Table of Contents


```
%> @file Simulation.m
%> @brief Script to launch the labyrinth simulation.
%> This script allows you to run a complete simulation of the
 labyrinth.
%> It stops according to the stop conditions that are entered in the
 script.

% simulation totale
clear
close all
```

%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%% %%%%%%%%%%%%

```
%> @param[in] n Number of iterations. Static dimension
n= 100; % static dimension

% state of laby
labyState=cell(n,9); % static dimension
etat =0; % static dimension
etatS=0; % static dimension

%> Initialization Of labyrinth
%numberOfPossibleCaught = 3;
%> @param noEscape Select if there is an escape or no
noEscape = 0;
% Initial laby state
%> @param labyInit Structure containing all element to initialize
 the labyrinth. It is described as follows :"walls.V_i" for Vertical
 Walls, "walls.H_i" for Horizontal Walls, "pacman_i" for Pacman
 Initial position and "ghost_i" for ghost Initial Position.
     labyInit.wallsV_i =   [1 0 1 0; 1 1 0 1; 0 0 0 0;0 1 1 1; 1 0 0
 0];%  dimension can change
     labyInit.wallsH_i =  [0 0 0 1 0; 0 1 0 1 0;0 1 0 1 0;0 1 0 0
 1]; %  dimension can change
%
%  labyInit.wallsV_i =   [1 1;
```

```
%               0 1;
%               0 0]; %  dimension can change
%  labyInit.wallsH_i = [1 0 0;
%                              1 0 0];  %  d
    Ms = max(size(labyInit.wallsH_i)); % size of lab  % static
 dimension

    labyInit.pacman_i = [2,1]; % static dimension

    labyInit.escape_i = {[5 5], 0}; % static dimension

    % initial value of walls command
    wallsInit.wallsCommand_i = 0; % dimension can change
    % =0 : begin with right move
    % =1 : begin with up move

    % initial value of pacman command
    pacmanInit.pacmanCommand_i= zeros(1,5);% dimension can change

    % initial value of ghost command

    % initial value of stop
    stopInit.escape = 0;
    stopInit.pacman = 0;
```

%%%%%%%%%%%%%%%%%%%%%%%%% MAIN SCRIPT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```
    i = 1 ;
    SimulationStopped = 0;

    % creation of needed class
    wrapper = Wrapper(11, 9, labyInit, wallsInit, pacmanInit,
 stopInit);
    % run

    wrapper=wrapper.updateConnexion(1,1);
    wrapper=wrapper.updateConnexion(3,1);
    in = zeros(1,11);
    labyState(1,:)=wrapper.get_out();
    i=i+1;
while (i<=n && ~SimulationStopped)
    wrapper = wrapper.orderer(in);
    labyState(i,:)= wrapper.get_out();
    stop=wrapper.get_stop();
    %%%%%%%%%%%%%% stop condition %%%%%%%%%%%%%%%%%%%%%%%
     if (sum(stop)~=0)
         SimulationStopped = 1;
     end
     i = i + 1;
```

```
     %pause
    %%%%%%%%%%%%%%%%%%
end
```

log message on terminal

```
fprintf('End of simulation :\n');
if(i>n) % sim finish
    fprintf('\t The simulation was not stopped (%d steps)\n',n);
else %sim break
    fprintf('\t the simulation have been stopped at the %d step on %d
\n',i,n);
    if(stop(1))
        fprintf('\t>Pacman escaped\n');
    end
    if(stop(3))
        fprintf('\t>Pacman trapped\n');
    end
end
    n = i-1; % new number of iteration;
```

```
End of simulation :
  the simulation have been stopped at the 23 step on 100
 >Pacman trapped
```

Create picture for each iteration and Video in file data

```
% repo = strcat('./data/Validation 8/', 'Test1_2');
% mkdir(repo);
% save(strcat(repo,'/state'),'labyState');
 CreatePituresAndVideo_textured(n,  labyInit.escape_i, labyState);
```

Published with MATLAB® R2015a