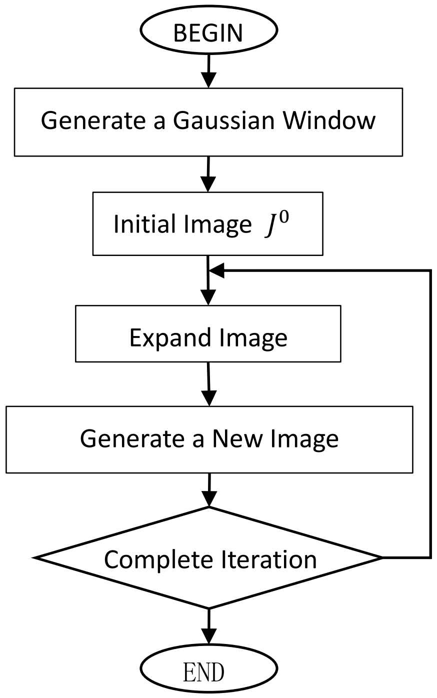
Rolling Guidance Filter

According to the paper, we can set the following flow chart.



Generate a Gaussian Window

We first define the structure scale as *the smallest Gaussian standard deviation* σ_s such that when this σsdeviation Gaussian is applied to an image, corresponding structure disappears. We denote the convolution process with the input image I and Gaussian $g_v(x,y)$ of variance $v=\sigma^2$ as

$$L_v = g_v * I$$

where $g_v = \frac{1}{\sqrt{2\pi v}}\exp(-\frac{x^2+y^2}{2v})$ and $*$ denotes convolution. L_v is the result at scale v. In scale-space theory, v is referred to as the scale parameter. When the image structure scale is smaller than$\sqrt{v}$ (i.e., σ_s), it will be completely removed in L_v.When applying Gaussians with varying σ_s to the image, structures are suppressed differently according to their sizes.

This function will generate g_v, sigma_s is σ_s.

```
function GaussianWindow = WindowBlock(sigma_s, GaussianPrecision)

%right below
pq = bsxfun(@plus, ([0:sigma_s*3].^2)', [0:sigma_s*3].^2);

% gaussian distribution
pqrb = exp(-pq/2/sigma_s^2);

% element that is less than GaussianPrecision ar equal zero
pqrb = pqrb .* (pqrb>GaussianPrecision);

% remove all zero column
```

```
pqrb(:, pqrb(1,:)==0) = [];

% remove all zero row
pqrb(pqrb(:,1)==0, :) = [];

%left below
pqlb = fliplr(pqrb);

%right upper
pqru = flipud(pqrb);

%left upper
pqlu = fliplr(pqru);

GaussianWindow = [pqlu(:, 1:end-1)     pqru;
                  pqlb(2:end, 1:end-1) pqrb(2:end, :)];

end
```

Expand Image

On the edge of the pixels extend outward N pixels.
Image is the origic image. N is the number of expansion need.

```
function imageExpand = ExpandBorder(image, N)

imageExpand = [repmat(image(1,:,:), [N,1,1]) ; image ; repmat(image(end,:,:),
[N,1,1])];
imageExpand = [repmat(imageExpand(:,1,:), [1,N,1])  imageExpand
repmat(imageExpand(:,end,:), [1,N,1])];

end
```

Generate a New Image

In this process, an image J is iteratively updated. We denote J^{t+1} as the result in the t-th iteration. The value of J^{t+1} in the t-th iteration is obtained in a joint bilateral filtering form given the input I and the value in previous iteration J^t:

$$J^{t+1}(p) = \frac{1}{K_p} \sum_{q \in N(p)} \exp\left(-\frac{\|p-q\|^2}{2\sigma_s^2} - \frac{\|J^t(p) - J^t(q)\|^2}{2\sigma_r^2}\right) I(q),$$

where

$$K_p = \sum_{q \in N(p)} \exp\left(-\frac{\|p-q\|^2}{2\sigma_s^2} - \frac{\|J^t(p) - J^t(q)\|^2}{2\sigma_r^2}\right)$$

for normalization. I is the same input image. σ_s and σ_r control the spatial and range weights respectively.

The following is the corresponding code:

```
H = GaussianWindow .* exp( -(J(i-N:i+N,j-N:j+N,k) - J(i,j,k)).^2/2/sigma_r^2 );
```

```
K_p = sum(sum(H));
J_plus(i-N,j-N,k) = 1/K_p*sum(sum(H .* I(i-N:i+N,j-N:j+N,k)));
```