

Universidade Federal do Pará

Instituto de Tecnologia

Faculdade de Engenharia da Computação e de Telecomunicações

EC01027 - Projetos de Engenharia III

Projeto *2012* de Desenvolvimento de Robô Autônomo para Torneio de Sumô: Relatório Final

Belém

2016



Universidade Federal do Pará

Instituto de Tecnologia

Faculdade de Engenharia da Computação e de Telecomunicações

EC01027 - Projetos de Engenharia III

Edson Bianor Ferreira Pinheiro

Edson da Silva Oliveira Júnior

Ingrid Caroline Aquino Pinto

Leandro Neves Rodrigues Santana

Otávio Augusto Alves Silva

Pedro Paulo Furtado Nazaré

Thiago Pina Soares

Projeto *2012* de Desenvolvimento de Robô Autônomo para Torneio de Sumô: Relatório II

Trabalho apresentado à disciplina EC01027 - Projetos de Engenharia III como requisito qualitativo final para conclusão da disciplina.

Prof. Dr. Marco José de Sousa

Belém

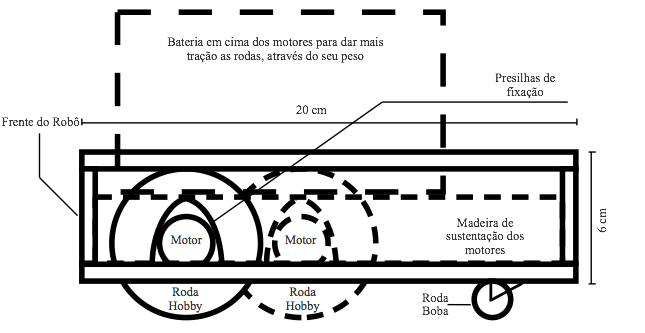
2016

**1. Introdução**

Para qualquer projeto - como a construção de um robô sumô - é necessário que exista uma pesquisa de requisitos, organização dos elementos e a análise de conjunto para o seu desenvolvimento. Para que haja sucesso, utilizam-se algumas metodologias iniciais, tais quais desenhos técnicos do objeto a ser criado, lista de materiais necessários para a construção, orçamentos dos materiais, construção e verificação dos circuitos, cálculos de autonomia da bateria e etc. Portanto, esse relatório tem como função ser um guia para o desenvolvimento e construção do robô sumô nomeado *2012*.

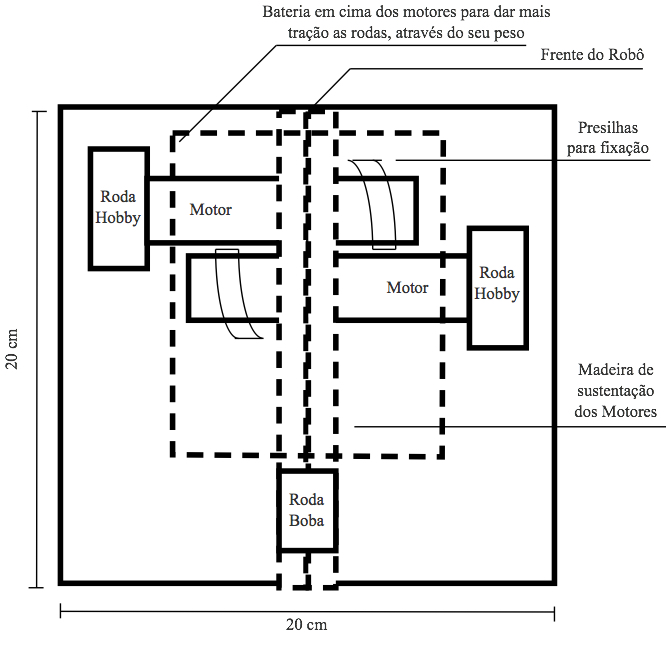
**2. Modelo do chassi**

A seguir pode-se visualizar a parte lateral e a parte superior do primeiro protótipo de chassi do robô.

**2.1. Visão lateral**

*Figura* 1. *- Esboço lateral do chassi*

**2.2. Visão superior**



*Figura* 2. *- Esboço superior do chassi*

**3. Lista de materiais**

A seguir pode-se visualizar a relação dos materiais escolhidos para a construção do robô, assim como as devidas justificativas para cada item.

* **Placa de PVC expandido -** Além de ser um material de fácil utilização, a equipe já possui uma quantidade expressiva do material.
* **Arduino Uno REV3** - Microcontrolador que será utilizado no projeto. A justificativa para sua escolha se deu por atender os requisitos necessários para o projeto e por grande parte da equipe projetista possuir o item e ter experiência em trabalhar com o mesmo.
* **Motor *JGB37-550*** - Motor de torque de *150 kg.cm* e com corrente de partida relativamente grande. A justificativa para a utilização do motor é a seu grande torque, fazendo com que robô possa ter uma força relativamente maior para empurrar seus adversários que estarão utilizando um motor de menor torque, como um servo motor.
  + Dimensões
    - 10 cm de comprimento.
    - 4 cm de diâmetro.
* **Módulo com relés** *Ywrobot 4 relay*,- Conforme apresentado em sala de aula, um modelo de ponte H usando relés é mais simples de construir, quando comparado com o modelo que utiliza MOSFETs na sua confecção, por exemplo. MOSFETs não possuem um preço tão acessível que favoreça uma boa relação custo-benefício para a construção de uma ponte H utilizando-se desse recurso. Para construir esse modelo seriam necessários pelo menos quatro unidades, o que aumentaria consideravelmente o orçamento. Foi decidido usar os módulos com relés, considerados mais vantajosos em comparação aos MOSFETs.
* **Regulador de Tensão LM7805** - O regulador de tensão será responsável pelo fornecimento de tensão ao módulo relé. A implementação desse regulador de tensão ocorre em virtude do módulo relé demandar uma quantidade muito grande de corrente, capaz de prejudicar o funcionamento do microcontrolador Arduino e até mesmo danificá-lo.
* **Sensor Ultrassônico -** O sensor ultrassônico será responsável pela detecção dos oponentes dentro do ring. O funcionamento desse sensor se dá por meio do envio de ondas ultrassônicas que, ao chocar com algum obstáculo, são refletidas e captadas pelo sensor. O sensor é facilmente encontrado nas lojas de
* **Sensor de Linha** - Os sensores de linha serão responsáveis pela identificação da demarcação do ringue, evitando a saída do nosso robô. A justificativa se dá por possuírem um custo acessível e sua implementação ser simples.
* **Baterias -** Como fonte de tensão será utilizada uma bateria de chumbo do mesmo tipo que se utiliza em no-breaks. A mesma foi escolhida por possuir um ótimo custo-benefício para o projeto, pois possui uma capacidade nominal de corrente (que será especificada mais a frente) a um bom preço e por ser uma bateria recarregável, o que permitirá a sua utilização mais de uma vez.
  + Dimensões
    - 15 cm de comprimento
    - 7 cm de largura
    - 10 cm de altura
* **Roda** *Hobby 65mm*- Rodas frontais onde serão implementados os motores para a movimentação do robô. As mesmas foram escolhidas por terem uma boa tração e também por já as possuirmos.
  + Dimensões
    - 3 cm de espessura.
    - 7 cm de diâmetro.
* **Roda boba** - Roda do tipo que é utilizada em móveis, fácil de encontrar em lojas de materiais de construção ou de móveis. Será utilizada na parte de trás do chassi. A escolha desta roda se deu em virtude de sua utilização dar um melhor equilíbrio ao robô, visto que será o seu único apoio na parte posterior, fazendo com que todas as rodas sempre mantenham contato com o chão.
  + Dimensões
    - 2 cm de espessura.
    - 4 cm de diâmetro.

**3. Orçamento preliminar**

A seguir pode-se visualizar o orçamento preliminar necessário para a construção do robô de sumo. A quantidade dos materiais, preço e loja podem variar com o decorrer do desenvolvimento do projeto.

*Tabela* 1*. - Orçamento preliminar*

|  |  |  |  |
| --- | --- | --- | --- |
| **Material** | **Quantidade** | **Loja** | **Preço** |
| Placa de PVC Expandido | 1 | Mercado Livre | R$ 42,00 |
| Arduino Uno R3 | 1 | FilipeFlop | R$ 59,90 |
| Motor JGB37-550 | 2 | AliExpress | R$ 105,00 |
| Módulo Rele Ywrobot 4 | 1 | FilipeFlop | R$ 34,90 |
| Regulador de Tensão LM7805 | 1 | TIP Eletrônica | R$ 2,50 |
| Sensor de Distância Ultrassônico HC - SR04 | 1 | FilipeFlop | R$ 18,90 |
| Sensor de Linha | 4 | RoboCore | R$ 40,00 |
| Bateria de Chumbo | 1 | TIP Eletrônica | R$100,00 |
| Roda Hobby 65mm | 2 | RoboCore | R$ 20,00 |
| Roda Boba | 1 | FilipeFlop | R$ 13,00 |
| Construção de circuitos¹ | N/A | N/A | R$ 50,00 |
| Frete² | N/A | N/A | R$ 55,01 |
| **Total** |  |  | **R$ 538,71** |

¹. Custos de matérias-primas para a construção dos circuitos.

². Valor total dos fretes.

**4. Análise de autonomia e técnicas de carga de bateria**

Como fonte de tensão para o circuito foi escolhida uma bateria de chumbo. A justificativa para sua escolha foi o custo-benefício entre sua capacidade nominal de corrente e o preço, além da bateria ter um peso considerável que dará mais tração ao robô. A bateria possui 12 volts de tensão, uma capacidade nominal de 7A e possui 2,2kg.

Atualmente o circuito do projeto está consumindo cerca de 3485mA. Este valor foi obtido através da soma de todas as correntes dos componentes do projeto. Leva-se em consideração: o Arduino Uno Rev. 3 com o seu microcontrolador ATmega328P, em operação normal com tensão de 5 volts utiliza 46,5mA; cada motor JGB37-550 em operação normal com tensão de 12 volts utiliza 1,6 A, e como serão utilizados dois motores, o consumo irá para 3200mA; o sensor de linha em operação normal utiliza 50mA, e como serão utilizados quatro sensores, o consumo irá para 200mA; o sensor infravermelho em operação normal utiliza 33mA; e o sensor ultrassônico em operação normal utiliza 3mA. Com isso podemos analisar a provável autonomia do nosso circuito, dividindo o total de corrente disponível pela quantidade que será utilizada.

A capacidade da bateria é de 7000mA e a soma de todos os componentes é igual a 3482,5mA. Logo teremos a autonomia dada pela seguinte equação:

Portanto, conforme o cálculo acima pode-se inferir que a duração da carga da bateria será de aproximadamente 2 horas.

A bateria será recarregada através de um no-break. Logo, não será necessário a construção de um circuito para tal fim.

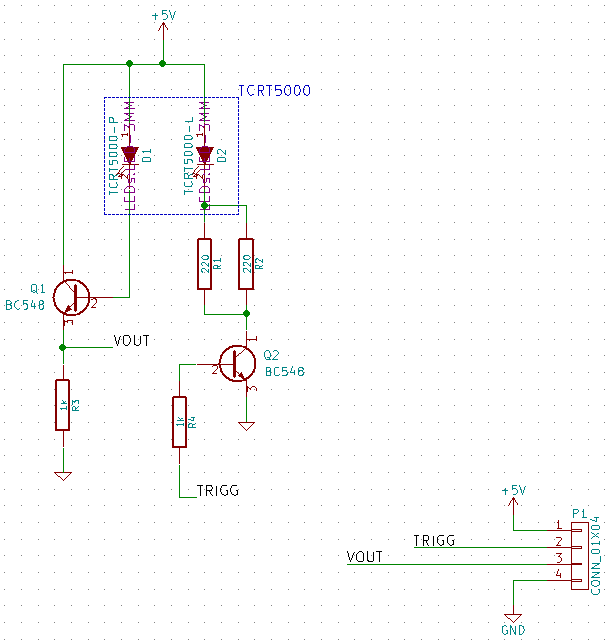
**5. Esquemático e *footprint* dos circuitos**

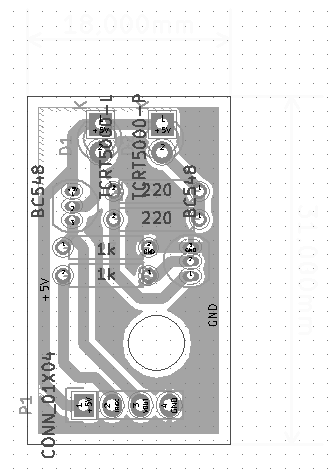
Abaixo encontram-se os circuitos de todos os componentes (sensor infravermelho, sensor de linha e módulo relé) que serão utilizados no projeto.

**5.1. Circuito do Sensor Infravermelho**

O sensor infravermelho utilizado será o modelo revisado apresentado pelo professor em classe. Trata-se de um modelo bem simples que utiliza um fototransistor e um LED IR. Usamos o processo de transferência térmica como prática de sala de aula para desenvolver a placa de circuito impresso para esse tipo de sensor.

A figura 3 mostra o esquemático do sensor. Já a figura 4 é a representação do footprint da placa de circuito impresso do mesmo sensor.

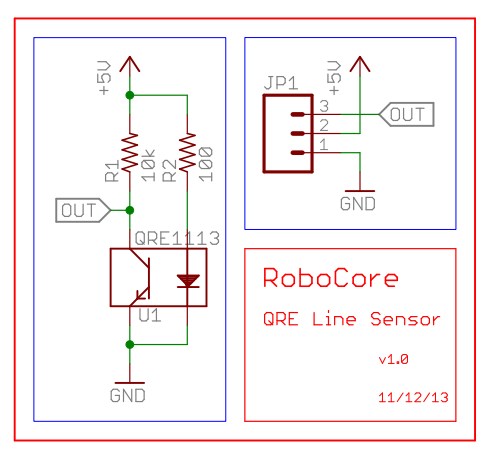
*Figura* 3. *Esquemático do sensor infravermelho*

*Figura* 4. *Footprint do sensor infravermelho*

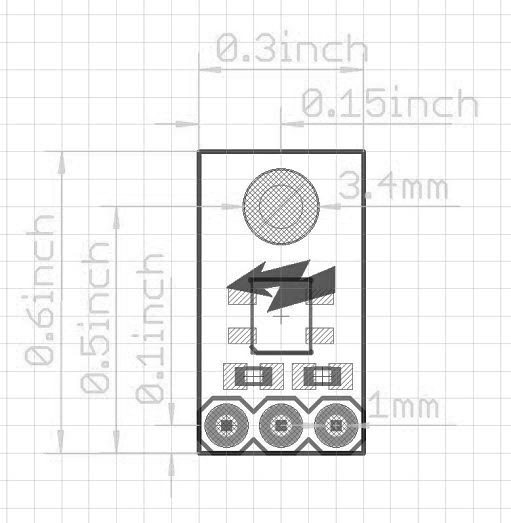
**5.2. Circuito do Sensor de Linha**

O sensor de linha a ser implementado será comprado na loja RoboCore. Trata-se de um modelo bem simples que utiliza um led emissor de luz infravermelha e um fototransistor sensível à luz infravermelha. Quando a placa é alimentada pelos pinos VCC e GND, o LED infravermelho irá acionar um resistor de 100 Ohm na placa e colocado em série com o LED, limita a corrente do mesmo. Um resistor de 10k Ohm coloca a saída em nível alto, mas quando a luz do led é refletida de volta para o fototransistor, a tensão da saída irá variar, diminuindo. Quanto maior a luz infravermelha sentida pelo fototransistor, menor será a tensão de saída no pino da placa.

A figura 5 mostra o esquemático do sensor. Já a figura 6 é a representação do footprint da placa de circuito impresso do mesmo sensor.



*Fig. 5 - Esquemático do sensor de linha*

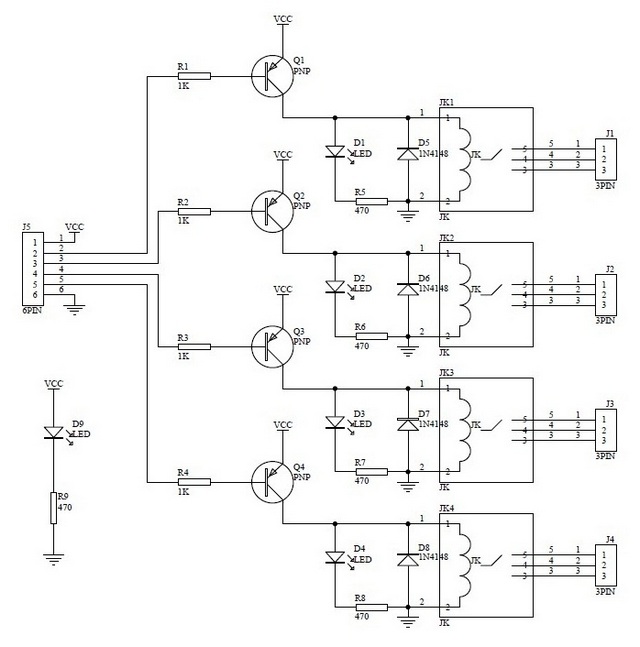


*Fig.6 - Footprint do sensor de linha*

**5.3. Circuito do Módulo Rele *Ywrobot 4 relay***

O módulo relé foi escolhido para realizar o chaveamento do circuito com o propósito alterar o sentido da rotação dos motores. O módulo escolhido para tal fim, foi o modelo *Ywrobot 4 relay*, um módulo que possui 4 relés (modelo SRD), com cada par responsável pelo controle de um motor. A escolha deste módulo se deu pelo mesmo suprir as necessidades de rotacionar os motores e por já possuirmos o mesmo.

Cada par estará conectado a um motor, onde ficará responsável pela rotação do mesmo. O chaveamento é feito de forma alternada, onde o circuito será fechado e irá fazer com que o motor rotacione na direção desejada, de acordo com o chaveamento. Na figura 7, pode ser visto o esquemático do módulo que demonstra o circuito e a sua conexão com o microcontrolador, onde o motor também será alimentado.



*Figura 7. - Esquemático do módulo rele*

**6. Esquemático do circuito integrado**

Abaixo segue o esquemático do circuito completamente integrado, com todos os módulos, sensores e motores conectados ao Microcontrolador e à bateria.

12910496_750636165038677_777146417_n.jpg

Figura 8. - Esquemático do circuito integrado

**7. Código-fonte**

Para que o robô possa executar as exigências do projeto é necessário que ele possua uma programação forte. É por meio da programação que o robô irá executar uma série de comandos que farão com que ele obedeça às regras determinadas, como, por exemplo, identificar o adversário e atacar ou, ao encontrar a linha de borda do ringue, recuar. Abaixo, iremos apresentar os códigos-fonte do conjunto completo do robô, já implementando os sensores ultrassônico e de linha.

*Tabela* 2*. - Código que movimenta o robô*

|  |  |
| --- | --- |
| Declaração das variáveis e importação das bibliotecas. | #include <Ultrasonic.h>  #define RELAY1 11 #define RELAY2 10 #define RELAY3 9 #define RELAY4 8  #define trigger 5 #define echo 4  **int** pinoSensorDireito = A0; **int** pinoSensorEsquerdo = A1;  **int** valorSensorDireito = 0; **int** valorSensorEsquerdo = 0; **int** valorPreto; |
| Setagem e verificação dos Timers. | /\*\* ---------------- SEÇÃO DE MULTITAREFA --------------------- \*\*/ **struct** Timer  {   **unsigned** **long** start; // Armazena o tempo de quando foi iniciado o timer   **unsigned** **long** timeout; // Tempo após o start para o estouro  };   **unsigned** **long** Now ( **void** )  {   **return** millis ( );//Retorna os milisegundos depois do reset  }   boolean TimerEstorou (**struct** Timer \* timer)  {   // Verifica se o timer estourou   **if** ( Now () > timer->start + timer->timeout) {   **return** true;   }   **return** false;  } // Após o tempo estourar temos que iniciar o timer com o tempo atual  **void** timerStart(**struct** Timer \* timer){   timer->start = Now();  }   **void** timerDesloc(**struct** Timer \* timer, **unsigned** **long** tempo){   timer->start = Now()+tempo;  }   Timer timerBorda = {0, 10}; // Verifica se achou a borda a cada 10 milisegundos  Timer timerAchou = {0, 100}; // verifica se achou o oponente a cada 100 milisegundos  Timer timerFrente = {0, 800}; // Tempo que o robô passa indo pra frente  Timer timerGira = {0, 3243}; // Tempo que o robô passa girando  Timer timerfinal = {0, 85000}; // Tempo de 1:30 minutos, final do round  boolean sentidoGiro=true; // Variável criada para o robô girar hora pra direita, hora pra esquerda |
| Setup das portas do Arduino. | /\*\* ----------------- SEÇÃO DE PREPARAÇÃO -------------------- \*\*/ **void** setup() {  Serial.begin(9600);    pinMode(RELAY1, OUTPUT);  pinMode(RELAY2, OUTPUT);  pinMode(RELAY3, OUTPUT);  pinMode(RELAY4, OUTPUT);   pinMode(trigger, OUTPUT);  pinMode(echo, INPUT);   digitalWrite(trigger, LOW);   digitalWrite(RELAY1, LOW);  digitalWrite(RELAY2, LOW);  digitalWrite(RELAY3, LOW);  digitalWrite(RELAY4, LOW);   calibrar();    delay(5000);   timerFrente.start= Now() - timerFrente.timeout; // atrasamos o Start do tempo de ir pra frente de modo ao robô começar a luta indo pra frente.   timerGira.start= Now();   timerAchou.start= Now();  } |
| Loops de verificações dos Timers. Aqui, as funções definidas na Seção de Multitarefa serão executadas. | /\*\* ------------------------ SEÇÃO DE LOOP ---------------------------- \*\*/ **int** i = 0; **void** loop(){  **if**(TimerEstorou(&timerfinal)){   **while**(1)   parado();// função a ser chamada no final de cada round. Se o robô continuar andando isso caracteriza punição   }     **if**(i=0){   timerStart(&timerBorda);   timerStart(&timerGira);   timerStart(&timerAchou);   i+=1;   }     **if**(TimerEstorou(& timerFrente)){   andaFrente();   timerStart(& timerFrente);   }     **if**(TimerEstorou(& timerBorda)){   **if**(achouLinha()){// verifica se achou a borda -- ver pino nomeado como borda   andaTras();   delay(1000);//delay de sobrevivência   timerStart(& timerFrente);//colocar o timer frente no início de seu ciclo   timerDesloc(& timerGira, -timerGira.timeout);// faz com que o robô gire   timerStart(& timerAchou);   }   timerStart(& timerBorda);   }   //dentro da função achou tem um delay,porém é na casa dos Microssegundos e isso o torna desprezível   **if**(TimerEstorou(& timerAchou)){   **if**(achou()){   andaFrente();   timerStart(& timerGira);   }   timerStart(& timerAchou);   }     **if**(TimerEstorou(& timerGira)){   **if**(sentidoGiro){   giraDireita();   sentidoGiro=!sentidoGiro;   }   **else**   {   giraEsquerda();   sentidoGiro=!sentidoGiro;   }   timerStart(& timerGira);   }  } |
| Conjunto de possíveis movimentos do robô | /\*\* ------------------ SEÇÃO DE MOVIMENTAÇÃO ------------------ \*\*/  //Conjunto de funções que indicam o movimento do robô  **void** parado(){  digitalWrite(RELAY1, LOW);  digitalWrite(RELAY2, LOW);  digitalWrite(RELAY3, LOW);  digitalWrite(RELAY4, LOW); }  **void** andaFrente(){  digitalWrite(RELAY1, LOW);  digitalWrite(RELAY2, HIGH);  digitalWrite(RELAY3, LOW);  digitalWrite(RELAY4, HIGH); }  **void** andaTras(){  digitalWrite(RELAY1, HIGH);  digitalWrite(RELAY2, LOW);  digitalWrite(RELAY3, HIGH);  digitalWrite(RELAY4, LOW); }  **void** giraDireita(){  digitalWrite(RELAY1, LOW);  digitalWrite(RELAY2, HIGH);  digitalWrite(RELAY3, HIGH);  digitalWrite(RELAY4, LOW); }  **void** giraEsquerda(){  digitalWrite(RELAY1, HIGH);  digitalWrite(RELAY2, LOW);  digitalWrite(RELAY3, LOW);  digitalWrite(RELAY4, HIGH);  } |
| Análise da distância do oponente e verificação da linha limite do ringue. | /\*\* ---------- SEÇÃO DE CALCULO DE DISTANCIA --------------- \*\*/ **float** distancia() {   **float** tempo, dist;   digitalWrite(trigger, HIGH);   delayMicroseconds(10);   digitalWrite(trigger, LOW);     tempo = pulseIn(echo, HIGH);   dist = tempo / 29.4 / 2;   Serial.print("Distancia em cm: ");  Serial.println(dist);    **return** dist;  }  **void** calibrar(){  valorSensorDireito = analogRead(pinoSensorDireito);  valorSensorEsquerdo = analogRead(pinoSensorEsquerdo);   valorPreto = (valorSensorDireito + valorSensorEsquerdo)/2;  Serial.print("Valor do preto: ");  Serial.println(valorPreto); }  boolean achouLinha(){  valorSensorDireito = analogRead(pinoSensorDireito);  valorSensorEsquerdo = analogRead(pinoSensorEsquerdo);    Serial.print("Sensor direito: ");  Serial.println(valorSensorDireito);  Serial.print("Sensor esquerdo: ");  Serial.println(valorSensorEsquerdo);     **if**((valorSensorDireito > valorPreto+125) || (valorSensorEsquerdo > valorPreto+125)){  Serial.println("Achou!");  **return** true;   }   **else** {  Serial.println("Continue andando!");  **return** false;  }  }  boolean achou(){  **if**(distancia() < 45)  **return** true;  **else**  **return** false; } |

**8. Fluxograma**

A figura 9 apresenta um fluxograma do funcionamento do código e do comportamento do robô ao obedecer os comandos.

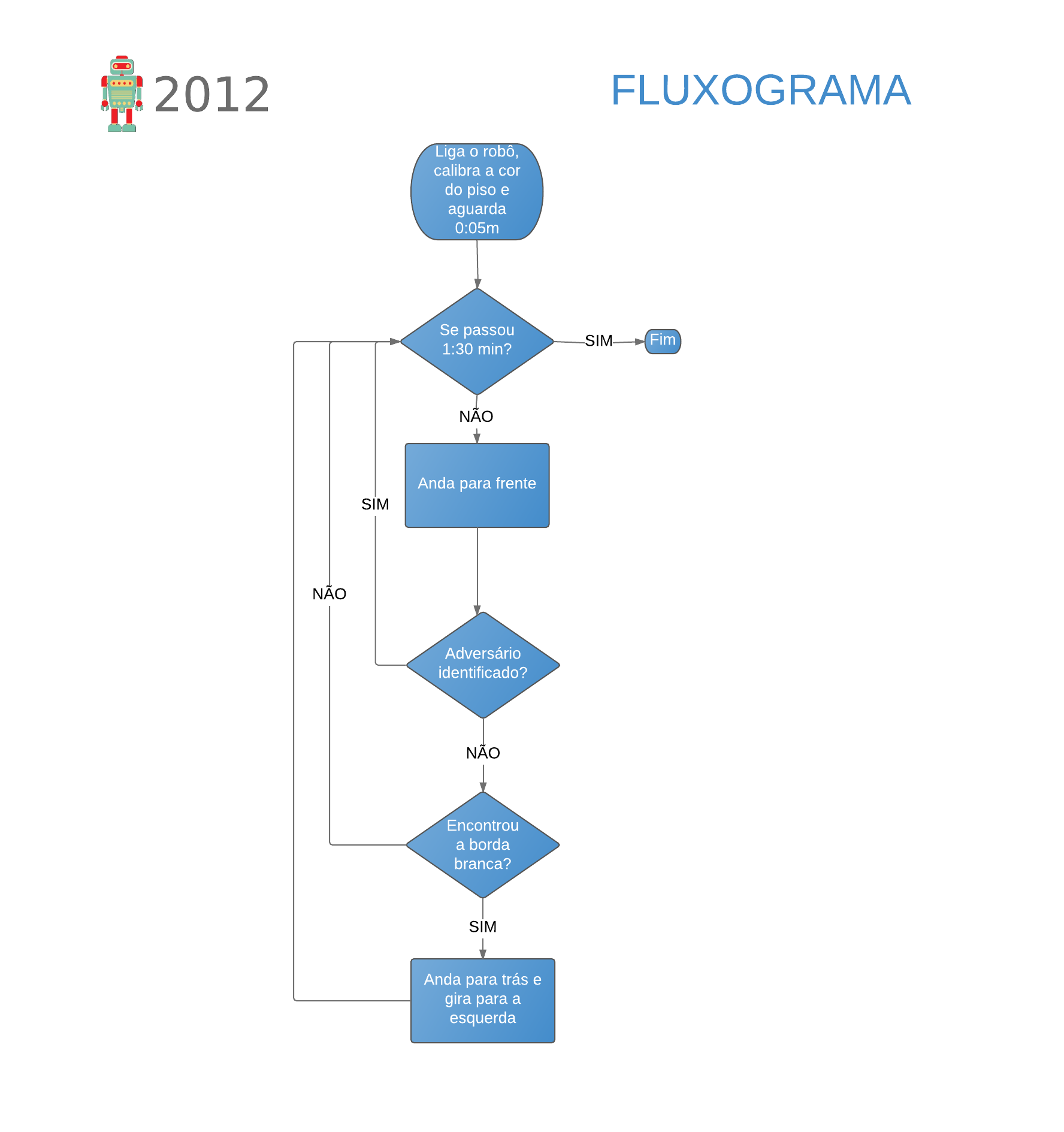


Figura 9. - Fluxograma da programação

**9. Imagens**



Figura 10. - Vista Frontal do Robô

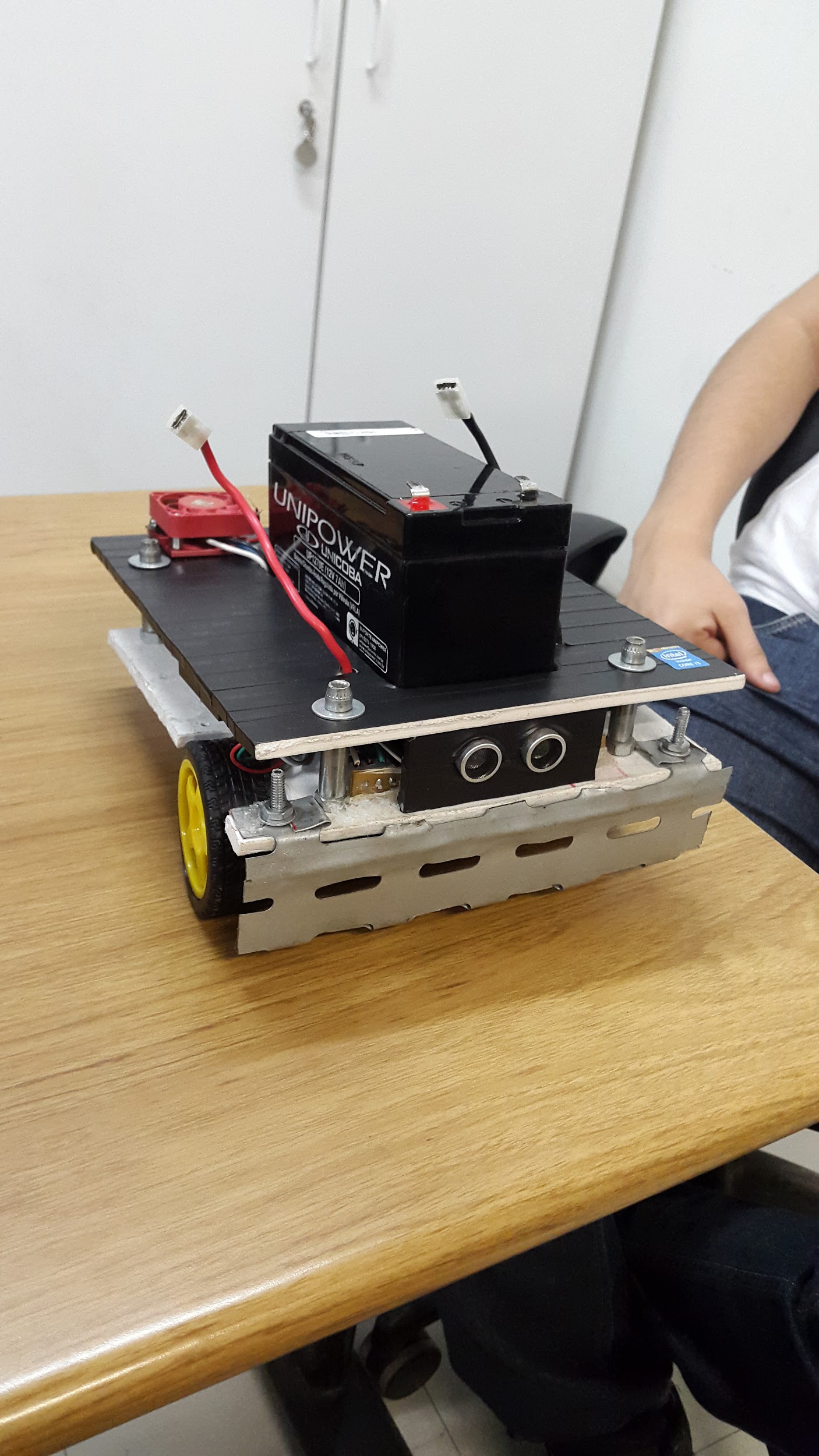


Figura 10. - Vista do Robô em Diagonal

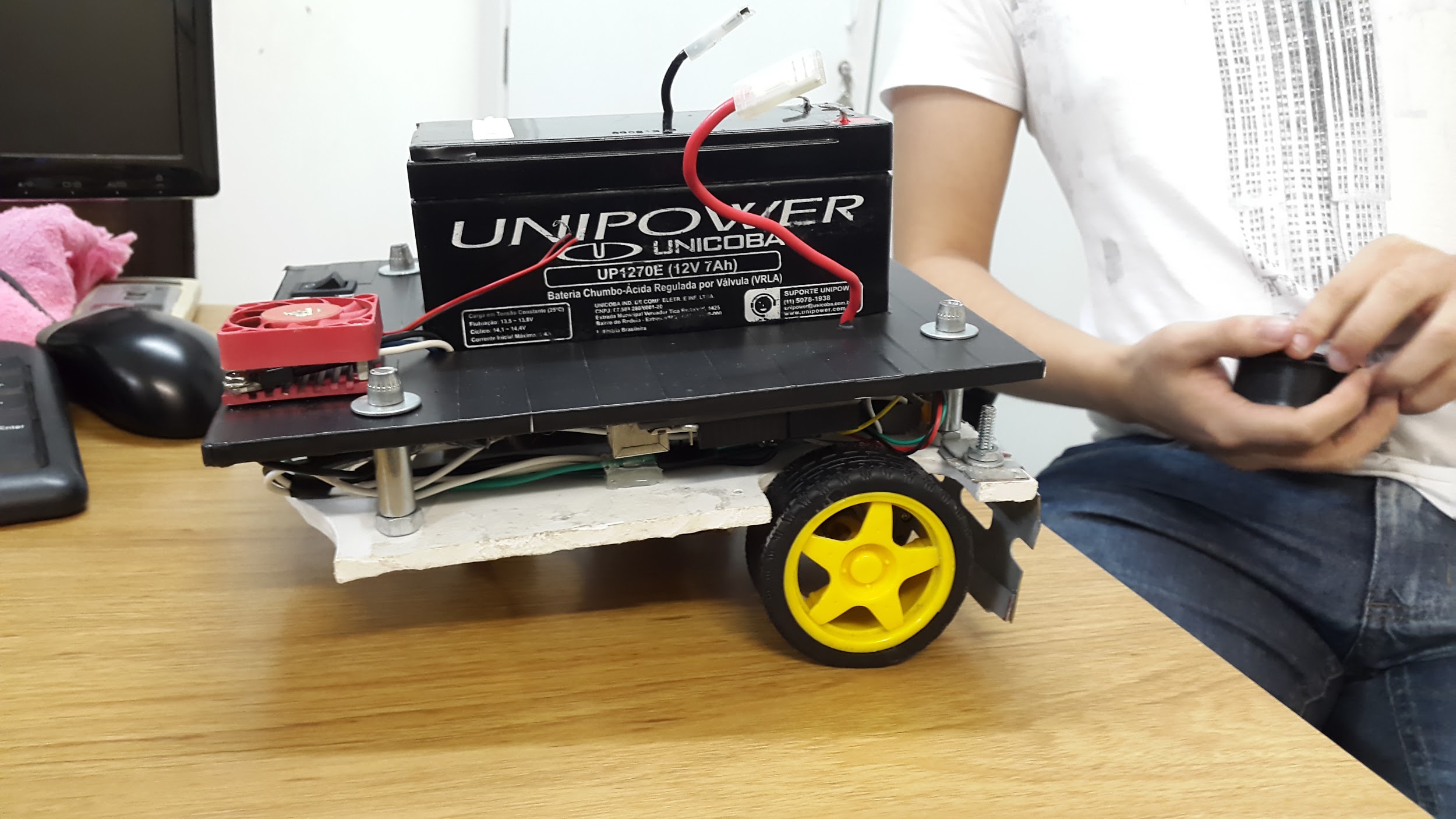


Figura 10. - Vista Lateral do Robô

**10. Referências**

Loja AliExpress - <http://pt.aliexpress.com/br_home.htm>

Loja RoboCore - <https://www.robocore.net/>

Loja Abiu Store - <https://www.abiustore.com.br>

Loja FilipeFlop - <https://www.filipeflop.com>

Loja UsinaInfo - https://www.usinainfo.com.br/reguladores-de-tensao/regulador-de-tensao-7805-5v-para-projetos-3074.html

TIP Eletrônica - <http://www.tipeletronica.com.br/v2/>

Módulo relé (incluindo *datasheet*) - <http://www.henrysbench.capnfatz.com/henrys-bench/5-volt-4-channel-arduino-relay-module-user-manual/>

Esquemático para o módulo rele - <http://www.hobbyist.co.nz/?q=interfacing-relay-modules-to-arduino>