

IBM Cortical Learning Center (CLC)

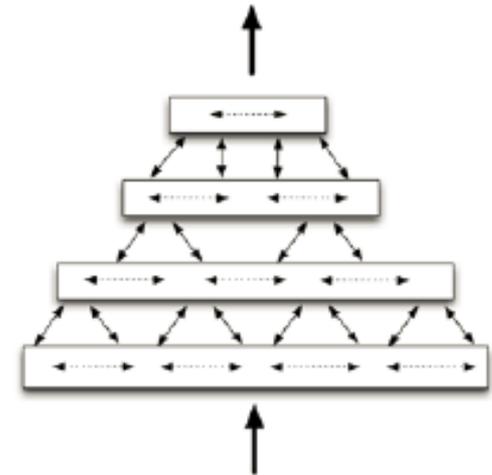
Winfried W Wilcke & CLC Team

winfriedwilcke@us.ibm.com

February 2015

NICE III Workshop

Albuquerque, NM



CLC Team Members in IBM Almaden and Yorktown

- **Wayne Imaino (Mgr)**
- **Kamil Rocki**
- **Chuck Cox**
- **Campbell Scott**
- **Bob Shelby**
- **Pritish Narayanan**
- **Geoffrey Burr**
- -----
- **Arvind Kumar**
- **Janusz Marecki**
- **Sharat Pankanti**
- **Kawan Bas**



IBM Almaden, San Jose, CA

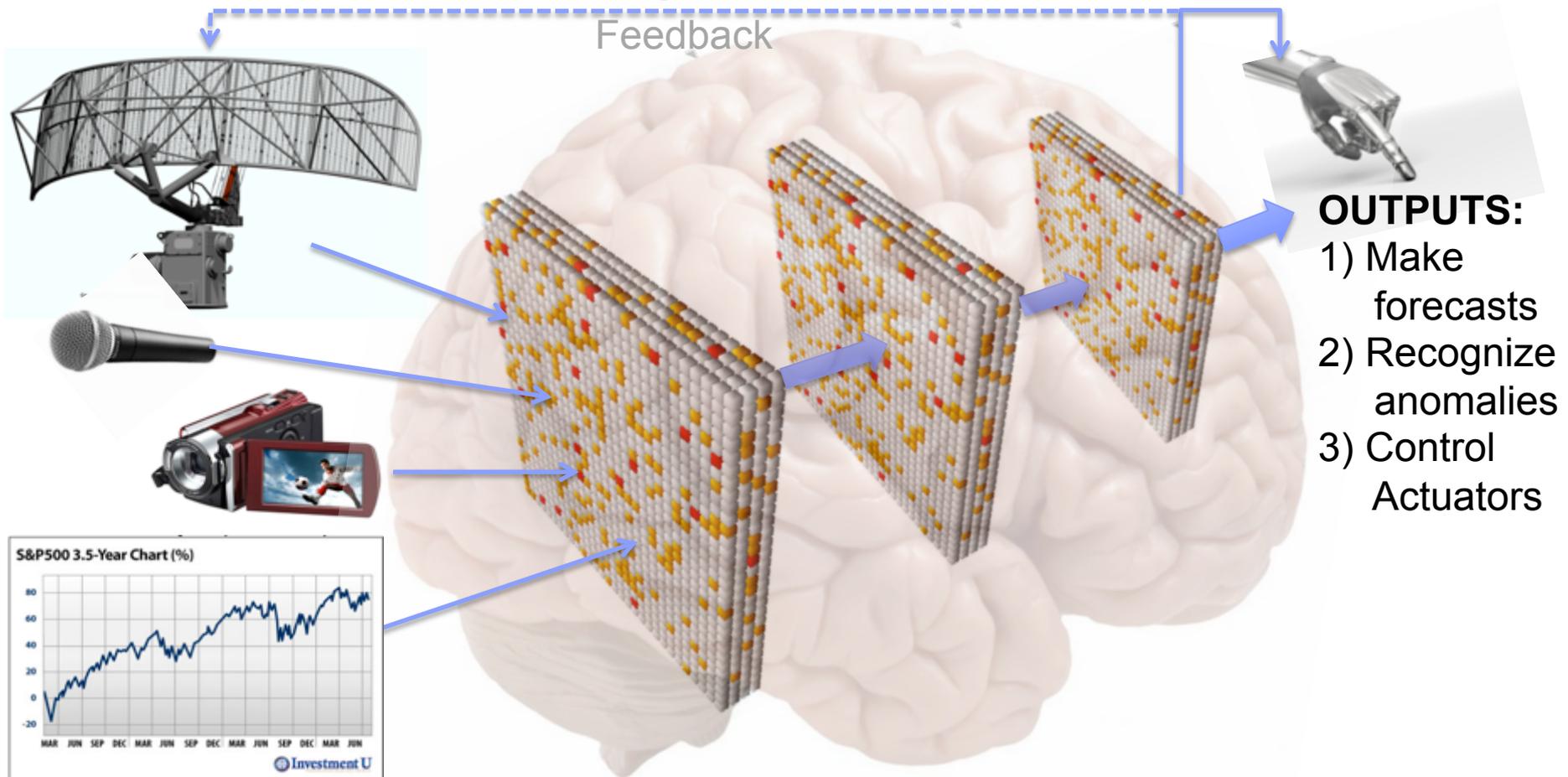


IBM Watson, Yorktown Heights, NY

Agenda

- **Cortical Learning Center & HTM in IBM**
 - Motivation
 - Technical Focus
 - Applications
 - Hardware
- **A Phase Change Device based neural net**

HTM Basic Concept



INPUTS:

Any type of spatial-temporal data stream

CORTICAL LEARNING ENGINE:

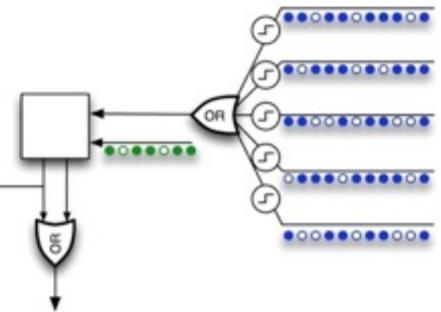
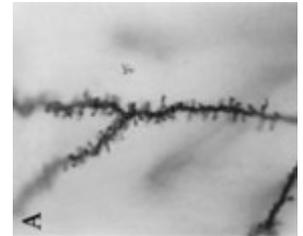
- 'Sequence Memories' form a *self-learning* system
- Detect and predict patterns in the input streams
- Based on 'Hierarchical Temporal Memory' theory

Comparison to other IBM cognitive projects

- **CLC – focus on *learning***
 - unsupervised - continuous - online
 - *Hierarchical Temporal Memory* as core cortical model
- **Watson is a rule-based Artificial Intelligence System**
 - is based on transferring existing expert knowledge into databases and applying sophisticated searches
 - very human labor intensive
- **Synapse**
 - is an energy efficient hardware engine for executing neural algorithms, using a spiking network
 - learning is done off-line in external computers

Why HTM?

- **It takes a ‘system level’ view of the neo-cortex**
- **As simple as possible, not simpler**
- **It’s build around unsupervised / online learning**
 - machine intelligence, not machine learning
- **Hits a sweet spot in biological fidelity**
 - learning occurs through formation of synapses rather than via (unrealistic) fine-grained weight changes
 - Hebbian learning, but no Spike Timing Dependent Plasticity
 - active dendrites
- **It is a rather universal model**
 - conceptually it always does the same
 - what it does depends on the sensors/actuators it is hooked up to
 - “no” need for new software for each new application



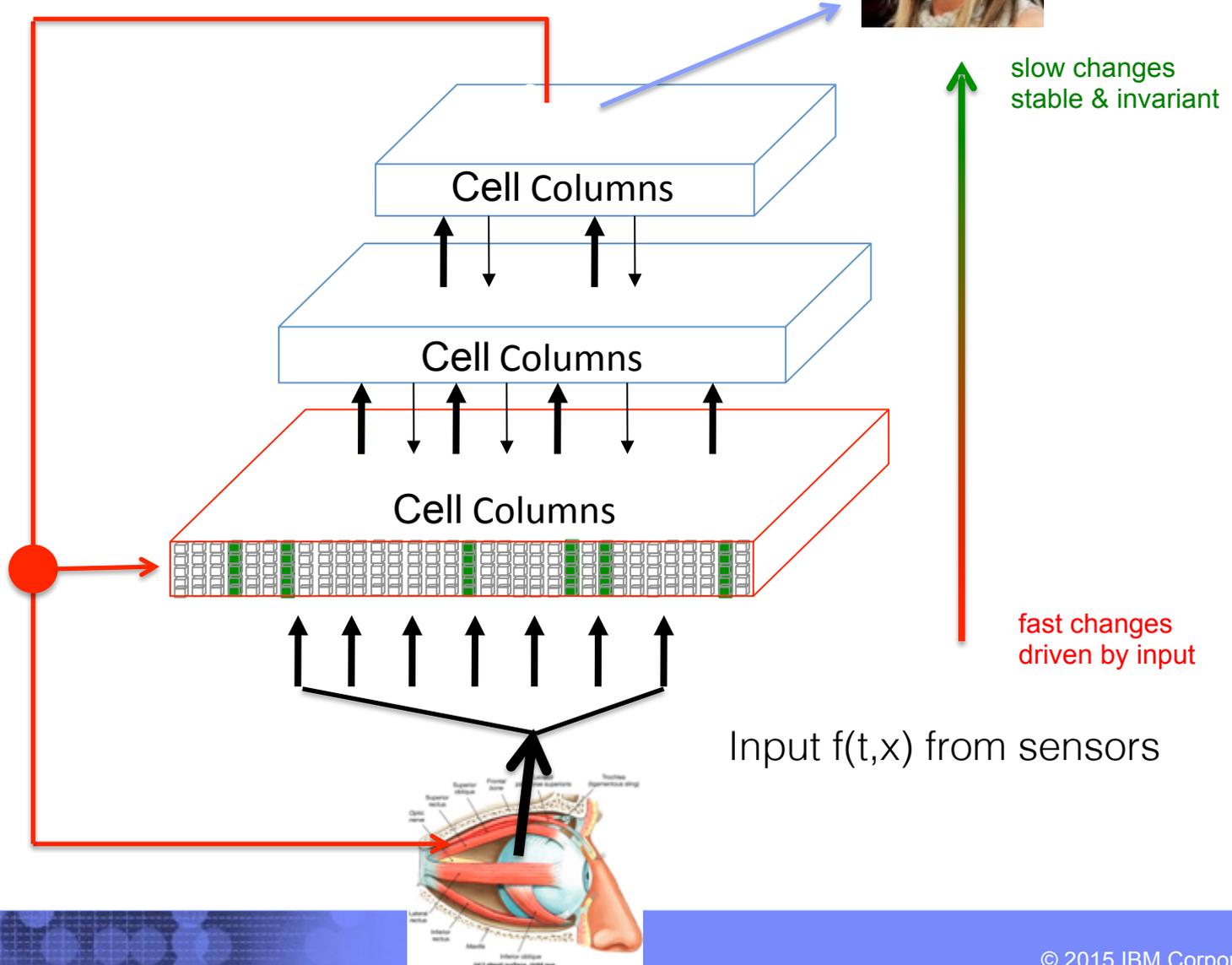


HTM Concepts

Output – recognition, predictions,
& motor commands

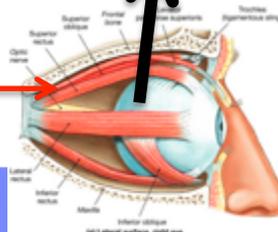


motor
commands



fast changes
driven by input

Input $f(t,x)$ from sensors



Why HTM - continued

- **Powerful data structure: Sparse Distributed Representations**
- **Time is a first class citizen in HTM**
- **Very few parameters**
- **Complexity captured in hierarchies**
 - very similar building blocks at each level
- **Cautionary notes:**
 1. Purpose built, fine-tuned machine-learning algorithms may be better for specific tasks (today)
 2. Some industries don't want self-learning systems
- **But we see a big potential for HTM**
 - for near term applications
 - as a plausible model of the neo-cortex
- **➔ Formal collaboration with Jeff Hawkins / Numenta**

HTM Applications of interest to us

Initial HTM applications

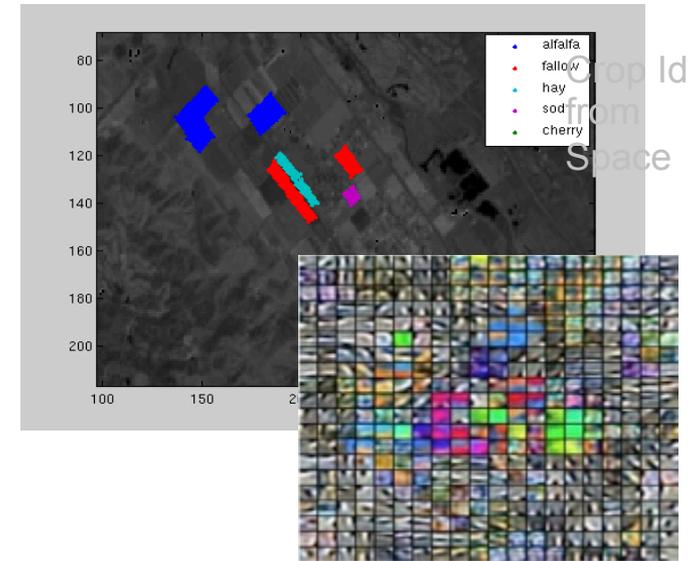
- Anomaly Detection for Datacenters (Numenta)
- Crop Analysis from Landsat 8 data (IBM)
- Subvocal voice analysis (IBM)
- Rotating machinery diagnosis ()

Longer Term Business Opportunities

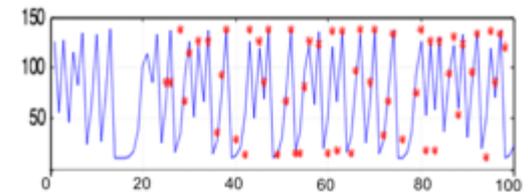
- Text Analytics (spatial and/or temporal)
- Anomaly detection for Cyber Security (Numenta/IBM)
- Internet of Things - vertical sensor data fusion
- Call center ‘Dialog predictor’
- automating STREAMS

Research Applications

- Saccadic Vision
- Semantic Text analysis & Machine Comprehension



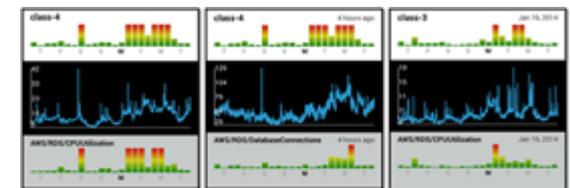
Saccadic Vision Base



Chaotic Time Series Prediction

Self-Learning

* Prediction (4 time steps out)



Numenta

How is IBM Streams Being Used today?



Telephony

- CDR processing
- Social analysis
- Churn prediction
- Geomapping



Transportation

- Intelligent traffic management



Energy & Utilities

- Transactive control
- Phasor Monitoring Unit
- Down hole sensor monitoring

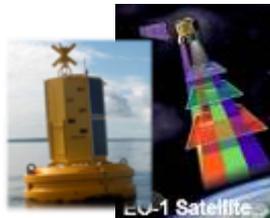


Health & Life Sciences

- Neonatal ICU monitoring
- Epidemic early warning system
- Remote healthcare monitoring

Law Enforcement, Defense & Cyber Security

- Real-time multimodal surveillance
 - Situational awareness
 - Cyber security detection



Natural Systems

- Wildfire management
- Water management

Stock market

- Impact of weather on securities prices
- Analyze market data at ultra-low latencies



Fraud prevention

- Detecting multi-party fraud
- Real time fraud prevention



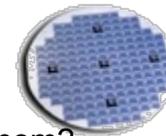
e-Science

- Space weather prediction
- Detection of transient events
- Synchrotron atomic research
 - Genomic Research



Other

- Manufacturing
- Text Analysis
- Who's Talking to Whom?
- ERP for Commodities
- FPGA Acceleration





Terminology

LAYERS

- 5 layers of distinct cells columns in biological brain
- each layer performs a different type of operation

LEVELS

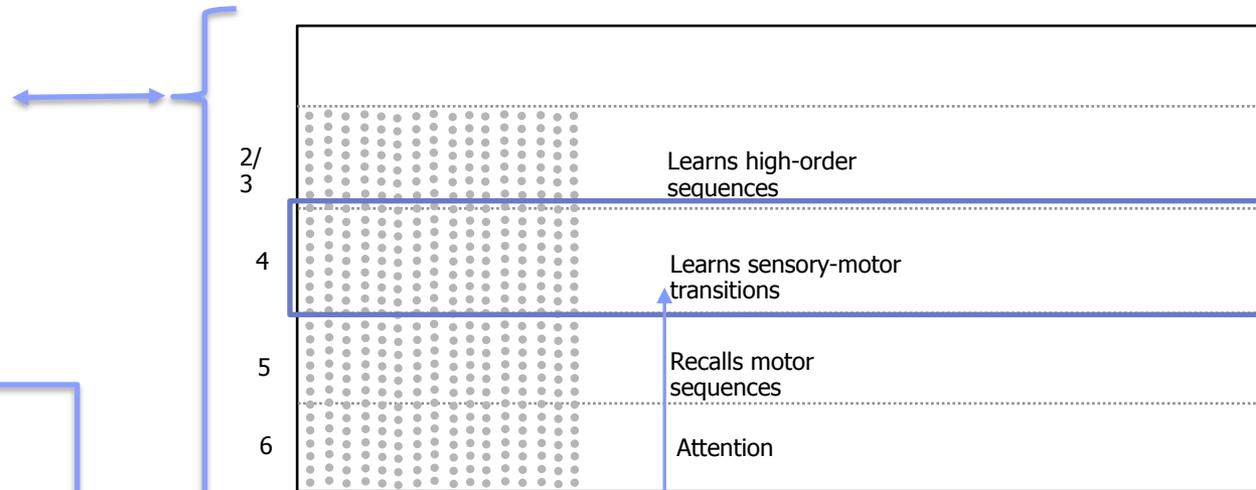
- a *logical* concept, the elements of the cortical hierarchy

REGIONS

- physical implementation of LEVELS
- horizontally distributed in biological brains

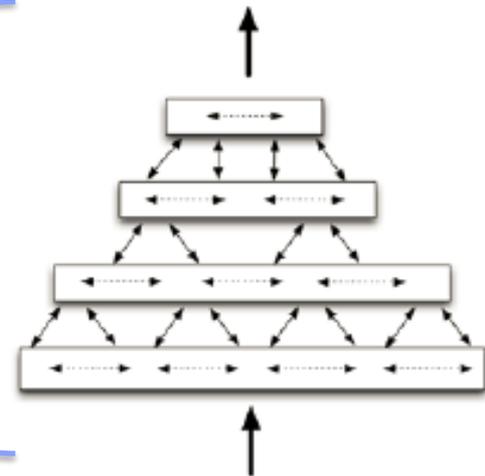
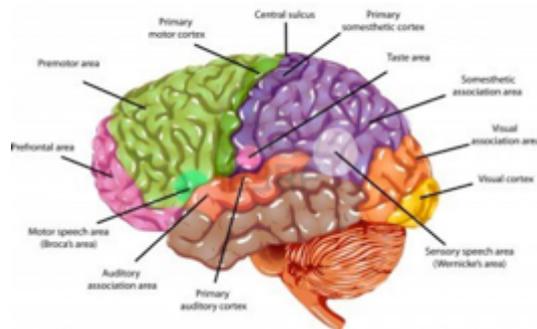
HTM building blocks

- software units, three blocks implement one LEVEL
 - spatial pooler + sequence memory + temporal pooler



Sensor/afferent data
Copy of motor commands

Regions of the Human Brain



Algorithms & HTM software

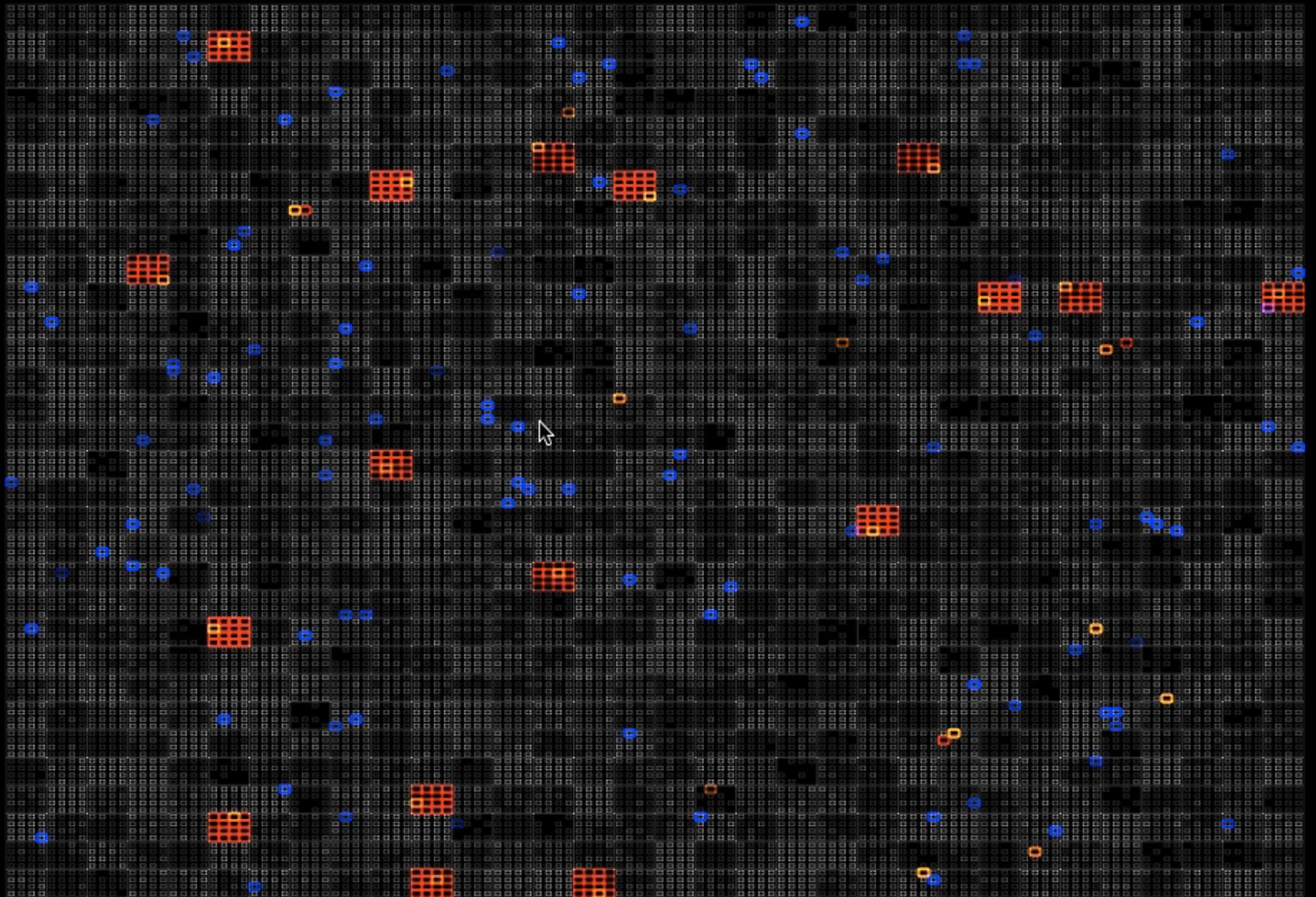
- **Refine HTM software building blocks**
- **Implementing HTM in internal software (not Nupic)**
- **demonstrate hierarchical stacks of LEVELS**
- **sensory/motor mechanism (“LAYER 4”)**

A few videos of IBM HTM software (removed here)

- predicting a chaotic time series (temporal)
- autoassociation (using MNIST numbers)
- early version of temporal pooler in operation
- Very Short demo of internal workings of HTM
- internal workings of HTM in operation 1
- internal workings of HTM in operation 2
- saccadic vision – columns find spatial patches



SP Memory usage: 6.15 MB, SWI Memory usage: 160.25 MB
 Time: 2014/08/28 17:17:57.680979, Since start: 92.803 s



Mouse Position: (x = 51, y = 700), Window size = (1024 x 669), Time/Frame = 0.016688 s, Max FPS: 59.92
 Camera Position: (0.010, -0.007, -290.000), Looking at = (0.000, 0.000, 0.000), Normal vector = (-0.004, -244.023, 133.318), rho = 290.00, theta = 270.00, phi = 360.00

Two Types of Applications

- **'Research applications' for deeper understanding of the brain**
- **Applications with potential for monetizing in the near/mid term**

Research Application: Saccadic Vision



Free examination.

1



Estimate material circumstances of the family

2



Give the ages of the people.

3



Surmise what the family had been doing before the arrival of the unexpected visitor.

4



Remember the clothes worn by the people.

5



Remember positions of people and objects in the room.

6



Estimate how long the visitor had been away from the family.

7

3 min. recordings of the same subject

Why Saccadic Vision as a 'Research application' ?

- **We love a challenge 😊**
- **Demonstrate motor/sensory integration**
- **Introduces time into vision – natural extension to videos**
- ***May* be more resource effective than e.g. deep neural network**
- **Should be less 'brittle' than conventional vision algorithms**

Several flavors of ‘Saccadic Vision’ problem

1. **Test of motor/sensory integration (LAYER 4)**
2. **Object recognition (in a static image and video)**
3. **Visual Turing test**

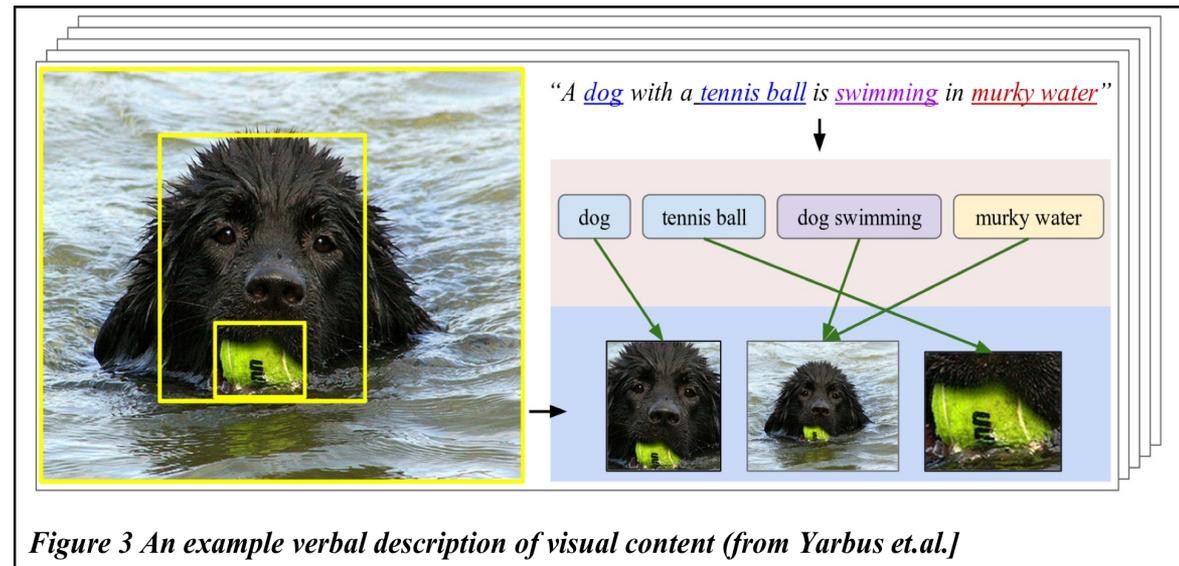


Figure 3 An example verbal description of visual content (from Yarbus et al.)

2: Object recognition in (static) images

■ Algorithm

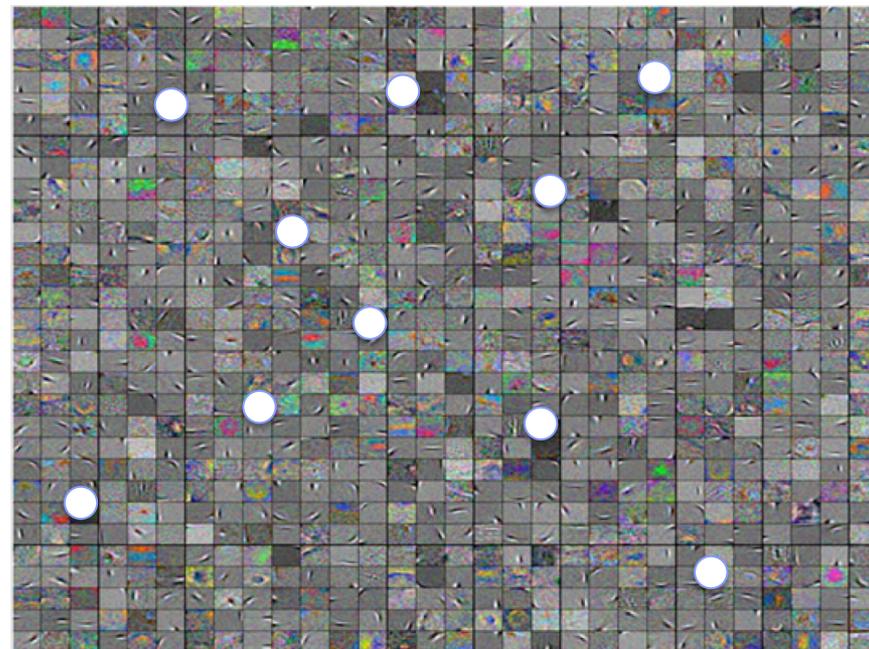
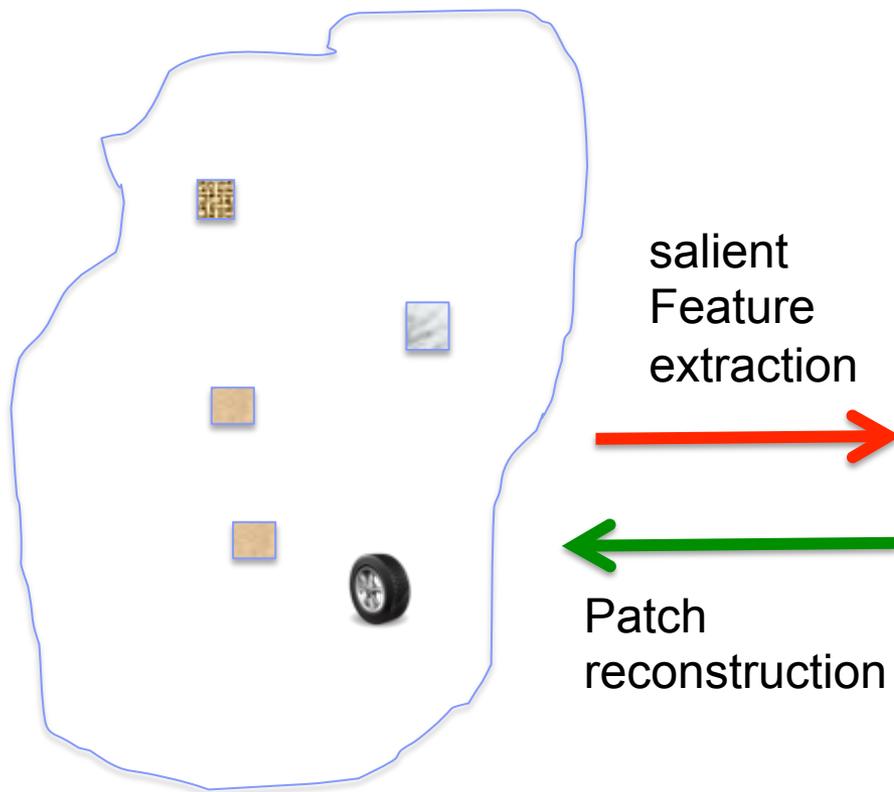
- take a large pool of images (Order of 100,000)
- select patches ($O(100)$) per image
 - try to cover the ‘interesting’ 10% of the image
- this creates a pool of $O(10 \text{ Million})$ patches
- extract a library of $O(1000)$ of ‘salient features’
 - such that all 10 Million patches can be reasonably well reconstructed by selecting and superimposing the right salient features

■ Work supported by DARPA MTO

Image space



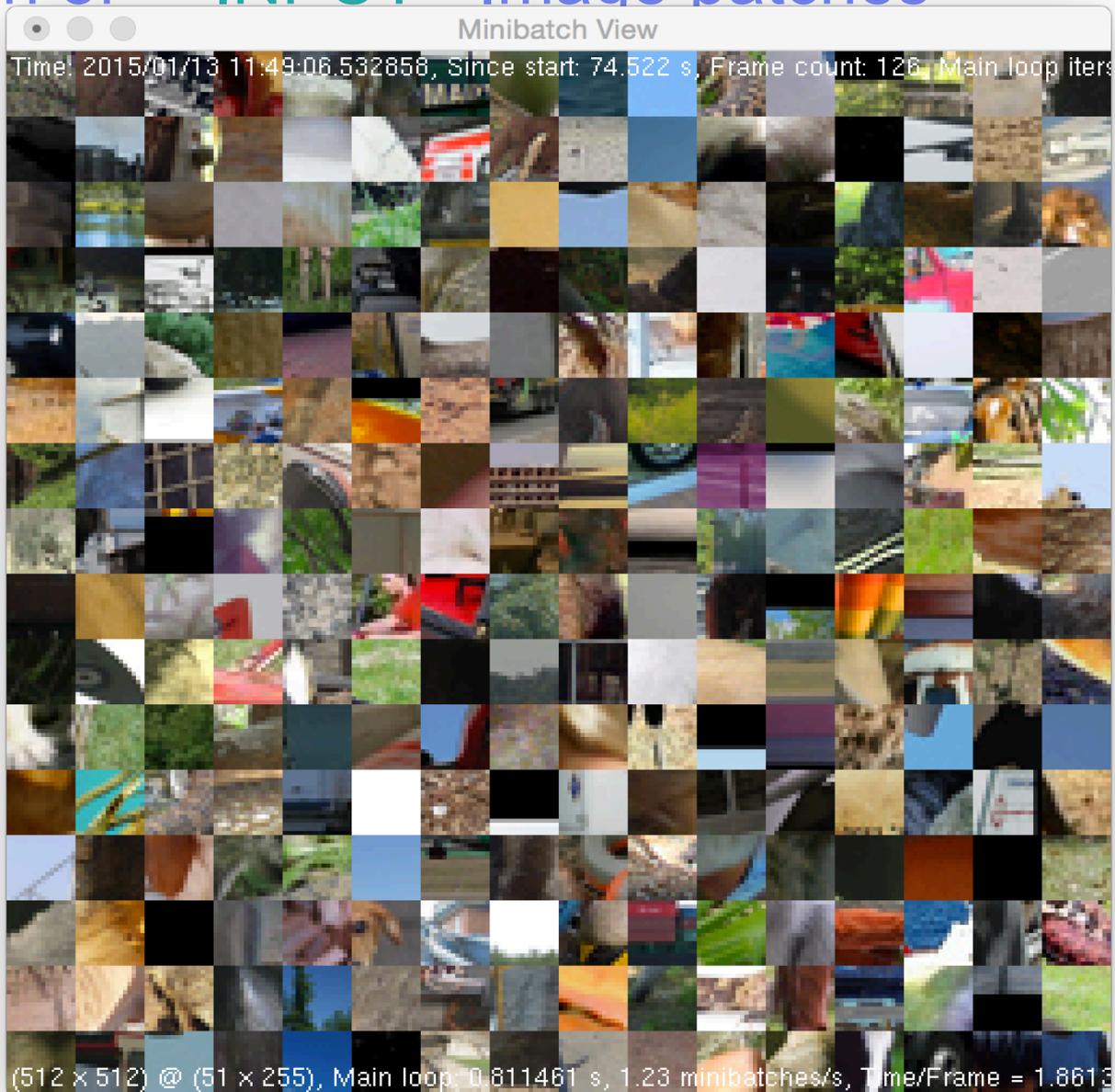
f-space



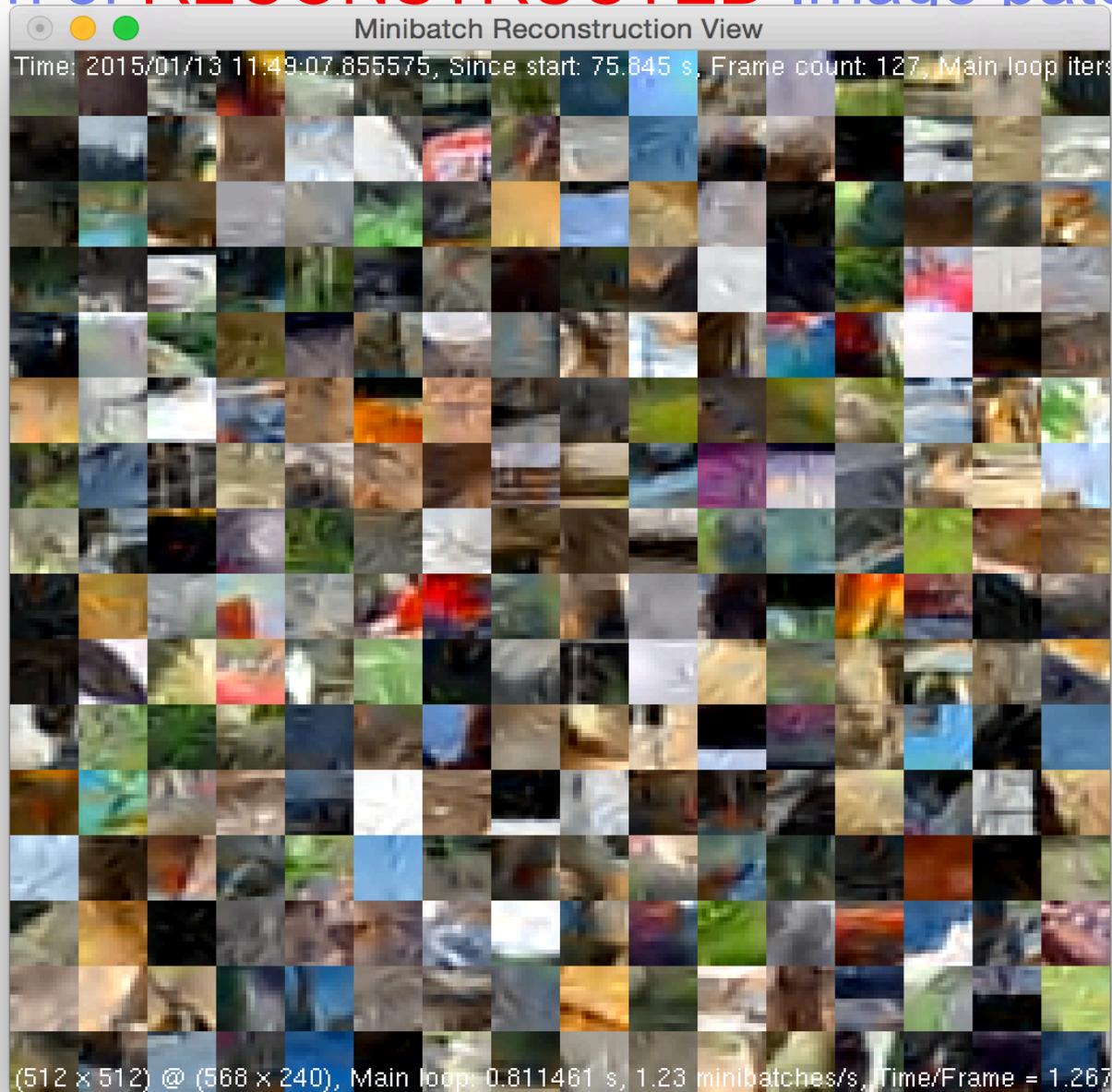
Pool of 10 million small **PATCHES**
selected from 100,000 images

Pool of ~ 1000 *salient* **FEATURES**
can represent millions of patches

Collection of INPUT Image patches

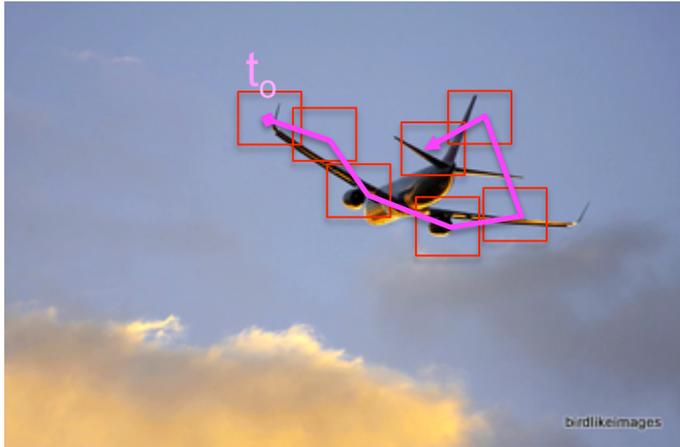


Collection of **RECONSTRUCTED** Image patches

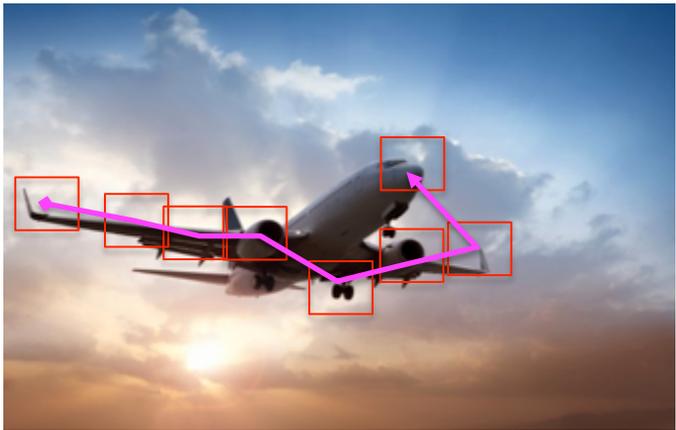


Object Recognition: combine path and patch info

known image 1



known image 2



time = eye motion steps →

t_0		t_1		t_2	
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
0	0	0	0	0	0
0	1	0	1	0	1
1	0	1	0	1	0

↑ patch location SDR
 ↑ patch description SDR = superposition of features vector

} combined into one 'spatial' SDR

- The combination of patch location SDR and patch description SDR is stored in HTM sequence memory as the representation of an object.
- Multiple object instances go into the temporal pooler

HTM Hardware Projects at CLC

- **ESCAPE**
- **3D waferstack**
- **Phase-Change Memory based Neural Network**

Hardware 'Philosophy'

- It's too early to cast learning algorithms/models into specialized silicon (or post silicon devices)
- Conventional computing languages still best way to test ideas and algorithms
- HTM learns mostly by changing the network *topology* – difficult to do in hardware
- → best platform for HTM development are **optimized** highly parallel von Neumann machines



Mapping HTM Algorithms to Hardware

HTM algorithms requires hardware matched to the algorithm's needs

1. High Connectivity (more expensive than the computational requirements)
2. Local Memory and Parameter Storage
3. Simple, Low Precision Computation
4. Configurable / Adaptable
5. Sparse activity

Dr. Dan Hammerstrom, NICE 2014

Conventional processors are a poor match to cortical algorithms:

- Constrained: processor/memory partition, limited parallelism
- Excessive: high precision, tiered caches, complex instruction sets, pipelines, etc.



Conventional Solutions

Custom architectures can address HTM's requirements:

- High-risk exotic devices unnecessary
- Utilize conventional CMOS fabrication optimized for HTM architecture/computational model
- Can benefit from latest advances in CMOS.



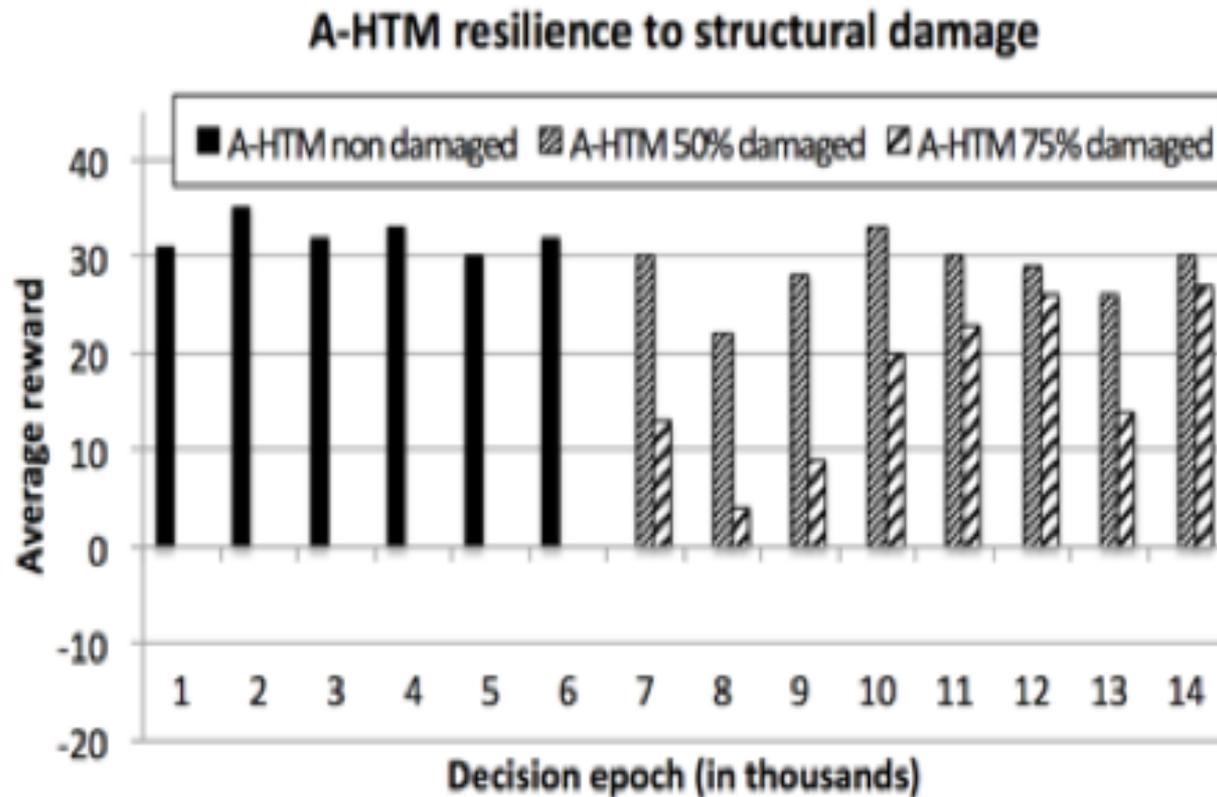
Hierarchical Temporal Memory

Specialized HTM processor

Key Insight

- **The extremely high connectivity of wafer-scale and 3D stacking is a great match for cortical simulations**
 - Performance is derived from high memory bandwidth feeding a large number of fairly simple processors
 - Very high communications performance between processors (message passing model)
- **The resilience of HTM algorithms (shown via simulations) makes wafer-scale yield problems irrelevant**

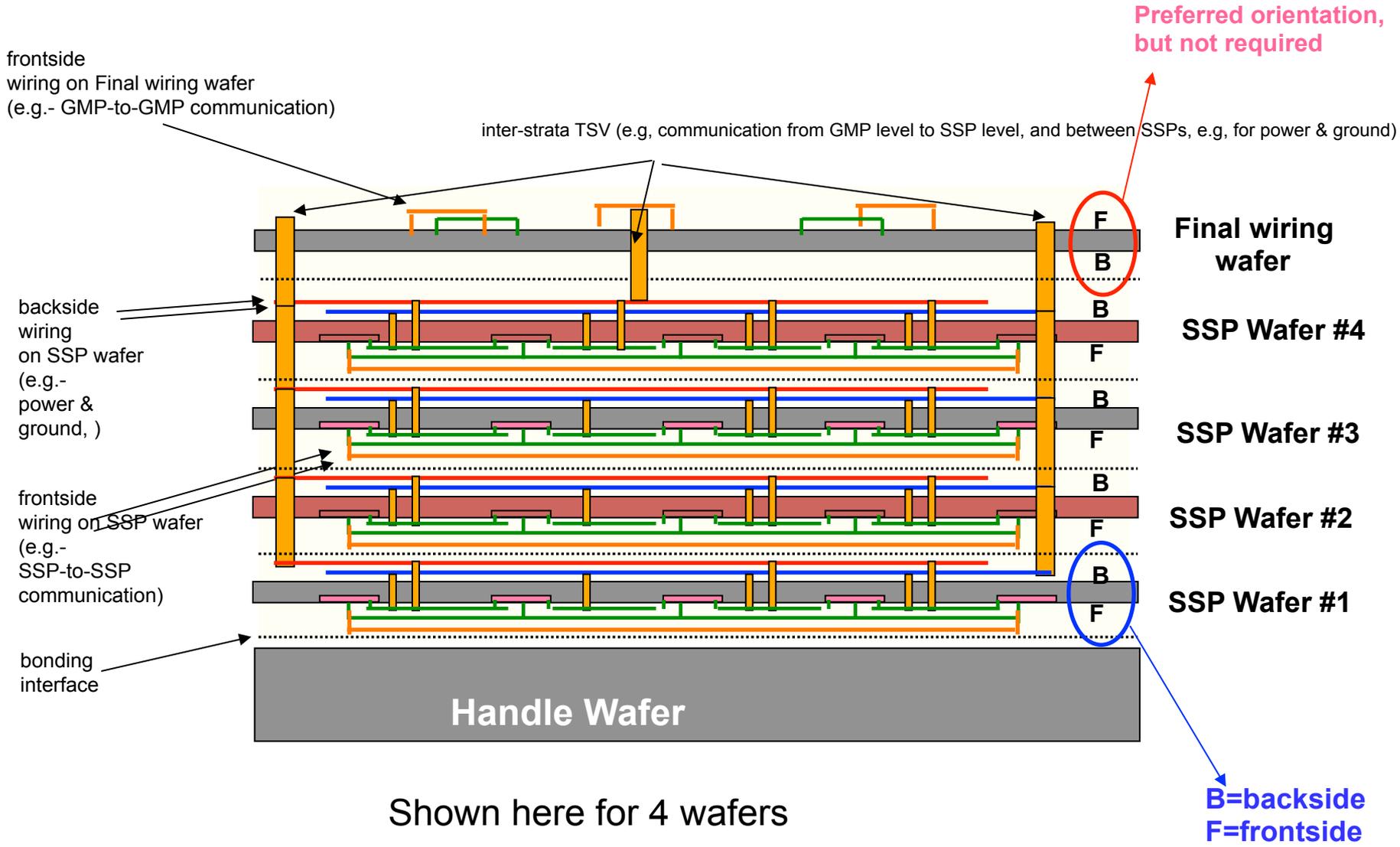
Effects of destroying 50% or 75% of columns in a-HTM (Janusz Marecki, IBM Yorktown)



Wafer Stack Architecture Description

- **N pairs of 300mm vertically stacked Si-wafers**
- **Each pair contains**
 - ‘ANT’ wafer: array of small processors + e-DRAM (cache)
 - DRAM wafer – mostly stores tables of synapse state
 - 0.5 TB DRAM per wafer
- **vertical communication via densely spaced vias**
- **horizontal communication on ANT wafers**
 - communications is not the bottleneck!
- **power dissipation will set the limit for N (a few kW)**
- **technology co-development with other ‘Gen3’ uses**

3D Wafer Scale Integration of wafers



ESCAPE

- **Large FPGA based system under development**
 - 1000 node, 1TB RAM ESCAPE system,
- **Dual purpose**
 - platform for detailed architecture design of 3D stack
 - will greatly speed up HTM simulations in 2016
 - needed for saccadic vision work



Experimental Demonstration of a Neural Network (165,000 synapses) built from Phase Change Devices

- Presentation at IEDM 2014 (Geoffrey Burr et.al.)
- First demo of a rather large NN where all synapses are pairs of Phase Change Memory Devices (not Silicon)
 - Because of strong asymmetry of PCM, each synapse is implemented as a voltage divider to allow changes up and down
- Implemented a backpropagation net and tested on MNIST
 - 82.9% accuracy
- Network is very resilient against random effects
 - device variability – yield – stochastic effects
- But highly sensitive to ‘gradient effects’ which steer all conductivities in one direction

Thank you!
Questions?