Robô Seguidor de Linha Utilizando MSP430G2231

Mikhaelle de C. Bueno, *UnB Gama,Matrícula:15/0018673*
Guilherme Felix De Andrade, *UnB Gama,Matrícula:14/0142223*

Resumo—O projeto consiste na criação de um robô seguidor de linha usando o microcontrolador MSP430G2553 da Texas Instruments. O robô seguidor de linha consegue se guiar por uma linha preta automaticamente, isso acontece por causa do interligamento entre o LDR (sensor de luz), transistores e motores. Como a linha preta absorve a luz, quando o ldr estiver apontado para ele, ele terá uma resistência muito baixa e quando estiver apontado para a parte branca sua resistência aumenta, mandando um sinal para o transistores e depois para o motor correspondente parar enquanto o outro ainda gira, até corrigir o posicionamento do robô.

Index Terms—Microcontrolador, MSP430, MSP430G2231, Line follower, seguidor de linha, robô.

1 Introdução

O robô seguidor de linha deverá percorrer um circuito branco com uma linha preta de maneira autônoma e se guiando pela luz refletida ou não. Para conseguir diferenciar o preto do branco haviam duas possibilidades de sensores, o infravermelho[1] e o LDR[2]. Entre os dois foi escolhido o LDR por causa da simplicidade da lógica que seria aplicada, tendo uma boa eficiência, além disso os membros já tinham familiaridade com o sensor. Poderia ser usado duas disposições diferentes de sensores, na primeira [1] seriam utilizados três sensores, portanto, enquanto apenas o sensor central estivesse na linha preta o robô seguiria andando mantendo as rodas na mesma velocidade, se o sensor esquerdo estivesse sobre a linha preta a roda esquerda travaria e a direita continuaria andando e se o sensor esquerdo estivesse sobre a linha preta, ocorreria o contrário. Também é possível manter a rotação do motor, sem necessidade de parar as rodas, como demonstrado na figura 1.

Outra disposição possível é a com apenas dois LDRs, enquanto os dois estivesse recebendo luz e portanto fora da linha os motores

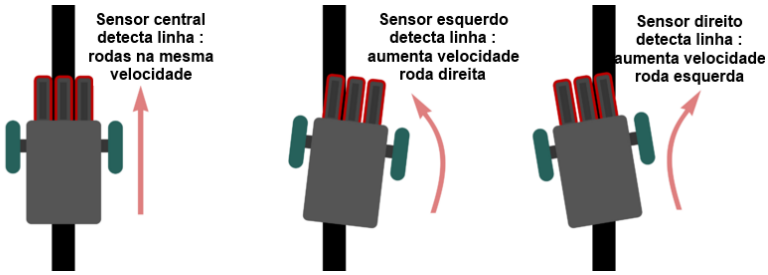

Figura 1. funionamento motor x ldr

funcionariam na mesma velocidade e o carrinho seguiria reto, se o direito deixar de captar, significa que o rôbo está girado para a esquerda e que o motor da direita deve ser travado, se o LDR esquerdo não estiver recebendo luz ocorrerá a lógica contrária.

Os motores de corrente contínua consistem numa forma simples e barata de se obter propulsão mecânica para dispositivos eletromecânicos. Existem diversos tipos de motores DC, tais como os de imã permanente, sem escovas ou de relutância variável. Os mais comuns são os que fazem o uso de escovas. Neles um conjunto de bobinas gira tendo sua corrente comutada por escovas que invertem o sentido da corrente a cada meia volta de modo a manter o movimento. Estes motores possuem um rendimento razoável quando usados em projetos de robótica e mecatrônica, sendo por

- *Mikhaelle E-mail: Mikhabueno@outlook.com.*
- *Guilherme E-mail:guilhermefelixandrade@gmail.com*

.

este motivo sua escolha para o projeto.

É importante ter recursos para se controlar a velocidade e o sentido da rotação do motor DC. O sentido de rotação dependerá da polaridade da tensão aplicada, ou seja, do sentido da corrente pelos enrolamentos, enquanto que a velocidade pode ser controlada de duas maneiras: pela tensão aplicada de forma contínua ou na forma de pulsos PWM. O modo mais simples de controlar essa velocidade é através de um circuito utilizando Mosfet, capacitor e resistor, como demonstrado no esquemático da figura 3.

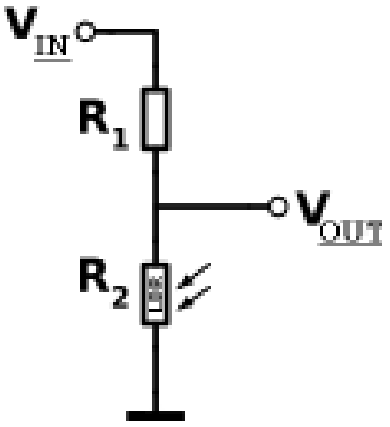

Figura 2. funcionamento LDR[4]

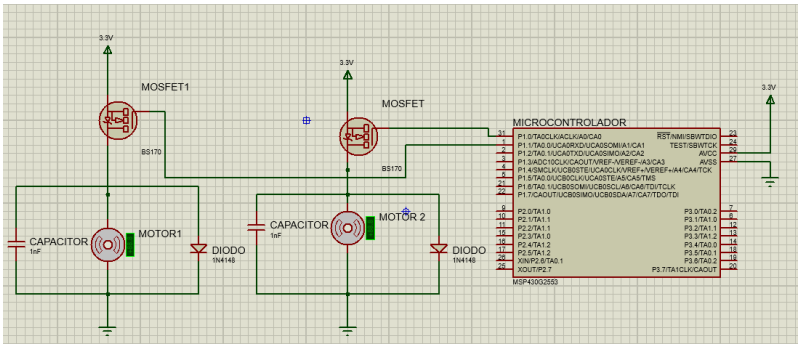

Figura 3. Diagrama esquemático do circuito do motor. Autoria própria

2 DESENVOLVIMENTO

2.1 Materiais utilizados

- Microcontrolador MSP430, Texas Instruments
- Resistores(ohms) - 220, 120, 10K e 22K
- Capacitores - 0.1uF
- Motores Dc 3-6 Volts
- Leds brancos de alto brilho
- Mosfet Canal-N - BS170
- Diodos - 1N4148
- Sensores LDR

2.2 Descrição de Hardware

Os sensores LDR'S detectam a linha preta e ficam com o valor de resistência alta e quando os sensores detectam luz ficam com o valor de resistência baixo. É colocado em série com um resistor de 22K ohms e faz o papel de um divisor de tensão. R1 =22K ohms, R2 = LDR, como esquematizado na figura 2. Vout = (Vin R2) / (R1+R2)

A figura 4 demonstra como foi feita a conexão dos LDRs, resistores e leds a placa MSP430. Inicialmente a equipe esquematizou com apenas um LDRs para fim de teste e depois montou a configuração e atualizou o software para 2 LDRs.

O diagrama esquemático(figura 3) do projeto inicial do motor foi feito a partir do software Protheus, onde temos como o controlador do circuito o MSP430, responsável em converter o sinal analógico de tensão dos sensores LDR's e digitalizá-lo para mandar o sinal do PWM (Pulse-Wildth Modulation) que comanda a rotação dos motores para que o robô siga a linha ou corrija sua posição.

2.3 Descrição de software

Para o controle do motor foi utilizado a função digitalwrite do software energia, que escreve um valor alto ou baixo em um pino digital. Os pinos P1.1 e P2.2 foram setados como saída do MSP430 onde será enviado um sinal de 3.3 volts (high) para o motor ou 0 volts (low) dependendo das condições dos sensores LDR's, se estão detectando luz ou não. O controle dos motores é uma função do código principal. O código está imprimido nas figuras 5 e 6.

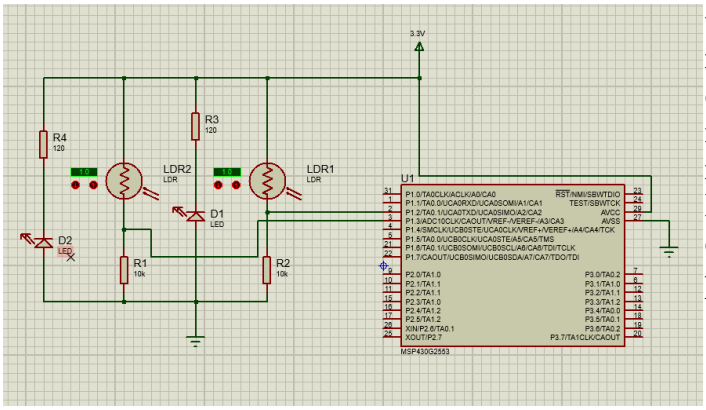

Figura 4. Diagrama esquemático do LDR. Autoria própia

Quanto ao código do ldr testado e operando, a lógica utilizada foi que, enquanto os ldrs estivessem sem estímulo da luz, portanto com uma tensão baixa, os leds da própria placa MSP430 das pinagens P1.0 e P1.6 estarão ligados, o que corresponde a dizer que o robô está sobre uma superfície preta. Quando o LDR da direita receber algum estímulo luminoso o led vermelho P1.0 apagará e o verde irá continuar aceso, o que significa que o robô está virado a direita e que a roda esquerda deve ser travado para ocorrer a correção da rota. Quando o LDR da esquerda receber o estímulo a lógica contrária da descrita anteriormente ocorrerá e quando os dois estiverem sendo iluminados os leds estarão apagados e o robô poderá seguir em linha reta, com velocidade igual nos dois motores. O código dos ldrs estão ilustrados nos códigos da fígura 7 e 8.

3 Resultados

O sensor LDR funcionou corretamente como desejado, porém este componente é muito sensível a qualquer iluminação, para isso a melhor escolha será isolar completamente o sensor a exposição da luz ambiente. Como a equipe não tinha três LDRs, optou-se pela construção prática da lógica com 2 LDRs, além disso os dois LDRS usados eram de fabricantes diferentes e portanto quando expostos a luz ambiente continham resistências diferentes, a correção dessa discrepância foi aplicada no código do LDR. Para os motores foi necessário a utilização do Mosfet para evitar ligá-lo diretamente no microcontrolador e ter problema em queimar e prejudicar o circuito. No primeiro instante foi feito os códigos para o teste dos motores e os resultados foram satisfatórios. Porém a dificuldade será juntar o código com o LDR para conseguir alcançar o objetivo do projeto.

4 Conclusão

Os sensores e os motores funcionaram como o esperado, sendo que os sensores enviam o sinal analógico de tensão para o microcontrolador e o mesmo digitaliza o sinal para que possa controlar os motores. Os próximos passos do projeto são integrar e otimizar o código, passar para código em C as funções e bibliotecas que o software energia proporciona e acrescentar interrupções e sub-rotinas significativas do código em Assembly.

Referências

[1] Adilson Thomsen, Como montar um Robô Seguidor de Linha com Arduino Motor Shield.Disponível em:https://www.filipeflop.com/blog/projeto-robo-seguidor-de-linha-arduino/. Acesso em: 26/08/2017.

[2] Amorim, Andrique. Robô seguidor de linha autônome itulizando o controlador proporcional-derivativo em uma plataforma de hardware/software livre.2011. Departamento de Ciências Exatas - UESB.

[3] Rwb, The Line Follower Robot with Texas Instruments 16-Bit MSP430G2231 Microcontroller. Disponível em:http://www.ermicro.com/blog/?p=2104. Acesso em: 30/08/2017

[4] Electrônica, Divisor de tensão. Disponível em:https://www.electronica-pt.com/divisores-tensao. Acesso em: 02/10/2017
//

5 Anexos

```
const int MOTOR1 = P1_1;
const int MOTOR2 = P1_2;

void setup() {
  pinMode(MOTOR1, OUTPUT);
  pinMode(MOTOR2, OUTPUT);
}

void controle_motor(void){

    if  (LDR1= 1 && LDR2 =0 && LDR3= 1 ){

        digitalWrite(MOTOR1, HIGH);
        digitalWrite(MOTOR2, HIGH);

      }else if(LDR1=0  && LDR2 =1 && LDR3= 1

        digitalWrite(MOTOR1, LOW);
        digitalWrite(MOTOR2, HIGH);

      }else if(LDR1=0  && LDR2 =0 && LDR3= 1){

        digitalWrite(MOTOR1, LOW);
        digitalWrite(MOTOR2, HIGH);

      }else if(LDR1=1  && LDR2 =1 && LDR3= 0
```

Figura 5. Código do motor

```
        digitalWrite(MOTOR1, HIGH);
        digitalWrite(MOTOR2, LOW);

      }else if(LDR1=1  && LDR2 =0 && LDR3= 0)

        digitalWrite(MOTOR1, HIGH);
        digitalWrite(MOTOR2, LOW);

     }else {
      digitalWrite(MOTOR1, LOW);
      digitalWrite(MOTOR2, LOW);
      }

  return (MOTOR1, MOTOR2);
}
```

Figura 6. Continuação código do motor

```
#include <msp430g2553.h>
#define LDR1 BIT1
#define LDR2 BIT3
#define LED1 BIT0
#define LED2 BIT6


int main()
{
  float i=1; // i = luminosidade natural do local
  float j=1.9;
  float Valorlido_LDR1, Valorlido_LDR2;
  WDTCTL = WDTPW | WDTHOLD;
  P1OUT |= LED1;
  P1DIR |= LED1;
  P1OUT |= LED2;
  P1DIR |= LED2;

  while (1){

    Valorlido_LDR1=P1IN&LDR1; // vai receber o valor lido do ldr1
    Valorlido_LDR2=P1IN&LDR2;// vai receber o valor lido do ldr2
    //Valorlido_LDR3=P1IN&LDR3; // vai receber o valor lido do ldr3
```

Figura 7. Código do LDR

```
    if (Valorlido_LDR1<=j && Valorlido_LDR2<=i)
      {
        P1OUT ^= LED1;
        P1OUT ^= LED2;
      }
    else if (Valorlido_LDR1>j && Valorlido_LDR2>i)
      {
        P1OUT &= ~LED1;
        P1OUT &= ~LED2;
      }
     else if (Valorlido_LDR1<=j &&Valorlido_LDR2>i)
     {
       P1OUT ^= LED1;
       P1OUT &= ~LED2;
     }

     else if (Valorlido_LDR1>j &&Valorlido_LDR2<=i)
     {
       P1OUT ^= LED2;
       P1OUT &= ~LED1;
     }

  }
  return 0;
}
```

Figura 8. Continuação código LDR