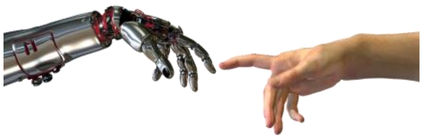
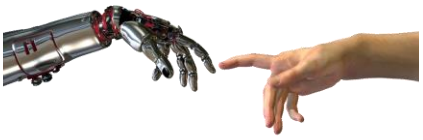
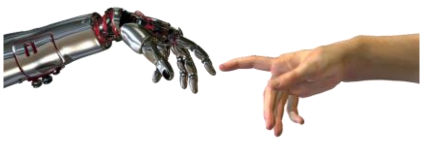


State-of-the-art implementation of the CM Architecture

Author:	Thor List, CMLabs
Created:	2 April 2017
Last modified:	14 July 2017



Contents



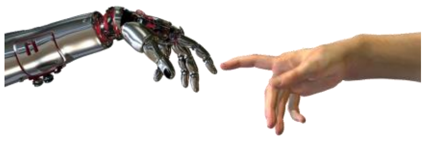
Introduction

The original Cognitive Map Architecture was created in collaboration with Honda Research Institute in California for the purpose of adding learning and human interaction to their ASIMO robot. It was implemented in Psyclone version 1 and ran a combination of C++, Java and LUA code.

Since then we have release Psyclone 2 which is a significantly improved architecture in terms of performance, stability and interaction with third-party software and hardware. It furthermore adds native support for Python which has emerged as a firm language of choice for robotics and AI.

This work package is about porting the relevant parts of the old Cognitive Map Architecture to run in Psyclone 2, in preparation of upgrading it to Collaborative Cognitive Maps for the collective use of multiple robots in human interaction scenarios.

CoCoMaps is funded by the ECHORD++ project through the European Union's 7th Framework Programme for Research, Technological Development and Demonstration, under grant agreement no 601116.



The Cognitive Map Architecture in Psyclone 1

This section summarizes the original CMA in Psyclone1, highlighting the aspects that have been ported to Psyclone 2, and thus remain (largely) unchanged. The system can be split into several areas of focus.

Input and Output Data

The input data consists of sensor and system data from a large collection of robot functions from vision and audio input across higher level contextual information about the robot state to low level system information such as CPU and temperature data. Psyclone lets module developers pick and choose what and when to receive which data, either in raw or pre-processes form.

Mixing Discrete and Streaming Data

Psyclone 1 had two ways of working with data. Discrete data was sent via typed messages which could be routed using context-aware publish-subscription mechanism and streaming data would be handled by advanced buffering structures which could be queried by modules using a set of interfaces, each also buffered locally. Data overflow could be handled either centrally or locally for each receiver with policies.

Detectors and Deciders

Both discrete and streaming data could be fed into special modules called Detectors and Deciders. Detectors' main task is to sift through raw data and detect events either temporally or in correlation with other data sources. Deciders will take the detected events from a number of Detectors and make choices based on these which will affect the rest of the system through the use of global contexts.

Blackboards

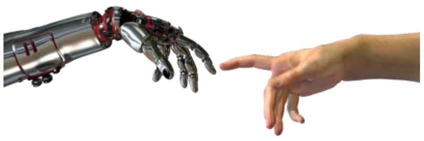
In Psyclone 1 Whiteboards (an advanced implementation of the traditional Blackboards) were the central nodes through which all data, both discrete and streaming, would go and the Whiteboards would be in charge of delivering the data to modules that subscribed to this particular type of data. Developers can use Whiteboards to monitor data in real-time, both during development and once the system has been put into production. Additionally, the Whiteboards store the data for a while and modules can query data dynamically as they need the data using custom parameters not necessarily known at compile time.

Mixing Multiple Languages (C, Java, LUA)

Modules in Psyclone 1 could be written in standard C, C++ and Java and the Cognitive Map architecture allowed the integration of the LUA language for scripted cranks in modules. This allowed the designers the ability to inject LUA code at runtime which would be used dynamically and could be changed again later on.

Environment Map

The Cognitive Map also kept information of the current state of the environment around it. This was stored in the Environment Map and represented everything the robot knew about the external world including positions of objects and people. The Cognitive Map could freely query the information stored in the Environment Map and use it for making decisions in real-time.

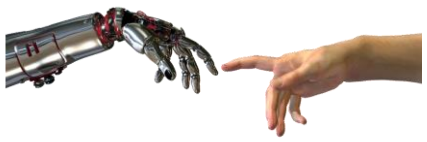


Task Matrix

A Task Matrix was designed to keep track of known tasks and the real-time status of each task as it executes through the system. In CoCoMaps this will be superseded by the Task Dialog Manager and was only partially ported across as part of this task.

Visualisation and Dashboards

A set of custom visualisations and dashboards were created as part of the Cognitive Map Architecture. They were created outside Psyclone and implemented external modules for communication to and from the rest of the Psyclone system. In CoCoMaps this will be superseded by the more advanced PsyProbe system and was only partially ported across as part of this task.



What Had to Change for Psyclone 2

Because Psyclone 2 is a complete rewrite of Psyclone 1, enhancing performance, reliability, and flexibility, many things had to change when porting across the existing functionality. Additionally, some parts of the original Cognitive Map Architecture were either out of date or didn't make sense as part of the larger Collaborative Cognitive Map's architecture, so they were either changed, rewritten, or deprecated altogether.

Benefits of Psyclone 2

Psyclone 2 offers a massive jump in performance, by a factor of 50 or more in terms of latency and resource use for sending messages between processes and in a computer network. Streaming data is now handled exactly the same as discrete data, which makes programming of modules using both much simpler.

The new PsyProbe web interface offers lots of built-in functionality as well as the ability to extend and add functionality for custom modules. You can now also create dashboards and whole websites as an integral part of PsyProbe without ever leaving the Psyclone system.

Data can still flow via Whiteboards, but do not have to which makes the Whiteboards more useful as you can filter what data goes onto them. And a range of built-in Catalogs can be used to manage data in various ways including saving it to disk between runs, making custom data available to custom-made webpages and viewing live timelines of data as well as live hierarchical structures, both in PsyProbe directly and in separate browser windows.

Specific Changes for Psyclone 2

The dataflow is much simplified now as discrete and streaming data can be treated the same way. Cutting out the Whiteboards especially for the streaming data has meant a huge improvement in performance and lowering of complexity.

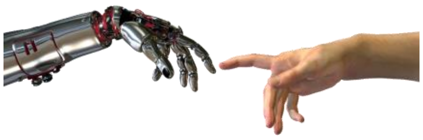
Detectors and Deciders are largely unchanged, only updated slightly to match the new Psyclone 2 API. Unifying discrete and streaming data in these has cut out a lot of complexity too.

Psyclone 2 now has native support for Python modules which is a much more useful language in terms of AI and robotics. Java has been superseded in that respect, but can be added back in at a later date. And Python can do what LUA used to do and much more.

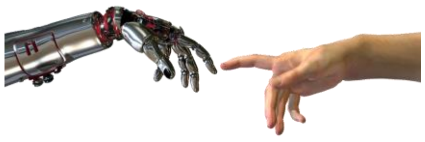
Whiteboards are largely unchanged as well except for less data flowing through now that not all data has to do so. That makes them more usable for monitoring and debugging too. Catalogs are now used to store up-front configuration data and runtime data which needs to persist between runs.

A custom range of Catalogs have been created to store the environmental data and the task information. They can subscribe to data and update themselves and can be queried easily and directly from the modules that need the data.

Finally, a custom set of pages have been created for PsyProbe which act like dashboards for various parts of the system. These are application specific and the MessageDataCatalogs are used for collecting, storing and



making internal data available to the dashboard pages. Some of the modules now also have custom tabs in PsyProbe where they can show internal data to help the developer debug and monitor what they do. This includes mini dashboards, streaming timeline graphs and live hierarchical diagrams, all of which can be opened in separate browser windows.



Additional Facilities of Psyclone 2

Psyclone 2 offer a number of additional features which can be used for the further development of the Collaborative version of the Cognitive Map.

Psyclone 2 introduces a new type of messaging called Signals. Signals are sent out by modules and are broadcast immediately to any other module that has subscribed to this signal. This offers a type of drumbeat which all modules can follow to synchronise their processing to happen at the same time. Like messages, signals can contain any number of data entries and carry large amounts of custom data.

New types of Catalogs can be created such as proxy catalogs that forward queries to Internet searches or accessing external databases, without requiring every module to know about any external dependencies.

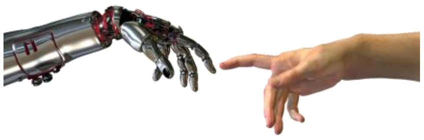
It is now easier to integrate with third-party software via Services. Services are web interfaces which can communicate directly with other components using whatever protocol they prefer, without requiring Psyclone or modules to know anything about this. This allows data to be output to external components and asynchronous input of data from external sources.

Modules can now be told to run in separate process spaces on a single computer. This means that if they become unstable or crash the whole container the rest of the system will continue to run and the container (called a Space) will automatically respawn and reconfigure the modules inside it to run again. Furthermore, all private data and state from each module in the crashed space will still be available after the modules restart.

Messages can now be tagged either manually or automatically with a unique identifier which will persist throughout the resulting chain of messages. This means that modules can receive the same data while still keeping track of who or what they are talking about, such as facial data about person with tag 15, kept separate from facial data about person 18.

Psyclone 2 also offers the ability to record particular messages and keep them in a database on disk. These messages can then be played back in the exact same sequence and temporal spacing when the system is run again and again, which provides an excellent way of reproducing scenarios during debugging and development.

CM LABS
Powering the AI Revolution™



ICELANDIC INSTITUTE FOR INTELLIGENT MACHINES
IIIM
VITVÉLASTOFNUN ÍSLANDS

Plan for Multi-Participant Architecture

The next steps include extending the current state-of-the-art system to realise the Collaborative Cognitive Map, using by a number of different and separate Psyclone systems, one for each robot that contains all processes and data relevant to that robot, and a single shared data resource called the Shared Social Database (SSD). Each Psyclone system will have seamless access to read and write raw data in the Shared Social Database and will be able to query and negotiate about higher level data, abstracted from the raw data.

The SSD will be a central repository of data and will have its own way of processing data internally and mediate between the multiple systems when they negotiate about data and tasks. It is not a central brain which directly controls the external systems, but rather like a central controller which facilitates agreement between the systems on ‘facts’ and how to manage tasks in a conflicting dynamic system.

Each participant (both human and robot) will be represented in the SSD and updates to this information may come from a multitude of sources, including raw data provided by the robots and interpreted data as results of internal or external processing of the raw data. This means that each robot can use the SSD to get information about both other participants and about itself, as perceived by the other participants.

The environment is mapped specifically in the SSD and information obtained by any participant will add to what is known about the environment. Additional data may come from scene sensors and other sources of data. If the raw data disagrees an arbitration process will determine what if any of the data may be valid.

Tasks are also handled specifically by the SSD and used to guide the participants in terms of what to do next. The execution of the tasks will be done by the robots themselves and the determination of which tasks to do next may happen by negotiation, if needed.

Interaction with humans will be done by the robots themselves, and high-level tasking such as information flow will be tasks that are delegated to individual robots. As such, the robots are working together collaboratively and still independently and will appear to rarely disagree in front of the humans – as such disagreements are handled by the SSD under the hood.

A participating Psyclone system will have its own view of the world which it will share with other Psyclone systems through the SSD. It will also receive information about the world as perceived by other Psyclone systems. Each Psyclone system will include a negotiation component through which disagreements and conflicting data can be resolved and it will be possible for the Psyclone systems to ‘agree to disagree’.

Tasks are handled in a similar manner where each Psyclone system is in charge of its own tasks. These are hierarchical and will break down into smaller subtasks which eventually are carried out, some in sequence and some in parallel. New information and task completion data from other systems (such as failures and interruptions) can cause the current task schedule to need re-planning or be abandoned altogether.

The same holds for interactions, both with humans and other robots. Since both are autonomous situations can arise where interactions will have to be re-planned or abandoned, such as if a human participant leaves or changes the topic – or if another robot experiences a technical problem.