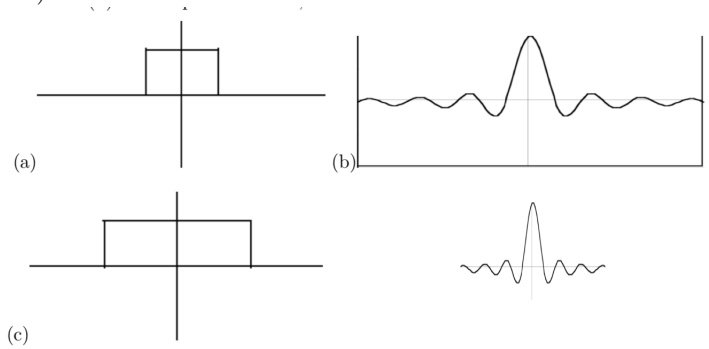
Filter Algorithm

1. Duality of Domains

1.1)



1.2) We use triangle function to approximate sinc function for practical application in code. Sinc has infinite extent in theory, even if the weights away from the center are small. However, computing convolution integral involves discrete multiplication of sample pixel values with discrete filter function value. So, we make approximations to triangle function which is easier to compute and has a finite extent.

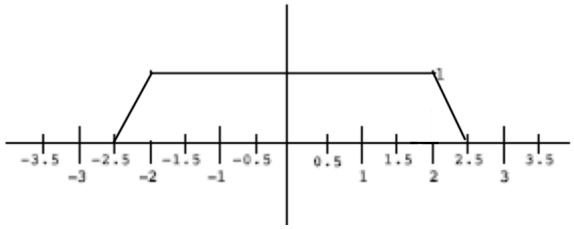
1.3) Discrete sample points of a continuous image.

2. Convolution

2.1)

a. x = 2.5: 0
b. x = 2.25: 0.25
c. x = 1.75: 0.75
d. x = 1.25: 1

2.2)

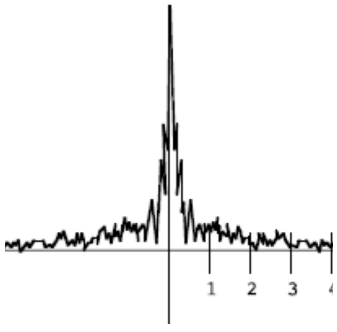


2.3) $f(x) * g(x) = F(X)G(X)$

3. Prefiltering

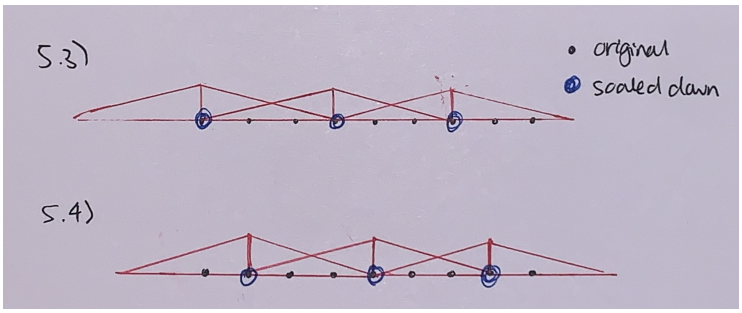
3.1) Largest frequency we can represent is n/2 per unit, according to the Nyquist limit.

3.2)



4. Blur
   4.1) Ideal blur filter in the frequency domain is a box filter, which eliminates high frequencies.
   4.2) The ideal blur filter in the spatial domain is a convolution with sinc function.
   4.3)$\sum_{n=x-2.5}^{x+2.5} image[n]kernel[n-x+2.5]$
   4.4) If the blur radius extends past the edge of the image, the convolution will be computed assuming that the pixel value of outer region is zero. Assuming that the product of filter and pixel values will be stored in an array, the sum can be normalized to account for unused part of filter.

5. Scaling
   5.1) 2
   5.2) $\frac{2}{n}$



   5.5)

   Scaled in X:
   (-1/6, 0) (1/2, 0) (7/6, 0)
   (-1/6, !) (1/2, 1) (7/6, 1)
   Scaled in Y:
   (-1/6, -1/6) (1/2, -1/6) (7/6, -1/6)
   (-1/6, 1/2) (1/2, 1/2) (7/6, 1/2)
   (-1/6, 7/6) (1/2, 7/6) (7/6, 7/6)

   5.6)

   a. Linear
   b. Gaussian
   c. Bad
   d. Triangle