EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Vysoká škola báňská – Technická univerzita Ostrava

ZPRACOVÁNÍ OBRAZU V MĚŘICÍ A ŘÍDICÍ TECHNICE

učební text

Radovan Hájovský, Radka Pustková, František Kutálek

Ostrava 2012

Recenze: Prof. Dr. Ing. Miroslav Pokorný
Prof. RNDr. Alena Lukasová,CSc.

Název: Zpracování obrazu v měřicí a řídicí technice
Autor: Radovan Hájovský, Radka Pustková, František Kutálek
Vydání: první, 2012
Počet stran: 189
Náklad: 20
Vydavatel a tisk: Ediční středisko VŠB – TUO

Studijní materiály pro studijní obor Měřicí a řídicí technika, Elektronika, Fakulty elektrotechniky a informatiky
Jazyková korektura: nebyla provedena.

Určeno pro projekt:
Operační program Rozvoj lidských zdrojů
Název: E-learningové prvky pro podporu výuky odborných a technických předmětů
Číslo: CZ.O4.01.3/3.2.15.2/0326
Realizace: VŠB – Technická univerzita Ostrava
Projekt je spolufinancován z prostředků ESF a státního rozpočtu ČR

ISBN 978-80-248-2596-0

OBSAH

POKYNY KE STUDIU

Zpracování obrazu v měřicí a řídicí technice

Pro předmět Zpracování obrazu v měřicí a řídicí technice v 2. semestru druhého ročníku magisterského studijního oboru Měřicí a řídicí technika a Elektronika jste obdrželi studijní balík obsahující

- integrované skriptum pro distanční studium obsahující i pokyny ke studiu
- CD-ROM s doplňkovými animacemi vybraných částí kapitol
- harmonogram průběhu semestru a rozvrh prezenční části
- rozdělení studentů do skupin k jednotlivým tutorům a kontakty na tutory
- kontakt na studijní oddělení

Prerekvizity

Pro studium tohoto předmětu se předpokládá absolvování předmětu Matlab a simulace a také předmětu Signály a soustavy.

Cílem předmětu

je seznámení se základními pojmy a principy digitálního zpracování obrazu pomocí SW systému Matlab/Simulink spolu se speciální nástavbou Video and Image Processing Blockset. Probíraná problematika je zaměřena především na možnosti aplikace algoritmů zpracování obrazu v technické praxi. Po prostudování modulu by měl student být schopen zanalyzovat vstupní úlohu z oblasti zpracování obrazu, navrhnout schéma pomocí Matlab/Simulink a aplikovat ji na reálnou úlohu v reálném čase.

Pro koho je předmět určen

Modul je zařazen do magisterského studia oborů Měřicí a řídicí technika a Elektronika studijního programu Elektrotechnika, ale může jej studovat i zájemce z kteréhokoliv jiného oboru, pokud splňuje požadované prerekvizity.

Skriptum se dělí na části, kapitoly, které odpovídají logickému dělení studované látky, ale nejsou stejně obsáhlé. Předpokládaná doba ke studiu kapitoly se může výrazně lišit, proto jsou velké kapitoly děleny dále na číslované podkapitoly a těm odpovídá níže popsaná struktura.

Při studiu každé kapitoly doporučujeme následující postup:

Čas ke studiu: xx hodin

Na úvod kapitoly je uveden **čas** potřebný k prostudování látky. Čas je orientační a může vám sloužit jako hrubé vodítko pro rozvržení studia celého předmětu či kapitoly. Někomu se čas může zdát příliš dlouhý, někomu naopak. Jsou studenti, kteří se s touto problematikou ještě nikdy nesetkali a naopak takoví, kteří již v tomto oboru mají bohaté zkušenosti.

Cíl: Po prostudování tohoto odstavce budete umět

- popsat ...
- definovat ...
- vyřešit ...

Ihned potom jsou uvedeny cíle, kterých máte dosáhnout po prostudování této kapitoly – konkrétní dovednosti, znalosti.

VÝKLAD

Následuje vlastní výklad studované látky, zavedení nových pojmů, jejich vysvětlení, vše doprovázeno obrázky, tabulkami, řešenými příklady, odkazy na animace.

Na závěr kapitoly jsou zopakovány hlavní pojmy, které si v ní máte osvojit. Pokud některému z nich ještě nerozumíte, vraťte se k nim ještě jednou.

Otázky

Pro ověření, že jste dobře a úplně látku kapitoly zvládli, máte k dispozici několik teoretických otázek.

Úlohy k řešení 1.1.

Protože většina teoretických pojmů tohoto předmětu má bezprostřední význam a využití v technické praxi, jsou Vám nakonec předkládány i praktické úlohy k řešení. V nich je hlavní význam předmětu a schopnost aplikovat čerstvě nabyté znalosti při řešení reálných situací.

KLÍČ K ŘEŠENÍ

Výsledky zadaných příkladů i teoretických otázek výše jsou uvedeny v závěru učebnice v Klíči k řešení. Používejte je až po vlastním vyřešení úloh, jen tak si samokontrolou ověříte, že jste obsah kapitoly skutečně úplně zvládli.

Úspěšné a příjemné studium s touto učebnicí Vám přejí autoři výukového materiálu

Radovan Hájovský, Radka Pustková, František Kutálek

1. ZÁKLADY PRÁCE S OBRAZEM

Čas ke studiu: 1,5 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- popsat princip digitalizace obrazu
- definovat základní pojmy z oblasti zpracování obrazu
- vysvětlit pojem prostorového a jasového rozlišení obrazu

Výklad

1.1 Úvod

Pokud chceme zpracovávat obraz na počítači, je nutné ho nejprve určitým způsobem převést do digitální podoby. Tímto se myslí převedení okolního (analogového) reálného světa, který vidíme, do digitální podoby. V podstatě jde o konverzi optické veličiny na elektrickou. Oblast digitálního zpracování a analýzy obrazu je dnes již velmi rozvinout disciplínou, zejména se uplatňuje v široké škále průmyslových aplikací, kde na metodách rozpoznání obrazu jsou založeny různé druhy automatizačních procesů.

Autoři textu by chtěli poděkovat tvůrcům literatury, zejména [1], [2], [4] a [7], jejichž materiály sloužily jako zdroj teoretických informací pro přednášky. V mnoha případech byly informace z uvedených textů použity přímo ve výukových kapitolách, kde jsou také řádně uvedeny jako citační zdroj.

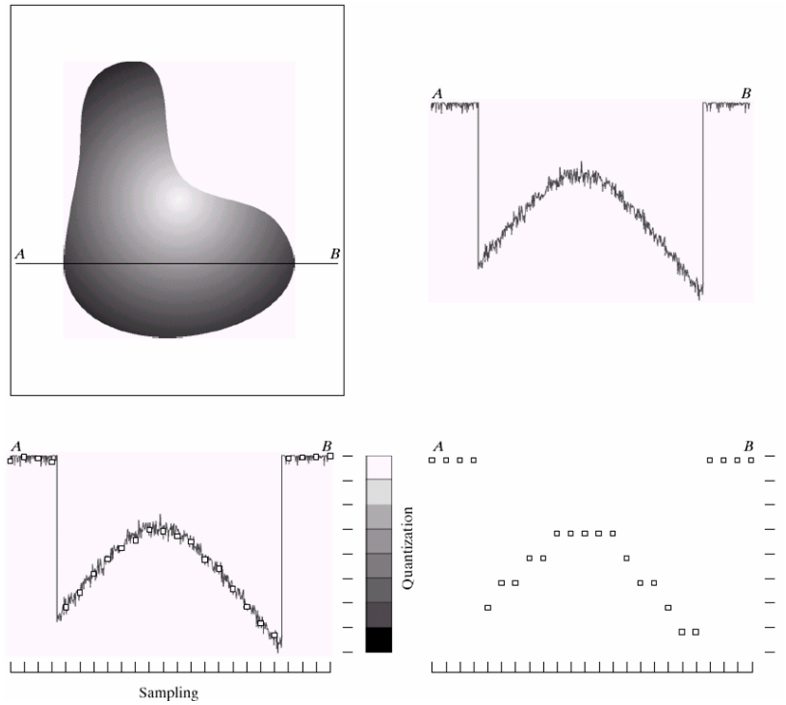
1.1.1 Digitalizace

Oblastí našeho zájmu probíraného v těchto studijních materiálech je digitální zpracování obrazu, které slouží pro řešení dané úlohy z technické praxe. V případě, že chceme obraz zpracovávat pomocí výpočetní techniky, je nutno jej nejprve převést do digitální formy - **Digitalizace obrazu.**

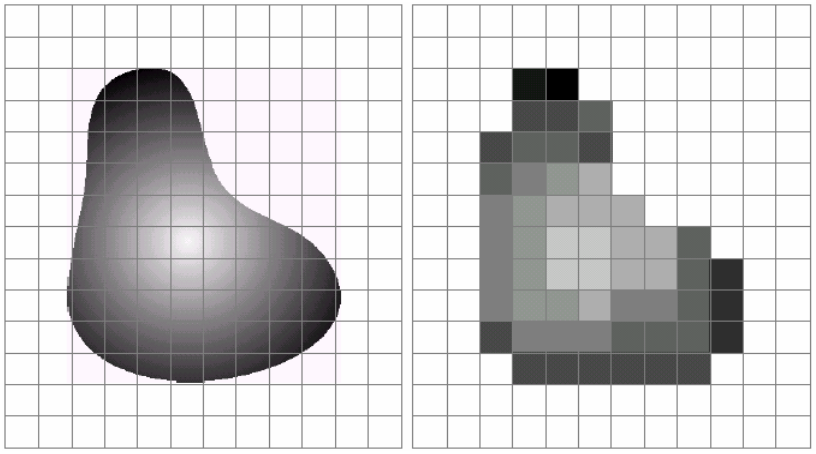
Digitalizace obrazu je proces, který je analogický k obecné digitalizaci jakéhokoliv analogového signálu do digitální podoby. Zahrnuje vzorkování, kvantování a kódování obrazového signálu. V dalším textu se zatím kódováním zabývat nebudeme a rozebereme si pouze první dvě etapy procesu digitalizace.

- **Vzorkování** je diskretizace souřadnic obrazu, tzn. vytvoření matice s M x N body obrazové funkce $f(x,y)$. Hovoříme o tzv. prostorovém rozlišení.
- **Kvantování** představuje diskretizace jasových úrovní, tzn. rozdělení spojité jasové úrovně každého obrázku do K úrovní. Hovoříme o jasovém rozlišení.

Proces digitalizace je zjednodušeně ukázán na obrázcích Obr. 1.1 a Obr. 1.2.
Díky **kvantování** nabývá jasová funkce v digitalizovaných obrazech celočíselných hodnot.
Čím jemnější je vzorkování (čím větší *M*, *N*) a kvantování (čím větší je počet jasových úrovní), tím lépe je aproximován původní spojitý obrazový signál.



Obr. 1.1 Ukázka vytvoření digitálního obrázku. A) Spojitý obraz. B) Vybraná přímka AB spojitého obrázku použitá k vysvětlení procesu vzorkování a kvantování. C) Vzorkování a kvantování. D) Digitalizovaná část obrázku. [1]



Obr. 1.2 A) Spojitý obraz projektovaný do vzorkovací mřížky. B) Výsledný obraz po vzorkování a kvantizaci. [1]

Při vytváření digitálního obrazu digitalizací spojité obrazové funkce *f(x,y)* se můžeme setkat s následujícími pojmy:

Interval vzorkování – vzdálenost mezi nejbližšími vzorkovacími body v obraze. Otázku vzdálenosti vzorků (nebo jinak řečeno plošné vzorkovací frekvence) řeší **Shannonova věta o vzorkování**.

- Pro jednorozměrné signály – vzorkovací frekvence, dle věty o vzorkování, musí být minimálně dvakrát větší než nejvyšší frekvence ve vzorkovaném signálu, a to z důvodu aby tento signál byl zpětně rekonstruovatelný.
- Pro dvojrozměrné signály (obrazy) – je zapotřebí zvolit interval vzorkování tak, aby byl minimálně dvakrát menší než velikost nejmenšího detailu v obraze.
  Pro optimální zpracování obrazového signálu se nejčastěji používá velikost vzorkovacího elementu 5-krát menší než je teoretická mez daná vzorkovací větou. Tedy při zpracování obrazu je dostatečnou vzorkovací frekvencí 1/5 velikosti nejmenšího detailu.

Vzorkovací mřížka - plošné uspořádání bodů při vzorkování. Obvykle se používá pravidelná mřížka. Existují jen tři pravidelné mnohoúhelníky jejichž síť plně pokrývá rovinu a to:

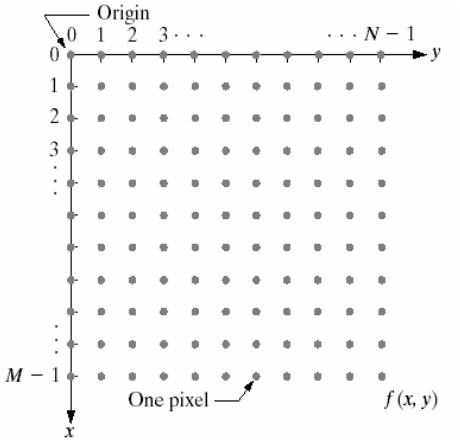
- rovnostranné trojúhelníky
- čtverce (nejčastěji v praxi)
- pravidelné šestiúhelníky

Kvantovací úroveň - amplituda ve kvantovaném obraze musí být pro zpracování počítačem vyjádřena jako digitální údaj.
Počet kvantovacích úrovní má být dostatečně velký, aby byly přesně vyjádřeny jemné detaily obrazu, nevznikaly falešné obrysy a aby se citlivost zařízení blížila citlivosti lidského oka.
Většinou se používá kvantování do *k* stejných intervalů. Jestliže je pro reprezentaci informace o obrazovém elementu použito *b* bitů, je počet úrovní jasu $k = 2^b$. Obvykle se používá 8 bitů na obrazový element, někdy 6, jindy postačí 4bity.
Největším problémem při kvantizaci je **vznik falešných obrysů** u obrazů kvantovaných do nedostatečného počtu jasových úrovní. Tento jev se stane člověku patrný při 50 a méně úrovní jasu a to proto, že lidský zrak je schopen rozlišit cca až 50 úrovní jasu v monochromatickém obraze.

Zlepšení*:* použití nelineárního kvantování, které zvětšuje rozsah těch intervalů jasu, jejichž zastoupení není v obraze pravděpodobné, ale v praxi se používá zřídka.

1.2 Reprezentace digitálního obrazu

Za předpokladu, že obrazová funkce *f(x,y)* byla vzorkována a výsledkem je digitální obraz, který má *M* řádků a *N* sloupců. To znamená, že souřadnice *(x,y)* teď budou diskrétní. Vzhledem k zavedeným konvencím budeme používat celočíselné hodnoty souřadnic.



Obr. 1.3 Reprezentace digitálního obrazu [1].

Jak je patrno z obrázku Obr.1.3 odpovídá jednomu vzorkovacímu bodu v digitalizovaném obraze jeden obrazový element, tzv. **pixel**. Po uspořádání do vzorkovací mřížky pokrývají pixely celý digitalizovaný obraz.

1.2.1 Prostorové rozlišení obrazu

Dvojrozměrné prostorové rozlišení obrazuje je svázáno se vzorkováním, resp. se vzdáleností mezi nejbližšími vzorkovacími body. Ukázka prostorového rozlišení je na obrázcích Obr. 1.4, Obr. 1.5. Na Obr. 1.5 B je vidět velmi jemný šachovnicový vzorek, který způsobuje větší zrnitost obrázku. Tento jev se mnohem více projevuje pro obrázky s prostorovým rozlišením 64x64 a 32x32, viz. Obr. 1.5 C,D.


256


128


64


32

Obr. 1.4 Prostorové rozlišení. 8-bitový obrázek o velikosti 256x256 převzorkován až do velikosti 32x32. [3]

A)

B)

C)

D)

Obr. 1.5 Převzorkované obrázky z Obr.4 na velikost 256x256. A) 256x256, B) 128x128, C) 64x64, D) 32x32. [3]

1.3.1 Jasové rozlišení

Jasové rozlišení se vztahuje k nejmenší rozpoznatelné změně v šedé úrovni v obrázku. Ukázka jasového rozlišení pro různý počet úrovní šedi je na obrázku Obr. 1.6

A)

B)

C)

D)

Obr. 1.6 Jasové rozlišení. A) 128 úrovní šedi. B) 16 úrovní šedi. C) 4 úrovně šedi. D) 2 úrovně šedi [3]

1.3 Vlastnosti digitálního obrazu

Statický monochromatický obraz je popsán obrazovou funkcí *f(x, y)* jejímiž argumenty jsou dvě souřadnice v rovině.
Multispektrální barevný obraz je obraz obsahující více spektrálních pásem. Každé dvojici plošných souřadnic *(x, y)* odpovídá vektor hodnot, např. jasů pro jednotlivé barevné složky obrazu.
Při zpracování obrazů počítačem se pracuje s **digitálními obrazovými funkcemi**, které jsou reprezentované maticemi. Jejich souřadnice i hodnoty jsou celočíselné.

- **Definiční obor obrazu** je rovinná oblast R

$$R = \{(x, y); 1 \le \mathrm{x} \ge \mathrm{x_m};\ 1 \le y \ge y_n\} \qquad (1.1)$$

kde x_m, y_n jsou maximálními souřadnicemi v obrazu.

Obrazová funkce má **omezený obor hodnot**. Toho lze využít při sumaci nebo integraci (při Fourierově a podobných transformacích), kdy lze použít nekonečné meze při zavedení nulovosti hodnot obrazu mimo definiční obor *R*.

- **Orientace souřadnic obrazu**
  - v kartézském smyslu (osa x je vodorovná rostoucí vpravo, osa y svislá rostoucí vzhůru)
  - v maticovém počtu, tj. (řádek, sloupec).

Obor hodnot obrazové funkce (jasu) je také omezený. Podle konvence odpovídá v monochromatickém obrazu nejnižší hodnota černé a nejvyšší bílé. Mezilehlé hodnoty odpovídají různým stupňům šedi.

- **Kvalita digitálního obrazu** je úměrná

Plošnému
Spektrálnímu
Radiometrickému
Časovému
} rozlišení

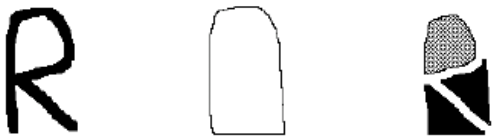
Plošné rozlišení je dáno vzdáleností vzorkovacích bodů, což po vytvoření fourierovského spektra odpovídá také **spektrálnímu rozlišení**.

Radiometrické rozlišení odpovídá počtu kvantizačních úrovní, tj. intuitivně řečeno počtu hodnot jasu. U časově proměnných obrazů mluvíme ještě o **časovém rozlišení** daným časovým intervalem mezi sejmutími dvou následujících obrazů.

1.3 Další pojmy

Hranice oblasti – hranice definičního oboru R je množina všech obrazových elementů oblasti, z nichž každý má alespoň jednoho souseda, který nepatří do oblasti R. Hranice oblasti odpovídá intuitivní představě bodů na samém okraji oblasti. Pro popis tvaru objektů (oblastí) se také používá konvexní obal [2].

Konvexní obal objektu – je nejmenší oblast obsahující objekt taková, že každé dva body oblasti mohou být spojeny úsečkou, jejíž všechny body patří do oblasti. Ukažme si konvexní obal na příkladě objektu, jehož tvar připomíná písmeno R, viz Obr. 1.7 a). Pro názornost si představme, že na objekt natáhneme gumičku. Konvexní obal potom bude oblast, jejíž hranice bude dána gumičkou, viz Obr. 1.7 b) [2].



a) b) c)

Obr. 1.7 Popis pomocí topologických složek: objekt tvaru písmene R, jeho konvexní obal, jezera a zálivy [2].

Množině bodů uvnitř konvexního obalu, které objektu nepatří, se říká deficit konvexnosti objektu. Existují dva druhy deficitu konvexnosti. První tzv. jezera (na Obr. 1.7 c) šrafováno) jsou plně ohraničena objektem. Druhé tzv. zálivy (na Obr. 1.7 c) vyplněno) leží mezi konvexním obalem a objektem. V některých aplikacích se objekty popisují pomocí konvexního obalu, jezer a zálivů.

Topologie – matematická disciplína, která dovoluje studovat vlastnosti, které se neopírají o pojem vzdálenosti.

Vlastnosti probírané dosud v této sekci se opírají o pojem oblasti, což předpokládá, že víme, které pixely obrazu k oblasti přiřadit. Jinak řečeno, očekává se, že umíme obraz interpretovat (neboli **segmentovat**).

Na počátku zpracování ale obvykle obraz segmentovat neumíme a obrazová funkce *f(x, y)* představuje dvojrozměrný signál. Hledají se takové nástroje, které ho umějí popsat bez nutnosti interpretace. K tomu se často používají **lokální operace** využívající jen bezprostřední okolí právě zpracovávaného pixelu. Významnou lokální informací o obrazu lze získat pomocí **hran**, které udávají, jak se lokálně mění obrazová funkce. V matematice o funkci analogicky vypovídá její první derivace (gradient).

Hrana – je vlastností obrazového elementu a jeho bezprostředního okolí. Odpovídá gradientu obrazové funkce *f(x, y)*. Jelikož je funkce *f(x, y)* dvojrozměrná, je hrana určena velikostí a směrem.

- Velikost hrany odpovídá modulu gradientu spojité obrazové funkce v příslušném pixelu.
- Směr hrany je kolmý na směr gradientu, který míří ve směru největšího růstu obrazové funkce.

Shrnutí pojmů

Digitalizace obrazu je proces převodu analogového signálu do digitální podoby. Zahrnuje vzorkování, kvantování a kódování obrazu.
Vzorkování je diskretizace souřadnic obrazu, tzn. vytvoření obrazové matice s *M x N* body. Hovoříme o tzv. prostorovém rozlišení.
Kvantování představuje diskretizaci jasových úrovní, tzn. rozdělení spojité jasové úrovně každého obrázku do K intervalů. Hovoříme o jasovém rozlišení.
Monochromatický obraz je popsán obrazovou funkcí *f(x, y)*. U **multispektrálního barevného obrazu** každé dvojici plošných souřadnic *(x, y)* odpovídá vektor hodnot, např. jasù pro jednotlivé barevné složky obrazu.
Definičním oborem obrazu je rovinná oblast *R*. Kde $R = \{(x, y); 1 \leq x \geq x_m; 1 \leq y \geq y_n\}$ a kde x_m, y_n jsou maximálními souřadnicemi v obrazu. Obrazová funkce má **omezený obor hodnot**. Toho lze využít při sumaci nebo integraci (při Fourierově a podobných transformacích), kdy lze použít nekonečné meze při předpokladu nulovosti hodnot obrazu mimo definiční obor *R*.
Hrana je vlastností obrazového elementu a jeho bezprostředního okolí. Odpovídá gradientu obrazové funkce *f(x, y)*. Hrana je určená velikostí, což je modul gradientu v příslušném pixelu, a směrem jež je kolmý na směr gradientu a míří ve směru nevětšího růstu obrazové funkce.

Otázky

1. Vysvětlete proces digitalizace.
2. Jestliže pro zakódování 1 pixelu v šedém obrázku použijete 8 bitů, kolik úrovní šedi můžete v obraze rozlišit?
3. Co je to jasové a prostorové rozlišení obrazu?

CVIČENÍ

Zpracování obrazu s využitím SW systému MATLAB

Náplní prvního cvičení je seznámit čtenáře s **Video and Image Processing Blockset™** 3 ve vývojovém prostředí **Matlab2010a**. Představíme základní funkční bloky, které slouží pro práci s obrazovými daty. Rozebereme základní operace s obrazy. Seznámíme se s principem načtení a zobrazení dat různých obrazových a video formátů. Ukážeme si přidání textové informace, import a export dat z a do workspace v Matlabu.

Video and Image Processing Blockset

Video and Image Processing Blockset (VIPB) je speciální nástavba pro Matlab/Simulink, která poskytuje algoritmy a nástroje pro návrh a simulaci zpracování obrazu a videa. Ve spojení s Image Acquisition Toolbox umožňuje navrhnuté algoritmy implementovat do reálného času, tzn. po připojení zdroje obrazu nebo videa zpracovávat danou úlohu v reálném čase. Pomocí speciálních funkcí je možno zpracovávat a řešit úlohy týkající se např. odstranění šumu, úpravu kontrastu, detekce hran, detekce pohybu, rozpoznávání objektů, stabilizace obrazu apod.

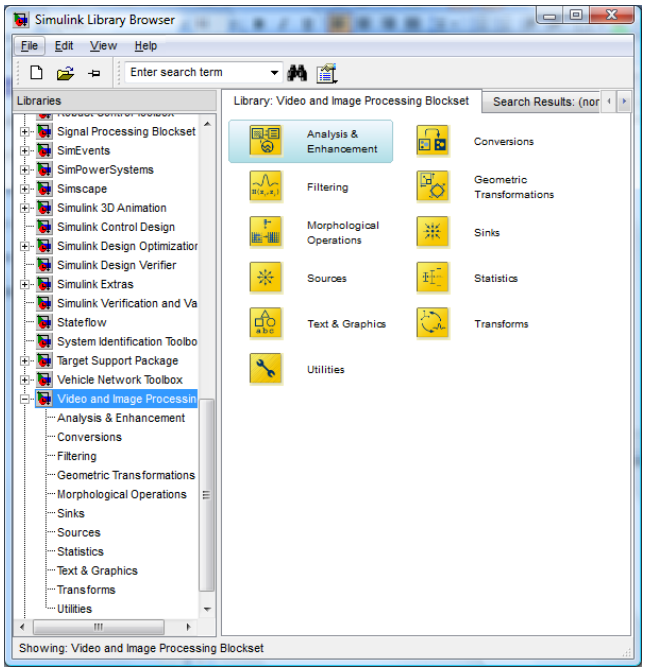
Mezi klíčové bloky a nástroje Image and Video Processing Toolboxu patří:

- Systémové předdefinované bloky pro použití v Matlabu a bloky pro použití v Simulinku.
- Bloky pro zpracování videa včetně deinterlacingu, stabilizace videa apod.
- Bloky pro zpracování obrazu včetně filtrování, geometrické transformace, detekce hran, segmentace apod.
- Bloky pro import a export multimediálních I/O dat.
- Programové prostředky pro automatické generování C-kódu.
- aj.

V úvodním cvičení se seznámíme s jednotlivými bloky v Matlab/Simulink a VIPB použitými v dalším textu pro z pracování obrazových dat.

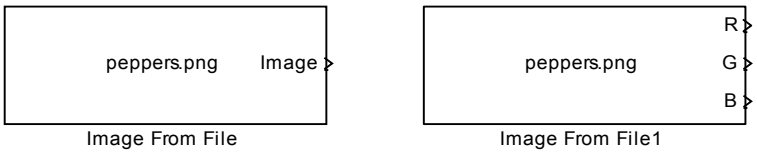
Načtení obrazu a zobrazení

Spustíme Matlab2010a. Nový model otevřeme pomocí ikony Simulink. Otevře se Simulink Library Browser. Vybereme ***File -> New -> Model***. Viz. Obr. 1.8



Obr. 1.8 Nástrojová sada Toolboxu Video and Image processing Blockset.

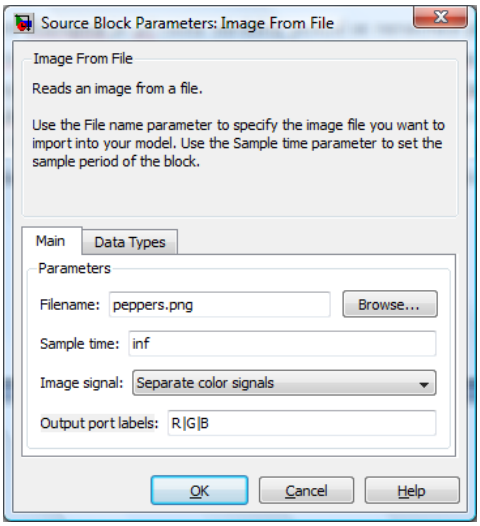
Otevřeme nástrojovou sadu ***Video and Image Processing Blockset***.



Obr. 1.9 Image From File vlevo pro jedno vícerozměrné výstupní pole, vpravo pro matici barevných RGB složek.

Z menu ***Sources*** vybereme blok ***Image from File***, který slouží k importu podporovaných formátů obrazových dat. Blok pracuje s maticí o velikosti $M \times N$, M a N jsou počty řádků a sloupců matice. Je-li obraz zadán ve formátu $M \times N$, tak je výstupem monochromatický obraz. Přidáním barevné složky P vznikne trojrozměrná matice ve tvaru $M \times N \times P$, jejímž výstupem je barevný obrázek.
Nastavení vstupních parametrů bloku ***Image from File*** provedeme otevření dialogového okna (dvojitý klik levým tlačítkem myši).

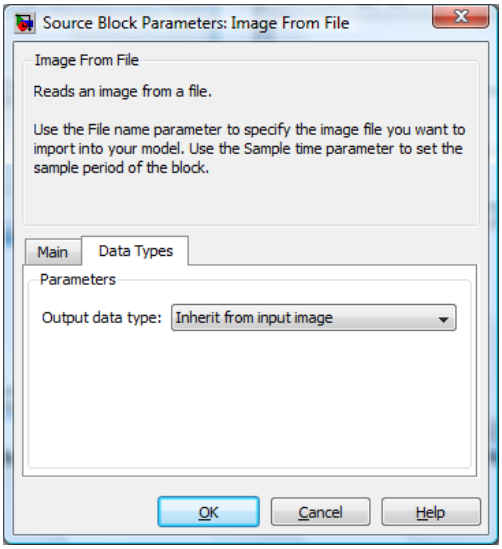
Nastavení vstupních parametrů provedeme pomocí záložky ***Main***.



Obr. 1.10 Nastavení vstupních parametrů bloku Image From File.

- Parametr ***Filename*** je url cesta obrázku, pokud se nenachází přímo v základním adresáři Matlabu. Tlačítko Browse slouží k nastavení jiné lokace obrázku.
- Parametr ***Sampletime***, implicitně nastavený na hodnotu ***inf***, definuje vzorkovaní a periodu výstupního signálu.
- Parametr ***Image signal*** definuje typ výstupní proměnné. ***One multidimesional signal*** zobrazuje výstup jako jedno vícerozměrné pole. Výstupem je matice $M \times N \times P$. Pro separaci jednotlivých barevných složek slouží ***Separate Color Signals***. U této volby je možné definovat ***Output port labels*** s názvy jednotlivých RGB složek. Výstupem je matice $M \times N$

Nastavení výstupních parametrů provedeme pomocí záložky ***Data Types***.



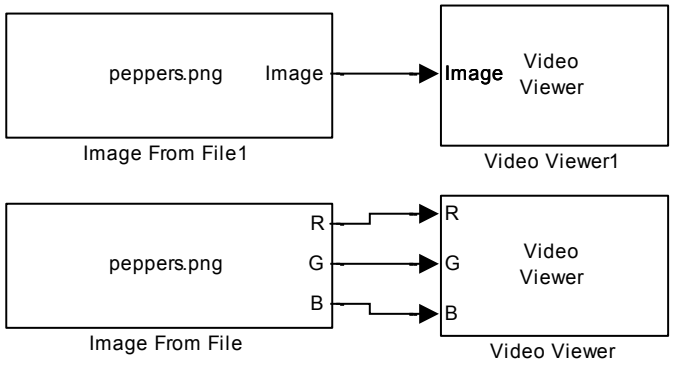
Obr. 1.11 Nastavení výstupních parametrů bloku Image From File.

- Parametr ***Output data type*** definuje typ výstupní složky. V tomto cvičení využijeme typ ***Inherent from input image***, tedy inherentní k vlastnímu vstupu.

Zobrazení obrazu

Z menu ***Sinks*** vybereme blok ***Video Viewer*** , který umožňuje zobrazit binární nebo RGB obrázek či video. Blok zajišťuje přehrávání, krokování a pauzování videa při běhu. Blok také poskytuje nástroje pro analýzu jednotlivých pixelů. Tento blok neprovádí žádné výpočty, pouze zobrazuje data v Real-Time.

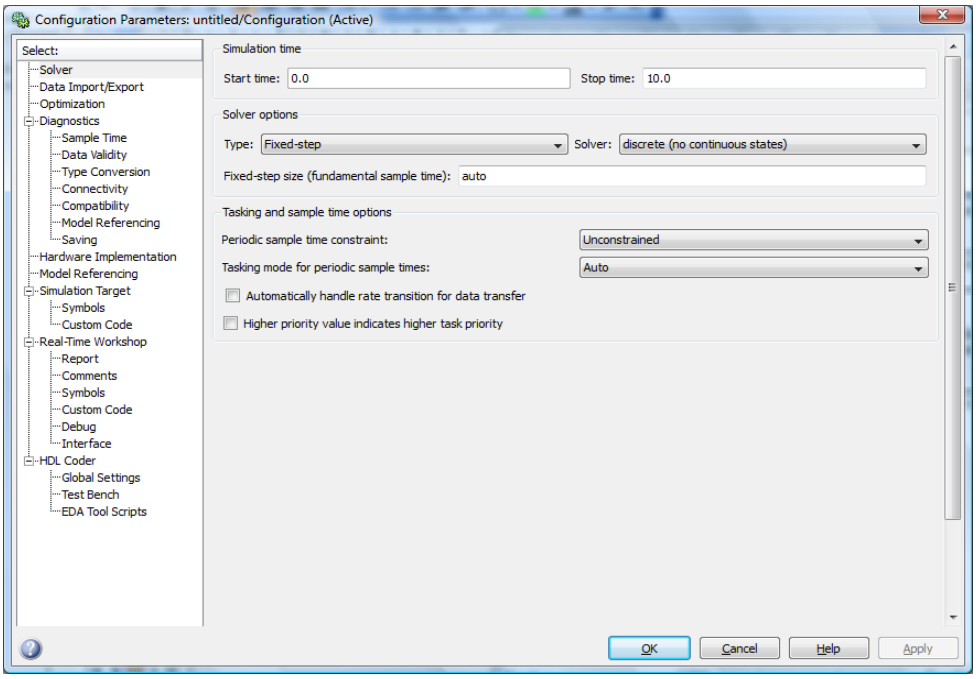
Nastavení vstupních parametrů bloku ***Video Viewer*** provedeme otevření dialogového okna (dvojitý klik levým tlačítkem myši).



Obr. 1.12 Načtení a zobrazení obrázku pomocí Videoimage ang Processing Blockset. Horní obrázek pro jedno multidimenzionální pole. Spodní obrázek pro matice RGB složek.

Nastavení parametrů Simulinku

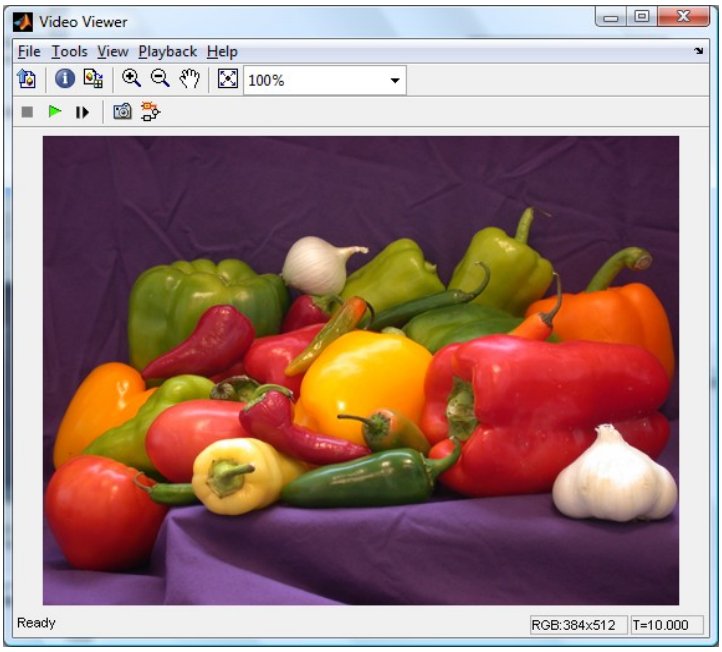
Video and Image Processing Blockset počítá hodnoty přímo místo pomocí diferenciálních rovnic. Pro zefektivnění operace je zapotřebí nastavit Simulink Solver jako časovač. Ten využívá Sample Time každého bloku k determinaci času provedení jednotlivých kódů. Zkrátí se tím časová odezva soustavy. Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***.



Obr. 1.13 Nastavení Configuračních paremetrů.

Pro zrychlení odezvy vyberte ze seznamu pro parametr ***Type*** hodnotu ***Fixed-step*** a pro parametr ***Solve*** vyberte hodnotu ***Discrete (no continuous states)***. Parametr Stop Time nastavte na hodnotu INF u video vstupů jak z pevného disku, tak živě. Hodnota Stop Time = 0 se používá u statických obrázků.

Nyní můžete model spustit (spustit simulaci) pomocí ikony ***Start Simulation***.



Obr. 1.14 Dialogové okno Video Viewer zobrazuje načtený obrázek.

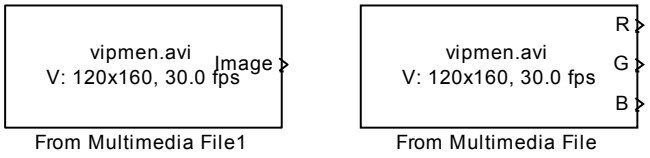
Import AVI souborů

Podporovány jsou tyto formáty :

- Video formáty: `.qt`, `.mov`, `.avi`, `.asf`, `.asx`, `.wmv`, `.mpg`, `.mpeg`, `.mp2`, `.mp4`
- Audio formáty: `.wav`, `.wma`, `.aif`, `.aifc`, `.aiff`, `.mp3`, `.au`, `.snd`

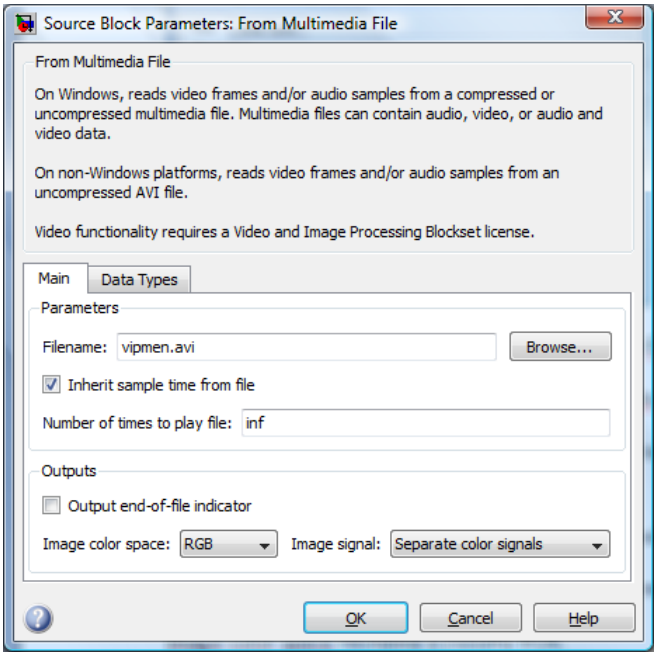
Import dat z pevného disku

Z menu ***Sources*** vybereme blok ***From Multimedia File***. Tento blok importuje multimediální data ze souboru do modelu Simulink. Nastavení vstupních parametrů bloku ***From Multimedia File*** provedeme otevřením dialogového okna (dvojitý klik levým tlačítkem myši). Jako vstupní data použijeme demo video z Matlabu s názvem *vipmen.avi*.



Obr. 1.15 Blok From Multimedia File vlevo pro vícerozměrné výstupní pole, vpravo pro výstupní signál rozdělený na složky RGB.

Nastavení vstupních parametrů provedeme pomocí záložky ***Main***.

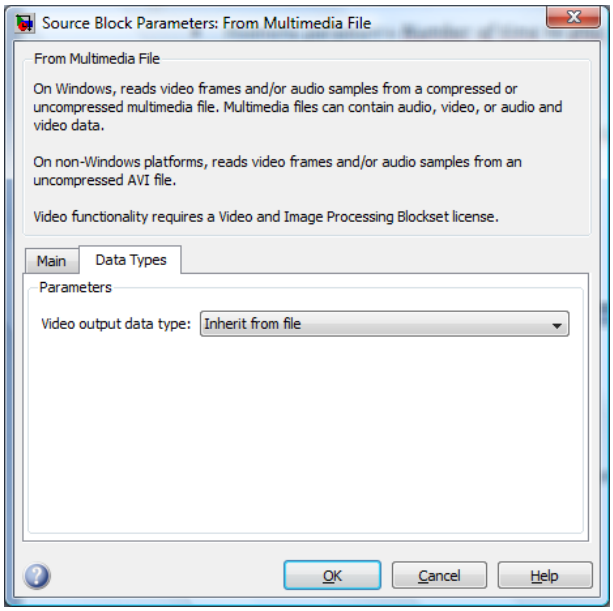


Obr. 1.16 Nastavení vstupních parametrů bloku From Multimedia File.

- Parametr ***File name*** specifikuje název multimediálního obrazu. Pomocí tlačítka ***Browse*** se provede výběr konkrétního souboru.
- Označení ***Inherit sample time from file*** zajistí zachování parametrů vstupního obrazu.

- Hodnotu parametru ***Number of time to play file*** ponecháme na ***inf***.
- ***Image color space*** necháme v rozsahu ***RGB***.
- ***Image signal*** definujeme na ***Separate color signals***.

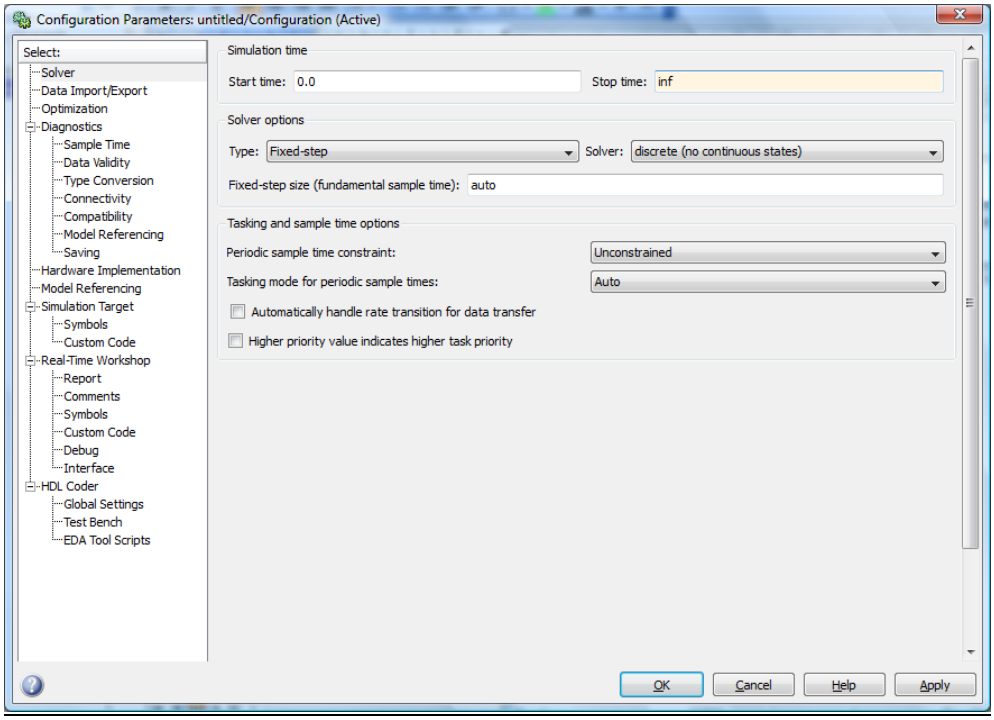
Nastavení výstupních parametrů provedeme pomocí záložky ***Data Types***.



Obr. 1.17 Nastavení výstupních parametrů bloku From Multimedia File.

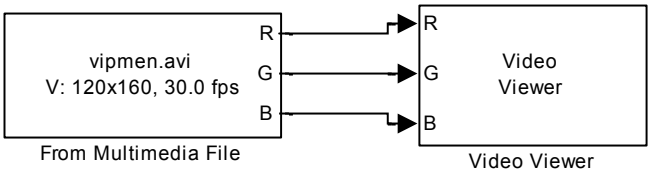
- Parametr ***Video output data type*** nastavíme na hodnotu ***Inherit from file***.

Nastavení parametrů Simulinku
Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***.



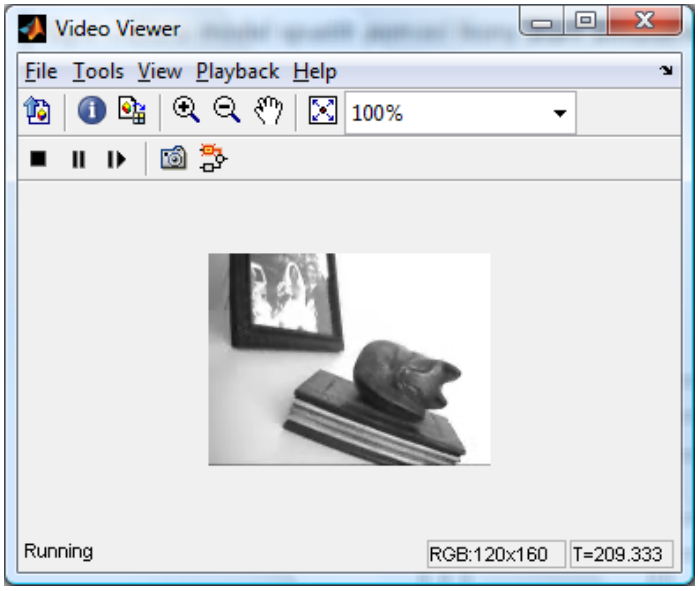
Obr. 1.18 Nastavení parametrů Simulinku pro opakované přehrávání videa a rychlejší zpracování.

- Parametr ***Stop time*** nastavíme na hodnotu ***inf***, čímž dosáhneme nekonečného opakování.
- Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.***
- Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***



Obr. 1.19 Blokové schéma pro přehrávání video záznamů.

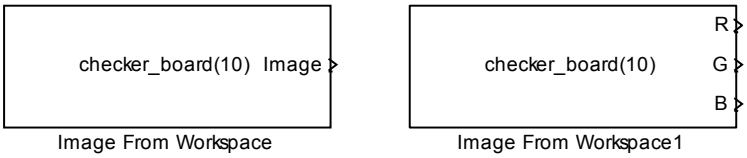
Nyní můžete spustit simulaci pomocí ikony ***Start Simulation***.



Obr. 1.20 Výsledek zobrazení avi souboru.

Import dat z Workspace

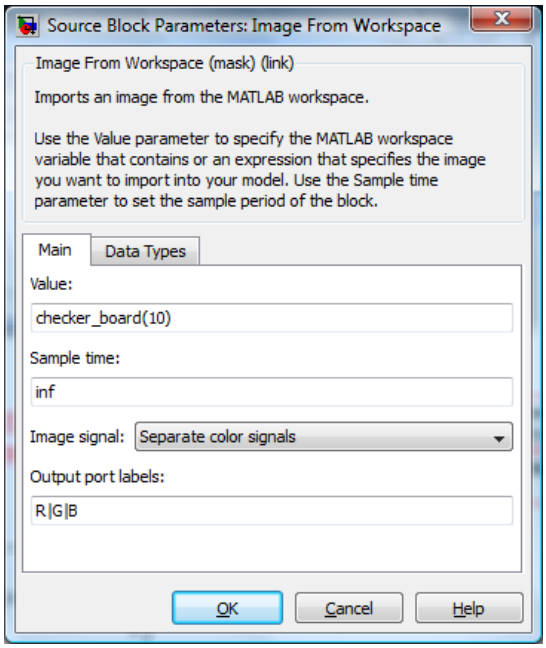
Z menu ***Sources*** vybereme blok ***Image From Workspace***. Tento blok importuje obrazová data z workspace Matlabu. Nastavení vstupních parametrů bloku ***Image From Workspace*** provedeme otevřením dialogového okna (dvojitý klik levým tlačítkem myši).



Obr. 1.21 Blok Image From Workspace vlevo pro vícerozměrné výstupní pole, vpravo pro výstupní signál rozdělený na složky RGB.

Nastavení vstupních parametrů provedeme pomocí záložky ***Main***.

- Parametr Value je název proměnné, ve které jsou vstupní obrazová data uložena. Je-li hodnota Value = checker_board(10), tak je zdrojem 10 prvek z proměnné „checker_board“.



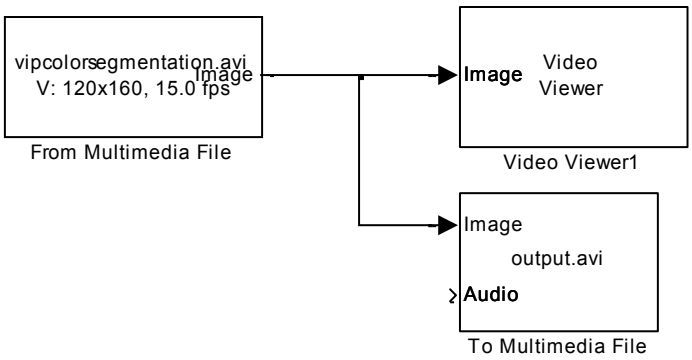
Obr. 1.22 Nastavení parametrů záložky Main bloku Image From Workspace.

Nastavení výstupních parametrů provedeme pomocí záložky ***Data Types***. Nastavení je totožné s nastavením pro import dat z pevného disku

Konverze obrazu a export souborů

Export dat na pevný disk

Konverze a export videa se provádí pomocí bloku To Multimedia File.

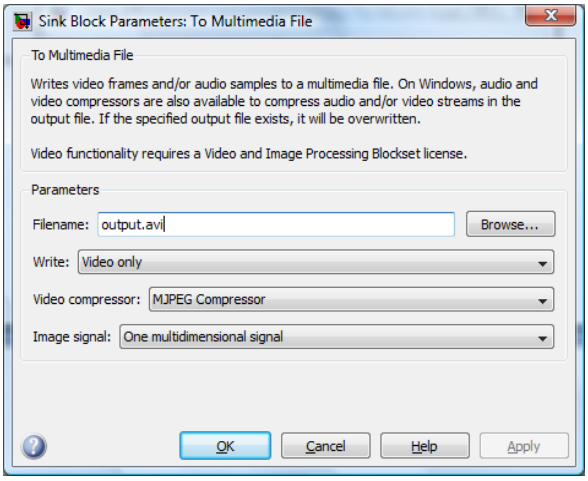


Obr. 1.23 Schéma modelu pro konverzi a export videa.

Jednotlivé bloky naleznete:

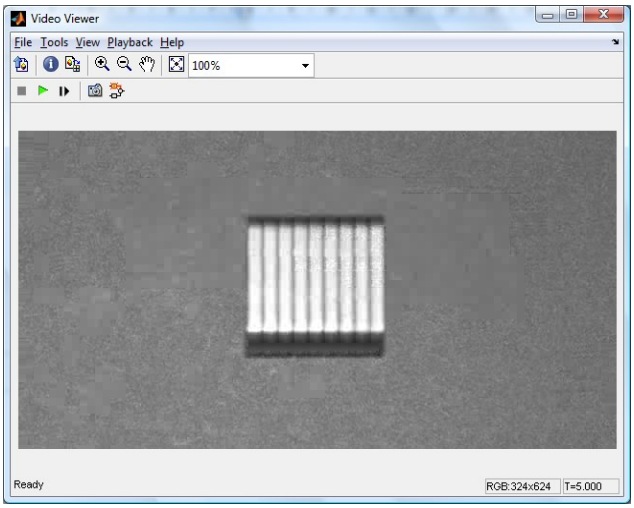
- Blok **To Multimedia File** nalezneme v menu ***Video and Image processing Blockset -> Sinks***
- Blok **Image Video Viewer** viz. předchozí text.
- Blok **From Multimedia File** viz. předchozí text.

Nastavení vstupních parametrů bloku **To Multimedia File** provedeme dle Obr.1.24. V jednotlivých záložkách provedeme následující nastavení. Parametr Filename definuje vstupní data, zvolíme tedy cestu zpracovávaného videa. Parametr Write nastavíme na hodnotu Video only. Parametr Video compressor nastavíme na hodnotu MJPEG Compressor a parametr Image signal ponecháme defaultně přednastaven na hodnotu One multidimensional signal.



Obr. 1.24 Nastavení parametrů bloku To Multimedia File.

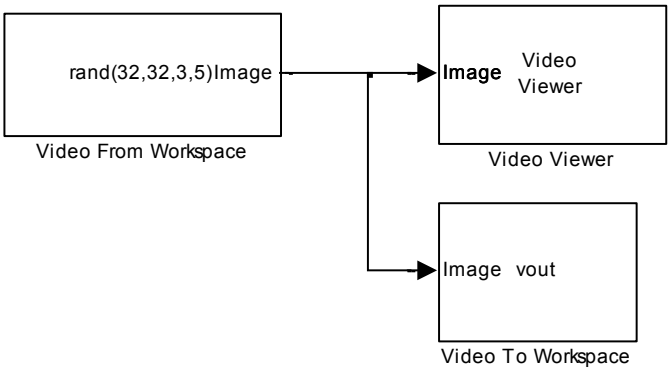
Proveďte nastavení jednotlivých bloků, které propojte dle schéma viz. Obr. 1.23 a spusťte simulaci.



Obr. 1.25 Konverze obrazu do formátu MPEG.

Export dat do Workspace

Konverze a export videa se provádí pomocí bloku Video To Workspace.

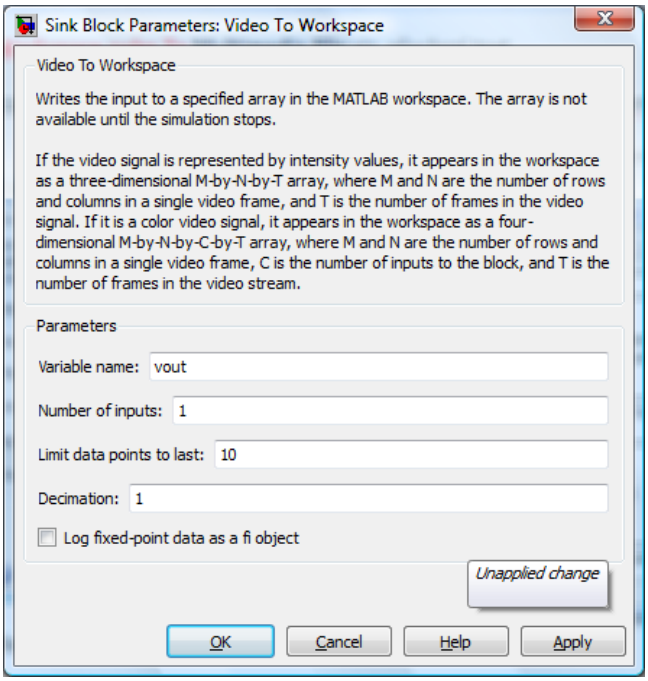


Obr. 1.26 Schéma modelu pro konverzi a export videa do Workspace.

Jednotlivé bloky naleznete:

- Blok Video **To Workspace** nalezneme v menu ***Video and Image processing Blockset -> Sinks***
- Blok **Image Video Viewer** viz. předchozí text.
- Blok Video **From Workspace nebo To Multimedia File** viz. předchozí text.

Nastavení vstupních parametrů bloku **Video To Workspace** provedeme dle Obr.1.27**.** Parametr Variable name označíme vout, parametr Number of inputs nastavíme na hodnotu 1. Parametr Limit data points to last nastavíme na hodnotu 10. Parametr Decimation ponecháme defaultně nastavený na hodnotu 1. Check box Log fixed-point data as a fi object ponecháme nezaškrtnutý.

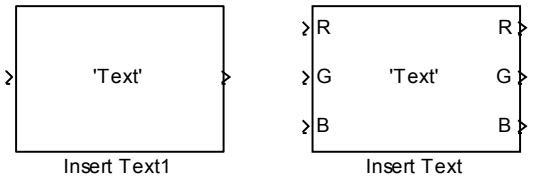


Obr. 1.27 Nastavení parametrů bloku Video To Workspace.

Proveďte nastavení jednotlivých bloků, které propojte dle schéma viz.Obr. 1.276 a spusťte simulaci.

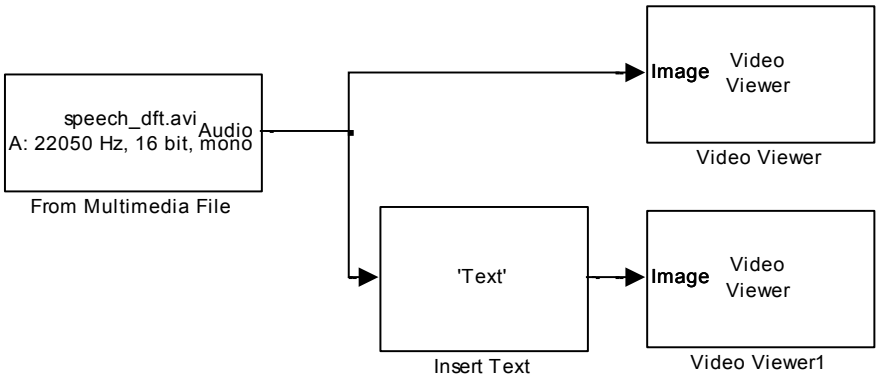
Přidání textu do obrazu

Přidání textu provedeme pomocí bloku Insert Text. Tento blok přidá do prohlíženého videa dynamický nebo statický text dle nastavení.



Obr. 1.28 Blok Insert text, vlevo pro vícerozměrné výstupní pole, vpravo pro výstupní signál rozdělený na složky RGB.

Vložení textu do videa provedeme pomocí schématu dle Obr. 1.29. Pro ukázku v našem případě ponecháme nastavení pro vícerozměrné pole.



Obr. 1.29 Blokové schéma pro vložení textu do zdrojového souboru.

Jednotlivé bloky naleznete:

Blok **Insert text** nalezneme v menu ***Video and Image processing Blockset*** -> ***Graphics & text***

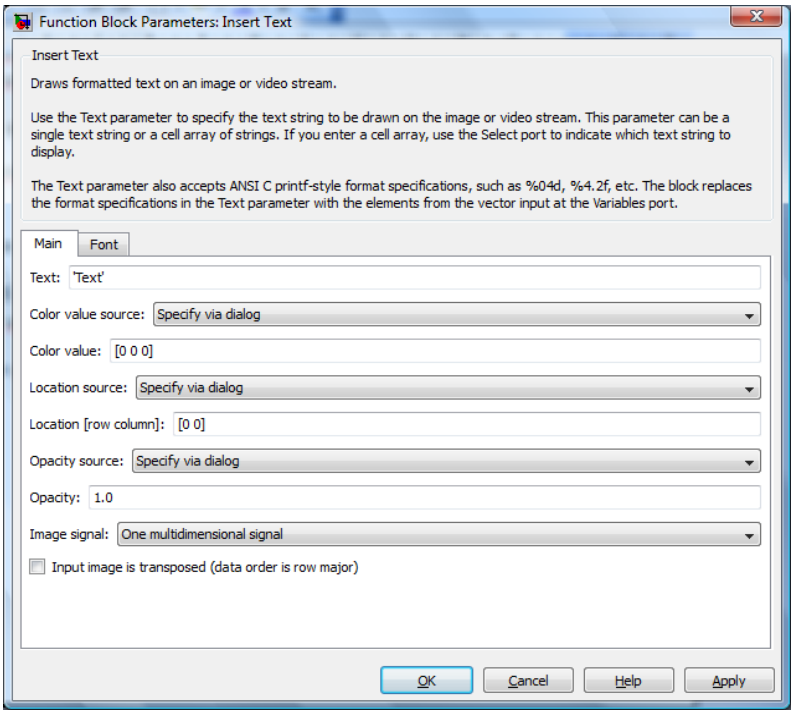
Bloky From Multimedia File a Video Viewer nalezneme viz. cvičení 1.

- Nastavení ***Image From File*** pro tuto aplikaci je následující. Sample time = inf, Image signal = separate color signals, Output port labels = RGB, Output data type = double.

- ***V bloku Video Viewer*** nastavíme parametr Image signal na hodnotu Saparate color signals.

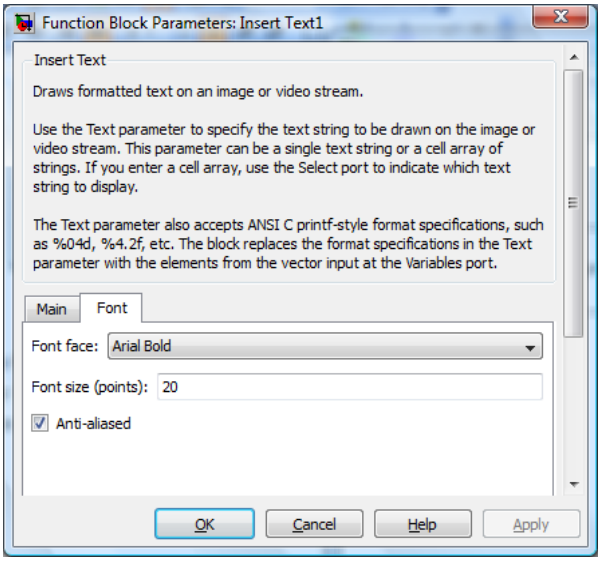
Nastavení vstupních parametrů

- V bloku **Insert text** provedeme nastavení parametrů dle Obr.1.30. **V záložce Main** nastavíme parametr Text = 'Moje video', parametr Location = [0 0]. Tento parametr slouží k identifikaci umístění textu v obraze. Počátek souřadného systému je definován maticí [0,0], fakticky je umístěn v levém horním rohu. Parametr Color value určuje barvou hodnotu písma, kdy souřadnice [0 0 0] odpovídají hodnotě [R G B].



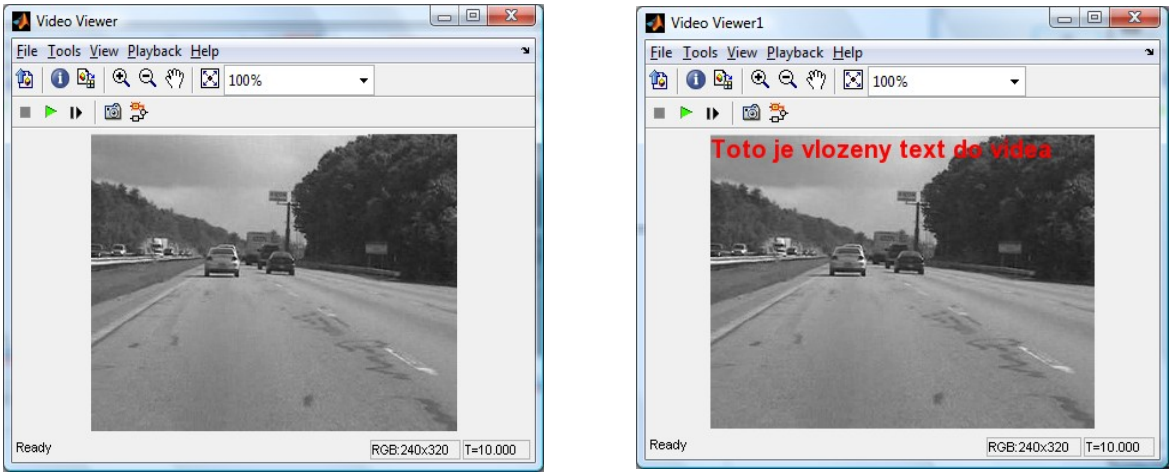
Obr. 1.30 Nastavení bloku Insert Text v záložky main.

- V záložce **Font** nastavíme parametr **Font face** , který udává typ písma, na hodnotu Arial Bold. Parametr Font Size (point) udává velikost písma, nastavíme jej na hodnotu 20. Zaškrtneme check box Anti-aliased pro potlačení alliasingu na hranách písma.



Obr. 1.31 Nastavení záložky Font bloku Insert Text.

Proveďte nastavení jednotlivých bloků, které propojte dle schéma viz. Obr. 1.29 a spusťte simulaci.



Obr. 1.282 Vložený text do videosnímku, vlevo původní snímek vpravo s vloženým textem.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 01 Uvod. swf.

2. KOREKCE OBRAZU A HISTOGRAM

Čas ke studiu: 2 hodiny

Cíl: Po prostudování tohoto odstavce budete umět

- popsat základní vlastnosti histogramu.
- definovat základní barevné rozlišení.
- vyřešit úlohu v Matlab/Simulinku a VIP pro korekci jasové složky a kontrastu v obraze.

Výklad

2.1 Úvod

V první kapitole bylo uvedeno, jakým způsobem dostaneme obraz do digitální podoby. Tomuto procesu se říká obecně **digitalizace** obrazu. Po převodu spojité obrazové informace (to co vidíme okem) do její digitální podoby (matice pixelů) následují většinou procedury zahrnující metody zpracování obrazu, tzn. cílem je analýza dat obsažených v obraze za účelem získání určité informace. Aby byl výsledek kvalitativně na vysoké úrovni, je třeba mít i zdrojová data (v našem případě zdrojový obrázek) v co nejlepší kvalitě. Toho nelze vždy s úspěchem dosáhnout. Při digitalizaci obrazu mohou nastat různé rušivé vlivy, které nám zdrojový obraz znehodnotí. Proto je nutno ještě před vlastní analýzou dané úlohy týkající se zpracování obrazu zjistit prvotní základní informace o digitalizovaném obrazu. K tomuto prvotnímu získání informací o obraze nám slouží (stejně jako v oblasti statistického zpracování dat) **histogram**.

2.2 Histogram

Histogram nám udává grafickou informaci o rozložení jednotlivých jasových složek v obraze.

Histogram digitálního obrazu s hodnotami jasu v rozsahu $\langle 0, L-1 \rangle$ je diskrétní funkcí

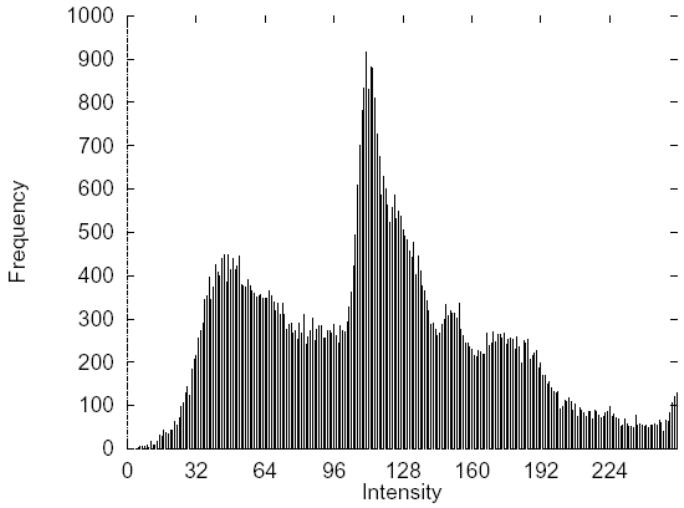
$$h(r_k) = n_k \qquad (2.1)$$

kde r_k je k-tá úroveň šedi a n_k je počet pixelů v obraze s úrovní šedi r_k.
V normalizovaném tvaru jsou jednotlivé složky poděleny celkovým počtem pixelů n v obraze. Normalizovaný tvar histogramu je určen rovnicí:

$$p(r_k) = \frac{n_k}{n} \qquad (2.2)$$

pro $k = 0, 1, \ldots, L-1$. Kde $p(r_k)$ udává pravděpodobnost výskytu úrovně šedi r_k.

Součet všech prvků v normalizovaném histogramu je roven 1. Potom lze histogram chápat jako odhad hustoty pravděpodobnosti rozdělení jasu v obraze. Histogram je zobrazen ve sloupcovém grafu, kde na vodorovnou osu jsou vynášeny diskrétní hodnoty jasových úrovní a na svislou osu hodnoty $h(r_k)$ nebo $p(r_k)$. Příklad histogramu je uveden na Obr. 2.1.



Obr. 2.1 Histogram jasu pro L = 256 jasových úrovní [3].

Histogram velmi často bývá jedinou globální informací o obraze. Můžeme ho použít při nastavování podmínek pro snímání a digitalizaci, při změnách jasové stupnice a při segmentaci obrazu na objekty a okolí. **Určitému obrazu je přiřazen jeden histogram. Jednomu histogramu však může odpovídat několik obrazů.** Například při změně polohy objektu na pozadí o konstantním jasu se histogram nezmění.

2.3 Ekvalizace histogramu

Ekvalizací histogramu rozumíme vyrovnání histogramu. Pro vysvětlení principu budeme pracovat se spojitou funkcí. Proměnné je zapotřebí normovat v intervalu $\langle 0,1 \rangle$ kde hodnota $r = 0$ odpovídá černé barvě a hodnota $r = 1$ odpovídá barvě bílé. Pro všechna $r = \langle 0,1 \rangle$ hledáme transformaci ve tvaru: $s = T(r) \quad pro \quad 0 \le r \le 1$ (2.3)

Transformační funkce musí splňovat tzn. podmínky jednoznačnosti:

a) $T(r)$ je jednoznačně a monotónně rostoucí v intervalu $0 \le r \le 1$.
b) $0 \le T(r) \le 1$ pro $0 \le r \le 1$.

Podmínka jednoznačnosti $T(r)$ zajišťuje existenci zpětné transformace a je tedy podmínkou nutnou. Podmínka monotónnosti zajišťuje uspořádání hodnot výstupního obrazu. Uspořádání výstupního hodnot bude vzestupně od $r = 0$ po $r = 1$, tedy od černé barvy k bílé. Zpětnou transformaci provedeme pomocí rovnice:

$$r = T^{-1}(s) \qquad (2.4)$$

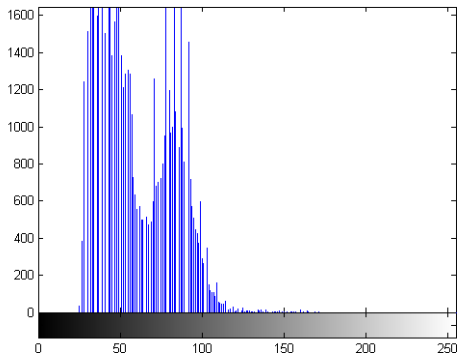
Na jasové úrovně v intervalu $r = \langle 0,1 \rangle$ lze pohlížet jako na statisticky náhodnou veličinu, kterou lze popsat pomocí funkce hustoty pravděpodobnosti. Nechť $p_r(r)$ a $p_s(s)$ jsou funkce hustoty pravděpodobnosti náhodných veličin r a s kde p_r a p_s jsou různé funkce. Dále nechť $p_r(r)$ a $T(r)$ jsou známy a $T^{-1}(s)$ splňuje podmínku (a), pak funkce hustoty pravděpodobnosti $p_s(s)$ transformované proměnné s můžeme získat užitím vztahu:

$$p_s(s) = p_r(r) \cdot \left| \frac{dr}{ds} \right| \qquad (2.5)$$

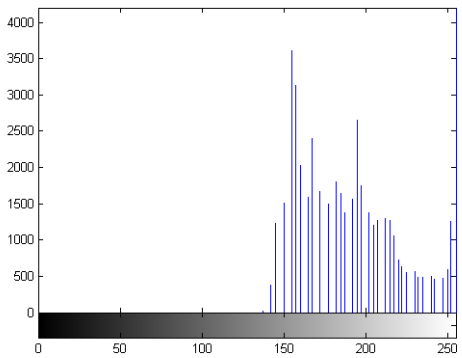
Hodnotu $p_s(s)$, funkce hustoty pravděpodobnosti proměnné s, získáme užitím vybrané transformační funkce na jasovou úroveň funkce hustoty pravděpodobnosti vstupního obrazu $p_r(r)$.

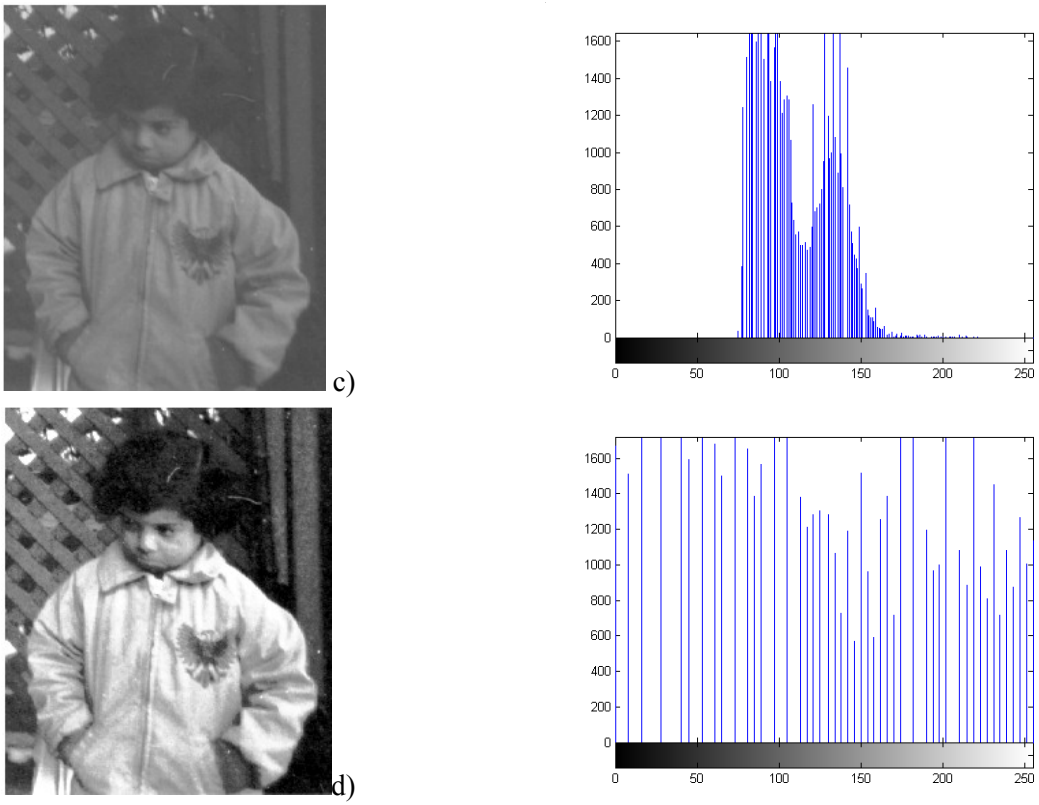
Na Obr. 2.2 jsou ukázány čtyři základní typy obrázků a jejich histogramy. Všimněte si, že v tmavém obrázku jsou komponenty histogramu koncentrovány v dolní (tmavé) části rozsahu úrovní šedi. Naproti tomu komponenty histogramu světlého obrázku jsou koncentrovány v horní části rozsahu úrovní šedi. Obrázek s nízkým kontrastem bude mít úzký histogram a bude umístěn zhruba uprostřed rozsahu úrovně šedi. Naproti tomu komponenty histogramu obrázku s vysokým kontrastem pokrývají celý rozsah úrovní šedi a navíc distribuce pixelů je velmi blízká rovnoměrnému rozložení.

a)



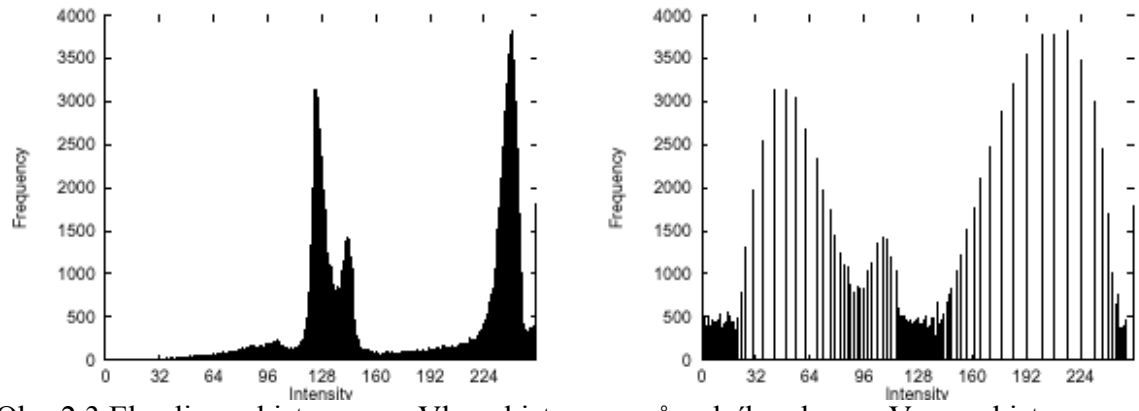
b)



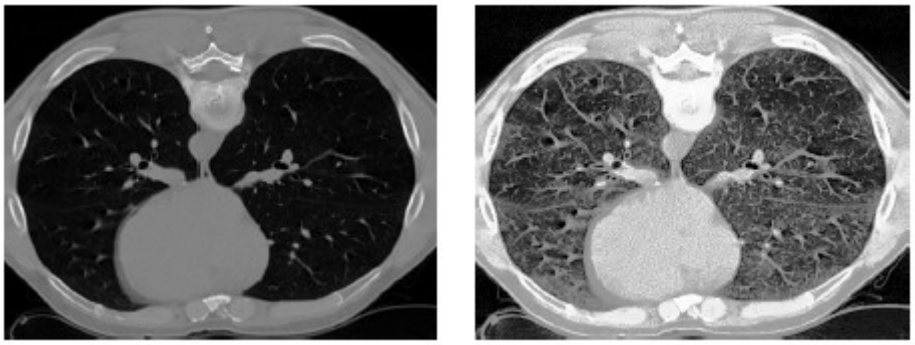


Obr. 2.2 Čtyři základní typy obrázků, tmavý (a), světlý (b), malý kontrast (c), velký kontrast (d) [3]

Ekvalizace histogramu = vyrovnání histogramu.
Ve vyrovnaném histogramu obrazu po transformaci jasové stupnice jsou jednotlivé jasové úrovně zastoupeny zhruba stejně četně. Ideální by byl histogram, kde by byly všechny četnosti stejné, ale ve světě digitálních obrázků s poměrně málo jasovými úrovněmi (stovky) a mnoha pixely (statisíce) to většinou nejde.
Ekvalizace zvýší kontrast pro úrovně jasu blízko maxim histogramu a sníží kontrast blízko minim histogramu. Příklad histogramu před a po ekvalizaci je uveden na Obr.2.3.



Obr. 2.3 Ekvalizace histogramu: Vlevo histogram původního obrazu. Vpravo histogram výsledného kontrastnějšího obrazu, viz. obr. 2.4 [2],[3]



Obr. 2.4 Ekvalizace histogramu: Vlevo původní obraz (tomografický řez plícemi). Vpravo zvýšení kontrastu po ekvalizaci [2],[3].

Na Obr. 2.4 je uvedena dvojice intenzitních obrazů (jeden řez z rentgenového tomografu napříč hrudníkem zobrazující plíce), jimž odpovídají histogramy z Obr.2.3 Je vidět, že u obrazu vpravo po ekvalizaci se zvýšil kontrast.

Pokud si shrneme vlastnosti histogramu obrazu, pak platí, že:
Histogram (pro šedý obraz)

- ukazuje kolik pixelů připadá na každou úroveň šedi.
- je sloupcový graf, kde výška každého sloupce je počet pixelů dané úrovně šedi.
- záznam podílu světlých a tmavých pixelů v obraze.
- poskytuje pouze obecnou informaci o obraze, tudíž nemůžeme přesně určit jakému obrazu histogram náleží.
- Přestože histogram nemůže podat detailní informaci o objektu v obrázku, může určit některé detailní charakteristiky.

Pokud plocha zobrazená na fotografii je známá, může být v některých případech určena plocha objektu na obrázku, pokud porovnáme počet pixelů připadajících na objekt vzhledem k celkovému počtu pixelů obrázku.

2.4 Barevný obraz

Obraz může být v jednodušším případě monochromatický. Je reprezentován jedinou obrazovou funkcí *f(x, y)*. Ve složitějším případě pracujeme s **barevným (multispektrálním) obrazem**. Každé dvojici plošných souřadnic *(x, y)* odpovídá vektor hodnot, např. jasů pro jednotlivé barevné složky obrazu.

Barva – je vlastnost objektů spojená s jejich schopností odrážet elektromagnetické vlnění různých vlnových délek.

- Při snímání a počítačovém zpracování nelze pracovat se všemi vlnovými délkami obrazu.
- Oblast vlnění zajímavou z hlediska zpracování se snažíme rozdělit do několika spektrálních pásem, která pokrývají požadovaný rozsah vlnových délek.
- Obraz potom snímáme více senzory, z nichž každý je citlivý na poměrně úzkou část spektrálního pásma.
- Výběr vlnových délek pro jednotlivá spektrální pásma závisí na aplikaci.

Barevný signál – se skládá ze tří samostatných spektrálních složek RGB

- červená má vlnovou délku 700 nm,
- zelená má vlnovou délku 546,1 nm,
- modrá má vlnovou délku 435,8 nm.

Smícháním složek vzniká barevný obraz. Každá složka se digitalizuje, přenáší a případně zpracovává počítačem samostatně.
Při vyhodnocení multispektrálních a barevných obrazů je třeba vzít v úvahu dva aspekty: velké množství dat a značnou podobnost mezi odpovídajícími daty ve složkách.

Shrnutí pojmů

Histogram nám udává grafickou informaci o rozložení jednotlivých jasových složek v obraze.
Ekvalizace histogramu je jeho vyrovnání, tzn. zvětšení kontrastu zdrojového obrazu.
Barva – je vlastnost objektů spojená s jejich schopností odrážet elektromagnetické vlnění různých vlnových délek.
Barevný signál – se skládá ze tří samostatných spektrálních složek RGB, červená má vlnovou délku 700 nm, zelená má vlnovou délku 546,1 nm, modrá má vlnovou délku 435,8 nm.

Otázky

1. Co je to histogram obrazu?
2. Vysvětlete postup ekvalizace histogramu a proč se provádí.
3. Jaký je počet úrovní jasu v obraze?

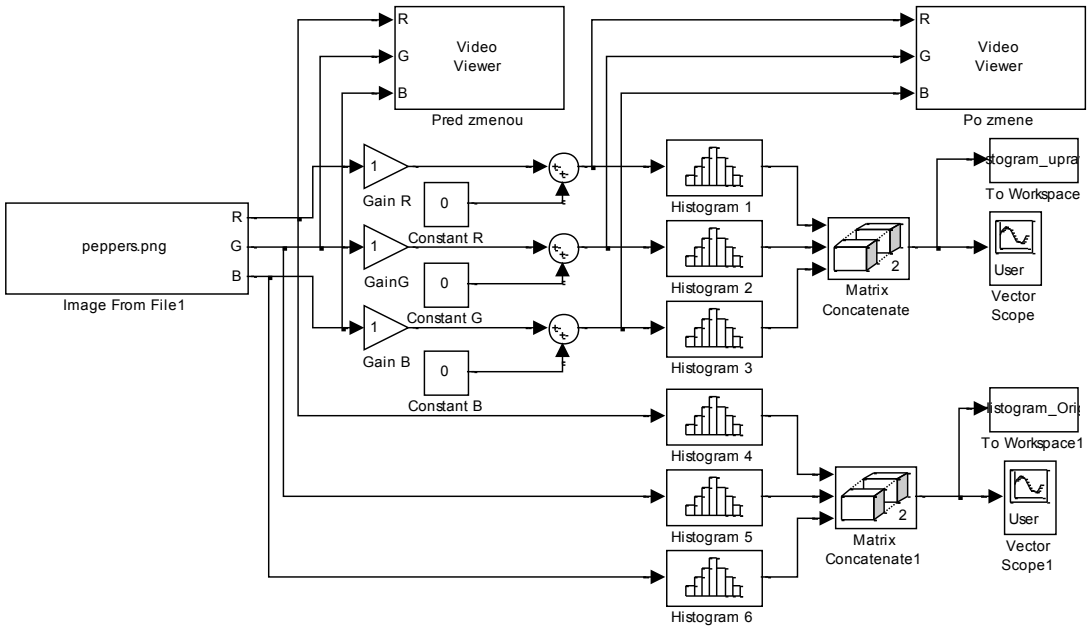
CVIČENÍ

Práce s histogramem, úprava jasu a kontrastu obrazu

V tomto cvičení se seznámíme s možnostmi korekce obrazu. Ukážeme si jak vytvořit model pro úpravu jednotlivých složek histogramu, tak také jak provést korekci jasových hodnot černobílého a barevného obrazu. Pomocí histogramu vidíme rozložení jasů a barev v obraze. Neexistuje žádné ideální rozložení, ale křivka jednotlivých rozložení se může lišit dle snímku. Je-li v obraze potlačena některá z jasových či barevných složek, je toto viditelné právě v histogramu.

Histogram

Model, se kterým budeme pracovat, zobrazuje histogram jednotlivých barevných složek a umožňuje jasovou a barevnou korekci těchto složek. Schéma soustavy sestavíme dle Obr.2.5.



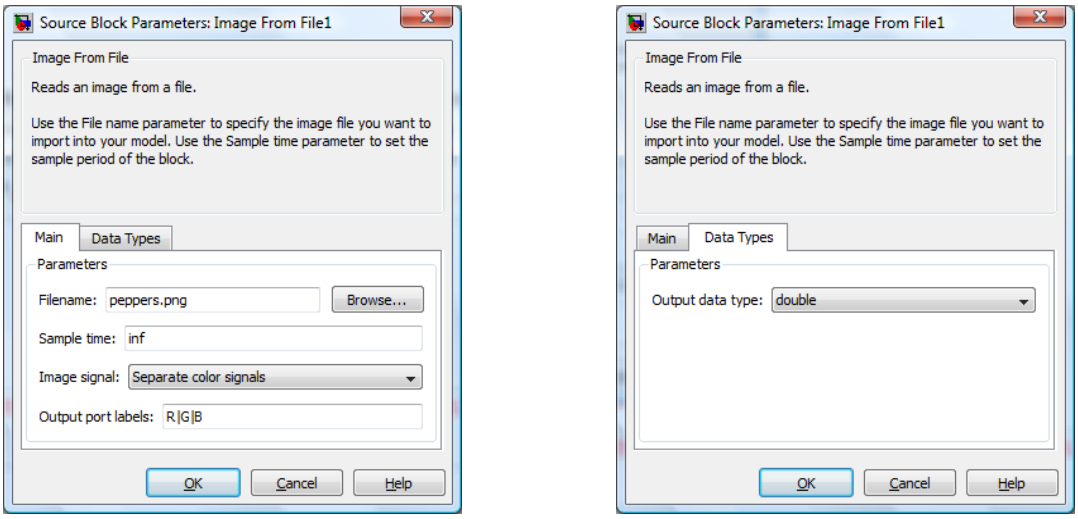
Obr. 2.5 Blokové schéma zapojení soustavy v Simulinku s možností manuální korekce jednotlivých barevných složek.

Jednotlivé bloky naleznete

- Blok ***Histogram*** nalezneme v menu ***Video and Image processing Blockset*** -> ***Statistics***.
- Blok ***Matrix Concatenate*** nalezneme v menu ***Signal Processing Boxset*** -> ***Math Functions*** -> ***Matrics and Linear algebra*** -> ***Matrics operations***.
- Blok ***Vector Scope*** nalezneme v menu ***Signal Processing Boxset*** –> ***Signal Processing Sinks***.
- Blok ***To workspace*** nalezneme v menu ***Simulink*** –> ***Sinks***. Jeho funkcí je převod dat do workspace.

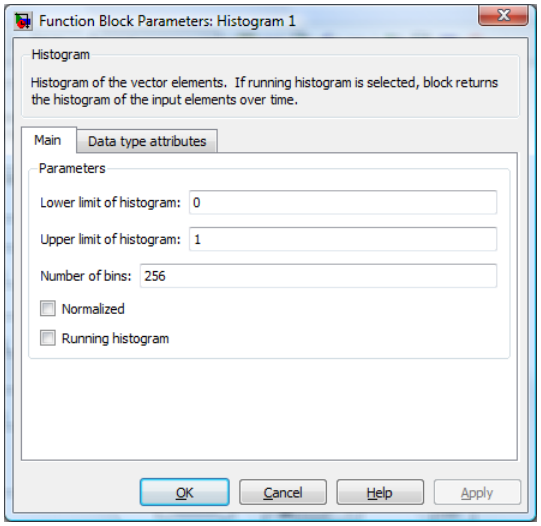
Nastavení vstupních parametrů

- Nastavení ***Image From File*** pro tuto aplikaci je následující. Parametr **Sample time** nastavíme na hodnotu **inf**, parametr **Image signal** nastavíme na hodnotu **Separate color signals**. Definice výstupních parametrů je následující. Parametr **Output port labels** nastavíme na hodnotu **RGB** a parametr **Output data type** nastavíme na hodnotu **double**.



Obr. 2.6 Nastavení vstupních parametrů bloku Image From File. Vlevo nastavení v záložce Main, vpravo nastavení v záložce Data Types.

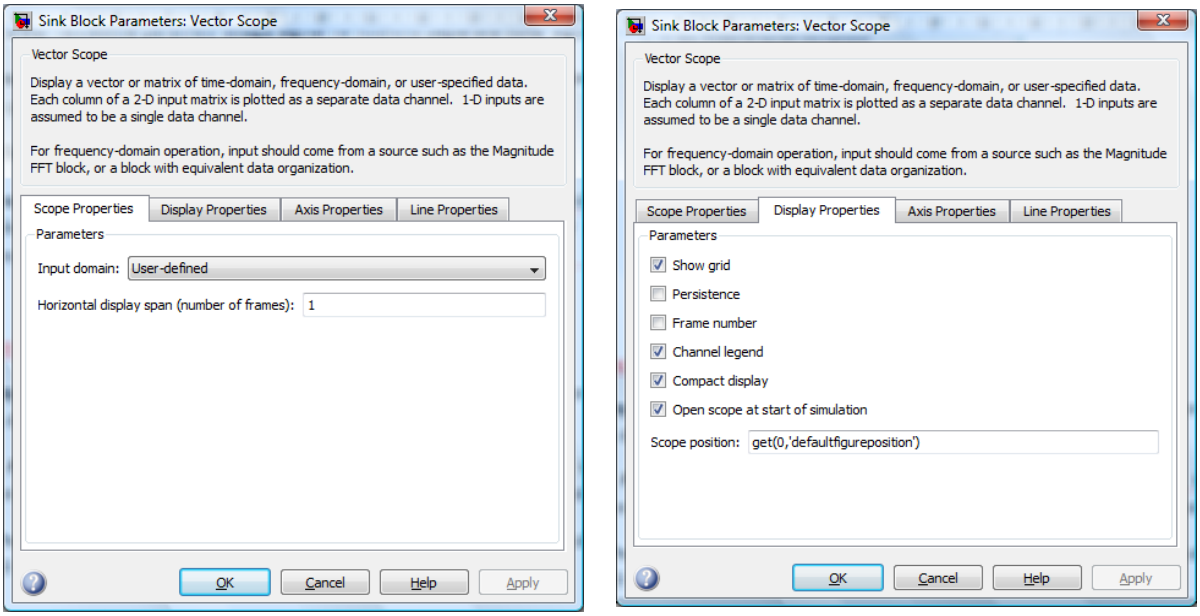
- V bloku ***Histogram*** nastavíme dle Obr.2.7 Parametr **Lower limit of histogram** nastavíme na hodnotu **0,** parametr **Upper limit of histogram** nastavíme na hodnotu **1** a parametr **Number of bins** nastavíme na hodnotu **256**.



Obr. 2.7 Nastavení parametrů bloku Histogram.

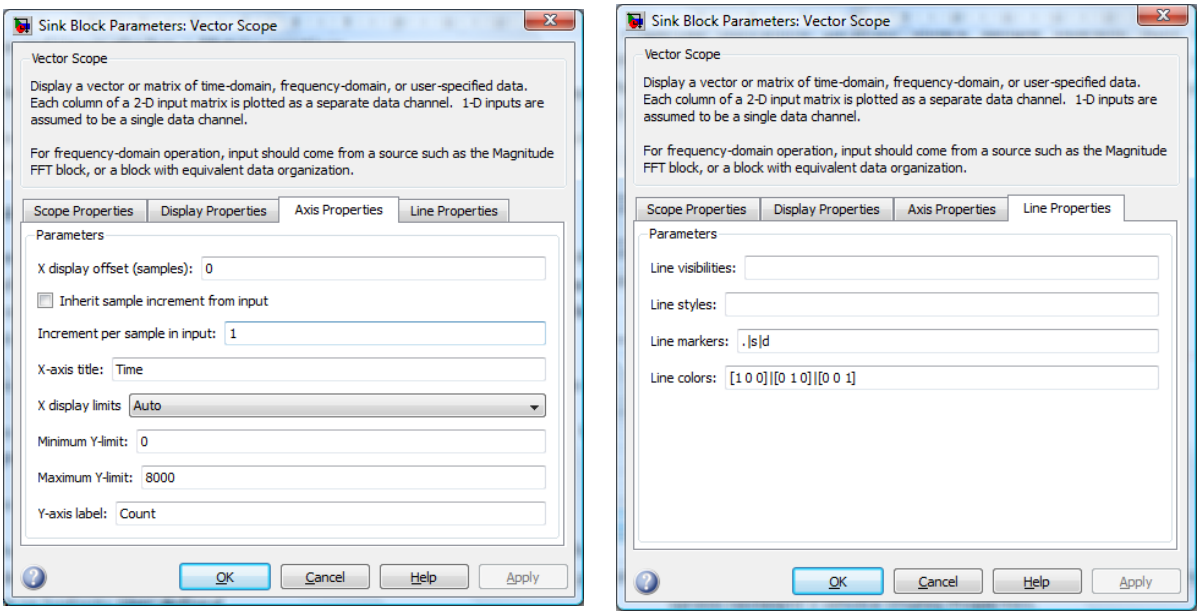
- V bloku ***Matrix Concatenate*** nastavíme parametr **Number of inputs = 3.**

- V bloku ***Vectore Scope*** provedeme nastavení parametrů dle následujících obrázků (Obr. 2.8 a Obr. 2.9.)**.** V jednotlivých záložkách provedeme následující nastavení. V záložce ***Scope Properties*** nastavíme parametr ***Input domain*** na hodnotu ***User defined.*** V záložce ***Display properties*** odstraníme parametr ***Frame number*** z výběru a parametry ***Cannel legend*** a ***Compact display*** do výběru přidáme.



Obr. 2.8 Nastavení vstupních parametrů bloku Vector Scope. Vlevo nastavení v záložce Scope Properties, vpravo nastavení v záložce Display Properties.

- V záložce ***Axis Properties*** odstraníme parametr **Inherit sample inherent from input**. Nastavíme parametry Minimum Y-limit = 0, Maximum Y-limit = 8000, Y-axis label = count. V záložce ***Line Properties*** nastavíme parametr Line markers = .|s|d a parametr Line colors = [1 0 0]|[0 1 0]|[0 0 1].
-



Obr. 2.9 Nastavení vstupních parametrů bloku Vector Scope. Vlevo nastavení v záložce Axis Properties, vpravo nastavení v záložce Line Properties.

- V bloku ***Video Viewer*** nastavíme parametr **Image signal** na hodnotu **Saparate color signals**.

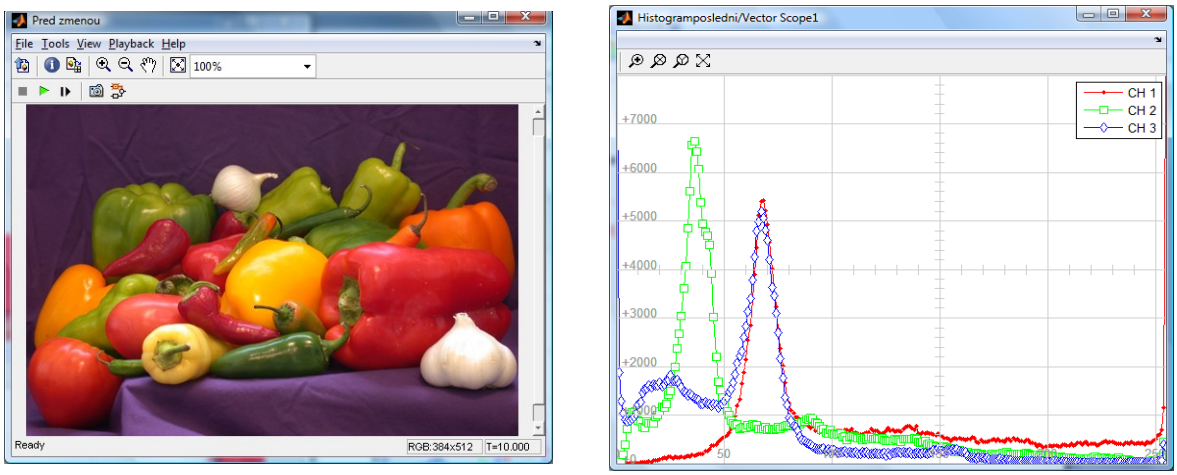
Nastavení konfiguračních parametrů v Simulinku

Při zpracování histogramu není vykreslení v kontinuálním čase podporováno. Tento fakt je zapotřebí ošetřit následujícím nastavením.

- Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

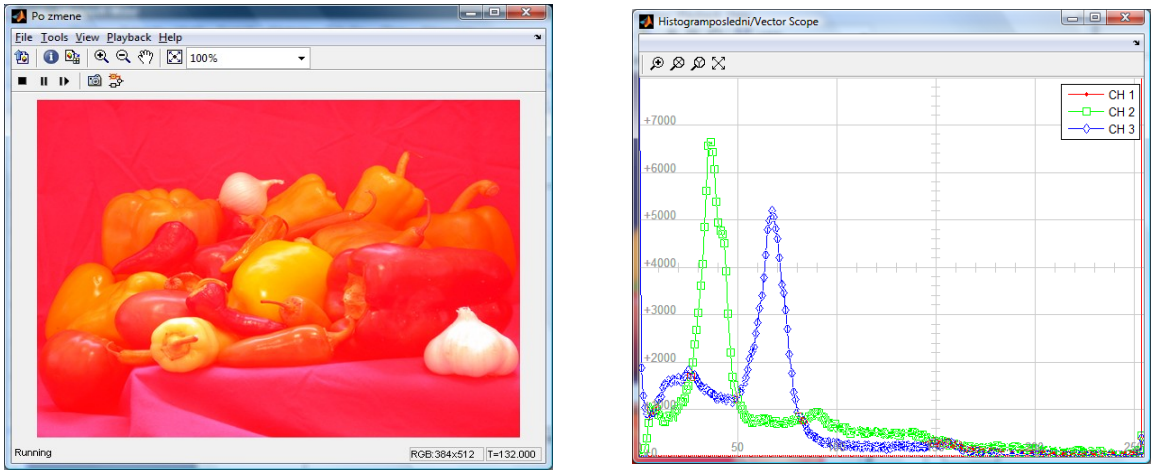
Propojte jednotlivé bloky mezi sebou a spusťte simulaci.

Výsledek simulace je zobrazen do několika oken, viz následující obrázky.



Obr. 2.10 Vlevo původní obrázek, vpravo histogram původního obrázku.

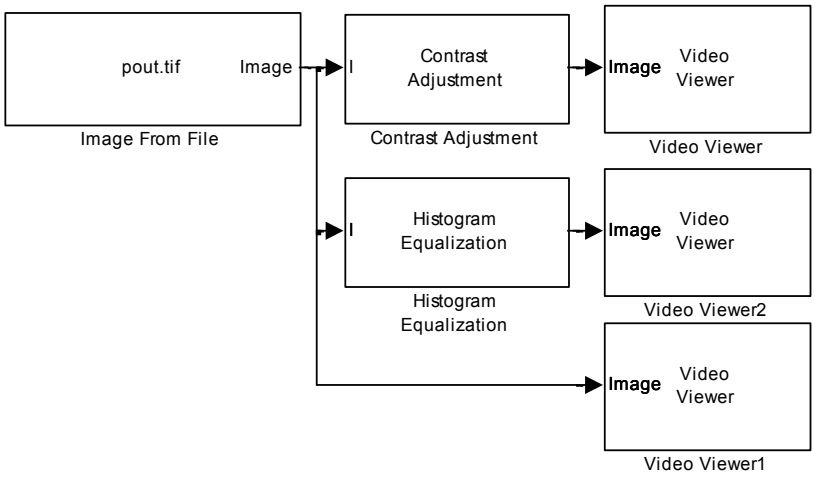
Chceme-li s obrázkem pracovat a upravit jednotlivé jeho složky, provedeme úpravu buď přičtením nebo odečtením konstanty v blocích Konstant pro jednotlivé složky RGB anebo násobením konstanty pro jednotlivé složky v blocích Gain.



Obr. 2.11 Změna červené složky histogramu. Vlevo upravený obrázek , vpravo histogram upraveného obrázku.

Úprava jasu a kontrastu BW obrazu

Sestavte model dle následujícího schématu (Obr. 2.12.).



Obr. 2.12 Blokové schéma zapojení.

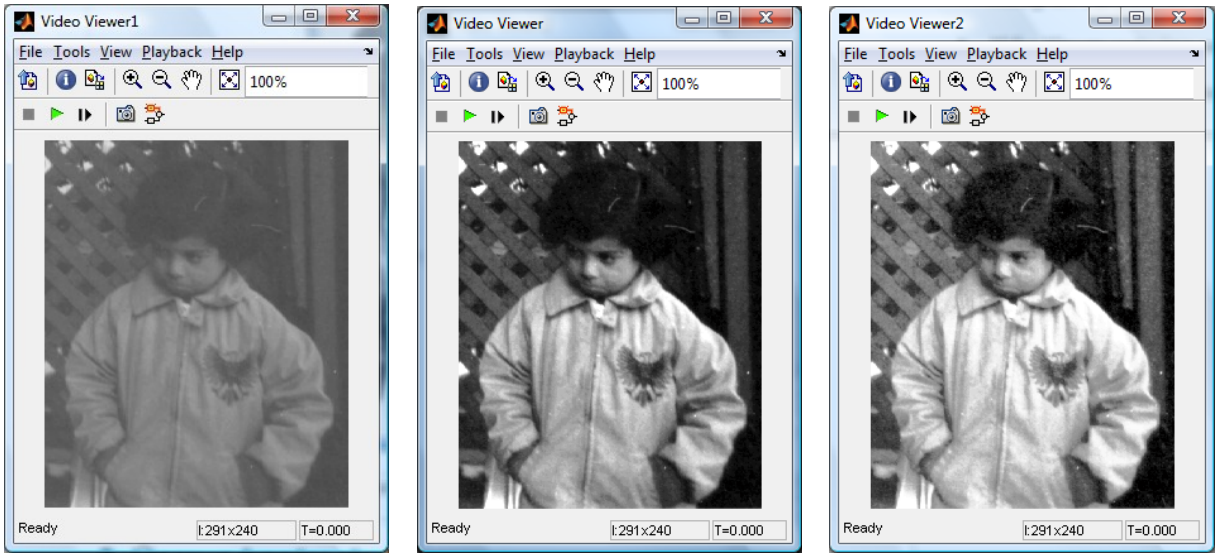
Jednotlivé bloky naleznete

- Blok ***Contrast Adjustment*** a ***Histogram equalization*** nalezneme v menu ***Video and Image processing Blockset*** -> ***Statistics***.

Nastavení vstupních parametrů

- Nastavení bloku ***Image From File*** pro tuto aplikaci je následující. Definujeme název souboru pomocí parametru **Filename = tire.tif**, parametr **Sample time** nastavíme na hodnotu **0**, parametr **Image signal** nastavíme na hodnotu **One multidimensional signal**.
- V bloku ***Contrast Adjustment*** ponecháme defaultní nastavení parametrů.
- V bloku ***Histogram Equalization*** ponecháme defaultní nastavení parametrů.
- V bloku ***Video Viewer*** nastavíme parametr **Image signal** na hodnotu **One multidimensional signal**.

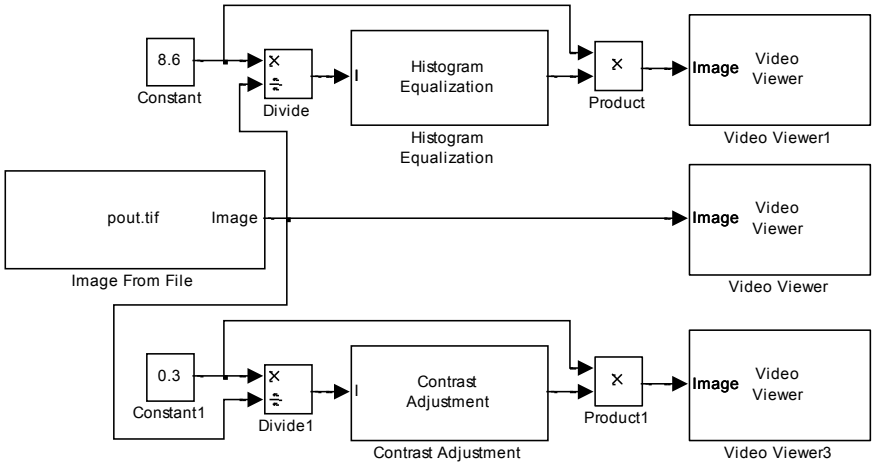
Propojte jednotlivé bloky mezi sebou (Obr. 2.12.), nastavte parametry simulace a spusťte simulaci.



Obr. 2.13 Vlevo originál obrázku, uprostřed úprava kontrastu pomocí Contrast Adjustment, vpravo úprava kontrastu pomocí Histogram Equalization.

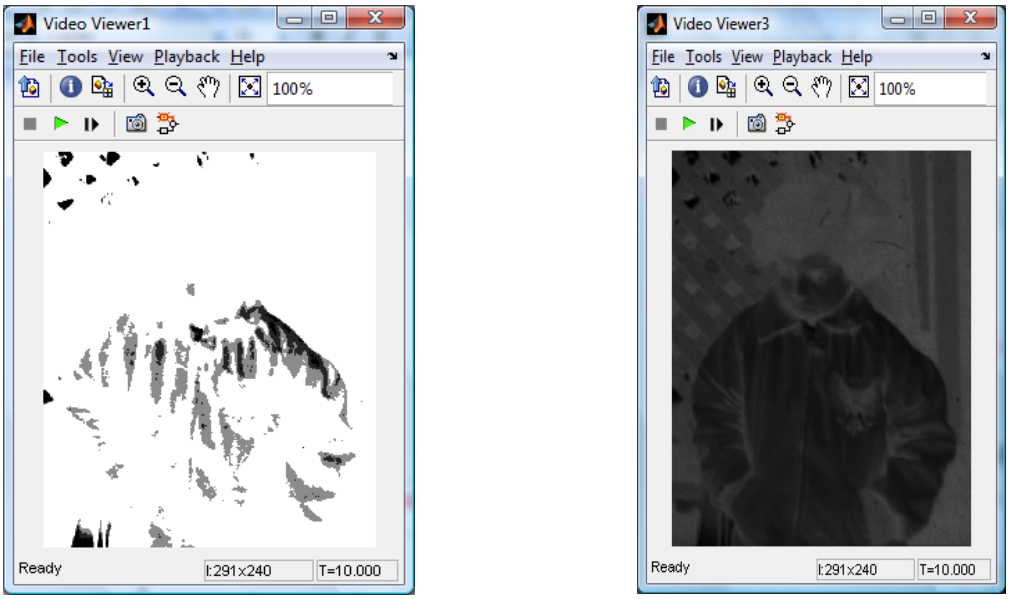
Na Obr. 2.13 je patrný vliv úpravy histogramu.

Manuální korekci jasu nebo histogramu pak můžeme provést pomocí schéma na Obr.2.14.



Obr 2.14. Schéma zapojení modelu pro změnu jasu a kontrastu.

Ve cvičení si vyzkoušejte změnit jednotlivé úrovně obou složek korekce obrazu.

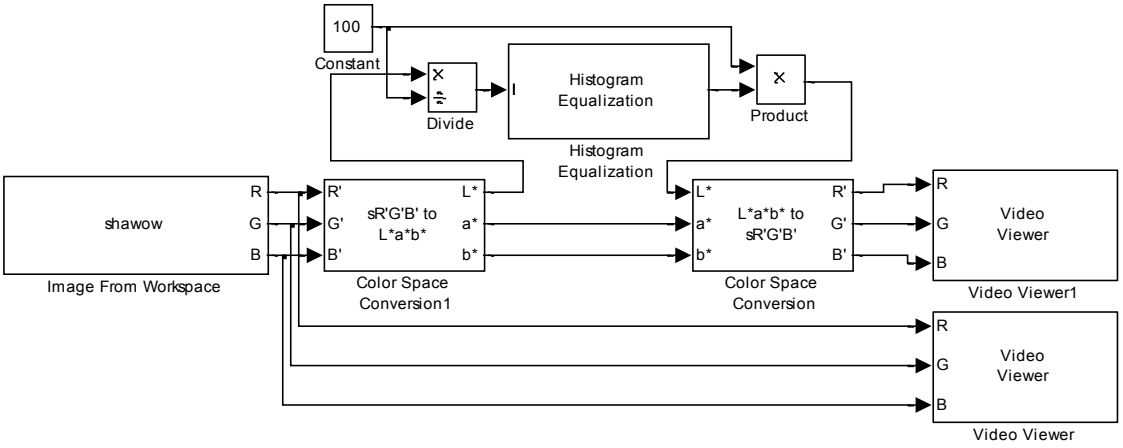


Obr.2.15. Vlevo úprava kontrastu na hodnotu 0,3, vpravo úprava jasu na hodnotu 8,9.

Na Obr. 2.15 jsou zobrazeny extrémní úpravy jasu a kontrastu pro názornost. Nelze obecně stanovit v jakém rozsahu se mají hodnoty pohybovat, jelikož každý obrázek má svá specifika a svou křivku histogramu a rozložení jasu. Nastavení, které se ukazuje jako ideální pro jednu scénu, může být zcela nevhodné pro scénu jinou.

Úprava jasu a kontrastu barevného obrazu

Sestavte model dle schématu na Obr.2.16.



Obr. 2.16. Schéma zapojení modelu.

Jednotlivé bloky nalezneme

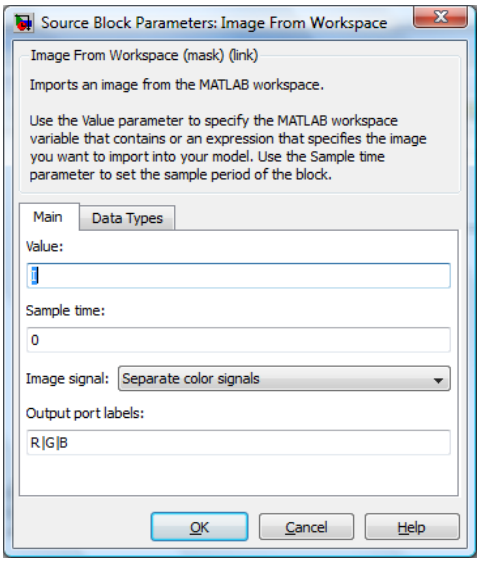
- Všechny použité bloky již byly popsány v předcházejících odstavcích.

Nastavení vstupních parametrů

Před tvorbou samotného modelu v Simulinku nejdříve uložíme vstupní obrázek do workspace Matlabu pomocí následujících dvou příkazů:

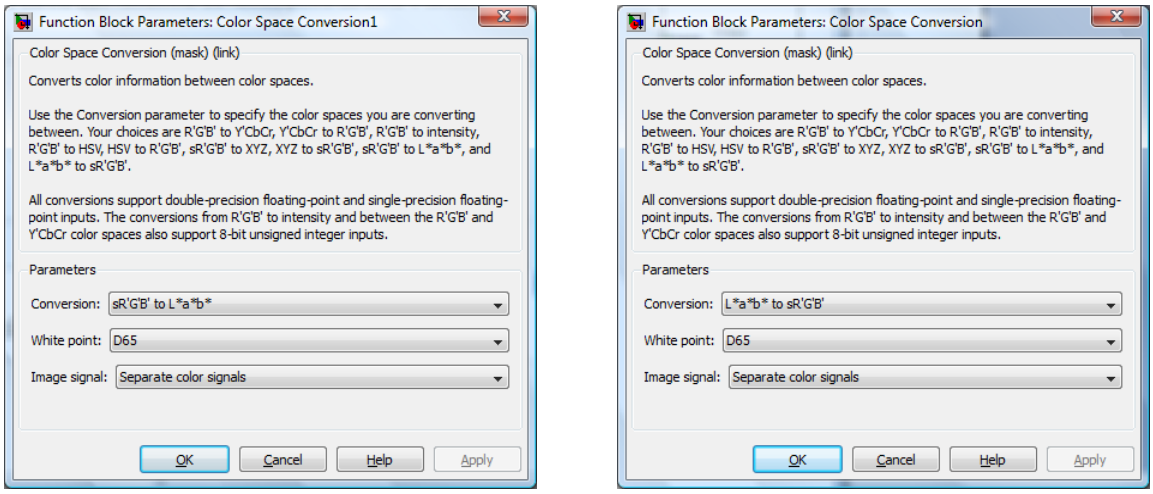
[X map] = imread('shadow.tif');	vloží obrázek do workspace
shadow = ind2rgb(X,map);	převede obrázek do RGB barevného prostoru

- Parametry bloku ***Image From Workspace*** nastavíme dle Obr. 2.17. V záložce Main nastavíme parametr **Value** na hodnotu **I**, parametr **Sample time** na hodnotu **0** a parametr **Image signal** na hodnotu **Separate color signals**.



Obr. 2.17 Nastavení parametrů bloku Image From Workspace v záložce Main.

- Parametry bloků ***Color Space Conversion*** a ***Color Space Conversion1*** nastavíme dle Obr.2.18. V záložce main nastavíme parametr **Conversion** na hodnotu **RGB to Lab**, parametr **Image signal** na **Separate color signals**. Nastavení bloku pro konverzi výstupního signálu provedeme následujícím způsobem. Parametr **Conversion** nastavíme na hodnotu **Lab to RGB** a parametr **Image signal** nastavíme na hodnotu **Separate color signals**. Parametr **White point** ponecháme u obou bloků přednastavený na hodnotu **D65**.



Obr. 2.18. Vlevo nastavení parametrů bloku Color Space Conversion pro vstupní signál, vpravo nastavení pro výstupní signál.

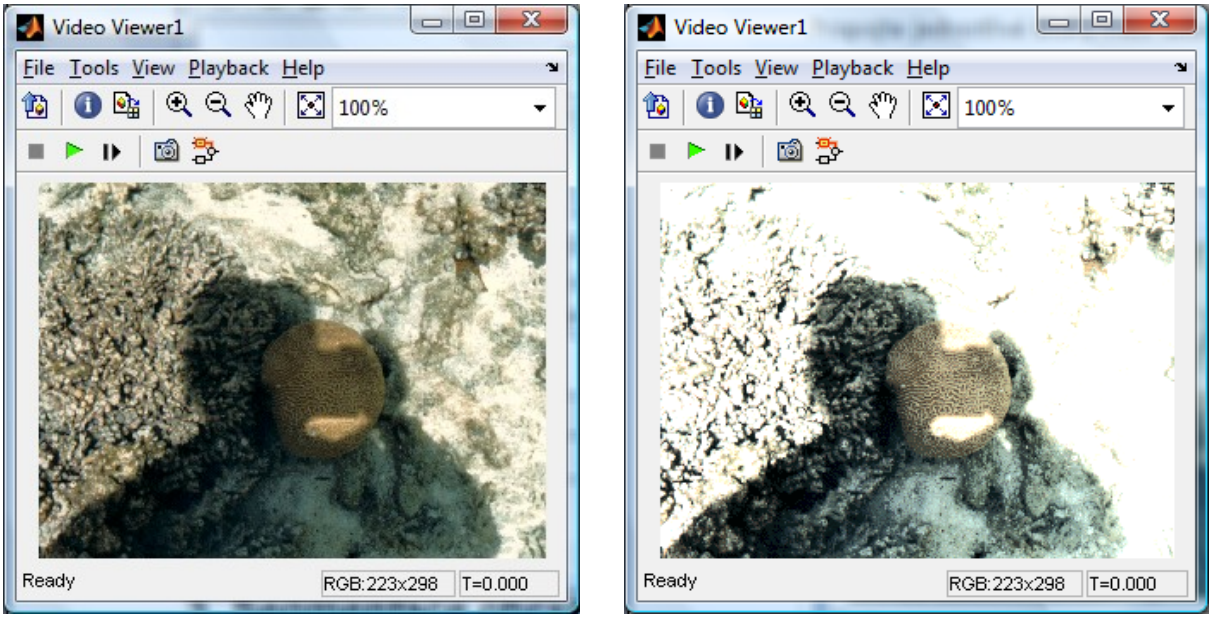
- Parametry bloku ***Histogram Equalization*** ponecháme defaultně přednastaveny.
- Parametr **Constant Value** bloku ***Constant*** nastavíme na hodnotu **200**.
- Parametry bloku ***Devide*** ponecháme defaultně přednastaveny.
- Parametry bloku ***Product*** ponecháme defaultně přednastaveny.
- Parametr **Image signal** u obou bloků ***Video Viewer*** nastavíme na hodnotu **Separate color signals**.

Nastavení konfiguračních parametrů v Simulinku

Úpravu provedeme v menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

Jednotlivé bloky mezi sebou propojíme, definujeme parametry simulace a můžeme spustit simulaci.

Na Obr. 2.19 je viditelné změna vůči původnímu obrazu při korekci jasu i kontrastu po průchodu soustavou bloků dle schéma na Obr. 2.16.



Obr. 2.19. Vlevo originál obrázku, vpravo po průchodu modelem při nastavení konstanty na hodnotu 200.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem *02 Korekce_obrazu1. swf* a *02 Korekce_obrazu_1_2.swf.*

Demonstrační video k této kapitole je k dispozici v adresáři Videa pod názvem *video_histogram.mpg*.

3. DETEKCE HRAN

Čas ke studiu: 3 hodiny

Cíl: Po prostudování tohoto odstavce budete umět

- popsat co je to hrana, jaké jsou hranové detektory a jak provedeme detekci hran v obraze pomocí definovaných hranových detektorů.
- definovat jednotlivé typy hran a zvolit tak nejvhodnější hranový detektor.
- vyřešit úlohu v Matlab/Simulink a VIP pro nalezení hran v obraze.

Výklad

3.1 Úvod

Detekce hran je postup v digitálním zpracování obrazu, sloužící k nalezení oblastí pixelů, ve kterých se podstatně mění jas. Hrana v objektu se nemusí krýt s hranicí mezi objekty ve scéně, hrany mohou vznikat a zanikat v závislosti na úhlu pohledu. Hrany tedy můžeme najít na hranici objektů nebo rozhraní světla a stínu tzn. **skoková hrana**, nebo v místech trojrozměrných hran objektů tzn. **trojúhelníková hrana**. Čáry v obrazu pak generují dvě hrany, jednu na každé své straně. Typická hrana na rozdíl od teoretické bývá ovšem zašuměná.
Pokud hranu definujeme jako velkou změnu jasové funkce, bude v místě hrany velká hodnota derivace jasové funkce. Maximální hodnota derivace bude ve směru kolmo na hranu. Kvůli jednoduššímu výpočtu se ale hrany detekují jen ve dvou, resp. ve čtyř směrech. Velká skupina metod na detekci hran aproximuje tuto derivaci pomocí konvoluce s vhodným jádrem. Nejjednodušší taková jádra jsou *(-1,1)* a *(-1, 0, 1)*.

3.1.1 Detekce nespojitostí

V této části výukového textu se budeme zabývat některými technikami pro vyhledávání tří základních typů nespojitostí monochromatického digitálního obrázku, a to: bodů, čar a hran. Nejčastěji používanou technikou je aplikace **konvoluční masky** na celý obrázek.

Konvoluční maska o velikosti 3x3 je např.

$$\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}$$

Technika využívající konvoluční masku zahrnuje výpočet součtu součinů koeficientů s úrovněmi šedi, které jsou obsažené v oblasti pokryté konvoluční maskou. To znamená, že výstup po konvoluci masky a obrázku v libovolném bodě je dán vztahem [4]

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i \tag{3.1}$$

kde z_i je úroveň šedi v pixelu, který koresponduje s koeficientem masky w_i

3.1.2 Detekce bodů

Samostatné body jsou detekovány v místech, kde platí

$$|R| \geq T$$

kde T je nezáporná prahová hodnota.

Jinými slovy se jedná o rozdíl mezi středovým bodem a jeho okolím. Předpokladem je, že izolovaný bod je zcela odlišný od svého pozadí, což může být jednoduše detekováno předchozí maskou.
Maska pro detekci bodů má **tvar hornopropustného filtru**. (Součet koeficientů masky je roven 0).

Příklad konvoluční masky je uveden níže.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Z příkladu konvoluční masky vyplývá, že v místě, kde je v obraze konstantní jas, bude odezva masky nulová.

3.1.3 Detekce čar

Princip detekce čar je stejný jako detekce samostatných bodů. Můžeme použít masky, které vybírají čáry jen v určitém směru.

Konvoluční masky mohou mít následující tvary

$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$	$\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$
Horizontální	+45°	Vertikální	-45°

Opět si můžete všimnout, že součet všech koeficientů v jednotlivých maskách je roven nule.

Pokud chceme najít **nejsilnější čáry** můžeme použít všechny masky individuálně pro daný obrázek. Předpokládejme, že odezvy na jednotlivé tvary konvolučních masek (horizontální, +45°, vertikální, -

45°) jsou R_1, R_2, R_3, R_4. Jestliže pro určitý bod v obrázku platí, že $|R_i| > |R_j|$ pro všechna $j \neq i$, tak můžeme říct, že bod nejvíce odpovídá čáře ve směru masky *i*.
Např. jestliže v určitém bodě obrázku je $|R_1| > |R_j|$ pro $j = 2,3,4$, pak můžeme říct, že tento bod koresponduje s horizontální čárou.

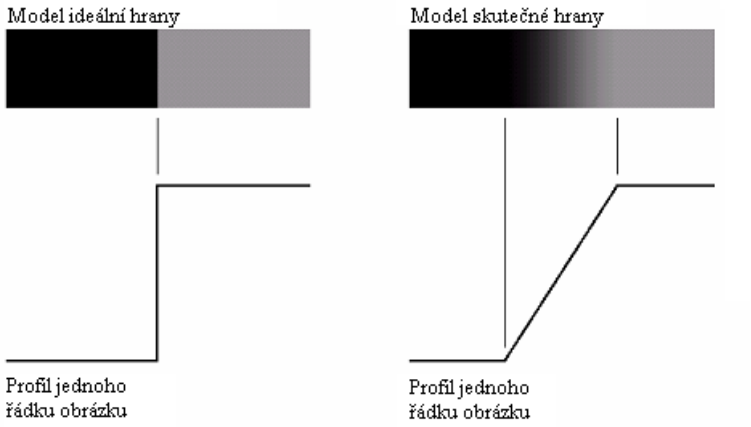
3.2 Detekce hran

❑ Pojem hrany a princip gradientních operátorů

Při vnímání okem člověka se jeví jako velmi důležitá ta místa v pozorovaném obraze, kde se náhle mění hodnota jasu.

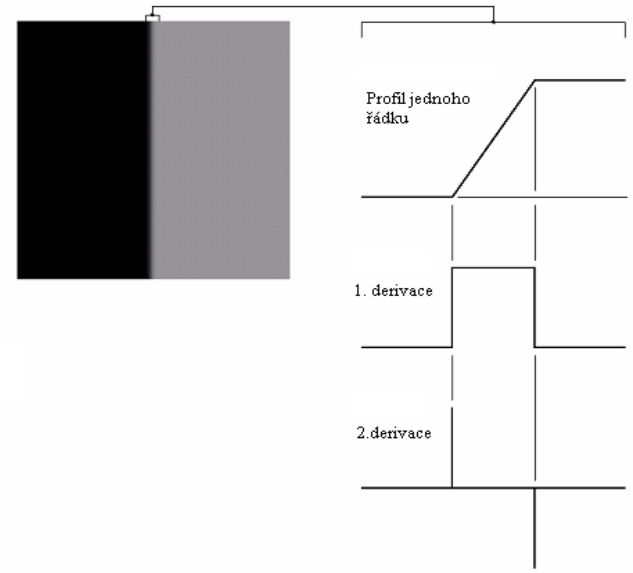
Odpovídající pixely se nazývají hrany. Například u perokresby, kdy je potřebné silně generalizovat a zaznamenat scénu velmi zjednodušeně, budou linie odpovídat právě náhlým jasovým změnám. K automatickému nalézání takových významných míst v obraze slouží právě lokální předzpracování - **hledání hran** [2].

Hrana v obraze je dána vlastnostmi obrazového elementu a jeho okolí.



Obr. 3.1 Model ideální a skutečné hrany.

Na Obr. 3.1 je zobrazen model **ideální hrany** a model **skutečné hrany**, která je vlivem optiky, vzorkování a dalších vlivů rozmazána. Tloušťka hrany je určena délkou rampy, tzn. jak se v místě hrany obrazová funkce mění. Tato délka je určena strmostí této funkce, která je zase dána stupněm rozmazání.



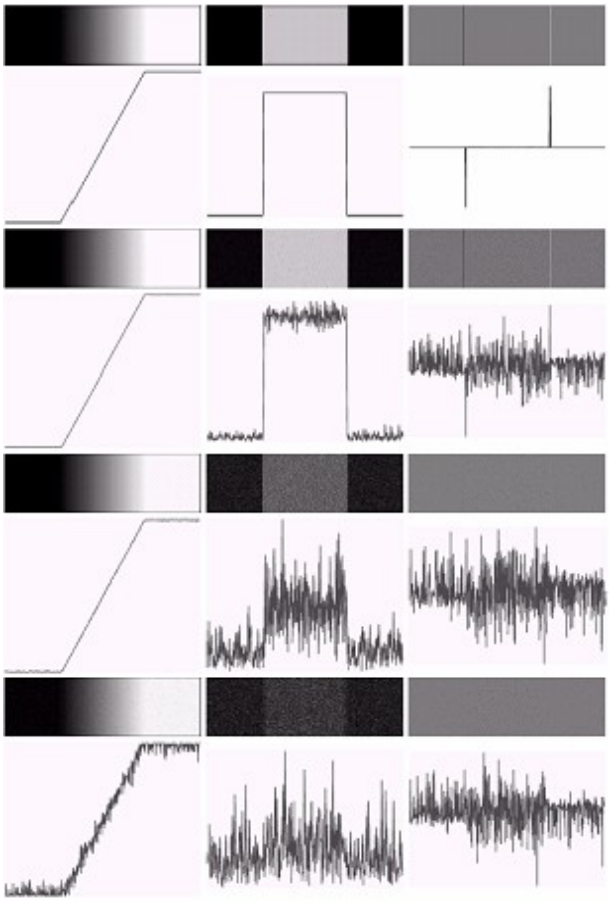
Obr. 3.2 Detail skutečné hrany, profil jednoho řádku, 1. a 2. derivace horizontálního profilu.

Na Obr. 3.2 je ukázán detail skutečné hrany spolu s horizontálním profilem hrany mezi dvěma oblastmi. Na obrázku jsou zachyceny také 1. a 2.derivace horizontálního profilu. První derivace je kladná v bodech přechodu a nulová pro místa s konstantním jasem. Druhá derivace je kladná na přechodu odpovídajícím tmavé straně hrany (funkce dosahuje minima) a záporná na přechodu odpovídajícího světlé straně hrany (funkce dosahuje maxima), v místě přechodu je 2.derivace nulová.

Z předchozího výkladu je patrné, že hrana může být určena z **amplitudy 1. derivace**, nebo může být podle **znaménka 2. derivace** určeno zda bod leží na tmavé nebo na světlé straně hrany.
Pokud pro detekci hran použijeme 2. derivaci všimněme si následujících aspektů:

1. Vždy vypočteme 2 hodnoty pro každou hranu (nevýhoda)
2. Pomyslná čára spojující maximální kladnou a zápornou hodnotu 2. derivace prochází nulou uprostřed hrany, což je výhodné pro detekci středu tlusté čáry.

Zatím jsme se zabývali jednoduchou hranou, která není zašuměná, Obr. 3.2. Na Obr. 3.3 je ukázka zašuměné hrany, spolu s 1. a 2. derivací obrazové funkce. Hrana je narušená Gaussovským šumem. Jak je vidět z prostředního sloupce, který představuje 1.derivaci obrazové funkce, čím je šum větší, tím je nalezení hrany obtížnější. Na výsledku je zajímavé to, že šum není na obrázku téměř viditelný. Z tohoto je patrná citlivost derivace na šum.



Obr. 3.3 Hrana narušená Gaussovským šumem s nulovou střední hodnotou

Pokud se podíváme na vyhledávání hrany s využitím druhé derivace je výsledek ještě zajímavější. Druhá derivace je ještě více citlivá na šum. Můžete si všimnout, že detekce kladných a záporných komponent, které jsou nutné pro určení hrany, by byla velmi obtížná.

Jak vyplývá z předchozího textu je hrana určena tím, jak náhle se mění hodnota obrazové funkce *f(x, y)*.

Matematickým nástrojem pro studium změn funkce dvou proměnných jsou **parciální derivace**. Změnu funkce udává její **gradient**

$$\nabla \mathrm{f} = \begin{bmatrix} \mathrm{G_x} \\ \mathrm{G_y} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathrm{f}}{\partial \mathrm{x}} \\ \frac{\partial \mathrm{f}}{\partial \mathrm{y}} \end{bmatrix} \tag{3.2}$$

Gradient je vektorová veličina ∇, určující směr největšího růstu funkce (směr gradientu) a strmost tohoto růstu (velikost, modul gradientu).

Pixely s velkým modulem gradientu se nazývají hranami.

Pro spojitou obrazovou funkci *f(x, y)* jsou **velikost gradientu** $|\nabla f(x, y)|$ a **směr gradientu** ψ

dány vztahy [2]

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{3.3}$$

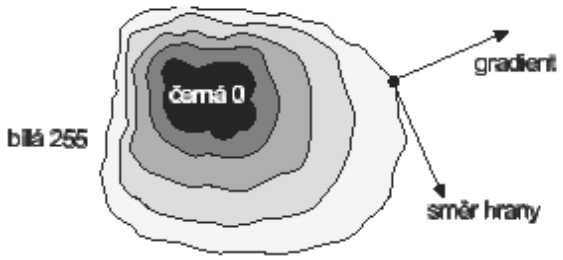
$$\psi(x,y) = \arg\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) \tag{3.4}$$

kde *arg(x, y)* je úhel (v radiánech) mezi souřadnou osou x a radiusvektorem k bodu (x, y).

Někdy nás zajímá pouze velikost gradientu (též velikost hrany) bez ohledu na její směr.

Pro odhad velikosti se používá **všesměrový lineární Laplaceův operátor** - Laplacián ∇^2, který vychází z druhých parciálních derivací

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{3.5}$$



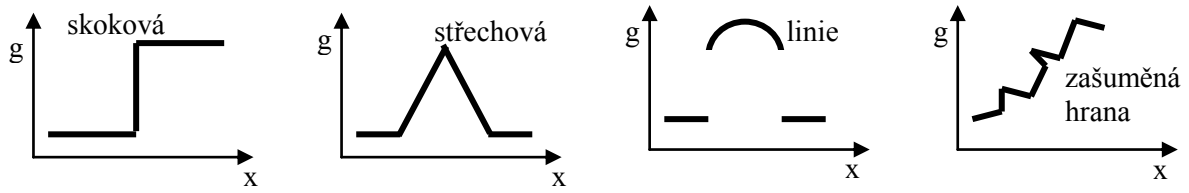
Obr. 3.4 Směr hrany ϕ je kolmý na směr gradientu ψ. [2]

Hrany nalezené v obraze lokálními operátory se někdy používají pro hledání hranic objektů.

Za předpokladu, že objektu odpovídá oblast homogenního jasu, jsou body hranice právě pixely s vysokou hodnotou gradientu. **Hranové pixely se spojují do hranic, a proto se směr hrany ϕ někdy definuje jako kolmý na směr gradientu ψ.** Hranové pixely se při takové volbě orientace hrany mohou přirozeně spojovat do hranic.

Hrany lze dobře třídit podle jednorozměrného jasového profilu ve směru gradientu v daném pixelu.

Typické příklady jsou na na Obr. 3.5**.** První tři profily, tj. skoková hrana, střechová hrana, tenká linie, jsou idealizované. Poslední profil odpovídá zašuměné hraně, kterou lze najít v reálném obrázku.



Obr. 3.5 Jasové profily nejběžnějších hran.[2]

3.2.1 Gradientní operátory

Gradientní operátoři nám obecně udávají strmost obrazové funkce. Je možno je rozdělit do tří základních typů:[2]

1. **Operátory aproximující derivace pomocí diferencí**. Některé operátory jsou invariantní vůči rotaci, např. Laplacián, a mohou být počítány konvolucí s jedinou maskou. Jiné, aproximující první derivaci, využívají několik masek odpovídajících příslušné orientaci. Z nich se vybere ta, která nejlépe lokálně aproximuje obrazovou funkci. Výběrem jedné z masek je určen i směr gradientu (orientace).
2. **Operátory založené na hledání hran v místech, kde druhá derivace obrazové funkce prochází nulou** (angl. zero-crossing). Příkladem je Marrův-Hildrethové operátor a Cannyho hranový detektor, o nichž bude dále řeč.
3. **Operátory snažící se lokálně aproximovat obrazovou funkci poměrně jednoduchým parametrickým modelem**, např. polynomem dvou proměnných.

Kromě hledání hran v obrázku můžeme **gradientní operátory** využít i pro **ostření obrazu**.

Cílem ostření obrazu je upravit obraz tak, aby v něm byly **strmější hrany**. Zostřený obraz bude pozorovat člověk.

Z hlediska plošných frekvencí odpovídá ostření zdůraznění vysokých frekvencí. Pro obraz f, který je výsledkem ostření obrazu g, lze napsat

$$f(x,y) = g(x,y) - C\,S(x,y) \tag{3.6}$$

kde *C* je kladný součinitel udávající požadovanou sílu ostření *S(i, j)* je operátor udávající strmost změny obrazové funkce v příslušném bodě. Strmost může být dána modulem gradientu nebo Laplaciánem. Ostření obrazu s využitím Laplaciánu ilustruje Obr. 3.6

Obr. 3.6 Ostření obrazu: (a) Výchozí obraz. (b) Odezva Laplaciánu v 8-okolí. (c) Výsledek ostření podle (3.6), kde S(x,y) je Laplacián s C=0,4 [2]

Ostření obrazu lze také interpretovat ve frekvenčním spektru jako zdůraznění vysokých frekvencí. Víme již, že výsledek Fourierovy transformace je lineární kombinací harmonických průběhů, např. *sin(nx),* n = 1, 2, … Derivace harmonické bude n *cos(nx)*. Je vidět, že čím vyšší frekvence (zde vyšší n), tím větší je amplituda její derivace. Jde o další zdůvodnění, **proč gradientní operátory zdůrazňují hrany**.[2]

Ostření obrazu se používá, když se má dosáhnout kontrastnějšího obrazu, a to buď na displeji, nebo při tisku. Na běžné laserové tiskárně a v řadě polygrafických technik (např. při sítotisku) se odstíny jasu vytvářejí polotónováním, tj. větší či menší hustotou černých bodů. Před tiskem pomocí polotónování bývá ostření obrazu nezbytné. Příslušný modul je součástí každého programového balíku pro publikování pomocí počítače (též DTP z angl. Desk Top Publishing).[2]

❑ Jednoduché konvoluční masky aproximující derivace obrazové funkce [2]

Dále budeme uvažovat ty operátory**, které lze vyjádřit jako masky pro konvoluci**. Jednotlivé operátory budou uváděny pomocí příslušného **konvolučního jádra *h***. U směrových operátorů je tolik jader *h*, kolik směrů operátor rozlišuje.

Robertsův operátor

Nejstarší a velmi jednoduchý je Robertsův operátor, který používá jen okolí 2 x 2 reprezentativního pixelu.
Jeho konvoluční masky jsou

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Velikost gradientu se počítá podle

$$|g(x,y) - g(x+1,y+1)| + |g(x,y+1) - g(x+1,y)| \qquad (3.7)$$

Hlavní **nevýhodou** Robertsova operátoru je **velká citlivost na šum**, protože okolí použité pro aproximaci je malé.

Laplaceův operátor

Již jsme se zmínili o velmi rozšířeném Laplaceově gradientním operátoru ∇^2

- aproximuje druhou derivaci
- je invariantní vůči otočení
- udává jen velikost hrany a ne její směr.

V digitálním obraze je také Laplacián aproximován diskrétní konvolucí. Dvě používaná konvoluční jádra (pro 4-sousedství a 8-sousedství) v okolí 3 x 3 jsou

$$h_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Někdy se používá Laplacián s **větší vahou pixelů blíže reprezentativnímu bodu masky**. V tomto případě se **ztrácí invariantnost vůči otočení**,

$$h_1 = \begin{bmatrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

Hlavní **nevýhodou** Laplaciánu:

- **velká citlivost na šum**, což je ostatně při snaze aproximovat druhou derivaci primitivními prostředky přirozené.
- **dvojité odezvy na hrany odpovídající tenkým liniím v obraze**.

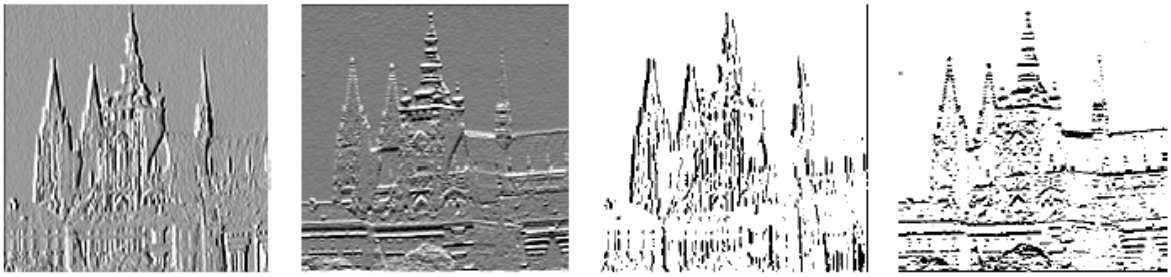
Výsledek činnosti Laplaceova operátoru jsme viděli na Obr. 3.6b).

Operátor Prewittové

Operátor Prewittové, podobně jako Sobelův, Kirschův a Robinsonův, které ukážeme dále,

- aproximuje první derivaci
- gradient je odhadován v okolí 3 x 3 pro osm směrů. Vybrána je jedna maska z osmi, a to ta, které odpovídá největší modul gradientu. Je přirozeně možné vytvářet větší masky s podrobnějším směrovým rozlišením. Abychom ušetřili místo, budeme zde ukazovat jen první tři konvoluční masky. Ostatní si čtenář vytvoří sám pootočením.

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}, \qquad h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$



(a) (b) (c) (d)

Obr. 3.7 Detekce hran operátorem Prewittové rozměru 3 x 3. (a) Hrany v severním směru (čím je pixel světlejší, tím je hrana silnější). (b) Hrany ve východním směru. (c) Silné hrany z (a), které byly získány prahováním. (d) Silné hrany z (b).[2]

Sobelův operátor

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}, \qquad h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Sobelův operátor se často používá **pro detekci vodorovných a svislých hran**, na což postačí masky h_1, h_3.

Robinsonův operátor

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}, \qquad h_3 = \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$$

❑ Hrany jako průchody nulou druhé derivace obrazové funkce [2]

Problémem výše popsaných operátorů **aproximujících derivaci diferencemi v malém okolí** je:

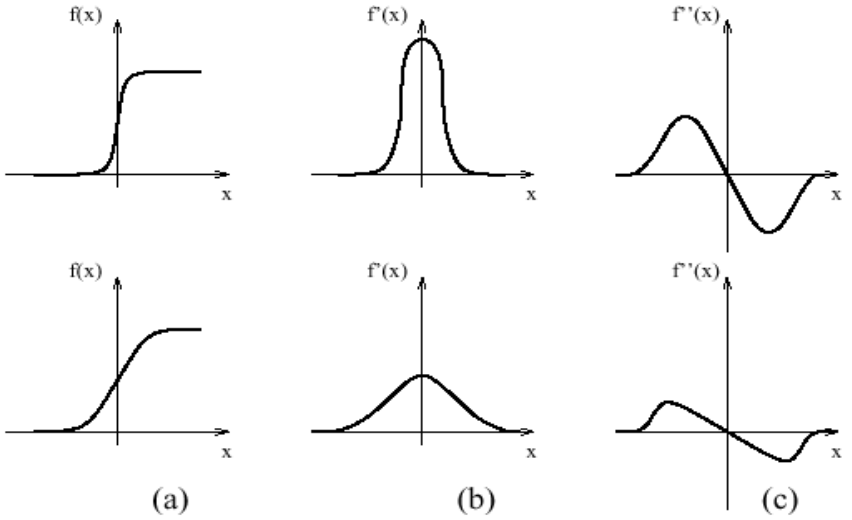
- velká závislost jejich chování na konkrétním obrázku.
- velikost masky musí odpovídat velikosti detailů v obrázku.
- také citlivost na šum je značná.

Koncem sedmdesátých let byla formulována **Marrova teorie**, která usilovala **o matematický model detekce skokových hran** odpovídající neurofyziologickým měřením na sítnici oka.

Základem přístupu je **hledání polohy hrany v obraze v místě průchodu druhé derivace obrazové funkce nulou.**

- První derivace obrazové funkce nabývá svého maxima v místě hrany.
- Druhá derivace protíná v místě hrany nulovou hodnotu.

Situaci ilustruje pro jednoduchost v 1D případě Obr. 3.8, kde (a) ukazuje skokovou hranu, (b) její první derivaci a (c) průběh druhé derivace. Hledat polohu hrany v místě průchodu nulou je díky strmosti přechodu mnohem spolehlivější než na plochém maximu u první derivace.



Obr. 3.8 1D jasový profil ilustrující polohu skokové hrany v místě průchodu 2. derivace obrazové funkce nulovou osou.[2]

Klíčovou otázkou je, jak robustně počítat druhou derivaci.

Naše předchozí zkušenost nás nabádá k ostražitosti.

- Odhad druhé derivace, jak jsme už viděli v případě Laplaciánu, by měl být na šum citlivější než odhad první derivace.
- Klíčem k robustnímu odhadu druhé derivace je **konvoluce obrazu s vyhlazujícím filtrem**, na který jsou dva požadavky:

    1. Filtr má být hladký a ve frekvenčním spektru přibližně odpovídající pásmové propusti, aby omezil možný počet frekvencí, při kterých k průchodu nulou může dojít.
    2. Požadavek na přesnost lokalizace hrany v rovině předpokládá, že filtr bude reagovat pouze na body z blízkého okolí hrany.

    Předchozí dva požadavky jsou v protikladu.

Marrova teorie přichází s kompromisem (ale ne optimem, jak uvidíme dále), kterým je **lineární filtr, jehož koeficienty v konvoluční masce odpovídají 2D gaussovskému rozložení**

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (3.8)$$

kde *x, y* jsou souřadnice v obraze, σ je střední kvadratická odchylka, která jediným parametrem udává, velikost okolí na kterém filtr aplikujeme. Pixely blíže středu mají při filtraci větší váhu a vliv pixelů vzdálenějších než 3σ je zanedbatelný.

Naším cílem je získat druhou derivaci obrazové funkce.
Konvoluci znázorníme pomocí symbolu *. Díky filtraci Gaussiánem připustíme rozmazání obrazové funkce, tj. *G(x, y, σ) * f(x, y)*.

Pro odhad druhé derivace můžeme použít všesměrový Laplacián ∇^2. Tento postup je některými autory nazýván jako LoG operátor (angl. Laplacian of Gaussian), *∇^2 [G(x, y, σ) * f(x, y)]*.

Díky linearitě zúčastněných operací lze zaměnit pořadí derivace a konvoluce, což je důležitý krok, a získat

$$\nabla^2(G(x,y,\sigma) * f(x,y)) = (\nabla^2 G(x,y,\sigma)) * f(x,y) \qquad (3.9)$$

Hodnoty derivace Gaussiánu ∇^2 G lze přepočítat analyticky, protože na konkrétním obraze nezávisí! Pro jednodušší derivaci, nahradíme $x^2 + y^2 = r^2$, kde r je Euklidovská vzdálenost od středu Gaussiánu. Substituce je přípustná, protože Gaussián je středově symetrický. Gaussián nyní zapíšeme jako funkci jedné proměnné r,

$$G(r) = e^{-\frac{r^2}{2\sigma^2}} \qquad (3.10)$$

První derivace G'(r) je

$$G'(r) = -\frac{1}{\sigma^2} r\, e^{-\frac{r^2}{2\sigma^2}} \qquad (3.11)$$

Druhá derivace G'' (r), tj. Laplacián Gaussiánu, je

$$G''(r) = \frac{1}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right) e^{-\frac{r^2}{2\sigma^2}} \qquad (3.12)$$

Když se zpětnou substitucí vrátíme k původním souřadnicím x, y a zavedeme **normalizační koeficient c** zajišťující, aby součet všech koeficientů v masce byl 0, dostaneme vztah poskytující

hodnoty v konvoluční masce

$$h(x,y) = c\left(\frac{x^2+y^2-\sigma^2}{\sigma^4}\right)e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (3.13)$$

Díky svému tvaru se masce říká „**mexický klobouk**“. Ukažme si, jak vypadá aproximace operátoru LoG v masce 5 x 5

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

- Při implementaci detektoru založeného na hledání průchodů nulou je třeba se **vyhnout** naivnímu řešení pomocí **prahování LoG obrazu v intervalu hodnot blízkých k nule**. Výsledkem by byly hodně nespojité hrany.
- Lepší je použít opravdový detektor průchodů nulou, např. v masce 2 x 2 s reprezentativním bodem třeba v levém horním rohu. Hrana se zde indikuje tehdy, pokud se uvnitř okna opravdu mění znaménko.

Operátor $\nabla^2 G$, tj. Laplacián Gaussiánu, lze efektivně aproximovat pomocí diference dvou obrazů, které vznikly konvolucí s Gaussiánem o různém σ. Operátor bývá označován zkratkou DoG z angl. Difference of Gaussians.

Ukažme si chování různých modifikací detektorů hran opírajících se o hledání průchodu druhé derivace obrazové funkce nulou na příkladech na Obr. 3.9.

Obr. 3.9 Hledání hran jako průchodu druhé derivace obrazové funkce nulou, výchozí obrázek je výřezem (a) Použit operátor DoG, tj. diference Gausiánů, σ1 = 0, 1, σ2 = 0, 09 . (b) Průchody nulou DoG obrazu. (c) Odstraněny hrany, kterým neodpovídaly významnější první derivace. (d) LoG, Laplacián Gaussiánů, σ = 0, 2 po odstranění hran bez významnějších prvních derivací. Všimněte si různého měřítka hran oproti (b), (c) díky jinému σ.[2]

Jak je vidět z obrazů, mají tradiční operátory LoG a DoG a následné hledání průchodů nulou také **nevýhody**:

- Za prvé příliš vyhlazují ostré tvary, například ostré rohy se ztrácejí.
- Za druhé se snaží spojovat hrany do uzavřených křivek. Posměšně se výsledné hrany někdy označují jako „talíř špaget", i když se nekříží. Tato druhá vlastnost byla v původních pracích označována za výhodu, ale v mnoha aplikacích se ukázala spíše nevýhodou.

Shrnutí pojmů

K **detekci nespojitostí** v obraze se nejčastěji používá aplikace **konvoluční masky**.
Detekce bodů je realizována rozdílem mezi středovým bodem a okolím. Maska pro detekci bodů má tvar hornopropustného filtru. V místě konstantního jasu v obraze je odezva masky nulová. (Součet koeficientů masky je roven 0).
Princip **detekce čar** je stejný jako detekce samostatných bodů. Můžeme použít masky, které vybírají čáry jen v určitém směru. Pokud chceme najít nejsilnější čáry můžeme použít všechny masky individuálně pro daný obrázek.
Hrana v obraze je dána vlastnostmi obrazového elementu a jeho okolí.

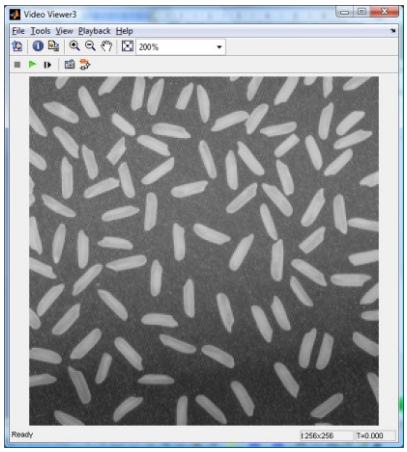
Otázky

1. Jak by jste definovali hranu z hlediska jasové funkce?
2. Jakým způsobem se detekují hrany v obraze?
3. Jakým frekvencím ve frekvenčním spektru odpovídají hrany?

CVIČENÍ

Detekce hran v obraze

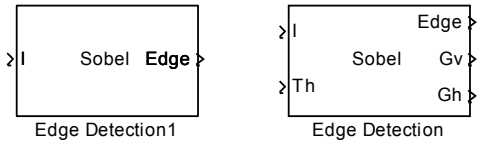
Detekce hran slouží k nalezení objektů v obraze. Hrana je reprezentována jako ostrý přechod jasové funkce. V jednom obraze se může nacházet samozřejmě několik takovýchto přechodů, tedy hran. K identifikaci slouží různé hranové detektory, které jsou citlivé na šum v obraze. Matematicky lze k hranové detekci použít operaci derivace nebo určení gradientu. Použitím těchto matematických metod však dochází k zintenzivnění nežádoucího šumu v obraze. Proto nezřídka bývá šum z obrazu před samotnou hranovou detekcí filtrován.
V průběhu cvičení budeme pracovat obrázkem, na kterém si postupně ukážeme základní čtyři metody zpracování detekce hran v obraze. Těmi jsou Sobelova, Prewittové, Robertsova a Cannyho metoda. Jako vstupní obrázek je zapotřebí použít snímek ve stupních šedi. V případě barevného snímku je zapotřebí nejdříve obrázek do stupně šedi převést. [3]



Obr. 3.10 Originál obrázku se kterým budeme pracovat v průběhu cvičení [3].

Detekce hran v obraze

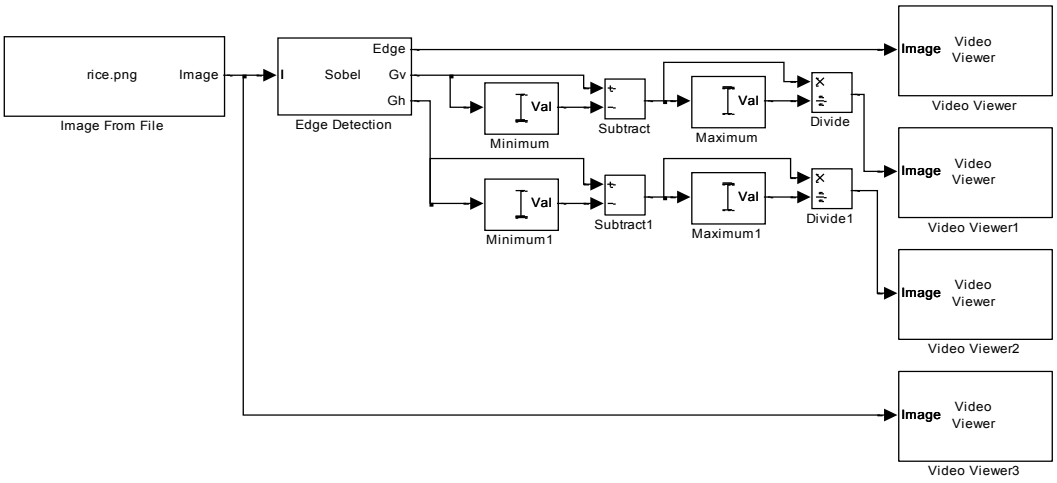
Pro nalezení hran v obraze je možné použít jednu ze čtyř základních metod. Sobelovu, Prewittové, Robertsovu nebo Cannyho metodu. Výstupem bloku Edge detection je binární obraz, tzv. matice Boolean hodnot, ve kterých hodnota pixelu = 1 koresponduje s hranou objektu. Alternativou jsou dva gradientní výstupy pro první tři metody (Obr. 3.11).



Obr. 3.11 Blok Edge detection, vlevo pro jedno vícerozměrné výstupní pole, vpravo pro matici výstupních složek.

Blok Edge Detection obsahuje několik základních parametrů, které také udávají použitou metodu. Parametr Output type určuje formát výstupu. Když pracujeme s binárním obrázkem, tak je na výstupu matice boolean hodnot. Nenulové hodnoty v této matici odpovídají hranám a nulové hodnoty reprezentují pozadí. Je možné také použití také gradientní výstup spolu s horizontálním a vertikálním určením hran. U metod Sobel nebo Prewitt jsou horizontální a vertikální hrany označené porty GV a GH. U metody Roberts jsou výstupní horizontální a vertikální složky zobrazeny v úhlech 45 a 135 stupňů a výstupní porty se tedy značí G45 a G135. U metody Canny je zapotřebí zadat jak vstupní tak výstupní parametry jako binární obraz.

Schéma modelu sestavte dle obrázku (Obr. 3.12).



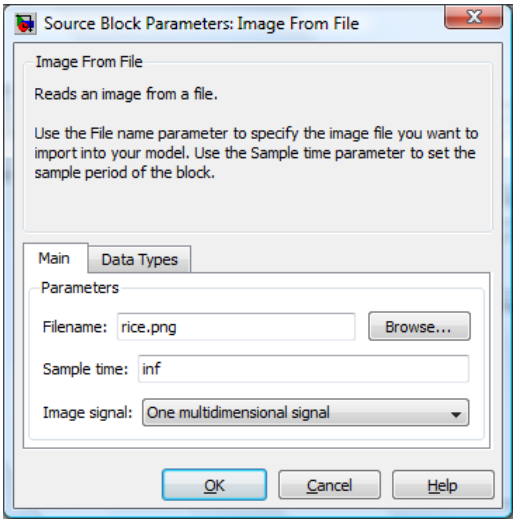
Obr. 3.12 Blokové schéma pro detekci hran v obraze.

Jednotlivé bloky naleznete:

- Blok ***Edge Detection*** nalezneme v menu ***Video and Image processing Blockset -> Analysis and Enhacement***
- Blok ***Minimum*** a ***Maximum*** nalezneme v menu ***Video and Image processing Blockset -> Statistic***
- Bloky ***Subtract*** a ***Divide*** nalezneme v menu ***Simulink -> Math Operation***

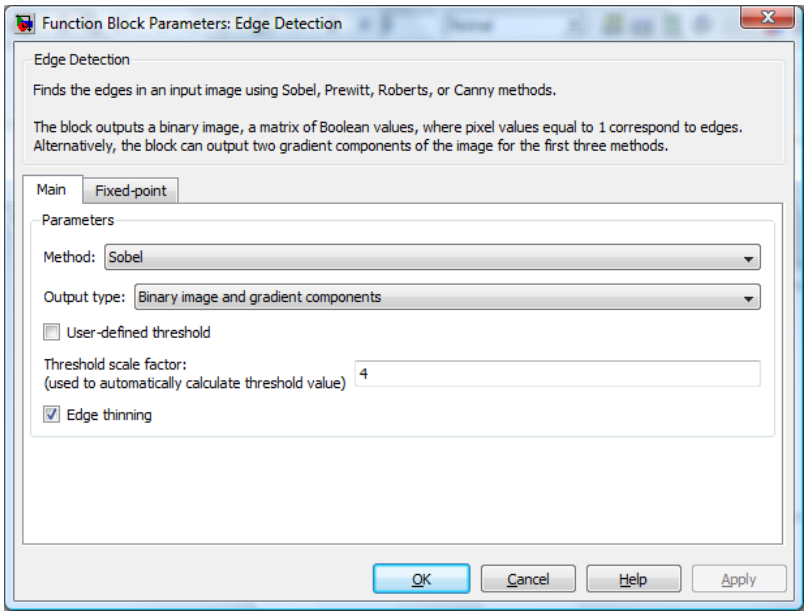
Nastavení vstupních parametrů:

- Parametry bloku ***Image From File*** nastavíme podle obrázku Obr. 3.13. V záložce ***Main*** nastavíme cestu k obrázku pomocí prohlížeče v poli ***Filename***. Použijeme nastavení parametru ***Image signal*** na hodnotu ***One multidimesiona signal.*** Hodnotu parametru ***Sample time*** nastavíme na ***inf***. V záložce ***Data Types*** provedeme nastavení parametru ***Output data type*** na hodnotu ***Single***.



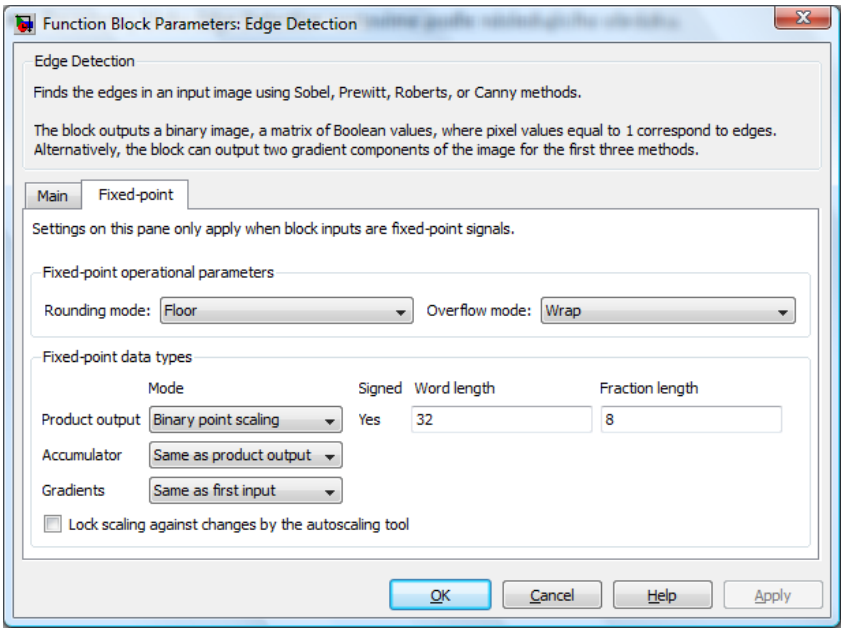
Obr. 3.13 Nastavení vstupních parametrů bloku Image From File.

- Parametry bloku ***Edge Detection*** nastavíme podle obrázku Obr. 3.14. V záložce ***Main*** nastavujeme metodu filtrace pomocí parametru ***Method***. Zde máme na výběr ze čtyř základních filtrů a to Sobel, Roberts, Prewits a Canny. Pro naši aplikaci použijeme nastavení hodnoty filtru na ***Sobel***. Parametr ***Output type*** nastavte na hodnotu ***Binary image and gradient components*** a zaškrtněte ***check box*** s parametrem ***Edge thinning***.



Obr. 3.14 Nastavení vstupních parametrů bloku Edge Detection v záložce Main.

- V záložce Fixed-point ponecháme původní nastavení parametrů, viz obrázek Obr. 3.15.



Obr. 3.15 Nastavení vstupních parametrů bloku Edge Detection v záložce Edge Detection.

- Parametry bloku ***Video Viewer block*** nastavíme podle následujícího postupu. V menu ***Main*** nastavujeme metodu filtrace pomocí parametru ***Method***. Nastavte jej na hodnotu ***Sobel***. Parametr ***Output type*** nastavte na hodnotu ***Binary image and gradient components*** a zaškrtněte ***check box*** s parametrem ***Edge thinning***.

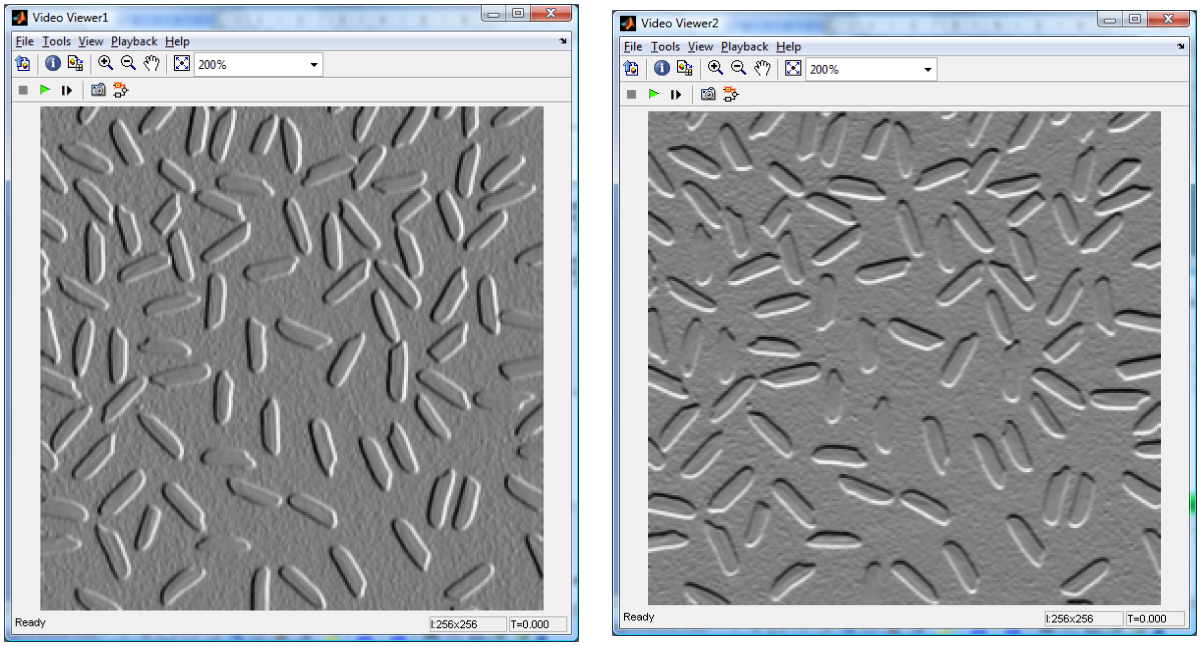
- Parametry bloku ***Minimum*** nastavte v záložce ***Main***. Parametr ***Mode*** nastavte na hodnotu ***Value***. Úkolem tohoto bloku je vyhledat minimální hodnoty v GV a GH maticích. Stejné nastavení proveďte i u bloku ***Maximum***, jehož úkolem je vyhledat maximální hodnoty v GV a GH maticích. Zde se hodnota pohybuje v rozmezí 0 - 1.

- Parametry bloků ***Subtract*** a ***Devide*** ponechte defaultně nastaveny.

- Parametry bloků Video Viewer ponechte defaultně nastaveny.

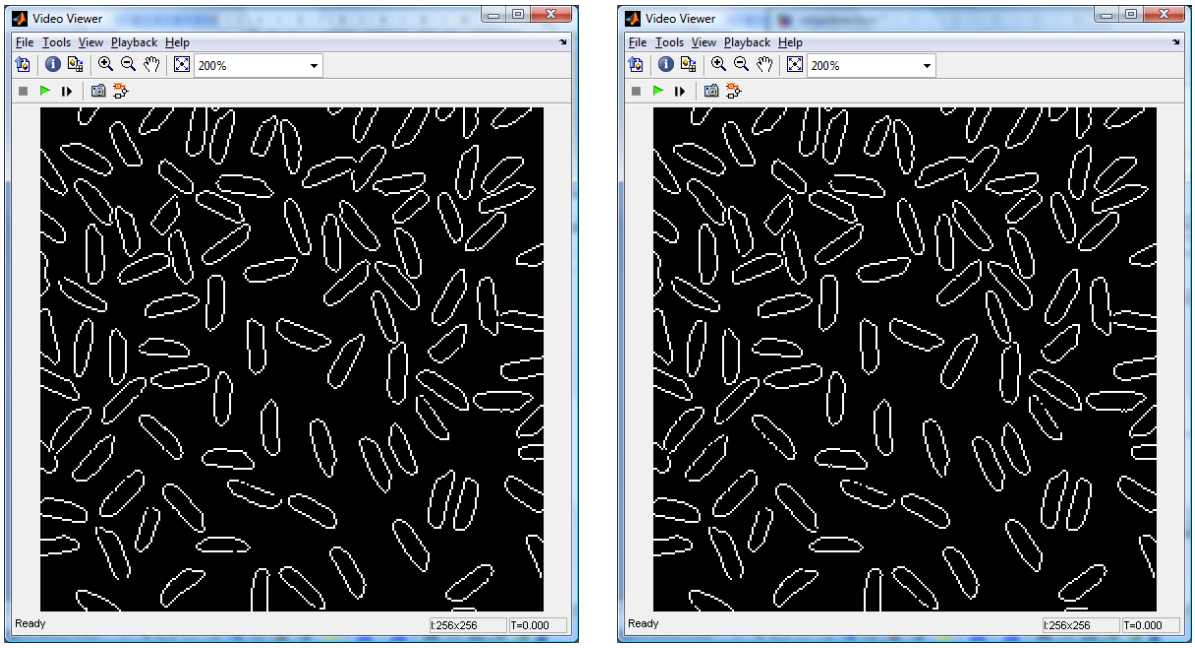
- Nezapomeňte nastavit konfigurační parametry. V menu Simulation -> Configuration Parameter nastavte parametr Stop time = 0, Type = Fised-step a Solver = Discrete (no continuous states).

Nyní propojte jednotlivé bloky dle schématu na Obr. 3.12, nastavte parametry a spusťte simulaci.

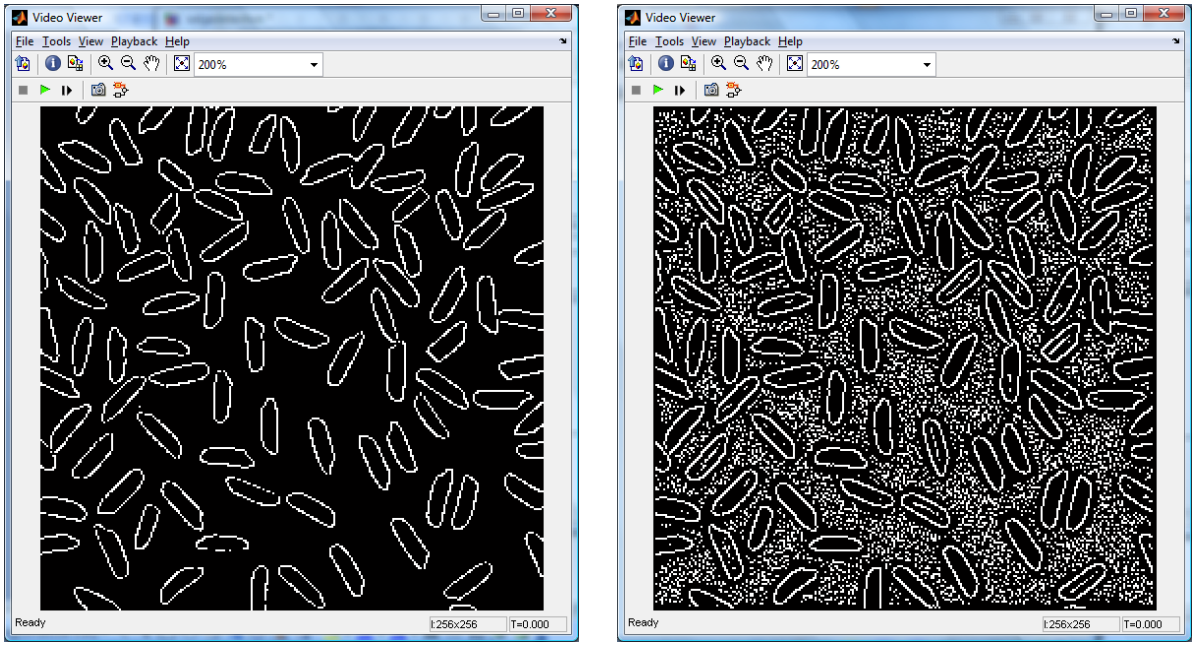


Obr. 3.16 Vlevo vertikální zvýraznění hran, vpravo horizontální zvýraznění hran v obraze. U obou snímků užito Sobelovy metody.

Jednotlivé metody zpracování detekce hran se mezi sebou liší.



Obr. 3.17 Vlevo Sobelova metoda detekce hran v obraze, vpravo metoda Prewittové.

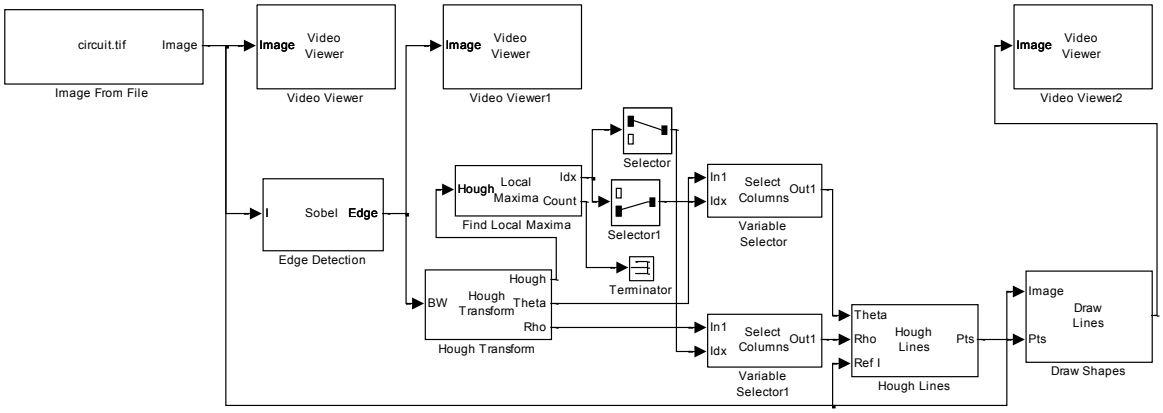


Obr. 3.18 Vlevo Robertsova metoda detekce hran v obraze, vpravo metoda Cannyho.

Hledání linií v obraze

Hledání linií v obraze slouží k detekci, měření a rozpoznávání objektů v obraze. Tato aplikace je rozšířením základních poznatků o detekci hran v obraze.

Schéma modelu nakreslete dle obrázku Obr. 3.19.



Obr. 3.19 Blokové schéma modelu pro liniovou detekci.

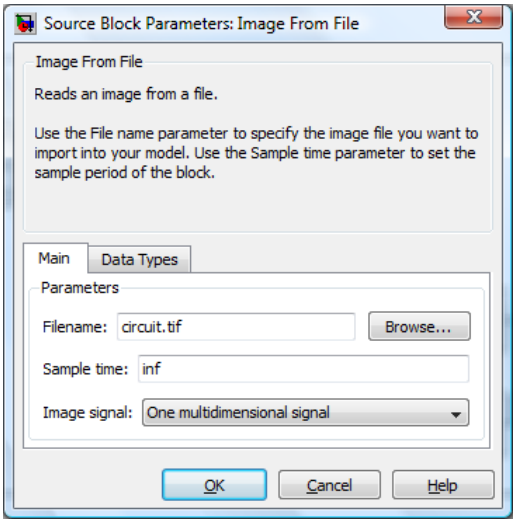
Jednotlivé bloky naleznete:

- Blok ***Edge Detection*** nalezneme v menu ***Video and Image processing Blockset -> Analysis and Enhacement***
- Blok ***Image from File*** nalezneme v menu ***Video and Image processing Blockset -> Sources***
- Bloky ***Hought Transform*** a ***Hought Lines*** nalezneme v menu ***Video and Image processing Blockset -> Transforms***

- Blok ***Find Local Maxima*** nalezneme v menu ***Video and Image processing Blockset -> Statistic***
- Blok ***Draw Shapes*** nalezneme v menu ***Video and Image processing Blockset -> Text &Graphics***
- Blok ***Video Viewer*** nalezneme v menu ***Video and Image processing Blockset -> Sinks***
- Blok ***Variable Selector*** nalezneme v menu ***Signal processing Blockset -> Signal Management -> Indexing***
- Blok ***Selector*** nalezneme v menu ***Simulink -> Signal Routing***
- Blok ***Terminator*** nalezneme v menu ***Simulink -> Sinks***

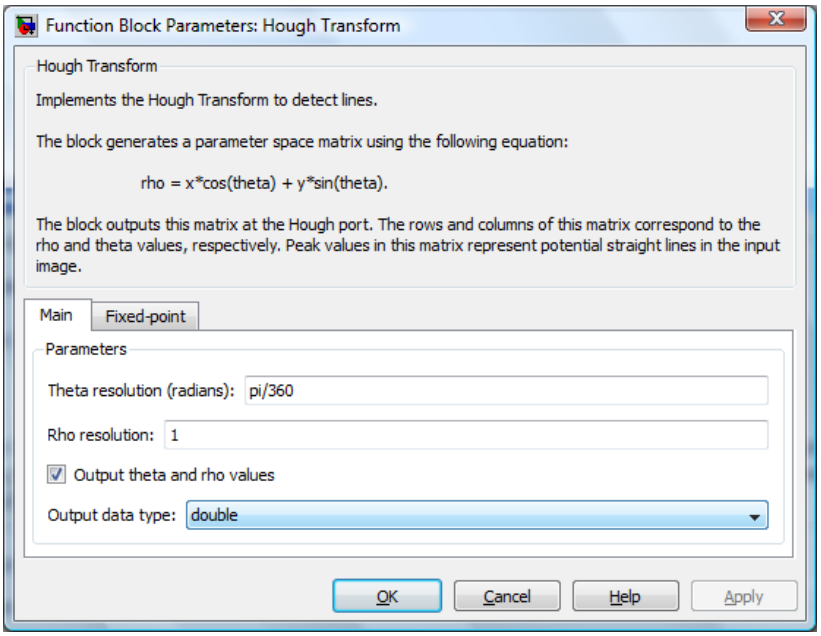
Nastavení vstupních parametrů:

- Parametry bloku ***Image From File*** nastavíme podle obrázku Obr. 3.20. V záložce ***Main*** nastavíme cestu k obrázku pomocí prohlížeče v poli ***Filename***. Použijeme nastavení parametru ***Image signal*** na hodnotu ***One multidimesiona signal.*** Hodnotu parametru ***Sample time*** nastavíme na ***0***. V záložce ***Data Types*** provedeme nastavení parametru ***Output data type*** na hodnotu ***One multidimensional signal***.



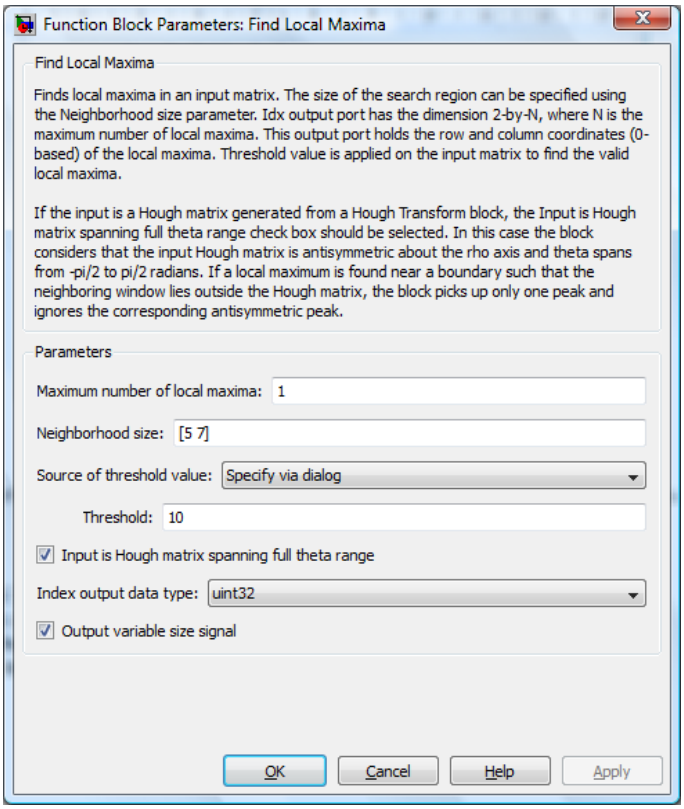
Obr. 3.20 Nastavení bloku Image From File v záložce Main.

- Parametry bloku ***Edge detection*** necháme defaultně přednastaveny.
- Parametry bloku ***Hough Lines*** ponecháme defaultně přednastaveny.
- Parametry bloků ***Video Viewer*** ponecháme defaultně přednastaveny.
- Parametry bloku ***Hough Transform*** nastavíme dle obrázku Obr. 3.21. Parametr **Theta resolution (radians)** nastavíme na hodnotu **PI/360**, parametr **Rho resolution** ponecháme na hodnotě **1** a zaškrtneme Checkbox **Output theta and rho values**. Parametr **Output data type** ponecháme na hodnotě **double**.



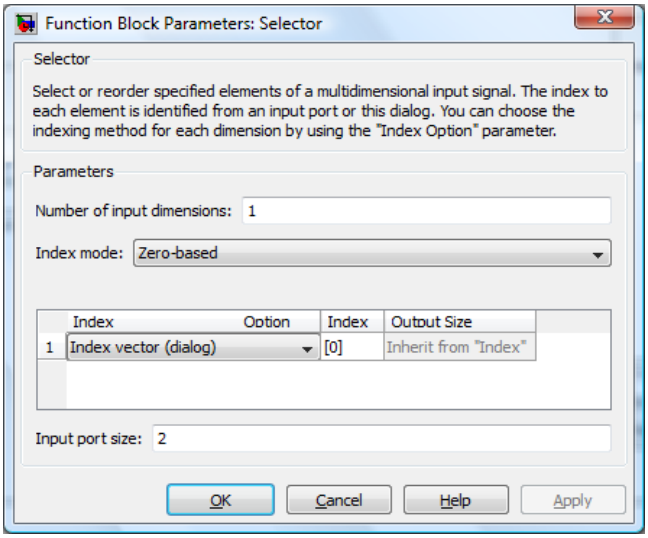
Obr. 3.21 Nastavení parametrů bloku Hough Transform v záložce Main.

- Parametry bloku ***Find Local Maxima block*** nastavíme dle obrázku Obr. 3.22. Parametr **Maximum number of local maxima (N)** nastavíme na hodnotu **1**. Zaškrtneme checkbox **Input is hough matrix spanning full theta rande**. Ostatní parametry můžeme ponechat přednastavené.



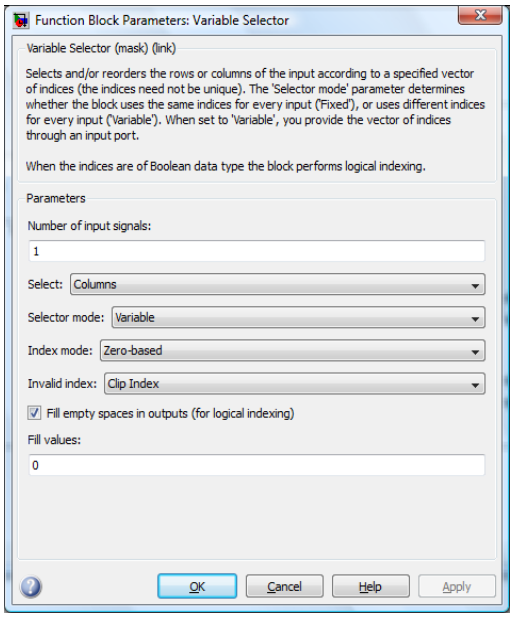
Obr. 3.22 Nastavení parametrů bloku Find Local Maxima.

- Parametry bloku ***Selector*** nastavíme dle obrázku Obr. 3.23. Parametr **Number of input dimesions** na hodnotu **1**, Parametr **Index mode** nastavíme na hodnotu **Zero-based**. Dále v **Index Option** nastavíme hodnotu **Index vector (dialog)**, **index = 0** a **Input port size** na hodnotu **2**. Parametry bloku ***Selector1*** obdobně. Parametr **Number of input dimesions** na hodnotu **1**, Parametr **Index mode** nastavíme na hodnotu **Zero-based**. Dále v **Index Option** nastavíme hodnotu **Index vector (dialog)**, **index = 1** a **Input port size** na hodnotu **2**.



Obr. 3.23 Nastavení parametrů bloku selector.

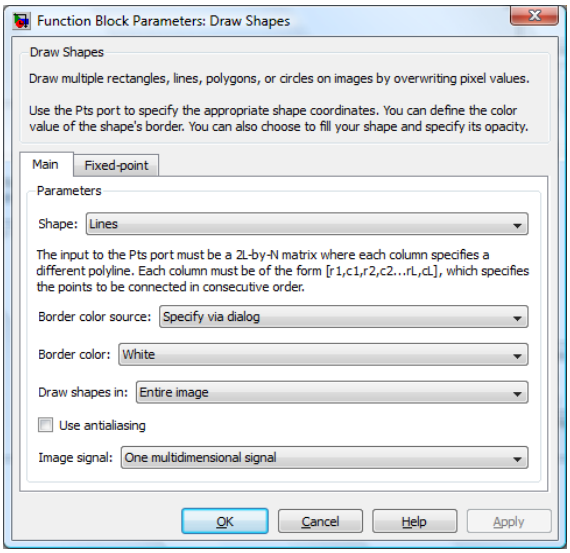
- Parametry bloku ***Variable Selector*** nastavíme dle obrázku Obr. 3.24. Parametr **Select** nastavíme na hodnotu **Columns** a parametr **Index mode** na hodnotu **Zero-based**. Ostatní parametry ponecháme přednastavené.



Obr. 3.24 Nastavení parametrů bloku Variable Selector.

- Parametry bloku ***Houh lines block*** nastavíme v záložce parametr **Sine value computation method** na **Trigonometric function**.

- Parametry bloku ***Draw Shapes*** nastavíme dle obrázku Obr. 3.25. V záložce main nastavíme parametr **Shape** na hodnotu **Lines** a parametr **Border Color** na hodnotu **White**. Ostatní parametry ponecháme přednastaveny.



Obr. 3.25 Nastavení parametrů bloku Draw Shapes v záložce Main.

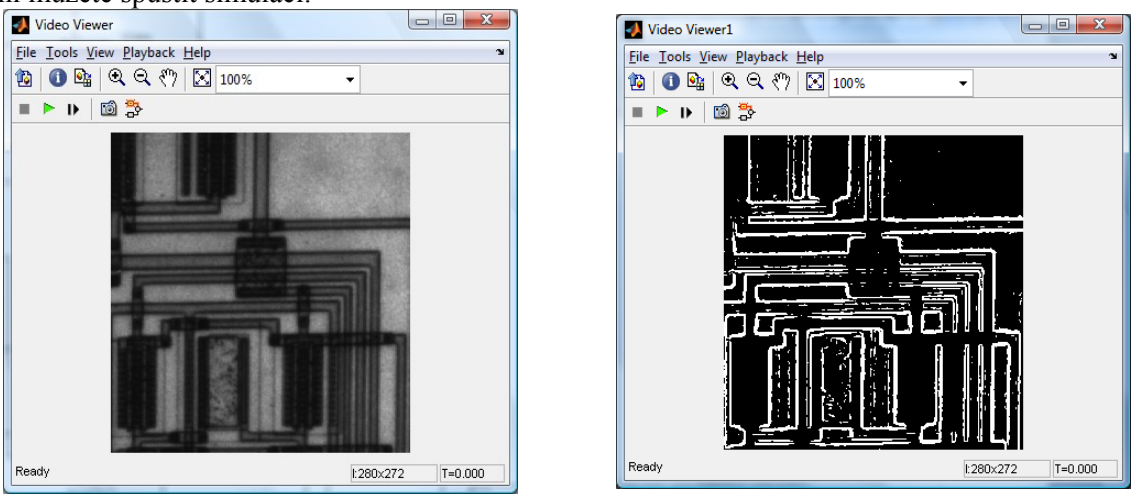
Nyní bloky spojíme dle schématu na Obr. 3.19.

Nastavení vstupních parametrů pro simulaci:

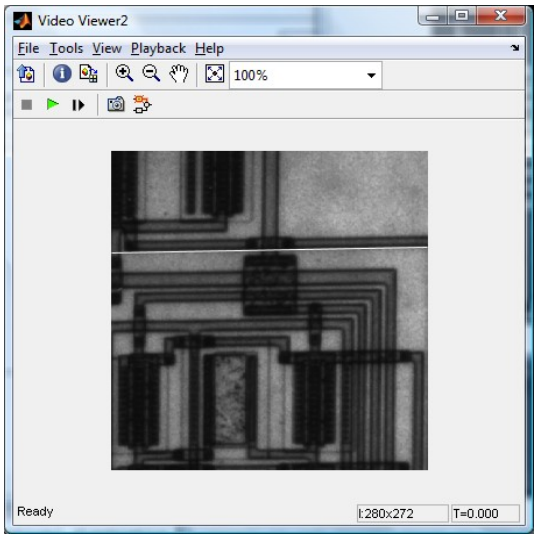
- Před spuštěním simulace nastavíme konfigurační parametry Simulinku. Nastavení provedeme v menu ***Simulation -> Configuration parameters -> Simulation***

- Parametr **Stop time** nastavíme na hodnotu **0**, parametr **Type** na hodnotu **Fixed step** a parametr **Solver** na hodnotu **discrete (no continuous states)**.

Nyní můžete spustit simulaci.



Obr. 3.26 Vlevo původní obrázek, vpravo nalezené hrany pro průchodem hranového detektoru.



Obr. 3.27 Nalezená linie v obraze.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem *03 Detekce_hran.swf*.

Demonstrační video k této kapitole je k dispozici v adresáři Videa pod názvem *video_detekce_hran.mpg*.

4. PRAHOVÁNÍ, SEGMENTACE OBRAZU

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat základní postupy pro segmentaci obrazu.
- popsat metodiku zpracování jednotlivých typů segmentace.
- vyřešit jednoduché úlohy v Matlab/Simulink a VIP pro zpracování obrazu a jeho segmentaci.

Výklad

4.1 Úvod

Hlavním úkolem digitálního zpracování obrazu je aplikace metod a funkcí k nalezení nebo vytvoření těch parametrů obrazu (nebo jejich částí), které jsou naším zájmem, tzn. naším cílem není zpracovávat obraz jako celek, ale většinou se soustředíme na jeho vybranou část. Prvotním cílem je tedy extrakce objektů, které se v obraze nacházejí. Tento proces, ve kterém jsou objekty separovány od nezajímavého pozadí se nazývá **segmentace obrazu**.
Segmentace obrazu bývá nejčastěji založena na principech detekce hran ohraničujících jednotlivé objekty nebo na detekci celých oblastí, kterými jsou jednotlivé objekty v obraze reprezentovány.

4.2 Segmentace obrazu

Jak již bylo uvedeno, **segmentace** je proces extrakce, v němž jsou objekty separovány od nezajímavého pozadí. Segmentací obrazu rozumíme:

- detekci hran ohraničující jednotlivé objekty
- detekci celých oblastí, kterými jsou jednotlivé objekty v obraze reprezentovány.

Segmentace obrazu je jedním z nejdůležitějších kroků vedoucích k analýze obsahu zpracovávaných obrazových dat. Snahou segmentace obrazu je rozčlenit obraz do částí, které mají úzkou souvislost s předměty či oblastmi reálného světa zachyceného na obraze. Výsledkem segmentace má být soubor vzájemně se nepřekrývajících oblastí, které buď jednoznačně korespondují s objekty vstupního obrazu, pak jde o kompletní segmentaci, nebo vytvořené segmenty nemusí přímo souhlasit s objekty obrazu a pak jde o částečnou segmentaci. Častá je situace, kdy je obraz tvořen kontrastními objekty na pozadí neměnného jasu - např. nedotýkající se chromozómy, krevní buňky, psaný text apod. Tam lze užít jednoduché globální postupy a dosáhnout kompletní segmentace obrazu na objekty a pozadí. Takové postupy nezávisejí na kontextu, neužívá se žádný model zpracovávané oblasti, k řízení procesu segmentace nepřispívají znalosti výsledné oblasti.

Při částečné segmentaci je výsledkem rozdělení obrazu do samostatných částí, které jsou homogenní vzhledem k určitým zvoleným vlastnostem, jako jsou jas, barva, odrazivost, textura. Po zpracování je

výsledkem seznam oblastí, které jsou homogenní v jistých zvolených rysech. Oblasti se obecně mohou překrývat. Na data popisující částečnou segmentaci je nezbytné aplikovat další postupy, které pomocí vyšší úrovně umožní získat výslednou segmentaci obrazu.

Dokonalá nebo správná segmentace složitých obrazů není v této fázi zpracování dosažitelná. Rozumným cílem je zisk částečné segmentace, jejíž výsledky mohou být zpřesněny při následném zpracování operacemi vyšších úrovní. Okamžitým přínosem segmentace je výrazná redukce objemu zpracovávaných dat. Jedním z hlavních problémů ovlivňujících segmentaci je nejednoznačnost obrazových dat, často doprovázená informačním šumem. Podle dominantní vlastnosti, které je pro segmentaci využíváno, lze metody rozdělit do tří skupin:

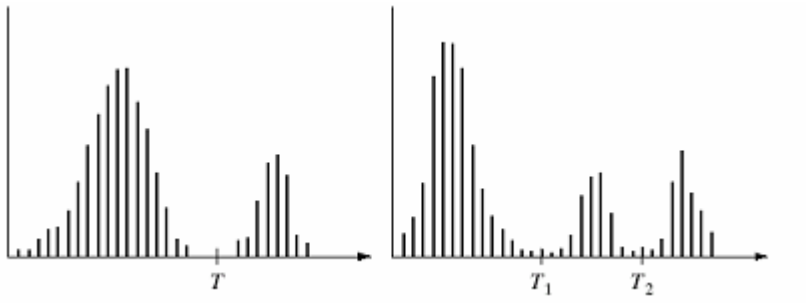
1. Metody využívají globální znalosti obrazu (nebo jeho části) reprezentované obvykle histogramem určitých vlastností.
2. Postupy vycházející z určování hranic mezi oblastmi (částmi) obrazu.
3. Postupy přímo vytvářející tyto oblasti (jejich části).

Není přitom podstatné, z jakých charakteristik (jas, textura, rychlostní pole apod.) při hledání hranic nebo při tvorbě oblastí vycházíme. Každá oblast je jednoznačně reprezentována svou hranicí a každá uzavřená hranice jednoznačně vypovídá o oblasti, kterou obemyká. V důsledku odlišného charakteru obrazových dat i odlišných algoritmů tvorby hranic a oblastí přinášejí všechny tři skupiny metod poněkud rozdílné segmentační výsledky. Výsledky postupů všech tří skupin lze proto kombinovat a vytvořit jedinou popisnou strukturu.

4.3 Prahování

Prahování patří k nejstarší a nejjednodušší metodě segmentace obrazu. I když má svá široká omezení co se týká nastavení a parametrů, řadí se k široce používané metodě. Výhodou této metody je její jednoduchost a tím pádem snadná implementace a časová nenáročnost.
Princip **prahování** a metody na něm založené spočívají v tom, že objekty a pozadí mají jinou úroveň intenzity jasu. Stačí tudíž určit tuto rozdílovou úroveň (práh) a poté každý pixel, který má menší hodnotu než zvolený práh, je určen jako pixel pozadí a všechny ostatní pixely jako pixely objektu, který chceme segmentovat.

Na Obr. 4.1a) je histogram odpovídající obrázku, složenému ze světlých bodů, které tvoří objekt a tmavých bodů, které tvoří pozadí. V takovémto případě jsou pixely objektu a pozadí seskupeny do dvou dominantních modulů. Jednou z možností je vyextrahovat objekty z pozadí výběrem prahovací úrovně T, která oddělí oba moduly. Potom se bod (x, y) pro který platí, že $f(x, y) > T$ nazývá bodem objektu, jinak se jedná o bod pozadí.



a) b)

Obr. 4.1 Histogram obrázku. a) jednoduché prahování, b) víceúrovňové prahování.

Na Obr. 4.1b) je znázorněn obecnější případ, kde máme tři dominantní moduly v histogramu (např. dva světlé objekty a tmavé pozadí). Zde se bude jednat o víceúrovňové prahování. Bod *(x, y)* patří jednomu objektu pokud $T_1 < f(x,y) \le T_2$, druhému objektu pokud $f(x,y) > T_2$ a pozadí pokud $f(x,y) \le T_1$. Častěji je tento problém řešen metodou narůstání oblasti.

Prahová hodnota *T* může být definována jako T = *T[x, y, p(x, y), f(x, y)],* kde *f(x, y)* je úroveň šedi bodu *(x, y)* a *p(x, y)* označuje nějakou lokální vlastnost tohoto bodu, např. střední úroveň šedi okolí kolem bodu *(x, y)*.

Obecně je prahovaný obrázek *g(x, y)* definován jako

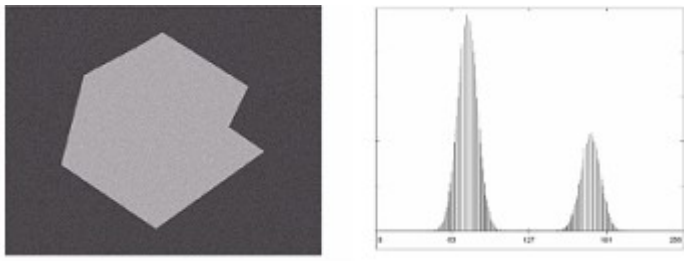
$$g(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) \le T \end{cases} \tag{4.1}$$

Pixely označené 1 odpovídají objektům, zatímco pixely označené 0 odpovídají pozadí.

- Pokud *T* závisí pouze na *f(x, y)*, pak se jedná o **globální prahování**.
- Pokud *T* závisí na *f(x, y) a p(x, y),* pak se jedná o **lokální prahování**.
- Pokud *T* závisí na prostorových souřadnicích *x* a *y*, pak se jedná o **dynamické nebo-li adaptivní prahování**.

4.3.1 Úloha osvětlení

Uvažujme Obr. 4.2a) jemuž odpovídá histogram na Obr. 4.2b) Histogram je čistě bimodální a můžeme jednoduše určit prahovací úroveň *T* jako střed údolí mezi oběma kopci v histogramu.



a) b)

Obr. 4.2 Obrázek a jeho histogram. [1]

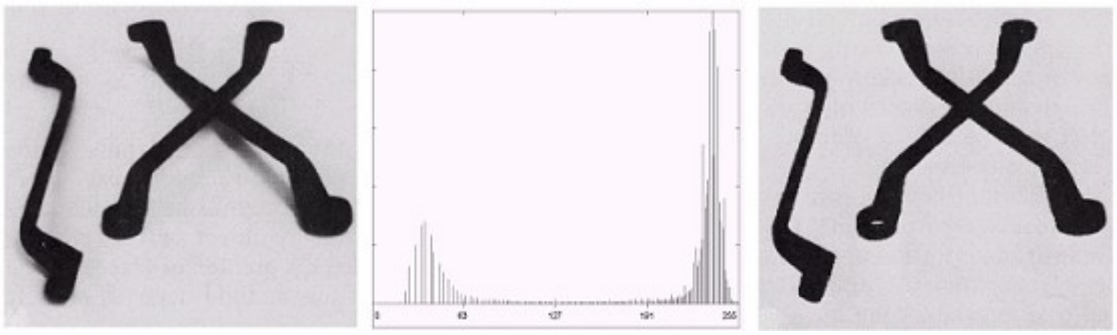
Nyní si představme, že Obr. 4.2a) vynásobíme s Obr. 4.3a) a vznikne nám obrázek Obr. 4.3b) Histogram tohoto obrázku je na Obr.4.3c). Můžete si všimnout, že původní údolí bylo eliminováno a provedení segmentace jednou prahovací úrovní je v podstatě nemožné.

a) b) c)

Obr. 4.3 Vliv osvětlení.[1]

4.3.2 Globální prahování

Jedná se o nejjednodušší případ použití jedné prahovací úrovně *T*. Segmentace je potom provedena tak, že se prochází pixel po pixelu a zjišťuje se zda se jedná o bod objektu nebo o bod pozadí, tzn. závisí to na tom zda je hodnota pixelu větší nebo menší než zvolená prahovací úroveň *T*.



a) b) c)

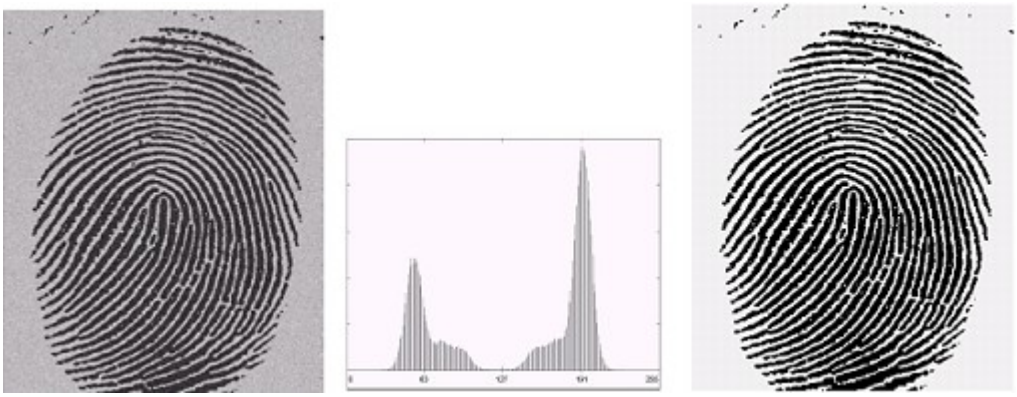
Obr. 4.4 Globální prahování. a) obrázek, b) histogram obrázku a), c) obrázek po prahování.[1]

Heuristický přístup je založen na vizuální inspekci histogramu. Prahovací úroveň T můžeme obdržet automaticky:

1. Výběr počátečního odhadu T.
2. Segmentace obrázku s použitím úrovně T. Získáme dvě skupiny pixelů: G_1 obsahující všechny pixely s úrovní šedi $>T$ a G_2 obsahující pixely s hodnotami $\leq T$.
3. Výpočet střední úrovně šedi μ_1 a μ_2 pro pixely oblastí G_1 a G_2.
4. Výpočet nové prahovací úrovně

$$T = \frac{1}{2}(\mu_1 + \mu_2) \tag{4.2}$$

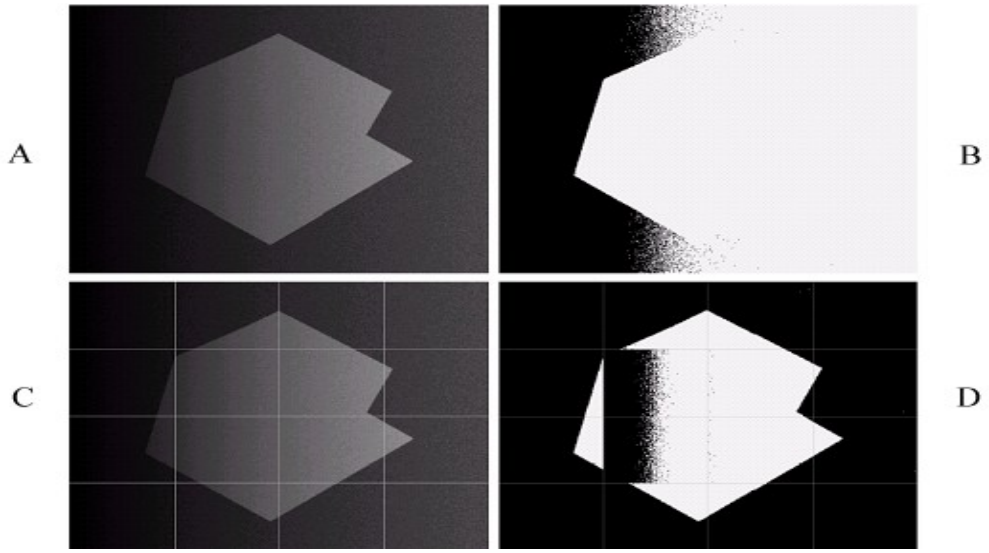
5. Opakování kroků 2 až 4 dokud není rozdíl v T v postupných iteracích menší než je předdefinovaný parametr T_0.



Obr. 4.5 Segmentace obrazu založená na odhadu prahovací úrovně. [1]

4.3.4 Adaptivní prahování

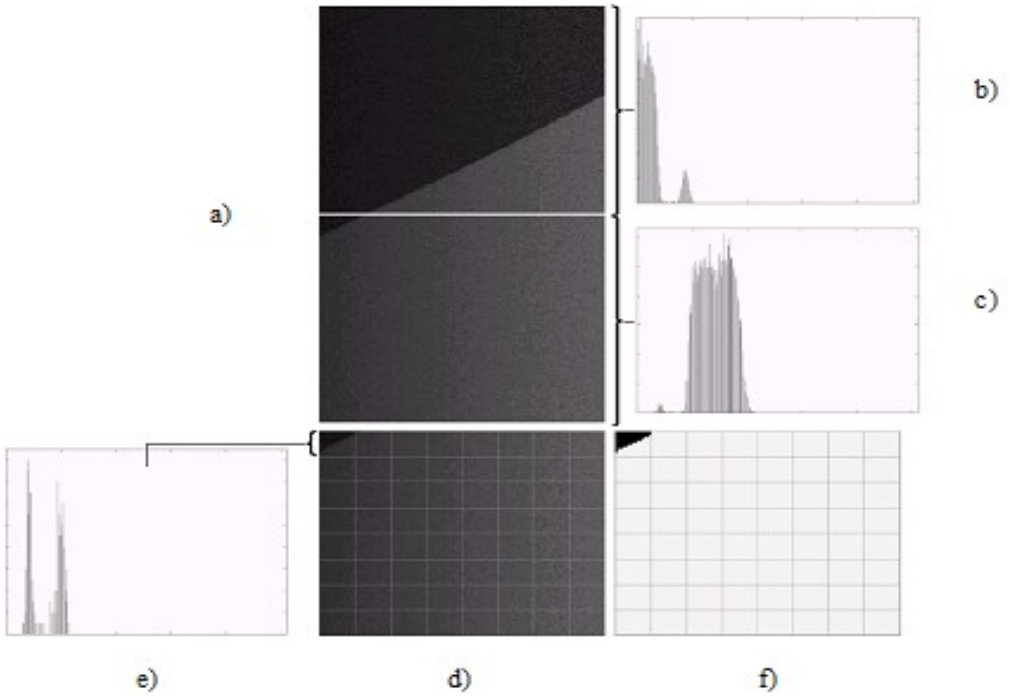
Jak bylo ukázáno na Obr. 4.3 b) nelze většinou použít jen jedinou prahovou úroveň. Jednou z možností je řešení této úlohy rozdělením původního obrázku na podobrázky, přičemž pro každý podobrázek se použije rozdílná prahovací úroveň. Základní otázkou je jak obrázek rozdělit a jak odhadnout prahovací úroveň pro každý podobrázek. Poněvadž prahovací úroveň použitá na každý pixel obrázku závisí na poloze bodu v obrázku, nazývá se tento způsob prahování **prahováním adaptivním**.



Obr. 4.6 Ukázka adaptivního prahování. A) původní obrázek. B) globální prahování. C) rozdělení a poobrázky. D) segmentace jednotlivých podobrázků. [1]

Na Obr. 4.6a) je původní obrázek. Obr. 4.6b) znázorňuje obrázek vytvořený globálním prahováním s prahovací úrovní uprostřed údolí mezi dvěma laloky histogramu. Jednou z možností eliminace vlivu nerovnoměrného osvětlení je rozdělení obrázku na menší podobrázky, tak že osvětlení každého podobrázku je téměř jednotné. Obr. 4.6c) ukazuje takovéto rozdělení. Obrázek byl rozdělen na čtyři stejné části a potom byla každá část rozdělena na další čtyři části. Na každou část obrázku byla aplikováno globální prahování s jednou prahovací úrovní, která byla vybrána jako střední bod mezi minimem a maximem v daném podobrázku. Výsledek této procedury je na Obr. 4.6d). s výjimkou dvou oblastí je evidentní zlepšení oproti segmentaci na Obr. 4.6b).

Na Obr.4.7 jsou vybrány nesprávně segmentované podobrázky. Na prvním obrázku je zobrazený histogram téměř bimodální (dva píky) s dobře rozlišitelnými píky a údolími. Na druhém obrázku je vykreslen histogram, který je téměř unimodální (jeden pík) s ne příliš jasným rozdílem mezi objektem a pozadím. Na obrázku 4.7d) je ukázán chybně segmentovaný podobrázek rozdělený na další podobrázky. 4.7e) pak ukazuje histogram horního podobrázku, který je opět bimodální a je tedy jednoduše segmentovatelný. Výsledek segmentace je na Obr. 4.7f). Z tohoto vyplývá, že je nutné zvolit vhodnou velikost podobrázku.



Obr. 4.7 Adaptivní prahování.[1]

4.3.5 Metoda narůstání oblasti

Tato metoda začíná pracovat s malými oblastmi, např. i s jednotlivými pixely, které postupně spojuje do větších oblastí. Dochází tak k narůstání oblastí. Proces se provádí tak dlouho, dokud v obraze existují oblasti, které lze spojovat.

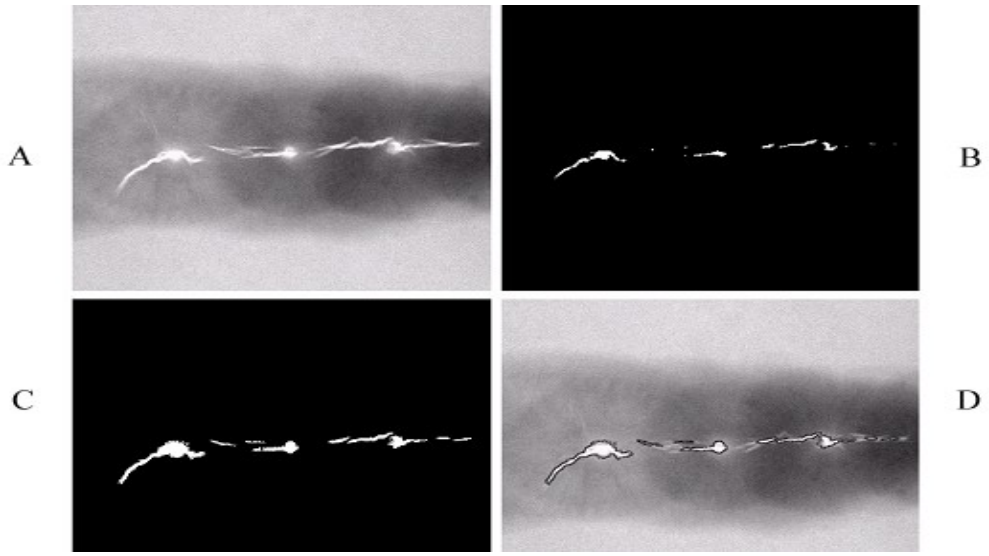
Postup řešení:
Nechť R reprezentuje celou obrazovou oblast. Segmentaci můžeme chápat jako proces, při kterém se R rozdělí na n podoblastí $R_1, R_2, R_3, \ldots, R_n$ a to tak, že

a) $\bigcup_{i=1}^{n} R_i = R$
b) R_i je souvislá oblast, i = 1, 2, …, n
c) $R_i \cap R_j = 0$ pro všechna $i \neq j$
d) $P(R_i) = \text{TRUE}$ pro i = 1, 2, …, n
e) $P(R_i \cup R_j) = \text{FALSE}$ pro $i \neq j$

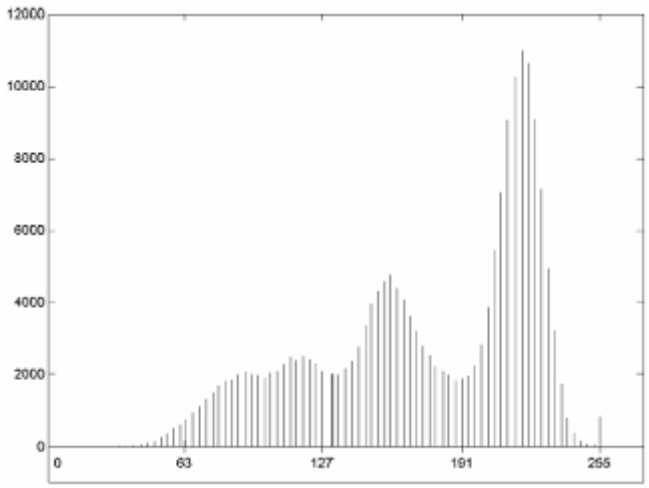
Kde $P(R_i)$ je logická operace definovaná na všechny body množiny R_i a 0 je prázdná množina.
Podmínka a) předpokládá, že každý pixel náleží nějaké oblasti.
Podmínka b) vyžaduje, aby body oblasti byly souvislé.
Podmínka c) signalizuje, že jednotlivé oblasti se nebudou prolínat.

Podmínka d) se zabývá vlastnostmi, které musí být splněny pro každý pixel v dané oblasti (např. pixely v oblasti R_i musejí mít stejnou úroveň šedi).
Podmínka e) říká, že oblasti R_i a R_j jsou nesouvislé ve smyslu vlastnosti P.

Proces narůstání oblastí začíná na malé základní množině bodů a na základě předdefinovaných kritérií (a - e) pak dochází k narůstání oblastí. Výběr počáteční množiny bodů je obvykle založen na přirozenosti úlohy. Výběr kritérií, podle kterých jsou jednotlivé pixely přiřazovány jednotlivým oblastem velmi často závisí na typu obrázku (barva, textura).



Obr. 4.8 Segmentace obrazu metodou narůstáním oblastí. [1]



Obr. 4.9 Histogram obrázku 4.8a) [1]

Obrázek Obr. 4.8. ukazuje rentgenový obrázek svaru obsahující několik trhlin. Použijeme metodu segmentace narůstáním oblastí k segmentaci poruch. Tyto znaky mohou být použity pro inspekci, databáze pro řízení automatizovaného svářecího systému a pro jiné aplikace.
Nejprve je nutné určit počáteční množinu bodů, ze kterých budeme vycházet. V této aplikaci víme, že pixely označující poruchu jsou světlé a mají maximální hodnotu (v tomto případě 255). Počáteční

množinu bodů vybereme na základě této informace. Všechny body, které budou mít hodnotu 255 budou považovány za počáteční body. Tyto body jsou ukázány na Obr. 4.8b)

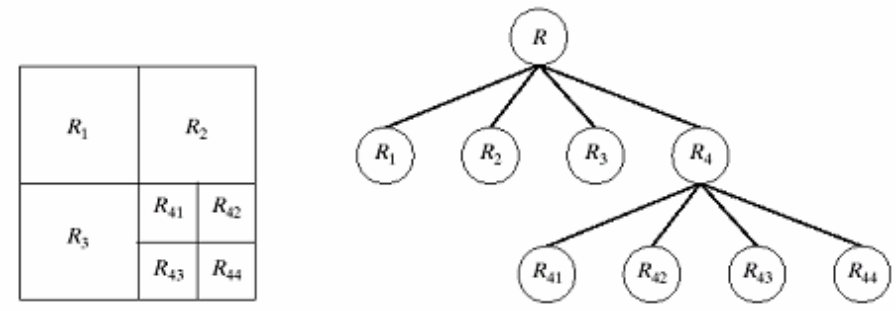
Dalším krokem je výběr kritéria pro narůstání oblasti. Zde vybereme dvě kritéria pro pixel náležící oblasti:

1) Absolutní rozdíl mezi úrovněmi šedi libovolného pixelu a počáteční množiny pixelů musí být menší než 65. Toto číslo je určeno z histogramu na Obr. 4.9. Jedná se o rozdíl mezi 255 a místem prvního údolí vlevo, které reprezentuje nejvyšší hodnotu pozadí.
2) Pixel bude náležet do oblasti pokud bude nejméně k jednomu bodu oblasti v 8-sousedství. Jestliže pixel bude v 8-sousedství s více než jednou oblastí, budou tyto oblasti spojeny.

Obr. 4.8c) ukazuje výsledek segmentace narůstáním oblastí. Pokud hranice těchto oblastí přidáme do původního obrázku jak to ukazuje Obr. 4.8d) vidíme, že tyto oblasti skutečně ukazují defekty svaru.

4.3.6 Metoda dělení oblastí

Tato metoda je opakem k metodě předchozí. Začíná se pracovat s jedinou oblastí tvořenou celým obrazem a dále se provádí její dělení na dílčí části. Proces se provádí tak dlouho, dokud není ve všech vzniklých oblastech dosaženo splnění kritéria homogenity.
Např. nechť R reprezentuje celou obrazovou oblast na níž platí tvrzení P. Jedním přístupem je segmentovat R dělením oblasti na menší a menší kvadranty tak, aby pro libovolnou oblast R_i platilo, že $P(R_i) = \text{TRUE}$. Začneme s celou obrazovou oblastí. Jestliže $P(R_i) = \text{FALSE}$ rozdělíme oblast na kvadranty. Jestliže P je FALSE pro libovolný kvadrant rozdělíme tento kvadrant na podkvadranty atd.



Obr. 4.10 Princip prahování pomocí metody dělení oblasti.

Postup:

1. Rozdělení do čtyř samostatných kvadrantů libovolné oblasti R_i, pro kterou $P(R_i) = \text{FALSE}$
2. Spojení přilehlých oblastí R_j a R_k pro které platí $P(R_j \cup R_k) = \text{TRUE}$.
3. Zastavení procesu dělení oblastí pokud další rozdělení nebo spojení oblastí není možné.

Σ Shrnutí pojmů

Segmentace je důležitý krok v předzpracování obrazu. Slouží k separování objektů od nezajímavého pozadí. K jednoduchým a základním postupům segmentace patří detekce oblastí pomocí prahování. K tomuto se dá výhodně využít histogram, který nám udává rozložení jednotlivých jasových složek v obrazu.
Prahování je proces kdy na základě např. histogramu obrazu zvolíme práh (číselná hodnota jasu). Veškeré pixely které leží nad touto hranicí patří do objektu našeho zájmu a veškeré pixely které leží pod touto hranicí patří do nezajímavého pozadí.

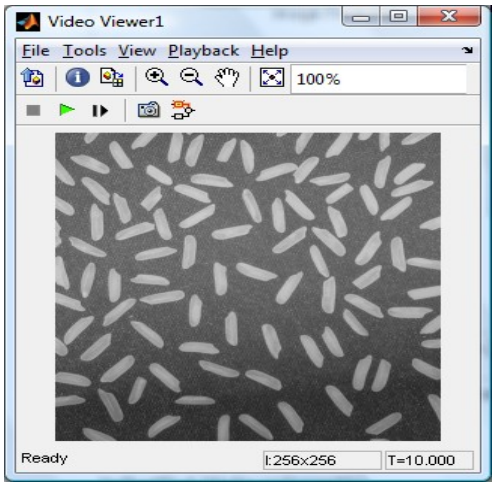
Otázky

1. Co je to segmentace obrazu a k čemu se používá?
2. Vyjmenujte jednotlivé metody prahování?
3. Vysvětlete souvislost mezi histogramem a prahováním?

CVIČENÍ

Segmentace obrazu

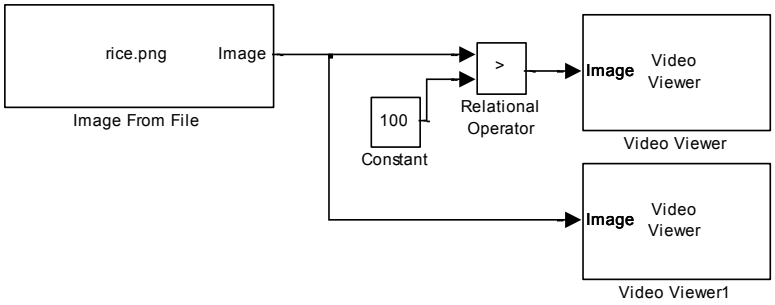
Segmentace obrazu slouží k nalezení jednotlivých složek obrazu. Jsou to malé či větší objekty, které jsou buď samostatné nebo tvoří soubor, jenž se v obraze nachází. V tomto cvičení se budeme zabývat prahováním, ke kterému se využívá v Matlabu metoda **Threshold**. Jedná se o určení optimálního prahu, při kterém jsou objekty v obraze ještě detekovatelné.
Prahování spočívá v převodu zpracovávaného snímku do dvouúrovňové hloubky, kdy hledaný objekt v obraze je odlišen od pozadí. Je zde využito základní dvoubitové hloubky black and white. V tomto cvičení budeme pracovat s obrázkem *rice.png* ze základního obrazového fondu Matlabu.



Obr. 4.11 Originál obrázku.

- **Manuální nastavení Threshold**

Segmentace pomocí Thresholdu může být provedena dvěma způsoby. Buď si připravíme obrázek pomocí manuálního nastavení nebo pomocí bloku Autothreshold. Zapojení simulačního modelu viz schéma na Obr. 4.12.



Obr. 4.12 Schéma zapojení modelu pro manuální nastavení thresholdu.

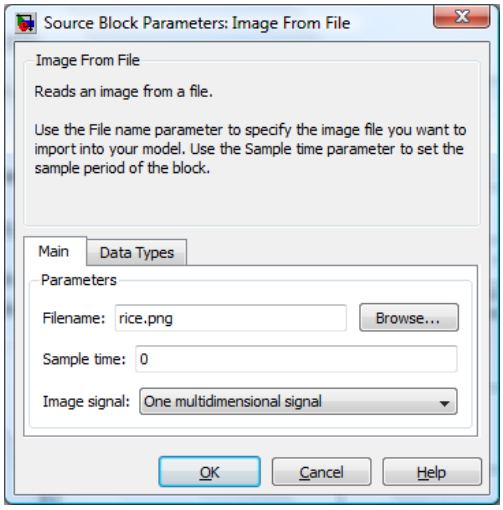
Pro tento příklad použijeme obrázek *rice.png*, který je obsažen v základní obrazové sadě Matlabu.

Jednotlivé bloky naleznete:

- Bloky ***Constant*** a ***Relational Operator*** nalezneme v menu ***Simuling -> Commonly Used Blocks***

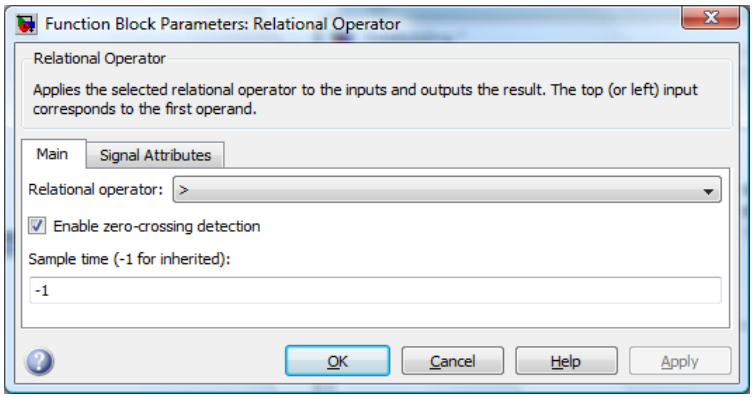
Nastavení vstupních parametrů:

- Parametry bloku ***Image from File*** nastavíme dle následujícího obrázku (Obr. 4.13). Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**, protože budeme pracovat se statickým obrazem. Parametr **Image signal** ponecháme v nastavení **One muldimesional signal**.



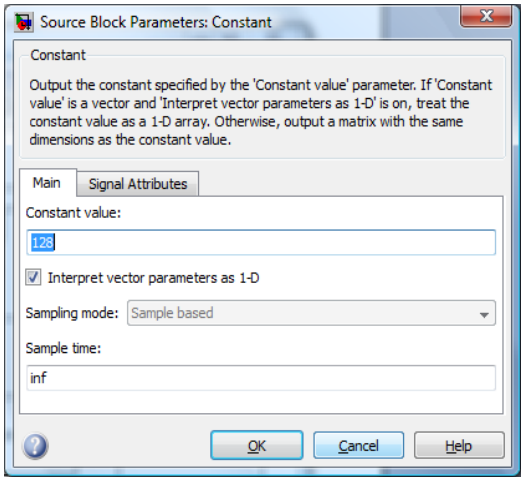
Obr. 4.13 Nastavení vstupních parametrů bloku Image From File.

- Parametry bloku ***Relational Operator*** nastavíme dle následujícího obrázku (Obr. 4.14). V záložce Main nastavíme parametr **Relational operator** na **>**, zaškrtneme checkbox **Enable zero-crossing detection** a parametr **Sample time** ponecháme přednastaven.



Obr. 4.14 Nastavení parametrů bloku Relational Operator.

- Parametry bloku **Constant** nastavíme dle potřeby, viz .Obr. 4.15.



Obr. 4.15 Nastavení bloku Constants.

- Blok **Video Viewer** ponecháme defaultně přednastavený.

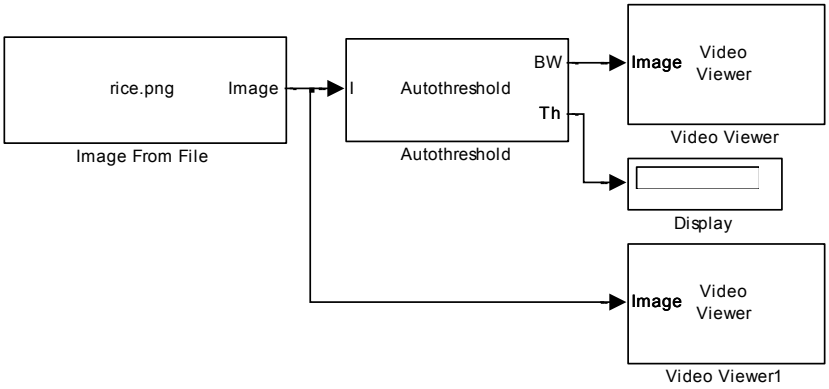
Nastavení parametrů Simulace

Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

Nyní propojte jednotlivé bloky dle schématu, nastavte parametry a můžete model spustit.

❑ Automatické provedení Threshold

Pro automatické nastavení threshold funkce odstraníme z předchozího schéma bloku Relational Operator a Constant a vložíme blok Autothreshold dle schématu na Obr. 4.16.



Obr. 4.16 Schéma zapojení modelu pro Autothreshold blok.

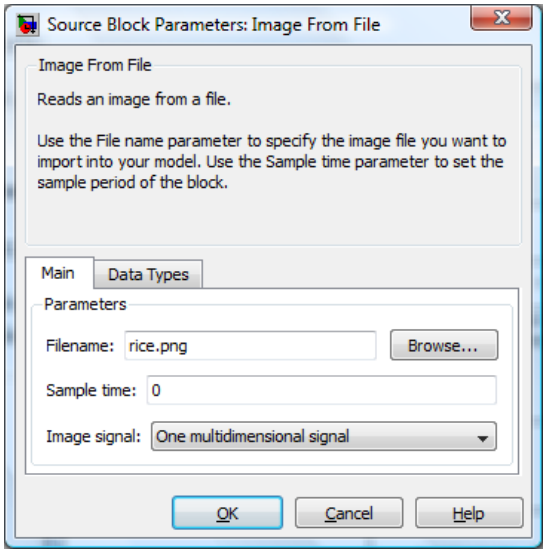
Jednotlivé bloky naleznete:

- Blok ***Autothreshold*** nalezneme v menu ***Video and Image processing Blockset -> Conversions***

- Blok ***Display*** nalezneme v menu ***Simulink -> Sinks***

Nastavení vstupních parametrů:

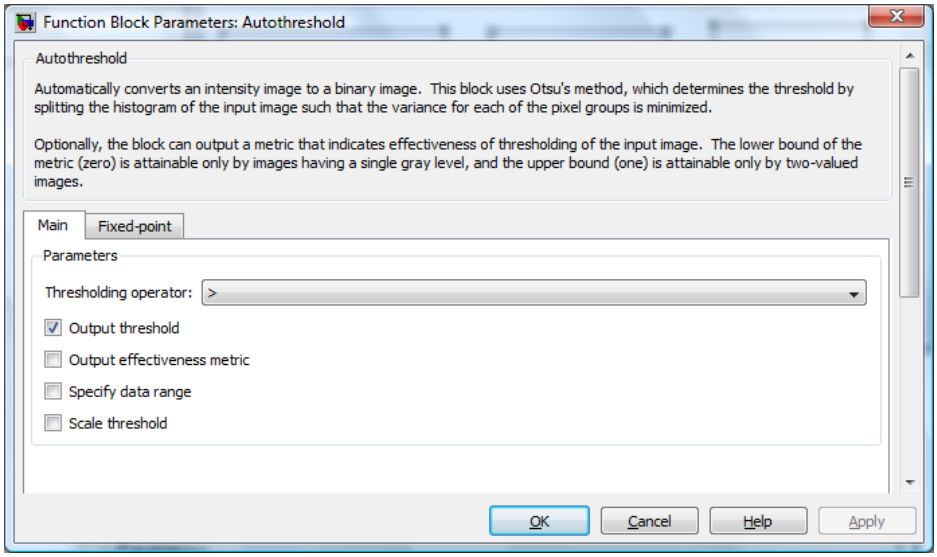
- Parametry bloku ***Image from File*** nastavíme dle následujícího obrázku (Obr. 4.17). Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**, protože budeme pracovat se statickým obrazem. Parametr **Image signal** ponecháme v nastavení **One muldimensional signal**.



Obr. 4.17 Nastavení vstupních parametrů bloku Image From File.

- Parametry bloku ***Relational Operator*** nastavíme záložce Main. Parametr **Relational operator** nastavíme na hodnotu >, zaškrtneme checkbox **Enable zero-crossing detection** a parametr **Sample time** ponecháme přednastaven.

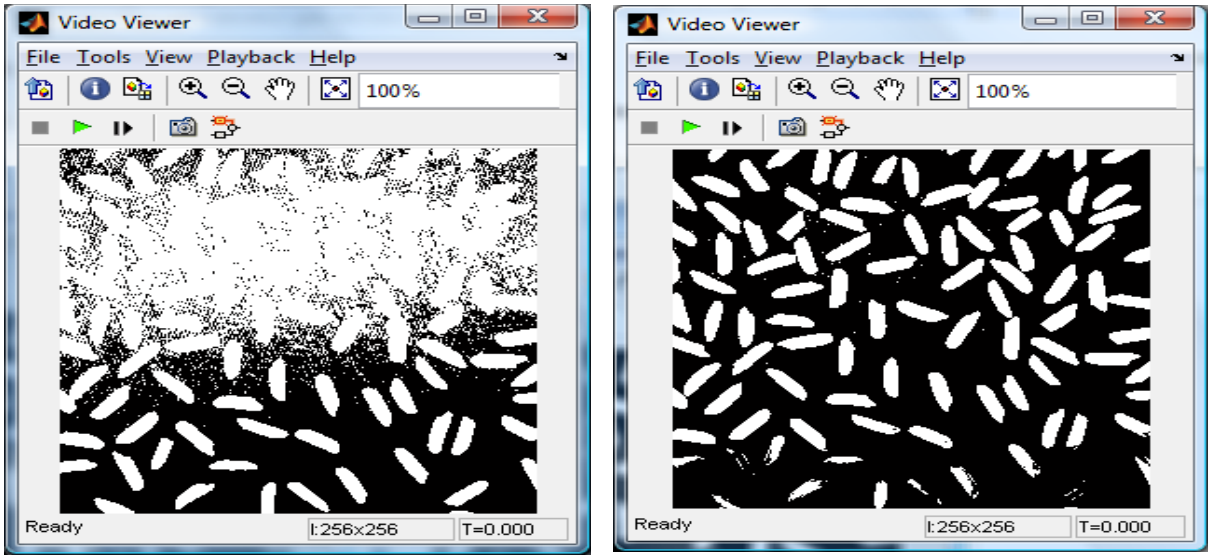
- Parametry bloku ***Autothreshold*** nastavíme dle následujícího obrázku (Obr. 4.18). V záložce main nastavíme parametr **Threshold operator** na > a zaškrtneme checkbox **Output threshold**.



Obr. 4.18 Nastavení parametrů bloku Autothreshold.

- Blok **Video Viewer** ponecháme defaultně přednastavený.

Nyní propojte jednotlivé bloky dle schématu Obr. 4.16, nastavte parametry a můžete spustit simulaci.

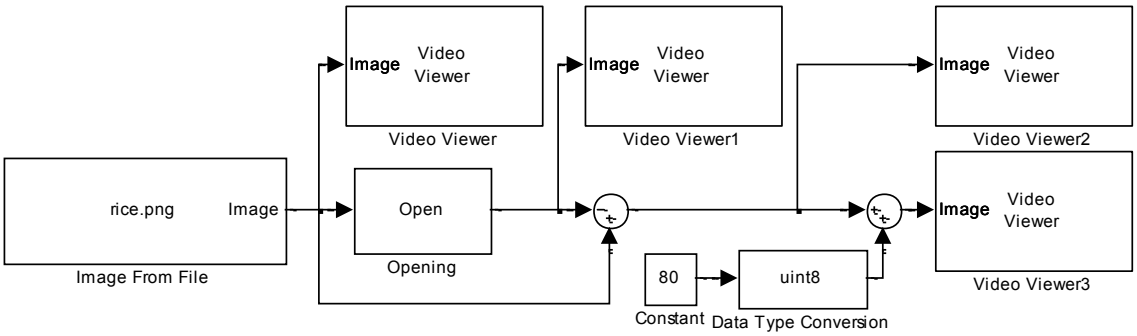


Obr. 4.19 Vlevo manuální nastavení thresholdu na hodnotu 100, vpravo užití bloku autothreshold při nastavení úrovně automaticky.

Obrázky Obr. 4.19 nám ukazují, že automatické nastavení je rychlejší a vyhledá optimální úroveň threshold. Při manuálním nastavení dosáhneme stejných výsledků, ale je zapotřebí vyzkoušet, jaká úroveň je optimální.

Korekce nerovnoměrného osvětlení

Ne vždy je zpracovávaný snímek dokonale osvětlen, viz. předchozí úloha (Obr. 4.19). Vlivem této nedokonalosti jsou některé objekty v obraze hůře detekovatelné. Proto je zapotřebí, je-li to možné, nehomogenitu takového osvětlení eliminovat. Na příkladu si ukážeme, jak se z obrazu tato nehomogenita odstraňuje.
V Simulinku nakreslete model dle schématu na Obr. 4.20. Jako vstupní obrázek použijte Obr. 4.11, který naleznete pod názvem *rice.png* v základní obrazové sadě Matlabu.



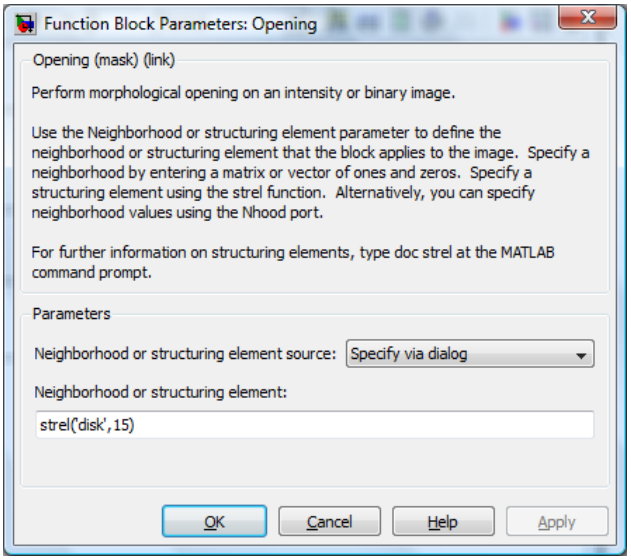
Obr. 4.20 Schéma zapojení modelu.

Jednotlivé bloky naleznete

- Blok ***Opening*** nalezneme v menu ***Video and Image processing Blockset*** -> ***Morphological Operations***
- Blok ***Sum*** nalezneme v menu ***Simulink -> Math Operations***
- Blok ***Data Type Conversion*** nalezneme v menu ***Simulink -> Signal Attributes***

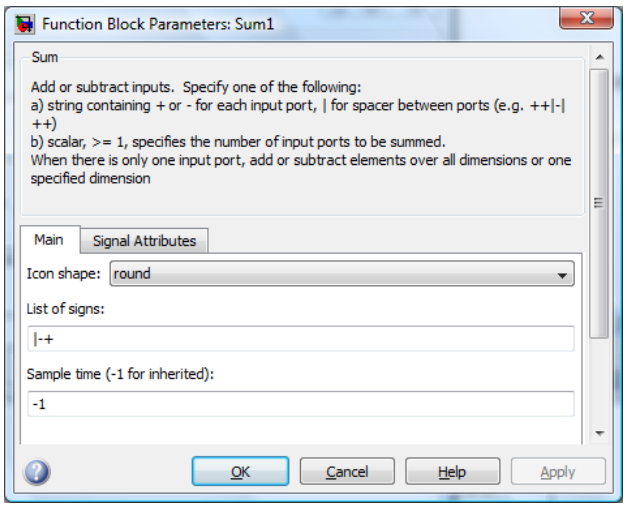
Nastavení vstupních parametrů

- Parametry bloku ***Image from File*** nastavíme dle obrázku z předchozí úlohy (Obr. 4.17). Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**. Parametr **Image signal** ponecháme v nastavení **One muldimensional signal**.
- Parametry bloku ***Opening*** nastavíme dle obrázku Obr. 4.11. Parametr **Neighborhood or structuring element** source ponecháme defaultně přednastaven. Parametr **Neightborhood of structures element** nastavíme na **hodnotu strel('disk',15).**



Obr. 4.11 Nastavení parametrů bloku Opening.

- Parametry bloku ***Sum*** nastavte dle obrázku Obr. 4.222. Parametr **Icon shape** ponecháme na hodnotě **round**, parametr **List of sings** nastavíme na hodnotu „-+“. Parametry druhého bloku ***Sum*** ponecháme defaultně přednastaveny.



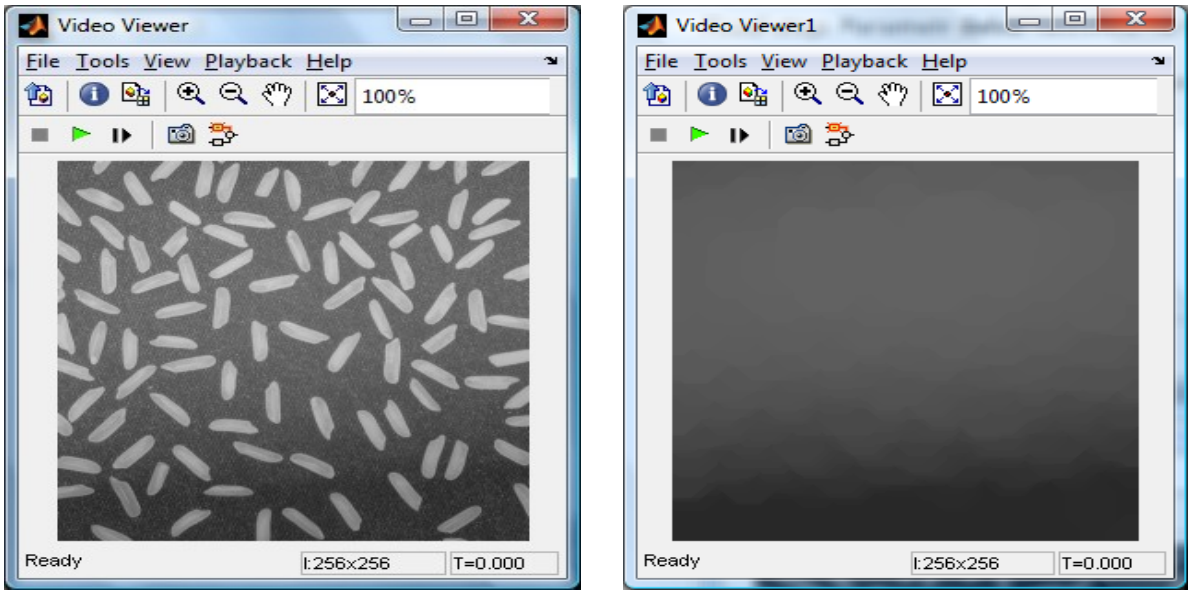
Obr. 4.22 Nastavení parametrů bloku Sum.

- V bloku ***Data Type Conversion*** nastavíme parametr **Output data type** na hodnotu **unit8**. Ostatní parametry nechám defaultně přednastaveny.
- V bloku ***Constant*** nastavíme parametr **Constant value** na hodnotu **80**.
- Parametry bloků ***Video Viewer*** ponecháme defaultně přednastaveny.

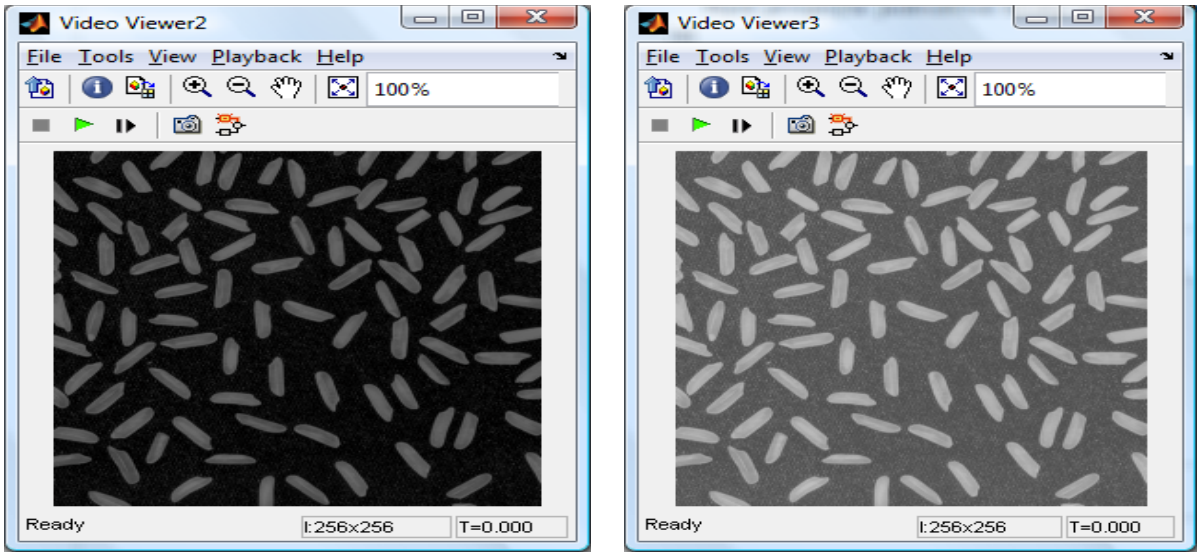
Nastavení parametrů Simulace

Při zpracování histogramu není vykreslení v kontinuálním čase podporováno. Tento fakt je zapotřebí ošetřit následujícím nastavením. Úpravu provedeme z menu ***Simulation*** -> ***Configuration***

Parameters. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***
Nyní propojte jednotlivé bloky dle schématu na Obr. 4.20, nastavte parametry a můžete spustit simulaci.

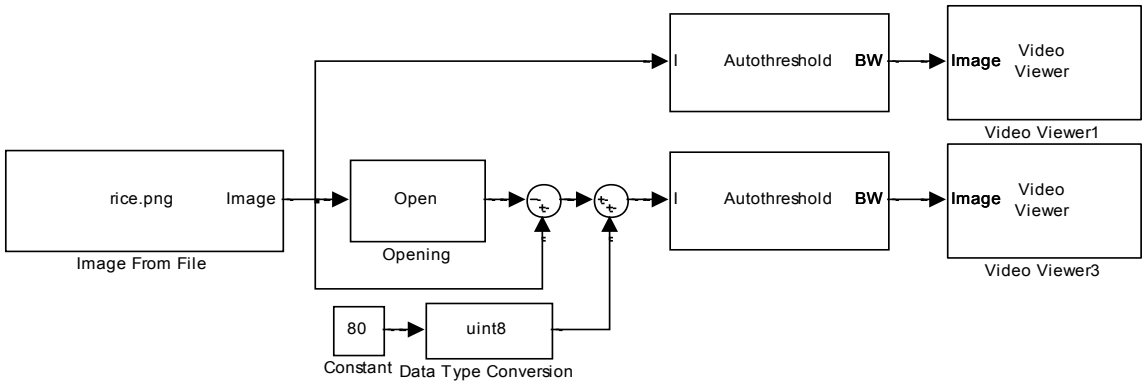


Obr. 4.23 Vlevo originál obrázku se špatným osvětlením objektu, vpravo odhadnuté pozadí obrázku.



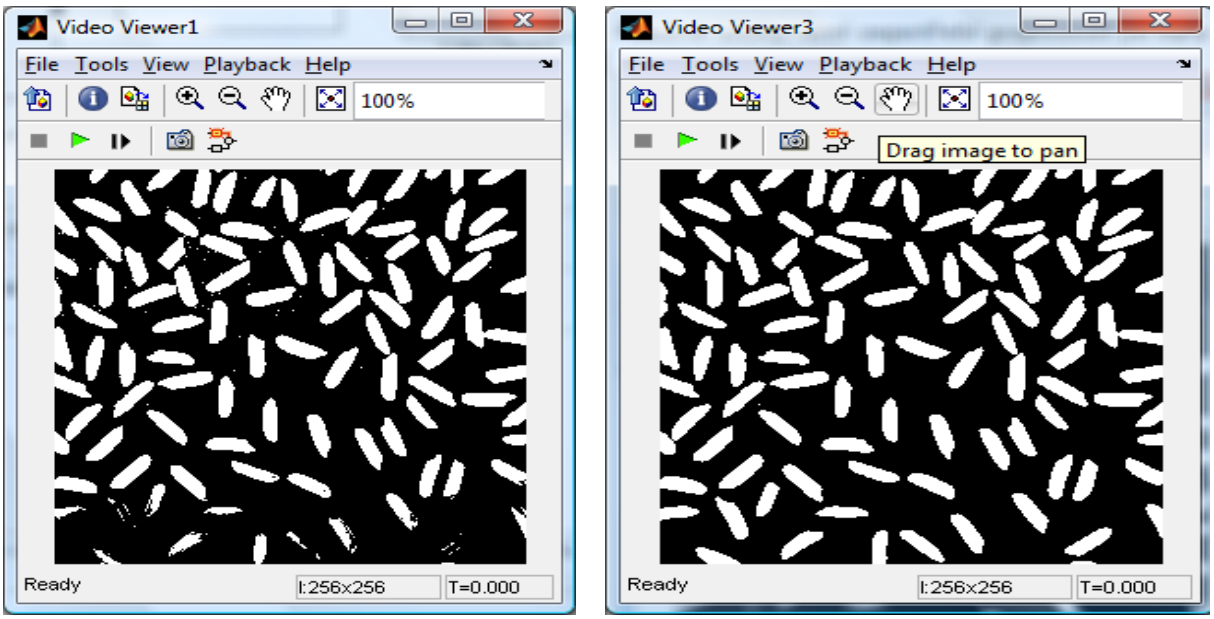
Obr. 4.24 Vlevo korekce v tmavém provedení, vpravo korekce ve světlém provedení.

Pro srovnání korekce výsledného efektu korekce osvětlení bylo použito schéma dle Obr. 4.25.



Obr. 4.25 Schéma zapojení modelu.

Použité bloky není zapotřebí popisovat, protože již byly v této kapitole popsány.



Obr. 4.26 Vlevo segmentace bez použití korekce osvětlení, vpravo s použitím korekce osvětlení.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem *05 Segmentace.swf*.

5. FILTRACE OBRAZU

Čas ke studiu: 2 hodiny

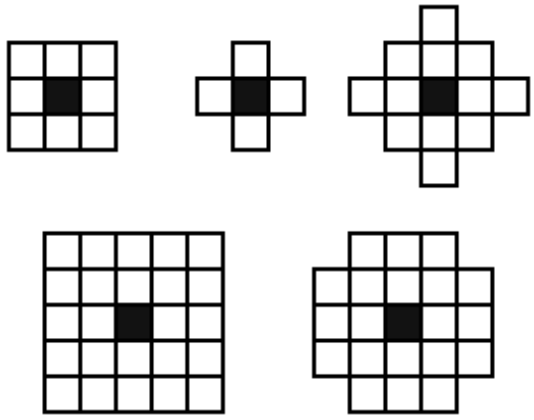
Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit k čemu se používá filtrace obrazu
- popsat způsoby filtrace a jednotlivé metody vyhlazování obrazu.
- popsat jednotlivé typy filtrů a účinně je aplikovat v návrhu řešení.
- vyřešit samostatnou úlohu pro filtraci šumu a zostření popř. rozostření obrazu pomocí nástroje Matlab/Simulink a VIP

Výklad

5.1 Úvod

V oblasti digitálního zpracování obrazu je také důležitá oblast filtrace resp. použití různých typů filtrů. Obecně znamená filtrace soubor transformací obrazu, které převádějí hodnoty jasu pixelů vstupního obrazu na jiné hodnoty jasu pixelů výstupního obrazu. Aplikace vybraných filtrů slouží ke zlepšení vlastnosti obrazu. Jde zejména o úpravu intenzity obrazového bodu (pixelu) s úzkou vazbou na jeho okolí, které může mít různý tvar. Příklady symetrických okolí jsou uvedeny na Obr. 5.1.



Obr. 5.1. Příklady symetrického okolí bodu

Metody filtrace lze aplikovat buď na celý obraz, nebo jen na jeho část, která je oblastí našeho zájmu. Základní rozdělení filtrů je na lineární a nelineární. Lineární můžeme dále dělit na filtry typu dolní a horní propust, nelineární na filtry typu minimum, maximum, medián apod.

5.2 Lokální filtrace obrazu

Lokální filtrace obrazu je založena na metodě filtrace (předzpracování) obrazu, které k výpočtu nové hodnoty pixelu využívá malé okolí *O* reprezentativního pixelu (ve smyslu právě zpracovávaného).

Používají se zde tzv. **prostorové filtry**, které jsou popsány **konvoluční maskou**, která je velmi malá. Což je výhoda oproti filtrům, které se používají ve frekvenční oblasti, kde velikost filtru $H(u,v)$ je stejná jako je velikost obrázku např. *MxN*. Filtrace ve frekvenční oblasti je však intuitivnější.

Princip lokální filtrace:

- Celý obraz se systematicky (např. po řádcích) prochází.
- Kolem reprezentativního bodu je zkoumáno malé okolí *O*, často malý obdélník.
- Výsledek analýzy je zapsán do výstupního obrazu jako hodnota reprezentativního pixelu.

Podle účelu použití se metody lokální filtrace obrazu rozdělují obecně do dvou skupin.
První skupina, **vyhlazování obrazu**, provádí potlačení šumu a osamocených fluktuací hodnot obrazové funkce. Aplikace této metody vede k potlačení vyšších frekvencí obrazové funkce. Metodicky je tato metoda příbuzná dolno frekvenčním filtrům.
Druhá skupina, **detekce hran**, nazývána také gradientní operátory, se snaží z hodnot v okolí reprezentativního pixelu odhadnout derivaci obrazové funkce. Metody detekce hran jsou příbuzné horno frekvenčním filtrům.

Z výše uvedeného je patrno, že vyhlazování a detekce hran jsou (alespoň ve své lineární podobě) v protikladu. Proto jsou navrhovány nelineární metody, které např. vyhlazují a přitom jsou šetrné k hranám a detailům v obraze.

5.2.1 Lokální vyhlazování obrazu

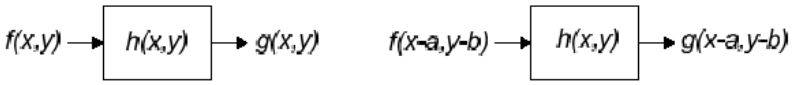
Nejsnadnější je vyhlazování náhodného šumu, když máme k dispozici několik vstupních originálních obrazů téže předlohy, které se liší právě aditivním šumem.

Potom je účinné průměrovat hodnotu pixelu o stejných souřadnicích přes více obrázků. Poznamenejme jen, že zde problémy s rozmazáváním přirozeně nevznikají.

Máme-li k dispozici jen jediný obraz téže předlohy, potom nezbývá než se spolehnout na obvykle značnou nadbytečnost údajů v obraze. Sousední pixely mají převážně tutéž nebo blízkou hodnotu jasu. Hodnotu obrazových elementů zkreslených šumem potom můžeme opravit na základě analýzy hodnot jasu v jeho vybraném okolí. Hodnota reprezentativního pixelu (tj. jeho jas) je nahrazena hodnotou typického reprezentanta mezi hodnotami v okolí nebo kombinací několika hodnot.

❑ Lineární metody vyhlazování

- Lineární metody vyhlazování vypočítávají novou hodnotu reprezentativního pixelu jako lineární kombinaci hodnot ve zkoumaném okolí.
- Pro digitální snímky lze v prostorové oblasti lineární kombinaci vyjádřit jako diskrétní konvoluci.
- Jednotlivé lineární filtry se liší váhami v lineární kombinaci, které jsou dány příslušnou konvoluční maskou h (dvojrozměrnou impulsní odezvou).

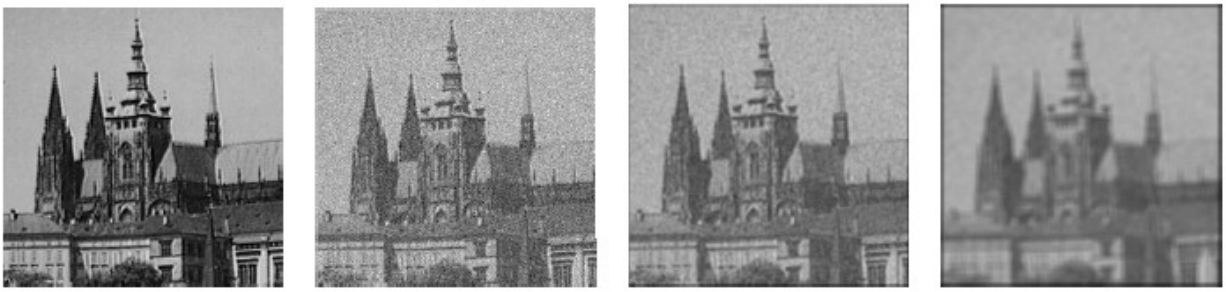


Obr. 5.2 Ilustrace lineárního filtru (vlevo) a lineárního prostorově invariantního filtru (vpravo). [2]

Ve zpracování obrazu se používá zvláštní třída lineárních filtrů. Říká se jim prostorově invariantní filtry (někdy se používá i název homogenní filtry), protože se chování filtru nemění při změně polohy v obrázku, Obr. 5.2. V teorii jednorozměrných signálů je analogickou vlastností časová invariantnost filtrů. Filtr opírající se o představu postupné konvoluce s malou maskou je prostorově invariantní.

Při zpracování skutečných obrazů je:

- předpoklad linearity narušen díky tomu, že hodnota obrazové funkce (jas, intenzita) je nezáporná a omezená.
- obrazy jsou ohraničeny v prostoru, a tak předpoklad prostorové invariantnosti platí jen pro omezené posuny konvolučních masek.



(a) (b) (c) (d)

Obr. 5.3 Vyhlazování šumu s Gausovým rozdělením: (a) výchozí obraz 256, (b) uměle přidán Gausovský šum v jasu, (c) výsledek obyčejného průměrování v okně 3x3, (d) výsledek obyčejného průměrování v okně 7 x 7.[2]

Základní metodou vyhlazování obrazu (vyhlazování šumu) je obyčejné **průměrování**, kde každému bodu (pixelu) přiřadíme hodnotu nového jasu, který je aritmetickým průměrem původních jasů ve zvoleném okolí. Odpovídající konvoluční maska h pro okolí 3 x 3 je

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Někdy se zvětšuje váha středového bodu masky nebo jeho 4-sousedů. Následující vztahy ukazují tyto masky pro okolí 3 x 3. Větší masky se vytvářejí analogicky;

$$h = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad h = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

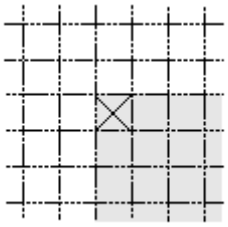
Příklad výsledků filtrace obyčejným průměrováním je zobrazen na Obr. 5.3. Z praktických důvodů bylo zvoleno malé prostorové rozlišení obrázků 256^2 (vyniknutí diskrétního charakteru operací). Základní nevýhodou praktického použití obyčejného průměrování je rozmazávání hran v obraze.
S ohledem na nežádoucí rozmazávání se obyčejné průměrování většinou používá jako pomocná metoda pro výpočet střední hodnoty jasu. Tento mezivýsledek je potom použit v důmyslnějších nelineárních vyhlazovacích metodách.

❑ Nelineární metody vyhlazování

Potíže s rozmazáváním hran (uvedené výše) částečně eliminují nelineární filtrační metody. Princip činnosti je následující:

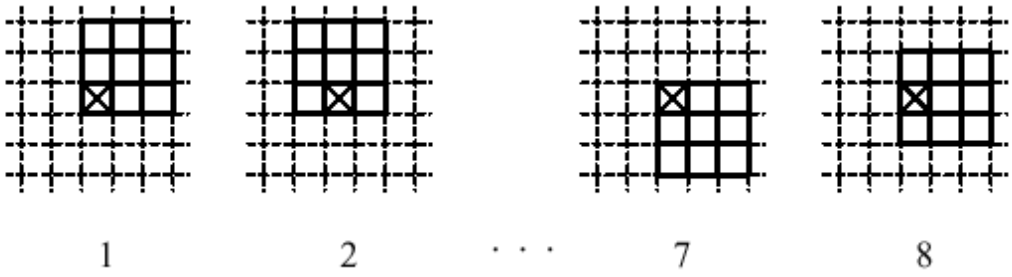
- V analyzovaném okolí *O* se snaží najít jen tu jeho část (oblast o zhruba konstantním jasu), do které reprezentativní bod patří.
- Jen pixely této oblasti se využijí pro hledání jasové hodnoty (např. aritmetickým průměrem nebo výběrem hodnoty jasu jednoho pixelu), která bude reprezentovat celé okolí *O* ve výstupním obrázku.

Představme si, že např. v obdélníkovém okolí *O* rozměru 5 x 5 leží roh tmavého objektu, viz Obr. 5.4. Je přirozené hledat reprezentanta jen uvnitř objektu, což předpokládá nelineární operaci - výběr.



Obr. 5.4 Na jasových rozhraních je potřeba filtrovat jen v pixelech příslušejících objektu. Křížek označuje reprezentativní pixel.

Jako příklad třídy metod vyhlazování, které filtrují jen ve vybraném okolí, uveďme **metodu rotující masky**. Kolem reprezentativního bodu rotuje malá maska, v našem příkladě na Obr. 5.4 je maskou čtverec 3 x 3. Možných poloh masky je 8 viz Obr. 5.5. V každé masce se spočte rozptyl jasů. Za homogenní okolí reprezentativního bodu se vybere maska s nejmenším rozptylem. Nová hodnota reprezentativního bodu může být dána aritmetickým průměrem hodnot ve vybrané masce. Metoda nerozmazává hrany obrazu, má dokonce mírně ostřící charakter.



Obr. 5.5 Osm možných poloh rotující masky. Je možné volit i jiné tvary masky než čtverce.

Metodu rotující masky lze použít iterativně. Tento proces poměrně rychle konverguje do stabilního stavu, kdy se již obraz dále nemění. Velikost a tvar masek ovlivňuje rychlost konvergence. Čím menší jsou masky, tím menší jsou změny a tím více iterací je potřeba.

Jako příklad další používané nelineární metody uveďme **obyčejné průměrování s omezením změn jasu**, které využívá obyčejného průměrování, ale brání rozmazání hran povolením pouze určitého intervalu diference mezi jasem původního bodu a výsledkem průměrování.
Když je diference menší než předem zvolený práh, použije se výsledek průměrování. V opačném případě se původní hodnota nezmění. Metoda se používá pro opravu velkoplošných chyb bez vlivu na zbytek obrazu nebo pro jednoduché vyhlazení bez poškození hran.

❑ Nelineární filtry

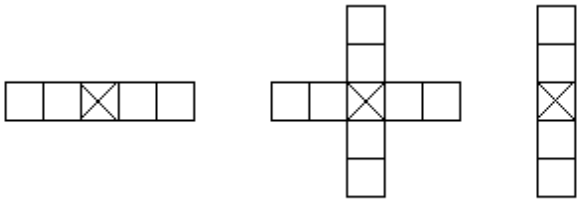
Nelineární filtry nepočítají intenzitu upravovaného bodu, ale vybírají z okolí vhodnou hodnotu, kterou pak dosazují do upravovaného bodu. Oproti lineárním filtrům mají tu výhodu, že nepřidávají do obrazu žádnou novou hodnotu intenzity

Principy nelineárních filtrů jsou založeny na metodách robustní statistiky, kde je snaha najít (v našem případě v obraze) v množině pixelů vychýlené hodnoty, vyloučit je z procesu zpracování a nahradit je nějakou typickou hodnotou.

Ve statistice ztělesňují myšlenku robustních statistik výběrové kvantily. Jejich nejjednodušším případem a ve zpracování obrazů nejčastěji používaným případem je **medián** M. Nechť je X náhodnou veličinou. Medián M je hodnota, pro kterou je pravděpodobnost jevu $x < M$ rovna jedné polovině. Výpočet mediánu je pro diskrétní obrazovou funkci jednoduchý. Stačí uspořádat vzestupně hodnoty jasu v lokálním okolí a medián určit jako prvek, který je uprostřed této posloupnosti. Aby se snadno určil prostřední prvek, používají se posloupnosti s lichým počtem prvků. Pokud se v obrázcích používá lokální čtvercové okolí, jsou jeho rozměry liché, např. 3 x 3, 5 x 5, atd. Výpočet ještě urychlí skutečnost, že k nalezení mediánu stačí částečné uspořádání posloupnosti.

(a) (b) (c)

Obr. 5.6 Filtrace mediánem: (a) Výchozí obraz. (b) Umělé porušení impulsním šumem. (c) Výsledek filtrace mediánem v okně 3 x 3. [2]



Obr. 5.7 Okolí používané pro zachování svislých a vodorovných čar, např. při mediánové filtraci.

Metoda filtrace **mediánem** stanoví jas výsledného bodu jako medián určený z hodnot jasu bodů v lokálním okolí (např. 3 x 3) vstupního obrazu. Metoda redukuje stupeň rozmazání hran a dobře potlačuje impulsní šum. Ukažme si výsledky filtrace mediánem na příkladě, Obr. 5.r. 5.6.

Původní obrázek (Obr. 5.6a) byl uměle porušen impulsním šumem (Obr. 5.6b). Výsledek filtrace mediánem s oknem rozměru 3 x 3 je na Obr. 5.6 c. Filtraci mediánem lze použít iterativně.
Hlavní nevýhodou filtrace mediánem v obdélníkovém okolí je to, že porušuje tenké čáry a ostré rohy v obraze.

Shrnutí pojmů

Filtrace nám slouží k odstranění nežádoucího šumu nebo poruch v obraze. Existují dvě základní metody filtrace – **lineární a nelineární**. **Lineární** metody jsou zejména založeny na principech dolno a horno propustných filtrů. **Nelineární** metody využívají metody statistické analýzy. Patří sem zejména průměrování a medián.

Otázky

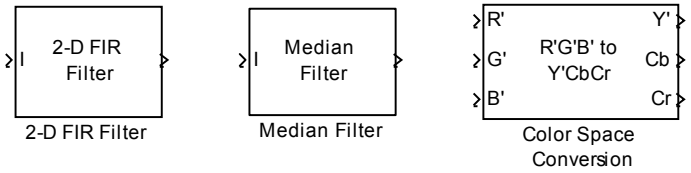
1. K čemu slouží filtry v oblasti digitálního zpracování obrazu?

2. Na jakém principu pracují lineární a nelineární metody filtrace?
3. Jaké metody využívá lokální filtrace?
4. Jaký je rozdíl mezi metodou filtrace využívající obyčejného průměrování a metodou filtrace mediánem. Jaké jsou jejich výhody a nevýhody?

CVIČENÍ

Filtrace obrazu

Filtrace nebo také vyhlazení obrazu slouží k zvýraznění nebo naopak potlačení určité informace která je v obraze obsažena. Obraz můžeme zpracovat vícero metodami. Můžeme z něj odstranit šum, který nám daný obraz zkresluje, můžeme jej vyhladit abychom zvýraznili plochy, můžeme pracovat s kontrastem nebo v neposlední řadě můžeme v obraze hledat hrany, linie nebo tvary. Předzpracováním obrazu pomocí filtračních metod je zpracováním lokálním a nejedná se tudíž o bodové operace. Při filtraci pracujeme s intenzitou bodu s návazností na jeho okolí. K tomu používáme buď metody pracující s váhami v definovaném okolí bodu nebo lineární metody upravují intenzitu tohoto bodu. Nelineární metody pracují s intenzitou z definovaného okolí.
Nejpoužívanějším filtrem jej filtr typu medián, který upravuje prostřední bod v rámci okolí.



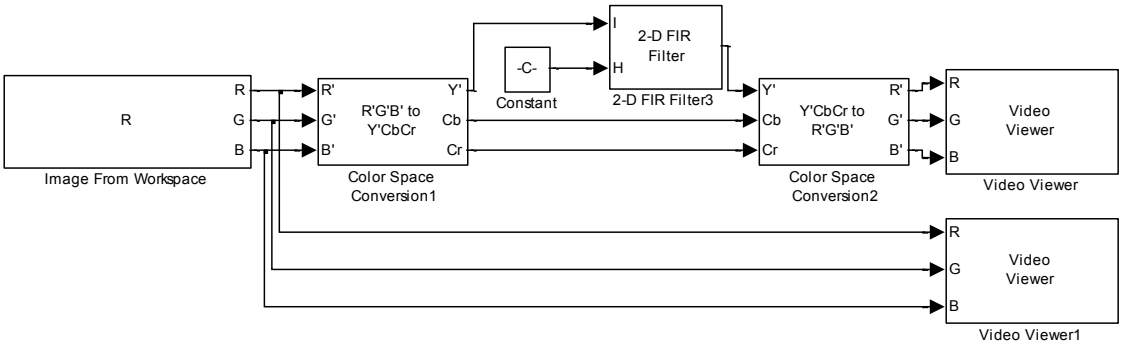
Obr. 5.8 Vlevo blok 2-D FIR filtr, uprostřed Medián Filtr a vpravo je zobrazen blok Color Space Conversion pro vícerozměrné výstupní pole.

Zostření a rozostření obrazu

Pro zostření obrazu je zapotřebí upravit jeho jasovou složku, která zastupuje barevnou informaci. Proto se provádí konverze barevného rozlišení v RGB do YCbCr barevného prostoru. K tomu se používá hornopropustný filtr aplikovaný na tuto jasovou složku. Následně je provedena zpětná konverze z YCbCr barevného prostoru do RGB a jsou zobrazeny výsledky.
Rozostření obrazu se provádí pomocí dolno propustného filtru aplikovaného na jasovou složku obrazu. V tomto cvičení si ukážeme použití 2-D FIR filtru v aplikaci pro zostření a rozostření obrazu.

Obr. 5.9 Originál obrázku

Schéma v Simulinku pro danou úlohu je zobrazeno na Obr. 5.10.



Obr. 5.10 Blokové schéma pro zostření a rozostření obrazu.

Jednotlivé bloky naleznete

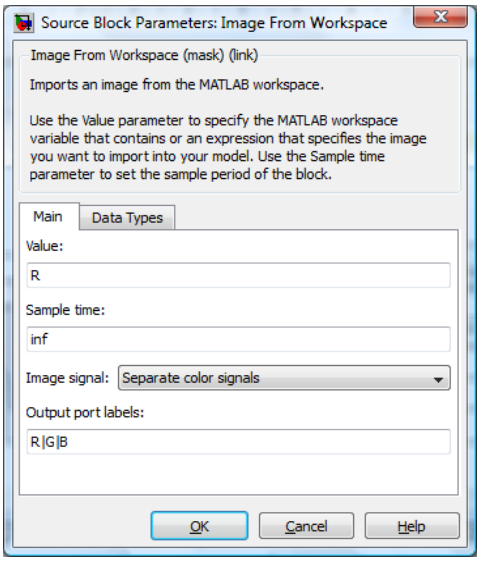
- Blok ***Color Space Conversion*** nalezneme v menu ***Video and Image processing Blockset -> Conversion***.
- Blok ***2-D FIR Filter*** nalezneme v menu ***Video and Image processing Blockset -> Filtering***
- Blok ***Constant*** nalezneme v menu ***Simuling -> Commonly Used Blocks***

Nastavení vstupních parametrů

Před tvorbou modelu v Simulinku nejdříve uložíme vstupní obrázek do workspace Matlabu pomocí následujících dvou příkazů:

R= im2double(imread('peppers.png'));	vloží obrázek do workspace
imshow(R)	zobrazí vložený obrázek

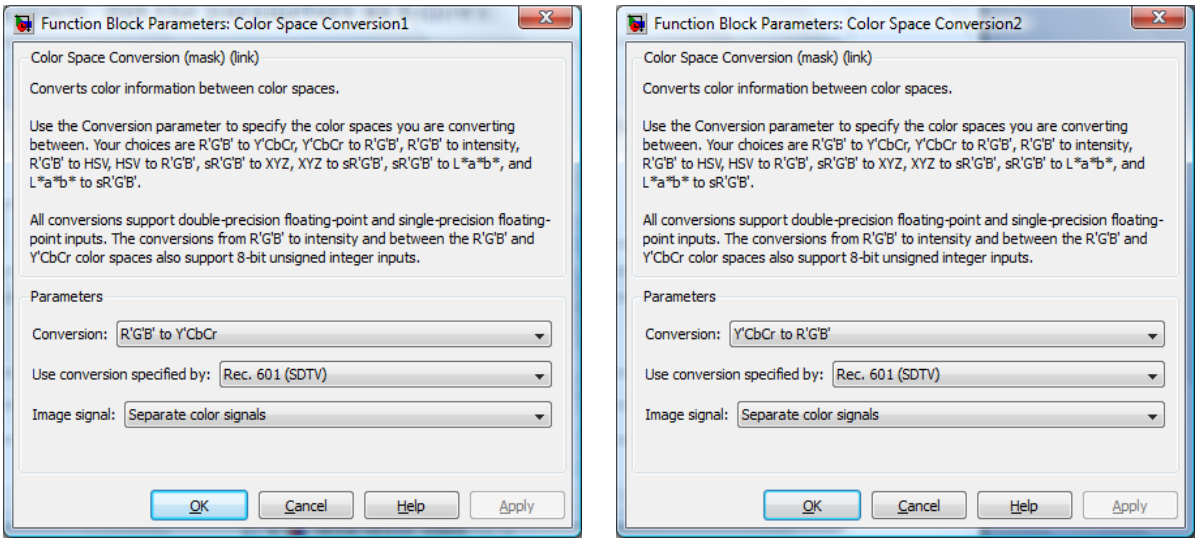
- Parametry bloku ***Image from Workspace*** nastavíme podle obrázku Obr. 5.11. V záložce Main nastavíme parametr **Value** na hodnotu **R**. Parametr **Image signal** nastavíme na hodnotu **Separate color signals**.



Obr. 5.11 Nastavení bloku Image from Workspace v záložce Main.

Parametry bloku ***Color Space Conversion*** nastavíme podle obrázku

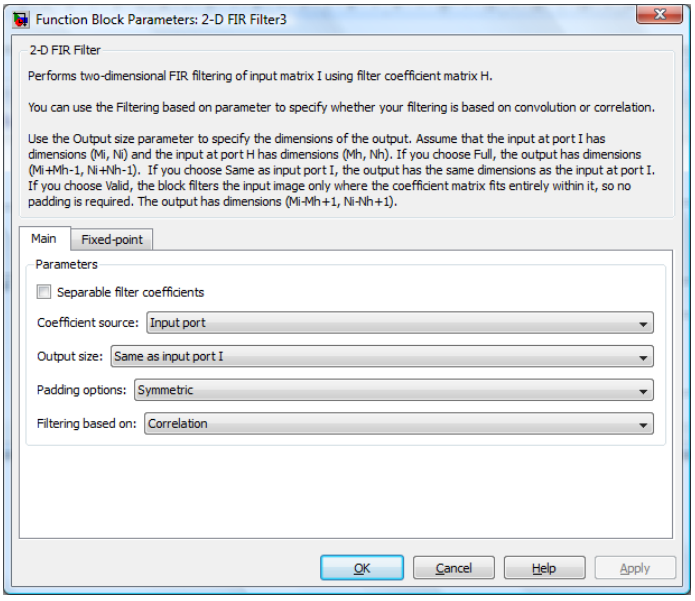
- Obr. 5.22. Pro vstupní signál použijeme nastavení parametru ***Conversion*** na hodnotu ***RGB to YCbCr***, pro výstupní signál z filtru nastavíme parametr **Conversion** na hodnotu opačnou. Parametr ***Use Conversion specified*** určuje konverzní kvalitu videa či obrazu. Není zapotřebí jej upravovat, proto jej ponecháme defaultně přednastavený. Parametr ***Image signal*** nastavíme na hodnotu ***Separate color signals***, abychom dostali všechny tři jasové složky obrazu. Parametry druhého bloku ***Color Space Conversion*** nastavíme stejně, pouze parametr ***Conversion*** nastavíme na hodnotu ***YCbCr to RGB***, jak již bylo zmíněno na začátku této odrážky.



Obr. 5.22 Nastavení parametru bloku Color Space Conversion v záložce main: vlevo pro vstupní signál, vpravo pro výstupní signál.

- Parametry bloku ***2-D FIR filter*** nastavíme dle obrázku (Obr. 5.33). V záložce Main nastavíme parametr **Coefficient source** na hodnotu **Input port.** Parametr **Output size** na hodnotu **Same**

as input port I a parametr **Padding options** na hodnotu **Symmetric**. Parametr **Filtering based on** nastavíme na hodnotu **Correlation**.

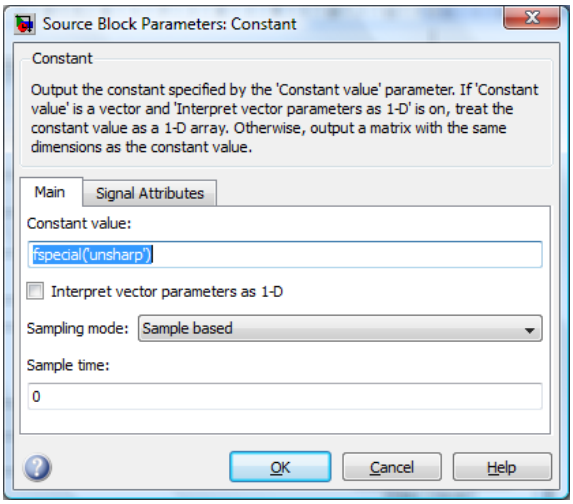


Obr. 5.33 Nastavení parametrů bloku 2-D FIR Filter v záložce Main.

- Parametry bloku ***Constant*** slouží k definování funkce pro práci s obrazem a tedy definici toho, zda-li budeme obraz zostřovat nebo jej rozostříme. Parametry nastavíme dle obrázku (Obr. 5.44) pod tabulkou (Tab.5.1). Zadáním funkce **fspecial('unsharp')** vytvoříme dvourozměrný koeficient pro hornopropustný filtr, který odstraňuje z obrazu nízkofrekvenční šum. Další filtry, které je možné v této aplikaci použít, naleznete v Tab.5.1.

Filtr	**Popis filtru**
average	Průměrování
disk	Kruhové průměrování (pillbox)
gaussian	Gaussův dolno propustný filtr
laplacian	Aproximační dvou rozměrný Laplaceův operátor
log	Laplaceův nebo Gaussův filtr
motion	Aproximace lineárního pohybu kamery
prewitt	Zdůrazňuje horizontální hranu
sobel	Zdůrazňuje vertikální hranu
unsharp	Zdůrazňuje kontrast hrany

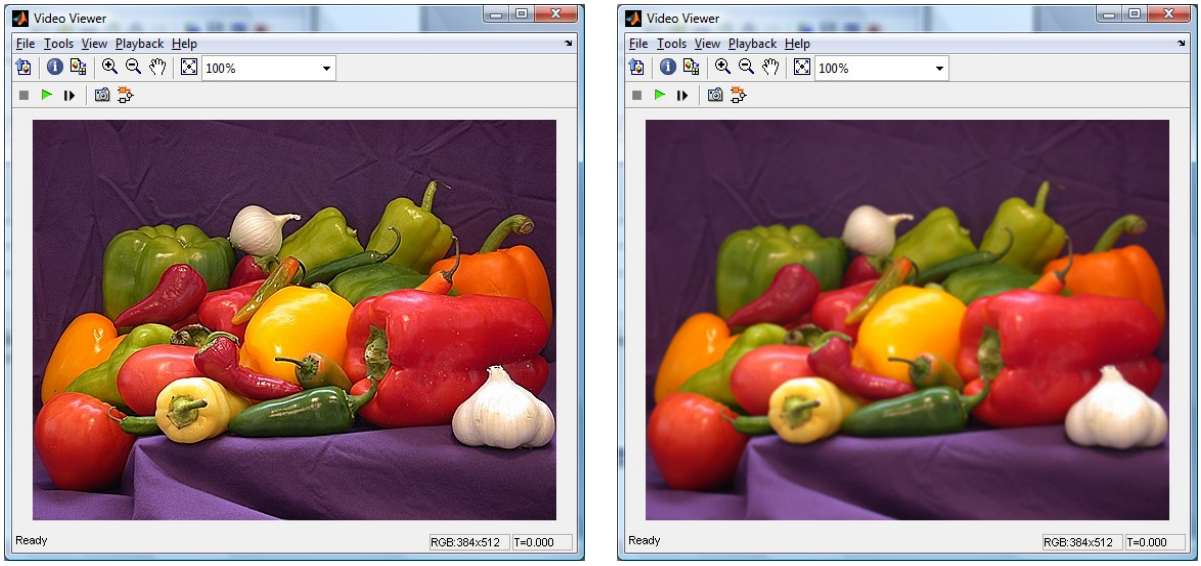
Tab.5.1 Typy filtrů pro úpravu obrazu.



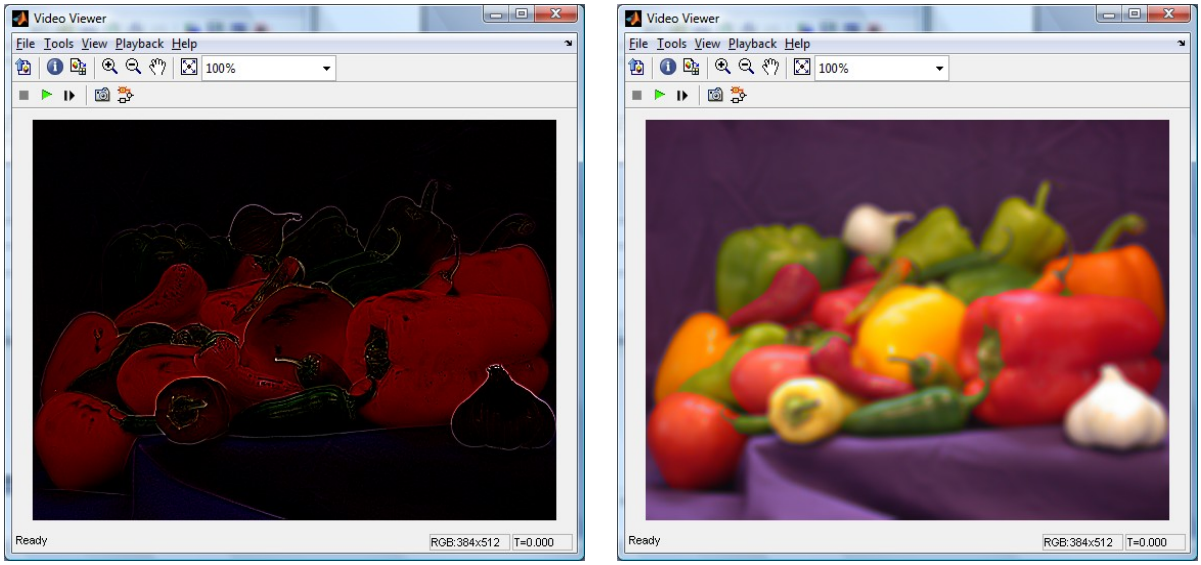
Obr. 5.44 Nastavení parametrů bloku Constant v záložce main.

- Parametry bloků ***Video Viewer*** nastavíme v záložce Main. Parametr nastavíme **Image signal** na hodnotu **Separate color singnals**.

Nyní jednotlivé bloky propojíme, zadáme parametry pro simulaci a model můžeme spustit.



Obr. 5.55 Vlevo užití funkce unsharp pro zostření obrazu, vpravo užití funkce average pro rozostření.

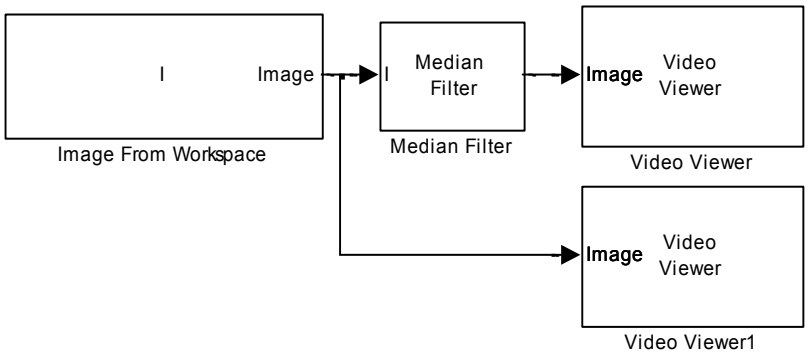


Obr. 5.66 Vlevo užití funkce log pro zvýraznění hran, vpravo užití funkce disk pro rozostření.

Odstranění „Sold and Pepper“ šumu

Pro odstranění šumu typu Sold and Pepper, který je prezentován drobným zrněním světlých a tmavých pixelů v obraze, se nejčastěji používá filtr Medián. Tento filtr je dostatečně citlivý pro lineární vyhlazení těchto extrémů. K odstranění šumu typu sold and pepper užitím filtru Medián tak dochází bez úpravy ostrosti obrazu.

Pro aplikaci filtru vytvoříme v Simulinku schéma dle obrázku (Obr. 5.77).



Obr. 5.77 Blokové schéma pro odstranění šumu typu „sold and pepper“.

Jednotlivé bloky nalezneme

- Blok ***Median Filter*** nalezneme v menu ***Video and Image processing Blockset -> Filtering***

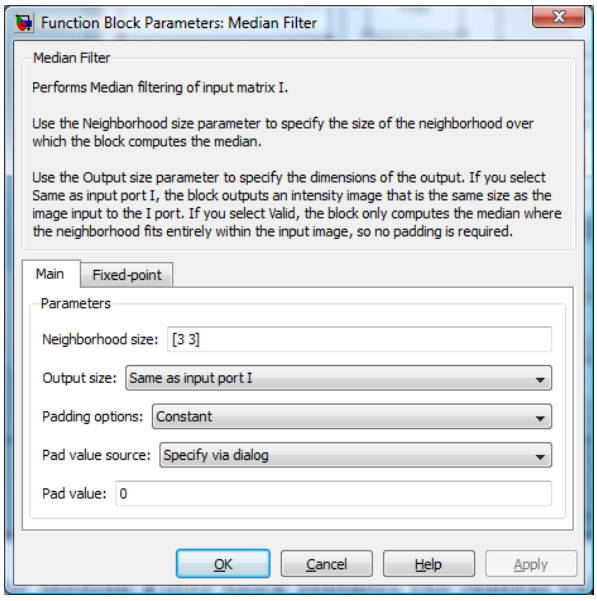
Nastavení vstupních parametrů

Před tvorbou modelu v Simulinku nejdříve uložíme vstupní obrázek do workspace Matlabu pomocí následujících tří příkazů:

I= double(imread('circles.png'));	vloží obrázek do workspace
I= imnoise(I,'salt & pepper',0.02);	vloží do obrázku šum, který budeme odstraňovat.
imshow(R)	vložený obrázek zobrazí

- Parametry bloku ***Image from Workspace*** nastavíme dle následujícího popisu. V záložce Main nastavíme parametr **Value** na hodnotu **R**. Parametr **Image signal** nastavíme na hodnotu **Separate color signals**.

- Parametry bloku ***Median Filter*** nastavíme dle obrázku (Obr. 5.88). V záložce Main nastavíme parametr **Neighborhood size** na hodnotu **[3 3]**, což je počet pixelů zahrnutých do oblasti sousedních bodů. Parametr **Output size** ponecháme přednastavený na hodnotu **Same as input port I**. Parametr **Padding options** ponecháme přednastavený na hodnotě **Constant**. Parametry **Pad value source** a **Pad value** ponecháme taktéž přednastaveny na hodnotách **Specify via dialog** a **0**.



Obr. 5.88 Nastavení parametrů bloku Median Filter v záložce Main.

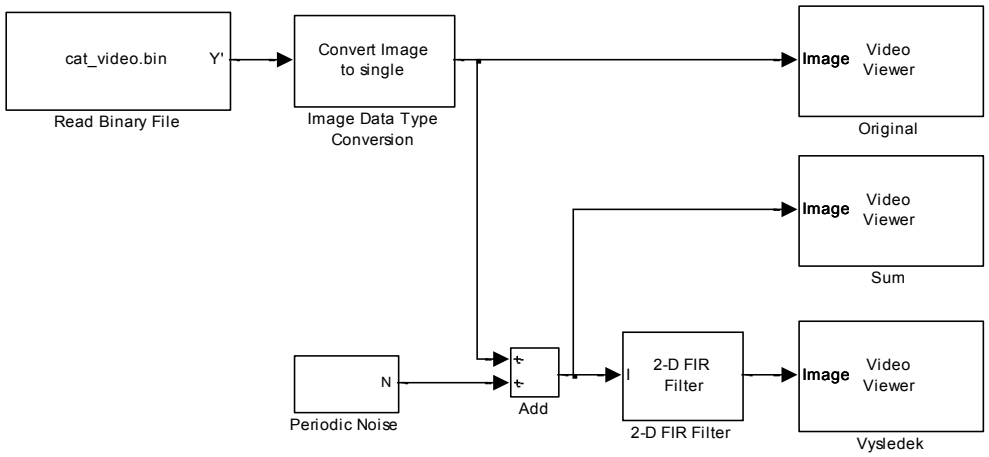
- Definici parametrů bloků ***Video Viewer*** provedeme v záložce Main. Parametr **Image signal** nastavíme na hodnotu **Separate color signals**.

Nyní jednotlivé bloky spojíme dle schématu na Obr. 5.77, nastavíme parametry simulace a můžeme spustit simulaci.

Obr. 5.99 Vlevo zobrazení původního obrázku, který obsahuje šum typu sold and pepper, vpravo obrázek po filtraci Median filtrem.

Odstranění periodického šumu

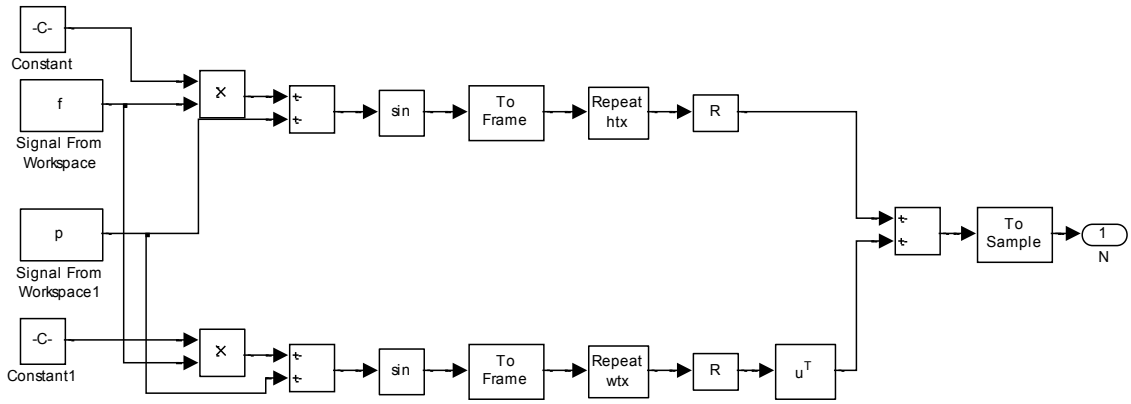
Pro odstranění periodického šumu vytvoříme v Simulinku model dle schématu na Obr. 5.20.



Obr. 5.20 Blokové schéma pro odstranění periodického šumu.

Jednotlivé bloky naleznete:

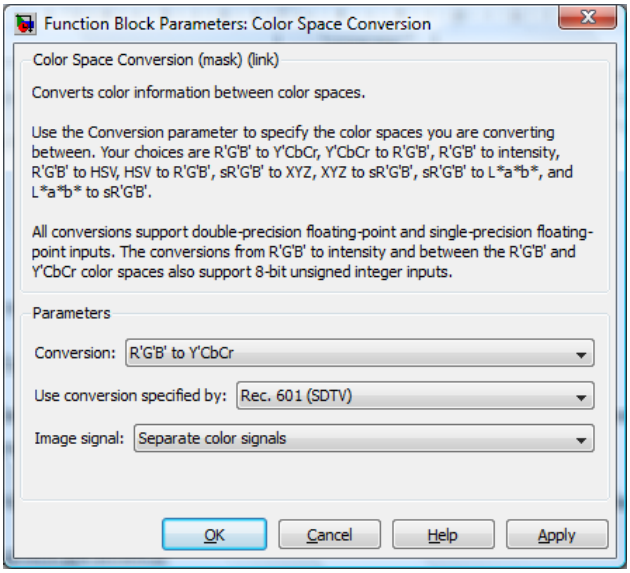
- Toto schéma nebudeme během cvičení vytvářet jelikož je příliš komplikované a jeho vytvoření je časově náročné. V Matlabu vyhledáme soubor vipstripes_all.mdl a spustíme. Schéma jednoho z bloků tohoto schématu je zobrazeno na Obr. 5.101. Toto schéma je zde zobrazeno pro demonstraci složitosti vnořených funkcí.



Obr. 5.101 Schéma bloku Period Noise, které je součástí souboru vipstripes_all.mdl.

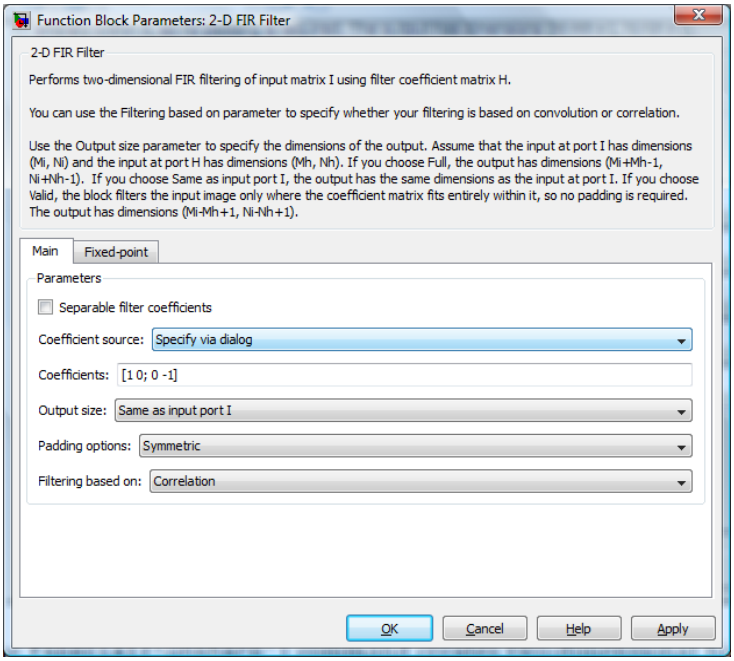
Nastavení vstupních parametrů

- Parametry bloku ***Color Space Conversion*** nastavíme dle Obr. 5.22. Pro vstupní signál použijeme nastavení parametru **Conversion** na hodnotu ***RGB to YCbCr***, pro výstupní signál z filtru nastavíme parametr **Conversion** na hodnotu opačnou tedy na **YCbCr to RGB**. Parametr ***Use Conversion specified*** , který určuje konverzní kvalitu videa či obrazu, nastavíme na hodnotu **Rec.601(SDTV).** Parametr ***Image signal*** nastavíme na hodnotu ***Separate color signals***, abychom dostali všechny tři jasové složky obrazu.



Obr. 5.112 Nastavení bloku Color Space Conversion.

- Parametry bloku ***2-D FIR Filter*** nastavíme dle obrázku (Obr. 5.113). V záložce **Main** nastavíme parametr **Coefficient source** na hodnotu **Specify via dialog**, parametr ***Output size*** nastavíme na hodnotu ***Same as input port I***, parametr ***Padding options*** nastavíme na hodnotu ***Symmetric*** a parametr ***Filtering based on*** nastavíme na hodnotu ***Correlation***. Parametry v záložce **Fixed point** ponecháme defaultně přednastavené.

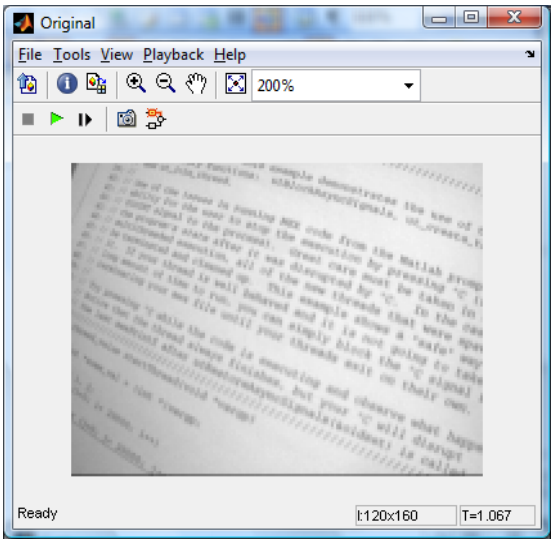


Obr. 5.123 Nastavení parametrů bloku 2-D FIR Filtr.

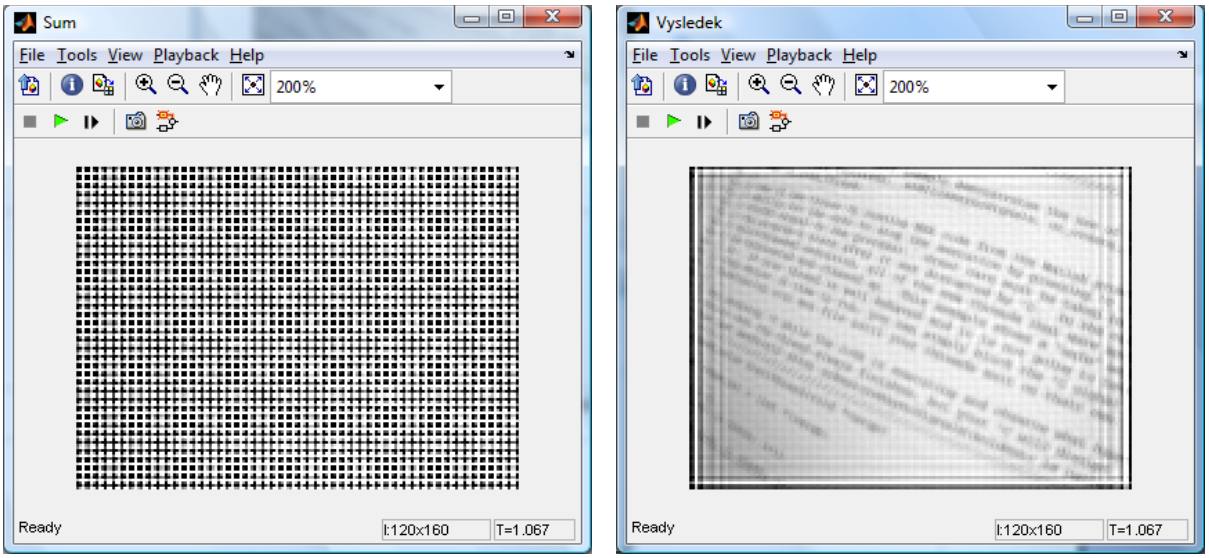
Nastavení parametrů simulace

Dále je zapotřebí provést nastavení konfiguračních parametrů. To provedeme v dialogovém okně ***Simulation -> Configuration parameters***. Je zapotřebí zadat parametr ***Stop time*** = 0, parametr ***Type*** = ***Fixed-step*** a parametr ***Solver = Discrete (no continuous states).***

Do Matlabu napíšeme příkazy **clear all**, jenž vymaže workspace a příkaz **vipdh_stripes**, který nastaví vstupní proměnnou **h**. Tato proměnná není v modelu definována a je zapotřebí provést její validaci. Nyní propojíme jednotlivé bloky dle schématu (Obr. 5.21), nastavíme parametry simulace a můžeme spustit simulaci.



Obr. 5.134 Původní video snímek.



Obr. 5.145 Vlevo periodický šum, vpravo snímek s odstraněním periodického šumu.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 05 Filtrace.swf.

6. GEOMETRICKÉ TRANSFORMACE OBRAZU

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat co je to geometrická transformace a kde se nejčastěji využívá.
- popsat jednotlivé typy interpolací, k čemu slouží a jak s nimi pracovat.
- vyřešit samostatnou úlohu v Matlab/Simulink a VIP týkající se geometrické transformace

Výklad

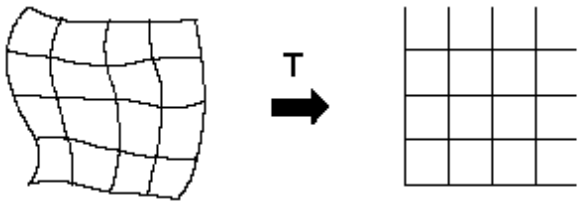
6.1 Úvod

Geometrické transformace obrazu (v našem případě 2D) popisují transformaci obrazové funkce *f(x; y)*, tj. souřadnic *x, y* při změně zobrazení (nejčastěji rotace, zvětšení, zmenšení apod.). Celý proces probíhá ve dvou fázích, nejprve se provede transformace souřadnic bodů a následně se provede transformace jasu. Učební text této kapitoly převzat a upraven podle [2].

6.2 Geometrické transformace

Geometrické transformace vypočtou na základě souřadnic bodů ve vstupním obraze souřadnice bodů ve výstupním obraze. Geometrické transformace jsou obvyklé v počítačové grafice, která často pracuje s objekty ve vektorovém tvaru. V digitálním zpracování obrazu navíc geometrické transformace dovolují odstranit geometrické zkreslení vzniklé při pořízení obrazu (např. korekce geometrických vad objektivu kamery, oprava zkreslení družicového snímku způsobená zakřivením zeměkoule). [2]

Mezi korekcí geometrického zkreslení a geometrickými transformacemi, jako je posunutí nebo rotace, není principiální rozdíl. Použijí se stejné algoritmy. V případě korekcí budeme mluvit o zkresleném, resp. nezkresleném obraze a u geometrických transformací o novém, resp. původním obraze



Obr.6.1 Ilustrace geometrické transformace v rovině. [2]

Geometrická transformace plošného obrazu je vektorová funkce ***T***, která zobrazí bod *x, y* do bodu *x', y'*. Situaci ilustruje Obr. 6.1, kde je bod po bodu transformována část roviny. Transformace ***T*** je definována dvěma složkovými vztahy:

$$x' = T_x(x, y) \;,\; y' = T_y(x, y) \qquad (6.1)$$

Transformační rovnice T_x a T_y mohou být buď známy předem, jako je tomu např. v případě rotace, posunu nebo zvětšení obrazu, nebo je možné hledat transformační vztah na základě znalosti původního i transformovaného obrazu.
Např. jestliže $T_x(x, y) = x/2$ a $T_y(x, y) = y/2$ jde o zmenšení obrázku v obou prostorových souřadnicích.

Při hledání transformace se obvykle využívá několika známých (tzv. lícovacích) bodů, které v obou obrazech odpovídají identickému objektu a lze je snadno najít.

Geometrická transformace sestává ze dvou kroků:
1. Transformace souřadnic bodů
2. Aproximace jasové stupnice

Transformace souřadnic bodů
- najde k bodu ve vstupním obraze s diskrétními souřadnicemi odpovídající bod ve výstupním obrazu.
- u výstupního bodu musíme počítat se spojitými souřadnicemi (reálná čísla), protože poloha výsledného bodu nemusí souhlasit s celočíselnou mřížkou.
- tato plošná transformace má tedy bodový charakter.

Aproximace jasové funkce
- hledá celočíselnou hodnotu jasu v celočíselné pozici, která nejlépe odpovídá nově vypočítané neceločíselné poloze *x'*, *y'*.

6.2.1Transformace souřadnic bodů

Jak určit souřadnice bodu v obraze, který je výsledkem geometrické transformace, uvádí v obecném tvaru vztah (6.2). Transformační vztah souřadnic se obvykle aproximuje polynomem m-tého stupně

$$x' = \sum_{r=0}^{m}\sum_{k=0}^{m-r} a_{rk} x^r y^k \qquad y' = \sum_{r=0}^{m}\sum_{k=0}^{m-r} b_{rk} x^r y^k \tag{6.2}$$

a_{rk}, b_{rk} - koeficienty transformace
- určujeme je pomocí metody nejmenších čtverců na základě přeurčené množiny dvojic sobě odpovídajících (lícovacích) bodů ve vstupním a výstupním obraze *x, y* a *x', y'*.

6.2.2 Bilineární transformace

- k jejímu stanovení stačí alespoň čtyři dvojice sobě odpovídajících bodů,

$$x' = a_0 + a_1 x + a_2 y + a_3 xy$$
$$y' = b_0 + b_1 x + b_2 y + b_3 xy$$

6.2.3 Afinní transformace

- zvláštní případ bilineární transformace
- zahrnuje rotaci, translaci a zkosení

$$x' = a_0 + a_1 x + a_2 y$$
$$y' = b_0 + b_1 x + b_2 y$$

Obecnější než transformace afinní jsou *transformace projektivní*. Jak afinní tak projektivní transformace zachovávají linearitu transformovaných útvarů (přímky se transformují opět na přímky). Na rozdíl od afinních transformací však projektivní transformace nezachovávají rovnoběžnost. Projektivní transformace se uplatňují především v případech, které nějakým způsobem souvisí se středovým promítání. Př. Transformace, při níž má být obraz deformován tak, aby budil dojem, že je umístěn na nějaké obecně umístěné rovině v prostoru, kterou zobrazujeme středovým promítáním.

Při popisu projektivních transformací se obvykle používají *homogenní souřadnice*.

Pozn.: *homogenní souřadnice*

- umožní v geometrických transformacích vyjádřit posun, rotaci, afinní transformaci jako součin s jedinou maticí.
- základní myšlenka - reprezentovat bod ve vektorovém prostoru o jednu dimenzi větším.

Bod $[x, y]^T \rightarrow$ v homogenních souřadnicích $\rightarrow [\lambda_x, \lambda_y, \lambda]^T$ kde $\lambda \neq 0$

Pro jednoduchost obvykle

$$[x, y, 1]^T$$

Afinní zobrazení se po zavedení homogenních souřadnic vyjádří maticově jako

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_0 \\ b_1 & b_2 & b_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Současně platí, je-li trojice (*x,y,w*) homogenními souřadnicemi nějakého bodu, pak také trojice (*x/w,y/w,1*) je homogenními souřadnicemi téhož bodu a dvojice(*x/w, y/w*) je jeho souřadnicemi afinními. Projektivní transformace bodu o afinních souřadnicích (*x,y*) je popsána rovnicí

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogenní souřadnice bodu po transformaci jsou (*x',y',w'*), odkud pro afinní transformace plyne vztah (*x'/w',y'/w'*).
Z výše uvedených vztahů vyplývá, že ve dvojrozměrném prostoru je projektivní transformace popsána devíti hodnotami $p_{11}, p_{12}, \ldots, p_{33}$. Uvedené hodnoty lze ovšem násobit libovolnou nenulovou reálnou konstantou, aniž by se změnil výsledek.

6.2.4 Výpočet projektivní transformace

V této části si ukážeme, jak vypočítat matici *P*, která reprezentuje projektivní transformaci mezi dvěma obrazy. V těchto dvou obrazech mějme tzv. lícovací body, tj. body které si v obrazech vzájemně korespondují (označeny červeným křížkem):

Obr. 6.2 Transformace obrazu

Projektivní transformace je taková, která váže souřadné systémy vztahy :

$$x_1 = \frac{a_{11}u_1 + a_{12}u_2 + a_{13}}{a_{31}u_1 + a_{32}u_2 + a_{33}},$$

(6.3)

$$x_2 = \frac{a_{21}u_1 + a_{22}u_2 + a_{23}}{a_{31}u_1 + a_{32}u_2 + a_{33}}.$$

(6.4)

Tyto vztahy můžeme přepsat do kompaktnějšího tvaru:

$$x_1 = \frac{\mathbf{A}_1\left[\mathbf{u}^T\,1\right]^T}{\mathbf{A}_3\left[\mathbf{u}^T\,1\right]^T}, \quad x_2 = \frac{\mathbf{A}_2\left[\mathbf{u}^T\,1\right]^T}{\mathbf{A}_3\left[\mathbf{u}^T\,1\right]^T}$$

(6.5)

Lícovacích dvojic nechť je *n*. Pro každou dvojici lícovacích bodů (x_i, u_i) mají platit vztahy (6.5), kam za ***x*** dosadíme ***xi*** a za ***u*** dosadíme ***ui***. Pak hledáme matici ***P*** takovou, aby (6.5) byla splněna pro všechny lícovací dvojice. Nechť prvky matice *A* jsou totožné s prvky matice *P*.
Nalezení ***P*** je obzvláště jednoduché, neboť vztahy (6.5) jsou v prvcích **P** lineární. Úpravou vztahu totiž dostaneme:

$$\mathbf{P}_3\left[\mathbf{u}^T\,1\right]^T x_1 - \mathbf{P}_1\left[\mathbf{u}^T\,1\right]^T = 0$$
$$\mathbf{P}_3\left[\mathbf{u}^T\,1\right]^T x_2 - \mathbf{P}_2\left[\mathbf{u}^T\,1\right]^T = 0$$

Utvoříme z prvků matice **P** vektor $\mathbf{p} = [\mathbf{P}1,\mathbf{P}2,\mathbf{P}3] = [p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}, p_{31}, p_{32}, p_{33}]^T$
a zapíšeme lineární soustavu rovnic pro tento vektor:

$$\begin{pmatrix} u_1^1 & u_2^1 & 1 & 0 & 0 & 0 & -x_1^1u_1^1 & -x_1^1u_2^1 & -x_1^1 \\ 0 & 0 & 0 & u_1^1 & u_2^1 & 1 & -x_2^1u_1^1 & -x_2^1u_2^1 & -x_2^1 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ u_1^n & u_2^n & 1 & 0 & 0 & 0 & -x_1^nu_1^n & -x_1^nu_2^n & -x_1^n \\ 0 & 0 & 0 & u_1^n & u_2^n & 1 & -x_2^nu_1^n & -x_2^nu_2^n & -x_2^n \end{pmatrix} \mathbf{p} = \mathbf{0}$$

Soustavu vyřešíme a tím dostaneme vektor **p**, který pak jen přeskládáme do matice **P** a tím jsme hotovi.

Vidíme, že každý ze zadaných bodů přispívá k nalezení transformační matice dvěma rovnicemi, což dává celkem 8 rovnic pro devět neznámých. Protože víme, že transformační matici je možné násobit libovolným reálným číslem, aniž by se výsledek transformace změnil, zvolíme k zajištění jednoznačnosti řešení doplňující podmínku. Nejsnáze se prakticky realizuje postup, kdy některý z prvků transformační matice položíme roven předem zvolené hodnotě.

Složitější geometrické transformace (zkreslení)

- je možné aproximovat tak, že se obraz rozdělí na menší obdélníkové podobrazy.
- Pro každý z podobrazů se použije jednodušší geometrická transformace (např. afinní) odhadnutá z dvojic sobě odpovídajících bodů.
- Zkreslení uvnitř každého podobrazu je opravováno zvlášť.

Pro ilustraci uveďme typická geometrická zkreslení, která se vyskytují v dálkovém průzkumu Země. Chybná poloha a orientace senzoru (nebo družice) vzhledem k pozorovanému objektu je hlavním zdrojem chyb rotace, zkosení a nelineárních zkreslení v řádcích.

- Panoramatické zkreslení se objevuje u skenerů s rovinným zrcadlem rotujícím konstantní rychlostí. (b)
- Nelineární zkreslení v řádku je způsobeno rozdílnou vzdáleností snímaných objektů od zrcadla senzoru. (a)
- Otáčení zeměkoule při snímání mechanickým skenerem způsobuje chybu zkosení.(c)
- Obecné lichoběžníkové zkreslení ukazuje (d).
- Pokud se při snímání změní vzdálenost od pozorovaných objektů, projeví se zkreslení změnou měřítka. (e)
- Perspektivní zobrazení odpovídá nejjednoduššímu modelu dírkové komory aproximujícím vlastnosti objektivu kamery ve 3D vidění. (f)

Obr. 6.3 Obvyklé typy geometrických zkreslení.

6.2.5 Aproximace jasové funkce

Po geometrické transformaci souřadnic bodů následuje druhý krok, kde se hledá hodnota jasu každého transformovaného bodu. Předpokládejme, že jsme již geometrickou transformací vypočítali souřadnice *x',y'* bodů ve výstupním obrazu.

- Souřadnice *x, y* výchozích vzorků obrazové funkce *f(x, y)* byly celočíselné, protože byly identické se vzorkovací mřížkou.
- Po geometrické transformaci (např. otočení o obecný úhel) již souřadnice bodů v obecném případě celočíselné být nemusí.
- Jelikož také obraz po geometrické transformaci reprezentujeme maticí, máme předepsanou pravoúhlou vzorkovací mřížku i ve výsledném obraze.

Např. pro názornost si obrazovou funkci představme jako spojitou plochu (např. jako povrch pohoří), kde máme informaci o obrazové funkci (výšce pohoří) jen v bodech vzorkovací mřížky. Po geometrické transformaci nás zajímají hodnoty obrazové funkce (výšky transformovaného pohoří) v místech předepsaných rastrem výstupního obrazu. K dispozici ale máme jen vzorky obrazové funkce v rastru vstupního obrazu (pohoří). Principiálně správnou představou je aproximovat (též interpolovat) průběh plochy výstupního obrazu z dostupných vzorků a stanovit hodnoty v bodech předepsaného rastru. V teorii aproximace se obvykle dostupné vzorky proloží polynomem (zde polynomem dvou proměnných x, y). Tím získáme analytickou rovnici a po dosazení vypočteme hodnoty transformované obrazové funkce v předepsaném výstupním rastru obrazu. Přesnost interpolace ovlivňuje kvalitu výstupního obrazu. Přesnější postupy pro aproximaci používají větší okolí, tj. aproximují z více vzorků.

Úloha interpolace jasu se obvykle vyjadřuje způsobem duálním vzhledem k předpisu geometrické transformace. Aproximuje se jas ve vstupním obraze, který odpovídá jasu hledaného bodu ve výstupní mřížce. Předpokládejme, že chceme vypočítat jas odpovídající pixelu o celočíselných souřadnicích (*x'*, *y'*) ve výstupní mřížce (na následujících obrázcích bude tato mřížka znázorněna plnou čarou). Souřadnice bodů *(x, y)* ve vstupním obraze lze vypočítat

$$(x, y) = \mathbf{T}^{-1}(x', y') \tag{6.6}$$

Jak jsme již zmínili, v obecném případě po výpočtu získáme reálná čísla, která nesouhlasí s celočíselným rastrem.

Jedinou dostupnou informací o původně spojité obrazové funkci *f(x, y)* je její vzorkovaná verze $g_S(l\Delta x, k\Delta y)$. Abychom získali hodnotu jasu v bodě *(x, y),* aproximujeme vstupní obraz.
Výsledkem aproximace (interpolace) je jas $f_n(x, y)$, kde index *n* rozlišuje jednotlivé interpolační metody. Jas lze vyjádřit jako dvojrozměrnou konvoluci

$$f_n(x, y) = \sum_{l=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} g_S(l\Delta x, k\Delta y) h_n(x - l\Delta x, y - k\Delta y) \tag{6.7}$$

h_n - interpolační jádro. Obvykle se používá interpolační jádro pokrývající jen malé okolí zpracovávaného bodu, aby se ušetřily výpočty. Vně tohoto okolí je hodnota jádra h_n nulová.

Obecný popis transformace

$$g(x,y) = f[\varphi(x,y), \psi(x,y)] \qquad (6.8)$$

Geometrická transformace je zcela zadána funkcemi $\varphi(x,y)$ a $\psi(x,y)$

Z uvedených vztahů lze vytvořit návod jak takovouto transformaci provést.

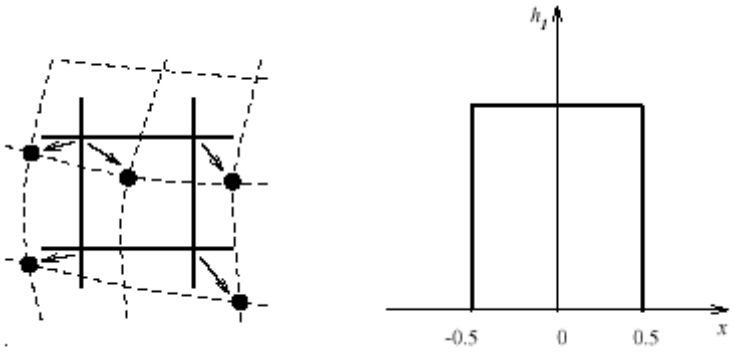
- Předpokládejme, že obrazy *f(x,y)* i *g(x,y)* jsou diskrétní.
- Na základě výše uvedených vztahů lze postupně probírat všechny body výstupního obrazu *g* a pro každý bod *(x,y)* určit odpovídající polohu $[\varphi(x,y), \psi(x,y)]$ ve vstupním obraze *f*. Hodnota jasu $f[\varphi(x,y), \psi(x,y)]$ je pak vzata jako hodnota g(x,y). x,y – celočíselné hodnoty (postupujeme po celočíselných souřadnicích pixelů výstupního obraz) $\varphi(x,y)$, $\psi(x,y)$ - reálné nikoli celočíselné.

  Podle předpokladu je i vstupní obraz *f* diskrétní, a proto i pro něj jsou hodnoty jasu známy jen v bodech o celočíselných souřadnicích. Pro získání hodnot v mezilehlých bodech o neceločíselných souřadnicích lze využít metody jako jsou
  - bilineární interpolace
  - interpolace vyšších řádů

V praxi se pro interpolaci používají jednoduché **aproximační polynomy**. Uvedeme z nich tři od nejjednodušších ke složitějším:

Metoda nejbližšího souseda

- přiřadí bodu *(x, y)* hodnotu jasu nejbližšího bodu g v diskrétní mřížce.



Obr. 6.4 Interpolace metodou nejbližšího souseda.

V pravé části Obr.6.4 je znázorněno interpolační jádro h_1 v jednorozměrném případě. Levá strana Obr. 6.4 přibližuje, jak je přiřazena nová hodnota jasu. Čárkované čáry ukazují, jak je celočíselný rastr výstupního obrazu zobrazen inverzní rovinnou transformací. Plné čáry ukazují vzorkovací mřížku vstupního obrazu. Interpolace metodou nejbližšího souseda je dána

$$f_1(x,y) = g_s(\mathrm{round}(x), \mathrm{round}(y))$$

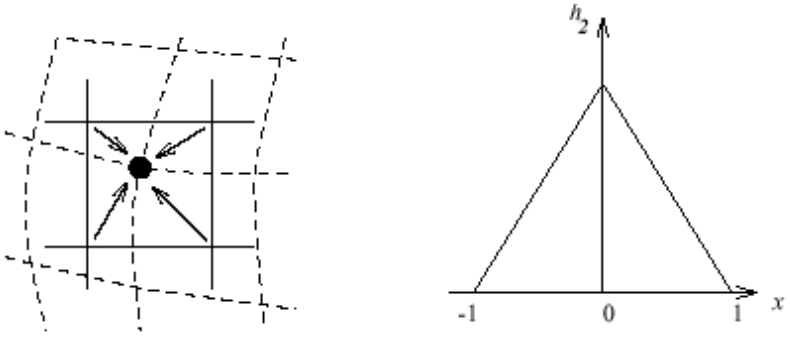
round - zaokrouhlený

Chyba v poloze bodu po interpolaci metodou nejbližšího souseda je nejvýše půl pixelu. Tato chyba je v obrázcích dobře zřetelná u objektů s přímočarými obrysy natočenými šikmo vůči rastru, které jsou schodovité.

Lineární interpolace
- využije okolí čtyř bodů sousedících se zpracovávaným bodem *(x, y)* a předpokládá, že obrazová funkce je lineární kombinací jasu těchto čtyř bodů. Vliv každého ze čtyř bodů v lineární kombinaci je úměrný jeho blízkosti ke zpracovávanému bodu.
Lineární interpolace je znázorněna na Obr. 6.5. Levá část obr. ukazuje, jak se čtyři body v okolí podílejí na interpolaci, a pravá část demonstruje interpolační jádro v jednorozměrném případě. Lineární interpolace je dána vztahem

$$f_2(x,y) = (1-a)(1-b)g_s(l,k) + a(1-b)g_s(l+1,k) + b(1-a)g_s(l,k+1) + abg_s(l+1,k+1)$$

kde l = round(x), a = x-l
k = round(y), b = y-k

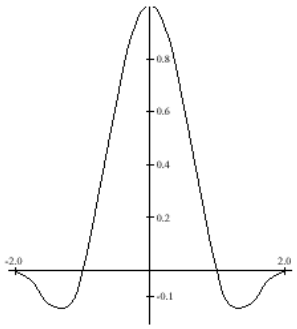


Obr. 6.5 Lineární interpolace.

Lineární interpolace způsobuje malý pokles rozlišení, protože obraz mírně rozmazává díky tomu, že hodnoty obrazu jsou vyhlazovány lineárním filtrem. Rozmazání je ovšem menší zlo než schodkovitost u interpolace metodou nejbližšího souseda. Zlepšení je nejvíce patrné na šikmých čarách.

Bikubická interpolace
- model obrazové funkce tím, že je lokálně interpolován bikubickým polynomem. Pro výpočet se používá okolí složené z 16 bodů.
Jelikož by dvojrozměrné znázornění bylo nepřehledné, ukážeme interpolační jádro v jednorozměrném případě

$$h_3 = \begin{cases} 1-2|x|^2+|x|^3 & \text{pro} \quad 0 \le |x| < 1 \\ 4-8|x|+5|x|^2-|x|^3 & \text{pro} \quad 1 \le |x| < 2 \\ 0 & \text{jinde} \end{cases}$$



Obr. 6.6 Interpolační jádro pro bikubickou interpolaci.

Interpolační jádro je vidět na obr. 6.6. Jeho tvar by ve 2D připomínal mexický klobouk. Bikubická interpolace zachovává mnohem lépe detaily v obraze než dvě výše zmíněné metody.
Používá se v rastrových displejích v zobrazovacích programech, které dovolují zoom v libovolném bodě.

Shrnutí pojmů

Geometrické transformace nám slouží k transformaci jednoho obrazu na druhý. Může při ní dojít ke změně původního obrazu. Mezi základní geometrické transformace patří translace, rotace, zkosení apod. Probíhá ve 2 základních krocích – **transformace souřadnic bodů** a **aproximace jasové stupnice**.

Transformace souřadnic bodů převádí souřadnice původního obrazu na souřadnice nového obrazu. **Následuje transformace jasové stupnice**, kdy hdnoty jasu nového pixelu jsou vypočteny na základě jasu původního pixelu.

Otázky

1. Vysvětlete pojem geometrická transformace a vyjmenujte některé geometrické transformace.
2. Vysvětlete jak probíhá transformace souřadnic bodů.
3. Co to je a jak probíhá aproximace jasové funkce pokud použijeme metody nejbližšího souseda?

CVIČENÍ

Geometrické transformace

K realizaci geometrické transformace obrazu slouží v Simulinku bloky knihovny Geometric Transformations library, kterou nalezneme ve Video and Image Processing Blockset. Bloky k této transformaci užívají interpolace pro kalkulaci příslušné hodnoty pixelu, a proto je možné provést rotaci, změnu velikosti, přeložení nebo střižení obrazu. Při výběru konkrétní metody máme možnost užití třech základních metod přepočtu obrazu. Je to metoda nejbližší soused, dále pak bikubická a konečně bilineární transformace, které jsou popsány níže v textu.

V průběhu cvičení budeme pracovat s obrázkem, na kterém si postupně ukážeme různé způsoby úpravy obrazu.

Obr. 6.7 Originál obrázku

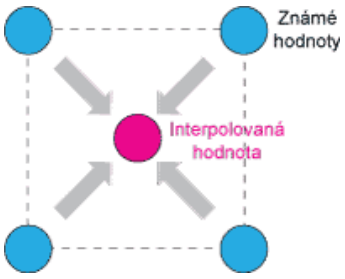
Interpolační metody

Nejbližší soused

Jedná se o nejjednodušší metodu interpolace, která jednoduše využije sousední pixel pro dopočet. Výsledným efektem je transformace obrazu, která připomíná zvětšení plochy jednotlivých pixelů. Tato metoda sice zachovává ostrost hran v obraze, ale má také velmi silný aliasing efekt. Výsledné hrany jsou zubaté. Z toho je zřejmé, že tato metoda se příliš nehodí pro zvětšování obrazu.

Bilineární interpolace

Bilineární interpolace používá jednoduše čtyři nejbližší sousední pixely (2 x 2) a z nich vypočítá váženým průměrem nový pixel uprostřed nich. Tato metoda má silný anti-aliasing účinek a výsledek je proto mnohem jemnější než u metody nejbližší soused. Jelikož se přepočítaný pixel vytváří jako průměr jeho sousedů, je výsledný sníme viditelně rozostřen. Proto je potřeba po aplikaci této interpolace provést následně ještě doostření obrazu.

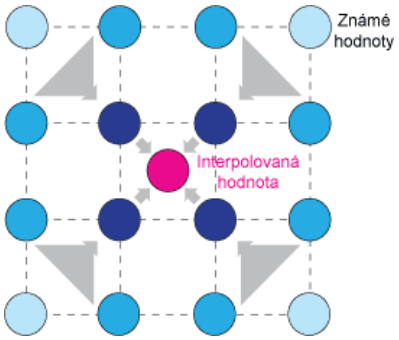


Obr. 6.8 Bilineární interpolace bodu

Bikubická interpolace

Bikubická interpolace je vyšší metodou interpolace který oproti bilineární používá pro stanovení hodnoty pixelu matici (4 x 4) tedy celkem 16 okolních pixelů. Váhy jednotlivých pixelů okolí jsou rozděleny dle vzdálenosti. Nejvyšší váhu má pixel s nejmenší vzdáleností. Tato metoda má nejsilnější

anti-aliasing účinek a výsledkem jsou velmi jemné přechody. Je považována za standardní metodu a je vhodná jak pro zvětšování tak zmenšování snímků. Díky větším počtu zpracovávaných bodů je však výpočetně náročnější. Hodnota pixelu je vypočítána stejně jako u bilineární interpolace jako průměr svých sousedů. Při zvětšení snímku z zde viditelné rozostření snímku a je potřeba jej následně doostřit.

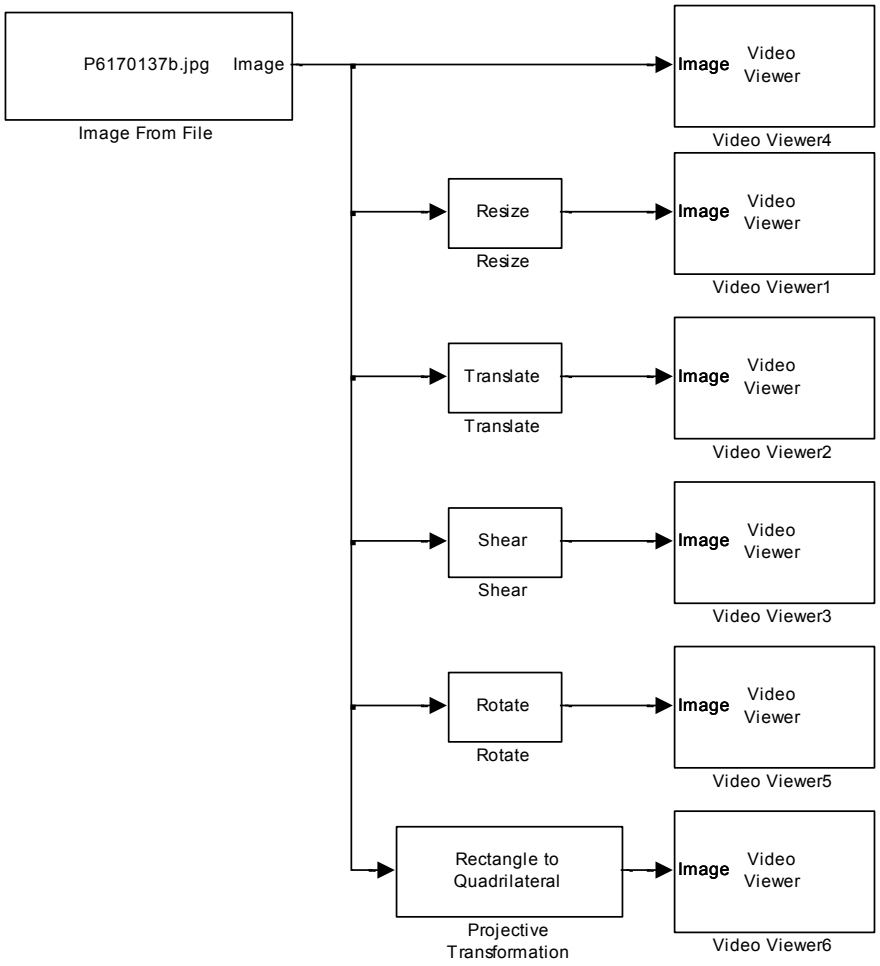


Obr. 6.9 Bilineární interpolace bodu.

Změna velikosti obrazu

Změna velikosti může být provedena jak pro celý obraz, tak pro oblast uvnitř obrazu. K určení této změny slouží parametr Specify. Vstupním obrazem mohou být data ve formátu skalárního čísla, které udává procentuální změnu. Tato změna je následně aplikována na jednotlivé řádky a sloupce matice obrazu. Změnu velikosti obrazu je také možné zadat pomocí vektoru, a to buď samostatně pro řádky a sloupce nebo pro obě složky současně. Zadáním jednotlivých složek však může dojít k deformaci původního obrazu, a to vlivem nedodržení poměru velikosti sloupců a řádků. Dalším parametrem bloku Resize je Interpolation method, který definuje použitou metodu transformace obrazu. Blok Resize také pracuje s parametrem Preform Antialiasing, který zajistí ořez nerovností na hranách způsobených změnou velikosti obrazu. V podstatě se postará o vyhlazení hran.

Pro aplikaci změny velikosti obrazu nakreslíme v Simulinku model dle následujícího schéma na Obr.6.10. Ze schématu zakreslíme pouze část s využitím bloku Resize.



Obr. 6.10 Blokové schéma pro geometrické transformace obrazu.

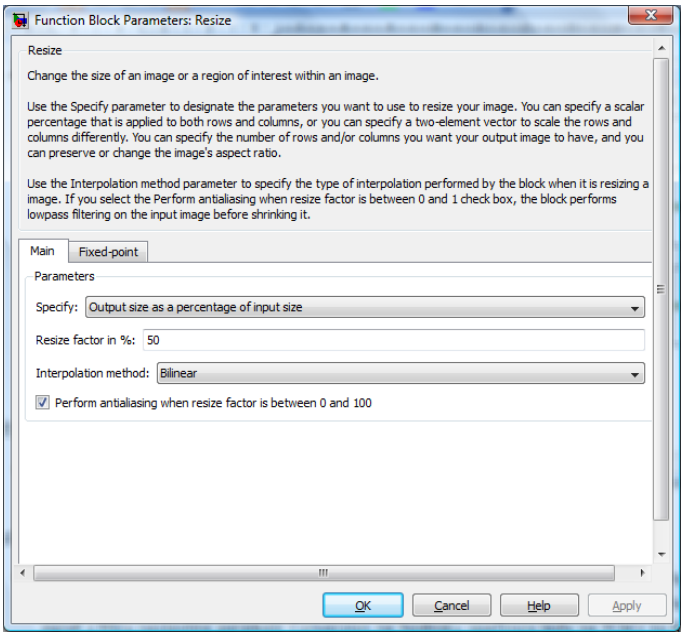
Jednotlivé bloky nalezneme

- Blok ***Resize*** nalezneme v menu ***Video and Image processing Blockset -> Geometric transformations***.

Nastavení vstupních parametrů

- V bloku ***Image from File*** nastavíme parametr **Filename** pro zpracování požadovaného obrázku. Ten vybereme z pevného disku pomocí tlačítka Browse. Ostatní parametry ponecháme defaultně přednastaveny.

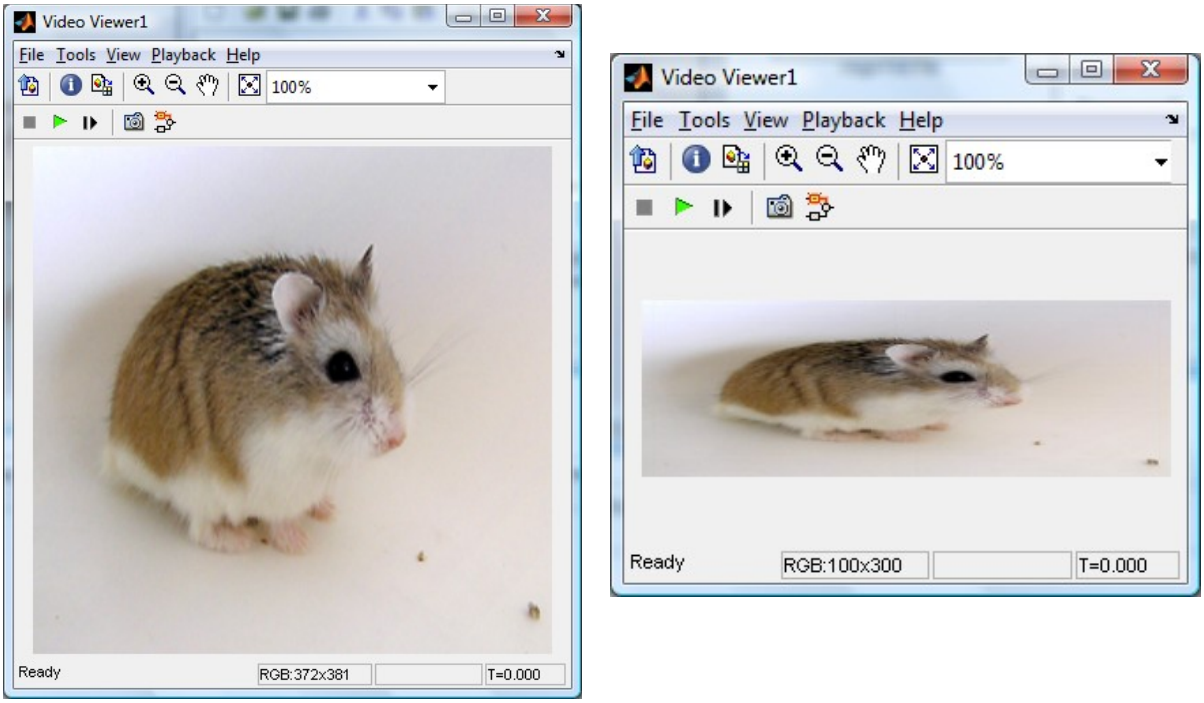
- Parametry bloku ***Resize*** nastavíme podle následujícího obrázku (Obr.6.11). V záložce main nastavíme parametr **Specify** na hodnotu **Output size as a percentage of input size**. Parametr **Resize faktor in %**, který udává procentuální změnu výstupního obrazu, nastavujeme dle potřeby. V našem případě použijeme číselnou hodnotu **50**. Parametr **Interpolation method**, který udává metodu transformace obrazu, ponecháme přednastaven na hodnotu **Bilinear**. Dále zaškrtneme checkbox **Perform antialiasin when resize factor is between 0-100**, čímž dosáhneme zahlazení hran v rozsahu změny 0-100 procent.



Obr. 6.11 Nastavení bloku Resize v záložce Main.

- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nyní propojíme jednotlivé bloky dle schématu na Obr.6.10 a vytvoříme model v Simulinku. Nastavíme parametry simulace a můžeme spustit simulaci.



Obr. 6.12 Vlevo obrázek upravený zadáním parametru Output size as a percentage of input size, vpravo obrázek upravený zadáním Number of outputs rows and columns.

Posun obrazu

K této operaci je v Simulinku určený blok ***Translate.*** Tento blok provede posunutí obrazu v horizontální a vertikální rovině. Pro správnou funkci bloku je zapotřebí specifikovat jeden ze dvou parametrů, a to buď **Offset vektor** nebo **Offset port**. Parametr **Offset Vektor** definuje vektor posunutí. První složka představuje posun v horizontální rovině. Zadáním kladných hodnot se obraz posouvá směrem dolů, zadáním záporných hodnot se posouvá směrem nahoru. Druhá složka vektoru definuje posun ve vertikální rovině. Zadáním kladných hodnot dojde k posunu směrem doprava, zadáním záporných hodnot se obraz posune směrem doleva.

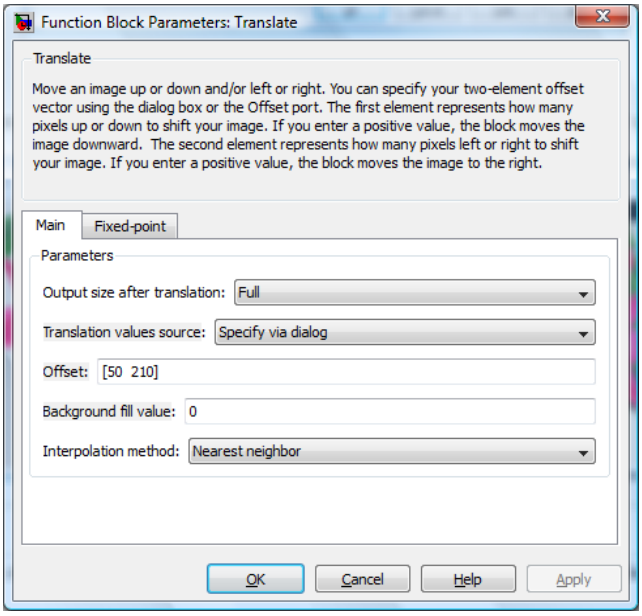
Jednotlivé bloky nalezneme

- Blok ***Translate*** nalezneme v menu ***Video and Image processing Blockset -> Geometric transformations***.

Nastavení vstupních parametrů

- V bloku ***Image from File*** nastavíme parametr **Filename** pro zpracování požadovaného obrázku. Ten vybereme z pevného disku pomocí tlačítka Browse. Ostatní parametry ponecháme defaultně přednastaveny.

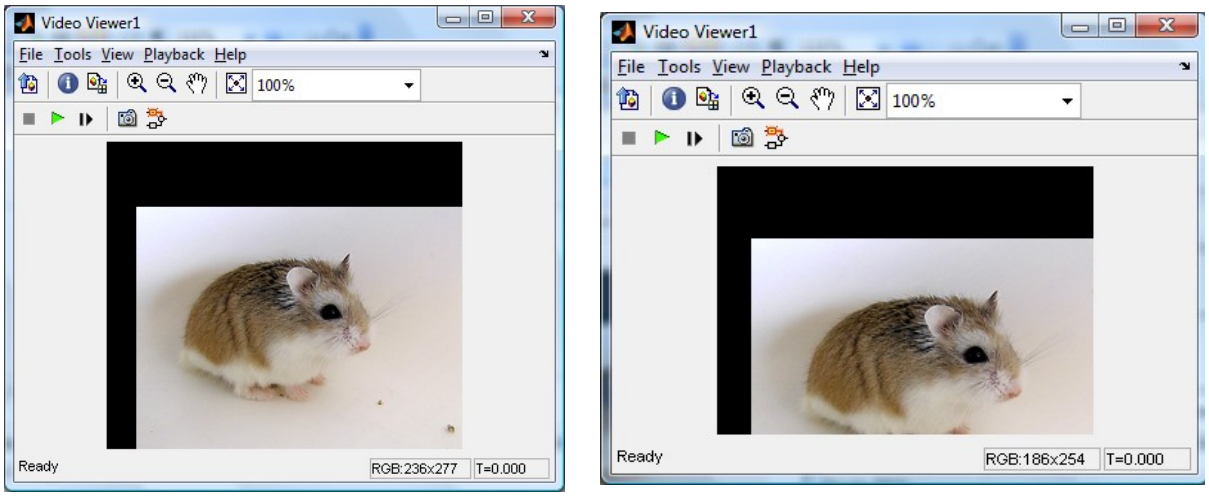
- Parametry bloku ***Translate*** nastavíme podle obrázku Obr.6.13. V záložce Main nastavíme parametr **Output size after translation** na hodnotu **full**. Parametr **Translation value source** ponecháme v původním nastavení na hodnotě **Specify Via dialog**. Parametrem **Offset** nastavíme posunutí v jednotlivých směrech. První složka vektoru posune obrázek v horizontální rovině, druhá složka zajistí posun ve vertikální rovině. Parametr **Background fill value**, který představuje pozadí obrázku, ponecháme defaultně přednastaven. Dalším parametrem je **Interpolation method**, který definuje jednu ze tří interpolačních metod. Ponecháme defaultně přednastavený na hodnotu Nearest neighbor. Pro zbylé dvě metody v nabídce (bilineární a bikubická) provedete zapojení a nastavení stejným způsobem jako pro metodu Nearest neighbor.



Obr. 6.13 Nastavení bloku Translate v záložce Main.

- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nyní jednotlivé bloky propojíme podle schématu na Obr.6.10 a vytvoříme model. Definujeme parametry simulace a můžeme spustit simulaci.



Obr. 6.14 Vlevo původní obrázek při nastavení parametru Output size after translation na hodnotu full, vpravo při nastavení na Same as input image .

Zkosení obrazu

Blok ***Shear*** provádí zkosení obrazu. Deformace obrazu je provedena lineárně v závislosti na rostoucí nebo klesající vzdáleností řádků a sloupců v původním obraze vůči zadané změně. Parametr Shear direction určuje horizontální nebo vertikální osu posunutí. Je možné jej také zadat pomocí dvousložkového vektoru ve tvaru [první poslední]. První prvek definuje počet pixelů, o které se posune první sloupec nebo řádek matice obrazu, druhý prvek definuje počet pixelů, o které se posouvá poslední sloupec nebo řádek.
Dle schématu na Obr.6.10 zakreslíme model v Simulinku. Pro zjednodušení použijeme pouze bloky Image From File, Shear a Video Viewer.

Jednotlivé bloky nalezneme

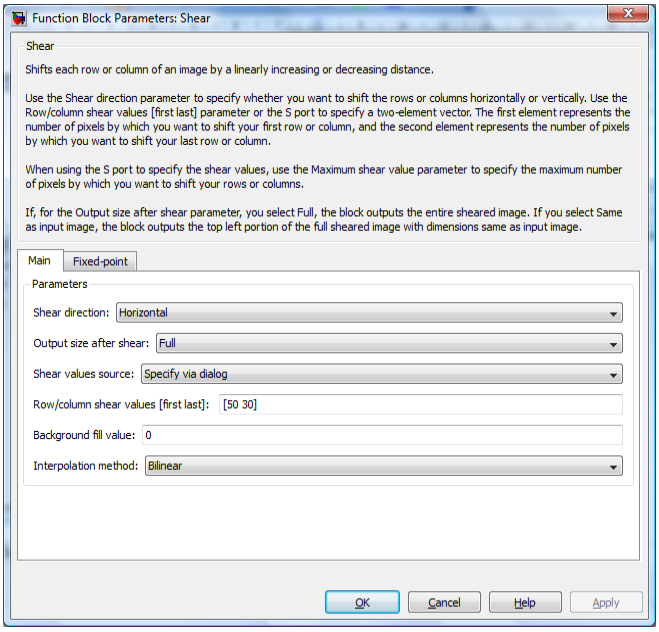
- Blok **Shear** nalezneme v menu ***Video and Image processing Blockset*** -> ***Geometric transformations***.

Nastavení vstupních parametrů

- V bloku ***Image from File*** nastavíme parametr **Filename** pro zpracování požadovaného obrázku. Ten vybereme z pevného disku pomocí tlačítka Browse. Ostatní parametry ponecháme defaultně přednastaveny.

- Parametry bloku ***Shear*** nastavíme dle Obr.6.15. Parametr **Shear direction** nastavíme na hodnotu **Horizontal**, jenž prezentuje změnu v horizontální rovině, tedy na úrovni řádků. Parametr **Output size after shear** nastavíme na hodnotu **Full**, který zajistí, že původní obrázek nebude ořezán. Parametr **Shear values source** ponecháme defaultně přednastaven na hodnotu **Specify via dialog**. Parametr **Row/Column shear value[first last]**, který definuje vektor zkosení, nastavíme na hodnotu **[50 30].** Parametr **Background fil value** ponecháme

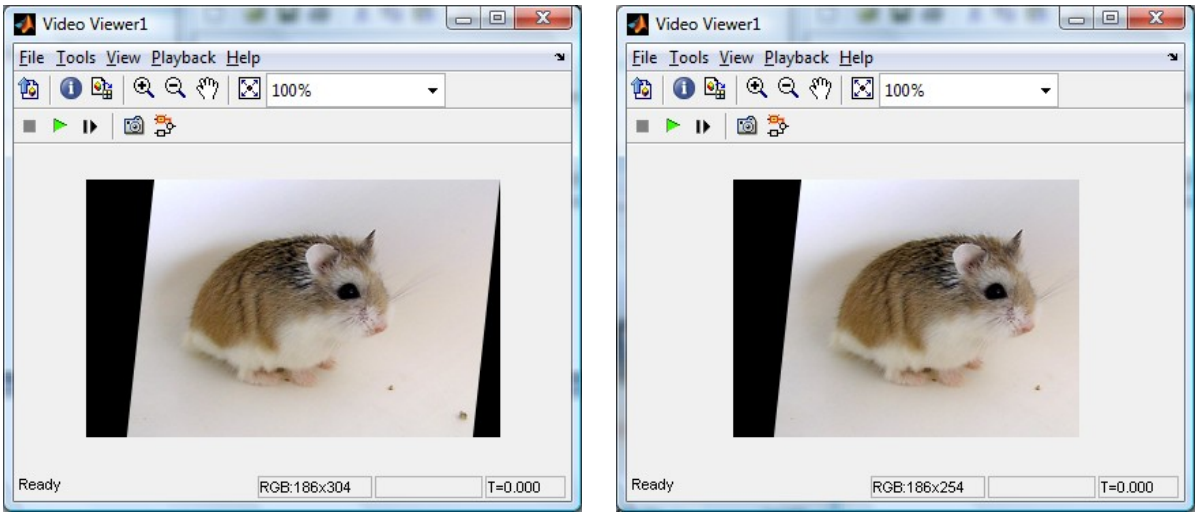
defaultně přednastaven na hodnotu **0**. Parametr **Interpolation method** nastavíme na hodnotu **Bilinear**.



Obr. 6.15 Nastavení bloku Shear v záložce Main.

- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nyní jednotlivé bloky propojíme podle schématu na Obr.6.10 a vytvoříme model. Definujeme parametry simulace a můžeme spustit simulaci.



Obr. 6.16 Vlevo původní obrázek při nastavení parametru Output size after shear na hodnotu full, vpravo nastavení na Same as input image.

Rotace obrázku

Úhel otočení definuje parametr **Angle**, který je nutné zadat v radiánech, a to v rozsahu hodnot 0 až násobků čísla *pi*. Velikost výstupu definuje parametr **Output size**. Zadáním hodnoty **Expanded to fit roted input image** je otočený obrázek uložen v plné velikosti. Matice výstupních hodnot obsahuje, kromě původních hodnot a otočení, také nulové body, které se zobrazí jako černé pozadí. Viz. Obr.6.18 vlevo. Zadáním hodnoty **Same as input image** dojde k ořezu obrazu. Matice výstupních hodnot a nul je zobrazena ve velikosti vstupního obrázku a tudíž dojde k ořezu některých otočených hodnot. Viz. Obr.6.18 vpravo.

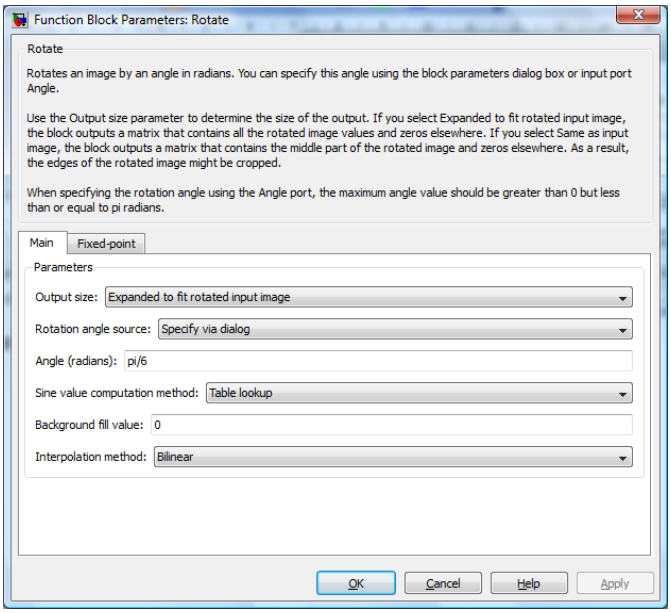
Jednotlivé bloky nalezneme

- Blok ***Rotate*** nalezneme v menu ***Video and Image processing Blockset -> Geometric transformations***.

Nastavení vstupních parametrů

- V bloku ***Image from File*** nastavíme parametr **Filename** pro zpracování požadovaného obrázku. Ten vybereme z pevného disku pomocí tlačítka Browse. Ostatní parametry ponecháme defaultně přednastaveny.

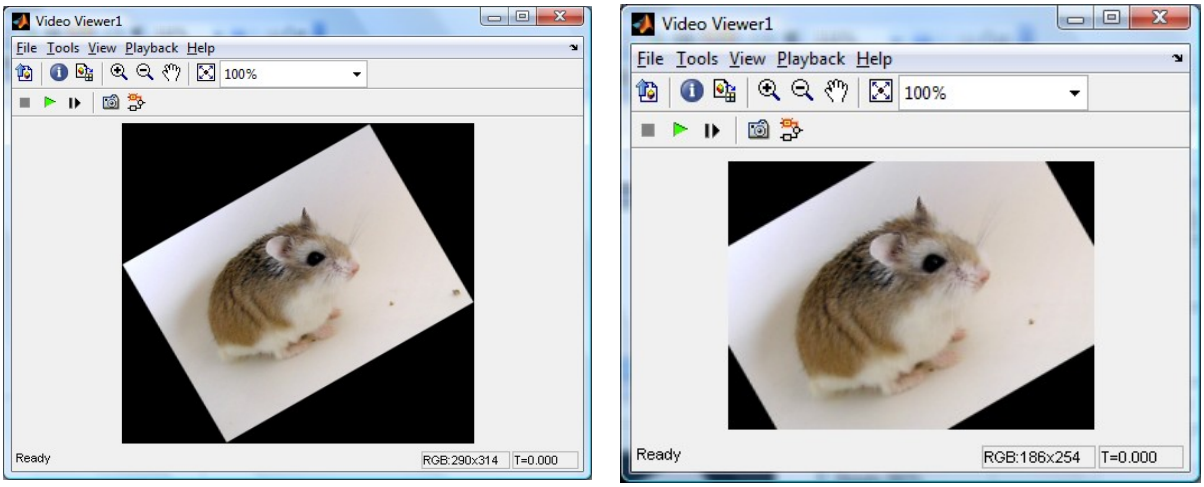
- V bloku ***Rotate*** nastavíme vstupní parametry dle Obr.6.17. Parametr **Output size** nastavíme na hodnotu **Expanded to fit rotated input image**. Parametr **Rotation angle source** ponecháme přednastaven na hodnotě **Specify via dialog**. Parametr **Angle(radians)** nastavíme na hodnotu **pi/6**. Parametr **Interpolation method** nastavíme na hodnotu **Bilinear**. Ostatní parametry ponecháme defaultně přednastaveny.



Obr. 6.17 Nastavení bloku Translate v záložce Main.

- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nyní jednotlivé bloky propojíme podle schématu na Obr.6.10 a vytvoříme model. Definujeme parametry simulace a můžeme spustit simulaci.



Obr. 6.18 Vlevo obrázek s použitím Expanded to fit rotated input image , vpravo použitím nastavení Same as input image.

Projekce obrazu

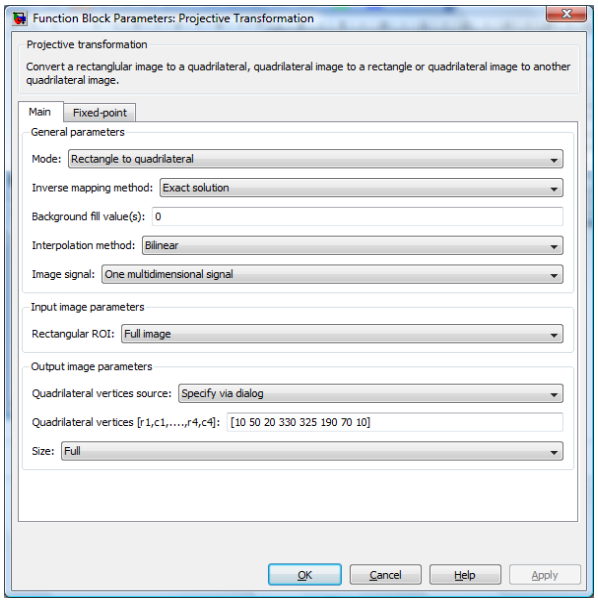
Pomocí bloku *Projective Transformation* provedeme změnu poměru jednotlivých stran obrazu.
Jednotlivé bloky nalezneme

- Blok nalezneme v menu ***Video and Image processing Blockset*** -> ***Geometric transformations***.

Nastavení vstupních parametrů

- V bloku ***Image from File*** nastavíme parametr **Filename** pro zpracování požadovaného obrázku. Ten vybereme z pevného disku pomocí tlačítka Browse. Ostatní parametry ponecháme defaultně přednastaveny.

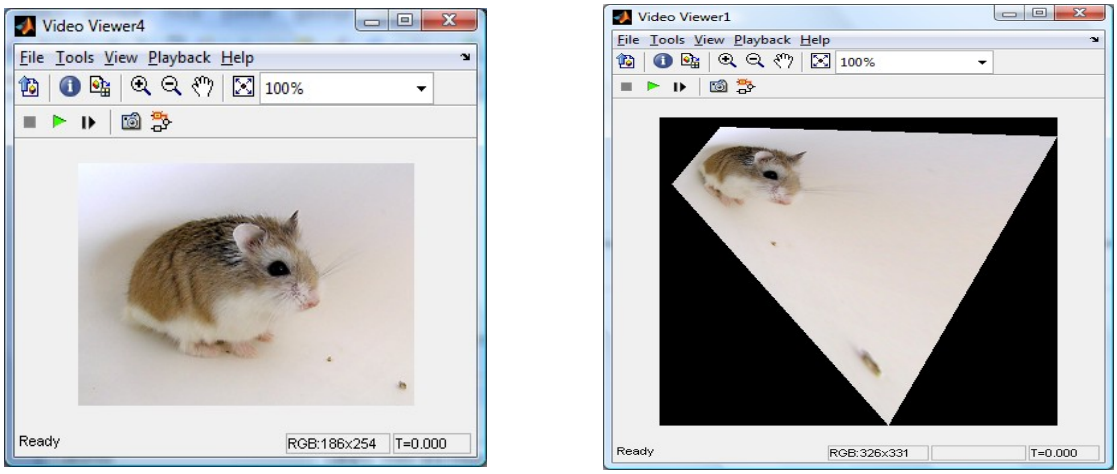
- Parametry bloku ***Projective Transformation*** nastavíme podle Obr.6.19. Záložka Main je rozdělena do tří částí. **General parameters** prezentují základní parametry nastavení modu, pozadí, interpolační metodu a také nastavení obrazového signálu. Tyto parametry ponecháme defaultně přednastaveny. **Input image parameters** prezentují vstupní parametry obrázku, které ponecháme defaultně přednastaveny. **Output image parameters** prezentují výstupní parametry obrazu, které taktéž ponecháme defaultně přednastaveny.



Obr. 6.19 Nastavení záložky main.

- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nyní jednotlivé bloky propojíme podle schématu na Obr.6.10 a vytvoříme model. Definujeme parametry simulace a můžeme spustit simulaci.



Obr. 6.20 Vlevo původní obrázek , vpravo s použitím projekce.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 06 Geometricke_transformace.swf.

7. TRANSFORMACE OBRAZOVÝCH SIGNÁLŮ, FOURIEROVA TRANSFORMCE (FT), POUŽITÍ FT

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat základní význam FT pro zpracování obrazu
- popsat spektra digitálního obrazu
- vyřešit zadanou úlohu v Matlab/Simulink a VIP týkající se FT

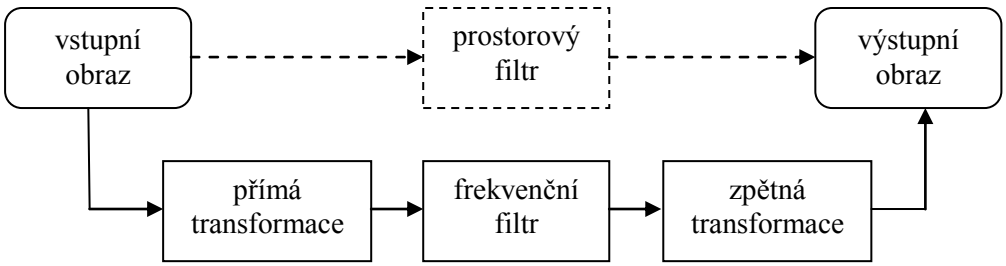
Výklad

7.1 Úvod

Fourierova transformace je matematická metoda, která je úspěšně použitelná v oblasti digitálního zpracování a analýze obrazu (signálu). V obecném případě se jedná o vyjádření funkce popisující obraz v jiných proměnných pomocí integrální transformace. S výhodou lze FT použít v oblastech detekce hran, úpravy kvality obrazu, rekonstrukce obrazu, komprese obrazu, detekce objektů aj. Učební text této kapitoly převzat a upraven podle [2].

7.2 Diskrétní lineární integrální transformace

Oblast digitálního zpracování obrazu za použití technik integrálních transformací (v našem případě FT) lze považovat za jedny ze základních přístupů pro dosažení výsledků analýzy daného obrazu.



Obr. 7.1 Obraz lze lineárními filtry zpracovat buď v prostorové, nebo frekvenční oblasti.

V této kapitole se budeme opírat o představu filtru jako bloku zpracování (viz obr. 7.1), na jehož vstupu i výstupu je dvojrozměrný signál (obraz). Jelikož nyní budeme mít na mysli lineární filtry, budeme se moci vydat dvěma ekvivalentními cestami:

Filtrace v prostorové oblasti

- obraz je zpracováván jako lineární kombinace vstupního obrazu s koeficienty filtru (často s mnohem menším definičním oborem než obraz, tzv. lokální filtry).

- základním matematickým nástrojem bude konvoluce.
- tento přístup je na Obr. 7.1 označen čárkovaně.

Filtrace ve frekvenční oblasti

- nejdříve se převede obraz pomocí lineární integrální transformace do frekvenčního spektra, kde se filtruje a výsledek filtrace se inverzní lineární integrální transformací převede zpět na obraz.
- v mnoha případech je vyjádření filtru ve frekvenční reprezentaci názornější.
- na Obr. 7.1 je cesta přes frekvenční reprezentaci naznačena plnou čarou.

7.3 Fourierova transformace

Fourierova transformace je modifikací Fourierovy řady a je užitečná pro řešení mnoha různých problémů. Používá se např. pro převedení řešení diferenciálních rovnic na řešení algebraických rovnic nebo pro frekvenční analýzu časově proměnných signálů. V oblasti zpracování obrazů je možné Fourierovu transformaci uplatnit pro úpravy kvality obrazů, ale také pro vyhodnocování prostorových frekvencí. Dvojrozměrná Fourierova transformace umožní převést rozložení obrazových intenzit *f(x, y)* vyhodnocovaného obrazu na obraz prostorových frekvencí *F(fx, fy)*. Dvojrozměrnou Fourierovu transformaci definuje vztah

$$F(f_x, f_y) = \iint_{-\infty}^{\infty} I(x,y).\exp[-2\pi i(xf_x + yf_y)]\,dxdy \tag{7.1}$$

Fourierův integrál:

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt \tag{7.2}$$

Jedná se o komplexní integrál s parametrem, který definuje transformaci (obecně komplexní) funkce *f(t)* na její Fourierův obraz *F(ω)*. Zpětná transformace je dána vztahem:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega \tag{7.3}$$

Diskrétní Fourierova transformace a její inverze:

$$A_k = \sum_{j=0}^{n-1} a_j e^{-i\frac{2\pi}{n}jk} \tag{7.4}$$

kde k=0,1,2 ... n-1

$$a_j = \frac{1}{n}\sum_{k=0}^{n-1} A_k e^{i\frac{2\pi}{n}jk}$$

(7.5)

kde j=0,1,2 ... n-1
Výsledkem Fourierovy transformace reálné funkce je komplexní funkce

$$F(u, v) = Re(u, v) + jIm(u, v) \tag{7.6}$$

kde *Re(u, v),* resp. *Im(u, v)* jsou reálnou a imaginární složkou *F(u, v)*.
Modul (velikost) |*F(u, v)*| se nazývá **amplitudové frekvenční spektrum** obrazu *f(m, n)*.

$$|F(u,v)| = \sqrt{Re^2(u,v) + Im^2(u,v)} \tag{7.7}$$

Fázové spektrum $\phi(u, v)$

$$\phi(u,v) = \tan^{-1}\left[\frac{Im(u,v)}{Re(u,v)}\right] \tag{7.8}$$

Výkonové spektrum *P(u, v)*.

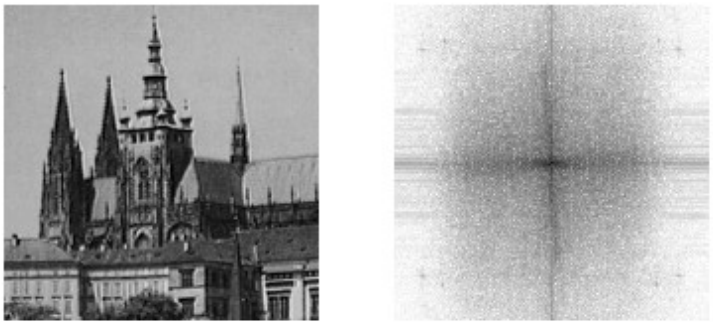
$$P(u,v) = |F(u,v)|^2 = Re^2(u,v) + Im^2(u,v) \tag{7.9}$$

Poznamenejme, že je potřeba dávat pozor na znaménka kvadrantů, protože funkce *tan* je periodická s periodou π a ϕ probíhá od 0 do 2π.

Příklad obrazu (v prostorové oblasti) a jeho frekvenčního spektra (ve frekvenční oblasti) je na Obr. 7. 2. Všimněte si, že ve spektru je patrný kříž se středem ve středu obrázku. Diskrétní Fourierova transformace předpokládá, že obraz je jedna perioda 2D periodické funkce. Přepokládá tedy, že okraje obrazu budou stejné a budou na sebe navazovat. Jestliže tento předpoklad není splněn, jako v případě obrazu Pražského hradu, jsou ve Fourierovském obrazu potřebné vysoké frekvence, aby bylo možné rekonstruovat nespojitosti na krajích obrazu. Ve spektru na Obr. 7.2 odpovídá nespojitostem centrální kříž. Jde o analogii případu obdélníkového průběhu, pro jehož rekonstrukci bychom potřebovali nekonečně mnoho harmonických složek.[2]

Při pečlivějším pohledu na spektrum si lze všimnout, že kříže jsou dva. Výraznější kříž se kryje s osami *u, v* spektra a vznikl transformací okrajů intenzitního obrazu. Druhý kříž je natočen asi o 10° proti směru hodinových ručiček vůči prvnímu kříži a nedosahuje až do krajů obrazu. Pochází z převažujících směrů jasových rozhraní v obrázku (gradientu obrazové funkce).

Pozn.: Hranice objektů vnímaných v obraze člověkem jsou pootočeny proti směru jasových rozhraní (ve frekvenčním spektru jde o harmonické bázové funkce). Proto svislé směry kříže ve spektru odpovídají vodorovným hranám v intenzitním obrázku a naopak.



(a) (b)

Obr. 7.2 Fourierovo frekvenční spektrum: (a) Vstupní obraz, (b) spektrum jako intenzitní obraz – tmavé pixely odpovídají vysokým spektrálním hodnotám. [2]

7.3.1 FFT spektrum obrázku

V oblasti digitálního zpracování obrazu můžeme pracovat se stejnými matematickými interpretacemi a postupy jako v oblasti zpracování analogových signálů (týká se to FFT). Na tomto základě je možné říct, že i digitální obrazy mají stejně jako signály spektra, ačkoli jsou interpretovány trošku odlišně.

Signály

- Složky s nízkými frekvencemi - Pomalé změny signálu
- Složky s vysokými frekvencemi - Rychlé změny signálu

Obrazy

- Nízkofrekvenční složky - Pomalý přechod mezi šedými úrovněmi
- Vysokofrekvenční složky - Hrany

Spektrum zahrnuje jak amplitudové tak fázové spektrum.

Protože máme dvou dimenzionální objekt spektrum obsahuje frekvenční data ve dvou směrech:

- podél řádků obrázku
- podél sloupců obrázku

Amplituda a fáze je pak prezentována ve 3. dimenzi. Nyní si ukážeme spektra pro tři jednoduché obrázky:

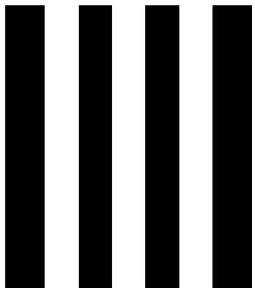
- vertikální pruhy
- horizontální pruhy
- šachovnice

V amplitudovém spektru každého obrázku se odráží frekvenční obsah obrázku.

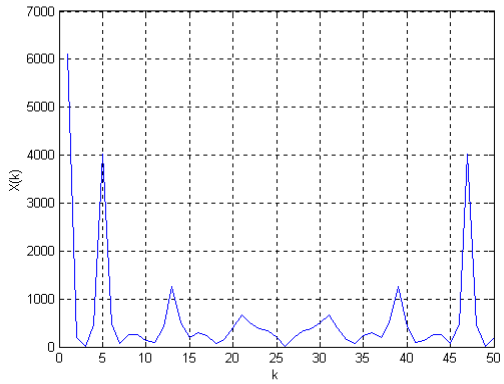
Pozn.: 2D DFT:

$$X[i,k] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] \exp(-j2\pi \frac{im}{M}) \exp(-j2\pi \frac{kn}{N}) \qquad (7.10)$$

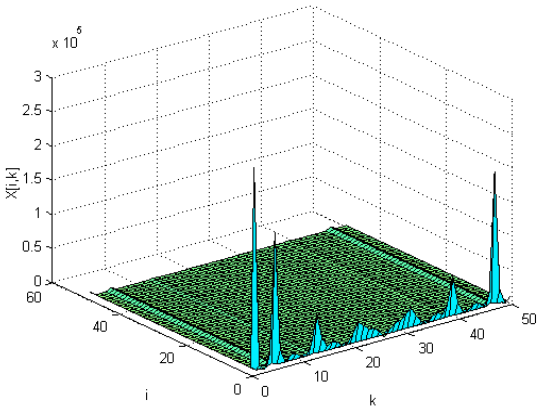
Příklad 7.1: Vertikální pruhy



Obr. 7.3 Matice o velikosti 50 x 50 pixelů.



Amplitudové spektrum |X[k]|. (řádky)



Amplitudové spektrum |X[i,k]|

Obr.7.4 Vykreslení amplitudového spektra u 2D DTF

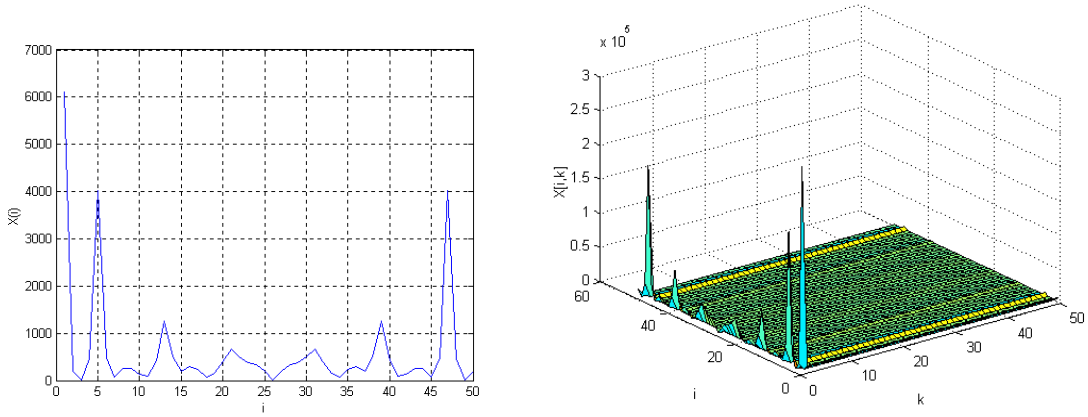
Amplitudové spektrum je obdrženo z 2D DFT obrázku.

Z Obr 7.3 a 7.4 můžeme vidět, že jde o 2-krokový proces. V 1. kroku jsou DFT brány podél řádků obrázku. V libovolném řádku se pixely střídají (bílé, černé), a tak DFT produkuje pro libovolný řádek spektrum obdélníkové vlny s obálkou sinus cardinalis.
Amplitudové spektrum pro každý řádek m může být popsáno jako $|X_m[k]|$ v závislosti na k, toto nám dává frekvenční zastoupení v řádku m obrázku.
Ve 2. kroku směřujeme k výpočtu 2D DFT. Výpočet je brán podél sloupců této matice a získáváme konečný výsledek $|X[i,k]|$. Poněvadž spektra $|X_m[k]|$ jsou identická pro každý řádek m, pak každý sloupec n matice 50 x 50 obsahuje (rozdílnou konstantní hodnotu). DFT konstantního signálu

odpovídá pouze stejnosměrné (DC, konstantní) složce, která vyjadřuje, že se v každém sloupci *k* objevuje pouze jeden nenulový spektrální element, v indexu *i = 0*, který koresponduje s DC složkou.

Příklad 7.2: Horizontální pruhy

Obr. 7.5 Matice o velikosti 50 x 50 pixelů.

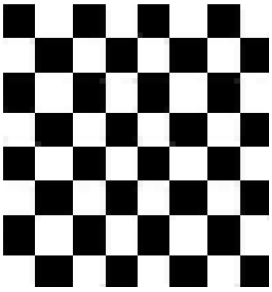


Amplitudové spektrum |X[i]|. (sloupce) Amplitudové spektrum |X[i,k]|

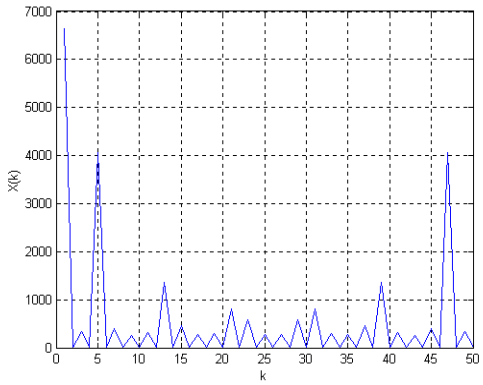
Obr.7.6 Vykreslení amplitudového spektra

Z Obr. 7.5 je patrné, že každý řádek obrázku má konstantní, DC, hodnotu, buď 0 (černá) nebo 255 (bílá). Postup zpracování je obdobný jako v příkladě 7.1.
1D DFT $X_m[k]$ pro každý řádek dává jedinou nenulovou spektrální složku v $k = 0$. Velikost této složky se liší od jednoho řádku k druhému, pulsuje mezi 255 (bílou) a 0 (černou).
Stejně jako v předchozím příkladě jsou DFT výsledky $X_m[k]$ pro řádky kombinovány do matice 50 x 50. První (nebo $k = 0$) sloupec obsahuje vysoké/nízké střídající se průběhy už popsané a zbytek sloupců jsou nuly, Obr. 7.6.
Pokud dokončíme 2D DFT tím, že vezmeme 1D DFT těchto sloupců, výsledkem je obálka sin-cardinalis podél souřadnice *i*, kde $k = 0$ a nuly jsou všude jinde.

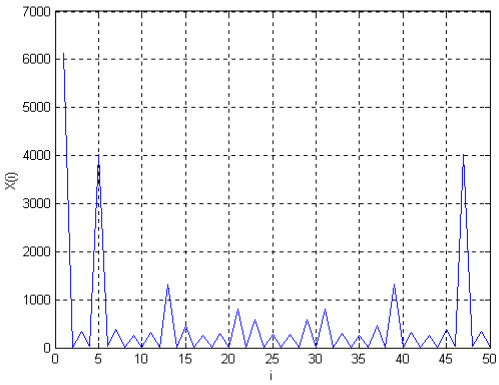
Příklad 7.3: Šachovnice



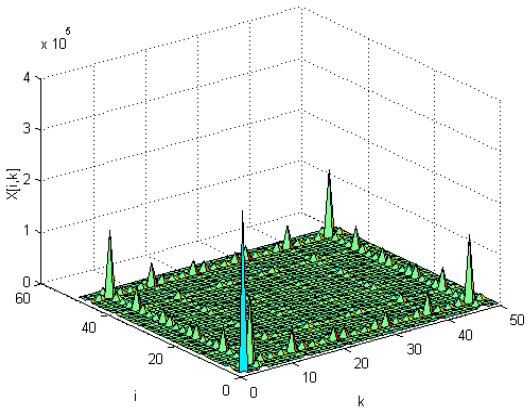
Obr. 7.7 Matice o velikosti 50 x 50 pixelů.



Amplitudové spektrum |*X[k]*|. (řádky)



Amplitudové spektrum |*X[i]*|. (sloupce)



Amplitudové spektrum |*X[i,k]*|

Obr.7.8 Vykreslení amplitudového spektra

Obálka amplitudového spektra podél libovolného řádku nebo sloupce má tvar sinc (sinus- cardinalis). Tvar sinc se ukazuje ve všech směrech a odráží obdélníkový průběh v obou (vertikální i horizontální) směrech původního obrázku (Obr. 7.7).

Podél diagonálního směru v původním obrázku dochází k silnému ztracení obdélníkové vlny, což vysvětluje menší špičky uprostřed amplitudového spektra.

Z výše uvedeného vyplývá, že :

- **amplitudové spektrum** nese informaci o barvách obrazu
- **fázové spektrum** nese informaci o umístění a okrajích objektů.

7.4.1 Vlastnosti Fourierovy transformace

Fourierova transformace – umožňuje snadno vypočítat konvoluci dvou obrázků jako součin jejich Fourierových obrazů.

Konvoluce obrazu:
Konvoluce obrazu popisuje průchod obrazu lineárním filtrem, což je základ filtrace obrazu v prostorové oblasti. Vypočteme-li konvoluci dvou obrazů f, h o stejných rozměrech M x N získáme (periodický) obraz g.

$$g(x,y)=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)h(x-m,y-n) \tag{7.10}$$

Konvoluční věta: *f, g, h* a jejich Fourierovy obrazy F, G, H jsou vázány rovnicí

$$G(u,v)=F(u,v)H(u,v) \tag{7.11}$$

která vyjadřuje součin po prvcích

$$g(x,y)=\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)H(u,v)\exp(2\pi j(\frac{xu}{M}+\frac{yv}{N})) \tag{7.12}$$

přičemž Fourierův obraz $F(u,v)$ vstupního signálu $f(m,n)$ je dán vztahem přímé Fourierovy transformace

$$F(u,v)=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)\exp\left[-2\pi j(\frac{mu}{M}+\frac{nv}{N})\right]$$

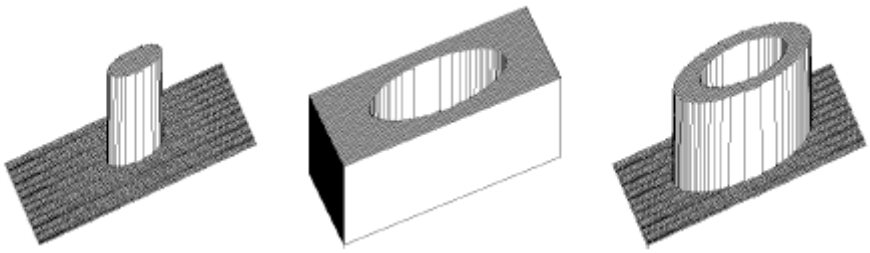
$$u=0,1,...,M-1 \qquad v=0,1,...,N-1 \tag{7.13}$$

Fourierova transformace dovoluje interpretovat metody předzpracování obrazu založené na konvoluci jako filtraci ve frekvenční oblasti. Někdy je to názornější.

Předpokládejme opět, že *f* je vstupní obraz a *F* výsledek Fourierovy transformace. **Konvoluční filtr** *h* lze vyjádřit jako jeho Fourierův obraz *H*. Matici *h* můžeme nazývat odezvou bodového zdroje (impulsní charakteristikou filtru) a matici *H* **frekvenční charakteristikou filtru**. Zda pro filtraci použijeme *h* nebo *H* je z teoretického hlediska lhostejné. V prvním případě pracujeme v prostorové

oblasti (dvojrozměrná analogie časové oblasti pro jednorozměrné průběhy) a ve druhém případě ve frekvenční.

Nyní následuje ukázka jednoduchého příkladu dvojrozměrné filtrace ve frekvenční oblasti (dvojrozměrné případy frekvenčně selektivních filtrů).



a) b) c)

Obr. 7.9 Frekvenční filtry zobrazené ve 3D prostoru: (a) Dolní propust, (b) horní propust, (c) pásmová propust.[2]

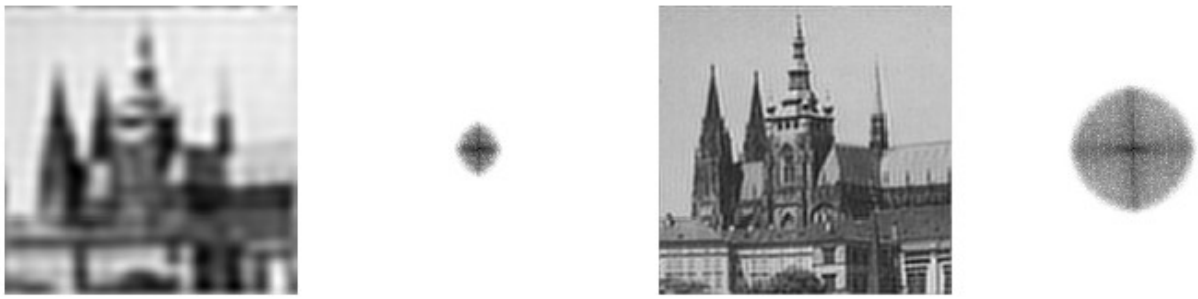
Dolní propust odpovídá frekvenční charakteristice filtru *H(u, v)* s malými hodnotami v místech hodně vzdálených od počátku (malé zesílení pro vysoké frekvence) a velkými hodnotami blízko počátku, viz Obr. 7.9 a). Filtr přenese nízké frekvence a potlačí vysoké frekvence. Chová se stejně jako filtrace průměrováním v prostorové oblasti, tedy rozmazává ostré hrany.

Horní propust odpovídá frekvenční charakteristice filtru *H(u, v)* s malým zesílením blízko počátku a velkým zesílením v oblasti vysokých frekvencí, tj. dále od počátku, viz Obr. 7.9 b).

Pásmová propust propustí jen frekvence v určitém pásmu, viz Obr. 7.9 c). Podobně jsou konstruovány filtry, které propustí jen signály o určitém směru, tj. fázi, dle terminologie Fourierovy transformace.

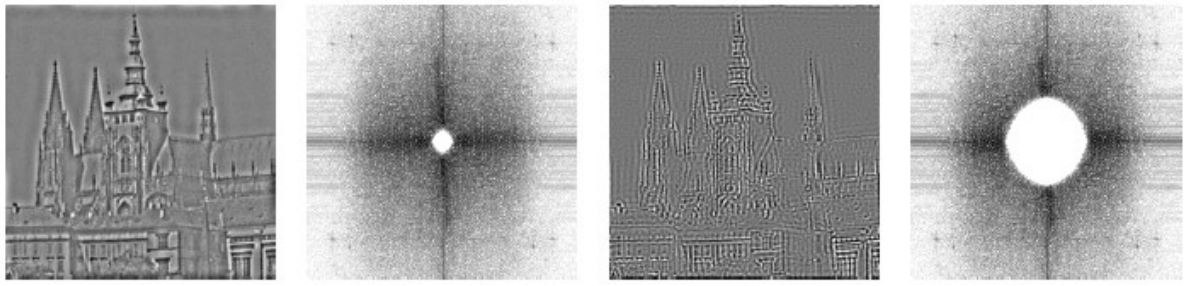
Lineární filtrace integrálními transformacemi se nejčastěji používá pro potlačení šumu, zvýraznění hran a odstranění strukturovaného šumu dobře odlišitelného od signálu ve frekvenčním spektru.
Šum obvykle mívá široké frekvenční pásmo, a proto ho potlačí, když jsou omezeny/odstraněny vysoké frekvence dolní propustí, viz Obr. 7.10. Naneštěstí spolu s šumem zmizí všechny jevy v obrázku odpovídající vysokým frekvencím (hrany, detaily jako tenké čáry, atd.). Obrázek je proto celkově rozmazán.

Hrany odpovídají místům, kde se náhle mění obrazová funkce, což přispívá k vysokým frekvencím ve spektru. Pokud vyšší frekvence relativně zesílíme, a to udělá horní propust, zvýrazní se v obrazu hrany, viz Obr. 7.11.



(a) (b) (c) (d)

Obr. 7.10 Filtrace dolní propustí, tj. odstraněny vysoké frekvence: (a) Výsledek inverzní Fourierovy transformace spektra b. (b) Spektrum obrázku s více potlačenými vysokými frekvencemi. (c) Výsledek inverzní Fourierovy transformace spektra d. (d) Spektrum obrázku s méně potlačenými vysokými frekvencemi [2]



(a) (b) (c) (d)

Obr. 7.11 Filtrací odstraněny nízké frekvence: (a) Výsledek inverzní Fourierovy transformace spektra b. (b) Spektrum obrázku s méně potlačenými nízkými frekvencemi (c) Výsledek inverzní Fourierovy transformace spektra (d) Spektrum obrázku s více potlačenými nízkými frekvencemi.[2]

Shrnutí pojmů

Fourierova transformace (FT) má v oblasti zpracování obrazu široké uplatnění. Používá se zejména v oblasti zlepšení kvality obrazu (ostření, detekce hran apod.). Použitím FT je možno získat spektrální vlastnosti obrazu pro následné aplikace různých druhů filtrů. Právě z frekvenčního a amplitudového spektra jsme schopni zjistit informace o rozložení hran a barevnosti obrazu

Konvoluce obrazu popisuje průchod obrazu lineárním filtrem, což je základ filtrace obrazu v prostorové oblasti.

Otázky

1. K čemu se ve zpracování obrazu využívá Fourierovy transformace?
2. Jaký je rozdíl mezi dolní a horní propustí filtru?

3. Proč filtrujeme obraz jak ve frekvenční tak prostorové oblasti?

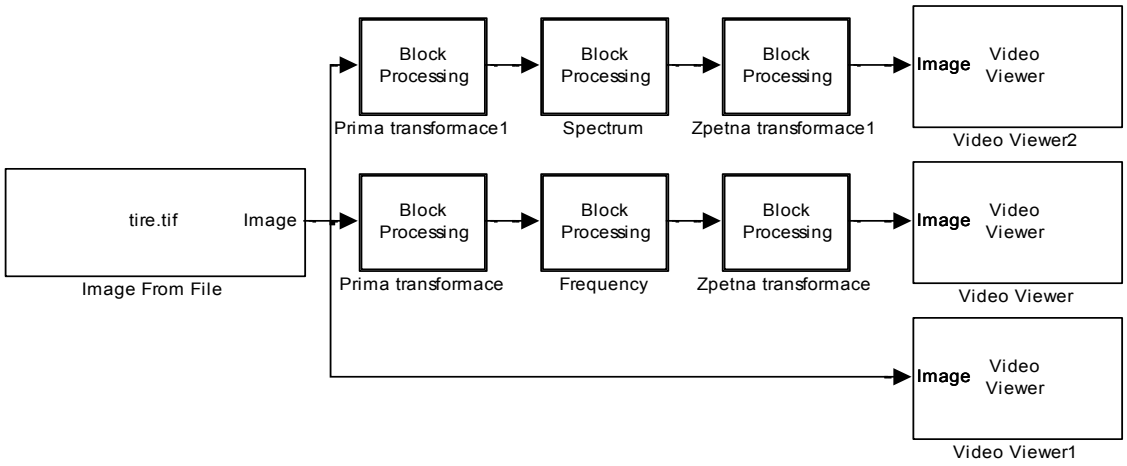
CVIČENÍ

Filtrace

V tomto cvičení za budeme zabývat praktickou ukázkou použití FT a to konkrétně tvorbou modelu pro aplikaci filtrů v prostorové a frekvenční doméně. Filtrace pomocí Fourierovy transformace se využívá zejména při odstraňování šumu a poruch v obraze.

Filtrace pomocí Fourierova spektra

Dle schématu na Obr.7.12 sestavíme v Simulinku model soustavy.



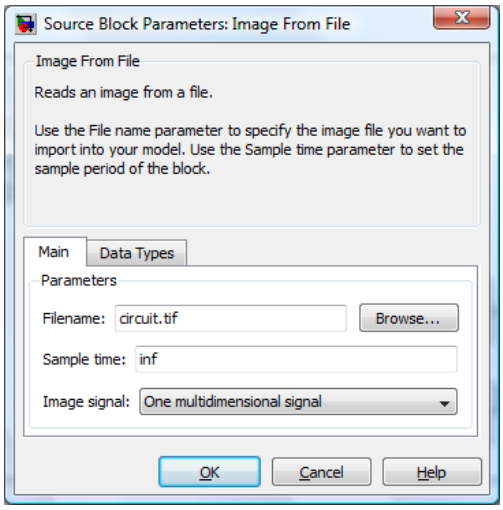
Obr. 7.12 Schéma zapojení modelu.

Jednotlivé bloky naleznete

- Blok ***Image From File*** nalezneme v menu ***Video and Image processing Blockset -> Sources***
- Blok ***Image Video Viewer*** nalezneme v menu ***Video and Image processing Blockset -> Sinks***
- Bloky ***Blok Processing*** nalezneme v menu ***Video and Image processing Blockset -> Utilities***

Nastavení vstupních parametrů

- Parametry bloku ***Image From File*** nastavíme podle obrázku (Obr.7.13). V záložce ***Main*** nastavíme cestu k obrázku pomocí prohlížeče v poli ***Filename***. Použijeme nastavení parametru ***Image signal*** na hodnotu ***One multidimesiona signal.*** Hodnotu parametru ***Sample time*** nastavíme na ***0***. V záložce ***Data Types*** provedeme nastavení parametru ***Output data type*** na hodnotu ***One multidimensional signal***.



Obr. 7.13 Nastavení bloku Image From File v záložce Main.

- Parametry bloků Video Viewer ponechte defaultně nastaveny.

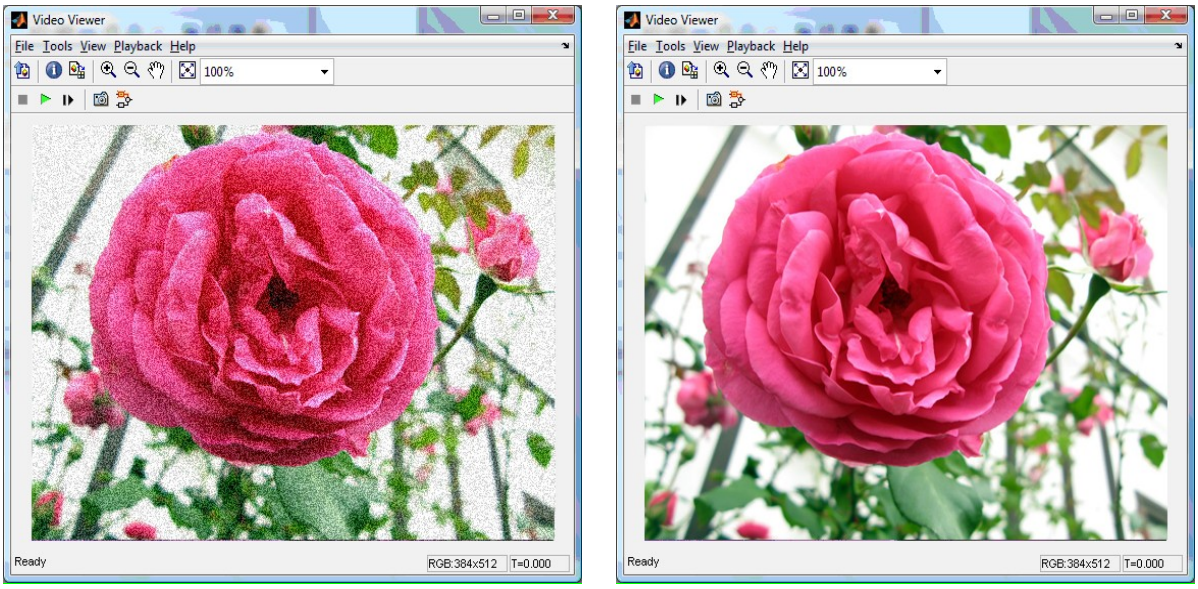
Nastavení parametrů Simulace

Jelikož pracujeme se statickým obrazem provedeme nastavení parametrů simulinku následujícím způsobem.

- Před spuštěním simulace nastavíme konfigurační parametry Simulinku. Nastavení provedeme v menu ***Simulation -> Configuration parameters -> Simulation***

- Parametr **Stop time** nastavíme na hodnotu **0**, parametr **Type** na hodnotu **Fixed step** a parametr **Solver** na hodnotu **discrete (no continuous states)**.

Nyní propojíme jednotlivé bloky dle schématu na Obr.7.12, nastavíme parametry pro simulaci a můžeme spustit simulaci.



Obr.7.19 Vlevo původní obraz, vpravo po aplikaci filtru

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 07 FT.swf.

8. BODOVÉ A ALGEBRAICKÉ OPERACE S OBRAZY

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat co je to matematická morfologie, binární obraz a základní metody bodového zpracování obrazu.
- popsat matematicky mechanismus jednotlivých metod dilatace a eroze obrazu.
- pracovat s metodami dilatace a eroze v aplikaci v Matlab/Simulink a VIP.

Výklad

8.1 Úvod

Morfologie obecně znamená nauka o tvarech. V oblasti zpracování obrazu se používá **matematická morfologie**, kdy za použití matematických nástrojů dochází k extrakci požadovaných částí obrazu pro další zpracování. Princip matematické morfologie je založen na nelineárních operacích s obrazem za použití terminologie teorie množin.

8.2 Matematická morfologie

Při zpracovávání obrazu se poměrně často setkáváme s šumem, jehož přítomnost může být zapříčiněna mnoha důvody. Pro odstranění šumu se využívá mimo jiné metod **matematické morfologie**.

- Využívá vlastností bodových množin
- Představa – obrázky lze modelovat pomocí bodových množin libovolné dimenze

V morfologických operacích pracujeme s obrazem X a strukturním elementem B. Strukturní element má význam jako maska u konvoluce, postupně jej přikládáme na jednotlivé pixely obrázku A.

Jak již bylo uvedeno, **matematickou morfologií** máme na mysli matematické nástroje pro extrakci požadovaných součástí obrazu. Tyto nástroje jsou založeny na nelineárních operacích v obrazu, přičemž obraz je chápán podle teorie množin (binární obraz-bodová množina v E2, RGB obrazo-bodová množina v E3).

Binární obraz

- Obrazová funkce v každém bodě nabývá jedné ze dvou možných hodnot.
- Je zpravidla výsledkem metod provádějících segmentaci obrazu (objekt = 1, pozadí = 0)
- Před tím než jsou analyzovány, mohou být zpracovány jedním z následujících postupů
- Objekt v obraze je popsán jako množina 2-D bodů s hodnotou =1.

Tyto nástroje jsou využívány při předzpracování obrazu a finálních úpravách obrazu, či k detekci hran atd. V podstatě se jedná o **morfologickou transformací *Y***, která je relací mezi **obrazem *X*** a **strukturním elementem *B***. Strukturní element B bývá zpravidla menší než obraz X. Na Obrázek 10 jsou uvedeny nejčastěji používané strukturní elementy používané v morfologických transformacích.

Základní operace matematické morfologie:

- Dilatace
- Eroze
- Otevření
- Uzavření
- Transformace… atd.

8.2.1 Dilatace

Dilatace zvětšuje objekty v obraze a používá se často pro zaplňování děr a zálivů. Ve své podstatě dilatace skládá body dvou množin pomocí vektorového součtu . Dilatace $X \oplus B$ je bodovou množinou všech možných vektorových součtů pro dvojice pixelů, vždy pro jeden z množiny *X* a jeden z množiny *B*.[2]

Definice:

$$X \oplus B = \left\{ p \in \varepsilon^2, p = x + b, x \in X, b \in B \right\} \tag{8.1}$$

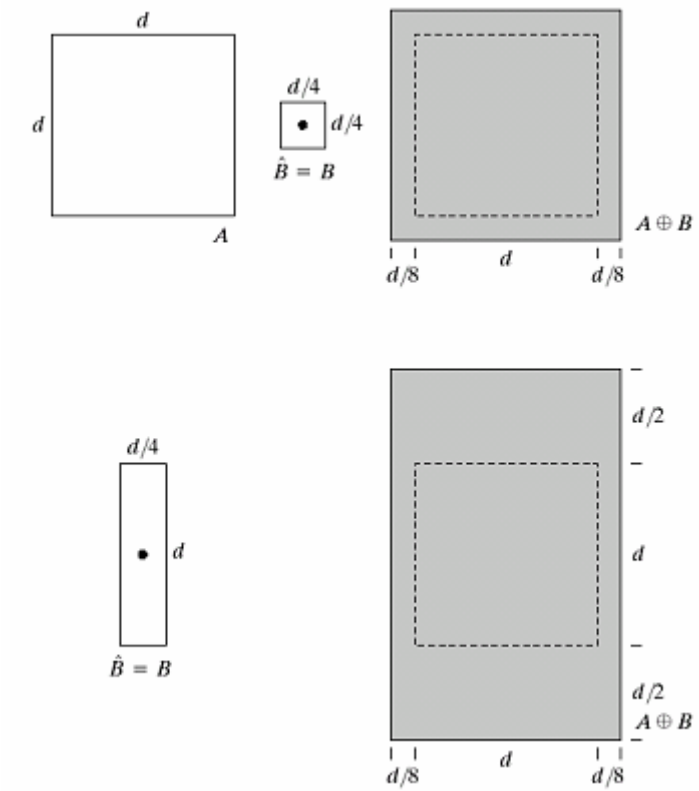
Dilatace má následující matematické vlastnosti:

- je komutativní $X \oplus B = B \oplus X$
- je asociativní $X \oplus (B \oplus C) = (X \oplus B) \oplus C$
- je rostoucí transformací $If \quad X \subseteq Y, \quad than \quad X \oplus B \subseteq Y \oplus B$

Rovnici dilatace je také možné zapsat v tvaru:

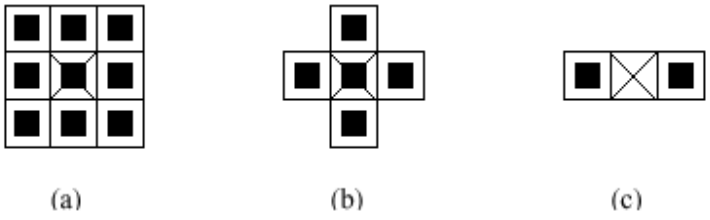
$$X \oplus B = \left\{ z \middle| \left(\hat{B} \right)_z \cap X \neq 0 \right\}$$

Kde je tato rovnice založena na získání zrcadlového obrazu *B* z jeho originálu a posunutím tohoto zrcadlového obrazu do *z*. Dilatace *X* a *B* je potom množina všech posunutí *z* takových, že $\hat{B}$ a *X* se překrývají nejméně v jednom elementu. *B* je označován jako **strukturní element**. Princip dilatace je zobrazen na Obr. 8.1.



Obr. 8.1 Princip dilatace

Při dilataci dochází k přičtení obrazového elementu *B* k původnímu obrazu *A*. Na obrázku Obr. 8.1 je znázorněn princip dilatace pro dva různé obrazové elementy. V horní části vlevo je původní obraz *A*. Vpravo je původní obraz po překrytí všech jeho bodů obrazovým elementem *B*. Stejný případ je ve spodní části obrázku, pouze obrazový element *B* je v jiném tvaru. Z výsledného obrazu je zřejmé, že výstup dilatace závisí na tvaru obrazového elementu. Strukturální elementy využívané pro dilataci bývají matice 2x2 nebo 3x3 prvky. Příklad matice 3x3 prvky je zobrazen na Obr. 8.2.

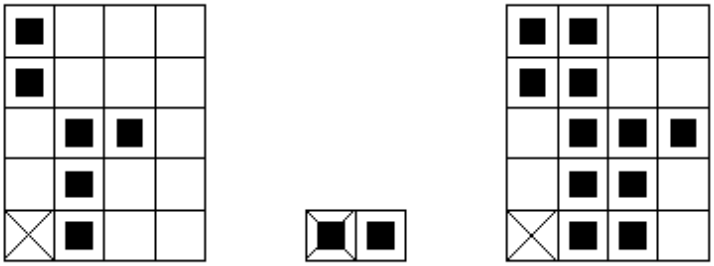


Obr. 8.2 Základní strukturální elementy dilatace.

Příklad dilatace pro vstupní obraz *A* a matici strukturálního elementu *B* je uveden na popsaný v následujícím příkladu na Obr.8.3. Křížek v prázdném poli představuje počátek souřadného systému obrazu. Křížek v poli elementu taktéž představuje počáteční bod. Obrazový element je postupně přikládán ke každé souřadnici. Ke změně dochází pouze v místech kde je v původním obraze hodnota=1. Tedy ke změně dochází pouze v místě černého bodu, ke kterému je přičten obrazový element.

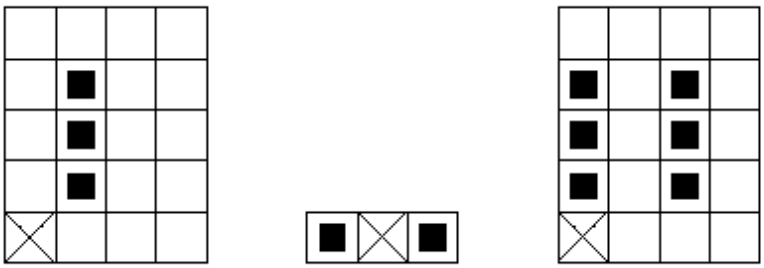
Příklad 8.1: Proveďte dilataci obrazu A elementem B.

$$A = \{(1,0),(1,1),(1,2),(2,2),(0,3),(0,4)\}$$
$$B = \{(0,0),(1,0)\}$$
$$A \oplus B = \{(1,0),(1,1),(1,2),(2,2),(0,3),(0,4),(2,0),(2,1),(2,2),(3,2),(1,3),(1,4)\}$$



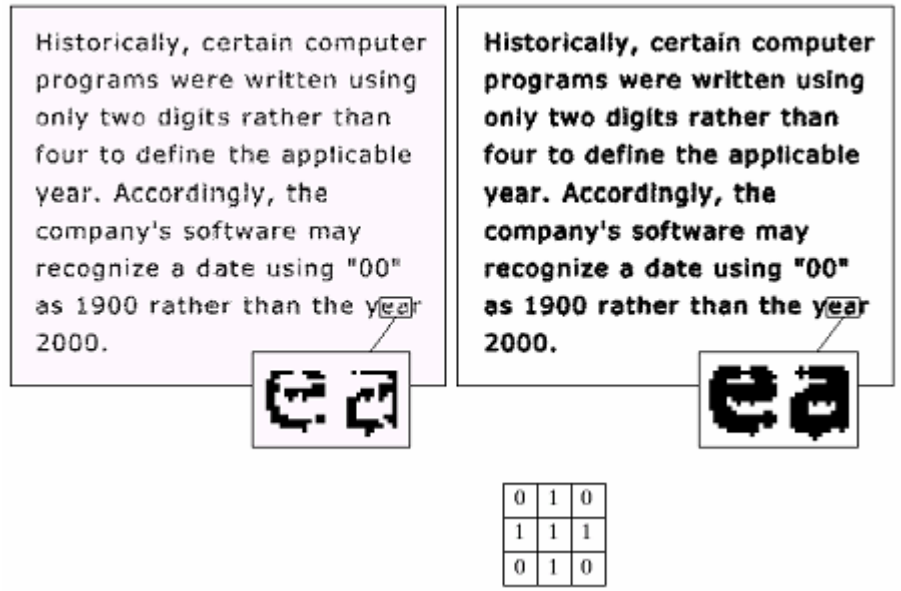
Obr. 8.3 Obrazový příklad dilatace.

Příklad dilatace pro případ, kdy počátek není prvkem strukturního elementu je zobrazen na Obr. 8.4. Princip je stejný jako u Obr. 8.3. Výsledek se podstatně liší od vstupního obrazu. Byla zde porušena souvislost.



Obr. 8.4 Dilatace pro případ, kdy počátek není prvkem strukturního elementu.

Na Obr. 8.5 je ukázka aplikace dilatace s maticí elementu *B* o velikosti 3x3. Vlevo originální obraz, vpravo zvýraznění textu pomocí dilatace. Vlivem dilatace došlo k rozšíření o jednu vrstvu a současně došlo k zaplnění děr o velikosti 1 pixelu. Písmo v obrázku je tak čitelnější.



Obr. 8.5 Příklad dilatace textového obrázku.

Využití dilatace:

- Samostatně k zaplnění malých mezer, úzkých zálivů, základ složitějších operací.
- Zvětšuje objekt
- Má-li se zachovat původní rozměr objektů kombinuje se dilatace s erozí.

8.2.2 Eroze

Eroze je duální operace k dilataci. Při erozi dochází k odstranění slupky v obraze a tudíž k jeho zmenšení. Eroze se využívá k odstranění drobných nerovností a vyhlazení obrazu.

Definice:

$$X \Theta B = \left\{ p \in \varepsilon^2, p + b \in X, \forall b \in B \right\} \tag{8.2}$$

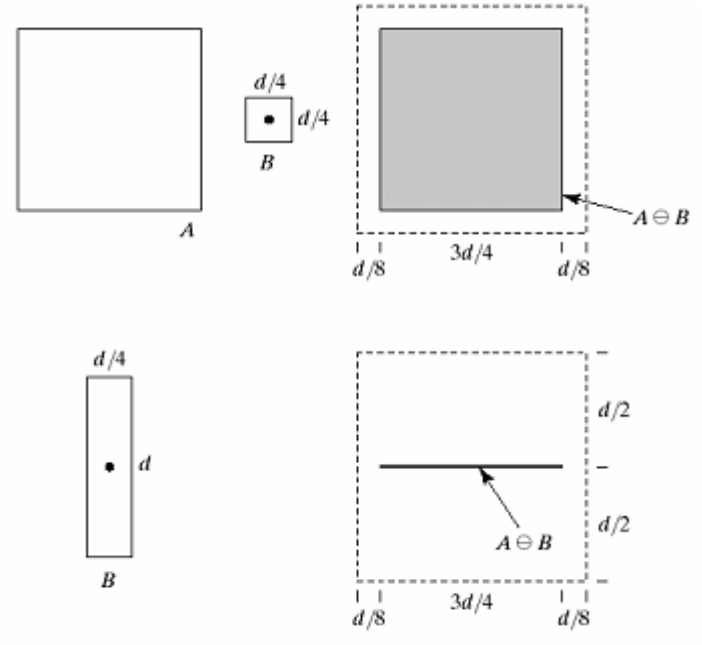
Kde p je bod v obraze, ε^2 představuje binární obrazový prostor. X představuje původní obraz a B je strukturálním elementem. Eroze skládá dvě bodové množiny s využitím rozdílů vektoru. Jak již bylo uvedeno,eroze je duální transformací k dilataci, ale není její inverzní transformací.

Erozi obrazu je možné zapsat také ve tvaru:

$$A \Theta B = \left\{ z \middle| (B)_z \subseteq A, \forall s \in S \right\} \tag{8.3}$$

Tato rovnice určuje, že eroze A a B je množina všech bodů z takových, že B posunuté do z náleží do A. Nejčastěji používaným strukturním elementem je stejně jako u dilatace element 3x3. Po aplikaci eroze na obraz zmizí objekty (čáry) tloušťky 2 a osamělé body. Objekty se zmenší o 1 slupku. Eroze se dá využít k získání obrysu objektu tak, že provedeme odečtení eroze obrazu od původního obrazu.

Eroze využívá stejných strukturálních elementů jako dilatace. Jejich aplikace je však odlišná. Princip eroze je zobrazen na Obr. 8.6.

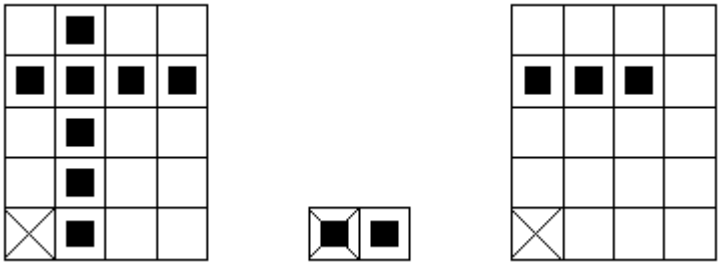


Obr. 8.6 Princip Eroze obrazu.

V horní části obrázku Obr. 8.6 je zobrazen původní obraz *A*. Uprostřed je tvar strukturálního elementu *B*. Eroze původního obrázku je znázorněna šedou barvou. Čárkovaně jsou obrysy původního obrazu. Případ jiného tvaru strukturálního elementu *B* je zobrazen ve spodní části obrázku. Po aplikaci na původní obraz je výstupem čára. Také u eroze záleží na tvaru a velikosti strukturálního elementu. Z původního obrazu je zachována pouze ta část, která se plně překrývá se elementem *B*. Pro každý bod obrazu *p* se ověřuje, zda pro všechna možná *p* + s leží výsledek v *B*. pokud ano zapíše se v reprezentativním bodě do výsledného obrázku 1 v opačném případě se zapíše 0.

Příklad 8.2: Proveďte erozi obrazu A elementem B.

$$A = \{(1,0),(1,1),(1,2),(0,3),(1,3),(2,3),(3,3),(1,4)\}$$
$$B = \{(0,0),(1,0)\}$$
$$A \Theta B = \{(0,3),(1,3),(2,3)\}$$



Obr. 8.7 Příklad eroze obrazu.

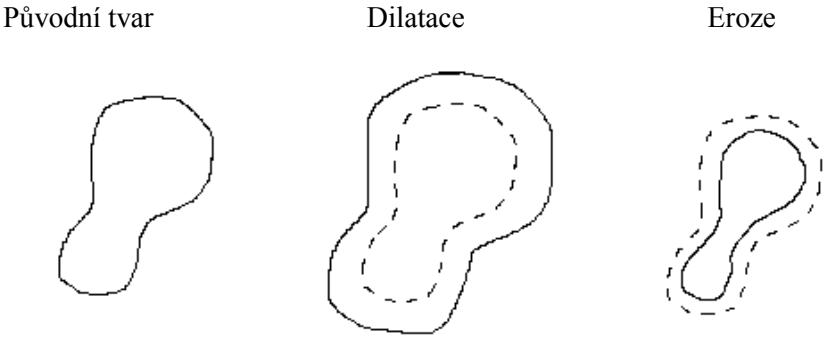
Na Obr. 8.7 je zobrazen příklad eroze obrazu. Bodová množina A je erodovaná strukturním elementem B, který není všesměrový. Svislá čára zmizí.

Využití eroze:

- Zjednodušení struktury objektů – objekty tloušťky 1 se ztratí, a tak se složitější objekt rozdělí na několik jednodušších.
- Eliminace irelevantních detailů

8.2.3 Dilatace versus Eroze

Použitím dilatace a eroze dochází k matematickým operacím s obrazem. Užití dilatace a eroze je porovnáno na Obr. 8.8 Je důležité mít na paměti, že eroze je duální transformací k dilataci ale nejedná se o operaci inversní. Původní tvar je u aplikace dilatace a eroze v obraze znázorněn čárkovaně.



Obr. 8.8 Základní rozdíl mezi dilatací a erozí obrazu.

Příklad užití eroze a následné dilatace obrazu je na Obr. 8.9. Předpokládejme, že chceme eliminovat všechny čtverce v obrázku kromě největších z nich. Velikosti čtverců v obraze jsou 1, 3, 5, 7 ,9 ,15. K efektivní eliminaci použijeme strukturní element o velikosti 13 x 13 pixelů a matematickou erozi obrazu. Jak je vidět z prostředního obrázku Obr. 8.9 v obraze zůstaly jen zbytky těch největších čtverců. K obnovení zbylých tří čtverců na jejich původní velikost 15 x 15 využijeme dilatace se stejným strukturním elementem, který jsme použili pro erozi. Výsledkem je obraz, na posledním obrázku Obr. 8.9, ze kterého byly odstraněny všechny prvky menší než strukturální element.

Obr. 8.9 Obrázek se čtverci o velikosti 1, 3, 5, 7, 9, 15 pixelů. Ukázka eroze a následné dilatace se stejným strukturním elementem o velikosti 13 pixelů.

8.2.4 Otevření a uzavření

Obě tyto operace jsou kombinací dilatace a eroze. Na rozdíl od eroze a dilatace je otevření a uzavření invariantní vzhledem k posunu strukturního elementu.

Otevření

Definice: $$A \circ B = (A \Theta B) \oplus B \tag{8.4}$$

Jedná se o erozi následovanou dilatací. Otevření obecně vyhlazuje kontury objektu, ruší úžiny a eliminuje úzké výčnělky.

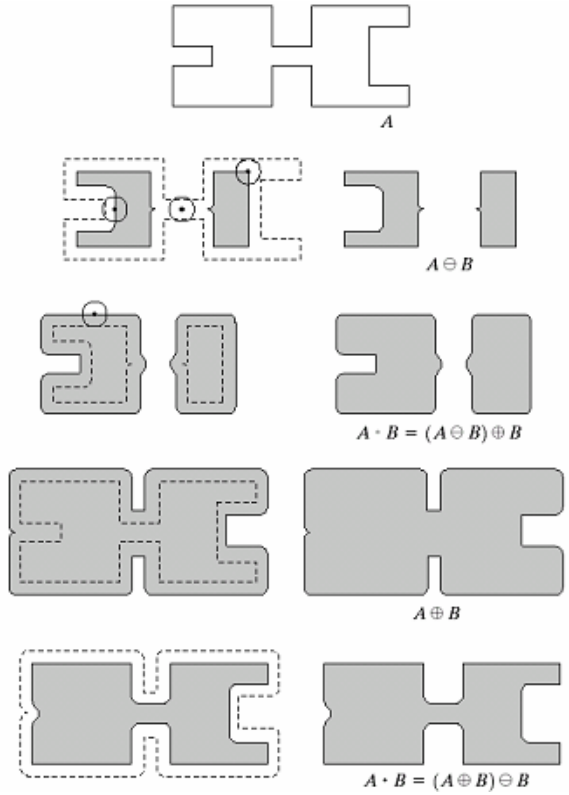
Uzavření

Definice: $$A \bullet B = (A \oplus B) \Theta B \tag{8.5}$$

Jde o dilataci následovanou erozí. Uzavření také vyhlazuje kontury objektu, ale na rozdíl od otevření spojuje úzké mezery a dlouhé úzké zálivy, eliminuje malé díry a zaplňuje mezery v konturách.

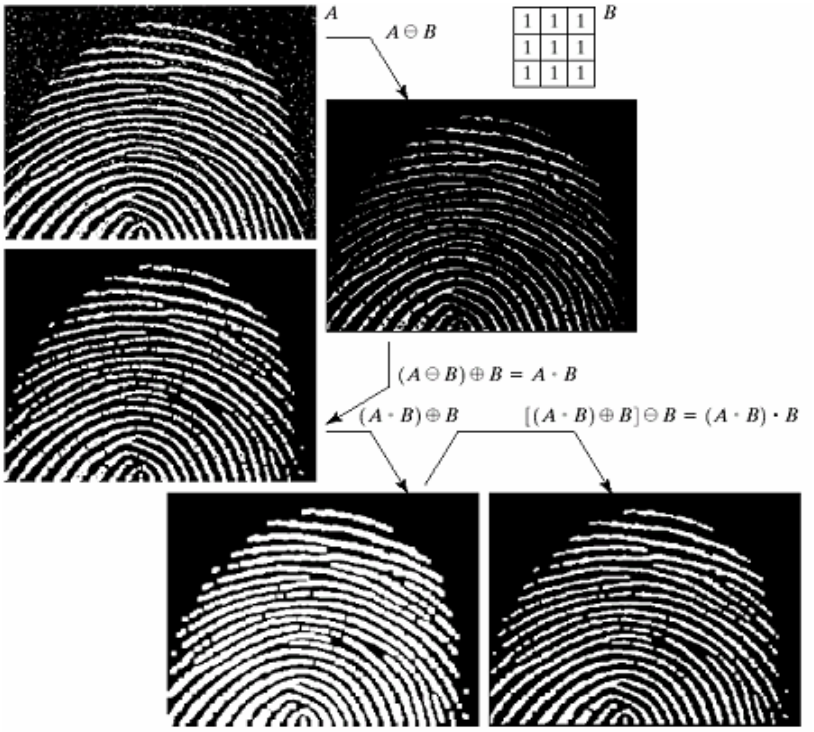
Pokud se obraz nezmění po otevření resp. uzavření strukturním elementem B, pak říkáme, že je obraz otevřený resp. uzavřený vzhledem ke strukturnímu elementu B.

Otevření oddělí objekty spojené úzkou šíjí a odstraní malé detaily. Uzavření spojí objekty, které jsou blízko u sebe a zaplní malé díry a úzké zálivy, přičemž význam pojmů „*malý*“, „*úzký*“, „*blízký*“ závisí na velikosti strukturního elementu. Princip obou operací je zobrazen na Obr. 8.10.



Obr. 8.10 Morfologické otevření a uzavření. Strukturním elementem je malý kruh.

Morfologické operace můžeme použít pro odstranění šumu z obrázku, viz. Obr. 8.9. Obrázek ukazuje otisk prstu narušený šumem. Šum se projevuje jako malé bílé body v obrázku. Nejprve provedeme erozi obrázku se strukturním elementem *B*. V druhém kroku je šum zcela potlačen, ale došlo k zeslabení kontur. Zesílení kontur provedeme pomocí operace otevření. Nicméně s využitím této operace vznikly nové mezery mezi hranami, které odstraníme pomocí dilatace. Dilatací byla většina poruch eliminována, ale došlo k zesílení kontur, proto použijeme operaci uzavření.



Obr. 8.11 Odstranění šumu pomocí morfologických operací.

Využití otevření a uzavření:

- používá se pro odstranění detailů v obraze, které jsou menší než strukturní element
- celkový tvar objektu se neporuší
- **otevření** – oddělí objekty spojené úzkou šíjí a tak zjednoduší strukturu objektů
- **uzavření** – spojí objekty, které jsou blízko sebe, zaplní malé díry a vyhladí obrys tím, že zaplní úzké zálivy.

Shrnutí pojmů

Matematická morfologie využívá vlastností bodových množin. V morfologických operacích pracujeme s obrazem A a strukturním elementem B. Strukturní element má význam jako konvoluční masky kde jej postupně přikládáme na jednotlivé pixely obrázku A.

Základními morfologickými operacemi jsou **Dilatace** a **Eroze**.

Dilatace je dána vzorcem $A \oplus B = \left\{ z \middle| \left(\hat{B}\right)_z \cap A \neq 0 \right\}$ Tato rovnice je založena na získání zrcadlového obrazu B z jeho originálu a posunutím tohoto zrcadlového obrazu do z. Dilatace A a B je potom množina všech posunutí z takových, že $\hat{B}$ a A se překrývají nejméně v jednom elementu. B je označován jako **strukturní element**.

Eroze je dána vzorcem: $A\Theta B = \left\{z \middle| (B)_z \subseteq A, \forall s \in S\right\}$ Tato rovnice určuje, že eroze A a B je množina všech bodů z takových, že B posunuté do z náleží do A.

Otevření je eroze následovaná dilatací.

Uzavření je dilatace následovaná erozí.

Otázky

1. Jaké jsou základní morfologické operace při práci s obrazem?
2. Jaký je rozdíl mezi Dilatací a Erozí obrazu?
3. K čemu slouží dilatace a eroze obrazu?

CVIČENÍ

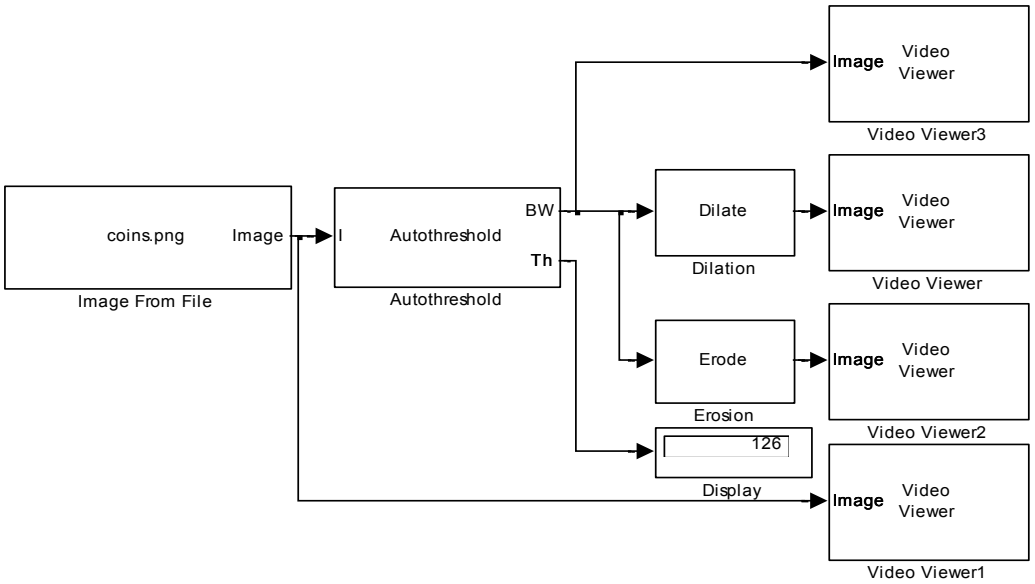
Morfologické operace

Základními morfologickými operace s obrazem jsou dilatace a eroze. Binární dilatace se používá k zaplnění malých děr v objektech. Při aplikaci této metody dochází k narůstání oblastí a tedy i doplnění drobných nerovností. Binární eroze je duální morfologickou transformací k dilataci. Užitím nejdříve dilatace a následně eroze nedojde ke změně velikosti objektu, ale drobné díry budou zaplněny.

Vlastnosti těchto dvou funkcí a jejich aplikaci si ukážeme v průběhu tohoto cvičení. Následně si ukážeme také možnost pozdější aplikace při počítání jednotlivých objektů. Využijeme k tomu Opening a Close funkce, které jsou kombinací dilatace a eroze v opačném pořadí.

Dilatace a Eroze

V Simulinku nakreslete model dle schématu na Obr. 8.12. Jako vstupní obrázek použijte coins.png ze základní obrazové sady Matlabu.



Obr. 8.12 Schéma zapojení modelu.

Jednotlivé bloky naleznete

- Blok ***Dilate*** a ***Erode*** nalezneme v menu ***Video and Image processing Blockset*** -> ***Morphological Operations***

Nastavení vstupních parametrů

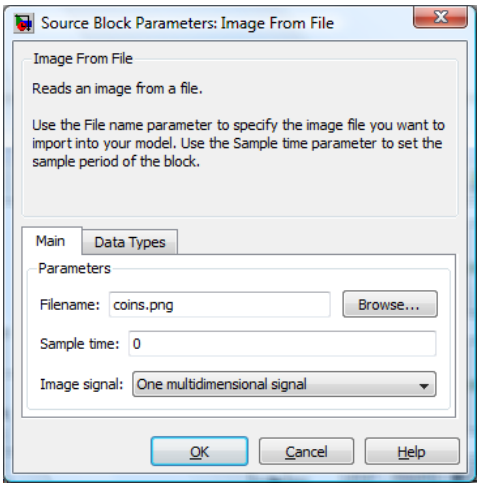
- Parametry bloku ***Image from File*** nastavíme dle Obr. 4. Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**, protože budeme pracovat se statickým obrazem. Parametr **Image signal** ponecháme v nastavení **One muldimensional signal**.

- Parametry bloku ***Autothreshold*** nastavíme dle Obr. 8.14. V záložce main nastavíme parametr **Threshold operator** na **>** a zaškrtneme checkbox **Output threshold**.

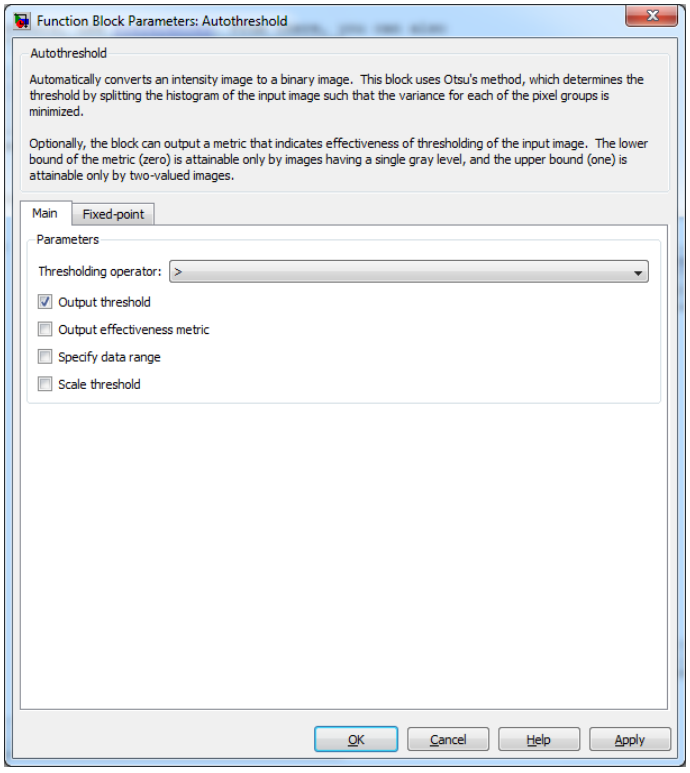
- V bloku Dilate nastavíme parametr Input type na hodnotu binary, protože pracujeme s binárním obrázkem.

- V bloku Erode nastavíme parametr Input type na hodnotu binary, protože pracujeme s binárním obrázkem.

- Parametry bloků ***Video Viewer*** ponecháme defaultně přednastaveny.



Obr.8.13 Nastavení parametrů bloku Image From File.

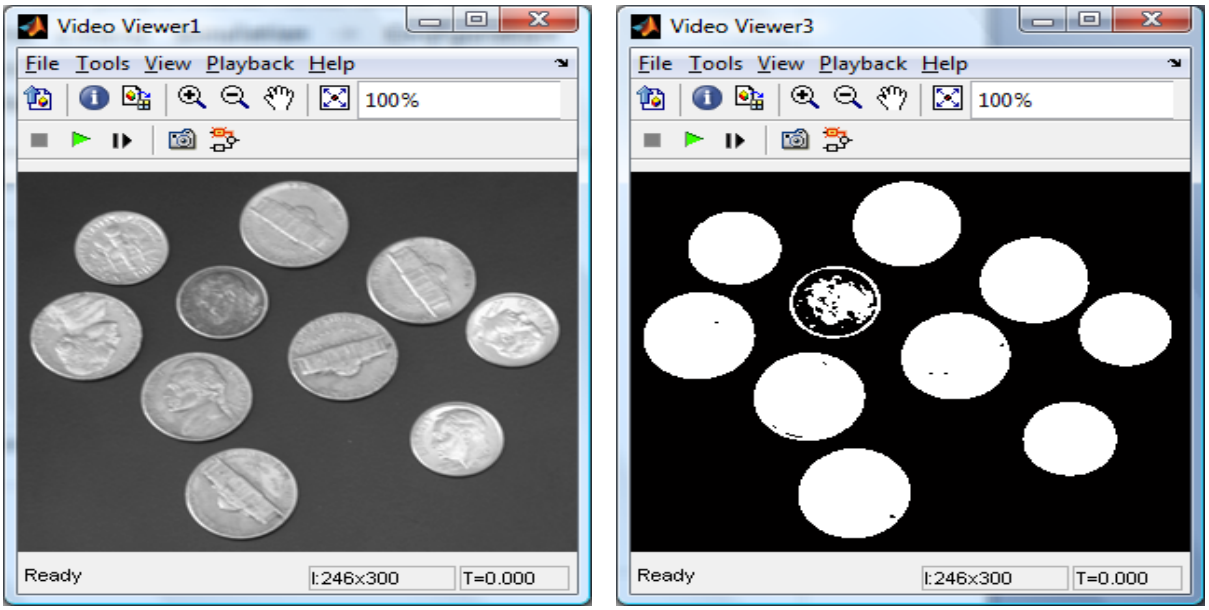


Obr.8.14 Nastavení parametrů bloku Autothreshold.

Nastavení parametrů Simulace

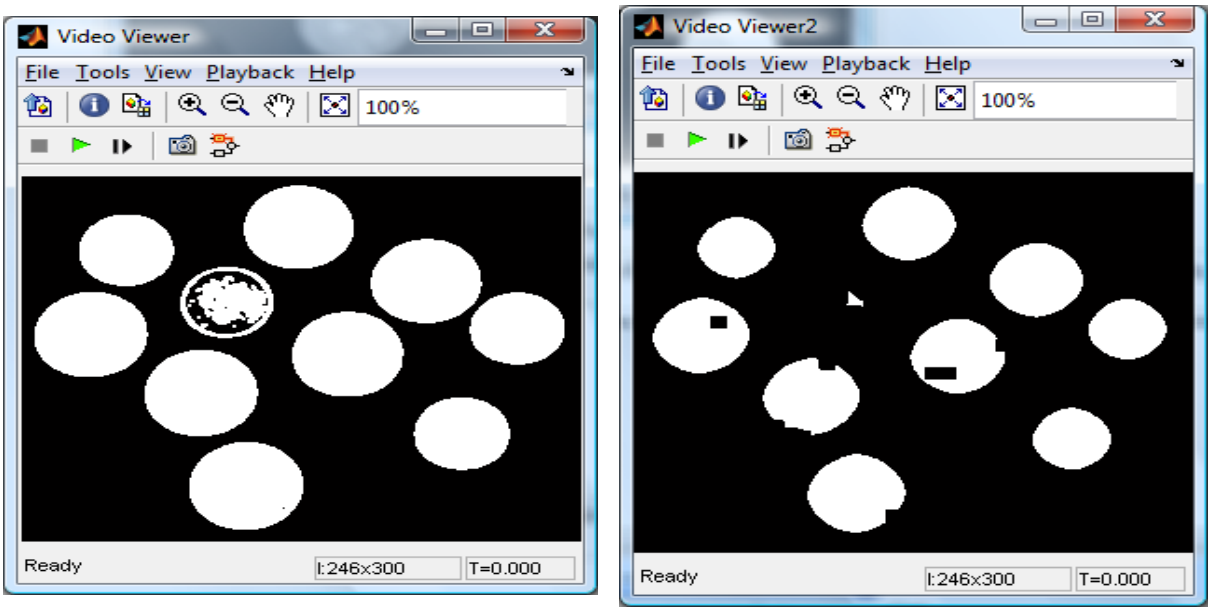
Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

Nyní propojte jednotlivé bloky dle schématu na Obr. 8.12, nastavte parametry a můžete spustit simulaci.



Obr. 8.135 Vlevo původní obrázek, vpravo po aplikaci prahování.

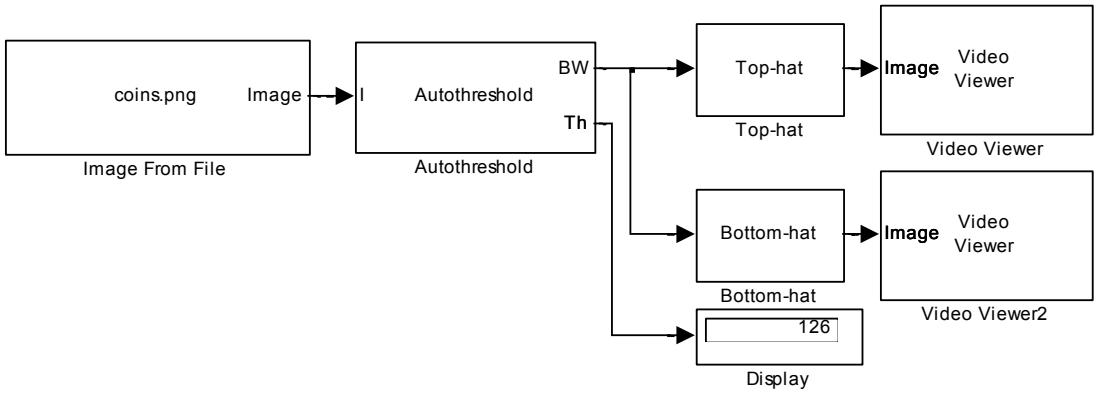
Po aplikaci prahování pomocí Autothreshold bloku jsou v obrázku patrné nerovnosti. K odstranění těchto nerovností jsou použity funkce dilatace a eroze.



Obr. 8.146 Vlevo užití bloku Dilate, vpravo užití bloku Erode.

Top a Bottom hat

V Simulinku nakreslete model dle schématu na Obr. 8.167. Jako vstupní obrázek použijte coins.png ze základní obrazové sady Matlabu.



Obr. 8.15 Schéma zapojení modelu.

Jednotlivé bloky naleznete

- Blok ***Top-hat*** a ***Bottom-hat*** nalezneme v menu ***Video and Image processing Blockset*** -> ***Morphological Operations***

Nastavení vstupních parametrů

- Parametry bloku ***Image from File*** nastavíme dle Obr. 4.. Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**. Parametr **Image signal** ponecháme v nastavení **One muldimensional signal**.

- Parametry bloku ***Autothreshold*** nastavíme dle Obr. 4.14. V záložce Main nastavíme parametr **Threshold operator** na **>** a zaškrtneme checkbox **Output threshold**.

- V bloku ***Top-hat*** nastavíme parametr **Input type** na hodnotu **binary**, protože pracujeme s binárním obrázkem. Parametr **Neighborhood or structuring element** nastavíme na hodnotu **strel('square',35);**

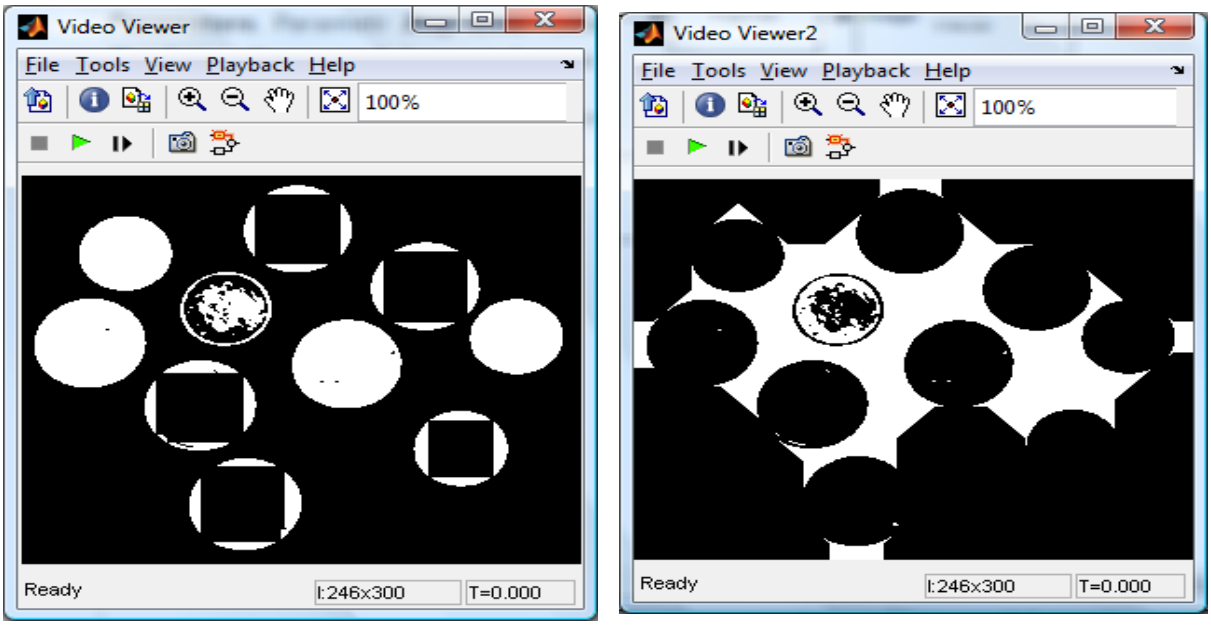
- V bloku ***Bottom-hat*** nastavíme parametr **Input type** na hodnotu **binary**, protože pracujeme s binárním obrázkem. Parametr **Neighborhood or structuring element** nastavíme na hodnotu **strel('octagon',15) ;**

- Parametry bloků ***Video Viewer*** ponecháme defaultně přednastaveny.

Nastavení parametrů Simulace

Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

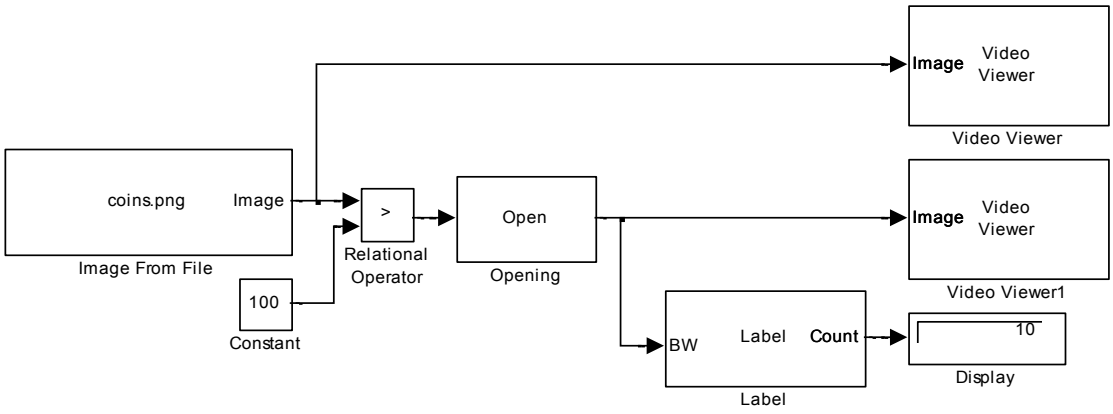
Nyní propojte jednotlivé bloky dle schématu na Obr. 8.157, nastavte parametry a můžete spustit simulaci.



Obr. 8.168 Vlevo po průchodu blokem Top-hat, vpravo po aplikaci prahování.

Počítání objektů

Dle schématu na Obr.8.19 sestavíme v Simulinku model soustavy.



Obr.8.19 Schéma zapojení modelu.

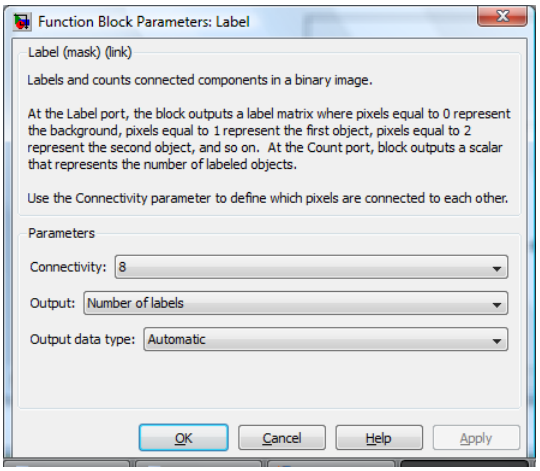
Jednotlivé bloky naleznete

- Blok ***Image From File*** nalezneme v menu ***Video and Image Processing Blockset => Sources***
- Blok ***Opening*** nalezneme v menu ***Video and Image Processing Blockset => Morphological Operations***
- Blok ***Label*** nalezneme v menu ***Video and Image Processing Blockset => Morphological Operations***

- Blok ***Video Viewer*** nalezneme v menu ***Video and Image Processing Blockset => Sinks***
- Blok ***Constant*** nalezneme v menu ***Simulink => Sources***
- Blok ***Relational Operator*** nalezneme v menu ***Simulink => Logic and Bit Operations***
- Blok ***Display*** nalezneme v menu ***Signal Processing Blockset => Signal Processing Sinks***

Nastavení vstupních parametrů

- Parametry bloku ***Image From File*** nastavíme dle obrázku Obr.8.13. V záložce Main nastavíme parametr **Filename** na hodnotu **coins.png**, parametr **Sample Time** na hodnotu **0** a parametr **Image signal** ponecháme přednastaven na hodnotě **One multidimensional signal**.
- Parametr **Constant Value** bloku ***Constant*** nastavíme na hodnotu **200**. Tento blok definuje úroveň Thresholdu pro blok Relational Operator.
- Parametr **Relational Operator** bloku ***Relational Operator*** nastavíme na hodnotu **>**.
- Parametry bloku ***Opening*** ponecháme defaultně přednastaveny.
- Parametry bloku ***Label*** nastavíme dle obrázku (Obr.8.20). Parametr **Output** nastavíme na hodnotu **Number of labels**. Ostatní parametry ponecháme defaultně přednastaveny.

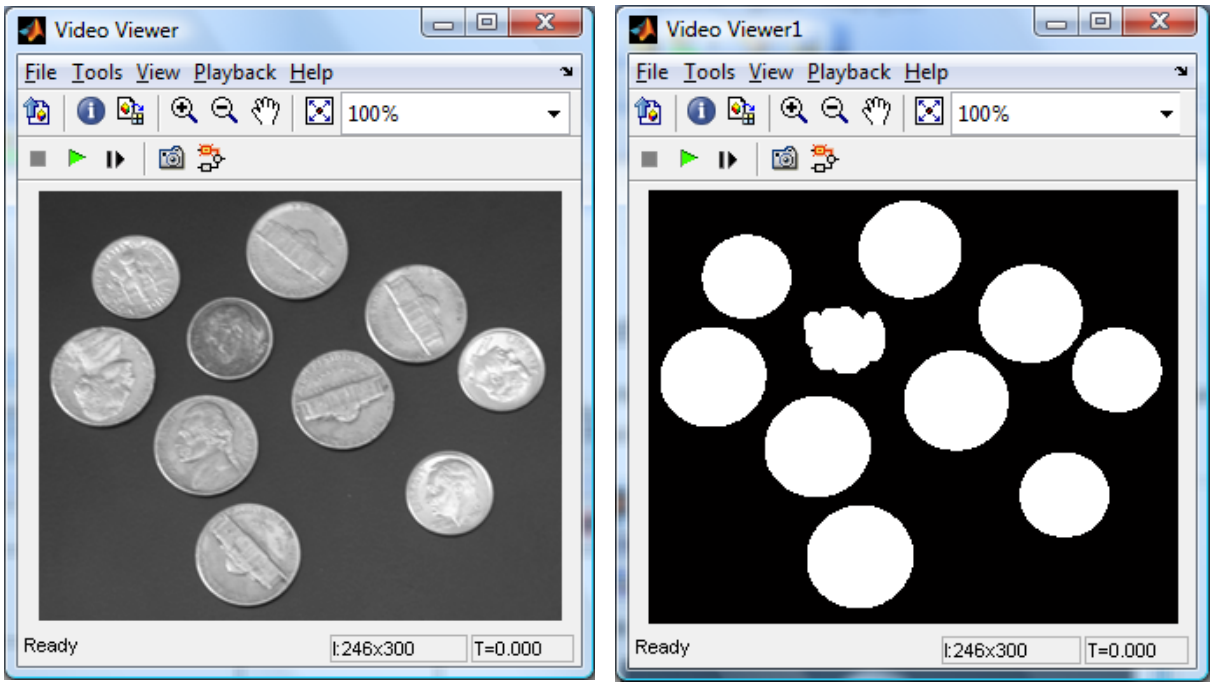


Obr.8.20 Nastavení parametrů bloku Label.

- Parametry bloku ***Display*** ponecháme defaultně přednastaveny.
- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nastavení parametrů Simulace

Nyní propojíme jednotlivé bloky dle schématu na Obr.8.19, nastavíme parametry pro simulaci a můžeme spustit simulaci.

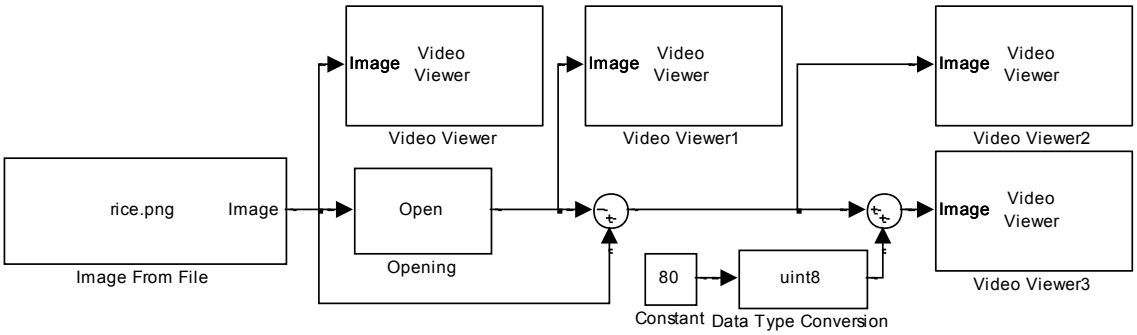


Obr.8.21 Vlevo původní obraz, vpravo po aplikaci morfologické metody label.

Korekce nerovnoměrného osvětlení

Pokaždé nemusí být zpracovávaný snímek dokonale osvětlen. Vlivem této nedokonalosti jsou některé objekty v obraze hůře detekovatelné. Proto je zapotřebí, je-li to možné, nehomogenitu takového osvětlení eliminovat. Na příkladu si ukážeme, jak se z obrazu tato nehomogenita odstraňuje.

V Simulinku nakreslete model dle následujícího schématu (Obr.8.22). Jako vstupní obrázek použijte rice.png ze základní obrazové sady Matlabu.



Obr.8.22 Schéma zapojení modelu.

Jednotlivé bloky naleznete

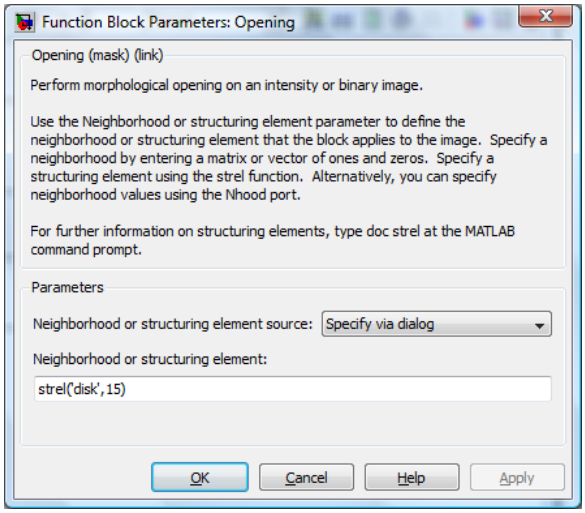
- Blok ***Opening*** nalezneme v menu ***Video and Image processing Blockset -> Morphological Operations***
- Blok ***Sum*** nalezneme v menu ***Simulink -> Math Operations***

- Blok ***Data Type Conversion*** nalezneme v menu ***Simulink -> Signal Attributes***

Nastavení vstupních parametrů

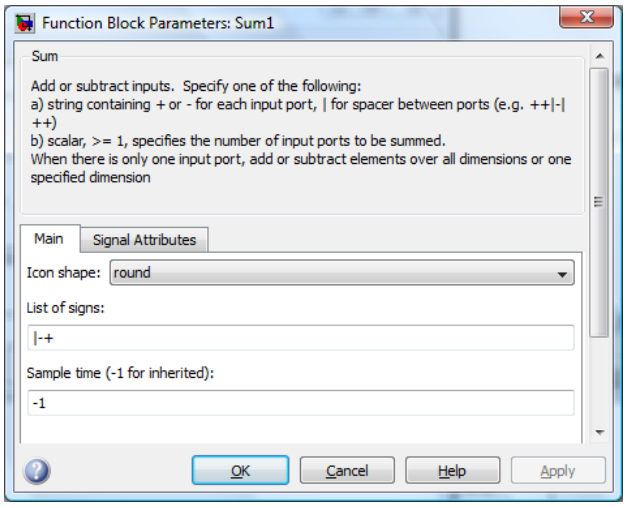
- Parametry bloku ***Image from File*** nastavíme dle následujícího postupu. Zadáme adresu vstupního obrázku pomocí parametru Filename. Parametr **Sample time** nastavíme na hodnotu **0**. Parametr **Image signal** ponecháme v nastavení **One muldimensional signal**.

- Parametry bloku ***Opening*** nastavíme dle obrázku (Obr.8.23). Parametr **Neighborhood or structuring element** source ponecháme defaultně přednastaven. Parametr **Neightborhood of structures element** nastavíme na **hodnotu strel('disk',15).**



Obr.8.23 Nastavení parametrů bloku Opening.

- Parametry bloku ***Sum*** nastavíme dle obrázku (Obr.8.24). Parametr **Icon shape** ponecháme na hodnotě **round**, parametr **List of sings** nastavíme na hodnotu **-+**. Parametry druhého bloku ***Sum*** ponecháme defaultně přednastaveny.



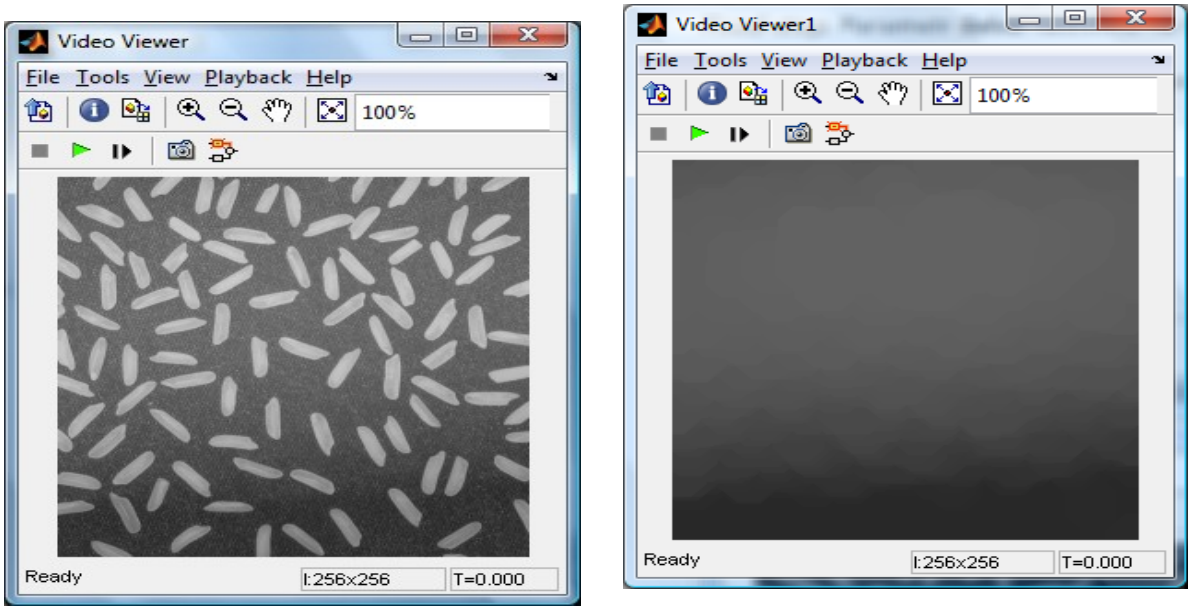
Obr.8.24 Nastavení parametrů bloku Sum.

- V bloku ***Data Type Conversion*** nastavíme parametr **Output data type** na hodnotu **unit8**. Ostatní parametry necháme defaultně přednastaveny.
- V bloku ***Constant*** nastavíme parametr **Constant value** na hodnotu **80**.
- Parametry bloků ***Video Viewer*** ponecháme defaultně přednastaveny.

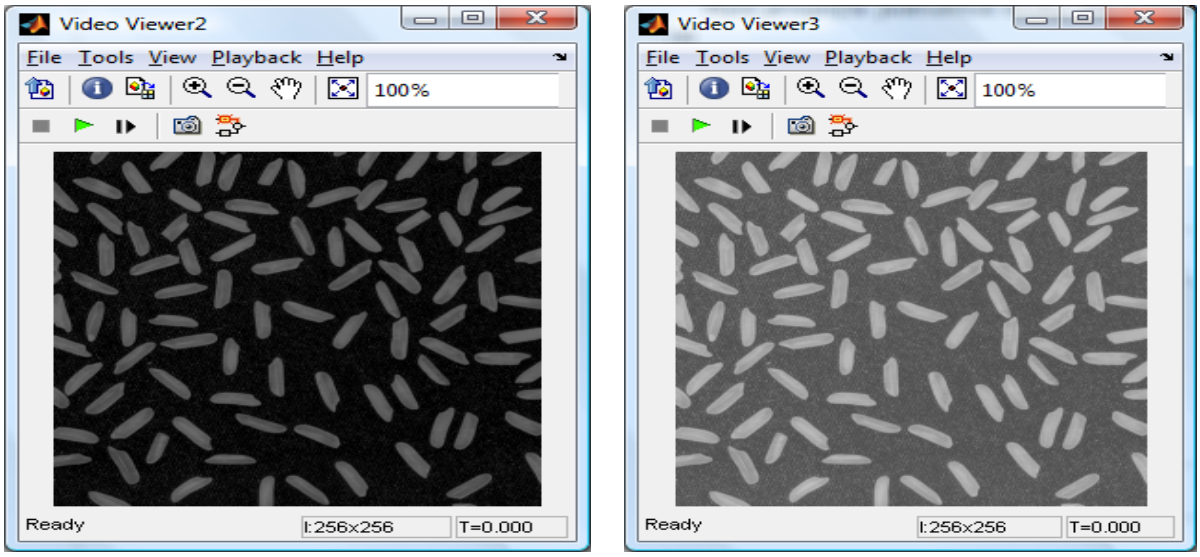
Nastavení parametrů Simulace

Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

Nyní propojte jednotlivé bloky dle schématu na Obr.8.22, nastavte parametry a můžete spustit simulaci.

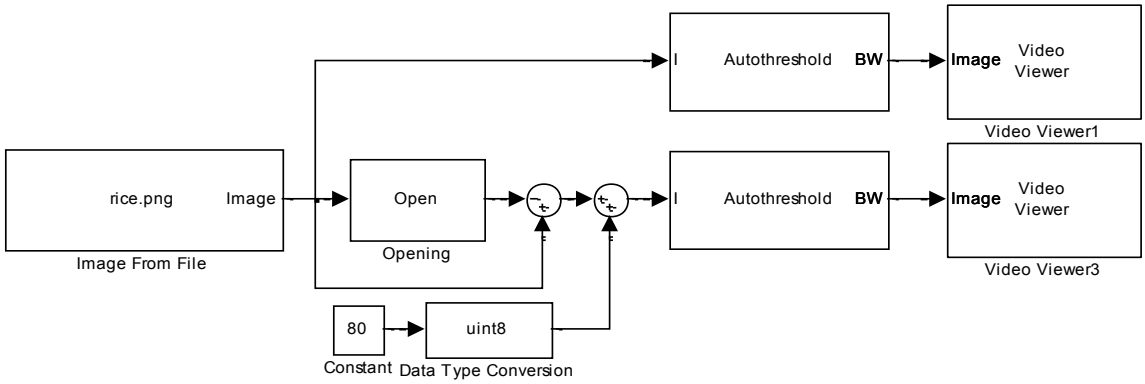


Obr.8.25 Vlevo originál obrázku se špatným osvětlením objektu, vpravo odhadnuté pozadí obrázku.



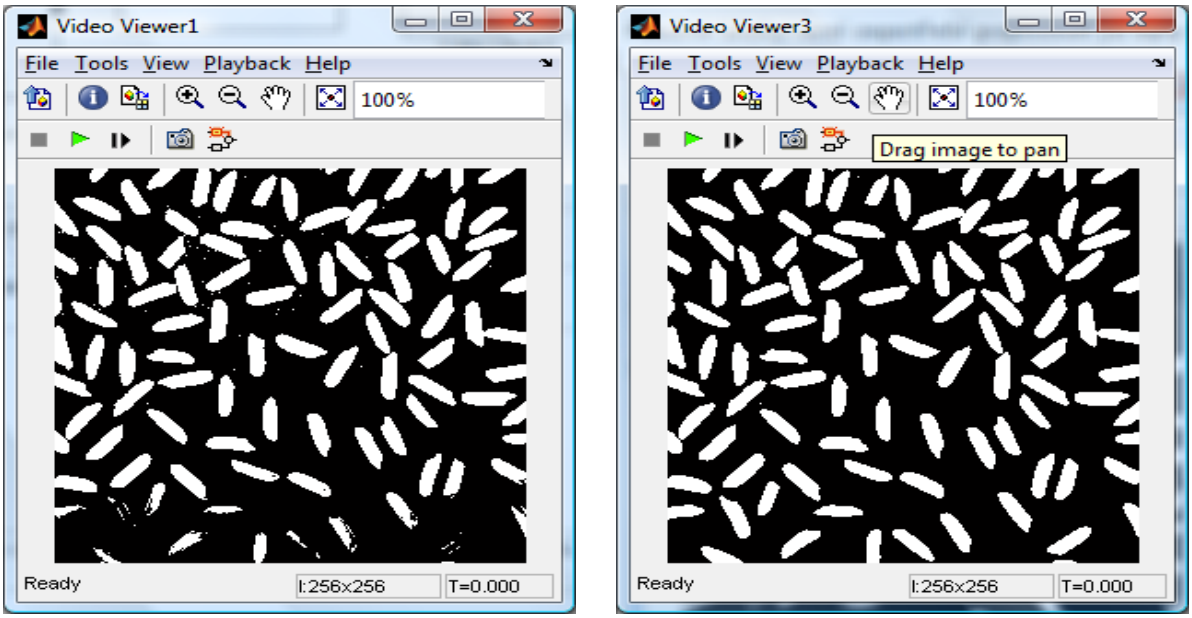
Obr.8.26 Vlevo korekce v tmavém provedení, vpravo korekce ve světlém provedení.

Pro srovnání korekce výsledného efektu korekce osvětlení bylo použito schéma na Obr.8.27.



Obr.8.27 Schéma zapojení modelu.

Použité bloky není zapotřebí blíže rozebírat, protože již byly v této kapitole popsány.



Obr.8.28 Vlevo segmentace bez použití korekce osvětlení, vpravo s použitím korekce osvětlení.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 08 Morfologicke_operace.swf.

Demonstrační video k této kapitole je k dispozici v adresáři Videa pod názvem video_pocitani_objektu.mpg

9. KOMPRESE OBRAZU

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat co je to ztrátová a bezztrátová komprese
- popsat jednotlivé kompresní metody pro statický obraz a video.
- popsat rozdíl mezi JPEG a MPEG kompresí.
- vyřešit zadanou úlohu v Matlab/Simulink a VIP

Výklad

9.1 Úvod

Obrazy jsou reprezentovány maticí hodnot, z čehož vyplývá velká spotřeba paměti. Spotřeba paměti nám limituje možnosti archivace obrazů i možnosti jejich transportu v počítačových sítích.

Bezztrátová komprese musí zaručit opětovnou obnovu původních dat. Zakódovaný soubor musí obsahovat stejné množství informace

U **ztrátové komprese** dochází ke ztrátám informace, to znamená, že data nejde obnovit úplně přesně do originální podoby. Z tohoto důvodu dochází - nebo může docházet - k jistému zkreslení. Ztrátová komprese se využívá hlavně pro kompresi dat určených k smyslovému vnímání (obrázky, zvuk, video). U těchto dat je důležitá co nejmenší velikost, a proto malé zkreslení nemusí být na závadu, pokud však nedochází k výraznějšímu snížení kvality.
U obrázků, jsou postupně definovány barvy jednotlivých bodů, přitom se většina barev mnohokrát opakuje. Jednou z možností, jak takový soubor komprimovat, je místo mnoha stejných čísel s kódem určité barvy za sebou použít takový způsob zápisu, kde bude kód barvy pouze jednou a u něj číslo, které udává, kolikrát se má tento kód opakovat. (Např. "aaaaaaaaaa" se dá zapsat jako "10a" a úspora je 7 míst).

9.2 Redukce barvonosné informace

Většinou máme obraz v digitální podobě v barevném prostoru RGB (aditivní míchání barev, Red, Green, Blue, bude probráno v následující kapitole). Tato reprezentace barev není ideální pro kompresi obrazu ani videa. Je to dáno tím, že je obecně známo, že lidské oko je daleko citlivější na změny jasu než na změny barev. Z těchto objektivních důvodů je vhodné použít této nedokonalosti lidského oka i při kompresi obrazu. K redukci barvonosné informace se barva reprezentuje nejčastěji v některém z prostorů YC_bC_r, *YUV*, *YIQ*. Složka *Y* je jas, zbývající dvě složky jsou barvonosné. Převodní vztahy mezi uvedenými prostory: [6]

$$Y = 0.299R + 0.5876G + 0.114B$$

$$C_b = B - Y \qquad C_r = R - Y$$

$$U = 0.565C_b \qquad V = 0.713C_r \tag{9.1}$$

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{9.2}$$

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.0311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{9.3}$$

Z důvodu dosažení redukce objemu barvonosné informace jsou složky C_b, U, I, C_r, V Q vzorkovány s nižší frekvencí než složka Y. Nejčastěji se používá formátů označovaných kódy: 4:2:0 nebo 4:2:2.

- 4:2:0 – používá pro barvonosnou informaci polovičního počtu vzorků na řádku a polovičního počtu řádků
  - čtveřice 2 x 2 pixelů obrazu je proto reprezentována čtyřmi hodnotami Y, jednou hodnotou $C_b(U,I)$ a jednou hodnotou $C_r(V,Q)$.

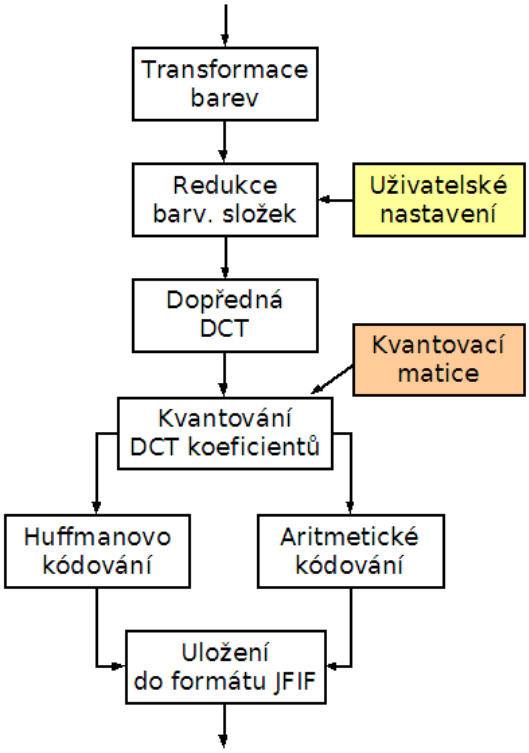
- 4:2:2 – používá polovičního počtu vzorků na řádku a plného počtu řádků.
  - čtveřice 2 x 2 pixelů obrazu je proto reprezentována čtyřmi hodnotami Y, dvěma hodnotami $C_b(U,I)$ a dvěma hodnotami $C_r(V,Q)$.

- 4:4:4 – formát, který používá plného počtu vzorků i pro složky barvonosné

JPEG komprese – redukce objemu barvonosné informace je nepovinná
MPEG-1 komprese – redukce se provádí vždy. Používá se barevného prostoru YC_bC_r, formátu 4:2:0
MPEG-2 komprese – podporuje i formát 4:2:2

9.3 Komprese JPEG (Joint Photographic Expert Group)

JPEG je metoda ztrátové komprese obrazu. Při této metodě komprese dat dojde v obraze k ztrátě určitého počtu dat. Tato ztráta ale díky nedokonalosti lidského oka není vizuálně postřehnutelná.
Při ztrátové kompresi obrazových dat pomocí standardu JPEG se vstupní rastrová data podrobují několika za sebou jdoucím operacím, jak je schematicky znázorněno na Obr. 9.1.



Obr. 9.1. Posloupnost operací při JPEG kompresi [8]

Stručný postup při JPEG kompresi dle Obr. 9.1. je následující:

1. Nejprve je provedena transformace barev z barvových prostorů *RGB*, *CMYK* či dalších (například CCD čipy mají vlastní barvový prostor) do barvového prostoru *YCbCr*. Tato transformace je bezeztrátová, tj. nedochází při ní k žádné ztrátě informací o obrázku.

2. Dále může podle konfigurace kodéru docházet k podvzorkování barvonosných složek. V barvovém prostoru *YCbCr* nese složka *Y* informaci o intenzitě pixelu a složky *Cb* a *Cr* informaci o barvě. Právě poslední dvě složky mohou být podvzorkovány, čímž dojde ke snížení objemu dat, ale i k určité (mnohdy zanedbatelné) ztrátě informace.

3. Další kroky jsou odděleně prováděny pro složku *Y* a pro barvonosné složky. Na bloky 8×8 hodnot je aplikována diskrétní kosinová transformace (DCT), která je už z principu bezeztrátová. Výsledkem DCT jsou bloky 8×8 hodnot, tentokrát ležící ve frekvenční rovině.

4. Bloky DCT o velikosti 8×8 hodnot jsou kvantovány pomocí vypočtených kvantizačních tabulek. Výsledkem je stav, kdy je mnoho hodnot v tomto bloku nulových, čehož se využívá v následujícím kroku zpracování. Právě při kvantizaci může docházet k největší ztrátě informace a tím i ke kýženému snížení bitové rychlosti (bitrate).

5. Kvantované DCT koeficienty jsou následně kódovány pomocí aritmetického či Huffmanova kódování. Aritmetické kódování je sice účinnější o cca 10%, je však mnohem výpočetně i aritmeticky náročnější. Proto se u většiny obrázků používá Huffmanovo kódování, které je sice méně účinné, ale může být prováděno i na málo výkonných čipech (například v mobilních telefonech či fotoaparátech).

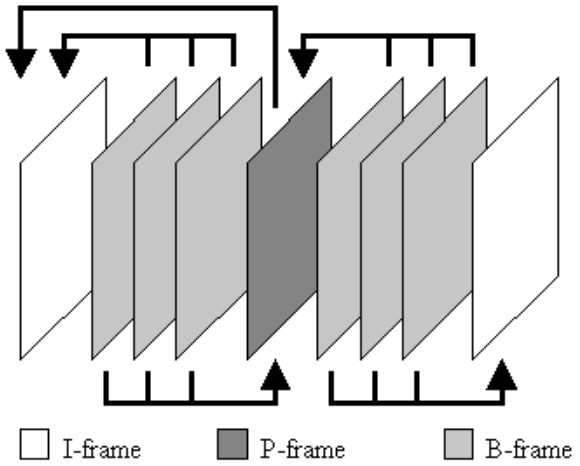
6.Posledním krokem zpracování je uložení vytvořených dat do souboru typu JFIF/JPEG. V podstatě se jedná o „obalení“ vzniklého datového toku vhodnou hlavičkou, přidání dalších doplňujících informací (včetně populárního EXIF či náhledového obrázku) a zakončení celého souboru patičkou.

9.4 Komprese MPEG (Moving Picture Expert Group)

MPEG je zkratkou pro Motion Pictures Experts Group. Tato skupina se zabývala kompresí videa a snažila se tento formát standardizovat. Formát byl dokončen v roce 1991. Kodek pracuje s rozlišením 352x288 při rychlosti 25 snímků za sekundu a datovém toku 1500bit/s.
MPEG používá pro kompresi tří druhy snímků I, B a P. I snímky (Intra Pictures) jsou základním kamenem ostatních snímků. Různé části snímku se mohou komprimovat různým stupněm komprese. I snímek má největší velikost. Po I snímku, většinou následuje P snímek, který se komprimuje pomocí předešlých I nebo P snímků. B-snímky (Bidirectional Pictures) jsou pak dopočítávané pomocí předchozích I nebo P-snímků. Mají nejmenší velikost na disku. Celá sekvence snímků (od jednoho I po další I snímek) se pak nazývá GOP (Group of Pictures). Běžné pořadí snímků je IBBPBBPBBPBBPBBPBB. Toto pořadí lze samozřejmě změnit spolu s četností I-snímků. Toto vede ke zlepšení kvality videa. Komprimované video obsahující proměnlivé vzdálenosti mezi klíčovými snímky se pak nazývá VKI (Variable Keyframe Interval). Komprese MPEG-1 se nehodí pro střih videa z důvodu vzdálených klíčových I-snímků.
MPEG1 má ovšem jednu obrovskou nevýhodu. Neumí totiž pracovat s prokládanými snímky a pracuje pouze s celými. Proto vznikl MPEG2 (v roce 1994) a stal se standardem pro kompresi digitálního videa. V podstatě se ale jedná o stejný princip komprese. MPEG2 podporuje vyšší rozlišení- oproti 352x288 je to 720x576. Dále podporuje proměnlivý datový tok, což se hodí pří vysoce dynamických scénách, anebo naopak pří statických scénách. MPEG2 má tedy oproti MPEG1 mnohem vyšší kvalitu obrazu. Vše ale není bez vrásek, a tak i tento formát přináší úskalí ve formě mnohem vyššího zatížení procesoru při přehrávání. Také se nehodí pro střih ze stejných důvodů jako MPEG1. MPEG2 se v dnešní době používá jako formát videa na DVD.

Jak již bylo uvedeno, standard MPEG používá 3 typy snímků, a to I, P, B.

- Snímek typu I (Interframe) představuje plnohodnotný snímek. Obsahuje veškeré informace a stává se základním pro další typy snímků.
- Snímek typu P (Predicted) je kódovaný ve vztahu k předešlému snímku typu I. V praxi to znamená, že je závislý na předchozím základním snímku. Snímek typu P nese informace o změně mezi snímkem I a jím samým. Jak se vyhodnocuje změna mezi snímky? Snímek je rozdělen na jednotlivé makrobloky a ty se vezmou do porovnávače. Porovnají se v bináru a totožné makrobloky se nepřenáší (modrá obloha, jednolité stejnobarevné větší plochy představují typický příklad pro vynechávání). Při promítání se tato vynechaná místa nahradí makrobloky ze základního snímku I.
- Poslední typ snímku je snímek typu B (Bidirectionally Predicted) a pro představu je nejsložitější. Tento snímek je závislý jak na předchozím snímku (a je jedno jestli I nebo P) a stejně tak i na následujícím snímku. Porovnává tedy makroblok, jak s předchozím snímkem, tak s následujícím a při zobrazování si půjčuje makrobloky z předchozího i následujícího snímku (tento typ neznal původní MPEG-1). Používá se při projíždění kamery krajinou, kde se krajní místa doplňují nadcházejícím snímkem typu P nebo I, tedy opět nebe a další plochy.



Obr. 9.2. Sled snímků při formátu MPEG (I, P, B)

Aby bylo co nejvíce ušetřeno místo, jsou obrázky ještě navíc překódovány diskrétní kosinovou transformací (DCT), která z obrázku odstraní ty části, které nejsou lidskému oku viditelné. Je založena na Fourierově transformaci, která převádí signál z časové oblasti do frekvenční jako součet sinů různé frekvence a velikosti amplitudy. Principiélně se jedná o to, že složkz o nízké frekvenci reprezentují velké plochy a složky o větší frekvenci oblasti s více detaily. Např. makroblok 8x8 bodů je překódován do 64 frekvencí, které mají na sebe jistou návaznost a možné je uchovat pouze jako změnu v jejich velikosti. První koeficient je tzv. stejnosměrná složka, která určuje základní charakteristiku makrobloku. Vyšší koeficienty jsou většinou 0 a proto se ořezávají. Počet oříznutí dává následně kvalitu výsledného obrazu a kompresi. Toto oříznutí většinou ve výsledném pohyblivém obrazu není viditelné, pouze při bližším zkoumání je vidět zdvojení ostrých přechodů (tzv. duch). Tomuto principu se říká kvantizace.

Shrnutí pojmů

Komprese nám slouží k zmenšení objemu dat daného souboru tím, že odstraníme nadbytečné informace,které lidské smysly nejsou schopné zachytit. Existuje **ztrátová** a **bezztrátová** komprese. **Ztrátová komprese** využívá nedokonalostí lidského oka a vypouští z dat nadbytečné informace, které člověk není schopen postřehnout. **Bezztrátová informace** zaručuje vždy stejný originál dat před i po kompresi. Pro obrazy se nejčastěji používá komprese **JPEG**. Pro video je často používaná komprese **MPEG**.

Otázky

1. Jaký je rozdíl mezi ztrátovou a bezeztrátovou kompresí?
2. Vysvětlete princip JPEG komprese.
3. Vysvětlete princip MPEG komprese.

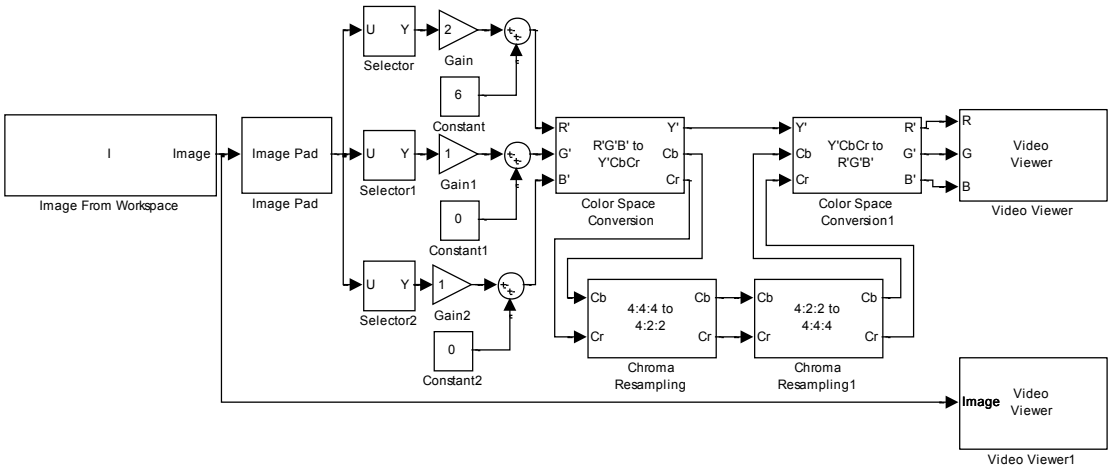
CVIČENÍ

Komprese obrazu

Ve cvičení se zaměříme na zpracování statického obrazu a videa pomocí kompresních metod. Ukážeme se nejdříve kompresní metody pro úpravu jasové složky. Vyzkoušíme si chromatické vzorkování obrazu jako jednu z kompresních metod. Dále si ukážeme jak vytvořit model pro zpracování statického obrazu. Pro zpracování videa použijeme demonstrační model vytvořený v systému Matlab/Simulink.

Chroma resampling

Chroma resampling je anglický ekvivalent chromatického vzorkování. Dle schématu na 9.3 sestavíme v Simulinku model soustavy.



Obr.9.3 Schéma zapojení modelu.

Jednotlivé bloky naleznete:

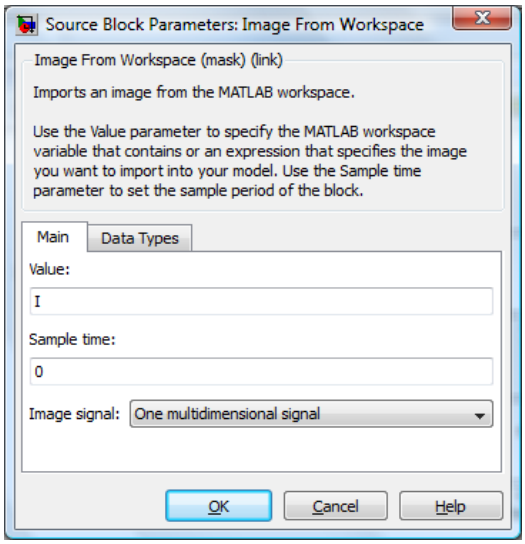
- Blok ***Image Pat*** nalezneme v menu ***Video and Image Processing blockset -> Utilities***
- Bloky ***Chroma Resampling*** nalezneme v menu ***Video and Image Processing blockset -> Conversions.***

Nastavení vstupních parametrů

Před tvorbou modelu v Simulinku nejdříve uložíme vstupní obrázek do workspace Matlabu pomocí následujících dvou příkazů:

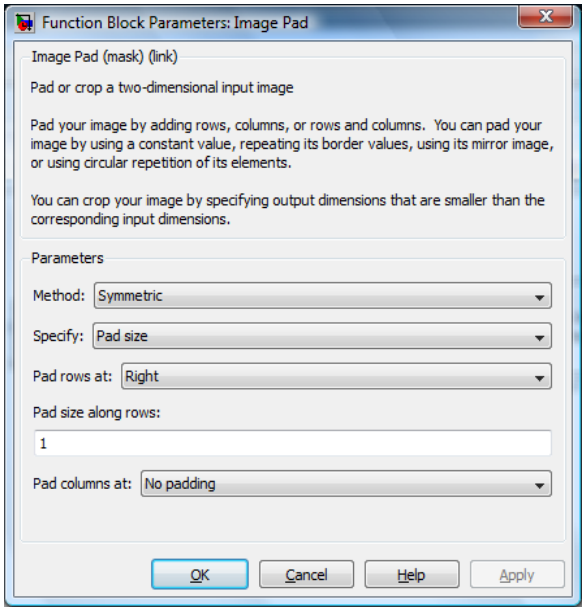
I= imread('autumn.tif');	načte vstupní obrázek do workspace
imshow(I)	zobrazí obrázek uložený do workspace

- Parametry bloku ***Image from Workspace*** nastavíme dle následujícího obrázku (Obr.9.4). Zadáme adresu vstupního obrázku pomocí parametru Filename = I. Parametr **Sample time** nastavíme na hodnotu **0**. Parametr **Image signal** nastavíme na hodnotu **One multidimensional signal**.



Obr.9.4 Nastavení parametrů bloku Image From Workspace.

- Parametry bloku ***Image Pad*** nastavíme dle Obr. 9.5. Parametr **Method** nastavíme na hodnotu **Symmetric**, parametr **Pad rows at** na hodnotu **Right**, parametr **Pad size along rows** nastavíme na hodnotu **1** a parametr **Pad columns at** na hodnotu **No padding**. Ostatní parametry ponecháme defaultně přednastaveny.



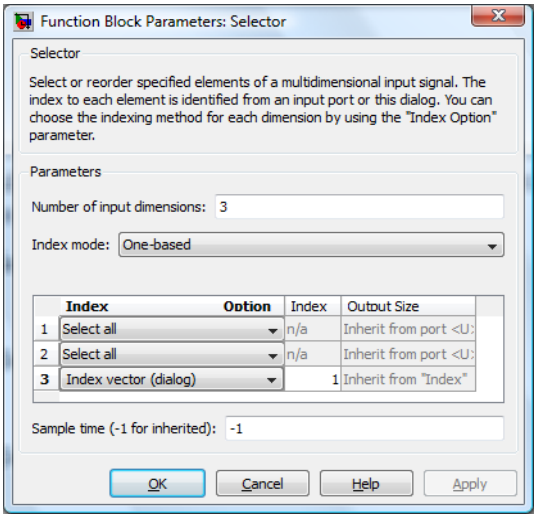
Obr.9.5 Nastavení parametrů bloku Image Pad.

- V blocích ***Gain*** nastavíme v záložce **Signal Attributes** parametr **Data output type** na hodnotu **Inherit: Same as input type**. V záložce **Parameter Attributes** nastavíme parametr **Parameter data type** na hodnotu **Inherit: Same as input**.

- V blocích Constant nastavíme v záložce **Signal Attributes** parametr **Output data type** na hodnotu **unit8**.

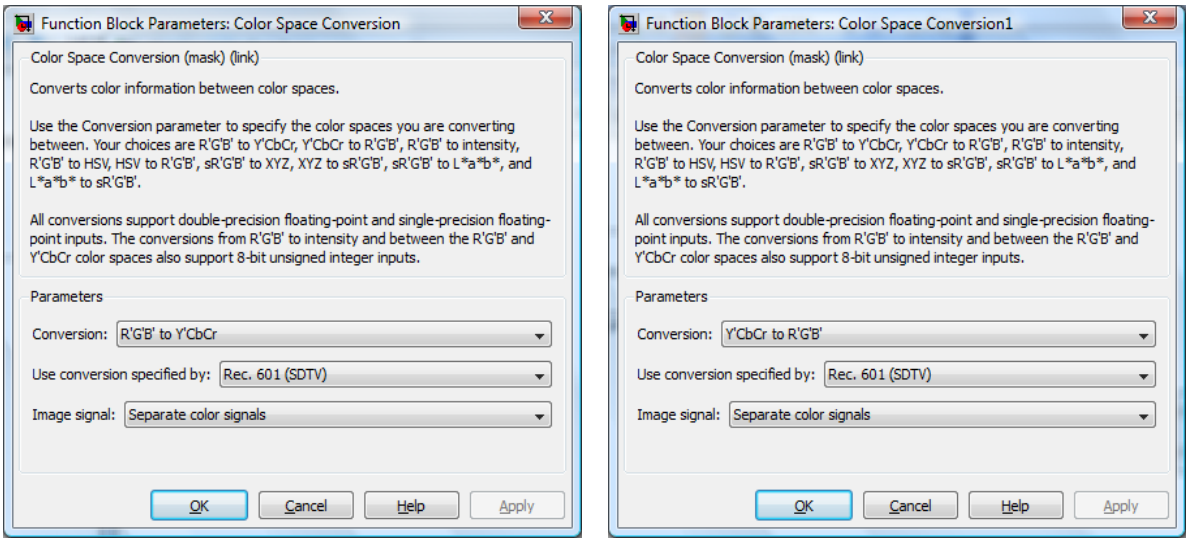
- Bloky ***Sum*** ponecháme defaultně přednastaveny.

- Parametry bloku **Selector** nastavíme dle Obr.9.16. Parametr **Number of input dimesions** nastavíme na hodnotu **3**, parametr **Index mode** ponecháme přednastavený na hodnotu **One-based**. Parametry **Index option** nastavíme následujícím způsobem. Pro 1. a 2. řádek vybereme možnost **Select all**, pro 3. řádek vybereme hodnotu **Index vector(dialog)** a nastavíme **Index** na hodnotu **1**. Stejné nastavení provedeme u zbývajících dvou bloků. V bloku ***Selector1*** nastavíme **Index** na hodnotu **2** a v bloku ***Selector 2*** nastavíme **Index** na hodnotu **3**.



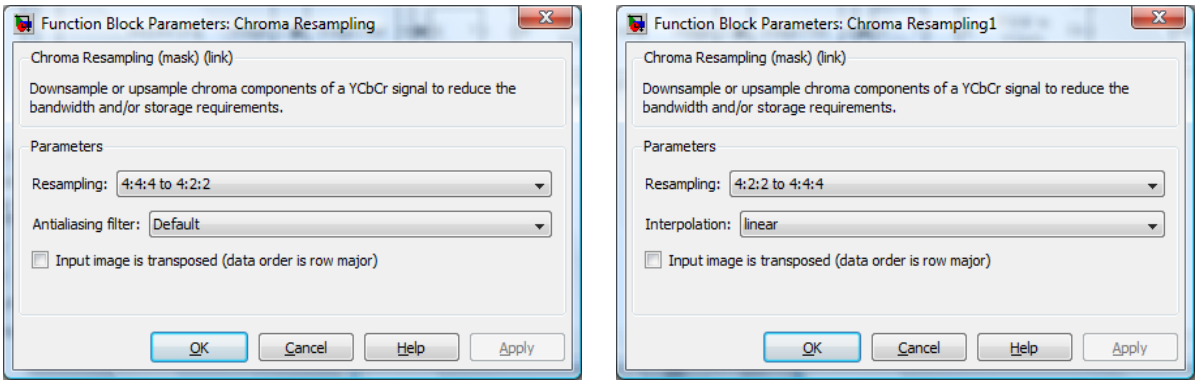
Obr.9.6 Nastavení vstupních parametrů bloku Selector.

- Bloky ***Color Space Conversion*** nastavíme dle následujících obrázků (Obr.9.7). Pro vstupní signál nastavíme parametr **Conversion** na hodnotu **RGB to YcbCr**, parametr **Use conversion specified by** ponecháme přednastaven na hodnotě **Rec.601(SDTV)** a parametr **Image signal** nastavíme na hodnotu **Separate color signals**. Pro výstupní signál nastavíme parametr **Conversion** na hodnotu **YcbCr to RGB**. Ostatní parametry nastavíme stejně.



Obr.9.7 Vlevo nastavení pro vstupní signál, vpravo nastavení pro výstupní signál.

- Bloky ***Chroma Resampling*** nastavíme dle následujících obrázků (Obr.9.8). Parametr **Resampling** vstupního bloku ponecháme defaultně nastavený na hodnotu **4:4:4 to 4:2:2**. Ostatní parametry ponecháme defaultně přednastaveny. Parametr **Resampling** výstupního bloku nastavíme pro opačnou kompresi na hodnotu **4:2:2 to 4:4:4**. Parametr **Interpolation** ponecháme přednastavený na hodnotě **linear**.



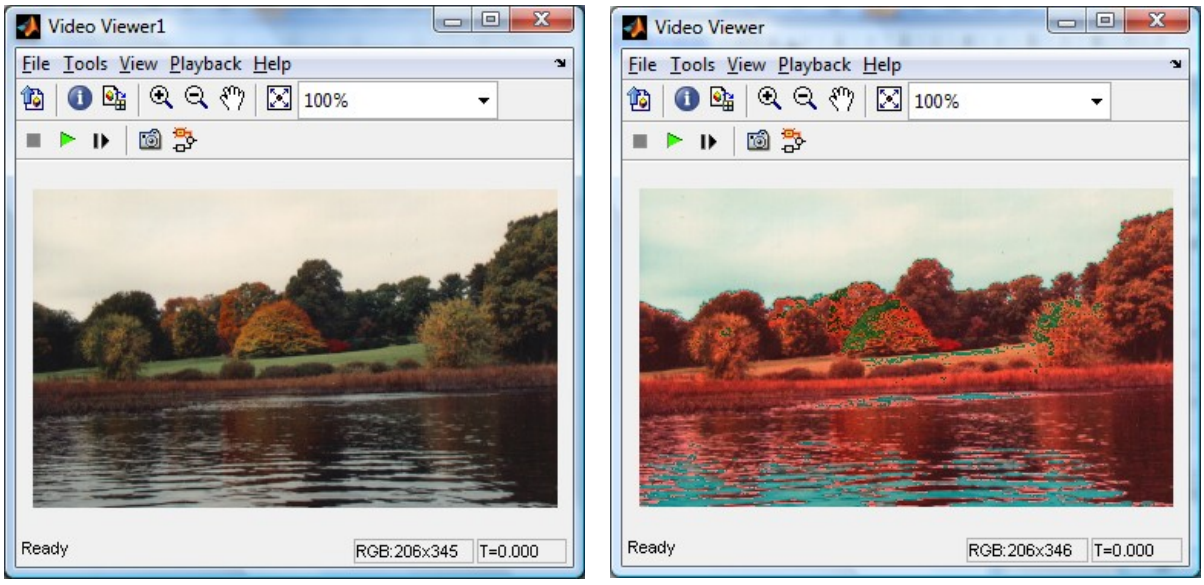
Obr.9.8 Vlevo nastavení vstupního bloku pro kompresi, vpravo nastavení výstupního bloku pro kompresi.

- Parametr **Image signal** bloku ***Video Viewer*** nastavíme na hodnotu **Separate color signals**.

Nastavení parametrů Simulace

Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

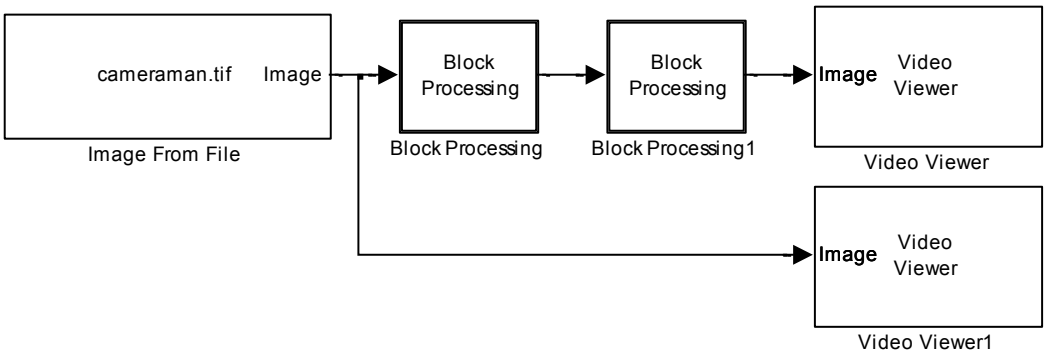
Nyní propojíme jednotlivé bloky dle schématu na Obr.9.3, nastavíme parametry pro simulaci a můžeme spustit simulaci.



Obr.9.9 Vlevo původní obrázek, vpravo obrázek po kompresi s upravenou červená složkou.

Komprese obrazu

Dle schématu na Obr.9.10 sestavíme v Simulinku model soustavy.



Obr.9.10 Schéma zapojení modelu pro kompresi obrazu.

Jednotlivé bloky naleznete

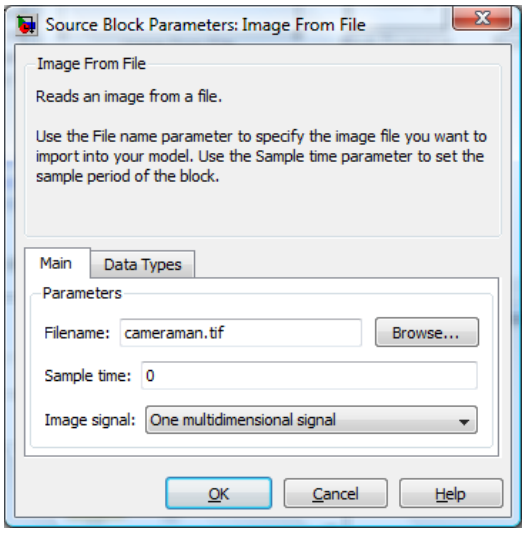
- Blok ***Image From File*** nalezneme v menu ***Video and Image Processing Blockset => Sources***
- Blok ***Block Processing*** nalezneme v menu ***Video and Image Processing Blockset => Utilities***
- Blok ***Video Viewer*** nalezneme v menu ***Video and Image Processing Blockset => Sinks***

Jednotlivé bloky pro tvorbu subsystémů

- Blok ***2-D DCT*** nalezneme v menu ***Video and Image Processing Blockset => Transforms***
- Blok ***Selector*** nalezneme v menu ***Simulink => Signal Routing***
- Blok ***Image Pad*** nalezneme v menu ***Video and Image Processing Blockset => Utilities***
- Blok ***2-D IDCT*** nalezneme v menu ***Video and Image Processing Blockset => Transforms***

Nastavení vstupních parametrů

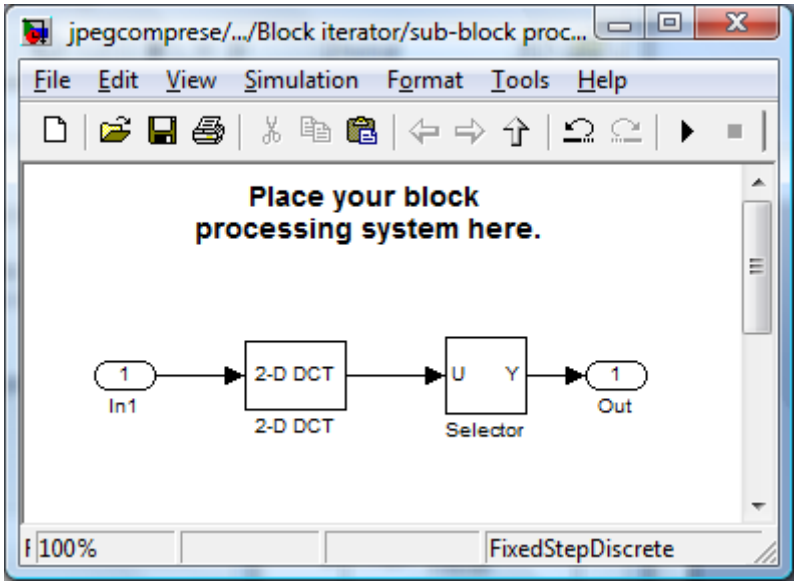
- Parametry bloku ***Image From File*** nastavíme dle Obr.9.11. V záložce Main nastavíme parametr **Filename** na hodnotu **cameraman.tif**, parametr **Sample Time** nastavíme na hodnotu **0** a parametr **Image Signal** nastavíme na hodnotu **One Multidimensional Signal**. V záložce Data Types nastavíme parametr **Output data type** na hodnotu **double**.



Obr.9.11 Nastavení parametrů bloku Image From File.

Komprese obrazu

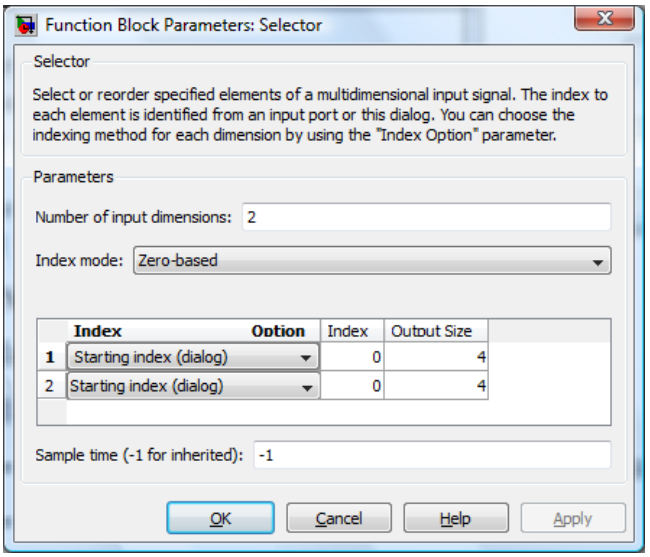
- Nastavení parametrů bloku ***Block Processing*** provedeme pomocí Obr.9.12. Je zapotřebí vytvořit subsystém pro přenos. To provedeme pomocí tlačítka **Open Subsystem**. Následně je zapotřebí vytvořit schéma dle následujícího modelu.



Obr.9.12 Schéma zapojení modelu subsystému.

- Vstupní parametry bloku 2-D DCT ponecháme defaultně přednastaveny.

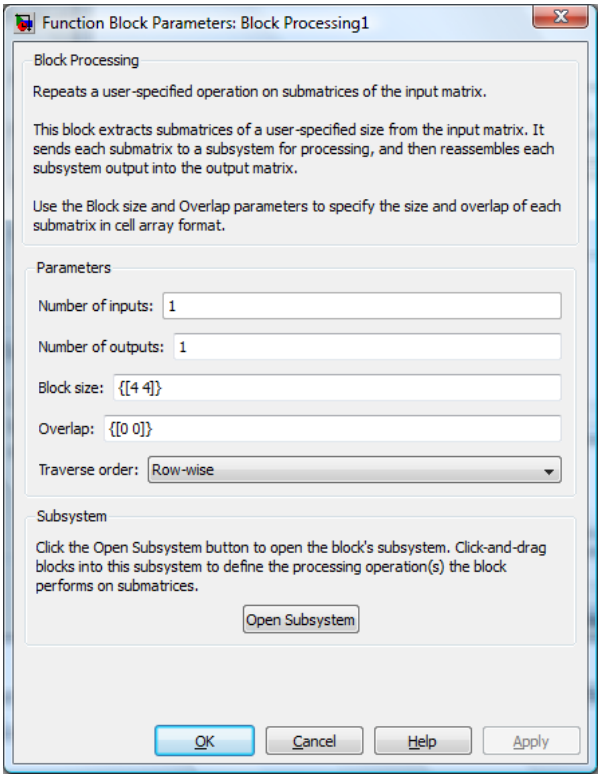
- Parametry bloku ***Selector*** nastavíme dle Obr.9.13. Parametr **Number of input dimensions** nastavíme na hodnotu **2**, parametr **Index mode** nastavíme na hodnotu **Zero-based**. Parametr **Index Option** v prvním řádku nastavíme na hodnotu **Starting index (dialog),** index na hodnotu **0** a **Output size** na hodnotu **4**. Stejné nastavení provedeme i v druhém řádku.



Obr.9.13 Nastavení parametrů bloku Selector.

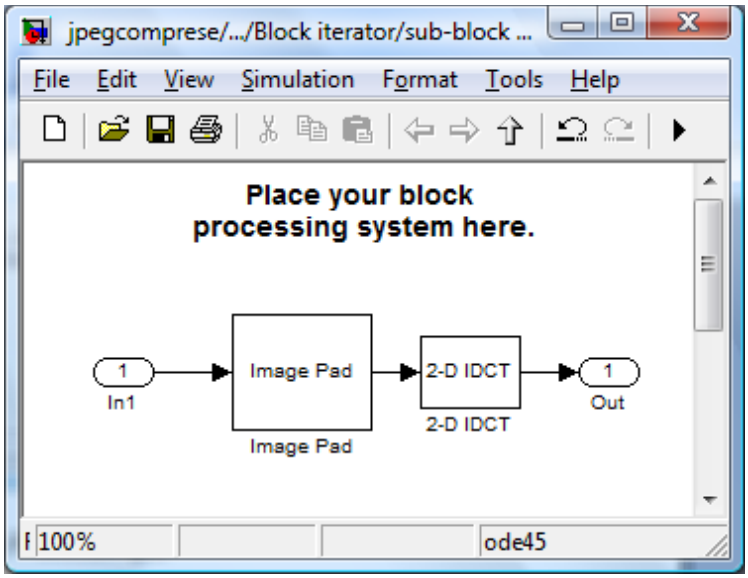
Nyní máme nakonfigurován blok ***Block Processing*** pro kompresi obrazu pro přesnost. Pomocí následujícího popisu si ukážeme jak takto upravený obraz transformovat zpět do časové domény a následně jej zobrazit jako komprimovaný obraz.

- Nastavení parametrů druhého bloku ***Block Processing1*** provedeme následujícím způsobem. Dle následujícího obrázku (Obr.9.14) provedeme nastavení parametrů. Parametr Block size nastavíme na hodnotu {[4 4]}. Dále je zapotřebí vytvořit subsystém pro zpětný přenos.



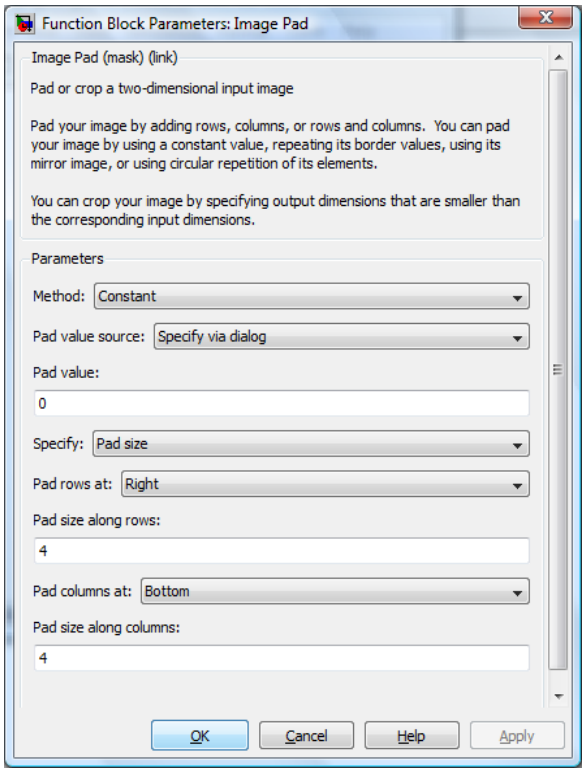
Obr.9.14 Nastavení parametrů bloku Blok Processing1.

To provedeme pomocí tlačítka **Open Subsystem**. Následně je zapotřebí vytvořit schéma dle modelu na Obr.9.15.



Obr.9.15 Schéma zapojení modelu subsystému.

- Nastavení parametrů bloku ***Image Pad*** provedeme dle následujícího obrázku. Parametr **Pad rows at** nastavíme na hodnotu **Right**. Parametr **Pad size along rows** nastavíme na hodnotu **4**. Parametr **Pad columns at** nastavíme na hodnotu **Bottom** a parametr **Pad size along columns** nastavíme na hodnotu **4**. Ostatní parametry ponecháme defaultně přednastaveny.



Obr.9.16 Nastavení parametrů bloku Image Pad.

- Parametry bloku ***2-D IDCT*** ponecháme defaultně přednastaveny.

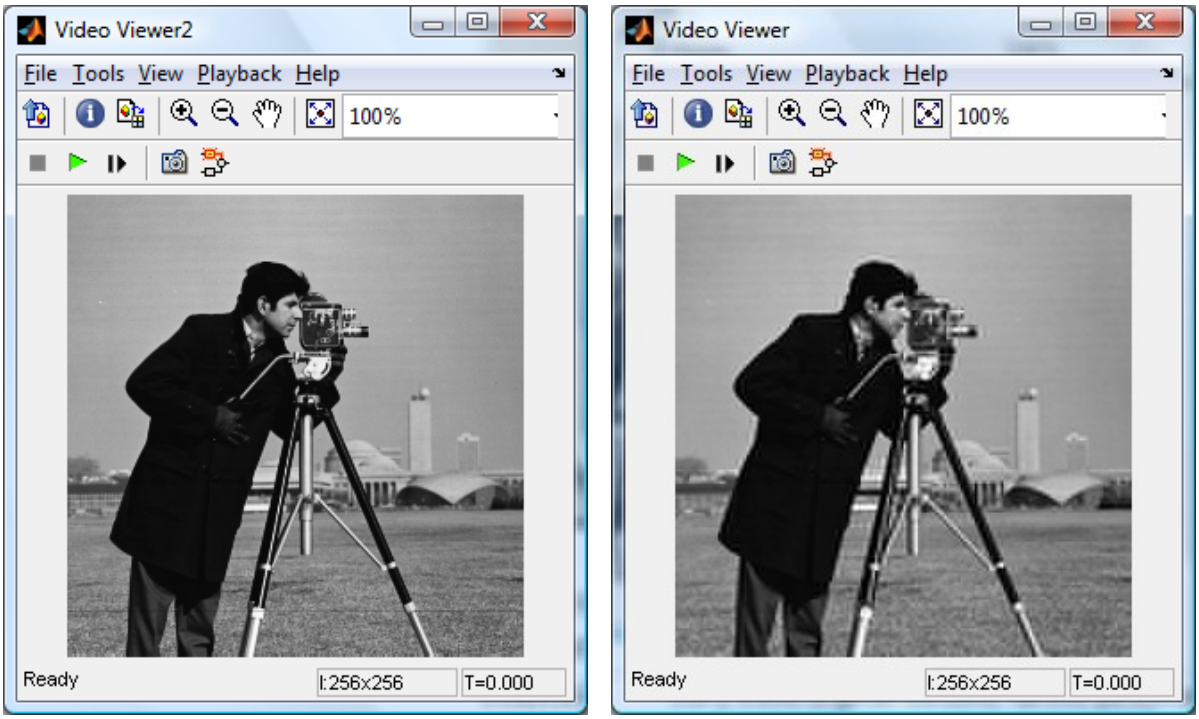
Subsystém uložíme a vrátíme se do první úrovně schématu viz. Obr.9.10.

- Parametry bloků ***Video Viewer*** ponecháme defaultně nastaveny.

Nastavení parametrů Simulace

Úpravu provedeme z menu ***Simulation*** -> ***Configuration Parameters***. Parametr ***Stop time*** nastavíme na hodnotu ***0.*** Parametr ***Type*** nastavíme na hodnotu ***Fixed-step.*** Parametr ***Solve*** nastavíme na hodnotu ***Descrete (no continuous states).***

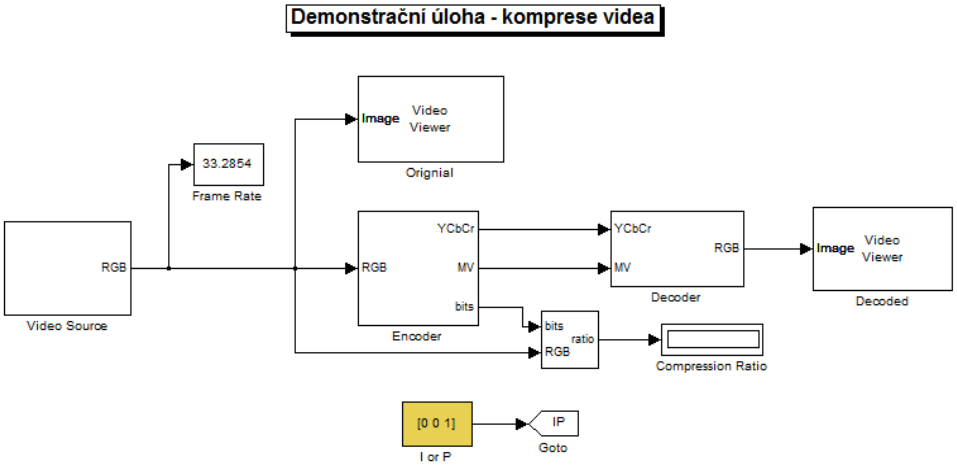
Nyní propojíme jednotlivé bloky dle schématu (Obr.9.10), nastavíme parametry pro simulaci a můžeme spustit simulaci.



Obr.9.17 Vlevo originál obrázku, vpravo po kompresi.

Video komprese

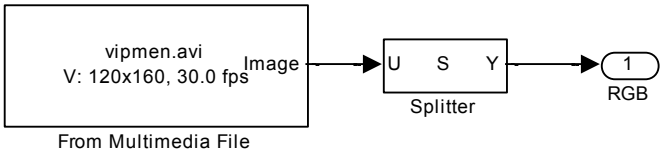
Pro ukázku komprese videa použijeme připravený demonstrační model přímo v programu Matlab/Simulink. Vyhledáme soubor vipcodec_color.mdl pro barevnou verzi nebo vipcodec.mdl pro jasovou verzi a spustíme model, viz. Obr.9.18.



Obr.9.18 Schéma zapojení modelu.

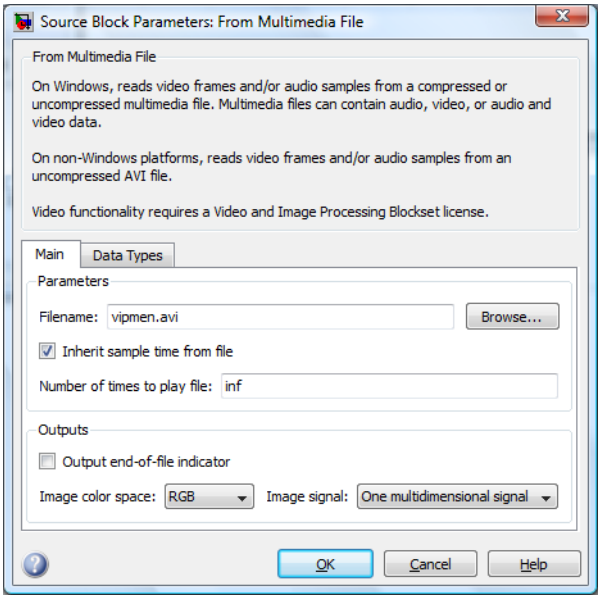
Nastavení vstupních parametrů

Vstupní video snímek určení pro přehrávání nastavíme následujícím způsobem. Otevřeme blok Video Source.



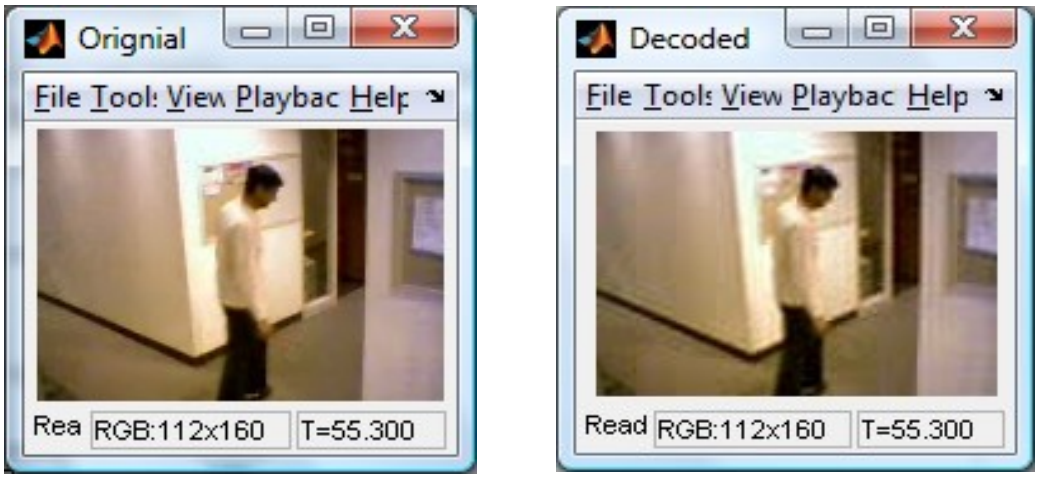
Obr.9.19 Schéma zapojení bloku Video Source.

V bloku From Multimedia File nastavíme parametr Filename na adresu videa dle Obr.9.20. Ostatní parametry ponecháme defaultně přednastaveny.



Obr.9.20 Výběr vstupního videa.

Nastavení parametrů Simulace
Parametry pro simulaci ponecháme defaultně přednastaveny a můžeme spustit simulaci.



Obr.9.21 Vlevo originál videa, vpravo video po kompresi.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem Komprese.swf.

10. CCD SNÍMAČE

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat princip činnosti CCD snímačů
- popsat jednotlivé typy CCD snímačů
- správně aplikovat barevný filtr
- vyřešit problematiku sestavení měřícího řetězce pro web kameru, přenést záznam do počítače a pomocí programu Matlab/Simulink zpracovat zachycený obraz v reálném čase.

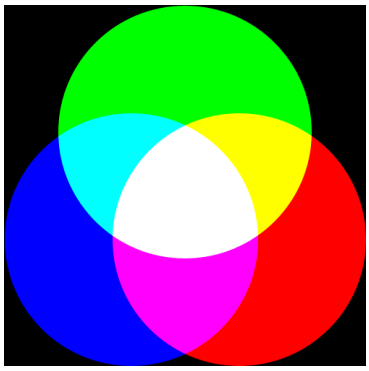
Výklad

10.1 Úvod

Zkratka **CCD** pochází z anglického charge coupled device a je jedním za dvou typů snímačů, které se používají ke snímání obrazu. Snímací prvky se objevují ve velkém množství nejrůznějších zařízení. Nejčatěji se CCD prvek používá v digitálních fotoaparátech a kenerech. Najdeme je však i v některých klasických videokamerách a používají se v různých obměnách v nejrůznějších zařízeních. Jejich funkce je zdánlivě jednoduchá. Jak jméno opisuje, tyto obvody snímají dopadající světlo a převádějí jej do podoby digitálního obrazu. Druhou využívanou technologií je **C-MOS** (complementary metal oxide semiconductor)

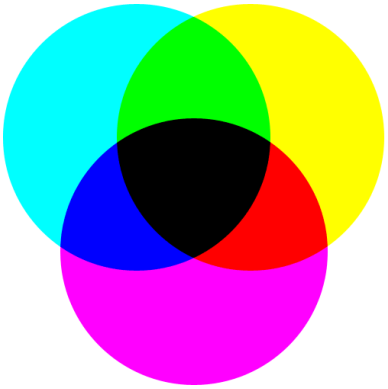
10.2 Barevné modely

Barevné modely slouží pro reprezentaci barev lidským okem. Veškeré barvy se dají složit smícháním tří základních barev a to červené (R-Red), zelené (G-Green) a modré (B-Blue). Z toho vychází základní barevný model RGB (aditivní míchání barev). Jednotlivé složky barev se sčítají a výsledek je světlo větší intenzity. Jak již bylo uvedeno, aditivní skládání barev pracuje se třemi základními barvami: červenou, zelenou a modrou. Podobá se skládání barevného světla - odpovídá vzájemnému prolínání tří světelných kuželů, které mají filtr odpovídající základní barvě. Tento způsob používají například monitory a displeje.



Obr.10.1 Aditivní míchání barev (RGB barevný model)

Narozdíl od modelu RGB, kdy se barvy sčítají, pracuje model CMY na principu odečítání barev. Proto je také nazýván subtraktivní barevný model. S každou přidanou barvou se ubírá část původního světla - světlo prochází jednotlivými barevnými vrstvami a je stále více pohlcováno. Výslednou barvu pak tvoří zbylé vlnové délky. Odpovídá míchání pigmentových barev. Základní barvy jsou azurová, purpurová a žlutá; smícháním všech těchto barev vznikne černá. Subtraktivní způsob míchání barev používají například tiskárny (např. různé druhy tiskových technik apod).

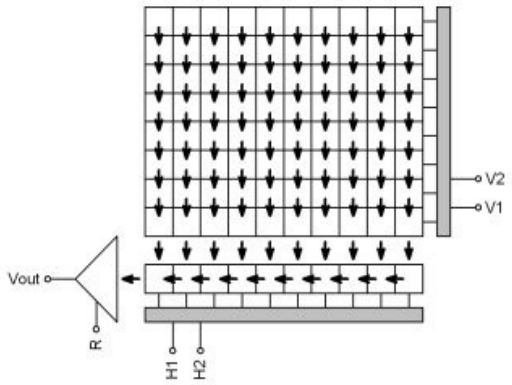


Obr.10.2 Subtraktivní míchání barev (CMY barevný model)

10.3 Princip CCD snímačů

Jak již bylo řečeno, CCD snímač přeměňuje dopadající intenzitu světla na elektrický náboj, neboli jinak řečeno, dopadající světlo na povrch křemíkové destičky v podobě fotonů se ukládá jako náboj v potenciálových jámách. Ty zabraňují volnému pohybu elektronů a tím i náboje po chipu a dochází tak k jeho kumulování (podobně jako se například plní sud přitékající vodou). Každá taková potenciálová jáma představuje jeden pixel CCD snímače. Velikost zachyceného náboje je hlavně ovlivňována intenzitou dopadajícího světla a dobou, po kterou necháme CCD chip světlu vystavený (Podobně jako objem vody v sudu odpovídá velikosti přítoku z hadice a době, po kterou je sud plněn). Zachycený náboj je nutné po nějaké době odebrat a převést na elektrický signál. Jinak by mohl dojít k přetečení potenciálové jámy, podobně jako může přetéct sud, když se z něj voda neodebírá. Obrazové CCD snímače obsahují matici pixelů (potenciálových jam), u nichž postupným přesouváním náboje z jedné

jámy do vedlejší dochází k jeho vysouvání na okraj chipu, kde je převáděn převodníkem na napěťový signál - viz Obr. 10.3.

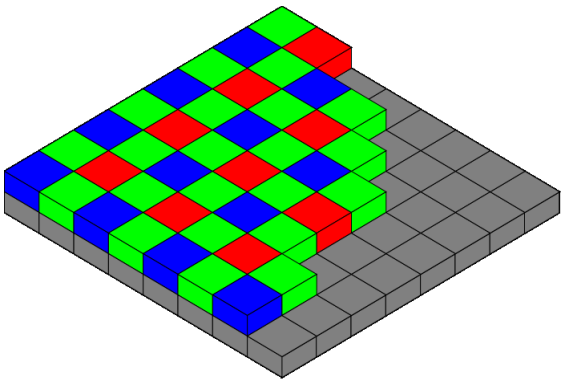


Obr.10.3 Princip převodu dopadajicího světla na napětí

Abychom vytvořili barevný obraz, potřebujeme snímat zvlášť ve třech barvách, obvykle v červené, zelené a modré. CCD snímač snímá ale pouze intenzitu světla, ne jeho barvu. Proto je zapotřebí k získání výsledné barvy světlo učitým způsobem filtrovat. V principu lze filtry k vytvoření barevného obrazu použít dvěma způsoby:

1. použít tři expozice toho samého obrazu pro jednotlivé filtry (červený, modý, zelený). Expozice barevného obrazu tímto způsobem chvíli trvá (je nezbytné měnit mezi expozicemi filtry), takže tento princip nelze použít u rychle se pohybujících objektů.
2. implementovat 3 barevný filtr přímo na CCD chip v podobě střídání sloupců pixelů pro jednotlivé barvy nebo "barevné" pixely rozložit dle Bayerovy masky. Zde se využívá vlastnosti lidského oka, které je více citlivé na jas než barevné podání. Tyto principy se využívají hlavně ve fotoaparátech a levných kamerách.

Bayerova maska (také Bayerův filtr, Bayerova mozaika) je pole barevných filtrů, který se používá k filtraci světla dopadajícího na snímací čip u většiny jednočipových digitálních fotoaparátů. Je pojmenována po svém tvůrci, Bryci E. Bayerovi z firmy Eastman Kodak, který ji patentoval v roce 1976, viz Obr. 10.4. Maska se skládá ze tří druhů filtrů, každý propouští jen světlo jedné vlnové délky – červené, modré nebo zelené. Jsou uspořádány v pravidelné mřížce, přičemž prvků propouštějících zelenou je 2× více než prvků propouštějící ostatní 2 barvy. Vyšší počet zelených elementů odráží vlastnosti lidského oka, které je nejcitlivější právě na tuto barvu.



Obr.10.4 Princip Bayerovy masky

10.4 Typy CCD snímačů

10.4.1 Prokládané snímače (interlaced)

Byly původně vyvinuty pro televizní a video techniku, ale můžeme se s nimi setkat i u mnoha digitálních fotoaparátů. Jejich konstrukce je přizpůsobena tomu, jak se zpracovává televizní obraz, tedy řádkově. Televizní obraz je rozložen na řádky a zvlášť se přenášejí liché a zvlášť sudé řádky. Pro tuto technologii jsou uzpůsobeny prokládané snímače. Ty po expozici nejprve zpracují liché řádky obrazu a pak zpracují sudé. U video kamer je to postup zcela přirozený a expozice sudých a lichých řádků je prováděna separátně stejně jako zpracování. U digitálních fotoaparátů je potřeba obraz zpětně složit. Úplně nejjednodušší variantou je pracovat pouze s lichými řádky. S tím jsme se mohli setkat třeba u CASIA QV-10, ale protože rozlišení snímače je příliš cenné, je to postup velmi výjimečný. Složitější cestou je elektronicky obraz složit. Samozřejmě je potřeba zaručit, že po dobu zpracování všech řádků se obraz nezmění, což se musí realizovat mechanickou závěrkou.

Postup práce u digitálního fotoaparátu je tedy následující :

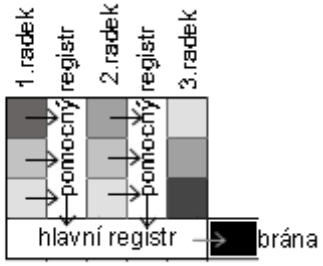
- Proběhne expozice senzoru.
- Pomocnými registry je odveden náboj z lichých řádků do hlavního registru, řádek po řádku.
- Následně je stejnou cestou zpracován náboj ze sudých řádků.
- Mimo snímač je obraz složen dohromady.
- Vše je znázorněno na obrázku. Ve skutečnosti jsou snímače u fotoaparátů otočeny o 90 stupňů, takže z řádků se stávají sloupce, což je vidět i na našem obrázku.

Výhody:

- Pro záznam video signálů je prokládané prvky ideální. Jednotlivé půlsnímky (sudé a liché řádky) se zpracovávají separátně a nevznikají žádné technické problémy. Díky masové výrobě video kamer jsou prvky poměrně levná a výrobní technologie je přijatelně komplikovaná.

Nevýhody pro digitální fotografii:

- Je nutné skládat obraz nebo se spokojit s polovičním rozlišením.
- Zpracování je pomalé a vylučuje rychlé časy závěrky.
- Vyžaduje mechanickou závěrku, aby nedošlo ke změně obrazu v době zpracování. Pokud není použita může docházet k rozostření nebo řádkovému posunu obrazu.
- Většina prvků používá pomocné registry a výrobci často používají interpolační algoritmy na dopočítání rozlišení.



Obr.10.5. Princip prokládaného snímače

10.4.2 Progresivní snímače (progressive)

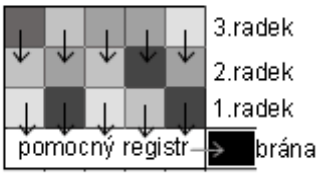
Druhou velkou skupinu snímačů tvoří takzvané progresivní snímače. Ty zpracovávají celý obraz najednou, což je sice technologicky složitější, ale přináší to velké výhody. Progresivní snímače se vyrábějí poměrně velmi komplikovanou technologií v malých sériích, takže jsou velmi nákladné. Co je důležité, informace se zaznamenává a zpracovává ve všech buňkách současně. To přináší vyšší ostrost, přesnost podání obrazu a samozřejmě to umožňuje použití elektronické závěrky s velmi krátkými časy. Celkově se tedy dá říct, že progresivní snímač je zatím nejlepším řešením pro digitální fotografii, které je k dispozici.
Zpracování může probíhat dvěma způsoby. U nejdražších modelů se používá technologie FTD (Frame Transfer Device), u které se ze všech buněk najednou odvede náboj do pomocných registrů a pak se dále sériově zpracovává. To je ta nejlepší a také nejkomplikovanější alternativa, takže se s ní můžeme setkat jen opravdu zřídka. Mnohem častější je takzvané řádkové čtení, které si detailněji popíšeme:
Dojde k expozici všech buněk.
Náboj z prvního řádku se přenese do pomocného registru a z něj je postupně zpracován bránou
Do prvního řádku se posune náboj z druhého řádku a postupně dojde k posunu náboje po celém snímači o jeden řádek dolů.
Opakuje se postup od bodu jedna, dokud není načten celý obrázek.

Výhody :

- Přesné zachycení obrazu s minimálním zkreslením.
- Umožňuje velice krátké časy a použití elektronické závěrky

Nevýhody :

- Výrobně velmi nákladné a složité řešení
- Nevhodné pro videokamery z čehož plyne malosériová výroba
- Snímače jako takové barvu dopadajícího světla nerozlišují. Každá buňka registruje pouze intenzitu světla, nikoli jeho frekvenci, která udává barvu světla. Snímač samotný je tedy barvoslepý a přirozeným výstupem je obrázek ve škálách šedé. S černobílou fotografií se v současnosti vystačit nedá, takže je k dispozici celá řada technologií, jak rozlišovat barvu světla dopadajícího na snímač. V dnešním a příštím dílu si popíšeme běžně požívané technologie.



Obr.10.6. Princip progresivního snímače

10.4.3 Řádkové snímače

Asi nejjednodušším použitím barevných filtrů jsou třířádkové snímače. Tyto prvky mají tři řádky buněk a nad každou řádkou je umístěn jeden barevný filtr. To znamená, že první řádka zaznamenává pouze červenou složku světla, druhý řádek zelenou a třetí řádek modrou. K získání barvy jednoho konkrétního bodu je tedy potřeba, aby se snímač 3x posunul tak, aby požadovaný bod změřila v každém řádku jedna buňka. To je přesně princip jednoprůchodových stolních skenerů vybavených CCD snímačem. Snímací hlava obsahuje třířádkový snímač a optiku, která zaručuje, že snímač obsáhne přesně maximální šířku předlohy. Snímací hlava se pak postupně pohybuje po řádcích od

shora dolů a tím je zaručeno, že každý bod, který se snímá je přečten celkem 3x (každým řádkem snímače jednou). Podmínkou úspěchu je pochopitelně to, že snímací hlava se pohybuje tak, aby docházelo ke snímání krokově v každém bodě. Třířádkový (tri-linear) snímač se používá i v některých specifických oblastech digitální fotografie.

Výhody:
- poměrně snadná a levná výroba

Nevýhody:
- vhodné pouze pro statické scény se statickým osvětlením
- vyžaduje přesnou mechaniku pro posun snímače
- snímání trvá dlouho a neumožňuje krátkou expozici

10.4.4 Multi-shot

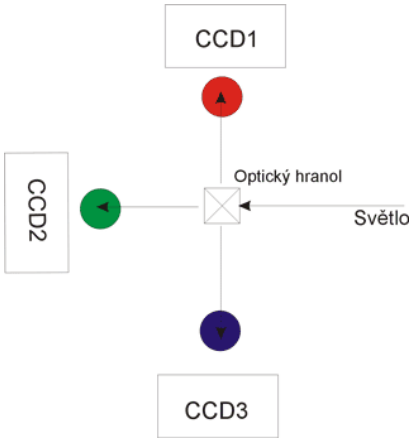
Další technologie, kterou si popíšeme je anglicky označena jako multi-shot, což ve volném překladu znamená více-snímková digitalizace. Princip je poměrně jednoduchý. Snímač jako takový není vybaven žádným barevným filtrem, ale barevný filtr je součástí optické soustavy. Snímání neprobíhá v rámci jedné expozice, ale celkem ve třech expozicích. Při každé expozici se vymění filtr se základní barvou a provede se snímání. Po dokončení všech tří expozic se pak obraz složí elektronicky dohromady.
Podmínkou úspěchu je samozřejmě statická scéna, fixní fotoaparát a stálé osvětlení. S touto technologií se můžeme setkat u studiových digitálních fotoaparátů, které lze používat pro fotografování produktů a jiných dalších statických scén.

Výhody:
- nesnižuje rozlišení a umožňuje velmi přesné snímání barev

Nevýhody:
- vhodné pouze pro statické scény se statickým osvětlením
- systém výměnných filtrů je poměrně velký

10.4.5 Multi-CCD

Poslední technologií, které se budeme dnes věnovat je použití více snímačů současně. Jde v podstatě o obměnu Multi-shotu odstraňující její největší nevýhodu. V jednom přístroji je umístěno více snímačů a před každým je jiný barevný filtr. Světlo přicházející z objektivu je pomocí optického hranolu rozloženo na jednotlivé snímače. V rámci jedné expozice je tedy možno provést snímání na všech CCD. Nejběžnější je varianta se třemi snímači, při které je před každým snímačem jeden z RGB filtrů.



Obr.10.7. Princip multi CCD

Výhody :

- nesnižuje rozlišení a umožňuje velmi přesné snímání barev
- umožňuje krátké expozice a snímání pohyblivých scén

Nevýhody :

- více snímačů výrazně zvyšuje cenu přístroje
- optika a více snímačů zvětšují rozměr fotoaparátu

Shrnutí pojmů

CCD snímač je fotocitlivý prvek sloužící k převodu intenzity světla na elektrický náboj. Existují 2 základní typy CCD snímačů – **prokládané** a **progresivní**. Pro tvorbu barvy se používají barevné modely. Základním je **RGB** – aditivní míchání barev. Dalším je např. **CMYK** – subtraktivní míchání barev.

Otázky

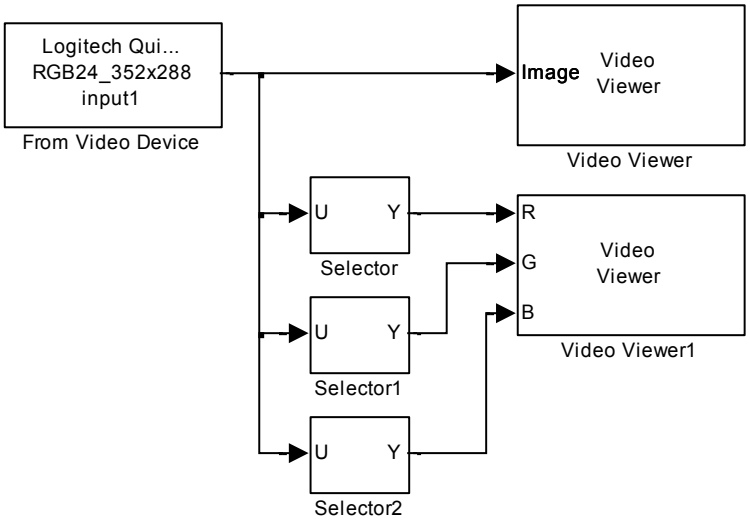
1. Na jakém principu pracují CCD snímače?
2. Jaké jsou nejpoužívanější typy CCD snímačů, jaké jsou jejich výhody a nevýhody a kde se používají?
3. Jakým způsobem dochází k míchání barev RGB a CMY?

CVIČENÍ

Zpracování obrazu

Cílem tohoto cvičení je naučit se zpracovávat obraz z web kamery a porovnat rozlišení jednotlivých kamer, které jsou v laboratoři k dispozici.

Zpracování obrazu z web kamery



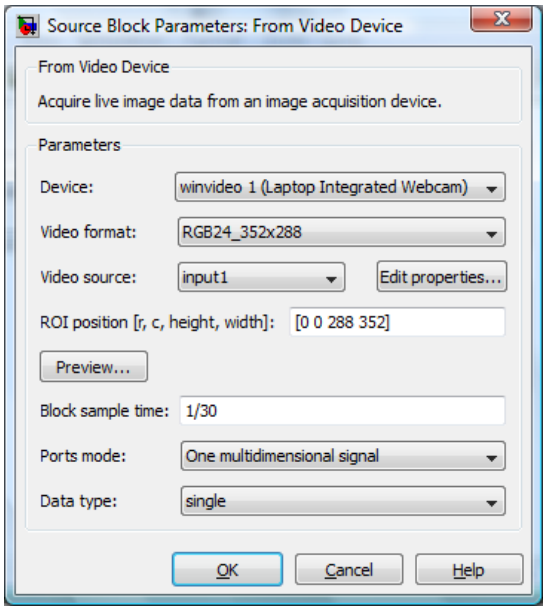
Obr.10.9 Schéma zapojení modelu.

Jednotlivé bloky naleznete

- Blok ***From Video Device*** nalezneme v menu ***Video and Image Processing Blockset => Sources***
- Blok ***Selector*** nalezneme v menu ***Simulink => Signal Routing***
- Blok ***Video Viewer*** nalezneme v menu ***Video and Image Processing Blockset => Sinks***

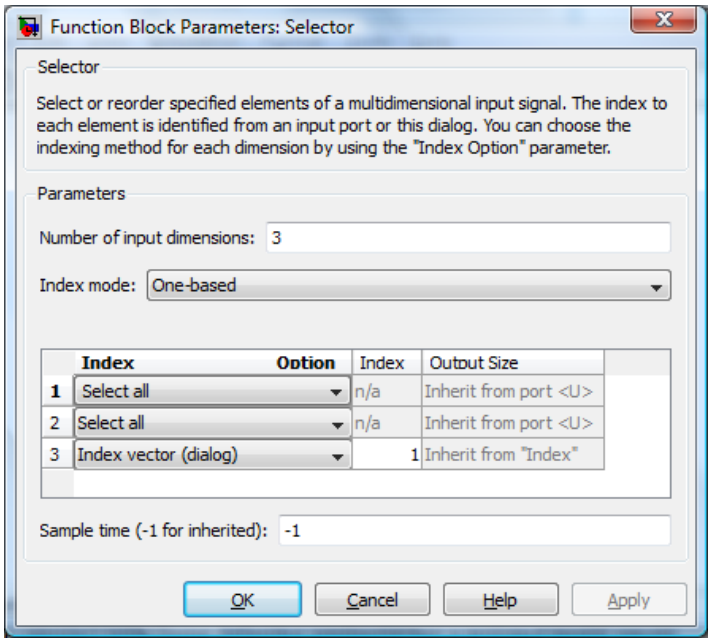
Nastavení vstupních parametrů

- Nastavením parametrů bloku ***Image From File*** provedeme dle následujícího obrázku (Obr.10.10). Parametr Device definuje zařízení, ze kterého chceme obraz snímat.



Obr.10.10 Nastavení parametrů bloku From Video Device

- Nastavení parametrů bloku ***Selector*** provedeme dle Obr.10.11. Parametr **Number of input dimesions** nastavíme na hodnotu **3**. Parametr **Index mode** ponecháme v nastavení **One-based**, parametr **Index Option** v řádku 1 nastavíme na **Select all**, parametr **Index Option** v řádku 2 nastavíme taktéž na hodnotu **Select all**. Parametr **Index Option** ve 3 řádku nastavíme na hodnotu **Index vector (dialog)** a hodnotu **Indexu** na **1**. Stejným způsobem postupujeme i u následujících dvou bloků Selector. S tím, že měníme pouze index. Tedy u bloku Selector 1 bude **index=2** a u bloku ***Selector 2*** bude **index=3**.

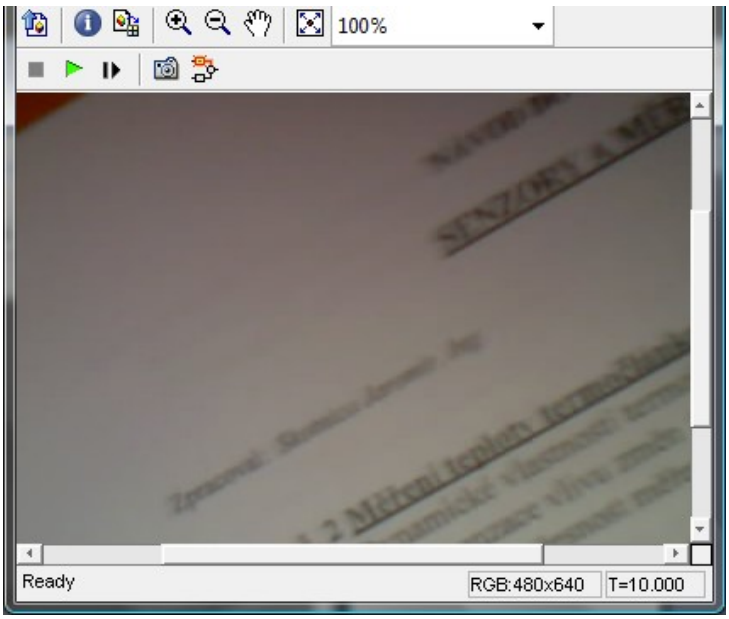


Obr.10.11 Nastavení parametrů bloku Selector.

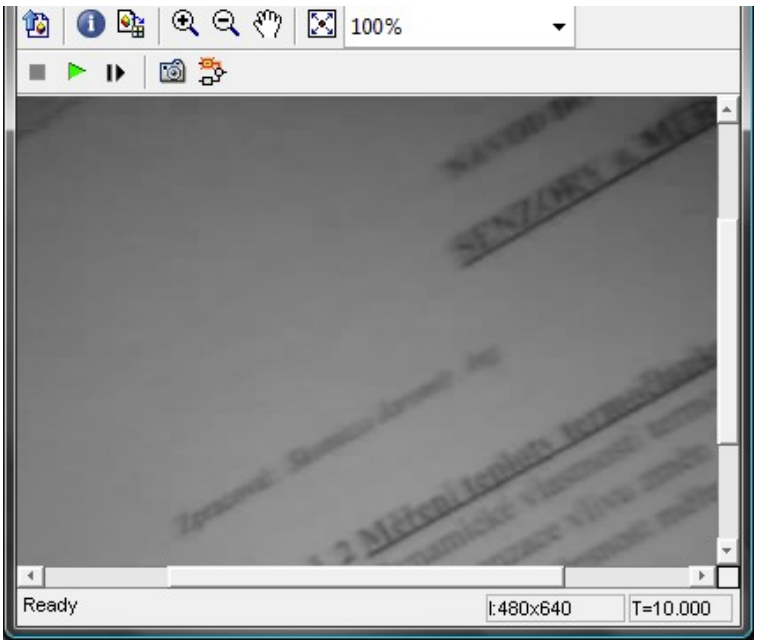
- Parametry bloku ***Video Viewer*** ponecháme defaultně přednastaveny.

Nastavení parametrů Simulace

Parametry pro simulaci ponecháme defaultně přednastaveny a můžeme model spustit.



Obr.10.12 Výstup z videokamery v laboratoři, pohled při rozkladu na RGB složky. Zobrazena B složka.



Obr. 10.13 Výstup z videokamery v laboratoři, při zobrazení RGB v jednom signálu.

CD-ROM

Výuková animace k této kapitole je vytvořena v prostředí Adobe Flash a je k dispozici v adresáři Animace pod názvem 10 CCD.swf.

V adresáři Videa se nacházejí ještě další demonstrační videa. Obsahují praktické ukázky použití digitálního zpracování obrazu pomocí systému Matlab/Simulink a VIP. Jedná se implementaci uvedených algoritmů a postupů pomocí bloků VIP, které řeší 2 názorné aplikace z průmyslové praxe. První z nich je aplikace pro rozpoznávání čárového kódu. Ta je k dispozici pod názvem video_carovy_kod.mpg. Druhá se zabývá rozpoznáváním dopravních značek. Naleznete ji pod názvem video_dopravni_znacky.mpg.

Další zdroje a použitá literatura

[1] Gonzales, R., Woods, R., Eddins, S.: Digital Image Processing using Matlab, Pearson Prentice Hall, USA 2004, ISBN 0-13-008519-7

[2] Hlaváč, V., Sedláček, M.: Zpracování signálů a obrazů, skripta ČVUT Praha, Vydavatelství ČVUT Praha 2005, ISBN 80-01-03110-1

[3] Image Processing Toolbox™ - User's Guide, The MathWorks, Inc., USA 2010

[4] Filipová, B.: Digitální zpracování a analýza obrazu, studijní materiály, VŠB-TU Ostrava 2004

[5] Hlaváč, V., Šonka, M.: Počítačové vidění, Grada Praha 1992, ISBN 80-85424-67-3

[6] Sojka, E.: Digitální zpracování a analýza obrazu, skripta VŠB-TU Ostrava, Vydavatelství VŠB-TU Ostrava 2000, ISBN 80-7078-746-5

[8] Klíma, M., Bernas, M., Hozman, J., Dvořák, P.: Zpracování obrazové informace, skripta ČVUT Praha, Vydavatelství ČVUT Praha 1999, ISBN 80-01-01436-3

[9] Video nad Image Processing Blockset™3 – User´s Guide, The MathWorks, Inc., 2010

[10] http://www.mathworks.com

Klíč k řešení

Kapitola 1.

1. Digitalizace obrazu je postup analogický k obecné digitalizaci spojité veličiny do digitální podoby. Zahrnuje v sobě vzorkování, kvantování a kódování. Je třeba myslet na Shannon-Kotělnikův teorém.
2. 256 úrovní
3. Jasové rozlišení je nejmenší rozlišitelná změna šedé úrovně v obrázku, Prostorové rozlišení je svázáno se vzorkováním, resp. se vzdáleností mezi nejbližšími vzorkovacími body.

Kapitola 2.

1. Histogram nám udává grafickou informaci o rozložení jednotlivých jasových složek v obraze.
2. Ekvalizace histogramu = vyrovnání histogramu. Ve vyrovnaném histogramu obrazu po transformaci jasové stupnice jsou jednotlivé jasové úrovně zastoupeny zhruba stejně četně. Ekvalizace zvýší kontrast pro úrovně jasu blízko maxim histogramu a sníží kontrast blízko minim histogramu.

Kapitola 3.

1. Hrana představuje místa s náhlou změnou úrovně jasové funkce.
2. Hrany se dají detekovat pomocí 1 a 2 derivace. Slouží nám k tomu hranové operátory.
3. Ve frekvenčním spektru odpovídají hrany vysokým frekvencím.

Kapitola 4.

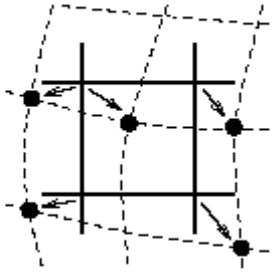
1. Segmentace obrazu je separování oblastí zájmu v obraze od nezajímavého okolí.
2. Mezi metody prahování lze zařadit lokální, globální, dynamické ,adaptivní prahování.
3. Z histogramu jsme schopni zjistit jednotlivé prahovací úrovně obrázku.

Kapitola 5.

1. Statistická filtrace obrazu se provádí na matematickém základě statistického zpracování dat, používají se obecně známé statistické metody a postupy.
2. Při použití metody lokální filtrace se k výpočtu nové hodnoty pixelu využívá malé okolí reprezentativního pixelu (ve smyslu právě zpracovávaného).
3. Obyčejné průměrování je základní metodou vyhlazování obrazu, kde každému bodu přiřadíme nový jas, který je aritmetickým průměrem původních jasů ve zvoleném okolí (např. okolí 3x3 bodů). Nevýhodou praktického použití obyčejného průměrování je rozmazávání hran v obraze. Metoda filtrace mediánem stanoví jas výsledného bodu jako medián určený z hodnot jasu bodů v lokálním okolí (např. 3 x 3 bodů) vstupního obrazu. Metoda redukuje stupeň rozmazání hran a dobře potlačuje impulsní šum.

Kapitola 6.

1. Geometrická transformace upravuje obraz tím, že na základě daných souřadnic ve vstupním obraze vypočte souřadnice ve výstupním obraze. Cílem je geometricky upravit vstupní obraz. V digitálním zpracování obrazu navíc geometrické transformace dovolují odstranit geometrické zkreslení vzniklé při pořízení obrazu (např. korekce geometrických vad objektivu kamery, oprava zkreslení družicového snímku způsobená zakřivením zeměkoule). Mezi základní typy patří rotace, resize a transformace.
2. Při aproximaci jasové funkce se hledá hodnota jasu každého transformovaného bodu. Pomocí metody nejbližšího souseda se přiřadí bodu hodnota jasu nejbližšího bodu v diskrétní mřížce.



Kapitola 7.

1. Pomocí FT převedeme obraz do frekvenční domény a následně jsme schopni s ním pracovat jako s klasickým signálem. Používá se především v problematice aplikace filtrů apod.
2. Dolní propust nám propustí pouze oblasti nízkého kmitočtu, naopak horní propust nám zvýrazní vysoké frekvence.
3. Pomocí filtrace ve frekvenční oblasti jsme schopni na obraz aplikovat filtry, např. pro zvýraznění hran hornopropustný filtr. Prostorová doména slouží zejména pro konvoluci obrazu.

Kapitola 8.

1. Morfologické operace s obrazy slouží nejčastěji k extrakci požadovaných částí obrazu. Jsou založeny na nelineárních operacích v obrazu. V morfologických operacích pracujeme s obrazem A a strukturním elementem B. Strukturní element má význam jako maska u konvoluce, postupně jej přikládáme na jednotlivé pixely obrázku A. Mezi základní morfologické operace patří dilatace, eroze, otevření, zavření.
2. Dilatace – slouží k zaplnění malých mezer, k vyplnění objektu, k zvětšení objektu.
   Eroze – ubírá z objektu, zeštíhluje objekt, eliminuje irelevantní detaily.
3. Obě operace slouží k eliminaci detailů v obraze, zjednodušení struktury apod.

Kapitola 9.

1. Při ztrátové kompresi dochází ke ztrátě nadbytečných dat, které nejsou lidskými smysly postřehnutelné. Bezztrátová komprese je založena na různých algoritmech, které komprimují data tak, aby nedošlo k žádné ztrátě informací.
2. JPEG komprese:

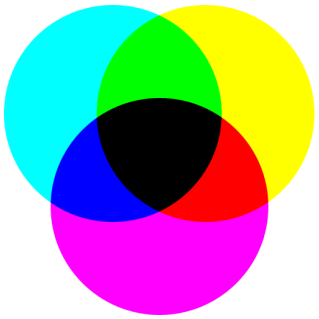
- založena na faktu, že na malé změny barvy je lidské oko méně citlivé než na malé změny jasu
- nevýznamné změny barev jsou odstraňovány, změny jasu jsou naopak s co největší přesností uchovávány
- separace R,G, B složek ze vstupního obrazu. Následná komprese probíhá po složkách.
- rozdělení barevných složek na submatice o rozměrech 8x8. Dále zpracovávány samostatně bez ohledu na „okolní“ submatice
- transformace RGB do YCBCR
- DCT, kódování

3. MPEG komprese:
   - používají se 3 druhy rámců – I, P, B
   - rámce I – kódovaný každý zvlášť pomocí principu JPG
   - na rozdíl od rámců typu I nejsou rámce typu P a B kódovány nezávisle, nýbrž vzhledem k jednomu resp. dvěma jiným referenčním rámcům
   - při kódování se využívá podobnosti rámce s rámci referenčními
   - rámce P (predicted) jsou kódovány vzhledem k jedinému předcházejícímu rámci, kterým mohl být rámec typu I nebo P
   - rámce typu B (interpolated bi-dimensionally) jsou kódovány vzhledem k nejbližšímu předchozímu a nejbližšímu budoucímu rámci typu I nebo P

Kapitola 10.

1. CCD snímače pracují na principu přeměny dopadajícího světelného toku na elektrický náboj.
2. Prokládané snímače - obraz je rozložen do řádků kdy se zpracovávají nejdříve liché řádky a následně sudé. Častěji se využívá u video záznamů, u fotografie je zapotřebí zajistit aby se obraze během zpracování řádků nezměnil. Výhody – ideální pro záznam videosignálů, jednotlivé půlsnímky se zpracovávají separátně, poměrně levná technologie. Nevýhody – obraz je nutné skládat nebo počítat s polovičním rozlišením, pomalé zpracování vylučuje rychlé závěrky, vyžaduje mechanickou závěrku, pro doplnění rozlišení bývají výrobcem užívány interpolační algoritmy. Progresivní snímače - zpracovávají celý obraz najednou. Výhody – přesné zpracování obrazu s minimem zkreslení, vyšší ostrost snímku, krátká časová expozice umožňuje použití elektronické závěrky. Nevýhody – cena a složitost řešení, nevhodné pro videokamery, nerozlišují barvu, ale pracují v RGB režimu.
3. V režimu RGB dochází k aditivnímu (součtovému) míchání barev a), narozdíl od režimu CMY, kdy dochází k subtraktivnímu (odečítacímu) míchání barev b).

a)



b)