Segmentace obrazu - hranové

Segmentace obrazu

- Segmentace obrazu
  - Rozčlenění obrazu na části, které úzce souvisí s objekty nebo částmi reálného světa
  - Nepřekrývající se oblasti

  - Segmentace
    - Kompletní (plně koresponduje s objekty reálného světa)
    - Částečná (nemusí plně korespondovat s objekty reálného světa)

  - Segmentace probíhá na základě stejnorodosti (homogenity) nějaké vlastnosti, např. velikosti jasu
  - Složitější segmentace, např. podle textury

Segmentace obrazu

- Segmentace obrazu na základě **detekce hran**
  - Vychází z pozorování, že hranice jsou zvýrazněny aplikací gradientních operátorů – místa prudkých změn intenzit jasů
  - Využívá se prahování

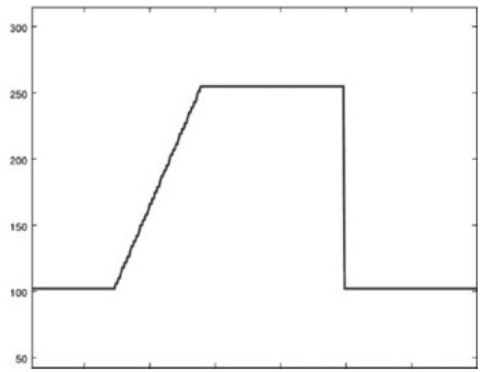
  - Výsledkem detekce hran je obraz, který ale není příliš použitelný
  - Proto následuje aplikace metod, které pospojují detekované hrany a vytvoří tak hranu (využívají ve větší či menší míře *apriorní* informace)

  - Marrova segmentace na základě inspirace biologickým viděním
  - Cannyho detektor hran

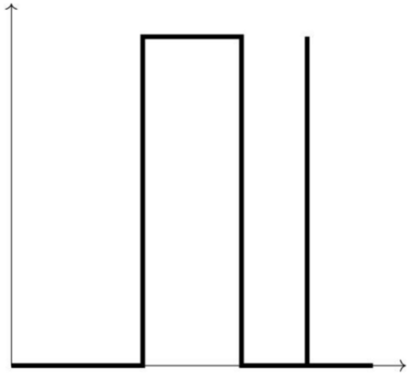
Detekce hran

- Hrany představují velmi užitečnou informaci v obraze.
- Mohou být využity pro měření velikosti objektů v obraze, pro oddělení objektů od pozadí, pro rozpoznání a klasifikaci objektů apod.
- Neformálně může být hrana definována jako lokální nespojitost v hodnotách pixelů, která překračuje danou mez. Jinými slovy se jedná o rozdíl v hodnotách sousedních pixelů.

šedotónový obrázek



profil hran f(x)



derivace profilu hran df/dx

Detekce hran

- Řada metod pro detekci hran je postavena na diferenci hodnot pixelů.
- Připomenutí definice derivace

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- Jiné vyjádření derivace: $\lim_{h \to 0} \frac{f(x) - f(x-h)}{h}, \quad \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$

- Pro diskrétní svět je hodnota jmenovatele *h* = 1, protože se jedná o vzdálenost dvou sousedních pixelů a místo o derivaci mluvíme o diferenci.

$$f(x+1) - f(x) \qquad f(x) - f(x-1), \qquad (f(x+1) - f(x-1))/2$$

Detekce hran

- **Gradient** – vektor parciálních derivací (diferencí) – míří ve směru největšího nárůstu hodnot

$$grad\ \ f(x,y,\ldots) = \nabla f(x,y,\ldots) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \ldots\right)$$

- Pro funkci dvou proměnných *f*(*x*,*y*), tj. v našem případě pro obrázek platí

$$\left|\nabla f\right| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \qquad \varphi = arctg\left(\frac{\partial f}{\partial x} \Big/ \frac{\partial f}{\partial y}\right)$$

velikost směr

- V diskrétním světě se derivace nahrazují diferencemi, a počítají se pomocí lineárních filtrů a masek, které jsou aproximacemi derivací.

Detekce hran

- **Filtry pro detekci hran** (výpočet aproximace 1. derivace)
  - Vyjdeme z definice derivace, resp. diference ve tvaru $f(x+1) - f(x-1)$
  - Potom můžeme realizovat horizontální a vertikální aproximaci derivace (diferenci) pomocí těchto masek lineárních filtrů:

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

  - Tyto filtry najdou horizontální, resp. vertikální hrany, ale jsou poněkud „syrové“.
  - Proto je výhodné výsledek vyhladit lineárním filtrem s maskami:

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Detekce hran

- Oba dílčí filtry můžeme zkombinovat do jednoho filtru, který se nazývá **filtr Prewittové**:

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

vertikální hrany　　　　horizontální hrany

- Pokud p_x a p_y jsou hodnoty (intenzity) pixelu získaného filtrací pomocí P_x , resp. P_y, potom velikost gradientu je dána vztahem

$$\sqrt{p_x^2 + p_y^2}$$

- Který je ale v praxi nahrazován jednodušším výpočtem a to $\max\{|p_x|, |p_y|\}$ nebo $|p_x| + |p_y|$

Detekce hran

- Příklad

původní obrázek

vertikální směr (P_x)

horizontální směr (P_y)

Detekce hran

- Zkombinování dílčích filtrovaných snímků

šedotónový obrázek
velikost gradientu
(zkombinování vertikální a horizontální filtrace)

prahovaný obrázek

Detekce hran

- Dalšími známými filtry jsou **Robertsův filtr**

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- a **Sobelův filtr** (mírně dává důraz na středový pixel)

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Varianta Sobelova filtru pro detekci hran (derivaci) v šikmém směru

$$h = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$$

Detekce hran

- Příklad Robertsova a Sobelova filtru

Robertsův filtr

Sobelův filtr

Detekce hran

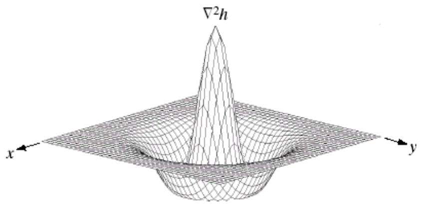
- **Filtry pro detekci hran** (výpočet aproximace 2. derivace)
  - Součet druhých derivací v obou směrech se nazývá **Laplaceův operátor** (*Laplacian*)

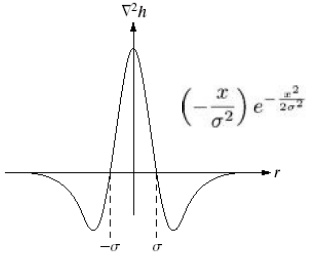
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

  - který může být aproximován maskou lineárního filtru (diskrétní Laplaceův operátor)

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{nebo} \quad h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{nebo} \quad \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix} \quad \text{nebo}$$

0	0	–1	0	0
0	–1	–2	–1	0
–1	–2	16	–2	–1
0	–1	–2	–1	0
0	0	–1	0	0

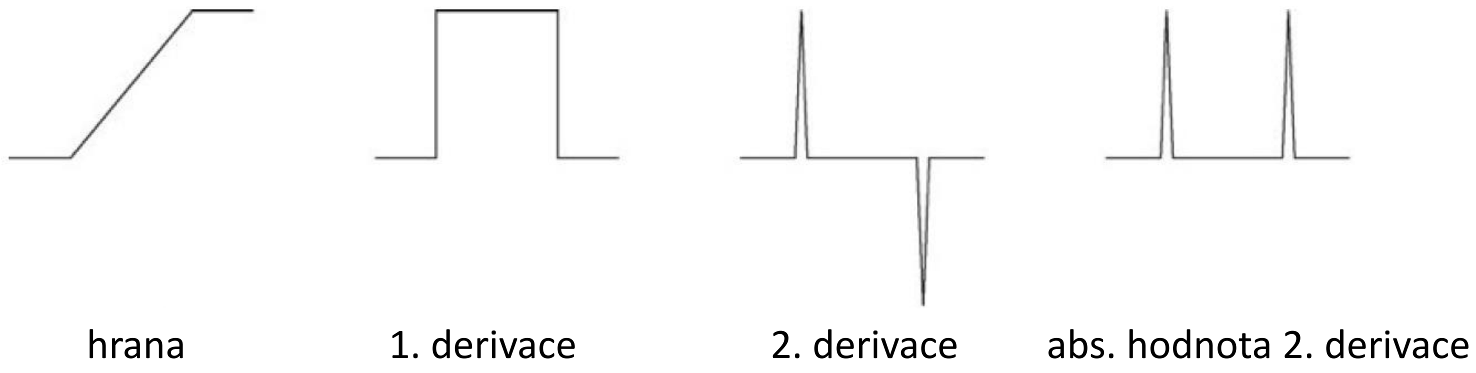




  - Oproti operátorům pro aproximaci 1. derivace se jedná o **izotropní** operátor.
  - Nevýhodou ale je, že je náchylný na šum.

Detekce hran

- Hrana a její derivace



- Druhá derivace způsobuje duplikování hran.

Detekce hran

- **Průchody nulou**
  - Možným způsobem, jak se vypořádat s duplicitou hran vzniklých použitím Laplaceova operátoru, je lokalizace míst, kde dochází k průchodu nulou (*zero crossings*) – v masce to jsou místa, kde dochází ke změně znaménka.
  - Místa průchodu nulou jsou definována pozicí pixelů, které splňují jednu z následujících podmínek:
    - Mají zápornou hodnotu a sousedí aspoň s jedním pixelem, který má nezápornou hodnotu (4-okolí).
    - Mají nulovou hodnotu a leží mezi pixelem se zápornou hodnotou a pixelem s kladnou hodnotou.

obrázek

50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50

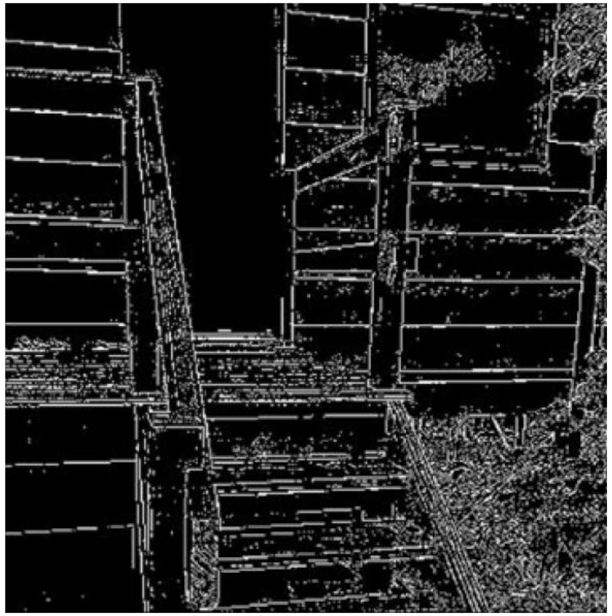
-100	-50	-50	-50	-50	-50	-50	-50	-50	-100
-50	0	150	150	150	150	150	150	0	-50
-50	150	-300	-150	-150	-150	-150	-300	150	-50
-50	150	-150	0	0	0	0	-150	150	-50
-50	150	-150	0	0	0	0	-150	150	-50
-50	150	-300	-150	0	0	0	-150	150	-50
-50	0	150	0	-150	0	0	-150	150	-50
-50	0	0	150	-300	-150	-150	-300	150	-50
-50	0	0	0	150	150	150	150	0	-50
-100	-50	-50	-50	-50	-50	-50	-50	-50	-100

filtrovaný obrázek pomocí Laplaceova operátoru + šedě vyznačené průchody nulou

Detekce hran

- Kombinací aplikace Laplaceova operátoru a následnou identifikací hranových pixelů pomocí detekce míst průchodů nulou získáme hranový detektor.
- Příklad

Filtrace pomocí Laplaceova operátoru



Po následné detekci míst průchodů nulou

- V obrázku je bohužel příliš mnoho hran díky různým drobným změnám v intenzitách.

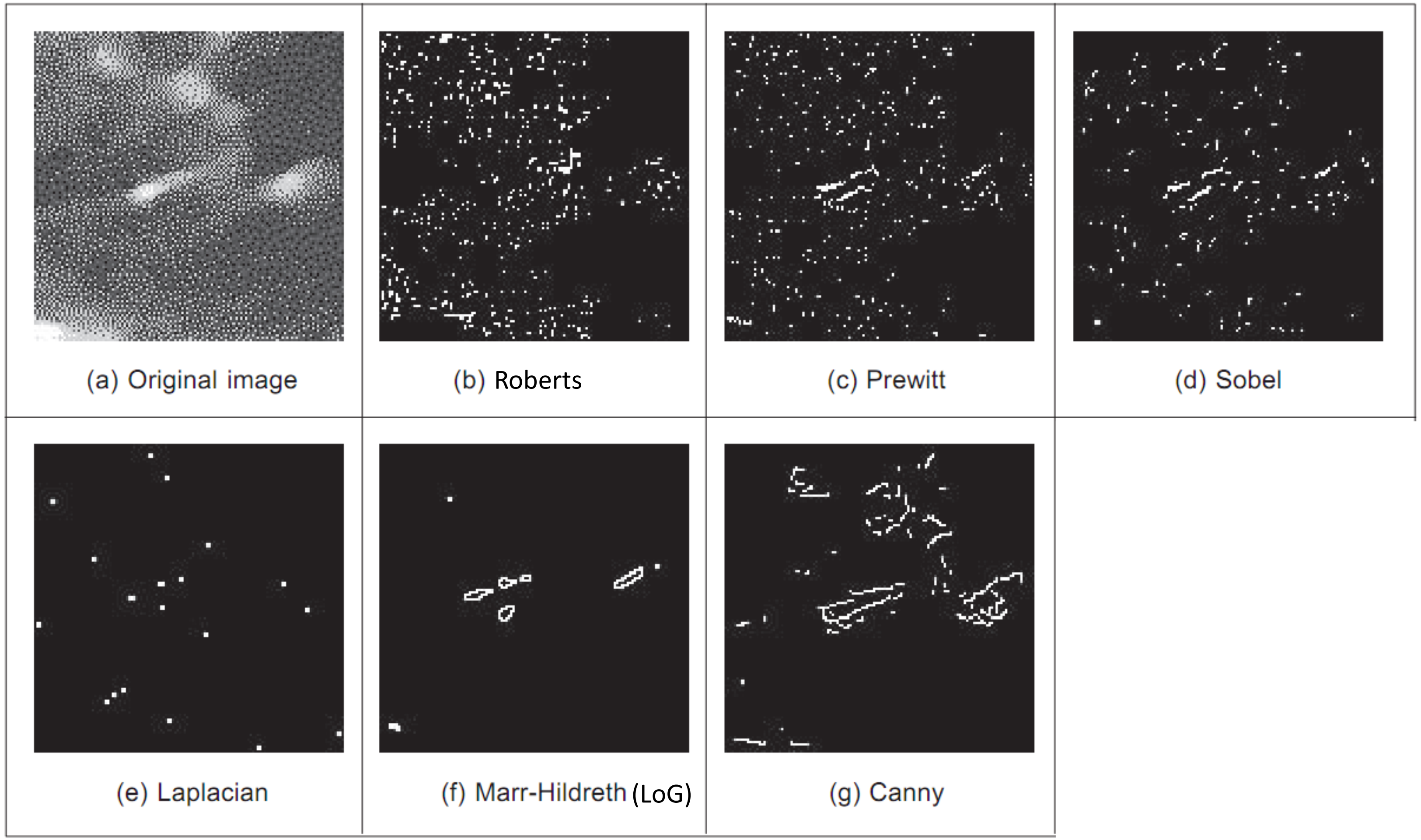
Detekce hran

- Aby nedocházelo k detekci přehnaného množství hran, je výhodné obrázek předem vyfiltrovat, např. pomocí Gaussova filtru – **Marrova-Hildrethova metoda**:
    1. Vyhlazení obrázku pomocí Gaussova filtru
    2. Následná filtrace pomocí Laplaceova filtru
    3. Identifikace míst průchodu nulou

- Kombinace prvních dvou kroků se nazývá **LoG filtr** („*Laplacian of Gaussian*“).
- Velmi dobře aproximuje biologickou filtraci obrazu v oku.

Detekce hran

- Srovnání různých hranových filtrů



Detekce hran

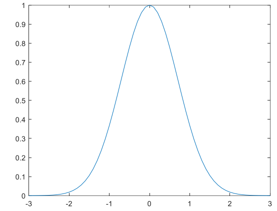
- **Cannyho detektor** – John Canny, 1986
  - Tři kritéria/požadavky na detektor:
    - **Nízká chybovost** – schopnost najít hrany a jen hrany
    - **Lokalizace hran** – vzdálenost mezi hranami v obrázku a detekovanými hranami by měla být minimální
    - **Jednoduchá odezva** – nalezeny by měly být hrany jednoduše reprezentované pixely, ne vícenásobné a široké hrany

  - Detekce hran v Cannyho detektoru probíhá v několika krocích:
    1. Potlačení šumu - snížení rizika chybné detekce hran Gaussův filtr
    2. Nalezení primárních hran - Sobelův filtr nebo derivace Gaussova filtru
    3. Potlačení nemaximálních hodnot
    4. Prahování s hysterezí

Detekce hran

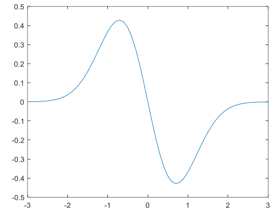
- Tyto kroky **předzpracování** obrázku lze shrnout takto:
  1. Provedeme **vyhlazení**, např. vytvoříme jednodimenzionální Gaussův filtr *g*.

$$e^{-\frac{x^2}{2\sigma^2}}$$



  2. Vypočítáme gradient (**detekujeme hrany**), např. použijeme Sobelův filtr nebo vytvoříme jednodimenzionální derivovaný Gaussův filtr *dg*.

$$\left(-\frac{x}{\sigma^2}\right) e^{-\frac{x^2}{2\sigma^2}}$$



  3. Spočítáme konvoluci *g* a *dg* a získáme filtr *gdg* (vyhlazení + detekce hran).
  4. Filtr gdg aplikujeme na původní obrázek *x* a získáme obrázek hran x_1 (vodorovný směr)
  5. Filtr gdg^T aplikujeme na původní obrázek *x* a získáme obrázek hran x_2 (svislý směr)
  6. Výsledný primární obrázek hran získáme jako $x_e = \sqrt{x_1^2 + x_2^2}$

- *Pozn. Využíváme toho, že Gaussův filtr i jeho derivace jsou separovatelné filtry.*

Detekce hran

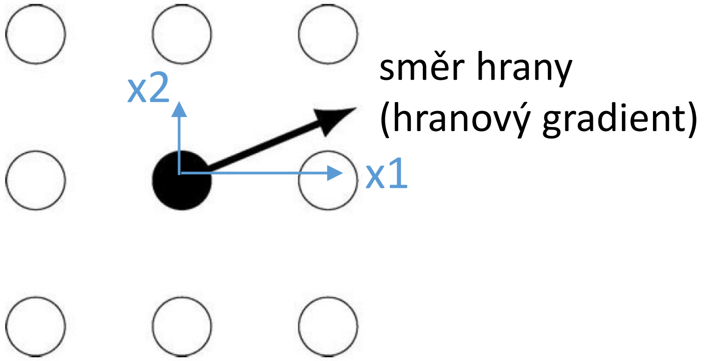
- Dalším krokem je lokální **potlačení nemaximálních hodnot** v obrázku (ztenčení hran).
- Cílem je ponechat jen ty pixely, které patří hraně.
- Obyčejným prahováním bychom nedosáhli dobrých výsledků.

- Místo toho každému pixelu přiřadíme (vypočítáme) na základě gradientu směr. Směr gradientu je kolmý na směr hrany v obraze, kterou se snažíme detekovat.
- Pixel, který tvoří hranu, potom musí mít větší **velikost** (intenzitu=velikost gradientu) než jeho lokální sousedé ve **směru** gradientu daného pixelu, tj. pixely, které jsou po straně detekované hrany.
- Máme-li například pixel, jímž prochází svislá hrana, musí mít jeho levý a pravý soused nižší hodnoty než uvažovaný pixel (= max. hodnota gradientu daného pixelu v lokálním okolí), aby byl daný pixel „uznán“ jako skutečná hrana.

Detekce hran

- Směr gradientu (vyjádřený úhlem) vypočítáme ze vztahu

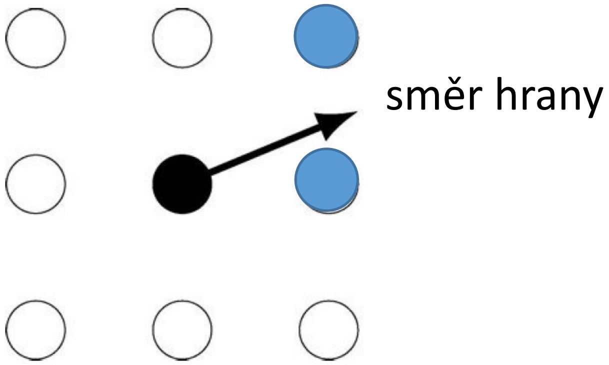
$$xg = \tan^{-1}\left(\frac{x2}{x1}\right)$$

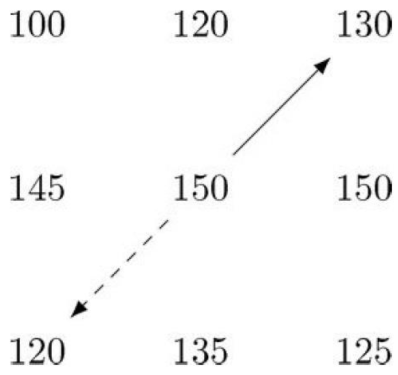


- Vypočtený směr gradientu obecně nemíří ke konkrétnímu pixelu, ale mezi pixely, takže je třeba zvážit míru jejich příspěvku pro výpočet velikosti gradientu na základě sousedních pixelů.

Detekce hran

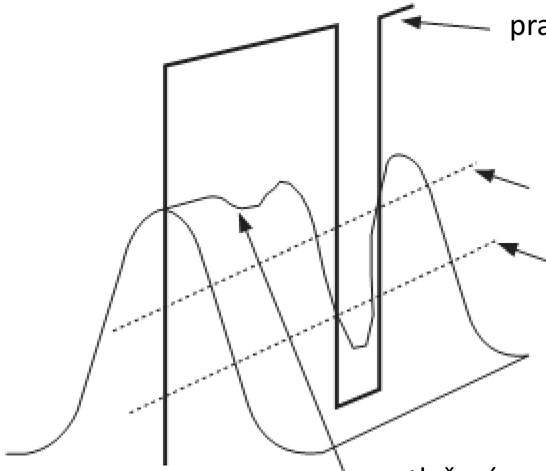
- Výpočet můžeme provést např. těmito dvěma způsoby:
  1. Za sousední pixely gradientu bereme ty, mezi které míří gradient a z nich spočítáme vážený průměr, kde vahami jsou míry vzdálenosti gradientu od jednoho resp. druhého pixelu. Např. na obrázku vlevo by se jednalo o dva modře zvýrazněné pixely vpravo nahoře. Příspěvek horního modrého pixelu pak bude o něco menší než dolního modrého pixelu.
  2. Za sousední pixely gradientu bereme jednak ten, co je blíže směru gradientu a ten, který leží přesně na opačnou stranu, viz obrázek vpravo. Pokud je hodnota středového pixelu větší než hodnota jeho dvou identifikovaných sousedů, pixel ponecháme (náš případ), jinak ho smažeme.

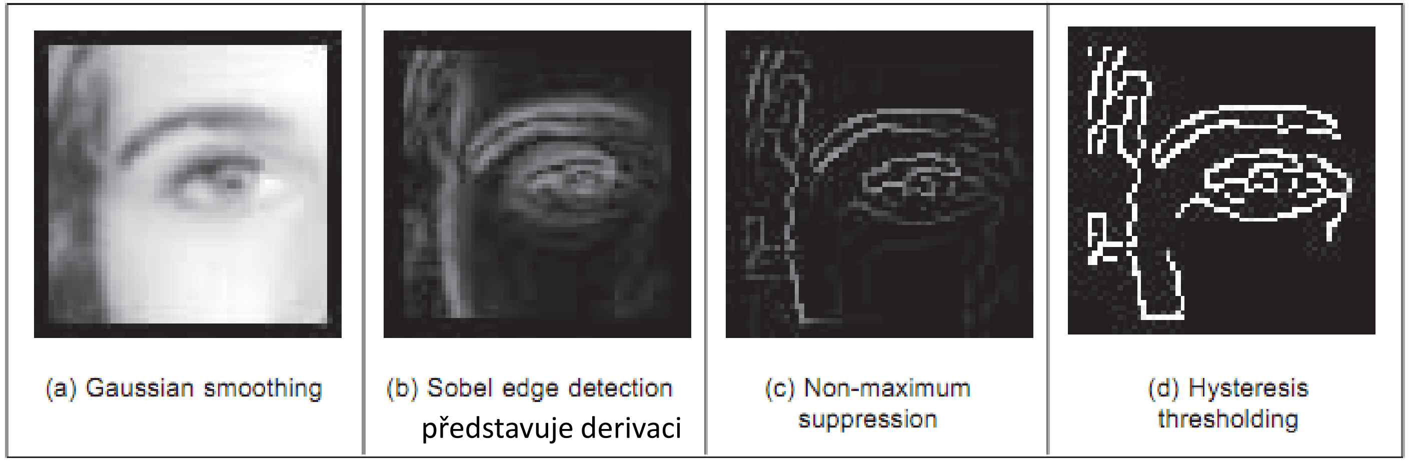




Detekce hran

- Následně provádíme **prahování**, abychom získali finální hrany.
- Cannyho detektor nepoužívá jednoduché prahování, ale tzv. **prahování s hysterezí** (*hysteresis thresholding*), které používá dva prahy: dolní práh t_L a horní práh t_H
  - Pixel, jehož hodnota je větší než t_H je považován za pixel hrany (strong edge pixel)
  - Pixel, jehož hodnota leží mezi t_L a t_H, a který současně sousedí (8-okolí) s jiným hranovým pixlem (strong edge pixel), je považován za pixel hrany (weak edge pixel).





(a) Gaussian smoothing

(b) Sobel edge detection

představuje derivaci

(c) Non-maximum suppression

(d) Hysteresis thresholding

Detekce rohů

- **Rohy** – místa, kde se setkávají dvě hrany, které jsou orientovány různým směrem.
- Existuje řada detektorů rohů:
  - Moravcův detektor rohů
  - Harrisův-Stephensův detektor (a jeho různá vylepšení)
  - Förstnerův detektor rohů
  - Wangův-Bradyho detektor rohů
  - Trajkovicův-Hedleyho detektor rohů
  - Tomasiho-Kanadeův detektor rohů
  - Beaudetův detektor rohů
  - …
- Nejznámější jsou Moravcův detektor a Harrisův-Stephensův detektor.

Detekce rohů

- Moravcův detektor rohů
  - Jeden z nejstarších a nejjednodušších detektorů
  - Roh je identifikován jako pixel, jehož okolí se výrazně liší od lokálních okolí jiných pixelů.

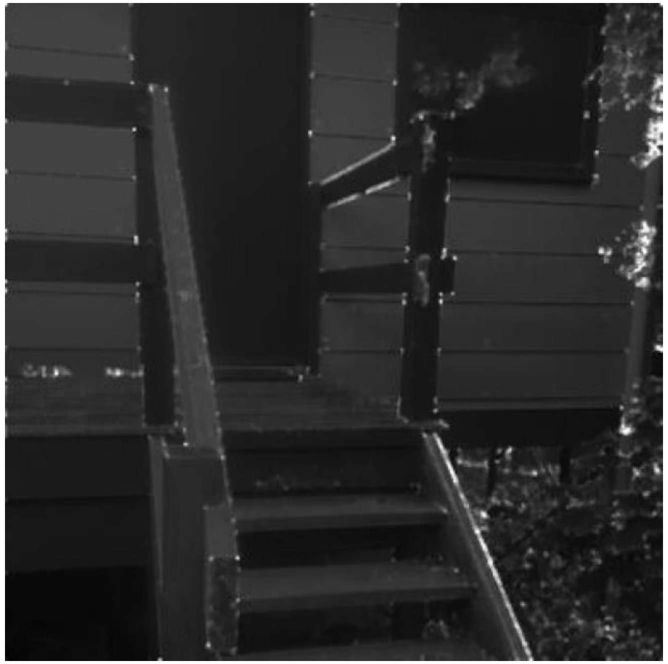
1. Předpokládejme, že pracujeme se čtvercovým oknem (maskou) *W*, která má rozměry o lichém počtu pixelů.
2. Okno svým středem umístíme na uvažovaný pixel *p*.
3. Okno postupně posouváme o jeden pixel ve všech osmi směrech od pixelu *p* a pro každý tento posun *s=(i, j)* vypočítáme rozptyl intenzity (intensity variation)

$$I_s = \sum (W(x,y) - W_s(x,y))^2$$

4. Následně vypočítáme minimum *M* ze všech hodnot I_s. Každému pixelu snímku odpovídá jedna hodnota *M*, která vyjadřuje míru, s jakou odpovídající pixel tvoří roh.
5. Celý postup opakujeme pro všechny pixely v obrázku (okraje obrázku doplňujeme nulami).

Detekce rohů

- Moravcův detektor rohů

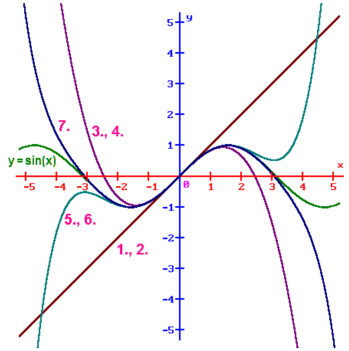


50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	150	150	150	150	150	150
50	50	50	50	150	150	150	150	150	150
50	50	50	50	150	150	150	150	150	150
50	50	50	50	150	150	150	150	150	150
50	50	50	50	150	150	150	150	150	150
50	50	50	50	150	150	150	150	150	150

5	0	0	0	0	0	0	0	0	5
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	0	0	0	0	20
0	0	0	10	20	0	0	0	0	20
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
5	0	0	20	20	0	0	0	0	45

- Nevýhodou Moravcova detektoru rohů je, že občas detekuje i hrany jako rohy a navíc nemusí detekovat různé natočené rohy – není izotropní.

Detekce rohů



- Harrisův-Stephensův detektor rohů
  - Též nazývaný jen Harrisův detektor
  - Založen na Taylorově rozvoji 1. řádu funkce dvou proměnných (lokální aprox. obrazu)

$$f(x+h, y+k) \approx f(x,y) + h\frac{\partial f}{\partial x}(x,y) + k\frac{\partial f}{\partial y}(x,y)$$

  - Derivace jsou počítány (aproximovány) pomocí lineárních filtrů.
  - Podobně jako Moravcův detektor rohů, i Harrisův detektor počítá „rozptyl“ intenzit různě posunutých lokálních okolí

$$\sum_{(u,v)\in K} (I(u+s, v+t) - I(u,v))^2$$

  - kde (*s, t*) je uvažovaný posun a *K* je okolí (maska).

Detekce rohů

- Využijeme Taylorův rozvoj funkce uvedený výše a aplikujeme ho na výpočet „rozptylu“ (parciální derivace označíme $I_x = \frac{\partial I(u,v)}{\partial x}$ a $I_y = \frac{\partial I(u,v)}{\partial y}$ – udávají míru změny intenzity ve směru *x* a *y*)

$$\begin{aligned}\sum_{(u,v)\in K} (I(u+s, v+t) - I(u,v))^2 &= \sum_{(u,v)\in K} (I(u,v) + sI_x(u,v) + tI_y(u,v) - I(u,v))^2 \\ &= \sum_{(u,v)\in K} (sI_x(u,v) + tI_y(u,v))^2 \\ &= \sum_{(u,v)\in K} (s^2 I_x^2 + 2stI_xI_y + t^2 I_y^2) \\ &= \sum_{(u,v)\in K} \begin{bmatrix} s & t \end{bmatrix} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} s \\ t \end{bmatrix}\end{aligned}$$

- Harrisův detektor se zaměřuje na určení matice parciálních derivací:

$$H = \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$$

Detekce rohů

$$f(x+h, y+k) \approx f(x,y) + h\frac{\partial f}{\partial x}(x,y) + k\frac{\partial f}{\partial y}(x,y)$$

- Využijeme Taylorův rozvoj funkce uvedený výše a aplikujeme ho na výpočet „rozptylu“ (parciální derivace označíme $I_x = \frac{\partial I(u,v)}{\partial x}$ a $I_y = \frac{\partial I(u,v)}{\partial y}$ – udávají míru změny intenzity ve směru *x* a *y*)

$$\begin{aligned}
\sum_{(u,v)\in K} (I(u+s, v+t) - I(u,v))^2 &= \sum_{(u,v)\in K} (I(u,v) + sI_x(u,v) + tI_y(u,v) - I(u,v))^2 \\
&= \sum_{(u,v)\in K} (sI_x(u,v) + tI_y(u,v))^2 \\
&= \sum_{(u,v)\in K} (s^2 I_x^2 + 2stI_xI_y + t^2 I_y^2) \\
&= \sum_{(u,v)\in K} \begin{bmatrix} s & t \end{bmatrix} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} s \\ t \end{bmatrix}
\end{aligned}$$

- Harrisův detektor se zaměřuje na určení matice parciálních derivací:

$$H = \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$$

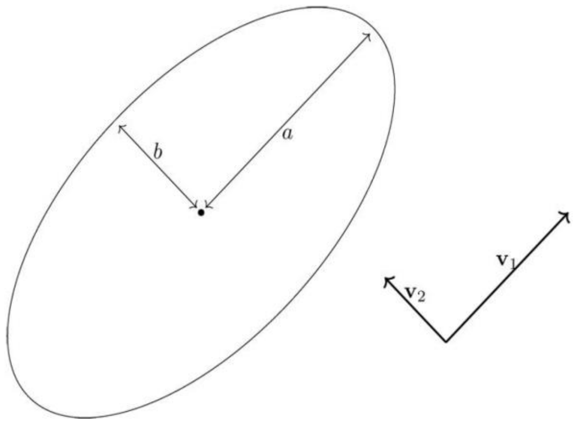
Detekce rohů

- Pro daný posun (s, t) je výraz $s^2 I_x^2 + 2st I_x I_y + t^2 I_y = c$ konstantní a rovnice představuje rovnici elipsy.
- Délky a směry poloos a, b elipsy jsou dány vlastními čísly λ_1 and λ_2, resp. vlastními vektory $\mathbf{v}_1$ a $\mathbf{v}_2$ matice H.

$$H\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

$$a = (\lambda_1)^{-1/2}, b = (\lambda_2)^{-1/2}$$

$$a^2 = \frac{1}{\lambda_1}, b^2 = \frac{1}{\lambda_2}$$



- Protože velikosti poloos nepřímo odpovídají velikostem vlastních čísel, je nejpomalejší (nejméně strmá) změna intenzit ve směru největších vlastních čísel a naopak.

Detekce rohů

- Na základě známých hodnot vlastních čísel rozlišujeme tři případy:
  1. Obě vlastní čísla jsou velká -> žádná významná změna v jakémkoliv směru (jednolitá oblast bez výraznějších změn)
  2. Jedno vlastní číslo je velké a druhé malé -> značí detekovanou **hranu** ve směru malého vlastního čísla (vektoru)
  3. Obě vlastní čísla jsou malá -> značí detekovaný **roh**

- Protože matice *H*, ze které se vypočítávají vlastní čísla, je diagonálně symetrická, dá se výpočet vlastních čísel zjednodušit.
- Mějme matici

$$M = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$$

- Potom lze najít vlastní čísla řešením kvadratické rovnice

$$\lambda^2 - (x+z)\lambda + (xz - y^2) = 0$$

Detekce rohů

- V uvedené rovnici je $x+z$=stopa(M) a $xz-y^2$=det(M).
- Protože výpočet odmocniny v uvedené kvadratické rovnici je výpočetně náročný, navrhli Harris a Stephens jednodušší přibližný výpočet, který počítá pouze stopu (trace) a determinant.
- Výsledná matice R (*corner response*) má rozměry původního obrázku a její hodnoty odpovídají míře detekovaných rohů (velké hodnoty odpovídají rohům)

$$R = \det(M) - k(\mathrm{Tr}(M))^2$$

- kde k je volitelný parametr citlivosti. Typicky se jeho hodnota pohybuje mezi 0,04 a 0,15.

Detekce rohů

- Výpočet detekce rohů lze shrnout do několika jednoduchých kroků
    1. Vypočítáme parciální derivace obrázku ve směrech x a y. Parciální derivace aproximujeme lokálními hranovými filtry, tj. detekujeme hrany ve vodorovném a svislém směru.
    2. Spočítáme prvky matice *H*, tj.

$$S = \sum I_x^2, \quad T = \sum I_x I_y, \quad U = \sum I_y^2$$

    3. Vypočítáme matici *R* (obrázek s detekovaný rohy; ty lze zvýraznit následným prahováním)

$$R = (SU - T^2) - k(S + U)^2$$

- Pro zvýšení robustnosti algoritmu lze v druhém kroku počítat místo prostého součtu intenzit v daném okně vážený součet daný Gaussovým filtrem, tj. spočítat konvoluce

$$S = (I_x^2) * G, \quad T = (I_x I_y) * G, \quad U = (I_y^2) * G$$

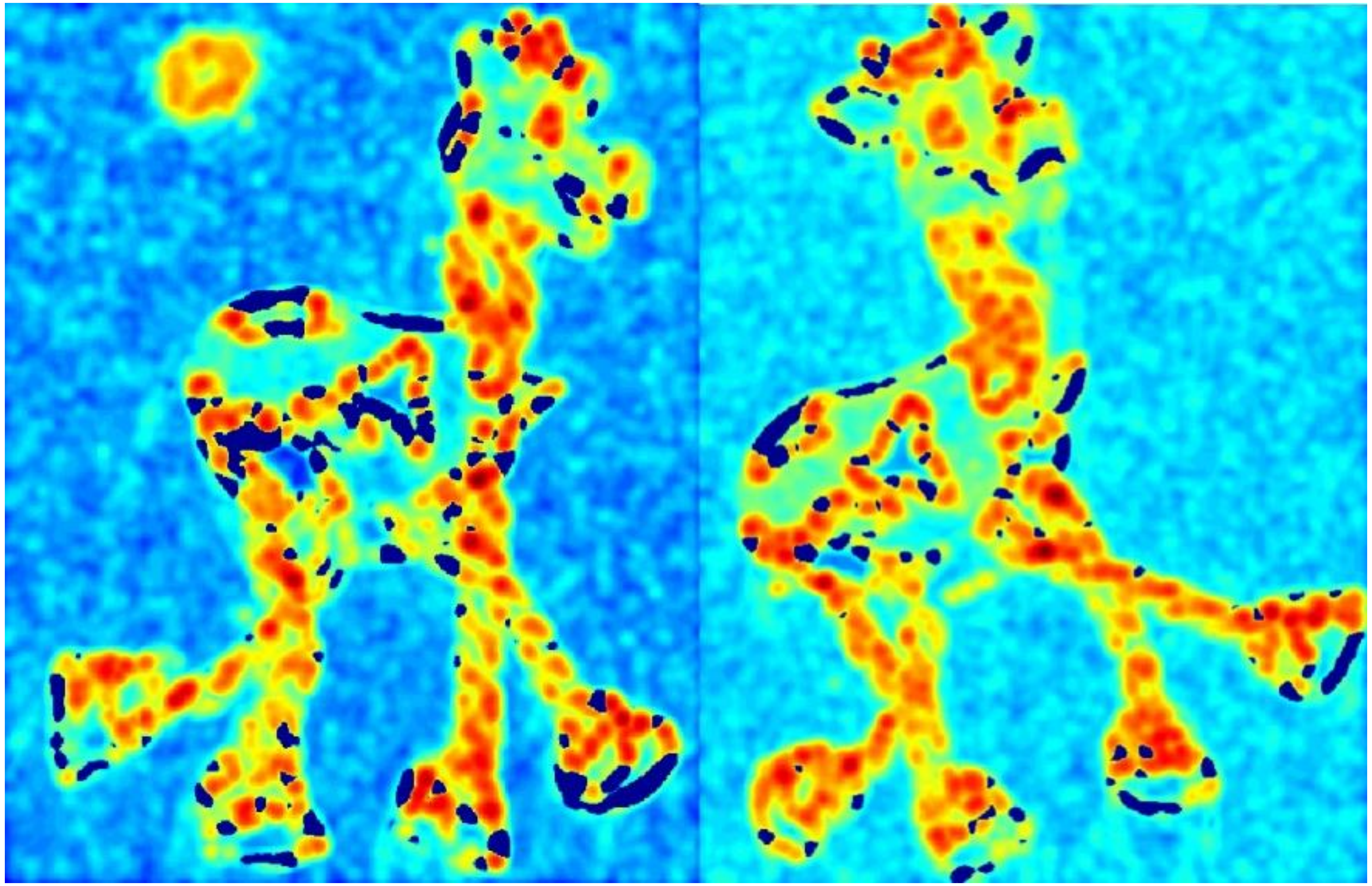
- kde *G* je Gaussův filtr (matice vah).

Detekce rohů – Harrisův detektor

- Původní obrázky

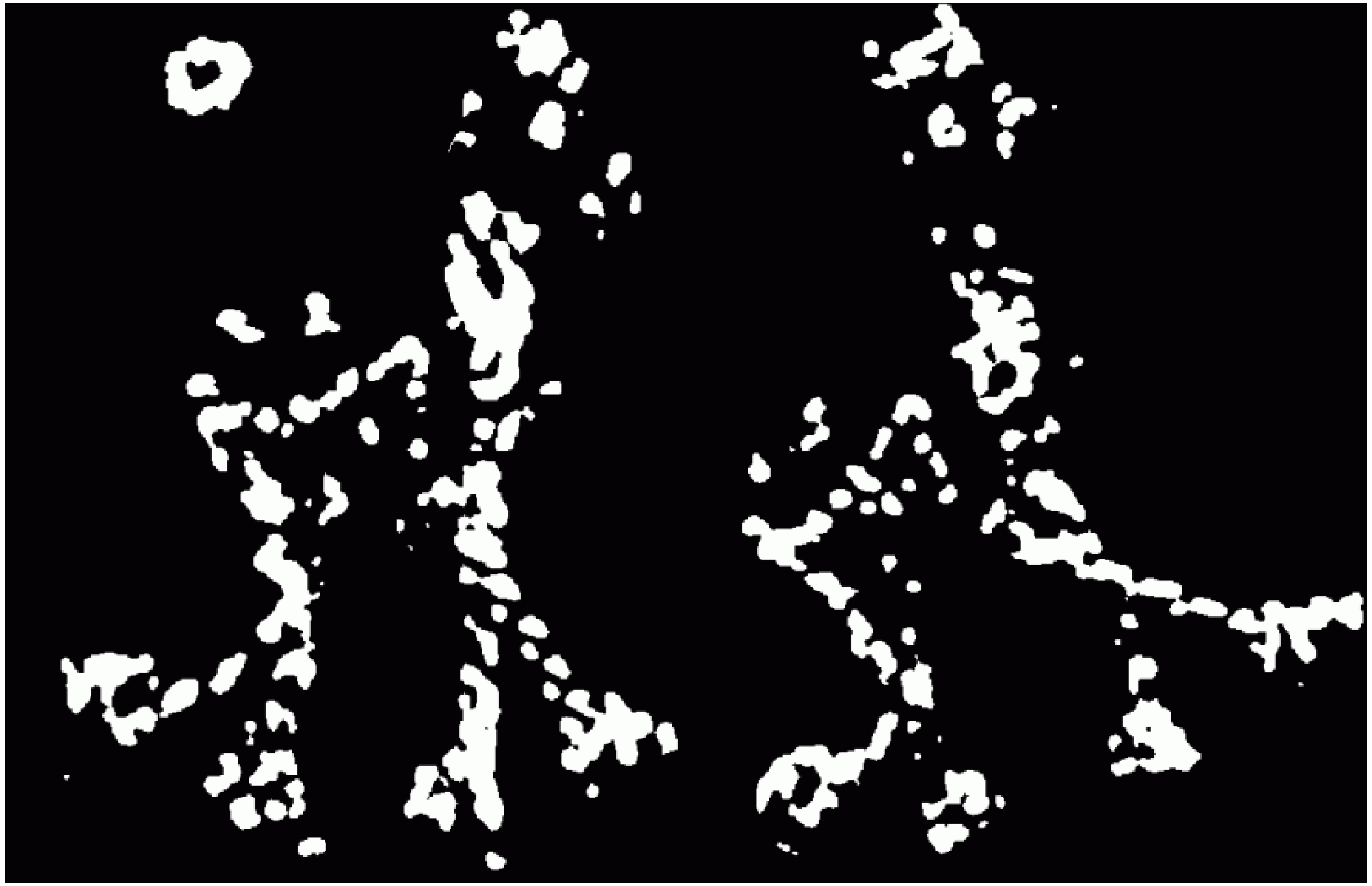
Detekce rohů – Harrisův detektor

- Matice R (*Corner response*)



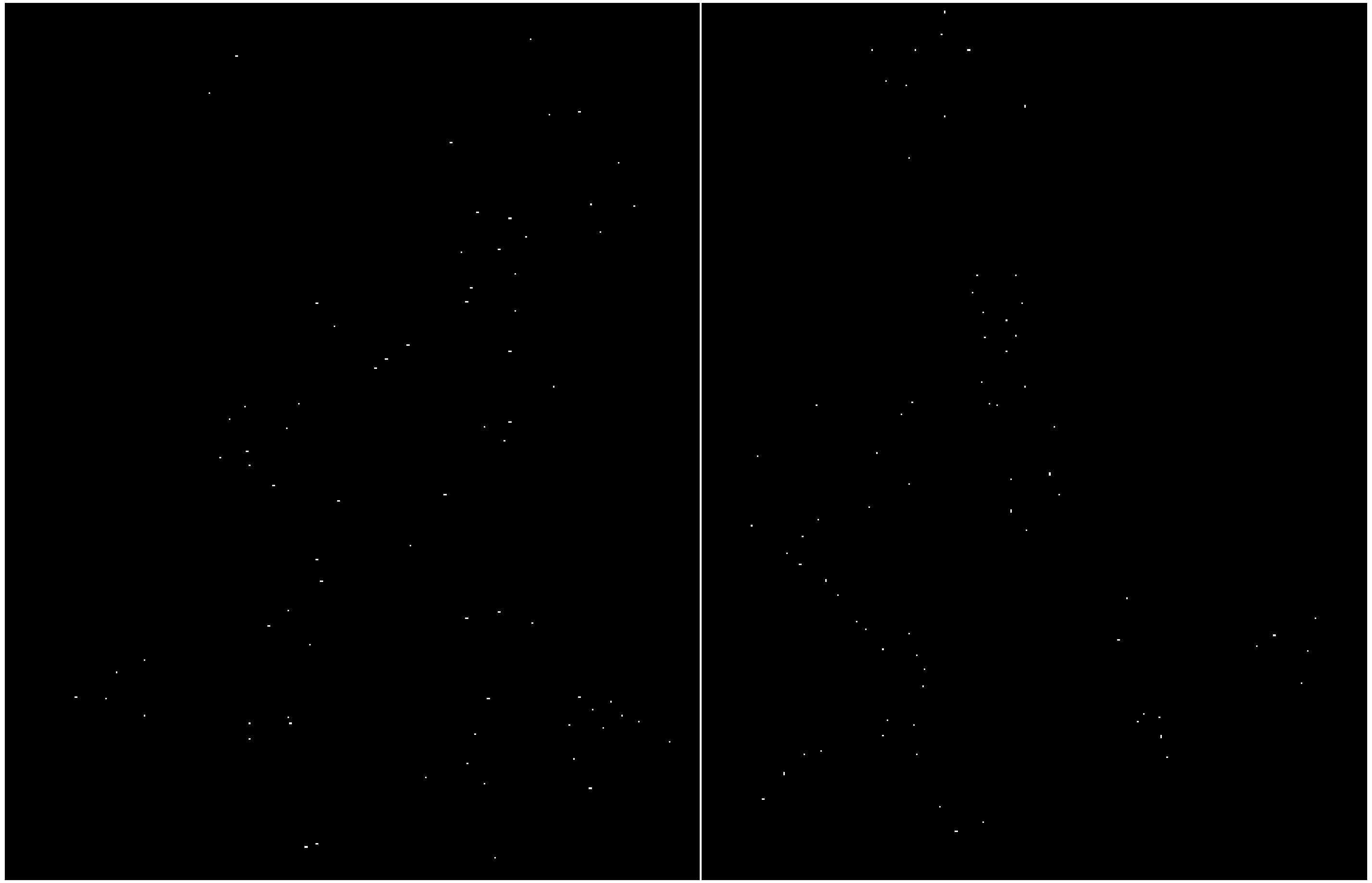
Detekce rohů – Harrisův detektor

- Oprahování R > daný práh



Detekce rohů – Harrisův detektor

- Lokální maxima (potlačení nemaximálních hodnot v lokálních okolích)



Detekce rohů – Harrisův detektor

- Výsledná detekce rohů

Literatura

- McAndrew A., Computational Introduction to Digital Image Processing, CRC Press, 2. vydání, 2016
- Sundararajan D., Digital Image Processing: A Signal Processing and Algorithmic Approach, Springer, 2017
- Birchfield S., Image Processing and Analysis, Cengage Learning, 2016
- Acharya T., Ray A. K., Image Processing: Principles and Applications, Wiley, 2005
- Burger W., Burge M. J., Principles of Digital Image Processing: Fundamental Techniques, Springer-Verlag, 2009