

UNIVERSIDADE FEDERAL DO PIAUÍ
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

ERMENSON MEZZOMO

**DESBRAVAMENTO E MAPEAMENTO DE ROTA
UTILIZANDO ROBÔ AUTÔNOMO**

PICOS-PI
2013

ERMENSON MEZZOMO

DESBRAVAMENTO E MAPEAMENTO DE ROTA UTILIZANDO ROBÔ AUTÔNOMO

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como parte dos requisitos para obtenção do Grau de Bacharel sob orientação do Prof. Esp. Ivenilton Alexandre de Souza Moura.

PICOS-PI

2013

Eu, **Ermenson Mezzomo**, abaixo identificado(a) como autor(a), autorizo a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação abaixo discriminada, de minha autoria, em seu site, em formato PDF, para fins de leitura e/ou impressão, a partir da data de hoje.

Picos-PI 20 de setembro de 2013.

Ermenson Mezzomo

Assinatura

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

M617d Mezzomo, Ermenson.
Desbravamento e mapeamento de rota utilizando robô autônomo / Ermenson Mezzomo. – 2013.
CD-ROM : il. ; 4 ¾ pol. (81 p.)

Monografia(Bacharelado em Sistemas de Informação) – Universidade Federal do Piauí. Picos-PI, 2013.
Orientador(A): Prof. Esp. Ivenilton Alexandre de Souza Moura

1. Robótica. 2. Arduino. 3. Mapeamento. I. Título.

CDD 005.1

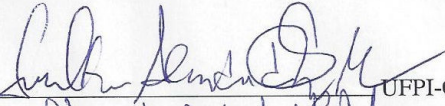
ERMENSON MEZZOMO

DESBRAVAMENTO E MAPEAMENTO DE ROTA UTILIZANDO ROBÔ AUTÔNOMO

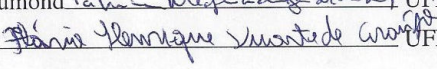
Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como parte dos requisitos para obtenção do Grau de Bacharel sob orientação do Prof. Esp. Ivenilton Alexandre de Souza Moura.

Data de Aprovação:

13 / 09 / 2013

Ivenilton A. de S. Moura  UFPI-CSHNB

Patricia M. L. de L. Drumond  UFPI-CSHNB

Flávio H. D. de Araújo  UFPI-CSHNB

PICOS-PI

2013

Dedico este trabalho à minha família em especial aos meus pais, Edgar Antonio Mezzomo e Loeci Malaquias da Rosa Mezzomo que me deram muito apoio nos momentos mais difíceis da minha vida, ao meu irmão Edson Luiz Mezzomo e minha cunhada Marcia Cristina Scheifer e meu sobrinho Daniel Augusto, que sempre estão ao meu lado, me ajudando, pois eles nunca mediram esforços para me ajudar e sempre acreditaram no meu potencial me dando total apoio ao longo desta caminhada, sendo de total importância tanto para eles quanto é para mim.

Quero agradecer, em primeiro lugar, a Deus, pela força e coragem durante toda esta longa caminhada. Agradeço também a todos os professores que me acompanharam durante a graduação, em especial ao Prof. Algeir Prazeres Sampaio e ao Prof. Ivenilton Alexandre de Souza Moura orientador deste trabalho, responsáveis pela realização deste trabalho. Aos meus amigos, por todo o companheirismo e pelos bons momentos que passamos ao longo do curso. A minha mãe, Loeci Malaquias da Rosa Mezzomo e ao meu pai, Edgar Antonio Mezzomo, pelas orientações e pelas palavras de apoio a cada momento difícil que passei durante o curso, sendo essa mais uma conquista vencida na minha vida.

"Tente uma, duas, três vezes e se possível tente a quarta, a quinta e quantas vezes for necessário. Só não desista nas primeiras tentativas, a persistência é amiga da conquista. Se você quer chegar a onde a maioria não chega, faça o que a maioria não faz".

(Bill Gates)

"A experiência que não é convertida em conhecimento é apenas informação".

(Denilson Alves)

"O covarde nunca tenta, o fracassado nunca termina e o vencedor nunca desiste".

(Norman Vincent Peale)

Resumo

A robótica tem apresentado importância crescente na sociedade, inicialmente colaborando para o aumento da produtividade nas fábricas e recentemente, na realização de tarefas domésticas e explorações de locais desconhecidos e/ou inacessíveis ao ser humano. Nesse contexto, iniciam-se a construção de dispositivos autônomos capazes de executar tarefas sem a necessidade de um controlador humano. Com isso, propõe-se utilizar um robô autônomo que seja capaz de se locomover em ambientes desconhecidos de forma autônoma, mapeando a sua trajetória feita naquele ambiente. Para realizar esse tipo de tarefa em um determinado ambiente é necessário à utilização de sensores que fazem as leituras das circunstâncias do ambiente mandando essas informações para os microcontroladores responsáveis pelo processamento e gerenciamento das informações onde são enviadas e transformadas em um mapa do percurso feito dentro do ambiente inserido. Ao mesmo tempo apresentar uma breve descrição sobre a robótica, mapeamento e deslocamento além de conceitos de eletrônica e da plataforma Arduino, utilizada para processar as informações e controlar os demais componentes.

Palavras-chave: Robótica, Arduino, Mapeamento, Autônomo, Desbravamento.

Abstract

Robotics has shown increasing importance in society, initially collaborating to increase productivity in factories and recently in domestic tasks and explorations of places unknown and / or inaccessible to humans. In this context, we begin the construction of autonomous devices capable of performing tasks without the need for a human controller. With this, it is proposed to use an autonomous robot that is able to get around in unfamiliar surroundings autonomously, mapping their trajectory of that environment. To perform this kind of task in a given environment is necessary the use of sensors that the readings of the circumstances of the environment by sending this information to the microcontrollers responsible for management and processing of information which are sent and transformed into a map of the route taken within the environment inserted. At the same time present a brief description about robotics, mapping and displacement beyond concepts of electronics and the Arduino platform, used to process information and control other components.

Keywords: Robotics, Arduino, Mapping, Autonomous, Exploring.

Lista de Figuras

Figura 1 - Robôs de Hoje em uma Linha de Produção	20
Figura 2 - Robô Móvel Inteligente	21
Figura 3 - Veículo Aéreo Não-Tripulado em Operação Militar	21
Figura 4 - Veículo Autônomo Subaquático	22
Figura 5 - Robô Sojourner	22
Figura 6 - Os Robôs Humanoides P3 e Asimo da Honda	22
Figura 7 - Robô Elektro e Sparko o Cão Robô	24
Figura 8 - Primeiro Tear Mecânico	24
Figura 9 - Unimate, Primeiro Robô Industrial	25
Figura 10 - Placa Arduino UNO	31
Figura 11 - Placa Arduino Mega 1280	32
Figura 12 - Placa Arduino Mega 2560	33
Figura 13 - IDE 1.0.1 do Arduino	34
Figura 14 - Atuação do Sensor Ultrassônico	35
Figura 15 - Sensores Ultrassônicos	36
Figura 16 - Motor de Passo Conectado ao MotorShield	37
Figura 17 - Módulos de RF	38
Figura 18 - Protoboard Externamente	39
Figura 19 - O Diagrama Interno da Ligações de uma Protoboard	39
Figura 20 - Circuito Simples para Acender um LED	40
Figura 21 - Protoshield	41
Figura 22 - PDE do Processing	42

Figura 23 - Ligação Entre os Componentes	45
Figura 24 - Ligação Entre Todos os Componentes	48
Figura 25 - Sistema de Tração do Protótipo	48
Figura 26 - Montagem do Protótipo	49
Figura 27 - Montagem do Protótipo Visão Lateral	49
Figura 28 - Visão Lateral do Protótipo	49
Figura 29 - Veículo Móvel Montado Visto de Frente	50
Figura 30 - Ligação da Antena RF com o Arduino	50
Figura 31 - Tela Inicial	52
Figura 32 - Tela Referente ao Andar para Frente	52
Figura 33 - Tela Referente ao Girar para Esquerda	53
Figura 34 - Tela Referente ao Girar para Direita	53
Figura 35 - Tela Referente ao Girar 180°	54
Figura 36 - Tela Referente a um Mapa Percorrido	54
Figura 37 - Quando Não Possui Obstáculo à Frente do Robô	55
Figura 38 - Obstáculo à Frente do Robô	55
Figura 39 - Obstáculo à Frente e na Direita do Robô	56
Figura 40 - Obstáculo à Frente e na Esquerda do Robô	56
Figura 41 - Obstáculo à Frente e na Direita e Esquerda do Robô	56
Figura 42 - Labirinto 01	57
Figura 43 - Resultado do Percurso do Labirinto 01	57
Figura 44 - Labirinto 02	58
Figura 45 - Resultado do Percurso do Labirinto 02	58
Figura 46 - Labirinto 03	58
Figura 47 - Resultado do Percurso do Labirinto 03	58
Figura 48 - Labirinto 04	58
Figura 49 - Resultado do Percurso do Labirinto 04	58

Figura 50 - Labirinto 05	59
Figura 51 - Resultado do Percorso do Labirinto 05	59
Figura 52 - Labirinto 06	59
Figura 53 - Resultado do Percorso do Labirinto 06	59
Figura 54 - Labirinto 07	59
Figura 55 - Resultado do Percorso do Labirinto 07	59
Figura 56 - Labirinto 08	60
Figura 57 - Resultado do Percorso do Labirinto 08	60
Figura 58 - Labirinto 09	60
Figura 59 - Resultado do Percorso do Labirinto 09	60
Figura 60 - Segundo Resultado do Percorso do Labirinto 09	60
Figura 61 - Labirinto 10	61
Figura 62 - Resultado do Percorso do Labirinto 10	61

Lista de Tabelas

Tabela 1 -	Especificações da Placa de Arduino Uno	31
Tabela 2 -	Especificações da Placa de Arduino Mega 1280	32
Tabela 3 -	Especificações da Placa de Arduino Mega 2560	33
Tabela 4 -	Especificação dos Sensores	36
Tabela 5 -	Especificação dos Motores de Passo	37
Tabela 6 -	Especificação dos Modulos RF	38
Tabela 7 -	Especificações Protoshield	41
Tabela 8 -	Utilização da Biblioteca Stepper.h	43
Tabela 9 -	Controle dos Motores de Passo	44
Tabela 10 -	Ativamento e Desativamento do Pino 8 Referente ao Sensor da Frente .	45
Tabela 11 -	Controle dos Motores de Passo	46
Tabela 12 -	Comunicação Entre os Arduinos	47
Tabela 13 -	Transmissão do sinal Correspondente	47
Tabela 14 -	Movimento e Sinal Correspondente	48
Tabela 15 -	Sketch que Recebe o Sinal e Envia Pela Porta Serial	51
Tabela 16 -	Tabela de Resultados dos Testes	61

Lista de abreviaturas e siglas

A	Amperagem
CI	Circuito Integrado
DC	Corrente Contínua
IA	Inteligência Artificial
IDE	Integrated Development Environment
LED	Light Emitting Diode
PDE	Processing Development Environment
PWM	Pulse Width Modulation
RF	Radio Frequência
USB	Universal Serial Bus

Sumário

1	Introdução	16
1.1	Motivação	16
1.2	Problema	16
1.3	Abordagem	17
1.4	Objetivos	17
1.4.1	Geral	17
1.4.2	Específico	18
1.5	Organização do Documento	18
2	Referencial Técnico	19
2.1	Robótica	19
2.1.1	Robôs	20
2.1.2	História da Robótica	23
2.2	Navegação Autônoma	26
2.3	Localização	26
2.4	Mapeamento	27
3	Concepção do Projeto	29
3.1	Veículo Móvel	29
3.1.1	Arduino	30
3.1.2	Sensores	35
3.1.3	Motor de Passo	36
3.1.4	Comunicação de Radio Frequência (RF)	38

3.1.5	Protoboard	38
3.2	Projeto da Base	40
3.2.1	Protoshield	40
3.2.2	Notebook	41
3.2.3	Processing	41
4	Implementação	43
4.1	Implementação do Protótipo Robótico	43
4.2	Implementação da Base	50
4.2.1	Recebe Sinal e Envia Pela Porta Serial	50
4.2.2	Recebe Pela Porta Serial e Transforma em Gráfico	52
5	Resultados	55
5.1	As Possíveis Ações Realizadas Pelo Robô	55
5.2	Testes Feitos	57
6	Considerações Finais	62
	Referências	63
	Anexo A – Sketch do Arduino Responsável por Controlar os Motores de Passo	65
	Anexo B – Sketch do Arduino Responsável pelo Controle dos Sensores	71
	Anexo C – Sketch do Arduino Responsável por Enviar os Dados	74
	Anexo D – Sketch do Arduino Responsável por Receber os Dados	76
	Anexo E – Sketch do Processing Responsável por Transformar os Dados Recebidos em Gráficos	77

1 Introdução

1.1 Motivação

Ao imaginarmos o futuro, percebemos que as máquinas ultramodernas estão surgindo com a finalidade de substituir o homem em algumas de suas tarefas, pois terão a capacidade de perceber, reagir e entender as situações que podem vir a surgir no decorrer de seu funcionamento, sem serem controladas por terceiros. Hoje em dia as máquinas já conseguem ter um alto grau de inteligência e com isso já realizam diversas tarefas humanas. Tais tarefas podem ser executadas facilmente e em muitos casos de maneira mais eficiente que os próprios seres humanos.

Nos últimos anos, pôde-se observar um grande crescimento do potencial de sistemas robóticos, isso vem acontecendo, graças ao grau de inteligência que os robôs vêm adquirindo com o passar do tempo. A robótica móvel terrestre é uma das áreas que acompanha tal potencial e envolvendo várias outras como: mapeamento de ambientes, desbravamento de lugares desconhecidos e em conjunto vem se destacando. Mapeamento de rotas pode estar ligado diretamente com diversas tarefas, pois é possível saber o deslocamento do protótipo em ambientes desconhecidos. Essas rotas podem ser salvas e utilizadas novamente como, por exemplo, na volta do próprio robô ao seu ponto de origem. Isso contribui na busca do caminho mais curto ou em novas pesquisas naquele ambiente.

1.2 Problema

A robótica móvel é uma área de pesquisa que vem crescendo e obtendo grande atenção. Um dos principais problemas na área da robótica é o mapeamento de rota, pois consiste na criação do mapa percorrido pelo robô na parte interna do ambiente. Para que isso ocorra precisamos desenvolver robôs móveis terrestres autônomos que sejam capazes de interagir com o ambiente em que se encontram, tomar decisões corretas para que suas tarefas sejam executadas com precisão. O desenvolvimento destes sistemas autônomos consiste em várias áreas multidisciplinares sendo muitas delas relativamente recentes como: inteligência artificial, aprendizado

de máquina, sistemas embarcados.

Com isso, o presente trabalho busca proporcionar o deslocamento e mapeamento de rotas, que tem como principal finalidade desbravar um ambiente desconhecido, mapeando a trajetória percorrida.

O protótipo pode ser utilizado em diversos lugares, principalmente em lugares onde o GPS não funciona em perfeitas condições, como por exemplo: tuneis, tubulações de ar, esgoto, cavernas, respeitando as suas limitações físicas.

O mapeamento de rotas é uma capacidade básica, que serve de suporte para que outras tarefas mais complexas possam ser executadas como: uma nova navegação naquele ambiente, localização e planejamento de trajetória. Esse plano de pesquisa propõe o estudo e a implementação de uma técnica de mapeamento métrico de rota utilizando um robô móvel terrestre.

1.3 Abordagem

Nesse trabalho foi desenvolvido um protótipo robótico móvel terrestre autônomo com a finalidade de se locomover em ambientes desconhecidos desbravando e mapeando o seu deslocamento naquele ambiente, descrevendo as suas funcionalidades e onde pode ser possível a sua aplicação pelo usuário.

Foi utilizada a IDE (*Integrated Development Environment*) do Arduino para a construção e embarcação dos *softwares* nos microcontroladores Arduino, pois eles são responsáveis pela parte de processamento das informações recebidas dos sensores, ativamente dos motores de passo e envios de mensagens para base. A linguagem de programação *Processing* é responsável por receber as informações enviadas à base pela porta serial, montando o mapa e mostrando para o usuário a trajetória feita pelo protótipo.

1.4 Objetivos

1.4.1 Geral

Desenvolver um protótipo robótico móvel terrestre, dotando-o de sensores, microcontroladores e comunicação de radio frequência que seja capaz de se deslocar em ambientes desconhecidos, mandando informações para base sobre o seu deslocamento. A base recebe essas informações e monta o mapa gráfico do deslocamento do protótipo.

1.4.2 Especifico

- Montar um protótipo robótico móvel terrestre;
- Estudar a plataforma Arduíno e sua linguagem de programação;
- Utilizar os mecanismos de sensoriamento que auxiliarão no controle do robô;
- Utilizar o mecanismo de comunicação em Radio Frequência (RF);
- Desenvolver os *sketches* para serem embarcados na plataforma Arduíno;
- Desenvolver um *sketch* na linguagem de programação *Processing* usada para construir o mapa de rota;

1.5 Organização do Documento

Este documento está organizado em seis capítulos:

O Capítulo 1 refere-se à introdução do trabalho que se encontra subdividida em tópicos como a motivação para realizá-lo, o problema, a abordagem, o objetivo geral e os específicos.

O Capítulo 2 traz o embasamento para a aplicação através do referencial teórico, abordando trabalhos e estudos já consolidados por outros autores e estudiosos da área da robótica, navegação autônoma, localização e mapeamento.

No capítulo 3 é descrita a concepção do projeto, apresentação e conceitos dos componentes eletrônicos utilizados no desenvolvimento deste trabalho, que influenciaram para que o estudo pudesse ser realizado.

No capítulo 4 é mostrada a implementação do robô móvel e da base utilizados neste trabalho, bem como, o *sketch* produzido para a construção do mapa.

No capítulo 5 são mostradas as possíveis ações realizadas pelo robô. São apresentados também os resultados obtidos durante os testes, tanto na fase de desenvolvimento do robô móvel, quanto do término de sua construção.

No capítulo 6 é apresentada a conclusão, abordando alguns aspectos que influenciaram o desenvolvimento do projeto e propostas para trabalhos futuros.

2 Referencial Teórico

2.1 Robótica

Em Zenelato (2014), encontra-se uma descrição que fala da interdisciplinaridade da robótica. Segundo ele, a robótica é uma ciência que combina em seus estudos conceitos de física, matemática, mecânica, eletrônica, *design*, inteligência artificial e programação. O objetivo da robótica é compreender todo o processo de montagem e controle de "sistemas que interagem com o mundo real com pouca ou mesmo nenhuma intervenção humana"(MARTINS, 1993). A esses sistemas, que precisam processar informações para realizarem ações específicas, dá-se o nome de robô.

Robótica é um ramo da tecnologia que engloba mecânica, eletrônica e computação, que atualmente trata de sistemas compostos por máquinas e partes mecânicas automáticas e controladas por circuitos integrados, tornando sistemas mecânicos motorizados, controlados manualmente ou automaticamente por circuitos elétricos, tendo também passagem por uma importante e atual área da computação a IA (Inteligência Artificial). As máquinas, pode-se dizer que são vivas, mas ao mesmo tempo são uma imitação da vida, não passam de fios unidos e mecanismos, isso tudo junto concebe um robô que para uma linguagem técnica é definido como um mecanismo que recebe informações do meio através de sensores, as processa com o uso de um processador e por fim executa uma tarefa através de seus atuadores (em sua maioria utilizados na robótica simples, motores elétricos) (CRISTIANO, 2010).

Como grande visionário que foi Azimov escreveu em seu livro “Eu, Robô”, conto este que ficou bem conhecido por causa do lançamento de um filme de mesmo nome, as diretrizes básicas que regem a convivência das máquinas com os seres humanos:

1. Um robô não poderá prejudicar um ser humano ou através de sua inatividade permitir que este esteja em perigo;
2. Um robô terá de obedecer às ordens dadas pelo ser humano, a menos que estas contradigam a primeira lei;
3. Um robô deverá preservar a própria existência a menos que isto contradiga a primeira e a segunda lei;

A quarta lei, considerada a de maior prioridade, foi lançada tempos após a divulgação

das três primeiras, e diz que um robô não pode prejudicar a humanidade nem permitir que ela o faça (OLIVEIRA, 2001).

2.1.1 Robôs

Os robôs são agentes físicos que executam tarefas manipulando o mundo físico. Para isso, eles são equipados com efetadores como pernas, rodas, articulações e garras. Os efetadores têm um único propósito: exercer forças físicas sobre o ambiente. Os robôs também estão equipados com sensores que lhes permitem perceber seu ambiente (RUSSELL, 2004).

Os sensores fazem a função dos nossos sentidos, fazendo com que, o robô perceba o ambiente. Para isso existem vários tipos de sensores que podemos citar: sensor de toque, que tem a função de medir a força adotada pelo robô no ambiente, temperatura, que mede a sensação térmica do ambiente, umidade, verifica a umidade do ar, ultrassônico, que é utilizado para medir o ambiente, giroscópio e acelerômetros, que servem para medir os movimentos do próprio robô, inclusive sensores câmeras que dão noção do ambiente, entre outros que existem.

Para cada tipo de situação pode existir um robô com finalidade específica, onde o robô sempre realizará os mesmos movimentos ou um único movimento conforme foi programado, podendo ser adaptável, agindo e reagindo de acordo com estímulos recebidos do ambiente ao qual está inserido.

Segundo Russell (2004), a maior parte dos robôs que hoje se enquadra em uma das três categorias principais. Os manipuladores, ou braços robôs, são aqueles que estão fisicamente ancorados (ou fixos) em seu local de trabalho, como por exemplo, em uma linha de montagem industrial ou na estação espacial internacional. O movimento do manipulador em geral envolve uma cadeia inteira de articulação controláveis, permitindo que esses robôs coloquem seus efetadores em qualquer posição dentro do local de trabalho, ver figura 1.



Figura 1 – Robôs de Hoje em uma Linha de Produção

Fonte: <<http://robotica-12c.com.br/Robotica.htm>>

A segunda categoria é o robô móvel. Os robôs móveis são aqueles que conseguem se

deslocar em um ambiente usando rodas, pernas, esteiras, hélice ou mecanismos semelhantes. Podendo ser classificado em quatro tipos:

- Veículos terrestres não-tripulado (ULV - *unmanned land vehicle*) robôs capazes de realizar a navegação autônoma sem condutor em auto-estradas, ver figura 2.
- Veículos aéreos não-tripulados (UAV - *unmanned air vehicle*) comumente usados para vigilância, pulverização de lavouras e operações militares, ver figura 3.
- Veículos autônomos subaquáticos (AUV - *autonomous underwater vehicle*) usados em exploração no fundo do mar, ver figura 4.
- E por último, os viajantes interplanetários, como o robô Sojourner, ver figura 5.



Figura 2 – Robô Móvel Inteligente

Fonte: <<http://icmc-usp.blogspot.com.br/2011/04/palestra-sobre-robos-moveis.html>>



Figura 3 – Veículo Aéreo Não-Tripulado em Operação Militar

Fonte: <http://www.ael.com.br/vant_skylark.php>



Figura 4 – *Véículo Autônomo Subaquático*

Fonte: <<http://revistapesquisa.fapesp.br/2013/04/25/poli-usp-desenvolve-veiculo-autonomo-subaquatico-2/>>

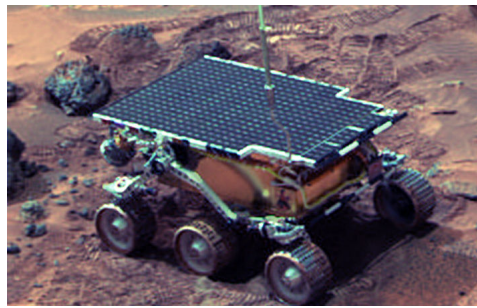


Figura 5 – *Robô Sojourner*

Fonte: <http://solarsystem.nasa.gov/scitech/display.cfm?ST_ID=2492>

O terceiro tipo são os híbridos: um robô móvel equipado com manipuladores. Eles incluem o robô humanoide, cuja estrutura física imita o torso humano, ver figura 6.



Figura 6 – *Os Robôs Humanoides P3 e Asimo da Honda*

Fonte: <http://www.21stcentury.co.uk/robotics/honda_asimo_robot.asp>

As ações feitas pelos robôs são instruções declaradas na fase de construção ou adicionadas de acordo com as necessidades. As instruções são definidas via um *software*, e depois armazenadas em uma memória do circuito integrado (CI) juntamente com o processador para que sejam executadas.

A construção e programação de robôs geralmente baseiam-se na simulação de ações humanas. Ou seja, têm o objetivo de executar processos repetitivos humanos. Contudo o formato e os processos não precisam estar caracterizados pelo fazer ou parecer humano. Uma máquina fixa e que controla e regula automaticamente processos mecânicos e eletrônicos, captando informações no e sobre o ambiente, pode ser considerado um robô. Atualmente o conceito de Inteligência Artificial se complexifica e se dinamiza nas mais diferentes áreas do conhecimento. Incorporar conceitos, técnicas e ideias novas aos sistemas autorregulados é desafio constante e que ganha proporções incomensuráveis nas tentativas de fazer o robô “aprender” características no ambiente com que “interage”. Cada vez mais buscamos construir robôs que possam “interagir” com ambientes cada vez menos pré-configurados, (IVOTE, 2012, p. 2).

2.1.2 História da Robótica

Já a palavra robótica foi utilizada pela primeira vez com o escritor russo, naturalizado americano, Isaac Azimov, no livro intitulado “Runaround” em 1921. Ele foi responsável por diversas publicações de ficção e publicações científicas. É considerado por vários especialistas o pai da robótica por ter uma visão muita além da sua época, descrevendo em seus vários textos ideias de como seria a tecnologia utilizada no futuro, principalmente no desenvolvimento de robôs, acertando grande parte das “previsões” feitas (MARTINS, 2012).

O conceito de robôs é dado por volta do início da história, quando os mitos faziam referências a mecanismos que ganhavam vida. Começando na civilização grega, os primeiros modelos de robôs encontrados, eram figuras com aparência humana ou de animal, ver figura 7, usavam sistemas de pesos e bombas pneumáticas, mas que não tinham nenhuma necessidade prática ou econômica e nenhum sistema complexo de produtividade que exigisse a existência deste tipo de aparelho.

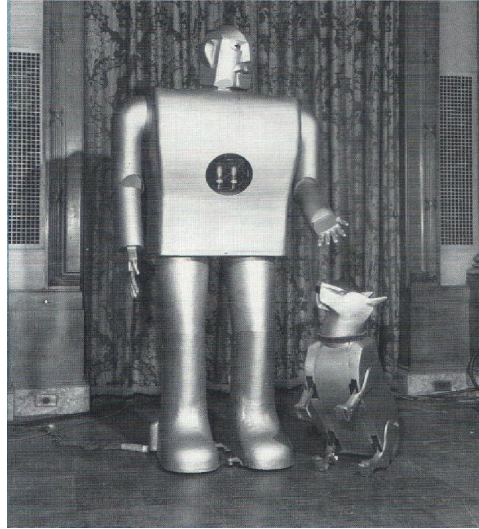


Figura 7 – Robô Elektro e Sparko o Cão Robô

Fonte: <<http://cyberneticzoo.com/?tag=westinghouse>>

Mais tarde, cientistas árabes acrescentaram um importante e novo conceito à ideia tradicional de robôs, concentrando as suas pesquisas no objetivo de atribuir funções aos robôs que fossem ao encontro das necessidades humanas. Como o homem, desde os primórdios do tempo, buscou soluções para facilitar, cada vez mais, a sua vida, a história da robótica surgiu paralelamente a essas necessidades. O homem foi construindo mecanismos que pudessem ocupar seus lugares em determinados trabalhos e com isso, podendo ter mais tempo para se dedicar a outras tarefas mais importantes, deixando de lado as tarefas mais pesadas e repetitivas, para os seus mecanismos "robotizados".

O desenvolvimento inicial dos *robots* baseou-se no esforço de automatizar as operações industriais, o qual começou no século XVIII, na indústria têxtil, com o aparecimento dos primeiros teares mecânicos, ver figura 8. Com o contínuo progresso da revolução industrial, as fábricas procuraram equipar-se com máquinas capazes de realizar e reproduzir, automaticamente, determinadas tarefas (ROBÓTICA, 2009).

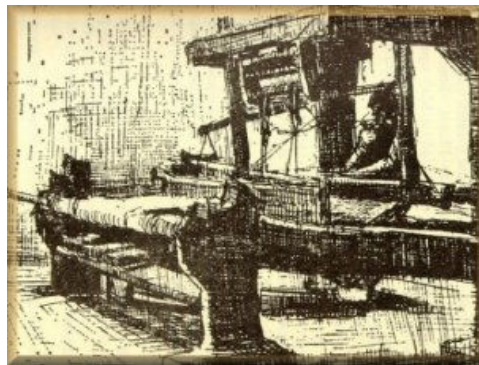


Figura 8 – Primeiro Tear Mecânico

Fonte: <<http://www.discursus.xpg.com.br/archistx/tearhis.html>>

O primeiro robô automático surgiu em 1954, projetado por George Devol. Com base no projeto de Devol a empresa Unimation Inc. concedeu um robô chamado “*Unimate*”. O *Unimate* começou a funcionar na linha de produção da General Motors em 1961. Trabalhava pegando pedaços quentes de metal e colando as peças nos chassis dos carros. Ele pesava 1.800 Kg e obedecia a comandos gravados em fitas magnéticas, ver figura 9.



Figura 9 – *Unimate, Primeiro Robô Industrial*

Fonte: <<http://nabalsa.blogspot.com.br/2010/06/historia-da-robotica.html>>

O *Unimate* foi o início de uma série de eventos que alavancaram as pesquisas e a criação de novos robôs. Em 1966, Joseph Weizenbaum lança, no MIT, o Eliza, primeiro programa de Inteligência Artificial. Três anos depois, Victor Scheinman, um estudante de engenharia mecânica do *Stanford Artificial Intelligence Lab* (SAIL) cria um braço mecânico chamado de *Stanford Arm*. Este braço se transforma em um padrão e até hoje influencia o *design* de braços e robôs. A corrida de descobertas continuava e em 1981, o engenheiro Takeo Kanade desenvolve e monta o primeiro braço mecânico com motor instalado diretamente nas junções do braço. Esta mudança faz com que os movimentos se tornem mais rápidos e precisos.

Em 1986, a Honda inicia suas pesquisas para a construção de um robô, que segundo palavras da própria empresa "deveria coexistir e cooperar com os humanos, fazendo aquilo que as pessoas não conseguem e cultivando uma nova dimensão de mobilidade, que tem como princípio beneficiar a sociedade". O Asimo, um autêntico humanoide, resultado destas pesquisas é lançado em 2.000. Um ano antes, a Sony coloca no mercado o Aibo, que seria o primeiro de uma série de robôs animais que chegariam ao mercado (AYRES, 2007).

Nos últimos anos, tem-se observado um vertiginoso crescimento do potencial de sistemas robóticos. Numa primeira etapa, houve um grande desenvolvimento na área de robótica industrial, com a utilização, sobretudo de robôs manipuladores. Numa segunda etapa de evolução, pesquisadores em robótica têm concentrado esforços na construção de robôs móveis, introduzindo as capacidades de mobilidade e autonomia para reagir adequadamente ao ambiente, onde abre um vasto campo de novas aplicações e conseqüentemente, muitos desafios.

2.2 Navegação Autônoma

Segundo Yukinobu (2010), a robótica móvel autônoma é uma área recente que possui como objetivo um mecanismo capaz de executar tarefas sem a necessidade de um controlador humano. Desta forma a robótica móvel terrestre defronta com três problemas fundamentais, sendo eles: mapeamento de ambientes, localização e navegação do robô.

Segundo Xavier (2012), a navegação autônoma de robô é atualmente um dos maiores desafios para os pesquisadores desta área. Do ponto de vista biológico, para se locomover de forma inteligente em um ambiente inicialmente desconhecido, faz-se necessário o mapeamento do ambiente, o que possibilitará encontrar melhores trajetórias para o cumprimento de uma determinada tarefa. Dentre as formas de mapeamento existentes, um dos métodos mais amplamente utilizados é o mapeamento métrico probabilístico, denominado “grade de ocupação”, proposto, inicialmente, por Elfes, e utilizado com sucesso em diversas pesquisas.

O controle da trajetória de robôs móveis autônomos é também considerado uma importante área de pesquisa. Para o desenvolvimento do processo de aprendizado de trajetórias são necessárias técnicas que possibilitem a comunicação do robô e o ambiente, permitindo escolher as melhores ações a serem executadas durante o aprendizado de caminhos. A teoria da análise do comportamento demonstra que os organismos vivos são capazes de aprender por meio de interações com o ambiente, recebendo estímulos reforçadores e punições em resposta às suas ações. Para o controle de trajetórias, esses estímulos podem modificar diretamente o comportamento do robô durante o processo de aprendizagem, reforçando ou inibindo determinados caminhos.

Alguns dos primeiros dispositivos de navegação para robôs foram desenvolvidos na década de 60. Eram robôs projetados para navegar em ambientes com obstáculos demarcados, onde tomavam decisões de navegação de acordo com seu objetivo principal. Seus sensores atualizavam as informações que eles tinham sobre o ambiente externo e ajudavam na construção de novos ambientes.

2.3 Localização

Localização consiste em estimar a posição de um robô no ambiente. Determinar sua própria posição é uma capacidade básica para que qualquer tarefa de navegação seja executada. É fundamental que um robô consiga determinar exatamente a posição em que se encontra dentro de um ambiente para que o mesmo possa planejar a uma trajetória até o seu destino e cumprir as tarefas que lhe forem alocadas (FOX, 1999).

A grande maioria dos robôs móveis possui um sistema de odometria que permite estimar

o deslocamento dos mesmos e conseqüentemente, calcular a posição dos robôs no ambiente. Normalmente, as informações odométricas são obtidas por meio de sensores acoplados às rodas dos robôs, de forma a calcular o deslocamento dos mesmos a partir do movimento destas. Na prática, existem diversos fatores que causam erros nesse tipo de dispositivo como a imprecisão mecânica do sistema e irregularidades do terreno em que o robô atua. Outro fator que dificulta e pode até inviabilizar a localização do robô a partir da odometria é o fato de que os pequenos erros causados durante o cálculo da posição do robô vão se acumulando ao longo do tempo.

Devido à sua grande importância dentro da área de robótica móvel, o problema da localização vem sendo amplamente estudado pelos pesquisadores da área e, atualmente, existem algoritmos bastante eficientes na solução do mesmo. Normalmente, é fornecido um mapa do ambiente e a localização é estimada com base nas informações fornecidas pelos sensores. O grande desafio na solução do problema de localização está no fato de que tanto as informações sobre o ambiente como os dados fornecidos pelos sensores são normalmente limitados e imprecisos (AUGUSTO, 2009).

2.4 Mapeamento

O levantamento de dados sobre o ambiente e a conseqüente construção de mapas (criação de uma representação do ambiente onde está inserido o robô), consiste em uma tarefa fundamental para o desenvolvimento de robôs móveis autônomos. No entanto esta sempre foi uma tarefa difícil. Normalmente, tarefas básicas executadas por um robô necessitam de uma representação do ambiente para serem executadas. Supondo, por exemplo, que um robô deva planejar uma trajetória para se locomover de sua posição atual até uma posição de destino. Nesse caso, um mapa do ambiente é fundamental para que o caminho mais eficiente seja encontrado. Utilizando o mapa, o robô verifica os possíveis caminhos que levam até a posição desejada e os obstáculos que devem ser evitados (THRUN, 2005).

Outra aplicação robótica que normalmente necessita de um mapa é a localização, que consiste em estimar a posição de um robô em um ambiente previamente conhecido. Dentro do contexto da navegação é necessário que o robô saiba com precisão sua localização dentro do ambiente para que o mesmo possa determinar uma trajetória até o ponto de destino. Uma vez que os sensores odométricos dos robôs comerciais normalmente apresentam algum grau de imprecisão, são utilizadas informações dos sensores, juntamente com um mapa do ambiente, para se estimar a posição do robô (AUGUSTO, 2009).

Existem também técnicas que permitem que o mapa seja criado ao mesmo tempo em que o robô se localiza no ambiente. Essas técnicas são chamadas de localização e mapeamento simultâneos (SLAM - *Simultaneous Localization And Mapping*), e são consideravelmente mais

complexas do que técnicas de mapeamento e localização em separado. Existem basicamente dois tipos de mapas criados por robôs móveis: topológicos e métricos. Os mapas topológicos representam o ambiente por grafos, no qual os vértices representam regiões de interesse no ambiente e os arcos representam as vias que interligam essas regiões. Os mapas topológicos são bastante eficientes para se estimar trajetórias em um ambiente, mas apresentam uma representação muito pobre do ambiente físico. Os mapas métricos representam os ambientes físicos em detalhes e também podem ser utilizados para se estimar trajetórias.

3 Concepção do Projeto

O presente projeto consiste na construção de um protótipo robótico móvel terrestre dotado de sensores para a percepção física do ambiente em que está atuando, mandando informações como distância dos obstáculos para um microcontrolador Arduino. Essas informações são recebidas, processadas e encaminhadas para outros microcontroladores, sendo um deles responsável por controlar os atuadores (motores de passo, responsável pela locomoção) e o outro responsável por fazer a comunicação entre o protótipo e a base. A base ao receber essas informações manda para o computador, onde são processadas e transformadas em um mapa. Este mapa contém toda a trajetória feita pelo protótipo dentro daquele ambiente.

A primeira parte da construção do projeto está descrito no item 3.1, esta seção contempla a construção do protótipo robótico e os componentes utilizados na sua engenharia. A segunda parte é a construção do projeto da base, descrita no item 3.2.

3.1 Veículo Móvel

A estrutura física do veículo móvel é composta por três placas de circuito controladora (sendo uma placa de Arduino Uno com microcontrolador ATmega328 e duas placas de Arduino MEGA sendo uma com microcontrolador ATmega1280 e a outra ATmega2560), mais detalhes, ver seção 3.1.1, ou seja, em cada placa encontra-se um chip microcontrolador, que tem como função executar o código programado em sua memória processando as informações recebidas pelas suas portas de entradas, tomando ações controlando as portas de saída.

A trajetória que o protótipo irá seguir baseia-se nas decisões que serão tomadas pelos microcontroladores a partir das leituras feitas pelos sensores. Os sensores utilizados no protótipo foram três sonares (sensor ultrassônico HC-SR04), ver seção 3.1.2, que tem como finalidade perceber o ambiente. Eles foram distribuídos da seguinte forma: na frente para conseguir perceber os obstáculos que se encontram à frente da trajetória do protótipo, um na sua esquerda e o outro na sua direita, que terão como finalidade observar suas laterais indicando se há ou não obstáculos quando o protótipo for mudar de direção ou sentido diferente do qual o robô se encontra.

Para que o protótipo consiga se deslocar no ambiente foram adicionadas três rodas, duas delas na parte da frente, ligadas diretamente a motores de passo, ver seção 3.1.3, responsáveis

pela tração do veículo, sendo controlados pelos microcontroladores e a outra localizada na parte de trás, responsável pela sustentação, definida como omnidirecional, gira em todas as direções e sentidos.

A comunicação entre o protótipo e a base é feita através de um par de antenas de radio frequência (RF), ver seção 3.1.4, uma envia um sinal e a outra recebe. Esses componentes fazem com que o protótipo robótico consiga se deslocar em ambientes desconhecidos desviando de possíveis obstáculos que possam aparecer no decorrer de seu percurso mandando informações de sua trajetória para a base. Todos os componentes estão interligados uns com os outros através de uma placa *Protoboard* descrita no item 3.1.5.

3.1.1 Arduino

Arduino é uma plataforma de prototipagem eletrônica de código aberto baseado em *hardware* e *software* flexível e de fácil utilização. É destinado aos mais diversos tipos de pessoas e interesses, como artistas, designers ou qualquer pessoa interessada em criar objetos ou ambientes interativos (ARDUINO, 2013).

A placa de Arduino é uma das placas de circuito integrado (CI) que está sendo bastante utilizada no controle de protótipos eletrônicos, principalmente nas experimentações, pois além de ser um *software* livre possui um *hardware* modular e de baixo custo. Foi desenvolvido com o objetivo de fácil manipulação até mesmo para aqueles que não possuem experiência com o seu *software* ou *hardware*. Além disso, todo o material (*software*, bibliotecas, *hardware*) é *open-source*, ou seja, podem ser usados e reproduzidos por todos, sem a necessidade de pagar os direitos autorais.

Segundo Banzi (2012), o Arduino é formado por dois componentes principais sendo eles: a placa Arduino, elemento de *hardware* com o qual você trabalha ao construir seus objetos e a IDE que pode ser utilizada para criar um *sketch* (esboço, um pequeno programa de computador), do qual você fará o *upload* para a placa Arduino. O *sketch* dirá a sua placa o que deve ser feito.

Hardware

As placas podem ser construídas à mão ou compradas módulos, o *software* pode ser baixado gratuitamente através o *site* oficial do Arduino www.arduino.cc. Os projetos de *hardware* estão disponíveis sob uma licença de código aberto, o usuário é livre para adaptá-los às suas necessidades (ARDUINO, 2013). A figura 10 mostra uma placa Arduino UNO, que pode ser adquirida facilmente através da *internet* ou em lojas de eletrônica e robótica.

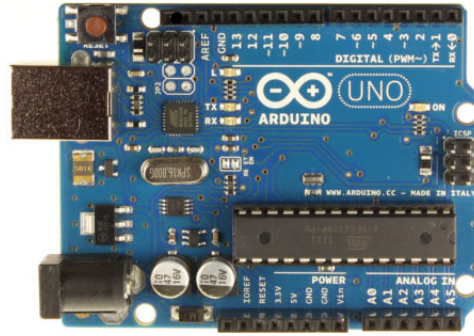


Figura 10 – Placa Arduino UNO

Fonte: http://arduino.cc/en/uploads/Main/ArduinoUno_R3_Front.jpg

As placas de Arduino contém um circuito integrado (CI) que quando adicionados códigos ou *sketch*, passam a processar informações correspondentes, gerenciando as portas de entrada e saída. O processamento dessas informações e o gerenciamento dessas portas fazem o controle e os movimentos do robô.

Existem vários tipos de Arduino, nesse trabalho foram utilizados quatro Arduino, três na construção do protótipo robótico, (sendo um Arduino UNO e dois Arduino MEGA) e um Arduino Uno na construção da base.

O Arduino Uno, ver figura 10, é uma placa de microcontrolador baseada no Atmega328, sendo constituída por 14 pinos digitais de entrada/saída (dos quais 6 podem ser usados como saídas PWM, Modulação por Largura de Pulso), e 6 entradas analógicas, um cristal oscilador de 16 MHz, uma conexão USB, um conector de alimentação, um cabeçalho ICSP, e um botão de reset, (ARDUINO, 2013), veja na tabela 1, suas principais especificações.

Tabela 1 – Especificações da Placa de Arduino Uno

Microcontrolador	ATmega328
Tensão de funcionamento	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limites)	6-20V
E / S digital Pinos	14 (dos quais 6 fornece uma saída PWM)
Pinos de entrada analógica	6
Corrente DC por I / O Pin	40 mA
Corrente DC 3.3V para Pino	50 mA
Memória Flash	32 KB (ATmega328), dos quais 0,5 KB utilizado pelo bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidade do Relógio	16 MHz

O Arduino Mega 1280, ver figura 11, é uma placa de microcontrolador baseada no ATmega1280. Ele possui 54 pinos digitais de entrada / saída (dos quais 14 podem ser usados como saídas PWM), 16 entradas analógicas, 4 UARTs (portas seriais de *hardware*), um cristal

oscilador de 16 MHz, uma conexão USB, um conector de alimentação, um cabeçalho ICSP, e um botão de reset. O mega é compatível com a maioria dos shields projetados para o Arduino Duemilanove ou Diecimila, (ARDUINO, 2013), veja na tabela 2, suas principais especificações.

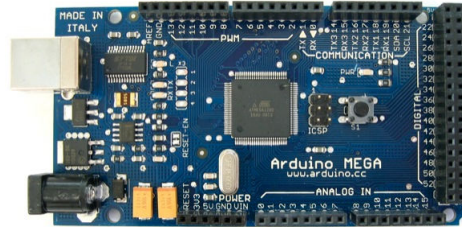


Figura 11 – Placa Arduino Mega 1280

Fonte: <<http://arduino.cc/en/Main/arduinoBoardMega>>

Tabela 2 – Especificações da Placa de Arduino Mega 1280

Microcontrolador	ATmega1280
Tensão de funcionamento	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limites)	6-20V
E / S digital Pinos	54 (dos quais 15 fornece uma saída PWM)
Pinos de entrada analógica	16
Corrente DC por I / O Pin	40 mA
Corrente DC 3.3V para Pino	50 mA
Memória Flash	128 KB da qual 4kb usados pelo bootloader
SRAM	8 KB
EEPROM	4 KB
Velocidade do Relógio	16 MHz

O Arduino Mega 2560, ver figura 12, é uma placa de microcontrolador baseado no ATmega2560. Possui 54 entradas / saídas digitais (dos quais 15 podem ser usados como saídas PWM), 16 entradas analógicas, 4 UARTs (portas seriais de *hardware*), a 16 MHz cristal oscilador, uma conexão USB, um conector de alimentação, um cabeçalho ICSP, e um botão de reset. Também é compatível com a maioria dos shields projetados para o Arduino Duemilanove ou Diecimila, (ARDUINO, 2013), veja na tabela 3, suas principais especificações.

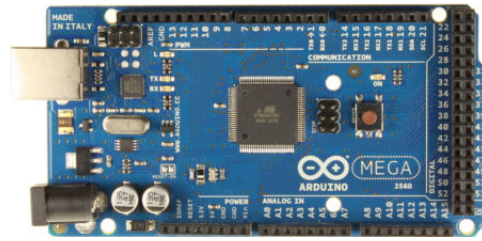


Figura 12 – Placa Arduino Mega 2560

Fonte: <<http://arduino.cc/en/Main/arduinoBoardMega2560>>

Tabela 3 – Especificações da Placa de Arduino Mega 2560

Microcontrolador	ATmega2560
Tensão de funcionamento	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limites)	6-20V
E / S digital Pinos	54 (dos quais 15 fornece uma saída PWM)
Pinos de entrada analógica	16
Corrente DC por I / O Pin	40 mA
Corrente DC 3.3V para Pino	50 mA
Memória Flash	256 KB da qual 8kb usados pelo bootloader
SRAM	8 KB
EEPROM	4 KB
Velocidade do Relógio	16 MHz

Software

A IDE do Arduino, além de ser um *software* de código fonte aberto, existem diversas versões, compatíveis com o *Windows*, *Mac OS X* e *Linux*. A IDE utilizada neste trabalho foi à versão 1.01, para ambiente *Windows* de 32 *Bits*, ver figura 13, foi adquirida no *site* www.arduino.cc.

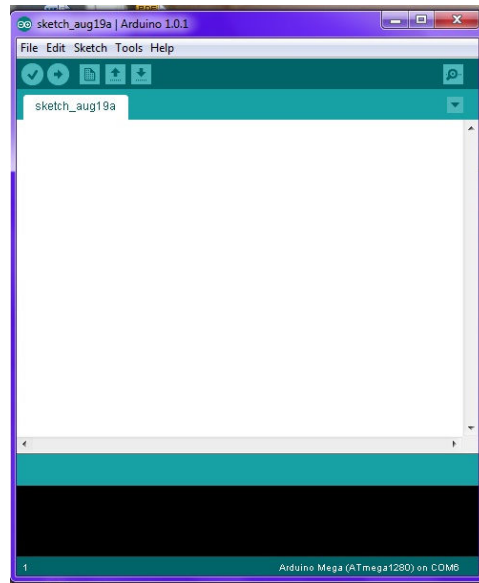


Figura 13 – IDE 1.0.1 do Arduino

Fonte: o Autor

O seu ambiente de desenvolvimento contém um editor de texto, que é utilizado para escrever códigos ou programas, esses programas escritos usando a IDE do Arduino são chamados de *sketches*, possui uma área de mensagens, um console de texto, uma barra de ferramentas com vários botões com diferentes funções específicas, e uma série de menus. A mesma ao se conectar ao *hardware* do Arduino, permite o *upload* dos *sketches* fazendo a comunicação entre eles (ARDUINO, 2013).

Essas *sketches* são feitas no editor de texto, e são salvas com a extensão ".ino", exemplo "arquivo.ino". Elas permitem algumas funcionalidades como recortar, colar, procurar e substituir texto. No espaço reservado para a área da mensagem, temos retorno de ações como salvar e exportar, além da exibição de erros. O console mostra a saída de texto do Arduino, incluindo mensagens de erro completas e outras informações. No canto inferior direito mostra a *board* atual e a porta serial. Os botões da barra de ferramentas permitem que possa ser verificado ou que faça o *upload* de programas, crie, abra ou salve os *sketches*, além de poder abrir monitor serial/ serial monitor (ARDUINO, 2013).

A plataforma de implementação (IDE do Arduino) é baseada nas linguagens C/C++, preservando a sua sintaxe em alguns aspectos, tais como: na declaração de variáveis, na utilização de operadores, na manipulação de vetores, na conservação de estruturas, bem como é uma linguagem sensível ao caso/ case-sensitive, mais ao invés de uma função *main()*, o Arduino necessita de duas funções elementares, sendo elas: *setup()* e *loop()* (ARDUINO, 2013).

3.1.2 Sensores

Os sensores são a interface perceptiva entre o robô e seus ambientes. Os sensores ativos enviam energia ao ambiente como, por exemplo, os sonares. Eles contam com o fato de que essa energia é refletida de volta para o sensor, mas ao custo de maior consumo de energia e com um perigo de interferência, quando vários sensores ativos são usados ao mesmo tempo (RUSSELL, 2004).

Os sonares trabalham com o conceito de ondas sonoras, e o eco é a propriedade, das ondas sonoras, que é à base de funcionamento dos sonares. Se você gritar na beira de um poço ou de um penhasco, você ouvirá a sua voz alguns segundos depois de ter gritado. Isso acontece porque as ondas sonoras que compõem seu grito refletem em alguma superfície e fazem todo o caminho de volta até o local onde você se encontra. Isso é propriamente chamado de eco, o retorno do som.

O microcontrolador responsável por controlar o sensor, envia um sinal para o sonar, ver figura 14, que emite um conjunto de ondas sonoras e ao mesmo tempo dispara um cronômetro para medir o tempo que passa até receber um eco. Recebido o eco, o microcontrolador para o cronômetro e realiza o cálculo da distância do radar até o obstáculo através da fórmula.

$$\text{distância} = (\text{velocidade-do-som} * \text{tempo-do-cronometro}) / 2$$

O resultado da multiplicação é dividido por dois, por que o som realiza o mesmo percurso duas vezes: do sonar até o objeto (ida) e do objeto até o sonar (volta).

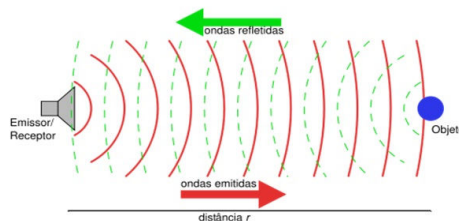


Figura 14 – Atuação do Sensor Ultrassônico

Fonte: <<http://www.mecatrons.xpg.com.br/Artigos/ArtigoVIII/artigoVIII.html>>

O sensor utilizado no projeto foi o sensor ultrassônico HC-SR04, ver figura 15, que funciona como um detector de objetos e permite medir distâncias mínimas de 2 centímetros podendo chegar a distancias máximas de até 450 centímetros com uma precisão de 3 milímetros, veja na tabela 4, suas principais especificações..

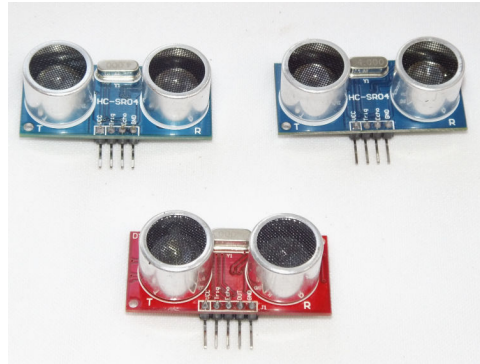


Figura 15 – Sensores Ultrassônicos

Fonte: o Autor

Tabela 4 – Especificação dos Sensores

Modelo	HC-SR04
Voltagem de Funcionamento	5V(DC)
Corrente	Menor que 2mA
Saída de Sinal	Electric frequency, high level 5V, low level 0V
Angulo de Sensor	Não maior que 15 degrees
Distância de Detecção	de 2cm a 450cm
Precisão	Acima de 3mm
Modo de Conexão	VCC/trig(T)/echo(R)/GND

3.1.3 Motor de Passo

Os Motores de Passo são dispositivos eletromecânicos que convertem pulsos elétricos em movimentos mecânicos que geram variações angulares discretas. O rotor ou eixo de um motor de passo é rotacionado em pequenos incrementos angulares, denominados “passos”, quando pulsos elétricos são aplicados em uma determinada sequência nos terminais deste. A rotação de tais motores é diretamente relacionada aos impulsos elétricos que são recebidos, bem como a sequência na qual os pulsos são aplicados reflete diretamente na direção que o motor gira. A velocidade que o rotor gira é dada pela frequência de pulsos recebidos e o tamanho do ângulo rotacionado é diretamente relacionado com o número de pulsos aplicados (GONCALVES, 2008).

Um motor de passo pode ser uma boa escolha sempre que movimentos precisos são necessários. Eles podem ser usados em aplicações onde é necessário controlar vários fatores tais como: ângulo de rotação, velocidade, posição e sincronismo. O ponto forte de um motor de passo não é a sua força (torque), tampouco sua capacidade de desenvolver altas velocidades ao contrário da maioria dos outros motores elétricos, mas sim a possibilidade de controlar

seus movimentos de forma precisa. Por conta disso este é amplamente usado em impressoras, scanners, robôs, câmeras de vídeo, brinquedos, automação industrial entre outros dispositivos eletrônicos que requerem precisão (GONCALVES, 2008).

O motor utilizado no protótipo, foi o motor de passo 28BYJ, ver figura 16, ele é um motor que possui grande precisão, alto torque e uma baixa velocidade, podendo ser alimentado com uma voltagem de 5V. Possui uma grande redução, equivalente à 1/64 que o torna incrivelmente preciso com 2048 passos por resolução. Acompanhado com *MotorShield* com ULN2003A que possui entradas para a alimentação do motor e entradas para sinais de controle do motor de passo, veja na tabela 5, suas principais especificações.

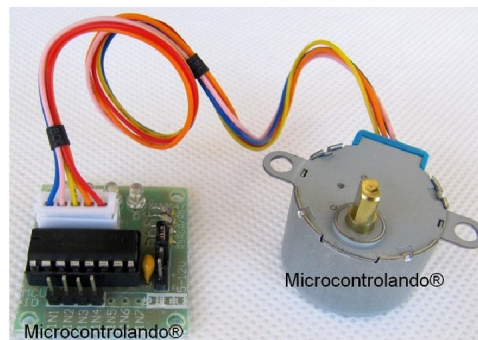


Figura 16 – Motor de Passo Conectado ao MotorShield

Fonte: <http://produto.mercadolivre.com.br/MLB-455383887-motor-de-passo-cabo-motorshield-shield-codigo-arduino-_JM>

Tabela 5 – Especificação dos Motores de Passo

Motor de Passo	28BYJ
Diâmetro	28 milímetros
Tensão	5V
Número de polos	4
Ângulo de passo	5,625 x 1/64
Relação de redução	01/-64
DC Resistencia	Ageitar
Frequência em marcha lenta de tração	>600Hz
Frequência quando ocioso	>1000Hz
Torque de tração	>34.3mN.m(120Hz)
Auto posicionamento	>34.3mN.m
Resistencia isolada	Ageitar
Energia isolada	600VAC/1mA/1s
Grau de isolamento	A
Aumento de temperatura	<40k(120Hz)
Ruido	<40db(120Hz,No load,10cm)

3.1.4 Comunicação de Radio Frequência (RF)

A abreviação de Radio frequência, RF se refere às frequências compreendidas na faixa do espectro das ondas de rádio. Quando aplicadas a uma antena, as correntes de RF criam campos eletromagnéticos que se propagam pelo ar. Todo campo de RF possui uma largura de onda que é proporcional ao inverso de sua frequência. Os módulos de Tato utilizam uma frequência de 433.92 MHz, o que corresponde a uma largura de onda de 0,69 metros. A frequência de 433MHz é classificada na faixa de UHF que é definida de 300MHz – 3GHz, veja na tabela 6, suas principais especificações.

As antenas de Radio Frequência, ver figura 17, podem ser utilizadas em diversas ocasiões como, controle remoto de robôs, aquisição de dados sem fio, sensores sem fio.

Tabela 6 – Especificação dos Módulos RF

Taxas de transferência	Alta (2400 – 19200 Bauds dependendo do controlador)
Conector	Padrão barra de pinos SIP, ideal para testes em protoboard
Compatível com	Todos os BASIC Step e outros microcontroladores
Fácil de utilização	Apenas um comando SEROUT é necessário
Modo de economia de bateria	Power Down
Alcance	150 metros com linha de visão



Figura 17 – Módulos de RF

Fonte: o Autor

3.1.5 Protoboard

Protoboard também conhecida como matriz de contatos foi projetada para a realização de montagens provisórias, teste de projetos entre outras inúmeras aplicações. Ela é constituída

por uma base de plástico contendo inúmeros furos, ver figura 18. Esses furos são organizados em linhas e colunas. As linhas encontram-se nas extremidades superior e inferior da base de plástico da placa, já as colunas são formadas exatamente por cinco furos cada, localizadas entre as linhas. Na parte inferior, abaixo da base de plástico, da *protoboard*, são instalados contatos metálicos interligados seguindo um padrão básico, ver figura 19, (ELETRÔNICA DIDÁTICA, 2012).

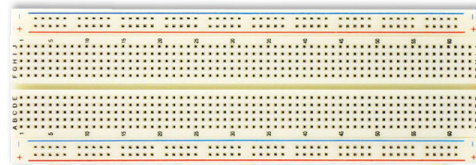


Figura 18 – *Protoboard Externamente*

Fonte: <<http://www.eletronicadidatica.com.br/equipamentos/protoboard/protoboard.htm>>

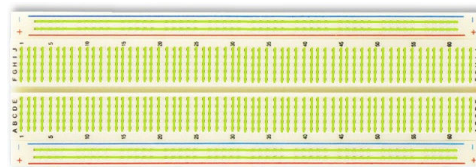


Figura 19 – *O Diagrama Interno da Ligações de uma Protoboard*

Fonte: <<http://www.eletronicadidatica.com.br/equipamentos/protoboard/protoboard.htm>>

As filas são reservadas para fazer a conexão entre a alimentação da bateria e os dispositivos eletrônicos inseridos nas colunas. O diferencial das *protoboards* é que elas não utilizam solda, ou seja, os dispositivos eletrônicos, *Led*, microprocessador, resistor, são simplesmente "plugados", ver figura 20 e podem ser facilmente retirados para serem utilizados posteriormente em novas montagens. Mas, como dito antes, as *protoboards* são utilizadas apenas para testes, pois, elas têm suas desvantagens: baixa capacidade de corrente, geralmente 1A, capacidade e resistência dos contatos internos consideráveis, susceptibilidade à captação de ruídos; dentre outros fatores. Portanto, comprovado o funcionamento desejado do circuito montado nessas placas, ele (circuito) deverá ser impresso em uma placa de circuito permanente para uso definitivo, (ELETRÔNICA DIDÁTICA, 2013).

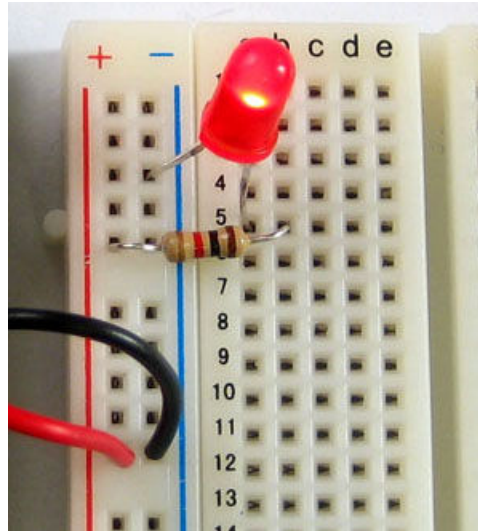


Figura 20 – Circuito Simples para Acender um LED

Fonte: <<http://www.eletronicadidatica.com.br/equipamentos/protoboard/protoboard.htm>>

3.2 Projeto da Base

A estrutura física da base é composta por uma placa microcontroladora, (Arduino Uno com ATmega328), descrito na seção 3.1.1, um *protoshield*, ver seção 3.2.1, que foi utilizado para fazer as ligações da antena RF e do *Led* no Arduino.

Foi utilizado um *notebook*, ver seção 3.2.2, para montar toda a programação e enviar para as placas de Arduino e rodar o programa *Processing*, ver seção 3.2.3, utilizado para montar o mapa da rota feita pelo protótipo de acordo com os dados recebidos do Arduino pela porta serial.

3.2.1 Protoshield

Este *protoshield*, ver figura 21, foi desenvolvido para facilitar o processo de prototipagem, encaixando-se perfeitamente sobre a placa Arduino, fornecendo ao usuário uma área de trabalho com 2 Led's de uso geral, 2 botões e todos os pinos referente ao Arduino usado devido a disposição dos barramentos. A placa pode ser utilizada com o Mini *Protoboard* ou sem, aproveitando a área de prototipagem para realizar as soldas de componentes ou fios tanto na camada superior como na inferior, veja na tabela 7, suas principais especificações.

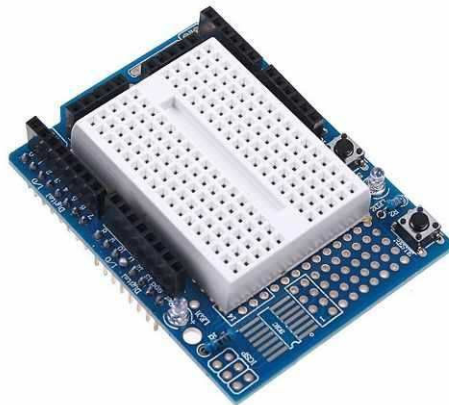


Figura 21 – Protoshield

Fonte: <http://produto.mercadolivre.com.br/MLB-482064062-prototypeprototipo-arduino-shield-mini-protoboard-_JM>

Tabela 7 – Especificações Protoshield

Qtde. Pontos	170
Cor	Azul
Principais funções:	Acesso para BlueSMiRF socket para comunicação sem fio entre Arduinos
	2 LEDs para uso em geral
	1 botão para uso em geral
	1 botão para reset do Arduino
	5V, GND no nível superior
	Todos os pinos inferiores estão na parte superior
Tamanho	Vem com adesivo para colar a Protoboard na shield
	4.3cm x 3.3cm

3.2.2 Notebook

Com o *notebook* foi feita toda a parte de programação do Arduino adicionando os códigos nas placas e testes de acordo com o andamento do projeto. Também foi feita a parte de programação do Processing e execução conforme o item 3.2.3.

No projeto foi utilizado um *notebook* com processador Intel® Pentium® Dual CPU T2390 @ 1.86GHz 1.87GHz, memória de 2,00 GB, 160 de HD, com o sistema operacional Windows 7 Ultimate, 32 Bits.

3.2.3 Processing

Processing é uma linguagem de programação com um ambiente de desenvolvimento parecido com o do Arduino, ver figura 22. O ambiente *Processing* pode ser baixado diretamente

do *site* <http://www.processing.org>, o *site* tem diversos recursos para aprendizado como tutoriais, além de uma variedade de exemplos como uma relação de livros, fórum e comunidades.

A PDE segue com menus, uma barra de ferramentas, um editor de textos, painéis para mensagens e para saída textual. Na documentação, o ambiente é chamado de PDE (*Processing Development Environment*). Os programas feitos com *Processing* são chamados de *sketches* e podem ser compostos de diversos arquivos de código e outros recursos como imagens, fontes, bibliotecas entre outros. Os *sketches* quando salvos são guardados como subdiretórios de uma pasta chamada *sketchbook*. Quando eles são executados e compilado (caso seja necessário) uma janela se abre onde é exibido o resultado. A PDE já vem com diversos exemplos incluídos que podem ser carregados e visualizados usando o menu *File > Examples*.

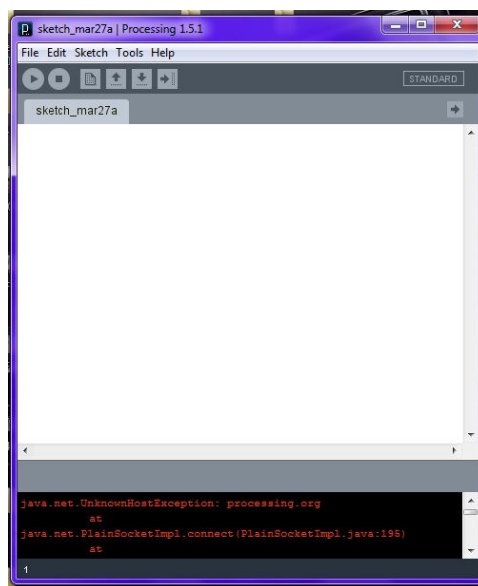


Figura 22 – PDE do Processing

Fonte: o Autor

4 Implementação

Neste capítulo são detalhados os processos de construção e desenvolvimento do protótipo robótico e da base, desde os primeiros testes até a sua versão final.

4.1 Implementação do Protótipo Robótico

Inicialmente, foi construído um chassis onde todos os outros acessórios como: motores, sensores, microcontroladores Arduino, *protoboard*, fossem acoplados de acordo com as funções que vão exercer.

O primeiro protótipo feito foi construído utilizando motores DC (*Direct Current / Corrente Contínua*) controlado por um Arduino Mega 1280, mas os motores foram descartados após alguns testes feitos na prática, pois ficou constatado que os motores não tinham como fornecer a precisão necessária para a construção de um mapa métrico do ambiente percorrido. Para resolver tal problema com sucesso os motores DC foram substituídos por motores de passo.

O segundo protótipo feito, já utilizando os motores de passo controlados por um Arduino Mega 1280, tornou-se necessária a utilização de uma biblioteca chamada *Stepper.h*, ela é quem faz o controle dos motores, veja a tabela 8.

Tabela 8 – Utilização da Biblioteca *Stepper.h*

1	<code>#include<Stepper.h></code>
2	
3	<code>const int stepsPreRevolution = 64</code>
4	<code>Stepper small_stepper(stepsPreRevolution, 2,4,3,5)</code>
5	<code>Stepper small_stepper2(stepsPreRevolution, 8,10,9,11)</code>

Na tabela 8, a linha 1 mostra, a declaração da biblioteca *Stepper.h* que é utilizada para controlar os motores de passo. A linha 3 mostra a declaração de uma variável que vai ser utilizada para especificar quantos passos o motor leva para dar uma volta completa ou 360°. Como os motores vão trabalhar de forma independente um do outro é necessário especificar em quais pinos estão conectados os dois motores, isso é feito e mostrado nas linhas 4 e 5.

A tabela 9 mostra como é feito o controle dos motores, as linhas 24 e 25 especificam qual a velocidade os motores deverão funcionar. A escolha da velocidade é dada pela precisão

da operação, sendo escolhido um valor intermediário de 250, pois quanto maior a velocidade menor a precisão.

Tabela 9 – Controle dos Motores de Passo

24	<code>small_stepper.setSpeed(250)</code>
25	<code>small_stepper2.setSpeed(250)</code>
26	<code>small_stepper.step(s)</code>
27	<code>small_stepper2.step(d)</code>

As linhas 26 e 27 contém duas variáveis (s) e (d) que vão especificar qual o sentido que os motores irão girar, valores positivos giram para o sentido horário, valores negativos para o sentido anti-horário, valor igual à zero os motores param de rodar. Para fazer as curvas é só adicionar um valor positivo em uma variável e um negativo na outra variável, de acordo com o lado que pretende que o protótipo gire, esse tipo de movimento faz com que o protótipo gire no próprio eixo.

Após o funcionamento correto dos motores de passo, os sensores foram adicionados no mesmo Arduino. Nos primeiros testes já foram notados alguns conflitos que impediam o perfeito andamento do protótipo, pois na função *void loop()*, que é utilizada como um laço de repetição, primeiramente tinha que fazer a verificação dos três sensores, verificando se possui ou não algum obstáculo à frente dos sensores. O tempo gasto nessa operação gerava um atraso notável no andamento do protótipo, a solução viável foi adicionar outro Arduino, onde um ficaria responsável por controlar os sensores e o outro por controlar os motores. A comunicação entre os Arduinos é feita através de pinos, tais pinos são ativados ou desativados de acordo com as verificações dos sensores. A figura 23, mostra a ligação feita entre os componentes.

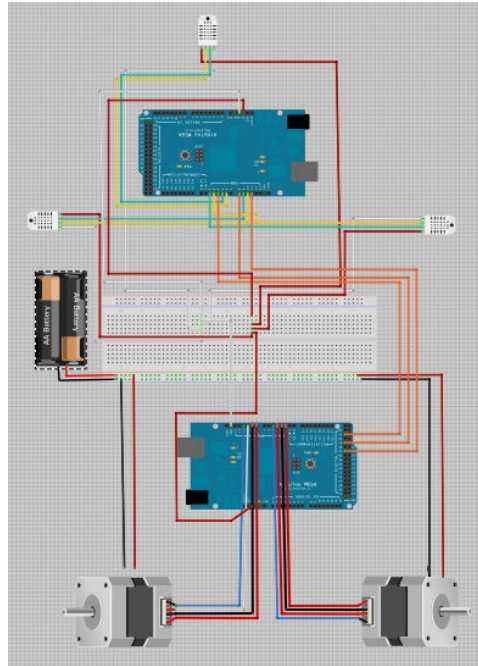


Figura 23 – *Ligação Entre os Componentes*

Fonte: o Autor

Se um sensor na sua verificação perceber que existe um obstáculo à frente com uma distância menos de 20 cm, ativa um pino correspondente ao sensor e se for superior a 20 cm ele desativa o pino. A tabela 10, demonstra essa ação correspondente ao sensor da frente do protótipo.

Tabela 10 – *Ativamento e Desativamento do Pino 8 Referente ao Sensor da Frente*

81	//sensor 1 da frente
82	if(cm1<20){
83	digitalWrite(8,LOW);
85	Serial.Println("Pino 8 Desligado);
86	}
87	else{
88	digitalWrite(8,HIGH);
90	Serial.Println("Pino 8 Ligado);
91	}

A tabela 10 mostra a parte do código que é responsável por ativar ou desativar o pino 8 referente ao sensor da frente. Na linha 82, é feita uma verificação onde examina se a variável cm1(variável que guarda o valor da distancia do sensor da frente até o obstáculo) é menor que 20, se for verdadeiro, isso afirma que o protótipo está a uma distância menor que 20 cm de um obstáculo à frente. Com essa afirmação é preciso desativar o pino 8, isso é mostrado na linha 83, caso contrario ativa o pino, na linha 88. Esses comandos de ativar e desativar também

são feitos com os outros dois sensores ativando e desativando os pinos correspondentes. Esses comandos mostram para o segundo Arduino se há obstáculos à frente dos sensores, para que ele possa tomar as melhores decisões controlando os motores para que possa desviar dos possíveis obstáculos, veja a tabela 11.

Tabela 11 – Controle dos Motores de Passo

72	<code>if(val1==0 && val2==0){//sensor da frente e direita</code>
73	<code>Serial.println("Girando 90° pra Esquerda);</code>
74	<code>s=1;</code>
75	<code>d=-1;</code>
76	
77	<code>int x=0;</code>
78	<code>while(x<1300){</code>
79	<code>small_stepper.step(s);</code>
80	<code>small_stepper2.step(d);</code>
81	<code>x++;</code>
82	<code>}</code>
91	<code>}</code>

A tabela 11 mostra uma parte do código onde o Arduino responsável, sabe para qual lado controlar os motores de acordo com os comandos dos pinos recebidos. Por exemplo, a linha 72, mostra como o protótipo sabe que existe um obstáculo à frente e a sua direita simultaneamente, pois quando os sensores ativam seus pinos significa que existe obstáculo à frente. Nesse caso os valores das variáveis `val1` (referente ao sensor da frente) e `val2` (referente ao sensor da direita) passam a ser 0, isso indica que o protótipo precisa girar para esquerda, esse movimento é feito entre as linhas 74 a 82, girando 90° à esquerda livrando o protótipo dos obstáculos.

Com os funcionamentos corretos dos sensores juntamente com os motores de passo, foi passado para a parte de comunicação de Radio Frequência (RF). Foi adicionado mais um Arduino Uno para o controle da comunicação para não atrapalhar no desempenho tanto dos motores quanto dos sensores. A comunicação entre o Arduino adicionado, também é feito através de ativamente de pinos, veja a tabela 12.

Tabela 12 – Comunicação Entre os Arduinos

72	if(val1==0 && val2==0){//sensor da frente e direita
73	Serial.println("Girando 90° pra Esquerda);
74	s=1;
75	d=-1;
76	
77	int x=0;
78	while(x<1300){
79	small_stepper.step(s);
80	small_stepper2.step(d);
81	x++;
82	}
83	
84	digitalWrite(41,HIGH);//Ligado
87	delay(200);
89	digitalWrite(41,LOW);//Desligado
91	}

A tabela 12, mostra que após a execução dos movimentos dos motores é feito o ativamente do pino 41 (referente ao movimento que ele fez, girando 90° à esquerda) por 200 milissegundos e depois o desativando, isso pode ser visualizado entre as linhas 84 a 89.

O Arduino responsável por fazer a comunicação de Radio Frequência, percebe que foi ativado um pino quando, por exemplo, a função *if()* da linha 70 da tabela 13, for verdadeira, que indica que o protótipo girou 90° a esquerda. Com essa afirmação, precisamos enviar um sinal correspondente ao movimento que foi feito para a base, essa transmissão é feita entre as linhas 74 a 76 da parte do código abaixo.

Tabela 13 – Transmissão do sinal Correspondente

70	if(val4==1){//Referente gira 90° a esquerda
71	Serial.println("--gira 90° a Esquerda--");
72	//Envia o sinal pra base;
74	const char*msg="5";
75	vw_send((unit8_t*)msg,strlen(msg));
76	vw_wait_tx();
80	}

Cada sinal enviado corresponde a ação realizada pelo protótipo. Na tabela 14, mostra os possíveis movimentos que o protótipo pode executar, sendo eles: andar para frente, girar 90° à direita, girar 90° à esquerda e girar 180°, todos os giros em relação ao próprio eixo. Em seguida os respectivos sinais são enviados à base.

Tabela 14 – Movimento e Sinal Correspondente

Frente	2
90° à Direita	3
90° à Esquerda	5
180° no próprio eixo	4

A figura 24 mostra o esquema completo da ligação dos componentes.

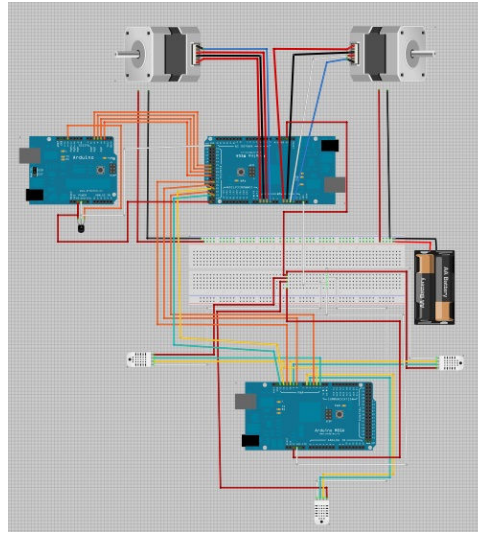


Figura 24 – Ligação Entre Todos os Componentes

Fonte: o Autor

Após a descrição da parte lógica do protótipo, outra parte que se destaca como importante é a estrutura física do veículo móvel. Será mostrada uma breve sequência de imagens que se referem à montagem e construção do robô móvel utilizado neste trabalho.

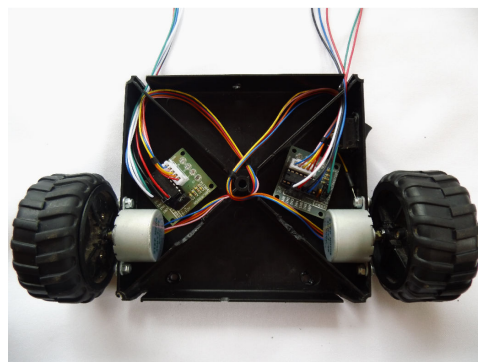


Figura 25 – Sistema de Tração do Protótipo

Fonte: o Autor



Figura 26 – Montagem do Protótipo

Fonte: o Autor

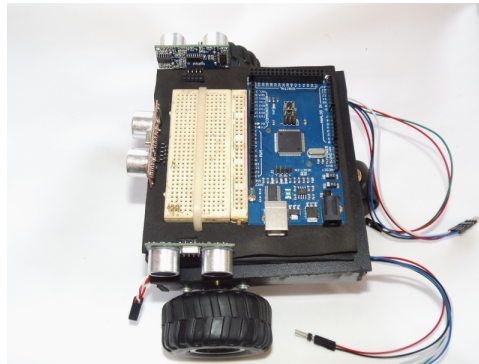


Figura 27 – Montagem do Protótipo Visão Lateral

Fonte: o Autor

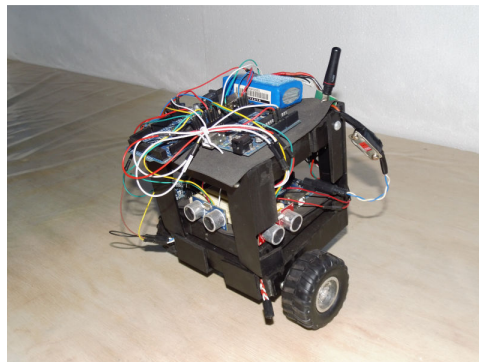


Figura 28 – Visão Lateral do Protótipo

Fonte: o Autor

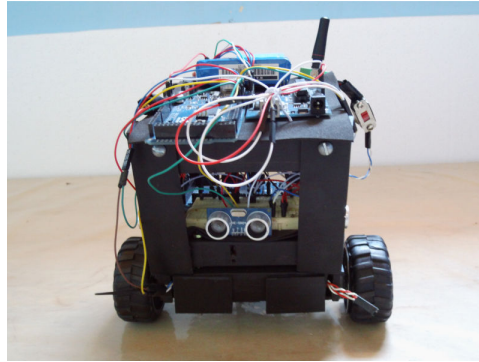


Figura 29 – Veículo Móvel Montado Visto de Frente

Fonte: o Autor

4.2 Implementação da Base

Após o protótipo robótico pronto, iniciou-se a implementação da base. A base é dividida em duas partes: a primeira parte recebe o sinal enviado pelo protótipo e envia para o computador pela porta serial, que é mostrado no item 4.2.1. A segunda parte é feita no computador, onde recebe o sinal pela porta serial transformando-o em gráfico, mostrado no item 4.2.2.

4.2.1 Recebe Sinal e Envia Pela Porta Serial

Inicialmente foi ligada uma antena de Radio Frequência ao Arduino. Para uma melhor montagem foi utilizado um *protoshield* ligando o Arduino a antena supracitada, veja a figura 30.

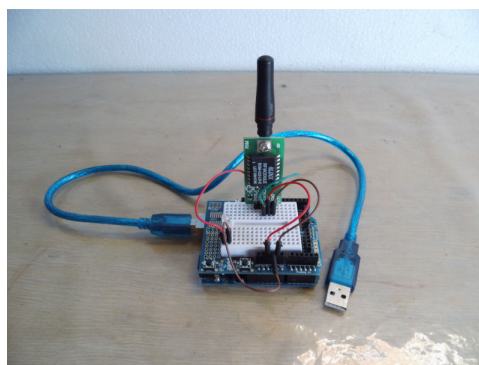


Figura 30 – Ligação da Antena RF com o Arduino

Fonte: o Autor

O objetivo da implementação é receber o sinal enviada pelo protótipo, por sua vez redirecionando para a porta serial. Segue o código abaixo na tabela 15.

Tabela 15 – Sketch que Recebe o Sinal e Envia Pela Porta Serial

1	#include<VirtualWire.h>
2	
3	char inf[30];
4	void setup(){
5	
6	Serial.begin(9600);
7	vw_set_ptt_inverted(true);
8	vw_setup(2000);
9	vw_rx_start();
10	pinMode(13,OUTPUT);
11	
12	}
14	
15	void loop(){
16	digitalWrite(13,LOW);
17	unit8_tbuf[VW_MAX_MESSAGE_LEN];
18	unit8_tbuflen = VW_MAX_MESSAGE_LEN;
19	if(vw_get_message(buf,&buflen))
20	{
21	digitalWrite(13,true);
22	info[0]=buf[0];
23	Serial.println(String(info));
24	
25	memset(&info,0,sizeof(info));
26	digitalWrite(13,false);
27	}
28	}

Na linha 1 do código é feito a declaração da biblioteca VirtualWire.h, que é uma biblioteca de comunicação de Radio Frequência. Na linha 3 é feito a declaração de um vetor, que é utilizado para armazenar o sinal recebido. Na função *setup()* na linha 9 é instanciada a *start* para começar a receber informação. Na linha 10 é instanciado um pino que tem a função de mostrar que está recebendo informações, se por acaso perder o sinal de comunicação o *led* para de piscar do instante que perdeu até a volta quando retornar a comunicação, com isso sabe quando a comunicação é perdida.

Na linha 19 é feito uma verificação se possui uma mensagem no buffer, caso possua a informação recebida é enviada pela porta serial, isso é feito na linha 23 e 25 é feito o reset da variável (*info*) para ser utilizado outra vez.

4.2.2 Recebe Pela Porta Serial e Transforma em Gráfico

Após as informações serem enviadas para a porta serial, elas serão transformadas em gráficos. Para que isso ocorra, foi utilizado um *software* denominado *Processing* descrito no item 3.2.3.

Quando executado o *sketch* do projeto abre-se uma janela igual a figura 31.



Figura 31 – Tela Inicial

Fonte: o Autor

Após o protótipo ser ligado, inicia-se o processo de comunicação entre ele e a base. Quando a base começar a receber as informações, é iniciada a construção do mapa de acordo com os dados recebidos, veja a figura 32.

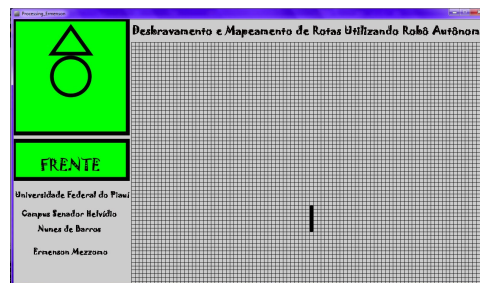


Figura 32 – Tela Referente ao Andar para Frente

Fonte: o Autor

A figura 32, mostra que a base está recebendo sinal correspondente a andar para frente. Se surgir algum obstáculo a sua frente, é realizada uma verificação nos sensores laterais, se, por exemplo, os dois estiverem livres ele escolhe um dos lados de modo aleatório e envia o sinal correspondente a sua decisão.

Se em algum caso existir a ocasião onde o protótipo se encontra com obstáculo na sua frente, na sua direita e na sua esquerda estiver livre, isso significa que o protótipo deve girar 90° no próprio eixo à esquerda, e enviar o sinal correspondente a esse movimento, veja a figura 33.

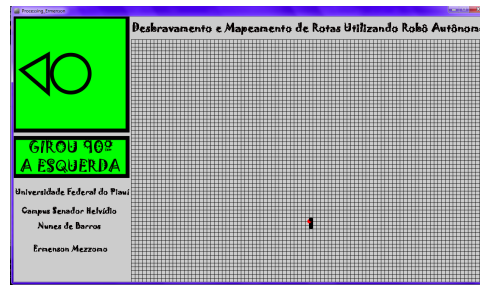


Figura 33 – Tela Referente ao Girar para Esquerda

Fonte: o Autor

A base quando analisa o sinal enviado, observa que o protótipo girou a esquerda mostrando no mapa a função correspondente e marca no gráfico um ponto vermelho que indica que naquele local o protótipo girou 90° à esquerda.

Se em outra ocasião o protótipo se deparar com obstáculos a sua frente e esquerda e na sua direita estiver livre, gira 90° no próprio eixo à direita e envia o sinal correspondente a esse tipo de movimento, veja a figura 34.

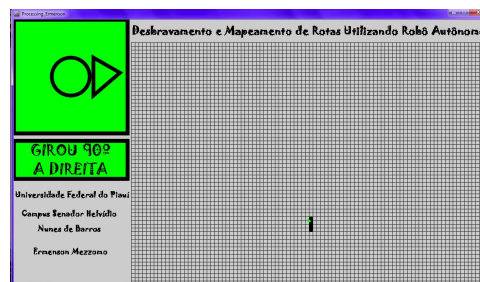


Figura 34 – Tela Referente ao Girar para Direita

Fonte: o Autor

A base quando recebe esse sinal analisa e sabe que o protótipo girou a direita. Portanto monta o mapa correspondente marcando no gráfico um ponto verde que indica que no local o protótipo girou 90° no próprio eixo à direita.

Em outra ocasião, se o protótipo se encontrar com obstáculo a sua frente, direita e esquerda, gira 180° no próprio eixo enviando o sinal correspondente a esse movimento, veja a figura 35, dando continuidade ao mapeamento.

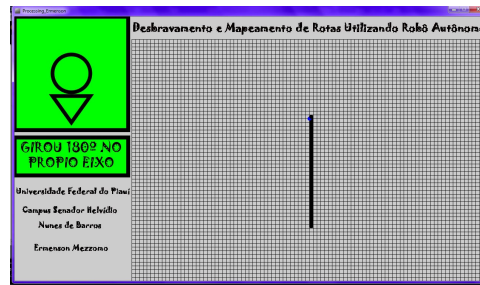


Figura 35 – Tela Referente ao Girar 180º

Fonte: o Autor

Após receber e analisar o sinal enviado percebe que o protótipo girou 180º, o que é correspondente, por exemplo, quando se depara em um beco sem saída. A base monta o mapa correspondente, marcando no gráfico um ponto azul.

O mapa é feito de acordo com o que o protótipo se deslocou dentro de um ambiente, cada quadrado deslocado no mapa é equivalente a 5cm percorrido pelo protótipo robótico. A figura 36 mostra um pequeno gráfico mostrando todos os possíveis movimentos que o protótipo pode fazer.

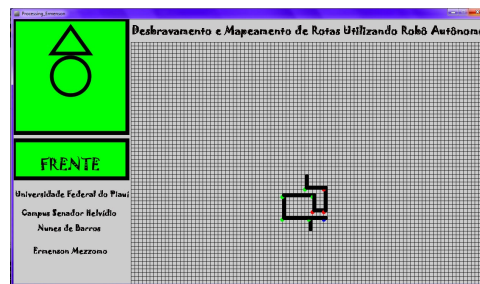


Figura 36 – Tela Referente a um Mapa Percorrido

Fonte: o Autor

A figura 36, mostra um gráfico onde inicialmente o protótipo se deslocou 20 cm para frente percebeu que existia um obstáculo a sua frente mas a sua direita e esquerda estavam livres, de modo aleatório escolheu a direita, deslocou por mais 20 cm até observar um obstáculo a sua frente, percebeu que também existia obstáculos na sua direita e esquerda, onde a única solução era girar 180º no próprio eixo, percorreu mais 55 cm encontrando mais obstáculos à sua frente e esquerda girou 90º à direita e percorreu mais 30 cm percebeu que tinha obstáculo à frente e esquerda girou 90º à direita andou mais 40 cm girou 90º à direita percorreu mais 20 cm girou 90º a esquerda andou mais 15 cm girou novamente à esquerda, percorreu mais 30 cm girou a esquerda se deslocou por mais 25 cm girou à direita percorreu 15 cm onde concluí-o o percurso. O usuário ao perceber que concluí-o o percurso por completo, interrompe a execução desligando o protótipo.

5 Resultados

5.1 As Possíveis Ações Realizadas Pelo Robô

As possíveis ações realizadas pelo robô nos testes para o desvio de obstáculos podem ser compreendidas com base nas ilustrações apresentadas a seguir.

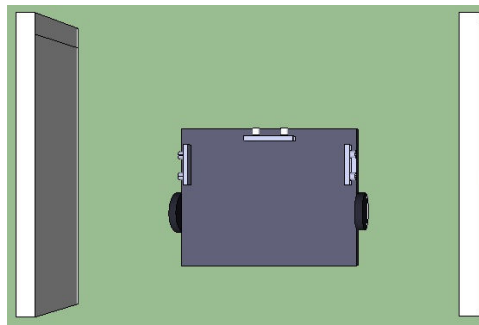


Figura 37 – Quando Não Possui Obstáculo à Frente do Robô

Fonte: o Autor

Ao iniciar os testes é feita uma verificação do sensor da frente, se ele estiver livre, como na figura 37, o protótipo se desloca para frente até encontrar um obstáculo.

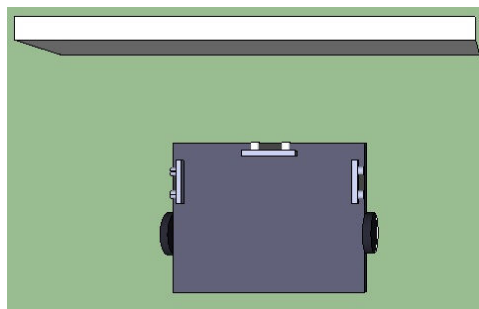


Figura 38 – Obstáculo à Frente do Robô

Fonte: o Autor

No caso apresentado na, figura 38, o sensor da frente irá detectar a presença de um obstáculo quando estiver, a uma distância inferior a 20 cm, em que ele fará a verificação dos outros dois sensores. Neste caso, estarão livres, podendo então girar para qualquer um dos lados, escolhido aleatoriamente.

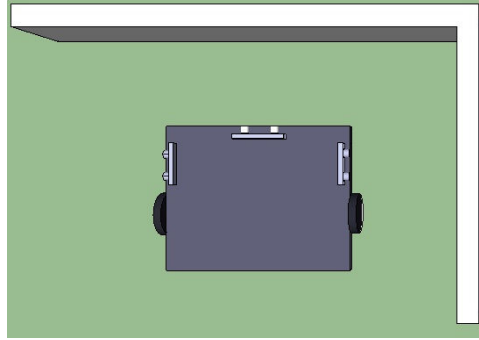


Figura 39 – *Obstáculo à Frente e na Direita do Robô*

Fonte: o Autor

Se em algum local o protótipo se deparar com obstáculo à frente e a direita, como na figura 39, o protótipo gira 90° no próprio eixo para a esquerda, livrando-se dos obstáculos.

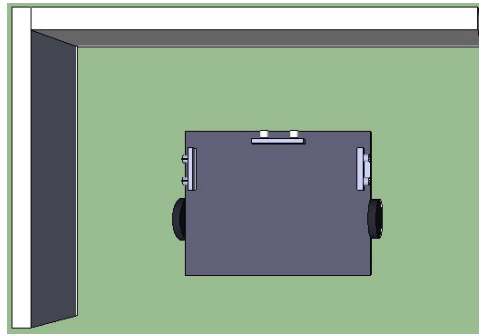


Figura 40 – *Obstáculo à Frente e na Esquerda do Robô*

Fonte: o Autor

Se da mesma forma se depara com obstáculos à frente e esquerda conforme a figura 40, o protótipo gira 90° no próprio eixo à direita, livrando-se dos obstáculos.

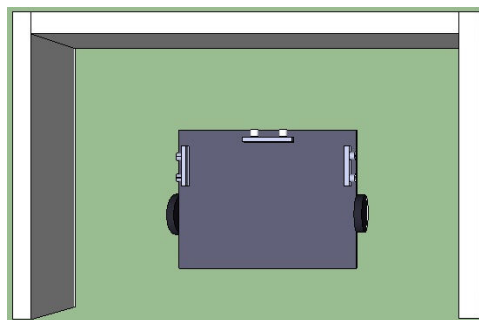


Figura 41 – *Obstáculo à Frente e na Direita e Esquerda do Robô*

Fonte: o Autor

E se em algum caso o protótipo se deparar com obstáculo à frente na sua direita e esquerda, como na figura 41, o protótipo gira 180° no próprio eixo.

5.2 Testes Feitos

Os primeiros testes foram feitos de forma separada de acordo com o andamento do projeto. Inicialmente foi testado o controle de um único motor de passo, o segundo teste foi controlar os dois motores de passo de forma independente.

Após o funcionamento correto dos motores, foram iniciados os testes com os sensores, onde, a exemplo dos testes com os motores, inicialmente foi testado um único sensor. Assim, ao chegar em uma distância programada de 20 cm, um *Led* acenderia, em seguida foram adicionados os outros dois sensores com a mesma finalidade mas controlando *Leds* distintos.

Usando os módulos de Radio Frequência (módulos que fazem a transmissão dos dados) foram feitos alguns testes de transmissão de dados em uma área aberta onde o alcance chegou a mais de 200 m de distância.

Neste momento do projeto, foram iniciados os testes com as partes unidas. Primeiramente foram testados juntos o sistema de tração do veículo (motores de passo) e o sistema de sensoriamento (o conjunto de 3 sonares que detectam distâncias). Em seguida foi adicionado ao veículo o sistema de comunicação (responsável pela transmissão dos dados ao computador). Então uma nova bateria de testes foi iniciada.

Com as partes funcionando juntas e de forma correta, iniciou-se a construção da base (antena que recebe os dados oriundos do meio). A base passou por uma bateria de testes para constatar o seu funcionamento. Então foram realizados mais dez testes diferentes para comprovar o real funcionamento do protótipo, da base e dos *sketchs* produzidos.

Todos os dez testes feitos a partir de agora foram feitos em um ambiente com o tamanho de 2 metros de comprimento por 1,60 de largura.

O primeiro teste foi feito no labirinto mostrado na figura 42. O protótipo percorreu todo o percurso no tempo de 03:00 (Três minutos), com o resultado do mapa feito mostrado na figura 43.

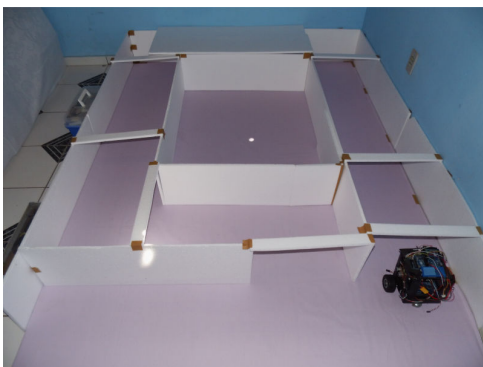


Figura 42 – Labirinto 01

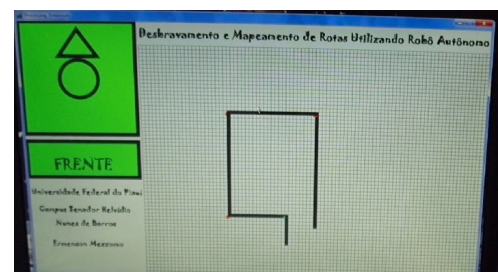


Figura 43 – Resultado do Percorso do Labirinto 01

O segundo teste foi feito no labirinto mostrado na figura 44, com o tempo de duração de 04:06 (quatro minutos e seis segundos), tendo como resultado do mapa percorrido mostrado

na figura 45.



Figura 44 – Labirinto 02

O terceiro teste foi feito no labirinto mostrado na figura 46, com o tempo de duração de 04:33 (quatro minutos e trinta e três segundos), tendo como resultado do mapa percorrido mostrado na figura 47.

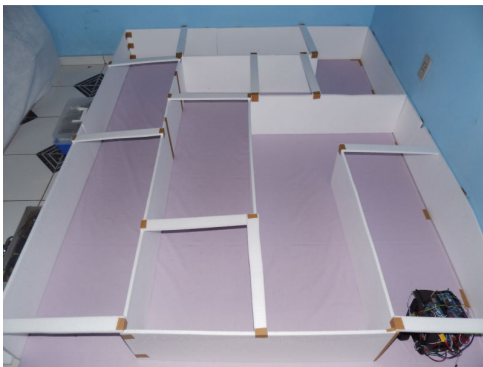


Figura 46 – Labirinto 03

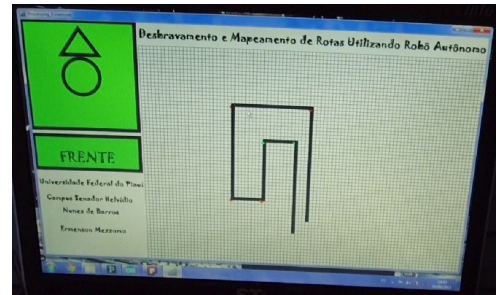


Figura 45 – Resultado do Percurso do Labirinto 02

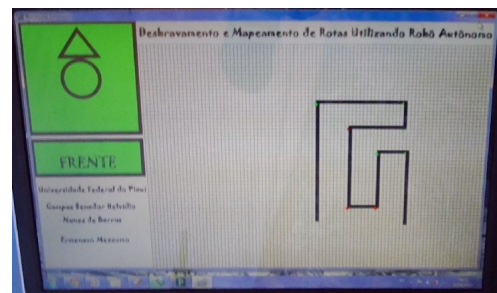


Figura 47 – Resultado do Percurso do Labirinto 03

O quarto teste foi feito no labirinto mostrado na figura 48, com o tempo de duração de 02:55 (dois minutos e cinquenta e cinco segundos), tendo como resultado do mapa percorrido mostrado na figura 49.

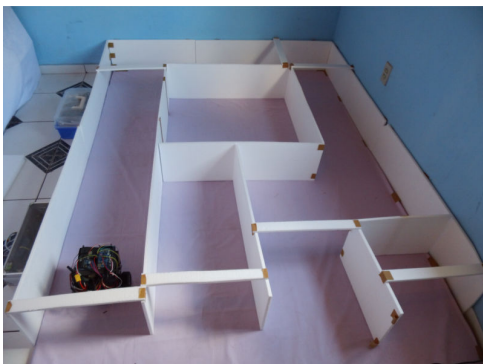


Figura 48 – Labirinto 04

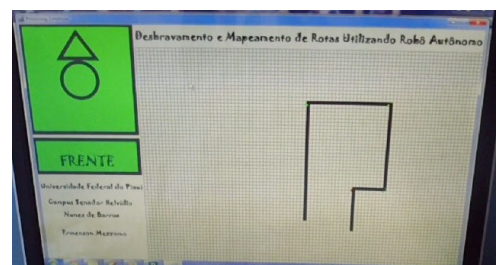


Figura 49 – Resultado do Percurso do Labirinto 04

O quinto teste foi feito no labirinto mostrado na figura 50, com o tempo de duração de 04:38 (quatro minutos e trinta e oito segundos), tendo como resultado do mapa percorrido mostrado na figura 51.



Figura 50 – Labirinto 05

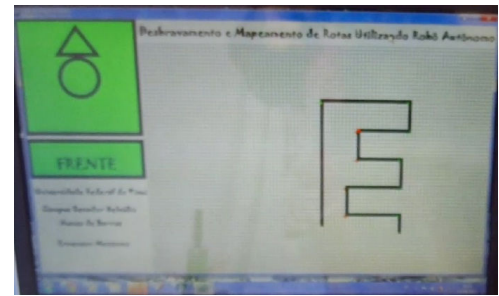


Figura 51 – Resultado do Percurso do Labirinto 05

O sexto teste foi feito no labirinto mostrado na figura 52, com o tempo de duração de 02:36 (dois minutos e trinta e seis segundos), tendo como resultado do mapa percorrido mostrado na figura 53.



Figura 52 – Labirinto 06

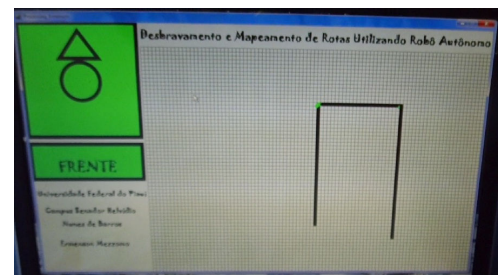


Figura 53 – Resultado do Percurso do Labirinto 06

O sétimo teste foi feito no labirinto mostrado na figura 54, com o tempo de duração de 01:53 (um minuto e cinquenta e três segundos), tendo como resultado do mapa percorrido mostrado na figura 55.



Figura 54 – Labirinto 07

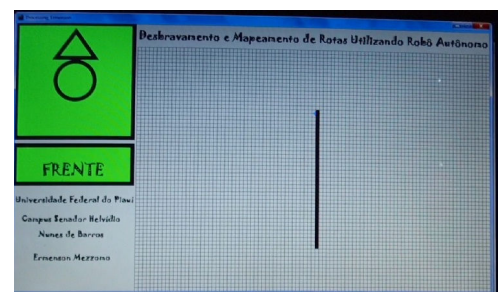


Figura 55 – Resultado do Percurso do Labirinto 07

O oitavo teste foi feito no labirinto mostrado na figura 56, com o tempo de duração de 02:36 (dois minutos e trinta e seis segundos), tendo como resultado do mapa percorrido mostrado na figura 57.

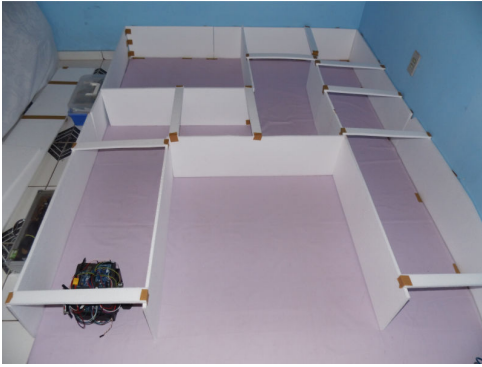


Figura 56 – Labirinto 08

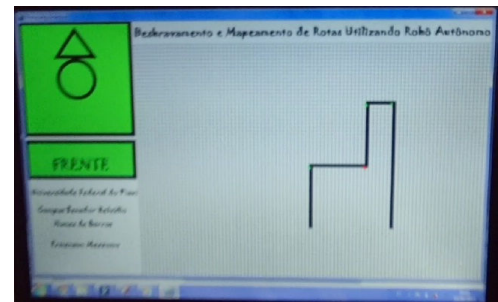


Figura 57 – Resultado do Percurso do Labirinto 08

O nono teste foi feito no labirinto mostrado na figura 58, e tem como resultado final duas saídas diferentes, a primeira saída teve um tempo de duração de 01:39 (um minuto e trinta e nove segundos), tendo como resultado do mapa percorrido mostrado na figura 59.



Figura 58 – Labirinto 09

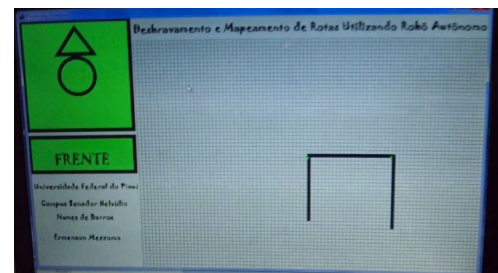


Figura 59 – Resultado do Percurso do Labirinto 09

O segundo mapa só foi conquistado após quatro testes consecutivos, pois em um momento dele é utilizado uma função que é feito a escolha aleatória de qual lado seguir. A segunda saída teve um tempo de duração de 02:44 (dois minutos e quarenta e quatro segundos), tendo como resultado do mapa percorrido mostrado na figura 60.

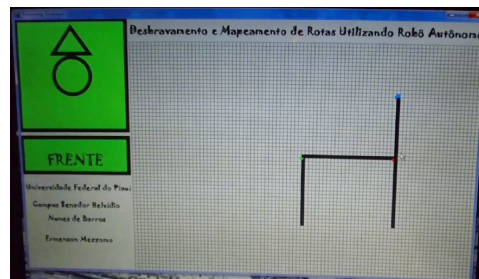


Figura 60 – Segundo Resultado do Percurso do Labirinto 09

Fonte: o Autor

O decimo teste foi feito no labirinto mostrado na figura 61, com o tempo de duração de 02:55 (dois minutos e cinquenta e cinco segundos), tendo como resultado do mapa percorrido mostrado na figura 62.



Figura 61 – Labirinto 10

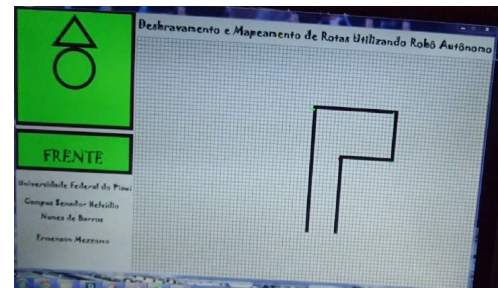


Figura 62 – Resultado do Percurso do Labirinto 10

Tabela 16 – Tabela de Resultados dos Testes

Labirintos	Tempo de Duração	Distância/cm	Volts Feitas
Labirinto 01	03:00	5,15	4
Labirinto 02	04:06	7,45	6
Labirinto 03	04:33	5,20	8
Labirinto 04	02:55	7,95	4
Labirinto 05	04:38	7,80	10
Labirinto 06	01:36	4,85	2
Labirinto 07	01:53	3,65	0
Labirinto 08	01:36	4,65	4
Labirinto 09-1	01:39	3,10	2
Labirinto 09-2	02:44	4,80	3
Labirinto 10	02:55	5,30	4

Nos primeiros testes, o robô mostrou alguns pontos que precisariam ser corrigidos, em relação ao solo em que o robô se deslocava como não era uma superfície plana atrapalhava no deslocamento. Se fosse lisa demais, causava a derrapagem das rodas e conseqüentemente um mapa incorreto.

Foram feitos alguns ajustes onde ele passa a observar também os seus lados, se tiver com uma distância menor que 06 cm de um obstáculo na sua lateral ele faz uma pequena correção de aproximadamente 10° para o outro lado para não se chocar com o obstáculo.

Foi notado que existem pontos cegos onde o robô não consegue perceber a presença de obstáculos, esses pontos localizam-se entre o ângulo do sensor da frente até os posicionamentos dos sensores laterais, podendo ser resolvido adicionando mais dois sensores um em cada lado preenchendo esse ponto cego.

6 Considerações Finais

Neste trabalho foram abordados temas relevantes que impulsionaram o crescimento da robótica nos últimos anos, como deslocamento e explorações em ambientes desconhecidos, mapeamento de trajetória, com pouca ou nenhuma intervenção humana, ou seja, de forma autônoma.

A realização desse trabalho se deu através de pesquisas envolvendo leituras específicas na área de deslocamento, mapeamento, desbravamento além de conhecimento na área eletrônica e de programação.

Ficou constatado que o tipo de solo pode influenciar no desempenho do sistema, sendo recomendável a mudança do sistema de tração de acordo com o terreno a ser percorrido.

O método empregado no robô apresentou resultados esperados e com baixa complexidade para a manutenção do sistema. O robô mostrou-se capaz de realizar a tarefa de desbravar um ambiente desconhecido, mapeando a sua trajetória dentro do ambiente em que está inserido.

Como trabalho futuro propõe-se:

- Melhoramento do sistema autônomo;
- Adicionar mais dois sensores nos pontos cegos;
- Implementar duas formas de desbravamento, uma de forma autônoma e a outra de forma, controlada pelo usuário;

Referências

- ARDUINO, Disponível em: <<http://arduino.cc/>>. Acesso em: 29 mar. 2013.
- AUGUSTO, Roberto Paschoal. *Robô para Mapeamento de Ambientes Estruturados*. Monografia apresentada como conclusão do curso de Engenharia da Computação da Universidade Positivo. Curitiba, 2009.
- AYRES, Marcelo. Robotica: *Conheça a história dos robôs*. Publicado na uol tecnologia, 2007. Disponível em: <<<http://tecnologia.uol.com.br/ultnot/2007/10/01/ult4213u150.jhtm>>>. Acesso em: 18 maio. 2013.
- BANZI, Massimo. *Primeiros passos com o Arduino*. Novatec Editora, 2012.
- CRISTIANO, *Robótica Educacional no Estado de Goiás*: Robótica. 2010. Disponível em: <<http://www.inf.ufrgs.br/roboteka/roboteka/index.php?option=com_content&view=article&id=7:robotica&catid=21:robotica>>. Acesso em: 16 abr. 2013.
- ELETRÔNICA DIDÁTICA, *Eletrônica Didática: Protoboard*. 2010. Disponível em: <<<http://www.eletronicadidatica.com.br/equipamentos/protoboard/protoboard.htm>>>. Acesso em: 5 abr. 2013.
- FOX, D., Burgard, W., and Thrun, S. (1999A). “*Markov localization for mobile robots in dynamic environments*”. *Journal of Artificial Intelligence Research*, vol.11, pp.391-427.
- GONCALVES, Felipe Brites; PUGA, Vinicius de Almeida Santos. *Motor de Passo*, Universidade Federal Fluminense, Curso de Engenharia de Telecomunicações, Niterói - RJ, 2008.
- IVOTI, Instituto de Educação. *Projeto Robótica Educacional: O que é robótica educacional*. 2012. Disponível em: <<<http://www.iei.org.br/curriculares/robotica/educacional/oqueeroboticaeducacional.php>>>. Acesso em: 13 mar. 2013.
- MARTINS, Elaine. *Isaac Asimov: o pai dos robôs*: Conheça um pouco mais sobre o homem que previu, em 1988, a chegada da interatividade na web. publicado no TecMundo, 2012. Disponível em: <<<http://www.tecmundo.com.br/robotica/21551-isaac-asimov-o-pai-dos-robos.htm>>>. Acesso em: 19 mai. 2013.

OLIVEIRA, Emerson de. *Protótipo de um Robô Rastreador de Objetos*. Monografia apresentada à Universidade Regional de Blumenau no curso de ciências da computação. Blumenau-SC, 2001.

PROCESSING, Disponível em: <<<http://processing.org/>>>. Acesso em: 29 mar. 2013.

ROBÓTICA. *História da Robótica*. Disponível em: <<<http://roboticagrupo4.blogspot.com.br/2009/05/historia-da-robotica.html>>>. Acesso em: 23 abr. 2013.

RUSSELL, Stuart J. *Inteligência artificial: tradução da segunda edição*. Stuart Russell, Peter Norvig. Rio de Janeiro: Elsevier, 2004 – 5ª reimpressão.

TATO, Disponível em: << <http://www.tato.ind.br/files/TXMRXM.pdf>>>. Acesso em: 29 mar. 2013.

THRUN, S.; Burgard, W.; Fox, D. (2005). *Probabilistic Robotics*. Cambridge: The MIT Press. 667p.

XAVIER, Alcides Benicasa. *Navegação autônoma de robôs baseada em técnicas de mapeamento e aprendizagem de máquina*. Artigo divulgado na Revista Brasileira de Computação Aplicada (ISSN 2176-6649), Passo Fundo, v. 4, n. 1, p. 102-111, 2012.

YUKINOBU, Alberto Hata. *Mapeamento de ambientes externos utilizando robôs móveis*. Dissertação de mestrado apresentado ao Instituto de Ciência Matemáticas e de Computação – ICMC-USP. São Carlos-SP, 2010.

ZANELATTO, Marcelo Stuari. *Robótica Educacional nos cursos de ciência da computação*. Monografia apresentada ao Curso de Ciência da Computação da Universidade Estadual de Londrina, como requisito parcial à obtenção do título de Bacharel. Londrina-PR, 2004.

ANEXO A – Sketch do Arduino Responsável por Controlar os Motores de Passo

```
#include <Stepper.h>    // Biblioteca dos Motores de Passos

const int stepsPerRevolution = 64;
Stepper small_stepper(stepsPerRevolution, 2,4,3,5);
Stepper small_stepper2(stepsPerRevolution, 8,10,9,11);

// Variaveis de movimento
int s = 0;
int d = 0;

int cs1 = 28;
int cs2 = 24;
int cs3 = 32;

int val1 = 0;
int val2 = 0;
int val3 = 0;

int randNumver;

int quantCOR1, quantCOR2, cor1, cor2;
int contadorCOR1 = 0, contadorCOR2 = 0;

void setup() {
  Serial.begin(9600);
  small_stepper.setSpeed(250); // V do motor de passo da D
  small_stepper2.setSpeed(250); //V do motor de passo da E
```

```

small_stepper.step(s);
small_stepper2.step(d);

//Recebe dados do Arduino que esta os sensores
pinMode(24,INPUT); //Referente ao sensor 02 da Direita
pinMode(28,INPUT); //Referente ao sensor 01 da Frente
pinMode(32,INPUT); //Referente ao sensor 03 da Esquerda

pinMode(25,INPUT);//Referente ao sensor da Esquerda coreção
pinMode(29,INPUT);//Referente ao sensor da Direita coreção

//Envia dados pro Arduino que esta a antena RF
pinMode(37,OUTPUT); // Referente anda 10cm Frente
pinMode(39,OUTPUT); // Referente gira 90° a Direita
pinMode(41,OUTPUT); // Referente gira 90° a Esquerda
pinMode(43,OUTPUT); // Referente gira 180° Graus
}

void loop(){
  val1 = digitalRead(cs1); // ler a entrada pino
  val2 = digitalRead(cs2); // ler a entrada pino
  val3 = digitalRead(cs3); // ler a entrada pino

  cor1 = digitalRead(25); // ler a entrada pino
  cor2 = digitalRead(29); // ler a entrada pino

  if(val1 == 0 && val2 == 0 && val3 == 0){// sensores = 0
    Serial.println("Gira 180° no mesmo eixo");
    s=-1;
    d=1;

    int x=0;
    while(x<2600){
      small_stepper.step(s);
      small_stepper2.step(d);
      x++;
    }
  }
}

```

```

}

//digitalWrite(43,LOW);
digitalWrite(43,HIGH);// ALTO LIGADO
    delay(200);
digitalWrite(43,LOW);//BAIXO DESLIGADO

} else{

if(val1 == 0 && val2 == 0){// sensor da direita
    Serial.println("Girando 90° pra Esquerda");
    s=1;
    d=-1;

    int x=0;
    while(x<1300){
        small_stepper.step(s);
        small_stepper2.step(d);
        x++;
    }

    digitalWrite(41,HIGH);// ALTO LIGADO
        delay(200);
    digitalWrite(41,LOW);//BAIXO DESLIGADO
}

if(val1 == 0 && val3 == 0){// sensor da esquerda
    Serial.println("Girando 90° pra Direita");
    s=-1;
    d=1;

    int y=0;
    while(y<1300){
        small_stepper.step(s);
        small_stepper2.step(d);
        y++;
    }
}

```

```

    }

    digitalWrite(39,HIGH); // ALTO LIGADO
        delay(200);
    digitalWrite(39,LOW); //BAIXO DESLIGADO
}
}

if(vall == 1){ // sensor da frente
    Serial.println("Andando pra Frente");
    s=1;
    d=1;
    int i=0;
    digitalWrite(37,HIGH); // ALTO LIGADO

    while(i<500){
        small_stepper.step(s);
        small_stepper2.step(d);
        i++;

        if(i==70){
            digitalWrite(37,LOW); //BAIXO DESLIGADO
        }
    }

    if (contadorCOR1 == 1 ){
        quantCOR1 ++; }

    if (quantCOR1 == 2){
        quantCOR1 = 0;
        contadorCOR1 = 0;}

    if (cor1 == 0 && contadorCOR1 == 0){ // Gira a D
        s=-1;
        d=1;
        int i=0;

```

```

        while(i<144){    // gira 10 graus
            small_stepper.step(s);
            small_stepper2.step(d);
            i++;
            contadorCOR1 = 1;
        }
    }// fim cor 01

if (contadorCOR2 == 1 ){
    quantCOR2 ++; }

if (quantCOR2 == 2){
    quantCOR2 = 0;
    contadorCOR2 = 0;}

if (cor2 == 0 && contadorCOR2 == 0) { Gira a E
    s=1;
    d=-1;
    int i=0;
    while(i<144){    // gira 10 graus
        small_stepper.step(s);
        small_stepper2.step(d);
        i++;
        contadorCOR2 = 1;
    }
} // fim cor 02
} // fim frente

if(val1 == 0 && val2 == 1 && val3 == 1){
    randNumver = random (2); //escolhe um valor aleatorio 0-1
    if(randNumver==0){
        s=1;
        d=-1;

        int i=0;
        while(i<1300){

```

```
        small_stepper.step(s);
        small_stepper2.step(d);
        i++;
    }

    digitalWrite(41,HIGH); // ALTO LIGADO
        delay(200);
    digitalWrite(41,LOW); //BAIXO DESLIGADO
}

if(randNumver==1){
    s=-1;
    d=1;
    int i=0;
    while(i<1300){
        small_stepper.step(s);
        small_stepper2.step(d);
        i++;
    }

    digitalWrite(39,HIGH); // ALTO LIGADO
        delay(200);
    digitalWrite(39,LOW); //BAIXO DESLIGADO
}
}
}
```

ANEXO B – Sketch do Arduino Responsável pelo Controle dos Sensores

```
// programa para o sensor ultrassonico
int ledPin = 13;
//sensor 1 da frente
int trig1 = 5;
int echo1 = 6;
//sensor 2 da direita
int trig2 = 2;
int echo2 = 3;
//sensor 3 esquerda
int trig3 = 9;
int echo3 = 10;

void setup( ){
  Serial.begin(9600);
  //sensor 1 da frente
  pinMode(trig1, OUTPUT);
  pinMode(echo1, INPUT);
  //sensor 2 da direita
  pinMode(trig2, OUTPUT);
  pinMode(echo2, INPUT);
  //sensor 3 da esquerda
  pinMode(trig3, OUTPUT);
  pinMode(echo3, INPUT);

  //Envia dados pro Arduino que esta os motores de passos
  pinMode(4,OUTPUT); // Referente ao Sensor 1 da frente
  pinMode(8,OUTPUT); // Referente ao Sensor 2 da direita
```



```
pinMode(11,OUTPUT); // Referente ao Sensor 3 da esquerda

pinMode(12,OUTPUT);
pinMode(13,OUTPUT);
}

void loop(){
  float tempo1, cm1;//Sensor 1 da frente
  float tempo2, cm2;//Sensor 2 da direita
  float tempo3, cm3;//Sensor 3 da esquerda

  digitalWrite(trig1, LOW);
  delayMicroseconds(2);
  digitalWrite(trig1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig1, LOW);
  delayMicroseconds(2);
  tempo1 = pulseIn(echo1, HIGH);
  cm1 = tempo1/29/2; //converte duration em centímetros

  digitalWrite(trig2, LOW);
  delayMicroseconds(2);
  digitalWrite(trig2, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig2, LOW);
  delayMicroseconds(2);
  tempo2 = pulseIn(echo2, HIGH);
  cm2 = tempo2/29/2; //converte duration em centímetros

  digitalWrite(trig3, LOW);
  delayMicroseconds(2);
  digitalWrite(trig3, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig3, LOW);
  delayMicroseconds(2);
```

```
tempo3 = pulseIn(echo3, HIGH);
cm3 = tempo3/29/2; //converte duration em centímetros

    delay(100);

//sensor 1 da frente
if(cm1<20){
    digitalWrite(8,LOW);
} else{
    digitalWrite(8,HIGH);
}

if(cm2<30){
    digitalWrite(4,LOW);
} else{
    digitalWrite(4,HIGH);
}

if(cm3<30){
    digitalWrite(11,LOW);
} else{
    digitalWrite(11,HIGH);
}

if(cm2<6){ //direita
    digitalWrite(12,LOW);
} else{
    digitalWrite(12,HIGH);
}

if(cm3<6){ // esquerda
    digitalWrite(13,LOW);
} else{
    digitalWrite(13,HIGH);
}
}
```

ANEXO C – Sketch do Arduino Responsável por Enviar os Dados

Uno Transmissor

```
#include <VirtualWire.h> // biblioteca de RF

int cs2 = 2;
int cs3 = 3;
int cs4 = 4;
int cs5 = 5;

int val2 = 0;
int val3 = 0;
int val4 = 0;
int val5 = 0;

void setup() {
  Serial.begin(9600);

  //Recebe dados do Arduino que esta os motores de passos
  pinMode(2, INPUT); // Referente anda 10cm Frente
  pinMode(3, INPUT); // Referente gira 90° a Direita
  pinMode(4, INPUT); // Referente gira 90° a Esquerda
  pinMode(5, INPUT); // Referente gira 180° Graus

  vw_set_ptt_inverted(true); // Required for DR3100
  vw_setup(2000); // Bits por segundo
}
```

```
void loop(){

    val2 = digitalRead(cs2);    // ler a entrada pino 2
    val3 = digitalRead(cs3);    // ler a entrada pino 3
    val4 = digitalRead(cs4);    // ler a entrada pino 4
    val5 = digitalRead(cs5);    // ler a entrada pino 5

    if(val2 == 1){// Referente anda 10cm Frente
        const char *msg = "2";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    } else{
    }

    if(val3 == 1){// Referente gira 90° a Direita
        const char *msg = "1";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    } else{
    }

    if(val4 == 1){//Referente gira 90° a Esquerda
        const char *msg = "5";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();
    } else{
    }

    if(val5 == 1){// Referente gira 180° Graus
        const char *msg = "4";
        vw_send((uint8_t *)msg, strlen(msg));
        vw_wait_tx();

    } else{
    }
}
```

ANEXO D – Sketch do Arduino Responsável por Receber os Dados

```
#include <VirtualWire.h>

char info[30];
void setup(){

Serial.begin(9600);
vw_set_ptt_inverted(true);
vw_setup(2000);
vw_rx_start();
pinMode(13, OUTPUT); }

void loop(){
digitalWrite(13, LOW);
uint8_t buf[VW_MAX_MESSAGE_LEN];
uint8_t buflen = VW_MAX_MESSAGE_LEN;
if (vw_get_message(buf, &buflen)) {

digitalWrite(13, true);
info[0]=buf[0];
Serial.println(String(info));
memset( &info, 0, sizeof(info) );
digitalWrite(13, false);
}
}
```

ANEXO E – Sketch do Processing Responsável por Transformar os Dados Recebidos em Gráficos

```
import processing.serial.*;
Serial port;

int xI=785;
int yI=555;
int xF=785;
int yF=555;

void setup() {
  size(1250, 700);

  for (int i = 310; i < width; i += 10) { //Vertical
    stroke(0);
    line(i, 60, i, height);
  }
  for (int i = 60; i < height; i += 10) { //Horizontal
    stroke(0);
    line(310, i, width, i);
  }
  port = new Serial(this, "COM8", 9600);
}

int FRENTE = 1;
int DIREITA = 2;
int RE = 3;
int ESQUERDA = 4;
```

```

int F = FRENTE;

void draw() {
    PFont font;
    font = loadFont("Jokerman-Regular-48.vlw");
    textFont(font, 28);
    textAlign(CENTER);
    fill (#050000);
    text("Desbravamento e Mapeamento de Rotas Utilizando
Robô Autônomo", 780, 40);
    font = loadFont("Jokerman-Regular-48.vlw");
    textFont(font, 20);
    textAlign(CENTER);
    fill (#030000);
    text("Universidade Federal do Piauí", 155, 470);
    text("Campus Senador Helvídio ", 150, 520);
    text("Nunes de Barros ", 150, 560);
    textFont(font, 20);
    text("Ermenson Mezzomo", 150, 620);

    stroke(1); // cor da linha
    strokeWeight(10); //largura da linha
    smooth(); // tipo da linha

    while (port.available ()>0) {
        int a=port.read();

        if ( a =='2') { // frente
            if (F == DIREITA) { //Se frente e igual a Direita
                fill(0, 255, 0);
                rect(0, 0, 300, 300);
                ellipse(150, 150, 100, 100);
                triangle(200, 90, 150, 20, 100, 90); //Frente

                font = loadFont("Jokerman-Regular-48.vlw");
                textFont(font, 40);
            }
        }
    }
}

```

```

textAlign(CENTER);
fill (0, 255, 0);
rect(0, 320, 300, 100);
fill (#030000);
text("FRENTE", 150, 400);

xF += 10;
line(xI, yI, xF, yF);
xI=xF;
}
if (F == ESQUERDA) { // Se frente e igual a Esquerda

fill(0, 255, 0);
rect(0, 0, 300, 300);
ellipse(150, 150, 100, 100);
triangle(200, 90, 150, 20, 100, 90); //Frente

font = loadFont("Jokerman-Regular-48.vlw");
textFont(font, 40);
textAlign(CENTER);
fill (0, 255, 0);
rect(0, 320, 300, 100);
fill (#030000);
text("FRENTE", 150, 400);

xF -= 10;
line(xI, yI, xF, yF);
xI=xF;
}
if (F == RE) { // se frente e igual a Re
fill(0, 255, 0);
rect(0, 0, 300, 300);
ellipse(150, 150, 100, 100);
triangle(200, 90, 150, 20, 100, 90); //Frente

font = loadFont("Jokerman-Regular-48.vlw");

```



```

    textFont(font, 40);
    textAlign(CENTER);
    fill (0, 255, 0);
    rect(0, 320, 300, 100);
    fill (#030000);
    text("FRENTE", 150, 400);

    yF += 10;
    line(xI, yI, xF, yF);
    yI=yF;
}
if (F == FRENTE) { // se Frente e igual a Frente

    font = loadFont("Jokerman-Regular-48.vlw");
    textFont(font, 40);
    textAlign(CENTER);
    fill (0, 255, 0);
    rect(0, 320, 300, 100);
    fill (#030000);
    text("FRENTE", 150, 400);

    fill(0, 255, 0);
    rect(0, 0, 300, 300);
    ellipse(150, 150, 100, 100);
    triangle(200, 90, 150, 20, 100, 90); //Frente

    yF -= 10;
    line(xI, yI, xF, yF);
    yI=yF;
}
}
if (a == '5') { //esquerda
    //F = ESQUERDA;

    fill(0, 255, 0);
    rect(0, 0, 300, 300);

```

```

ellipse(150, 150, 100, 100);
triangle(90, 100, 20, 140, 90, 200); //Direita

font = loadFont("Jokerman-Regular-48.vlw");
textFont(font, 40);
textAlign(CENTER);
fill (0, 255, 0);
rect(0, 320, 300, 100);
fill (#030000);
text("GIROU 90°", 150, 360);
text("A ESQUERDA", 150, 410);

if (F == DIREITA) F = FRENTE;
else if (F == RE) F = DIREITA;
else if (F == ESQUERDA) F = RE;
else if (F == FRENTE) F = ESQUERDA;

stroke(255, 0, 0); // cor da linha
line(xI, yI, xF-5, yF+5);
}    if (a == '1') { //direita
//F = DIREITA;

fill(0, 255, 0);
rect(0, 0, 300, 300);
ellipse(150, 150, 100, 100);
triangle(210, 100, 280, 140, 210, 200); //Esquerda

font = loadFont("Jokerman-Regular-48.vlw");
textFont(font, 38);
textAlign(CENTER);
fill (0, 255, 0);
rect(0, 320, 300, 100);
fill (#030000);
text("GIROU 90°", 150, 360);
text("A DIREITA", 150, 410);

```

```

    if (F == DIREITA) F = RE;
    else if (F == RE) F = ESQUERDA;
    else if (F == ESQUERDA) F = FRENTE;
    else if (F == FRENTE) F = DIREITA;

    stroke(0, 255, 0); // cor da linha
    line(xI, yI, xF-5, yF+5);
}    if (a == '4') { // F = RE;

    fill(0, 255, 0);
    rect(0, 0, 300, 300);
    ellipse(150, 150, 100, 100);
    triangle(200, 210, 150, 280, 100, 210); //Re

    font = loadFont("Jokerman-Regular-48.vlw");
    textFont(font, 34);
    textAlign(CENTER);
    fill (0, 255, 0);
    rect(0, 320, 300, 100);
    fill (#030000);
    text("GIROU 180° NO", 150, 360);
    text("PROPIO EIXO", 150, 400);

    if (F == DIREITA) F = ESQUERDA;
    else if (F == RE) F = FRENTE;
    else if (F == ESQUERDA) F = DIREITA;
    else if (F == FRENTE) F = RE;

    stroke(0, 0, 255); // cor da linha
    line(xI, yI, xF-5, yF+5);
}    else {
    line(xI, yI, xF, yF);    }
}
}

```