Nome: Fabio Grassiotto
RA: 890441
Disciplina: IA941A, 1º S 2018

Aula 15 – CST: Controlando o WorldServer3D

Objetivo

Utilizar o toolkit CST para controlar uma criatura artificial no ambiente virtual WordServer3D (WS3D) através de implementação de um aplicativo utilizando linguagem de programação Java.

Atividade 1

Nesta primeira atividade, foi proposta a alteração do código do WS3DApp, criando um mecanismo para a busca de cristais e alimentos.

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

run-R5-A1.sh / run-R5-A1.bat

Descrição da implementação

Esta seção do relatório visa descrever as principais alterações necessárias ao código fonte do WS3DApp. Para a implementação desta primeira atividade, o código da WS3DApp foi alterado para adicionar novos Memory Objects e Codelets. O diagrama Figura 1 ilustra o relacionamento dos memory objects e codelets desta solução.

1. Memory Objects

- Memória Sensorial:

visionMO: Para acesso aos objetos na visão da criatura.

innerSenseMO: Para acesso aos dados de posicionamento e combustível da criatura.

- Memória Perceptual:

knownFoodsMO: contém a lista de todos os objetos de ítens de comida no campo visual da criatura.

closestFoodMO: contém o ítem da lista acima que se encontra mais próximo da criatura.

knownJewelsMO: contém a lista de jóias no campo visual da criatura.

leafletJewelMO: contém o ítem da lista acima que se encontra mais próximo da criatura e que faz parte dos requisitos dos leaflets.

closestObstacleMO: contém o objeto no ambiente que se encontra mais próximo da criatura.

leafletsDoneMO: indica que a criatura já coletou todas as jóias necessárias de acordo com as especificações dos leaflets.

atDeliverySpotMO: indica que a criatura está na posição do Delivery Spot.

- Memória Motora:

legsMO: Para execução da movimentação da criatura.

handsMO: Para execução de ações de coleta de jóias e consumo de comida.

2. Memory Container

Devido ao fato que mais de uma decisão relativa à movimentação da criatura ser possível, foi implementado um Memory Container (*legsDecisionMC*) para arbitrar qual conteúdo de memory object é escrito na memória motora de acordo com ítens de avaliação.

3. Codelets

- Codelets Sensoriais

Sem alterações em relação ao aplicativo base, utilizando *Inner Sense* e *Vision* para detectar o estado corrente da criatura e os objetos no campo de visão.

- Codelets Perceptuais

Food Detector: Codelet que analisa o ambiente e carrega o memory object com a lista dos ítens de comida.

Closest Food Detector: Codelet que determina qual é o ítem de comida mais próximo da criatura.

Jewel Detector: Codelet que analisa o ambiente e carrega o memory object com a lista das jóias do ambiente.

Leaflet Jewel Detector: Codelet que determina qual é a próxima jóia que a criature deve procurar no ambiente de acordo com a especificação dos leaflets.

Closest Obstacle Detector: Codelet que analisa o ambiente e carrega o memory object com o objeto mais próximo da criatura.

Leaflets Done Detector: Codelet que analisa se todas as jóias do leaflet foram coletadas.

At Delivery Spot Detector: Codelet que analisa se a criatura está nas proximidades do Delivery Spot.

- Codelets Comportamentais:

Go To Closest Food: Codelet que instrui a criatura a se movimentar na direção do ítem de comida mais próximo utilizando a arquitetura de subsumption, no caso do combustível da criatura estar com um nível inferior a 40% do total.

Go To Leaflet Jewel: Codelet que instrui a criatura a se movimentar na direção da jóia do leaflet utilizando a arquitetura de subsumption.

Go To Delivery Spot: Codelet que instrui a criatura a se movimentar na direção do Delivery Spot.

Eat Closest Food: Codelet que instrui a criatura a consumir um ítem de comida.

Pickup Jewel: Codelet que instrui a criatura a coletar uma jóia do ambiente conforme especificado nos leaflets.

Remove Obstacle: Codelet que detecta quando um obstáculo está próximo da criatura, e determina sua remoção se não for uma jóia dos leaflets.

Deliver Leaflets: Codelet que executa a entrega das jóias dos leaflets.

Wander: Codelet que instrui a criatura a girar no próprio eixo quando jóias do leaflet ou ítens de comida não estiverem presentes no ambiente.

- Codelets Motores:

Foi apenas alterado o codelet de movimentação das pernas, para utilizar a saída do Memory Container.

Resultados

Notou-se que com as alterações implementadas o agente consegue coletar as jóias do ambiente e entregá-las na posição do Delivery Spot.

Atividade 2

Nesta segunda atividade, foi proposta a alteração do código do WS3DApp, 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-R5-A2.sh / run-R5-A2.bat

Descrição da implementação

Esta seção do relatório visa descrever as principais alterações necessárias ao código fonte do WS3DApp. De forma similar à primeira atividade, o código da WS3DApp foi alterado para adicionar novos Memory Objects, Codelets e uma solução de roteamento em grid. O diagrama Figura 2 ilustra o relacionamento dos memory objects e codelets desta solução.

1. Memory Objects

- Memória Sensorial:

visionMO: Para acesso aos objetos na visão da criatura.

innerSenseMO: Para acesso aos dados de posicionamento e combustível da criatura.

- Memória Perceptual

BrickListMO: contém a lista de todos os tijolos no campo visual da criatura.

ClosestBrickMO: contém o tijolo da lista acima mais próximo da criatura.

CloseBrickFoundMO: indica que um tijolo foi encontrado a poucas unidades de distância da criatura.

- Memória Motora:

legsMO: Para execução da movimentação da criatura.

2. Codelets

- Codelets Sensoriais

Sem alterações em relação ao aplicativo base, utilizando *Inner Sense* e *Vision* para detectar o estado corrente da criatura e os objetos no campo de visão.

- Codelets Perceptuais

BrickDetector: Codelet que analisa o ambiente e carrega o memory object com a lista dos tijolos.

ClosestBrickDetector: Codelet que determina qual é o tijolo mais próximo da criatura.

- Codelets Comportamentais

Go To Destination: Codelet que instrui a criatura a se movimentar no ambiente evitando a colisão com os tijolos. Para tanto é utilizada uma biblioteca de roteamento em grid bidimensional.

AvoidBrick: Codelet que ao encontrar um tijolo muito próximo à criatura, provoca o replanejamento do trajeto a ser seguido pela criatiura.

4. Biblioteca de Roteamento em grid

Uma nova classe no arquivo PathFinder.java foi criada para utilizar a biblioteca GridNav-java (*https://github.com/elnygren/GridNav-java*) para executar o roteamento da criatura no ambiente até o novo destino. Essa biblioteca fornece pathfinding usando alguns algoritmos distintos. Para esta solução foi selecionado o A-Star.

Resultados

Notou-se que com as alterações implementadas o agente consegue selecionar a ação de desviar dos blocos quando necessário.

Possíveis Melhorias

- O algoritmo de roteamento da criatura empregado opera em um grid. Por vezes foi possível notar que não foi possível encontrar um caminho. Esse algoritmo poderia ser melhorado ou outra biblioteca poderia ser utilizada.

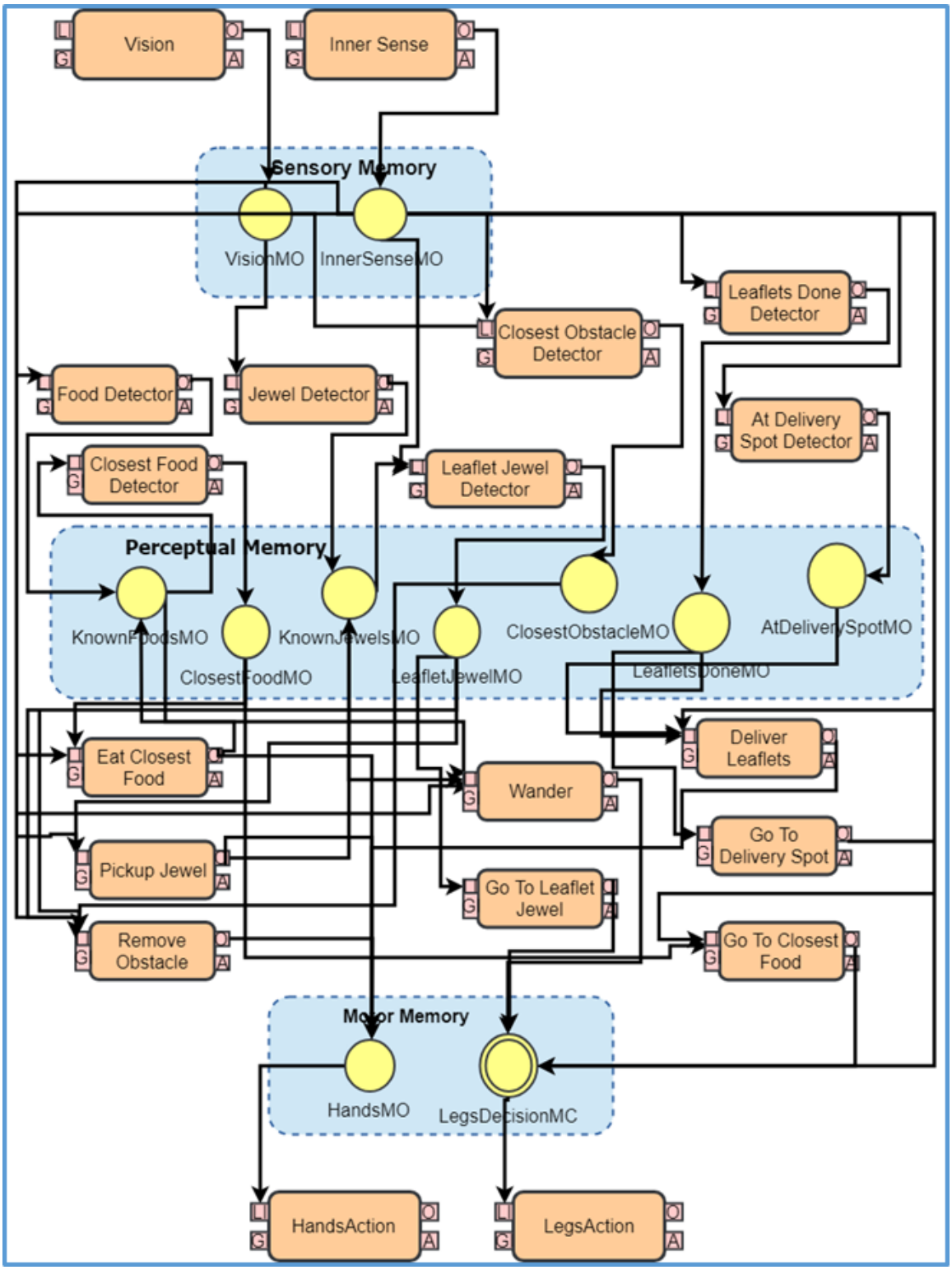
Vision
Inner Sense
Sensory Memory
VisionMO
InnerSenseMO
Leaflets Done Detector
Closest Obstacle Detector
Food Detector
Jewel Detector
At Delivery Spot Detector
Closest Food Detector
Leaflet Jewel Detector
Perceptual Memory
KnownFoodsMO
KnownJewelsMO
ClosestObstacleMO
AtDeliverySpotMO
ClosestFoodMO
LeafletJewelMO
LeafletsDoneMO
Deliver Leaflets
Eat Closest Food
Wander
Go To Delivery Spot
Pickup Jewel
Go To Leaflet Jewel
Remove Obstacle
Go To Closest Food
Motor Memory
HandsMO
LegsDecisionMC
HandsAction
LegsAction

Figura 1

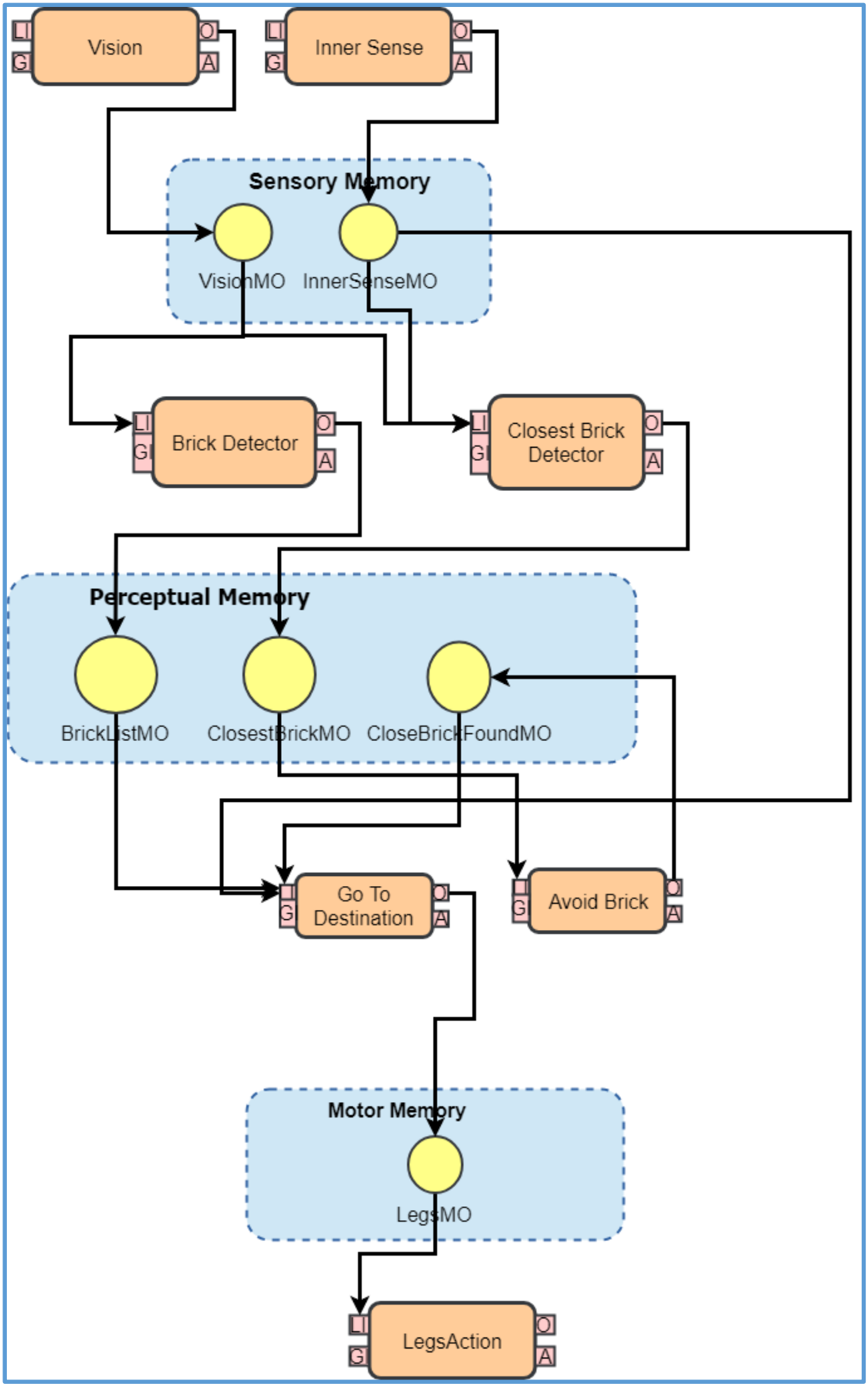
Vision
Inner Sense
Sensory Memory
VisionMO
InnerSenseMO
Brick Detector
Closest Brick Detector
Perceptual Memory
BrickListMO
ClosestBrickMO
CloseBrickFoundMO
Go To Destination
Avoid Brick
Motor Memory
LegsMO
LegsAction

Figura 2