

Does the Brain do Inverse Graphics?

Geoffrey Hinton, Alex Krizhevsky, Navdeep Jaitly,
Tijmen Tieleman & Yichuan Tang

Department of Computer Science
University of Toronto

The representation used by the neural nets that work best for recognition (Yann LeCun)

- Convolutional neural nets use multiple layers of feature detectors that have local receptive fields and shared weights.
- The feature extraction layers are interleaved with sub-sampling layers that throw away information about precise position in order to achieve some translation invariance.

Combining the outputs of replicated features

- Get a small amount of translational invariance at each level by averaging some neighboring replicated detectors to give a single output to the next level.
 - This reduces the number of inputs to the next layer of feature extraction, thus allowing us to have many more different types of feature at the next layer.
 - Taking the maximum activation of some neighboring detectors works slightly better.

Why convolutional neural networks are doomed

- Convolutional nets are doomed for two reasons:
 - Sub-sampling loses the precise spatial relationships between higher-level parts such as a nose and a mouth. The precise spatial relationships are needed for identity recognition
 - But overlapping the sub-sampling pools mitigates this.
 - They cannot extrapolate their understanding of geometric relationships to radically new viewpoints.

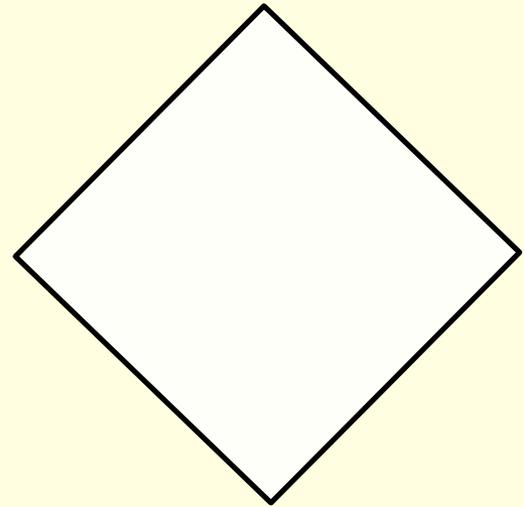
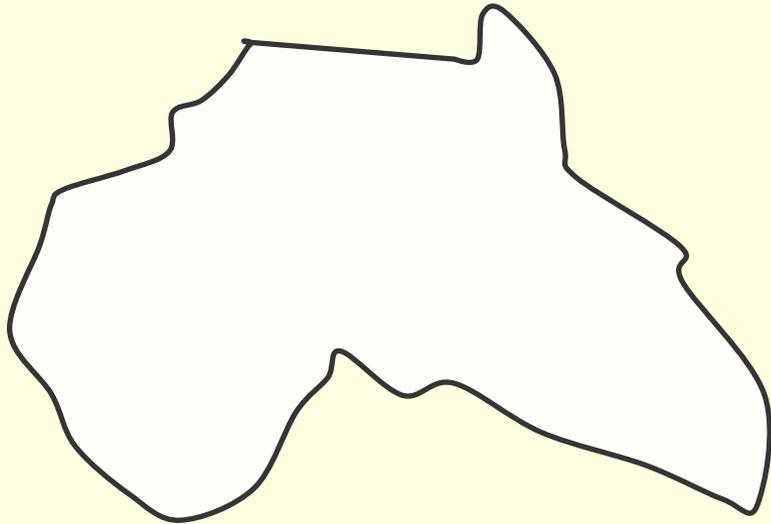
Equivariance vs Invariance

- Sub-sampling tries to make the neural activities invariant for small changes in viewpoint.
 - This is a silly goal, motivated by the fact that the final label needs to be viewpoint-invariant.
- Its better to aim for equivariance: Changes in viewpoint lead to corresponding changes in neural activities.
 - In the perceptual system, its the weights that code viewpoint-invariant knowledge, not the neural activities.

What is the right representation of images?

- Computer vision is inverse computer graphics, so the higher levels of a vision system should look like the representations used in graphics.
- Graphics programs use hierarchical models in which spatial structure is modeled by matrices that represent the transformation from a coordinate frame embedded in the whole to a coordinate frame embedded in each part.
 - These matrices are totally viewpoint invariant.
 - This representation makes it easy to compute the relationship between a part and the retina from the relationship between a whole and the retina.
 - Its just a matrix multiply!

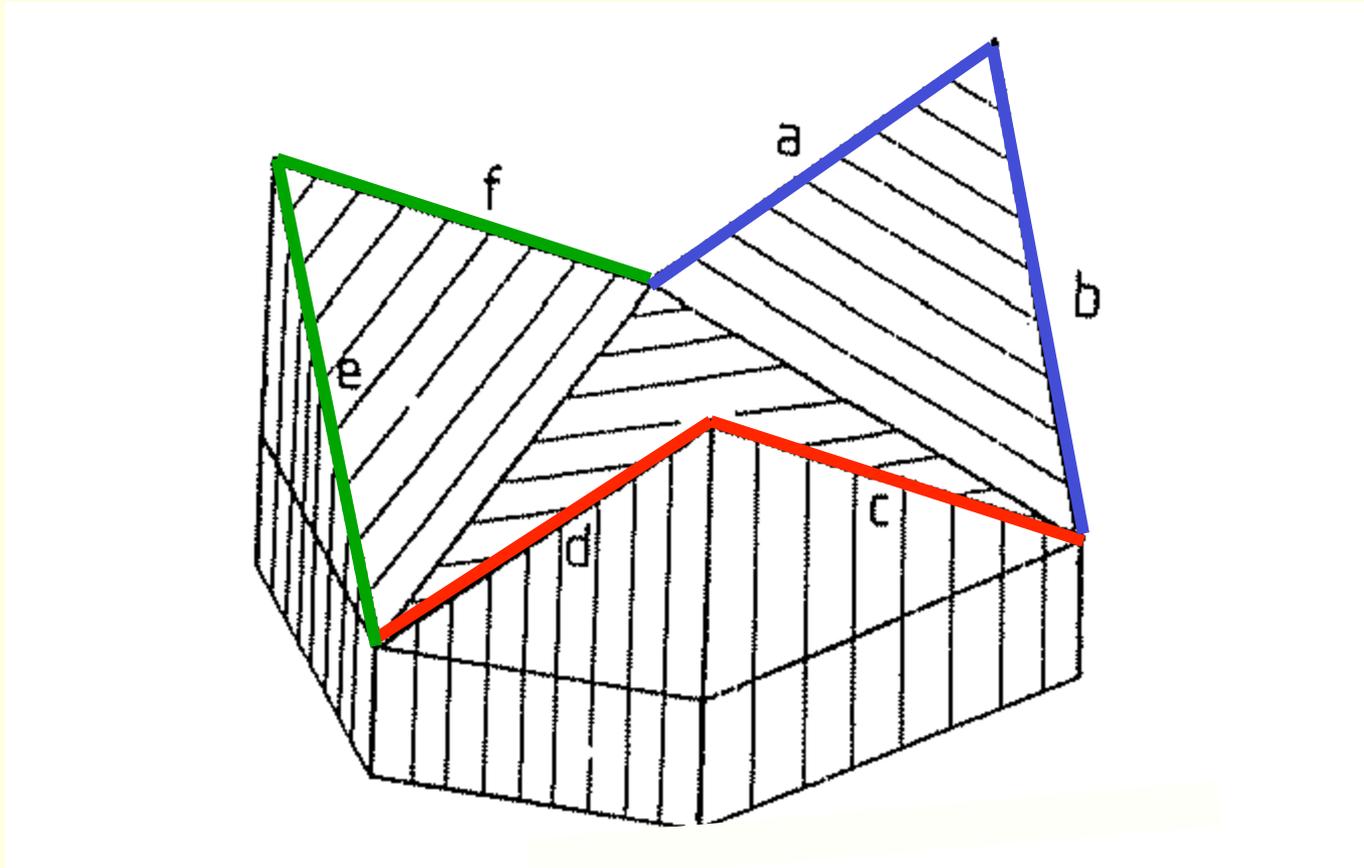
Some psychological evidence that our visual systems impose coordinate frames in order to represent shapes (after Irvin Rock)



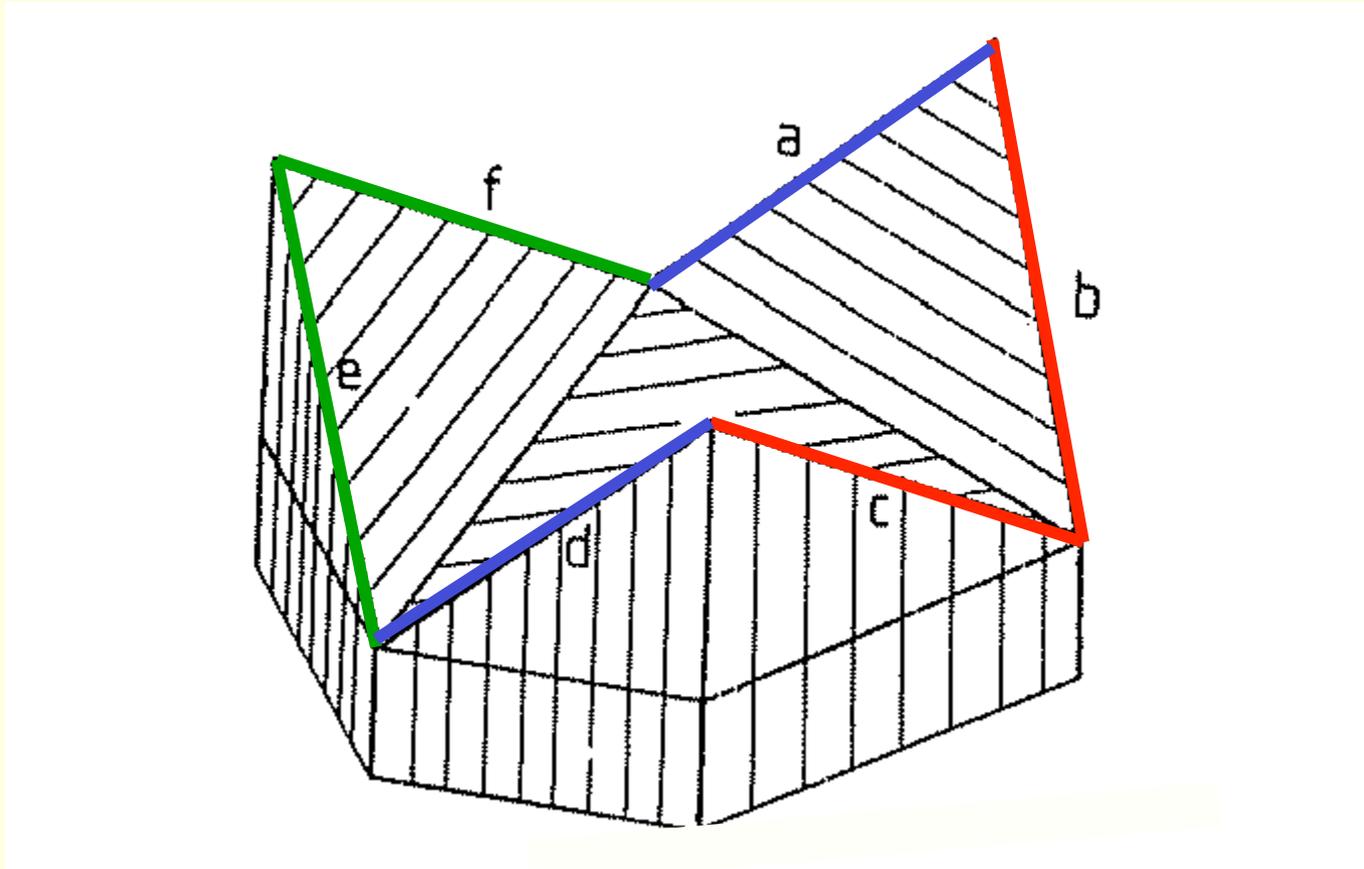
The cube task

- You can all imagine a wire-frame cube.
- Now imagine the cube standing on one corner so that body-diagonal from that corner, through the center of the cube to the opposite corner is vertical.
- Where are the other corners of the cube?

An arrangement of 6 rods



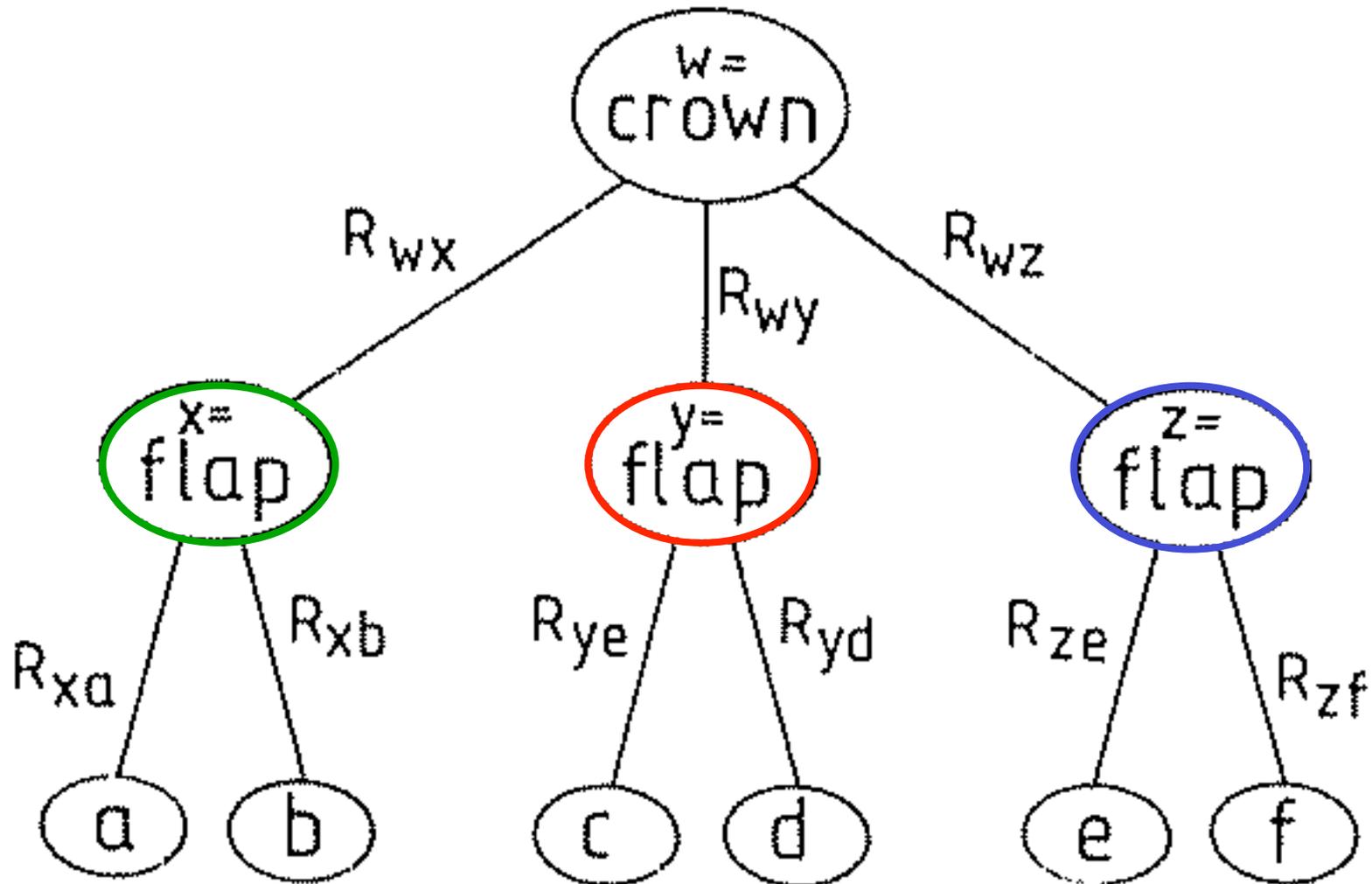
A different percept of the 6 rods



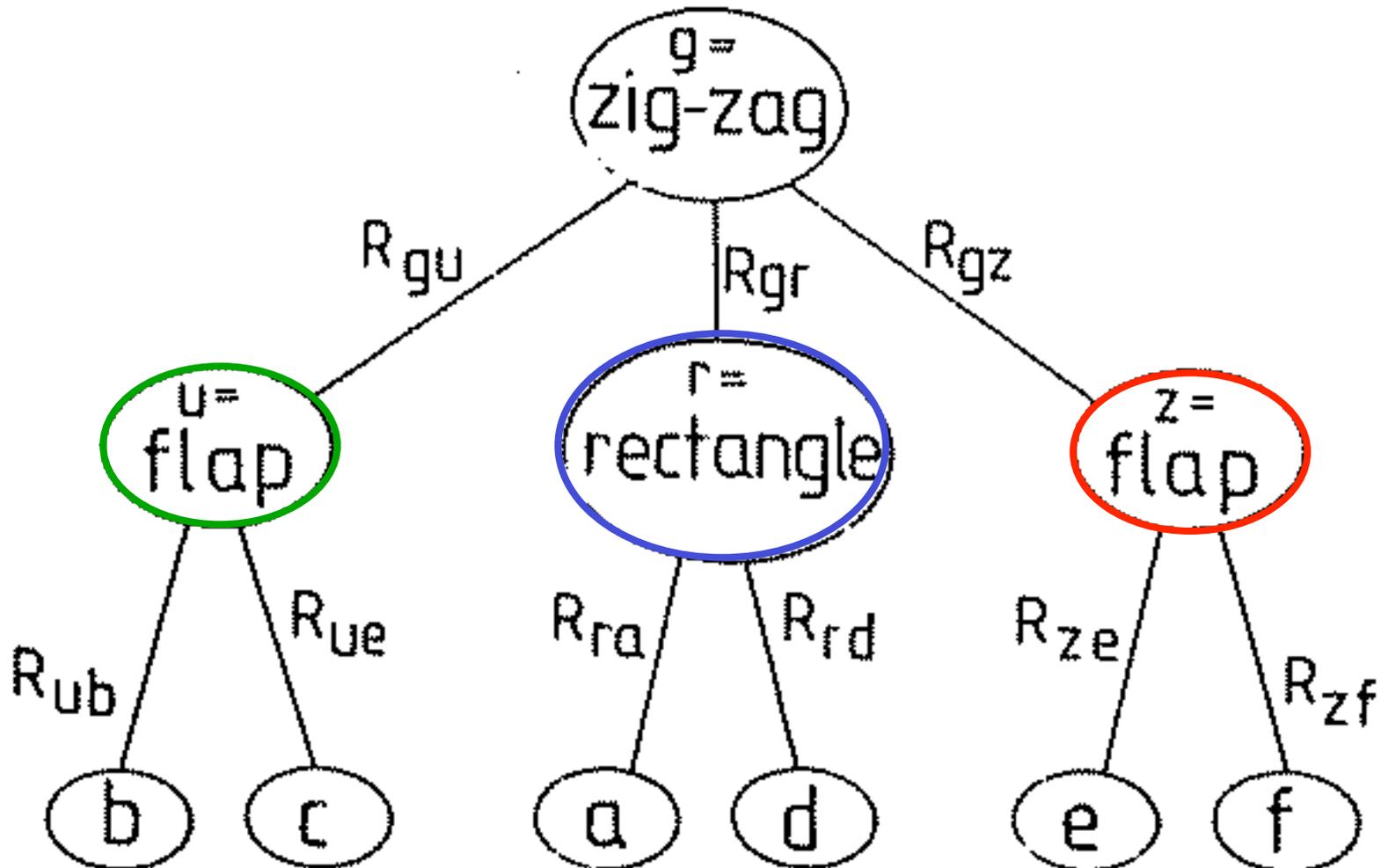
Alternative representations

- The very same arrangement of rods can be represented in quite different ways.
 - Its not like the Necker cube where the alternative percepts disagree on depth.
- The alternative percepts do not disagree, but they make different facts obvious.
 - In the zig-zag representation it is obvious that there is one pair of parallel edges.
 - In the crown representation there are no obvious pairs of parallel edges because the edges do not align with the intrinsic frame of any of the parts.

A structural description of the “crown” formed by the six rods

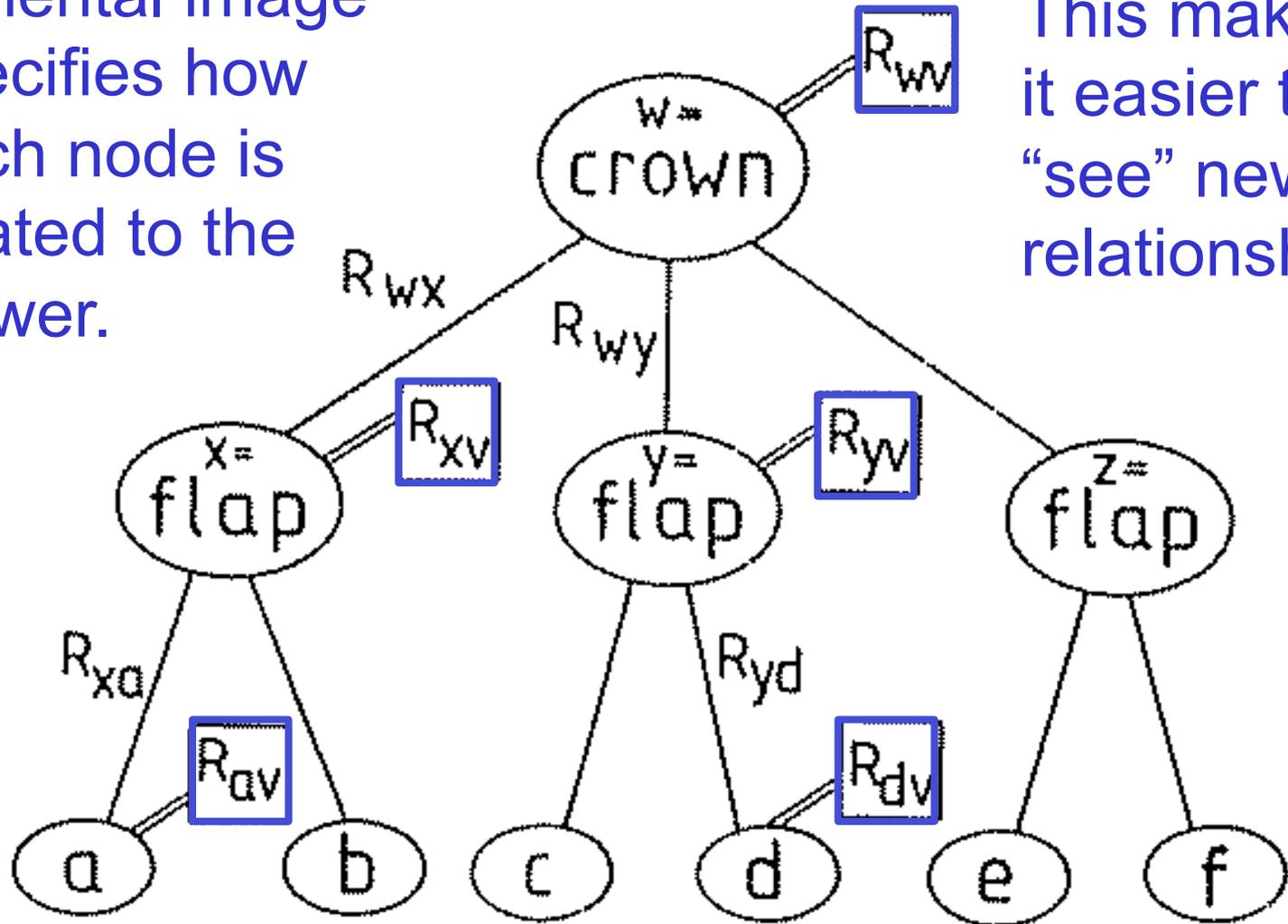


A structural description of the “zig-zag”



A mental image of the crown

A mental image specifies how each node is related to the viewer.



This makes it easier to “see” new relationships

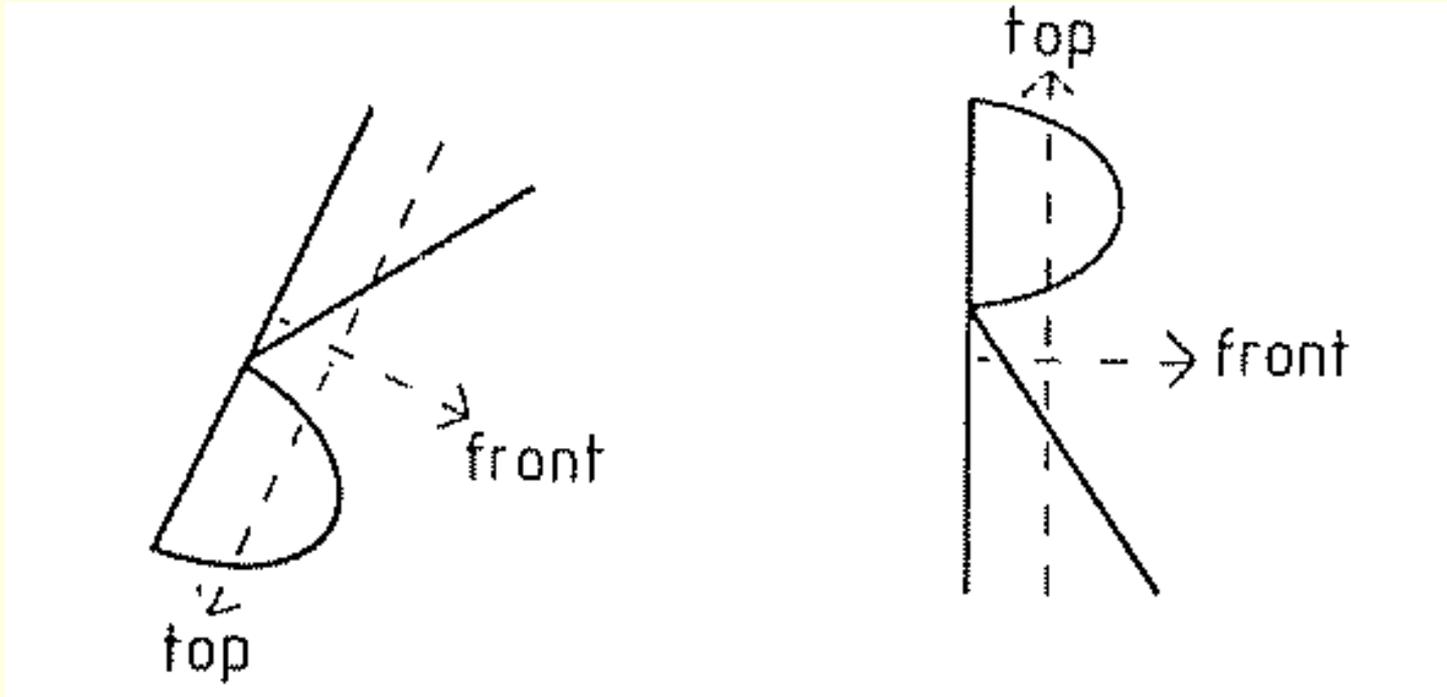
A task that forces you to form a mental image

- Imagine making the following journey:
 - You go a mile east.
 - Then you go a mile north.
 - Then you go a mile east again.
- What is the direction back to your starting point?

Analog operations on structural descriptions

- We can imagine the “petals” of the crown all folding upwards and inwards.
 - What is happening in our heads when we imagine this continuous transformation?
 - Why are the easily imagined transformations quite different for the two different structural descriptions?
- One particular continuous transformation called “mental rotation” was intensively studied by Roger Shepard and other psychologists in the 1970’s
 - Mental rotation really is continuous: When we are halfway through performing a mental rotation we have a mental representation at the intermediate orientation.

Mental rotation: More evidence for coordinate frames



We perform mental rotation to decide if the tilted R has the correct handedness, not to recognize that it is an R.

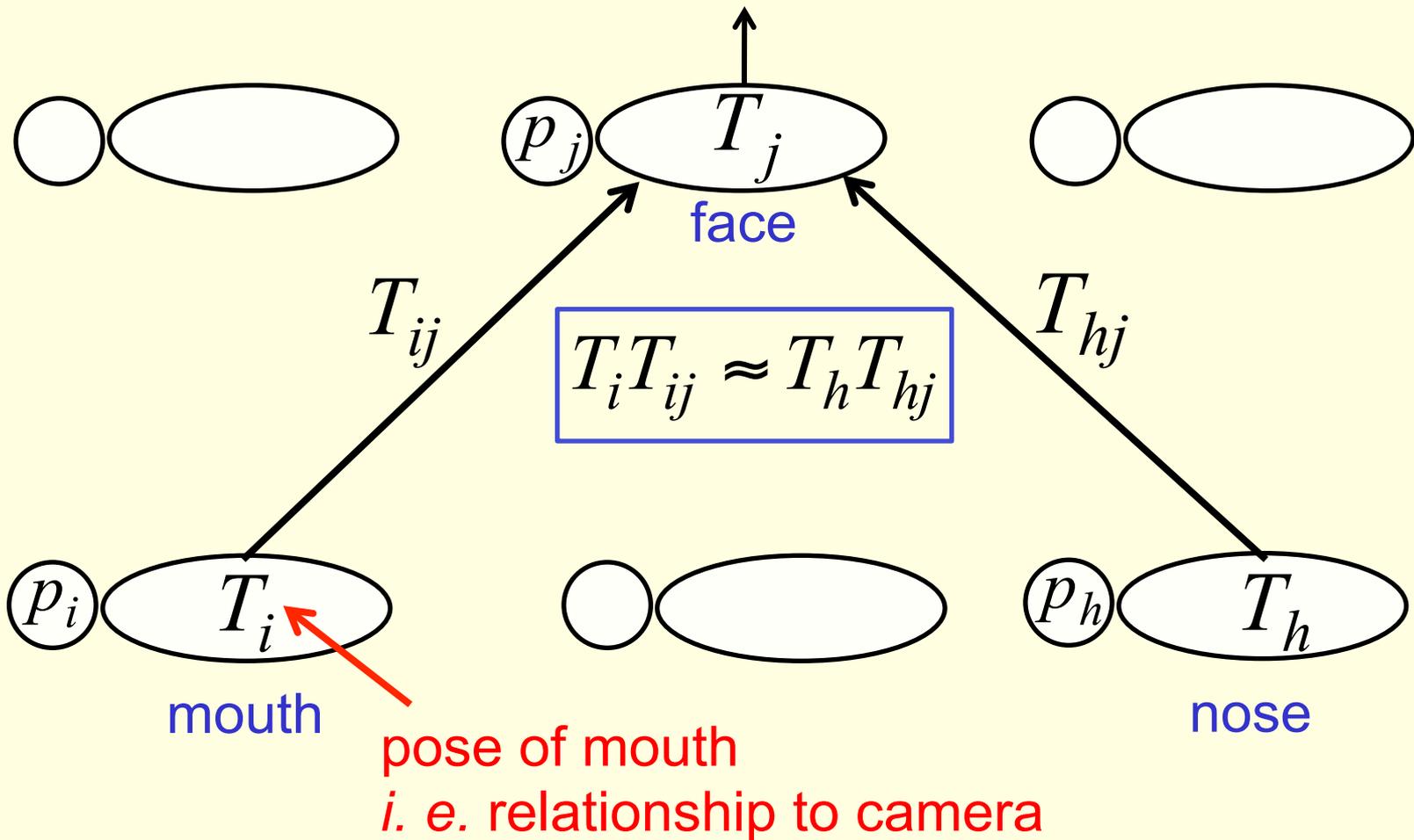
But why do we need to do mental rotation to decide handedness?

What mental rotation achieves

- It is very difficult to compute the handedness of a coordinate transformation by looking at the individual elements of the matrix.
 - The handedness is the sign of the determinant of the matrix relating the object to the viewer.
 - This is a high-order parity problem.
- To avoid this computation, we rotate to upright *preserving handedness*, then we look to see which way it faces when it is in its familiar orientation.
- If we had individual neurons that represented a whole pose, we would not have a problem with identifying handedness, because these neurons would have a handedness.

Two layers in a hierarchy of parts

- A higher level visual entity is present if several lower level visual entities can agree on their predictions for its pose.



A crucial property of the pose vectors

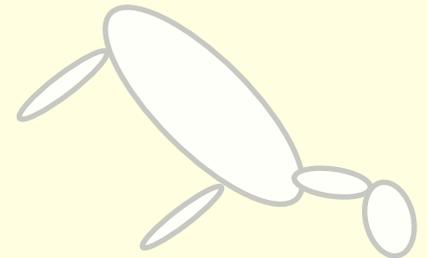
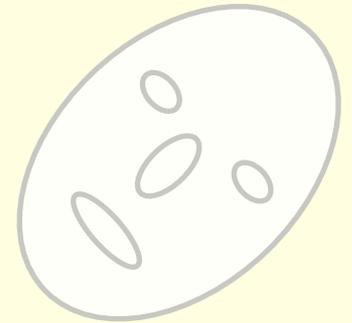
- They allow spatial transformations to be modeled by linear operations.
 - This makes it easy to learn a hierarchy of visual entities.
 - It makes it easy to generalize across viewpoints.
- The invariant geometric properties of a shape are in the weights, not in the activities.
 - The activities are equivariant.

Extrapolating shape recognition to very different sizes, orientations and positions that were not in the training data.

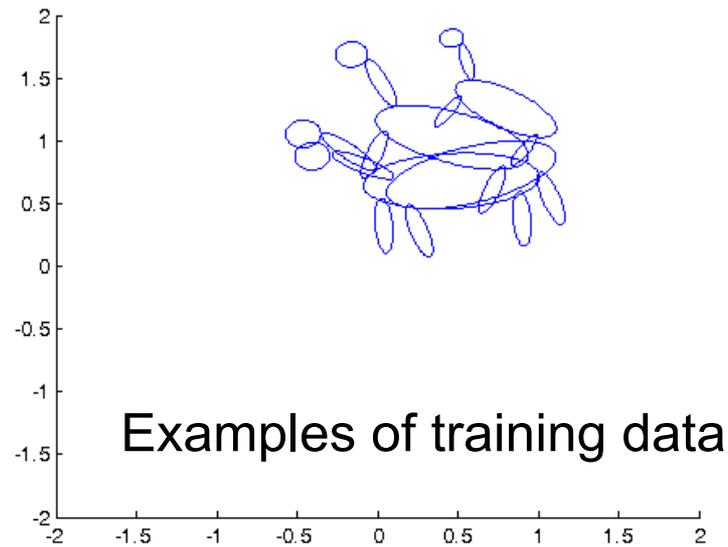
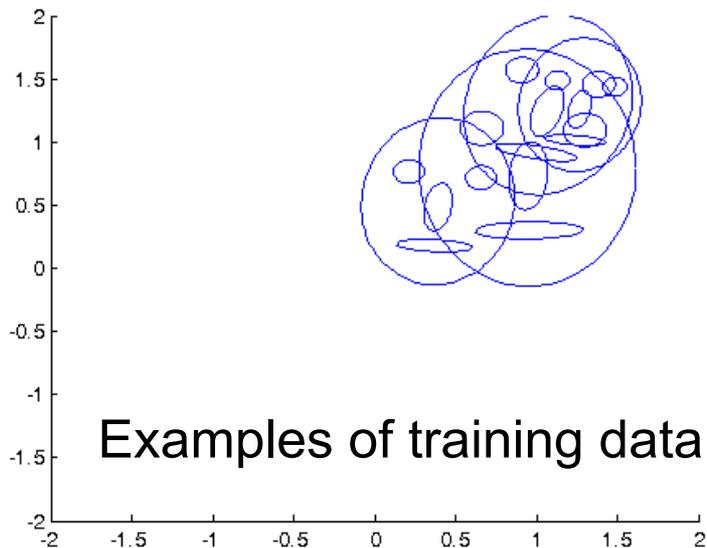
- Current wisdom (e.g. Andrew Ng's talk)
 - Learn different models for very different viewpoints.
 - Get a **lot** more training data to cover all significantly different viewpoints.
- The only reasonable alternative?
 - Massive extrapolation requires linear models.
 - So capture the knowledge of the geometry of the shape in a linear model.

A toy example of what we could do if we could reliably extract the poses of parts of objects

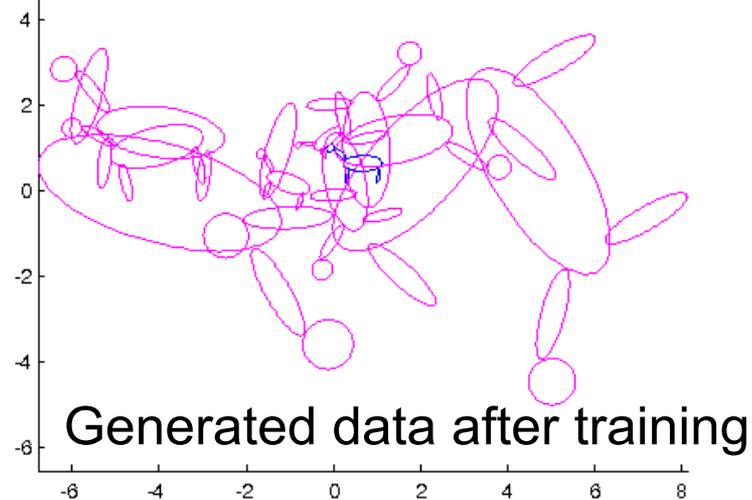
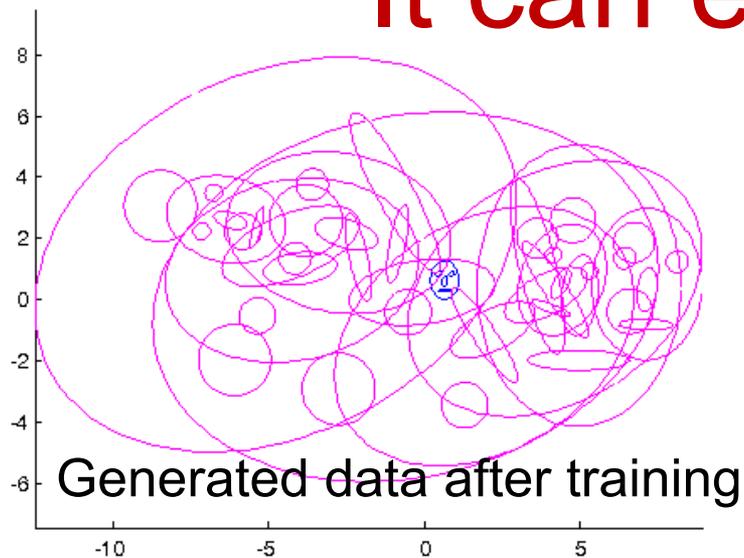
- Consider images composed of five ellipses.
 - The shape is determined entirely by the spatial relations between the ellipses because all parts have the same shape.
- Can we sets of spatial relationships from data that contains several different shapes?
 - Can we generalize far beyond the range of variation in the training examples?
 - Can we learn without being told which ellipse corresponds to which part of a shape?



Examples of two shapes (Yichuan Tang)

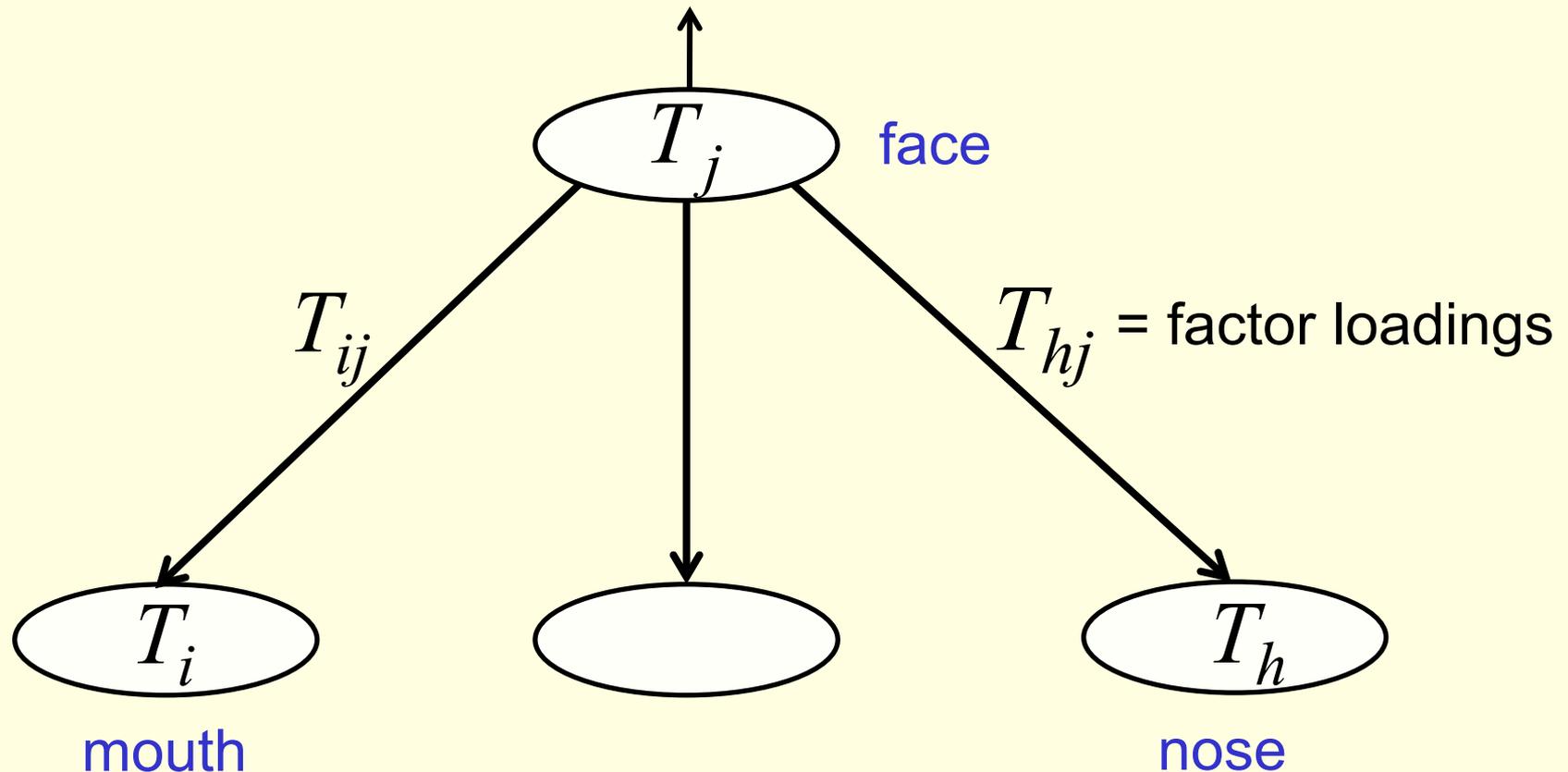


It can extrapolate!



Learning one shape using factor analysis

- The pose of the whole can predict the poses of all the parts. This generative model is easy to fit using the EM algorithm for factor analysis.



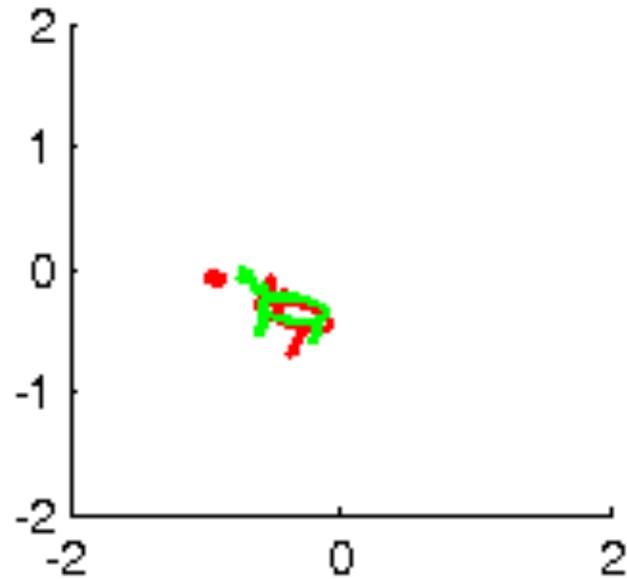
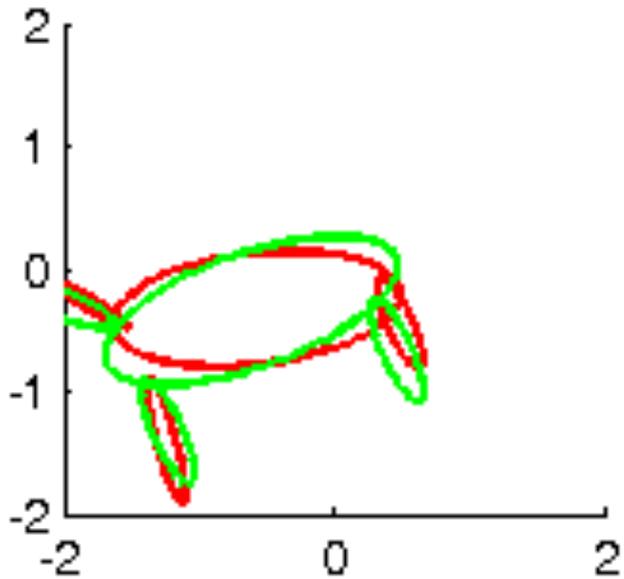
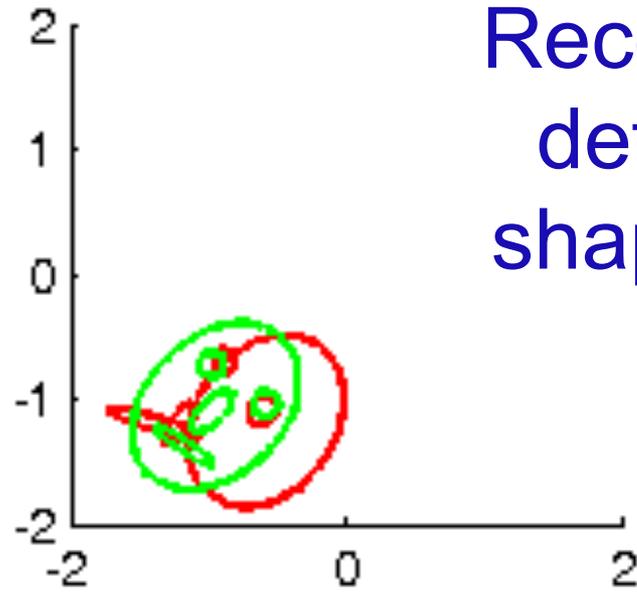
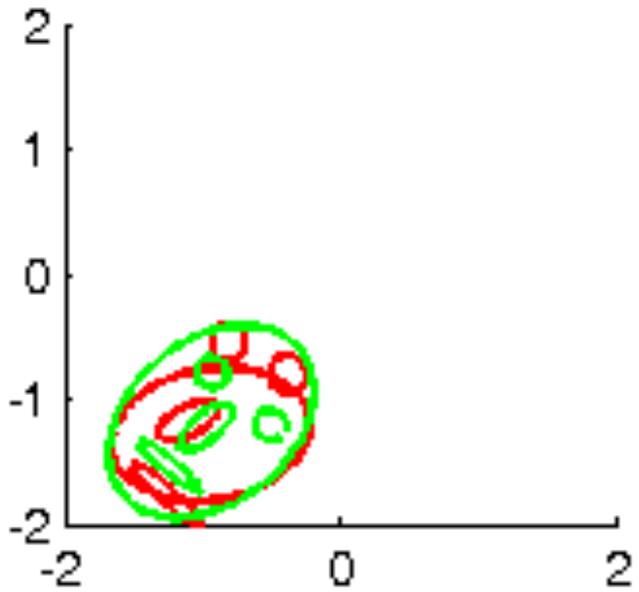
Learning the right assignments of the ellipses to the parts of a shape.

- If we do not know how to assign the ellipses to the parts of a known shape we can try many different assignments and pick the one that gives the best reconstruction.
 - Then we assume that is the correct assignment and use that assignment for learning.
- This quickly leads to models that have strong opinions about the correct assignment.
- We could eliminate most of the search by using a feed-forward neural net to predict the assignments.

Fitting a mixture of shapes

- We can use a mixture of factor analysers (MFA) to fit images that contain many different shapes so long as each image only contains one example of one shape.

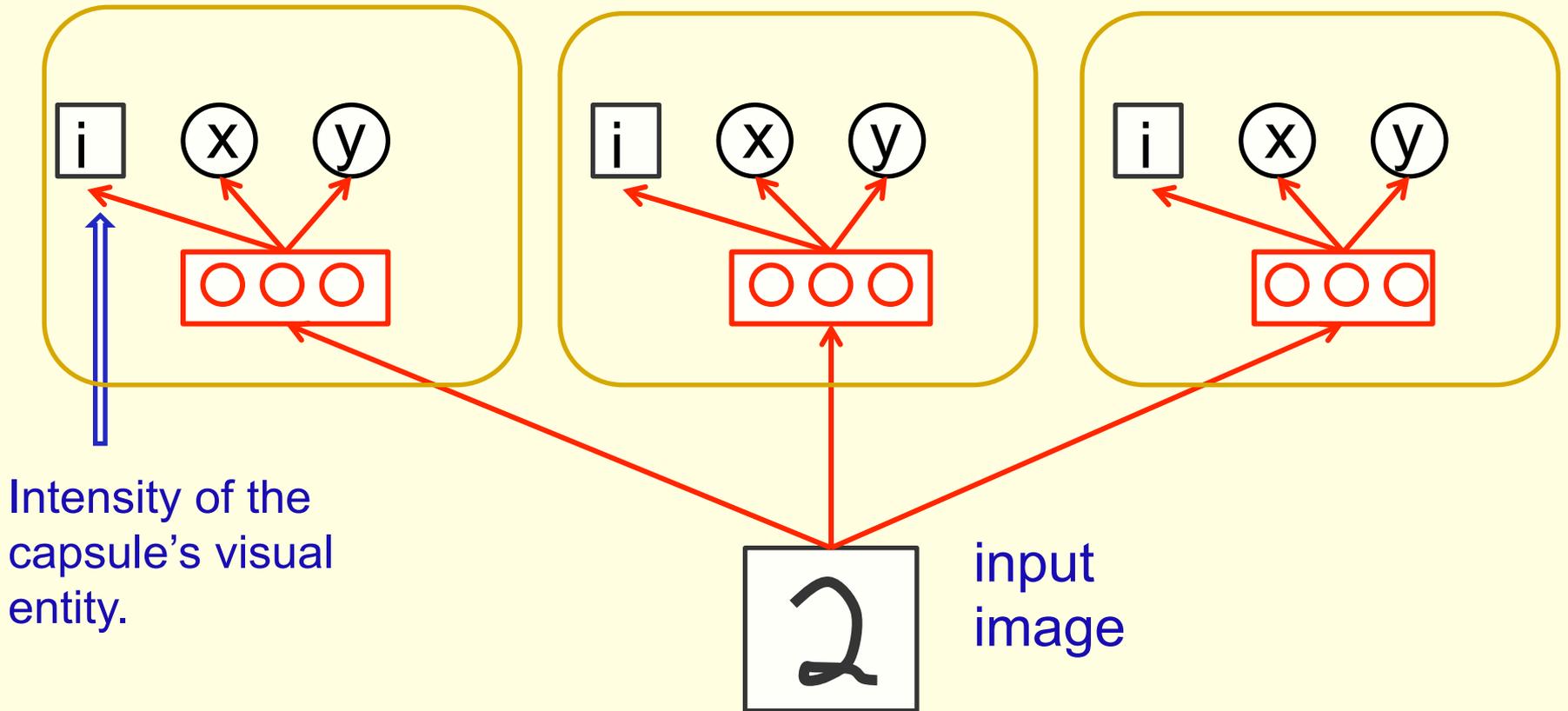
Recognizing deformed shapes with MFA



A big problem

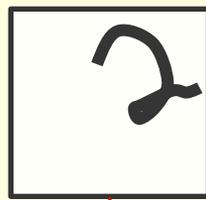
- How do we get from pixels to the first level parts that output explicit pose parameters?
 - We do not have any labeled data.
 - This “de-rendering” stage has to be very non-linear.

A picture of three capsules



Extracting pose information by using a domain specific decoder (Navdeep Jaitly & Tijmen Tieleman)

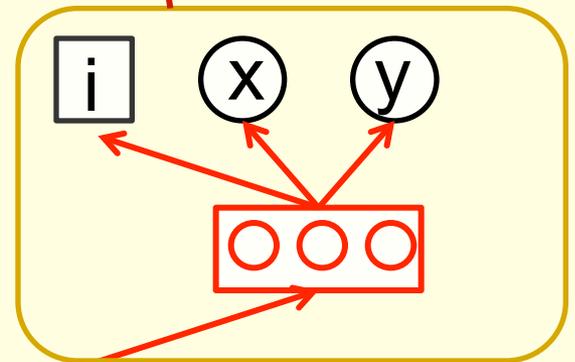
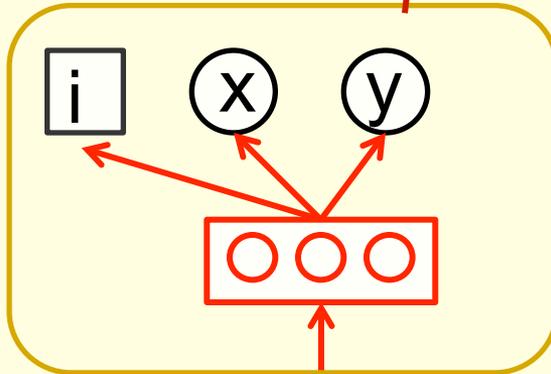
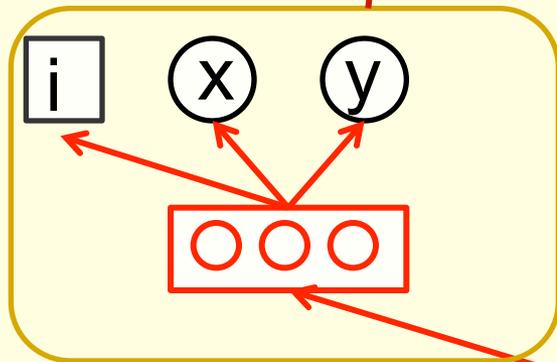
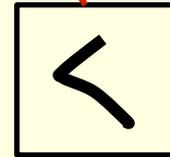
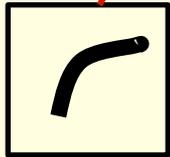
- The idea is to define a simple decoder that produces an image by adding together contributions from each capsule.
- Each capsule learns a fixed “template” that gets intensity-scaled and translated differently for reconstruction each image.
- The encoder must learn to extract the appropriate intensity and translation for each capsule from the input image.



output image

Decoder: Add together intensity-scaled and translated contributions from each capsule.

learned template



Train encoder using backpropagation



The templates learned by the autoencoder

Each template is multiplied by a case-specific intensity and translated by a case-specific Δx , Δy

Then it is added to the output image.



image

reconstruction

recon. error

capsule 1

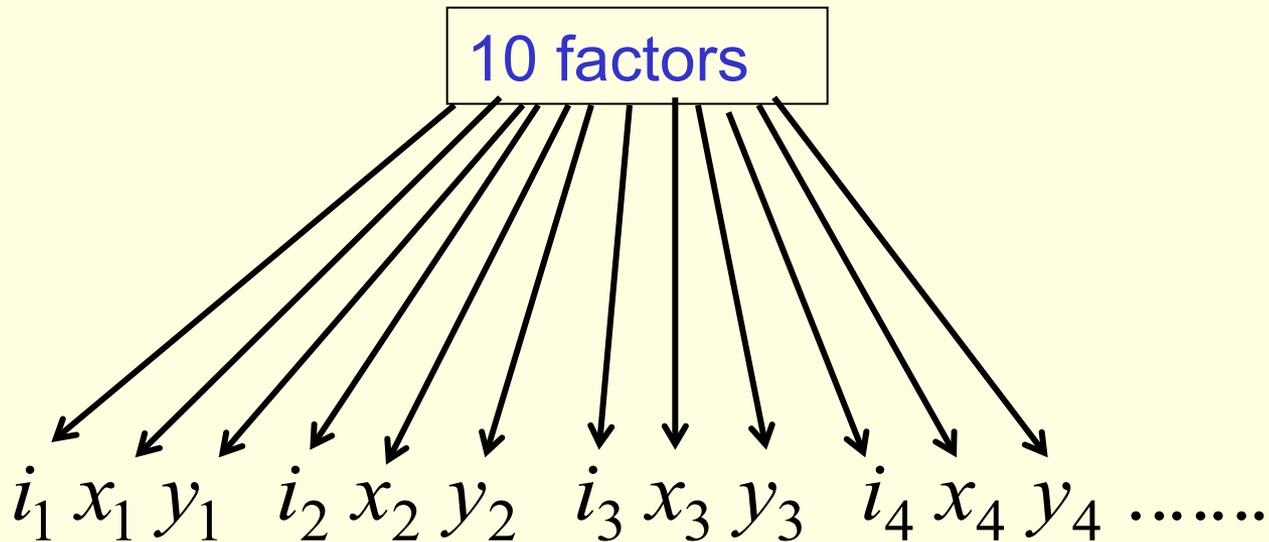
capsule 2

capsule 3

capsule 4

7	7											
2	2											
6	6											
9	9											
3	3											
8	8											
1	1											
6	6											
5	5											

Modeling the capsule outputs with a factor analyser



We learn a mixture of 10 factor analysers on unlabeled data. What do the means look like?

Means discovered by a mixture of factor analysers

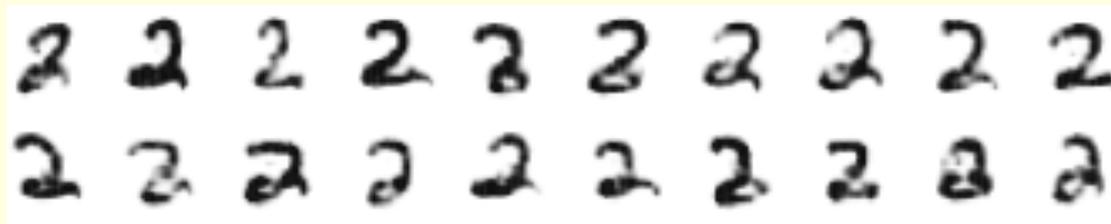


directly on pixels



on outputs of capsules

mean - 2σ of each factor



mean + 2σ of each factor

Another simple way to learn the lowest level parts

- Use pairs of images that are related by a known coordinate transformation
 - *e.g.* a small translation of the image.
- We often have non-visual access to image transformations
 - *e.g.* When we make an eye-movement.
- Cats learn to see much more easily if they control the image transformations (Held & Hein)

Learning the lowest level capsules

- We are given a pair of images related by a known translation.

Step 1: Compute the capsule outputs for the first image.

- Each capsule uses its own set of “recognition” hidden units to extract the x and y coordinates of the visual entity it represents (and also the probability of existence)

Step 2: Apply the transformation to the outputs of each capsule

- Just add Δx to each x output and Δy to each y output

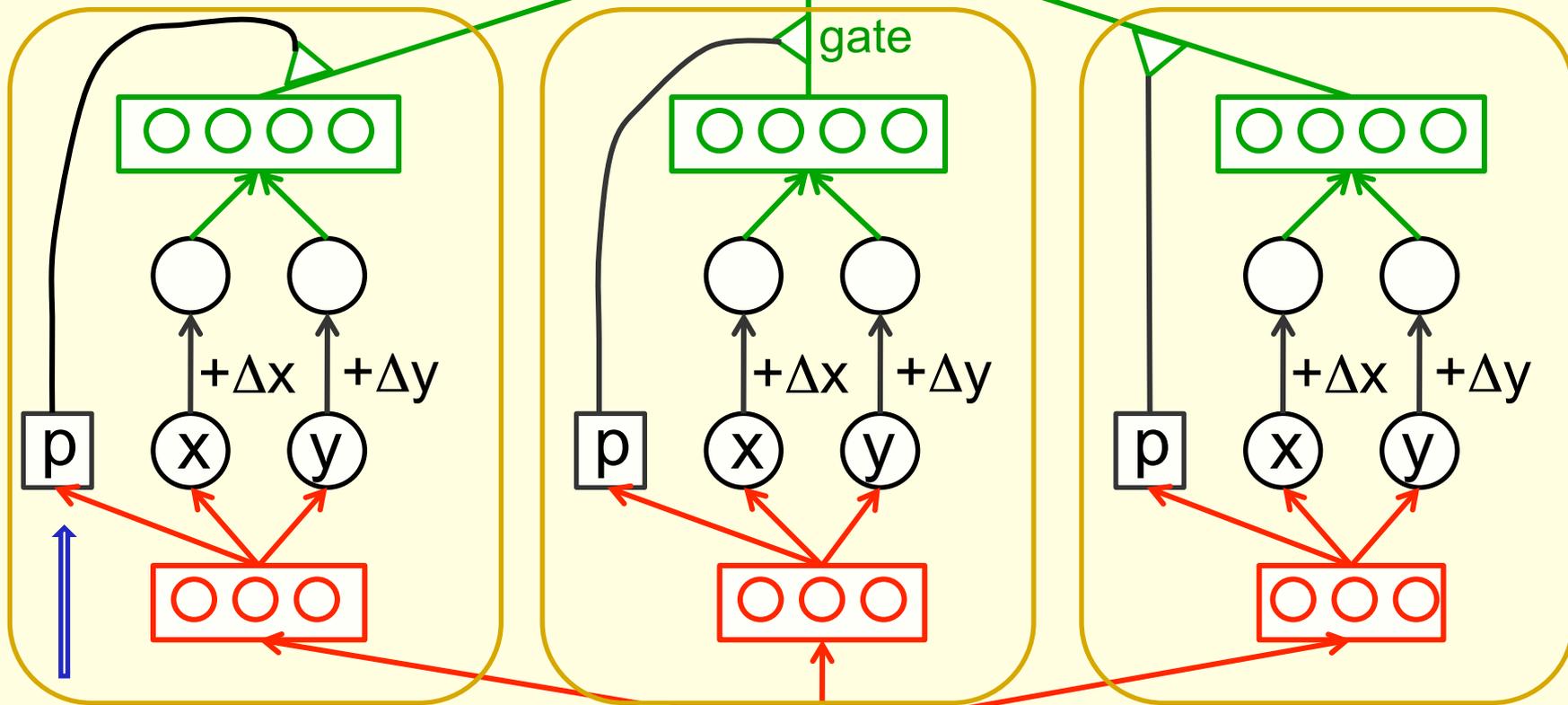
Step 3: Predict the transformed image from the transformed outputs of the capsules

- Each capsule uses its own set of “generative” hidden units to compute its contribution to the prediction.

actual
output



target
output



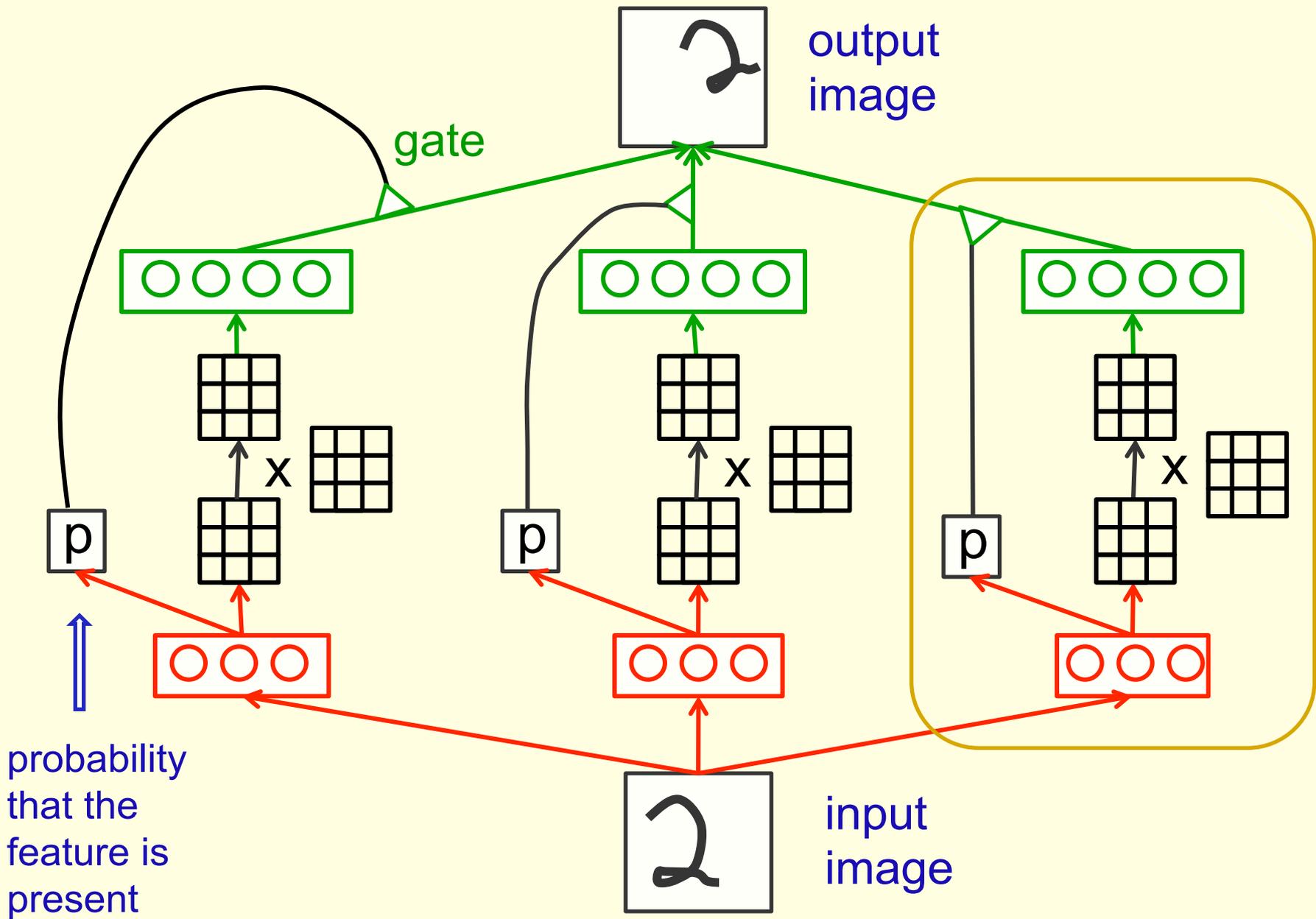
probability that
the capsule's
visual entity is
present



input
image

Why it has to work

- When the net is trained with back-propagation, the only way it can get the transformations right is by using x and y in a way that is consistent with the way we are using Δx and Δy .
- This allows us to force the capsules to extract the coordinates of visual entities **without having to decide what the entities are or where they are.**



Dealing with the three-dimensional world

- Use stereo images to allow the encoder to extract 3-D poses.
 - Using capsules, 3-D would not be much harder than 2-D if we started with 3-D pixels.
 - The loss of the depth coordinate is a separate problem from the complexity of 3-D geometry.
- At least capsules stand a chance of dealing with the 3-D geometry properly.
 - Convolutional nets in 3-D are a nightmare.

Dealing with 3-D viewpoint (Alex Krizhevsky)

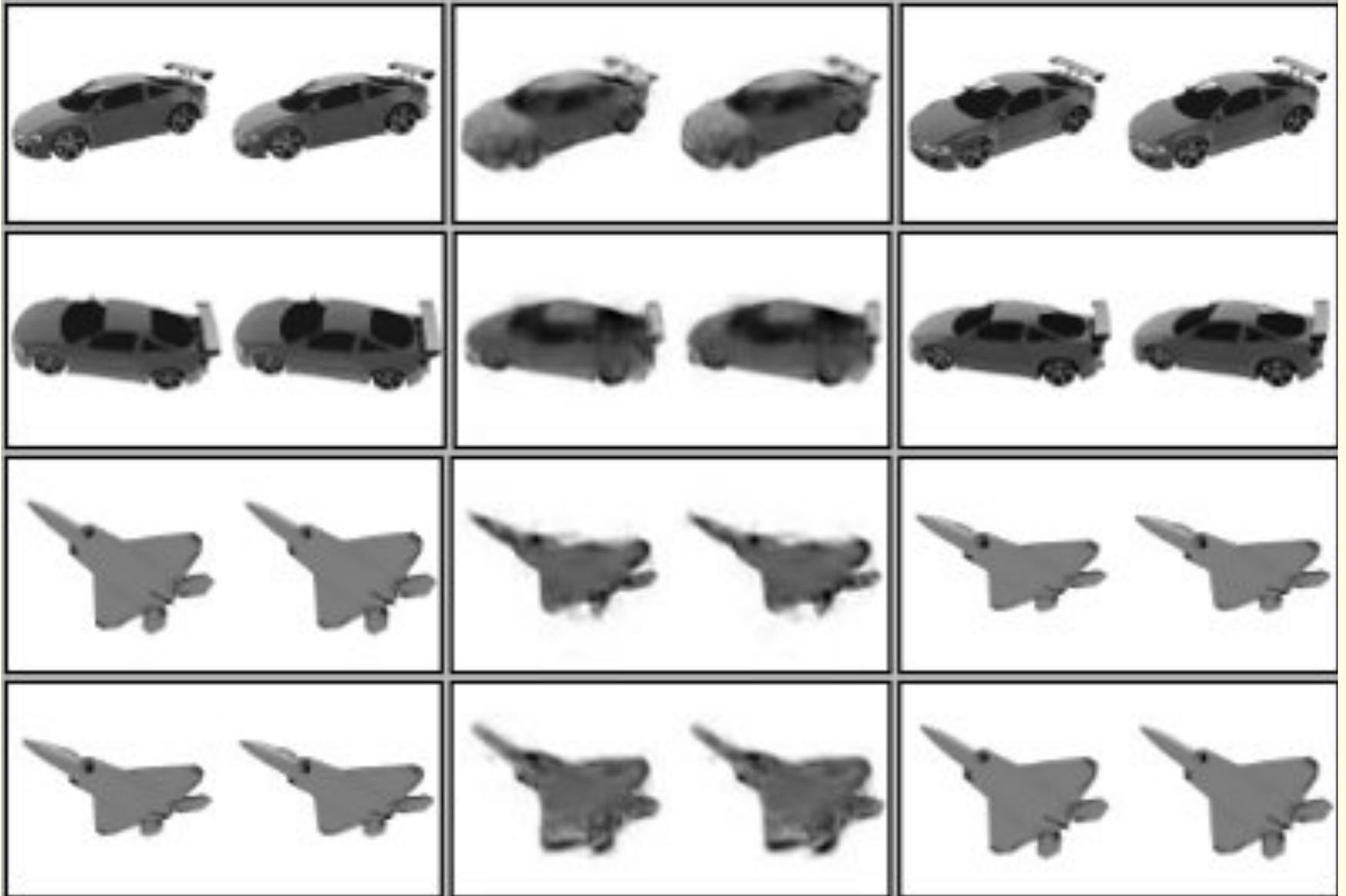
Input stereo pair

Output stereo pair

Correct output



It also works on test data



the end

- The work on transforming autoencoders is in a paper called “transforming autoencoders” on my webpage:

<http://www.cs.toronto.edu/~hinton/absps/transauto6.pdf>

- The work on “dropout” described in my previous lecture is in a paper on arxiv:

<http://arxiv.org/abs/1207.0580>

Relationship to Slow Feature Analysis

- Slow feature analysis uses a very primitive model of the dynamics.
 - It assumes that the underlying state does not change much.
 - This is a much weaker way of extracting features than using precise knowledge of the dynamics.
- Also, slow feature analysis only produces dumb features that do not output explicit instantiation parameters.

Relationship to a Kalman filter

- A linear dynamical system can predict the next observation vector.
 - But only when there is a linear relationship between the underlying dynamics and the observations.
- The extended Kalman filter assumes linearity about the current operating point. *It's a fudge.*
- Transforming autoencoders use non-linear recognition units to map the observation space to the space in which the dynamics is linear. Then they use non-linear generation units to map the prediction back to observation space
 - This is a much better approach than the extended Kalman filter, especially when the dynamics is known.