evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
Slezská univerzita v Opavě
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Zpracování obrazu

Studijní text k předmětu UF/0D201 - Snímání obrazu s využitím programovatelných logických polí a multiprocesorových systémů

RNDr. Hynek Sekanina,
Ing. Jaroslav Zeman, BcA. Bc. Miroslav Zeman

Slezská univerzita v Opavě
Filozofickou-přírodovědecká fakulta v Opavě
Ústav fyziky

Snímání a úpravy obrazu

Studijní text k předmětu UF/0D201 - Snímání obrazu s využitím programovatelných logických polí a multiprocesorových systémů

RNDr. Hynek Sekanina, Ing. Jaroslav Zeman, BcA. Bc. Miroslav Zeman

Ústav fyziky
Filozofickou-přírodovědecká fakulta v Opavě
Slezská univerzita v Opavě
Bezručovo náměstí 13, Opava

Tato inovace předmětu UF/0D201 - Snímání obrazu s využitím programovatelných logických polí a multiprocesorových systémů je spolufinancována Evropským sociálním fondem a Státním rozpočtem ČR, projekt č. CZ.1.07/2.2.00/28.0014, „Interdisciplinární vzdělávání v ICT s jazykovou kompetencí“.

Anotace:

Tato publikace je určena studentům, kteří mají zájem snímání a zpracování obrazu za účelem využíti obrazové informace v řídících systémech. Obsahem publikace je studijní text pro předmět „UF/0D201 -Snímání obrazu s využitím programovatelných logických polí a multiprocesorových systémů“. Text podává studentům základní informace z oblasti snímání obrazu a jeho digitálním zpracování, tak aby výstup zpracování obrazu mohl být využit v řídícím či rozhodovacím systému. V studijním textu jsou uvedeny teoretické základy pro zpracování obrazu, tak aby bylo možné navrhnout program či HW komponentu na bázi FPGA, který s obrazem provede všechny potřebné manipulace, přičemž jeho výstupem bude datový údaj použitelný pro účely řízení a rozhodování.

Obsah

1 Snímání obrazu

RYCHLÝ NÁHLED KAPITOLY

V této kapitole je ozřejmeno jak je snímán obraz pomocí prvků CCD a CMOS.

CÍLE KAPITOLY

Po prostudování této kapitoly porozumíme:

- konstrukčnímu uspořádání snímačů,
- technickým rozdílům mezi snímači, které se promítají do zpracování obrazu.

KLÍČOVÁ SLOVA KAPITOLY

CCD, CMOS, dělení CCD snímačů FT, IT, rastrové filtry, barevné modely

Nejběžnějšími komerčně dostupnými snímači pro snímání obrazu jsou technologie na bázi CCD a CMOS technologie. Obě technologie využívají fotoelektrický jev v polovodičích a polovodičových strukturách typu PN, jen různým způsobem, viz dále.

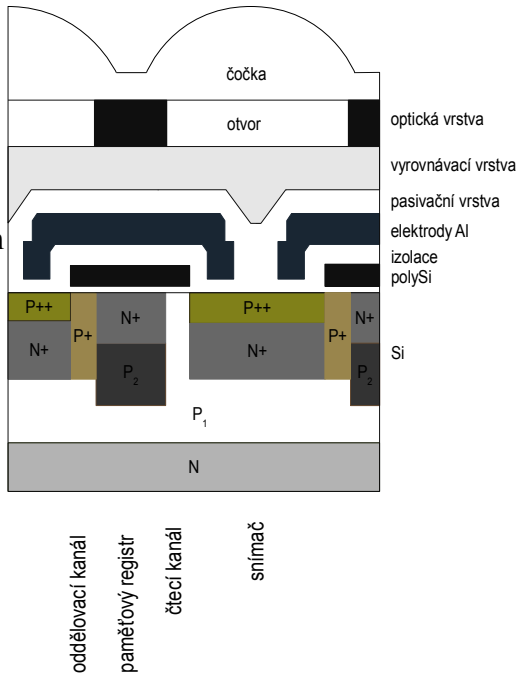
Vlastní polovodiče (Si, Ge) jsou upravovány přidáním příměsí (např. P, In) na nevlastní polovodiče typu P a N a pomocí nich jsou vytvářeny struktury zvané PN přechody.

Fotony, které nesou obrazovou informaci snímané scény, vždy nesou i energii, a potom, co dopadnou do polovodiče v případě zániku předávají svoji energii atomům v polovodiči. Atom, který absorbuje energii fotonu, při splnění definovaných podmínek, ze svého elektronového obalu uvolní elektron (tento elektron označujeme jako volný a může se volně pohybovat v polovodiči). Volné elektrony pak ovlivňují fyzikální vlastnosti polovodiče, např. vodivost polovodiče[1]. V případě struktur PN pak volné elektrony ovlivňují chování této struktury, nejčastěji vodivost.

[1] V PN přechodu diody, podle toho v jakém režimu dioda pracuje, buď volné elektrony ovlivňují vodivost a dioda pracuje jako senzor, v tomto případě je na diodu přivedeno závěrné napětí, nebo jsou elektrony odváděny mimo PN přechod, pak dioda pracuje jako fotočlánek.

V zásadě finální snímače CCD a CMOS mohou poskytovat informace o snímaném obraze jak v analogové formě, ale v současnosti zpravidla již v digitální formě, avšak vždy pro součinnost vyžadují vnější řídící obvod, který generuje potřebné řídící signály pro snímač, jde zejména o signály řídící chod pohybu náboje v CCD snímačích a chod A/D převodníků a v neposlední řadě i chod elektronických závěrek CCD snímačů, v případě CMOS snímačů pak chod časování čtení z jednotlivých prvků snímače (řádky, sloupce) včetně nulování a chod A/D převodníku. Samotná zpracování obrazu z hlediska barevného modelu a jeho korekcí včetně komprese již zpravidla probíhá mimo vlastní snímač zpravidla s využitím proprietárních obvodů[2] v případě komerčních produktů - fotoaparáty, TV kamery.

Existuje i celá řada snímačů, které používají i fotoelektrický jev bez použití polovodičů, např. vidikony, optokony, optické násobiče, ty však jsou používány jen v úzkém specifickém oborovém rozsahu (např. CT) a proto zde nebude uváděná jejich konstrukce.



Obr. 1. Řez strukturou CCD prvku

1.1 CCD snímače

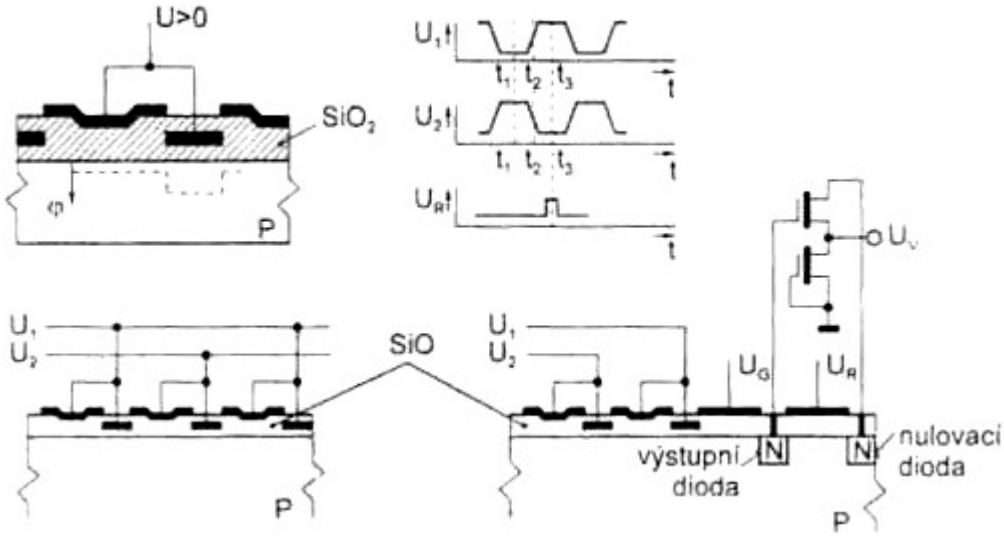
V CCD (Charge-Coupled Device) technologii je určující pro zjišťování množství zachycených fotonů ve snímači množství volných elektronů, které tyto fotony uvolní, a toto volné množství elektronů bude prostřednictvím svého náboje na dané ploše snímače určovat intenzitu dopadajícího světla, tzn. náboj bude úměrný intenzitě dopadajícího světla.

Svobodnému přemísťování volných elektronů v CCD snímači brání důmyslný systém elektrod a PN přechodů včetně vrstev izolantů mezi těmito strukturami - viz Obr. 1. (výrobní proces zahrnuje řádově 300 operací) ale i správné časování a velikosti napětí přiváděných na jednotlivé elektrody - viz Obr. 2. a Obr. 3. Elektrody (zpravidla z napařeného hliníku nebo

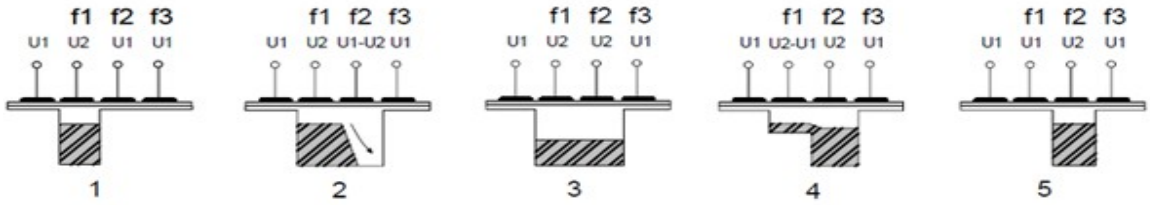
[2] Z hlediska výpočetního se jedná zpravidla o upravené procesory, ve kterých je kladen důraz na rychlost výpočtů v oblasti zpracování obrazu, tzn. jsou v něm zpravidla přítomny různé urychlovací komponenty na bázi DES.

polykrystalického křemíku - ten se chová jako vodič), jsou na obou stranách snímače (spodní strana je zpravidla celistvá, horní strana je tvořena několika mřížkami).

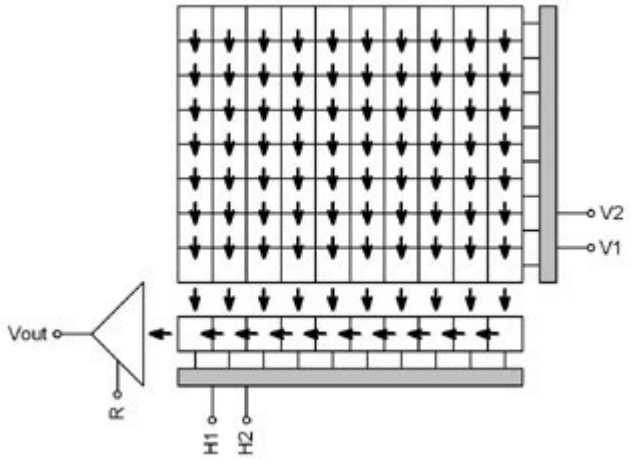
Celkové uspořádání snímače včetně elektronických obvodů řídících chod snímače je na Obr. 5.



Obr. 2. Akumulace náboje v CCD prvku a řídící napětí pro pohyb náboje



Obr. 3. Příklad 3 fázového řídícího napětí pro pohyb náboje v CCD prvku

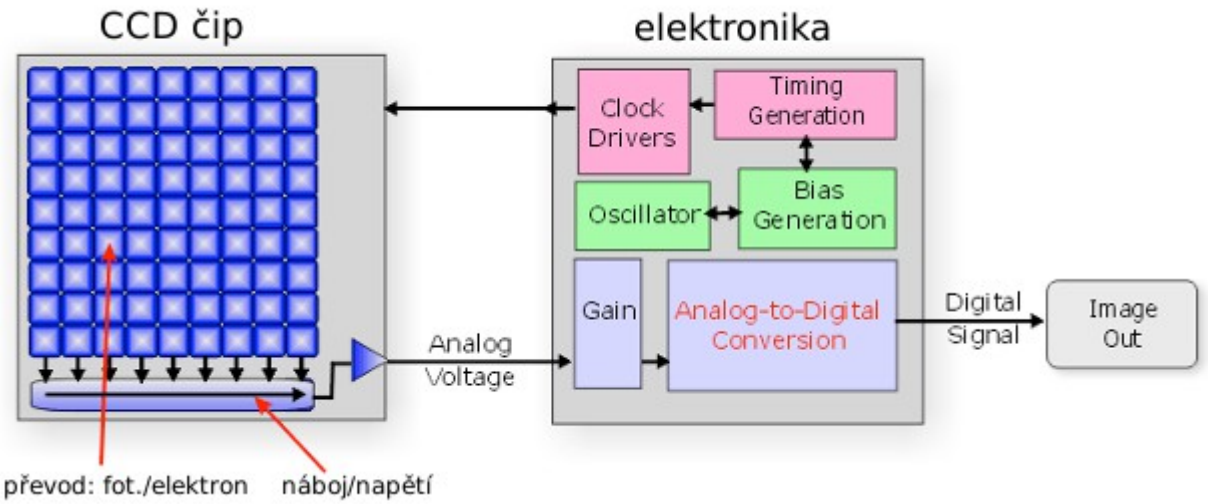


Obr. 4. Způsob pohybu náboje na ploše CCD FF čipu

CCD snímač pracuje ve třech fázích:

1. **fáze mazání** – v této fázi je veškerý volný náboj ze snímače odstraněn, to je zajištěno pomocnou elektrodou, která odvede veškeré volné elektrony ze snímače,

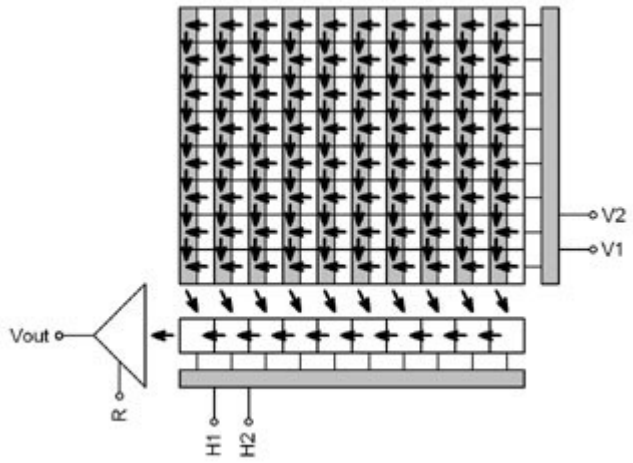
2. **fáze expozice** – v této době necháme na snímač dopadat světlo a akumulujeme volné elektrony ve snímači - náboj, v této fázi bývá na jednu z vodivých mřížek přivedeno kladné napětí, přičemž toto napětí způsobí, že záporný náboj elektronů se bude shromažďovat právě pod touto mřížkou,



Obr. 5. Celkové uspořádání elektroniky CCD

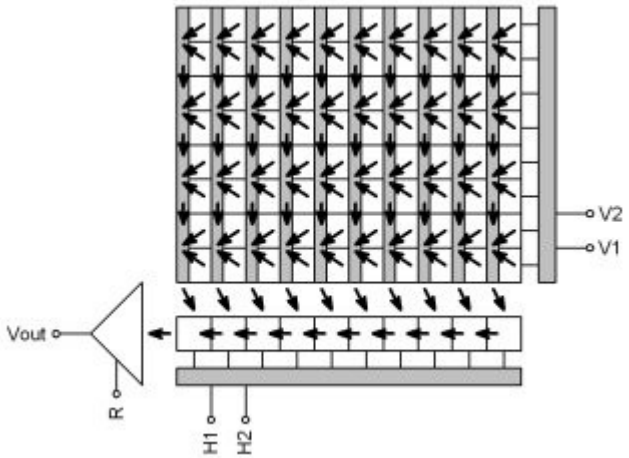
3. **fáze snímání** – v této fázi postupně na jednotlivé vodivé mřížky přivádíme kladné vícefázové napětí[3] - viz Obr. 2. a Obr. 3, přičemž toto fázování zabezpečí, že volný náboj z plochy snímače budeme postupně posouvat v daném směru, a to ke kraji snímače, kde budeme toto množství volného náboje snímat snímací elektrodou.

Ve fázích 1 a 3 je nutno zamezit dopadu světla na snímač, jinak by byl obraz degradován. V kamerách, kde je používána mechanická závěrka, se používají čipy CCD FF – full frame - FT - viz 4., zde je snímán obraz z celé plochy čipu (čipy mají vyšší citlivost). Ve spotřební elektronice jsou však mechanické závěrky nahrazeny elektronickými závěrkami, které jsou realizovány přímo na snímači CCD v podobě oddělených oblastí pro snímání (SA – storage area) a expozici (označené IA – image area) označujeme je jako IT - Obr. 6. Proto v konstrukci CCD snímače rozlišujeme snímače na FT nebo IT (frame transfer či interline transfer). V IT ve fázi 2 (expozice) se obraz exponuje v IA, pak je exponovaný obraz rychle přenesen do SA oblasti, odkud je ve fázi 3 (snímání) postupně snímán. S FT a IT snímáním i tak souvisí, jak je načítán náboj z SA či IA, kdy může být načítán jako půl snímku, celé snímky, nebo jako součet z více snímacích prvků – binning - viz Obr. 7., odtud nesou čipy označení jako interlace, grogresive, frame read.



Obr. 6. Uspořádání a pohyb náboje na CCD IT

[3] Průběh napětí je přesně definován a může se jednat o 2, 3 a 4 fázové posouvání.

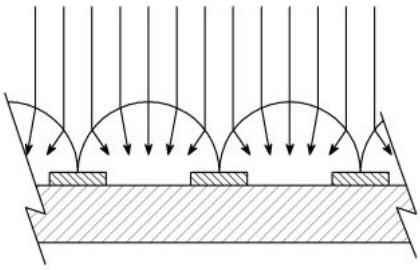


Obr. 7: Pohyb náboje v případě bindigu na CCD čipu

Snímací, závěrkové a resetovací elektrody na ploše snímače vytvářejí samostatné oddělené oblasti – jednotlivé pixely, kterými je snímaný obraz v digitální podobě reprezentován.

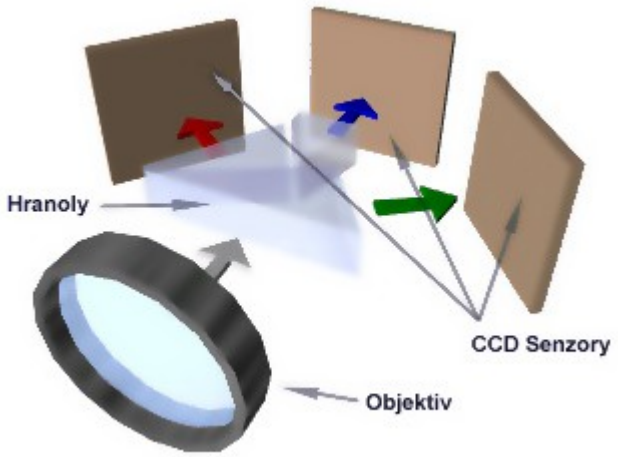
Způsob posunu náboje ve snímači, počet míst, kde se náboj snímá, určuje max. rychlost snímání obrazu ze snímače, resp. kolik snímků za jednotku času ze snímače jsme schopni přenést, proto výrobci snímačů využívají různých technik, aby dobu snímání zkrátili, zpravidla přidávají na počtu snímačů nábojů, tzn. snímač je rozdělen na regiony a každý region má svoji řídící a převodní část.

Mřížky, které se nachází na povrchu polovodičové destičky CCD snímače, jsou zpravidla nepropustné pro světlo (zpravidla se jedná o napařený hliník), takže citlivost snímače a množství náboje, který snímáme, je mnohonásobně menší, než by bylo možno získat z čisté plochy snímače, proto výrobci nad těmito mřížkami mohou ve výrobním procesu realizovat miniaturní čočky Obr. 8., které mají za úkol zvýšit efektivitu zachycování fotonů. Jako další alternativou zvýšení citlivosti snímačů se využívá princip průhledných elektrod či velmi tenkých vrstev, nebo je možno světlo přivádět přes zadní plošnou elektrodu snímače.



Obr. 8. Realizace čoček nad snímacím prvkem CCD.

V případě barevného snímání obrazu, je nutno zajistit oddělené zpracování barev obrazu, přičemž se používá RGB nebo CMY barevný model. Oddělené zpracování barev se realizuje:



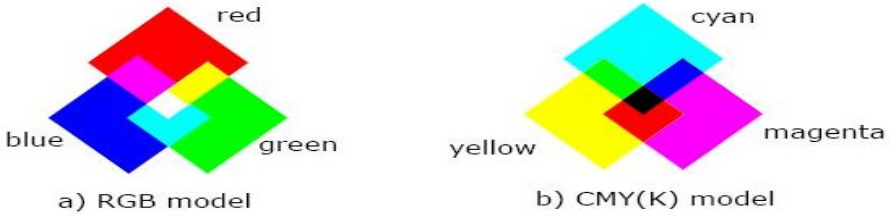
Obr. 9. Tříčipové uspořádání kamery pro barevné snímání scény

1. 3 samostatnými snímači - viz Obr. 9., kdy každý snímač snímá danou barvu odděleně (zpravidla RGB model), tzn. světlo přicházející z objektivu je nutno rozdělit na 3 samostatné barevné složky – toto řešení je dražší z důvodu počtu snímačů, ale může poskytovat vyšší citlivost i vyšší rozlišení,

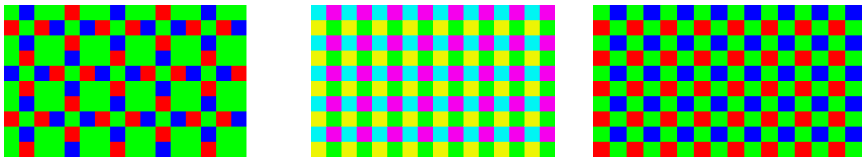
2. rastrovými filtry přímo realizovanými na ploše snímače - viz Obr. 12. vycházejících z barevných modelů RGB či CYK - viz Obr. 10., kdy výsledný pixel obrazu je tvořen několika samostatnými snímacími prvky, což však se promítá do složitosti výroby samotného snímače, ale i max. dosažitelného rozlišení i celkové citlivosti snímače.

3. jediným snímačem, ale s výměnnými barevnými filtry, které se vřazují do optické cesty mezi snímač a objektiv (toto řešení používají astronomické pozorovací přístroje), přístroj disponuje i vyšší citlivostí než dříve uvedené varianty.

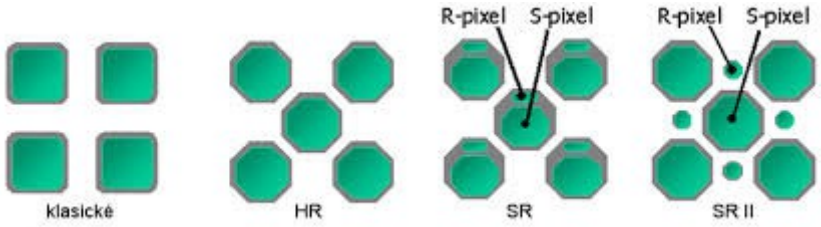
V případě použití rastrových filtrů pro vlastní barevnou interpretaci se zpravidla používá pro rekonstrukci barevné informace v daném bodě obrazu Bayerův filtr - viz Obr. 11. Tento filtr vychází z fyziologie lidského oka a jeho maximální citlivost na žluto-zelenou barvu. Tato vlastnost se pak promítá i do fyzické realizace rastrových filtrů tím, že jeden výsledný pixel obrazu je snímán dvěma snímači pro zelenou barvu – mód RGBG - viz Obr. 12. Z hlediska propustnosti filtrů a dosažení max. citlivosti snímače by bylo lépe použít model CMY, avšak výroba fialového filtru je extrémně složitá, proto výrobci používají smíšený model CMYG Obr. 12.



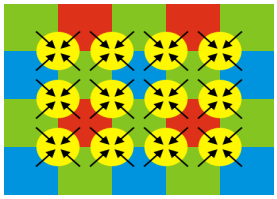
Obr. 10. Barevné modely RGB a CYK.



Obr. 12: Příklady barevných rastrů na CCD čipu pro barevné podání obrazu

V současnc  novat

Obr. 13: Uspořádání snímacích prvků na CCD

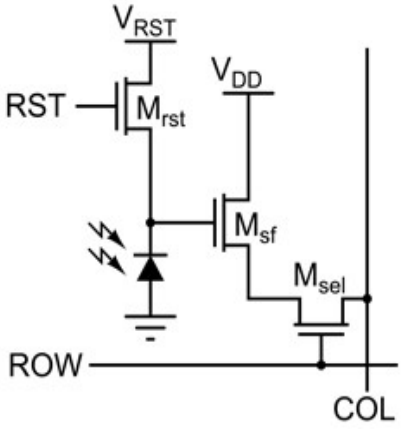


Obr. 11. Realizace Bayerova filtru

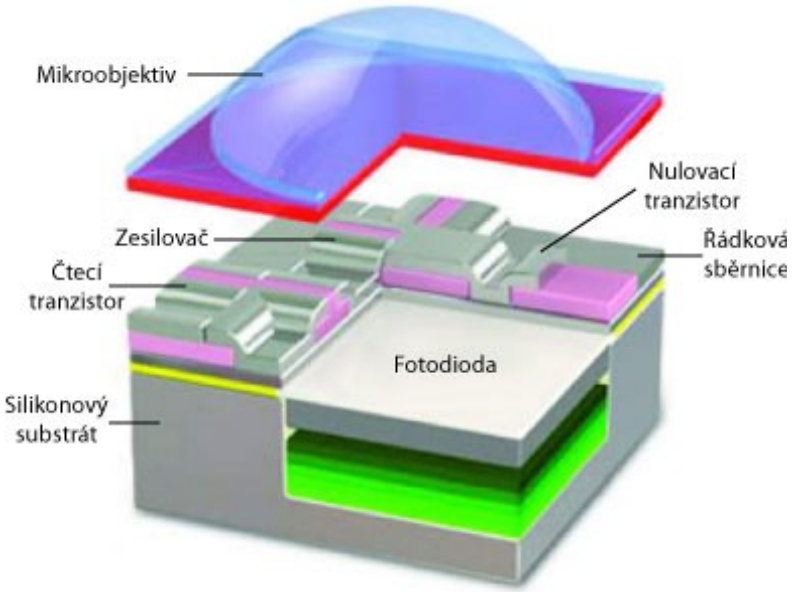
1.2 CMOS snímače

V CMOS snímačích však volné elektrony neshromažďujeme jako v CCD, ale jsou využity pro ovlivňování vodivosti PN přechodu - viz Chyba: zdroj odkazu nenalezen, na který dopadá světlo, měříme přímo tuto vodivost.

- CMOS snímač se skládá - viz Obr. 15:
- z fotocitlivé PN či PIN diody,
- z pomocného obvodu, který resetuje zbytkový náboj snímací diody,
- z adresovatelného čtecího obvodu, který může být pasivní či aktivní (v aktivním je použit předzesilovač a korektor šumu snímací diody).

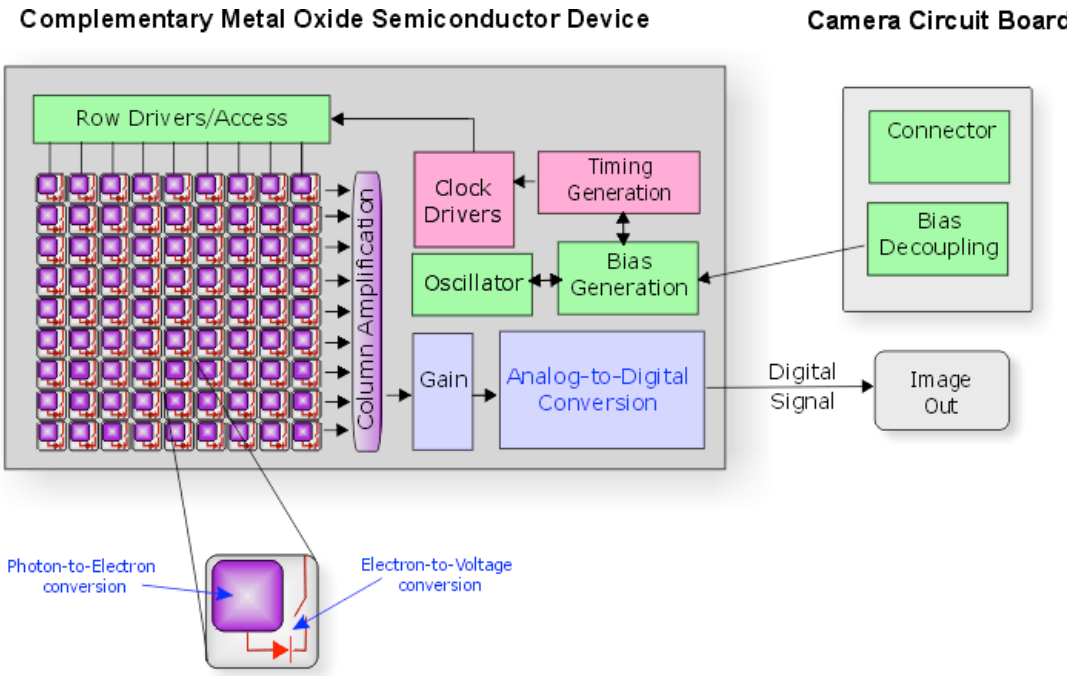


Obr. 14. Snímací prvek CMOS snímače



Obr. 15: Celkové uspořádání snímacího prvku CMOS snímače s čočkou

Celkové uspořádání snímače je pak zachyceno na Obr. 16.



Obr. 16. Celkové uspořádání CMOS snímače s elektronikou

Výhodou CMOS snímačů je oproti CCD:

- nižší výrobní náklady,
- jednodušší způsob řízení,
- adresovatelné čtení kterékoliv snímací diody,
- vyšší rychlost čtení snímků,
- možnost korekce šumu již v obvodu snímače,
- odolnost proti bloomingu.

Výhody CCD oproti CMOS:

- akumulace náboje, tzn. vyšší citlivost v případě velmi dlouhých časů expozice,
- lineární citlivost oproti logaritmické,
- nižší šum.

1.3 Praktické dopady snímacích prvků na zpracování signálů

Z hlediska reálné konstrukce obrazových snímačů je nutno si uvědomit, že při snímání reálného obrazu scény dochází ke ztrátě rozlišení:

- **v ploše** - je daná snímacím rastrem, velikostí snímacích prvků rastru a jejich počtem – rozlišením, dále poměrem aktivní části snímače k ostatním neaktivním plochám[4],
- **v barvě či úrovni** - je dána barevnou hloubkou tj. kvantizací úrovně dané barvy v daném obrazovém bodu, tzn. rozsahem hodnot, které můžeme obdržet,
- **v čase** - je dáno způsobem, jak je snímán pohyblivý obraz – prokládaně či neprokládaně, ale i počtem snímků za jednotku času.

[4] Plošná ztráta zobrazení je největším problémem v medicíně v případě CT, ultrazvukového vyšetření i jiných moderních přístrojů.

Dále dochází k:

- **Ztrátě barevného podání** – na ní se podílí konstrukce snímače, optická cesta mezi snímaným obrazem a snímačem, světelné podmínky při snímání obrazu. Rovněž záleží na typu zvoleného barevného modelu pro záznam obrazu a jeho výpočtu z hodnot získaných z reálného snímače. Mnoho výrobců ukládá ve svých zařízeních surový obraz přímo, tak jak byl získán ze snímače, pod označením RAW, pak pro zobrazení obrazu je nutno tato surová data přepočítat do barevného modelu zobrazovače.
- **Přidání šumu k vlastní obrazové informaci** – šum je vedlejším produktem každé elektronické součástky, neboť přítomnost volných nosičů nábojů v polovodičích i vodičích souvisí se stochastickým náhodným procesem jejich uvolňování, pohlcování ale i pohybu. Největším zdrojem je samotný termický šum, který lze eliminovat jen jediným způsobem, a to provozem součástek při snížených teplotách.
- **Rozmazání pohybujícího se obrazu** vlivem délky expozice[5].

Ke všem faktorům je pak nutno přihlížet při dalším zpracování obrazu tak, aby negativní vliv těchto faktorů byl maximálně potlačen.

[5] Při některých způsobech snímání lze snímací zařízení synchronizovat s pohybem obrazu, pak lze rozmazání obrazu částečně eliminovat. Existuje i matematický způsob zpětné korekce rozmazaného obrazu vlivem pohybu pomocí FT.

2 Zpracování obrazového signálu

RYCHLÝ NÁHLED KAPITOLY

V této kapitole je ozřejmeny všechny způsoby digitálního zpracování obrazu.

CÍLE KAPITOLY

Po prostudování této kapitoly porozumíme:

- geometrickým transformacím,
- bodovým filtracím,
- prostorovým filtracím.

Dále jsou v této kapitole probrány geometrické transformace, barevné modely a interpolace.

KLÍČOVÁ SLOVA KAPITOLY

bodová filtrace, prostorová filtrace, konvoluce, FT, WT, interpolace

Zpracování digitálního obrazu můžeme rozdělit na:

- Předzpracování,
- Detekce objektů nebo příznaků v obraze,
- Klasifikace objektů či příznaků a reprezentace výsledků.

Tento učební text se nezabývá komprimací a dekomprimací obrazu, neboť v případě rozpoznávání či jiném finálním užití dat obrazu je bezpředmětné pracovat během výpočtu s komprimovaným obrazem. Navíc ztrátové komprimace např. JPEG vedou ke ztrátě informací uvnitř obrazu a mohou způsobit nefunkčnost celého systému.

2.1 Předzpracování

Zahrnuje:

- **geometrické transformace** - odstranění chyb snímačů a optických soustav,
- **bodové filtrace** – které jsou představovány úpravami kontrastu, jasu, gama korekce, odstranění vinětace, ekvalizace (logaritmická či jiná) a prahování,
- **prostorové filtrace** – které jsou realizovány pomocí konvolučních masek (lze zajistit filtraci šumu, vyhlazování obrazu, detekci hran) nebo pomocí nelineárních, rekurzivních, statistických filtrů, filtrů využívajících FT a WT či jiných matematických nástrojů, např. Radonova či Houghova transformace či morfologických operací,
- **prostorová transformace** - přechod mezi různými matematickými modely scény, např. přechod z 2D do 3D při stereoskopickém snímání scény, nebo pořízením snímků z jiných míst při známém souřadnicovém umístění kamery, či doplnění informací o vzdálenosti jednotlivých objektů k snímanému obrazu (systémy označované jako LIDAR).

2.2 Předzpracování obrazu

2.2.1 Geometrické úpravy obrazu

Z hlediska geometrických úprav, se jedná o:

- posun,
- zvětšení/zmenšení,
- rotaci,
- zkosení,
- distorzi – válcovou, soudečkovou, kulovou, panoramatickou
- náklon.

Všechny operace jsou snadno implementovatelné, kdy pro každou hodnotu jasu v souřadnicovém sytému původního obrazu vypočteme pomocí transformačních funkcí novou hodnotu souřadnice v novém obraze. Transformace mohou být buď lineární, tzn. geometrické

prvky v originálním a změněném obraze nemění svůj tvar, tzn. přímka zůstává přímkou, nebo nelineární (často se takováto transformace označuje jako warping či morphing). Některé nelineární transformace např. nemusí být realizovány přímo, ale mohou být složeny z lineárních či jednodušších nelineárních, kdy obraz je rozdělen na více segmentů a transformace probíhá v každém segmentu samostatně.

Obecně transformaci zapíšeme:

$$x' = f(x, y)$$
$$y' = g(x, y)$$

kde: x, y – původní souřadnice,
x', y' – nová hodnota souřadnic,
f, g – obecná transformační funkce, může být definována pro obě souřadnice, či jen jednu z nich, příklad viz tabulka níže.

Některé z lineárních transformací jsou uvedeny v tabulce níže, výhodou těchto transformací je, jdou-li po sobě, mohou být realizovány v jediném kroku, často se k tomu používá maticový tvar výpočtu a systém homogenních souřadnic, přičemž se určí hodnoty jednotlivých koeficientů transformační matice tím, že transformační matice prostě vynásobíme[6].

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} z_{11}{}' & z_{12} & t_x \\ z_{21} & z_{22} & t_y \\ 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

[6] Pozor, ale neplatí zde princip komutativní, tzn. závisí na pořadí

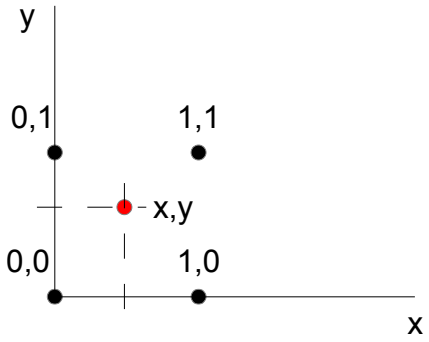
Transformace	**Výpočet**
translace	$x' = a + x \quad y' = b + y$ a,b - koeficienty posunu
zoom	$x' = a \cdot x \quad y' = b \cdot y$ a,b - koeficienty zvětšení >1, zmenšeni <1
rotace okolo počátku[7]	$x' = x\cos(\varphi) - y\sin(\varphi)$ $y' = x\sin(\varphi) + y\cos(\varphi)$ φ - úhel natočení
zrcadlení	$V: x' = w - x, y' = y$ $H: x' = x, y' = h - y$ w, h - šířka a výška obrázku $D: x' = y, y' = x$
zkosení	$V: x' = a + x + by, y' = y$ $H: x' = x, y' = a + y + bx$ a,b – koeficienty zkosení

Při nelineárních transformacích nová souřadnice může být např. vypočtena pomocí polynomu n-tého stupně, kde polynomem aproximujeme požadovaný výsledek:

$$x' = x + \sum_{r=0}^{m}\sum_{k=0}^{m} a_{rk} x^r y^k$$
$$y' = y + \sum_{r=0}^{m}\sum_{k=0}^{m} b_{rk} x^r y^k$$

2.2.2 Interpolace

Poněvadž obraz je zpravidla uložen v čtvercové pravoúhlé síti[8], takže jednotlivé body leží vždy na celočíselných násobcích souřadnic (viz Obr. 17. černě označené body se souřadnicemi [0,0], [0,1], [1,0], [1,1]). Při transformacích je třeba řešit, jaký jas má pro danou barevnou složku bod nabýt, nemá-li nová souřadnice hodnotu celého čísla (viz Obr. 17. červeně označený bod se souřadnicemi [x,y]). Proto při určení jasu přistupujeme k interpolaci, která má zohlednit umístění bodu [x,y] mezi body ležící na celočíselných souřadnicích v pravoúhlé síti. Níže jsou uvedeny některé z metod, jak přepočítat jas daného bodu (pro každou z barevných složek samostatně)[9]:



Obr. 17. Interpretace bodů v interpolaci

[7] Požadujeme-li rotaci okolo jiného bodu než počátku, provedeme nejdříve translaci, pak rotaci v počátku, a zpět translaci vůči bodu rotace.

[8] Existují i hexagonální interpretace a ukládání obrazových informací.

[9] Postup je obrácený, v novém obraze zjišťujeme pro daný bod x', y', kde bude ležet ve starém obraze a ze znalosti hodnot jasu původního obrazu odvodíme hodnotu v transformovaném obrazu.

Nejbližší soused

$$f_{x,y} = f_{round(x),roudn(y)}$$

kde: round je zaokrouhlovací funkce.

Bilineární interpolace:

V této interpolaci vycházíme z hodnot 4 okolních bodů, viz Obr. 17. Základní tvar rovnice:

$$f_{x,y} = (1-x)(1-y) f_{0,0} + x(1-y) f_{1,0} + y(1-x) f_{0,1} + x\,y\,f_{1,1}$$

kde: x, y - jsou souřadnice bodu uvnitř mřížky x, y = <0, 1>

$f_{i,j}$ - jsou hodnoty jasu 4 bodů tvořící rohy mřížky

$f_{x,y}$ - je vypočítaná hodnota jasu.

Nebo taky pomocí maticového vyjádření:

$$f_{x,y} = \begin{vmatrix} 1-x & x \end{vmatrix} \begin{vmatrix} f_{0,0} & f_{0,1} \\ f_{1,0} & f_{1,1} \end{vmatrix} \begin{vmatrix} 1-y \\ y \end{vmatrix}$$

Souřadnice bodu 0,0 také získáme odstraněním desetinné části čísla, x, y je pak desetinná část čísla.

Bikubická interpolace:

V této interpolaci vycházíme z hodnot jasu 16 okolních bodů v rastru 4x4. Základní rovnice:

$$f_{x,y} = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{i,j}\, x^i\, y^j$$

kde $a_{i,j}$ jsou konstanty vypočítané na základě interpolace.

Výpočet lze realizovat i na základě maticové operace:

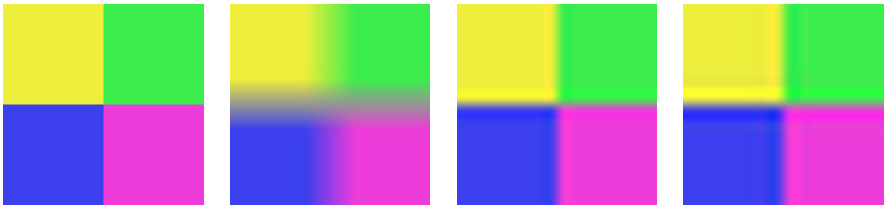
$$f_{x,y} = \begin{vmatrix} 1 & y & y^2 & y^3 \end{vmatrix} A \begin{vmatrix} f_{-1,-1} & f_{0,-1} & f_{1,-1} & f_{2,-1} \\ f_{-1,0} & f_{0,0} & f_{1,0} & f_{2,0} \\ f_{-1,1} & f_{0,1} & f_{1,1} & f_{2,1} \\ f_{-1,2} & f_{0,2} & f_{1,2} & f_{2,2} \end{vmatrix} A^T \begin{vmatrix} 1 \\ x \\ x^2 \\ x^3 \end{vmatrix} \quad A = \frac{1}{6} \begin{bmatrix} 0 & 6 & 0 & 0 \\ -2 & -3 & 6 & -1 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

V odborné literatuře najdeme i bikubickou interpolaci definovanou takto:

$$f_{x,y} = \begin{vmatrix} g_{1x} & g_{2x} & g_{3x} & g_{4x} \end{vmatrix} \begin{vmatrix} f_{0,0} & f_{0,1} & f^{y}_{0,0} & f^{y}_{0,1} \\ f_{1,0} & f_{1,1} & f^{y}_{1,0} & f^{y}_{1,1} \\ f^{x}_{0,0} & f^{x}_{0,1} & 0 & 0 \\ f^{x}_{1,0} & f^{x}_{1,1} & 0 & 0 \end{vmatrix} \begin{vmatrix} g_{1y} \\ g_{2y} \\ g_{3y} \\ g_{4y} \end{vmatrix}$$

$$g_{1q} = 1 - 3q^2 + 2q^3 \qquad g_{2q} = 3q^2 - 2q^3 \qquad g_{3q} = q - 2q^2 + q^3 \qquad g_{4q} = -q^2 + q^3 \qquad q = x\,;\,y$$

$$f^{x}_{p,q} = \frac{f_{p+1,q} - f_{p-1,q}}{2} \qquad f^{y}_{p,q} = \frac{f_{p,q+1} - f_{p,q-1}}{2}$$



Obr. 18. Příklady interpolace
(Gimp - nearest, linear, cubic, sinc, původní obraz 16x16 px se 4 barevnými čtverci zvětšený na 100x100 px)

2.3 Bodové filtry

Tyto filtry nevyužívají hodnot vedlejších obrazových bodů, ale jen hodnotu daného bodu obrazu. Poněvadž obraz může nést barevnou informaci nebo jen černo/bílou v různé škále šedi, mohou bodové filtrace pracovat jak s jednotlivými barevnými složkami, tak s celkem, přičemž je nutné brát na zřetel, jak se pohlíží na barevný obraz jako na celek.

Dále je nutné mít na zřeteli, že barevný obraz může být uložen v různém barevném modelu, přičemž v každém barevném modelu může konkrétní úprava obrazu poskytovat lepších výsledků než v jiném barevném modelu.

2.3.1 Barevné modely

Informace o barvě jednoho bodu v digitálním obraze je uložena zpravidla v některém z těchto barevných modelů[10]:

- **RGB** či **RGBA** – (Red Green Blue, Alpha) - model používá aditivní míchání barev a je přirozený např. pro zobrazovací systémy CRT, LED, LCD, ale i mnoho snímacích zařízení, např. kamery, fotoaparáty, skenery[11]. V mnoha programových nástrojích je tento model rozšířen o kanál Alpha, což je kanál udávající průhlednost.
- **CMY** či **CMYK** – (Cyan Magenta Ylow, Black) - model používá subtraktivní míchání barev, tento model je přirozený pro tiskárny, CMYK model je jeden z modelů, ve kterém nelze konkrétní barvu z jiného modelu stanovit jednoznačně,
- **HSV** či **HSB** (Hue Saturation Value, Balance) – model se blíží lidskému vnímání barev,
- **HSL** (Hue Saturation Lightness) – podobný HSV, ale má odstraněny některé nedostatky HSV,
- **YUV** (Luminance Chroma) – model používaný v přenosových TV systémech, např. PAL, vycházející z citlivosti oka. Model vyniká tím, že hlavní složka Y nese převážnou část obrazové informace, zato složky UV jsou v poměru 4:1:1 nebo 4:2:2 vůči složce Y, tzn. nesou menší část informace vůči jasové složce. Vlastnost tohoto modelu je použita v komprimovaném standartu JPEG, MPEG,
- **LAB** – matematický model, ve kterém se na ose z vynáší jas, osa nese označení jako "L" (bílá, šedá, černá), na x ose - označení "a" se pracuje s barvou zelenou -a červenou +a, na y ose - označení "b" se pracuje s barvou žlutou +b a modrou -b. Tento model se rovněž přibližuje citlivosti lidského oka.

[10] Abstraktní matematický model, který se snaží číselně popsat barvu v digitálním obraze, z hlediska fyzikálního jsme sice schopni barevný prostor změřit absolutně, a to ve fyzikálních jednotkách, ale prakticky jde jen o jisté přiblížení. Barva může mít jen konkrétní jednu vlnovou délku, nebo může být složena z více vlnových délek, nebo může být i směsicí spojitých spekter. Paradoxně v mnohých modelech i při max. počtu bitů, které máme k dispozici, nejsme schopni uložit barevnou informaci, kterou by bylo zdravé oko schopno rozlišit, přesto, že oko rozlišuje cca jen 10 mil. barev.

[11] Viz snímací prvky, používá se modifikovaný model sRGB.

Mezi všemi modely existuje matematický přepis[12], který popisuje převod mezi barevnými modely pro normalizované hodnoty, aby v každém modelu byla konkrétní barva subjektivně[13] vnímána totožně jak na zobrazovacím[14] , tak na snímacím zařízení. Touto problematikou se samostatně zabývá barevná teorie, která zavádí řadu pojmů, např. barevný prostor, barevný režim, teplota barev apod. Nutno si však uvědomit, že převod mezi modely si vždy vyžádá jistou barevnou ztrátu, neboť se často zaokrouhluje, a to finálně na celá čísla.

Jak již bylo uvedeno, v konstrukci snímačů se můžeme setkat s tím, že obraz nemusí odpovídat žádnému z uvedených modelů, ale je uložen v surovém RAW tvaru tak, jak byl získán přímo ze snímače. Pak je nutno disponovat přepočítacími koeficienty potřebnými pro přepočet do daného modelu a soustavou přepočtových rovnic, které zpravidla uvádí přímo výrobce snímacího prvku, nebo jsou rovnice stanoveny experimentálně.

Samotnou otázkou je, jak v daném modelu ukládat hodnotu dané barevné složky, lze použít procentuální vyjádření 0 až 100% nebo vyjádření v poměru 0 až 1 (používá se pak reálné číslo), ale toto procentuální či číselné vyjádření 0 až 1 nic neříká o barevné hloubce – barevném režimu či gamutu. Neboli jinak - kolik bitů je použito pro úschovu barevné informace v dané složce daného modelu, a může to být od 4 až 32 bitů na jednu složku či 16 až 96 bitů na všechny složky dohromady, přičemž od počtu bitů se odvíjí, kolik barev je schopen model pojmout, např. běžně pro RGB model se používá 24 bitová barevná hloubka, tj. 8 bitů na každou barevnou složku, kdy každá ze složek může nabýt hodnot 0-255, tzn. systém může vytvořit celkem 2^{24} různých barev.

V případě digitálních úprav obrazů pro další rozpoznávání či jiné užití však nemusí být žádoucí dodržovat standardizaci v modelech, ale bývají často používány převodní nástroje tak, aby bylo možné v obraze lépe segmentovat jeho jednotlivé části, tzn. např. před vlastním prahováním je vytvořen jen černobílý obraz ve stupních šedi, přičemž můžeme některou z barevných složek zcela potlačit nebo rozlišit v obraze segmenty, které by obdržely stejnou úroveň jasu v daném modelu. Přepočet do škály šedi - viz jednoduchý vzorec níže. Výpočet

[12] Z důvodů počtu modelů nejsou zde uvedeny matematické přepisy pro přepočet mezi modely, lze je snadno vyhledat v Internetu.

[13] Subjektivní vnímání zde zohledňuje citlivost lidského oka na jednotlivé vlnové délky světla.

[14] Zobrazovací zařízení nemusí být jen monitor, ale i tiskárna. Snímací zařízení nemusí být jen fotoaparát, ale i videokamera a skener.

může zahrnovat celou škálu převodních funkcí f využívající např.: log, sin, tang, vč. aplikace fuzzy logiky.

$$g = F(f_1(RGB), f_2(RGB), f_3(RGB))$$

Kde: **F , $f_{1,2,3}$** - jsou obecné funkce jedné až 3 proměnných,

g - hodnota šedi[15],

RGB - hodnota složek RGB modelu.

2.3.2 Úpravy barevného podání

Pro úpravy barvených složek modelu obrazu lze využít obecnou rovnici:

$$f'_{x,y} = g\,(f_{x,y} + \varepsilon)^{\lambda}$$

kde: **$f'_{x,y}$** - nová hodnota složky

$f_{x,y}$ - původní hodnota složky

ε - posun

g - obecná lineární či nelineární funkce, bývá odvozena z histogramu

λ - exponent.

Úpravu můžeme realizovat na jednotlivé složky modelu či na celý model, nebo v případě, že model disponuje výpočtem jasu, tak jen na jasovou složku.

Např. inverze složky se realizuje (pro normalizovanou hodnotu 0 až 1):

$$f'_{x,y} = 1 - f_{x,y}$$

Pro úpravu složky se často používá Gama korekce[16], která je dána výrazem:

$$f'_{x,y} = \lambda f^{\gamma}_{x,y}$$

kde: **γ** a **λ** - jsou kladné reálné konstanty, průběh křivky viz Obr. 27 [17].

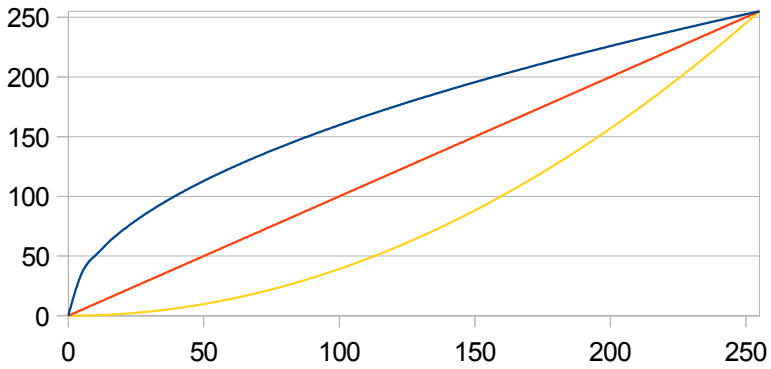
Gama korekce avšak sama nemění hodnotu nejsvětlejší a nejtmavší části.

[15] Nutno brát v potaz, s kolika bity pracujeme a zda používáme normalizovaných hodnot složek a počítáme normalizovanou hodnotu šedi, a to z důvodů podtečení či přetečení hodnoty.

[16] Korekce souvisela s úpravou svítivosti pixelů na CRT monitorech a jejich principem fungování, přičemž svítivost závisí exponenciálně na napětí mřížky, i když CRT monitory ustupují, korekce se stále zachovává na straně kamer.

[17] Normalizací se myslí, že hodnota λ je stanovena tak, aby výsledná hodnota nabyla vždy max. rozsahu.

Pro stanovení průběhu funkce g se s výhodou využívá statistické zpracování obrazu v podobě histogramu, tj. podíl počtu obrazových bodů s danou složkou v obraze k celkovému počtu obrazových bodů, kdy můžeme buď histogram vyrovnávat – tj. zajistit rovnoměrné rozložení, nebo zpracovávat jen část obrazové informace, tzn. g funkce může být jakákoliv parametrická funkce či polynom n-tého stupně a existují matematické nástroje, jak určit koeficienty polynomu z histogramu za účelem jeho vyrovnání.



Obr. 19. Příklad Gama korekce pro hodnoty γ: 0,5 - modrá; 1,0 - červená; 2,0 - žlutá a normalizovaná

2.3.3 Prahování

Prahování převádí obraz zpravidla jen na binární - viz výraz níže - či omezený počet jasových hodnot. V tomto případě na základě např. histogramu či jiným parametrickým nástrojem se určí hodnota prahu/ů p, q[18]. Prahovat lze i na libovolnou barevnou hloubku obrazu, pak je nutno stanovit odpovídající počet prahů.

$$f'_{x,y} = \begin{cases} 1 & když\ f_{x,y} < p \\ 0 & jinak \end{cases}$$

$$f'_{x,y} = \begin{cases} 1 & když\ f_{x,y} > p \wedge f_{x,y} < q \\ 0 & jinak \end{cases}$$

2.4 Prostorové filtry

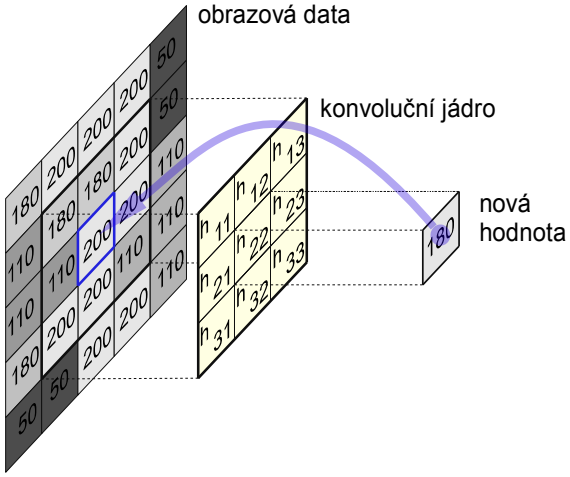
Tyto filtry využívají pro svoji funkci hodnoty z daného prostoru obrazu, resp. pracují s více jak jedním bodem současně. Níže jsou uvedeny všechny základní filtrace a transformace, které je možno běžně použít pro úpravu obrazové informace v obraze. Mezi filtry, které lze velice snadno realizovat patří filtry využívající principu konvoluce - viz konvoluční filtry. Do

[18] Prahovat lze jen rozdělením intervalu, máme jen jeden práh, nebo pásmově, pak určujeme hranici spodního a horního prahu.

prostorových filtrů však patří i všechny ostatní filtry, které jsou uvedeny až do konce této kapitoly ale je jim věnována větší pozornost.

2.4.1 Konvoluční filtry

Konvoluce je maska, která se přiloží na obraz ve všech místech původního obrazu a s její pomocí se vypočte nová hodnota obrazového bodu zpravidla ve středu masky. Jádro konvoluce je v mnoha případech matice veliká jen 3x3 nebo 5x5. Pomocí konvoluce lze definovat velké množství operací, např. filtrace šumu, derivace, zvýraznění a detekce hran. V základních programech pro úpravu obrázků je převážná většina filtrů právě definována pomocí konvoluce. Konvoluci, respektive její jádro, lze odvodit z Fourierovy transformace pro diskrétní podobu signálu a požadovaný průběh, pro obrázky má konvoluce tento matematický tvar:



Obr. 20. Ukázka realizace konvoluce

$$f * h_{x,y} = \varepsilon + \frac{1}{n} \sum_{i=-k}^{k} \sum_{j=-k}^{k} h_{i,j} \, f_{x-i,y-j}$$

Kde: f * h - je výsledek konvoluce

h(i, j) - je jádro konvoluce

ε - je posun

n - jasová korekce

k - udává velikost jádra konvoluční matice, k ≥ 2, zpravidla 2n-1., tj: 3, 5, 7

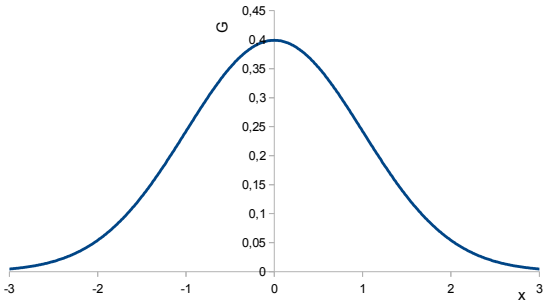
Při vlastní tvorbě jádra konvoluce je vhodné mít na zřeteli, že součet hodnot jádra konvoluce, aniž by konvoluce ovlivnila celkový jas obrazu, musí být 1, v opačném případě je ovlivněn celkový jas obrazu.

Výhodou konvoluce je, že v případě, chceme-li nad obrazem provádět několik operací pomocí konvoluce postupně, můžeme všechny tyto operace provést během jediné konvoluční transformace, kdy všechna jednotlivá konvoluční jádra sečteme.

2.4.2 Odstranění šumu

Lze použít čtvercový nebo kruhový filtr:

$$\frac{1}{9}\begin{vmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{vmatrix} \qquad \frac{1}{5}\begin{vmatrix}0 & 1 & 0\\ 1 & 1 & 1\\ 0 & 1 & 0\end{vmatrix}$$



Obr. 21. Průběh Gaussovského rozložení

V případě, že hodnoty koeficientů jádra konvoluční matice budou odpovídat Gaussovskému rozložení pravděpodobností, hovoříme o Gaussovském filtru:

$$G_x = \frac{1}{\sqrt{2\pi}\,\delta}\cdot e^{-\frac{x^2}{2\delta^2}}$$

$$G_{x,y} = \frac{1}{\sqrt{2\pi}\,\delta^2}\cdot e^{-\frac{x^2+y^2}{2\delta^2}}$$

zde je např. jádro konvoluce:

$$\begin{matrix}0 & -1 & 0\\ -1 & 5 & -1\\ 0 & -1 & 0\end{matrix} \text{ nebo } \frac{1}{16}\begin{vmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{vmatrix} \text{ nebo } \frac{1}{159}\begin{vmatrix}2 & 4 & 5 & 4 & 2\\ 4 & 9 & 12 & 9 & 4\\ 5 & 12 & 15 & 12 & 5\\ 4 & 9 & 12 & 9 & 4\\ 2 & 4 & 5 & 4 & 2\end{vmatrix}$$

Obdobně hodnoty koeficientů mohou kopírovat průběh Laplaceovy funkce:

$$\begin{vmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{vmatrix} \text{ nebo } \begin{vmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{vmatrix} \text{ nebo } \begin{vmatrix}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{vmatrix} \text{ nebo } \begin{vmatrix}-1 & 0 & -1\\ 0 & 4 & 0\\ -1 & 0 & -1\end{vmatrix}$$

$$\begin{vmatrix}0 & 1 & 1 & 2 & 2 & 2 & 1 & 1 & 0\\ 1 & 2 & 4 & 5 & 5 & 5 & 4 & 2 & 1\\ 1 & 4 & 5 & 3 & 0 & 3 & 5 & 4 & 1\\ 2 & 5 & 3 & -12 & -24 & -12 & 3 & 5 & 2\\ 2 & 4 & 0 & -24 & -40 & -24 & 0 & 5 & 2\\ 2 & 5 & 3 & -12 & -24 & -12 & 3 & 5 & 2\\ 1 & 4 & 5 & 3 & 0 & 3 & 5 & 4 & 1\\ 1 & 2 & 4 & 5 & 5 & 5 & 4 & 2 & 1\\ 0 & 1 & 1 & 2 & 2 & 2 & 1 & 1 & 0\end{vmatrix}$$

2.4.3 Filtry pro detekci hran

Obecně lze stanovit, že hrana se vyskytuje v místě změny jasu, proto pro detekci hrany se používá první nebo druhá derivace změny jasu v obraze. Hrana se pak vyskytuje v místě, kde první derivace dosahuje nenulové hodnoty nebo druhá derivace mění znaménko.

Výpočet derivací lze snadno realizovat, avšak v případě reálných obrázků je tento jednoduchý model často zkomplikován tím, že hrany jsou zašuměny a rozmazány a k tomu se přidává fakt, že prostorová interpretace obrazu je omezena na pravoúhlou síť.

V rámci zjednodušení jsou z první derivace odvozeny různé konvoluční operátory, viz dále.

Základní Robertsův operátor (otočen o 90°):

$$\begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} \quad \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix}$$

Sobelovy operátory pro detekci hran jsou odvozeny z 1 derivace (lze je i otočit o 45°):

$$\begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix} \quad \begin{vmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{vmatrix} \quad \begin{vmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{vmatrix}$$

Dalším je operátor Prewittové (nutno použít obě jádra současně otočené o 90°). Ve variantě 5 x 5, doplníme operátor o hodnotu -2/+2:

$$\begin{vmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix} \quad \begin{vmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{vmatrix}$$

Kirschův operátor (základní tvar, nutno ho rotovat o 45°):

$$\begin{vmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{vmatrix}$$

Robinsonův operátor (opět nutno rotovat):

$$\begin{vmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{vmatrix}$$

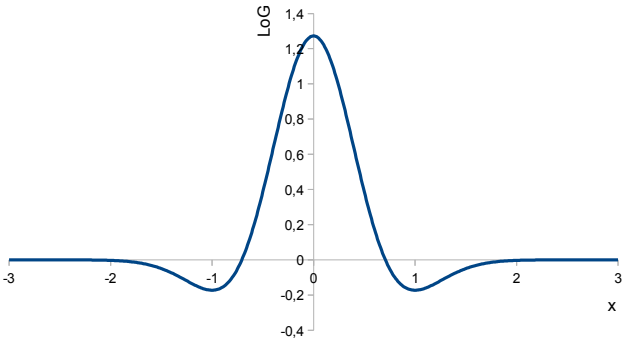
Laplaceovy operátory pro detekci hran z druhé derivace:

$$\begin{vmatrix} -1 & 2 & -1 \\ 2 & -8 & 2 \\ -1 & 2 & -1 \end{vmatrix} \quad \begin{vmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad \begin{vmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{vmatrix} \quad \begin{vmatrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{vmatrix} \quad \begin{vmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{vmatrix}$$

Operátory lze s výhodou použít pro detekci vyšších frekvencí v obrazu a v případě pořízení násobných obrazů téhož objektu z téhož místa, ale s různými ohniskovými vzdálenostmi[19], lze pomocí nich obraz analyzovat. Získáme tak masku, ze které zjistíme, které části obrazu nesou informace o detailech, a sestavíme jediný obraz pomocí masek (maska určí, z kterého obrazu se přenese informace o pixelu ve skládaném obrazu[20]), ve kterém budou umístěny detaily z každého jednotlivého obrazu[21].

Laplaceův operátor bývá často spojován s Gaussovým operátorem, abychom odstranili vliv šumu, pak operátor se nazývá LoG nebo GoL.

$$LoG_{x,y} = -\frac{1}{\pi \delta^4}\left(1 - \frac{x^2+y^2}{2\delta^2}\right) e^{-\frac{x^2+y^2}{2\delta^2}}$$



Obr. 22. Příklad průběhu LoG

Konvoluční matice:

$$\begin{vmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{vmatrix}$$

Nevýhodou všech výše uvedených operátorů je vysoká citlivost na šum a citlivost na směr hrany, tzn. operátory je nutno často aplikovat pro vertikální a horizontální směr nebo dokonce otáčet o 45° a operátory aplikovat 4×.

Kromě detektorů hran existují i detektory rohů, které se používají pro identifikaci řídících bodů např. při zpětné stereoprojekci.

---

[19] Každý obraz pak ve své konkrétní části nese jen část hloubky obrazu, tzn. z důvodů ohniskové vzdálenosti a ostření jsou části obrazu ostré a jiné rozostřené, např. budou ostré objekty 0 - 15m, ostatní neostré, nebo neostré 0 -15, ostré pak 15 až ∞.

[20] Můžeme provést buď přímé vkládání z každého obrazu nebo vážený průměr, kdy maska nám poskytne váhové koeficienty, ty však musí být normalizovány.

[21] Některé komerční fotografické přístroje mají tento systém již přímo vestavěn.

2.4.4 Cannyho detektor hran

Jedná se o vícekrokový algoritmus skládající se z odstranění šumu, hledání změn gradientů jasu pro každý obrazový bod, výpočet derivace ve směru největšího gradientu pomocí vhodného konvolučního operátoru. Po nalezení maxim derivací provedeme prahování s hysterezí.

2.4.5 Zostření obrazu

Opět existují konvoluční matice, např.:

$$\frac{1}{2}\begin{vmatrix} 0 & -1 & 0 \\ -1 & 6 & -1 \\ 0 & -1 & 0 \end{vmatrix} \quad \text{nebo} \quad \begin{vmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{vmatrix}$$

2.4.6 Reliéf

Znovu jde o konvoluční matici, kterou můžeme nechat rotovat o 45°:

$$\begin{vmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{vmatrix}$$

2.4.7 Hledání přímky

Opět jde o konvoluční matici, kterou můžeme nechat rotovat o 45°:

$$\begin{vmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{vmatrix}$$

2.5 Nelineární filtry

Nelineární filtry rovněž diskrétně zpracovávají jednotlivé body obrazu velice podobným způsobem jako konvoluční filtry, avšak využívají různých nelineárních funkcí popsaných matematickými formulemi.

Např. pro odstranění šumu typu sůl/pepř lze použít takto definovaný matematický předpis:

$$\begin{aligned} (a \geq b+2) \vee (a > b \wedge b \leq c) \vee (c > b \wedge b \leq a) \vee (c \geq b+2) &\rightarrow b = b+1 \\ (a \leq b-2) \vee (a < b \wedge b \geq c) \vee (c < b \wedge b \geq a) \vee (c \leq b+2) &\rightarrow b = b-1 \end{aligned},$$

kde a, b, c jsou po sobě jdoucí body v obraze, přičemž formule se aplikuje pro všechny 4 základní směry: S-J, V-Z, SV-ZJ, SZ-JV (S-sever, J-jih, V-východ, Z-západ – mapová orientace).

Mezi nelineární filtry můžeme i zařadit hledání hran pomocí textury[22].

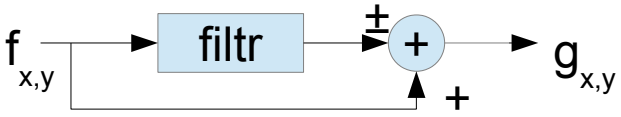
2.6 Rekurzivní filtry

Jedná se o filtry působící v prostorové nebo frekvenční oblasti, kdy výslednou odezvu filtru v daném obrazovém bodě nelze stanovit přímo výpočtem, ale je nutno výpočet realizovat rekurzivně. Způsob stanovení filtru v mnoha případech vychází z matematické analýzy využívající Fourierovu transformaci požadovaného filtru a požadované úpravy obrazu filtrem. Samotný rekurzivní filtr nezaručuje možnost realizace jakéhokoliv filtru, tzn. není východiskem pro všechny typy filtrace, ale může být časově méně náročný než jiný typ filtrace. Při nevhodně zvolených parametrech však filtry nefungují a nekonvergují k hledanému řešení. Základem rekurzivního filtru je následující vztah:

$$g_i = a * f + c * g_{i-1}$$

kde: * - představuje konvoluci
a, c - jsou jádra konvoluce, musí splňovat jistá kritéria pro konvergenci
f - hodnoty obrazu v určitém okolí
g - aktuální stav výpočtu v i-té iteraci, kdy na začátku výpočtu je zpravidla $g_0=0$

Často se setkáváme s použitím rekurzivního filtru při zpracování pohyblivé scény, pak filtr pracuje na základě funkčního diagramu na Obr. 24. Jako subfiltr může být použito konvoluce (nejčastější případ) nebo statistický filtr. V rámci požadované finální operace může být žádoucí složky odečíst či sečíst.



Obr. 23. Filtr využívající subfiltraci a původní obrazovou hodnotu

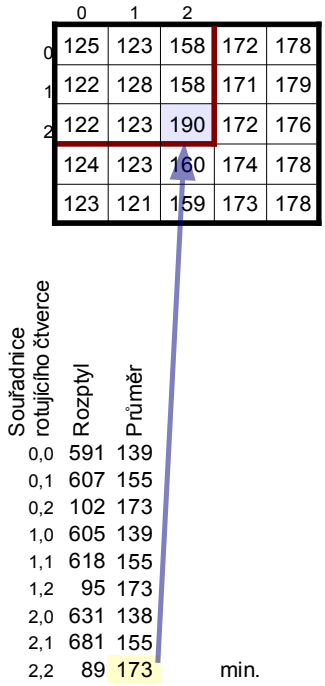
[22] Této problematice se budu věnovat později, časem tento text o tyto informace doplním.

2.7 Statistické filtry

Sem patří filtr využívající zvaný median, conservative s moothing a filtr s rotující maskou.

Filtr s rotující maskou pracuje následovně:

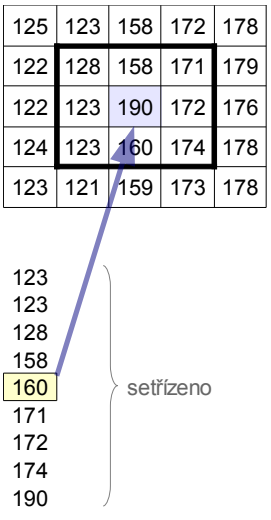
1. Pro každý nově počítaný bod v obraze vybereme kolem tohoto bodu oblast např. 5x5 obrazových bodů, nahrazovaný bod bude středem této oblasti.
2. V této oblasti necháme postupně rotovat masku 3x3 bodů a v každé této oblasti vypočteme velikost rozptylu jasu a střední hodnotu.
3. Hodnotu obrazového bodu ve středu výbrané oblasti 5x5 pak nastavíme na střední hodnotu jasu té rotující masky, ve které je rozptyl jasů minimální.



Obr. 24. Rotující maska

Filtr Median pracuje následovně - viz Obr. 25.:

1. Pro každý nově počítaný bod v obraze vybereme kolem tohoto bodu oblast např. 3x3[23] obrazových bodů, nahrazovaný bod bude středem této oblasti.
2. V této oblasti setřídíme hodnoty jasu.
3. Hodnotu obrazového bodu ve středu výběrové oblasti 3x3 pak nastavíme na hodnotu jasu nacházející se uprostřed setříděné posloupnosti, tj. na 5 pozici.

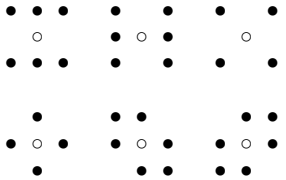


Obr. 25. Median

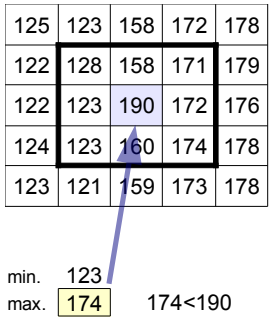
[23] Oblasti mohou být i větší, např. 5x5.

Filtr Conservative Smooth pracuje následovně (filtr vhodný pro filtraci rušení typu pepř a sůl), viz Obr. 17.:

1. Pro každý nově počítaný bod v obraze vybereme z obrázku oblast např. 3x3 obrazových bodů, nebo body vybereme podle předem daného morfologického operátoru - viz Obr. 28., přičemž nahrazovaný bod bude středem této oblasti.
2. V této oblasti nalezneme min. a max. hodnotu jasu.
3. V případě, že hodnota obrazového bodu uprostřed oblasti má větší jas než nalezené maximum, nastavíme tuto hodnotu na nalezené maximum, v případě, že má hodnota obrazového bodu hodnotu menší než nalezené minimum, nastavíme jí toto minimum.



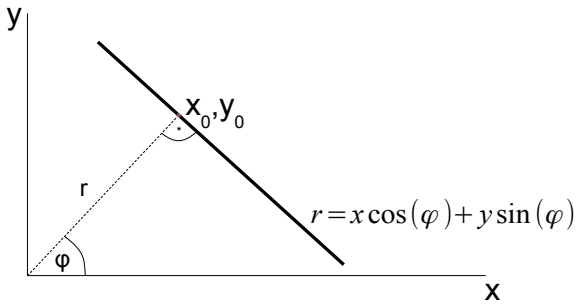
Obr. 26. Příklad morfologických operátorů



Obr. 27. C. Smooth

2.8 Houghova transformace

V podstatě jde o transformaci, kdy prostorovou oblast obrazu transformujme pomocí matematického aparátu do jiné oblasti. Původně Houghova transformace se zejména zabývala matematickým aparátem převodu prostorové oblasti obrazu do prostoru normálových přímek[24]:



Obr. 28. Parametrizace HT, základní pojmy

$$r = x\cos(\varphi) + y\sin(\varphi)$$

kde x, y jsou souřadnice bodu na přímce, která je určena:

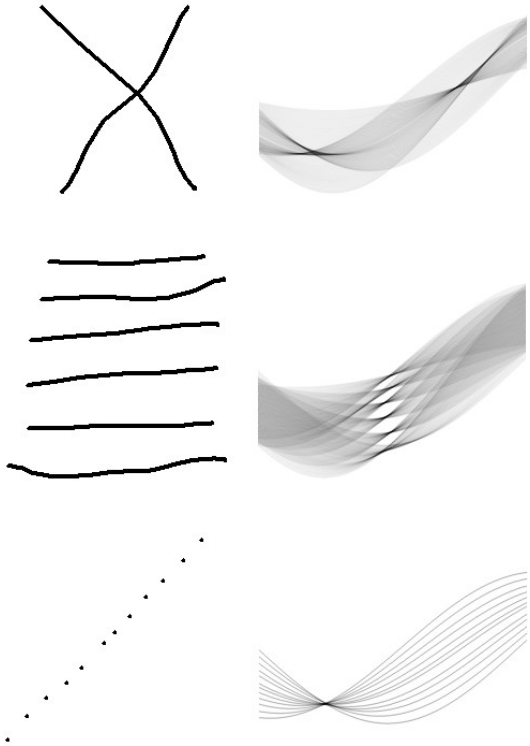
r - vzdáleností přímky od počátku souřadnicového systému,

φ - úhlem vedeným kolmo k přímce z počátku souřadnicového systému.

Pak všem bodům obrazu x_0, y_0, které splňují daná kritéria (např. předem provedeme detekci hran s prahováním, např. bod bude mít hodnotu jasu = 1), přiřadíme všechny možné hodnoty r, φ tak, aby byla splněna rovnice přímky v normálovém tvaru.

Postup transformace probíhá následovně:

1. Všem bodům prostoru r, φ na počátku nastavíme na hodnotu 0 a tyto body budeme považovat za akumulátor,
2. Dále budeme procházet bod po bodu transformovaným obrazem a tam, kde stanovíme, že se nachází zájmový bod, pak pro tento bod vygenerujeme všechny



Obr. 29: Příklady Houghovy transformace

[24] Výhodou je, že parametry přímek nezávisí na orientaci os.

možné kombinace hodnot r, φ, přičemž k akumulátoru na souřadnicích r, φ budeme přičítat +1.

3. V transformované oblasti r, φ pak na základě hodnoty akumulátoru určíme místa shluků s nejvyššími hodnotami, které pak signalizují výskyt přímky v obraze právě s parametry r, φ.

V rámci bodu č. 3. je možno použít jak jednoduché prahování, tak i shlukovou analýzu, neboť je nutno si uvědomit, že během transformací dochází vlivem kartézského systému obrazu a zaokrouhlování k statistickým chybám a pro danou přímku nemusí být souřadnice r, φ jednoznačné. Dále je nutno si uvědomit, že parametry r, φ nestanovují, kde přímka začíná a končí, k tomu je nutno použít nějaký programový způsob detekce „počátku a konce" přímky.

Existují způsoby, jak výpočet urychlit, které jsou publikovány pod názvem Randomizovaná Houghova transformace, přičemž tyto způsoby se opírají o tyto metody:

1. náhodný výběr bodů transformovaného obrazu, tzn. nejsou zpracovány všechny body obrazu,
2. odstranění již detekované přímky z obrazu v případě, že hodnota akumulátoru dosáhne předem definované hodnoty.

Transformaci lze použít i pro další geometrická primitiva jakými jsou kružnice a elipsy:

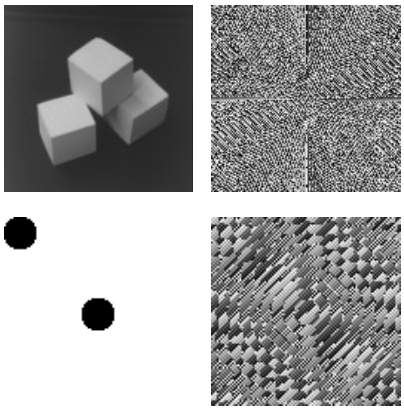
$$(x-a)^2+(y-b)^2=r^2$$

Výsledný transformovaný prostor má však již 3 rozměry a, b, r pro kružnici a 4 rozměry pro elipsu. V případě, že jsme schopni upřesnit velikost průměru kružnice či velikosti zakřivení elipsy, adekvátně nám klesne počet rozměrů a detekci lze urychlit a zjednodušit.

V současné době existuje již obecný popis Houghovy transformace pro libovolné grafické primitivum, které lze parametricky popsat a transformovat prostorový obraz do nového prostoru daného parametry primitiva, parametry však vytvářejí n-rozměrný prostor, ve kterém je nutno provádět detekci primitiva současně, nejčastěji shukovou analýzou.

2.9 Filtry využívající FT, WT

Pro filtraci obrazu je možno s výhodou použít i Fourierovu a Waveletovu (vlnkovou) transformaci, kdy obraz nejdříve převedeme do frekvenční/časově-měřítkové/ oblasti, v této oblasti provedeme její úpravu pomocí různých filtrů jako jsou různé propusti[25] či zvýrazníme nebo potlačíme některé části oblasti obrazu. Poté obraz převedeme zpětnou transformací do oblasti prostorové. Podle druhu transformace můžeme tak docílit nejen potlačení šumu, ale i odstranění detailů nebo naopak jejich zvýraznění.



Obr. 30: Příklady FT

Obr. 31: Pásmový filtr pomocí FT a IFT

Tyto filtry jsou však časově náročné i přesto, že mají pro diskrétní podobu signálů (obrázky) matematické předpisy, které výpočet výrazně urychlují, např. diskrétní rychlá Fourierova transformace. Existuje i předpis pro konkrétní jádra WT, proto filtry se zpravidla využívají jen tehdy, není-li kladen důraz na zpracování obrazu v reálném čase, v opačném případě se realizují pomocí specializovaných DSP.

Algoritmus Fourierovské transformace pro diskrétní hodnoty lze snadno realizovat[26], přičemž v současnosti existuje několik variant algoritmu podle toho, kolik prvků bude zahrnuto či podrobeno transformaci. Právě pro počet prvků $n=2^N$, kde $N > 0$, je algoritmus

[25] High, Low, Band pass – horní, dolní propusť, pásmová propusť nebo zádrž.

[26] Záměrně zde neuvádím matematický aparát, užitím kterého byl algoritmus odvozen, neboť je velice dobře zdokumentován v celé škále odborné literatury a není předmětem tohoto kurzu.

velice snadný, je též označován jako rychlá Fourierova transformace FFT, lze ho popsat těmito kroky[27] a probíhá dle Obr. 32.:

1. vytvoř vektor komplexních čísel[28] o velikosti n, vstupní hodnoty ulož do reálné části tohoto vektoru, imaginární část $\Im$ tohoto vektoru nastavíme na hodnotu 0,

2. prvky vstupního vektoru přeuspořádej tak, že nové pořadí prvku je dáno jeho inverzní bitovou hodnotou jeho indexu (viz tabulka níže pro hodnotu n=8),

Původní index		**Nový index**	
Dec.	**Bin.**	**Dec.**	**Bin.**
0	000	0	000
1	001	4	100
2	010	2	010
3	011	6	110
4	100	1	001
5	101	5	101
6	110	3	011
7	111	7	111

   přičemž výpočet se bude opírat o motýlkové schéma skládající se ze sekcí, v kterých se provádí motýlkový výpočet,

3. nastav l = 1, a bod 4. opakuj m krát, kde $n=2^m$, l můžeme považovat za vzdálenost mezi motýlky, a tento cyklus vytváří oddíly,

[27] Záměrně zde uvádím nerekurzivní algoritmus, který je výrazně efektivnější než algoritmy odvozené z matematických důkazů.

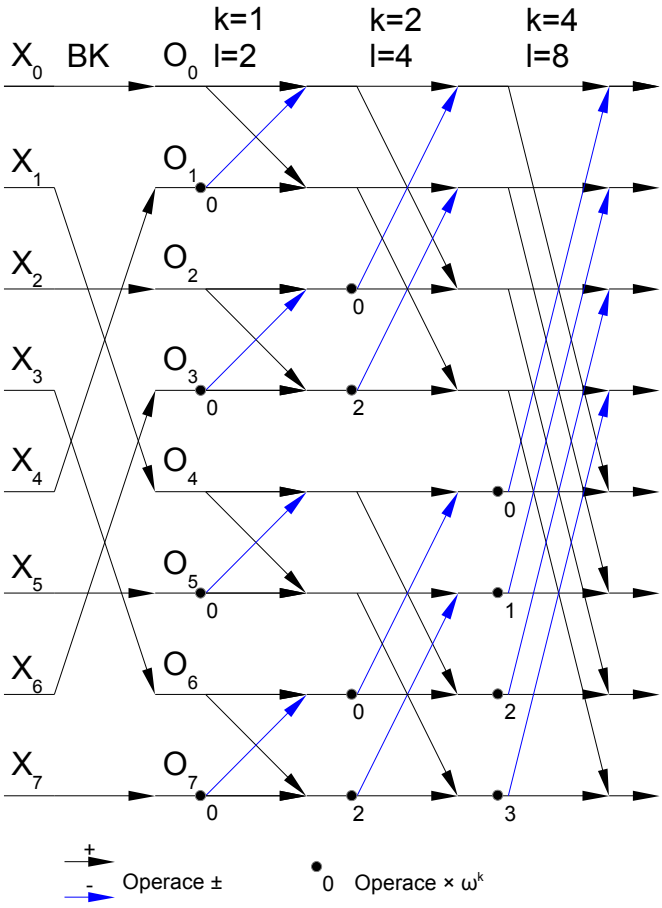
[28] Budeme uvažovat, že číslování prvků vektoru začíná od 0.

4. nastav k = l a zvětši hodnotu l = 2 × l a opakuj bod 5. pro: j = 0, 1, …, dokud j < k, toto můžeme považovat za cyklus vytvářející sekce, kdy k můžeme považovat za vzdálenost mezi křídly motýlka,

$$A = O[i]$$
$$B = O[i+k]$$
$$f = 2\pi\, j/n$$
$$C_{\Re} = B_{\Im}\cdot\sin(f) + B_{\Re}\cdot\cos(f)$$
$$C_{\Im} = B_{\Im}\cdot\cos(f) - B_{\Re}\cdot\sin(f)$$
$$B_{\Re} = A_{\Re} - C_{\Re}$$
$$B_{\Im} = A_{\Im} - C_{\Im}$$
$$A_{\Re} = A_{\Re} + C_{\Re}$$
$$A_{\Im} = A_{\Im} + C_{\Im}$$

5. posunuj motýlka po i = j, j + 1, ..., n s krokem l a vypočti motýlek na pozici:po ukončení cyklu z bodu 4., pokračuj v opakování bodu 4. dle počtu v bodu 3.

Algoritmus používá tzv. motýlkové schéma - viz Obr. 32. po n=8, kde jsou zobrazeny hodnoty, kterých nabývají proměnné I1 a I2 a je i znázorněna BK (bitová konverze). Algoritmus lze použít i pro inverzní FFT, dále iFFT, postačuje provést min. úpravy. V případě iFFT postačuje prohodit sin a cos a zaměnit znaménko rozdílu při výpočtu imaginární části pomocného čísla C.



Obr. 32: Motýlkový algoritmus FFT pro n=8

Výsledný vektor komplexních čísel při přímé transformaci pak obsahuje jednotlivé frekvenční složky s kmitočtem 0, Δ, 2Δ ,…, přičemž amplitudu a fázi získáme níže uvedeným vzorcem:

$$\Delta = f / n$$
$$A_i = \sqrt{A_{\Re i}^2 + A_{\Im i}^2}$$
$$\varphi = atan\left(\frac{A_{\Im i}}{A_{\Re i}}\right)$$

kde **f** je kmitočet vzorkování.

V případě, že máme n vzorků a n není mocninou základu 2, aplikujeme FFT buď podle publikovaných zdrojů, kdy např. n = a . b, na průběžné signály aplikujeme FFT pomocí okénka[29], které vyhovuje definici FFT výše, přičemž opětovně existuje mnoho variant úpravy okénkového algoritmu FFT (obdélníkové, Hannigovo, Hammigovo, Blackmanovo, trojúhelníkové, Kairesovo okno) tak, aby byla potlačena nespojitost frekvenční obálky transformace a zvětšen odstup signálu k šumu po transformaci. Okénko pak po vzorcích posouváme.

Hodnoty funkcí cos a sin mohou být již předem vypočteny a uloženy ve vektoru, kdy v algoritmu jen indexujeme správné prvky tohoto vektoru. Rovněž není potřeba používat komplexní čísla, ale běžné vektory, kdy si uvědomíme, že na sudých prvcích budou ležet reálné části čísel a na lichých jejich imaginární části.

Zde je část kódu v jazyku C, pro FFT i iFFT:

```
#define N 1024
int n = N;   //počet vzorků
double x[N]; //zde vložit data pro FFT, v případě iFFT, reálná složka
double y[N]; //pro iFFT fáze
int dir = 1; //určuje FFT nebo iFFT, 1 pro přímou FFT

//určíme mocninu
int m = 1;
for(int s=2; s<n; m++, s*=2){;}

//binární inverze, je 2x rychlejší než operace s posunem registrů
int i2 = n >> 1;
int j = i2;
int md = n-1;
for (int i=1; i <= i2; i++)
{
```

[29] V okénku využíváme již definovaný DFFT algoritmus a okénko postupně posouváme nad vzorky.

```
  if (i < j)
  {
    int tx = x[i];
    x[i] = x[j];
    x[j] = tx;
    tx = y[i];
    y[i] = y[j];
    y[j] = tx;
    if (i2 > j) //zrcadlení bitů přes negaci
    {
      int nj = md-j;
      int ni = md-i;
      int tx = x[ni];
      x[ni] = x[nj];
      x[nj] = tx;
      tx = y[ni];
      y[ni] = y[nj];
      y[nj] = tx;
    }
  }
  int k = i2;
  while (k <= j)
  {
    j -= k;
    k >>= 1;
  }
  j += k;
}

//vlastní FFT
double c1 = -1.0; //nahrazuje výpočet sin, cos
double c2 = 0.0;  //nahrazuje výpočet sin, cos
int l = 1;
for (int o=0;o<m;o++) //počet oddílů n=2^m
{
  int k = l;
  l <<= 1;
  double u1 = 1.0; //opět nahrazuje výpočet sin, cos
  double u2 = 0.0; //opět nahrazuje výpočet sin, cos
  for (int j=0;j<k;j++) //vytváří sekce motýlků
  {
    for (int i=j;i<n;i+=l) //posun motýlku v sekci
    {
      int ik = i + k; //výpočet motýlku
      double t1 = u1 * x[ik] - u2 * y[ik];
      double t2 = u1 * y[ik] + u2 * x[ik];
      x[ik] = x[i] - t1;
      y[ik] = y[i] - t2;
      x[i] += t1;
      y[i] += t2;
    }
    double z =  u1 * c1 - u2 * c2; //výpočet sin, cos
    u2 = u1 * c2 + u2 * c1;
    u1 = z;
  }
  c2 = sqrt((1.0 - c1) / 2.0); //výpočet sin, cos
```

```
  if (dir == 1) c2 = -c2;
  c1 = sqrt((1.0 + c1) / 2.0);
}
```

Poznámky ke kódu:

- Oproti uváděnému popisu algoritmu jsou optimalizovány jednotlivé cykly (kód lépe kopíruje postup znamenaný v obrázku).
- Nepoužívá komplexních čísel (x – je reálná složka, y – je imaginární složka komplexního čísla) z důvodů jednoduší adresní aritmetiky.
- Výpočet sin, cos je nahrazen skrze výpočet jednotkového vektoru v komplexní rovině, výpočet druhé odmocniny je rychlejší než výpočet trigonometrické funkce.
- Výpočet lze ještě optimalizovat skrze předpočítaný vektor hodnot sin a cos.
- Spektrum po FFT je jen poloviční, neboť je souměrné kolem osy y (amplitud), tj. okolo vzorku n/2.
- Hodnoty uložené v x a y po FFT je nutné normalizovat, tj. dělit počtem vzorků, tj. n.
- V případě iFFT, je nutné reálnou a imaginární část duplikovat do každé z polovin x, y souměrně okolo n/2 vzorků.

V případě realizace FFT uvnitř FPGA, je součástí FPGA prvek DSP, který provádí automaticky polovinu výpočtu motýlku, binární inverze indexů není potřeba, postačuje remapovat adresní sběrnici (otočit) a hodnoty sin a cos jsou předem předpočítány a uloženy v interní paměti FPGA. Pro n = 8, chceme-li dosáhnout maximální rychlosti výpočtu (3 hodinové cykly)[30], potřebujeme pro výpočet FFT 24 DSP bloků[31].

Aplikujeme-li FFT na obrázky, pak aplikujeme FFT nejprve na řádky a posléze na sloupce[32], stejnosměrnou složku máme však umístěnou v horním levém rohu a v jednotlivých kvadrantech pak vždy ½ spektra, opět spektrum je poloviční než počet prvků n a je souměrné

[30] Bloky DSP je vhodné na sebe vázat pomocí záchytných registrů, proto ty 3 hodinové impulsy – protože máme 3 oddíly.

[31] FPGA bývají běžně vybaveny DSP bloky v rozsahu 180 až 5520, což by nám dávalo možnost provádět FFT až pro n=512 v 9 hodinových cyklech.

[32] Opět zde neuvádím matematické odvození pro 2D FFT.

okolo diagonály. Chceme-li vytvořit filtr (horní propust, dolní propust, zádrž), ve výsledné matici vynulujeme (nebo zmenšíme) prvky potřebné oblasti. Z hlediska názornosti je lépe výslednou matici po FT kvadrantově přeuspořádat, pak filtry mají kruhovou podobu (střed, vnějšek, mezikruží), ale z hlediska výpočetního je toto přeuspořádání zbytečné a filtr může být aplikován přímo na výslednou matici (pak již ale nemá podobu kruhu).

WT - oproti FT, která nám poskytuje informaci o frekvenční oblasti signálu - nám poskytuje informaci o časově měřítkovém nebo též jinak - časově frekvenčním obrazu signálu. Opět pro výpočet WT existují varianty, které pracují s diskrétním signálem a existuje rychlý algoritmus pro výpočet přímé a zpětné WT[33], který byl odvozen na základě průchodu analyzovaného signálu skrze ideální dolní a horní propust (propusti jsou zrcadlové, tzn. zpětným výpočtem získáme identický vstupní signál), přičemž obecný algoritmus je pro WT následující:

1. Nastavíme číslo dekompozice na j = 0 a vstupní signál uložíme do vektoru jako a[j],.
2. Prodloužíme vektor a[j] o m-1 prvků vpravo i vlevo, m je velikost jádra konvoluce.
3. Prodloužený vektor podrobíme filtraci filtrem g, což je konvoluce s daným jádrem.
4. Z prodlouženého vektoru odstraníme prodloužení vlevo i vpravo o m-1 prvků.
5. Z výsledného vektoru uložíme polovinu hodnot (ε=0 - sudých či 1 - lichých[34]) do výsledné matice transformace d[j], tj. provedeme podvzorkování.
6. Bod 3. až 5. provedeme znova pro filtr h, výsledek však uložíme do vektoru a[j+1].
7. Zvětšíme hloubku dekompozice na j=j+1, není-li dosaženo potřebné hloubky, pokračujeme v bodě 2.

[33] Opět zde záměrně vynechávám celý matematický aparát WT, který je velice dobře zdokumentovaný v celé řadě odborných článků či odborné literatuře věnující se zpracování signálu.

[34] Pozor, číslování indexu začíná počítáním od 1, tzn. {4, 8, 2, 3, 5} při ε=0 nám zůstane {8, 3}.

Inverzní iWT můžeme popsat tímto algoritmem:

1. Nastavíme j na hloubku dekompozice z WT.
2. Vektor a(j) nadvzorkujeme, tzn. zdvojnásobme jeho délku, stejným způsobem jako jsme provedli podvzorkování (přidání sudých či lichých prvků),
3. Vektor a(j) podrobíme filtraci g a h tranponovaným filtrem – provedeme konvoluci a výsledky uložíme do $a_g(j)$, $a_h(j)$.
4. Výsledky ve vektorech $a_g(j)$, $a_h(j)$ sečteme.
5. Z výsledku vysekneme střední část délky vektoru a výsledek uložíme do d(j-1), první prvek je určen pozicí: m – (m % 2)-1+2 ε[35] a nový vektor označme jako a(j-1). V každém případě, při j=1 musí mít vektor d(0) délku n.
6. Snížíme j=j-1, není-li j=0, vrátíme s k bodu 2., jinak výsledný signál je uložen v a(0).

K tomuto obecnému algoritmu však existují již daleko rychlejší algoritmy (např. Lifting), zde je jednoduchý kód pro WT využívající Haarovy vlnky:

```
void wt(double *x, int n)
{
  for (int s = 2; s <= n; s <<= 1)
    for (int k = 0; k < n; k += s)
    {
      int s2 = s >> 1;
      x[k+s2] -= x[k];
      x[k] += x[k+s2] / 2;
    }
}
void iwt(double *x, int n)
{
  for (int s = n; s >=2 ; s >>= 1)
    for (int k = 0; k < n; k += s)
    {
      int s2 = s >> 1;
      x[k] -= x[k+s2] / 2;
      x[k+s2] += x[k];
  }
}
```

[35] Nezapomeňme, jak je definováno počítání, a to od 1. Znak % je zbytek po dělení.

Poznámky ke kódu:

- Hodnota n musí být $n=2^m$, kde $m>1$.
- Výsledek je vždy uložen v tomtéž vektoru.
- Koeficienty jsou uloženy „na přeskáčku“, tzn. v pořadí dělení intervalu 2, tj. pro n=8: $j^3_{x[0]}\ j^0_{x[1]}\ j^1_{x[2]}\ j^0_{x[3]}\ j^3_{x[4]}\ j^0_{x[5]}\ j^1_{x[6]}\ j^0_{x[7]}$.
- Tato transformace není zrovna ideální pro odstraňování šumu.

V případě 2D WT a iWT opětovně postupujeme nejprve po řádcích a pak po sloupcích. Tak jako v FT, výsledek transformací je uložen po kvadrantech obdobně jako u FT.

Samotná WT a iWT umí výrazně pomocí potlačit šum[36] v obraze, ale lze provádět i morfologické analýzy obrazu, např. lze sledovat změnu zemského povrchu z dlouhodobého hlediska. WT našla v současnosti uplatnění jako nová metoda JPEG2000 komprimace obrazu, kdy WT nahradila FT (resp. její kosinovou verzi) a je označována jako fraktální komprese obrazu.

FT a WT 2D transformace či konvoluce s jádrem odvozené z FT či WT v různém pořadí a při současné transformaci souřadnic mezi kartézskými a polárními mohou sloužit pro úpravu obrazu před vlastním rozpoznáváním či hledání shody se vzory. Např. polární transformace, FT, konvoluce s WT jádrem, iFT slouží jako nástroj pro úpravu obrazu před autentifikací osob pomocí tváře.

Rovněž FT může sloužit pro určení posunu, otočení a zoom obrazu. Postup vychází z matematického aparátu FT, kdy použijeme FT na obrazy, z transformovaných obrazů určíme podíl, na který použijeme iFT, kdy ve výsledném obraze získáme max. hodnotu v místě, které udává posun. (Pro rotaci pracujeme v polárních souřadnicích, pro zoom použijeme polární souřadnice v logaritmickém měřítku – problém se přenese do oblasti posunu). Existují i metody, jak provádět tyto výpočty v případě, že obraz obsahuje šumovou složku.

[36] V tomto případě je více jak vhodné provést nejdříve analýzu z hlediska jaký základ WT zvolit, tzn. jaký by měla mít tvar konvoluční matice g a h, neboť správnou volbou lze eliminovat šum, ale při min. kvadratické chybě upraveného signálu a původního signálu.

2.10 Radonova transformace

Radonova transformace[37] nepřímo souvisí s Houghovou transformací. Přímá transformace Radonova se opírá o integrál obrazové funkce[38] počítaný po přímce vedené v obrazové rovině. Přímka vystupuje ve vzorci v normálovém tvaru, tzn. její určující parametry jsou φ a r tak, jak se používají v Houghove transformaci.

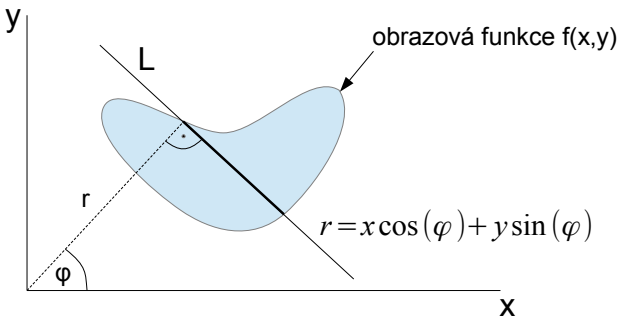
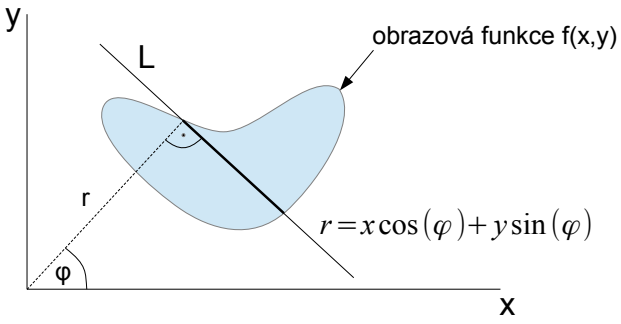


Fig. 33. Integrace po přímce L skrz obrazovou funkci f(x,y)

Integrál obrazové funkce pak provedeme pro všechny hodnoty φ (0 až π) a r (leží v R) a hodnoty integrálu opětovně vynášíme do transformované spektrální roviny s parametry φ a r zvané sinogram. V rovině sinogramu pak funkční hodnota integrálu obrazové funkce je vlastně úměrná součtu funkčních obrazových funkcí v jednotlivých bodech přímky.

K přímé Radonově transformaci existují matematické nástroje pro inverzní Radonovu transformaci, tzn. ze spektrální roviny sinogramu lze zpět zrekonstruovat původní hodnoty obrazové funkce v rovině x, y. Matematický aparát inverzní Radonovy transformace je odvozen od dvourozměrné Fourierovy transformace skrze definici Laplaciánu[39], přičemž mezi Radonovou a Fourierovou tranformací existují matematické vztahy z hlediska řezu, které pro odvození zpětné transformace jsou použity. Složitý matematický aparát je dnes nahrazen numerickým výpočtem využívající filtrovanou zpětnou projekci - FBP (Filtered Back Projection).



Obr. 34. Integrace po přímce obrazové funkce

Výpočet bez FBP lze zrealizovat takto:

[37] Odvodil ji matematik Radon narozený v Čechách.
[38] Funkce popisuje analyticky jas obrazového objektu v x, y rovině, tj. obrazové rovině.
[39] Laplacián je obecná definice druhé derivace obecné vícerozměrné funkce.

1. Data sinogramu uložíme do matice, kde v každém sloupci leží hodnota útlumu pro daný úhel φ.
2. Provedeme jednorozměrnou FFT nad každým sloupcem, výsledky uložíme do matice.
3. Nově vypočtenou matici transformujeme z polárních souřadnic do kartézských souřadnic pomocí zvolené sady interpolací, teď jsme vlastně získali Fourierovo dvourozměrné spektrum obrazu.
4. Provedeme dvourozměrnou inverzní FFT nad interpolovanou kartézskou soustavou souřadnic.

Přeneseme-li matematický popis Radonovy transformace do reálného světa počítačové tomografie[40], pak integrál obrazové funkce je nahrazen útlumem rentgenového záření procházejícího skrze snímanou část objektu[41], v CT je obecný objekt nahrazen lidským tělem. Výsledkem jedné části transformace pro daný úhel φ je běžný rentgenový snímek[42], který zachytí dané útlumy tkání[43]. Samotný tomograf pořizuje 50 až 500 snímků s různými úhly φ, tzn. rentgenové snímky se pořizují pod různými úhly, kdy zdroj záření a snímač se pootočí o určitý úhel. Z takto pořízených snímků se vytvoří spektrum – sinogram - snímaného obrazu, tj. ve výsledku lidského těla. Na takto pořízené spektrum – sinogram - se aplikuje inverzní Radonova transformace, čímž získáme informace o původním obrazu a jeho řezu.

Nutno si uvědomit, že sinogram z CT je zašuměn, pracuje se s digitalizovanými daty a dochází k zaokrouhlování během výpočtu, takže získaný obraz obsahuje spoustu artefaktů, které v originálním obrazu neexistují. Z těchto důvodů se hledají různá řešení, ať na úrovni matematického aparátu nebo na úrovni algoritmů FBP (Filtered back projection), jak šum a artefakty odstranit (zpravidla data získaná v bodu 3. před projekcí jsou filtrována za použití RAMP či Ramakrishan-Lakshiminarayanan (Ram-Lak) filtrů[44] a uživatelských filtrů, např. Hann, Hamming, Butterworth[45]).

---

40 CT se používá v lékařství při zpracování rentgenových snímků pořízených na tomografech, mezi laickou veřejností označovanou jako počítačová tomografie pod zkratkou CT (Computing Tomograph).

41 V CT se bere v úvahu, že útlum rentgenového záření není lineární, ale je exponenciálně závislý $I = I_0 e^{-\mu s}$, kde μ je koeficient útlumu a s dráha, I_0 je intenzita před vstupem do prostředí.

42 Snímky se nesnímají na fotografický materiál, ale snímají se již elektronickým detektorem.

43 Hodnotě r pak odpovídá snímaná rovina.

44 Jde o filtr typu horní propust.

45 Jde o filtry typu dolní propust.

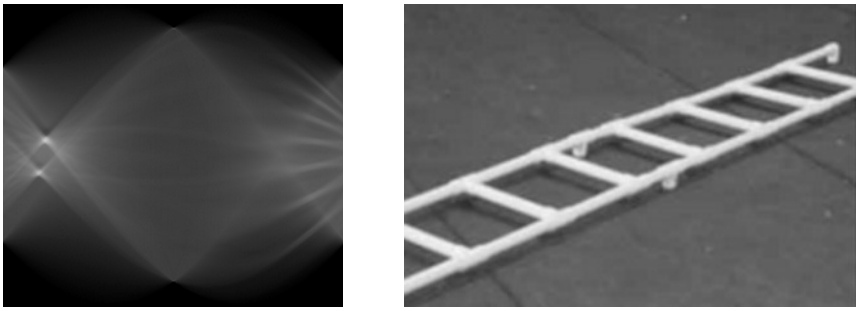
Iterativní rekonstrukce se opírají o to, že již máme nějakým způsobem zrekonstruovaný obraz (např. pomocí výše matematicky popsaného způsobu), pak provedeme nad takto získaným obrazem přímou transformaci a porovnáme ji s originálním pořízeným sinogramem. Pak v každé iteraci upravíme hodnotu pixelu zrekonstruovaného obrazu o rozdíl[46] mezi vypočteným sinogramem a originálním získaným sinogramem. Iterace zastavíme v okamžiku, kdy dosáhneme požadovanou shodu mezi originálním a vypočteným sinogramem nebo po požadovaném počtu kroků. Nevýhodou tohoto postupu je, že nemusí konvergovat vlivem šumu, ale i nedostatečným počtem projekcí.

Existuje i možnost zpětné transformace bez použití FT a FBP, kdy obraz budeme postupně iterativně upřesňovat. Výpočet lze charakterizovat takto:

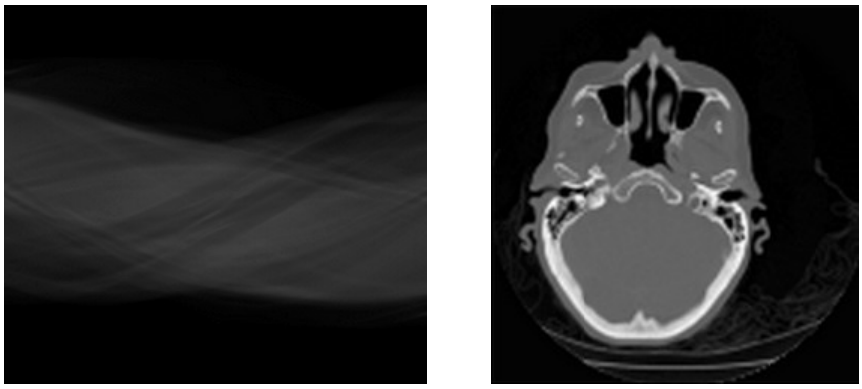
1. Všechny body rekonstruovaného obrazu nastav na hodnotu 0. Pokračuj bodem 2.
2. Zpracovávej postupně všechny sloupce sinogramu podle bodu 3.[47]
3. Pro vybraný sloupec sinogramu (tj. parametr φ) opakuj bod 3.
4. Pro každou hodnotu jasu ve sloupci, která určuje hodnotu r, urči polohu přímky v rekonstruovaném obraze (známe tak r a φ), a na této přímce spočítej průměrnou hodnotu jasu, tuto hodnotu odečti od hodnoty jasu sinogramu a získej tak rozdíl. Rozdíl pak rovnoměrně rozděl všem bodům na přímce tak, aby průměrný jas na přímce odpovídal hodnotě jasu v sinogramu.
5. Po zpracování celého sinogramu lze pokračovat dále již zpřesňujícím iterativním výpočtem uvedeným v předchozím odstavci.

[46] Hodnotu rozdílu nelze aplikovat přímo, ale je nutný k tomu jistý matematický aparát, který lze nalézt v odborné literatuře, zpravidla se jedná o systém řešení nelineárních diferenciálních rovnic, nebo též aparát zvaný EM - Expecation Maximizaton a ML - maximum likelihood.

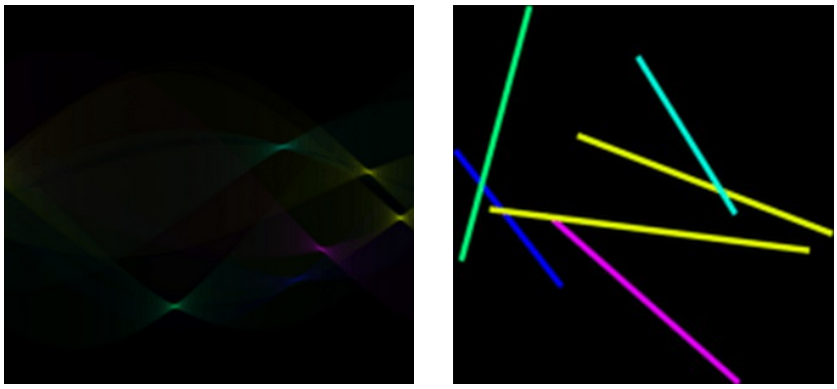
[47] Na pořadí zpracování nezáleží, ale s výhodou lze využít fakt, že hodnotám úhlů φ=0° a 90° odpovídají přímky jdoucí po řádcích a sloupcích v rekonstruované matici, nebo správnou volbou pořadí volby úhlů (0°, 90°, pak 45°, 135° a pak vždy polovina předchozího) zlepšuje finální odhad obrazu.



Obr. 35. Sinogram čb. fotografie



Obr. 36. Sinogram z CT a rekonstrukce řezu lidské lebky



Obr. 37. Sinogram barevného testovacího obrazu

Samotná Radonova transformace v případě použití pro účely filtrace může být použita pro odstranění malých objektů z obrazu či zvýraznění čárových prvků (např. používá se při zpracování fotografií nebeských těles a detekci meteoritů, kdy úspěšnost detekce lze zvýšit až 90%).

2.11 Filtry využívající matematickou morfologii

Jedná se o binární dilataci, erozi, otevření a uzavření. Filtry lze úspěšně aplikovat jak na binární obrazy, tak i obrazy ve stupních šedi. Filtry využívají faktu, že na obraz je možno pohlížet jako na množinu, přičemž mezi jednotlivými množinami můžeme realizovat součet či rozdíl ale i průnik. Dilatace je pak definována jako součet dvou množin dvou obrazů nebo téhož obrazu, který je posunut o daný vektor. Zpravidla je volena velikost vektoru rovna hodnotě 1, tj. právě o jeden zobrazovací bod. Vzhledem ke kartézské soustavě, používané pro vyjádření informace o obrazu, však může vektor nabýt i při hodnotě velikosti 1 až 8 směrů, přičemž výsledek operace je právě závislý na tomto směru. Eroze je definována jako množinový rozdíl, opět téhož obrazu posunutého o daný vektor, je to operace duální k operaci dilatace.

Otevření je definováno jako eroze následovaná dilatací a uzavření jako dilatace následovaná erozí. Jednotlivé operace se používají k vyplnění děr, ztenčení objektů, odstranění malých objektů, nalezení hranice mezi texturami[48], zjištění zastoupení textury dané velikosti.

0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0	1	1	1	0	0
0	0	0	1	1	1	0	0
0	0	1	0	1	1	0	0
0	1	0	0	1	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0

zdrojový obrázek

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

eroze (pro posun vlevo)

0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	1	1	0	1	1	0	0
1	1	0	0	1	1	0	0
0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0

dilatace (pro posun vlevo)

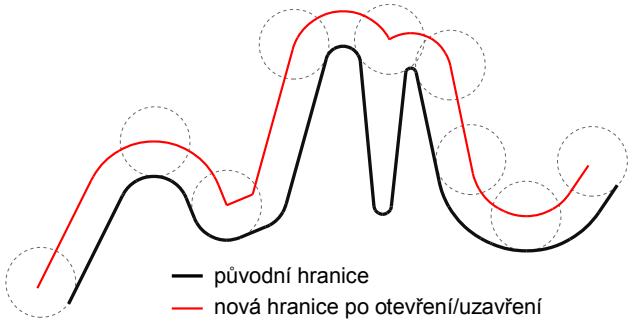
Obr. 38. Příklady operací eroze a dilatace

V případě nebinárních obrázků lze morfologické operace realizovat taktéž, ale nutno mít na zřeteli, že budeme hledat maxima jasu nebo maxima jasu v rámci doplňku. Operaci otevření a uzavření lze snadno realizovat pomocí valící se kružnice po hraně segmentu s identickým jasem[49] viz Chyba: zdroj odkazu nenalezen. Morfologické operace pak můžeme použít např. pro určení gradientu v obrázku.

[48] V textuře musí být rozdíl ve velikosti, pak menší texturu odstraníme uzavřením skrze masku odpovídající velikosti menší textuře, větší texturu otevřeme skrze masku, která je přibližně tak velká jako vzdálenost mezi prvky textury, nakonec upravíme hranici.

[49] Rozdíl mezi otevřením a uzavřením je, kterou stranu hranice použijeme pro valení kružnice, v příkladu na Chyba: zdroj odkazu nenalezen tedy horní nebo spodní strana původní hranice.

Dále morfologické operace využívají následující známé algoritmy. Všechny algoritmy lze matematicky popsat pomocí základních morfologických operátorů a množinové algebry.



Obr. 39. Otevření/uzavření nebinárních obrázků pomocí valící se kružnice

2.11.1 Hit or Miss filtry

Jde o jednoduchý algoritmus, který spadá mezi morfologické filtry, využívající práci s maskou n × n bitů, zpravidla 3 × 3 (jako konvoluční maska), realizující logické i množinové operace mezi maskou a binárním obrazem. Masku posouváme po celém obrazu jako konvoluční masku.

Operaci lze zapsat následovně:

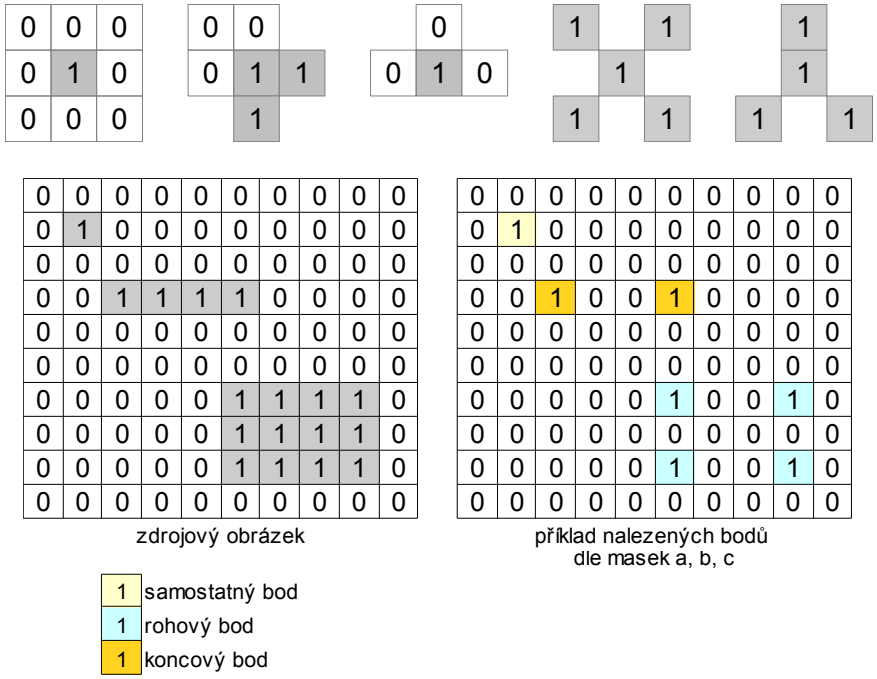
$$A = A - B_1 \cap A^c - B_2$$

Princip algoritmu:

Maskou projdi celý obrázek, tam, kde se shodují hodnoty pozadí a popředí dle masky nastav v místě řídícího bodu hodnotu 1, jinak 0. Řídící bod je uvnitř masky, zpravidla střed. Maska je tvořena hodnotami pozadí: 0 = B_1, hodnotami popředí: 1 = B_2 tam, kde v masce není určena hodnota, na hodnotě v obrázku nezávisí.

Algoritmus umí na základě vhodných masek - viz Obr. 40. - najít:

- samostatně ležící bod – maska a
- rohový bod – maska b - nutno rotovat po 90°
- koncový bod úsečky – maska c - nutno rotovat po 90°
- křížení či větvení – masky d, e - nutno rotovat po 45°
- či jiné oblasti, masky mohou mít i sudý rozměr.

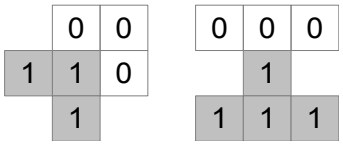


Obr. 40. Příklady základních masek Hit or Miss a výsledných operací (zleva a, b, c, d, e)

Existuje celá řada heuristických algoritmů, které používají v základu Hit or Miss filtr, kdy se používají dále dilatace, eroze, negace původního obrazu, k nim různé dodatkové operace jako je průnik, rozdíl, sjednocení ale i logický součet a součin. Některé algoritmy jsou uvedeny níže, ale kromě nich lze hledat i doplňky k segmentům obrazu, hledat min. rozměr segmentu, aby byl konvexní, a jiné operace.

2.11.2 Kostra segmentu

Algoritmem využívajícím masky lze vytvořit kostru segmentu, tzn. zúžit na 1 bod. Algoritmus probíhá iterativně:

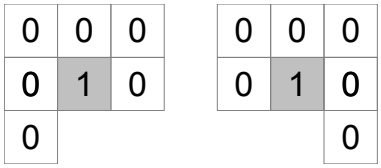


Obr. 41. Masky pro nalezení kostry obrazu

Najdi všechny body (nalezený bod se určuje vůči centru masky), které vyhovují maskám - viz Obr. 41., které rotuj o 90°, nalezené body odečti od původního obrázku, úpravy ukonči, když již nebyl nalezen v obrázku žádný bod nalezený pomocí masek.

Nevýhodou tohoto algoritmu je, že ve složitějších obrázcích vytváří parazitní větvení - pahýly. Tyto pahýly lze eliminovat aplikací níže uvedených vzorů dle. Obr. 43., vzory je nutno rotovat o 90°, ale použití těchto vzorů je omezené, neboť by mohli odstranit celou kostru.

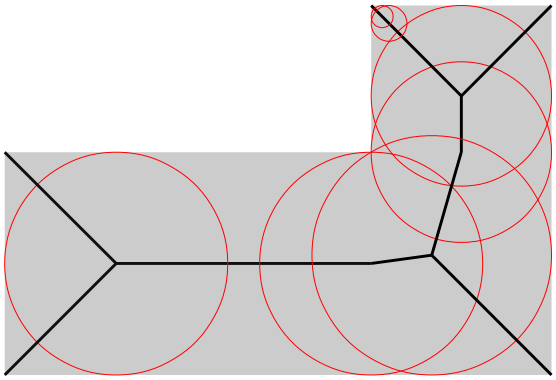
Obr. 42. Ukázka algoritmu hledání kostry segmentu



Obr. 43. Vzor odstraňující pahýly

V případě, že segment vytváří uzavřenou oblast, lze odstranit všechny pahýly bez obav, že by kostra objektu byla odstraněna úplně. Existují i další alternativy, jak eliminovat parazitní pahýly u segmentů, které nevytváří uzavřené struktury, např. tím, že nejdříve označíme konce segmentu, nalezneme kostru a tu pomocí dilatace budeme přičítat k nalezeným koncům.

Hledání kostry obrázku má i alternativní matematický algoritmus, který je však výpočetně náročný, kdy algoritmus hledá v obrázku místa, kde může vepsat kružnici o max. poloměru, přičemž si uschovává informaci o středu kružnice.



Obr. 44. Příklad kostry určené nejmenšími vepsanými kružnicemi

2.11.3 Obrys segmentu

Zjištění obrysu vyplněného segmentu v obrázku pomocí eroze lze tímto algoritmem:

Uschovej původní verzi obrázku, originální obrázek projdi erozní maskou, viz Obr. 45., tvořenou 3 × 3 body, resp. proveď operaci logického součinu z obrázku na obvodu masky do jejího centrálního řídícího bodu, výsledek odečti od uchovaného obrázku - viz Obr. 46., neboli:

1	1	1
1	1	1
1	1	1

Obr. 45. Erozní maska

$$X = A - (A \ominus B)$$

0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	1	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1

zdrojový obrázek

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0

eroze obrázku

0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	1	0	0
0	0	1	1	0	1	0	0
0	0	0	1	1	1	1	1
0	0	0	0	0	1	0	1
0	0	0	0	0	1	1	1

výsledek

Obr. 46. Příklad fungování hledání obrysu

2.11.4 Vyplnění ohraničené plochy

Vyplnění ohraničené plochy lze rovněž zajistit pomocí algoritmu využívajícího dilataci:

Uschovej původní obrázek, původní obrázek procházej dilatační maskou od inicializačního bodu, původní obrázek sečti s výsledkem. Dilatační maska je z 3 × 3 bodů s centrálním bodem ve středu.

	1	
1	1	1
	1	

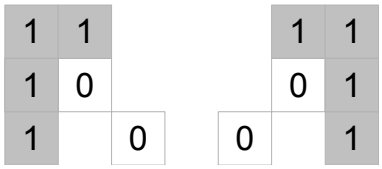
Obr. 47. Dilatační maska

Existují i jiné algoritmy, např. semínkový[50].

[50] Samotný semínkový algoritmus lze snadno popsat a vyhledat na Internetu, nutno však předeslat, že při bezmyšlenkovém přístupu k realizaci tohoto algoritmu, algoritmus v poměru test/změna barvy, pracuje velice neefektivně a provádí některé testy téhož bodu až 4x. Rovněž na Internetu jsou algoritmy, které nebudou pracovat bezchybně.

2.11.5 Rozšiřování objektů

Objekty, které mají malou kompaktnost, můžeme rozšířit tak, aby jejich kompaktnost narostla pomocí masek s dilatací - viz Obr. 49. Masky je nutno rotovat o 90°.



Obr. 48. Masky pro rozšíření

3 Detekce příznaků

RYCHLÝ NÁHLED KAPITOLY

Kapitola nastiňuje problémy jak předzpracovaný obraz dále zpracovat, tak aby obrazovou informaci bylo možno využít pro řídící algoritmy či jiné rozhodovací procesy.

CÍLE KAPITOLY

Po prostudování této kapitoly nahlédneme do problematiky:

- vyhledávání vzorů v obraze,
- identifikace vzorů a přiřazení vzorům jednoznačné klasifikátory.

KLÍČOVÁ SLOVA KAPITOLY

segmentace, parametrizace, moment, pravidlové systémy

Zpracování digitálního obrazu můžeme rozdělit na:

Úkolem je najít relevantní údaje v obraze a přiřadit jim příznaky. Je důležité, aby příznaky byly:

- spolehlivé - tzn. pro stejnou třídu objektů musí být příznak totožný,
- diskriminovatelné – tzn. pro každou třídu musí být příznak odlišný,
- nezávislé - tzn. mezi příznaky nejsou vazby.

Obecně detekce příznaků je zpravidla netriviální úloha vyžadující hlubší rozbor celého problému a k vlastní detekci se používají dále uvedené postupy. Přesto, že této problematice se věnuje velká pozornost, informací použitelných v obecných rovinách je poskrovnu.

3.1 Segmentace

Segmentace má za úkol rozdělit obraz do oblastí, jež můžeme označit za stejnorodé, např. oddělení pozadí od zájmových částí. Převážná část algoritmů používá prahování s použitím

histogramu a různé metody testování shodnosti okolí vůči testovanému obrazovému bodu nebo různé heuristické algoritmy.

3.1.1 Region growing

Algoritmus předpokládá, že obdrží již binárně zpracovaný obraz, který postupně procházíme a na základě odbarvovacích masek postupně obarvíme jednotlivé objekty.

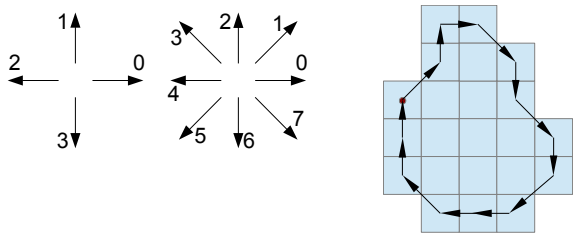
3.1.2 Štěpení a spojování oblastí

Obraz je nejdříve rozdělen do skupin, potom na základě testování jasu mohou nastat dva případy:

- je-li jas v oblasti různorodý, tak obraz rozštěpíme,
- jsou-li ve dvou sousedících oblastech jasy shodné, oblasti sloučíme.

3.1.3 Popis segmentů

- Pro popis segmentů je možno použít:
- velikost plochy – problém lze vyřešit jednoduchým sečtením počtu obrazových bodů patřících do segmentu,
- směr - určíme tak, že kolem objektu se snažíme opsat co nejmenší obdélník tak, aby objekt byl obsažen v obdélníku, jeho delší strana pak určuje směr,
- popis pomocí průběhu obvodu a jeho délky - řetězcový kód je složen z čísel, kterými jsou očíslovány úsečky - viz Obr. 47., které reprezentují směr, ve kterém se nachází následující obrazový bod patřící do obvodu segmentu. Pro uvedený obrazec - viz Obr. 47. , je řetězcový kód od výchozího bodu: 1207676544322. Někdy se tento kód ještě diferencuje tak, že zkoumáme, kolikrát musíme před dalším krokem obrázkem otočit po chodu hodinových ručiček, aby následující chod byl ve směru 0, tzn. pro uvedený obrázek je: 1167717770770, čili



Obr. 49. Příklad číslování směrů pro určení řetězcového kódu (vlevo) a příklad segmentu pro ukázku řetězcového kódu (vpravo)

pro výpočet i-té pozice: $x_i = (n + x_i - x_{i-1}) \bmod n$,

kde: n = 4 pro 4 směrový, 8 pro osmi-směrový chod, x0=0. Počet chodů nám dává délku obvodu,

- kompaktnost – určuje se jako poměr obvodu segmentu (její druhé mocniny) k jeho ploše, kruh má toto číslo nejmenší,
- výstřednost, podloudnost, pravoúhlost - můžeme ji určit buď z poměru stran opsaného obdélníku s co nejmenším obsahem, nebo jako měření nejdelších na sobě kolmých tětiv. U pravoúhlosti jde o poměr mezi plochou zkoumaného objektu a velikostí obsahu opsaného obdélníka,
- Eulerovo číslo, tj. počet děr v segmentu,
- vzdálenost bodů od hrany či její průměrná hodnota - viz Obr. 51. Výpočet lze zrealizovat pomocí eroze, ale existuje i efektivní algoritmus popsaný např. Rosenfeldem a Pfaltzem.

0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	0	1	1	1	1	1	0
0	0	1	1	1	1	1	0
0	0	1	1	1	1	1	0
0	0	1	1	1	1	1	0
0	0	1	1	1	1	1	0
0	0	0	0	0	0	0	0

zdrojový obrázek

0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	0	1	1	1	1	1	0
0	0	1	2	2	2	1	0
0	0	1	2	3	2	1	0
0	0	1	2	2	2	1	0
0	0	1	1	1	1	1	0
0	0	0	0	0	0	0	0

výpočet vzdálenosti

Obr. 50. Vzdálenost bodů od hrany

3.1.4 Moment

Moment je definovaný jednoduchým matematickým vztahem:

$$m^{p,q} = \sum_{\Omega} \sum_{\Omega} x^p y^q f_{x,y}$$

$$m_c^{p,q} = \sum_{\Omega} \sum_{\Omega} (x - x_c)^p (y - y_c)^q f_{x,y}$$

$$x_c = \frac{m^{1,0}}{m^{0,0}} \qquad y_c = \frac{m^{0,1}}{m^{0,0}}$$

$$m_s^{p,q} = \frac{m^{p,q}}{m^{0,0\varepsilon}} \qquad \varepsilon = \frac{p+q}{2} + 1$$

kde: **p, q** - jsou celočíselné konstanty větší jak 0 a udávající stupeň, řád momentu,

$\mathbf{f_{x,y}}$ - hodnota jasu v bodu x, y,

Ω - je množina bodů spadající do daného klasifikovaného segmentu.

Moment je citlivý na posun, velikost i rotaci segmentu, což lze korigovat již při výpočtu momentu, kdy moment budeme počítat k těžišti (centrum) objektu m_c, a zavedeme normalizaci na velikost (size) segmentu m_s, do výpočtu můžeme zahrnout i natočení segmentu. Z hodnot momentů můžeme určit jak podlouhlost, tak i úhel natočení objektu i integrální jas objektu.

3.2 Klasifikace

Výstupem klasifikátoru je jednoznačný popis - numerický či syntaktický zkoumaného obrazu s možností realizovat na základě tohoto popisu regulační smyčku či realizovat řídící algoritmus nebo též poskytnout již potřebná finální data pro nějaký vyšší rozhodovací algoritmus či systém.

Nelze obecně definovat způsob klasifikace, neboť záleží na faktoru, co má být v obraze klasifikováno, např. o zcela jiný způsob půjde v případě, že výstupem klasifikátoru budou informace typu, které získáme při OCR - optickém rozpoznávání písma nebo rozpoznávání objektů, u kterých předem známe tvar, např. značky, vodící čáry či jinak vymezující značení, nebo obecných objektů (úsečky, kružnice, rohy a jiné geometrické útvary), ze kterých dále vyvozujeme další závěry nebo provádíme další výpočty, např. vytvoření 3D - viz zpětná stereo-projekce. Samostatnou kapitolou jsou nástroje pro klasifikaci v medicínské oblasti, kdy chceme nalézt objekty, které signalizují anatomické změny – výdutě, nádory, zúžení či slepá ramena.

Samotnou klasifikaci je možné provádět pomocí různých klasifikátorů, např.:

- statistických, např. pomocí shlukové analýzy nebo analýzy SVM (Support Vector Machines)[51], tzn. s předpokladem, že segmenty v n rozměrném prostoru vytvářejí shluky či jsou od sebe odděleny (každému segmentu náleží jeden prostor), přičemž pro právě hledanou klasifikaci buď vypočteme Eulerovou vzdálenost od již předem klasifikovaných obrazů a vybereme pak ten klasifikátor, který má min. vzdálenost, nebo v n rozměrném prostoru se snažíme oddělit jednotlivé hodnoty klasifikátorů pomocí přímky či jiné funkce (polynomické, sigmoid, Gauss),

[51] Pracuje podobně jako neuronová síť, ale lze snadněji spočítat průběh dělící funkce.

- pravidlových, syntaktických či fuzzy-logic, kdy předpokládáme, že v obraze jsme již provedli segmentaci a známe popis segmentů a na základě těchto hodnot zkoušíme pomocí pravidel nebo syntaxí určit hodnotu klasifikátoru,
- neuronových,
- transformačních či korelačních – např. pro triviální klasifikaci přímek se nabízí právě Houghova tranformace, nebo obrazy porovnat na základě výpočtu korelace.

Neuronové sítě jsou popsány v učebním textu: Autonomní systémy, ty mohou být použity jako filtr a obraz upravit (např. síť typu BAM), např. odstraněním chyb v obraze nebo též stanovit přímou hodnotu klasifikátoru (např. neuronové sítě lze s výhodou použít pro učící se systém OCR).

Zde je příklad pravidlového systému, který má za úkol rozpoznat konkrétní tvar, systém může pracovat např. takto:

1. V pravidlovém systému máme uloženo n morfologických prvků a před vlastní klasifikací máme uloženo v bázi znalostí řetězec či strom, ve kterém máme uložen výskyt jednotlivých morfologických prvků v každém klasifikovaném tvaru.
2. Po předložení klasifikovaného tvaru začneme v něm hledat jednotlivé morfologické prvky, vždy od většího k menšímu.
3. V případě, že morfologický prvek nalezneme, tuto část z klasifikovaného tvaru vyjmeme a vytváříme řetězec (strom), do kterého vkládáme informaci o nalezeném morfologickém prvku[52], v případě, že ještě v klasifikovaném tvaru zbývá nezpracovaná část, postupujeme znovu bodem 2, jinak pokračujeme bodem 4.
4. Z báze znalostí pak porovnáme získaný řetězec s řetězcem v bázi znalostí nebo hledáme danou větev ve stromu, při nalezení shody v řetězcích či nalezení větve ve stromu jsme nalezli i daný klasifikovaný tvar.

[52] Sytém může obsahovat i morfologické prvky, které do syntaxe nebudou zahrnuty – např. okraje okolo prohledávaného tvaru či jiné údaje, rovněž ve stromu může a nemusí být uložena informace o pozici, kde morfologický prvek byl nalezen.

4 Zpětná stereo-projekce

RYCHLÝ NÁHLED KAPITOLY

V této kapitole je ozřejmeny postupy pomocích kterých můžeme pomocí 2D obrazů vymodelovat 3D snímanou scénu.

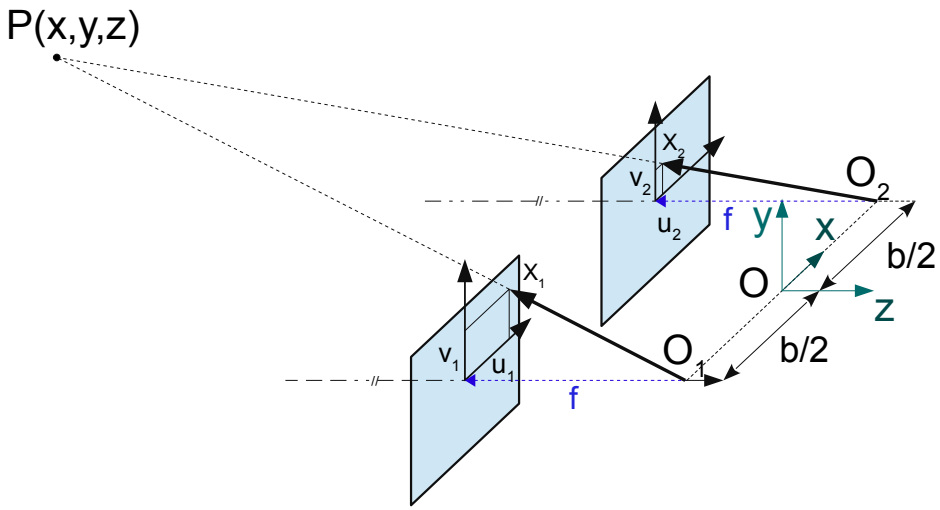
CÍLE KAPITOLY

Po prostudování této kapitoly porozumíme základním matematickým postupům zpětné stereo-projekce.

KLÍČOVÁ SLOVA KAPITOLY

stero-projekce, kamerový systém, souřadnicový systém

Zpětná stereo-projekce nám umožňuje na základě dvou pořízených obrazů téže scény vytvořit 3D model scény nebo např. určit vzdálenost určitého vybraného bodu P od kamerové osy. Nejjednodušší je situace v případě, kdy použijeme dvě totožné kamery se stejnými ohniskovými vzdálenostmi f, které mají rovnoběžné osy snímání a jsou umístěny v téže snímací rovině - viz Obr. 43., a počátek souřadného systému O položíme na střed mezi kamerami.



Obr. 51. Zobrazení zpětné stereoprojekce

Pak z podobnosti příslušných trojúhelníků můžeme pro bod P velice rychle odvodit:

$$z=\frac{bf}{u_1-u_2} \qquad x=\frac{b(u_1+u_2)}{2(u_1-u_2)} \qquad y=\frac{b(v_1+v_2)}{2(u_1-u_2)}$$

Výpočet lze zrealizovat, i když dříve uvedený předpoklad nesplníme, tzn. osy kamer nemusí být rovnoběžné, kamery jsou na různých souřadnicích a mají různou ohniskovou vzdálenost, je však nutno k problému přistoupit skrze homogenní souřadnice, vyjádřit pomocí nich vlastnosti kamer a zavést transformace souřadnicových systémů kamer do souřadnicového systému O. Potom:

$$u_i=P_i\cdot x, \quad x=(x,y,z,1), \quad u_i=(qu_i,qv_i,q), \quad q\neq 0$$

Matice P_i má rozměr 3x4 prvky a popisuje transformaci bodu P do zobrazovacích rovin kamer. Její hodnoty pak určíme kalibrací kamerového systému, min. počet kalibračních bodů je 6 pro každou kameru. Poněvadž systém rovnic je předeterminován, měření souřadnic u, v je zatíženo chybami, máme však možnost systém řešit metodou nejmenších čtverců a chyby minimalizovat.

Pro normované souřadnice a kameru bez zkreslení perspektivy můžeme pro bod x' [53]na zobrazovací ploše kamery a obecný bod x v prostoru zapsat:

$$z\begin{vmatrix}x_1\\y_1\\1\end{vmatrix}=\begin{vmatrix}f&0&0\\0&f&0\\0&0&1\end{vmatrix}\begin{vmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\end{vmatrix}\begin{vmatrix}x\\y\\z\\1\end{vmatrix} \qquad pro: x'_{R2}=\begin{vmatrix}x_1\\y_1\\1\end{vmatrix}; K_f=\begin{vmatrix}f&0&0\\0&f&0\\0&0&1\end{vmatrix}; \Pi_0=\begin{vmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\end{vmatrix}$$

$$\lambda x_{1_{R2}}=K_f \Pi_0 X_{R3} \qquad \textit{po přidání translace souřadnic kamery:}$$

$$\lambda\begin{vmatrix}x_1\\y_1\\1\end{vmatrix}=\begin{vmatrix}fs_x&fs_\theta&o_x\\0&fs_y&o_y\\0&0&1\end{vmatrix}\Pi_0\begin{vmatrix}x\\y\\z\\1\end{vmatrix}$$

$$\lambda x_{1_{R2}}=K\,\Pi_0 X_{R3}$$

kde: **K** je transformační matice kamerového systému a zahrnuje transformaci středu objektivu vůči souřadnicovému systému umístěnému v levém horním rohu, zachování poměru stran x/y, zkosení roviny a ohniskovou vzdálenost.

[53] R2 - dvourozměrné, R3 - třírozměrné vyjádření.

Pro 2 kamery a totožný bod na dvou obrázcích platí: $\lambda_1 x_1 = R \lambda_2 x_2 + T$

kde λ, R, T jsou neznámé hodnoty a parametry představují:

R - otočení kamery 2 vůči kameře 1,

T - posun kamery 2 vůči kameře 1,

λ - obecný koeficient, zahrnující hloubku.

Pak hledáme řešení výše uvedené rovnice, kterou formalizujeme. Máme několik matematických možností, jak systém vyřešit: pro lineární řešení je nutné mít min. 8 párů bodů, pak však máme více rovnic než je neznámých (můžeme zahrnout i více než 8 bodů) a z důvodů chyb při odečtu kamerových souřadnic používáme přibližné řešení metodou nejmenších čtverců. Pro nelineární řešení postačuje již 5 párů bodů, avšak obdržíme až 10 řešení.

Známe-li z rektifikace parametry kamer, a jsou-li kamery paralelně, můžeme na základě disparity bodů v obraze určit hloubku bodu v obrazu.

Není třeba zdůrazňovat, že chceme-li, aby systém sám vytvářel 3D model zcela automaticky, neobejdeme se bez předchozích nástrojů jako je odstraňování šumu, detekce hran a rohů, nástrojů pro ztotožňování téhož prvku ve dvou různých obrazech (morfologické prohledávání) a až posléze můžeme přistoupit ke konstrukci 3D modelu.

5 Pár slov na závěr

Tento dokument nemohl obsáhnout velice široký rozsah problematiky týkající se digitálního zpracování obrazu[54], ale je jen jistým vodítkem, aby bylo možno již v této chvíli přistoupit k fyzické realizaci jednotlivých výše probraných postupů při digitálním zpracování obrazu na úrovni FPGA a porovnat toto řešení (v oblasti rychlosti, spotřeby energie, rychlosti vývoje) s běžnými postupy využívající běžné procesory na úrovni jazyků C a Assembler. Materiály, které se dotýkají programové části, neuronových sítí, speciálních matematických postupů (např. stimulované žíhání) a FPGA jsou uvedeny v dokumentu: **Autonomní řídící systémy**.

[54] Dokument bude dále doplňován tak, aby dále pokryl mnohé postupy, na kterých se teprve pracuje (např. morfologické hledání hran) nebo nebyly zmíněny z důvodů časové náročnosti zpracování této problematiky, aby ji bylo možno prezentovat ve zhuštěné formě.