Computational Vision
U. Minn. Psy 5036
Daniel Kersten
Lecture 17: Shape from shading

Initialize

```
Off[General::spell1];
SetOptions[ArrayPlot, ColorFunction → "GrayTones",
  DataReversed → True, Frame → False, AspectRatio → Automatic,
  Mesh → False, PixelConstrained → True, ImageSize → Small];
SetOptions[ListPlot, ImageSize → Small];
SetOptions[Plot, ImageSize → Small];
SetOptions[Plot3D, ImageSize → Small, ColorFunction → "GrayTones"];
SetOptions[DensityPlot, ImageSize → Small, ColorFunction → GrayLevel];
nbinfo = NotebookInformation[EvaluationNotebook[]];
dir = ("FileName" /. nbinfo /. FrontEnd`FileName[d_List, nam_, ___] ⧴ ToFileName[d]);
```

Outline

Last time

Inference of shape from shading--set in context of other cues to shape,

Perception of shape from shading

Introduce generative models for the inverse problem "shape from shading":

$L_R = \boldsymbol{re}\, \hat{N} \cdot\ \hat{E}$

Invert the forward model: $L_M(p_n,q_n) = \mathbf{n.e} = \frac{\{p_n,q_n,-1\}.\{pe,qe,-1\}}{\sqrt{(p_n{}^2+q_n{}^2+1)(pe^2+qe^2+1)}}$

Classic Ikeuchi & Horn solution:

Assume light source direction is known. Constant reflectance. Known surface normals at the smooth occluding boundary.

Find local surface orientation (p's and q's) such that E_D+ E_S is small.

Generative model:

1) data cost function given by

$$E_D(\mathrm{p(x,y), q(x,y)}) = (L_M(p_n, q_n) - L_D(x, y))^2$$

measures the discrepancy between the data, i.e. intensity at a point L_D(x,y) and the model prediction, L_M(pn,qn)
based on the current guess of orientation. Note that p_n, q_n are functions of x: $p_n(x, y)$, q_n(x,y).

2) Prior assumption of surface smoothness insist that guesses of orientation also encourage small values of a measure
of smoothness E_S. (i.e. spatial second derivatives of p(x,y) and q(x,y) should be small). This gives a data-independent
part of the cost function:

$$E_S\left((\nabla^2 p)^2,\ \nabla^2 q^{\,2}\right).$$

One can construct a local cost function:

$$e(x, y) = (L(x, y) - L_R(p, q))^2 + \lambda((\nabla^2 p)^2 + (\nabla^2 q)^2)$$

where we require e(x,y) to be small. But we want this to be the case over the whole image, so the grand goal is to find values of p(x,y) and q(x,y) over the whole image to make the sum of all local costs to be as small as possible:

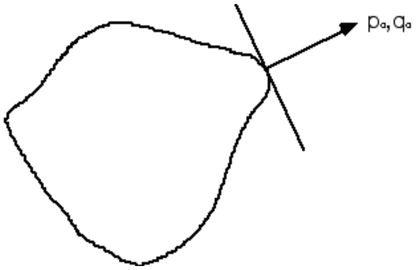
$$e|p(x, y),\ q(x, y| = \iint (L(x, y) - L_R(p, q))^2 + \lambda((\nabla^2 p)^2 - (\nabla^2 q)^2)\,dx\,dy$$

These kinds of minimization problems require numerical optimization routines, for example that implement gradient descent.

The idea is to start off with random values of p(x,y) and q(x,y), calculate -∇e(p,q) at each x.

This vector points in the direction that the error changes most rapidly--the steepest direction to take down this high-dimensional mountain. One then updates the values of p,q to reduce e, and then repeats the step until convergence.

If one measures the project values of p and q and the image boundary, these can be used at startup, and fixed throughout:



Note that cost minimization can be formulated as Bayes maximum a posteriori estimation of surface normals, with a smoothness prior probability:

$$p(\mathbf{n} \mid L) \propto p(L \mid \mathbf{n})\, p(\mathbf{n}) = e^{-E_D}\; e^{-E_S} = e^{-(E_D + E_S)}$$

Other representations of local shape, e.g. slant/tilt, can be used.

Today

Other approaches to shape from shading

Perceptual ambiguities in bas-relief sculptures

More on task analysis, integrating out and the genericity principle

Solutions to the shape from shading problem for the linear case

There is a general result that if your cost function is quadratic, then the minimum may be found using a matrix operator--in other words, there is a linear filter solution.

Linear shape from shading

Linear generative model

$$L_R(p,q) = \frac{r(pp_e + qq_e + 1)}{\sqrt{(p^2 + q^2 + 1)(p_e^{\,2} + q_e^{\,2} + 1)}}$$

```
Lmodel[pn_,qn_,pe_,qe_]:= ({pn,qn,-1}/Sqrt[pn^2+qn^2 + 1]).({pe,qe,-1}/Sqrt[pe^2+qe^2 + 1]
```

Taylor's series

We can expand the image luminance in a Taylor series about {pn,qn}={0,0}. Recall that this is done by calculating successive derivatives at {pn,qn}={0,0}, and then substituting {pn,qn}->{0,0}.

Mathematica's **Series[]** function puts the Taylor series together for us. Here is the expansion up to linear terms:

```
Series[Lmodel[pn,qn,pe,qe], {pn, 0, 1}, {qn, 0, 1}]
```

$$\left(\frac{1}{\sqrt{1+\text{pe}^2+\text{qe}^2}} + \frac{\text{qe qn}}{\sqrt{1+\text{pe}^2+\text{qe}^2}} + O[\text{qn}]^2\right) + \left(\frac{\text{pe}}{\sqrt{1+\text{pe}^2+\text{qe}^2}} + O[\text{qn}]^2\right)\text{pn} + O[\text{pn}]^2$$

You could improve the approximation by including quadratic terms: **Series[Lmodel[pn,qn,pe,qe], {pn, 0, 2}, {qn, 0, 2}]**. But then one would have to decide whether the non-linearity was worth the added computational complications.

▸ 1. How could you test whether the linear generative model reflects the built-in knowledge humans have about shading?

▸ 2. How could you use the above linear generative model for "shading from shape" to devise a solution to the inverse problem of "shape from shading"?

(See Appendix in previous lecture. And also Appendix below for an inverse solution using the Fourier Transform.)

Learning scene parameters from images

If one has available a set of images $\{L_i\}$, together with a set of unit surface normals for each image $\{p_i, q_i\}$ (e.g. derived from a set of range images), then one could do statistical regression to find a mapping. A linear model suggests that one could simply do a linear regression to find a map: L -> $\{p, q\}$, that might be a workable approximation to shape from shading from an image L that was not in the training class $\{L_i\}$.

One early application used a neural network-like technique (Widrow-Hoff learning rule for weight or "synaptic" modification) to do regression learning "on-line", rather than in "batch mode" (Knill & Kersten, 1990). This method was general and empirical in the sense, that one can imagine applying it to other scene-from-image problems (Kersten et al., 1988). How well it works depends on the linear approximation. Non-linear neural networks and learning techniques such as error back-propagation have also been used to find solutions for non-linear formulations (Lehky & Sejnowski, 1988). Freeman et al. (2000) developed the idea of supervised learning of scene properties from images and applied it to super-resolution.

For recent applications of machine learning to estimating scenes, see: http://papers.nips.cc/paper/5851-deep-convolutional-inverse-graphics-network,
https://www.cv-foundation.org//openaccess/content_iccv_ 2015_workshops/w10/papers/Shelhamer_Scene_Intrinsics _and _ICCV _ 2015_paper.pdf

Shading & the bas-relief transform

Generalizing the simple lambertian generative model

The lambertian model thus far is very limiting. Even apart from the restriction to a certain type of uniform matte surface, it doesn't take into account *attached* or *cast* shadows or multiple light sources.

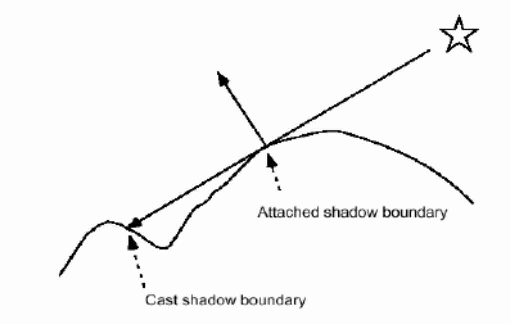
An *attached shadow* occurs whenever the surface normal is at an angle greater than 90 degrees relative to the light source vector. The cosine of an oblique angle is negative, but we can't have negative light. If there is no ambient light, then an attached shadow region is black. So a better model would be:

$$L = \text{Max}[\mathbf{n} \cdot \mathbf{e}, 0].$$

We may have more than one light source, and since light intensity is additive, we can sum up all the contributions:

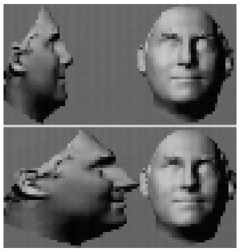
$$L = \sum_i \text{Max}[\boldsymbol{n} \cdot \boldsymbol{e}_i, 0]$$

Note: A *cast shadow* boundary occurs on a receiving surface whenever another surface (or part of one) is between the receiving surface and the light source.



The bas-relief effect

An illustration of surface z-compression/light source slant ambiguity



The above top right and bottom right images are identical. Yet, the top right image is rendered from a surface shape with the profile shown on the top left, and the bottom right image is rendered with the elongated surface profile shown on the bottom left. The only difference in the rendering conditions for the right two images is that the light source slant is different.

The pictures below illustrate the bas-relief ambiguity. Belhumeur, P. N., Kriegman, D. J., & Yuille (1997) showed that there is a simple transformation of a surface (i.e. adding a slanted and tilted plane to a surface z(x,y) and a compression in depth) that induces (together with an albedo and illumination adjustment) a well-defined equivalence class of surfaces for a given image:

$$z(x; y) = \lambda z(x; y) + \mu x + \nu y.$$

For each image of a lambertian surface z(x,y), there is an identical image of a bas-relief produced by a transformed light source. Further this holds for both shaded and shadowed regions. And for the classical bas-relief in sculpture (no added slant, just compression), the image is insensitive to small motions and illumination changes.

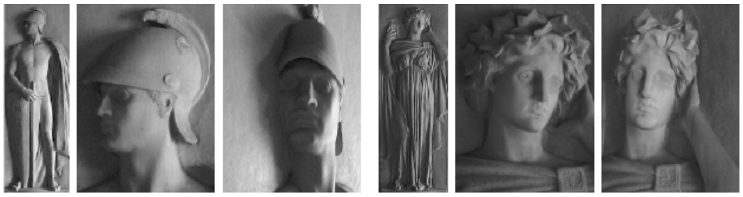


Figure 1: Frontal and side views of a pair of marble bas-relief sculptures: Notice how the frontal views appear to have full 3-dimensional depth, while the side views reveal the flattening the sculptures rise only 5 centimeters from the background plane. While subtle shading is apparent on the faces, the shadows on the women's pleats are the dominant perceptual cue in the body.

Belhumeur, Kriegman and Yuille (1999) note:

"Leonardo da Vinci, while comparing painting and sculpture, criticized the realism afforded by reliefs [11]: 'As far as light and shade are concerned low relief fails both as sculpture and as painting, because the shadows correspond to the low nature of the relief, as for example in the shadows of foreshortened objects, which will not exhibit the depth of those in painting or in sculpture in the round.'

*It is true that when illuminated by the same light source a relief surface ($\lambda < 1$) and a surface "in the round" ($\lambda = 1$) will cast different shadows. However, Leonardo's comment appears to overlook the fact that for any classical bas-relief transformation of the surface, there is a corresponding transformation of the light source direction such that the shadows are the same. This is not restricted to classical reliefs but, as we will show, applies equally to the greater set of generalized bas-relief transformation*s."

Task analysis: "Shape *for* X"

Common shape representation vs. task-dependent?

Shape for object recognition

Shape for grasp

Bayesian approach to task analysis applied to shape from shading

Recall that earlier in the course we made the distinction between *primary variables* which we want to estimate explicitly and precisely, and *secondary* (or generic) variables that contribute to the image data, but which should be discounted. In shape from shading, we considered surface normals to be the primary variables, and the illumination to be the secondary variable. Suppose we don't know the light source direction AND we don't want to estimate it explicitly. Somehow we need to "discount" the illumination.

"Discount" in Bayesian terms means to integrate out their contributions to the posterior probability. So suppose that S_{prim} = {surface normals}, and S_{sec} = {light source direction}. Further, suppose we have a model that prescribes p(S_{prim},S_{sec} | I), where I is the image measurement (e.g. I = luminance L). Then if we can do the following integral,

$$p(S_{prim} \mid I) = \int p(S_{prim}, S_{sec} \mid I) dS_{sec}$$

we could then base our decision on p(S_{prim} | I), where the S_{sec} is no longer explicit, but its effects have gotten folded into the posterior p(S_{prim} | I). Bayes rule can be used to express the joint posterior in terms of the likelihood and the prior:

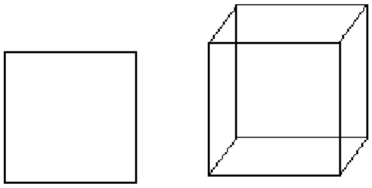
$$p(S_{prim}, S_{sec} \mid I) \propto p(I \mid S_{prim}, S_{sec}) p(S_{prim}, S_{sec})$$

The lambertian shading equation is sufficient to determine the likelihood term. As we've seen, usually one assumes a known light source direction (S_{sec}), and then the prior is some measure that effectively ranks the probability of surfaces based on how smooth or rough they are.
Freeman (1994) showed that this process of "integrating out" or "marginalization" can act to disambiguate the shape in the shape from shading problem. In other words, by assuming up front that illumination direction is a secondary variable, one can find solutions to the shape from shading problem with less reliance on a priori smoothness constraint. A task assumption can behave like a non-uniform prior.

Amibiguity reduction using task constraints: "genericity"

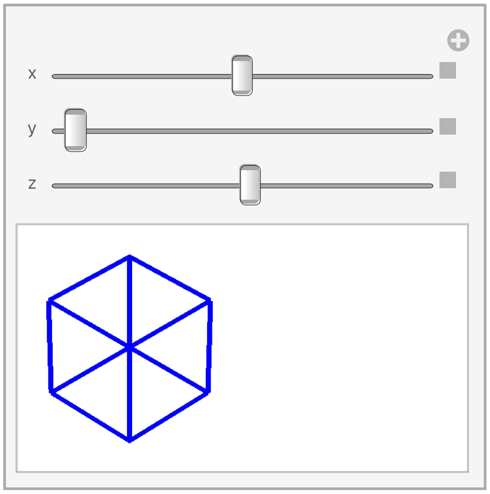
General viewpoint constraint (Lowe). Why is the figure on the left seen as a square rather than as a cube?

Cube: "Accidental" views



The default setting for viewpoint of a cube (below) illustrates another accidental view.

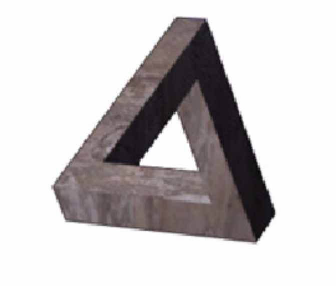
```
Manipulate[
 Graphics3D[{EdgeForm[{Thick, Blue}], FaceForm[{Pink, Opacity[0.0]}], Cuboid[]},
  Boxed → False, ViewPoint → {x, y, z}, ImageSize → Tiny],
 {{x, 10}, 0, 20}, {{y, 10}, 0, 1000}, {{z, 10}, -1, 20}]
```



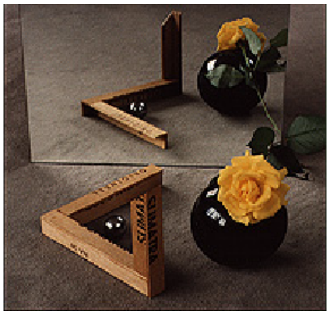
Play with the sliders.Which views seem to be “generic” and which “accidental”

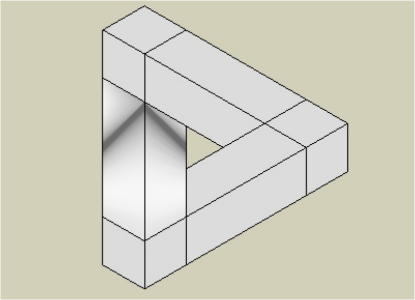
Penrose triangle

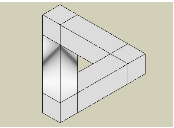
The generic view or "general viewpoint constraint" is so strong that the human visual system can sometimes prefer impossible to possible objects (e.g. Penrose triangle, http://en.wikipedia.org/wiki/Penrose_triangle).



Is there a real object that corresponds to this image? Below, the photograph by Bruno Ernst on the left shows a reflection of a penrose triangle. The right panel shows an image from an object on google's Sketchup database.







For a google sketchup model see: https://3dwarehouse.sketchup.com/model.html?id=5676b5d661012aadbe7144c5b7b0fcb6

Can you rotate the 3D object below to find the accidental view?

```
SetDirectory[dir];
Import["penrose.obj", ImageSize → Small]
```

```
Import::nffil : File not found during Import. ≫
```

```
$Failed
```

General principle?

In the previous lecture, we introduced the principle for shape from shading that said human vision should resolve ambiguity by choosing the interpretation of shape that would be most stable under perturbations of the illumination direction.
A general statement of this principle is:

Perception's model of the explicit variables in a scene should be robust to variations in the generic (secondary) variables
This principle follows from marginalization over illumination direction (Freeman, W. T. (1994). The generic viewpoint assumption in a framework for visual perception. *Nature, 368*, 542-545.)

For shape from shading the idea is to imagine wiggling the light source and ask: which interpretation gives the smallest image variation? Freeman showed that the circularly symmetric bump interpretation below gave the least variation, and in fact corresponds to the most probable interpretation.

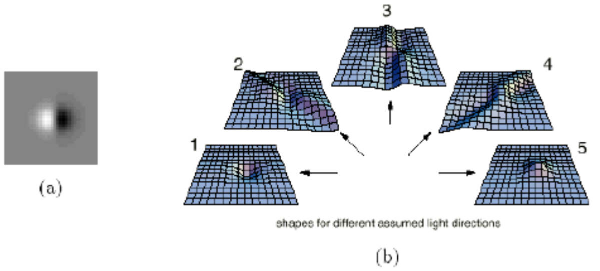


Figure 6: (a) Perceptually, this image has two possible interpretations. It could be a bump, lit from the left, or a dimple, lit from the right. (b) Mathematically, there are many possible interpretations. For a sufficiently shallow incident light angle, if we assume different light directions, we find different shapes, each of which could account for the observed image.

Figure from: http://www.merl.com/reports/docs/TR93-11a.pdf. See too: W. T. Freeman, Exploiting the generic viewpoint assumption, International Journal Computer Vision, 20 (3), 243-261, 1996.)

Lambertian vs. shiny ambiguity

Let's look at an another example from Freeman. This time it is applied to resolving ambiguity about material reflectance (the primary variable). The generic (secondary or nuisance) variable is surface orientation with respect to the viewer.

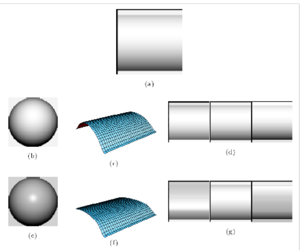
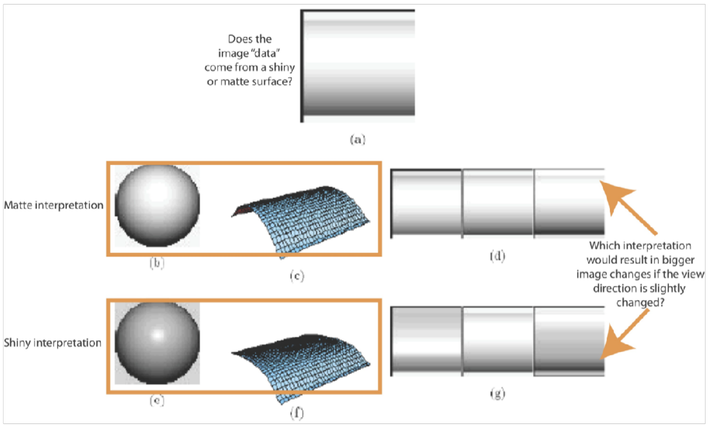


Figure 2: The image (a) appears to be a cylinder (c) painted with a Lambertian reflectance function (b) (shown on a hemisphere). However the flatter shape of (f) and a shiny reflectance function (e) also explain the data equally well. We can distinguish between the competing accounts for (a) by imagining rotating each shape. Images of each shape at three nearby orientations are shown in (d) and (g). We see that the image made assuming a Lambertian reflectance function (b) is more stable than that made assuming a shiny reflectance function (e). The reflectance function of (b) provides more angles over which the image looks nearly the same. If all viewpoints are equally likely, and the shapes and reflectances of (b-c) and (e-f) are equally likely to occur in the world, then (b-c) is a more probable interpretation than (e-f).


Does the image "data" come from a shiny or matte surface?
(a)
Matte interpretation
(b)
(c)
(d)
Which interpretation would result in bigger image changes if the view direction is slightly changed?
Shiny interpretation
(e)
(f)
(g)

Now imagine wiggling the orientation of the shape a little. The biggest variations are shown in (g). Smaller variations are seen in the (d) images. Thus the more probable scene interpretation is (b, c), rather than (e,f). (Figure from: Mitsubishi Tech Report, TR93-15. www.merl.com/papers/docs/TR93-15a.pdf.
See too: W. T. Freeman, Exploiting the generic viewpoint assumption, International Journal Computer Vision, 20 (3), 243-261, 1996.)

Freeman showed how integrating out the secondary variable sharpened the posterior for for several other problems (motion disambiguation).

See the Appendix, and Kersten (1999) for an example from depth from shadows, and for the connection between the robustness principle and prior constraints. For depth from shadows, the primary variable is relative depth and the secondary variable light source direction.

More on shape representations

Representations and mechanisms relevant to human vision?

Metric. vs. qualitative revisited

Are certain shape attributes treated as qualitatively different categories by the human visual system?
E.g. straight vs. curved contours? (e.g. Biederman, 1987; Knill & Kersten, 1991)
Depth discrimination on surfaces: Reichel, Todd, J. T., Yilmaz, E. (1995)
Convex vs. concave surfaces? (e.g. cue integration and outliers).
Shape recipes (see Freeman and Torralba, 2003). These are regression coefficients of a linear mapping from local bandpass image fragments to bandpass local shapes.

▶ 3. Use the generic viewpoint constraint to explain why a straight line in an image is not perceived as caused by a 3D curve

Building blocks, and deformable shape representations

One could represent shapes in terms of a basic vocabulary of building-blocks and relationships. This raises the question of how many types of blocks. One could go lego style with few blocks (“parts”) but then complicated objects are represented by many relationships. Or one could have many types and few spatial relationships. One approach is to have a few parts, but also use spatial deformations to allow for variations in shape, or in viewpoint. For example, see: Yuille et al., 1992.

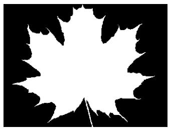
Human vision has the ability to analyze shapes across spatial scale, suggesting a hierarchical representation of shape. Later on we will look at affine deformations as an approximate way to allow for deformations in the image due to projection. But we also need to be able to allow for deformations within a class of objects (e.g. imagine trying to organize shape information to describe all the different kinds of shoes).

Basis-function contour shape representations

This example comes from Mathematica’s documentation.

Compute the Fourier descriptors of a shape:

```
In[1]:= i = 
```



```
;
```

Extract the contour coordinates and compute the Fourier transform of their complex representation:

```
In[2]:= c = ImageMeasurements[i, "Contours"][[1, 1]];
```

```
In[3]:= contourFourierDescriptors[contour_, n_] :=
         Module[{z},
          z = Fourier[Complex @@@ contour];
          z[[Ceiling[n / 2] ;; -Ceiling[n / 2]]] = 0;
          z
         ]
```

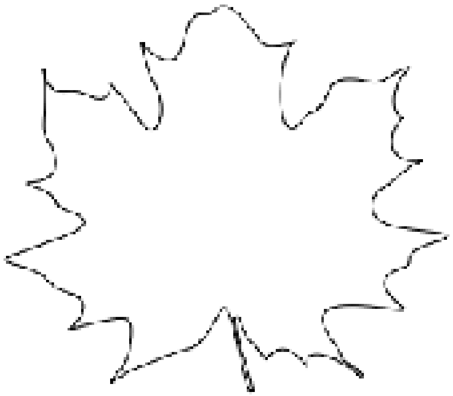
```
In[4]:= contourFourierDescriptors[c, 10] // Short

Out[4]//Short= {7147.36 + 5135.72 ⅈ, ≪1821≫, ≪19≫ + ≪18≫ ⅈ}
```

The original shape can be reconstructed using only a portion of the descriptors:

```
In[5]:= reconstructContour[descriptors_] := ReIm[InverseFourier[descriptors]]
```

```
In[6]:= Graphics[Line[reconstructContour[contourFourierDescriptors[c, 100]]]]
```

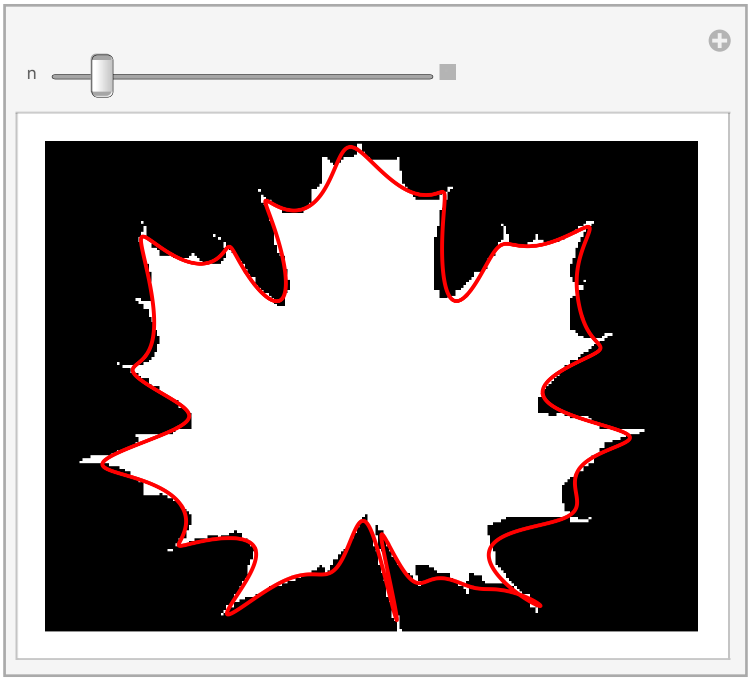
Out[6]=



Control the contour smoothness by interactively setting the number of descriptors:

```
In[7]:= Manipulate[
         HighlightImage[i, Line@reconstructContour[contourFourierDescriptors[c, n]]],
         {{n, 50}, 5, 500, 5},
         SaveDefinitions → True
        ]
```

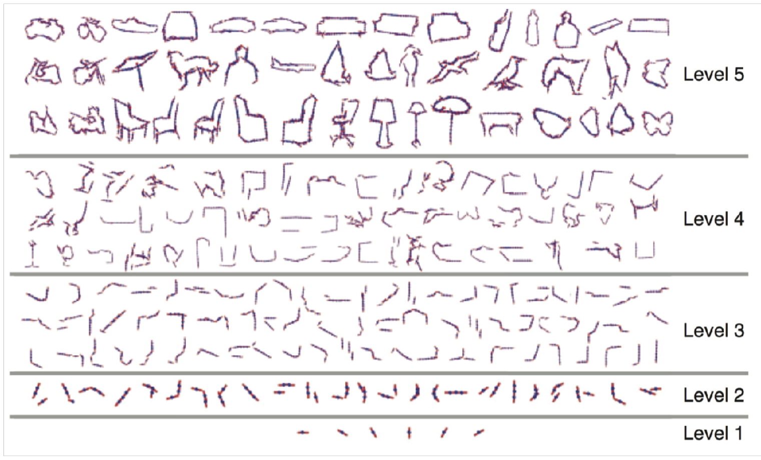
Out[7]=



Hierarchical representations of shape for object recognition

Machine learning approaches: 1) task-dependent (outcome-based); 2) unsupervised compositional learning

In task-dependent learning, features are learned that support the task requirements. We'll see later how image features (think of them as image "fragments") can be chosen to efficiently discriminate between different categories of objects. Here, shape is only implicit in the fragments learned. "Implicit" in this context means it isn't easy to access or "decode" shape from the representations.
In unsupervised compositional learning, the idea is to discover suspicious cooincidences and then use these to build up a hierarchy of features. In addition, the "compositional" requirement means that lower-level features or parts should be "resuable" or "shareable" at higher levels. The figure below shows examples of the mean shapes of visual concepts automatically learned for multiple objects with part sharing between objects (L. Zhu, Chen, Torralba, et al., 2011; L. Zhu, Chen, & Yuille, 2011). The specificity and the number of types of features increases as one goes up the hierarchy.



Compositional schemes are related to Biederman's geon representations which we will look at later.

Appendix

A linear solution to the inverse problem for lambertian shape from shading (Pentland)

Recall linear approximation from previous lecture (see also Appendix in that lecture)

Last time we derived a linear approximation to the lambertian shading model, L= **n.e.** The image luminance is given by

L(pn,qn) =**n.e**= $\frac{\{pn,qn,-1\}.\{pe,qe,-1\}}{\sqrt{(pn^2+qn^2+1)(pe^2+qe^2+1)}}$,

where n={pn,qn,-1}/$\sqrt{(pn^2+qn^2+1)}$ and e={pe,qe,-1}/$\sqrt{(pe^2+qe^2+1)}$ are the unit surface normal vector and illumination vectors respectively.

Using a Taylor series expansion about {pn,qn}={0,0}, we were able to derive a linear approximation to the shading equation.

```
In[763]:= Lmodel[pn_,qn_,pe_,qe_]:= ({pn,qn,-1}/Sqrt[pn^2+qn^2 + 1]).({pe,qe,-1}/Sqrt[pe^2+qe^2 + 1]
          Series[Lmodel[pn,qn,pe,qe], {pn, 0, 1}, {qn, 0, 1}]
```

Out[764]=
$$\left(\frac{1}{\sqrt{1+pe^2+qe^2}}+\frac{qe\,qn}{\sqrt{1+pe^2+qe^2}}+O[qn]^2\right)+\left(\frac{pe}{\sqrt{1+pe^2+qe^2}}+O[qn]^2\right)pn+O[pn]^2$$

```
In[765]:= Lapprxom[x_,y_] := (1+qe*qn[x,y] + pe*pn[x,y])/Sqrt[1+pe^2 + qe^2];
```

Using the fourier transform to estimate surface depth z, from image intensity L

A standard result from fourier transform theory is, given g(x) and its fourier transform $F_g(f_x)$, then it is easy to write down the fourier transform of the derivative of g(x). It is just the fourier transform of g(x)

times 2πifx. So, FourierTransform[$\frac{\text{dg}}{\text{dx}}$] = $2\pi i f_x\, F_g(f_x)$.(cf. Gaskill).
Let z(x,y) be any differentiable function. Let $F_z[f_x,f_y]$ be the fourier transform of z, which can be written in terms of the (complex) amplitude and phases as:
a[fx,fy] $e^{i\phi(\text{fx,fy})}$.
Using the above result, we have:

Fp[fx,fy] = 2π i fx a(fx,fy) $e^{i(\phi(\text{fx,fy}))}$ and Fq[fx,fy] = 2π i fy a(fx,fy) $e^{i(\phi(\text{fx,fy}))}$.

Let's simplify L = (1+qe*qn[x,y] + pe*pn[x,y])/Sqrt[1+pe^2 + qe^2] by ignoring the constant term 1.

Let's express the direction of the light source in terms of tilt and slant. Then $\tau=\tan^{-1}(q_e/p_e)$, and $\sigma = \tan^{-1}\left(\sqrt{{q_e}^2 + {p_e}^2}\right)$.

So we can write L = $\cos\sigma$ + p $\sin\sigma$ + q $\sin\tau \sin\sigma$. Let the fourier transform of the image function L be: a_L[fx,fy] $e^{i\phi_L(\text{fx,fy})}$. With a little algebra, one can show that:

$$\text{Fz(fx,fy)} = \frac{a_L(\text{fx,fy})\, e^{i(\phi_L(\text{fx,fy})-\pi/2)}}{2\,\pi \sin\sigma\, \text{fx}(\text{cosfy} \cos\tau + \text{sinfy} \sin\tau)}$$

The solution requires a known point light source (τ,σ), constant reflectance. Further the linear approximation is good for small p's and q's --i.e. for shallow bas-relief like shapes.
The inverse fourier transform provides an estimate of the surface depth z. Note that if we ignore the constant term in image luminance, we can replace z-> z + k1x + k2 y + k3, we obtain the same solution. Shortly, we will look at recent results that show how general shape ambiguity is resolved when the light source is unknown.

▶ 4. Write a program to implement and test the above shape from shading method

Mathematica demo of bas-relief ambiguity

First surface

The code below shows how to take one surface (the "First" surface), and transform it to another one through the "bas-relief" transform. The bas-relief transform simply adds a slanted and tilted plane to the first surface. The inverse of the transpose of bas-relief transform can be used to adjust the light source direction so that the resulting image is the same as for the image of the first surface.
We use the notation **a[ ]** for reflectance or albedo, and **s** for the light source direction vector.

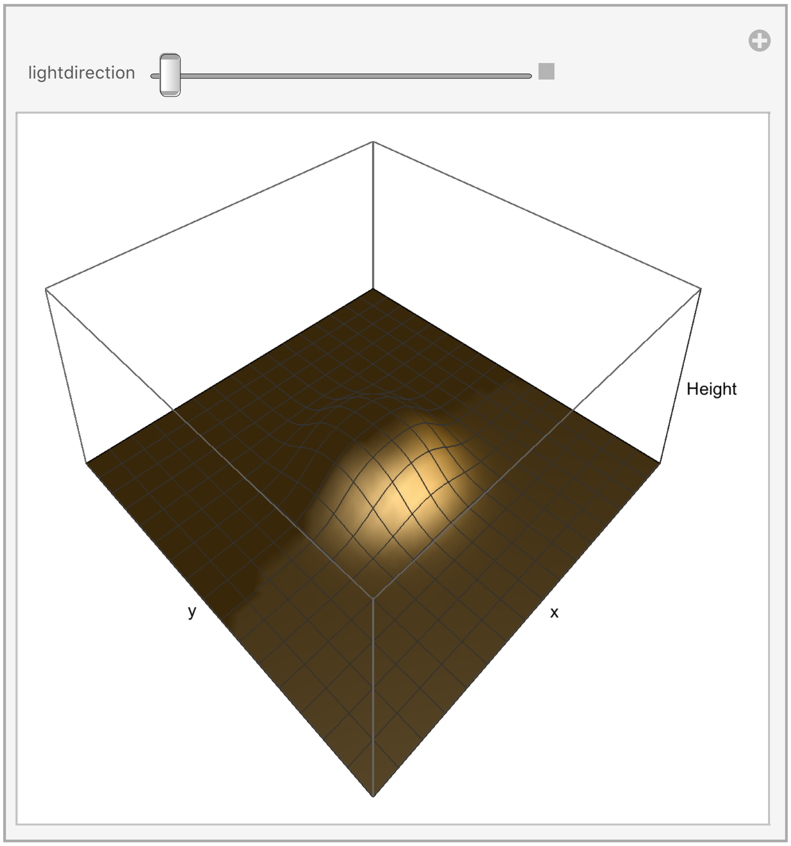
Define First surface bump

Use *Mathematica*'s Plot3D to render the bump.

```
In[766]:= bump[x_, y_] := (1 / 4) (1 - 1 / (1 + Exp[-5 (Sqrt[x^2 + y^2] - 1)]));
          p1[θ_] := RotationTransform[θ, {0, 0, 1}][{0, 20, 0}];
```

```
In[768]:= Manipulate[g1 = Plot3D[bump[x, y], {x, -3, 3},
              {y, -3, 3}, PlotPoints → 32, Mesh → True,  ViewPoint → {1, 1, 1},
              AxesLabel → {"x", "y", "Height"}, Ticks → False, AspectRatio → 1,
              PlotRange → {{-3, 3}, {-3, 3}, {0, 1}}, Lighting → {{"Ambient", GrayLevel[.25]},
                {"Point", GrayLevel[.75], p1[lightdirection]}}], {lightdirection, 0, 2 * Pi}]
```

Out[768]=



Now we'll render it using the lambertian shading equation.

Calculate surface normals of first surface bump

```
In[769]:= nx[x_, y_] := Evaluate[D[bump[x, y], y]];
          ny[x_, y_] := Evaluate[D[bump[x, y], x]];
```

Rendering specification for normals, light, reflectance first surface bump

Unit surface normals

Normalize the surface normal vector:

```
In[771]:= normbump[x_, y_] := {nx[x, y], ny[x, y], 1} / Sqrt[nx[x, y] ^ 2 + ny[x, y] ^ 2 + 1];
```

Point Light source

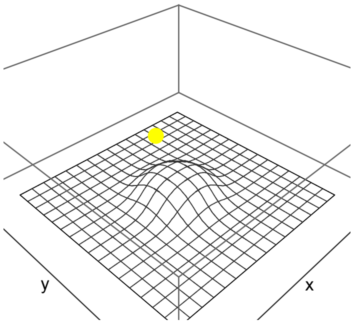
```
In[772]:= s = {1, 0, 1};
          s = s / Sqrt[s.s]
```

Out[773]= $\left\{\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}\right\}$

Plot point light source position

```
In[774]:= g1b = Plot3D[bump[x, y], {x, -3, 3}, {y, -3, 3}, PlotPoints → 32, Mesh → True,
              ViewPoint → {1, 1, 1}, AxesLabel → {"x", "y", "Height"}, Ticks → False,
              AspectRatio → 1, PlotRange → {{-4, 4}, {-4, 4}, {0, 1}}, PlotStyle → None];

In[775]:= g2 = ListPointPlot3D[{s}, ViewPoint → {1, 1, 1}, AxesLabel → {"x", "y", "Height"},
              Ticks → False, PlotStyle → {PointSize[.05], {RGBColor[1, 1, 0]}},
              AspectRatio → 1, PlotRange → {{-4, 4}, {-4, 4}, {0, 1}}, ImageSize → Small];

In[776]:= Show[g1b, g2]
```

Out[776]=



Reflectance

```
In[777]:= a[x_, y_] := 1;
```

Rendering equation for first surface bump

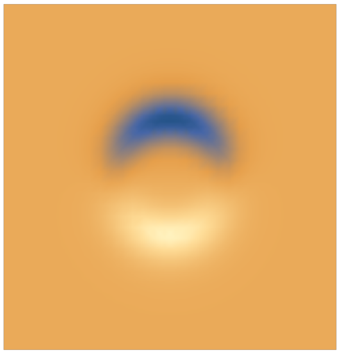
The lambertian model:

```
In[778]:= imagebump[x_, y_] := a[x, y] * normbump[x, y].s;
```

Render first surface bump

```
In[779]:= DensityPlot[imagebump[x, y], {x, -3, 3}, {y, -3, 3},
           Mesh → False, Frame → False, PlotPoints → 32, PlotRange → {0, 1}]
```

Out[779]=



A second surface

Now let's specify an arbitrary bas-relief transform. z(x; y) = λz(x; y)+ μx+ νy. You can play with the values of λ,μ,ν.

Bas relief transform, G : Second surface

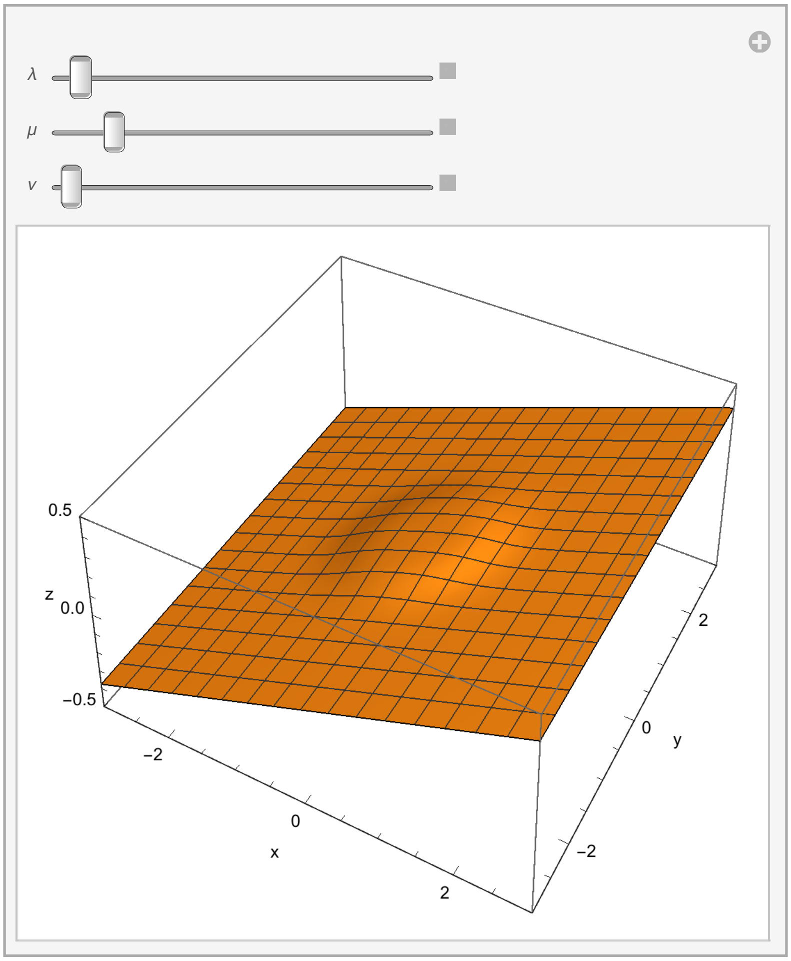
```
In[780]:= basreliefG[λ_, μ_, ν_] := {{λ, 0, -μ}, {0, λ, -ν}, {0, 0, 1}};

In[781]:=  λ0 = .25; μ0 = .125; ν0 = 0;
           λ = λ0; μ = μ0; ν = ν0;
          G = basreliefG[ λ, μ, ν];
          iGt = Inverse[Transpose[G]];
          brbump[x_, y_] := λ * bump[x, y] + μ * x + ν * y;

In[786]:= basreliefnormbump[x_, y_] := G.(a[x, y] * normbump[x, y]);
          s2 = iGt.s;
          a2[x_, y_] := Sqrt[basreliefnormbump[x, y].basreliefnormbump[x, y]];
          normbump2[x_, y_] :=
            basreliefnormbump[x, y] / Sqrt[basreliefnormbump[x, y].basreliefnormbump[x, y]];
```

Plot new bas-relief surface height

(*brpointbump[x_,y_]:=G.{x,y,bump[x,y]};*)

```
In[790]:= Manipulate[
  brbump2[x_, y_] := λ * bump[x, y] + μ * x + ν * y;
  Plot3D[brbump2[x, y], {x, -3, 3}, {y, -3, 3},
   AxesLabel → {"x", "y", "z"}, PlotPoints → 32, Mesh → True,
   AspectRatio → 1, PlotRange → {{-3, 3}, {-3, 3}, {-.5, .5}}],
  {{λ, λ0}, 0, 10}, {{μ, μ0}, 0, 1}, {{ν, ν0}, 0, 1}]
```

Out[790]=

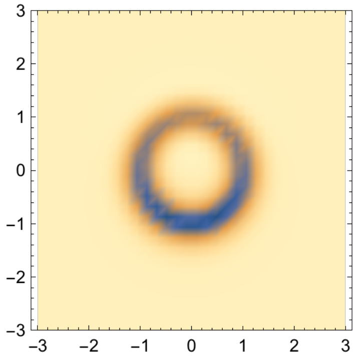


Plot reflectance

In[791]:=

```
DensityPlot[a2[x, y], {x, -3, 3}, {y, -3, 3},
 PlotPoints → 32, Mesh → False, PlotRange → {.9, 1.1}]
```

Out[791]=

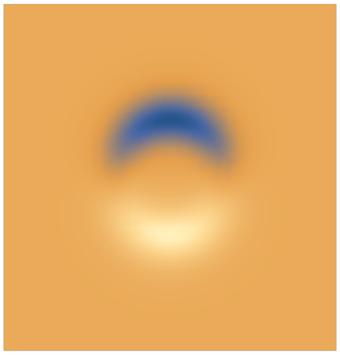


Render the second bump with the compensated light source and albedo

In[792]:=

```
imagebump2[x_, y_] := a2[x, y] * normbump2[x, y].s2;
DensityPlot[imagebump2[x, y], {x, -3, 3}, {y, -3, 3},
 Mesh → False, Frame → False, PlotPoints → 32, PlotRange → {0, 1}]
```

Out[793]=



What happens if you use the old albedo/reflectance map?

Compare first and second surfaces

Make plot of light source direction vectors for new and old

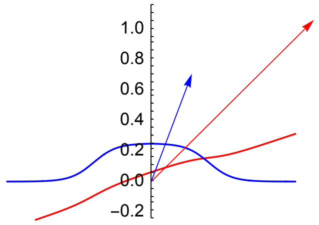
In[794]:=

```
units = Drop[s, {2}];(*units=units/Sqrt[units.units];*)
units2 = Drop[s2, {2}];(*units2=units2/Sqrt[units2.units2];*)
glight1 = ListVectorFieldPlot[{ {{0, 0}, units2}, {{0, 0}, units}},
   Ticks → False, Axes → False, PlotRange → {{-1.5, 2.5}, {-.25, 1}}];
```

Plot surface height cross-sections of new and old

```
In[797]:= gheight = Plot[{brbump[x, 0], bump[x, 0]},
   {x, -2.5, 2.5}, PlotRange → {{-3, 3}, {-.25, 1.15}}, PlotStyle →
    {{Thickness[.005], RGBColor[1, 0, 0]}, {Thickness[.005], RGBColor[0, 0, 1]}},
   Ticks → True, Axes → {False, True}, Epilog → {{RGBColor[1, 0, 0],
      Arrow[{{0, 0}, units2}]}, {RGBColor[0, 0, 1], Arrow[{{0, 0}, units}]}}]
```

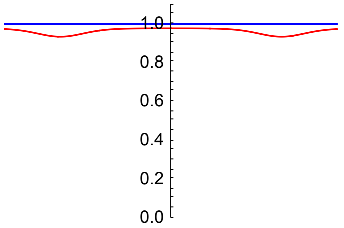
Out[797]=



Plot reflectance cross-sections for new and old

```
In[798]:= greflectance = Plot[{a2[x, 0] - 0.03, a[x, 0]}, {x, -1.5, 1.5},
   PlotRange → {0, 1.1}, PlotStyle → {{Thickness[.005], RGBColor[1, 0, 0]},
     {Thickness[.005], RGBColor[0, 0, 1]}}, Axes → {False, True}]
```

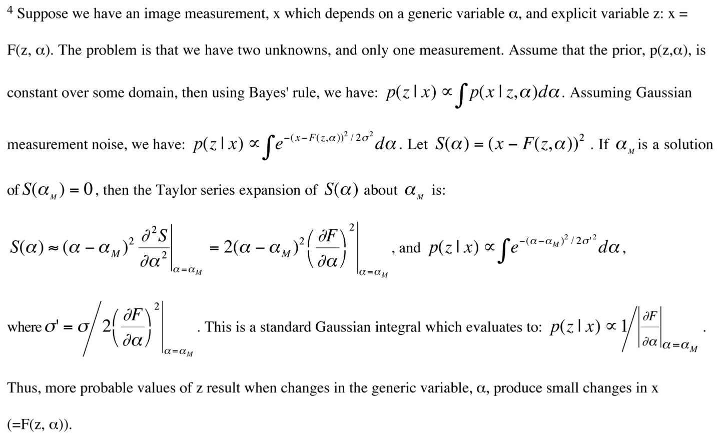
Out[798]=



Try transforming the bump, slight direction using other values of λ, μ, ν.

Applying the generic "view" principle to generic "light direction"

Excerpt from Kersten (1999) for the connection between the robustness principle and prior constraints.



[4] Suppose we have an image measurement, x which depends on a generic variable α, and explicit variable z: x = F(z, α). The problem is that we have two unknowns, and only one measurement. Assume that the prior, p(z,α), is constant over some domain, then using Bayes' rule, we have: $p(z \mid x) \propto \int p(x \mid z,\alpha) d\alpha$. Assuming Gaussian measurement noise, we have: $p(z \mid x) \propto \int e^{-(x-F(z,\alpha))^2/2\sigma^2} d\alpha$. Let $S(\alpha) = (x - F(z,\alpha))^2$. If α_M is a solution of $S(\alpha_M) = 0$, then the Taylor series expansion of $S(\alpha)$ about α_M is:

$$S(\alpha) \approx (\alpha - \alpha_M)^2 \left.\frac{\partial^2 S}{\partial \alpha^2}\right|_{\alpha=\alpha_M} = 2(\alpha - \alpha_M)^2 \left.\left(\frac{\partial F}{\partial \alpha}\right)^2\right|_{\alpha=\alpha_M}, \text{ and } p(z \mid x) \propto \int e^{-(\alpha-\alpha_M)^2/2\sigma'^2} d\alpha,$$

where $\sigma' = \sigma \Big/ 2\left.\left(\frac{\partial F}{\partial \alpha}\right)^2\right|_{\alpha=\alpha_M}$. This is a standard Gaussian integral which evaluates to: $p(z \mid x) \propto 1 \Big/ \left|\frac{\partial F}{\partial \alpha}\right|_{\alpha=\alpha_M}$.

Thus, more probable values of z result when changes in the generic variable, α, produce small changes in x (=F(z, α)).

References

Belhumeur, P. N., Kriegman, D. J., & Yuille, A. (1997). *The Bas-Relief Ambiguity.* Paper presented at the IEEE Conf. on Computer Vision and Pattern Recognition.

Belhumeur PN, Kriegman D, Yuille A (1999) The Bas-Relief Ambiguity. Int'l Journal of Computer Vision 35:33-44.

Biederman I (1987) Recognition-by-components: A theory of human image understanding. Psychological Review 94:115-147.

Brady, M., & Yuille, A. (1983). An Extremum Principle for Shape from Contour. *Proceedings IJCAI*, 969-972.

Bülthoff, H. H., & Yuille, A. (1990). Shape-from-X: Psychophysics and computation. *Proceedings of the SPIE, 1383*, 165-172.

Dupuis, P., & Oliensis, J. (1994). An optimal control formulation and related numerical methods for a problem in shape reconstruction. *The Annals of Applied Probability, 4*(No, 2), 287-346.

Freeman, W. T. (1994). The generic viewpoint assumption in a framework for visual perception. *Nature, 368*(7 April 1994), 542-545.

Freeman WT, Pasztor EC, Carmichael OT (2000) Learning low-level vision. Intl J Computer Vision 40:25-47.

Freeman, William T. and Torralba Antonio, (2003) Shape Recipes: Scene Representations that Refer to the Image in Advances in Neural Information Processing Systems 15 (NIPS), MIT Press, 2003 (ftp://publications.ai.mit.edu/ai-publications/2002/AIM-2002-019.pdf)

Gray, Alfred. Modern Differential Geometry of Curves and Surfaces with Mathematica, Second Edition. (CRC Press, 1997) (http://store.wolfram.com/view/ISBN0849371643/?39FED89E-38F0)

Horn, B., & Brooks, M. J. (1989). *Shape from shading*. Cambridge, Mass.: MIT Press.

Ikeuchi, K., & Horn, B. K. P. (1981). Numerical shape from shading and occluding boundaries. *Art. Intel., 17,*, 141-184.

Kersten, D., O'Toole, A. J., Sereno, M. E., Knill, D. C., & Anderson, J. A. (1987). Associative learning of scene parameters from images. *Appl. Opt., 26*, 4999-5006.

Kersten, D. (1999) High-level vision as statistical inference. The New Cognitive Neurosciences, 2nd Edition, Gazzaniga (Ed.). MIT Press. (pdf preprint, http://gandalf.psych.umn.edu/~kersten/kersten-lab/papers/CogNeuro98-final.pdf)

Knill, D. C., & Kersten, D. (1990). Learning a near-optimal estimator for surface shape from shading. CVGIP, 50(1), 75-100.

Knill DC, Kersten DK (1991) Ideal Perceptual Observers for Computation, Psychophysics, and Neural Networks. In: Pattern Recognition by Man and Machine (Watt RJ, ed), pp 83-97, chapter 87: MacMillan Press.

Knill, D. C. (1992). Perception of surface contours and surface shape: from computation to psychophysics. *J Opt Soc Am [A], 9*(9), 1449-1464.

Koenderink, J. J. (1998). Pictorial relief. *Philosophical Transactions of the Royal Society of London Series a-Mathematical Physical and Engineering Sciences, 356*(1740), 1071-1085.

Koenderink, J. J. (1990). *Solid Shape*. Cambridge, MA: MIT Press.

Koenderink, J. J., van Doorn, A. J., & Kappers, A. M. (1992). Surface perception in pictures. *Perception & Psychophysics, 5*, 487-496.

Koenderink, J.-J., Kappers, A., & van Doorn, A. (1992). Local Operations: The Embodiment of Geometry. *Artificial and Biological Vision Systems*.

Langer, M. S., & Zucker, S. W. (1994). Shape from Shading on a Cloudy Day. *Journal of the Optical Society of America A, 11*(2), 467-478.

Lehky, S. R., & Sejnowski, T. J. (1988). Network model of shape-from-shading: neural function arises from both receptive and projective fields. *Nature, 333*, 452-454.

Mamassian, P., Kersten, D., & Knill, D. C. (1996). Categorical local-shape perception. *Perception, 25*(1), 95-107.

Mamassian P & Kersten D. (1996). Illumination, shading and the perception of local orientation. *Vision Research., 36*(15), 2351-2367.

Mingolla, E., & Todd, J. T. (1986). Perception of Solid Shape from Shading. *Biological Cybernetics, 53*, 137-151.

Oliensis, J. (1991). Uniqueness in Shape From Shading. *The International Journal of Computer Vision, 6*(2), 75-104.

Pentland, A. P. (1990). Linear shape from shading. *International Journal of Computer Vision, 4*, 153-162.

Pentland, A. P. (1988). *Shape Information from Shading: A Theory About Human Perception. TR-103.* Paper presented at the M.I.T. Media Lab Vision Sciences.

Reichel, F. D., Todd, J. T., & Yilmaz, E. (1995). Visual-Discrimination of Local Surface Depth and Orientation. *Perception & Psychophysics, 57*(8), 1233-1240.

Todd, J. T., & Mingolla, E. (1983). Perception of Surface Curvature and Direction of Illumination from Patterns of Shading. *Journal of Experimental Psychology: Human Perception & Performance, 9*(4), 583-595.

Todd JT, Reichel FD (1989) Ordinal structure in the visual perception and cognition of smoothly curved surfaces. Psychol Rev 96:643-657.

Todd, J. T., Norman, J. F., Koenderink, J. J., & Kappers, A. M. L. (1997). Effects of texture, illumination, and surface reflectance on stereoscopic shape perception. *Perception, 26*(7), 807-822.

Yuille, A. L., Hallinan, P. W., & Cohen, D. S. (1992). Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, *8*(2), 99–111.

Zhu, L., Chen, Y., Lin, C., & Yuille, A. (2010, Au- gust). Max Margin Learning of Hierarchical Con- figural Deformable Templates (HCDTs) for Effi- cient Object Parsing and Pose Estimation. Inter- national Journal of Computer Vision, 93(1), 1– 21.

Zhu, L., Chen, Y., & Yuille, A. (2011, April). Re- cursive Compositional Models for Vision: De- scription and Review of Recent Work. Journal of Mathematical Imaging and Vision, 41(1-2), 122– 146.

Zhu, L., Chen, Y., Torralba, A., Freeman, W., & Yuille, A. (2011, January). Part and ap- pearance sharing: Recursive compositional mod- els for multi-view multi-object detection. IEEE Computer Society Conference on Computer Vi- sion and Pattern Recognition, 1919–1926.

kersten.org