

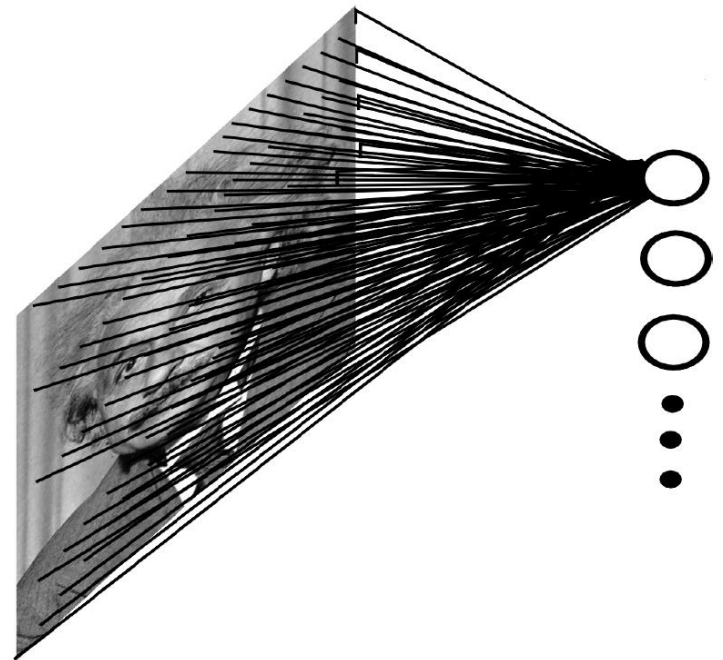
CONVOLUTIONAL NEURAL NETWORKS (CNNs)

Motivation & basic operations

MOTIVATION

Fully connected neural network

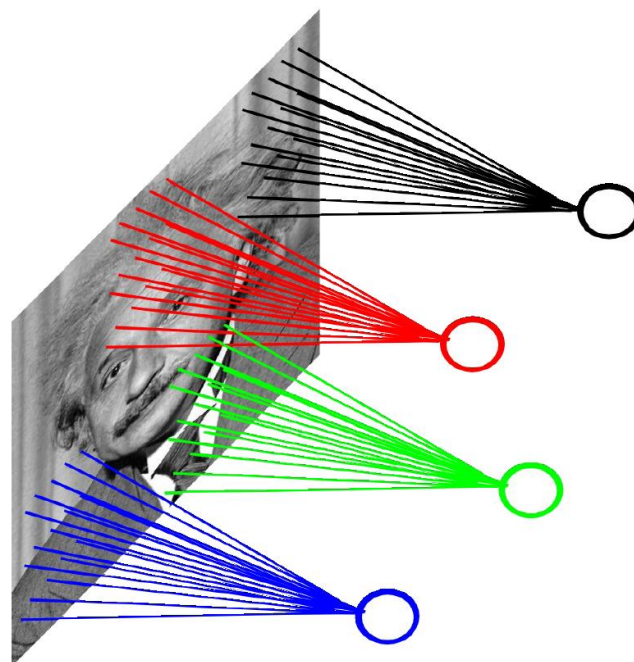
- Example
 - 1000x1000 image
 - 1M hidden units
 - 10^{12} ($= 10^6 \times 10^6$) parameters!
 - Multi-layers?
- Let's encode their locality
 - Spatial correlation is local



Locally connected neural net

- Example
 - 1000x1000 image
 - 1M hidden units
 - Filter size: 10x10
- $10^8 (= 10^6 \times 10 \times 10)$ parameters!

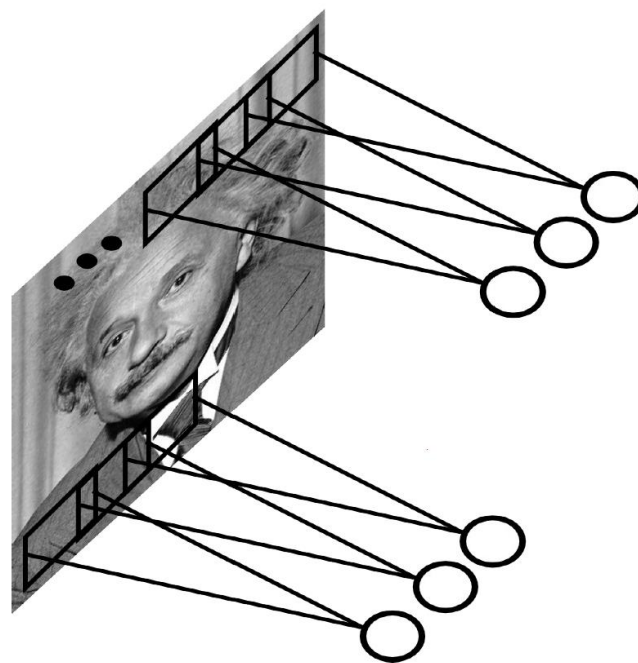
- Let's encode the invariance
 - Statistics is similar at different locations



Convolution neural networks

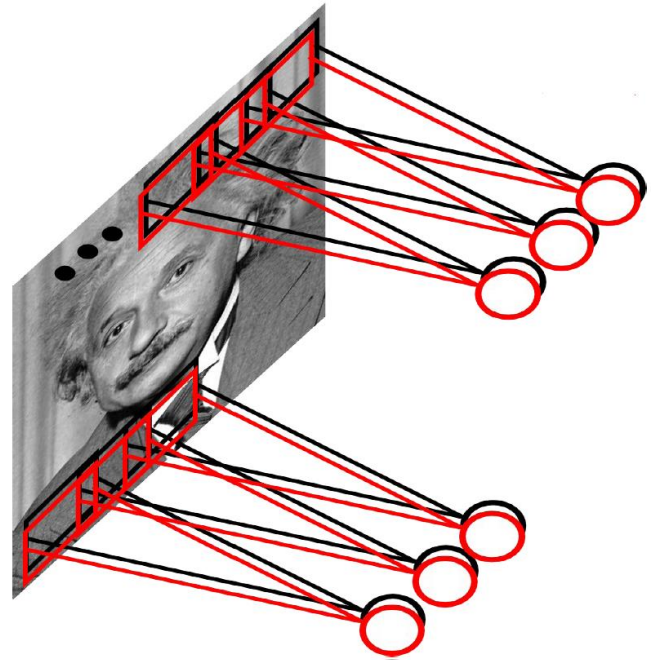
- Share the same parameters across different locations
 - Convolution with learned kernels
 - Filter size: 10×10

→ 10^2 parameters



Convolution neural networks

- Learn multiple filters
 - 1000x1000 image
 - 100 Filters
 - Filter size: 10x10
- 10,000 parameters

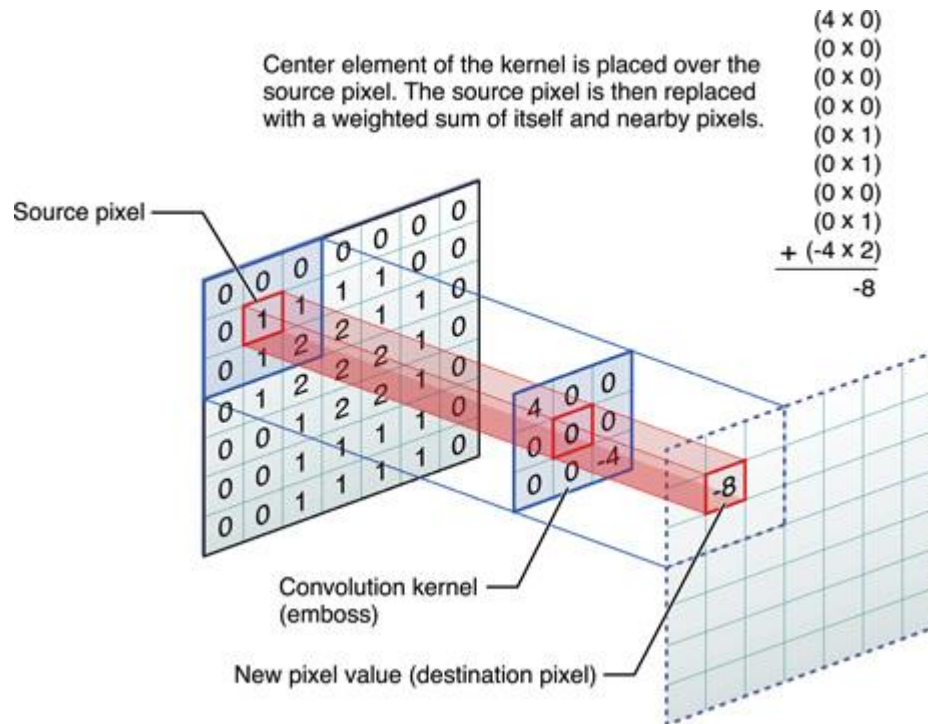


Convolution neural networks

- We can design neural networks that are specifically adapted for image-related problems
 - Must deal with very high-dimensional inputs
 - Can exploit the 2D topology of pixels
 - Can build in invariance to certain variations we can expect
 - Translations, etc
- Ideas
 - Local connectivity
 - Parameter sharing

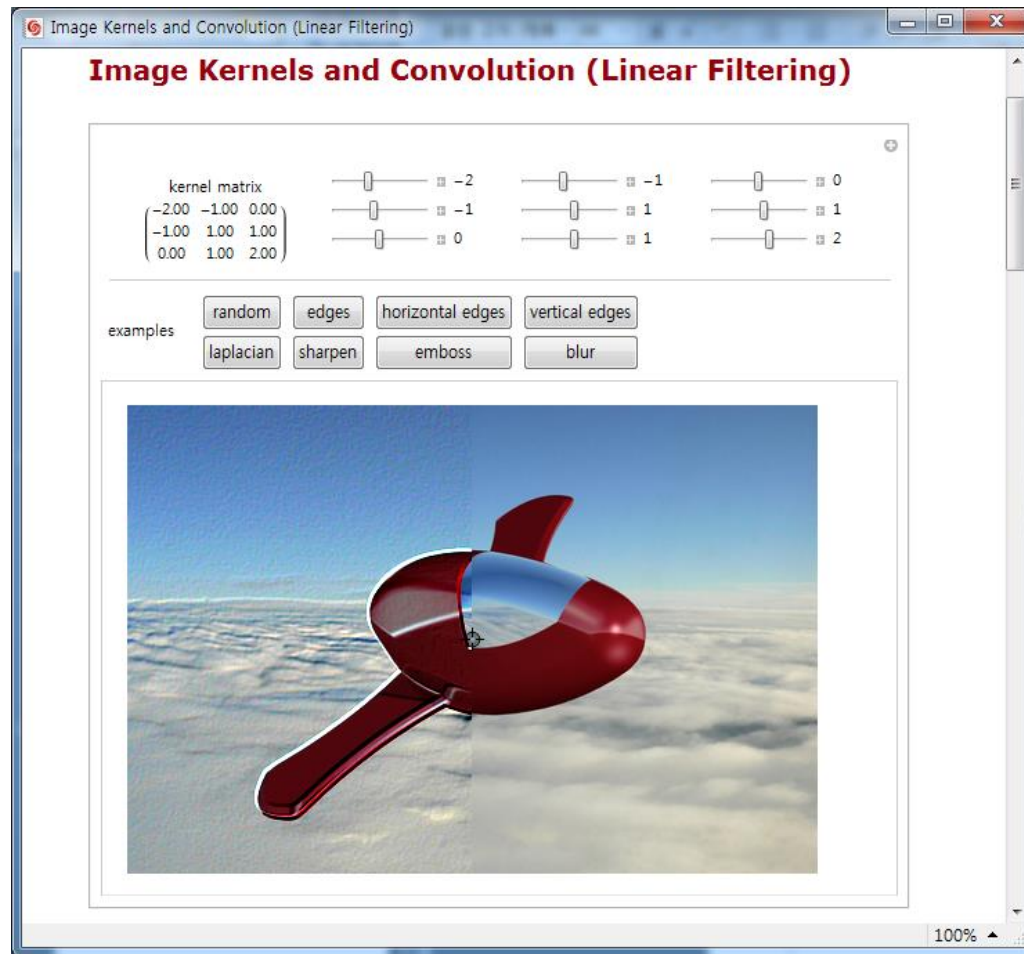
CONVOLUTION (IMAGE PROCESSING)

Convolution



from: <https://developer.apple.com/library/ios/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html>

Linear filter



Linear filter (Gaussian)

Gaussian Filtering for Blurring


Wolfram Demonstrations Project demonstrations.wolfram.com

Gaussian Filtering for Blurring

radius 5

deviation 3

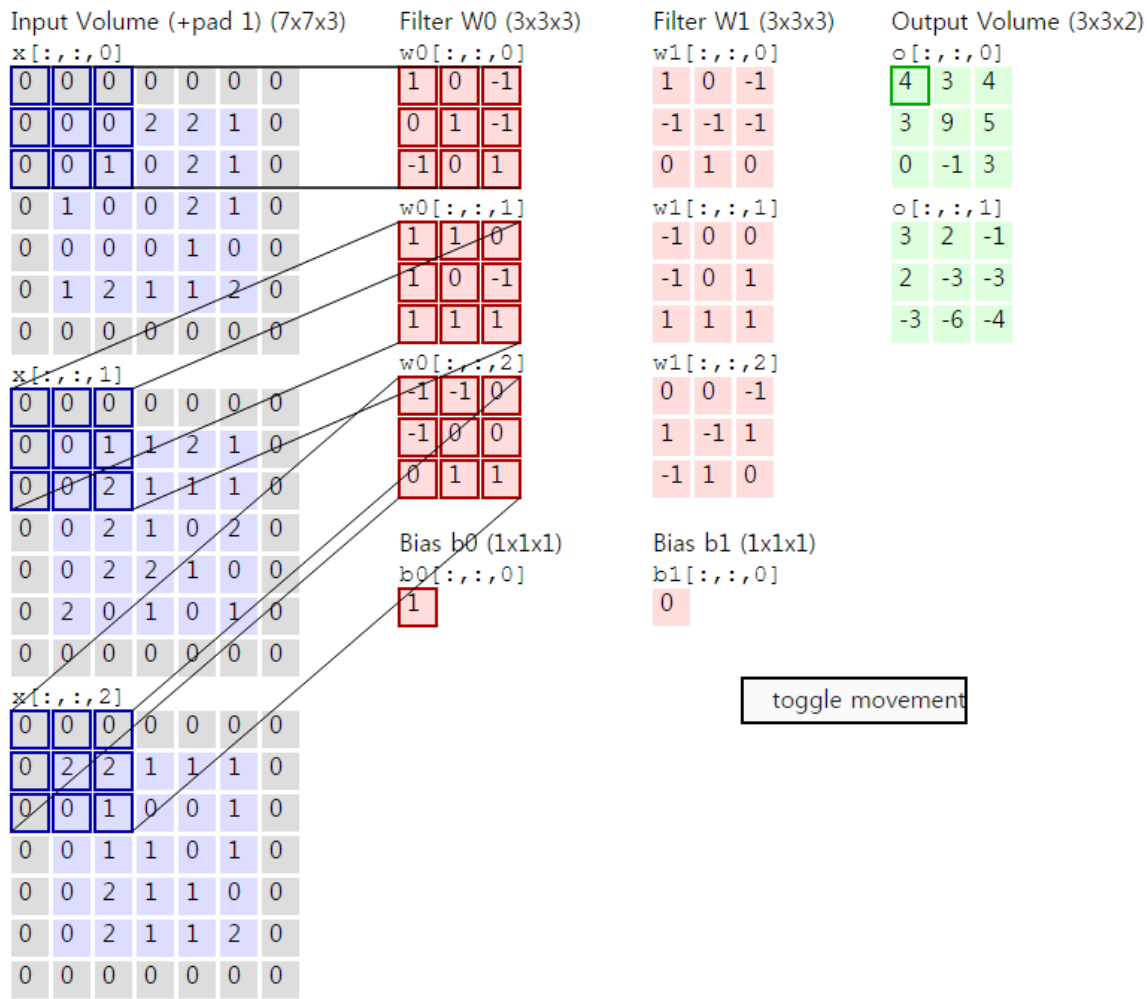
view result only thumbnail of original image side-by-side comparison

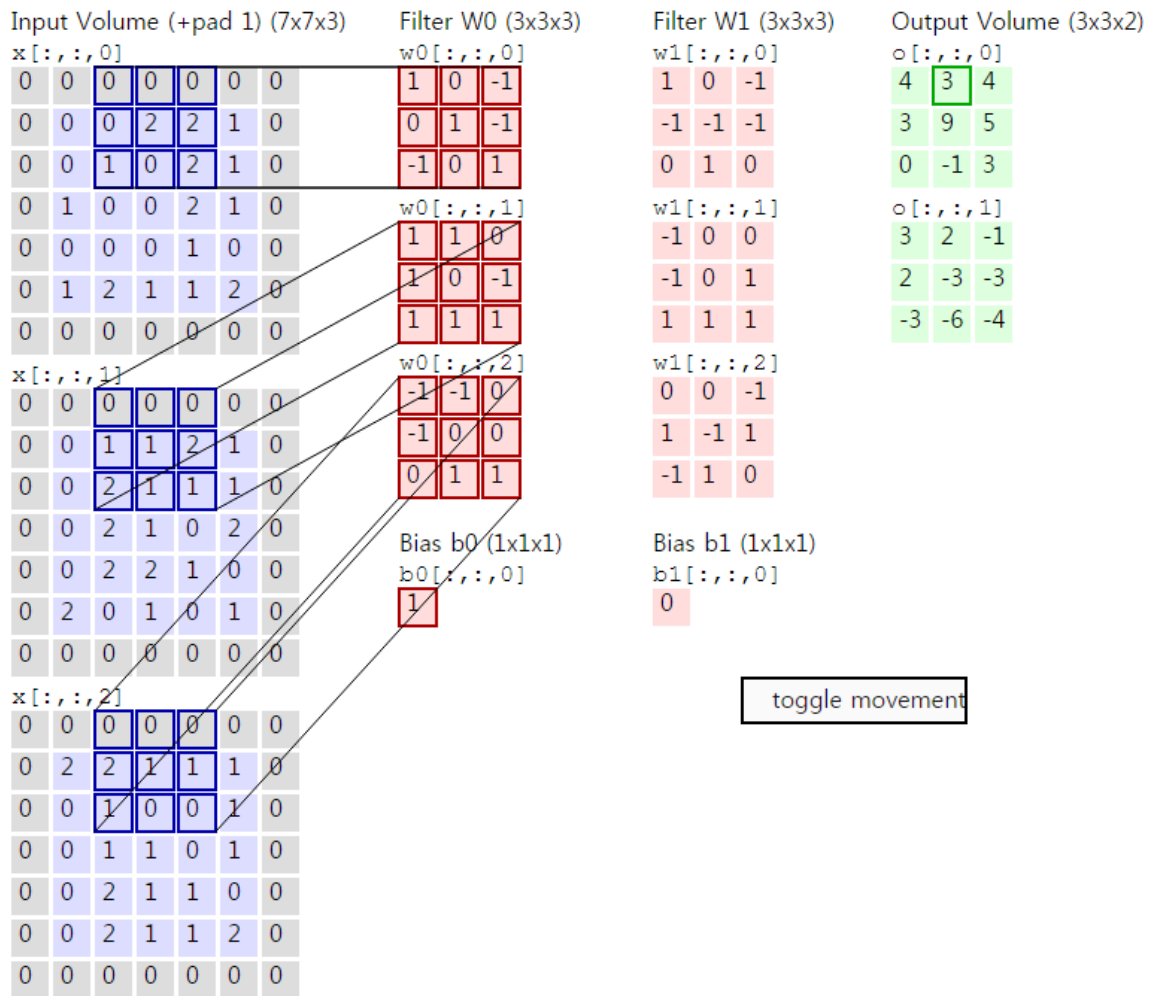


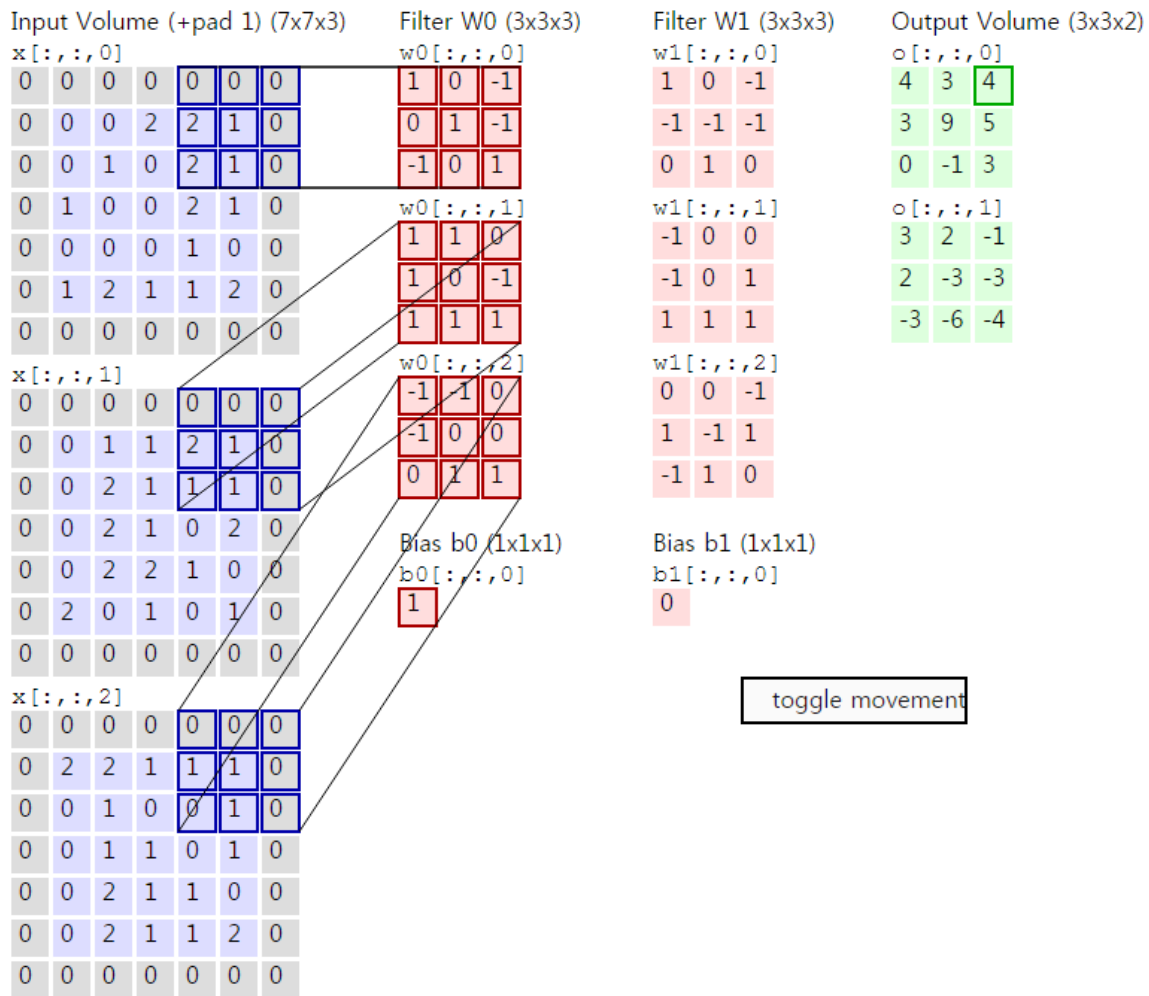
In this Demonstration, the image is blurred using a Gaussian function. Gaussian filters are widely used to reduce the effect of noise and sharp details in the image. They are often used as a preprocessing stage in many algorithms in order to enhance the quality of images. Mathematically, when a Gaussian filter is applied to

100%

CONVOLUTION (IN CNN)







Input Volume (+pad 1) (7x7x3)

```
x[:, :, 0]
0 0 0 0 0 0 0
0 0 0 2 2 1 0
0 0 1 0 2 1 0
0 1 0 0 2 1 0
0 0 0 0 1 0 0
0 1 2 1 1 2 0
0 0 0 0 0 0 0
```

```
x[:, :, 1]
0 0 0 0 0 0 0
0 0 1 1 2 1 0
0 0 2 1 1 1 0
0 0 2 1 0 2 0
0 0 2 2 1 0 0
0 2 0 1 0 1 0
0 0 0 0 0 0 0
```

```
x[:, :, 2]
0 0 0 0 0 0 0
0 2 2 1 1 1 0
0 0 1 0 0 1 0
0 0 1 1 0 1 0
0 0 2 1 1 0 0
0 0 2 1 1 2 0
0 0 0 0 0 0 0
```

Filter W0 (3x3x3)

```
w0[:, :, 0]
1 0 -1
0 1 -1
-1 0 1
```

```
w0[:, :, 1]
1 1 0
1 0 -1
1 1 1
```

```
w0[:, :, 2]
-1 -1 0
-1 0 0
0 1 1
```

Bias b0 (1x1x1)

```
b0[:, :, 0]
1
```

Filter W1 (3x3x3)

```
w1[:, :, 0]
1 0 -1
-1 -1 -1
0 1 0
```

```
w1[:, :, 1]
-1 0 0
-1 0 1
1 1 1
```

```
w1[:, :, 2]
0 0 -1
1 -1 1
-1 1 0
```

Bias b1 (1x1x1)

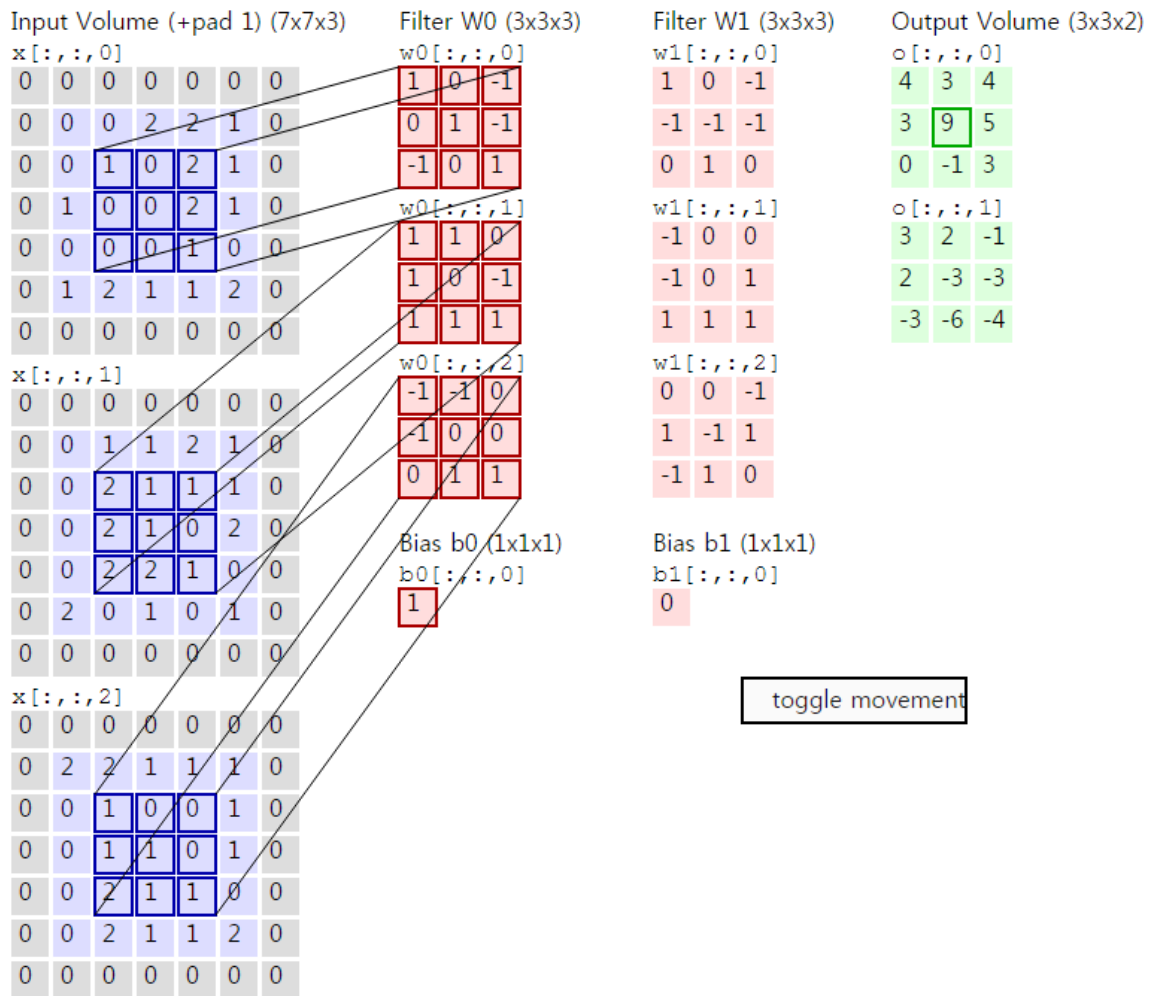
```
b1[:, :, 0]
0
```

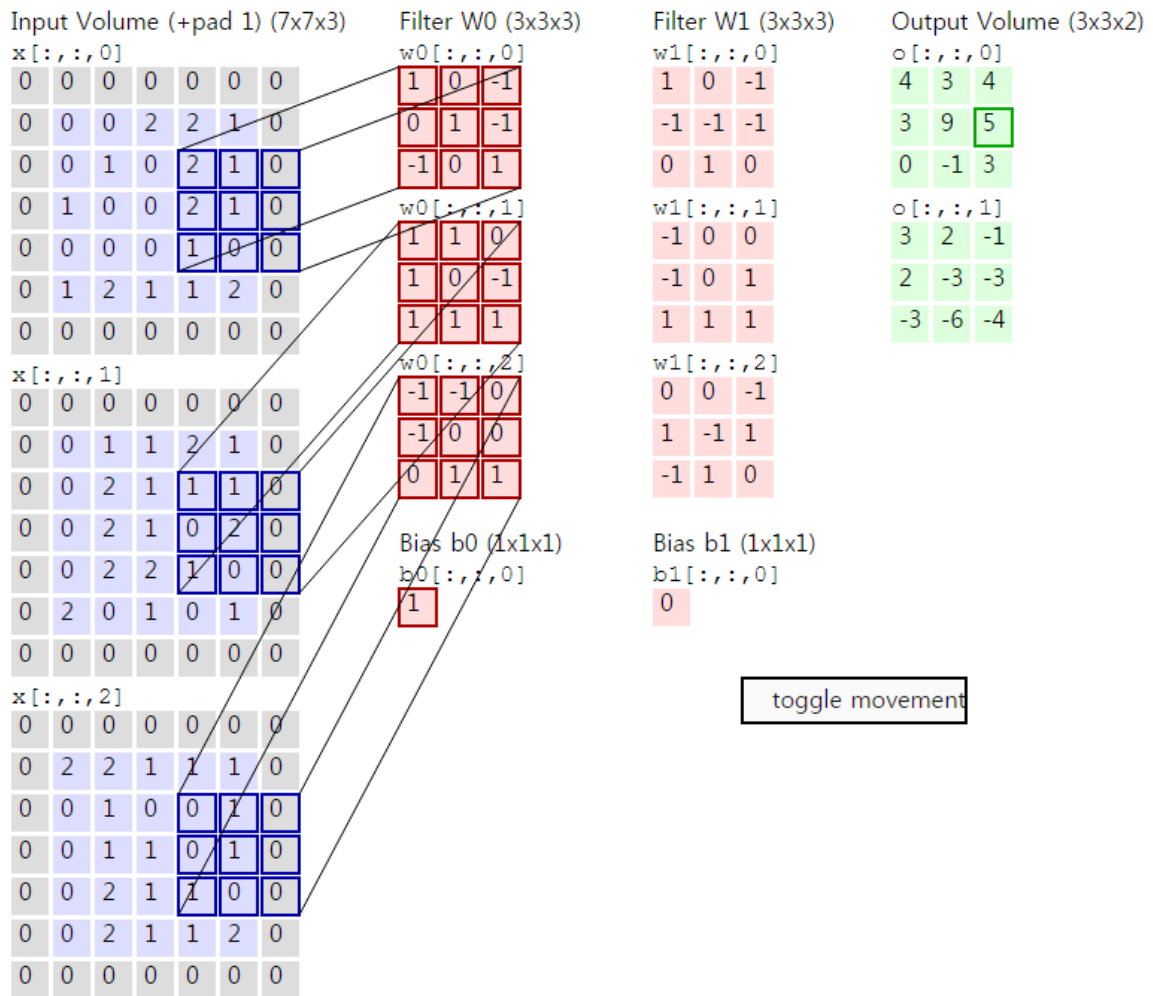
Output Volume (3x3x2)

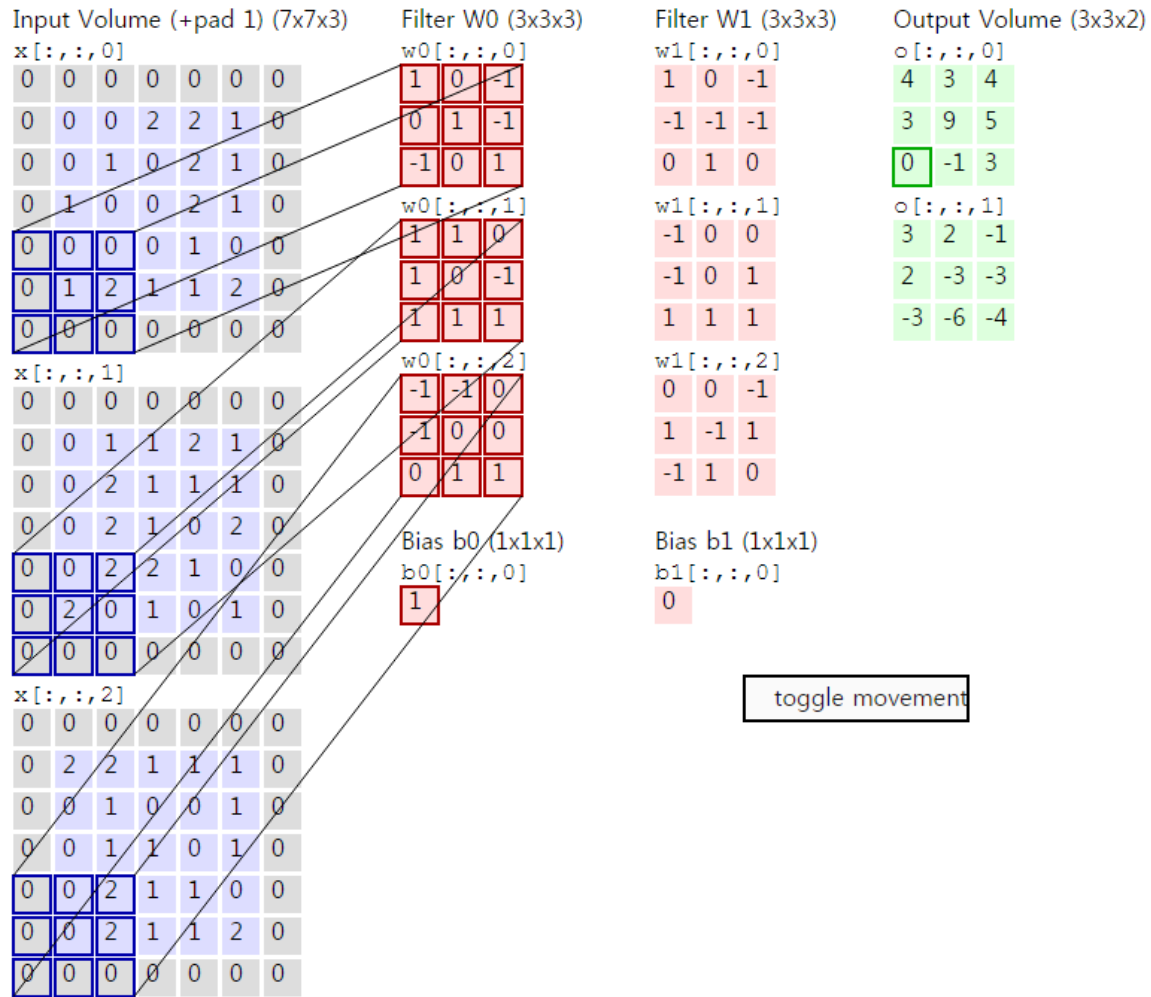
```
o[:, :, 0]
4 3 4
3 9 5
0 -1 3
```

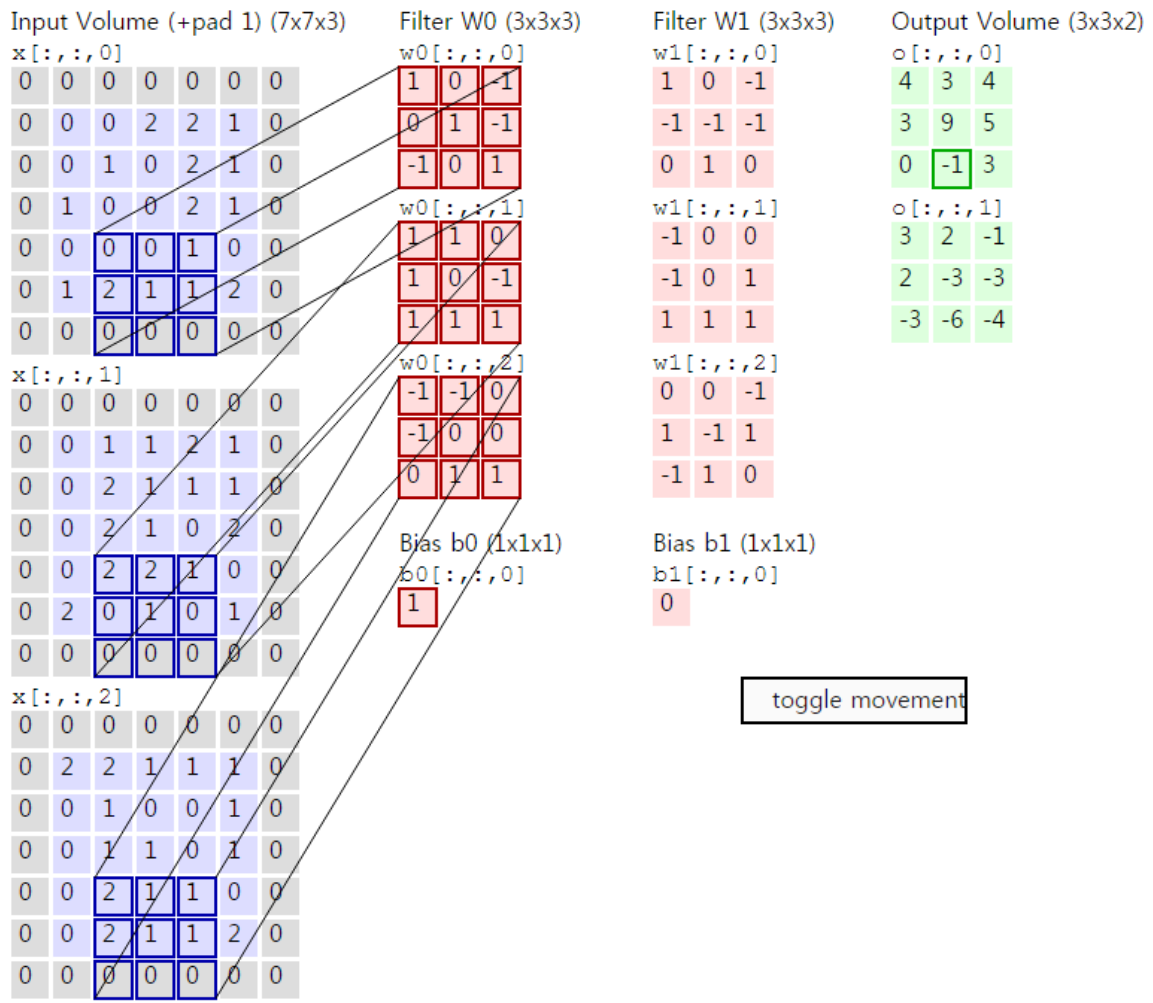
```
o[:, :, 1]
3 2 -1
2 -3 -3
-3 -6 -4
```

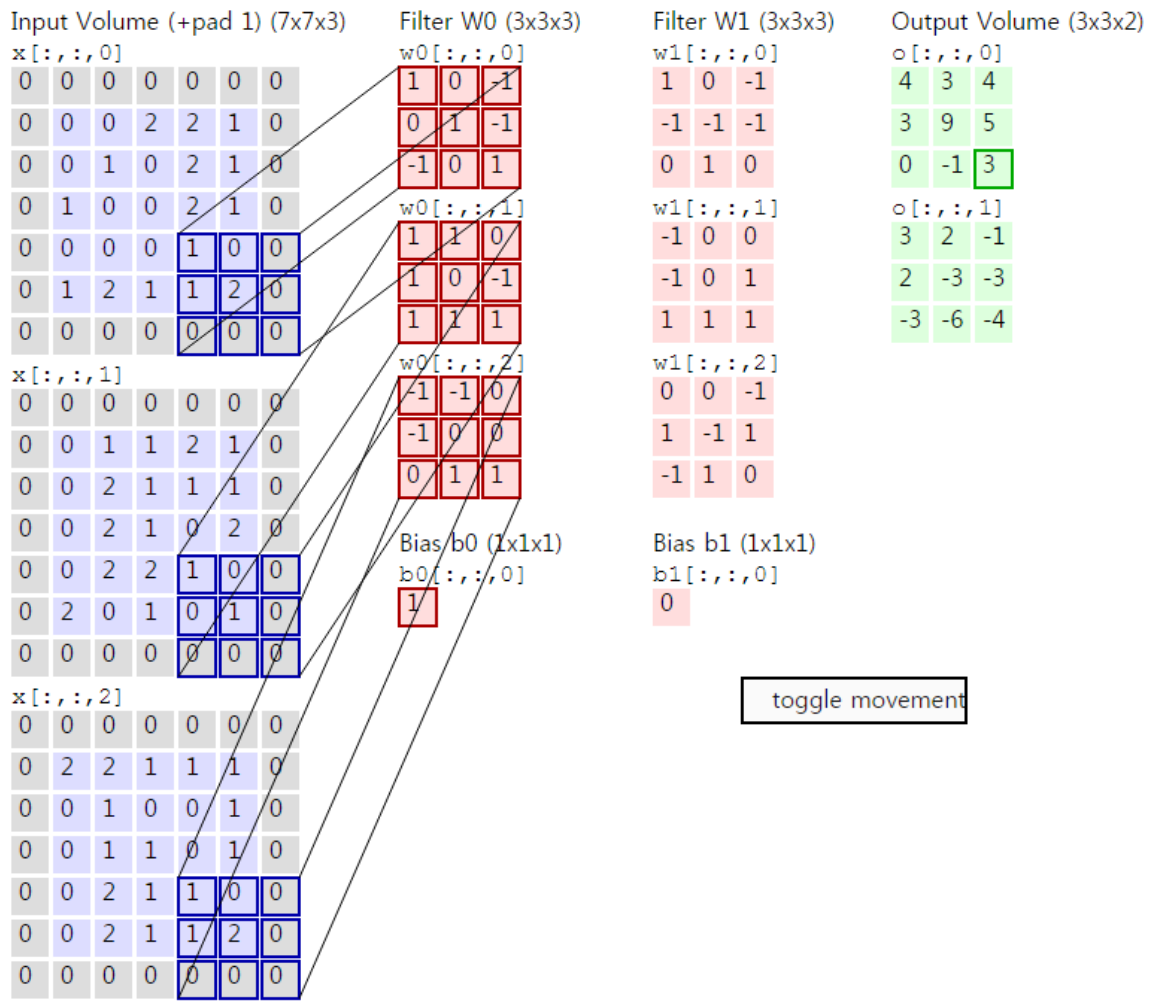
toggle movement

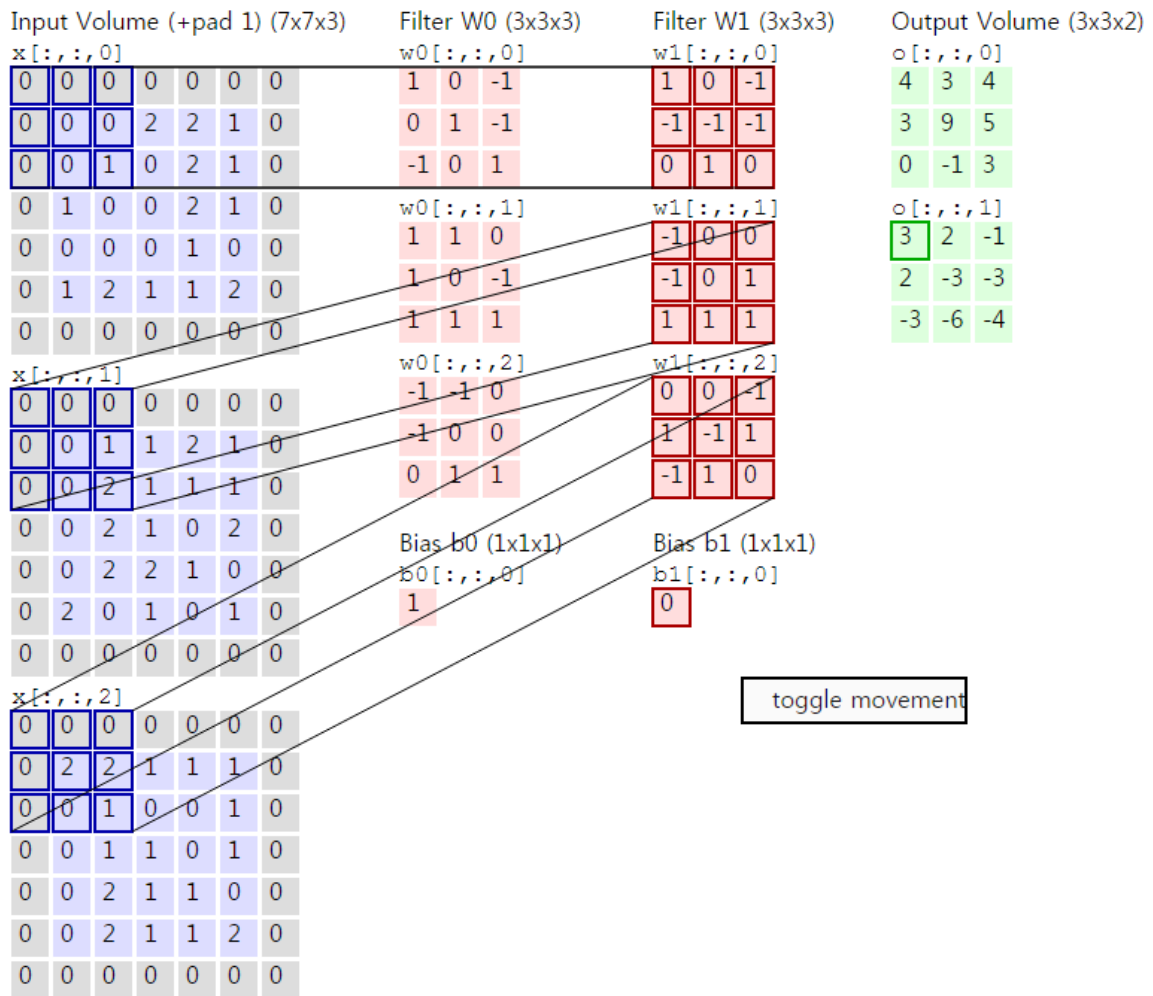












toggle movement

