# 降采样

降采样（Downsample）也称下采样（Subsample），按字面意思理解即是降低采样频率。对于一幅N\*M的图像来说，如果降采样系数为k,则降采样即是在原图中每行每列每隔k个点取一个点组成一幅图像的一个过程。

不难得出，降采样系数K值越大，则需要处理的像素点越少，运行速度越快。

利用Graphics.Blit(rt1, rt2, \_Material);这种方式rt2长宽是rt1长宽的一半，这种方法实际用的是GPU内置的双线性滤波

缩小后的图片像素更少，计算更快

# 卷积核kernel

就是后面高斯模糊中的权重矩阵，决定最后像素的混合权重，卷积核有很多种，也不一定只有9个数，这里高斯只是其中一种

# 均值模糊

基本原理：取4个方向的周边像素进行叠加，得到的结果就是模糊的

具体操作: vert种取每个顶点上下左右一定范围的像素的uv，frag直接对上面的uv采样，得到结果混合起来

Demo中使用了降采样，可以调节降采样次数，均值模糊降采样次数多时效果看起来不是很好

均值采样比较重要的是对周边点采样的半径

均值采样优点明显就是计算简单

Demo 中的 SimpleBlurPostEffect就是均值采样

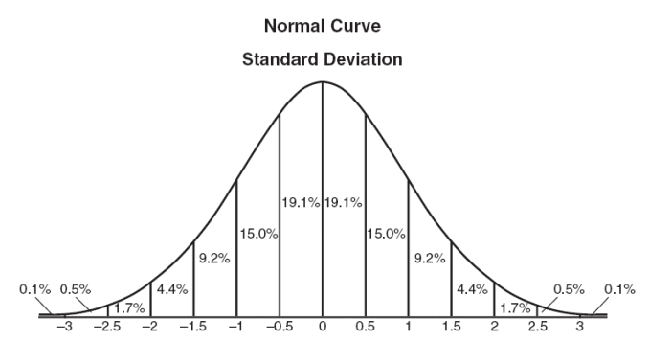
# 高斯模糊

<https://blog.csdn.net/qinghuaci666/article/details/81870277>

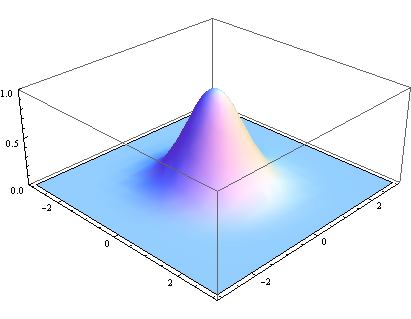
基本原理:取8个方向的周边像素进行叠加，8个方向的像素对最终结果影响的权重不同

高斯模糊原理解释

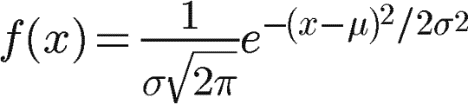
正态分布



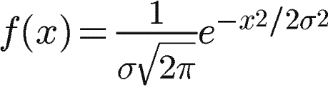
二维正态分布



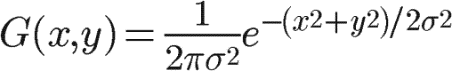
正态分布的密度函数叫高斯函数(Gaussian function)

一维

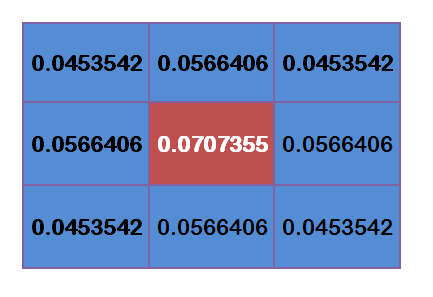
其中，μ是x的均值，σ是x的方差。因为计算平均值的时候，中心点就是原点，所以μ等于0。



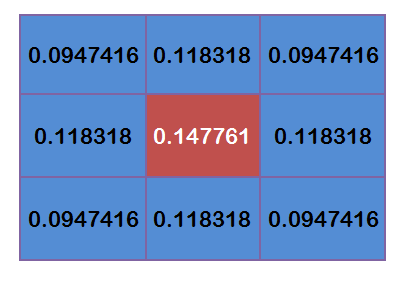
根据1维高斯，推到2维

二维，用这个函数计算周边点权重

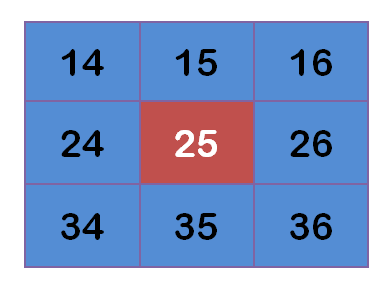
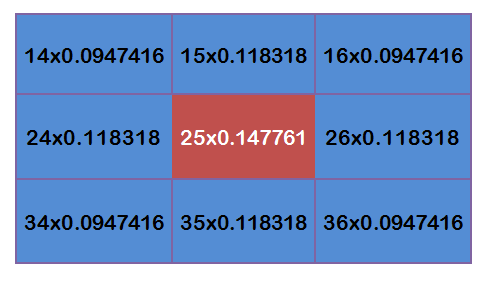
为了计算权重矩阵，需要设定σ的值。假定σ=1.5，则模糊半径为1的权重矩阵如下：



这9个点的权重总和等于0.4787147，如果只计算这9个点的加权平均，还必须让它们的权重之和等于1，因此上面9个值还要分别除以0.4787147，得到最终的权重矩阵。



下面举例

原始像素与权重矩阵相乘，结果直接叠加就好

Demo中GaussBlur 使用的权重矩阵跟原理中的有些不同，但原理基本一致