The Grendel One project documentation and white paper

by Christopher Rehm, CSEG, M.B.A.

https://github.com/johnfire
http://computers.christopherrehm.de
http://grendelone.christopherrehm.de
christopherrehm@web.de

Revisions:

June 29 2017 – begun
Sept 26 2017 – added Grendel One website
June 9 2018 – updated hardware and software diagrams

Abstract:

A project to build a home robot with machine learning and Artificial Intelligence capabilities as a home learning project.

Index:

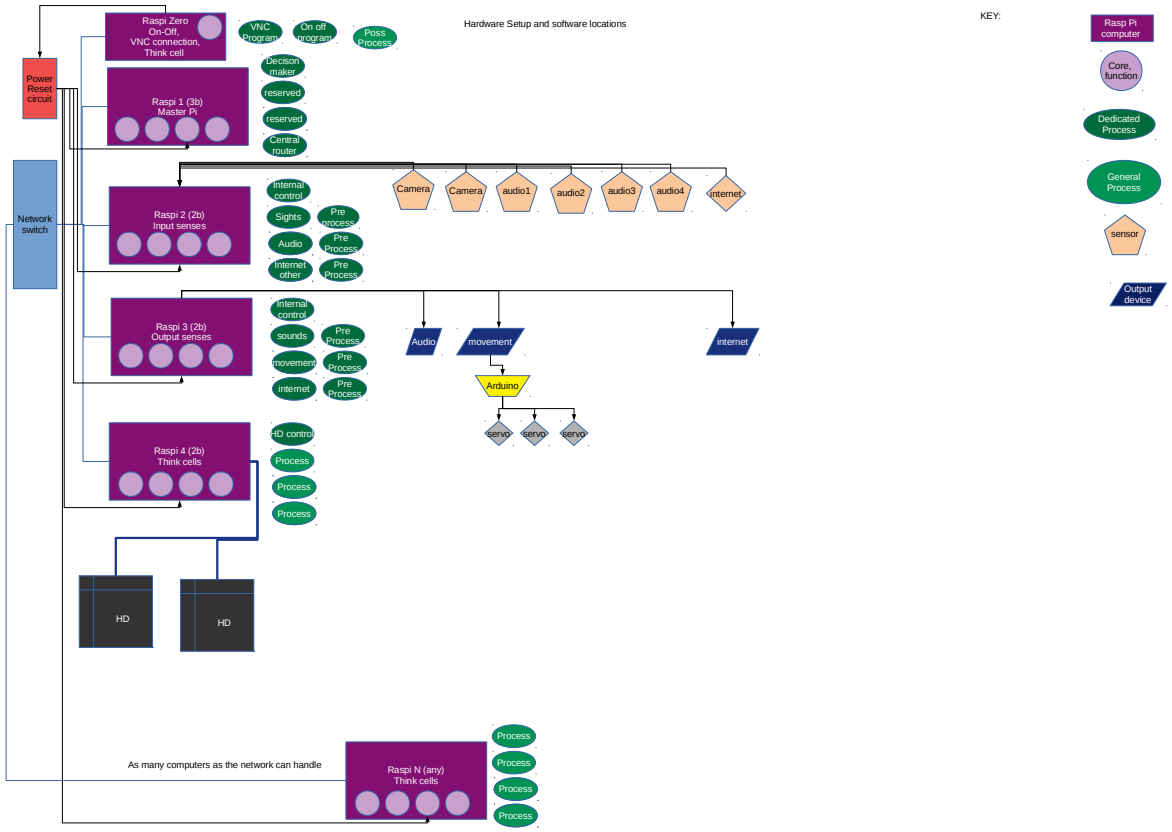
Why?

Robotics and artificial intelligence are a growing, burgeoning field at this point in time. The best learning comes from hands on development.

Concept and goals:

To build a functioning robot with both machine learning and artificial intelligence capabilities, and to see how far a home experimenter can go on limited research funds to develop such a system.

Computer system, hardware:



Money is a primary issue. The goal is to build a functioning device for the minimum investment. All parts, design issues etc come from that thinking.

Computers for cluster raspberry pis, each with 4 cores. Possible expansion with other experimenter boards in the future.

2.5 external hard drives for long term memory storage.

Off the shelf USB hubs, network switches, cabling, etc.

Need: dedicated network router for the cluster, so it can travel with the device where ever it goes.

Current set up:

3 Raspberry Pia computers mark 2 model B

1 Intenso 1.75 TB ext hard drive

2 off the shelf USB hubs, delivering 500 mA per port. One with 7 connections one with 10, both self powered.

1 off the shelf Ethernet network switch 10 ports

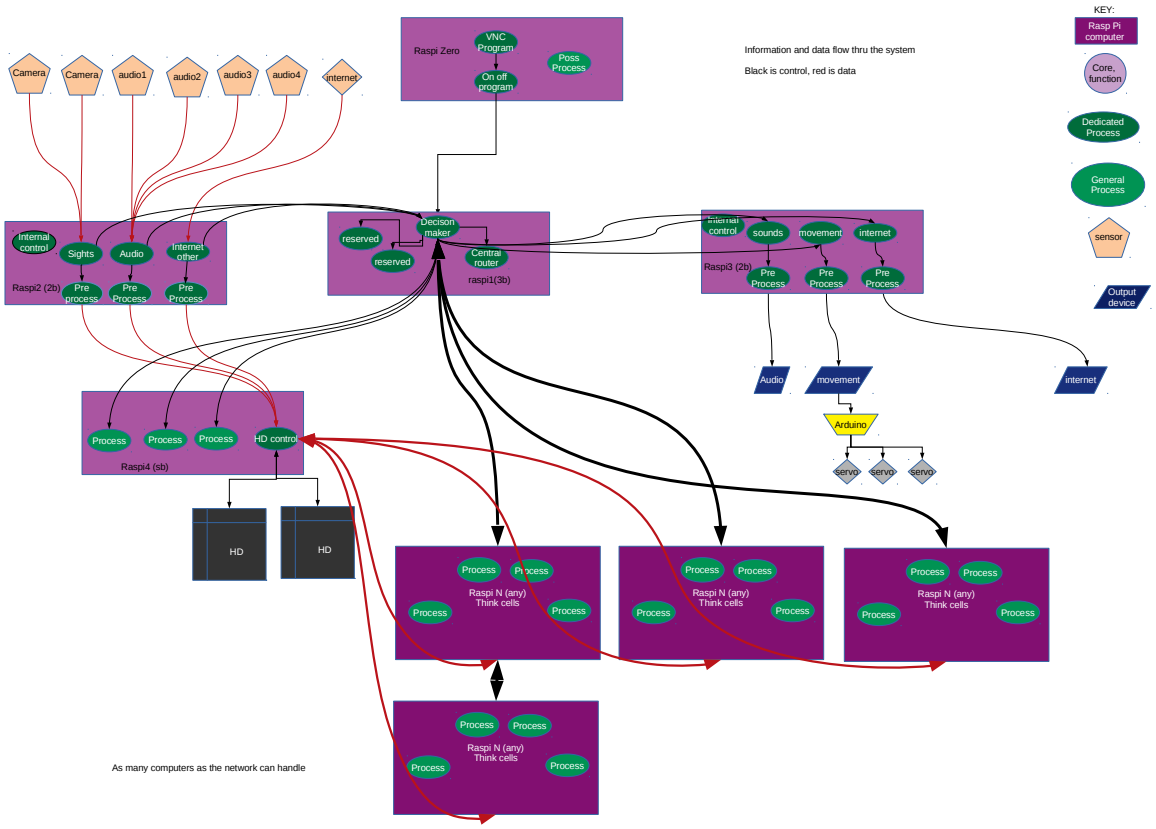
cabling as needed to connect everything.

Power consumption estimates: (to be used in designing battery functionality)

		Total Consumption:	7.120

Item	Consumption (milli Amperes)	Quantity	Total Amperes
rasp pi 2	0.400	3	1.200
rasp pi 3	0.740	3	2.220
hard drive ext	0.900	3	2.700
servos	0.100	10	1.000

Computer system, software architecture:



The above diagram is a conceptual representation of how the computer software is designed for this project.

The system is divided into two general parts. The first is the grendelbody program, which operates on one pi on 4 different threads, and manages both input and output from the device. The second part is an extensible system of Pia, where Think cells can be generated as needed and run on whatever processor hardware is attached and properly configured. A Think cell is a combination of a decision making function, a central message passing function and a number of analyzers functions to analyze incoming data. There is a master think cell that governs overall behavior of the device; all other think cells are generated as needed by the master for specific problems.

Each function contains a built in analyzer, which computes and records how long operations take when a functions various processes are called. This allows for easy monitoring of the various programs and for future optimization, both self optimization and external. Additional functionality is built into each object as follows:

- ability to write and read text files

- standard message passing interface
- internal analytics functionality

Interfacing:

The two software programs, grendelbody and grendelmind interface with each other by using socket communication between units. They can both access a common database for storage of data.

An apache spark server will be implemented to use multiple processor data processing. All computers will be configured as clients to the server, and the server will handle assignment of work packages to cores in various processors as needed.

Earlier work:

I started this idea about 2 years ago. The first year was spent trying to develop a MPI cluster computer using the hardware above, and writing a skeleton Java control program. Problems developed installing MPI express and additionally it was discovered that this does not easily allow for separate threads doing separate tasks as assigned by the programmer. Other projects consumed available time and I have only returned to this project in the last week. Given the problems and time consuming nature of the lengthy install to get a network cluster working as envisioned it was decided to go with a multi program architecture using sockets for message passing and a dedicated server / environment for analysis work, which allows for easier processing of all data used in machine learning and AI.

Link to the old software, never developed:

https://github.com/johnfire/Grendel-One

First version :

https://youtu.be/CYoaq0aXF0c

Computer system, software:

The software package currently consists of two main software packages grendelbody and grendelmind.

Grendelbody:

https://github.com/johnfire/grendelbody

This is designed to handle all interfacing with the outside world. It consists of 4 threads executing the following functionality

visionin(thread 1): a image capture routine
soundin(thread 2): a sound capture routine
internetinout(thread 3): an Internet interface
soundout(thread 4): allows the computer to play mp3 files or other sound
movement:(thread 4) specifies instructions for movement

each module/ function has a preprocessing function associated with it that does whatever is needed to take the raw data from the various input output modules and process it for further use.

Major messaging functionality is indicated in the given diagram by blue arrows. Each function does have built in messaging capabilities to allow every function to have the ability to communicate directly with all other functions but this is not a normal operating set up.

The computer running grendelmind should also have apache spark client installed, so that this raspberry pi can do additional data processing when cores are available.

Grendelmind:

https://github.com/johnfire/grendelmind

This is designed to do any and all machine learning and AI tasks.

The basic unit is a think cell, consisting of a decision making function, a neural network function to handle data processing, messages etc, and a number of analysis functions running concurrently on the apache spark environment. There is a master think cell, and additional think cells can be generated as needed by the master and its analyzer functions. The decision function runs on one thread and the neural network runs on a separate thread. Analyzers are assigned by the spark environment.

As part of the apache spark system, a server (Hadroop) will be installed as the central database for the device. All data will be stored in this one data base. All data should be read accessible by analyzer functions. Raw input data should be secured from overwriting, as well as all intermediate and final result data. Storage is cheap, it is considered more useful to save everything and then see what can be cleaned at a later date.

Apache spark environment:

https://spark.apache.org/

NFS environment:

https://help.ubuntu.com/lts/serverguide/network-file-system.html

Sockets Reference:

https://docs.oracle.com/javase/tutorial/networking/sockets/

programming environment:

Netbeans 8.2 , open JDK 8 SE. also Perl and Python 3, Emacs 24, Linux Jessie operating systems, both full and lite versions.

Programs are created on a laptop and downloaded by Netbeans to individual Pia using Netbean's built in remote execution feature. All Pia are configured with static IP addresses.

Common objects that are used to derive all others:

basic object:

- has built in self analysis capability to analyze time usage of various methods.
- Can read and write .txt files
- messaging capability send and receive

message object structure:

- stores ID from sending object
- stores ID from receiving object
- stores an action code, what to do with data
- stores data

action codes:

- save data
- load data
- process data using XXX

visionin:
captures raw photo and video data

visionin processor:
saves raw data to database, informs think cell new data is available

soundin:
captures raw sounds from environment

sound processor:
saves raw data to database, informs think cell new data is available

Internet:
sends and receives raw data from FTP, HTTP, ssh, etc

Internet processor:
saves raw data to database. Informs think cell new data is available Sends output data to correct Internet location. Receives requests from think cells to get Internet data. Accesses correct address

soundoutput:
plays sounds thru robot microphone.

Soundoutput processor:
stores locations of data to be played, manages play list

movement:
sends movement commands to Arduino

movement processor:
TBD

decision object:
makes decisions at a given level in the hierarchy.

Router:
passes messages to correct location, manages analysis projects.

Analysis object:
specific objects to be used to analyze data

think cell:
specific combination of a decision object, router and associated analysis objects, possibly other think cells.

Skeleton, body:

Currently this consists of a framework that allows for two degrees of movement, side to side and up and down, for the vision and hearing parts of the device. Future plans include expanding the device to a full body with ambulatory capabilities and some kind of robotic arms so as to allow it to interact with the environment.

Skeleton and body hardware:

Current device:

Meccano head, with two degrees of movement.
2 model airplane servos for movement control.
1 Arduino Uno for servo motor control. Currently not interfaced. Plan is to send signals via SPI or i2c interface from a dedicated raspberry pi thread in the grendelbody program to the Arduino and the Arduino Uno generates PWM signals as needed for the various servo motors.

Appendices:

OLD sections, now replaced:

1. hardware software diagram, not clear: