Nome: Fabio Grassiotto

RA: 890441

Disciplina: IA941A, 1º S 2018

Aula 13 – LIDA: Controlando o WorldServer3D

**Objetivo**

Utilizar a arquitetura cognitiva LIDA para controlar uma criatura artificial no ambiente virtual WordServer3D (WS3D) através de implementação de um aplicativo utilizando linguagem de programação Java e modificação dos arquivos de configuração XML.

**Atividade 1**

Após realizar o download do código da aplicação base DemoLIDA, foi executada a sincronização com os códigos da biblioteca WS3DProxy e a aplicação WS3D. Foi estudado então o código da aplicação, notando o uso do código Java e a integração com os arquivos de configuração XML.

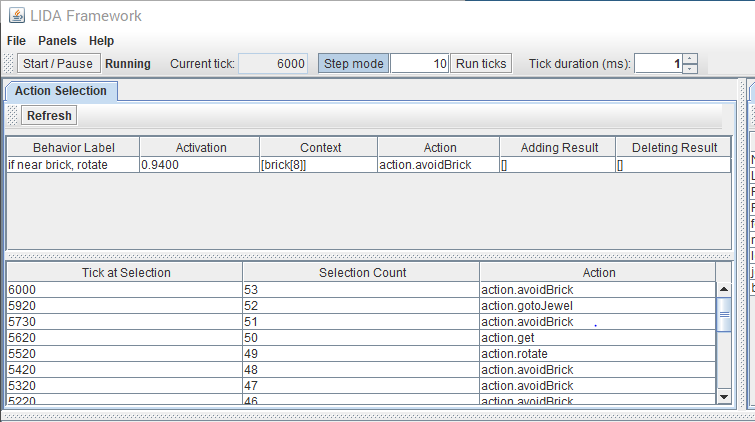
**Atividade 2**

Nesta segunda atividade foi proposta a modificação da estrutura dos painéis do Lida Framework para adicionar a visualização das ações a serem tomadas pela criatura a cada ciclo de execução.

Para tanto o código do arquivo guipanels.properties foi modificado, adicionando um novo painel na posição A (linhas 17 e 18) para efetuar a visualização conforme abaixo:

1. ###############################################################################
2. # Copyright (c) 2009, 2011 The University of Memphis.  All rights reserved.
3. # This program and the accompanying materials are made available
4. # under the terms of the LIDA Software Framework Non-Commercial License v1.0
5. # which accompanies this distribution, and is available at
6. # http://ccrg.cs.memphis.edu/assets/papers/2010/LIDA-framework-non-commercial-v1.0.pdf
7. ###############################################################################
9. #name = panel title, class name, Position [A,B,C,FLOAT, TOOL], tab order, Refresh after load
11. # \_\_\_\_Tool Bar\_\_\_\_
12. ToolBarToolBar=ToolBar,edu.memphis.ccrg.lida.framework.gui.panels.ControlToolBarPanel,TOOL,1,Y,1,20
14. # \_\_\_\_A Section\_\_\_\_
15. #environ =AlifeEnvironment,alifeagent.guipanels.ALifeGuiPanel,A,1,Y,configs/icons.properties,40
17. **# Action Selection Panel - Use A position.**
18. **actionPanel=Action Selection,edu.memphis.ccrg.lida.framework.gui.panels.ActionSelectionPanel,A,0,Y,20**
20. # \_\_\_\_B Section\_\_\_\_
21. pamTable = PAM Table,edu.memphis.ccrg.lida.framework.gui.panels.NodeStructureTable,B,0,Y,PerceptualAssociativeMemory
22. pamGraph = PAM Graph,edu.memphis.ccrg.lida.framework.gui.panels.NodeStructurePanel,B,1,Y,PerceptualAssociativeMemory
23. activationChart=Activation Chart,edu.memphis.ccrg.lida.framework.gui.panels.ActivationChartPanel,B,2,Y,100,leafletJewel
24. perceptualBufferGraph = Perceptual Buffer,edu.memphis.ccrg.lida.framework.gui.panels.NodeStructurePanel,B,3,Y,Workspace.PerceptualBuffer
25. globalWorkspace = Global Workspace,edu.memphis.ccrg.lida.framework.gui.panels.GlobalWorkspaceTablePanel,B,4,Y,10
26. proceduralMemory=Procedural Memory,edu.memphis.ccrg.lida.framework.gui.panels.ProceduralMemoryPanel,B,5,Y
27. #actionSelection=Action Selection,edu.memphis.ccrg.lida.framework.gui.panels.ActionSelectionPanel,B,6,Y,5
29. # \_\_\_\_C Section\_\_\_\_
30. LogPanel=Logging,edu.memphis.ccrg.lida.framework.gui.panels.LoggingPanel,C,1,N, alifeagent, edu.memphis.ccrg.alife
31. runningTasks = Running Tasks,edu.memphis.ccrg.lida.framework.gui.panels.FrameworkTaskPanel,C,2,Y, PerceptualAssociativeMemory
32. taskQueue=Task Queue,edu.memphis.ccrg.lida.framework.gui.panels.TaskQueuePanel,C,3,N
33. configFiles = Configuration Files,edu.memphis.ccrg.lida.framework.gui.panels.ConfigurationFilesPanel,C,4,N

O resultado na janela do Lida Framework pode ser visualizado na imagem:



**Atividade 3**

Nesta terceira atividade, foi proposta a alteração do código do DemoLIDA criando um mecanismo por meio do qual a criatura pudesse detectar blocos a partir do ambiente, e se movimentasse de uma origem até um destino sem colidir com os blocos.

Para execução das alterações são providos scripts nos sistemas operacionais Linux e Windows:

run-R4.sh / run-R4.bat

**Descrição da implementação**

Esta seção do relatório visa descrever as principais alterações necessárias ao código fonte do DemoLIDA.

1. **Alterações no código Java do DemoLIDA**

Para adicionar novos blocos no ambiente de simulação, o código do arquivo Environment.java foi modificado para inserir de forma aleatória alguns blocos no ambiente. Isso pode ser verificado no método Environment::init():

1. // Create some bricks around the environment.
2. Random rand = **new** Random();
3. **for** (**int** x = 0; x < World.getInstance().getEnvironmentWidth(); x = x + 100) {
4. **for** (**int** y = 0; y < World.getInstance().getEnvironmentHeight(); y = y + 100) {
6. // Valid bricks are at least at a distance of 50 units from the creature in
7. // the x and y axis. We do that so there are no problems for maneuvering around
8. // in the environment.
9. **if** (abs(crX - x) > 50 && abs(crY - y) > 50) {
10. // coordinates are ok. Discard randomly most of them so there is no overcrowding.
11. **int** r = rand.nextInt(4);
12. **if** (r == 0) {
13. World.createBrick(rand.nextInt(6), x, y, x + 10, y + 10);
14. }
15. }
16. }
17. }

Outras alterações foram necessárias no arquivo Environment.java para executar a detecção de blocos no ambiente de simulação e execução de ações para alterar a trajetória da criatura, nos métodos getstate(), updateEnvironment() e performAction(), conforme abaixo:

1. **public** Object getState(Map<String, ?> params) {
2. Object requestedObject = **null**;
3. String mode = (String) params.get("mode");
4. **switch** (mode) {
5. **case** "food":
6. requestedObject = food;
7. **break**;
8. **case** "jewel":
9. requestedObject = jewel;
10. **break**;
11. **case** "thingAhead":
12. requestedObject = thingAhead;
13. **break**;
14. **case** "leafletJewel":
15. requestedObject = leafletJewel;
16. **break**;
17. **case "brick":**
18. **requestedObject = brick;**
19. **break;**
20. **default:**
21. **break;**
22. }
23. **return** requestedObject;
24. }
26. **public** **void** updateEnvironment() {
27. creature.updateState();
28. food = **null**;
29. jewel = **null**;
30. leafletJewel = **null**;
31. thingAhead.clear();
32. brick = **null**;
34. **for** (Thing thing : creature.getThingsInVision()) {
35. **if (thing.getCategory() == Constants.categoryBRICK**
36. **&& creature.calculateDistanceTo(thing) <= DISTANCE\_TO\_BRICK) {**
37. **// Identifies we are close to a brick.**
38. **brick = thing;**
39. **break;**
40. } **else** **if** (creature.calculateDistanceTo(thing) <= Constants.OFFSET) {
41. // Identifica o objeto proximo
42. thingAhead.add(thing);
43. **break**;
44. } **else** **if** (thing.getCategory() == Constants.categoryJEWEL) {
45. **if** (leafletJewel == **null**) {
46. // Identifica se a joia esta no leaflet
47. **for** (Leaflet leaflet : creature.getLeaflets()) {
48. **if** (leaflet.ifInLeaflet(thing.getMaterial().getColorName())
49. && leaflet.getTotalNumberOfType(thing.getMaterial().getColorName()) > leaflet.getCollectedNumberOfType(thing.getMaterial().getColorName())) {
50. leafletJewel = thing;
51. **break**;
52. }
53. }
54. } **else** {
55. // Identifica a joia que nao esta no leaflet
56. jewel = thing;
57. }
58. } **else** **if** (food == **null** && creature.getFuel() <= 300.0
59. && (thing.getCategory() == Constants.categoryFOOD
60. || thing.getCategory() == Constants.categoryPFOOD
61. || thing.getCategory() == Constants.categoryNPFOOD)) {
63. // Identifica qualquer tipo de comida
64. food = thing;
65. }
66. }
67. }
69. @Override
70. **public** **void** processAction(Object action) {
71. String actionName = (String) action;
72. currentAction = actionName.substring(actionName.indexOf(".") + 1);
73. }
75. **private** **void** performAction(String currentAction) {
76. **try** {
77. //System.out.println("Action: " + currentAction);
78. **switch** (currentAction) {
79. **case** "rotate":
80. creature.rotate(1.0);
81. //CommandUtility.sendSetTurn(creature.getIndex(), -1.0, -1.0, 3.0);
82. **break**;
83. **case** "gotoFood":
84. **if** (food != **null**) {
85. creature.moveto(3.0, food.getX1(), food.getY1());
86. }
87. //CommandUtility.sendGoTo(creature.getIndex(), 3.0, 3.0, food.getX1(), food.getY1());
88. **break**;
89. **case** "gotoJewel":
90. **if** (leafletJewel != **null**) {
91. creature.moveto(3.0, leafletJewel.getX1(), leafletJewel.getY1());
92. }
93. //CommandUtility.sendGoTo(creature.getIndex(), 3.0, 3.0, leafletJewel.getX1(), leafletJewel.getY1());
94. **break**;
95. **case** "get":
96. creature.move(0.0, 0.0, 0.0);
97. //CommandUtility.sendSetTurn(creature.getIndex(), 0.0, 0.0, 0.0);
98. **if** (thingAhead != **null**) {
99. **for** (Thing thing : thingAhead) {
100. **if** (thing.getCategory() == Constants.categoryJEWEL) {
101. creature.putInSack(thing.getName());
102. } **else** **if** (thing.getCategory() == Constants.categoryFOOD || thing.getCategory() == Constants.categoryNPFOOD || thing.getCategory() == Constants.categoryPFOOD) {
103. creature.eatIt(thing.getName());
104. }
105. }
106. }
107. **this**.resetState();
108. **break**;
109. **case "avoidBrick":**
110. **if (brick != null) {**
111. **// The action here should be to manouver the creature to avoid the wall.**
113. **double crX = creature.getPosition().getX();**
114. **double crY = creature.getPosition().getY();**
116. **double targetX, targetY;**
118. **// Check coordinates to drive the creature around the bricks in the environment.**
119. **if (crY >= brick.getY2() || (crY >= brick.getY1() && crY <= brick.getY2())) {**
120. **// creature is below the brick.**
121. **// manouver from under it.**
122. **targetY = brick.getY2() + BRICK\_MANOUVER\_DIST;**
123. **} else {**
124. **// creature is above the brick.**
125. **targetY = brick.getY1() - BRICK\_MANOUVER\_DIST;**
126. **}**
128. **if (crX >= brick.getX2() || (crX >= brick.getX1() && crX <= brick.getX2())) {**
129. **targetX = brick.getX2() + BRICK\_MANOUVER\_DIST;**
130. **} else {**
131. **targetX = brick.getX1() - BRICK\_MANOUVER\_DIST;**
132. **}**
134. **creature.moveto(3.0, targetX, targetY);**
135. **}**
136. **break;**
137. **default**:
138. **break**;
139. }
140. } **catch** (Exception e) {
141. e.printStackTrace();
142. }
143. }
144. }

Finalmente, foi adicionada a implementação de um detector adicional para os blocos do ambiente, no arquivo BrickDetector.java.

1. **package** detectors;
3. **import** java.util.HashMap;
4. **import** java.util.Map;
6. **import** edu.memphis.ccrg.lida.pam.tasks.BasicDetectionAlgorithm;
7. **import** ws3dproxy.model.Thing;
9. **public** **class** BrickDetector **extends** BasicDetectionAlgorithm {
11. **private** **final** String modality = "";
12. **private** Map<String, Object> detectorParams = **new** HashMap<>();
14. @Override
15. **public** **void** init() {
16. **super**.init();
17. detectorParams.put("mode", "brick");
18. }
20. @Override
21. **public** **double** detect() {
22. Thing brick = (Thing) sensoryMemory.getSensoryContent(modality, detectorParams);
23. **double** activation = 0.0;
24. **if** (brick != **null**) {
25. activation = 1.0;
26. }
27. **return** activation;
28. }
29. }
30. **Alterações nos arquivos de configuração XML**

Para implementação da detecção dos blocos no ambiente, foi adicionada ao arquivo **factory.xml** uma task inicial extra para inicializar o detector de blocos:

1. **<task** name="BrickDetector"**>**
2. **<class>**detectors.BrickDetector**</class>**
3. **<ticksperrun>**3**</ticksperrun>**
4. **<associatedmodule>**SensoryMemory**</associatedmodule>**
5. **<associatedmodule>**PerceptualAssociativeMemory**</associatedmodule>**
6. **</task>**

No arquivo de definição do agente, Agent.xml, as seguintes alterações foram realizadas:

* Adição de uma nova task no módulo de memória perceptual associativa (*PerceptualAssociativeMemory)* e novo node para detecção de bloco (*brick)* no ambiente:

1. **<param** name="nodes"**>**
2. food,nearObject,leafletJewel,jewel,brick
3. **</param>**
4. **<task** name="BrickDetector"**>**
5. **<tasktype>**BrickDetector**</tasktype>**
6. **<ticksperrun>**3**</ticksperrun>**
7. **<param** name="node" type="string"**>**brick**</param>**
8. **<param** name="learnable.baseLevelActivation" type="double"**>**0.0**</param>**
9. <**param** name="learnable.baseLevelRemovalThreshold" type="double"**>**-1.0**</param>**
10. **<param** name="learnable.baseLevelDecayStrategy" type="string"**>**defaultDecay**</param>**
11. **<param** name="learnable.baseLevelExciteStrategy" type="string"**>**defaultExcite**</param>**
12. **<param** name="learnable.totalActivationStrategy" type="string"**>**DefaultTotalActivationStrategy**</param>**
13. **</task>**

* No módulo de atenção (*Attention Module)*, um novo codelet foi adicionado para trazer para a consciência o evento de detecção de um bloco:

1. **<task** name="BrickAttentionCodelet"**>**
2. **<tasktype>**BasicAttentionCodelet**</tasktype>**
3. **<ticksperrun>**5**</ticksperrun>**
4. **<param** name="nodes" type="string"**>**brick**</param>**
5. **<param** name="refractoryPeriod" type="int"**>**50**</param>**
6. **<param** name="initialActivation" type="double"**>**1.0**</param>**
7. **<param** name="learnable.baseLevelActivation" type="double"**>**1.0**</param>**
8. **<param** name="learnable.baseLevelRemovalThreshold" type="double"**>**-1.0**</param>**
9. **<param** name="learnable.baseLevelDecayStrategy" type="string"**>**defaultDecay**</param>**
10. **<param** name="learnable.baseLevelExciteStrategy" type="string"**>**defaultExcite**</param>**
11. **<param** name="learnable.totalActivationStrategy" type="string"**>**DefaultTotalActivationStrategy**</param>**
12. **</task>**

* No módulo de memória procedural, foi adicionada a ação a ser tomada quando da detecção de um bloco no ambiente. Procurou-se ajustar o nível base de detecção de forma a privilegiar a ação de se desviar de blocos:

1. **<module** name="ProceduralMemory"**>**
2. **<class>**edu.memphis.ccrg.lida.proceduralmemory.ProceduralMemoryImpl**</class>**
3. **<param** name="proceduralMemory.ticksPerStep" type="int"**>**14 **</param>**
4. **<param** name="proceduralMemory.conditionDecayStrategy"**>**defaultDecay**</param>**
5. **<param** name="proceduralMemory.schemeSelectionThreshold" type="double"**>**0.1**</param>**
6. **<param** name="proceduralMemory.contextWeight" type="double"**>**1.0**</param>**
7. **<param** name="proceduralMemory.addingListWeight"  type="double"**>**1.0**</param>**
8. **<param** name="proceduralMemory.schemeClass"**>**edu.memphis.ccrg.lida.proceduralmemory.SchemeImpl**</param>**
9. <!--   \_\_\_\_scheme\_name\_\_ \_\_\_\_\_ scheme\_label\_\_\_\_\_\_\_|\_context\_\_\_\_\_\_|\_\_action\_name\_\_\_\_|result|baseLabelActiv. -->
10. **<param** name="scheme.1"**>**if no leaflet jewel, rotate|(jewel)()|action.rotate|()()|0.3**</param>**
11. **<param** name="scheme.2"**>**if leaflet jewel, goto jewel|(leafletJewel)()|action.gotoJewel|()()|0.3**</param>**
12. **<param** name="scheme.3"**>**if food, goto food|(food)()|action.gotoFood|()()|0.3**</param>**
13. **<param** name="scheme.4"**>**if nearObject, get object|(nearObject)()|action.get|()()|0.01**</param>**
14. **<param** name="scheme.5"**>**if near brick, rotate|(brick)()|action.avoidBrick|()()|0.2**</param>**
15. **<taskspawner>**defaultTS**</taskspawner>**
16. **<initializerclass>**edu.memphis.ccrg.lida.proceduralmemory.BasicProceduralMemoryInitializer**</initializerclass>**
17. **</module>**

**Resultados**

Notou-se que com as alterações implementadas o agente consegue selecionar a ação de desviar dos blocos quando necessário e a coleta das jóias no ambiente foi realizada com sucesso.