Adaptive game engine

Defining architectures

By Vokovyy Pavlo

# Decomposing the task

The program consists of 2 major parts:

1. Creation of a game world
2. Creation of a story by simulating the game world

However, there are several different ways to distribute these 2 aspects through the timeline.

1. As 2 separate stages. Whole game world is built before the start of the simulation.

* Inefficient. Requires excessive amount of data to be downloaded and stored, some of which will never be used.
* Complex. Features answering the abstract questions like what is <object>, what characteristics has <object>
* ?

1. Simulation part use Creation part as a data retrieving service.

* Efficient. Uses only with required data.

# Workflow

1. User generates first game world with minimum data

// Example: An Elf and A dwarf are standing in the forest near the 2 boxes, the green and the red one.

1. The game world undergoes several improvements.

// Example: Elf, Dwarf, boxes get position in the forest object.

// Also subjects are being distinguished from objects

// Also subjects get goals (if there are none)

1. The simulation process starts

//…

1. All subjects (proactive agents) start their work. They can perceive from the environment object, execute actions with

//Example:

//dwarf:see() ->forest

//forest:send\_data()-> dwarf {elf:100,20 ; box:90,34} // other box is beyond perception

//dwarf:action(‘eat’,’box’)->forest

//forest: processAction(‘eat’, dwarf, box, forest) ->dwarf

//dwarf is upset by being unable to eat the box

// ANOTHER FORM

//dwarf is given the goal to stop being hungry

//dwarf looks around and finds the elf and the box (position and some info)

//dwarf looks through action available to do with the box and the elf

//dwarf tries to eat the box, the result is determined by characteristics of dwarf, box, end environment. First it is determined that dwarf is close enough to eat the box. Then the size of the box is compared to the size of the dwarf. Box is bigger -> dwarf can’t eat the box. Attempt reduces dwarfs vitality.

//dwarf receives the “fail” message.

//dwarf attempts to eat elf

//dwarf dies from being stubbed by the elf.

//// The problem is that dwarf does not know anything about the world. The knowledge about what it can do and what is impossible is absent

// EXAMPLE2

//dwarf gets data from the environment (including general knowledge facts ( some of the actionProcess functions)

//dwarf analyses possible outcomes and chooses not to eat anyone but to break the box

//dwarf moves and realises there is another box

//dwarf brakes another box

//dwarf tells elf “give food”

//dwarf tells elf “eat”

//….

//// Here the problem is that dwarf does not include recent experience

// EXAMPLE3

//…

//dwarf brakes the first box and remembers that result is negative

//dwarf asks elf “give food” and after not receiving any answer stops the attempts

//dwarf goes a couple of tiles forward and after not being receiving any food stops

//dwarf …

////

* Semantic web example

<http://wordnetweb.princeton.edu/perl/webwn?s=dwarf&sub=Search+WordNet&o2=1&o0=1&o8=1&o1=1&o7=1&o5=1&o9=&o6=1&o3=1&o4=1&h=00000000000>

* Game with generated storyline

<http://twistedtreegames.com/forest-of-sleep/#more-information>

* Some very useful data (Tale spin, OZ project)

<https://www.cc.gatech.edu/~riedl/pubs/dissertation.pdf>

* Storytelling software from image

<https://github.com/ryankiros/neural-storyteller>

* TaleSpin examples

<http://eliterature.org/images/microtalespin.txt>