3 page pilot

Attempting a cat

- **P1P2. Vision Oriented Hippocampus**

. tags object.
tracks object

Receives CYCs
attach CYC + object

produce SIMs
solves and update

- done within time limit

Updates parameters every SIM solution

P1P2 describes a Mapping Trio and a Watching Trio part of the

Vision Processing (which comes after receiving vision input)

Main goal of this part is to combine labeled and tracked data

Onto CYC-related relevance (CYCs are behavioral drivers)

We then try to make further parts

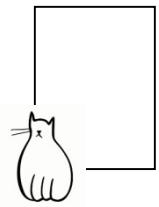
(Beacons & Bev CYC)

Which are reproductions of (emulations of)

Social cortex & Epigenetic Cycles

Beacons

are social cortex emulation



cat vs pulses + pulsing table

- tags object
- have them pulse
- have a pulse table
- on how to response
- or hedge
- or prepare

separate types thru activity and uncertainty

Then it creates a node for hedging signals

These signals are akin to

/ frequent barrage of possible hedges

That we treat / alter our behavior on;

Given our awareness of such objects

Notable one include a object type

which represent the Self

we then

Bev-cyc

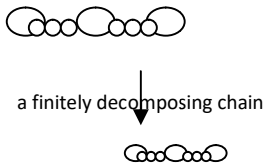
- Sequence movitation system (Which is core driver of the agent)
- These are simultaneously the action-sequences
- (following ROS)

- Depicted via a chain
- of Decomposable Nodes

  decomposed until muscle % tension

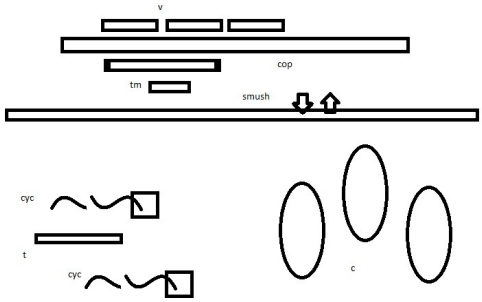
  solved per time limit and

  updates policy for best outcome



onto

DEMO (demo 0.01 .doc)



Attempt to make this

- Cyc goes forward;
- Cyc + modulator
- Serviced by P1P2 Nodes and Themes
- Themes are SIM space of P1P2
- Smush those two
- Onto an action bar
- For resolution within time limit

Output looks like this

| n n n n n n | a block of CYC divided onto 6

Those blocks of n are filled with the processing / update calls of above parts

(vision nodes and themes and cyc resolution)

Then the modulator would decide for every | n n n n n n | block

Which one goes to follow and which decomposition should happen

We smush first;

we learn as it gets graded afterwards

And we optimize these simulations further later on

Converting them onto Active Inference Formulas

(they then are called quickcalc)

Summary:

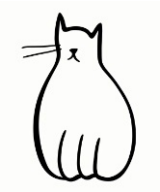
- Get vision input
- Label , tag , process
- Combine with CYC
- Becomes SIM
- Learn SIM outcome
- Update

Yupp

HOW DO YOU EXPLAIN A WHOLE CAT

IN A FEW PAGES

??



YOU USE BEV-CYC :D

WAKE > HAPPY > SLEEP

Decompose HAPPY*

(as an example~)

(it can branch onto many things!)

Fill Simulation w/ Vision Nodes

Special mention to Beacons & Social Cortex

- **Finally**

We smush the nodes

Both labeling & solving

Per time limit

Abstract

Hello, this paper is about

Amateur attempt at Cat-making

We'd like to create a cat

(a decomposable action sequence agent)

(using ROS style action checkpoints)

Describe the Specs

What is the scope of this cat?

We are attempting a

- Agent cat that

- Starts with decomposable action-seq
- Starts with vision logging for all objects
- On to attaching each objects with BEACONS
- On to processing each beacons and a reflexive system using Vision
  (map trio ; watch trio)
- On to ending up in either a SIM

  Or a QUICKCALC solution / variable update

- Quickcalcs are a way to simplify the simulation run onto a FE % calculation
- Via active inference.

- Methods are currently handwritten, optimization still on work

Contents:

What is to be expected?

Pilot will jump onto a

BEACONS & BevCYC

Representing a emulation of biological concept of

“SocialCortex” & “EpigeneticDrive”

(inherited & learned)

We will move on to a DEMO 0.01

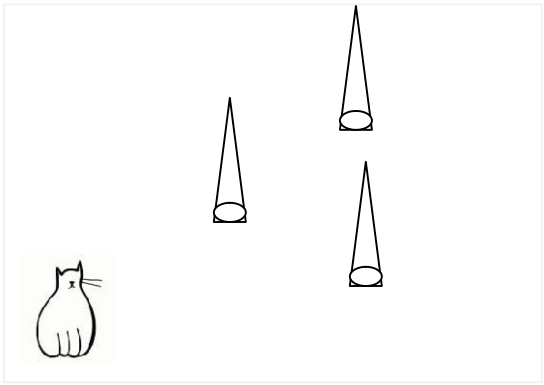
Then recap

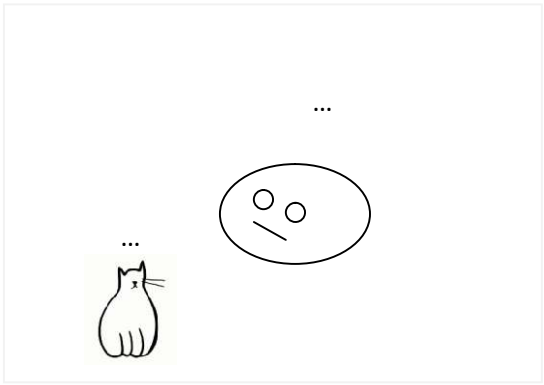
Chapter 1

Step 0: social cortex that handles] cat's general policy vs each object

- **BEACONS.doc**

Pic:





Pic1. Cat looks at objects
Objects are pulsing!
Do we hedge? Says cat

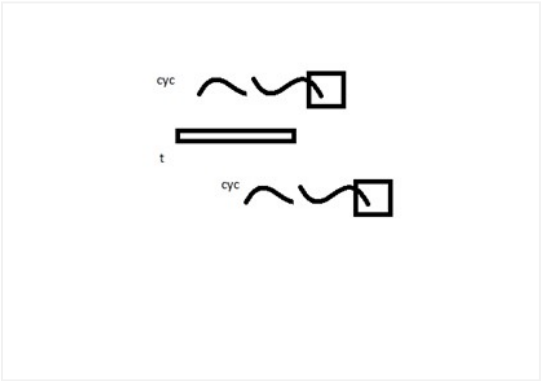
P2. Looking at owner
Owner is super pulsing!
Do we kungfu? Thinks cat

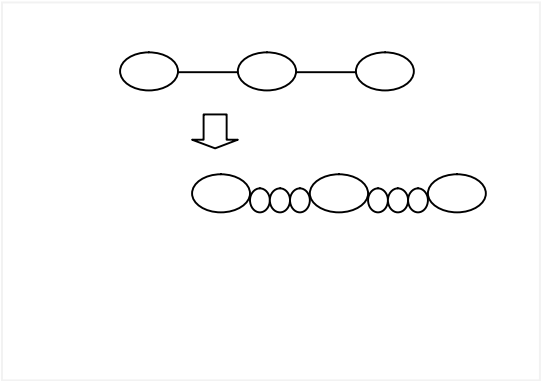
SocialCortex –esque emulation;
thru attaching
PULSES (+response table)

How –nodes- of P1P2 gathers the tagged object
And puts them onto a bucket

. to make in ROS .
all SIM ENTITY are graded by significance
We then assign a pulse beacon type
& reply with response

- **BEV-CYC.doc**





Pic 1.
CYC (getting transcribed)
along with all the other processes

a modulator is then deciding

which is most important; and how

does each update through learning

Pic 2.
CYC is described
in 3-3-3 action sequence

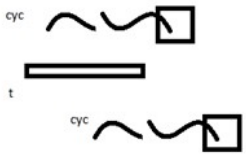
(ala ROS for this attempt)

this is decomposable

to the smallest degree which

doubles as a muscle % decision

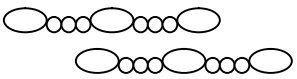
Starts with



CYC pool; cyc modulator

Cyc becomes chain

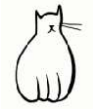
Cyc chain gets decomposed



Cyc gets processed thru

SIM/quickcalc;

Each results in a micro action



recalls

Updated via SIM

run .perbit.

-

BEV-CYC is

Epigenetic-esque emulation;

by describing a cat's main drive

(inherited to then learned)

And the process on how it is decomposed

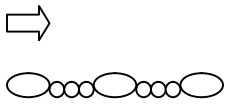
These are all described and attempted in ROS / Robotics term

With each of these BEV-CYCs

As a chain of nodes that each node can be decomposed

All of them are first inherited (hardcoded from birth)

And all of them are then learned (edited as the cat goes thru the age)



modulate. execute (%)(%) . Update (Learn)

modulate again. Ex. Update (again)

Decomposable action sequence chain

Gets handled by an LLM/agent to refine

By which and how to update

Sequences are made on form of state-state-state

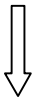
Hoping to finally break it down to

The smallest nodes which simultaneously represent

Muscle % tension (is already an action)

.

For example: a yowling cat



(social) BOGUS(uncertainsoc) - ACT – OBSERVE

(social) BOGUS-YOWL-OBSERVE

(social) BOGUS-YOWL (BOGUS-MUSCLE-OBSERVE) - OBSERVE

The above chain is a hypothetical social cortex related action

Which is a BEV CYC that exist as a snippet for when

Cat spots a presence of engage-able (uncertainsocial)

(these conditions are later set – via a prior related Agent Node / processing node)

When it decides to YOWL & OBSERVE

YOWL is then decomposed onto smaller and smaller nodes

YOWL in itself is not the smallest;

YOWL likely (just like in robotics)

Include steps such as : adjusting ; engaging ; provoking ; observing ;

YOWL Y- YOWLING – YOWLX

YOWLY – (YOWL YOWL YOWL*) - YOWLX

Construction of these

tree-of hardcoded behavior are

Initially handwritten; with them being associated to each its own SIM type

(type of Game simulation to resolve)

In this game sim;

there can be learning where the decisions matter

But overall; for decomposition it always first tries the same thing

Upon failure; we only then take the failure signal

(these are considered Chem in Chempool)

Lets label this "molecule to insist on learning if possible"

Lessay to be called "PON-###" (ponder)

For a deliberate learning behavior

As with the Chempool description on P1P2

And how the practice of converting highest CYC onto a Module

Such module will then (to publish)

Have its own Chempool for watching this PON-###

PON-### will be passed to another module

(currently we are using

ModStream

ModStream records this as a (one of many) queued

This queued THEME comes from the

PON-### origin

So thus it gets resolved also the same way

1. Run the modulator for CYC-search
2. CYC-search hits with the latest Cluster of Info
3. (should be a cluster)
4. modStream publishes a latch (modLatch)
5. (note Latch = nested SIM (forced 3 roll))
6. modLatch maintains this Latch

imagine modlatch as a Node that assigns an Agent for each Stream / Latch

(in other words; a concurrent call for Simulations)

7. giving it satisfactory % - (floating satisfactory % to judge a latch ending;)

8. every Latch (SIM) is processed with TIME LIMIT

9. the agent has a BUDGETING TABLE
10. for how long this activity should usually take
11. (imagine a depleting [100%] getting depleted by % % from factors / inputs
12. That it accounts to deplete faster

Anyhow. This is the part where we mention | n n n n n n |

Where each n

is forcibly filled with both a vision-node update; and a THEME update;

an action- and

a next-cyc prescription;

Thus we are just filling each n with processes that have to happen (or be queued)

And the subsequent judgement of each node

Represent the agent's action of choice

Usually this happens

In terms of YOWLING

By this way

From before

YOWL Y- YOWLING – YOWLX

YOWLY – (YOWL YOWL YOWL*) - YOWLX

Imagine each of those

Decomposed YOWL

Are a | n n n n n n | node of CYC

Where at this lowest decomposition;

Its just "more muscle tension" or "less muscle tension"

Determined by the | n | that we fill for each

(what the cat actively sees and thinks)

(an n for vision update

(an n for theme solution

(an n for cyc modulator

(an n for checking* (mentioned in DEMO doc))

Or what have you

These are scaffolds that are called by the

CYC-MODULATOR

It calls them in form of | n n n n n n |

With what each n contains

And its cached | n | resolution table

We are not good at automizing

Or making a good math formula

For consistency!

But we are pretty certain we are able to make

Full behavior table for any video / activity that is requested!

(Sir/Ma'am can test by giving a link for me to decompose at spot)

(Even if wonky; it will probably only take one or two polish to make it right)

Anyhow back to BEV CYC

Summary; is that we have decomposed nodes

ROS style with action checkpoints

(for completion or failure)

That is : 1. Forcibly resolved per time limit

2. Each resolution results in learning (spontaneous)

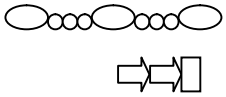
3. Even if a PON-### chemical is triggered and logged

4. such will be dealt separately (a queued chem.)

5. depending on the identity's breathe & habit

(which is also a form of CYC with its unique completion Tree)

For YOWL



Will then lead to further decomposition

(YOWL -> YOWLING muscle %)

Check method at BEVCYC . doc

For current plans on checking outcome

(step-check-step)

And how this chain runs separately from

AC (the main executable)

There's also a lot of Copying (Entire CYCs)

Or re-refering to existing & live CYCs that will be involved

We are hoping to describe these manually in site

(show video; compose; retry;)

DEMO 0.01.doc

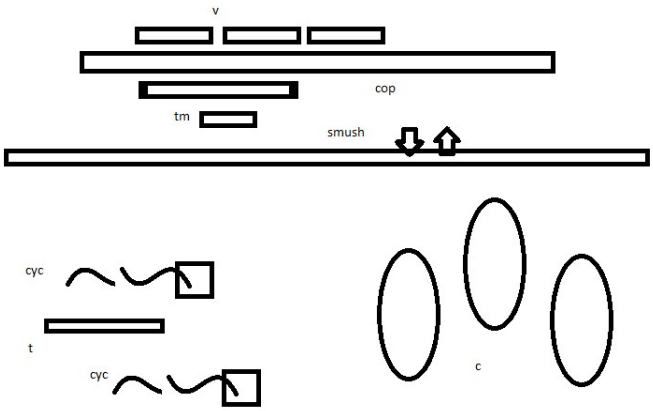
Almost a recap; for this DEMO

Hoping with a help of external

ROS + VisionModel

For this attempt;

We aim to create this



(1)

1. CYC is there; with its modulator
2. (small t)
3. © is a container for
4. Object types / etc , from vision

(2)

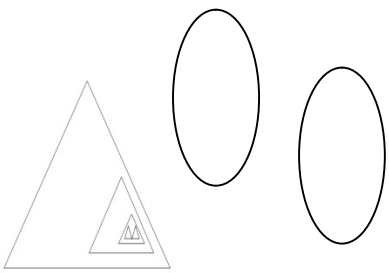
1. Run visionNodes (P1P2)
2. Starts logging things goes onto ©
3. CYC runs along; plans; drives
4. visionNodes react; modLook & modWatch
5. starts focusing on things that matter for such
6. by “starts focusing”

(3)

1. Is already a CYC-node
2. And it already becomes a SIM
3. > Because all of these processes are time-limited
4. Each of them will always gets resolved per Breath(n)
5. Time for each breath period

(4)

1. As described in P1P2
2. Each SIM resolution
3. Always end with parameters update for both the CYCnode
4. And the objecttype
5. After it happens several times (rule yet to be defined)
6. Body attaches a NOVELTY % to that SIM type
7. Below novelty% becomes a Quickcalc
8. Which is a optimization method for us
9. To simply convert the gameworld SIM onto a Formula
10. With FE %-% (range)
11. As a defining property
12. Then nested along with its associated
13. Cyc nodes



(this is the triangle alongside the Containers ©)

(all of them are just update-ables working data)

(5)

1. In terms of agent running
2. In a Gymnasium
3. This starts with the Agent having a prior Run (prep) of the Territory
4. If it hasn't done that before; it will prep one quickly
5. (with the Templates of known environment)
6. Same with objects
7. Refer to P1P2 for the agent arrival for these
8. It will then quickly call the associated objecttypes
9. Of that particular environment
10. Whilst the VisionTrio & ScanningTrio
11. Runs per always; always resolving to SIM

Early Summary:

- Guy started on P1P2 making the vision nodes..
- We then suddenly jump onto the more fundamental
- ... (on which the reactivity & processing of these nodes will apply)

then we go onto the more basic (epigenetic & social cortex)

- we Beacons
- A protocol for reaction vs entity in the Simulation

- we Bev-Cyc
- A decomposable chain from birth to update as cat learns
  Currently depicted using ROS-style
  Action-Checkpoint-Sequence

- We Demo 0.01

(we smush n run |n n n n n n |)

Onto..

Demo & P1P2 up

Some mention on P1P2

Final Summary

Cat Gist

So what is the project like when proposed?

Reviewing the lingo;

We aim to; make CYC run as a from-birth habit

(wakesup; tries to)

aim to; gather n resolve

along vision nodes (P1P2)

we then smush for ordering

and simply run them per

each time-limit

→

Cyc goes forward

→ Vision follows after

Themes Come →

Gets Smushed

Runs per bit-by-bit

Updates bit-by-bit

For longer learning;

1. Kitten has engaged in a high-stress
2. CYC updating session
3. For many days.. (with many PON-###)

(we aren't clear on this yet, refer P1P2 for the prep)

4. Kitten & Cat both has a self-referential
5. (re-bagging everything & process in mind)
6. Process; mentions briefly in P1P2
7. Mostly this will happen by Copying
8. The AC and the STATES
9. (whole described Demo0.01)
10. And having a separate (long)THEME
11. Insist upon these lingering
12. THEMEs to resolve

Cat.	- is a cat
Crabs.	-agent/llm like modulator of input & output
	Cutter mentioned in AC P1P2 is this
CYCs	- sequence from inherited to learned
Crabs.	- more cutting! (modulating CYC)
Vision.	-ScanningTrio (scans env.calls.watches)
	-VisionTrio (applies processes to objects)
Container.	Logs the entries
	ObjTypes; EnvirTypes. etc
AC.	Becomes the smushed thing above (demo)
Theme	also known as SIMULATION (nested) ; from (P1P2)
Smushed	whence any input is time-resolved onto one
Updates	Updates every SIM / quickcalc resolution
Triangles	Optimization onto FE range

x

P1P2 recap

So whats there?

P1P2

Pretty much just starts with

Giving a High Overview

Of the Modules & Vision Nodes

Back then this was made to represent "Vision-Oriented Hippocampus"

Hmm.. I guess this was correct

(its more like Sensory-collection-and-combiner-with-CYCs)

Anyway.

We describe P1P2

With some terminology

- Screens
- Chempool
- Modules
- Sims
- Themes
- Latches

We describe them now! (sry just minor gist)

All of those is important for Robotics purposes (porting this onto ROS)

Screens

Are **transposed** Vision input;

We recognize that this process is actually also a

CYC -> SIM -> UP -> CYC

But for this overly frequent components

(much more frequent than 6 nodes per sec)

(pretty much constant)

We simply denote a Screen

Which is a transposed Vision Input

Its like a constant-lingering-subconscious awareness depiction

Sample of notable

- Combiner of each eye's vision input
- Combiner of vision+other senses to default presence space
- Tranpose of that vision space onto different scales
- Automatic having 3/4/5 (arb number) of such scales

aaa

- For Bat, lessay they'll have one for the whole cave
- For Salmon; lessay they'll have one for direction to their spawning pool
- For Whale; lessay they'll have one for a whole ocean
- Most these are hardcoded by biology
- (its actually a CYC but its so lingering that it makes a structure in the brain)
- Anyhow; we simply denote important Screens

Anyhow back to the notable

- A 3d room space awareness (Transpose vision onto Overhead)

- An Overhead view
- A Screen for ENGAGED-Social-Object
- Refer to BEACONS
- For socialobjects

- Cat will assign a LiveScreen representing
- How the cat thinks this creature can move
- (this can be done using a THEME / SIM / CYC / AC-Copying)
- But after realizing for the size / scale of the difficulty
- We realize its better to just
- Hardcode a screen for some crude automated
- Representation of known moves

(consider any SocialObject-types to always come with a Screen)

Representing emphatized – condition

(programming wise; we then simply hardcode what this SIM will be given a Focus%)

(this is much easier! Than emulating all the steps on which this empathy will increase

Or describing how an empathized object (any object can be felt this way)

Grew from a ObjecType to a

modTude to a test

To another sampling to then a AC copy for resolving Queries

on

How dangerous / how unpredictable / source of such

We can actually do the above steps manually!

Simply describe how a cat is trained to follow the strongest signal

(CYC-sample) - (almost always persistent every XX second)

Which is a person waggling his hand / finger weirdly

Then we can draw how a cat kept

getting his

CYC-(sampling) to always have to include

An EMPHATIZED human in his THEMES / Simulations

Then fill those EMPATHIZED Themes

With known information of the person

Then confirm these cues bit by bit

Given how the person's subsequent movement does

IF we wish to avoid this!

We simply draw the outline of the Cat's Pondering

(just

a PON-### for the social Object

a blot/censor for deferring to this Screen <

a input-### for taking from that Screen! <

(on the deciding THEMEs / cyc-modulator tree)

(on deciding the | n n n n n n | fills)

so we could cut out most of the manual-empathy-checking

for person to person clue-confirmations

to simply a hardcoded

Screen; with known typical fillable information

Chempool

Chempol is an attached Display to a module

To track the input and timing

Of Chemical Signals / Known Molecules / Hormones

That dictate a certain rise / fall of certain calculations

Usually this is done when describing the Module

We describe a Module

Then we describe what Screen & what Chempool

Does it carry with it

There's probably some sort of advanced

combination Pool

That we have to make, to port for biology

But for now! This simple tally will do!

For stresss / satiety / what have you

Modules

Modules are independent nodes

With a lot of independent processes

Attached to them; run entirely separately

Guy used to think to program a cat!

He needs a help with + 36 pc (>_<)

(such backwater counting standards)

These pcs! Will have its own Process watchers

To Constantly process inputs

And talk to each other

These modules are defined / enshrined by their

CRITICAL OUTPUT

As in; these are the grades of processes

On which if its missing; cat is considered highly crippled

Or just fail to function

Modules are based on

Hidden Organs that may be

(hardcoded critical structure)

Hidden in the brain / suchlike

One cheating (or not cheating! Just not quite defined yet)

Module ; is when we make a rule where

The HIGHEST running CYC

(or several of them)

Should have its own MODULE

(instead of having a THEME call constantly going on)

(simply have a Module to produce its critical output for the

Current activity)

I.e. When in forest; Roaming; becomes a module;

modROAM

with a attached Chempool + Screen

Seems weird to have a special rule like this

But perhaps all of these nested structures are just

Bigger / more attached form of Molecule;

And we are simply drawing some scales / sequential processing

As much as we need to produce a consistent agent

BTW. Ontop of that ONE MODULE

Where its actually a graduated THEME

We have:

- ScanningTrio (sometimes referred as MappingTrio)
- VisionTrio (sometimes referred as Watching/TrackingTrio)

Scanning trio are three boxes that deal with environmental data processing

Vision trio are three modules responsible for object data processing

One scans; one routes; one maps

One watches; one looks; one tudes (tests)

Refer to P1P2 for each

Of their individual operations

And what they require and produce

To the Container

SIM/THEME

We also have a special two combo of Modules

modStream and modLatch

one deals with THEME producing (from CYC to theme) / (trigger->cyc->theme)

one deals with LATCH maintaining (from theme(latch1) -> subsequent latches -> resolve)

They deal with prescribing the appropriate GAME SIMULATION

Their contents; their abbreviation (by novelty%) and their update

AC / BM1 / BM2

P1P2 also mention

an AC

Which is the same structure of

SMUSHED |n n n n n n |

Tracker;

Getting its queued CYC|CYC|CYC

To be accomplished

Meanwhile BM1 & BM2

Is a separate, independent

Module of the AC

That is supposed to be dealing with:

1. Checking for predicted outcome
2. Checking for limits

(re-checking "is this what should've happened?")

and (budgeting "how much can I handle before I freak out?")

>

Some of the pre structure for this

Is described in DEMO 0.01 doc

As a next-in-line update

-

Refer to docs:

-DEMO 0.01	for project doc and planned update
-Beacons.doc	social cortex emulation
-BevCyc.doc	epigenesdriver emulation
-P1P2.doc	small update