3
Localização e Mapeamento Simultâneos - Híbrido

O modelo detalhado neste capítulo foi projetado para um robô equipado com sonares e bússola a mapear e localizar-se simultaneamente em ambientes que contenham apenas planos. Ambientes como estes são comuns em locais projetados por seres humanos e que contêm uma grande quantidade de superfícies planas, como paredes.

3.1
Algoritmo

O fluxograma da Figura 21 descreve as etapas do algoritmo proposto. Assume-se que a posição inicial do robô é conhecida e que o mesmo encontra-se em um ambiente composto apenas por planos. Este trabalho aborda apenas o problema do *SLAM*, por este motivo supõe-se que o robô conheça a trajetória a ser seguida e que o sistema de controle seja capaz de mantê-lo na mesma. A seguir, será feita uma breve descrição do algoritmo. Nas secções seguintes, partes específicas de cada passo serão detalhadas.

No primeiro passo, faz-se uma varredura do ambiente com os sonares e atualiza-se o mapa de ocupação em grade, este processo é realizado em cada interação.

Após atualizar o mapa de ocupação em grade, calcula-se, para cada sonar, a possibilidade de o mesmo estar detectando uma reta. Inicialmente não haverá retas no mapa, pois o ambiente ainda não foi explorado o suficientemente. Dessa forma, a leitura do sonar vai resultar apenas em pontos e todos os pontos gerados desta maneira são armazenados em uma espécie de repositório.

Utilizando o mapa de ocupação em grade, selecionam-se apenas os pontos que estão em regiões ocupadas. A este conjunto de pontos é dado o nome de Pontos Válidos.

Utilizando o conjunto de pontos válidos, usa-se o DBSCAN para buscar por agrupamentos de pontos que juntos formem uma reta. Desta forma, caso um conjunto de pontos seja agrupado com sucesso, estes são removidos do repositório e passam agora a pertencer ao conjunto de pontos que formam aquela reta

específica. A reta detectada dessa forma passa a pertencer ao mapa de retas, servindo como um marco para o robô utilizar em sua localização.

Quando uma reta está visível a um sonar, o robô é capaz de utilizá-la para estimar a sua posição via FKE, e o ponto de contato entrará para o conjunto de pontos que compõem a reta, com este novo conjunto de pontos recalcula-se a reta através de mínimos quadrados. Desta forma, a cada nova detecção, as retas do mapa são gradativamente atualizadas.

Nas subsecções que seguem, as etapas do algoritmo serão detalhadas, assim como as fórmulas usadas para atualização do mapa de ocupação em grade, do repositório de retas e do FKE.

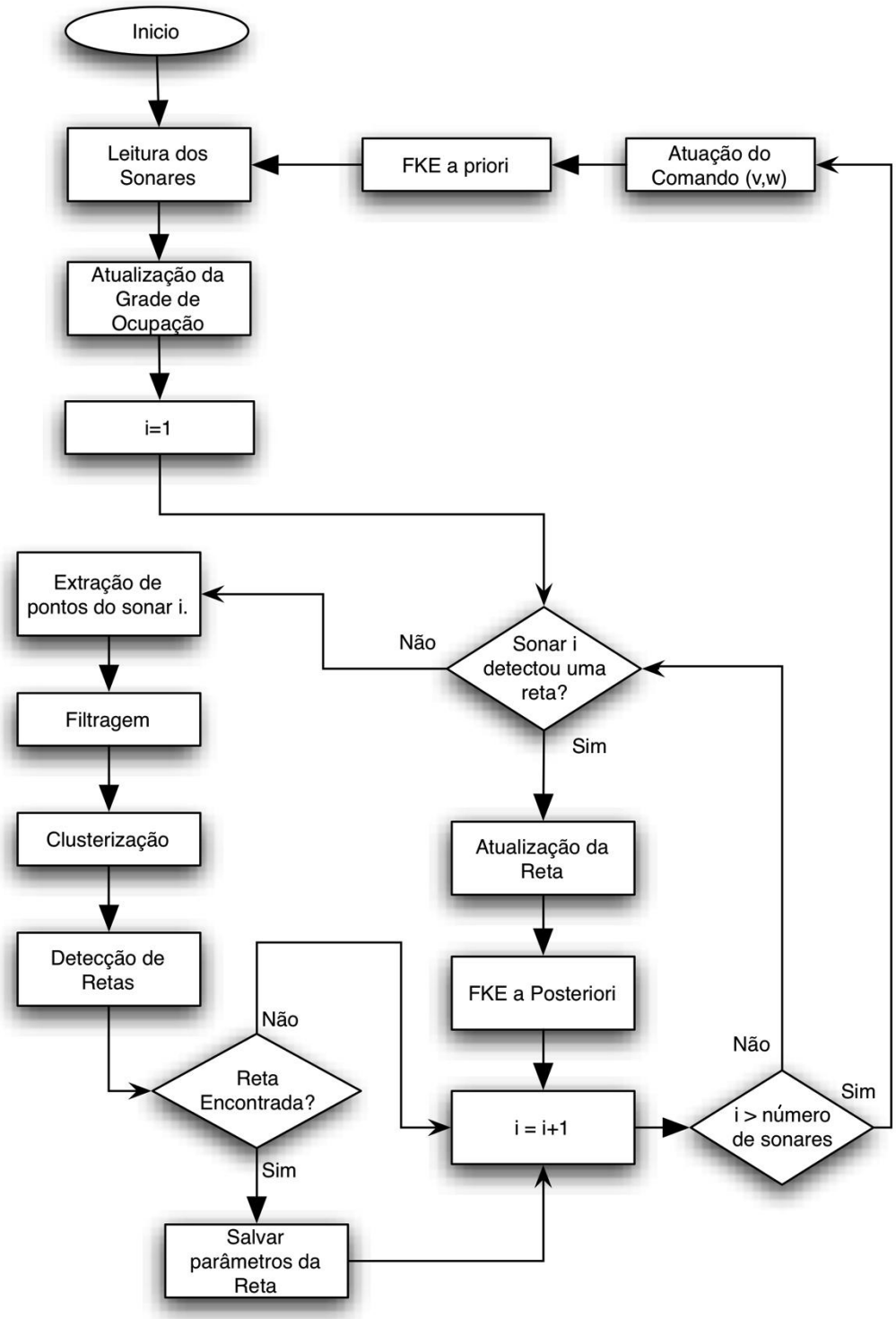

Figura 21 - Algoritmo do Modelo.

3.2
Extração de Pontos

Quando o eco do sonar é refletido por um obstáculo à sua frente, a distância calculada pelo *TOF*, idealmente, é a menor distância do sonar ao obstáculo. Isso porque, na grande maioria dos casos, o sonar está programado para captar o

primeiro eco. Entretanto, apesar de saber a distância até o ponto mais próximo ao obstáculo, o exato ponto de contato do som com o obstáculo não é conhecido, por esta razão, admite-se um ou mais pontos ao longo do semicírculo da extremidade do cone do sonar, representado em verde na Figura 22.

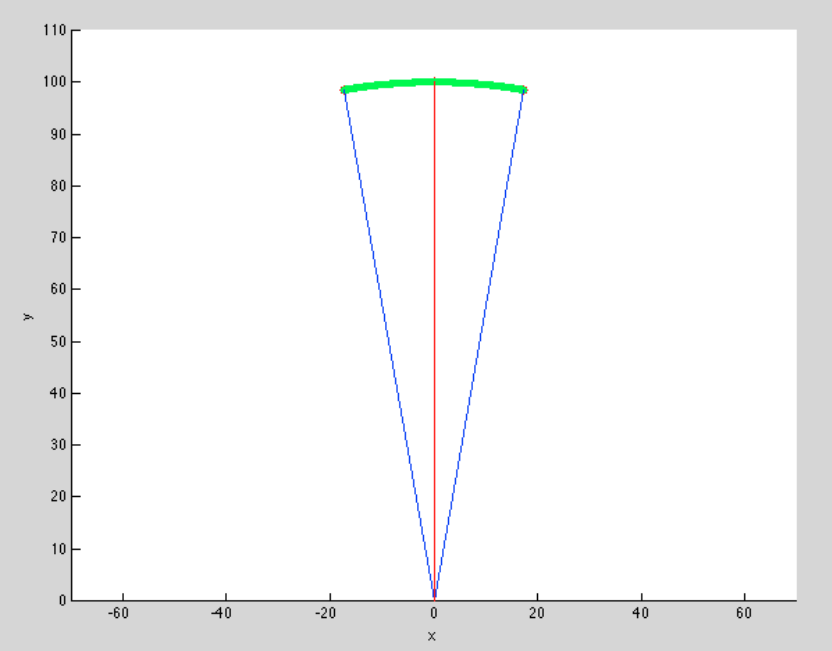

Figura 22 - Características de uma leitura do sonar.

Pode-se, portanto, aquisitar infinitos pontos de cada leitura do sonar, desde que essa leitura seja válida. Considera-se uma leitura válida, aquela em que a distância retornada, d, do sonar, esteja entre d_{min} e d_{max}, onde d_{min} é o menor alcance do sonar e d_{max} o maior alcance do sonar.

$$d_{min} < d < d_{max} \tag{16}$$

Leituras que estão fora deste intervalo provavelmente são resultados de *specular reflection*s ou de *cross-talk* e, por esta razão, não contém informações relevantes, podendo ser naturalmente descartadas.

Supondo um sonar centrado em um sistema de coordenadas cartesianas, como mostrado na Figura 22, com o seu eixo principal, representado em vermelho, alinhado com o eixo y do sistema de coordenadas, os pontos extraídos localizam-se ao longo da extensão do semicírculo do cone, representado em verde. Logo, as coordenadas x e y do ponto ${}^{S}P = ({}^{S}x, {}^{S}y)$ no sistema de coordenadas do sonar são dados pelas Equações (17) e (18) respectivamente.

$${}^{S}x_P = d.\cos(\psi) \tag{17}$$

$$^{S}y_P = d.\sin(\psi) \tag{18}$$

Onde ψ é o ângulo que a reta que vai do centro do sistema de coordenadas até o ponto P faz com o eixo x. Sabendo que β é a abertura do cone do sonar, a faixa de valores possíveis para ψ é dada pela Equação (19):

$$\frac{\pi}{2} - \frac{\beta}{2} \leq \psi \leq \frac{\pi}{2} + \frac{\beta}{2} \tag{19}$$

Supondo que em uma leitura qualquer, o sonar retornou uma distância $d = 100$ cm, e que a abertura do cone é igual a $\beta = 20^o$. Variando o valor de ψ, como mostrado na Equação (19), é possível obter infinitos pontos. Para obter os pontos $^{S}P_1$, $^{S}P_2$ e $^{S}P_3$, mostrados na Figura 23, utilizaremos como valores de ψ 100^o, 90^o e 80^o respectivamente (devidamente convertidos em radianos), as coordenadas desse ponto, no sistema de coordenadas do sonar, são: $^{S}P_1 = (-17{,}36; 98{,}48)$, $^{S}P_2 = (0; 100)$ e $^{S}P_3 = (17{,}36; 98{,}48)$.

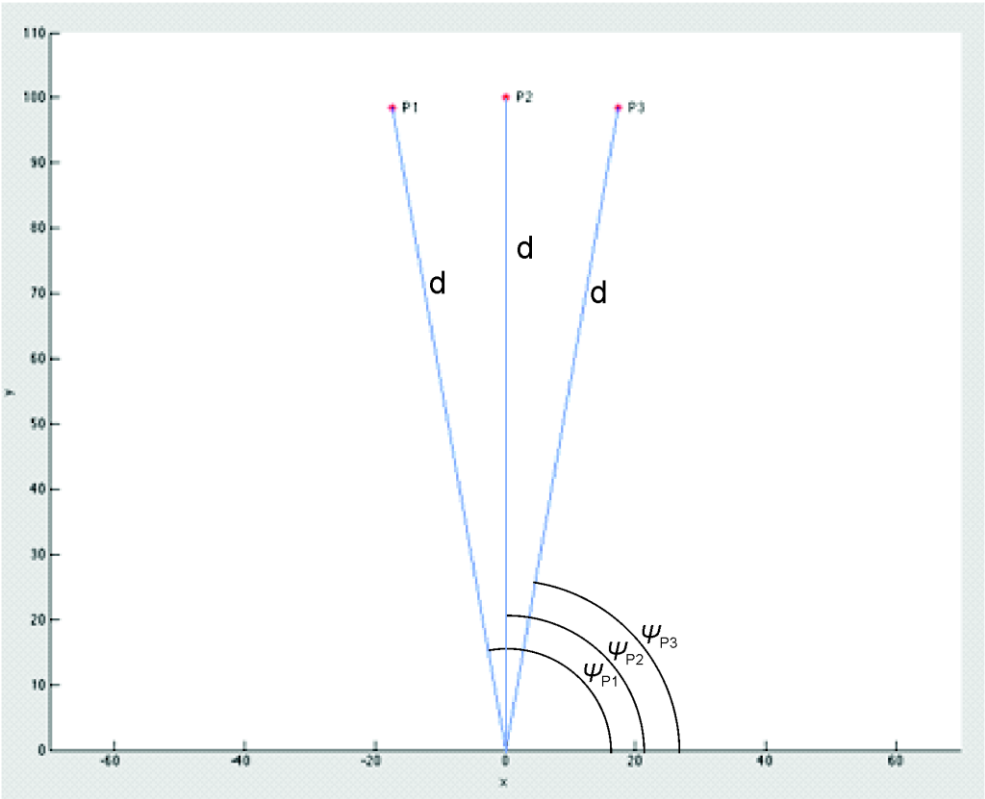

Figura 23 - Estimação de três pontos com uma única leitura do sonar.

Os pontos extraídos desta forma estão no sistema de coordenadas do sonar, entretanto, a referência do sonar muda à medida que o robô se desloca, por este motivo, os pontos devem ser representados em um sistema de coordenadas inerte.

Costuma-se chamar este sistema de coordenada inerte como o sistema de coordenadas do mundo. Sendo assim é necessário mudar o sistema de coordenadas com base no sonar para o sistema de coordenadas do mundo. Só após esse procedimento os pontos são transferidos para o repositório de pontos.

3.3
Criação da Grade de Ocupação

Optou-se por utilizar duas formas de representação do ambiente: grade de ocupação e contínua. Por esta razão, deu-se o nome do algoritmo de Localização e Mapeamento Híbrido – *LMS-H.* A grade de ocupação irá conter informações a respeito da ocupação do ambiente enquanto a representação contínua nos fornecerá marcos, que permitirá utilizar o Filtro de Kalman Estendido para localizar o robô, em nosso modelo utilizou-se retas como marcos. Além disso, o mapa de ocupação em grade proverá informações relevantes que serão utilizadas para filtrar os pontos lidos pelo sonar.

O sonar, idealmente, fornece boa informação quanto a ocupação do meio, pois todo o volume interior ao cone, em teoria, está vazio e a extremidade possui pelo menos uma célula ocupada. Mas, como as leituras são altamente ruidosas, é necessário tratar os dados de forma mais eficiente, um dos trabalhos pioneiros foi o de (Moravec & Elfes, 1985). Mais tarde sucessivos trabalhos refinaram a representação inserindo técnicas probabilísticas e modelos mais complexos do sonar: (Matthies & Elfes, 1988), (Alberto Elfes, 1992), (A Elfes, 1991), (Konolige, 1997), (A Elfes, 1989).

Como o modelo proposto não se baseia inteiramente no mapa de ocupação em grade será adotado um modelo mais simples, mas que satisfaça duas premissas:

- Mudanças na ocupação da célula devem ser detectadas quando as mesmas forem revisitadas;
- Quanto mais o ambiente for observado, mais definido deverá ficar o mapa de ocupação em grade..

Abaixo seguem detalhes da implementação do método de ocupação em grade utilizado, que está baseado no trabalho de (Moravec & Elfes, 1985), sendo que a representação final do mapa foi modificada para facilitar a identificação das regiões ocupadas e desocupadas.

Para compor o mapa, duas matrizes MxN são utilizadas, a primeira representa a região ocupada ($Occ(X,Y)$) e a segunda representa a região vazia ($Emp(X,Y)$). O cone do sonar é modelado por uma função que representa a confiança de uma célula estar ocupada ou vazia. Esta confiança é dada por um peso calculado para cada célula dentro do cone do sonar.

Supondo o sonar posicionado no centro de um sistema de coordenadas polares e orientado em 0^o, como mostrado na Figura 24, a região livre vai de $r = [d_{min}\ ,\ d - \varepsilon]$ e $\psi = [\frac{\pi}{2} - \frac{\beta}{2}\ ,\ \frac{\pi}{2} + \frac{\beta}{2}]$, onde d é a medida de distância retornada pelo sonar, d_{min} o alcance mínimo do sonar e β a abertura do cone do sonar. Já a região ocupada vai de $r = [d - \varepsilon, d + \varepsilon]$ e $\psi = [\frac{\pi}{2} - \frac{\beta}{2}, \frac{\pi}{2} + \frac{\beta}{2}]$. A região em azul é, portanto, a região desocupada e a região em vermelho a região ocupada.

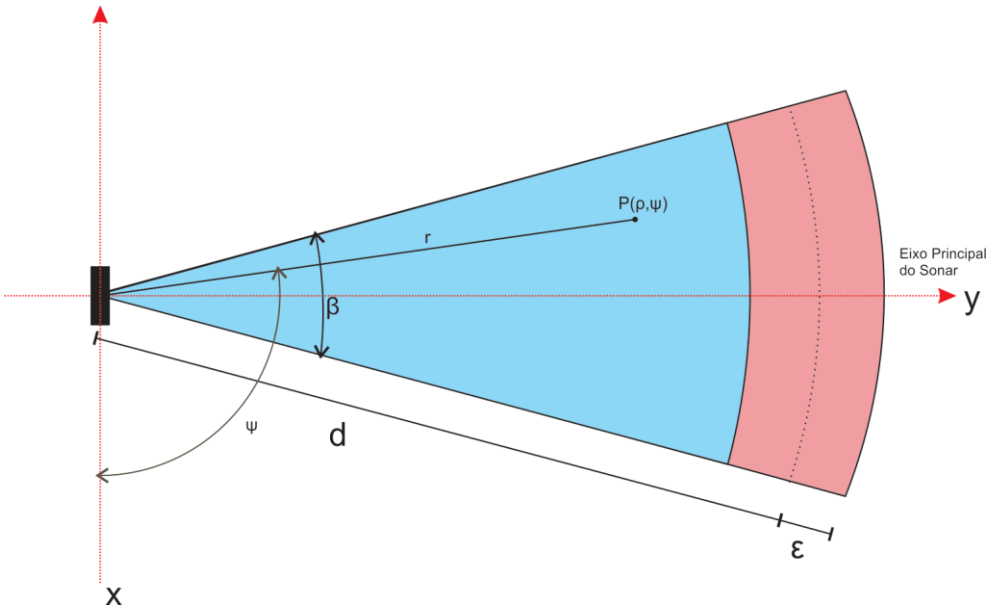

Figura 24 – Representação 2D da leitura do sonar.

Para regiões vazias, quanto maior a proximidade das células do sonar (até o limiar da distância mínima de detecção) ao eixo principal do cone, maior será o peso atribuído. Para regiões ocupadas, o peso será maior próximo ao eixo principal e à distância de detecção retornada pelo sonar. Assim, as funções que ponderam as áreas vazias (P_E) e ocupadas (P_O) são dadas respectivamente pelas Equações (20) e (21).

$$P_E(r,\psi) = E_r(r)E_a(\psi) \tag{20}$$

$$P_O(r,\psi) = O_r(r)O_a(\psi) \tag{21}$$

Onde:

$$E_r(r) = 1 - \left(\frac{r - d_{min}}{r - \varepsilon - d_{min}}\right)^2 \tag{22}$$

$$E_r(\psi) = 1 - \left(2.\frac{|\frac{\pi}{2} - \psi|}{\beta}\right)^2 \tag{23}$$

$$O_r(r) = 1 - \left(\frac{r - d}{\epsilon}\right)^2 \tag{24}$$

$$O_a(\psi) = 1 - \left(2.\frac{|\frac{\pi}{2} - \psi|}{\beta}\right)^2 \tag{25}$$

$E_r(r)$ e $E_a(\psi)$ assumem valores iguais a zero fora da região vazia, assim como $O_r(r)$ e $O_a(\psi)$ assumem valor zero fora da região ocupada.

Observa-se que o local de maior peso de P_E e P_O está sobre o eixo principal do cone do sonar a uma distância d_{min} e d respectivamente. Isso porque objetos sobre o eixo principal do cone do sonar são mais facilmente detectáveis.

O perfil das funções de peso P_O e P_E pode ser visto na Figura 25 e Figura 26 respectivamente, com o sonar posicionado em $P = (0{,}100)$ e orientado em 90^o.

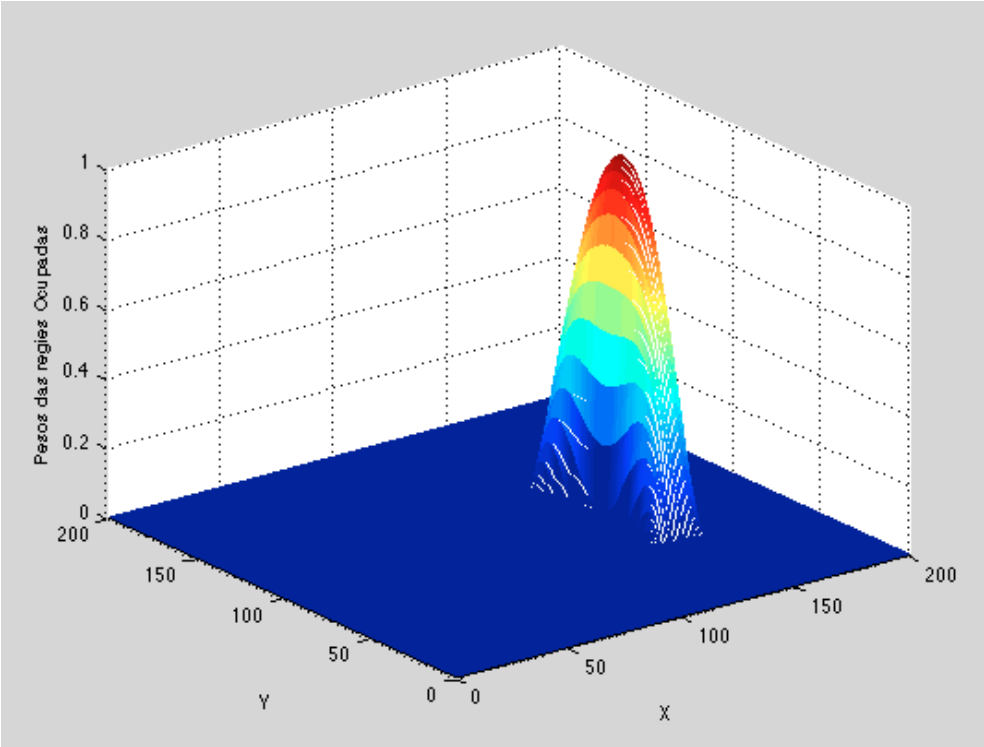

Figura 25 - Função peso para regiões ocupadas, P_O.

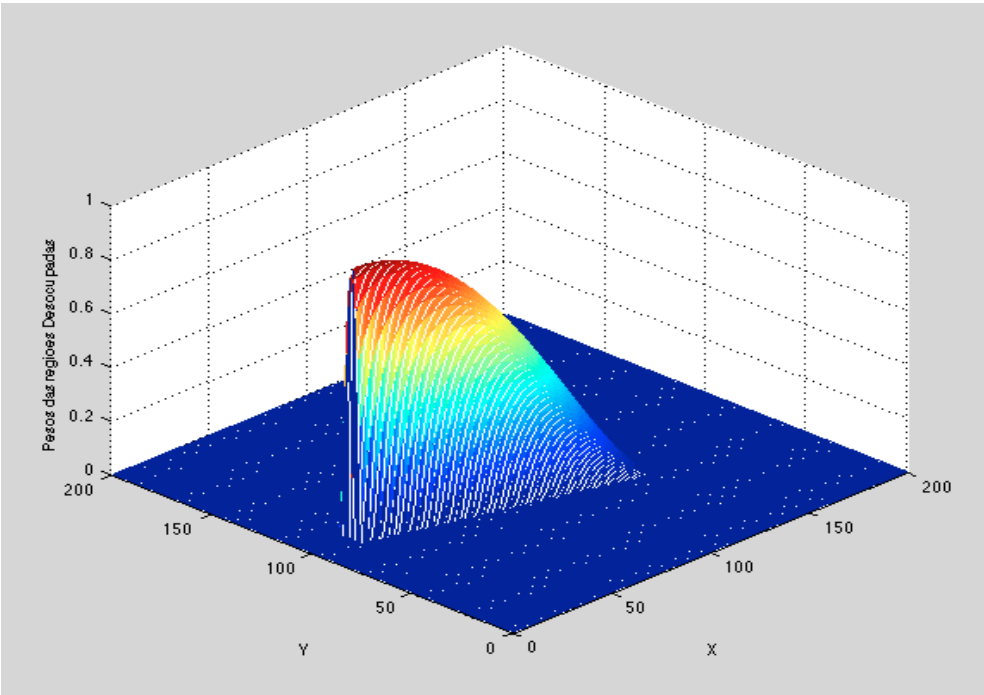

Figura 26 - Função peso para regiões vazias, P_E.

Costumeiramente, utilizam-se diversos sonares montados sobre o robô, além disso, um mesmo sonar faz leituras em posições diferentes todas as vezes que o

robô se desloca. Para compor os mapas Occ e Emp é feita, então, uma simples soma da leitura anterior com as informações já coletadas, como segue:

$$Occ = P_O + Occ \tag{26}$$

$$Emp = P_E + Emp \tag{27}$$

As matrizes Occ e Emp acumularão as informações captadas pelos diferentes sonares em diferentes posições. Para reunir esta informação em apenas uma matriz, faz-se:

$$Map = \log_{10}\left(\frac{1 + Occ}{1 + Emp}\right) \tag{28}$$

Dessa forma, células em Map negativas possuem maior chance de serem regiões vazias, células positivas de serem regiões ocupadas e células não visitadas possuem valor zero, como mostra a Figura 27. De forma heurística, um limiar de decisão dirá se a célula está ocupada ou vazia.

$$\begin{cases} Map_{ij} \geq limiar_{OCC} & = Região\ Ocuoada \\ Map_{ij} \leq limiar_{EMP} & = Região\ Vazia \end{cases} \tag{29}$$

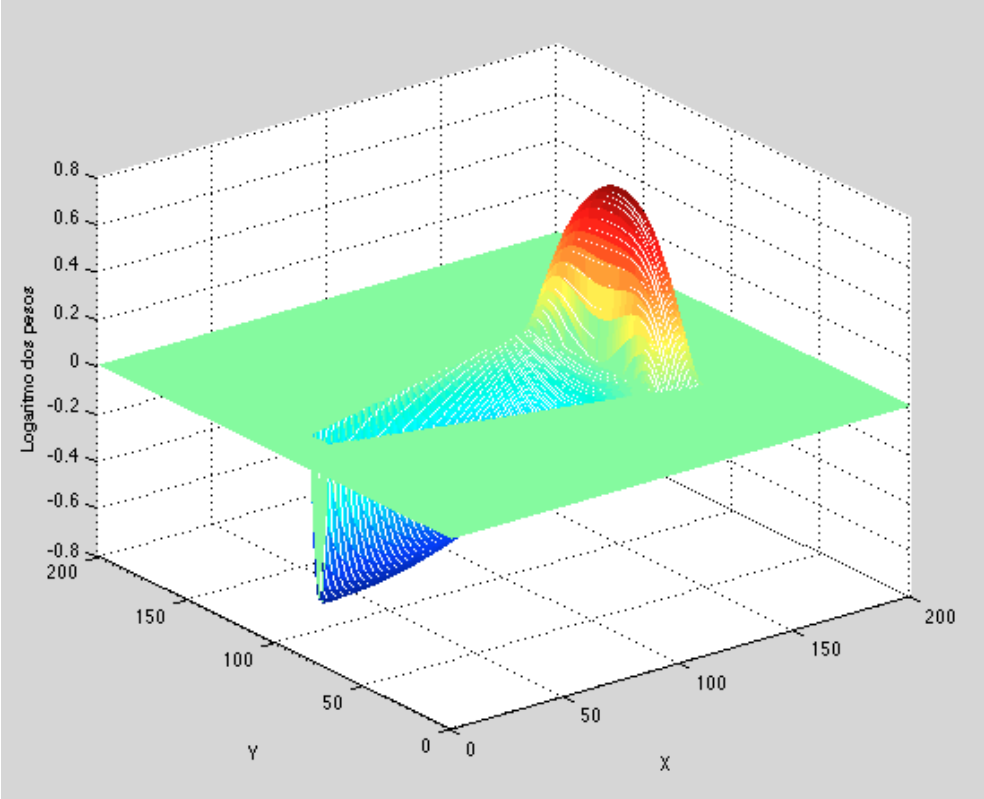

Figura 27 - Mapa logarítmico que engloba as regiões ocupadas e vazias.

3.4
Clusterização dos Pontos e Criação das Retas

Os sonares continuamente fazem leituras do ambiente à medida que o robô se move, e como foi visto na seção 3.2, em cada leitura válida do sonar, pelo menos um ponto pode ser extraído e, como mostrado na seção 3.3, um mapa de ocupação do ambiente é criado. Nesta seção, será visto como unir estas duas informações e criar retas que servirão de marcos para o robô. Diversos trabalhos usam retas como primitivas geométricas para localização e/ou mapeamento, principalmente quando se trata de ambientes feitos por humanos (James L Crowley, 1985), (J L Crowley, 1989), (Meng et al., 2000), (Grosmann, 2001), (Ip & Rad, 2004). Alguns utilizam ainda círculos para representar regiões salientes como cantos e cilindros (Lee & Song, 2010).

Busca-se, portanto, uma forma eficiente de selecionar e agrupar os pontos gerados pelo sonar, a fim de identificar uma reta que se assemelhe ao máximo às superfícies planas do ambiente. Contudo, não é possível determinar o exato ponto de origem do eco, como já foi discutido anteriormente.

Os pontos gerados com as leituras do sonar raramente localizam-se no local exato da reflexão do som. Alguns, porém, podem estar próximos o suficiente para que sejam usados. Para encontrar os pontos que estão, de certa fora, próximos o suficiente do plano que se deseja detectar, usa-se o mapa em grade de ocupação, mostrado na Secção 3.3. Através do mapa de ocupação, filtram-se os pontos localizados em regiões dadas como vazias e utilizam-se apenas os que estão em regiões ocupadas.

Após a filtragem, entretanto, ainda é necessário distingui-los, ou seja, saber quais pontos pertencem a qual superfície. Por exemplo, suponha o robô se deslocando ao longo de um corredor, como mostrado na Figura 28. Alguns sonares detectarão uma parede à sua esquerda e outros detectarão uma parede à sua direita. Depois de filtrados, ainda será necessário saber quais pontos pertencem a parede à esquerda do robô e quais pontos pertencem a parede à direita do robô. Para resolver este problema utilizou-se o algoritmo DBSCAN (*Density Based Spatial Clustering of Application with Noise*) (Sander, Ester, Kriegel, & Xu, 1998), (Ester, Xu, Kriegel, & Sander, 1996).

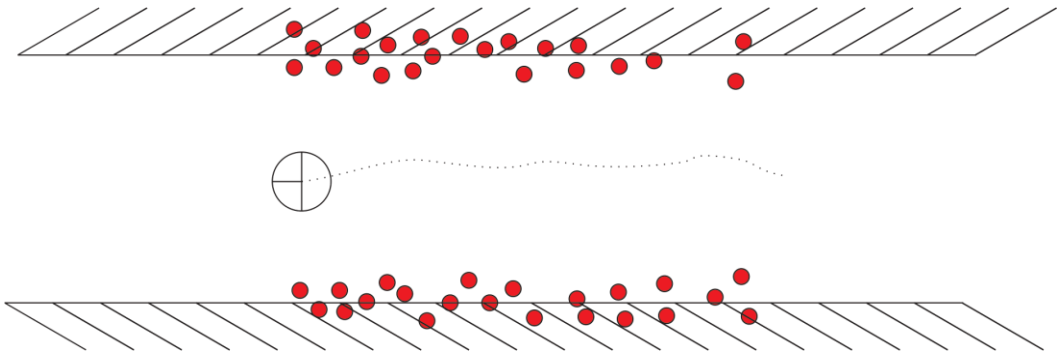

Figura 28 - Robô movimentado-se ao longo do corredor.

O DBSCAN procura por conjuntos densos seguidos de áreas de baixa densidade, o que se encaixa perfeitamente ao problema. O DBSCAN possui três parâmetros de entrada:

- Raio da Vizinhança - ϕ;
- Mínimo Número de Pontos – min_p;
- Conjunto de Dados (pontos).

A saída é um conjunto k de clusters. Essa é uma das desejáveis características do DBSCAN, pois não há a necessidade de informar com antecedência a quantidade de conjuntos desejados, os conjuntos serão

naturalmente separados de acordo com a concentração e distribuição dos pontos no espaço.

Para iniciar o algoritmo, um ponto q_1 qualquer é eleito (o que o torna ainda insensível a ordem dos elementos) e então agrupa-se todos os elementos que estão a pelo menos uma distância ϕ dele. Se a quantidade de pontos dentro deste subconjunto for maior ou igual a min_p então um novo agrupamento é criado e o ponto q_1 é rotulado como um ponto central. Repete-se então este procedimento para todos os pontos.

No final, teremos três tipos de pontos, como pode ser observado no exemplo da na Figura 29, onde o $\phi = 20\ cm$ e $min\_p\ = 4$:

- **Pontos Centrais**: São pontos que possuem pelo menos $min_p - 1$ pontos a um raio ϕ dele (pontos em preto);
- **Pontos Fronteiras**: São pontos que estão dentro de um agrupamento, mas não são pontos centrais (ponto em amarelo);
- **Pontos *Outliers***: Pontos que não estão em nenhum agrupamento (pontos em laranja).

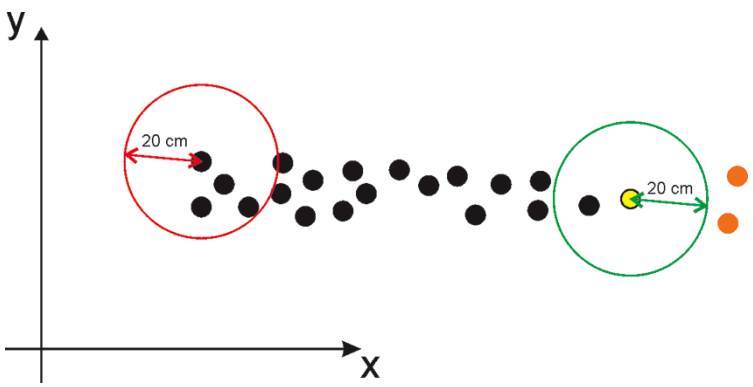

Figura 29 - Exemplo de clusterização pelo DBSCAN, $min_p = 4$.

Agrupamentos que compartilham pelo menos um ponto (Ponto Central ou Ponto de Fronteira) são unidos em um único agrupamento e os pontos Outliers são descartados.

Voltando ao exemplo do corredor, ao final do DBSCAN os pontos serão divididos, formando dois clusters distintos, um contendo os pontos da parede à esquerda do robô e outro contendo os pontos da parede à direita do robô, Para se determinar retas a partir desses pontos, foi usado o método dos mínimos quadrados para os pontos contidos em cada um dos clusters identificados pelo DBSCAN, tem-se, então, as duas paredes representadas por duas retas distintas.

Supondo agora o robô fazendo uma curva ao longo de um canto, como mostrado na Figura 30. Os pontos coletados ficarão ao longo das duas paredes que se cruzam e certamente o DBSCAN irá agrupá-los em um único grupo. Claramente, ajustar uma reta com mínimos quadrados não corresponderá a realidade, por esta razão, a última faze do DBSCAN foi alterada de forma que os subgrupos apenas serão unidos se o erro do ajuste de uma reta nos pontos pertencentes aos dois subgrupos for menor que um valor ϵ. Dessa forma, os pontos coletados na quina formada pela junção das duas paredes, como foi exemplificado, serão naturalmente subdivididos em duas retas distintas.

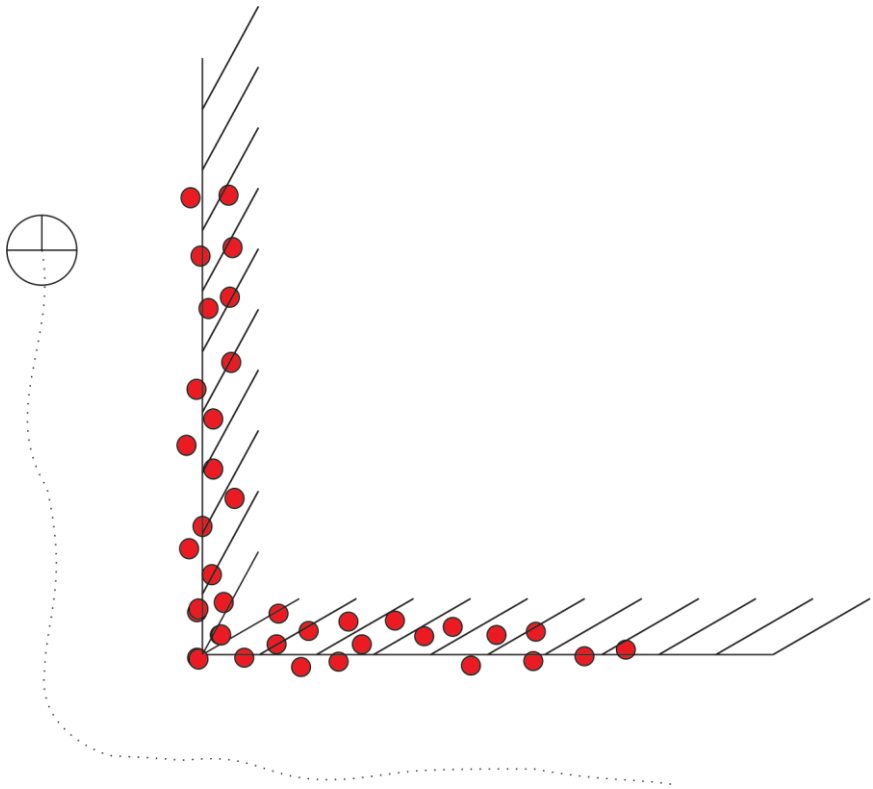

Figura 30 - Robô locomovendo-se ao longo de uma quina formada por duas paredes.

Assim que uma reta é detectada com sucesso, esta passará a fazer parte do mapa de retas. Neste momento a representação contínua do ambiente é criada. Diferente da grade de ocupação, a representação contínua não informa nada a respeito da ocupação do ambiente, ao invés disso, ela contém primitivas que descrevem o ambiente, neste caso: retas. Para cada reta extraída dessa forma, além dos pontos, outros sete parâmetros também são mantidos:

"a", "b" e "c": Parâmetros da reta "$ax + by + c = 0$";

"ρ" e "α": Parâmetros da reta no espaço de Hough;

"pc" - Ponto de contato: Ponto de contato entre a reta e uma outra reta perpendicular a ela e que passa pelo centro do sistema de coordenadas;

"ϵ": Erro médio quadrático.

Os parâmetros "a", "b" e "c" são calculados via mínimos quadrados como mostrados nas Equações (30) a (32).

$$a = \sum x_i \sum y_i - \sum y_i \sum x_i y_i \tag{30}$$

$$b = \sum y_i \sum x_i^2 - \sum x_i \sum x_i y_i \tag{31}$$

$$c = \left(\sum x_i y_i\right)^2 - \sum x_i^2 \sum y_i^2 \tag{32}$$

Os parâmetros "ρ" e "α" são calculados através dos parâmetros "a", "b" e "c" segundo as Equações (33) e (34).

$$\rho = \frac{|c|}{\sqrt{a^2 + b^2}} \tag{33}$$

$$\alpha = \operatorname{atan} 2\left(-\frac{b}{a}\right) - \frac{\pi}{2} \tag{34}$$

O ponto de contato é dado por:

$$pc_x = \rho . \cos(\alpha) \tag{35}$$

$$pc_y = \rho . \sin(\alpha) \tag{36}$$

A Figura 31 mostra um exemplo de uma reta localizada no espaço de Hough, juntamente com o ponto de contato.

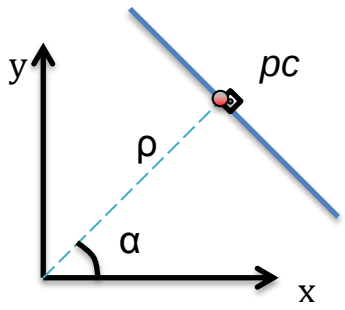

Figura 31 – Reta localizada no espaço de Hough.

3.5
Detecção e Atualização das Retas

Uma reta, depois de ser criada, servirá de marco para o robô se localizar. Entretanto, é desejável que quanto mais o robô observe uma reta, melhor seja sua estimação. Assim, esta secção mostrará como as retas já existentes são detectadas pelos sonares e como essa leitura é utilizada para melhorar a estimação da mesma.

Para uma reta ser detectada ela precisa estar no campo de visão do sonar. De forma resumida, a reta só será detectada se ela estiver tangente ao semicírculo da extremidade do cone do sonar, como mostrado na Figura 32.

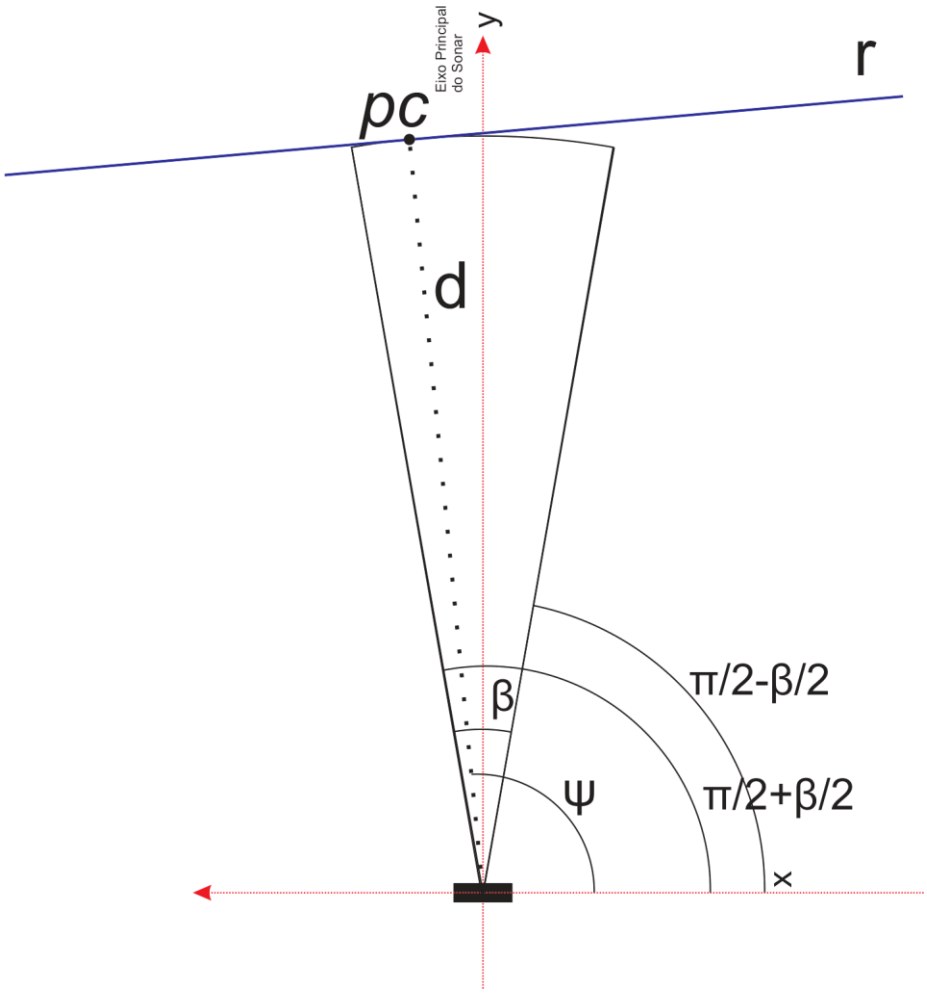

Figura 32 - Exemplo de reta que está no campo de visão do sonar.

Para formalizar o conceito acima, dada uma reta $r_1 = (\rho_1, \alpha_1)$ no espaço de Hough, identificada segundo o sistema de coordenadas do sonar, ela será detectada pelo sonar se as Inequações (37) e (38) forem satisfeitas.

$$\frac{\pi}{2}+\frac{\beta}{2}+\xi_\alpha \leq \alpha_1 \leq \frac{pi}{2}-\frac{\beta}{2}-\xi_\alpha \tag{37}$$

$$d-\xi_\rho \leq \rho_1 \leq d+\xi_\rho \tag{38}$$

Onde ξ_ρ e ξ_α são margens de tolerância devido ao erro inerente do sonar.

A Figura 33 mostra uma reta que não pode ser observada pelo sonar dada a posição em que o sonar encontra-se em relação a ela.

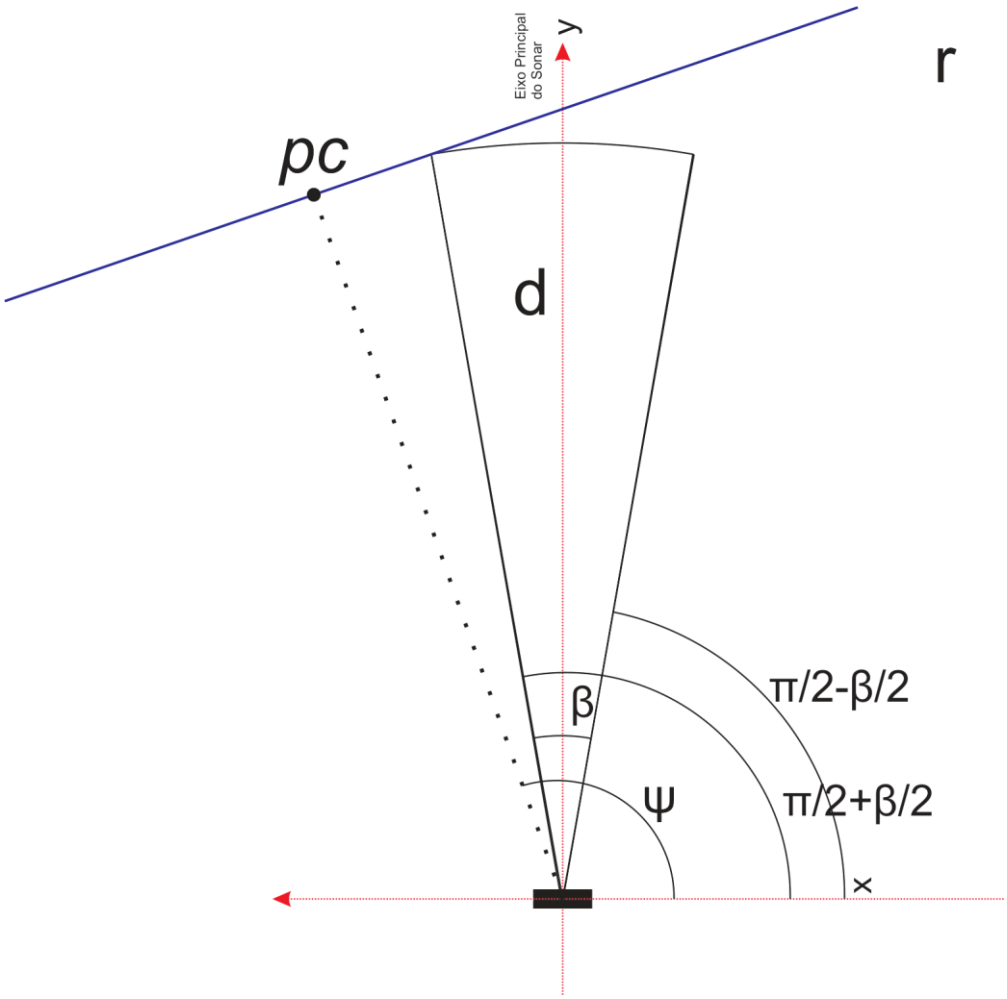

Figura 33 - Exemplo de reta que não está no campo de visão do sonar.

Sempre que uma reta é identificada desta maneira, o ponto de contato (pc) é adicionado ao conjunto de pontos que forma esta reta e seus parâmetros são recalculados segundo as Equações (30) a (36), esse procedimento aprimora a localização da reta e é desta forma que elas são atualizadas.

3.6
Atualização da Pose do robô por Filtro de Kalman Estendido

O Filtro de Kalman Estendido é amplamente utilizado em problemas de localização e mapeamento de robôs móveis. Neste trabalho, utilizaram-se as retas extraídas do ambiente, como descrito nas secções anteriores, para localizar o robô.

A primeira parte do FKE é a atualização da pose do robô, X_{t+1}, e da sua covariância, Σ_{t+1}, apenas com as informações dos encoders e da bússola. Essa atualização é chamada de a priori, pois não leva em consideração as informações do meio externo, e é dado pelas Equações (39) e (40).

$$\bar{\mu}_{t+1} = \mu_{t-1} + \begin{bmatrix} v.\cos(\theta) \\ v.\sin(\theta) \\ \omega \end{bmatrix}.\Delta t \tag{39}$$

$$\bar{\Sigma}_t = G_t \bar{\Sigma}_{t-1} G_t^T + V_t R_t V_t^T \tag{40}$$

Onde v é a velocidade linear do robô, ω a velocidade angular, Δt o intervalo de tempo entre duas atualizações ou intervalo de integração, R é a covariância de transição de estado, V_t o jacobiano da função de transição de estado em relação as velocidade linear e angular e G_t o jacobiano da função de transição de estado em relação a pose do robô. Os Jacobianos G_t e V_t são dados pelas Equações (41) e (42) respectivamente.

$$V_t = \begin{bmatrix} \cos(\mu_{t-1,\theta}).\Delta t & 0 \\ \sin(\mu_{t-1,\theta}).\Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \tag{41}$$

$$G_t = \begin{bmatrix} 1 & 0 & -v.sin(\mu_{t-1,\theta}).\Delta t \\ 0 & 1 & v.\cos(\mu_{t-1,\theta}).\Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{42}$$

Observe que na Equação (39), é utilizado $\mu = [\mu_X\ \mu_Y\ \mu_\theta]$ para representar a pose do robô. O FKE requer que as variáveis do sistema sejam variáveis aleatórias gaussianas. Como o ponto da gaussiana de maior probabilidade é exatamente a média μ, esta é usualmente considerada como a posição do robô.

Além dos *encoders*, o robô utilizado neste trabalho está equipado também com uma bússola eletrônica. Para fundir a informação da bússola utiliza-se também o filtro de Kalman. Para isso, calcula-se o ganho de Kalman, dado por k_b e mostrado na Equação (43).

$$k_b = \frac{\sigma_e^2}{\sigma_e^2 + \sigma_b^2} \tag{43}$$

Onde σ_e^2 é a variância da pose estimada do robô pelos encoders, ou seja, $\bar{\Sigma}_t(3,3)$ e σ_b^2 a variância de medição da bússola.

Assim, atualizamos a orientação $\bar{\mu}_{\theta,t}$ do robô como mostrado na Equação (44).

$$\bar{\mu}_{\theta,t} = \bar{\mu}_{\theta,t} + k_b.(\theta_b - \bar{\mu}_{\theta,t}) \tag{44}$$

Onde θ_b é a medida retornada pela bússola.

A nova variância da orientação do robô $\bar{\Sigma}_t(3,3)$ é então dada pela Equação (45).

$$\bar{\Sigma}_t(3,3) = \bar{\Sigma}_t(3,3) - k.\bar{\Sigma}_t(3,3) \tag{45}$$

A segunda parte do FKE utiliza a informação do meio para atualizar a sua posição, como já citado anteriormente, os sensores proprioceptivos fornecem uma boa localização do robô, entretanto, o erro cumulativo dos *encoders* prejudicam a estimação quando o robô tem que se deslocar por grandes distâncias. Por esta razão, são necessárias informações inertes do meio externo para servir como referência em sua navegação.

Como já mencionado anteriormente, usa-se apenas retas como marcos para o robô. Dessa forma, sempre que uma reta é detectada é possível utilizar o Filtro de Kalman Estendido para corrigir a sua localização.

Os algoritmos usados para resolver o problema do SLAM via FKE utilizam como variáveis de estado tanto a pose do robô como a localização dos marcos encontrados. Neste trabalho, todavia, a posição dos marcos (retas) não é atualizada via Filtro de Kalman Estendido. Diferentemente, as retas são atualizadas à medida que são detectadas pelo sonar, como mostrado na secção 3.3. Por esta razão, a parte a posteriori do FKE modificará apenas a pose do robô, ou seja, o FKE apenas é utilizado para localização do robô.

Como mostrado na secção 3.5, existem dois critérios para que o sonar seja capaz de detectar uma reta. Para cada leitura do sonar procura-se em todas as retas existentes no mapa e em todos os intervalos válidos de α por uma correspondência válida segundo as Equações (37) e (38). Supondo que uma correspondência válida com a reta i do mapa seja encontrada, aplica-se o FKE a posteriori para calcular a pose a posteriori do robô, bem como a sua covariância,

dadas pelas Equações do FKE, repetidas aqui por conveniência nas Equações (46) a (49) ..

$$S_t^i = H_t^i . \bar{\Sigma}_t . \left[H_t^i\right]^T + Q_t \tag{46}$$

$$K_t^i = \bar{\Sigma}_t . \left[H_t^i\right]^T . [S_t^i\ ]^{-1} \tag{47}$$

$$\bar{\mu}_t = \bar{\mu}_t + K_t^i . (z_t^i - \hat{z}_t^i) \tag{48}$$

$$\bar{\Sigma}_t = \left(I - K_t^i . H_t^i\right) . \bar{\Sigma}_t \tag{49}$$

Onde H_t^i é o Jacobiano da função de observação, Q a covariância de observação, z_t^i é a posição da reta observada no sistema de coordenadas do espaço de Hough e $\hat{z}_t^i$ a posição estimada da reta dada a posição estimada do robô. A função de observação é simplesmente a reta i no sistema de coordenadas do robô, assim, podemos escrevê-la como mostrado na Equação (50).

$$h_t^i = \begin{bmatrix} \rho_R^i \\ \alpha_R^i \end{bmatrix} = \begin{bmatrix} |x . \cos\left(\alpha_W^i\right) + y . \sin\left(\alpha_W^i\right) - \rho_W^i| \\ \alpha_W^i + \theta - \frac{\pi}{2} \end{bmatrix} \tag{50}$$

Onde ρ_R^i e α_R^i são as coordenadas da reta i no sistema de coordenadas do robô.

Devido ao módulo existente na Equação (50), há três possibilidades para o Jacobiano da função de observação. Se $x . \cos\left(\alpha_W^i\right) + y . \sin\left(\alpha_W^i\right) - \rho_W^i > 0$ então o Jacobiano será:

$$H_t^i = \begin{bmatrix} \cos\left(\alpha_W^i\right) & \sin\left(\alpha_W^i\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{51}$$

Se $x . \cos\left(\alpha_W^i\right) + y . \sin\left(\alpha_W^i\right) - \rho_W^i < 0$, então:

$$\rho_t^i = -(x . \cos\left(\alpha_W^i\right) + y . \sin\left(\alpha_W^i\right) - \rho_W^i) \tag{52}$$

E o Jacobiano da função de observação será:

$$H_t^i = \begin{bmatrix} -\cos\left(\alpha_W^i\right) & -\sin\left(\alpha_W^i\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{53}$$

Quando $x . \cos\left(\alpha_W^i\right) + y . \sin\left(\alpha_W^i\right) - \rho_W^i = 0$ não é possível calcular o Jacobiano da função de observação, pois este é um ponto de descontinuidade, e,

por conseguinte, não é possível utilizar a parte a posteriori do Filtro de Kalman Estendido. Felizmente esta é uma situação bastante rara e não interfere no desempenho do algoritmo.

As observações z_t^i e $\hat{z}_t^i$, mostrados na Equação (48), correspondem à reta observada e estimada, respectivamente. A reta estimada é simplesmente o resultado da função de observação, ou seja, $\hat{z}_t^i = (\rho_R^i, \alpha_R^i)$. Já a reta observada é composta pela distância que o sonar retornou e o ângulo da reta estimada: $z_t^i = (d_t, \alpha_R^i)$. Este é o motivo pelo qual é tão importante o uso da bússola neste modelo, pois apenas uma leitura do sonar é incapaz de informar o ângulo que o robô está em relação as reta, a falta dessa informação impossibilita a correção do erro angular do robô.