1.) Polar coordinates

a.)

The distance between 2 points in polar coordinates can calculate with this code.

distance.m

```
function [dic] = distance(radius1, angle1, radius2, angle2);
%Abfrage abstände und winkel
%x = r * cos( phi );
%y = r * sin( phi );
%winkel = winkel1 - winkel2;
dic = (radius1^2 + radius2^2 - 2 * radius1 * radius2 * cos( angle1 - angle2 ))^(1 / 2);
```

b.)

The distance d between the points $A(3, {}^{\pi}/_{8})$ and $B(7, {}^{3\pi}/_{4})$ is $d = 8.6066$.

2.) sum calculation

The sum can compute with this code.

calc_sum.m

```
max = 1000; % maximal number
sum = zeros(1,max);
x = 1:1:max;
% computation of the sum
for i = 1:1:max
        for j = 1:1:i
        sum(i) = sum(i) + ((-1).^(j)/(2*j+1));
        end;
end;
plot1 = semilogx(x,sum); %log axis
xlabel ( ’ x ’ ) ; %labeling the axes
ylabel ( ’ sum ’ ) ;
% computation of the sum for the a, b and c.)
x = zeros(1,3);
sum = zeros(1,3);
for i = 2:2:6
        for j = 1:1:10.^(i)
        sum(round(i/2)) = sum(round(i/2)) + ((-1).^(j)/(2*j+1));
        end;
  x(round(i/2))= 1*10.^(i)
end;
plot2 = semilogx(x,sum); %log axis
xlabel ( ’ x ’ ) ;
ylabel ( ’ sum ’ ) ;
%title( ’sum’ ) ;
```

For the result see figure (1)

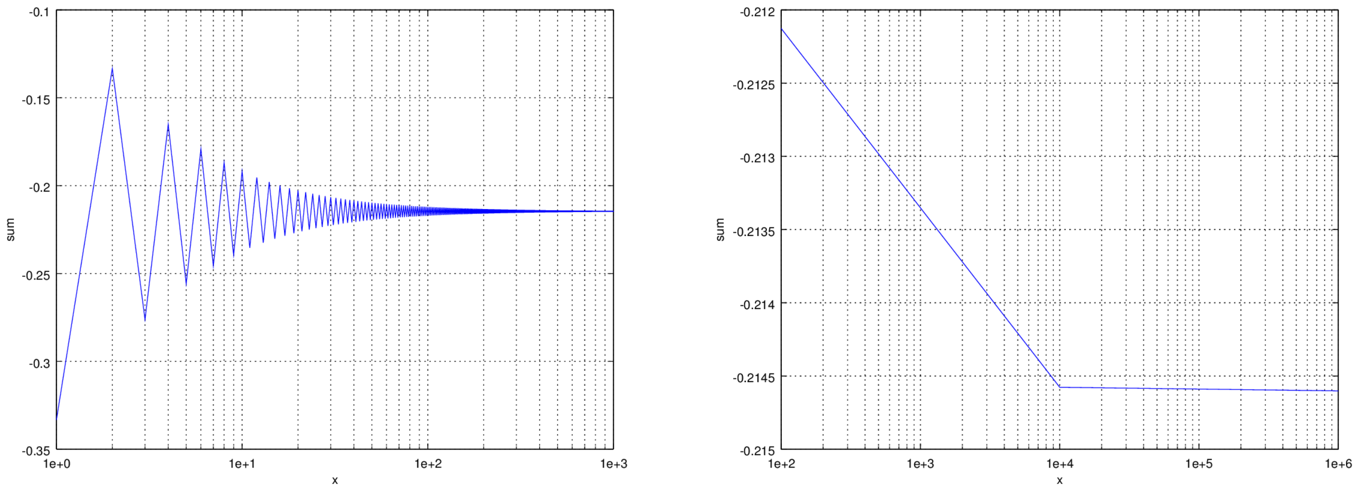


Figure 1: sum calculation

3.) Picture puzzle

a.)

b.)

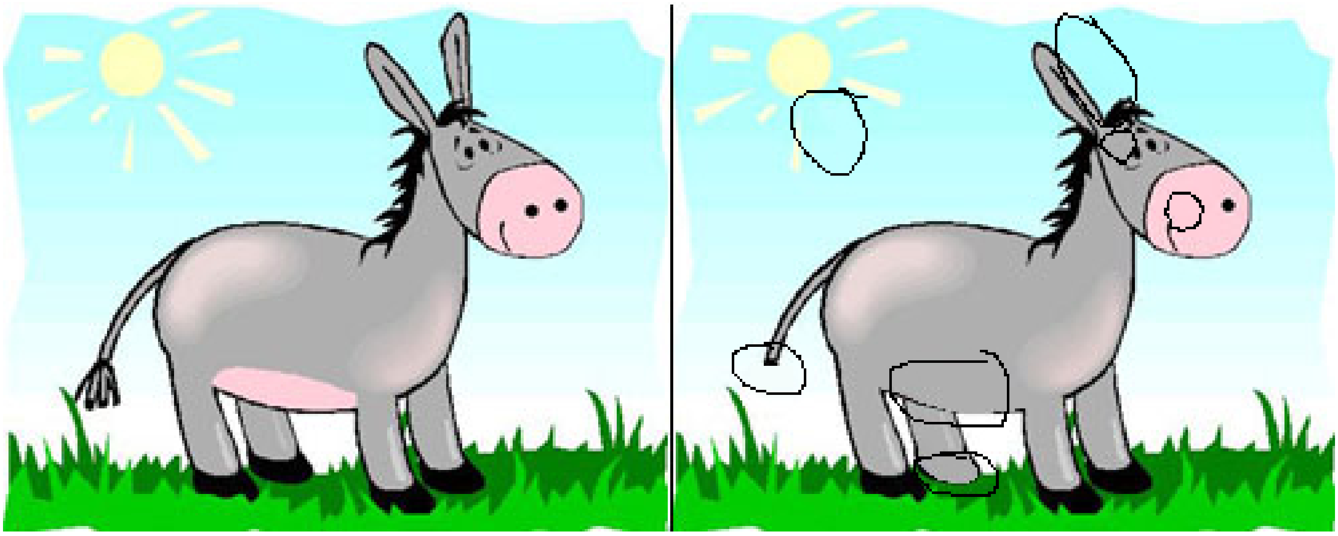


Figure 2: marked mistakes

c.)

This code can count the mistakes of picture puzzle, if the pictures are the same in pixel by pixel without the mistakes.

```
        search.m
```

```
function [mistakes] = search(picture)
%MISTAKES
% This function can calculate the counts of mistakes of a puzzle picture
%rgb color to gray
picture = rgb2gray(picture);
%determination of the size of the picture
sizes = size(picture);
% extract to single pictures
picture1 = picture(1:sizes(1),1:(round(sizes(2)/2)-1));%
picture2 = picture(1:sizes(1),(round(sizes(2)/2+1)):sizes(2));%
% extraction of the mistakes
minuspic1 = minus(picture2,picture1);
%imshow(minuspic1);
%count and determine of mistakes
[struc,mistakes] = bwlabel(minuspic1);
```

4.) gamma correction

a.)

A histogram is a diagram, which it can display the counts of pixels of every intensity level. The number of pixels are set it as x-axis and the intensity levels are set it as y-axis. Usually the number of pixels are display with vertical lines, but sometimes it is useful to use a bar plot for a better understanding. Also you can illustrate it as points or a single graph. For example a histogram of this picture (3).

```
matlab_uebung_2_Ch.m
```

```
% picture input
picture = imread('flower.jpg');
% create a histogram
hist(picture);
imhist(picture);
```

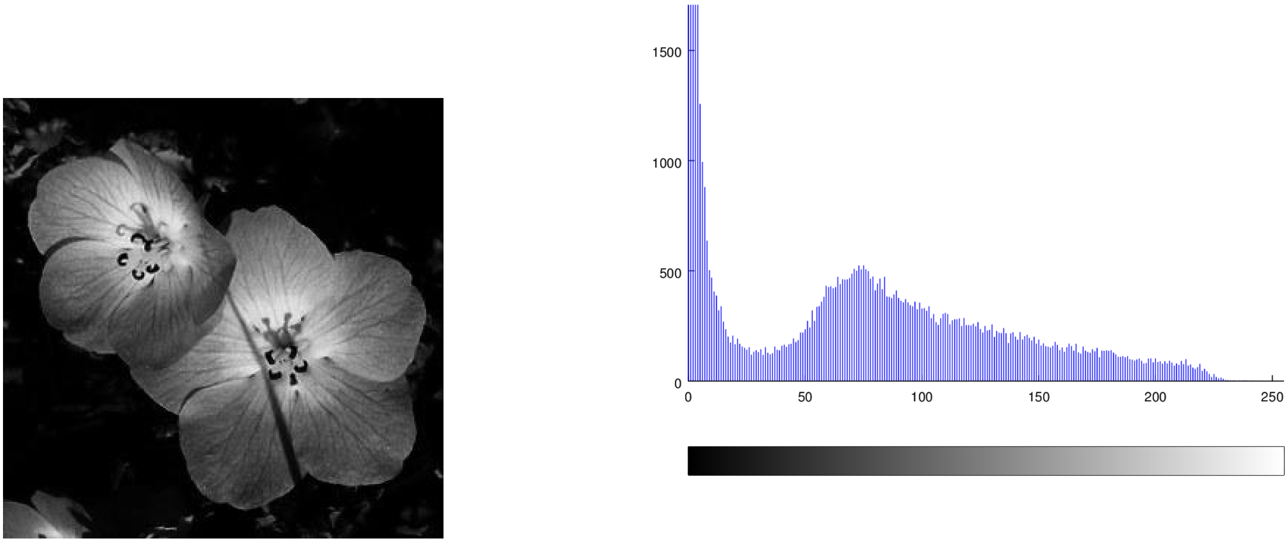


Figure 3: histogram of a picture

b.)

The histgram equalization can performed with MATLAB commando 'histeq()' .

```
% histrogram equalization
histoeq = histeq(picture, 256);
imshow(histoeq);
imhist(histoeq);
```

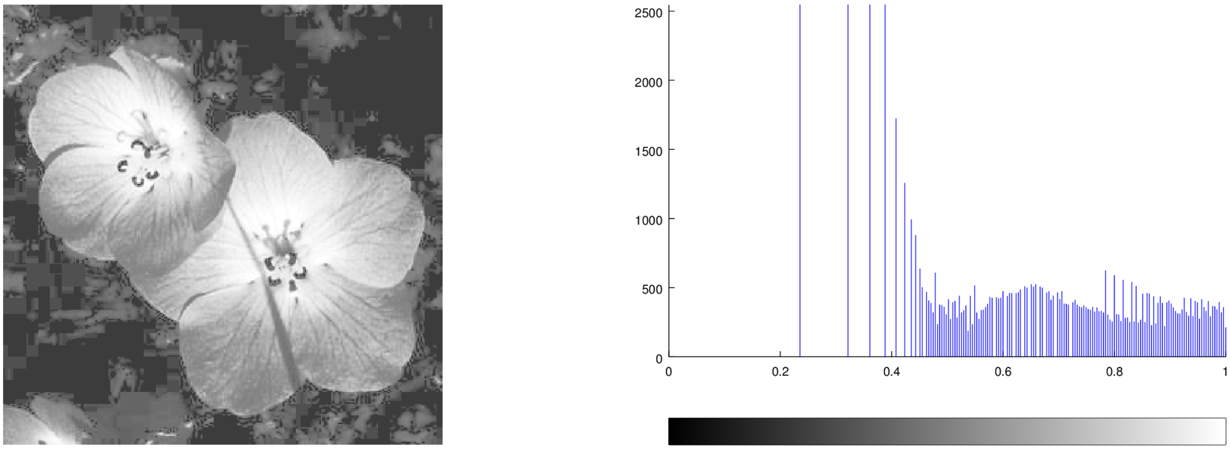


Figure 4: histogram equalization of a picture

The transformation function can be find with these commandos.

```
% transformation function
histnorm = imhist(picture)./numel(picture);
cdf = cumsum(histnorm);
x = linspace(0,1,256);
```

```
%plot
plot(x,cdf),  xlabel ( ' x ' ), ylabel ( ' f(x) ' );
```

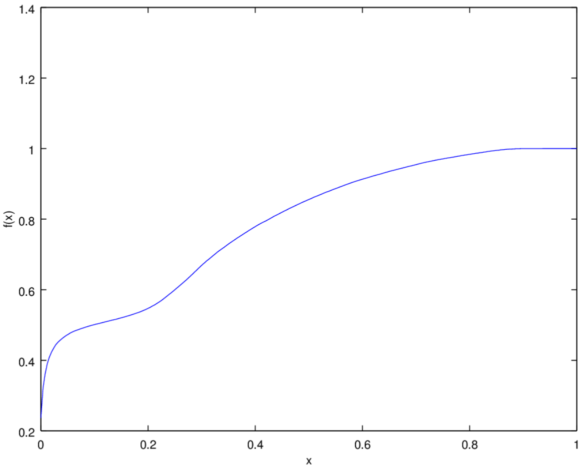


Figure 5: plot of the transformation function $f(x)$

c.)

```
%contrast enhancement
low = log(double(picture));
imshow(low);
high = 10.^(double(low));
imshow(high);
```

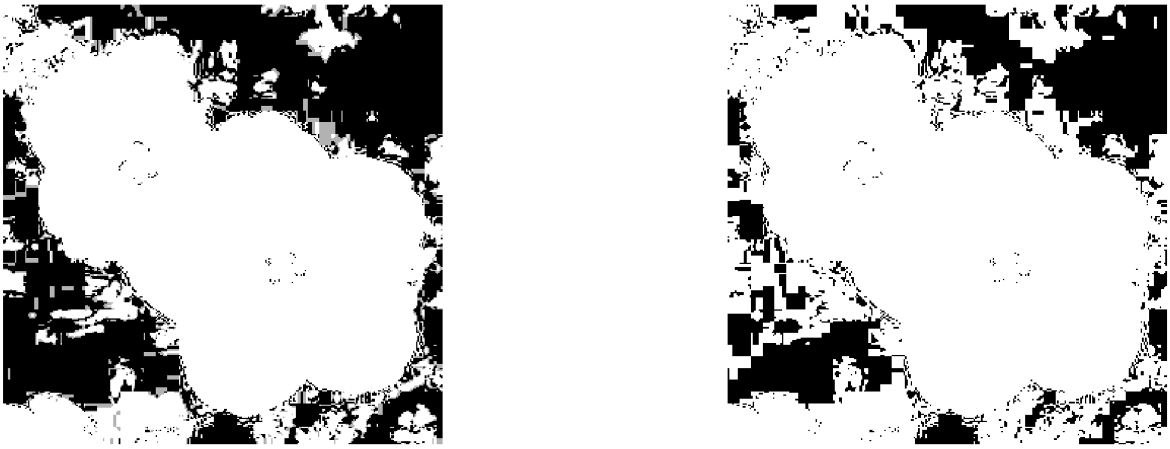
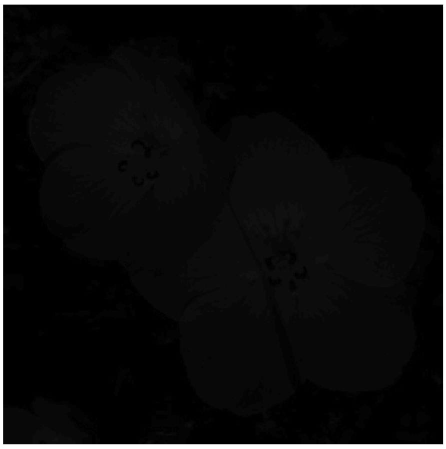


Figure 6: contrast enhancement by fraction of pixels in the lower (left) and higher (right) saturation region

d.)

```
% gamma correction
gamma1 = picture.^(0.5);
```

```
gamma2 = picture.^(2.0);
imshow(gamma1), figure, imshow(gamma2);
```



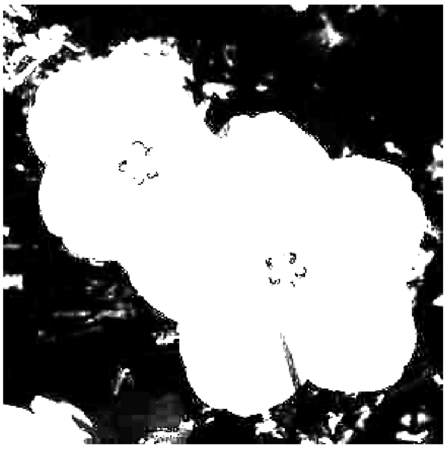
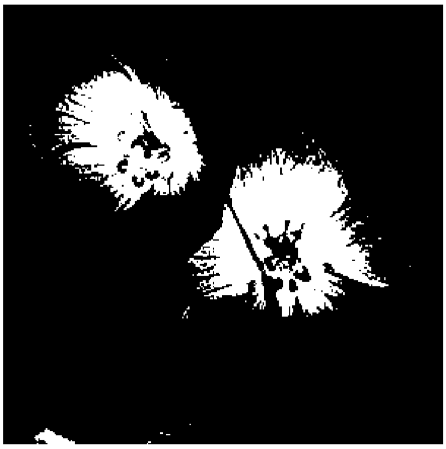


Figure 7: gamma correction with $\gamma = 0.5$ (left) and $\gamma = 2.0$ (right)

e.)

```
%binary pictures
binpic1 = im2bw(picture);
imshow(binpic1);
binpic2 = im2bw(gamma1);
imshow(binpic2);
binpic2 = im2bw(gamma2);
```



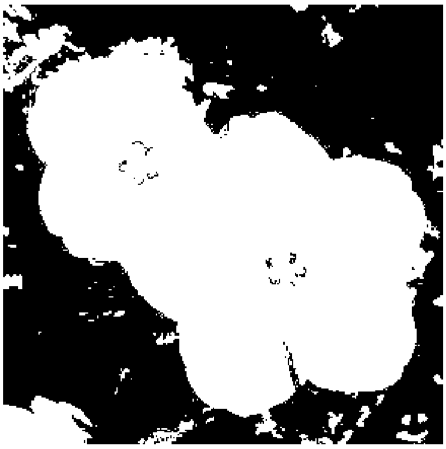


Figure 8: binary image before (left) and after (right) gamma correction with $\gamma = 2.0$

f.)

```
% enhancement technices
imshow(picture);
imshow(histoeq);
adjustpic = imadjust(picture);
imshow(adjustpic);
adapthistpic = adapthistoeq(picture);
imshow(adapthistpic);
```

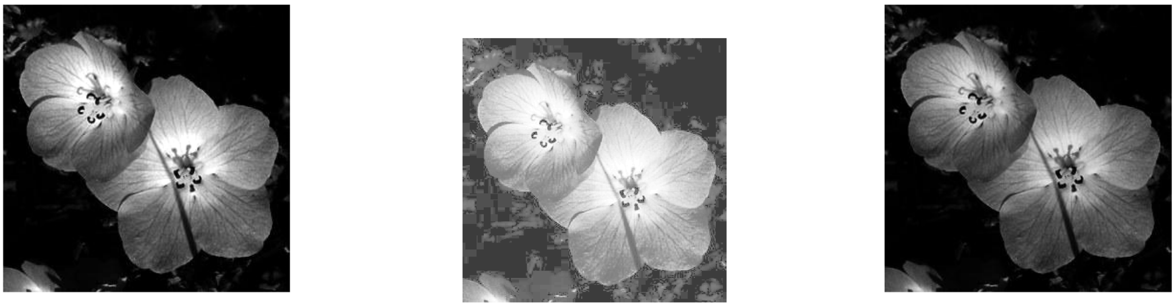


Figure 9: function imadjust (left), histeq (mid) and adapthisteq (right)