Trabalho de Conclusão de Curso

CARRO ROBÔ AUTO-RECARREGÁVEL ATRAVÉS DE CÉLULAS FOTOVOLTAICAS E BUSCA DE LOCAIS COM LUMINOSIDADE.

Wylken dos Santos Machado
wylken.ufal@gmail.com

Orientador:

Dr. André Luiz Lins de Aquino

Maceió, Março de 2015

Wylken dos Santos Machado

CARRO ROBÔ AUTO-RECARREGÁVEL ATRAVÉS DE CÉLULAS FOTOVOLTAICAS E BUSCA DE LOCAIS COM LUMINOSIDADE.

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação do Instituto de Computação da Universidade Federal de Alagoas.

Orientador:

Dr. André Luiz Lins de Aquino

Maceió, Março de 2015

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação do Instituto de Computação da Universidade Federal de Alagoas, aprovada pela comissão examinadora que abaixo assina.

Dr. André Luiz Lins de Aquino - Orientador
Instituto de Computação
Universidade Federal de Alagoas

Dr. Leonardo Viana Pereira - Examinador
Instituto de Computação
Universidade Federal de Alagoas

M.ª Evellyn Soares Cavalcante - Examinadora
Instituto de Computação
Universidade Federal de Alagoas

Maceió, Março de 2015

Resumo

Este trabalho apresenta implementação de um robô com o objetivo de encontrar áreas com incidência de raios solares, para recarregar suas baterias através de uma célula fotovoltaica. Na implementação foi utilizado o *Arduino UNO* em combinação com sensores de luminosidade, ultrassom e motores DC, além de outros materiais. Para alcançar o objetivo proposto, foram elaborados dois algoritmos, um para estabelecer a direção com maior luminosidade, e outro para o desvio dos possíveis obstáculos que poderão aparecer ao longo do caminho. Os testes realizados apresentam resultados satisfatórios para os cenários propostos, e possibilita estudos complementares com a finalidade de aprimorar e aperfeiçoar a busca realizada.

Abstract

This paper presents the implementation of a robot in order to find areas with incidence of sunrays, with the purpose of recharge batteries through photovoltaic cell.
In the implementation were used the arduino UNO combined with lightness sensors, ultrasound and DC engines and other materials. To achieve the proposed objective, two algorithms were developed, one to establish the direction with greater luminosity and other to deviate possible obstacles on the way.
The tests results were satisfactory for the proposed scenarios, and allows further studies to improve to search mechanism

Agradecimentos

Agradeço em primeiro lugar a Deus, que me deu saúde, força, coragem e uma família maravilhosa.
A minha mãe Ednalva S. Machado, ao meu pai Cremilson M. da Silva e meu irmão Cleberson S. Machado, pelo apoio, incentivo, amor e carinho ao longo desta longa caminhada que é a vida. Obrigado também pela força para que eu pudesse subir mais esse degrau.
A minha esposa Rafaela S. da Silva, companheira e amiga, por toda a atenção, amor e carinho que a mim foram dados, que de forma especial me deu força e coragem, me apoiando nos momentos de dificuldades.
Ao Professor Dr. André Luiz Lins de Aquino, pela disponibilidade e atenção prestadas. Obrigado também pela ajuda nos momentos difíceis. Também pelos conselhos e inúmeras sugestões que foram importantes para a realização deste trabalho.
E todos aqueles que de alguma forma estiveram e estão próximos de mim, fazendo esta vida valer cada vez mais a pena.

Wylken dos Santos Machado

"A menos que modifiquemos a nossa maneira de pensar, não seremos capazes de resolver os problemas causados pela forma como nos acostumamos a ver o mundo."

Albert Einstein

Conteúdo

Lista de Figuras

Lista de Códigos

Lista de Tabelas

Introdução

Um dos grandes sonhos da humanidade é aproveitar todo o tempo disponível para se divertir e aproveitar a vida, nada de trabalho ou preocupação com obrigações das mais diversas formas, assim, o homem vem buscando a cada dia criar ferramentas que substituam o trabalho humano. A automação; tecnologia que utiliza sistemas mecânicos, elétricos, eletrônicos e computacionais; vêm se desenvolvendo rapidamente com o intuito de cumprir com esse objetivo. A automação visa diminuir o uso de mão de obra, facilitar os processos produtivos, e dessa forma, produzir bens em maior quantidade, menor tempo, menor custo e maior qualidade [21]. Esse trabalho é um exemplo de automação, que pode ser utilizada em diversos setores da tecnologia.

O setor que mais obteve avanço foi o de automação industrial, definido como "Qualquer sistema, apoiado em computadores, que substitui o trabalho humano, em favor da segurança das pessoas, da qualidade dos produtos, rapidez da produção ou da redução de custos, assim aperfeiçoando os complexos objetivos das indústrias, dos serviços ou bem estar" [15].

Hoje em dia, temos utilização de robôs nos mais diversos meios, temos robôs atuando na área médica (o médico manipula o robô), realizando vários tipos de cirurgias; fabricação de carros, de aviões, logística interna das empresas, acessibilidade, atendimento ao público, atividades militares Figura 4 etc.

Na Figura 1 temos o sistema robôtico daVince, que em 1997 realizou sua primeira cirurgia em Bruxelas, por Jacques Himpens e Cardiere [19]. Já a Figura 2 apresenta uma forma de soldagem automática, o qual executa operações de soldagem, após programação, sem ajuste ou controle por parte do operador de solda [14].

O robo TUG, Figura 3 foi criado para atuar na logística de hospitais, ajudando no trabalho de equipes médicas e de enfermagem. Este robo se desloca pelo hospital transportando alimento, amostras de sangue, medicamentos, etc. Fabricado pela empresa Aethon, os robôs ja são utilizados em alguns hospitais nos EUA [10].

De acordo com a Federação Internacional de Robótica, o Japão é o país mais robotizado do mundo, com 360 robôs para cada 10 mil trabalhadores. Segundo esse mesmo Órgão, em 2010 o mercado de robôs industriais movimentou cerca de 5,7 bilhões de dólares em todo o mundo [6].

Na área de sensoriamento remoto, a capacidade da bateria tem um enorme impacto no

Figura 1: Sistema robótico daVinci

Figura 2: Soldagem Robotizada

desenvolvimento das aplicações, muitas vezes reprimindo grande parte do potencial de processamento, a fim de reduzir o consumo de energia, e por consequência, estender o tempo de carga da bateria. Com o intuito de reduzir essa e outras dificuldades com relação à independência energética de sistemas eletrônicos, utilizando conceitos de eletrônica, circuitos digitais e programação fornecidos pelo curso de graduação em Ciências da Computação (UFAL), o presente trabalho descreve a implementação de um robô que possui a necessidade de auto alimentação, tornando-se uma máquina que não depende de intervenção humana

Figura 3: Robô TUG, utilizado para transportar diversos materiais em hospitais

para se manter em funcionamento durante um longo período de tempo. Para tanto serão utilizadas uma bateria recarregável e uma célula fotovoltaica, com a finalidade de transformar luz solar em energia elétrica.

Esse estudo poderá ser utilizado para implementação de outras máquinas que precisem de autonomia energética a fim de realizar suas atividades.

Figura 4: Robô militar LS3, desenvolvido pela empresa Boston Dynamics para auxiliar em operações táticas.transportando suprimento em terrenos acidentados

Capítulo 1

Fundamentação

No presente capítulo iremos apresentar todos os componentes e conceitos utilizados no desenvolvimendo de nosso robô. Será apresentado o esquema e as caracteriśticas dos *hardwares*, os conceitos utilizados na modelagem do *software* responsável pelo controle de todo o sistema.

1.1 Conseitos de Robôtica

A ideia de robôs, ou algum tipo de máquina para ajudar as pessoas é muito antiga, ainda não foi possível fixar o ponto onde se originou [22]. A palavra robô foi popularizada por Karel Capek em 1921, ao longo do tempo, o conceito de robô vem sendo modificado, acompanhando o desenvolvimento da tecnologia. Atualmente inclui máquinas que podem processar pensamento, raciocínio, resolução de problemas e até mesmo emoções e consciência.

Segundo M. J. Mataric [22], um robô é um sistema autônomo que existe no mundo físico, que pode sentir o ambiente e agir sobre ele para atingir algum objetivo, ou seja, ele atua com base em suas próprias decisões, um robô deve responder a estímulos gerados por seus sensores. Uma máquina que não tenha essas características não pode ser considerada um robô, não importa o quão sofisticado seja o sistema de emulação em computador, este não pode ser considerado um robô, pois não está presente no universo físico. O termo robótica se refere a área de ciência e tecnologia que estuda os robôs, essa palavra foi criada por Isaac Asimov, um grande escritor de ficção científica.

Um grande e complexo campo de estudo essencial e que faz parte da robótica é a Inteligência Artificial. Essa área tem como objetivo criar inteligência em máquinas, ou seja, entidades que tenham a capacidade de processar modelos lógicos, memorização, comunicação, aprendizado, planejamento, resolução de problemas etc. O campo de inteligência artificial foi oficializado em 1956, em conferência da Universidade de Dartmouth, em Hanover. Esse encontro reuniu os principais pesquisadores do tema, entre eles: Marvin Minsky, Herbert Simon, John McCarthy e Allan Newell, considerados os fundadores desse campo.

Sensores e Atuadores são essenciais para o desenvolvimento das atividades de um robô, em analogia, podemos comparar sensores aos nossos sentidos e atuadores aos músculos e membros. Sensores são mecanismos desenvolvidos para transformar fenômenos analógicos do meio físico em sinais elétricos, que podem ser processados; alguns exemplos de sensores são: sensor de temperatura, luminosidade, obstáculo, umidade, pressão, eletromagnetismo, toque, movimento, gás, cor, posicionamento, altitude etc [22].

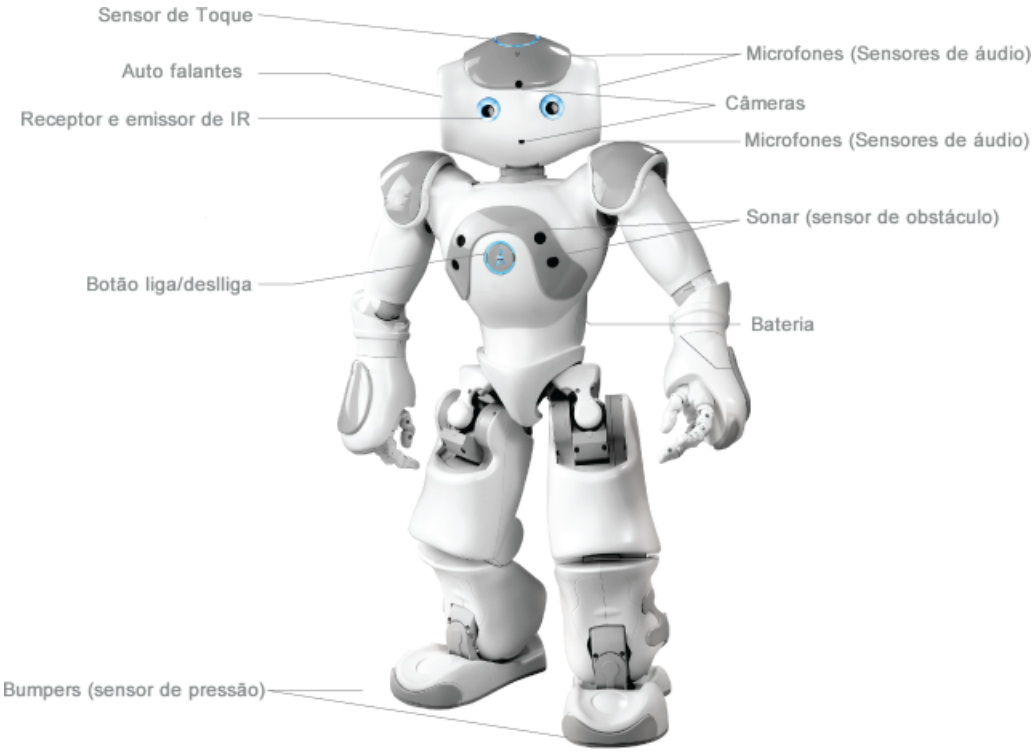


Figura 1.1: Descrição dos sensores utilizados no robô NAO, criado para o ensino e a pesquisa em robótica e inteligência artificial.

Os atuadores transformam energia elétrica em energia mecânica e, dessa forma, são responsáveis pela locomoção e interação do sistema com o mundo real, os grandes representantes dessa classe são os motores, que podem ser: motores DC (item 1 Figura 1.2), motores de passo (item 2 Figura 1.2) e servo motores (item 3 Figura 1.2). Existem dispositivos que não convertem energia elétrica em mecânica, porém, são considerados atuadores, pois permitem interação, são eles: led, lâmpadas, auto falantes e displays [22].

Para que um robô tenha a capacidade de tomar decisões, ou seja, ser autônomo, necessita de um mecanismo que seja capaz de avaliar os acontecimentos do ambiente, e a partir desses dados, escolher a ação mais relevante para que o objetivo seja alcançado; o dispositivo que tem essa responsabilidade é o controlador, Figura 1.3, *hardware* com capacidade de processamento de informações. Temos diversos controladores comercializados no mercado, entre eles:

1. Arduino UNO R3: Arduino Uno (Figura 1.3) é uma placa de microcontrolador baseado no ATmega328 . Ele possui 14 pinos digitais de entrada/saída (dos quais 6 podem ser

Figura 1.2: 1- Motor DC, 2 - Motor de passo, 3 - Servo motor

usados como saídas PWM), 6 entradas analógicas, um ressonador cerâmico 16 MHz, uma conexão USB, entrada p4 para alimentação, um cabeçalho ICSP, e um botão de reinicialização [1] .

2. Vários modelos de microcontroladores da Microchip: Possui microcontroladores para toda a gama de 8 bits, 16 bits e 32 bits, entre esses o 16F628 e 12F675 Figura 1.3 , com uma poderosa arquitetura, tecnologias de memória flexíveis, ferramentas de desenvolvimento abrangente e de fácil de utilização e uma boa documentação técnica [5].

3. 8051 da Intel: Microcontrolador da família MCW-51 com 8bit, 64K de espaço para endereçamento de memória, 4K de memória de programa, 128 bytes de memória RAM, oscilador interno e dois contadores de 16 bits [13].

4. 68HC11 da Motorola: Fabricado pela motorola e colocado no mercado em 1985, possui 512 bytes de EEPROM 2 256 bytes de memoria ram [24].

5. TMS370 da Texas Instruments: Microcontrolador que segundo o próprio fabricante, Texas Instruments, possui várias instruções obsoletas e não é recomendado o uso em novos projetos. Ele possui 8K bytes de ROM, 8K bytes de EPROM, 256 Bytes de Data EEPROM e 256 Bytes de RAM estática usada em seus registradores [9].

6. Raspbery PI, Figura 1.3: É um minicomputador, do tamanho de um cartão de crédito, que possui conexões para mouse, teclado, monitor, cartão SD, áudio. Também possui um cabeçalho GPIO com até 40 pinos e versões com processadores e memória RAM de até 900 MHz e 1GB. Devido ao seu cabeçalho GPIO, pequeno tamanho e consumo e preço acessível está sendo cada vez mais utilizado em projetos embarcados, que necessitam de uma boa capacidade de processamento [7].

;

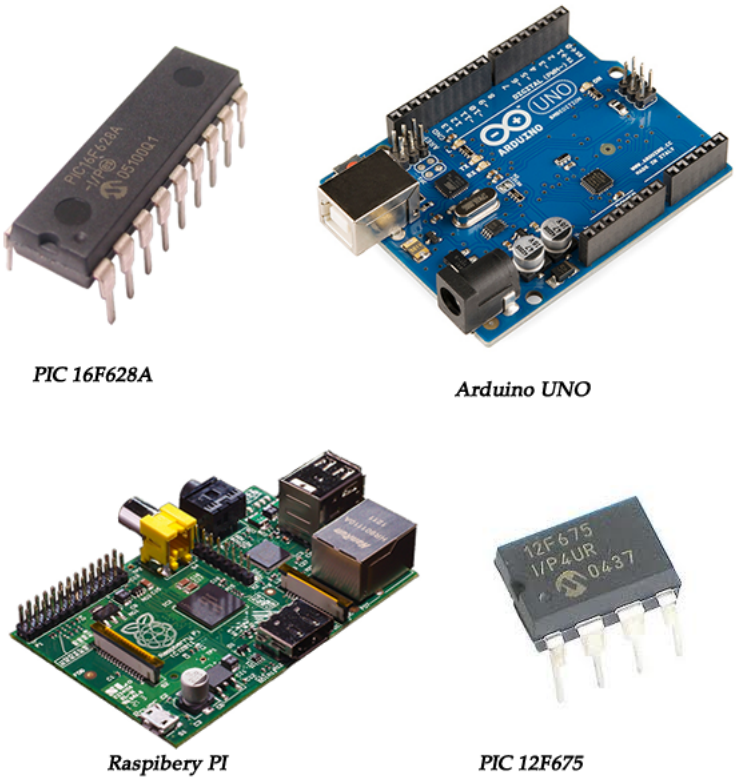


Figura 1.3: Controladores utilizados no desenvolvimento de vários projeto, o Arduino e o Raspberry PI são open source.

Com o avanço de tecnologia o campo da robótica vem obtendo grandes avanços, vários projetos e protótipos, com as mais diversas finalidades, estão sendo desenvolvidos. Hoje temos robôs complexos e sofisticados como o cão robótico Aibo da SONY [8], Figura 1.4, com capacidade de demonstrar estado emocional, possui humor e muda de comportamento de acordo com o ambiente; já o humanoide ASIMO da Honda [2], Figura 1.4, que possui esse nome em homenagem a Isaac Asimov, possui capacidade de andar sobre duas pernas, ele pode andar em superfícies irregulares, pegar objetos, reconhecer pessoas, correr a aproximadamente 9 km/h, pular, se comunicar e escrever.

1.1.1 Computação Física

Segundo Banzi M. (2011) [11] "computação física envolve o projeto de objetos interativos que podem se comunicar com humanos utilizando sensores e atuadores controlados por um comportamento implementado como *software*, executado dentro de um microcontrolador".

Sensores e atuadores são componentes que permitem a um equipamento eletrônico interagir com o mundo. Sensores são caracterizados como qualquer sistema elétrico que transforma dados e acontecimentos do mundo real em sinais elétricos que podem ser processados (LDR, Sensor de temperatura, sensor de movimento etc), já os atuadores transformam sinais elétricos em reações diversas no mundo real (LED, Motores DC etc).

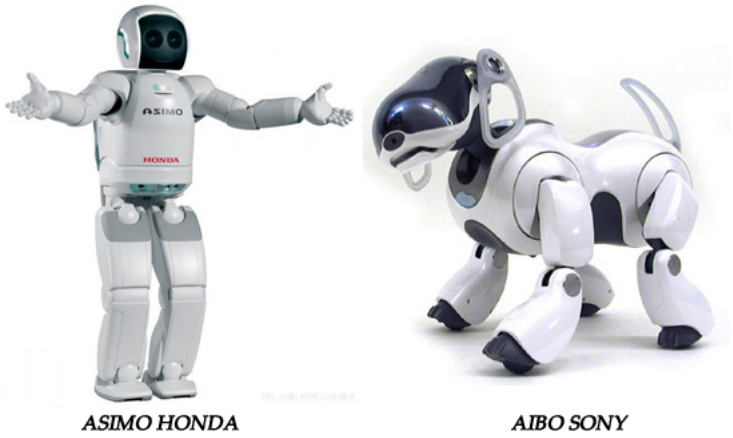


Figura 1.4: Os robores ASIMO e AIBO, estã entre os mais sofisticados desenvolvidos até hoje.

1.2 Arduino

Arduino [1] é uma plataforma para prototipagem eletrônica de *hardware* livre para computação física ou embarcada, ou seja um sistema que pode interagir com seu ambiente por meio de hardware e *software*, projetado com um microcontrolador Atmel [3], empresa fabricante de microcontroladores entre outros componentes eletrônicos, que domina uma grande parte desse mercado, seus componentes são utilizados em diversos dispositivos produzidos. Possui um ambiente de desenvolvimento simples e ampla documentação, seu hardware possui pinos de entrada e saída de dados digitais e analógicos, além de comunicação serial. Sua linguagem de programação se chama *Processing*, essencialmente C/C++, o principal objetivo dessa plataforma é criar ferramentas que são acessíveis, de baixo custo, flexíveis e fáceis de usar por iniciantes em eletrônica, além de fornecer vantagens didáticas à alunos e professores [11].

Segundo McRoberts [23], "a maior vantagem do Arduino sobre outras plataformas de desenvolvimento de microcontroladores é a facilidade de sua utilização; pessoas que não são da área técnica podem, rapidamente, aprender o básico e criar seus próprios projetos em um intervalo de tempo relativamente curto".

O *hardware* e o *software* do Arduino são ambos de fonte aberta, as placas podem ser montadas manualmente, ou compradas pré-montadas, é possível fazer o *download* gratuito do IDE de código aberto em `www.arduino.cc`. O IDE do Arduino possibilita a criação de *sketches* para sua placa, na compilação o código escrito é traduzido para a linguagem C, em seguida é transmitido para o compilador avr-gcc, que realiza a tradução final de seus comandos em código binário, por final este último arquivo é gravado na placa Arduino via comunicação serial pela porta USB, esse ciclo está descrito na Figura 1.5.

No projeto será utilizado o Arduino UNO R3, Figura 1.6, baseado no microcontrolador Atmega328, possui 14 pinos digitais de entrada/saída, 6 entradas analógicas, um ressonador cerâmico de 16 MHz, uma conexão USB, um conector de alimentação, uma porta ICSP (pos-

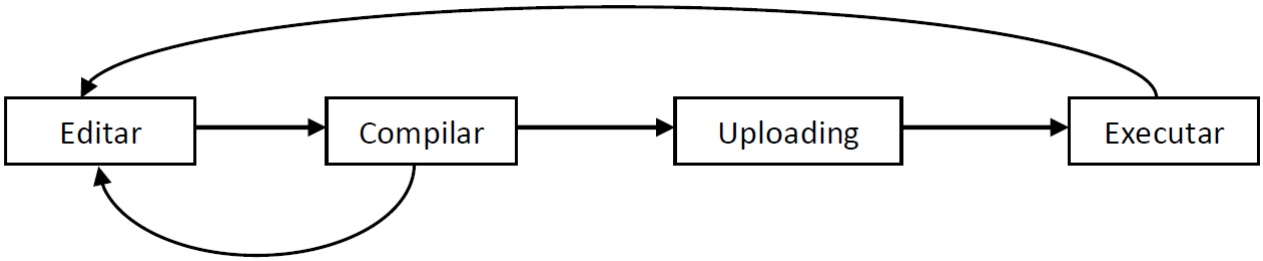


Figura 1.5: Ciclo de compilação e instalação do *software* no arduino.

sibilita a gravação de microcontroladores atmega em circuito), e um botão de reset. Segue as características:

Microcontrolador	ATMega328
Tensão de funcionamento	5V
Tensão de Entrada	7-12V
Pinos Digital I/O	14 (dos quais 6 oferecem saída PWM)
Pinos de Entrada Analógica	6
Corrente DC por Pino I/O	40 mA
Corrente DC para o Pino de 3,3V	50V
Memória Flash	32KB (ATMega328), dos quais 0,5 são utilizados pelo sistema de inicialização
SRAM	2KB (ATMega328)
EEPROM	1KB (ATMega328)
Clock	16MHz

Tabela 1.1: Configuração do Arduino UNO [1].

O Arduino pode ser estendido utilizando Shields, placas de circuito que adicionam funcionalidades como receptor de GSM (Figura 1.7 item 3), módulos Wifi (Figura 1.7 item 1), controlador de Motores (Figura 1.7 item 5) etc. Os Shields são conectados ao Arduino e estendem os pinos até o topo para que todos estes estejam disponíveis para uso.

1.2.1 IDE

A IDE Arduino, Figura 1.8, possui três componentes principais: a Toolbar no topo, a Sketch Window central (código) e a janela de mensagens na base. O Toolbar possui sete botões cujas funcionalidades são descritas na Tabela 1.2.

O monitor serial, Figura 1.9 é uma ferramenta muito útil para depuração do código, este exibe os dados enviados pelo Arduino (USB ou Placa serial), como também fornece a funcionalidade de enviar dados de volta.

O IDE do Arduino é bem simples; é possível aprender e utilizá-lo com rapidez e facilidade, principalmente se possui conhecimento na programação C.

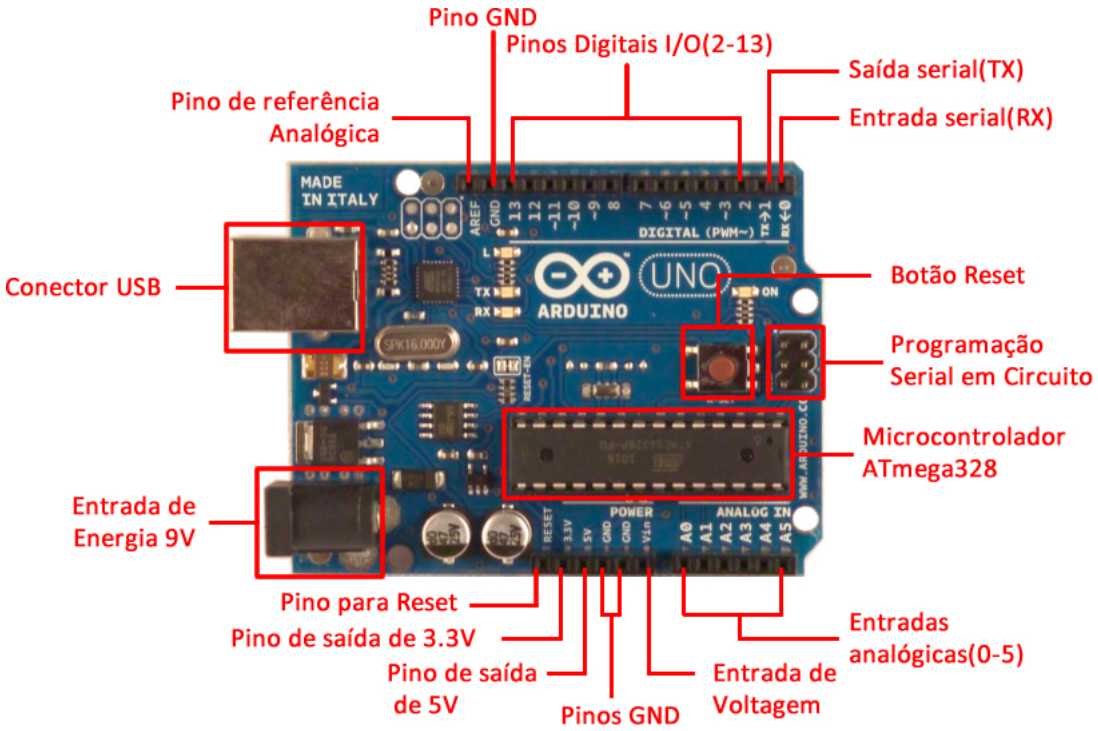


Figura 1.6: Descrição visual dos componentes do Arduino UNO, imagem montada segundo dados do site www.arduino.cc .

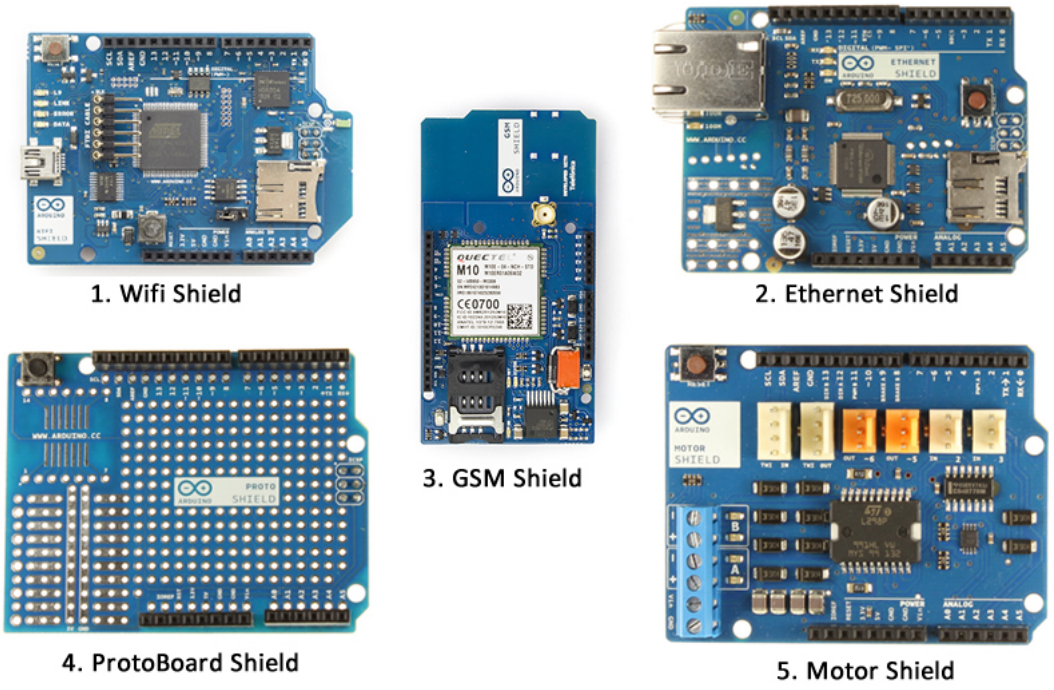


Figura 1.7: Principais shields utilizados [1] .

1.2.2 Microcontrolador

Um microcontrolador é construído de forma a integrar diversos componentes num único circuito integrado, evitando, assim, a necessidade de adicionar componentes externos ao

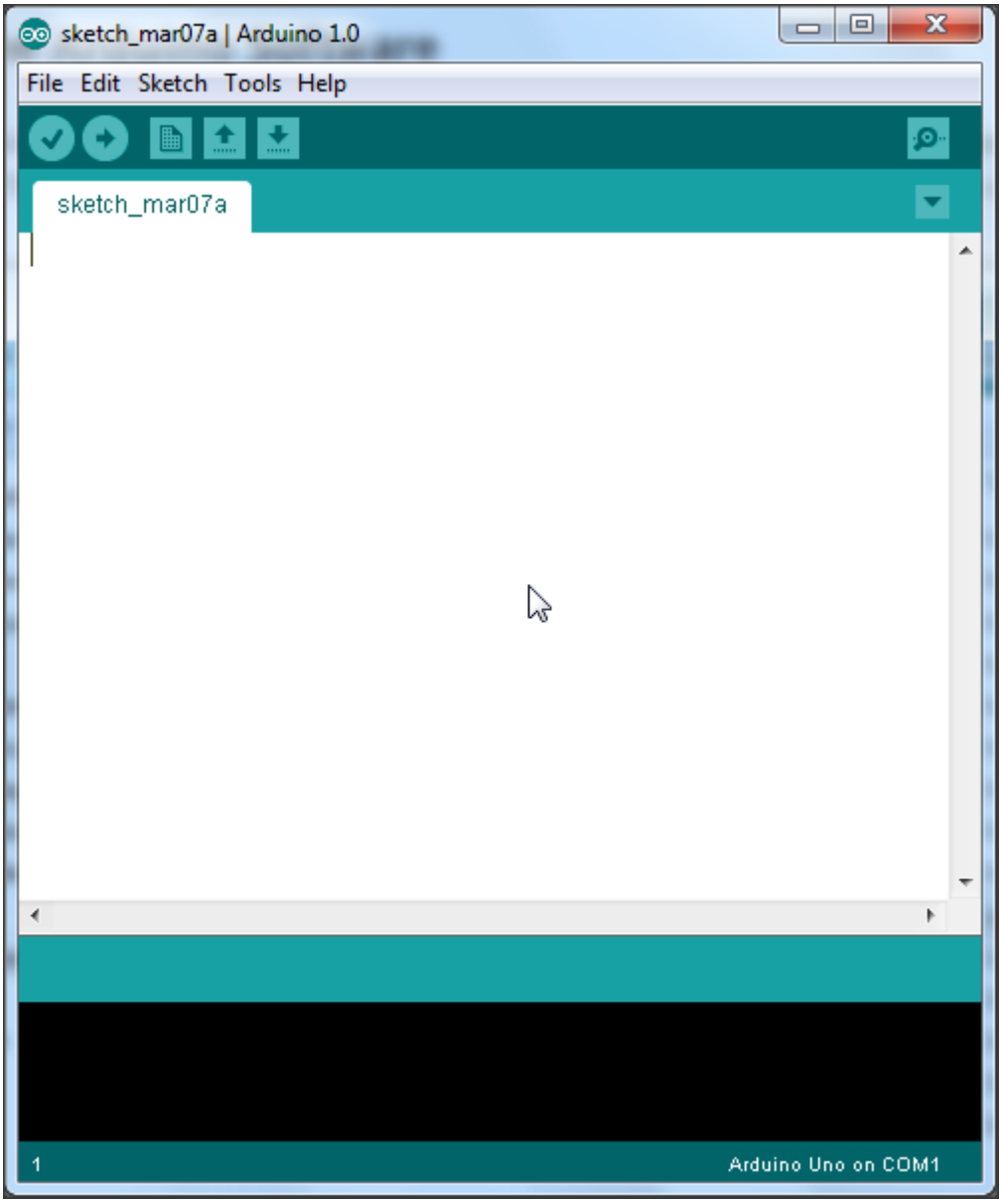


Figura 1.8: Tela inicial da IDE Arduino.

microcontrolador, que permitem a execução de todas as suas funcionalidades [26]. Este componente é composto por: memória RAM, memória ROM, portas seriais, entradas e saídas digitais e analógicas, temporizadores, CPU, UART etc.

São três as memórias utilizadas nos microcontroladores: memoria *flash* que permite o armazenamento do bootloader (programa de inicialização) e o/textit sketch a ser executado; a memória SRAM (Static Random Access Memory), que se comporta de forma semelhante a memória RAM dos computadores convencionais, é na SRAM que o programa é executado, esse tipo de memória apenas armazena os dados enquanto permanecer com energia; já a memória EEPROM (Electrically-Erasable Programmable Read-Only Memory), terceiro tipo de memória do microcontrolador, é utilizada para armazenar dados estáticos, ou seja, dados persistentes, que não são apagados quando o microcontrolador é desligado. O Arduino UNO

Verify/Compile	Verifica se há erros no código
Stop	Interrompe o monitor serial
New	Cria um sketch em branco
Open	Abre uma lista de sketches salvos para abrir
Save	Salva o sketch atual
Upload	Faz o upload do sketch para o Arduino
Serial Monitor	Exibe os dados seriais enviados do Arduino

Tabela 1.2: Funcionalidades do Toobar.

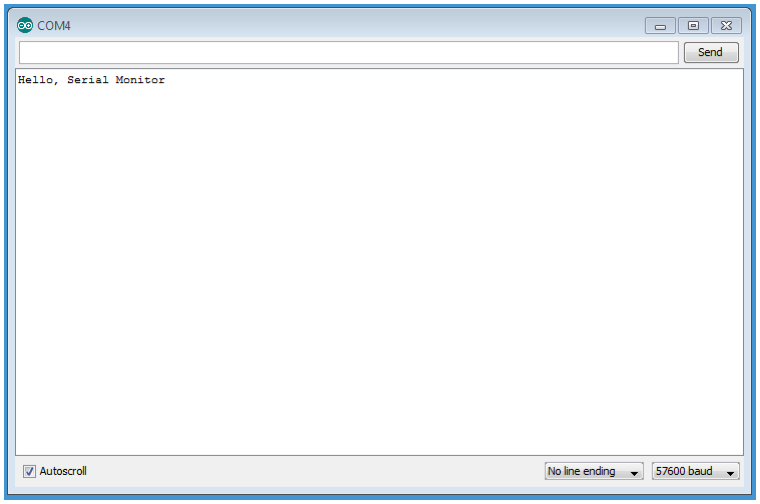


Figura 1.9: Tela do monitor serial.

utiliza o microcontrolador ATMega328, que possui 32kb de memória flash, 2kb de memória SRAM e 1kb de memória EEPROM. Para comunicação com o mundo externo, o ATMega possui quatorze pinos digitais 3 analógicos, uma saída de 5v e outra de 3,3v; é importante referir que a corrente máxima por cada pino analógico e digital é de 40mA, e as saídas de 3,3v e 5v é de 50mA. Das quatorze saídas digitais do Arduino UNO, seis possibilitam saída do sinal em PWM (*Pulse Width Modulation*), tem uma grande aplicação pois permite obter tensão analógica a partir de um sinal digital. O microcontrolador mencionado possui um conversor analógico digital de 10 bits, ou seja: $10^{20} = 1.024$

Como a tensão máxima de saída do pino é de 5V, para calcular a variação mínima de tensão detectada pela entrada analógica, dividimos: $5V/1.024 = 0,00488 \simeq 5mV$

1.2.3 Software - Funções Básicas

void setup()

Esta é executada uma vez quando o Arduino é iniciado, esta função é utilizada para inicializar variáveis e configuração dos pinos, esta função só é executada novamente quando

acionado o botão reset, ou quando o microcontrolador é desligado e ligado novamente.

Exemplo de utilização:

```
1
2  void setup(){
3        pinMode(3, INPUT); //Defini o pino 3 como input
4  }
```

Código 1.1: função setup().

void loop()

Essa função cria um laço de repetição infinito, semelhante com o comando `while(true)`, todas as instrução existentes em seu interior são repetidas infinitamente. É nessa função que colocamos o código propriamente dito do nosso projeto, nela podemos ler sucessivamente as portas de entrada, ler as leituras provenientes de sensores, mudar a voltagem dos pinos de saída como resposta a um determinado evento etc.

Exemplo de utilização:

```
1
2  int val;
3  void loop(){
4    val = analogRead(3); // Permite a leitura analogica do pino 3
5     if(val > 500){
6       digitalWrite(13, HIGH); //Atribui 5V para o pino 13
7    }
8  }
```

Código 1.2: funcao loop().

Variáveis

As variáveis declaradas podem ter o escopo global ou local, variáveis globais são semelhantes ao C/C++, basta declarar a variável fora das função, a variável deve ser declarada antes de sua utilização pela função. Os tipos suportadas estão listadas na Tabela . O Arduino também suporta conversão de variáveis, as funções utilizadas são: int(x), long(x), float(x), char(x), byte(x); onde x é o valor a ser convertido.

Funções

1. `pinMode(_numeroDoPino, _modo)` - determina o comportamento do pino, e dessa forma podemos defini-lo como *input* ou *output*, normalmente é utilizado dentro da função `setup();`

Nome	**Descrição**
`boolean`	`"true"` ou `"false"`
`char`	Pode guardar um único caractere da tabela *ASCII*, ocupa 1 *byte* de memória.
`byte`	Armazena um número do tipo `unsigned`, entre 0 e 255, ocupa 1 *byte* na memória.
`int`	Possibilita guardar um valor inteiro de 2 bytes, ou seja, números entre -32768 e 32767. Podemos declarar a variável`int` como `unsigned`, dessa forma deixamos de ter a parte negativa e podemos armazenar valores entre 0 e 65535.
`long`	Pode guardar valores numéricos de até 4 bytes, valores entre -2147483648 e 2147483647. Podemos declaras a variável `long` como `unsigned`, dessa forma deixamos de ter a parte negativa e podemos armazenar valores entre 0 e 4294967295.
`float`	Apresenta uma maior resolução que a variável do tipo inteiro, ocupa 4 bytes de memória interna, pode conter valores no intervalo de $3,4028235 \times 10^{38}$ e $-3,4028235 \times 10^{38}$.
`double`	Possui as mesmas características e a mesma precisão que o tipo `float`. Não foi encontrato motivo para que esses dois tipo de variáveis coexistam.
`array`	Armazena um conjunto de variáveis do mesmo tipo.

Tabela 1.3: Variáveis suportadas pelo Arduino

2. `digitalWrite(_numeroDoPino, HIGH/LOW)` - utilizado para determinar se o pino de saída, configurado como *output* utilizando a função `pinMode`, está com valor alto(`HIGH` = 5V) ou baixo (`LOW` = 0V);

3. `digitalRead(_numeroDoPino)` - possibilita a leitura de um pino de entrada digital, configurado como *input* utilizando a função `pinMode`. O retorno é um valor lógico 0 ou 1, ou seja, `LOW` ou`HIGH` respectivamente;

4. `analogRead(_numeroDoPino)` - possibilita a leitura do valor analógico do pino especificado, desde que este possua um conversor *A/D*, valor é um inteiro entre 0 e 1023, que representa a grandeza de diferença de potencial compreendida entre 0V e 5V.

5. `analogWrite(_numeroDoPino,_valor)` - possibilita a utilização dos pinos *PWM (Pulse Width Modulation)*;

6. `millis()` - retorna a quantidade de tempo, em milissegundos na forma de uma variável do tipo `unsigned long`, que passou desde que o programa atual começou a ser executado, ou seja, do momento em que o Arduino foi inicializado;

7. `micros()` - mesmas características da função `millis()`, porém a variável `long` irá representar microssegundos;

8. `delay(_milisegundos)` - pausa a execução do programa por uma quantidade de tempo, em milissegundos, especificada;

9. `min(_valor1, _valor2)` - retorna o menor valor entre os dois valores;

10. `max(_valor1,_valor2)` - retorna o maior valor entre os dois valores;

11. `abs(_valor)` - instrução que retorna o módulo de um número;

12. `constrain(_valorTestar, _valorMenor, _valorMaior)` - limita o valor de retorno a um intervalor especificado no parâmetro 2 e 3.

    valor = valor se "valor" $\in$ [valor1, valor2]; valor = valor1 se valor < valor1; valor = valor2 se valor > valor2.

13. `pow(_valor, _expoente)` - calcula o resultado de um número a um determinado expoente;

14. `sqrt(_valor)` - calcula a raiz quadrada do valor especificado;

15. `sen(_valor)` - retorna o seno do valor em radianos;

16. `cos(_valor)` - retorna o coseno do valor em radianos;

17. `tan(_valor)` - retorna a tangente do valor em radianos;

18. `random()` - permite gerar números pseudo-aleatórios, pode ser utilizado de duas formas:

    `random(_valorMaximo)` – retorna um valor até o limite estabelecido como parâmetro, este, não está incluso no intervalo;

    `random(_valorMinimo, _valorMaximo)` – retorna um valor pertencente ao intervalo especificado, "Resultado" $\in$ [Valor mínimo; (Valor máximo) - 1].

Interrupções

Uma interrupção é a execução de uma função mediante a alteração de estado de um determinado pino, independente do ponto em que se encontra a execução do programa. O Arduino UNO possui 2 pinos que suportam interrupções, os pinos digitais 2 e 3.

Para configurar as interrupções é utilizada a função `attachInterrupt(_interrupcao,_funcao,_modo)`:

1. `_interrupcao`:indica o a interrupção a ser utilizada, o inteiro 0 está associado ao pino 2 e 1 ou pino 3;

2. `_funcao`: função que será chamada caso a interrupção ocorra, esta função não pode ter retorno, deve ser do tipo `void`, é conhecida como rotina de serviço de interrupção;

3. `_modo`: define quando a interrupção deve ser acionada, são definidas quatro constantes para definir seu comportamento:

   `LOW`: aciona a interrupção sempre que o pino tiver valor baixo;

   `CHANGE`: aciona a interrupção sempre que o pino sofrer alteração de valor;

   `RISING`: chama a interrupção sempre que o valor do pino passa de `LOW` para `HIGH`;

   `FALLING`: a interrupção é acionada toda vez que o valor do pino é alterado de `HIGH` para `LOW`.

A Figura 1.10 mostra a implementação de interruçao no pino 2, o tipo escolhido foi `CHANGE`, ou seja, sempre que o pino 2 mudar de estado, a funçao `piscar()` será executada.

Comunicação Serial A Comunicação Serial é utilizada para possibilitar o Arduino a se comunicar com outros dispositivos(Módulos GPS, GSM, WIFI, PC etc), é através deste canal que é realizado o *upload* do código para o microcontrolador.

O Arduino Uno possui um canal de comunicação que está ligado aos pinos RX e TX, pinos digitais 0 e 1 respectivamente, do ATmega328 , esse pinos são ligados ao ATmega16U2, responsável pela tradução do sinal para comunicação USB.

Figura 1.10: Exemplo de utilização de Interrupções.

A plataforma Arduino possui várias função que possibilitam o desenvolvimento de aplicações que utilizam comunicação serial, essas funções são utilizadas para envio e recebimento de dados, são elas:

1. `Serial.begin(_speed, _config)` – função utilizada para começar a comunicação serial, ela configura a taxa de transmissão, e a quantidade de *bytes* a ser enviado, esse último parâmetro é opcional;

2. `Serial.avaliable()` – retorna quantidade de bytes disponíveis para leitura, 64 é quantidade máxima de *bytes* no *buffer*;

3. `Serial.read()` – ler o primeiro *byte* do buffer de entrada serial;

4. `Serial.print()` – escreve na saída serial caracteres no formato ASCII, a função aceita números inteiros, flutuantes, caracteres e palavras. A função possui um segundo parâmetro, opcional, que define o tipo de base utilizada para formatar o valor enviado (BIN, DEC, HEX etc).

1.3 Motores

Os motores elétricos são atuadores mecânicos, ou seja, transformam energia elétrica em mecânica, basicamente utilizam imãs para se movimentar, utilizando princípios básicos do magnetismo.
Os motores mais utilizados na robótica são os de corrente contínua, de passo e os servomotores, apesar de todos seguirem o mesmo princípio, tem características e utilidades diferentes.

1.3.1 Motores de corrente contínua

A criação dos motores de corrente contínua, Figura 1.11, teve a contribuição de vários pesquisadores, começando com a descoberta do eletromagnetismo por Hans Oersted em 1820, e dando prosseguimento com pesquisas de William Sturgeon, Joseph Henry, Andre Marie Ampere, Michael Faraday, Thomas Davenport entre outros. O primeiro motor utilizando eletroímãs foi demonstrado por Anyos Jedik em 1828 [28].

O principal componente dos motores elétricos são as bobinas, que consistem em enrolamento de fios. Quando a corrente elétrica passa de uma extremidade a outra é gerado um campo magnético, o torque produzido por este campo magnético gira o eixo do motor, gerando energia cinética, esse é o princípio usado nos motores elétricos [22].

Para fazer o motor funcionar é necessário fornecer-lhe energia elétrica na faixa de tensão ideal, indicada na sua especificação. Se a tensão for baixa o motor funcionará, porém, com baixa potência, ao aumentar a tensão, o motor terá sua potência incrementada, entretanto, quanto maior a tensão, menor o tempo de vida do motor [22].

A maioria dos motores CC tem velocidades entre 3000 e 9000 rotações por mínuto, que aquivale a 50 e 150 revoluções por segundo, respectivamente. Apesar da alta velocidade, esse motores possuem um baixo torque, e infelizmente os robôs necessitam de torque elevado, para se mover, puxar cargas e levantar seus manipuladores [22].

Para realizar atividades que necessitam de torque, os motores CC são combinados com diversos tipos de engrenagens, através da combinação de engrenagens com diferentes raios,

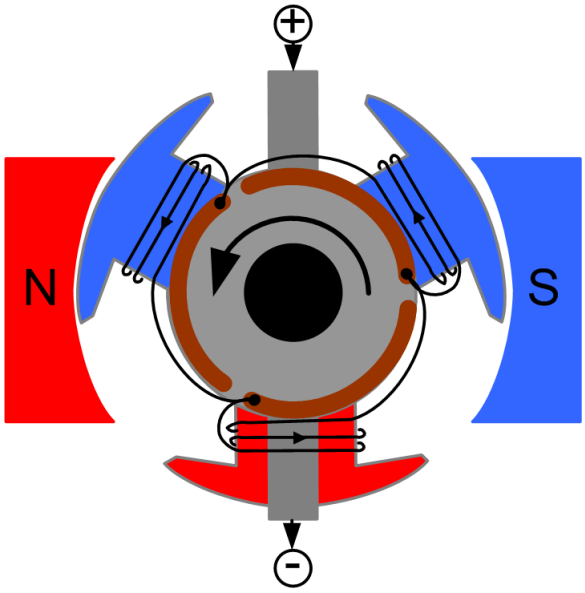


Figura 1.11: Esquema do motor de Corrente Contínua.

Figura 1.12, podemos manipular a quantidade de força que será gerada. Se a engrenagem de saída é maior do que a de entrada, o torque aumenta. Se a engrenagem de saída é menor do que a de entrada, o torque diminui. Os equipamentos que implementão essas características são chamados de Caixas de Redução [22].

Motores CC, Figura 1.13, geralmente são utilizados para locomoção de robores pneumáticos ou de esteiras, que não necessitam de grande precisão em seus movimentos, Figura 1.14.

1.3.2 Servomotores

Servomotores, Figura 1.15, são motores elétricos com a capacidade de girar seu eixo para uma posição específica. São muito utilizados na fabricação de brinquedos, tais como: ajuste de direção de carros e controle da direção das asas de aviões de controle remoto. Esta posição, normalmente, está entre 0º e 180º graus a partir de um ponto de referência, dessa forma, o eixo do servo motor está limitado a uma rotação de 180º graus [22].

Para definir o ângulo, é enviado ao motor um sinal no formato *PWM (Pulse Width Modulation)*, esse sinal pode ter entre 0 e 5 volts. O circuito do servo motor faz o monitoramento desse sinal e percebe alterações desse valor durante 1 até 2 milisegundos, a duração desse pulso define a posição do braço do motor [22].

Robôs do tipo *Hexapod* [4], Figura 1.16, utilizam diversos servomotores para movimentar

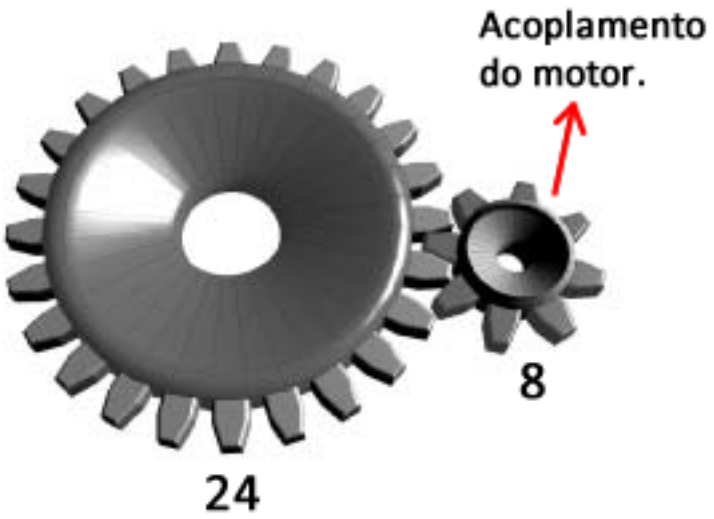


Figura 1.12: Caixa de redução para aumentar o torque

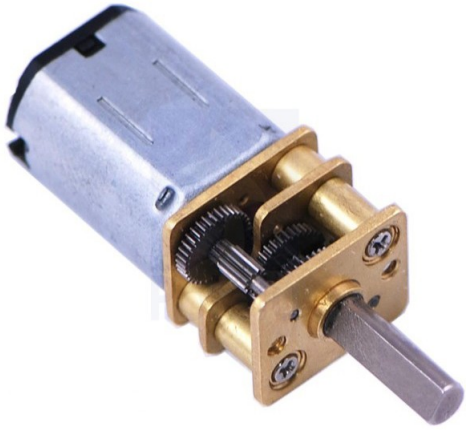


Figura 1.13: Motor de Corrente Contínua com Caixa de Redução acoplada.

Figura 1.14: Robo que utiliza motor CC para se mover

.

Figura 1.15: Imagem de servomotor

suas ”pernas”.

Figura 1.16: Robô haxapod controlado por *Arduino*

1.3.3 Motor de Passo

Motores de passo, Figura 1.17, são motores de corrente contínua que possuem duas ou mais bobinas independentes, Figura 1.18, estas bobinas devem ser energizadas em intervalos fixos para manter a rotação do eixo do motor, ou seja, para que o giro ocorra, o rotor deve ser conduzido em uma sequência horária ou antiorária de energização [29].

Os motores de passo possuem um número definido de passos ou intervalos magnéticos, cada vez que uma bobina é ligada em sequência, o eixo do motor roda um passo. Esses motores são utilizados em aplicações que necessitam de grande precisão como impressoras, máquinas CNC e algumas aplicações de robôtica [29].

Motores de passo são utilizados em braços robóticos, Figura 1.19, esses robôs necessitam de grande precisão para realizar suas tarefas.

Figura 1.17: Imagem de motor de passo muito utilizado na fabricação e impressoras 3D

Figura 1.18: Esquema interno de um motor de passo

Figura 1.19: Imagem de braço robótico que utiliza motores de passo.

1.4 Baterias

Historicamente a primeira bateria foi inventada em 1800 por Alessandro Volta, Figura 1.20, daí por diante essa tecnologia avançou muito, possibilitando novas descobertas, invenções e aplicações. As indústrias de eletrônicos, informática e comunicação, engenharia elétrica e grande parte das indústrias de química foram fundadas nas descobertas possibilitadas pela bateria [28].

As baterias utilizam um fenômeno químico chamado de reação redox, que ocorre quando os elétrons são transferidos do átomo oxidado para o átomo reduzido, esses elementos são chamados de Anodo e Catodo, respectivamente. A energia liberada por essa reação pode ser utilizada para realizar trabalho elétrico [27]. A combinação de materiais utilizados como anodo e catodo, caracterizam o tipo da bateria, voltagem fornecida por célula, se a bateria poderá ser recarregada, a densidade de armazenamento (razão entre quantidade de carga elétrica e peso) e quantidade de ciclos de vida. Hoje em dia são utilizados vários tipos de baterias: bateria de níquel cádimo NiCd, de hidreto metálico de níquel Ni-MH, Acumulador de Chumbo, de íon-polímero LiPo.

1.4.1 Níquel Cádimo – NiCd

A bateria de NiCd, Figura 1.21, foi a segunda bateria recarregável a ser desenvolvida, foi utilizada durante vários anos como a principal bateria dos equipamentos eletrônicos, porém

Figura 1.20: Primeira bateria criada por Alessandro Volta

está sujeita a uma condição chamada de “memória”, que ocorre quando é carregada antes da descarga completa, isso faz com que a vida útil da bateria seja reduzida consideravelmente. Cada célula possui cerca de 1,2 volts, e estão normalmente disponíveis em tamanhos padrão de bateria alcalina tipo AA e AAA [29].

Figura 1.21: Bateria de NiCd no padrão AAA

1.4.2 Hidreto Metálico de Níquel - NiMH

Essa bateria, Figura 1.22 tem as mesmas aplicações que a bateria de NiCd, porém praticamente não sofrem o efeito “memória”, oferece elevado grau de armazenamento para o seu

tamanho, muitas vezes entre 1000 mAh a 4500 mAh, e tipicamente encontradas em células de 1,2v, tem vida útil entre 400 e 600 recargas, devido ao material utilizado em sua fabricação possui um preço mais elevado que as baterias de NiCd [29].

Figura 1.22: Bateria NiMH utilizada tem telefones sem fio

1.4.3 Polímero do Lítio - LiPo

Bateria de polímero de lítio, Figura 1.23, é uma das tecnologias mais atuais, possuindo uma alta relação potência-carga, geralmente cada célula desta bateria tem 3,7 v. Essas baterias são leves e têm a capacidade de fornecer uma grande quantidade de corrente muito rapidamente, têm o valor acessível e são muito utilizadas em aeromodelismo e automodelismo. As baterias LiPo estão disponíveis em pacotes de 7,4 v (2 células), 11,1 v (3 celulas), 18,5 v (4 celulas) e 22,2 v (5 celulas) [29]. Apesar da grande capacidade desta bateria, devemos tomar cuidado com a sua carga e descarga, pois se sua diferença de potencial por célula for inferior a 3,0 volts, esta corre um grande risco de se incendiar, esse mesmo fato pode ocorrer caso a bateria não seja carregada de forma adequada [29].

1.4.4 Bateria de Chumbo-Ácido

Esse tipo de bateria, Figura 1.24, é utilizada nos carros, barcos, sistema de energia solar, sistemas de *backup*. São pesadas e volumosas, porém tem a capacidade de fornecer uma grande potência e uma excelente capacidade de armazenamento, estão disponíveis no mercado baterias de 5 Ah a 150 Ah. Possui placas de chumbo disposta em série, cada uma produz cerca de 2 v, a espessura dessas placas determina o tipo de utilização da bateria [29].

Cada tipo de bateria tem suas características, vantagens e desvantagens, a tabela 1.4 faz um comparativo das baterias citadas acima.

Figura 1.23: Bateria LiPo 2200 mAh utilizada em aeromodelismo

Figura 1.24: Bateria de chumbo de 1,2 Ah utilizada em Nobreaks

Tipo	Voltagem	Volts/Cel	Células	Peço em U$	Peso	Amp/Hora
LiPo	11,1 v	3,7 v	3	$32,00	367 g	5000 mAh
NiCad	12 v	1,2 v	10	$49,99	907,2 g	5000 mAh
NiMh	12 v	1,2 v	10	$49,99	907,2 g	5000 mAh
Chumbo - Ácido	12 v	2 v	6	$15,99	1814,4 g	5000 mAh

Tabela 1.4: Tabela comparativa dos principais tipos de bateria, dados fornecidos por [29]

1.5 Energia Solar

Com a escassez de combustíveis fósseis, fontes de energia alternativa estão sendo cada vez mais importante para nossa sociedade. A energia solar é uma das mais promissoras, já que pode ser encontrada facilmente em praticamente todas as regiões do nosso planeta [20].

O efeito fotovoltaico foi identificado pela primeira vez em 1839 por Becquerel. Ele detectou uma diferença de potencial entre dois eletrodos imersos em um eletrólito, quando incidisse luz. A primeira célula solar de silício, com as principais características semelhantes às atuais, foi desenvolvida por Gerald Pearson, Calvin Fuller e Daryl Chapin em 1954 [16].

A célula solar ou fotovoltaica é o dispositivo que possibilita converter radiação solar em energia elétrica, isto é, a incidência de fótons no dispositivo produz uma diferença de potencial e corrente elétrica. Quando uma célula solar é exposta ao sol, os fótons que incidem geram pares de elétrons-lacunas (ausência de elétrons) em uma região da célula, essas lacunas provocam a extração de elétrons em outra região da célula, Figura 1.25, dessa forma, se este dispositivo for conectado a um circuito externo obtêm-se uma corrente elétrica e uma diferença de potencial [16].

Hoje existem painéis fotovoltaicos com as mais variadas potências, desde placas maiores para uso residencial de 60 a 150 watts,Figura 1.26, até placas menores para uso em dispositivos portáteis geralmente entre 1 a 2 watts, Figura 1.27.

1.6 Sensores

No mundo da robótica o sensor pode ser considerado o “sentido” do robô, ou seja, é por meio deste que a máquina irá perceber o ambiente interno e externo, e dessa forma ter a capacidade de tomar decisões sobre os eventos ocorridos, atingir metas e agir com inteligência [22].

O avanço da tecnologia vem desenvolvendo a cada dia sensores dos diversos tipos e características, impulsionando o desenvolvimento da robótica. Um robô tem normalmente dois tipos de sensores [22]:

- Sensores Proprioceptivos: Estes percebem o ambiente interno da máquina. Geral-

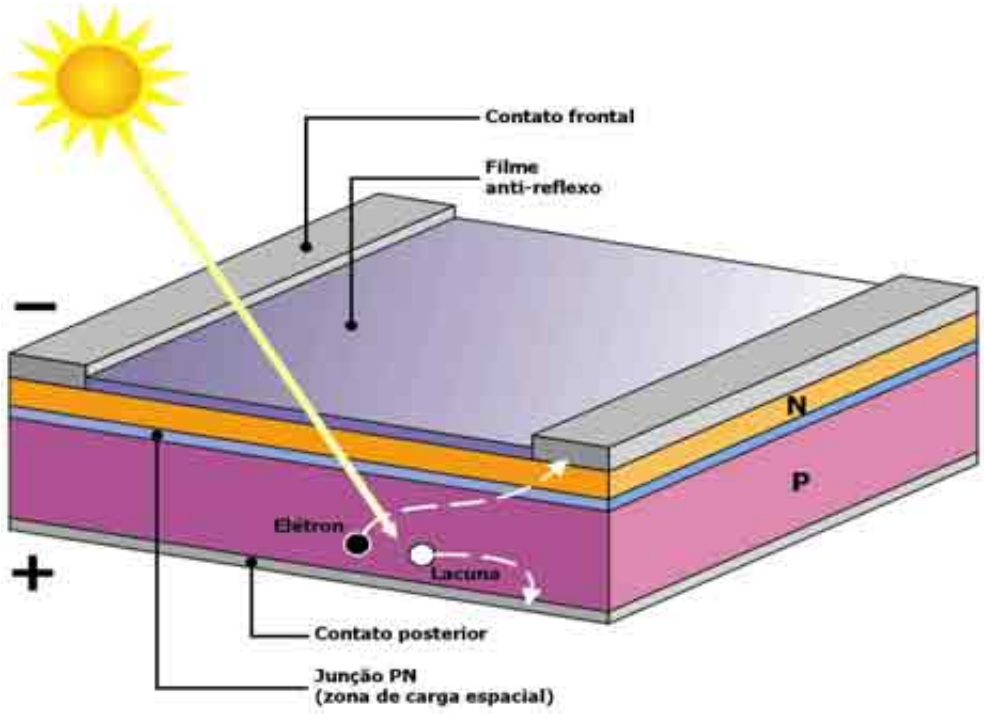


Figura 1.25: Esquema de funcionamento de uma placa solar, com suas regiões.

Figura 1.26: Placa solar residencial de 150 watts.

Figura 1.27: Micro placa fotovoltaica de 0,05 watts.

mente são utilizados para verificar posição ou angulação de roda, braço, câmeras e outros sensores.

- Sensores Exteroceptivos: Estes sensores percebem o ambiente externo ao robô tal como os níveis de luminosidade, distância a um objeto, presença de som, movimento, pressão, temperatura, umidade etc.

A utilização desses dois tipos de sensores constituem o sistema preceptor de um robô [22]. Segue na tabela 1.5 os tipos de sensores mais utilizados:

Propriedade Física	**Tecnologia de Sensor**
Contato	*bump*, *switch*
Distância	Ultrassom, radar, infra vermelho
Nível de luminosidade	Fotocélulas e câmeras
Nível de som	Microfones
Aceleração	Acelerômetro e giroscópio
Rotação	Encoders e potenciómetros
Magnetismo	Magnetismo
Cheiro	Sensores químicos
Temperatura	Termômetro e infra vermelho
Inclinação	Inclinômetro e giroscópio
Pressão	Medidos de pressão
Altitude	Altímetro

Tabela 1.5: Alguns sensores utilizados na robôtica

Apesar dos grandes avanços os sensores sofrem com problemas de precisão, uma vez que os dados capturados por estes podem sofre interferências diversas, causados pelo meio externo ou pelos seus próprios componentes internos [22].

Figura 1.28: Botão utilizado nas interfaces diversos equipamentos eletrônicos

Figura 1.29: LDR

1.6.1 Sensor de Contato

Esse tipo de sensor, Figura 1.28, é provavelmente o mais simples, seu funcionamento ocorre utilizando uma base binária, circuito aberto ou fechado. Esse princípio pode ser aplicado em diversas situações, entre elas:

- Detectar contato com outro objeto;
- Estágio de posicionamento e limite de curso;
- Sensor de eixo de encoder para detectar quantidade de giro de um motor;
- Para acionamento de funcionalidades da interface dos dispositivos eletrônicos, botões de mouse, teclado, telefone, calculadora, entre outros.

1.6.2 Sensor de Luminosidade

Sensores de luminosidade, Figura 1.29, medem a quantidade de luz sobre uma célula fotoelétrica. Essas células têm sua resistência elétrica interna dependente do índice de luminosidade, a resistência é baixa quando a célula está sendo iluminada e alta no escuro [22]. Esse tipo de sensor normalmente é conhecido como LDR (*Light Dependent Resistor*).

A fotocélula tem uma trilha ondulada em seu interior, o material que forma essas ondulações é responsável pela variação da resistência. Esses sensores podem detectar uma ampla gama de comprimento de ondas, incluindo luz ultravioleta e infra-vermelho [22].

1.6.3 Sensor Ultrasônico de Distância

Esse sistema utiliza frequências sonoras, não perceptíveis a audição humana, para verificar a existência de obstáculos. O sensor, Figura 1.30, é constituído por um emissor e um receptor. Quando o emissor emite a frequência, o sensor inicia um temporizador que é interrompido quando o receptor reconhece o retorno da frequência, dessa forma, conhecendo o tempo que a frequência levou para percorrer o caminho de ida e volta, e a velocidade do som, podemos calcular a distância entre o sensor e o obstáculo, para isso utilizamos a fórmula [22]:

$$Distância = (TempoParaRetornoDaFrequencia * VelocidadeDoSom)/2$$

A velocidade do som pode ser considerada igual a 340 m/s, na fórmula a divisão por 2 deve-se ao fato de que a onda é enviada e rebatida, percorrendo o dobro a distância até que o sensor capte o eco [22].

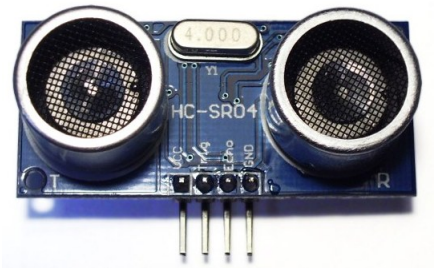


Figura 1.30: Sensor Ultrasônico HC-SR04, bastante utilizado com o Arduino

1.6.4 Sensor de Corrente elétrica

O sensor de corrente,Figura 1.31, é utilizado para medir a intensidade de corrente que passa por um determinado ponto do circuito. Esse sensor é bastante útil para mensurar o gasto de energia de um circuito, e dessa forma, possibilitar o desenvolvimento de técnicas para aumentar o tempo de autonomia do robô [23].

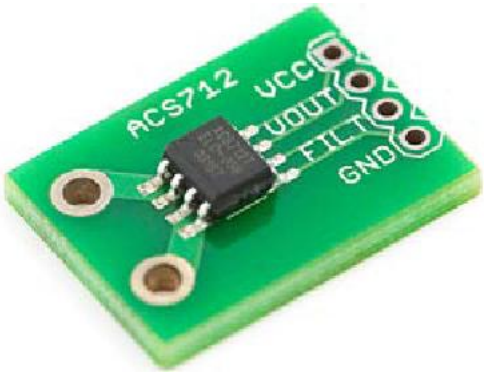


Figura 1.31: Sensor de Corrente ACS 712, muito utilizado em projetos de robótica

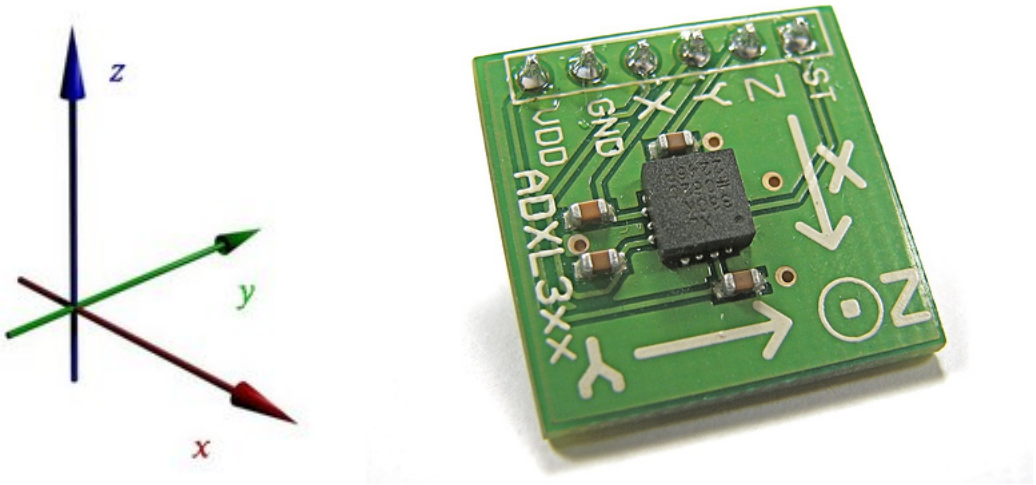


Figura 1.32: Acelerometro com detecção da angulação dos 3 eixos.

1.6.5 Acelerômetro

O acelerômetro, Figura 1.32, é utilizado para medir a força gravitacional ou aceleração. Ao inclinar esse sensor é possível medir a força gravitacional em relação a quantidade de inclinação. Esses sensores possuem até 3 eixos de detecção [29].

Hoje em dia muitos dispositivos utilizam o acelerômetro para detecção de choque mecânico, estabilização de plataformas, auto nivelamento ou inclinação, acessibilidade, entre outros [29].

1.6.6 Sensor de Temperatura

O sensor de temperatura converte temperatura em um sinal elétrico através de transdutores. Existem 4 (quatro) tipos de transdutores: RDTs, Termistores, CI Sensores, Pares termoelétricos [12].

- **RDTs:** *Resistance Temperature Detectors*, Figura 1.33, são resistores que variam sua resistividade de acordo com a temperatura, a maioria usa como material sensor a platina. A corrente de medida que atravessa esse sensor pode provocar o seu aquecimento, e dessa forma, criar medições erradas de temperatura, essa é a principal desvantagem na utilização deste tipo de sensor [12].

- **Termistores**: Figura 1.34, Tem as mesmas características dos RDTs, no entanto são fabricados com materiais cerâmicos semicondutores e apresentam um volume muito pequeno, refletindo uma baixa capacidade térmica e dessa forma menor possibilidade de afetar a medição do corpo em questão [12].

- **CIs Sensores:** A grande vantagem da utilização desse tipo de sensor é que eles permitem obter uma resposta linear, no entanto, devido ao seu tamanho e a necessidade de

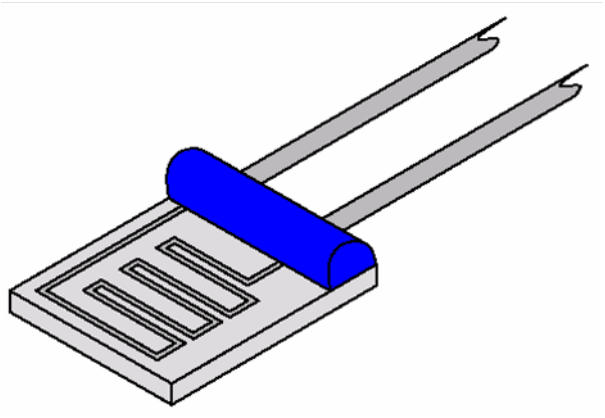


Figura 1.33: Desenho de um RDTs

Figura 1.34: Termistor NTC 20K (SEN005)

alimentação externa, o aquecimento desse sensor pode alterar o resultado da medição [12].Um CI bastante utilizado é o LM35, Figura 1.35.

A indústria está cada vez mais fornecendo sensores “espertos”, tendo uma maior capacidade de fornecer as informações corretas para computação [12].

Figura 1.35: LM35 - CI sensor de temperatura bastante popular

- **Pares Termoelétricos:** Esse tipo de sensor é indicado para medida de temperaturas

muito elevadas e não precisam de fonte de alimentação. O princípio de funcionamento, Figura 1.36, consiste na junção de dois metais diferentes submetidos a temperaturas diferentes, esse fato ocasiona uma diferença de tensão proporcional à diferença de temperatura, dessa forma, a voltagem indicada nos terminais do sensor é traduzida em temperatura [12].

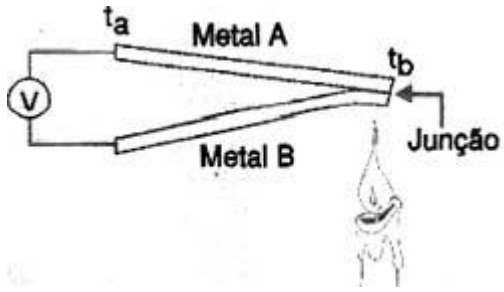


Figura 1.36: Esboço do funcionamento de um Par Termoelétrico

1.6.7 Sensor de Som (Microfone)

Microfone é um dispositivo que converte vibrações sonoras em sinais elétricos, são aplicados em telefones, equipamentos de gravação e sensores de distância baseados em ultrasom [25]. O sensor SEN-00001, Figura 1.37, possui muita documentação de utilização com o Arduino.

A grande parte dos microfones utilizam um diafragma que vibra quando as ondas sonoras incidem sobre ela, esta por sua vez induz uma corrente variável numa bobina eletromagnética [25].

1.6.8 Sensor de Vibração

O sensor de vibração é utilizado pra captar vibração de estruturas físicas, convertendo essas vibrações em sináis elétricos. São muito utilizados na industria para verificar funcionamento de motores, centrífugas ou qualquer equipamento que utilize algum tipo de movimento em seu funcionamento. Esse monitoramento ajuda na prevenção de danos nos equipamentos, uma vez que o mau funcionamento pode ser identificado no início [25].

Esses sensores são geralmente construídos de materiais piezoelétricos, Figura 1.38,que possuem capacidade de gerar tensão apartir de um esforço mecânico [25].

1.6.9 Sensor de Campo Magnéico

O sensor de campo magnético, como seu nome diz, destina-se a medir campos magnéticos (que podem ser os produzidos pela terra ou gerados através de imãs), é muito utilizado nos aviões, pois com a medida do campo magnético terrestre, é possível, junto com o acelerômetro, monitorar a direção e a inclinação do aparelho. Também é possível utiliza-lo para medir

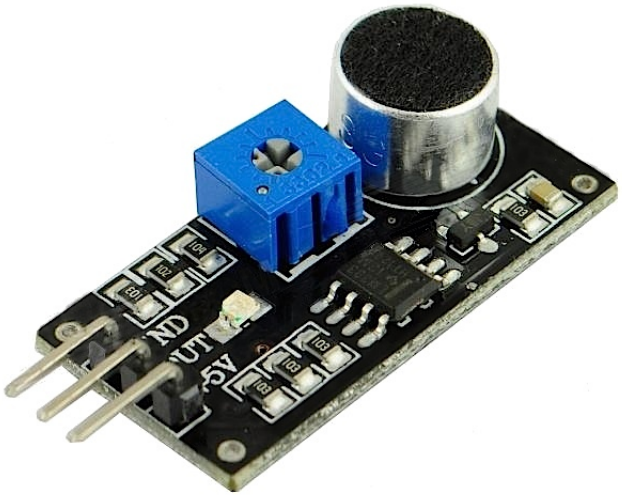


Figura 1.37: Sensor de som SEN-00001 muito utilizado com o Arduino

Figura 1.38: Sensor de Vibração Piezoelétrico Meas

a rotação de um motor, fazendo a leitura da variação do campo magnético exercido pelo imã do motor [25]. Segue Figura 1.39 o sensor 177 725z, utilizado para medição de campos magnéticos.

Figura 1.39: 177 725z - Sensor de campo magnético

Esse sensor também pode ser utilizado para medir corrente elétrica. Um fio que é percorrido por uma corrente elétrica gera um campo magnético ao seu redor, medindo esse campo e utilizando as fórmulas que os relacionam, é possível verificar a quantidade de corrente conduzida pelo fio [25].

1.6.10 Sensor de peso

Através deste sensor é possível medir o peso de algum objeto ou uma força aplicada sobre ele, os modelos cuja utilização é mais simples são os resistivos, quanto maior a força exercida sobre ele, menor será a resistência entre seus terminais [25].

Os sensores de peso disponíveis no mercado precisam de cuidado em sua utilização, pois possuem peso máximo suportado, exercendo uma força maior que a recomendada, o sensor pode ser danificado permanentemente [25].A Figura 1.40 traz o sensor de peso IESP-12 e SF4.

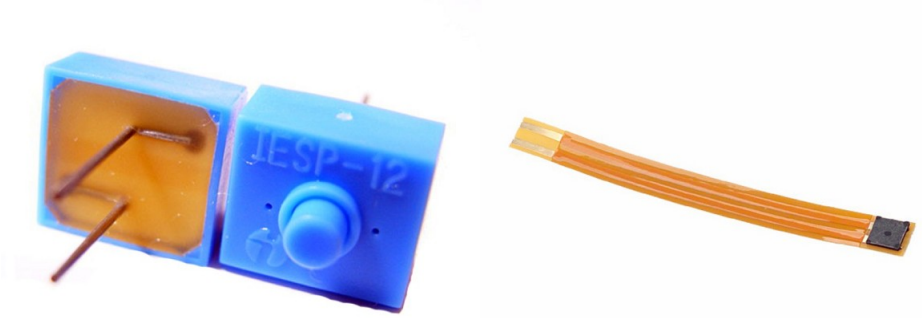


Figura 1.40: Sensores de peso IESP-12 e SF4

1.6.11 Codificadores Óticos

Codificadores Óticos ou Encoders são utilizados para fornecer posição angular com base no ângulo de rotação de seu eixo. Esse tipo de sensor, utiliza um disco acoplado em seu eixo contendo uma ou mais trilhas de pequenas janelas. De um dos lados do disco são posicionados diodos emissores de luz (LED), e do outro, fotodetectores, cada vez que uma janela passa defronte o LED, o fotodetector emite um sinal. A combinação dos sinais dos diversos detectores nas diferentes trilhas fornece uma codificação única para cada posição angular. A Figura 1.41 mostra um motor DC com Codificador Ótico interno.

Os codificadores utilizam duas técnicas de codificação da trilha: código binário e código Gray. O código binário, Figura 1.42, pode ser utilizado diretamente, porém pode apresentar ruído durante a transição das janelas. No código Gray, Figura 1.43, ocorre apenas um bit de transição em cada janela, eliminando eventuais ruídos, contudo, este código necessita de uma tabela de conversão para o código binário.

Figura 1.41: Motor com codificador interno

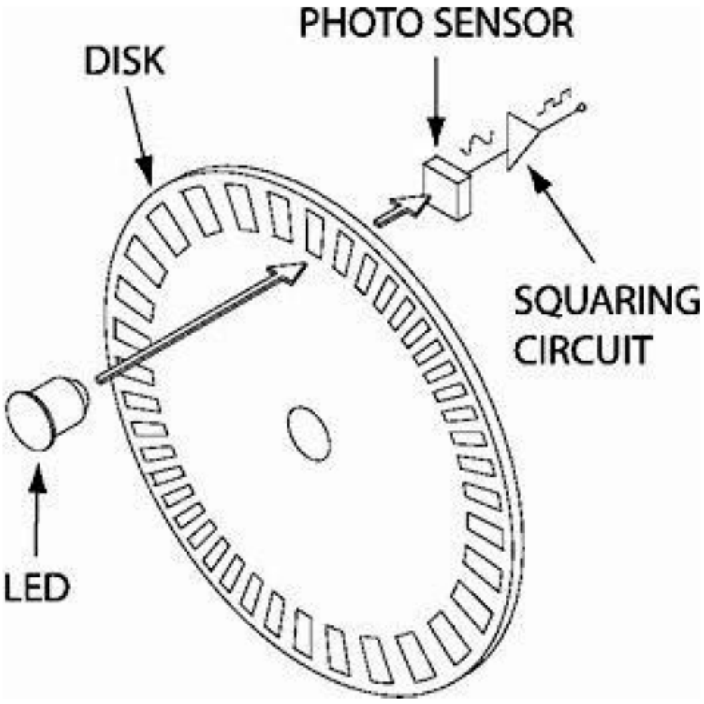


Figura 1.42: Disco para codificação binária sequencial

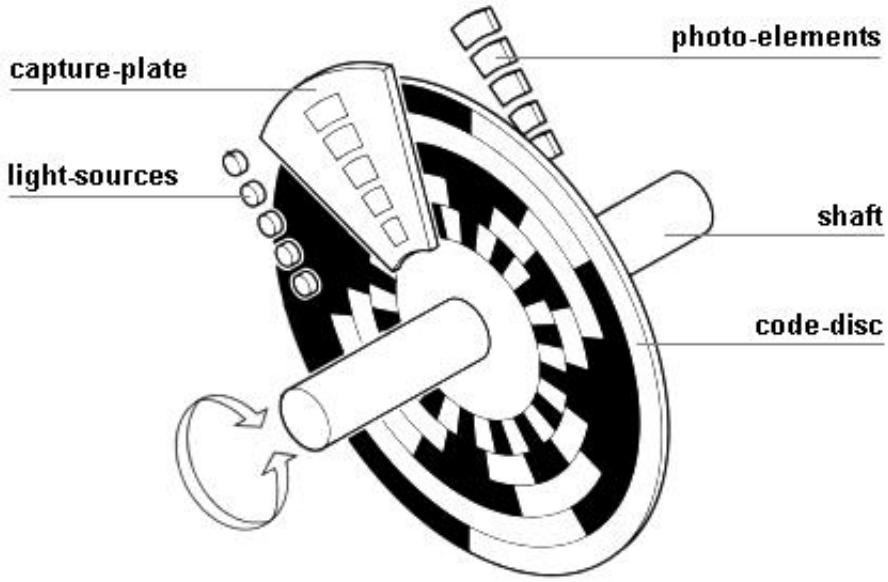


Figura 1.43: Disco com os sensores luminosos utilizados para detectar a angulação do motor, utilizando código Gray

Capítulo 2

Desenvolvimento

2.1 Materiais Utilizados

2.1.1 Arduino UNO

No projeto foi utilizado o Arduino UNO R3, Figura 2.1, que utiliza o microcontrolador ATmega328, este possui 14 pinos digitais de entrada e saída, 6 entradas analógicas, 3 pinos de referência, 1 pino de 5 v, 1 pino de 3,3 v, uma frequência de trabalho de 16 MHZ, memoria flash de 32 KB, SRAM de 2 KB e EEPROM de 1 KB.

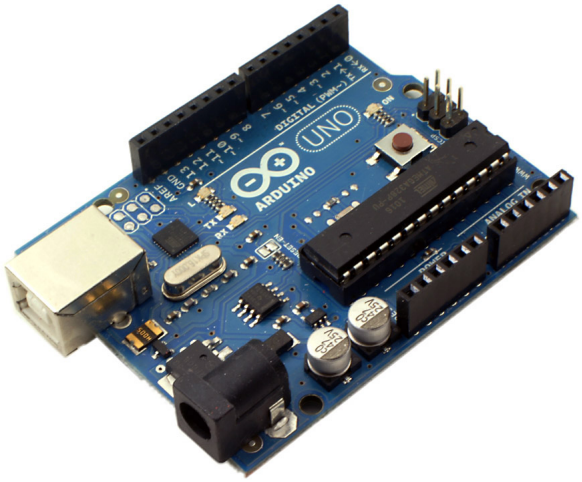


Figura 2.1: Arduino UNO

2.1.2 Bateria Solar

Como fonte de captação e armazenamento de energia elétrica foi utilizada um carregador de celular e tablet, Figura 2.2, com painel solar de 0,7 Watts, com capacidade de armazena-

mento aproximado de 3,84 Ah a 5 v, duas saídas USB de 5 v DC e uma entrada para recarregar.

Figura 2.2: Imagem da bateria utilizada

A descrição da bateria informa que a mesma possui 30 Ah a 5 V, para confirmar essa característica foi realizado um teste de verificação. Para o teste foi utilizado um Arduino UNO, a bateria, um recipiente de proteção e um cabo USB.

Figura 2.3: Teste para verificar a capacidade real da bateria

O teste consiste em utilizar a bateria com um equipamento que tenha a energia/hora consumida bem definida, e dessa forma podemos utilizar a fórmula abaixo para verificar a capacidade real:

Figura 2.4: Teste para verificar a capacidade real da bateria

$$CapacidadeDaBateria = ConsumoPorHora * HoraDeFuncionamento$$

Ou seja, multiplicamos a carga utilizada por hora, pela quantidade de horas que a bateria conseguiu alimentar o dispositivo. Em nosso teste foi conectado um Arduino UNO a bateria através de um cabo USB, foi verificado através de um multímetro que o dispositivo estava consumindo aproximadamente 40 mAh. O circuito foi armazenado em um recipiente de poliéster devidamente lacrado e protegido da luz solar, Figura 2.3 e 2.4, nessas condições a bateria manteve o circuito em funcionamento durante aproximadamente 4 dias, e utilizando a fórmula descrito acima temos que a capacidade a aproximada da bateria é de 3,84 Ah, muito abaixo do especificado na descrição.

Porém essa diferença de potência, não implica em mudanças no projeto original.

2.1.3 Sensor Ultrassônico HC-SR04

O sensor ultrassônico HC-SR04, Figura 2.5, é utilizado para detectar objetos e medir distâncias entre estes objetos e o sensor, podendo medir distâncias de 2cm a 4m, com uma precisão de 3 milímetros. O módulo possui um transmissor ultrasônico, um receptor e um circuito de controle [17].

Possui quatro pinos: entrada de energia de 5 v, GND, pino para disparar o envio do sinal e pino de interrupção, que apresenta nível alto quando o sinal é recebido.

2.1.4 Motores

Foram utilizados 2 motores DC com caixa de redução e rodas de plastico e borracha, Figura 2.6.

Cada motor funciona de 3 v-6 v, em nosso projeto estamos utilizando os motores com 6 v para obter um maior torque na movimentação. Nessa configuração temos as segintes

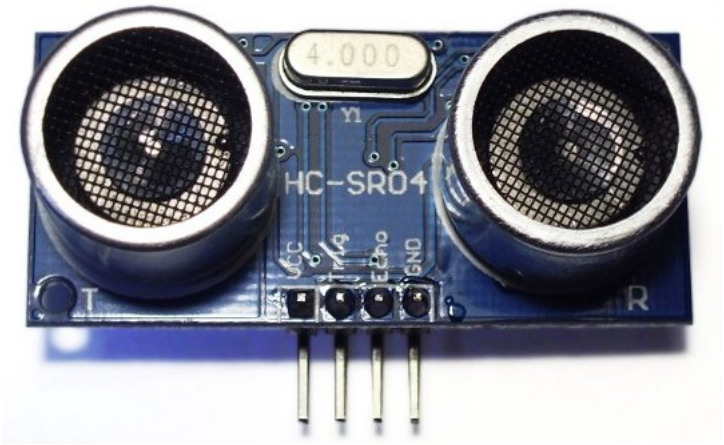


Figura 2.5: Sensor HC-SR04

Figura 2.6: Par de motor com caixa de redução e rodas

caracteriśticas: 230 RPM, corrente de 150 mA, distância pecorrida de 47,7 m por minuto e torque de 1,1 Kg/cm.

2.1.5 Conversor Dc Dc Step Up Xl6009

Esse componente, Figura 2.7, foi utilizado para elevar a tensão que alimenta os motores. Possui faixa de tensão de entrada de 3 v–32 v e faixade tensão de saída de 5 v–35 v, corrente máxima de entrada de 4 Ah e uma eficiência de conversão de até 94%. O valor da tensão de saída pode ser regulado por um *trimmer* localizado na placa.

Figura 2.7: Módulo Step Up Xl6009

2.1.6 CI L293D

Esse CI, Figura 2.8, implementa duas pontes H, pode controlar motores de até 36 v com corrente constante de 600 mAh ou corrente de pico em torno de 1,2 Ah. Para controlar o Motor 1, utiliza-se os pinos 2 e 7. Para o Motor 2, os pinos 15 e 10 como *INPUT*. Quando se aplica 5 v a um input, o motor correspondente gira pra um lado, colocando 5 v no outro, o motor gira do lado inverso. Os pinos 3 e 6 servem para alimentar o Motor 1 e os pinos 14 e 11, o Motor 2. Segue mais detalhes na Figura 2.9 e na Tabela 2.1.

Figura 2.8: L293D

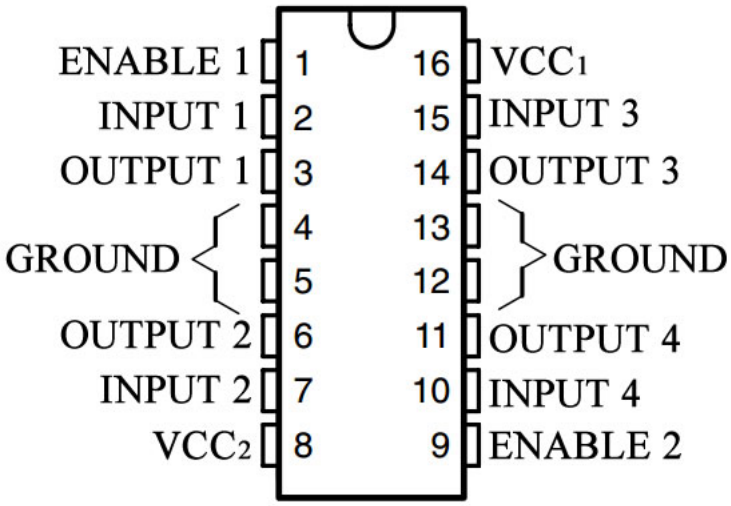


Figura 2.9: Pinagem retirada do datasheet do L293D [18]

Pino	Descrição
Enable 1	Ativa os pinos Input e Output 1 e 2
Input 1 e 2	Controla o sentido do Motor 1
Output 1 e 2	Saída, onde os terminais do Motor 1 são conectados.
Ground	Referência
VCC1	Alimentação do CI, 5 v
VCC2	Alimentação dos motores, suporta até 36 v
Enable 2	Ativa os pinos Input e Output 3 e 4
Input 3 e 4	Controla o sentido do Motor 2
Output 3 e 4	Saída, onde os terminais do Motor 2 são conectados.

Tabela 2.1: Descriçao dos pinos do CI L293D [18]

2.1.7 Demais materiais utilizados

- 5 (cinco) LDRs;
- 5 (cinco) resistores de 10 k e 1/4 w de potência;
- Placa de programação de 830 pontos;
- Mini roldana giratória 360º;
- Resina e catalisador de poliester;
- Fibra de vidro;
- Furadeira;

Os materiais utilizados para o desenvolvimento do projeto foram adquiridos fora do brasil, através do site www.aliexpress.com e www.ebay.com. Todos os produtos foram entregues como planejado, porém, devido a deficiência de trabalho da Receita Federal nas Unidades de Tratamento Internacionais, as encomendas ficaram 3 meses aguardando a triagem realizada por esta autarquia. Foi gasto aproximadamente R$ 250,00 e o tempo para desenvolver o projeto foi de 5 meses.

2.2 Arquitetura do Robô

O desenvolvimento de um sistema robótico é bastante complexo, dessa forma, se faz necessário a criação de um conceito de alto nível para discriminar a organização e a estrutura dos componentes utilizados no seu desenvolvimento. A arquitetura utilizada para o desenvolvimento do projeto proposto por este trabalho está representada na Figura 2.10

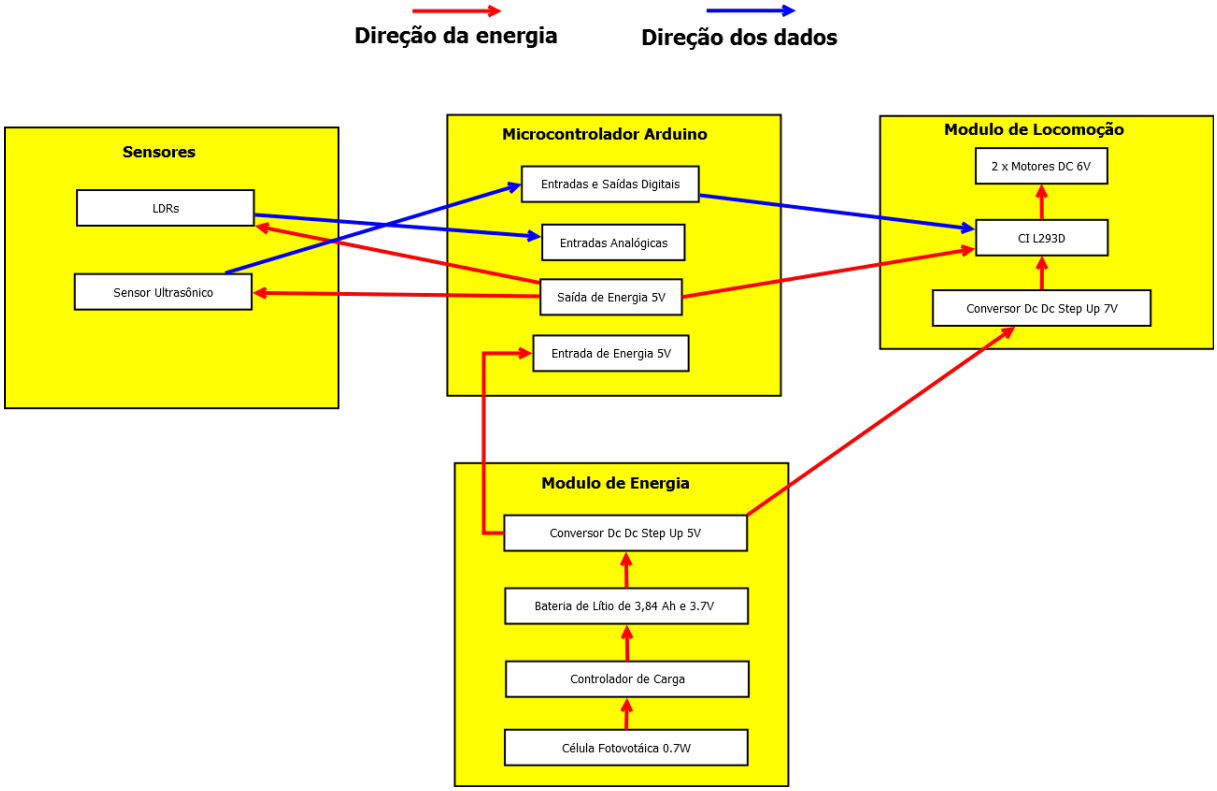


Figura 2.10: Estrutura dos componentes utilizados para desenvolver do sistema robótico

2.2.1 Microcontrolador

Descrição Módulo responsável por todo comportamento do robô.

O Arduino Uno foi a plataforma escolhida para o desenvolvimento do projeto, os motivos para sua utilização foram: custo acessível, quantidade de portas I/O compatível com

o projeto, vários sensores disponíveis no mercado e a grande quantidade de informações e manuais disponíveis na internet.
O Arduino é responsável por processar todas as informações dos sensores de luminosidade e obstáculo, além de enviar comandos para o módulo de locomoção, é nele que está implementado o algoritmo responsável pela busca dos pontos luminosos, com o objetivo de recarregar as baterias. Ele está ligado diretamente a todos os módulos do sistema, coordenando e gerando respostas para os estímulos gerados pelos sensores.
As portas I/O digitais são utilizadas para controlar os motores e sensor Ultrassônico. As portas analógicas fazem a leitura dos LDRs, e a saída de 5 v é utilizada para alimentar o CI L293D, sensor Ultrassônica e os LDRs.

2.2.2 Módulo de Locomoção

Descrição Módulo responsável pelo deslocamento.

O módulo de locomoção, como o próprio nome diz, é responsável pela locomoção do robô. Ele é formado por 3 (três) componentes, o primeiro é o conjunto de motores de corrente contínua de 6 v, esses possuem caixas de redução, com a finalidade de aumentar o seu torque, e dessa forma, possibilitar a força necessária para deslocar o robô até o destino desejado.

O segundo componente utilizado é o Circuito Integrado L293D, esse circuito possue duas funções, a primeira é o fornecimento de corrente necessária para o funcionamento dos motores, visto que as portas do Arduino Uno trabalham com baixa corrente, em média 50 mAh, e cada motor utilizado necessita de 150 mAh. A segunda função é chavear o sentido de rotação dos motores e dessa forma definir a direção que robô irá seguir. Para realizar o controle do fluxo da corrente o CI L293D implementa duas pontes H, Figura 2.11, circuito muito utilizado na robótica.

O terceiro componente, Conversor DC DC Step Up, foi utilizado para corrigir algumas características da bateria e do CI L293D, o primeiro fornece tensão de 5 v, e o motor utilizado necessita de 6 v, e o segundo reduz a tensão em 0,7 v, devido as características dos diodos utilizados da contrução da ponte H, dessa forma foi necessário aumentar a tensão fornecida ao CI L293D de 5 v para 7 v. O Conversor possui um *trimpot* para regular essa tensão de saída.

2.2.3 Módulo de Energia

Descrição Módulo responsável pela alimentação do sistema e recarga das baterias.

O módulo de energia é responsável pelo fornecimento de energia a todo o sistema, através de sua célula fotovotáica ele converte os raios solares em energia elétrica e armazena essa energia em duas baterias de Lítio de 3,7 v.

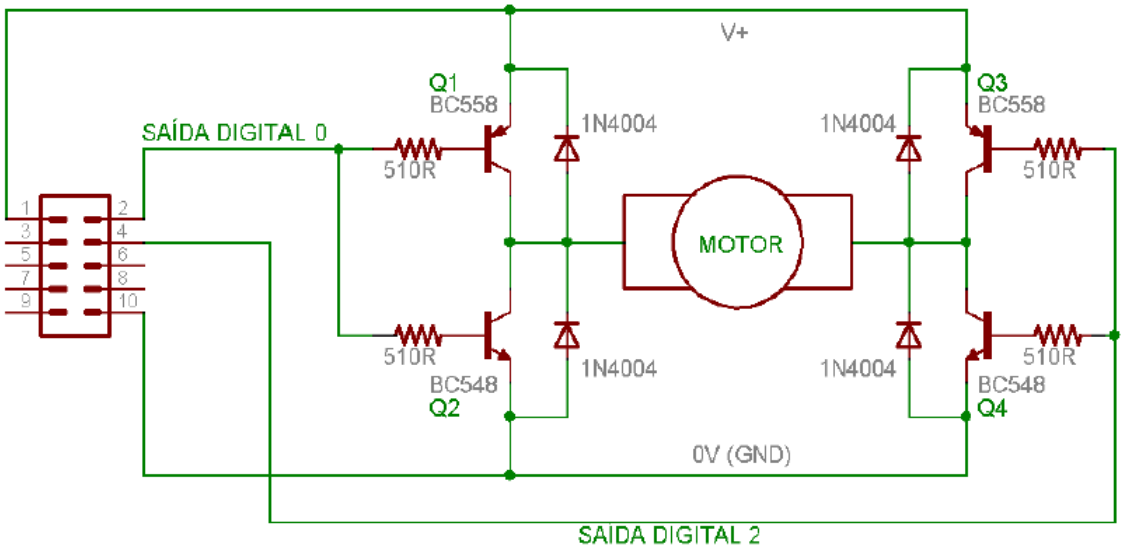


Figura 2.11: Exemplo de implementação de uma Ponte H, as portas 2 e 4 do CI controlam o sentido de rotação, e as portas 1 e 10 são os polos positívo e negativo, respectivamente, responsáveis pelo fornecimento de corrente.

Como o módulo de bateria utilizado originalmente é destinado para recarregar celulares, ele possue um conversor DC-DC que eleva os 3,7 v da bateria para 5 v. Esse conversor alimenta diretamente o microcontrolador Arduino e o conversor DC-DC do modulo de Locomoção.

2.2.4 Sensores

Descrição Módulo contém o sensor ultrassônico e os LDRs.

A fim de perceber o ambiente externo, nosso robô utiliza dois tipos de sensores: LDR e Sensor Ultrasônico de Distância. Foram utilizados 4 LDRs ao longo das direções norte, sul leste e oeste do chassis, além de um LDR próximo da célula solar, no módulo de energia, para verificar se o robô está na posição mais favorável para captar os raios solares. O sensor Ultrasônico foi instalado na parte frontal para verificar se existem barreiras que possam impedir o deslocamento.

2.3 Montagem do Robô

A primeira etapa na montagem do robô foi a fabricação de seu chassi. Para tal, foi utilizado uma folha de fibra de vidro, resina de laminação, catalizador, pincel, tiner, tesoura, caneta para marcação e molde de emborrachado, conforme Figura 2.12.

O formato do robô foi desenhado no Software FireWorks e impresso em uma folha de A4 comum, logo após, o desenho foi transferido para uma folha de emborrachado de espes-

Figura 2.12: Fabricação do chassis

sura média, recortado o emborrachado, estava pronto a fôrma para a fibra de vidro. Logo em seguida foram recortados três pedaços da folha de fibra de vidro no mesmo formato do emborrachado, com 30 30 cm de largura e 22 22 cm de comprimento.

A segunda etapa foi a preparação da resina de laminação, para isto, foi utilizado uma vasilha pequena, um bastão de madeira, uma seringa de 20 ml e outra de 3 ml. Para evitar a cura prematura da resina, a mistura foi preparada em duas vezes. Em cada etapa foi adicionado 100 ml de resina de laminação e 1 ml de catalizador, após 1 ou 2 minutos de mistura a produto está pronto para aplicação.

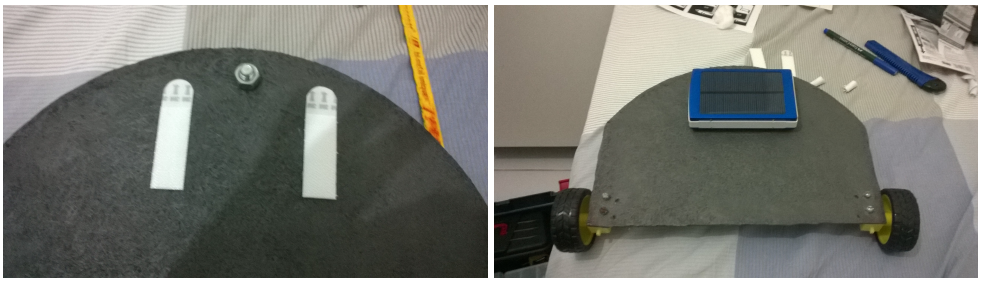
A primeira camada de resina foi aplicada por cima da fôrma de emborrachado, logo após foram aplicadas três vezes as camadas de fibra de vidro e resina, alternadamente. Após o período de cura da resina, 24 h, foi realizado o acabamento das bordas e os furos para fixação da roldana e dos motores, Figura 2.13a.

Para melhorar o acabamento, a plataforma foi pintada com tinta a óleo para portão na cor prata, em seguida os motores e a roldana foram fixados, Figura 2.13b.

(a) Chassi depois do acabamento (b) Chassi com roldana e motores

Figura 2.13: Acabamento final.

Após a construção do chassi, o Módulo de Energia foi fixado, para tanto, foi utilizado um jogo de fixador de quadro de parede da marca 3M, esse fixador permite a fácil remoção e recolocação do módulo. O processo de fixação do Módulo de Energia está registrado nas Figuras 2.14a à 2.14b.

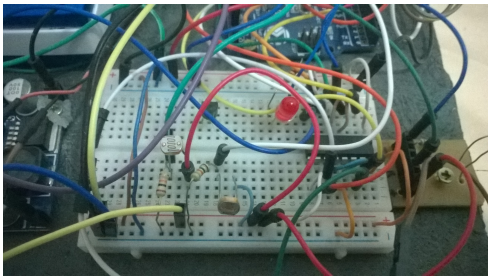


(a) Fixador aplicado no chassi (b) Fixação finalizada

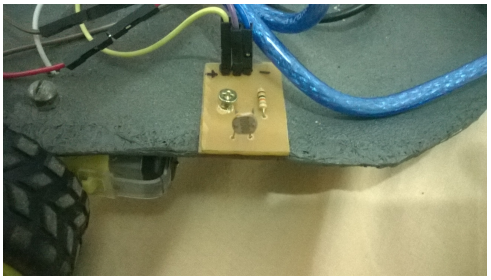
Figura 2.14: Fixação do módulo de Energia.

Para possibilitar uma fácil alteração do circuito, foi utilizada uma placa de prototipação com 480 pinos de 83x55 mm, a *protoboard* foi fixada com fita duplaface de silicone.

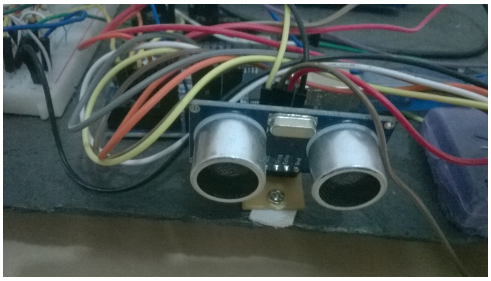
Após todos os componentes do robô serem instalados, Figura 2.15a à 2.15g, começou a fase de instalação do *software* de controle. Para o envio do código fonte foi utilizada a interface do arduino e sua IDE instalada em um notebook.



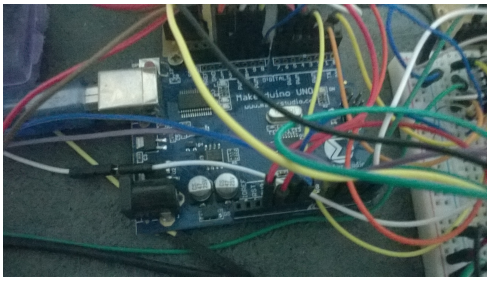
(a) Placa de prototipação com o circuito



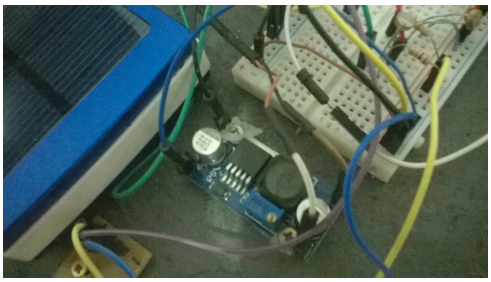
(b) LDR instalado no chassis



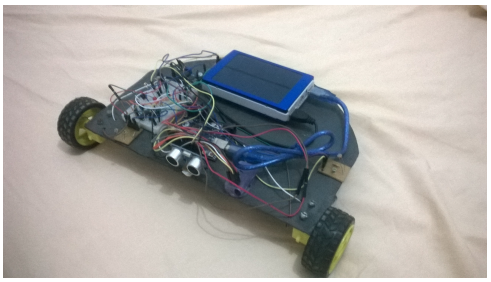
(c) Sensor ultrassônico instalado



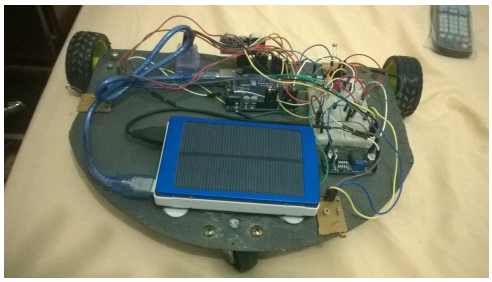
(d) Arduino instalado no chassis



(e) Módulo Step Up instalado no chassis



(f) Motagem do robo completa - Frente



(g) Motagem do robo completa - Traz

Figura 2.15: Monstagem do robô

2.4 Desvio de Obstáculos

Para que nosso robô consiga alcançar as áreas com maior luminosidade, ele necessita desviar de possíveis obstáculos que podem aparecer no caminho, para isso, foi desenvolvido um algoritmo de desvio de obstáculos, que é executado sempre que o robô se desloca, conforme algoritmo 2.16.

Esse algoritmo verifica se existe algum objeto sólido a uma distância entre 44 cm e 22 cm, em caso positivo, o robô considera seguro para tentar desviar e contornar tal objeto, porém,

se o valor for inferior a 22 cm, devido as dimenções deste, não é seguro executar o desvio, dessa forma, é realizado o movimento de ré e reexecutado o algorítimo de desvio.

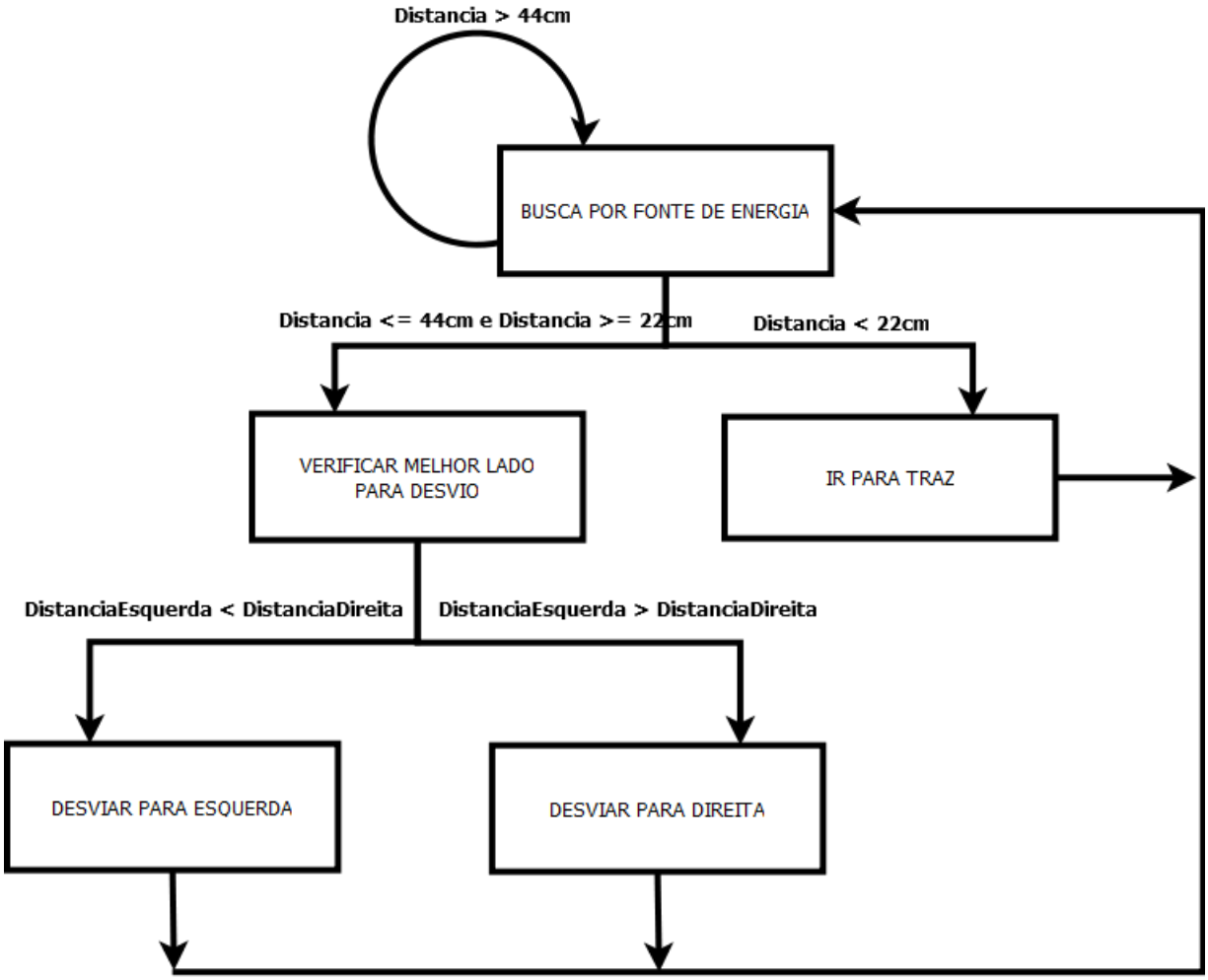


Figura 2.16: Algoritmo de desvio

2.5 Busca por Fonte de Energia

O objetivo principal do projeto, é desenvolver uma máquina que seja capaz de restabelecer a carga de suas baterias através de energia solar, para isso, o robô precisa executar tarefas que lhe permita encontrar regiões com insidência direta de raios solares. Foram utilizados 5 LDRs, distribuídos conforme Figura 2.17, para verificar a direção mais propícia para encontrar tais regiões.

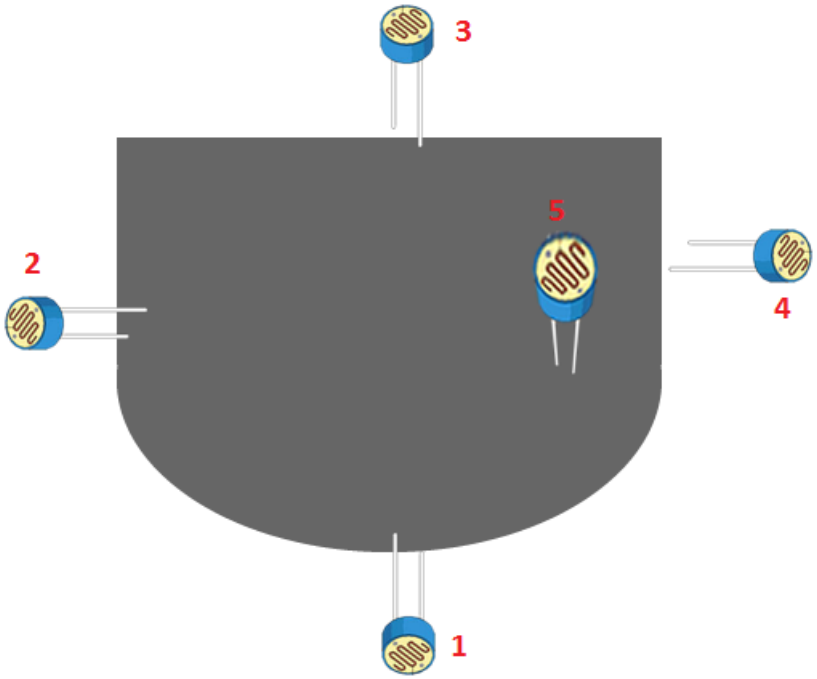


Figura 2.17: Distribuição dos LDRs no chassi do robô

Pra verificar a direção ideal, o Algoritimo 2.18 verifica qual LDR possue maior valor, após essa verificação, o robo é direcionado para o lado com maior luminosidade, anda 1 passo para frente e executa novamente o processo. Porém, antes de tudo, o algoritimo verifica se o LDR 5 possue o maior valor, em caso positivo, nenhum procedimento anterior é executado, pois significa que posição atual é a melhor.

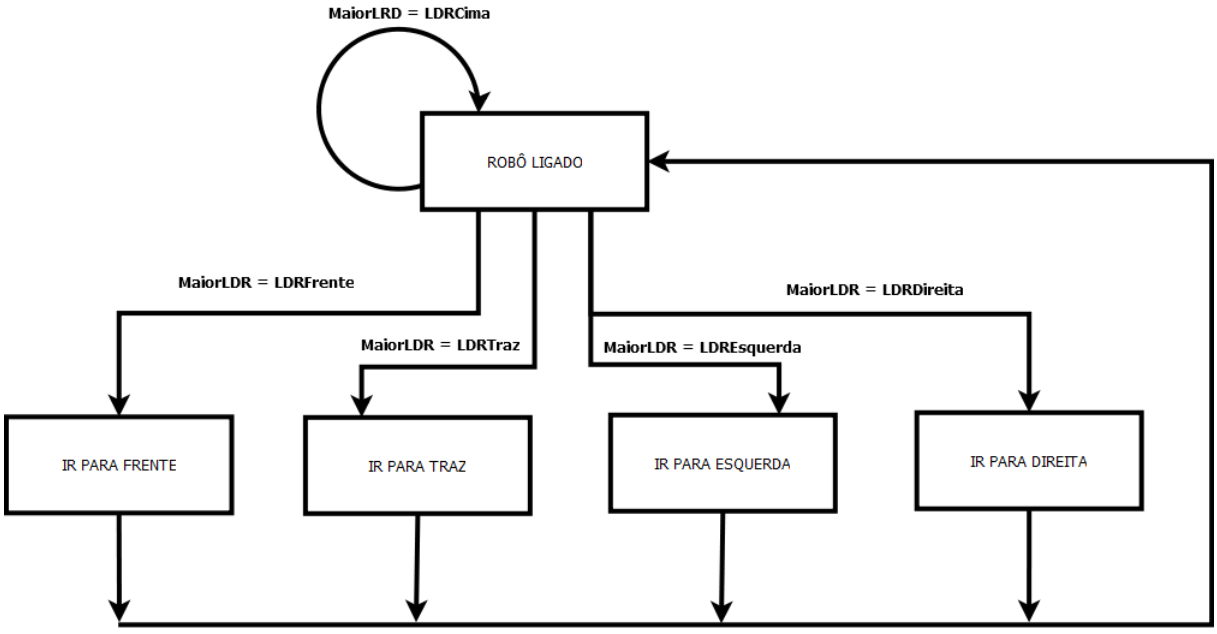


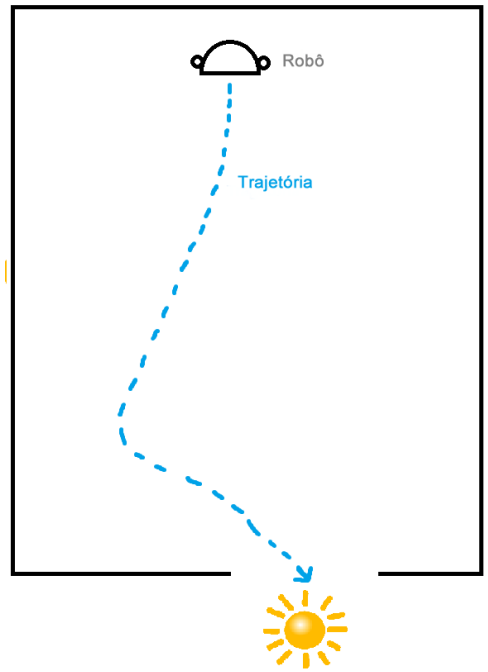
Figura 2.18: Algoritmo de Busca por Fonte de Energia

2.6 Testes Relizados e Resultados

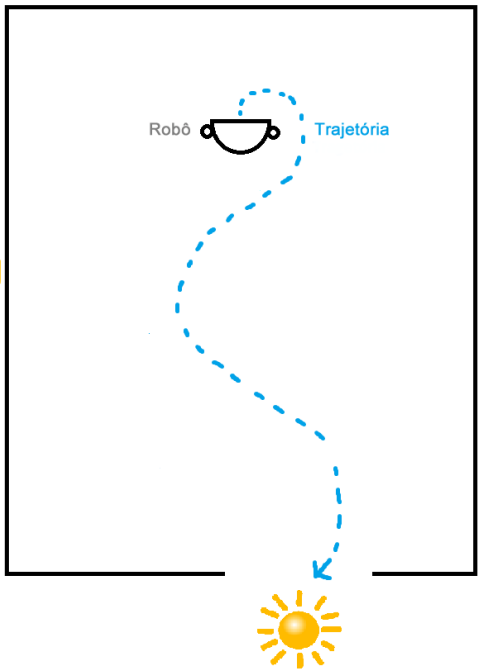
Para verificar a viabilidade do algoritmo utilizado, foram realizados 5 testes, todos em ambiente pré-configurado, piso plano, sem buracos, pedras ou outros objetos pequenos que possam atrapalhar o deslocamento do robô. O resultado apresentado, Figura 2.19a a 2.20b, é uma média das trajetórias realizadas, em cada configuração o robô foi submetido a 15 tentativas, apresentando os resultados descritos na tabela 2.2.

Teste	Descrição	Acertos	Tempo médio	Problemas encontrados
Cenário 1	Cenário mais simples, para verificar se o robô consegue alcançar seu objetivo em um ambiente sem obstáculos e já direcionado.	100%	57 seg	nenhum
Cenário 2	Para verificar se o robô consegue alcançar seu objetivo em um cenário sem obstáculos, porém com direcionamento oposto ao da fonte luminosa .	100%	90 seg	nenhum
Cenário 3	Para verificar se o robô consegue alcançar seu objetivo em um cenário sem obstáculos, porém com a fonte luminosa direcionada do lado direito .	100%	98 seg	nenhum
Cenário 4	Para verificar se o robô consegue alcançar seu objetivo em um cenário com 1 (um) obstáculo .	75%	141 seg	Problema com a detecção de obstáculo, sensor ultrassônico não consegue identificar o obstáculo do cenário, e o robô fica preso.Isso ocorre devido as características do sensor ultrassônico, dependendo do ângulo em que o som incide sobre a superfície do obstáculo, esse não consegue retornar ao sensor.
Cenário 5	Para verificar se o robô consegue alcançar seu objetivo em um cenário com 2 (dois) obstáculo .	75%	141 seg	Problema com a detecção de obstáculo, sensor ultrassônico não consegue identificar o obstáculo, e o robô fica preso

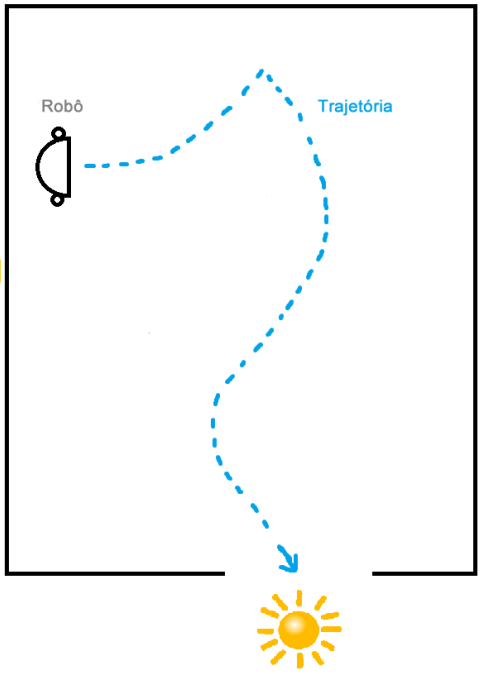
Tabela 2.2: Resultados obtidos



(a) Primeiro cenário.



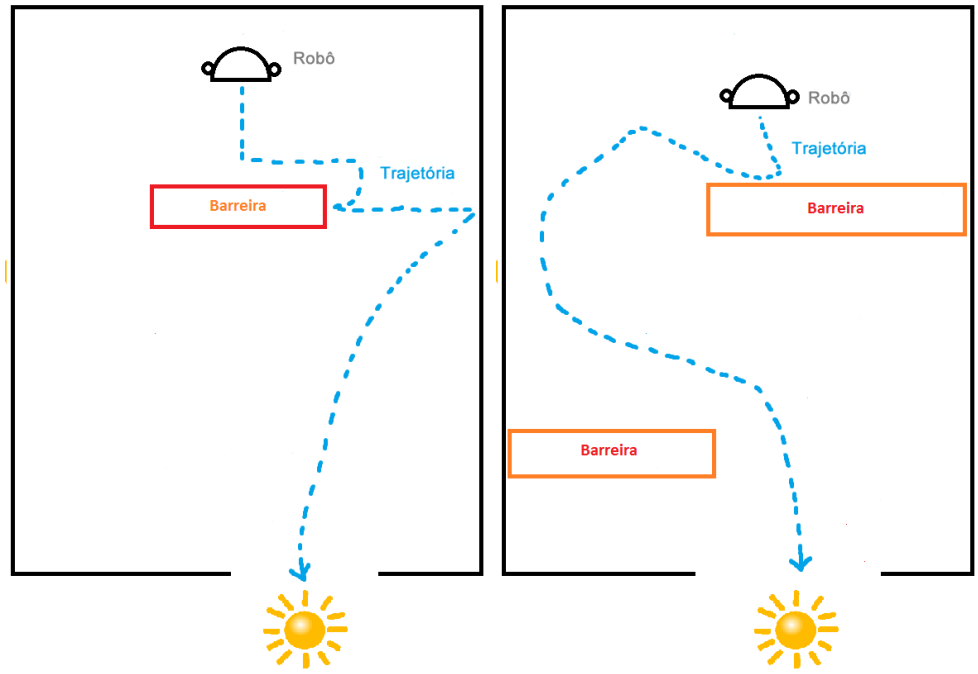
(b) Segundo cenário.



(c) Terceiro cenário.

2.7 Problemas Encontrados

Ao realizar os testes foi verificado que a estrutura do robô, motores e roldana, não possibilitam o deslocamento em terrenos acidentados, com pedregulhos e pequenos obstáculos, para resolver esse problema poderíamos substituir os motores de 6 v por motores de 12 v, além de aumentar o diâmetro dos pneus, e dessa forma conseguir um maior torque . Além



(a) Quarto cenário. (b) Quinto cenário.

disso, o sensor ultrassônico não foi eficiente para verificar obstáculos pequenos, como também não possibilita mensurar o tamanho do obstáculo, dado que seria de suma importância para um cálculo preciso da trajetória do desvio. Como melhoria, poderíamos utilizar um sensor de distância a laser, que fornece dados pontuais, acoplado a um motor de passo, e dessa forma realizar uma varredura em todo o ambiente, como exemplificado na Figura 2.21.

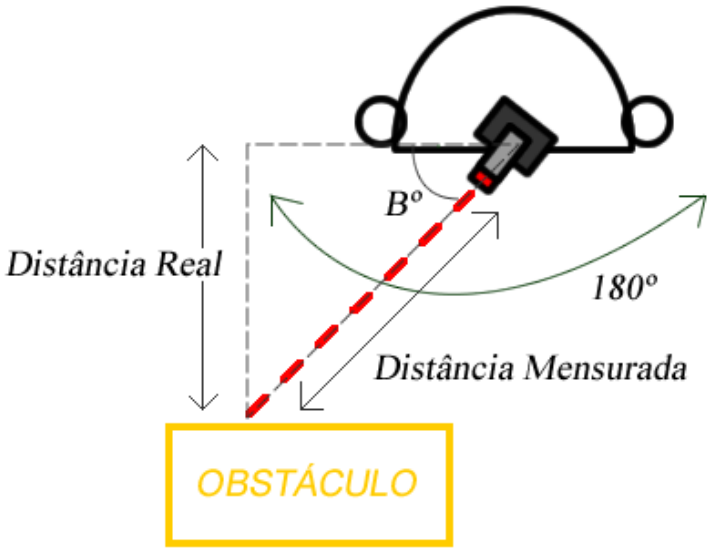


Figura 2.21: Varredura a laser

Os sensores de luminosidade não distinguem luz solar de luz artificial gerada por uma

lâmpada comum, dessa forma o robô irá gastar energia excessiva a noite procurando o melhor lugar para recarregar suas baterias, porém a célula solar não iria fornecer nenhuma carga. Para evitar isso, podemos adicionar um módulo relógio ao *Arduino*, com isso, o robô teria a informação se está de noite ou de dia, evitando gasto desnecessário de energia.
Por algum motivo desconhecido, ocasionalmente, um dos motores para de funcionar e pouco tempo depois é reativado, foi realizado vários testes com configurações de *hardware* e *software* diferentes, porém a solução para esse mau funcionamento esporádico não foi encontrada, acreditamos que o CI de ponte H utilizado possui um mau funcionamento quando as direções dos motores são alteradas rapidamente durante algum tempo.
Devido a ausência de ferramentas adequadas para a produção do chassis do robô, este não encontra-se alinhado, dessa forma, ao se deslocar, o robô apresenta um pequeno desvio para direita, fato que não interfere na busca realizada.
Como o robô não possui giroscópio e acelerômetro instalados, impossibilitado pela limitação de portas analógicas do Arduino UNO, este não tem meios para definir a velocidade e a direção de sua trajetória, dessa forma, quando ocorre algum imprevisto em relação ao direcionamento direita/esquerda, este é desviado da trajetória desejada, importuno que será corrigida após alguns passos, pelos LDRs e sensor Ultrassônico.

2.8 Cógidos Utilizados

Todo código foi desenvolvido na IDE 1.0.5-r2 do *Arduino* e possui o total de 363 linhas. O mesmo está anexo a este trabalho

Capítulo 3

Conclusão

Analisando os testes realizados, verificamos que o objetivo de criar um robô para se auto carregar, procurando luz solar, foi alcançado, porém, o mesmo necessita de um ambiente propício para sua locomoção. Essa circunstância possibilita novos trabalhos, cujo foco seria atualizar a estrutura do robô, permitindo o deslocamento do mesmo em ambientes acidentados e de difícil acesso.

Outra modificação necessária, seria a instalação de um Port Expander Shield para estender o número de portas do Arduino Uno, e dessa forma, possibilitar a instalação de um acelerômetro, giroscópio, GPS e bussola, com esses sensores o robô teria dados para conhecer seu posicionamento no ambiente e dessa forma executar algoritmos de busca mais eficientes. Juntamente com esses sensores poderíamos implementar métodos para mapeamento de ambiente, e assim, armazenar a trajetória para os melhores locais com luz solar.

O princípio deste trabalho pode ser utilizado em diversas áreas da tecnologia, possibilitando que equipamentos de transmissão de dados remotos, entre outros, obtenham um maior tempo de funcionamento ou até sua autonomia total, e dessa forma contribuindo para o desenvolvimento da tecnologia em geral.

Bibliografia

[1] Arduino products. http://www.arduino.cc/en/Main/Products. Accessed: 30-01-2015.

[2] Asimo. http://world.honda.com/ASIMO/history/e0/index.html. Accessed: 06-02-2015.

[3] Atmel site. http://www.atmel.com/pt/br/. Accessed: 08-02-2015.

[4] Hexapod with arduino. http://www.instructables.com/id/DIY-handmade-Hexapod-with-arduino-Hexdrake/?lang=pt. Accessed: 16-02-2015.

[5] Microchip products. http://www.microchip.com/pagehandler/en-us/products/home.html. Accessed: 30-01-2015.

[6] Os 10 países mais robotizados do mundo. http://www.exame.abril.com.br/economia/noticias/os-10-paises-mais-robotizados-do-mundo. Accessed: 28-01-2015.

[7] Raspberry pi products. http://www.raspberrypi.org/products/. Accessed: 30-01-2015.

[8] Sony aibo. http://www.sony-aibo.co.uk. Accessed: 06-02-2015.

[9] Tms370 mcus. http://www.ti.com/mcu/docs/mcucodeexnewsltrbuglist.tsp?sectionId=98&tabId=1516. Accessed: 30-01-2015.

[10] Tug robot. http://www.aethon.com/tug/benefits/. Accessed: 25-01-2015.

[11] M. Banzi. *Primeiros Passos com Arduino*. Novatec, 1º edition, 2011.

[12] N. Braga. Arduino introdução e recursos avançados. http://www.newtoncbraga.com.br/index.php/como-funciona/6097-art764, 2014. [Online; accessed 10-Octuber-2014].

[13] I. Corporation. *MCS 51 MICROCONTROLLER FAMILY USER'S MANUAL*. Intel Corporation, http://www.web.mit.edu/6.115/www/document/8051.pdf, 1994.

[14] M. R. de Albuquerque Barros. Estudo da automação de células de manufatura para monstagens e soldagens industriais de carrocerias automotivas. dissertation, Escola Politécnica da Universidade de São Paulo, 2006.

[15] M. e Castrucci. *Engenharia de Automação Industrial*. Editora LTC, 1º edition, 2007.

[16] D. Eberhardt. Desenvolvimento de um sistema completo para caracterização de células solares. dissertation, PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL, 2005.

[17] E. Freaks. *Datasheet - Ultrasonic Ranging Module HC - SR04*. Elec Freaks, http://www.micropik.com/PDF/HCSR04.pdf.

[18] T. Instruments. *DataSheet L293D*. Texas Instruments, http://www.ti.com/lit/ds/symlink/l293d.pdf, 1986.

[19] D. E. D. S. James Skinovsky, Maurício Chibata. Realidade virtual e robótica em cirurgia - aonde chegamos e para onde vamos? *Revista do Colégio Brasileiro de Cirurgiões*, pages 334–337, 2008.

[20] J. B. J.S. Agnaldo. Celulas solares de tio 2 sensibilizado por corante. *Revista Brasileira de Ensino de Física*, pages 77–84, 2006.

[21] G. M. Martins. *Princípios de Automação Industrial*. Universidade Federal de Santa Maria, 1º edition, 2007.

[22] M. J. Mataric. *The Robotics Primer*. Massachusetts Institute of Technology, 1° edition, 2007.

[23] M. McRoberts. *Arduino Básico*. Novatec, 1º edition, 2011.

[24] P. Musumeci. *68HC11 Programmer's Reference Manual*. Departamento de Engenharia Elétrica UFRJ, http://www.dee.ufrj.br/microproc/HC11/68hc11ur.pdf, 1999.

[25] L. F. Patsko. *Aplicações, Funcionamento e Utilização de Sensores*. Maxwell Bohr Instrumentação Elétrônica, http://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_-_aplicacoes_e_funcionamento_de_sensores.pdf, 2006.

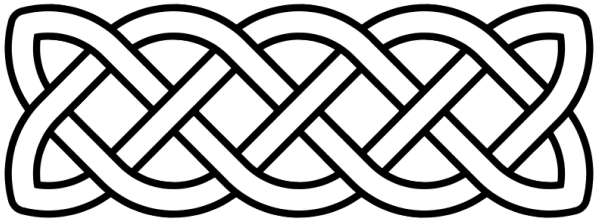
[26] N. P. Santos. *Arduino Introdução e Recursos Avançados*. Escola Naval, Departamento de Engenheiros Navais Ramo de Armas e Eletrônica, http://www.isegi.unl.pt/docentes/vlobo/escola_naval/MFC/Tutorial%20Arduino.pdf, 2009.

[27] B. E. B. Theodore L. Brown, H. Eugene Lemay. *Química - A Ciência Central*. Pearson, 9º edition, 2005.

[28] C. Vogel. *Build Your Own Electric Motocycle*. McGraw-Hill, 1º edition, 2009.

[29] J.-D. Warren. *Arduino Robotics*. Technology in Action, 1º edition, 2011.

Este trabalho foi redigido em LaTeX utilizando uma modifição do estilo IC-UFAL. As referências bibliográficas foram preparadas no JabRef e administradas pelo BibTeX com o estilo LaCCAN. O texto utiliza fonte Fourier-GUT*enberg* e os elementos matemáticos a família tipográfica Euler Virtual Math, ambas em corpo de 12 pontos. A numeração dos capítulos segue com a família tipográfica Art Nouveau Caps.



Anexo

Código utilizado.

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"
#include "printf.h"

//Configuracao do Sensor Ultrasonico
#include "Ultrasonic.h"
Ultrasonic ultrasonic(7,6); //Definindo os pinos do sensor ultrasonico

//
// Hardware configuration
//
int pinAntena1 = 8;
int pinAntena2 = A5;
//Variaveis Auxiliares utilizadas em Loop
int i;
int aux;
int flagDesviarDireita = 0; // indica se o robo ja fez uma tentativa para desviar para direita;
int flagDesviarEsquerda = 0; // indica se o robo ja fez uma tentativa para desviar para esquerda;

//Configuracao dos pinos dos motores
int motor1Pin1 = 2;
int motor1Pin2 = 3;
int motor2Pin1 = 4;
int motor2Pin2 = 5;
int passo = 800; // Tempo que o motor fica ativo em cada passo do robo.

//Configuracao dos LDRs
int ldrFrente = A2;
int ldrTraz = A1;
int ldrEsquerda = A0;
int ldrDireita = A3;
int ldrCima = A4;
int listaValorLdr[5]; // Utilizado para o calculo de maior valor
int listaValorDistancia[3];
```

```
36
37 int valorLdrFrente;
38 int valorLdrTraz;
39 int valorLdrEsquerda;
40 int valorLdrDireita;
41 int valorLdrCima;
42
43
44 // Configurando o radio no nRF24L01 nos pinos 9 e 10
45
46 RF24 radio(9,10);
47
48 //Definindo o endereco dos dois "nos" de comunicacao
49 const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL };
50
51 typedef enum { role_ping_out = 1, role_pong_back } role_e;
52
53 role_e role = role_pong_back;
54
55 void setup(void)
56 {
57
58   Serial.begin(57600);
59   printf_begin();
60
61   //pinMode(ledAlert, OUTPUT);  // define o pino de alerta como saida
62
63   //
64   // Setup and configure rf radio
65   //
66
67   radio.begin();
68
69   // Almenta o tempo de espera entre as tentativas
70   radio.setRetries(15,15);
71
72   radio.openWritingPipe(pipes[1]);
73   radio.openReadingPipe(1,pipes[0]);
74
75   //
76   // Esperando conexao
77   //
78
79   radio.startListening();
80
81   //Configurando Pinos do Motor
82   pinMode(motor1Pin1, OUTPUT);
83   pinMode(motor1Pin2, OUTPUT);
84   pinMode(motor2Pin1, OUTPUT);
85   pinMode(motor2Pin2, OUTPUT);
86
```

```
87    //Configurando pino da antena
88    pinMode(pinAntena1, INPUT);
89    pinMode(pinAntena2, INPUT);
90
91
92
93    radio.printDetails();
94    printf("Iniciando interface de comunicacao MARK-01\n");
95  }
96
97  void pararMotores(){
98
99    digitalWrite(motor1Pin1,LOW);
100   digitalWrite(motor1Pin2,LOW);
101   digitalWrite(motor2Pin1,LOW);
102   digitalWrite(motor2Pin2,LOW);
103   delay(500);
104
105 }
106
107 void traz(){
108
109    pararMotores();
110    digitalWrite(motor1Pin1, LOW); // set pin 2 on L293D low
111    digitalWrite(motor1Pin2, HIGH); // set pin 7 on L293D high
112    digitalWrite(motor2Pin1, LOW); // set pin 2 on L293D low
113    digitalWrite(motor2Pin2, HIGH); // set pin 7 on L293D high
114    delay(passo);
115    pararMotores();
116 }
117
118 void frente(){
119
120    pararMotores();
121    digitalWrite(motor1Pin1, HIGH); // set pin 2 on L293D low
122    digitalWrite(motor1Pin2, LOW); // set pin 7 on L293D high
123    digitalWrite(motor2Pin1, HIGH); // set pin 2 on L293D low
124    digitalWrite(motor2Pin2, LOW); // set pin 7 on L293D high
125    delay(passo);
126    pararMotores();
127 }
128
129 void esquerda(){
130    pararMotores();
131    digitalWrite(motor1Pin1, LOW); // set pin 2 on L293D low
132    digitalWrite(motor1Pin2, HIGH); // set pin 7 on L293D high
133    digitalWrite(motor2Pin1, HIGH); // set pin 2 on L293D low
134    digitalWrite(motor2Pin2, LOW); // set pin 7 on L293D high
135    delay(passo);
136    pararMotores();
137 }
```

```
138
139 void direita(){
140   pararMotores();
141    digitalWrite(motor1Pin1, HIGH); // set pin 2 on L293D low
142    digitalWrite(motor1Pin2, LOW); // set pin 7 on L293D high
143    digitalWrite(motor2Pin1, LOW); // set pin 2 on L293D low
144    digitalWrite(motor2Pin2, HIGH); // set pin 7 on L293D high
145    delay(passo);
146    pararMotores();
147 }
148
149 void loop(void)
150 {
151
152   //
153   // Recebe e responde os pacotes
154   //
155
156   if ( role == role_pong_back )
157   {
158     // se tiver dados para ler
159     if ( radio.available() )
160     {
161       printf("Dados disponiveis...\n");
162       // Pegando todos os dados disponiveis
163       char command;
164       bool done = false;
165       while (!done)
166       {
167         //Retorna os dados e verifica se e o ultimo
168         done = radio.available();
169         radio.read( &command, sizeof(char) );
170
171         printf("Comando Recebido %c...",command);
172
173         //Movimetando o robo de acordo com o comando recebido
174         if(command == 'F')
175           frente();
176         if(command == 'T')
177           traz();
178         if(command == 'E')
179           esquerda();
180         if(command == 'D')
181           direita();
182         delay(20);
183       }
184
185       //Para a de receber para que possamos responder
186       radio.stopListening();
187
188       // Respondendo com confirmacao do recebimento da mensagem
```

```
189       radio.write( &command, sizeof(char) );
190       printf("Enviando confirmacao de recebimento.\n\r");
191
192       // Voltando a receber os pacotes
193       radio.startListening();
194     }
195
196     //Caso nao tenha nenhum comando de locomocao o robo entra no
          algoritmo proposto para a busca de raios solares
197     else{
198       valorLdrFrente = analogRead(ldrFrente);
199       valorLdrTraz = analogRead(ldrTraz);
200       valorLdrEsquerda = analogRead(ldrEsquerda);
201       valorLdrDireita = analogRead(ldrDireita);
202       valorLdrCima = analogRead(ldrCima);
203
204       listaValorLdr[0] = valorLdrFrente;
205       listaValorLdr[1] = valorLdrTraz;
206       listaValorLdr[2] = valorLdrEsquerda;
207       listaValorLdr[3] = valorLdrDireita;
208       listaValorLdr[4] = valorLdrCima;
209
210       boolean ordenado = false;
211       printf("Entrou no Loop\n");
212       printf("Tamanho da Lista: %i\n",sizeof(listaValorLdr)/sizeof(
            int));
213       while(!ordenado){
214          ordenado = true;
215          for(i = 0; i < 4; i++){
216            printf("Valores: %i, %i, %i, %i, %i\n", listaValorLdr[0],
                 listaValorLdr[1],listaValorLdr[2], listaValorLdr[3],
                 listaValorLdr[4]);
217            if(listaValorLdr[i] > listaValorLdr[i+1]){
218                aux = listaValorLdr[i+1];
219                listaValorLdr[i+1] = listaValorLdr[i];
220                listaValorLdr[i] = aux;
221                ordenado = false;
222            }
223          }
224
225       }
226       printf("Saiu no Loop\n");
227       int maiorValor = listaValorLdr[4];
228       printf("Maior Valor encontrado: %i", maiorValor);
229
230       if(valorLdrFrente == maiorValor){
231         if(!desviar())
232           frente();
233       }
234       if(valorLdrTraz == maiorValor){
235          esquerda();
```

```
236             esquerda();
237             desviar();
238           }
239           if(valorLdrEsquerda == maiorValor){
240             esquerda();
241             desviar();
242           }
243           if(valorLdrDireita == maiorValor){
244             direita();
245             desviar();
246           }
247
248           delay(500);
249         }
250       }
251
252 }
253
254 boolean desviar(){
255
256  int distanciaFrente;
257
258  distanciaFrente = ultrasonic.Ranging(CM);
259
260  if(distanciaFrente <= 44 && distanciaFrente >= 22){
261
262    int distanciaDireita = 0;
263    int distanciaEsquerda = 0;
264
265    //Verificando qual lado tem a menor distancia
266    if(flagDesviarDireita == 0 && flagDesviarEsquerda ==0)
267    {
268      direita();
269      distanciaDireita = ultrasonic.Ranging(CM);
270      esquerda();
271      esquerda();
272      distanciaEsquerda = ultrasonic.Ranging(CM);
273      direita();
274
275      if(distanciaDireita > distanciaEsquerda){
276           flagDesviarEsquerda = 1;
277      }
278      else{
279          flagDesviarDireita = 1;
280      }
281    }
282
283     if(flagDesviarDireita == 0){
284       if(!desviarDireita()){
285         flagDesviarDireita = 1;
286       }
```

```
287
288       desviar();
289     }
290     else{
291       if(flagDesviarEsquerda == 0){
292         if(!desviarEsquerda()){
293           flagDesviarEsquerda = 1;
294         }
295
296         desviar();
297       }
298       else{
299         traz();
300         traz();
301         flagDesviarDireita=0;
302         flagDesviarEsquerda=0;
303         desviar();
304       }
305     }
306     return true;
307   }
308   else if(distanciaFrente < 22){
309     traz();
310     traz();
311     desviar();
312     return true;
313   }
314   else if(digitalRead(pinAntena1)){
315     traz();
316     traz();
317     desviarDireita();
318
319   }
320   else if(digitalRead(pinAntena2)){
321     traz();
322     traz();
323     desviarEsquerda();
324
325   }
326   else{
327         flagDesviarDireita=0;
328         flagDesviarEsquerda=0;
329         return false;
330   }
331 }
332
333 boolean desviarDireita(){
334
335    direita();
336    int distanciaFrente = ultrasonic.Ranging(CM);
337     if(distanciaFrente < 44){
```

```
338       esquerda();
339       return false;
340     }
341   frente();
342   frente();
343   esquerda();
344   return true;
345 }
346 
347 boolean desviarEsquerda(){
348 
349   esquerda();
350   int distanciaFrente = ultrasonic.Ranging(CM);
351   if(distanciaFrente < 44){
352     direita();
353     return false;
354   }
355   frente();
356   frente();
357   direita();
358   return true;
359 
360 }
361 
362 void ordenarLista(int *valores){
363 
364   int tamanhoLista = sizeof(valores)/sizeof(int);
365 
366   boolean ordenado = false;
367   while(!ordenado){
368      ordenado = true;
369      for(i = 0; i < tamanhoLista; i++){
370        if(valores[i] > valores[i+1]){
371            aux = valores[i+1];
372            valores[i+1] = valores[i];
373            valores[i] = aux;
374            ordenado = false;
375        }
376     }
377 
378   }
379 
380 }
```