CENTRO UNIVERSITÁRIO SENAC

Guilherme Giacchetto Moreira

Mario Leandro Pires Toledo

Desenvolvimento de robôs semi-autônomos baseados em LEGO e NXT

São Paulo

2012

Guilherme Giacchetto Moreira
Mario Leandro Pires Toledo

Desenvolvimento de robôs semi-autônomos baseados em LEGO e NXT

Trabalho de conclusão de curso apresentado ao Centro Universitário Senac – Campus Senac, como exigência parcial para obtenção de grau de Bacharelado em Ciências da Computação.

Orientador: Fábio Roberto de Miranda

São Paulo
2012

Giacchetto, Guilherme M.; Toledo, Mario L. P.

Desenvolvimento de robôs semi-autônomos baseados em LEGO e NXT / Guilherme Giachetto Moreira – Mario Leandro Pires Toledo. – São Paulo, 2012. 75f.

Trabalho de Conclusão de Curso – Centro Universitário Senac – Bacharelado em Ciência da Computação.

Orientador: Fábio Roberto de Miranda

Alunos: Guilherme Giacchetto Moreira

Mario Leandro Pires Toledo

Título: Desenvolvimento de robôs semi-autônomos baseados em LEGO e NXT

A banca examinadora dos Trabalhos de Conclusão de Curso em sessão pública realizada em 10/12/2012 considerou os candidatos:

( ) aprovados ( ) reprovados

1) Examinador (a): _____

2) Examinador (a): _____

3) Presidente: _____

AGRADECIMENTOS

Antes de tudo, agradecemos a nossos pais, pelos ensinamentos sobre a vida, e por todas as oportunidades oferecidas.

Agradecemos a nosso orientador Fábio Miranda, que nos ajudou neste processo de desenvolvimento do projeto, e nos auxiliou com grande profissionalismo.

Agradecemos a professora Joyce Bevilacqua, que nos ensinou muito mais do que ementas acadêmicas.

Agradecemos à Bárbara Giacchetto, pela ajuda oferecida na prototipagem do robô.

Gostaríamos também de agradecer a todos que de alguma forma contribuíram comeste projeto, e nos deram forças para terminar este trabalho. Entre estas pessoas, um agradecimento especial para Evelyn Carlin e Munyra Guarnieri, cujas palavras de apoio serviram como inspiração durante este processo criativo.

A todos vocês, nosso muito obrigado.

"Antes de embarcar em uma jornada de vingança, cave duas covas."

(Confúncio)

RESUMO

Este trabalho têm como propósito o desenvolvimento de um robô semi-autônomo do modelo LEGO NXT, que utiliza de um smartphone baseado no sistema Android como o principal sensor ao meio externo, e principal fonte de processamento. A câmera do smartphone é utilizada como receptor de imagens, e são aplicados métodos de visão computacional para atribuir uma autonomia ao robô.

Palavras chave: Robótica, Android, NXT, Visão Computacional, Inteligência Artificial

ABSTRACT

This work has implications for the development of a semi-autonomous robot LEGO NXT model, which uses a smartphone based on Android system as the primary sensor to the external environment, and main source of processing. The smartphone camera is used as the image receptor, and apply computer vision methods to assign an autonomy to the robot.

Keywords: Robotics, Android, NXT, Computer Vision, Artificial Intelligence.

SUMÁRIO

1. Introdução

A robótica promove diversos benefícios à sociedade. Os robôs podem, normalmente, realizar trabalhos repetitivos e enfadonhos para um humano, dificilmente cometendo erros. A níveis industriais, um robô é um trabalhador incansável, usado em linhas de produção, reduzindo custos e aumentando a produtividade do processo.

Segundo DA SILVA(2003) “a robótica esta com desafios cada vez maiores. Não bastam apenas serem capazes de desviarem de obstáculos e evitarem colisões. Os robôs móveis de hoje devem ser dotados de alguma "inteligência" que lhes possibilite, por exemplo, reconhecer um dentre vários objetos.” (DA SILVA, LUCIANO ROTTAVA. Análise e programação de robôs móveis autônomos na plataforma Eyebot).

O ato de desenvolver um robô autônomo é complexo em si, e requer o envolvimento de muitas áreas da computação, desde arquitetura de software para a composição do controle lógico do robô, até Inteligência Artificial, para a composição de algoritmos para automação do mesmo. Alguns destes processos requerem conhecimentos específicos sobre robótica, principalmente para a arquitetura de protótipos robóticos e o tratamento de linguagens de baixo nível. Trabalhar com projetos robóticos proporciona o entendimento destas demais áreas.

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 ações baseando-se nos resultados destes sensores.

O robô Baxter (2008, Rethink Robots), por exemplo, 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, operações maquinarias e muitas outras atividades.

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.

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 visual (www.nsk.com).

Diferente do robô da empresa NSK, este projeto oferece uma solução de baixo custo para a implementação 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 a movimentação.

Neste trabalho, utilizamos um aparelho smartphone como o principal sensor do robô construído. A escolha de um smartphone se deu devido a evolução tecnológica que os aparelhos celulares tiveram.

O iPhone é considerado o primeiro smartphone atual, com capacidade de processamento suficiente para suportar a execução de múltiplas tarefas. Desde seu lançamento, em 2007, o iPhone têm aberto portas para diversas possibilidades no mercado mobile. Devido a seu sistema de aplicativos, desenvolvedores que estivessem aptos a criação de pequenas ferramentas para a plataforma, puderam desenvolver um novo nicho de mercado na área de desenvolvimento de software.

Como concorrente direto da Apple, em 2008, a Google lançou seu sistema Android, para competir com o iOS (o sistema da Apple). Seu sistema baseado na plataforma Linux deu mais liberdade a seus desenvolvedores, e permitiu ser utilizado em diversos modelos de celulares.

A evolução eminente dos smartphones garantiu que os aparelhos se tornassem um dispositivo essencial para profissionais que necessitem de informações a fácil acesso, além de recursos de processamento e outras utilidades práticas. Hoje em dia, os smartphones possuem capacidades comparáveis a notebooks de fácil acesso. O Galaxy SIII, por exemplo, é um dos smartphones atuais mais avançados, quando comparado a suas especificações técnicas. Quando comparado a primeira geração do iPhone, é possível ver a grande evolução que os smartphones sofreram nos últimos anos.

	iPhone (primeira geração)	**Samsung Galaxy i9300 SIII**
Conexões de rede sem fio	2G (GSM 850 / 900 / 1800 / 1900)	2G (GSM 850 / 900 / 1800 / 1900), 3G (HSDPA 850, 900, 1900, 2100) e 4G (LTE)
Resolução do display	320x480 (~165 ppi pixels de densidade).	720x1080 (~306 ppi pixels de densidade).
Memória	Sem slot de memória externa, e 4/8/16 GB de memória interna.	Com slot de memória externa para até 64 GB, 16/32/65 GB de memória interna, e 1GB de memória RAM.

Câmera	2MP e resolução de 1600x1200 pixels.	foco automático, flash de LED, 8MP e resolução de 3264X2448 pixels.
Vídeo	Não grava.	Gravação de vídeos em 1080p, a 30 frames por segundo.
CPU	412 MHz ARM 11	Quad-core 1.4 GHz Cortex-A9
GPU	PowerVR MBX	Mali-400MP

Tabela 1 – Comparação entre um iPhone da primeira geração, e um Galaxy SII. (http://www.gsmarena.com/)

Neste projeto, foi utilizado um *smartphone* com sistema Android, devido a facilidade para desenvolvimento com as ferramentas disponibilizadas pela Google, e também pela compreensão da linguagem de programação Java, uma vez que aplicativos Android são construídos com esta linguagem. A câmera do celular, além de seu GPS embarcado, permite que o robô consiga ampliar suas capacidades sensoriais. Além disto, todo o processamento é passado a ser feito no *smartphone*, que concentra toda a parte lógica do robô.

1.1 Objetivos

Este trabalho tem como propósito o desenvolvimento de um robô semi-autônomo, que utilizará um smartphone como principal fonte de processamento, além de ser o principal sensor ao meio externo, utilizando sua câmera como guia para se orientar por trilhas ou caminhos, a fim de se chegar a um destino.

O robô utilizado neste trabalho é do modelo LEGO NXT. Seu protótipo foi arquitetado para que o robô conseguisse se locomover em pisos retos, sem muitas variações de terreno. O robô é conectado via *bluetooth* a um

smartphone com o sistema Android embarcado. A utilização de um celular com Android se deu devido a facilidade com desenvolvimento para o mesmo, além do domínio sobre a linguagem Java.

Este tipo de projeto pode ser utilizado em soluções para diversos problemas da sociedade. Entre eles, o auxílio a pessoas com deficiência visual para se locomoverem em ambientes fechados, onde o robô pode ser capaz de guiar pessoas com problemas de visão em ambientes fechados.

O uso dos recursos do smartphone favorecem para a implementação dos variados sensores do robô. Entre eles, a possibilidade do tratamento das imagens captadas pela câmera, e a capacidade de conexão via Bluetooth entre o robô e o celular.

Os estudos feitos neste projeto promovem avanços para a integração entre dispositivos móveis e arquiteturas fechadas, possibilitando a construção de robôs com sensores mais complexos, baseados nos recursos integrados a smartphones atuais.

1.2 Organização do Texto

O capítulo 2 de fundamentos conceituais introduz os conceitos referentes a robótica, incluindo sua história até os robôs mais atuais. Também são mostrados conceitos para automação robótica, e os diferentes tipos de robôs autônomos.

Ainda neste capítulo, são mostradas as diferentes aplicações robóticas, com modelos de robôs para áreas específicas. Também são mostrados alguns tipos de sensores que auxiliam os robôs a realizarem suas tarefas. Por fim, também é falado sobre os smartphones e suas capacidades, além de um detalhamento sobre o sistema Android.

O capítulo seguinte fala sobre o desenvolvimento do projeto, desde sua concepção até a montagem do protótipo do robô. Também é detalhado o

desenvolvimento da aplicação de controle, usada no smartphone, detalhado seus demais módulos e funcionalidades.

No capítulo 4, são descritos os resultados obtidos, e no capítulo 5, é descrita a conclusão e os possíveis trabalhos futuros.

2. Fundamentos Conceituais

Neste trabalho, que trata da construção de um robô semi-autônomo capaz de se locomover em ambientes fechados através de técnicas de visão computacional, são empregados conceitos básicos sobre robótica, utilizados para a construção de um robô do tipo land rover – um robô capaz de se locomover em terrenos planos, com pouca variação.

Também são empregadas técnicas de automação robótica, para permitir que o robô tome decisões a partir da resposta de seus sensores. Além disto, também são utilizadas técnicas de visão computacional sobre o desenvolvimento para sistemas Android, permitindo o robô de se orientar através da câmera do celular.

A seguir, serão expostos conceitos relevantes de cada um destes assuntos, de maneira a permitir contextualizar melhor o trabalho.

2.1 Evolução da Robótica

Leonardo Da Vinci é lembrado como o criador do primeiro robô em forma humana construído em 1495, tendo como base seus estudos sobre a anatomia humana. Estes estudos sobre anatomia permitiram o surgimento de diversos exemplares de bonecos com articulações mecânicas, que moviam as mãos, braços e pernas, e alguns até mesmo tocavam instrumentos.

Em 1801, o francês Jacquard desenvolveu um tear controlado por cartões perfurados, levantando-os conforme uma programação pré-estabelecida Estes teares permitiram o avanço na indústria têxtil, onde as demais fábricas procuraram não só equipar-se com teares mecânicos, mas também investiram no desenvolvimento de novas máquinas para reproduzir automaticamente determinadas tarefas.

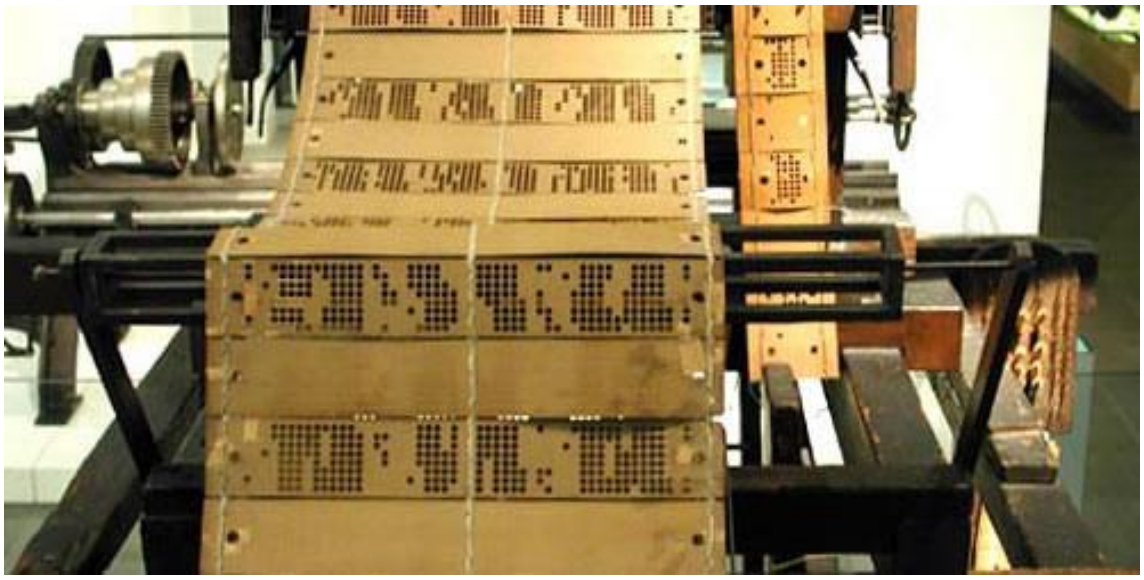


Figura 3 - O tear programável de Jacquard (http://odur.let.rug.nl/koester/musicbox/jacquard2.jpg).

Somente em 1924 que surgiu o primeiro robô mecânico, desenvolvido pelo engenheiro elétrico Ron J. Wensley. O dispositivo podia, através de um sistema de telefonia, ligar ou desligar qualquer sistema que estivesse conectado a ele. Três anos depois, Wensley criou o Televox, um pequeno robô com aspecto humano, que conseguia executar comandos básicos, como sentar-se e cumprimentar de acordo com os comandos de seu operador.

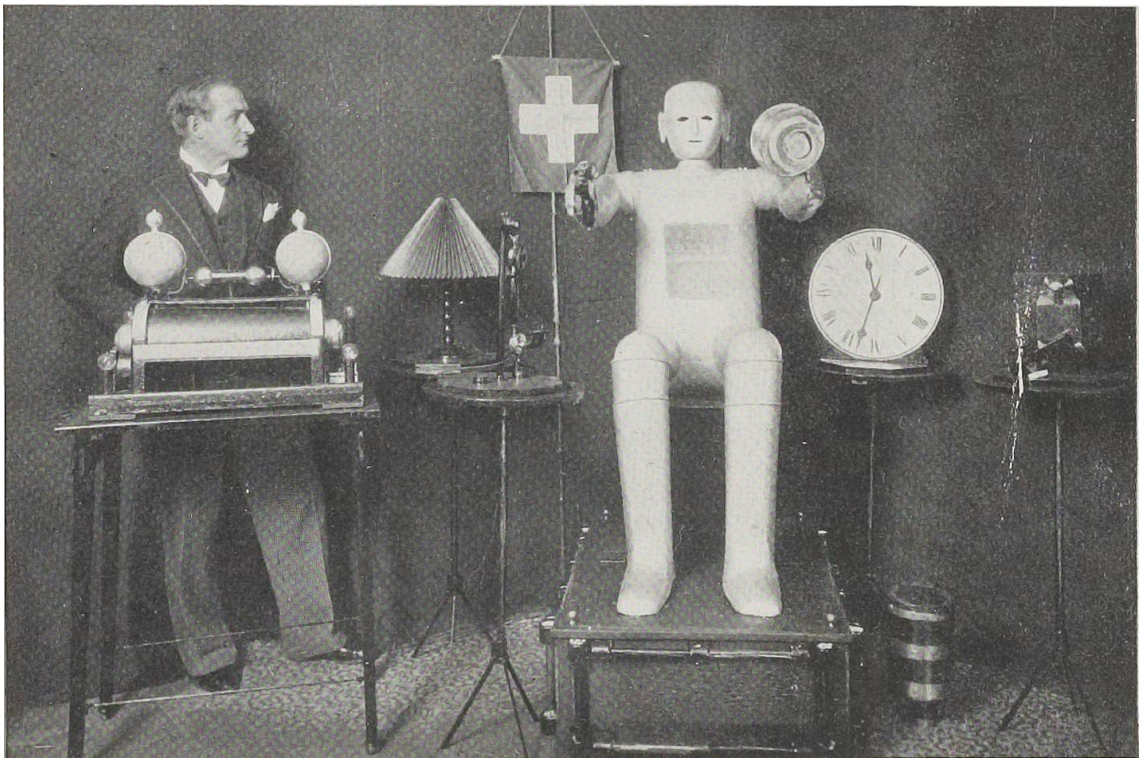


Figura 4 - Televox, o robô desenvolvido por Wenlex, capaz de executar funções básicas, de acordo com seu operador. (http://cyberneticzoo.com/wp-content/uploads/Televox-Wendling1930(1).jpg)

Em 1940 e 1983, o escritor Isaac Asimov publicou os livros “I, Robot” e “Machines That Think” onde descreveu as leis da robótica como parte de um universo futurista. O objetivo destas leis seriam proteger os seres humanos de qualquer dano que possa ser causado pelas maquinas inteligentes, e acabaram servindo como os primeiros resquícios da inteligência artificial de um robô. São elas:

1. Lei Zero: Um robô não pode fazer mal à humanidade e nem, por inação, permitir que ela sofra algum mal.
2. Primeira Lei: Um robô não pode fazer mal a um ser humano nem, por inação, permitir que algum mal lhe aconteça.
3. Segunda Lei: Um robô deve obedecer às ordens dos seres humanos, exceto quando estas contrariem a primeira lei.
4. Terceira Lei: Um robô deve proteger sua integridade física, exceto quando isto contrariar a primeira ou a segunda lei.

Os robôs tele operados continuaram a serem desenvolvidos até 1954, onde George Devol projetou o primeiro robô automático, um braço mecânico, capaz de ser programado para repetir tarefas como agarrar ou levantar peso. A partir deste braço mecânico, foi desenvolvido o Unimate, o primeiro robô comercial de uso industrial. Este robô começou a funcionar na linha de produção da General Motors, em 1961, e trabalhava pegando pedaços quentes de metal e colando as peças nos chassis dos carros.

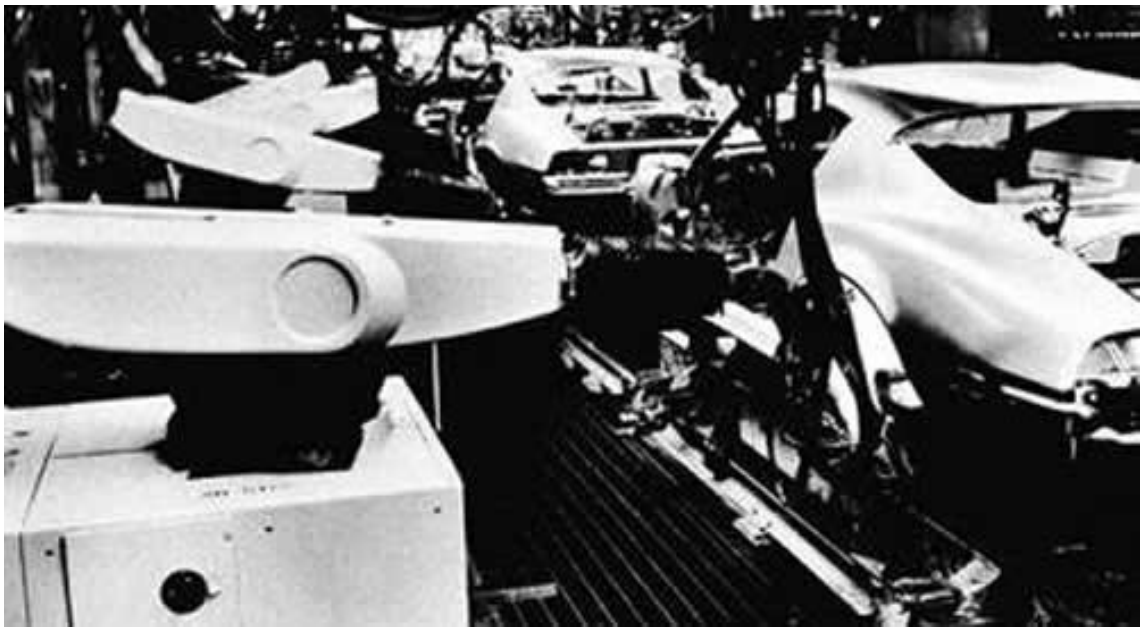


Figura 5 - O Unimate foi o primeiro robô comercial de uso industrial (http://robot.org/files/2011/10/unimate.jpg).

Em 1966, Joseph Weizenbaum lançou, no MIT, o primeiro programa de Inteligência Artificial. Chamado de Eliza, o programa permite que o usuário converse com o sistema, e o software, com apenas 200 linhas de código, emite uma resposta baseado em palavras-chaves do usuário. Este programa foi utilizado para terapias psiquiátricas, para pacientes contarem seus problemas através do software.

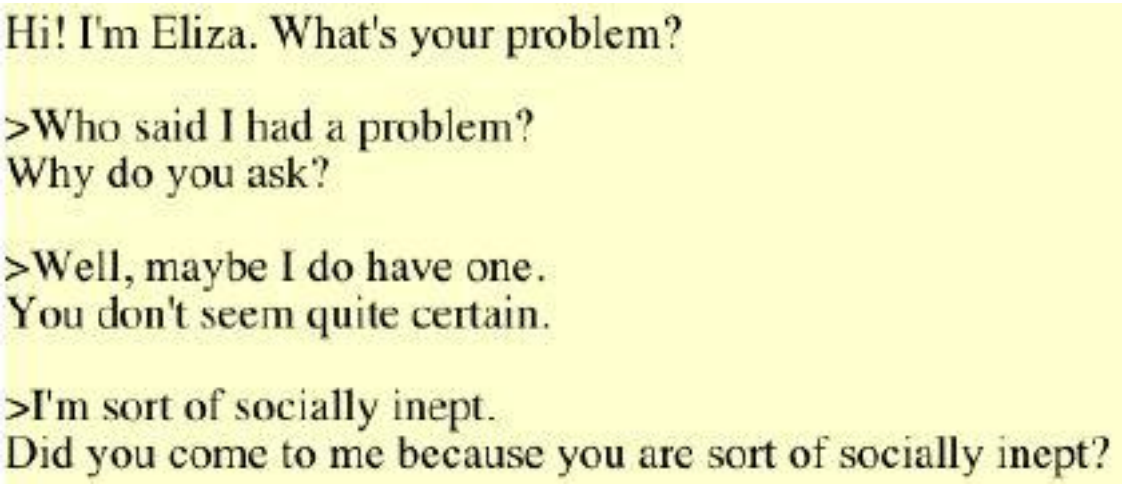


Figura 6 - Exemplo de conversa com Eliza, o primeiro programa de Inteligência Artificial (http://nickm.com/if/elist_images/eliza.jpg).

Em 1986, Rodney Brooks apresentou a arquitetura Subsumption desenvolvida para atuar em um ambiente dinâmico, afirmando que “o mundo deve ser seu próprio modelo” (Brooks, R. A., "Achieving Artificial Intelligence Through Building Robots", MIT AI Lab Memo No. 899, May 1986). Em 1987, Ronald Arkin desenvolveu a arquitetura hibrida AuRA (Autonomous Robot Architecture), para navegação robótica, baseado em técnicas tradicionais de inteligência artificial.

A NASA passou a pesquisar robôs para auxílio em missões espaciais. Grande parte destes robôs se baseiam em braços robóticos, para coleta de materiais espaciais, ou land rovers, capazes de se locomover por terrenos diferenciados, coletando o máximo de informação.

Em 1997, a NASA enviou o robô Sojourner para Marte, com a missão de exploração dos terrenos do planeta, medindo as propriedades do solo e realizar analises químicas de suas rochas. O veículo possui 6 rodas para locomoção, com uma câmera frontal para captação de imagens, e uma placa solar para armazenamento de energia.

Em 2012, a NASA enviou o robô Curiosity, também a Marte, com o objetivo principal de determinar se Marte, em algum momento, já suportou vida em seu terreno, além de determinar a existência de água, e estudar o clima e a

geologia do planeta, ajudando a preparar futuras explorações humanas no local.

Figura 7 - O robô Curiosity, desenvolvido pela NASA para exploração em Marte (http://cdn.theatlantic.com/static/mt/assets/science/curiosity615.jpeg).

Para completar seus objetivos, a Curiosity possui uma série de instrumentos e sensores, a fim de captar informações e conseguir completar sua missão. São eles:

- Mast Camera: A Mast Camera prove multiplos spectros e imagens em cores reais, compostas por duas câmeras. Estas câmeras conseguem capturar imagens em 1600x1200 pixels de resolução, e vídeos de até 10 frames por segundo, em uma resolução de 1280x720 pixels;
- Chemical Camera: A ChemCam é um instrumento composto de dois sensores: o LIBS, um spectoscópio a laser, provê a composição elemental das rochas e do solo; O RMI, um micro telescópio, provê imagens em alta resolução dos locais que o LIBS identifica;
- Navigation Cameras: A Curiosity possui dos pares de câmeras, capazes de capturar imagens em preto e branco, a fim de auxiliar a locomoção no terreno;

- Estação de monitoramento do ambiente: Chamada de REMS, a estação de monitoramento do ambiente comprime diversos instrumentos para análise do ambiente de Marte, tais como a umidade, pressão, temperatura, velocidade do vento e radiação ultravioleta;
- Câmeras para detecção de perigo: O robô possui 4 pares de câmeras chamadas Hazcams, que capturam imagens em preto em branco, a fim de detectar obstáculos a frente, além de buracos localizados em seu caminho. Estas câmeras auxiliam a automoção do robô;
- Braço robótico: A Curiosity possui um braço robótico, capaz de pulverizar pedras encontradas no local, e entregar a sensores internos, para análise de suas substâncias;

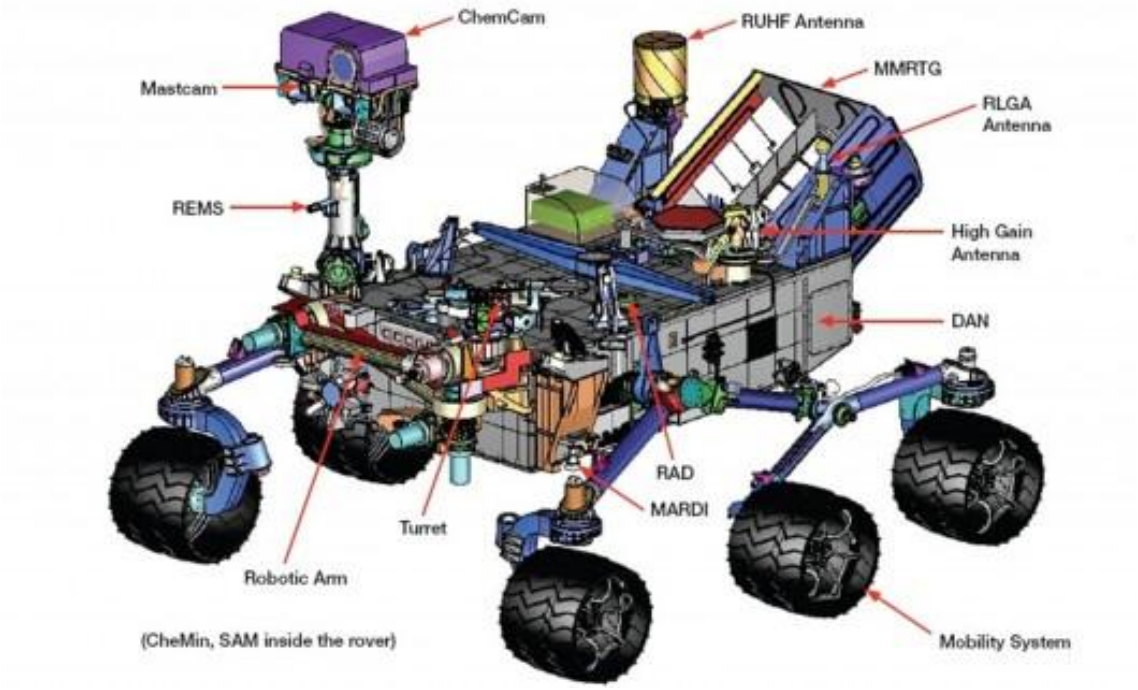


Figura 8 - Disposição dos sensores do robô Curiosity. (http://images.gizmag.com/inline/curiosity-third-drive-9.jpg)

2.2 Tipos de robôs

Os robôs podem ser divididos em diferentes tipos, onde cada um se encaixa com propósitos diferentes, arquitetados de maneiras distintas. Este projeto trabalha com um robô móvel, sobre locomoção em ambiente plano. O

que se segue são detalhes a respeito destes tipos de robôs, e suas aplicações no mundo real.

2.2.1 Robôs de mobilidade terrestre

Um robô móvel é um dispositivo autônomo capaz de interagir com um ambiente especifico. A robótica móvel tem como objetivo dotar um robô de autonomia suficiente para realizar determinada tarefa através da utilização de informações pré-estabelecidas, informações dos sensores, inteligência artificial e programação lógica.

Dispositivos robóticos terrestres utilizam o solo para se locomover, e geralmente são utilizados em ambientes específicos como escadas, terrenos acidentados ou subidas. Estes robôs podem ser divididos em categorias específicas, que variam de acordo com a locomoção de cada robô.

2.2.2.1 Robôs com rodas

Os robôs com rodas são os mais utilizados, pois seu desenvolvimento é mais simples, não necessitando de hardware tão complexo quanto o de um robô que se utiliza de pernas móveis. Os robôs que baseiam seus movimentos em rodas podem ter duas, três ou quatro rodas. Para máquinas acima de quatro rodas, apenas duas delas são usadas para controlar a direção do robô.

O sistema de direção em robôs com duas rodas é mais simples que os demais: consiste no uso de um motor para cada roda, acionando ambos na mesma direção quando necessário ir para frente ou para trás. Quando necessário realizar uma curva sobre o próprio eixo, é necessário girar um motor na direção inversa do outro, até completar o grau deseja do da curva.

Estes robôs também possuem um apoio traseiro, de ponta arredondada, solta de forma a se orientar na direção que o robô seguir.

O Axel Rover é um exemplo de robô com duas rodas. Desenvolvido pela NASA, o robô permite uma maior versatilidade em seu movimento, e é usado para exploração de terrenos em Marte, em lugares que a Curiosity pode ter dificuldade de acesso.

Figura 9 - O Axel Rover, desenvolvido pela NASA para maior versatilidade na locomoção dos terrenos de Marte (http://www-robotics.jpl.nasa.gov/roboticImages/img811-413-browse.jpg).

No caso de robôs com três rodas, podem existir diferentes formas para o controle de direção. A forma mais comum se baseia na utilização de apenas 1 roda para o controle de direção, enquanto outras duas movimentam o robô para frente ou para trás.

Com quatro rodas, o mais comum é haverem duas rodas frontais para o controle de direção (ambas se movendo para a mesma direção), enquanto as rodas traseiras são usadas para mover o robô para frente ou para trás.

2.2.2.2 Robôs com esteiras

Os robôs que baseiam sua locomoção por esteiras são muito utilizados em terrenos acidentados. Este sistema constitui de uma sequência de peças de metal debaixo de uma roda, a fim de permitir um processo contínuo. Para rotação, as esteiras devem ser acionadas em direções inversas, a fim de girar o robô no próprio eixo.

Um exemplo destes robôs são os modelos EOD-Robots, construídos para poupar a vida de soldados no campo de batalha. Estes robôs são equipados com esteiras iguais a tanques de guerra, a fim de suportar grandes pesos em diferentes tipos de terrenos. Alguns destes robôs, por exemplo, são utilizados para o desarmamento de bombas.

Figura 10 - O EOD-Robot é um exemplo de robô com esteira, construído para poupar vida de soldados em batalha (http://www.almc.army.mil/alog/issues/MayJun08/Story_Images/ms435_imageN.jpg).

2.2.2.3 Robôs com pernas

Os robôs com pernas são os mais complexos a serem desenvolvidos. Esta categoria pode ser dividida em dois tipos de robôs: os bípedes e os quadrúpedes.

Os robôs bípedes normalmente são projetados para simular a aparência humana. A maior complexidade na projeção destes robôs esta no equilíbrio do mesmo, requerendo diversos sensores ativos para mantê-lo em pé. A locomoção destes robôs é baseada no acionamento de uma perna de cada vez, muito parecido com a locomoção humana.

Entre estes robôs esta o Asimo, projetado para ter uma aparência humana, e capaz de auxiliar as pessoas nas suas mais diferentes necessidades.

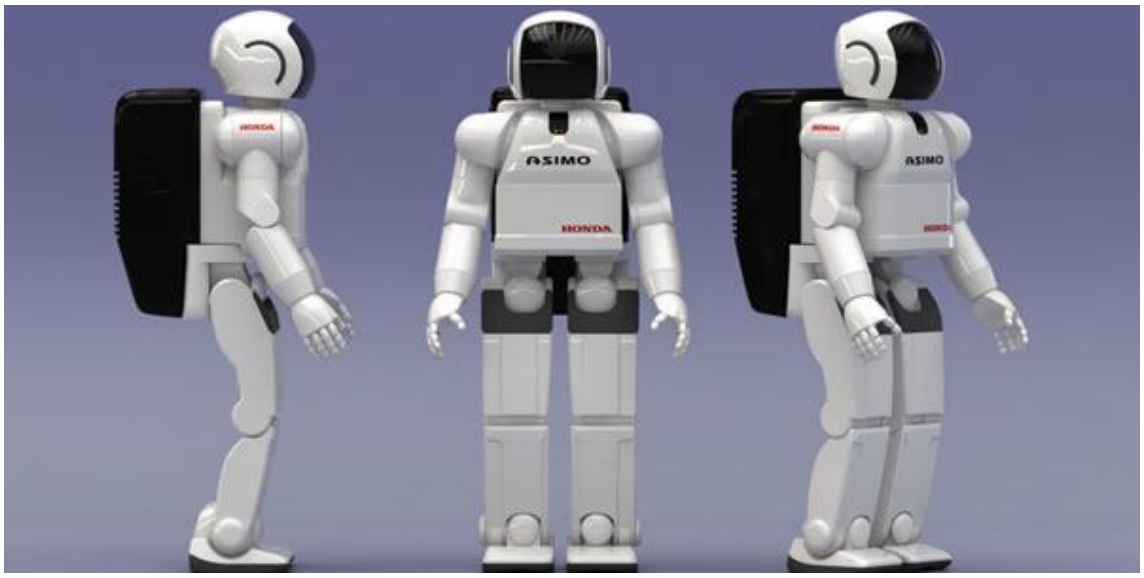


Figura 11 - O Asimo é um exemplo de robô com duas pernas, projetado para possuir uma aparência próxima da humana (http://www.stevesideas.net/blog/wp-content/uploads/2010/02/asimo_v2.jpg).

No caso dos robôs quadrúpedes (ou com mais pernas), a preocupação com o equilíbrio é menor. Para sua locomoção, uma perna dianteira permanece no chão como apoio, enquanto a outra segue em frente. O mesmo acontece com as pernas traseiras, mas com os lados trocados.

Um exemplo de robô quadrúpede é o BigDog, projetado para fins militares. Este robô é capaz de carregar 150 quilos, possibilitando a movimentação de tropas em diversos terrenos.

Figura 12 - O BigDog é um exemplo de robô quadrúpede, usado para carregar equipamentos militares (http://www.bostondynamics.com/img/BigDog_ClimbRubble.png).

2.3 Aplicações da Robótica

A Robotics Industries Association (RIA) descreve um robô como: um manipulador programável multifuncional capaz de mover materiais, partes, ferramentas ou dispositivos específicos através de movimentos variáveis programados para realizar uma variedade de tarefas.

No século 20 a robótica começou a tomar força, por motivos principalmente industriais pela necessidade de aumentar a produção e melhorar a qualidade dos produtos, somente em 1961 foi instalado o primeiro robô industrial Unamites desenvolvido por George Devol e Joe Engleberger.

Atualmente os robôs possuem diferentes formas e variadas aplicações e podem ser divididos em quatro grandes áreas de aplicação: robôs industriais, robôs de serviço, robôs pessoais, robôs educacionais.

2.3.1 Robôs Industriais

A aplicação mais popular dos robôs está nas indústrias, onde possuem a capacidade de movimentos similares ao braço humano e são aplicados em trabalhos como soldagem, pintura e outros demais trabalhos de risco.

Figura 13 - Exemplo de robô industrial, usado em linha de produção (http://davidthach.edublogs.org/files/2010/04/INDUSTRIAL-ROBOTS.jpg).

Os AGV's (Automated Guided Vehicle) são exemplos de robôs autônomos, usados para locomoção interna de fábricas, carregando cargas pesadas, sem a intervenção humana.

Figura 14 - Exemplo de robô AGV, que utiliza referência óptica para locomoção (http://www.koroberi.com/images/rmt/ADAM_From_RMT_Robotics.jpg).

Para este tipo de navegação autônoma, os AVG's podem ter incluso diversos tipos de sistemas de referência, permitindo a detecção de padrões de guias e caminhos. Segundo RAMOS e CALADO (RAMOS, B. RAMOS, R.

CALADO, S, Automatic Guided Vehicle), estes sistemas podem ser classificados da seguinte forma:

- Referência Óptica: É uma solução de baixo custo, e muito utilizada. Consiste na marcação do trajeto através de uma linha no chão, e no uso de sensores ópticos para detecção da linha. Isto permite que o robô siga seu trajeto, sempre identificando a linha por suas características próprias (tais como espessura e cor);
- Referência Indutiva: Consiste em condutores metálicos, embutidos no chão para formar um trajeto, detectável pelo robô, que utiliza um sensor magnético. Este sistema é viável, porém, possui maior complexidade em sua implementação devido a locais que existam muitos materiais ferrosos ou fontes de campo magnético;
- Referência por Laser: Consiste em refletores espalhados pelo edifício. Um sistema emissor/receptor Laser emite e capta as reflexões e determina continuamente a sua posição. Estes sensores medem a distância e o ângulo robô, garantindo sua exata posição;
- Referência Inercial: Utiliza um giroscópio para medição de desvios, comparando o percurso armazenado na memória do robô. Para seu uso, são instalados pares de imãs no chão ao longo do trajeto desejado, em distâncias regulares;
- Referência por GPS: Utiliza um dispositivo GPS para saber a localização Global do robô. Este tipo de sistema é utilizado apenas em robôs que se locomovem em ambientes externos;

2.3.2 Robôs de Serviço

Desenvolvidos para auxiliar os seres humanos em tarefas especificas como na inspeção de tubulações, cirurgias delicadas, vigilância e monitoramento, limpeza doméstica ou sistemas de defesa. Estes robôs foram feitos para serem usados em conjunto com trabalhos efetuados por seres humanos, e devido a isto, possuem uma aparência mais próxima da humana. Porém, são de custos elevados, devido a possuírem diversos tipos de

sensores, e manterem sua arquitetura fechada, não permitindo sua modificação para trabalhos que não estejam previamente programados.

O robô Baxter, da empresa Rethink Robotics, se encaixa neste modelo. O robô é capaz de executar diversas tarefas humanas que exigem repetição. Para sua automoção, são inclusos diversos tipos de sensores, como detectores de peso e sensores de proximidade humana. Este robô pode ser usado em trabalhos como empilhamento e desempilhamento, operação maquinaria, empacotamento e outras atividades que exijam repetição de esforço.

Outro exemplo de robô de serviço é o chamado Da Vinci. Desenvolvido pela empresa americana Intuitive Surgical, o robô é capaz de realizar cirurgias com mais de um instrumento, com incisões de poucos milímetros. Este robô não é autônomo, e necessita da operação de um cirurgião, que visualiza em seu display os locais onde as incisões são realizadas.

Figura 15 - O robô Da Vinci, utilizado a serviço da medicina (http://2.bp.blogspot.com/-gBVAk6PS-HA/T8_pBcX64FI/AAAAAAAABvk/EK7UQ3aMK34/s1600/davinci-robotic-surgery.jpg).

2.3.3 Robôs Educacionais

Existem robôs que são vendidos com intuitos educacionais, utilizados em instituições de ensino para que alunos tenham um aprendizado básico sobre robótica, ou utilizados para pequenos projetos, normalmente em fases

experimentais como um pequeno protótipo, ou usados em projetos acadêmicos.

Entre estes robôs, esta o ASURO, desenvolvido para propósitos educacionais pelo Instituto de Robótica e Mecatrônica do Centro Aeroespacial alemão. Trata-se de um robô simples de baixo custo, muito flexível, com estrutura física aberta, utilizando a linguagem C como linguagem de programação.

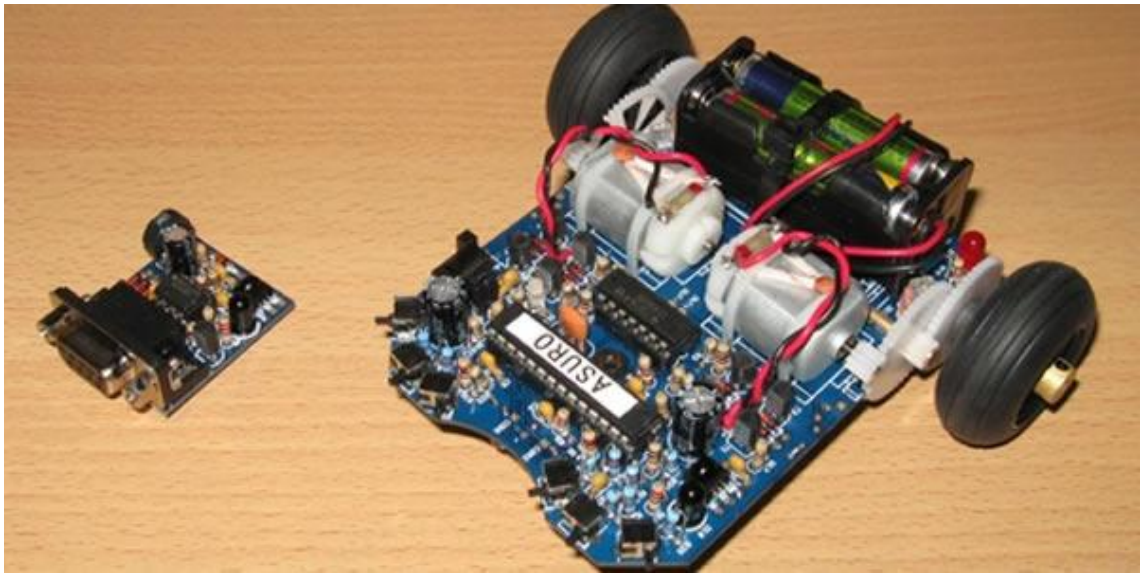


Figura 16 - O ASURO é um pequeno robô com propósitos educacionais (http://letsmakerobots.com/files/field_primary_image/Asuro_small.jpg).

O ASURO, assim como outras arquiteturas, requer um conhecimento básico em eletrônica, necessário para trabalhar com seus circuitos, além de possuir poucos acessórios para comunicação com outros dispositivos, tornando-o limitado a outras necessidades.

A seguir, seguem detalhes sobre a linha de robôs LEGO Mindstorms NXT, utilizados neste trabalho.

2.3.3.1 LEGO Mindstorms NXT

LEGO Mindstorms NXT foi lançado comercialmente em 2006, resultado de uma parceria entre o Lab Media do Massachusetts Institute of Technology

(MIT) e o LEGO Group. Em 2009, foi lançada a versão 2.0, com diversos novos recursos, e uma nova variedade de peças.

O LEGO NXT permite que usuários que não tenham familiaridade com a área da robótica possam criar protótipos com maior facilidade. Suas peças possuem encaixes próprios, e seus kits possuem manuais de iniciantes, com exemplos de modelos a serem construídos.

Figura 17 - O modelo LEGO NXT é muito utilizado por usuários que não tenham conhecimento técnico na área da robótica (http://www.tuvie.com/wp-content/uploads/lego-mindstorms-nxt-2.0-robots4.jpg).

Os robôs do modelo LEGO NXT possuem um componente principal chamado de “Brick”. Esta peça retangular concentra toda a parte lógica do robô, e através de entradas para cabos RJ12, pode controlar até 3 motores e 4 sensores conectados a si. Além disto, a peça ainda possui um pequeno adaptador interno para conexão Bluetooth, permitindo a troca de informação com outros dispositivos que também possam utilizar comunicação Bluetooth.

Figura 18 - O componente "Brick", responsável pelo controle lógico do robô NXT (http://www.nxt8547.org/legonxt.jpg).

O firmware para o NXT é Open Source, isto é, aberta para o desenvolvimento. Juntamente com o kit de desenvolvimento para o NXT, são acompanhadas documentações para desenvolvimento a partir da Brick. São eles:

- Software Developer Kit (SDK), que inclui informações a respeito do driver USB e as referências em bytecode para interpretação da Brick;
- Hardware Developer Kit (HDK), que inclui a documentação e a esquematização para a Brick e seus sensores;
- Bluetooth Developer Kit (BDK), que inclui documentos dos protocolos usados para a comunicação Bluetooth;

O NXT acompanha um software básico de desenvolvimento, chamado NXT-G. Este software promove a criação de rotinas robóticas, através do uso de uma linguagem gráfica, onde o usuário pode arrastar blocos de código prontos, criando uma lógica simples para o robô.

Porém, o NXT-G é limitado para qualquer tipo de lógica que não esteja previamente programada. Devido a isto, existem firmwares criados para utilização de outras linguagens para desenvolvimento de aplicações de controle do NXT. Entre estas firmwares, esta a leJOS, que permite o uso da linguagem

Java para desenvolvimento da lógica do robô. A leJOS ainda inclui uma biblioteca previamente programada em Java, que auxilia no desenvolvimento de funções básicas do robô, como navegação, mapeamento ou locomoção.

```
import lejos.nxt.Motor;
import lejos.nxt.Button;
public class Example{
  public static void main(String[] args){
    Motor.A.forward();
    Button.waitForPress();
    Motor.A.backward();
    Button.waitForPress();
    System.exit(1);
  }
}
```

Figura 19 - Exemplo de código em Java utilizando LeJOS.

O LEGO NXT possui uma comunidade muito grande de hobbystas e estudantes que tendem a utiliza-lo para projetos pessoais ou acadêmicos. Um destes exemplos é o estudo realizado por cientistas chineses para o desenvolvimento de um controle autônomo de detecção de obstáculos e correção de trilhas (SHIH, Bih-Yaw. CHEN, Chen-Yuan. CHOU Wei-Chung. Obstacle avoidance using a path correction method for autonomous control of a biped intelligent robot, 2010). Neste estudo, um robô do modelo LEGO NXT é utilizado, e através do seu sensor de ultrassom, são captadas informações dos obstáculos a sua frente, a fim de implementar métodos para o melhor desvio destes obstáculos.

Neste projeto, foi utilizado um robô do modelo LEGO NXT devido a sua praticidade quanto a montagem de protótipos, além da possibilidade do desenvolvimento de códigos na linguagem Java, e sua capacidade de comunicação via Bluetooth.

2.4 Automação Robótica

Segundo SOUZA “A automação é um passo que veio após a mecanização, e constitui no uso de controle de sistemas, comando numérico (CDC), controladores lógicos programáveis (PLC) e informática (CAD, CAM, CAx) para controlar maquinaria industrial e processos industriais, reduzindo a

necessidade de intervenção humana.” (DE SOUZA, FELIPPE, Automoção Industrial e Robótica).

Robôs autônomos podem tomar decisões e realizar tarefas especificas em ambientes desestruturados sem a intervenção humana. Diferentes tipos de robôs podem ser autônomos em diferentes formas e níveis. Alguns robôs em utilizados em indústrias como da categoria AGV (Automated Guided Vehicle) possuí certa autonomia em ambientes fechados e controlados por marcações que transmitem ao robô informações relevantes para que possa determinar suas ações.

Existem diferentes tipos de automação, que introduzem técnicas diferentes para a realização de determinada tarefa:

- Robôs executores: repetem instruções sequenciais, como a pintura ou soldagem de uma placa;
- Robôs controlados por sensores: possuem malhas fechadas ou dispositivos sensoriais que possibilita interação com o meio externo. Estes robôs se baseiam em seus sensores para uma tomada de decisão, variando dependendo do resultado;

O que se segue são detalhes sobre sensores, e alguns exemplos dos demais tipos utilizados neste projeto.

2.4.1 Sensores

Os sensores são essenciais em robôs autônomos. Eles permitem que um dispositivo robótico móvel perceba e interaja com o ambiente em que se encontra. Normalmente, estes sensores servem para assimilar ao robô os sentidos humanos (visão, audição, tato, paladar e olfato).

Os robôs autônomos se baseiam nas informações captadas pelos sensores para auxiliar em sua tomada de uma decisão. Através dos sensores, é possível adquirir informações sobre o ambiente em que estão, de forma que possam executar suas ações previamente programadas.

Na maioria das vezes, os sensores são utilizados para ações como detecção de colisão, distância e desvio de obstáculos. Neste projeto, os sensores utilizados permitiram que o robô pudesse se locomover, identificando obstáculos a sua frente. Alguns destes sensores são detalhados a seguir.

2.4.1.1 Sensor Ultrassônico

O sensor ultrassônico é usado para fazer medições precisas entre sua posição e um objetivo em seu caminho. Também possuí um emissor e um receptor. O emissor emite um sinal sonoro que é refletido por um objeto e retorna para seu sensor (receptor), este processo é conhecido como eco. Conhecendo a velocidade do som (344 m/s no ar) é possível determinar a distância percorrida na formula abaixo onde v é a velocidade do som, t é o tempo de deslocamento e d metade da distância percorrida.

$$d = \frac{v * t}{2}$$

Fórmula 20 – Fórmula do cálculo da distância feito pelo ultrassom.

Este sensor necessita de um tempo mínimo de processamento para detectar o sinal sonoro refletido, não possuí um longo alcance, algumas matérias que absorvem a energia sonora diminuem a amplitude do eco e objetos pequenos, atenuados, moveis, podem contribuir para uma medição errada.

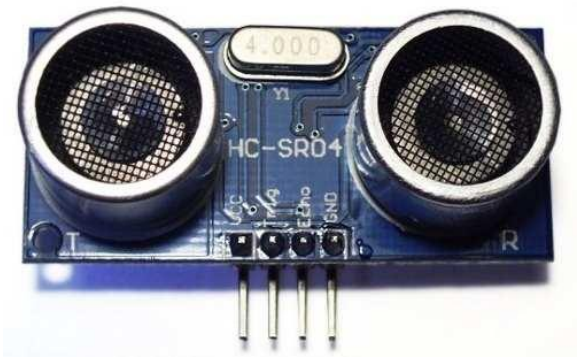


Figura 20 - Exemplo de um modelo de sensor ultrassônico (http://kemper.com/wordpress/2011/06/24/red-montando-uma-bancada/)

Este sensor é utilizado neste projeto para detecção de obstáculos a sua frente. O sensor ultrassônico é viável para diversos tipos de aplicações robóticas que necessitem de automoção.

2.4.1.2 Laser

Este sensor é utilizado para medir a distância e opera de maneira parecida com o sensor ultrasônico. A partir da emissão de um feixe de luz é determina a distância de um objeto a partir da medição do tempo de reflexão desta luz pelo objeto. Também pode ser utilizado junto com espelhos, onde controla-se o ângulo do espelho até que o feixe atinja o sensor, podendo então ser calculada a distância por triangulação.

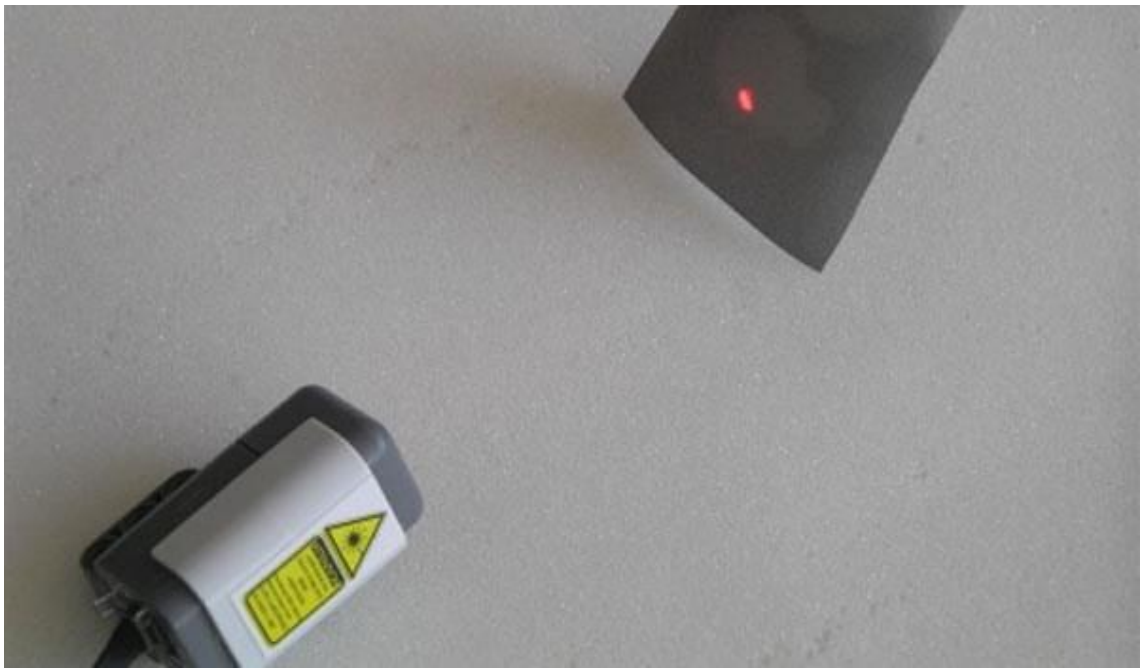


Figura 21 - O sensor laser (http://nxt-wiki.rjmcnamara.com/index.php?title=R.J.McNamara).

Este sensor detecta uma distância maior e é mais preciso que o sensor ultrasônico, pois possuí menor dispersão no sinal emitido. Contudo a velocidade da luz é mais rápida que a do som, exigindo um processamento mas rápido o que torna este sistema mais caro.

Este sensor não é utilizado neste projeto, uma vez que o sensor ultrassônico ja é utilizado para detectar obstáculos. Uma possível aplicação do

sensor de Laser seria a verificação de que o robô não corra o risco de cair de ambientes mais altos (como degrais, por exemplo). Porém, neste projeto, o robô deverá estar sempre se guiando por trilhas e caminhos seguros, descartando o uso deste sensor.

2.4.1.3 Receptor GPS

O Receptor GPS (Sistema Global de Posicionamento) tem como função determinar a posição sua posição atual, através da obtenção de informações de três ou mais satélites distribuídos ao redor da Terra. Desta forma, é possível determinar sua localização através do método de trilateração. O receptor calcula a diferença entre o tempo que um satélite envia um sinal e o tempo que o receptor o recebe. Usando as informações coletadas dentre vários sinais, o receptor calcula sua posição por triângulação. Receptores GPS possuem boa precisão normalmente variando em 15 metros.

O receptor GPS, porém, não consegue identificar posições fixas, como em lugares indoors. Este sensor só pode ser utilizado para ambientes abertos, inviabilizando seu uso neste projeto, uma vez que este robô trabalha apenas em ambientes fechados.

2.4.1.4 Câmeras de vídeo

Câmeras de vídeo são utilizadas para capturar imagens digitais. Usando este dispositivo junto com a área da visão computacional é possível extrair diversas informações sobre o ambiente, reconhecer padrões, cores, fazer medições.

Neste projeto, é utilizado a câmera de vídeo de um celular, a fim de capturar imagens a frente do robô, permitindo que o mesmo possa tomar decisões baseado nas imagens captadas. Este processo é melhor detalhado nos próximos tópicos.

2.5 Visão Computacional

A visão computacional desempenha papel de grande importância na robótica móvel. Quando se pretende utilizar o sentido da visão para a

localização e navegação, problemas maiores surgem. A integração de câmeras de vídeo ao já complexo hardware traz complicadores, por exemplo, o tratamento e processamento destas imagens capturadas vão requerer rotinas elaboradas para a extração de informações úteis (Orth, 2001).

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 em um conjunto de dados.

O sistema de visão computacional, segundo TRINDADE (Técnicas de Visão Computacional para Rastreamento de Olhar em Vídeos, 2009), 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 celular, e permitir a

tomada de decisão a partir das informações encontradas, visando determinar um caminho ou trajeto para ser seguido pelo robô. O que se segue são detalhes das técnicas de visão computacional utilizadas.

2.5.1 Color Blob Detection

Na area da visão computacional, blob detection se refere a um módulo visual para detecção de pontos e regiões que diferem em propriedades como cor ou brilho comparado as regiões ao seu redor.

No caso do Color Blob Detection, seu propósito se baseia na identificação de áreas e pontos de determinadas cores. Desta forma, é possível destacar regiões que possuam uma determinada tonalidade da cor vermelha, por exemplo.

Para que a detecção de regiões de cores semelhantes possa ser efetuada, é necessário transpor a imagem de seu padrão de cores RGB, que se baseia na união das cores vermelho, verde e azul para formar qualquer outra tonalidade, para o padrão HSV, que se baseia em matiz, saturação e brilho da mesma. Este procedimento permite separar as cores da imagem com maior facilidade.

Figura 22 - O uso do padrão HSV permite dividir melhor as cores em uma imagem (http://www.shervinemami.info/blobsSkinDetector.jpg).

Muitos robôs que necessitam reconhecer padrões de cores utilizam técnicas de Color Blob Detection, como é o caso do Projektairbeit (CUBEK, 2010), capaz de identificar objetos de uma determinada coloração, colocados em sua base.

Este projeto utiliza o método de Color Blob Detection para construir um módulo capaz de reconhecer trajetos lineares, baseados em uma cor específica.

2.5.2 Lane Detection

O sistema de Lane Detection nada mais é do que uma série de aplicações geométricas a partir do processamento em cima de imagens. Utilizando estas técnicas, é possível estimar a geometria de uma pista a frente, além de posicionar o robô em uma relação de distância quanto a geometria a sua frente e sua posição atual.

O Lane Detection permite destacar visualmente a pista ao qual o robô deve seguir. Este mesmo processo é utilizado para o desenvolvimento de métodos de segurança em pistas, além da redução do tempo de transporte. Existem exemplos de implementação do algoritmo de Lane Detection em dispositivos móveis para um processo automotivo, como o uso de um iPhone para detecção de pistas (REN, F. HUANG, J. TERAUCHI, M. JIANG, R. KETTLE, R. Lane Detection on the iPhone), capaz de reconhecer rotas em auto estradas.

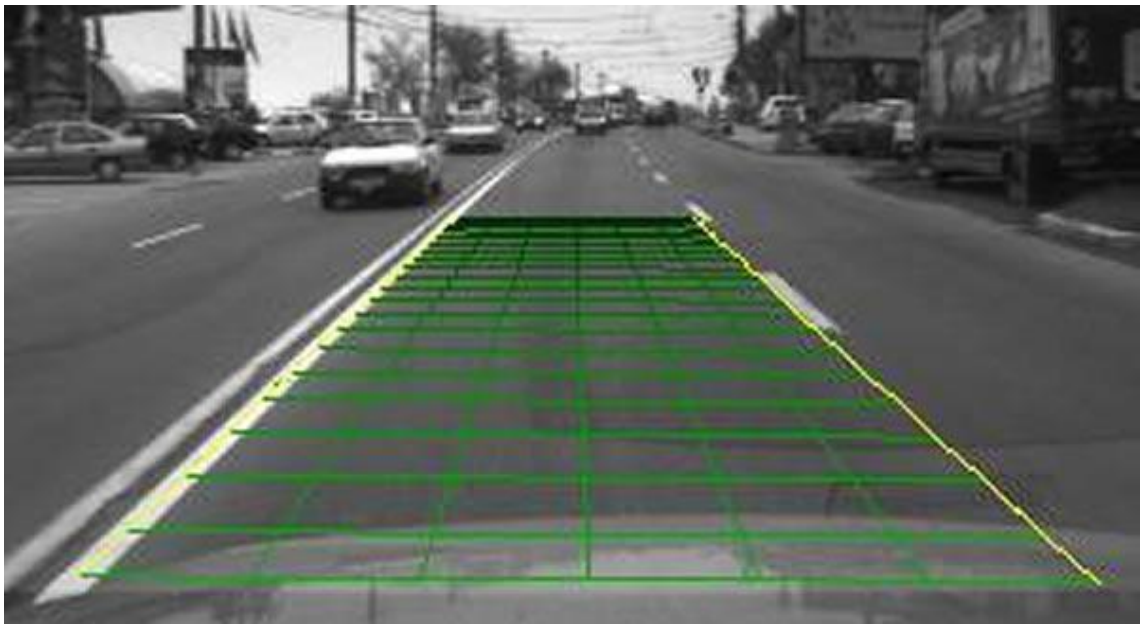


Figura 23 - O método de Lane Detection permite o reconhecimento de pistas (http://ac.utcluj.ro/tl_files/utcn/img/doc_images/image008.jpg).

O primeiro passo para o desenvolvimento da Lane Detection é a aplicação de filtros na imagem, para que a mesma seja transformada para uma escala de cinzas. Este processo permite mapear a imagem, e transforma-la para uma linguagem binária, analizando as possíveis rotas para detecção.

Para detecção das possíveis rotas, o primeiro passo se baseia na detecção das bordas da imagem. Existem diferentes algoritmos para detecção de bordas, mas neste projeto, foi o utilizado o Canny Edge Detector. Seu algoritmo analisa, antes de mais nada, a imagem inteira e seus níveis de cinza, atribuindo valores para cada um em uma matriz, calculando uma função para seus níveis.

Uma vez que a função esteja atribuída, é calculado a derivada da função

deste sinal, utilizando a função Gaussiana, para aproximar o operador que otimiza o produto do raio do sinal e localização.

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{d}{dx}G = f * G_x$$

$$f_y = \frac{\partial}{\partial x}(f * G) = f * \frac{d}{dy}G = f * G_y$$

Fórmula 2 - Derivada da função Gaussiana

A direção da borda é calculada a partir dos gradientes formados entre as diferenças de cores. O algoritmo de Canny detecta apenas a magnitude das funções das bordas (identificado pela fórmula abaixo), e posteriormente, a direção da magnitude.

$$magn(i,j) = \sqrt{fx^2 + fy^2}$$

Fórmula 3 - Fórmula para cálculo da direção da magnitude

Com os valores de magnitude, o algoritmo de Canny analisa os valores de máximo e mínimo das funções, para identificar as possíveis bordas da imagem.

Uma vez que as bordas foram detectadas, utilizamos o algoritmo de Hough Transform para detectar as linhas retas da imagem, afim de localizarmos um caminho em frente (como um corredor, por exemplo, onde as paredes foram linhas em paralelo). A linha em Hough Transformation é dada pela seguinte equação:

$$p = x\cos\emptyset + y\sin\emptyset$$

Fórmula 4 - Equação para cálculo de linhas em paralelo

Com a detecção das linhas retas da imagem processada, é necessário destacar aquelas que seguem o trajeto necessário. Para isso, é calculado o ponto de fuga da imagem, onde as demais retas convergem.

Com o ponto de fuga estabelecido, basta destacar as retas que estão paralelas e cruzam neste ponto, a fim de estabelecer uma rota para o robô.

O método de Lane Detection é utilizado neste projeto a fim de permitir que o robô identifique possíveis rotas a serem seguidas, melhorando seu desempenho quanto a sua locomoção.

2.6 Smartphone

Um smartphone nada mais é do que um celular com capacidades de processamento superiores, permitindo a execução de aplicativos e programas por intermeio de um sistema operacional.

Os smartphones de hoje em dia possuem características comparáveis a de um notebook, com processadores capazes de executar múltiplas tarefas. O Samsung Galaxy SIII, por exemplo, é o smartphone atual mais poderoso do mercado, possuindo um processador quad-core de 1.4 Ghz, e até 1Gb de memória RAM.

Figura 24 - O Galaxy SIII é o atual smartphone mais potente do mercado (http://www.portaltech.blog.br/wp-content/uploads/2012/07/Galaxy-S-III-1.jpg)

A principal característica dos smartphones é a possibilidade de desenvolvedores criarem seus próprios aplicativos, e poderem distribui-los nas plataformas vigentes de cada sistema operacional. Para isto, cada sistema

possui seu próprio kit de desenvolvimento (SDK), permitindo que haja o desenvolvimento de aplicações para aquela plataforma.

2.6.1 Android

O Android é o sistema operacional da Google, feito para smartphones. Sua principal característica esta no fato de ser um software livre para uso, permitindo que diferentes tipos de smartphones a integrem. Por possuir um kernel baseado no sistema Linux, o Android se tornou um dos sistemas favoritos entre os desenvolvedores, que passaram não só a se forcar no desenvolvimento de aplicações, mas também no desenvolvimento de novos recursos para o sistema.

Desde sua primeira versão, lançada em 2008, o Android se tornou o principal concorrente do iOS, o sistema operacional mobile da Apple. Devido a esta concorrência, o sistema vêm recebido diversas atualizações, trazendo novos recursos, além de melhorias em seu sistema. Sua atual versão, a 4.2 (chamada de Jelly Bean), traz melhorias na performance da CPU, além de redução na latência do toque e a possibilidade de um buffering triplo.

O desenvolvimento de aplicações para o Android é feitas a partir da linguagem Java, utilizando o kit de desenvolvimento para o sistema, integrado com uma IDE de desenvolvimento Java, como o Eclipse. Os desenvolvedores possuem liberdade para testarem seus aplicativos em seus próprios dispositivos, desde que os mesmos utilizem o Android como sistema operacional. Para distribuição, o Android disponibiliza sua loja virtual, chamada de Google Play, onde os desenvolvedores podem distribuir seus aplicativos livremente, podendo ou não atribuir preços de compra para os mesmos (caso isto seja feito, o Google recebe uma porcentagem para cada venda do aplicativo).

O desenvolvimento interno utilizando o SDK para Android utiliza um conceito chamado MVC (model – view – controller), permitindo dividir a codificação em camadas distintas.

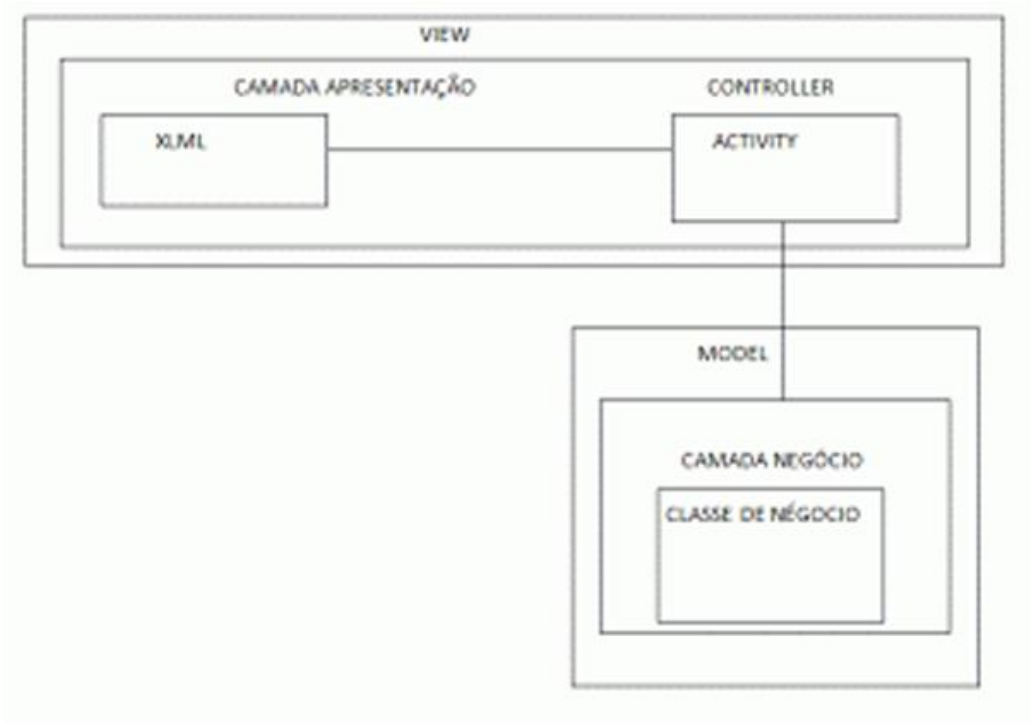


Figura 25 – Arquitetura MVC usada nas aplicações Android (http://www.androidbrasilprojetos.org/wp-content/uploads/2011/06/MVN-ANDROID-300x211.gif)

A execução dos códigos no Android são geralmente atribuídas dentro de Activitys, classes que são executadas junto com a interface. Para cada tela dentro de uma aplicação Android, existe uma Activity que garante sua execução.

Este projeto utiliza o sistema Android como base para o desenvolvimento de uma aplicação de controle do robô. A escolha do sistema se deu por alguns motivos. Entre eles, pelo uso de Java como principal linguagem para desenvolvimento de aplicações, além da facilidade da execução da aplicação diretamente no dispositivo.

3. Sobre o Projeto

Este capítulo aborda questões sobre o desenvolvimento do projeto, sua concepção, arquitetura, dificuldades e soluções encontradas.

3.1 Concepção

O projeto deu início com a ideia 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.

Figura 26 - O projeto pode ser utilizado para orientação de pessoas com deficiência visual a se locomoverem em ambientes fechados, como universades e outros espaços.

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.

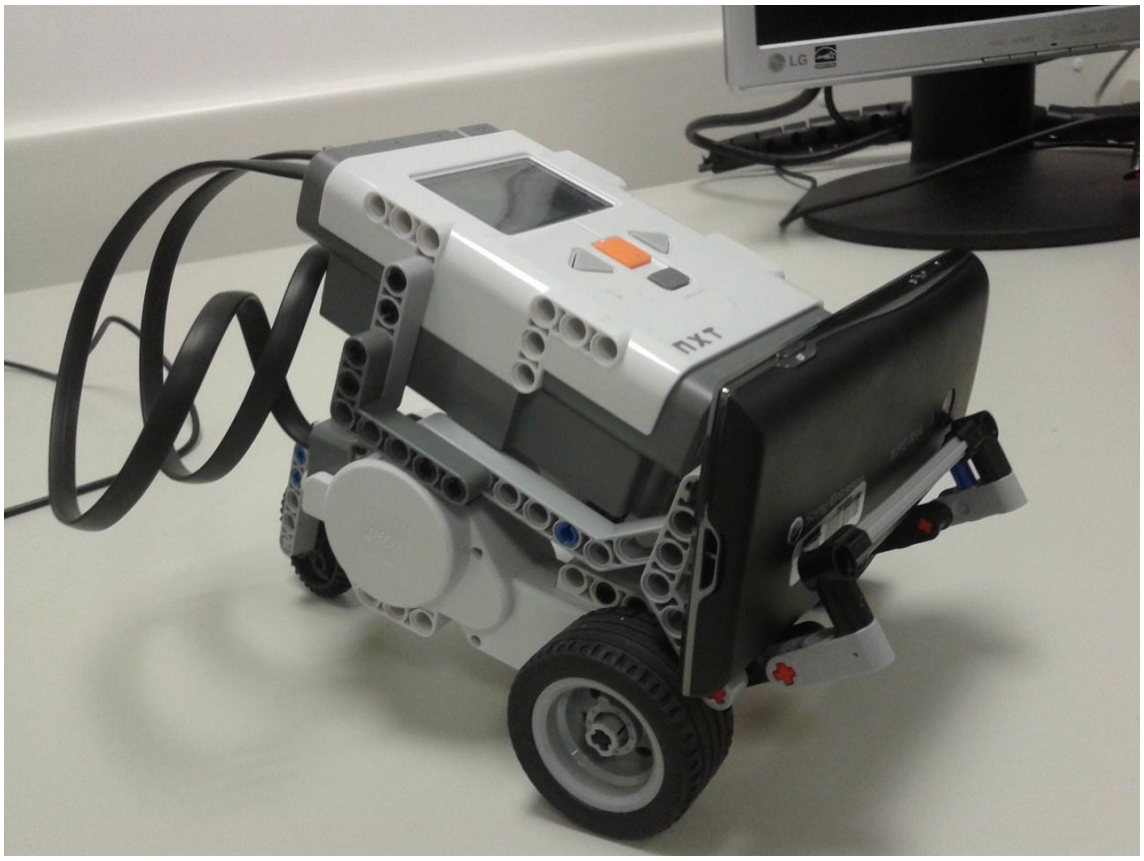


Figura 27 - O celular ficará a frente do robô, captando suas imagens e as processando para determinar suas ações.

O desenvolvimento do projeto foi dividido em três partes: desenvolvimento do protótipo do robô, desenvolvimento de uma aplicação de controle e desenvolvimento da autonomia do robô.

3.2 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 28 - Protótipo final do robô utilizado neste projeto

Alguns eixos foram adicionados para suporte da peça, além de uma roda traseira, conectada apenas por uma aste, 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.

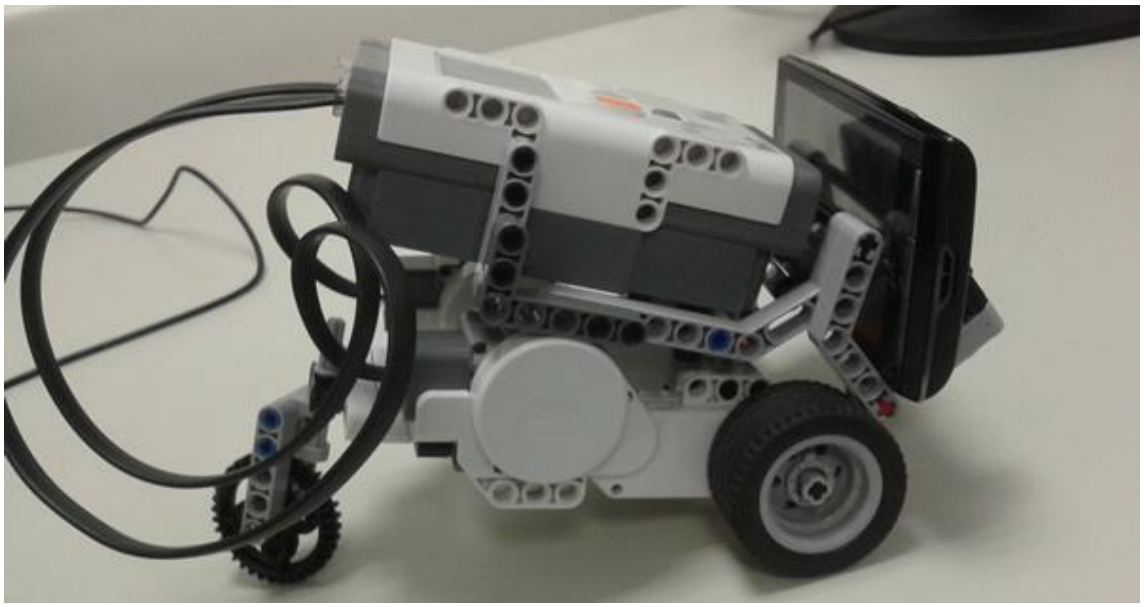


Figura 29 - O robô é suportado por uma aste traseira, e possui um suporte a sua frente, a fim de segurar o smartphone.

3.3 Desenvolvimento da 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 intruções 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 Hardware Developer Kit (LEGO, 2006) 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 possuí dois tipos de endereços:

1. Um endereço I²C único para cada sensor digital.

2. Um endereço interno para leitura ou escrita.

A documentação "Appendix 7-LEGO MINDSTORMS NXT Ultrasonic Sensor I2C communication protocol" do sensor ultrassônico I²C protocolo de comunicação, diz que seu endereço I²C é definido como padrão 0x02 e o pacote deste protocolo é composto por 3 bytes sendo:

Byte 0	Byte 1	Byte 2
Endereço I²C deve ser enviado 0x02 (00000010)	Endereço de memória interno (descrito na documentação do sensor)	0x02 para escrita, 0x03 para leitura, na posição de memória interna

Figura 30 - Pacote de informações de envio do protocolo I2C

Com este protocolo é possível estabelecer um meio de comunicação entre a Brick e o sensor Ultrassônico.

Para a comunicação entre o celular e a Brick, utilizamos o protocolo LCP (Lego Communication Protocol). O LCP é um protocolo de comunicação, utilizado estabelecer a comunicação entre a Brick e dispositivos terceiros como Celulares, Computadores (PC's), entre outros, e sua documentação esta descrita em "LEGO MINDSTORMS NXT Bluetooth Developer Kit".

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 31 - 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, leia a documentação em "Appendix 2-LEGO MINDSTORMS NXT Directcommands" e "Appendix 1-LEGO MINDSTORMS NXT Communication protocol", para aprender sobre as possibilidades de entrada em cada byte no pacote.

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 próximos.

Figura 32 - 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. Ao fazer isto, o celular passa a se conectar com o robô, e exibe a lista de seus possiveis módulos de controle do robô.

Figura 33 - Tela de escolha do módulo de controle.

O desenvolvimento desta aplicação utilizou conceitos MVC (model – view – controller) para arquitetura do software, conforme pode ser visto na imagem abaixo.

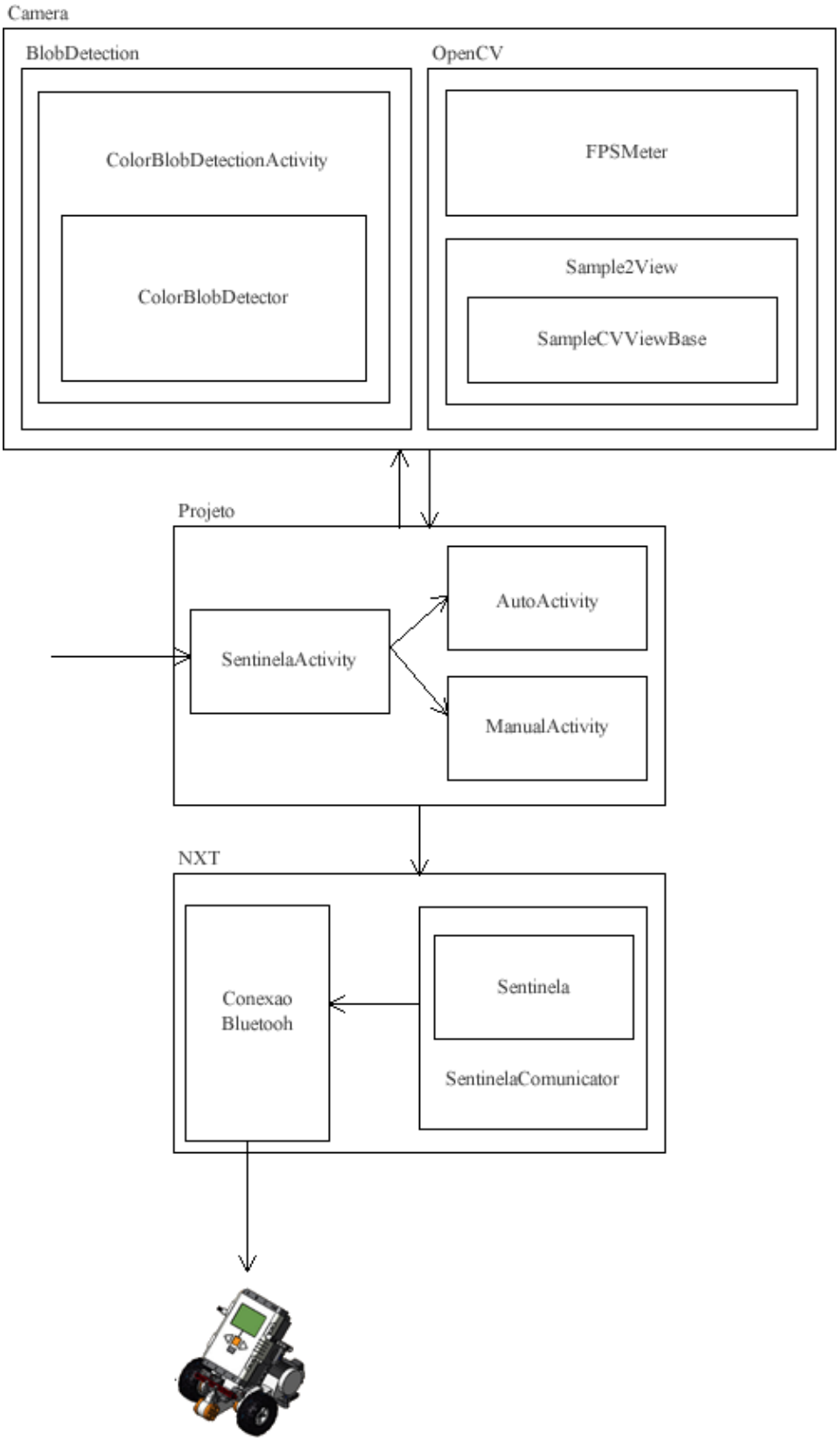


Figura 34 – Arquitetura de classes do projeto.

3.3.1 Módulo de controle manual

Para fins de controlar manualmente o robô, foi desenvolvido na aplicação de controle um pequeno módulo que permite que o usuário controle a movimentação do robô através da interação com smartphone. Ao selecionar a opção de controle manual, uma tela é exibida com 4 botões dispostos, cada qual indicando uma possível ação do robô, onde o usuário pode movimenta-lo para frente ou para trás, além de poder gira-lo para a esquerda ou para a direita.

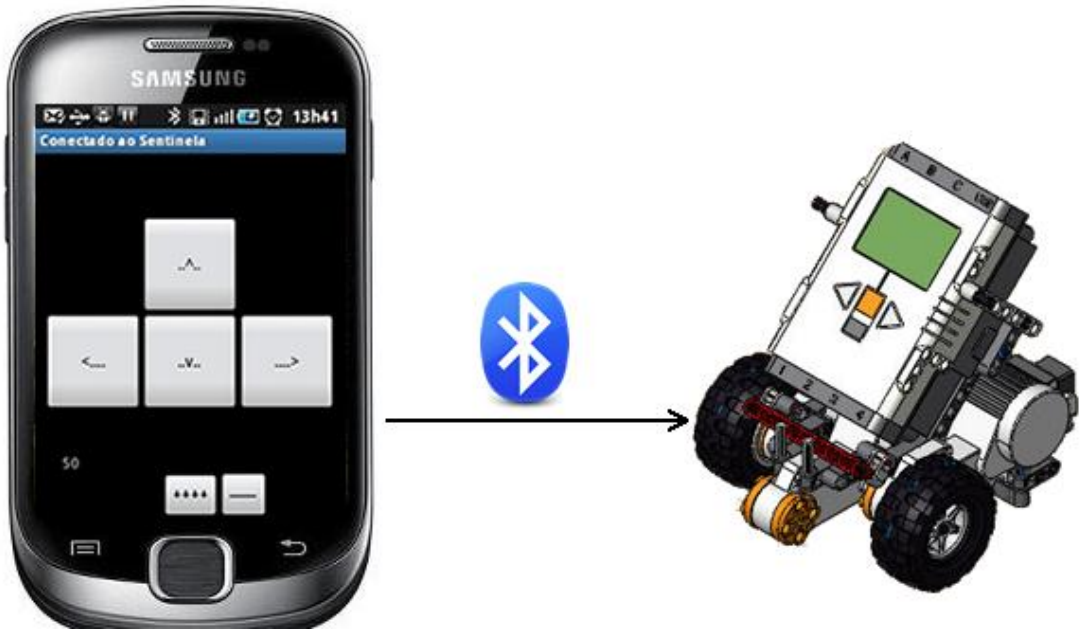


Figura 215 - Através do controle manual, o usuário tem a opção de controlar o robô através da interação com os botões do aplicativo.

Para cada botão pressionado, uma instrução é enviada para o robô via o canal Bluetooth, indicando a ação desejada. Este processo permitiu o mapeamento das instruções a serem passadas para o robô, além de permitir testar suas funções básicas de locomoção, arquitetando métodos padrões para estas ações.

3.3.2 Módulo de 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 atuasse não sofresse muitas adaptações para seu funcionamento. Devido a isto, o uso de técnicas de visão computacional foram 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.

Ao selecionar a opção de controle autônomo a partir da aplicação de controle, a mesma passa a exibir uma nova tela com as imagens capturadas diretamente da câmera do smartphone.

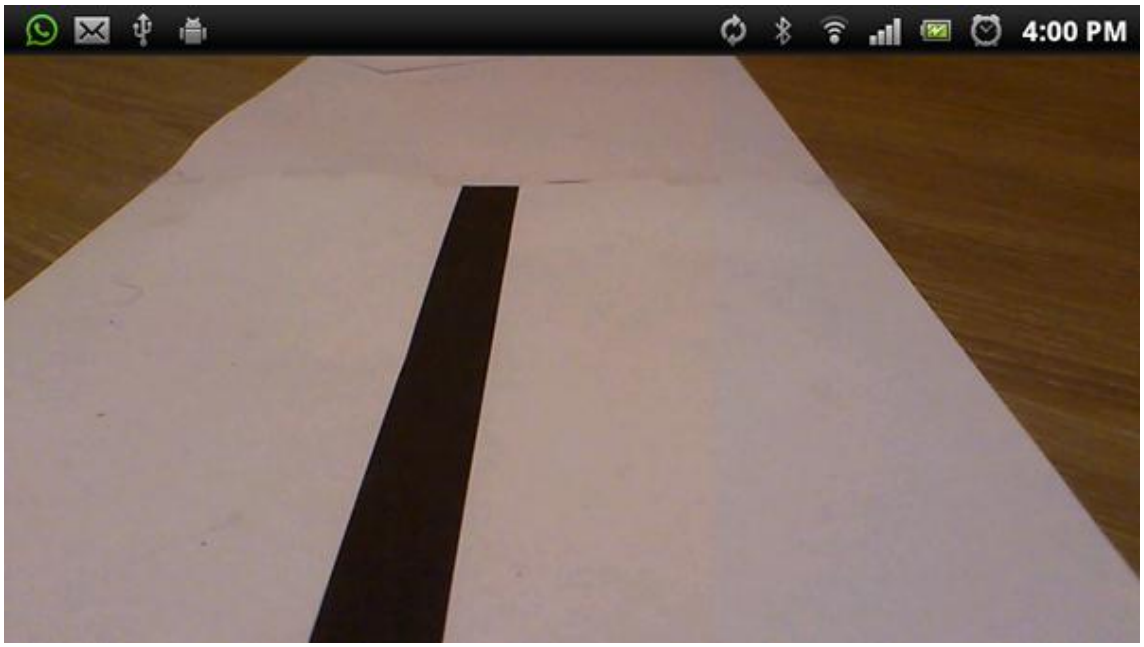


Figura 36 - Após selecionar o uso do módulo de controle autônomo, o aplicativo passa a exibir as imagens captadas pela câmera do smartphone.

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 de

processamento. Os métodos do OpenCV permitiram processar a imagem de cada quadro capturado, aplicando filtros ou detectando padrões.

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

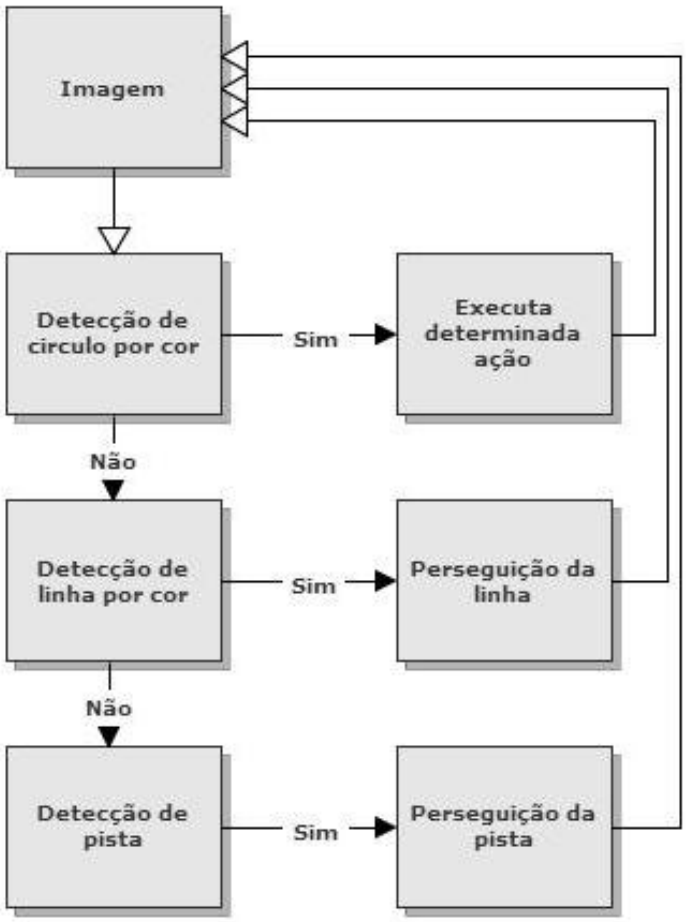


Figura 37 – Arquitetura do funcionamento dos submódulos constituintes do controle autônomo.

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

3.3.2.1 Módulo de detecção de linhas por cores

O módulo de detecção de cores e linhas foi desenvolvido com o propósito de permitir que o robô detecte um trajeto ou caminho específico a ser seguido, como por exemplo, uma guia de piso táctil para portadores de deficiência visual. Estes caminhos devem sempre ter uma cor específica, permitindo que o robô consiga rastrea-la.

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, utilizamos o método inRange do OpenCV para extrair um range de cores específicos, escolhida pelo usuário, passando uma imagem nos padrões HSV para o método. Este método retorna uma imagem binarizada, onde os pontos na cor branca estavam na faixa de cores especificada.

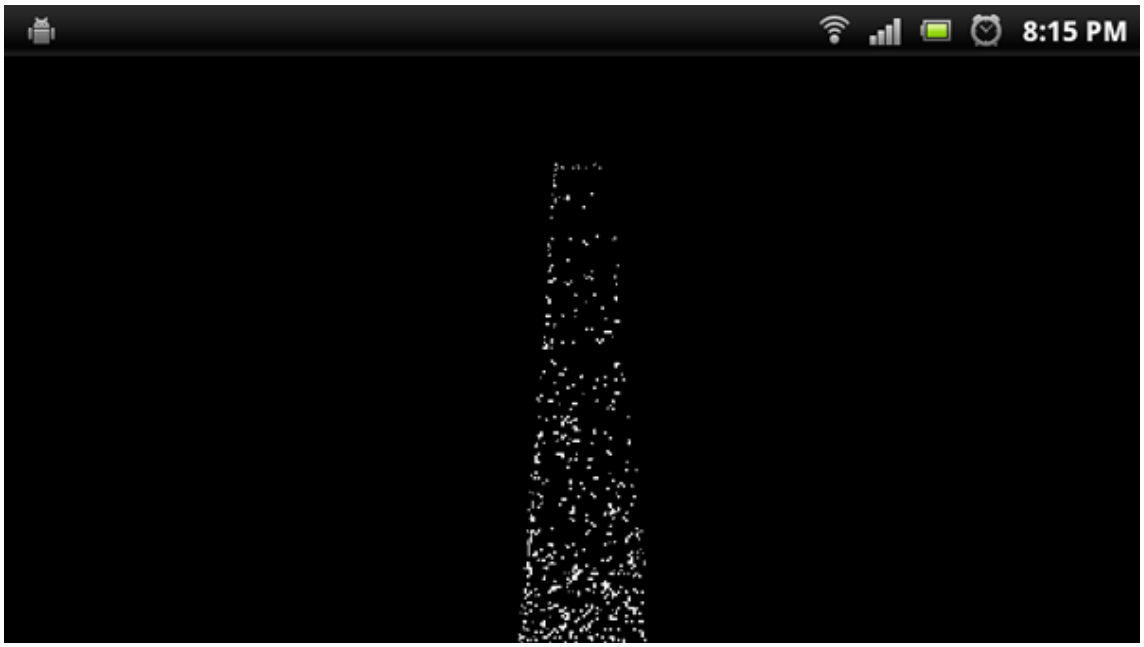


Figura 38 - O retorno do método inRange é uma imagem binarizada, com as cores no padrão HSV.

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 (Suzuki, Z. 1985). 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. Sendo assim, a partir do momento em que o usuário pressiona o dedo em cima da linha na tela do celular, o algoritmo passa a exibir os contornos ao redor do caminho desenhado.

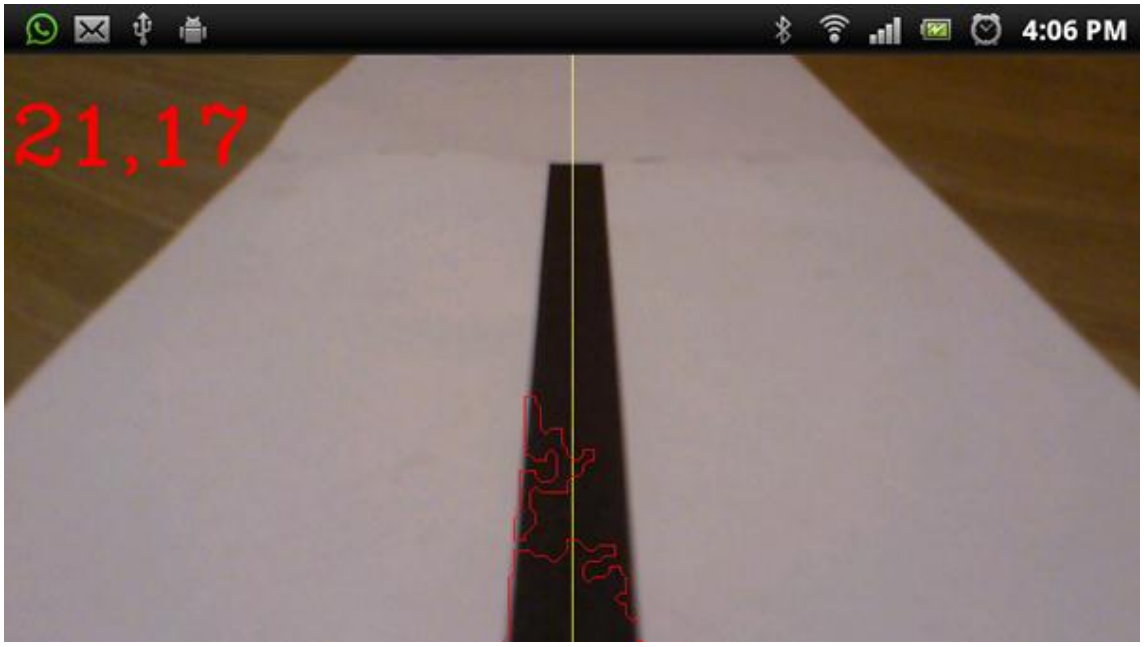


Figura 39 - Com o uso do método findCountours permite encontrar os contornos ao redor das áreas da cor escolhida.

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:
$u2 = u2 * 0{,}5$

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

3.3.2.2 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 detecta locais que tenham características de uma pista, e passa a locomover o robô na direção da convergência da mesma.

Este módulo de detecção de pistas utiliza métodos de Lane Detection. 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.

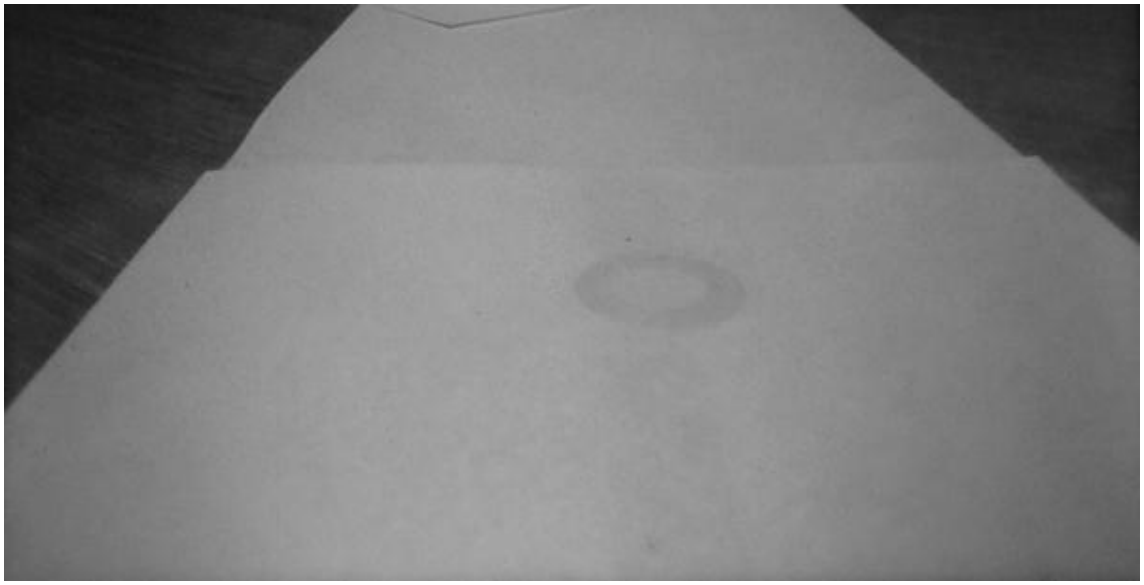


Figura 40 – Aplicação do filtro de escala de cinza utilizando o método cvtColor.

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 41 – 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).

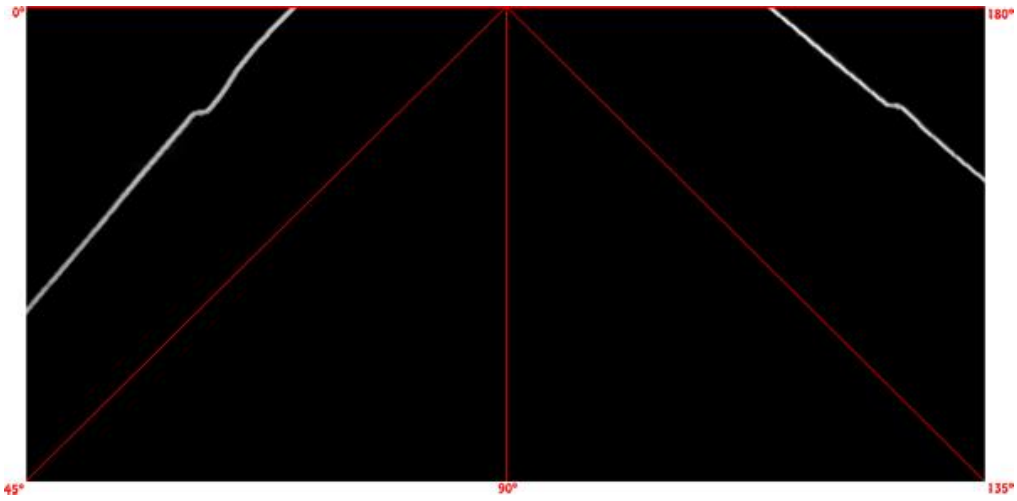
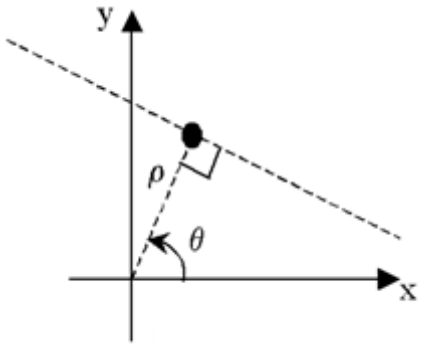


Figura 41 – Possíveis ângulos das retas detectadas pelo método de Hough.

Com o retorno do método HoughLines, são obtidos os ângulos em relação ao centro da imagem, permitindo desenha-las na tela.

Para cada uma das retas:

Ângulo Θ e Distância ρ



Onde a equação da reta é dada por:

$$r = x * \Theta * \cos(\rho) + \mathrm{y} * \Theta * \sin(\rho)$$

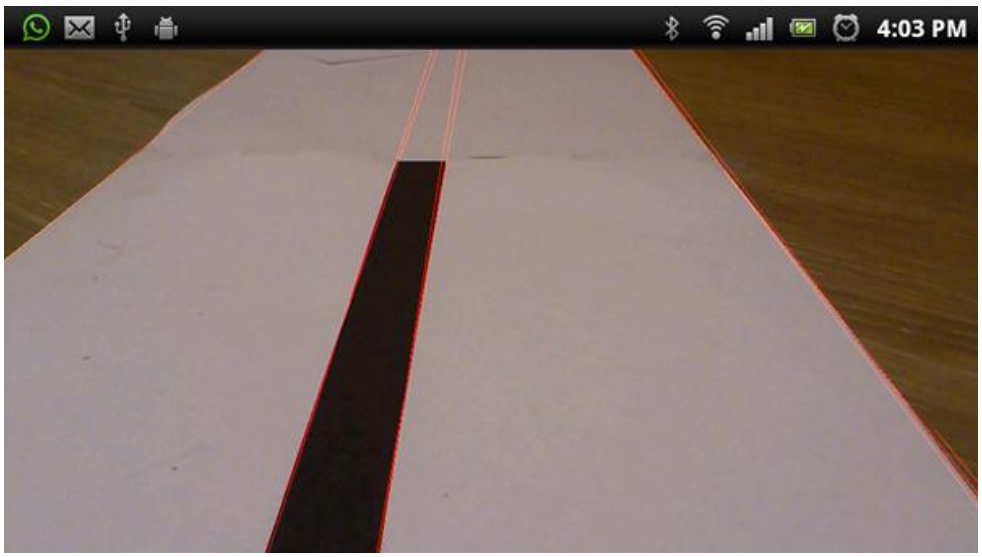


Figura 42 – O retorno do método HoughLines permite exibir as linhas das possíveis rotas.

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.

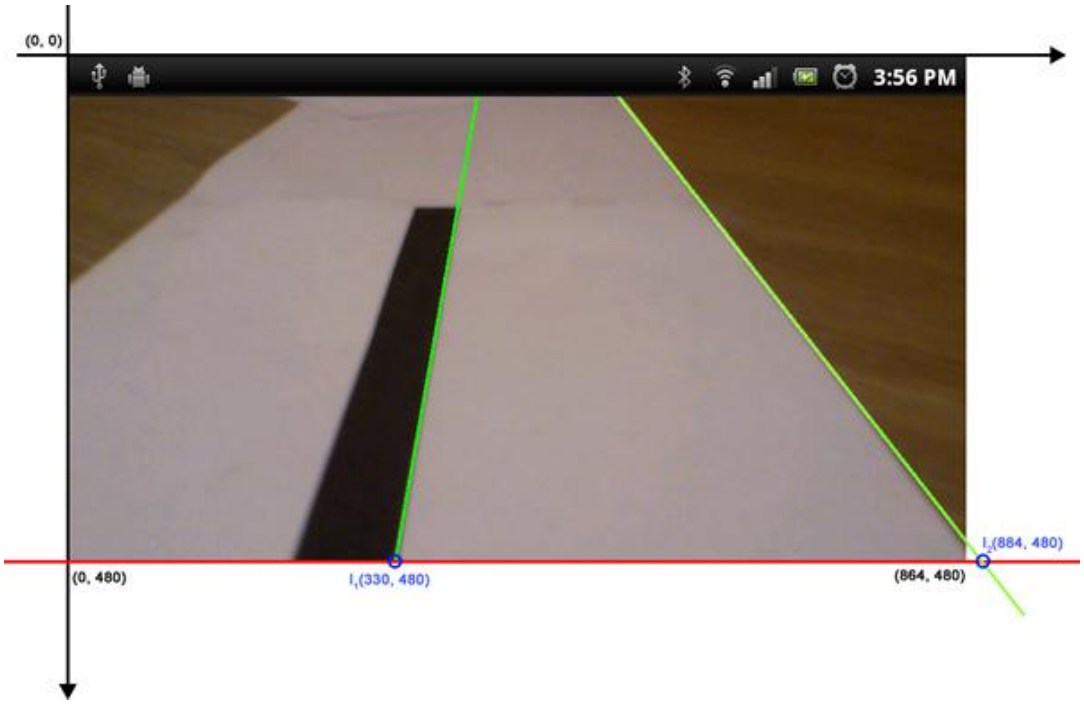


Figura 43 – Detecção de uma possível rota.

Quando pelo menos uma das retas que formam a pista é encontrada, é calculada a velocidade de cada motor do robô, possibilitando sua locomoção. Esta velocidade é dada pela seguinte fórmula:

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

$$Caso\ o\ ângulo\ da\ reta\ encontrada\ seja\ \ maior\ que\ \frac{\pi}{2}$$

$$velocidade_{esquerda} = velocidade_{minima} + velocidade_{variada} * \left(\frac{\pi}{2} - \emptyset_{direita}\right)$$

$$velocidade_{direita} = velocidade_{minima} + velocidade_{variada} * \left(\frac{\pi}{2} - \emptyset_{direita}\right)$$

$$Caso\ o\ ângulo\ da\ reta\ encontrada\ seja\ \ menor\ que\ \frac{\pi}{2}$$

$$velocidade_{esquerda} = velocidade_{minima} + velocidade_{variada} * \left(\emptyset_{direita} - \frac{\pi}{2}\right)$$

$$velocidade_{direita} = velocidade_{minima} + velocidade_{variada} * (\emptyset_{direita} - \frac{\pi}{2})$$

Também foi implementado uma previsão restritiva quanto à obtenção de uma pista viável.

Com base na distância e o ângulo para cada uma das retas filtradas (viáveis) é feita a média para cada um destes valores.

O calculo da restrição do ângulo é dada pela seguinte formula:

$$P_{\boldsymbol{\theta}|n+1} = \frac{\sum_{i=0}^{i=n} \boldsymbol{\theta}_i}{n}$$

O calculo da restrição da distância é dada pela seguinte formula:

$$P_{\boldsymbol{\rho}|n+1} = \frac{\sum_{i=0}^{i=n} \boldsymbol{\rho}_i}{n}$$

Onde $P_{\boldsymbol{\theta}}$ representa a previsão restritiva do ângulo, $P_{\boldsymbol{\rho}}$ representa a previsão restritiva da distância e n a quantidade de frames (imagens obtidas pela câmera).

Também deve ser definida duas constantes $C_{\boldsymbol{\theta}}$ e $C_{\boldsymbol{\rho}}$ para determinar uma faixa de valores em que a próxima reta viável deve estar presente.

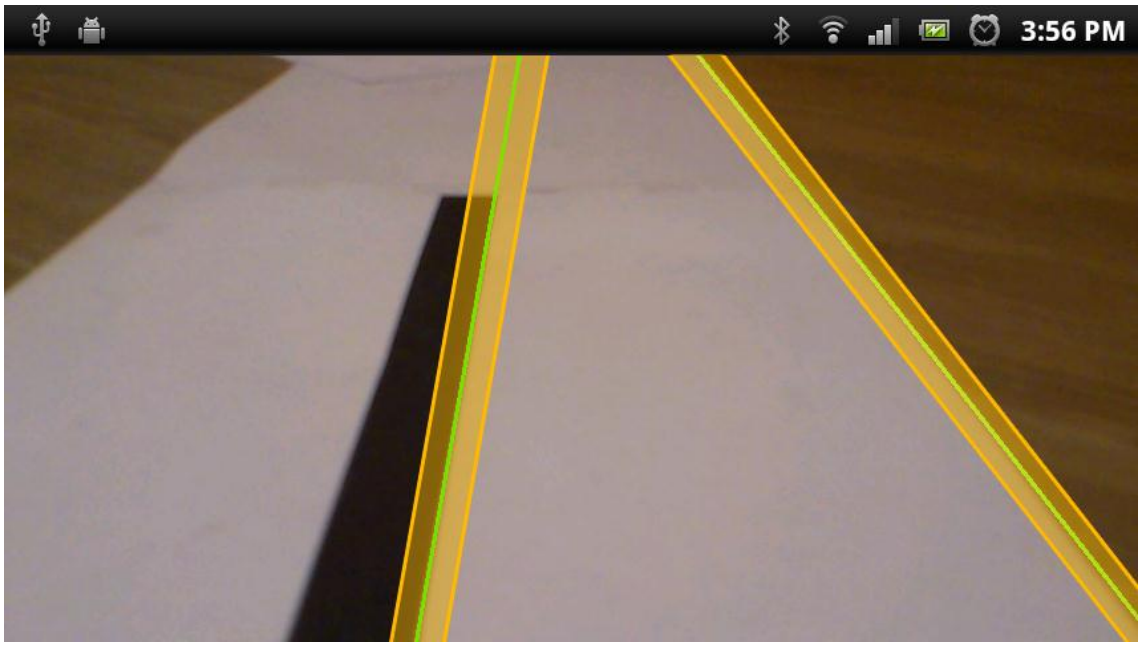


Figura 44 – Ilustração da previsão restritira demonstrada pelas faixas em amarelo. As retas em verde foram determinadas como viáveis por estarem dentro da restrição.

Esta previsão restritiva deve ser calculada de forma independente para cada um dos lados da tela (esquerda e direita) e suas retas filtradas.

3.3.2.3 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 lomoção 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 aplicado um e 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.

Figura 45 – Detecção dos contornos de uma circuferência.

Com os contornos encontrados, é realizado 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.

A filtragem dos contornos que são circulares é determinada por duas métricas M_1 e M_2,são dadas pela seguinte formula:

$$M_1 = \frac{A * 4 * \pi}{p^2}$$

$$M_2 = \frac{A * 16}{p^2}$$

Onde A é a área total do contorno, p é dado pelo tamanho total do curva do arco formado pelo contorno (método arcLenght). Para os valores onde a métrica M_1 é maior que $0{,}75$ e os valores da métrica M_2 é maior que 1, o contorno é circular.

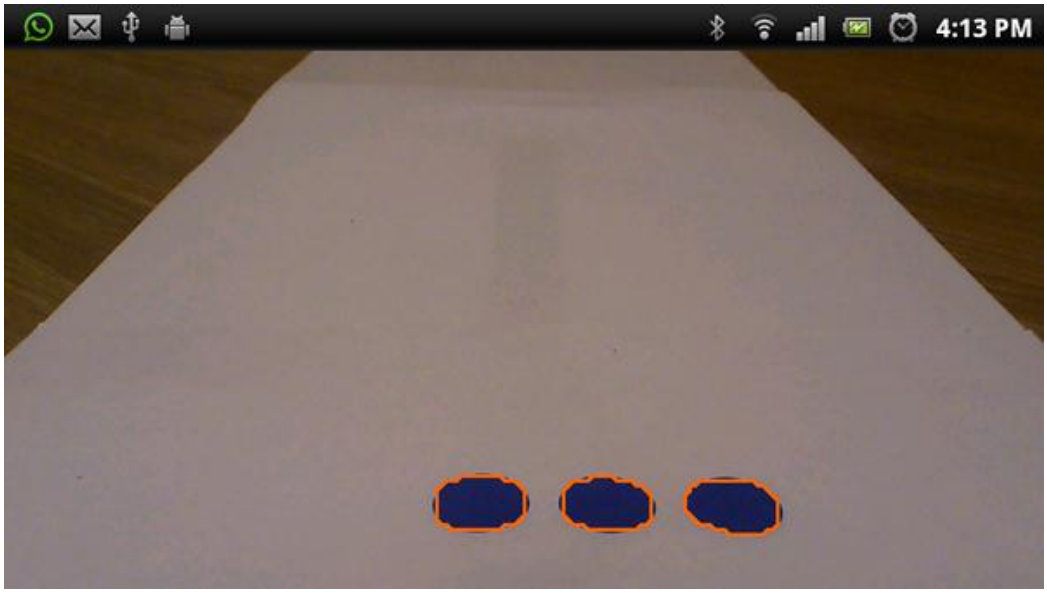


Figura 45 – Aplicação do filtro para detecção de símbolos circulares.

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.

4. Resultados

Neste projeto, foram construídos diversos tipos de protótipos para a validação da tecnologia implementada. O protótipo inicial se baseou na utilização do módulo de controle manual, onde a partir da interação do smartphone, pudemos controlar o robô na direção desejada. Um dos problemas encontrados neste módulo foi o tratamento da conexão bluetooth. Caso a mesma não fosse fechada corretamente após seu uso, a aplicação passava a exibir mensagens de erro, impossibilitando seu uso em diante.

Com o protótipo inicial fechado, foi possível trabalhar no desenvolvimento da autonomia do robô. 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 melhor 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 atuam 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, e 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 pára ao identificar um número de círculos pré-determinados.

5. 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. A primeira etapa consistia no desenvolvimento de uma aplicação para o sistema Android capaz de mandar instruções para locomoção do robô. Nesta etapa, foi identificado a necessidade da construção de um protocolo de comunicação com o robô, a fim de que as instruções enviadas via o canal Bluetooth possibilitasse acionar os motores do robô.

A segunda etapa consistiu no desenvolvimento da locomoção autônoma do robô. Para tal, foram pesquisados métodos para seguimento autônomo, e identificado a 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.

Este projeto abre novas possibilidades para caminhos da robótica, integrando o celular Android como uma fonte principal de processamento, possibilitando utilizar seus recursos para abrir novas possibilidades a robôs.

Além disto, também impõe uma solução de baixo custo para sistemas de automação robótica, permitindo o uso de linguagens de baixo nível para seu desenvolvimento, como a linguagem Java.

5.1 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.

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.

6. Bibliografia

[1] 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>.

[2] DE SOUZA, FELIPPE. Automação Industrial e Robótica. Disponível em: <http://webx.ubi.pt/~felippe/texts3/autom_ind_cap1.pdf>.

[3] BROOKS, R. A., Achieving Artificial Intelligence Through Building Robots, MIT AI Lab Memo No. 899, May 1986. Disponível em: <http://people.csail.mit.edu/brooks/papers/aim-899.pdf>

[4] http://www.newtoncbraga.com.br/index.php/robotica/4018-rp0001.html

[5] http://asimo.honda.com/

[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] http://thenxtstep.blogspot.com.br/

[8] http://mindstorms.lego.com

[9] SHIH, Bih-Yaw. CHEN, Chen-Yuan. CHOU Wei-Chung. Obstacle avoidance using a path correction method for autonomous control of a biped intelligent robot. 2010.

[10] 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>.

[11] 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.

[12] 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.

[13] REN, F. HUANG, J. TERAUCHI, M. JIANG, R. KETTLE, R. Lane Detection on the iPhone

[14] Suzuki, S. and Abe, K., *Topological Structural Analysis of Digitized Binary Images by Border Following*. CVGIP 30 1, pp 32-46 (1985)