

卷积核

<https://ssshooter.github.io/canvas-img-process/>

粒子效果

<https://kongchenglc.github.io/Demo/particle/index.html>

Canvas

```
<canvas id="canvas" width="400" height="300"/>

Var canvas = document.getElementById('canvas');
Var ctx = canvas.getContext('2d');
```

一个画布的创建

关于 canvas 的宽高。

Canvas 是一个画板和一张画纸，画板相当于一个容器，画图/作业是在画纸上进行的，

画板和画纸的默认宽高是 300*150，当画纸与画板宽高相等时，图像不会被拉伸，当画纸与画板宽高不一样时，图像就会被拉伸（变形）。

1. width 和 height 属性是设定画板和画纸的宽高，
如： width="300" height="300" 即画板的宽高是 300*300，画纸的宽高也是 300*300，作业的 300*300 的对角线图像就不会被拉伸

2. style 样式 里设定的是仅画板的宽高，画纸的宽高还是为默认值 300*150

```
canvas.drawImage(this.video, 0, 0, this.width, this.height);

canvas.getImageData(0, 0, this.width, this.height) → 输出一个 imageData 对象

canvas. createImageData(imageData 对象)
```

关于 imageData 对象

```
▼ ImageData {data: Uint8ClampedArray(245760), width: 320, height: 192} ⓘ
  ▶ data: Uint8ClampedArray(245760) [211, 215, 26, 0, 211, 215, 26, 0, 211, 215, 26, 0,
    height: 192,
    width: 320]
```

一个一维数组，包含以 RGBA 顺序的数据，数据使用 0 至 255（包含）的整数表示。

ImageData.data



第 i 个像素：

R — ImageData.data[4 * i + 0]
G — ImageData.data[4 * i + 1]
B — ImageData.data[4 * i + 2]
A — ImageData.data[4 * i + 3]

第 x 行 第 y 列的像素：

$i = x * \text{width} + y$
R — ImageData.data[4 * i + 0]
G — ImageData.data[4 * i + 1]
B — ImageData.data[4 * i + 2]
A — ImageData.data[4 * i + 3]

方法不允许操作非此域名外的图片资源，所以如果想本地试试文章中的例子，直接写图片路径就会报错，






关于粒子效果：

灰度公式

$$\text{Brightness} = 0.21 * r + 0.72 * g + 0.07 * b$$

卷积 + ImageData = ?

图像归根到底就是一大堆的颜色点矩阵，我们完全可以把颜色点代替上面的数字矩阵处理，不同的卷积核对图片的处理结果如下（图片来自维基百科）

Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	