U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

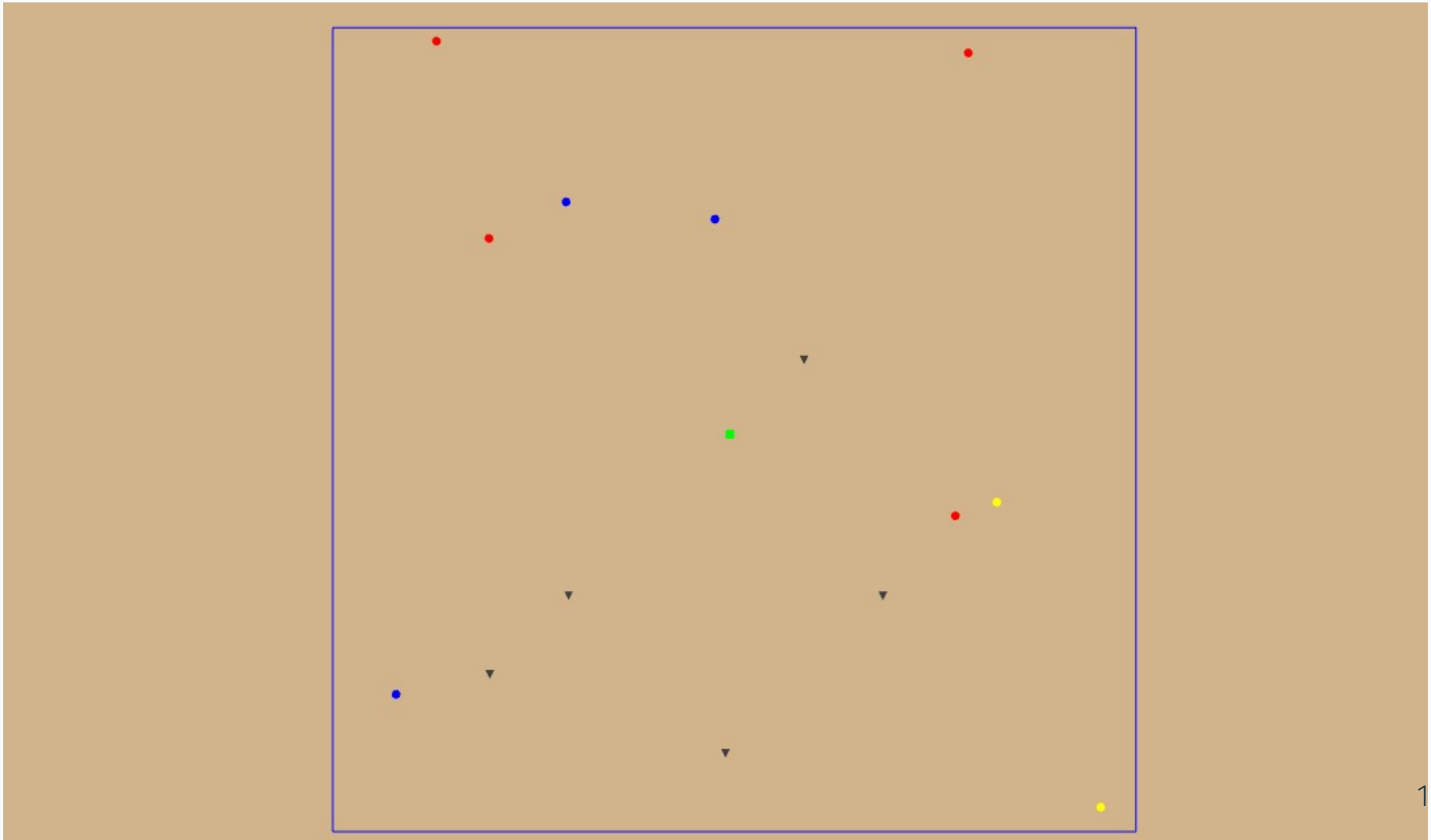
AGENTES E INTELIGÊNCIA ARTIFICIAL DISTRIBUÍDA

Planet Explorer - Informação adicional

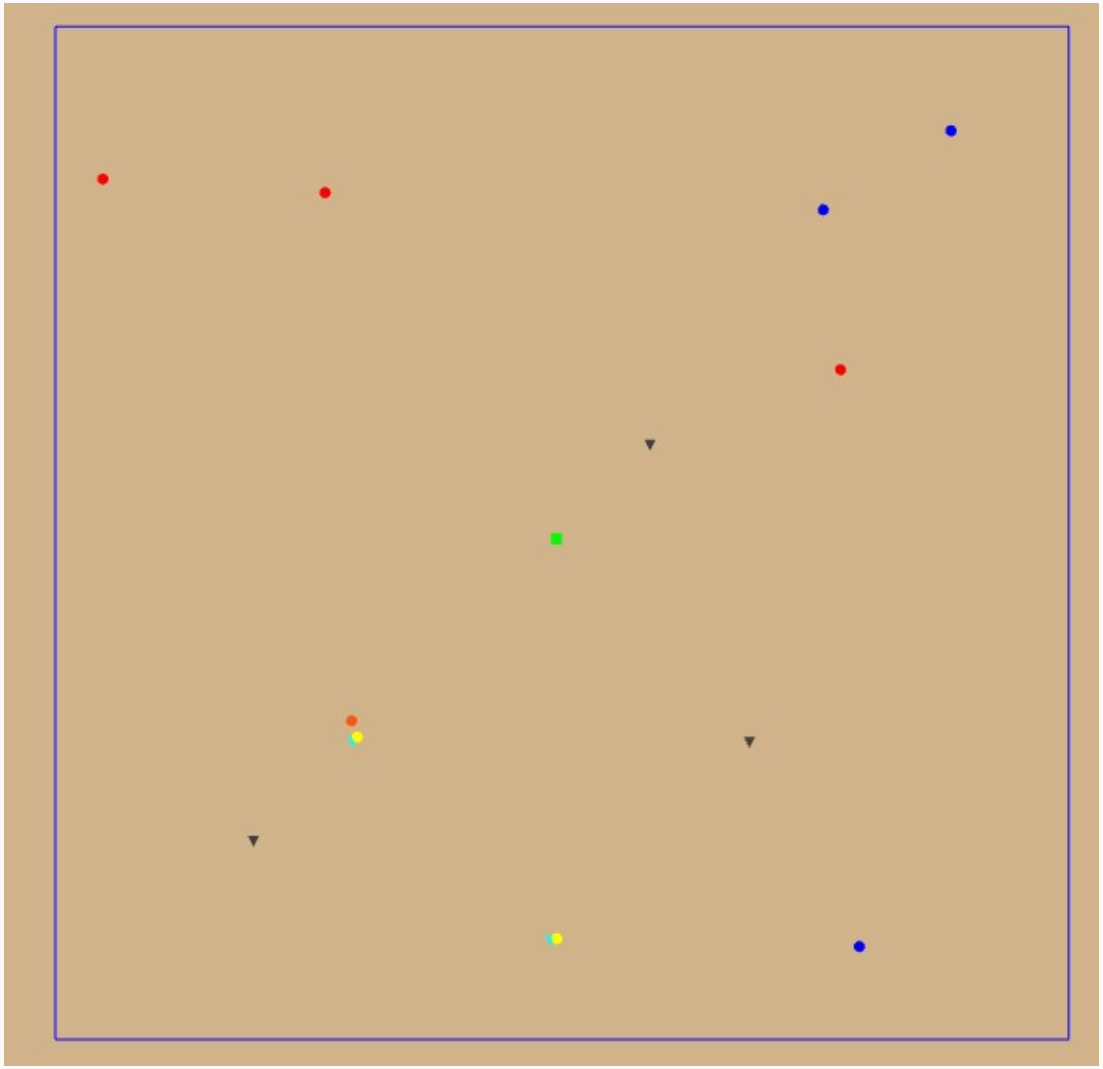
Nuno Marques
Gonçalo Oliveira
Ricardo Ferreira

13 de dezembro de 2020

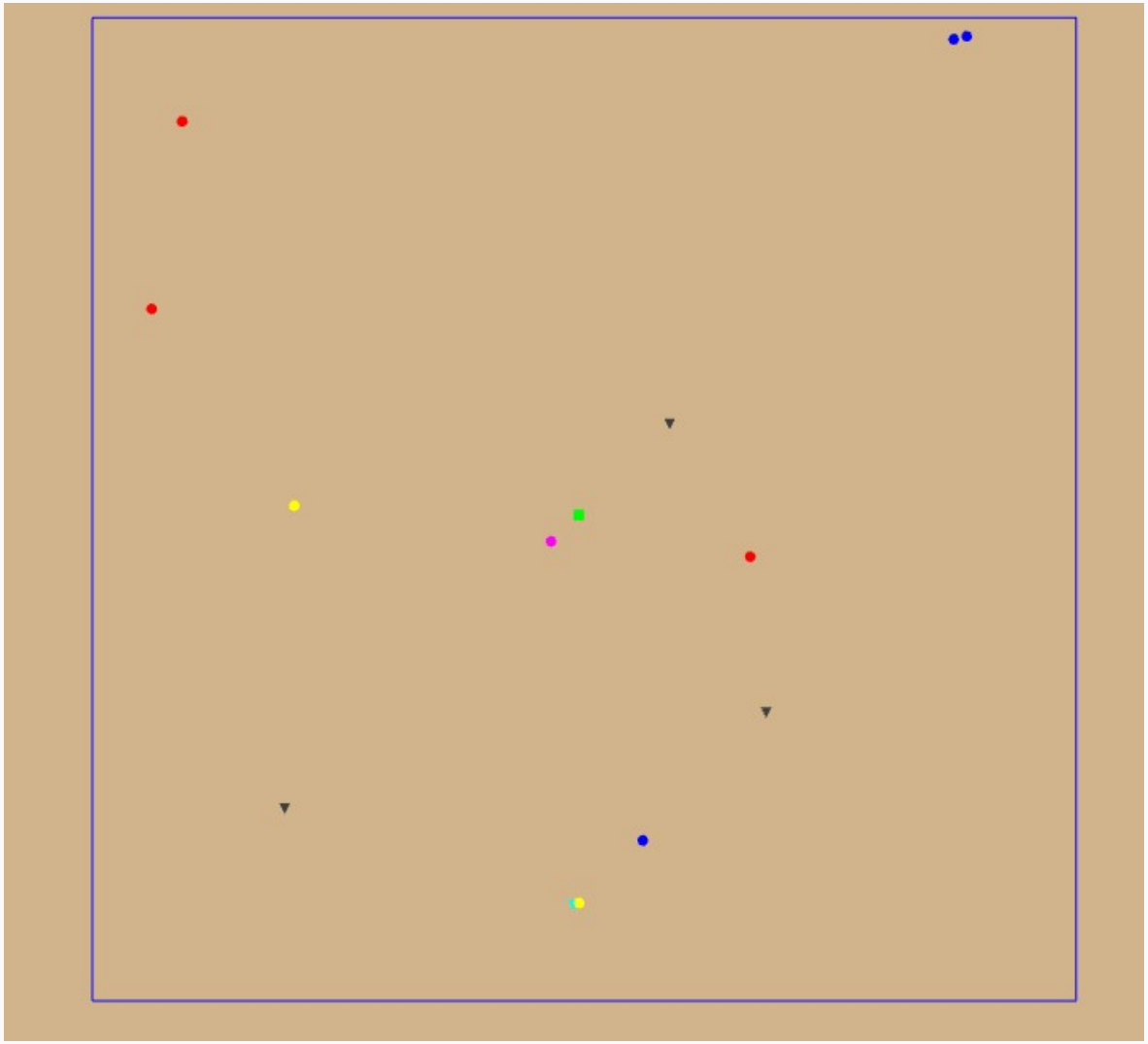
Exemplos detalhados de execução



Exemplos detalhados de execução



Exemplos detalhados de execução



Exemplos detalhados de execução

Na *GUI* estão representadas as seguintes entidades:

- **Oval vermelho**: Representa um transportador sem contrato
- **Oval laranja**: Representa um transportador com contrato a caminho de um coletor
- **Oval magenta**: Representa um transportador com contrato a entregar recursos à base
- **Oval azul**: O oval azul representa o explorador e o circulo exterior amarelo representa o alcance dos sensores do explorador
- **Oval amarelo**: Representa um colector
- **Triângulo cinzento**: Representa um recurso que ainda não foi descoberto por um explorador
- **Triângulo ciano**: Representa um recurso que já foi descoberto por um explorador
- **Quadrado verde**: Representa a localização da base
- **Linhas pretas**: Representam os limites da exploração

Exemplos detalhados de execução

```
Header
210 180 140
50 50
Base
25 20
Explorers
50 50
50 50
50 50
Transporters
30 30
40 50
60 30
Collectors
50 20
Resources
20 20 100
20 10 70
20 30 80
20 40 50
```

(a) Exemplo de um mapa

```
planet explorer log
Total Time: 35.57 seconds
TPAgent0    80
TPAgent1    100
TPAgent2    120
Collector0  300
```

(b) Exemplo de output final

Os mapas contêm os limites de exploração e quantidade e posições iniciais das entidades. O output mostra a quantidade de recursos obtida por cada colector e a quantidade transportada por cada transportador.

Exemplos detalhados de execução

Agente Base

- É o agente responsável por coordenar todas as operações. Recebe e envia mensagens para todos os agentes.
- Todas as mensagens com o *perfomative inform* que recebe (ACL Messages enviadas por outros agentes) são enviadas em *broadcasts* para todos os agentes.
- Ao receber um pedido de transporte por um coletor, o agente inicia uma oferta de contrato através da classe *ContractNetInitiator*. Após receber propostas, decide o transportador que vai executar a tarefa.

Exemplos detalhados de execução

Agente Explorer

- Este agente explora o mapa movendo-se em direcções aleatórias até encontrar um recurso.
- Envia para a base a mensagem *[FOUND XX YY]* indicando que encontrou um recurso nas coordenadas XX e YY.
- Recebe da base mensagens com o mesmo formato que envia o que indica que outros exploradores encontraram um recurso e como tal já não precisa de sinalizar o recurso.

Exemplos detalhados de execução

Agente Collector

- Este agente move-se em direções aleatórias até receber uma mensagem [FOUND XX YY] da base, indicando que foi encontrado um recurso na posição (XX, YY). Esta mensagem causa o movimento do colector até ao recurso.
- Uma vez chegado ao recurso, este agente, começa a minerá-lo e envia mensagens [MINING XX YY] para a base que são depois enviadas para todos os outros colectores para que estes ignorem o recurso e voltem à patrulha, pois o recurso na posição (XX, YY) já está a ser minerado.
- Quando o recurso estiver minerado, o agente envia uma mensagem [RETRIEVE XX YY] para a base que representa um pedido de transporte do recurso minerado para a base. Quando o transportador chegar à posição (XX, YY), este envia uma mensagem [TRANSPORT XX YY], o recurso é entregue ao transportador e o colector volta a patrulhar o mapa.

Exemplos detalhados de execução

Agente Transporter

- Este agente explora o mapa movendo-se em direcções aleatórias até receber uma oferta de contrato.
- Ao receber uma oferta de contrato, se não tiver um contrato ativo e conseguir cumprir o contrato, envia uma proposta para a base com a distancia até ao recurso.
- Se a proposta for aceite o agente recolhe os recursos no contrato e depois, se não tiver outro contrato disponível e aceite, vai entregar os recursos à base e volta ao primeiro comportamento.

Geração de dados

Para além dos gráficos e logs é também possível obter resultados através de um file sink detalhado com o valor das variáveis dependentes em cada tick.

```
"tick","tp_carrying","cl_mined"
1.0,0.0,0.0
101.0,0.0,0.0
201.0,0.0,0.0
301.0,0.0,0.0
401.0,0.0,0.0
501.0,0.0,0.0
601.0,0.0,0.0
701.0,0.0,0.0
801.0,0.0,0.0
901.0,0.0,0.0
...
2089201.0,100.0,280.0
...
2945901.0,70.0,350.0
```

Outras observações

Foram implementadas as seguintes classes:

- **BaseAgent**: Agente responsável por coordenar as operações dos outros agentes
- **ExplorerAgent**: Agente que irá procurar recursos no mapa e enviar a sua localização para a base.
- **CollectorAgent**: Agente que irá ao local dos recursos para os recolher e envia mensagem para a base quando os recursos estão prontos para serem recolhidos
- **TransporterAgent**: Agente que irá ao local onde os recursos já foram colectados para os transportar para a base.
- **DrivingBehaviour**: Este comportamento é partilhado por todos os agentes que se movem no mapa e define as propriedades para um agente se mover de um ponto para outro.
- **Map**: Esta classe cria o mapa e a sua representação e guarda uma lista dos recursos disponiveis.
- **Resource**: Representa um recurso com a respectiva localização e quantidade.
- **Vec2**: Classe utilizada para a representação das coordenadas no espaço 2D e com funções utilitárias tais como o calculo de distância entre dois pontos.