Serviços Cognitivos- atividade discente supervisionada 1

Prof. Mozart Hasse

LEIA ATENTAMENTE TODAS AS INTRUÇÕES ATÉ O FINAL DA ÚLTIMA PÁGINA. CADA PALAVRA CONTA!

Seu objetivo é implementar o software embarcado para um robô de salvamento. O propósito do robô é resgatar seres humanos perdidos em labirintos. Os requisitos são os seguintes:

- O labirinto pode ter qualquer tamanho, sempre medido em unidades inteiras de igual largura e altura.
- Tanto o humano quanto o robô ocupam uma unidade de tamanho. Quando o robô coletar o humano à sua frente, ambos passarão a ocupar a mesma unidade de tamanho no caminho de volta.
- O robô e o humano são frágeis, por isso os comandos enviados para o robô JAMAIS podem fazê-lo colidir com uma parede, muito menos atropelar o humano (**FALTAS GRAVES!**).
- A equipe de qualidade exigiu que o robô gerasse um log auditável de operação, com as leituras dos sensores do momento em que o robô é ligado até a ejeção do humano.
- O registro do log deve ter uma linha para a leitura dos sensores ao ligar o dispositivo assim como uma linha para o resultado dos sensores após cada comando enviado aos atuadores.
- Pode assumir que SEMPRE será possível chegar até o humano a partir da entrada do labirinto (afinal, ele conseguiu chegar lá pela mesma entrada que o robô).
- A entrada do labirinto pode ficar em qualquer posição da borda, tendo sempre apenas UMA unidade de largura.

Atendendo aos requisitos de custo, o hardware do robô tem as seguintes limitações:

- Os sensores do robô só foram colocados na frente, no lado esquerdo e no lado direito.
- Cada sensor só pode identificar o que está na célula imediatamente na sua frente ou em um dos seus lados. Se um humano ou uma parede estiverem a 2 unidades de distância, não serão identificáveis pelos sensores.
- O hardware dos sensores só consegue identificar 3 coisas: humano, parede ou espaço vazio (para onde o robô pode se deslocar).
- Só foram implementados 4 comandos nos atuadores do robô:

A	Avançar para frente UMA POSIÇÃO
G	Girar o robô 90 graus à direita (ou seja, no sentido horário)
P	Pegar o humano imediatamente à frente do robô (só tem carga na bateria para UMA tentativa)
E	Ejetar humano para fora do labirinto (só tem carga na bateria para UMA tentativa)

A operação e o comportamento do robô serão os seguintes:

- O robô será colocado na única entrada, virado na direção do interior do labirinto, quando então será ativado.
- A partir daí, cabe ao software embarcado no robô cuidar de maneira totalmente autônoma da busca e salvamento, garantindo que o robô procurará em todos os cantos do labirinto até achar o humano.
- Após achar e coletar o humano, o robô DEVE retornar por um caminho eficiente que não o faça passar por nenhum beco sem saída. Não precisa ser o caminho mais curto, porém após a coleta do humano os 3 sensores do robô JAMAIS podem ver parede ao mesmo tempo, pois o humano resgatado provavelmente é claustrofóbico.
- Ao ser ativado na entrada do labirinto, o robô deve assumir que a saída está imediatamente atrás dele. O robô JAMAIS deve se deslocar para fora do labirinto, apenas realizará a ejeção do humano após tê-lo coletado e trazido até ficar de frente para a saída. Pode ser necessário memorizar o trajeto feito na busca para saber quando a saída foi encontrada, pois os sensores não conseguem diferenciar se o espaço vazio próximo ao robô está dentro ou fora do labirinto.

Considerando que o robô é uma agente inteligente, o hardware NÃO precisa saber nada além disso.

Para fazer os seus testes, alimente o ambiente virtual onde o robô será colocado com mapas no formato texto, com cada caractere representando uma posição e conforme a seguinte codificação:

H	humano
E	entrada
*	parede
Espaço em branco	área por onde o robô pode passar

O software embarcado deve gerar um arquivo de log em formato CSV (ou seja, valores separados por vírgulas) com o seguinte conteúdo:

Coluna	Valores possíveis
Comando enviado	LIGAR, A, G, P, E
Leitura do sensor do lado esquerdo do robô após execução do comando	PAREDE, VAZIO, HUMANO
Leitura do sensor do lado direito do robô após execução do comando	PAREDE, VAZIO, HUMANO
Leitura do sensor do lado direito do robô após execução do comando	PAREDE, VAZIO, HUMANO
Situação do compartimento de carga	SEM CARGA, COM HUMANO

Cada arquivo de log gerado deve ter o nome do arquivo de entrada correspondente com a extensão CSV.

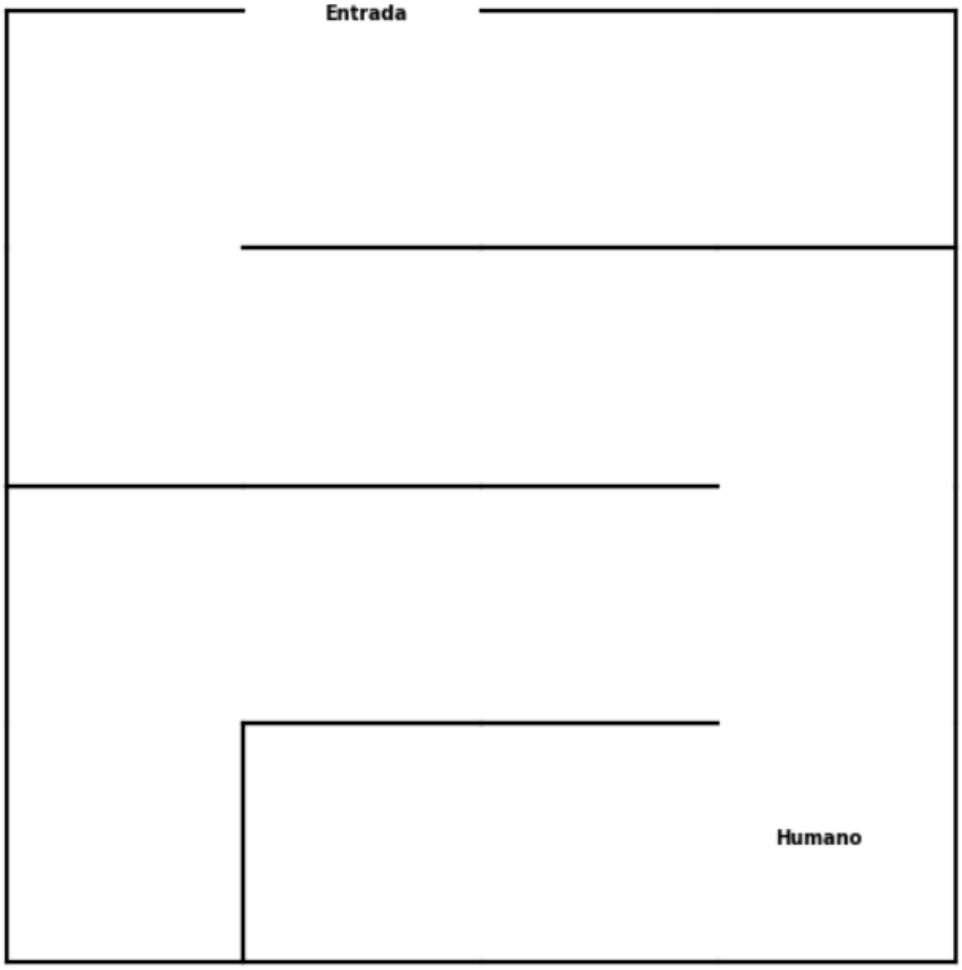
Sua entrega deve ter os seguintes componentes (mínimo um arquivo-fonte para cada):

- Simulador do ambiente virtual do robô (pode ser em texto, desde que atenda os requisitos).
- Implementação do hardware embarcado no robô, que deve ser acionado a partir do seu simulador para receber as leituras dos sensores a cada ação dos atuadores.
- Casos de teste próprios (mínimo 3).

Exemplo de arquivo de entrada e saídas esperadas

```
***E***
*     *
*  *****
*     *
*****  *
*     *
*  ***  *
*  *   H*
*******
```

Este texto representa a seguinte situação:

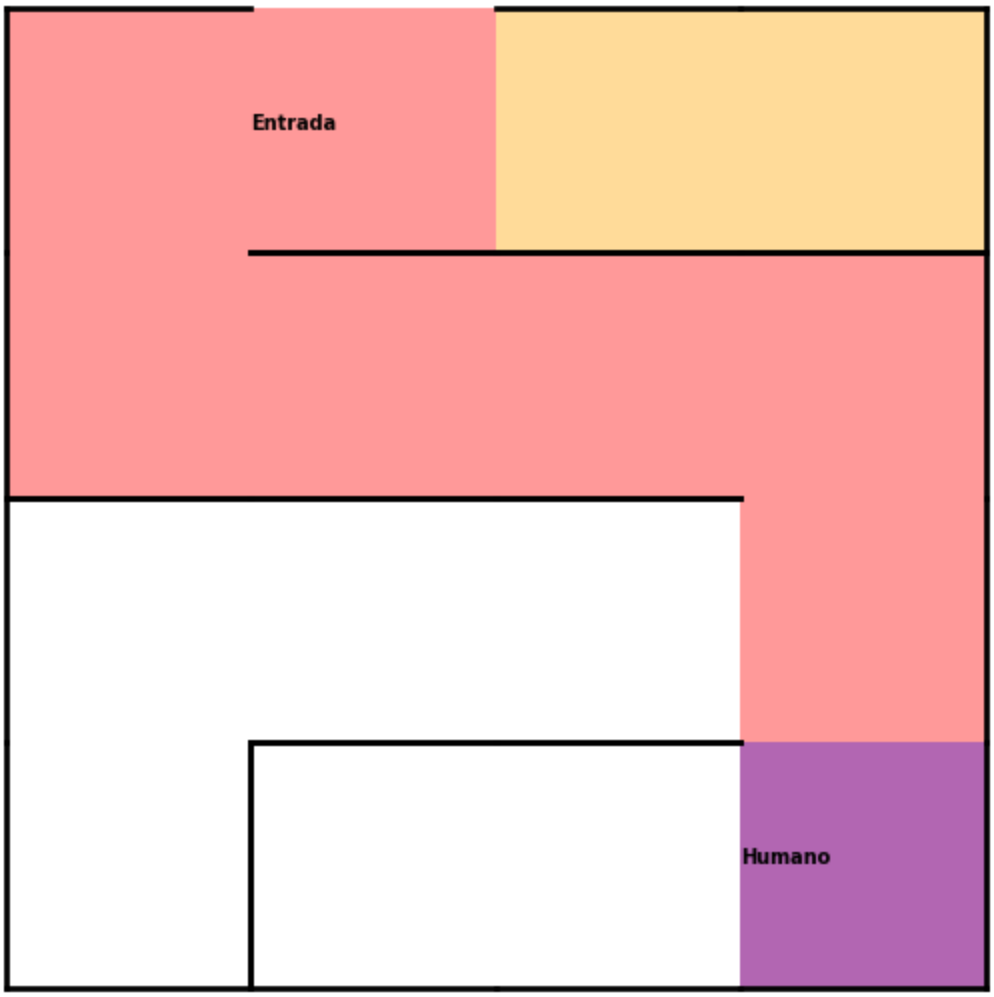


Um **exemplo** da sequência de comandos enviados aos atuadores do robô **poderia ser**:

ADDDAADDDAAADDADDDAAADAP

DDADDDAAADADDDE

O trajeto correspondente a este exemplo é demonstrado a seguir:



A seguir as primeiras linhas de log geradas para o exemplo acima, com o trecho em que o humano é coletado e o final com a ejeção do humano:

```
LIGAR,PAREDE,PAREDE,VAZIO,SEM CARGA
A,VAZIO,VAZIO,PAREDE,SEM CARGA
D,PAREDE,PAREDE,VAZIO,SEM CARGA
D,PAREDE,VAZIO,VAZIO,SEM CARGA
D,VAZIO,PAREDE,VAZIO,SEM CARGA
...
A,PAREDE,VAZIO,HUMANO,SEM CARGA
P,PAREDE,VAZIO,VAZIO,COM HUMANO
D,VAZIO,VAZIO,PAREDE,COM HUMANO
...
D,VAZIO,VAZIO,VAZIO,COM HUMANO
E,VAZIO,VAZIO,HUMANO,SEM CARGA
```

Cuidados na implementação

- Validações que DEVEM estar presentes no código:
    - Alarme de colisão com paredes.
    - Alarme de atropelamento de humano.
    - Alarme de beco sem saída após coleta de humano.
    - Alarme em caso de tentativa de ejeção sem humano presente.
    - Alarme em caso de tentativa de coleta sem humano à frente do robô.
- Mesmo que você tenha certeza que seu código não vai fazer nada errado, DEVEM haver mecanismos de checagem no seu código confirmando que o comportamento do robô sempre estará dentro de todos os parâmetros.
- Logs da execução do seu algoritmo, indicando as leituras dos sensores do robô ao ligar e após cada comando enviado aos atuadores para os casos fornecidos pelo professor.

Linguagens permitidas: Python ou C# (**em uma versão que GARANTIDAMENTE rode de maneira NATIVA no Linux, não me obrigue a instalar o Wine ou máquinas virtuais!**)

Seu algoritmo será avaliado pela qualidade e capacidade de generalização, portanto:

- O seu robô deve funcionar para labirintos maiores ou menores do que os fornecidos pelo professor, com entrada em qualquer ponto da borda que atenda aos critérios fornecidos (sugestão trivial: "espelhe" os arquivos fornecidos).
- O professor vai valorizar o esforço em buscar gerar um caso de teste para cada requisito. Um caso que merece atenção especial é para os casos de teste que cubram o código que dispara cada um dos alarmes (**SEGURANÇA EM PRIMEIRO LUGAR!**). Este código deve estar claramente identificado a ponto de mesmo um leigo na linguagem consiga encontrá-lo.
- Uma boa documentação é a que está presente no próprio código. Um README muito complicado denuncia um código mal organizado. Pior do que isso só a ausência de documentação ou comentários esparramados ao longo de todo o código.
- A correção é automatizada, portanto falha nos registros dos arquivos de logs será considerada uma falha séria.
- O percentual de cobertura do código pelos testes será levado em conta com bastante atenção.
- O professor reservou labirintos novos não acessíveis aos alunos para usar na correção. Portanto, recomenda-se que, além dos exemplos fornecidos, cada equipe gere seus próprios labirintos para teste.
- A nota será a soma dos requisitos corretamente atendidos conforme descrito neste documento.

Instruções para entrega

O trabalho deve ser entregue em UM arquivo em formato ZIP, enviado por apenas UM membro da equipe. Apenas o ÚLTIMO envio será considerado.

O trabalho DEVE conter o nome completo e matrícula de TODOS os integrantes. Erros ou omissões nesta parte serão considerados FALTAS GRAVES.

Recomenda-se a divisão do trabalho nas seguintes etapas/atividades, que são interdependentes mas podem ser feitas em paralelo:

- **Organização dos requisitos e casos de testes por ordem de complexidade;**
- **Definição da arquitetura das classes (INVISTAM TEMPO NISSO!);**
- **Montagem do simulador;**
- **Montagem de casos de teste para cobrir todos os requisitos do simulador;**
- **Montagem do código embarcado no robô;**
- **Montagem de casos de teste para cobrir todos os requisitos do software embarcado no robô;**
- **Peer-review do código;**
- **Validação do trilho de auditoria gerado pela aplicação.**

Observações gerais

O trabalho pode ser feito em equipes de até 4 alunos. A EQUIPE TODA É IGUALMENTE RESPONSÁVEL PELO SUCESSO DO TRABALHO E PELA IDENTIFICAÇÃO CORRETA E COMPLETA DE TODOS OS SEUS INTEGRANTES.

É TERMINANTEMENTE PROIBIDO compartilhar arquivos entre equipes, incluindo os casos de teste. Qualquer tentativa de fazer isso implicará na atribuição de nota ZERO a TODOS os membros de TODAS as equipes envolvidas.

Protótipo

Se não gostou, mande sua sugestão com o respectivo prompt e link da ferramenta generativa.