Projeto 1 - Robótica Móvel

OSVALDO XAVIER DIAS JUNIOR E EDUARDO DE OLIVEIRA MORAES *

*Ciência da Computação - Pós-Graduação
E-mail: osvaldoxdjr@gmail.com
eduardomoraesmail@gmail.com

***Resumo* – Este relatório trata o projeto 1 (P1) da disciplina de robótica móvel, o trabalho é sobre o simulador robótico V-REP, um sistema de cômputo da odometria e extração de características para o robô Pioneer P3-DX, isso é relevante pois abrange os conceitos básicos sobre simualação de robôs em um ambiente de software. Utilizou-se rotinas em Python para comunicação com o simulador V-REP, com o propósito de efetuar a localização do robô e o mapeamento da cena. Foram testados, durante o desenvolvimento, sensores como encoder para odometria, e sonar e laser 2D para mapeamento, os resultados obtidos foram bons, com exceção da odometria para longas distâncias percorridas pelo robô. A parte de extração de características ficou por conta da aplicação das transformadas de Hough sobre a imagem da nuvem de pontos, os segmentos de reta do mapa foram encontrados com êxito.**

***Palavras-chave* – V-REP, P3-DX, Robô, odometria, extração de características, transformada de Hough e ground truth**

I. INTRODUÇÃO

O desafio de desenvolver robôs é notoriamente grande e complexo, devido a interdisciplinaridade de assuntos que envolvem, majoritariamente, engenharia. Posta essa grande complexidade, sabe-se que para confeccionar um robô é requerido um determinado custo, que em alguns casos podem ser muito elevados. Logo, existem softwares que são capazes de fazer a simulação de robôs, como por exemplo o V-REP que foi utilizado nesse projeto 1.

O V-REP, em linhas gerais, é um simulador no qual faz-se a escolha de uma dinâmica de operação, e também pode-se colocar objetos como sensores, atuadores, robôs, anteparos, e etc. na cena (ambiente simulado no qual um protótipo de robô será inserido). Uma das formas de tomar ações é através de comunicação remota com os objetos da cena utilizando-se de diferentes linguagens de programação como C/C++, Python, Java, Matlab, Octave e Lua, além disto os objetos podem possuir rotinas de simulação, ou childscripts, que estão codificadas em Lua; estas rotinas são pré-programações dos objetos para tomar determinadas ações quando a simulação é inciada.

Um ponto fundamental quando se deseja desenvolver um robô móvel é localização, pois esta é a premissa básica para que o robô possa se mover de um ponto a outro, sem interação humana. A localização consiste de ter conhecimento de onde o robô está localizado com referência ao referencial da cena simulada, dada a importância desse tópico existe um ramo da robótica que estuda uma parte desse problema que é a odometria. Esta consiste no estudo da posição do robô através do deslocamento incremental de sua mecânica de locomoção,

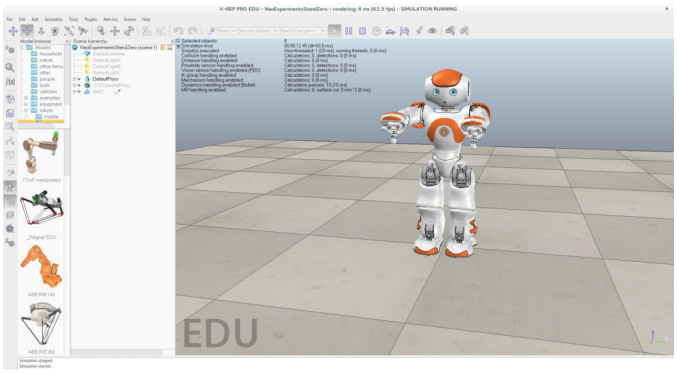

Figura 1. Ambiente de simulação V-REP

por exemplo através do cômputo da movimentação do robô através da velocidade angular de rodas. Para o robo diferencial (robô que movimenta-se através de duas rodas montadas num eixo comum e controladas por motores independentes, um para cada roda) tem-se que o cálculo da odometria é dado por:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_t = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_0 \sum_{k=1}^{t} \begin{bmatrix} \Delta_s cos(\theta + \frac{\Delta\theta}{2}) \\ \Delta_s sen(\theta + \frac{\Delta\theta}{2}) \\ \Delta\theta \end{bmatrix} \quad (1)$$

$$\Delta_s = \frac{R(Vr + Vl)}{2}\Delta_t \quad (2)$$

$$\Delta_\theta = \frac{R(Vr - Vl)}{2L}\Delta_t \quad (3)$$

Onde x, y e z são as coordenadas; Δ_s é a distância percorrida; Δ_θ é a variação angular; R é o raio das rodas; L é a distância entre as rodas; Vr é a velocidade da roda direita e Vl é a velocidade da roda esquerda.

Com as inforamções acima é possível estimar a trajetória do robô ao longo do tempo, porém nota-se pela somatória temporal que pode ocorrer um acúmulo de erro ao longo do tempo, resultando em uma trajetória diferente daquela indicada pelo ground truth.

O mapeamento da cena consiste da elaboração de um mapa do ambiente simulado, e está intimamente atrelado a movimentação do robô, pois através daquele é possível movimentar o robô com maior eficiência evitando obstáculos e executar deslocamentos mais rapidamente, uma vez que, sabe-se exatamente a trajetória a ser percorrida. O mapeamento

pode ser realizado utilizando-se diversos sensores como sonares, radares, lasers, cameras e etc. ou também utilizando um conjunto dos componentes citados anteriormente.
Contudo, uma questão fundamental em localização, odometria e mapeamento é o comparativo entre referencial inercial (referencial global) e referencial de um objeto qualquer. Quando o robô realiza qualquer medição que envolva posicionamento em cena, é preciso sempre tomar a precaução de trabalhar com coordenadas em referencial inercial. Uma ferramenta para solucionar esse problema é utilizar-se de conceitos de algebra linear - matrizes de rotação e translação:

$$T_{trans} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$$T_{trans} = \begin{bmatrix} cos(\alpha) & -sen(\alpha) & 0 \\ sen(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$Pos_I = T_{trans} * T_{rot} * Pos_r \tag{6}$$

Onde Δx representa a variação das coordenadas x dos referenciais, Δy representa a variação das coordenadas y dos referenciais e α representa o ângulo entre os refenciais. A equação 6 é utilizada para rotacionar e transladar o ponto de um referencial local para o referencial global, respeitando o princípio de não comutatividade de matrizes.
Para extrair características do mapeamento de uma cena é possível utilizar a transformada de Hough que é uma técnica matemática que realiza a detecção de formas geométricas em imagens digitais. No caso específico de um mapa 2D seria a detecção de linhas a partir de uma nuvem de pontos.
A partir daqui este trabalho encontra-se organizado da seguinte forma: a seção 2 apresenta "Trabalho Proposto". A seção 3 descreve "Materiais e Métodos". Os "Resultados e Discussão"são apresentados na seção 4, e as "Conclusões"são apresentadas na seção 5.

II. Trabalho Proposto

As soluções propostas foram baseadas em mapeamento de cena utilizando sonar e laser, cálculo de odometria utilizando o enconder das rodas laterais do robô e extração de característica utilizando transformada de Hough. Segue a lista tarefas executadas, em sequência cronológica, durante o desenvolvimento:

1) Ambientação com ambiente de simulação
2) Localização via ground truth
3) Localização via odometria
4) Mapeamento através de sonar
5) Mapeamento através de laser 2D
6) Extração de característica

III. Materiais e Métodos

Para desenvolver o trabalho foi utilizado o remote API do V-REP em Python, baseado em uma comunicação cliente-servidor. O robô utilizado na simulação foi o P3-DX, um robô diferencial que, portanto, possui duas rodas com um motor em cada uma delas e 16 sonares já pré-instalados. Este robô possui um childscript em Lua que executa algumas tarefas, entre elas a navegação através do algorítmo de Braitenben. Este código nativo é capaz fazer a navegação do robô e evitar colisões, considerando a distância medida pelo sonar e ativando os motores das rodas de forma ponderada, esta navegação foi adotada para a execução das simulações. Foi inserido uma linha de código no servidor para que este ficasse esperando a conexão com o cliente.
Já no script do cliente, todo desenvolvido em Python, foi criada a lógica para leitura dos sensores e interpretação dos dados. Porém, antes do desenvolvimento foi feita a inserção de mais dois sensores além daqueles originais: um laser 2D no topo do robô e no mesmo referencial x e y deste, além de um giroscópio exatamente no mesmo referencial do robô.
No código Python, para atuar nos objetos da cena foi necesário coletar os handles (referências) do robô, rodas e sensores. A localização ground truth é observada diretamente do cliente, não sendo necessário nenhum cálculo para tal. Entretanto, para cômputo da odometria foram utilizadas as equações 1, 2 e 3. Em linhas gerais, esses cálculos foram implementados usando listas que acumulavam os valores que tinham que ser adicionados as coordenadas x, y e θ, cabe salientar que foi necessário realizar a conversão dos ângulos dos encoders dos motores para representar valores entre 0 e 2π, ao invés π e -π. Foi tentado utilizar um giroscópio para melhorar o odometria, porém devido a possivelmente um problema da comunicação cliente-servidor os valores apresentados do giroscópio mostraram-se pouco consistentes. A odometria foi compara com o ground truth para verificar se os cálculos estavam sendo obtidos de forma coerente.
O mapeamento através de nuvem de pontos foi feito utilizando-se dois tipos de sensores e duas lógicas de cálculo. Foram utilizados os sensores sonar e laser, e para cálcular o valor da posição absoluta, no referencial inercial, foi utilizado como base o ground truth e a odometria; sendo que as mudança de coordenada local para global foram feitas com as equações 4, 5 e 6. Os mapas obtidos nas 4 configurações diferentes foram comparados com a cena do simulador.
Para extrair linhas dos dados colhidos durante a fase de mapeamento foi utilizada a transformada de Hough Probabilística. A biblioteca OpenCV implementa o algorítmo . Os parâmetros de threshold, minLineLength e maxLineGap foram ajustados de maneira empírica até se obter um resultado satisfatório.

IV. Resultados e Discussão

O primeiros resultados obtidos foram os gráficos do mapa da cena (nuvem de pontos), obtido através dos sensores sonar e laser. Além do mapa da cena, o gráfico também contém a posição do robô através do ground truth e odometria.

Os gráficos da figura 2 e 3 mostram a nuvem de pontos computadas com referência ao ground truth. Com relação ao mapa, estas duas imagens mostram claramente a diferença entre os sensores, o laser mostra-se muito mais preciso comparado ao sonar, haja visto que o mapa do sonar pouco assemelha-se com a cena, enquanto que o mapa do laser

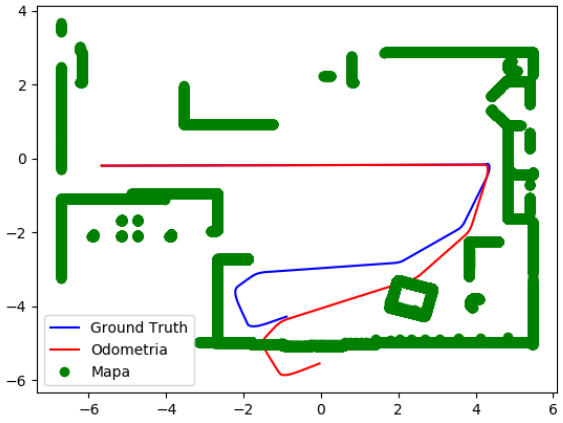

Figura 2. Mapa gerado por ground truth, com adição da posição via odometria e ground truth - sensor laser

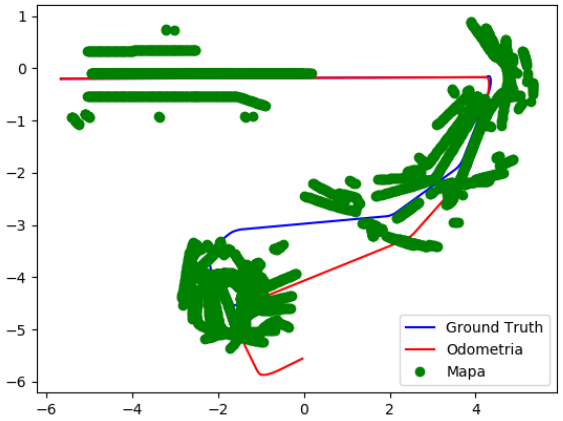

Figura 3. Mapa gerado por ground truth, com adição da posição via odometria e ground truth - sensor sonar

apresenta uma grande verrosimilhança. Soma-se isto aos fatos de que laser apresenta alcance maior que o sonar e uma menor suscetibilidade a interferências como clutters. Em ambas as imagens foram adicionadas a trajetória do robô via ground truth e via odometria, nota-se claramente uma diferença que começa a acentuar-se na primeira curva do robô e segue piorando nas curvas subsequentes, mostrando assim que a odometria é bastante sensível as mudanças de direção. Como o cálculo da odometria é feito de forma acumulativa, o erro é incrementado iteração após iteração resultando em uma posição final bastante diferente do ground truth para a odometria. Um explicação para esta diferença nas trajetórias é que o modelo cinemático do robô é simplista, pois o modelo não considera atrito do chão, escorregamento das rodas e outros fatores que o simulador possui como premissa que existam. As figuras a seguir representam o mesmo experimento realizado anteriormente, entretanto o cômputo da nuvem de pontos é feito com referência na odometria ao invés do ground truth, seguem os resultados:

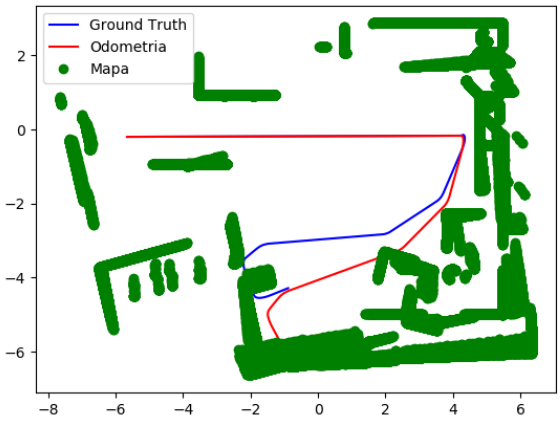

Figura 4. Mapa gerado por odometria, com adição da posição via odometria e ground truth - sensor laser

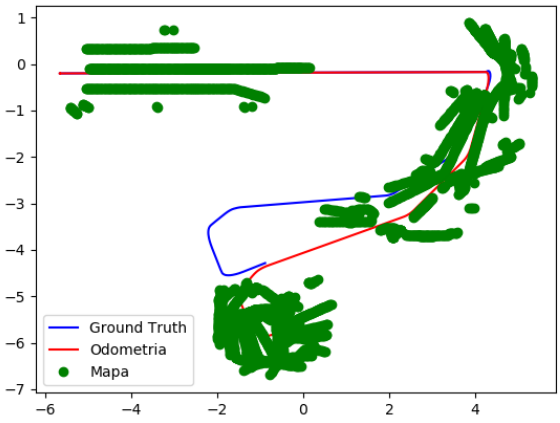

Figura 5. Mapa gerado por odometria, com adição da posição via odometria e ground truth - sensor sonar

Nota-se que a trajetória descrita pelo robô continua sendo a mesma tanto na odometria quanto no ground truth, contudo a nuvem de pontos mostrou-se modificada com relação aquela que foi observada na figura 2 e 3. Isso se deve ao fato de que a odometria representa uma trajetória destorcida do robô, portanto os pontos obtidos com referência a essa trajetória também estarão distorcidos, isso pode ser visto mais claramente comparando as duas figuras obtidas com as medidas do laser (figura 2 e 4), pois estas representam com uma precisão muito maior o mapa da cena em detrimento dos mapas elaborados por sonar.
Verifica-se pelas figuras 6 e 7 que os algorítmos empregados são eficazes em delimitar as linhas que coincidem com as parades da simulação. Embora isso aconteça, aplicá-los em

tempo real não foi possível no contexto deste trabalho. O processamento tornou a simulação lenta e impraticável, portanto optou-se pelo pós processamento das imagens geradas pelas nuvens de pontos com base na odometria, bem como a gerada a partir do ground truth. Além disso, ainda há espaço para melhoras nesse procedimento através do ajuste dos parâmetros tanto da função de detecção de bordas (Canny) como dos parâmetros da função HoughLines da biblioteca OpenCV.

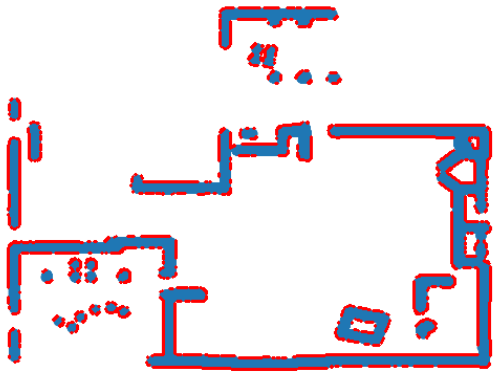

Figura 6. Mapa gerado por ground truth e extração de características - sensor laser

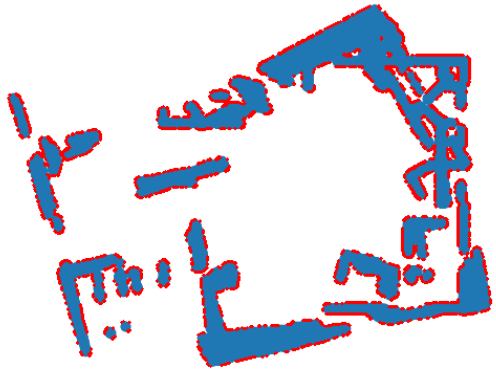

Figura 7. Mapa gerado por odometria e extração de características - sensor laser

V. Conclusões

Pode-se concluir dos experimentos realizados que a odometria quando feita utilizando um único sensor não é uma boa estimativa para a posição do robô, principalmente quando existem muitas curvas na trajetória do robô; o modelo cinemático do robô implementado não atende suficientemente o propósito de obter uma odometria precisa.

Verificou-se a importância da escolha dos sensores para elaborar uma nuvem de pontos, o sonar e o laser apresentaram resultados muito distindos. Além disso, destacamos que o método de extração de caracteristicas escolhido é eficaz, mas consome tempo de processamento, que no contexto da simulação, inviabilizou a utilização em tempo-real. Acreditamos que se faça necessário uma análise mais aprofundada deste ponto e, assim, determinar um algoritmo que balanceie precisão e custo computacional.

Referências

[1] R. Siegwart and Illah R. Nourbakhsh *Autonomous Mobile Robots*, 1st ed. Massachusetts Institute of Technology, 2004

[2] Matas, J. and Galambos, C. and Kittler, J.V *Robust Detection of Lines Using the Progressive Probabilistic Hough Transforms*. OpenCV