Politechnika Świętokrzyska

Laboratorium

Przetwarzanie obrazów i systemy wizyjne

Ćwiczenie 8

Wykorzystanie transformacji Radona i Hougha.

Cel ćwiczenia

Celem ćwiczenia jest nabycie umiejętności wykorzystywania transformacji Rodona i Hougha do wykrywania linii w obrazie.

dr inż Robert Kazała

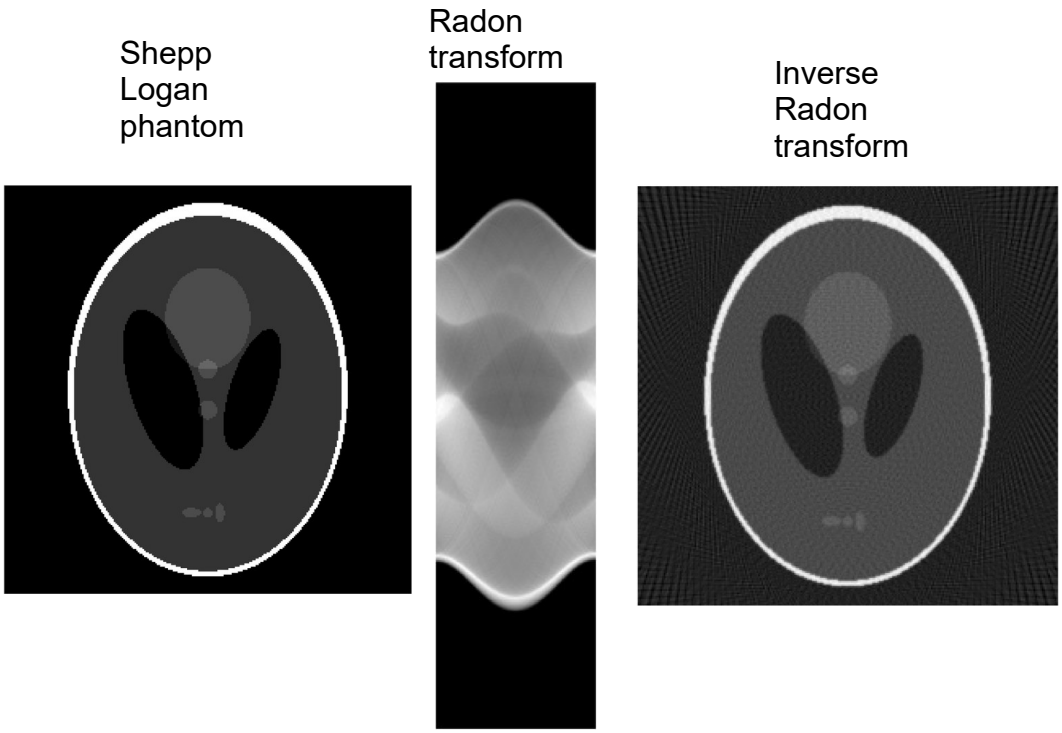
Wprowadzenie

Sinogram jest tablicą danych uzyskanych podczas tomografii komputerowej (np. PET). Uzyskuje się go poprzez zapisanie w wierszach tabeli kolejnych wyników projekcji (każda projekcja wykonywana jest pod innym kątem). Nazwa 'sinogram' wywodzi się od sinusoidalnego wyglądu zmian intensywności przedstawionych w tej tablicy. Sinogram nie jest wykorzystywany do diagnostyki, na jego podstawie w wyniku procesu rekonstrukcji uzyskiwany jest obraz przekroju badanego obiektu. Do odtwarzania wykorzystywana jest odwrotna transformata Radona.

Sinogram mozna utworzyć dla dowolnego zarejestrowanego obrazu wykonując transformację Radona. W celu rekonstrukcji obrazu wykorzystywane jest odwrotne przekształcenie Radona.

Obrazy testowe - Shepp-Logan phantom

W celu ułatwienia rozwoju i zapewnienia możliwości porównania algorytmów rekonstrukcji wykorzystuje się specjalne obrazy testowe o znanych właściwościach. Jednym z najczęściej wykorzystywanych jest Shepp-Logan phantom przypominający przekrój ludzkiej głowy , zawierający elipsy z różnymi stopniami pochłaniania.



Transformacja Radona

Funkcje modułu scikits-image realizujące prostą i odwrotną transformatę Radona.

radon

```
skimage.transform.radon(image, theta=None)
```

Funkcja wyliczająca transformatę Radona obrazu dla określonych kątów projekcji.

Parametry wejściowe:

`image` – obraz wejściowy

`theta` – kąty projekcji w stopniach (domyslnie np.arange(180))

Wyjście :

`output` – transformata Radona (sinogram)

iradon

```
skimage.transform.iradon(radon_image, theta=None,
output_size=None, filter='ramp', interpolation='linear')
```

Funkcja wyznaczająca odwrotną transformatę Radona, wykorzystująca algorytm filtrowanej projekcji wstecznej (ang. filtered back projection algorithm)

Parametry wejściowe:

`radon_image` – obraz zawierający sinogram, w którym każda kolumna odpowiada projekcji wzdłuż innego kąta,

`theta` – kąty pod jakimi dokonywana jest rekonstrukcja, domyślnie od 1 do 180 stopni

`output_size` – liczba wierszy i kolumn w rekonstruowanym obrazie,

`filter` – filtr stosowany do filtrowania w dziedzinie częstotliwości, dostępne filtry to ramp, shepp-logan, cosine, hamming, hann,

`interpolation` – metoda interpolacji stosowana w rekonstrukcji, dostępne metody to nearest, linear.

Wyjście :

`output` – zrekonstruowany obraz

frt2

```
skimage.transform.frt2(a)
```

Funkcja wyliczająca 2-wymiarową skończoną transformatę Radona (FRT) dla tablicy liczb całkowitych o rozmiarze nxn.

Parametry wejściowe:

a – tablica liczb całkowitych o wymiarach nxn

Wyjście:

tablica zawierająca współczynniki transformaty o wymiarach (n+1) x n.

ifrt2

```
skimage.transform.ifrt2(a)
```

Funkcja wyliczająca 2-wymiarową odwrotną skończoną transformatę Radona dla tablicy o wymiarach (n+1) x n.

Parametry wejściowe:

a – tablica wejściowa o wymiarach (n+1) x n

Wyjście :

output - tablica o wymiarach nxn

Przykład 1

Przykład pokazujący wykorzystanie transformat Radona.

```
import matplotlib.pyplot as plt

from skimage.io import imread
from skimage.color import rgb2gray
from skimage import data_dir
from skimage.transform import radon, iradon

image_color = imread(data_dir + "/phantom.png")
image = rgb2gray(image_color)

#image = rescale(image, scale=0.4)

plt.figure(figsize=(8, 8.5))
plt.subplot(221)
plt.title("Original");
plt.imshow(image, cmap=plt.cm.Greys_r)

plt.subplot(222)
projections = radon(image, theta=[0, 45, 90])
plt.plot(projections);
plt.title("Projections at\n0, 45 and 90 degrees")
plt.xlabel("Projection axis");
plt.ylabel("Intensity");
```

```
projections = radon(image)
plt.subplot(223)
plt.title("Radon transform\n(Sinogram)");
plt.xlabel("Projection axis");
plt.ylabel("Intensity");
plt.imshow(projections)

reconstruction = iradon(projections)
plt.subplot(224)
plt.title("Reconstruction\nfrom sinogram")
plt.imshow(reconstruction, cmap=plt.cm.Greys_r)

plt.subplots_adjust(hspace=0.4, wspace=0.5)
plt.show()
```

Transformacja Hougha

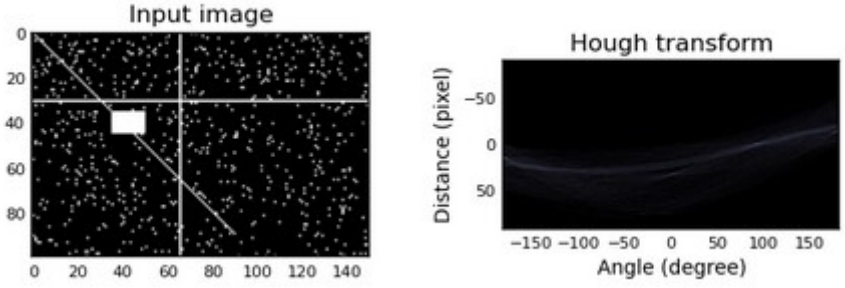
Transformacja Hougha (ang. Hough transform) jest szczególnym przypadkiem transformaty Radona i wykorzystywana jest do wykrywania regularnych kształtów w obrazach. Oryginalna metoda Hougha służy do wykrywania prostych. Metodę tę później uogólniono na wykrywanie kształtów dających się opisać analitycznie np. okręgów oraz na wykrywanie dowolnych kształtów.

Funkcje modułu scikit-image realizujące transformatę Hougha:

- skimage.transform.hough_line
- skimage.transform.hough_circle
- skimage.transform.hough_ellipse

Przykład 2

Przykład pokazujący wykorzystanie transformaty Hougha



```
import numpy as np
import matplotlib.pyplot as plt

from skimage.transform import hough_line

img = np.zeros((100, 150), dtype=bool)
img[30, :] = 1
img[:, 65] = 1
img[35:45, 35:50] = 1
for i in range(90):
    img[i, i] = 1

img += np.random.random(img.shape) > 0.95
out, angles, d = hough_line(img)

plt.subplot(1, 2, 1)
plt.imshow(img, cmap=plt.cm.gray)
plt.title('Input image')

plt.subplot(1, 2, 2)
plt.imshow(out, cmap=plt.cm.bone,
           extent=(d[0], d[-1],
                   np.rad2deg(angles[0]), np.rad2deg(angles[-1])))
plt.title('Hough transform')
plt.xlabel('Angle (degree)')
plt.ylabel('Distance (pixel)')

plt.subplots_adjust(wspace=0.4)
plt.show()
```

Przykład 3

Przykład pokazujący wykorzystanie transformaty Hougha do detekcji linii.

```
import numpy as np

from skimage.transform import hough_line, hough_line_peaks
from skimage.feature import canny
```

```
from skimage.draw import line
from skimage import data

import matplotlib.pyplot as plt
from matplotlib import cm


# Constructing test image
image = np.zeros((200, 200))
idx = np.arange(25, 175)
image[idx, idx] = 255
image[line(45, 25, 25, 175)] = 255
image[line(25, 135, 175, 155)] = 255

# Classic straight-line Hough transform
# Set a precision of 0.5 degree.
tested_angles = np.linspace(-np.pi / 2, np.pi / 2, 360, endpoint=False)
h, theta, d = hough_line(image, theta=tested_angles)

# Generating figure 1
fig, axes = plt.subplots(1, 3, figsize=(15, 6))
ax = axes.ravel()

ax[0].imshow(image, cmap=cm.gray)
ax[0].set_title('Input image')
ax[0].set_axis_off()

angle_step = 0.5 * np.diff(theta).mean()
d_step = 0.5 * np.diff(d).mean()
bounds = [np.rad2deg(theta[0] - angle_step),
          np.rad2deg(theta[-1] + angle_step),
          d[-1] + d_step, d[0] - d_step]
ax[1].imshow(np.log(1 + h), extent=bounds, cmap=cm.gray, aspect=1 / 1.5)
ax[1].set_title('Hough transform')
ax[1].set_xlabel('Angles (degrees)')
ax[1].set_ylabel('Distance (pixels)')
ax[1].axis('image')

ax[2].imshow(image, cmap=cm.gray)
```

```
ax[2].set_ylim((image.shape[0], 0))
ax[2].set_axis_off()
ax[2].set_title('Detected lines')

for _, angle, dist in zip(*hough_line_peaks(h, theta, d)):
    (x0, y0) = dist * np.array([np.cos(angle), np.sin(angle)])
    ax[2].axline((x0, y0), slope=np.tan(angle + np.pi/2))

plt.tight_layout()
plt.show()
```

Literatura

Lyons R. G.: Wprowadzenie do cyfrowego przetwarzania sygnałów, WKŁ, Warszawa 1999.

Oppenheim A. V., Schafer R. W.: Cyfrowe przetwarzanie sygnałów, WKŁ, Warszawa 1976.

Tadeusiewicz R., Korohoda P.: Komputerowa analiza i przetwarzanie obrazów, Społeczeństwo Globalnej Informacji, Kraków 1997.

Zadania

1. Uruchomić przykład transformaty Radona znajdujący się w instrukcji.
2. Uruchomić przykłady transformaty Hougha znajdujący się w instrukcji.
3. Dla przykładu transformaty Radona znajdującego się w instrukcji dokonać transformaty i rekonstrukcji dla różnych wartości kątów projekcji.
4. Wykonać transformację Rodona dla własnych obrazów.
5. Wykonać transformację Hougha dla własnych obrazów.
6. Wykonać transformację Hougha dla obrazów zawierających okręgi z wykorzystaniem skimage.transform.hough_circle.
7. Dla własnych obrazów testowych wykorzystać transformację Hougha do wyznaczenia linii w obrazie.