THE UNIVERSITY of ADELAIDE

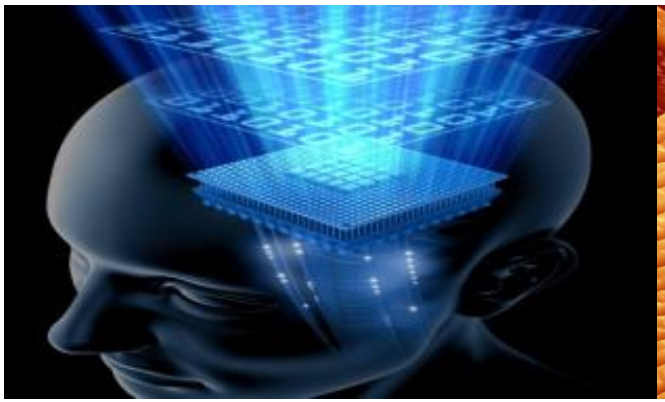
Perceptual Hierarchical Grouping in Soar

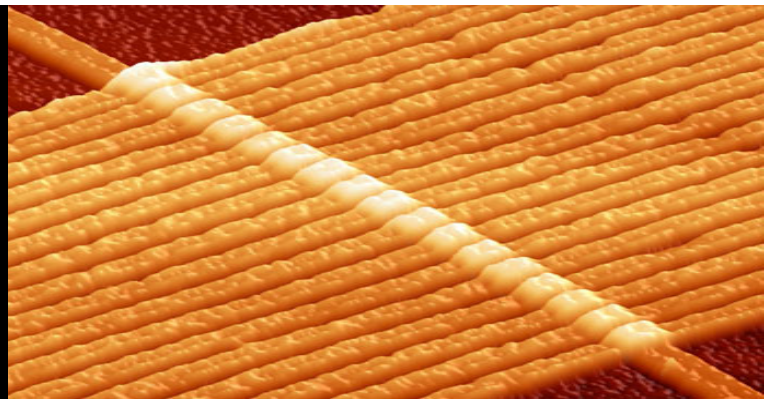
Why Elaborations Rock!

Dr. Danny Gibbins

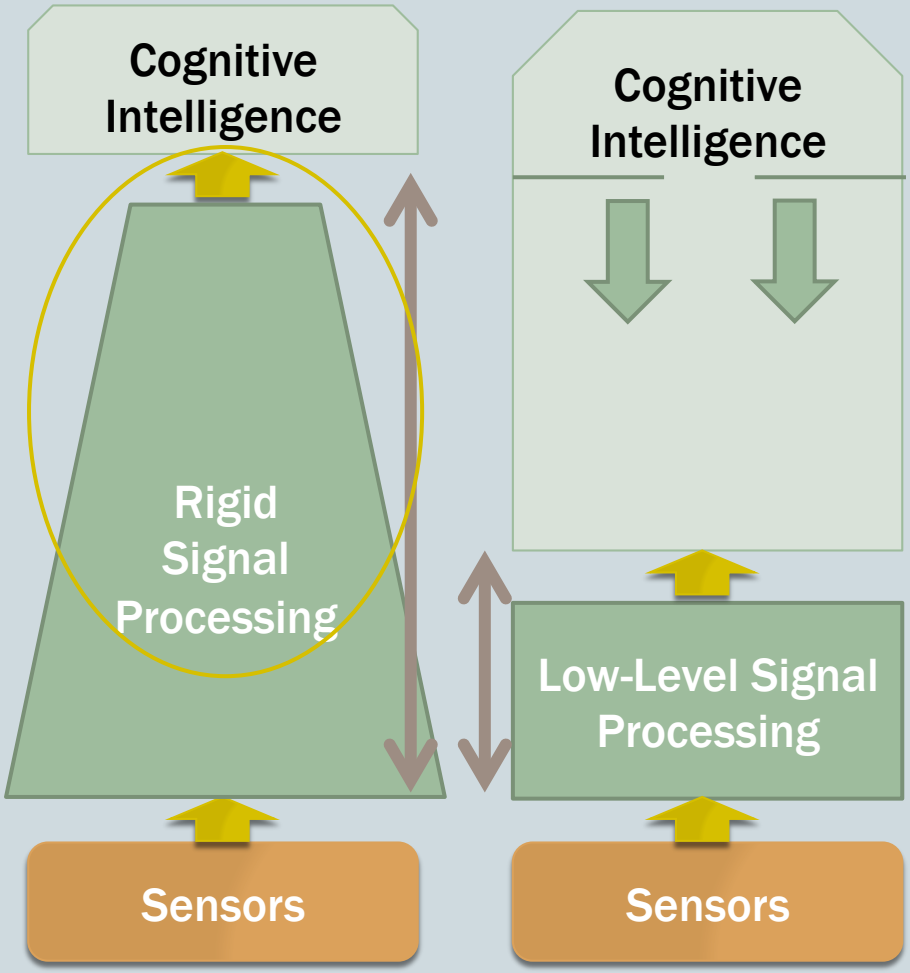
Simon Keen

Dr. Braden Phillips



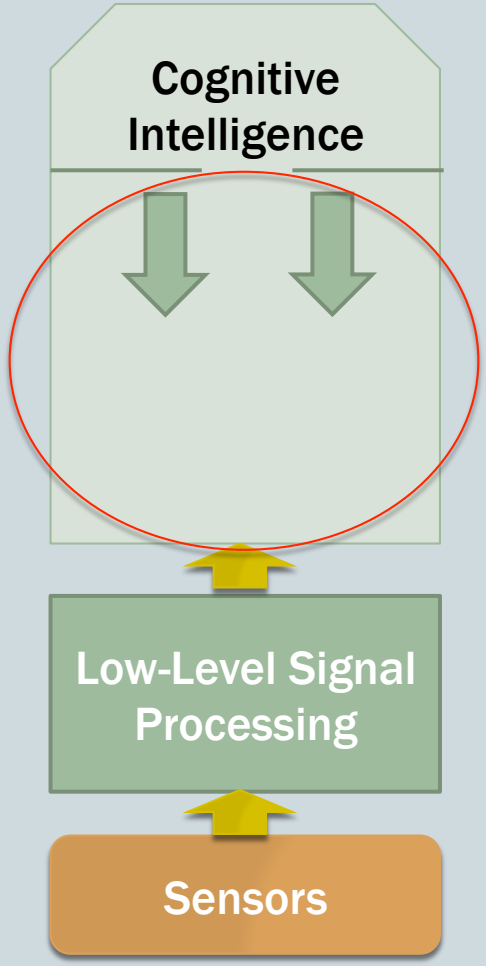


Big Picture



- Reduce gap between Cognitive Intelligence and Sensors
- Apply the concepts of Cognitive Intelligence to lower level perceptual tasks

Big Picture

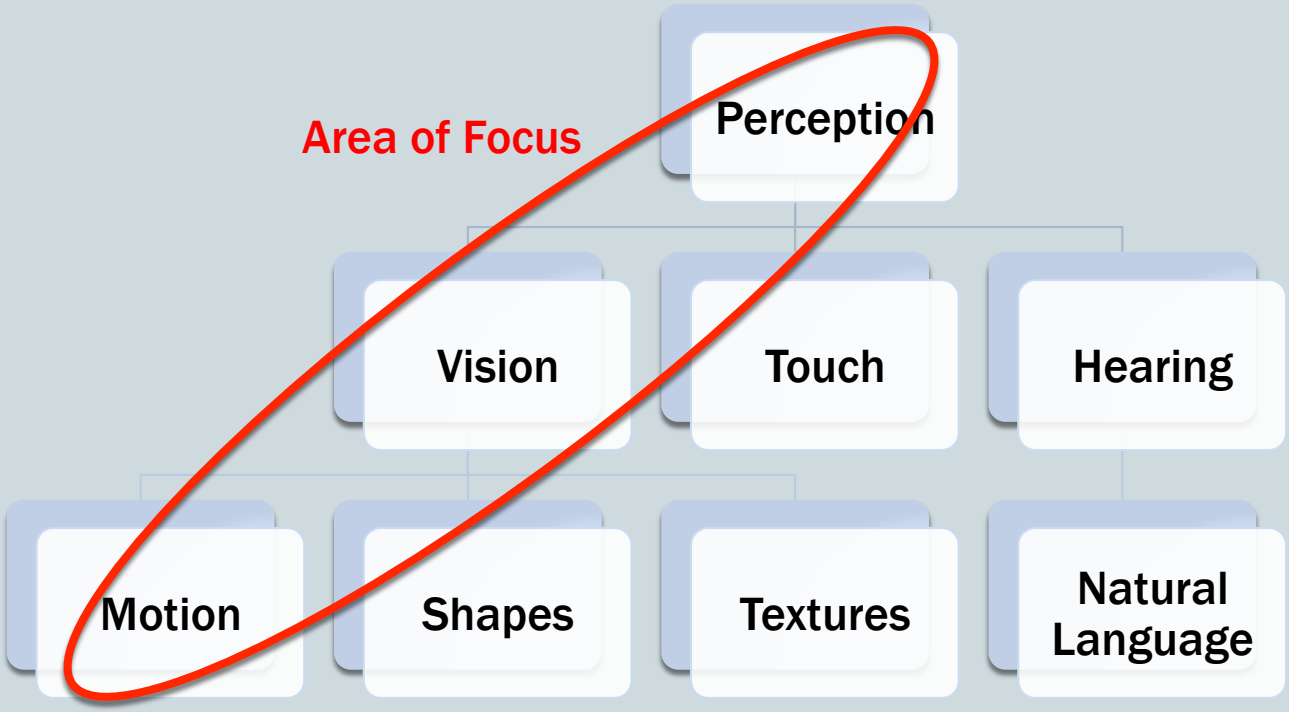


- Concepts
  - Reasoning – what did we do last time?
  - Learning – What is important?
  - Knowledge Representation
  - Planning

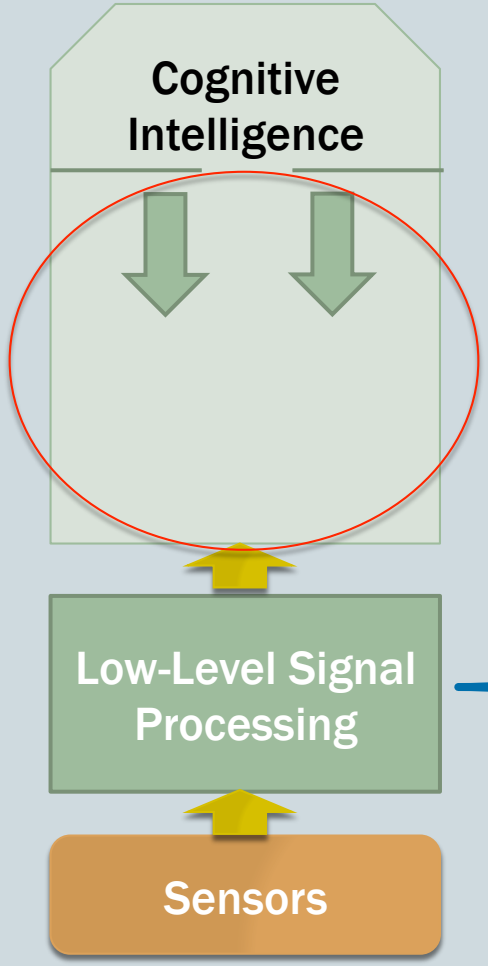
Can we avoid designing specific lower level reasoning system and learn how to perceive information?

Perception

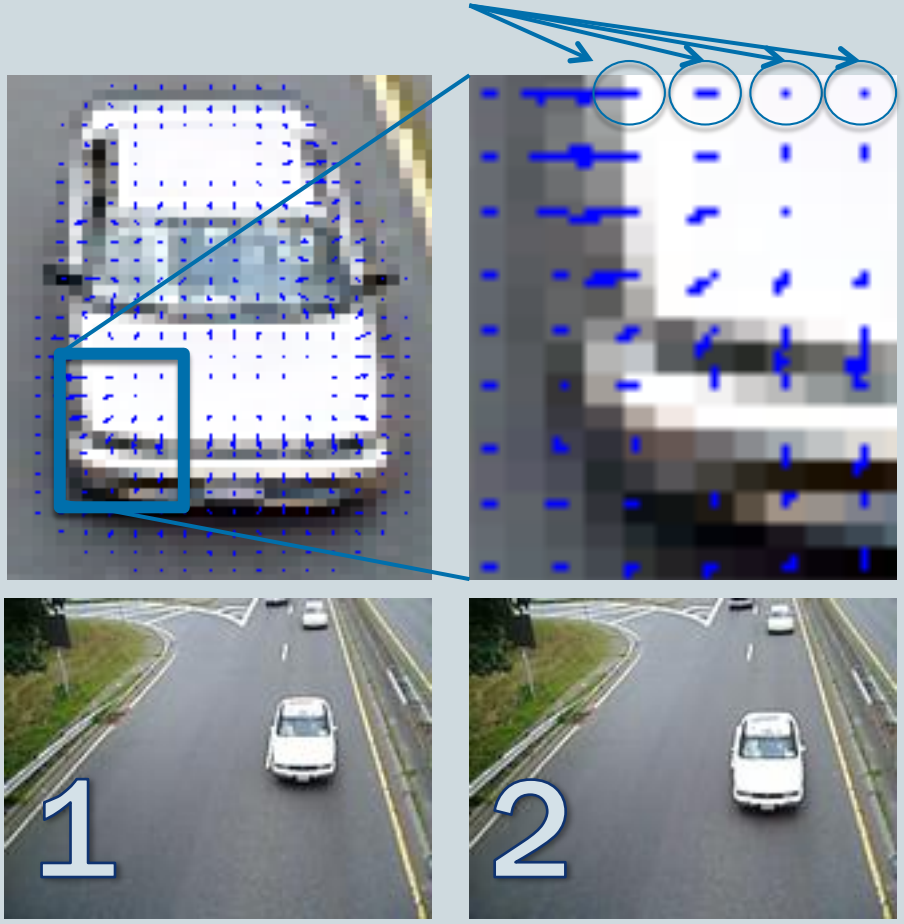
- Chosen to focus on Motion as a starting point.



Motion



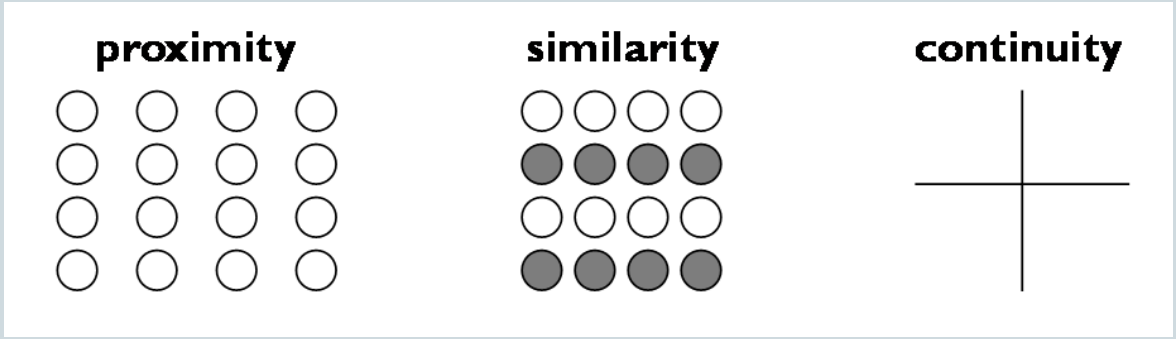
- Optical Flow (Objects)



Grouping

Create a Set of simple rules which can combine lower level information

Possible approaches include Gestalt Rules "perceptual organization" – sum of its parts

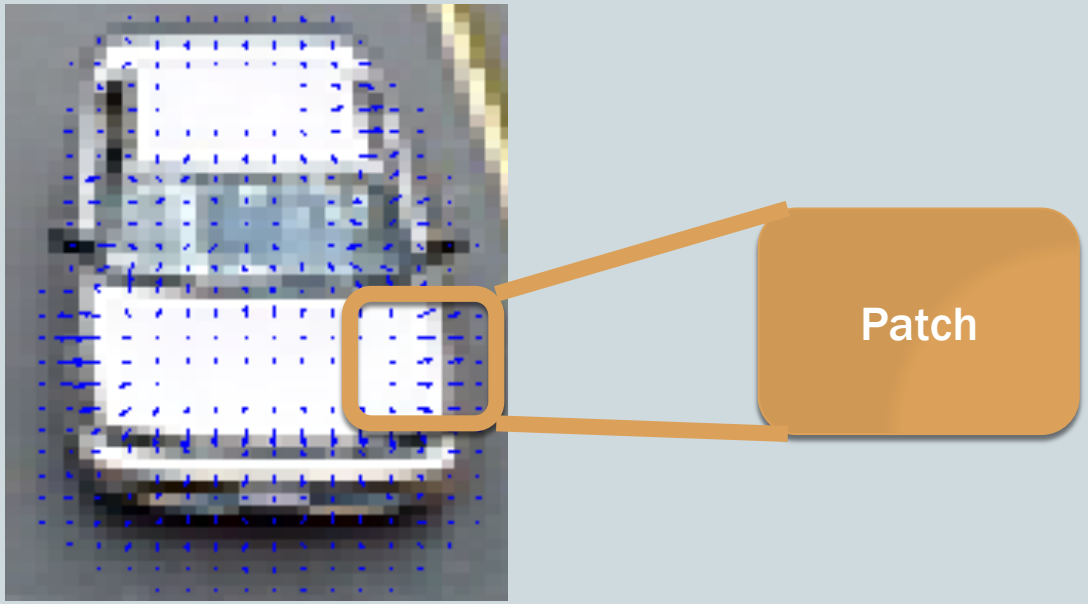


http://faculty.washington.edu/ionefine/S&P2009pdfs/S&P2009_chapter4.pdf

http://leesbirdblog.files.wordpress.com/2013/07/mixed-flock-of-birds-flying-in-a-v-formation-put-together-free-no-license-bankoboev-ru_staya_ptic_letit_na_yug.jpg

Patch

- Patch is a sub-region for which a proximity is defined

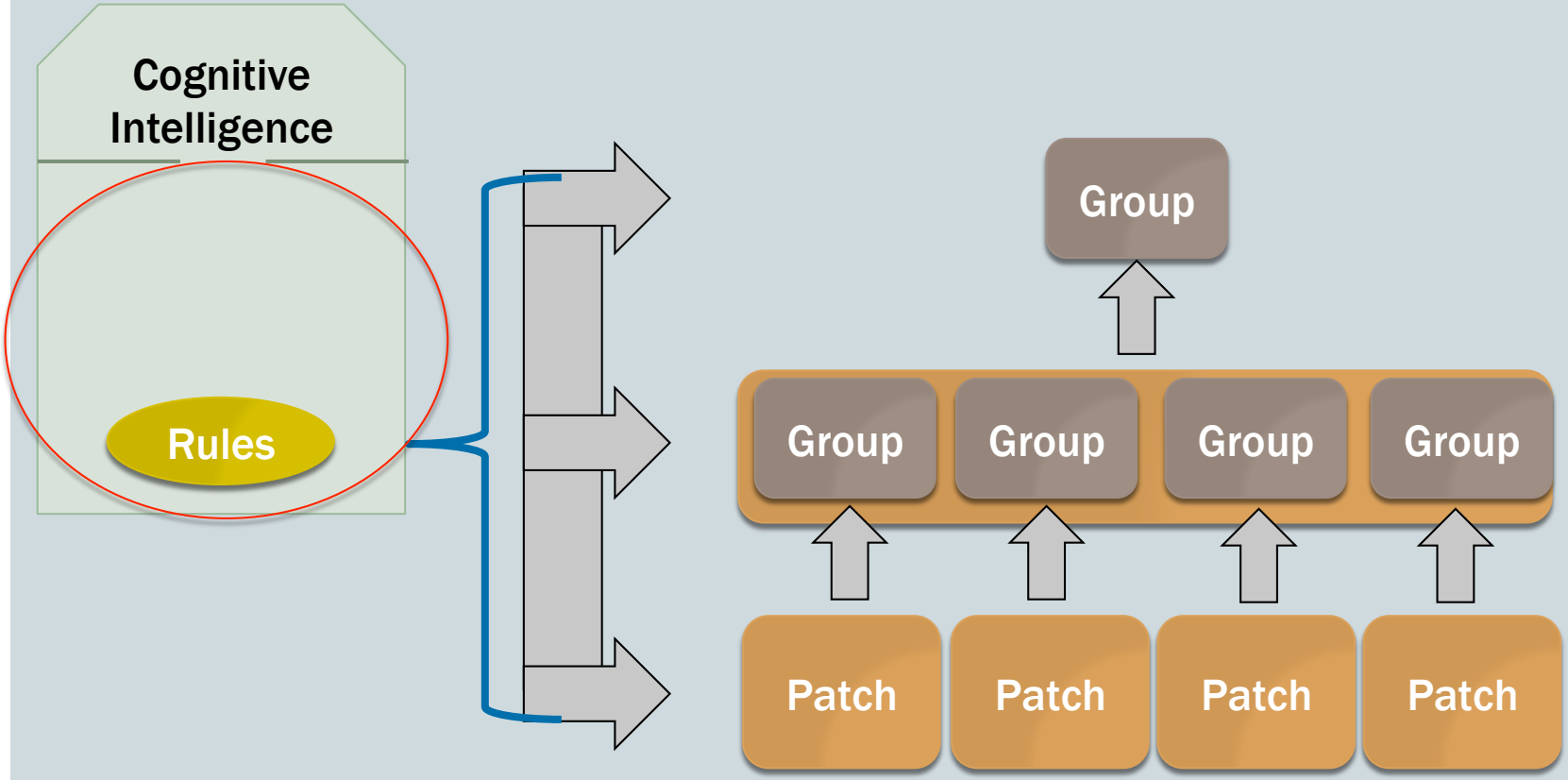


How do we Group a local Patch

- Simple set of Rules
  - What is “proximity”?
  - What is “similar”?
    - Angle
    - Magnitude

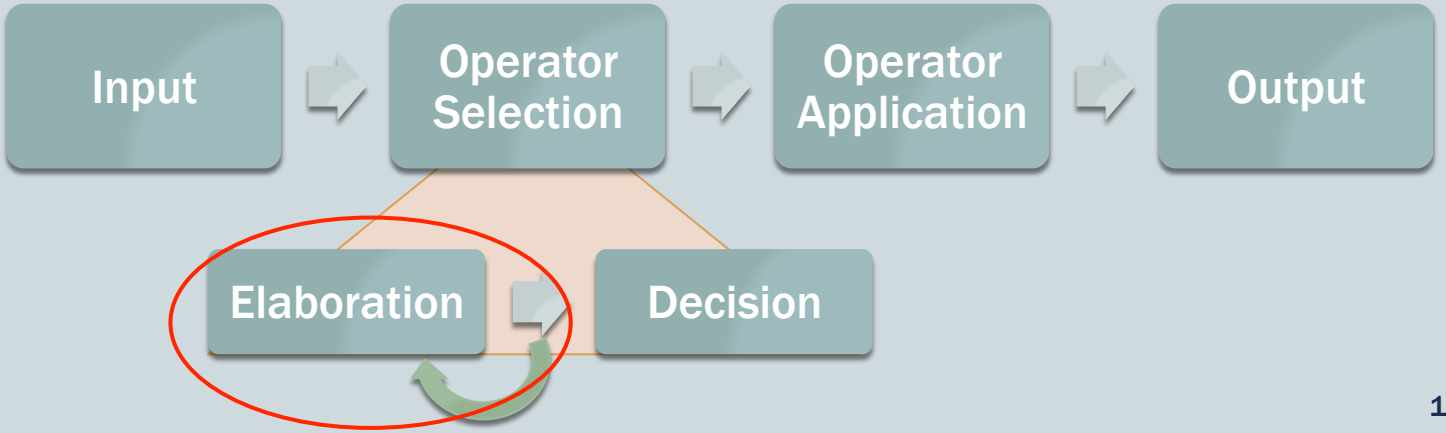
Hierarchical Grouping

- Rules are the Same and apply at all levels allowing for multi-scale

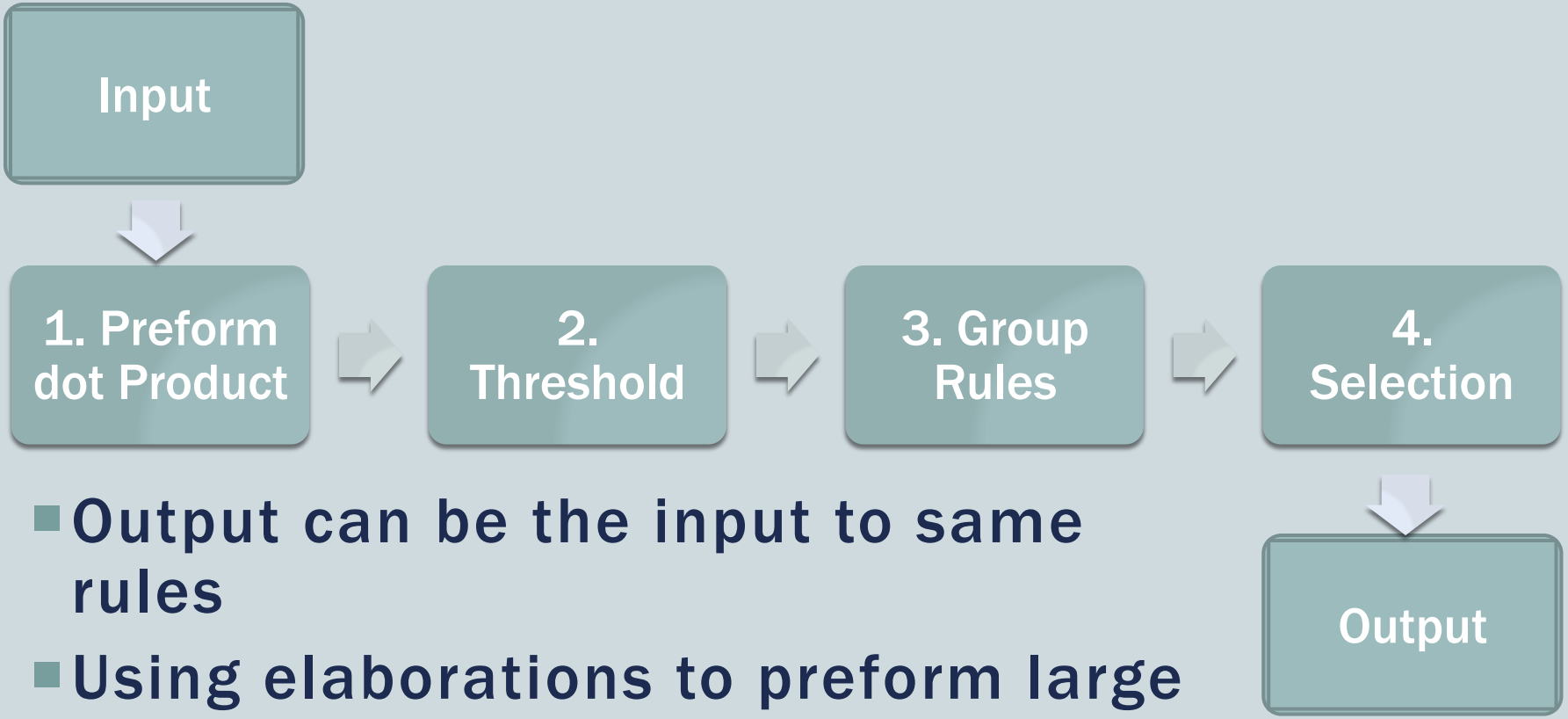


Why Soar?

- Production system allows for ‘in theory’ parallel execution
- Synergy with hardware implementation of Soar were working on
- Available Learning Capabilities
- Power of using Elaborations to generate parallel evaluation in a single Soar cycle



Power of Elaborations



- Output can be the input to same rules
- Using elaborations to preform large amount of evaluations of input data

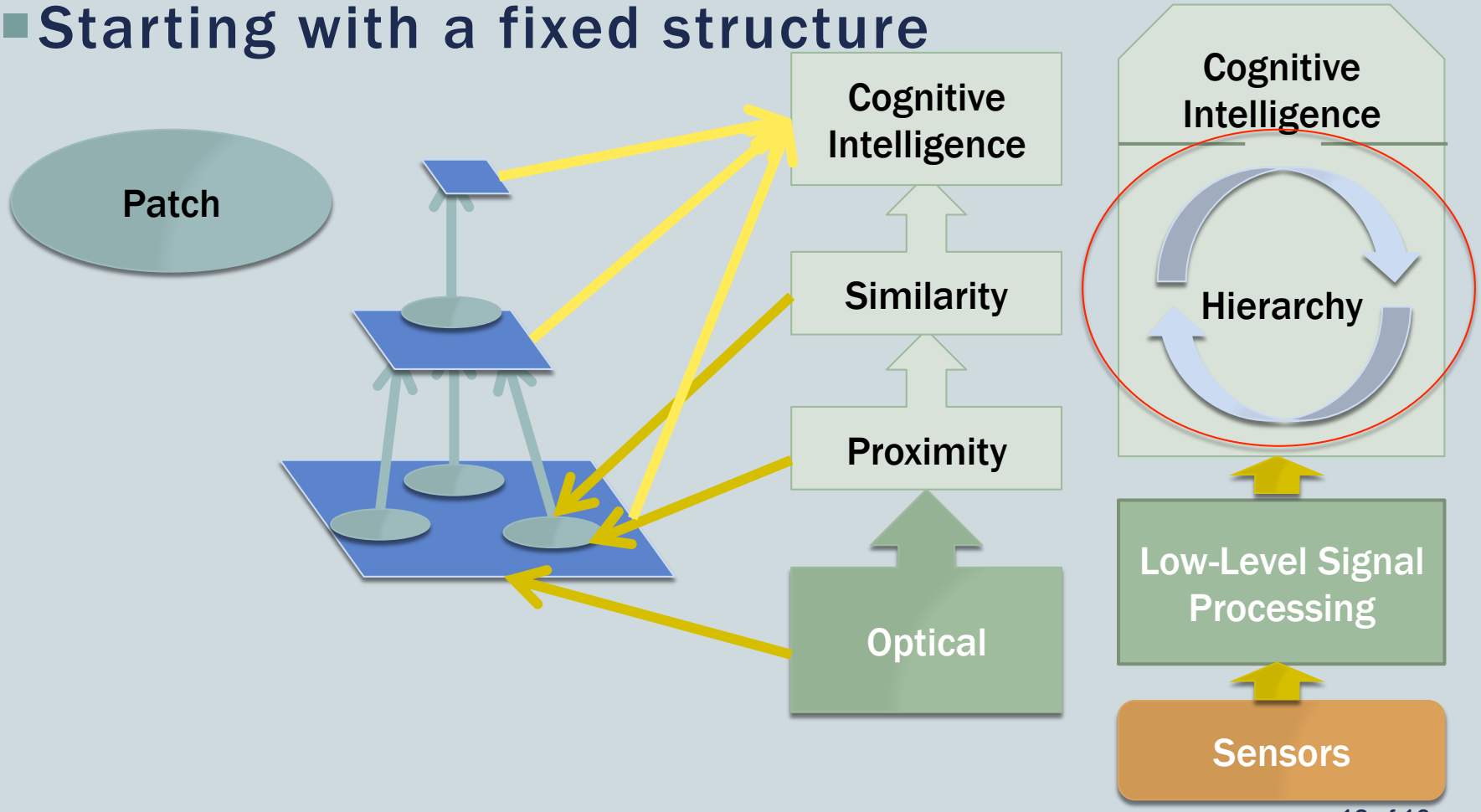
Grouping Rules

- What is defined as similar?
- What is good proximity?

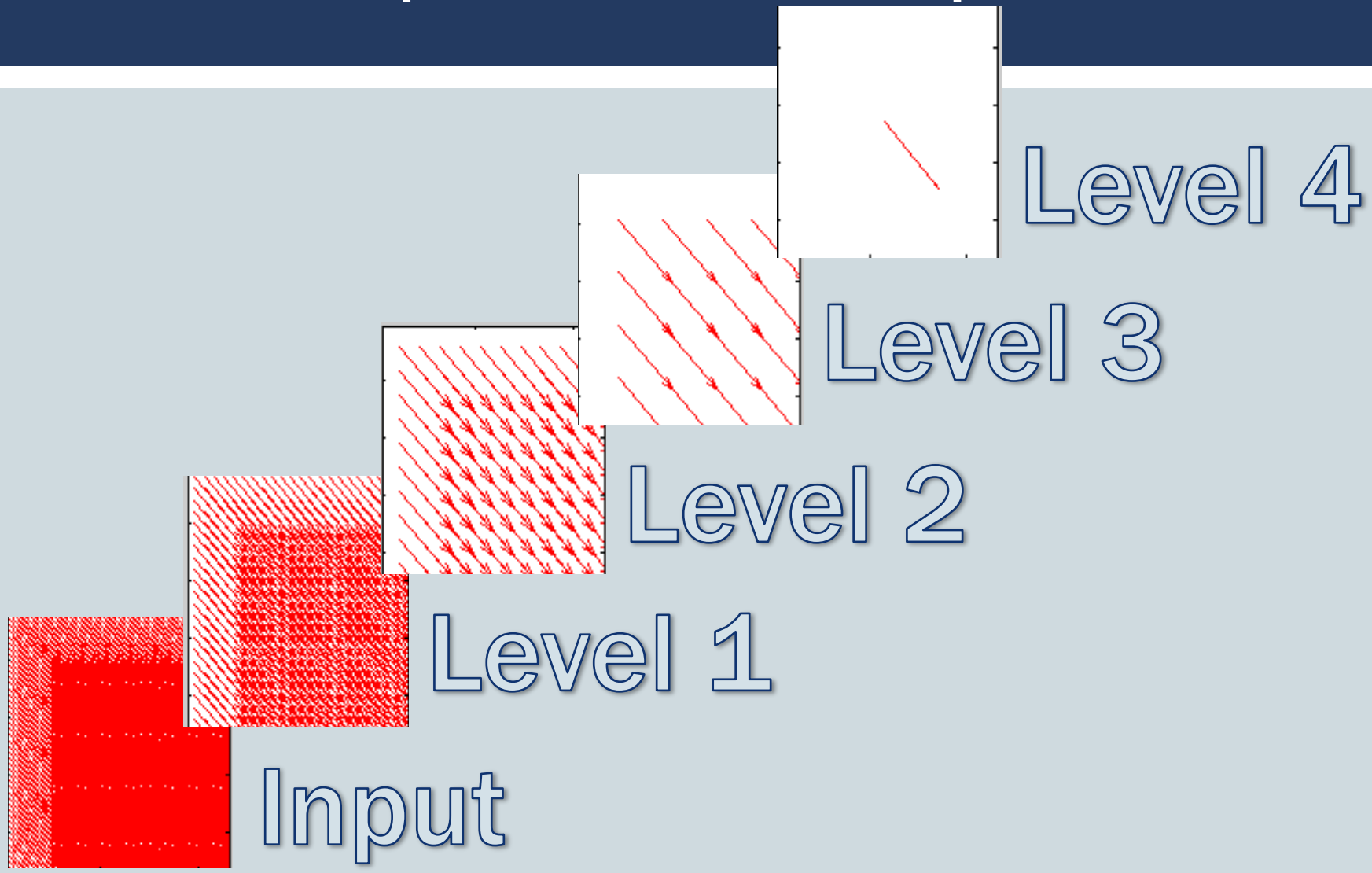
- These change with time and are dynamic based on application
- Knowledge can be used to dynamically tune parameters

How do we implement it?

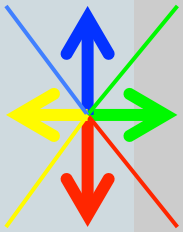
- Starting with a fixed structure

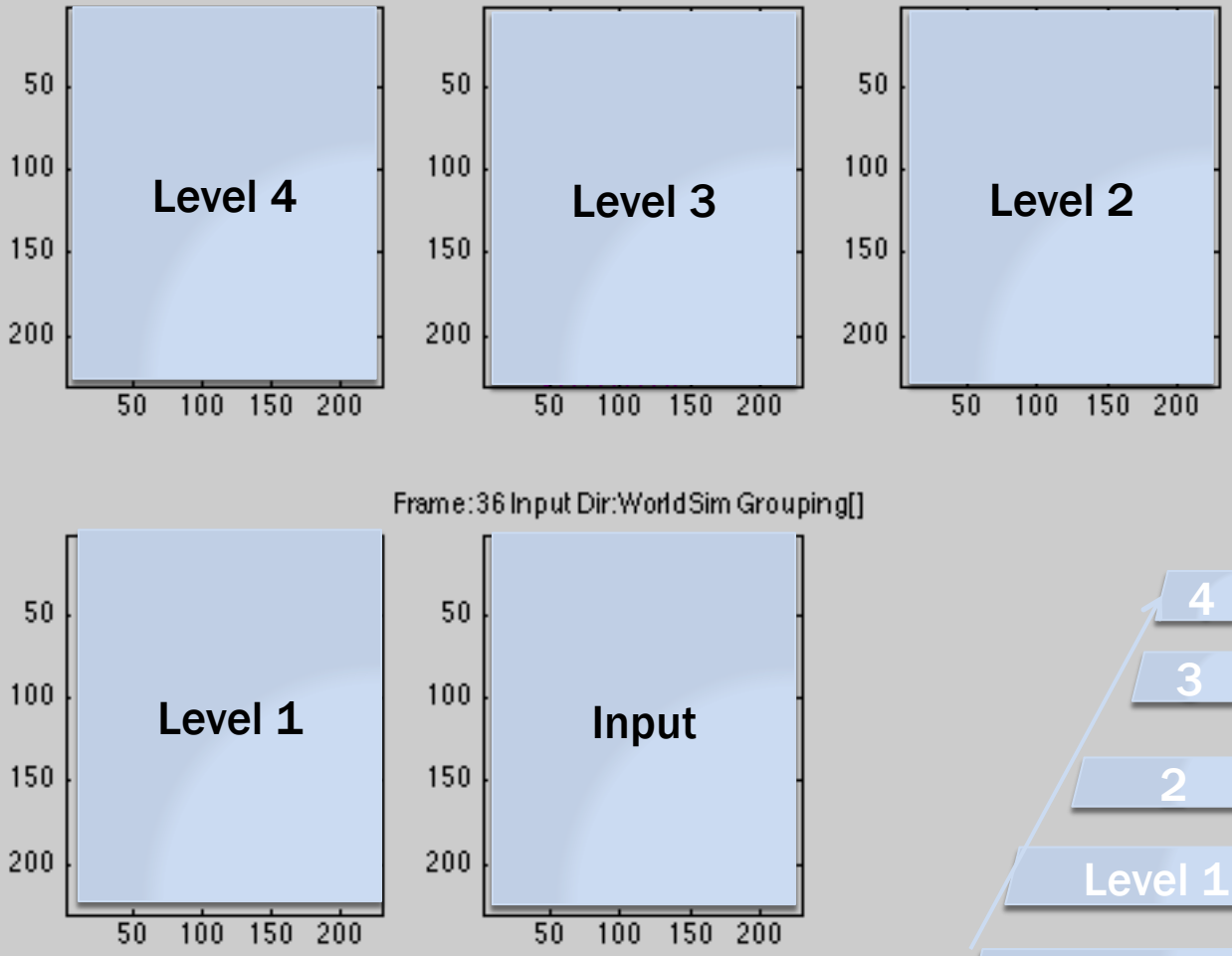


Optical Flow Example

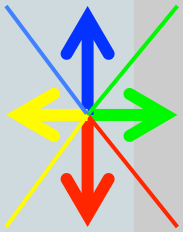


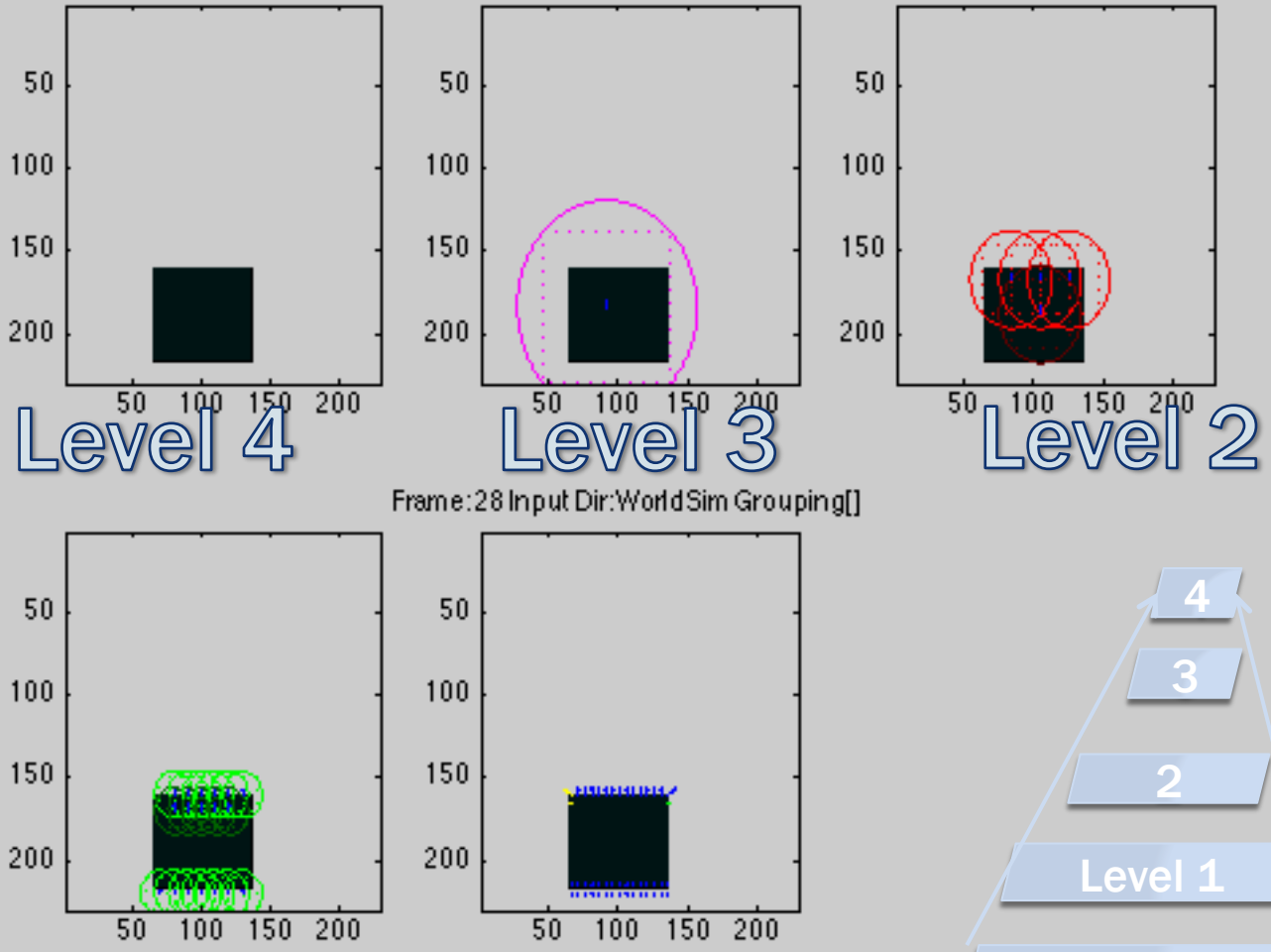
Example Layouts



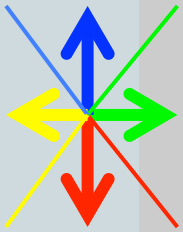


Simulated Example



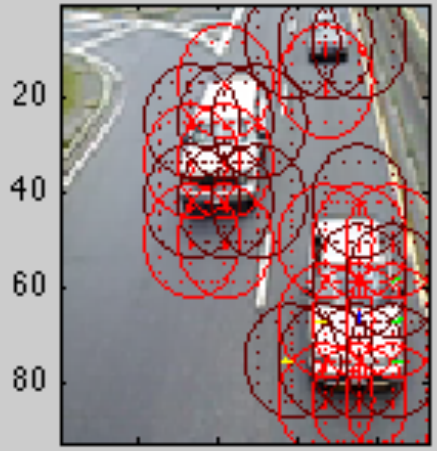


Real Example



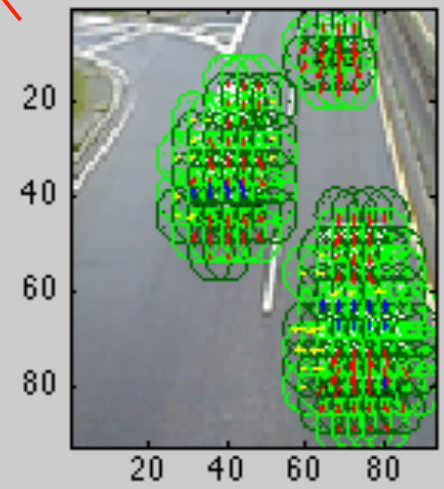
Level 4

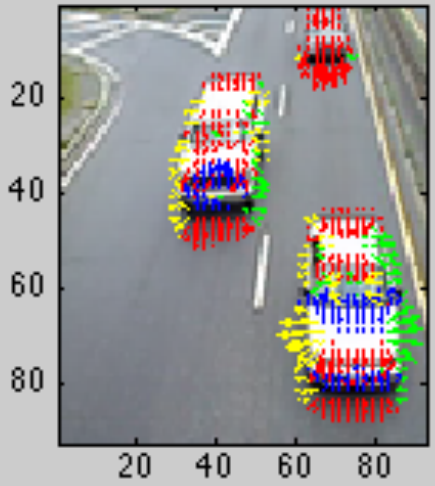
Level 3

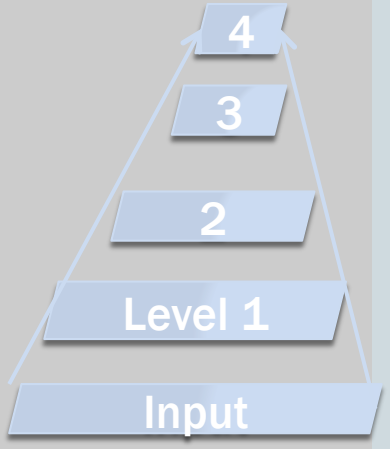


Level 2

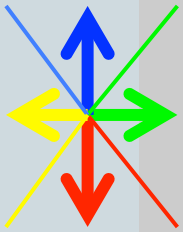
Frame:36 Input Dir:traffic.mj2 Grouping[]







Real Example

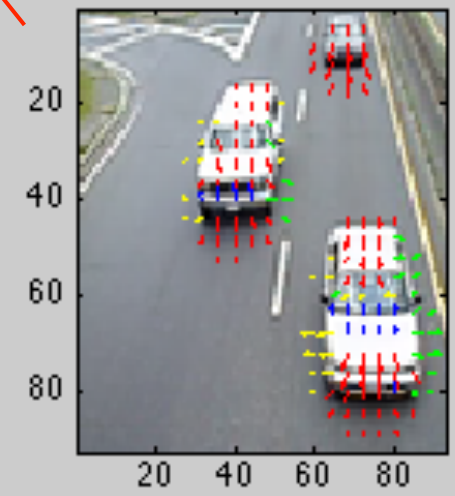


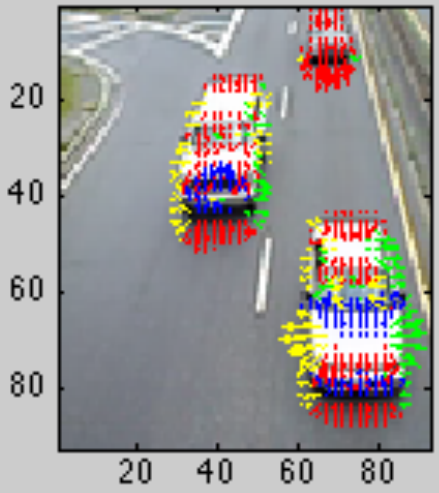
Level 4

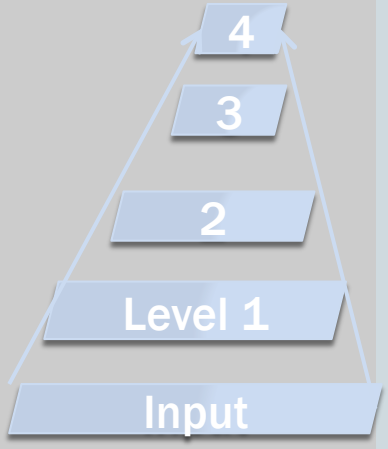
Level 3

Level 2

Frame:36 Input Dir:traffic.mj2 Grouping[]







What's Next?

- Implementation in Soar
- Expansion to Multiple representations per patch
- Application of Soars Learning

- Grouping over spatially distant groups (looking outside original patch proximity)

Soar Rules

Working Memory (WM)

(S1 ^Group G1)

(G1 ^Flow F1)

(G1 ^Flow F2)

…

(F1 ^mag 1
^H 707
^V -707)

(F2 ^mag 1
^H 316
^V -949)

…

```
#Preform dot product in productions
sp {elaborate*Dot*Product
  (state <s> ^name HierAgent
          ^Group <grp>
          ^GroupDot <grpDot>)
  (<grp> ^Flow <a1>
       ^Flow <a2>)
  (<a1> ^H <h1>
       ^V <v1>)
  (<a2> ^H <h2>
       ^V <v2>)
-->
  (<grpDot> ^Group <grp2>)
  (<grp2> ^element <a1>
        ^element <a2>
        ^dot (+ (* <h1> <h2>)
(* <v1> <v2>)))
}
```

```
#Mark Valid thresholds
sp {elaborate*Threshold
  (state <s> ^name HierAgent
          ^Threshold <thresh>
          ^GroupDot <grpDot>)
  (<grpDot> ^Group <grp2>)
  (<grp2> ^dot > <thresh>
        ^element <a1>
        ^element <a2> <>
<a1>)
-->
  (<grp2> ^valid 1)
}
```

Conclusion So Far

- Elaborations Rock
- Hardware realization can allow for exploiting parallelism to efficiently preform perceptual processing in Soar
- Hardware implementation could tip the balance between architectural vs knowledge based realization of cognitive capabilities
- What can you do with elaborations if they were truly parallel?