

From attention to goal-oriented scene understanding

Laurent Itti – University of Southern California





Tourists.mov



File Edit Movie QTV Window Help



00:00:00





Tourists.mov



File Edit Movie QTV Window Help



00:00:00



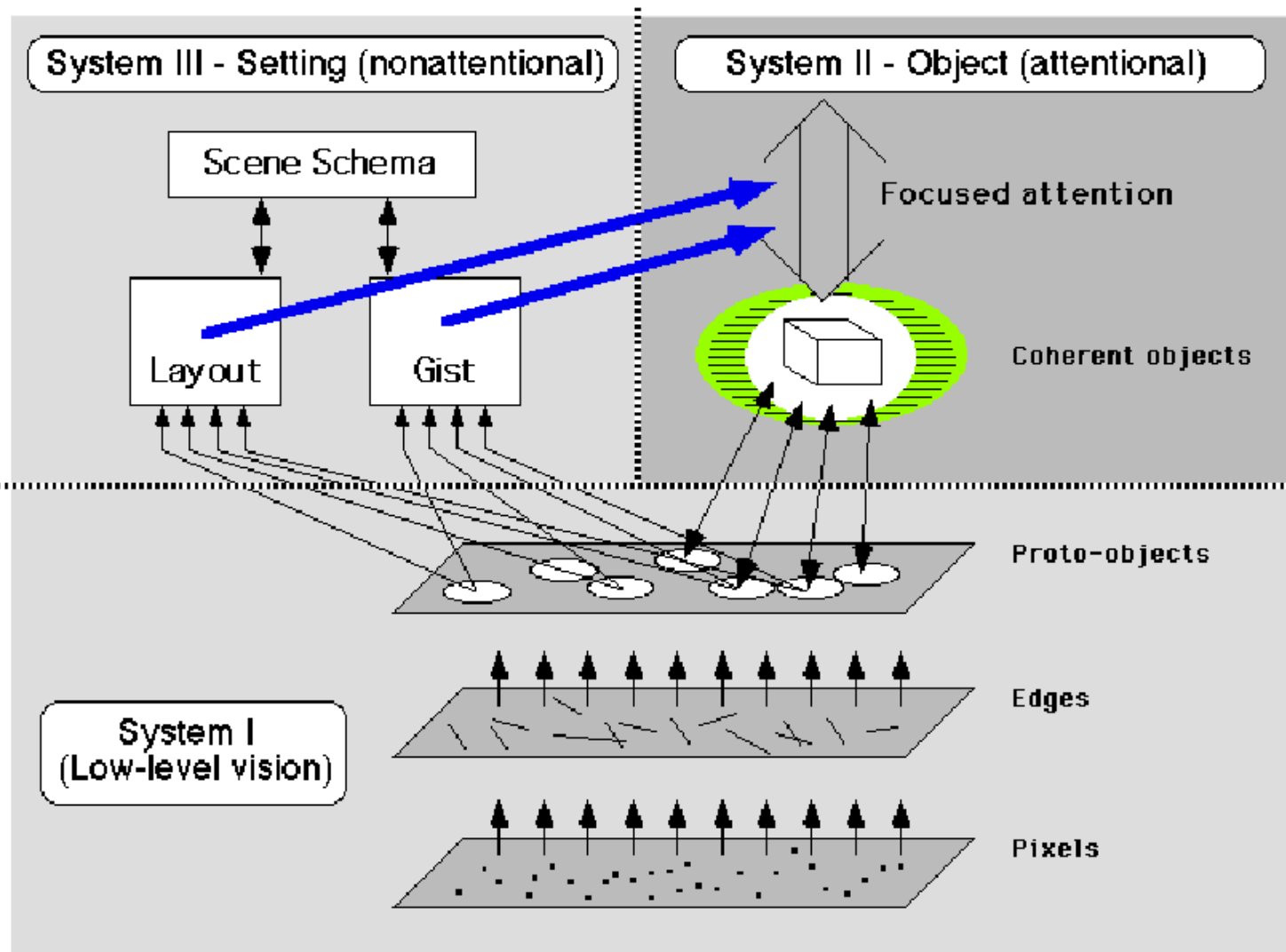


Figure 4

Figure 4. Triadic Architecture. It is suggested that the visual perception of scenes may be carried out via the interaction of three different systems. System I: Early-level processes produce volatile proto-objects rapidly and in parallel across the visual field. System II: Focused attention acts as a hand to "grab" these structures; as long as these structures are held, they form an individuated object with both temporal and spatial coherence. System III: Setting information—obtained via a nonattentional stream—guides the allocation of focused attention to various parts of the scene, and allows priorities to be given to the various possible objects.

Goal-oriented scene understanding?

- Question: describe what is happening in the video clip shown in the following slide.



Goal for our algorithms

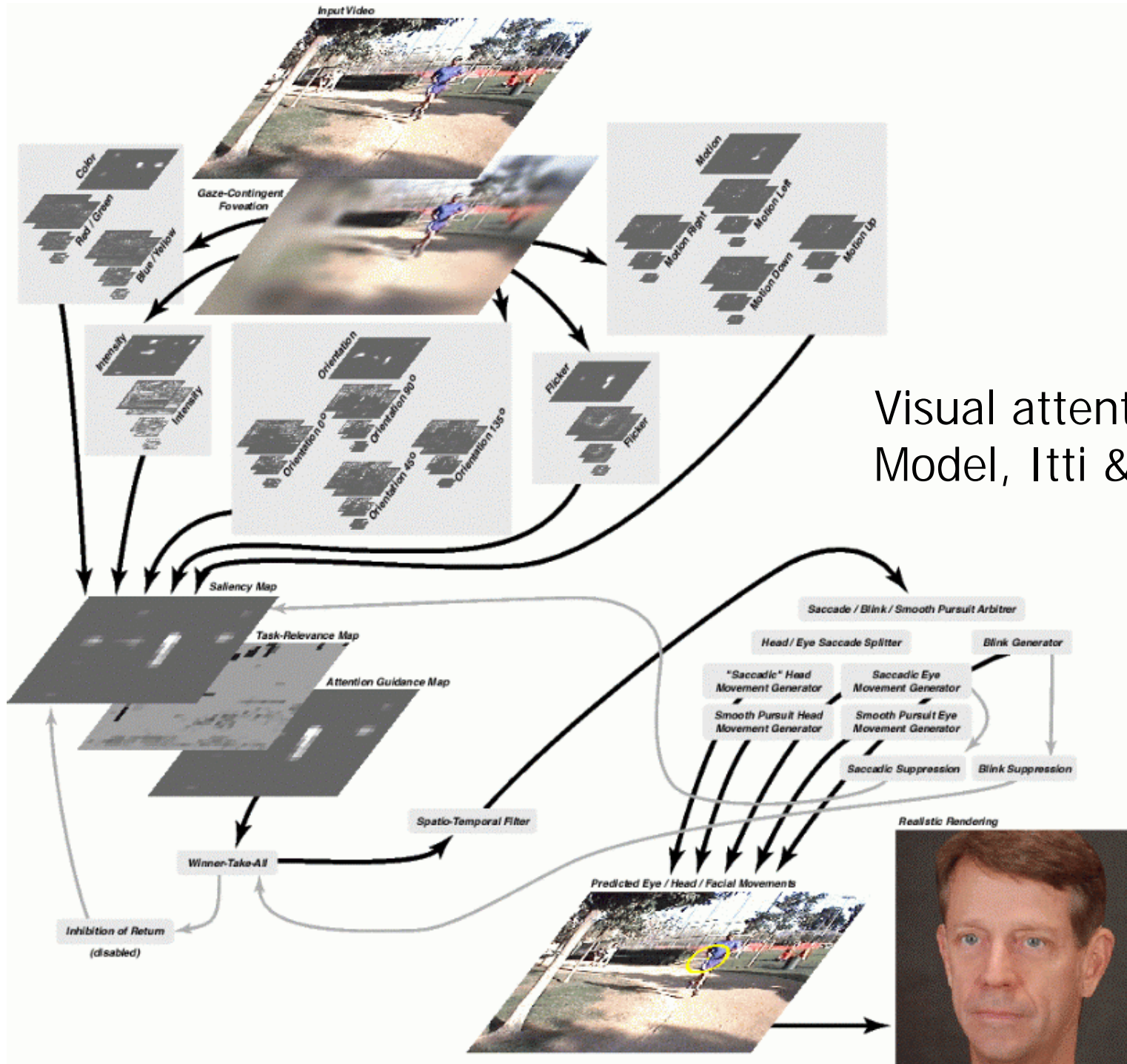
- Extract the “minimal subscene[®]”, that is, the smallest set of actors, objects and actions that describe the scene under given task definition.
- E.g.,
 - If “who is doing what and to whom?” task
 - And boy-on-scooter video clip
 - Then minimal subscene is “a boy with a red shirt rides a scooter around”

Challenge

- The minimal subscene in our example has **10 words**, but...
- The video clip has over 74 million different pixel values (about **1.8 billion bits** once uncompressed and displayed – though with high spatial and temporal correlation)
- Note: The concept of minimal subscene has further linkages to the evolution of language in humans, investigated by Itti and Arbib at USC but not explored here.

Starting point

- Can attend to salient locations (next slide)
- Can identify those locations?
- Can evaluate the task-relevance of those locations, based on some general symbolic knowledge about how various entities relate to each other?



Visual attention Model, Itti & Koch



Task influences eye movements

- Yarbus, 1967:
 - Given one image,
 - An eye tracker,
 - And seven sets of instructions given to seven observers, ...
- ... Yarbus observed widely different eye movement scanpaths depending on task.

Yarbus, 1967: Task influences human eye movements

1) Free examination

2) estimate material circumstances of family

3) give ages of the people

4) surmise what family has been doing before arrival of "unexpected visitor"

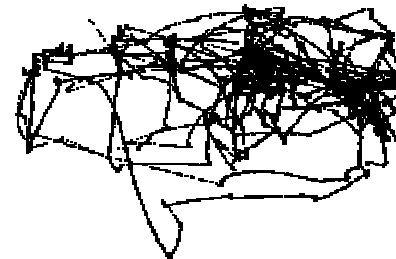
5) remember clothes worn by the people

6) remember position of people and objects

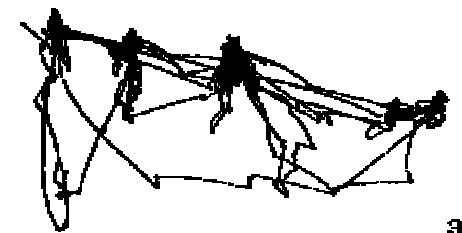
7) estimate how long the "unexpected visitor" has been away from family



1



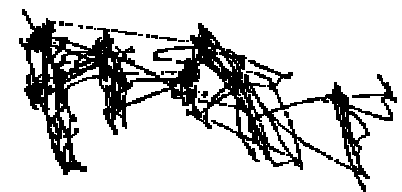
2



3



4



5

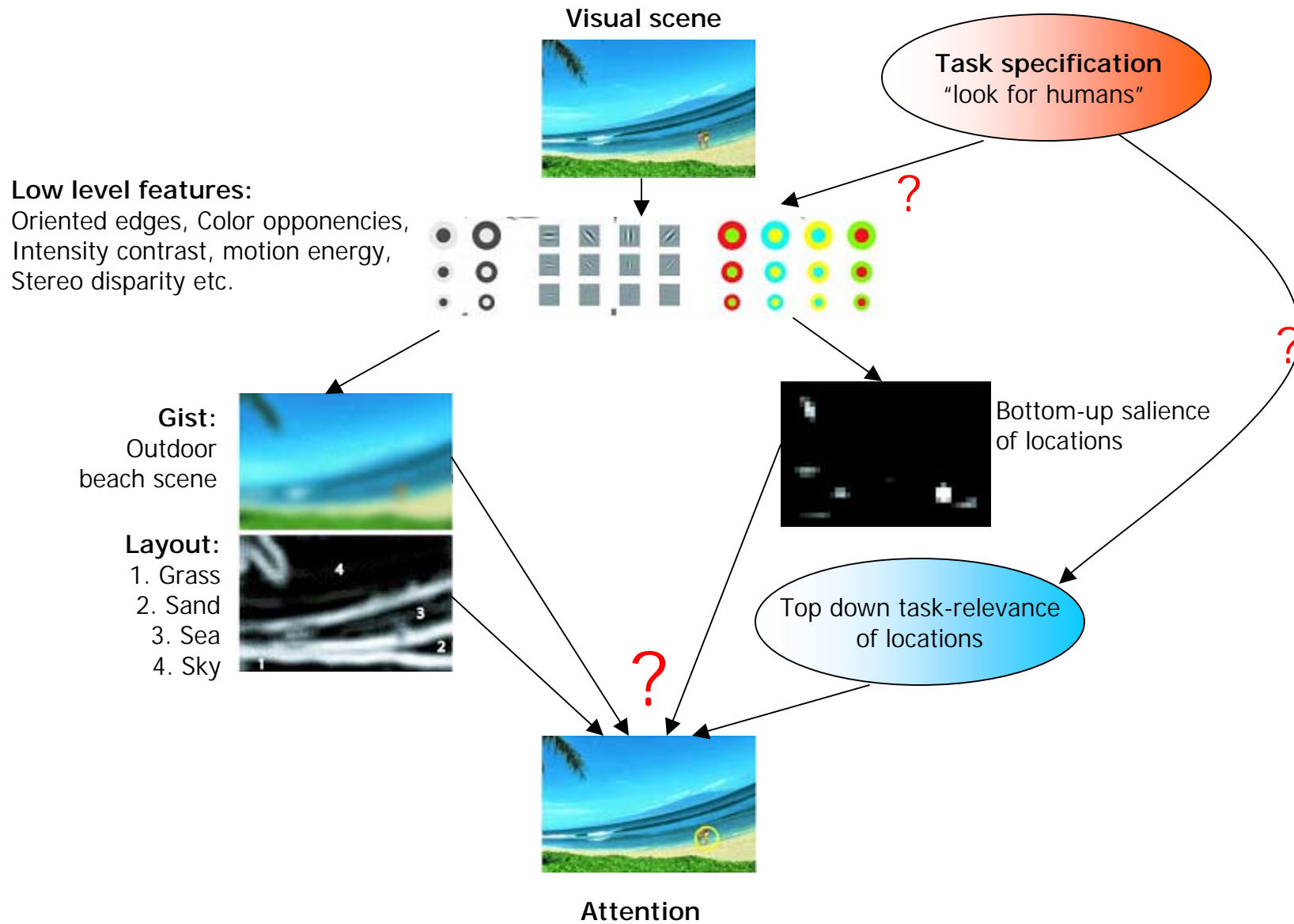


6

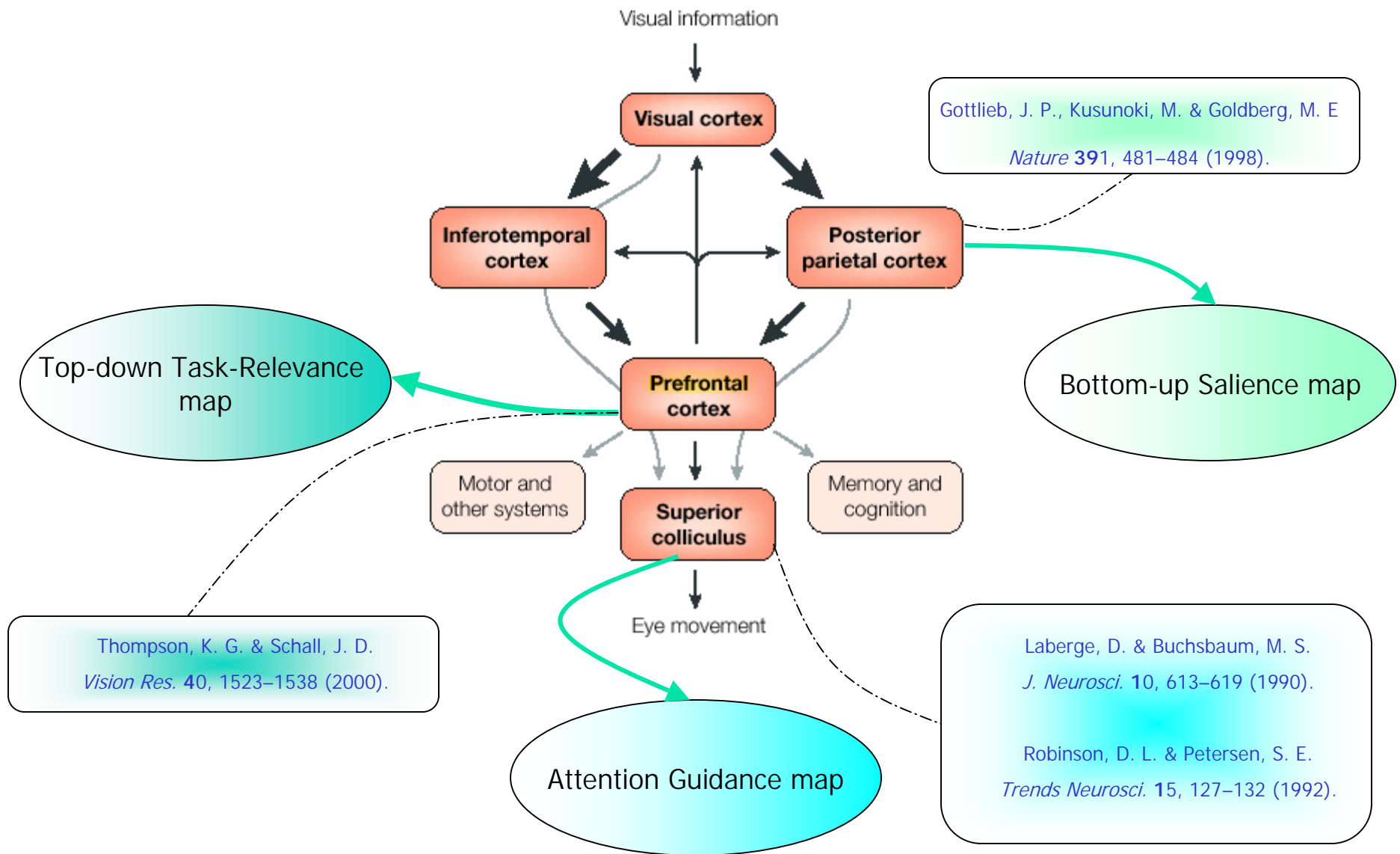


7

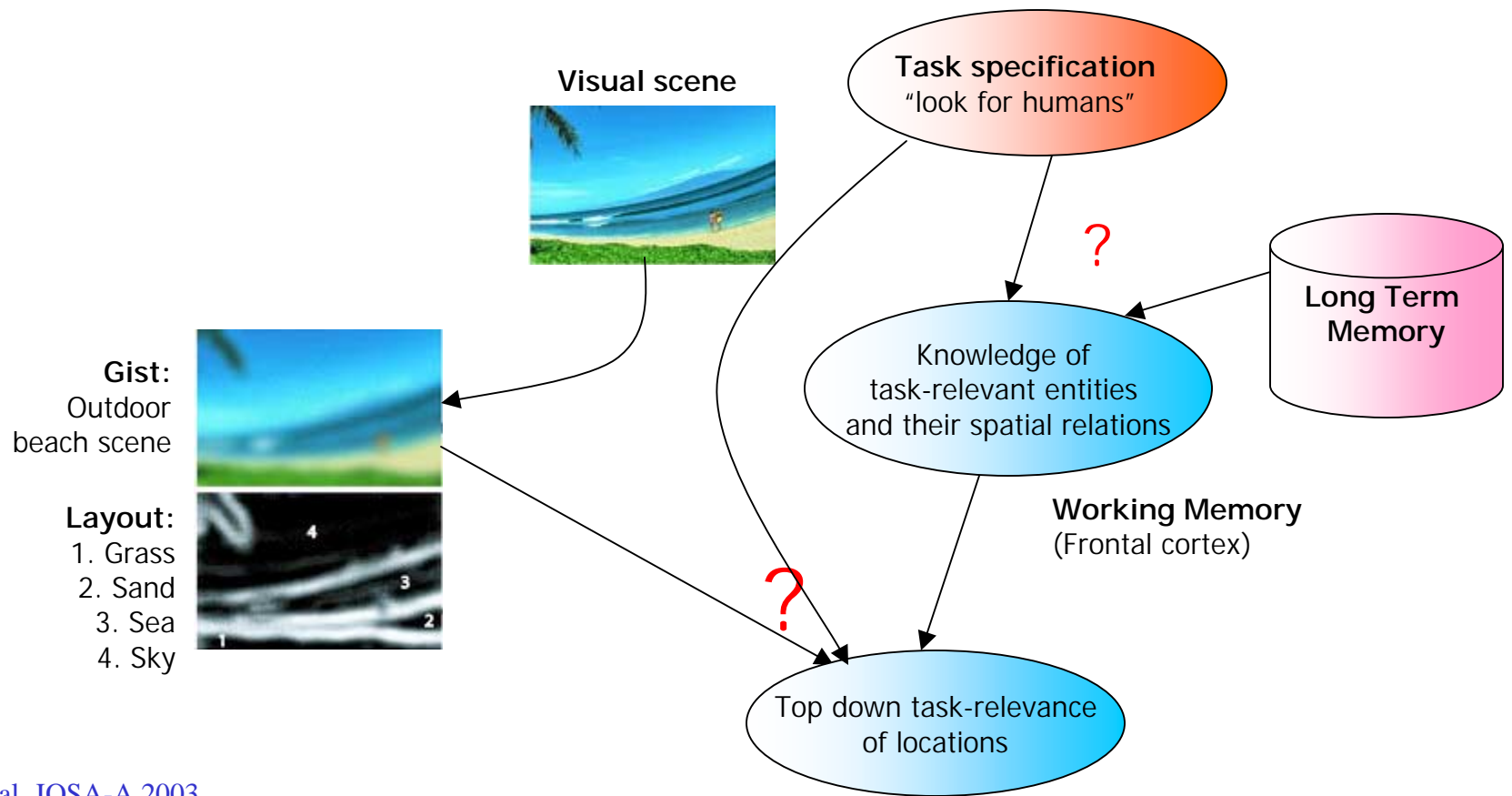
How does task influence attention?



How may task and salience interface?



Towards modeling the influence of task on relevance



Components of scene understanding model

- Question/task, e.g., “who is doing what to whom?”
- Lexical parser to extract key concepts from question
- Ontology of world concepts and their inter-relationships, to expand concepts explicitly looked for to related ones
- Attention/recognition/gist+layout visual subsystems to locate candidate relevant objects/actors/actions
- Working memory of concepts relevant to current task
- Spatial map of locations relevant to current task

Towards a computational model

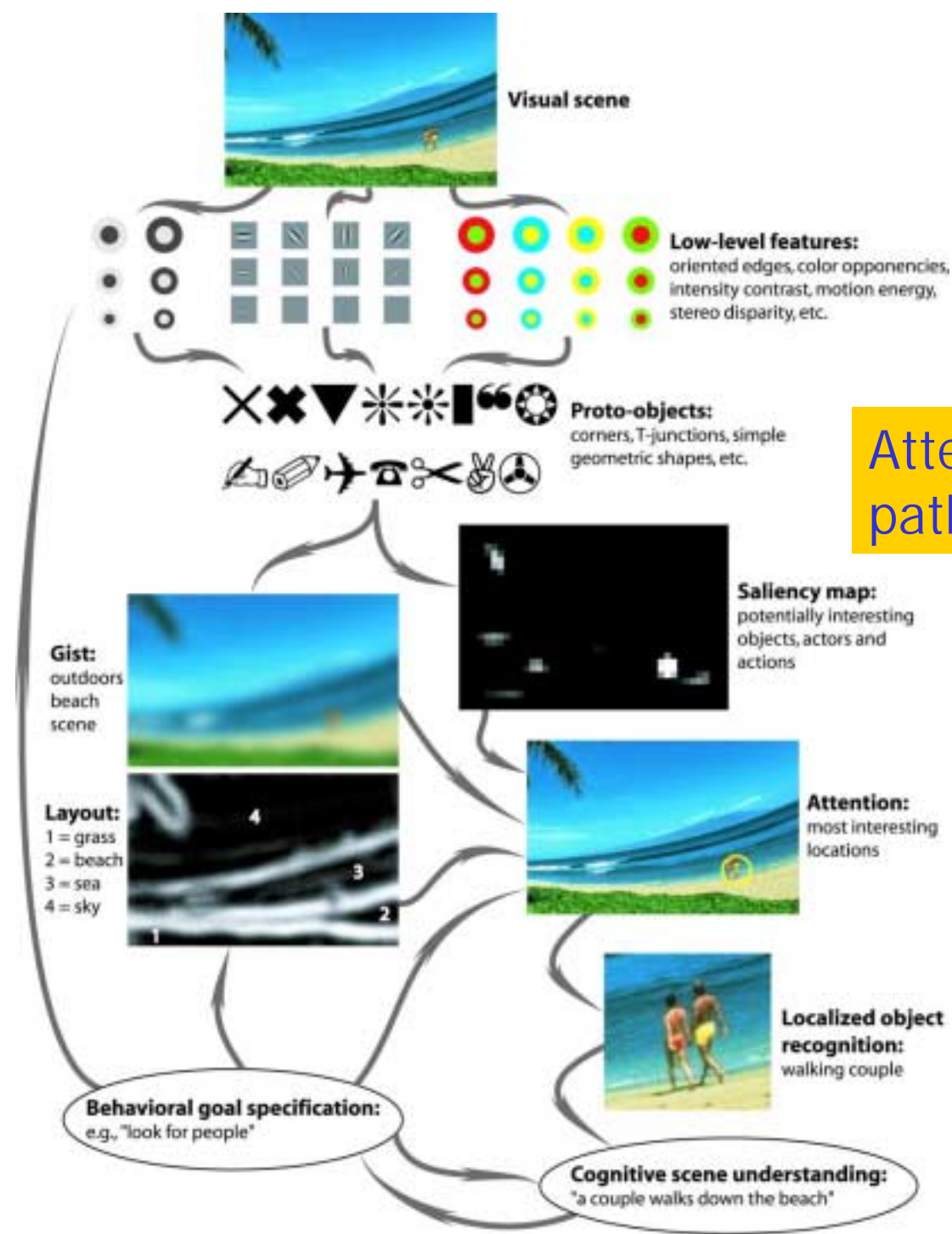
- Consider the following scene (next slide)
- Let's walk through a schematic (partly hypothetical, partly implemented) diagram of the sequence of steps that may be triggered during its analysis.



Two streams

- Not where/what...
- But attentional/non-attentional
 - Attentional: local analysis of details of various objects
 - Non-attentional: rapid global analysis yields coarse identification of the setting (rough semantic category for the scene, e.g., indoors vs. outdoors, rough layout, etc)

Setting pathway

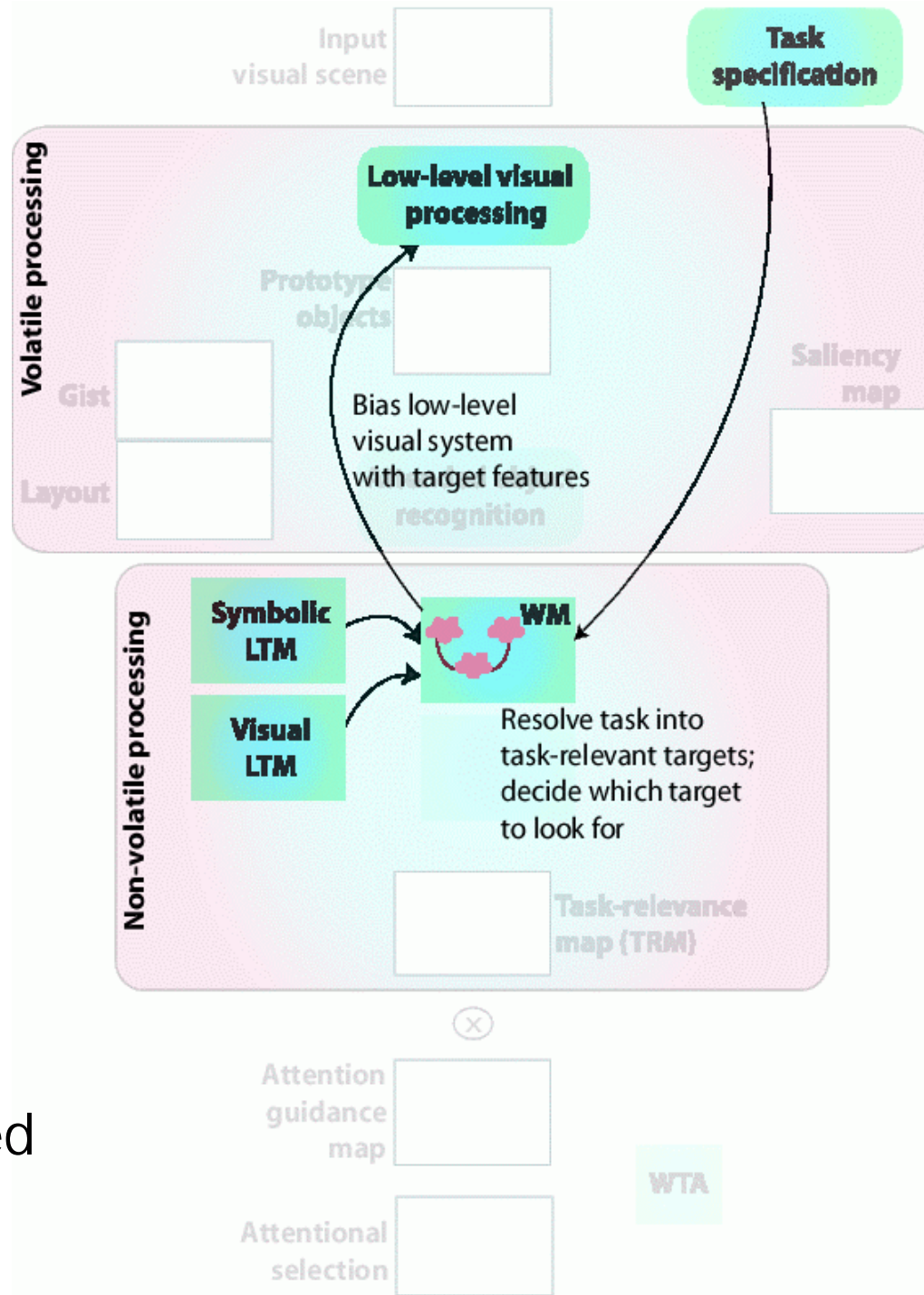


Attentional pathway

Itti 2002, also see Rensink, 2000

Step 1: eyes closed

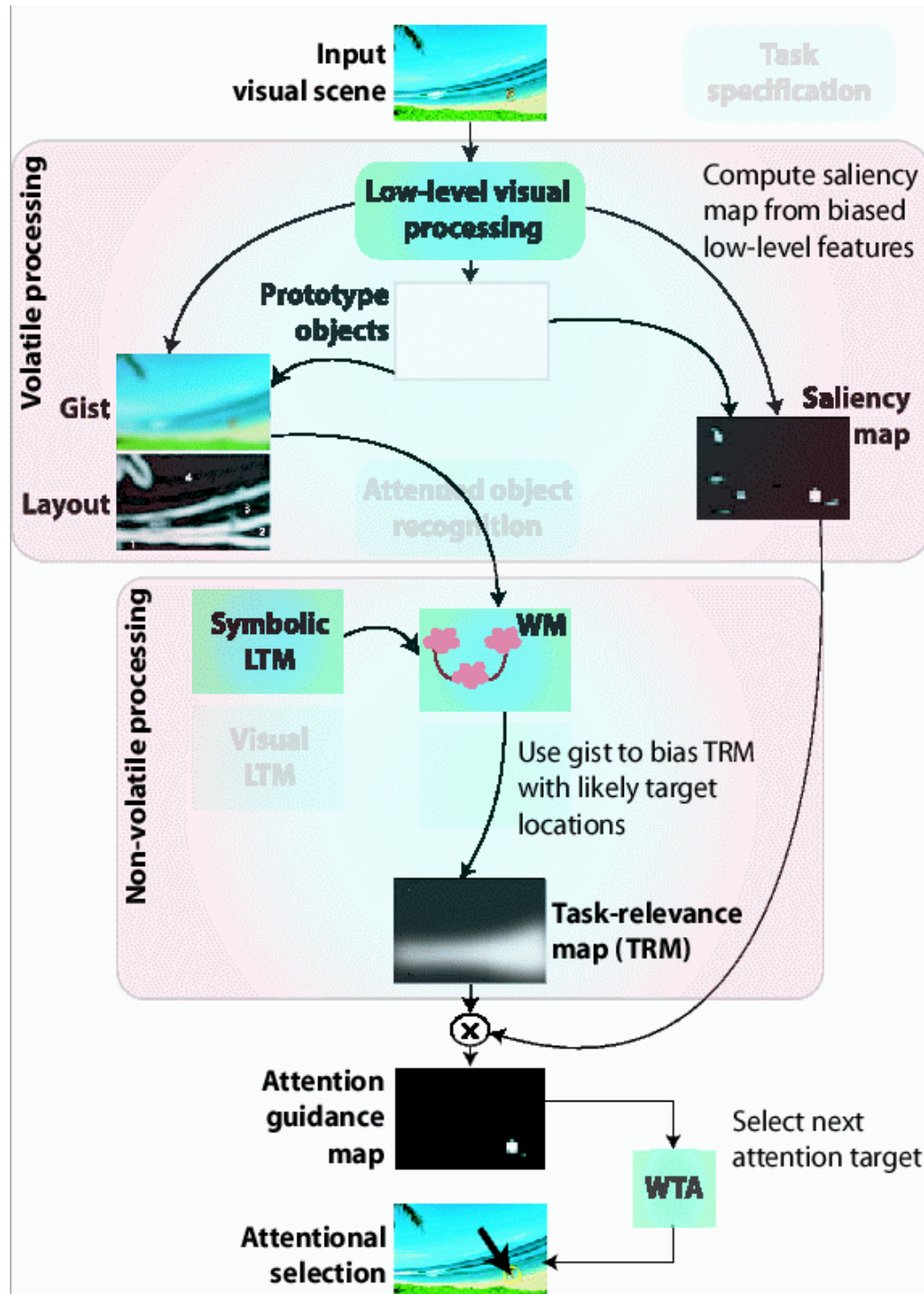
- Given a task, determine objects that may be relevant to it, using symbolic LTM (long-term memory), and store collection of relevant objects in symbolic WM (working memory).
 - E.g., if task is to find a stapler, symbolic LTM may inform us that a desk is relevant.
- Then, prime visual system for the features of the most-relevant entity, as stored in visual LTM.
 - E.g., if most relevant entity is a red object, boost red-selective neurons.
 - C.f. guided search, top-down attentional modulation of early vision.



1. Eyes closed

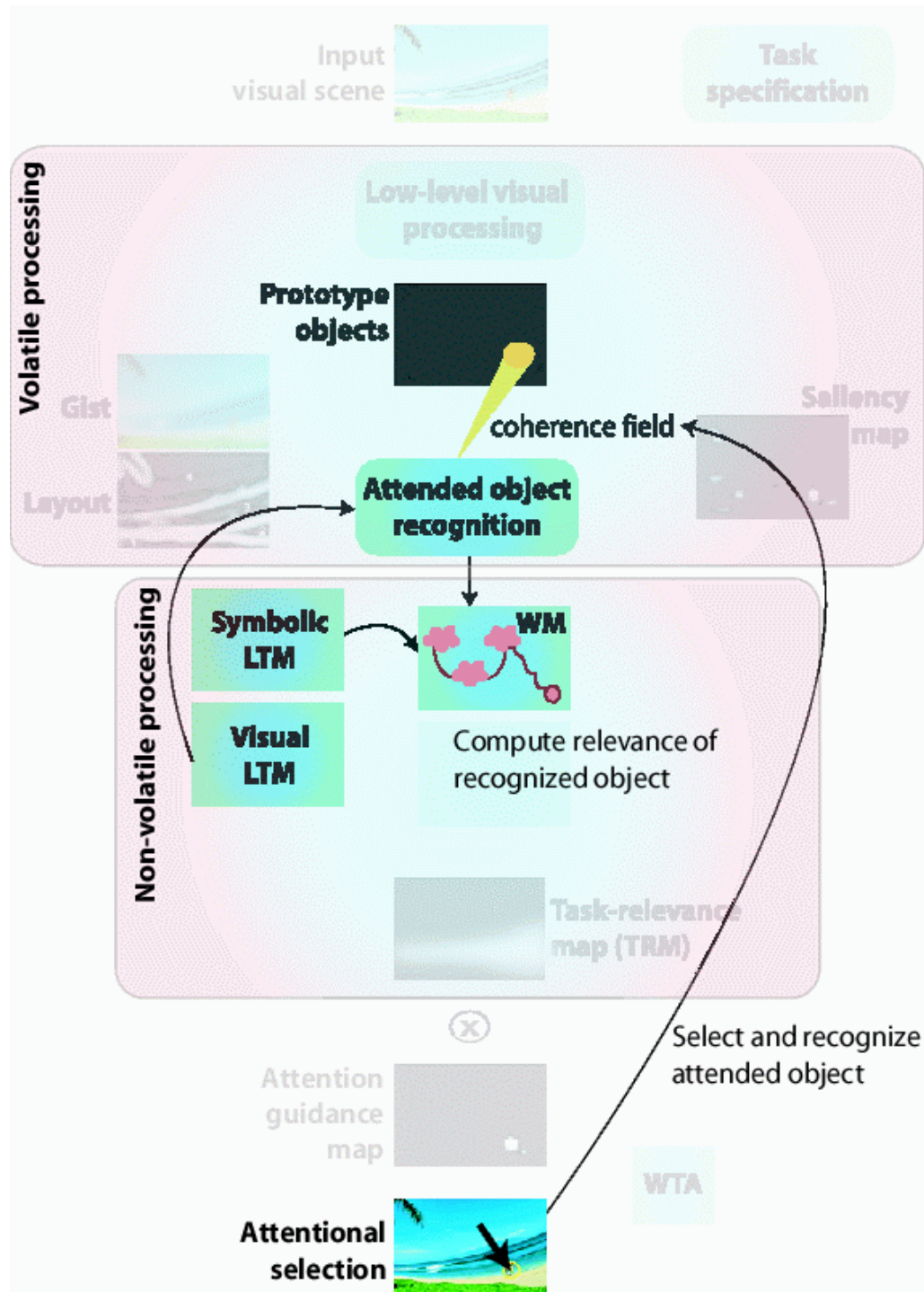
Step 2: attend

- The biased visual system yields a saliency map (biased for features of most relevant entity)
 - See Itti & Koch, 1998-2003, Navalpakkam & Itti, 2003
- The setting yields a spatial prior of where this entity may be, based on very rapid and very coarse global scene analysis; here we use this prior as an initializer for our “[task-relevance map[®]](#)”, a spatial pointwise filter that will be applied to the saliency map
 - E.g., if scene is a beach and looking for humans, look around where the sand is, not in the sky!
 - See Torralba, 2003 for computer implementation.



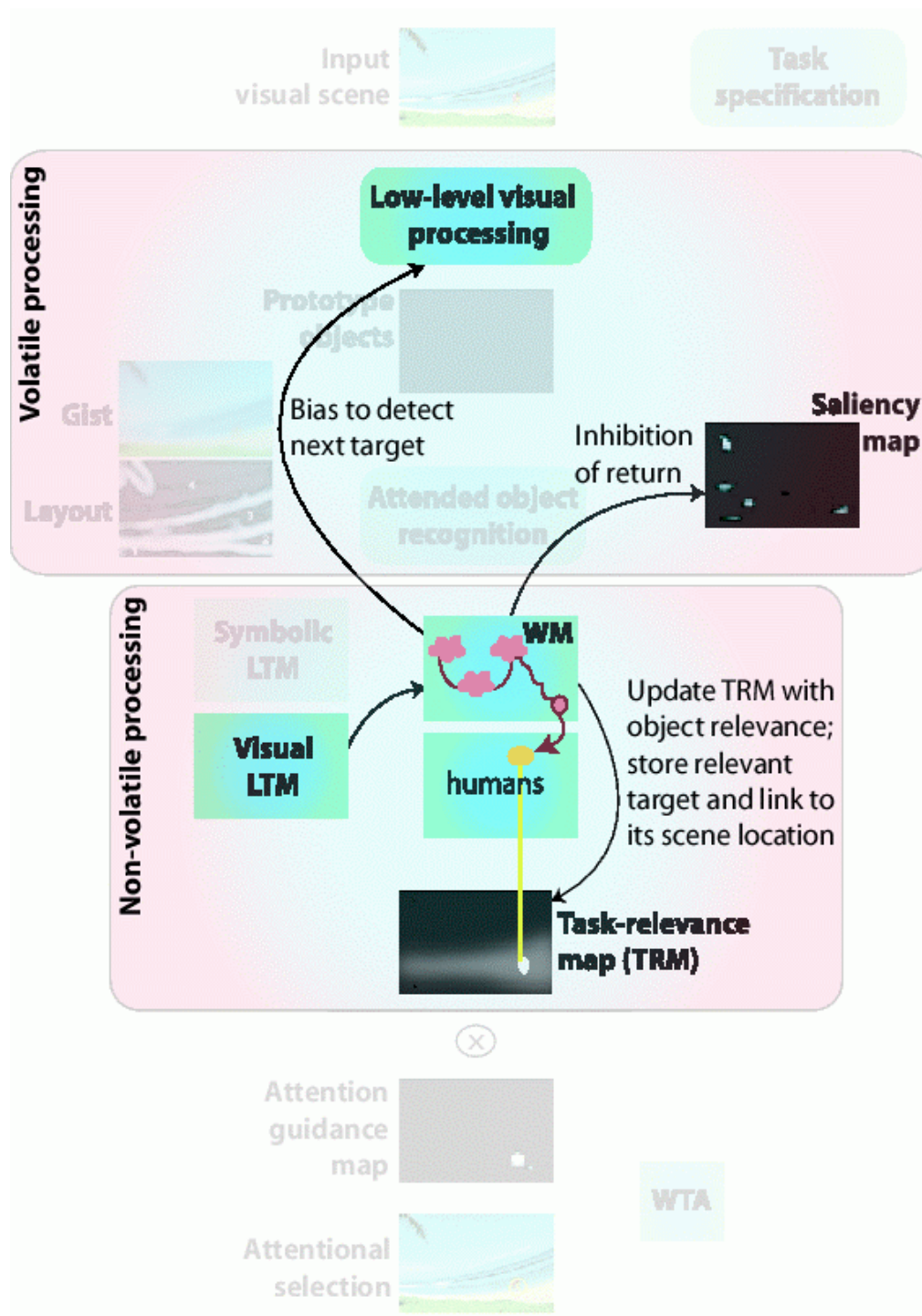
2. Attend

3. Recognize



4. Update

- As an entity is recognized, its relationships to other entities in the WM are evaluated, and the relevance of all WM entities is updated.
- The task-relevance map (TRM) is also updated with the computed relevance of the currently-fixated entity. That will ensure that we will later come back regularly to that location, if relevant, or largely ignore it, if irrelevant.



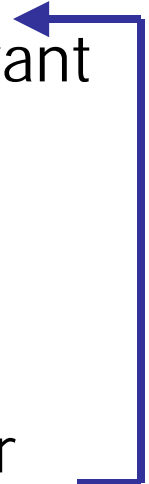
4. Update

Iterate

- The system keeps looping through steps 2-4
- The current WM and TRM are a first approximation to what may constitute the “Minimal subscene”:
 - A set of relevant spatial locations with attached object labels (see “object files”), and
 - A set of relevant symbolic entities with attached relevance values

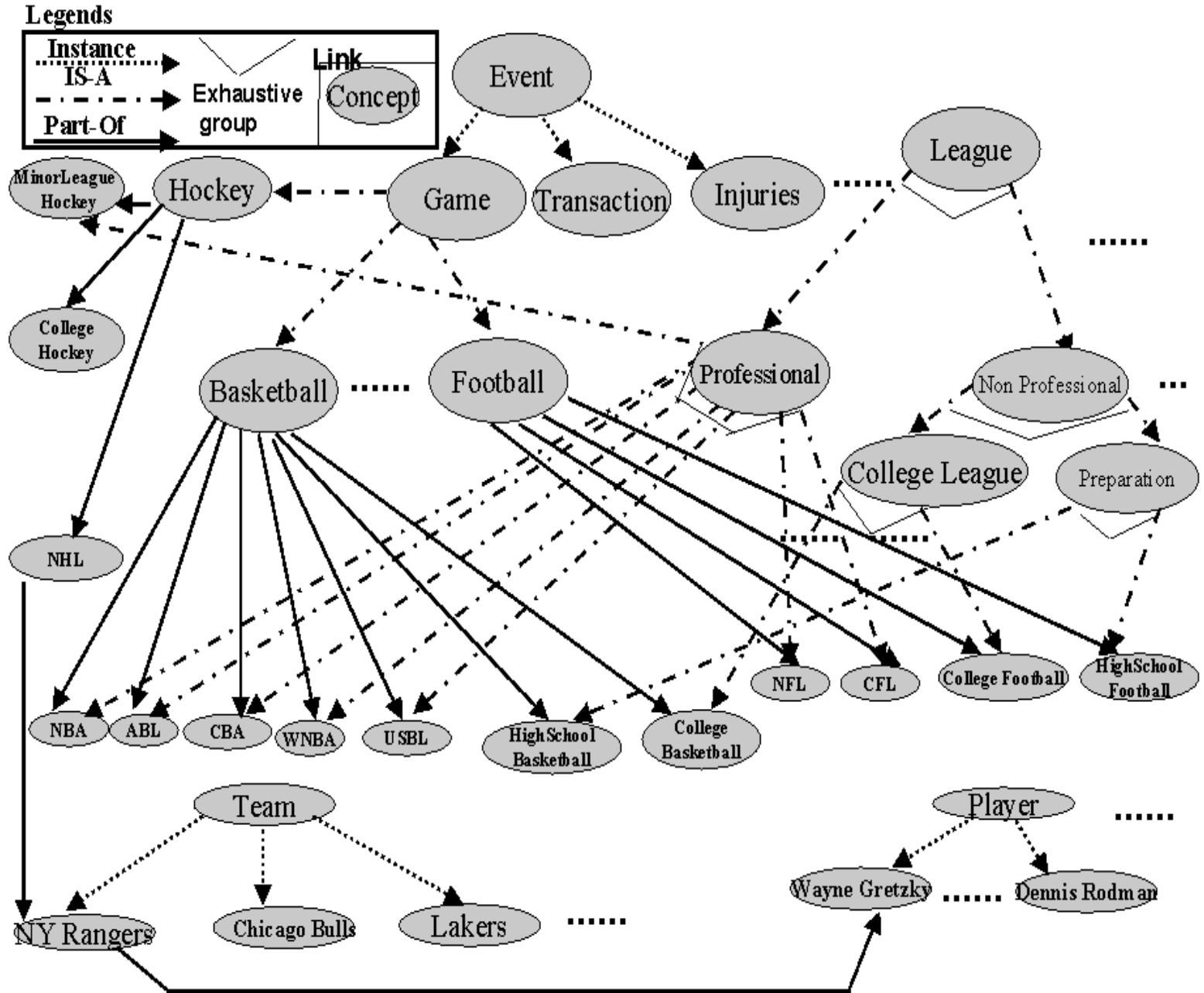
Prototype Implementation

Model operation

- Receive and parse task specification; extract concepts being looked for
 - Expand to wider collection of relevant concepts using ontology
 - Bias attention towards the visual features of most relevant concept
 - Attend to and recognize an object
 - If relevant, increase local activity in task map
 - Update working memory based on understanding so far
- 

After a while: task map contains only relevant regions, and attention primarily cycles through relevant objects

Ontology



Khan & McLeod, 2000

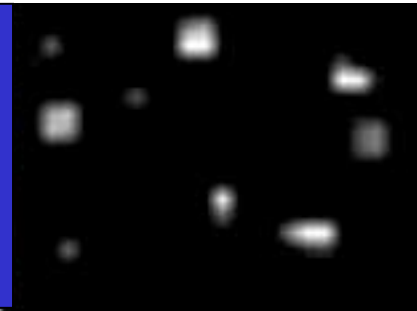
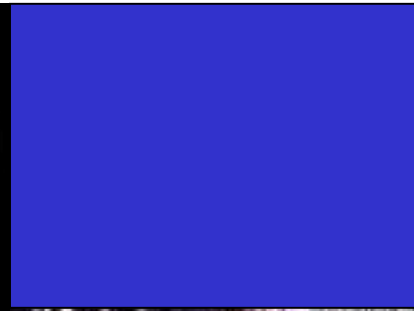
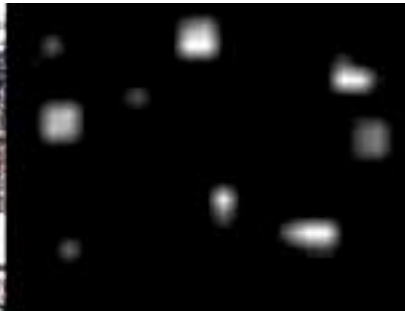
Frame

saliency map

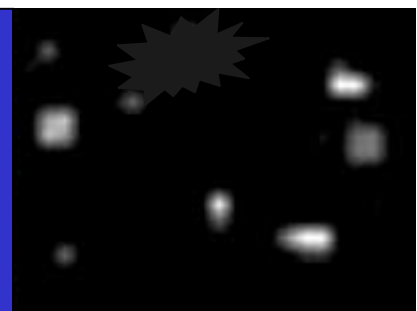
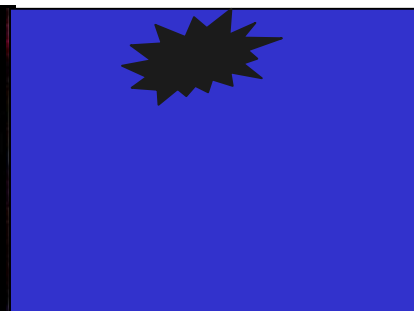
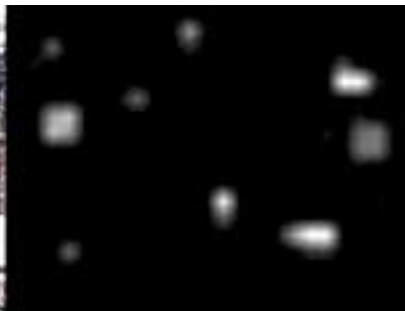
task map

pointwise product

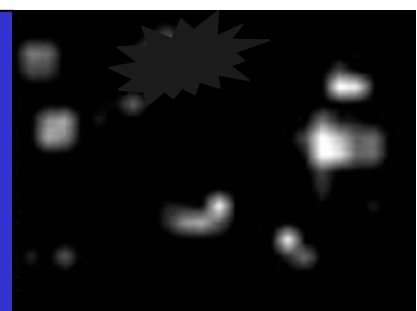
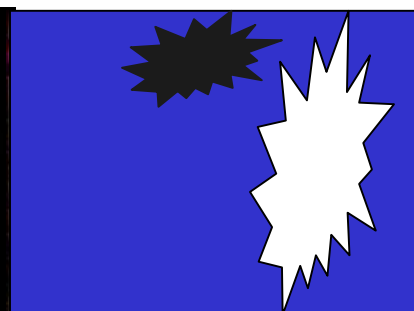
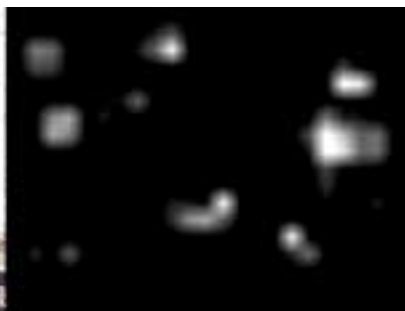
8



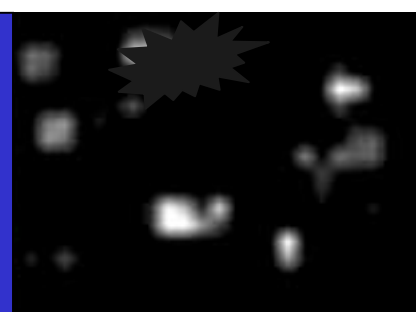
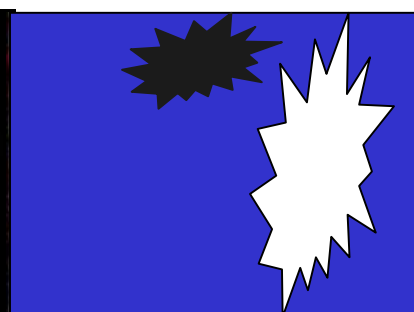
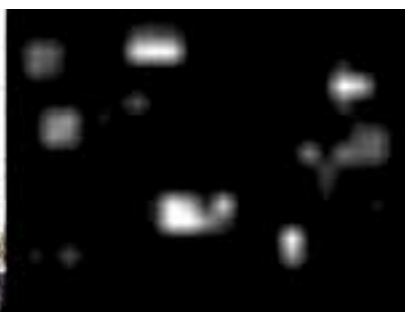
9



16

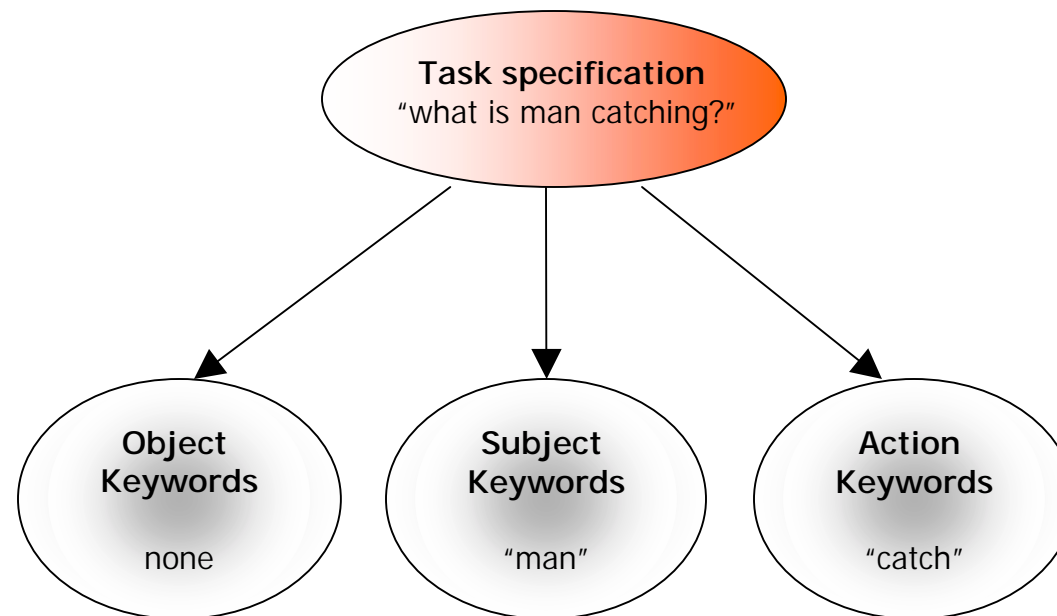


20

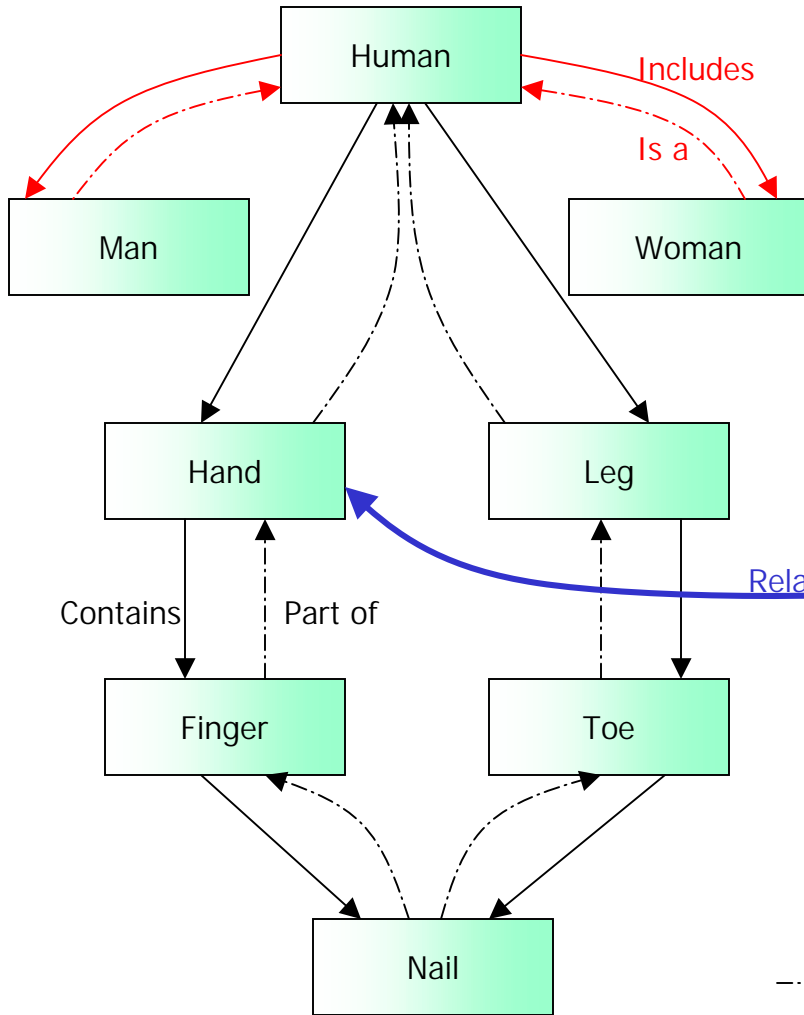


Task Specification

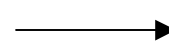
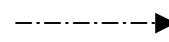
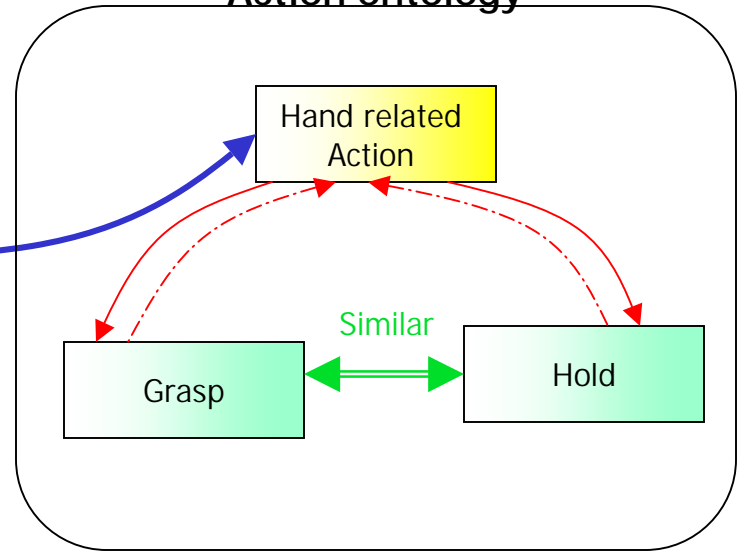
- Currently, we accept tasks such as “**who** is doing **what** to **whom**?”



Subject ontology



Action ontology



Part of

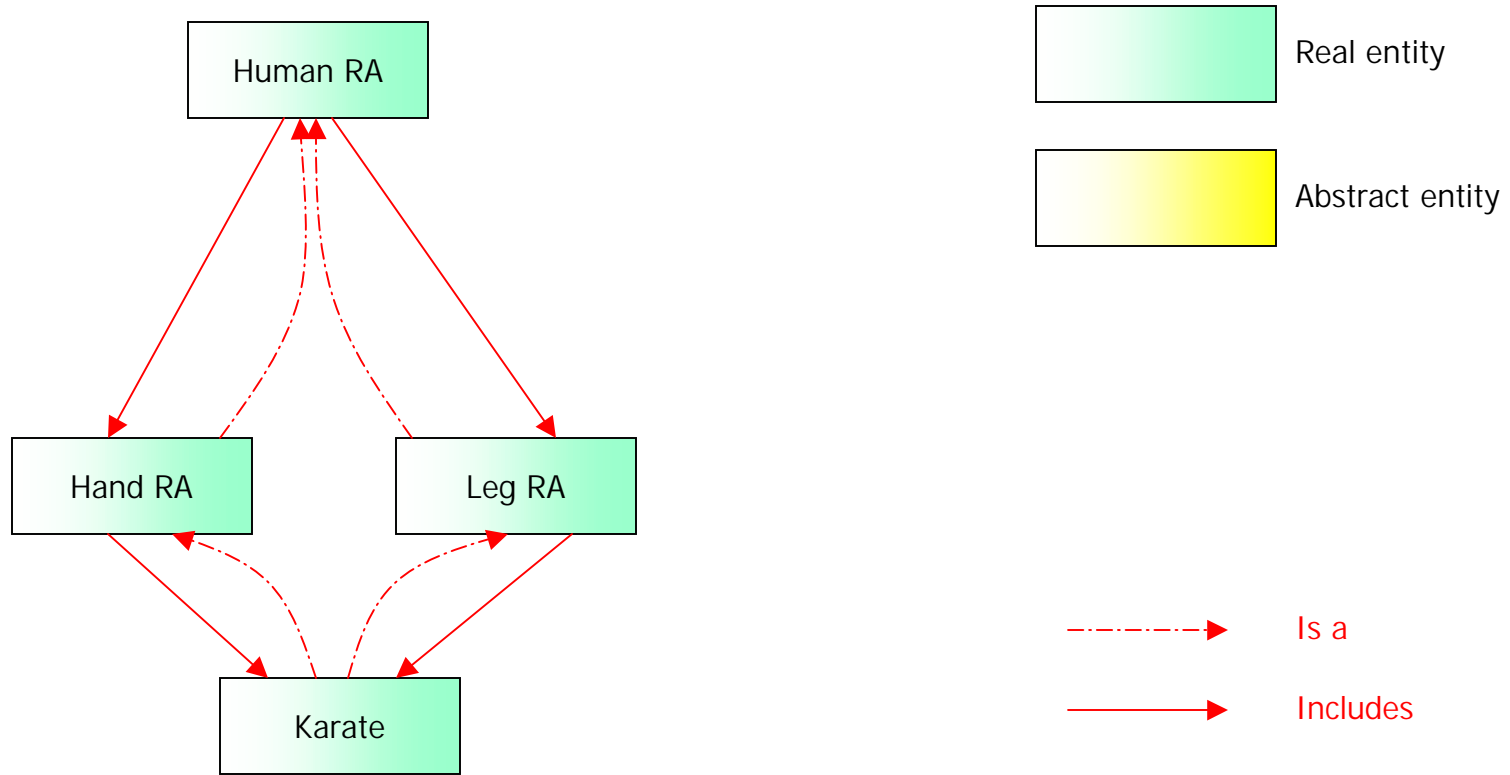
Contains



Is a

Includes

What to store in the nodes?



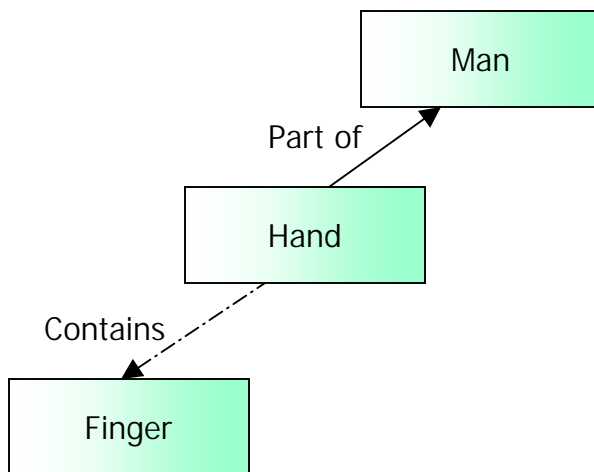
| | Leg RA | Leg RA | |
|---------|--------|--------|------------|
| Hand RA | 0.65 | 0.35 | Properties |
| Hand RA | 0 | 0 | |

Probability of occurrence

What to store in the edges?

Task: "find hand"

Suppose we find Finger and Man, what is more relevant?

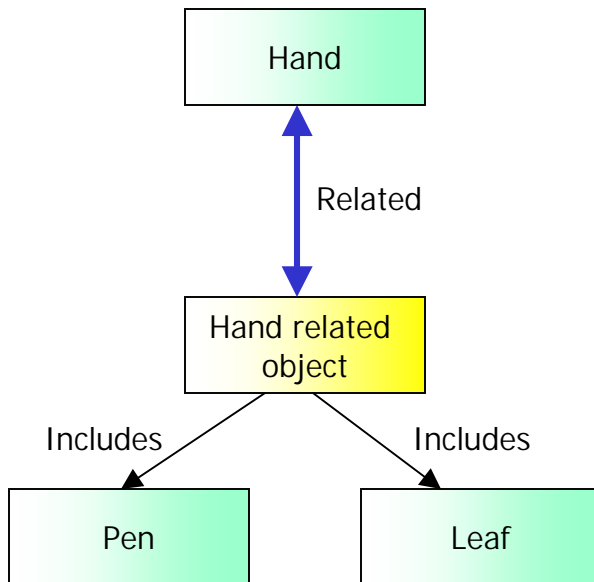


Granularity $g(u,v)$

$g((\text{Hand}, \text{Finger})) > g((\text{Hand}, \text{Man}))$

In general, $g(\text{contains}) > g(\text{part of})$
 $g(\text{includes}) > g(\text{is a})$
 $g(\text{similar}) = g(\text{related})$

Edge information



Co-occurrence(u,v)

Probability of joint occurrence of u and v

Task: "find hand"

Suppose we find Pen and Leaf, what is more relevant?

$P(\text{Pen is relevant} \mid \text{Hand is relevant})$

vs.

$P(\text{Leaf is relevant} \mid \text{Hand is relevant})$



$P(\text{Hand occurs} \mid \text{Pen occurs})$

vs.

$P(\text{Hand occurs} \mid \text{Leaf occurs})$



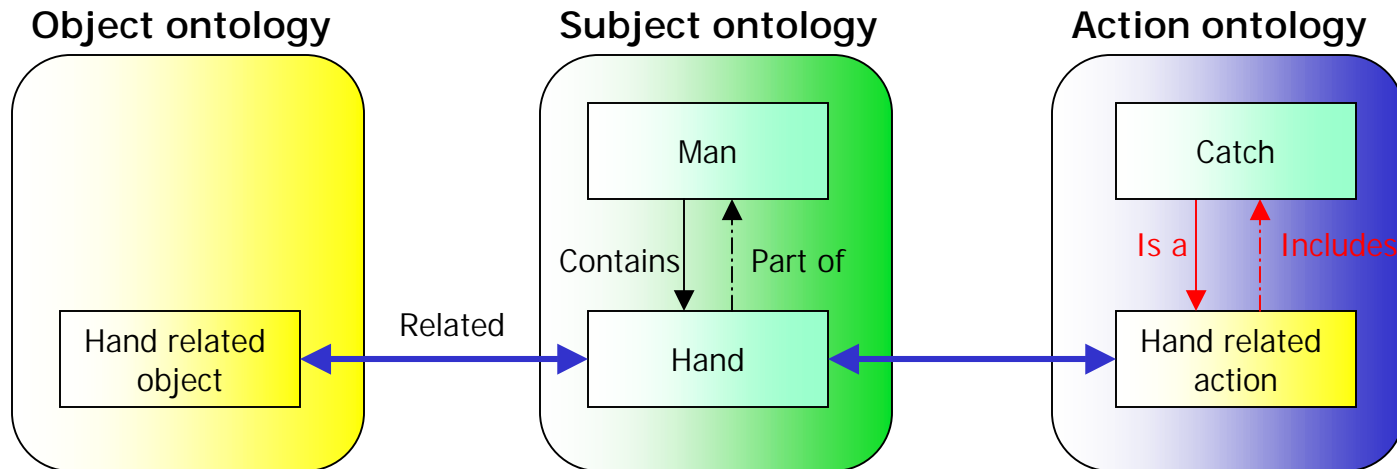
$P(\text{Hand, Pen} \mid \text{Pen})$

vs.

$P(\text{Hand, Leaf} \mid \text{Leaf})$

Working Memory and Task Graph

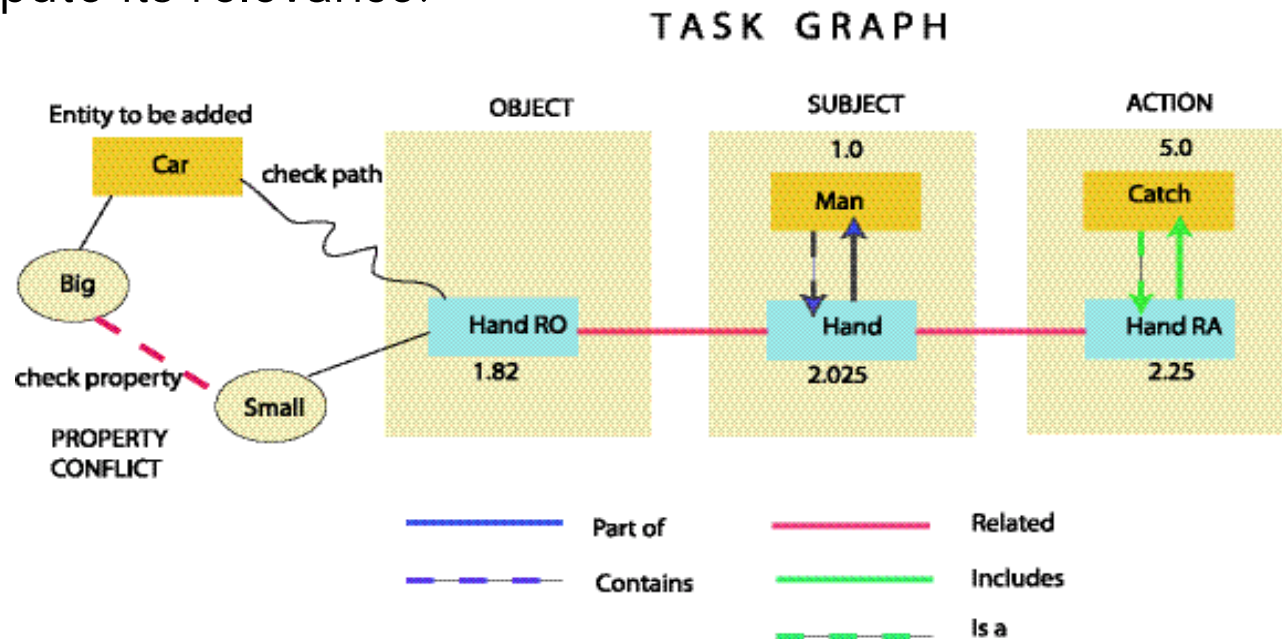
- Working memory creates and maintains the task graph
- Initial task graph is created using the task keywords and is expanded using “is a” and “related” relations.



Task: What is man catching?

Is the fixation entity relevant?

- **Test1:** Is there a path from fixation entity to task graph?
- **Test2:** Are the properties of fixation entity consistent with properties of task graph?
- If (**Test1 AND Test2**) then fixated entity is relevant
 - Add it to the task graph
 - compute its relevance.



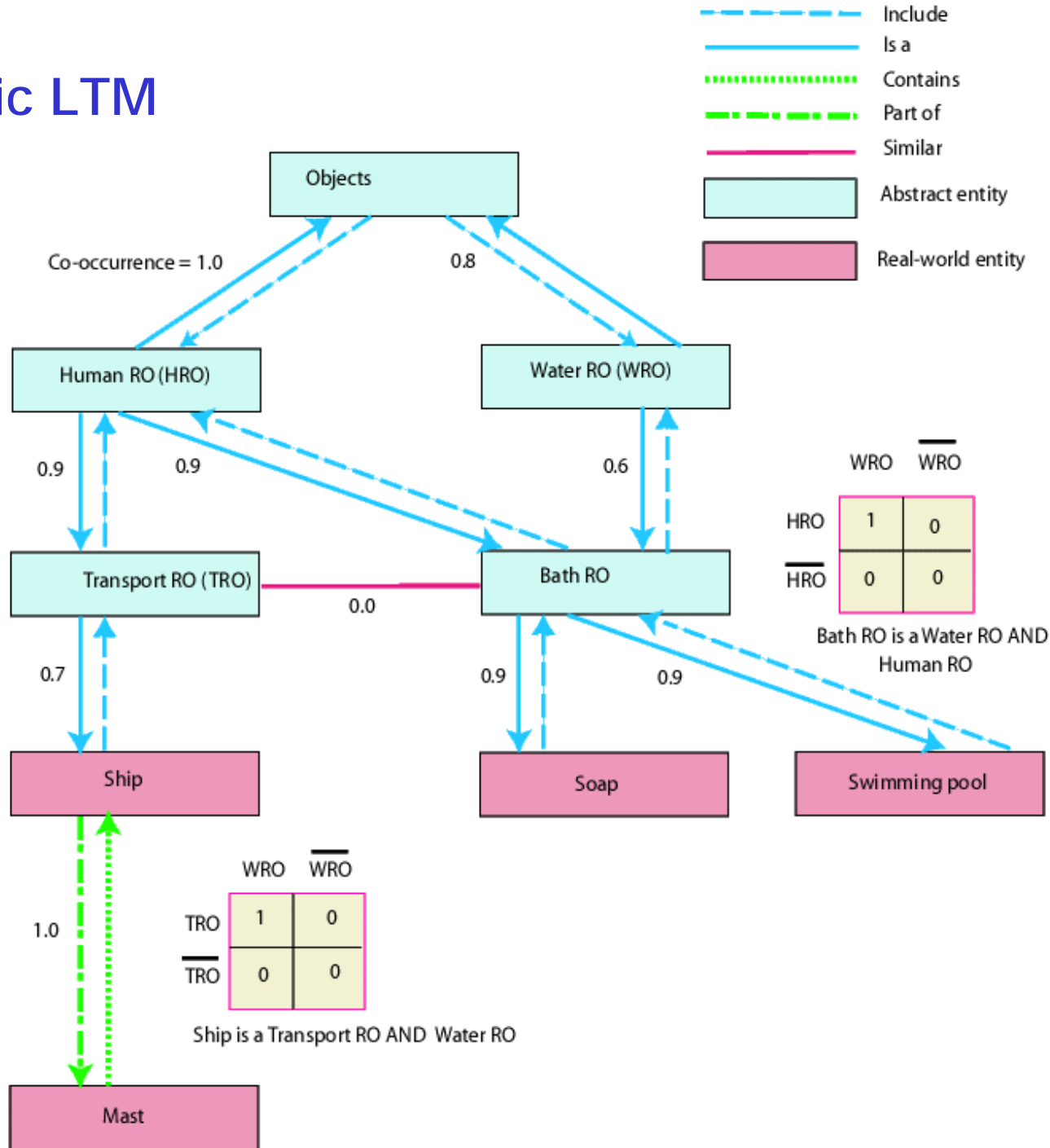
Computing Relevance

- Relevance of fixation entity depends on relevance of its neighbours and the connecting relations.
 - Consider the influence of u on relevance of fixation v
 - Depends on Relevance of u ---- R_u
 - Depends on granularity of edge(u,v) ---- $g((u,v))$
 - Depends on P(u occurs/ v occurs) ---- $c(u,v) / P(v)$
- mutual influence between 2 entities decreases as their distance increases (modelled by decay_factor where $0 < \text{decay_factor} < 1$)

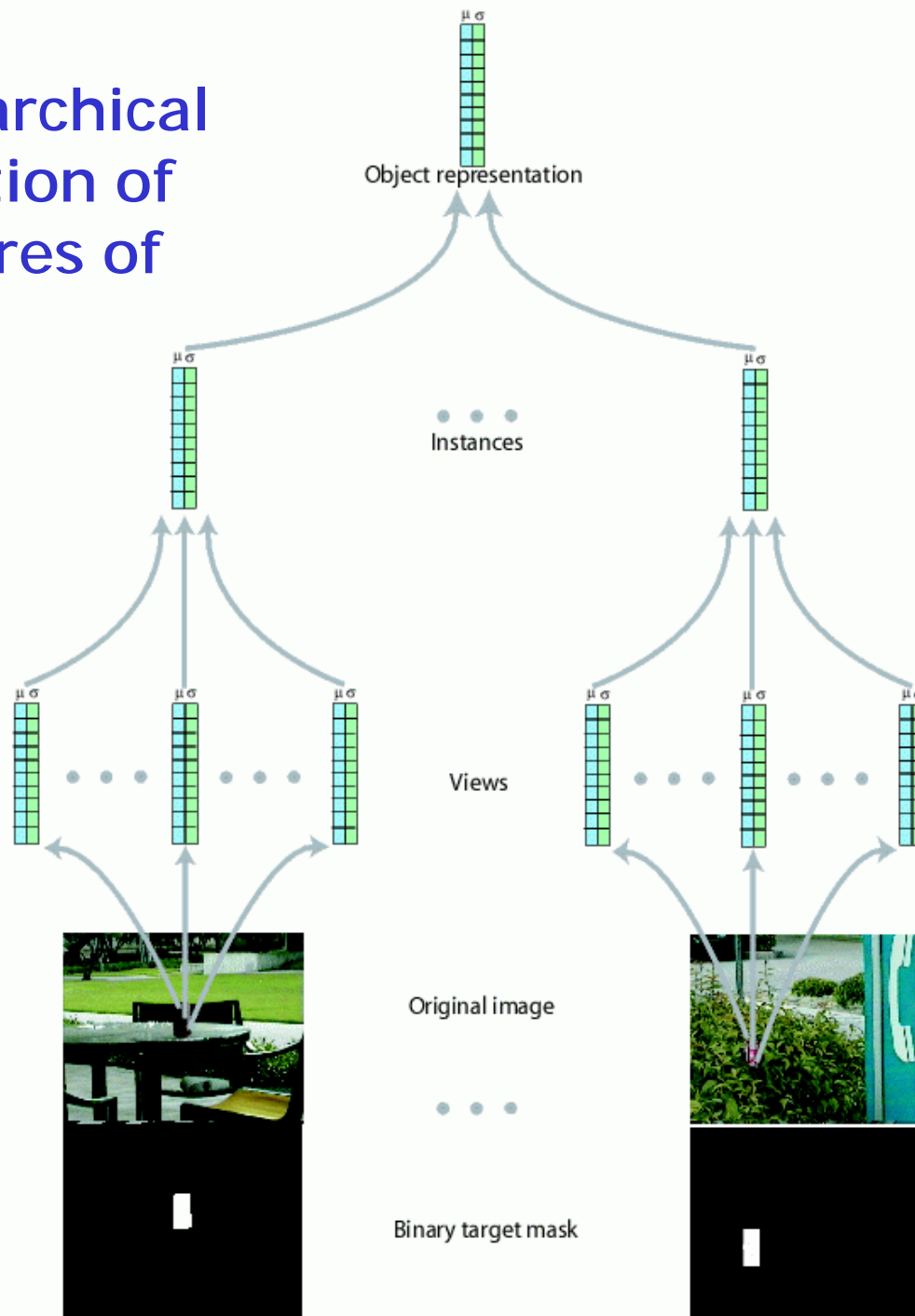
$$R_v = \max_{u: (u,v) \text{ is an edge}} (\text{Influence of u on v})$$

$$R_v = \max_{u: (u,v) \text{ is an edge}} (R_u * g(u,v) * c(u,v) / P(v) * \text{decay_factor})$$

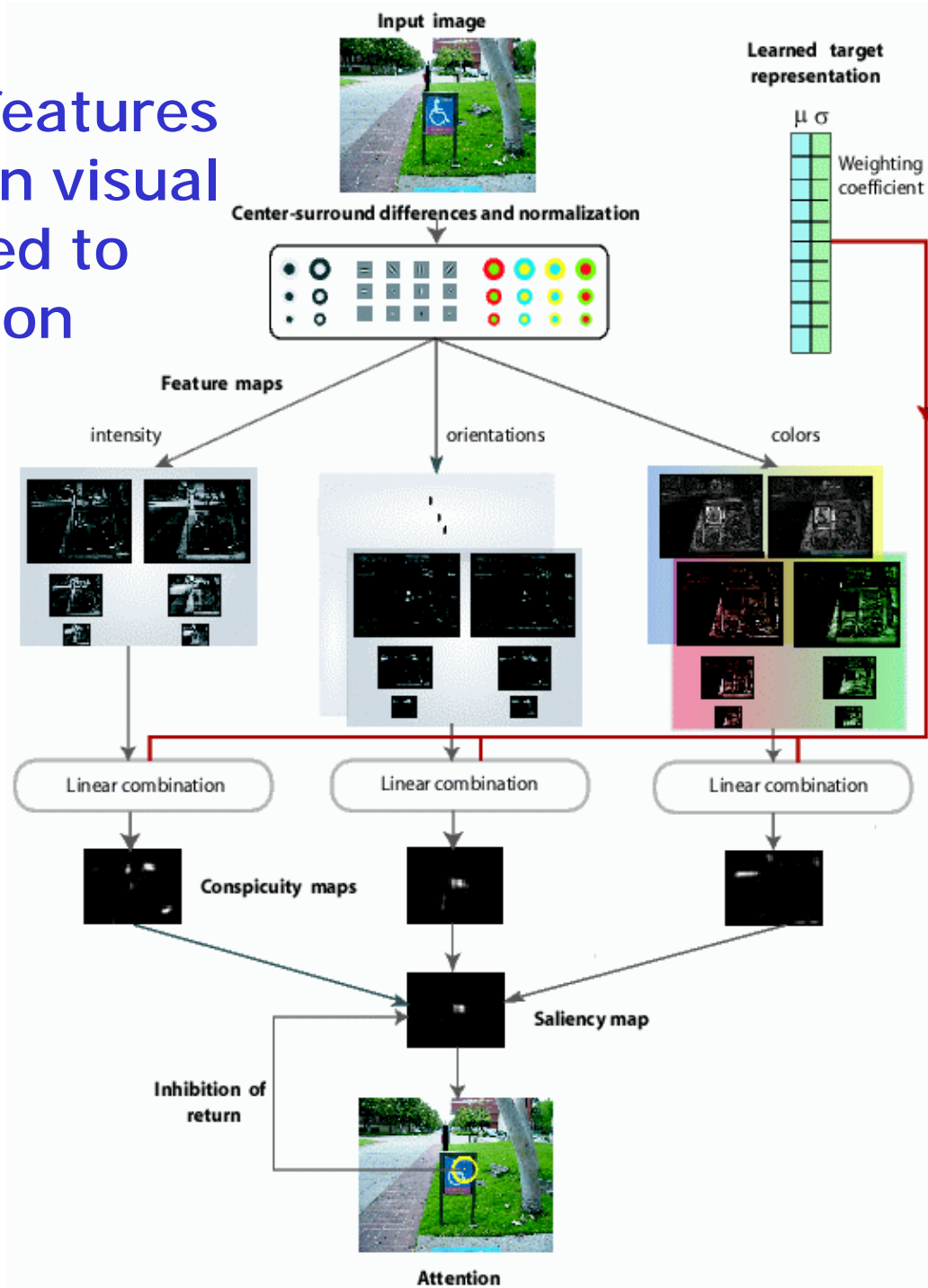
Symbolic LTM



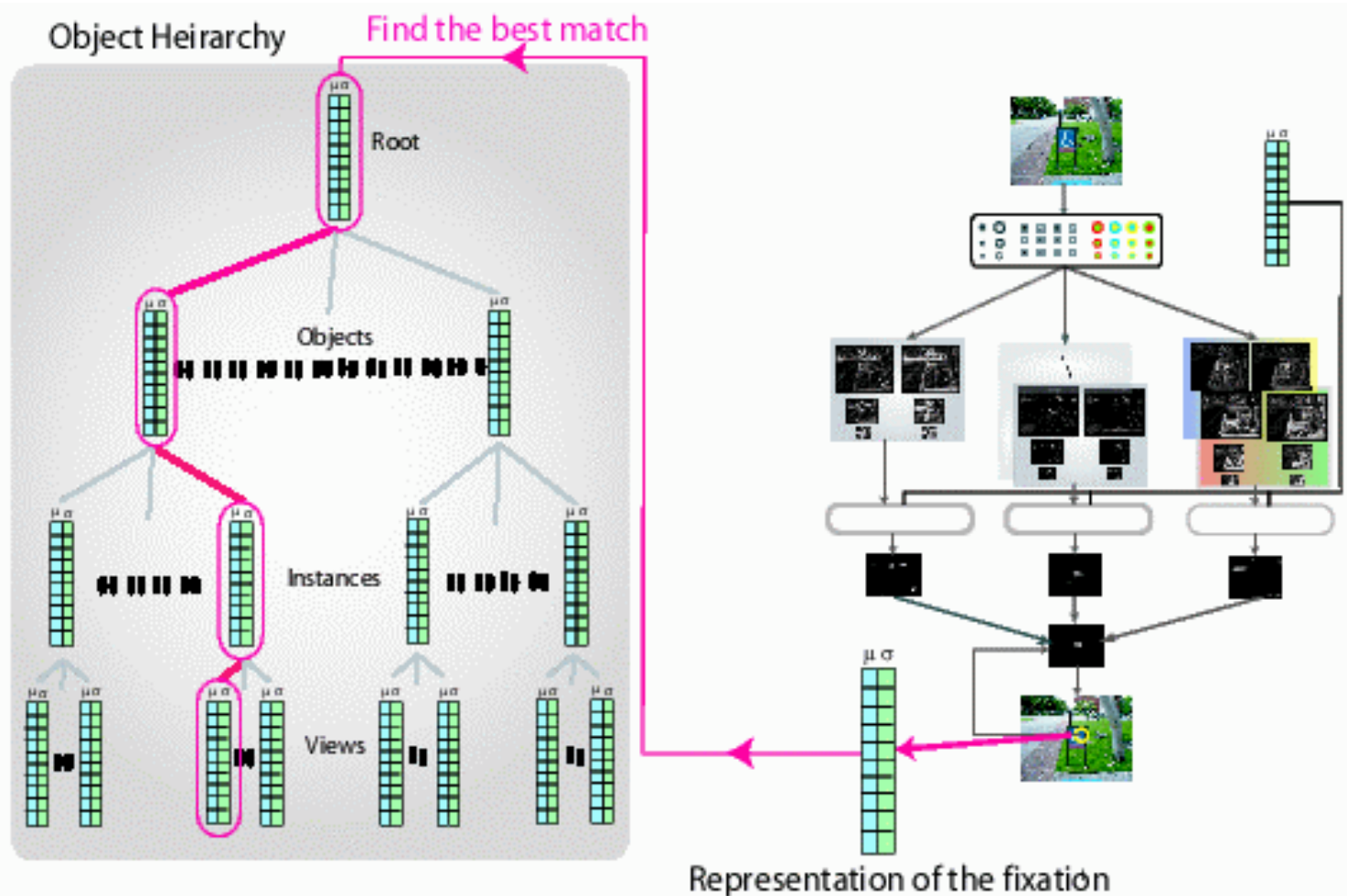
Simple hierarchical Representation of Visual features of Objects



The visual features
Of objects in visual
LTM are used to
Bias attention
Top-down



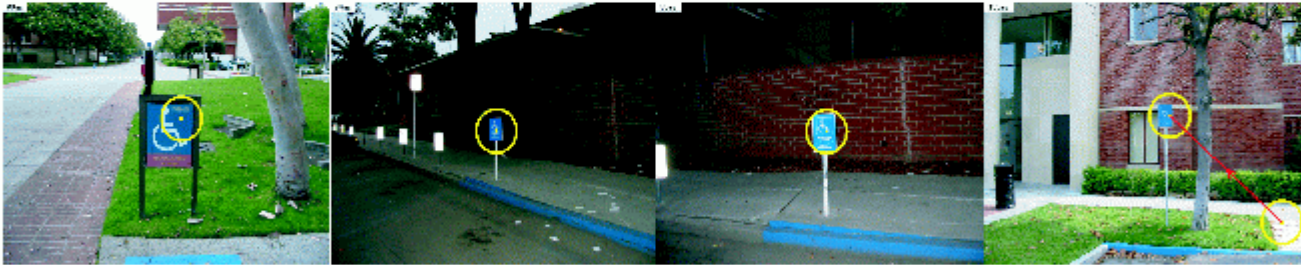
Once a location is attended to, its local visual features are matched to those in visual LTM, to recognize the attended object



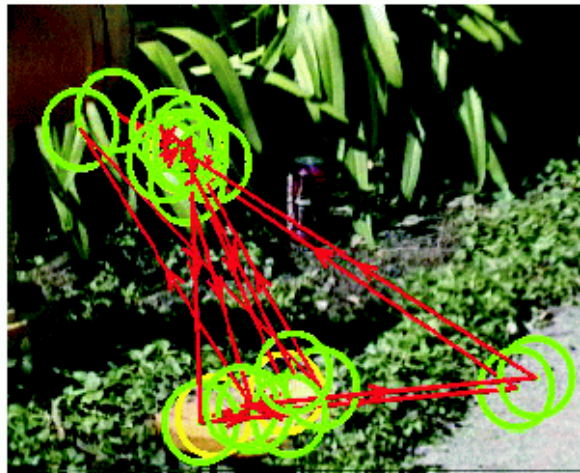
Training image



Learning object features
And using them for
biasing



Test image



Naïve: Looking for
Salient objects



Biased: Looking for
a Coca-cola can

| Target | Distractor | 95% <i>conf_{salience}</i> | 95% <i>conf_{time}</i> | 95% <i>conf_{shifts}</i> | Hypothesis | Remarks |
|--------|------------|------------------------------------|--------------------------------|----------------------------------|------------|--|
| | | [0.78, 2.20] | [0.00, 14.87] | [0.86, 1.06] | H_0 | Biasing improves detection time |
| | | [1.21, 3.84] | [0.72, 1.25] | [0.70, 1.09] | H_0 | Biasing does not affect detection time |
| | | [0.13, 0.32] | [0.10, 0.20] | [0.08, 0.16] | H_1 | Biasing increases detection time |
| | | [1.11, 1.40] | [1.02, 1.04] | [1.00, 1.00] | H_1 | Pop-out |
| | | [1.06, 2.30] | [1.76, 3.06] | [1.91, 3.88] | H_0 | Biasing improves detection time |
| | | [0.26, 1.00] | [0.09, 0.21] | [0.18, 0.36] | H_1 | Biasing increases detection time |
| | | [0.90, 1.01] | [1.00, 1.00] | [1.00, 1.00] | H_1 | Pop-out |
| | | [17.73, 964.89] | [0.37, 1.11] | [0.73, 1.09] | H_2 | Biasing does not affect detection time |
| | | [1.00, 1.11] | [1.00, 1.00] | [1.00, 1.00] | H_1 | Pop-out |
| | | [0.00, 3319.3] | [7.85, 23.26] | [11.61, 19.18] | H_0 | Biasing improves detection time |
| | | [2.09, 8.72] | [4.47, 10.46] | [5.69, 13.30] | H_2 | Biasing improves detection time |
| | | [1.02, 1.19] | [1.03, 1.55] | [1.00, 1.00] | H_1 | Biasing improves detection time |
| | | [0.97, 1.18] | [1.00, 1.46] | [1.03, 1.77] | H_1 | Biasing improves detection time |
| | | [3032.20, 7060.60] | [16.02, 17.85] | [20.00, 20.00] | H_2 | Biasing improves detection time |
| | natural | [2.48, 23.79] | [0.49, 1.06] | [0.53, 1.15] | H_2 | Biasing does not affect detection time |
| | natural | [0.00, 15.13] | [0.47, 1.37] | [0.49, 1.53] | H_0 | Biasing does not affect detection time |
| | natural | [1.00, 4.39] | [1.07, 2.17] | [1.09, 2.66] | H_0 | Biasing improves detection time |
| | natural | [1.88, 2.77] | [1.74, 2.59] | [1.79, 2.39] | H_0 | Biasing improves detection time |

Exercising the model by requesting that it finds several objects



Figure 12: Sequential detection of multiple targets: The model initialized the working memory with the targets to be found and their relevance (handicap sign, relevance = 1; fire hydrant, relevance = 0.5). It biased for the most relevant target (in this case, the handicap sign), made a false detection, recognized the fixation (fire hydrant), updated the state in its working memory (recorded that it found the fire hydrant), and proceeded to detect the remaining target by repeating the above steps.

Example 1

- Task1: find the faces in the scene
- Task2: find what the people are eating

Original scene TRM after 5 fixations TRM after 20 fixations



Example 2

- Task1: find the cars in the scene
- Task2: find the buildings in the scene

Original scene TRM after 20 fixations Attention trajectory



Learning the TRM through sequences of attention and recognition

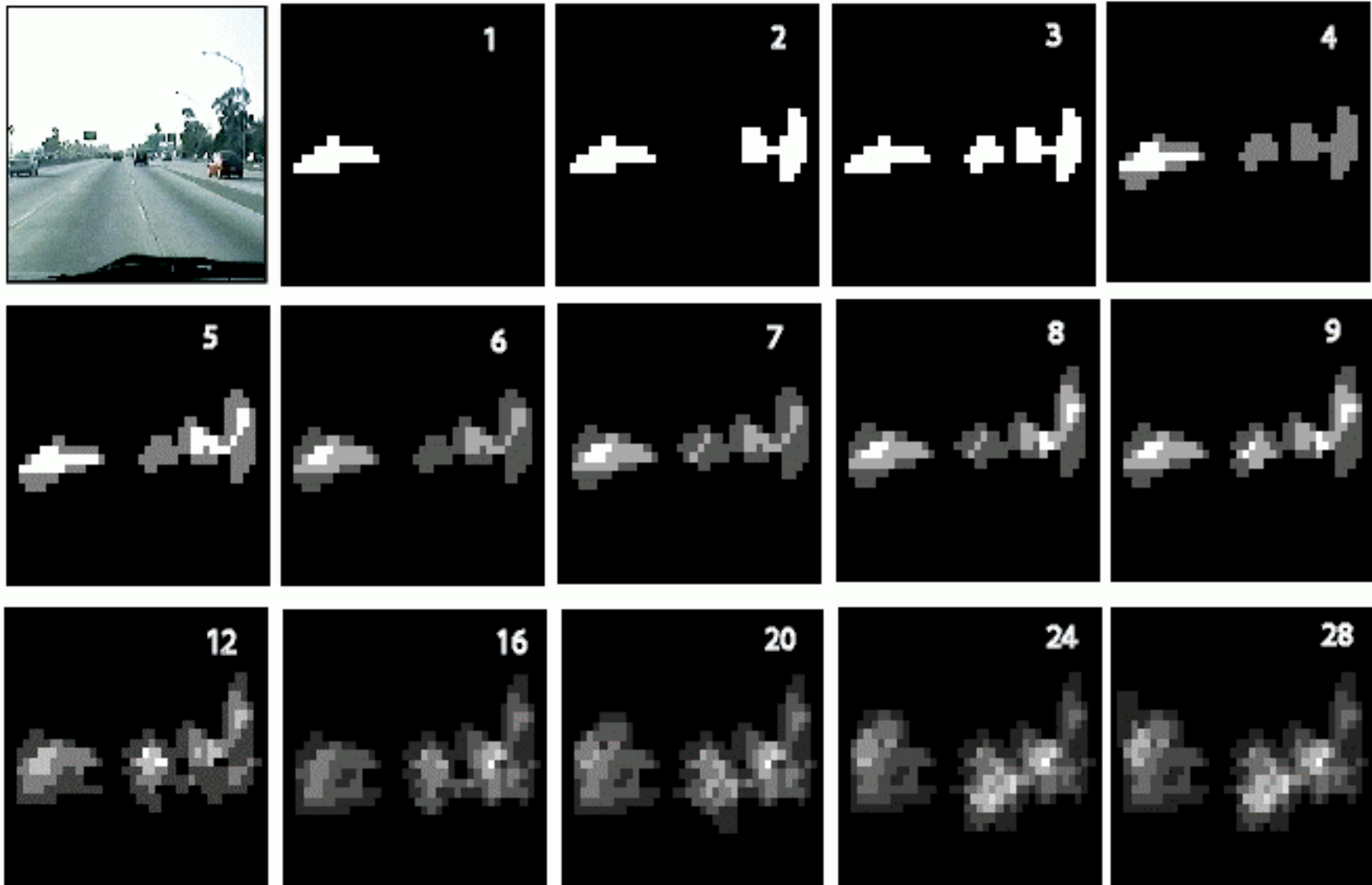


Figure 14: Learning the TRM: the model learned the TRM for a driving task by attending, estimating the relevance of attended scene locations and updating the TRM. The development of the TRM across 28 fixations is shown here. Note that the TRM does not change significantly after a while and is learned to a reasonable precision within the first 5-10 fixations.

Outlook

- **Open architecture** – model not in any way dedicated to a specific task, environment, knowledge base, etc. just like our brain probably has not evolved to allow us to drive cars.
- **Task-dependent learning** – In the TRM, the knowledge base, the object recognition system, etc., guided by an interaction between attention, recognition, and symbolic knowledge to evaluate the task-relevance of attended objects
- **Hybrid neuro/AI architecture** – Interplay between rapid/coarse learnable global analysis (gist), symbolic knowledge-based reasoning, and local/serial trainable attention and object recognition
- **Key new concepts:**
 - **Minimal subszene** – smallest task-dependent set of actors, objects and actions that concisely summarize scene contents
 - **Task-relevance map** – spatial map that helps focus computational resources on task-relevant scene portions