Uniwersytet Rzeszowski

Wydział Nauk Ścisłych i Technicznych

Instytut Informatyki

BIOMETRYCZNE SYSTEMY ZABEZPIECZEŃ

INSTRUKCJA DO ĆWICZEŃ LABORATORYJNYCH

TREŚCI KSZTAŁCENIA: TRANSFORMATA HOUGHA

Spis treści

1. Cele laboratorium

Celem zajęć jest zapoznanie się z transformacją Hough'a w zadaniu wykrywania linii i okręgów w testowych obrazach binarnych oraz obrazach tonowanych pozyskiwanych ze skanerów tęczówki oka.

2. Wprowadzenie

Transformata Hougha

Klasyczna transformata Hougha stosowana przy rozpoznawaniu linii prostych odwzorowuje przestrzeń kartezjańską w tzw. przestrzeń Hougha, której każdy punkt jest obrazem prostej w przestrzeni kartezjańskiej. Jedną ze współrzędnych tego punktu jest odległość prostej od środka układu, drugą kąt nachylenia normalnej prostej do osi x. Odpowiednikiem punktu w przestrzeni kartezjańskiej w transformacie Hougha jest pewna krzywa. Tworzą ją punkty będące obrazami wszystkich prostych przechodzących przez ten punkt.

Transformatę Hougha można także zastosować do znajdowania bardziej skomplikowanych kształtów niż linie proste np. okręgów.

1. Transformata Hougha – wersja liniowa

Transformata Hougha liniowa opiera się na znalezieniu parametrów (a, b) prostych przechodzących przez punkt (x, y). Prosta opisana równaniem 1 stanowi podstawę przekształcenia:

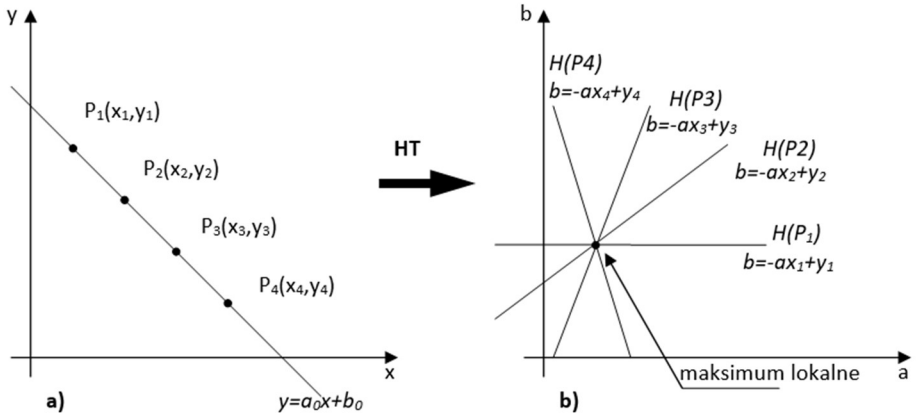
$$y = ax + b \tag{1}$$

gdzie: x, y – współrzędne punktu w przestrzeni obrazu, a, b – parametry opisujące linię prostą.

Równanie 1 opisuje zależność pomiędzy punktami obrazu (x, y) leżącymi na prostej a parametrami (a, b) opisującymi tą prostą w przestrzeni obrazu. Wynika stąd, że obrazem prostej z przestrzeni obrazu jest punkt (a, b) w przestrzeni parametrów (dla transformaty Hougha zwanej również przestrzenią Hougha). Równanie 1 przedstawia również jedno z możliwych odwzorowań punktów obrazu z przestrzeni obrazu w przestrzeń parametrów. Po przekształceniu równania 1 otrzymujemy następującą postać:

$$b = -ax + y \tag{2}$$

Zgodnie z powyższym równaniem transformacja punktu (x, y) odpowiada obliczeniu wszystkich wspólnych parametrów (a, b) linii prostych przechodzących przez ten punkt. Proste w przestrzeni Hougha przecinają się w jednym punkcie, którego współrzędne stanowią parametry (a, b) dla linii prostej łączącej punkty na obrazie. Zobrazowano to na Rys. 1.



Rys. 1. Transformacje punktów obrazu leżących na tej samej prostej do przestrzeni parametrów a) punkty w przestrzeni obrazu, b) proste w przestrzeni parametrów przecinające się w punkcie o parametrach (a, b).

Przestrzeń parametrów przedstawiana jest za pomocą tablicy zwanej macierzą akumulatorów. W stanie początkowym wszystkie elementy tablicy są zerami. Akumulator jest licznikiem zwiększanym o jeden, gdy linia w przestrzeni (a, b) przechodzi przez dany akumulator. Rys. 2 przedstawia przykładowy akumulator. Aby znaleźć parametry (a, b) dla punktów leżących na tej samej prostej, należy w macierzy akumulatorów znaleźć lokalne maksimum.

8										
7										
6	1									
5	2	2	1							
4	1	2	2	2				1	1	1
3	1	1	3	4	**[6]**	4	3	1	1	1
2	1	1				2	2	2	1	1
1							1	2	2	1
0									1	1
-1										1
	-1.8	**-1.6**	**-1.4**	**-1.2**	**-1**	**-0.8**	**-0.6**	**-0.4**	**-0.2**	**0**

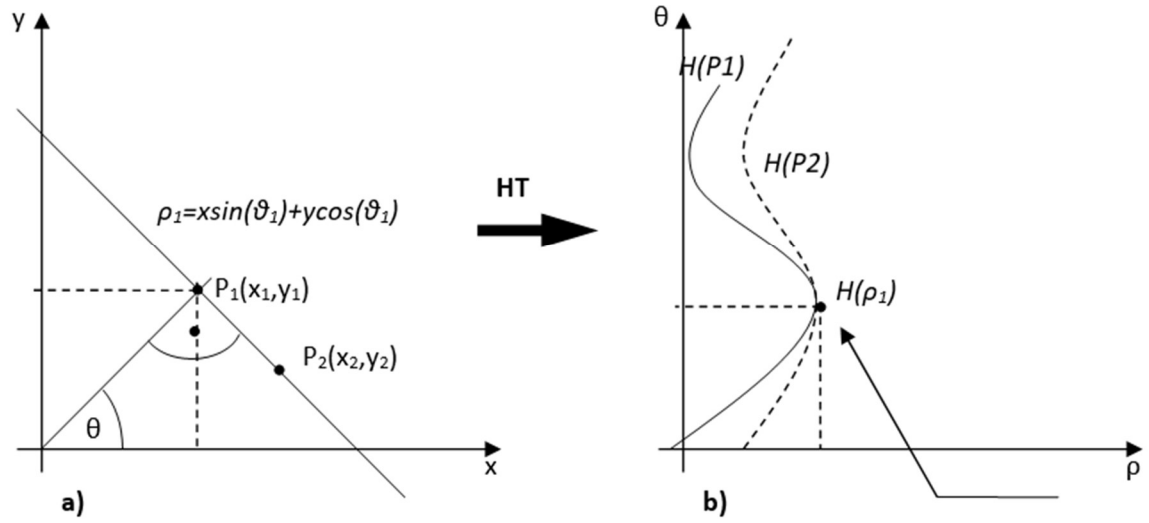
Rys. 2. Tablica akumulatorów, maksimum lokalne oznaczone nawiasem kwadratowym.

Takie same efekty można uzyskać wychodząc z postaci normalnej równania prostej, przedstawionego równaniem 3:

$$\rho = xsin(\theta) + ycos(\theta) \tag{3}$$

gdzie: θ – przyjmuje zazwyczaj wartości $[0, 180]$

W wyniku zastosowania równania 3 punktowi (x, y) z przestrzeni obrazu odpowiada sinusoida w przestrzeni Hougha (ρ, θ), poszukiwana prosta opisana jest parametrami (ρ, θ) punktu przecięcia się tych sinusoid. Obrazem prostej z przestrzeni obrazu jest punkt w przestrzeni (ρ, θ). Rys. 3 przedstawia powyższe zależności.



Rys. 3. a) Prosta opisana równaniem 3. b) obraz tej prostej w przestrzeni parametrów (ρ, θ).

Na rysunku 3a. widoczny jest przykładowy obraz w przestrzeni kartezjańskiej złożony z punktów $P1$ i $P2$ leżących na jednej prostej oraz na rysunku 3.b. odpowiadająca temu obrazowi przestrzeń parametrów. Reprezentacją punktów $P1$ i $P2$ z przestrzeni kartezjańskiej są sinusoidy $H(P1)$ i $H(P2)$ w przestrzeni parametrów. Maksimum lokalne oznaczone $H(\rho 1)$ w przestrzeni parametrów daje poszukiwane wartości $\rho 1$ i $\theta 1$. Wartości te służą do wyznaczenia poszukiwanej prostej opisanej równaniem 3.

Funkcje Matlaba oznaczają

- size(w)- ustala wymiar wektora w i zwraca liczbę wierszy i kolumn,
- round - zaokrąglanie do najbliższej liczby całkowitej,
- zeros(m,n) - funkcja zwracająca macierz z elementami zero, o rozmiarach m – wierszy na n - kolumn.

Przykładowy skrypt realizujący wersję liniową tej transformaty zawiera poniższy fragment kodu:

```
% Transformata Hougha-wersja liniowa, liczba prostych oznacza ilość poszukiwanych maksimów.
% ========================================================================================
liczba_max = 3;
%wczytanie obrazu bitmapy
[X, map] = imread('trojkat.bmp');
% przekształcanie struktury indeksowanej do postaci binarnej
bw = im2bw(X, map);
[nr, nc] = size(bw);
figure(1)   % wskazane okno kontekstu graficznego
imshow(bw)  % wizualizacja krawędzi
axis([0 nc 0 nr])
axis xy  %skalowanie osi
pause(1);

% wymiar akumulatora akum[romax,tetamax],
% teta przyjmuje wartosci od 0 do teta_max=180,
% co kilka stopni = rozdzielczosc
ro_max = nc;     %opcjonalnie: round(sqrt(2)*length(bw));

rozdz_teta = 1; teta_max = 180;
il_teta = round(teta_max/rozdz_teta);
akum = zeros(ro_max, il_teta);
```

```
%obliczenie wzoru 3
for x = 1 : nc
    for y = 1 : nr
        if (bw(x, y) == 0)
            for teta = 1 : rozdz_teta : teta_max
            % miara w radianach wersja transformaty promień/kąt
            ro = round(x*cos(teta*pi/180) + y*sin(teta*pi/180));
                if (ro<ro_max & ro>0)
                    akum(ro, teta) = akum(ro, teta) + 1;
                end
            end
        end
    end
end

% Opcjonalnie zapis tablicy akumulatorowej
% save akum.mat akum

figure(2)
imagesc(akum)
xlabel('ro')
ylabel('teta')
grid on

% Wyszukiwanie lokalnych maksimów równych ilości zadanych linii lmax
% Mile widziana samodzielna konstrukcja wykrywająca zadaną liczbę maksimów
% w przestrzeni akumulatorowej Hougha
peaks= houghpeaks(akum, liczba_max);

%wyrysowanie wyodrębnionych prostych
figure(3)
imshow(bw)
hold on
for k = 1 : liczba_max
    x_line=1: 1 : nc;
        rad=(peaks(k,2)*pi)/180;
        y_line= -x_line*((cos(rad))/sin(rad))+peaks(k,1)/sin(rad);
        plot(y_line, x_line, 'g')
        axis([0 ro_max 0 ro_max])
        xlabel('x')
        ylabel('y')
end
grid on
hold off
```

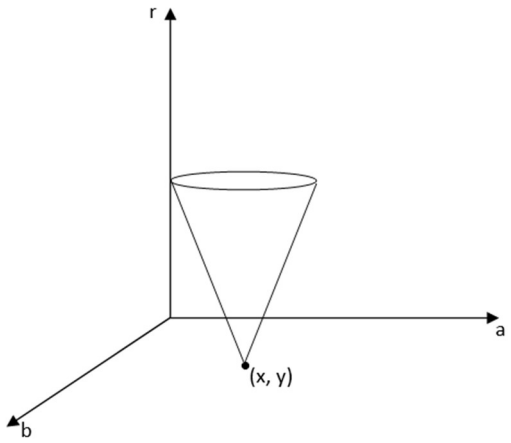
2. Transformata Hougha – Wersja Kołowa

Transformatę Hougha można stosować również do detekcji położenia i promienia okręgu na płaszczyźnie:

$$(x-a)^2 + (y-b)^2 = r^2 \tag{4}$$

gdzie: a, b – oznaczają współrzędne środka okręgu, r – promień okręgu, x, y– współrzędne w przestrzeni obrazu.

Przestrzeń parametrów w tym przypadku jest przestrzenią trójwymiarową, stąd wynika, że akumulator jest macierzą trójwymiarową. Każdy punkt (x, y) z przestrzeni obrazu generuje stożek w przestrzeni parametrów (a, b, r), jak pokazano na Rys. 4.



Rys. 4. Przykładowy wynik otrzymany dla obrazu testowego

W detekcji okręgu należy przyjąć dopuszczalny minimalny i maksymalny promień wykrywanych okręgów:

$$r = [r_{min} \dots r_{max}] \tag{5}$$

Zwiększane będą w przestrzeni parametrów punkty (a, b, r), które leżą na powierzchni stożka. Dla ustalonego promienia r w przestrzeni parametrów (a, b) otrzymamy okrąg. W innej postaci:

$$x = a + rcos(\theta) \tag{6}$$

$$y = b + rsin(\theta) \tag{7}$$

stąd po przekształceniu otrzymujemy:

$$a = x - rcos(\theta) \tag{8}$$

$$b = y - rsin(\theta) \tag{9}$$

Zatem mechanizm zliczania wystąpień w tablicy akumulatorowej jest taki sam jak dla wersji liniowej tej transformaty.

Wyodrębnianie maksimów

Zakładając, że wszystkie parametry z powyższej zależności są znane, oprócz parametru a, zatem do obliczenia tego parametru zależności można przyjąć następującą postać:

$$(x - a)^2 = r^2 - (y - b)^2 \tag{10}$$

$$\sqrt{(x - a)^2} = \sqrt{r^2 - (y - b)^2} \tag{11}$$

$$a = x \mp \sqrt{r^2 - (y - b)^2} \tag{12}$$

Największa wartość punktu w danej warstwie oznacza również, że znajduje się tam środek okręgu, w tym samym wierszu i kolumnie, co szukany punkt.

Algorytm transformacji Hougha dla okręgów składa się z poniższych etapów:

1. Wstępne przetwarzanie obrazu
   - Konwersja do skali szarości (jeśli wymagane).
   - Zastosowanie filtru krawędziowego (np. operatora Canny'ego).
   - Opcjonalne zastosowanie filtru Gaussa do redukcji szumów.
2. Inicjalizacja akumulatora

- Tworzymy trójwymiarową macierz akumulacyjną (a, b, r), gdzie każda komórka odpowiada możliwemu okręgowi.

3. Głosowanie w przestrzeni parametrów*
   - Dla każdego punktu krawędziowego (x, y) oraz dla wybranych promieni r:
     - Obliczamy możliwe środki okręgu (a, b) rozwiązując równanie okręgu.
     - Zwiększamy wartość akumulatora w pozycjach (a, b, r).
4. Wykrywanie maksymalnych wartości w akumulatorze
   - Wybieramy współrzędne (a, b, r), dla których akumulator osiąga największe wartości.
   - Te wartości odpowiadają wykrytym okręgom.
5. Rysowanie wykrytych okręgów na obrazie
   - Zaznaczamy okręgi na oryginalnym obrazie, rysując okrąg o promieniu r wokół wykrytych środków.

***** Głosowanie w przestrzeni parametrów to kluczowy etap transformacji Hougha, w którym każdy wykryty punkt krawędziowy przyczynia się do akumulacji głosów w zbiorze możliwych parametrów kształtu (dla okręgów są to środek (a, b) i promień r).

Działanie głosowanie w przestrzeni parametrów w transformacie Hougha dla okręgów:

Przestrzeń parametrów

Dla każdego punktu krawędziowego (x, y) rozważamy wszystkie możliwe okręgi, które mogą przez niego przechodzić. Przestrzeń parametrów składa się z: a, b – możliwe współrzędne środka okręgu, r – możliwe promienie okręgu.

Równanie okręgu

Okrąg można opisać równaniem:

$$(x - a)^2 = r^2 - +(y - b)^2 \tag{13}$$

Dla danego r i punktu krawędziowego (x, y) obliczamy możliwe wartości (a, b):

$$a = x \mp \sqrt{r^2 - (y - b)^2} \tag{14}$$

lub

$$b = x \mp \sqrt{r^2 - (x - a)^2} \tag{15}$$

Ponieważ dla każdego punktu krawędziowego wiele możliwych środków (a, b) może spełniać to równanie, dlatego wyznaczenie tablicy akumulatorowe Hougha w tym przypadku oznacza iterowanie w trój wymiarowej przestrzeni parametrów.

Akumulator (tablica głosowania)

Tworzymy trójwymiarową tablicę akumulatora $A(a, b, r)$, gdzie każdy indeks przechowuje liczbę głosów dla danego środka (a, b) i promienia r.

Dla każdego punktu krawędziowego (x, y) i każdego r obliczamy możliwe (a, b).

Zwiększamy wartość $A(a, b, r)$ – to znaczy, że dany punkt głosuje na ten okrąg.

Wybór najlepszych wyników

Po zakończeniu głosowania przeszukujemy akumulator w celu znalezienia jego maksimów – wartości (a, b, r) o najwyższych liczbach głosów. Te wartości odpowiadają najbardziej prawdopodobnym okręgom w obrazie.

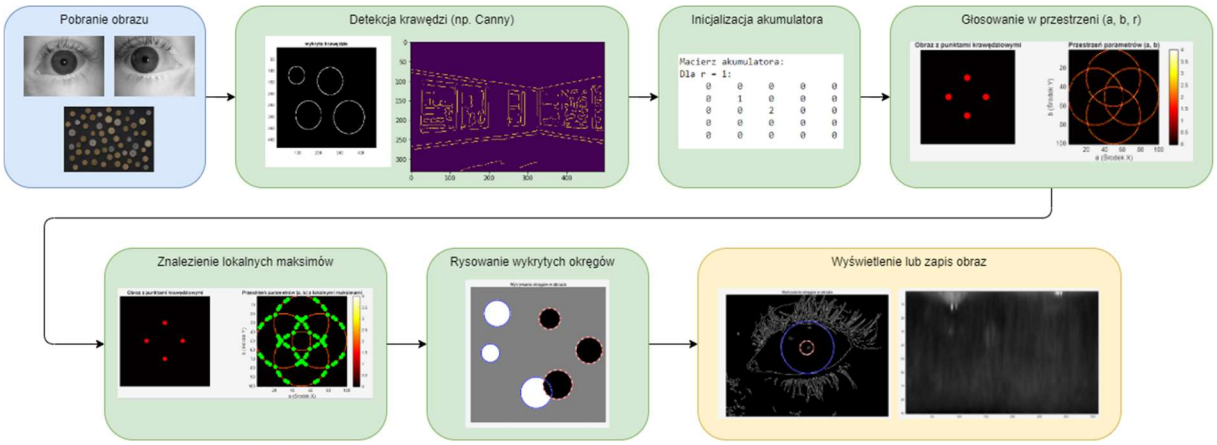
Przykład głosowania

Wyobraźmy sobie, że mamy obraz z białym okręgiem na czarnym tle. Po wykryciu krawędzi dostajemy zbiór punktów na brzegu okręgu. Każdy z tych punktów głosuje na możliwe wartości (a, b, r), a po zakończeniu głosowania najczęściej wybierane wartości w akumulatorze wskazują rzeczywisty okrąg.

Optymalizacja algorytmu

Transformata Hougha dla okręgów jest kosztowna obliczeniowo ze względu na trzeci wymiar (promień r). Aby zredukować liczbę obliczeń można ograniczyć zakres badanych promieni jak również można użyć metody przybliżonej np. Transformata Hougha z gradientem. Wersje algorytmu opierające się na metodach probabilistycznych (np. Randomized Hough Transform) mogą znacznie przyspieszyć obliczenia.

Na Rys. 5 przedstawiono schemat algorytmu transformacji Hougha dla okręgów



Rys. 5. Schemat algorytmu transformacji Hougha

3. Zadania do samodzielnego rozwiązania

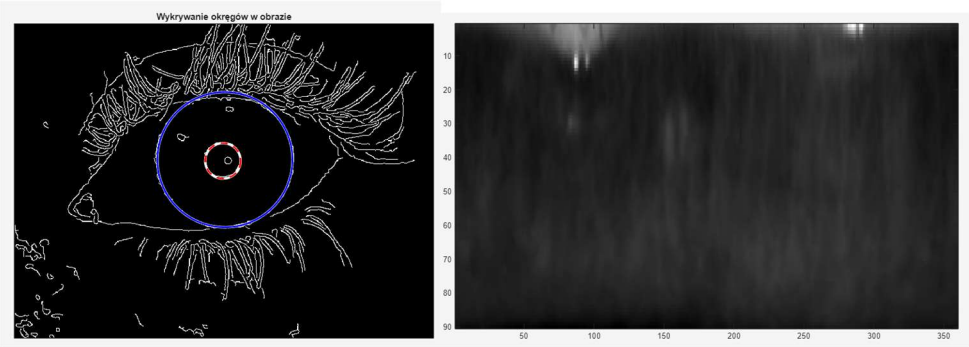
Zadanie 1. Wykrywanie i normalizacja tęczówki oka przy użyciu transformacji Hougha

Zaproponuj implementację algorytmu wykrywania tęczówki w obrazie oka przy wykorzystaniu transformacji Hougha w wersji kołowej. Następnie należy przeprowadzić normalizację wykrytej tęczówki poprzez przekształcenie jej do współrzędnych biegunowych.

W tym celu należy:

1. Wczytać obraz oka i jeżeli jest to konieczne zmienić go do skali szarości tzw. *gray level*.
2. Przeprowadzić wykrywanie krawędzi: w tym celu zastosuj operator Canny'ego do wykrycia krawędzi. Dostosuj próg detekcji krawędzi, aby uzyskać możliwie wyraźne kontury.
3. Wykonać wykrywanie okręgów metodą Hougha: w tym celu wykorzystaj funkcję `imfindcircles` do wykrycia granicy między twardówką a tęczówką (większy okrąg), granicy między tęczówką a źrenicą (mniejszy okrąg). Ustaw odpowiednie zakresy promieni dla obu okręgów oraz zwizualizuj wykryte okręgi na obrazie.
4. Przeprowadzić transformację wyznaczonego pola tęczówki do postaci macierzowej: w tym celu przeprowadź przekształcenie wykrytej tęczówki do współrzędnych biegunowych. Ustaw odpowiednie parametry dla tej transformacji (promień podziału, zakres kąta). Zwizualizuj przekształcony obraz w nowych współrzędnych biegunowych.
5. Dokonać wizualizacji wyników: w tym celu wyświetl wykryte okręgi na oryginalnym obrazie oraz wyświetl obraz tęczówki w układzie biegunowym.

Przykładowe wyniki, które mogą być uzyskane w trakcie prowadzonych symulacji:



Rys. 6. Przykładowe wyniki: wyodrębnione pole tęczówki w obrazie oraz jego rozwinięta postać we współrzędnych biegunowych

Poniżej zamieszczono przykład kodu, który można wykorzystać w celu zaimplementowania algorytmu wykrywania tęczówki w obrazie oka przy wykorzystaniu transformacji Hougha w wersji kołowej.

```
%Program demonstrujący wykorzystanie wersji kołowej transformaty Hougha
%w zadaniu wyodrębniania pola tęczówki
%=========================================================================
clear all, close all, clc

oko = imread('oko2.jpg'); %lub oko1
oko= mat2gray(oko);
imshow(oko)
% Wszystkie wystąpienia ???? należy uzupełnić samodzielnie dobranymi
% parametrami transformaty
Rmin = ????;  %twardówka - tęczówka
Rmax = ????;
[srodki_j, promienie_j] = imfindcircles(oko,[Rmin Rmax],'ObjectPolarity',??????, ...
    'Sensitivity', ?????);

Rmin = ????;  %źrenica - tęczówka
Rmax = ????;
[srodki_c, promienie_c] = imfindcircles(im2bw(oko, 0.12),[Rmin Rmax],'ObjectPolarity',...
    'dark','Method', ??????, 'Sensitivity', ??????);

viscircles(srodki_j, promienie_j,'Color','b'); %Wizualizacja okręgów
viscircles(srodki_c, promienie_c,'LineStyle','--'); %Wizualizacja okręgów
hold on
title('Wykrywanie okręgów w obrazie');
hold off
% parametry normalizacji
promien_rozdz = 90;
kat_zakres = 360;
% Rozwijanie tęczówki do współrzędnych biegunowych
[polar_tab, szum_tab] = polar_transform(oko, ...
    srodki_c(1), srodki_c(2), promienie_c, ...
    srodki_j(1), srodki_j(2), promienie_j, ...
    promien_rozdz, kat_zakres);
figure
imagesc(polar_tab)
colormap(gray)
```

Zadanie 2. Automatyczne zliczanie monet na obrazie z wykorzystaniem transformacji Hougha

Celem zadania jest automatyczne wykrycie, zliczenie oraz określenie nominałów monet na obrazie wejściowym za pomocą transformacji Hougha dla okręgów. Należy także oszacować złożoność obliczeniową metody oraz przetestować ją na różnych zbiorach obrazów. Zapoznaj się z poniższym kodem i zaproponuj brakujące rozwiązania.

```
% Program pomocniczy demonstrujący w jaki sposób może zostać wykonane zliczanie monet z
% wykorzystaniem wersji kołowej transformaty Hougha
% ========================================================================
clear all, close all, clc
% Załaduj obraz zawierający przykładowe okręgi -> plik dostępny w zasobach pakietu Matlab
A = imread('circlesBrightDark.png');
imshow(A)

Rmin = 15;  % dobór tych parametrów ma kluczowe znaczenie dla
Rmax = 95;  % dalszej pracy algorytmu

[srodki_j, promienie_j] = imfindcircles(A,[Rmin Rmax],'ObjectPolarity','bright');

[srodki_c, promienie_c] = imfindcircles(A,[Rmin Rmax],'ObjectPolarity','dark');

viscircles(srodki_j, promienie_j,'Color','b'); %Wizualizacja okręgów

viscircles(srodki_c, promienie_c,'LineStyle','--'); %Wizualizacja okręgów
hold on
title('Wykrywanie okręgów w obrazie');
hold off
% ========================================================================
% Na podstawie kodu powyżej zliczyć monety w zbiorach monety1-4.jpg :)
% ========================================================================

A = imread('monety4.jpg');
A= A(380:1262,711:1657);
A= imresize(A,0.5);
imshow(A>55)
A = edge(A, 'canny');

% dla jpg-ów może być potrzebna konwersja do gray
% A= (A(:,:,1)+A(:,:,2)+A(:,:,3))./3;
% lub A= mat2gray(A);

Rmin = 30; % Zakres promieni poszukiwanych okręgów
Rmax = 55;

[srodki, promienie] = imfindcircles(A,[Rmin Rmax], 'ObjectPolarity','bright');
if length(srodki)>0
    viscircles(srodki, promienie,'Color','b'); %Wizualizacja okręgów
    najw_moneta= find(promienie==max(promienie)); % indeks największej monety
    najmn_moneta= find(promienie==min(promienie)); % indeks najmniejszej monety
    viscircles(srodki(najw_moneta,:), promienie(najw_moneta) ,'Color','r'); %Wizualizacja
okręgów
    viscircles(srodki(najmn_moneta,:), promienie(najmn_moneta) ,'Color','y'); %Wizualizacja
okręgów
    hold on
    title(['Zliczanie monet w obrazie. Liczba monet to: ' num2str(length(srodki))]);
    hold off
end
% Mile widziany fragment zliczający sumę nominałów monet :)
% return

% ========================================================================
% Przetwarzanie zbioru monety1.jpg
% Szacowanie złożoności obliczeniowej
% ========================================================================
```

```
start= tic;
im = imread('monety1.jpg');
A=im;
A= (A(:,:,1)+A(:,:,2)+A(:,:,3))./3;
e = edge(A, 'canny', 0.6);
radii = 60:1:90;    %15:1:40;
h = circle_hough(e, radii, 'same', 'normalise');
peaks = circle_houghpeaks(h, radii, 'nhoodxy', 11, 'nhoodr', 7, 'npeaks', 90);
imshow(im);
hold on;
for peak = peaks
    [x, y] = circlepoints(peak(3));
    plot(x+peak(1), y+peak(2), 'g-');
end
hold off
stop= toc(start);
```