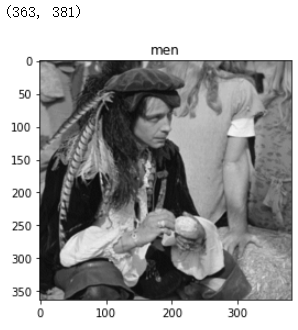
**《图像分析与理解》课程——模糊方法实验**

212138 卓旭

**Part 1 – 模糊边界提取**

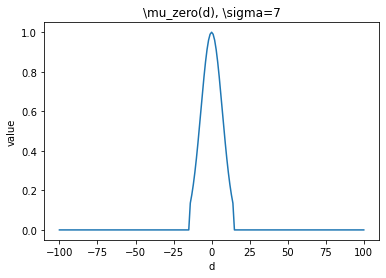
**1. 读取men.txt**

按行扫描读取即可，尺寸正确，显示如下：



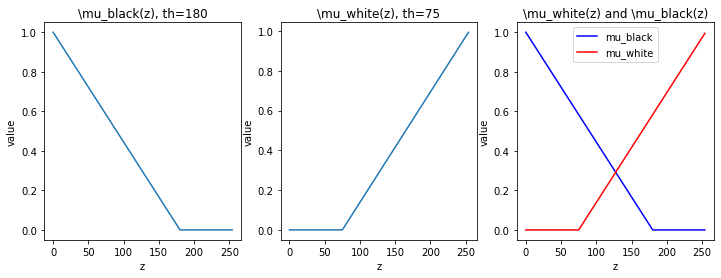
**2. 的定义**

按照所给公式计算即可，得到曲线：



**3. 、的定义**

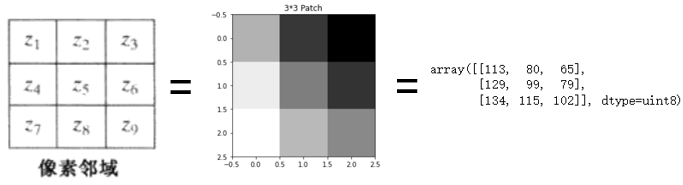
同样按照所给公式计算即可，得到曲线：



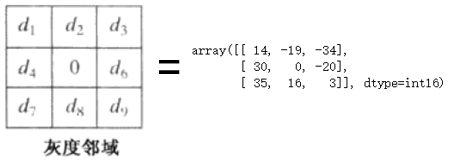
第三子图是将两条隶属度曲线画在一起的结果，可以看到中间部分形成了类似于软阈值的效果。

**4. 模糊集操作**

a) 考察第一个3×3邻域如下：

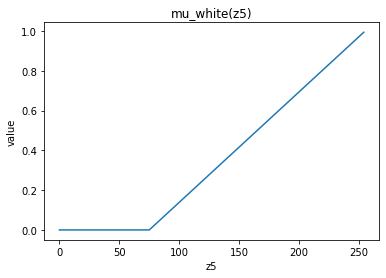


b) 考察像素差如下：

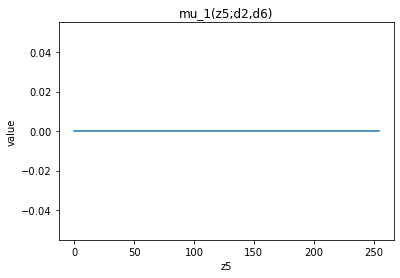


c) 程序求得、，验证正确，因为-19和-20的绝对值均超出2*σ* = 14。

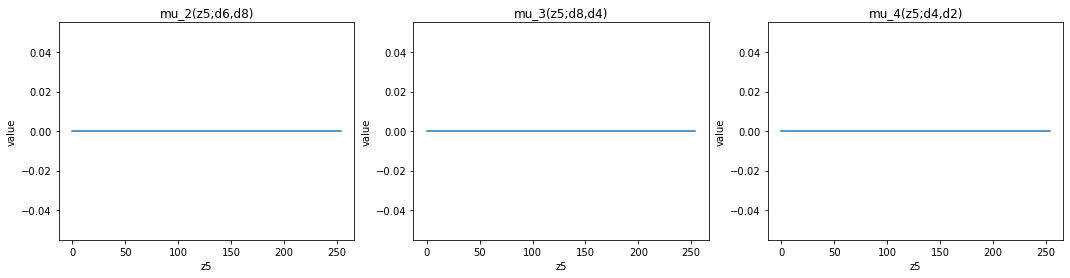
进一步，计算，得到变化曲线如下：



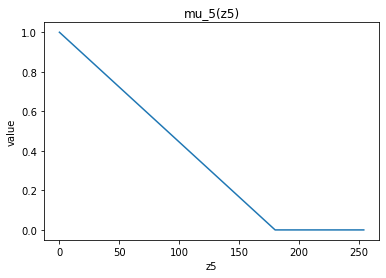
进一步求得到曲线如下：



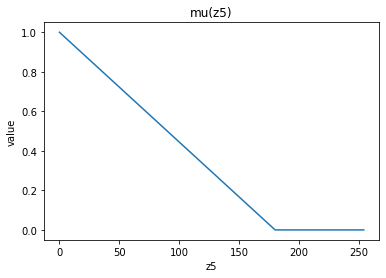
d) 同理得到：



e) 第五条规则曲线如下，与前文3.步骤的结果相同



f) 总的隶属度函数曲线如下：



g) 使用重心规则来defuzzify，求后得到59.67，考虑到隶属度类似于概率分布的含义，使用四舍五入来取整更合适，得到60。普遍地，最终结果应四舍五入后裁剪到[0, 255]并转回uint8型。

**5. 全图提取边界**

将上述步骤整理为一个函数，删去无用的中间输出部分，从而可对每个像素（每个邻域）进行处理，函数在helper.py的fuzzyEdgeExtract3x3Py。

考虑到并行计算的可能性，使用CUDA编写了GPU并行加速版的相应函数，在cuFuzzy.cu，调用接口在cuFuzzy.py的fuzzyEdgeExtract3x3。未加速版本运行时间3分20秒，CUDA版本运行时间0.03秒，加速约6666倍。

最终的边界提取结果如下：

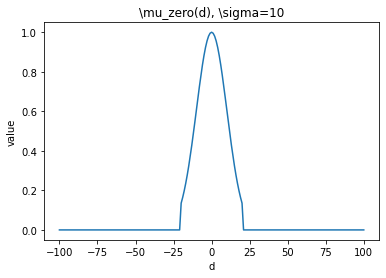


**6. 调整*σ* = 10后重复**

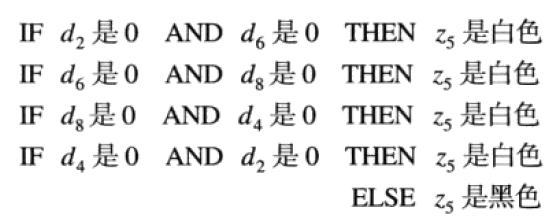
前文使用*σ* = 7，本节调整*σ* = 10，其余步骤保持一致，得到边界提取结果如下：



简要解释如下。此时关于“是不是0”的隶属度函数曲线变为：



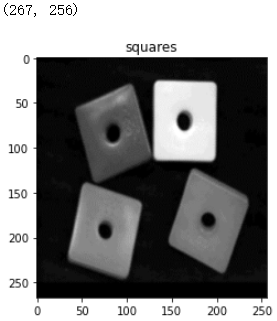
相比前文的曲线，钟形放宽了。这意味着专家知识中，“是0 AND 是0”的四条规则更容易满足了，那么是白色的情况就变多了，反映到提取的边界中时就是黑色像素变少了。更进一步地解释，是“像素属于边界”的条件变得更严格了，即需要更大的差异才能满足2σ的限定，所以提取到的边界更少了，留下的更多属于“很明显、很强”的边界。



**Part 2 – 模糊阈值分割**

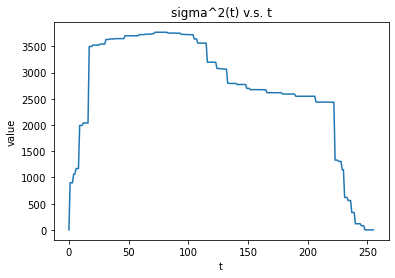
**1. 读取squares.txt**

按行扫描读取即可，尺寸正确，显示如下：

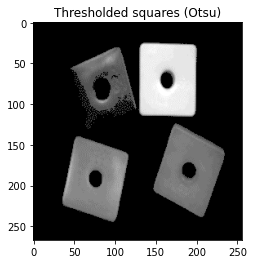


**2. 直接Otsu阈值分割**

按描述步骤实现Otsu阈值分割算法，求取的函数为calcOtsuSigma2t，试验0~255阈值得到变化曲线如下：



取argmax位置对应灰度作为阈值，为78。将squares小于阈值的像素全部置0（黑），得到分割后结果图如下：

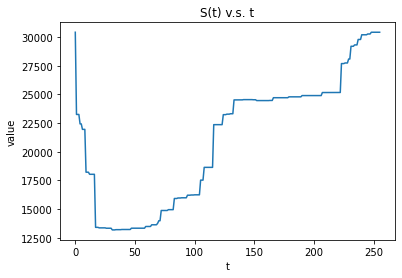


可以看到左上角的方块分割效果不好，出现残缺。

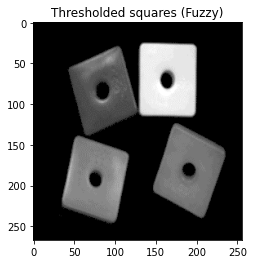
**3. 模糊阈值分割**

同样按描述步骤实现模糊阈值分割算法，求取全体像素熵和的函数为calcFuzzyEntropy。注意到熵的公式可能在数值上出现导致计算报错，由于，只需让结果受前项支配，可在的操作数中加一小量解决问题。

得到熵和随阈值变化曲线如下：



按最小熵原则取argmin对应t为31。同样将squares小于阈值的像素全部置0（黑），得到分割后结果图如下：



可以看到左上角的方块分割效果变好了，没有残缺。