PC2 - Ponto de Controle 02

Projeto "Meu Cão Guia"

Násser Yousef Santana Ali - 13/0034398
Universidade de Brasília - FGA
Área Especial de Indústria Projeção A, UNB - DF-480-Gama Leste, Brasília - DF, 72444-240
nasser.yousef.unb@gmail.com

Natália Santos Mendes – 13/0128082
Universidade de Brasília - FGA
Área Especial de Indústria Projeção A, UNB - DF-480-Gama Leste, Brasília - DF, 72444-240
nataliammendes04@gmail.com

Resumo: **A Raspberry pode ser explicada como um computador em uma pequena placa. Logo, seu poder de processamento é alto, e sua utilização ampla. Pensando nisso, pensou-se no projeto "Meu Cão Guia", que é um robô capaz de passar em pequenos ambientes, com capacidade de "visualizar" o lugar antes do usuário.**

Palavras-chave: **Raspberry Pi 3; Robô; Robô com camera.**

***Abstract*— Raspberry can be explained as a computer on a small board. Therefore, its processing power is high, and its use wide. With this in mind, the "My Dog Guide" project was thought of as a robot capable of passing in small environments, with the ability to "visualize" the place before the user.**

Keywords: Raspberry Pi 3; Robot; Camera Robot.

I. INTRODUÇÃO

Tendo em visto que tudo envolve tecnologia, algumas áreas ainda não contam com essa ajuda para tarefas do cotidiano. O acesso a determinadas áreas pode ser complicado e pode necessitar de maiores investimentos. A caráter de exemplificação, existem dutos de ar, lajes, áreas que oferecem algum tipo de risco, necessitam de verificação constante ou precisam de manutenção em algum momento. O acesso proporcionado à essas áreas demanda serviço especializado, ou pelo menos envolve dúvidas e dificuldades em resolver os problemas associados.

Com o intuito de resolver tal, pensou-se no projeto "Meu Cão Guia". Tal como os cães guias direcionam o deficiente para os lugares que eles precisam e não podem enxergar, o robô proposto teria tamanho suficiente para adentrar em lugares de difícil acesso, "enxergar" todo o ambiente para o usuário e assim seria possível decidir o que deve ser feito no local antes mesmo de entrar nele. Em um projeto de maior porte pode-se citar que o próprio robô seja o agente neutralizador de problemas.

II. OBJETIVOS

O objetivo é criar um robô que possa ser controlado pelo usuário para andar em ambientes de difícil acesso oferecendo imagens do local. Para isso, o robô terá uma câmera acoplada onde oferecerá as imagens em tempo real, além de um LED que servirá como flash caso o ambiente esteja escuro. Também terá que ter um tamanho reduzido e rodas que possam superar alguns obstáculos.

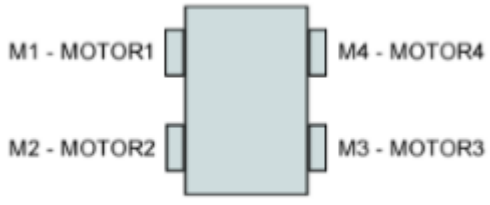
III. DESCRIÇÃO DO HARDWARE

Para todo o projeto, serão utilizados seguintes componentes:

- Raspberry Pi 3;
- Controladores de Motor L293D;
- Baterias de alimentação;
- Bateria p/ Recarga de Celular;
- Reguladores de Tensão p/ 5v;
- Protoboards;
- Câmera Module 5MP p/ Raspberry Pi;
- Estrutura do Robô (Chassi 4WD);
- Motores DC, c/ caixa de redução;
- Fios e Jumpers Macho-Fêmea;
- Transistor NPN BC547;
- Servomotor Futaba S003;
- Resistores;
- LEDs;
- Computador Pessoal;
- Rede em Comum (Wi-Fi - Wireless);

Porém, como o projeto ainda não está totalmente pronto, apenas a parte que envolve o controle dos motores e dos LEDs está disponível com todos os componentes.

É necessário, portanto, conhecer como se baseia o funcionamento dos motores, bem como o seu controle. Os motores DC, para a devida implementação, precisam agir de forma coordenada. Para ir para frente, todos os motores precisam ir para frente; para trás, todos os motores devem girar para trás; para a direita, parte deve ir para frente e parte para trás, assim como para a esquerda. A figura 01 pode ilustrar tal ação com o seguinte esquemático.



AÇÃO

	M1	M2	M3	M4
P/ FRENTE	Frente	Frente	Frente	Frente
P/ TRÁS	Trás	Trás	Trás	Trás
P/ ESQUERDA	Trás	Trás	Frente	Frente
P/ DIREITA	Frente	Frente	Trás	Trás

Figura 1 - Esquemático de trabalho dos motores para realização dos movimentos.

Para haver a comutação de sentido de rotação do motor, é necessário uma ponte H, que inverte a polaridade que está sendo direcionada ao motor DC, que faz que com que seja possível a comutação de sentido. O esquemático básico de uma ponte H pode ser ilustrado na figura 02. As comutações de ligações permitem que sejam invertidas a polaridades da alimentação que é dada ao motor, fazendo com que tal gire no sentido horário e anti-horário. Mais especificamente, uma ponte H é construída com transistores e diodos de proteção, para que garanta o chaveamento e a passagem de corrente elétrica na direção correta para cada caso.

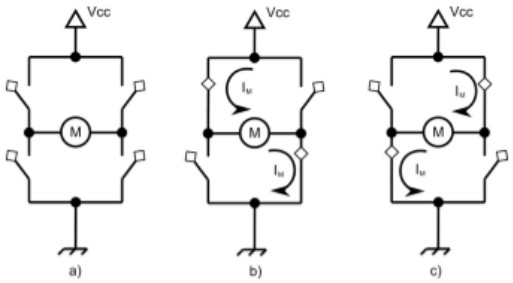


Figura 2 – Esquemático de funcionamento de uma ponte H.

O driver utilizado é o L293D, a qual permite o controle de 2 motores DC. O esquemático de ligação e controle do mesmo poder ser visualizado na Figura 03. Os pinos Pin 4, Pin 5, Pin 6 e Pin 7 são aqueles que são conectados ao RPi, responsáveis pela comutação dos motores. O pino 16 é responsável pela alimentação do circuito, ligado em 5v. Já os pinos 1 (motor A) e 9 (motor B) são pinos que recebem uma entrada analógica de 0v a 5v e, de acordo com a tensão, há o controle da velocidade dos motores. Ambos são colocados em +5v. O pino 8 é responsável pela alimentação dos motores, e tal pino pode variar de 3v a 36v. Tal foi colocado com 7,4v, diretamente com a alimentação da bateria

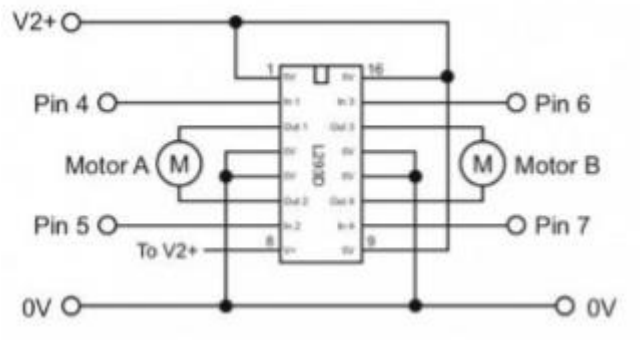


Figura 3 - Esquema de ligação e alimentação do CI L293D.

IV. DESCRIÇÃO DO SOFTWARE

Primeiramente, foram declaradas as bibliotecas usadas para esse código. Foi usada a WiringPi.h para a parte GPIO, Pthread.h para as threads utilizadas, stdio.h e stdlib.h para funções como printf, e termios.h e unistd.h para a função de obter as entradas. Posteriormente é definido todos os pinos que serão utilizados através do #define.

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h> // Biblioteca WiringPi
#include <pthread.h> // Para as threads
#include <termios.h> // As trÃªs bibliotecas seguintes sÃ£o
#include <unistd.h> // para a funÃ§Ã£o getch.
#define motor1Pin1 0 // Definindo os pinos para
#define motor1Pin2 1 // cada um dos dois fios de
#define motor2Pin1 2 // cada motor
#define motor2Pin2 3
#define motor3Pin1 4
#define motor3Pin2 5
#define motor4Pin1 6
#define motor4Pin2 7

#define SERVO 21
#define BUZZER 22

#define LED1 23
#define LED2 24
#define LED3 25
#define LED4 26
#define LED5 27
#define LED6 28
#define LED7 29
```

Depois disso, era necessário indicar quais pinos seriam portas de saída e portas de entrada. Como no projeto não foram usados sensores, todas as portas são de saída. Para indicar isso, foi usado o pinMode.

```
    pinMode(motor1Pin1, OUTPUT);
    pinMode(motor1Pin2, OUTPUT);
    pinMode(motor2Pin1, OUTPUT);
    pinMode(motor2Pin2, OUTPUT);
    pinMode(motor3Pin1, OUTPUT);
    pinMode(motor3Pin2, OUTPUT);
    pinMode(motor4Pin1, OUTPUT);
    pinMode(motor4Pin2, OUTPUT);

    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
    pinMode(LED3, OUTPUT);
    pinMode(LED4, OUTPUT);
    pinMode(LED5, OUTPUT);
    pinMode(LED6, OUTPUT);
    pinMode(LED7, OUTPUT);
```

Então, o programa principal primeiramente obtém-se a entrada, que são letras (ou seja, tipo char) e podem ser “w” (para o robô ir para frente), “a” (para ir para esquerda), “s” (para ir para trás), “d” (para ir para direita) e “l” (para ligar ou desligar a lanterna (LEDs). Para conseguir armazenar essa entrada sem ser preciso apertar “enter”, foi necessário usar uma função que usa a struct que existe na biblioteca termios.h. Dessa forma, ao digitar a entrada, não é necessário apertar “enter”, pois a entrada já é armazenada assim que é inserida.

```
int getch(void)
{
   struct termios oldattr, newattr;
   int ch;
   tcgetattr( STDIN_FILENO, &oldattr );
   newattr = oldattr;
   newattr.c_lflag &= ~( ICANON | ECHO );
   tcsetattr( STDIN_FILENO, TCSANOW, &newattr );
   ch = getchar();
   tcsetattr( STDIN_FILENO, TCSANOW, &oldattr );
   return ch;
}
```

Depois de obter a entrada, foi feito uma série de condições para analisar quais pinos seriam setados e quais zerados. Para isso, usou-se o “digitalWrite” da biblioteca wiringpi.h. Toda a lógica para isso foi usada a partir do que foi descrito no item “Descrição do hardware”. Além disso, isso foi colocado em um laço infinito para o robô andar continuamente.

A lista de comandos de ir para frente, trás, esquerda e direita, que são dados aos controladores de motor L293D, podem ser vistos na figura 03.

```
/*********************************************************
------LISTA DE COMANDOS DO DRIVER DE MOTOR (L293D):-------
*********************************************************

I1 I2 AÇÃO
0  0  PARADO
0  1  SENTIDO HORÁRIO (IR P/ FRENTE)
1  0  SENTIDO ANTI-HORÁRIO (IR P/ TRÁS)
1  1  PARADO
```

Figura 3 – Inputs do CI L293D

V. RESULTADOS

Tendo feito e executado tal projeto, foram encontradas dificuldades na montagem e organização para que tudo funcionasse em conformidade. Os motores DC, sua dificuldade ficou em sua plena alimentação, bem como a conformidade dos 4 motores para realizarem os movimentos de ir para frente, trás, esquerda e direita.

É importante ressaltar que, o pensamento inicial era colocar um LDR, um dispositivo que conforme a luminosidade no local muda sua resistência, para o controle automático dos LEDs de iluminação. Porém, o robô sempre vai se encontrar em locais escuros, logo esse sensor não seria muito interessante. Portanto foi colocado apenas um trecho de código que acenda os LEDs conforme a entrada do usuário.

Como a streaming (transmissão ao vivo), requer bastante processamento da RPi, o programa foi simplificado ao máximo, com o intuito de dispor apenas de 2 threads (fluxos de execução) principais, sendo o primeiro o controle dos motores e o segundo para a recepção da entrada do usuário para ligar ou desligar os LEDs de iluminação. O problema de mutithreading pode ser visível e significativo quanto se está trabalhando com aplicações que estão rodando em paralelo e necessitam de maior poder de processamento.

Porém, com todas as dificuldades, ao final, a etapa fora concluída com êxito.

VI. CONCLUSÃO

O RPi é uma boa plataforma de desenvolvimento de projetos, consegue se sair bem em projetos e controle, como fora feito com o robô. Por seus pinos serem de baixa voltagem e corrente com relação aos outros microcontroladores, pode-se notar que o seu propósito é o controle de operações.

Dentre as dificuldades, uma das maiores foi a definição da melhor característica e configuração de bateria que era necessária para alimentação dos 4 motores DC, 2 CIs L293D e alimentação dos LEDs, de modo que tudo funcionasse de forma plena.

De maneira geral, tudo ocorreu conforme o esperado, dentro do comportamento e aparição de possíveis problemas no desenvolvimento do código de controle e execução por parte da RPi.