

Knowledge Representation



- Knowledge engineering: principles and pitfalls
- Ontologies
- Examples

Knowledge Engineer



- Populates KB with facts and relations
- Must study and understand domain to pick important objects and relationships
- **Main steps:**
 - Decide what to talk about
 - Decide on vocabulary of predicates, functions & constants
 - Encode general knowledge about domain
 - Encode description of specific problem instance
 - Pose queries to inference procedure and get answers

Knowledge engineering vs. programming



Knowledge Engineering

Programming

1. Choosing a logic
2. Building knowledge base
3. Implementing proof theory
4. Inferring new facts

Choosing programming language
Writing program
Choosing/writing compiler
Running program

Why knowledge engineering rather than programming?

Less work: just specify objects and relationships known to be true, but leave it to the inference engine to figure out how to solve a problem using the known facts.

Properties of good knowledge bases



- Expressive
- Concise
- Unambiguous
- Context-insensitive
- Effective
- Clear
- Correct
- ...

Trade-offs: e.g., sacrifice some correctness if it enhances brevity.

Efficiency



- **Ideally:** Not the knowledge engineer's problem

The inference procedure should obtain same answers no matter how knowledge is implemented.

- **In practice:**
 - use automated optimization
 - knowledge engineer should have some understanding of how inference is done

Pitfall: design KB for human readers

- KB should be designed primarily for inference procedure!
- e.g., *VeryLongName* predicates:

BearOfVerySmallBrain(Pooh) does not allow inference procedure to infer that Pooh is a bear, an animal, or that he has a very small brain, ...

Rather, use:

Bear(Pooh)

$\forall b, \text{Bear}(b) \Rightarrow \text{Animal}(b)$

$\forall a, \text{Animal}(a) \Rightarrow \text{PhysicalThing}(a)$

...

[See AIMA pp. 220-221 for full example]

Debugging

- In principle, easier than debugging a program,

because we can look at each logic sentence in isolation and tell whether it is correct.

Example:

$\forall x, \text{Animal}(x) \Rightarrow \exists b, \text{BrainOf}(x) = b$

means

“there is some object that is the value of the BrainOf function applied to an animal”

and can be corrected to mean

“every animal has a brain”

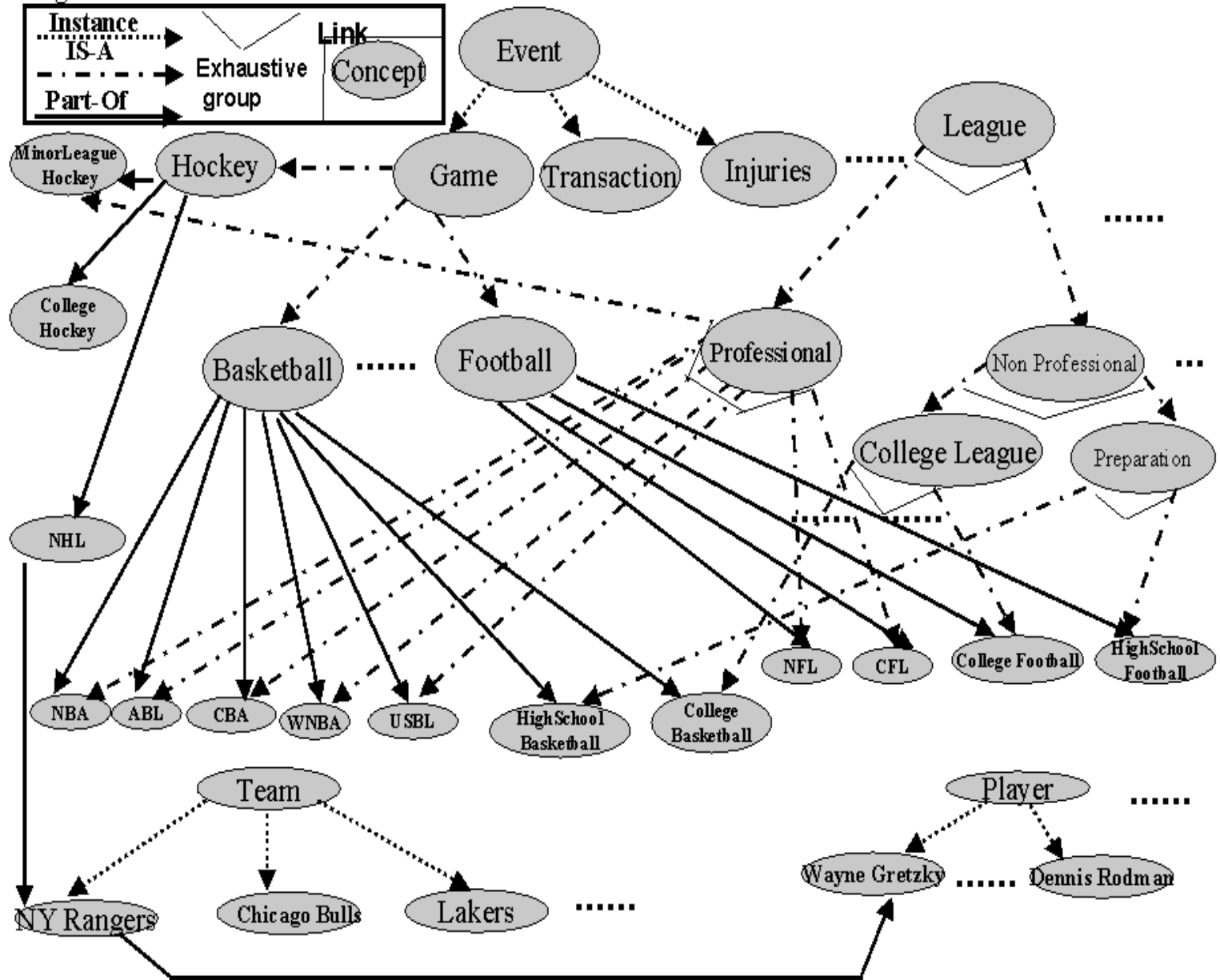
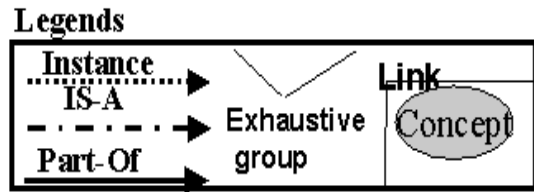
without looking at other sentences.

Ontology



- Collection of concepts and inter-relationships
- Widely used in the database community to “translate” queries and concepts from one database to another, so that multiple databases can be used conjointly (database federation)

Ontology Example



Khan & McLeod, 2000

Towards a general ontology



- Develop good representations for:
 - categories
 - measures
 - composite objects
 - time, space and change
 - events and processes
 - physical objects
 - substances
 - mental objects and beliefs
 - ...

Representing Categories

- We interact with individual objects, but...
much of reasoning takes place at the level of categories.
- **Representing categories in FOL:**
 - use **unary predicates**
e.g., $Tomato(x)$
 - **reification**: turn a predicate or function into an object
e.g., use constant symbol *Tomatoes* to refer to set of all tomatoes
"x is a tomato" expressed as " $x \in Tomatoes$ "
- Strong property of reification: can make assertions about reified category itself rather than its members
e.g., $Population(Humans) = 5e9$

Categories: inheritance



- Allow to organize and simplify knowledge base

e.g., if all members of category *Food* are edible

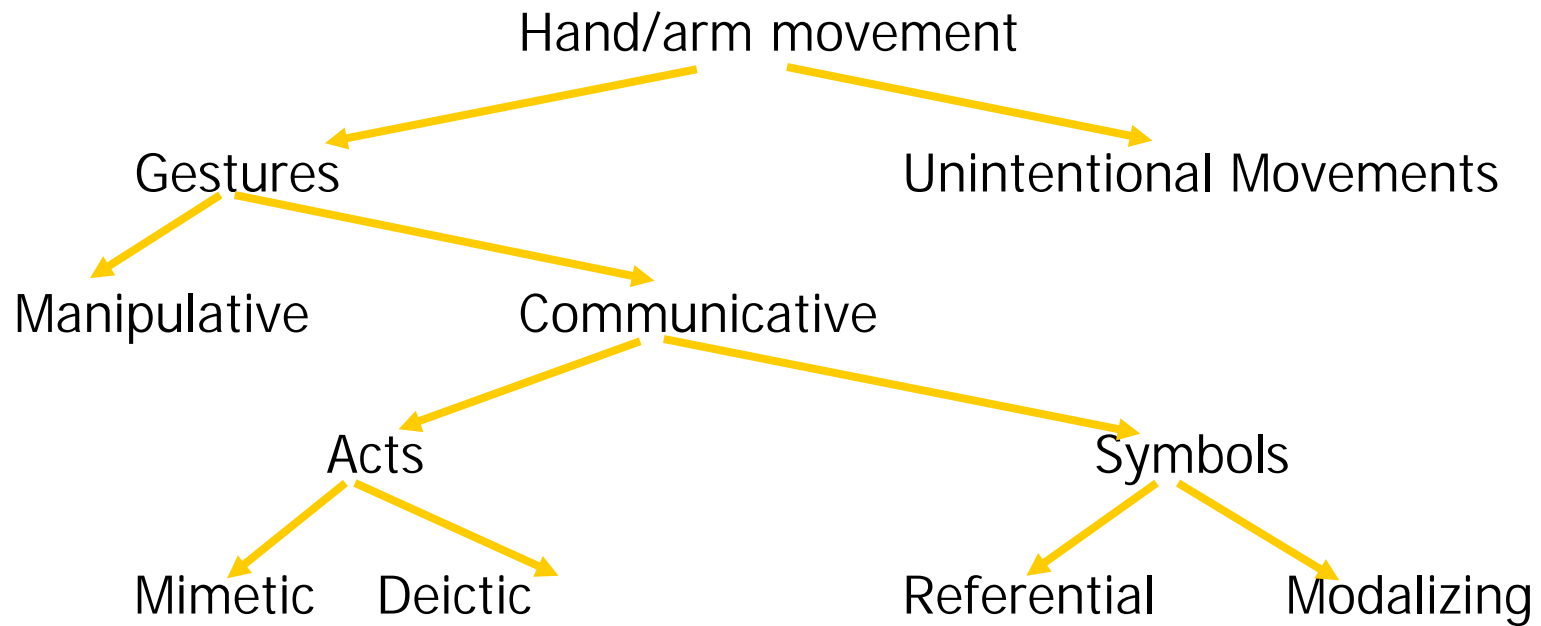
and *Fruits* is a subclass of *Food*

and *Apples* is a subclass of *Fruits*

then we know (through inheritance) that apples are edible.

- **Taxonomy**: hierarchy of subclasses
- Because categories are sets, we handle them as such.
e.g., two categories are **disjoint** if they have no member in common
a disjoint exhaustive decomposition is called a **partition**
etc...

Example: Taxonomy of hand/arm movements



Quek, 1994, 1995.

Measures

- Can be represented using units functions
e.g., $\text{Length}(L_1) = \text{Inches}(1.5) = \text{Centimeters}(3.81)$
- Measures can be used to describe objects
e.g., $\text{Mass}(\text{Tomato}_{12}) = \text{Kilograms}(0.16)$
- Caution: be careful to distinguish between measures and objects
e.g., $\forall b, b \in \text{DollarBills} \Rightarrow \text{CashValue}(b) = \(1.00)

Composite Objects



- One object can be part of another.
- PartOf relation is transitive and reflexive:
e.g., PartOf(Bucharest, Romania)
PartOf(Romania, EasternEurope)
PartOf(EasternEurope, Europe)
Then we can infer Part Of(Bucharest, Europe)
- Composite object: any object that has parts

Composite Objects (cont.)

- Categories of composite objects often characterized by their structure, i.e., what the parts are and how they relate.

e.g., $\forall a \text{ Biped}(a) \Rightarrow$

$\exists \text{ ll, lr, b}$

$\text{Leg}(\text{ll}) \wedge \text{Leg}(\text{lr}) \wedge \text{Body}(\text{b}) \wedge$

$\text{PartOf}(\text{ll}, a) \wedge \text{PartOf}(\text{lr}, a) \wedge \text{PartOf}(\text{b}, a) \wedge$

$\text{Attached}(\text{ll}, \text{b}) \wedge \text{Attached}(\text{lr}, \text{b}) \wedge$

$\text{ll} \neq \text{lr} \wedge$

$\forall x \text{ Leg}(x) \wedge \text{PartOf}(x, a) \Rightarrow (x = \text{ll} \vee x = \text{lr})$

- Such description can be used to describe any objects, including events. We then talk about **schemas** and **scripts**.

Events



- Chunks of spatio-temporal universe

e.g., consider the event WorldWarII

it has parts or sub-events: SubEvent(BattleOfBritain, WorldWarII)

it can be a sub-event: SubEvent(WorldWarII, TwentiethCentury)

- **Intervals:** events that include as sub-events all events occurring in a given time period (thus they are temporal sections of the entire spatial universe).
- Cf. situation calculus: fact true in particular situation
event calculus: event occurs during particular interval

Events (cont.)

- Places: spatial sections of the spatio-temporal universe that extend through time
- Use $In(x)$ to denote subevent relation between places; e.g. $In(\text{NewYork}, \text{USA})$
- **Location function:** maps an object to the smallest place that contains it:

$$\forall x, I \text{ Location}(x) = I \Leftrightarrow At(x, I) \wedge \forall II \text{ At}(x, II) \Rightarrow In(I, II)$$

Times, Intervals and Actions

- Time intervals can be partitioned between moments (=zero duration) and extended intervals:
- Absolute times can then be derived from defining a time scale (e.g., seconds since midnight GMT on Jan 1, 1900) and associating points on that scale with events.
- The functions Start and End then pick the earliest and latest moments in an interval. The function Duration gives the difference between end and start times.

$\forall i \text{ Interval}(i) \Rightarrow \text{Duration}(i) = (\text{Time}(\text{End}(i)) - \text{Time}(\text{Start}(i)))$

$\text{Time}(\text{Start}(\text{AD1900})) = \text{Seconds}(0)$

$\text{Time}(\text{Start}(\text{AD1991})) = \text{Seconds}(2871694800)$

$\text{Time}(\text{End}(\text{AD1991})) = \text{Seconds}(2903230800)$

$\text{Duration}(\text{AD1991}) = \text{Seconds}(31536000)$

Times, Intervals and Actions (cont.)

- Then we can define predicates on intervals such as:

$$\forall i, j \text{ Meet}(i, j) \Leftrightarrow \text{Time}(\text{End}(i)) = \text{Time}(\text{Start}(j))$$

$$\forall i, j \text{ Before}(i, j) \Leftrightarrow \text{Time}(\text{End}(i)) < \text{Time}(\text{Start}(j))$$

$$\forall i, j \text{ After}(j, i) \Leftrightarrow \text{Before}(i, j)$$

$$\forall i, j \text{ During}(i, j) \Leftrightarrow \text{Time}(\text{Start}(j)) \leq \text{Time}(\text{Start}(i)) \wedge \\ \text{Time}(\text{End}(j)) \geq \text{Time}(\text{End}(i))$$

$$\forall i, j \text{ Overlap}(i, j) \Leftrightarrow \exists k \text{ During}(k, i) \wedge \text{During}(k, j)$$

Objects Revisited



- It is legitimate to describe many objects as events
- We can then use temporal and spatial sub-events to capture changing properties of the objects

e.g.,

Poland event

19thCenturyPoland temporal sub-event

CentralPoland spatial sub-event

We call **fluents** objects that can change across situations.

Substances and Objects

- Some objects cannot be divided into distinct parts –
e.g., butter: one butter? no, some butter!
⇒ butter substance (and similarly for temporal substances)
(simple rule for deciding what is a substance: if you cut it in half, you should get the same).

How can we represent substances?

- Start with a category
e.g., $\forall x, y \quad x \in \text{Butter} \wedge \text{PartOf}(y, x) \Rightarrow y \in \text{Butter}$
- Then we can state properties
e.g., $\forall x \text{ Butter}(x) \Rightarrow \text{MeltingPoint}(x, \text{Centigrade}(30))$

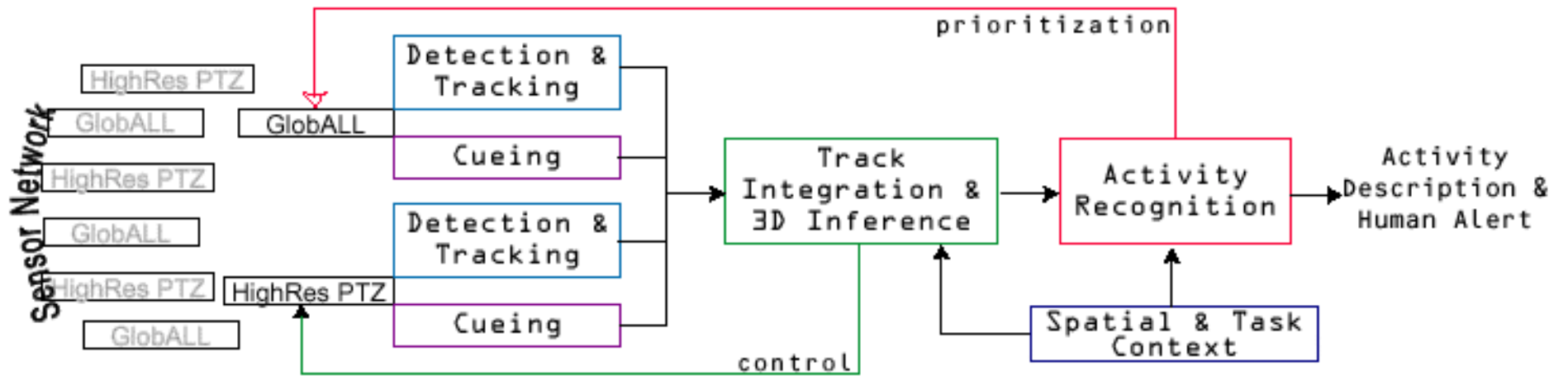
Example: Activity Recognition



- **Goal:** use network of video cameras to monitor human activity
- **Applications:** surveillance, security, reactive environments
- **Research:** IRIS at USC

Human activity detection

- Nevatia/Medioni/Cohen



Low-level processing

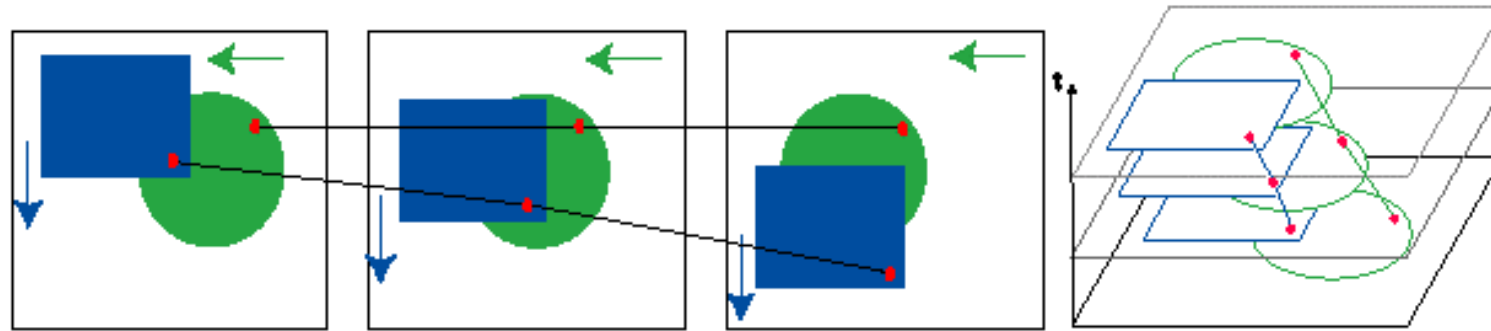


Figure 4: *Example of construction of paths from optical flow field in the $2D + t$ space.*

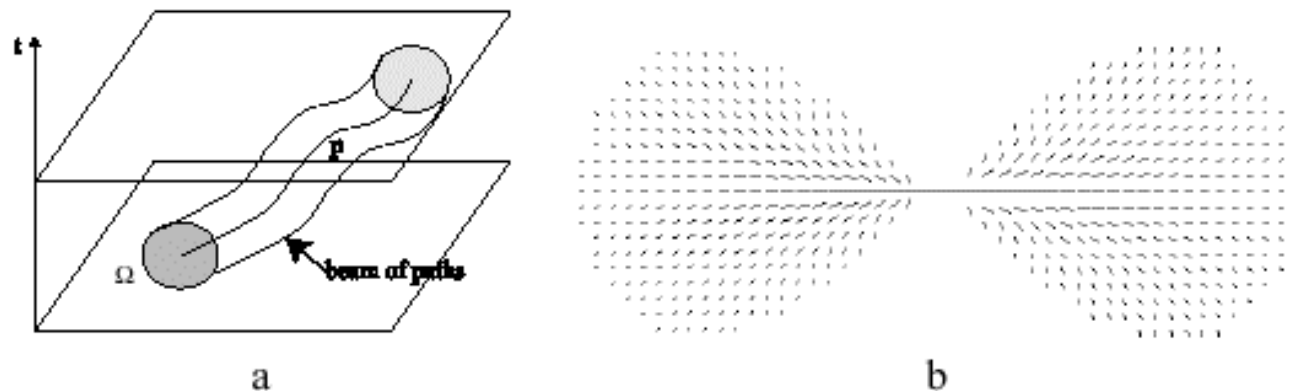


Figure 5: *Integration along a beam of paths of the motion field for robust inference of a pixel trajectory.*
a. *Illustration of the beam for a circular domain Ω .* **b.** *Illustration of the measure function $\mu(\omega)$ along the x -axis.*

Spatio-temporal representation

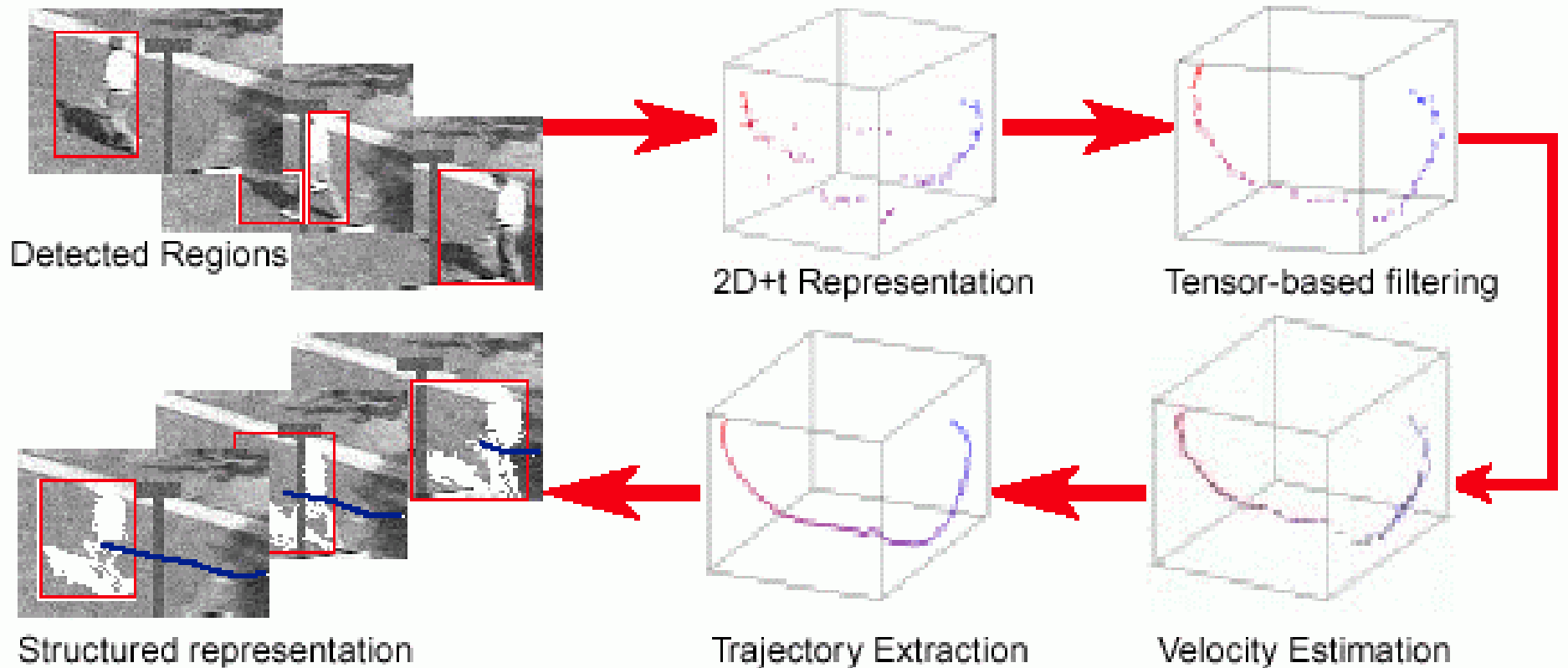
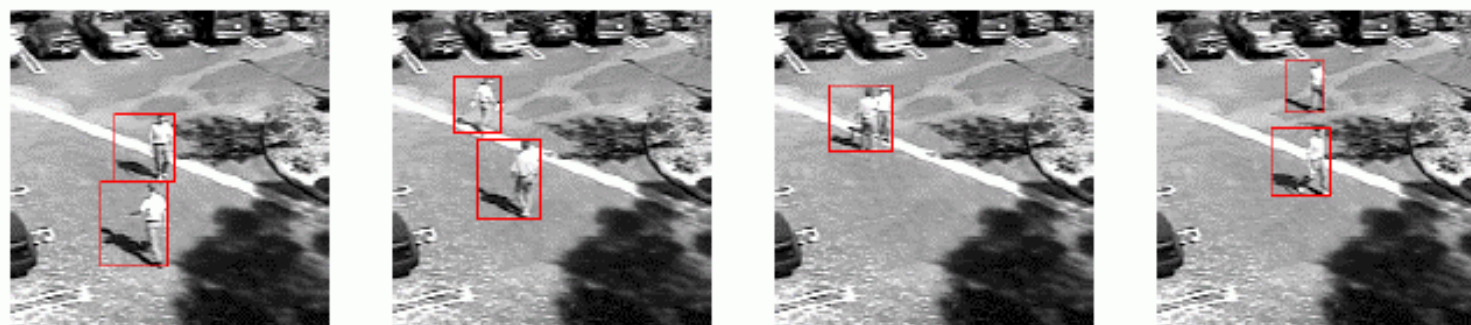
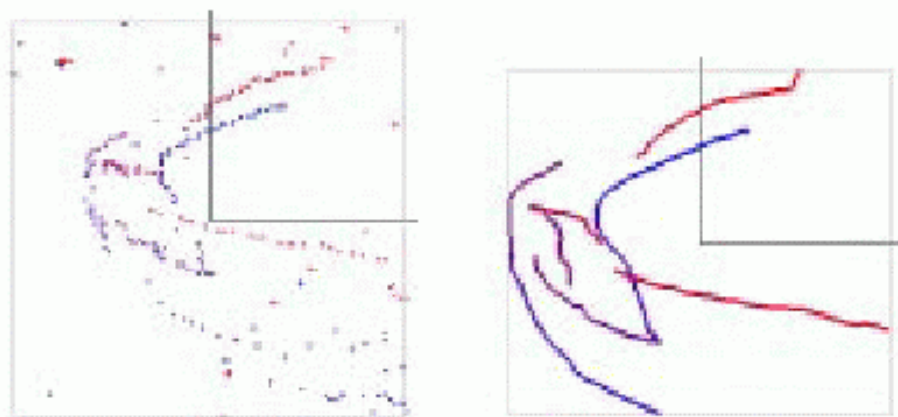


Figure 6: *Inferring the structured representation of a video stream.*



(a)



(b)



(c)

Figure 7: Structured representation of a video stream of two persons moving in a parking lot. (a) Detected moving regions, (b) $2D + t$ representation and inference of trajectories, (c) Mapping of the structured representation onto the original video frames.

Modeling Events

Spatial Location			Primary Motion	
at	between	above / below	toward / away	along
inside / outside	among	the front/back of	up / down	around
near / far	on top of	the left / right of	into / out of	through / across
next to	on bottom of		past	after / before

Table 1: English spatial prepositions (simplified from [27])

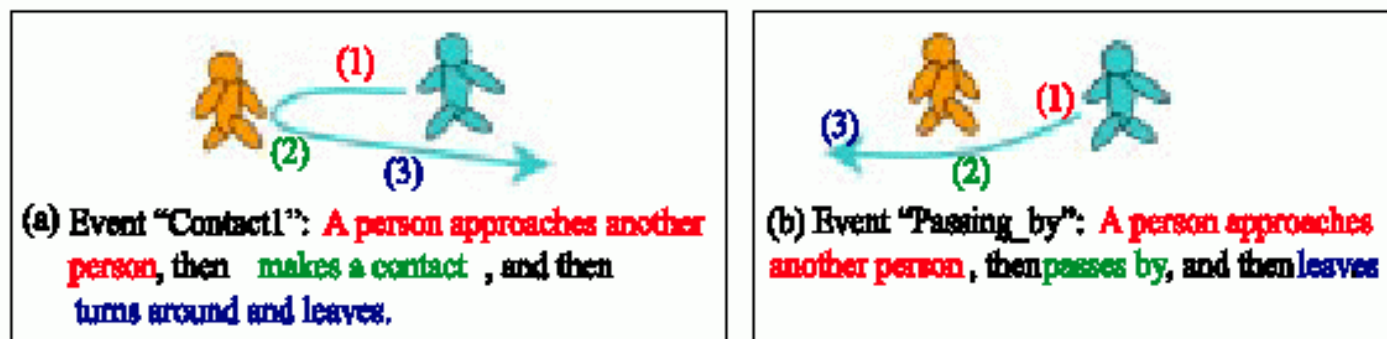
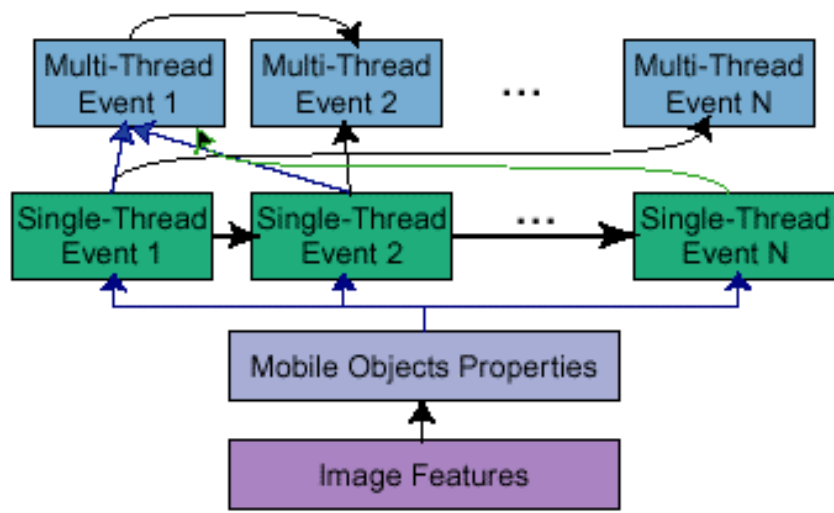
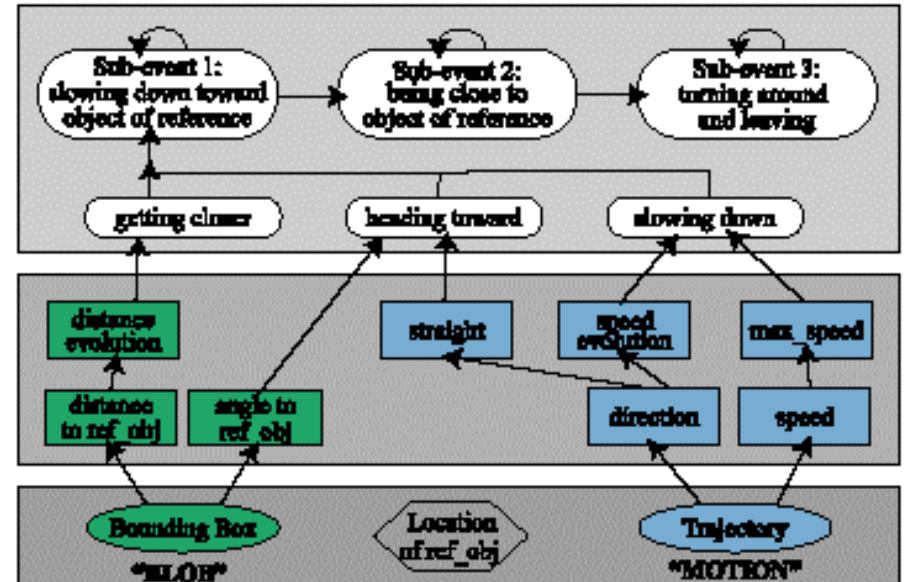


Figure 12: Modeling of two similar complex, single-thread events related to the meeting pattern of two persons. Each event is composed of three simple sub-events.

Modeling Events



(a) Event Modeling Schema

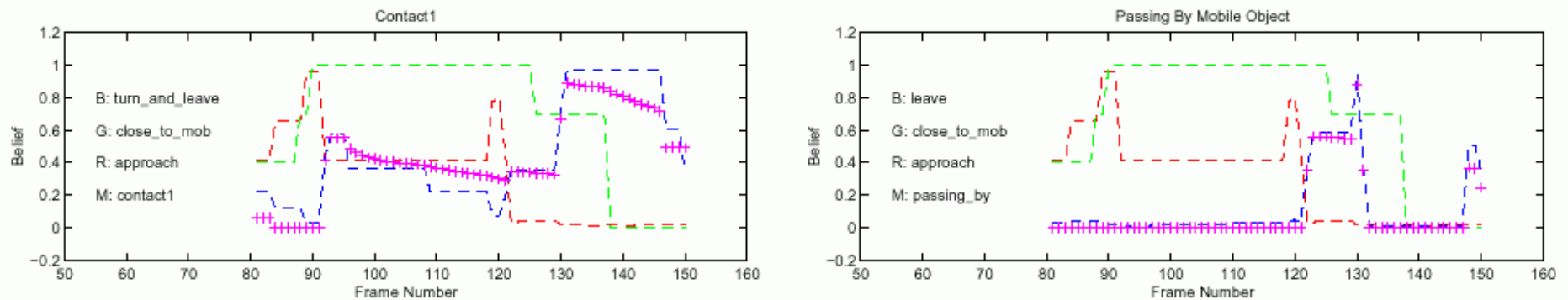


(b) A representation of complex event "Contact1"

Figure 13: A global view of our proposed scenario modeling; scenarios are defined as a single-thread or a multi-thread event which is described by the associated mobile object properties and image features.



(a) Detection and tracking of moving regions for scenario “CONTACT1”.



(b) Recognition results of two competing activities.

Figure 15: (a) Input sequence **A** shows a complex, single thread event “Contact1”. Object 1 (at the top) approaches object 2 (at the bottom), makes contact (both objects have merged as they meet), turns around and leaves. (b) Event “Contact1” is recognized with $P(MS^*|O) = 0.7$. Event “Passing By” is recognized with lower probability (almost 0 at the end) since sub-event “leaving without turning around” is not established.

Example 2: towards autonomous vision-based robots



- **Goal:** develop intelligent robots for operation in unconstrained environments
- **Subgoal:** want the system to be able to answer a question based on its visual perception

e.g., “Who is doing what to whom?”

While the robot is observing its environment.

Example

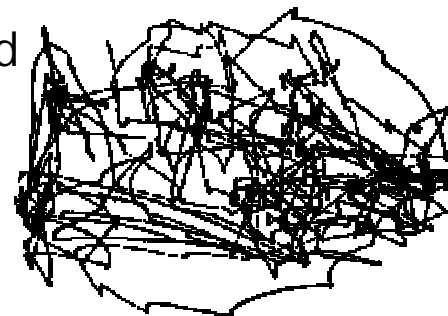
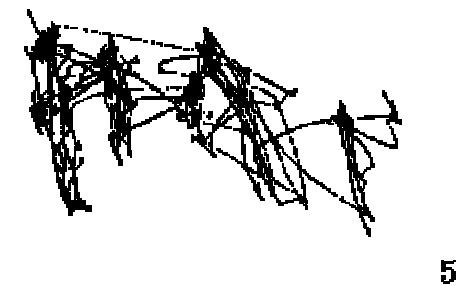
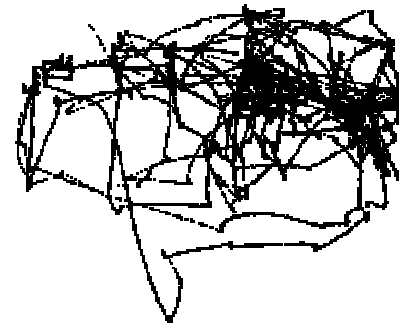
- **Question:** “who is doing what to whom?”



- **Answer:** “Eric passes, turns around and passes again”

Motivation: Humans

- 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



Yarbus, 1967

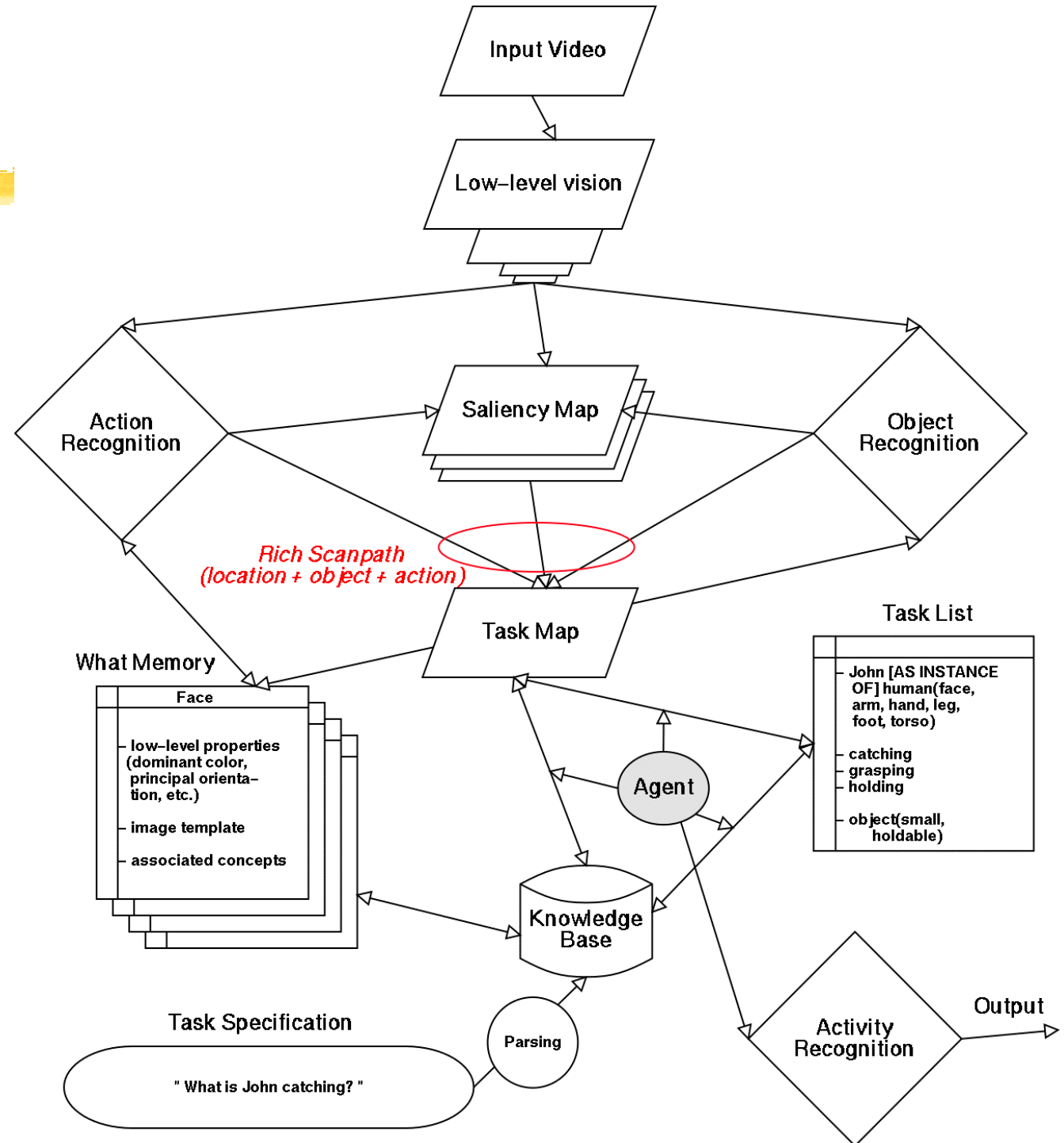
Minimal subscene



Extract “minimal subscene” (i.e., small number of objects and actions) that is relevant to present behavior.

Achieve representation for it that is robust and stable against noise, world motion, and egomotion.

General architecture



Example of operation

- Question: "What is John catching?"
- Video clip: John catching a ball

1) Initially: empty task map and task list

2) Question mapped onto a sentence frame

allows agent to fill some entries in the task list:

- concepts specifically mentioned in the question
- related concepts inferred from KB (ontology)

e.g., task list contains:

"John [AS INSTANCE OF] human(face, arm, hand, leg, foot, torso)"	(all derived from "John")
"catching, grasping, holding"	(derived from "catching")
"object(small, holdable)"	(derived from "what").

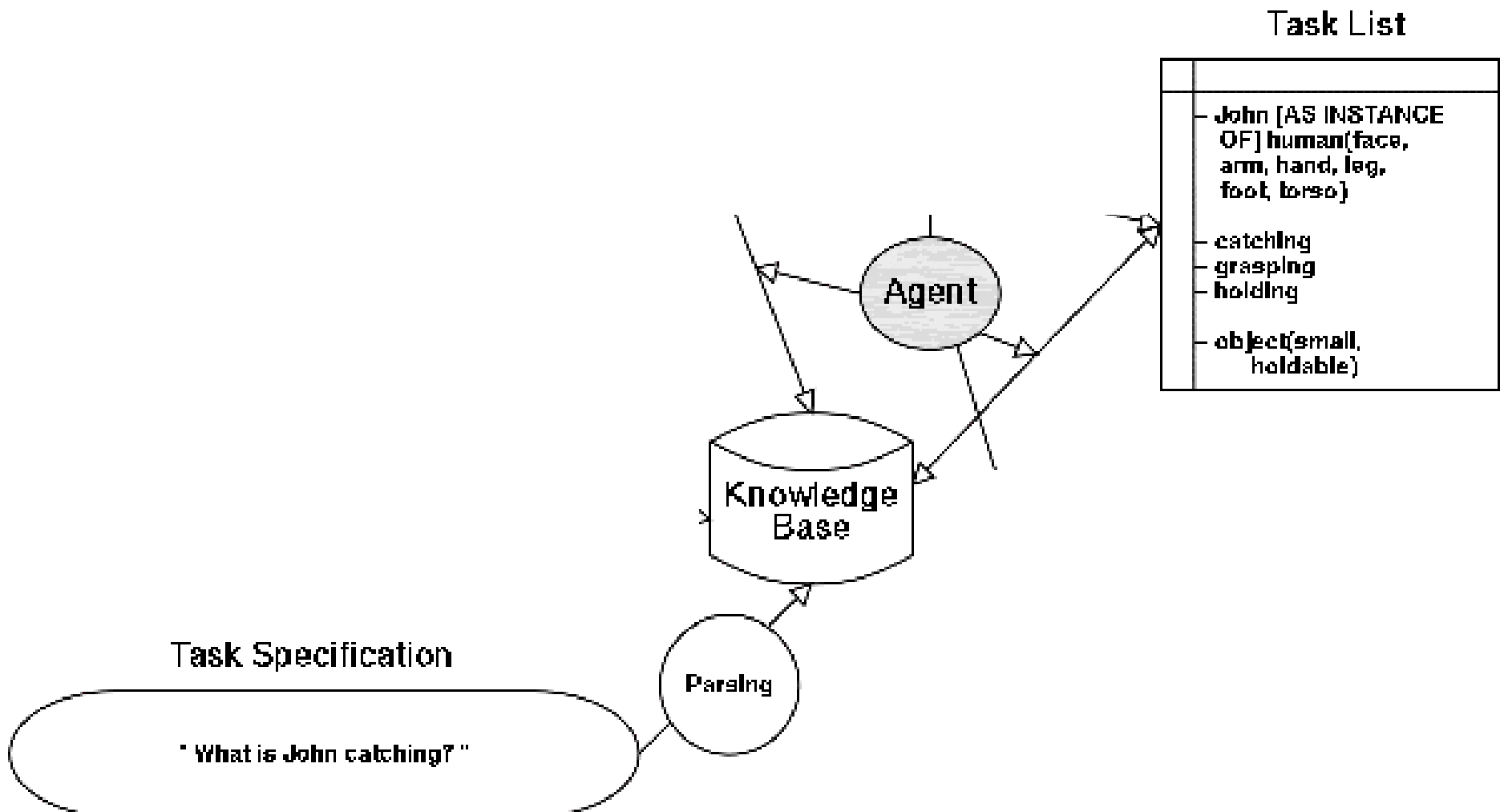
More formally: how do we do it?

- Use ontology to describe categories, objects and relationships:
Either with unary predicates, e.g., $\text{Human}(\text{John})$,
Or with reified categories, e.g., $\text{John} \in \text{Humans}$,
And with rules that express relationships or properties,
e.g., $\forall x \text{Human}(x) \Rightarrow \text{SinglePiece}(x) \wedge \text{Mobile}(x) \wedge \text{Deformable}(x)$
- Use ontology to expand concepts to related concepts:
E.g., parsing question yields “ $\text{LookFor}(\text{catching})$ ”
Assume a category HandActions and a taxonomy defined by
 $\text{catching} \in \text{HandActions}$, $\text{grasping} \in \text{HandActions}$, etc.
We can expand “ $\text{LookFor}(\text{catching})$ ” to looking for other actions in the
category where catching belongs through a simple expansion rule:
 $\forall a, b, c \quad a \in c \wedge b \in c \wedge \text{LookFor}(a) \Rightarrow \text{LookFor}(b)$

More formally: how do we do it?

- Use composite objects to describe structure and parts:

$$\begin{aligned} \forall h \text{ Human}(h) \Rightarrow & \exists f, la, ra, lh, rh, ll, rl, lf, rf, t \\ & \text{Face}(f) \wedge \text{Arm}(la) \wedge \text{Arm}(ra) \wedge \text{Hand}(lh) \wedge \text{Hand}(rh) \wedge \\ & \text{Leg}(ll) \wedge \text{Leg}(rl) \wedge \text{Foot}(lf) \wedge \text{Foot}(rf) \wedge \text{Torso}(t) \wedge \\ & \text{PartOf}(f, h) \wedge \text{PartOf}(la, h) \wedge \text{PartOf}(ra, h) \wedge \text{PartOf}(lh, h) \wedge \\ & \text{PartOf}(rh, h) \wedge \text{PartOf}(ll, h) \wedge \text{PartOf}(rl, h) \wedge \text{PartOf}(lf, h) \wedge \\ & \text{PartOf}(rf, h) \wedge \text{PartOf}(t, h) \wedge \\ & \text{Attached}(f, t) \wedge \text{Attached}(la, b) \wedge \text{Attached}(ra, b) \wedge \text{Attached}(ll, b) \wedge \\ & \text{Attached}(rl, t) \wedge \text{Attached}(lh, la) \wedge \text{Attached}(rh, ra) \wedge \\ & \text{Attached}(lf, ll) \wedge \text{Attached}(rf, rl) \wedge \text{Attached}(rh, ra) \wedge \\ & la \neq ra \wedge lh \neq rh \wedge ll \neq rl \wedge lf \neq rf \wedge \\ & \forall x \text{ Leg}(x) \wedge \text{PartOf}(x, a) \Rightarrow (x = ll \vee x = rl) \wedge \text{ [etc...]} \end{aligned}$$

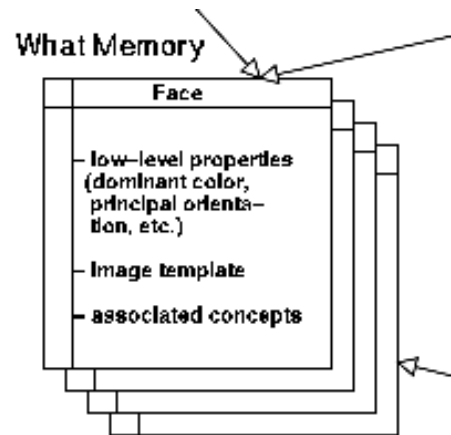


Example of operation

3) Task list creates top-down biasing signals onto vision, by associating concepts in task list to low-level image features in “what memory”

e.g., “human” => look for strong vertically-oriented features

“catching” => look for some type of motion



In more complex scenarios, not only low-level visual features, but also feature interactions, spatial location, and spatial scale and resolution may thus be biased top-down.

More formally: how do we do it?

- Use measures to quantify low-level visual features and weights:

e.g., describing the color of a face:

$\forall f \text{ Face}(f) \Rightarrow$

$$\text{Red}(f) = \text{Fweight}(0.8) \wedge \text{Green}(f) = \text{Fweight}(0.5) \wedge \text{Blue}(f) = \text{Fweight}(0.5)$$

[or use predicates similar to those seen for intervals to express ranges of feature weights]

e.g., recognizing face by measuring how well it matches a template:

$$\forall f \text{ RMSdistance}(f, \text{FaceTemplate}) < \text{Score}(0.1) \Rightarrow \text{Face}(f)$$

e.g., biasing the visual system to look for face color:

$$\forall f \text{ Face}(f) \wedge \text{LookFor}(f) \Rightarrow \text{RedWeight} = \text{Red}(f) \wedge \text{GreenWeight} = \text{Green}(f) \wedge \text{BlueWeight} = \text{Blue}(f)$$

[may eliminate $\text{Face}(f)$ if $\text{Red}()$, $\text{Green}()$ and $\text{Blue}()$ defined for all objects we might look for]

Example of operation


- 4) Suppose that the visual system first attends to a bright-red chair in the scene.

Going through current task list, agent determines that this object is most probably irrelevant (not really “holdable”)

Discard it from further consideration as a component of the minimal subscene.

Task map and task list remain unaltered.

Task List



John [AS INSTANCE OF] human(face, arm, hand, leg, foot, torso)
catching
grasping
holding
object(small, holdable)

More formally: how do we do it?

- What is the task list, given our formalism?

it's a question to the KB: $ASK(KB, \exists x \text{ LookFor}(x))$

- Is the currently attended and recognized object, o , of interest?

$ASK(KB, \text{LookFor}(o))$

- How could we express that if the currently attended & recognized object is being looked for, we should add it to the minimal subscene?

$$\forall x \text{ Attended}(x) \wedge \text{Recognized}(x) \wedge \text{LookFor}(x) \wedge \\ x \notin \text{MinimalSubscene} \Rightarrow x \in \text{MinimalSubscene}$$

with:

$$\forall x \exists t \text{ RMSdistance}(x, t) < \text{Score}(0.1) \Rightarrow \text{Recognized}(x)$$

and similar for $\text{Attended}()$

[Note: should be temporally tagged; see next]

Example of operation



5) Suppose next attended and identified object is John's rapidly tapping foot.

This would match the "foot" concept in the task list.

Because of relationship between foot and human (in KB), agent can now prime visual system to look for a human that overlap with foot found:

- feature bias derived from what memory for human
- spatial bias for location and scale

Task map marks this spatial region as part of the current minimal subscene.

Example of operation



6) Assume human is next detected and recognized

System should then look for its face
how? from KB we should be able to infer that resolving

"? [AS INSTANCE OF] human"

can be done by looking at the face of the human.

Once John has been localized and identified, entry

"John [AS INSTANCE OF] human(face, arm, hand, leg, foot, torso)"

simplifies into simpler entry

"John [AT] (x, y, scale)"

Thus, further visual biasing will not attempt to further localize John.

More formally: how do we do it?

- How do we introduce the idea of successive attentional shifts and progressive scene understanding to our formalism?

Using situation calculus!

- Effect axioms (describing change):

$$\forall x,s \text{ Attended}(x, s) \wedge \text{Recognized}(x, s) \wedge \text{LookFor}(x, s) \Rightarrow \\ \neg \text{LookFor}(x, \text{Result}(\text{AddToMinimalSubscene}, s))$$

with AddToMiminalSubscene a shorthand for a complex sequence of actions to be taken (remember how very long predicates should be avoided!)

- Successor-state axioms (better than the frame axioms for non-change):

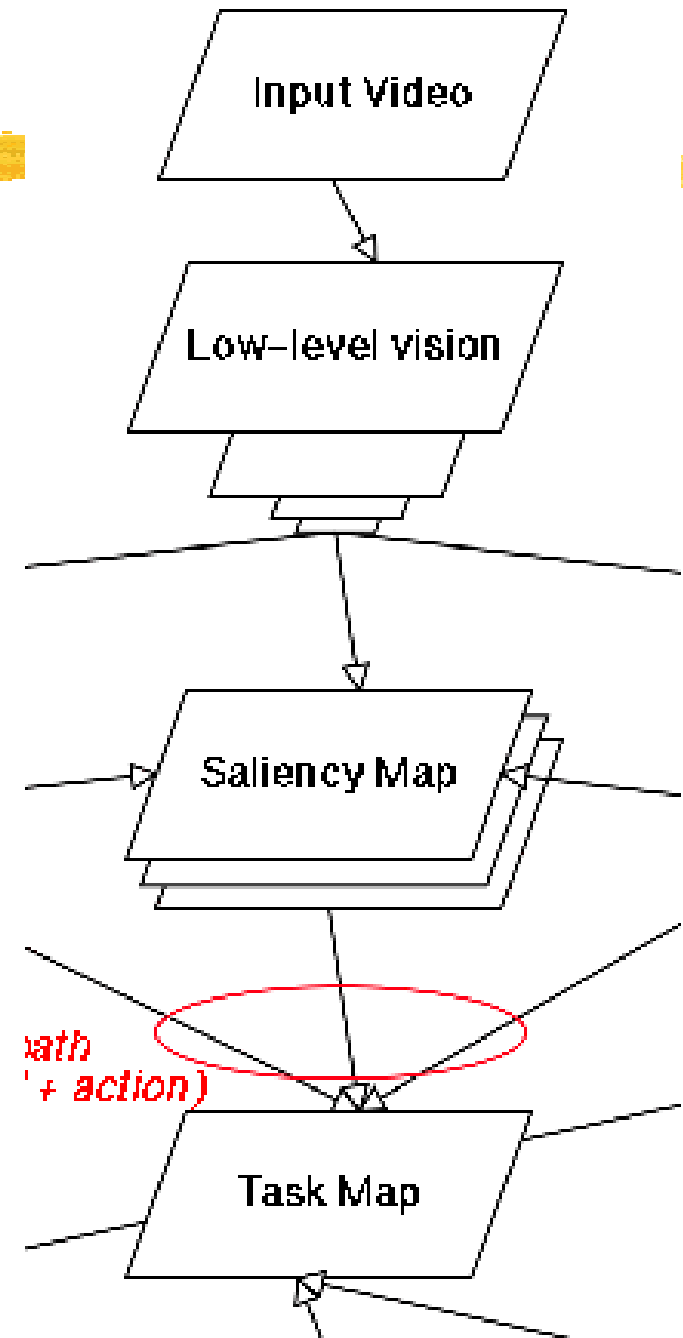
$$\forall x,a,s \quad x \in \text{MinimalSubscene}(\text{Result}(a, s)) \Leftrightarrow \\ (a = \text{AddToMinimalSubscene}) \vee \\ (x \in \text{MinimalSubscene}(s) \wedge a \neq \text{DeleteFromminimalSubscene})$$

Example of operation

7) Suppose system then attends to the bright green emergency exit sign in the room

This object would be immediately discarded because it is too far from the currently activated regions in the task map.

Thus, once non-empty, the **task map** acts as a filter that makes it more difficult (but not impossible) for new information to reach higher levels of processing, that is, in our model, matching what has been identified to entries in the task list and deciding what to do next.



Example of operation



8) Assume that now the system attends to John's arm motion

This action will pass through the task map (that contains John)

It will be related to the identified John (as the task map will not only specify spatial weighting but also local identity)

Using the knowledge base, what memory, and current task list the system would prime the expected location of John's hand as well as some generic object features.

Example of operation



- 9) If the system attends to the flying ball, it would be incorporated into the minimal subscene in a manner similar to that by which John was (i.e., update task list and task map).
- 10) Finally: activity recognition.

The various trajectories of the various objects that have been recognized as being relevant, as well as the elementary actions and motions of those objects, will feed into the activity recognition sub-system

=> will progressively build the higher-level, symbolic understanding of the minimal subscene.

e.g., will put together the trajectories of John's body, hand, and of the ball into recognizing the complex multi-threaded event "human catching flying object."

Example of operation



- 11) Once this level of understanding is reached, the data needed for the system's answer will be in the form of the task map, task list, and these recognized complex events, and these data will be used to fill in an appropriate sentence frame and apply the answer.

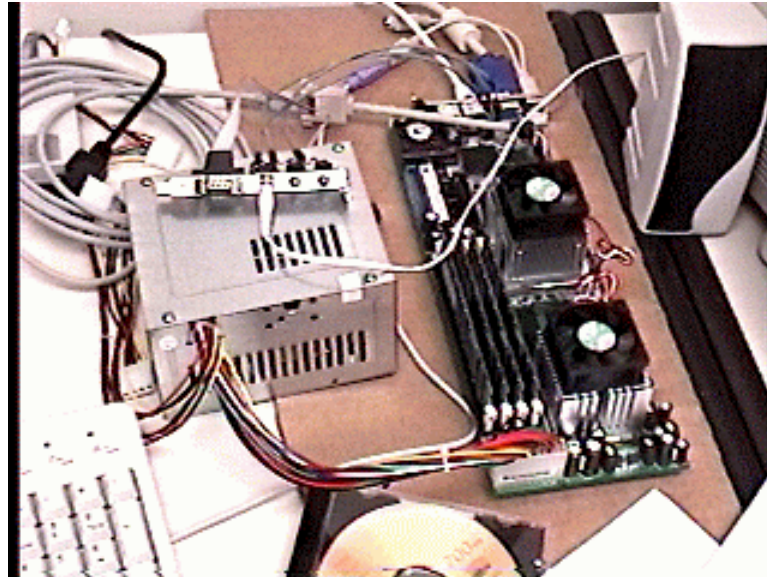
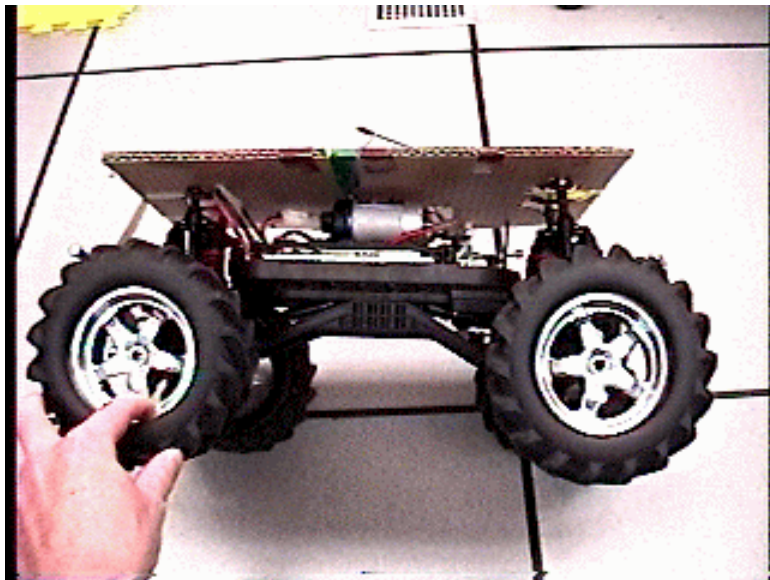
Reality or fiction?



Ask your colleague, Vidhya Navalpakkam!

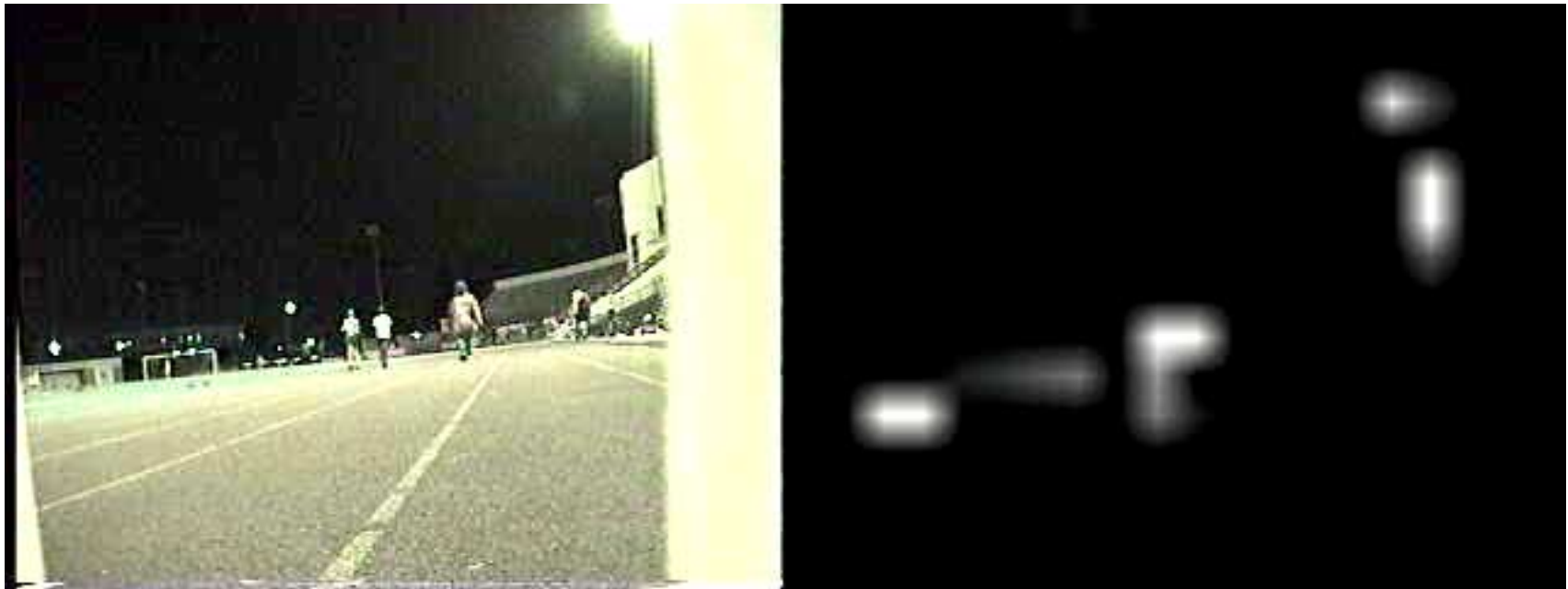
Meanwhile...

- Beobots are coming to life!



Meanwhile...

And they can see!



Example

- **Question:** “who is doing what to whom?”



- **Answer:** “Eric passes, turns around and passes again”