David Teles IST181053, Leonor Silva IST181155, Filipe Silva IST176985, Pedro Soares IST181585

[[1]](#footnote-1)

Mapping using a Laser Range Finder(SA 1S 2017/18)

*Abstract*—Este documento serve como relatório final do projeto M1 – “Mapping with Laser Range Finder” para a cadeira de Sistemas Autónomos, lecionada no primeiro semestre do ano letivo 2017/18. Descreve o mapeamento com Occupancy Grid onde se utilizou um robô móvel com um laser para ser recolher os dados.

# INTRODUCTION

O nosso trabalho é sobre mapeamento com o Laser Range Finder. Este trabalho aborda um tema fundamental para robôs móveis, que é o facto de estes necessitarem de criar um mapa do ambiente onde estam para se poderem localizar e concretizar as suas tarefas, como por exemplo deslocarem-se de A para B.

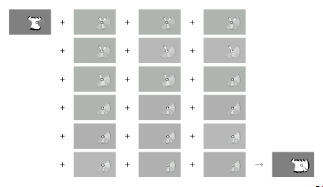
 Para o mapeamento usamos Occupancy Grid Maps, que consistem em representar o ambiente em cubos os quais contêm informação sobre a possibilidade de estarem ocupados ou não. Claro que este mapeamento não é completamento certo, no entanto se se escolher um tamanho pequeno para cada célula este mapa é bastante preciso. Assim o sensor recolhe os dados do ambiente os quais são processados, alterando a probabilidade de cada célula estar ocupada de modo a inferir o mapa que depois pode ser usado para controlo e navegação. Este update de valores é feito através de técnicas probabilísticas e é dado pelo conjunto de mapeamentos de cada frame (fig.1). Esta técnica não permite o laser saber a sua posição global, portanto para isso usamos o sistema de motion caption.

Figura 1- Composição do mapeamento

O funcionamento do laser baseia-se em medir a distância aos obstáculos através da emissão de raios laser dentro do seu raio de visão, que no nosso caso era de 270˚. No entanto, estes lasers estão sujeitos aos efeito de especularidade, ou seja, caso o raio laser encontre uma superfície lisa, o raio seja desviado. Logo, este raio pode não retornar ao laser ou então retornar como consequência de ter sido refletido várias vezes e, portanto, transmitindo dados falsos. Consequentemente, cria incerteza no mapa. Para além disso, pode haver informação redundante resultante, por exemplo, de o laser permanecer na mesma posição. Por informação redundante, entende-se informação que não contribui para o mapeamento, pois apenas reforça dados anteriormente adquiridos, podendo estes estar errados.

No caso do Occupancy Grid Map ao considerarmos as células independeste, ao invés de como um todo, onde o facto de uma célula estar ocupada aumenta a probabilidade de a célula ao lado também estar (como é o caso de uma parece) também pode resultar em mapas inconsistentes.

Algumas limitações deste tipo de mapeamento é o facto de que o ambiente tem de ser estático, pois como resulta do incremento de várias observações, se o ambiente muda invalida todas as observações feitas.

# Laser Range Finder

Para realizar o mapeamento usando um LRF, é necessário compreender o seu funcionamento de forma a desenvolver software que seja capaz de interpretar os dados obtidos por este em um formato de mais conveniente utilização para o objetivo final deste projeto.

Figura 2- Modelo simples do funcionamento de um LRF

Como é possível ver na Figura 1 o LRF realiza diferentes medições com diferentes ângulos obtendo um campo de visão que depende do equipamento em questão.

Para os testes realizados durante o desenvolvimento deste trabalho foi usado o Hokuyo 04LX-UG01 capaz de realizar medições em apenas 100ms/scan, a objetos de 200mm a 5600mm com uma resolução de 2mm, num ângulo de visão de 240° com uma resolução angular de 0.36°.

Como as medições realizadas por este equipamento se baseiam num ponto laser, foi considerado durante o resto deste trabalho que o “beam angle” é aprox. 0 (, o que por fim significa que que a cada leitura corresponde apenas um ponto no mapa.

Como estamos a obter s leituras do laser usando ROS, subscrevendo ao rostopic /scan, o software é bastante agnóstico ao equipamento que está na verdade a ser usado devido á componente de mensagens standard que o ROS proporciona.

Do tópico é obtida varia informação bastante importante para a manipulação dos dados posteriormente, sendo estes:

Frame\_id

Angle\_min

Angle\_max

Angle\_increment

Range\_min

Range\_max

Ranges[]

Frame\_id é importante para no futuro se realizar as transformações de Frame necessárias.

Angle\_min e angle\_max necessários para calcular o ângulo a que certa medição foi efetuada.

Angle\_increment é incremento de ângulo entre as várias leituras, este vai depender da resolução que por sua vez depende do LRF a ser usado. Quanto menor for este valor, significa maior resolução, mais leituras, mais tempo de processamento.

Range\_min e Range\_max importantes para descartar todos os valores fora do intervalo estre os dois valores.

Ranges é um vetor composto por todas a medições efetuadas (em m).

Para realizar o mapa usando estes dados é necessário criar uma Point\_Cloud e para isto foram usadas as seguintes equações:

No fim de processar uma mensagem complete já é possível obter uma ideia do mundo em redor do sensor, como é possível ver na Figura 2, mas este output é dinâmico e associado ao frame do laser pelo que precisamos de trocar os estes pontos para o frame do mapa, e para isto vamos usar TF.



Figura 3 Representação gráfica de uma point\_cloud criada usando uma leitura do LRF

# TF

O que é uma TF, uma composta por 2 valores, sendo estes a translação e rotação entre dois frames. Com estes dados é possível realizar a transformação de coordenadas entre vários frames de uma forma simples.

Para se obter a diferentes TFs usadas neste trabalho foi necessário subescrever ao rostopic /tf onde é possível obter 4 informações importantes:

* Frame\_id
* Child\_frame\_id
* Point
* Quaternion

Frame\_id e Child\_frame\_id permite-nos saber entre que frames é que as os dados são aplicáveis.

Point dá-nos a posicçao do Child\_frame\_id no Frame\_id e o Quaterninon dá-nos a rotação entre estes.

Como os dados usados durante os testes foram adquiridos usando o equipamento de Motion Caption as Tf usadas foram:

Base\_Link\_Laser ➡ Base\_Link ➡ Mocap\_marker➡ Map

Se o método de localização fosse baseado em Odometria seria necessário obter a tf entre o Base\_Link ➡ Odometry

Para uma manipulação mais pratica e intuitiva dos dos dados recorreu-se a uma matriz de rotação que foi obtida através do seguinte método:

Aplicando as equações apresentadas a cima é possível obter as coordenadas no novo frame, sendo apenas possível realizar as transformações entre dois frames adjacentes, foi necessário aplicar este processo 3 vezes até obter os pontos no referencial correto.

Sabendo agora a posicçaõ do laser e transformando toda a point\_cloud obtida anteriormente, recorrendo ao método anterior, é necessário calcular quais são as células atravessadas por cada leitura e para isto realizou-se a função de rasterização.

# Rasterization

Com o intuito de determinar o vector de células a actualizar probabilisticamente entre o laser (x1, y1) e a medição (x2, y2), desenvolveu-se uma função de rasterização de linha baseada no algoritmo de Bresenham com ligeiras diferenças para contemplar todas as células percorridas (Figura 4).

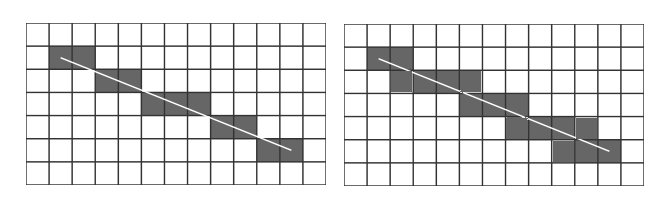


Figura Funcionamento do algoritmo de Bresenham (à esquerda) e da função de rasterização de linha usada (à direita)

Neste algoritmo é traçada uma recta entre (x1, y1) e (x2, y2) e assume-se como origem a coordenada do laser no mapa.  
Este algoritmo divide-se consoante o declive, negativo (quadrantes 2 e 4) ou positivo (quadrantes 1 e 3) mas funciona de maneira análoga para todos os quadrantes.

Vejamos o exemplo do primeiro quadrante:

Primeiramente são calculados o declive e a ordenada no origem e tem-se como altura de referencia y1:

Para todas as células entre x1 e x2 calcula-se a altura h:

Sendo c o iterador. Após calculado h e tendo yreff fazem-se o seguintes arredondamentos (difere do algoritmo de Bresenham):

O número de células verticais a actualizar em cada iteração será então:

Itera-se então pelas células verticais e adiciona-se os pontos

Ao vector de células percorridas. Sendo i o iterador do ciclo interior. Após este ciclo actualiza-se h:

Assim sucessivamente para todas as células seguintes até adicionar todo o vector de células percorridas pela linha do laser.

Para rectas () totalmente verticais foi criada uma excepção que ajusta o declive para:

ou

sendo res a resolução do mapa.

# Occupancy Grid Map

## Creation

A criação de um “occupancy grid map” começa pela alocação de uma matriz com dimensões suficientes para conter todo o espaço a mapear dividido equitativamente em células de dimensões iguais ( , onde i refere-se ao índice da célula na matriz ) como pode ser observado na figura 3.

Uma imagem com edifício, shoji

Descrição gerada com confiança alta

Figura Representação gráfica de uma matriz criada para albergar um mapa

Esta matriz é posteriormente inicializada com o valor probabilístico de 0.5 em cada célula , que indica a ausência de conhecimento relativa ao estado da célula ( ocupada ou livre ).

## Probabilistic Model

Tal como já referenciado anteriormente as células da matriz possuem uma valor probabilístico que indica a certeza , com que, de acordo com as medições efetuadas a célula está ocupada , esse modelo é apresentado em seguida :

## A partir da rasterização é obtido um “caminho” desde a célula onde se encontra o laser e a ultima célula medida por este , sem seguida o algoritmo classifica com uma probabilidade de 0.3 todas as células até á ultima célula medida ( estas células são consideradas livres pois encontram-se sempre a uma distância menor que a distância máxima de medição que o laser atinge ) , para a ultima célula é calculada a distância desta ao laser , caso essa distância seja inferior á máxima distância de medição que o laser atinge esta é considerada como ocupada e é lhe atribuída uma probabilidade de 0.6 ( é marcada como ocupada ) , caso contrário é considerada também como livre sendo-lhe atribuída uma probabilidade de 0.3 também.

## Updating

Este processo é repetido ciclicamente tendo em conta as probabilidades anteriores das células , e os resultados das novas medições obtidas pelo laser de acordo com o modelo apresentado abaixo , de modo ás células irem acumulando uma probabilidade que melhor ilustre o seu estado real .

No entanto as probabilidades por uma razão de eficiência ,facilidade e de modo a não existir problemas numéricos com probabilidade muito próximas de zero não são explicitamente calculadas como se demonstra acima , sendo utilizada a forma ” log oods “ das probabilidades que assenta no principio ilustrado na equação seguinte .

As probabilidades finais que definem os estados das células são recuperadas usando a formula em baixo , que reverte o processo anterior da transformação das probabilidades em “log oods” , permitindo que seja obtida uma matriz com probabilidades que ilustram o estado das células , pronta a ser exportada para a construção do mapa real da sala e respetivos obstáculos e paredes . Todo este processo é realizado em tempo real sendo que no projeto em questão o mapa não é renderizado no fim mas sim ao longo de cada medição do laser o que permite como se mostra nas figuras x e y observar a evolução do estado das células enquanto o laser efetua mais medições sobre as mesmas e as probabilidades vão se acumulando .

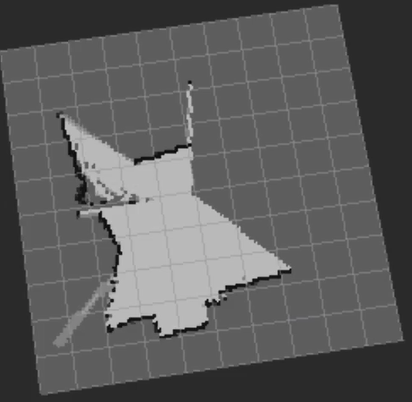


Figura Representação gráfica de um estado inicial do mapa.

Uma imagem com satélite, fotografia, transporte

Descrição gerada com confiança alta

Figura Representação gráfica de um estado com mais medições efetuadas do mapa.

# Results

## Rosbag

Para a realização de testes ao longo foi desenvolvimento foi adquirido um Rosbag de forma a testar todas as funcionalidades do software desenvolvido.

Usando esta informação foi obtido seguinte mapa (Figura 8).

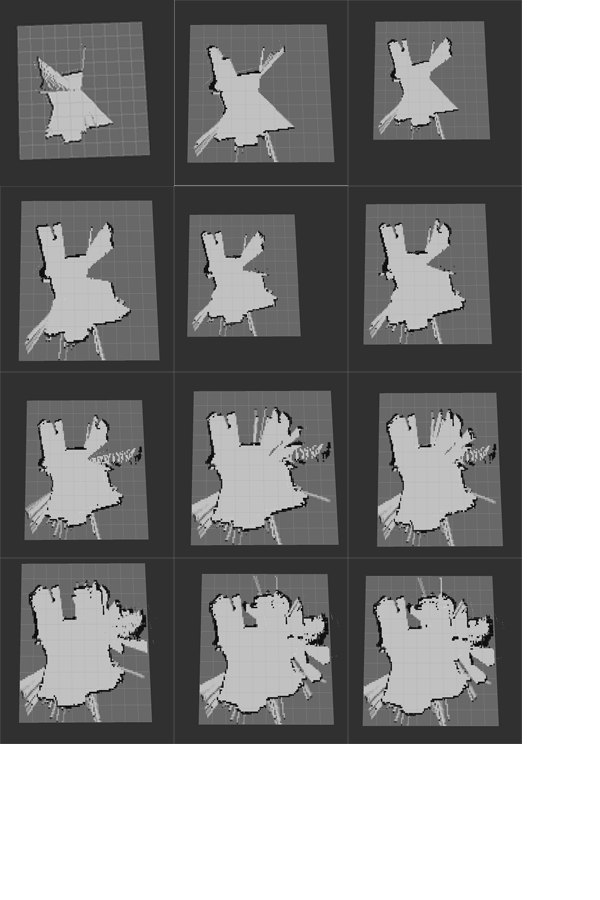


Figura 8 Mapa obtido com Rosbag adquirido no 8 Piso da Torre Norte do Técnico Lisboa, usando um Pioneer equipado com um Hokuyo LRF localizado usando equipamento de Motion Capture. Visualizado no RVIZ

## Stage simulations

Devido à falta de novos espaços, devido ao facto de o software foi desenvolvido considerando que a posição do robot era conhecida de um modo preciso, foi usado um RosStage Visível na Figura 9) de forma a demonstrar o total funcionamento do software desenvolvido.

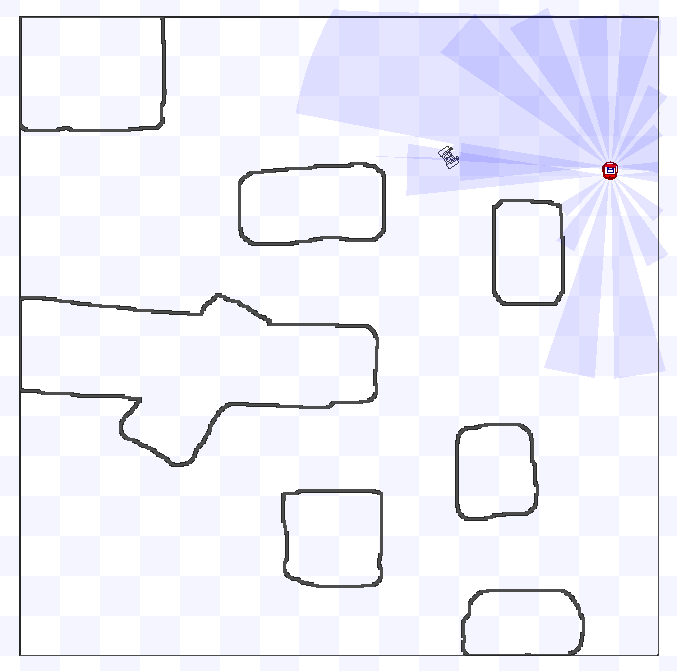


Figura 9 Top View ROS Stage usado para mapeamento, linhas pretas demarcam as paredes, ponto vermelho o robot

Desta simulação foi obtido o seguinte mapa (Figura 6).

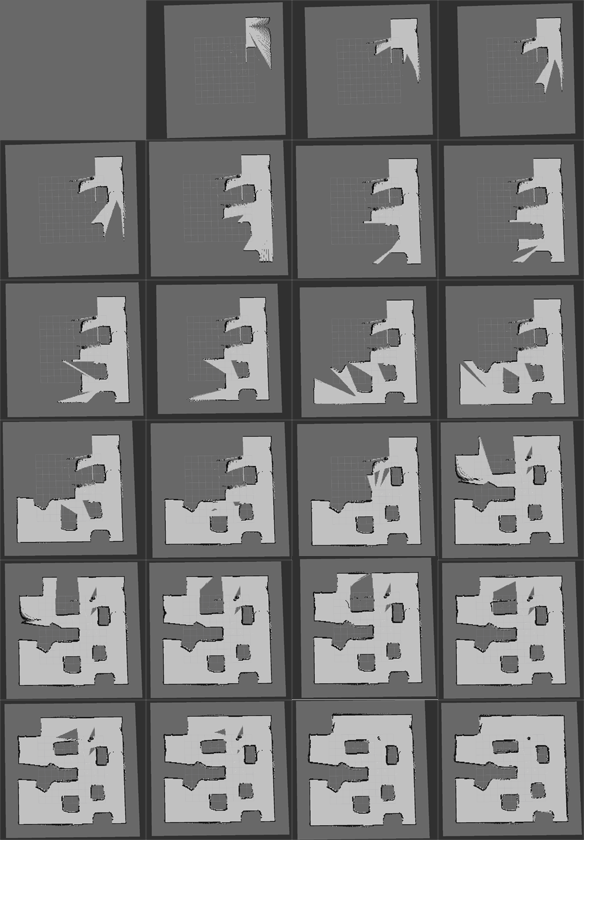


Figura 10 Mapa obtido usando ROS Stage apresentado na Figura 5. é possível verificar a evolução da construção do mapa conforme o robot se desloca pelo mapa. Visualizado no RVIZ

# Conclusion

Apesar de os resultados não serem totalmente os esperados, o algoritmo devolve uma aproximção bastante razoável da área explorada pelo lazer.

Devido á capacidade de envio de dados do LRF ser elevado e de o nosso algoritmo não estar optimizado o update do Occupancy Grid leva algum tempo, não sendo portanto em tempo real. No entanto, se se diminuir a resolução do mapa a rapidez do algoritmo aumenta.

Partindo deste trabalho inicial, depois de optimizado, poder-se-ia adicionar um algoritmo de localização, como por exemplo o Extended Kalman Filter ou o Monte Carlo, inutilizando o uso do Sistema de motion caption. Passando o algoritmo a ser autónomo e útil num ambiente não controlado como o que foi utilizado.

References

Lima, Oscar “GitHub: Autonomous Systems”

Available [online] at “htps://github.com/socrob/autonomous\_systems/

John Baker, Martin “Maths - Conversion Quaternion to Matrix”

Available [online] at http://www.euclideanspace.com/maths/geometry/

Lima, Pedro; Ventura Rodrigo; Ribeiro, Maria Isabem “Class Presentations”

Open Source Robotics Foundtion “ROS”

Available [online] at “http://wiki.ros.org/”

Python Software Foundation “Python”

Available [online] at <https://wiki.python.org/moin//>

# Authors

David Teles IST181053 MEEC

Leonor Silva IST181155 MEIC

Filipe Silva IST176985 MEEC

Pedro Soares IST181585 MEEC

1. LRF- “Laser Range Finder” equipamento que realiza medições de distancia usando um emissor laser e um recetor, usando o raio refletido e o tempo de viagem do pulso luminoso consegue obter medições de distância com uma precisão elevada. Este equipamento pode ser unidirecional ou omnidirecional usando alguma componente mecânica de forma a mover atingir diversos pontos usando apenas um emissor e recetor.

   ROS- “Robot Operating System” software que permite interligar vários sensores e atuadores de um modo fácil e pratico, tornando os sistemas mais permutáveis devido ao uso de mensagens standard.

   Frame- Referencial associado sensor ou atuador [↑](#footnote-ref-1)