Comparação entre métodos de odometria e extração de características para o robô Pioneer P3-DX

ANDERSON NOGUEIRA COTRIM
Engenharia de Computação
Graduação
ra163993@students.ic.unicamp.br

DIOGO HIDEKI SHIRAISHI
Engenharia de Computação
Aluno Especial
diogohshiraishi@gmail.com

RICARDO AKIRA NAGAISHI
Engenharia de Computação
Aluno Especial
r.a.nagaishi@gmail.com

***Resumo* – Este é o primeiro projeto da matéria de robótica móvel, seu objetivo é fazer uma introdução ao uso do simulador em robótica V-REP, navegar pelo cenário a partir de comandos simples de controle, mapear o ambiente utilizando sensores propícios e realizar o estudo da odometria do robô de maneira a comparar os resultados com a posição real.**

De maneira geral, o trabalho visa destacar a importância em escolher os sensores mais apropriados para certas tarefas e realizar um estudo sobre a precisão dos modelos de odometria, destacando os motivos das falhas para cada extração de característica, reforçando a ideia de utilizar vários sensores para resolver certo problema de maneira que um sensor supra a deficiência de outro no objetivo de proporcionar a melhor experiência de sensoriamento.

***Palavras-chave* – robô diferencial, modelo cinemático, extração de características, simulação, odometria, localização, mapeamento**

I. INTRODUÇÃO

No estudo de robótica móvel o principal ponto de partida é a implementação do modelo cinemático de um robô, que consiste em estudar a estrutura do mesmo, como sua forma de locomoção, momento de inércia, raio do corpo, das rodas, estudo preditivo de sua posição e orientação, além de identificar sensores que podem ser adicionados com o intuito de auxiliar nesse processo de exploração do ambiente.

É importante também ressaltar que o estudo inicial é sempre realizado em ambiente simulado devido a sua simplicidade em adicionar e remover sensores ao robô, rapidez na validação de ideias e na possibilidade de comparar as informações obtidas com as verdadeiras.

Desta maneira este trabalho visa retratar em simulação o estudo da cinemática do robô Pioneer P3-DX, utilizando-se de sensores como sonar, laser *rangefinder*, giroscópio e câmera para navegar e retirar informação do ambiente em tempo real. Adicionalmente foi realizado um estudo da odometria do robô, comparando o deslocamento real, fornecido pelo simulador, com a odometria auxiliada por sensores de detecção de mudanças na orientação.

O trabalho está dividido em quatro sessões. A primeira busca explicar o trabalho proposto destacando os problemas tratados; a segunda está focada em destacar e explicar os materiais utilizados durante o experimento e a metodologia utilizada nos estudos realizados, bem como testes realizados; a terceira sessão procurará expor e discutir os resultados obtidos expondo-os ao leitor; a quarta e última sessão será composta pela conclusão frente aos resultados obtidos e ao trabalho como um todo, destacando pontos positivos e negativos do mesmo.

II. TRABALHO PROPOSTO

O trabalho consistiu em utilizar o ambiente de simulação V-REP para simular o comportamento de um robô modelo Pioneer P3-DX já incluído em uma cena previamente fornecida (Figura 1). O modelo do robô apresenta sensores ultrassônicos em todo seu diâmetro não sendo portanto necessário adicioná-los ao modelo. Contudo, foi necessária a adição de outros sensores tais como sensor laser, câmera e giroscópio. Pode-se dividir o objetivo do trabalho em quatro partes principais:

- Compreender o funcionamento do ambiente de simulação V-REP e sua integração com o controle através de um script externo
- Modelagem de trajetória do modelo P3-DX de maneira que consiga percorrer uma área expressiva do cenário sem ficar preso nos obstáculos presentes
- Integração e utilização de diferentes sensores, compreendendo o modo de atuação de cada um e como obter remotamente os dados, tratá-los e utiliza-los para a extração de características do meio
- Aplicação dos conceitos de odometria para estimativa de posição e orientação, utilizando estes dados como base para a extração de características e comparar os resultados obtidos.

III. MATERIAIS E MÉTODOS

Para a simulação no V-REP foi utilizada a comunicação externa via Remote API. Esta comunicação permite o controle da simulação por meio de um *script* externo em qualquer uma das linguagens suportadas (C++, Java, Python, Matlab), o que adiciona agilidade na execução e tratamento dos dados provenientes da simulação.

É necessário primeiramente abrir conexão de um servidor Remote API atrelado à simulação através do comando *simRemoteApi.start(25000, 250, true)*. Dessa forma, o servidor fica aberto a receber conexões de qualquer programa cliente em execução no mesmo endereço de rede e executando na

Figura 1. Visão superior da cena do V-REP utilizada para os testes com robô diferencial Pioneer P3-DX

mesma porta, bastando para isso realizar a conexão também por parte da aplicação cliente.

Para o projeto foi escolhida a linguagem Python 3 para a programação do lado cliente da aplicação, devido a existência nesta linguagem de várias bibliotecas facilitadoras para visualização dos dados.

Primeiramente foi necessário a familiarização com o simulador, através de um programa simples que objetivava mover o robô pelo cenário. Uma vez realizada esta tarefa, foi necessário estabelecer um algoritmo de movimentação do modelo P3-DX, que consistiu em utilizar os sonares distribuídos pela circunferência do robô, recuperar a posição do obstáculo relativa ao sensor e por meio da posição do sonar calculada previamente, determinar a distância entre os dois e utilizar essa informação para evitar colisões. Dessa maneira adotou-se uma abordagem do paradigma reativo da seguinte forma: originalmente o robô caminha para frente enquanto nenhum sensor detecta obstáculos, quando há detecção em alguma das laterais cuja distância seja menor que 0.4 metros, o robô realiza uma curvatura para o outro lado, escapando do contato com os obstáculos. A movimentação é realizada pelo método *robot.drive()*, que recebe como parâmetros a velocidade linear e angular respectivamente.

Para a extração de características do meio foi determinada a realização de um mapeamento do ambiente utilizando-se os sensores que foram julgados mais adequados para este fim. Inicialmente, foram utilizados os sonares, onde cada obstáculo detectado no alcance do conjunto de 16 sonares são coletados no referencial de cada sonar e, portanto, devem ser transformados para a coordenada global através da posição real (*Ground Truth*) do robô, uma vez transformados são plotados no mapa.

Foi utilizado também um sensor laser para realizar esse mesmo mapeamento, sensor este que é mais adequado para este fim, pois consegue abranger uma área maior que os sonares e sua detecção é precisa independentemente da distância do obstáculo ao sensor, algo que não ocorre nos sonares cuja precisão cai com a distância do obstáculo ao mesmo, devido à natureza cônica do feixe de detecção. Desta maneira, foi utilizado o sensor *Hokuyo URG 04LX UG01 fast* cuja utilização é bastante simples, pois sua detecção retorna uma nuvem de pontos com as posições dos obstáculos detectados dentro do círculo de atuação do laser, bastando tratar os dados, retirando as coordenadas em x e y e transformá-las do referencial do sensor para o referencial global, plotando o resultado no mesmo mapa.

Foi também adicionado uma câmera responsável por capturar informações do ambiente de simulação, detectando linhas retas, o que pode ter utilidade no futuro. O processo de captura da imagem ocorre por meio da chamada do método *simxGetVisionSensorImage()*, retornando um *array* com as informações de x, y e *rgb* codificadas em *UINT8* (*char*). Para detecção de retas foi necessário aplicar primeiramente um filtro de Canny para binarização da imagem seguido de uma transformada de Hough probabilística para detecção de linhas, rotinas utilizadas através da biblioteca OpenCV, assim como o plot da mesma na tela.

Para a odometria do robô, foram avaliados três métodos clássicos para diversas condições de velocidades:

1) Odometria *simples*: é computada utilizando apenas velocidade aplicada em cada um dos motores.
2) Odometria por *encoder*: o deslocamento linear e angular do robô é calculado utilizando um *encoder*, um sensor onde é possível extrair a posição angular dos eixos dos motores.
3) Odometria por *encoder* e o giroscópio: neste método o deslocamento linear é calculado utilizando *encoder*, mas o deslocamento angular é calculado através do dado obtido pelo giroscópio.

O software foi desenvolvido modularizando-o em três classes: *Simulator*, responsável por estabelecer a conexão com o servidor Remote API e implementar métodos mais gerais de obtenção de *handles* e inicialização de sensores; *Robot*, responsável por chamar os métodos de *Simulator*, obter a posição e orientação momentânea, rotinas de movimentação de acordo com o modelo cinemático; por fim a classe *Main*, responsável por instanciar um objeto de Robot e chamar através dele os métodos relevantes para o estudo.

Para a obtenção dos resultados o programa executa durante aproximadamente 100 segundos, onde vai acumulando os pontos em um mapa, ao final de sua execução ocorre o plot do mapa com as informações discriminadas por cor.

IV. Resultados e Discussão

Após executar o programa algumas vezes, alterando a quantidade de dados incluídos no plot final, os seguintes resultados foram obtidos:

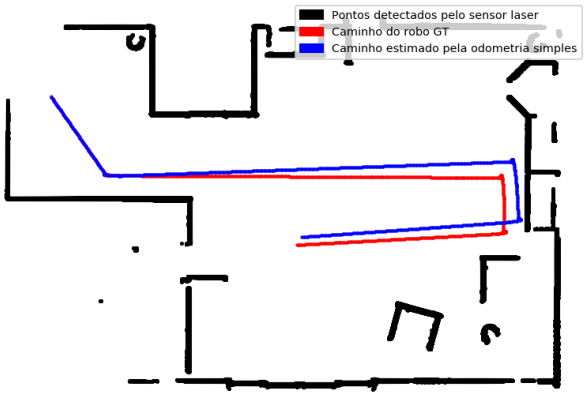

Figura 2. Odometria simples para o caso do robô submetido a uma velocidade baixa: 20cm/s em linha reta e 0.15 rad/s nos desvios

Em uma primeira análise da Figura 2, pode-se notar que enquanto não houve alteração na orientação do robô, a odometria conseguiu se igualar com o caminho real traçado P3-DX. Contudo, ao haver mudança de orientação, a odometria passou a se distanciar do real, devido ao acumulo de erro proveniente de inúmeros fatores (que são problemas inerentes à física do robô) como escorregamento das rodas durante uma curva, atrito com o solo etc. No entanto, grande parte do erro advém do motivo da odometria não ser feita por *feedback* do deslocamento das rodas, pois este modelo não leva em conta a aceleração assumindo velocidade instantânea de acordo com o sinal fornecido ao motor. Portanto ele é muito ineficiente quando aumentamos bruscamente a velocidade como na Figura 3.

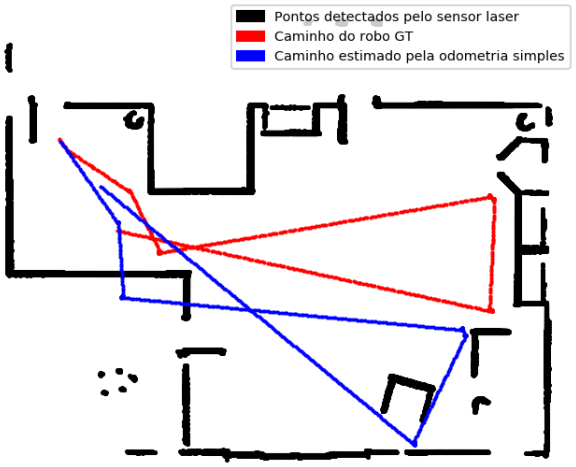

Figura 3. Odometria simples para o caso do robô submetido a uma velocidade média: 50cm/s em linha reta e 0.30 rad/s nos desvios

Neste exemplo a odometria distanciou-se muito do caminho real. Isso reforça mais a ideia de que a odometria sofre influência direta do ambiente, uma vez que a velocidade empregada nesse teste foi aumentada, o que aumenta também o atrito e derrapamento das rodas. Mas, o principal motivo do distanciamento do caminho real, como já mencionado, foi a aceleração que é ligeiramente maior do que na Figura 2. Além desse cenário de alta velocidade, esse modelo de odometria se torna completamente errado quando há algum impedimento externo para as rodas girarem, novamente uma limitação devido a falta de *feedback* que poderia ser feito pelos *encoders*.

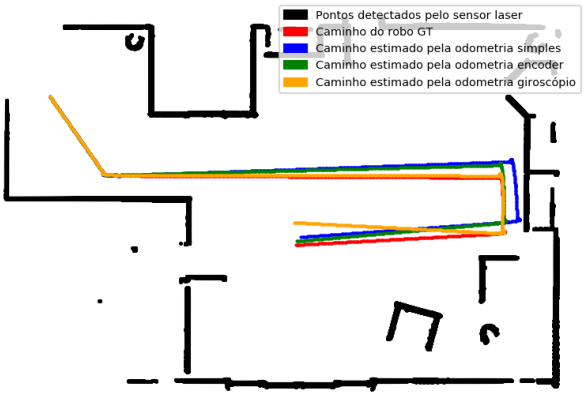

Figura 4. Comparação entre os três métodos de odometria para velocidades baixas: 20cm/s em linha reta e 0.15 rad/s nos desvios

Neste caso da Figura 4, foi inserido a odometria utilizando a posição do *encoder* e outra utilizando o giroscópio para obter a angulação momentânea do robô ao invés do calculo baseado na velocidade angular. Embora, para pequenas velocidades todos os métodos funcionam relativamente de forma similar, pode-se notar que a odometria por *encoder* aproximou-se mais levemente do caminho real e a com o giroscópio em grande parte coincidiu com o real.

Mais uma vez pode-se perceber o impacto da velocidade na determinação da odometria nas Figura 5 e Figura 6, porém de forma bem menor do que o método simples. Na primeira, há a aplicação de uma velocidade menor que na segunda, sendo a odometria mais imprecisa a medida que a velocidade aumenta, levando também em consideração as alterações de direção em sua trajetória. O desalinhamento dos *encoders* durante as curvas pode ser uma fonte de erro considerável.

Uma posterior comparação foi realizada entre o mapeamento do ambiente por sonares e por laser. Pode-se constatar uma inerente maior imprecisão dos pontos adquiridos pelos sonares, dada as circunstâncias apontadas anteriormente, como o fato de sua coletar dar-se por um cone de sensoriamento. Além disso, o alcance dos sonares é relativamente pequeno, quando comparado com o laser. Dada as mesmas condições espaciais e de locomoção do robô, vê-se uma menor cobertura do ambiente pelos sonares. Deve-se relevar que ambos os mapeamentos foram gerados utilizando a posição real do robô, relativa ao mundo.

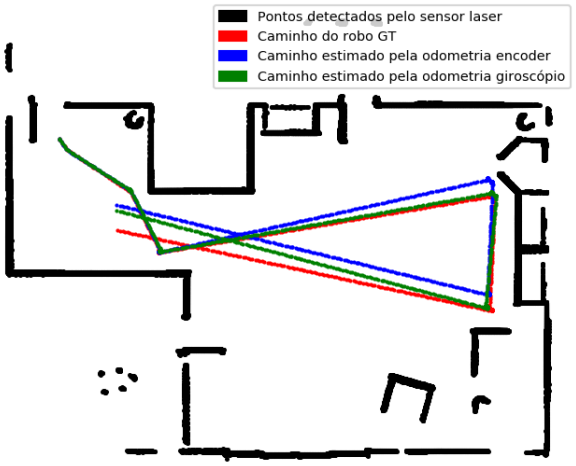

Figura 5. Comparação entre odometria por *encoder* e giroscópio para o caso do robô submetido a uma velocidade média: 50cm/s em linha reta e 0.30 rad/s nos desvios

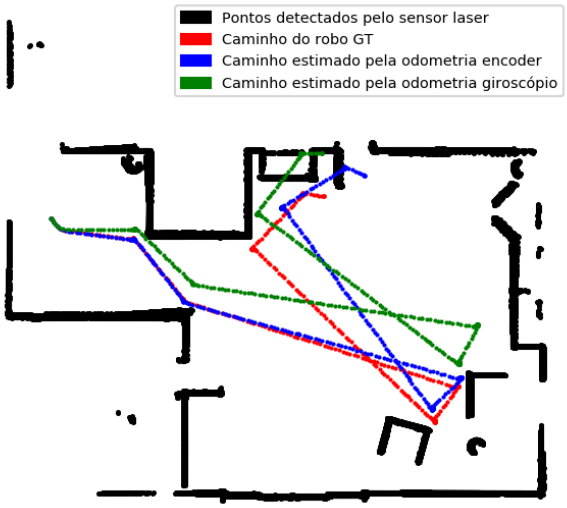

Figura 6. Comparação entre odometria por *encoder* e giroscópio para o caso do robô submetido a uma velocidade alta: 80cm/s em linha reta e 0.40 rad/s nos desvios

V. Conclusões

Através do trabalho desenvolvido, pode-se concluir que em uma aplicação de robótica móvel deve-se atentar para as características de cada sensor empregado, para assim conseguir uma melhor escolha baseada nas informações que se deseja obter do ambiente. Por exemplo, o sonar se mostrou ótimo para aplicações de navegação, mas mostrou-se deficiente em tarefas de mapeamento, algo que foi cumprido com perfeição utilizando-se do sensor laser. A câmera é um sensor recomendado para a extração de características do ambiente como segmentação de cor, determinação e caracterização de objetos.

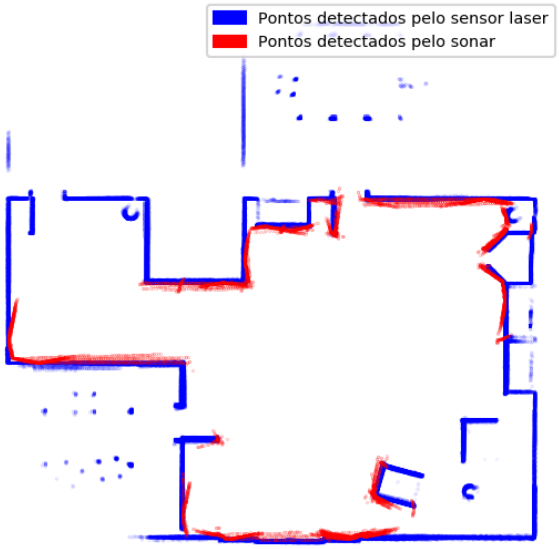

Figura 7. Comparação entre o mapeamento por sonares e por laser *rangefinder* - ambos utilizando a posição real *ground truth* do robô

Dessa forma, a determinação dos sensores utilizados para cada fim é algo de fundamental importância para qualquer projeto de robótica móvel, impactando na qualidade e na simplicidade da solução empregada no robô.

Em relação a odometria, foi acertada a estratégia de utilizar o giroscópio para a definição da orientação do robô ao invés da estimativa pela variação angular das rodas. A utilização do *encoder* também proporcionou uma melhora na estimativa de posição. Foi também evidente a influência dos fatores externos como atrito e escorregamento das rodas na imprecisão da odometria.

Referências

[1] Coppelia, "Remote API Functions (Python)." [Online]. Available: http://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm

[2] Nikolai, "Python robot simulation with v-rep — v-rep tutorial." [Online]. Available: https://www.youtube.com/watch?v=SQont-mTnfM

[3] R. N. SIEGWART, *Introduction to Autonomous Mobile Robots*. Cambridge, Massachusetts: MIT Press,, 2004.