# L3 informatique, 2024-2025, Projet de LFC

# Création d'un langage de script pour un système multi-agents

# ETAPE 5, à rendre au plus tard le 18 avril

L'étape 5 est consacrée à l'analyse sémantique des fichiers d'entrée et la création d'un fichier matlab permettant de visualiser (en partie) l'état du système.
Il est indispensable que le travail demandé lors du TP2 soit terminé pour cette étape.

## Analyse sémantique

Les vérifications à effectuer sont les suivantes :

- Vérification de la force d'un contexte : celle-ci doit être comprise entre 0 et 100 (inclus).
- Vérification des positions des agents et des contextes : celles-ci doivent être compatibles avec la taille de l'environnement.
- Ordre des déclarations : lorsqu'on crée un nouvel agent, le type d'agent auquel il appartient doit avoir été déclaré au préalable.
- Vérification des attributs des agents :
    - Les noms des attributs d'un nouvel agent doivent correspondre à ceux du type d'agent auquel l'agent appartient. Pour simplifier, on considérera que l'ordre des attributs instanciés lors de la déclaration d'un agent doit être le même que celui des attributs déclarés dans un type d'agent.
    - Les types des valeurs affectées aux attributs doivent correspondre aux types des attributs du type d'agent auquel l'agent appartient.

## Création du fichier matlab

Ajoutez au programme les instructions permettant la création d'un fichier matlab qui contiendra les instructions de création d'une matrice dans laquelle on va positionner les agents et les contextes.

La création de l'environnement correspond à la création d'une matrice représentant une image en vraies couleurs dont tous les pixels sont noirs.

Affectation d'une couleur à chaque type d'agent plus une couleur commune à tous les contextes. Une table de couleurs pourra être prédéfinie en début de programme, une couleur étant définie par ses composantes rouge, verte et bleue (des valeurs réelles entre 0 et 1 inclus).

Placement d'un agent à la position (x,y) de l'environnement. Dans la matrice représentant l'image, un point de la couleur choisie pour le type d'agent est placé à la position (x,y).

Placement d'un contexte localisé à la position (x,y) avec une zone d'influence de rayon r. Dans la matrice représentant l'image, on place un disque de la couleur choisie pour les contextes centré en (x,y) et de rayon r. Attention, si un agent se trouve dans la zone d'influence d'un contexte, les 2 doivent être visibles (la zone d'influence et l'agent).

### *Rappel des instructions matlab nécessaires*

- Création d'une matrice représentant une image en vraies couleurs de m lignes et n colonnes dont tous les pixels sont noirs : `M = zeros(m,n,3)`
- Coloration du point (x,y) de l'image avec la couleur (r,v,b) : `M(x,y,:) = [r,v,b]`

Remarque : r, v et b doivent être des réels entre 0 et 1 inclus et les réels en matlab s'écrivent avec un point.

- Enregistrement de l'image couleur représentée par une matrice M dans un fichier de type bmp : `imwrite(M,'nom_fichier','bmp')`. Remarque : le fichier image obtenu à partir de la matrice

représentant les agents et contextes localisés dans l'environnement n'est pas le résultat attendu de la compilation mais l'instruction enregistrant cette image peut être ajoutée automatiquement à la fin du programme matlab généré pour que l'on puisse vérifier rapidement la bonne traduction du fichier d'entrée.

## *Rappel 2 : Exécution d'un programme matlab avec Octave*

Pour lancer Octave, il faut soit cliquer sur l'icône GNU Octave, soit ouvrir un terminal et lancer la commande :
`octave –force-gui`

Pour lancer un programme .m dans Octave,

1. Dans la zone « Current directory » (ou « Navigateur de fichiers »), se placer dans le dossier contenant le fichier .m,
2. Dans la zone « Command windows » (ou « Fenêtre de commandes »), taper le nom du fichier, sans le .m, et valider.

## *Exemples*

Les fichiers *exemple2.txt*, *exemple3.txt*, *systeme2.bmp* et *systeme3.bmp* sont sur plubel dans le dossier *Fichiers_exemples_etape5* en dessous de l'énoncé de l'étape 5.

A partir du fichier exemple2.txt

Si on choisit la couleur rouge (1,0,0) pour les agents de type pieton et la couleur grise (0.5,0.5,0.5) pour les contextes localisés, le fichier exemple2.txt va donner le fichier systeme2.m suivant :

```
rue = zeros(500,50,3) ; %définition de l'environnement
rue(400,10,:) = [1,0,0] ; %positionnement du piéton p1
rue(300,20,:) = [1,0,0] ; %positionnement du piéton p1
rue(350,15,:) = [1,0,0] ; %positionnement du piéton p1
%positionnement du contexte localisé arbre :
for lig = (450-5) : (450+5)
  for col = (10-5) : (10+5)
    if (rue(lig,col,:)==[0,0,0] && sqrt((lig-450)^2+(col-10)^2)<=5)
      rue(lig,col,:) = [0.5 , 0.5 , 0.5];
    endif
  endfor
endfor
imwrite(rue,'systeme2.bmp','bmp') %enregistrement de l'image
```

Et l'exécution de ce programme va donner l'image systeme2.bmp.

A partir du fichier exemple3.txt

Si on choisit la couleur rouge (1,0,0) pour les agents de type pieton, la couleur bleue (0,0,1) pour les agents de type voiture, et la couleur grise (0.5,0.5,0.5) pour les contextes localisés, le fichier exemple3.txt va donner le fichier systeme3.m suivant :

```
rue = zeros(100,30,3) ;
rue(18,5,:) = [1,0,0] ;
rue(55,6,:) = [1,0,0] ;
rue(20,20,:) = [0,0,1] ;
rue(40,20,:) = [0,0,1] ;
for lig = (15-8) : (15+8)
  for col = (10-8) : (10+8)
    if (rue(lig,col,:)==[0,0,0] && sqrt((lig-15)^2+(col-10)^2)<=8)
      rue(lig,col,:) = [0.5 , 0.5 , 0.5];
    endif
  endfor
endfor
for lig = (80-3) : (80+3)
  for col = (25-3) : (25+3)
    if (rue(lig,col,:) == [0,0,0] && sqrt((lig-80)^2+(col-25)^2)<=3)
      rue(lig,col,:) = [0.5 , 0.5 , 0.5];
    endif
  endfor
endfor
imwrite(rue,'systeme3.bmp','bmp')
```

Et l'exécution de ce programme va donner l'image systeme3.bmp.

# Travail à rendre au plus tard le 18 avril à 18h (pour tous les groupes)

Informations importantes :

Nous vous conseillons de commencer cette étape dès que possible (ne pas attendre le prochain TP).

Nous préférons que vous rendiez un programme incomplet mais qui fonctionne plutôt qu'un programme dans lequel vous avez essayé de tout implémenter mais qui ne fonctionne pas. Gardez donc toujours une version qui fonctionne avant de commencer à programmer un nouveau point.

Les fichiers (regroupés dans une archive zip) sont à rendre sur plubel en utilisant le lien « Dépôt étape 5 projet ».

Joignez un makefile ou un fichier texte dans lequel vous donnerez les commandes de compilation.

Rappel (TP2) : vous devez également fournir un document dans lequel est décrite la structure dans laquelle vous stockez toutes les informations concernant les identificateurs et les constantes.

Rendez un seul travail par groupe de projet. Dans les programmes lex et yacc, indiquez votre numéro de groupe de projet, ainsi que les noms des membres de celui-ci.