Yukun Duan

Computer Vision for HCI AU’22

Homework Assignment #3

1）

Here is the original picture: it has 1249\*1249 pixels, I used the picture with odd\*odd pixels so that it can generate pyramid three times successfully based on image size formula.

树上的叶子

描述已自动生成

Firstly, I convert the rgb image to grayscale, which named as gaussian\_0.png, located on the far left, and the three pictures next to it are the three layers of pyramids I got. We can find that each layer of the pyramid has half as many pixels in the x coordinate and half as many pixels in the y coordinate as the previous layer, so the size of the whole image is 1/4 of the previous layer, and the detail is reduced accordingly, which is not as clear as the previous layer.

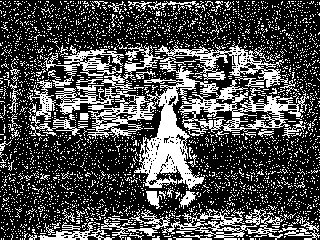
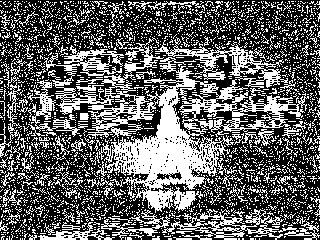


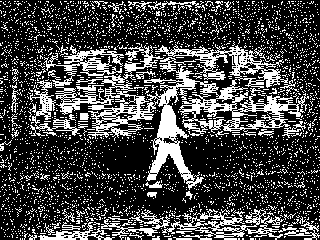
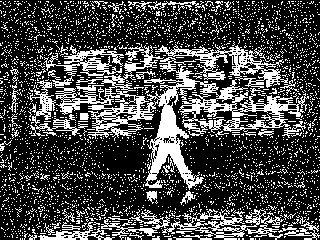
Here are the Laplacian pyramid pictures I get:

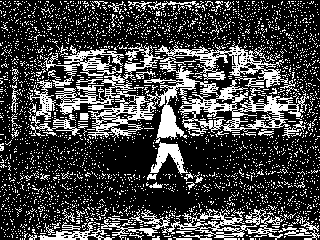
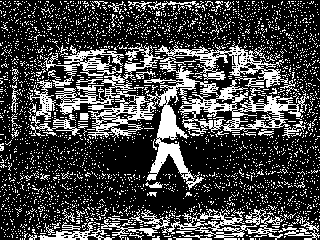


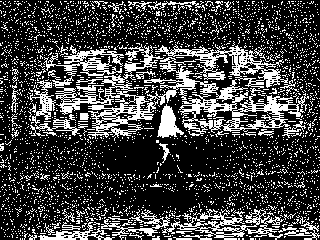
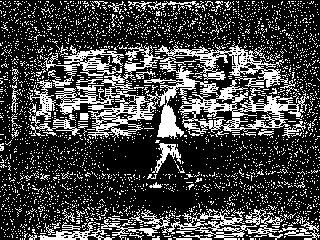
Through the comparison of the Gaussian pyramid and Laplacian error pyramid, we can know that the Gaussian pyramid shows the low-frequency images, and the Laplacian pyramid shows the high-frequency images, because outlines are very clear in Laplacian pyramid.

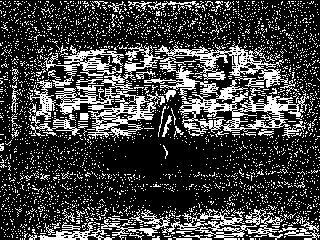
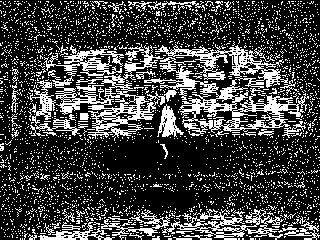
2）I set twelve different T value, get the a series of images. From left to right, top to bottom, the corresponding T value is [10 ,20 ,30 ,40 ,50 ,75 ,100 ,125 ,150 ,175 ,200 ,255].

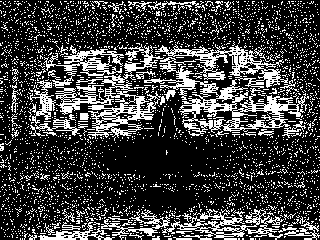










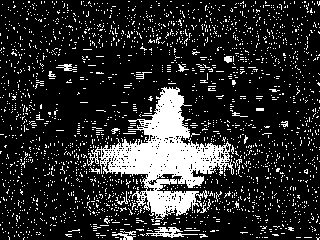
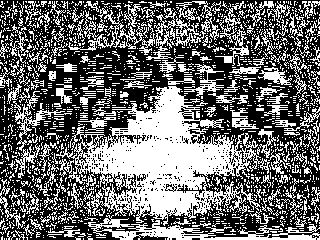
水中的倒影

描述已自动生成

Overall, all the pictures have too much noise in the background and it’s hard to completely separate the person. But the person will become more separated as T value increases, and the best result occurs at T = 50, The outline of the person is the clearest, especially the lines of the limbs and the surrounding noise have a very clear transition. And after that, the outline and the body of the person is gradually melted and disappeared, while most of the noise in the background still remains.

3)

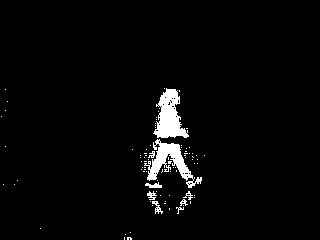
In this question, I used 40 sets of T value from 1 to 40. From left to right, top to bottom, the corresponding T value is: 1:40.



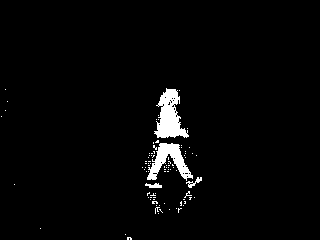




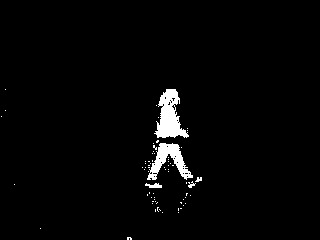


图片包含 游戏机, 烟花

描述已自动生成

图片包含 游戏机

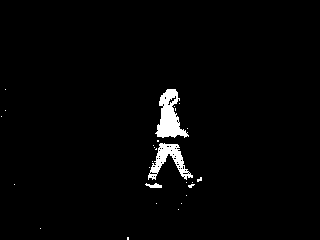
描述已自动生成

图片包含 游戏机

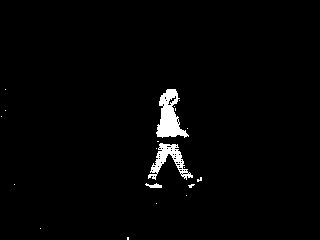
描述已自动生成

游戏机里面的人物

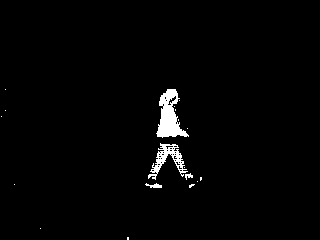
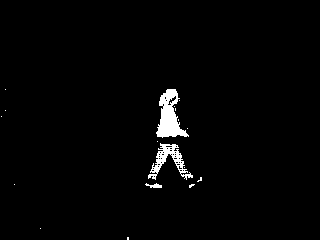
低可信度描述已自动生成

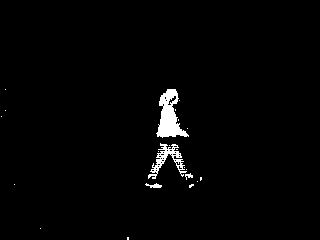
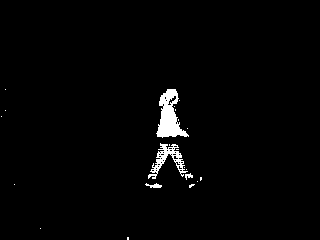
图片包含 游戏机

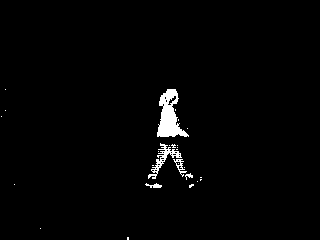
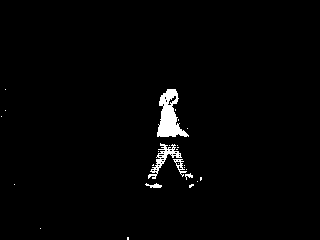
描述已自动生成

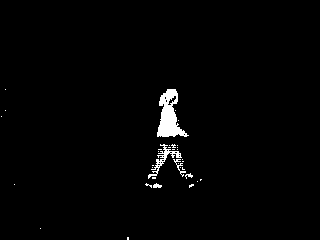
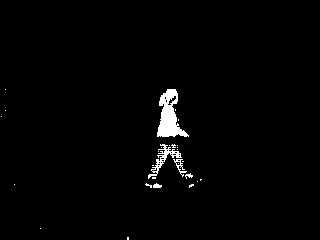
游戏机里面的人物

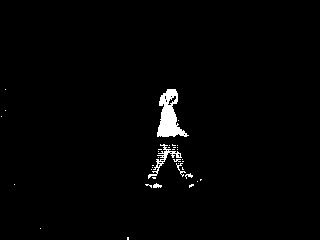
低可信度描述已自动生成



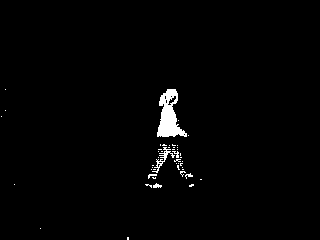




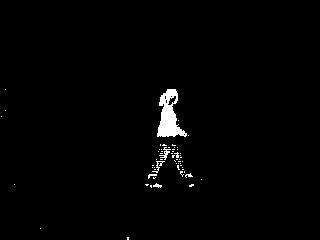


游戏机里面的人物

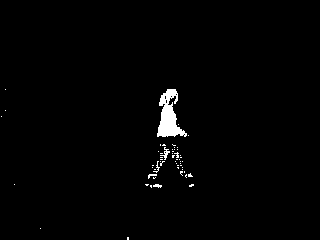
低可信度描述已自动生成

人在黑暗中

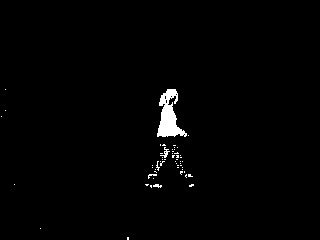
低可信度描述已自动生成

黑暗中亮着灯

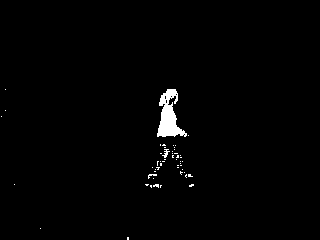
低可信度描述已自动生成

人在黑暗中

低可信度描述已自动生成

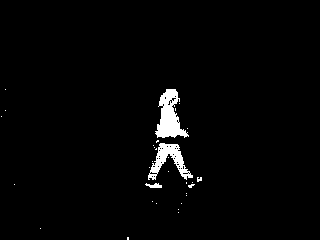
黑暗中亮着灯

中度可信度描述已自动生成

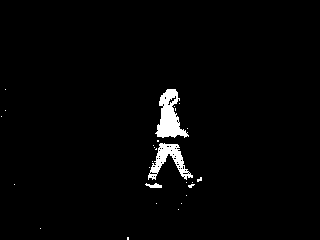
黑暗中的光

中度可信度描述已自动生成

Clearly, the result is much better than the second question. When the Threshold increases from 1 to 16 or so, we can see that the background becomes darker while people’s outline has better high contrast, which means that separation takes effect. The best outcome, in my perspective, occur when T=16, in this image, (comparison of T=15,16,17, and 18 are shown below) there are only a few white pixels remain in other regions and the target object is separated completely. The best separation is only extracting objects from the background without losing body parts. After that, we can see the target object is losing part of the segment as T grows up, and some of the body parts are separated.



T=15 T=16

图片包含 游戏机

描述已自动生成

T=17 T=18

4)

Based on the observations from question 3, I used t16.png in this question, through dilate operation, we can find that the outline of the people in the image becomes more rounded, and a few pixels around the edge from the input image are all dilated in the result. The small tiny pixel around the left bottom side is enlarged as well, and that makes sense because the dilation operation will take OR-ed and put the whole structuring element on the result as long as a single pixel exists in that covered region.

Here is the original one: This is the dilated one:

游戏机里面的人物

低可信度描述已自动生成 图片包含 游戏机, 烟花

描述已自动生成

5）

By using bwlabel() function, we select the desired region, usually the largest one. The connected components will be labeled the same as an entire region. We use an 8-connected selection so that we won’t miss corner pixels, for example, this person’s front feet. We select the second-most frequent label as the target object because 0 is always the most frequent. The left image is the image before selection, and the right image is the selected region. Now the person is almost entirely separated and extracted from the background. However, some part of the back foot is missing, which might be caused by the color of his back ankle having less difference than other body parts from the background.

The dilated image: The selected Image:

图片包含 游戏机, 烟花

描述已自动生成图片包含 游戏机, 画, 飞机

描述已自动生成

Attached code:

% Yukun Duan

% CSE5524 - HW3

% 9/10/2022

%% Problem 1

a=0.4;

wx=[0.25-0.5\*a,0.25,a,0.25,0.25-0.5\*a];

wy=reshape(wx,[5,1]);

img=imread("pic.jpg");

img\_gray=rgb2gray(img);

img\_filter=double(img\_gray);

for i =0:3

imwrite(img\_filter/255, sprintf('gaussian\_%d.png',i)); % print out the pic

GaussianPyramid = blurImage(img\_filter, wx, wy);

estimate = interpolation(GaussianPyramid);

imwrite((img\_filter-estimate)/255, sprintf('laplacian\_%d.png',i));

img\_filter = GaussianPyramid; % evaluate loop

end

%% Problem 2

backgroundIm = double(imread('bg000.bmp'));

inputIm = double(imread('walk.bmp'));

binaryOutput=abs(inputIm-backgroundIm);

T\_list=[10 ,20 ,30 ,40 ,50 ,75 ,100 ,125 ,150 ,175 ,200 ,255];

for T = T\_list

binaryOutput1=binaryOutput;

binaryOutput1(binaryOutput1>T)=1;

binaryOutput1(binaryOutput1~=1)=0;

imwrite(binaryOutput1, sprintf('T%d.png',T));

end

%% Problem 3

for i = 1:30

filename = sprintf('bg%03d.bmp', i-1);

Im(:,:,i) = double(imread(filename));

end

stdev = std(Im, 0, 3); % Calculate stdev matrix

avg = mean(Im, 3); % Calculate mean matrix

BinaryOutput = (inputIm - avg).^2 ./ stdev.^2;

t\_list2 = 1:40;

for t = t\_list2

BinaryOutput1=BinaryOutput;

BinaryOutput1(BinaryOutput1>t^2)=1;

BinaryOutput1(BinaryOutput1~=1)=0;

imwrite(BinaryOutput1, sprintf('t%d.png',t));

end

%% Problem 4

bsIm = double(imread('t16.png')); % Based on output, 't16.png' is the best one

d\_bsIm = bwmorph(bsIm, 'dilate');

imwrite(d\_bsIm, 'dilatedImage.png');

%% Problem 5

[L, num] = bwlabel(d\_bsIm, 8);

second\_most = mode(L(L ~= 0), 'all');

L(L ~= second\_most) = 0;

imshow(L);

imwrite(L, 'Output.png');

%% Problem 1-Blur the image

function bluredSample= blurImage(image, wx,wy)

blured = imfilter(imfilter(image, wx, 'replicate'), wy, 'replicate');

bluredSample = blured(1:2:end, 1:2:end); % Sample the image to 1/2 size

end

%% Problem 1-Interpolation

function newImage=interpolation(image)

newImage=zeros(size(image)\*2-1);

newImage(1:2:end,1:2:end)=image;

row\_ave = conv2(image, [1 1], 'valid')/2;

col\_ave = conv2(image, [1;1], 'valid')/2;

newImage(1:2:end,2:2:end) = row\_ave;

newImage(2:2:end,1:2:end) = col\_ave;

mid\_ave = conv2(row\_ave,[1;1], 'valid')/2;

newImage(2:2:end, 2:2:end) = mid\_ave;

end