**Computer Vision Notes**

*Created: 2024-09-06*

*Updated: 2024-09-23*

**REFERENCES**:

* Introduction to Image Understanding course at the University of Toronto

**LINEAR FILTERS** (TODO: Tb ch 3.2)

***Digital Image***: a map or a matrix of integer intensity values , is in a grayscale image, in a color image.

Problem: want to locate object in image.

Solution: slide and compare the image of the object.

Problem: noise in image.

Solution: modify pixel by applying function on a neighborhood of pixels e.g., average neighbors (assumes neighbors similar, noise independent) using moving average with (non-)uniform weights.

***Correlation*** (cv2.filter2D, 2D moving average with (non-)uniform weights): Given input , where

where size of the weight **kernel/mask** is and its entries are **filter coefficients**.

where .

Let where , then

***Normalized Cross-Correlation***: exact match of image crop and filter results in 1. Normalized prevents that is all or almost all white (255) to generate large response.

**Types of Filters**

Sharpening Filter:

Gaussian Filter: smooth/blur, reduce noise, neighbors closest to a center have the most influence.

Generic Gaussian Filter: anisotropic (asymmetric), .

**Effect of Size of Filter and Variance**

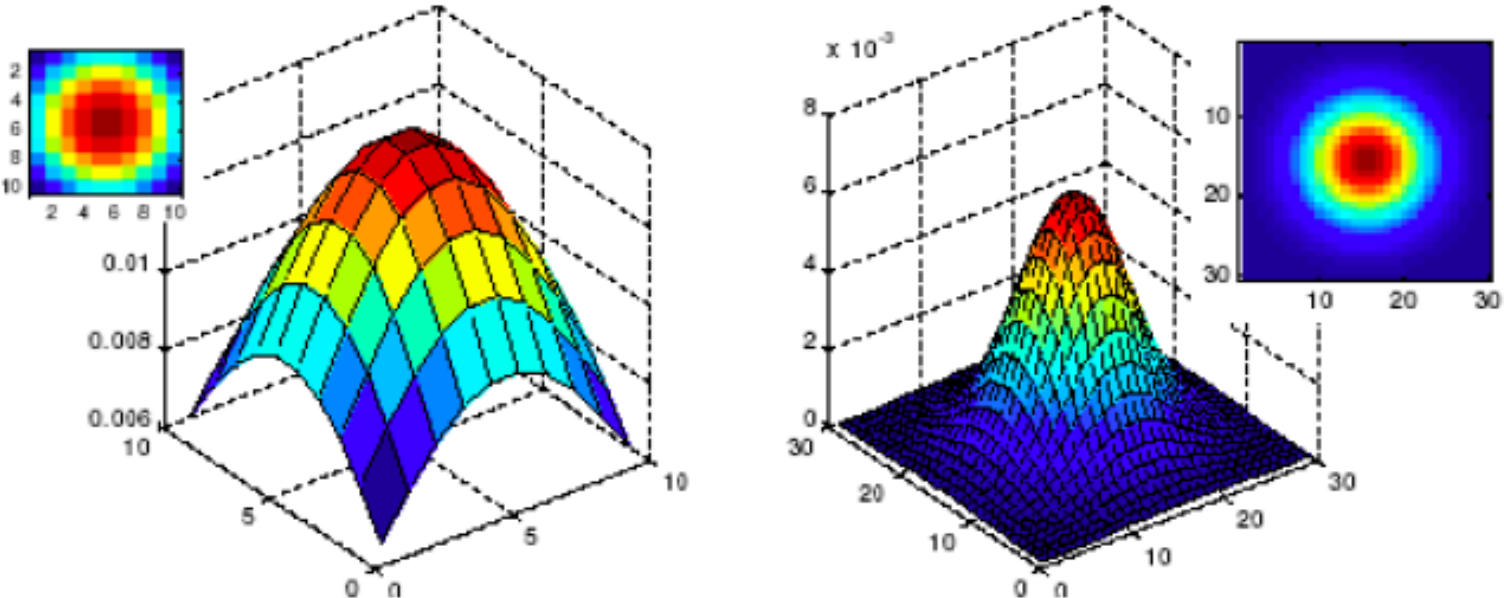


Figure: same different filter/mask/kernel size 10x10 vs 30x30.

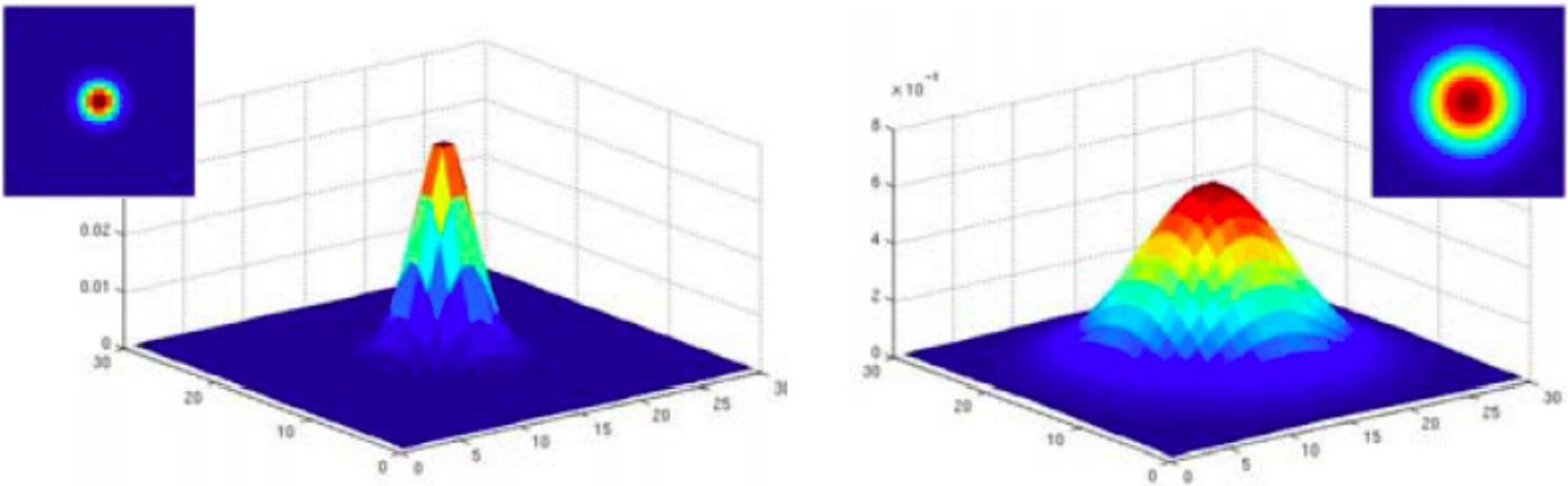


Figure: same size 30x30, (left) (right), larger is more smoothing.

**Properties of Smoothing**

* All values positive
* Sum to 1; prevents rescaling image
* Low-pass filter; removes high frequency (rate of change in pixel intensity values) components which include edges.

***Convolution***: operator that flips filter horizontally and vertically then applies correlation. Given input , where

**Properties of Convolution**

|  |  |
| --- | --- |
| Commutative |  |
| Associative |  |
| Distributive |  |
| Associative with scalar multiplier |  |
| Convolution Theorem ( is Fourier Transform) |  |

**Implications of Convolution Theorem**

Method 1: convolution runs in .

Method 2: FFT and IFFT run in and mult in .

**Separable Filters**

[TODO]

**DEEP LEARNING FOR COMPUTER VISION**

* Template matching is inadequate for object recognition when there are difficult scene conditions (occlusion, changes in viewing angle, articulation of parts) and too many variations.
* Use networks instead by collecting training images and labels, training a classifier, evaluate the classifier.

**Linear Model for Image Classification**

Where is a vectorized image and its length is the dimensionality of the problem; a point in N-D space.

Where each row in is a hyperplane that acts as a decision boundary.

Where sign of inner product tells the side of the hyperplane.

Limitations: dataset not linearly separable

**Fully Connected Network**

3-layer MLP (multilayer perceptron): where nonlinearity occurs as activation functions between layers, e.g., ReLU(x) = max(0, x).

Limitations: destroys special relationship, weights don’t scale.

**Convolutional Neural Network**

A diagram of a diagram of a diagram

Description automatically generated

<https://computersciencewiki.org/index.php/Convolutional_neural_networks_%28CNNs%29>

* Exploits spatial structure, scales to varying input sizes, good performance.

[TODO]

**RESCURSIVE MODEL**

**Sequence Modeling**

* Limitations of FFNs, CNNs:
  + requires fixed input sizes
  + input often treated as orderless (lacks temporal, sequential modeling)
* Sequential modelling allows:
  + Ordered inputs/outputs of different lengths.
* Types of sequential models:
  + One-to-one: vanilla NN
  + One-to-many: image captioning (image 🡪 sequence of words).
  + Many-to-one: action prediction (sequence of images 🡪 action class).
  + Many-to-many: video captioning (sequence of images 🡪 sequence of words).

**Recurrent Neural Networks (RNNs)**

[TODO: 19]

**VISION TRANSFORMERS**

**Vision Transformer (ViT): Object Classification**

* Self-attention is O(n^2) thus token/pixel is too many tokens, pick patch size large enough to fit into GPU memory e.g., 16x16 patch size (does linear projection over patch).
* Image 🡪 patches 🡪 flatten into vectors 🡪 linear projection 🡪 transformer
* Problem: transformers do not remember order. Solution: add sinusoids with different frequencies to the linear projections.
* Add a CLS token, learnable class token which is used for the final classification.

**DETR (2020): Object Detection Transformer Model**

* Image 🡪 CNN 🡪 set of image features + positional encoding 🡪 transformer encoder-decoder 🡪 set of box predictions.

**Swin Transformer (hierarchical vision transformer using shifted windows)**

* Shrink by 4 times in each dimension. HxWx3 🡪 H/4 x W/4 x C
* Problem with W-MSA (window based multi-head self-attention): Each window passed to separate MSA block and each MSA only observes part of actual object.
* Solution follow up with SW-MSA (shifted window based multi-head self-attention), but problem: empty spaces in windows and can still have partial object in image.

**CLIP (Contrastive Language Image Pretraining): adding language supervision**

* Start with image with caption, then encode text into a vector T, image into a vector I. Want Ti to be very similar to Ii. Maximize cosine similarity and minimize similarity with other images.

**Masked Autoencoder (MAE):**

* Mask most patches, pass unmasked patches 🡪 linear projection 🡪 individual vectors 🡪 encoder, use output to predict masked output 🡪 unshuffled 🡪 decoder 🡪 target.
* During training fine-tune encoder or use linear probing.

**DINO: self-DIstillation with NO labels.**

* Works by knowledge distillation.
* Take two patches of image. First patch is larger patch (teacher) second patch is smaller (student). Make student prediction more similar to teacher.

[TODO: refine]