PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO

VITOR DE ALMEIDA SILVA
Matricula: 2016.1.0033.0549-7

TRABALHO 1 N2: CARRO AUTONOMO

TALLES MARCELO G DE A BARBOSA

GOIÂNIA,
2019

VITOR DE ALMEIDA SILVA
Matricula: 2016.1.0033.0549-7

TRABALHO 1 N2: CARRO AUTONOMO

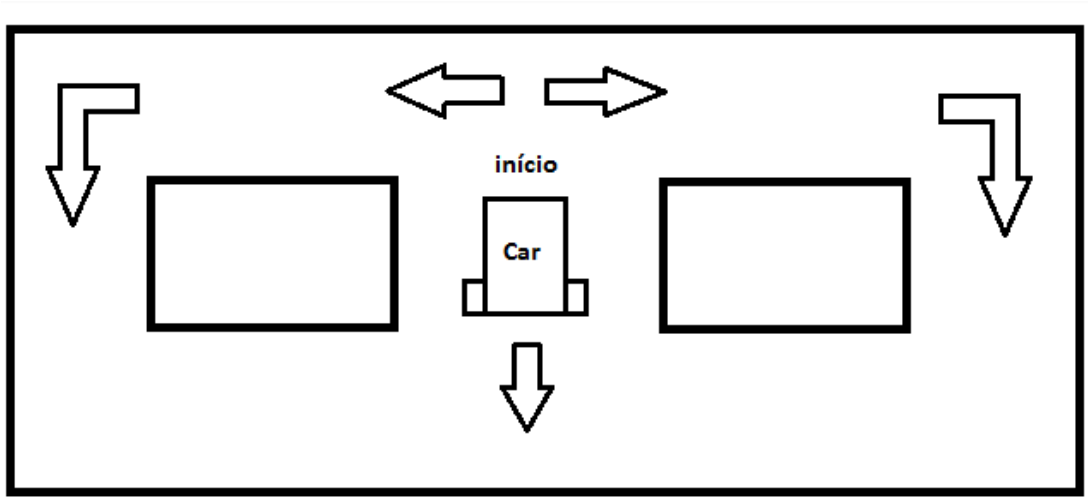
Trabalho apresentado como requisito parcial para obtenção de nota na disciplina Sistemas Embarcados no Curso de Engenharia da computação, na Pontifícia Universidade Católica de Goiás.
Talles Marcelo G de a Barbosa

GOIÂNIA,
2019

1 ENUNCIADO

Controlando os valores de prioridade, o tempo de duração e as seções críticas envolvendo as tarefas, faça com que o veículo funcione de maneira autônoma. A Figura 1 mostra o ambiente de tarefas. Considere que este é completamente observável e determinístico. O veículo pode operar com velocidade constante.

Figura 1: ambiente de tarefas



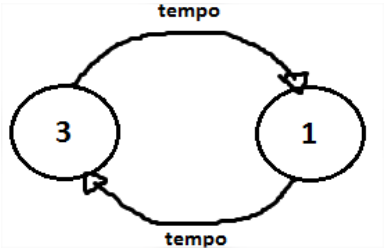
O carro tem tração diferencial.

Tarefas: Fazer um total de 3 percursos diferentes.

1) Seguir adiante;
2) Marcha ré;
3) Virar á esquerda;
4) Virar a direita.

O exemplo de um roteiro é mostrado na Figura 2.

Figura 2: exemplo de roteiro



2 DESENVOLVIMENTO

2.1 *Free RTOS*

Para o desenvolvimento do algoritmo foi utilizado a biblioteca ***Free RTOS.*** Tal biblioteca permite dividir as funções em tasks (tarefas). Cada task pode ser configurada com um valor de prioridade que vai de 0 até 3, sendo 0 a maior prioridade (ARDUINO PROJECT HUB, 2019).

Deste modo, a biblioteca permite o escalonamento das tasks de forma a gerenciar a mudança de contexto de acordo com as prioridades definidas para cada uma. Pode-se imaginar as tasks como threads, já que, os comportamentos são similares.

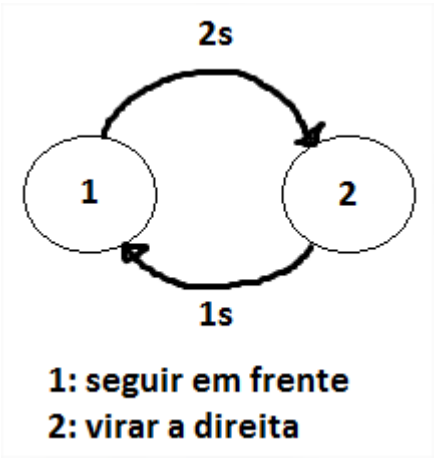
As prioridades vão de 0 até 3, sendo que a maior prioridade é sempre a de menor número. De início as tasks são iniciadas ao mesmo tempo, sendo a que continua é a com prioridade de menor número.

2.2 Definição de roteiros e circuitos

Antes de definir o período e as prioridades de cada tarefa, antes, é necessário definir o roteiro e o circuito que o veículo irá percorrer. Utilizando os modelos indicado em sala, o roteiro 1 é mostrado na Figura 3 e o circuito 1 definido é mostrado na Figura 4. São um total de 3 percursos.

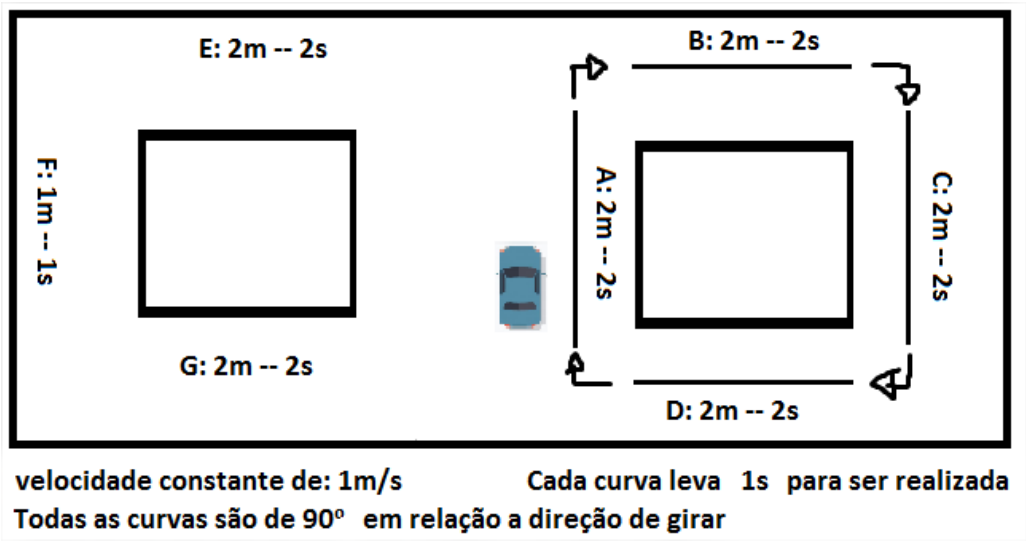
Percurso 1:

Figura 3: roteiro



Fonte: Autoral

Figura 4: circuito 1



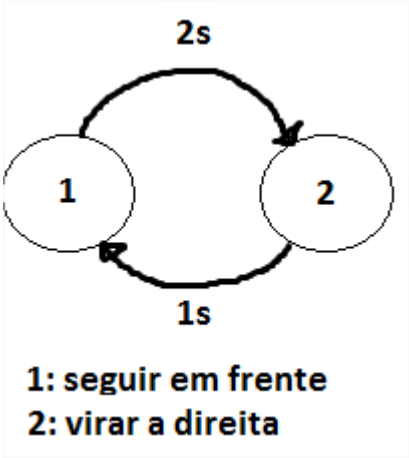
Fonte: Autoral

Os tempos exigidos para percorrer cada distância já foram definidos na imagem, assim como, a quantidade em metros do caminho.

Percurso 2:

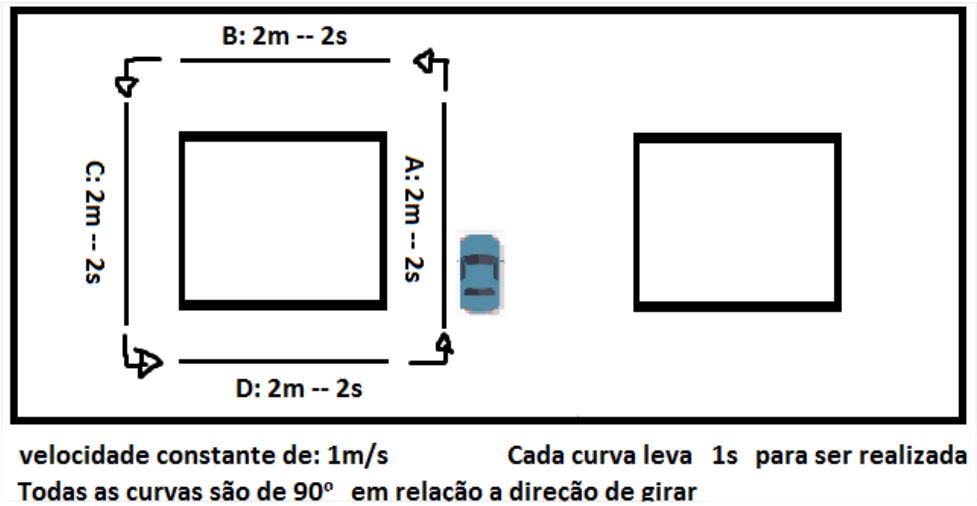
O roteiro e o circuito são mostrados na Figura 5 e 6 respectivamente.

Figura 5: roteiro



Fonte: Autoral

Figura 6: Circuito 2

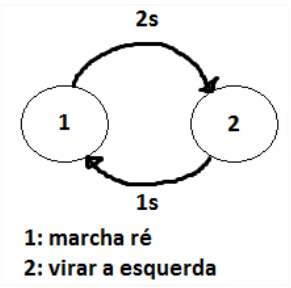


Fonte: Autoral

Percurso 3:

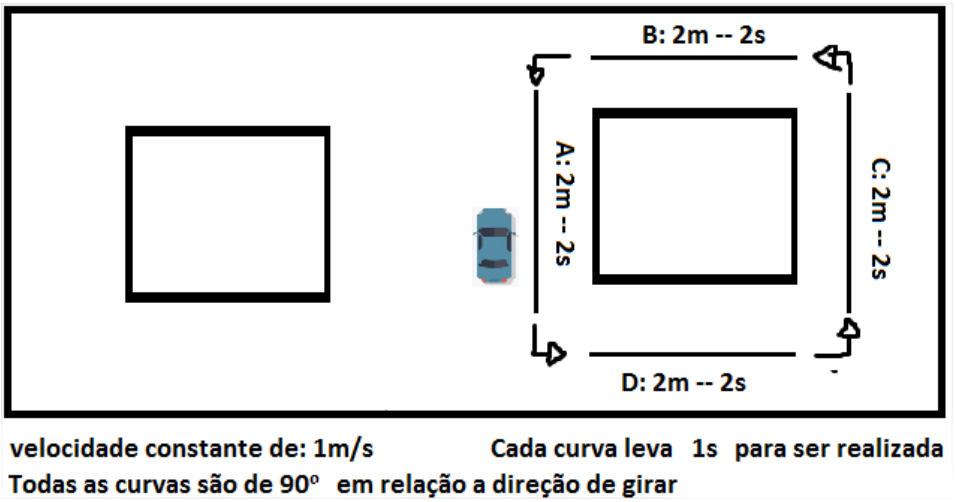
O roteiro e o circuito são mostrados na Figura 7 e 8 respectivamente.

Figura 7: roteiro



Fonte: Autoral

Figura 8: circuito 3



Fonte: Autoral

Análise sobre o escalonamento

Para o controlar o escalonamento das tasks (processos) foi utilizada a função “vTaskDelay()”. Pode-se assumir está função como sendo a deadline de cada tarefa, ela estende o tempo da tarefa em termos de milissegundos, de forma que ao fim desta contagem o processamento continua em frente de onde o código parou.

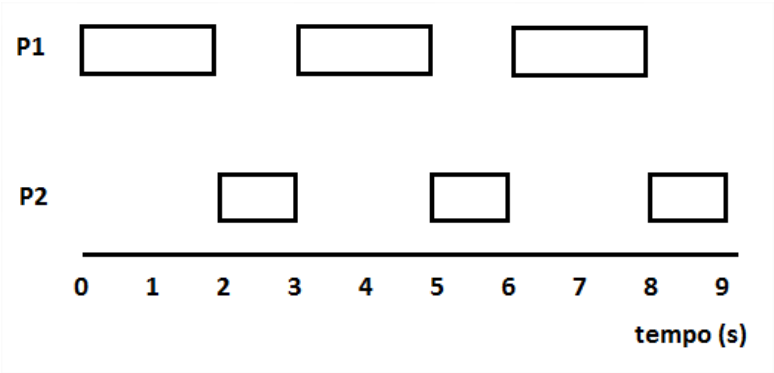
A princípio as tarefas são todas iniciadas ao mesmo tempo, sendo a que continua a processar é a de maior prioridade. A Figura 9 mostra um exemplo de escalonamento de tarefas, refere-se ao percurso 1, a Tabela 1 mostra os dados dos processors.

Tabela 1: processor percurso 1

Processos	Tempo (s)	deadline
P1 (ir em frente)	2	3
P2 (virar a direita)	1	3

Fonte: Autoral

Figura 9: exemplo de escalonamento



Fonte: Autoral

O processo se inicia com o processo P1, ele dura 2 segundos, logo após a deadline do processo P1 se inicia o processo P2. Desse modo o veículo continuara seguindo em frente e virando a direita, fazendo um movimento circular.

//OBS: OS CÓDIGOS DO PROGRAMA SE ENCONTRAM NA PASTA TRABALHO CARRO

REFERÊNCIAS

ARDUINO PROJECT HUB (Eua). **Using FreeRTOS multi-tasking in Arduino © GPL3+.** Disponível em: <https://create.arduino.cc/projecthub/feilipu/using-freertos-multi-tasking-in-arduino-ebc3cc>. Acesso em: 01 dez. 2019.