Desenvolvimento de uma estratégia de contagem e localização de objetos utilizando uma câmera RGB-D e um robô móvel ⋆

Matheus Loureiro *, **Fabiana S. V. Machado** *, **Ricardo Mello** *, **Anselmo Frizera** *

* *Programa de Pós-Graduação de Engenharia Elétrica, Universidade Federal do Espírito Santo, ES, (e-mail: matheus.loureiro@edu.ufes.br, fabiana.machado@edu.ufes.br, ricardo.c.mello@ufes.br, anselmo.frizera-neto@ufes.br).*

Abstract: Robotic devices capable of interacting with the environment are increasingly common, with autonomous robots as an example of this. However, it is necessary for these devices to be able to extract as much information as possible from the environments that they are envolved. To this, computer vision techniques are used to process images of the environment. So, the objective of this work is to propose a strategy that uses computer vision techniques to identify objects to count and estimate their position, from an RGB-D camera coupled to an autonomous mobile robot. This strategy was validated through testing in 3 different scenarios, which explore the robot's autonomous navigation capability and camera image capture. The results of all scenarios presented a relevant object count and a average error less than 0.19 meters in the pose estimate.

Resumo: Dispostivos robóticos capazes de interagir com o ambiente são cada vez mais comuns, sendo robôs autônomos, um exemplo disso. Contudo, é cada vez mais necessário que esses dispositivos sejam capazes de extrair o maior número possível de informações dos ambientes que estão. Para conseguir isso, utiliza-se técnicas de visão computacional para processamento de imagens do ambiente. Então o objetivo desse trabalho é propor uma estratégia que utilize técnicas de visão computacional de identifcação de objetos para realizar a contagem e estimativa de posição dos mesmos, a partir de um câmera RGB-D acoplada em um robô móvel autônomo. Essa estratégia foi validada por meio de teste em 3 diferentes cenários, que exploram a capacidade de navegação autõnoma do robô e de captura de imagem da câmera. Os resultados obtidos apresentaram, em todos os cenários, uma contagem pertinente dos objetos e um erro médio inferior 0,19 metros na estimativa de suas posições.

Keywords: Autonomos Navigation; Computer Vision; Mobile Robot; Object Count; Pose Estimate; RGB-D Camera;

Palavras-chaves: Navegação Autonôma; Visão Computacional; Robô Móvel; Contagem de Objetos; Estimativa de Posição; Câmera RGB-D;

1. INTRODUÇÃO

A evolução da robótica permitiu a criação de dispositivos capazes de replicar ou aprimorar ações anteriormente realizadas por seres humanos. Esses robôs estão cada vez mais capazes de se mover e interagir com pessoas e objetos (Tapus et al., 2007), além de também serem utilizados em âmbitos domésticos, empresariais e industriais (Bormann et al., 2015; Benotsmane et al., 2018). Dentre esses dispositivos pode-se destacar os robôs móveis, que são capazes de se locomover por ambientes de forma autônoma ou por operação remota.

Para que possam navegar de forma autônoma, informações dos ambientes ajudam esses robôs a se localizar utilizando algoritimos especializados como o *SLAM* (simultaneous localization and mapping). Esse algoritmo utiliza diferentes sensores do robô, como LiDAR ou câmeras estereoscópicas, para auxiliar a navegação (Singandhupe and La, 2019). Um exemplo disso é o Song et al. (2018), que usa SLAM baseado nas informações do LiDAR para permitir o robô tomar decisões e guiar usuários evitando obstáculos, ou em (Zhang et al., 2021), onde o robô é usado para realizar inspeções em uma usina nuclear.

No entanto, para tomada de decisões mais complexas, a utilização de câmeras é uma importante ferramenta de auxilío. Por exemplo, em Akbari et al. (2020), foi desenvolvido um sistema inteligente de navegação autonôma para exploração de espaços confinados. Para isso, foi utilizado uma câmera 3D para processar as informações semânticas

⋆ Os autores desse trabalho agradecem à UFES (pela bolsa de Iniciação Científica concedida ao primeiro autor), ao CNPq (304049/2019-0, 403753/2021-0) e FAPES (2021-V4J3L, 84322292) pelo auxílio financeiro.

do ambiente para obter um conjunto de dados completo do espaço.

Nesse contexto de obtenção mais completas de dados em ambientes, tem-se o uso da linha de pesquisa da visão computacional, que desenvolve estratégias para ensinar aos computadores a capacidade de interpretar visualmente imagens e vídeos permitindo-o enxergar na perspectiva de um ser humano (Szeliski, 2010). Uma técnica de visão computacional muito utilizada é a de identificação de objetos a partir de imagens.

Em Santad et al. (2018), os autores utilizaram um método de identificação de objetos chamado YOLO (*You Only Look Once*) (Redmon and Farhadi, 2018), para desenvolver uma aplicação de identificação de bagagens abandonadas. Métodos de identificação de objetos também podem ser utilizadas para realizar a contagem dos mesmos, como em Lee and Yang (2017), que utiliza câmeras para a contagem de objetos em cenários indústrias ou em Del-Blanco et al. (2012) para contagem e rastreamento em aplicações de segurança.

Contudo, o desenvolvimento dessas estratégias de visão citadas utilizam imagens fornecidas de câmeras fixas em determinados pontos, o que torna necessário a utilização de diversas câmeras se comunicando para não haver pontos oclusos nos ambiente. Para contornar essa limitação, faz-se necessário a utilização de robôs móveis associados a técnicas de visão computacional.

A combinação de visão computacional e robótica surge como uma forma de desenvolver robôs com sistemas de decisões ainda mais robustos para replicação e aprimoramento de tarefas humanas, contribuindo para interações human-robô-ambiente (Tapus et al., 2007). Além disso, oferece a aquisição contínua de informações referentes ao ambiente, já que a câmera se locomoverá junto com o robô.

O desenvolvimento de trabalhos com visão computacional aplicada a robótica móvel gera aplicações em diversas áreas, como na utilização de drones com câmera acoplada capaz de navegar por ambientes realizando a contagem de objetos que tenham um QR *code* anexado (Manjrekar et al., 2021) ou para identificação dos tipos de obstáculos em carros autônomos (Caesar et al., 2020).

Essas informações adicionais provinda das câmeras, auxiliam na construção de mapas que extraem informações semânticas do meio no qual os dispositivos robóticos estão inseridos, auxiliando assim no entendimento de cenários do mundo real que estão em constante mudanças, como o trabalho feito em (Bersan et al., 2018).

Então este artigo apresenta uma estratégia de criação de um mapa semântico para realizar a contagem e estimativa de posição de determinados objetos alvos utilizando uma câmera RGB-D acoplada em um robô móvel. A partir do vídeo da câmera, é utilizado o método de identificação de objetos, YOLO, para identificar objetos alvos, cuja posição no espaço tridimensional é estimada a partir da informação de profundidade proveniente da câmera. No robô móvel foi desenvolvido uma estratégia de navegação autônoma para permitir a exploração dos ambientes propostos. Com isso, esse sistema será capaz de inspecionar ambientes de forma autônoma.

Esse trabalho é divido em 4 seções. A Seção 2 apresenta os equipamentos e técnicas utilizadas no projetos. A Seção 3 apresenta os resultados e discussões da pesquisa, e a Seção 4 a conclusão e trabalhos futuros.

2. METODOLOGIA

A estratégia de contagem e localização se baseia em 3 componentes principais: o SolverBot (2Solve, Brasil), um robô móvel do tipo skid-steer, uma câmera RGB-D, D435i (Intel, EUA) e o notebook Odyssey (Samsung, Coreia do Sul).

O robô móvel é equipado com um RPLIdar A2 (SLAMTEC, China) e módulo de odometria baseado em uma IMU e um encoder para motor.

A câmera foi conectada ao notebook para realizar o processamento de imagem, vale destacar que a imagem RGB e a de profundidade estão devidamente alinhadas, e ambos os dispositivos foram posicionados sobre o robô móvel. Na figura 1, pode-se ver o robô com a câmera acoplada em seu centro.

Para comunicação entre os módulos envolvidos (câmera, robô e notebook) foi utilizado o framework *Robot Operating System* (Stanford Artificial Intelligence Laboratory et al., 2018), ROS, versão *melodic*, uma ferramenta *open source*, que possui um conjunto de ferramentas que colaboram para a construção de aplicações em robôs. Além disso, foram utilizados as seguintes bibliotecas do ROS para o desenvolver a navegação autônoma do robô: *gmapping*, *amcl*, *move base* e *tf* .

Figura 1. Robô móvel com a câmera acoplada.

Por fim, foi utilizado o método de detecção de objetos YOLOv3 (*You Only Look Once*) (Redmon and Farhadi, 2018) considerado o estado da arte para detecção de objetos em tempo real, que utiliza uma rede neural treinada para classificar objetos.

A estratégia foi dividida em 6 etapas: navegação autônoma, identificação dos objetos, distância dos objetos, localização dos objetos, contagem dos objetos e experimentos.

2.1 Navegação autônoma

Para o robô agir de forma autônoma, poder planejar caminho e desviar de obstáculos, foram feitos mapas dos ambientes usando o pacote do ROS *gmapping* (Grisetti et al., 2007). Esse pacote, utiliza a odometria junto as informações de distâncias obtidas pelo LiDAR para geração dos mapas dos ambientes de navegação. Os mapas gerados foram utilizados na navegação como base para a localização dos objetos identificados pela câmera.

Além disso, foi utilizado o pacote de localização probabilística para navegação 2D *amcl* (Adaptive Monte-Carlo Localizer)(Fox et al., 1999) para o robô se localizar nos mapas. Também foi usado, o pacote *move base*, que realiza o planejamento de rotas do robô a partir do mapa gerado, da odometria do robô e do LiDAR.

O *move base* gera dois *costmaps*, o *global costmap* que se baseia no algoritmo de *Dijkstra* (Dijkstra et al., 1959) para encontrar o caminho mais curto entre o ponto de partida até o ponto de chegada e o *local costmap*, gerado pelo *Dynamic Window Approach* (DWA) (Fox et al., 1997), que evita a colisão do robô com obstáculos durante a navegação. Na Figura 2 é possível visualizar esse fluxo de informações do *move base*. Além disso, é utilizado o pacote *tf* (TransForm), que fornece a posição dos sensores do robô em relação a sua base.

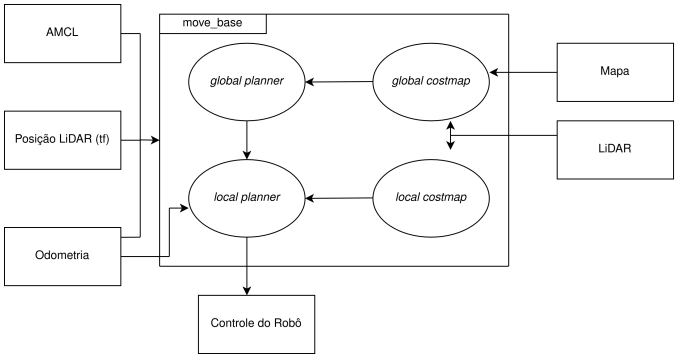

Figura 2. Fluxo de informação no *move base* utilizado nesse trabalho.

2.2 Identificação dos objetos

O YOLOv3 foi utilizado para realizar a identificação de objetos a partir das imagens fornecidas pela câmera. Cada objeto identificado possui uma *bounding box* associado a ele, um retângulo que representa a ocupação do objeto dentro dos pixels da imagem. As informações das *bounding boxes* irão auxiliar na estimativa de distância e ângulo dos objetos da câmera. Na Figura 3 é possível visualizar 4 *bounding boxes* para 4 objetos. Todo esse processamento de imagem do YOLO e a navegação autonôma são processados pelo notebook posicionado em cima do robô.

2.3 Distância dos objetos

Para estimar a distância dos objetos, foi utilizado a informação de profundidade da câmera RGB-D435. Experimentalmente, foi verificado que a distância retornada pelas câmeras de profundidade são valores perpendiculares a ela, visto que ao deslocar um objeto apenas horizontalmente na frente da câmera, sua distância praticamente não se altera.

Na Figura 3 estão objetos identificados pelo YOLO, na Figura 4 estão os mesmos objetos, porém, na câmera de profundidade. Para estimar a distância de cada objeto, foi escolhido um conjunto de pontos centrais no interior de cada *bounding box* e calculado sua média. O valor obtido é a distância perpendicular de cada objeto da câmera, porém para estimar a posição do objeto, também é necessário ter a sua informação dos ângulos horizontais e verticais.

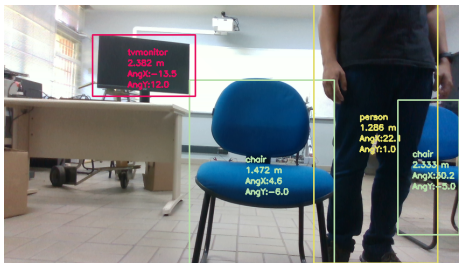

Figura 3. Detecção de objetos realizados pelo YOLO.

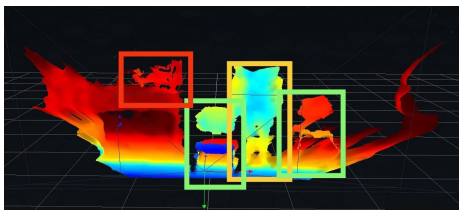

Figura 4. Distâncias dos objetos detectados na câmera RGB-D.

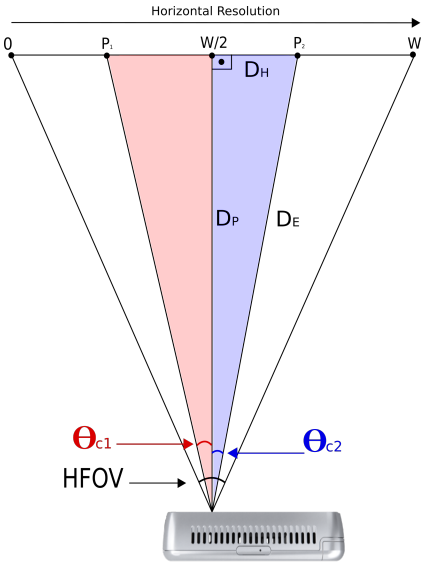

Figura 5. Estimativa de ângulo horizontal com a câmera D435i.

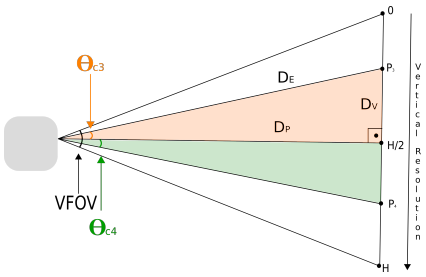

Figura 6. Estimativa de ângulo vertical com a câmera D435i.

Como é possível observar nas Figuras 5 e 6, a câmera tem uma resolução de WxH pixels, sendo W (*width*) a representação dos pixels da horizontal e H (*height*) os pixels da vertical. Para a câmera D435i utilizada, o campo de visão horizontal (HFOV) é de (69 ±1)°, de acordo com seu datasheet. Então, com objetivo de utilizar o centro da imagem sendo 0°, adotou-se a métrica que do primeiro pixel até o pixel central ($\frac{W}{2}$) tem-se um ângulo positivo de (34.5 ±0.5)°, já do pixel central até o pixel final (W) tem-se um ângulo negativo de (-34.5 ±0.5)°. Dessa forma, analisando a Figura 5 e sendo P_X o pixel que se deseja saber o ângulo, é possível obter o ângulo horizontal de qualquer pixel da imagem, utilizando a equação (1):

$$\Theta_{Hx} = \frac{(34.5 \pm 0.5) * ((\frac{W}{2}) - P_X)}{\frac{W}{2}}. \quad (1)$$

Para o campo de visão vertical da câmera (VFOV) a analogia é a mesma. Sendo assim, utilizando o valor de VFOV do datasheet de (42 ±1)°, do primeiro pixel até o pixel central ($\frac{W}{2}$) tem-se um ângulo positivo de (21.0 ±0.5)° e do pixel central até o pixel final (H) tem-se um ângulo negativo de (-21.0 ±0.5)°. Assim, para calcular o ângulo vertical de qualquer pixel da imagem, utiliza-se a Equação (2):

$$\Theta_{Vx} = \frac{(21 \pm 0.5) * ((\frac{H}{2}) - P_X)}{\frac{H}{2}}. \quad (2)$$

Com a estimativa de ângulos horizontais e verticais com as equações (1) e (2), respectivamente, e sendo D_P, na Figura 5 e 6, a distância perpendicular dos objetos da câmera discutido na Figura 4. Pode-se obter as distâncias euclidiana (D_E), horizontal (D_H) e vertical (D_V) dos objetos a partir das equações (3), (4) e (5). Sendo $cos_H(\Theta)$, o cosseno do ângulo do centro da *bounding box*, $tg_H(\Theta)$, a tangente do ângulo do centro da *bounding box* e $tg_V(\Theta)$, a tangente do ângulo da extremidade superior da mesma:

$$D_E = \frac{D_P}{cos_H(\Theta)}, \quad (3)$$

$$D_H = D_P * tg_H(\Theta), \quad (4)$$

$$D_V = D_P * tg_V(\Theta). \quad (5)$$

Para D_H, além do pixel central, também são utilizados os pixels das extremidades laterais da *bounding box* para estimar a largura dos objetos. E na vertical, o pixel da extremidade superior é usado para estimar a altura que se encontra o objeto.

2.4 Localização dos objetos

Para localizar os objetos no ambiente, utiliza-se o mapa gerado na etapa de navegação. Assim, a informação de distância dos objetos até o robô pode ser transformada em uma posição utilizando o mapa como referência, criando assim o mapa semântico.

A partir da câmera, que está acoplada no centro do robô, é possível estimar as distâncias euclidiana, horizontal e vertical dos objetos, conforme mostrado nas Equações (3), (4) e (5).

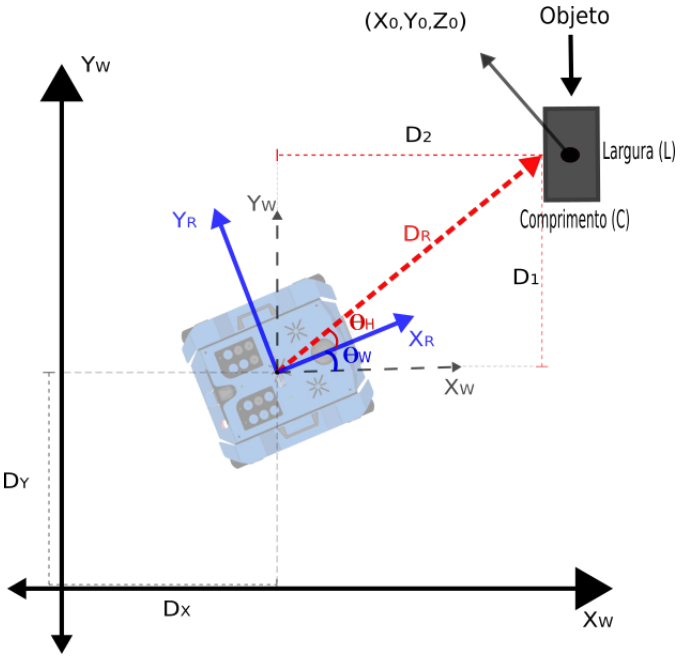

Figura 7. Robô móvel com a câmera acoplada identificando um objeto ao seu redor.

Na Figura 7, tem-se uma representação do robô móvel navegando por um ambiente previamente mapeado. Para obter a posição central dos objetos no mapa, é necessário utilizar a distância que o robô se deslocou no eixo X (D_X), no eixo Y (D_Y) e θ_W, que corresponde a orientação do robô em relação ao eixo X. Esse deslocamento é obtido pelo pacote *amcl*.

Então, para se obter a distância do robô até o centro dos objetos em relação ao eixo X e Y, decompõe-se D_R em D_1 e D_2. Para isso, utiliza-se o ângulo horizontal (Θ_H) e a distância euclidiana dos objetos da câmera (D_R), obtidos pelas equações (1) e (3).

$$D_1 = (cos(\Theta_W + \Theta_H) * D_R), \quad (6)$$

$$D_2 = (tg(\Theta_W + \Theta_H) * D_1). \quad (7)$$

Além disso, é preciso do comprimento (C) e da largura (L) dos objetos para estimar sua posição central no mapa. O comprimento é um parâmetro que precisa ser estimado manualmente de acordo com o tipo de objeto identificado. Visto que apenas com a informação 2D da imagem, não é possível saber a extensão do objeto por trás da imagem. Então, utiliza-se um comprimento condizente para o tipo de objeto identificado.

Já a largura é possível calcular utilizando a equação (4), utilizando os ângulos das extremidades da *bounding boxes* para obter duas medidas de D_H e subtrai-lás para obter a largura.

Vale ressaltar, que os valores D_1 e D_2 são componentes de distância do robô ao objeto paralelas aos eixos X e Y do

mapa. Contudo, enquanto X e Y do mapa são um ponto de refêrencia fixo, D_1 e D_2 variam de acordo o ângulo θ_W do robô. Por exemplo, na Figura 7, D_1 é paralelo a Y e D_2 paralelo a X, mas D_1 pode acabar sendo paralelo a X e D_2 a Y, dependendo do valor de θ_W. Com isso, foi feita a divisão do mapa em zonas de identificação de acordo com a rotação do robô, conforme mostra a Figura 8, para estimar corretamente a posição de cada objeto no mapa.

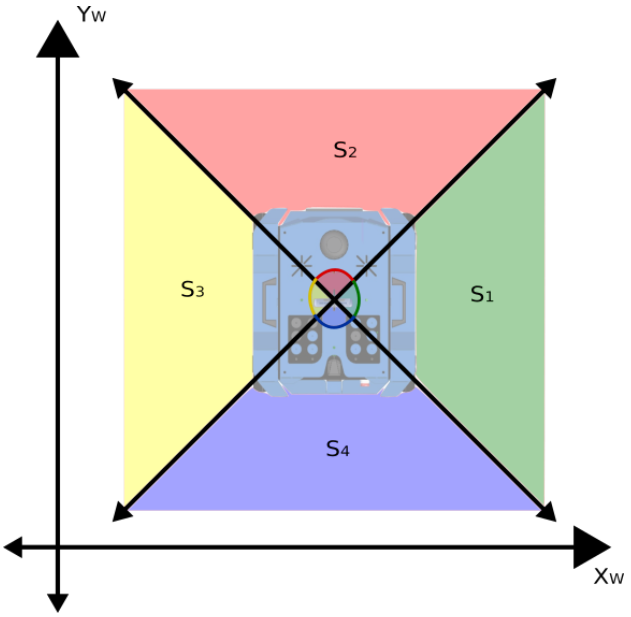

Figura 8. Zonas criadas para o localização correta dos objetos identificados no mapa.

Na zona S1, que é o caso na Figura 7, o robô está a um ângulo de $\pm 45°$ do eixo X. Já para S2, o ângulo está entre $45°$ e $135°$, para S3 está entre $135°$ e $225°$, e por fim, em S4 o ângulo do robô com o eixo X está entre $225°$ e $315°$. Para obter as coordenas X_O e Y_O de cada zona, utiliza-se as equações 8 e 9, respectivamente.

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = D_X + \begin{bmatrix} D_1 \\ D_2 \\ -D_1 \\ -D_2 \end{bmatrix} + \begin{bmatrix} \frac{C}{2} \\ 0 \\ -\frac{C}{2} \\ 0 \end{bmatrix} \tag{8}$$

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} = D_Y + \begin{bmatrix} D_2 \\ -D_1 \\ -D_2 \\ D_1 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{L}{2} \\ 0 \\ \frac{L}{2} \end{bmatrix} \tag{9}$$

Para obter a coordenada Z_O, não existe influência da zona do robô, visto que Z_O é um valor de altura, sem relação com X_O e Y_O, então seu valor equivale ao mesmo que D_V.

Com as coordenadas centrais de cada objeto (X_O, Y_O e Z_O), é possível estimar suas posições em relação ao mapa.

2.5 Contagem dos objetos

Para realizar a contagem dos objetos, utiliza-se as coordenadas centrais, o comprimento e a largura dos objetos, que foram previamente calculados. Então com esses valores, toda vez que um objeto é identificado, é possível associar a ele um retângulo no mapa.

A medida que novos objetos são identificados, a posição de seu retângulo é comparada com a posição dos objetos anteriores. Caso os retângulos estejam se sobrepondo no mapa e tenham valores semelhantes de altura (Z_O), o algoritmo assume que é o mesmo objeto, então tira a média de suas posições e gera um posição mais atualizada. Caso contrário, não havendo sobreposição, o algoritmo assume que é um novo objeto e adiciona ele a contagem de objetos. Isso é representado na Figura 9, onde os retângulos vermelhos representam objetos que já foram adicionados no algoritmo, enquanto os verdes representam novos objetos identificados.

Na primeira situação, os retângulos estão se sobrepondo, então caso eles tenham o mesmo valor de altura (Z_O), são identificados como mesmo objeto, logo, o algoritmo faz a média de suas coordenadas e atualiza a sua posição. Já na segunda situação, não há sobreposição, então são dois objetos diferentes, com isso, o algoritmo irá adicioná-lo aos objetos identifcados.

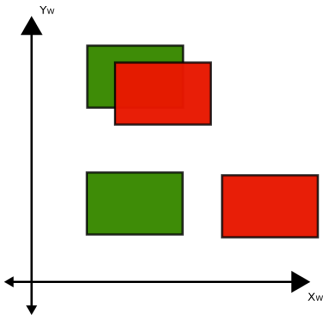

Figura 9. Critério utilizado para contagem de objeto, no caso de cima os retângulos estão se sobrepondo, o que representa o mesmo objeto, sendo o contrário no caso de baixo

2.6 Experimentos

Para validar a metodologia, foram realizados 3 experimentos com 3 diferentes cenários. Em todos os casos, o objeto-alvo utilizado para a detecção foram mochilas devidamente posicionadas nos diferentes cenários. O comprimento das mochilas escolhido foi de $0,4$m, que equivale ao comprimento da maior mochila utilizada. A escolha da mochila como objeto-alvo é apenas para validação da estratégia, sendo ela sido desenvolvida para um escopo geral.

No primeiro cenário, 5 mochilas foram posicionadas no chão ao longo do corredor, sem utilizar a informação de altura(Z_O). Foi determinado uma rota a ser realizada pelo robô, no qual ele percorre todo corredor em linha reta e retorna para sua posição inicial de forma autonôma, realizando 3 voltas. Nesse primeiro cenário foram explorados apenas as zonas S1 e S3 do robô. Na Tabela 1 , estão os valores X_O e Y_O, correspondentes a posição real central de cada mochila no corredor.

No segundo cenário, 6 mochilas foram posicionadas no chão e uma sétima foi posicionada em cima de uma mesa, a fim de explorar a identificação com diferentes

Tabela 1. Posição real do centro do objeto, em metros, das mochilas no cenário 1.

Mochila	X_O	Y_O
1	4,75	0,50
2	5,15	0,50
3	9,75	-0,75
4	12,00	-0,75
5	12,50	-0,75

alturas (Z_O). Foi determinado uma rota, na qual o robô percorre toda sala em um formato de quadrado e retorna para sua posição inicial refazendo o quadrado de forma autonôma, realizando 3 voltas. Nesse segundo cenário foram explorados todas as 4 zonas. Na Tabela 2 , estão os valores X_O, Y_O e Z_O correspondentes a posição real central de cada mochila na sala.

Tabela 2. Posição real do centro do objeto, em metros, das mochilas no cenário 2 e 3. As mochilas 6 e 7 estão nas mesmas coordenadas X_O e Y_O, porém, a mochila 7 está em cima de uma mesa, modificando seu valor em Z_O.

Mochila	X_O	Y_O	Z_O
1	1,25	0,75	0,45
2	0,25	3,50	0,45
3	-0,25	1,75	0,45
4	-1,75	3,00	0,45
5	-2,00	1,75	0,45
6	1,00	-0,75	0,45
7	1,00	-0,75	1,00

No terceiro cenário, foram adotados as mesma condições do segundo cenário, entretanto, foram colocados obstáculos no percuso da rota, para que o robô desviasse de forma autônoma e fosse possível testar a identificação de objetos.

3. RESULTADOS E DISCUSSÕES

Após a realização dos experimentos para o cenário 1, a posição central obtida das bolsas na primeira volta (X_{O_I}, Y_{O_I}) e na volta final (X_{O_F}, Y_{O_F}) está mostrada na Tabela 3. Essas foram as voltas escolhidas, pois a primeira dá o feedback inical das posições dos objetos e a volta final representa a média de todas as voltas. Na Tabela 4, tem-se um comparitivo do erro absoluto entre a posição real, mostrada na Tabela 1 e a posição obtida na Tabela 3. Além disso, as distribuições das mochilas, após a última volta, estão representadas na Figura 10, junto a rota realizada pelo robô.

No cenário 1, a volta do robô consiste em sair do ponto A ir até o ponto B e retornar ao ponto A. Ao realizar a primeira volta, o algoritmo foi capaz de identificar as 5 mochilas que estavam dispostas no ambiente e estimou a posição delas com um erro absoluto médio de 0,24 metros em X_O e um

Tabela 3. Posição estimada, em metros, do centro das mochilas no cenário 1 na primeira e última volta.

Mochila	X_{O_I}	Y_{O_I}	X_{O_F}	Y_{O_F}
1	4,90	0,33	4,90	0,37
2	5,33	0,42	5,33	0,42
3	9,88	-0,73	9,93	-0,70
4	12,54	-0,73	12,25	-0,74
5	12,72	-0,72	12,70	-0,73

Tabela 4. Erro absoluto, em metros, entre a posição real e a posição estimada das mochilas no cenário 1.

Mochila	Ex_{O_I}	Ey_{O_I}	Ex_{O_F}	Ey_{O_F}
1	0,15	0,17	0,15	0,13
2	0,18	0,08	0,18	0,08
3	0,13	0,02	0,18	0,05
4	0,54	0,02	0,25	0,01
5	0,22	0,03	0,20	0,02
Média	0,24	0,06	0,19	0,06

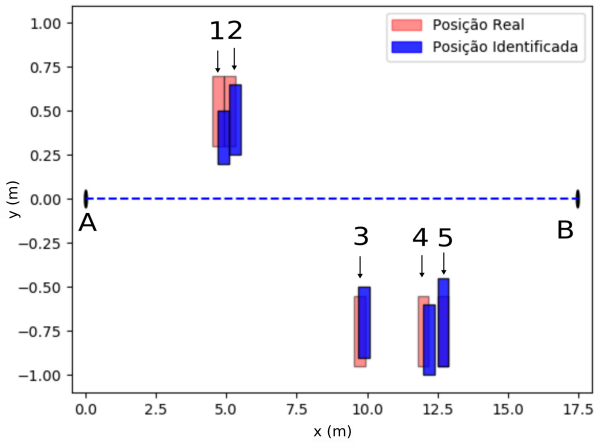

Figura 10. Distribuição estimada, em metros, das mochilas e rota do robô no cenário 1.

erro de 0,06 metros em Y_O. Após concluir todas as voltas propostas, houve uma atualização na posição das mochilas que reduziu o erro médio em X_O para 0,19 metros, mas manteve o erro médio de Y_O em 0,06 metros.

Essa redução do erro acontece, pois a medida que o robô vai realizando mais voltas, vai obtendo mais informações sobre as posições da mochilas e com isso gerando uma posição mais precisa. Outro ponto a se destacar no cenário 1, é que ao realizar a primera parte da volta (De A até B), não foi realizada a identificação da bolsa 2, devido a proximidade dela com a bolsa 1. Isso aconteceu, pois a proximidade das bolsas fez com que o metódo de identificação de objeto identificasse a bolsa 1 e 2 como uma só. Entretanto, ao completar a volta, a bolsa 2 foi corretamente identificada.

Para o cenário 2, a posição central obtida das bolsas na primeira volta (X_{O_I}, Y_{O_I}, Z_{O_I}) e na volta final (X_{O_F}, Y_{O_F}, Z_{O_F}) está mostrada na Tabela 5. Na Tabela 6 tem-se um comparitivo do erro absoluto entre a posição real na Tabela 2 e a posição obtida na Tabela 5. Além disso, na Figura 11, tem-se as distribuições no mapa das mochilas ao final da última volta, junto a rota realizada pelo robô.

No cenário 2 e 3, a volta do robô consiste em sair do ponto A ir até o ponto B, C, D, E e chegar ao A novamente. Depois disso, o robô faz o percurso contrário, isto é, passando por E, D, C, B até chegar em A. Para o cenário 2 foi definido um tolerância de raio 0,1 metros para o robô alcançar o ponto, ou seja, alcançar um distância de até 0,1 metros do ponto já é válido para que ele se desloque para o próximo. No cenário 3, a tolerância foi de 0,25 metros, pois

Tabela 5. Posição estimada, em metros, do centro mochilas no cenário 2 na primeira e última volta.

Mochila	X_{O_I}	Y_{O_I}	Z_{O_I}	X_{O_F}	Y_{O_F}	Z_{0_F}
1	1,36	0,57	0,46	1,28	0,79	0,46
2	0,20	3,32	0,48	0,20	3,32	0,47
3	-0,06	1,71	0,48	-0,05	1,70	0,49
4	-1,75	2,96	0,49	-1,79	2,96	0,50
5	-1,91	1,86	0,46	-1,96	1,88	0,48
6	1,00	-0,64	0,46	1,00	-0,64	0,46
7	1,06	-0,62	1,09	1,06	-0,64	1,10

Tabela 6. Erro absoluto, em metros, entre a posição real e a posição estimada das mochilas no cenário 2.

Mochila	Ex_{O_I}	Ey_{O_I}	Ez_{O_I}	Ex_{O_F}	Ey_{O_F}	Ez_{0_F}
1	0,11	0,18	0,01	0,03	0,04	0,01
2	0,05	0,18	0,03	0,05	0,18	0,02
3	0,19	0,04	0,03	0,20	0,05	0,04
4	0,00	0,04	0,03	0,04	0,04	0,05
5	0,09	0,11	0,01	0,05	0,13	0,03
6	0,00	0,11	0,01	0,00	0,11	0,01
7	0,06	0,13	0,09	0,06	0,11	0,10
Média	0,07	0,11	0,03	0,06	0,09	0,03

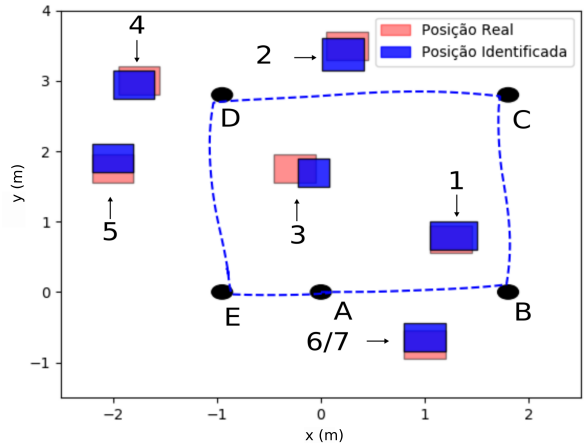

Figura 11. Distribuição estimada das mochilas e rota do robô no cenário 2.

como há obstáculos nas rotas, foi colocada uma tolerância maior.

No cenário 2, o da sala, o algoritmo foi capaz de identificar, na primeira volta, as 7 mochilas que estavam dispostas no ambiente com um erro absoluto médio de 0,07 metros em X_O, um erro de 0,11 metros em Y_O e um erro de 0,03 metros em Z_O. Após concluir todas as voltas propostas, houve uma atualização na posição das mochilas que reduziu o erro médio em X_O para 0,06 metros e o erro em Y_O para 0,09 metros, mas manteve o erro em Z_O em 0,03 metros. Essa redução ocorreu, como já mencionado anteriormente, a obteção de mais informações e atualização das posições ao longo da volta.

Para o cenário 3, o da sala com obstáculos na rota, a posição central obtida das bolsas na primeira volta (X_{O_I}, Y_{O_I}, Z_{O_I}) e na volta final (X_{O_F}, Y_{O_F}, Z_{O_F}) está mostrada na Tabela 7. Na Tabela 8 tem-se um comparitivo do erro absoluto entre a posição real na Tabela 2 e a posição obtida

Tabela 7. Posição estimada, em metros, do centro das mochilas no cenário 3 na primeira e última volta.

Mochila	X_{O_I}	Y_{O_I}	Z_{O_I}	X_{O_F}	Y_{O_F}	Z_{0_F}
1	1,29	0,60	0,46	1,25	0,51	0,44
2	0,28	3,27	0,44	0,28	3,30	0,47
3	-0,08	1,68	0,49	-0,18	1,62	0,45
4	-1,61	3,02	0,44	-1,54	2,98	0,44
5	-1,95	1,92	0,45	-1,97	1,92	0,46
6	0,73	-0,93	0,46	0,94	-0,94	0,47
7	-	-	-	0,94	-0,94	0,98
8	-	-	-	2,10	-1,41	0,47

Tabela 8. Erro absoluto, em metros, entre a posição real e a posição estimada das mochilas no cenário 3.

Mochila	Ex_{O_I}	Ey_{O_I}	Ez_{O_I}	Ex_{O_F}	Ey_{O_F}	Ez_{0_F}
1	0,04	0,15	0,01	0,00	0,19	0,01
2	0,03	0,23	0,01	0,03	0,20	0,02
3	0,17	0,07	0,04	0,07	0,13	0,00
4	0,14	0,02	0,01	0,19	0,02	0,01
5	0,05	0,17	0,00	0,03	0,18	0,02
6	0,27	0,18	0,01	0,06	0,19	0,02
7	-	-	-	0,06	0,19	0,02
Média	0,11	0,13	0,01	0,06	0,15	0,01

na Tabela 7. Além da distribuição das mochilas no mapa ao fim da última volta na Figura 12, junto a rota realizada pelo robô.

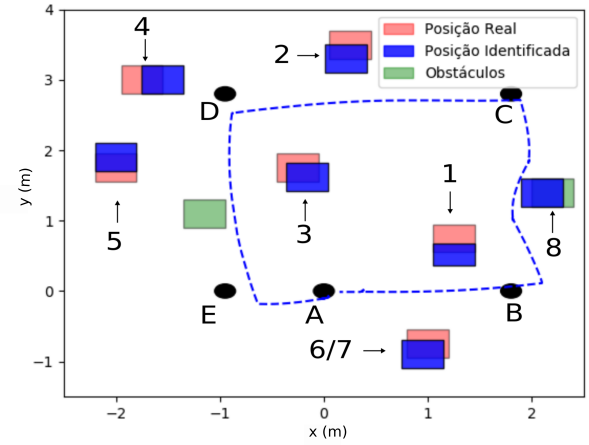

Figura 12. Distribuição estimada das mochilas e rota do robô no cenário 3.

No cenário 3, o algoritmo foi capaz de identificar, na primeira volta, 6 mochilas que estavam dispostas no ambiente com um erro absoluto médio de 0,11 metros em X_O, um erro de 0,13 metros em Y_O e um erro de 0,01 metros em Z_O. Após concluir todas as voltas propostas, a câmera foi capaz de identificar a mochila 7, que não havia sido identificada no começo, todavia, identificou um oitavo objeto, que não estava prevista no cenário. Essa mochila 8, não era uma mochila, era um dos obstáculos colocados para forçar a mudança de percurso do robô. Isso ocorreu, pois o metódo de identificação de objeto acabou identificando-o, erroneamente, como mochila.

Analisando a atualização na posição das mochilas deste cenário, houve uma redução no erro em X_O para 0,06

metros, aumentou no erro em Y_O para 0,15 metros, mas manteve o erro em Z_O em 0,01 metros. Nesse cenário, pode-se notar que mesmo com a atualização das posições ao longo das voltas, o erro em Y_O aumentou ao invés de diminuir. Nesse caso, o robô teve que refazer sua rota devido aos obstáculos, fazendo então, com que a identificação das mochilas fosse feita ao longo do percurso em pontos desfavoráveis.

4. CONCLUSÃO E METAS FUTURAS

A proposta de contagem e localização de objetos a partir de uma câmera acoplada em um robô móvel obteve resultados que indicam um desempenho satisfatório, mesmo sem ter sido feito um treinamento específico a respeito da identificação dos objetos, utilizando uma base de dados aberta. Com isso, essa estratégia aprimorada para realizar inspeções de contagem ou segurança em cenários reais.

Para isso como metas futuras, pretende-se fazer um treinamento específico para identificação de objetos, para que se implementado numa indústria, o algoritmo seja capaz de identificar objetos conforme a necessidade específica do local. Um exemplo é a contagem de produtos em armazéns, que têm o constante fluxo de entrada e saída de mercadorias diversificadas. Um treinamento com visão computacional para identificar essas mercadorias específicas, auxiliaria para o controle de contagem das mesmas. Outra aplicação é para localização de objetos não previstos em ambientes específicos, como a presença de objetos inflamáveis em áreas com atmosfera explosiva.

Além disso, novos experimentos serão planejados para ampliar a complexidade do protocolo e testes para a contagem e estimação de múltiplos objetos diferentes, adição de falsos positivos aos cenários e o estudo de formas para aprimorar a estimativa de pose 3D atráves de informações 2D.

REFERÊNCIAS

Akbari, A., Chhabra, P.S., Bhandari, U., and Bernardini, S. (2020). Intelligent exploration and autonomous navigation in confined spaces. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2157–2164. IEEE.

Benotsmane, R., Dudás, L., and Kovács, G. (2018). Collaborating robots in industry 4.0 conception. In *IOP Conference Series: Materials Science and Engineering*, volume 448, 012023. IOP Publishing.

Bersan, D., Martins, R., Campos, M., and Nascimento, E.R. (2018). Semantic map augmentation for robot navigation: A learning approach based on visual and depth data. In *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*, 45–50. IEEE.

Bormann, R., Hampp, J., and Hägele, M. (2015). New brooms sweep clean-an autonomous robotic cleaning assistant for professional office cleaning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 4470–4477. IEEE.

Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11621–11631.

Del-Blanco, C.R., Jaureguizar, F., and Garcia, N. (2012). An efficient multiple object detection and tracking framework for automatic counting and video surveillance applications. *IEEE Transactions on Consumer Electronics*, 58(3), 857–862.

Dijkstra, E.W. et al. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1), 269–271.

Fox, D., Burgard, W., Dellaert, F., and Thrun, S. (1999). Monte carlo localization: Efficient position estimation for mobile robots. *AAAI/IAAI*, 1999(343-349), 2–2.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1), 23–33.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1), 34–46.

Lee, S.H. and Yang, C.S. (2017). A real time object recognition and counting system for smart industrial camera sensor. *IEEE Sensors Journal*, 17(8), 2516–2523.

Manjrekar, A., Jha, D., Jagtap, P., Yadav, V., et al. (2021). Warehouse inventory management with cycle counting using drones. *Siddhi and Jagtap, Pratiksha and Yadav, Vinay, Warehouse Inventory Management with Cycle Counting Using Drones (May 7, 2021)*.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv*.

Santad, T., Silapasupphakornwong, P., Choensawat, W., and Sookhanaphibarn, K. (2018). Application of yolo deep learning model for real time abandoned baggage detection. In *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, 157–158. IEEE.

Singandhupe, A. and La, H.M. (2019). A review of slam techniques and security in autonomous driving. In *2019 third IEEE international conference on robotic computing (IRC)*, 602–607. IEEE.

Song, K.T., Chiu, Y.H., Kang, L.R., Song, S.H., Yang, C.A., Lu, P.C., and Ou, S.Q. (2018). Navigation control design of a mobile robot by integrating obstacle avoidance and lidar slam. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 1833–1838. IEEE.

Stanford Artificial Intelligence Laboratory et al. (2018). Robotic operating system. URL `https://www.ros.org`.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

Tapus, A., Tapus, C., and Mataric, M.J. (2007). Hands-off therapist robot behavior adaptation to user personality for post-stroke rehabilitation therapy. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 1547–1553. doi:10.1109/ROBOT.2007.363544.

Zhang, D., Cao, J., Dobie, G., and MacLeod, C. (2021). A framework of using customized lidar to localize robot for nuclear reactor inspections. *IEEE Sensors Journal*, 22(6), 5352–5359.