Ako vidí robot alebo ako trafiť robotom do dverí

Andrej Lúčny

Katedra aplikovanej informatiky

Fakulta matematiky, fyziky a informatiky

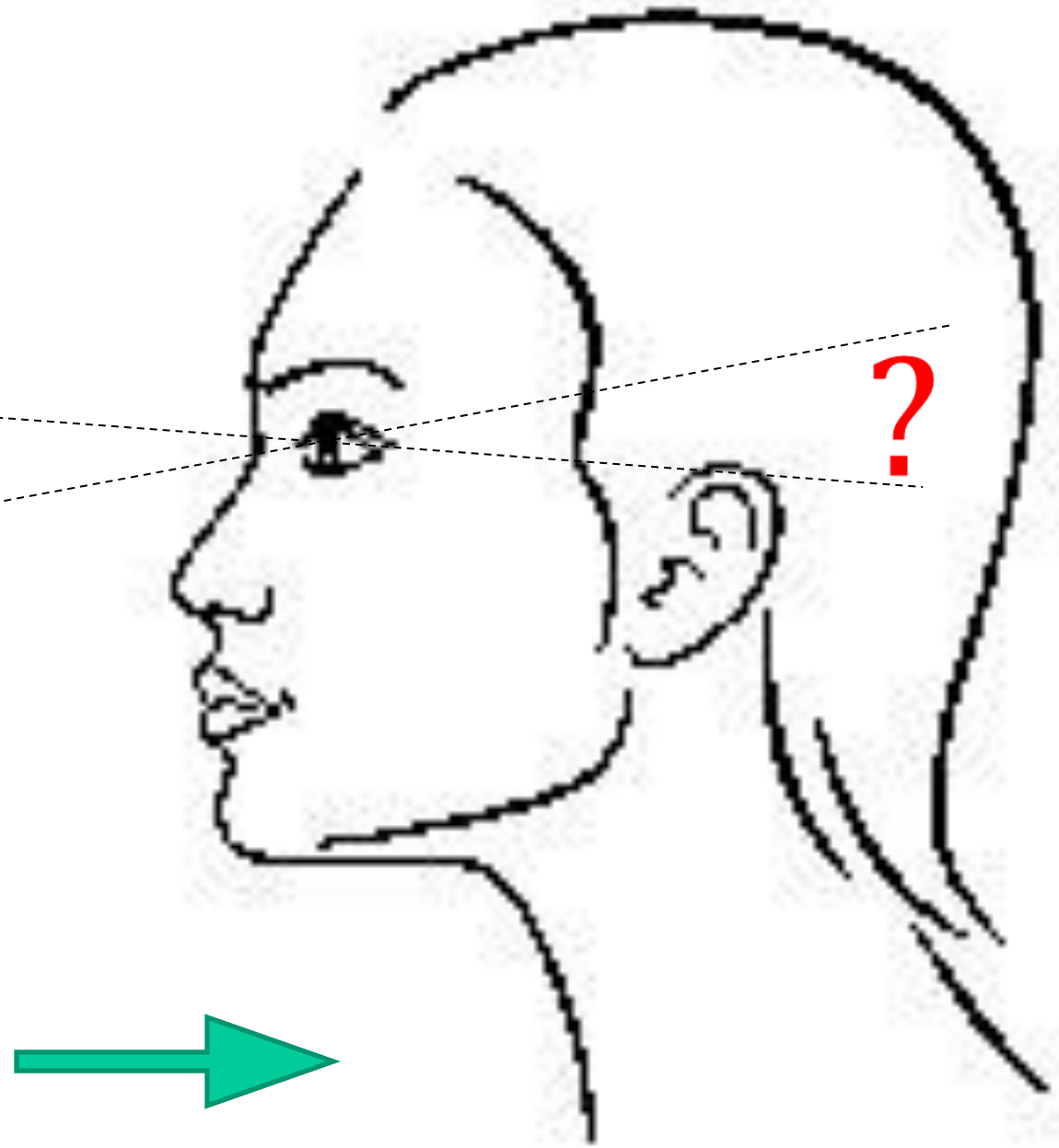
Univerzita Komenského v Bratislave

& MicroStep-MIS

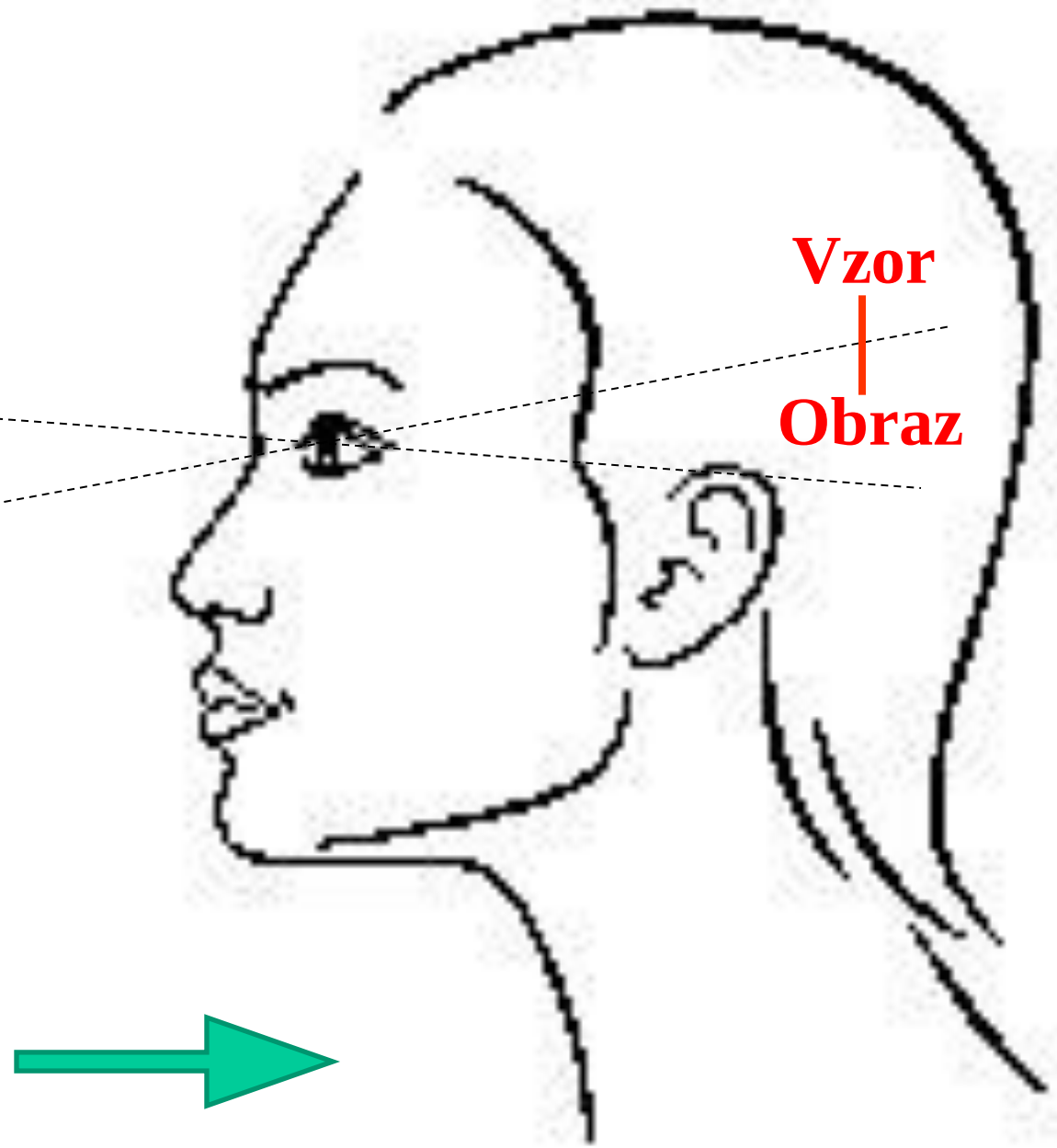
andy@microstep-mis.com

www.microstep-mis.com/~andy

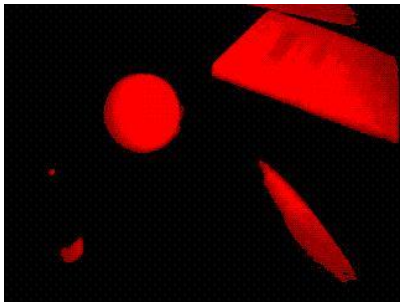
Čo to znamená vidieť ?

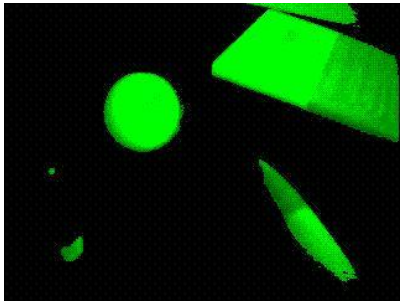


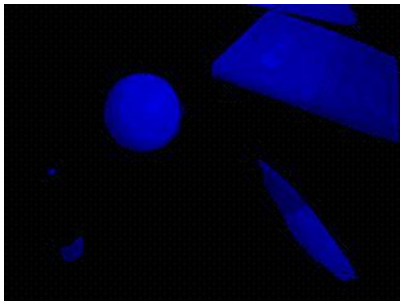
Čo to znamená vidieť ?

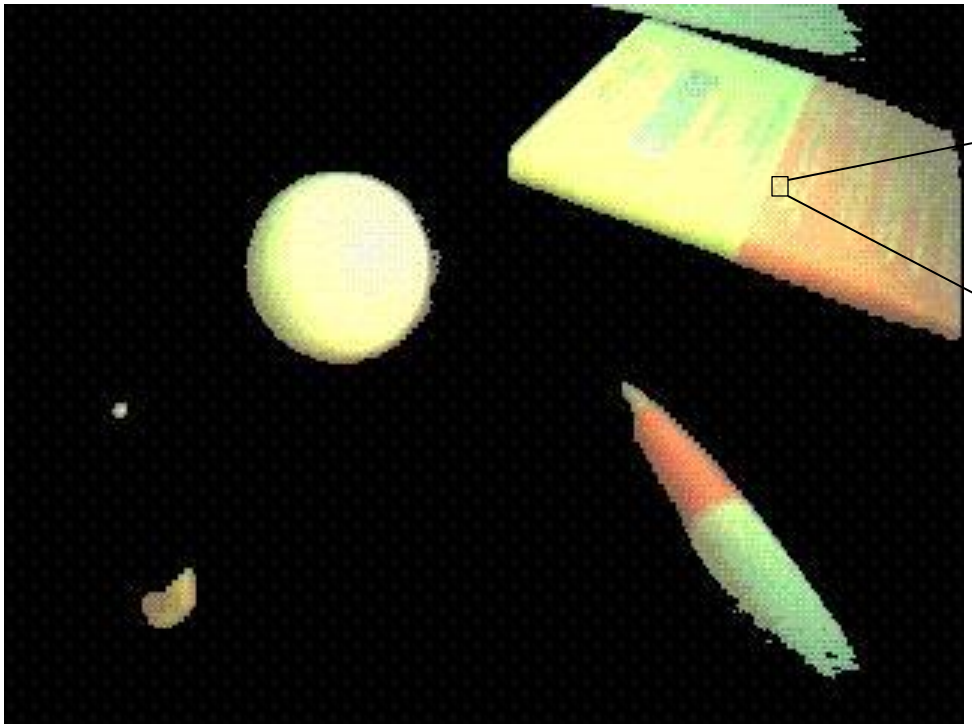


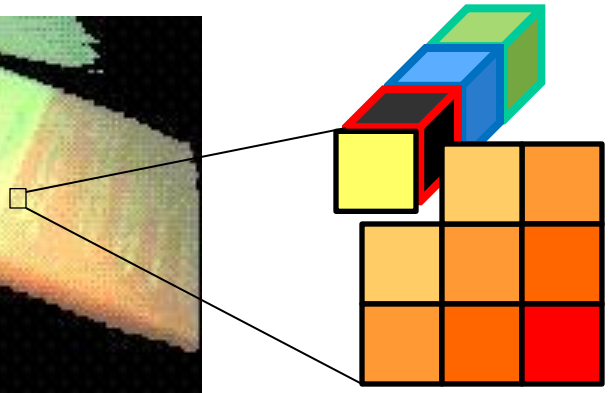
Obraz











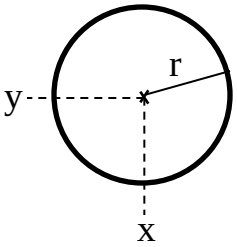
Tri polia r[h,w], g[h,w], b[h,w], každý ich prvok je číslo 0..255 a predstavujú červenú, zelenú a modrú zložku farby

Vzor

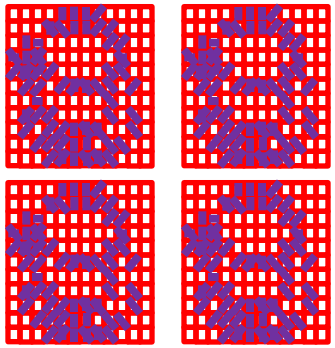
HOUGH

DOT (HOG)

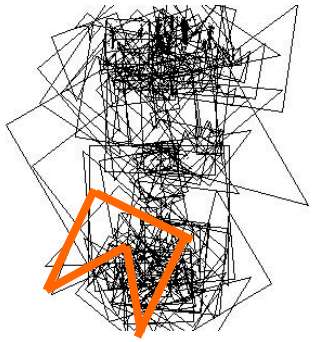
SIFT (SURF, ORB)



(x,y,r)



((0,0,0,0,3,4,..) (0,0,0,A,F,….)…)
((0,0,0,0,3,4,..) (0,0,0,A,F,….)…)
((0,0,0,0,3,4,..) (0,0,0,A,F,….)…)
((0,0,0,0,3,4,..) (0,0,0,A,F,….)…)



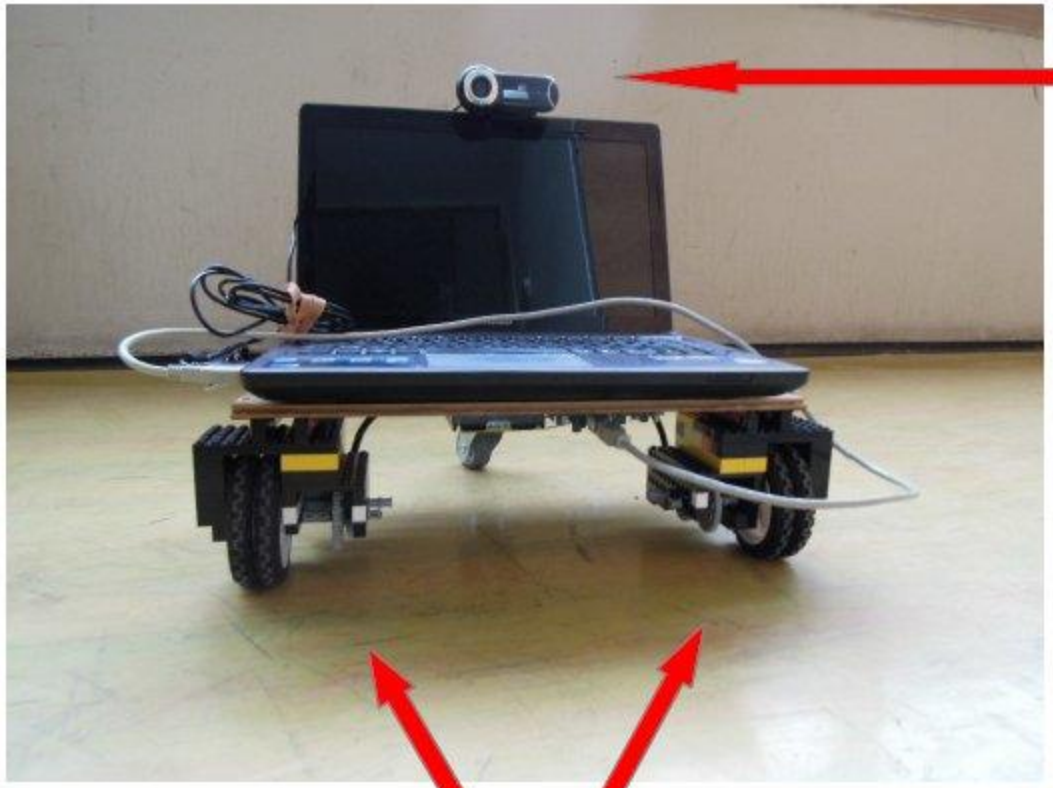
((123,234,94,1,4,13,22,240,2,…),
(123,234,32,102,44,33,232,420,…),
…
(23,130,295,2,144,133,32,40,…))

Úloha

Nekolízny pohyb mobilného robota s dvojkolesovým podvozkom a jednou kamerou v prostredí chodieb na Matfyze

Prostredie

Mobilný notebook

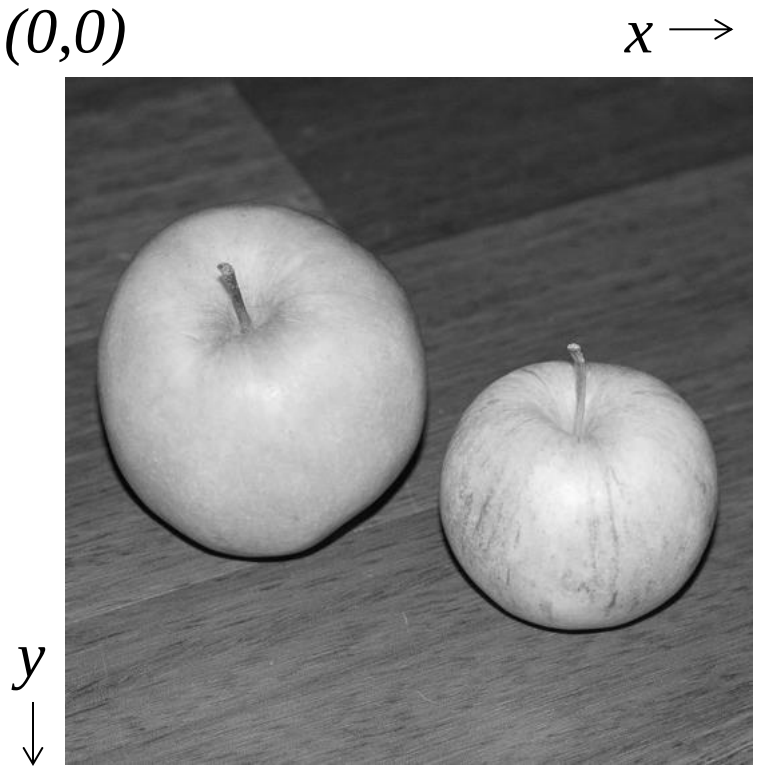


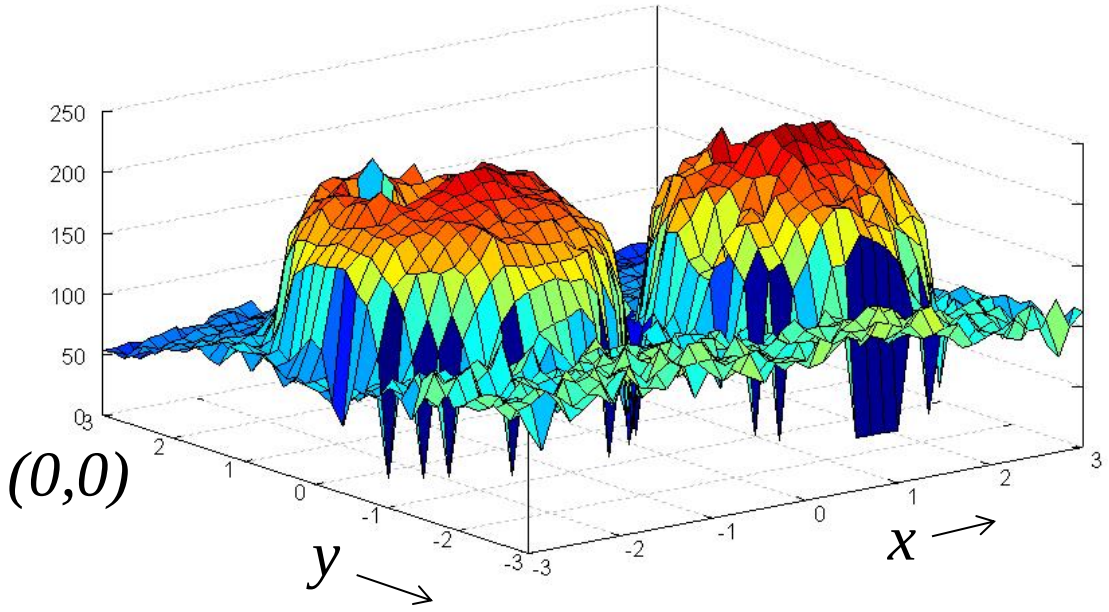
Diplomová práca
Ondrej Mikuláš
2013

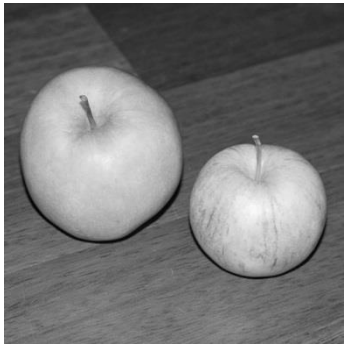
Situácia: prejdenie chodbou

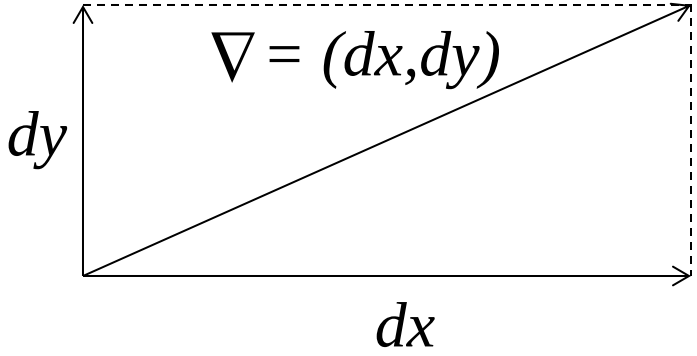
Farebný obraz na čiernobiely

Obraz ako 2-rozmerná funkcia



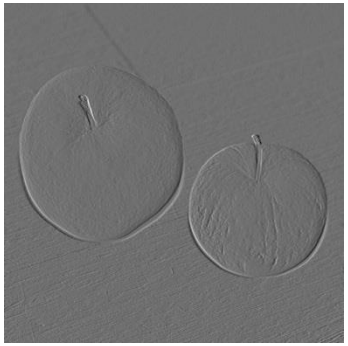




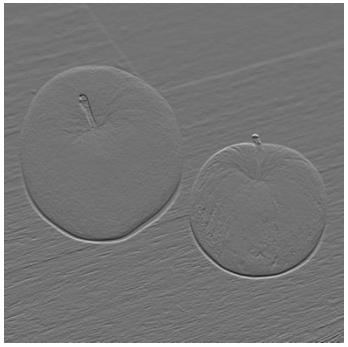


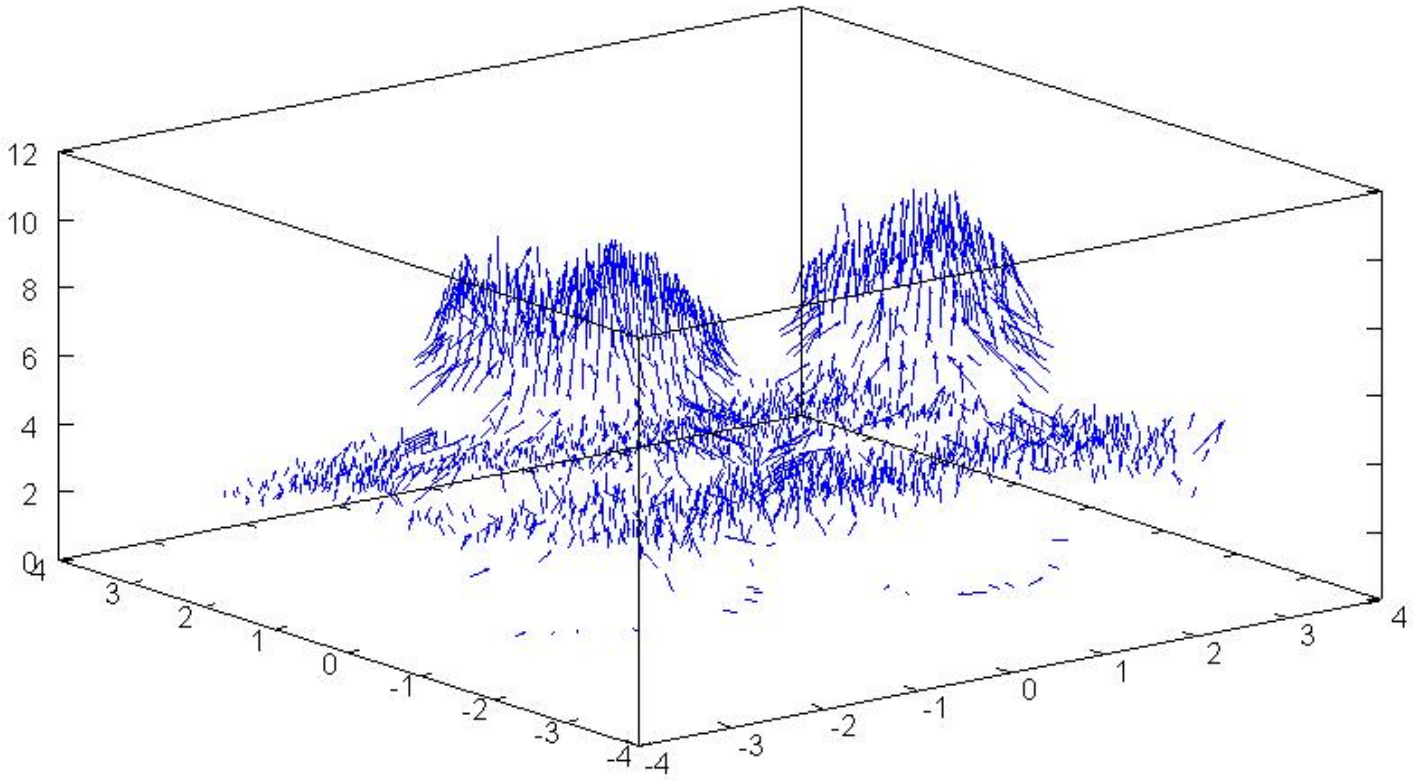
Gradient ∇

dx



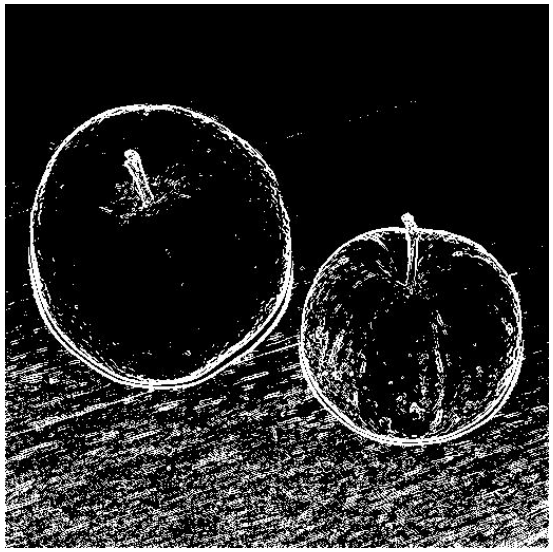
dy

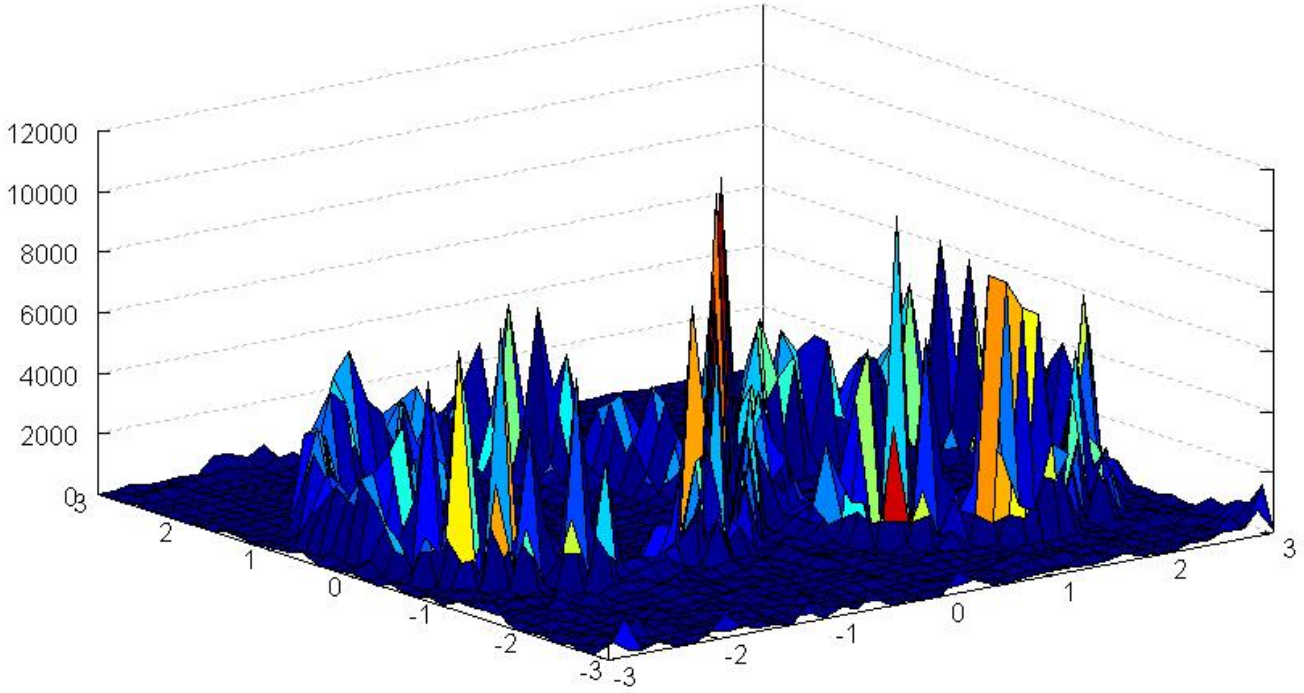




Laplacian Δ

druhá mocnina veľkosti gradientu





$\Delta = \nabla \nabla$ $= dx^2 + dy^2$ Hrany = $\| \Delta \|$

Blur (Smooth) - potláča šum za cenu straty ostrosti

veľmi dôležité !

Blur (Smooth)

- Hodnotu nahradíme (váženým) priemerom vedľajších

$a_{i-1,j-1}$	$a_{i-1,j}$	$a_{i-1,j+1}$
$a_{i,j-1}$	$a_{i,j}$	$a_{i,j+1}$
$a_{i+1,j-1}$	$a_{i+1,j}$	$a_{i+1,j+1}$

o

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

	$b_{i,j}$	

$$b_{i,j} = (a_{i-1,j+1} + a_{i,j+1} + a_{i+1,j+1} + a_{i-1,j} + a_{i,j} + a_{i+1,j} + a_{i-1,j-1} + a_{i,j-1} + a_{i+1,j-1}) / 9$$

Sobelov operátor

- používa rovnaký trik, len v trochu rafinovanejšej verzii, keď zohľadňuje aj vedľajšie riadky

$a_{i-1,j-1}$	$a_{i-1,j}$	$a_{i-1,j+1}$
$a_{i,j-1}$	$a_{i,j}$	$a_{i,j+1}$
$a_{i+1,j-1}$	$a_{i+1,j}$	$a_{i+1,j+1}$

o

-1	0	1
-2	0	2
-1	0	1

=

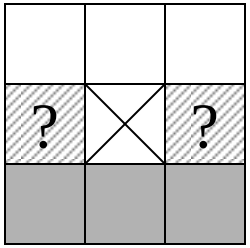
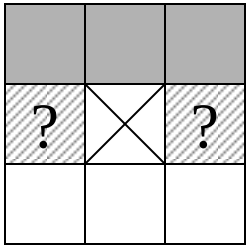
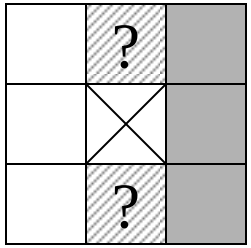
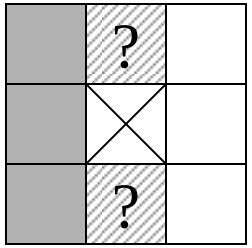
	$b_{i,j}$	

$$b_{i,j} = | a_{i-1,j+1} + 2a_{i,j+1} + a_{i+1,j+1} - a_{i-1,j-1} - 2a_{i,j-1} - a_{i+1,j-1} |$$

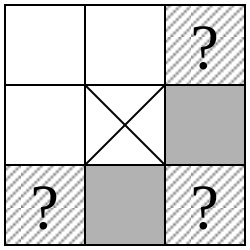
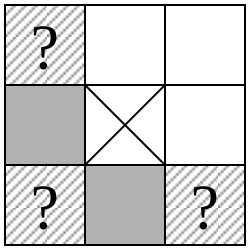
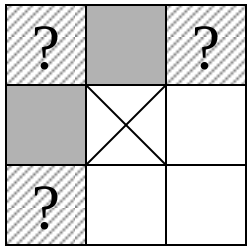
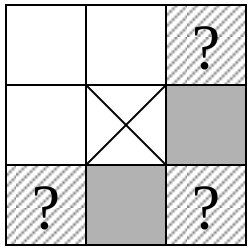
Sobelov operátor (1. fáza Canny)

Stenčovanie hrán

- vodorovné a zvislé hrany:

- rohy:

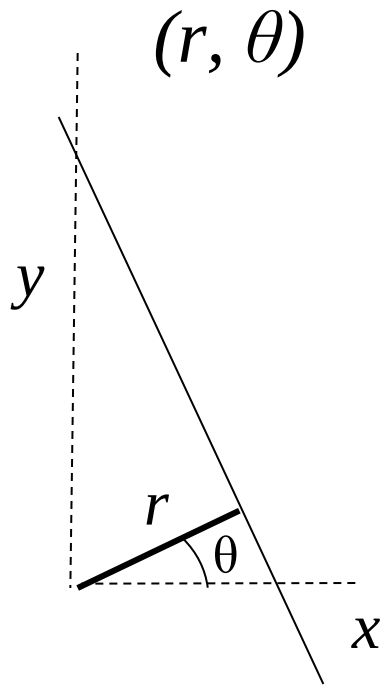





- izolované body

Hrany z algoritmu Canny

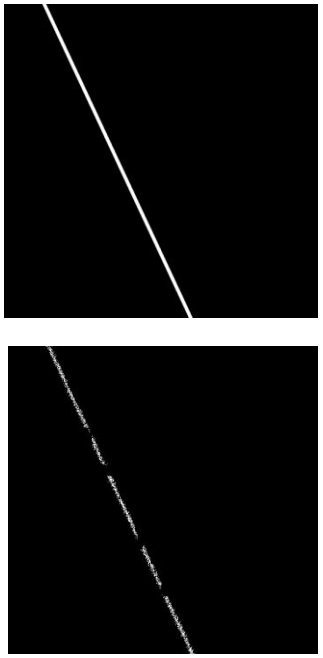
Houghova transformácia

$$r = x \cos\theta + y \sin\theta$$



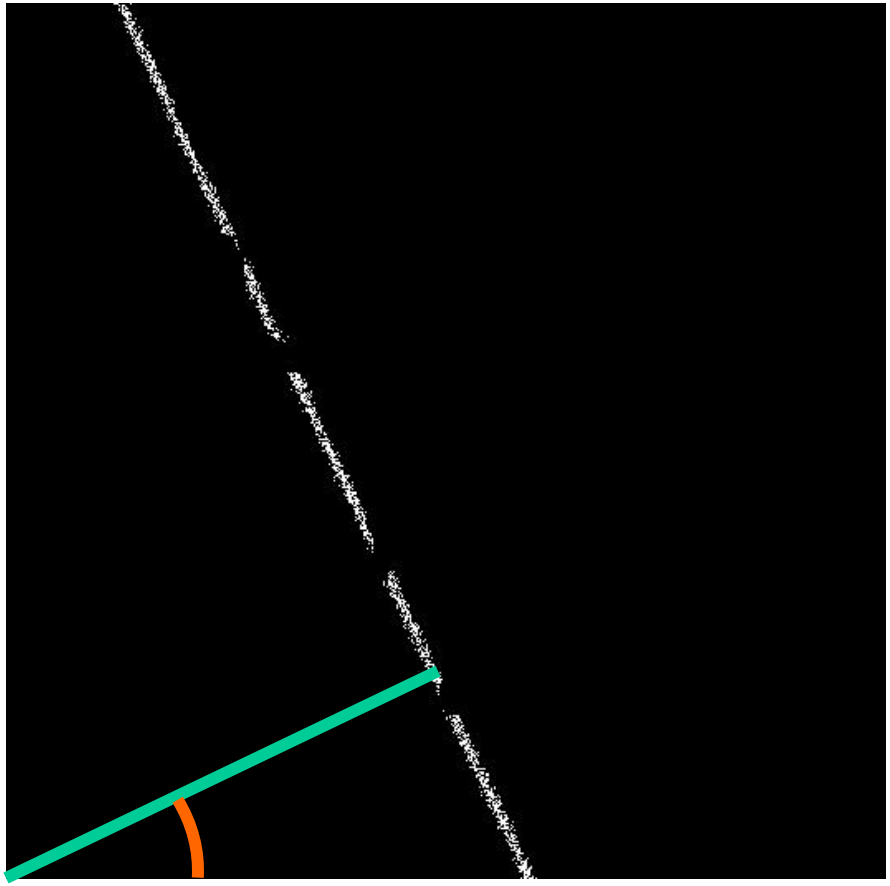
Rendering

Houghova transformácia

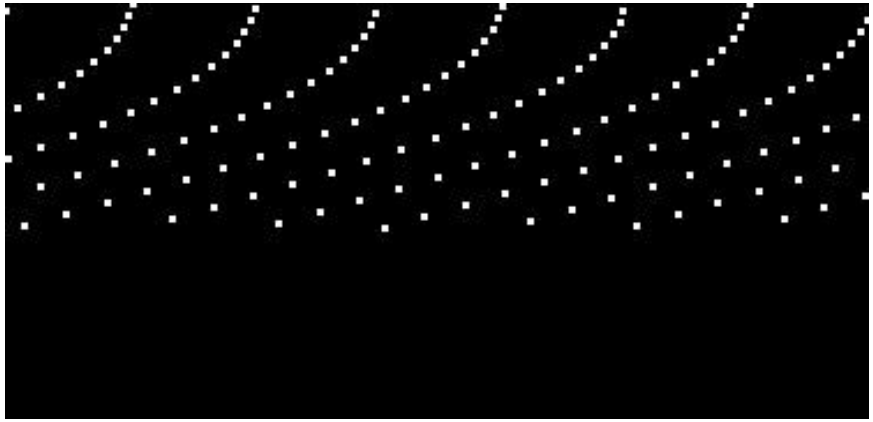


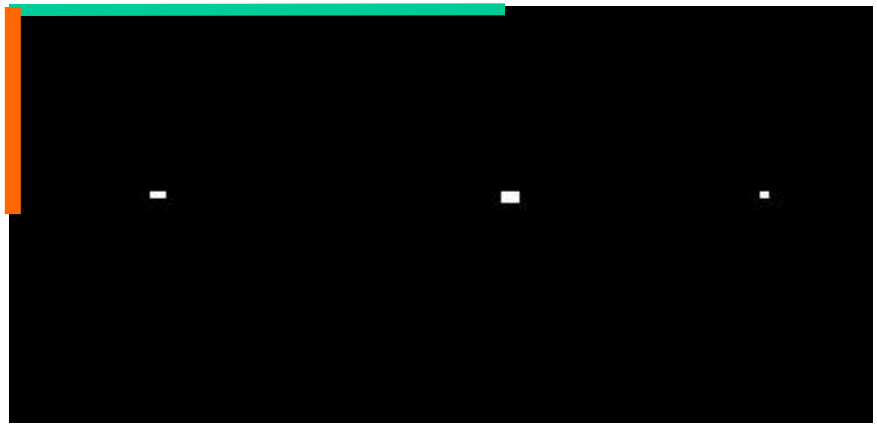
Houghova transformácia

Hlasujúce bodky



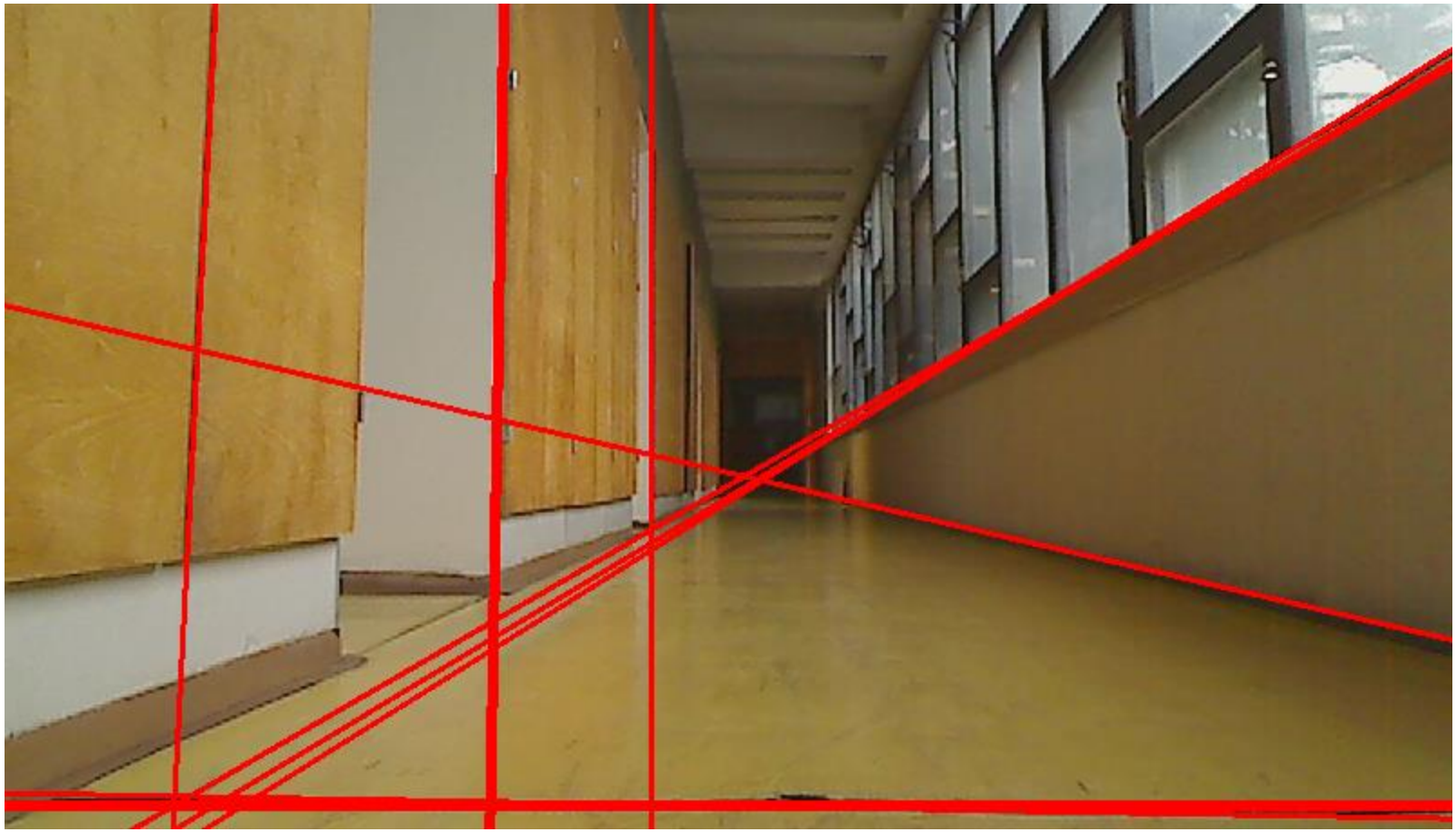
hlasy jednej bodky





sumár hlasov všetkých

Houghova transformácia - priamky

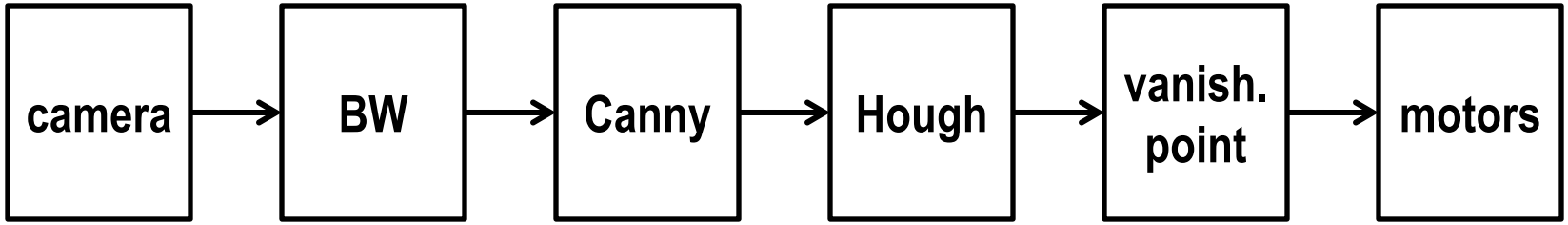


Houghova transformácia - úsečky

Úsečky relevantné pre úbežný bod

Úbežný bod – priesečník

Pohyb robota k úbežnému bodu



Nadviazaním jednotlivých metód do tzv. pipeline dostávame riadiaci systém, ktorý zvládne úlohu za jednej konkrétnej situácie (prechod chodbou)

Ako sa otáčať

- Na základe straty úbežného bodu sa aktivuje nové správanie, ktoré sa raz za určitý čas votrie do základného správania (t.j. do pohybu naslepo vpred)
- Umlčíme pritom pôvodné príkazy pre motory a točíme sa na mieste na jednu či na druhú stranu (protichodným pohybom motorov) hľadajúc nový úbežný bod

Ako sa otáčať

- Smerom do strany možno otáčanie načasovať, t.j. stačí merať čas
- Motory sú ale nepresné, takže vrátiť sa týmto spôsobom späť do správneho smeru je problém
- Používame preto vizuálnu informáciu
- **Ako zistíme, že vidíme to isté, čo predtým ?**

Ako si pamätať tvar ?

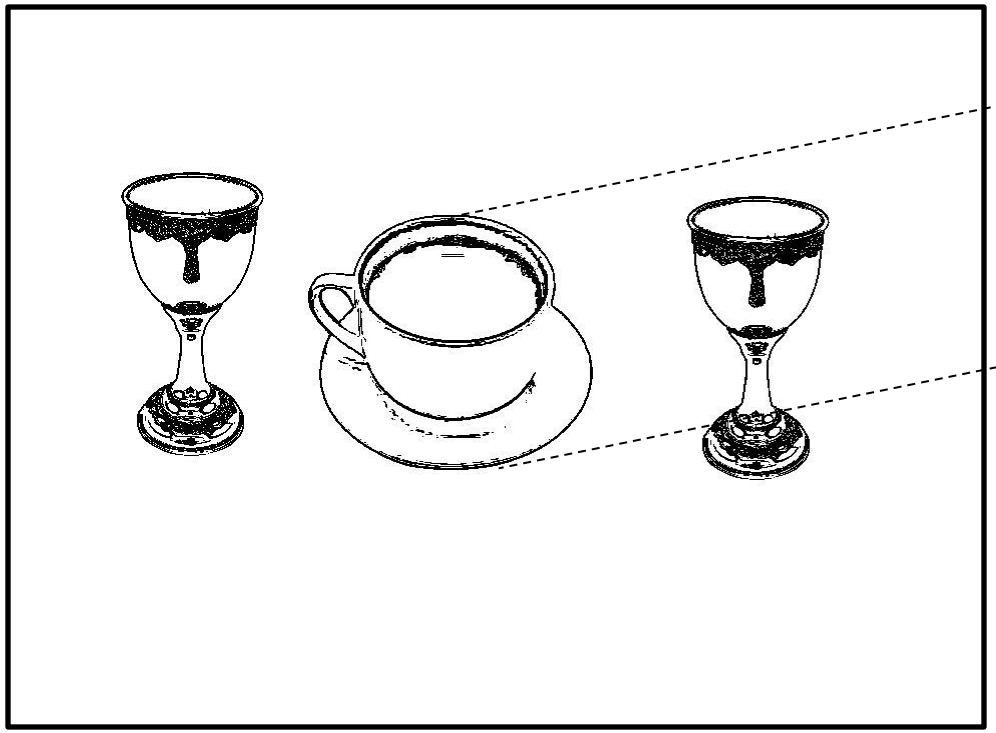
Dominant orientation templates

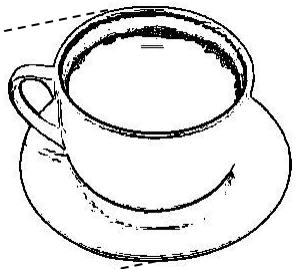
Šablóny význačných orientácii

- Jedna z najjednoduchších, ale účinných metód na rozpoznávanie objektov nepravidelného tvaru

Motivácia

šablóna





obraz

zamerajme sa
na hrany

Vstup: obraz z kamery

Tri polia r[h,w], g[h,w], b[h,w], každý ich prvok je číslo 0..255 a predstavujú červenú, zelenú a modrú zložku farby

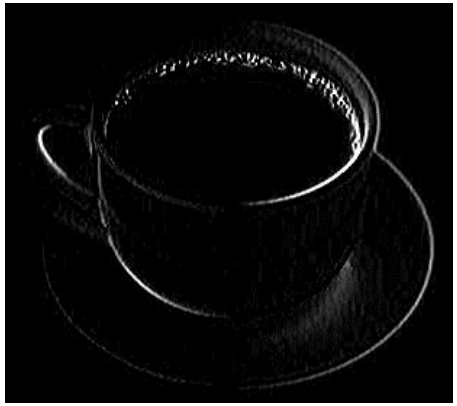
Čiernobiely obraz

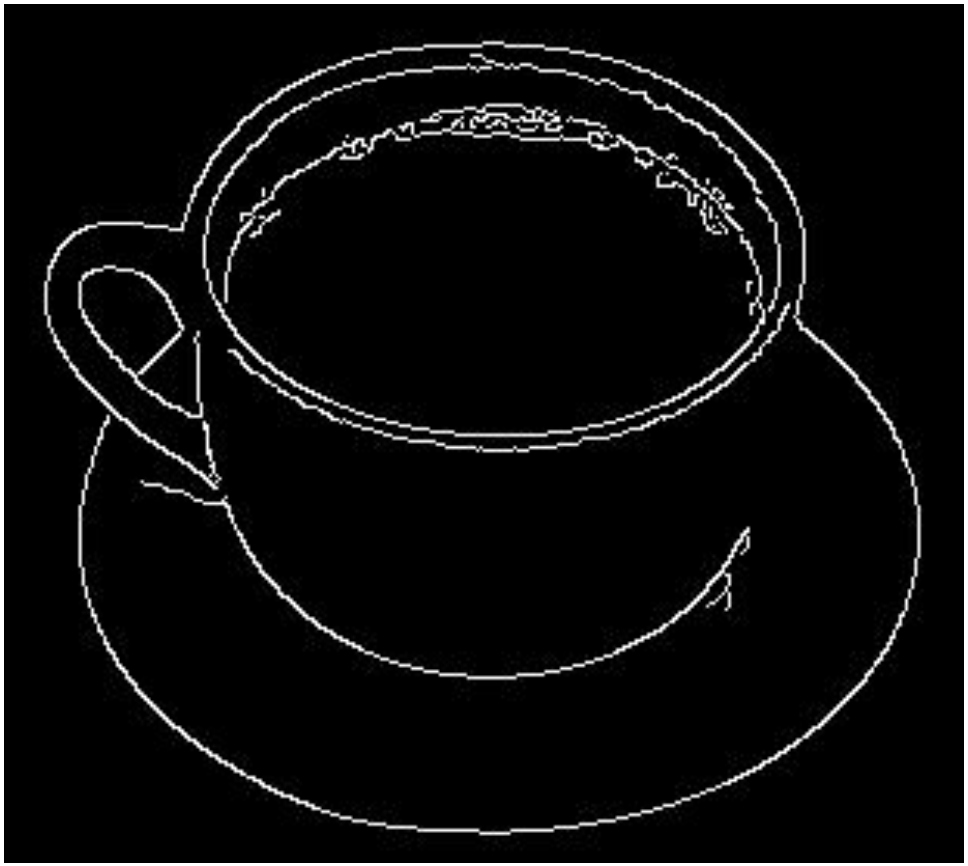
Pole bw[h,w], každý jeho prvok je číslo 0..255 a predstavuje intenzitu svetla

bw[i,j] = 0.3*r[i,j] + 0.59*g[i,j] + 0.11*b[i,j]

Hrany

dx

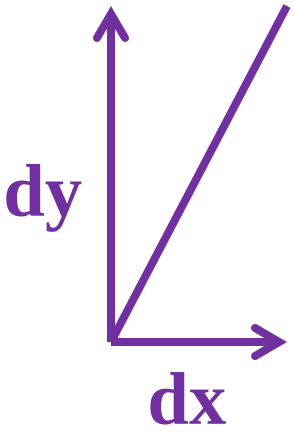


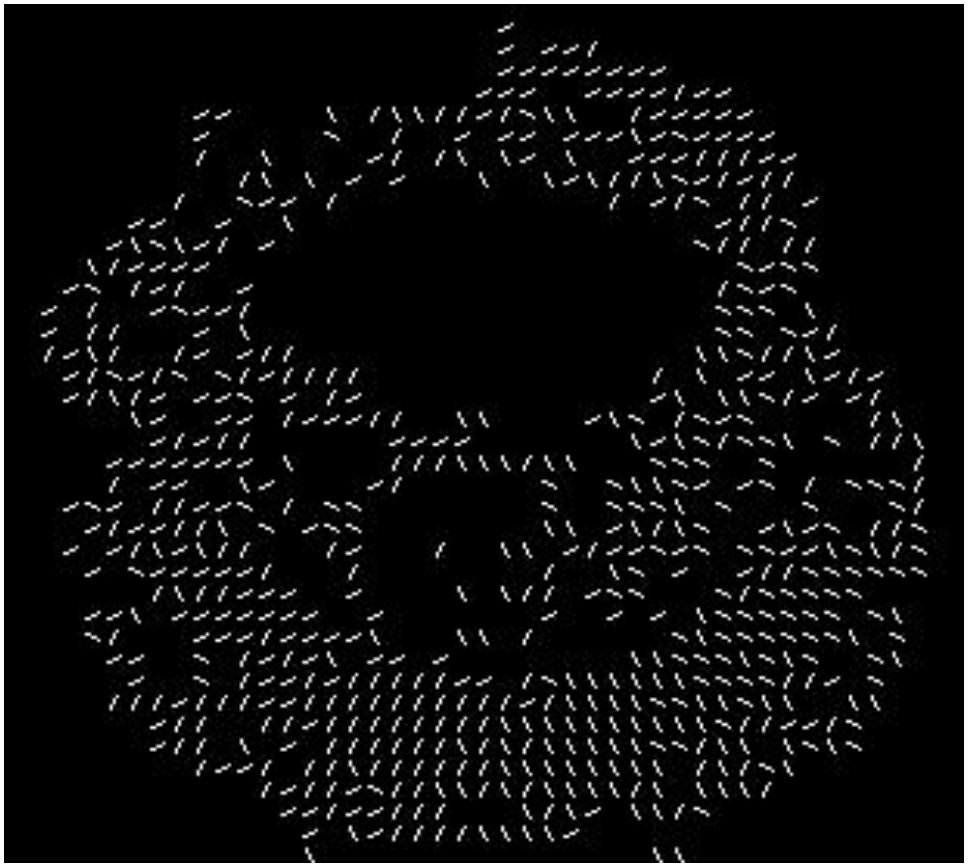


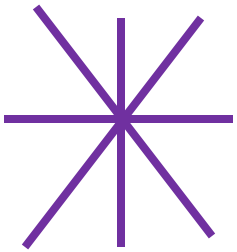
dy

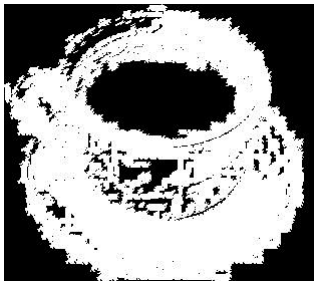
dx a dy môžeme zlúčiť a pomocou vhodného prahu premeniť obraz hrán na binárny. Stenčovaním čiar dostaneme výsledné hrany

Orientácie









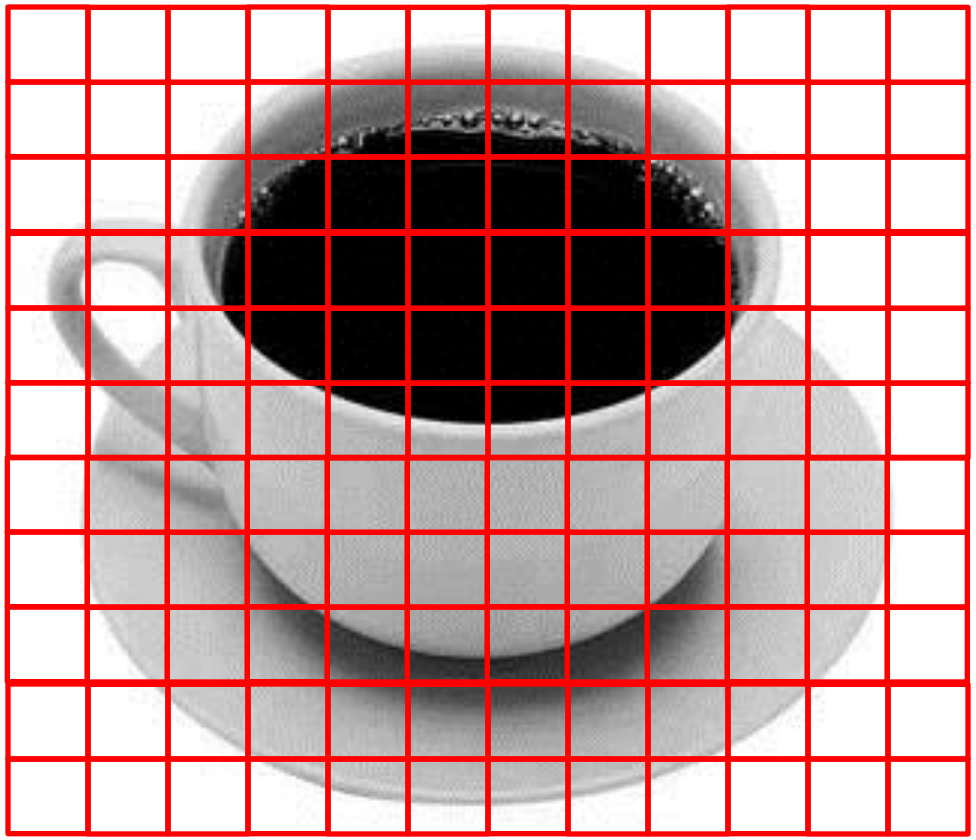
z dx a dy môžeme určiť orientácie, v každom bode smer v ktorom sa mení jas, pričom prechody z tmavého do svetlého a opačne považujeme za rovnaké

Šablóna

- Práve orientácie použijeme na zostrojenie šablóny rozpoznávaného objektu

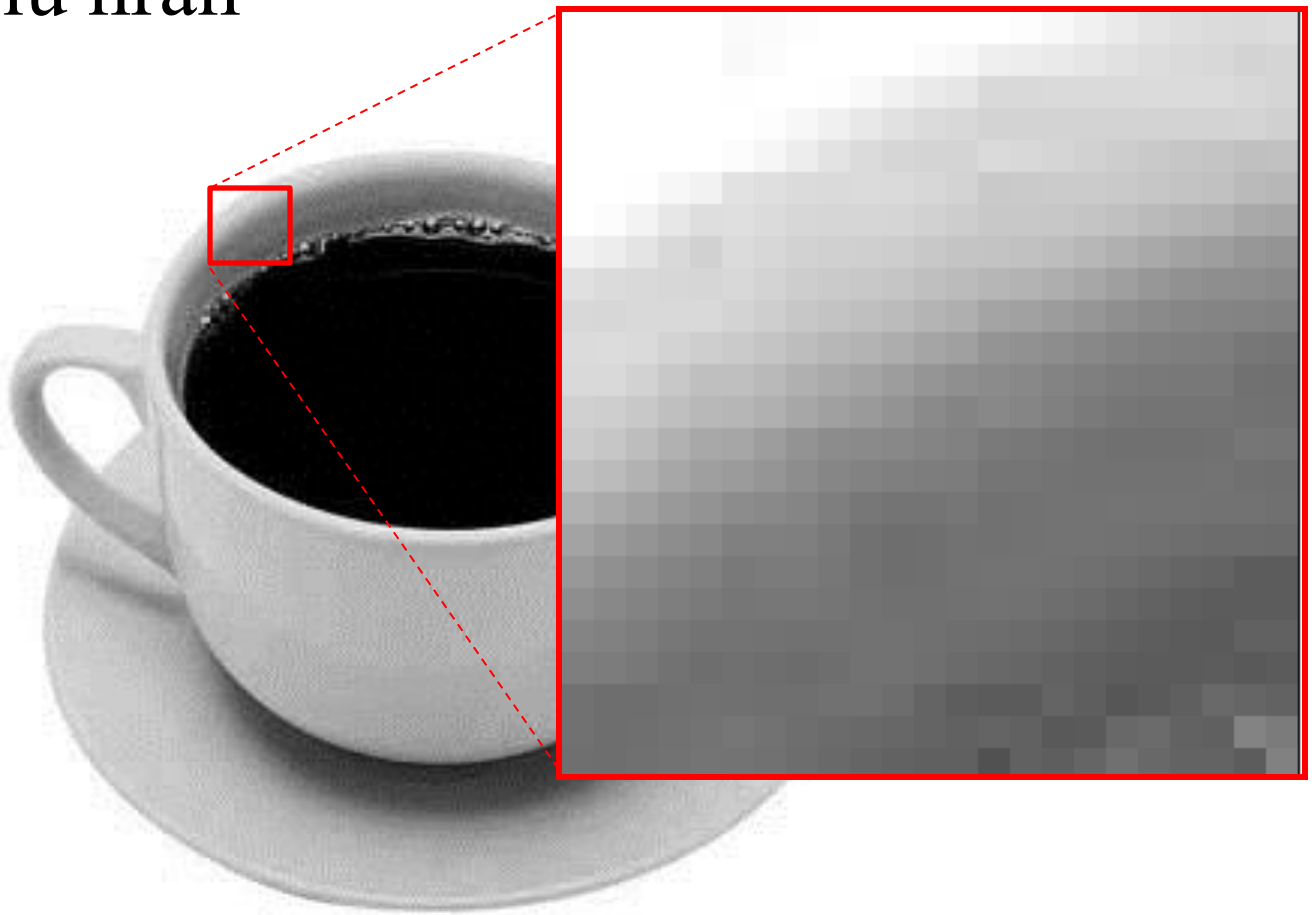
Šablóna

- objekt pokryjeme neprekrývajúcimi sa regiónmi



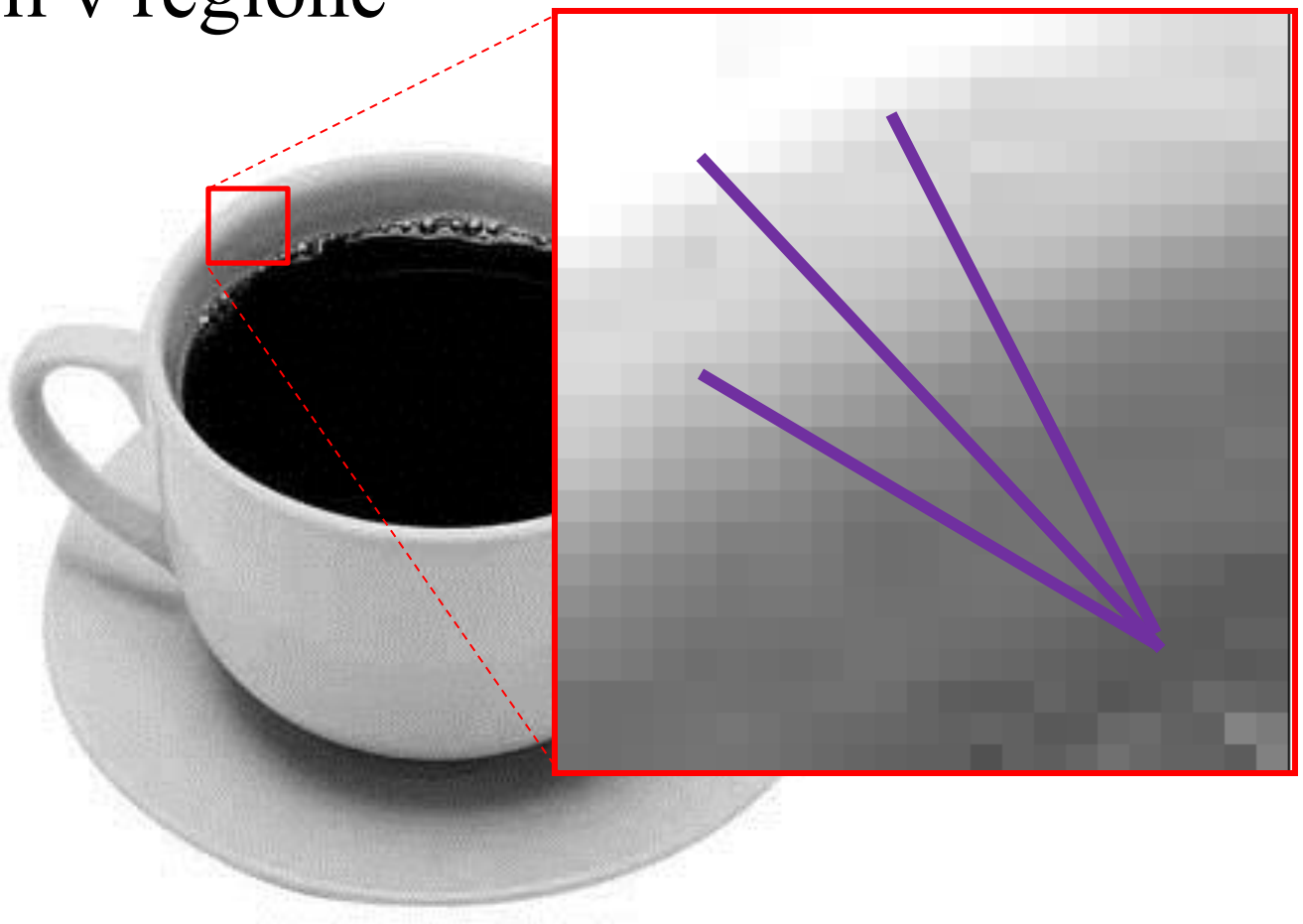
Šablóna

- Ku každému pixelu v regióne vieme orientáciu hrán



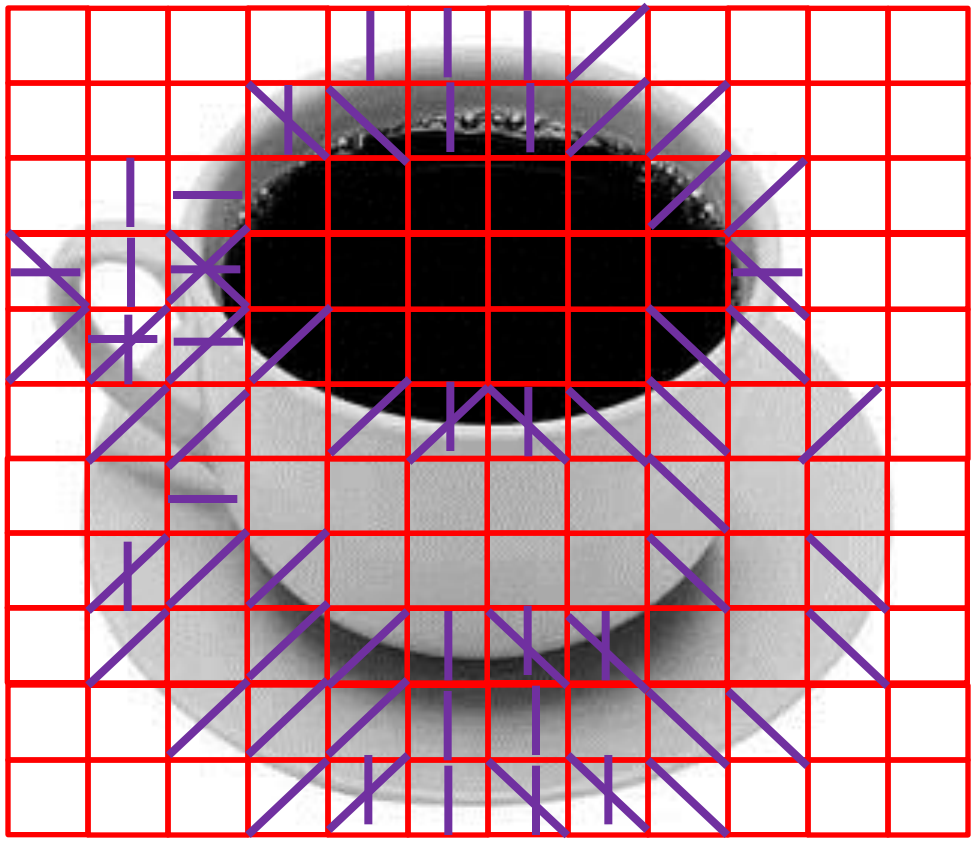
Šablóna

- Určíme z nich sadu prevládajúcich orientácii v regióne



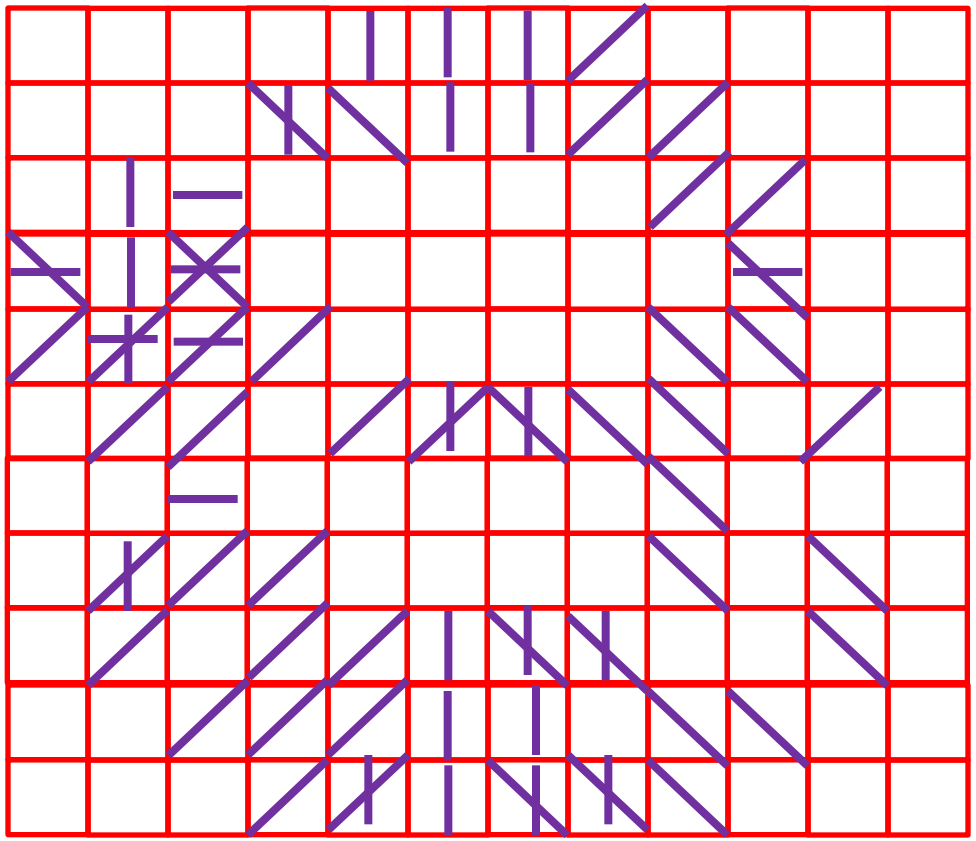
Šablóna

- Pre každý štvorček tak dostaneme sadu orientácií



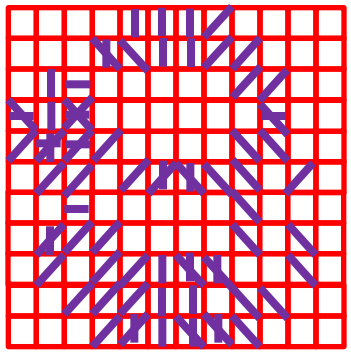
Šablóna

- A sada týchto orientácii bude tvoriť šablónu, tj. reprezentovať objekt



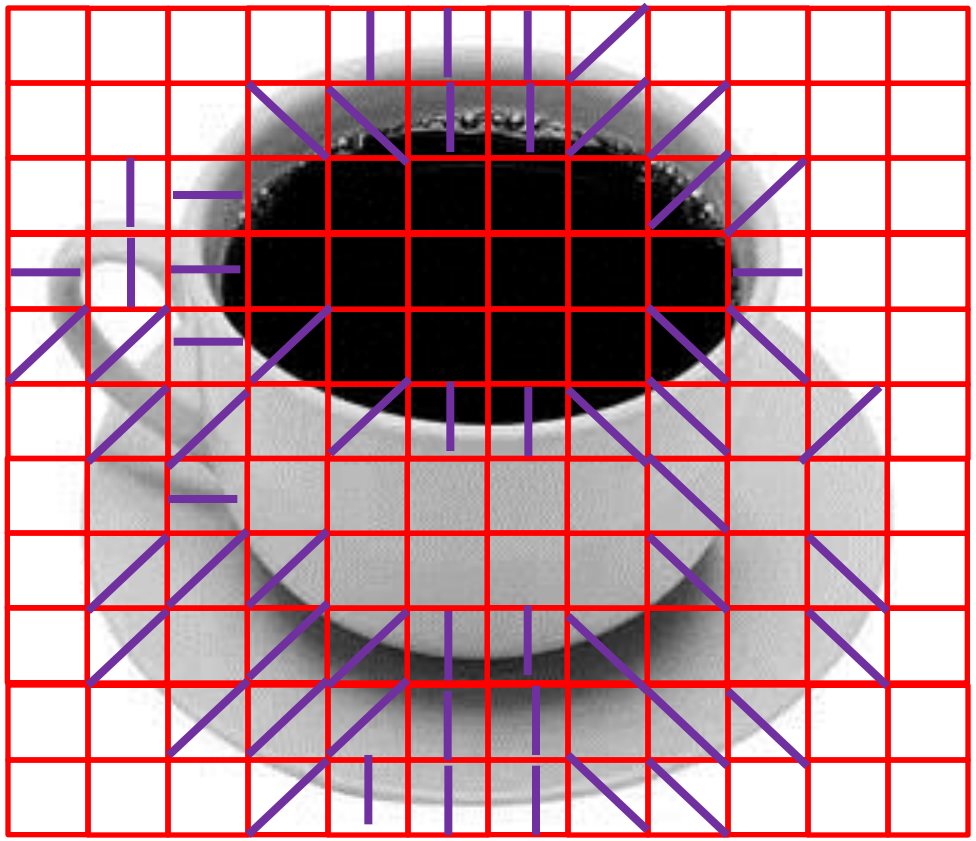
Hľadanie podľa šablóny

- Obrázok rozdelíme na regióny a v každom určíme jedinú, tzv. dominantnú orientáciu



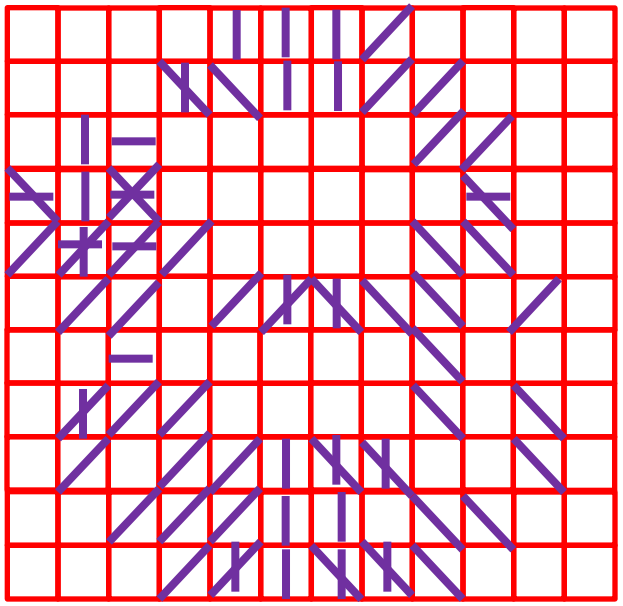
šablóna

obraz

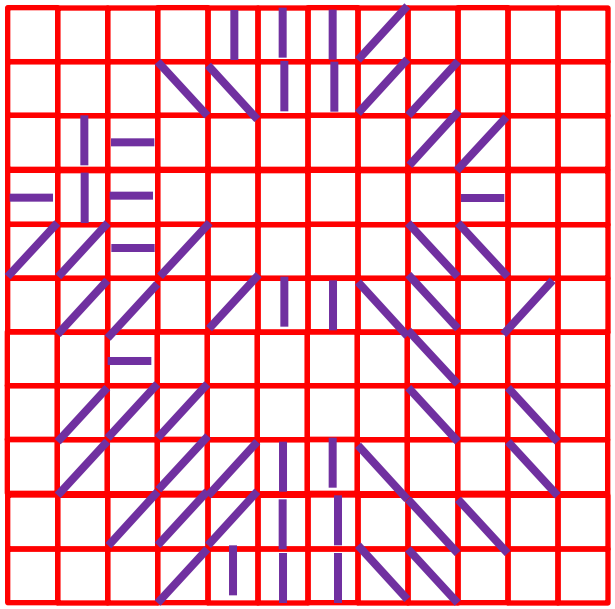


Hľadanie podľa šablóny

- Ak sa pre väčšinu regiónov nachádza dominantná orientácia v šablóne, tak OK.



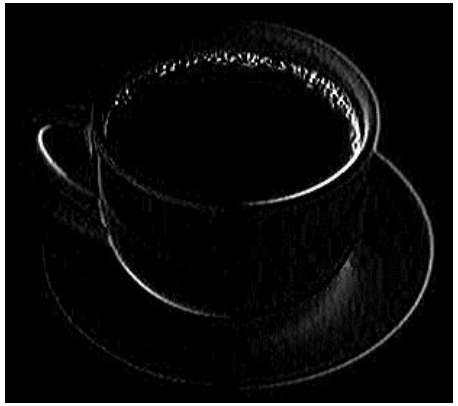
šablóna



obraz

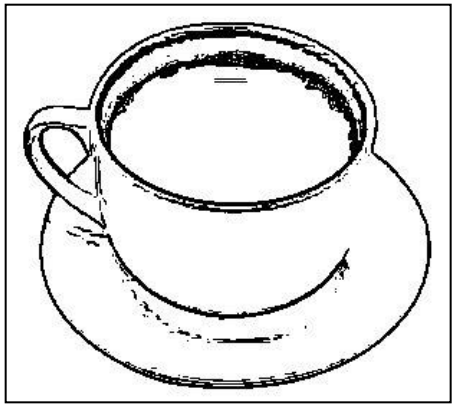
Reprezentácia nepravidelného tvaru

obraz

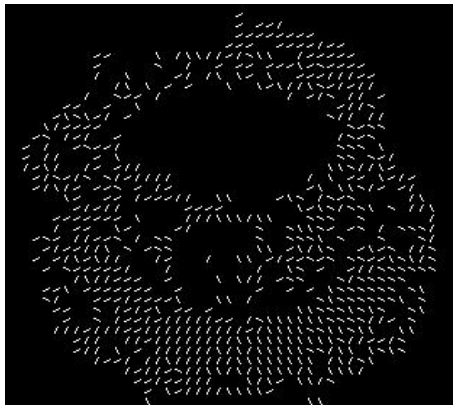


dx

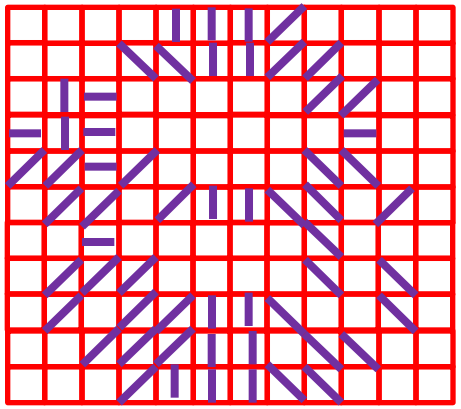
dy



hrany

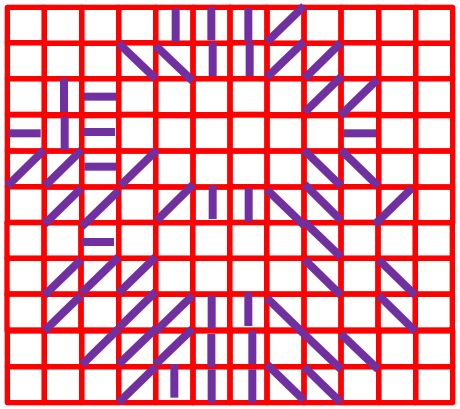


orientácie

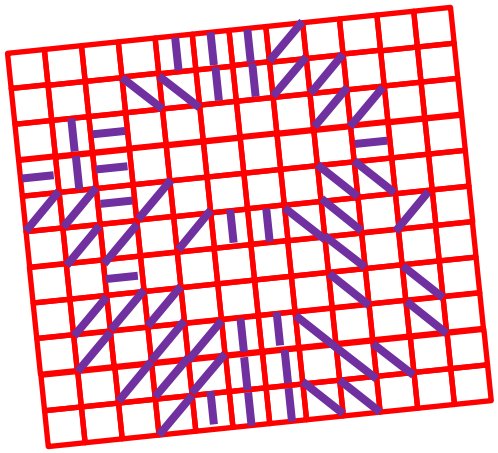


šablóny

Invariantnosť na otočenie a veľkosť



šablóna

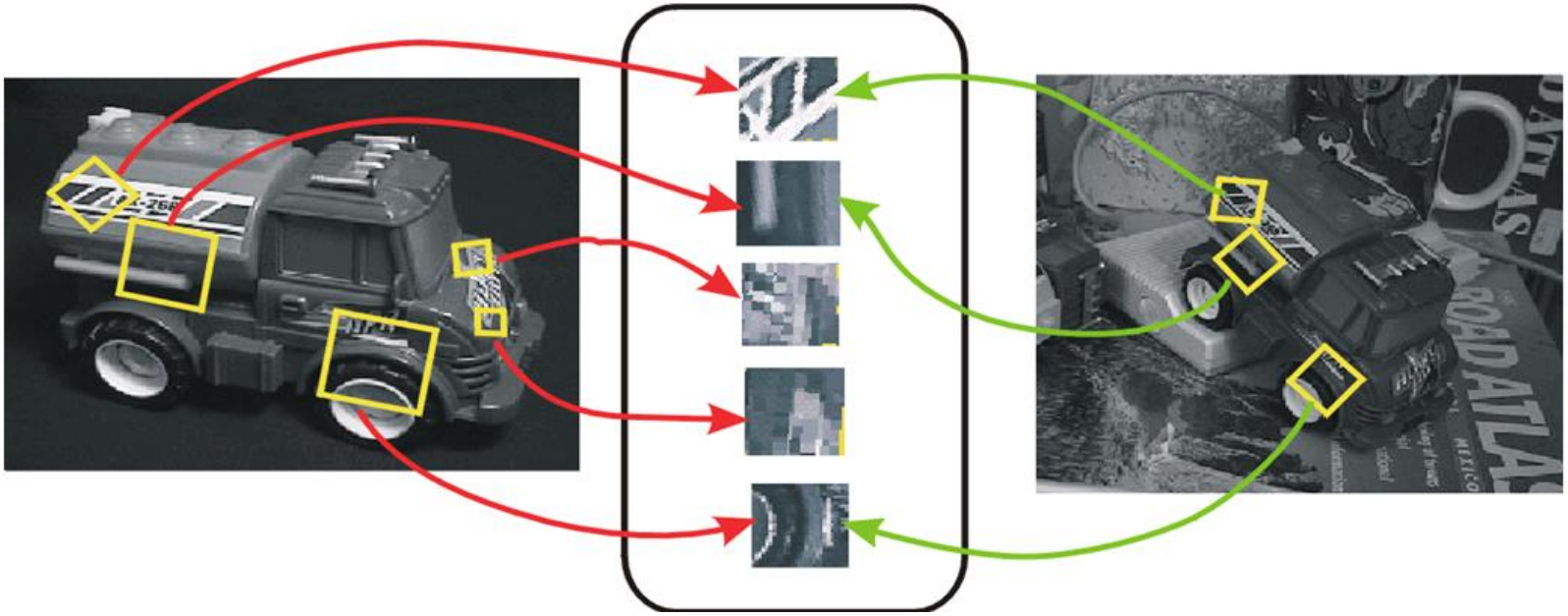


normalizovaná poloha

chýba bod otáčania → treba hľadať význačné body

SIFT (Scale Invariant Feature Transform)

Vzor predstavujú význačné body, ktorých okolie nachádzame v inej mierke a otočení na inom obraze



SIFT (Scale Invariant Feature Transform)

1 x podvzorkovať

8 x blur

1 x podvzorkovať

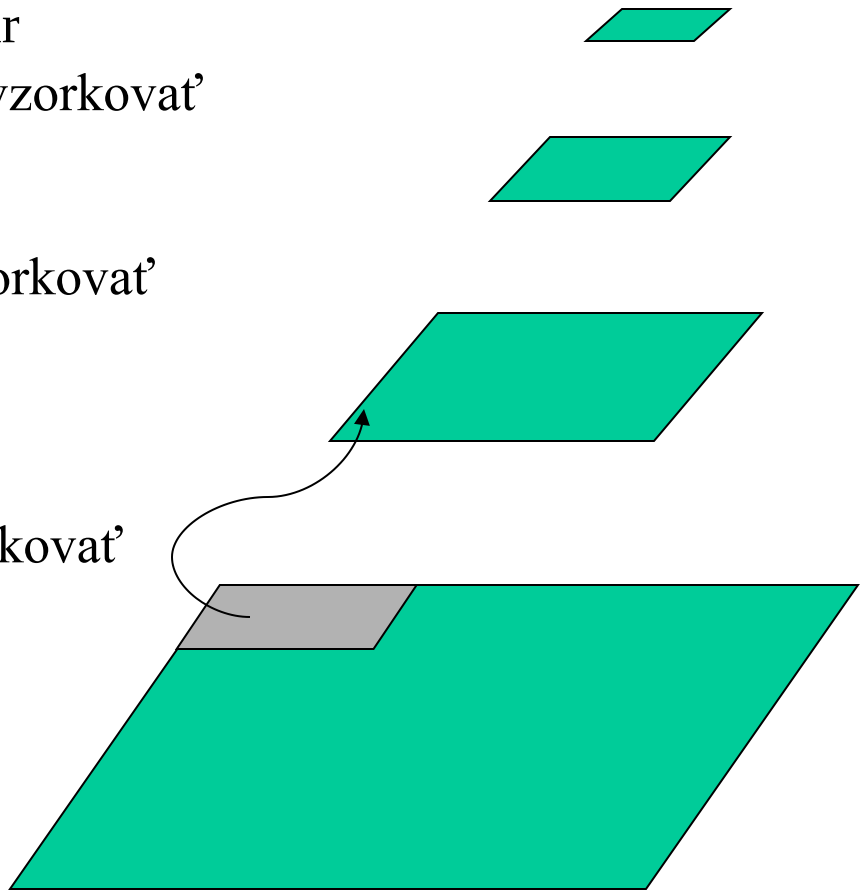
8 x blur

1 x podvzorkovať

8 x blur

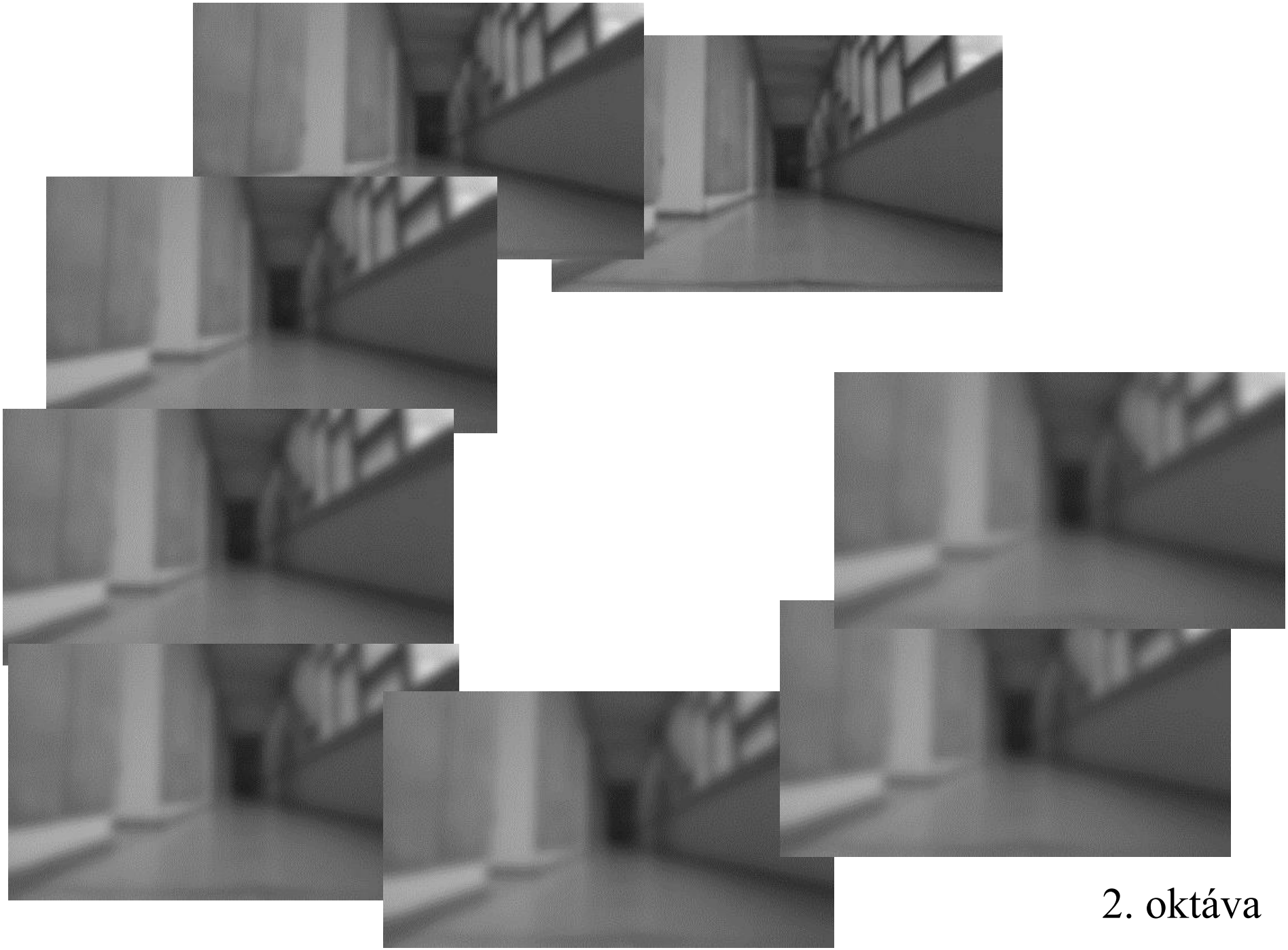
1 x podvzorkovať

8 x blur

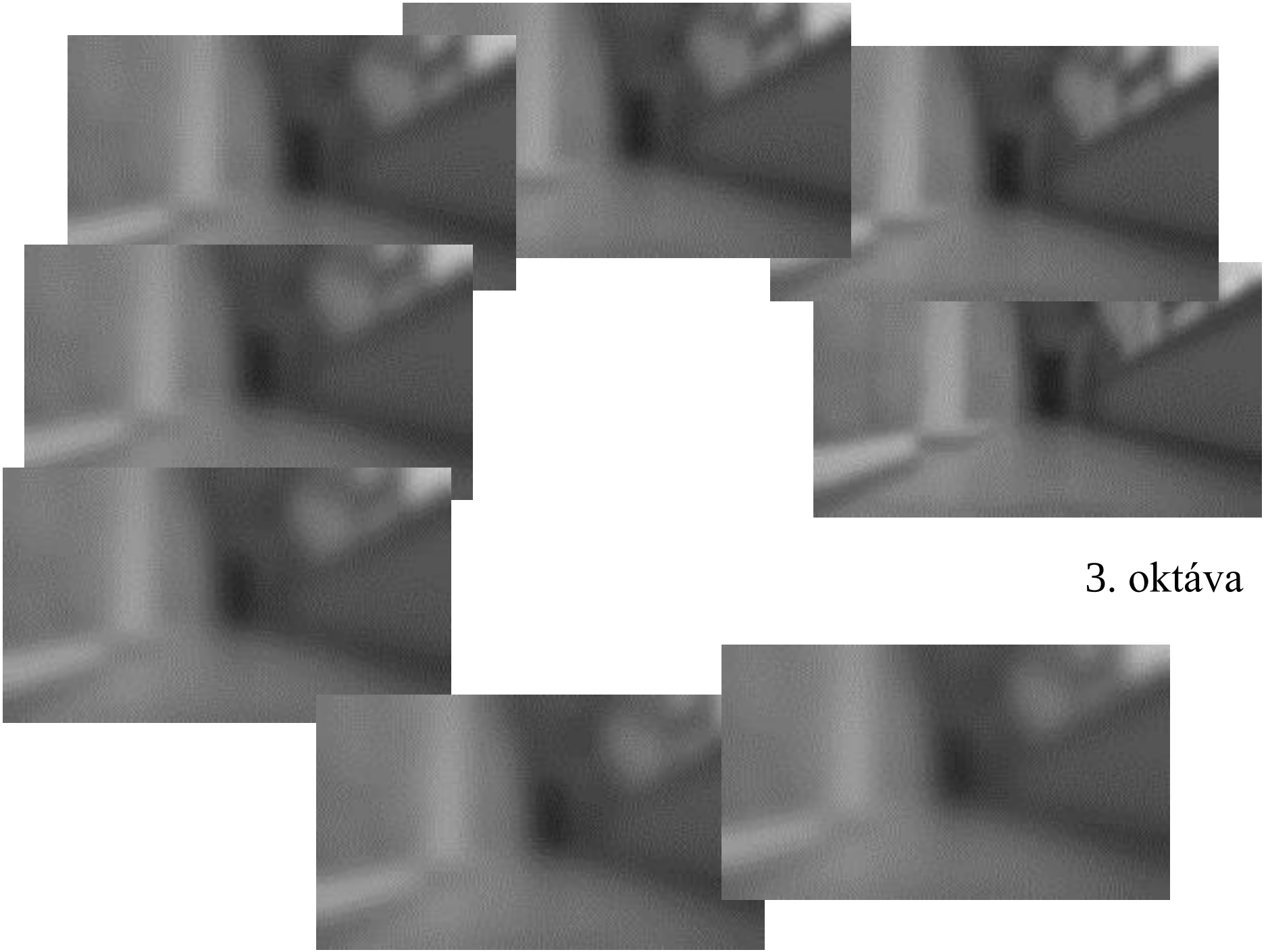


Význačné body spočítame ich ako doležité detaily, ktoré sa stratia pri podvozkovaní obrazu

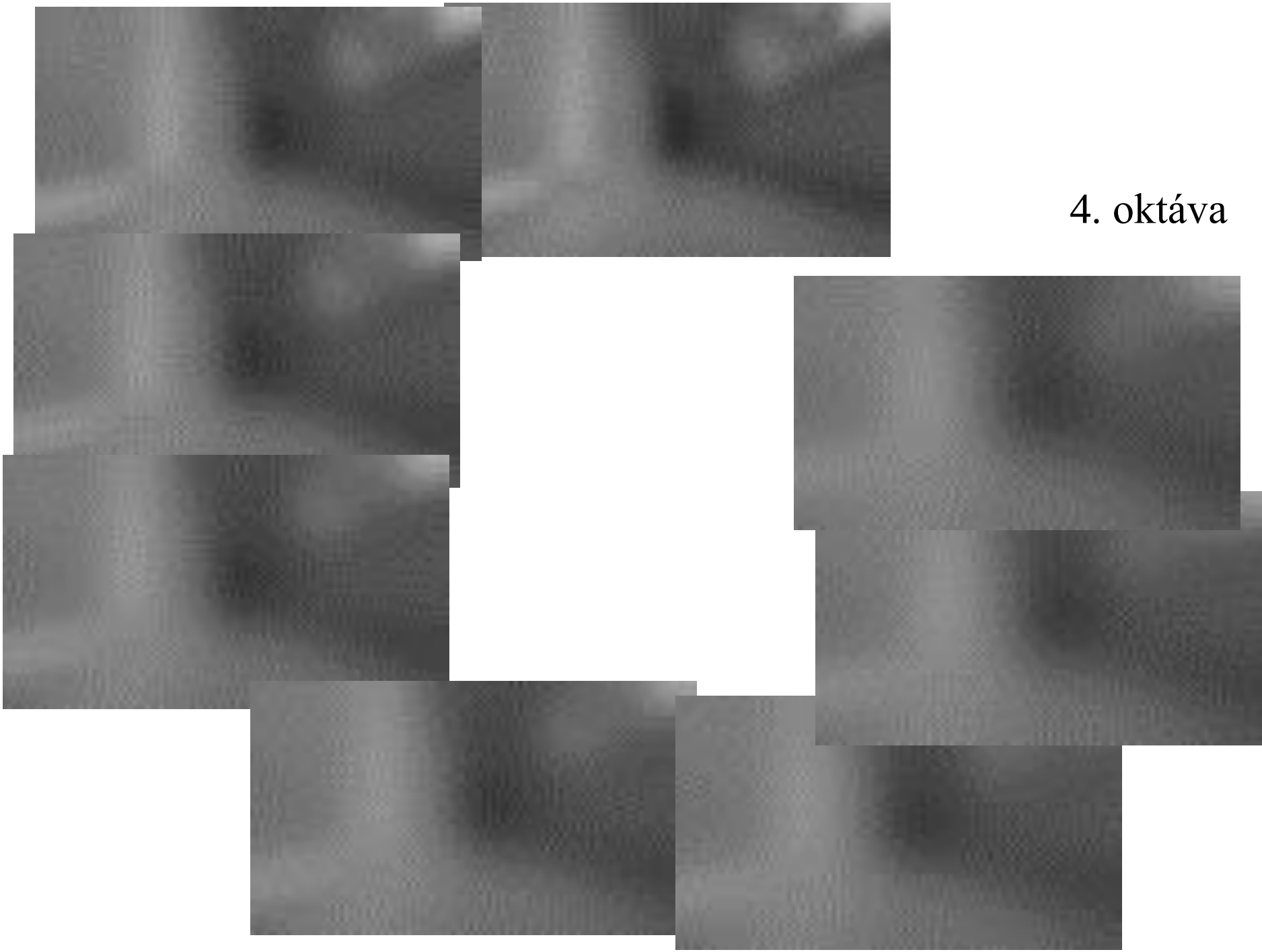
1. oktáva



2. oktáva

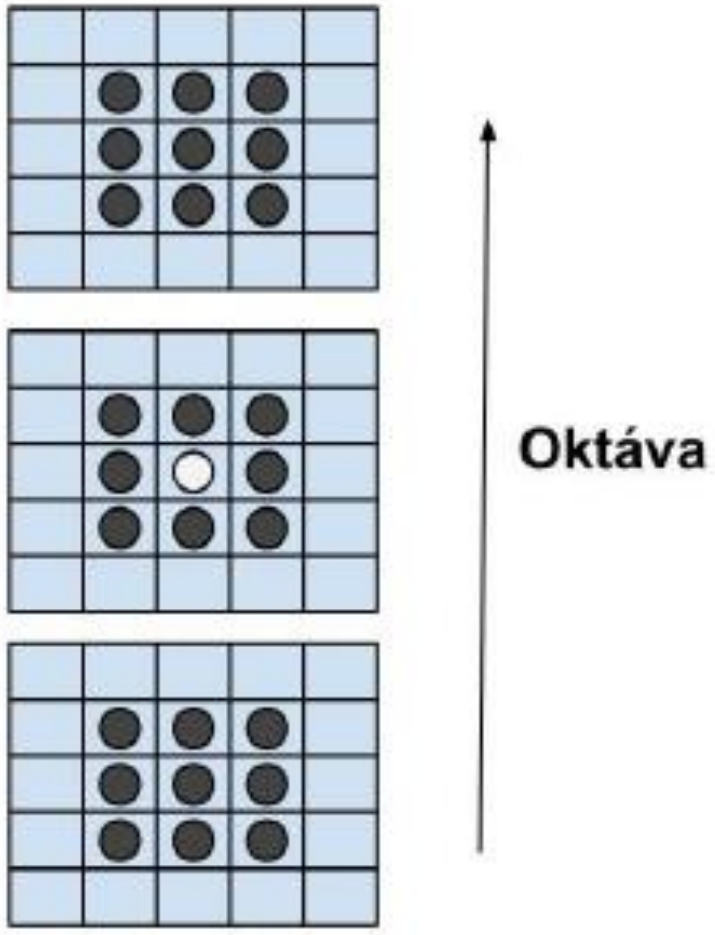


3. oktáva



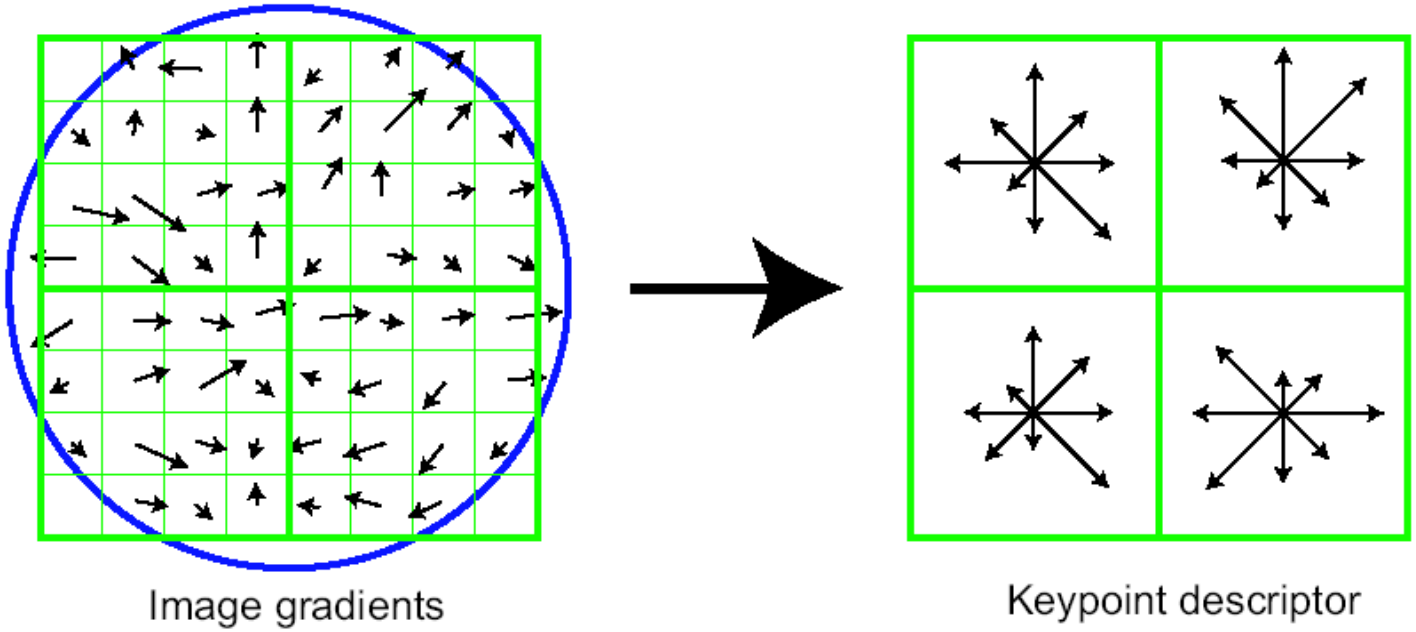
4. oktáva

SIFT (Scale Invariant Feature Transform)



Význačný bod hľadáme ako extrém v 3D

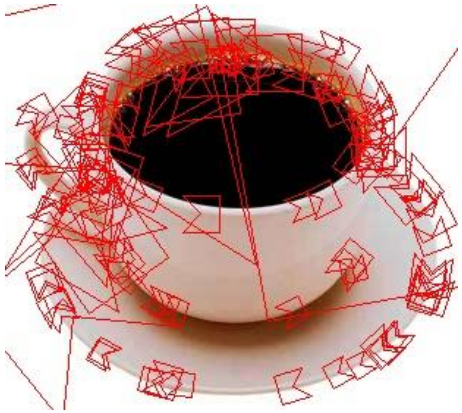
SIFT (Scale Invariant Feature Transform)



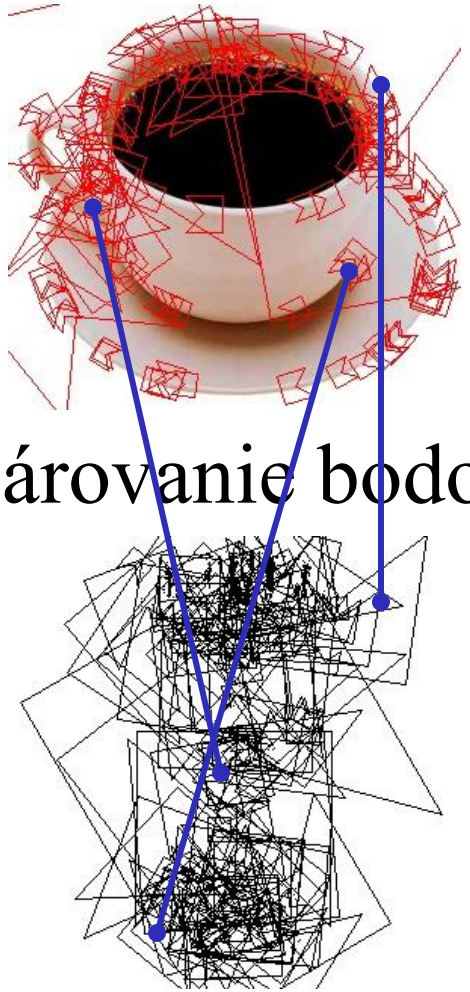
Pre každý bod urobíme popis jeho okolia – deskriptor (podobný šablóne)

Objekty s charakteristickými plochami

obraz

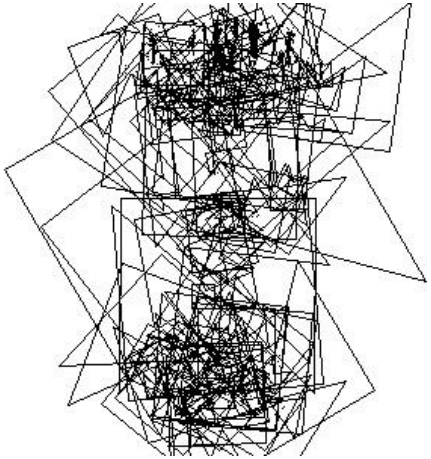


SIFT



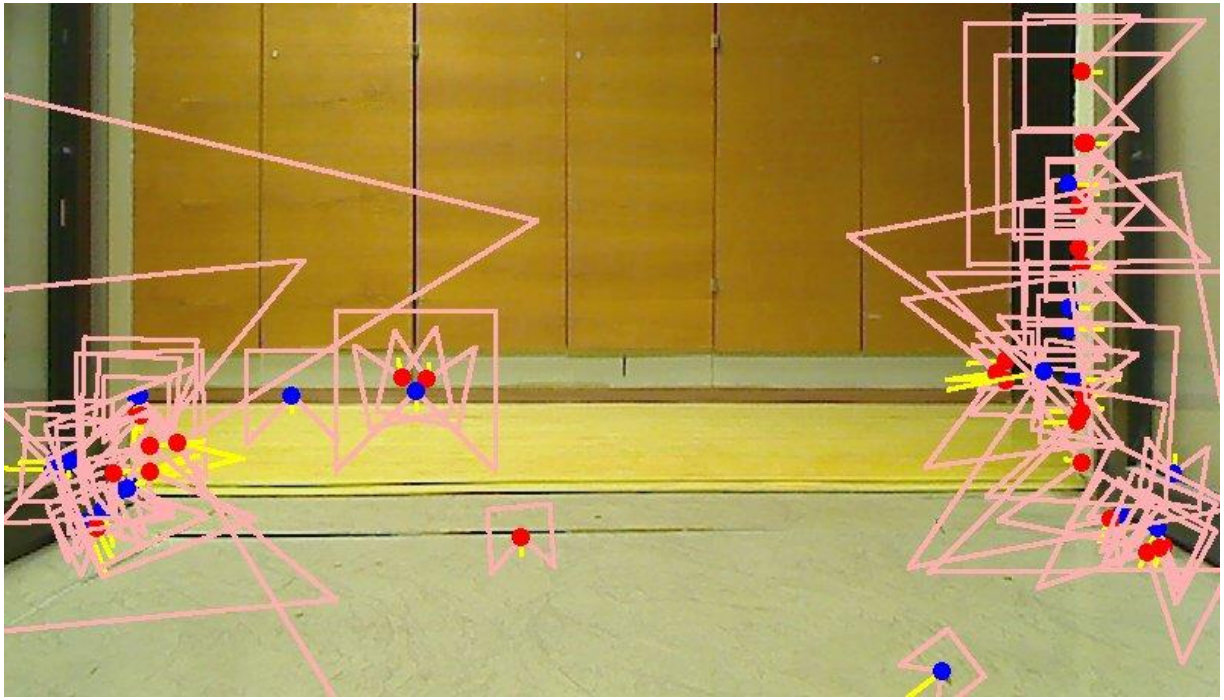
párovanie bodov

vzorový obraz



vzor

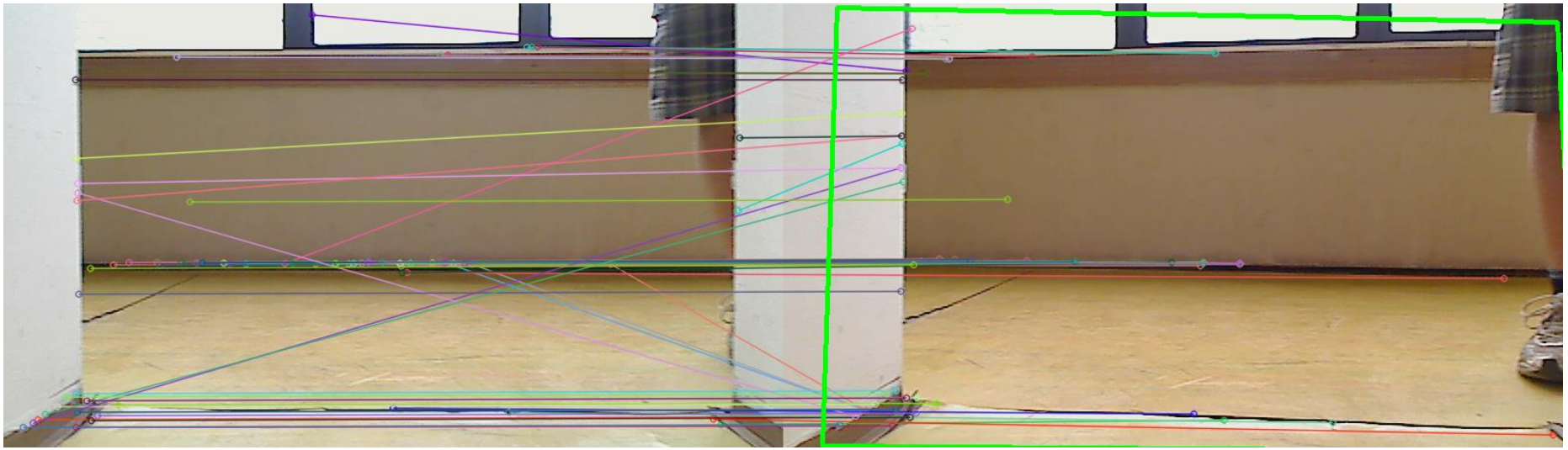
SIFT (Scale Invariant Feature Transform)



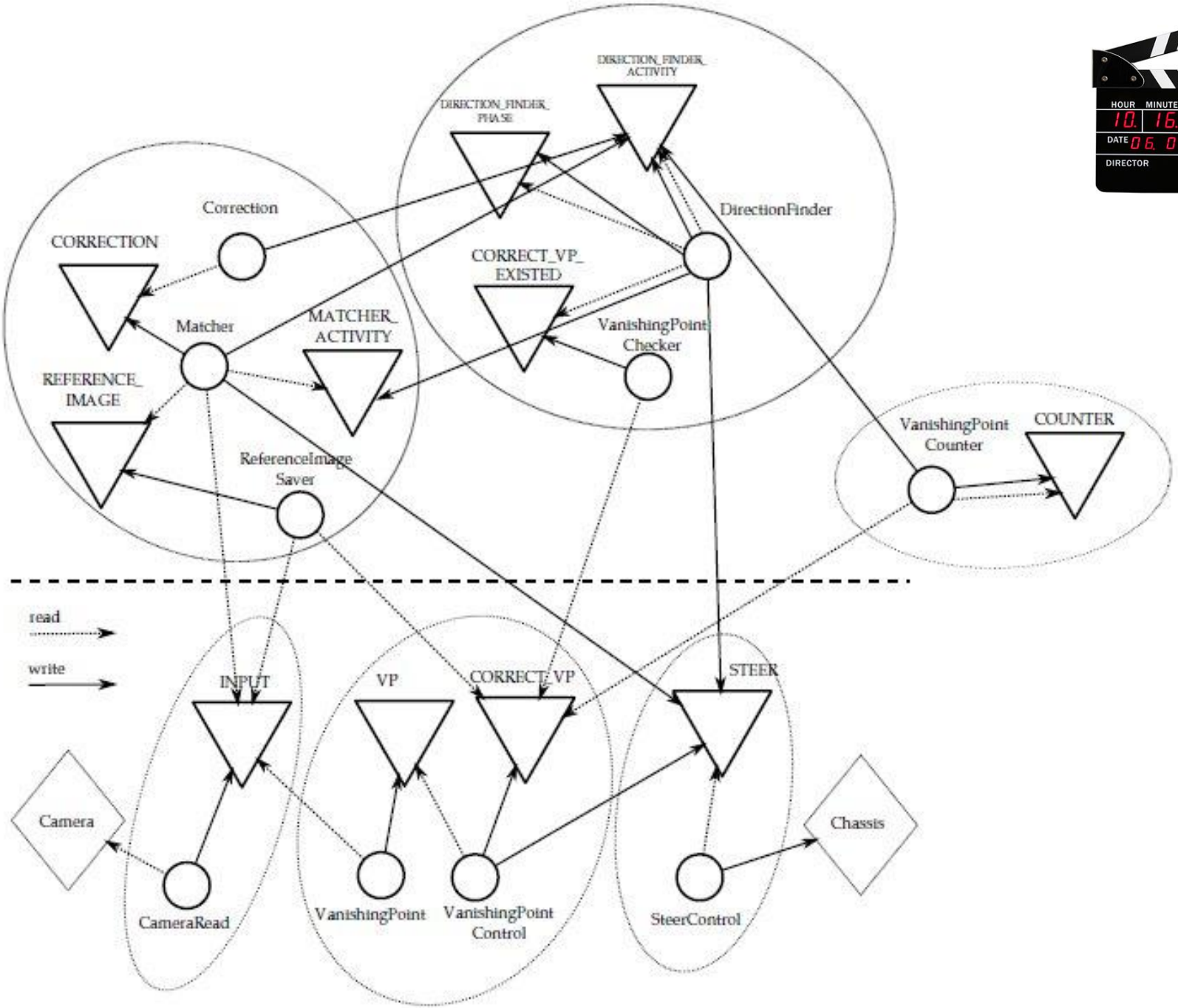
SIFT určí význačné body a popíše ich okolie deskriptorom

SIFT (Scale Invariant Feature Transform)

Tu si robot zapamätá, čo videl a počas otáčania vie z párovania bodov z detektoru SIFT o koľko sa otočil



Párovanie je založené na porovnávaní deskriptorov
Algoritmus RANSAC (RANdom SAmple Consensus)


DIRECTION_FINDER_ACTIVITY
DIRECTION_FINDER_PHASE
DirectionFinder
Correction
CORRECTION
CORRECT_VP_EXISTED
Matcher
MATCHER_ACTIVITY
VanishingPoint Checker
REFERENCE_IMAGE
ReferenceImage Saver
VanishingPoint Counter
COUNTER
read
write
INPUT
VP
CORRECT_VP
STEER
Camera
Chassis
CameraRead
VanishingPoint
VanishingPoint Control
SteerControl

Ďakujem za pozornosť !

Andrej Lúčny

Katedra aplikovanej informatiky

FMFI UK Bratislava

& MicroStep-MIS

andy@microstep-mis.com

www.microstep-mis.com/~andy