

18. Color differential structure

Jan-Mark Geusebroek, Bart M. ter Haar Romeny, Jan J. Koenderink, Rein van den Boomgaard, Peter Van Osta

```
In[14]:= TextGrid[{
```



```
Style["From the book: \n ter Haar Romeny, B. M.:  
Front-end vision and multi-scale
```

```
image analysis: \n Multi-scale computer vision  
theory and applications, written in Mathematica.
```

```
ISBN: 978-1402015038
```

```
Publisher: "Hyperlink["Springer Dordrecht (2003)",
```

```
"https://link.springer.com/book/10.1007/978-1-4020-8840-7"],
```

```
FontSize -> 14, FontWeight -> Bold, FontFamily -> "Arial"]},
```

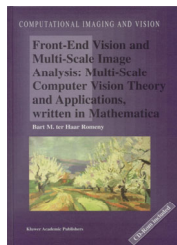
```
{Style["Download:", FontSize -> 14, FontWeight -> Bold],
```

```
Style[Hyperlink["www.romeny.info/FEV", "https://www.romeny.info/FEV"],
```

```
FontSize -> 14, FontWeight -> Bold,
```

```
FontFamily -> "Arial"]}], Alignment -> {Left, Top}]
```

```
Out[14]=
```



From the book:

ter Haar Romeny, B. M.:

Front-end vision and multi-scale image analysis:

Multi-scale computer vision theory

and applications, written in Mathematica.

ISBN: 978-1402015038

Publisher: Springer Dordrecht (2003)

Download: www.romeny.info/FEV

Initialization

18.1 Introduction

Color is an important extra dimension. Information extracted from color is useful for almost any computer vision task, like segmentation, surface characterization, etc. The

field of *color science* is huge [Wyszecki2000], and many theories exist. It is far beyond the scope of this book to cover even a fraction of the many different approaches. We will focus on a single recent theory, based on the color sensitive receptive fields in the front-end visual system. We are especially interested in the extraction of multi-scale differential structure in the spatial *and* the color domain of color images. This scale-space approach was recently introduced by Geusebroek et al. [Geusebroek1999a, Geusebroek2000a], based on the pioneering work of Koenderink's Gaussian derivative color model [Koenderink1998a]. This chapter presents the theory and a practical implementation of the extraction of color differential structure.

18.2 Color image formation and color invariants

What is color invariant structure? To understand that notion, we first have to study the process of color image formation.

```
In[28]:= p1 = Plot[Sin[0.06` λ] + Sin[0.1` λ] + 5, {λ, 350, 700}, PlotRange → {2, 8},
  ImageSize → 275, Ticks → {Automatic, None}, AspectRatio → 0.4` ]
p2 = DensityPlot[λ, {λ, 350, 700}, {y, 0, 2},
  AspectRatio → 0.1`, ColorFunction → (Hue[-0.001964` #1 + 1.375` ] &),
  ColorFunctionScaling → False, Frame → False, ImageSize → 275]
```

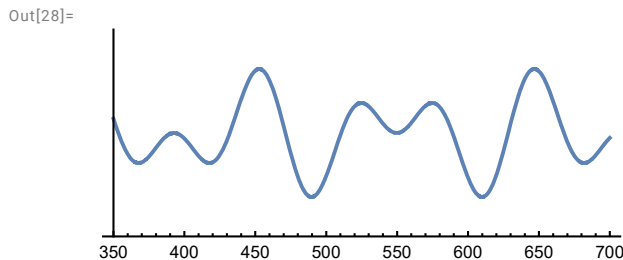


Figure 18.1 An arbitrary spectrum (distribution of the energy over the wavelengths) reflected from an object. For different colored objects we have different spectra. Horizontal scale: nm, vertical scale: energy (W/m^2).

Figure 18.1 shows an arbitrary spectrum that may fall onto the eye (or camera). The spectrum as reflected by an object is the result of light falling onto the object, of which part of the spectral energy is reflected towards the observer. Hence, the light spectrum falling onto the eye results from interaction between a light source, the object, and the observer. Color may be regarded as the measurement of spectral energy, and will be handled in the next section. Here, we only consider the interaction between light

source and material.

Before we see an object as having a particular color, the object needs to be illuminated. After all, in darkness objects are simply black. The emission spectra $l(\lambda)$ of common light sources are close to Planck's formula [Wyszecki 1999]

$$\ln[30]:= \mathbf{h} = 6.626176 \times 10^{-34}; \mathbf{c} = 2.99792458 \times 10^8; \mathbf{k} = 1.38066 \times 10^{-23};$$

$$\xi[\lambda_, T_] := 8 \pi \mathbf{h} \mathbf{c} \lambda^{-5} \left(\mathbf{E}^{\frac{\mathbf{h} \mathbf{c}}{\mathbf{k} T \lambda}} - 1 \right)^{-1}$$

where h is Planck's constant, k Boltzmann's constant, and c the velocity of light in vacuum. The *color temperature* of the emitted light is given by T , and typically ranges from 2,500K (warm red light) to 10,000K (cold blue light). Note that the terms "warm" and "cold" are given by artists, and refer to the sensation caused by the light. Representative white light is, by convention, chosen to be at a temperature of 6500K. However, in practice, all light sources between 2,500K and 10,000K can be found. Planck's equation is adequate for incandescent light and halogen. The spectrum of daylight is slightly different, and is represented by a *correlated color temperature*. Daylight is close enough to the Planckian spectrum to be characterized by a equivalent parameter.

The part of the spectrum reflected by a surface depends on the surface *spectral reflection function*. The spectral reflectance is a material property, characterized by a function $c(\lambda)$. For planar, matte surfaces, the spectrum reflected by the material $e(\lambda)$ is simply the multiplication between the spectrum falling onto the surface $l(\lambda)$ and the surface spectral reflectance function $c(\lambda)$: $e(\lambda) = c(\lambda) l(\lambda)$. For example, figure 18.2 shows the emission spectrum of three light sources, resp. of 2500K, 4500K and 10000K.

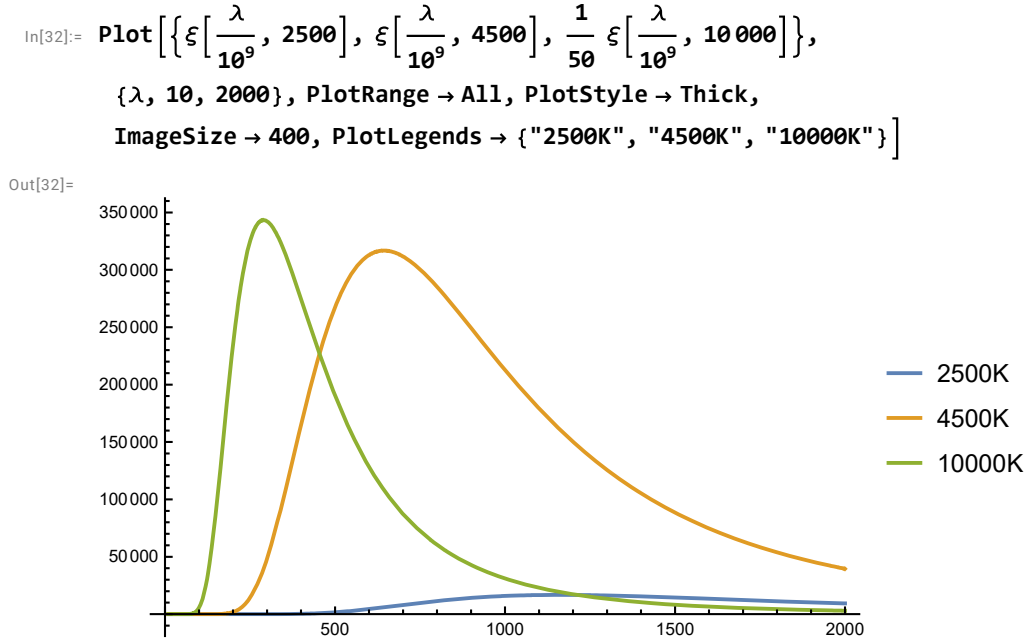


Fig 18.2 Emission spectrum of a black body light source with a temperature of resp. 2500K, 6500K and 10,000K. The emission at 10,000K is much larger than at the 2 lower temperatures, and for that reason, is plotted at 1/50 of its amplitude.

Now that we have the spectral reflectance function, we can examine how the reflected spectrum would look with a different light source. In figure 18.3 the reflected spectrum for a 2500K, 4500K and a 10,000K radiator is demonstrated.

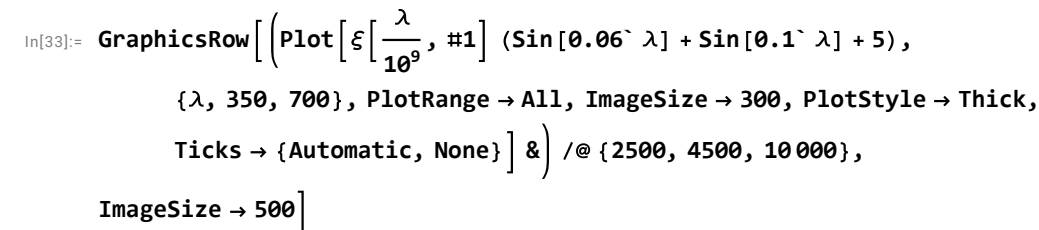


Fig 18.3 Object reflectance function for the observed spectrum shown in figure 18.1 for a resp. 2500K, 6500K and 10,000K light source. As opposed to the reflected spectrum (fig. 18.1), this object reflectance function is a material property, independent of the illumination source.

At this point it is meaningful to introduce spatial extent, hence to describe the spatio-spectral energy distribution $e(x,y,\lambda)$ that falls onto the retina. Further, for three-dimen-

sional objects the amount of light falling onto the objects surface depends on the energy flux, thus on the local geometry. Hence *shading* (and shadow) may be introduced as being a wavelength independent multiplication factor $m(x,y)$ in the range $[0...1]$: $e(x, y, \lambda) = c(x, y, \lambda) l(\lambda) m(x, y)$.

Note that the illumination $l(\lambda)$ is independent of position. Hence the equation describes spectral image formation of matte objects, illuminated by a single light source. For shiny surfaces the image formation equation has to be extended with an additive term describing the Fresnel reflected light, see [Geusebroek2000b] for more details.

The *structure* of the spatio-spectral energy distribution is due to the three functions $c(\cdot)$, $l(\cdot)$, and $m(\cdot)$. By making some general assumptions, these quantities may be derived from the measured image. Estimation of the object reflectance function $c(\cdot)$ boils down to deriving material properties, the "true" color invariant which does not depend on illumination conditions. Estimation of the light source $l(\cdot)$ is well known as the color constancy problem. Determining $m(\cdot)$ is in fact estimating the shadows and shading in the image, and is closely related to the shape-from-shading problem.

For the extraction of color invariant properties from the spatio-spectral energy distribution we search for algebraic or differential expressions of $e(\cdot)$, which are independent of $l(\cdot)$ and $m(\cdot)$. Hence the goal is to solve for differential expressions of $e(\cdot)$ which results in a function of $c(\cdot)$ only.

To proceed, note that the geometrical term m is only a function of spatial position. Differentiation with respect to λ , and normalization reduces the problem to only two functions: $e(x, \lambda) = c(x, \lambda) l(\lambda) \Rightarrow \frac{1}{e(x,\lambda)} \frac{\partial e(x,\lambda)}{\partial \lambda} = \frac{l_\lambda}{l} + \frac{c_\lambda}{c}$ (indices indicate differentiation). After additional differentiation to the spatial variable x or y , the first term vanishes, since $l(\cdot)$ only depends on λ :

```
In[34]:= Clear [c, ξ, e];
```

```
In[35]:= e [x, λ] = c [x, λ] × l [λ];
```

```
∂x (  $\frac{\partial_\lambda e [x, \lambda]}{e [x, \lambda]}$  ) // Simplify
```

```
Out[36]= 
$$\frac{-c^{(0,1)} [x, \lambda] c^{(1,0)} [x, \lambda] + c [x, \lambda] c^{(1,1)} [x, \lambda]}{c [x, \lambda]^2}$$

```

With the function `shortnotation[]` we can write it shorter:

```
In[37]:= e[x, λ] = c[x, λ] × l[λ];
          ∂x (  $\frac{\partial_\lambda e[x, \lambda]}{e[x, \lambda]}$  ) // Simplify // shortnotation
```

```
Out[38]=  $\frac{c[x, \lambda] c_{x\lambda} - c_x c_\lambda}{c[x, \lambda]^2}$ 
```

The left-hand side, after applying the chain rule,

```
In[39]:= ∂x (  $\frac{\partial_\lambda e[x, y, \lambda]}{e[x, y, \lambda]}$  ) // Simplify // shortnotation
```

```
Out[39]=  $\frac{e[x, y, \lambda] e_{x\lambda} - e_x e_\lambda}{e[x, y, \lambda]^2}$ 
```

is completely expressed in spatial and spectral derivatives of the observable spatio-spectral energy distribution.

As an example in this chapter we develop the differential properties of the invariant color-edge detector $\mathcal{E} = \frac{1}{e} \frac{\partial e}{\partial \lambda}$, where the measured spectral intensity $e = e(x, y, \lambda)$. Spatial derivatives of \mathcal{E} , like $\frac{\partial \mathcal{E}}{\partial x}$, contain derivatives to the spatial as well as to the wavelength dimension due to the chain rule. In the next section we will see that the zero-th, first and second order derivative-to- λ kernels are acquired from the transformed RGB space of the image directly. The derivatives to the spatial coordinates are acquired in the conventional way, i.e. convolution with a spatial Gaussian kernel.

18.3 Koenderink's Gaussian derivative color model

We have seen in the previous chapters how spatial structure can be extracted from the data in the environment by measuring the set of (scaled) derivatives to some order. For the spatial domain this has led to the family of Gaussian derivative kernels, sampling the spatial intensity distribution. These derivatives naturally occur in a local Taylor expansion of the signal.

Koenderink proposed in 1986 to take a similar approach to the sampling of the color dimension, i.e. the spectral information contained in the color. If we construct the Taylor expansion of the spatio-spectral energy distribution $e(x, y, \lambda)$ of the measured light to wavelength, in the fixed spatial point (x_0, y_0) , and around a central wavelength λ_0 we get (to second order):

In[40]:= **Series**[**e**[**x0**, **y0**, **λ**], {**λ**, **λ0**, **2**}

Out[40]=

$$e[x_0, y_0, \lambda_0] + e^{(\theta, \theta, 1)} [x_0, y_0, \lambda_0] (\lambda - \lambda_0) + \frac{1}{2} e^{(\theta, \theta, 2)} [x_0, y_0, \lambda_0] (\lambda - \lambda_0)^2 + 0[\lambda - \lambda_0]^3$$

We recall from chapter 1 that a physical measurement with a aperture is mathematically described with a convolution. So for a measurement of the luminance L with aperture function $G(x, \sigma)$ in the (here in the example 1D) spatial domain we get:

$L(x; \sigma) = \int_{-\infty}^{\infty} L(x - \alpha) G(\alpha, \sigma) d\alpha$ where α is the dummy spatial shift parameter running over all possible values.

For the temporal domain we get $L(t; \sigma) = \int_{-\infty}^{\infty} L(t - \beta) G(\beta, \sigma) d\beta$ where β is the dummy temporal shift parameter running over all possible values in time (in chapter 20 we will take a close look at this temporal convolution). Based on this analogy, we might expect a measurement along the color dimension to look like:

$L(\lambda; \sigma) = \int_{-\infty}^{\infty} L(\lambda - \gamma) G(\gamma, \sigma) d\gamma$ where λ is the wavelength and γ is the dummy wavelength shift parameter.

The front-end visual system has implemented the shifted spatial kernels with a grid on the retina with receptive fields, so the shifting is implemented by the simultaneous measurement of all the neighboring receptive fields. The temporal kernels are implemented as time-varying LGN and cortical receptive fields (explained in detail in chapters 11 and 20). However, in order to have a wide range of receptive fields which shift over the wavelength axis in sensitivity, would require a lot of different photo-sensitive dyes (rhodopsins) in the receptors with these different -shifted- color sensitivities.

The visual system may have opted for a cheaper solution: The convolution is calculated at just a single position on the wavelength axis, at around $\lambda_0 = 520$ nm, with a standard deviation of the Gaussian kernel of about $\sigma_\lambda = 55$ nm. The integration is done over the range of wavelengths that is covered by the rhodopsins, i.e. from about 350 nm (blue) to 700 nm (red). The values for λ_0 and σ_λ are determined from the best fit of a Gaussian to the spectral sensitivity as measured psycho-physically in humans, i.e. the Hering model.

So we get for the spectral intensity $e(\vec{x}, \lambda_0; \sigma_\lambda) = \int_{\lambda_{\min}}^{\lambda_{\max}} e(\vec{x}, \lambda) G(\lambda, \lambda_0, \sigma_\lambda) d\lambda$. This is a 'static' convolution operation. It is not a convolution in the familiar sense, because we don't shift over the whole wavelength axis. We just do a *single* measurement with a Gaussian aperture over the wavelength axis at the position λ_0 . Similarly, the derivatives to λ

$$\frac{\partial e(\vec{x}, \lambda_0)}{\partial \lambda} = \sigma_\lambda \int_{\lambda_{\min}}^{\lambda_{\max}} e(\vec{x}, \lambda) \frac{\partial G(\lambda, \lambda_0, \sigma_\lambda)}{\partial \lambda} d\lambda \text{ and}$$

$$\frac{\partial^2 e(\vec{x}, \lambda_0)}{\partial \lambda^2} = \sigma_\lambda^2 \int_{\lambda_{\min}}^{\lambda_{\max}} e(\vec{x}, \lambda) \frac{\partial^2 G(\lambda, \lambda_0, \sigma_\lambda)}{\partial \lambda^2} d\lambda$$

describe the first and second order spectral derivative respectively. The factors σ_λ and σ_λ^2 are included for the normalization, i.e. to make the Gaussian spectral kernels dimensionless.

Here are the graphs of the 'static' normalized Gaussian spectral kernels to second order as a function of wavelength:

```
In[41]:= gauss $\lambda$ [ $\lambda$ _,  $\sigma$ _] =  $\partial_\lambda$  gauss[ $\lambda$ ,  $\sigma$ ];
gauss $\lambda\lambda$ [ $\lambda$ _,  $\sigma$ _] =  $\partial_{\{\lambda,2\}}$  gauss[ $\lambda$ ,  $\sigma$ ];
 $\lambda\theta$  = 520;  $\sigma\lambda$  = 55;
max = gauss[ $\theta$ ,  $\sigma\lambda$ ];
Plot[{gauss[ $\lambda$  -  $\lambda\theta$ ,  $\sigma\lambda$ ] / max, - $\sigma\lambda$  gauss $\lambda$ [ $\lambda$  -  $\lambda\theta$ ,  $\sigma\lambda$ ] / max,
   $\sigma\lambda^2$  gauss $\lambda\lambda$ [ $\lambda$  -  $\lambda\theta$ ,  $\sigma\lambda$ ] / max}, { $\lambda$ ,  $\lambda\theta$  - 4  $\sigma\lambda$ ,  $\lambda\theta$  + 4  $\sigma\lambda$ }, PlotRange  $\rightarrow$  All,
  AxesLabel  $\rightarrow$  {" $\lambda$  (nm)", ""}, PlotStyle  $\rightarrow$  {Directive[Green, Thick],
  Directive[Blue, Thick], Directive[{Red, Thick}]}, ImageSize  $\rightarrow$  400]
```

Out[45]=

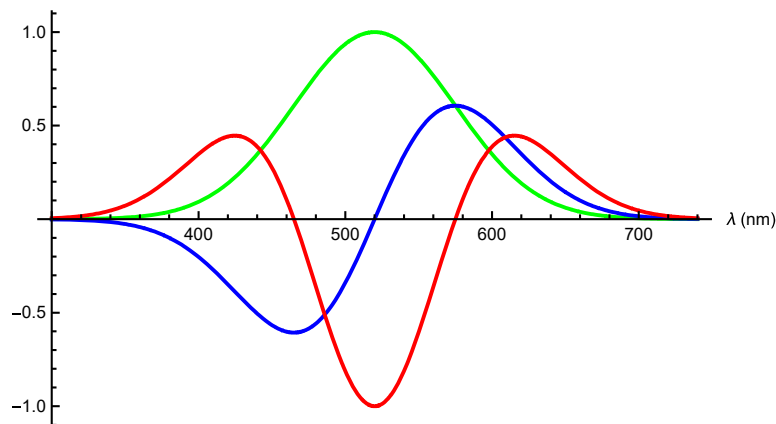


Figure 18.4 The zero-th, first and second derivative of the Gaussian function with respect to wavelength as models for the color receptive field's wavelengths sensitivity in human color vision. After [Koenderink 1986]. The central wavelength is 520 nm, the standard deviation 55 nm.

We recall from the human vision chapter that the color sensitive receptive fields come in the combinations red-green and yellow-blue center-surround receptive fields. The subtraction of yellow and blue in these receptive fields is well modeled by the first order derivative to λ , the subtraction of red and green minus the blue is well modeled by the second order derivative to λ . Alternatively, one can say that the zero-th order receptive field measures the luminance, the first order the 'blue-yellowness', and the second order the 'red-greenness'. The sensitivity of the three types of cones (L-, M- and S-cones: long, medium and short wavelength sensitivity) is given in figure 18.5.

```
In[46]:= Import[url <> "ConesColors.gif", ImageSize -> 300]
```

```
Out[46]=
```

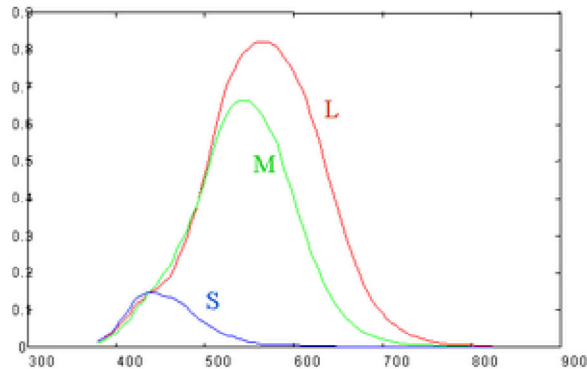


Figure 18.5 The relative spectral sensitivity of the three types of cones in the human retina. From [Geusebroek et al. 1999a].

The Gaussian model approximates the Hering basis [Hering64a] for human color vision when $\lambda_0 \approx 520$ nm and $\sigma_\lambda \approx 55$ nm (see figure 18.6). They also fit very well the CIE 1964 XYZ basis, which is a famous coordinate system for colors, much used in technical applications involving color.

Note: the wavelength axis is a half axis. It is known that for a *half axis* (such as with positive-only values) a logarithmic parametrization is the natural way to 'step along' the axis. E.g. the scale axis is logarithmically sampled in scale-space (remember the 'orders of magnitudes'), the intensity is logarithmically transformed in the photoreceptors, and, as we will see in chapter 20, the time axis can only be measured causally when we sample it logarithmically. We might conjecture here a better fit to the Hering model with a logarithmic wavelength axis.

```
In[47]:= Import[url <> "HeringColormodel.gif", ImageSize -> 500]
```

```
Out[47]=
```

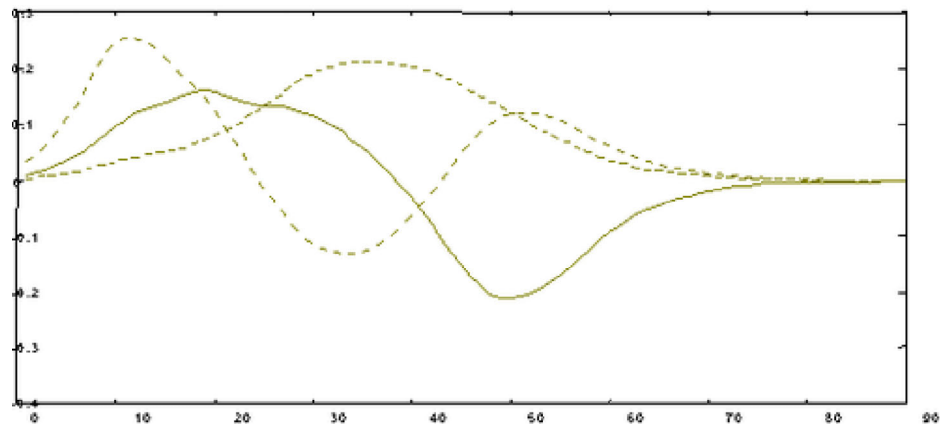


Figure 18.6 The Hering basis for the spectral sensitivity of human color receptive fields. From [Hering1964a].

The Gaussian color model needs the first three components of the Taylor expansion of the Gaussian weighted spectral energy distribution at λ_0 and scale σ_λ . An RGB camera measures the red, green and blue component of the incoming light, but this is *not* what we need for the Gaussian color model. We need a method to extract the Taylor expansion terms from the RGB values. The figure below shows the RGB color space. The black color is diagonally opposite the white cube.

```
In[48]:= Graphics3D[Table[
  {RGBColor[{i, j, k} / 8], Sphere[{i, j, k}, 1 / 3]}, {i, 8}, {j, 8}, {k, 8}]
Out[48]=
```

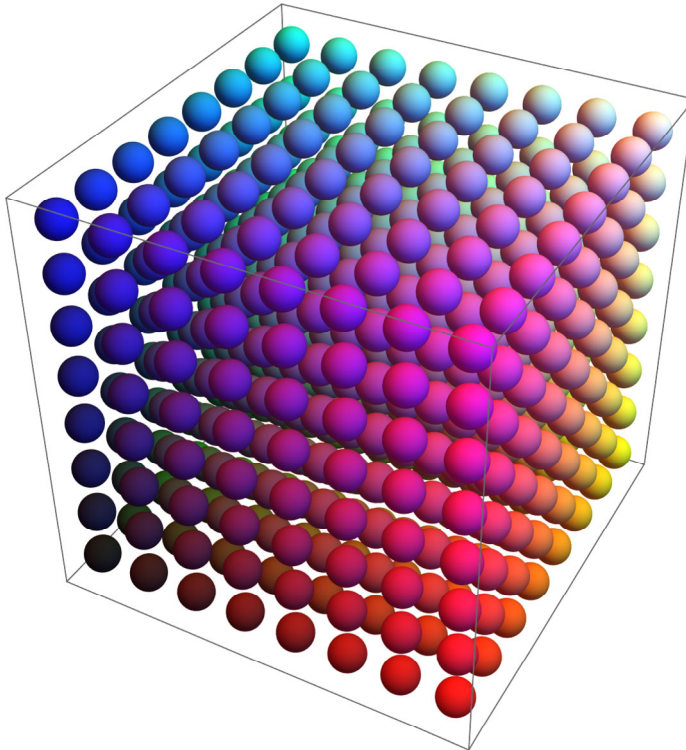


Figure 18.7 RGB color space.

This plots every 10th pixel of a color image as points (with their RGB color) in RGB space:

```
In[49]:= im = ImageData[orig = Import[url <> "hibiscus2.jpg"]];
```

```

In[50]:= Dimensions[im]
Out[50]=
{178, 280, 3}

In[51]:= data = Flatten[im, 1];
GraphicsRow[
  {Show[orig], Graphics3D[({RGBColor@@#1, PointSize[0.008], Point[#1]} &) /@
    Part[data, 1 ;; All ;; 10], Axes → True,
    AxesLabel → {"R", "G", "B"}]}, ImageSize → 500]
Out[52]=

```

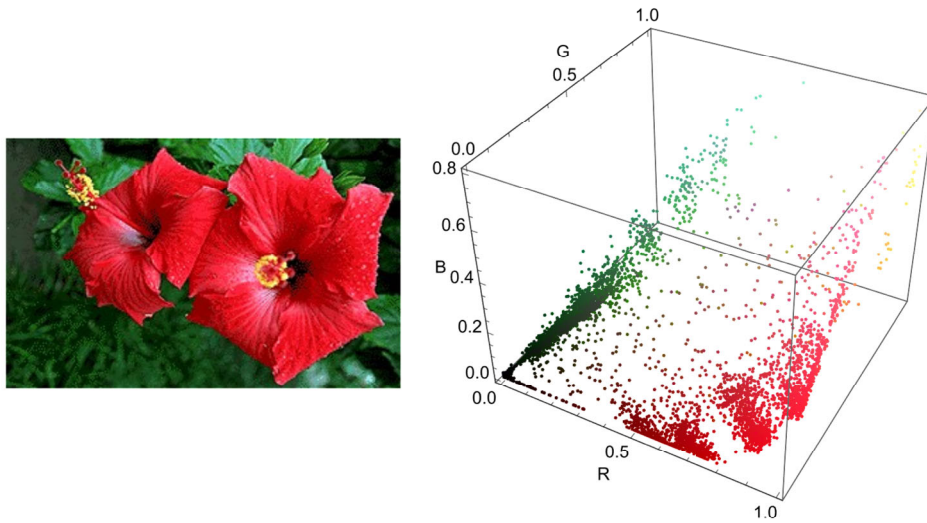


Figure 18.8 Left: color input image. Right: distribution of the pixels in the RGB space. Note the wide range of green colors (in the left cluster) and red colors (in the right cluster) due to the many shadows and lighting conditions.

An RGB camera approximates the CIE 1964 XYZ basis for colorimetry by the following linear transformation matrix:

$$\text{In[53]:= } \mathbf{rgb2xyz} = \begin{pmatrix} 0.621 & 0.113 & 0.194 \\ 0.297 & 0.563 & 0.049 \\ -0.009 & 0.027 & 1.105 \end{pmatrix};$$

Geusebroek et al. [Geusebroek2000a] give the best linear transform from the XYZ values to the Gaussian color model:

$$\text{In[54]:= } \mathbf{xyz2e} = \begin{pmatrix} -0.019 & 0.048 & 0.011 \\ 0.019 & 0. & -0.016 \\ 0.047 & -0.052 & 0. \end{pmatrix};$$

The resulting transform from the measured RGB input image to the sampling 'à la human vision' is the dot product of the transforms:

```
In[55]:= colorRF = xyz2e.rgb2xyz; colorRF // MatrixForm
```

```
Out[55]//MatrixForm=

$$\begin{pmatrix} 0.002358 & 0.025174 & 0.010821 \\ 0.011943 & 0.001715 & -0.013994 \\ 0.013743 & -0.023965 & 0.00657 \end{pmatrix}$$

```

Indeed, when we study this transform and plot the rows, we see the likeness with the Gaussian derivative sensitivity (see figure 18.6).

```
In[56]:= GraphicsRow[Table[
  ListPlot[colorRF[[i]], Joined → False, Filling → Axis, FillingStyle → Thick,
  PlotStyle → PointSize[0.05`], Ticks → None], {i, 3}], ImageSize → 400]
```

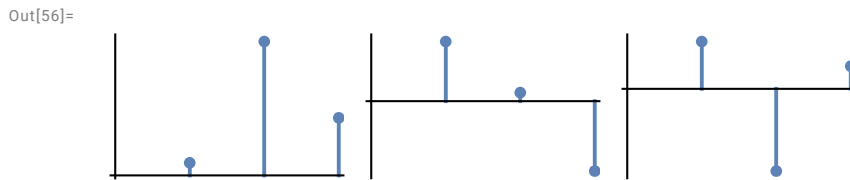


Figure 18.9 The transformations of the input RGB values to the Gaussian color model coarsely resemble the Gaussian derivatives to λ . Left: zero-th order. Middle: first order, right: second order. The three center wavelengths roughly correspond with 400, 475 and 575 nm of the Gaussian derivative functions of figure 18.4.

The set of spatio-spectral Gaussian derivative cortical simple cells thus looks like (from [Geusebroek et al. 2000a]):

```
In[57]:= Import[ur1 <> "ColorRFs.gif", ImageSize → 400]
```

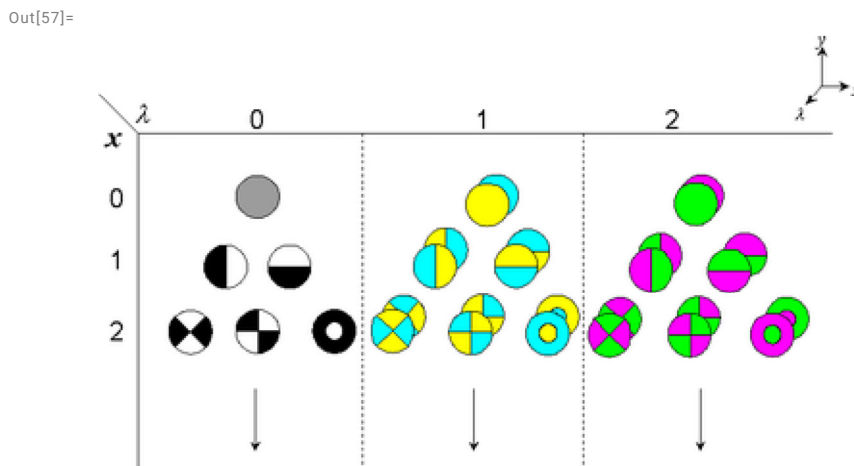


Figure 18.10 The Gaussian color model for cortical receptive fields. Left: zero-th order to wavelength, measuring the luminance and the spatial derivative structure. Middle: first order derivative to wavelength, yellow/blue - spatial derivatives. Right: second order derivative to wave-

length, red/green-spatial derivatives.

The Gaussian color model is an approximation, but has the attractive property of fitting very well into Gaussian scale-space theory. The notion of image structure is extended to the wavelength domain in a very natural and coherent way. The similarity with human differential-color receptive fields is more than a coincidence.

Now we have all the tools to come to an actual implementation. The RGB values of the input image are transformed into Gaussian color model space, and plugged into the spatio-spectral formula for the color invariant feature.

Next to the derivatives to wavelength we need spatial derivatives, which are computed in the regular way with spatial Gaussian derivative operators. The full machinery of e.g. gauge coordinates and invariance under specific groups of transformations is also applicable here. The next section details the implementation.

18.4 Implementation

We start with the import of an RGB color image (see figure 11). The color pixels are RGB triples in the very first element of the imported object:

```
In[58]:= image = Import [url <> "colortoys2.jpg"]
```

```
Out[58]=
```



```
In[59]:= im = ImageData [image] ;
```

```
Dimensions [im]
```

```
Out[60]=
```

```
{ 228, 179, 3 }
```

The RGB triples are converted into measurements through the color receptive fields in

the retina with the transformation matrix `colorRF` defined above:

```
In[61]:= colorRF = xyz2e.rgb2xyz; colorRF // MatrixForm
Out[61]//MatrixForm=

$$\begin{pmatrix} 0.002358 & 0.025174 & 0.010821 \\ 0.011943 & 0.001715 & -0.013994 \\ 0.013743 & -0.023965 & 0.00657 \end{pmatrix}$$

```

To transform every RGB triple we map the transformation to our input image as a pure function at the second listlevel:

```
In[62]:= observedimage = Map[Dot[colorRF, #] &, im, {2}];
```

The three 'layers' of this observed image `obs` represent resp. e , e_λ and $e_{\lambda\lambda}$. We 'slice' the dataset smartly by a reordering `Transpose` whereby the e -, e_λ - and $e_{\lambda\lambda}$ -values each form a plane:

```
In[63]:= obs = Transpose[observedimage, {2, 3, 1}];
Dimensions[obs]
Out[64]=
{3, 228, 179}
```

Let us inspect what the color receptive fields see:

```
In[65]:= GraphicsRow[Prepend[ArrayPlot[/@obs, Show[image]], ImageSize -> 470]
Out[65]=
```

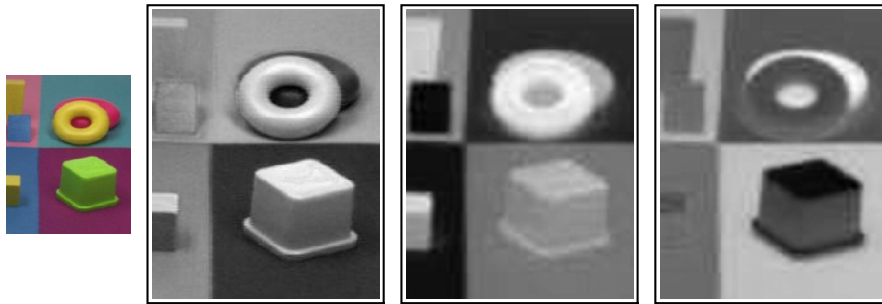


Figure 11 The input image (left) and the observed images with the color differential receptive fields. Image resolution 228x179 pixels.

We now develop the differential properties of the invariant color-edge detector

$\mathcal{E} = \frac{1}{e} \frac{\partial e}{\partial \lambda}$, where the spectral intensity $e = e(x, y, \lambda)$. The derivatives to the spatial and spectral coordinates are easily found with the chain rule. Here are the explicit forms:

$$\begin{aligned} \text{In[66]:=} & \frac{\partial_{\lambda} e[\mathbf{x}, \mathbf{y}, \lambda]}{e[\mathbf{x}, \mathbf{y}, \lambda]} \\ \text{Out[66]=} & \frac{e^{(0,0,1)}[\mathbf{x}, \mathbf{y}, \lambda]}{e[\mathbf{x}, \mathbf{y}, \lambda]} \end{aligned}$$

So:

$$\begin{aligned} \text{In[67]:=} & \varepsilon := \frac{\partial_{\lambda} e[\mathbf{x}, \mathbf{y}, \lambda]}{e[\mathbf{x}, \mathbf{y}, \lambda]}; \\ \text{Out[68]=} & \frac{\partial_{\mathbf{x}} \varepsilon}{e[\mathbf{x}, \mathbf{y}, \lambda]^2} \\ & = \frac{e^{(0,0,1)}[\mathbf{x}, \mathbf{y}, \lambda] e^{(1,0,0)}[\mathbf{x}, \mathbf{y}, \lambda]}{e[\mathbf{x}, \mathbf{y}, \lambda]^2} + \frac{e^{(1,0,1)}[\mathbf{x}, \mathbf{y}, \lambda]}{e[\mathbf{x}, \mathbf{y}, \lambda]} \end{aligned}$$

To make this more readable, we apply *pattern matching* to make the expression shorter (with **shortnotation**).

$$\begin{aligned} \text{In[69]:=} & \partial_{\mathbf{y}} \varepsilon // \text{shortnotation} \\ \text{Out[69]=} & \frac{e[\mathbf{x}, \mathbf{y}, \lambda] e_{\mathbf{y}\lambda} - e_{\mathbf{y}} e_{\lambda}}{e[\mathbf{x}, \mathbf{y}, \lambda]^2} \end{aligned}$$

$$\begin{aligned} \text{In[70]:=} & \partial_{\lambda} \varepsilon // \text{shortnotation} \\ \text{Out[70]=} & \frac{-e_{\lambda}^2 + e[\mathbf{x}, \mathbf{y}, \lambda] e_{\lambda\lambda}}{e[\mathbf{x}, \mathbf{y}, \lambda]^2} \end{aligned}$$

$$\begin{aligned} \text{In[71]:=} & \partial_{\mathbf{x},\lambda} \varepsilon // \text{shortnotation} \\ \text{Out[71]=} & \frac{e[\mathbf{x}, \mathbf{y}, \lambda]^2 e_{\mathbf{x}\lambda\lambda} + 2 e_{\mathbf{x}} e_{\lambda}^2 - e[\mathbf{x}, \mathbf{y}, \lambda] (2 e_{\mathbf{x}\lambda} e_{\lambda} + e_{\mathbf{x}} e_{\lambda\lambda})}{e[\mathbf{x}, \mathbf{y}, \lambda]^3} \end{aligned}$$

$$\begin{aligned} \text{In[72]:=} & \partial_{\mathbf{y},\lambda} \varepsilon // \text{shortnotation} \\ \text{Out[72]=} & \frac{e[\mathbf{x}, \mathbf{y}, \lambda]^2 e_{\mathbf{y}\lambda\lambda} + 2 e_{\mathbf{y}} e_{\lambda}^2 - e[\mathbf{x}, \mathbf{y}, \lambda] (2 e_{\mathbf{y}\lambda} e_{\lambda} + e_{\mathbf{y}} e_{\lambda\lambda})}{e[\mathbf{x}, \mathbf{y}, \lambda]^3} \end{aligned}$$

The gradient magnitude (detecting yellow-blue transitions) becomes:

$$\begin{aligned} \text{In[73]:=} & \mathcal{G} = \sqrt{(\partial_{\mathbf{x}} \varepsilon)^2 + (\partial_{\mathbf{y}} \varepsilon)^2}; \mathcal{G} // \text{shortnotation} \\ \text{Out[73]=} & \sqrt{\frac{(e[\mathbf{x}, \mathbf{y}, \lambda] e_{\mathbf{x}\lambda} - e_{\mathbf{x}} e_{\lambda})^2 + (e[\mathbf{x}, \mathbf{y}, \lambda] e_{\mathbf{y}\lambda} - e_{\mathbf{y}} e_{\lambda})^2}{e[\mathbf{x}, \mathbf{y}, \lambda]^4}} \end{aligned}$$

The second spectral order gradient (detecting purple-green transitions) becomes:

```
In[74]:=  $\mathcal{W} = \sqrt{(\partial_{x,\lambda}\varepsilon)^2 + (\partial_{y,\lambda}\varepsilon)^2}; \mathcal{W} // \text{shortnotation}$ 
```

```
Out[74]= 
$$\sqrt{\left(\frac{1}{e[x, y, \lambda]^6} \left( (e[x, y, \lambda]^2 e_{x\lambda\lambda} + 2 e_x e_\lambda^2 - e[x, y, \lambda] (2 e_{x\lambda} e_\lambda + e_x e_{\lambda\lambda}))^2 + (e[x, y, \lambda]^2 e_{y\lambda\lambda} + 2 e_y e_\lambda^2 - e[x, y, \lambda] (2 e_{y\lambda} e_\lambda + e_y e_{\lambda\lambda}))^2 \right)\right)}$$

```

Finally, the total edge strength \mathcal{N} (for all color edges) in the spatio-spectral domain becomes:

```
In[75]:=  $\mathcal{N} = \sqrt{(\partial_x\varepsilon)^2 + (\partial_y\varepsilon)^2 + (\partial_{x,\lambda}\varepsilon)^2 + (\partial_{y,\lambda}\varepsilon)^2}; \mathcal{N} // \text{shortnotation}$ 
```

```
Out[75]= 
$$\sqrt{\left(\frac{1}{e[x, y, \lambda]^6} \left( e[x, y, \lambda]^2 (e[x, y, \lambda] e_{x\lambda} - e_x e_\lambda)^2 + e[x, y, \lambda]^2 (e[x, y, \lambda] e_{y\lambda} - e_y e_\lambda)^2 + (e[x, y, \lambda]^2 e_{x\lambda\lambda} + 2 e_x e_\lambda^2 - e[x, y, \lambda] (2 e_{x\lambda} e_\lambda + e_x e_{\lambda\lambda}))^2 + (e[x, y, \lambda]^2 e_{y\lambda\lambda} + 2 e_y e_\lambda^2 - e[x, y, \lambda] (2 e_{y\lambda} e_\lambda + e_y e_{\lambda\lambda}))^2 \right)\right)}$$

```

As an example, we implement this last expression for discrete images.

As we did in the development of the multi-scale Gaussian derivative operators, we replace each occurrence of a derivative to λ with the respective plane in the observed image \mathbf{rf} (by the color receptive fields). Note that we use $\mathbf{rf}[[\mathbf{n}\lambda+1]]$ because the zero-th list element is the **Head** of the list. We recall the internal representation of a derivative in *Mathematica*:

```
In[76]:= FullForm[e(1,0,2)[x, y, λ]]
```

```
Out[76]//FullForm= Derivative[1, 0, 2][e][x, y, \[Lambda]]
```

We will look for such patterns and replace them with another pattern. We do this *pattern matching* with the command /. (**ReplaceAll**). We call the observed image at this stage \mathbf{rf} , without any assignment to data, so we can do all calculations symbolically first:

```
In[77]:= Clear[rf0, rf1, rf2, σ];
rf = {rf0, rf1, rf2};
N = N /. {Derivative[nx_, ny_, nλ_] [e] [x, y, λ] =>
Derivative[nx, ny] [rf[[nλ + 1]]] [x, y], e[x, y, λ] => rf[[1]]} // Simplify
```

Out[79]=

$$\sqrt{\left(\frac{1}{rf0^6} \left(rf0^2 \left(rf1[x, y] rf0^{(0,1)}[x, y] - rf0 rf1^{(0,1)}[x, y] \right)^2 + \right. \right. \\ \left. \left(2 rf1[x, y]^2 rf0^{(0,1)}[x, y] - 2 rf0 rf1[x, y] rf1^{(0,1)}[x, y] + \right. \right. \\ \left. \left. rf0 \left(-rf2[x, y] rf0^{(0,1)}[x, y] + rf0 rf2^{(0,1)}[x, y] \right) \right)^2 + \right. \\ \left. rf0^2 \left(rf1[x, y] rf0^{(1,0)}[x, y] - rf0 rf1^{(1,0)}[x, y] \right)^2 + \right. \\ \left. \left(2 rf1[x, y]^2 rf0^{(1,0)}[x, y] - 2 rf0 rf1[x, y] rf1^{(1,0)}[x, y] + \right. \right. \\ \left. \left. rf0 \left(-rf2[x, y] rf0^{(1,0)}[x, y] + rf0 rf2^{(1,0)}[x, y] \right) \right)^2 \right)$$

Note that we do a delayed rule assignment here (\Rightarrow instead of \rightarrow) because we want to evaluate the right hand side only after the rule is applied. We finally replace the spatial derivatives with our familiar spatial Gaussian derivative convolution \mathbf{gD} at scale σ :

```
In[80]:= N = N /. {Derivative[nx_, ny_] [rf_] [x, y] => gD[rf, nx, ny, σ],
rf1[x, y] => rf1, rf2[x, y] => rf2}
```

Out[80]=

$$\sqrt{\left(\frac{1}{rf0^6} \left(rf0^2 \left(rf1 \text{DerivativeFilter}[rf0, \{0, 1\}, \sigma] - \right. \right. \right. \\ \left. \left. rf0 \text{DerivativeFilter}[rf1, \{0, 1\}, \sigma] \right)^2 + \right. \\ \left. rf0^2 \left(rf1 \text{DerivativeFilter}[rf0, \{1, 0\}, \sigma] - \right. \right. \\ \left. \left. rf0 \text{DerivativeFilter}[rf1, \{1, 0\}, \sigma] \right)^2 + \right. \\ \left. \left(2 rf1^2 \text{DerivativeFilter}[rf0, \{0, 1\}, \sigma] - 2 rf0 rf1 \text{DerivativeFilter}[\right. \right. \\ \left. \left. rf1, \{0, 1\}, \sigma] + rf0 \left(-rf2 \text{DerivativeFilter}[rf0, \{0, 1\}, \sigma] + \right. \right. \right. \\ \left. \left. \left. rf0 \text{DerivativeFilter}[rf2, \{0, 1\}, \sigma] \right) \right)^2 + \right. \\ \left. \left(2 rf1^2 \text{DerivativeFilter}[rf0, \{1, 0\}, \sigma] - 2 rf0 rf1 \text{DerivativeFilter}[\right. \right. \\ \left. \left. rf1, \{1, 0\}, \sigma] + rf0 \left(-rf2 \text{DerivativeFilter}[rf0, \{1, 0\}, \sigma] + \right. \right. \right. \\ \left. \left. \left. rf0 \text{DerivativeFilter}[rf2, \{1, 0\}, \sigma] \right) \right)^2 \right)$$

This expression can now safely be calculated on the discrete data:

```
In[81]:= {rf0, rf1, rf2} = obs;  $\sigma = 1.$ ;  
ArrayPlot[- $\mathcal{N}$ , PlotRange  $\rightarrow$  {-0.4, 0}, ImageSize  $\rightarrow$  300]  
Out[82]=
```

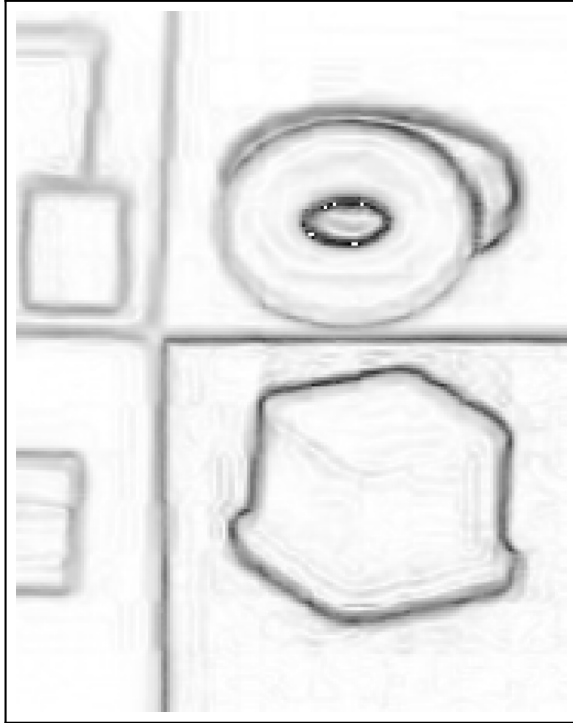


Figure 18.12 The color-invariant \mathcal{N} calculated for our input image at spatial scale $\sigma = 1$ pixel. Primarily the total color edges are found, with little edge detection at intensity edges. Image resolution 228x179 pixels.

The following routines are implemented as standard FEV functions:

- $\mathcal{E}[\mathbf{im}, \sigma]$ $\mathcal{E} = \frac{1}{e} \frac{\partial e}{\partial \lambda}$ color invariant $\frac{1}{e} \frac{\partial e}{\partial \lambda}$
- $\mathcal{E}\lambda[\mathbf{im}, \sigma]$ $\frac{\partial \mathcal{E}}{\partial \lambda}$ first wavelength derivative of \mathcal{E}
- $\mathcal{E}\lambda\lambda[\mathbf{im}, \sigma]$ $\frac{\partial^2 \mathcal{E}}{\partial \lambda^2}$ second wavelength derivative of \mathcal{E}
- $\mathcal{G}\mathcal{G}[\mathbf{im}, \sigma]$ $\sqrt{\left(\frac{\partial \mathcal{E}}{\partial x}\right)^2 + \left(\frac{\partial \mathcal{E}}{\partial y}\right)^2}$ yellow-blue edges
- $\mathcal{G}\mathcal{W}[\mathbf{im}, \sigma]$ $\sqrt{\left(\frac{\partial^2 \mathcal{E}}{\partial x \partial \lambda}\right)^2 + \left(\frac{\partial^2 \mathcal{E}}{\partial y \partial \lambda}\right)^2}$ red-green edges


```

In[88]:=  $\varepsilon\lambda$ [im_,  $\sigma$ ] :=
Module[{colorRF, observedimage}, colorRF = ( {
  {0.621, 0.113, 0.194},
  {0.297, 0.563, 0.049},
  {-0.009, 0.027, 1.105}
} ). ( {
  {-0.019, 0.048, 0.011},
  {0.019, 0., -0.016},
  {0.047, -0.052, 0.}
} ); observedimage = Map[colorRF.#1 &, im, {2}];
Transpose[observedimage, {2, 3, 1}][[3]];

In[89]:=  $g$ [im_,  $\sigma$ ] :=
Module[{ $\mathcal{G}$ out, rgb2xyz, xyz2e, observedimage, obs, rf,
  rf0, rf1, rf2,  $\lambda$ , e,  $\varepsilon$ }, rgb2xyz = ( {
  {0.621, 0.113, 0.194},
  {0.297, 0.563, 0.049},
  {-0.009, 0.027, 1.105}
} );
xyz2e = ( {
  {-0.019, 0.048, 0.011},
  {0.019, 0., -0.016},
  {0.047, -0.052, 0.}
} );
observedimage = Map[rgb2xyz.xyz2e.#1 &, im, {2}];
obs = Transpose[observedimage, {2, 3, 1}];
 $\varepsilon$  :=  $\partial_\lambda e[x, y, \lambda] / e[x, y, \lambda]$ ;
 $\mathcal{G}$ out = Simplify[Sqrt[( $\partial_x \varepsilon$ )^2 + ( $\partial_y \varepsilon$ )^2]];
Clear[rf0, rf1, rf2];
rf = {rf0, rf1, rf2};  $\mathcal{G}$ out =
Simplify[ $\mathcal{G}$ out /. {Derivative[nx_, ny_,
  n $\lambda$ ][e][x, y,  $\lambda$ ]  $\Rightarrow$ 
  Derivative[nx, ny][rf[[n $\lambda$  + 1]][x, y],
  e[x, y,  $\lambda$ ]  $\Rightarrow$  rf[[1]]}];
 $\mathcal{G}$ out =  $\mathcal{G}$ out /. {Derivative[nx_,
  ny_][rf_][x, y]  $\Rightarrow$  gD[rf, nx, ny,  $\sigma$ ],
  rf1[x, y]  $\Rightarrow$  rf1, rf2[x, y]  $\Rightarrow$  rf2}; {rf0, rf1, rf2} =
obs;  $\mathcal{G}$ out]

```



```

In[91]:= gW[im_, σ_] :=
Module[{gout, rgb2xyz, xyz2e, observedimage, obs, rf,
  rf0, rf1, rf2, λ, e, ε}, rgb2xyz = ( {
  {0.621, 0.113, 0.194},
  {0.297, 0.563, 0.049},
  {-0.009, 0.027, 1.105}
});
xyz2e = ( {
  {-0.019, 0.048, 0.011},
  {0.019, 0., -0.016},
  {0.047, -0.052, 0.}
});
observedimage = Map[rgb2xyz.xyz2e.#1 &, im, {2}];
obs = Transpose[observedimage, {2, 3, 1}];
ε :=
D[e[x, y, λ], λ] / e[x, y, λ];
wout =
Simplify[
  Sqrt[D[ε, x, λ]^2 +
  D[ε, y, λ]^2]];
Clear[rf0, rf1, rf2]; rf = {rf0, rf1, rf2};
wout =
Simplify[wout /. {Derivative[nx_, ny_,
  nλ_][e][x, y, λ] =>
  Derivative[nx, ny][rf[[nλ + 1]][x, y],
  e[x, y, λ] => rf[[1]]}];
wout = wout /. {Derivative[nx_,
  ny_][rf_][x, y] => gD[rf, nx, ny, σ],
  rf1[x, y] => rf1, rf2[x, y] => rf2};
{rf0, rf1, rf2} = obs; wout]

```

Here is an example of finding blue-yellow edges:

```
In[92]:= image = Import[url <> "terre_indigo.jpg"];
im = ImageData[image];
GraphicsRow[{Show[image], ArrayPlot[-g $\mathcal{G}$ [im, 1], PlotRange -> {-0.4, -0.02}],
ImageSize -> 500]
```

Out[94]=



Figure 18.13 The yellow-blue edge detector $\mathcal{G} = \sqrt{\left(\frac{\partial \mathcal{E}}{\partial x}\right)^2 + \left(\frac{\partial \mathcal{E}}{\partial y}\right)^2}$ calculated at a spatial scale $\sigma = 1$ pixel. Image resolution 288x352. Note the absence of black-white intensity edges in the day and time indication. Example due to J. Sporing.

This example shows the color-selectivity of the spatio-color differential invariants:

```
In[95]:= image = Import[url <> "colortoy.jpg"];
In[96]:= im = ImageData[image];
GraphicsRow[
{Show[image], ArrayPlot[-g $\mathcal{G}$ [im,  $\sigma = 1.$ ], PlotRange -> {-0.75, -0.02}],
ArrayPlot[-g $\mathcal{W}$ [im,  $\sigma = 1.$ ], PlotRange -> {-2.1, 0}], ImageSize -> 500]
```

Out[97]=

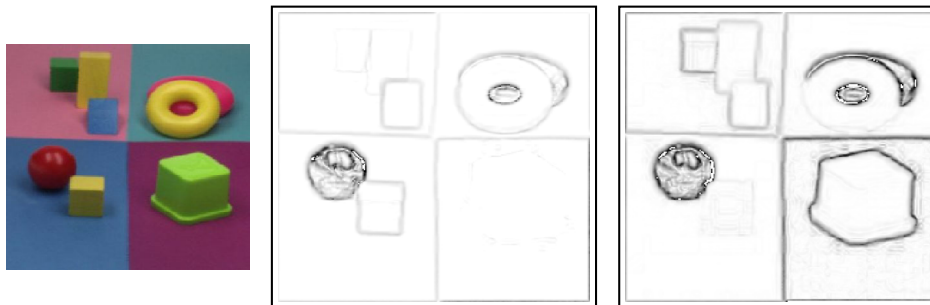


Figure 18.14 Detection of the yellow-blue edge detector $\mathcal{G} = \sqrt{\left(\frac{\partial \mathcal{E}}{\partial x}\right)^2 + \left(\frac{\partial \mathcal{E}}{\partial y}\right)^2}$ (middle) and the red-green edge detector $\mathcal{W} = \sqrt{\left(\frac{\partial^2 \mathcal{E}}{\partial x \partial \lambda}\right)^2 + \left(\frac{\partial^2 \mathcal{E}}{\partial y \partial \lambda}\right)^2}$ (right) at a scale of $\sigma = 1$ pixel.

As a histological example, spatio-color differential structure can accentuate staining patterns. The next example shows the advantage of a specific red-green detection in the localization of the edges of a stained nucleus in *paramecium caudatum*, a common freshwater inhabitant with ciliary locomotion.

```
In[98]:= im = ImageData[image = Import[url <> "Paramecium_caudatum.jpg"]];
In[99]:= GraphicsRow[
  {Show[image], ArrayPlot[-gG[im, σ = 3.], PlotRange → {-0.012, 0}],
  ArrayPlot[-gW[im, σ = 3.], PlotRange → {-0.15, 0}]}, ImageSize → 500]
Out[99]=
```

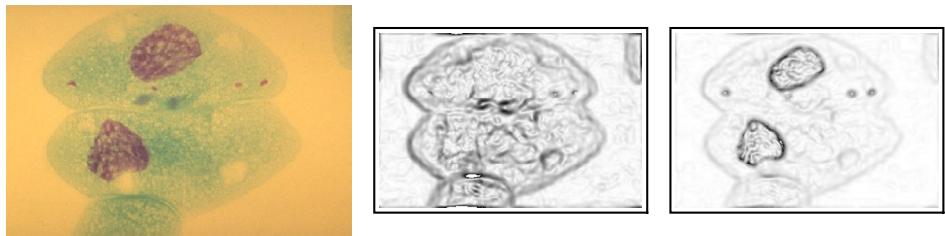


Figure 18.15 The color-invariant \mathcal{G} (middle) and \mathcal{W} (right) calculated for a histological image of a fungus cell, *paramecium caudatum* (left). The red-green color edges found by \mathcal{W} form a good delineation of the cell's nucleus.

18.5 Combination with spatial constraints

Interesting combinations can be made when we combine the color differential operators with the spatial differential operators. E.g. when we want to detect specific blobs with a specific size and color, we can apply feature detectors that are best matching the shape to be found. We end the chapter with two examples: color detection of blobs in a regular pattern, and locating stained nuclei in a histological preparation.

```
im = ImageData[Import[url <> "colorblobs.gif"]];
imbw = ImageData[
  ColorConvert[im = Import[url <> "colorblobs.gif"], "Grayscale"]];
lww = gauge2DN[imbw, 0, 2, 2];

lgc = gD[imbw, 2, 0, 2] × gD[imbw, 0, 2, 2] - gD[imbw, 1, 1, 2]2;
e1 = ελ[im, 2];
e11 = ελλ[im, 4];
t1 = Map[If[# > 0, 1, 0] &, lww, {2}];
t2 = Map[If[# > 0, 1, 0] &, lgc, {2}];
t3 = Map[If[# > 0, 1, 0] &, e1, {2}];
t4 = Map[If[# > 0.001, 1, 0] &, e11, {2}];
t5 = Map[If[# > 0, 1, 0] &, e1 + e11, {2}];

imbw = colorToBW[im = rasterpixels[image = Import["colorblobs.gif"]]];
lww = gauge2DN[imbw, 0, 2, 2];
lgc = gD[imbw, 2, 0, 2] × gD[imbw, 0, 2, 2] - gD[imbw, 1, 1, 2]2;
e1 = ελ[im, 2];
e11 = ελλ[im, 4];
t1 = Map[If[# > 0, 1, 0] &, lww, {2}];
t2 = Map[If[# > 0, 1, 0] &, lgc, {2}];
t3 = Map[If[# > 0, 1, 0] &, e1, {2}];
t4 = Map[If[# > 0.001, 1, 0] &, e11, {2}];
t5 = Map[If[# > 0, 1, 0] &, e1 + e11, {2}];

p1 = Show[image];
blue = ArrayPlot[(1 - t1) t2 (1 - t5) (1 - (1 - t5) t3)];
yellow = ArrayPlot[(1 - t1) t2 (1 - t4) t5];
redandmagenta = ArrayPlot[(1 - t1) t2 t4];
Show[GraphicsRow[{p1, blue, yellow, redandmagenta}], ImageSize → 420]
```

```

Block[{$DisplayFunction = Identity},
  p1 = Show[image];
  blue = ListDensityPlot[(1 - t1) t2 (1 - t5) (1 - (1 - t5) t3)];
  yellow = ListDensityPlot[(1 - t1) t2 (1 - t4) t5];
  redandmagenta = ListDensityPlot[(1 - t1) t2 t4];
  Show[GraphicsArray[{p1, blue, yellow, redandmagenta}], ImageSize -> 420]

```

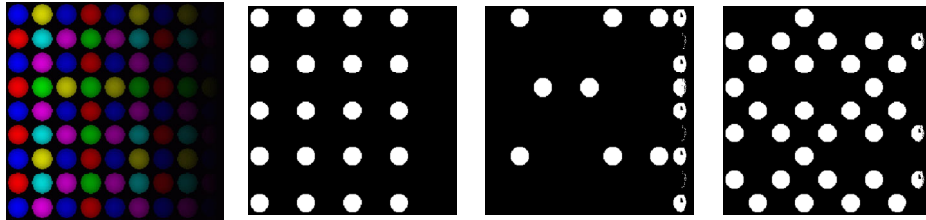


Figure 18.16 Detection of color blobs with spatial and color differential operators. Blobs are spatially described as locations with positive Gaussian curvature and positive second order directional derivative in the gradient direction of the image intensity, color is a boolean combination of the color differential features. Example due to P. van Osta.

Task 18.1 Find the total detection scheme for all individual colors in the example above, and for combinations of two and three colors.

A last example detects stained carbohydrate deposits in a histological application.

```

imbw = colorToBW[im = rasterpixels[image = Import["pas.jpg"]]];
lww = gauge2DN[imbw, 0, 2, 4];
lgc = gD[imbw, 2, 0, 4] × gD[imbw, 0, 2, 4] - gD[imbw, 1, 1, 4]2;
e1 = ελ[im, 4];
e11 = ελλ[im, 4];
t1 = Map[If[# > 0, 1, 0] &, lww, {2}];
t2 = Map[If[# > 0.2, 1, 0] &, lgc, {2}];
t6 = Map[If[# > 0, 1, 0] &, e11 - e1, {2}];

DisplayTogetherArray[
  {Show[image], ListDensityPlot[t1 t2 t6]}, ImageSize -> 500]

```

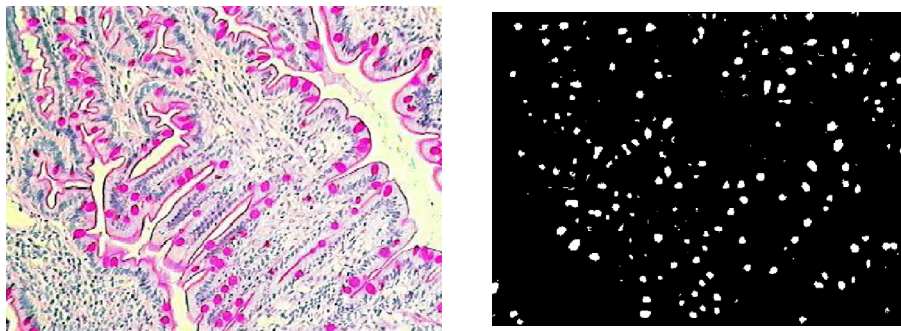


Figure 18.17 Detection of carbohydrate stacking in cells, that are specifically stained for carbohydrates with periodic acid Schiff (P.A.S.). The carbohydrate deposits are in magenta, cell nuclei

in blue. The blob-like areas are detected with positive Gaussian curvature and positive L_{ww} , the magenta with a boolean combination of the color invariant \mathcal{E} and its derivatives to λ . Example due to P. Van Osta. Image taken from <http://www.bris.ac.uk/Depts/PathAndMicro/CPL/pas.html>

18.6 Summary of this chapter

The scale-space model for color differential structure, as developed by Koenderink, states that the wavelength differential structure is measured at a single scale ($\sigma_\lambda = 55$ nm) for zeroth order (intensity), first order (blue-yellow) and second order (red-green). There is a good analogy with the color receptive field structures found in the mammalian visual cortex.

The chapter shows an actual implementation of this color differential structure. The wavelength derivatives can be extracted from the transformed RGB triples in the data.

In a similar way as for the spatial differential structure analysis of gray valued data, invariants can be defined, with combined color and spatial coordinate transformation invariance. A number of these color-spatial differential invariants are discussed and implemented on examples.