

iLab C++ Neuromorphic Vision Toolkit Overview



- Components:
 - Basic image processing and vision
 - Attention-related neural components
 - Object recognition-related neural components
 - Scene gist/layout-related neural components
 - Basic knowledge base / ontology
 - Hardware interfacing
 - Beowulf message passing
 - Applications
- Implementation:
 - C++, somewhat Linux-specific
 - Additional perl/matlab/shell scripts for batch processing
 - Uniprocessor as well as Beowulf

Basic functionality



Find the most interesting location in the image (next slide)



The model's prediction



Here is what our model of bottom-up, saliency-based attention found
(next slide)

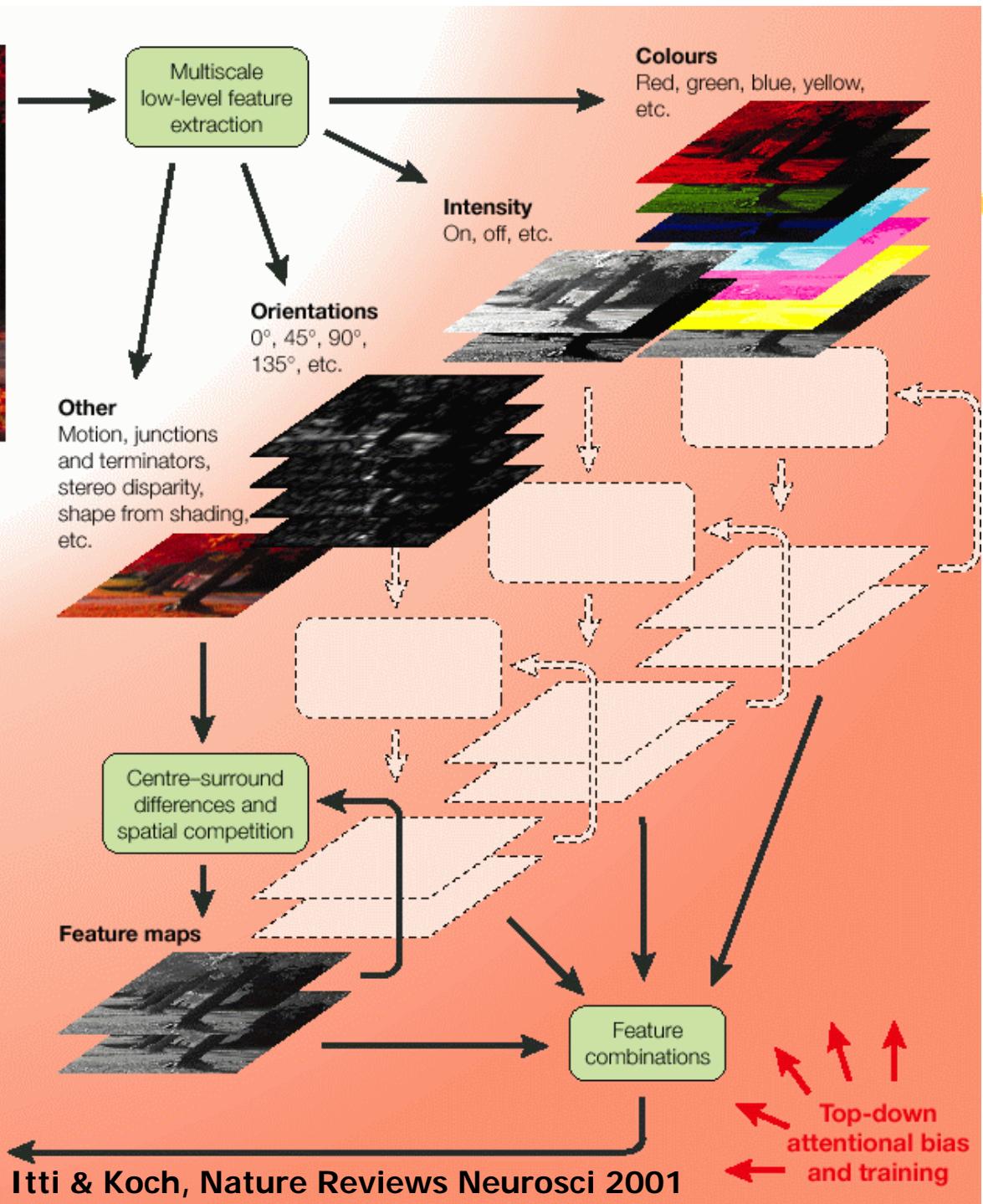


The basic architecture



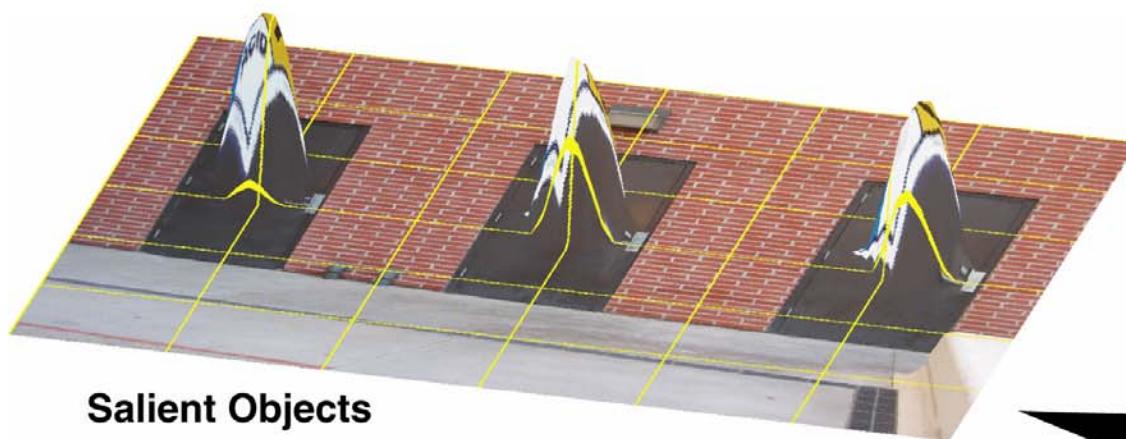
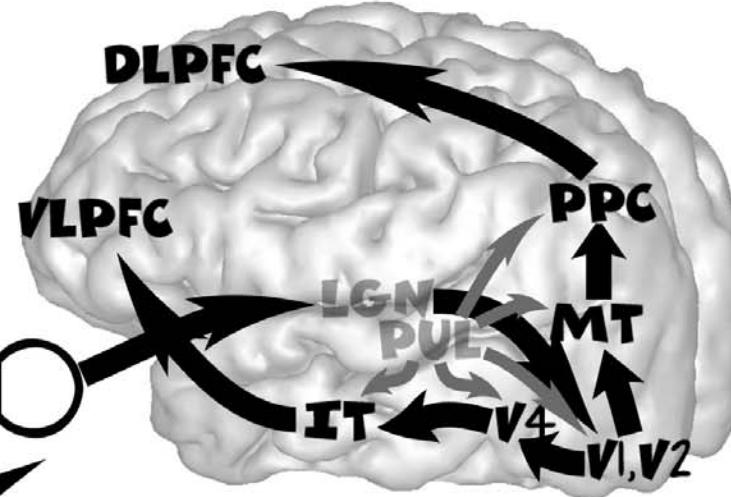
- The diagram on the next slide is an overview of this computational neuroscience model
- Suggested readings: see <http://iLab.usc.edu/publications/>
 - Start with [Itti & Koch, Nature Reviews Neuroscience, 2001](#), for an overview
 - Then see [Itti, Koch and Niebur, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998](#), for the core algorithm
 - Then see [Itti & Koch, Vision Research, 2000](#) and [Itti & Koch, Journal of Electronic Imaging, 2001](#), for more advanced competition for salience
 - See papers by [Vidhya Navalpakkam](#) for more on scene understanding
 - See papers by [Nathan Mundhenk](#) for more on contour integration
 - See papers by [Nitin Dhavale](#) for more on eye movements
 - See papers by [Chris Ackerman](#) for more on gist
 - Etc...

Input image

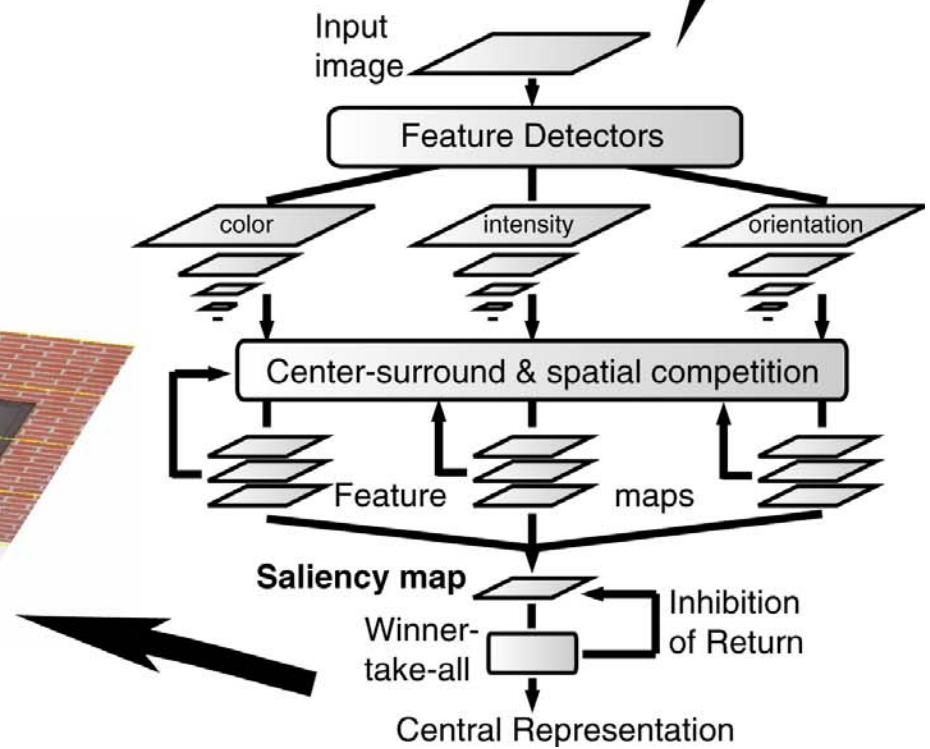


Architecture

Visual Scene



Salient Objects



iLab - University of Southern California - E3 2002 Report - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address <http://ilab.usc.edu/events/2002-e3/> Windows

2002 E3 Report (Electronics Entertainment Expo)

From our exclusive iLab correspondent **April Tsui**, Master's student at the **Art Center College of Design**.

The view from E3

 10M MPEG  4.2M MPEG

The **2002 E3** came and went, but here is a full report on the future trends set by the gaming industry. The gaming community, with its various salient interfaces and its much vaunted multi-million dollar booths that exhibited the current reigning consoles were captured in our exclusive footage. Presented through iLab, complete with predictions from our **visual attention model** (captured by yellow and green circles) and saliency maps (shown to the right of each video clip).

Purpose

 1.4M MPEG  1.1M MPEG

The purpose of this footage is to test the visual saliency algorithm on video games. iLab is investigating possible applications of the **saliency software** to the development of more sophisticated opponents. These opponents will use intelligent agents that would employ the iLab algorithms for visual perception. Testing is also done on the **Beobot** robotics platform to be released this year, with external components designed by yours truly.

Line-up

 Zelda 10M MPEG  S. Monkey Ball2 9.4M MPEG

At left are MPEG movie clips showing the following line-up: Legend of Zelda, Super Mario Sunshine, Hawk's Pro Skater 4, Super Monkey Ball 2, and Eternal Darkness. The visual saliency software analyzes each frame of the scene: color, intensity, motion, and other factors. In this case, the software is not very attuned to even more specific characteristics (a specific color, for example), it is not the case here, so as to give each game the same evaluation. The saliency software also considers the 'inhibition of return,' which means that once a specific location has been visited, it will not return to the location before 1 to 2 seconds. Each movie clip is in real-time, and the software updates at 30 frames per second.

See here (URL at top) for examples of Processing movies

Results

Internet

Start Microsoft PowerPoint - iNVT-intro iLab - University of Souther...

3:09 PM

2 Visual Attention - Movies - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/bu/movie/index.html

Visual Attention: Movies

In the first short MPEG video, we show both the original input image and the corresponding saliency map in dynamical evolution. The saliency map begins with charging up from the input. Then the most salient location is found and transiently inhibited. The yellow circle on top of the original image represents the current focus of attention.

The first image shows simple objects in noise; objects are selected by order of decreasing saliency (here luminance contrast). After all objects have been attended, less salient locations in the background noise are attended. The previously attended and inhibited locations progressively charge-up again and may be again attended.

Then, in the examples of pop-out target detection, the target is often detected by the model independently of the number of distractors. In the example of conjunction search, the target does not compete with other features in the scene (here orientation features are competing, the target does not pop-out). In the example of pop-in target detection, the target is detected by the model, which performs a serial search).

See here for examples of Processing movies

The Batman™ poster demonstrates how, generally, the attentional trajectories generated by the model seem to agree with our reading of the image. Note how a good coverage of the image is obtained (attention does not only go to the 3 faces, but looks around about everywhere), with frequent checking on the most salient objects. Good performance was also obtained with a database of traffic sign images from Daimler Benz, Inc. Since road signs have been designed to be salient (the car is also salient here), they are found before the system looks around in the trees.

We conclude this demonstration with an example of robustness to noise.

In this second video, the spatial competition among conspicuous locations within each feature map is demonstrated. Details about the implementation of such spatial competition can be found in our [2001 Journal of Electronic Imaging paper](#).

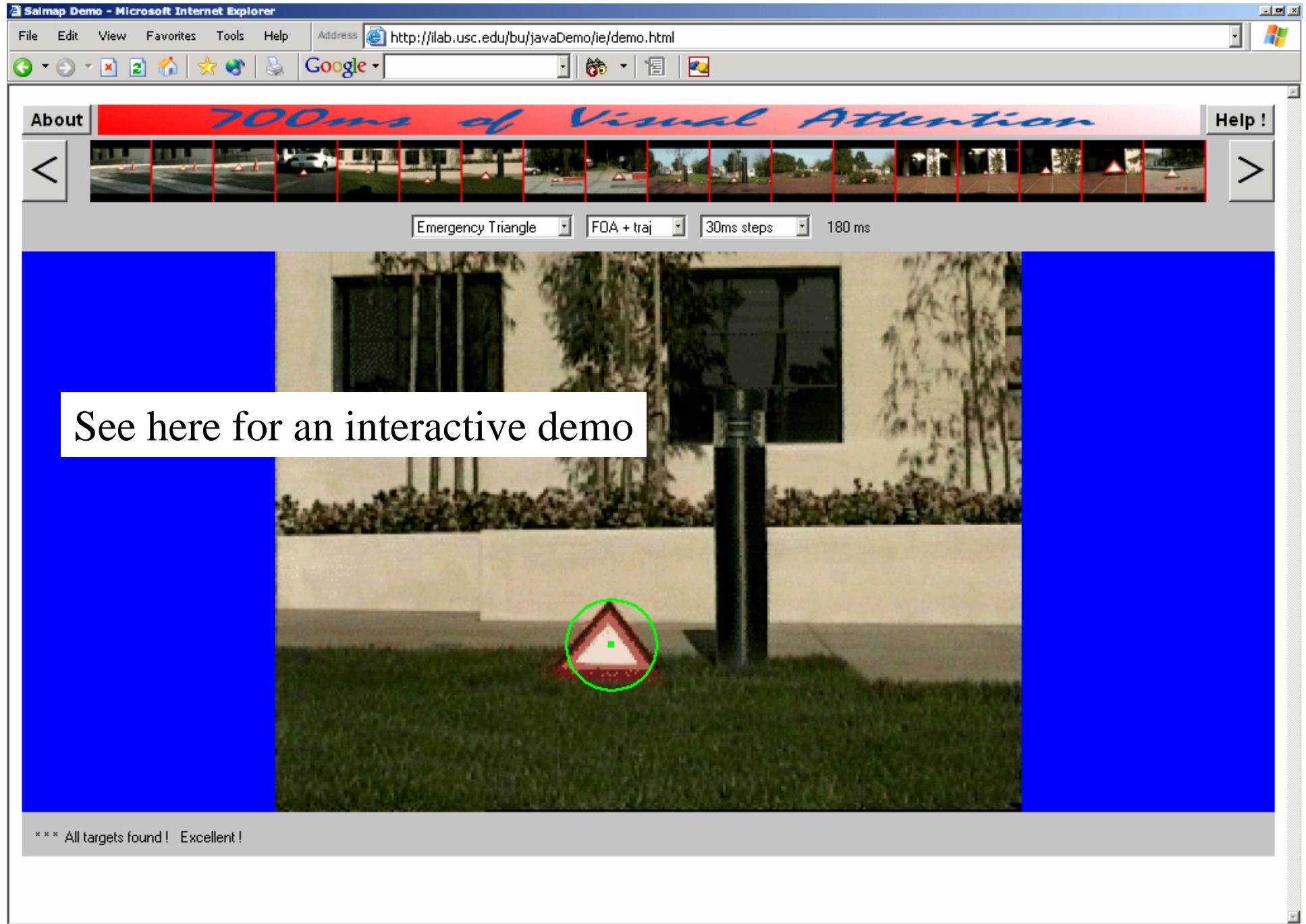
(2.3 Mb MPEG-1 file)

(3.8 Mb MPEG-1 file)

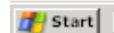
Done

Internet

Start 0312_INVTintro Microsoft PowerPoint - iNVT-in... Visual Attention - Movies - ... 3:14 PM



Applet started

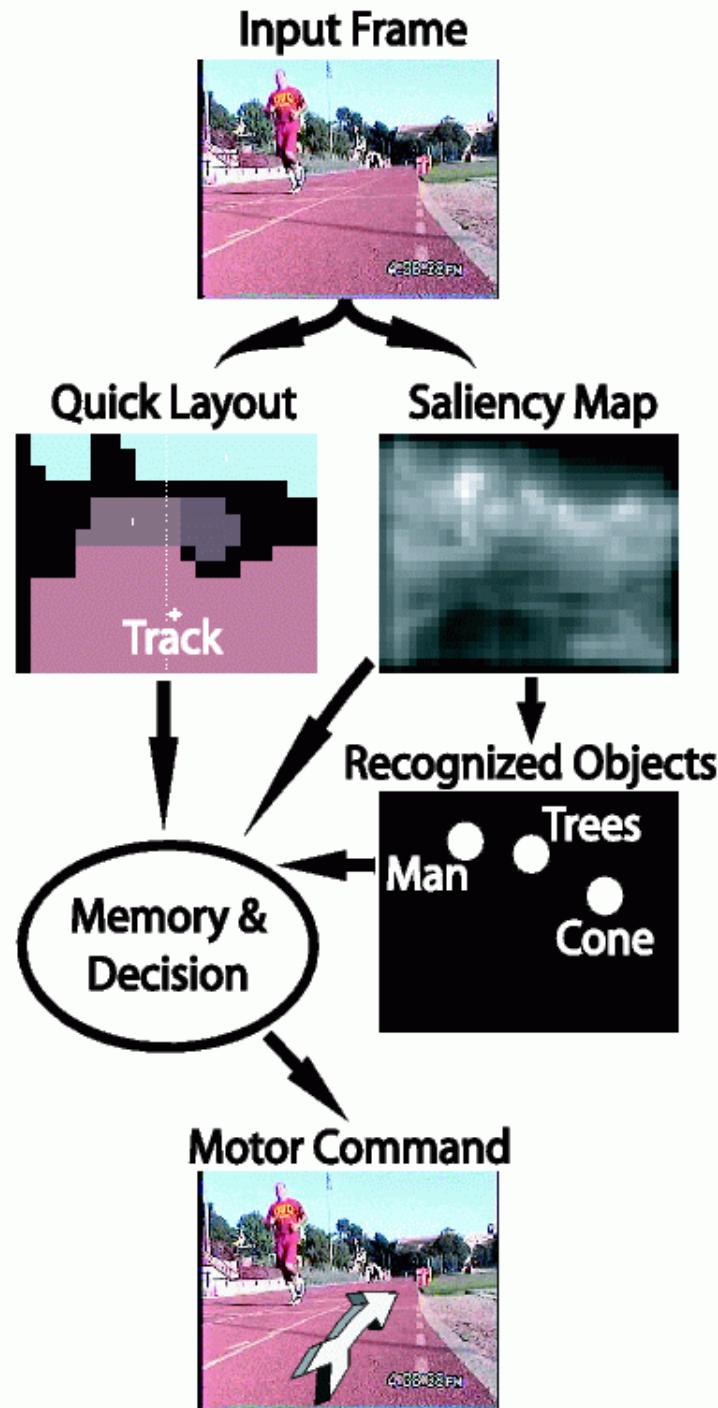


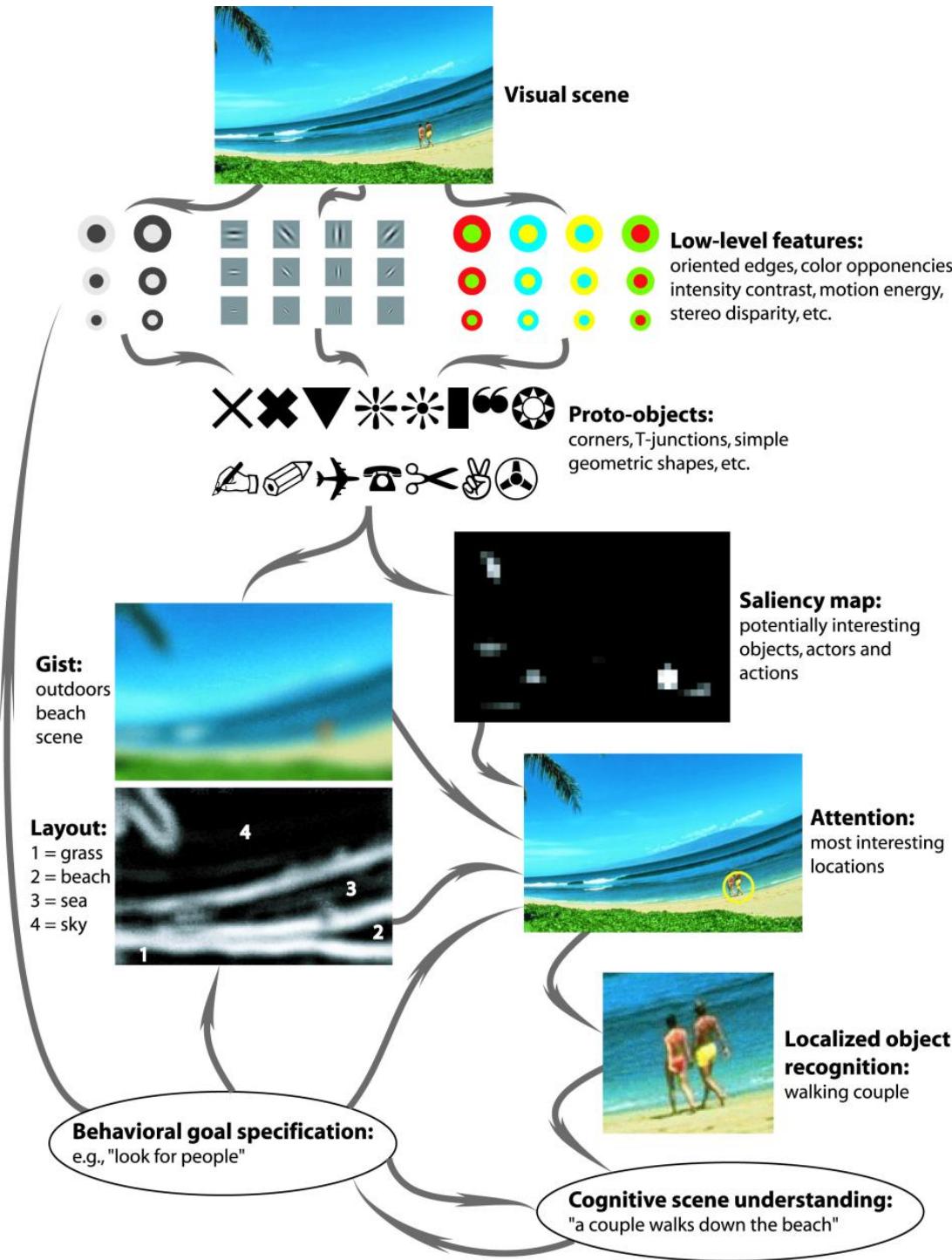
Internet



3:16 PM

The big
picture...

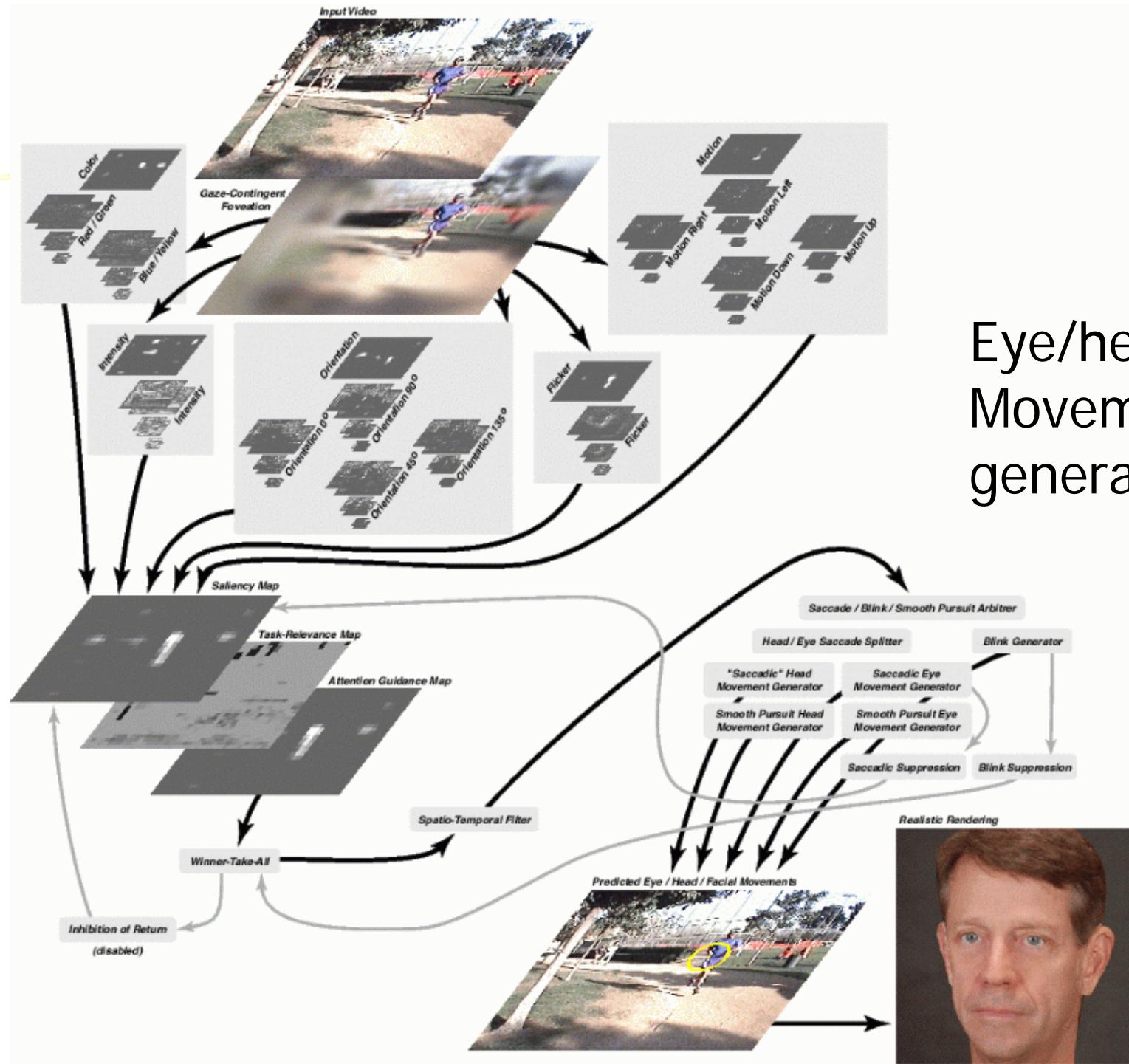




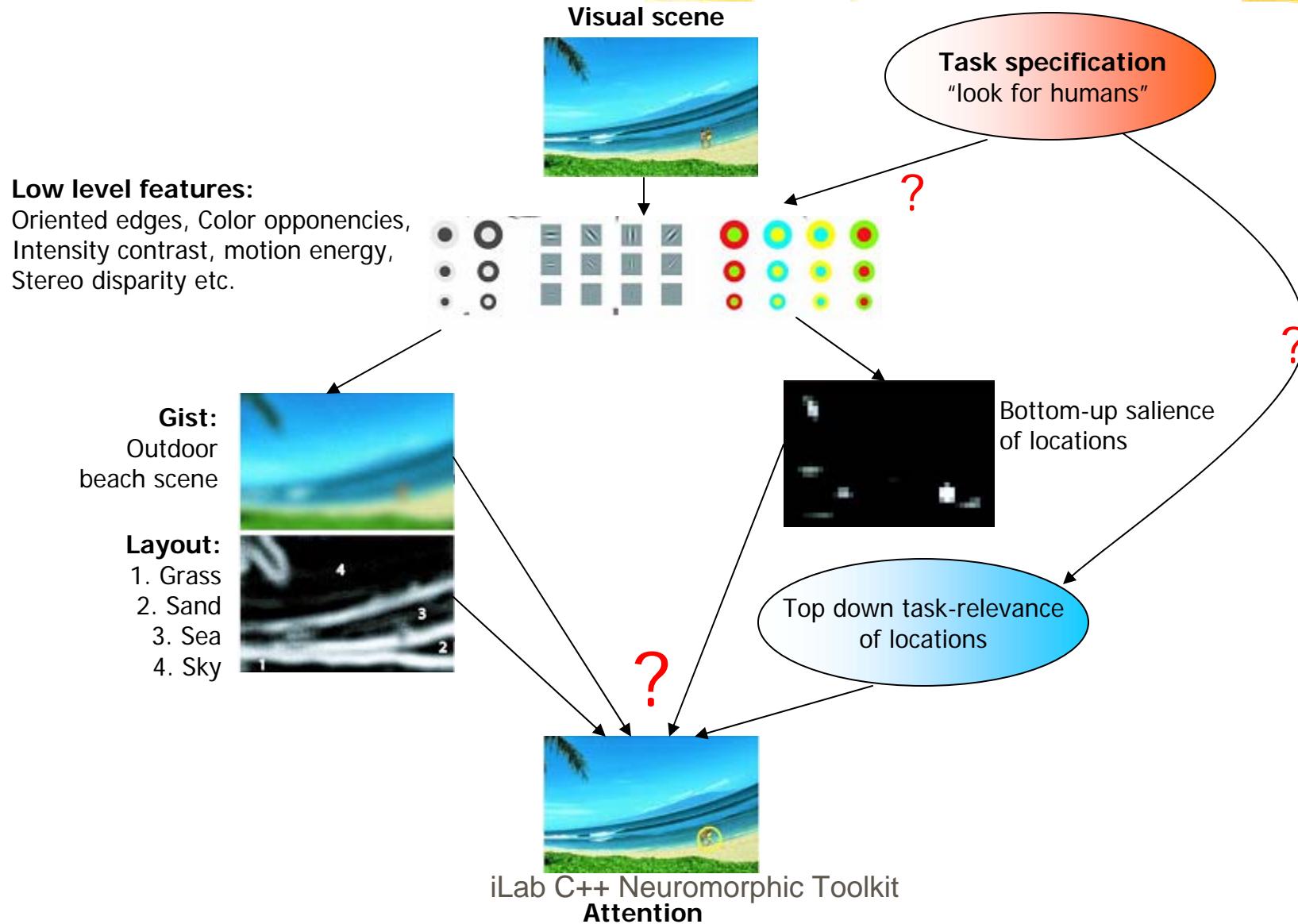
<http://iLab.usc.edu/bu/>

oolkit

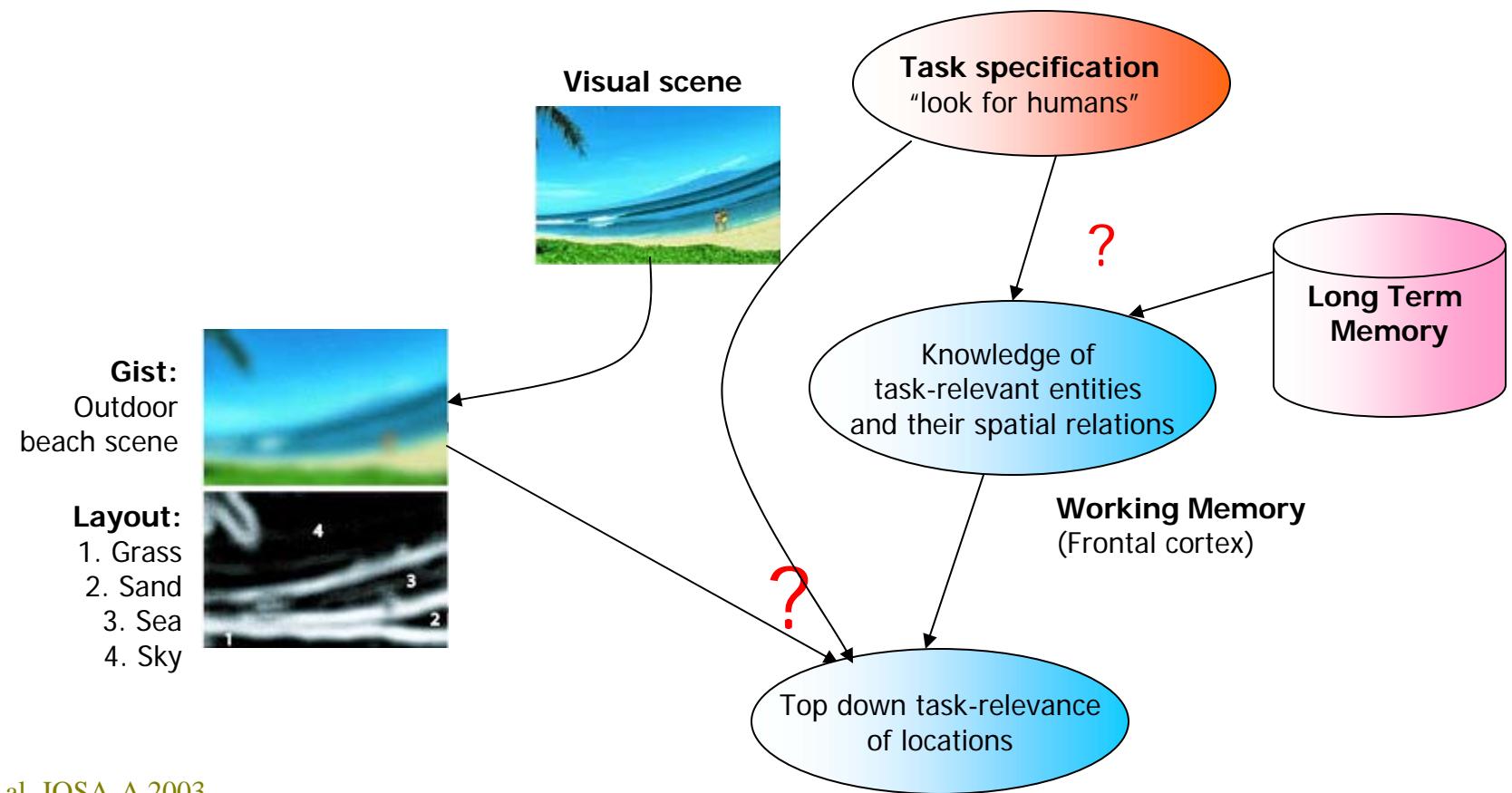
Eye/head Movement generation

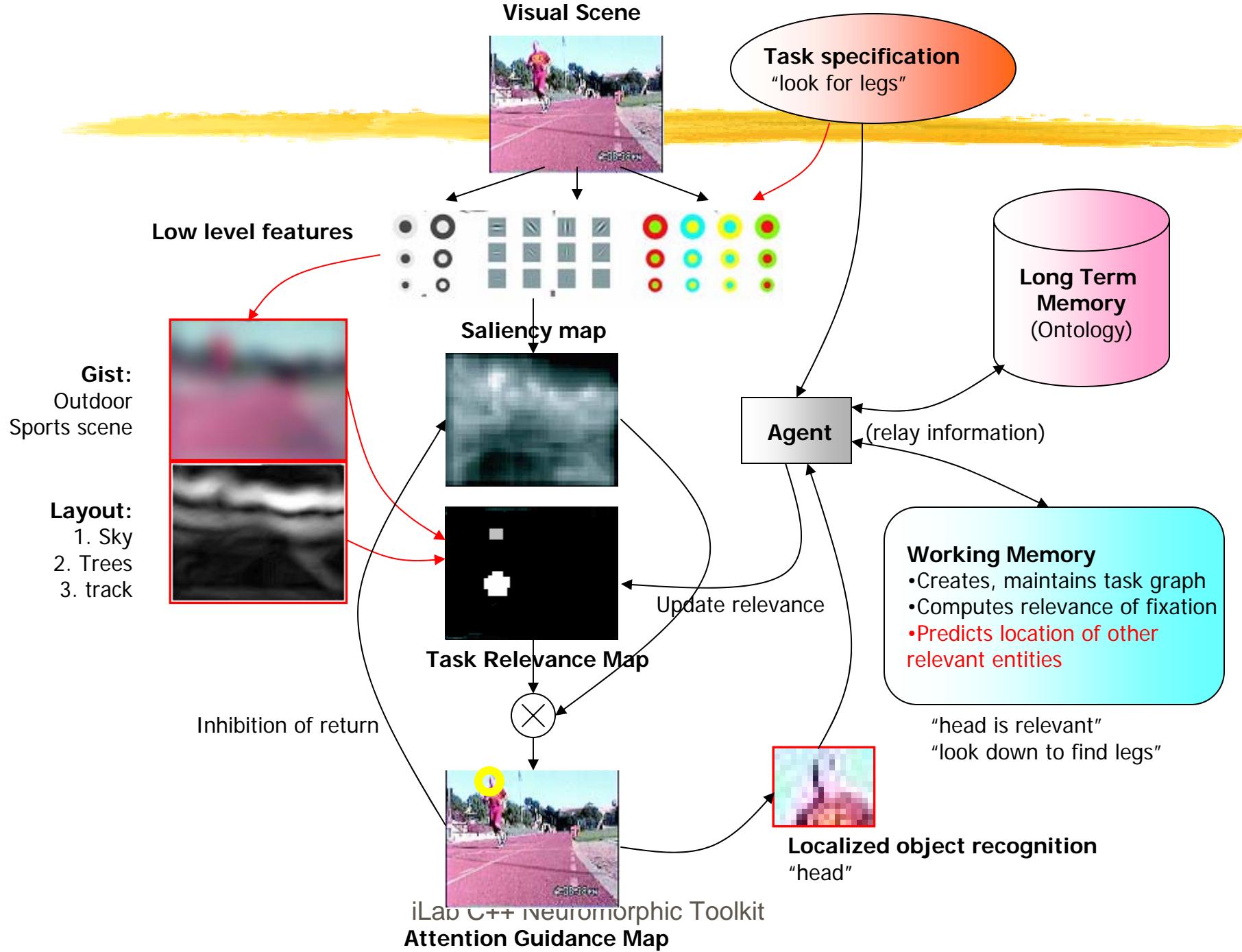


How does task influence attention?

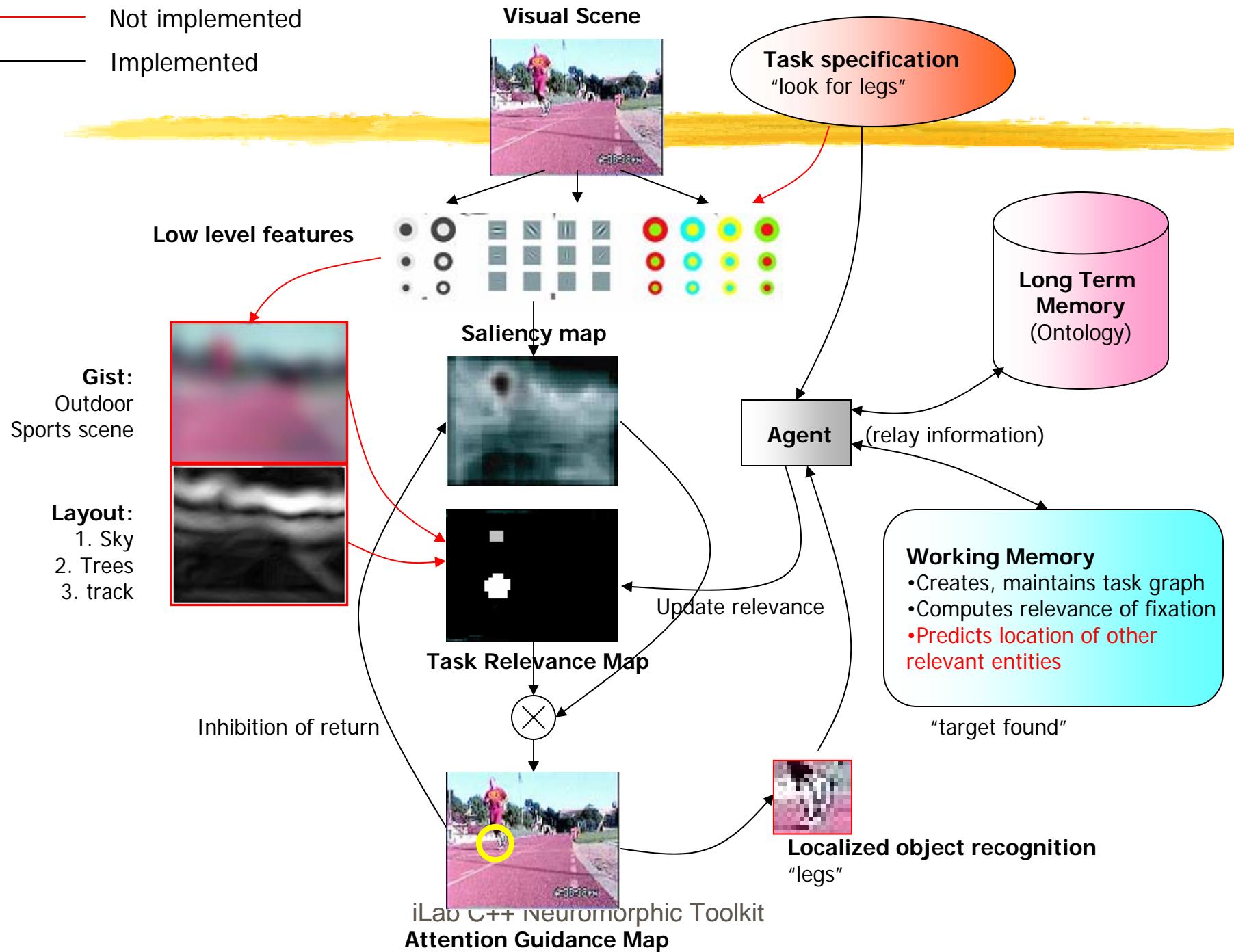


Towards modeling the influence of task on relevance

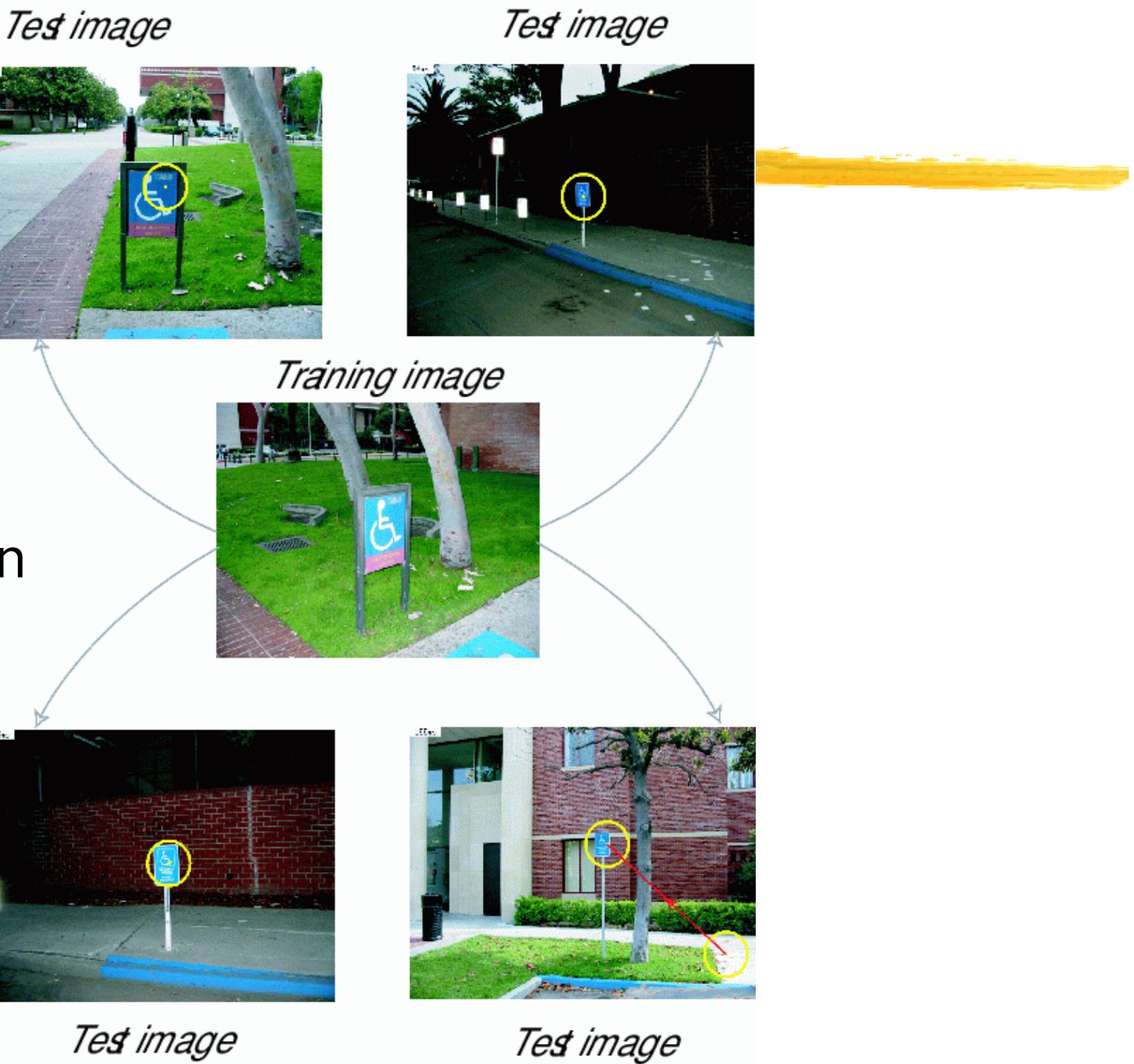




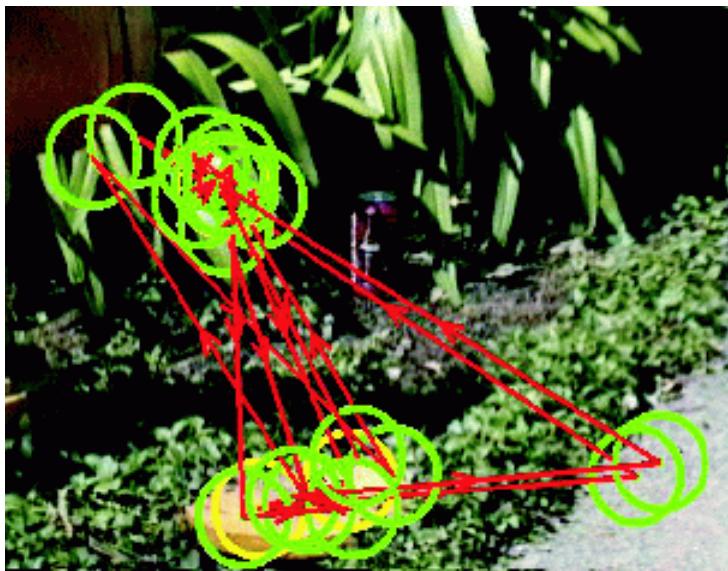
Not implemented
 Implemented



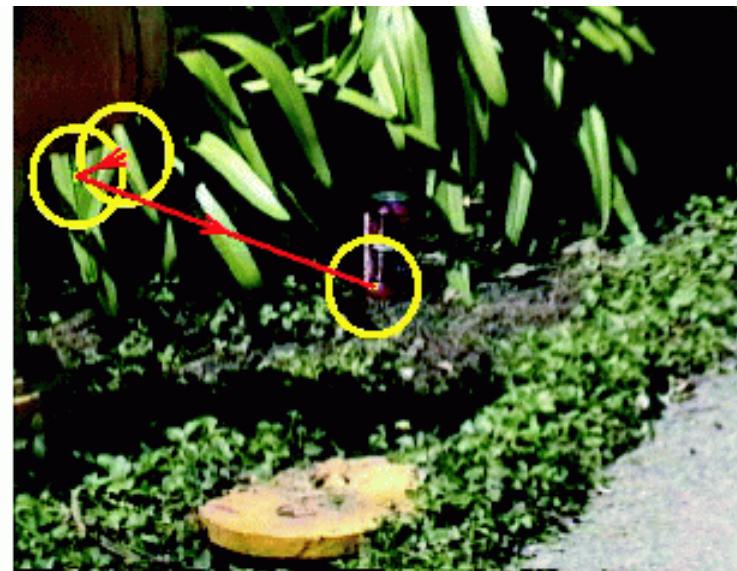
Object recognition



Top-down biasing to guide attention towards Known objects

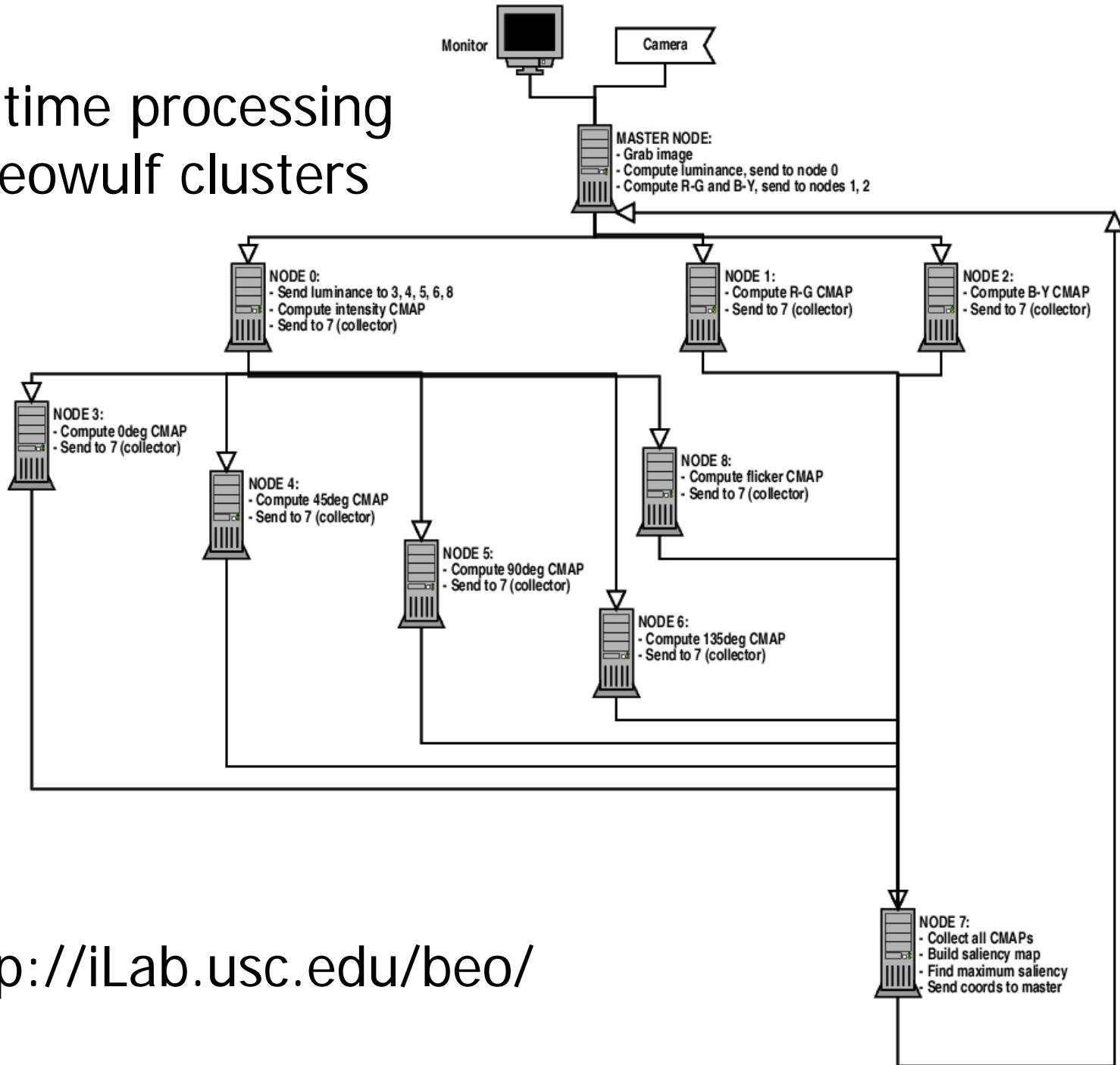


Unbiased

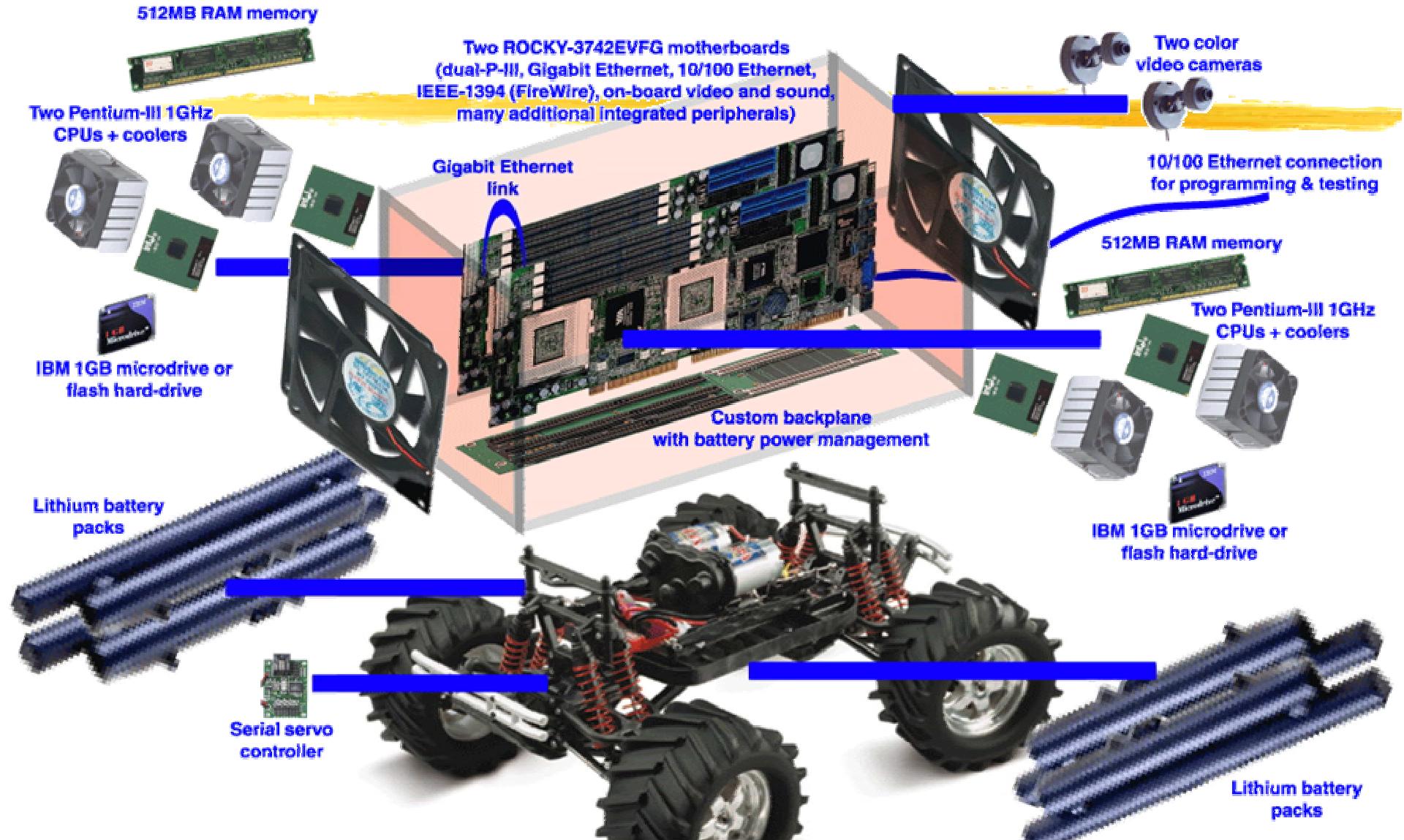


Biased for coke cans

Real-time processing On Beowulf clusters



<http://iLab.usc.edu/beo/>



Beowulf + robot =
"Beobot"

Traxxas E-Maxx
4WD chassis
(stiffened suspension, lowered gear ratio, foam-filled tires)

<http://iLab.usc.edu/beobots/>



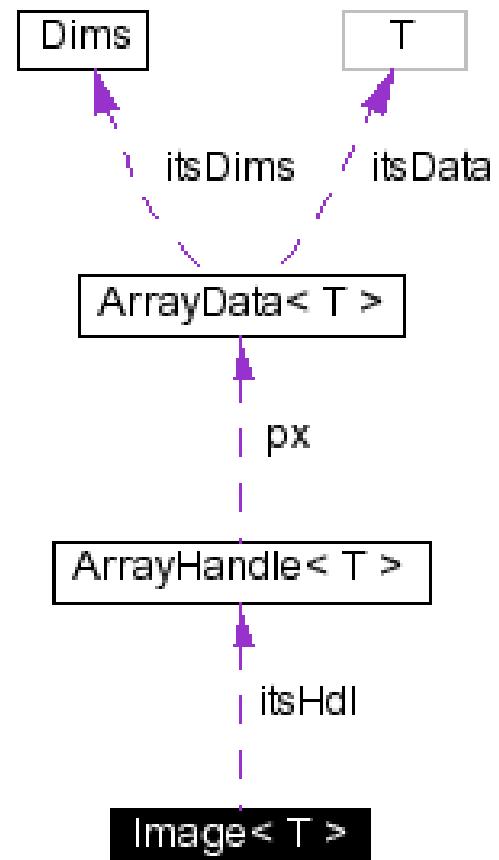
iLab C++ Neuromorphic Vision Toolkit Overview



- Components:
 - Basic image processing and vision
 - Attention-related neural components
 - Object recognition-related neural components
 - Scene gist/layout-related neural components
 - Basic knowledge base / ontology
 - Hardware interfacing
 - Beowulf message passing
 - Applications
- Implementation:
 - C++, somewhat Linux-specific
 - Additional perl/matlab/shell scripts for batch processing
 - Uniprocessor as well as Beowulf

Root: Image class

- Template class
 - e.g., `Image<byte>`, `Image<PixelRGB<float>>`, `Image<Neuron>`
- Implemented using copy-on-write/ref-counting
 - Makes copying a light operation
- Many associated methods
 - Shape ops
 - Color ops
 - Mono only
 - Math ops
 - I/O
 - Filter ops
 - Transforms



C++ Templates

- **The old way:** ByteImage, FloatImage, ColorImage, etc. yields lots of duplicated code that achieves essentially the same operations.
- **The C++ way:** write your algorithm only once, and make it operate on an unknown data type T. The compiler will then generate machine code corresponding to your algorithm and various data types for T, such as, T=byte, T=float, T=MyClass, etc

```
template <class T> class Image {  
public:  
    Image();  
    T getPixelValue(const int x, const int y) const;  
    void setPixelValue(const T& value, const int x, const int y);  
private:  
    T* data;  
};  
  
int main(const int argc, const char **argv) {  
    Image<float> myImage; myImage.setPixelValue(1.23F, 10, 10);  
    return 0;  
}
```

See Image.H

Operator overloads

- C++ allows you to define operators such as +, -, *, etc for your various classes.
- Example:

See Pixels.H, Image.H

```
Image<byte> img1, img2;
```

```
img1 += 3; // calls Image<T>::operator+=(const T& value)
```

```
img1 = img1*2 + img2/3; // calls operator*(const T& value),  
// operator/(const T& value),  
// and operator+(const Image<T>& im)
```

Automatic type promotions

- Using type traits to determine at compile time whether the result of an arithmetic operation will fit in the same type as the operands.
- Extends the canonical C++ promotions to non-canonical types.
- Examples:

`Image<byte> im;`

See Promotions.H,
Pixels.H, Image.H

`im + im` is an `Image<int>`

`im * 2.0F` is an `Image<float>`

`im * 2.0` is an `Image<double>`

Automatic type demotion with clamping



- Assignment from a strong type into a weak type will ensure that no overflow occurs.
- Example:

```
Image<byte> im1, im2;  Image<float> im3;
```

```
im1 = im3;      // will clamp values of im3 to 0..255 range and convert
```

```
im2 = im1 * 2.0; // will create an Image<double> containing the  
                  // result of im1 * 2.0, then clamp this image to  
                  // 0..255 pixel range, then assign to im2.
```

Copy-on-write / ref counting



- The standard way:

Image object contains an array of pixels:

Image<T> object

int width, height;

T* data;



Problem: copy is expensive, need to copy the whole array.

Copy-on-write / ref counting



In particular, this makes it very expensive to return `Image` objects from functions, hence essentially forbidding the natural syntax:

```
Image<float> source;
```

```
Image<float> result = filter(source);      With a function:
```

```
Image<float> filter(const Image<float>& source) {  
    Image<float> res;  
    // fill-up pixel values of res, processing values from source  
    return res;  
}
```

Indeed what happens here is:

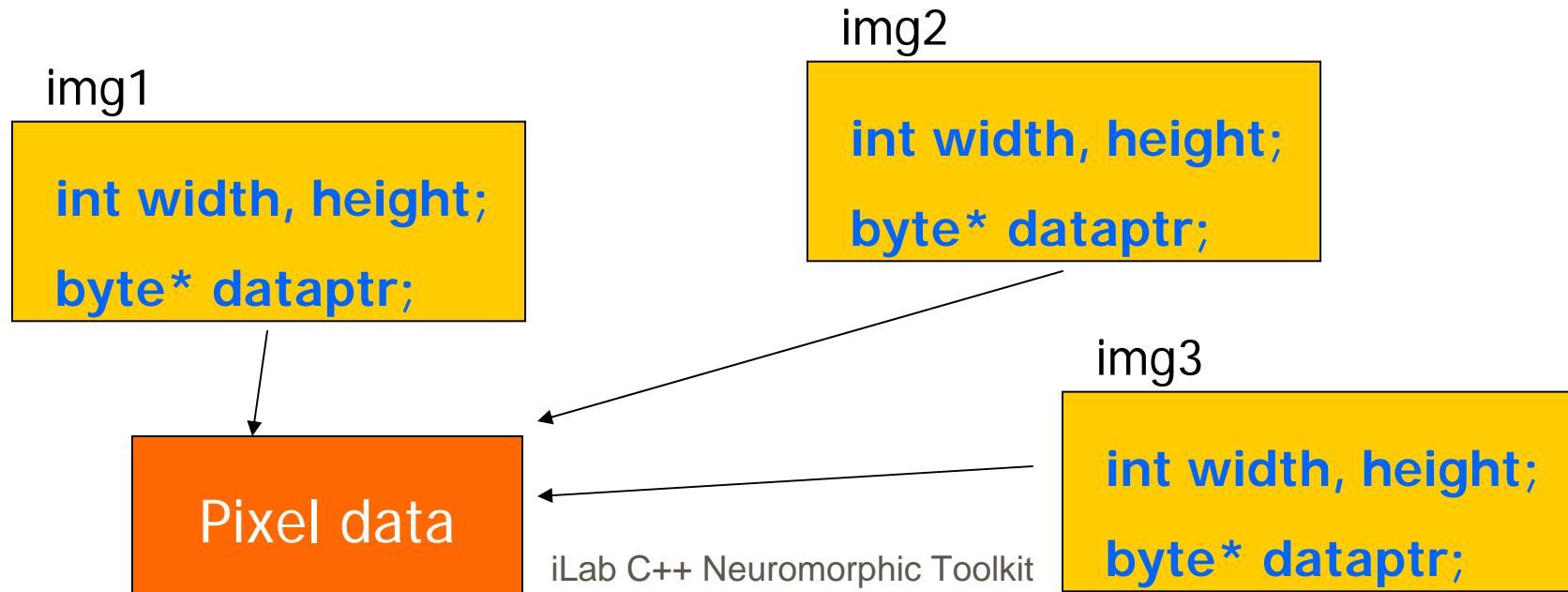
- 1) Inside `filter()`, allocate a new image `res` to hold the result
- 2) In the '`return`' statement, copy that local image to some temporary
- 3) In the '`=`' statement, copy that temporary to `Image 'result'`

Copy-on-write / ref counting

See `ArrayData.H`, `Image.H`

- The smart way: only keep a pointer to the actual pixel data in each `Image` object. When making copies of the `Image` object, keep track of how many are pointing to the same pixel data. When the last `Image` object is destroyed, free the pixel data. If the user attempts to modify the contents of one of the images that point to the same data, first make a copy of the data.

```
Image<byte> img1, img2, img3;    img2 = img1; img3 = img1;
```



Free functions rather than methods

- Given the copy-on-write mechanism, it is now very cheap to return Image objects. Thus, the more natural ‘free function’ syntax may be used for most image processing functions, instead of the ‘class method’ syntax.
- Example: let’s say I want to pass an image through 3 successive filters, filter1(), filter2() and filter3():

Class method syntax: the filterX() are methods of class Image

```
const Image<float> source;  
Image<float> result1, result2;  
result1.filter1(source);  
result2.filter2(result1);  
result1.filter3(result2);  
result2.freeMem();
```

See Image_*.H

Free function syntax: the filterX() are functions not attached to a class

```
const Image<float> source;  
Image<float> result = filter3(filter2(filter1(source)));
```

iLab C++ Neuromorphic Toolkit

Iterators



- Accessing data via pointers is error-prone, use iterators instead. Our classes that hold some data that can be iterated on provide iterator support very similar to that of the STL classes.
- Example:

See Image.H

```
Image<byte> img;
```

```
Image<byte>::iterator itr = img.beginw(), stop = img.endw();  
while (itr != stop) { *itr++ = 0; }
```

Shared pointers

- When objects communicate with lots of other objects, it is often difficult to know who will run out of scope first. When new memory is allocated for an object that will be passed around and used by several objects, we would like an automatic way of freeing the memory when everybody is done with it.
- Hence the class `SharedPtr<T>` which behaves like a pointer, except that when the last `SharedPtr` to an object runs out of scope, it will destroy/free the memory for that object.
- Example:

In obj1: `SharedPtr<Message> mymsg(new Message());`

In obj2: `SharedPtr<Message> mymsg2(mymsg);
mymsg2->function();`

See `SharedPtr.H`

Message will be destroyed only when its `SharedPtr`'s have run out of scope in both obj1 and obj2.

Elementary core classes

- [Dims](#): for 2D (width, height) dimensions Dims.H
- [Point2D](#): An (i, j) 2D point Point2D.H
- [Point2DT](#): A Point2D plus a time Point2DT.H
- [PixRGB<T>](#): a (red, green, blue) triplet Pixels.H
- [BitObject](#): object defined by connected pixels BitObject.H
- [Timer](#): to count time with arbitrary accuracy Timer.H
- [CpuTimer](#): to measure time and CPU load CpuTimer.H
- [Range](#): specifies a numeric range of values Range.H
- [LevelSpec](#): specifies scales for feature/saliency map LevelSpec.H
- [Rectangle](#): a rectangle Rectangle.H
- [SharedPtr<T>](#): a shared pointer SharedPtr.H
- [VisualEvent](#)
- [VisualObject](#)
- [VisualFeature](#)
- ...

Core definitions



- [Promotions.H](#): the automatic type promotion rules
- [atomic.H](#): atomic (one-CPU-instruction) operations
- [Saliency.H](#): a few generic helper functions like MAX, MIN, etc and basic type definitions like byte, int32, uint64, etc
- [colorDefs.H](#): various default color definitions
- [Log.H](#): comprehensive logging facility
- [StringConversions.H](#): convert various datatypes to/from string
- [TypeTraits.H](#): compile-time information about types
- ...

Logs



- Provide a unified, convenient mechanism for text message output.
- 4 levels: LDEBUG, LINFO, LERROR, LFATAL
- printf()-like syntax
- Automatically adds class/function name, system error messages (use prefix 'P'), a user id (use prefix 'ID'), a line number (compile-time option)
- Can print to stderr or syslog

The hard way:

```
fprintf(stderr, "In myFunction(), could not open file '%s' (error: %s)\n",
        filename, strerror(errno));
>>> In myFunction(), could not open file `test' (error: file not found)
```

The easy way:

```
PLERROR("Could not open file '%s' ", filename);
>>> MyClass::myFunction: Could not open file 'test' (file not found)
```

See log.H

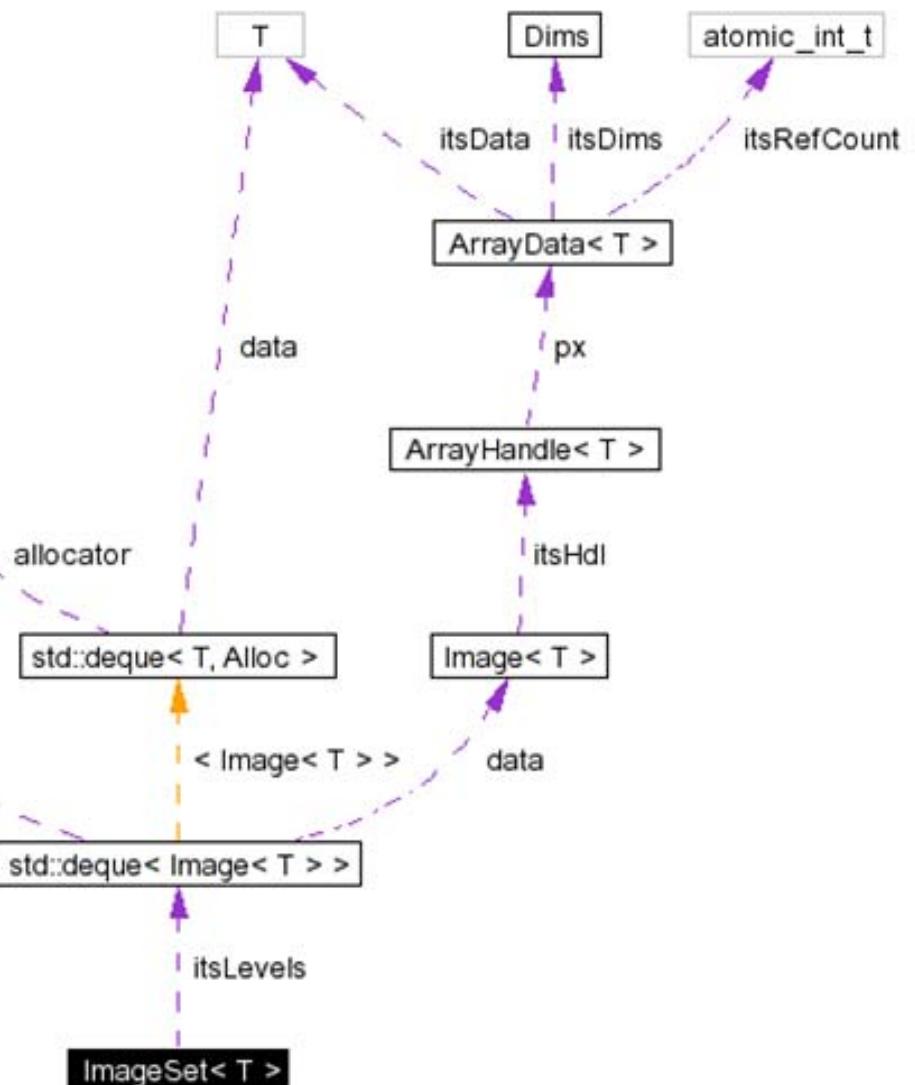
Helper classes



- [Raster](#): to read/write/display Images in various formats
- [V4Lgrabber](#): to grab images from video source (PCI/USB)
- [IEEE1394grabber](#): idem for FireWire cameras
- [XWindow](#): to display image collections & interact
- [VCC4](#): to control pan/tilt/zoom camera
- [SSC](#): to control pan/tilt on beobot camera
- Etc...

ImageSets, a.k.a. Image Pyramids

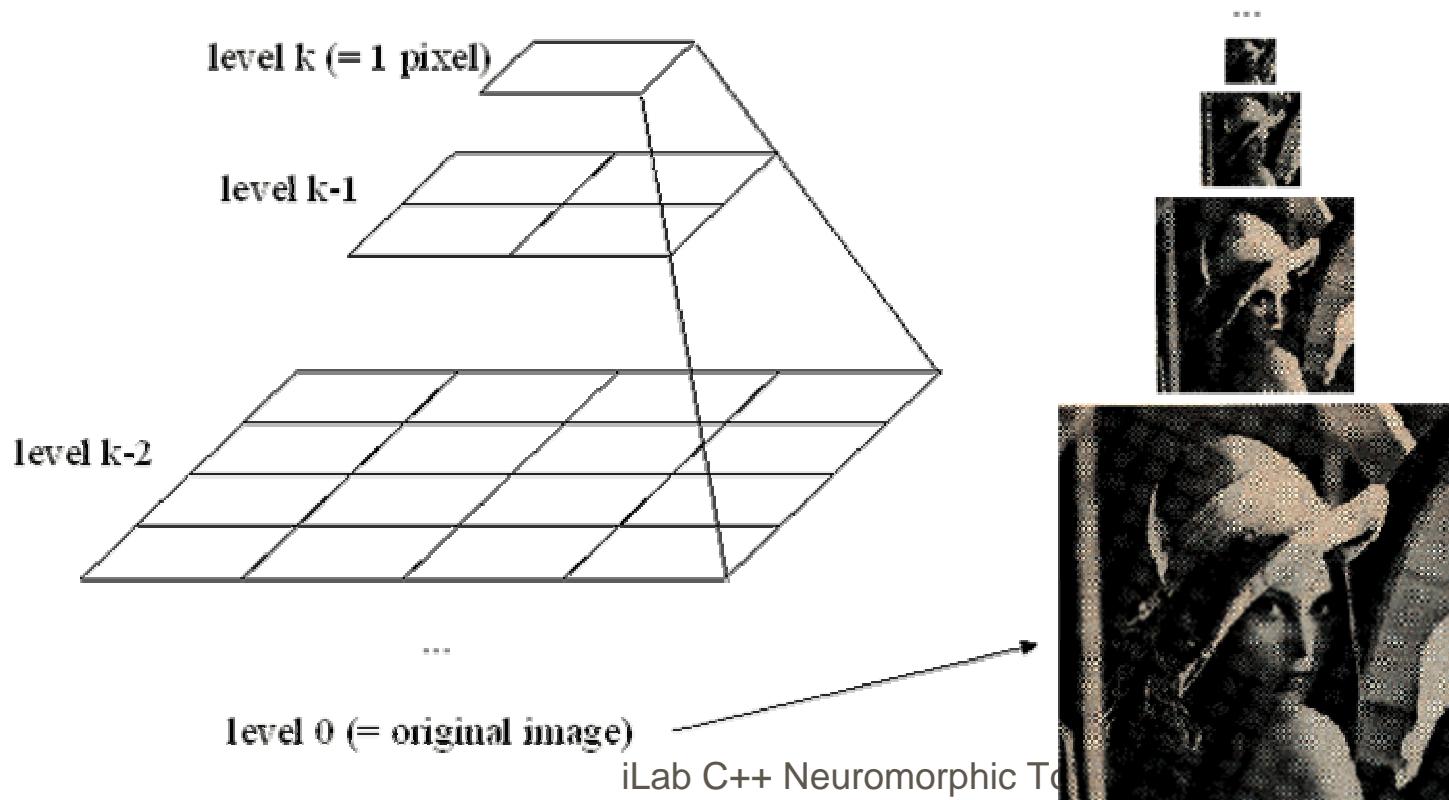
- Collection of images
- Dyadic image reduction from one level to next
- Various filters applied before reduction



See `ImageSet.H`, `PyrBuilder.H`,
`Pyramid_Ops.H`, etc

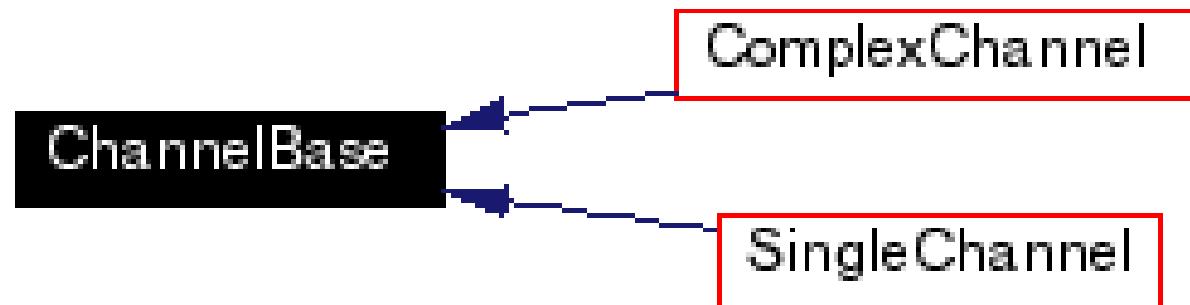
Gaussian Pyramid

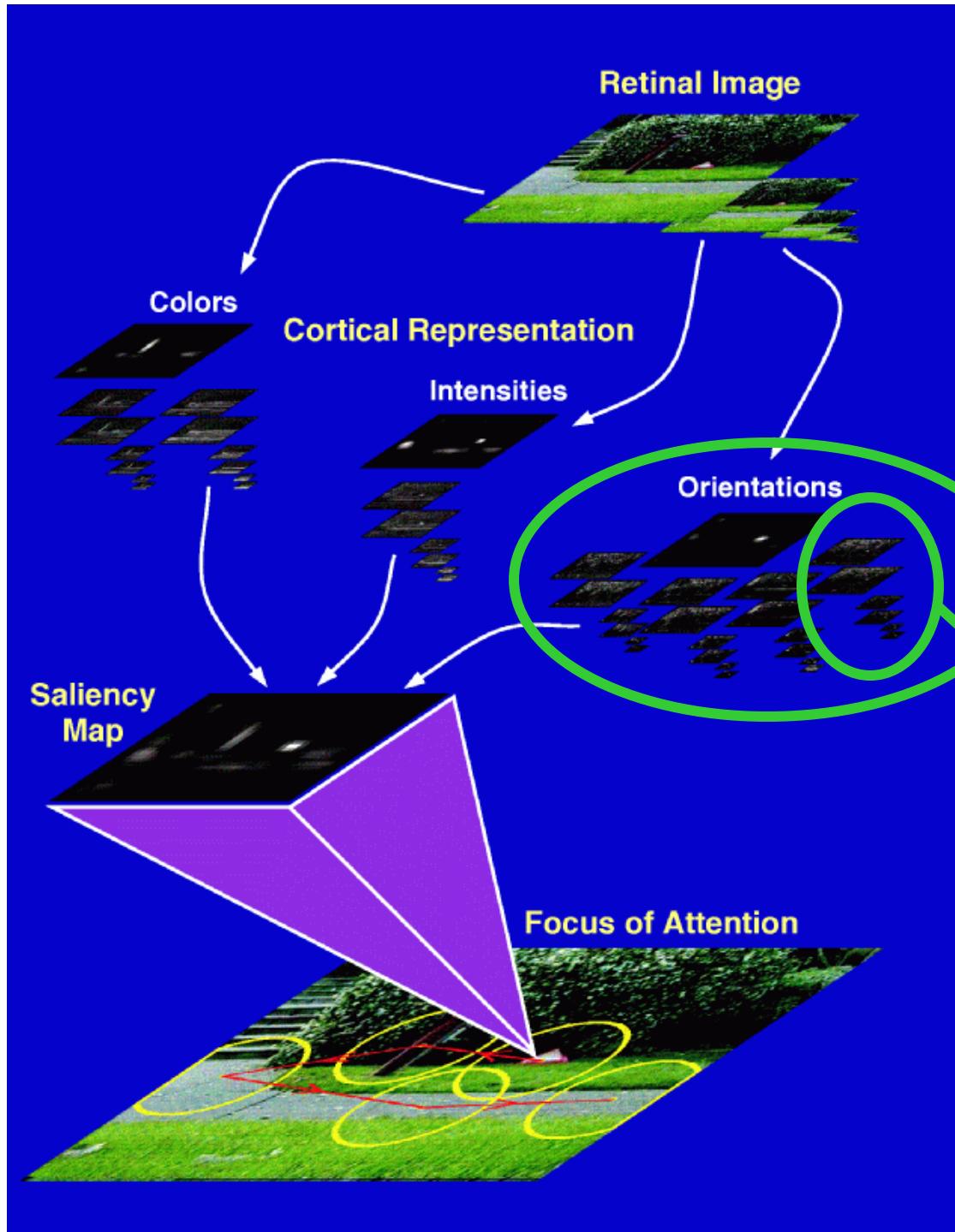
Idea: Represent NxN image as a “pyramid” of
1x1, 2x2, 4x4,..., $2^k \times 2^k$ images (assuming N=2^k)



Channels

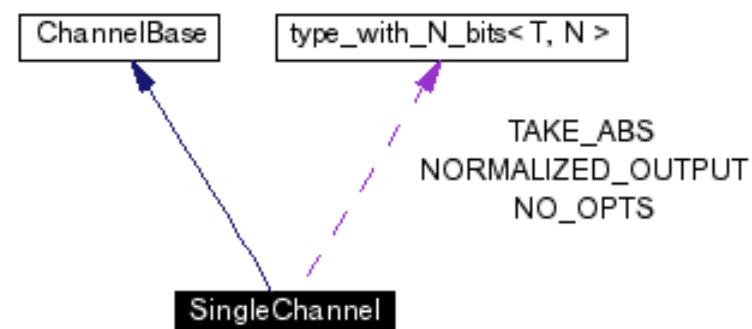
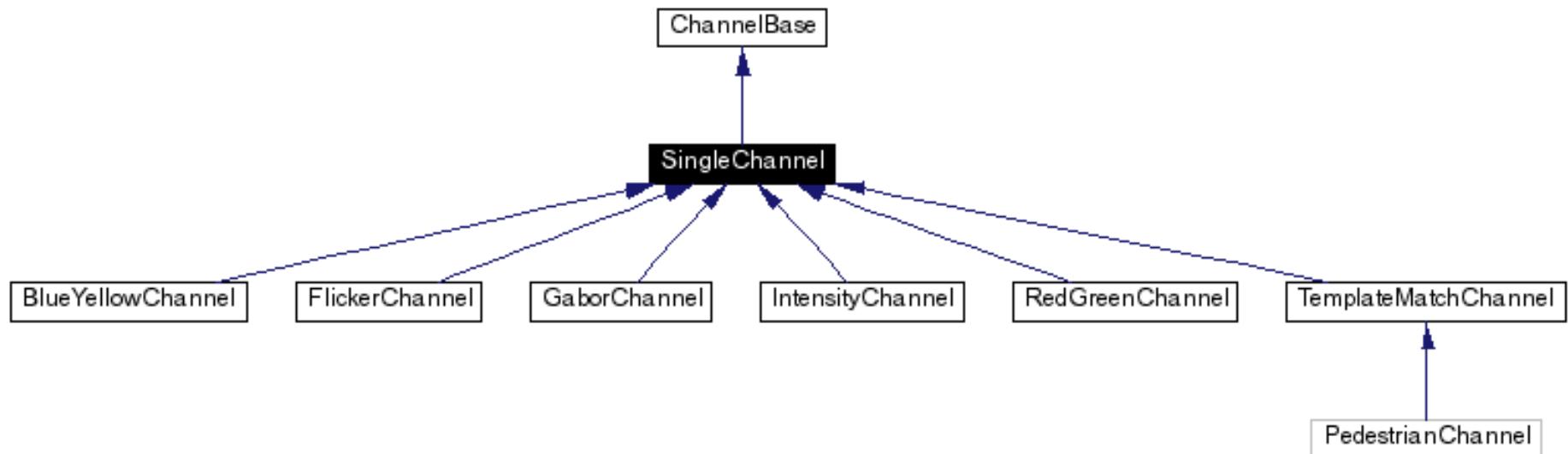
- Implement a pyramid or collection of pyramids plus some I/O functions and additional processing
- Various derived instances can be identified by name
- SingleChannel: contains one pyramid
- ComplexChannel: contains a collection of SingleChannels



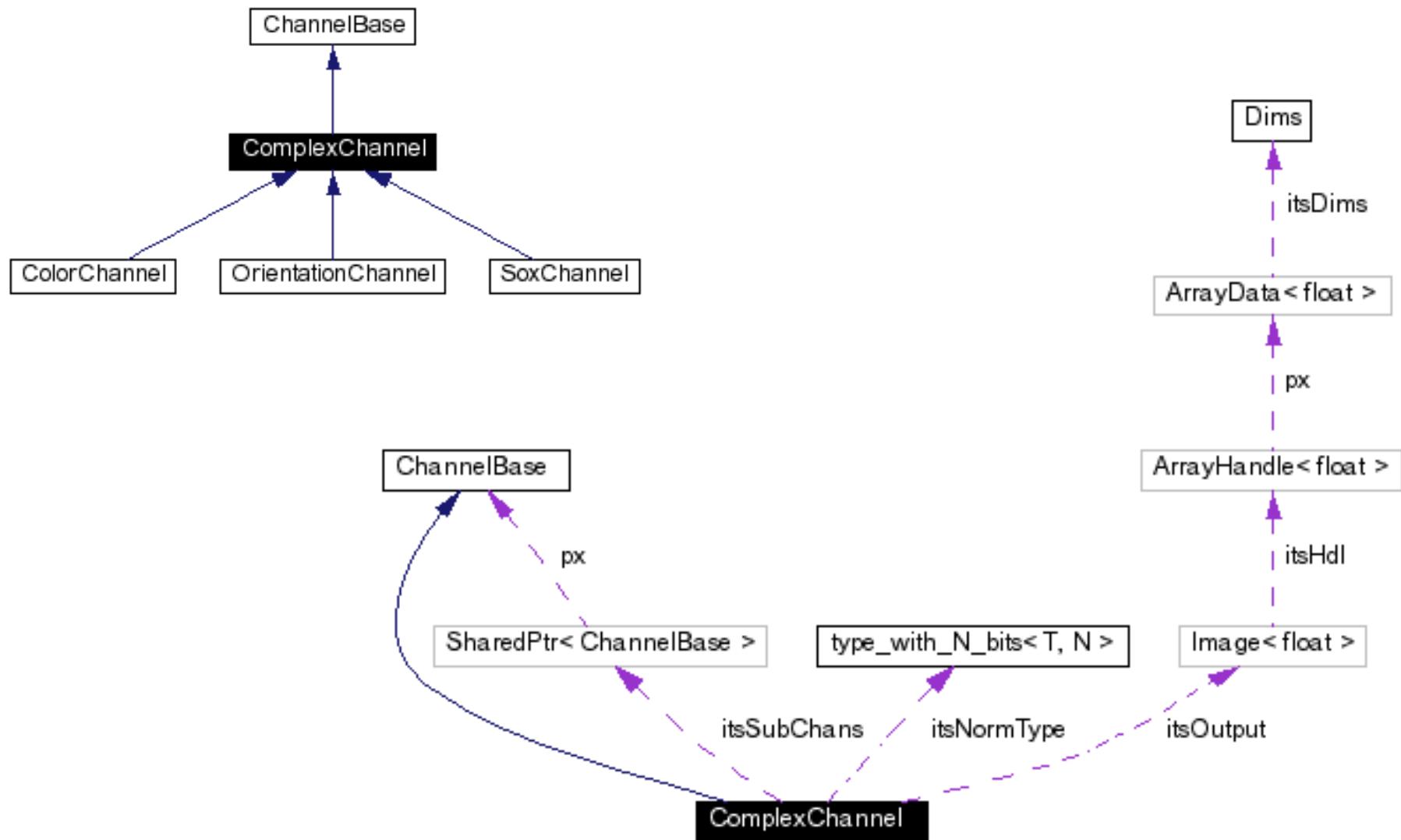


Itti & Koch,
Vision Research 2000
Toolkit

Single Channels

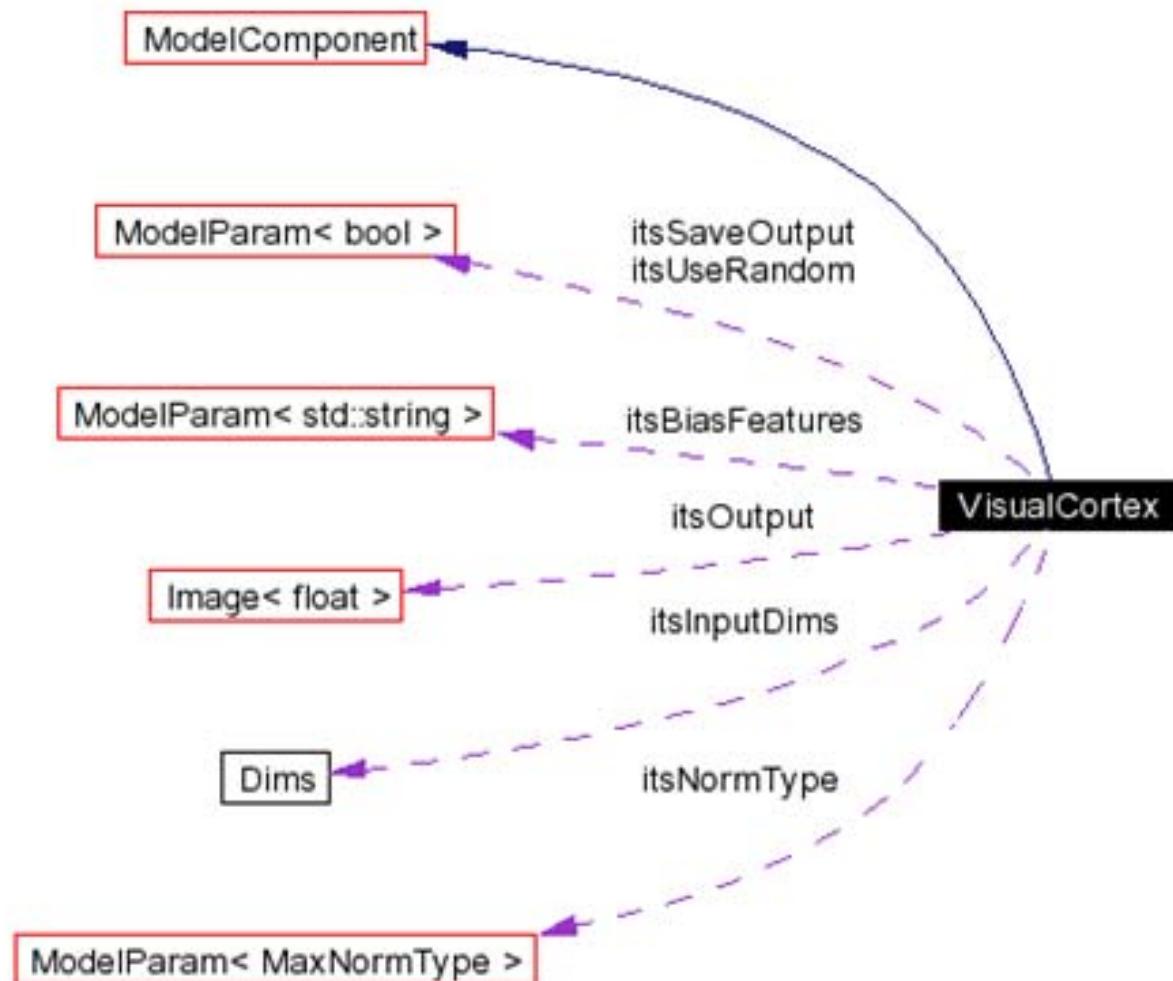


Complex channels



VisualCortex

- Run-time configurable collection of channels, plus additional I/O and access methods



Brain

VisualCortex plugged-in
at run-time



iLab C++ Neuromorphic

Brain: basic operation

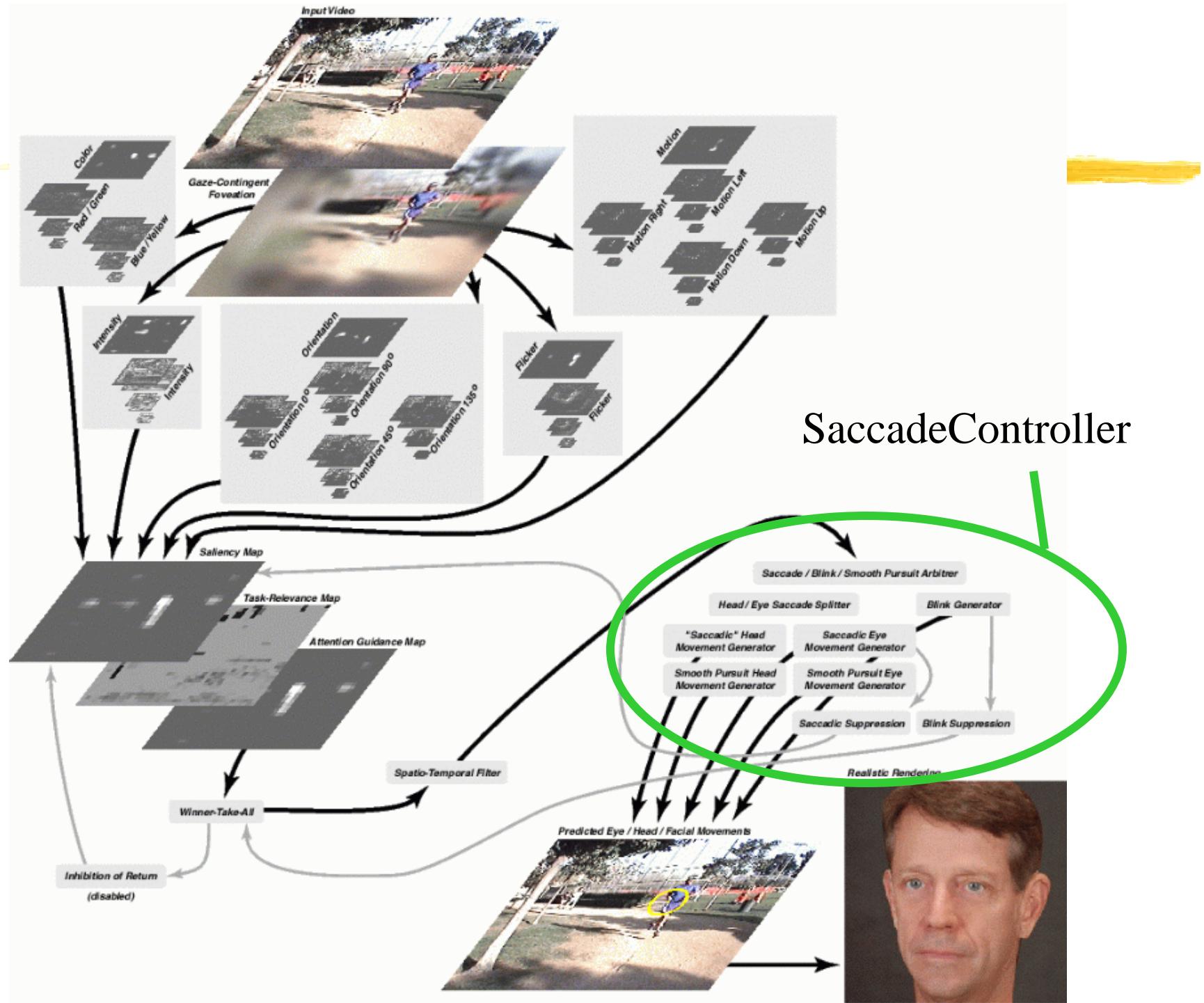


In Brain::input(), called for every new input image

- Get an input image
- Process it through VisualCortex, get saliency map input

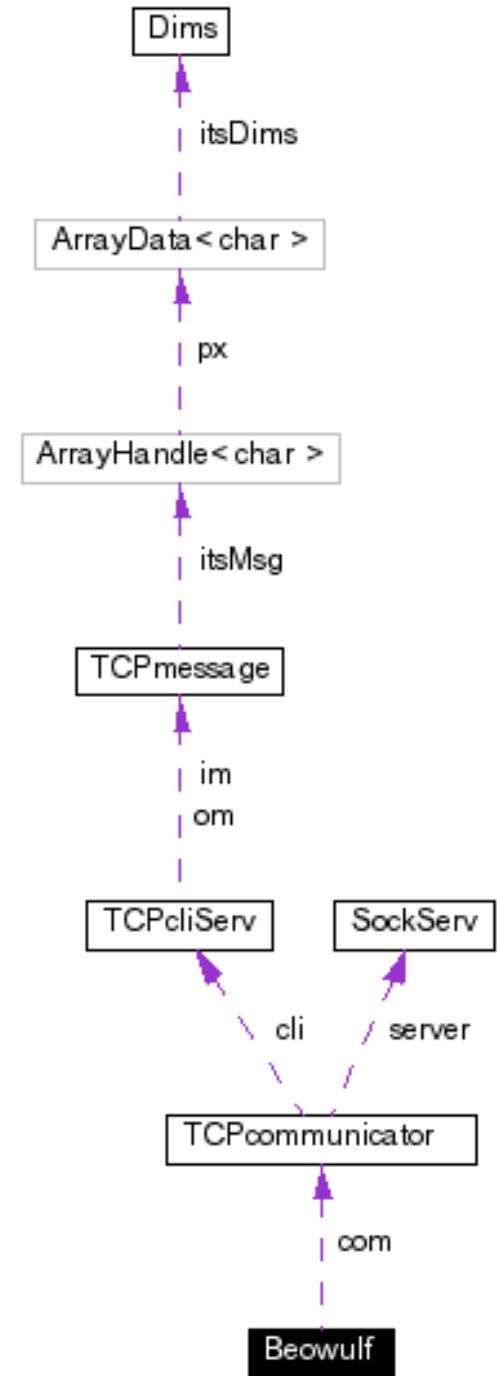
In Brain::evolve(), called every 0.1ms of simulated time

- Feed saliency map
- Let saliency map evolve
- Let task-relevance map evolve
- Combine saliency map and task-relevance map outputs to feed attention-guidance map
- Let attention-guidance map evolve
- Feed output of attention-guidance map to winner-take-all
- Get winner-take all output, if any
- Feed that to saccade controller
- Also feed it to shape estimator
- Activate inhibition of return
- ...

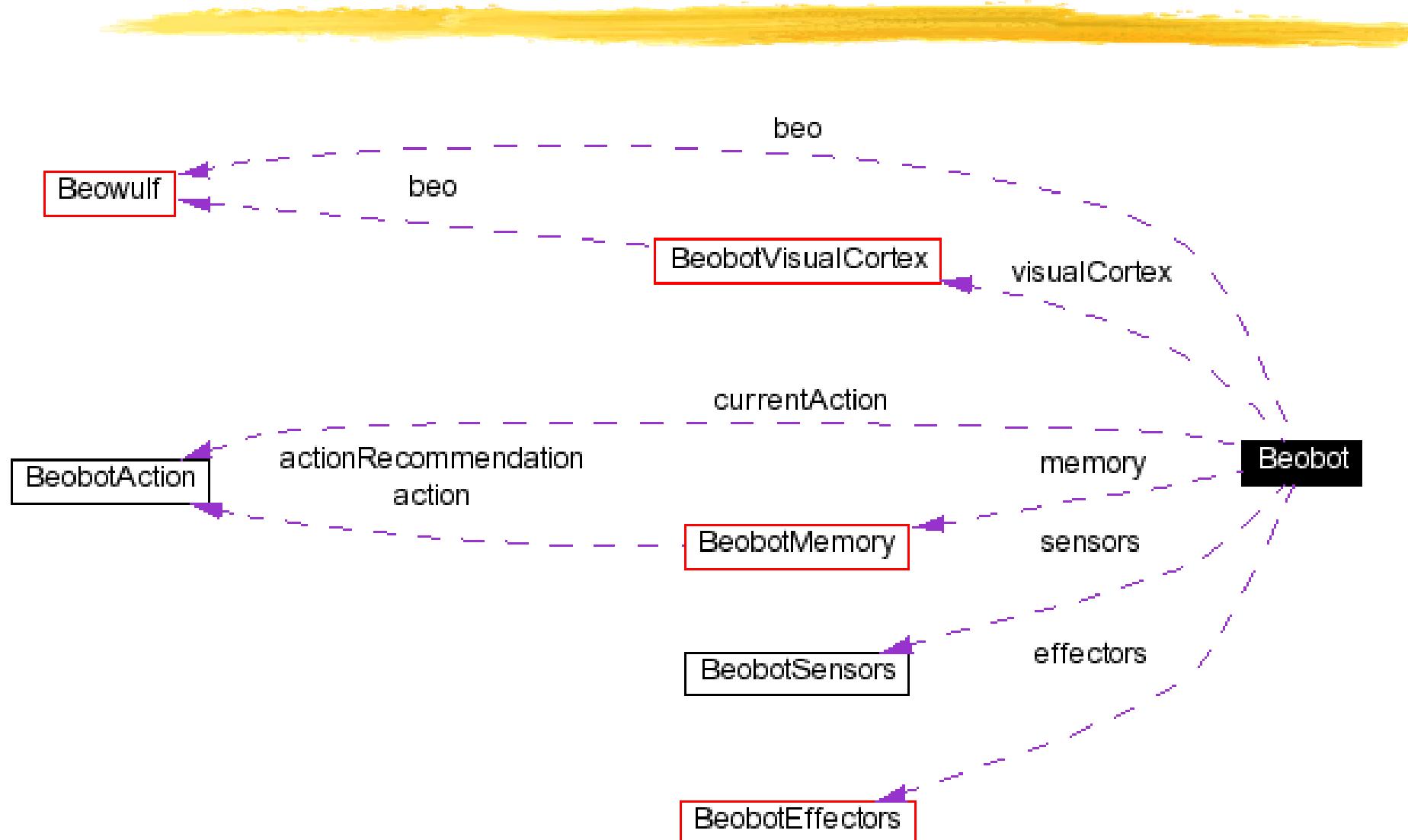


Beowulf

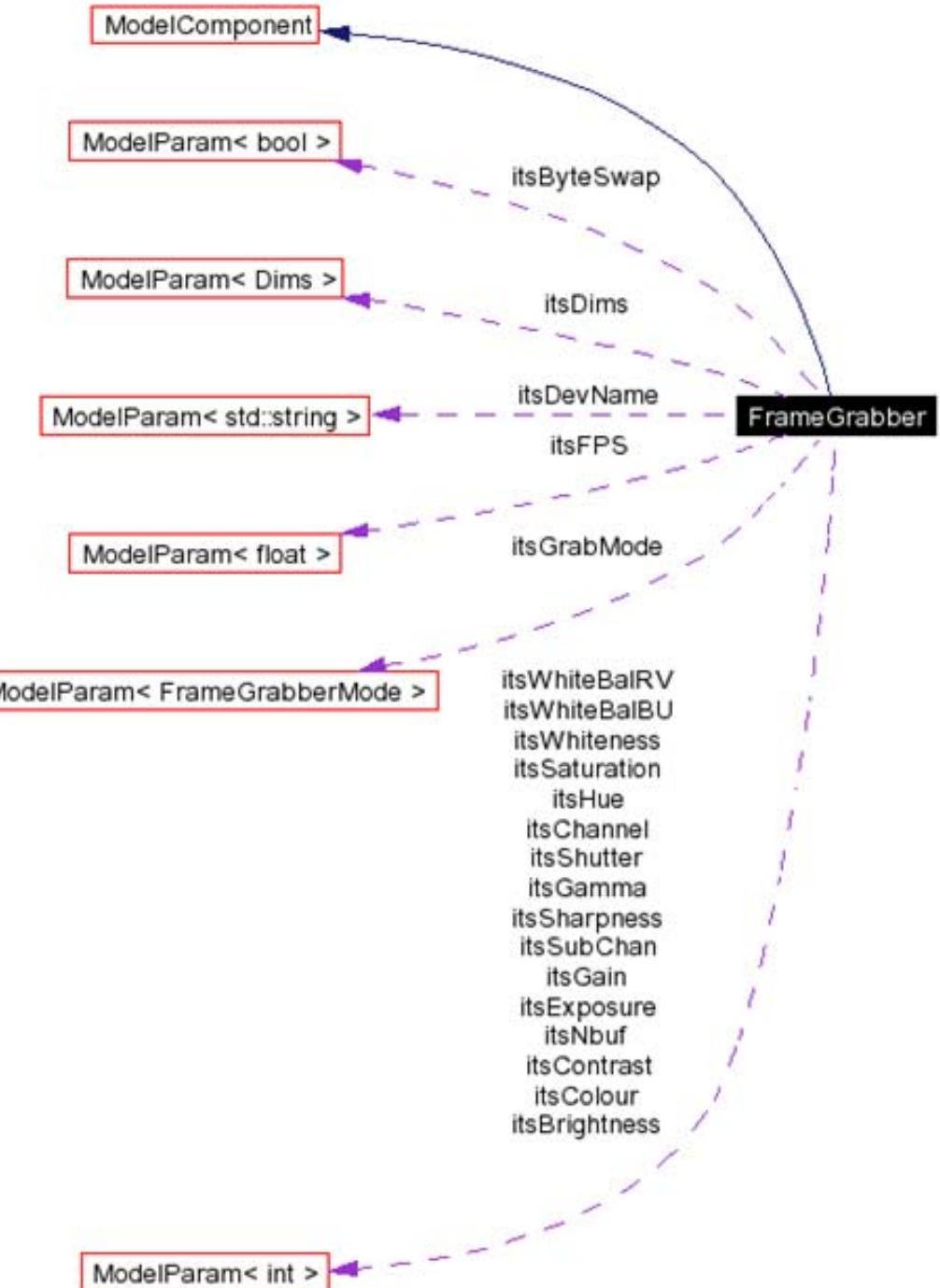
- Multi-threaded class
- Handles transparent passing of TCPmessages
 - TCPmessages are run-time collections of objects
- TCPmessages implemented using COW
- Uses TCP communications for distant nodes
- Uses shared memory for local nodes

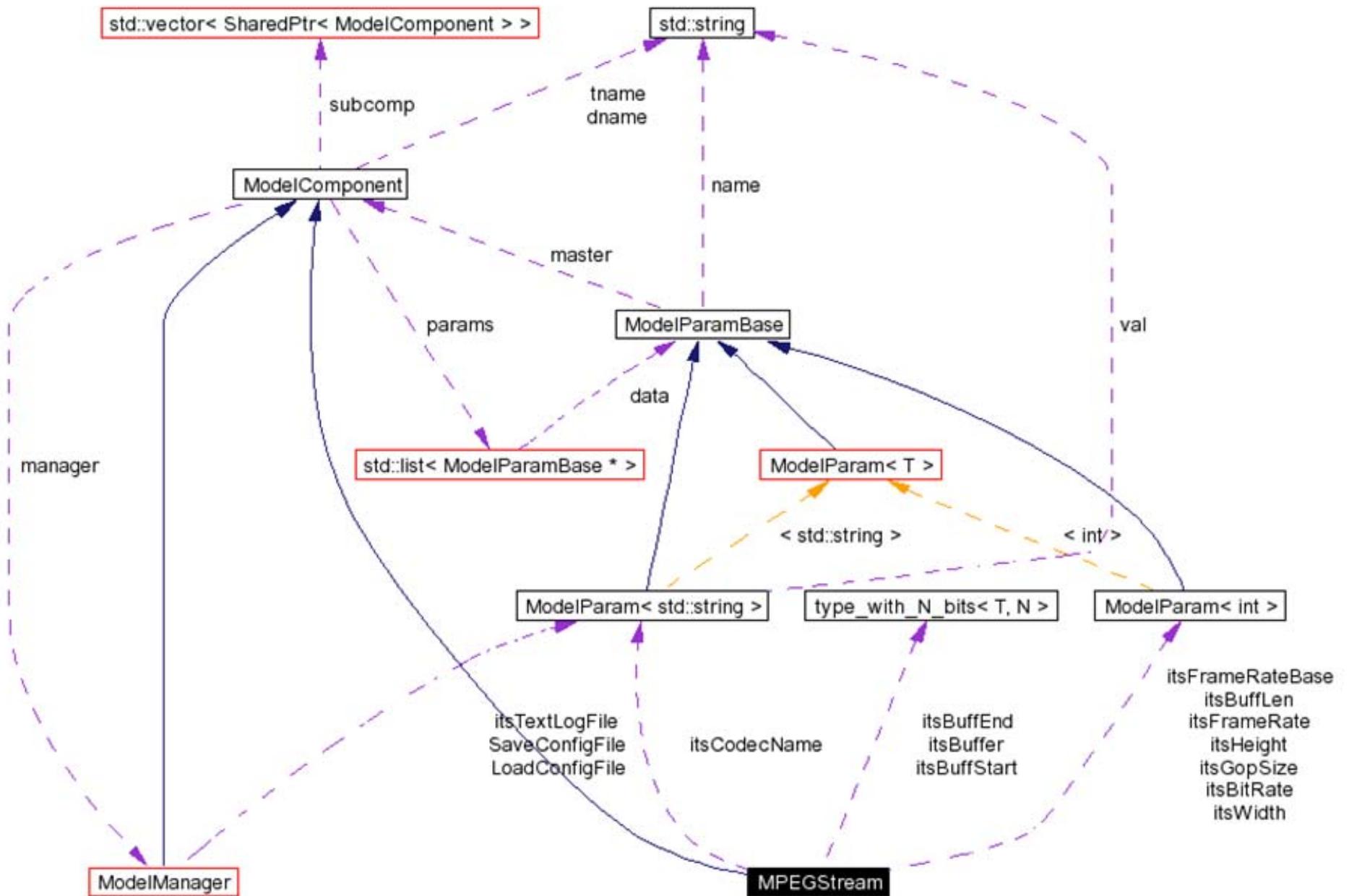


Beobot



FrameGrabber, etc, etc





Welcome to iLab! - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/ Google

USC UNIVERSITY OF SOUTHERN CALIFORNIA

iLab

UNIVERSITY OF SOUTHERN CALIFORNIA

Welcome to iLab at the University of Southern California!

Research Publications

People Facilities

Classes Opportunities

Events + Links

Search

http://ilab.usc.edu/publications/ beobots beobot030901 Welcome to iLab! - Microsoft... Internet 1:12 AM

Welcome to the Beobot Project - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/beobots/ Google

iLab

beobot

Towards Visually-Guided Neuromorphic Robots

[Home](#)

[Overview](#)

[News](#)

[Hardware](#)

[Software](#)

[Gallery](#)

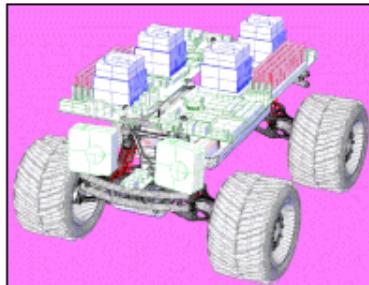
[Downloads](#)

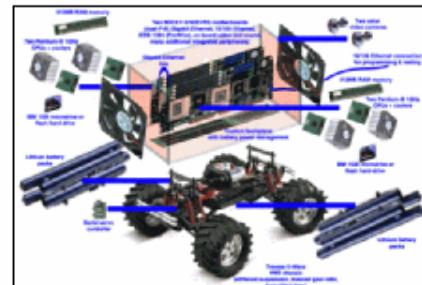
[Team](#)

[Publications](#)

[Sponsors](#)

[Links](#)





Welcome to the Beobot Project!

Beobots are autonomous robots whose brains are standard Linux clusters of computers which run real-time neuromorphic vision algorithms.

Just like **Beowulf Clusters** have revolutionized the world of high-performance computing, replacing costly and slowly-evolving custom supercomputer hardware by assemblies of inexpensive, mass-produced personal computers, we hope that Beobots (a **Beowulf** cluster on a mobile **robot**) will lead the way towards a new generation of robotics systems that are inexpensive, rapidly evolving, built from standard mass-produced components, and armed with sufficient computational power to run real-time neuromorphic vision algorithms.

- So **what exactly** is a Beobot?
- What **hardware** is it made of?
- What **software** does it run?
- Who are the **people** working on it?

Last CVS commit: Mon Sep 1 19:06:20 2003

Done Start beobots beobot030901 Welcome to the Beobot Pro... 1:14 AM

2 iLab C++ Neuromorphic Vision Toolkit Coding Home Page - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/toolkit/ Google

Recent CVS / Forum Activity

Ordered by last CVS commit date/time.

User	Last CVS Commit	Version	Last iLab Forum Post
walther	2003-09-01 at 19:06 saliency/src3/shapeEstimatorWebpage.C	1.5	Wed Aug 27 08:41:39 2003
zhanshi	2003-08-29 at 10:15 saliency/src3/dummySTL.H	1.3	Sun Aug 31 19:44:22 2003
itti	2003-08-27 at 11:04 saliency/src3/SimulationViewerEyeMvt.C	1.9	Fri Aug 29 14:04:17 2003
mundhenk	2003-08-19 at 20:40 saliency/src3/stats.conf	1.21	Sat Aug 23 15:57:03 2003
rjpeters	2003-08-02 at 11:06 saliency/src3/corrcoef.C	1.1	Wed Jul 23 18:03:45 2003
vidhya	2003-06-25 at 01:03 saliency/src3/VisualCortex.H	1.73	Sun Jan 12 11:43:12 2003
daesu	2003-05-13 at 14:40 saliency/src3/test-roadShape.C	1.1	---
dhavale	2003-05-09 at 18:58 saliency/src3/wrapping/test-Cam.C	1.2	Wed Aug 20 05:47:41 2003
beobot	2003-02-20 at 21:56 saliency/bin/bbsync	1.5	---
rirata	2002-10-23 at 14:54 beobots/software/gyro/Gyro2.C	1.2	---
jsn	2001-12-10 at 13:16 beobots/software/lcd/lcd.C	1.2	---
juliet	---		Sun Jun 1 23:25:23 2003
aprilla	---		Mon Jun 3 01:53:33 2002

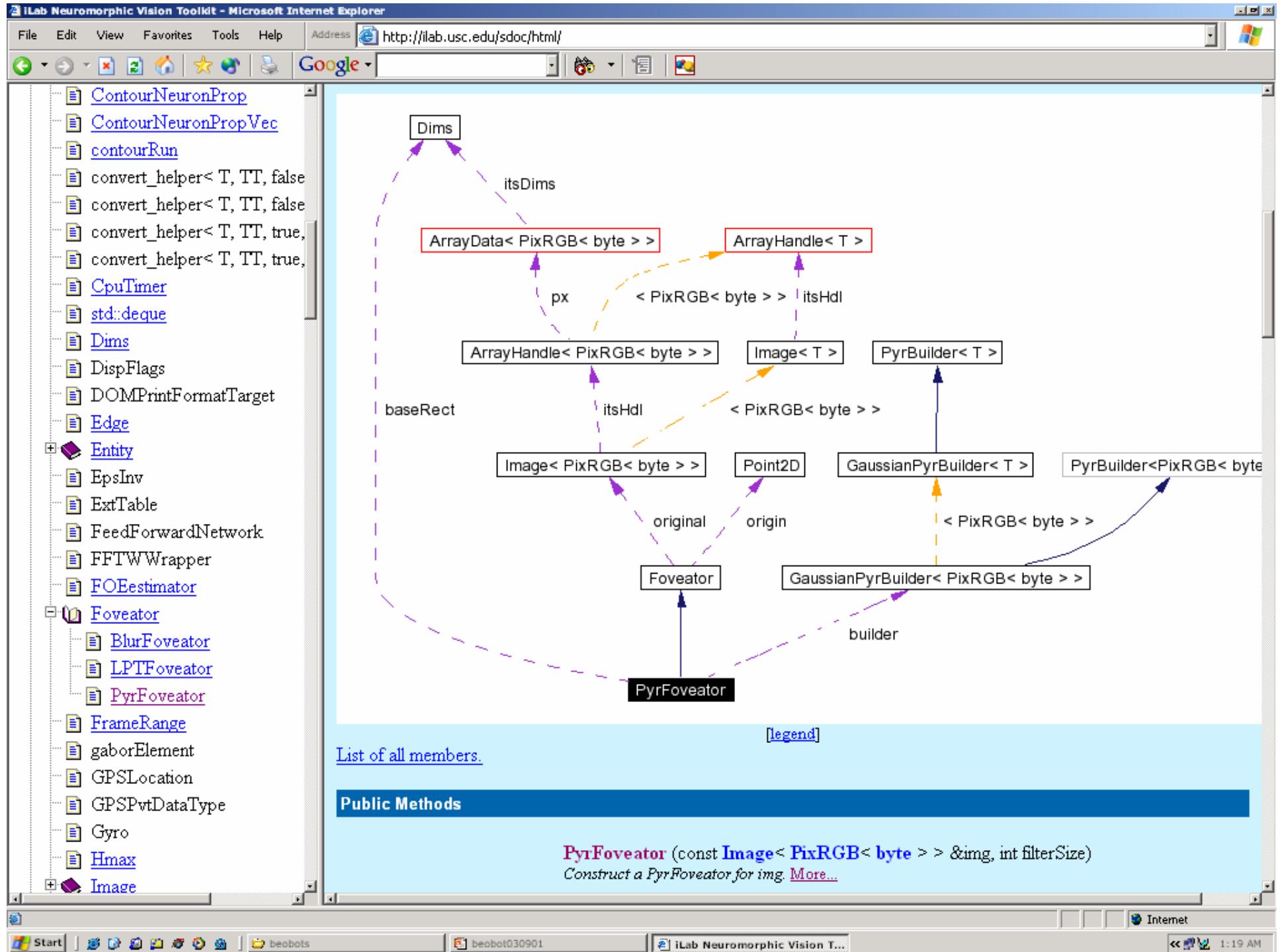
Latest CVS commits

Ordered by commit date/time.

Date	Time (PST)	User	Version	File	Log Message

Done Start Internet 1:16 AM

beobots beobot030901 iLab C++ Neuromorphic Vis... 1:16 AM



iLab Forum - Index - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/cgi-bin/yabb/YaBB.pl

Google

iLab Forum

Hey, Laurent Itti, you have 7 messages.
Sep 2nd, 2003, 1:20am

Home Help Search Members Profile Notification Logout

iLab Forum « Index »

iLab Forum

News

Forum name	Topics	Posts	Last post
General			
News Read about the latest happenings of iLab Moderators: Forum Admin , Laurent Itti	17	25	Apr 25 th , 2003, 1:46pm by Laurent Itti
Openings Find openings for positions at iLab Moderators: Forum Admin , Laurent Itti	2	3	Dec 3 rd , 2002, 8:46am by Laurent Itti
C++ Neuromorphic Vision Toolkit			
General Discussion General discussion around the iLab C++ Neuromorphic Vision Toolkit Moderators: Forum Admin , Laurent Itti	35	258	Aug 30 th , 2003, 2:19am by lynnxn
Bugs Bugs and other problems Moderators: Forum Admin , Laurent Itti	32	207	Aug 29 th , 2003, 12:18pm by yamini
Feature Requests Feature Requests Moderators: Forum Admin , Laurent Itti	22	155	Aug 31 st , 2003, 7:44pm by zhanshi
Neuroscience Issues Discussion of neuroscience issues and their implementation in the toolkit Moderators: Forum Admin , Laurent Itti	4	59	Oct 31 st , 2002, 2:44pm by Dirk Walther
Architecture Issues Discussion of general architecture issues, in particular regarding the abstraction of brain operating	4	72	Jun 28 th , 2003, 12:00pm by zhanshi

Done

Start | beobots | beobot030901 | iLab Forum - Index - Micros... | 1:20 AM

iLab Forum - Feature Requests - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/cgi-bin/yabb/YaBB.pl?board=features

Google

iLab Forum **Forum**

Hey, Laurent Itti, you have 7 messages.
Sep 2nd, 2003, 1:21am

Home Help Search Members Profile Notification Logout

iLab Forum « Feature Requests »

iLab Forum
C++ Neuromorphic Vision Toolkit
Feature Requests (Moderators: Forum Admin, Laurent Itti)

Feature Requests

Pages: 1 2

Mark Topics as Read Start new topic

Subject	Started by	Replies	Views	Last post
X-Windows as command-line option	zhanshi	14	142	Aug 31 st , 2003, 7:44pm by zhanshi
IEEE1394 update	Laurent Itti	6	76	Aug 30 th , 2003, 9:27pm by zhanshi
STL, doxygen, and graphviz	zhanshi	4	53	Aug 29 th , 2003, 10:43am by Laurent Itti
Added PNG write capability	Rob Peters	1	16	Jul 23 rd , 2003, 6:34pm by Laurent Itti
Methods for computing orientations	Dirk Walther	7	71	Apr 28 th , 2003, 11:12am by Laurent Itti
Dust off the Raster interface?	Rob Peters	11	88	Mar 21 st , 2003, 5:49pm by Rob Peters
Pyramids; LOGVERB+FULLTRACE « Pages 1 2 »	Rob Peters	22	185	Mar 8 th , 2003, 4:53pm by Laurent Itti
tests that take a long time	Laurent Itti	11	96	Mar 6 th , 2003, 4:29pm by Laurent Itti
threadsafe refcounting	Laurent Itti	4	54	Jan 25 th , 2003, 11:20am by Laurent Itti
Detection of targets by biasing features	vidhya	1	57	Jan 13 th , 2003, 5:20pm by Laurent Itti
itsLevels in PyramidBase	Dirk Walther	4	67	Dec 3 rd , 2002, 11:44am

Done

Start beobots beobot030901 iLab Forum - Feature Reque... 1:21 AM

iLab Publications - University of Southern California - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://ilab.usc.edu/publications/ Google

USC iLab University of Southern California Hedco Neuroscience Building Los Angeles, CA 90089-2520 - USA

Publications

Welcome to the iLab Publication Server!

115 publications, 73 with abstract, 55 available as PDF.

Publications by Year

in-press	2003	2002	2001	2000	1999	1998	1997	1996	1995	1994	1992	1991
1990	1989											

Publications by Type and by Theme

- All Publications
- Journal Articles
- Publications in Press
- Book Chapters
- Proceedings from International Conferences
- Master Theses
- Patents and Copyrights
- Ph.D. Theses
- Beobots
- Model of Bottom-Up Saliency-Based Visual Attention
- Computer Vision
- Human Eye-Tracking Research
- Functional Neuroimaging
- Medical Research
- Medical Image Processing
- Computational Modeling
- Press Coverage
- Human Psychophysics
- Review Articles and Chapters

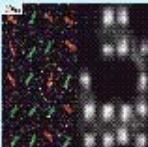
Done Internet

Start 0312_iNTIntro Microsoft PowerPoint - iNVT-in... iLab Publications - Universi... 3:05 PM

This project was started at Caltech with **Prof. Christof Koch**. It is actively being pursued both here and at Caltech (both jointly and in different directions).

 **The Theory**
Details about the trainable model of bottom-up, task-independent visual attention under development in our laboratory.

 **The Images**
A short overview of example images and the corresponding attentional trajectories. Test images, psychophysical stimuli, target detection images, natural scenes, artwork, etc.

 **The Movies**
Several MPEG movies showing attentional trajectories and the temporal dynamics of the Saliency Map for test, psychophysical, artistic and natural images. Also shown are 3D warping of the original image onto the evolving saliency map.

 **The Interactive Demo**
An interactive demonstration of the dynamic behavior of our attentional model, for a variety of complete image databases. Most recent Java™-aware Web browser required.

 **The Publications**
Some pre-versions of our papers describing this research are available in HTML, Postscript and PDF format.

 **The Ongoing Projects**
New! Previews of a few of our ongoing projects and preliminary screenshots. These include our **SaliencyVehicle** off-road muscle car, our real-time **SaliencyCam** which computes attentional deployment on live video feeds (15 frames/s), our **SaliencyAgent** which detects salient pedestrians in natural color scenes, and other exciting projects.

 **The C++ Source Code**
The C++ source code and associated doxygen documentation are available through our CVS server. You will need the latest version of g++ (3.x) and several non-standard packages installed on your Linux distribution (e.g., IEEE1394

2 Nerd-Cam at USC iLab - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://nerd-cam.com/ Google

iLab Home Nerd Cam Gallery KLab Home Beobot home CINNIC Page WHAT IS THIS? Virtual Retina USC Home

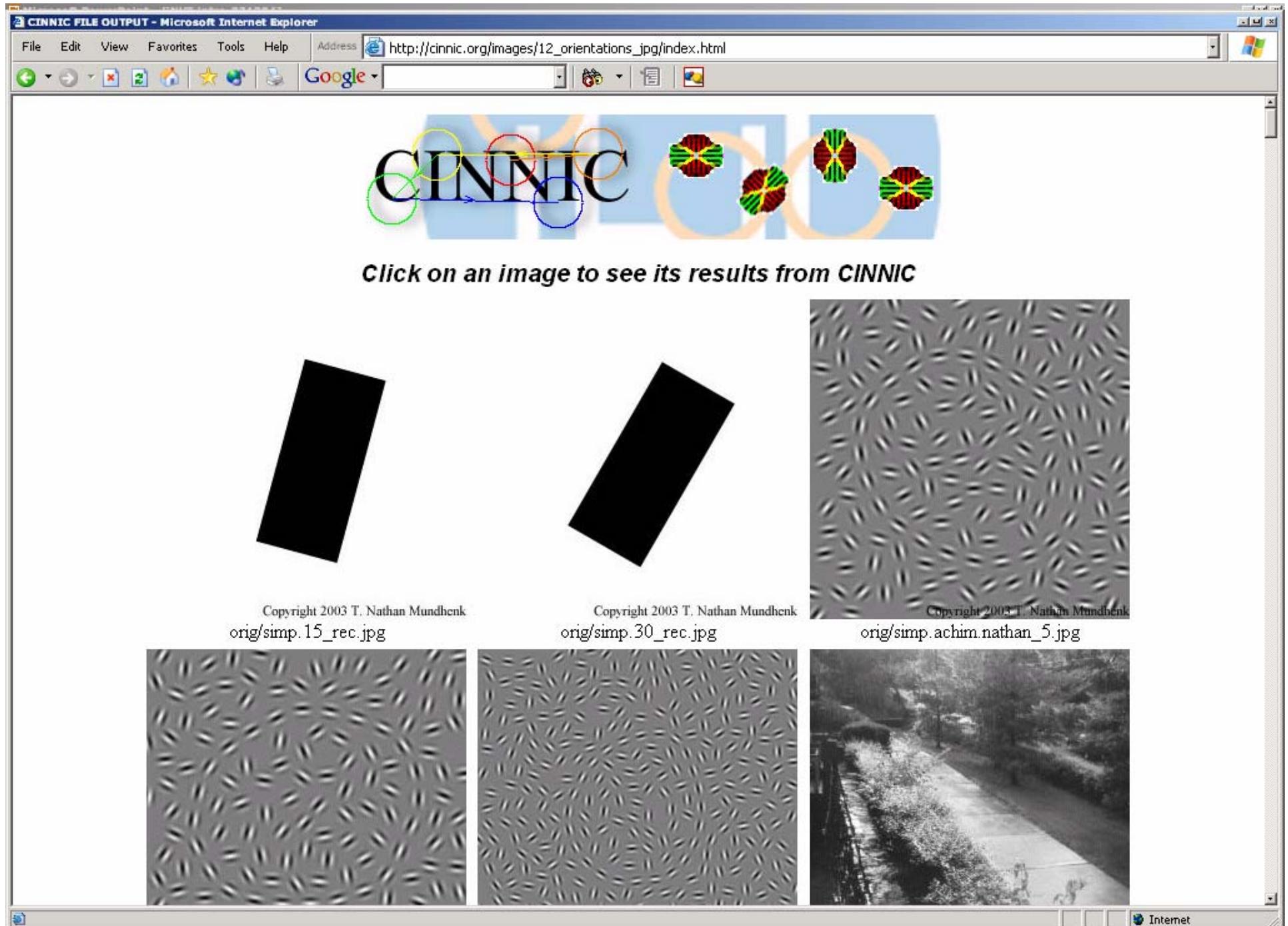
This page will update in 40 seconds: The Lab is OPEN: luminosity 127.652

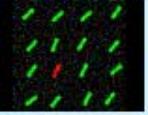
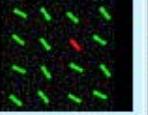
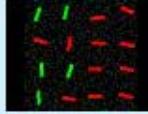
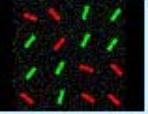
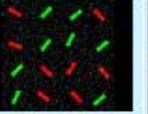
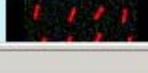
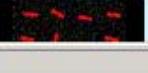
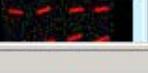
OPEN: since Mon Sep 22 22:47:21 PDT 2003 - luminosity 146.664 13 23:12:02 PDT 2003 - luminosity 157.895

Live Cam: Most Salient Points Live Cam: Most Salient Contours

Done Internet

Start 0312_INVIntro Microsoft PowerPoint - iNVT-in... Nerd-Cam at USC iLab - Mic... 3:07 PM



iLab Image Databases						
Samples	Database	# Images	Size	Description	Master Reference	
  	STIMart.tar	20	4.0 MB	Miscellaneous artwork, posters and portraits	Itti et al., IEEE PAMI, 1998	
  	STIMautobahn.tar	90 + 90	56 MB	Color images with German traffic signs + target masks	Itti & Koch, J. Elec. Imag., 2001	
  	STIMcoke.tar	104 + 104	53 MB	Color images with a red can + target masks	Itti & Koch, J. Elec. Imag., 2001	
  	STIMcolor.tar	180 + 180	2.1 MB	Color popout search arrays + target masks	Itti & Koch, Vis. Res., 2000	
  	STIMoricol.tar	180 + 180	2.1 MB	Orientation/color conjunctive search arrays + target masks	Itti & Koch, Vis. Res., 2000	
  		180 +	2.1	Orientation popout search arrays +	Itti & Koch, Vis. Res.	