Image Processing		
Praktikan	**Aslab**	
Nama: xxxx Nim: xxxx	Annur Hangga Prihadi	065001800028
	Faiz Kumara	065001800003

PRAKTIKUM 11

DATA SAINS DAN ANALITIK

Topik pertemuan praktikum ke-sebelas adalah mengolah data gambar dengan menggunakan Image Hashing untuk mencari kemiripan suatu gambar, Hough Transfrom untuk mendeteksi garis gambar, Morfologi untuk mendeteksi elemen penataan, Orb untuk mendeteksi poin kecil pada suatu gambar

Source Code:

Image Hashing:
https://github.com/hanggaa/PrakDSDA/blob/main/Prak_11_Hashing1.ipynb

Plot Morphology:
https://github.com/hanggaa/PrakDSDA/blob/main/Prak_11_Plot_Morphology.ipynb

Plot ORB:
https://github.com/hanggaa/PrakDSDA/blob/main/Prak_11_Plot_Orb.ipynb

Plot Hough Transform:
https://github.com/hanggaa/PrakDSDA/blob/main/Prak_11_Plot_Circular_Elliptical_Hough_Transform.ipynb

Latihan

Image Hashing

1. **Memasang library yang dibutuhkan**

```
In [1]: import sys
        !{sys.executable} -m pip install imagehash

Requirement already satisfied: imagehash in c:\users\hangg\anaconda3\lib\site-packages (4.2.1)
Requirement already satisfied: pillow in c:\users\hangg\anaconda3\lib\site-packages (from imagehash) (8.2.0)
Requirement already satisfied: six in c:\users\hangg\anaconda3\lib\site-packages (from imagehash) (1.15.0)
```

2. Implementasi Average Hash Image 1

In [2]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.average_hash(Image.open('D:/Gambar/Whale.jpg')) #Sesuaikan dengan directory
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.average_hash(Image.open('D:/Gambar/Whalee.jpg')) #Sesuaikan dengan directory
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: fffff9efff0f0000
Hash Gambar 2: ffff9ff7fff00000
Gambar berbeda, perbedaan hash: 14
```

3. Implementasi Average Hash Image 2

In [3]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.average_hash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.average_hash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: fffff9efff0f0000
Hash Gambar 2: fffff9efff0f0000
Gambar mirip!
```

4. Implementasi Perception Hashing 1

In [4]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.phash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.phash(Image.open('D:/Gambar/Whalee.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: 81cf6e90d1370dce
Hash Gambar 2: d49a3bc5a462d89b
Gambar berbeda, perbedaan hash: 34
```

5. Implementasi Perception Hashing 2

In [5]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.phash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.phash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: 81cf6e90d1370dce
Hash Gambar 2: 81cf6e90d1370dce
Gambar mirip!
```

6. Implementasi Difference Hashing 1

In [6]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.dhash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.dhash(Image.open('D:/Gambar/Whalee.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: e0f1631ef0fc1fc0
Hash Gambar 2: f0703987f0c007f4
Gambar berbeda, perbedaan hash: 20
```

7. Implementasi Difference Hashing 2

In [7]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.dhash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.dhash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: e0f1631ef0fc1fc0
Hash Gambar 2: e0f1631ef0fc1fc0
Gambar mirip!
```

8. Implementasi Wavelet Hashing 1

In [8]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.whash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.whash(Image.open('D:/Gambar/Whalee.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

```
Hash Gambar 1: 10fef1efff0f0000
Hash Gambar 2: 087f8ff7fff00000
Gambar berbeda, perbedaan hash: 20
```

9. Implementasi Wavelet Hashing 2

In [9]:

```
#Import Library
from PIL import Image
import imagehash

#Membuat Objek hash gambar 1
HashGambar1 = imagehash.whash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 1: ' + str(HashGambar1))

#Membuat Objek hash gambar 2
HashGambar2 = imagehash.whash(Image.open('D:/Gambar/Whale.jpg'))
print('Hash Gambar 2: ' + str(HashGambar2))

#Membandingkan Objek Hash
if(HashGambar1 == HashGambar2):
    print("Gambar mirip!")
else:
    print("Gambar berbeda, perbedaan hash: " + str(HashGambar1 - HashGambar2))
```

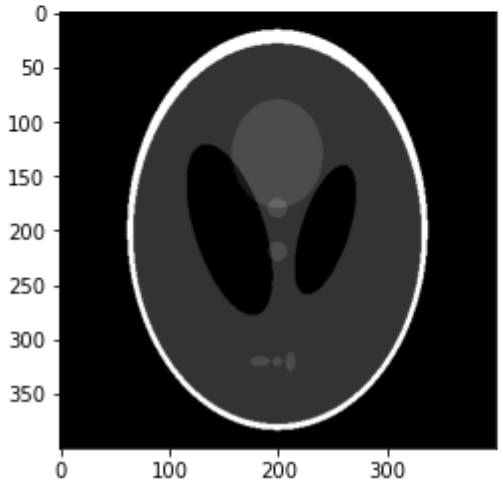
```
Hash Gambar 1: 10fef1efff0f0000
Hash Gambar 2: 10fef1efff0f0000
Gambar mirip!
```

Plot Morphologi

1. Memasang Library dan Membaca Data

In [2]:
```
# Impor Library
import matplotlib.pyplot as plt
from skimage import data
from skimage.util import img_as_ubyte

# Membaca Data
orig_phantom = img_as_ubyte(data.shepp_logan_phantom())
fig, ax = plt.subplots()
ax.imshow(orig_phantom, cmap=plt.cm.gray)
```

Out[2]:
```
<matplotlib.image.AxesImage at 0x1b9f7612a30>
```

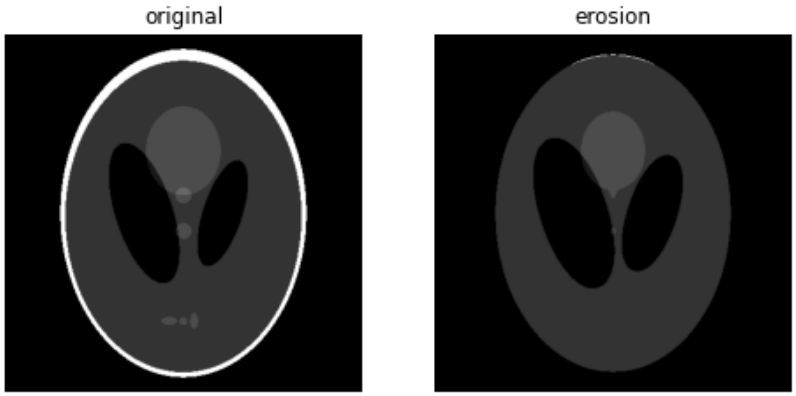


2. Membuat Fungsi Perbandingan Plot

In [3]:
```
# Membuat fungsi perbandingan plot
def plot_comparison(original, filtered, filter_name):

    fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(8, 4), sharex=True,
                                   sharey=True)
    ax1.imshow(original, cmap=plt.cm.gray)
    ax1.set_title('original')
    ax1.axis('off')
    ax2.imshow(filtered, cmap=plt.cm.gray)
    ax2.set_title(filter_name)
    ax2.axis('off')
```

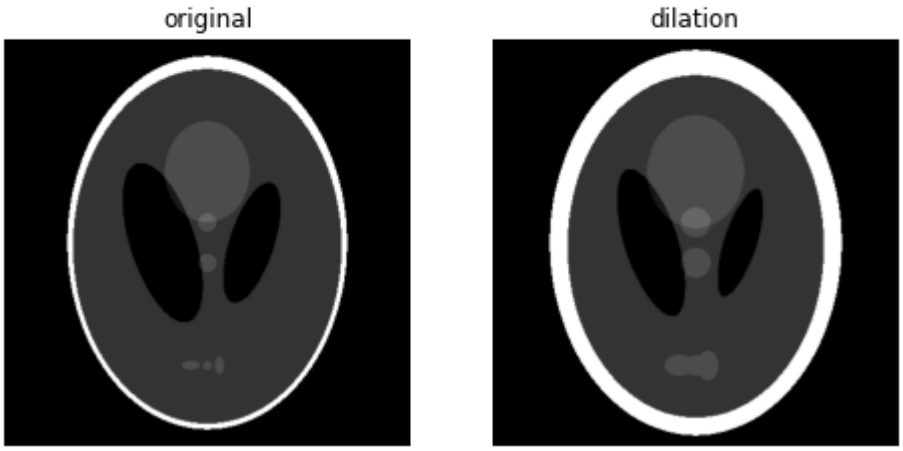
3. Implementasi Morfologi Erosi

In [4]:
```
# Import Library
from skimage.morphology import (erosion, dilation, opening, closing,
                                white_tophat)
from skimage.morphology import black_tophat, skeletonize, convex_hull_image
from skimage.morphology import disk

# Implementasi Morfologi Erosion
footprint = disk(6)
eroded = erosion(orig_phantom, footprint)
plot_comparison(orig_phantom, eroded, 'erosion')
```



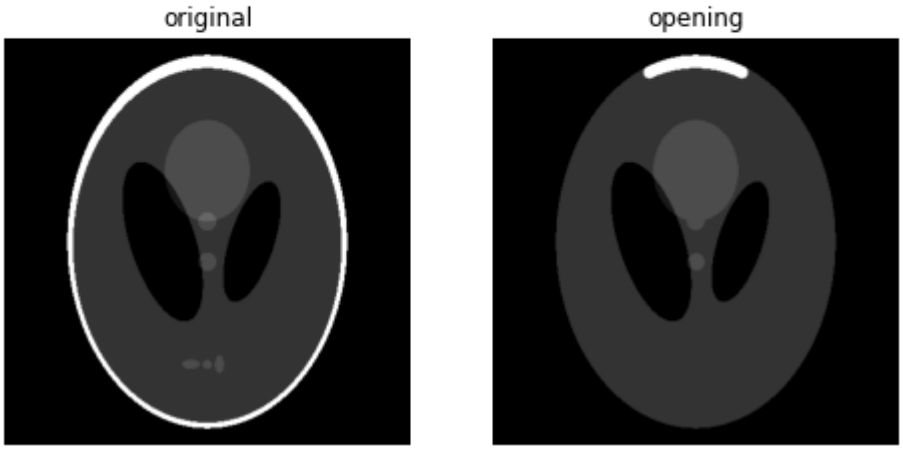
4. Implementasi Morfologi Dilation

In [5]:
```
# Implementasi Morfologi Dilation
dilated = dilation(orig_phantom, footprint)
plot_comparison(orig_phantom, dilated, 'dilation')
```



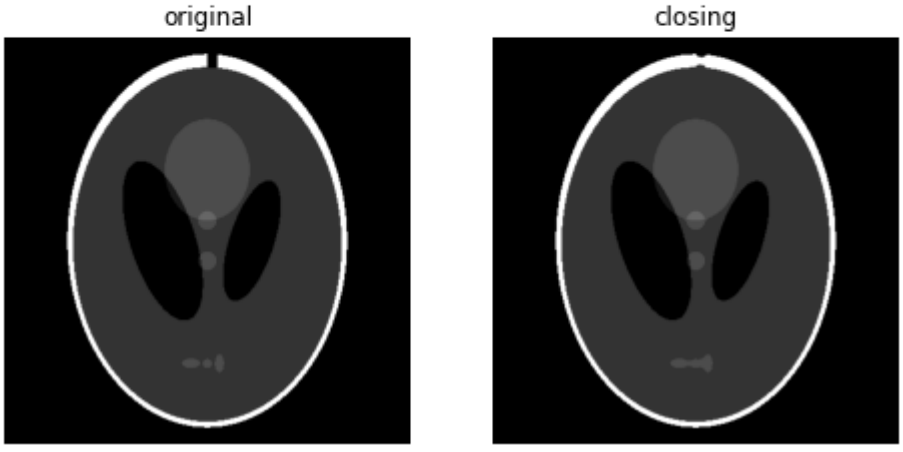
5. **Implementasi Morfologi Opening**

In [6]:
```
# Implementasi Morfologi Opening
opened = opening(orig_phantom, footprint)
plot_comparison(orig_phantom, opened, 'opening')
```



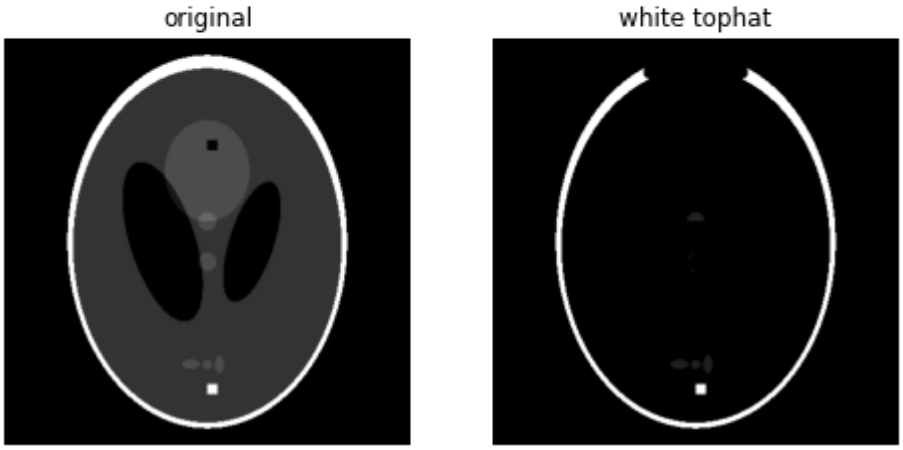
6. **Implementasi Morfologi Closing**

In [7]:
```
# Implementasi Closing
phantom = orig_phantom.copy()
phantom[10:30, 200:210] = 0

closed = closing(phantom, footprint)
plot_comparison(phantom, closed, 'closing')
```



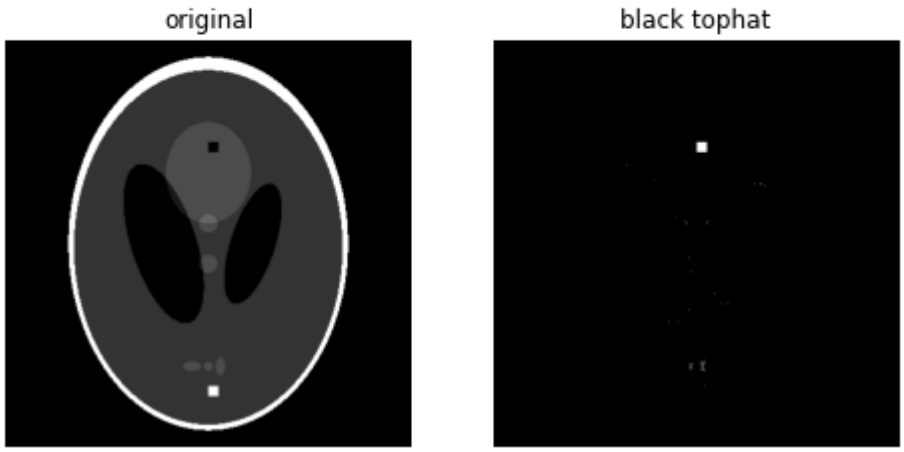
7. Implementasi Morfologi White Tophat

In [8]:
```
# Implementasi White Tophat
phantom = orig_phantom.copy()
phantom[340:350, 200:210] = 255
phantom[100:110, 200:210] = 0

w_tophat = white_tophat(phantom, footprint)
plot_comparison(phantom, w_tophat, 'white tophat')
```



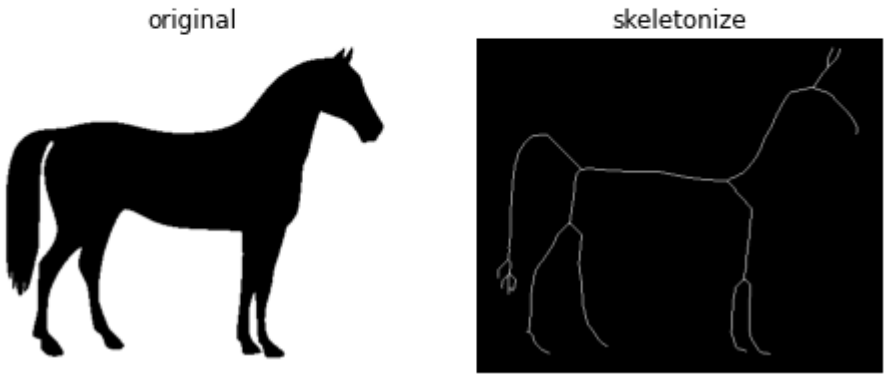
8. Implementasi Morfologi Black Tophat

In [9]:
```
# Implementasi Black Tophat
b_tophat = black_tophat(phantom, footprint)
plot_comparison(phantom, b_tophat, 'black tophat')
```



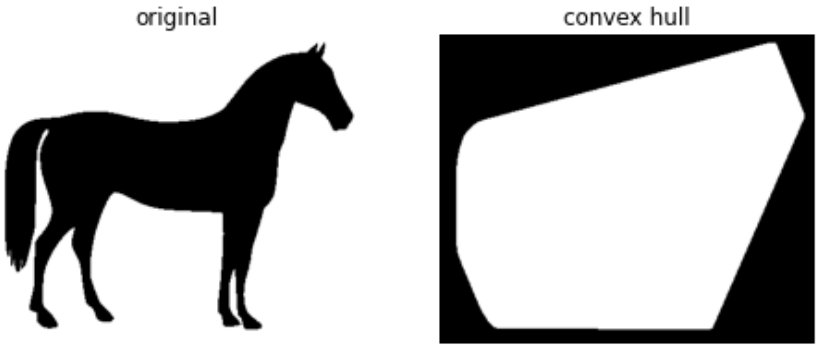
9. Implementasi Morfologi Skeletonize

In [10]:
```
# Implementasi Morfologi Skeletonize
horse = data.horse()

sk = skeletonize(horse == 0)
plot_comparison(horse, sk, 'skeletonize')
```



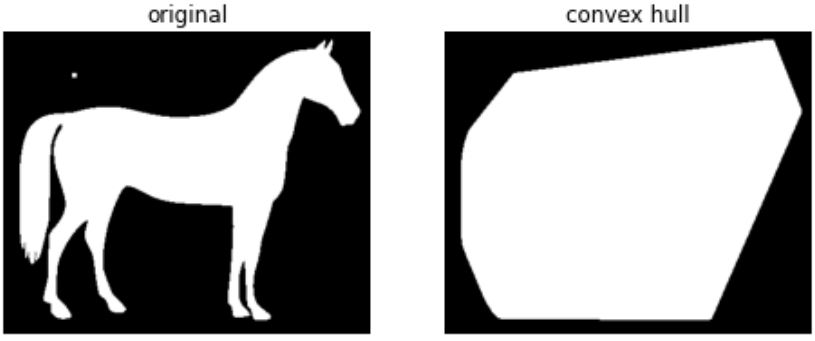
10. Implementasi Morfologi Convex Hull 1

In [11]:
```
# Implementasi Morfologi Convex Hull
hull1 = convex_hull_image(horse == 0)
plot_comparison(horse, hull1, 'convex hull')
```



11. Implementasi Morfologi Convex Hull 2

In [12]:
```
horse_mask = horse == 0
horse_mask[45:50, 75:80] = 1

hull2 = convex_hull_image(horse_mask)
plot_comparison(horse_mask, hull2, 'convex hull')
```



Plot ORB

1. **Memasang library yang dibutuhkan**

In [2]:
```
from skimage import data
from skimage import transform
from skimage.feature import (match_descriptors, corner_harris,
                             corner_peaks, ORB, plot_matches)
from skimage.color import rgb2gray
import matplotlib.pyplot as plt
```

2. **Membaca data**

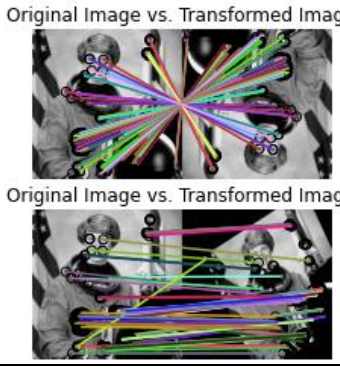
In [3]:
```
img1 = rgb2gray(data.astronaut())
img2 = transform.rotate(img1, 180)
tform = transform.AffineTransform(scale=(1.3, 1.1), rotation=0.5,
                                  translation=(0, -200))
img3 = transform.warp(img1, tform)
```

3. **Implementasi ORB**

In [4]:
```
descriptor_extractor = ORB(n_keypoints=200)
descriptor_extractor.detect_and_extract(img1)
keypoints1 = descriptor_extractor.keypoints
descriptors1 = descriptor_extractor.descriptors
descriptor_extractor.detect_and_extract(img2)
keypoints2 = descriptor_extractor.keypoints
descriptors2 = descriptor_extractor.descriptors
descriptor_extractor.detect_and_extract(img3)
keypoints3 = descriptor_extractor.keypoints
descriptors3 = descriptor_extractor.descriptors
matches12 = match_descriptors(descriptors1, descriptors2, cross_check=True)
matches13 = match_descriptors(descriptors1, descriptors3, cross_check=True)
```

4. **Visualisasi Hasil ORB**

In [5]:
```
fig, ax = plt.subplots(ncols=1, nrows=2, figsize=(12, 4))
plt.gray()
plot_matches(ax[0], img1, img2, keypoints1, keypoints2, matches12)
ax[0].axis('off')
ax[0].set_title("Original Image vs. Transformed Image")
plot_matches(ax[1], img1, img3, keypoints1, keypoints3, matches13)
ax[1].axis('off')
ax[1].set_title("Original Image vs. Transformed Image")
plt.show()
```



Plot Hough Transform

1. **Memasang library yang dibutuhkan**

```
In [2]:
import numpy as np
import matplotlib.pyplot as plt
from skimage import data, color
from skimage.transform import hough_circle, hough_circle_peaks
from skimage.feature import canny
from skimage.draw import circle_perimeter
from skimage.util import img_as_ubyte
```

2. **Membaca Data**

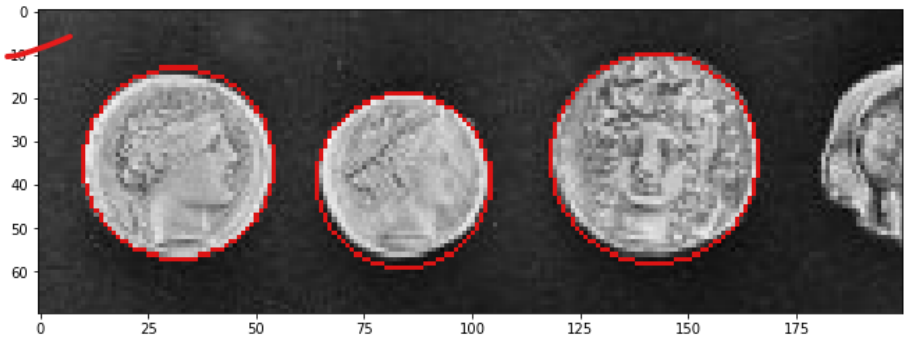
```
In [3]:
image = img_as_ubyte(data.coins()[160:230, 70:270])
```

3. **Implementasi Hough Transform Circle Detection**

```
In [4]:
edges = canny(image, sigma=3, low_threshold=10, high_threshold=50)
hough_radii = np.arange(20, 35, 2)
hough_res = hough_circle(edges, hough_radii)
accums, cx, cy, radii = hough_circle_peaks(hough_res, hough_radii,
                                           total_num_peaks=3)
```

4. **Visualisasi hasil Hough Transform Circle Detection**

```
In [5]:
fig, ax = plt.subplots(ncols=1, nrows=1, figsize=(12, 4))
image = color.gray2rgb(image)
for center_y, center_x, radius in zip(cy, cx, radii):
    circy, circx = circle_perimeter(center_y, center_x, radius,
                                    shape=image.shape)
    image[circy, circx] = (220, 20, 20)
ax.imshow(image, cmap=plt.cm.gray)
plt.show()
```



5. **Memasang library Ellipse Detection**

```
In [6]:
import matplotlib.pyplot as plt
from skimage import data, color, img_as_ubyte
from skimage.feature import canny
from skimage.transform import hough_ellipse
from skimage.draw import ellipse_perimeter
```

6. Membaca Data Ellipse

In [7]:
```
image_rgb = data.coffee()[0:220, 160:420]
image_gray = color.rgb2gray(image_rgb)
```

7. Implementasi Hough Transform Ellipse Detection

In [8]:
```
edges = canny(image_gray, sigma=2.0,
              low_threshold=0.55, high_threshold=0.8)
result = hough_ellipse(edges, accuracy=20, threshold=250,
                       min_size=100, max_size=120)
result.sort(order='accumulator')

# Menentukan parameter Ellipse
best = list(result[-1])
yc, xc, a, b = [int(round(x)) for x in best[1:5]]
orientation = best[5]

# Menunjukkan bentuk Ellipse pada gambar asli
cy, cx = ellipse_perimeter(yc, xc, a, b, orientation)
image_rgb[cy, cx] = (0, 0, 255)

# Menunjukkan bentuk Ellipse pada gambar hasil
edges = color.gray2rgb(img_as_ubyte(edges))
edges[cy, cx] = (250, 0, 0)
```

8. Visualisasi Hasil Ellipse Detection

In [9]:
```
fig2, (ax1, ax2) = plt.subplots(ncols=2, nrows=1, figsize=(12, 4),
                                sharex=True, sharey=True)
ax1.set_title('Original picture')
ax1.imshow(image_rgb)
ax2.set_title('Edge (white) and result (red)')
ax2.imshow(edges)
plt.show()
```