el simulador Gazebo, les físiques ja estan implementades a l'API i només s'han de gestionar pel model del Neato.

Amb la utilització del simulador, podríem reduir els costos considerablement, ja que no és necessari cap tipus de connexió a la xarxa elèctrica, ni tant sols del material. Només es necessita tenir un ordinador amb l'aplicació de Gazebo instal·lada i descarregar els models.

1.4.2. Social

El projecte es realitza a la ciutat de Barcelona en un estat de benestar. La universitat permet la realització de projectes amb un pressupost reduït, tot i que l'estat ha cancelat algunes de les ajudes a moltes universitats i centres de recerca.

El desenvolupament d'aquest projecte permetrà la millora en alguns aspectes de la robòtica mòbil. Que un robot mòbil sigui capaç de detectar l'error que està cometent i readreçar la seva trajectòria pot afavorir a la construcció de nous robots i la millora de les prestacions.

Cal afegir que el projecte serà de gran ajuda per a l'assignatura de robòtica. Permetrà que els alumnes desenvolupin pràctiques d'una manera més fàcil ent servir el model del robot Neato per al simulador Gazebo. A més podran usar diferents funcions per tal de realitzar noves pràctiques.

Per universitats amb poc pressupost o universitats de països en desenvolupament, aquest projecte podria ser de gran ajuda. Usant únicament el simulador Gazebo, els alumnes poden estudiar els diferents models de robòtica mòbil i incrementar els seus coneixements sobre un tema d'actualitat. A més, si es el projecte es fés públic, podria servir també a gent que no pot accedir a comprar un robot.

El report social també té inconvenients, es tracta de la inclusió de robots en el món de la indústria. Els robots són de gran ajuda, i “fan la feina de l’home” per tal d’ajudar-lo. Però al fer les tasques de les persones, poden perjudicar-les deixant sense feina a molts treballadors. La revolució de les màquines es un llarg tema a tractar, però en el cas d’aquest petit projecte, no afecta negativament en la societat.



Figura 12: Alumnes usant un robot Lego Mindstorms

1.4.3. Ambiental

El robot Neato funciona amb energia elèctrica. Els robots que usem estan connectats a la xarxa elèctrica de la Universitat Politècnica de Barcelona al Campus Nord. La procedència d’aquesta energia és totalment transparent, ja que no sabem quina és la companyia elèctrica que ens la transmet. Segons la companyia podríem saber si aquesta energia és generada de manera renovable o amb matèries fòssils.

El robot no genera residus de cap tipus, i la resta de components tampoc. En el cas de llençar el robot, és totalment reciclable, ja que la seva carcassa és completament de plàstic, i els components poden anar a una planta de reciclatge on serien degudament reutilitzats. La placa Raspberry Pi 2 pot ser reutilitzada en infinitat de projectes més i per tant mai serviria de deixalles. En tot cas, els robots seguiran muntats per als alumnes de l’assignatura de robòtica, i els podran utilitzar sempre que vulguin accedir als laboratoris docents.

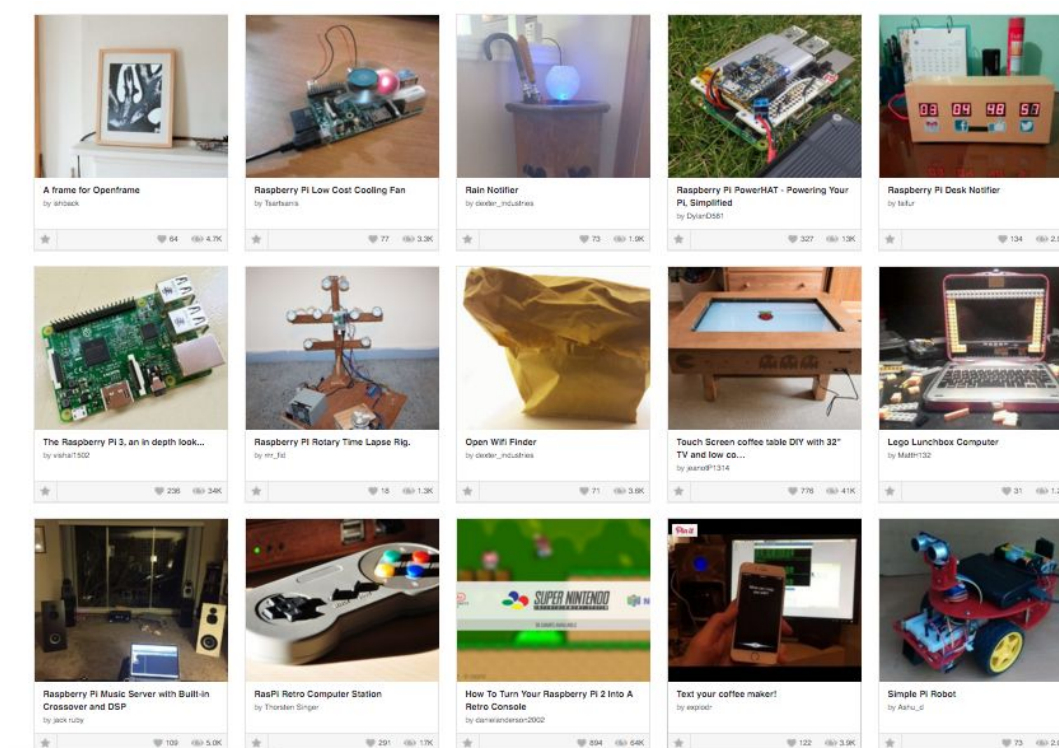


Figura 13: Exemples d'ús de la Raspberry Pi 2

El robot Neato ha estat comprat de fàbrica i desmantellat per tal d'instalar la Raspberry Pi 2 i la resta de components necessaris per a l'estudi i treball. No podem saber si els materials usat per construir les plaques han estat recol·lectats per uns treballadors en bones condicions. Sabem que molts materials usats en la construcció de components ve de països del tercer món on mafies esclavitzen persones per treballar a les mines i extreure'n material. L'empresa Neato Robotics no explica la procedència dels seus materials, ja que la majoria de usuaris dels robots de neteja no saben les condicions de treball de les persones que recullen aquests materials. Personalment crec que seria viable dissenyar el robot amb materials reciclats, sobretot la carcassa.

Amb l'ús del simulador, es podrien reduir els residus d'aquest projecte. L'únic material físic usat seria un ordinador i la corrent que gasta.

1.4.4. Matriu de sostenibilitat

	PPP	Vida Útil	Riscos
Ambiental	Consum del disseny [0:10]	Petjada Ecològica [0:20]	Riscos ambientals [-20:0]
	10	17	-1
Econòmic	Factura [0:10]	Pa de viabilitat [0:20]	Riscos econòmics [-20:0]
	6	15	-4
Social	Impacte personal [0:10]	Impacte social [0:20]	Riscos socials [-20:0]
	9	12	-5
Rang	25	44	-10
Sostenibilitat [-60:90]	59		

Figura 14: Matriu de sostenibilitat

1.5. Lleis i regulacions

L'únic inconvenient de usar una API com és la de GazeboSIM és que s'ha de tenir en compte les seves lleis d'ús. Les regulacions estan basades en *Apache Licenses* una empresa que otorga i regula llicències de software i API's.

2. Objectius

2.1. A curt termini

Els objectius d'aquest treball són realitzar un simulador que funcioni correctament. Per tal de fer aquest simulador, hauré de realitzar un estudi de com funcionen els robots per tal de replicar el comportament dins del simulador. A més hauré de crear uns pluguins de connexió que puguin ser connectats de la mateixa manera que el robot. També serà necessari crear les comandes per rebre informació dels sensors del robot.

2.2. A llarg termini

L'objectiu a llarg termini, és que aquesta practica pugui ser utilitzada en les pràctiques dels alumnes de l'assignatura de Robòtica.

3. Introducció

En aquesta part del treball realitzaré una introducció de l'entorn de treball. Estudio el comportament del robot i les seves característiques per tal de poder realitzar correctament el treball. Les fases incloses són:

- Repàs de nocions de Robòtica
- Primer contacte amb el robot.
- Estudi de l'error típic.

3.1. Neato XV essential

3.1.1. Introducció

El robot que utilitzaré per aquest treball és el robot Neato XV essential, un robot de neteja domèstic. És un robot molt ben implementat en comparació a altres i amb un preu molt competitiu.



Figura 15: Imatge del robot.

3.1.2. Sensors

El Neato XV essential conté diferents tipus de sensors, els quals el fan ser més precís que la resta dels robots del mercat. Els tres sensors que utilitza són:

- Bumpers: El paraxocs delantera utilitza 3 sensors (laterals i central) per tal d'identificar amb quina part ha xocat.
- Sensor de paret: Permet identificar a quina distància està d'una paret. Està només ubicat a la part esquerra del robot.
- Làser: Un sensor làser que ens permet veure 360° d'on estem.
- Sensors de caiguda frontal: Per tal que el robot no pugui caure en el cas d'haver una escala o un precipici.
- Encoders: Per tal de saber quan han avançat les rodes. Tenen 1mm de resolució.



Figura 16: Detall dels sensors i actuadors.



Figura 17: Mides del robot.

3.1.3. Equipament i tecnologia

Per estudiar la robòtica, el millor equipament possible és tenir un robot que tingui encoders i un sensor làser de 360°. D'aquesta manera es poden realitzar pràctiques i estudis fent moure el robot en un entorn per tal d'identificar landmarks i reduir l'error de moviment. Aquest tema es tractarà en el següent apartat.

L'empresa Neato Robotics, amb seu principal a Califòrnia, és el creador d'aquest magnífic robot. Ells mateixos realitzen vídeos per demostrar la diferència entre els seu model i d'altres més exitosos.

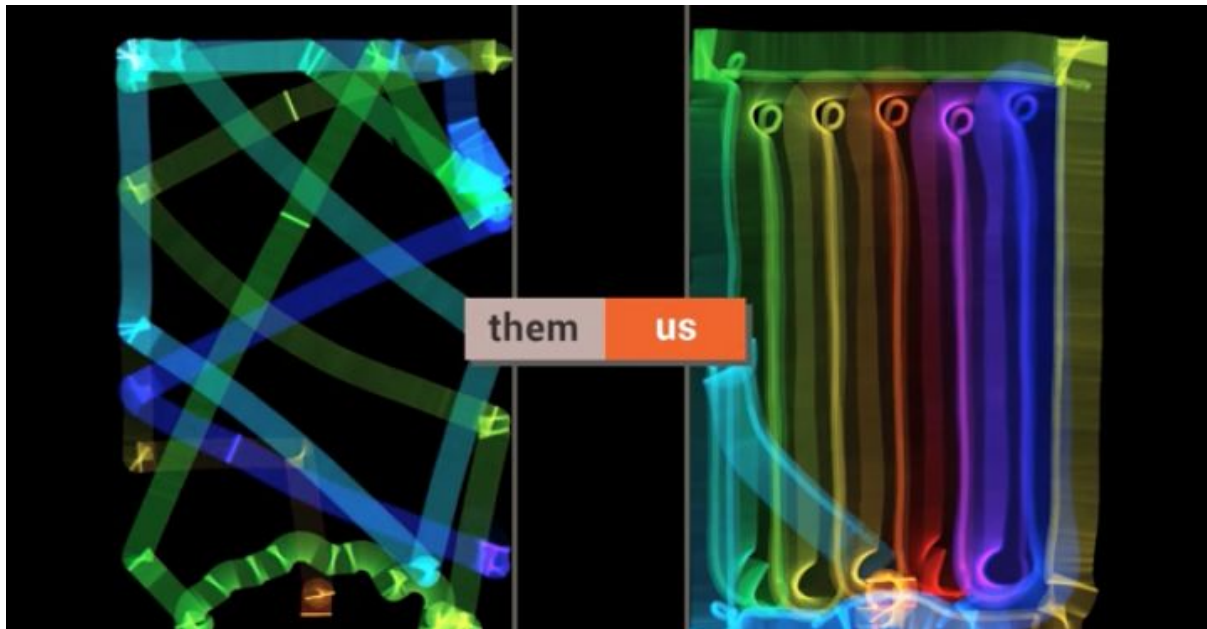


Figura 18: Comparació entre Neato i Roomba.⁷

En la comparació de la figura 18, podem observar la diferència entre el Neato i el Roomba. Podem observar que el Roomba funciona per cops; utilitzant els seus sensors “bumpers”, que fan canviar la seva direcció al xocar contra les parets. El Neato XV essential utilitzant el seu potent sensor és capaç d’ubicar-se dins d’una casa, i reconèixer per on ha passat.

3.1.4. Connexió i modificació

El robot Neato té un port usb que hem fet servir per connectar-hi una Raspberry Pi 2, amb la qual ens connectem mitjançant un socket per wifi i ho connectem a l'ordinador. Des de l'ordinador podem enviar comandes als actuadors del robot i inclús rebre la informació dels sensors del robot. Amb tota aquesta informació podem crear un script des d'una aplicació per fer moure el robot de la manera que més ens convingui.

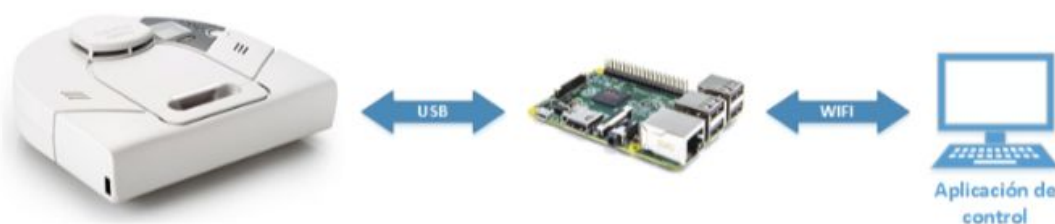


Figura 19: Connexió del robot

⁷ “Neato Robotics”, video de demostració. <https://www.neatorobotics.com/es/>

3.2. Detecció de l'entorn

Per tal de fer moure un robot en un entorn necessitem saber moltes dades. Aquestes dades les hem d'obtenir dels sensors acoplats al robot.

A l'assignatura de Robòtica de la Universitat, les pràctiques es realitzen únicament amb sensors làser i encoders. Aquests sensors ens permeten tenir un visió de l'error que cometem els robots en desplaçar-se i a rectificar-lo.

Quan fem que el robot realitzi un moviment, no sempre el fa a la perfecció. Això és degut a diferents tipus d'errors: sistemàtics i aleatoris.

Els errors sistemàtics són provocats per els encoders. Els encoders són un tipus de sensor que s'ubica en els eixos de la roda. Solen ser discos amb forats que permeten passar la llum d'un led entre ells. A mesura que la roda gira els forats del disc van girant fent passar la llum o no. Sabent el diàmetre i els forats del disc, podem obtenir quantes voltes ha fet, i per tant la distància. Segons la resolució, distància entre forats, d'aquests podem tenir errors de moviment, ja que si la resolució dels forats és de 5mm, si ens movem 4mm ens indicarà que ens hem mogut 0mm.

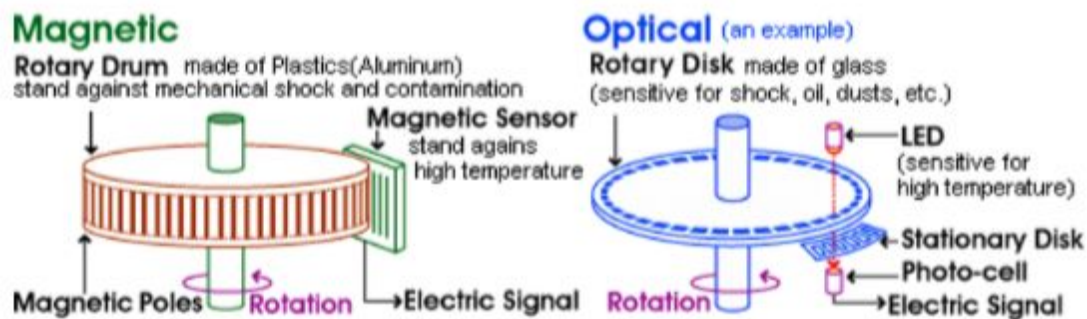


Figura 20: Imatge de un encoder

Un altre error sistemàtic és l'acceleració i la frenada del robot. Segons com estigui configurat pot fer que les rodes patinin i aquestes detectin moure's quan el robot en conjunt no s'ha mogut.

La resta d'errors són aleatoris, ja que són produïts per les irregularitats de la superfície per la qual ens movem. Per exemple, que en el camí hi hagi sorra, pedres o un sot a la calçada, poden fer que una roda giri més que una altra en un cert punt, de manera que perdem la direcció i/o la distància correcta.

En el cas de les rotacions, els errors sistemàtics produeixen un gran efecte en l'error de la trajectòria del robot. Girar un sol grau més i seguir endavant ens fa diferir molt de la posició a on creiem que estem. S'utilitzen 2 posicions: la que creu que està i a la que realment està.

Per aquesta raó s'utilitza el sensor làser. El sensor làser és molt fiable, i permet identificar l'entorn a on estem. D'aquesta manera es poden trobar punts de referència, que s'anomenen Landmarks. Abans d'un moviment s'identifiquen els landmarks, i una vegada s'ha realitzat el moviment es torna a identificar aquest Landmark. Llavors es pot veure la diferència entre la posició real a on veiem el Landmark i la que realment hauria d'estar. Així es pot identificar l'error que el robot ha comès i recalculat la ruta des d'aquell punt.

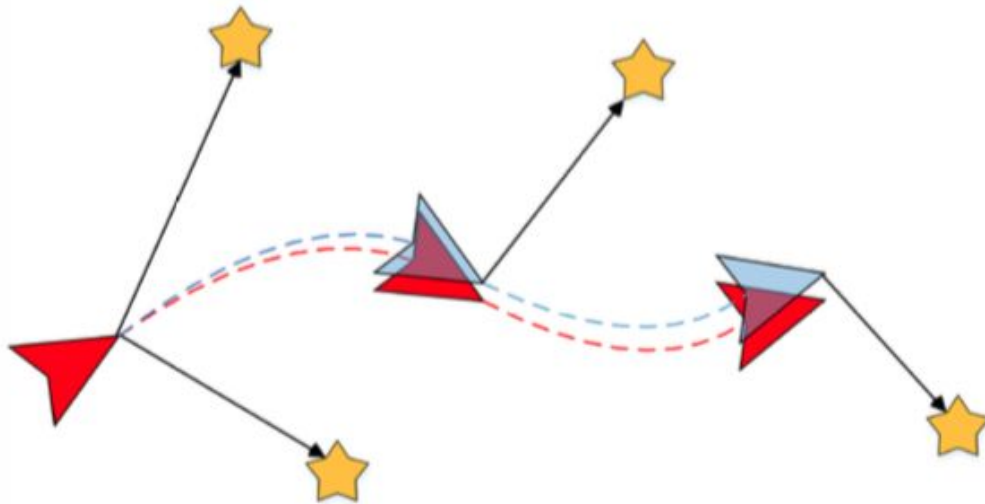


Figura 21: Usant els landmarks trobem la posició del robot.

Els diferents tipus de landmarks són:

- Landmarks Artificials: Són aquells que "l'home" ha col·locat, com ara potes de cadira, llunes... per reduir l'ambigüitat de l'entorn. Són molt fàcils de detectar, ja que al llegir-los fan un canvi molt brusc de valors
- Landmarks Naturals: Són aquells que estan a l'entorn com cantonades i punts molt característics de les parets. Són més difícils de detectar, ja que és necessari estar suficientment aprop per poder veure-ls, i en podem perdre la referència si ens allunyem.

3.3. Estudi error típic

L'estudi de l'error típic del robot és molt important, ja que determina quines tendències a error sol fer el robot. Per la poca disponibilitat només s'ha pogut realitzar un test. Aquest estudi està realitzat amb un mesurador làser. Aquest aparell s'enganxava al robot, i d'aquesta manera es podia calcular la distància real recorreguda. S'utilitzava fent una lectura abans de fer moure el robot i una altra acabat el recorregut, enfocant a una paret perpendicular a la seva direcció. Aquesta pràctica pot generar errors sistemàtics, ja que la posició no sempre serà perfecte; això podria provocar que el robot no circulés totalment perpendicular a la paret i fes diferències d'un parell de graus.

També, amb l'ajuda de matlab, podia aconseguir el valor dels encoders una vegada passat cada prova.

Llegenda de la taula

DATA1 i DATA2: Són les referències amb les quals es mesurava la distància total que s'havia mogut el robot.

REAL PATH: és la distància recorreguda pel robot.

Encoder RIGHT: Lectura de l'encoder dret al final de la prova.

Encoder LEFT: Lectura de l'encoder esquerra al final de la prova.

RIGHT [mm]: Diferència entre distància real (REAL PATH) i distància de l'encoder.

RIGHT [%]: Tant per cent de diferència.

LEFT [mm]: Diferència entre distància real (REAL PATH) i distància de l'encoder.

LEFT [%]: Tant per cent de diferència.

Test realitzat amb:

Distància = 1000mm

Velocitat = 100 mm/s

NEAT O A	EXTERNAL LASER DATA			ENCODERS		ERROR			
#TEST	DATA 1	DATA 2	REAL PATH	Encoder RIGHT	Encoder LEFT	RIGHT [mm]	RIGHT [%]	LEFT [mm]	LEFT [%]
1	3727	2722	1005	1001	1001	4	0.398009 9502	4	0.398009 9502
2	3725	2720	1005	1001	1002	3	0.298507 4627	3	0.298507 4627
3	3725	2719	1006	1001	1001	5	0.497017 8926	5	0.497017 8926
4	3728	2721	1007	1001	1002	6	0.595829 1956	5	0.496524 3297
5	3728	2723	1005	999	1001	6	0.597014 9254	4	0.398009 9502
6	3730	2724	1006	1001	1003	5	0.497017 8926	3	0.298210 7356
7	3730	2726	1004	1002	1001	2	0.199203 1873	3	0.298804 7809
AVERAGE	3727.571 429	2722.142 857	1005.428 571	1001	1001.571 429	4.428571 429	0.440371 5009	3.857142 857	0.383583 586

Figura 22: Taula amb els valors de l'estudi.

Cada robot té un error diferent, però aquest estudi està realitzat amb el robot A. Podem observar que tendeix a moure's més del que toca. Les proves s'estan realitzant amb una distància de 1000mm i la mitja de distància que es mou es de 1005mm. Sabent que els encoders fan una mitja de 1001mm en lectura, l'error està en la calibració dels encoders.

Es podria crear un programa per tal de detectar aquest error, utilitzant el làser incorporat que porta el robot. Exactament aquest és el sistema que es sol utilitzar per reduir l'error. Els encoder solen fallar en freqüència, ja que les seves lectures no són precises. El làser és totalment fiable, però utilitzar-lo ens genera un cost més elevat que el dels encoders. Per tal de poder fer aquests càlculs, ens hem de basar en un landmark, que són punts de referència en l'entorn. La detecció de Landmarks no és senzilla, es tractarà el tema més endavant.

4. Desenvolupament

En aquest apartat s'explicarà el desenvolupament del treball. Entre els quals els plugins del simulador, l'entorn 3D..

Aquesta part engloba les fases:

- Familiarització amb el simulador
- Disseny 3D del robot Neato
- Socket de connexió Gazebo
- Funcions bàsiques del robot al Simulador Gazebo
- Millora de la simulació
- Disseny 3D de l'entorn

4.1. Desenvolupament del Simulador

4.1.1. Introducció

Gazebo SIM és una API desenvolupada per la Universitat de Southern California. És un simulador molt ben programat, que permet recrear situacions reals de la robòtica amb un motor de físiques espectacular.

A la pàgina de l'API, hi ha un munt de tutorials que ensenyen a l'usuari a realitzar simulacions senzilles, fins a crear els teus propis robots.

Aquest programa també permet crear els teus propis sensors i escenaris, fent ús de plugins i programes de disseny 3D.

La idea d'utilitzar el simulador, es basa en el fet que els sistemes *hardware* solen fallar en freqüència. Els robots que usem al laboratori fallen sovint, habitualment s'ha de reiniciar la connexió, i de vegades es penja el programa que els fa córrer. Aquests errors són habituals al treballar amb prototips *hardware*.

Per això es va decidir realitzar un model del nostre robot *Neato essential XV* amb eines de 3D per tal de passar-lo al simulador Gazebo.

Per tal de simular exactament l'entorn del robot, es va decidir crear uns plugins de connexió que imitiessin el socket de connexió que tenim des de l'ordinador a la Raspberry Pi2. D'aquesta manera podríem córrer els programes realitzats en la plataforma Matlab o Python en el simulador, senzillament canviant z<z el port de connexió.

4.1.2. Creació del model del robot

Per tal de dissenyar el robot he utilitzat el programa FreeCad, que permet dissenyar objectes en 3D.

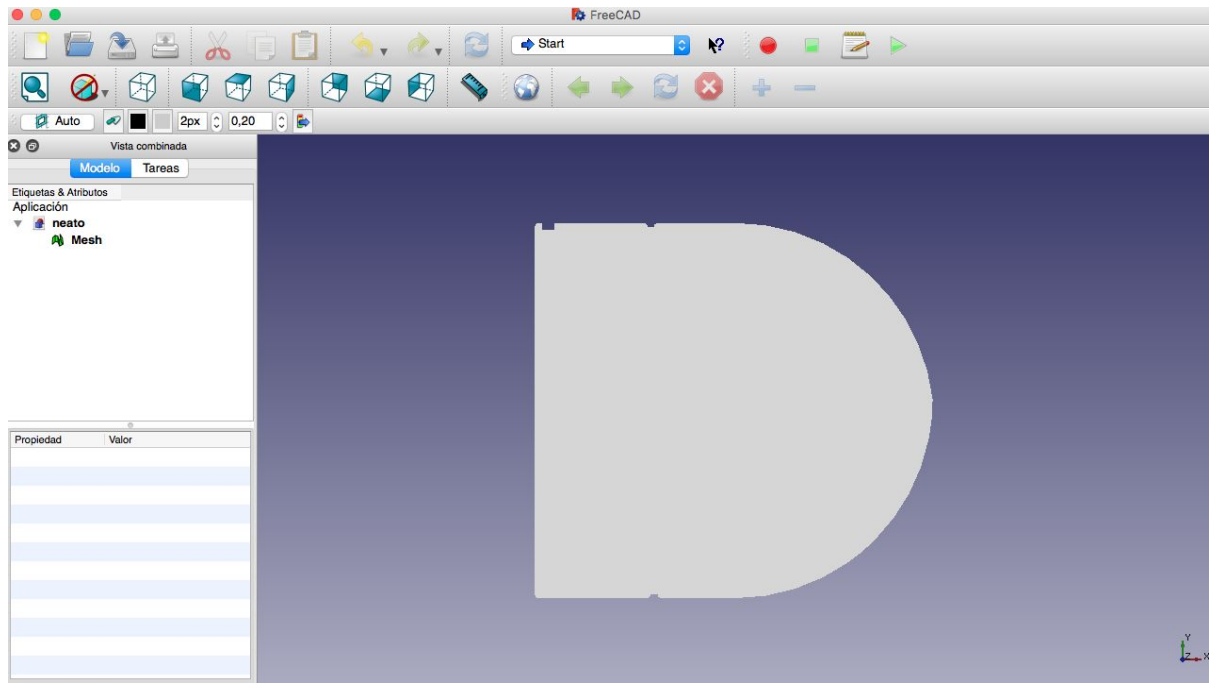


Figura 23: Imatge en planta del disseny del 3D del robot

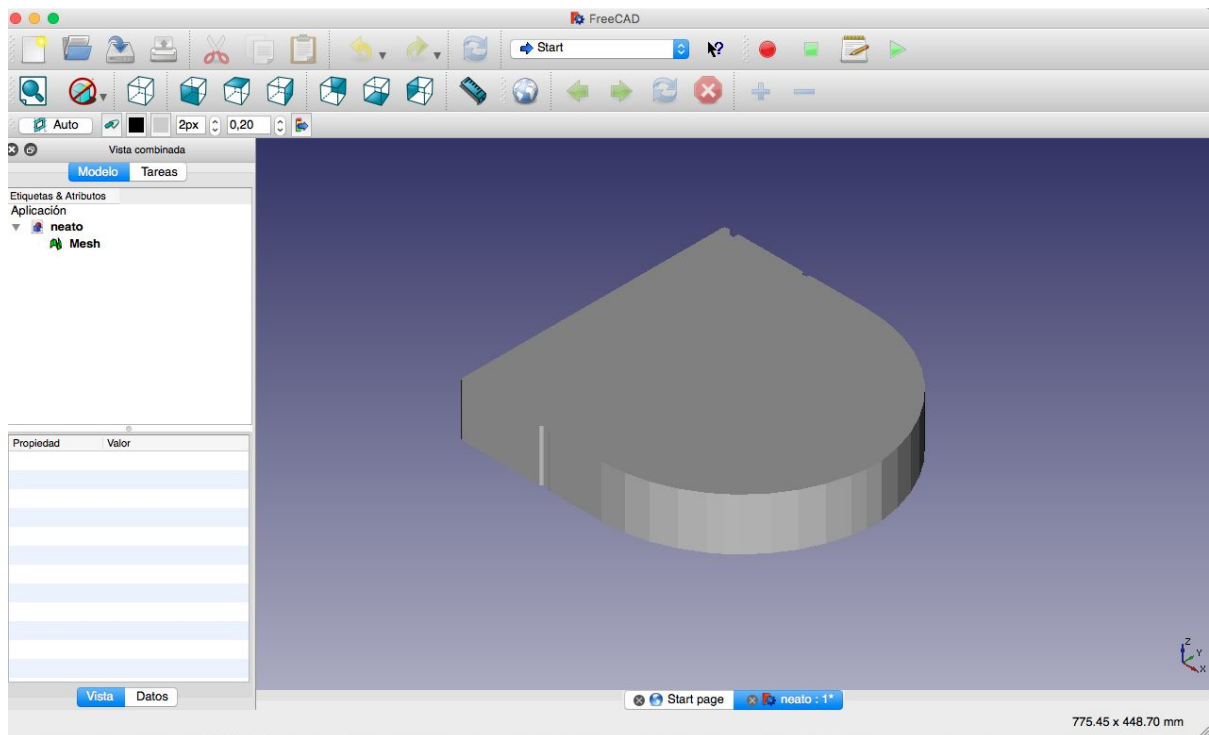


Figura 24: Imatge en perspectiva del disseny del 3D del robot

La realització del model ha estat senzilla. La forma del robot és tant minimalista que només utilitza un rectangle amb una semicircunferència contiguous a una de les arestes. He realitzat una versió molt senzilla del model per tal de optimitzar en velocitat i efectivitat. En les simulacions es prioritzen les físiques del robot i no la qualitat de la imatge; una imatge del robot amb més resolució i detall provocaria que el simulador es relantitzés.

Les físiques del robot estan documentades en un arxiu. L'API permet afegir sensors i configurar-los de la manera desitjada.

4.1.3. Creació plugin connexió Simulador

He creat el plugin per fer la connexió del socket imitant el mateix que usa la Raspberry Pi2, per tal que els alumnes de l'assignatura de robòtica hi puguin treballar correctament. Com s'ha dit anteriorment, el plugin es connecta via socket, si canviem el port del programa que controla el robot realitzat amb matlab, podem variar la connexió entre el robot real i el robot de la simulació.

El plugin de connexió està fet perquè rebi les mateixes instruccions que el robot. És a dir, està preparat per rebre les instruccions bàsiques que el permet fer moure, rebre les lectures dels sensors de les rodes (encoders) i de les lectures del làser.

4.1.4. Creació de funcions bàsiques del Simulador

El robot Neato té un gran repertori de funcions programades, tals com emetre sons o retornar a la base per recarregar. Aquestes funcions estan implementades per funcionar a nivell domèstic. Per realitzar les pràctiques de robòtica només ens calen 3 funcions, ja que no ens es necessari les funcions d'usuari. Aquestes 3 funcions ens permeten fer moure el robot i conèixer l'entorn que l'envolta:

- SetMotor: Permet donar una velocitat i distància als motors del robot.
- GetMotors: Aquesta funció ens retorna el valor dels encoders de cada roda
- GetLDSScan: Aquesta funció ens retorna la distancia de cada grau del laser.

La funció GetLDSScan ha estat la que més problemes ha donat. He utilitzat un sensor làser programable que proporciona l'API de GazeboSim. Aquest sensor ha causat molts problemes ja que no es proporcionava en cap manual l'ús d'aquest. Vaig plantejar crear un sensor des de zero però l'intent va ser fallit. Finalment vaig aconseguir trobar la manera d'instanciar el sensor, tot i que no va ser fàcil.

Utilitzant un parser diferencio entre les diferents comandes que s'envien, una vegada interpretades es crida la funció corresponent.

```
switch (parser.command) {
    case CommandParser::SET_MOTORS:
        if (parser.parameters.count("LWheelDist")) {
            left_wheel.src_j_angle = left_wheel.unwrap_j_angle;
            left_wheel.dst_j_angle = left_wheel.unwrap_j_angle +
                                    parser.parameters["LWheelDist"] / (1000.0 * radius);
        }
        if (parser.parameters.count("RWheelDist")) {
            right_wheel.src_j_angle = right_wheel.unwrap_j_angle;
            right_wheel.dst_j_angle = right_wheel.unwrap_j_angle +
```

```

radius);
    }
    if (parser.parameters.count("Speed")) {
        speed = parser.parameters["Speed"] / (1000.0 * radius);
    }
    client_socket.writeline("OK");
    break;

```

Figura 25: Codi de la funció SetMotors

```

case CommandParser::GET_MOTORS: {

    std::string temp = "%s_Encoder,%f %s_PositionInMM,%i ";
    std::string output;
    if (parser.parameters.count("LeftWheel")) {
        char buff[500];
        memset(buff, 0, 500);
        sprintf(buff, temp.c_str(), "LeftWheel", left_wheel.unwrap_j_angle * 100 / 3,
"LeftWheel",
        left_wheel.unwrap_j_angle * 1000 * radius);
        output += buff;
        std::cout << "LeftWheel: " << left_wheel.unwrap_j_angle * 100 / 3 << std::endl;
    }
    if (parser.parameters.count("RightWheel")) {
        char buff[500];
        memset(buff, 0, 500);
        sprintf(buff, temp.c_str(), "RightWheel", right_wheel.unwrap_j_angle * 100 / 3,
        "RightWheel", right_wheel.unwrap_j_angle * 1000 * radius);
        output += buff;
        std::cout << "RightWheel: " << right_wheel.unwrap_j_angle * 100 / 3 << std::endl;
    }
    client_socket.writeline(output);
    break;

```

Figura 26: Codi de la funció GetMotors.

```

case CommandParser::GET_LDS_SCAN: {

    oss << "##GetLDSScan;AngleInDegrees,DistInMM,Intensity,ErrorCodeHEX";
    angle = lidar->AngleMin().Degree();
    res = lidar->AngleResolution()*180/M_PI;
    for (int i = 0; i < lidar->GetRayCount(); ++i){
        oss << ";" << (angle+i*res) << "," <<
        (int)(lidar->GetRange(i)*1000+0.5) << ",1,0";
    }
    oss << ";ROTATION_SPEED,5.1";
    client_socket.writeline(oss.str());
}
break;

```

Figura 27: Codi de la funció LDS Scan.

4.1.5. Ús del plugin

Necessitem utilitzar dos terminals per tal de poder provar el simulador.

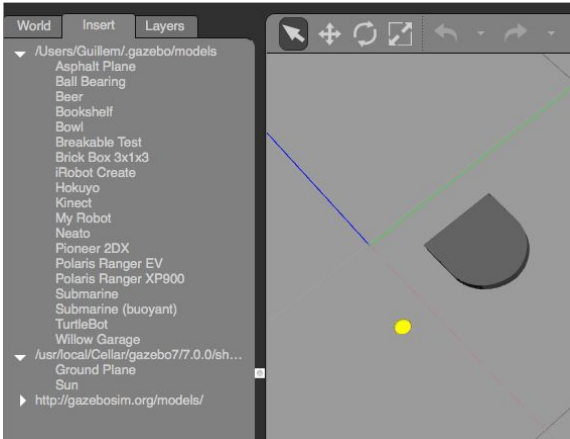
TERMINAL SERVIDOR	TERMINAL CLIENT
<code>cd .gazebo/models/neato/plugin/sockets_cpp</code>	<code>cd .gazebo/models/neato/plugin/sockets_cpp</code>
<code>/gazebo</code>	
Posar el robot Neato a un "World" 	
	<code>./simple_client 50000</code>
Retorna: New connection to client!	

Figura 28: Conexió client servidor

Una vegada connectats, podem fer ús de les comandes creades.

4.1.6. Ús de les comandes

4.1.6.1. SetMotor

Distància de la roda Esquerra: *LWheelDist [mm]*

Distància de la roda Dreta: *RWheelDist [mm]*

Velocitat de les rodes: *Speed [mm/s]*

SetMotor LWheelDist x RWheelDist x Speed x

Retorna: -

4.1.6.2. GetMotors

Retorna el valor dels encoders de les rodes.

GetMotors LeftWheel RightWheel

Retorna: Els valors dels encoders de les rodes. A nosaltres ens interessa “Wheel_PositionInMM” ja que ens proporciona la distància que ha recorregut el robot; segons els sensors dels encoders, ja que pot diferir de la distància real recorreguda.

```
LeftWheel_MaxPWM,65536 LeftWheel_PWM,-858993460 LeftWheel_mVolts,1310  
LeftWheel_Encoder,0 LeftWheel_PositionInMM,0 LeftWheel_RPM,- 13108  
RightWheel_MaxPWM,65536 RightWheel_PWM,-858993460  
RightWheel_mVolts,1310 RightWheel_Encoder,0 RightWheel_PositionInMM,0  
RightWheel_RPM,-13108
```

Figura 29: Exemple de retorn de GetMotors

4.1.6.3. GetLDSScan

Retorna el rang de cada angle del sensor làser.

GetLDSScan

Retorna: La distància en mil·límetres per cada angle. També un codi d'error i la intensitat del senyal.

```
AngleInDegrees,DistInMM,Intensity,ErrorCodeHEX 0,221,1400,0  
1,223,1396,0 2,228,1273,0 (. . .) 359,220,1421,0 ROTATION_SPEED
```

Figura 30: Exemple de retorn de GetLDSScan

4.1.7 Exemple d'utilització

De la funció GetLDSScan podem saber a quin entorn estem i inclús realitzar la detecció de landmaks. En aquest exemple he utilitzat una simulació de GazeboSim per detectar diferents superfícies.

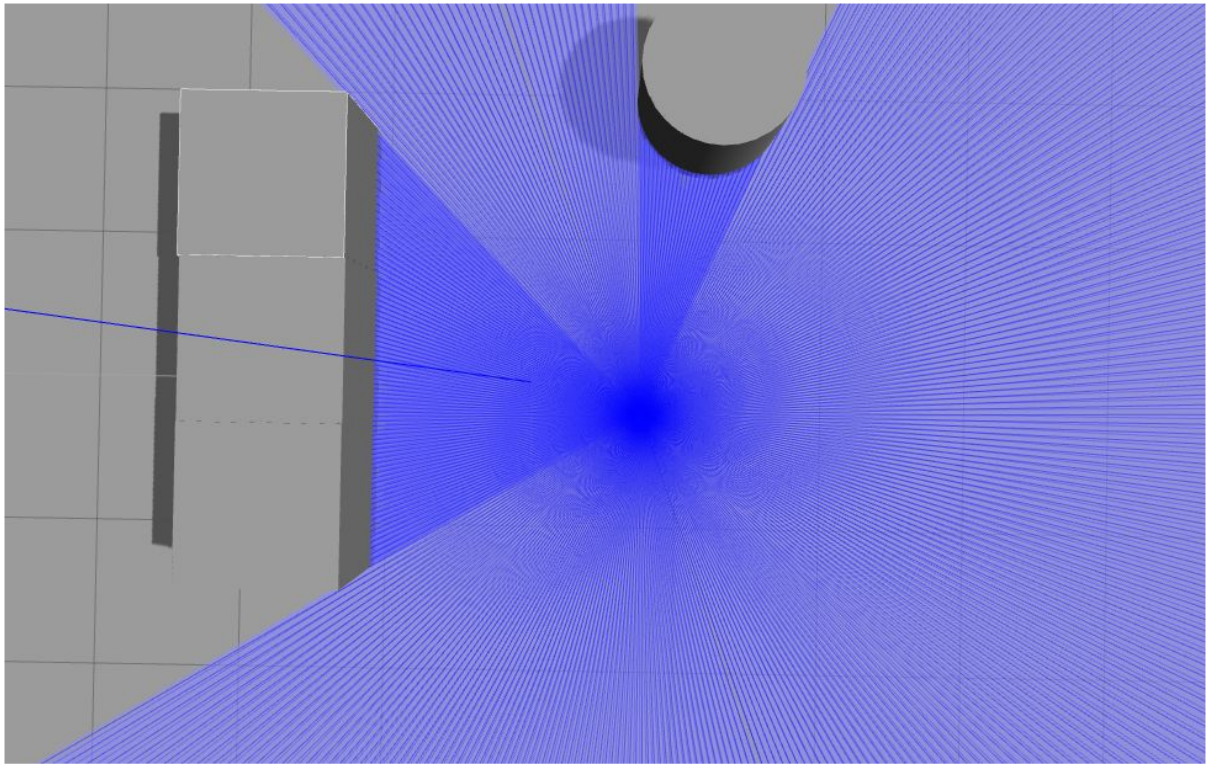


Figura 31: Entorn proposat

En la figura 31 podem observar l'entorn proposat, tres cubs que formen un rectangle i un cilindre. Una vegada hem obtingut els valors del sensor làser, podem utilitzar-los per trobar els landmarks i l'entorn a on estem.

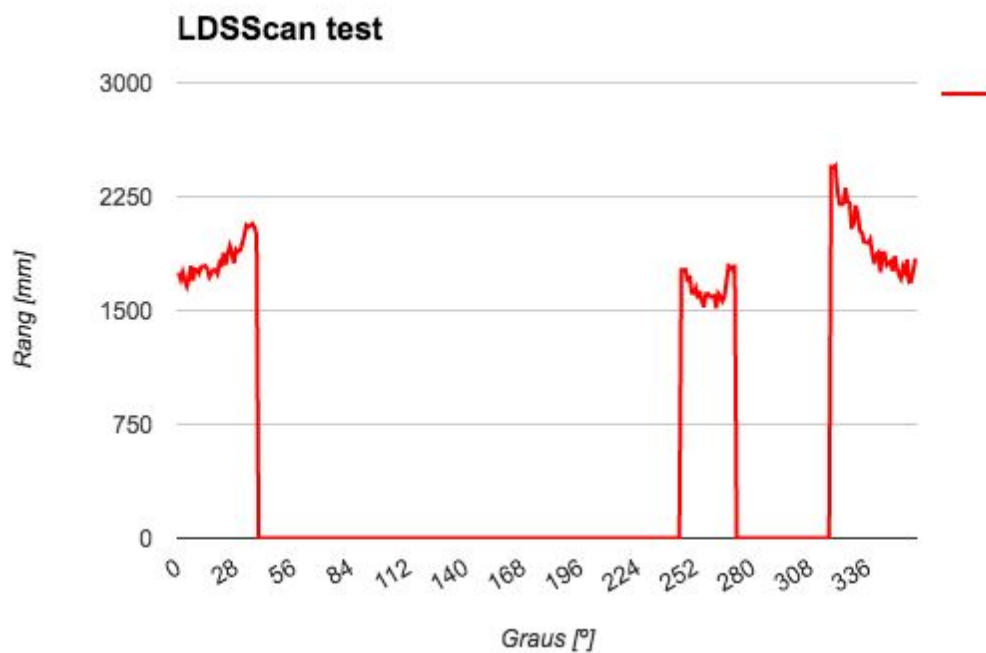


Figura 32: imatge del gràfic creat a partir de les dades.

En el gràfic de la figura 32 podem observar els valors del làser envers l'entorn donat. El làser descarta els valors quan aquest passa d'un cert rang i per aquesta raó hi ha molts valors a zero. Es pot identificar el cilindre entre els graus 250° i 280° . A causa de la seva forma la detecció en la gràfica és aquesta. Dels graus 0° al 28° i del 308° al 360° podem observar la paret construïda a base de cubs. El robot està col·locat de manera perpendicular a la paret formada, i el làser comença a fer la lectura a la direcció del robot. Podem observar com la paret s'allunya entre els graus 0° i 28° i com aquesta s'apropa dels graus 308° al 360° .

Utilitzant aquests valors som capaços de detectar landmarks.

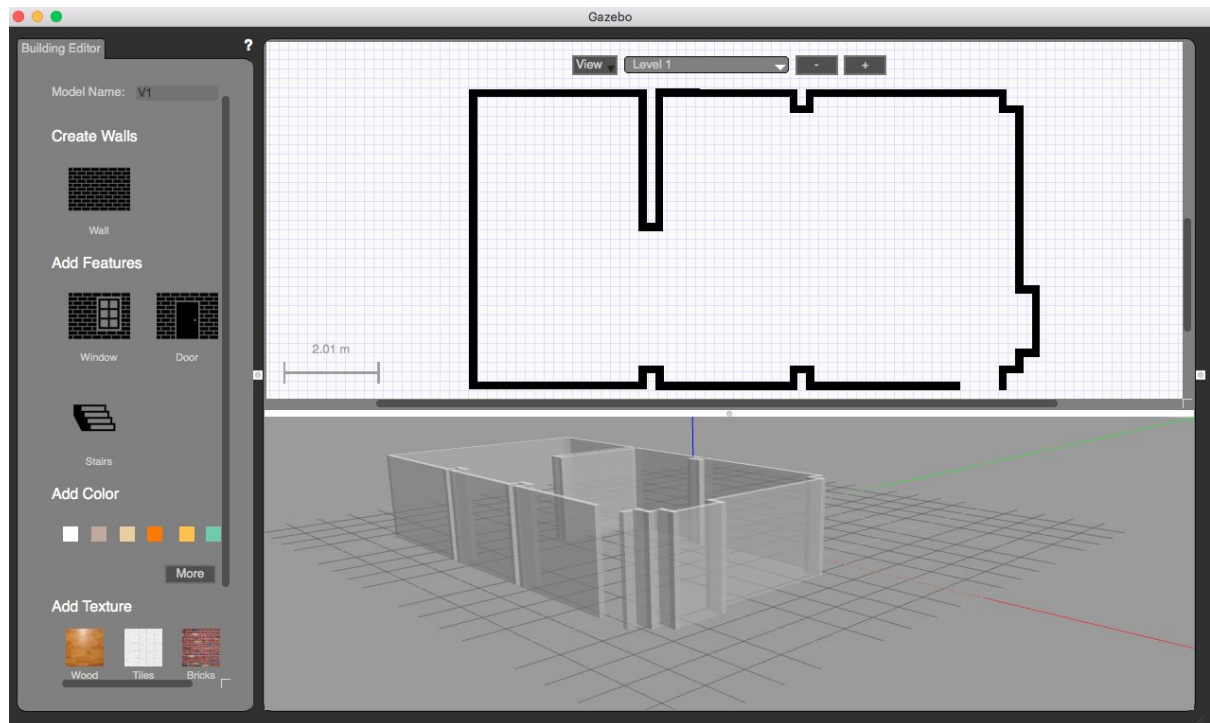


Figura 34: Vista del dissenyador d'edificis.

El problema que planteja el dissenyador d'edificis (*Building editor*), és que no es pot modificar una vegada guardat; és a dir s'ha de fer a la primera i tot seguit. A més, la interfície és complicada i poc eficient: les parets s'enganxen entre elles mateixes, cosa que ajuda a crear habitacions, però si després de posar la mida de una paret mous la que està enganxada, totes les parets es mouen i perden la seva forma i mida que has determinat.

Un altre problema lligat a l'anterior, és el fet que no hi ha la funció de "ctrl + z", és a dir, la funció de "desfer". Si t'equivoques en algun moment i sense voler espatlles tot el disseny (sol passar amb freqüència) no hi ha manera de tornar enrera i s'ha de començar de nou.

Després de varis intents i problemes, vaig aconseguir realitzar un model estricte a les mesures preses anteriorment.

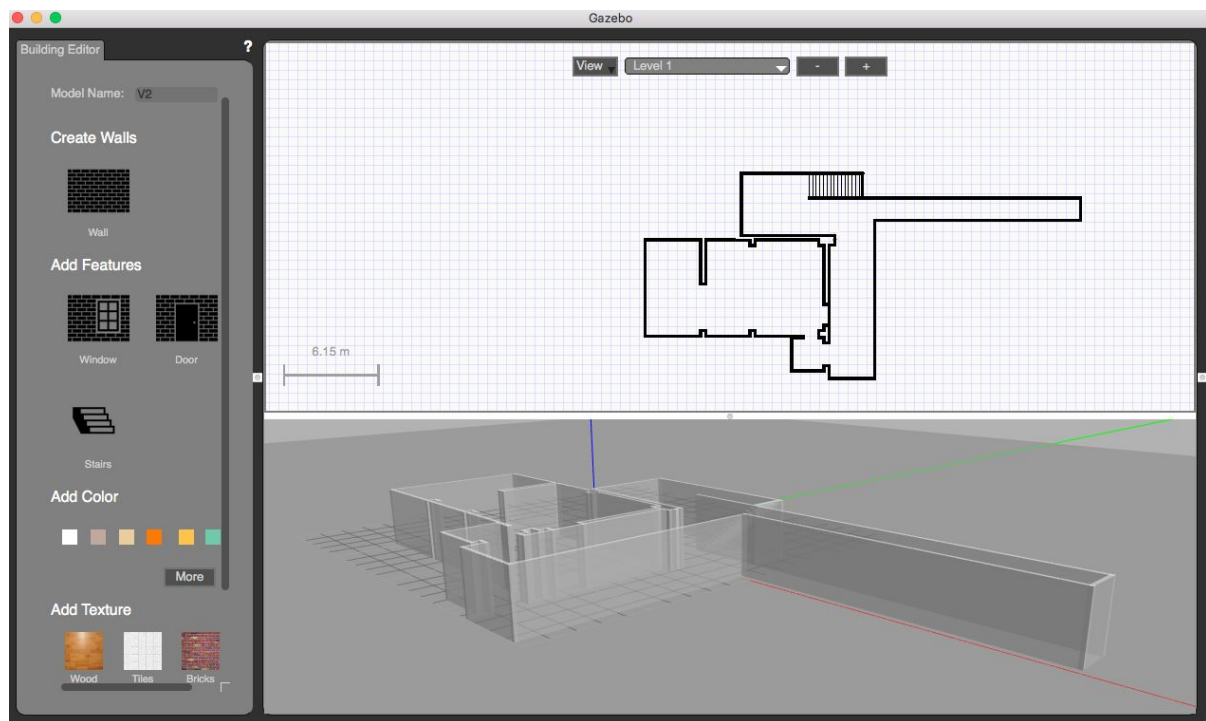


Figura 35: Alçat i prespectia de l'entorn acabat.

Quan tenim el disseny ja guardat, l'API GazeboSim permet importar-lo sempre que volguem al nostre entorn de treball "world". Una vegada posat a un world, podem afegir-hi material i tot tipus d'objectes. En aquest cas he intentat recrear l'entorn el màxim possible, col·locant les taules pròpies del aulari i els diferents armaris de material.

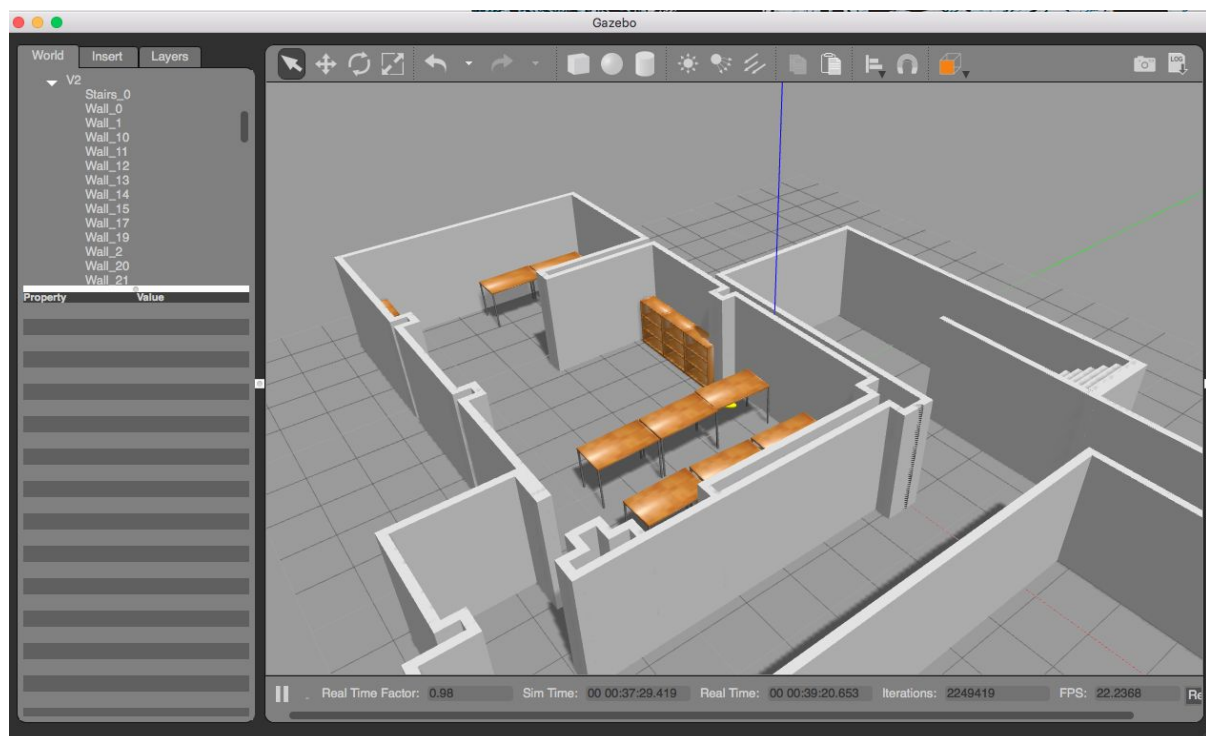


Figura 36: Imatge de l'entorn 3d

Al realitzar les pràctiques al laboratori, s'usen uns tubs cilíndrics per tal d'intentar localitzar-los. En aquest cas he usat unes llaunes com a exemple.



Figura 37: Visió en planta de l'aula.

Aquestes llaunes i objectes es poden moure de la manera que més convingui, ja que l'estructura de l'aula pot variar.

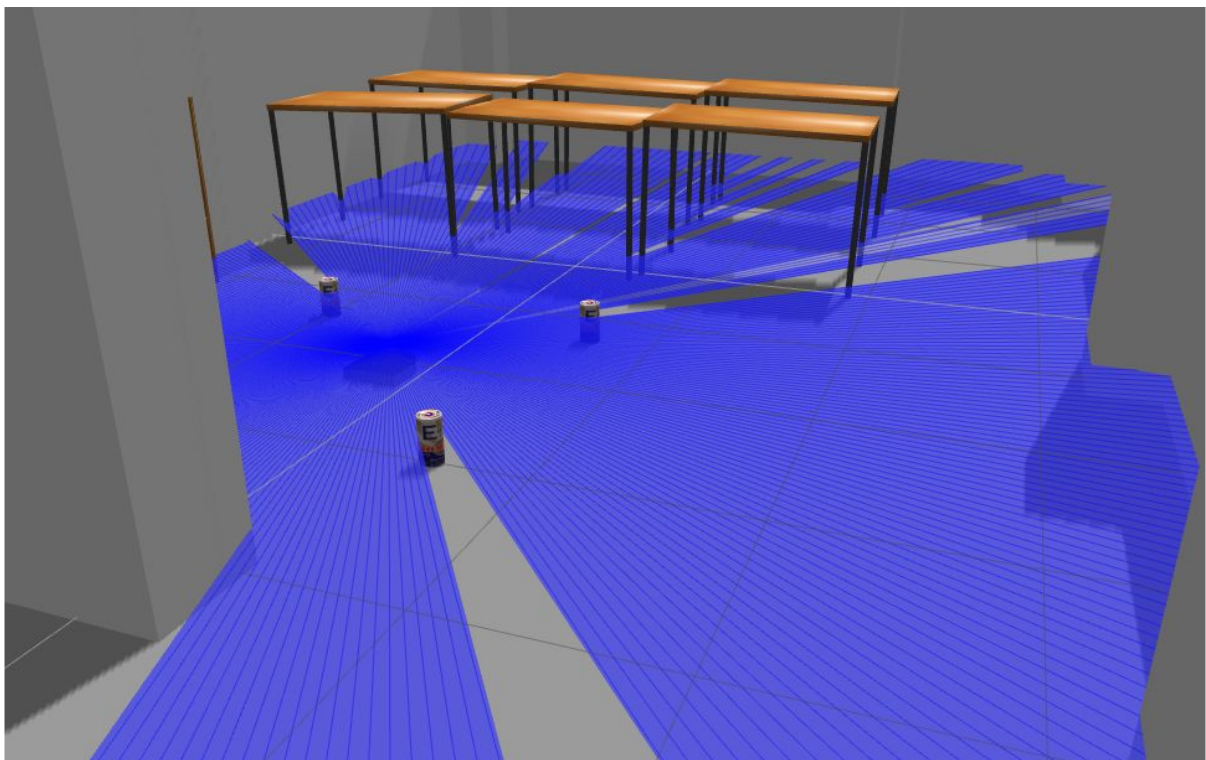


Figura 38: Captura de pantalla del robot en acció.

En la figura 38 podem observar com el làser “xocaria” amb els objectes de l’entorn i com ens seria de fàcil detectar els landmarks naturals.

Ja que moltes vegades utilitzem un entorn controlat, en el meu disseny de l’aulari he afegit unes llaunes que tenen una forma semblant als cilindres que s’utilitzen per simular landmarks artificials. Podem veure a la figura 39 que utilitzant la lectura del làser podem determinar les ubicacions dels landmarks.

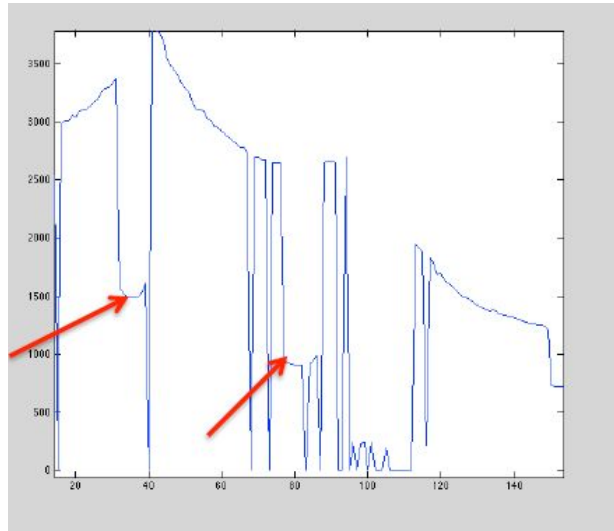


Figura 39 : Plot dels 360° de la lectura del làser en un instant de temps concret.

A la figura 39 es mostren els landmarks artificials cilíndrics detectats pel làser del Neato XV. Es tracta d’un *plot* de *matlab* de totes les lectures (360°) en un instant de temps del Neato movent-se en un entorn amb 4 cilindres.

5. Conclusions

5.1. Objectius aconseguits

Els objectius del treball han estat assolits de manera satisfactòria. En el treball s'han realitzat estudis del comportament del robot, els quals han servit per adequar el simulador el més semblant possible a la realitat. El plugin del simulador ha resultat ser un èxit. S'ha pogut realitzar aquest plugin tal i com seria la connexió amb el robot, ja que només canviant el port de connexió podem connectar-nos al simulador.

5.2. Funcionalitats

S'ha aconseguit fer funcionar de manera correcta les comandes bàsiques del robot. Aquestes comandes són les que s'utilitzen en les pràctiques de l'assignatura de robòtica. Entre elles la funció "SetMotors", que permet fer avançar el robot en l'entorn. La funció "GetMotors" que permet obtenir les dades dels encoders del robot. I la funció "GetLDSScan", que dona els valors del sensor làser 360°.

5.3. Desenvolupament de l'entorn

S'ha realitzat la feina de documentar l'entorn de treball, per tal de poder realitzar-ne una versió en 3 dimensions. Aquesta versió és molt fidel a la realitat.

5.4. Valoració personal

Personalment, la realització d'aquest treball ha estat molt gratificant. Arribar a fer un simulador que podrà ser usat per els alumnes d'una assignatura és un bon concepte.

Trobar tota la informació necessària per realitzar el treball ha estat una feina molt laboriosa. Cal afegir que la documentació del sensor làser és escassa, i aconseguir fer-lo funcionar ha estat una de les parts més difícils del treball.

Estic molt content d'haver realitzat aquest treball, ja que m'ha ajudat molt a seguir el meu estudi sobre la robòtica.

6. Apèndix

6.1. Definicions

- **Sensors** → objecte capaç de detectar magnituds físiques i químiques, anomenades variables d'instrumentació, i transformar-les en variables elèctriques.
- **Landmark** → Estructura fàcilment reconeixible. En l'àmbit de la robòtica es refereix a cantonades o punts per on el robot pugui identificar on està.
- **Encoder** → Tipus de sensor que permet saber la distància que ha recorregut una roda.
- **Raspberry Pi2** → Plataforma hardware programable.
- **Socket** → designa un concepte abstracte pel qual dos programes (possiblement situats en ordinadors diferents) poden intercanviar qualsevol flux de dades, generalment de manera fiable i ordenada.
- **Bumpers** → Paratxocs. En l'àmbit de la robòtica s'usa com a sensor per saber si el robot ha xocat.
- **Parser** → També anomenat analitzador sintàctic. És una de les parts d'un compilador que transforma la seva entrada en un arbre de derivació.
- **API** → *Application Programming Interface*, és el conjunt de subrutines, funcions i procediments (o mètodes, en la programació orientada a objectes) que ofereix certa biblioteca per ser utilitzat per un altre programari com una capa d'abstracció.
- **GazeboSim** → Simulador robòtic de físiques avançades.

6.2 Users manual simulador

Aquest és el Manual d'usuari per tal de que els alumnes puguin utilitzar el plugin.

Neato Plugins

```
Last login: Tue Mar 29 09:17:36 on ttys001
MacBook-Pro-de-Guillem:~ Guillem$ ls
Applications      Documents         Dropbox           Movies            NetBeansProjects  Public
Desktop           Downloads        Library          Music             Pictures          VirtualBox VMs
MacBook-Pro-de-Guillem:~ Guillem$ cd .gazebo/
MacBook-Pro-de-Guillem:.gazebo Guillem$ ls
CMakeLists.txt  client-11345     diagnostics      gui.ini           models            ogre.log        server-11345
MacBook-Pro-de-Guillem:.gazebo Guillem$ cd models/
MacBook-Pro-de-Guillem:models Guillem$ ls
asphalt_plane   bookshelf        brick_box_3x1x3  kinect            pioneer2dx        submarine        willowgarage
ball_bearing    bowl             create           my_robot          polaris_ranger_ev  submarine_buoyant
beer            breakable_test   hokuyo           neato             polaris_ranger_xp900  turtlebot
MacBook-Pro-de-Guillem:models Guillem$ cd neato/
MacBook-Pro-de-Guillem:neato Guillem$ ls
meshes          model.config     model.sdf        plugin
MacBook-Pro-de-Guillem:neato Guillem$ cd plugin/
MacBook-Pro-de-Guillem:plugin Guillem$ ls
CMakeCache.txt      CommandParser.cpp      NonBlockingServerSocket.cpp      Socket.h      libneato_control.dylib
CMakeFiles          CommandParser.hpp      NonBlockingServerSocket.hpp      SocketException.h      neato_control.cc
CMakeLists.txt      Makefile              Socket.cpp                      cmake_install.cmake    sockets_cpp
MacBook-Pro-de-Guillem:plugin Guillem$ cd sockets_cpp/
MacBook-Pro-de-Guillem:sockets_cpp Guillem$ ls
ClientSocket.cpp      NonBlockingServerSocket.cpp      ServerSocket.h      SocketException.h      simple_server
ClientSocket.h         NonBlockingServerSocket.hpp      Socket.cpp          simple_client          simple_server_main.cpp
ClientSocket.o         NonBlockingServerSocket.o        Socket.h            simple_client_main.cpp  simple_server_main.o
Makefile              ServerSocket.cpp              Socket.o            simple_client_main.o
MacBook-Pro-de-Guillem:sockets_cpp Guillem$
```

EDIT

You need to change some path on the plugins:

On the model.sdf doc, you need to change the relative path on this line:

```
<plugin name="model_push"
filename="/Users/Guillem/.gazebo/models/neato/plugin/libneato_control.so">
```

MAKE

To run your program, is needed compile the plugins:

On your *command line/bash/terminal* inside *models/neato/plugins/sockets_cpp* you need to execute *make*.

```
MacBook-Pro-de-Guillem:sockets_cpp Guillem$ make
make: Nothing to be done for 'all'.
MacBook-Pro-de-Guillem:sockets_cpp Guillem$
```

Also on the plugins folder you need to:

Inside the plugins folder: *models/neato/pligins*

- ***cmake CMakeLists.txt***
- ***Make***

```

MacBook-Pro-de-Guillem:plugin Guillem$ cmake CMakeLists.txt
-- Boost version: 1.60.0
-- Found the following Boost libraries:
--   thread
--   signals
--   system
--   filesystem
--   program_options
--   regex
--   iostreams
--   date_time
-- Boost version: 1.60.0
-- Looking for OGRE...
-- Found Ogre Cthugha (1.7.4)
-- Found OGRE: optimized;/usr/local/Cellar/ogre/1.7.4/lib/libOgreMain.dylib;debug;/usr/local/Cellar/ogre/1.7.4/lib/libOgreMain.dylib
-- Looking for OGRE_Paging...
-- Found OGRE_Paging: optimized;/usr/local/Cellar/ogre/1.7.4/lib/libOgrePaging.dylib;debug;/usr/local/lib/libOgrePaging.dylib
-- Looking for OGRE_Terrain...
-- Found OGRE_Terrain: optimized;/usr/local/Cellar/ogre/1.7.4/lib/libOgreTerrain.dylib;debug;/usr/local/lib/libOgreTerrain.dylib
-- Looking for OGRE_Property...
-- Found OGRE_Property: optimized;/usr/local/Cellar/ogre/1.7.4/lib/libOgreProperty.dylib;debug;/usr/local/lib/libOgreProperty.dylib
-- Looking for OGRE_RTShaderSystem...
-- Found OGRE_RTShaderSystem: optimized;/usr/local/Cellar/ogre/1.7.4/lib/libOgreRTShaderSystem.dylib;debug;/usr/local/lib/libOgreRTShaderSystem.dylib
-- Configuring done
CMake Warning (dev):
  Policy CMP0042 is not set: MACOSX_RPATH is enabled by default. Run "cmake
  --help-policy CMP0042" for policy details. Use the cmake_policy command to
  set the policy and suppress this warning.

  MACOSX_RPATH is not specified for the following targets:

    neato_control

This warning is for project developers. Use -Wno-dev to suppress it.

-- Generating done
-- Build files have been written to: /Users/Guillem/.gazebo/models/neato/plugin
MacBook-Pro-de-Guillem:plugin Guillem$ make

```

```

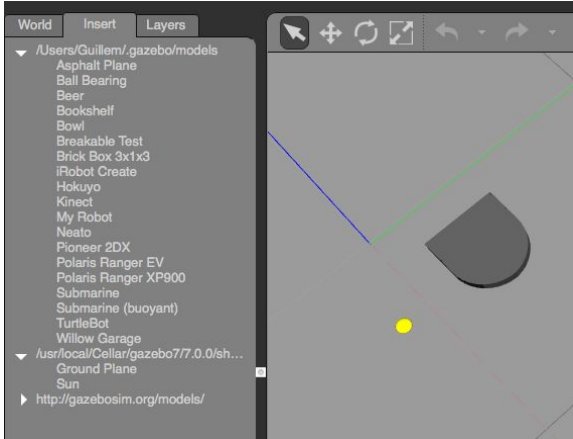
[ 25%] Building CXX object CMakeFiles/neato_control.dir/neato_control.cc.o
/Users/Guillem/.gazebo/models/neato/plugin/neato_control.cc:288:39: warning:
format string is not a string literal (potentially insecure)
[-Wformat-security]
    sprintf(buff, temp.c_str());
                                ^~~~~~
/Users/Guillem/.gazebo/models/neato/plugin/neato_control.cc:298:39: warning:
format string is not a string literal (potentially insecure)
[-Wformat-security]
    sprintf(buff, temp3.c_str());
                                ^~~~~~
2 warnings generated.
[ 50%] Linking CXX shared library libneato_control.dylib
[100%] Built target neato_control
MacBook-Pro-de-Guillem:plugin Guillem$

```

Plugins use on command line

Neato Connect

Open 2 terminals

SERVER TERMINAL	CLIENT TERMINAL
<code>cd .gazebo/models/neato/plugin/sockets_cpp</code>	<code>cd .gazebo/models/neato/plugin/sockets_cpp</code>
<code>/gazebo</code>	
Insert a Neato Robot on the World 	
	<code>./simple_client 50000</code>
Return: New connection to client!	

Now you're connected to the Neato robot.

Neato commands

Now you're able to give commands to the robot, **sending the right command on the SERVER terminal.**

SetMotor

You can use it to move the robot.

LWheelDist [mm]

RWheelDist [mm]

Speed [mm/s]

SetMotor LWheelDist x RWheelDist x Speed x

Returns: -

GetMotors

Returns the movement value of the wheels.

GetMotors LeftWheel RightWheel

Returns:

LeftWheel_MaxPWM,65536 LeftWheel_PWM,-858993460 LeftWheel_mVolts,1310
LeftWheel_Encoder,0 LeftWheel_PositionInMM,0 LeftWheel_RPM,- 13108
RightWheel_MaxPWM,65536 RightWheel_PWM,-858993460 RightWheel_mVolts,1310
RightWheel_Encoder,0 RightWheel_PositionInMM,0 RightWheel_RPM,-13108

GetLDSScan

Returns the range from the laser sensor scan.

GetLDSScan

Returns:

AngleInDegrees,DistInMM,Intensity,ErrorCodeHEX 0,221,1400,0 1,223,1396,0
2,228,1273,0 (. . .) 359,220,1421,0 ROTATION_SPEED (in Hz, Floating
Point),5.00

6.3. Referencies

1. Patricia Puentes. *eldiario.es: Los robots domésticos llegan a casa*. Barcelona, 19/01/2015. [Consulta: 2/03/2016]. Disponible a: http://www.eldiario.es/turing/casa-inteligente_0_347515619.html
2. "Adaptive Manipulation of a Hybrid Mechanism Mobile Robot". P. Moubarak, P. Ben-Tzvi. *IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. Montreal, Canada, 2011, pp. 113 - 118
3. "Cnet" [pàgina web], Coline West McDonald. <https://www.cnet.com/products/neato-xv-essential-robot-vacuum/review/> [Consulta 15 de Maig de 2016]
4. "Raspberry Pi 2" [pàgina web], <https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/> [Consulta 15 de Maig de 2016]
5. "The Money Converter" [pàgina web]. <http://themoneyconverter.com/ES/USD/EUR.aspx> [Consulta 18 d'Octubre de 2016]
6. "Idealista" [pàgina web], búsqueda de oficinas a Barcelona, mitja de les rellevants. <https://www.idealista.com/alquiler-oficinas/barcelona-barcelona/> [Consulta 15 de Maig de 2016]
7. "Neato Robotics", video de demostració. <https://www.neatorobotics.com/es/>