Rolling Guidance Filter

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

BEGIN

Generate a Gaussian Window

Expand Image

Generate a New Image

Complete Iteration

END

Initial Image

Generate a Gaussian Window

We first define the structure scale as *the smallest Gaussian standard deviation* 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 of variance as

where and ∗ denotes convolution. is the result at scale v. In scale-space theory, is referred to as the scale parameter. When the image structure scale is smaller than (i.e., ), it will be completely removed in .When applying Gaussians with varying to the image, structures are suppressed differently according to their sizes.

This function will generate ，sigma\_s is .

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 is iteratively updated. We denote as the result in the t-th iteration. The value of in the t-th iteration is obtained in a joint bilateral filtering form given the input and the value in previous iteration :

where

for normalization. is the same input image. and 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)));