Low-level vision: shading, paint, and texture

Bill Freeman

October 27, 2008

Why shading, paint, and texture matters in object recognition

- We want to recognize objects independently from
  - surface colorings
  - lighting
  - surface texture
- One approach: learn appearance-based models of objects, spanning the space of all possible
- Alternate approach: develop bottom-up processing to separate shading from paint from texture. Hence, we study those issues today.

Separating shading from paint

Separating shading from paint

- From a single image:
  - identify all-shading versus all-paint
  - locally separate shading from paint
- From a sequence of images:
  - separate stable from varying component
- From a stereo pair
  - separate shading, paint, occlusion.

Separating shading from paint

- From a single image:
  - identify all-shading versus all-paint
  - locally separate shading from paint
- From a sequence of images:
  - separate stable from varying component
- From a stereo pair
  - separate shading, paint, occlusion.

Shading

Paint

Untitled3

Bayesian model of surface perception

William T. Freeman
MERL, Mitsubishi Electric Res. Lab.
201 Broadway
Cambridge, MA 02139
freeman@merl.com

Paul A. Viola
Artificial Intelligence Lab
Massachusetts Institute of Technology
Cambridge, MA 02139
viola@ai.mit.edu

Abstract

Image intensity variations can result from several different object surface effects, including shading from 3-dimensional relief of the object, or paint on the surface itself. An essential problem in vision, which people solve naturally, is to attribute the proper physical cause, e.g. surface relief or paint, to an observed image. We addressed this problem with an approach combining psychophysical

8.50 x 11.00 in

survey instructions

Untitled5

Pretend that each of the following pictures is a photograph of work made by either a painter or a sculptor.

The painter could use paint, markers, air brushes, computer, etc., to make any kind of mark on a flat canvas. The paint had no 3-dimensionality; everything was perfectly flat.

The sculptor could make 3-dimensional objects, but could make no markings on them. She could mold, sculpt, and scrape her sculptures, but could not draw or paint. All the objects were made out of a uniform plaster material and were made visible by lighting and shading effects.

The subjects used a 5-point rating scale to indicate whether each image was made by the painter (P) or sculptor (S): S, S?, ?, P?, P.

x 11.00 in

survey responses

Untitled7

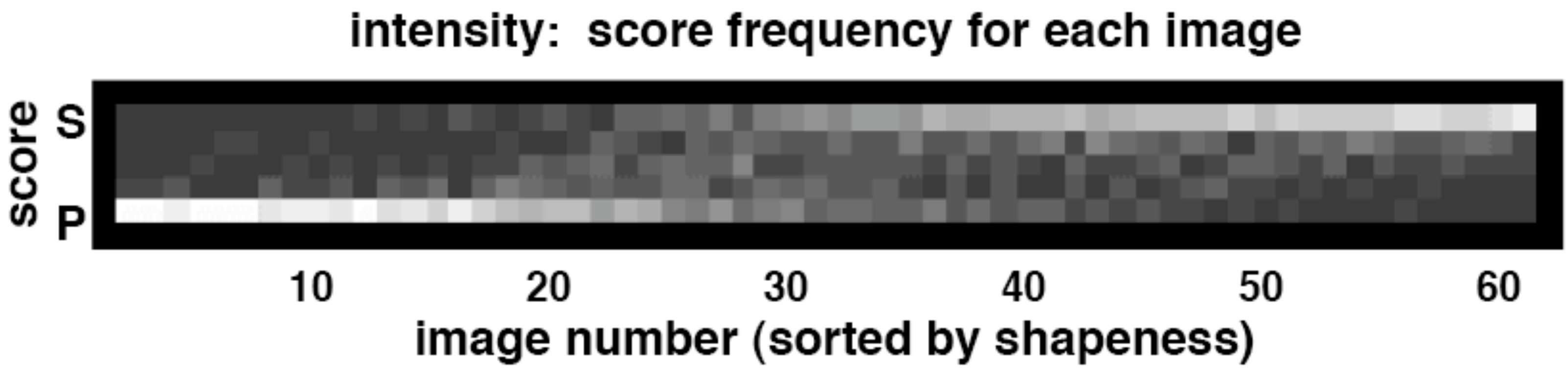
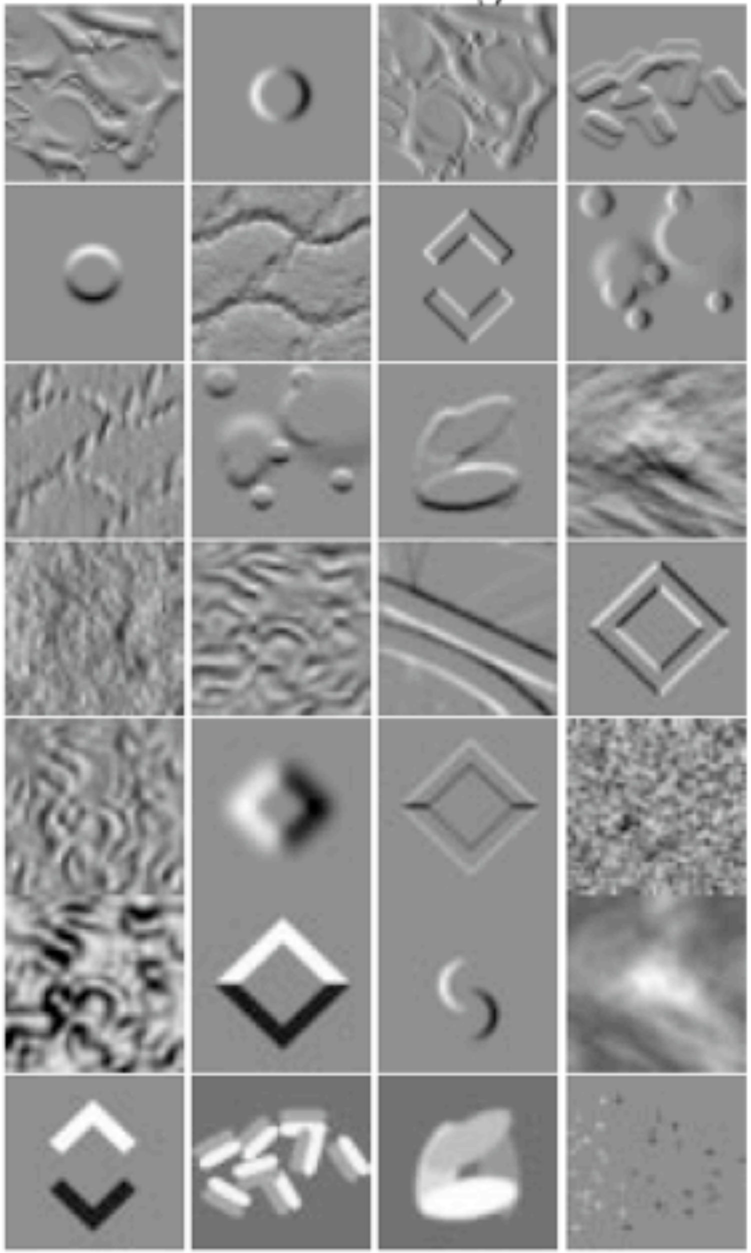


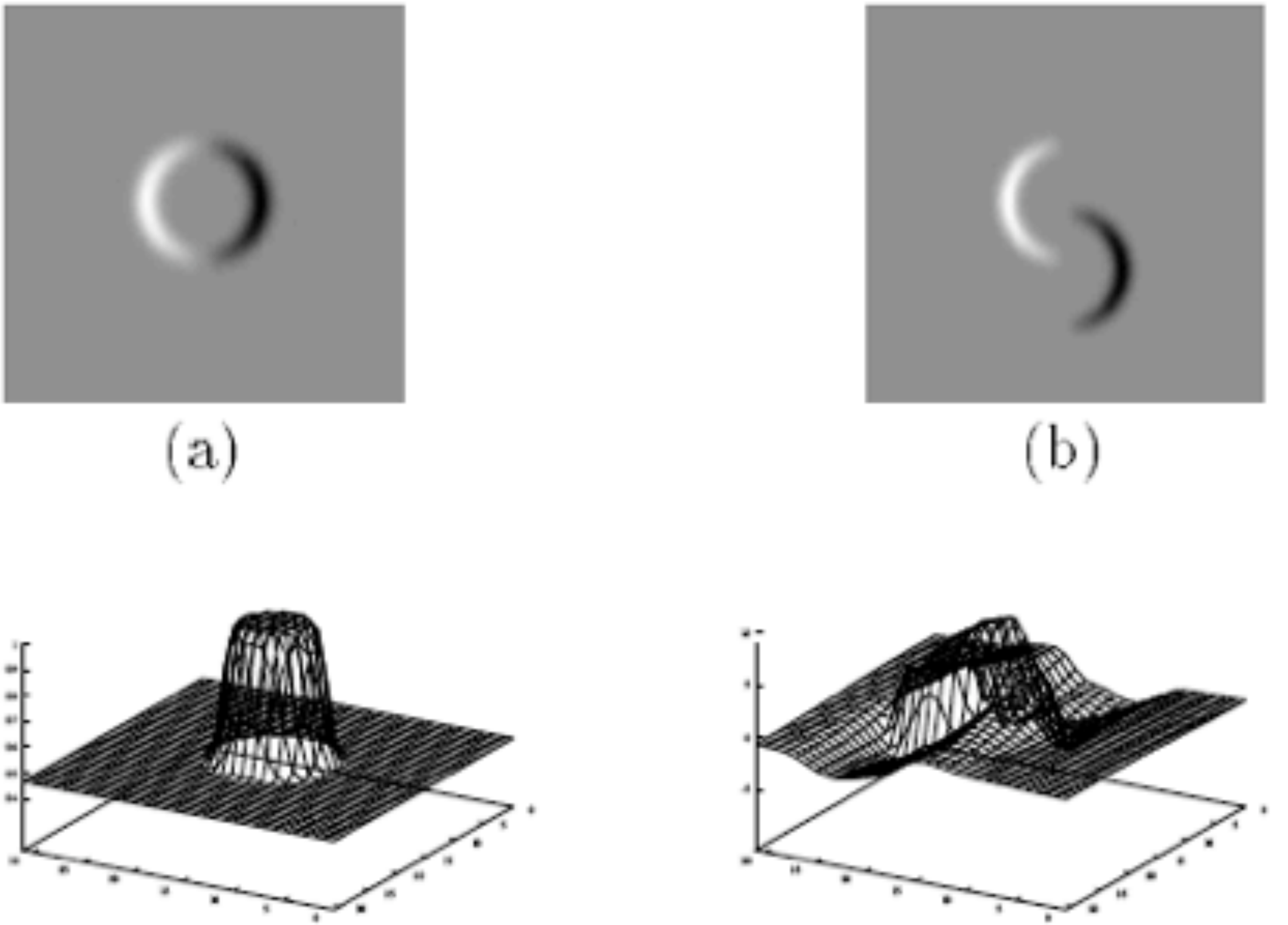
Figure 2: Histogram of survey responses. Intensity shows the number of responses of each score (vertical scale) for each image (horizontal, sorted in increasing order of shapeness).

00 in

Human Rankings



Evaluate the prior probability of the all-shape and all-reflectance explanations



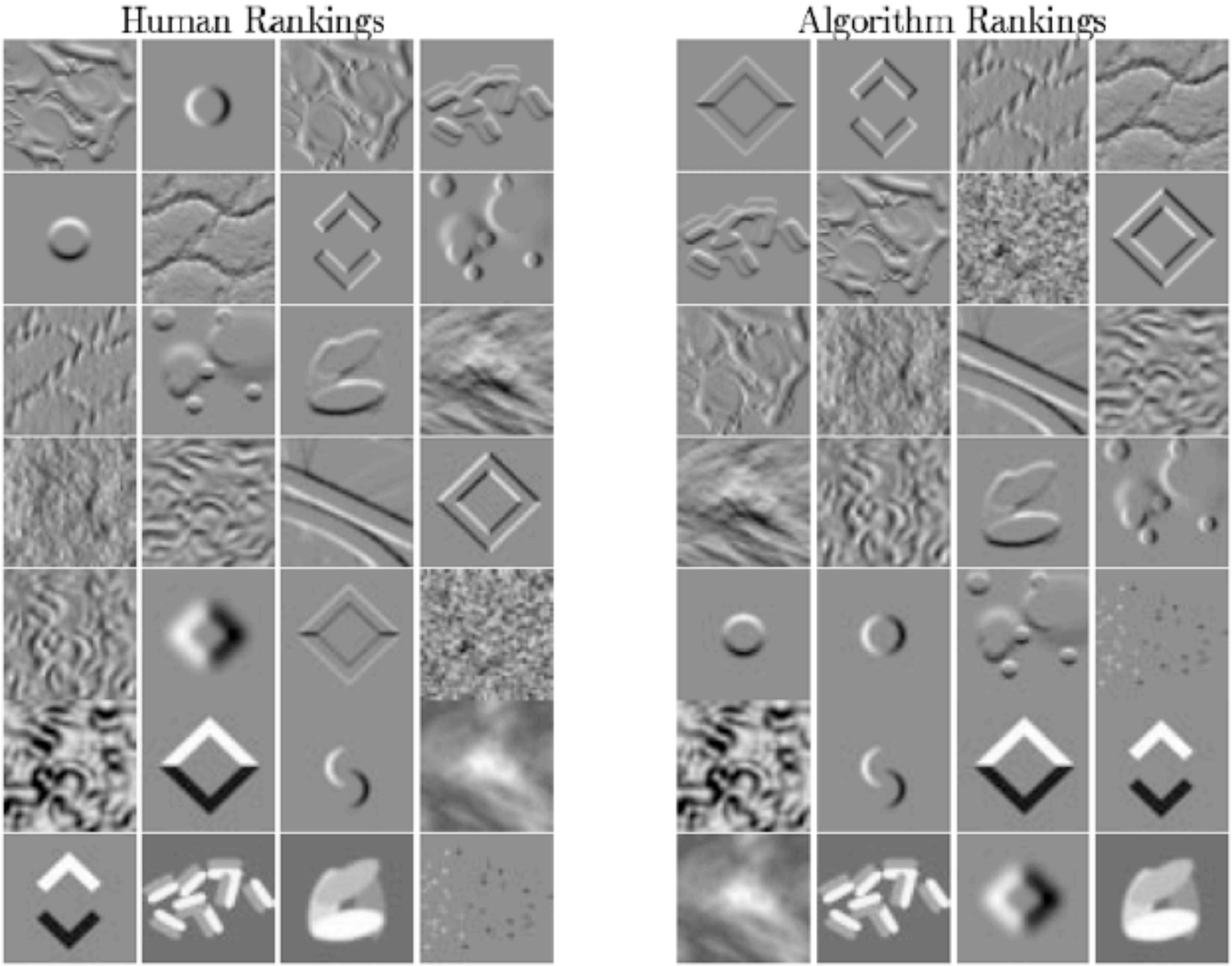


Figure 3: 28 of the 60 test images, arranged in decreasing order of subjects' shapeness ratings. Left: Subjects' rankings. Right: Algorithm's rankings.

algorithm performance vs people

Untitled11

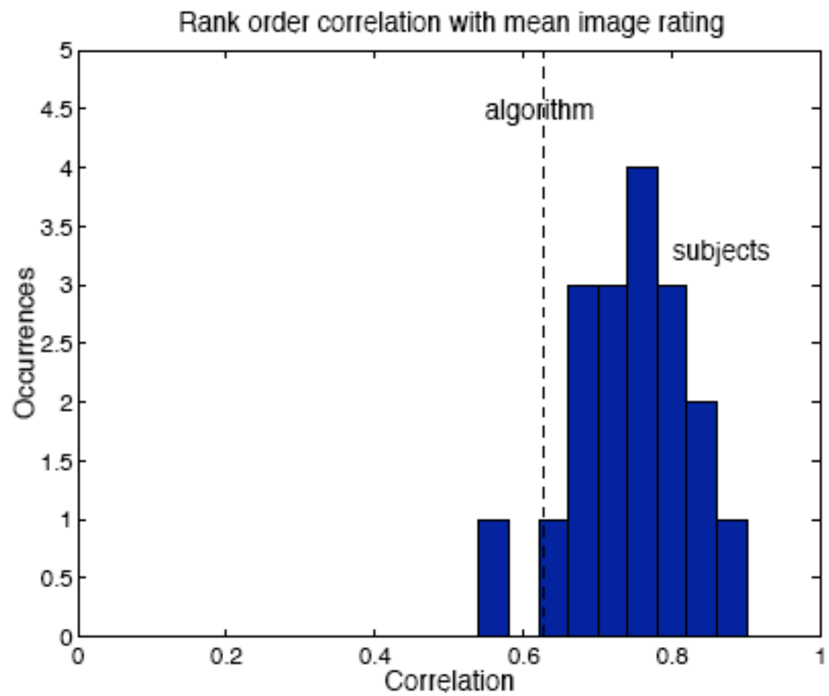


Figure 4: Correlation of individual subjects' image ratings compared with the mean rating (bars) compared with correlation of algorithm's rating with the mean rating (dashed line).

Separating shading from paint

- From a single image:
  - identify all-shading versus all-paint
  - locally separate shading from paint
- From a sequence of images:
  - separate stable from varying component
- From a stereo pair
  - separate shading, paint, occlusion.

Learning to separate shading from paint

Marshall F. Tappen[1]

William T. Freeman[1]

Edward H. Adelson[1,2]

[1]MIT Computer Science and Artificial Intelligence Laboratory (CSAIL)

[2]MIT Dept. Brain and Cognitive Sciences

Forming an Image

Surface

Forming an Image

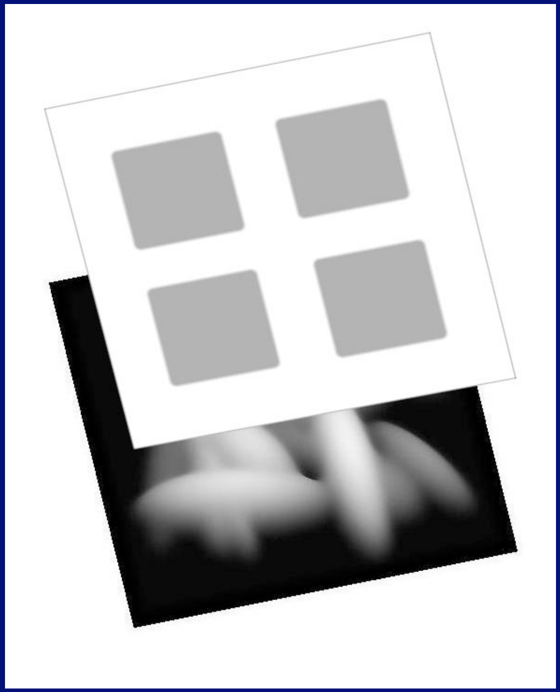
Illuminate the surface to get:

Surface

Shading Image

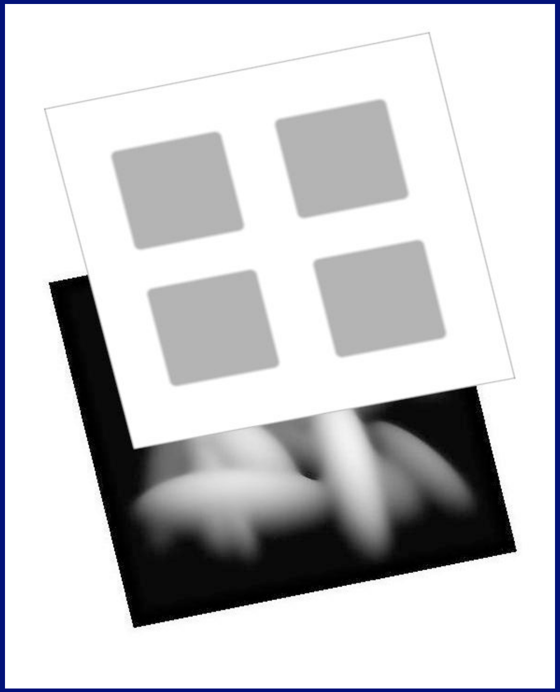
The “shading image” is the interaction of the shape of the surface and the illumination

Painting the Surface

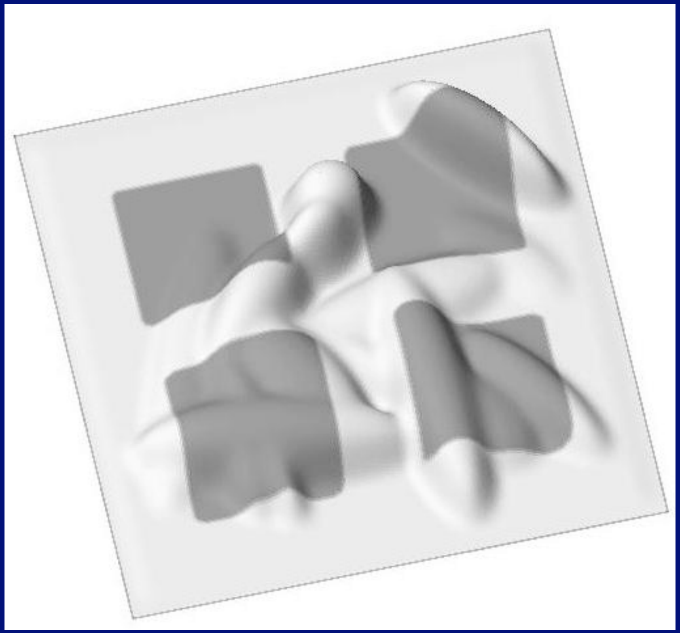


Scene

Painting the Surface

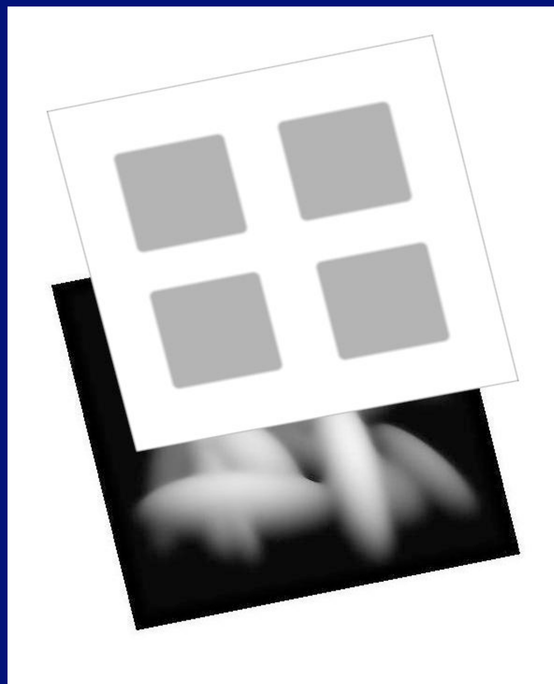


Scene

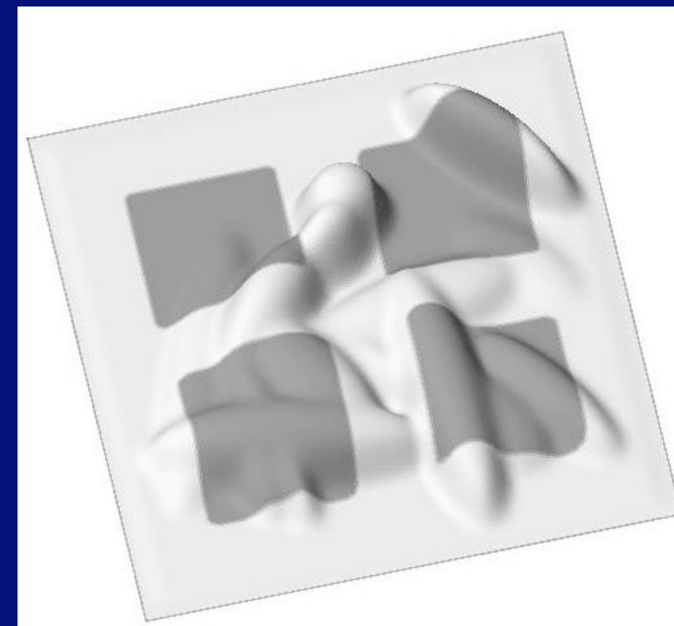


Image

Painting the Surface



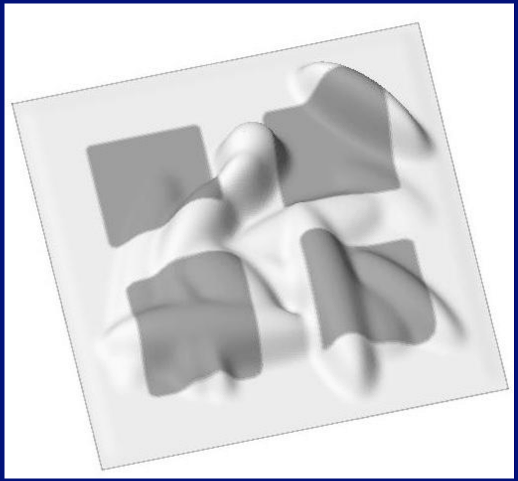
Scene



Image

We can also include a reflectance pattern or a “paint” image. Now shading and reflectance effects combine to create the observed image.

Goal: decompose the image into shading and reflectance components.

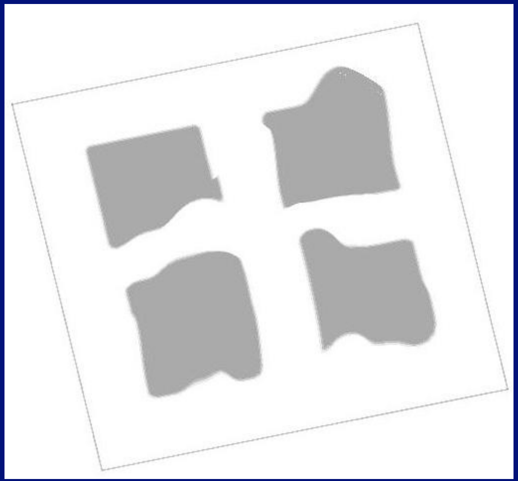


Image

=

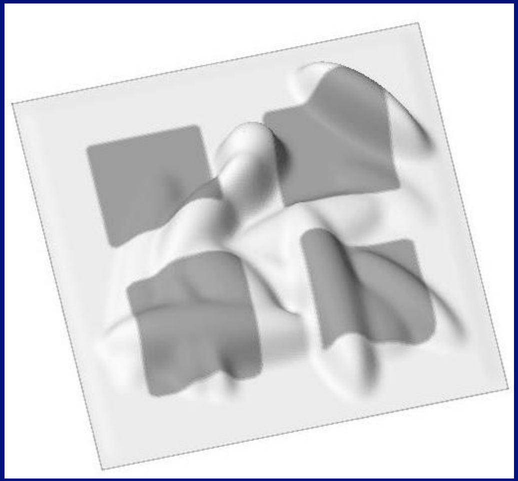
Shading Image

×



Reflectance Image

Goal: decompose the image into shading and reflectance components.

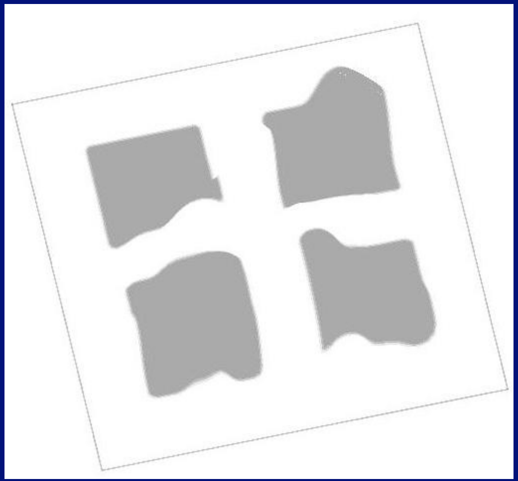


Image

=

Shading Image

×



Reflectance Image

- These types of images are known as intrinsic images (Barrow and Tenenbaum).
- Note: while the images multiply, we work in a gamma-corrected domain and assume the images add.

Why compute these intrinsic images

Why compute these intrinsic images

- Ability to reason about shading and reflectance independently is necessary for most image understanding tasks.
  - Material recognition
  - Image segmentation
- Want to understand how humans might do the task.
- <u>For image editing</u>, want access and modify the intrinsic images separately.

Treat the separation as a labeling problem

Treat the separation as a labeling problem

- We want to identify what parts of the image were caused by shape changes and what parts were caused by paint changes.

Treat the separation as a labeling problem

- We want to identify what parts of the image were caused by shape changes and what parts were caused by paint changes.
- But how represent that? Can’t label pixels of the image as “shading” or “paint”.

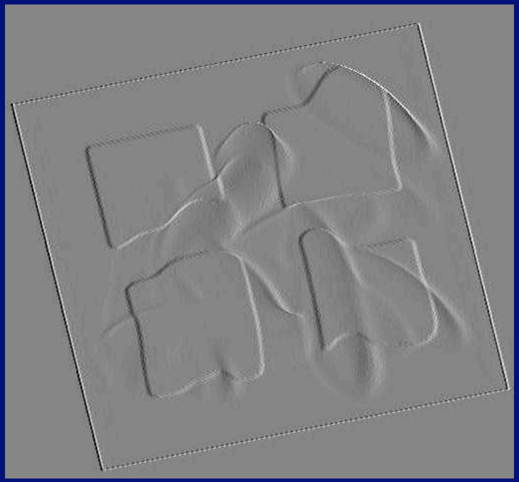
Treat the separation as a labeling problem

- We want to identify what parts of the image were caused by shape changes and what parts were caused by paint changes.
- But how represent that? Can't label pixels of the image as "shading" or "paint".
- Solution: we'll label *gradients* in the image as being caused by shading or paint.

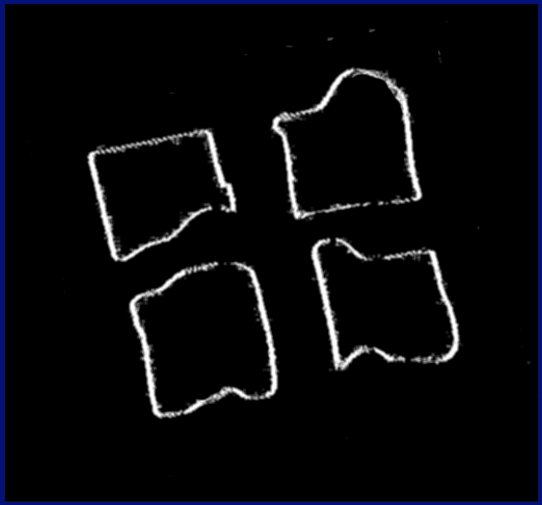
Treat the separation as a labeling problem

- We want to identify what parts of the image were caused by shape changes and what parts were caused by paint changes.
- But how represent that? Can't label pixels of the image as "shading" or "paint".
- Solution: we'll label *gradients* in the image as being caused by shading or paint.
- Assume that image gradients have only one cause.

Recovering Intrinsic Images



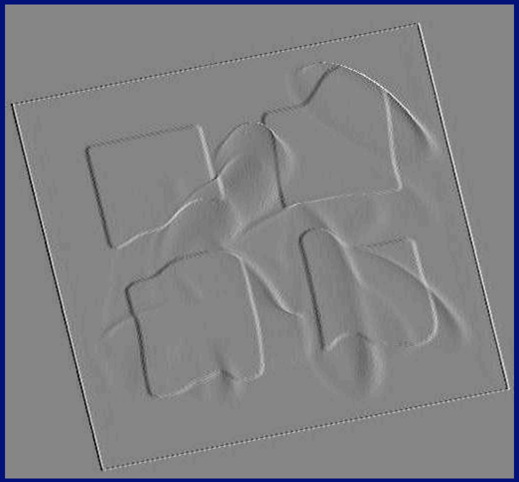
Original x derivative image



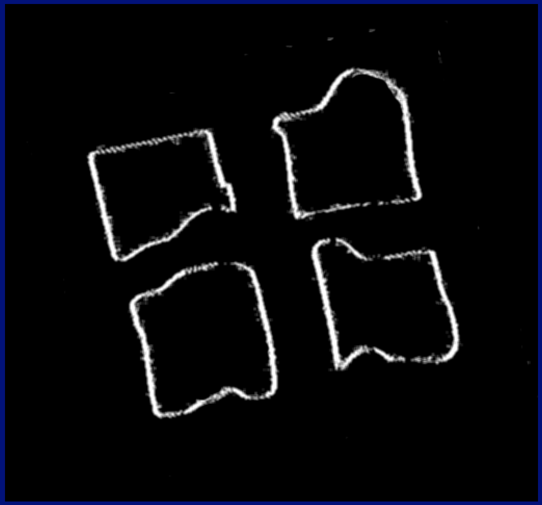
Classify each derivative
(White is reflectance)

Recovering Intrinsic Images

- Classify each *x* and *y* image derivative as being caused by *either* shading or a reflectance change



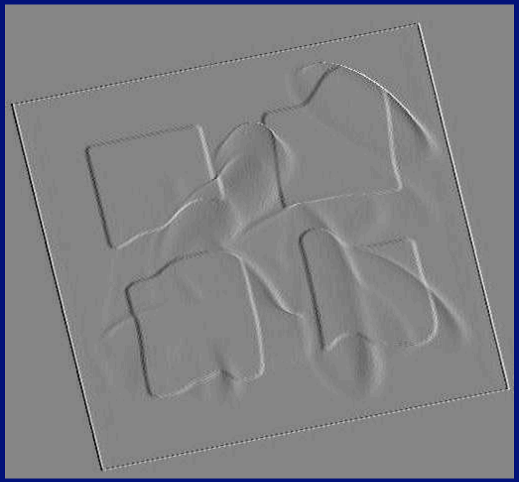
Original *x* derivative image



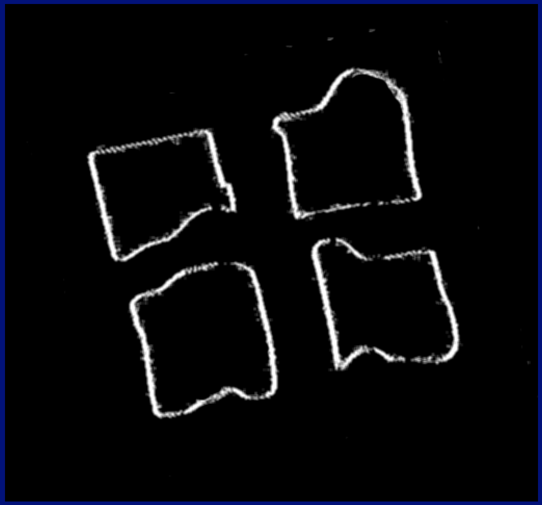
Classify each derivative
(White is reflectance)

Recovering Intrinsic Images

- Classify each *x* and *y* image derivative as being caused by *either* shading or a reflectance change
- Recover the intrinsic images by finding the least-squares reconstruction from each set of labeled derivatives. (Fast Matlab code for that available from Yair Weiss's web page.)



Original *x* derivative image



Classify each derivative (White is reflectance)

Classic algorithm: Retinex

(a) An example of a Mondrian image.

(b) The reflectance pattern of the image.

(c) The illumination pattern of the image.

- Assume world is made up of Mondrian reflectance patterns and smooth illumination
- Can classify derivatives by the magnitude of the derivative

Outline of our algorithm

Outline of our algorithm

- Gather local evidence for shading or reflectance

Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)

Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)
  - Form (local image patterns)

Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)
  - Form (local image patterns)
- Integrate the local evidence across space.

Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)
  - Form (local image patterns)
- Integrate the local evidence across space.
  - Assume a probabilistic model and use “belief propagation”.

Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)
  - Form (local image patterns)
- Integrate the local evidence across space.
  - Assume a probabilistic model and use “belief propagation”.

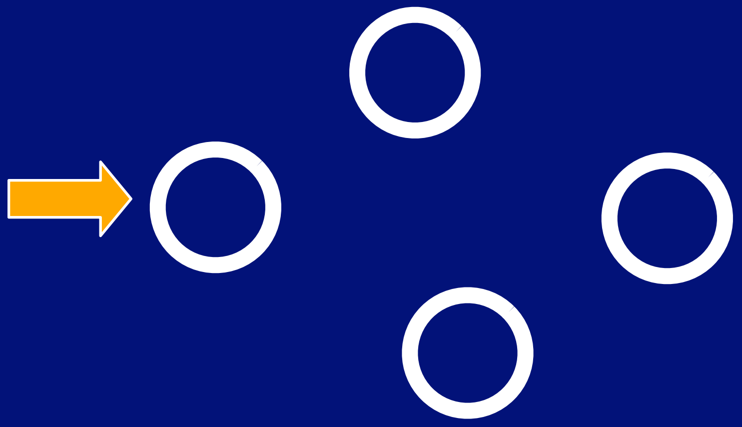
Outline of our algorithm

- Gather local evidence for shading or reflectance
  - Color (chromaticity changes)
  - Form (local image patterns)
- Integrate the local evidence across space.
  - Assume a probabilistic model and use “belief propagation”.

- Results shown on example images

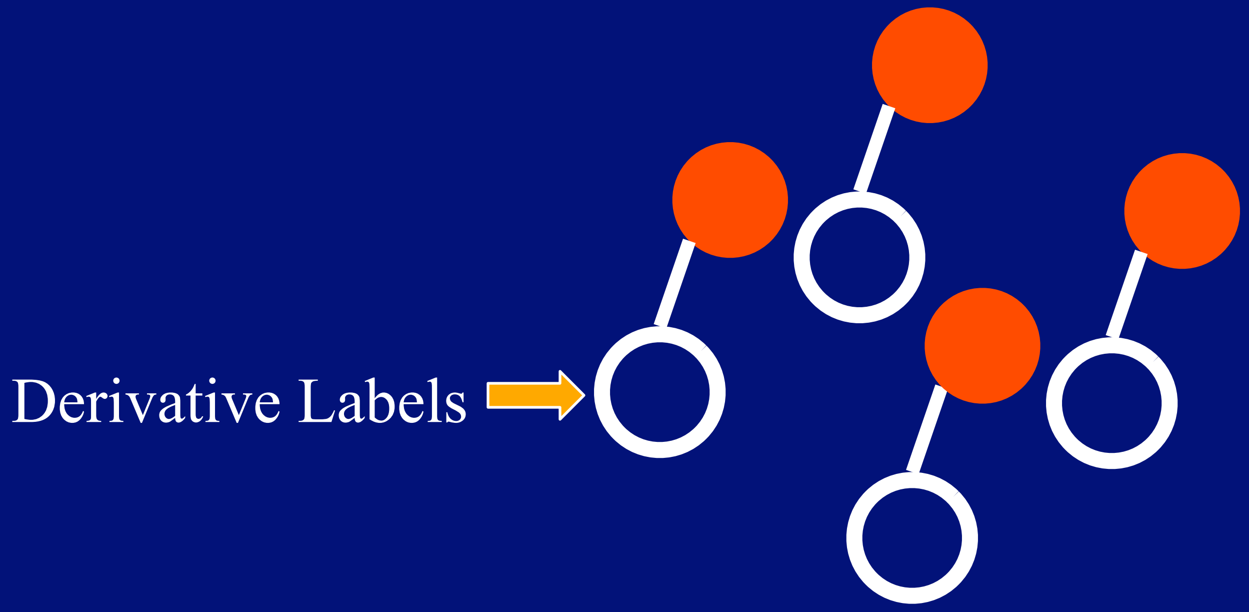
Probabilistic graphical model

Unknown Derivative Labels (hidden random variables that we want to estimate)



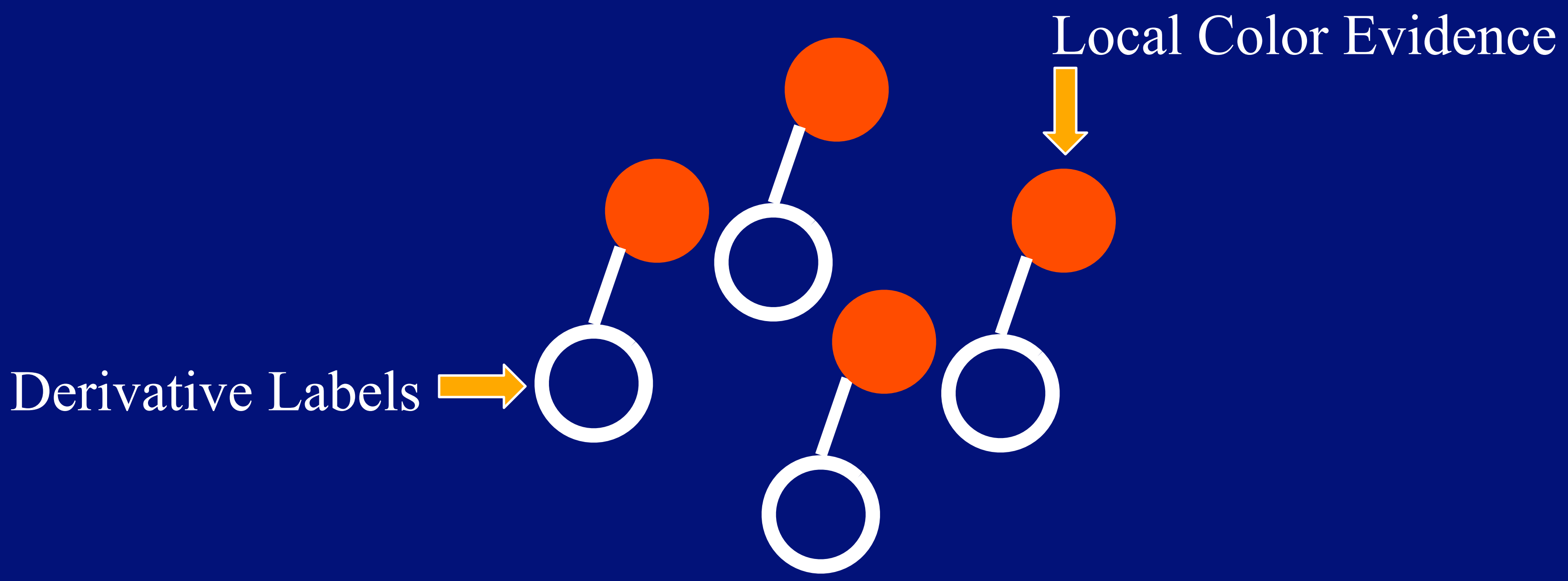
Probabilistic graphical model

- Local evidence



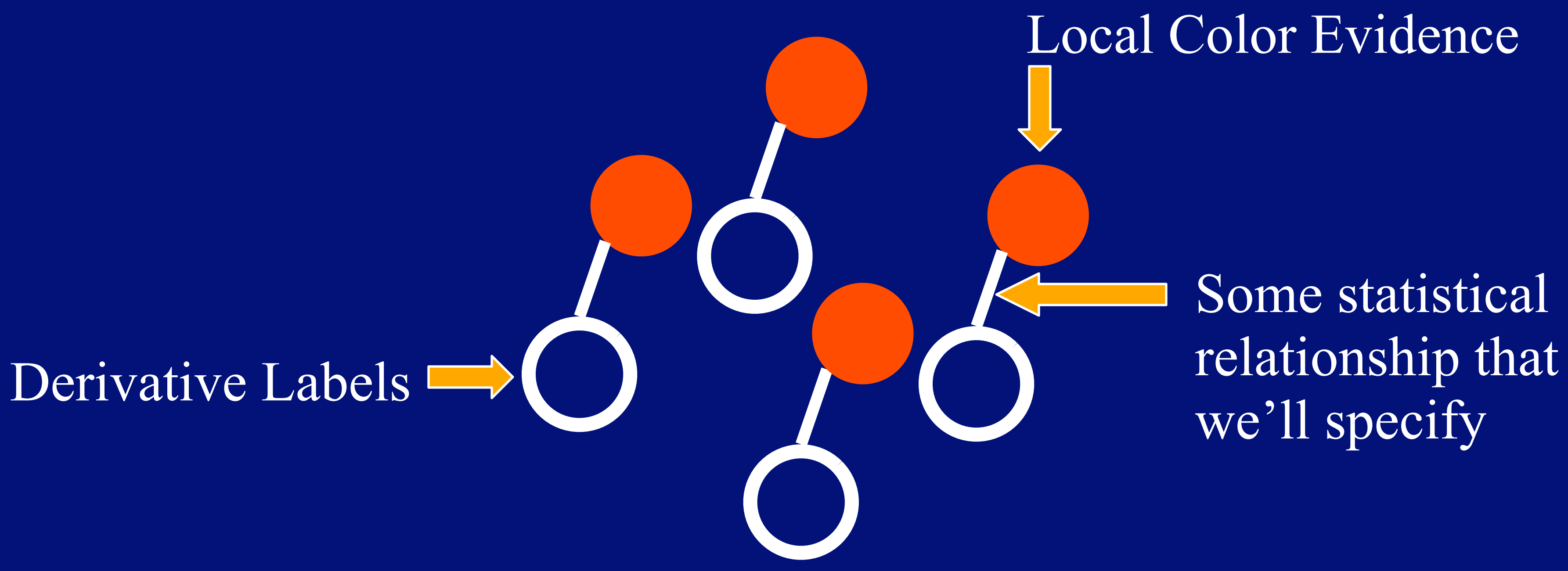
Probabilistic graphical model

- Local evidence



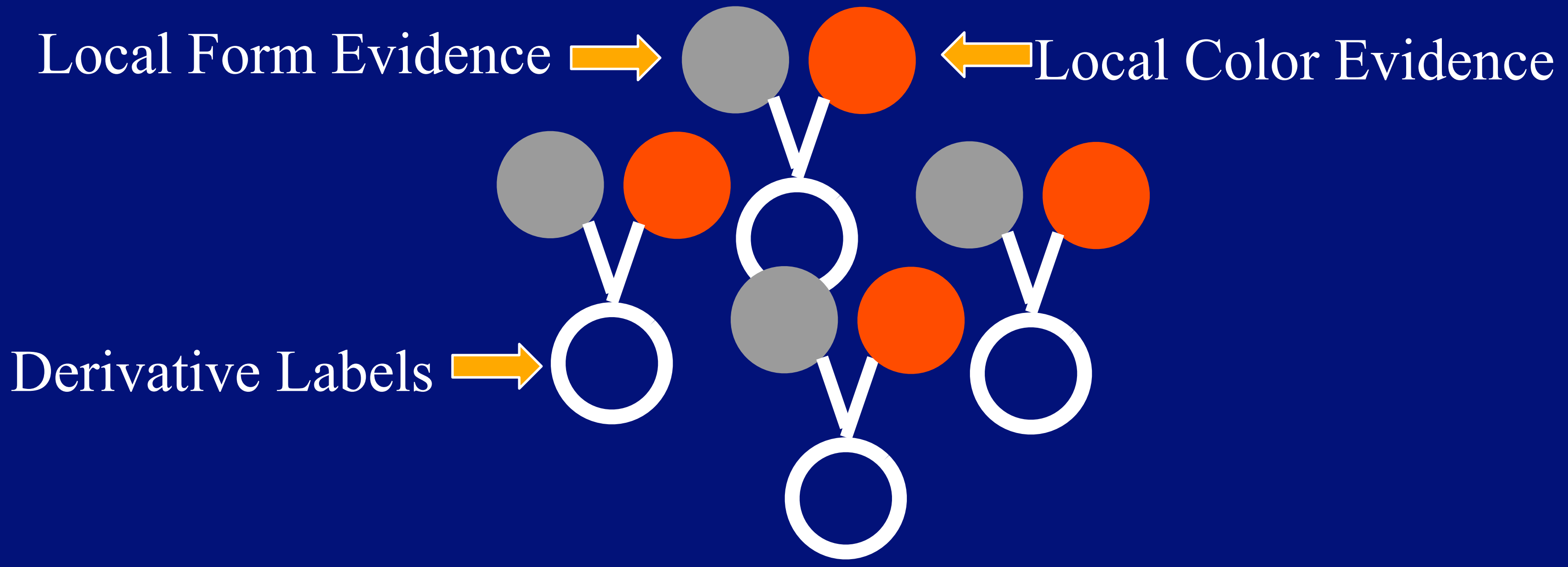
Probabilistic graphical model

- Local evidence



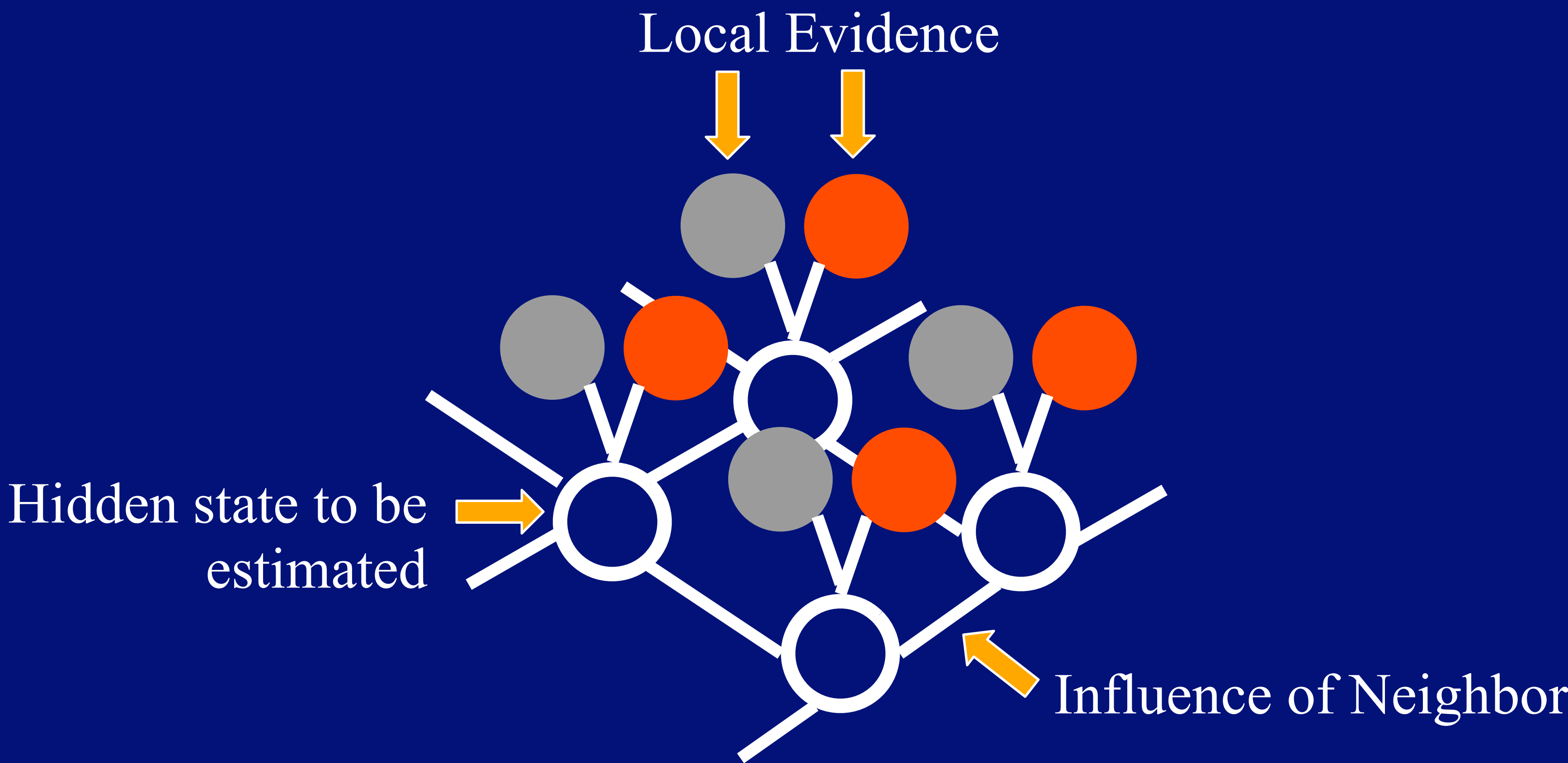
Probabilistic graphical model

- Local evidence



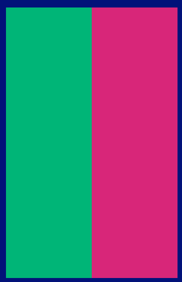
Probabilistic graphical model

Propagate the local evidence in Markov Random Field.
This strategy can be used to solve other low-level vision problems.

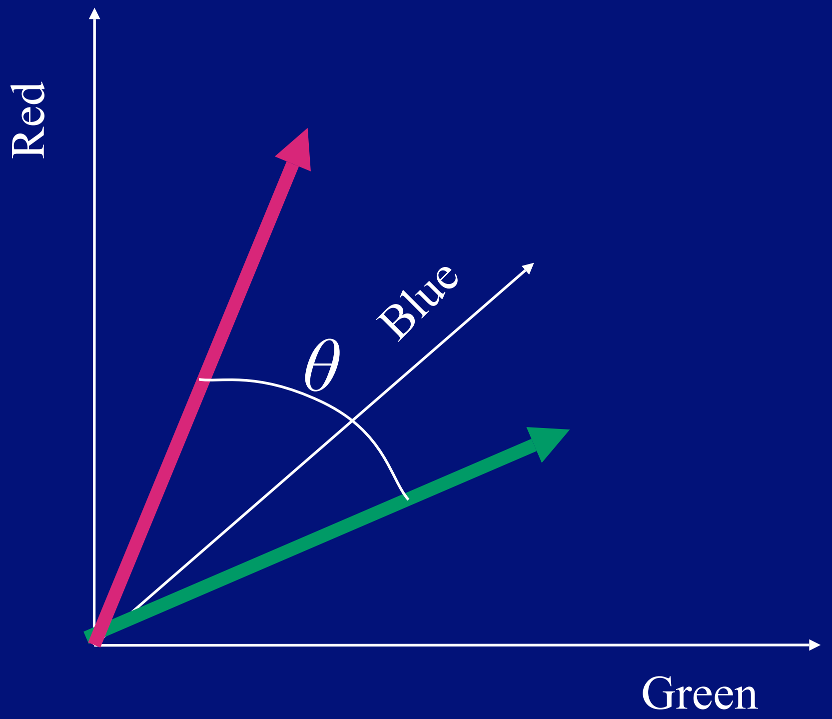


Classifying Color Changes

Chromaticity Changes

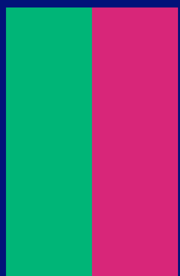


Angle between the two vectors, θ, is greater than 0

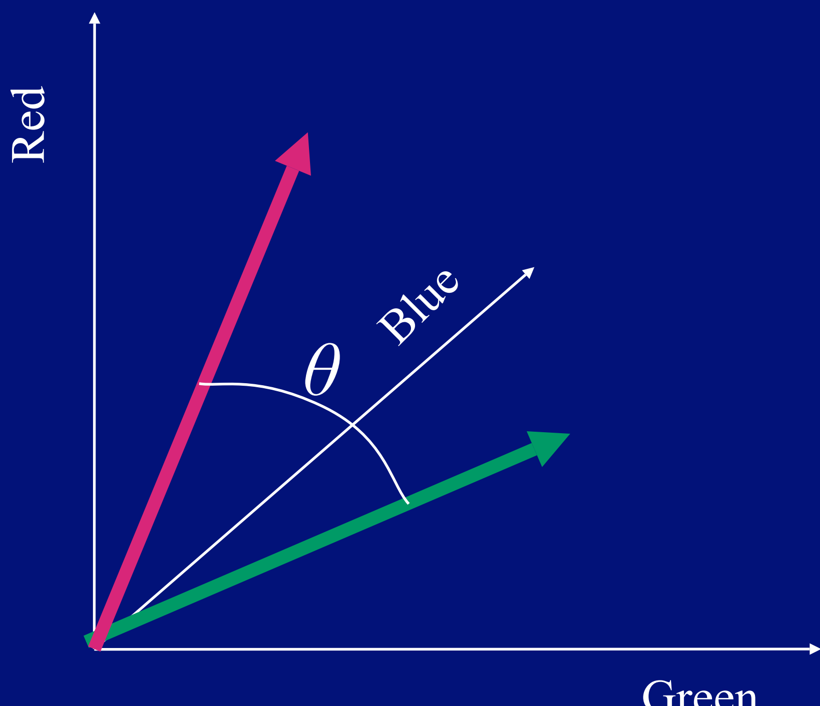


Classifying Color Changes

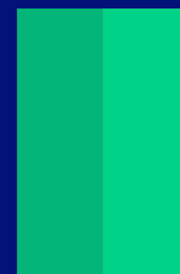
Chromaticity Changes



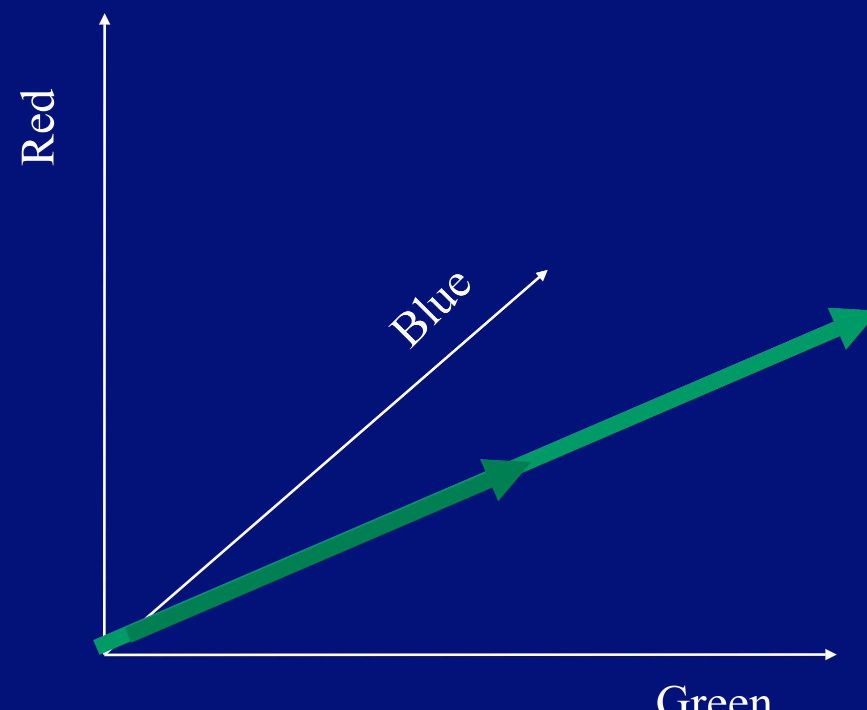
Angle between the two vectors, θ, is greater than 0



Intensity Changes

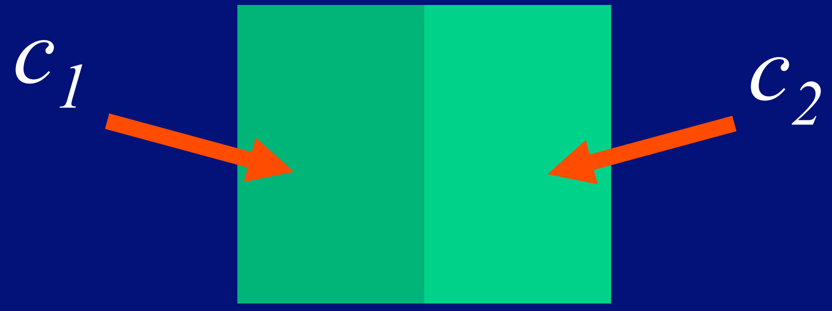


Angle between two vectors, θ, equals 0



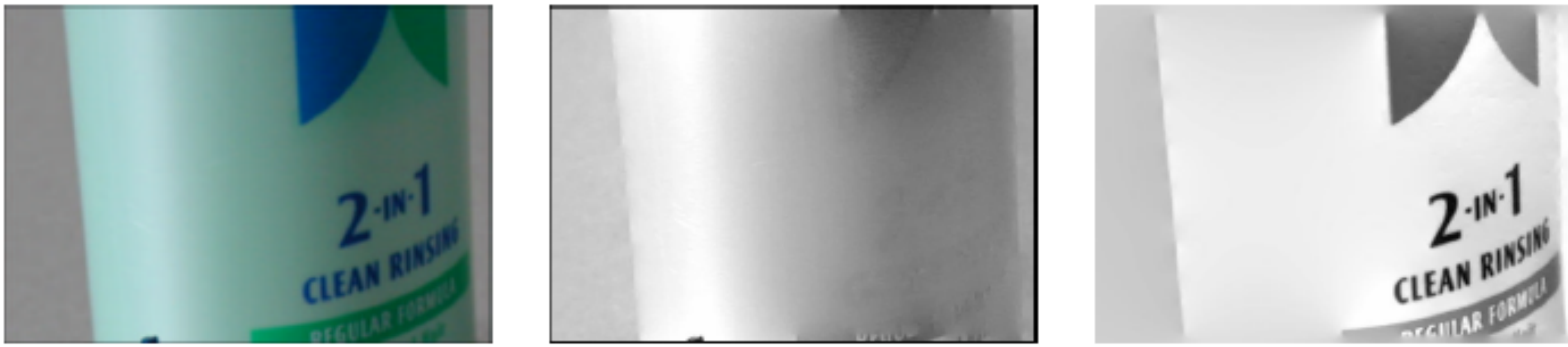
Color Classification Algorithm

1. Normalize the two color vectors c_1 and c_2



2. If $(c_1 \cdot c_2) > T$
   - Derivative is a reflectance change
   - Otherwise, label derivative as shading

Result using only color information



(a) Original Image (b) Shading Image (c) Reflectance Image

Figure 1: Example. Computed using Color Detector. To facilitate printing, the intrinsic images have been computed from a gray-scale version of the image. The color information is used solely for classifying derivatives in the gray-scale copy of the image.

Results Using Only Color

Input

Results Using Only Color

Input

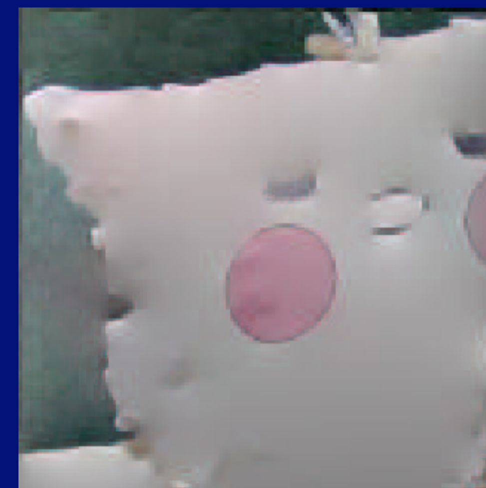
Shading

Reflectance

Results Using Only Color

Input

Shading



Reflectance

- Some changes are ambiguous

Results Using Only Color

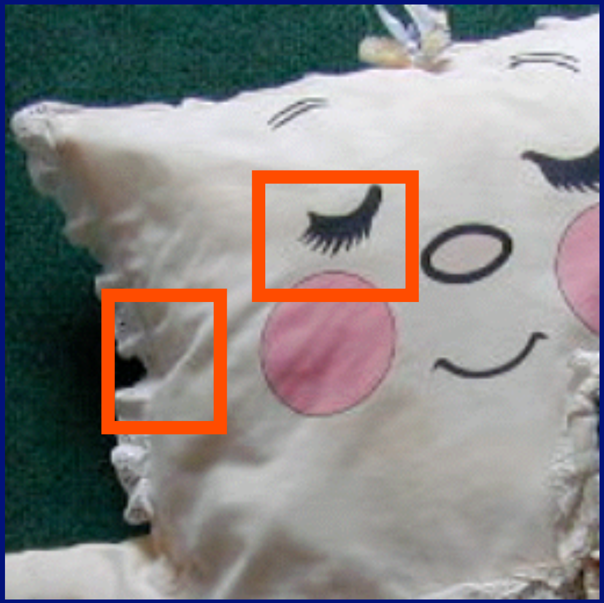
Input

Shading

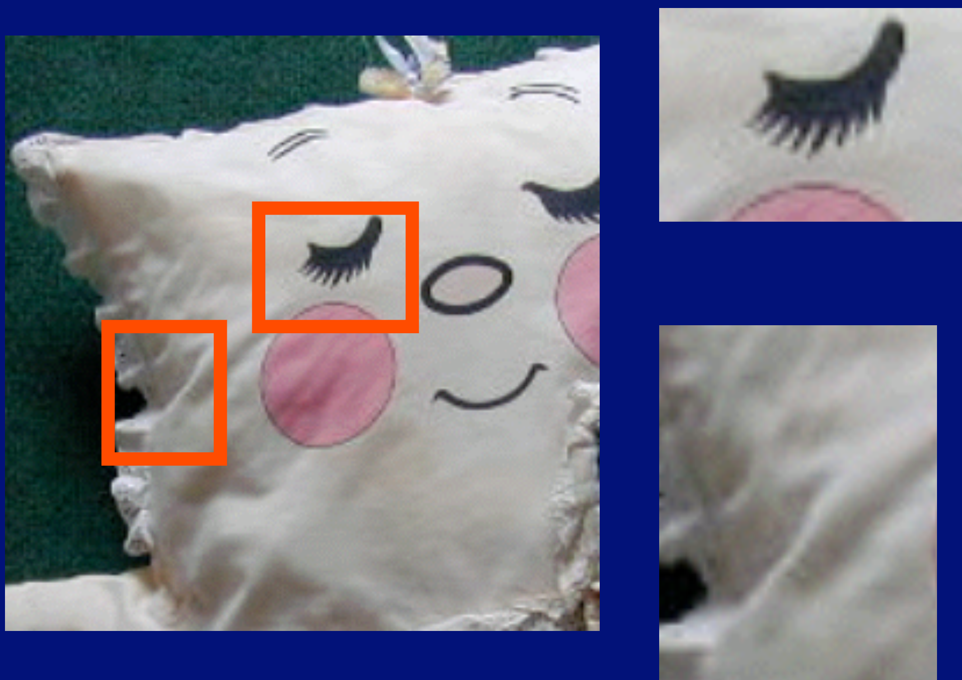
Reflectance

- Some changes are ambiguous
- Intensity changes could be caused by shading or reflectance
  - So we label it as "ambiguous"
  - Need more information

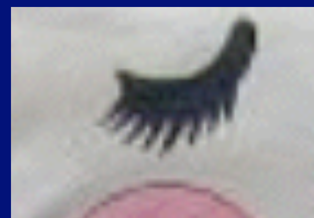
Utilizing local intensity patterns



Utilizing local intensity patterns



Utilizing local intensity patterns

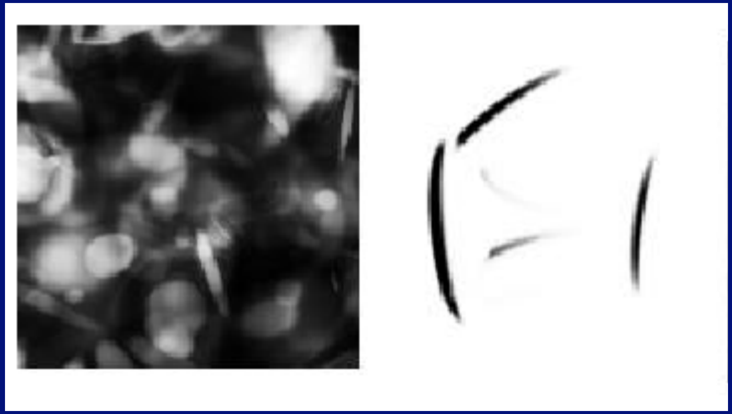


• The painted eye and the ripples of the fabric have very different appearances

• Can learn classifiers which take advantage of these differences

Shading/paint training set

Examples from Reflectance Change Training Set



Examples from Shading Training Set

From Weak to Strong Classifiers: Boosting

- Individually these weak classifiers aren't very good.
- Can be combined into a single strong classifier.
- Call the classification from a weak classifier $h_i(x)$.
- Each $h_i(x)$ votes for the classification of x (-1 or 1).
- Those votes are weighted and combined to produce a final classification.

$$H(x) = \text{sign}\left(\sum_i \alpha_i h_i(x) \right)$$

Using Local Intensity Patterns

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative
- The classification of a derivative:

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative
- The classification of a derivative:

I

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative
- The classification of a derivative:

$$I * F$$

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative
- The classification of a derivative:

abs (

I

*

F

)

Using Local Intensity Patterns

- Create a set of weak classifiers that use a small image patch to classify each derivative
- The classification of a derivative:

I

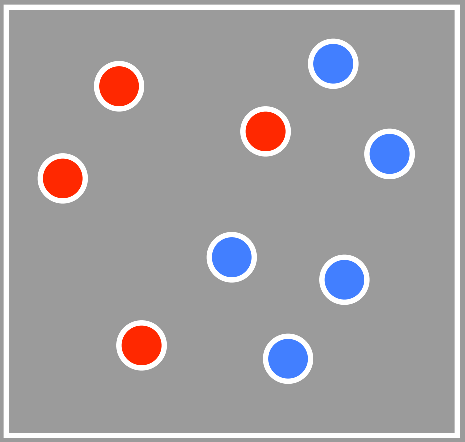
F

$$\text{abs}\left(I * F \right) > T$$

AdaBoost

(Freund & Shapire '95)

Initial uniform weight on training examples



$$f(x) = \theta\left(\sum_t \alpha_t h_t(x)\right)$$

$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t}\right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

AdaBoost

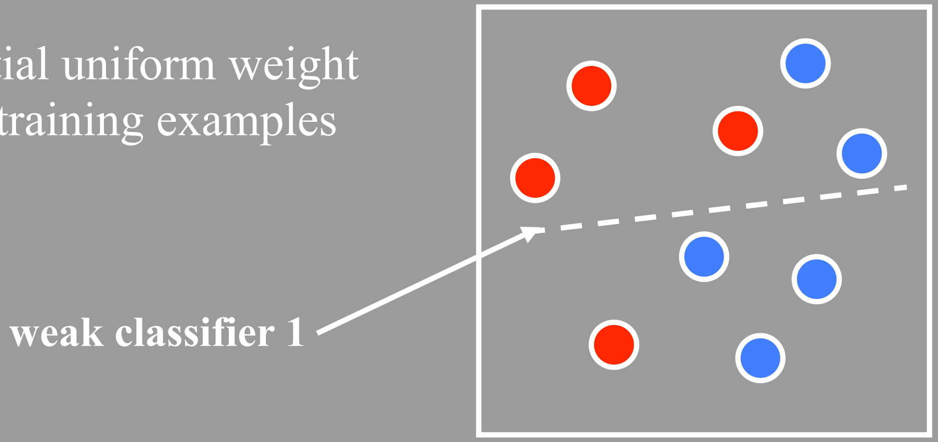
(Freund & Shapire '95)

$$f(x) = \theta\left(\sum_t \alpha_t h_t(x)\right)$$

$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t}\right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$

Initial uniform weight on training examples

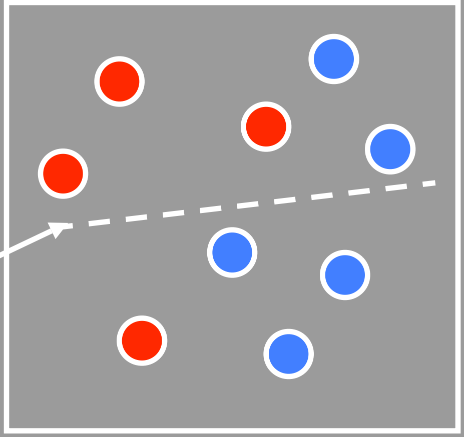


Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

AdaBoost

(Freund & Shapire '95)

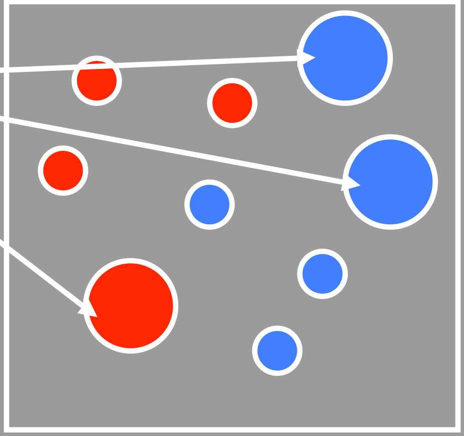
Initial uniform weight
on training examples



weak classifier 1

$$f(x) = \theta\left(\sum_t \alpha_t h_t(x)\right)$$

Incorrect classifications
re-weighted more heavily



$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t}\right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

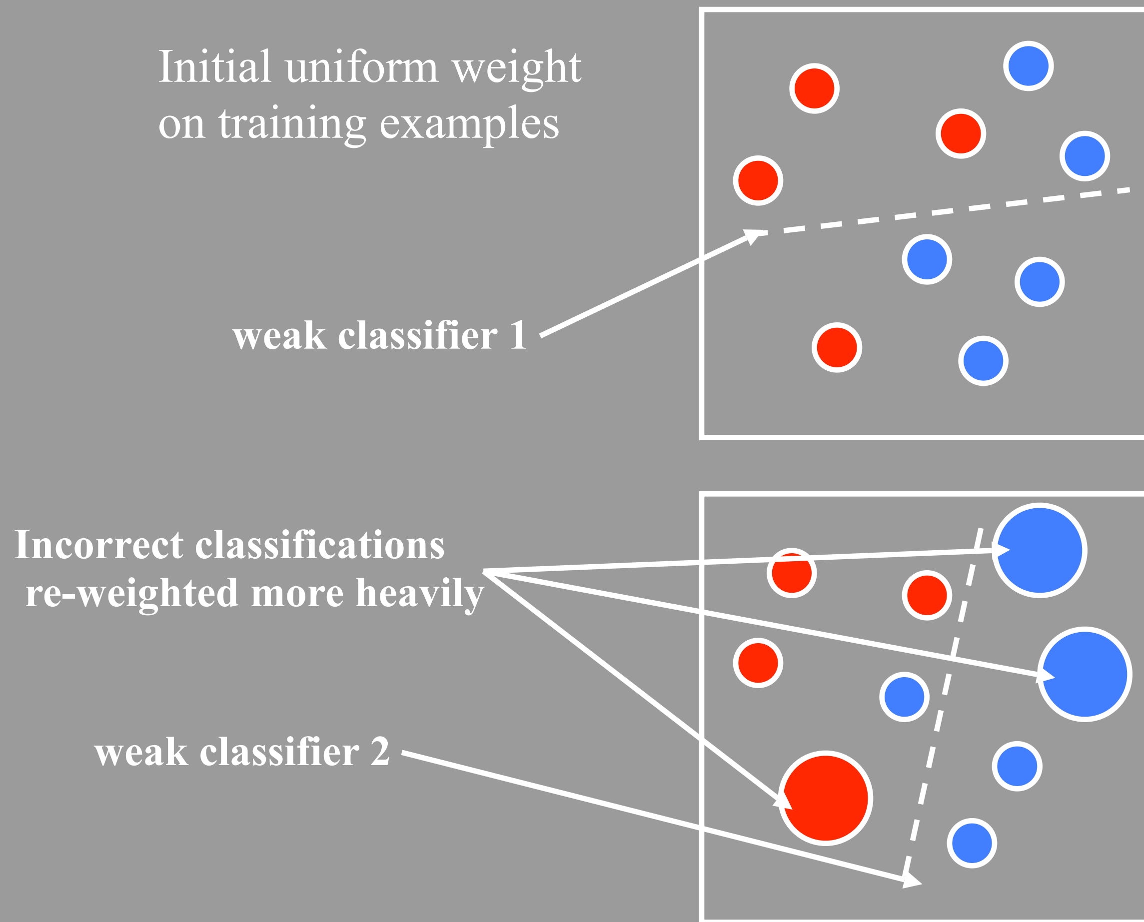
AdaBoost

(Freund & Shapire ’95)

$$f(x) = \theta\left(\sum_t \alpha_t h_t(x) \right)$$

$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t} \right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$



Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

AdaBoost

(Freund & Shapire '95)

$$f(x) = \theta\left(\sum_t \alpha_t h_t(x)\right)$$

$$\alpha_t = 0.5 \log\left(\frac{error_t}{1 - error_t}\right)$$

$$w_t^i = \frac{w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}{\sum_i w_{t-1}^i e^{-y_i \alpha_t h_t(x_i)}}$$

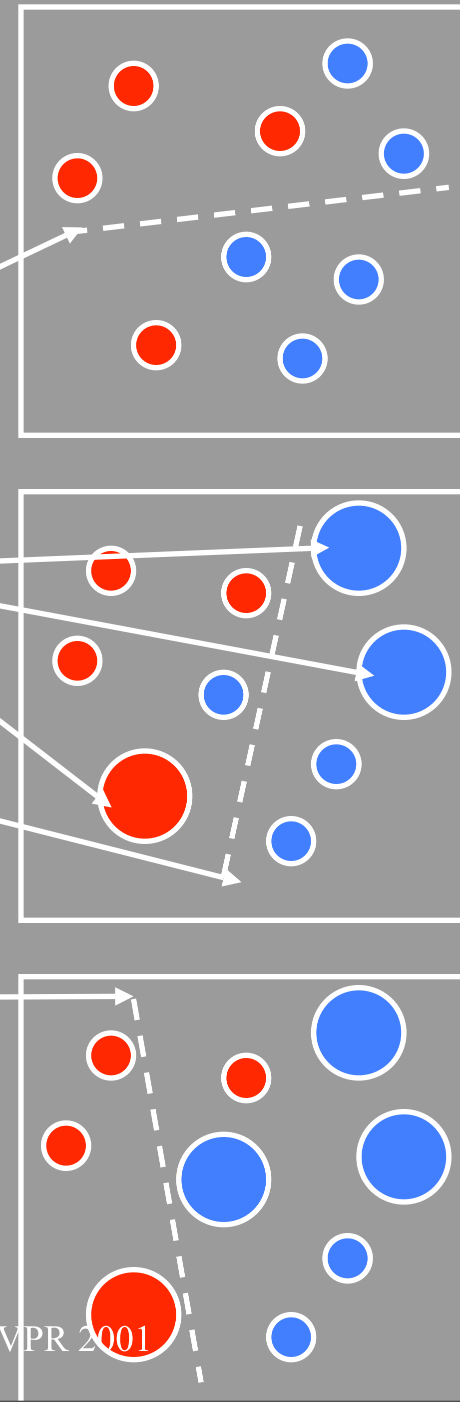
Initial uniform weight on training examples

weak classifier 1

Incorrect classifications re-weighted more heavily

weak classifier 2

weak classifier 3



Final classifier is weighted combination of weak classifiers

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Beautiful AdaBoost Properties

- Training Error approaches 0 exponentially
- Bounds on Testing Error Exist
  - Analysis is based on the Margin of the Training Set
- Weights are related the margin of the example
  - Examples with negative margin have large weight
  - Examples with positive margin have small weights

$$f(x) = \sum_i \alpha_i h_i(x)$$

$$C(x) = \theta\left(f(x)\right)$$

$$\min \sum_i e^{-y_i f(x_i)} \geq \sum_i \left(1 - y_i C(x_i)\right)$$

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

Ada-Boost Tutorial

- Given a Weak learning algorithm
  - Learner takes a training set and returns the best classifier from a weak concept space
    - required to have error < 50%
- Starting with a Training Set (initial weights 1/n)
  - Weak learning algorithm returns a classifier
  - Reweight the examples
    - Weight on correct examples is decreased
    - Weight on errors is decreased

$$\boxed{\sum_{i \in Errors} w_i = \sum_{j \in Correct} w_j}$$

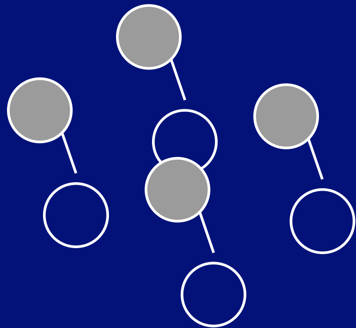
- Final classifier is a weighted majority of Weak Classifiers
  - Weak classifiers with low error get larger weight

Viola and Jones, Robust object detection using a boosted cascade of simple features, CVPR 2001

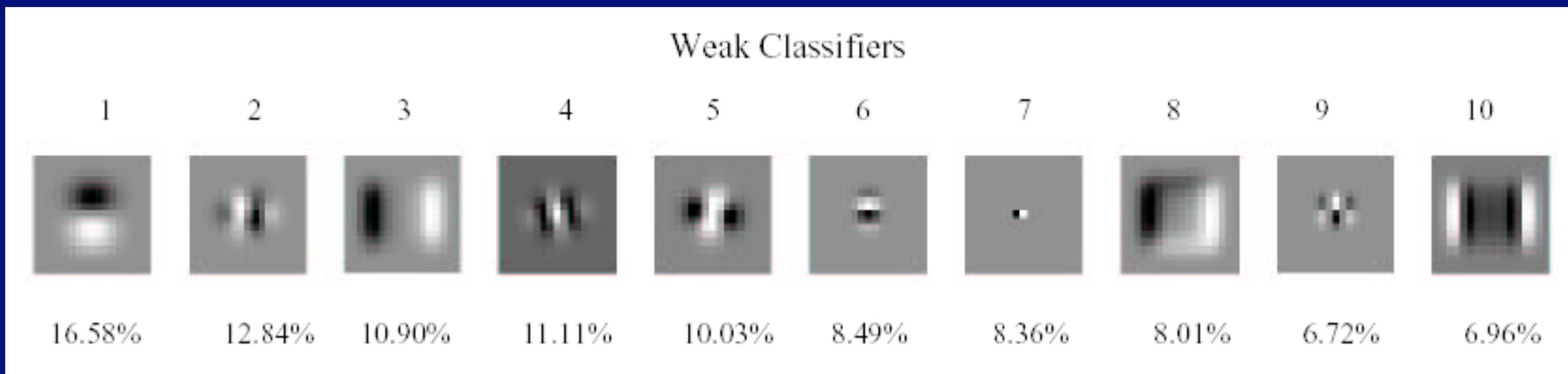
Learning the Classifiers

- The weak classifiers, $h_i(x)$, and the weights α are chosen using the *AdaBoost* algorithm (see www.boosting.org for introduction).
- Train on synthetic images.
- Assume the light direction is from the right.

- Filters for the candidate weak classifiers—cascade two out of these 4 categories:
  - Multiple orientations of 1^{st} derivative of Gaussian filters
  - Multiple orientations of 2^{nd} derivative of Gaussian filters
  - Several widths of Gaussian filters
  - impulse

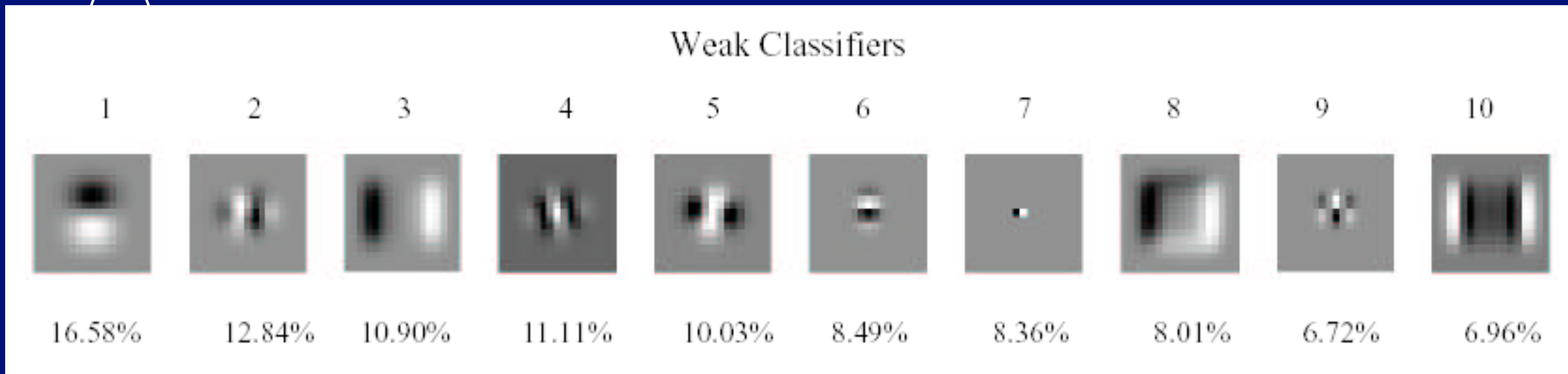


Classifiers Chosen
(assuming illumination from above)

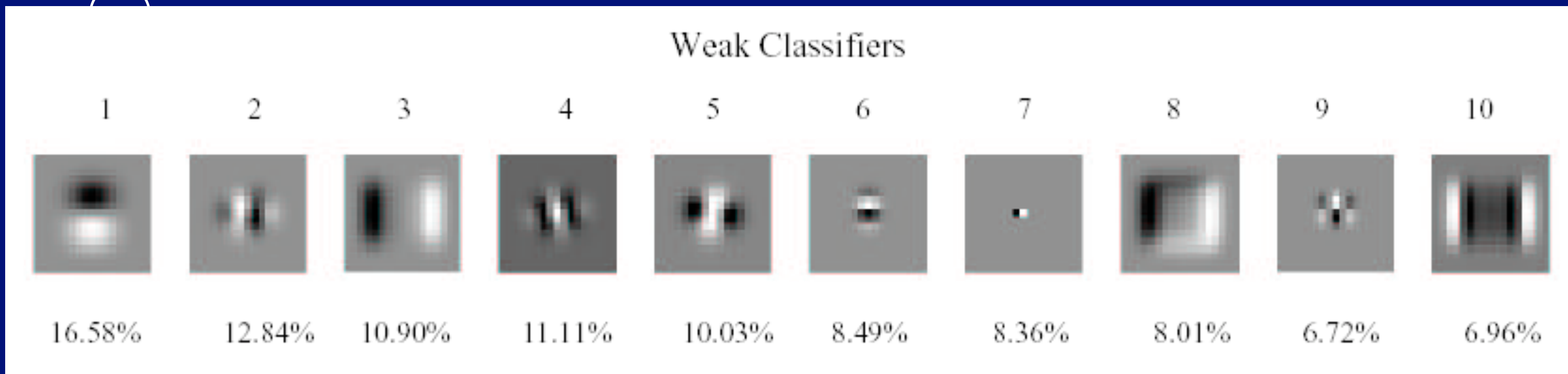


- These are the filters chosen for classifying vertical derivatives when the illumination comes from the top of the image.
- Each filter corresponds to one $h_i(x)$

Characterizing the learned classifiers

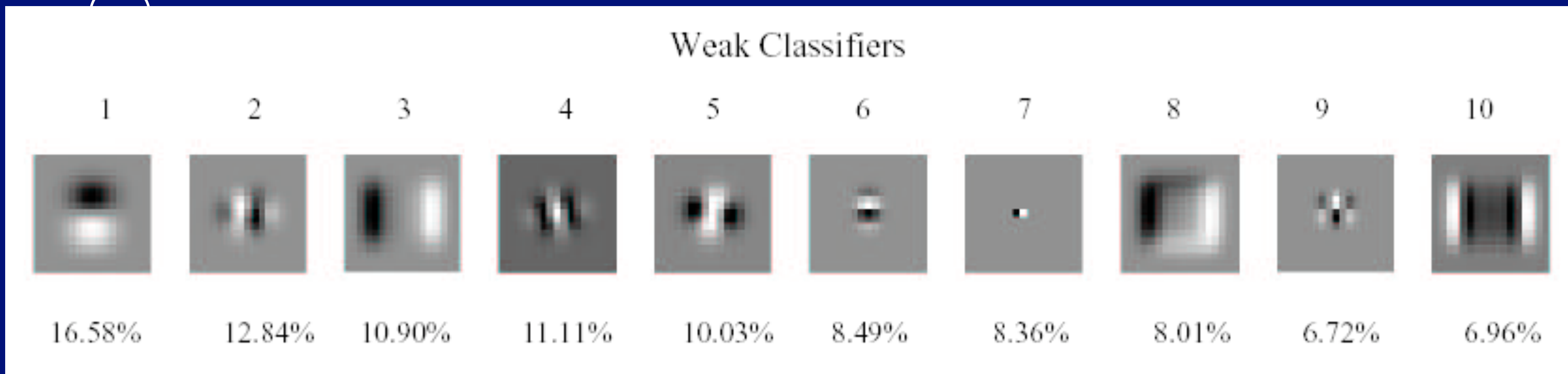


Characterizing the learned classifiers



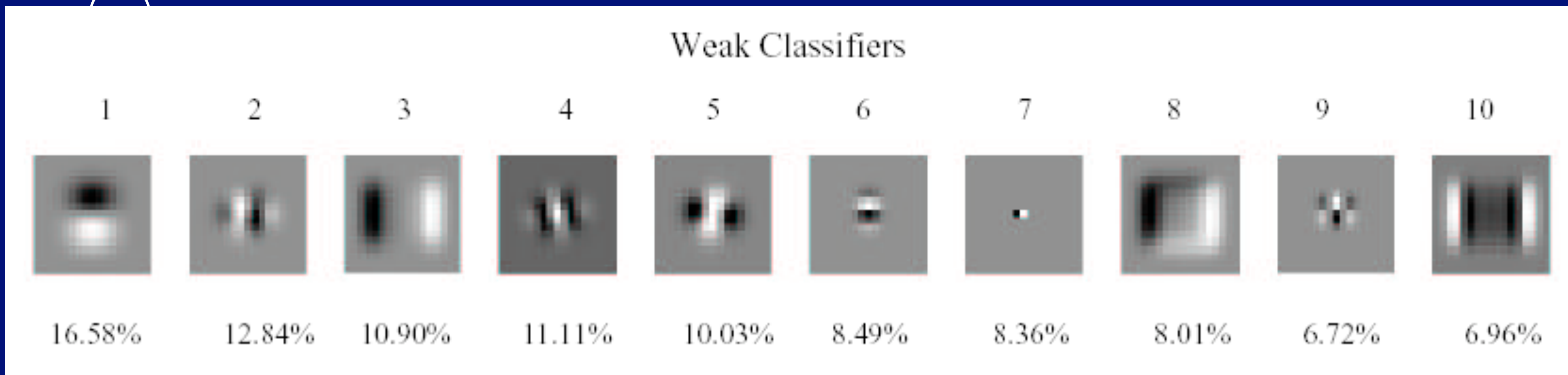
- Learned rules for all (but classifier 9) are: if rectified filter response is above a threshold, vote for reflectance.

Characterizing the learned classifiers



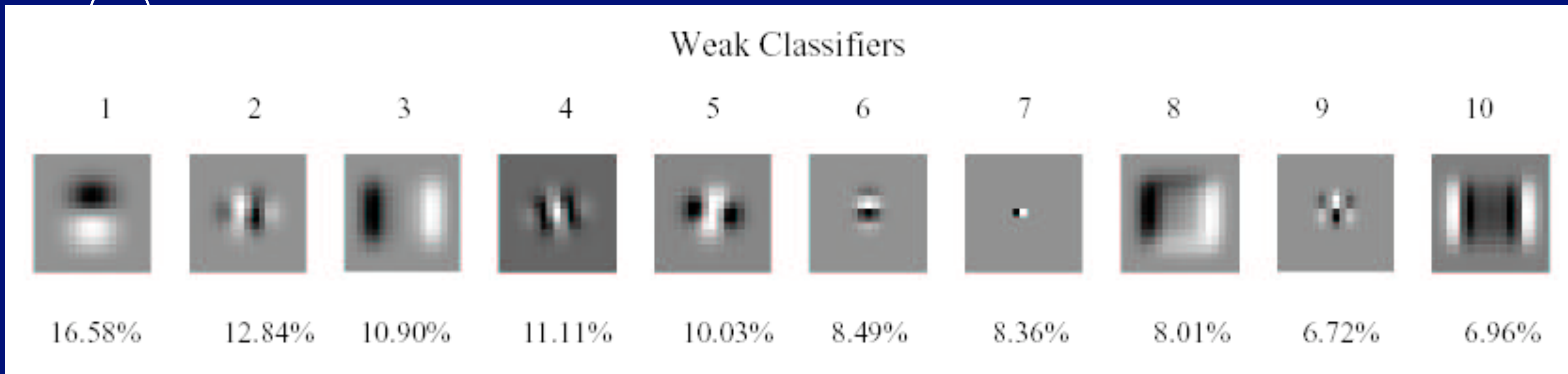
- Learned rules for all (but classifier 9) are: if rectified filter response is above a threshold, vote for reflectance.
- Yes, contrast and scale are all folded into that. We perform an overall contrast normalization on all images.

Characterizing the learned classifiers



- Learned rules for all (but classifier 9) are: if rectified filter response is above a threshold, vote for reflectance.
- Yes, contrast and scale are all folded into that. We perform an overall contrast normalization on all images.
- Classifier 1 (the best performing single filter to apply) is an empirical justification for Retinex algorithm: treat small derivative values as shading.

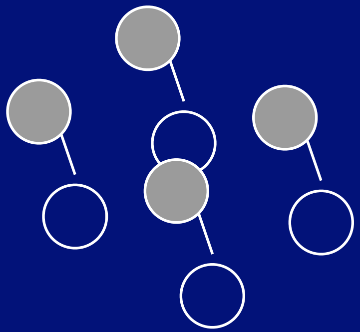
Characterizing the learned classifiers



- Learned rules for all (but classifier 9) are: if rectified filter response is above a threshold, vote for reflectance.
- Yes, contrast and scale are all folded into that. We perform an overall contrast normalization on all images.
- Classifier 1 (the best performing single filter to apply) is an empirical justification for Retinex algorithm: treat small derivative values as shading.
- The other classifiers look for image structure oriented perpendicular to lighting direction as evidence for reflectance change.

Results Using Only Form Information

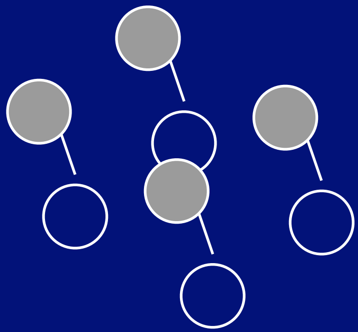
Input Image



Results Using Only Form Information

Input Image

Shading Image



Results Using Only Form Information

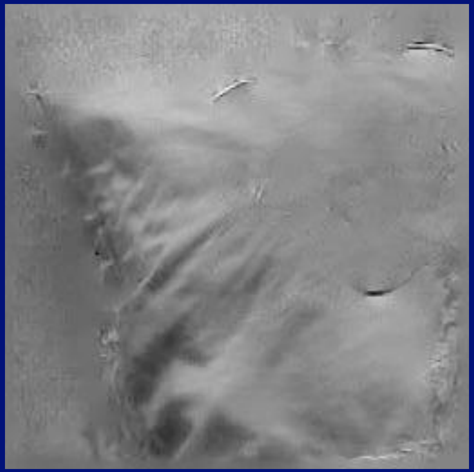
Input Image

Shading Image

Reflectance Image

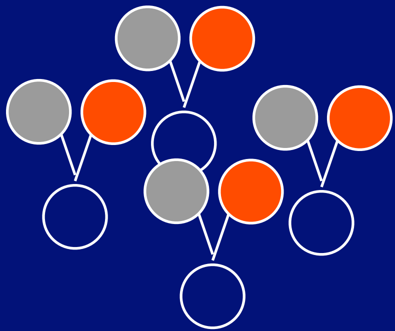
Using Both Color and Form Information

Input image



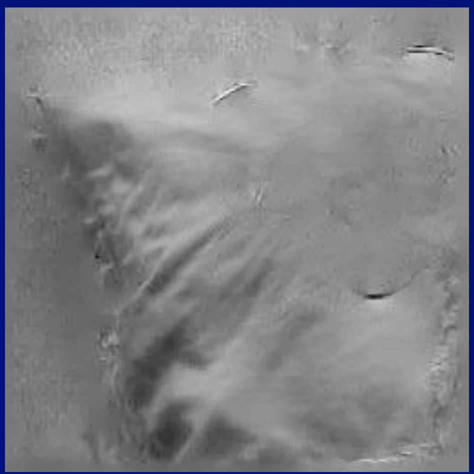
Shading

Reflectance



Using Both Color and Form Information

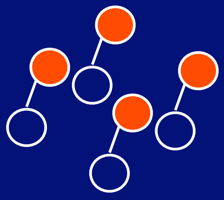
Input image



Shading

Reflectance

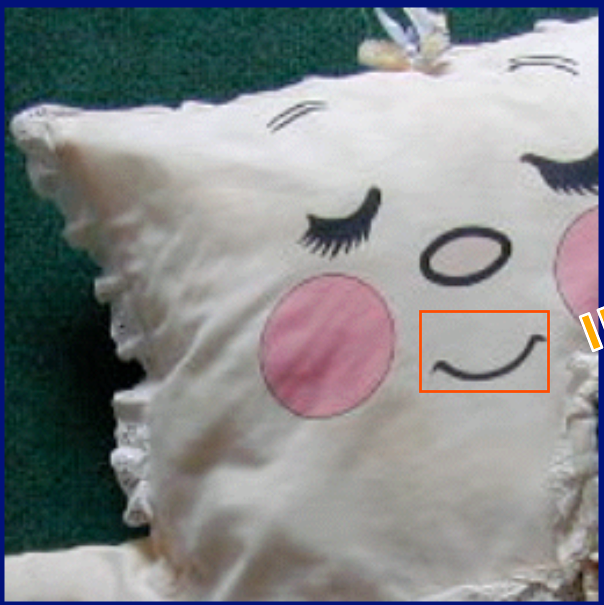
Results only using chromaticity.



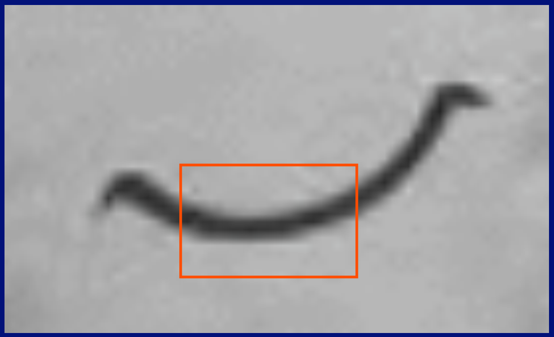
Some Areas of the Image Are Ambiguous

Input

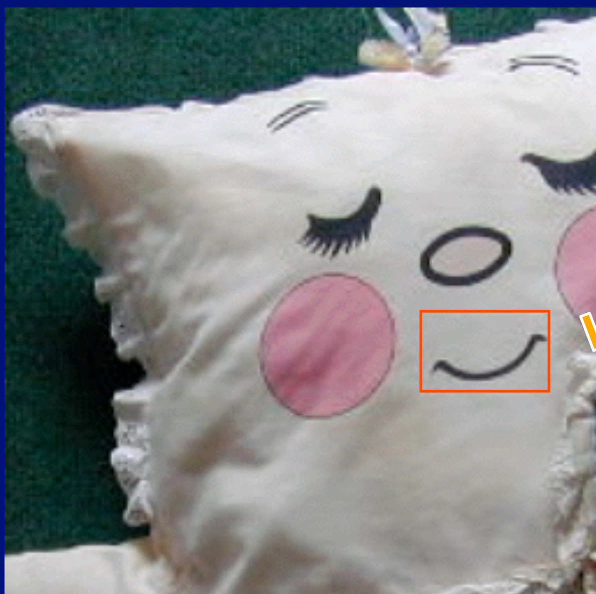
Some Areas of the Image Are Ambiguous



Input



Some Areas of the Image Are Ambiguous

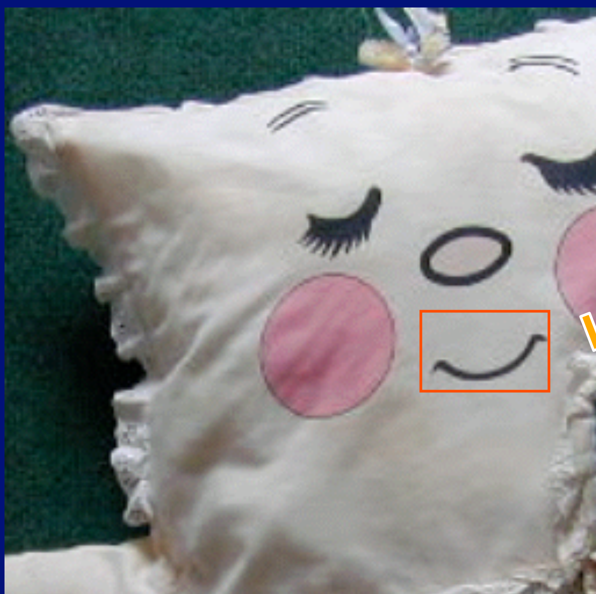


Input

Is the change here better explained as

Shading

Some Areas of the Image Are Ambiguous



Input

Is the change here better explained as

Shading

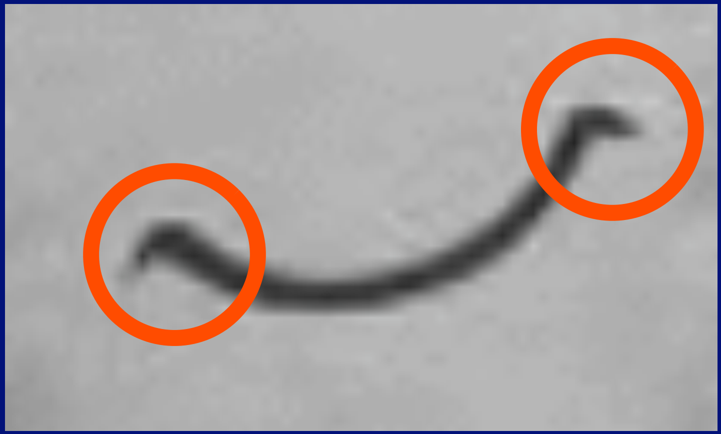
or

?

Reflectance

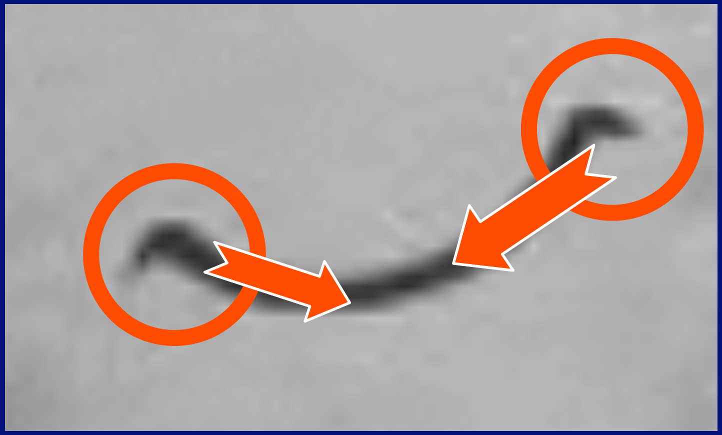
Propagating Information

- Can disambiguate areas by propagating information from reliable areas of the image into ambiguous areas of the image



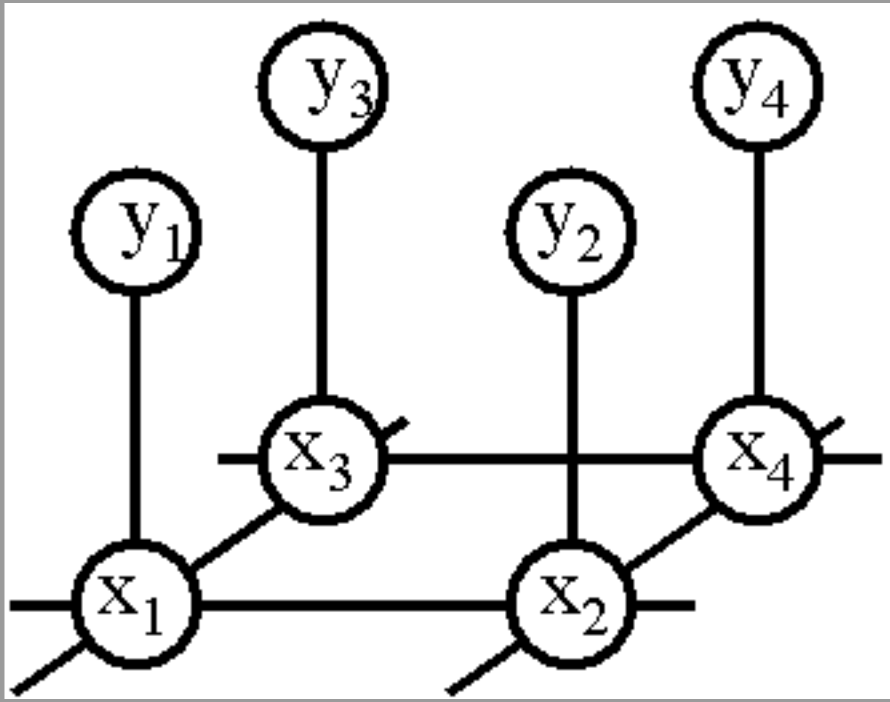
Propagating Information

- Can disambiguate areas by propagating information from reliable areas of the image into ambiguous areas of the image



Markov Random Fields

- Allows rich probabilistic models for images.
- But built in a local, modular way. Learn local relationships, get global effects out.



Network joint probability

$$P(x, y) = \frac{1}{Z} \prod_{i,j} \Psi(x_i, x_j) \prod_i \Phi(x_i, y_i)$$

scene

image

Scene-scene compatibility function

neighboring scene nodes

Image-scene compatibility function

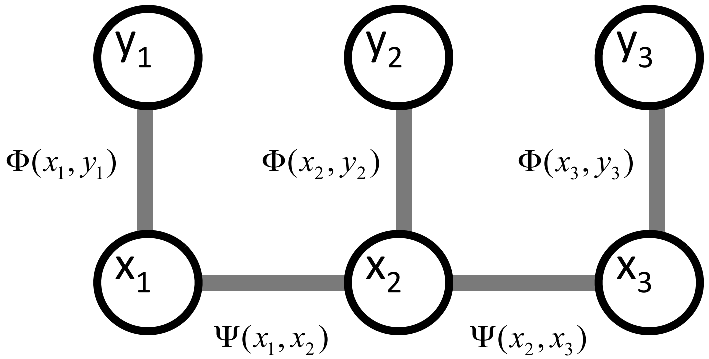
local observations

Inference in MRF's

- Inference in MRF's. (given observations, how infer the hidden states?)
  - Gibbs sampling, simulated annealing
  - Iterated condtional modes (ICM)
  - Variational methods
  - Belief propagation
  - Graph cuts

See www.ai.mit.edu/people/wtf/learningvision for a tutorial on learning and vision.

Derivation of belief propagation

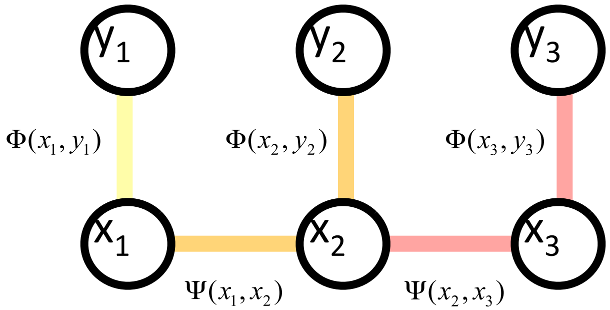


$$x_{1MMSE} = \underset{x_1}{\text{mean}} \ \underset{x_2}{\text{sum}} \ \underset{x_3}{\text{sum}} \ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

The posterior factorizes

$$x_{1MMSE} = \underset{x_1}{\mathrm{mean}}\ \underset{x_2}{\mathrm{sum}}\ \underset{x_3}{\mathrm{sum}}\ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

$$= \underset{x_1}{\mathrm{mean}}\ \underset{x_2}{\mathrm{sum}}\ \underset{x_3}{\mathrm{sum}}\ \Phi(x_1, y_1)\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$

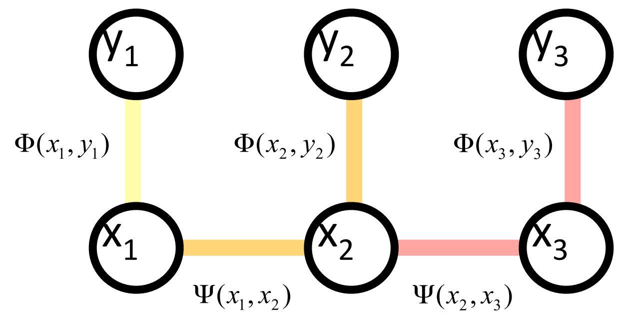


Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ \Phi(x_1, y_1)\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \Phi(x_1, y_1)\ \underset{x_2}{\text{sum}}\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \underset{x_3}{\text{sum}}\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$



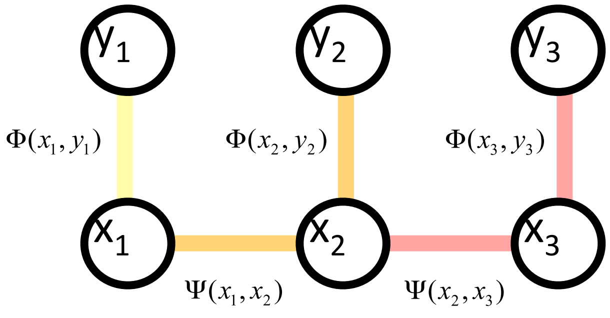
Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1, y_1)$$

$$\underset{x_2}{\text{sum}}\, \Phi(x_2, y_2)\, \Psi(x_1, x_2)$$

$$\underset{x_3}{\text{sum}}\, \Phi(x_3, y_3)\, \Psi(x_2, x_3)$$

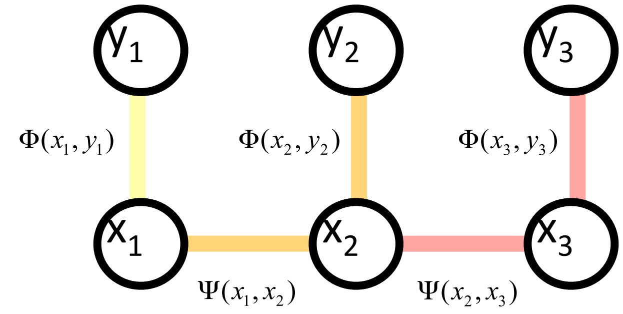
$$M_1^2(x_1) = \underset{x_2}{\text{sum}}\; \Psi(x_1, x_2)\;\; \Phi(x_2, y_2)\;\; M_2^3(x_2)$$



Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1, y_1)\, \underset{x_2}{\text{sum}}\, \Phi(x_2, y_2)\, \Psi(x_1, x_2)\, \boxed{\underset{x_3}{\text{sum}}\, \Phi(x_3, y_3)\, \Psi(x_2, x_3)}$$

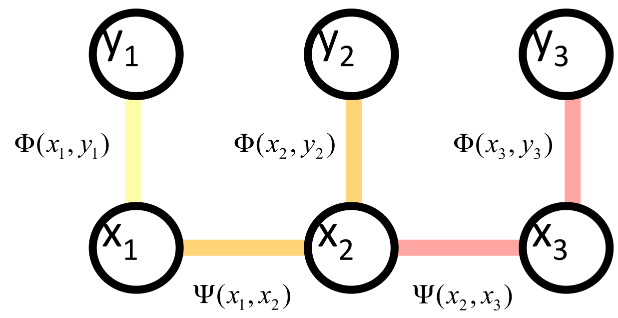
$$M_1^2(x_1) = \underset{x_2}{\text{sum}}\, \Psi(x_1, x_2)\ \Phi(x_2, y_2)\ \boxed{M_2^3(x_2)}$$



Propagation rules

$$x_{1MMSE} = \operatorname*{mean}_{x_1} \Phi(x_1, y_1) \operatorname*{sum}_{x_2} \Phi(x_2, y_2)\, \Psi(x_1, x_2) \operatorname*{sum}_{x_3} \Phi(x_3, y_3)\, \Psi(x_2, x_3)$$

$$\boxed{M_1^2(x_1)} = \operatorname*{sum}_{x_2} \Psi(x_1, x_2)\; \Phi(x_2, y_2)\, \boxed{M_2^3(x_2)}$$



Belief propagation: the nosey neighbor

"Given everything I've heard, and I know how you think about things, here's what <u>you</u> should think."

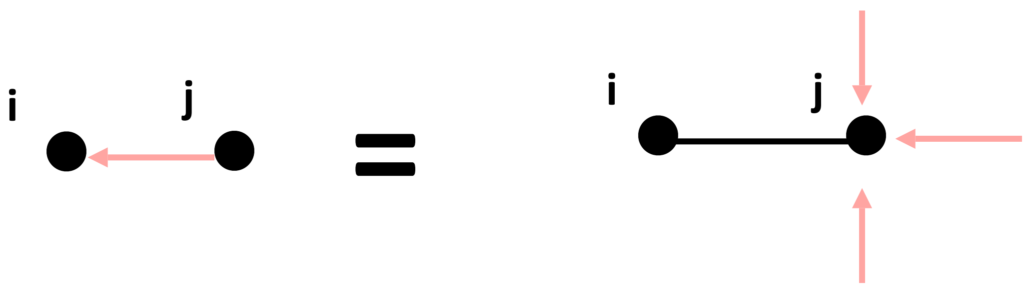
(Given the probabilities of my being in different states, and how my states relate to your states, here's what I think the probabilities of your states should be)

Belief propagation messages

<u>A message</u>: can be thought of as a set of weights on each of your possible states

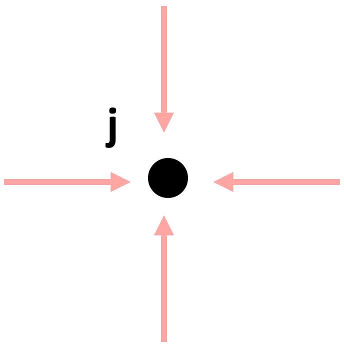
<u>To send a message</u>: Multiply together all the incoming messages, except from the node you're sending to,
then multiply by the compatibility matrix and marginalize over the sender's states.

$$M_i^j(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in N(j) \setminus i} M_j^k(x_j)$$



Beliefs

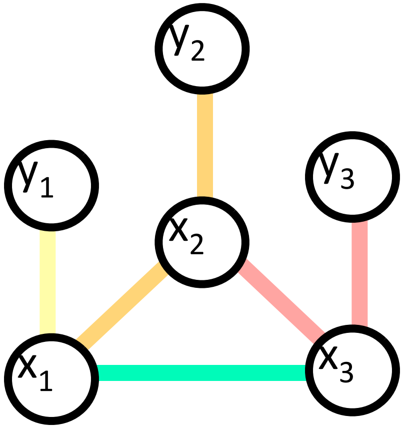
To find a node's beliefs: Multiply together all the messages coming in to that node.



$$b_j(x_j) = \prod_{k \in N(j)} M_j^k(x_j)$$

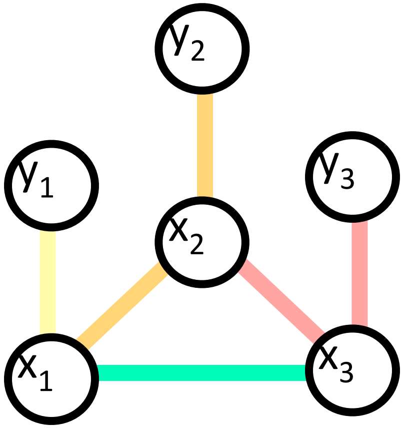
Optimal solution in a chain or tree:

- "Do the right thing" Bayesian algorithm.
- For Gaussian random variables over time: Kalman filter.
- For hidden Markov models: forward/backward algorithm (and MAP variant is Viterbi).



y_2
y_1
y_3
x_2
x_1
x_3

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \Phi(x_1, y_1)\ \underset{x_2}{\text{sum}}\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \underset{x_3}{\text{sum}}\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)\ \Psi(x_1, x_3)$$

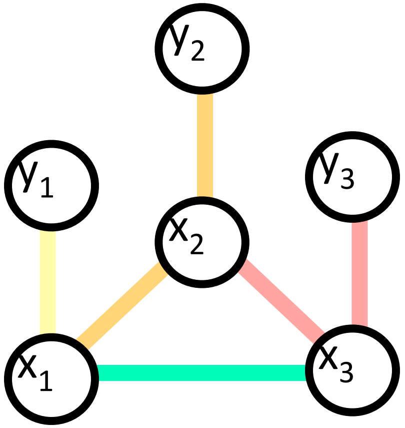


No factorization with loops!

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1, y_1)$$

$$\underset{x_2}{\text{sum}}\, \Phi(x_2, y_2)\, \Psi(x_1, x_2)$$

$$\underset{x_3}{\text{sum}}\, \Phi(x_3, y_3)\, \Psi(x_2, x_3)\; \Psi(x_1, x_3)$$

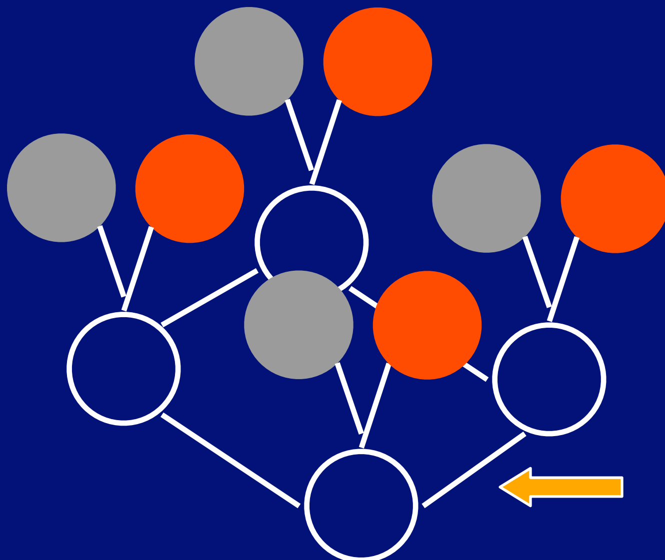


Justification for running belief propagation in

- Experimental results:
  - Error-correcting codes — Kschischang and Frey, 1998; McEliece et al., 1998
  - Vision applications — Freeman and Pasztor, 1999; Frey, 2000
- Theoretical results:
  - For Gaussian processes, means are correct.
    Weiss and Freeman, 1999
  - Large neighborhood local maximum for MAP.
    Weiss and Freeman, 2000
  - Equivalent to Bethe approx. in statistical physics.
    Yedidia, Freeman, and Weiss, 2000

Propagating Information

- Extend probability model to consider relationship between neighboring derivatives



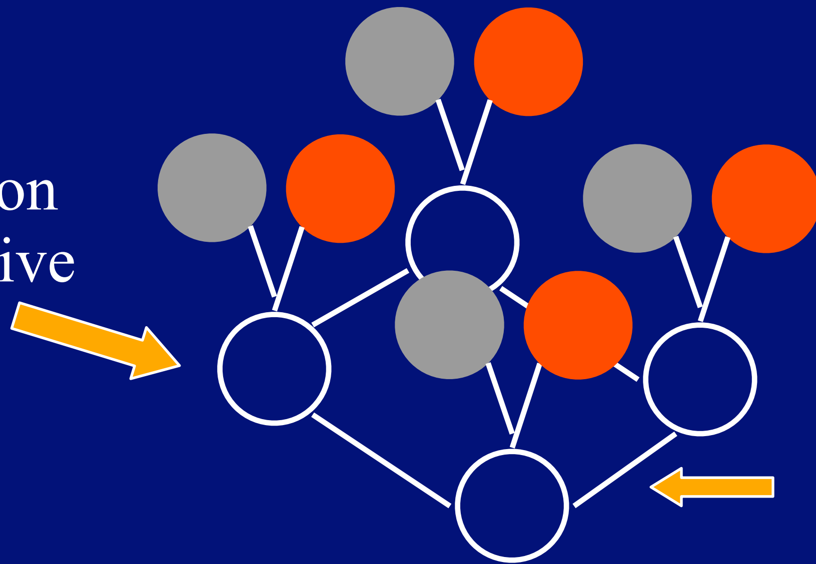
$$\psi(x_i, x_j) = \begin{bmatrix} \beta & 1-\beta \\ 1-\beta & \beta \end{bmatrix}$$

- β controls how necessary it is for two nodes to have the same label
- Use Generalized Belief Propagation to infer labels. (Yedidia et al. 2000)

Propagating Information

- Extend probability model to consider relationship between neighboring derivatives

Classification of a derivative



$$\psi(x_i, x_j) = \begin{bmatrix} \beta & 1-\beta \\ 1-\beta & \beta \end{bmatrix}$$

- β controls how necessary it is for two nodes to have the same label

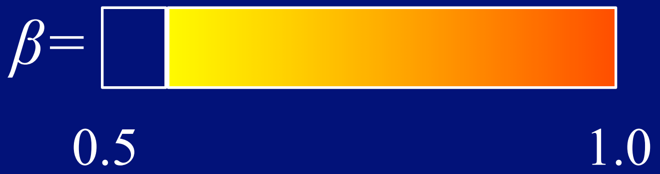
- Use Generalized Belief Propagation to infer labels. (Yedidia et al. 2000)

Setting Compatibilities

- All compatibilities have form

$$\psi(x_i, x_j) = \begin{bmatrix} \beta & 1-\beta \\ 1-\beta & \beta \end{bmatrix}$$

- Assume derivatives along image contours should have the same label
- Set β close to 1 when the derivatives are along a contour
- Set β to 0.5 if no contour is present
- β is computed from a linear function of the image gradient's magnitude and orientation

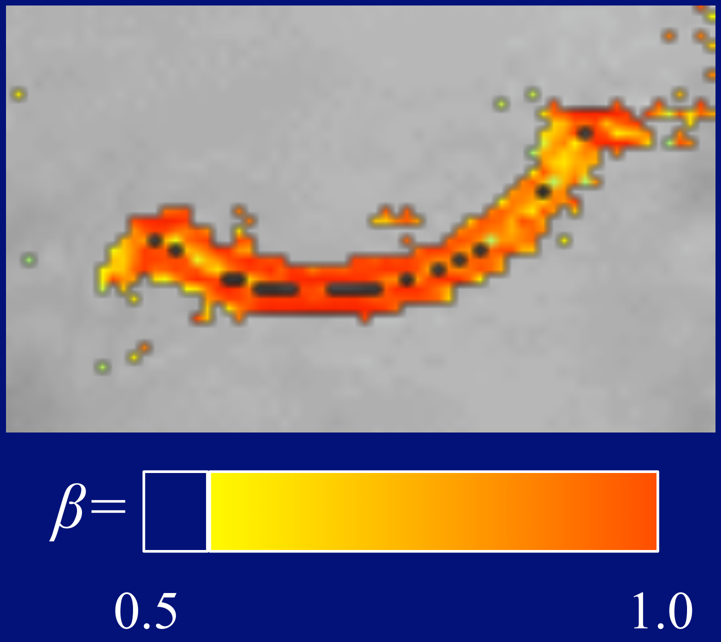


Setting Compatibilities

- All compatibilities have form

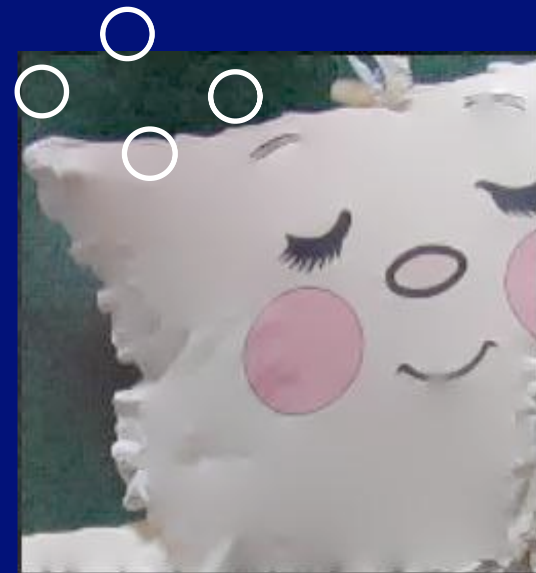
$$\psi(x_i, x_j) = \begin{bmatrix} \beta & 1-\beta \\ 1-\beta & \beta \end{bmatrix}$$

- Assume derivatives along image contours should have the same label
- Set β close to 1 when the derivatives are along a contour
- Set β to 0.5 if no contour is present
- β is computed from a linear function of the image gradient's magnitude and orientation

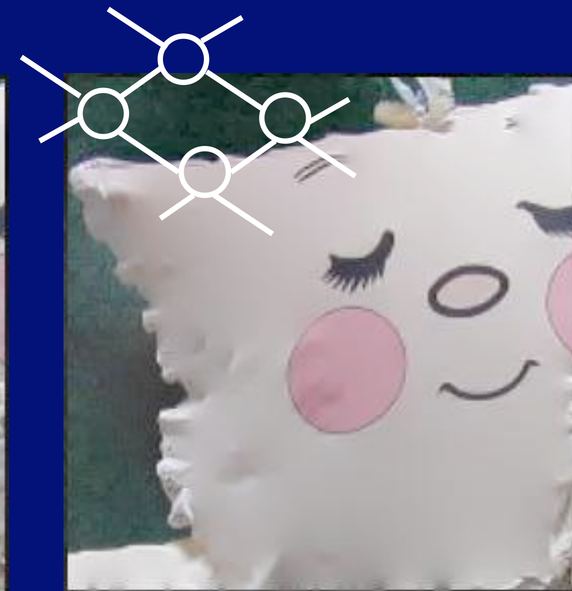


Improvements Using Propagation

Input Image



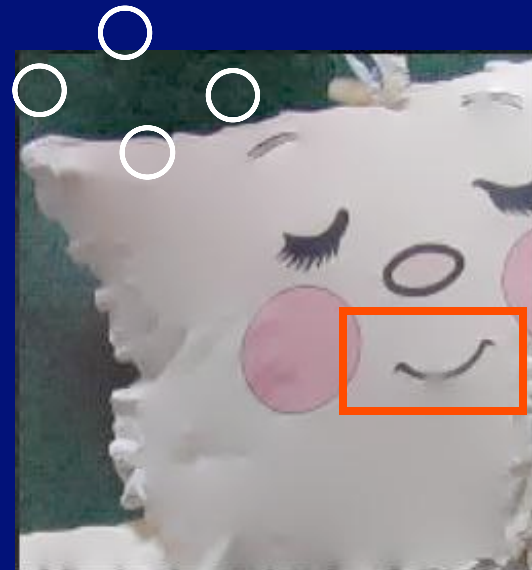
Reflectance Image
Without Propagation



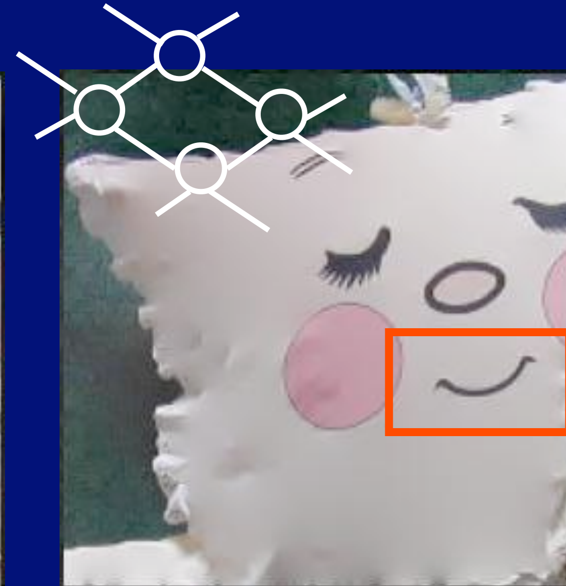
Reflectance Image
With Propagation

Improvements Using Propagation

Input Image



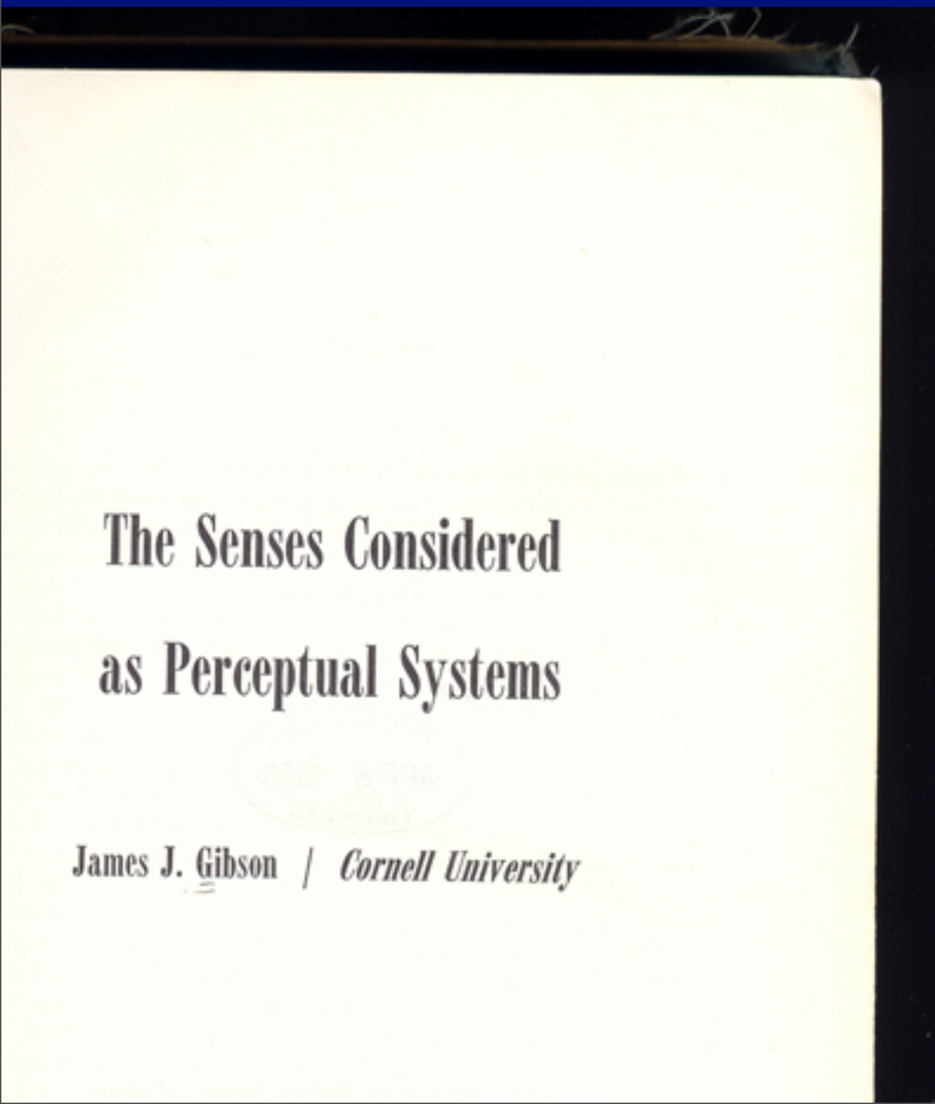
Reflectance Image
Without Propagation



Reflectance Image
With Propagation

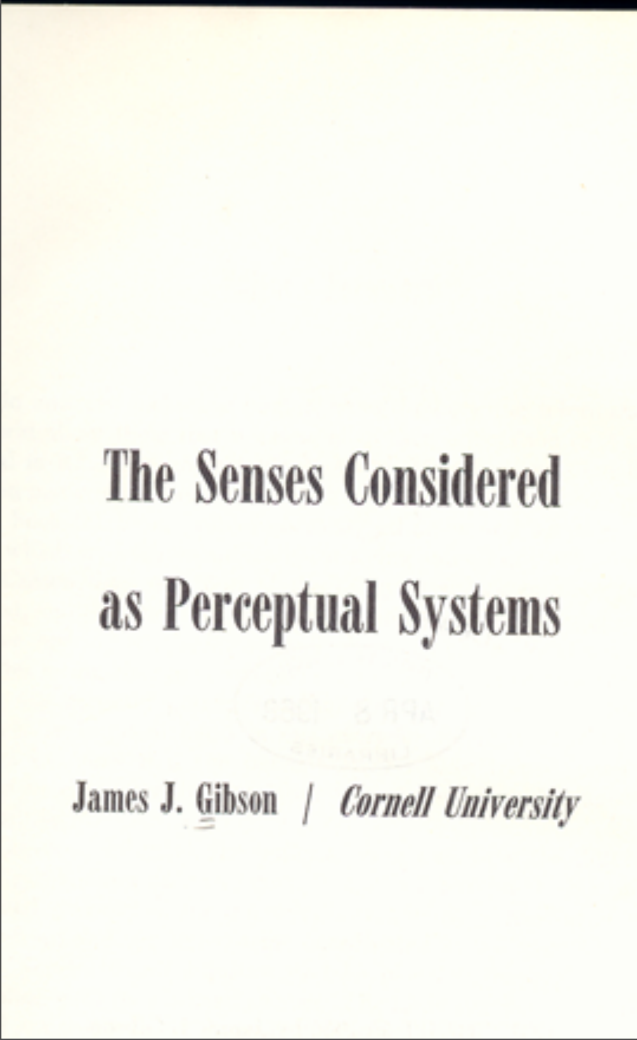
More results…

J. J. Gibson, 1968


The Senses Considered as Perceptual Systems

James J. Gibson / *Cornell University*

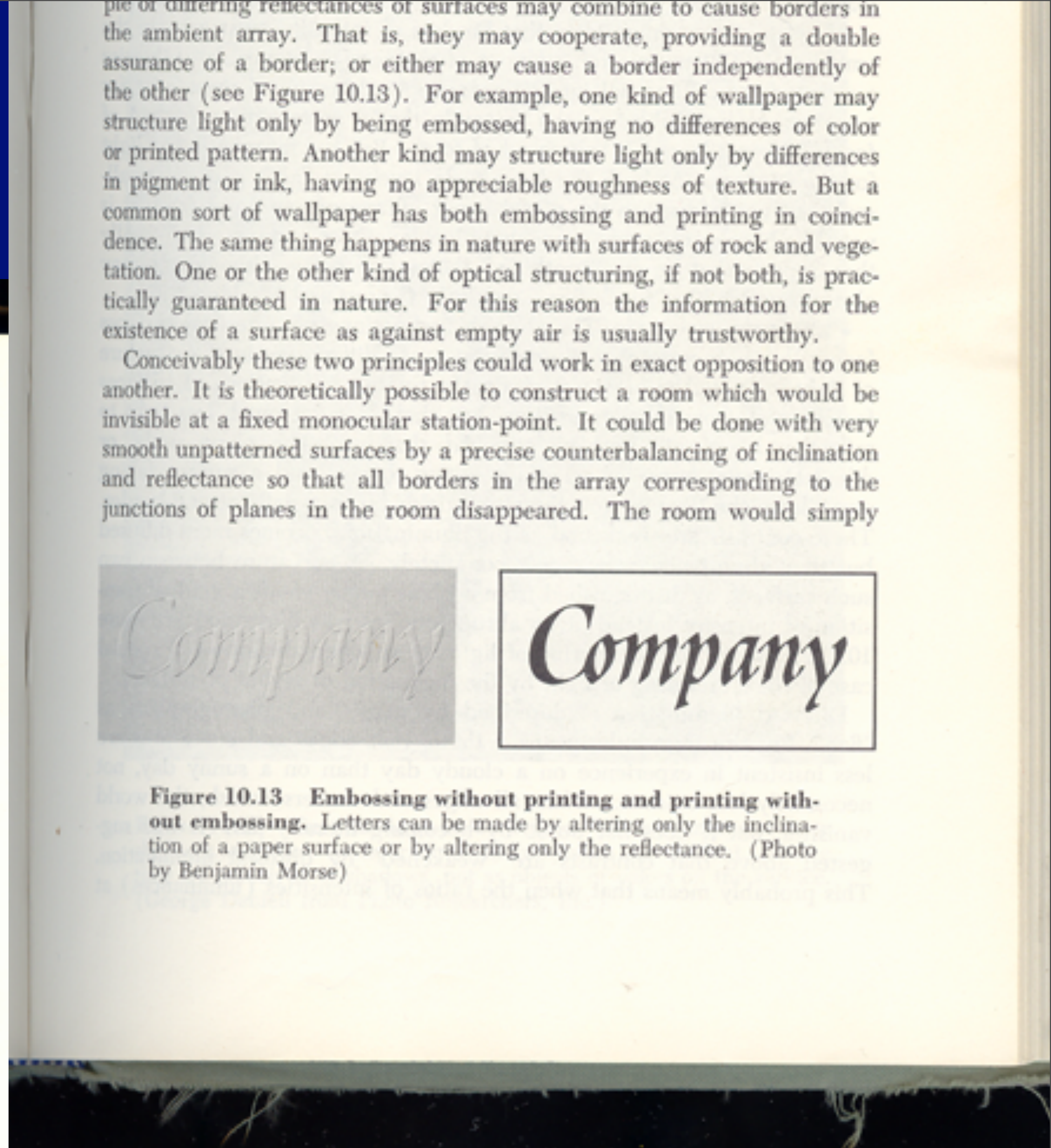
J. J. Gibson, 1968



The Senses Considered

as Perceptual Systems

James J. Gibson / *Cornell University*



ple of differing reflectances of surfaces may combine to cause borders in the ambient array. That is, they may cooperate, providing a double assurance of a border; or either may cause a border independently of the other (see Figure 10.13). For example, one kind of wallpaper may structure light only by being embossed, having no differences of color or printed pattern. Another kind may structure light only by differences in pigment or ink, having no appreciable roughness of texture. But a common sort of wallpaper has both embossing and printing in coincidence. The same thing happens in nature with surfaces of rock and vegetation. One or the other kind of optical structuring, if not both, is practically guaranteed in nature. For this reason the information for the existence of a surface as against empty air is usually trustworthy.

Conceivably these two principles could work in exact opposition to one another. It is theoretically possible to construct a room which would be invisible at a fixed monocular station-point. It could be done with very smooth unpatterned surfaces by a precise counterbalancing of inclination and reflectance so that all borders in the array corresponding to the junctions of planes in the room disappeared. The room would simply

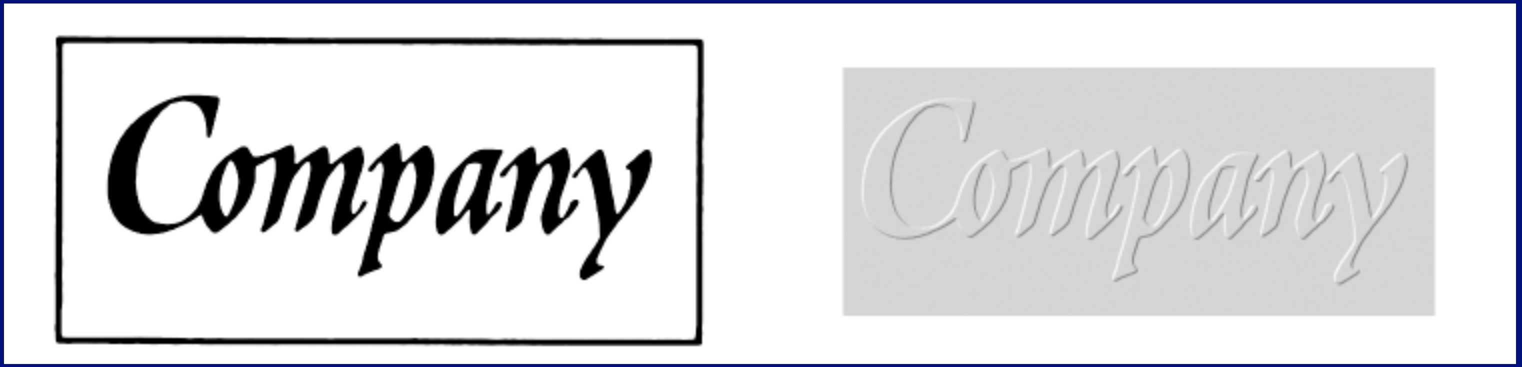
Company　Company

Figure 10.13 Embossing without printing and printing without embossing. Letters can be made by altering only the inclination of a paper surface or by altering only the reflectance. (Photo by Benjamin Morse)

Gibson image

Gibson image

original



Gibson image

original

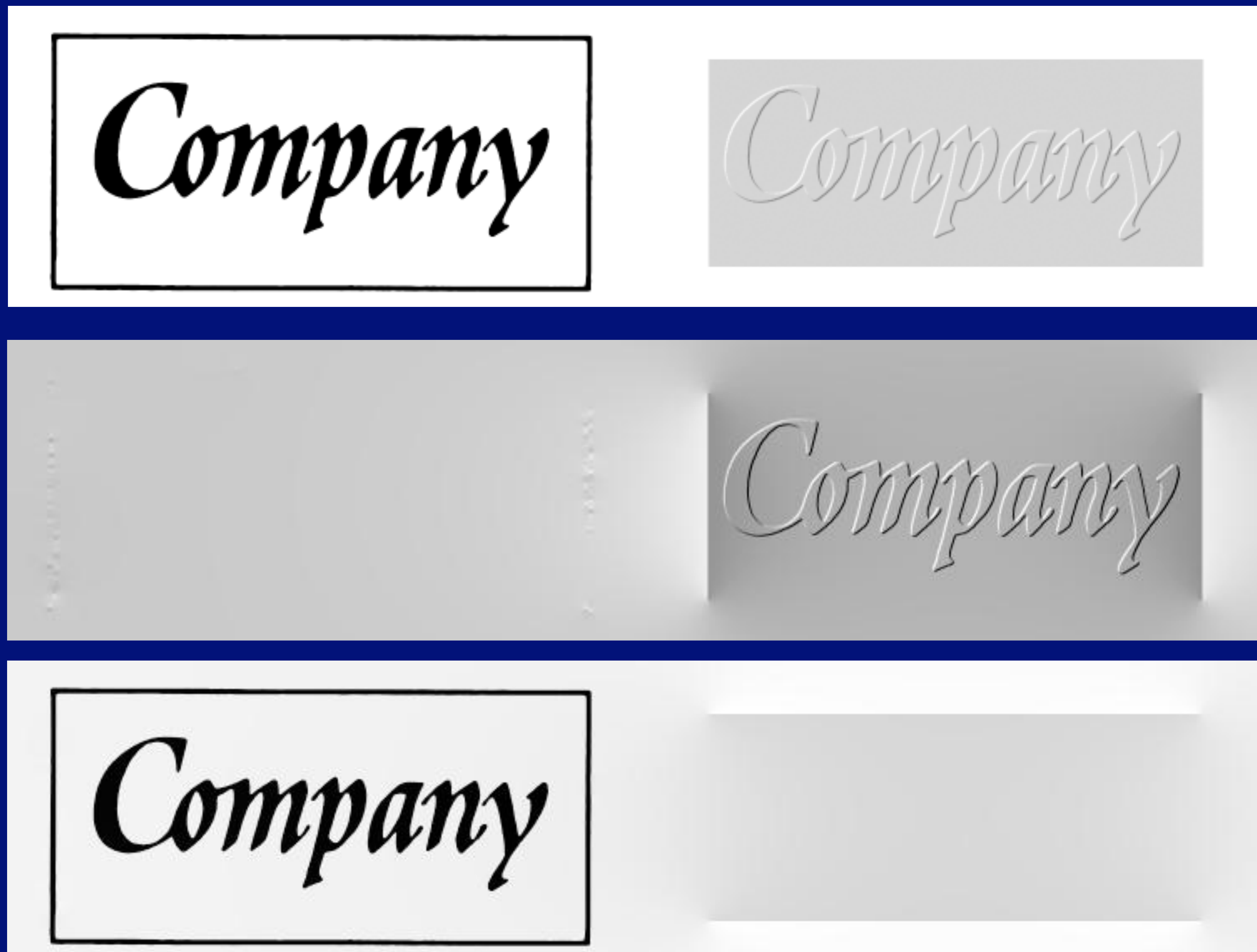
shading

Gibson image

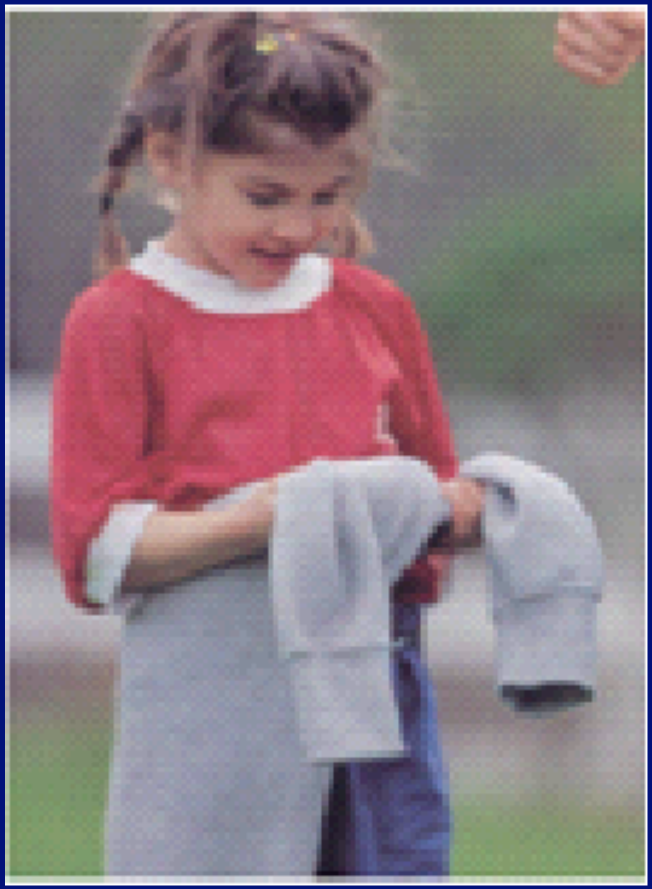
original

shading

reflectance



Clothing catalog image



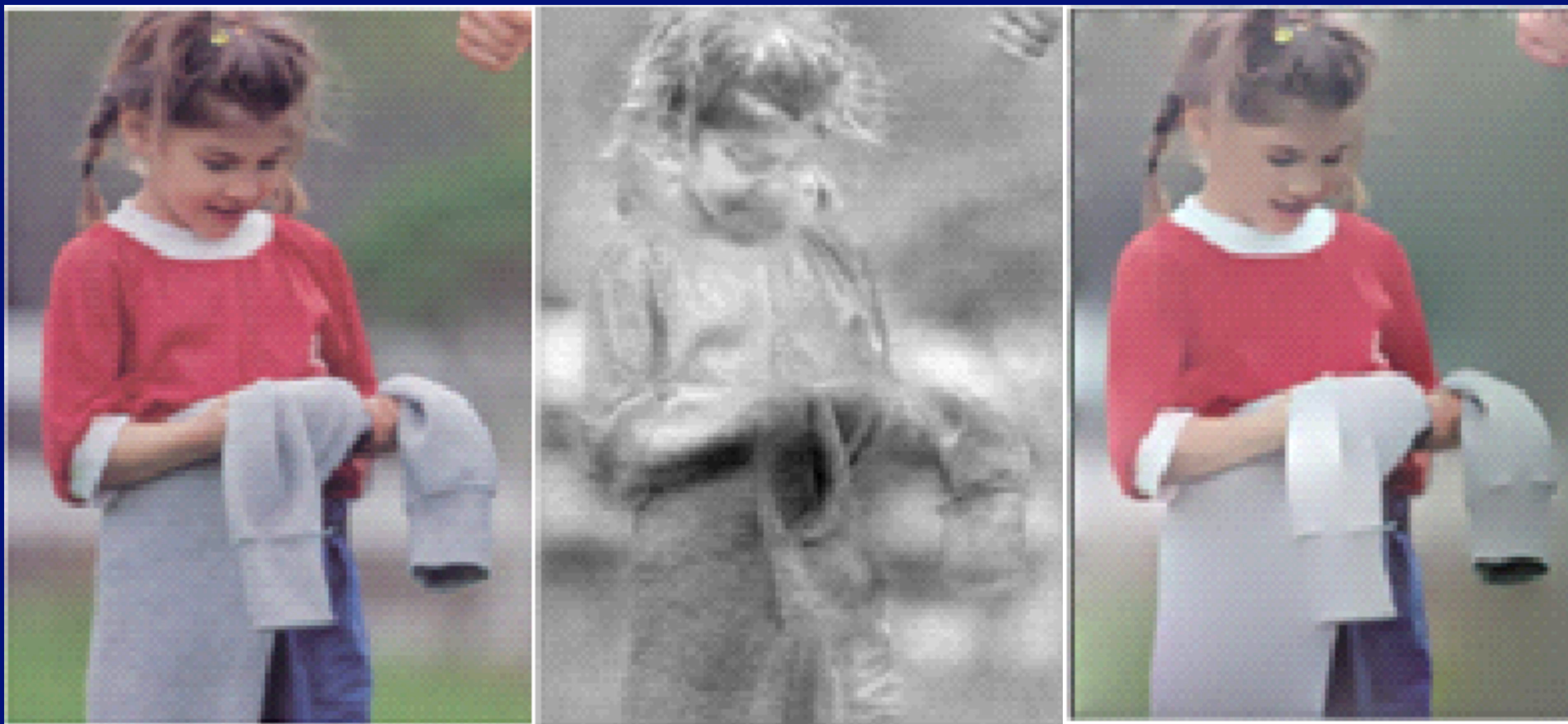
Original
(from LL Bean catalog)

Clothing catalog image

Original
(from LL Bean catalog)

Shading

Clothing catalog image



Original
(from LL Bean catalog)

Shading

Reflectance

Sign at train crossing

Separated images

original

Separated images

original

shading

Separated images

original

shading

reflectance

Separated images

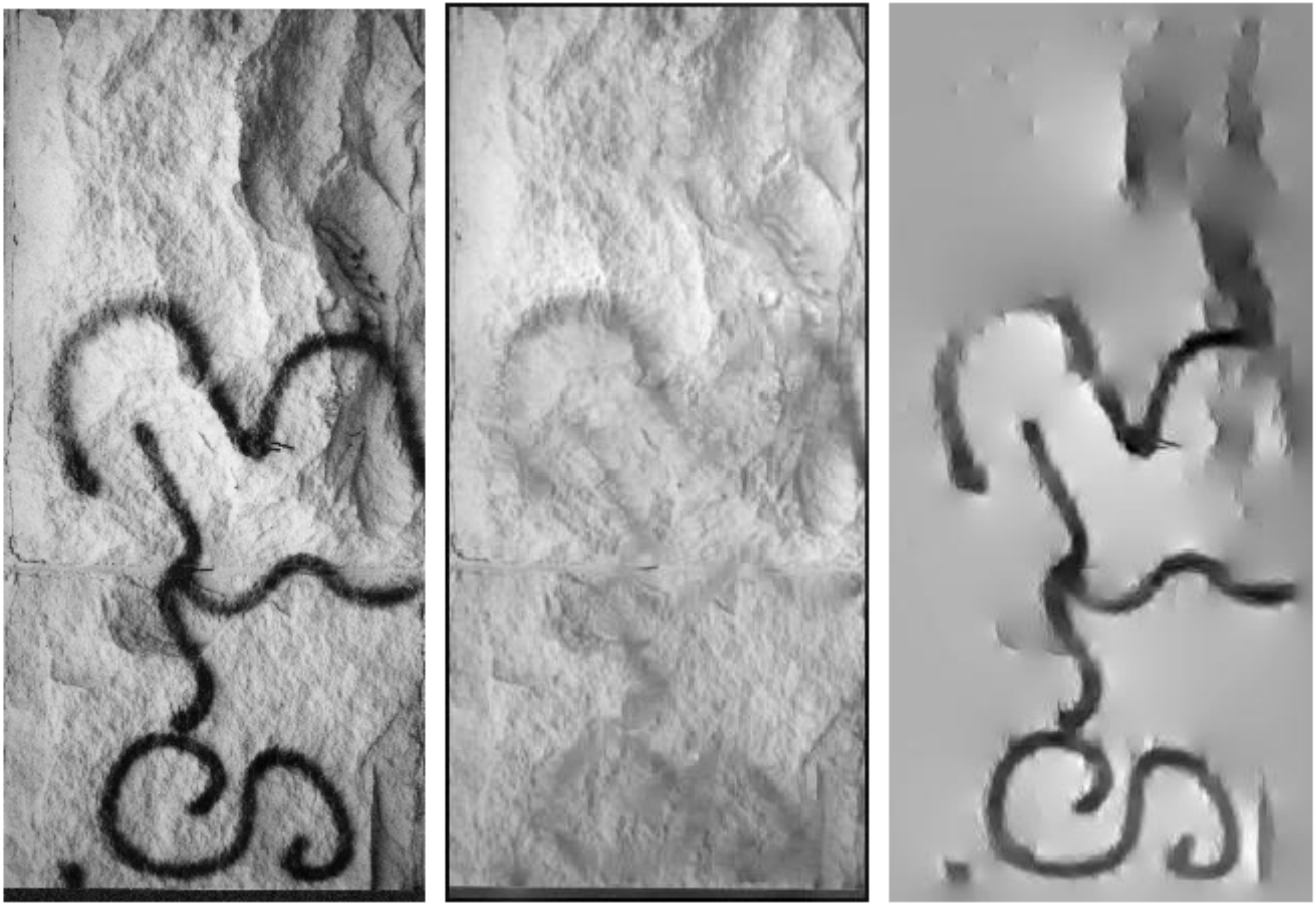
original

shading

reflectance

Note: color cue omitted for this processing

(a) Original Image



(a) Original Image (b) Shape Image (c) Reflectance Image

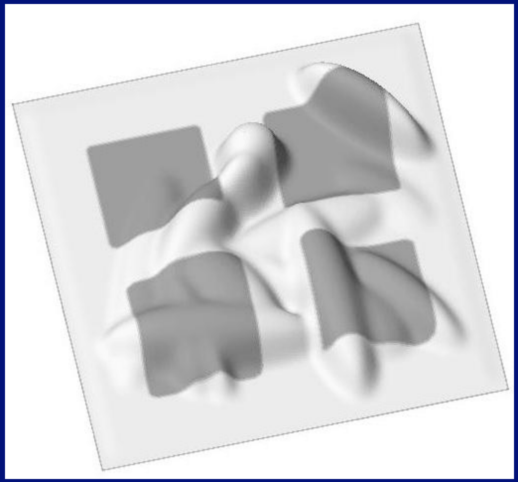
Finally, returning to our explanatory example…

input

Ideal shading image

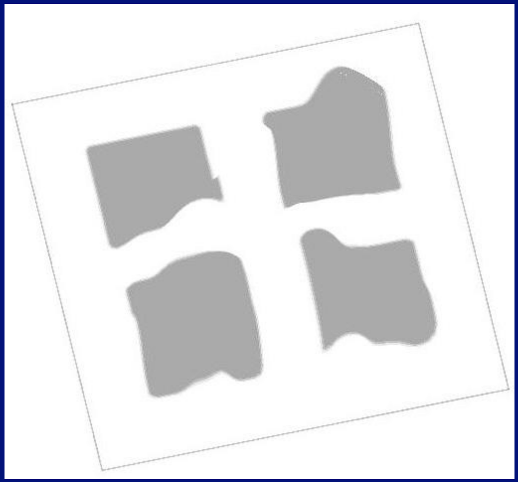
Ideal paint image

Finally, returning to our explanatory example…



input

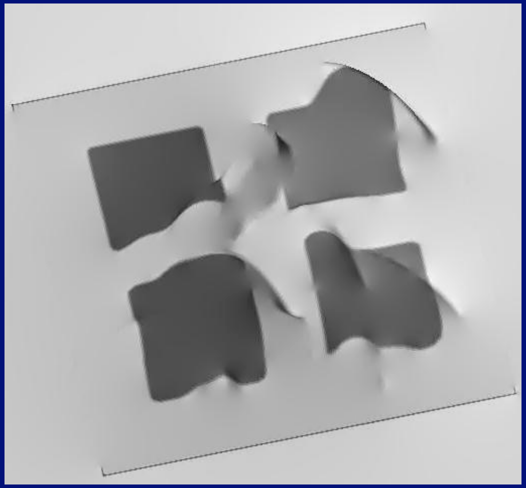
Ideal shading image



Ideal paint image

Algorithm output.

Note: occluding edges labeled as reflectance.



Separating shading from paint

- From a single image:
  - identify all-shading versus all-paint
  - locally separate shading from paint
- <u>From a sequence of images</u>:
  - separate stable from varying component
- From a stereo pair
  - separate shading, paint, occlusion.

Yair Weiss's ICCV 2001 paper

Untitled13

In: Proc ICCV (2001)

Deriving intrinsic images from image sequences

Yair Weiss
Computer Science Division
UC Berkeley
Berkeley, CA 94720-1776
yweiss@cs.berkeley.edu

Abstract

Intrinsic images are a useful midlevel description of scenes proposed by Barrow and Tenenbaum [1]. An image is de-

based template matching and shape-from-shading would be significantly less brittle if they could work on the intrinsic image representation rather than on the input image.

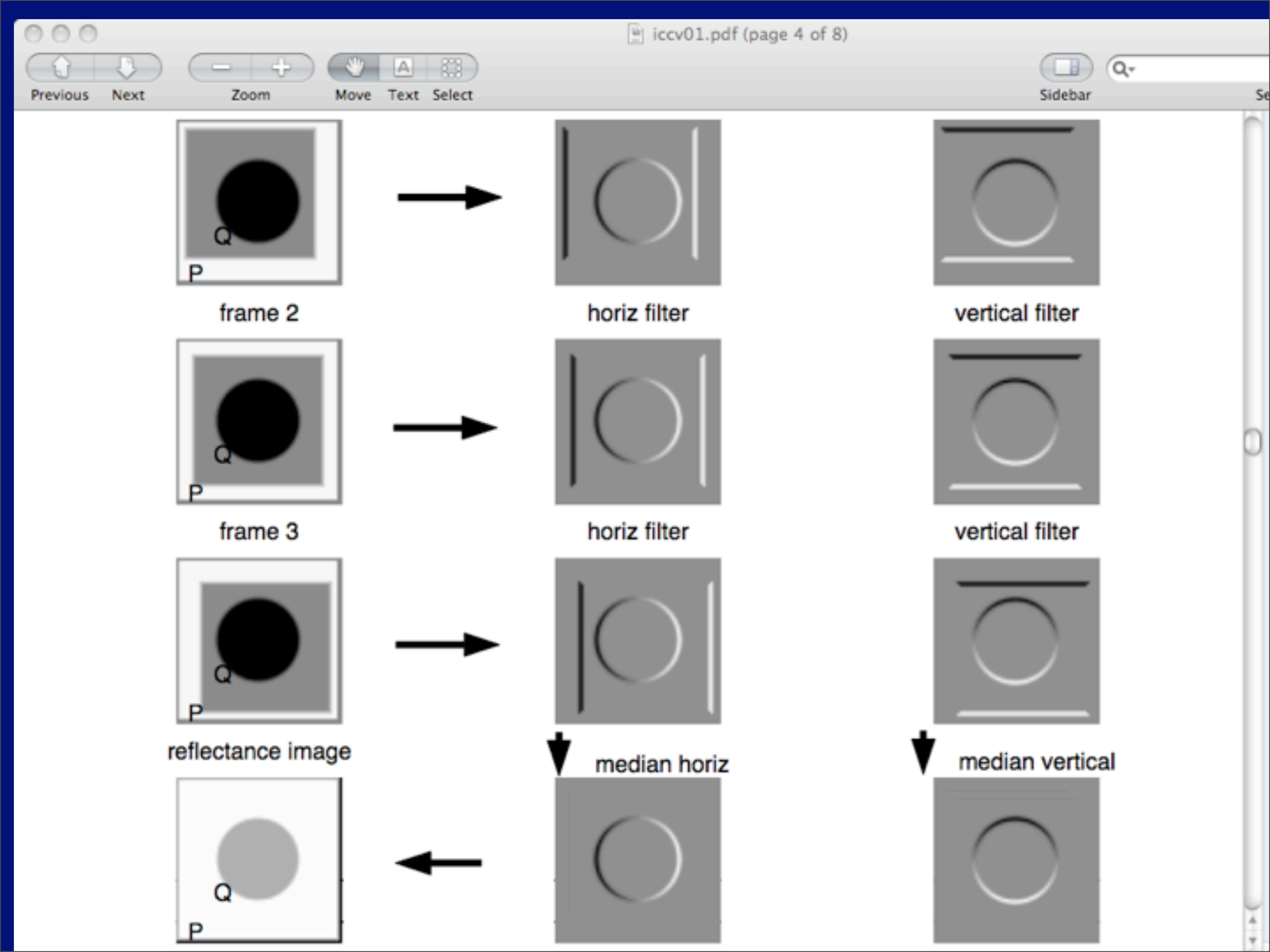
Recovering two intrinsic images from a single input im-

8.50 x 11.00 in

Assume multiple images where reflectance is constant but lighting varies

Untitled1

ure 2: Images from a “webcam” at www.berkeley.edu/webcams/sproul.html. Most of the changes are changes
mination. Can we use such image sequences to derive intrinsic images?



iccv01.pdf (page 4 of 8)
Previous
Next
Zoom
Move
Text
Select
Sidebar
Q
P
frame 2
horiz filter
vertical filter
Q
P
frame 3
horiz filter
vertical filter
Q
P
reflectance image
median horiz
median vertical
Q
P

synthetic example

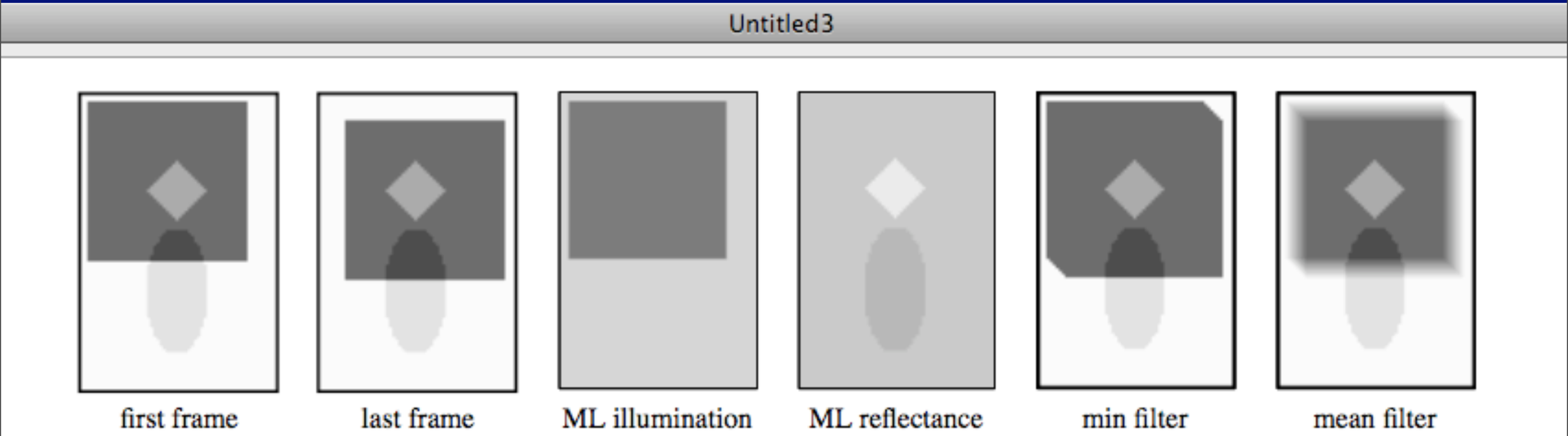
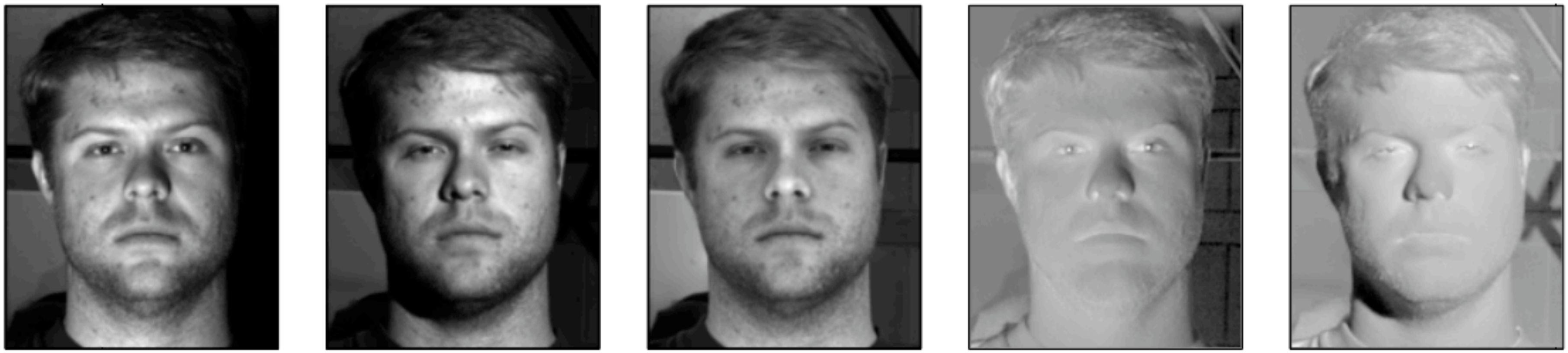


Figure 5: A synthetic sequence in which a square cast shadow translates diagonally. Note that the pixels surrounding the diamond are always in shadow, yet their estimated reflectance is the same as that of pixels that were always in light. In the min and mean filters, this is not the case and the estimated reflectances are quite wrong.

Result from Yair's multi-image algorithm

Untitled15



frame 2 | frame 11 | ML reflectance | ML illumination 2 | ML illumination 11

Figure 7: Results on one face from the Yale Face Database B [5]. There were 64 images taken with variable lighting. Note that the recovered reflectance image is almost free of specularities and is free of cast shadows. The ML illumination images are shown with a logarithmic nonlinearity to increase dynamic range.

Result from Yair's multi-image algorithm

Untitled1

frame 1

frame 11

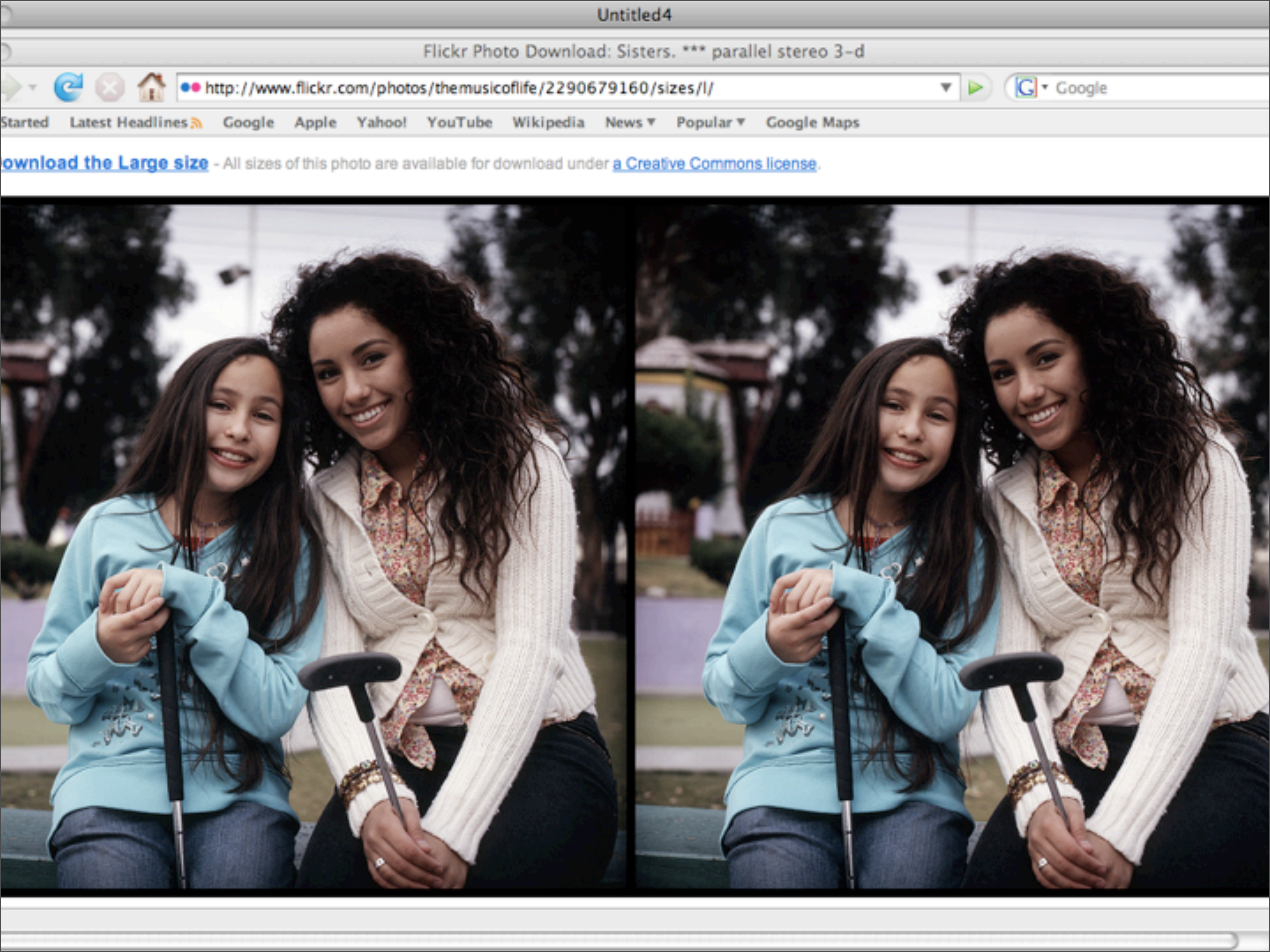
ML reflectance

ML illumination 1

ML illumination 2

Separating shading from paint

- From a single image:
  - identify all-shading versus all-paint
  - locally separate shading from paint
- From a sequence of images:
  - separate stable from varying component
- <u>From a stereo pair</u>
  - separate shading, paint, occlusion.


Untitled4
Flickr Photo Download: Sisters. *** parallel stereo 3-d
http://www.flickr.com/photos/themusicoflife/2290679160/sizes/l/
Google
Started
Latest Headlines
Google
Apple
Yahoo!
YouTube
Wikipedia
News
Popular
Google Maps
ownload the Large size - All sizes of this photo are available for download under a Creative Commons license.

Intrinsic images from stereo

- Input: stereo pair (from Flickr or other)
- Output: shading image, reflectance image, material/lighting parameters for different regions, occluding contours.

- This may help make stereo better (fewer unexplained phenomena). And could provide a great training set for the monocular image intrinsic image problem.