
MONA: HIERARCHICAL CONTEXT-LEARNING IN A GOAL-SEEKING ARTIFICIAL NEURAL NETWORK

Version 1.1, February 2014

Tom Portegys, portegys@gmail.com

CONTENTS

Description.....	4
Neurons	4
Needs, goals and motivation.	6
Learning	6
Algorithms	7
Sense.....	8
Enable	9
Learn	10
Drive	11
Respond	12
Projects	12
Muzz World	12
Atani Robot.....	14
Related work.....	16
HTM/NuPIC.....	16
BECCA	17
Future work	17
References.....	17

DESCRIPTION

Mona is a goal-seeking artificial neural network (ANN) that learns hierarchies of cause and effect contexts. These contexts allow Mona to predict and manipulate future events. The structure of the environment is modeled in long-term memory; the state of the environment is modeled in working memory. Mona is an active system: it uses environmental contexts to produce responses that navigate the environment toward goal events that satisfy internal needs. Goal-seeking thus also filters learning to retain more relevant information about the environment.

Biological neural networks effectively and sometimes uniquely exhibit a number of desirable properties, such as pattern classification, noise tolerance, adaptability, intelligence, etc. The aim of ANNs is to also exhibit these properties through modeling at some level of abstraction. ANN programming is also done via learning, as opposed to brittle domain-specific encoding.

It is tempting to model biological neurons and networks to a fine level of detail, however there are cautionary considerations:

- There are certainly important mechanisms yet to be discovered. For example, are glial cells involved in memory?
- It is largely unknown which mechanisms are essential, incidental, or are implementations specific to a biological medium. Neurochemical simulations are also computing intensive. Can artifice surpass evolution in a different medium?

While most ANN research remains focused on the perceptron architecture, which has seen great success mainly as a pattern classifier, Mona is an attempt to simulate animal-like behavior that is likely a necessary, though possibly not sufficient, condition for artificial intelligence.

Mona addresses several crucial problems that ANNs tend to struggle with, yet biological neural networks have solved to a large extent:

- **Stability:** how well is previous learning retained while new learning occurs? ANNs typically require the retraining previously learned patterns in order to incorporate new patterns. This is much less the case with Mona, since new information is stored separately in memory.
- **Plasticity:** how quickly can new learning be done, especially in the presence of previous learning? ANNs typically require extensive repetitions to learn new patterns. Mona is capable of single-trial learning, a skill vital for living organisms.
- **Modularity:** can independently learned things naturally integrate? Mona can seamlessly navigate environments previously learned piece-wise, not as a whole.

NEURONS

A Mona network contains three types of neurons: receptors, motors, and mediators:

- **Receptor neurons** are associated with input sensors, with a possible filtering/feature detection layer between the sensors and the receptor neurons. Currently sensor vectors are compactly represented by centroids in sensor space. Sensors can also be grouped into

modalities, which are sub-vectors of related values. For example, a set of rangefinder readings could be segregated from a touch sensor reading. A receptor neuron can represent a specific modality or the entire sensor vector, in which case it is non-modal.

- Motor neurons are associated with output responses. For example, a firing motor neuron can cause a robot to make a right turn.
- Mediator neurons capture the long-term memory notion of a causal relationship between environmental events signified by neuron firings:
 - One component neuron represents the cause event, and another the effect event. Often a motor neuron resides between the cause and the effect, representing a response event. Figure 1 depicts this stimulus-response-stimulus relationship. A neuron can be reused as a modular component of multiple mediators.
 - Mediator neurons can be components of higher level mediators, building hierarchical structures of cause and effect.
 - The *enablement* of a mediator represents the cause and effect reliability of the mediator. This is a fluid quantity that shifts between events, depending on their firing state.
 - Working memory is implemented in the *enabling* state of a mediator. Initially, the enablement of a mediator enables the cause event. When the cause neuron fires, the response event becomes enabled. When the response motor neuron fires, the effect event becomes enabled, indicating that the neuron is expected to fire within a time frame proportional to the level of the mediator. When its effect event fires, the mediator fires.
 - Higher level over-arching mediators represent hierarchical enabling contexts for a component mediator. Enabling a component mediator temporarily increases its enablement, meaning that the mediator is more likely to fire in that context. As a figurative example, a mediator controlling the opening of a locked door would only be expected to work in the contexts of a) having the door key, and b) being at the correct door.

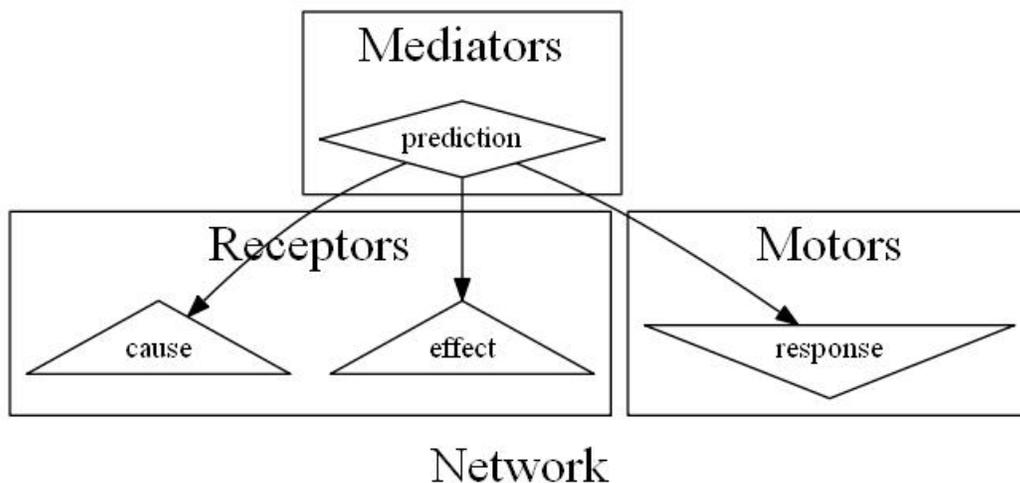


Figure 1 – A simple Mona network.

NEEDS, GOALS AND MOTIVATION.

The network also contains one or more *needs*, which are simple numeric variables that can, for example, represent homeostatic quantities such as hunger or thirst. A need is associated with one or more goals, which are neurons whose firings serve to decrease the value of the need by a *goal value* amount. The goal value of a neuron can either be programmed into the network or learned when its firing coincides with need changes. The latter allows reinforcement training of behavior by external manipulation of need, i.e., reward.

Through their goal neurons, needs *drive* the network to produce a sequence of motor responses that navigate from current enabling contexts toward the firing of goal neurons. This embodies an integrated planning capability for interacting with the environment to achieve goals that satisfy needs.

Drive back-propagates *motive* from effect to response to cause events within mediators, and between mediators of different levels. Motive is channeled through a gating network formed by the enabling state of the mediators. For example, if a mediator is enabled, it will channel motive into either into its cause, response, or effect event neurons depending on which is enabled. If the mediator is not enabled, it will channel motive into higher level mediators for which it is an effect. This will in turn help to fire cause events that create an enabling context for it. Motor neurons accumulate motive as it propagates to them through the network. The response is chosen from motor neurons containing the highest quantities of motive.

The structure and state of the network capture long-term and short-term knowledge of the environment. Needs and goals constitute a separate control of the network. This means that the same network can be used for different tasks. In other words, keeping control and data separate (a standard computer science practice) allows multiple applications of the data. The standard perceptron model combines data with control.

Multiple needs can simultaneously drive the network toward multiple goals. Depending on the strength of the needs and goal values, this can result in seemingly single minded behavior, behavior that appears to be skillfully planned to achieve multiple goals, or behavior that seems scatter brained. When a need is satiated to zero, it no longer drives the network, meaning that behaviors motivated by other needs will take over.

LEARNING

A receptor neuron is created when a sensor vector does not match any existing sensor centroid in sensor space within a specified tolerance. Motor neurons are typically added at network initialization to effect a set of output responses.

A mediator neuron is created from sequentially correlated neuron firing events that are not already encompassed by an existing mediator. A mediator represents a speculative cause and effect relationship between its events. As previously mentioned, a mediator's effect event is allowed to fire within a prescribed span of time. For higher level mediators, this span is typically

longer. However, for learning a new mediator from sequential events, the span of time separating the events is typically set to a lesser amount, again depending on the level of the mediator. This scheme reduces the spurious formation of mediators from events that are actually uncorrelated.

The lowest level mediators are composed of receptor and motor events. Higher level mediators are composed of lower level mediator events, typically the next level down. When sensor modalities are in use, a non-modal receptor is initially incorporated as the effect event, speculating that the entire sensor vector is significant. When the effect event of such a mediator fails to fire as expected, a generalized version of it can be created by substituting a firing modal receptor as the effect event, a speculation that a subset of the sensor vector is actually significant.

A new mediator is added to the network and immediately fired, allowing it to serve as an event in the creation of a higher level mediator.

A new mediator neuron is assigned an initial quantity of enablement. After that, its enablement is updated to reflect the probability with which it predicts the firing of its effect event given the firing of its cause event and, if equipped, its response event. This update can be dependent on context, meaning that a mediator often firing in an enabling context could have an inflated enablement.

The rate, or velocity, of enablement change is typically different for increase and decrease. Enablement increases relatively gradually, reflecting a slow build up in confidence of a mediator. Conversely, enablement decreases relatively quickly to penalize mediators failing outside of inflating contexts, for example.

The network has a maximum mediator capacity, so retention in the network depends on a mediator's enablement and firing history relative to other mediators. These two quantities define a mediator's *utility*. This means, for example, that a less reliable mediator that is close to a goal could be driven frequently despite a significant firing failure rate. In this case, the mediator's usage would boost its utility, providing it a measure of retention protection. A mediator cannot be deleted until all of its parent mediators are deleted since its parents cannot exist without it.

ALGORITHMS

Mona's processing cycle, shown in Figure 2, is executed for each input-output interchange with the environment.

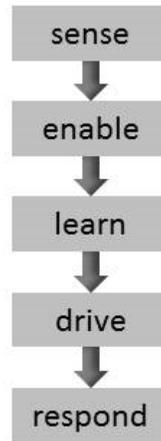


Figure 2 – Processing cycle.

The high-level algorithms for each function are described in the following sections. The code is at www.codeplex.com/mona

SENSE

The sense function maps inputs from the environment to receptor neurons.

Sense:

Update need based on sensors.

Fire receptors matching sensor modes:

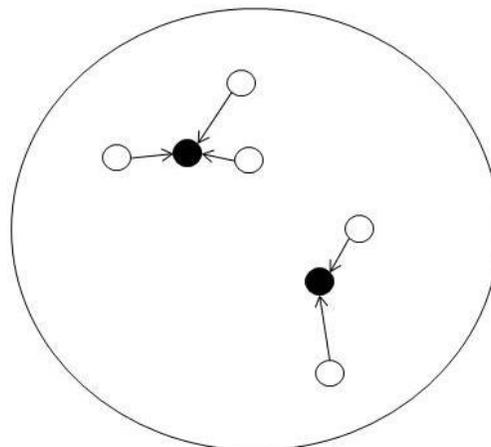
Apply sensor mode to sensors.

Get firing receptor based on sensor vector and mode.

Create a receptor that matches sensor vector?

Fire receptor.

Update receptor goal value.



● = centroid vector

○ = sensor vector

Figure 3 – Centroid sensor space.

Floating point sensor values are quantized by mapping them into centroid sensor spaces, one for each sensor modality. This is shown in Figure 3. A new centroid is created when a sensor vector does not map to an existing centroid within a specified distance. Modifying this distance changes the sensory “focus” on the environment, making it fuzzier or sharper. The centroid spaces are maintained by a fast multi-dimensional lookup algorithm:

T. E. Portegys, "A Search Technique for Pattern Recognition Using Relative Distances", [*IEEE Transactions on Pattern Analysis and Machine Intelligence*](#), Vol. 17, Number 9, September, 1995.

ENABLE

Using the motor neuron firing for the previous cycle and the receptor neuron firings for the current cycle, the enable function hierarchically fires mediator neurons. This produces changes in the enabling distribution among mediator events, which will channel motive in the subsequent drive function.

```
Enable:
  Notify mediators of response firing events.
  Notify mediators of effect firing events.
  Notify mediators of cause firing events.
  Update mediator effective enablements.

Firing of mediator cause event:
  Determine enabling strength.
  Transfer enabling to next neuron.

Firing of mediator response event:
  Transfer enabling to effect event.

Firing of mediator effect event:
  Update goal value.
  Update enablement and utility.
  Fire the mediator.
  Notify parent mediators of event firing.

Update enablement:
  If neuron is firing:
    Enablement +=
      (1 - Enablement) *
```

```
Update value *
  LEARNING_INCREASE_VELOCITY
Else neuron failed to fire
  Enablement -=
    Enablement *
    Update value *
    LEARNING_DECREASE_VELOCITY
```

Update mediator effective enablement:

The effective enablement combines a mediator's enablement with the enablements of its overlying mediator hierarchy to determine its ability to predict its effect event.

LEARN

After the enable function fires neurons, the learn function creates new mediators by correlating sequences of firing events. These events are saved in a set of time-stamped lists spanning the maximum allowable amount of time that events can be correlated over. There is an event list for each mediator level.

Learn:

```
Purge obsolete events.
Time-stamp and save events.
Create new mediators from effect events.
Create mediators with generalized effect events.
Delete excess mediators.
Increment event clock.
```

Create new mediators for given effect:

```
Find cause event candidates in event lists.
Choose causes and create mediators:
  Add a response?
  Create the mediator.
  Duplicate?
  Make new mediator available for learning.
```

Create generalized mediators:

```
Find effect event candidates:
  Is this a modal superset of the mediator's effect?
Choose effects and create mediators:
  Make a probabilistic decision to create mediator.
  Create the mediator.
```

Duplicate?
Make new mediator available for learning.

DRIVE

The enable function sets the stage for driving motive emanating from needs via goals through the network, channeled by the enabling state of the mediators. Motive is accumulated in motor neurons, subsequently used by the respond function to produce a response.

Drive:

- Initialize neurons for drive.
- Drive goal receptors.
- Drive goal motors.
- Drive goal mediators.

Initialize neuron drive:

- Assign drive weights to destinations:

The input motive is divided into the "down" value that is driven to mediator component events and the "up" value that is driven to parents. The down value is proportional to the mediator effective enablement. The remaining fraction goes up. For receptors, all motive is driven to parents; for motors, none is.

- Divide the down amount:

- Distribute the down amount among components.

- Distribute the up amount among parents.

Neuron drive:

- Accumulate need change due to goal value.
- Accumulate motive.
- Distribute motive to component events:
 - Drive motive to cause event.
 - Drive motive to response event.
 - Drive motive to effect event.
- Drive motive to parent mediators.

Parent mediator drive:

- Accumulate motive.
- Drive motive to cause event.

```
Drive motive to response event.  
Drive motive to parent mediators.
```

RESPOND

The respond function selects a response from the most motivated motor neurons.

Respond:

```
Get response potentials from motor motives.  
Incorporate minimal randomness.  
Make a probabilistic response selection?  
Select maximum response potential.  
Fire responding motor.
```

PROJECTS

There have been a number of projects to develop Mona and in support of published research. For example, a maze-learning task was employed to compare Mona with Elman and LSTM ANNs in regard to modular sequence learning. Mona was uniquely able to integrate separately learned contexts to solve composite mazes.

In a recent demonstration of a hybrid instinct/experiential learning ANN, Mona and a recurrent ANN were partnered to cooperatively solve a set of discrimination/generalization tasks. For this, artificial creatures are born into one of a set of possible worlds with innate general knowledge of the worlds. This confers generally useful skills for navigating any of the worlds. However, when this knowledge fails due to variations in the creature's specific world, experiential learning overrides to bridge the navigation gap. Mona played the part of the experiential learner, since it is well suited for predicting patterns that are not well delimited in a sensory-motor stream.

Below are two projects demonstrating Mona's capabilities. The first is an artificial block world in which food and water are goals for hunger and thirst needs. In the second Mona served as the controller for a simulated maze-learning robot. The code for Mona and these projects is at www.codeplex.com/mona

MUZZ WORLD

The Muzz World is a block world inhabited by muzzes. Blocks in the world are marked with characters. There are also marked ramps in the world that can lead between levels. A muzz needs food and water, and the world contains mushrooms and pools as goals for these needs. A muzz can sense the character or object on the block or ramp directly ahead, and whether the forward, left and right directions are accessible. For responses, a muzz and move forward, turn right and left, eat, drink and wait. Figure 4 shows a split overview and muzz view of the world.

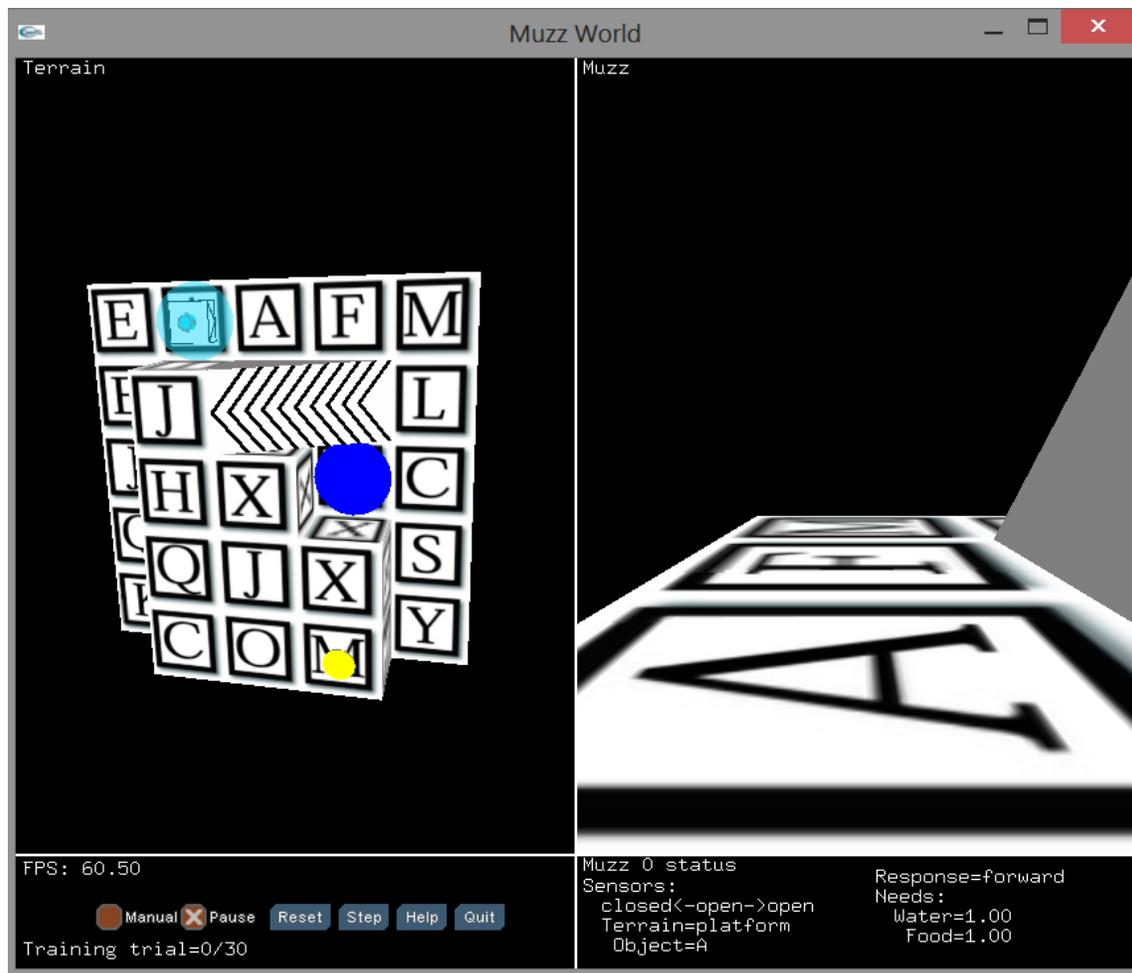


Figure 4 – Muzz World.

An expedient way of training a muzz to find food and water is by overriding its responses with correct ones. When this is done for a number of trials, the muzz will be able to do so on its own. Using reward to incrementally train is also possible.

Figure 5 shows the muzz's neural network after a few steps of training. Mediators rapidly build up to associate learned cause and effect correlations. Some of the graphic notations denote event relationship and firing status.

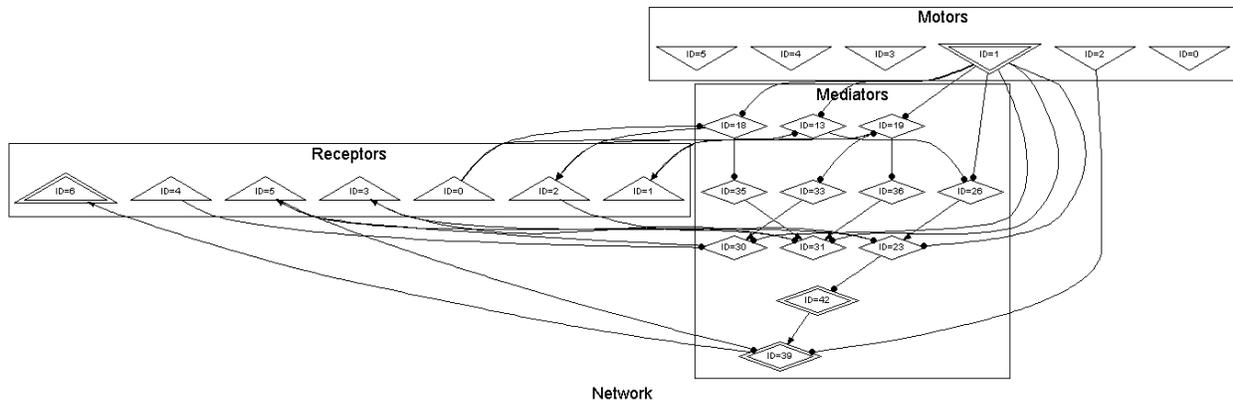


Figure 5 – Neural network after a few steps.

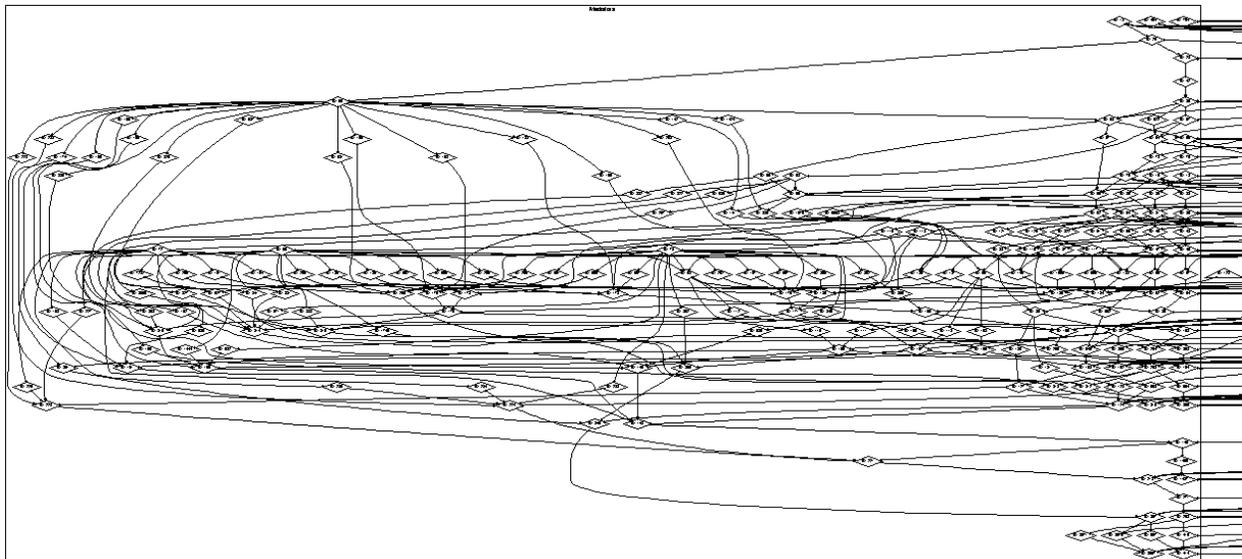


Figure 6 – Neural network after learning food and water task.

After thirty training trials, the mediator block of the network looks like Figure 6. This is a rich, redundant fabric of intertwined and overlaid mediators containing the know-how necessary to solve the task.

ATANI ROBOT

This is a simulated Pioneer 3DX robot controlled by a Mona neural network that learns T-mazes, a popular vehicle for animal learning experiments. It uses a high-fidelity simulation of the robot body, physics, and sensors: laser range finder, webcam, and bumper. A demo can be viewed at tom.portegys.com/research.html#atani.

Figure 7 shows a close-up of the robot in the maze. Figure 8 shows an overview of a T-maze.



Figure 7 – Atani robot close-up.

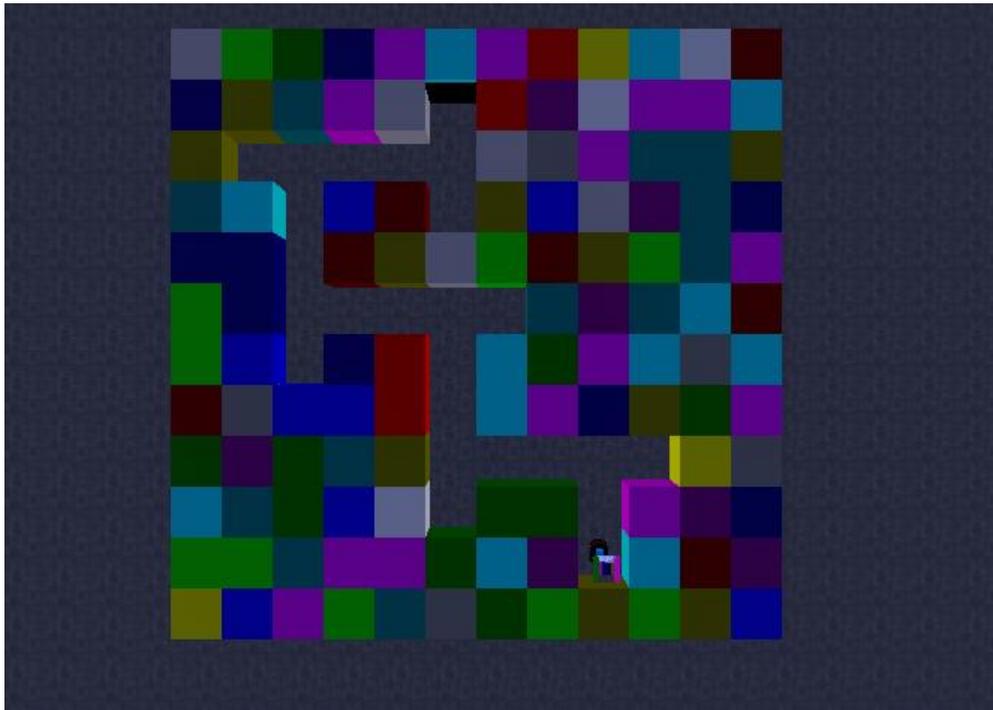


Figure 8 – T-maze overview.

The color webcam sensors and the laser range finder were floating point quantities with significant variability. For this the centroid sensor space served well as a means of quantizing inputs.

The robot was trained quickly with the forced response method used for training muzzes, but a dashboard was also available to train the robot by reward. As anyone who has tried to do this with an animal, it is quite a tedious process.

RELATED WORK

HTM/NuPIC

Hierarchical Temporal Memory (HTM)/Numenta Platform for Intelligent Computing (NuPIC) is a model of the architecture and algorithms (in the form of the Cortical Learning Algorithms (CLA)) of the mammalian neocortex.

These are some similarities with Mona:

- In HTM and Mona hierarchical sequence learning is key for making predictions.
- In HTM, modular reuse is important. In Mona, a neuron can be reused as a component for multiple mediators.
- Online learning. In HTM, there seems not to be an inherent difference between training and testing. The same is so for Mona, which learns very quickly to adapt to new environments without formal training epochs.
- Sparse distributed representations are used in HTM to try to extract the most salient input patterns. In Mona, sensory field centroids are used for this purpose, although perceptron-like feature detectors could also be plugged in.

These are some differences:

- In the initial HTM, the behavior/response aspect is de-emphasized, but in humans and animals responses are vital to be able to explore the environment. In Mona, the stimulus-response-stimulus construct is a basic scheme, implemented by receptor and motor neurons orchestrated by mediator neurons. Note: there are recent efforts to incorporate motor activity into the HTM model in light of neurological discoveries about how widespread and integrated motor outputs are in the neocortex.
- Needs and goals for filtering and motivation. It isn't feasible for a brain just to sponge up a complex world. Animals and humans filter the environment by retaining that which achieves goals satisfying needs and discarding the rest.
- In HTM, the "reptile" (older and deeper structures) brain is not thought to be necessary to the achievement of intelligence, but I believe that it is actually crucial in that it supplies motivations that filter and control sensory-response learning, and instincts that serve as substrates for the higher level behavior that is generated by the neocortex.

HTM/NuPIC reference:

www.numenta.com/htm-overview/education/HTM_CorticalLearningAlgorithms.pdf

BECCA

Brain-Emulating Cognition and Control Architecture (BECCA) is a general learning program that consists of an automatic feature creator and a model-based reinforcement learner. Reinforcement learning is based on biologically-inspired algorithms. BECCA, like Mona, assumes nothing about the environment, and interacts with it through sensors and responses. BECCA uses reinforcement learning as a means of training: an agent takes actions and receives sensory information at each time step. Its goal is to maximize its reward. Mona also implements reinforcement learning that associates neuron firings with need reductions. Mona also supports response override training.

BECCA reference:

www.sandia.gov/~brrohre/index.html

FUTURE WORK

1. Planning and learning by internally/speculatively firing neurons. Can something like planning or imagination be modeled by internally fired neurons? This is inspired by the well-known observation that animals internally rehearse previously learned behaviors. In terms of Mona, when a causal context is in effect, speculatively firing the effect neuron could generate an internal stimulus akin to planning future activity. When this is done with multiple independent contexts, novel stimuli could be produced suitable for learning.
2. Context-specific mediator states. Currently a mediator's enabling state, which is a result of its event neuron firings, is the same for all its contextual parent mediators. This is restrictive and unnatural considering that the same type of cause and effect relationship could be in a different state in different contexts. Philosophically, the current implementation views an individual mediator as an object rather than a class or type of object. Viewing a mediator as a type allows instances of it to occupy multiple places in an environment model. This is related to the famous AI frame problem. A demonstration might consist of a world with moving/moveable objects situated in fixed backgrounds.

REFERENCES

More references, abstracts and some papers at tom.portegys.com/research.html.

"A Connectionist Model of Motivation", *IJCNN'99 Proceedings*.

"Goal-Seeking Behavior in a Connectionist Model", *Artificial Intelligence Review*. 16 (3):225-253, November, 2001.

"An Application of Context-Learning in a Goal-Seeking Neural Network", *The IASTED International Conference on Computational Intelligence (CI 2005)*.

"A Maze Learning Comparison of Elman, Long Short-Term Memory, and Mona Neural Networks", *Neural Networks*, 2010 Mar; 23(2):306-13.

"Discrimination Learning Guided By Instinct", *International Journal of Hybrid Intelligent Systems*, 10 (2013) 129-136.