EDUARDO BRENDLER VIECILI

EXPLORAÇÃO ROBÓTICA ATIVA USANDO CÂMERA DE PROFUNDIDADE

Dissertação apresentada ao Programa de Pós-Graduação em Computação Aplicada, do Centro de Ciências Tecnológicas, da Universidade do Estado de Santa Catarina, como requisito parcial para a obtenção do grau de Mestre em Computação Aplicada.

Orientador: Prof. Dr. Marcelo da Silva Hounsell

Joinville
2014

Ficha Catalográfica elaborada pela Biblioteca Central

V656e Viecili, Eduardo Brendler
Exploração robótica ativa usando câmera de profundidade / Eduardo Brendler Viecili. - 2014.
140 p. : il ; 21 cm

Orientador: Marcelo da Silva Hounsell
Bibliografia: 129-140
Dissertação (mestrado) - Universidade do Estado Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Computação Aplicada, Joinville, 2014.

1.Robô móvel. 2.Busca Visual. 3.Câmera RGB-D. I. Hounsell, Marcelo da Silva. II. Universidade do Estado Santa Catarina. Programa de Pós-Graduação em Computação Aplicada. III. Exploração robótica ativa usando câmera de profundidade.

CDD 629.892 - 20.ed.

Universidade do Estado de Santa Catarina
Biblioteca Universitária da UDESC - BU
Av. Madre Benvenuta, 2007 - 88035-001 - Florianópolis, SC
e-mail: bc@udesc.br
www.bu.udesc.br

EDUARDO BRENDLER VIECILI

EXPLORAÇÃO ROBÓTICA ATRAVÉS DA BUSCA VISUAL ATIVA USANDO CÂMERA DE PROFUNDIDADE

Dissertação apresentada ao Curso de Mestrado Acadêmico Computação Aplicada como requisito parcial para obtenção do título de Mestre em Computação Aplicada na área de concentração "Ciência da Computação".

Banca Examinadora

Orientador:

Prof. Dr. Marcelo da Silva Hounsell
CCT/UDESC

Membros

Prof. Dr. André Tavares da Silva
CCT/UDESC

Prof. Dr. Fernando Santos Osório
ICMC/USP

Prof. Dr. Roberto S Ubertino Rosso Junior
CCT/UDESC

Joinville,SC, 17 de março de 2014.

RESUMO

VIECILI, Eduardo Brendler. **Exploração robótica ativa usando câmera de profundidade**. 140 f. Dissertação (Mestrado) – Universidade do Estado de Santa Catarina. Programa de Pós-Graduação em Computação Aplicada, Joinville, 2014.

Robôs móveis devem ter a capacidade de buscar (explorar o ambiente e reconhecer objetos) de forma autônoma e eficiente. Este trabalho desenvolveu um robô móvel capaz de executar a busca a um objeto (3D) em ambiente desconhecido, utilizando somente uma câmera de profundidade (RGB + distância) como sensor e executando uma estratégia de visão ativa. A Microsoft Kinect foi a câmera adotada. Também construiu-se um robô móvel (XKBO) que utiliza o Sistema Operacional Robótico (ROS), e com a arquitetura adaptada da norma STANAG 4586. Foi possível usar algoritmos existentes para reconhecer objetos 3D usando o Kinect graças as ferramentas presentes no ROS. E o uso do Kinect facilitou a geração de mapas do ambiente. Desenvolveu-se uma nova estratégia de exploração ativa que considera o esforço de movimentação para as regiões de fronteiras (áreas ocultas), e a existência de indícios da presença do objeto. As métricas utilizadas demonstram que o uso de câmeras de profundidade para tarefas de busca tem potencial para evolução por associar informação visuais com as de profundidade, permitindo que o robô possa entender o ambiente e o objeto alvo da busca.

Palavras-chave: Robô Móvel. Exploração. Busca Visual. Câmera RGB-D. Microsoft Kinect. Robot Operating System.

ABSTRACT

Mobile robots should be able to seek (explore the environment and recognize the objects) autonomously and efficiently. This work developed a mobile robot capable of performing the search for a 3D object in an unknown environment, using only one depth camera (RGB + Depth) as sensor and executing a strategy of active vision. The Microsoft Kinect was adopted as sensor. Also a mobile robot (XKBO) was build using the Robot Operating System (ROS), and with its architecture adapted from the norm STANAG 4586. A new active exploration strategy was developed in which considers the effort of the robot to move to the frontier regions (occult areas), and the presence of traces of the object. The metrics used demonstrated that the use of depth cameras for visual search tasks have a potential for deployment because associates visual and depth information, allowing the robot to better understand the environment and the target object of the search.

Keywords: Mobile Robot. Exploration. Visual Search. Camera RGB-D. Microsoft Kinect. Robot Operating System.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AV	*Active Vision*
C4I	*Command, Control, Communications, Computers and Intelligence*
GM	*Graphical Map*
HCI	*Human Computer Interface*
KBO	*Robot Cable*
NATO	*North Atlantic Treaty Organization*
PCL	*Point Cloud Library*
POMDP	*Partially observable Markov decision process*
PPM	*Position Probability Map*
ROS	*Robot Operating System*
SAE	*SAE International, Society of Automotive Engineers*
S&R	*Search and Rescue*
STANAG	NATO *Standardization Agreement*
TF	*Transformation Frames*
TPM	*Target Probability Map*
UAV	*Unmanned Aerial Vehicle*
UCS	UAV *Control System*
UGV	*Unmanned Ground Vehicles*
VSM	*Vehicle Specific Module*
XKBO	*Robot XCable*

SUMÁRIO

1 INTRODUÇÃO

Desde a introdução dos robôs na indústria, a pesquisa em robótica tem evoluído ao longo do tempo no desenvolvimento de sistemas para auxiliar os humanos em tarefas perigosas, de risco, desagradáveis e entediantes. A medida que a complexidade destas tarefas aumenta, como no caso da transição de foco das fábricas para ambientes não industriais, exige por parte dos robôs uma maior flexibilidade de interação e inteligência nas suas decisões.

Os robôs para ambientes não industriais são conhecidos como robôs de serviços. Robôs nesta área são em sua maioria robôs móveis pois estes possuem a capacidade de se movimentar e interagir no ambiente. Dos diversos tipos de robôs móveis, no presente trabalho será dada atenção especial aos que possuem uma estrutura de veículo robótico terrestre (*Unmanned Ground Vehicle* – UGV)

UGVs de serviço versáteis para aplicações em ambientes não industriais, devem possuir a capacidade de interagir com objetos dispostos pelo ambiente. Porém, esta interação só é possível através do conhecimento da localização dos objetos. No entanto, em uma situação comum onde o ambiente é dinâmico (envolvendo a presença de pessoas) não se pode esperar que os objetos se encontrem parados à disposição do campo de visão do robô, ou que o robô conheça a localização de cada objeto específico que venha a utilizar no futuro. Assim a busca por objetos em ambientes onde sua localização é desconhecida representa uma tarefa não trivial.

Para exemplificar tarefas de busca de objeto, pode-se citar: o controle de *containers* em armazém de portos; em tarefas domésticas ou de escritórios, para trazer itens como alimentos, utensílios ou ferramentas; e no resgate de vítimas em desastres urbanos.

Neste último tem-se a importância do uso que uma busca robótica pode oferecer nas primeiras horas da ocorrência de um desastre urbano que são críticas, pois as equipes de busca e resgate (*Search and Rescue* – S&R) precisam agir rápido para encontrar as vítimas em potenciais zonas de risco (VENTURA; LIMA, 2012). Desta forma as operações de S&R vem recebendo um apoio crescente de robôs e dispositivos tecnológicos semelhantes, especialmente na primeira fase da operação, onde os robôs são usados para explorar o terreno atingido por uma catástrofe, pois a situação oferece riscos para uma exploração por humanos.

Além disso os dados cartográficos conhecidos do terreno se tornam inválidos, uma vez que eventos como explosão ou terremoto destroem caminhos e salas e até abrem outras passagens. Como consequência, os robôs devem estar equipados com uma estratégia inteligente e eficiente de movimentação durante a busca em ambiente desconhecido. Conforme Liu e Nejat (2013) colocam, os benefícios do uso de robôs em S&R comparados com humanos são:

- Metódicos e incansáveis na exploração;
- Atentos e detalhados na identificação;
- Não sofrem cansaço ou “estresse”;
- Podem ser aplicados em larga escala (paralelismo para reduzir tempo);
- Podem ser reparados ou substituídos enquanto o mesmo não é possível com pessoas e cachorros;
- Não requerem ter um número suficiente de pessoas capacitadas para a tarefa no local do desastre, como especialistas que estão em outra cidade ou país, e;

- Evitam-se riscos à equipe de resgate como danos físicos (cortes, queimaduras, ossos quebrados), respiratórios (materiais tóxicos, fumaça, poeira, monóxido de carbono) e experiências emocionais traumáticas devido às cenas presenciadas.

A justificativa de que os robôs sejam autônomos na S&R, se deve ao fato de que os operadores humanos (ao operar robôs remotamente) sofrem de dificuldades de percepção em ambientes complexos, pois ao se basear apenas na imagem o operador humano pode perder a referência e orientação do robô devido a ambiguidades e as condições de iluminação, o que pode resultar na perda do robô ao ficar "preso" entre os destroços (LIU; NEJAT, 2013).

Uma especificação detalhada em relação aos requisitos dos robôs (em especial para S&R) pode ser encontrada em Habib, Baudoin e Nagata (2011), onde os pontos principais são: mobilidade mecânica para ambientes não estruturados tipo labirinto; modelagem autônoma do ambiente 3D e mapas de localização, pois sensores GPS não tem a precisão necessária e podem se tornar inválidos em ambientes internos; possuir técnicas de navegação e localização; ter autonomia de decidir qual o próximo lugar a ir; possuir capacidades de interação e comunicação, e desta forma o robô deve ser capaz de operar como um rádio, e; serem rápidos, precisos e com custo acessível.

1.1 O PROBLEMA

Em um cenário de busca visual, seja por objetos (ambiente doméstico) ou vítimas (S&R), um robô móvel autônomo deve explorar o ambiente desconhecido apenas com os seus sensores. Como as informações são adquiridas apenas no momento da exploração, isto exige portanto, encontrar uma estratégia eficiente de busca e exploração visual que levará ao sucesso da ta-

refa.

Assim a pergunta do problema é: Qual a estratégia que um robô deve adotar para encontrar um objeto específico 3D em um ambiente desconhecido?

Este problema pode ser dividido em dois: explorar o ambiente desconhecido e; reconhecer o objeto 3D.

1.1.1 Exploração do ambiente

Como Basilico e Amigoni (2011) colocam, a busca é uma tarefa crítica em tempo e pode ser descrita como o problema de maximizar a quantidade de área coberta pelos robôs no menor tempo possível. Isto, sem o conhecimento *a priori* da estrutura do ambiente ou localização das vítimas, e com restrições impostas: pelo alcance do campo de visão do sensor; tamanho físico do robô; limitações de comunicação e; capacidade de movimentação do robô. Então, a exploração necessita envolver a geração do modelo do ambiente onde, para tal, faz-se necessário: mensurar o espaço; realizar a localização do robô no espaço; definir as área exploradas, desconhecidas e os obstáculos, e; possuir estratégias envolvendo heurísticas para movimentação durante a exploração.

Com relação às estratégias de exploração utilizadas, até o presente momento, a mais usual é explorar de forma incremental a partir de pontos de localização para novas observações. Estes pontos de localização são escolhidos através de uma função conhecida na literatura como "a próxima melhor vista" (*Next Best View process*) (JULIÁ; GIL; REINOSO, 2012). O algoritmo deste processo consiste nos seguintes passos:

1. Realizar uma leitura do sensor;
2. Atualizar o mapa com a nova informação;
3. Analisar o mapa e gerar uma lista de futuras possíveis

posições para o robô;

4. Através de certas métricas definidas calcular para cada uma das futuras posições a sua pontuação, e;

5. Definir como a "próxima melhor vista" (futura posição para o robô) a que obtiver com a maior pontuação.

1.1.2 Reconhecer o objeto

Para reconhecer o objeto alvo da busca, o robô precisa ver, e isto é feito através da visão computacional, que trata do processo de adquirir informações sobre o ambiente através de imagens (PEDRINI; SCHWARTZ, 2007). E como a visão do robô é realizada através de câmeras, de acordo com Shubina e Tsotsos (2010), estas são limitadas em termos de:

- Campo de visão, dificuldade em adquirir uma imagem de toda a região que será suficiente para a detecção do objeto de interesse;

- Resolução, mesmo que o objeto se encontre no campo de visão, a resolução da imagem (muito longe) pode não ser suficiente para o reconhecimento, de modo que se faz necessária outra imagem mais próxima com resolução suficiente;

- Oclusão, existe a possibilidade de outros objetos estarem ocultando partes da área de busca. Como resultado, se exige que outros pontos de vista sejam obtidos para que o objeto não fique escondido;

- Custo, onde para cada ponto de vista adicional incorrem custos extras onde o tempo gasto para mover o hardware para o próximo ponto especificado e, em seguida, processar a imagem resultante para o objeto de interesse.

O processo de capturar a imagem na visão computacional tem duas abordagens conforme a mobilidade:

1. Passiva, onde o método de aquisição é fixo, ou seja, arquiteturas onde a câmera não possui mobilidade;

2. Ativa, onde o método de aquisição é dinâmico, ou seja, arquiteturas onde a câmera possui mobilidade, ou seja, como definido em Davies (2005): "mover a câmera de forma a focar ou observar um item em particular de interesse no ambiente".

A escolha pela abordagem de visão ativa permite através da movimentação, contornar as limitações da câmera relativas ao campo de visão e a oclusão (SHUBINA; TSOTSOS, 2010).

1.2 SENSORIAMENTO

O hardware é o ponto inicial de um desenvolvimento do sistema de busca visual, necessitando ser capaz de mensurar e capturar a imagem do ambiente. Uma alternativa seria utilizar dois sensores, um para adquirir a imagem e outro para medir a distân-cia. No entanto, é possível ter ambas as funcionalidades em apenas um dispositivo, que é o caso das câmeras de profundidade (*Red, Green, Blue and Depth* – RGB-D). Estas câmeras, surgiram em 2010 como alternativa (de baixo custo) ao uso dos dispositivos capazes de mensurar informação de profundidade como câmeras estéreo e sensores a laser.As câmeras RGB-D mais conhecidas são a Microsoft Kinect e a Asus Xtion sendo que ambas utilizam como unidade de medição o sensor da PrimeSense (GONZALEZ-JORGE et al., 2013). A Microsoft Kinect, inicialmente desenvolvida como um sensor para utilização no console do videogame Xbox, pode ser utilizada na robótica como sensor para tarefas de modelagem do ambiente, mape-

amento, localização e desvio de obstáculos (CORREA et al., 2012).

Como a Kinect integra um sistema de captura 3D (câmera de profundidade) mais imagem 2D (câmera RGB) e áudio, é possível então utilizá-lo para obter uma percepção 3D do ambiente a baixo custo, o que justifica a escolha como hardware para a aquisição do sistema de busca visual. Esta percepção 3D do ambiente é possível dada a capacidade da câmera de profundidade em gerar uma nuvem de pontos (*point cloud*) que representa as distâncias equivalentes das superfícies do ambiente a partir do ponto de visualização (*viewpoint*) da câmera. Estes dados da nuvem de pontos podem ser processados e convertidos em representações geométricas as quais são partes componentes da reconstrução do ambiente.

As vantagens relacionadas ao uso do Kinect incluem:

- Mensuração do ambiente, pois a nuvem de pontos se encontra na forma de uma matriz de *pixels* organizada, com valores de distância bem definidos, que permite realizar uma reconstrução 3D do ambiente e assim mensurar os valores físicos deste;

- Precisão, o erro de distância de cada *pixel* é definido para da faixa de distância de operação (após calibragem) dependendo exclusivamente do sensor;

- Identificação de obstáculos e ocupação, em sistemas que interagem fisicamente no ambiente a identificação de informação do que se encontra a frente, em termos de obstáculos ou ocupação, evita o choque físico (que pode danificar o sistema) durante tarefas de deslocamento no ambiente e manipulação de objetos;

- Segmentação, é possível separar os objeto uns dos outros presentes na cena através das informações tridimensionais;

- Iluminação, os algoritmos de processamento de imagem sofrem diretamente os efeitos da iluminação sobre um objeto, porém no caso da imagem de profundidade esta não é afetada pela iluminação.

Apesar de suas vantagens, o uso de câmeras RGB-D para sensoriamento em sistemas robóticos ainda não é largamente empregado dado ao seu mercado recente e em desenvolvimento. Porém, com o amadurecimento da tecnologia RGB-D e entrada de novos competidores, é possível que a próxima geração de câmeras RGB-D (menores e com maior resolução) sejam o sensor padrão nos robôs de serviço dada sua aplicação em ambientes civis.

1.3 OBJETIVO

Desenvolver um robô móvel capaz de executar a busca a um objeto específico (3D) em ambiente desconhecido, utilizando somente uma câmera de profundidade como sensor (e executando uma estratégia de visão ativa).

Os objetivos específicos deste trabalho são:

- Desenvolver um *framework* para visão ativa;
- Construir um robô móvel para a realização dos testes;
- Validar estratégias de exploração ativa para robôs autônomos na tarefa de busca por objetos em ambientes desconhecidos.

1.4 ESCOPO

As definições e limitações para o uso da tecnologia deste trabalho são:

- O espaço físico (ambiente interno devido ao Kinect) no qual será realizada a tarefa será limitado a uma sala com dimensões conhecidas, porém a sua configuração é desconhecida. Ou seja, não serão informados mapa nem descrição prévia do ambiente;
- O objeto de interesse que será o alvo da busca será apresentado e terá suas características descritivas treinadas préviamente ao início da tarefa. Quanto a sua localização no ambiente esta será em um ponto fixo porém desconhecido para o robô;
- Um único robô móvel será utilizado na busca, que não terá colaboração ou comunicação com outros robôs, e irá operar sem intervenção humana para conduzir o robô, o qual irá se movimentar de forma autônoma;
- A câmera de profundidade estará fixa na estrutura do robô sem a opção de rotacioná-la;
- No momento da realização da tarefa não existirão humanos presentes transitando pelo ambiente.

1.5 METODOLOGIA

A metodologia do trabalho consistiu na realização de uma pesquisa bibliográfica relacionada à tarefa de busca visual ativa e exploração robótica, assim como as ferramentas e técnicas necessárias para a sua implementação. E também se utilizou de uma pesquisa exploratória, pois uma solução inicial foi proposta e implementada para se identificar possíveis novos desdobramentos e melhorias.

Na caracterização desta pesquisa, pode se dizer que foi:

- Empírica, a comprovação para fundamentar a teoria é realizada com experimentos práticos (MORRIS, 1935);

- Aplicada, envolve fornecer uma solução para um problema imediato e com possível ganho (GREGORY, 1942);
- Exata, são aquelas cujos resultados são precisos, dos tipos funciona ou não funciona (WAZLAWICK, 2010);
- Tecnocrática, trata-se de um problema no qual se busca a solução através da construção de uma arquitetura que só pode ser comprovada através de testes, tendo o conhecimento a posteriori da confiabilidade e segurança destes programas (EDEN, 2007);
- Exploratória, pois o problema não é inteiramente conhecido e descrito, restando construir hipóteses que melhorem o entendimento e o conhecimento no tema. "Visa proporcionar maior familiaridade com o problema com vistas a torná-lo explícito ou a construir hipóteses." (GIL, 1991);
- Quantitativa, considera que tudo pode ser quantificável, em números e valores (SILVA; MENEZES, 2005), pois os resultados dos experimentos são avaliados através de métricas quantificáveis, e;
- Maturidade emergente, caracterizada pelo objetivo de mostrar uma nova ideia para resolução de um problema pertinente, porém não apresenta comparativos com outros trabalhos pois faz o uso de métricas ainda não padronizadas pela comunidade internacional científica. (WAZLAWICK, 2010).

1.6 ESTRUTURA DO TRABALHO

O restante do trabalho está organizado da seguinte forma: O Capítulo 2 descreve os conceitos necessários para o entender a solução, e envolve os temas de robótica móvel, a ferramenta

ROS, visão computacional, busca visual ativa e exploração de ambientes; O Capítulo 3 detalha e compara os trabalhos relacionados a tarefa de busca visual ativa; O Capítulo 4 descreve a proposta de solução; O Capítulo 5 descreve a implementação; O Capítulo 6 analisa e discute os resultados alcançados; E por fim, no Capítulo 7 são apresentadas conclusões, contribuições e as propostas para trabalhos futuros.

2 CONCEITOS

Neste capítulo serão apresentados conceitos relacionados a: robótica móvel e exploração robótica; o sistema operacional robótico ROS, sua estrutura, operação, e ferramentas para uso em um robô móvel; visão ativa e a tarefa de busca visual, que envolve o reconhecimento de objeto e em específico a técnica de reconhecimento SURF.

2.1 ROBÓTICA MÓVEL

Robôs móveis são robôs que podem se mover de um lugar para outro de forma autônoma (TZAFESTAS, 2013). Estes robôs móveis se dividem em aéreos, aquáticos, terrestres e espaciais. Sendo os terrestres (com rodas) os mais conhecidos devido a sua simplicidade e baixo consumo de energia.

Embora robôs móveis apresentem um amplo conjunto de aplicações e mercados, há um fato que acontece com praticamente todos os robôs móveis: o seu projeto envolve a integração de diversos tipos diferentes de conhecimento, o que torna a robótica móvel um campo interdisciplinar. Para resolver os problemas de locomoção, a robótica móvel deve compreender conceitos como cinemática, dinâmica e teoria de controle. Já para criar sistemas robustos de percepção, a robótica móvel deve aproveitar os conhecimentos como visão computacional, dentre outros, e empregar corretamente um grande número de tecnologias de sensores. A localização e navegação demandam também conhecimentos de algoritmos e técnicas envolvendo teoria da informação, inteligência artificial e probabilidade (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011).

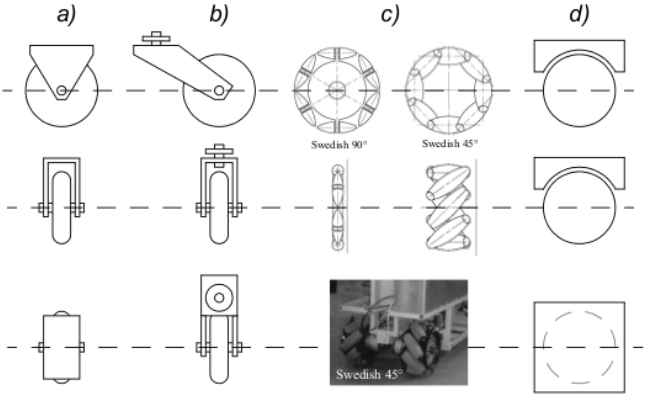
Quanto aos mecanismos de locomoção que lhe permitem mover-se, estes podem ser na forma de: andar, pular, correr,

rastejar, nadar, voar, entre outros. Sendo a maioria destes mecanismos de locomoção inspirados em equivalentes biológico, com exceção da roda que é uma invenção humana eficiente em terrenos planos.

Com relação a forma de locomoção dos robôs móveis terrestres (com rodas), esta depende do tipo, apresentados na Figura 1, onde tem-se:

- 1.a) Roda padrão;
- 1.b) Roda com pivô;
- 1.c) Roda sueca, e;
- 1.d) Roda esférica ou bola.

Figura 1 – Tipos de rodas.



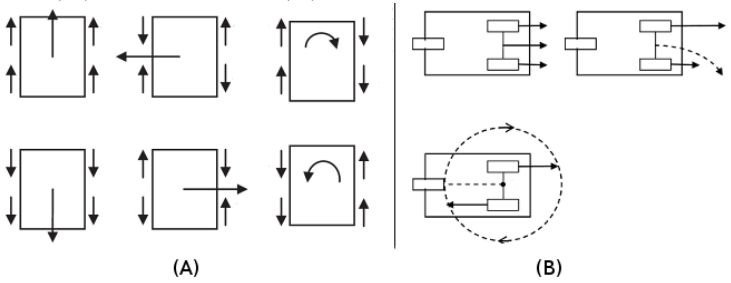
Fonte: (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011)

E número de rodas utilizadas (Figura 2), onde se classifica em:

- 2.a) Holonômicos, com três graus de liberdade de movimentação, permite andar em todas as direções no plano (*omnidirectional*).

- 2.b) Não holonômicos, com menos de três graus de liberdade, tem uma complexidade menor por serem construídos com menos de três motores e por isso são limitados em certos movimentos.

Figura 2 – Forma de locomoção de um robô móvel terrestre com rodas: (A) Holonômico; (B) Não Holonômico



Fonte: (TZAFESTAS, 2013)

Quanto ao equilíbrio do robô este é assegurado para configurações com três ou mais rodas, e é necessária uma suspensão nas situações com quatro ou mais rodas para garantir o contato com o solo.

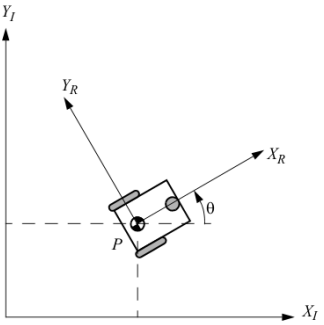
Com relação a cinemática, o modelo de um robô móvel (representado geometricamente com seus respectivos eixos de coordenada na Figura 3) é constituído como um corpo rígido sobre rodas, operando em um plano horizontal.

Os respectivas eixos de movimentação no plano são de três graus de liberdade,onde tem-se dois graus para a posição no plano (x, y) e um para a orientação no eixo ortogonal ao plano (θ) (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011).

Com relação a navegação de um robô móvel, esta requer o desenvolvimento em quatro blocos fundamentais que são (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011):

- Percepção, que é a extração e interpretação de dados dos sensores;

Figura 3 – Modelo de um robô móvel no plano.



Fonte: (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011)

- Localização, onde o robô deve determinar sua posição no ambiente;
- Raciocínio, através do qual são decididas as ações a serem tomadas para realizar o objetivo;
- E controle de movimentação, que envolve o controle dos atuadores (motores) para cumprir a trajetória planejada.

No entanto, para aplicações práticas, principalmente em ambientes reais onde são dinâmicos e imprevisíveis, é preferível utilizar um modelo cinemático probabilístico (THRUN; BURGARD; FOX, 2005), ou seja, um modelo no qual o robô tem componentes estatísticos na tomada de decisão que permite a presença de incertezas.

2.1.1 Exploração Robótica

Os robôs móveis ao navegarem de forma autônoma fazem uso de um mapa do ambiente. A criação deste mapa nos casos em que o ambiente é desconhecido, ou seja não encontra-se

disponível a priori, é chamada de exploração robótica (JULIÁ; GIL; REINOSO, 2012).

Um dos problemas da construção do mapa envolve a ausência da informação de posição global e, assim necessita de um algoritmo de localização e mapeamento simultâneos (*Simultaneous Localization and Mapping* – SLAM (THRUN, 2008)), ou seja construir um mapa e atualizá-lo ao mesmo tempo em que se localiza e mantêm controle sobre sua posição atual.

As técnicas de exploração podem ser divididas em clássicas, reativas e híbridas, onde:

- As técnicas clássicas consistem em identificar as regiões não exploradas, selecionar uma e se direcionar a ela. Após chegar na nova posição, uma nova observação é feita para atualizar com informações de ocupação o mapa métrico (que é uma matriz composta pela divisão do ambiente em um conjunto de células com dimensões físicas conhecidas). Como exemplo de informações de ocupação tem-se: célula livre; célula ocupada; ou ainda não explorada (ocupação desconhecida);

- As técnicas reativas fazem uso apenas de comportamento reativos como: manter movimentação em direção a fronteira, desviar de obstáculo e evitar outros robôs;

- E as técnicas híbridas que fazem uso de comportamentos reativos no comportamento de baixo nível, ou seja, na área ao redor do robô. E em um nível de planejamento mais elevado é usada alguma heurística, ou então uma subdivisão do ambiente em uma árvore com nós, ou um mapa topológico (um modelo no qual não possui as dimensões, apenas as relações de conexão entre certas áreas/regiões em ex: sala 1 - corredor - sala 2);

A abordagem de exploração geralmente tem como procedimento mover o robô para as áreas não exploradas e incluir

esta nova observação no mapa. As técnicas de decidir qual a próxima área a se mover, dependem da aplicação e envolvem diversos tipos de algoritmos. Por exemplo, o caso do algoritmo do caixeiro viajante (OUAARAB; AHIOD; YANG, 2014) no qual a ordem das áreas ainda não exploradas a serem visitadas são escolhidas de forma a minimizar a distância total viajada pelo robô; assim como técnicas mais simples, como o algoritmo guloso (CORMEN, 2013), no qual dirigi-se à região mais próxima; ou técnicas mais elaboradas envolvendo heurísticas.

A técnica a ser escolhida deve levar em consideração os seguintes fatores: a qualidade do mapa gerado, que depende da precisão do sensor e do algoritmo de SLAM utilizado (STACHNISS et al., 2005); robustez; e o tempo requerido para completar a tarefa. No geral, são os objetivos que determinam a técnica a ser escolhida, pois a redução de tempo de exploração e construção de mapa entram em conflito, e portanto, deve se escolher um meio termo entre tempo e qualidade.

A principal diferença entre as técnicas de exploração está em sua função objetivo, F(X), que é responsável pela seleção do próximo ponto de vista na direção a qual o robô irá se movimentar.

2.2 O SISTEMA OPERACIONAL ROBÓTICO ROS

O *Robot Operating System* (ROS) foi desenvolvido originalmente pela Willow Garage em conjunto com o Stanford Artificial Intelligence Laboratory (QUIGLEY et al., 2009). No presente momento o ROS é mantido pela *Open Source Robotics Foundation* e não é um sistema operacional no sentido tradicional de gerenciar processos e escalonamento. Na realidade trata-se de uma camada de comunicação estruturada acima de um sistema operacional hospedeiro de computação.

As motivações por trás da criação do ROS deve-se ao fato

de que escrever *software* para os robôs ser uma tarefa complexa, e para os diferentes tipos de robôs com *hardware* nada semelhantes tornam o reuso de código uma tarefa não trivial. Uma vez que esta diversidade de conhecimentos (e algoritmos) necessários está além das capacidade de um único pesquisador, busca-se então arquiteturas de *software* que permitam uma integração em larga escala entre os pesquisadores permitindo utilizar *software* disponibilizados por outros e focar em apenas uma parte de interesse.

No livro de Martinez e Fernández (2013), o ROS é apresentado como um *framework* onde sua filosofia é permitir que uma parte de software possa ser reaproveitada em diferentes robôs com pequenas mudanças de código, e criar funcionalidades que podem ser compartilhadas e utilizadas sem muito esforço.

Como facilidades, o uso do ROS oferece uma abstração de *hardware* (*drivers* para câmeras, sensores,..), controle de dispositivos de baixo nível (comunicação com microcontroladores, *drivers* de motores, entre outros) e transporte de mensagens (imagem, dados, comandos) entre processos. O ROS é baseado em uma arquitetura de grafos com uma topologia centralizada onde o processamento ocorre em nós (*nodes*) que podem receber ou postar informações, como de múltiplos sensores, controles, estados, planejamento e atuadores.

Três níveis de conceitos fazem parte do ROS:

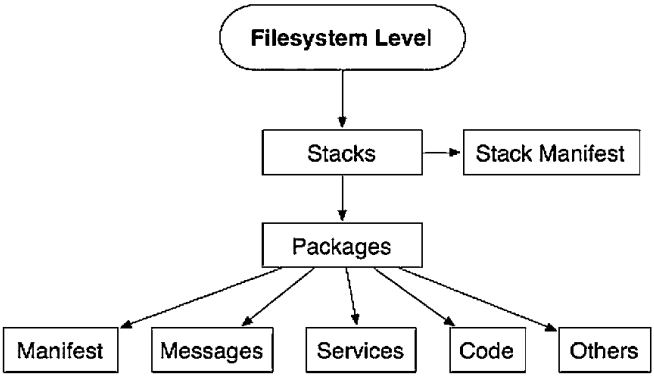
- O sistema de arquivos, que é formado internamente por uma estrutura de pastas e arquivos mínimos para operar;

- A computação em grafos, onde-se tem uma arquitetura na forma de um grafo composto por nós (programas), que permite que cada nó realize um processo individual e ao mesmo tempo exista a comunicação entre eles;

- A comunidade ROS, onde são compartilhados o conhecimento, algoritmos e códigos de qualquer desenvolvedor.

O sistema de arquivos de um programa no ROS é dividido em pastas, com arquivos que descrevem as suas funcionalidades, conforme pode ser visto na Figura 4, e tem-se:

Figura 4 – Sistema de arquivos do ROS.



Fonte: (MARTINEZ; FERNÁNDEZ, 2013)

- Pacote (*Packages*), estes formam o nível atômico de ROS. Um pacote tem a estrutura mínima e conteúdo para criar um programa dentro do ROS. Pode ter os processos de execução (nodes), arquivos de configuração e assim por diante;

- Manifesto (*Manifest*), é um arquivo (manifest.xml) que fornece as informações sobre um pacote, como licença, as dependências, informações para o compilador, e assim por diante;

- Conjuntos (*Stacks*), é quando se reúne vários pacotes com certas funcionalidades. Existem diversos conjuntos nos repositórios do ROS como por exemplo: *Navigation Stack* que é conjunto de algoritmos e funcionalidades utilizados para a navegação dos robôs; *Industrial Stack* que é para

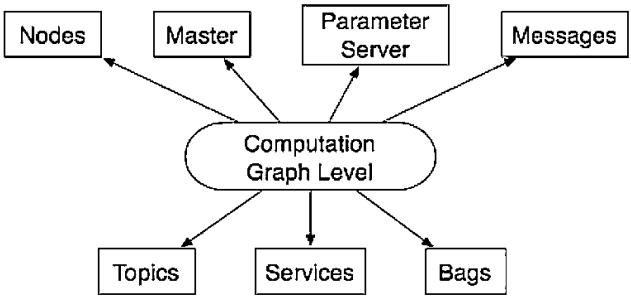
o uso do ROS em robôs industriais, contendo algoritmos como cinemática inversa para manipuladores com mais de seis graus de liberdade, percepção 2D e nuvem de pontos 3D;

- Mensagens (*Messages*), que é a informação que um processo envia para outros processos. O ROS tem vários tipos de mensagens padrões e podem ser criadas novas, onde então as descrições das mensagens são armazenadas em um arquivo *nomedamensagem.msg*;

- Serviços (*Services*), contém as descrições de serviços, e são armazenados em um arquivo *nomedoservico.srv*.

Com relação a computação em grafos do ROS, este cria uma rede em que todos os processos são conectados (Figura 5). Qualquer nó do sistema pode acessar esta rede, interagir com outros nós, e ver as informações dos dados que são enviados e transmitidos para a rede.

Figura 5 – Computação em grafos do ROS.



Fonte: (MARTINEZ; FERNÁNDEZ, 2013)

Os elementos básicos deste nível são os nós (*nodes*), o mestre (*master*), o servidor de parâmetros (*parameter server*), as mensagens (*messages*), os serviços (*services*), os tópicos (*topics*)

e os recipientes (*bags*). Assim, os elementos fornecem dados para o grafo (*computation graph level*) de diferentes maneiras, onde:

- Os nós (*nodes*) são programas em que o processamento é realizado. Para ter um programa que pode interagir com outros nós, é necessário criar um nó e conectá-lo à rede de ROS. Geralmente, um sistema terá vários nós para controlar diferentes funções (por exemplo, um nó controla o sensor, outro nó controla os motores, outro nó realiza a localização e outro a visualização), sendo recomendável ter vários nós e cada um executando ma única função, ao invés de um único nó que realiza tudo no sistema. Os programas contidos nos nós são escritos através de uma biblioteca de ROS, por exemplo o roscpp (c++) ou rospy (python);

- Mestre (*master*), é o gerente do ROS que fornece serviços de registro e pesquisas de nome para o resto dos nós. Este é necessário estar em execução para poder existir a comunicação entre os nós, serviços, mensagens e outros, pois é o mestre que controla as publicações e subscrições dos tópicos. No entanto, no momento em que dois nós se localizam estes comunicam na forma ponto-a-ponto (*peer-to-peer*). Também é possível ter o mestre em um computador e os nós executando em outros computadores;

- Servidor de parâmetro (*parameter server*), oferece a possibilidade de ter os dados armazenados usando chaves em uma localização central, e desta forma é possível configurar nós enquanto está em funcionamento ou para alterar o trabalho dos nós;

- Mensagens (*messages*), os nós se comunicam entre si através de mensagens. Uma mensagem contêm informações

de dados a serem enviados para outros nós. O ROS possui diversos tipos padrões de mensagens e é possível desenvolver tipos específicos. Como exemplo de informações a serem enviadas por mensagens, tem-se imagens, mapas, velocidades e ângulos;

- Tópicos (*topics*), cada mensagem deve ter um nome para ser encaminhada pela rede ROS. Quando um nó envia dados, é dito que o nó esta publicando um tópico. Os nós podem receber tópicos de outros nós simplesmente subscrevendo ao tópico. Não é necessário que o mesmo esteja publicando dados o que facilita o desacoplamento no desenvolvimento. Também, os tópicos do ROS podem ser transmitidos através de TCP/IP e UDP, ou seja, não é necessário os programas estarem na mesma máquina. O que permite dividir o processamento, ou operar remotamente;

- Serviços (*services*), quando é necessário requerer um pedido ou uma resposta a partir de um nó, de forma que este tipo de interação não é possível realizar apenas com tópicos;

- Recipientes (*bags*), são um formato de arquivo para gravar e reproduzir os dados das mensagens do ROS, tais como dados do sensor, que podem ser difícil de reproduzir mas necessário para desenvolver e testar algoritmos. O que permite posteriormente (com o recipiente gravado) analisar os dados, reproduzir, parar, voltar, realizar operações, interagir e visualizar nas exatas condições do experimento.

A comunidade ROS é o principal fórum para documentar as informações sobre o ROS. Qualquer pessoa pode se inscrever e contribuir com a sua própria documentação, fornecer correção ou atualização, escrever tutoriais, respostas, e mais.

O ROS possui diversas ferramentas, mas as principais e que integram este trabalho são: o sistema de transformações

geométricas; o módulo de navegação (*Navigation stack*); o servidor de mapas; e as bibliotecas compativeis com o ROS como a de visão computacional (OpenCV) e percepção 3D (OpenNI);

O sistema de transformações geométricas se faz necessário devido a interação entre os sistemas de coordenadas das diversas partes do robô. Como transformações entre coordenadas estáticas, como por exemplo, combinar os dados da câmera ou laser que está fixo em um robô para a base do robô, ou coordenadas dinâmicas, como a posição entre os eixos de um braço robótico e a sua base. Assim, o ROS possui dentro de seu sistema esses quadros de transformação (*Transformation Frames* – TF) que incluem as informações geométricas para os dados, dispositivos e componentes do robô. Esta representação tem a forma de uma árvore e os quadros recebem uma marcação de tempo, o que permite que a qualquer momento seja possível converter pontos, vetores, posições e demais dados entre partes com coordenadas de referência diferentes como: base do robô, origem do ambiente, base da camera, ponta do braço, etc.

O ROS tem muitos algoritmos que podem ser utilizados para a navegação, através do *Navigation Stack*, que usam as informações de sensores e odometria do robô, e seu controle é feito por mensagens. Para utiliza-los, se faz necessário construir e configurar um robô de acordo com requisitos e limitações como: usar apenas robôs com rodas diferenciais ou holonômicas; ter a base apenas a forma retangular; publicar in-formações e relações entre todas as articulações e posições do sensor; publicar regularmente sua informação de odometria; enviar mensagens com velocidades lineares e angulares, e; para construir um mapa, ou se localizar dentro de um mapa já construído, deve-se receber as informações de um sensor a laser (também é possível gerar artificialmente os dados do sensor a laser com outro sensor, como por exemplo o Kinect);

Na arquitetura do *Navigation Stack* o controle da navegação é dividido entre: planejamento global, que é usado para criar

o caminho para um objetivo no mapa ou uma distância longa; e o planejamento local, que é usado para criar caminhos nas distâncias curtas e evitar obstáculos. Para distâncias curtas, um envoltório quadrado (de 2x2m por exemplo) em torno do robô, permite recalcular o caminho durante o movimento caso ocorra de uma pessoa ficar na frente do robô, dentro deste envoltório.

Quanto às bibliotecas de visão computacional, tem-se a integração com o OpenCV que possui diversos algoritmos para processar imagens, como reconhecimento de objetos e *drivers* de câmeras USB, *firewire* e *ethernet*. Também, o ROS possui a integração com a biblioteca de nuvem de pontos (*Point Cloud Library* – PCL) onde é possível manipular dados tridimensionais assim como imagens de profundidade (dados do sensor Kinect), que inclui algoritmos como filtros, detecção de *features*, rotulação, *octrees* entre outros. Outra biblioteca que tem integração com o ROS é a OpenNI, que interpreta as informações do Kinect de forma a permitir uma interface de interação natural.

2.3 VISÃO ATIVA E BUSCA VISUAL ATIVA

A abordagem passiva de visão tem dominado a literatura de visão computacional e é largamente utilizada. Gibson (1986) questiona: "o que causa os olhos a se moverem em um direção ao invés de outra, e em parar em uma parte da região ao invés de outra? A resposta somente pode ser que certas estruturas particulares chamam a atenção da fóvea em direção a elas.".

O conceito de que a percepção visual do ambiente é ativa foi introduzido por (BAJCSY, 1985 apud BAJCSY, 1996) sendo um problema relacionado às estratégias de controle inteligente na aquisição de dados. Desta forma, sensores como câmeras podem ser usados ativamente ajustando seus vários parâmetros como: zoom, posição, orientação, foco, resolução, e abertura (controle de iluminação).

O uso do termo visão ativa (*Active Vision* – AV) para descrever o problema foi introduzido por Aloimonos, Weiss e Bandyopadhyay (1988) onde mostraram que um número de problemas não lineares para um observador passivo são significativamente reduzidos para um observador ativo. A ideia de que um componente ativo é necessário em sistemas de visão foi popularizado por Ballard (1991).

Como referência, a visão humana é bastante ativa, os olhos convergem ou divergem para ajustar o foco, as pupilas abrem ou fecham para adaptar a iluminação, a cabeça se movimenta ou muda de posição a fim de obter uma melhor visão de algo, e em alguns casos o indivíduo se movimenta em direção para conseguir uma vista mais próxima. Como detalhado por Pfeifer e Scheier (1999) esta adaptação é crucial para a sobrevivência em um mundo incerto e geralmente não amigável.

Definindo AV tem-se (DAVIES, 2005): "mover a câmera para focar ou observar um item em particular de interesse no ambiente; ou observar a uma região particular da imagem e interagir com o objeto até que surja uma interpretação aceitável".

O controle ativo do sensor de visão oferece um número de benefícios que são: ver parte do campo visual que de outra maneira estaria oculto; compensar um processamento; aumentar a resolução espacial; remover a ambiguidade em certas situações; permitir melhores formulações matemáticas para um problema particular (TSOTSOS, 1992).

AV tem recebido uma crescente atenção em aplicações que necessitam de análise em tempo real, como o caso da robótica, onde não é praticável processar e interpretar a cena inteira com o intuito de reconhecer tudo (FERMÜLLER; ALOIMONOS, 1995). Assim este princípio de AV, no contexto da robótica, significa poder perceber alguma característica (ex: cor) do objeto na cena mas não identificar o objeto. Apenas que existe a possibilidade do objeto estar presente ou que não existe e deve estar em um local não visto e, com base nesta informação, realizar

uma ação. Uma destas ações, por exemplo, consiste em se aproximar e interagir com o ambiente, inclusive obter imagens de outros ângulos de vista, até que possa ser possível encontrá-lo.

Em uma implementação deste sistema de AV tem-se como elementos principais a detecção de alguma das características do objeto em uma respectiva localização, e realizar a análise sobre esta informação. Ye e Tsotsos (1999) analisaram o problema e encontraram este sendo NP-Completo, pois a estratégia de AV geralmente requer informações a priori do estado do ambiente para selecionar a próxima ação a ser tomada. Realizar esta aquisição e ação requer construir um conhecimento a priori da tarefa de busca visual que reflete o estado atual do ambiente explorado, e definir quais as ações constituem uma forma eficiente do planejamento da busca visual que é uma das funções da AV.

Assim a busca visual por objetos é uma tarefa na qual o robô deve procurar por um objeto de interesse com base na visão. Uma vez que a busca é em um ambiente civil, este é dinâmico o qual impede de armazenar a localização do objeto de interesse de forma permanente. E, devido as limitações impostas pelo campo de visão do robô (este não vê a totalidade da cena), é necessário explorar o ambiente de forma a adquirir novas imagens que permitam encontrar o objeto de interesse.

Portanto, a seleção do campo de visão da próxima aquisição (próxima posição do robô) é o principal fator na busca, assim como o é na AV. Mas, cada nova aquisição envolve um custo tanto em termos de movimenta-ção quanto em processamento de imagem.

Assim é necessário gerar um modelo/mapa do ambiente de forma a selecionar os novos pontos de vista a serem adquiridos. Como consequência, esta seleção dos pontos de vista para encontrar um objeto, é similar a aquisição do modelo 3D do ambiente onde os novos pontos são determinados pelas áreas não vistas.

Conforme Ruangpayoongsak, Roth e Chudoba (2005) os robôs relacionados a tarefa de busca visual ativa tem aplicações em cenários de S&R, construção de mapa e exploração do ambiente.

2.3.1 Reconhecimento de objetos

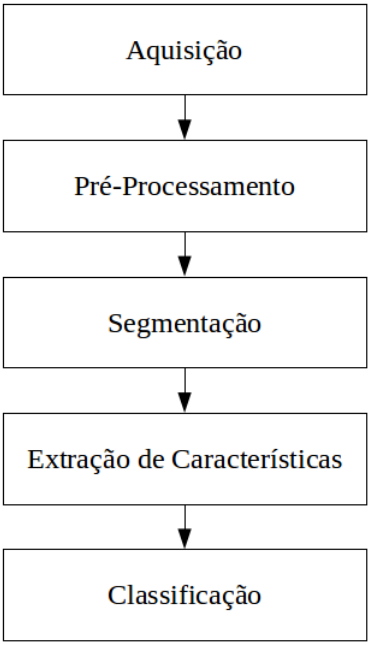
Uma componente da busca visual refere-se a capacidade de reconhecer o objeto de interesse. Embora o sistema visual biológico humano consiga extrair informações das imagens e descobrir o objeto, os sistemas computacionais de visão não tem o mesmo desempenho. De fato, a análise de imagens do mundo real é uma tarefa complexa, até detectar os contornos de um objeto geralmente falha pois ocorre de não estar claramente definido e se mistura com o ambiente ao redor (HYVÄRINEN; HURRI; HOYER, 2009). Também existem os problemas relacionados a oclusões de parte do objeto e variações de iluminação que ocorrem.

No início das pesquisas em reconhecimento de objetos a abordagem mais usual era utilizar uma cena com um único objeto em um fundo uniforme, sendo possível segmentar e extrair. Recentemente o que se tem feito é adquirir o máximo de imagens e usar uma técnica de aprendizado de máquina (KALAL; MATAS; MIKOLAJCZYK, 2010) ou *deep learning* (HINTON; SALAKHUTDINOV, 2006). Porém, esta abordagem nem sempre é possível pois, na maioria das situações, não existe uma base de imagens suficiente.

Assim na tarefa de interpretar uma imagem, geralmente se recorre a intuição de decompor o problema em subproblemas e diversos níveis de representação. Como exemplo, o processo de reconhecimento de objetos pode ser descrito em uma sequência de etapas como visto na Figura 6:

1. A partir dos *pixels* na aquisição, onde se realiza a captura

Figura 6 – Etapas do processo de reconhecimento de objetos



Fonte: produção do próprio autor

da imagem com a câmera;

2. Pré-processamento, onde parâmetros como iluminação, remoção de ruídos, entre outros, são corrigidos;

3. Segmentação, processo onde a imagem de interesse é segmentada do fundo;

4. Extração das características, onde identifica-se na imagem as características;

5. E termina-se na classificação, onde é feita a análise das características e permite definir a qual objeto se refere.

Penatti (2009) realizou um levantamento de diversas técnicas de extração de características e afirma que o sistema para reconhecimento de objetos deve ser tolerante a transformações de rotação, escala, translação, iluminação, mudança de ponto de vista e organização.

Uma das características para descrever um objeto são os pontos correspondentes (*keypoints*) em uma imagem, que são uma região de *pixels* pertencente a um objeto. A forma de descrever um ponto correspondente se dá através de um descritor, como por exemplo o caso do SIFT (LOWE, 2004) o descritor é calculado pelo histograma do gradiente de orientação ao redor do ponto de interesse e armazena em um vetor de 128 posições (8 orientações em 4x4). Uma limitação do SIFT é que a dimensão do descritor impacta diretamente no processamento.

Para compensar a limitação em velocidade do SIFT, Bay, Tuytelaars e Gool (2006) apresentaram um método de extração de características denominado "SURF: *Speeded Up Robust Features*". O descritor é dado pela distribuição de *Haar-Wavelets* (VIOLA; JONES, 2001) horizontais e verticais nos pontos dentro da região de vizinhança. A operação também pode ser realizada em paralelo, e a licença de uso do SURF é livre.

2.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capítulo foi visto que um robô móvel possui diversas formas de locomoção, e dentre elas escolheu-se para este trabalho a movimentação do tipo não holonômico devido a sua menor complexidade. Para o desenvolvimento do robô foi visto que se faz necessário quatro blocos fundamentais (percepção, locomoção, raciocínio e controle de movimentação) os quais serão desenvolvidos nos próximos capítulos. Também foi visto a possibilidade do uso da ferramente ROS na implementação software, a qual será utilizada no robô devido as suas vantagens e

benefícios principalmente a questão da navegação. Com relação a tarefa a ser executada pelo robô, uma breve revisão foi apresentada na qual determinou-se que um método de visão ativa se faz necessário e portanto, será o método utilizado para este trabalho.

E por fim, dada a possibilidade de utilizar qualquer algoritmo de reconhecimento de objeto (o qual não é o foco deste trabalho), por familiaridade e experiencia do autor em outros trabalhos, o algoritmo SURF foi escolhido. O próximo capítulo detalhará os trabalhos relacionados usados como referência no desenvolvimento da proposta e implementação deste trabalho.

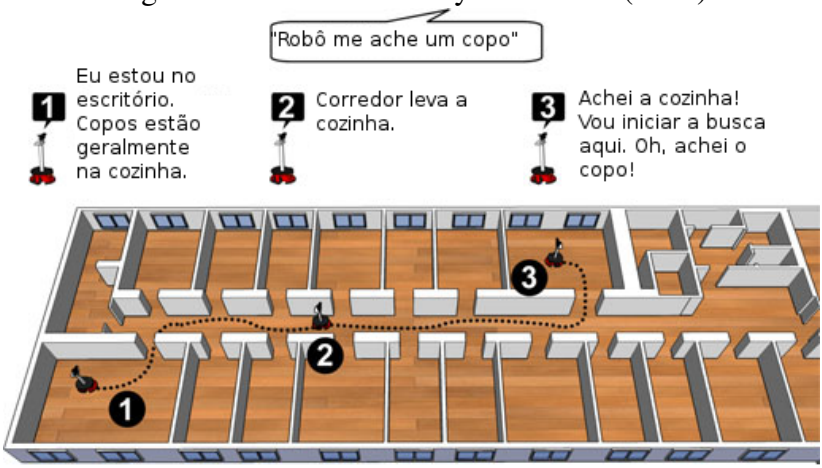
3 TRABALHOS RELACIONADOS

Neste capítulo serão apresentados os trabalhos em busca visual ativa mais semelhantes a este trabalho, descrevendo suas características e estratégias utilizadas. E por fim, será mostrado um comparativo e análise.

3.1 BUSCA VIA EXPLORAÇÃO EM GRANDES AMBIENTES

O trabalho de Aydemir et al. (2013) assume um cenário realista no qual o robô tem como tarefa, encontrar um objeto em um ambiente de grandes dimensões como mostrado na Figura 7.

Figura 7 – Ambiente em Aydemir et al. (2013)



Fonte: Adaptado de Aydemir et al. (2013)

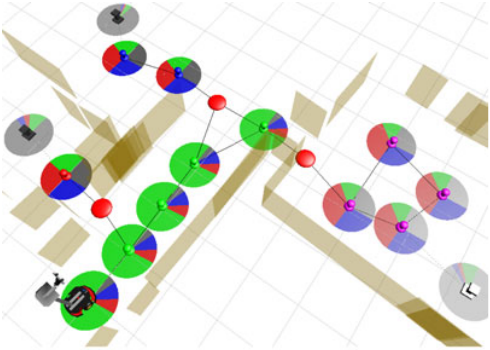
A função custo é definida pelo tempo total (menor tempo para chegar ao objetivo). O robô não possui conhecimento prévio

do ambiente e, o sistema apresentado é capaz de extrair dados semânticos pela aparência, geometria e topologia do ambiente combinando com o conhecimento semântico dos espaços interiores para gerar os locais de interesse.

O método utilizado é probabilístico, e o algoritmo de mapeamento utiliza grafo. A técnica de planejamento utilizada é uma combinação de processo de decisão de Markov parcialmente observável (*partially observable Markov decision process* - POMDP (SPAAN, 2012)) com planejamento contínuo. Em termos de desempenho foi comparado com métodos de estratégias de busca gulosa e estratégia de busca humana (obtida com o uso de participantes), e concluiu-se que o uso do conhecimento semântico melhora a eficiência da busca.

Quanto ao modelo de espaço de busca este é representado por um grafo indireto chamado mapa de lugar (*place map*). Os nós do grafo correspondem a lugares discretizados do ambiente. As arestas representam os caminhos diretos entre os lugares. Assim, juntos, os lugares e caminhos representam a topologia do ambiente. A Figura 8 mostra o *place map*, onde os lugares são agrupados em salas, e as partes inexploradas recebem o nome de *placeholders*.

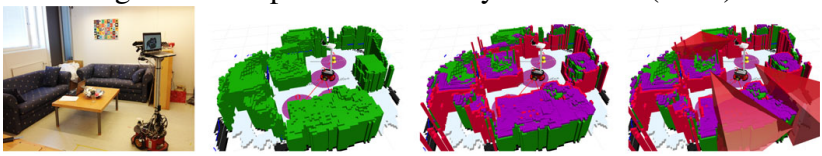
Figura 8 – Mapa topológico em Aydemir et al. (2013)



Fonte: Aydemir et al. (2013)

Também é criada, para cada sala, uma representação em um modelo métrico 3D (representado na Figura 9) que suporta seleção de ponto de vista e desvio de obstáculos. Esta representação é feita utilizando uma matriz 3D de células de mesmo tamanho (Figura 9), onde cada célula contém a informação de ocupação e a probabilidade do objeto estar na célula.

Figura 9 – Mapa métrico em Aydemir et al. (2013)



Fonte: Aydemir et al. (2013)

Os experimentos foram realizados em um ambiente com dimensões 33 m x 12 m, compostos por 15 salas diferentes, sendo 12 delas escritórios. O robô utilizado foi um Pioneer 3, equipado com Hokuyo URG *laser scanner*, uma câmera Microsoft Kinect, e uma câmera de alta resolução para o reconhecimento de imagem. Os objetos utilizados foram uma caixa de cereal, um grampeador e uma xícara. Para detecção do objeto foi utilizado o *BLORT visual toolkit* (MÖRWALD et al., 2010). Para visualização do mapa utilizou *CURVE software library* e as células do mapa 3D tem tamanho de 10cm x 10cm x 10cm.

3.2 BUSCA VIA RESTRIÇÕES

O trabalho de Andreopoulos et al. (2011) diz respeito a uma busca visual por um objeto em um ambiente tridimensional com restrições de tempo e utilizando um robô humanoide bípede com 26 graus de liberdade.

As suposições feitas referem-se a:

- Existência de apenas uma instância do objeto na cena;
- Dependência do sistema em três sistemas de coordenadas que são: calcanhar (*heel*), localizada no centro do robô humanoide; global (*world*), localizada na posição inicial no ambiente; e olho (*eye*), localizada na câmera;
- A região 3D é limitada ao espaço de busca, onde o mapa do alvo (*target map*) é a discretização do espaço de busca em células tridimensionais;
- o mapa de obstáculo (obstacle map) é uma discretização do espaço de busca na forma binária em células tridimensionais indicando se contém uma estrutura sólida;
- e mapa não visto (*never-viewed*) também é uma divisão do espaço de busca na forma binária em células tridimensionais indicando se a célula já foi vista.

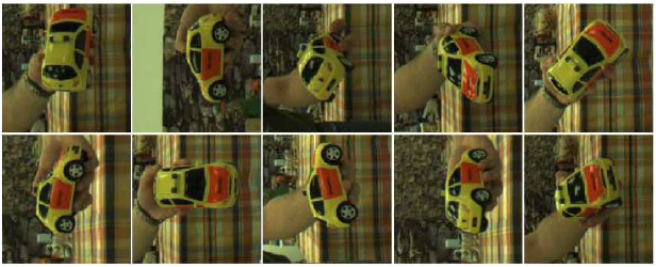
As discretizações de todos os mapas consiste em células de igual volume com dimensões de 5 cm x 5 cm x 5 cm. O reconhecimento do objeto é realizado através de uma arquitetura a qual consiste de duas etapas: a primeira etapa refere-se ao aprendizado do objeto o qual é mostrado em diferentes ângulos em frente a câmera do robô (Figura 10) e este gera um modelo composto com as *features* relevantes o qual é armazenado em um banco de dados; a segunda etapa consiste em buscar no banco de dados o modelo o qual tem maior relevância em relação a imagem vista.

A representação do robô está presente na Figura 11, onde pode-se observar suas respectivas coordenadas e a divisão do ambiente em células.

A estratégia utilizada tem a seguinte sequência:

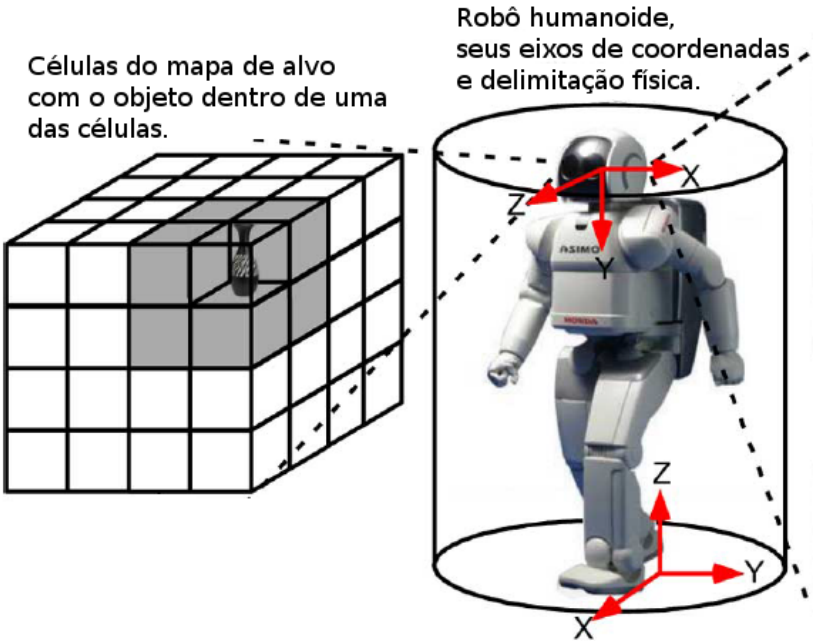
1. O processo inicia com a atualização dos mapas tridimensionais (*update 3D maps*), e encerra o processo caso a

Figura 10 – Treinamento do modelo através de imagens de vários ângulos.



Fonte: Andreopoulos et al. (2011)

Figura 11 – Representação do robô no sistema.



Fonte: Adaptado de Andreopoulos et al. (2011)

condição de solução esteja completa. Esta função de atualização dos mapas é executada no mapa do alvo, que contém valores de probabilidade de se encontrar o objeto para cada posição, e no mapa de obstáculo juntamente com a "regiões não vista" (*never-viewed map*), que representa a estrutura do ambiente;

2. A função responsável pela geração de hipóteses e cinemática inversa gera uma lista de hipóteses. Esta lista de hipóteses com as informações da estrutura do ambiente (mapas de obstáculos e regiões não vistas) é usada no planejamento de caminhos que tem como saída uma lista de caminhos ótimos;

3. Com base nesta lista de caminhos, e demais informações de mapas (do alvo, de obstáculos e regiões não vistas) e a lista com as hipóteses, chega-se na hipótese ótima. Esta hipótese tem como saída a coordenada para o próximo movimento. Esta coordenada do próximo movimento é transmitida então para o controlador de movimento do robô. Que executa duas funções, repassar a informação de odometria, e adquirir um novo par de imagem;

4. Do par de imagens obtidas é realizada a extração da imagem estéreo que gera o mapa de profundidade e a imagem usada para o algoritmo de reconhecimento do objeto, que é repassado para os mapas de confiabilidade do alvo. Com estas informações volta-se para o início do processo.

3.3 BUSCA VIA ATENÇÃO VISUAL

No trabalho de Shubina e Tsotsos (2010) considera-se o problema de selecionar o ponto de vista e campo visual para a tarefa de encontrar um objeto conhecido em um ambiente tridi-

mensional desconhecido com o uso de um robô móvel. Este problema é considerado como instância de uma abordagem de AV. Para otimizar a busca é utilizada uma técnica de atenção visual. A atenção visual é um fenômeno relacionado a área de interesse da imagem (ROTHENSTEIN; TSOTSOS, 2008).

Esta tarefa de busca visual realizada por Shubina e Tsotsos (2010) tem as seguintes suposições estabelecidas:

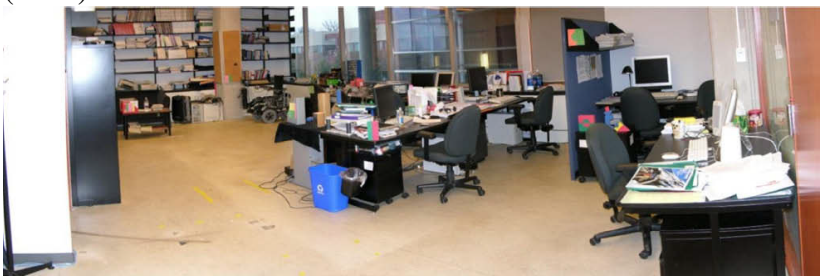
- Os limites do espaço tridimensional do ambiente a ser utilizado são conhecidos, porém a sua configuração interna não;
- O mapa do ambiente é dividido em uma matriz de células cúbicas não sobrepostas;
- A operação consiste em obter uma imagem dada uma configuração da câmera (com posição e orientação) e analisar se o objeto está presente na imagem;
- A cálculo do custo para a operação retorna o tempo necessário para a execução, o que inclui mover o sensor de uma configuração para outra, adquirir uma imagem, processar o algoritmo de reconhecimento e atualizar o estado do conhecimento sobre o ambiente;
- A função de busca está baseada na probabilidade de encontrar o objeto a cada nova operação realizada.

O problema é NP-Difícil e a estratégia de busca é dividida em dois sub problemas: onde olhar e, se mover. Durante o primeiro subproblema o robô tem sua posição fixa e altera apenas o campo de visão adquirindo novas imagens. Na solução do segundo subproblema é calculada a nova posição para o qual o robô deverá se deslocar com base no conhecimento do ambiente devido a posição atual.

No experimento, o robô utilizado é um Pioneer 3 e é usada uma câmera estéreo tanto para obter as informações em

relação ao ambiente (mapa de profundidade) quanto detecção do objeto. A técnica utilizada no reconhecimento do objeto é um algoritmo de SIFT (LOWE, 2004) e os resultados são desenvolvidos em um ambiente com dimensões de 9m x 5m x 2.5 m, visto na Figura 12, sendo o piso dividido em um matriz de 1 x 1 m^2 e o espaço de busca em cubos *voxels* com 5 cm x 5 cm x 5 cm. O tamanho do objeto é de 23cm.

Figura 12 – Ambiente de testes utilizado em Shubina e Tsotsos (2010)



Fonte: Shubina e Tsotsos (2010)

A primeira das etapas da busca é representada na Figura 13, onde tem-se o mapa gráfico (*Graphical Map* – GM), o mapa de probabilidade do alvo (*Target Probability Map* – TPM) e mapa de probabilidade da posição (*Position Probability Map* – PPM). O retângulo amarelo representa o robô, enquanto que o retângulo vermelho representa o objeto. As áreas verdes representam onde existe uma ocupação. No mapa de probabilidade de posição (PPM) os valores são representados em escala de cinza, onde claro igual a maior probabilidade e escuro menor. O campo de visão do robô para reconhecimento do objeto é representado graficamente por um cone e tem um alcance diferente do sensor estéreo usado para marcar os obstáculos. O processo consiste em:

- Posição 1 do robô, estado inicial (canto superior à esquerda

na Figura 13): Probabilidade de posição (PPM) antes da busca iniciar. A solução de Shubina e Tsotsos (2010) parte do pressuposto que, como não existe nenhuma informação sobre o ambiente ainda, a maior probabilidade é que esteja no centro do ambiente, ou seja, nessa posição uma grande área pode ser explorada;

- Posição 1, primeira vista (segunda linha na Figura 13, em GM e TPM): Como o robô desconhece a forma do ambiente e a localização do objeto, a primeira ação é direcionar a vista para o centro da sala. Além de que, ao direcionar para o centro, o robô explora a maior área possível a partir desta posição o que maximiza a probabilidade de encontrar o alvo;

- Posição 1, segunda vista (terceira linha na Figura 13): a cada ação o robô atualiza a ocupação do ambiente com os novos obstáculos detectados;

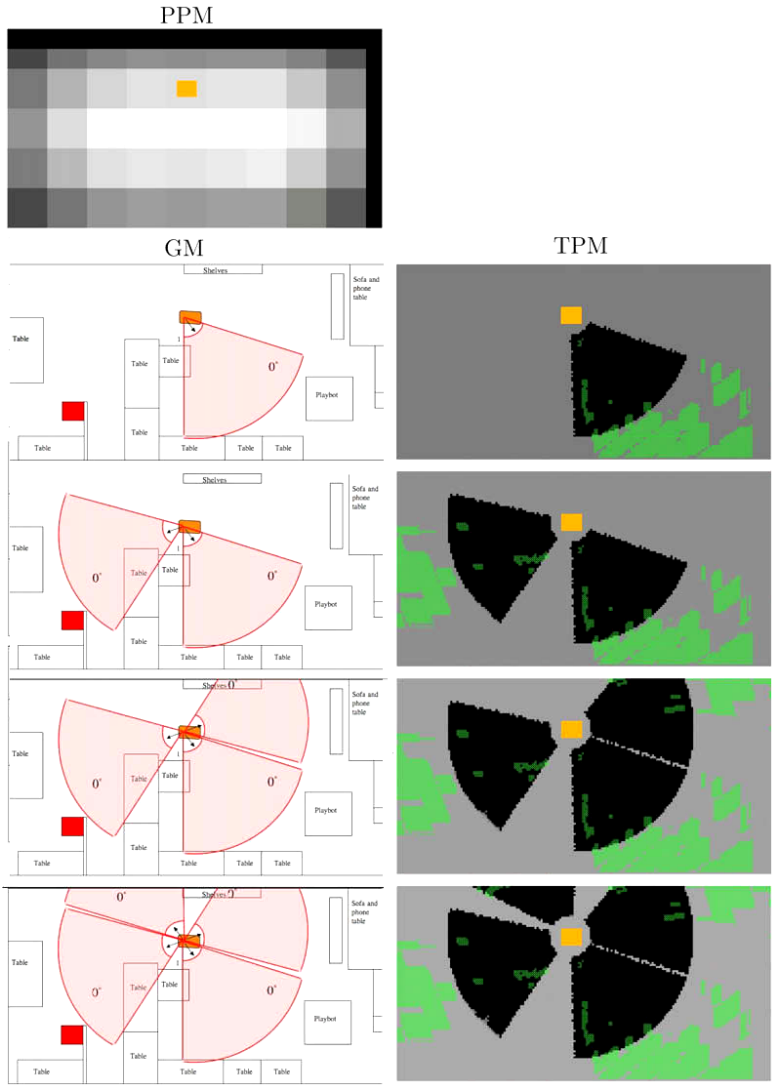
- Posição 1, terceira vista (quarta linha na Figura 13): com o avanço da busca, as regiões do mapa de probabilidade do alvo (TPM) são atualizadas de forma que as regiões não vistas tornam-se mais claras (aumenta a probabilidade), enquanto que as áreas já vista se tornam escura (baixa probabilidade);

- Posição 1, quarta vista (ultima linha na Figura 13): com esta vista a probabilidade de detectar o alvo nesta posição tem seu valor abaixo do limiar (*threshold*) estabelecido para que o robô então passa a buscar uma nova posição para se movimentar.

A continuação da busca pelo objeto segue na Figura 14:

- Escolha da posição 2 do robô (canto superior esquerdo na Figura 14): A escolha da nova posição do robô é realizada

Figura 13 – Representação de uma das etapas de buscas descritas no trabalho de Shubina e Tsotsos (2010)



Fonte: Shubina e Tsotsos (2010)

pelo mapa de probabilidade de posição (PPM), que considera apenas regiões onde o robô pode alcançar através de uma linha reta e escolhe a com o maior valor de probabilidade. As posições que não são possíveis são marcadas com uma cruz;

- Posição 2, primeira vista (segunda linha na Figura 14): O robô se desloca para esta nova posição e faz uma observação;

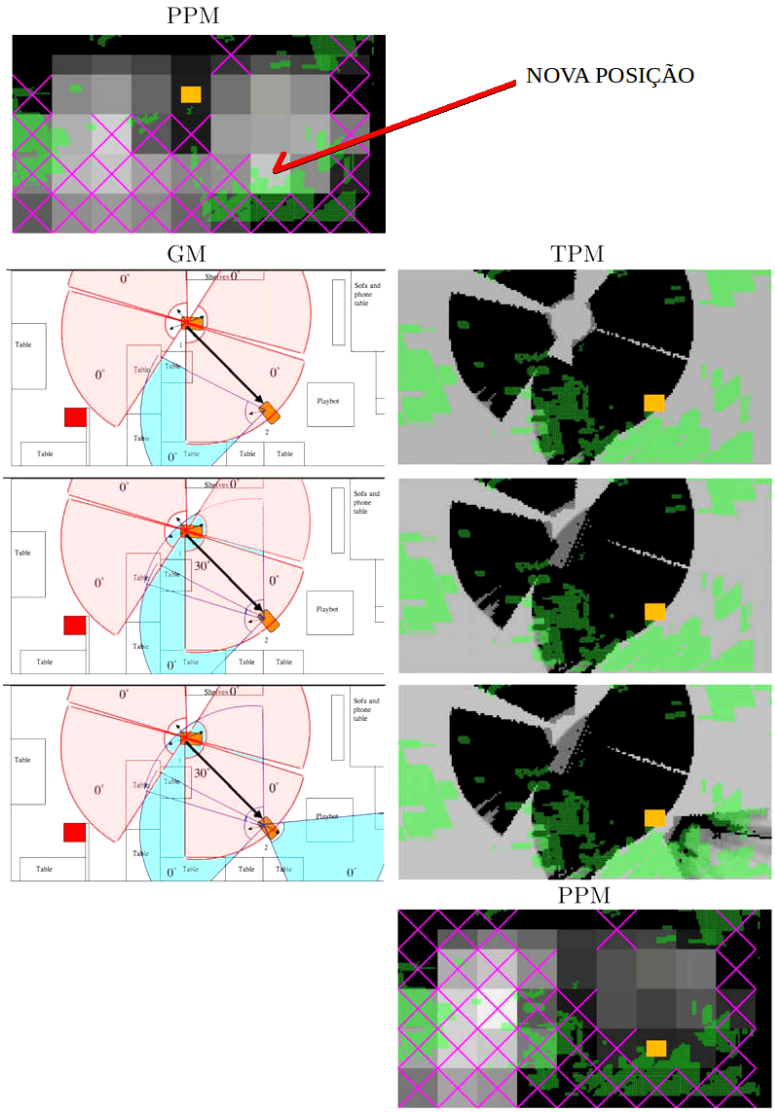
- Posição 2, segunda vista (terceira linha na Figura 14): como a região horizontal que tem o maior potencial para a próxima vista já foi observada anteriormente (na posição 1) o robô então escolhe uma vista que está próxima da horizontal porém com uma alteração em torno de 30 graus;

- Posição 2, terceira vista (quarta linha na Figura 14) : a última vista para a posição, pois o valor de probabilidade nesta posição fica abaixo do limiar necessário para acionar a função mover;

- Escolha da posição 3 (última linha na na Figura 14) : Embora o melhor lugar para se mover se encontra na esquerda, esta posição não é possível de chegar, portanto é escolhida a próxima região que se encontre em um ponto acima da atual;

- Continuação da busca: e a busca continua com essa estratégia até encontrar o objeto.

Figura 14 – Representação de uma das etapas de buscas descritas no trabalho de Shubina e Tsotsos (2010)



Fonte: Shubina e Tsotsos (2010)

3.4 COMPARATIVO E ANÁLISE

Com base nos trabalhos relacionados à Busca Visual Ativa foi possível construir a Tabela 1, onde se identificou as características principais a serem consideradas como comparativo.

Dentre as contribuições dos trabalhos relacionados destacam-se:

- Em Shubina e Tsotsos (2010), as métricas utilizadas e a estratégia de busca;

- Em Andreopoulos et al. (2011), as caracterizações dos mapas, o sistema de coordenadas utilizado e o modelo de arquitetura para o sistema;

- E em Aydemir et al. (2013), o uso do Kinect e a utilização de um comparativo para a busca baseado na estratégia cognitiva humana.

Porém, neste trabalho, não será utilizada nenhuma representação semântica do ambiente, como no trabalho de Aydemir et al. (2013), pois representações semânticas exigem por parte do robô um conhecimento prévio das relações do ambiente com o objeto e se aplicam a ambientes maiores subdivididos. No escopo deste trabalho o robô não possui nenhum conhecimento prévio em relação ao ambiente ou a utilidade do objeto, ou seja, se o ambiente é uma cozinha ou escritório é indiferente para o robô.

A geração de um mapa 3D, como no trabalho de Andreopoulos et al. (2011), não será realizada, devido ao robô móvel deste trabalho não possuir mobilidade no eixo ortogonal ao solo. Assim a construção de um mapa 3D apenas aumenta o processamento sem trazer ganhos significativos à busca.

Tabela 1 – Comparativo dos trabalhos relacionados.

[!htb]

	Shubina e Tsotsos (2010)	Andreopoulos et al. (2011)	Aydemir et al. (2013)	Este Trabalho
Ambiente	Limitado	Limitado	Extenso	Limitado
Tipo de Câmera	Estéreo	Estéreo	Kinect + Câmera HD	Kinect
Tipo de Robô	Pioneer 3	Asimo	Pioneer 3	XKBO
Tipo de Mapa	Métrico	Métrico	Métrico e Topológico	Métrico
Téc. de Busca Visual	Atenção Visual	Por Restrições	Probabilístico	Fronteiras
Téc. de Rec. do Objeto	SIFT	Aprendizado	BLORT	SURF
Algoritmo de Navegação	Grafo	Dijkstra's	POMDP	Nav. Stack
Análise Semântica	-	-	✓	-
*Tempo	✓	✓	✓	✓
*Número de Movimentos	✓	-	-	✓
*Distância Percorrida	-	✓	✓	✓

(*) Métricas utilizadas

Fonte: produção do próprio autor

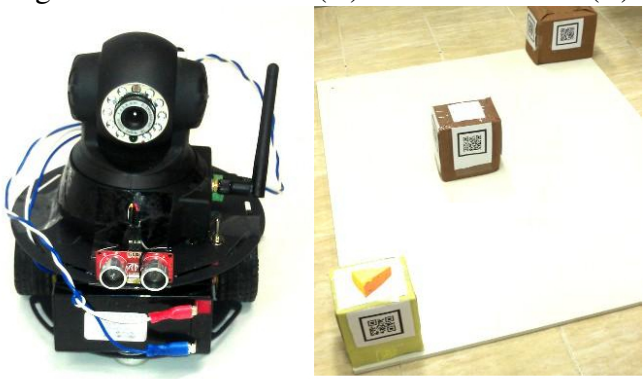
Não se fará o uso de uma câmera estéreo, como no trabalho de Shubina e Tsotsos (2010), pois exige um processamento extra entre as duas imagens para obter o mapa de profundidade, que é diretamente dependente da distância entre as câmeras, a distância focal, a sua calibração e o algoritmo para encontrar os *pixels* correspondes entre as duas imagens utilizado. E, com relação a precisão do mapa de profundidade, este varia conforme a distância, onde obtêm-se regiões com uma margem de erro, sendo necessário um pós-processamento para utilizar o mapa de profundidade enquanto que o Kinect já dispõe o mapa de profundidade para o uso final.

Este trabalho permite usar vários algoritmos de reconhecimento de objeto, não só uma solução específica como os trabalhos apresentados.

4 PROPOSTA

O XKBO (ex·ca·ble [ɛksˈcābəl]) é um robô móvel que tem como objetivo realizar exploração e busca visual ativa. A origem do nome tem referência ao personagem Cable da Marvel Comics©, que é filho do personagem Cyclops que tem apenas um olho, como o robô que tem apenas um sensor. A inclusão do X na frente do nome é para representar uma versão aumentada (*eXtended*), pois a sua criação é uma continuação do projeto KBO, original iniciado por Alcantara (2012) e seguido em Viecili et al. (2012). O KBO consistia em uma plataforma para o ensino de engenharia e robótica composta por um robô modular simples de arquitetura aberta, com o acionamento das rodas em malha aberta, no qual se fez uso da visão computacional através de uma *webcam* convencional para permitir a exploração de um ambiente desconhecido. A Figura 15 apresenta o KBO (A) com duas rodas e uma câmera acoplada e o ambiente (B) composto por caixas marcadas com código de barra bidimensional (QR-Code).

Figura 15 – Robô KBO (A) e seu ecosistema (B)



Fonte: (VIECILI et al., 2012).

As limitações do KBO que justificaram o desenvolvimento do XKBO eram:

- o tamanho físico insuficiente para suportar o processamento local contornado pelo uso de uma conexão externa (nem sempre presente) para tomada de decisão;

- seu desenvolvimento não seguiu nenhum modelo nem padrão para *hardware* nem *software*;

- tipo de câmera utilizada que não oferecia a capacidade para mensurar o ambiente em 3D, sendo necessário a presença de código de barras bidimensionais nos objetos e nos obstáculos do ambiente para serem reconhecidos, e;

- sistema de movimentação sem a força e precisão necessária para operar com base em métricas.

Para contornar as limitações do KBO, o XKBO adota então:

- tamanho físico suficiente para adotar processamento local e suportar inclusão de dispositivos periféricos;

- adaptação de um modelo de arquitetura estabelecida por norma internacional (ver adiante), diferente dos trabalhos relacionados os quais não tem por base uma norma;

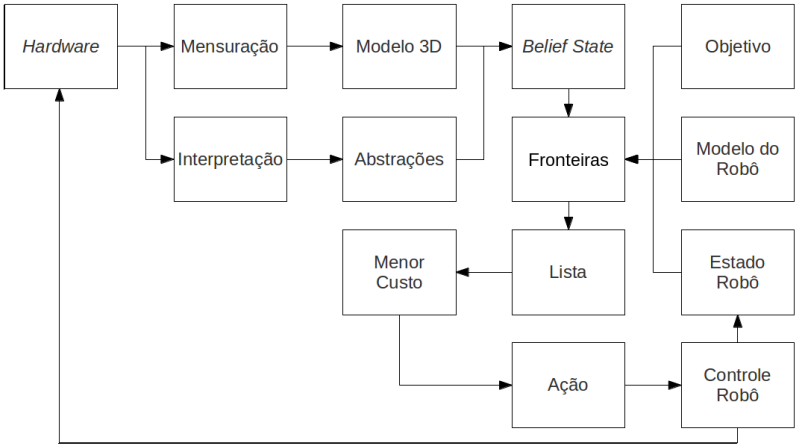
- tipo de câmera com capacidade de mensurar o ambiente em 3D, e;

- sistema de movimentação robusto e preciso, com controle de movimentação e velocidade, além de disponibilizar a informação de odometria.

4.1 MODELO DO PROCESSO DE BUSCA VISUAL

Baseado nos conceitos apresentados no capítulo 2, propõe-se a figura 16 para sintetizar o processo de uma busca visual ativa.

Figura 16 – Hipótese de um processo de busca ativa.



Fonte: produção do próprio autor

O processo segue os seguintes passos:

1. A informação é adquirida pelo *hardware* (câmera), a qual divide-se na mensuração, que possibilitará a reconstrução do modelo físico do ambiente, e na interpretação da imagem vista para obter abstrações sobre o que está presente na cena;

2. Com o processamento das informações contidas no modelo físico e na abstração da imagem chega-se ao *Belief State*, que significa a crença/conhecimento do robô em

relação ao estado completo ou parcial do ambiente explorado;

3. Este *belief state* é então processado junto com o objetivo (objeto de interesse), o modelo do robô (características físicas e de restrições) e estado do robô (posição e orientação) na função Fronteiras (detalhado adiante em 5.2.3.3), que é a função principal do algoritmo e tem como resultado uma lista dos pontos os quais o robô deverá se direcionar e realizar as novas aquisições com a câmera;

4. Esta lista gerada é então analisada e calcula-se qual o melhor ponto (menor custo e ou maior pontuação, detalhado adiante em 5.2.3.3) para o robô percorrer. Esta escolha leva à primeira ação de movimentação que é passada ao controle do robô para este então alterar o estado do robô e realizar uma nova aquisição iniciando novamente o ciclo do processo.

4.2 PLATAFORMA DO XKBO

A plataforma robótica do XKBO é baseada na norma NATO STANAG 4586 (NATO, 2005) que diz respeito a padronização para operação de sistemas aéreos não tripulados (*unmanned aerial vehicle* - UAV). Esta norma define as arquiteturas, interfaces, protocolos de comunicação, elementos de dados e formatos de mensagem relacionado para operar e gerenciar os UAVs em ambientes e operações de combate das forças combinadas da NATO.

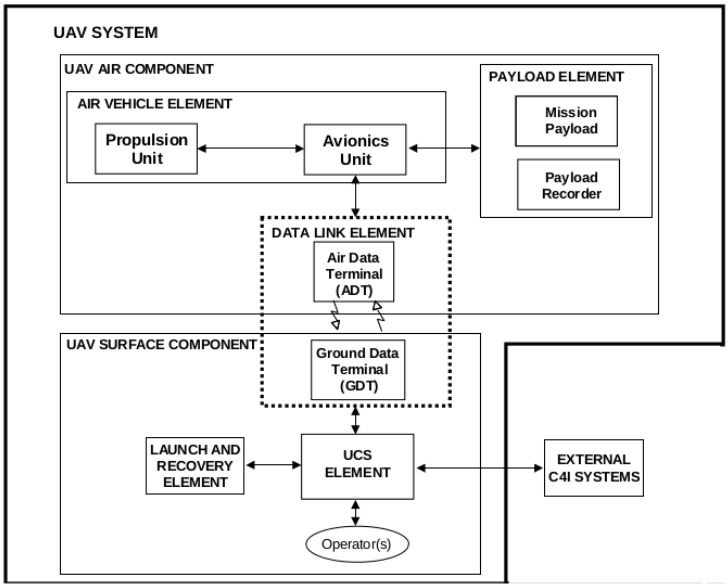
Embora a norma tenha como público alvo os UAVs, também chamados *drones*, sua estrutura pode ser utilizada em robôs terrestres por meio de adaptações, pois até o presente momento não existe uma norma específica para o uso em robôs terrestres. A norma mais próxima encontrada é a SAE AS5710 (SAE,

2008) porém, esta diz respeito apenas a questão da interoperabilidade.

Com relação aos benefícios em usar uma norma para o XKBO e em outros robôs móveis, estes consistem em: permitir a integração com outros sistemas; realizar a avaliação e comparação com outros robôs, e; realizar a futura comercialização do robô.

Conforme (NATO, 2005) a arquitetura de um sistema UAV (Figura 17) pode ser dividido em elementos onde:

Figura 17 – Elementos de um sistema UAV



Fonte: (NATO, 2005)

- O elemento de veículo aéreo (*air vehicle element*), consiste na estrutura, propulsão e os dispositivos necessári-os ao veículo aéreo e ao seu gerenciamento de vôo;
- O elemento de carga (*payload*), podem ser sistemas de sensores e dispositivos de gravação que estão instalados,

ou podem consistir de armamentos e mecanismos de controle / *feedback*, ou ambos;

- O elemento de dados (*data link element*), consiste na conexão do terminal de dados local do veículo e o terminal de dados remoto;

- O elemento de controle (*unmanned control system* – UCS), incorpora a funcionalidade de gerar, carregar e executar a missão do robô e disseminar produtos de dados de informação utilizável para vários sistemas C4I (comando, controle, comunicações, computadores e inteligência);

- O operador (*operator*), é responsável pela visualização do sistema e possui uma interface humano computador (*Human Computer Interface* - HCI);

- O elemento de lançamento e recuperação (*launch and recovery element*), incorpora a funcionalidade necessária para lançar e recuperar o veículo aéreo;

A proposta de adaptação deste sistema aéreo para o ambiente terrestre (e uso no robô XKBO) pode ser feita da seguinte maneira:

- O elemento de veículo aéreo é substituído pelo robô móvel terrestre, e consiste na estrutura, propulsão e os eletrônicos necessários ao robô e ao gerenciamento de movimentação. É representado pela estrutura física do robô móvel XKBO;

- O elemento de carga, passa a ser o conjunto de dispositivos periféricos (não usados no momento) como um braço robótico por exemplo;

- O elemento de dados passa a ser a rede *wireless*;

- O elemento de controle passa a ser executado pelo algoritmo de estratégia de busca, que decide e executa a missão de buscar por um objeto;
- O operador passa a ser a interface para o controle remoto e visualização dos dados da busca do XKBO;
- e o elemento de lançamento e recuperação do veículo, é substituído por um humano que coloca o robô na posição inicial ou, em caso de algum problema, pode resgatar o robô na sala.

4.3 ARQUITETURA CONCEITUAL DO XKBO

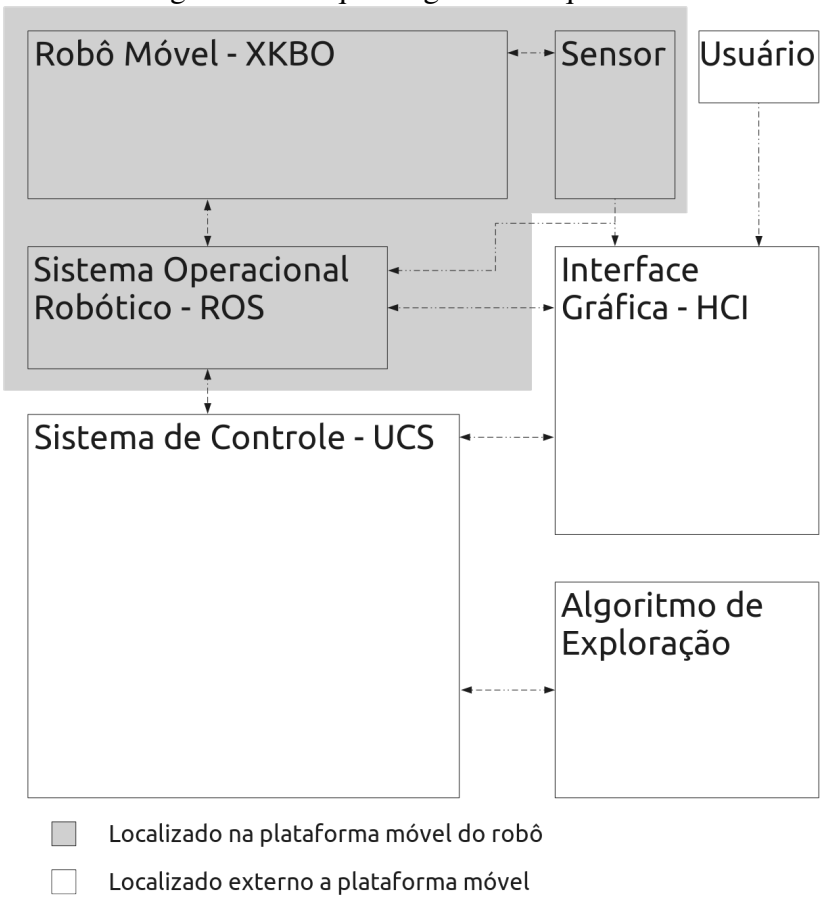
A união da arquitetura da STANAG 4586 com o modelo do processo de busca visual ativa, forma um modelo de arquitetura que interliga os diversos módulos constituintes e detalhado na Figura 18. Onde deve-se notar os módulos que ficam embarcados (com fundo cinza) e os que ficam remotos (sem fundo cinza). Observe a semelhança da Figura 18 com a arquitetura da Figura 17.

4.3.1 Bloco do robô móvel

O bloco do robô móvel (detalhado na Figura 19) é composto pela unidade de propulsão e a unidade do sistema do robô. Na unidade de propulsão tem-se: o microcontrolador (que recebe um comando pela USB e controla o *driver*); o *driver* (que energiza e aciona os motores), e; os motores (que transmitem força às rodas que movimentam o robô).

A unidade do sistema do robô é composta por 3 nós: controle do robô, onde é realizada a conversão entre as informações de velocidade providas do *Navigation Stack* do ROS para os va-

Figura 18 – Esquema geral da arquitetura.

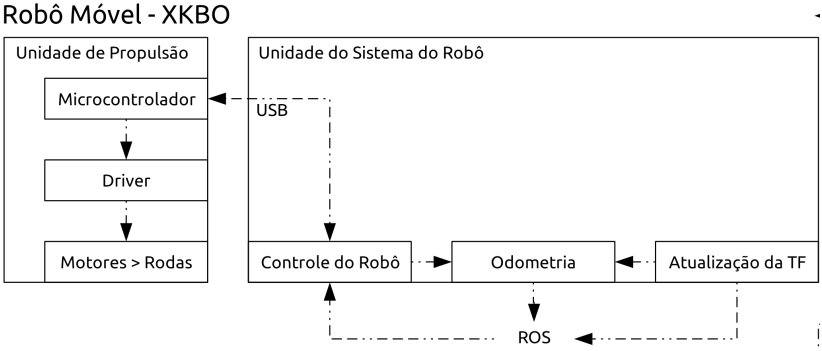


Fonte: produção do próprio autor

lores aceitos pelas características do robô e como resultado os novos valores de velocidades são repassados para a odometria, e também enviados na forma de um comando para o microcontrolador; a publicação da odometria, que é responsável por atualizar as informações de posição e orientação do robô, assim como sua velocidade para o sistema; e a atualização das transforma-

das geométricas, que sincroniza as informações da coordenada da câmera com a odometria do robô.

Figura 19 – Esquema do bloco Robô.



Fonte: produção do próprio autor

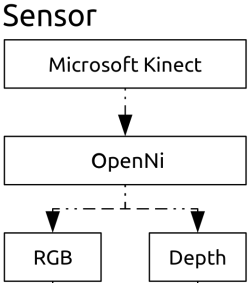
4.3.2 Bloco do sensor

O bloco do sensor (detalhado na Figura 20) é composto pela câmera Kinect e seu *driver* OpenNI. Sua principal função é prover ao sistema a imagem RGB e a imagem de profundidade (*Depth*).

4.3.3 Bloco do ROS

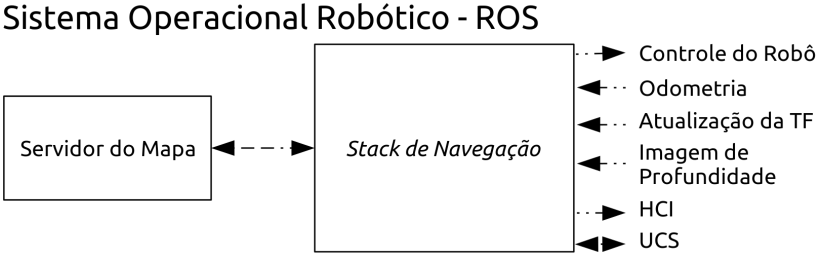
O bloco do ROS (detalhado na Figura 21) é composto pelo sistema operacional ROS e tem por objetivo armazenar as informações do mapa (através do *map server*), e executar o módulo do *Navigation Stack* que faz a interface do planejamento de movimento do robô com o mapa.

Figura 20 – Esquema do bloco do sensor.



Fonte: produção do próprio autor

Figura 21 – Esquema do bloco do ROS.



Fonte: produção do próprio autor

4.3.4 Bloco do sistema de controle

O bloco do sistema de controle (detalhado na Figura 22) é o bloco que representa o núcleo do sistema, e é responsável por integrar os dados de todas os blocos provendo autonomia ao robô.

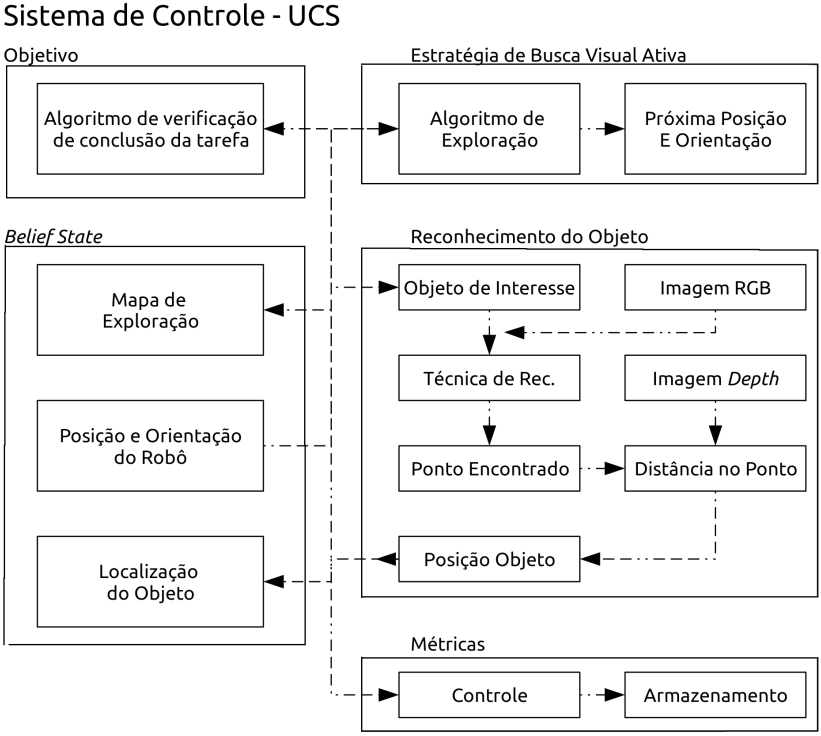
Suas partes componentes são:

- O objetivo, responsável por verificar a conclusão da tarefa;
- A estratégia de exploração, que realiza a exploração e encontra a próxima posição e orientação para o robô se mo-

vimentar;

- Reconhecimento de objeto, realiza a análise da imagem para o conhecimento da presença do objeto na cena.
- *Belief State*, onde é armazenada as informações do conhecimento do robô.
- e as métricas, onde as informações das métricas estabelecidas são processadas.

Figura 22 – Esquema do bloco do sistema de controle.

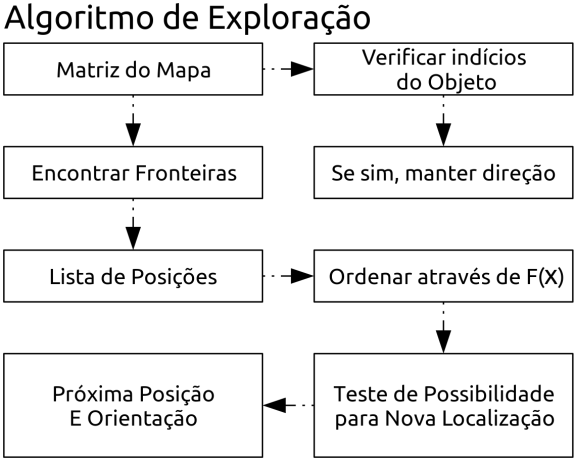


Fonte: produção do próprio autor

4.3.4.1 Algoritmo de exploração

O bloco de exploração (detalhado na Figura 23) contém o algoritmo responsável pela exploração do robô no mapa. Seu processo consiste em: adquirir a matriz de ocupação do mapa e verificar se existe a presença de indícios do objeto no campo de visão do robô, caso afirmativo mantêm movimentação em direção ao provável objeto; caso negativo procede para encontrar os pontos de fronteiras e gera uma lista de posições; essa lista de posições é ordenada baseada na pontuação obtida através da função F(X) adotada, e; por fim é escolhida a primeira posição de localização (possível fisicamente) para o robô se movimentar.

Figura 23 – Esquema bloco Robo.



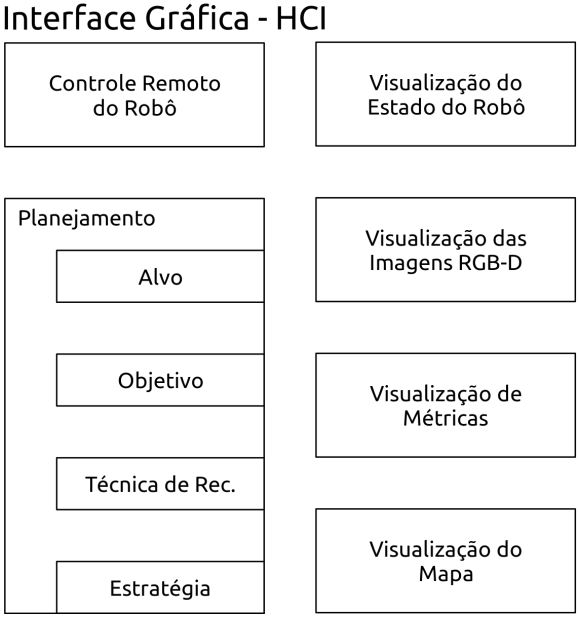
Fonte: produção do próprio autor

4.3.5 Bloco da interface gráfica

O bloco da interface gráfica (detalhado na Figura 24) tem por objetivo permitir a um operador (usuário em um computador

remoto) acesso às funções do sistema. Suas partes componentes são:

Figura 24 – Esquema do bloco de interface gráfica.



Fonte: produção do próprio autor

- O controle remoto do robô, onde é possível operar remotamente a movimentação do robô;
- A visualização do estado do robô, que provê informações relativas ao robô como posição, orientação, velocidade, alarmes, situação da conexão *wireless*, a ação que está sendo realizada no momento e a localização do ponto de destino ao qual o robô se direciona;
- A visualização das imagens, que permite visualizar as imagens que estão sendo capturadas pela câmera do Kinect;

- A visualização das métricas, que mostra as métricas do sistema como o tempo, número de ações realizadas e distância percorrida;
- A visualização do mapa, que representa a informação visual do mapa métrico;
- E o planejamento, que é onde o operador insere as informações relativas: ao alvo (imagem do objeto de interesse); ao objetivo (se busca ou exploração); a técnica de reconhecimento de objetos (algoritmo SURF); e a estratégia de busca que será utilizada (algoritmo de estratégia deste trabalho, outros algoritmos de estratégia).

4.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capítulo foi definido a proposta do trabalho, onde se propôs um modelo para o processo de uma busca visual ativa, na qual é composta pelos blocos descritos anteriormente e terá sua arquitetura adaptada da norma internacional STANAG 4586. No próximo Capítulo será visto como foi executada a implementação da proposta deste trabalho.

5 IMPLEMENTAÇÃO

A implementação foi realizada com a construção de um robô móvel com as seguintes definições presentes na Tabela 2:

Tabela 2 – Definições para a implementação

Sistema operacional:	Lubuntu 13.04
Distribuição do ROS:	Hydro
IDE para interface gráfica:	QT Creator 5.2
Versão do Kinect:	Kinect 360 for Windows
Versão do OpenNI:	2.0
Dimensões físicas do Robô:	40 x 50 x 40 cm
Locomoção:	Não holonômico
Cinemática:	Diferencial
Tipo de rodas:	2 tracionadas independentemente, 1 pivô livre
Dimensões físicas do Objeto:	15 x 15 x 5 cm
Técnica de Reconhecimento:	SURF

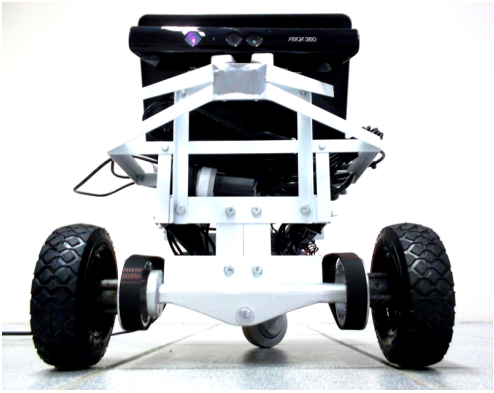
Fonte: produção do próprio autor

5.1 HARDWARE

O hardware é representado pela estrutura física do robô, vista nas Figuras 25 e 26, e observe que o Kinect está na parte superior do robô e posicionado no centro sob o eixo das rodas porque desta forma reduz a complexidade de transformadas geométricas durante movimento. A sua dinâmica de movimentação (representada na Figura 27) é composta por dois tipos de movimentos: deslocamento, onde se movimenta linearmente para

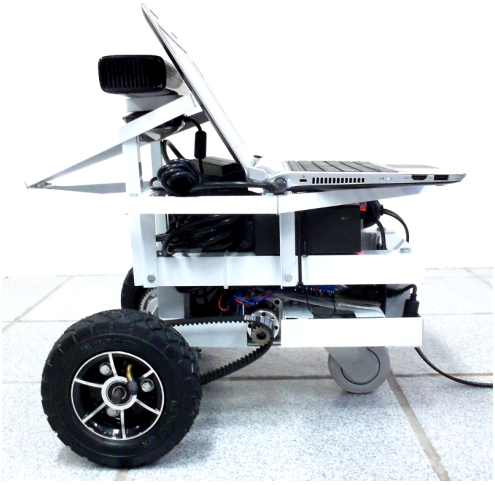
frente ou para trás; e rotação, em torno do eixo ortogonal ao plano do solo.

Figura 25 – Estrutura do Robô XKBO - vista frontal



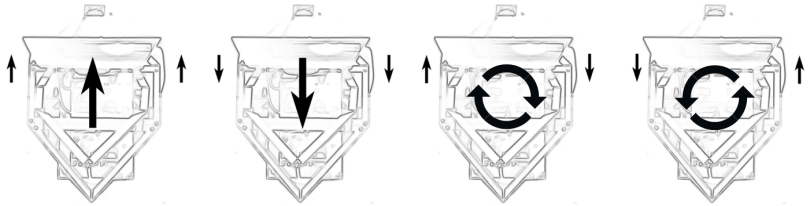
Fonte: produção do próprio autor

Figura 26 – Estrutura do Robô XKBO - vista lateral



Fonte: produção do próprio autor

Figura 27 – Dinâmica de movimento



Fonte: produção do próprio autor

Observe que este tipo de movimentação facilita o controle do robô, pois ambos os motores são acionados simultaneamente e mantêm-se em sincronia. Embora seja possível controlar os motores de forma assíncrona (com velocidades diferentes) para executar um movimento na forma de arco, neste trabalho isto não será implementado devido as limitações de processamento impostas pelo microcontrolador (ver Tabela 4).

5.1.1 Unidade de processamento

Para realizar o processamento foi utilizado um *netbook* fixado na estrutura do robô. As suas características são descritas na Tabela 3.

5.1.2 Driver

Para controlar o *driver* do motor e fazer a interface com o computador, foi utilizado a plataforma Arduino UNO R3 que é um placa baseada no microcontrolador ATmega328. O Arduino contém todos os componentes necessários para suportar o microcontrolador como oscilador, comunicação serial, regulador

Tabela 3 – Dados da unidade de processamento

Modelo:	DM1-3255BR
Fabricante:	HP
Processador:	Dual Core E-350 1,6 GHz
Memória:	6 GB
Placa de video:	AMD Radeon HD 6310
Sistema Operacional:	Lubuntu 13.04 (Raring Ringtail)

Fonte: produção do próprio autor

de tensão, leitor ICSP, e botão de *reset*. Pode ser alimentado pelo cabo USB, por uma fonte ou bateria. As características da placa se encontram na Tabela 4.

Tabela 4 – Dados do controlador.

Modelo:	Arduino Uno R3
Fabricante:	Arduino CC
Microcontrolador:	ATmega328
Tensão de operação:	5V
Pinos:	14 (6 PWM) I/O digitais e 6 analógico
Memória:	Flash 32 KB, SRAM 2 KB
Velocidade do Clock:	16 MHz
Comunicação:	USB
Dimensões:	6.9 x 5.3 x 1.8 cm

Fonte: produção do próprio autor

A placa do *driver* responsável por energizar os motores foi projetada e elaborada pelo próprio autor para suportar dois motores de passo e tem as suas características descritas na Tabela 5 e representação física na Figura 28. Observe que existe

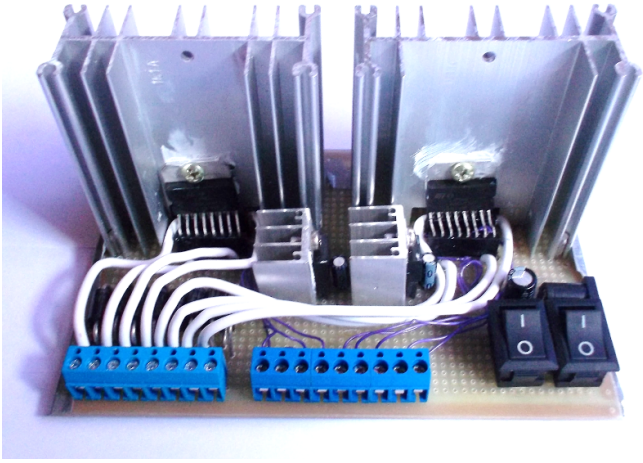
a presença de dois dissipadores com *coolers* na placa, devido ao fato da corrente utilizada ser elevada. A alimentação desta placa é provida por uma fonte externa que converte a tensão de 220 V em corrente alternada para a tensão de 12 V em corrente contínua e com capacidade de saída de até 4 A.

Tabela 5 – Dados do *Driver*.

Chave:	L298N (2x)
Descrição:	*Dual Full-Bridge Driver*
Tensão de operação:	12 V
Corrente máxima por motor:	2 A
Dimensões:	15 x 10 x 7 cm

Fonte: produção do próprio autor

Figura 28 – *Driver* para motor



Fonte: produção do próprio autor

5.1.3 Atuador

Como atuador são utilizados dois motores de passo (para controle angular preciso), um para cada roda, com as características apresentadas na Tabela 6 e ilustrado pela Figura 29.

Tabela 6 – Dados do motor

Modelo:	36JX30K100G / 42STH38-1684A
Fabricante:	DONGZHENG MOTOR CO. LTD.
Tipo:	Bipolar
Ângulo do passo:	1.8°
Torque:	48 kg-cm
Rotação máxima:	44 RPM
Tensão:	2.8V - 12 V
Corrente:	1.7 A
Formato:	NEMA-17 Bipolar 4-wire

Fonte: produção do próprio autor

Figura 29 – Motor de passo

Fonte: produção do próprio autor

5.1.4 Rodas

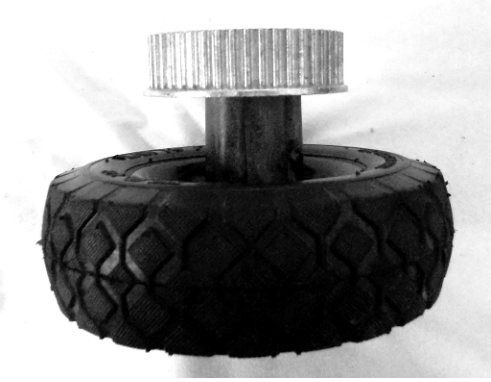
Para as rodas utilizou-se um modelo de rodas tracionadas com correia dentada, suas características se encontram na Tabela 7 e pode ser vista na Figura 30.

Tabela 7 – Dados da roda

Modelo:	Roda Tração Completa Skate Elétrico Off Road 700w
Fabricante:	TwoDogs
Câmera:	Sim, a Ar (22 psi)
Dimensões da Roda:	
Aro:	15 cm
Largura:	5 cm

Fonte: produção do próprio autor

Figura 30 – Roda



Fonte: produção do próprio autor

5.1.5 Custos

O custo dos componentes listados para a construção do *hardware* pode ser encontrado na Tabela 8.

Tabela 8 – Lista de custos dos componentes usados no *hardware*

		Valor USD*	Valor BRL
Chassis			**92,00**
	Alumínio		92,00
Atuador			**382,17**
	Rodas		176,00
	Pinhões		48,88
	Correias		66,00
	Eixo Central		91,29
Força		**110,28**	**229,68**
	Motores de Passo	87,30	
	Fonte 12V	11,49	
Eletrônicos		**34,20**	**71,82**
	Microcontrolador	23,70	
	Driver do Motor	10,50	
		Hardware	**775,67**
Kinect			360,00
Notebook			1100,00
		Total XKBO	**2235,64**

*Os valores de conversão de dólar para real foram realizados com base na data de compra

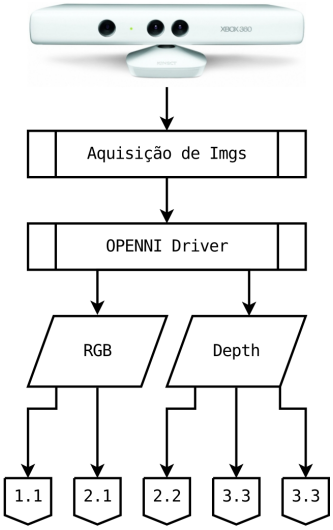
Fonte: produção do próprio autor

5.2 SOFTWARE

5.2.1 Imagens

A aquisição e publicação das imagens é realizada conforme o diagrama da Figura 31, onde pode-se observar a sequência do Kinect para o *driver* OpenNI, e deste, separado em imagem RGB e Depth, as quais são publicadas para o sistema. Os tópicos que publicam estas mensagens são o "/camera/rgb/image" para imagem RGB e o "/camera/depth/image" para a imagem de profundidade. Ambos têm como tipo de formato o "sensor_msgs/Image", que é uma variável de transporte de imagem do ROS. O nó responsável por publicar estas mensagem é o "/camera/camera_nodelet_manager". Os nós que recebem estas mensagens são o "/xkbo_UCS" e o "/xkbo_Interface".

Figura 31 – Diagrama do módulo aquisição da câmera

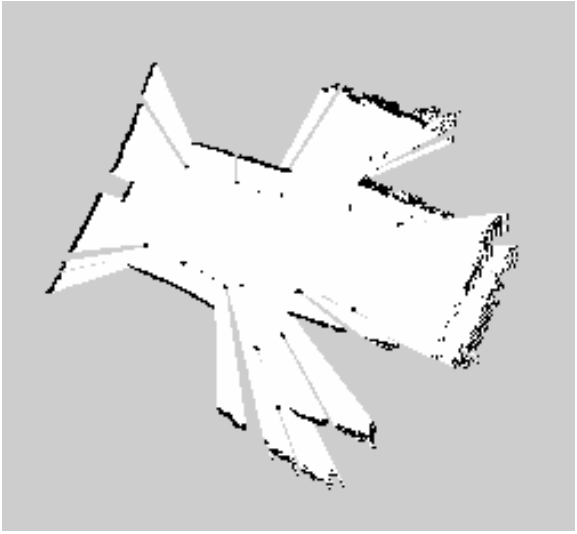


Fonte: produção do próprio autor

5.2.2 Mapa

O mapa métrico de exploração do ambiente é um corte plano 2D de um espaço 3D, e é publicado pelo */map_server*, que é gerado por um filtro de particulas Rao-Blackwellized (GRISETTI; STACHNISS; BURGARD, 2007) que consiste em uma representação de uma matriz de células 2D, onde cada célula contém o valor de ocupação sendo: branco como célula livre; o preto como célula ocupada, e; cinza como célula desconhecida. A Figura 32 representa um exemplo de mapa[1] obtido a partir da rotação do robô colocado no centro do ambiente de teste (ver Figura 40, adiante).

Figura 32 – Exemplo de mapa



Fonte: produção do próprio autor

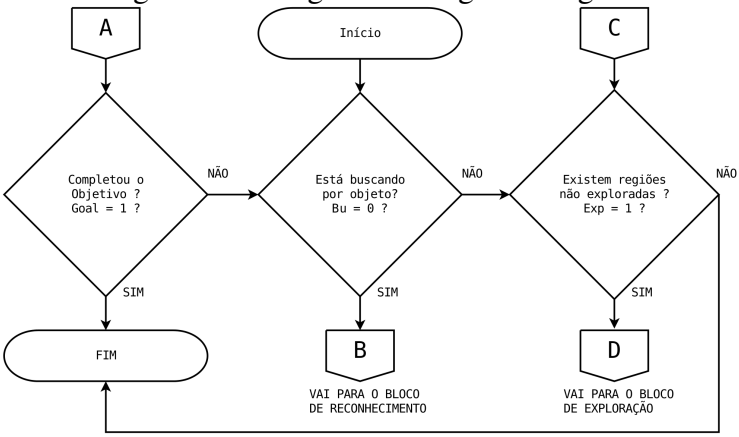
[1] a imagem apresentada foi recortada e centralizada, pois a imagem original do arquivo é consideravelmente maior que a sala.

5.2.3 Busca Ativa

5.2.3.1 Algoritmo geral

O processo de busca inicia-se com o módulo geral (diagrama representado na Figura 33). A primeira tomada de decisão é pelo objetivo: se está buscando pelo objeto ou simplesmente explorando um ambiente. Caso esteja buscando o objeto deve-se seguir para o módulo de reconhecimento (B). E, se não estiver buscando por um objeto deve seguir ao processo seguinte que questiona se existem regiões ainda não exploradas (definido pelo módulo de exploração), em caso afirmativo deve se seguir para o módulo de exploração (D), caso contrário, encerra-se a missão. Tanto o módulo de reconhecimento quanto o de exploração tem em suas saídas o conector (A) e que segue para o processo no qual questiona-se se o objetivo foi concluído, em caso afirmativo encerra-se a missão, e se for negativo, fecha-se o ciclo e continua o algoritmo.

Figura 33 – Diagrama do algoritmo: geral



Fonte: produção do próprio autor

5.2.3.2 Algoritmo de reconhecimento

O módulo seguinte (Figura 34) é referente ao reconhecimento e opera da seguinte maneira:

- Recebe-se a imagem RGB da câmera (provida pelo Kinect) na qual é processado um algoritmo de reconhecimento de objetos (neste trabalho foi usado o SURF), que compara as características extraídas da imagem com a do objeto, armazenado previamente pelo usuário;

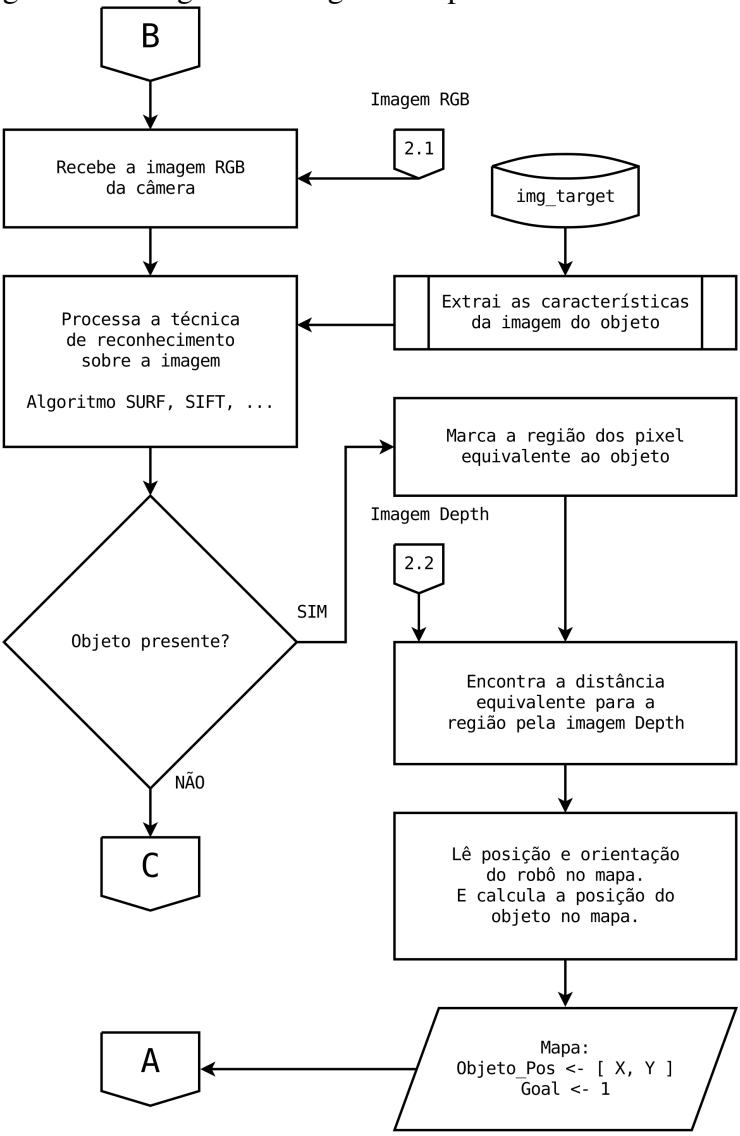
- A saída do algoritmo de reconhecimento tem como valor a presença ou não do objeto a qual é questionada, e em caso negativo, volta para o módulo geral;

- No caso afirmativo, ou seja, o objeto se encontra presente, a região dos *pixels* equivalente ao objeto na imagem RGB é marcada. Esta marcação é utilizada para uma sobreposição com a imagem de profundidade (provida pelo Kinect), onde torna-se possível encontrar a distância equivalente que o objeto se encontra na frente do robô;

- Com a posição e orientação do robô é possível usar este valor de distância e marcar a posição do objeto no mapa;

- Para encerrar o módulo, a variável de objetivo é marcada como concluída e vai ao módulo geral, que recebe esta variável de conclusão e encerra a busca.

Figura 34 – Diagrama do algoritmo: parte de Reconhecimento



Fonte: produção do próprio autor

5.2.3.3 Algoritmo de exploração ativa

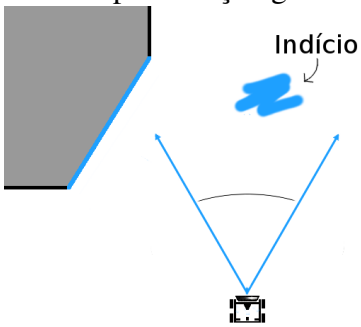
O módulo de exploração (Figura 36) opera da seguinte forma:

- Inicia-se recebendo a imagem de profundidade (provida pelo Kinect) e converte-se os valores dos *pixels* para distância;

- Estes valores de distância são utilizados para atualizar (sobrescrever os *pixels* de estado desconhecidos) o mapa do ambiente com informações de estado como livre ou ocupado;

- Após esta atualização, verifica-se se há um indício da característica do objeto (neste caso cor definida) no campo de visão do robô (exemplificado na Figura 35), caso afirmativo o robô mantêm sua movimentação à frente até ser possível confirmar ou descartar a presença do objeto para aquele indício;

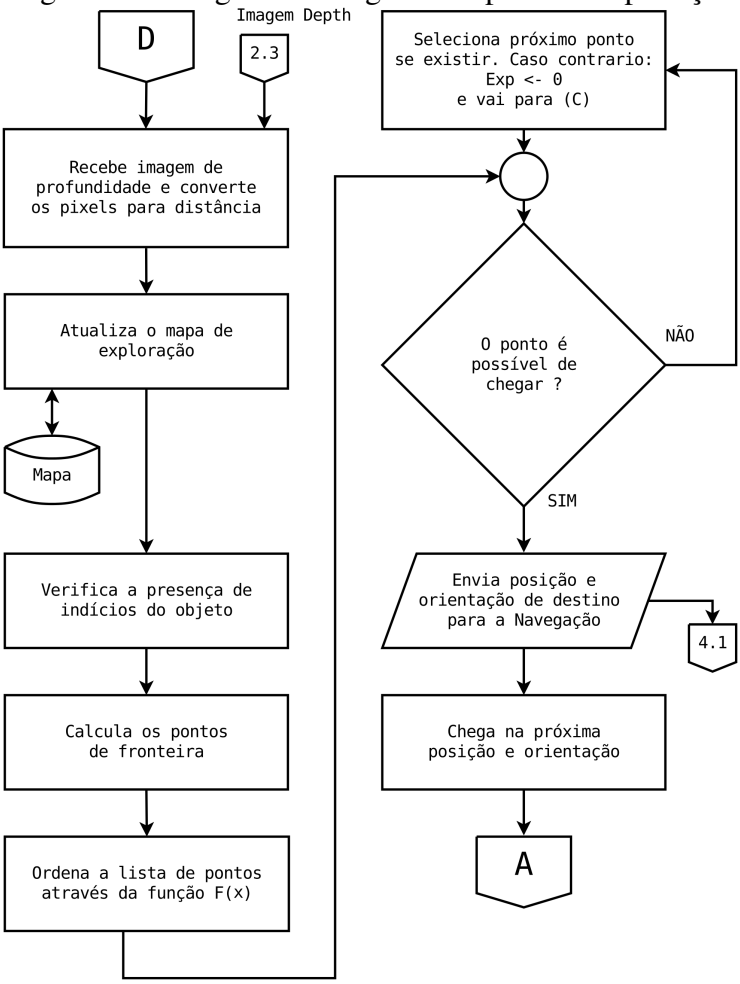
Figura 35 – Representação gráfica do indício.



Fonte: produção do próprio autor

- Caso negativo (não há indício no campo de visão), é calculada uma lista com os pontos de fronteira e com base na função F(X) ordena-se a lista em ordem de maior score (este processo será descrito posteriormente no texto).

Figura 36 – Diagrama do algoritmo: parte de exploração



Fonte: produção do próprio autor

- O processo seguinte é analisar se o primeiro ponto da lista é possível fisicamente, dadas as limitações físicas do robô. Em caso afirmativo, envia-se esta posição de destino para a navegação (Anexo 1). Em caso negativo, seleciona-se o próximo ponto da lista enquanto existir. Se a lista terminar sem um ponto possível, determina-se que a exploração esteja concluída e volta-se para o módulo geral;

- Para encerrar o módulo espera-se o robô se mover para a nova posição e do módulo geral fechando o ciclo e dando prosseguimento à busca.

Algoritmo de busca pelas fronteiras

Para desenvolver uma estratégia de busca que faz uso das informações oriundas da câmera de profundidade, idealizou-se uma solução que levasse em conta a amplitude da próxima área a ser explorada. Como o robô não conhece a extensão do ambiente, o único indicativo da amplitude do espaço ainda desconhecido é o tamanho da fronteira a estas áreas. Por isso, propôs-se uma nova estratégia denominada busca pelas fronteiras. O processo que calcula os pontos de fronteira e ordena por proximidade tem a seguinte operação:

1. Recebe-se o mapa métrico do servidor (Figura 37-A) equivalente ao estado inicial e discretizado em uma imagem de 500x500 *pixels* (configurável);

2. Da imagem do estado inicial marcam-se as regiões de fronteiras que são os *pixels* brancos que estão em contato com os *pixels* cinza (Figura 37-B);

3. Separam-se estas regiões em uma nova imagem (Figura 37-C);

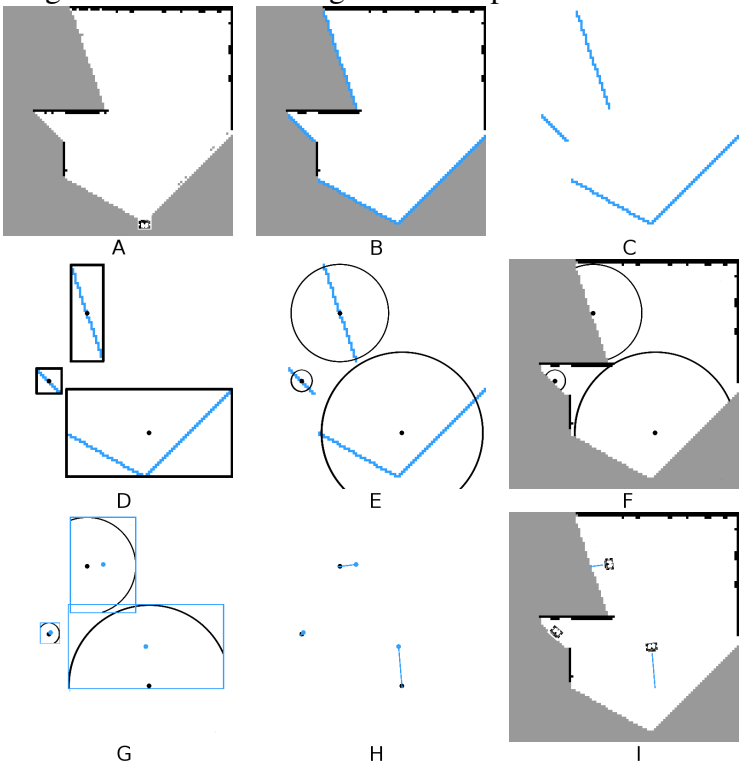
4. Demarcam-se os envoltórios para estas regiões (Figura 37-D);

5. Com base na maior dimensão (largura ou altura) do envoltório gera-se um círculo de mesmo diâmetro com o centro na mesma posição do centro do envoltório (Figura 37-E);

6. Subtrai-se destes círculos a parte equivalente à região desconhecida do mapa inicial (cinza) (Figura 37-F);

7. Demarcam-se os envoltórios para a parte restante dos círculos e encontram-se o centro destes envoltórios (Figura 37-G);

Figura 37 – Passos do algoritmo dos pontos de fronteira



Fonte: produção do próprio autor

8. Encontram-se os ângulos entre a posição do centro destes envoltórios com o centro do envoltório das regiões de fronteira (Figura 37-H);

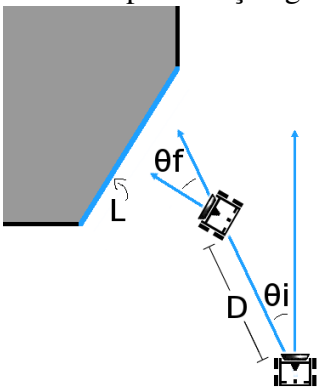
9. Definem-se as próximas posições e orientações com base nos pontos e ângulos (Figura 37-I);

Algoritmo para priorizar da lista de próximas posições e orientações

Para priorizar o melhor "próximo ponto" a ser visitado pelo robô é calculado, para cada um dos pontos de fronteira, o seu *score* através da função F(X) e seleciona-se o ponto com o maior *score*.

O cálculo da função F(X) leva como parâmetros (detalhado na Figura 38) os valores nos pontos de fronteira relativos a:

Figura 38 – Representação gráfica do parâmetros.



Fonte: produção do próprio autor

- Distância (D), que é a distância entre o robô e o próximo ponto candidato;

- Ângulo inicial (θ_i), que é o ângulo entre a orientação atual do robô e o ponto;
- Ângulo final (θ_f), que é a diferença entre o ângulo atual e o ângulo de orientação final para o ponto, e;
- Tamanho da linha (L), que é o número de *pixels* que compõe a região de fronteira a ser explorada (região oculta) para o ponto, e que tem um valor máximo limitado pelo campo de visão do robô;

Estes parâmetros então compõem um vetor P dado por:

$$P(n) = [D, \theta_i, \theta_f, L]$$

Onde: $D = D_{max} - D$, pois a distância deve ter um comportamento proporcional inverso, ou seja quanto maior a distância menor a pontuação. E, de maneira semelhante para os ângulos $\theta_i = \pi - \theta_i$, e $\theta_f = \pi - \theta_f$. No caso do L mantêm-se inalterado pois tem o comportamento da pontuação diretamente proporcional.

Cada parâmetro é ponderado em função da estratégia de exploração adotada por um vetor Q dado por:

$$Q(n) = [a, b, c, d]$$

No caso deste trabalho os valores (empíricos) encontrados foram: $a = 1.5, b = 4.5, c = 0.5, d = 1$. E considerou-se $D_{max} = 25m$ que é o comprimento máximo do mapa utilizado (500 pixel $\times$ 0,05 metros/pixel = 25 metros).

Portanto a função F(X) pode ser equacionada como o somatório dos valores ponderados dos parâmetros para cada ponto:

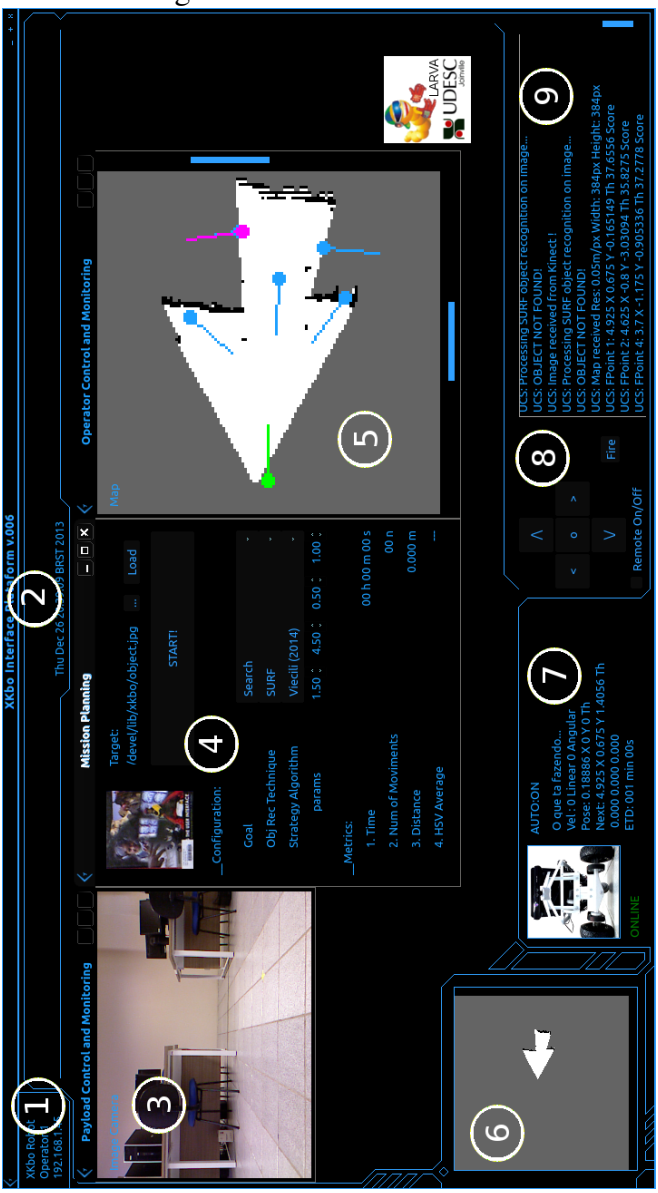
$$F(X) = \sum_{n=1}^{4} P(n) \times Q(n) \qquad \text{Equação 1}$$

5.2.4 Interface humano computador (IHC)

A interface gráfica para visualização foi desenvolvida no QT Creator 5.2 e atende a todos os requisitos encontrados na seção 4 da proposta. Sua representação está na Figura 39, onde:

1. Info, mostra as informações do nome do robô, do usuário e endereço de IP do robô;
2. Data, mostra as informações de dia e horário;
3. Controle e visualização do sensor, onde são reproduzidas as imagens do Kinect;
4. Planejamento da missão, mostra as informações relativas ao planejamento da busca e métricas;
5. Mapa de exploração, representa visualmente o mapa métrico;
6. Mini mapa, é a informação visual do mapa reduzida e com a posição do robô nele. Isto é devido ao fato de que o mapa completo tem dimensões extensas para comportar a construção do mapa a apartir do ponto central, ou seja, no pior dos casos a representação de ambiente inteiro explorado pode estar em um único quadrante do mapa;
7. Situação do robô, contém as informações de situação do robô, como que tarefa executa no momento, a posição (X, Y), a orientação (Theta), as velocidade linear e angular, o destino, etc;
8. Controle remoto do robô, permite a operação de movimentação pelos botões, e;
9. Mensagens, visualização da troca de mensagens do sistema;

Figura 39 – Interface Gráfica



Fonte: produção do próprio autor

5.3 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Neste capítulo foi documentado a implementação do trabalho e suas respectivas características para a construção do robô (*hardware*) e algoritmos (*software*). O próximo Capítulo envolvera os resultados e discussão do trabalho.

6 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados: as métricas e o ambiente de testes a ser utilizado para os experimentos de validação do trabalho; os resultados dos experimentos; e por fim uma análise e discussão dos resultados.

6.1 DEFINIÇÃO DAS MÉTRICAS

As métricas a serem utilizadas nos experimentos de busca são:

- Tempo decorrido até encerrar a missão – T (segundos);
- Número de ações (movimentações ou rotações executadas) – A (unitário);
- Distância percorrida durante a execução – D (centímetros);
- Sucesso – G (binário).

6.2 CARACTERÍSTICAS DOS EXPERIMENTOS E CENÁRIO

Os experimentos realizados foram:

- Para validar a capacidade de reconhecimento de objetos colocou-se na frente da câmera objetos em diferentes ângulos de rotação e foi medido, de qual distância foi possível reconhecer os objetos;
- Para mensurar a quantidade de área explorada por unidade de tempo imposta pelas limitações físicas do robô, movimentou-se o robô em percursos fechados e, então foi

calculada a razão dos metros quadrados explorados pelo tempo decorrido;

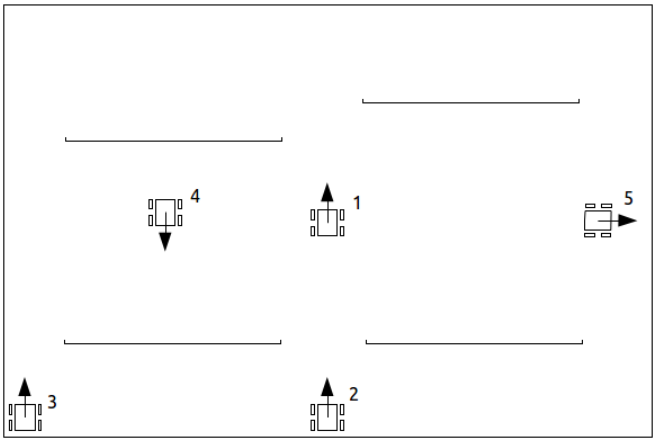
- Para validar a proposta na tarefa de busca visual ativa e as capacidades do robô móvel no reconhecimento do objeto, exploração, localização, tomada de decisão e desvio de obstáculos foi realizado um experimento numa sala (Figura 40) onde o robô móvel foi colocado em uma posição no ambiente (números 1 a 5 na Figura 41), porém esta posição não foi declarada para o robô. Então, o *framework* de busca visual ativa pelo objeto de interesse foi inicializado, e o robô móvel iniciou a exploração pelo ambiente desconhecido com base apenas nas informações coletadas pelo Kinect.

Como dados coletados foram obtidos os valores definidos nas métricas e um mapa do ambiente explorado.

Figura 40 – Local de realização dos experimentos.

Fonte: produção do próprio autor

Figura 41 – Pontos de partida do robô móvel nos experimentos.



Fonte: produção do próprio autor

6.3 RESULTADOS DOS TESTES

Os experimentos da capacidade de reconhecimento de objetos utilizaram dois objetos (objeto 1 visto na Figura 42-A e objeto 2 na Figura 42-B).

Figura 42 – Imagem dos objetos utilizados nos experimentos.

A B

Fonte: produção do próprio autor

Encontram-se na Tabela 9 as distâncias máxima que o robô foi capaz de reconhecer o objeto para determinado ângulo

de rotação do objeto usando a técnica de reconhecimento SURF sendo:

- (A) objeto 1 com imagem de referência do objeto obtida a 1 metro de distância;
- (B) objeto 1 com imagem de referência do objeto obtida a 0,6 metro de distância;
- (C) objeto 2 com imagem de referência do objeto obtida a 1 metro de distância;
- (D) objeto 2 com imagem de referência do objeto obtida a 0,6 metro de distância;

Observou-se no experimento que a máxima distância possível foi de 174 cm em 0° na situação C, enquanto que a distância mínima para reconhecer foi de 48 cm em 60° na situação B.

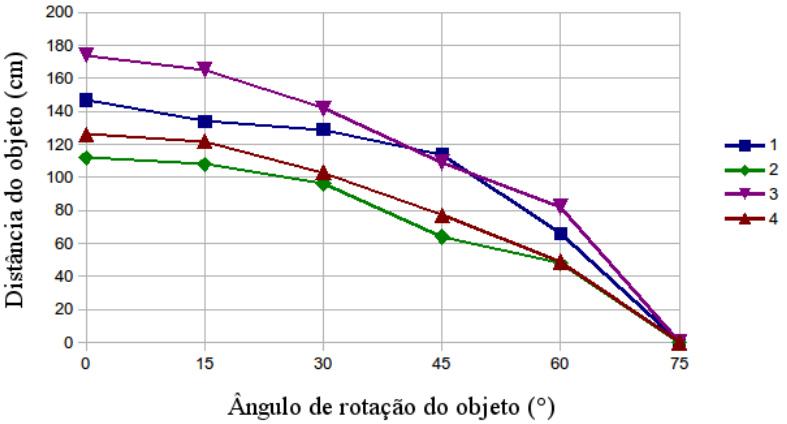
Tabela 9 – Resultado dos testes de reconhecimento

Ângulo de rotação do objeto	Distância do objeto (cm)			
	(A)	(B)	(C)	(D)
0°	147	112	174	126
15°	134	108	165	122
30°	129	96	142	103
45°	114	64	109	77
60°	66	48	82	49
75°	-	-	-	-

Fonte: produção do próprio autor

O gráfico da Figura 43 representa um comparativo entre os resultados para as distâncias máximas de reconhecimento de objeto em relação ao ângulo de rotação.

Figura 43 – Gráfico de reconhecimento: ângulo vs distância.



Fonte: produção do próprio autor

Os resultados para o experimento da tarefa de busca realizados pelo robô se encontram na Tabela 10.

Tabela 10 – Resultado experimento: robô

Posição	Tempo	N Ações
1	2139	6
2	809	6
3	537	3
4	1263	5

Fonte: produção do próprio autor

Posição 1

O procedimento que o robô adotou na exploração realizada partindo da posição 1 (vista na Figura 44) pode ser descrito da seguinte maneira:

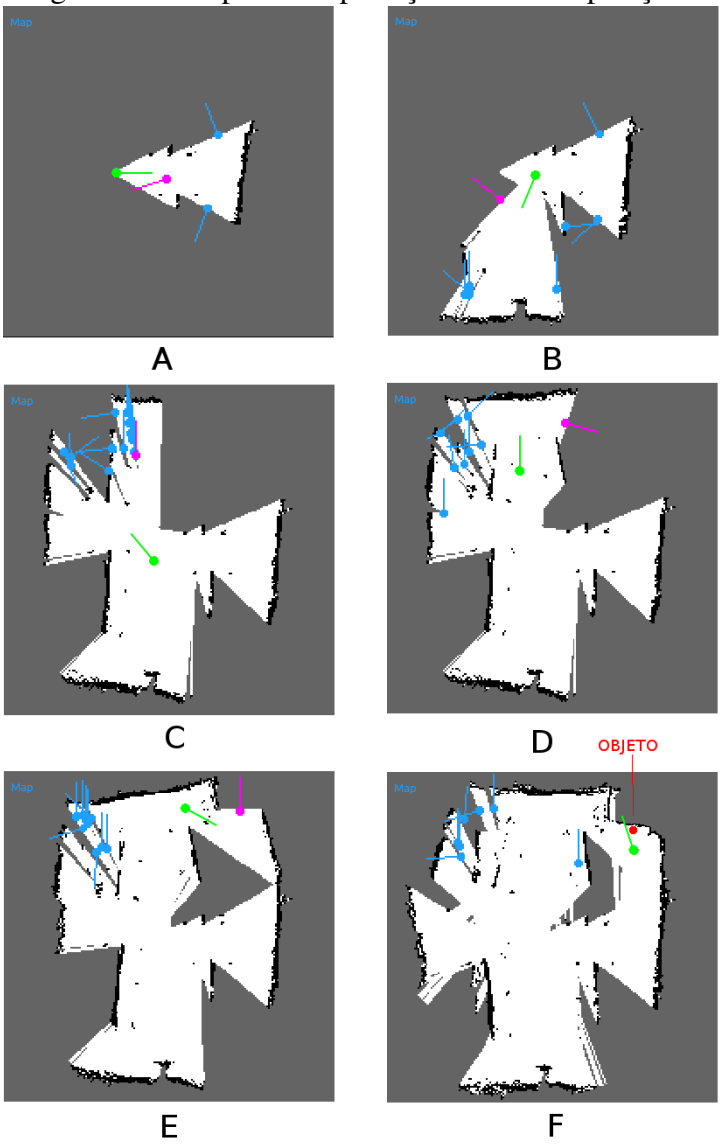
Figura 44 – Localização inicial do robô saindo da posição 1

Fonte: produção do próprio autor

Na posição inicial (Figura 45-A), onde: o robô está representado pelo ponto verde; os possíveis candidatos para a nova localização do robô em azul, e; a próxima localização escolhida através da maior pontuação (*score*) obtida pela função F(X) em magenta. O robô desloca-se para a nova localização (Figura 45-B) e realiza novo calculo de candidatos e seleciona o próximo com melhor *score*.

O robô desloca-se para a nova posição (Figura 45-C). Observe que existe uma leve diferença na localização escolhida anteriormente e a atual em que se encontra o robô. Isto se deve ao fato de que escolheu-se deixar uma margem de erro aceitável (em X, Y e θ) para evitar que o robô fique tentando se posicionar exatamente sobre o ponto (perdia muito tempo e não conseguia

Figura 45 – Etapas da exploração saindo da posição 1



Fonte: produção do próprio autor

dado as limitações físicas de menor deslocamento), e observou-se empiricamente que esta diferença não altera o resultado.

A exploração prossegue (Figura 45 D, E e F) de forma semelhante selecionando o próximo ponto com maior *score* e se deslocando até o mesmo.

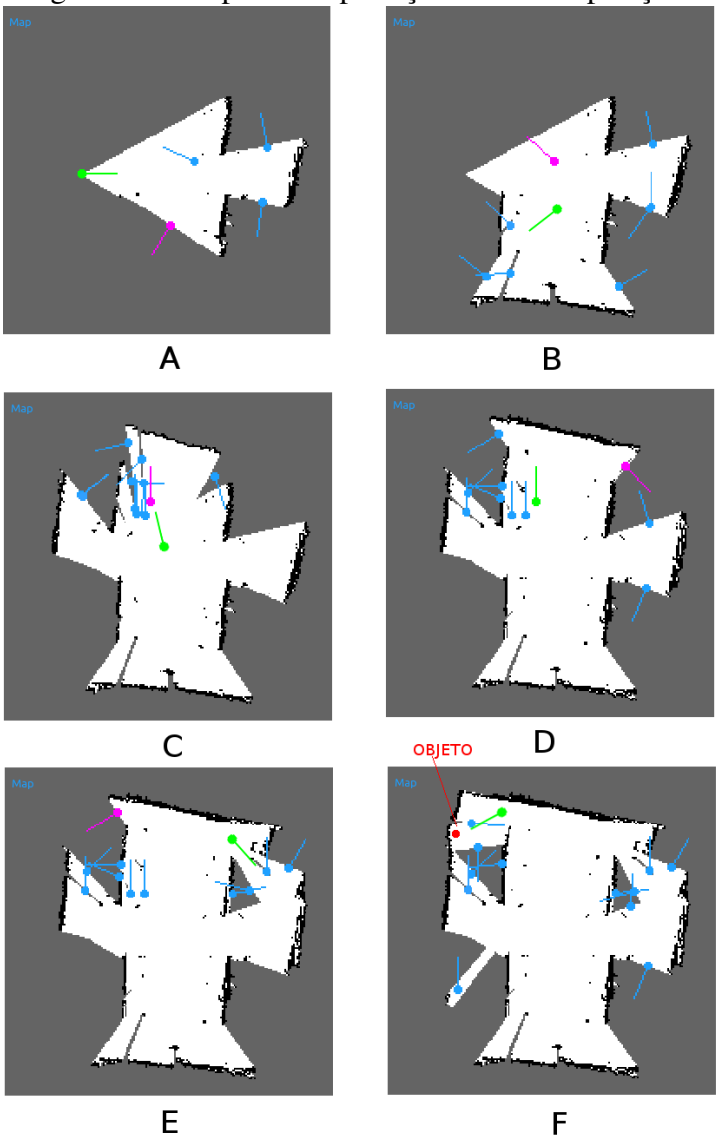
Posição 2

Na exploração realizada partindo da posição 2 (vista na Figura 46) que está localizada um pouco atrás da posição 1, observe que já existe uma diferença na escolha da primeira próxima posição (Figura 47-A). Embora os próximos passos (Figura 47-B, C e D) sigam parecidos ,existe uma diferença no último passo (Figura 47-E e F) o que demonstra a tendência para os locais com maiores regiões ocultas. No entanto, note que há semelhança entre a Figura 45-F e Figura 47-F, afinal é o mesmo espaço físico.

Figura 46 – Localização inicial do robô saindo da posição 2

Fonte: produção do próprio autor

Figura 47 – Etapas da exploração saindo da posição 2



Fonte: produção do próprio autor

Posição 3

A exploração realizada partindo da posição 3 (vista na Figura 48) pode-se observar que a localização inicial influência diretamente a quantidade de área explorada por movimento. Veja pela Figura 49-A que inicialmente o conhecimento do mapa era relativamente baixo, mas ao se deslocar para a frente e realizar a rotação de 180°para o próxima localização uma grande área de exploração foi coberta Figura 49-B.

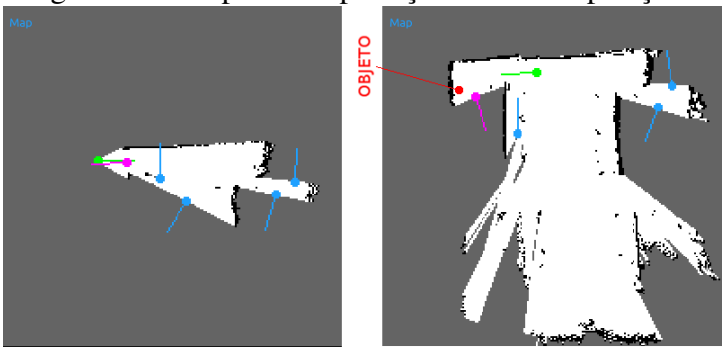
Figura 48 – Localização inicial do robô saindo da posição 3

Fonte: produção do próprio autor

Posição 4

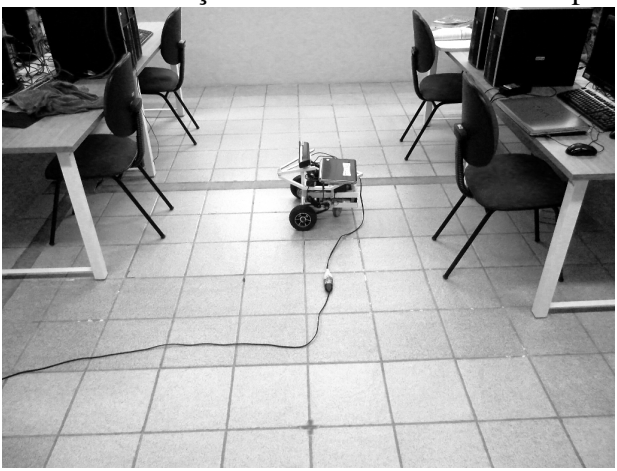
Na exploração realizada partindo da posição 4 (vista na Figura 50) embora o primeiro ponto encontrado (Figura 49-A) esteja muito próximo do inicial o processo de rotação do robô compensa garantindo uma área maior explorada. E as etapas seguem pela Figura 49-B, C, D, E e F.

Figura 49 – Etapas da exploração saindo da posição 3



Fonte: produção do próprio autor

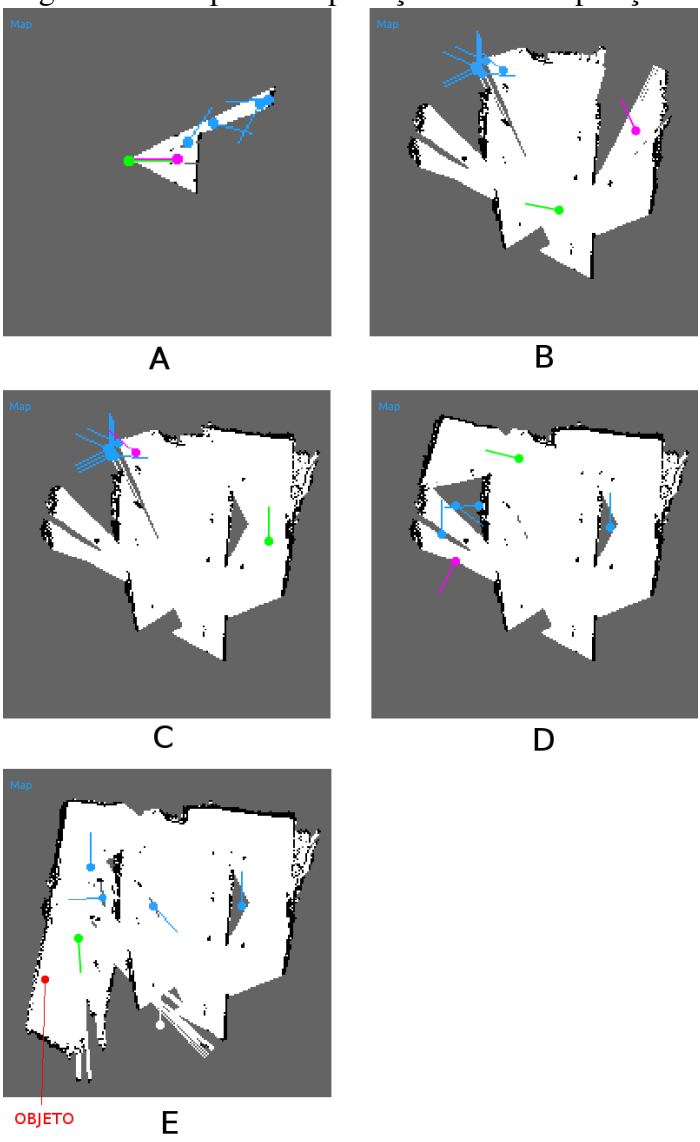
Figura 50 – Localização inicial do robô saindo da posição 4



Fonte: produção do próprio autor

.

Figura 51 – Etapas da exploração saindo da posição 4



Fonte: produção do próprio autor

6.4 DISCUSSÃO

6.4.1 Hardware

Do capítulo 2 (página 23), onde foram apresentados os requisitos de Habib, Baudoin e Nagata (2011), o XKBO contempla 6 dos 7 itens citados que são:

- Mobilidade mecânica para ambientes não estruturados tipo labirinto: Sim ✓;
- Modelagem autônoma do ambiente 3D e mapas de localização: Sim ✓;
- Possuir técnicas de navegação e localização: Sim ✓;
- Ter autonomia de decisão para qual o próximo lugar a ir: Sim ✓;
- Possuir capacidades de interação: Não ✗, porém é possível se integrar um braço robótico na estrutura;
- Possuir capacidades de comunicação: Sim ✓;
- Ser rápido, preciso e com custo acessível: Sim ✓(ver discussão adiante).

Do Capítulo 2, dos blocos fundamentais listados por Siegwart, Nourbakhsh e Scaramuzza (2011), o XKBO contempla todos, que são:

- Percepção: Sim ✓, o XKBO extrai e interpreta os dados dos sensores;
- Localização: Sim ✓, o XKBO é capaz de determinar sua posição no ambiente;

- Raciocínio: Sim ✓, o XKBO decide qual as ações a serem tomadas para realizar o objetivo, e;

- Controle de movimentação: Sim ✓, o XKBO é capaz de controlar os motores para cumprir a trajetória planejada.

O *hardware* do robô apresentado neste trabalho foi o quinto protótipo construído, pois a cada protótipo desenvolvido se observaram problemas e melhorias se fizeram necessárias.

No primeiro protótipo o problema estava relacionado ao tamanho e peso excessivo. O segundo protótipo teve o peso e tamanho reduzidos porém os motores de passo utilizados não possuíam a força necessária. Observou-se também que a posição da câmera não estava sobre o eixo entre-rodas, então reduziu-se o tamanho para o terceiro protótipo e colocou-se a câmera corretamente sobre o eixo e substituiram-se os motores de passo por motores de corrente contínua.

No entanto, o controle de malha aberta dos motores impedia realizar a odometria corretamente assim, no quarto protótipo acoplaram-se *encoders* nas rodas para contar os pulsos de deslocamento e aproveitou-se também para substituir toda a estrutura do chassi por uma configuração em alumínio mais leve. Mesmo com os *encoders* o erro da odometria estava agora nas correias utilizadas (possuía folga) e o diâmetro excessivo das rodas (que eram de bicicleta e produziam movimento demasiado).

Assim no quinto protótipo desenvolveu-se um novo chassi para as novas rodas tracionadas com correias de esteira. A estrutura permite agora organizar os diversos itens que compõem o robô como o Kinect, fontes, placas, *drivers*, motores e computador em seus devidos compartimentos. Também no quinto protótipo os motores foram substituídos por motores de passo com força e precisão maiores, pois o controle anterior para motores de corrente continua não apresentava resultados.

A ausência de baterias e necessidade em utilizar um cabo de energia no robô, se deve ao fato de que as baterias necessi-

tam constantemente serem recarregadas. O que é um transtorno no desenvolvimento, pois o robô não poderia ficar permanentemente ligado por longos períodos, sendo necessário interromper e esperar que as baterias estivessem carregadas. Também a inclusão de baterias aumenta consideravelmente o custo e peso do robô, e exige a inclusão de um inversor para alimentação das fontes aumentando a complexidade da estrutura. Porém, nada impede que se use baterias, mas este não é o foco do trabalho pois o ambiente permite usar um cabo de energia, o que é mais simples e atende as necessidades.

Constatou-se é que a velocidade de deslocamento e a precisão do movimento possuem uma relação inversa, tanto na construção do mapa do ambiente, quanto na odometria para a posição do robô. Os movimentos do robô tiveram que ser suavizados e encontrado um compromisso entre velocidade e precisão. Também existem as limitação do processamento dos dados dos sensores que limita a velocidade e para contornar este problema parte dos dados são descartados, ou seja, dos 30 *frames* que o Kinect publica por segundo apenas 1 é processado e os 29 restantes são apagados.

Embora seja possível colocar o Kinect em qualquer posição na estrutura do robô, é preferível deixa-lo precisamente sobre o eixo das rodas. Esta configuração reduz os cálculos de conversão necessários e melhora a leitura durante o movimento de rotação.

O *hardware* do XKBO é superior ao do KBO em todos os quesitos, e é suficiente para os objetivos deste trabalho:

- Na robustez do controle de movimentação do robô, devido ao uso de motores de passo acoplados a uma roda tracionada com esteira e configuração não holonômica, e possibilitou um desempenho superior ao KBO que se movimentava completamente dependente da imagem da câmera e sem precisão;

- Usa uma única câmera de profundidade (Kinect) que permite capturar a imagem da cena e mensurar distâncias conjuntamente;
- Explora e navega em ambientes desconhecidos (e sem marcadores) em busca de um objeto 3D (e não de um código de barras bidimensional previamente preparado);
- Adota uma estratégia de exploração visual ativa o que lhe confere um comportamento autônomo, otimizável e extensível;
- Adota uma arquitetura baseada uma norma internacional (STANAG 4586) e um *framework* amplamente utilizado (ROS), que permitem rápido desenvolvimento, reutilização de código, interoperabilidade e facilidade em integrar novos dispositivos, assim como integrar com outros sistemas, e realizar comunicações com outros robôs;
- A incorporação de uma unidade de processamento na estrutura do robô eliminou a sobrecarga na rede *wireless*, permitindo realizar boa parte das operações embarcada, e uma maior flexibilidade de inclusão de dispositivos no robô como o sensor Kinect;
- Dispõe de uma interface gráfica que além de mostrar a imagem capturada do ponto de vista do robô, permite observar o mapa sendo construído (e o ambiente sendo reconhecido).

Embora construir o *hardware* do robô envolva problemas e limitações, se for comparar os custos de comprar uma plataforma robótica comercial (baseada no ROS), há uma grande diferença de valores, como mostrado no comparativo da Tabela 11, onde se observa uma economia de ao menos 40% para o modelo comercial com o menor custo (desconsiderando os impostos).

Tabela 11 – Comparativo de valores com outras plataformas robóticas móveis

Valor*	USD	BRL
XKBO	931,52	2235,64
Turtle Bot 2	1600,00	3822,44
Pioneer 3	6500,00	15573,35
Husky	32000,00	76668,80

1 USD = 2,40 BRL

Fonte: produção do próprio autor

Com relação as adaptações realizadas da STANAG 4586 para o uso em robô móvel, estas foram possíveis e úteis devido a serem meramente terminologias, e não foi necessário uma mudança para acrescentar ou remover funções.

6.4.2 Software

6.4.2.1 Uso do ROS

Com relação às facilidades em utilizar o ROS como o *framework* para desenvolver o *software*, inclui-se:

- As ferramentas disponíveis, principalmente o *Stack* de navegação e o servidor do mapa;
- A troca de informações via mensagens entre os programas e pela rede, principalmente o envio e recebimento das imagens do Kinect;
- A arquitetura em grafo, que permitiu desenvolver diver-

sos programas para atender a problemas e funcionalidades específicas, e;

- O programa de visualização *rviz* que permite acesso direto para visualizar dos dados publicados em todo o sistema.

Como dificuldades que se encontrou no uso do ROS, tem-se:

- A adequação aos formatos das mensagem definidos pelo sistema, onde se fez necessário adaptar toda a arquitetura e dados em função dos formatos e operações do ROS, e;

- A questão de mudanças repentinas no ambiente a qual não é possível de ser realizada, ou seja, o robô deve se movimentar de forma suave, caso contrário o ROS perde a posição e cria um mapa equivocado que não condiz com a realidade do ambiente.

6.4.2.2 Reconhecimento

Uma vantagem na escrita do *software* foi a disponibilidade dos algoritmos para o reconhecimento de objetos como o SURF através do OpenCV, onde foi possível integrar à arquitetura do trabalho de forma direta (*plug-and-play*) e que recebe no formato correto a imagem do Kinect. Porém os resultados do reconhecimento estavam diretamente relacionados com a composição da imagem de referência. Ou seja, o segundo objeto utilizado nos experimentos possui uma textura mais complexa que o primeiro objeto e permitiu ser detectado a uma distância maior.

Com relação a resolução das imagens de referência dos objetos, para ambos, uma menor resolução permitiu reconhecer o objeto a uma distância maior, porém as imagens de referência

com uma maior resolução eram menos susceptíveis a variações de rotação.

Com relação a área do objeto, observou-se que o objeto com a área maior é reconhecido sobre um ângulo maior de rotação, embora para ambos os objetos a diferença entre 0 e 15°de rotação não apresentou diferença aparente e, a partir dos 45°o rendimento dos algoritmos reduz bastante, não sendo possível detectar a ângulos iguais ou superiores a 75°. Assim, conclui-se que, como o reconhecimento dos objetos a partir da técnica SURF ocorreu a apenas uma distância reduzida (menor que 2 metros), se faz necessário então utilizar outras características dos objetos para detectar indícios a uma maior distância, como a cor por exemplo. Essa informação de cor foi então incorporada no sistema e passou a ser utilizada para determinar a movimentação do robô para um ponto mais próximo a fim de ser possível reconhecer, o que enfatiza o processo de visão ativa.

6.4.2.3 Algoritmo de exploração

A aderência da implementação ao modelo do *framework* de busca visual apresentado (seção 4.1) mostrou-se bastante útil no desenvolvimento do XKBO, pois serviu como o ciclo principal do algoritmo de busca. Os resultados foram próximos ao esperado, porque foi possível garantir uma boa parte da área explorada, embora tenham restado pequenas regiões não alcançáveis dado o tamanho físico do robô. Percebeu-se que as métricas de tempo tem influência direta no *hardware* e unidade de processamento usado, enquanto que o número de ações, movimentos e distância estão diretamente relacionadas ao algoritmo de exploração.

Observou-se também que os parâmetros da função F(X) influenciam nos resultados dos mapas, onde dada as escolhas feitas (de ponderação para a distância, ângulo inicial, ângulo fi-

nal, e tamanho da linha de fronteira) emergiu um comportamento específico de exploração, o que significa que ficou em aberto realizar um comparativo em maior escala. Pode-se afirmar que os mapas foram "alcançados" de todas as posições.

6.4.2.4 Interface gráfica

O desenvolvimento da interface gráfica foi útil para controlar o robô e monitorar em um único local as principais informações do sistema durante a execução da tarefa.

Como a arquitetura do trabalho foi implementada no ROS, para utilizar outras estratégias de exploração diferentes, basta apenas substituir o nó responsável pelo algoritmo de exploração e manter o resto do sistema como está.

6.4.2.5 Câmera de profundidade

A vantagem que o Kinect trouxe para o trabalho foi a união das informações visuais com as informações de profundidade, que permitiu com apenas um dispositivo, reconhecer o objeto e mensurar o mapa. No entanto, uma desvantagem foi o consumo de processamento exigido pelo Kinect, que por não ser possível configurar a frequência de imagens publicadas pelo *hardware* do dispositivo para porta USB (responsável pela comunicação), esta recebia 17 MB/s de dados que eram processados e convertidos pelo *driver* OpenNI, apesar de que a maioria era posteriormente descartada pelo sistema do XKBO.

7 CONCLUSÃO

A área de pesquisa que envolve encontrar soluções para as tarefas realizadas por robôs de serviço em ambiente civis é um grande desafio, e tem como possíveis resultados para sociedade uma nova revolução no mercado de trabalho, como ocorreu anteriormente nas fábricas.

Um dos problemas que está sendo pesquisado é a tarefa de busca visual, que possibilitará aos robôs fazerem uso dos objetos disposto pelo ambiente na execução de suas tarefas. As pesquisas para este problema são recentes, no entanto o desenvolvimento e uso das câmeras RGB-D, que permitem associar informações visuais com as de profundidade, tem avançado as pesquisas pois provem ao robô a capacidade de conhecer o ambiente e o objeto da busca. Também, tem-se a exploração robótica móvel como uma das partes constituinte deste problema, e o seu desenvolvimento está direcionado principalmente na estratégia a ser utilizada.

O objetivo deste trabalho foi desenvolver um robô móvel capaz de executar a tarefa de busca por um objeto em um ambiente desconhecido através do uso de uma câmera de profundidade. O robô móvel desenvolvido, XKBO, uma eXtensão melhorada do robô KBO, é capaz de se mover autonomamente pelo ambiente graças às informações de profundidade da câmera Kinect e é capaz de reconhecer objetos armazenados na base de imagens, devido aos algoritmos de reconhecimento que usam a imagem RGB provida pelo Kinect.

O uso de visão ativa trouxe um auxílio à tarefa de busca visual, pois as técnicas de reconhecimento de objeto têm limitações na detecção relacionadas a distância e ângulo em que o objeto se encontra, o que caracteriza que a inteligência do robô deve ser capaz de perceber alguma informação do objeto-alvo e, com base nesta informação, realizar uma aproximação a fim de

refutar ou não a presença do objeto.

As principais contribuições obtidas com este trabalho incluem:

- O desenvolvimento de um robô móvel de baixo custo com um único sensor (câmera de profundidade), capaz de explorar salas em busca de um objeto, e que possibilita o desenvolvimento de pesquisas na área de robótica móvel e visão computacional, com ênfase na exploração e mapeamento de ambientes;

- A adaptação da arquitetura presente na norma STANAG 4586 para robôs móveis terrestres;

- A proposição de uma nova estratégia, Busca por Fronteiras, para visão ativa que foi validada para um mapa 2D, sensor RGB-D único, e faz o uso das informações visuais do objeto para direcionar a nova posição pela presença de indícios (cor) do objeto na cena;

- A proposição de um novo algoritmo para listar e priorizar próximas posições de exploração para um mapa 2D e sensor RGB-D único, com estratégia de exploração considerando distância, ângulo, e tamanho da região de fronteira;

- A concepção de um *framework* e interface IHC representando a busca visual, com possibilidade de testar diversas estratégias e técnicas de reconhecimento na forma *plug-and-play* (modular).

O principal diferencial deste trabalho em relação a literatura está na concepção e caracterização de um *framework* que representa a decomposição da tarefa de busca por objetos.

E conclui-se que este trabalho permite, de agora em diante, ser uma plataforma para realizar pesquisas na área de robótica de serviços.

7.1 PROPOSTA DE TRABALHOS FUTUROS

As propostas de trabalhos futuros serão divididas em dois conjuntos, o primeiro de menor complexidade que são:

- Realizar um levantamento das estratégias de exploração existentes, e realizar um comparativo das funções, parâmetros e informações utilizadas para decidir qual a próxima posição (do robô) a ser escolhida, pois o grande diferencial encontrado está exatamente na função F(X) escolhida para a exploração. Assim, para chegar a uma nova função F(X) pode-se testar, de forma incremental, a inclusão de novos parâmetros e informações e sua relevância /contri-buição para os resultados;

- Pesquisar o uso e controle para a inclusão de um braço robótico com garra, para ser possível a interação com os objetos, e desempenho de outras tarefas;

- Expandir a aplicação para mais de um ambiente, ou seja, situações com várias salas, e;

- Pesquisar e descobrir um sensor alternativo para que seja possível utilizar o robô XKBO em ambientes externos.

E para trabalhos de maior densidade e conceituais, propõe-se:

- Pesquisar estratégias de exploração coletivas (com múltiplos robôs) para o uso em situações de S&R, porque nestas situações o tempo é fator crítico e o ambiente possui uma larga escala. Assim a divisão da exploração (paralelismo) pode ser desenvolvida para reduzir o tempo e contornar o

problema. No entanto, deve-se observar como será realizada a divisão da tarefa, comunicação e troca de informações, pois devido a estrutura do ambiente na forma de labirinto com paredes densas, os sinais de comunicação podem ser bloqueados, e;

- Pesquisar e desenvolver uma proposta de configuração de *hardware* otimizada, considerando todos os usos do robô, porque ainda não existe um modelo de configuração *hardware* padronizado. Isto pode ser feito através da definição de parâmetros que representam a constituição da configuração física do robô, e pelo uso de simulações em um número considerável de ambientes complexos, utilizar algoritmos genéticos para encontrar os valores para os parâmetros que representam a melhor configuração de *hardware*;

REFERÊNCIAS

ALCANTARA, M. F. de; **Códigos de barras bidimensionais com identificação de posição e orientação no espaço tridimensional**. Dissertacão de mestrado, Universidade do Estado de Santa Catarina - UDESC. 2012.

ALOIMONOS, J.; WEISS, I.; BANDYOPADHYAY, A.; **Active vision**. International Journal of Computer Vision, 1(4):333-356. Kluwer Academic Publishers. 1988.

ANDREOPOULOS, A.; HASLER, S.; WERSING, H.; JANSSEN, H.; TSOTSOS, J.; KORNER, E. **Active 3d object localization using a humanoid robot**. IEEE Transactions on Robotics, 27(1):47-64. IEEE. 2011.

AYDEMIR, A.; PRONOBIS, A.; GOBELBECKER, M.; JENSFELT, P. **Active visual object search in unknown environments using uncertain semantics**. Robotics, IEEE Transactions on, PP(99):1-17. IEEE. 2013.

BAJCSY, R. **Active perception vs. passive perception**. University of Pennsylvania, Department of Computer and Information Science. 55-59. 1985.

BAJCSY, R. **From active perception to active cooperation:** fundamental processes of intelligent behavior. Visual Attention and Cognition, Advances in Psychology. 116:309-321. North-Holland. 1996.

BALLARD, D. H. **Animate vision**. Artificial Intelligence, 48(1):57-86. 1991.

BASILICO, N.; AMIGONI, F. **Exploration strategies based on multi-criteria decision making for searching**

environments in rescue operations. Autonomous Robots, 31(4):401-417. 2011.

BAY, H.; TUYTELAARS, T.; GOOL, L. **SURF: Speeded Up Robust Features**. Computer Vision – ECCV, Lecture Notes in Computer Science, 3951:404-417. Springer Berlin Heidelberg. 2006.

CORMEN, T.H; **Algorithms Unlocked**. Computer Science, Ed.3. MIT Press. 2013

CORREA, D. S. O.; SCIOTTI, D. F.; PRADO, M. G.; SALES, D. O.; WOLF, D. F.; OSÓRIO, F. S. **Mobile robots navigation in indoor environments using kinect sensor**. In Critical Embedded Systems (CBSEC), 2012 Second Brazilian Conference on, 36-41. IEEE. 2012.

DAVIES, E. R. **Machine Vision:** Theory, Algorithms, Practicalities. Morgan Kaufmann, Ed.3. 2005.

EDEN, A. H. **Three Paradigms of Computer Science**. Minds & Machines, 17:135-136. 2007.

FERMULLER, C.; ALOIMONOS, Y. **Vision and action**. Image and Vision Computing, 13(10):725-744. 1995.

GIBSON, J. J. **The Ecological approach to visual perception**. Psychology Press, new edition edition. 1986.

GIL, A. C. **Como elaborar projetos de pesquisa**. São Paulo: Atlas. 1991.

GONZALEZ-JORGE, H.; RIVEIRO, B.; VAZQUEZ-FERNANDEZ, E.; MARTÍNEZ-SÁNCHEZ, J.; ARIAS, P. **Metrological evaluation of Microsoft Kinect and Asus Xtion sensors**. Measurement: Journal of the International Measurement Confederation, 46(6):1800-1806. 2013.

Gregory, S. R. **Pure and Applied Science**. Journal of the Royal Astronomical Society of Canada, 36:13-16. 1942.

GRISETTI, G.; STACHNISS, C.; BURGARD, W. **Improved techniques for grid mapping with rao-blackwellized particle filters**. IEEE Transactions on Robotics, 23(1):34-46. IEEE. 2007.

HABIB, M.K.; BAUDOIN, Y.; and Nagata, F. **Robotics for rescue and risky intervention**. IECON Proceedings (Industrial Electronics Conference). 3305-3310. 2011.

HINTON, G.; SALAKHUTDINOV, R. R. **Reducing the dimensionality of data with neural networks**. Science, 313(5786):504-507. American Association for the Advancement of Science. 2006.

HYVÄRINEN, A.; HURRI, J.; HOYER, P. O. **Natural Image Statistics:** A Probabilistic Approach to Early Computational Vision. Computational Imaging and Vision, Volume 39. Springer London. 2009.

Juliá, M.; Gil, A.; Reinoso, O. **A comparison of path planning strategies for autonomous exploration and mapping of unknown environments**. Autonomous Robots, 33(4):427-444. Springer US 2012.

KALAL, Z.; MATAS, J.; MIKOLAJCZYK, K. **Pn learning:** Bootstrapping binary classifiers by structural constraints. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010 , 49-56. IEEE. 2010.

LIU, Y.; NEJAT, G. **Robotic urban search and rescue: A survey from the control perspective**. Journal of Intelligent and Robotic Systems: Theory and Applications, 72(2):147-165. 2013.

LOWE, D. G. **Distinctive image features from scale-invariant keypoints**. International journal of computer vision, 60(2):91-110. Springer. 2004.

MARTINEZ, A.; FERNÁNDEZ, E. **Learning ROS for Robotics Programming**. [S.I]. 2013.

MORRIS, C. W. **The Relation of the Formal and Empirical Sciences within Scientific Empiricism**. Erkenntnis, 6-16. 1935.

MÖRWALD, T.; PRANKL, J.; RICHTSFELD, A.; ZILLICH, M.; VINCZE, M. **Blort-the blocks world robotic vision toolbox**. Proc. ICRA Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation. 2010.

NATO. **Standard Intefaces of UAV Control System (UCS) for NATO UAV Interoperability**. [S.I]. 2005.

OUAARAB, A.; AHIOD, B.; YANG, X. **Improved and Discrete Cuckoo Search for Solving the Travelling Salesman Problem**. Cuckoo Search and Firefly Algorithm, Studies in Computational Intelligence, 516:63-84. Springer International Publishing. 2014.

PEDRINI, H.; SCHWARTZ, W. R. **Análise de imagens digitais: princípios, algoritmos e aplicações**. Thomson Learning. 2007.

PENATTI, O. A. B. **Estudo comparativo de descritores para recuperação de imagens por conteúdo na web**. Dissertação de Mestrado, Universidade Estadual de Campinas. 2009.

PFEIFER, R.; SCHEIER, C. Understanding Intelligence. MIT Press, Cambridge. 1999.

QUIGLEY, M.; CONLEY, K.; GERKEY, B.; FAUST, J.; FOOTE, T.; LEIBS, J.; WHEELER, R.; NG, A. Y. **ROS: an open-source Robot Operating System**. ICRA workshop on open source software, 3(3.2). 2009.

ROTHENSTEIN, A. L; TSOTSOS, J. K. **Attention links sensing to recognition**. Image and Vision Computing, 26(1):114-126. 2008.

RUANGPAYOONGSAK, N.; ROTH, H.; CHUDOBA, J. **Mobile robots for search and rescue**. Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International, 212-217. 2005.

SAE. **AS5710 JAUS Core Service Set**. [S.I]. 2008.

SHUBINA, K.; TSOTSOS, J. K. **Visual search for an object in a 3d environment using a mobile robot**. Computer Vision and Image Understanding, 114(5):535-547. 2010.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots**. Intelligent robotics and autonomous agents. MIT Press. 2011.

SILVA, E. L. de; MENEZES, E. M. **Metodologia da pesquisa e elaboração de dissertação**. Florianópolis: UFSC, p14. 2005.

SPAAN, M. T. J. **Partially Observable Markov Decision Processes**. Reinforcement Learning, Adaptation, Learning, and Optimization, 12:387-414. Springer Berlin Heidelberg. 2012.

STACHNISS, C.; HÄHNEL, D.; BURGARD, W.; GRISETTI, G. **On actively closing loops in grid-based FastSLAM**. Advanced Robotics, 19(10):1059-1079. 2005.

THRUN, S. **Simultaneous Localization and Mapping**. Robotics and Cognitive Approaches to Spatial Mapping, 38:13-41. Springer Berlin Heidelberg. 2008.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic Robotics**.Intelligent robotics and autonomous agents series. MIT Press. 2005.

TSOTSOS, J. K. **On the relative complexity of active vs. passive visual search**. International Journal of Computer Vision. 7(2):127-141. 1992.

TZAFESTAS, S.G. **Introduction to Mobile Robot Control**. Elsevier Science. 2013.

VENTURA, R.; LIMA, P.U. **Search and rescue robots:** The civil protection teams of the future. Proceedings - 3rd International Conference on Emerging Security Technologies - EST, 12-19. 2012.

VIECILI, E.; ALCANTARA, M.; ROMÃO, A.; HOUNSELL, M.; PILLON, M.; HEINEN, M. **Desenvolvimento de um robô móvel com visão para propósitos educacionais**. XL Congresso Brasileiro de Educação em Engenharia. Abenge. 2012.

VIOLA, P.; JONES, M. **Rapid object detection using a boosted cascade of simple features**. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1:I-511-I-518. 2001.

WAZLAWICK, R. S. **Uma Reflexão sobre a Pesquisa em Ciência da Computação à Luz da Classificação das Ciências e do Método Científico**. Revista de Sistemas de Informação da FSMA, (6):3-10. 2010.

YE, Y.; TSOTSOS, J.K. **Sensor planning for 3d object search**. Computational Visual Imagem Understanding. 73(2):145-168. 1999.

APÊNDICE A -- Diagramas de Software

A.1 ALGORITMO DE NAVEGAÇÃO

A navegação é realizada pelo *Navigation Stack* do ROS. O qual tem a configuração interna de acordo com o diagrama detalhado na Figura 52.

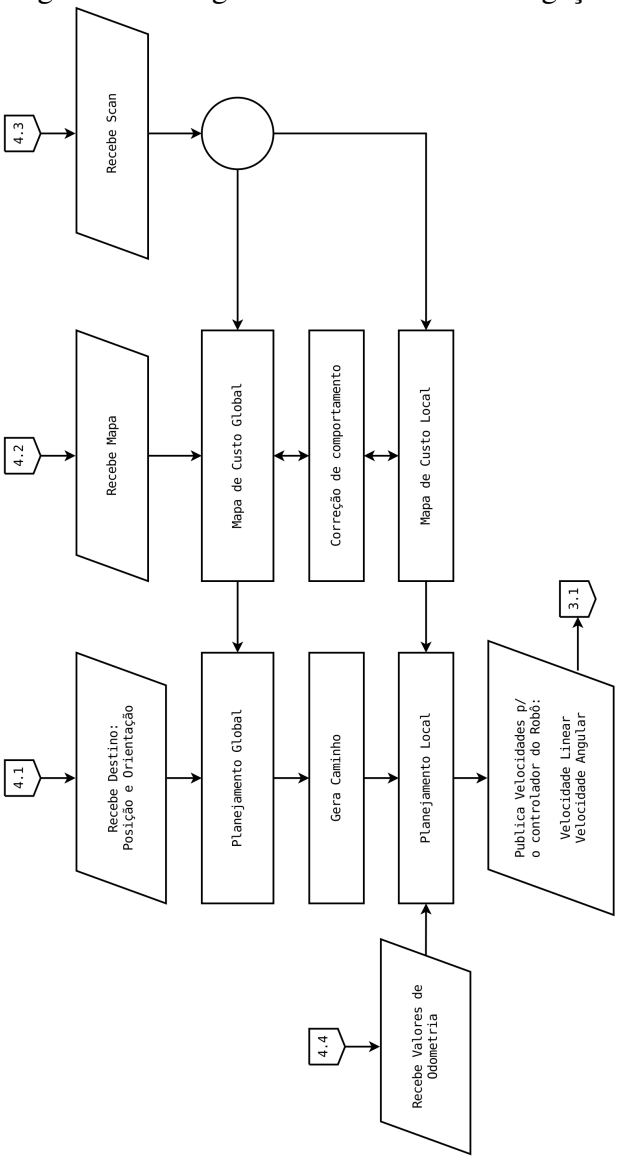
A.2 AGORITMO DE CONTROLE DO ROBÔ

O controle do robô é detalhado na Figura 53

A.3 ALGORITMO DE ODOMETRIA

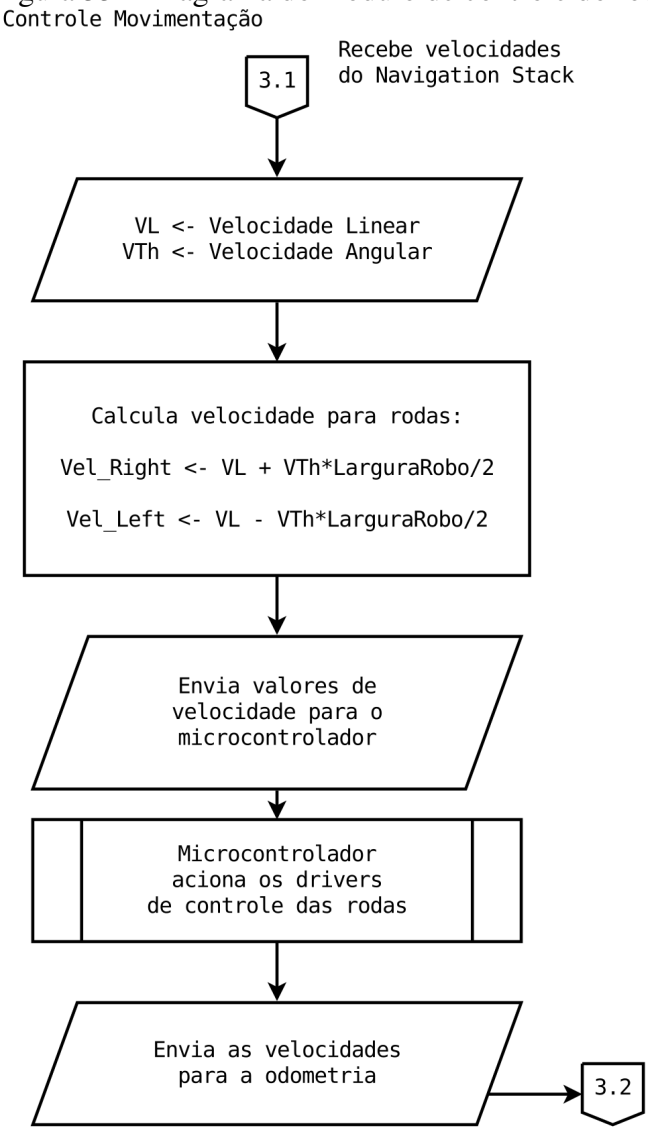
A odômetria do robô é detalhado na Figura 54, a qual é realizada em malha aberta porém sua precisão é garantida pelo motor de passo.

Figura 52 – Diagrama do módulo de navegação



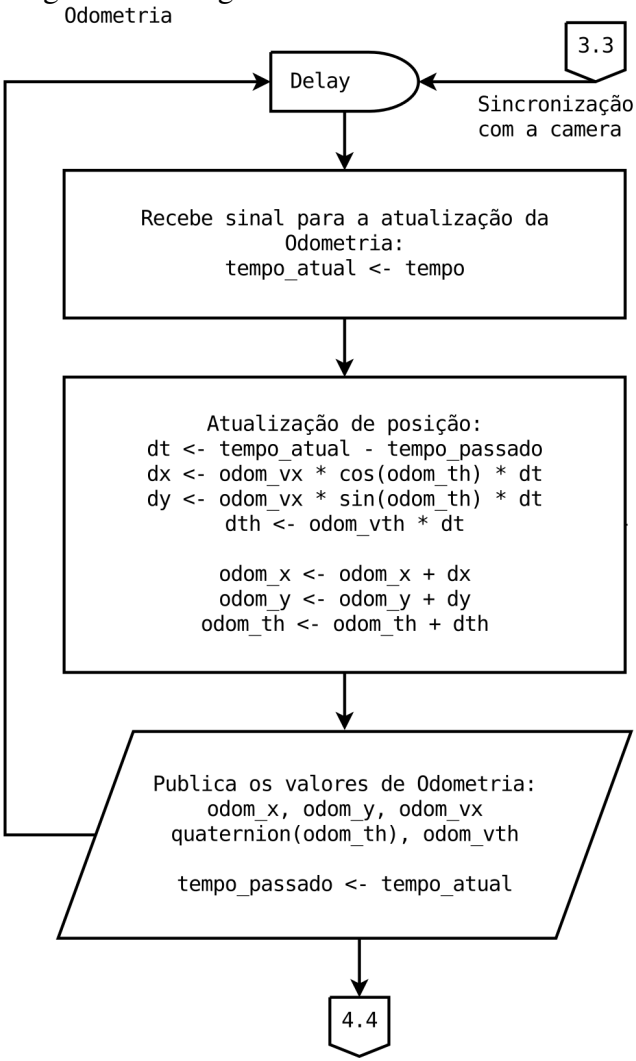
Fonte: produção do próprio autor

Figura 53 – Diagrama do módulo de controle do robô



Fonte: produção do próprio autor

Figura 54 – Diagrama do módulo de odômetria.



Fonte: produção do próprio autor