

Proposta de uma arquitetura de navegação para um robô móvel diferencial

Proposal of a navigation architecture for a differential mobile robot

DOI:10.34117/bjdv8n5-342

Recebimento dos originais: 21/03/2022

Aceitação para publicação: 29/04/2022

Leonardo Gonçalves Batista

Graduado em Engenharia de Controle e Automação

Instituição: Instituto Federal do Espírito Santo - Campus Serra

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP: 29173-087 Manguinhos, Serra, ES, Brasil

E-mail: leonardo-baptista@live.com

Lucas Mantuan Ayres

Graduado em Engenharia de Controle e Automação

Instituição: Instituto Federal do Espírito Santo - Campus Serra

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP: 29173-087 Manguinhos, Serra, ES, Brasil

E-mail: lucasmantuan@live.com

Joabe Ruela da Silva

Graduado em Engenharia de Controle e Automação

Instituição: Instituto Federal do Espírito Santo - Campus Serra

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP: 29173-087 Manguinhos, Serra, ES, Brasil

E-mail: joaberuella@gmail.com

Vinicius da Rocha Motta

Mestre em Engenharia de Controle e Automação

Instituição: Instituto Federal do Espírito Santo - Campus Serra

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP: 29173-087 Manguinhos, Serra, ES, Brasil

E-mail: viniciusdarochamotta@gmail.com

Vinicius Moura Marques

Doutor em Engenharia Elétrica

Instituição: Instituto Federal do Espírito Santo

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP: 29173-087 Manguinhos, Serra, ES, Brasil

E-mail: vinicius.marques@ifes.edu.br

Marco Antonio de Souza Leite Cuadros

Doutor em Engenharia Elétrica

Instituição: Instituto Federal do Espírito Santo - Campus Serra

Endereço: Instituto Federal do Espírito Santo - Campus Serra, Rod. ES 010, Km 6,5,

CEP:29173-087 Manguinhos, Serra, ES, Brasil

E-mail: marcoantonio@ifes.edu.br

RESUMO

O presente trabalho propõe uma arquitetura de navegação para robôs móveis não holonômicos, com a capacidade de interpretar comandos de voz e relacioná-los com posições conhecidas no mapa de navegação. Além disso, através do algoritmo de busca informada A* (lê-se A Estrela), a arquitetura tem a capacidade de planejar um caminho viável entre o ponto atual e o ponto de destino. A navegação do robô móvel é garantida com a utilização do controlador *Backstepping* com a finalidade de percorrer uma trajetória pré-determinada. Portanto, ao final, são apresentados resultados, conclusões e recomendações.

Palavras-chave: arquitetura de navegação, algoritmo A*, controlador backstepping, algoritmo de reconhecimento de fala.

ABSTRACT

This paper proposes a navigation architecture for non-holonomic mobile robots, with the ability to interpreted voice commands and relate them to known positions in the navigation map. In addition, through the informed search algorithm A* (reads A Star), the architecture can plan a viable path between the current point and the target point. The navigation of the mobile robot is guaranteed with the use of the Backstepping controller in order to follow a previously determined trajectory. Therefore, at the end of this paper, results, conclusions and recommendations are presented.

Keywords: navigation architecture, A* algorithm, backstepping controller, speech recognition algorithm.

1 INTRODUÇÃO

Atividades humanas repetitivas, perigosas ou em ambientes de difícil acesso estão sendo realizadas por robôs com o objetivo de facilitar a sua realização, ou seja, o ser humano está sendo substituído por essas máquinas quando a tarefa a ser realizada envolve certo nível periculosidade. Robôs autônomos são capazes de executar tarefas e navegar sem intervenção humana (Goris, 2005). Existem diversas aplicações de robótica na vida cotidiana, como: agricultura, limpeza, serviço médico, mineração, segurança, serviço militar, ambientes industriais (Algabri et al., 2015), inclusive em explorações espaciais (Davison, 1998).

A navegação de robôs móveis consiste no controle do movimento do robô de um ponto de partida até um ponto de destino em um determinado ambiente de trabalho, que

pode ter a capacidade de evadir obstáculos. Assim, o ambiente de navegação pode ser classificado como ambiente estruturado (conhecido), ambiente semi-estruturado e ambiente não-estruturado (desconhecido) (Algabri et al., 2015).

Em diversas aplicações da robótica móvel a interação do ser humano com o robô se faz necessária, e uma das formas de realizar essa interação é através do uso de algoritmos de reconhecimento de fala.

A tarefa de uma arquitetura de navegação é planejar um caminho para um objetivo especificado, de forma a evitar obstáculos conhecidos ordenando o deslocamento do robô móvel durante sua navegação. (Siegwart & Nourbakhsh, 2004). Por esta ótica, este trabalho propõe uma arquitetura de navegação autônoma, utilizando um robô móvel não holonômico, cujo usuário/operador, por meio de comandos de voz, define os destinos em um ambiente interno conhecido. O planejador de caminho utiliza o algoritmo de busca informada A* que fica responsável por determinar o melhor caminho entre a posição atual e a posição de destino do robô móvel. Posteriormente, é realizada a geração da trajetória de referência, e desta forma, esses dados juntamente com os dados de localização determinados pela odometria (*Dead Reckoning*) são utilizados para o controle de trajetória. Neste contexto, foi utilizado o controlador *Backstepping*. Para validar a navegação do robô móvel, foi utilizado um sistema de localização por imagem utilizando uma câmera externa a qual foi calibrada e testada.

Este artigo está organizado da seguinte forma: na seção 2 são apresentados o ambiente de trabalho e as ferramentas utilizadas. Na seção 3 é descrita a proposta de arquitetura de navegação. Na seção 4 é mostrado o ambiente de simulação. Na seção 5 são mostrados os resultados práticos. Finalizando, na seção 6 são discutidas as conclusões e recomendações.

2 DESCRIÇÃO DO AMBIENTE DE TRABALHO

Na Figura 1 se mostra o ambiente de realização dos experimentos práticos. Nele desenvolveu-se um mapa para a realização de testes com o robô móvel.

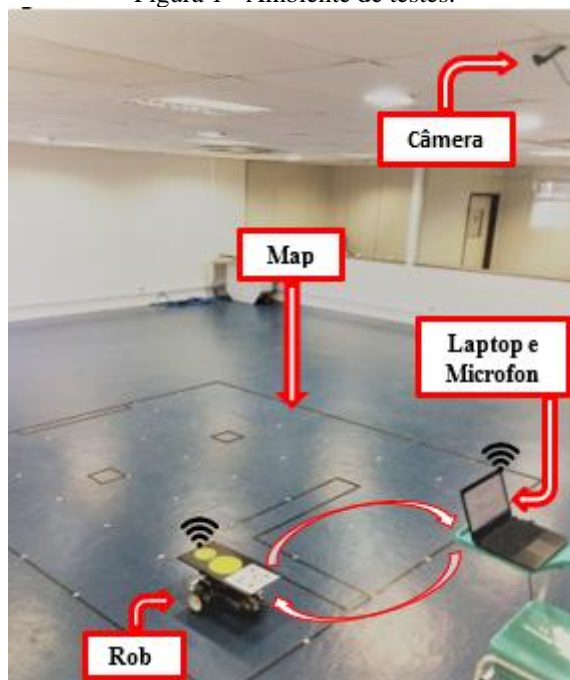
O software LabVIEW® foi utilizado para o desenvolvimento de toda arquitetura de navegação pois o mesmo é compatível com a estrutura robótica utilizada, o robô DaNI 2.0, que possui uma placa NI sbRIO. Este sistema embarcado integra um processador de tempo real, uma placa FPGA e entradas e saídas digitais e analógicas (National Instruments, 2014).

Um roteador Wi-Fi foi adicionado à estrutura do robô móvel, garantindo a comunicação com o *laptop* externo e o monitoramento em tempo real do robô.

O *laptop* responsável por fazer o condicionamento das informações dos sinais de fala é o modelo Vostro 5470, por meio de um microfone de mesa da fabricante Maxprint (Maxprint, 2017).

Para validar a navegação do robô móvel, foi utilizado a câmera LifeCam HD-3000 (Microsoft, 2014) presa ao teto e conectada ao *laptop*, a fim de identificar um marco visual fixo no robô, e a partir deste, determinar a localização real do robô móvel.

Figura 1 - Ambiente de testes.



3 PROPOSTA DA ARQUITETURA DE NAVEGAÇÃO

Segundo Souza (2012), uma arquitetura de navegação compreende as seguintes etapas sequenciais:

- Mapeamento: Representação espacial com ou sem obstáculos físicos, armazenada no robô.
- Localização: Consiste na determinação de posição e orientação no ambiente de navegação.
- Planejamento do Caminho: Estratégia adotada para encontrar o caminho viável entre a posição inicial e final.
- Geração de Trajetória: Define previamente a descrição temporal da posição, velocidade linear e angular do robô.

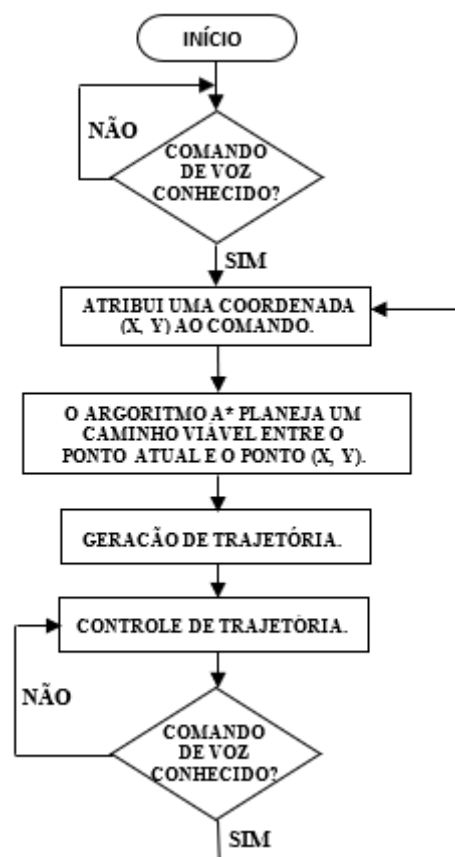
- Execução da trajetória: Etapa onde as ações dos atuadores são determinadas e adaptadas às mudanças ambientais, de modo a executar a trajetória pré-definida.

A arquitetura de navegação foi projetada, para que a mudança de trajetória seja feita a qualquer momento, via comando de voz. A aquisição, tratamento e identificação da fala são executados no *laptop*, e através de uma variável na rede são transmitidos os dados para o robô móvel.

A

Figura 2 apresenta o fluxograma da arquitetura de navegação proposta.

Figura 2 – Fluxograma da arquitetura de navegação.



3.1 MAPEAMENTO

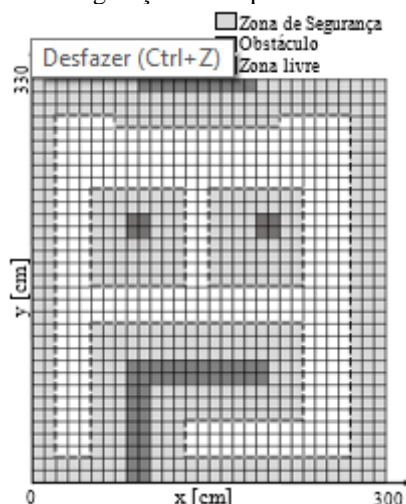
Uma das configurações dos mapas métricos é o da Grade de Ocupação (*Occupancy Grid*), visto sua maior capacidade de representação detalhada. Essa estrutura descreve o cenário em espaços discretizados, de forma que este seja representado por uma matriz: cada célula representa um espaço ocupado, vazio ou inexplorado (Elfes, 1989).

Para fins de utilização do algoritmo A* na construção do planejador de caminho, desenvolveu-se um mapa, com dimensões internas de 300cm x 330cm, mapeado via

Grade de Ocupação, via software LabVIEW®. As grades foram divididas de modo que uma célula corresponda a 10cm x 10cm no espaço real. Na configuração do mapa representando na Figura 3, as regiões em cinza escuro indicam os obstáculos físicos presentes no ambiente real. As áreas em cinza claro, delimitadas pelas linhas tracejadas, foram acrescidas de forma a garantir que o algoritmo de caminho planejasse os cursos considerando uma margem de segurança entre o robô e os obstáculos. As células brancas são espaços livres para planejamento de caminho e para navegação do robô.

Ressalta-se que o robô, ao navegar, pode invadir a zona de segurança devido a suas dimensões em relação ao mapa. Portanto, como o robô DaNI 2.0 apresenta dimensões 405mm x 368mm x 150mm (National Instruments, 2014), zonas de segurança entre 20cm e 40cm de comprimento foram adicionadas em torno dos obstáculos após se observar o comportamento do robô durante os testes de simulação ao longo do mapa virtual que será apresentado na seção 4.

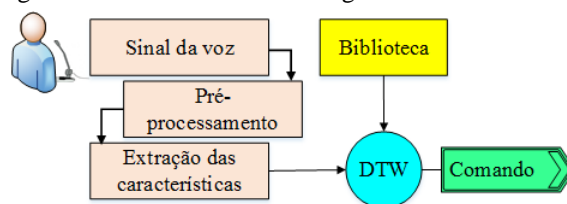
Figura 3 – Configuração do mapeamento do ambiente.



3.2 RECONHECIMENTO DE FALA

Na Figura 4 se mostra o diagrama do algoritmo de reconhecimento de fala.

Figura 4 – Diagrama de funcionamento do algoritmo de reconhecimento de fala.



Cada sinal de fala é obtido pelo microfone em uma resolução de 16 bits, amostragem de 11025 Hz, duração de 3s e é tratado por um filtro passa-baixa da classe IIR (*Infinite Impulse Response*) com topologia Butterworth de 5ª ordem para eliminar ruídos. No pré-processamento é realizado a pré-ênfase para compensar a parte das altas frequências que foram suprimidas durante a fala e melhorar da relação sinal/ruído e o janelamento é onde se divide o sinal em “janelas” de 20ms. Em seguida é feita a extração das características em cada “janela” do sinal da fala aplicando FFT (*Fast Fourier Transform*) e, portanto, é realizada a análise espectral, onde se obtém vetores com os MFCCs (*Mel-Frequency Cepstral Coefficients*). Na sequência, os vetores dos MFCCs são comparados com os vetores dos MFCC armazenados na biblioteca utilizando a ferramenta matemática DTW (*Dynamic Time Warping*), que fará a medição da similaridade entre duas séries temporais. Se ambos os vetores forem semelhantes, executa-se o comando previamente relacionado ao comando de fala analisado; caso contrário, retorna-se ao início para obter um novo comando por voz.

Para o algoritmo de reconhecimento de fala foram estabelecidos quatro comandos para serem reconhecidos: “PONTO A”, “PONTO B”, “PONTO I” e “PONTO J”. Em cada comando reconhecido está agregado duas constantes que representam as coordenadas x e y (em centímetros) no mapa.

Na

Tabela 1 demonstram-se as coordenadas de referência para os quatro comandos de voz e na Figura 5 se resume a interface de usuário. Quando um comando é detectado e reconhecido, seu nome é exibido na caixa de texto “COMANDO RECEBIDO” e o indicador luminoso relativo ao comando recebido é aceso na interface.

Tabela 1 – Pontos do mapa e suas respectivas coordenadas.

Comandos	Ponto A	Ponto B	Ponto I	Ponto J
x [cm]	160	260	140	140
y [cm]	30	240	280	260

Figura 5 - Interface de usuário.



O algoritmo de reconhecimento de fala foi treinado em um ambiente controlado por apenas um usuário e foram utilizadas 20 amostras de fala para cada um dos quatro comandos, totalizando 80 amostras de treinamento. Com o objetivo de testar o desempenho, foi dado 50 comandos para cada ponto, totalizando 200 comandos. O Ponto A, B, I e J resultou em 42, 36, 34 e 35 acertos, respectivamente.

3.3 ALGORITMO A*











Para aplicações em configurações métricas, o algoritmo A* é um dos mais utilizados na busca de caminho. Ele é definido como um algoritmo *Best-First Search*, pois explora cada nó da configuração espacial e expande o mais promissor em relação ao objetivo (Duchon et al., 2014). Para tanto, faz uso de uma função de avaliação e submete-a a cada nó. O nó que retornar o menor custo na função de avaliação, será expandido (Hart et al., 1968). Na Equação (1) se mostra a função de avaliação $f(n)$ que estima o custo total do nó atual até o nó alvo.

$$f(n) = g(n) + h(n) \quad (1)$$

Onde: $g(n)$ é o custo do caminho percorrido e $h(n)$ é o valor da heurística do nó n até um nó objetivo (distância em linha reta no caso de distancias espaciais).

A estratégia do algoritmo é avaliada utilizando o mapa métrico representado na Figura 6. As células de índice (1,2) e (1,5) indicam a origem e destino, respectivamente. Em I se obtém os sucessores II, III, IV e V. O algoritmo A* aplica a função de avaliação para esses nós e estima $f(n_{II})=10+40=50$, $f(n_{III})=14+50=64$, $f(n_{IV})=10+40=50$ e $f(n_V)=14+30=44$. Verifica-se que o nó V é expandido, pois o custo total retornado é menor, comparado ao custo total dos demais nós. Em seguida, o mesmo procedimento é aplicado aos sucessores de V, os nós VI, VII e VIII. Após avaliação do custo total, o nó VI é expandido, e na análise posterior, o nó X. Desta forma, ao aplicar esse procedimento, o algoritmo A* garante que esse é o caminho de custo mínimo para alcançar o destino.

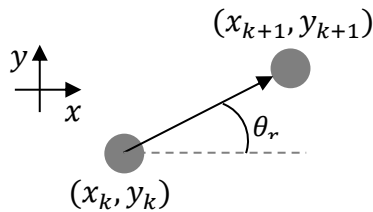
Figura 6 – Avaliação do algoritmo A* em mapa métrico.

	1	2	3	4	5
1					
2					
3					
4					

3.4 ALGORITMO DE GERAÇÃO DE TRAJETÓRIA

Neste trabalho, o algoritmo A* fornece em sua saída uma matriz de posição contendo o caminho entre o ponto atual e o ponto de destino. Entretanto, é necessário seguir algumas etapas para obter uma matriz de dados suficiente para ser utilizada como referência para o controlador de trajetória. O passo inicial é interpolar o caminho para criar um número maior de referências de posição. O método utilizado de interpolação foi o *Hermite Interpolation* (Ciarlet & Raviart, 1972). Em seguida, é feito o cálculo da orientação θ_r (Figura 7) para cada índice da matriz de referências pela Equação (2), obtendo assim a matriz $[x_r, y_r, \theta_r]^k$.

Figura 7 – Orientação de referência.



$$\theta_r = \tan^{-1} \left(\frac{y_{k+1} - y_k}{x_{k+1} - x_k} \right) \quad (2)$$

Com a trajetória interpolada e com as orientações de referência para cada índice da trajetória, calcula-se os referenciais de velocidade linear e angular de cada ponto pela Equação (3), onde Δt é o *time step* do programa de execução.

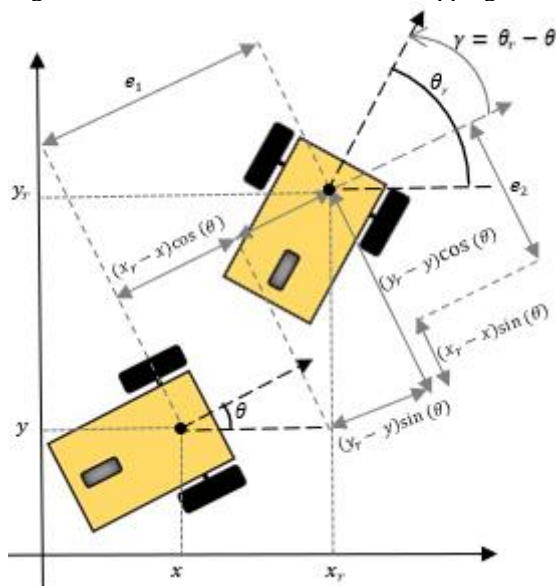
$$\begin{bmatrix} v_r \\ \omega_r \end{bmatrix} = \left(\frac{1}{\Delta t} \right) \begin{bmatrix} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2} \\ \theta_{k+1} - \theta_k \end{bmatrix}$$

3.5 CONTROLADOR DE TRAJETÓRIA

A escolha do controlador de trajetória foi importante para obter resultados satisfatórios de navegação. Sua escolha foi baseada em resultados encontrados na literatura. Segundo Tomasi et. al. (2015), o controlador *Backstepping* foi o que obteve melhores resultados para a trajetória do tipo lemniscata, utilizando o robô DaNI 2.0. Sendo assim, este controlador foi escolhido para ser utilizado neste trabalho.

A técnica de controle *Backstepping* utiliza os dados de referência determinados pelo gerador de trajetória $[x_r, y_r, \theta_r]^k$ e a localização determinada pela odometria $[x, y, \theta]^k$ para obter seus erros característicos. Na Figura 8 mostra-se os erros considerados no desenvolvimento deste controlador, sendo representados na forma matricial na Equação (4). Este controlador utiliza o ponto de centro de massa do robô como referência.

Figura 8 – Erros do controlador Backstepping.



$$\begin{bmatrix} e_1 \\ e_2 \\ \gamma \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \quad (4)$$

A lei de controle do controlador *Backstepping* é apresentada em Kanayama et al. (1990) e é mostrada abaixo na Equação (5):

$$\begin{bmatrix} v_c \\ \omega_c \end{bmatrix} = \begin{bmatrix} v_r \cos(\gamma) + k_1 e_1 \\ \omega_r + k_2 v_r e_2 + k_3 v_r \sin(\gamma) \end{bmatrix} \quad (5)$$

onde v_c e ω_c são as velocidades linear e angular de saída do controlador respectivamente, e v_r e ω_r são velocidades de referência determinadas pelo gerador de trajetória.

Os parâmetros do controlador a serem determinados são: k_1 , k_2 e k_3 . Para os testes de simulação e práticos utilizou-se o método de tentativa e erro, de forma que os erros de posição e orientação fossem minimizados. Na Tabela 2 se apresenta tais parâmetros.

Tabela 2 – Sintonia do controlador Backstepping.

	k_1	k_2	k_3
Simulação	5,00	5,00	7,00
Real	3,10	2,25	8,28

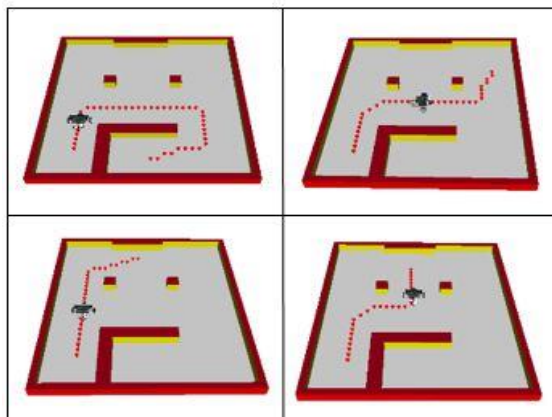
3 AMBIENTE DE SIMULAÇÃO

Os experimentos iniciais com o algoritmo A* e o controlador *Backstepping* foram realizados em um ambiente de simulação. O LabVIEW® Robotics Module permite a implementação de algoritmos para o robô DaNI 2.0 em um ambiente virtual. Com algumas modificações, é possível testar os códigos no robô real. A fim de verificar melhores resultados, foi desenvolvido um mapa virtual em três dimensões (

Figura 9), seguindo as mesmas medidas do mapa real.

Utilizando funções internas do LabVIEW®, foi possível obter a localização do robô móvel. Entretanto, por se tratar de um ambiente de simulação, erros não-sistemáticos da navegação não são considerados, por exemplo: escorregamento, desalinhamento e diâmetros diferentes das rodas do robô. Desta forma, a navegação do robô será validada apenas em função dos resultados práticos.

Figura 9 – Mapa virtual.



5 RESULTADOS PRÁTICOS

Os testes práticos foram realizados em um laboratório com o objetivo de avaliar o desempenho no mapa real da arquitetura implementada. Em todos os testes, o robô foi posicionado inicialmente na coordenada $x = 50\text{cm}$ e $y = 50\text{cm}$, e em seguida foi dado o comando por voz, por exemplo “Ponto A”; assim, o robô seguiu em direção a este ponto de destino.

Para obter a localização real do robô móvel foi utilizado uma câmera de vídeo realizando o processamento contínuo de imagens, segmentando um marco visual fixo ao robô móvel (Figura 1), essa estratégia é semelhante a que Pimentel et al. (2014) usa em seu trabalho. Em virtude da altura do marco visual em relação ao chão, um erro considerável foi apresentado nas medições. Entretanto, utilizando-se semelhança de triângulos e distância euclidiana foi feita uma correção na estimativa, e ao analisar os valores determinados por este sistema em relação a medição real, verificou-se que os erros absolutos de posição ficaram entre $\pm 1\text{cm}$. Devido sua eficiência, esse sistema foi utilizado para avaliar a navegação do robô móvel.

Na Figura 10 e na Figura 11 são demonstrados os resultados dos testes práticos para as trajetórias do Ponto A, B, I e J.

Figura 10 – Trajetórias para os pontos A, B, I e J – linhas contínuas: referência; linhas pontilhadas: odometria.

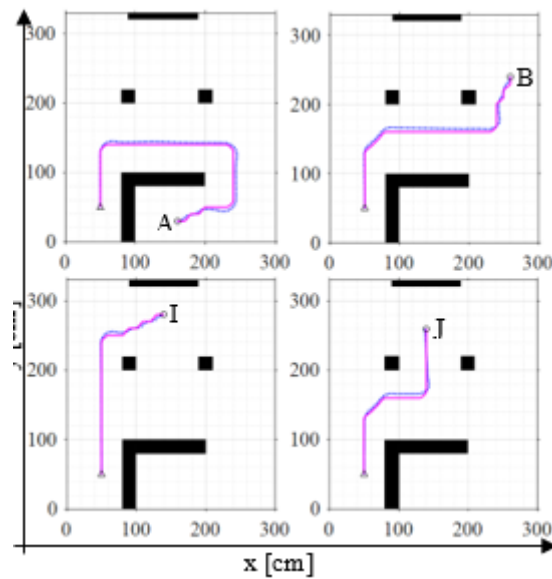
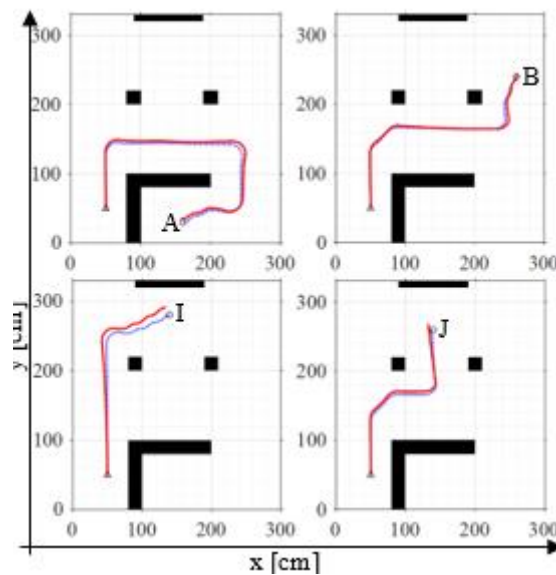


Figura 11 - Trajetórias para os pontos A, B, I e J – linhas contínuas: real; linhas pontilhadas: odometria.



Na Figura 10, as linhas contínuas foram obtidas pelo algoritmo de geração de trajetória (referência) e as linhas pontilhadas foram obtidas pela odometria. Na Figura 10, as linhas contínuas foram obtidas por processamento de imagens e as linhas pontilhadas foram obtidas pela odometria.

6 CONCLUSÕES E RECOMENDAÇÕES

Os resultados mostram que o algoritmo de comando de voz foi inserido numa estrutura de navegação de um robô móvel de forma funcional, abrindo espaço para diferentes aplicações.

Através da análise dos resultados obtidos, pode-se afirmar que o algoritmo A* funcionou como esperado na prática e na simulação, garantindo um caminho de custo

mínimo entre o ponto atual e o de destino, respeitando principalmente os obstáculos e a zona de segurança imposta no mapeamento.

Para observar o desempenho da arquitetura, foram realizados testes para os quatro caminhos, com o intuito de explorar o cenário e os obstáculos físicos. A partir da Figura 10, é possível concluir que o controlador funcionou como esperado, mantendo a posição do robô (medida pela odometria) bem próxima da referência determinada pelo gerador de trajetória. Entretanto, a navegação pode ser avaliada pela Figura 11, já que esta apresenta a posição real do robô móvel.

Pela análise gráfica realizada a partir da Figura 11, constatou-se um erro de aproximadamente 13 cm na trajetória do Ponto I. Este foi o maior erro detectado. Pode-se observar que o objetivo de chegar ao ponto final foi realizado, e verificou-se que o robô em nenhuma das trajetórias propostas colidiu com os obstáculos físicos do mapa real.

Para aperfeiçoamento da arquitetura, será proposto em trabalho futuros a associação da odometria com a detecção de marcos visuais (*landmarks*) e sensores inerciais baseado em fusão sensorial, via filtro de Kalman. Desta forma, poderá se obter um sistema de localização mais preciso. Também se pretende aumentar as dimensões do mapa utilizado. Além disso, propõe-se inserir um sistema dinâmico de desvio de obstáculos.

AGRADECIMENTOS

À Cooperação CAPES/FAPES - Programa de Desenvolvimento da Pós-Graduação - PDPG, através do projeto “TIC+TAC: Tecnologia da Informação e Comunicação + Tecnologia de Automação e Controle, As Tecnologias Inteligentes, Prioritárias,” pelo apoio financeiro da pesquisa, por meio do Edital FAPES/CNPq N° 23/2018 – PRONEM (Termo de Outorga 133/2021 e Processo N° 2021-CFT5C).

REFERÊNCIAS

Algabri, M., Mathkour, H., Ramdane, H. and Alsulaiman, M. (2015). Comparative study of soft computing techniques for mobile robot navigation in an unknown environment. *Computers in Human Behavior*, 50, pp.42-56.

Ciarlet, P.; Raviart, P. (1972) General lagrange and hermite interpolation in R^n with applications to finite element methods. *Archive for Rational Mechanics and Analysis*, v. 46, n. 3.

Davison, A. (1998). *Mobile Robot Navigation Using Active Vision*. Doutorado. University of Oxford.

Duchoň, F., Babinec, A., Raphael, B., Kajan, M., Beňo, P., Florek, M., Fico, T., Jurišica, L. (2014). Path Planning with Modified a Star Algorithm for a Mobile Robot. *Procedia Engineering Modelling of Mechanical and Mechatronic Systems*, Vol. 96, pp. 59-69.

Elfes, A. (1989). *Occupancy Grid: A Probabilistic Framework for Robot Perception and Navigation*, Tese de doutorado, Carnegie Mellon University, Pennsylvania, USA.

Goris, K. (2005). *Autonomous mobile robot mechanical design*. Vrije Universiteit Brussel: Thesis academic degree Civil Mechanical-Electrical Engineering.

Hart, P., Nilsson, N. and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), pp.100-107.

Kanayama, Y., Kimura, Y., Miyazaki, F. and Noguchi, T. (1990). A stable tracking control method for an autonomous mobile robot. *Proceedings., IEEE International Conference on Robotics and Automation*.

Maxprint. (2017). Site corporativo da Maxprint. [online]. Disponível em: www.maxprint.com.br/#/produto/602568 [Acessado em 26 de Março. 2017].

Microsoft. (2014). *Technical Data Sheet LifeCam HD-3000*. 01 p.

National Instruments. (2014). *NI LabVIEW Robotics Starter Kit - Data Sheet - National Instruments*. [online]. Disponível em: <http://www.ni.com/datasheet/pdf/en/ds-217> [Acessado em 5 de Abril. 2017].

Pimentel, L. S. S.; Cuadros, M. A. S. L.; Almeida, G. M.; Amaral, R. P.; Gamarra, D. F.T. (2014). Development of a Mobile Robotics Platform for Navigation Tasks Using Image Processing. In: *Asia-Pacific Computer Science and Application Conference (CSAC 2014)*, Shanghai.

Siegwart, R. and Nourbakhsh, I. (2004). *Introduction to autonomous mobile robots*. 1^a Ed. Cambridge, MIT Press, pp.291-298.

Souza, A. A. S. (2012). *Mapeamento robótico 2,5-d com representação em grade de ocupação elevação*, Tese. Programa de Pós-Graduação em Engenharia Elétrica e de Computação da Universidade Federal do Rio Grande do Norte.

Tomasi, E. E. V.; Faria, H. G.; Cuadros, M. A. S. L.; Almeida, G. M.; Resende, C. Z. e Gamarra, D. F.T. (2015) “Estudo Comparativo de Controladores de Seguimento de Trajetória para Robôs de Tração Diferencial: Fuzzy, Ganhos Fixos e Backstepping”, XII Simpósio Brasileiro de Automação Inteligente (SBAI).