VictimSim2: User Manual

Summary

1 Simulator

The simulator was created to test AI algorithms in a natural disaster, attack, or accident scenario. A 2D grid with victims, obstacles, and agents represents the environment.

The scenarios that can be simulated are composed of agents who try to locate the victims and agents (the same or others) who take a rescue kit to the found victims.

The simulator counts the number of victims located by the vital signs readings taken by the agents. The number of rescued victims is counted by the number of rescue kits left, one per victim.

1.1 Operation

The simulator engine is in the vs. environment[1], more precisely implemented in the *run method*. As long as an agent is in the ACTIVE or IDLE state, the simulator remains in the agent invocation loop. Below is a pseudo-code of the simulation engine:

```
While there is at least one IDLE or ACTIVE agent:
   For each ACTIVE agent:
      Invoke the agent method Deliberate()
      If the agent has timed out, then change its state to DEAD
      If the agent has no more actions to do:
         If the agent is at the base position, then change the state to ENDED
         If the agent is not at the base position, then change the state to DEAD
```

The interaction between the agents and the simulator happens by the *deliberate method* of the *vs.AbstAgent* class. Every agent created by the user should implement such a class. The idea is that the agents execute the cycle of *perceive-deliberate-actuate*.

2 Environment

Figure 1 shows an illustrative example of an environment: a grid of 4 columns (GRID_WIDTH) by five rows (GRID_HEIGHT). The base cell is at coordinates (2, 3).

The set $\mathrm{A_e} = \{E_1, E_2\}$ contains the explorer agents, the set $\mathrm{A_s} = \{S_1, S_2\}$, the rescuers, the set $V = \{v_1, v_2, v_3\}$, the victim. The black squares represent the obstacles (all are unsurmountable rocks or walls).

[1] vs is the directory where the main simulator source codes are located (vs = Victim Simulator)

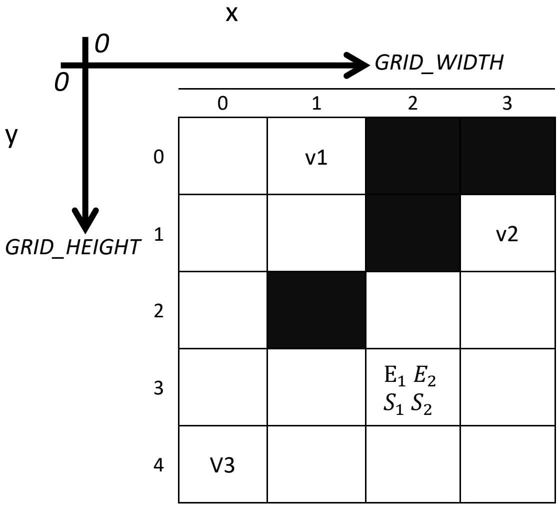

Figure 1: Environment with agents in the base position, casualties (V={v1, v2, v3}), and obstacles in black.

Each cell has a degree of ease or difficulty of access ranging from]0, 100]. In short, we call it the *difficulty of access.* It only affects the *walk* action of agents as specified below:

- **]0, 1[:** the cell offers ease of movement.
- **1**: the cell offers neither ease nor difficulty (it is the default).
- **]1, 100[:** the cell offers difficulty, but is still traversable.
- **100**: the cell is impassable (e.g., a wall, rock).

For example, suppose an agent has a default time of 1.5 for the action walk diagonally and enters a cell of difficulty 2. The simulator calculates that the *consumed time* is equal to 1.5 * 2.

The base cell is one of the input parameters for the environment simulator. The victims are distributed on the grid, with only one victim per cell. No cell can contain a victim and an insurmountable obstacle.

2.1 Environment configuration files

The environment changes according to the settings contained in the files below. Agents that the user creates **should not read these files**.

env_size.txt: the size of the room that is a grid of height x length.

```
BASE 0.0             ## Agents' Starting Position
GRID_WIDTH 30        ## Grid Width in Cells
GRID_HEIGHT 30       ## Grid Height in Cells
WINDOW_WIDTH 400     ## Window size in pixels
WINDOW_HEIGHT 400    ## Window height in pixels
DELAY 1.0            ## delay to delay GUI update by seconds
```

env_obst: Contains the placement of the walls in the grid in coordinates (lin, col)

```
x1,y1,d1             ## coordinate (x, y) of obstacle 1 and d1 difficulty of
access
...
xm,ym,dm             ## Coordinate (X, Y) of Obstacle M and DM Difficulty of
Access
```

3 Victim

Each victim occupies a cell in the environment, and there cannot be more than one victim per cell or in cells with insurmountable obstacles. The victims do not move. Their positions are determined in the environment configuration (see env_victims.txt).

A victim has a set of vital signs, as Table 1 shows. We remark the agents cannot read the gravity value or its label.

Acronym	Extensive	Range of values
pSist	Systolic pressure	[5, 22]
pDiast	Diastolic pressure	[0, 15]
qPA	Pressure quality	[-10, 10] such as: 0: the best balance between pSist and pDiast -10: Worst quality when pSist and/or pDiast are low +10: is the worst quality when pSist and/or pDiast are high
pulse	Pulse beats/minute	[0,200] bpm
resp	Respiratory rate/minute	[0.22] FPM
grav	Injury severity value (float)	[0, 100]
label	Risk of death class	1=CRITICS 2=UNSTABLE 3=POTENTIALLY STABLE 4=STABLE

Table 1: Vital signs of victims. Agents do not directly read the severity values or the label.

3.1 Vital Signs Data Archive

This file should not be accessed directly by agents; it should only be accessed on tasks requiring supervised machine-learning techniques.

env_victims.txt: contains the positioning of the victims on the grid in coordinates (lin, col). This file is related to env_vital_signals.txt (there must be a 1:1 match)

```
x1,y1              ## coordinate (x, y) of victim 1
...
xn,yn              ## coordinate (x, y) of victim n
```

env_vital_signals.txt:

For a victim, we have 5 vital signs that result in the severity level of the victim. All values are real numbers created randomly within the ranges shown, except for the label, which is an integer representing the severity of injuries.

$$i,\ pSist, pDiast, qPA, pulso, resp, sev, label$$

Example

```
seq, pSist, pDiast,  qPA,    pulse,   resp,   sev,     label
0,  8.5806, 2.2791, -8.4577, 56.8384, 9.2229, 33.5156,   2
```

Note that each env_victims.txt line is positionally related to the respective env_vital_signals.txt line.

env_victims.txt	env_vital_signals.txt
3, 1 1, 0 0, 4	0, 8.5806, 2.2791, -8.4577, 56.8384, 9.2229, 33.5156, 2 1, ... 2, ...

Table 2: Relationship between env_victims.txt and env_vital_signals.txt files

4 Agents

An agent can walk vertically, horizontally, and diagonally, one cell at a time. Two or more agents can occupy the same cell. One or more agents may occupy the same cell as a victim.

Generally, an agent can detect collisions with insurmountable obstacles, grid ends, and casualties. Also, it can read a victim's vital signs.

Agents cannot access the environment coordinate system. Therefore, they must create a coordinate system.

4.1 Configuration file for an agent

Each agent must have a configuration file. **All parameters must have values even if they are not used to avoid errors.** The costs of the actions are the patterns affected by the difficulty of access to the cells.

```
NAME EXPLORER1            ## Name of the agent to print messages
COLOR (255, 0, 127)       ## Color for Agent Drawing
TRACE_COLOR (255,153,204) ## color to leave footprints on visited cells
TLIM 40.0                 ## Timeout for the agent to perform its task
COST_LINE 1.0             ## Standard time to walk one cell in the hour. or
vertical
COST_DIAG 1.5             ## Default time to walk a cell diagonally
COST_READ 2.0             ## Standard time to read a victim's vital signs
COST_FIRST_AID 1.0        ## Default time to leave the first aid kit
```

4.2 Creating an Agent

Create a class that implements the abstract class *vs.AbstAgent***.**

See the public *vs.AbstAgent* **methods and examples available at https://github.com/tacla/VictimSim2**

5 Performance Metrics

5.1 Basic Definitions

Consider the definitions of the sets below in the performance calculation formulas calculated by the simulator.

- V: the set of victims dispersed in the environment, where the number of victims (cardinality of the set). This parameter is calculated from the reading of the input file. Victims are classified according to their risk of death. Therefore, set V is composed of the union of the following disjoint sets: $V = \bigcup_{i=1}^{4} v_i$
  - V_1: **CRITICAL** STATE (CLASS=1)
  - V_2: **UNSTABLE** STATE (CLASS=2)
  - V_3: **POTENTIALLY UNSTABLE STATE** (CLASS=3)
  - V_4: **STEADY STATE** (CLASS=4)
- V_e: set of victims located by the exploiter such that $|V_e| \leq |V|$. The set of victims found is also composed of the union of the disjunct sets: $V_e = \bigcup_{i=1}^{4} v_{e_i}$
  - V_{e_1}: **CRITICAL** STATE (CLASS=1)
  - V_{e_2}: **UNSTABLE** STATE (CLASS=2)
  - V_{e_3}: **POTENTIALLY UNSTABLE STATE** (CLASS=3)
  - V_{e_4}: **STEADY STATE** (CLASS=4)
- t_e: time spent by the exploiter such that $t_e \leq T_e$

The definitions for all the victims saved are identical to those for the victims found. So, replace the subscript *e* with *s.*

5.2 Search Metrics

The victim search performance metrics are as follows (calculated by the simulator).

Acronym	Description	Formula
$\boldsymbol{pve}$	Percentage of victims found	$\text{pve} = \lvert V_e \rvert / \lvert V \rvert$
$\boldsymbol{pte}$	Percentage of operating time used	$\text{pte} = t_e / \text{T}_e$
$\boldsymbol{veg}$	Found victims weighted by severity class. It shows the agent's ability to locate victims in a more severe condition, using the highest weighting for the most serious victims.	$veg = \frac{6\lvert \text{V}_{e_1} \rvert + 3\lvert \text{V}_{e_2} \rvert + 2\lvert \text{V}_{e_3} \rvert + \lvert \text{V}_{e_4} \rvert}{6\lvert V_1 \rvert + 3\lvert V_2 \rvert + 2\lvert V_3 \rvert + \lvert V_4 \rvert}$
$\boldsymbol{peg}$	The cumulative severity of the found victims over the cumulative severity values of all victims	$peg = \frac{\sum_{i=1}^{\lvert V_e \rvert} grav_i}{\sum_{j=1}^{\lvert V \rvert} grav_j}$

Table 3: Metrics for evaluating the performance of victim search agents.

5.3 Rescue Metrics

The performance metrics of rescued victims are as follows (calculated by the simulator).

Acronym	Description	Formula
$\boldsymbol{pvs}$	Percentage of victims rescued (received rescue kit)	$\text{pvs} = \lvert V_s \rvert / \lvert V \rvert$
$\boldsymbol{pts}$	Percentage of rescue time used	$\text{pts} = t_s / \text{T}_s$
$\boldsymbol{vsg}$	Rescued victims weighted by severity class. It shows the agent's ability to help a victim in a more severe condition, using the highest weighting for the most serious victims.	$vsg = \frac{6\lvert \text{V}_{s_1} \rvert + 3\lvert \text{V}_{s_2} \rvert + 2\lvert \text{V}_{s_3} \rvert + \lvert \text{V}_{s4} \rvert}{6\lvert V_1 \rvert + 3\lvert V_2 \rvert + 2\lvert V_3 \rvert + \lvert V_4 \rvert}$
$\boldsymbol{psg}$	The cumulative severity of the victims rescued over the cumulative severity values of all victims	$psg = \frac{\sum_{i=1}^{\lvert V_s \rvert} grav_i}{\sum_{j=1}^{\lvert V \rvert} grav_j}$

Table 4: Metrics for evaluating the performance of victim search agents.