Practical Exercise 2
Abgabe: Lena C. Wolos, Sven Niendorf, Isabelle Maye

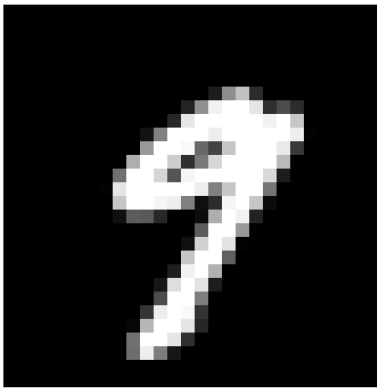
Der Code ist zusätzlich als Datei einsehbar.

a)

Code:

```
#Exercise 2
# Task a

load("digit0.mat")

% function to show a certain amount of images (im_num) for each digit
function show_data(im_num)
  for j= 0:9
      s=strcat("digit",int2str(j),".mat");
      load(s)
    for i = 1:im_num
        I = D(i,:);
        I = reshape(I, [28,28]);  % converts I to size to 28 x 28
        I=imrotate(I,270);    % rotates I by 270 degrees
        figure(1), imshow(fliplr(I),[]); % shows flipped I
        pause(0.1);     % pauses for 0.1 seconds
    end
  end
endfunction
```

Die Funktion erzeugt wie gefordert im_num Beispielbilder für jede Ziffer. Ruft man die Funktion mit im_num = 5 auf und wartet bis das letzte Beispielbild angezeigt wird, wird folgendes angezeigt:

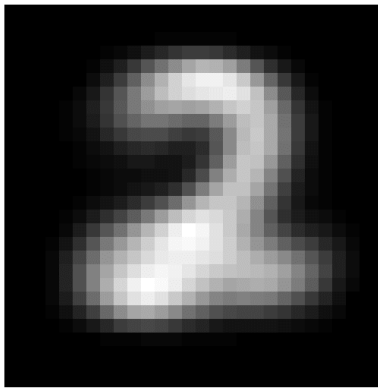


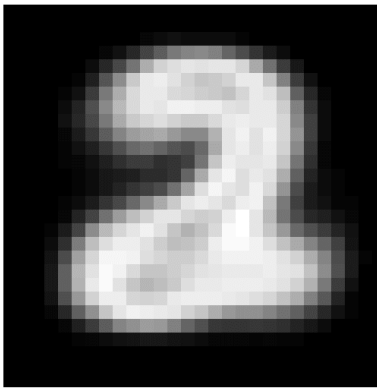
b)

Code:

```
load("digit2.mat")

# Task b, Calculating the mean

Sum= zeros(1,784);
s_D= size(D,1);
for im_num= 1: s_D
  for i=1:784
     I = D(im_num,:);
     Sum(i) = Sum(i) + I(i); % adds the value of the pixel i in picture im_num to the sum of values of pixel i
  end
end

Avg=Sum/s_D;                % devides the sums by the number of images
I = reshape(Avg, [28,28]); % converts I to size to 28 x 28
I = imrotate(I,270);
figure(1), imshow(fliplr(I),[]);

# Calculating the variance
SumDev= zeros(1,784);
for im_num= 1: s_D
  for i=1:784
     I = D(im_num,:);
     SumDev(i) = SumDev(i)+((I(i)-Avg(i))^2); % adds the square of the value of pixel i minus the mean to the sum
  end
end

Var=SumDev/s_D; % devides the sums by the number of pictures

I = reshape(Var, [28,28]); % converts I to size to 28 x 28
I=imrotate(I,270);
figure(2), imshow(fliplr(I),[]);
```

Es werden die folgenden Bilder ausgegeben (links Figure 1, rechts Figure 2):



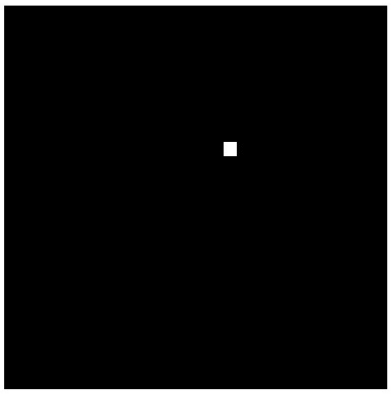


c) und d)

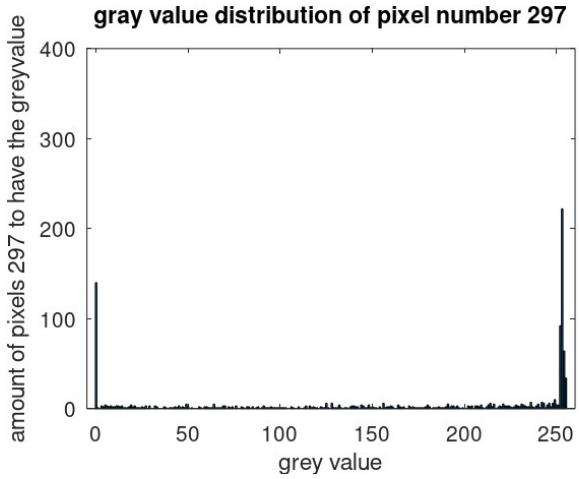
<u>Code:</u>

```
#Task c and d

load("digit3.mat")
s_D = size(D,1);  % number of images
k = 1;      % index of the figures
m_x = 0:255;

Pixel = [297,155,203,210,217,399,410,595,605,715]; % 10 interesting pixel

for j = 1:10
  p = Pixel(j);
  M = zeros(1,256);
  P = zeros(1,784);
  P(p) = 255;   % changes the value at the given pixel p to 255

  for im_num= 1: s_D
       I = D(im_num,:);
       i=I(p)+1;
       M(i)=M(i)+1; % M(i) counts the number of images with value i-1 at pixel p
  end

  P = reshape(P, [28,28]); % converts P to size to 28 x 28
  P=imrotate(P,270);
  figure(k); imshow(fliplr(P),[]);

  figure(k+1);
  bar(m_x,M)
  t = cstrcat("gray value distribution of pixel number ", int2str(p));
  title(t);
  axis([-5, 260, 0, 400]);
  xlabel ('grey value');
  y = cstrcat("amount of pixels ", int2str(p) , " to have the greyvalue");
  ylabel(y);
k = k+2
end
```

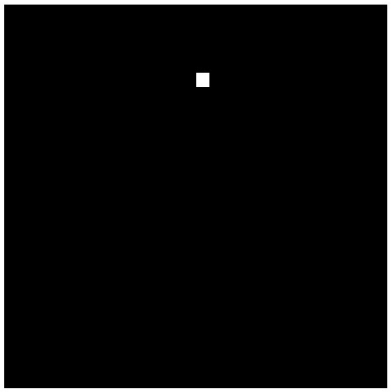
Es werden folgende Bilder ausgegeben:



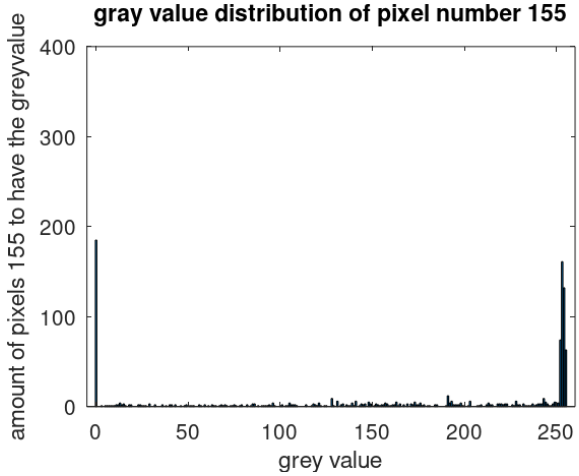
1


gray value distribution of pixel number 297
amount of pixels 297 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

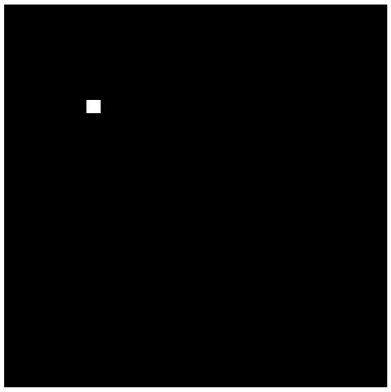
2



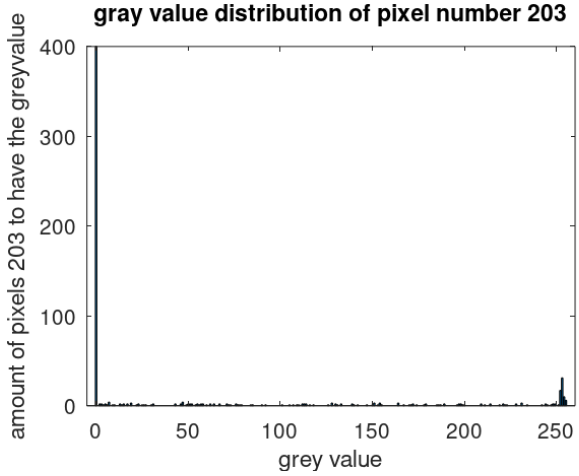
3


gray value distribution of pixel number 155
amount of pixels 155 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

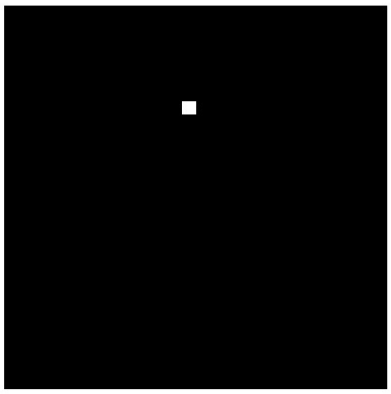
4



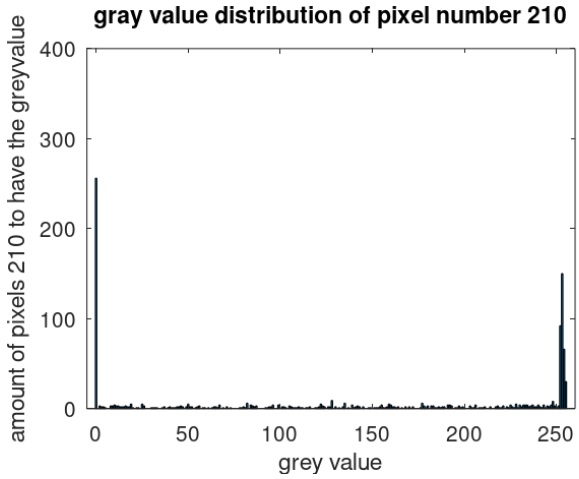
5


gray value distribution of pixel number 203
amount of pixels 203 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

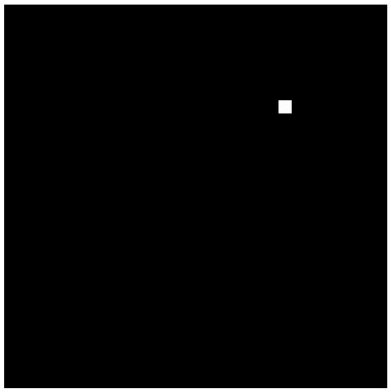
6



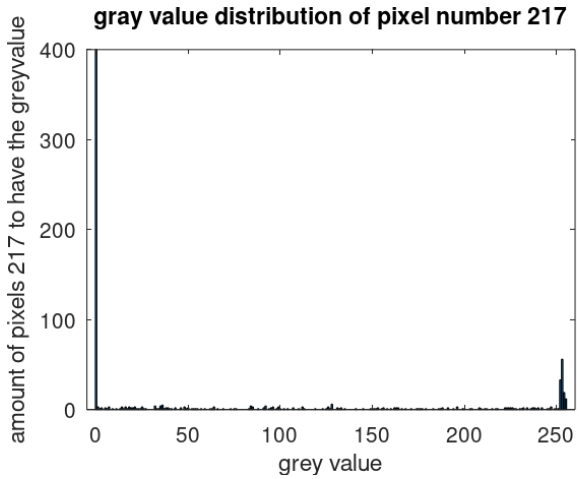
7


gray value distribution of pixel number 210
amount of pixels 210 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

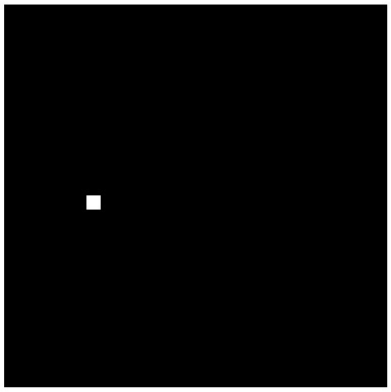
8



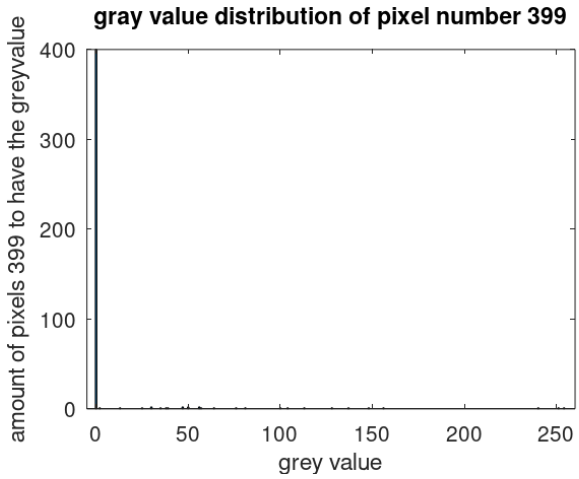
9


gray value distribution of pixel number 217
amount of pixels 217 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

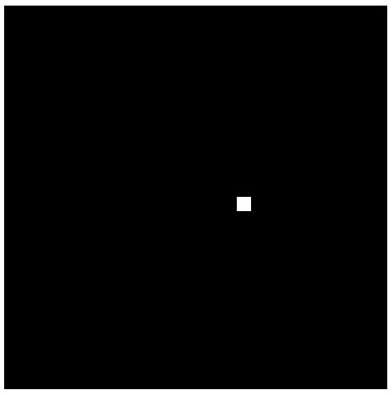
10



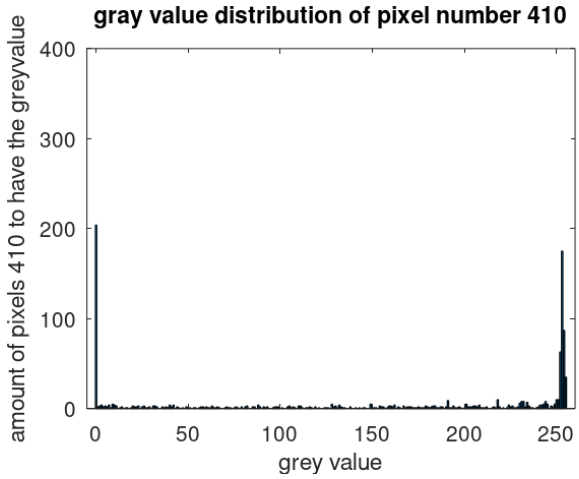
11


gray value distribution of pixel number 399
amount of pixels 399 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

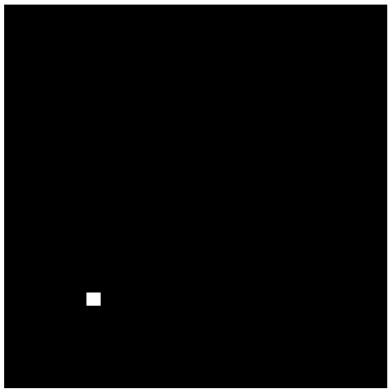
12



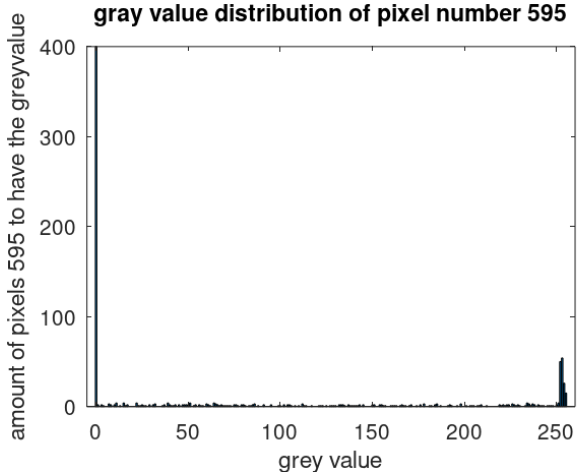
13



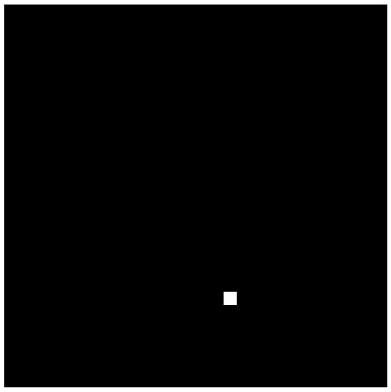
14



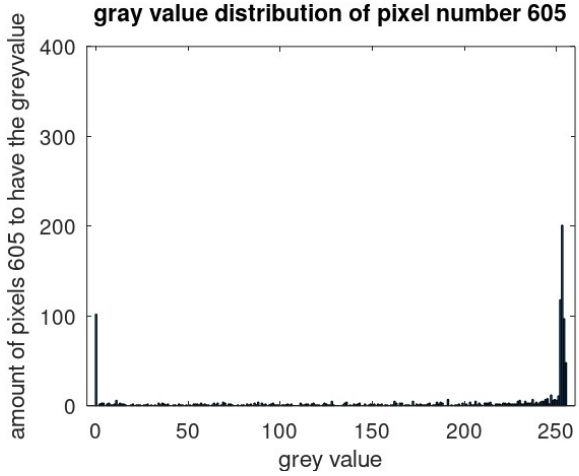
15



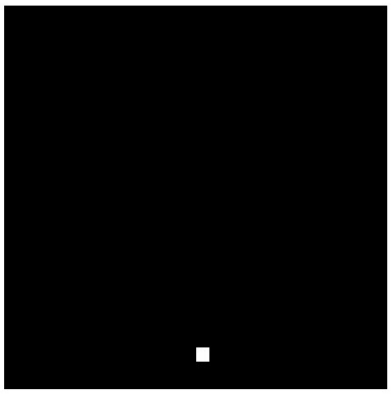
16



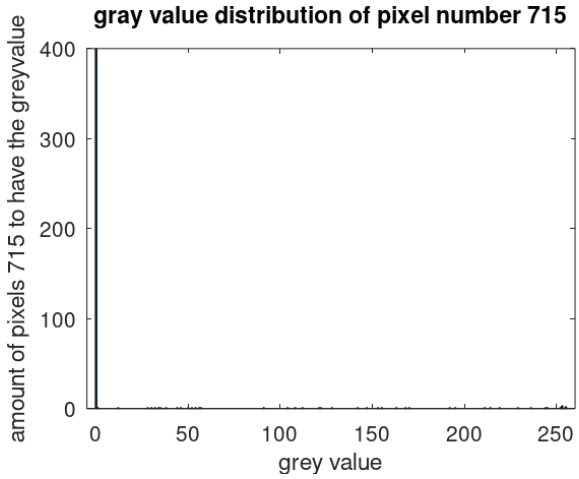
17



18



19


gray value distribution of pixel number 715
amount of pixels 715 to have the greyvalue
400
300
200
100
0
0
50
100
150
200
250
grey value

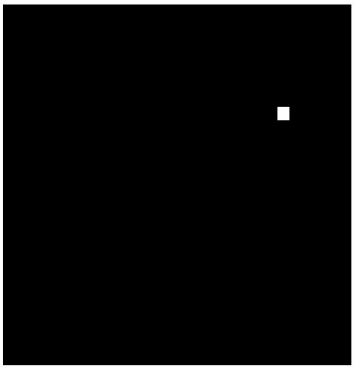
20

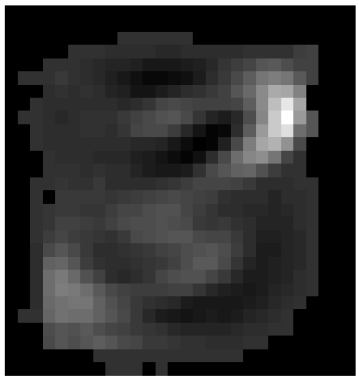
e)

Code:

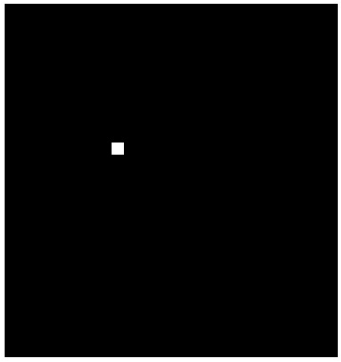
```
#Task e

C=[];
function show_correlation (dig, xb, yb)

  s = strcat("digit", int2str(dig), ".mat");
  load(s)

  b_value= 28*(yb-1)+xb;
  base= rotdim(D(:,b_value)); % extracting the grey values of each image for base pixel

  C=[];

  for i=1: 784
    pixel = rotdim(D(:,i)); % extracting the grey values of each image for pixel i
    c = corr(pixel,base);   % correlation between pixel i and base pixel
    c=(c+1)*255/2;          % converting correlation to grey value
    C=[C,c];                % building vector of grey values for correlation image
  end

  P = zeros(1,784);
  P(b_value)=255;


  P = reshape(P, [28,28]); % converts P to size 28 x 28
  P=imrotate(P,270);
  figure(1); imshow(fliplr(P),[]);

  C=reshape(C,  [28,28]); % converts C to size to 28 x 28
  C = imrotate(C,270);
  figure(2), imshow(fliplr(C),[]);

endfunction
```

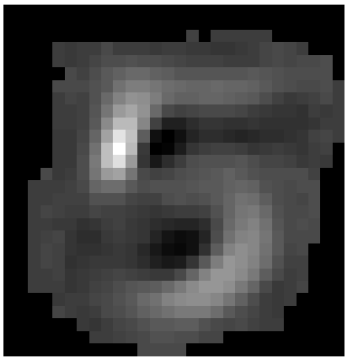
Der Aufruf show_correlation (2, 23, 9) erzeugt:





Für show_correlation (5, 10, 12) erhalten wir:



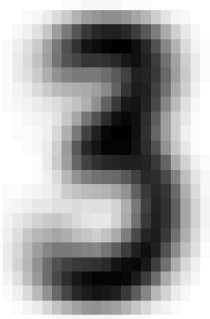


f)

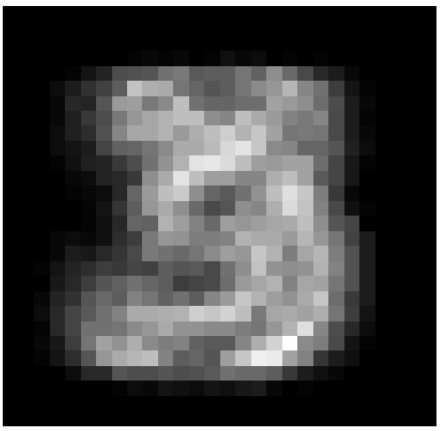
Code:

```
#Task f

 a=1;

function show_joint_probs(v_min, v_max, digit)
  s = strcat("digit",int2str(digit),".mat");
  load(s)
  size_D = size(D,1);
  P=[];

  for i=1:784
    pixel = rotdim(D(:,i));              % extracting the grey values of each image for pixel i
    pixel_max = pixel < v_max;
    pixel_min = pixel >=v_min;
    pixel_count = pixel_max & pixel_min;
    sum_p = sum(pixel_count);
    p = sum_p / size_D;                  % the probabilite of pixel i having a grey value in the interval [v_min,v_max]
    P=[P,p];
  endfor

  P;
  p_min= min(P);
  p_max= max(P);
  range= p_max - p_min;
  G=[];

  % transforming the probabilities into grey values
  for i=1:784
    g=P(i);
    g=(g-p_min)*255/range;
    G=[G,g];
  endfor

   G = reshape(G,  [28,28]); % converts G  to  size to 28  x 28
   G = imrotate(G,270);
   figure; imshow(fliplr(G),[]);

 endfunction

 show_joint_probs(0, 1,3)
 show_joint_probs(1, 50,3)
 show_joint_probs(50, 150,3)
 show_joint_probs(250, 255,3)
```

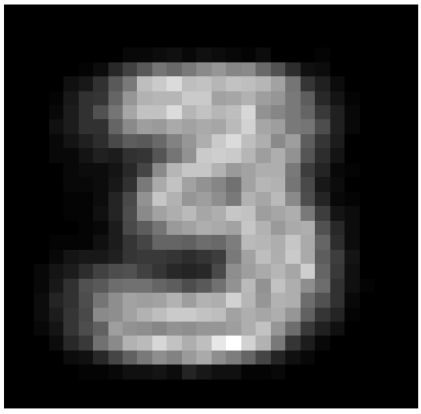
Wir erhalten für das Intervall [0, 1[das Bild:



Wir erhalten für das Intervall [1, 50] das Bild:



Wir erhalten für das Intervall [50, 150] das Bild:



Wir erhalten für das Intervall [250, 255] das Bild: