Relatorio 03 - Clarion: Controlando o WorldServer3D

Leonardo de Oliveira Ramos RA171941

May 2018

1 Entrega

Na pasta ClarionApp se encontra o código fonte da criatura. Para executar apenas abra a pasta bin e rode o script run.sh com o comando "sh ./run.sh" ou simplesmente "./run.sh".

2 Teste 01

Neste teste o comportamento da criatura programado é de ir até a joia do leaflet mais próxima no campo de visão dele, senão girar para observar outras joias, que potencialmente estão no leaftlet dele. E no caminho se ele passar por uma comida ele come e se passar por uma joia ele pega.

3 Teste 02

Em comparação, o comportamento da criatura com o SOAR está em vantagem, visto que ele tem conhecimento de todas as joias do ambiente, enquanto o do Clarion tem apenas o conhecimentos das joias em seu campo de visão. Além disso a criatura do SOAR, gananciosamente, vai atrás de comida e deixa a criatura do Clarion sem energia ao longo do tempo, visto que ela não vai atrás das comidas, apenas come as que seguem pelo caminho.

Dito isso eles tem comportamentos parecidos, muitas vezes competindo pelas joias mais próximas.

4 Funcionamento do código

Neste código as principais mudanças foram feitas nos arquivos MainClass.cs, ClarionAgent.cs e MindViewer.cs e foi criado um arquivo para gerar números aleatórios.

Primeiramente na Main eu criei um código para gerar joias aleatoriamente e alterei as paredes para ficar depois da área andável.

Após isso, no arquivo ClarionAgent.cs eu criei novas ações EAT_FOOD, GET_JEWEL E GO_TO_FOOD, que respondem a estímulos sensoriais com as keywords:

- foodAhead: que verifica se tem comida na frente da criatura
- jewelAhead: que verifica se tem joia na frente da criatura
- needFood: que verifica se há joias que estão no campo de visão e que fazem parte do leaflet

4.1 ClarionAgent.cs

A maior parte do código alterado fica neste arquivo, onde tiveram os seguintes métodos alterados:

- Construtor ClarionAgent: onde instanciou-se os possíveis inputs através dos dimension value pairs, e os outputs através dos external action chunks.
- ProcessSelectedAction: onde programei o que cada tipo de ação realmente faria.
- SetupACS: onde instanciei as FixedRules, SupportCalculators e deu commit nas FixedRules.
- PrepareSensoryInformation: onde através de querys se interpretou os valores do Proxy para usar como input sensório.
- Fixed Rules: para cada ação externa criei um método FixedRule.