人工智能导论实验

SAD实现立体视觉匹配

算法步骤：

输入：两幅图像，一幅Left-Image，一幅Right-Image且两幅图像已经校正实现行对准

对左图，依次扫描，选定一个锚点：

（1）构造一个小窗口,类似于卷积核；

（2）用窗口覆盖左边的图像,选择出窗口覆盖区域内的所有像素点；

（3）同样用窗口覆盖右边的图像并选择出覆盖区域的像素点；

（4）左边覆盖区域减去右边覆盖区域，并求出所有像素点灰度差的绝对值之和；

（5）移动右边图像的窗口，重复（3）-（4）的处理（这里有个搜索范围,超过这个范围跳出）；

（6）找到这个范围内SAD值最小的窗口,即找到了左图锚点的最佳匹配的像素块。

代码：

winsize = 40;

limit = 70;

stride = 10;

skip = 3;

I = imread('/Users/janechen/Desktop/课程/人工智能/大作业/Baby1/view5.tif');

J = imread('/Users/janechen/Desktop/课程/人工智能/大作业/Baby1/view1.tif');

[height, width, chanel] = size(I);

disp = ones(height,width);

hstride = floor((height-winsize)/stride);

wstride = floor((width-winsize-limit\*skip)/stride);

for i = 0:hstride

for j = 0:wstride

result = [];

for k= 0:limit

I0 = I(i\*stride+1:(i\*stride+winsize),j\*stride+1:(j\*stride+winsize),:);

J0 = J(i\*stride+1:(i\*stride+winsize),(j\*stride+k\*skip+1):(j\*stride+winsize+k\*skip),:);

a=imabsdiff(I0,J0); //图像每个像素值相减取绝对值，求和

a=sum(a(:));

result = [result,a];

end

result0 = min(result);

index = find(result==result0);

index = min(index);

disp(i\*stride+1:(i\*stride+stride),j\*stride+1:(j\*stride+stride)) = index\*ones(stride,stride);

end

end

imshow(disp,[])

用了函数imabsdiff

等价于：

difference = img1 - img2;

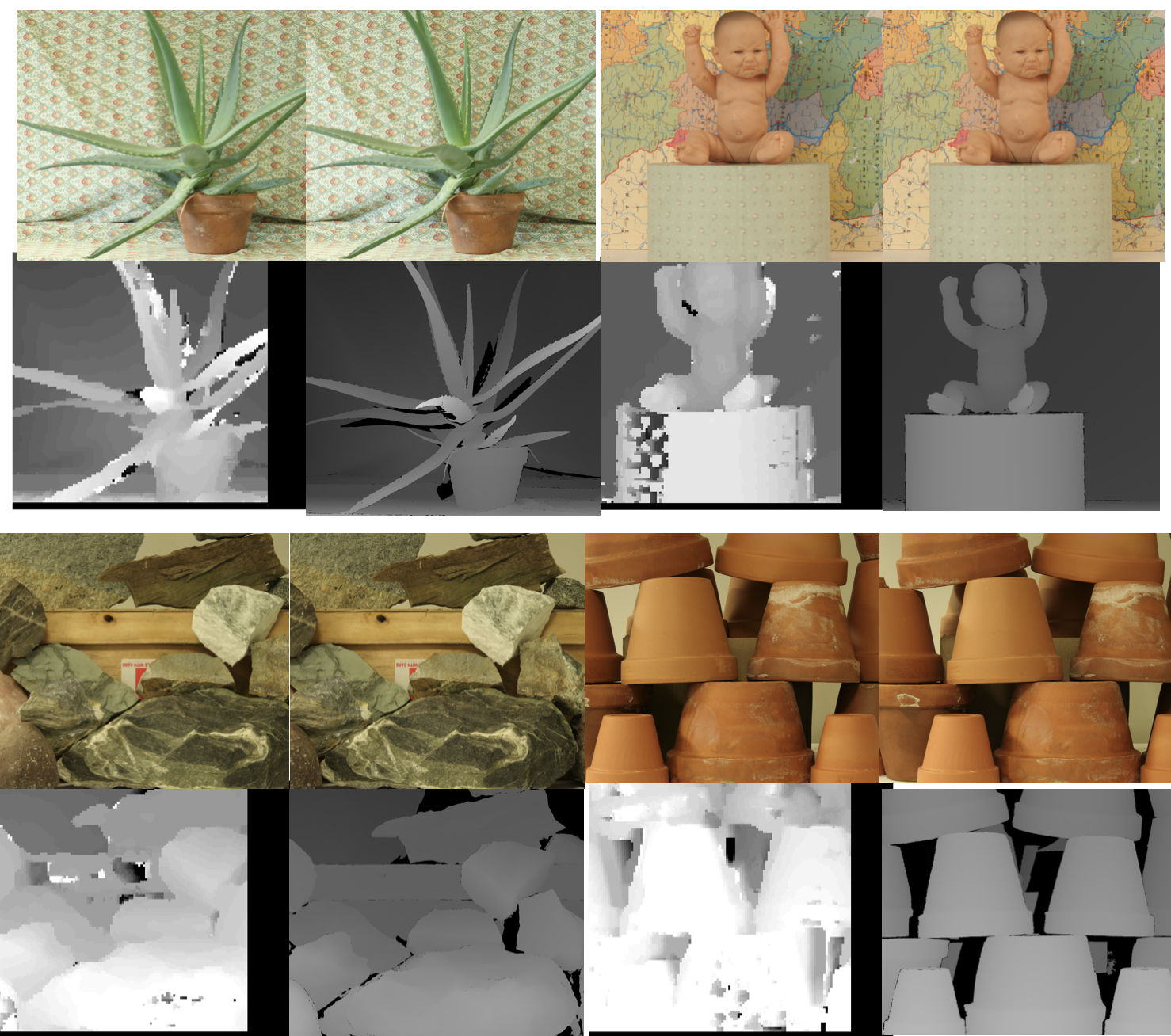
absolute = abs(difference);

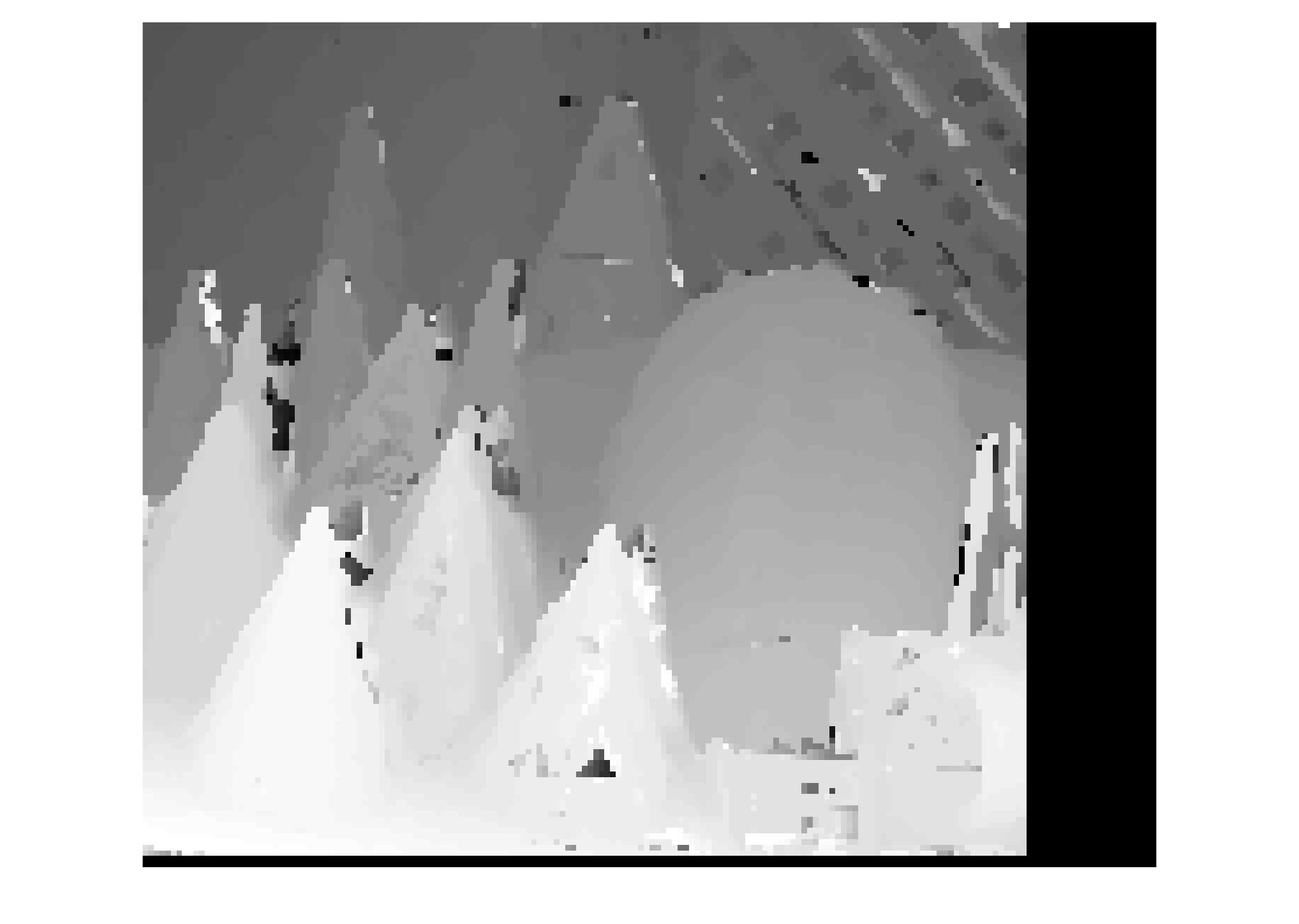
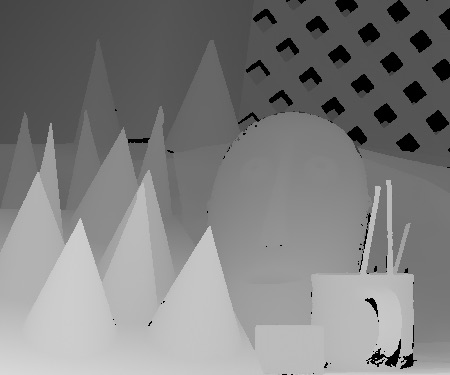
result = sum(absolute(:));

但是由于是图像，相减之后负值都变成0了，用函数可以解决这个问题。

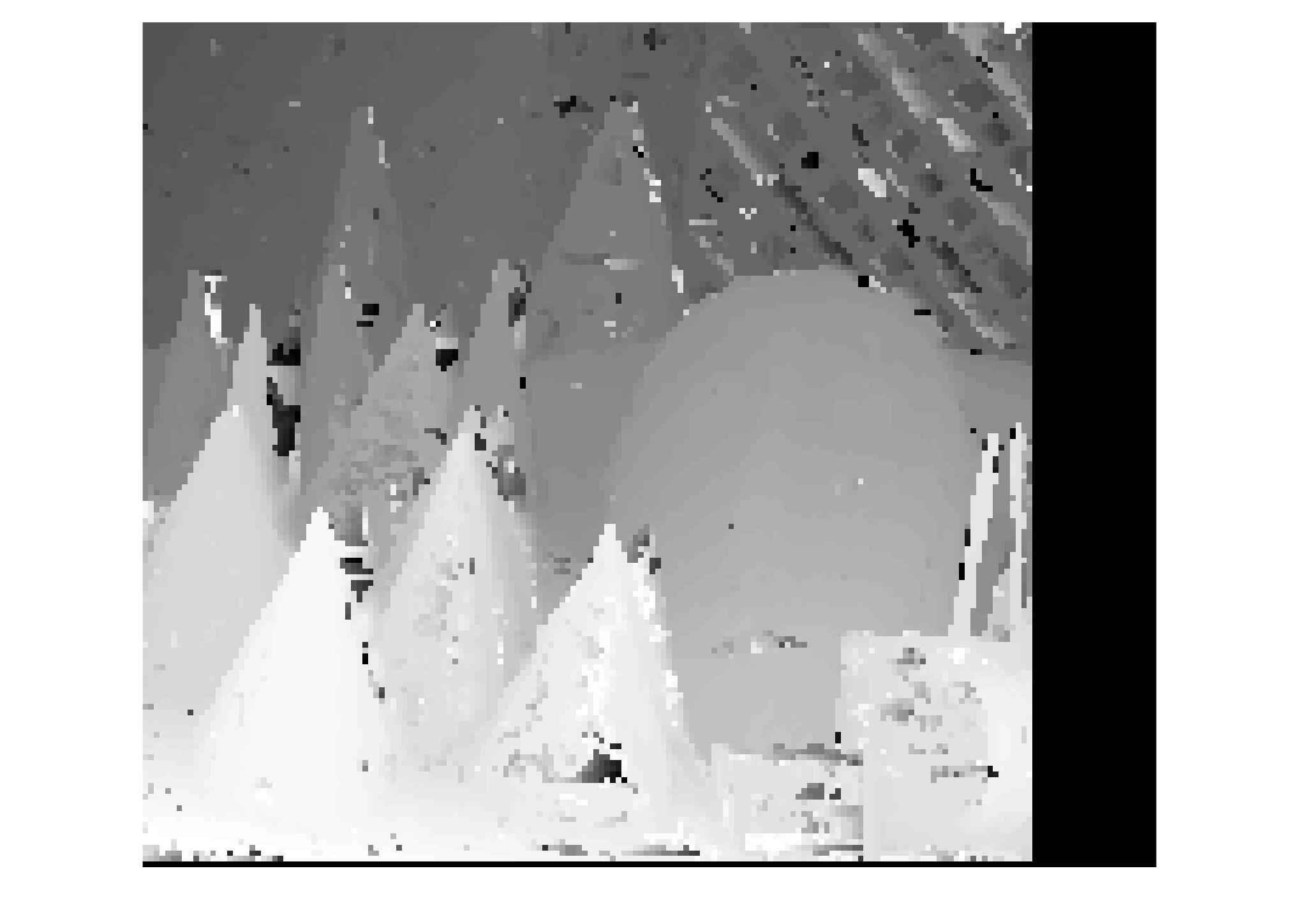
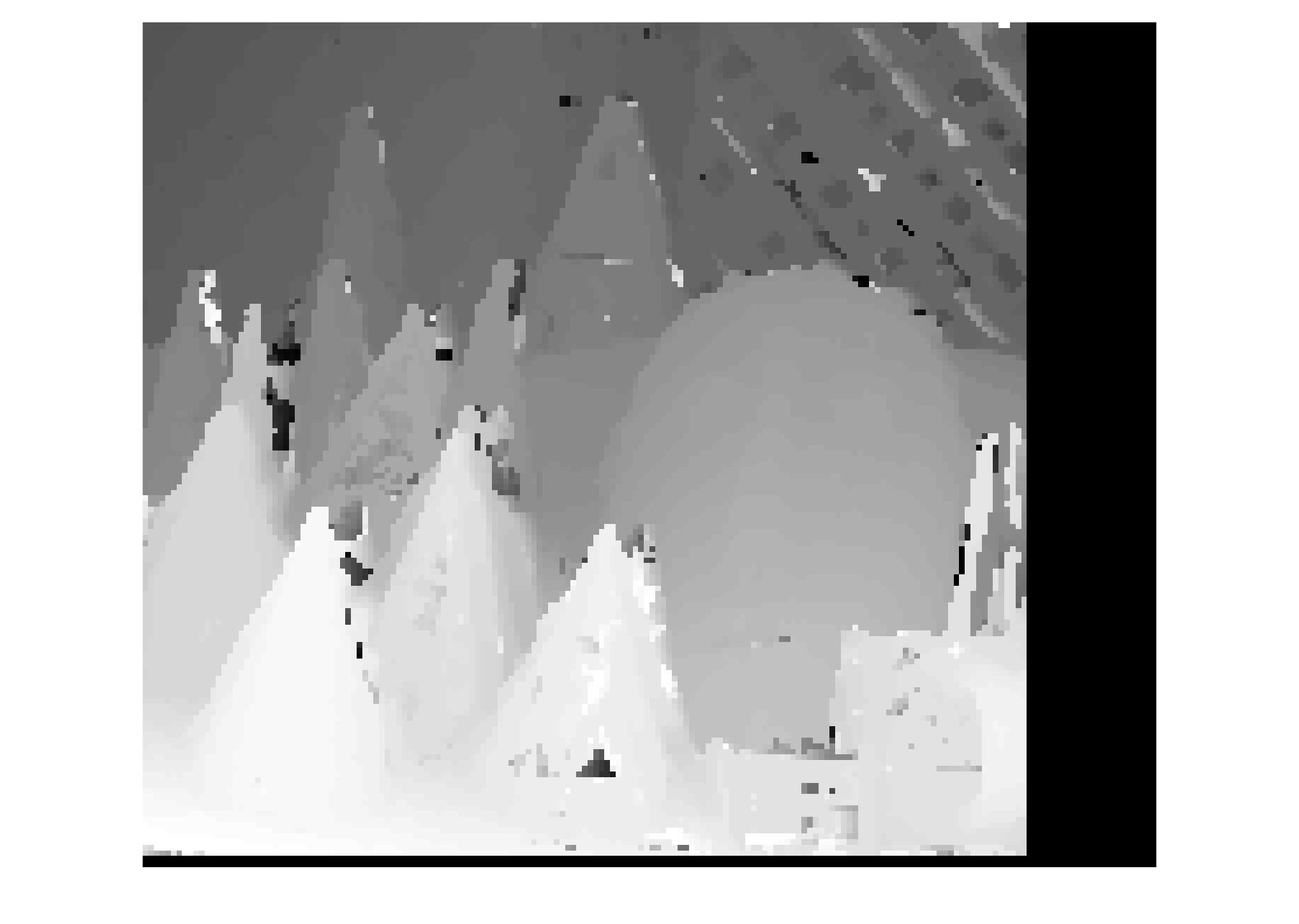
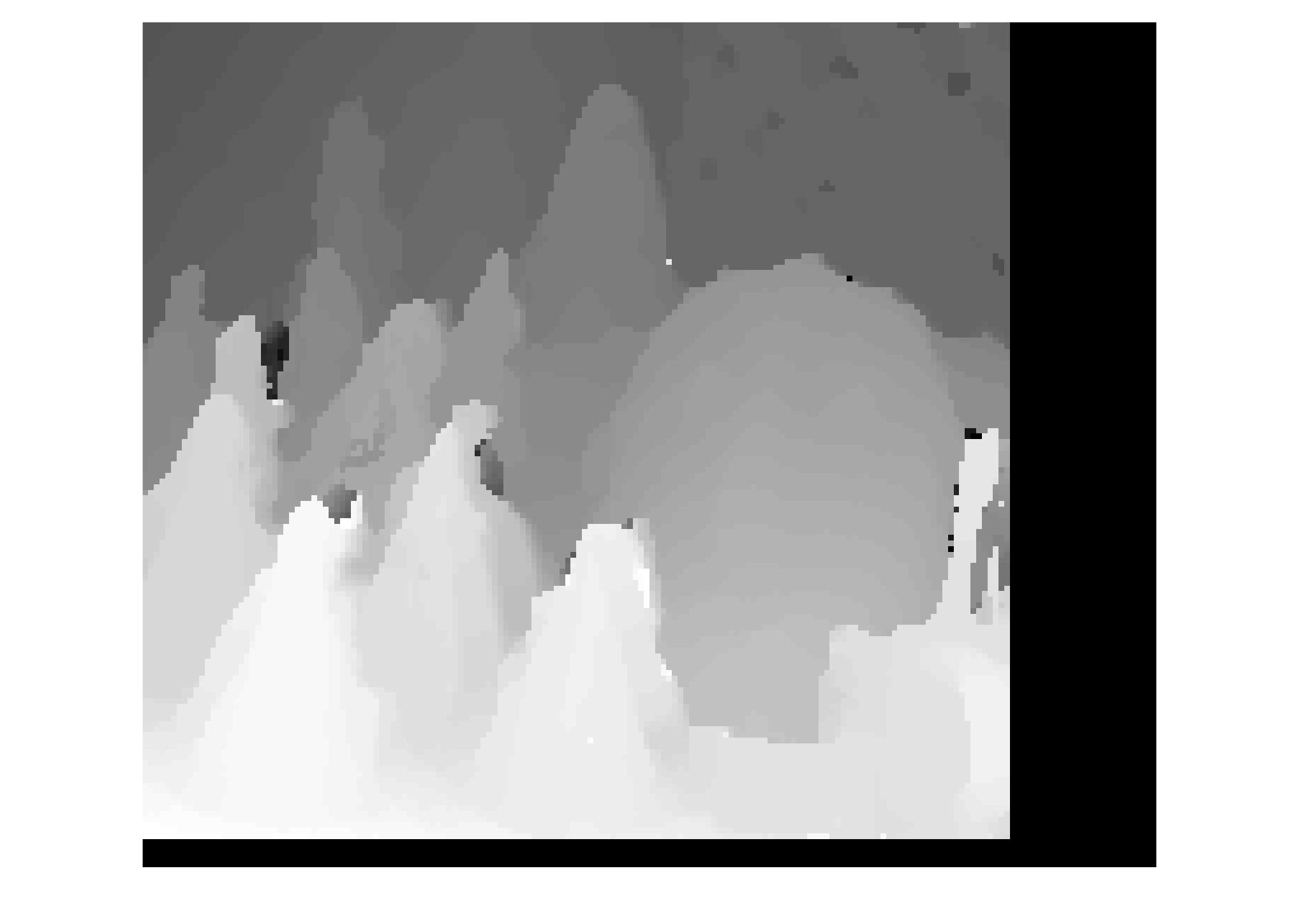
实验结果：

在2006数据集上的结果（winsize = 40; limit = 70; stride = 10; skip = 3; fullsize=1240\*1110; 运行时间约35s）

图像上的黑边是因为有一部分右侧的图像只出现在右图中，这里肯定是匹配不上的，就不做匹配。

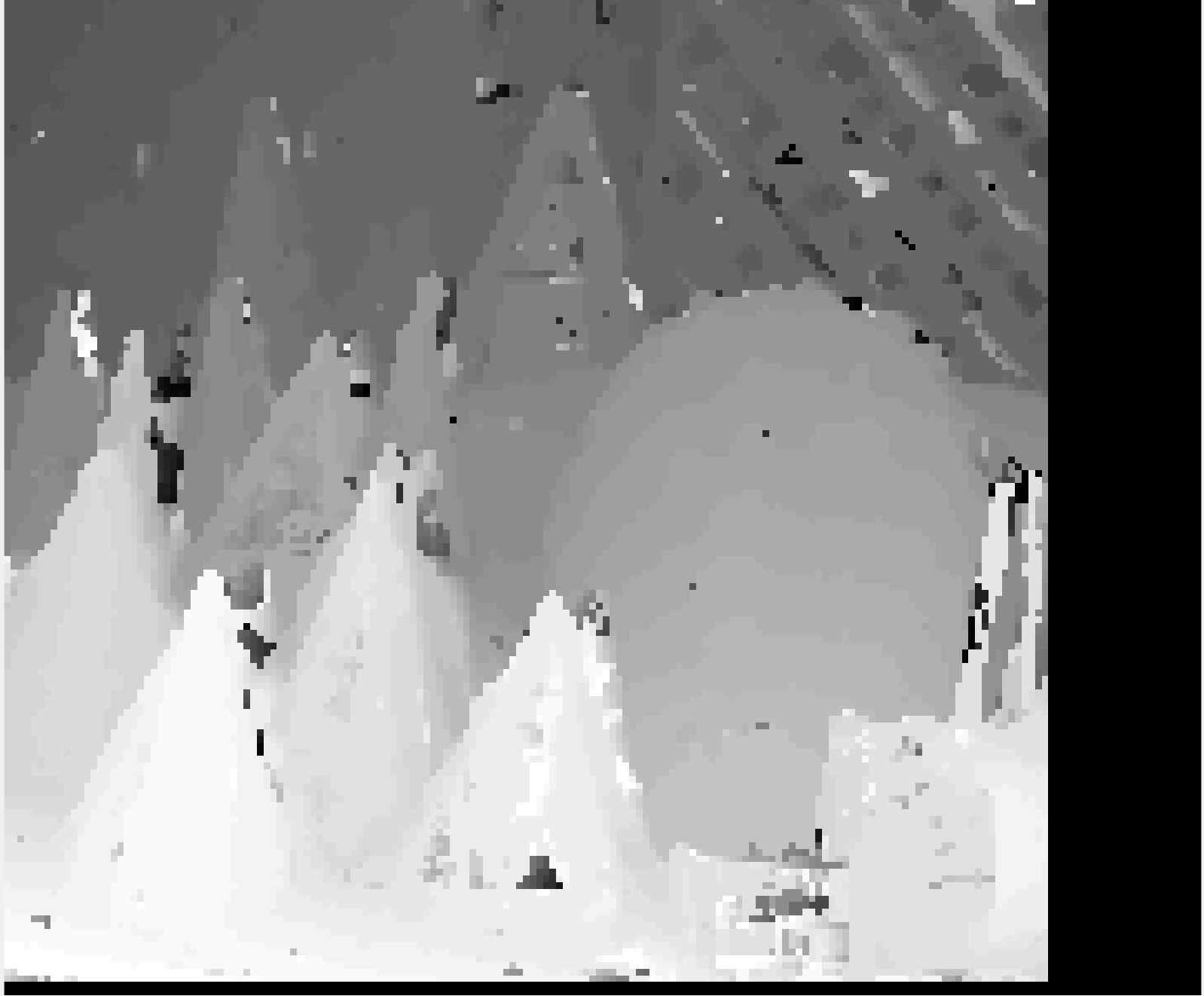
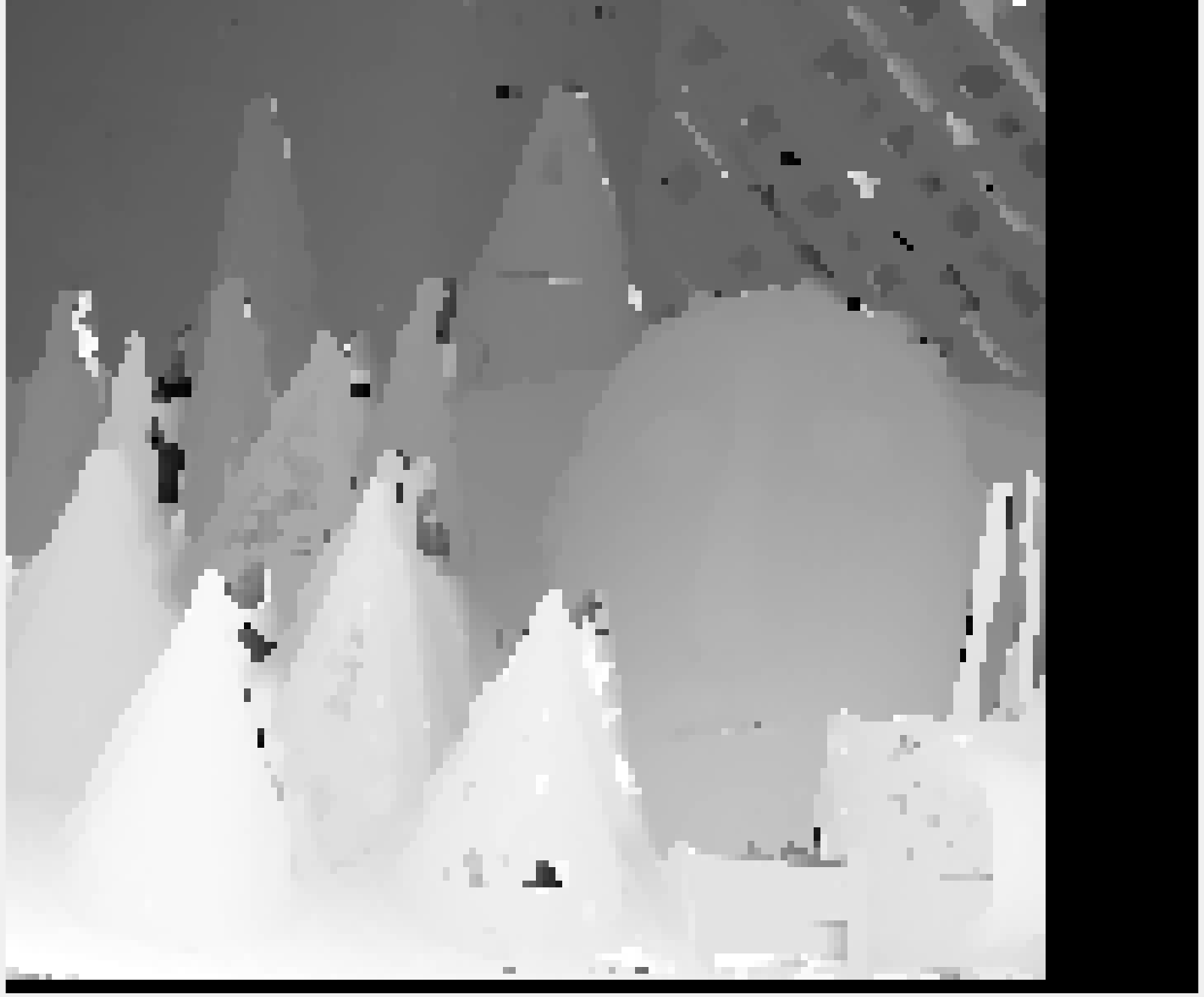
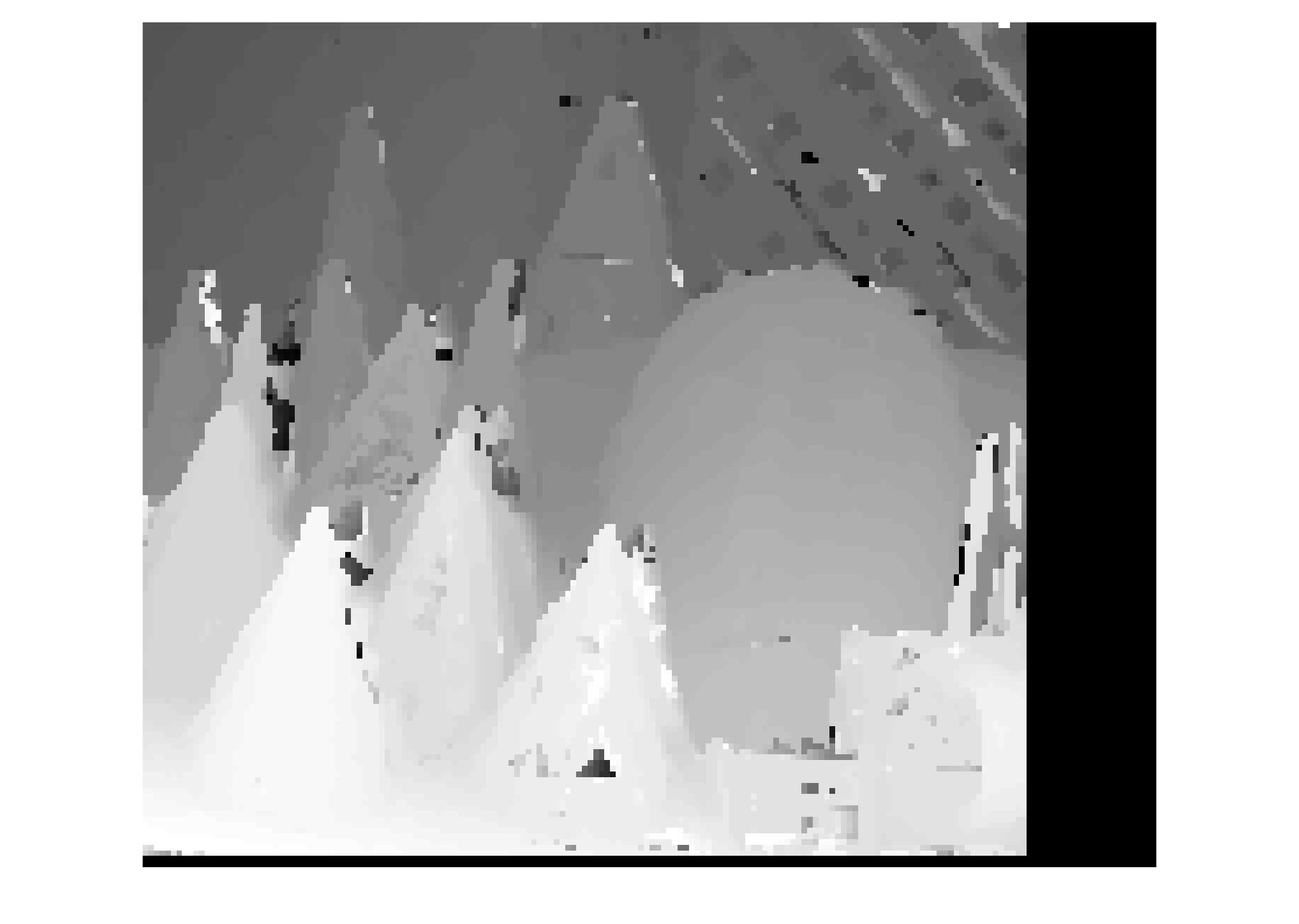
在2003数据集上的结果：（winsize = 30; limit = 70; stride = 10; skip = 3; fullsize=1240\*1110; 运行时间约30s）

结果分析：

改变窗口的大小：

Win size = 20; winsize=30; winsize=60

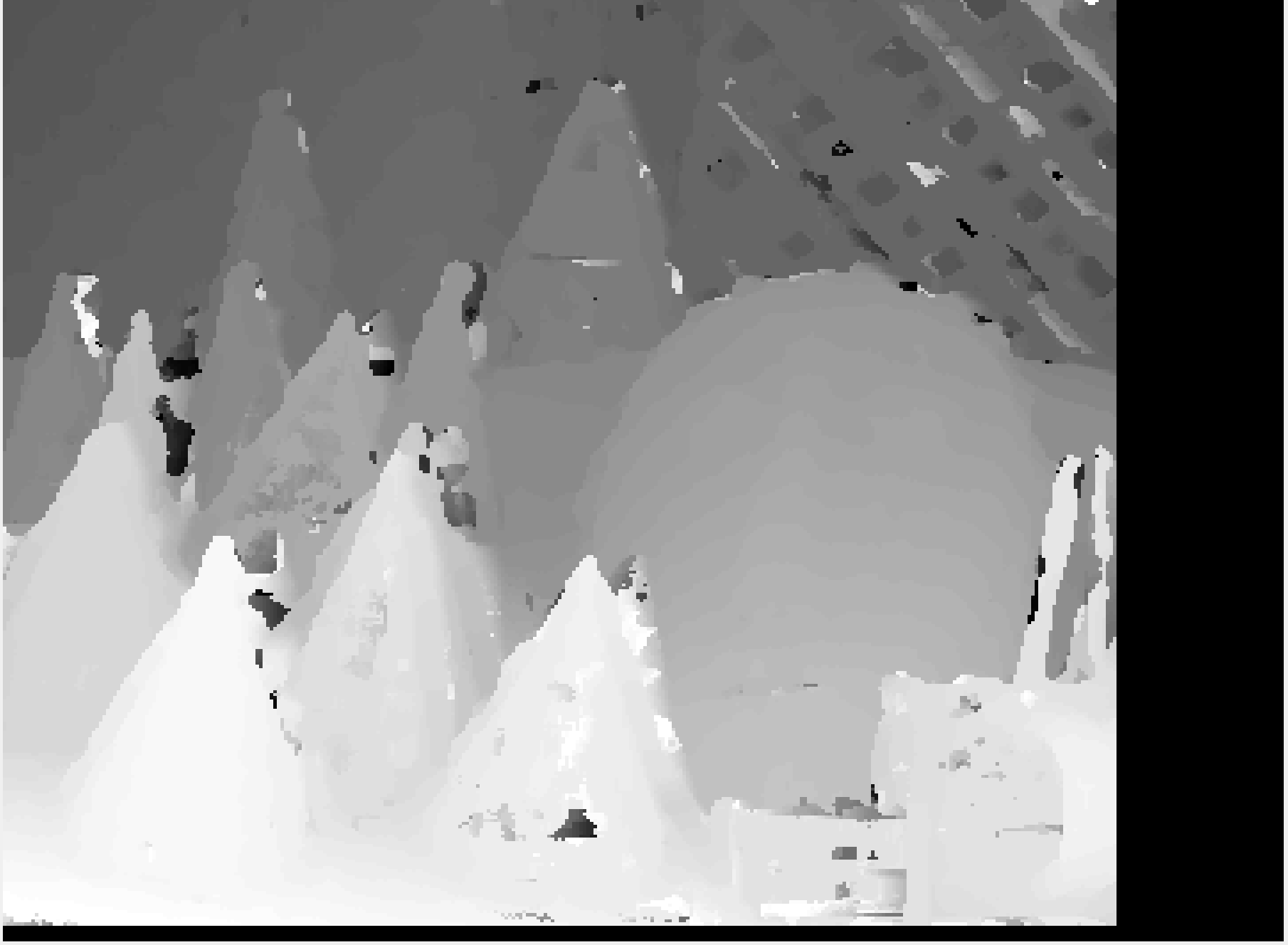
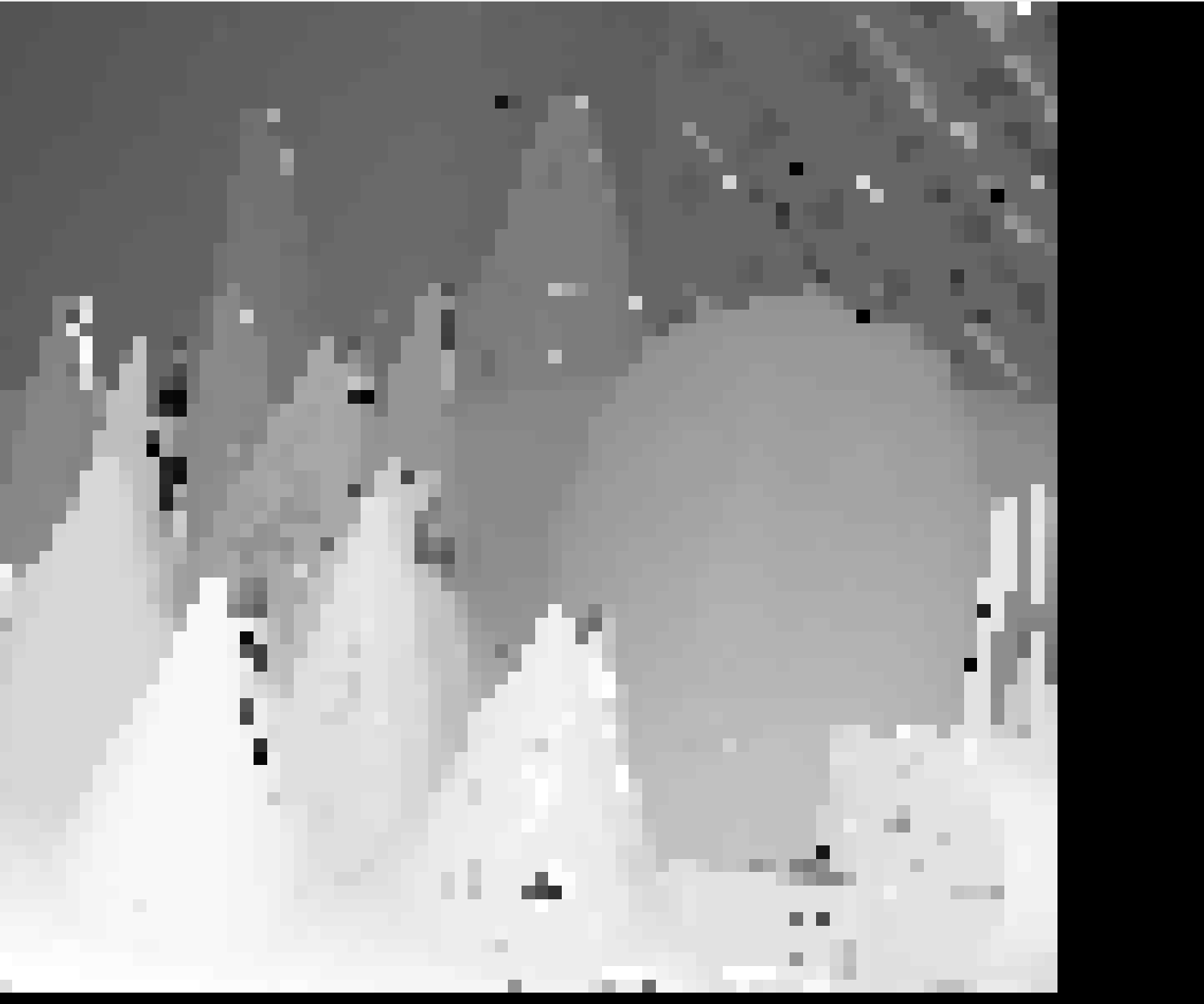
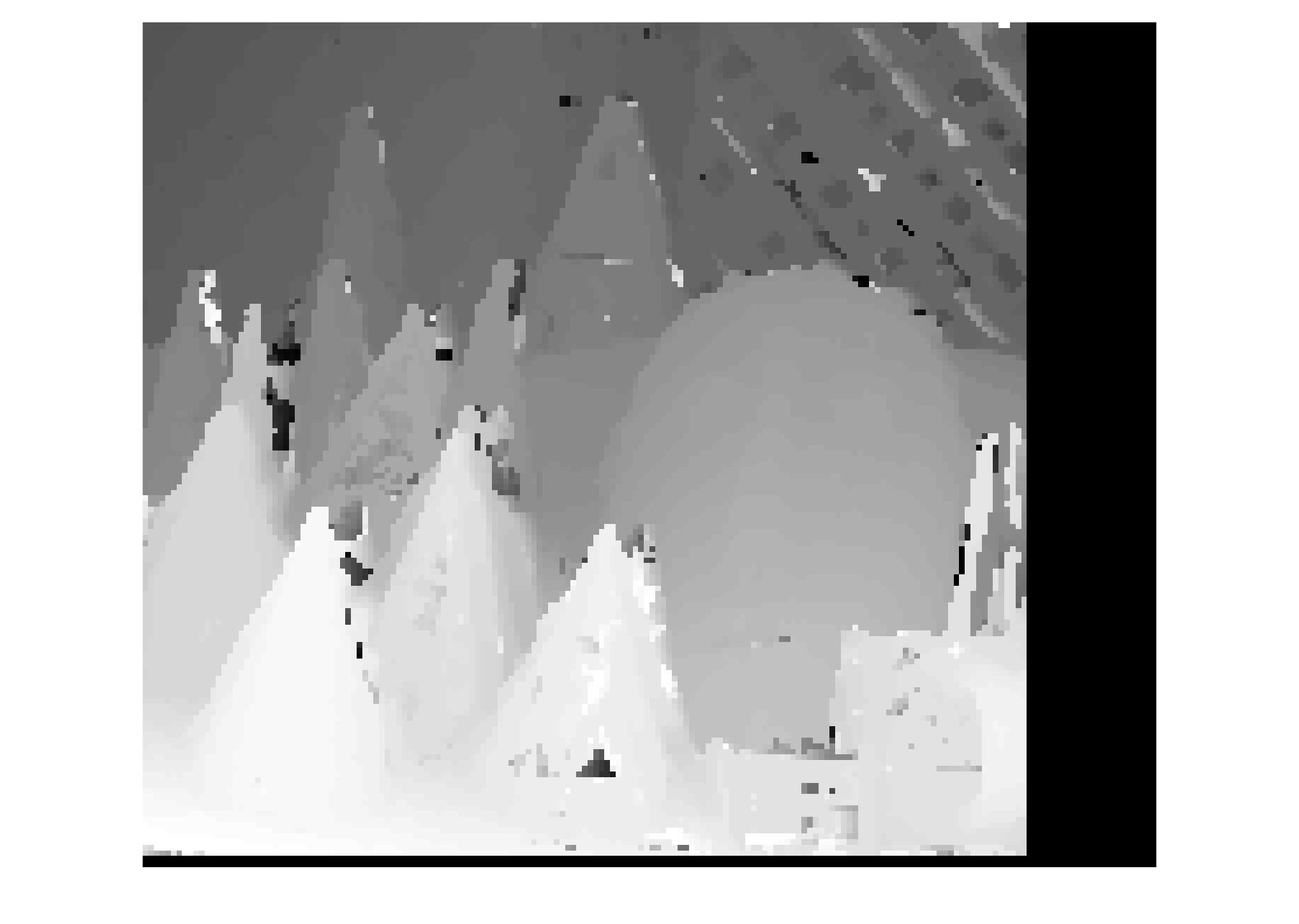
可以看到，窗口小的时候图像精细，但是有很多噪声，因为一个噪声像素对整体的影响变大了。窗口大的时候看起来像平滑过了，噪声影响变小，但是损失了细节。同时大窗口的处理时间会变长，综合考虑认为30的结果最好。

改变**skip**的大小（即搜索的时候每次移动的长度，为了保持搜索范围不变，limit也要改变）：

Skip=1； skip=3； skip=5

=

可以看到skip比较小的比较精细，这是自然的，因为搜索的粒度变小。在skip=5时在面具上能看到明显的波纹和噪声，而1和3的差别并不是很大。但是1的时候计算量太大，一幅图要3分钟，所以还是3最好。

改变**stride**的大小（即深度图的精细程度）

Stride =5; stride=10; stride=20

看到三张结果的差别看上去并不大，因为本身这个算法的结果就不精细，结果的分辨率的影响比起噪声对图像的影响较小。