

Mecatrónica – 2013/2014

ROS - Detector de metais

Projecto

Luís Rocha

2010127532

José Medeiros

2010129934

Introdução

Neste trabalho foi incorporado um sensor inductivo ao robô Roomba para a detecção de minas terrestres através do uso de ROS. Para o controlo do robô foi necessário fazer uso de um API[5] que permitisse a comunicação com o mesmo e a plataforma ROS. Com este foi possível controlar os movimentos do robô através do teclado usando um terminal.

Quando na presença de metais e usando os dados devolvidos pelo sensor - proximidade e frequência - é possível determinar se o robô se encontra perante uma mina. Com base neste princípio é possível detectar a presença de minas.



(a) Robot Roomba



Bobine LDC1000 Interface USB - μ MSP430

(b) Componentes do LDC100EVM

Figura 1: Roomba e LDC100EVM com os componentes

Todo list

■ Colocar as duas unidas	11
■ Colocar o pseudo Código	14
■ Colocar os comandos para executar, as portas sudo chmod e os launchs	14
■ nos das tfs e nos a comunicarem- imagem	14
■ Colocar a simulação rviz	15
■ Robo a andar com rviz activo	15
■ Foto a controlar com o keyboard e a detectar minas	15

Indíce

1	Lista de acrónimos	5
2	Material disponível	5
3	Descrição do problema	6
4	Conhecimentos necessários	6
5	Solução proposta	7
5.1	Hardware	7
5.1.1	Introdução ao sensor	8
5.1.2	Arquitectura	9
5.1.3	Implementação	10
5.2	Software	13
5.2.1	Como executar	13
5.2.2	Pseudocódigo	14
5.2.3	Utilização do ROS	14
5.2.4	Comunicação com o sensor	14
5.2.5	Protocolo VNC	15
6	Resultados	15
7	Conclusões	15

Indice de figuras

1	Roomba e LDC100EVM com os componentes	1
2	Placa PCB + Sensor	7
3	Estrutura do robô Roomba com o sensor LDC100EVM	7
4	Sensor LDC1000EVM	8
5	Esquemático do sensor	8
6	Imagen real do robô	9
7	LCD1000EVM com indicação das perfurações	10
8	LCR Meter 186	11
9	Medição da indutância da bobine e da capacidade do condensador	11
10	Resistência do paralelo (R_p) na ausência de objectos condutores	12
11	Resistência do paralelo (R_p) na presença de objectos condutores	12
12	Registros do sensor LDC1000EVM	13
13	Protocolo VNC	15

1 Lista de acrónimos

ROS

Robot Operation System - Sistema Operativo para Robos

LRC Meter

Aparelho que mede indutância, capacidade e resistência de um componente[2]

PCB

Printed Circuit Board - Placa de Circuito Impresso

API

Application Programming Interface - Interface de Programação de Aplicações

USB

Universal Serial Bus

VNC

Virtual Network Computing

2 Material disponível

Computadores

Notebook

Netbook

Componentes/Aparelhos Electrónicos

Bobina de 1.71 mH

Condensador de 3.09 nF

LRC Meter - Modelo 186

Robô Detector de Minas

Robô Roomba 555

Detector de Metais - LDC1000EVM

API/Wrapper para comunicação com o Roomba

Acessórios

Teclado Wireless

USB Hub com quatro portas

3 Descrição do problema

Pretende-se detectar minas terrestres através do uso de um robô Roomba e de um sensor indutivo LCD1000EVM capaz de medir a proximidade e a frequência associada á presença de um metal nas imediações. Este sensor é composto por uma bobine em PCB, um LDC1000 que faz uso da bobine e um microcontrolador - MSP430, tal como se pode observar na figura 1b.

Para ser possível a concretização deste projecto é necessário implementar uma interface de comunicação entre o computador e o sensor para a aquisição de dados. Esta interface vai ser baseada num protocolo de comunicação que terá que ser o mesmo que é usado pelo microcontrolador. Será portanto necessário entender bem o protocolo visto que toda a base deste projecto se baseia neste sensor.

Após a interligação entre o sensor e o computador através de uma porta USB é necessário combinar o robô Roomba com o sensor utilizando a plataforma ROS¹. Para isso é obrigatório fazer uso de transformadas ($tf's$) para referenciar o sensor ao eixo/referencial em relação ao referencial principal do robô. Mais á frente no relatório explicaremos melhor o conceito e a maneira como interligamos as transformações na nossa implementação. Este passo é importantíssimo para que se possa calcular as posições onde o sensor detecta minas.

4 Conhecimentos necessários

Após analisarmos os requisitos deste projecto chegámos á conclusão de que era necessário familiarizarmos com a plataforma ROS. Sendo assim seguimos os tutoriais existentes no website WikiRos[1]. Após concluidos os tutoriais, ficamos aptos a usar ROS usando C++ como linguagem de programação.

Os seguintes conceitos/utilidades foram usados no nosso projecto, a sua explicação breve apresenta-se abaixo:

Tf

Estruturação dos referenciais do robô assim como as transformações necessárias entre os diferentes componentes² do robô para que se possa saber as posições de qualquer elemento num dado instante.

Marker

Estrutura de dados que permite usar as bibliotecas do *rviz* para marcar minas no mapa. A posição das minas é obtida pela transformação da *tf* do sensor em relação ao referencial base do robô.

ArrayMarker

Guardamos os markers (minas) que detectamos num array de markers. Para tal é necessário fazer *push_back* de um marker para o array quando é detectada a presença de uma mina.

Serial Port

Protocolo de comunicação usado para comunicar com o sensor através de uma porta USB.

¹Distribuição Hydro

²Sensores, plataformas, rodas, entre outros

5 Solução proposta

5.1 Hardware

Como podemos ver na figura 2, foi soldado numa placa PCB o condensador e foram feitas as ligações necessárias de modo a facilitar a mudança de bobinas e para garantir que o circuito está em contacto com os componentes. Foi ainda soldado ao sensor dois fifos conectores para tornar mais fácil o seu uso.



Figura 2: Placa PCB + Sensor

Em termos de ligações é mais facil de compreender na figura 3, onde é possível ver a ligação dos componentes ao Roomba e ao sensor. O computador vai comunicar com o Roomba e receber dados do sensor. Através da plataforma ROS é feito o controlo do Roomba por teclado e feita a interligação dos dados enviados pelo sensor e o robô.

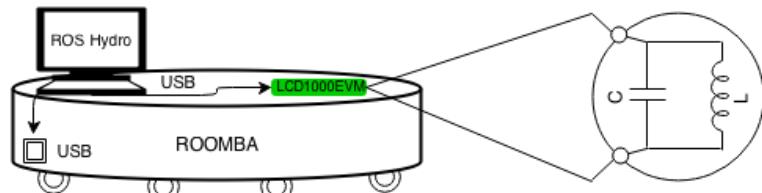


Figura 3: Estrutura do robô Roomba com o sensor LDC100EVM

5.1.1 Introdução ao sensor

O LDC100EVM é um sensor indutivo que permite medir a presença de metais retornando duas variáveis: posição do metal e a frequência de oscilação. É um circuito integrado que contém um microcontrolador MSP430 que é usado para fazer a interface entre o LDC (sensor) e um computador. Este integrado vêm com uma bobine que pode ser mudada se se partir a PCB na zona a tracejado e colocando ai uma bobine á nossa escolha. É possivel ver na figura 4 a zona a tracejado:



Figura 4: Sensor LDC100EVM

Para dimensionar a bobine é necessário satisfazer duas condições para que o sensor possa funcionar na gama ideal:

1. A frequência de ressonância deve estar compreendida entre 5 kHz e 5 MHz
2. A resistência de ressonância³ deve estar compreendida entre 798Ω e $3.93\text{ M}\Omega$

Para o cálculo da frequência e a respectiva resistência usa-se as seguintes fórmulas:

$$f_{\text{ressonância}} = \frac{1}{2\pi\sqrt{LC}} \quad (1)$$

$$R_p = \frac{1}{R_s} \frac{L}{C} \quad (2)$$

Abaixo encontra-se representado o esquemático do circuito que dá origem ás equações (1) e (2) acima descritas:

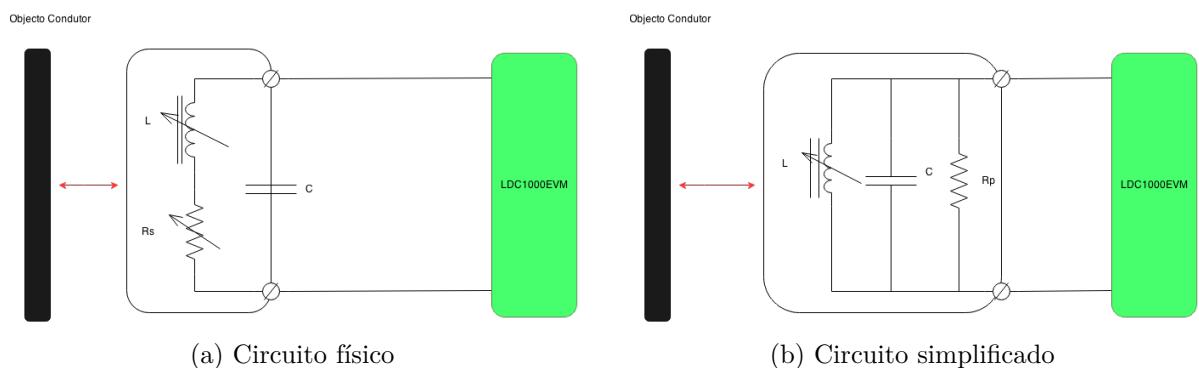


Figura 5: Esquemático do sensor

É importante referir que o circuito presente na figura 5a é o circuito real do sensor e o da figura 5b é uma simplificação que em termos prácticos permite ao sensor variar a sua gama de funcionamento afinando o valor minimo e máximo de R_p . Este processo faz com que o sensor possa discriminar diferentes tipos de metal se bem afinado e projectado.

³Simboliza as perdas associadas ás correntes de Eddy

5.1.2 Arquitectura

Em termos de hardware para este trabalho foi utilizado um computador que vai controlar o Roomba e o sensor LCD1000EVM. No sensor foi ainda substituída a bobine que vinha com o mesmo por uma bobine e um condensador de valores por nós pretendidos.

O robô Roomba encontra-se conectado por usb a um computador que o controla. Este computador encontra-se ainda ligado a um sensor LCD1000EVM que por sua vez tem ligado a si uma bobine e um condensador por nós projectados e adaptados.

Na figura 6 é visível o robô já adaptado à nova bobine. Foi necessário criar uma estrutura que suportasse a bobine mas que mantivesse a estabilidade do robô, para tal é necessário o uso de contra-pesos como é visível.



Figura 6: Imagem real do robô

5.1.3 Implementação

Como foi anteriormente explicado, o circuito pode ser alterado e dimensionado para se trabalhar na gama de valores desejada. A seguir iremos explicar os passos necessários para implementar o novo circuito e adaptá-lo ao sensor.

1. O primeiro passo será escolher simplesmente a nova bobine por nós pretendida e escolher um condensador que permita cumprir as condições impostas à frequência de ressonância(1) e à resistência de ressonância(2).
2. O próximo passo será com o auxílio de um “LCR meter”, colocando as pontas deste medidor em paralelo com o paralelo do condensador e da bobine que escolhemos têmos que anotar o valor da resistência R_p lido no medidor, na ausência de qualquer objecto magnético.
3. Neste passo temos que colocar um objecto magnético o mais próximo possível da bobine e anotar o novo valor de R_p .
4. É necessário anotar também a frequência de varrimento que se encontra definida no medidor.
5. No passo seguinte é necessário actualizar os registos do sensor para que se adapte ao novo circuito. Para modificar os registos é necessário conectar o sensor a um computador coonduoresm windows e utilizando a gui application fornecida pela texas instruments na secção de alteração dos registos teremos que modificar três registos. O valor do registo R_{pMAX} deve ser alterado para o dobro do valor anotado na ausência de objectos c, como os valores disponíveis são discretos deve ser escolhido o mais próximo deste valor por excesso. O valor do registo R_{pMIN} deve ser alterado para metade do valor anotado na presença de um objecto condutor, como os valores disponíveis são discretos deve ser escolhido o mais próximo deste valor por defeito. O registo da freqüência deve ser alterado para no máximo 80% do valor da frequência de varrimento.
6. Por fim é necessário partir o sensor onde diz “coil perforation” na figura 7 e soldar o paralelo da bobine e do condensador no sítio onde diz “conections for custom LC tank” na figura 7.



Figura 7: LCD1000EVM com indicação das perfurações

Com o auxílio do medidor “LCR meter” figura 8 conseguimos adaptar o sensor à nova bobine. Para cumprir o primeiro passo referido anteriormente, primeiro medimos a indutância da nova bobine e de seguida escolhemos um condensador e medimos a sua capacitância como mostra a figura 9.



Figura 8: LCR Meter 186

Tivemos que garantir que o novo circuito se encontra na gama da frequência de ressonância, como tal utilizando a (1) calculámos a nova frequência de ressonânciā e obtivemos:

$$f_{\text{ressonância}} = \frac{1}{2\pi\sqrt{LC}} = \frac{1}{2\pi\sqrt{1.7098 \times 10^{-3} \times 3.0863 \times 10^{-9}}} = 69.283 KHz \quad (3)$$

Verifica-se que a nova frequência de ressonância se encontra entre $5 KHz$ e $5 MHz$

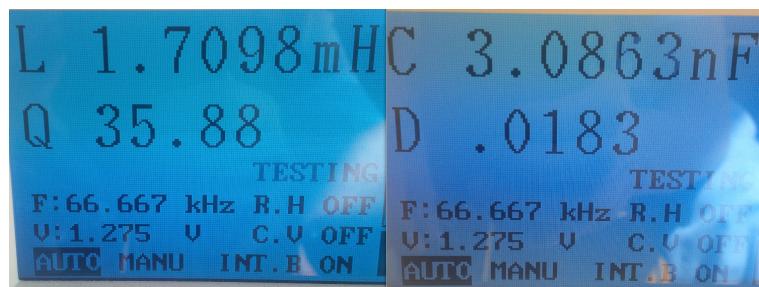


Figura 9: Medição da indutância da bobine e da capacidade do condensador

De seguida anotámos o valor de R_p na ausência e na presença de objectos condutores como é visível nas imagens 10 e 11 respectivamente. Foi ainda anotada a frequência de varrimento que é visível em qualquer uma das imagens seguintes, sendo o seu valor $66.667 KHz$.

Colocar as duas unidas



Figura 10: Resistência do paralelo (R_p) na ausência de objectos condutores

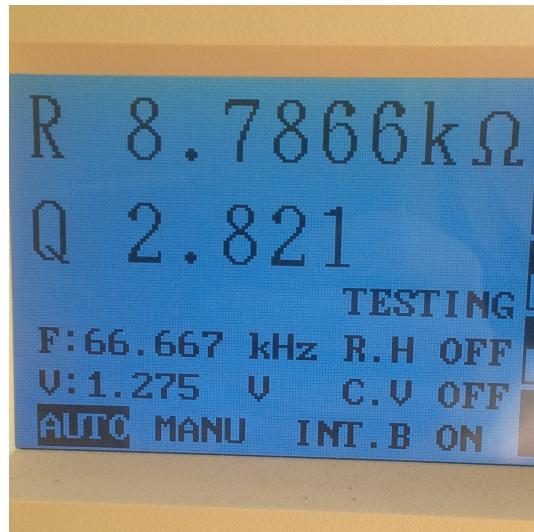


Figura 11: Resistência do paralelo (R_p) na presença de objectos condutores

Garantimos assim que também a resistência de ressonância se encontra dentro da gama de valores do referidos anteriormente. Por fim, fizémos a alteração dos registos do sensor para se adaptar aos novos valores, pelo que foram modificados os seguintes registos :

1. RP_{MAX} foi alterado para $38.785K\Omega$, valor que foi aproximado por excesso como era exigido.
2. RP_{MIN} foi alterado $4.309K\Omega$, valor que foi aproximado por defeito como era exigido.
3. Sensor frequency foi alterado para $52.231KH_z$, valor que foi aproximado para o valor mais próximo de 80% da frequência de varrimento, por defeito.

Os registos que foram alterados encontram-se a vermelho na figura 12. Com todos os passos cumpridos o novo sensor ficou pronto a ser utilizado.

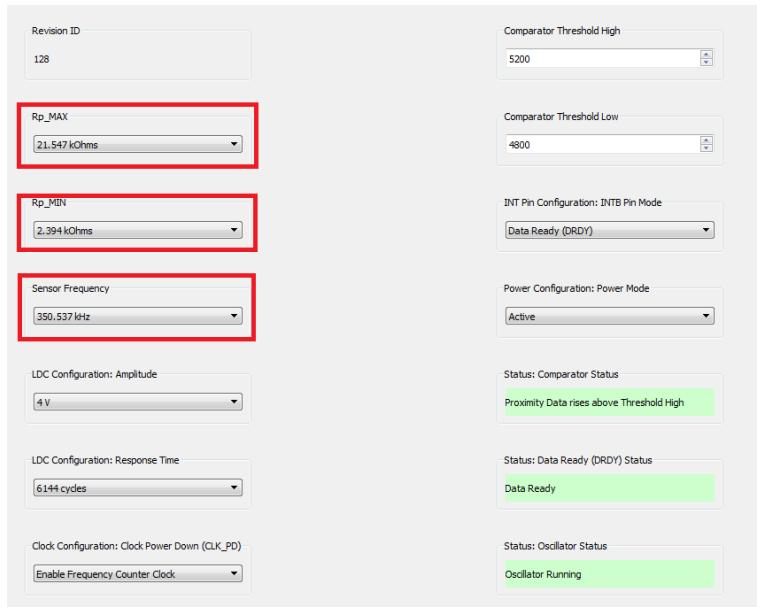


Figura 12: Registos do sensor LDC1000EVM

5.2 Software

5.2.1 Como executar

Em termos de aplicação ROS, elaboramos dois tipos de programas. O primeiro contém um nó que lê valores do sensor e publica esses dados usando mensagens e um outro nó que recebe esses dados e imprime para o ecrã. O segundo usa um ó que publica os dados por mensagens e um outro nó recebe esses dados e analisa se se trata de uma mina e em caso afirmativo marca-a num mapa desenhando um cubo de pequenas dimensões na posição em que detectou a mina. Abaixo encontram-se os passos necessários para os executar:

Primeiro Código:

1. roscore
2. rosrun sensor le_sensor threshold(int)
3. rosrun sensor recebe_dados

Segundo Código:

1. roscore
2. roslaunch roomba Bringup roomba_555.launch
3. roslaunch sensor sensor.launch
4. rosrun sensor le_sensor threshold (int)
5. rosrun sensor points_sensor

6. rosrun roomba_teleop keyboard_teleop.launch

7. rosrun rviz rviz

Nota I: Treshold(int) é o valor para o qual o programa identifica como sendo uma mina.

Nota II: No rviz é preciso carregar o modelo do robot, as $tf's$, os markers e o array de markers.

Colocar o pseudo Código

Colocar os comandos para executar, as portas sudo chmod e os launchs

nos das tfs e nos a comunicarem- imagem

5.2.2 Pseudocódigo

5.2.3 Utilização do ROS

Utilizámos o Robot Operating System (ROS) como plataforma de comunicação entre os dois periféricos por nós utilizados e o computador. Com o ROS é possível implementar bibliotecas de cliente ROS como roscpp, rospy ou roslisp. Por outras palavras significa que facilmente se consegue adaptar programas criados em outras línguas (de entre C++, Python e LISP). No nosso caso em específico adaptámos o programa de leitura do sensor em cpp para a plataforma ROS. Numa segunda parte através do API disponibilizado pelo Gonçalo Cabrita[4] e pelo Bruno Gouveia[3] conseguimos controlar o Roomba pelo teclado do computador. Através da leitura dos encoders e do controlo dos motores foi possível controlar a velocidade angular e linear do robô.

Utilizando os nós do ROS foi possível assim controlar o robô e utilizá-lo para a deteção e marcação nas minas. De uma forma mais detalhada foi usada como referência a transformada "ODOM" que nos fornece a posição inicial em que o robô se encontra, com o nó entre esta transformada e o base link do Roomba sabe-se sempre a posição relativa à posição inicial. De seguida criando o nó da transformada sensor(criada por nós) é assim possível, com a informação devolvida pelo sensor, saber se o robô se encontra na presença de uma mina. Caso seja verificada a presença de uma mina são enviadas as coordenadas (x,y,z) para uma função "markers" que irá marcar no mapa as minas em relação a "ODOM", ou seja em relação à posição inicial do robô.

5.2.4 Comunicação com o sensor

A primeira parte deste trabalho foi concluir que o sensor utiliza porta série como protocolo de comunicação quando comunica com o computador. Como tal, para facilitar o uso futuro deste sensor criámos código para as leituras do sensor em três plataformas diferentes de programação. Adaptámos por isso o nosso código às plataformas [C++](#), [Matlab](#) e [Python](#). Estes três códigos encontram-se mais a baixo na tabela(Fazer referencia à tabela).

Analisando as partes mais importantes do código, o primeiro passo é definir a porta série pela qual estamos a comunicar, no nosso caso em particular foi a porta "COM9" em *Windows* e "/dev/ttyACM0" em *Linux*. De seguida definimos o tipo de controlo de fluxo, que neste caso é do tipo "software flow control". É necessário definir a frequência de comunicação, Baud Rate, que foi definida por nós a 9600. O próximo passo é definir a paridade e o número de bits da comunicação,

neste caso não existe paridade e são utilizados 8bits na comunicação. Tal como na maioria dos dispositivos electrónicos na comunicação por porta série apenas é necessário de 1 "stop bit", foi ainda definido o tempo de espera para leitura máximo("Timeout") a 5 segundos. Definimos o modo de leitura contínua para funcionar de forma assíncrona. Por fim ao escrever na porta série o comando 0x33(hexadecimal) ou 3(ASCII), iniciamos a stream de dados.

5.2.5 Protocolo VNC

Este protocolo baseia-se no conceito de partilha remota de um computador. Para isso existe um computador que partilha o monitor e, se o utilizador permitir, os controlos do mesmo (servidor) e um ou mais computadores (clientes) que se conectam ao servidor e são capazes de realizar as acções permitidas pelo mesmo .

Como a base do VNC foi construída/projectada sobre o protocolo TCP/IP, este é capaz de funcionar via internet. É usado em aulas interactivas onde um professor permite a visualização do seu monitor e os alunos conectam-se podendo assim visualizar os passos do professor.

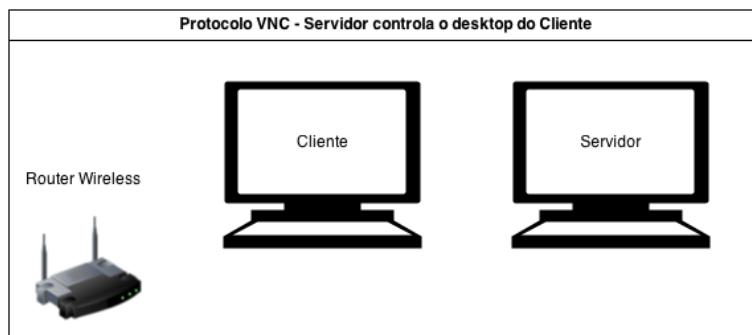


Figura 13: Protocolo VNC

Nota: Rápido e eficaz numa rede local com boa largura de banda.

6 Resultados

Colocar a simulação rviz

Robo a andar com rviz activo

Foto a controlar com o keyboard e a detectar minas

7 Conclusões

Com este projecto foi possível adaptar o robô ROOMBA para a deteção de minas com o auxílio do sensor LCD1000EVM. Numa primeira parte do projecto foi criada a interface para que o sensor pudesse ser utilizado em Linux. Com o auxílio da plataforma ROS (Robot Operatng System) foi possível fazer a comunicação dos três periféricos (computador, robô e sensor) esta plataforma permitiu ainda criar o controlo por teclado da velocidade e direcção do robô. Por fim

foi possível adaptar o sensor para uma bobine por nós desejada.

Este projecto permitiu perceber a potencialidade tanto da plataforma ROS como do sensor LCD1000EVM. A grande vantagem da plataforma ROS é que no futuro muito facilmente qualquer pessoa poderá adaptar este sensor a outros robôs ou outros sistemas robóticos, é a vantagem do sistema de transformadas em que o ROS trabalha. O sensor LCD1000EVM tem a grande vantagem de com alguns pequenos passos se poder modificar para qualquer bobine que o utilizador pretenda criando-se assim um sensor à medida do que o utilizador precisa.

Referências

- [1] Robot operating system. <http://wiki.ros.org/>, 2007. Hydro Edition.
- [2] Lrc meter, 2014. http://en.wikipedia.org/wiki/LCR_meter.
- [3] Gouveia Bruno. frames id. https://github.com/ras-sight/hratc2014_entry_template, 2014.
- [4] Cabrita Gonçalo. cereal port. https://github.com/goncabrita/cereal_port, 2013.
- [5] Cabrita Gonçalo. Roomba api. <http://wiki.ros.org/Robots/Roomba>, 2013.