

4

Resultados da Simulação

Algumas simulações foram feitas para testar os algoritmos realizados neste trabalho, e são apresentados neste capítulo. Os resultados da simulação são divididos em : análise de "Correspondência de Varreduras" de dados obtidos mediante a simulação do movimento do robô móvel, do sensor LRF e do ambiente; a solução do problema de SLAM mediante Correspondência de Varreduras; e a solução do problema de SLAM usando o algoritmo DP-SLAM.

4.1

Correspondência de Varreduras

O ambiente de simulação, onde o robô móvel faz o seu movimento, foi idealizado para avaliar alguns parâmetros relevantes para o processo de otimização, tais como o tamanho de grade para o algoritmo NDT, o número de iterações, e o tamanho de população, importantes no processo de otimização mediante Evolução Diferencial (Algoritmos Genéticos).

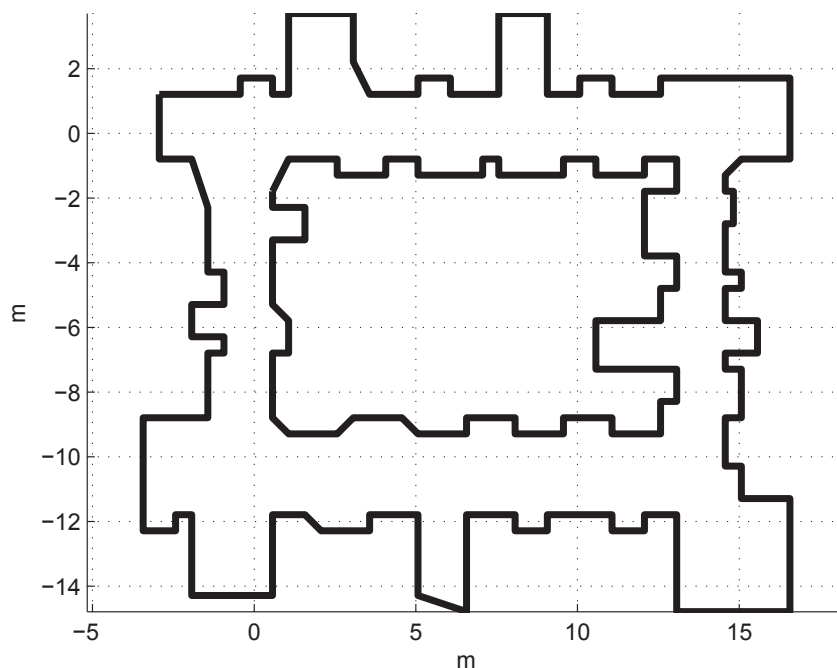


Figura 4.1: Ambiente Simulado

Para um ambiente simulado mostrado na Figura 4.1, durante o movimento do robô móvel dentro do ambiente foram tomadas 227 leituras com o LRF.

Uma vantagem de realizar simulações é que a trajetória do robô móvel dentro do ambiente é conhecida. Assim os valores dos parâmetros do processo de otimização ($\Delta x, \Delta y, \Delta \theta$) são perfeitamente conhecidos e comparados com os obtidos mediante o algoritmo de Correspondência de Varreduras.

É importante a análise do tamanho de grade pois o algoritmo NDT, que aproxima uma função objetivo entre duas varreduras sucessivas do sensor, dela depende, como esta explicado no Capítulo 2. Assim, analisou-se o erro para uma grade $1,0m$ e de $0,5m$. A Figura 4.2 apresenta o erro dos parâmetros ($\Delta x, \Delta y, \Delta \theta$), estimados pelo algoritmo de Correspondência de Varreduras.

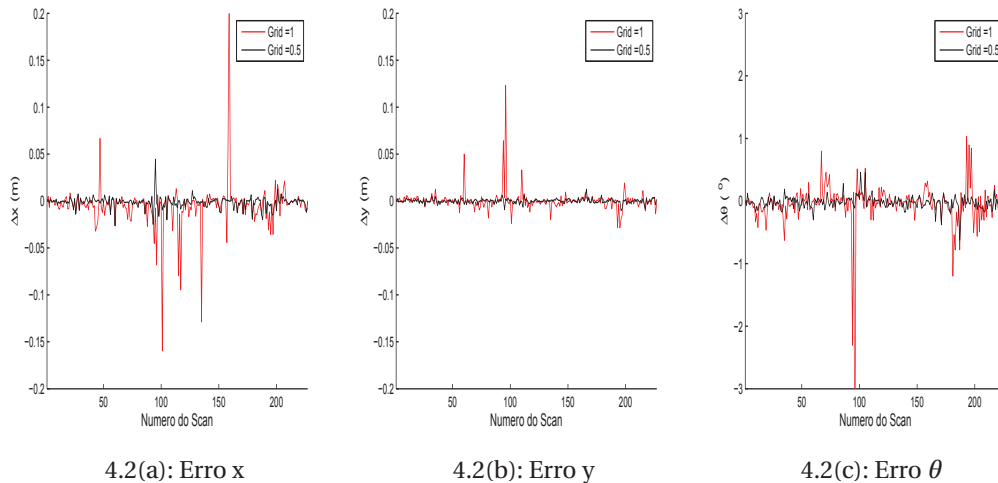
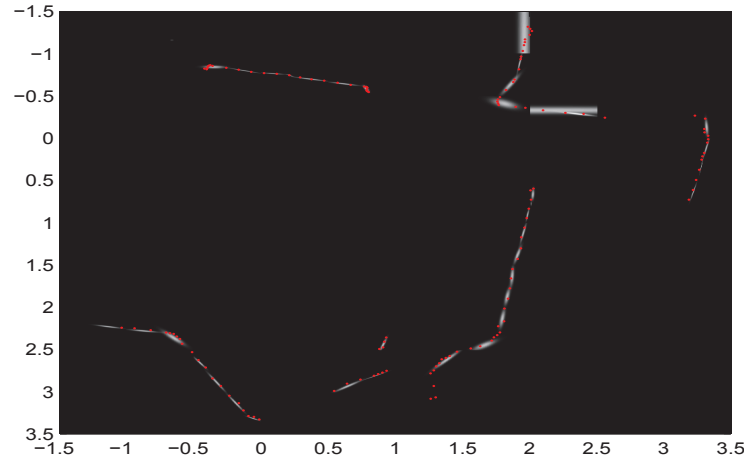


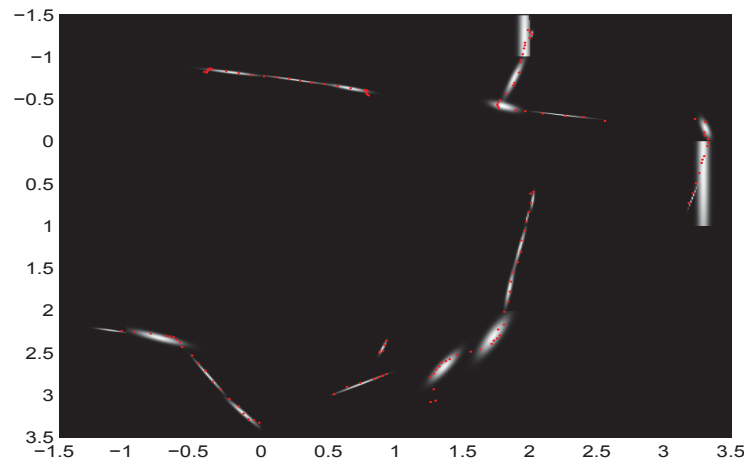
Figura 4.2: Erro da Correspondência de Varreduras para duas tamanhos de grade ($0,5m$; $1,0m$)

Uma das características do sensor *URG – 04LX – UG01* é sua alta densidade de pontos (cada varredura possui 682 pontos), o que facilita considerar um tamanho de grade pequena, porque existe uma maior probabilidade de haver mais de 3 pontos da Varredura dentro da grade. Mas o custo computacional é maior, por outro lado, um tamanho de grade alto piora a qualidade da função estimada. A Figure 4.3 mostra que em (a) a região branca é a melhor aproximação da Varredura, mas para um tamanho de grade maior $1,0m$ aparecem algumas descontinuidades, como se mostra em (b). Para os demais testes e simulações neste trabalho, o tamanho de grade é fixado em $0,5m$.

Numa otimização por Algoritmos Genéticos (Evolução Difetencial), descrito no Capítulo 2, a definição dos operadores genéticos é fundamental para a



4.3(a): Grade 0,5m



4.3(b): Grade 1,0m

Figura 4.3: Algoritmo NDT para duas resoluções de Grade

qualidade do processo evolutivo. Uma vez definida a representação cromossômica, é necessário estabelecer operadores que estejam de acordo tanto com a representação escolhida quanto com as características do problema de otimização.

O algoritmo de Evolução Diferencial depende dos operadores genéticos tais como o tamanho da população, número de iterações, taxa do crossover e taxa de mutação. A taxa de crossover utilizada neste é de $CR = 0,95$ e a taxa de mutação utilizada $SF = 1,0$, de acordo com o trabalho prévio[2], estes valores foram testados aqui. Nas figuras a seguir 4.4-4.9 apresentamos o comportamento do erro dos parâmetros ($\Delta x, \Delta y, \Delta \theta$) para diferentes números de populações e iterações.

Assim, mantendo o número de iterações em 50, para um tamanho da grade de $0,5m$, mas variando o tamanho da população, o erro dos parâmetros do algoritmo de Correspondência de Varreduras é analisado. A Figura 4.4 mostra o erro para uma população de 30 (vermelho) e uma população de 40 (azul). A Figura 4.5 mostra o erro para uma população de 50 (vermelho) e uma população de 60 (azul). E a Figura 4.6 mostra o erro para uma população de 70 (vermelho) e uma população de 100 (azul).

Na Figura 4.4 podemos observar que, para valores pequenos de população, o processo de otimização por evolução diferencial não converge totalmente, mas, à medida que aumenta o tamanho da população, melhora a convergência. A Figura 4.5 apresenta uma convergência em quase todos os pontos, e para valores altos de população a convergência é estabilizada, ver Figura 4.6.

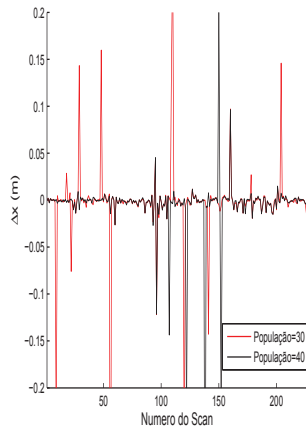
Outro parâmetro a considerar no processo de otimização por Evolução Diferencial é o número de iterações. Assim varia-se o número de iterações, mas para um tamanho da grade $0.5m$ e tamanho da população mantido constante igual a 100. As Figuras 4.7, 4.8 e 4.9 apresentam os erros dos parâmetros ($\Delta x, \Delta y, \Delta \theta$), estimados pelo algoritmo de Correspondência de Varreduras.

A Figura 4.7 apresenta o erro para um número iterações baixas, no qual se pode observar que a convergência não é muito boa. Mas, para um número de iterações maior, a convergência melhora, ver Figuras 4.8 e 4.9. Observe que, para valores muito grandes de iterações, a convergência não varia muito. Mas consomem tempo de processamento, e portanto o número de iterações é fixado em 50 para os demais testes. A Tabela 4.1 apresenta os valores dos parâmetros a serem utilizados com dados reais.

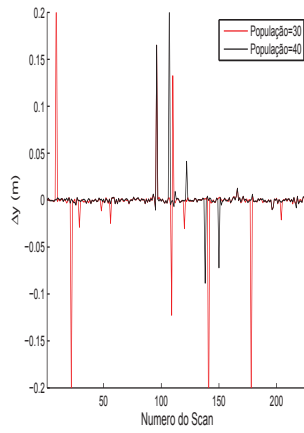
Tabela 4.1: Parâmetros a serem utilizados nos experimentos.

Tabela de Parâmetros

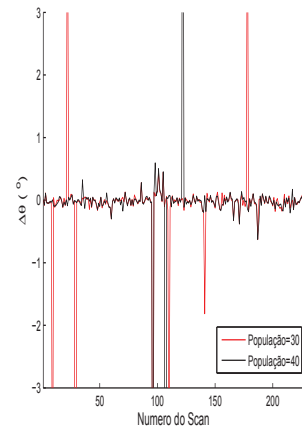
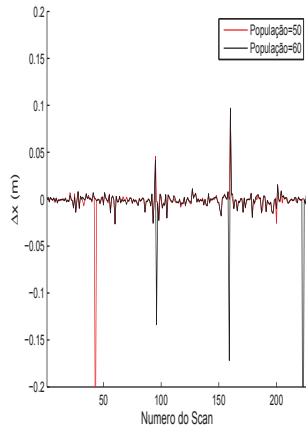
Algoritmo	Parâmetro	Valor
Evolução Diferencial	Mutação	1,0
	Crossover	0,95
	População	100
	Gerações	50
NDT	Grade	0,5



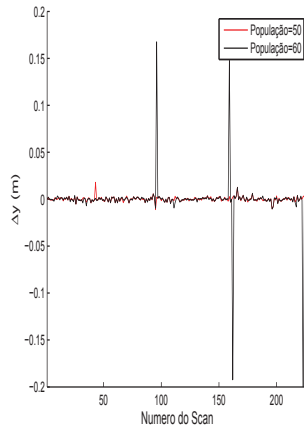
4.4(a): Erro x



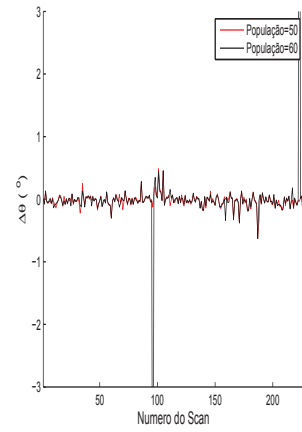
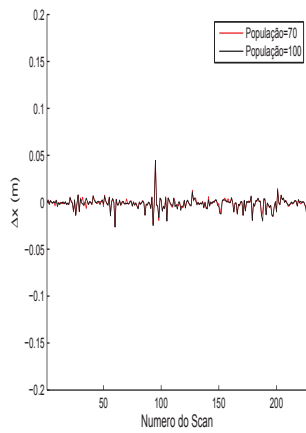
4.4(b): Erro y

4.4(c): Erro θ Figura 4.4: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Pequenas Populações.

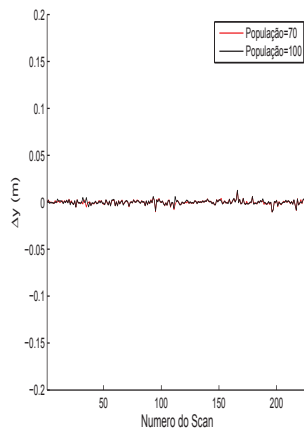
4.5(a): Erro x



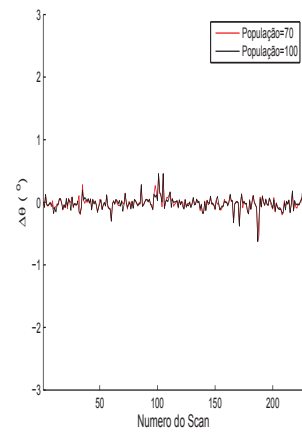
4.5(b): Erro y

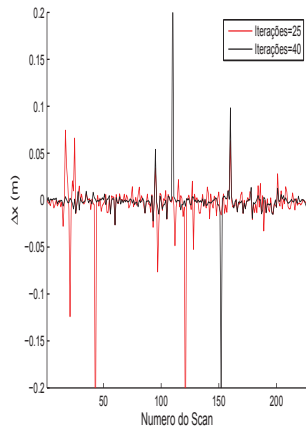
4.5(c): Erro θ Figura 4.5: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Populações Médias.

4.6(a): Erro x

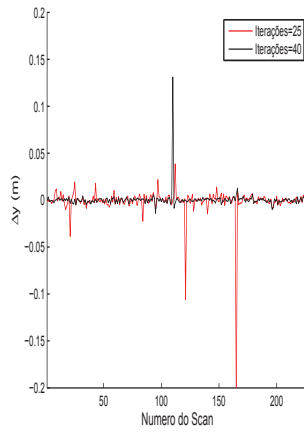


4.6(b): Erro y

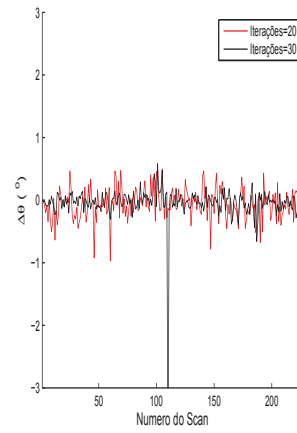
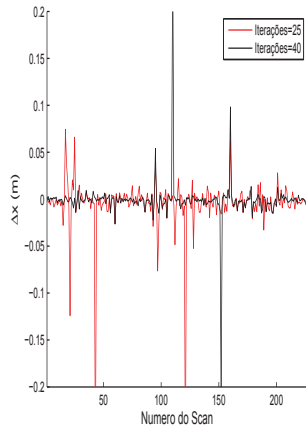
4.6(c): Erro θ Figura 4.6: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Grandes populações.



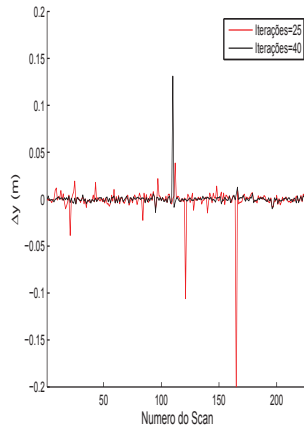
4.7(a): Erro x



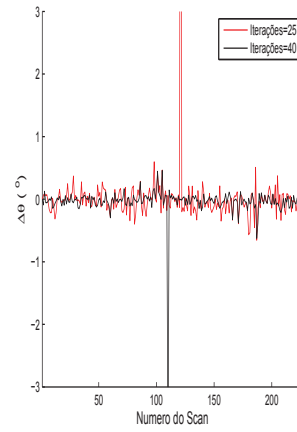
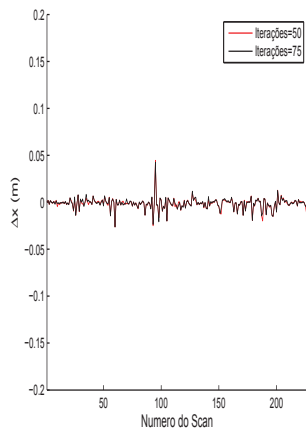
4.7(b): Erro y

4.7(c): Erro θ Figura 4.7: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Iterações 20 e 30.

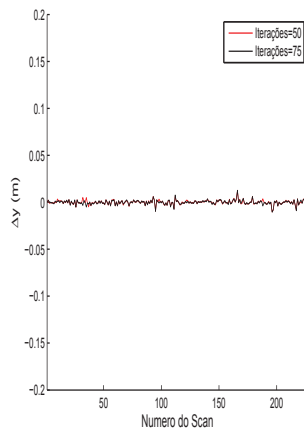
4.8(a): Erro x



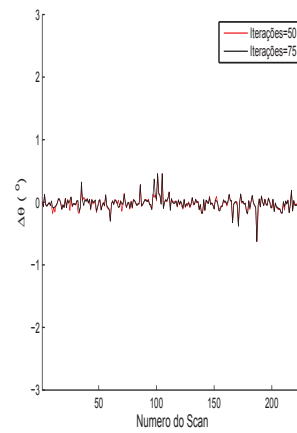
4.8(b): Erro y

4.8(c): Erro θ Figura 4.8: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Iterações 25 e 40.

4.9(a): Erro x



4.9(b): Erro y

4.9(c): Erro θ Figura 4.9: Erro do deslocamento, $(\Delta x, \Delta y, \Delta \theta)$. Iterações 50 e 75.

4.2

SLAM mediante Correspondência de Varreduras

O Algoritmo de Correspondência de Varreduras pode ser aplicado para resolver o problema de SLAM. Foi mostrado que é possível obter o deslocamento $(\Delta x, \Delta y, \Delta \theta)$ de cada leitura do sensor. Executando uma transformação de coordenadas para todas as leituras do sensor, de acordo com o deslocamento de duas Varreduras sucessivas, pode-se obter a trajetória e o mapa do robô móvel.

Assim, o mapa adquirido pelo algoritmo de Correspondência de Varreduras é apresentado na Figura 4.10. Note que o erro de cada Correspondência de Varreduras é cumulativo, portanto no momento de fechar o mapa em um circuito fechado pode ocorrer um desalinhamento. A Figura 4.10 mostra o mapa criado por Correspondência de Varreduras e o desalinhamento ao final de sua trajetória. Deve-se notar que a estrutura principal do ambiente é mantida, apesar do erro.

Da mesma maneira como o mapa foi criado por Correspondência de Varreduras, a trajetória do robô móvel é apresentada na Figura 4.11, Uma vez que

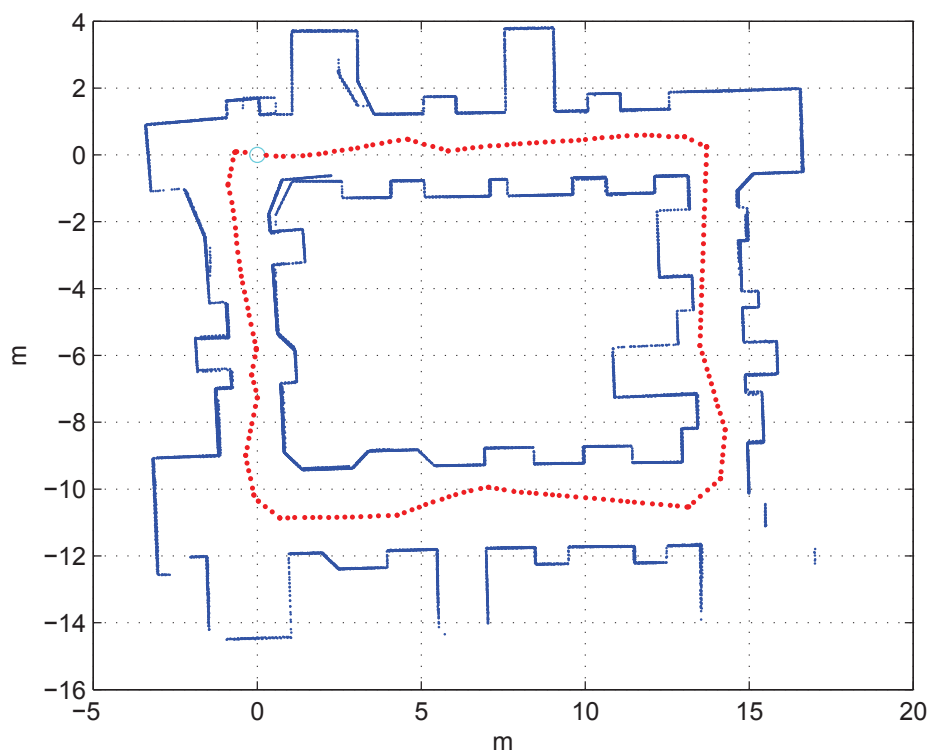


Figura 4.10: Mapeamento do Ambiente Simulado mediante Correspondência de Varreduras

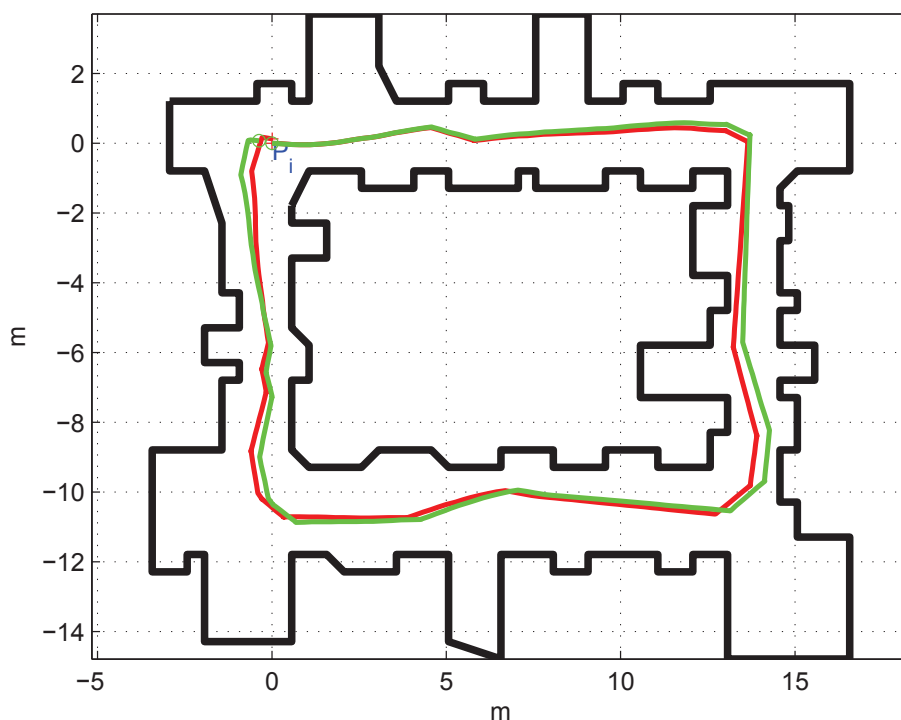


Figura 4.11: Trajetória do Ambiente Simulado. Trajetória real(vermelho) e Trajetória Estimada (verde)

o mapa foi criado por software, a trajetória original é conhecida e pode ser comparada com a trajetória estimada pelo algoritmo de Correspondência de Varreduras. A linha vermelha representa a trajetória real do robô móvel, enquanto a trajetória estimada é mostrada pela linha verde. No início do movimento, como o erro acumulado é pequeno, as duas trajetórias são praticamente iguais, mas com o passar do tempo, o erro acumulado aumenta e a trajetória estimada difere da original, ver Figura 4.11.

4.3 DP-SLAM

SLAM mediante o algoritmo de Correspondência de Varreduras apresenta uma aproximação do mapa e da trajetória do ambiente. Para melhorar o mapa criado, neste trabalho se utiliza o algoritmo DP-SLAM, que fornece uma representação de grade de ocupação para o mapa. Além disso, DP-SLAM é usado para reduzir os erros produzidos pelo algoritmo de Correspondência de Varreduras.

Para implementar o algoritmo DP-SLAM é preciso conhecer o modelo matemático do movimento do robô móvel. Mas um dos objetivos deste trabalho é utilizar um único sensor (LRF) para obter o mapa e a localização do robô mó-

vel nele, sem uso de odometria ou outros sensores, muito menos as informações dos comandos enviados ao robô ou seu modelo. Então, para encontrar o deslocamento do robô móvel em cada instante de tempo, em vez de usar sensores como odômetros são utilizados os resultados do algoritmo de Correspondência de Varreduras para implementar um modelo de movimento.

4.3.1

Modelo de Movimento

Desenvolvendo um modelo matemático para o deslocamento do robô móvel, para uma posição (R_{t-1}) no instante $t - 1$, dado um deslocamento

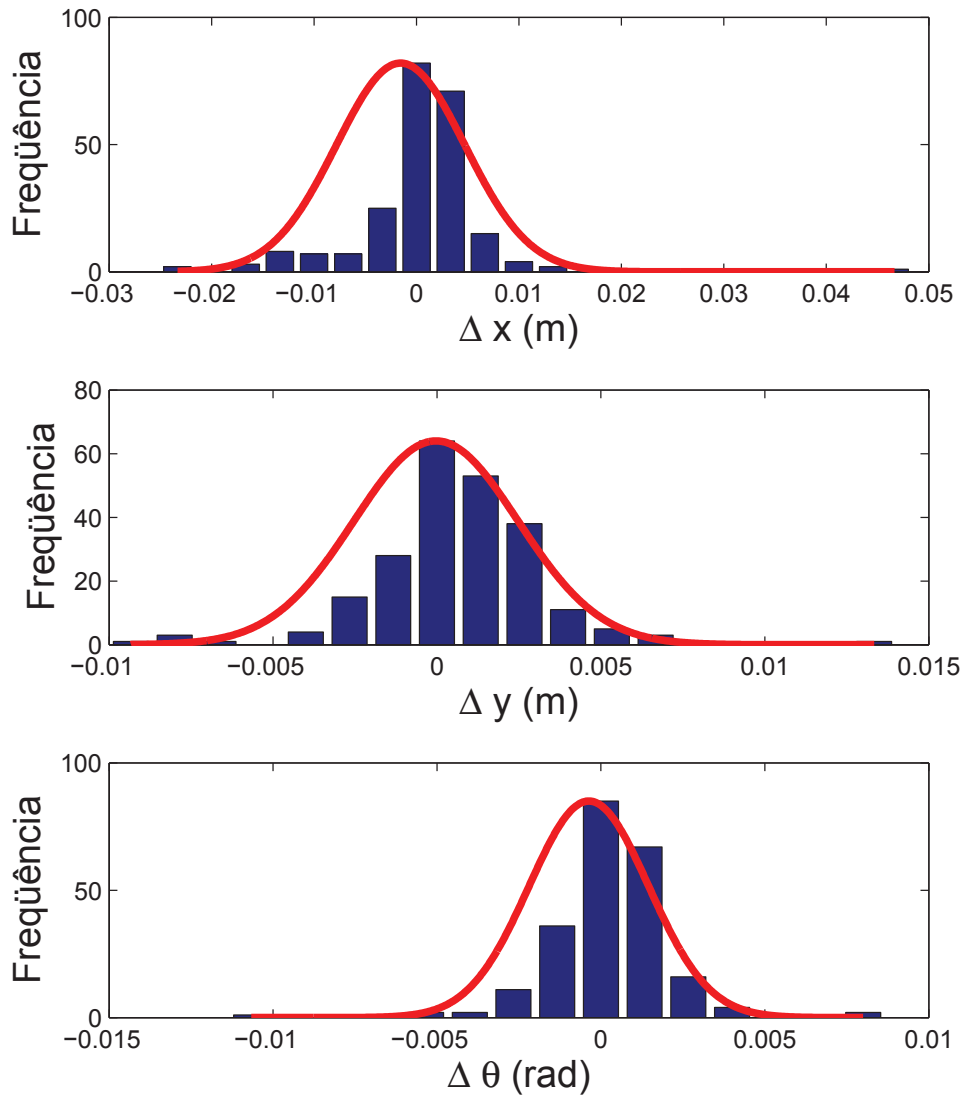


Figura 4.12: Distribuição Normal através do histograma do erro ($\Delta x, \Delta y, \Delta \theta$)

$(\Delta x, \Delta y, \Delta \theta)$ entre o instante $t - 1$ e t , a posição (R_t) , no instante t , é dada por:

$$R_t = R_{t-1} + \begin{pmatrix} d \cos(\alpha) \\ d \sin(\alpha) \\ \Delta \theta \end{pmatrix}$$

onde: $\alpha = R_{\theta_{t-1}} + \tan^{-1}(\frac{\Delta x}{\Delta y})$ e $d = \sqrt{(\Delta x)^2 + (\Delta y)^2}$.

Os deslocamentos $(\Delta x, \Delta y, \Delta \theta)$ apresentam um erro, o qual deve ser introduzido ao modelo de movimento, assim com a análise de dados simulados feita, e calculando o histograma do erro; a partir deste, pode-se criar uma distribuição normal para o erro de cada variável do deslocamento Δx , Δy e $\Delta \theta$. A Figura 4.12 mostra um exemplo a distribuição normal aproximada do erro.

Para os dados da Figura 4.12, a distribuição do erro em Δx pode ser aproximada por uma distribuição gaussiana com média $-0,0016m$ e um desvio padrão de $0,02m$, da mesma forma o erro em Δy pode ser aproximado por uma distribuição gaussiana com média zero e um desvio padrão de $0,01m$; e em $\Delta \theta$ por uma distribuição gaussiana com média zero e um desvio padrão de $0,008rad$.

Tabela 4.2: Modelo do Movimento de Correspondência de Varreduras [2].

<i>Algoritmo_Sample_Motion</i> (R_{t-1}, u_t)	
1:	$\Delta \hat{x} = \Delta x + \text{Sortear}(0.02 + \text{abs}(\frac{\Delta x}{10}))$
2:	$\Delta \hat{y} = \Delta y + \text{Sortear}(0.01 + \text{abs}(\frac{\Delta y}{10}))$
3:	$\Delta \hat{\theta} = \Delta \theta + \text{Sortear}(0.008 + \text{abs}(\frac{\Delta \theta}{10}))$
4:	$d = \sqrt{(\Delta \hat{x})^2 + (\Delta \hat{y})^2}$
5:	$\alpha = R_{\theta_{t-1}} + \text{atan2}(\Delta \hat{y}, \Delta \hat{x})$
6:	$R_{x_t} = R_{x_{t-1}} + d \cos(\alpha)$
7:	$R_{y_t} = R_{y_{t-1}} + d \sin(\alpha)$
8:	$R_{\theta_t} = R_{\theta_{t-1}} + \Delta \hat{\theta}$
9:	Retornar $R_t = (R_{x_t}, R_{y_t}, R_{\theta_t})$

As médias e os desvios padrão destes três movimentos são usados no modelo de movimento. Embora o modelo de movimento tenha sido em informações de dados simulados, este modelo será estendido para trabalhar com

Tabela 4.3: Algoritmo Aproximado de uma Distribuição Normal

Sortear: <i>Normal_Distribution</i> (<i>v</i>)	
1:	Retornar $\frac{1}{6} \sum_{i=1}^{12} rand(-v, v)$

dados reais.

O modelo de movimento será capaz de amostrar o deslocamento do robô móvel. O modelo de movimento é mostrado na Tabela 4.2, onde a função *Sortear* retorna um valor aleatório aproximadamente normal, segundo a função descrita na Tabela 4.1, e *atan2*() é a função arco-tangente tal que $a = atan2(b, c) \Rightarrow sin(a) = b$ e $cos(a) = c$.

4.4
Mapeamento e Localização usando DP-SLAM

O algoritmo DP-SLAM fornece um mapeamento de maior qualidade para um ambiente. Assim com a adição do modelo de movimento, descrita acima e as leituras do sensor LRF, nesta seção apresentamos 2D mapas, para o ambiente simulado.

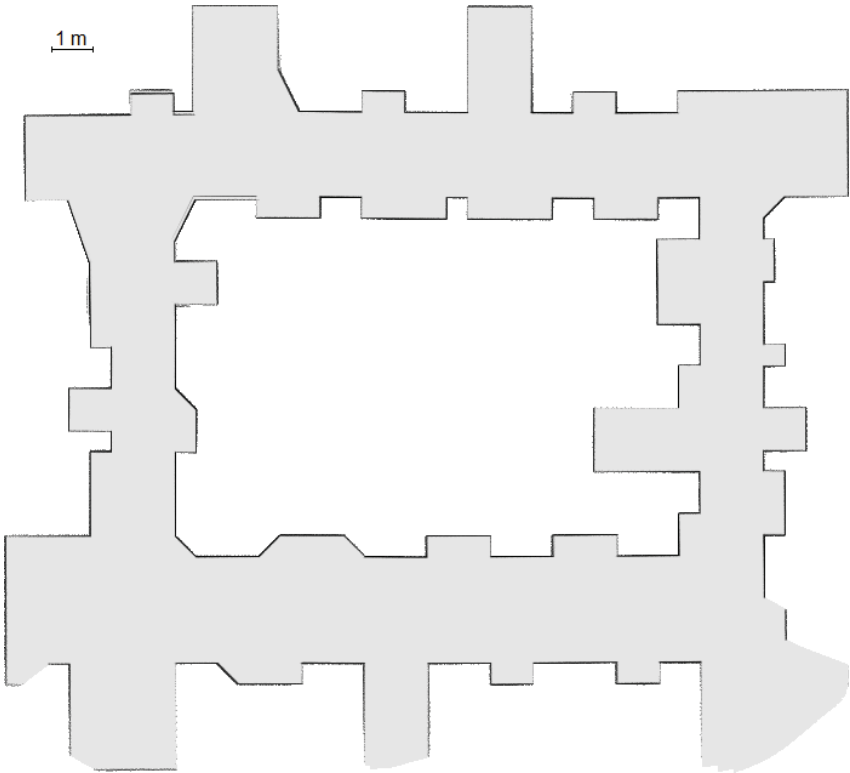


Figura 4.13: Mapa 2D do ambiente simulado obtido usando o algoritmo DP-SLAM

A Figura 4.13 mostra o mapa 2D do ambiente simulado. A grade de ocupação utilizada para representar o mapa tem uma resolução equivalente a $1m = 35grades$, onde os espaços pretos indicam que está ocupado e os espaços cinza indicam que está livre, e os espaços em branco são lugares que se encontram fora do alcance do sensor durante a experiência.

Como pode ser visto na Figura 4.13, o mapa obtido por DP-SLAM mostra um mapa de maior qualidade, em comparação com aquele criado pelo algoritmo de Correspondência de Varreduras. O mapa apresenta um pequeno desalinhamento, muito menor do que aquele mapa da Figura 4.10.

Os parâmetros do algoritmo DP-SLAM usados para todos os experimentos realizados estão na Tabela 4.4, estes parâmetros são recomendadas em [4].

Tabela 4.4: Parâmetros DP-SLAM dos experimentos.

Parâmetro	Valor
Resolução	$1m = 35grids$
Número de Partículas LSLAM	500
Número de Partículas HSLAM	500

No próximo capítulo, os experimentos realizados e seus resultados são apresentados.