7.1 Houghova transformácia

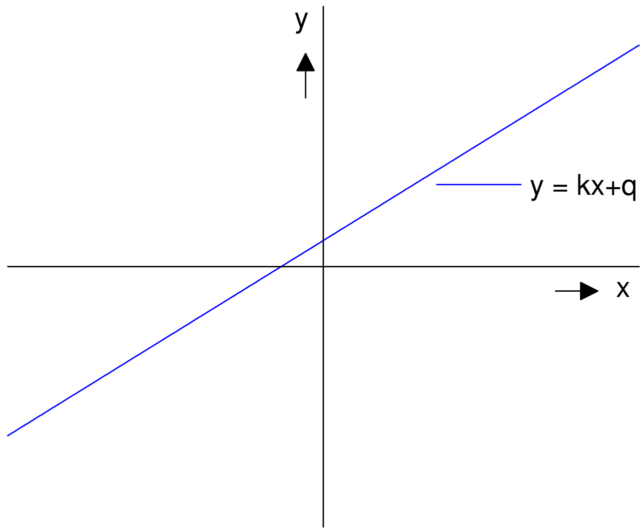
Houghova transformácia sa používa na detekciu parametricky opísateľných tvarov a kriviek v obrazovej analýze, počítačovom videní a digitálnom spracovaní obrazu. Účelom tejto transformácie je nájsť objekty, teda analytický opis hľadaných tvarov a kriviek v obraze. Tento proces prebieha formou hlasovania v parametrickom priestore hľadaných kriviek na základe najväčšej tvarovej zhody. Pôvodná Houghova transformácia bola navrhnutá na identifikovanie priamok v obraze. Túto metódu vyvinul a patentoval *Paul Hough* v roku 1962, po ktorom aj získala meno. Odvtedy sa upravovala a neskôr bola rozšírená na identifikáciu všeobecných tvarov a kriviek. Najčastejšie sa používa na detekciu kruhov a elíps.

Teória – Houghova transformácia

Rovnica priamky v smernicovom tvare má vo všeobecnosti tvar:

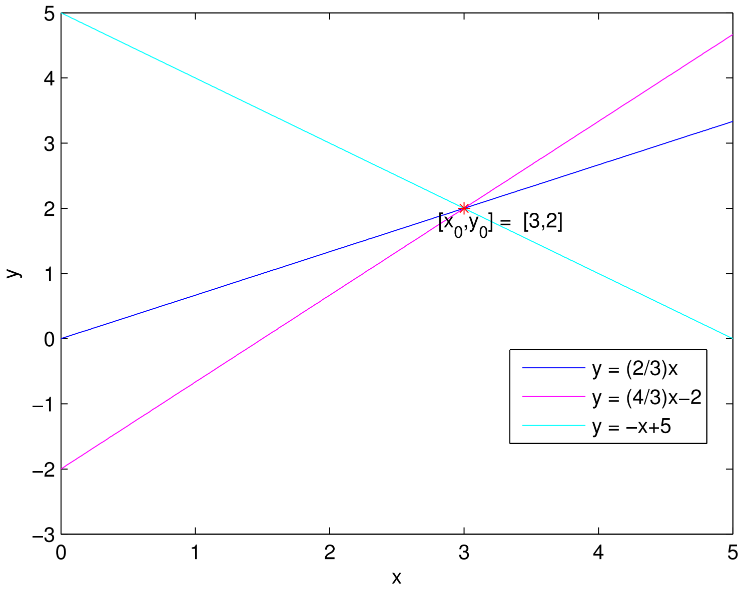
$$y = kx + q \tag{7.5}$$

kde *k* je smernica priamky a *q* je priesečník s osou *y*.



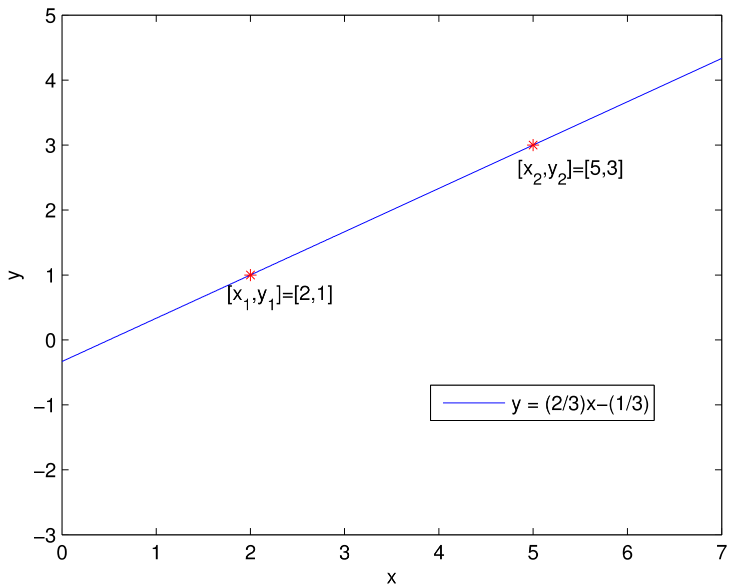
Obr. 7.1: Smernicový tvar priamky

Zvoľme si bod v priestore $[x_0,y_0]$, ktorým chceme aby priamka prechádzala. Riešením sú všetky priamky, ktorých parametre (k, q) spĺňajú rovnicu $y_0 = kx_0 + q$ (Obr. 7.1).



Obr. 7.1: Priamky prechádzajúce bodom $[x_0,y_0]$

Takýchto riešení, je nekonečne veľa. Na jednoznačné definovanie priamky potrebujeme minimálne dva body. Zvoľme si teda body $[x_1,y_1]$ a $[x_2,y_2]$, ktoré jednoznačne definujú priamku (Obr. 7.2).

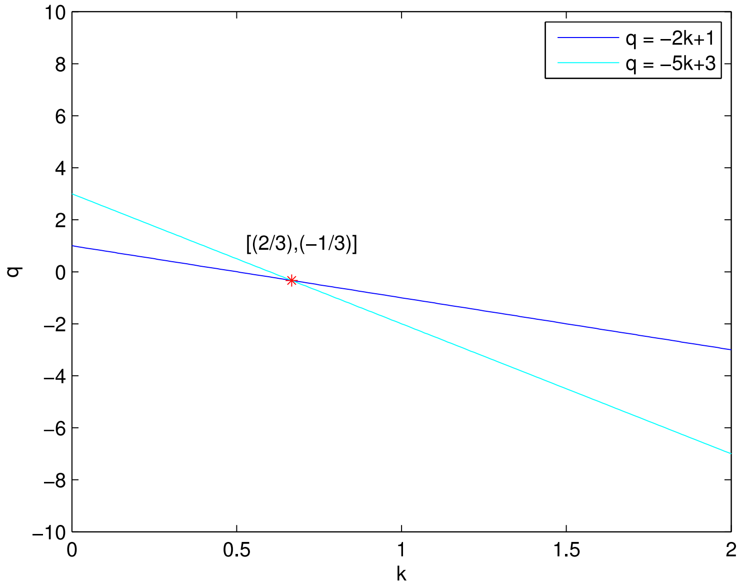


Obr. 7.2: Priamka definovaná bodmi $[x_1,y_1],[x_2,y_2]$

Rovnicu priamky vieme taktiež vyjadriť" v tvare:

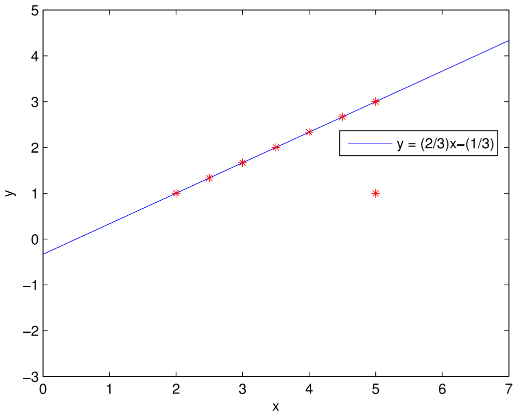
$$q = -xk + y \tag{7.6}$$

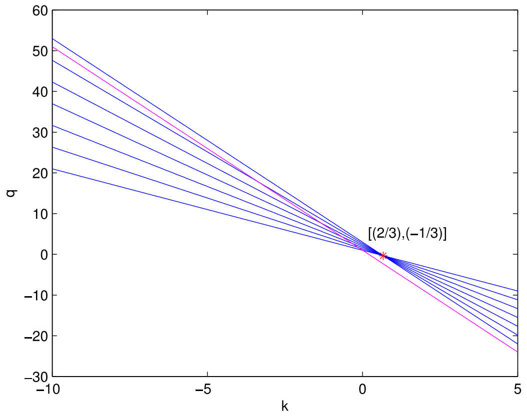
kde uvažujeme, že (x, y) sú konštanty a (q, k) sú parametre danej priamky. Na danú úpravu sa môžeme pozerať ako na transformáciu priamky z priestoru (x, y) do priestoru parametrov (q, k). Na obrázku Obr. 7.3 je priamka definovaná bodmi $[x_1,y_1],[x_2,y_2]$ v priestore parametrov.



Obr. 7.3: Priamky definované bodmi $[x_1,y_1],[x_2,y_2]$ v priestore parametrov

Priesečník priamok v parametrickom priestore označuje spoločné parametre (k, q), pre ktoré platí, že priamka s týmito parametrami prechádza oboma bodmi $[x_1,y_1]$ a $[x_2,y_2]$. V prípade, že existuje viacero bodov v priestore, ktoré ležia na jednej priamke, tak vznikne v priestore parametrov viacnásobný priesečník pre každý takýto bod (Obr. 7.4,7.6).





Obr. 7.4: Viacero bodov ležiacich na priamke

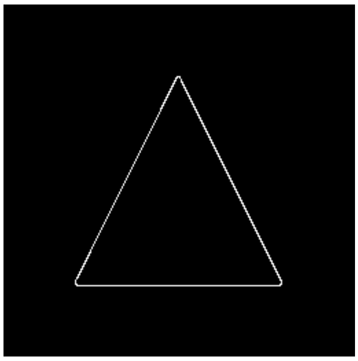
Obr. 7.6: Priestor parametrov pre viacero bodov

Na obrázku 7.6 je vidieť, že v priestore parametrov sa vytvorilo toľko priesečníkov, koľko je možno vytvoriť priamok pomocou daných bodov v priestore. Červená priamka v parametrickom priestore reprezentuje bod, ktorý neleží na priamke v obrázku Obr. 7.4.

Z týchto poznatkov vyplýva, že ak chceme detekovať priamku, musíme nájsť čo najviac bodov obrazu, ktoré spĺňajú jej predpis a majú viacnásobný priesečník v priestore parametrov.

Pri prehľadávaní bodov obrázka sa potrebujeme obmedziť na body, ktoré reprezentujú hrany, resp. prechody medzi jednotlivými objektami. Práve tieto body obrázka ležia na hľadanej priamke, ktorú chceme nájsť" a majú viacnásobný priesečník v parametrickom priestore.

Táto úprava sa najčastejšie realizuje prechodom obrázka cez hranový filter (najčastejšie Cannyho, Sobelov a podobne). Na obrázku 7.7.7 a 7.7.8 je obrázok pred a po prechode Cannyho hranovým filtrom.



Obr. 7.7: Obrázok pred prechodom Cannyho filtrom Obr. 7.8: Obrázok po prechode Cannyho filtrom

Hľadané priamky nájdeme pomocou priesečníkov v parametrickom priestore, kde každá dvojica priamok reprezentuje jeden hranový bod obrázka. Toto hľadanie je realizované formou hlasovania v akumulátore. Akumulátor je pomocný diskrétny priestor parametrov (*q, k*), ktorý je možné reprezentovať" maticou:

$$\begin{bmatrix} k_{min}, q_{min} & \cdots & k_{min}, q_{max} \\ \vdots & \vdots & \vdots \\ k_{max}, q_{min} & \cdots & k_{max}, q_{max} \end{bmatrix} \tag{7.7}$$

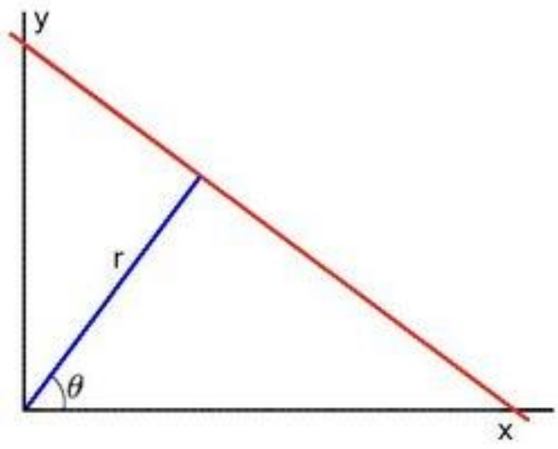
Pre každý hranový bod obrázka (*x, y*), potom inkrementujeme odpovedajúcu hodnotu parametrov (*k, q*) v akumulátore, ktoré vyhovujú rovnici $b = -xk + y$. Extrémy hodnôt akumulátora potom odpovedajú parametrom hľadaných priamok v obrázku. Problém smernicového vyjadrenia priamok však je, že pri zvislých priamkach hodnoty parametrov(*k, q*) divergujú k nekonečnu. Preto sa toto vyjadrenie používa len v prípadoch, kedy hľadané priamky na obrázku nie sú zvislé. Riešením tohto problému je použitie vyjadrenia priamky v polárnych súradniciach (Duda, Hart 1972):

$$y = -\frac{\cos\theta}{\sin\theta}x + \frac{r}{\sin\theta} \tag{7.8}$$

Úpravou dostaneme výraz:

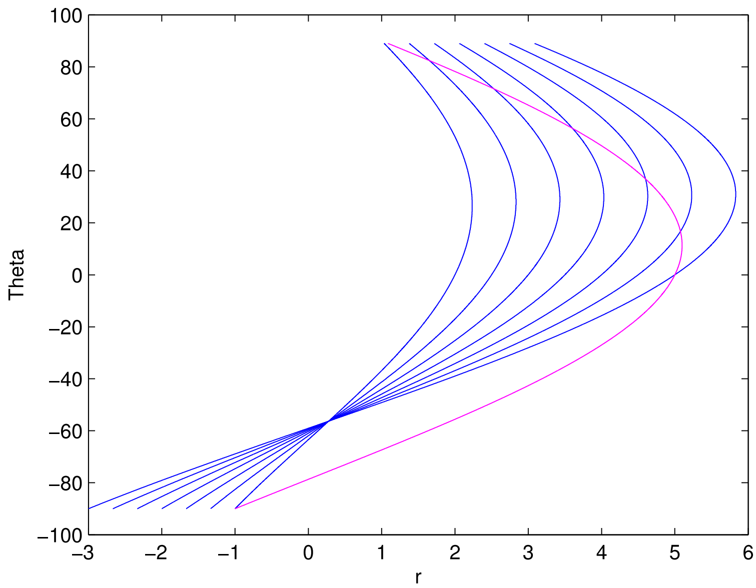
$$r = x\cos\theta + y\sin\theta \tag{7.9}$$

kde r je vzdialenosť" normály priamky k počiatku súradnicového systému a θ je uhol medzi kladnou časťou x-ovej osi a tejto normály (Obr. 7.9).



Obr. 7.9: Priamka definovaná v polárnych súradniciach

Výhoda tohto vyjadrenia spočíva v lineárnom a konečnom rozsahu parametrov θ a r. Teda nevznikajú problémy s ich divergenciou. Každému bodu obrázka bude teda prislúchať jedna sínusoida v priestore parametrov (θ, r), ktorý nazývame Houghov priestor. Príklad Houghovho priestoru bodov z obrázka Obr. 7.4 je na obrázku Obr. 10.



Obr. 10: Houghov priestor parametrov z obrázka 5

Akumulátor bude pre toto vyjadrenie v tvare:

$$\begin{bmatrix} r_{min}, \theta_{min} & \cdots & r_{min}, \theta_{max} \\ \vdots & \vdots & \vdots \\ \hline r_{max}, \theta_{min} & \cdots & r_{max}, \theta_{max} \end{bmatrix} \tag{7.10}$$

Pseudokód celého algoritmu by sme mohli napísať nasledovne:

Načítaj obrázok

Použi hranový filter

Inicializuj akumulátor na nulového hodnoty

for all x **do**

    for all y **do**

        if (x, y) je hranový bod **then**

            for all θ **do**

                r = xcosθ + ysinθ

                Inkrementuj hodnotu (θ, r) v akumulátore

            end for

        end if

    end for

end for

Príklad 1

S využitím Houghovej transformácie nájdite v obrázku 7.7.7 priamky tvoriace trojuholník.

Riešenie

Pre riešenie tejto úlohy sme napísali funkciu *Hough_priamka*, ktorá názorne prezentuje funkčnosť Houghovej transformácie v obrazovej analýze. Prvým krokom je načítanie obrázka trojuholníka podľa zadania, ktorý je vstupným argumentom našej vytvorenej funkcie. Funkcia má niekoľko parametrov, ktoré slúžia na optimalizáciu grafického a funkčného výstupu. Ich vysvetlenie je v popise danej funkcie. Jednoducho sa dá zobraziť v *Matlabe* príkazom: help Hough_priamka.

Popis funkcie po zadaní daného príkazu:

Funkcia na detekciu priamok v obraze

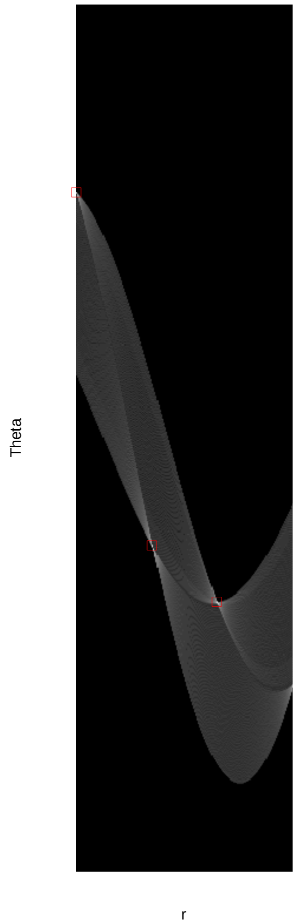
```
 Vstupne argumenty:
 img - obrazok pred prechodom hranovym filtrom
     - povinny parameter
 gamma - kontrast zobrazeneho Houghovho priestoru v rozsahu <0-1>
       - volitelny parameter (defaultna hodnota 0.4)

 citlivost - rozhodovacia hranica hlasovania v akumulatore v rozsahu <0-1>
           - vyssia hodnota znamena vyssiu citlivosti, teda viac najdenych
           - objektov v obraze
           - volitelny parameter (defaultna hodnota 0.7)
%
 Priklad spravneho pouzitia funkcie
 % nacitame nas obrazok
 I = imread('cube.png');
 [Akumulator,theta,r] = Hough_priamka(I, 0.4, 0.7);
%
```

Načítame obrázok a zadáme ho ako vstupný parameter našej funkcie:

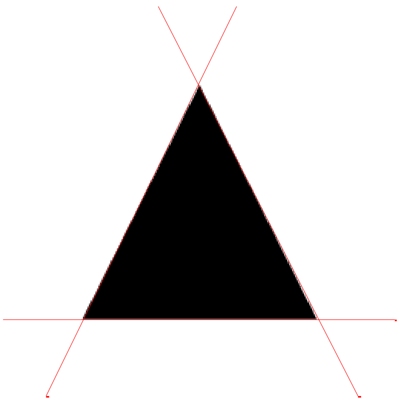
```
% nacitame nas obrazok
I = imread('triangle.png');
% zavolame nasu funkciu Houghovej transformacie pre detekciu priamok
[Akumulator,theta,r] = Hough_priamka(I, 0.4, 0.7);
```

Funkcia vypočíta formou hlasovania v akumulátore parametre hľadaných priamok, zobrazí akumulátor a vyznačí v ňom nájdené maximá, ktoré daným priamkam odpovedajú. Následne zobrazí nájdené priamky v danom obrázku. Akumulátor s vyznačenými maximami je na obrázku 7.7.11.



Obr. 7.11: Houghov priestor (akumulátor) pre obrázok trojuholníka

Po vykreslení nájdených priamok do obrázka dostaneme :



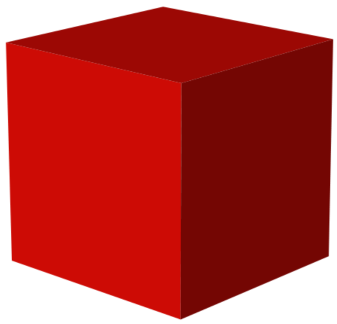
Obr. 12: Obrázok trojuholníka s nájdenými priamkami pomocou Houghovej transformácie

Príklad 2

S využitím Houghovej transformácie nájdite v obrázku kocky (Obr. 7.13) jej hrany a zvýraznite ich.

Riešenie

Pre riešenie tejto úlohy taktiež použijeme funkciu *Hough_priamka*. Prvým krokom je načítanie obrázka kocky podľa zadania, ktorý je vstupným argumentom našej vytvorenej funkcie.

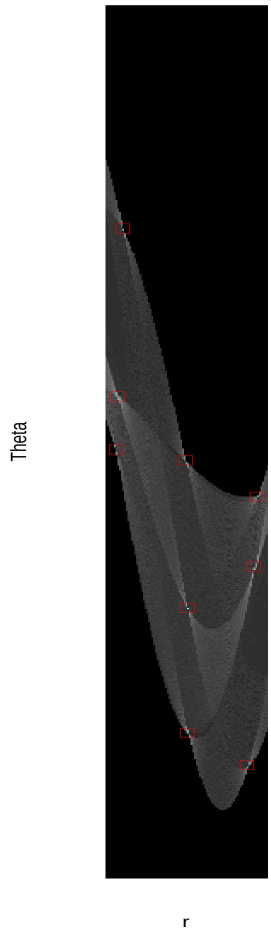


Obr. 7.13: Obrázok kocky

Zavoláme našu funkciu s iným argumentom:

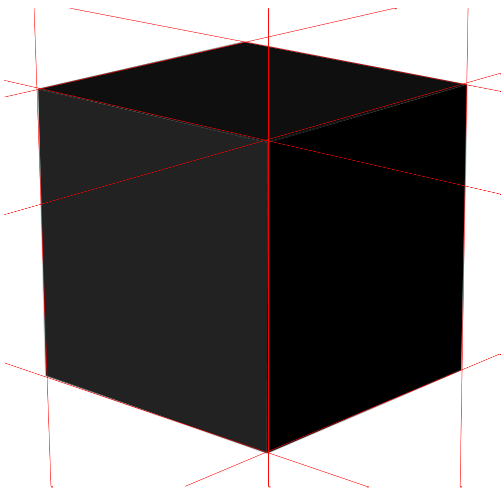
```
% nacitame nas obrazok
I = imread('cube.png');
% zavolame nasu funkciu Houghovej transformacie pre detekciu priamok
[Akumulator,theta,r] = Hough_priamka(I, 0.4, 0.7);
```

Výsledkom je opäť zobrazenie Houghovho priestoru s označenými maximami a obrázok kocky so zvýraznenými nájdenými priamkami, ktoré odpovedajú hranám kocky.



Obr. 7.14: Houghov priestor (akumulátor) pre obrázok kocky

Po vykreslí nájdených priamok do obrázka vidíme, že nájdené priamky skutočne odpovedajú hranám kocky (Obr. 7.15).



Obr. 7.15: Obrázok kocky so zvýraznenými hranami pomocou Houghovej transformácie

Teória – všeobecná Houghova transformácia

Houghovu transformáciu, ktorú poznáme dnes rozvinuli a upravili *Richard Duda* a *Peter Hart* v roku 1972. Nazvali ju všeobecná Houghova transformácia, pretože umožňuje detekciu všeobecných parametrických kriviek v obraze. Ide o rozšírenie pôvodnej Houghovej transformácie, tak aby sa dala použiť pre ľubovoľnú parametrickú krivku (najčastejšie kruh, elipsa).

Detekcia parabolických kriviek

Všeobecná rovnica paraboly má predpis:

$$y - y_0 = a(x - x_0)^2 \tag{7.11}$$

Oproti rovnici priamky, kde sme mali dve neznáme (θ, r) tentokrát nepoznáme 3 parametre : x_0, y_0 a a. Z toho vyplýva, že akumulátor bude trojrozmerný priestor a výpočtová náročnosť algoritmu sa zvýši. V praxi sa používa obmedzenie parametra a, ktorý určuje tvar paraboly v určitom odhadovanom rozsahu. Odhad je stále závislý na konkrétnej aplikácii, teda konkrétneho obrazu a hľadanej paraboly. Toto obmedzenie možno vynechať, čím sa ale výpočtová náročnosť algoritmu výrazne zvýši.

Algoritmus hľadania parabol môžeme zapísať pseudokódom :

```
Načítaj obrázok
Použi hranový filter
Inicializuj akumulátor na nulového hodnoty
for all x do
    for all y do
        if (x, y) je hranový bod then
            for all x0 do
                for all y0 do
                    a = (y-y0)/(x-x0)^2
                    Inkrementuj hodnotu (x0, y0, a) v akumulátore
                end for
            end for
```

```
            end if
        end for
end for
```

Príklad 3

S využitím všeobecnej Houghovej transformácie nájdite v obrázku (Obr. 7.16) paraboly a zvýraznite ich.

Obr. 7.16: Obrázok dáždnika

Riešenie

Pre riešenie tejto úlohy použijeme ďalšiu funkciu *Hough_parabola*. Funkcia má niekoľko parametrov, ktoré slúžia na optimalizáciu grafického a funkčného výstupu. Ich vysvetlenie je v popise danej funkcie. Jednoducho sa dá zobraziť v *Matlabe* príkazom: help Hough_parabola.

Popis funkcie po zadaní daného príkazu:

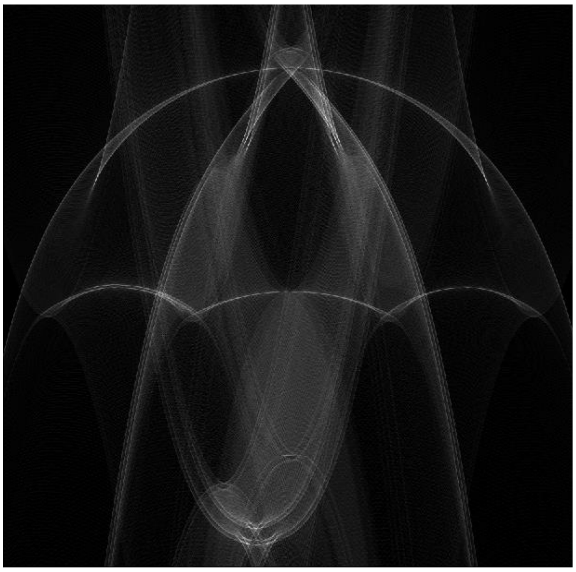
```
 Funkcia na detekciu parabol v obraze
 Vstupne argumenty:
 img - obrazok pred prechodom hranovym filtrom
     - povinny parameter
 a - vektor hodnot konstanty a (min:krok:max)
```

```
   - povinny parameter
 citlivost - rozhodovacia hranica hlasovania v akumulatore v rozsahu <0-1>
          - vyssia hodnota znamena vyssiu citlivosti, teda viac najdenych
          - objektov v obraze
          - volitelny parameter (defaultna hodnota 0.15)
%
 Priklad spravneho pouzitia funkcie
 % nacitame nas obrazok
 I = imread('dazdnik.jpg');
 % zavolame nasu funkciu Houghovej transformacie pre detekciu parabol
 [Akumulator] = Hough_parabola(I, 0.001:0.001:0.05, 0.15);
%
```

Prvým krokom je načítanie obrázka dáždnika podľa zadania, ktorý je vstupným argumentom našej vytvorenej funkcie:

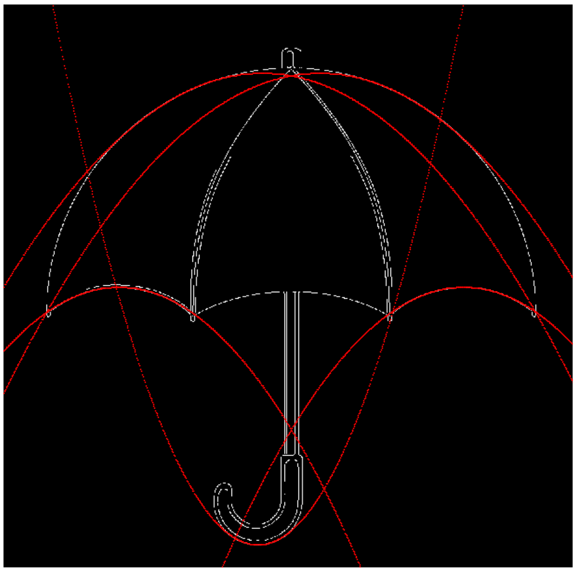
```
% nacitame nas obrazok
I = imread('dazdnik.jpg');
% zavolame nasu funkciu Houghovej transformacie pre detekciu parabol
[Akumulator] = Hough_parabola(I, 0.001:0.001:0.05, 0.15);
```

Vektor parametrov *a* sme obmedzili v rozsahu 0,001 až 0,05 s krokom 0,001. Taktiež sme zmenili parameter citlivosti pre potlačenie nevýrazných nájdených parabol v obrázku. Zadaním týchto príkazov dostaneme časť trojdimenzionálneho Houghovho priestoru parabolických kriviek (Obr. 7.17).



Obr. 7.17: Časť Houghovho priestoru parabolických kriviek

Zobrazenie nájdených parabolických kriviek je na obrázku 7.18. Môžeme vidieť, že sa algoritmu nepodarilo nájsť všetky paraboly, ktoré by človek očakával. Zmenu nájdených kriviek vieme dosiahnuť zmenou obmedzení parametra *a*, prípadne ponechaním jeho celého rozsahu. Alebo taktiež zmenou parametra citlivosti, ktorý filtruje len tie najvýraznejšie nájdené krivky v danom Houghovom priestore parametrov. Taktiež na počet a tvar nájdených kriviek vplýva kvalita samotného obrázka. V niektorých prípadoch je vhodné zmeniť citlivosť hranového filtra tak, aby boli výsledné hrany spojité a nepretŕhané.



Obr. 7.18: Obrázok dáždnika s nájdenými parabolickými krivkami

Vykreslené parabolické krivky však odpovedajú hranám v obrázku a teda algoritmus sa nám podarilo úspešne implementovať.

Detekcia kružníc

Všeobecná rovnica kružnice má predpis:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{7.12}$$

Rovnako ako pri parabole nepoznáme 3 parametre : x_0, y_0 a r. Z toho vyplýva, že akumulátor bude zase trojrozmerný priestor a výpočtová náročnosť algoritmu bude rovnaká ako pri parabolických krivkách. V praxi sa používa obmedzenie polomeru kružníc r, ktorý sa pohybuje v určitom odhadovanom rozsahu. Odhad je stále závislý na konkrétnej aplikácii, teda konkrétneho obrazu a veľkosti hľadaných kružníc. Toto obmedzenie možno vynechať, čím sa ale výpočtová náročnosť algoritmu výrazne zvýši.

Algoritmus hľadania kružníc môžeme zapísať pseudokódom :

Načítaj obrázok
Použi hranový filter
Inicializuj akumulátor na nulového hodnoty
for all x **do**
  for all y **do**

if (x, y) je hranový bod **then**

for all x_0 **do**

for all y_0 **do**

r = sqrt($(x-x_0)^2 + (y-y_0)^2$)

Inkrementuj hodnotu (x_0, y_0, r) v akumulátore

end for

end for

end if

end for

end for

Príklad 4

S využitím všeobecnej Houghovej transformácie nájdite v obrázku (Obr. 7.19) kružnice a zvýraznite ich.

Obr. 7.19: Obrázok mincí

Riešenie

Pre riešenie tejto úlohy použijeme ďalšiu funkciu *Hough_kruznica*. Funkcia má niekoľko parametrov, ktoré slúžia na optimalizáciu grafického a funkčného výstupu. Ich vysvetlenie je v popise danej funkcie. Jednoducho sa dá zobraziť v *Matlabe* príkazom: help Hough_kruznica.

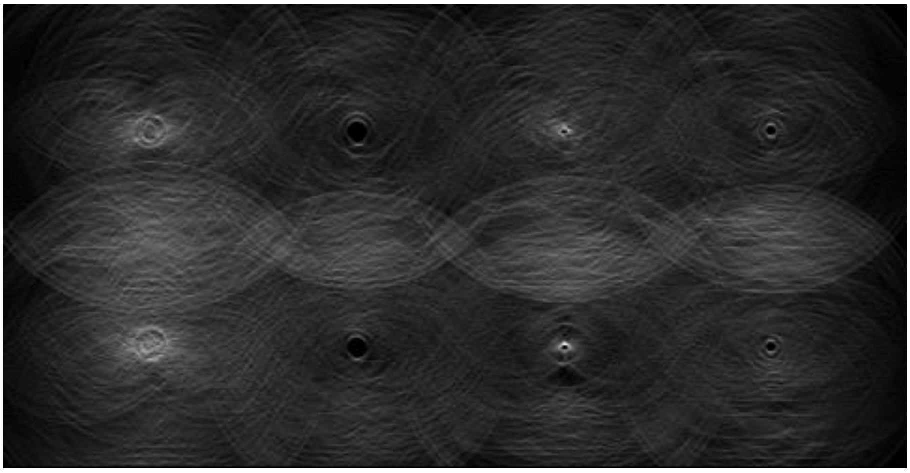
Popis funkcie po zadaní daného príkazu:

```
  Funkcia na detekciu kruznic v obraze
  Vstupne argumenty:
  img - obrazok pred prechodom hranovym filtrom
      - povinny parameter
  r - vektor hodnot polomerov kruznic (min:krok:max)
    - povinny parameter
  citlivost - rozhodovacia hranica hlasovania v akumulatore v rozsahu <0-1>
            - vyssia hodnota znamena vyssiu citlivosti, teda viac najdenych
            - objektov v obraze
            - volitelny parameter (defaultna hodnota 0.45)
 %
  Priklad spravneho pouzitia funkcie
  % nacitame nas obrazok
  I = imread('coins.jpg');
  % zavolame nasu funkciu Houghovej transformacie pre detekciu kruznic
  [Akumulator] = Hough_kruznica(I, 30:60, 0.45);
 %
```

Prvým krokom je načítanie obrázka mincí podľa zadania, ktorý je vstupným argumentom našej vytvorenej funkcie:

```
% nacitame nas obrazok
I = imread('coins.jpg');
% zavolame nasu funkciu Houghovej transformacie pre detekciu kruznic
[Akumulator] = Hough_kruznica(I, 30:60, 0.45);
```

Vektor polomerov *r* sme obmedzili v rozsahu 30 až 60 s krokom 1. Taktiež sme zmenili parameter citlivosti pre potlačenie nevýrazných nájdených kružníc v obrázku. Zadaním týchto príkazov dostaneme časť trojdimenzionálneho Houghovho priestoru kružníc (Obr. 7.20).



Obr. 7.20: Časť Houghovho priestoru pretínajúcich sa kružníc

Zobrazenie nájdených kružníc je na obrázku 7.21. Môžeme vidieť, že sa algoritmu podarilo nájsť všetky kružnice reprezentujúce mince. Zmenu nájdených kružníc vieme dosiahnuť zmenou obmedzení polomeru *r*, prípadne ponechaním jeho celého rozsahu.

Obr. 7.21: Obrázok mincí s nájdenými kružnicami

Vykreslené kružnice odpovedajú hranám v obrázku a teda algoritmus sa nám podarilo úspešne implementovať.

Detekcia elipsy ako detekcia poloosí symetrie

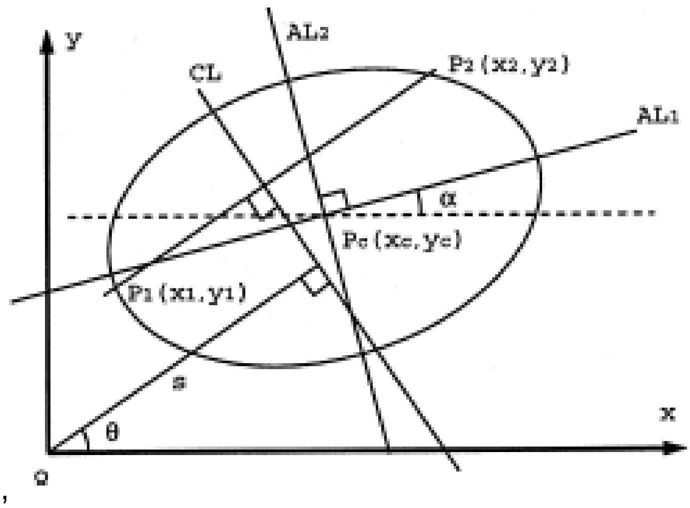
Houghovu transformáciu môžeme použiť na detekciu elipsy [YiWo98], pričom využijeme jej symetrické vlastnosti. Elipsa má 2 osi symetrie – hlavnú a vedľajšiu poloos. Pre každú dvojicu bodov na elipse môžeme nájsť ich spojnicu a zostrojiť os tejto spojnice. Elipsa musí obsahovať 2 poloosi, ktoré predstavujú zároveň osi symetrie.

Ak pre dvojicu pixelov na elipse zadefinujeme dvojicu parametrov (r=s,θ) v Houghovom priestore, ktoré predstavujú priamku prechádzajúcu touto dvojicou bodov, platí nasledovná rovnica:

$$s = x_m \cdot \cos(\theta) + y_m \cdot \sin(\theta) \qquad (7.13)$$

kde (x_m, y_m) sú súradnice stredu medzi dvojicou bodov

Pre symetrickú krivku, napr. elipsu, všetky páry pixelov, ktoré sú symetrické vzhľadom na poloosi, budú mať rovnaké hodnoty (r=s,θ). Podobne ako pri Houghovej transformácii pre identifikáciu priamky, aj tu hľadáme maximum v (r=s,θ) priestore. Na základe myšlienky zhody medzi dvojicou pixelov a ich spojnicou, bol vytvorený algoritmus, hľadajúci všetky osi symetrie z hranových pixelov na obraze.



Obr. 7.25 Parametre elipsy, [YiWo98]

Elipsa je definovaná piatimi parametrami. Súradnice stredu elipsy $P_c(x_c,y_c)$, natočenie α a dĺžka dvoch poloosí a, b (osi AL_1,AL_2). Pre dvojicu pixelov $P_1(x_1,y_1)$ a $P_2(x_2,y_2)$ na obryse definujeme priamku s parametrami (r=s,θ).

Algoritmus hľadania poloosí

Predpoklady:

Je potrebné definovať minimálnu vzdialenosť pre dvojicu pixelov D_{min} a akumulátor ACC[s][θ].

1. vyčisti akumulátor ACC
2. pre každý hranový pixel (x_1,y_1) vykonaj kroky 3. až 6.
3. pre každý iný hranový pixel (x_2,y_2) vykonaj kroky 4. až 6., ak je splnená podmienka $|x_2-x_1|>D_{min}$ alebo $|y_2-y_1|>D_{min}$
4. z dvojice pixelov (x_1,y_1) a (x_2,y_2) definujeme priamku, ktorej parametre sú s^* a Θ^*, ktoré vyrátame nasledovne
$$\Theta^* = \arctan(\frac{y_2 - y_1}{x_2 - x_1})$$
$$s^* = \frac{\cos\Theta^* \times (x_1 + x_2) + \sin\Theta^* \times (y_1 + y_2)}{2}$$
5. inkrementuj hodnotu akumulátora v bode $ACC[s^*][\theta^*]$
$ACC[s^*][\theta^*] = ACC[s^*][\theta^*] + 1/L$, kde
$L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ je vzdialenosť medzi dvojicou pixelov
6. opakuj kým nie sú spočítané hodnoty pre všetky dvojice pixelov
7. hľadaj výrazné hodnoty v akumulátore ACC. Urči maximum v bode (s_p, θ_p) z Hougovho priestoru. Tieto parametre určujú jednu z poloosí.
8. nájdi druhé maximum, ktorého parametre spĺňajú podmienku kolmosti dvoch poloosí.
9. koniec

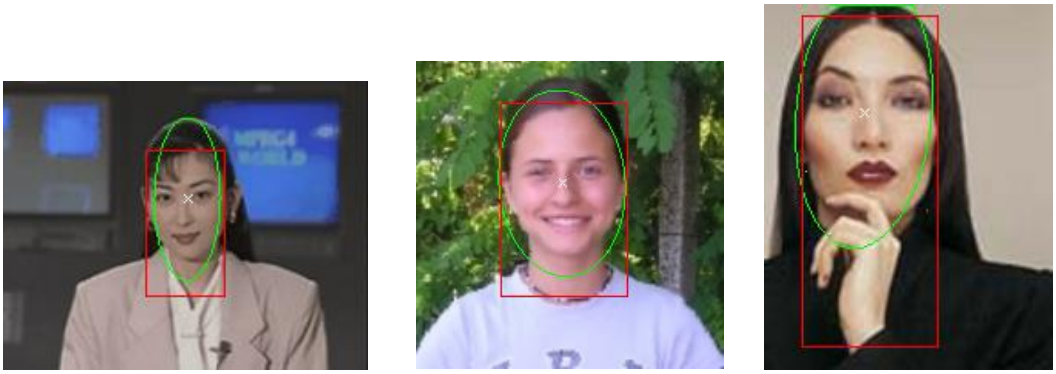
Výstupom tohto postupu sú parametre 2 navzájom kolmých priamok- poloosí elipsy, ktorých prienik určuje stred elipsy a natočenie θ. V implementácii sú do postupu zahrnuté iba pixely, ktorých vzdialenosť je väčšia ako D_{min} z nasledovných dôvodov.

- Priamka určená z dvoch bodov ležiacich vedľa seba je veľmi ovplyvnená šumom a nepresnosťou

- Výpočtová náročnosť je nižšia vzhľadom k vylúčeniu množstva susedných pixelov z výpočtu

Rozmery akumulátorového poľa priamo súvisia s presnosťou odhadu. Príliš malé, ale aj veľké pole znižuje presnosť výsledku.

V 5. kroku výpočtu inkrementujeme hodnotu akumulátorovej bunky o prevrátenú hodnotu vzdialenosti, čím kompenzujeme rôzne veľkosti symetrických objektov na obraze.



Obr. 7.26 *Výsledky použitiaHoughovej transformácie na základe symetrie poloosí elipsy*