DETECÇÃO DE OBJETOS EM AMBIENTE CONTROLADO POR MEIO DE MAPEAMENTO, UTILIZANDO UMA PLATAFORMA ROBÓTICA ARDUINO

Elton Cardoso do Nascimento – 3º ano do Curso Técnico Integrado ao Ensino Médio em Automação Industrial[1]

Professores Vera Lúcia da Silva[1], Masamori Kashiwagi [1]

verals@ifsp.edu.br, masamori@ifsp.edu.br

[1] Instituto Federal de Educação, Ciência e Tecnologia do Estado de São Paulo – Campus Suzano
Suzano – SP

Categoria: ARTIGO BÁSICO

Resumo: Este trabalho apresenta uma solução para um dos problemas apresentados pela Olimpíada Brasileira de Robótica (OBR), referente a localização de objetos na sala 3 (busca e resgate de vítima). Utilizou-se o robô Zumo, desenvolvido pela Pololu, em conjunto com a Plataforma Arduíno Mega e sensores ToF. Aplicaram-se técnicas de processamento de dados para uma tentativa simplificada de mapeamento da sala. Os resultados obtidos são promissores, mas ainda precisa de alguns ajustes.

Palavras Chaves: Competição Robótica, OBR, Mapeamento, busca e resgate.

Abstract: *This paper presents a solution to one of the problems presented by the Brazilian Robotics Olympiad (OBR) regarding the location of objects in room 3 (victim search and rescue). The Zumo robot developed by Pololu was used in conjunction with the Arduino Mega Platform and ToF sensors. Data processing techniques were applied for a simplified room mapping attempt. The results are promising, but still need some adjustments.*

Keywords: *Robotics Competition, OBR, Mapping. search and rescue.*

1 INTRODUÇÃO

Este projeto foi criado com objetivo de superar um dos desafios propostos pela Olimpíada Brasileira de Robótica (OBR), que é uma das principais competições brasileiras de robótica para o ensino médio.

A competição apresenta um percurso dividido em três salas, sendo que na terceira, a sala de resgate, é uma área com piso e paredes brancas. Nesta sala o robô deve vasculhar o ambiente em busca de vítimas, representadas por bolas de isopor prateadas ou pretas, e colocá-las em uma área elevada com uma parede preta em um dos cantos da sala. Esta área é conhecida como Zona de Resgate [OBR, 2018].

O grande desafio nesse ambiente é fazer com que um robô autônomo encontre e identifique as vítimas, e as conduzam para a área segura. Identificar e encontrar a Zona de Resgate sem sensores de cor, também apresenta-se como um grande desafio.

Este trabalho propõe mapear a sala por meio dos dados obtidos com sensores de distância ToF e giroscópio para medir distâncias do robô, localizado no centro da sala, até os elementos ao seu redor, e assim elaborar um mapeamento da sala, que permita determinar a localização da Zona de Resgate.

Este artigo estrutura-se da seguinte forma: a seção 2 descreve os materiais utilizados para realizar o projeto. Na seção 3 descreve-se o projeto proposto, mais especificamente na seção 3.1 como o robô foi fisicamente alterado; na seção 3.2 como os dados foram obtidos; e na seção 3.3 como foram processados. Na seção 4, os resultados dos testes; e por fim a seção 5 contém a conclusão baseadas nos resultados obtidos.

2 MATERIAIS

Para o desenvolvimento desse projeto, utilizou-se o robô proprietário "Zumo", que é "uma plataforma robótica com esteira controlada por Arduino. O robô possui dois micromotores com redução, acompanhados de um par de esteiras de silicone, e acelerômetro, giroscópio e magnetômetro de três eixos para detecção de impacto e orientação [Pololu, 2017][1].

O robô foi modificado, incluindo na sua arquitetura um sensor *Time-of-Fly* (ToF) VL53L0X, que é um sensor de distância, semelhante ao ultrassônico, que mensura o tempo que um pulso laser demora para ser emitido e refletido de volta ao sensor. Este modelo é capaz de fazer medidas de 5 a 120 cm (ADA, 2018).

Também fez-se necessário criar uma placa eletrônica para permitir que o Robô Zumo fosse capaz de comportar uma Plataforma Arduino Mega para seu controle, substituindo a Plataforma Arduino UNO Rev 3.

A programação foi realizada na IDE Arduino com a Linguagem de Programação C++.

1 Traduzido do original: "an Arduino-controllable tracked robot platform [..]. It includes two micro metal gearmotors coupled to a pair of silicone tracks, [...] a 3-axis accelerometer, magnetometer, and gyro for detecting impacts and tracking orientation."

3 O TRABALHO PROPOSTO

Para a detecção da zona de resgate, localizada na sala 3 do desafio da OBR, planejou-se a elaboração de uma arquitetura de robô e a programação capaz de obter dados da sala de resgate, e a partir disso detectar a localização da zona de resgate.

Para isso, utilizou-se sensores de distância ToF e giroscópio para medir distâncias do robô, no centro da sala, até os elementos ao seu redor, e assim elaborar um modelo da sala, um mapa 2D. O mapa posteriormente é processado com técnicas de processamento de sinais e outros, para conseguir a localização da zona de resgate.

3.1 Modificações Físicas do Robô Zumo

Para ser capaz de realizar as tarefas da solução proposta para o desafio da sala 3, foi preciso inserir nas laterais do robô sensores de distância, os ToF lasers. Foram utilizados dois sensores, para capturar mais dados em menos tempo.

Os sensores foram posicionados entre as esteiras do robô, para que estejam na altura certa para detectar a zona de resgate e as vítimas. A Figura 1 exibe o posicionamento dos sensores:

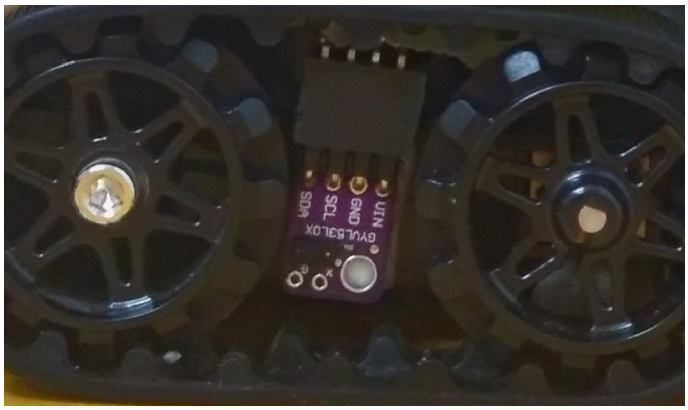

Figura 1: Foto do sensor posicionado entre as esteiras

O modelo VL53L0X foi escolhido por ser um dos únicos com dimensões para ajustar-se nessa posição.

Além do sensor, foi confeccionada uma placa eletrônica, inserida sobre o Robô Zumo de forma semelhante a Plataforma Arduino Uno Rev 3, para o qual o robô foi projetado, que é utilizada para conectar a Plataforma Arduino Mega ao robô. A Figura 2 mostra o Robô Zumo acoplado a placa desenvolvida.

Essa modificação possibilitou uma capacidade maior de memória e portas de E/S para o robô, permitindo resolver outros desafios da OBR.

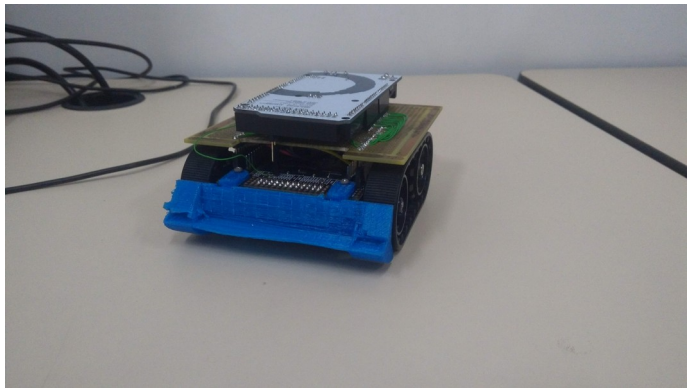

Figura 2: Foto do Arduino Mega no robô Zumo

Nas fotos (Figuras 1 e 2), observa-se também alterações mecânicas com PLA impresso na arquitetura do Robô Zumo. Essas modificações ocorreram para tornar possível a realização de outros desafios da OBR, como passar por redutores.

3.2 Captura dos Dados

Após a preparação estrutural do robô, foi possível utilizá-lo para obter dados da sala de resgate. Para isso, o robô foi posicionado no centro da sala, com uma de suas faces paralela a parede, com a zona de resgate posicionada em um dos cantos e sem nenhuma vítima. Com o giroscópio, calculou-se a angulação do robô, e portanto dos sensores, como demonstra a fórmula 1, e sua aproximação para as leituras discretas, na fórmula 2:

$$\text{grau}(T) = \int_0^T \omega(t)\,dt$$

Fórmula 1: Cálculo contínuo do grau de rotação do robô

em que:
grau(T) – angulação do robô no momento T (em grau)
ω(t) - velocidade angular do robô, mensurada pelo giroscópio (em grau/s)

$$grau_T = \sum_{t=0}^{T} \frac{\Delta t \times (\omega_t + \omega_{t+1})}{2}$$

Fórmula 2: Cálculo discreto do grau de rotação do robô

em que:

$grau_t$ – angulação do robô no momento T (em grau)

Δt – tempo entre as leituras (em s)

ω_t – velocidade angular do robô no momento t, mensurada pelo giroscópio (em grau/s).

Obtendo a angulação do robô, que é arredondada para graus inteiros, foi possível armazenar as leituras dos sensores de distância, sendo a de um sensor sempre em 180º a mais que a do outro, e calcular a média das leituras obtidas em um espaço de tempo, para cada ângulo inteiro. Obtém-se então 360 medidas correspondendo as distâncias até as paredes ou zona de resgate nos 360 graus, como mostra o gráfico na Figura 3:

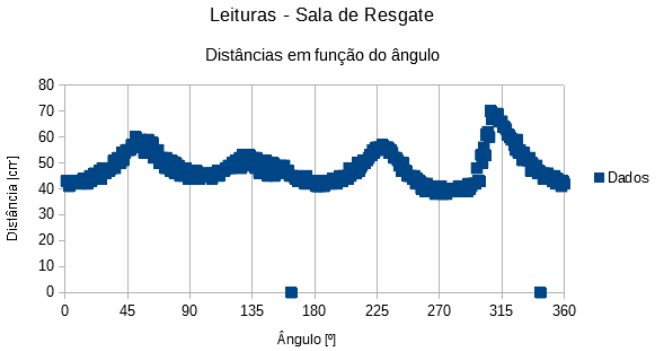

Figura 3: Leituras do Sensor - Distâncias em função do ângulo

3.3 Processamento.

Após a obtenção dos dados em forma mais bruta, eles são convertidos para valores (x,y) nos eixos cartesianos, como mostra as fórmulas 3 e 4, obtendo o gráfico da Figura 4:

$$x = leitura \cdot \cos(ângulo)$$

Fórmula 3: Conversão para foma retangular dos pontos - eixo x

$$y = leitura \cdot sen(ângulo)$$

Fórmula 4: Conversão para foma retangular dos pontos - eixo x

em que:

x – distância em x da leitura

y – distância em y da leitura

ângulo – angulação do robô em que foi obtida a leitura

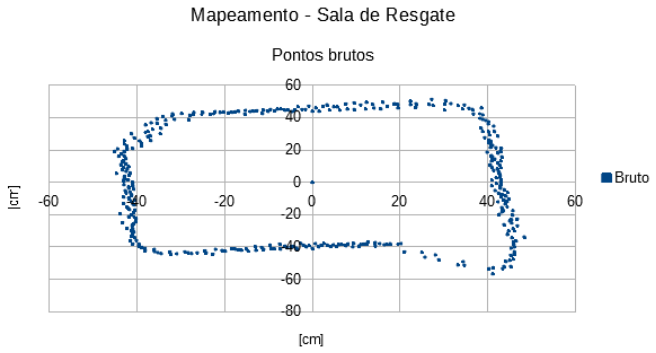

Figura 4: Mapeamento - Sala de Resgate - Pontos Brutos

No mapa da sala, nota-se a forma semelhante a um quadrilátero que é a sala, uma ponta que sai desse quadrilátero (quarto quadrante), que é a entrada da sala, e uma ponta que entra na sala, fazendo uma canto diagonal (segundo quadrante), que é a zona de resgate.

Porém, para uso desse modelo, encontrou-se a necessidade de filtrá-lo para torná-lo mais nítido. Para isso, programou-se dois filtros do tipo média móvel, um para cada eixo dos pontos, com um período de filtragem arbitrário de 10, como mostra as fórmulas 5 e 6:

$$x_i = \frac{\sum_{j=(p\div 2)}^{(p\div 2)} x_j}{p}$$

Fórmula 5: Filtragem dos dados - eixo x

$$y_i = \frac{\sum_{j=(p\div 2)}^{(p\div 2)} y_j}{p}$$

Fórmula 6: Filtragem dos dados - eixo y

em que:

xi/yi – medida x/y do ponto i filtrado

p – período do filtro

xj/yj – medida x/y da leitura j bruta

Depois da filtragem, o mapa obtido, como observa-se na Figura 5, torna-se mais nítido e preciso:

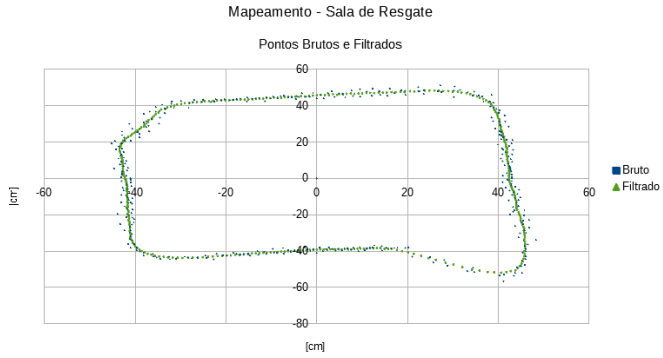

Figura 5: Mapeamento - Sala de Resgate: Pontos Brutos e Filtrados

Neste momento, tornou-se possível observar claramente a Figura e determinar o canto em que se encontra a zona de resgate, porém era preciso criar um algorítimo para detectá-lo. Para tal, calculou-se quatro retas, que representam os quatro cantos da sala da seguinte forma:

Primeiramente, considerando que os lados do robô estavam paralelos as paredes da sala, utiliza-se os pontos de até 10 cm dos pontos sobre os eixos, ou seja, os pontos centrais das paredes nos ângulos 0, 90, 180 e 270; para realizar regressões lineares com mínimos múltiplos quadrados e calcular primeiras versões das retas. Essas retas são depois ajustadas com os pontos de até 1,5 cm próximos a elas, recalculando-as; esse processo repete-se várias vezes, pois ao ajustá-las mais pontos se tornam próximos a elas. A versão final dessas retas pode ser visto na Figura 6:

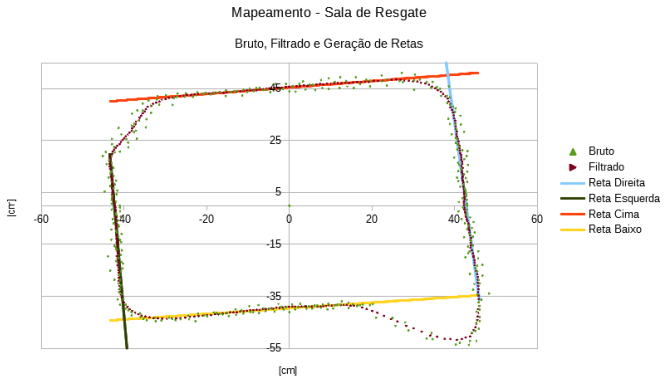

Figura 6: Mapeamento - Sala de Resgate - Bruto, Filtrado e Geração de Retas

Porém, o real proveito das retas não foram elas em si, mas a capacidade de excluir do conjunto de pontos onde pode estar a zona de resgate todos os pontos próximos a elas. Reúnem-se nesse conjunto de pontos retirados, todos os pontos que se encontram fora do quadrilátero determinado pelas quatro retas. Obtêm-se então, um reduzido conjunto de pontos onde pode estar a zona de resgate, como observado na Figura 7:

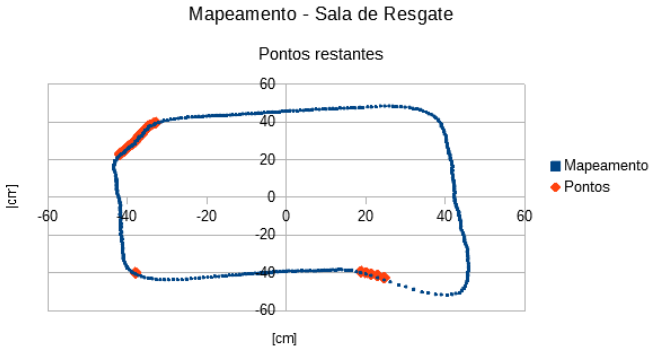

Figura 7: Mapeamento - Sala de Resgate - Pontos restantes

Por fim, verifica-se o subconjunto que possui a maior quantidade de pontos adjacentes, visto que a zona de resgate é a maior perturbação nas leituras, e define-se o ângulo médio desses pontos como o ângulo da zona de resgate.

Após a elaboração do algorítimo com base em uma amostra experimental, foram obtidas mais três amostras, colocando o robô no centro da sala e o rotacionado. Os dados foram enviadas ao computador, onde o algoritmo foi testado.

Não foram realizados testes com o processamento no software embarcado diretamente no robô, por conta de sua memória limitada, tornando-o incapaz de lidar com grande número de dados.

4 RESULTADOS E DISCUSSÃO

Os mapas obtidos nos três testes, com suas respectivas retas, podem ser observados nas Figuras 8, 9 e 10:

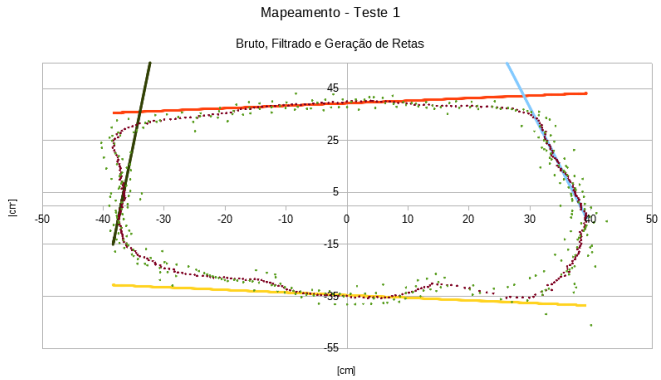

Figura 8: Teste 1

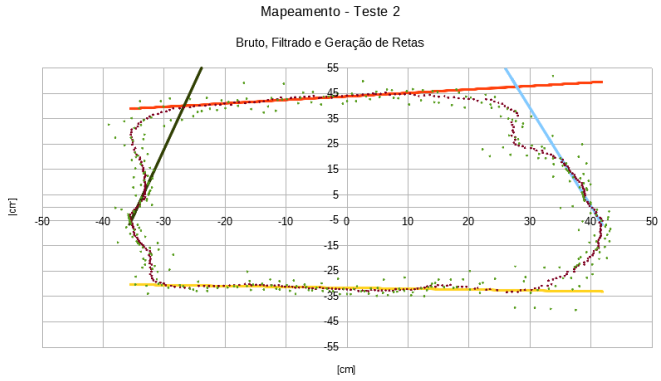

Figura 9: Teste 2

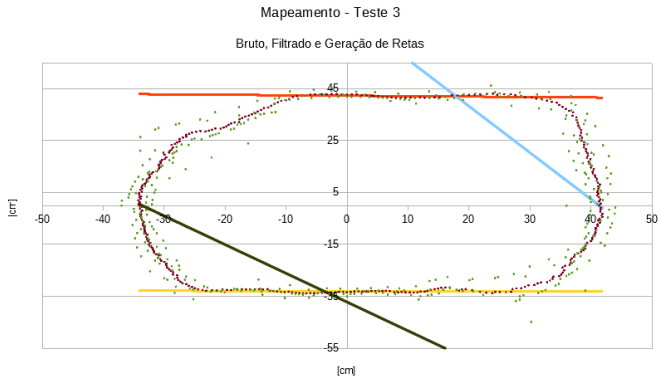

Figura 10: Teste 3

Nota-se que as retas obtidas ajustam-se aos lados do quadrilátero na maior parte das vezes, porém com grandes falhas na terceira tentativa.

Isso deve-se as falhas obtidas nas leituras do sensor, que ocorreram provavelmente pela disformidade da velocidade angular em que o robô rotacionava, que era muito aparente por conta do sistema de esteiras que ele utiliza-se para locomoção, e pela dificuldade de alimentá-lo corretamente, o que dificultou o movimento e, em alguns momentos, chegou a pará-lo completamente por alguns instantes. Devido ao desgaste das pilhas, foi preciso esperar alguns minutos entre cada tomada de dados.

Entretanto, os problemas detectados não impediram o algorítimo de formar os conjuntos de pontos e encontrar, a partir deles, a zona de resgate.

Por fim, não tornou-se possível embarcar, efetivamente, o algorítimo desenvolvido no robô, devido a memória da Plataforma Arduino Mega não possuir capacidade para processar a quantidade de dados necessários. Desta forma, os

testes foram executados em um computador de mesa, com os dados obtidos pelo robô.

5 CONCLUSÕES

Conclui-se que a solução apresentada efetivamente pode ser um caminho para detectar a zona de resgate, porém que ainda precisa de ajustes. Primeiramente, como é usada como hipótese no algoritmo que os pontos 0, 90, 180 e 270 são os centos das retas, é preciso encontrar uma forma de alinhar corretamente o robô dentro da sala, ou remover essa necessidade, gerando as retas de outra forma.

É necessário também a ampliação da capacidade de memória e provavelmente de processamento do controlador, possivelmente substituindo-o por outro ou com uso de memórias como cartões SD.

Por fim, necessita-se melhorar a estrutura do robô, para tornar mais precisa e suave sua movimentação, assim gerando mapas mais confiáveis, e efetuar um número maior de testes com mais dados.

REFERÊNCIAS BIBLIOGRÁFICAS

OBR. Manual de Regras e Instruções: Etapa Regional/Estadual. Disponível em: <http://www.obr.org.br/wp-content/uploads/2018/03/OBR2018_MP_ManualRegras Regional_v1Mar.pdf>. Acesso em 27 fev. 2019.

POLOLU. Zumo Robot for Arduino, v1.2 (Assembled with 75:1 HP Motors). Disponível em: <https://www.pololu.com/product/2510>. Acesso em 29 nov. 2017.

ADA, Lady [Limor Fried]. Adafruit VL530LX Time of Flight Micro-LIDAR Distance Sensor Breakout. 2018. Disponível em: <https://cdn-learn.adafruit.com/download/pdf/adafruit-vl53l0x-micro-lidar-distance-sensor-breakout.pdf?timestamp=1542823307>. Acesso em 21 nov. 2018.

Observação: O material multimídia deste trabalho encontra-se disponível em: www.mnr.org.br/mostravirtual.