Computer Vision applied to mobile devices for automotive robotics

Guilherme Giaccheto Moreira
Centro Universitário Senac
São Paulo, Brasil
guilherme.moreira@projetosentinela.com.br

Mario Leandro Pires Toledo
Centro Universitário Senac
São Paulo, Brasil
mario.toledo@projetosentinela.com.br

Fábio Roberto de Miranda
Centro Universitário Senac
São Paulo, Brasil
fabio.rmiranda@sp.senac.br

Abstract – This work presents the results of using an Android phone as a control unit for a Lego NXT Robot. The smartphone's camera and superior processing power where compared to the NXT CPU enable the application of some computer vision techniques in the control of the robot.

Keywords – computer vision, robotics

I. INTRODUÇÃO

A robótica promove diversos tipos de benefícios a sociedade. Os robôs podem, normalmente, realizar trabalhos enfadonhos e repetitivos para um humano, dificilmente cometendo erros.

A grande maioria dos robôs que são autônomos (robôs que não necessitam da intervenção humana para execução de suas tarefas) necessita de sensores para a detecção do ambiente a sua volta, executando suas ações baseando-se nos resultados obtidos por estes sensores.

Os demais dispositivos sensoriais aplicados a um robô permitem que o mesmo adquira informações sobre o ambiente a sua volta, simulando os sentidos humanos.

A visão computacional desempenha um papel de grande importância na robótica móvel. Quando se pretende utilizar o sentido da visão para a localização e navegação, problemas maiores aparecem. A integração de câmeras de vídeo ao já complexo hardware traz complicadores, como por exemplo, o requerimento de rotinas elaboradas para a extração de informações úteis a partir do tratamento e processamento das imagens obtidas [1].

Visão computacional nada mais é do que um conjunto de técnicas e métodos que permitem que um sistema consiga interpretar imagens. A interpretação de uma imagem pode ser representada computacionalmente pela transformação de um conjunto de dados digitais representando uma imagem.

O sistema de visão computacional, segundo [2] pode ser dividido em etapas, classificadas da seguinte forma:

- Aquisição da Imagem: O processo de aquisição da imagem é feito a partir de um sensor de visão (como uma câmera, por exemplo), podendo capturar uma ou várias imagens sequenciais;
- Pré-processamento: Consiste em aplicar métodos de processamento de imagem, antes de extrair informações da mesma. Um exemplo deste processo é a captação de contornos em uma determinada área da imagem;
- Extração de características: Este processo consiste em capturar as características de uma imagem e transforma-las em modelos matemáticos, como texturas transformadas em matrizes;
- Detecção e segmentação: Este processo consiste em detectar uma região da imagem, com a finalidade de, por exemplo, detectar regiões semelhantes na imagem;
- Processamento de alto nível: Neste processo, os dados são passados para métodos de validação das informações, para verificação da satisfação dos dados, além da classificação dos objetos detectados;

Neste projeto, são utilizadas técnicas de visão computacional para processar imagens recebidas a partir da câmera de um smartphone com sistema Android, permitindo a locomoção autônoma de um robô do modelo LEGO NXT através de ambientes fechados (como salas ou corredores), objetivando o auxílio à locomoção de pessoas com deficiência visual.

Este artigo discute a abordagem da robótica autônoma baseada em visão computacional, apresenta uma arquitetura para o sistema e mostra os resultados iniciais e finais do projeto, indicando também seus possíveis trabalhos futuros.

II. TRABALHOS RELACIONADOS

O robô Baxter e o robô da empresa NSK são exemplos de robôs autônomos baseados em sensores, e a seguir, são comparados com o robô proposto neste projeto.

O robô Baxter [3] foi construído para se adaptar ao ambiente que foi designado, podendo efetuar uma série de habilidades, como empacotamento e desempacotamento, empilhamento ou desempilhamento de itens em superfícies móveis ou fixas ou operações.

Estes processos só podem ser desempenhados devido aos demais sensores inclusos em Baxter, como um sistema de detecção de presença humana através de um sonar, identificação de objetos por câmeras, sensor de peso e força nas juntas dos braços e outros sensores menores.

Figura 1 - O robô baxter, em uma atividade industrial (www.newventurist.com).

O desenvolvimento de um robô que utiliza sensores externos não é prático, e requer um grande custo operacional, além da necessidade da adaptação de cada sensor e a compreensão do protocolo de comunicação de cada um, diferente do uso neste projeto, onde a utilização de um smartphone permite concentrar os demais sensores em um único dispositivo, além de tornar sua câmera o principal meio externo de comunicação.

Para o problema de guia de pessoas com deficiência visual, a empresa japonesa NSK desenvolveu em 2011 um robô capaz de guiar pessoas cegas em ambientes públicos. Este robô possui uma série de câmeras interligadas, e o aparelho Kinect (o sensor de movimentos da Microsoft, usado para jogos) como seu principal sensor.

Figura 2 - O robô da empresa NSK, capaz de identificar escadas e guiar pessoas com deficiência.

Diferente do robô da empresa NSK, este projeto oferece uma solução de baixo custo para o desenvolvimento de um robô capaz de guiar pessoas com deficiência visual. Este projeto também visa, inicialmente, a locomoção do robô dentro de ambientes fechados, se baseando em rotas e guias para sua movimentação.

III. SOBRE O PROJETO

O projeto deu início com a proposta do desenvolvimento de um robô capaz de se locomover autonomamente. Este tipo de solução pode ser utilizado em diversos problemas da sociedade, como por exemplo, uma forma de auxiliar deficientes visuais a se locomoverem em ambientes fechados, guiando-os até seu destino.

A ideia da utilização de um dispositivo móvel como a principal fonte de processamento e interação com o mundo externo surgiu pelo próprio interesse do desenvolvimento com uso da plataforma Android. O celular deve estar conectado ao robô, utilizando sua câmera para visualizar o caminho a sua frente, tomando decisões a partir do processamento das imagens captadas, utilizando técnicas de visão computacional.

A. Montagem do Robô

O modelo escolhido para o protótipo do robô foi o LEGO NXT, disponível para uso acadêmico, e de fácil manutenção e operação. Este modelo de robô possui uma única peça principal, chamada de "Brick", responsável pela embarcação de código e envio de sinal aos demais periféricos conectados a ele, como motores e outros sensores.

A arquitetura do robô é semelhante ao protótipo do Tribot da LEGO, com algumas modificações. Existem dois motores, localizados em cada parte da peça principal, e são responsáveis pela locomoção do robô. Para movimentar o robô para frente e para trás, basta acionar os motores na mesma direção e velocidade, enquanto é possível faze-lo girar em seu próprio eixo acionando os motores em direções diferentes.

Figura 3 - Protótipo final do robô utilizado neste projeto

Alguns eixos foram adicionados para suporte da peça, além de uma roda traseira, conectada apenas por uma haste, que fica solta para garantir o suporte do robô.

A frente do protótipo foi construído um suporte para encaixe do smartphone, de forma que a câmera do disposto fique sempre voltada a sua frente.

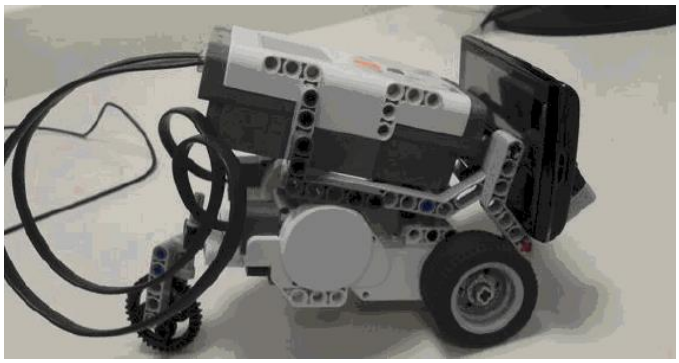

Figura 4 - O robô é suportado por uma haste traseira, e possui um suporte a sua frente, a fim de segurar o smartphone.

B. Aplicação de Controle

O processamento lógico do robô deve ficar por conta do smartphone. Para isto, foi desenvolvido um aplicativo para o sistema Android, com a função de gerenciar todo o processamento necessário para realizar a autonomia do robô. Desta forma, o smartphone passa a processar as informações, enquanto o robô fica apenas com a função de receber instruções de locomoção. O desenvolvimento desta aplicação foi realizado a partir da IDE do Eclipse, juntamente com o SDK de desenvolvimento para Android.

Para que o celular e o robô se comuniquem, a aplicação deve se conectar ao robô utilizando a tecnologia Bluetooth para que haja troca de informações. Neste caso, a aplicação deve habilitar o suporte bluetooth do celular, conectando-se ao sinal do robô, enviando e recebendo informações.

A comunicação entre os sensores digitais e a Brick é feita utilizando o protocolo I²C. O [4] diz que cada sensor é tratado como um dispositivo de memória e cada transação é tratada como leitura ou escrita em uma posição da memória.

Cada sensor digital possui dois tipos de endereços:

1. Um endereço I²C único para cada sensor digital.
2. 2. Um endereço interno para leitura ou escrita.

Para a comunicação entre o celular e a Brick, utilizamos o protocolo LCP (Lego Communication Protocol). O LCP é um protocolo de comunicação, utilizado para estabelecer a comunicação entre a Brick e dispositivos terceiros como Celulares e Computadores (PC's), conforme descrito na documentação em [5].

Para a comunicação via Bluetooth é definido o seguinte padrão de pacote a ser enviado:

Length, LSB	Length, MSB	Command Type	Command	Byte 5	Byte 6	Etc.

Bluetooth® protocol packages

Figura 5 - Protocolo de envio de informações Bluetooth.

Os dois primeiros bytes são utilizados especialmente para comunicação via Bluetooth e dizem informações sobre o tamanho do pacote esperado pelo receptor.

Com o protocolo de comunicação especificado, foi possível enviar instruções para locomoção do robô através do celular. Com esta base, foi possível desenvolver os próximos módulos de controle do robô.

A aplicação de controle possui uma interface simples, e necessita da interação inicial do usuário para conexão com o robô. Para tal, ao abrir o aplicativo, o usuário tem a opção de pressionar o botão "Conectar", acionando o dispositivo bluetooth do celular e pesquisando pelos demais dispositivos bluetooth na proximidade.

Figura 6 - Tela inicial e tela de pesquisa de dispositivos via Bluetooth.

Após encontrar os dispositivos próximos, o usuário tem a opção de selecionar a opção equivalente ao robô do modelo NXT, desde que o mesmo esteja na lista. Desta forma, o usuário pode dar entrada a uma das opções de destino, iniciando assim o processo de deslocamento autônomo do robô.

IV. CONTROLE AUTÔNOMO

O módulo de controle autônomo do robô é composto do funcionamento de 3 sub-módulos, que demandam da utilização de técnicas de visão computacional.

A proposta deste projeto considerou o desenvolvimento da autonomia do robô de uma forma que o ambiente em que ele atue não sofra muitas adaptações para seu funcionamento. Devido a isto, o uso de técnicas de visão computacional foi de extrema necessidade, fazendo com que o robô consiga detectar seu caminho sem muita dificuldade. Com os 3 sub-módulos atuando juntos, a aplicação de controle é capaz de identificar rotas e caminhos a serem seguidos pelo robô.

O desenvolvimento destes módulos utilizam de recursos da biblioteca OpenCV. Ela permite realizar tratamentos específicos em imagens, variando entre filtros, calibração de câmera, reconhecimento de objetos e outros recursos.

Cada um dos sub-módulos operantes possui como entrada uma imagem capturada a partir da câmera do celular. A cada segundo, a câmera captura um número variante de imagens, compatíveis com sua capacidade processamento. Os métodos do OpenCV permitem processar a imagem de cada quadro capturado, aplicando filtros ou detectando padrões.

A atuação de cada módulo é detalhada melhor na imagem abaixo.

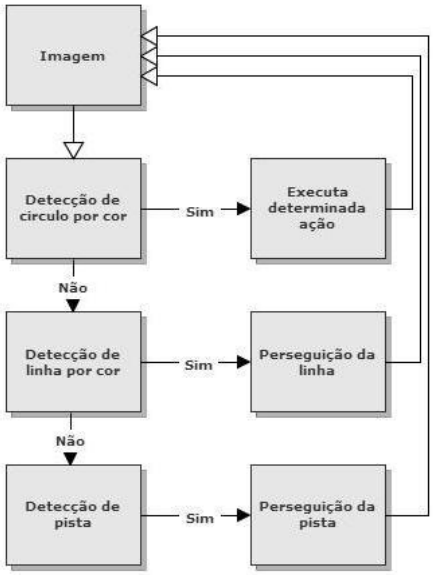

Figura 7 - Arquitetura do funcionamento dos submódulos constituintes do controle autônomo.

O que se segue são detalhes de cada um dos módulos que constituem o controle autônomo do robô.

A. Módulo de detecção de linhas por cores

Este módulo de detecção de linhas por cores utiliza métodos de Color Blob Detection. Utilizando a interação do smartphone, o usuário pode selecionar um ponto específico da tela. Ao fazer isto, o método inRange do OpenCV é utilizado para extrair um range de cores específicos escolhido pelo usuário, passando uma imagem nos padrões HSV para o método. Este método retorna uma imagem binarizada, onde os pontos na cor branca estão na faixa de cores especificada.

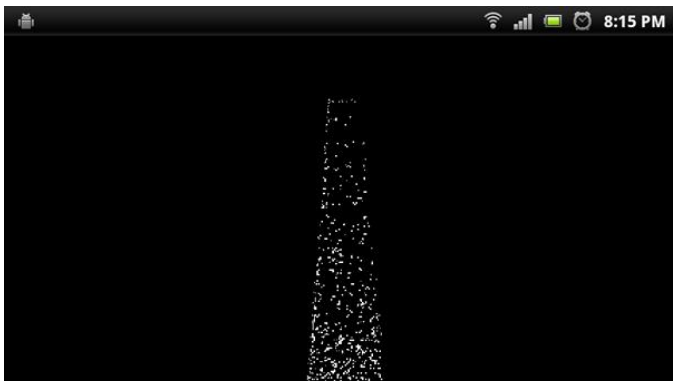

Figura 8 - O retorno do método inRange é uma imagem binarizada, com as cores no padrão HSV.

A partir do momento em que existe uma cor selecionada, utilizamos os métodos da biblioteca OpenCV para encontrar os contornos na imagem processada, utilizando o algoritmo Suzuki85. Com o método findCountours, podemos facilmente localizar os contornos em uma imagem binarizada. Os contornos são uma ferramenta poderosa para análise de forma e detecção de objetos.

O método findContours permite encontrar os contornos ao redor das áreas baseadas na cor selecionada. Para cada contorno encontrado, utilizamos um filtro que considera apenas as áreas que possuem uma área maior que 10% da maior região encontrada. Isto permite eliminar pontos e pequenos locais expostos na área de visão da câmera.

Este módulo permite que a câmera do celular detecte uma linha no chão, ou em específico, um trajeto a ser seguido. Para tal, desenhamos um trajeto em uma cor visível para a câmera do celular, possibilitando a detecção do contorno.

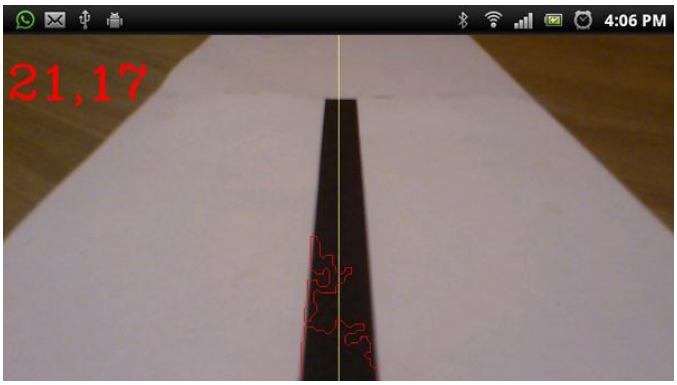

Figura 9 - Com o uso do método findCountours permite encontrar os contornos ao redor das áreas da cor escolhida.

Baseado nos contornos encontrados a partir do filtro, o algoritmo calcula a quantidade de pontos localizados a esquerda e a direita do meio da tela. Esta quantidade de pontos permite calcular a velocidade de cada motor do robô, definida da seguinte forma:

$$Sejam\ velocidade_{minima}\ e\ velocidade_{variada}\ constantes\ préviamente\ definidas.$$

$$u1 = \left(\frac{quantidade\ de\ pontos\ na\ esquerda}{total\ de\ pontos}\right) * velocidade_{minima}$$

$$u2 = \left(\frac{quantidade\ de\ pontos\ na\ direita}{total\ de\ pontos}\right) * velocidade_{minima}$$

Para u1 maior que u2: $u1 = u1 * 0{,}5$

Para u2 maior que u1: $u2 = u2 * 0{,}5$

$$velocidade_{esquerda} = velocidade_{minima} + u1$$
$$velocidade_{direita} = velocidade_{minima} + u2$$

Figura 10 - Fórmula da velocidade de cada motor do robô.

B. Módulo de detecção de pistas

Através do módulo de detecção de pistas, o robô consegue identificar possíveis rotas a sua frente, de uma forma a se adaptar melhor a ambientes diferentes, ou que não possuam uma linha definida para seguir. Este módulo utiliza métodos de Lane Detection, detectando locais que contenham características de uma pista, e passando a locomover o robô na direção da convergência da mesma.

O primeiro procedimento no algoritmo é transformar a imagem capturada pela câmera para uma escala de cinza. Este processo é feito através do método cvtColor da biblioteca OpenCV. Com a imagem em escala de cinza, é aplicado um filtro de blur, a fim de remover os demais ruídos da imagem, deixando-a mais suave. Para isto, é utilizado o método blur do OpenCV.

Em seguida, é utilizado o filtro de Canny, a fim de detectar as bordas da imagem. Utilizando o método Canny da biblioteca OpenCV, é possível obter o retorno da imagem com suas bordas destacadas.

Figura 11 - Aplicação do filtro de Canny.

Por questões de controle do robô e melhor detecção da pista à frente, a imagem foi dividida ao meio, em uma metade esquerda e outra direita. Para cada parte, é aplicado o método de HoughTransform, através do uso do método HoughLines do OpenCV. Desta forma, são obtidas as linhas da imagem, descritas pelo ângulo e distância do centro da imagem (coordenada 0, 0), permitindo desenha-las na tela.

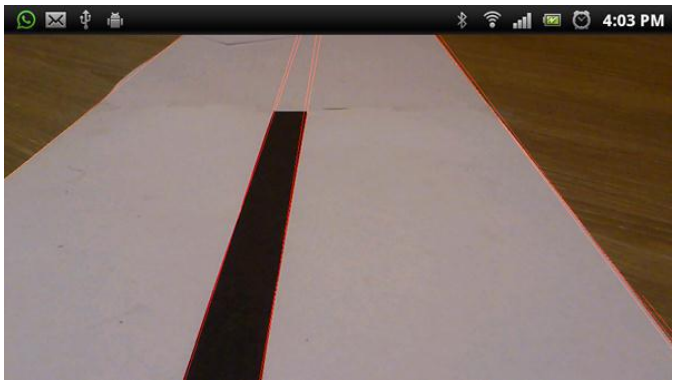

Figura 12 - O retorno do método HoughLines permite exibir as linhas das possíveis rotas.

Em seguida, é necessário realizar um filtro em cima das retas encontradas para cada metade da tela (esquerda e direita), a fim de destacar a rota a ser seguida. Para isto, o algoritmo se baseia na coordenada x de cada reta com o limite inferior da câmera, selecionando a intersecção que estiver mais próxima do centro da tela. Quando pelo menos uma das retas que formam a pista é encontrada, é calculada a velocidade de cada motor do robô, possibilitando sua locomoção.

C. Módulo de detecção de símbolos

O módulo de detecção de símbolos permite que o robô consiga determinar quando chegou a seu destino. Seu propósito esta na detecção de diferentes símbolos em seu trajeto, parando sua locomoção quando encontrado a quantidade de círculos desejada (indicando a chegada na posição desejada).

Este módulo faz um pré-processamento parecido com o do módulo de detecção de pista. A partir da imagem capturada pela câmera, é utilizado o método findContours para detecção dos contornos, porém, com uma faixa de cores hsv diferente da utilizada no módulo de detecção de pistas.

Com os contornos encontrados, é realizada uma filtragem para detecção de símbolos circulares. Para tal, é utilizado o método arcLenght da biblioteca OpenCV, a fim de se obter o tamanho do contorno, e compara-lo se o mesmo possui o tamanho de 2π.

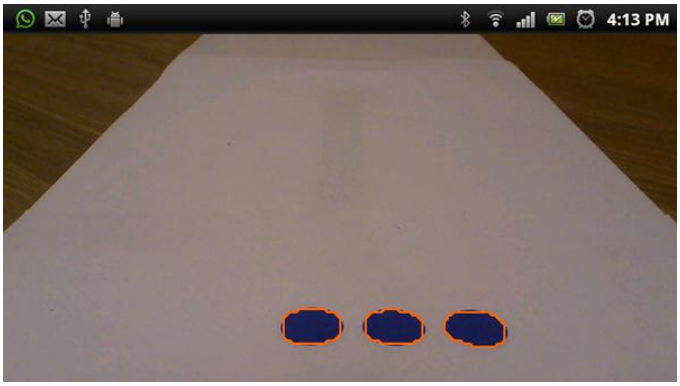

Figura 13 - Aplicação do filtro para detecção de símbolos circulares.

Para fins de mobilidade, o algoritmo consegue identificar múltiplos círculos. A partir disto, basta identificar o local de parada por um número específico de símbolos. Também é possível identificar outras ações ao robô, como girar em 90 ou 180 graus, dependendo do número de círculos encontrados.

V. RESULTADOS

Neste projeto, foram construídos diversos tipos de protótipos para a validação da tecnologia desenvolvida. O módulo de detecção de linhas por cores foi desenvolvido com o propósito da detecção de trajetos e rotas baseado em cores específicas. Na primeira versão deste módulo, o aplicativo acionava ambos os motores em uma velocidade constante sobre a linha, girando sobre o próprio eixo quando a necessidade de realizar curvas.

Para fins de testes, foram impressos trajetos de linhas pretas, para que o robô conseguisse se locomover. O robô conseguiu se locomover em linha reta como esperado quando identificado à rota, porém, teve problemas quando o trajeto possuía curvas, ou o robô saísse de uma trajetória linear. Desta forma, foi identificada a necessidade de regular a velocidade dos motores baseado no número de pontos encontrados no trajeto, garantindo uma locomoção mais suave.

Com o desenvolvimento do módulo de detecção de pistas, o robô passou a ter uma autonomia maior, independendo da

necessidade específica de rotas pré-determinadas. Porém, quando ambos os módulos atuaram juntos, o robô atingiu maiores velocidades, sem perder sua rota no trajeto.

O desempenho da locomoção do robô também é influenciado pela luz ambiente. Em alguns testes, houveram objetos de cores similares ou iguais a cor da rota utilizada, onde o robô identificou alguns pontos de semelhança, perdendo seu trajeto. Isto se tornou comum quando a rota utilizou de cores escuras, próximas da cor preta, e com um ambiente com uma grande concentração de sombras. Para tal, foram utilizadas cores claras para a confecção de um trajeto para testes.

O módulo de detecção de símbolos permite que o robô consiga identificar quando chegar a seu caminho. A princípio, este processo seria realizado via o reconhecimento de QR Code, porém, foi identificado uma maior complexidade neste processo, e comparado ao tempo final de entrega do projeto, a identificação de símbolos utilizando a biblioteca OpenCV se tornou uma saída viável. Desta maneira, o robô para ao identificar um número de círculos pré-determinados.

VI. CONCLUSÃO

O objetivo do projeto foi alcançado com êxito. Este consistia no desenvolvimento da autonomia robótica baseando-se na integração de um smartphone com sistema Android, utilizando sua câmera como o principal sensor externo.

As demais etapas do desenvolvimento foram concluídas completamente. Para a etapa de desenvolvimento da autonomia do robô, foram pesquisados métodos para seguimento autônomo, e identificado à necessidade da identificação de uma trajetória. Desta forma, foram aplicadas técnicas de visão computacional, utilizando a biblioteca OpenCV, a fim de construir os módulos de detecção de cores (para trajetórias previamente marcadas) e identificação de pistas (para localização de ambientes de locomoção, como corredores). Estes módulos demandaram um maior tempo de desenvolvimento, devido suas complexidades e ajustes para adaptação dos métodos ao projeto.

A última etapa consistiu no desenvolvimento de um sistema de reconhecimento de símbolos, necessário para indicar à aplicação de controle a chegada do robô a seu destino. De início, foi proposta a idéia de desenvolvimento de um sistema reconhecedor de QR Codes, porém, devido a complexidade e tempo escasso para desenvolvimento deste módulo, foi desenvolvido um sub-módulo capaz de reconhecer círculos, identificando sua chegada com um número pré-determinado destes símbolos.

VII. TRABALHOS FUTUROS

Como trabalhos futuros, sugere-se aprimorar as técnicas de visão computacional, para permitir que o robô consiga se adaptar a novos ambientes fechados, como faculdades e outros locais públicos, além de métodos para melhorias de detecção de linhas.

Também se sugere a implementação de novos recursos, como a possibilidade de locomoção em ambientes abertos, utilizando o GPS como sensor externo, aproveitando a utilização do módulo de lane detection para locomoção autônoma.

O robô também pode sofrer adaptações em sua estrutura, para melhor locomoção, com a inclusão de novos sensores para permitir a interpretação do mundo externo. Entre elas, a inclusão do sensor ultrassônico para identificação de obstáculos a frente, além do sensor laser, para permitir detecção de degraus a frente.

REFERÊNCIAS

[1] G. Orth, A. Desenvolvimento de um sistema de visão para medir o desgaste de flanco de ferramentas de corte. Dissertação de Mestrado, Universidade Federal de Santa Catarina, 2001.

[2] Trindade, F. Rehem, A. Técnicas de Visão Computacional para Rastreamento de Olhar em Vídeos, 2009.

[3] Rethink Robots. How Baxter is Different, 2008. <Disponível em http://www.rethinkrobotics.com/index.php/products/how-baxter-is-different>.

[4] LEGO. LEGO MINDSTORMS NXT Hardware Developer Kit, 2006.

[5] DA SILVA, LUCIANO ROTTAVA. Análise e programação de robôs móveis autônomos na plataforma Eyebot. Disponível em: <http://www.das.ufsc.br/~rottava/download/dissertacao.pdf>.

[6] RAMOS, B. RAMOS, R. CALADO, S, Automatic Guided Vehicle. Disponível em: <http://users.isr.ist.utl.pt/~pjcro/cadeiras/api0405/pdf_files/G03_SEM.pdf>

[7] CUBEK, RICHARD. A Color Blob Based Robot Vision, 2010. Disponível em: <http://amser.hs-weingarten.de/cms/administrator/components/com_intranet/uploads/paper/70/pa_cubek.pdf>.

[8] MELO, Leonimer Flávio de. Proposta de Simulador Virtual para Sistemas de Navegação de Robôs Móveis Utilizando Conceitos de Prototipagem Rápida, 2007. 293p. Tese (Doutorado) – Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas.