Algorithm details

Let `SampleImage` contain the image we are sampling from and let `Image` be the mostly empty image that we want to fill in (if synthesizing from scratch, it should contain a 3-by-3 seed in the center randomly taken from `SampleImage`, for constrained synthesis it should contain all the known pixels). `WindowSize`, the size of the neighborhood window, is the only user-settable parameter. The main portion of the algorithm is presented below:

```
function GrowImage(SampleImage,Image,WindowSize)
  while Image not filled do
    progress = 0
    PixelList = GetUnfilledNeighbors(Image)
    foreach Pixel in PixelList do
      Template = GetNeighborhoodWindow(Pixel)
      BestMatches = FindMatches(Template, SampleImage)
      BestMatch = RandomPick(BestMatches)
      if (BestMatch.error < MaxErrThreshold) then
        Pixel.value = BestMatch.value
        progress = 1
      end
    end
    if progress == 0
      then MaxErrThreshold = MaxErrThreshold * 1.1
  end
  return Image
end
```

Function `GetUnfilledNeighbors()` returns a list of all unfilled pixels that have filled pixels as their neighbors (the image is subtracted from its morphological dilation). The list is randomly permuted and then sorted by decreasing number of filled neighbor pixels. `GetNeigborhoodWindow()` returns a window of size `WindowSize` around a given pixel. `RandomPick()` picks an element randomly from the list. `FindMatches()` is as follows:

```
function FindMatches(Template,SampleImage)
  ValidMask = 1s where Template is filled, 0s otherwise
  GaussMask = Gaussian2D(WindowSize,Sigma)
  TotWeight = sum i,j GaussiMask(i,j)*ValidMask(i,j)
  for i,j do
    for ii,jj do
      dist = (Template(ii,jj)-SampleImage(i-ii,j-jj))^2
      SSD(i,j) = SSD(i,j) + dist*ValidMask(ii,jj)*GaussMask(ii,jj)
    end
    SSD(i,j) = SSD(i,j) / TotWeight
  end
  PixelList = all pixels (i,j) where SSD(i,j) <= min(SSD)*(1+ErrThreshold)
  return PixelList
end
```

`Gaussian2D()` generates a two-dimensional Gaussian in a window of given a size centered in the center and with a given standard deviation (in pixels). In our implementation the constant were set as follows: `ErrThreshold = 0.1`, `MaxErrThreshold = 0.3`, `Sigma = WindowSize/6.4`. Pixel values are in the range of 0 to 1.