Gaussian derivatives and Eigen-images

```
<< MathVisionTools`;
SetOptions[RasterPlot, Frame -> False];
readpixels[im_] := ColorSeparate[Import[im]][[1, 1]];
```

MathVisionTools

It has been shown that the so-called Eigen-images of a large series of image small patches have great similarity to partial Gaussian derivative functions [Olshausen1996, Olshausen1997]. The resulting images are also often modeled as Gabor patches and wavelets. In this section we will explain the notion of Eigen-images and study this statistical technique with Mathematica.

We read the many patches as small square subimages of $\delta = 12$ x12 pixels, slightly overlapping, at 25 horizontal and 25 vertical positions, leading to a series of 625 patches. Figure 12.1 (next page) shows the location of the patches. These 625 images form the input set.

```
im = readpixels["forest06.gif"]; δ = 12;
RasterPlot[im,
 Epilog -> {Gray, Table[Line[{{i, j}, {i + δ, j},
       {i + δ, j + δ}, {i, j + δ}, {i, j}}],
    {j, 2, 251, 10}, {i, 2, 251, 10}]}, ImageSize -> 400]
```

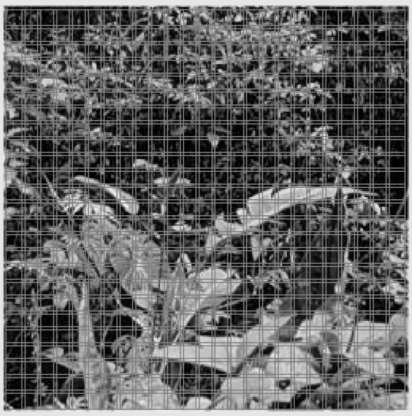


Figure 1. Location of the 625 small 12x12 pixel patches taken from a 256^2 image of a forest scene.

The small 12x12 images are sampled with **SubMatrix**:

```
δ = 12; set = Table[Take[im, {j, j + δ - 1}, {i, i + δ - 1}],
   {j, 2, 250, 10}, {i, 2, 250, 10}]; Dimensions[set]
```

```
{25, 25, 12, 12}
```

and converted into a matrix **m** with 289 rows of length 144. We multiply each small image with a Gaussian weighting function to simulate the process of observation, and subtract the global mean:

```
σ = 4;
g = Table[Exp[-(x^2 + y^2)/(2 σ^2)], {x, -5.5, 5.5}, {y, -5.5, 5.5}];
set2 = Map[g # &, set, {2}];
```

```
m = Flatten[Map[Flatten, set2, {2}], 1];
mean = (Plus @@ #)/(Length[#]) &;
m = N[m - mean[Flatten[m]]]; Dimensions[m]
```

```
{625, 144}
```

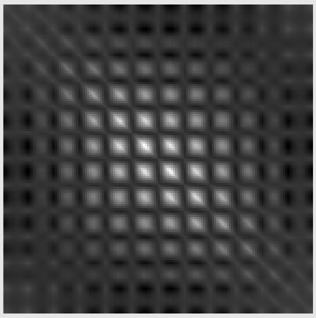
```
RasterPlot[m]
```

We calculate $m^T m$, a 144^2 matrix with the **Dot** product, and check that it is a square matrix:

```
Dimensions[mTm = N[Transpose[m] .m]]
```

```
{144, 144}
```

```
RasterPlot[mTm]
```

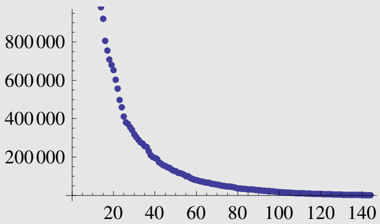


The calculation of the 144 Eigen-values of a 144^2 matrix goes fast in Mathematica. Essential is to force the calculations to be done numerically with the function **N[]**. Because **mTm** is a symmetric matrix, built from two 289x144 size matrices, we have 144 (nonzero) Eigen-values:

```
Short[Timing[evs = eigenvalues = Eigenvalues[mTm]], 5]
```

```
{0., {6.31627×10^7, 1.95781×10^7, 9.56876×10^6, 6.30868×10^6,
  3.01455×10^6, <<135>>, 1084.8, 990.491, 912.436, 798.279}}
```

```
ListPlot[evs]
```



We calculate the **Eigenvectors** of the matrix **mTm** and construct the first Eigen-image by partitioning the resulting 144x1 vector 12 rows. All Eigen-vectors normalized: unity length.

```
SetOptions[ListPlot3D, Axes → False];
```

```
eigenvectors = Eigenvectors[mTm]; eigenimages =
 Table[Partition[eigenvectors〚i〛, δ], {i, 1, 8}];
GraphicsRow[RasterPlot /@ eigenimages, ImageSize → 800]
GraphicsRow[ListPlot3D /@ eigenimages, ImageSize → 800]
```

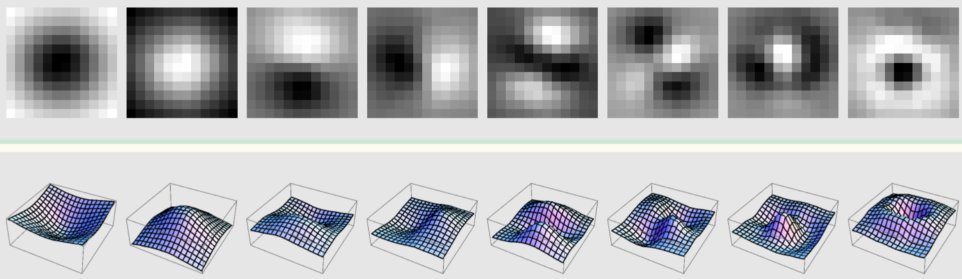


Figure 2. The first 8 Eigen-images of the 289 patches from figure 12.10.

Here are the Eigen-images for white noise (we take the same 289 12x12 patches again):

```
noise = RandomReal[{0, 1}, {256, 256}]; δ = 12;
set = Table[Take[noise, {j, j + δ - 1}, {i, i + δ - 1}],
   {j, 3, 256, 15}, {i, 3, 256, 15}];
m = Flatten[Map[Flatten, set, {2}], 1];
m = N[m - mean[Flatten[m]]]; mTm = N[Transpose[m].m];
{eigenvaluesn, eigenvectorsn} = Eigensystem[mTm];
eigenimagesn =
  Table[Partition[eigenvectorsn〚i〛, δ], {i, 1, 8}];
GraphicsRow[RasterPlot /@ eigenimagesn, ImageSize → 460]
```

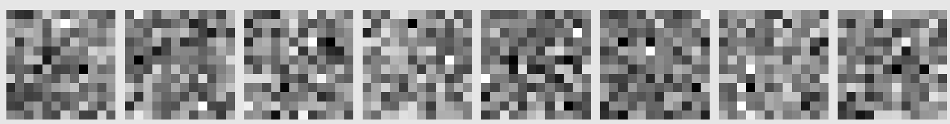


Figure 3. The first 8 eigen-images of 289 patches of 12x12 pixels of white noise. Note that none of the eigen-images contains any structure.

A striking result is obtained when the image contains primarily vertical structures, like trees. We then obtain Eigenpatches resembling the horizontal high order Gaussian derivatives / Gabor patches (see figure 12.4).

```
im = readpixels["forest02.gif"]; δ = 12;
set = Table[Take[im, {j, j + δ - 1}, {i, i + δ - 1}],
   {j, 2, 246, 5}, {i, 2, 246, 5}];
δδ = (δ - 1) / 2; σ = δδ; g = Table[N[Exp[-(x^2 + y^2)/(2 σ^2)]],
   {x, -δδ, δδ}, {y, -δδ, δδ}]; set2 = Map[g # &, set, {2}];
m = Flatten[Map[Flatten, set2, {2}], 1];
mean = (Plus @@ #)/(Length[#]) &;
m = N[m - mean[Flatten[m]]]; mTm = N[Transpose[m].m];
eigenvectors = Eigenvectors[mTm]; eigenimages =
  Table[Partition[eigenvectors[[i]], δ], {i, 1, 25}];
```

```
p1 = RasterPlot[im]; p2 = Show[GraphicsGrid[
   Partition[RasterPlot /@ eigenimages, 5]]];
Show[GraphicsRow[{p1, p2}, ImageSize → 800]]
```

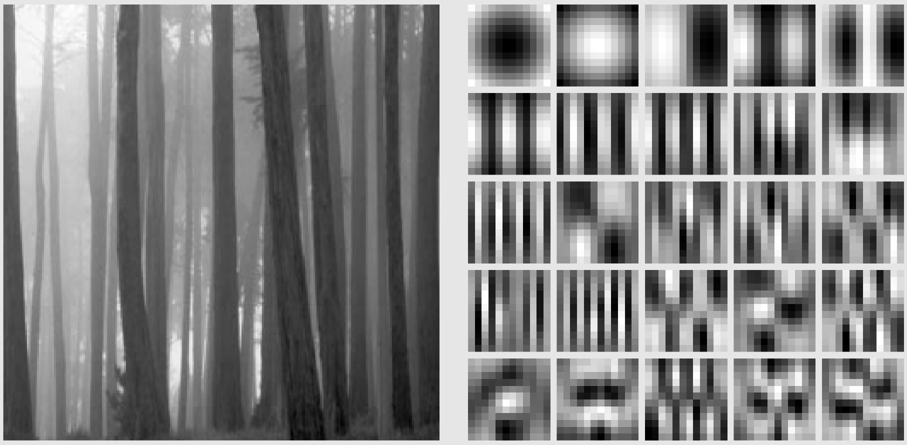


Figure 4. Eigen-images for 2401 slightly overlapping patches of 12x12 pixels from the image with the trees. Due to the larger number of patches we get better statistics. Note that the first 9 eigenpatches resemble the high order horizontal derivatives.