The Robotic Algorithm for an Associative Self Learning System, simulating neuronal pathway generation

Article by Fredrik Lübbing, M. Sc., Nordic Health Sciences
September 2011

Introduction
This algorithm simulates the creation of neuronal pathways. It does not include a model for a physical human brain memory, that function is automatically generated through the Association and Environmental linkage data base and the Effect of Choice data base (a series of historical events, i. e. a linked list of classes). With one exception. For programming reasons, the input from the 5 senses are stored as if there where a physical memory in the brain.

The software mimics the experience that we all have as growing up, from infant to teenager. Some moral value parameters, and reaction patterns, are predefined (Predefined Choice Patterns). There is a Random Generator that generates Choices other than predefined, and a dynamic response data base with associative links are built up as a result of the random choice, with a Value Parameter ranging from -3 to +3, from “very bad to very good” (a 0 indicating no experience yet).

Background of the project
During the time 1981-1984 I was a Systems Analyst at US computer company Sperry UNIVAC. I worked with CAM systems for electronic designs, and especially a system written in the Norwegian object oriented language Simula, for simulation of effective throughput of advanced electronic circuit designs and and their physical implementation. Throughput in the sense that the various hardware pieces all are produced with statistical deviances of their internal function. I worked together with Professor Peter Jensen, of Copenhagen University, and Norsk Regnecentral, who invented the language Simula.

The simulations took place on a Sperry 1100/84 multiprocessor system, with a star bus oriented hardware internal design, and was capable of 800 Mega FLOPS (FLOating Points instructions per Second) at that time.

I was also experimenting with a software application that I wrote myself, LEARNING-ROBOT, which function could be explained as “How to program a robot so it becomes a self learning system”.

This documents describes the general structure of this application, and includes necessary external equipments like interfaces (simulating the 5 senses), and existing data bases.

Association
A list of current associations present in the robots “mental state” right now, saved in a class (compare: a snap-shot of current activated neurological connections in a living human brain).
Values
The Value data base stores the Reaction Choice Value Parameters for each trial situation, and consists of a number of objects with names and values in multi-dimensional matrixes (situation, choice, reaction, effect and some internal pointers).

Experiences
The values are linked to the situations, and stored as computerized “images” of what have happened before, i.e. the history of the robot’s life.

Set of Choice
A predefined database of what choices are possible or not possible for each situation constitutes the “playground” for the robot.

Effect of Choice – “Learning”
As the system is presented with randomized situations, and the Choice Manager selects randomly a choice, the Value of the Effect is stored together with the Situation and the Choice. The cumulative values are then used to bias the choice selection, when good enough values are found.

Predefined Choice Patterns
Here are stored some choices that will override the randomization, and acts as a “moral data base” that will guide the robot in certain situations.

Priorities
When the system is running in realtime (or simulated) multi-processor mode, we have a dynamic Priority Data Base, that works like a multi-functional Interrupt Vector or matrix. The priorities might change during the processing of a specific Choice, and then re-allocates dynamically to interrupt vector preferences.

Simulation of activity
Another random generator creates certain activities, that will alter the environment for the robot, and hence the choice selection possibilities.

Environmental description Data Base
An Environmental Description limits the choices for certain situations. For example when driving a car, the choices could be reduced to physically handling the car, i.e. steering wheel parameters, speed regulation, visual and auditory inputs from 5 Senses, etc. Even though the Interrupt Vector might dynamically interfere, for instance proclaiming a mobile phone call coming in to the phone in one of the pockets of the robot, and that needs another type of reaction, other than described in the car driving environment data base.

Active parameters
A list of current parameters present for the currently active Associations present.
Simulation - Current Situation - Events
In a specific time during the simulation, the current activities of the robot are stored in an Event Manager, describing what the robot is doing right now. For instance: Driving a car and thinking of his girlfriend. One physical activity and one mental. Also mental sub-conscious and bodily physiological internal activities are possible activities.

Events
Are a set of processes running in the robot and in the current environment.

5 Senses Dynamic Patterns Build Up
A dynamic matrix of objects describing what information are running into the robot by the 5 senses interfaces.

Biological Functions Interface – External
For internal description of the simulated robots bodily and mental functions, a physiological data base with biological AI system is needed. This is running in parallel with the robot activity simulation.

Computer Hardware - External
A description of what hardware are running the system.

5 Senses realtime interfaces – External
Auditory, visual etc hardware channel for connection to computer system.

SOFTWARE IMPLEMENTATION
The software is based on multi-processor capabilities, dynamic linking and dynamic memory allocation, physical interrupt vector assembler/OS interface capabilities.

Data Structures

<table>
<thead>
<tr>
<th>Situation</th>
<th>= Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>= Numerical Array</td>
</tr>
<tr>
<td>Effect of Choice</td>
<td>= Object Matrix, parameters for each choice</td>
</tr>
<tr>
<td>Associations</td>
<td>= Object Matrix of existing Links</td>
</tr>
<tr>
<td>Priority Data Base</td>
<td>= Object</td>
</tr>
<tr>
<td>5 Senses</td>
<td>= Object</td>
</tr>
<tr>
<td>Activities</td>
<td>= Object</td>
</tr>
<tr>
<td>Environment</td>
<td>= Object</td>
</tr>
<tr>
<td>Environmental Data Base</td>
<td>= Object</td>
</tr>
<tr>
<td>Parameters</td>
<td>= Object</td>
</tr>
<tr>
<td>Predefined Choice Patterns</td>
<td>= Object</td>
</tr>
<tr>
<td>Choice Value Parameters</td>
<td>= Object</td>
</tr>
</tbody>
</table>
Biological Functions Interface

A software interface to a biological AI simulation system describing bodily internal functions.

Predefined Procedures

This system is based on the Simula Language incorporation of the System Classes SIMSET and SIMULATION.

EXTERNAL PROCEDURES

Operating Systems procedure for dynamic processor allocation for each software process, including dynamic linkage.

Interrupt Vector procedure (read / write the interrupt vector in the operating system)

HARDWARE IMPLEMENTATION

5 Senses realtime physical interfaces, i.e. information providers, could be camera, microphone, smell simulator, taste simulator and touch simulator.

Appendix – An example of a Main Program procedure

This program generates a new learning experience for the robot:

PROCEDURE Create_New_Learning_Experience;
BEGIN
  Current_Event :- NEW Event_Generator (Situation, Parameters);
  ACTIVATE 5_Senses;
  Choice_Limiter :- NEW Choice_Limiter (Current_Event.PossibleChoice (Situation, Parameters));
  Choice_Pointer :- Generate_Choice (Choice_Limiter);
  ACTIVATE Effect_Manager;
  DEACTIVATE 5_Senses;
END;

Where:
Current_Event is a pointer to a class describing an event

Event_Generator is a procedure class with 2 receiving parameters: Situation and Parameters, both being classes. Returns a pointer to a class.

5_Sences is pointer to a running simulation class, handling the I/O of the 5 Sences interface.

Choice_Limiter is a pointer to a class describing the what choices are possible for the moment

Choice_Pointer is a pointer to the class Choice

Current_Event.Possible_Choice is an internal procedure in the class Event, that updates the possible choice in this moment according to Situation and Parameters

Effect_Manager is a simulation class that evaluates the effects of a choice, and stores that in the effects data base

ACTIVATE and DEACTIVATE are Simula Simulation class functions for handling realtime processes

Appendix 2 – An example of processing an Event

When a new event occurs, we check with the event history data base to see if it has occurred before. Suppose it has. Then we see if we can get a more then 80 % match with the environmental parameters. If so, we choice the same choice as in that previous event situation. Registers the effect, and if good or very good (+2 or +3), we increment the experience value for that event and environment.

If no match, we go to 5 sences database and look for a pattern recognition. If we find some, we try the associated choices. Of all possible hits, we select the best one, and store it as the best choice so far. We also store environment and parameters with that choice-effect experience. And we also randomize a choice, and see if we get a +2 or +3. If not, we skip it and randomize a new one. So we go through all possible choices. Of all possible hits, we select the best one, and store it as the best choice so far. We also store environment, parameters and 5 senses pattern with that choice-effect experience.

Below : Snap shot diagram of data structure in the above