RASCUNHO Projeto 6

MAC0318 Introdução à Programação de Robôs Móveis

13 de Junho de 2018

1 Objetivo

Nesse exercício de programação vocês deverão montar um robô que brinque de pega-pega.

O objetivo é que o seu robô seja o menos pego, ou seja, que o número total de vezes que ele foi pego seja o menor entre todos.

2 Regras

Haverá um campo com diversos robôs (no mínimo dois). Em qualquer momento de jogo haverá apenas um pegador.

Ao iniciar o seu programa você será informado das dimensões do campo e das linhas que descrevem seus obstáculos. Você não deve invadir essas linhas e nem sair fora do campo. Caso uma dessas linhas senha ultrapassada, você deve ficar parado 5 segundos como penalidade.

Sempre que alguém for pego, deve ficar parado 5 segundos (tempo existentes para que os outros fujam). Se o seu robô esperar esses 5 segundos, você será penalizado.

No campo de pega-pega não haverá paredes. Cabe a você manter o seu robô dentro do mesmo.

3 Funcionamento

Todos robôs terão um identificador visual que contêm um id único. Esse marcador deve estar posicionado de modo que ele fique paralelo ao chão. Não pode estar amassado ou dobrado.

Haverá uma câmera conectada a um computador. Esse computador irá coordenar o jogo. Ele observará se todas as regras estão sendo obedecidas e informará aos participantes a posição de todos os robôs presentes no campo.

Cada participante deve se conectar a esse servidor e controlar o seu robô. Vocês podem utilizar um computador convencional e controlar o robô via bluetooth ou podem utilizar uma Raspberry Pi conectada diretamente ao robô. No diretório `player-lejos` você poderá ver um exemplo completo de como utilizar o bluetooth com o Lejos.

4 Penalidades

As penalidades acumuladas durante o jogo serão a seguintes, quem acumular menos ganha a competição:

Penalidade	Descrição
2	Sair do campo seu robô será recolocado manualmente por uma pessoa na posição mais perto da que ele saiu dentro do campo.
1	Ser pego
5	Não esperar os 5 segundos depois de ser pego
5	Não esperar os 5 segundos depois de passar por cima de um obstáculo. Após passar por cima de um obstáculo e esperar 5 segundos, ele terá 2 segundos para sair do mesmo.

5 Complicações

Felizmente, o a recepção dos dados da câmera e a identificação dos marcadores demandam um pouco de tempo. Esse atraso pode ser percebido executando o exemplo do `player-camera`. Você deve encontrar uma solução para mitigar esse atraso.

Você pode, por exemplo, diminuir a velocidade de reação dos do seu robô, assim a posição real dele estará mais próxima da informada pela câmera. Você também pode tentar prever o próximo estado que seu robô estrá depois de uma ordem de movimento.

6 Utilizando o servidor

Para se conectar ao servidor você deve instanciar um objeto do tipo `TagClient`. Para isso, deve fornecer o endereço IP da máquina que o servidor está sendo executado e a porta da aplicação.

Para se conectar ao servidor você deve utilizar `client.info(id, xs, ys, xtag, ytag)`. Onde:

- `id`, o número do marcador do seu robô. Se for um robô virtual, pode ser qualquer número inteiro.
- `xs, ys`, as dimensões do robô.
- `xtag, ytag`, posição do centro do marcador referente ao canto de trás esquerdo do robô. Veja a classe `Robot`.

Esses valores são necessários para descrever o retângulo que está em torno do robô. Esse retângulo deve ser proporcional ao tamanho real do robô. Na figura 1 abaixo temos um exemplo de marcador. A direção do marcador é definida pelos três quadrados pretos em suas bordas.

Você estará conectado no servidor em quanto a constante do objeto `quit` estar no estado `False`. Durante essa condição você pode perguntar sobre informações do campo, `getFieldInfo()`. Perguntar se o jogo está pausado ou não `getStatus()`. Pegar a informação de quem é o pegador atual, `getTagInfo` (esse método retorna apenas o id dele e o

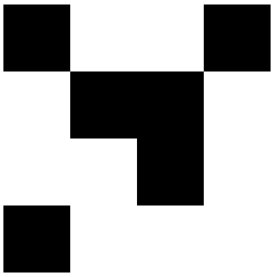


Figura 1: Note que a posição do marcador determina a orientação do robô. No marcador acima o robô está orientado para cima.

tempo de fuga restante). Ou receber um vetor com as últimas posições informadas pelo servidor (`getPosition()`). Nesse vetor haverá o id do robô, a sua posição e os valores `xs`,`ys`, `xtag` e `ytag`.

Para ver um exemplo funcional do código veja o arquivo `player-virtual`. Esse arquivo simula um jogador, no entanto o próprio código informa a posição do jogado. Você não deve fazer isso. Use como esqueleto do seu projeto o arquivo `player-camera` ou o `player-lejos`.

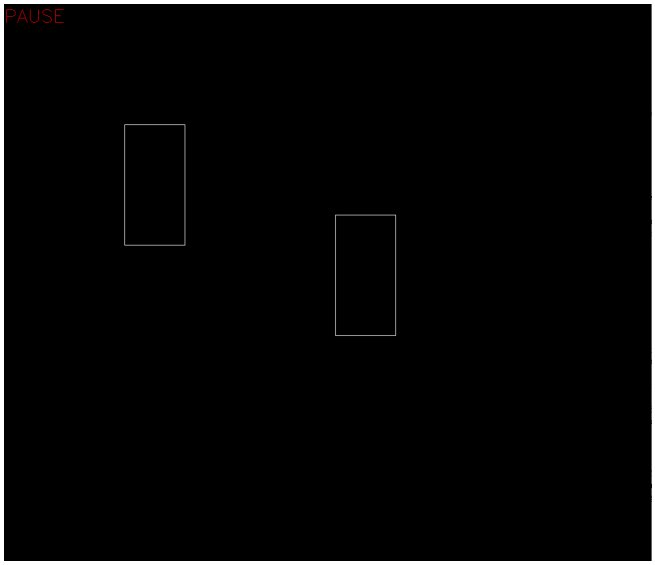
Para iniciar o servidor vocês devem executar o código `main.py`. Vocês podem rodar com a flag `-n` para que não haja a necessidade de câmera. Essa função serve para vocês testarem. Note que quando não há câmera, não há o atraso do processamento.

7 Simulação

Para executar o simulador você primeiramente deve iniciar o servidor sem câmera. Para isso execute o seguinte comando:

```
python3 main.py  -n
```

Feito isso, o servidor vai ser iniciado e uma tela como a seguinte será mostrada:



Depois disso você deve conectar o se robô ao servidor. Já temos um exemplo de robô pronto em `player-virtual`. Assim que o robô for conectado ele aparecerá na tela. Você pode conectar múltiplos robôs reais ou virtuais ao servidor, no entanto eles devem ter `id`s diferentes.

Para iniciar o jogo é necessário no servidor apertar a tecla `s`. Para pausar `p` e para sair `q`. Assim que o jogo iniciar, o pegador ficará marcado de vermelho.

8 Tarefas

Para concluir esse EP você deve montar um robô que consiga jogar. Você pode implementar qualquer estratégia.