



Zemědělská
fakulta
Faculty
of Agriculture

Jihočeská univerzita
v Českých Budějovicích
University of South Bohemia
in České Budějovice

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH ZEMĚDĚLSKÁ FAKULTA

Katedra zemědělské, dopravní a manipulační techniky

Diplomová práce

Vývoj softwaru pro identifikaci střetu ptactva se skleněnou
překážkou

Autor práce: Mgr. Roman Bumbálek

Vedoucí práce: doc. RNDr. Petr Bartoš, Ph.D.

České Budějovice
2021

Prohlášení

Prohlašuji, že jsem autorem této kvalifikační práce a že jsem ji vypracoval pouze s použitím pramenů a literatury uvedených v seznamu použitých zdrojů.

V Českých Budějovicích dne

.....
Podpis

Abstrakt

Diplomová práce je zaměřená na problematiku zpracování obrazu, jejím výstupem je software pro detekci změn v obrazu s implementovanými základními algoritmy této oblasti, který je vytvořen v programovacím jazyku MATLAB. První část orientující se na teoretický úvod do tématu se dělí na tři kapitoly, z nichž první se zabývá lineárními filtry prostorové domény a jejich konvolucí s obrazem. Druhá kapitola stručně popisuje operace ve frekvenční doméně, Fourierovu transformaci a základní frekvenční filtry, a ta třetí se zabývá segmentací obrazu se zaměřením na detekci hran. Druhá část práce zahrnuje představení vývoje programu včetně jeho blokového schématu, pseudokód aplikovaných algoritmů s jeho krátkým popisem a přiblížení tvorby grafického uživatelského prostředí.

Klíčová slova: Zpracování obrazu, konvoluce, lineární filtry, Fourierova transformace, segmentace obrazu, detekce hran,

Abstract

The thesis is focused on the issue of image processing, its output is software for detecting changes in the image with implemented basic algorithms of this area, which is created in the programming language MATLAB. The first part is divided into three chapters and oriented on the theoretical introduction of the topic. The first chapter deals with linear filters of the spatial domain and their convolution with the image. The second chapter briefly describes the operations in the frequency domain, the Fourier transform, and the basic frequency filters, and the third deals with image segmentation with a focus on edge detection. The second part of the thesis includes an introduction to the development of the program, including its block diagram, pseudocode of applied algorithms with its short description, and an approach to the creation of a graphical user environment.

Keywords: Image processing, convolution, linear filters, Fourier transform, image segmentation, edge detection

Poděkování

Rád bych poděkoval své drahé polovičce Míše za velkou podporu a trpělivost, vedoucímu mé diplomové práce panu doc. RNDr. Petru Bartošovi, Ph.D. za jeho cenné rady udělené nejen v rámci studia, jichž si velice vážím, a také kolegům za jejich ochotu poskytnout pomoc při studiu i práci.

Obsah

Úvod.....	7
1 Úpravy obrazu v prostorové doméně	8
1.1 Lineární filtry	8
1.1.1 Konvoluce	8
1.1.2 Filtrace průměrováním	11
1.1.3 Gaussův vyhlazovací filtr.....	14
1.1.4 Laplaceův operátor	18
1.2 Nelineární filtry	22
1.2.1 Mediánový filtr.....	22
2 Úpravy obrazu ve frekvenční doméně	25
2.1 Fourierova transformace.....	25
2.2 Diskrétní Fourierova transformace.....	27
2.2.1 Maticové vyjádření diskretní Fourierovy transformace.....	27
2.2.2 Rychlá Fourierova transformace	29
2.3 Dvourozměrná diskretní Fourierova transformace.....	30
2.3.1 Maticové vyjádření dvoudimenzionální diskretní Fourierovy transformace	32
2.4 Filtrace obrazu ve frekvenční doméně	33
2.4.1 Low-pass filtry	35
2.4.2 High-pass filtry	38
2.4.3 Band-pass filtry	41
3 Segmentace obrazu.....	45
3.1 Detekce hran.....	46
3.2 Hledání hran na základě první derivace	49
3.2.1 Robertsův operátor	50
3.2.2 Operátor Prewittové	51

3.2.3	Sobelův operátor	52
3.2.4	Robinsonův operátor	54
3.2.5	Kirschův operátor.....	55
3.2.6	Cannyho hranový detektor	55
3.3	Hledání hran na základě druhé derivace.....	58
3.3.1	LoG filtr	59
4	Návrh algoritmu a implementace vybraných operátorů.....	62
4.1	Implementace Gaussova filtru	69
4.2	Implementace konvoluce a zrcadlového rozšíření obrazu.....	70
4.3	Implementace Sobelova operátoru a operátoru Prewittové.....	72
4.4	Implementace Robinsonova a Kirschova operátoru.....	73
4.5	Implementace LoG operátoru.....	75
4.6	Implementace Cannyho operátoru.....	77
4.7	Grafické uživatelské prostředí.....	81
4.7.1	Opatření GUI chybovými hláškami	85
5	Výsledky a diskuze	87
	Závěr	98
	Seznam použité literatury.....	99
	Seznam obrázků	105
	Seznam tabulek	109

Úvod

Se vzestupem moderních informačních technologií a rychlým nárůstem dostupného výpočetního výkonu nabývají na významu nové metody zpracování a vyhodnocování dat, z nichž k zásadním se řadí zpracování obrazu a počítačové vidění, které nachází využití v širokém spektru oborů, zvláště v průmyslu (Jha a Swami, 2020; Kurka a Salazar, 2019; Zhu et al., 2021), zdravotnictví (Kuo et al., 2020; Ramani a Shanthamalar, 2020) a geografii (Heras et al., 2019). Doprava je další oblastí, kde narůstá důraz na aplikaci metod počítačového vidění, například při monitoringu dopravy (Jeong et al., 2021; Sangnoree a Chamnongthai, 2017), detekci dopravních značek (Bruno et al., 2012) či u systémů hlídání jízdních pruhů (Chen et al., 2020). Ani zemědělství za ostatními sektory v tomto ohledu nezaostává, zpracování obrazu nalézá uplatnění v živočišné výrobě, kupříkladu pro analýzu chování drůbeže (Li et al., 2020) a diagnostiku zdravotních problémů hospodářských zvířat (Zhao et al., 2018), i rostlinné výrobě, při identifikaci poškození semen (Monteiro et al., 2020) nebo detekci chorob rostlin (Iqbal et al., 2018; Vishnoi et al., 2021).

Problematika zpracování obrazu zahrnuje značné množství metod, které lze kategorizovat dle Cao et al. (2019) jako oblast prostorovou, jejíž základ tvoří filtry s konvolučními maskami, a frekvenční, kde je hlavní podstatou transformace obrazu a následné úpravy pomocí frekvenčních filtrů. Wang et al., (2017) dělí metody do čtyř skupin na předzpracování, segmentaci, extrakci příznaků a klasifikaci.

Cílem diplomové práce je návrh a realizace softwaru pro identifikaci střetu ptactva se skleněnou překážkou za využití detekce změn v obrazu, implementace vhodných algoritmů umožňujících optimální zpracování nasnímaných dat v rámci projektu TRIO FV30234.

1 Úpravy obrazu v prostorové doméně

1.1 Lineární filtry

Ke zvýšení kvality obrazu, jeho vylepšení, je možné využít lineárních filtrů umožňujících potlačení obrazového šumu či zvýraznění charakteristických příznaků, které jsou modifikací hodnot obrazových bodů realizovanou přímou manipulací s nimi v prostorové doméně (Yoo, 2004). Hlavní podstatou aplikace lineárních filtrů je operace konvoluce, kdy novou hodnotu každého pixelu vstupního obrazu získáme výpočtem na základě hodnot pixelů v sousední oblasti původního pixelu a koeficientů kernelu daného operátoru, což je obvykle čtvercová matice hodnot s lichým počtem řádků a sloupců (Forsyth a Ponce, 2012). V literatuře jsou filtry kategorizovány dle využití do tří skupin. První skupinu tvoří lineární vyhlazovací filtry, které slouží k rozmazání a potlačení šumu, kdy dochází ke snížení ostrých přechodů v úrovních jasu a drobných detailů. K hlavním zástupcům patří filtrace průměrováním a Gaussův filtr (Jain et al., 1995; Petrou a Petrou, 2010). Do druhé skupiny se řadí filtry zostrující obraz, jejichž funkce spočívá ve zvýraznění drobných detailů či obnovení rozmazaných prvků, tuto skupinu reprezentuje Laplaceův operátor, který bývá řazen i do skupiny poslední, tedy mezi filtry hranové označované také jako hranové detektory (Gonzales a Woods, 2002; Russ, 2002), jenž jsou dále rozpracovány v části práce zabývající se segmentací obrazu.

1.1.1 Konvoluce

Diskrétní konvoluce je operace používaná pro aplikaci lineárních filtrů na vstupní obraz, při níž je vypočítána hodnota nového pixelu náležícího výstupnímu obrazu z hodnot sousedních pixelů (Forsyth a Ponce, 2012). Při výpočtu dochází k násobení vybraného pixelu a pixelů v jeho okolí, které je určeno velikostí konvolučního kernelu filtru, označovaného také jako maska filtru či konvoluční maska, s váženými koeficienty kernelu a následně k součtu všech součinů (Nixon a Aguado, 2002). V případě aplikace filtru o velikosti $m \times n$, kdy $m = 2i + 1$ a $n = 2j + 1$, na obraz o velikosti $M \times N$, kdy $x = 0, 1, 2, \dots, M - 1$ a $y = 0, 1, 2, \dots, N - 1$ můžeme konvoluci zapsat ve tvaru

$$h(x, y) = \sum_{k=-i}^i \sum_{l=-j}^j g(k, l) f(x + k, y + l), \quad (1.1)$$

kde $g(k, l)$ je konvoluční kernel, $f(x, y)$ vstupní obraz a $h(x, y)$ výstupní obraz, tedy výsledek konvoluce (Gonzales a Woods, 2002; Sojka et al., 2011). Operace konvoluce je zapisována pomocí znaku $*$, lze ji tedy také vyjádřit zápisem 1.2 nebo zápisem 1.3.

$$h(x, y) = f(x, y) * g(x, y) \quad (1.2)$$

$$h(x, y) = (f * g)(x, y) \quad (1.3)$$

Dle Yoo (2004) má diskrétní konvoluce, stejně jako konvoluce spojitých signálů, nezávisle na hodnotách kernelu mnoho důležitých vlastností, díky nimž je konvoluce považována za hlavní operaci pro lineární filtrování. K nejdůležitějším vlastnostem zahrnuje:

- linearitu

$$\begin{aligned} (Af_1(x, y) + Bf_2(x, y)) * f_3(x, y) &= \\ = A(f_1(x, y) * f_3(x, y)) + B(f_2(x, y) * f_3(x, y)) \end{aligned} \quad (1.4)$$

- komutativnost

$$f_1(x, y) * f_2(x, y) = f_2(x, y) * f_1(x, y) \quad (1.5)$$

- asociativnost

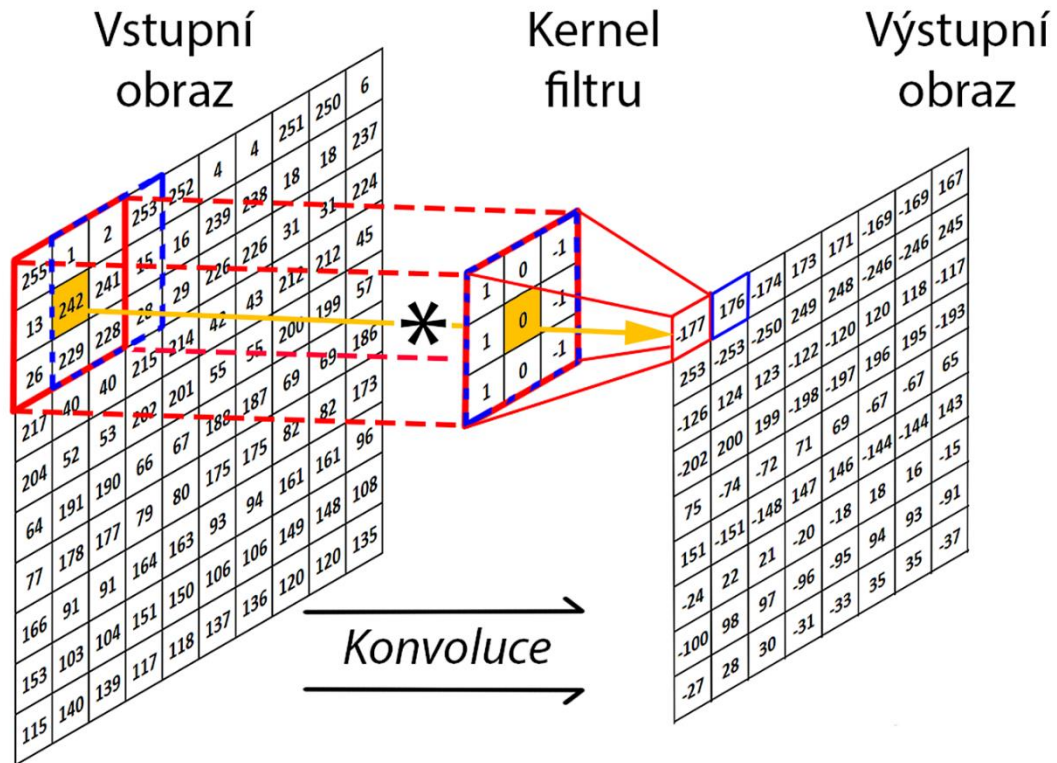
$$(f_1(x, y) * f_2(x, y)) * f_3(x, y) = f_1(x, y) * (f_2(x, y) * f_3(x, y)) \quad (1.6)$$

- distributivnost

$$\begin{aligned} f_1(x, y) * (f_2(x, y) + f_3(x, y)) &= \\ = f_1(x, y) * f_2(x, y) + f_1(x, y) * f_3(x, y) \end{aligned} \quad (1.7)$$

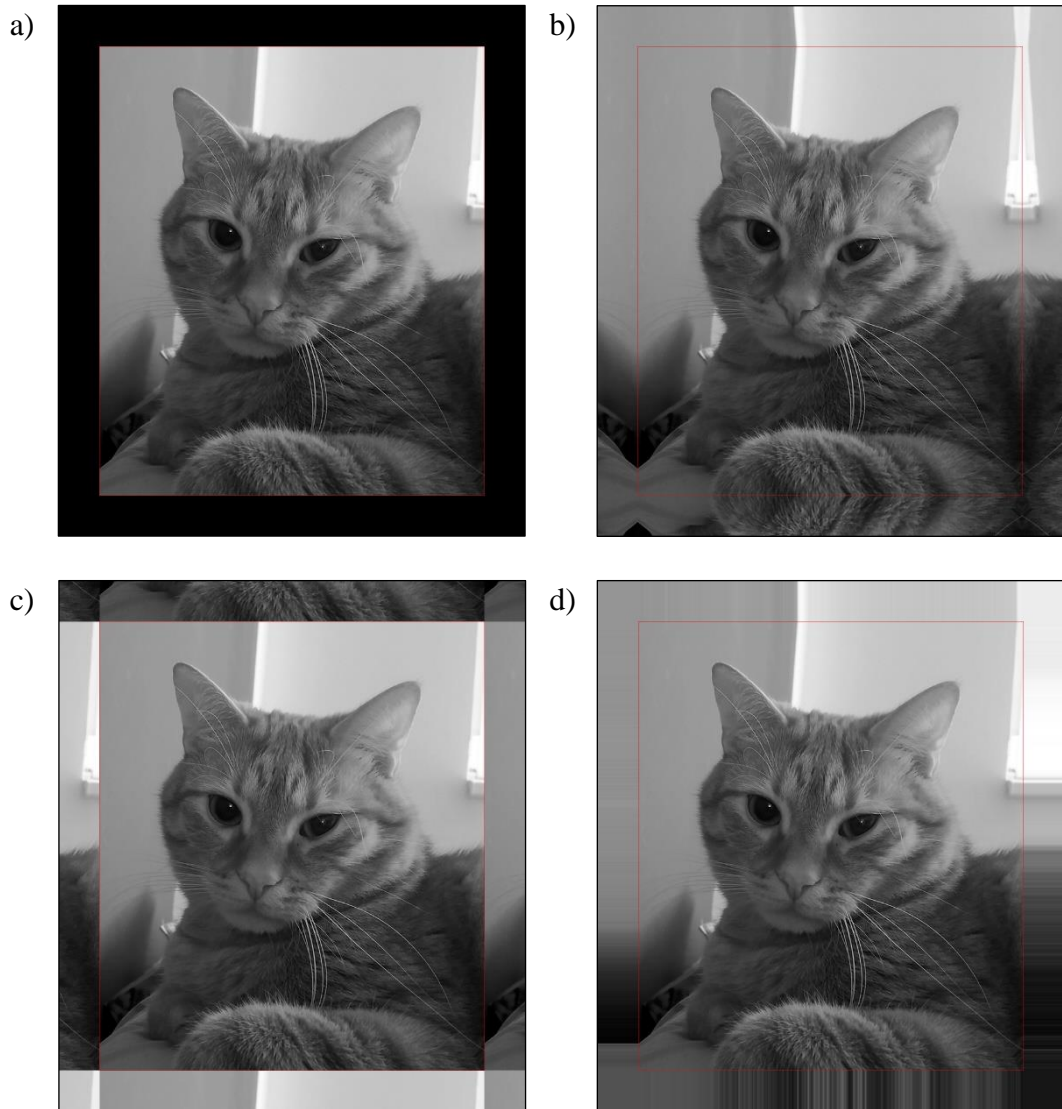
Základním principem aplikace konvoluce je provedení této operace nad každým pixellem obrazu tak, že poloha centrálního členu kernelu odpovídá poloze pixelu, pro který je počítána nová hodnota, jak je znázorněno na obrázku 1. Problém však může nastat v případě krajních pixelů obrazu, případně pixelů ležících ve vzdálenosti menší než $(2m - 1)/2$ od vertikálních krajů obrazu nebo ve vzdálenosti $(2n - 1)/2$ od horizontálních krajů, kdy část koeficientů kernelu svou polohou přesahuje hranice obrazu. Tuto překážku lze řešit několika způsoby, z nichž nejjednodušší je omezení posunu kernelu do výše zmíněných oblastí krajů. Touto podmínkou je zajištěno, že všechny pixely výsledného obrazu vzniknou za působení celého kernelu, ale je zde nevýhoda v podobě menších rozměrů výsledného obrazu, původní rozlišení $M \times N$

je sníženo na $(M - 2m + 1) \times (N - 2n + 1)$, což není při využití kernelů malých rozměrů příliš důležité, avšak s velkými kernely dochází již k významnější ztrátě obrazových bodů (Gonzales a Woods, 2002).



Obrázek 1.1: Znázornění průběhu konvoluce

Pro zachování rozměrů je možné použít metodu zvanou padding, jedná se o vytvoření barevného rámu, nejčastěji černého, bílého či šedého, kolem původního obrazu o šířce $(2m - 1)/2$ a $(2n - 1)/2$, nebo zrcadlové prodloužení obrazu, kdy je okolí obrazu vyplněno pixely zrcadlenými dle os s polohou v okrajích obrazu. K dalším metodám patří periodické prodloužení obrazu, v rámci kterého jsou za hodnoty pixelů stanoveného rámu obrazu doplňovány hodnoty pixelů obrazu nacházejících se u opačného kraje obrazu tak, že vzdálenost mezi pixelem obrazu a pixelem rámu se stejnou hodnotou, je vždy rovna šířce obrazu, pokud se jedná o pixely u vertikálního kraje obrazu či výšce v případě horizontálního kraje obrazu. K doplnění hodnot pixelů do okolí obrazu lze také aplikovat replikaci hodnot krajních pixelů (Russ a Russ, 2008).



Obrázek 1.2: Rozšíření vstupního obrazu a) vytvořením černého rámu, b) zrcadlovým prodloužením, c) periodickým prodloužením obrazu, d) replikací hodnot krajních pixelů

1.1.2 Filtrace průměrováním

Jedním z nejjednodušších způsobů lineární filtrace šumu je nahrazení každého pixelu obrazu aritmetickým průměrem hodnot bodů z jeho okolní oblasti. Filtrace průměrováním, také označována jako uniformní filtr, je realizována konvolucí obrazu s maskou filtru, kdy jsou všechny její hodnoty stejné a jejich součet je roven jedné, jak je znázorněno na obrázku 1.3 (Padmavathi et al, 2009). V tomto případě lze upravit předpis a snížit výpočetní náročnost konvoluce, jelikož hodnota všech členů kernelu $g(x, y)$ o velikosti $m \times n$ je stejná, tedy

$$g_{0,0} = g_{0,1} = \dots = g_{m-1,n-1} = \frac{1}{mn}, \quad (1.8)$$

a tak platí vztah

$$h(x, y) = \frac{1}{mn} \sum_{k=-i}^i \sum_{l=-j}^j f(x + k, y + l). \quad (1.9)$$

Pomocí výše zmíněné úpravy se výpočetní náročnost aplikace filtru na obraz velikosti $M \times N$ snižuje o $MN(mn - 1)$ součinů (Jain et al., 1995). Účinnost filtru vzhledem k potlačení šumu roste se zvyšující se velikostí konvolučního kernelu, avšak s tím je i přímo spojeno snížení úrovně detailů (Nixon a Aguado, 2002).

a)	<table style="border-collapse: collapse; text-align: center;"> <tr><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td><td style="border: 1px solid black; padding: 5px;">1/9</td></tr> </table>	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9																
1/9	1/9	1/9																								
1/9	1/9	1/9																								
1/9	1/9	1/9																								
b)	<table style="border-collapse: collapse; text-align: center;"> <tr><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td></tr> <tr><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td><td style="border: 1px solid black; padding: 5px;">1/25</td></tr> </table>	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25																						
1/25	1/25	1/25	1/25	1/25																						
1/25	1/25	1/25	1/25	1/25																						
1/25	1/25	1/25	1/25	1/25																						
1/25	1/25	1/25	1/25	1/25																						

Obrázek 1.3: Konvoluční masky filtru o velikosti a) 3×3 a b) 5×5

Gonzales a Woods (2002) uvádí filtraci pomocí váženého průměru, při které jsou okolní pixely násobeny rozdílnými koeficienty, což dává některým pixelům větší váhu než ostatním. Typicky bývá největší koeficient umístěn ve středu kernelu, tedy pixel s odpovídající hodnotou vůči středu kernelu je násoben nejvyšší hodnotou a stává se bodem s největším vlivem na výpočet průměru. Vliv bodů v okolí centrálního pixelu klesá s jejich vzdáleností od středu, přičemž základní myšlenkou je snížit účinky rozmazání při vyhlazování obrazu. Součet všech koeficientů kernelu musí být rovný nule, vytvořený kernel je třeba normalizovat, tedy každý jeho koeficient vydělit sumou všech koeficientů nebo sumou vydělit hodnotu pixelu vzniklého konvolucí s vytvořeným kernelem, z čehož vychází vztah

$$h(x, y) = \frac{1}{\sum_{k=-i}^i \sum_{l=-j}^j g(k, l)} \sum_{k=-i}^i \sum_{l=-j}^j g(k, l) f(x + k, y + l). \quad (1.10)$$

V literatuře se liší forma interpretace konvolučního kernelu, například Jain et al. (1995), obr. 1.4a), narozdíl od Gonzales a Woods (2002), obr. 1.4b), uvádí zápis již normalizovaného filtru.

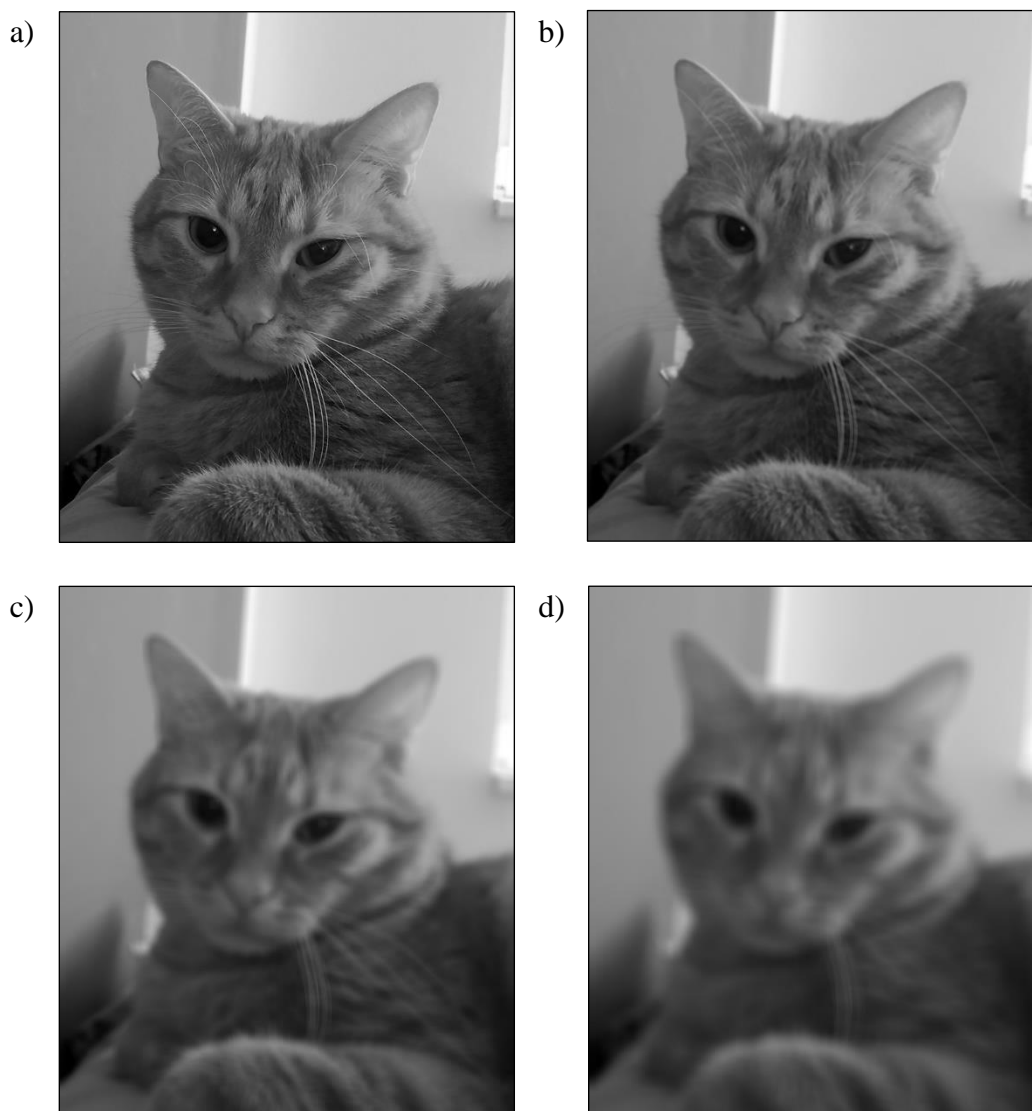
a)

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

b)

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Obrázek 1.4: Konvoluční masky filtru váženého průměru, a) zápis dle Jain et al. (1995), b) zápis dle Gonzales a Woods (2002)



Obrázek 1.5: Obraz po aplikaci filtrace průměrováním a) vstupní obraz bez vyhlazení, b) kernel o velikosti 5×5, c) kernel 15×15, d) kernel 25×25

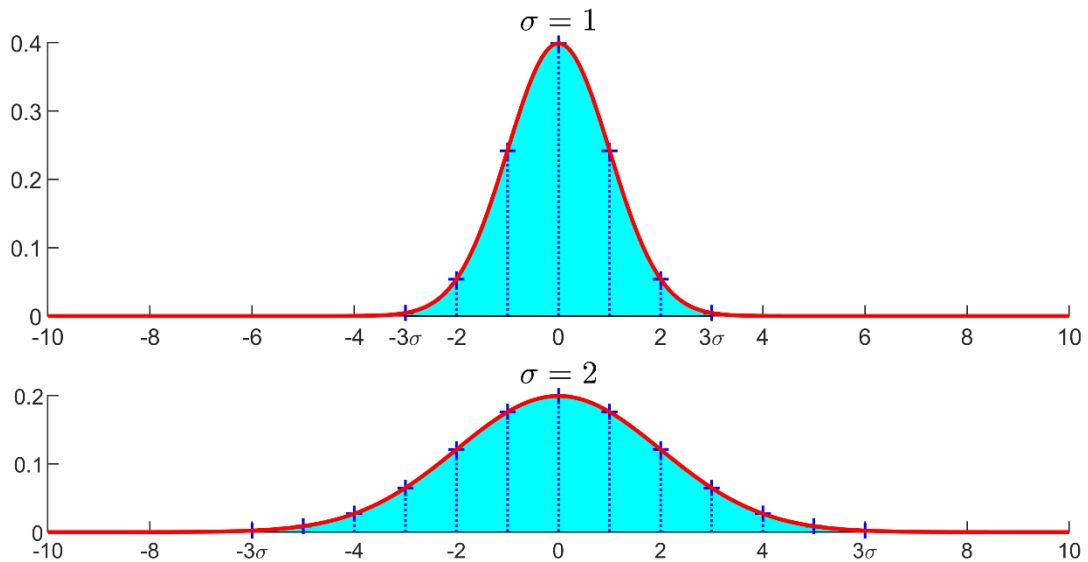
1.1.3 Gaussův vyhlazovací filtr

Gaussovo rozostření je izotropní operací sloužící k odstranění extrémních hodnot bodů v obrazu projevujících se jako šum a vychází z Gaussova pravděpodobnostního rozdělení, uváděného též jako normální rozdělení (Pathmanabhan a Dinesh, 2007). Základem je Gaussova funkce s předpisem

$$g(x) = e^{-\frac{x^2}{2\sigma^2}} \quad (1.11)$$

ze které je odvozen předpis jednodimenziálního Gaussova rozdělení hustoty pravděpodobnosti s rozptylem σ^2 a střední hodnotou v bodě $x = 0$

$$h(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \quad (1.12)$$



Obrázek 1.6: Grafy Gaussova normálního rozdělení pro σ o velikosti 1 a 2

Flusser et al. (2016) uvádí, že hodnoty Gaussova kernelu jsou dopočítávány pomocí funkce dvoudimenziálního Gaussova rozdělení, která je definována jako

$$G(x, y) = h(x)h(y) \quad (1.13)$$

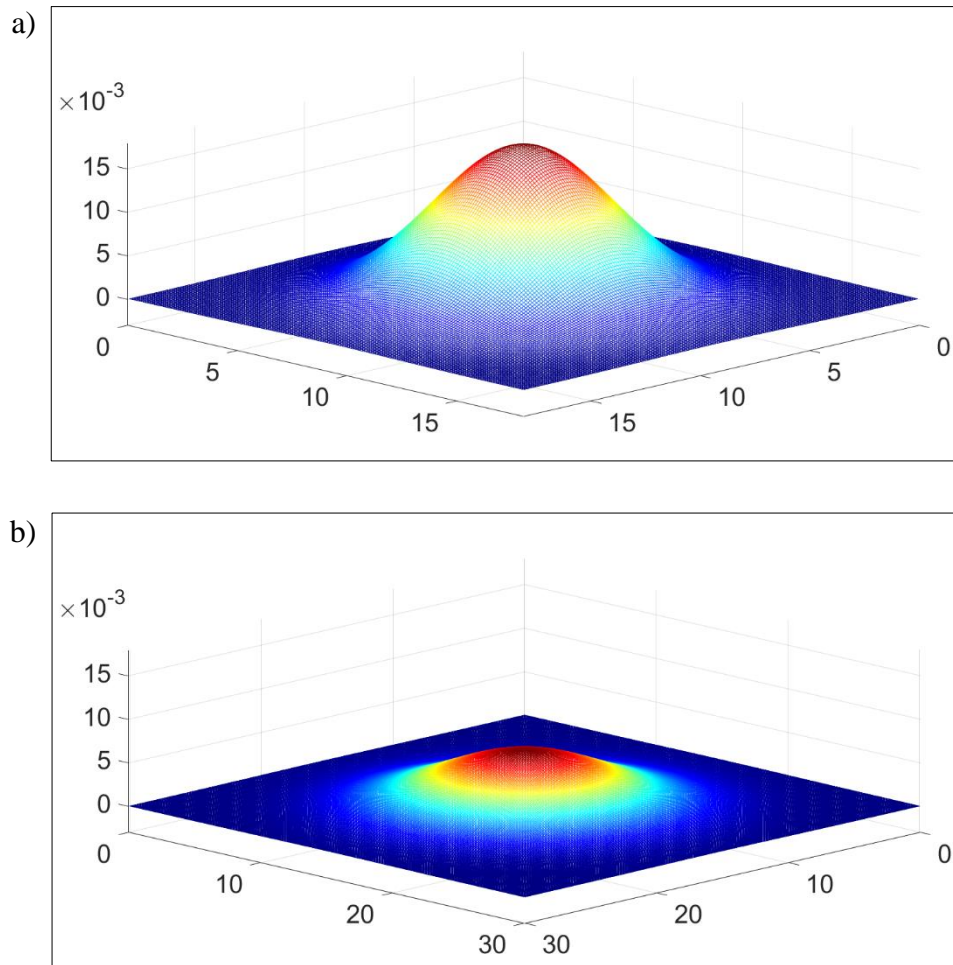
kde $\sigma > 0$ a $h(x), h(y)$ jsou funkce Gaussova rozdělení, kdy po úpravě vznikne vztah

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad (1.14)$$

který je zmíněn také v Pathmanabhan a Dinesh (2007), Sojka et. al. (2011), Yoo (2004).

Dle Jain et al. (1995) má Gaussova funkce pět základních vlastností, které ukazují, že filtr Gaussova rozostření je účinným nástrojem k potlačení šumu v prostorové i frekvenční doméně a tak může být efektivně využit při předzpracování obrazu. K těmto vlastnostem se řadí:

- 1) Rotační symetrie Gaussovi funkce ve dvou dimenzích. Tato vlastnost ukazuje, že rozostření provedené tímto filtrem je ve všech směrech stejné, tedy není třeba filtr aplikovat ve více směrech.
- 2) Existence pouze jednoho lokálního maxima, které je zároveň ostrým globálním maximem. Rozostření Gaussovým filtrem nahradí každý pixel obrazu váženým průměrem sousedních pixelů tak, že váha daná sousedním pixelům monotónně klesá s rostoucí vzdáleností od centrálního pixelu, díky tomu nedochází k výraznému narušení hran.
- 3) Velikost Gaussova filtru přímo souvisejí s hodnotou σ , s jejíž zvýšením se zvětšuje i velikost filtru a zároveň roste stupeň rozostření.
- 4) Separabilita Gaussovy funkce, umožňující účinnou implementaci Gaussových filtrů velkých rozměrů, protože dvoudimenzionální konvoluce může být nahrazena postupnou konvolucí jednodimenzionálního Gaussiánu v horizontálním a vertikálním směru s obrazem, čímž potřebný výpočetní výkon roste vzhledem k velikosti filtru lineárně, nikoliv kvadraticky jako v případě 2D konvoluce.
- 5) Existence pouze jednoho lokálního maxima, které je zároveň ostrým globálním maximem i po převodu do frekvenční oblasti pomocí Fourierovy transformace. Po aplikaci Gaussova filtru ve frekvenční doméně dochází k eliminaci nežádoucích vysokofrekvenčních signálů, jako je obrazový šum, se zachováním množství žádoucích příznaků skládajících se z částí z oblasti vysokých i nízkých frekvencí, což jsou například hrany.



Obrázek 1.7: Graf kernelu Gaussova filtru, kdy a) $\sigma=3$, b) $\sigma=5$

Velikost kernelu Gaussovy funkce lze vyjádřit jako $(2k + 1) \times (2k + 1)$, kernel tedy vždy obsahuje lichý počet prvků v horizontálním i vertikálním uspořádání (Forsyth a Ponce, 2012), kdy je optimální zvolit $k = \lceil 3\sigma \rceil$, protože v intervalu $\langle -3\sigma; 3\sigma \rangle$ je zastoupeno 99,7 % hodnot koeficientů kernelu, což vyplývá také z jednodimenzionálního Gaussova rozdělení, kdy je ve zmíněném intervalu pod křivkou plocha odpovídající 99,7 % celé plochy pod křivkou funkce (Russ, 2002).

Dle Petrou a Petrou (2010) a Dobeš (2008) je třeba vytvořený kernel normalizovat, aby součet jeho hodnot byl roven jedné, z důvodu zachování hodnot plochého pole, v důsledku jde o dodržení mezních hodnot u bodů vzniklých konvolucí kernelu s obrazem. Normalizované hodnoty Gaussova kernelu lze vypočítat vydělením každého prvku kernelu součtem všech jeho prvků dle vztahu (1.15), jehož úpravou je vyjádřen vzorec (1.18) pro přímý výpočet normalizovaných hodnot.

$$G_n(x, y) = \frac{G(x, y)}{\sum G(x, y)} \quad (1.15)$$

$$G_n(x, y) = \frac{\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}}{\sum_{x=-k}^k \sum_{y=-k}^k \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}} \quad (1.16)$$

$$G_n(x, y) = \frac{\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}}{\frac{1}{2\pi\sigma^2} \sum_{x=-k}^k \sum_{y=-k}^k e^{-\frac{x^2+y^2}{2\sigma^2}}} \quad (1.17)$$

$$G_n(x, y) = \frac{e^{-\frac{x^2+y^2}{2\sigma^2}}}{\sum_{x=-k}^k \sum_{y=-k}^k e^{-\frac{x^2+y^2}{2\sigma^2}}} \quad (1.18)$$

Někteří autoři, jako Nixon a Aguado (2002) či Petrou a Petrou (2010), vypočítávají hodnoty kernelu ze vzorce 1.13, kam však místo funkce Gaussova rozdělení dosazují Gaussovu funkci, jak je uvedeno v rovnici 1.19

$$G(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad (1.19)$$

hodnoty kernelu se poté liší o $1/2\pi\sigma^2$, avšak po aplikaci normalizace jsou hodnoty shodné s hodnotami po normalizování kernelu vypočítaném z rovnice 1.14. Tato skutečnost také vyplývá z rovnice 1.18.

a)	b)
$\begin{bmatrix} 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \\ 0,0000 & 0,0117 & 0,0862 & 0,0117 & 0,0000 \\ 0,0002 & 0,0862 & 0,6366 & 0,0862 & 0,0002 \\ 0,0000 & 0,0117 & 0,0862 & 0,0117 & 0,0000 \\ 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \end{bmatrix}$	$\begin{bmatrix} 0,0000 & 0,0000 & 0,0003 & 0,0000 & 0,0000 \\ 0,0000 & 0,0183 & 0,1353 & 0,0183 & 0,0000 \\ 0,0003 & 0,1353 & 1,0000 & 0,1353 & 0,0003 \\ 0,0000 & 0,0183 & 0,1353 & 0,0183 & 0,0000 \\ 0,0000 & 0,0000 & 0,0003 & 0,0000 & 0,0000 \end{bmatrix}$
c)	d)
$\begin{bmatrix} 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \\ 0,0000 & 0,0113 & 0,0837 & 0,0113 & 0,0000 \\ 0,0002 & 0,0837 & 0,6182 & 0,0837 & 0,0002 \\ 0,0000 & 0,0113 & 0,0837 & 0,0113 & 0,0000 \\ 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \end{bmatrix}$	$\begin{bmatrix} 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \\ 0,0000 & 0,0113 & 0,0837 & 0,0113 & 0,0000 \\ 0,0002 & 0,0837 & 0,6182 & 0,0837 & 0,0002 \\ 0,0000 & 0,0113 & 0,0837 & 0,0113 & 0,0000 \\ 0,0000 & 0,0000 & 0,0002 & 0,0000 & 0,0000 \end{bmatrix}$

Obrázek 1.8: Hodnoty kernelu Gaussova filtru pro $\sigma=0,5$ a) vypočítané dle vztahu 1.14, b) vypočítané dle vztahu 1.19, c) hodnoty kernelu a) po jeho normalizování, d) hodnoty kernelu b) po jeho normalizování



Obrázek 1.9: Obraz po aplikaci Gaussova filtru a) vstupní obraz bez vyhlazení, b) $\sigma=0,5$, c) $\sigma=5$, d) $\sigma=9$

1.1.4 Laplaceův operátor

Laplaceův operátor je izotropní lineární operátor vycházející z druhých parciálních derivací hodnot obrazu daného funkcí $f(x, y)$, který je definován vztahem

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (1.20)$$

Dle metody konečných diferencí můžeme derivaci jednodimenzionální funkce $f(x)$ v bodě x aproximovat s využitím dopředné difference pomocí vztahu

$$\frac{\partial f}{\partial x} \approx \frac{f(x+h) - f(x)}{h}, \quad (1.21)$$

kde $h > 0$. Pro aproximaci v blízkém okolí vybraného bodu zvolíme $h = 1$, tedy platí

$$\frac{\partial f}{\partial x} = f'(x) \approx f(x+1) - f(x). \quad (1.22)$$

Aproximace druhé derivace definujeme jako

$$\frac{\partial^2 f}{\partial x^2} = f''(x) \approx f'(x+1) - f'(x), \quad (1.23)$$

po dosazení rovnice 1.22 do vztahu 1.23 získáme

$$\frac{\partial^2 f}{\partial x^2} \approx (f(x+2) - f(x+1)) - (f(x+1) - f(x)) \quad (1.24)$$

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+2) - 2f(x+1) + f(x). \quad (1.25)$$

Aproximovat druhou derivaci dvoudimenzionální funkci $f(x, y)$ dle x lze analogicky dle vztahu 1.25

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+2, y) - 2f(x+1, y) + f(x, y), \quad (1.26)$$

z čehož vyplývá, že je aproximace centrována kolem bodu $[x+1, y]$, nahrazením $x-1$ za x získáme

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+1, y) - 2f(x, y) + f(x-1, y), \quad (1.27)$$

kdy je aproximace již soustředěná kolem bodu $[x, y]$. Stejným způsobem získáme aproximaci druhé derivace funkce dle y , tedy

$$\frac{\partial^2 f}{\partial y^2} \approx f(x, y+1) - 2f(x, y) + f(x, y-1). \quad (1.28)$$

Aproximace Laplaceova operátoru může být tedy vyjádřena dosazením rovnic 1.27 a 1.28 do rovnice 1.20 zápisem

$$\begin{aligned} \nabla^2 f \approx & f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) \\ & - 4f(x, y), \end{aligned} \quad (1.29)$$

který pro větší názornost můžeme doplnit o další členy tak, aby bylo doplněno čtvercové okolí bodu $[x, y]$.

$$\begin{aligned}
\nabla^2 f \approx & 0f(x-1, y+1) + 1f(x, y+1) + 0f(x+1, y+1) \\
& + 1f(x-1, y) - 4f(x, y) + 1f(x+1, y) \\
& + 0f(x-1, y-1) + 1f(x, y-1) \\
& + 0f(x+1, y-1).
\end{aligned} \tag{1.30}$$

Z této rovnice vychází konvoluční maska, znázorněná na obrázku 1.10a), s níž lze Laplaceův operátor na obraz aplikovat (Gonzales a Woods, 2002; Jain et al., 1995; Nixon a Aguado, 2002). V literatuře je možné se také setkat s maskou o opačných hodnotách (obr. 1.10b)), které jsou získány použitím zpětné diference, či s rozšířenou konvoluční maskou pro osmiokolí, viz obr. 1.10c) (Russ, 2002).

a)	<table border="1" style="border-collapse: collapse; text-align: center;"><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>-4</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr></table>	0	1	0	1	-4	1	0	1	0
0	1	0								
1	-4	1								
0	1	0								

b)	<table border="1" style="border-collapse: collapse; text-align: center;"><tr><td>0</td><td>-1</td><td>0</td></tr><tr><td>-1</td><td>4</td><td>-1</td></tr><tr><td>0</td><td>-1</td><td>0</td></tr></table>	0	-1	0	-1	4	-1	0	-1	0
0	-1	0								
-1	4	-1								
0	-1	0								

c)	<table border="1" style="border-collapse: collapse; text-align: center;"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>-8</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	-8	1	1	1	1
1	1	1								
1	-8	1								
1	1	1								

Obrázek 1.10: Hodnoty kernelu Laplaceova operátoru pro a) základní kernel pro čtyřokolí, b) opačné hodnoty základního kernelu, c) rozšířený kernel pro osmiokolí

Po konvoluci vznikne obraz se zvýrazněnými hranami a nespojitými body v odstínech šedé a nevýrazným tmavým pozadím s minimem detailů. Spojením původního a upraveného obrazu, tedy součtem jejich hodnot, pokud je koeficient ve středu masky kladný, či rozdílem v případě záporné hodnoty centrálního koeficientu, dojde k navrácení detailů pozadí a zachování efektu zостření získaného aplikací Laplaceova operátoru (Russ a Russ, 2008).

Výše zmíněný postup lze následující úpravou zjednodušit. Spojení obrazu lze vyjádřit vztahem

$$I(x, y) = f(x, y) - f(x, y) * g(x, y), \tag{1.31}$$

kde za konvoluci obrazu s operátorem dosadíme rovnici 1.29

$$\begin{aligned}
I(x, y) &= f(x, y) - [f(x+1, y) + f(x-1, y) + \\
&+ f(x, y+1) + f(x, y-1) - 4f(x, y)] \\
&= 5f(x, y) - f(x+1, y) - f(x-1, y) - \\
&- f(x, y+1) - f(x, y-1),
\end{aligned} \tag{1.32}$$

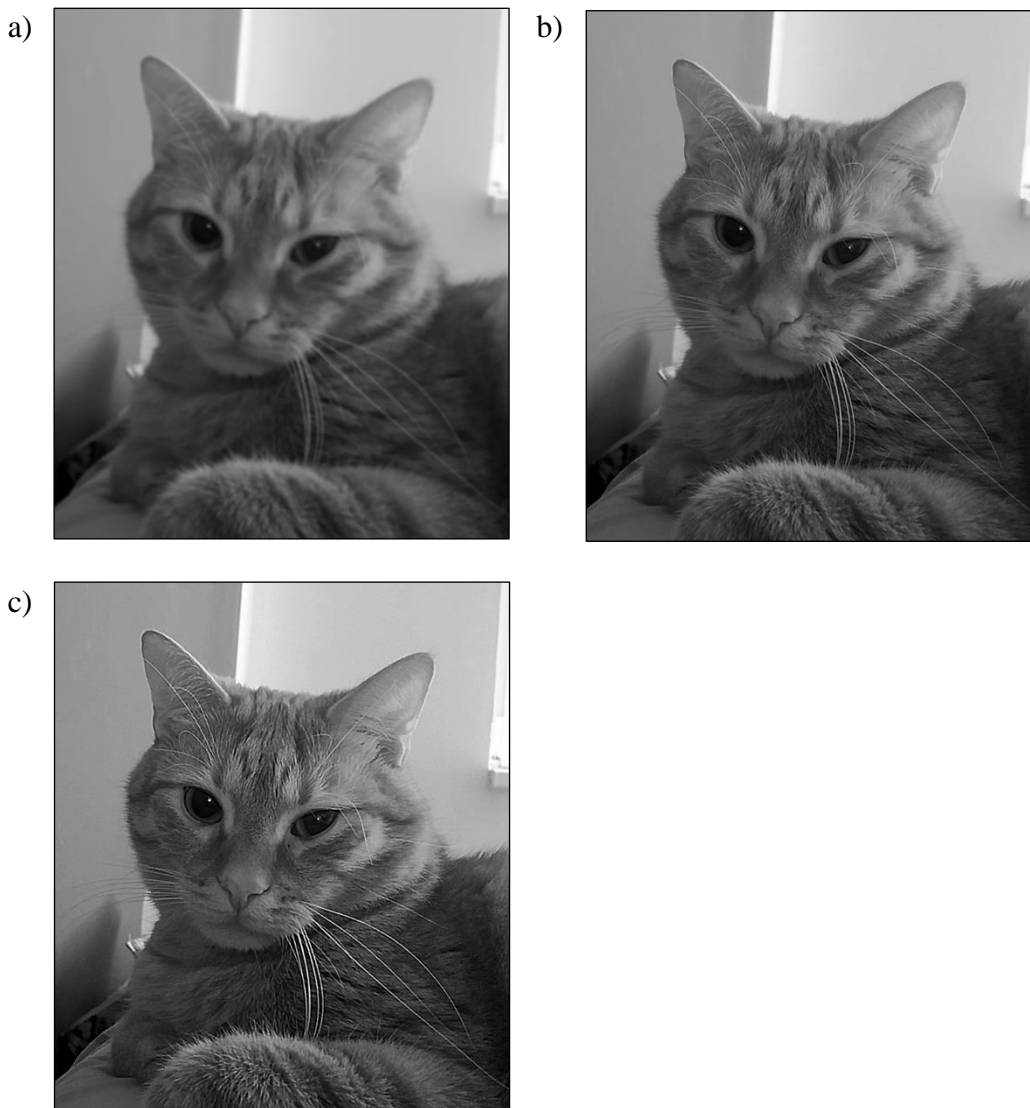
dostaneme upravený předpis, ze kterého můžeme odvodit složenou konvoluční masku (obr. 1.11a)), analogicky bude upravena i maska pro osmiokolí (obr. 1.11b)), po jejich

aplikaci je vytvořen zostřený obraz, jenž není třeba sloučit s původním obrazem (Gonzales a Woods, 2002).

a)	<table border="1"><tr><td>0</td><td>-1</td><td>0</td></tr><tr><td>-1</td><td>5</td><td>-1</td></tr><tr><td>0</td><td>-1</td><td>0</td></tr></table>	0	-1	0	-1	5	-1	0	-1	0
0	-1	0								
-1	5	-1								
0	-1	0								

b)	<table border="1"><tr><td>-1</td><td>-1</td><td>-1</td></tr><tr><td>-1</td><td>9</td><td>-1</td></tr><tr><td>-1</td><td>-1</td><td>-1</td></tr></table>	-1	-1	-1	-1	9	-1	-1	-1	-1
-1	-1	-1								
-1	9	-1								
-1	-1	-1								

Obrázek 1.11: Upravená konvoluční maska Laplaceova operátoru pro a) čtyřokolí, b) osmiokolí



Obrázek 1.12: Obraz po aplikaci Laplaceova operátoru zostření, a) vstupní obraz bez vyhlazení, b) upravená konvoluční maska pro čtyřokolí, c) upravená konvoluční maska pro osmiokolí

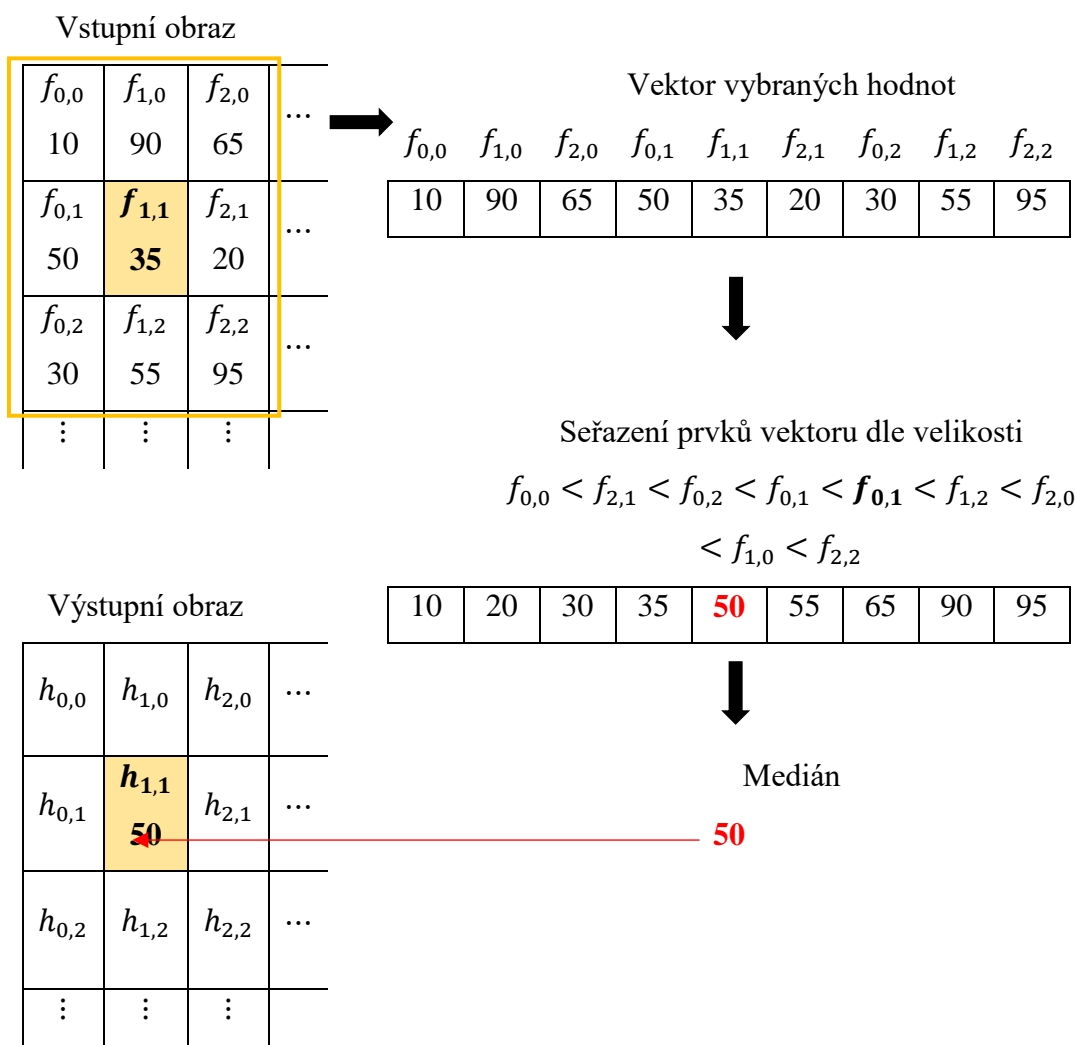
1.2 Nelineární filtry

Nelineární filtrování je alternativním přístupem k potlačení obrazového šumu při zachování hran, který má základ ve statistických metodách (Gonzales a Woods, 2009). Na rozdíl od lineárních filtrů nemají tyto filtry ekvivalent ve Fourierově frekvenční doméně (Russ, 2002). Gasterados a Andreadis (2000) rozdělují tyto operace na dvě skupiny, filtry na základě pořadí a morfologické filtry. První skupina se vyznačuje velmi dobrou robustností a zajišťuje dobré výsledky v mnoha případech, kdy lineární filtry nevykazují optimální účinnost. K těmto filtrům se řadí modální filtr, který z vybrané oblasti vybere hodnotu v ní nejčastěji se vyskytující (Petrou a Petrou, 2010), filtr typu minimum a maximum nahrazující hodnotu vybraného pixelu minimální či maximální hodnotou z jeho okolí, Olympijský filtr, kdy jsou odstraněny nejvyšší a nejnižší hodnoty z oblasti sousedních pixelů a poté průměrovány zbylé hodnoty bodů (Jensen, 2015) a mediánový filtr (Win et al., 2019), který je podrobněji popsán v následující kapitole. Morfologické filtry zastupují například operace otevření, uzavření či skeletonizace (Smith, 1999).

1.2.1 Mediánový filtr

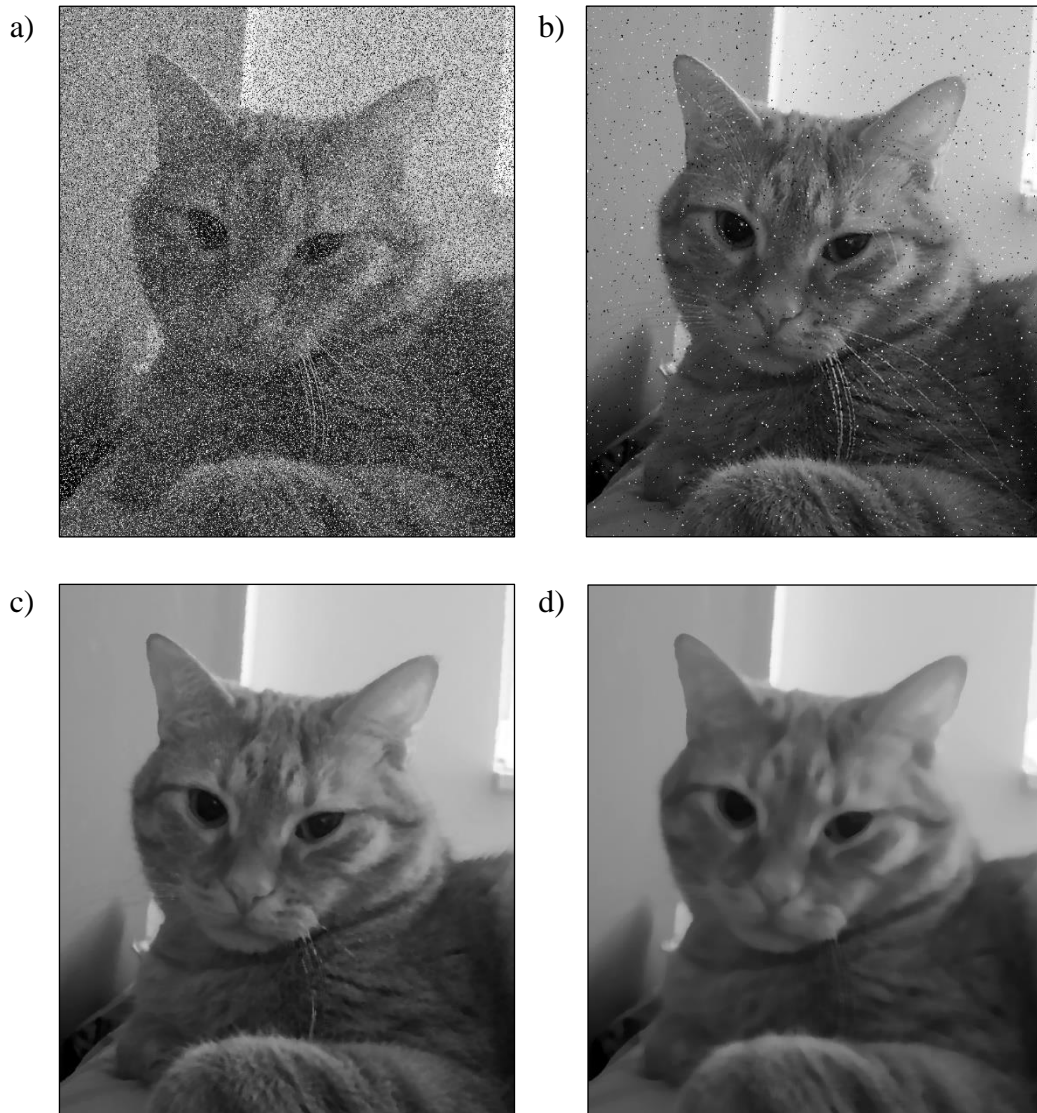
Mediánový filtr je nejčastěji využívaným statistickým filtrem založeným na pořadí hodnot vybrané oblasti. Činnost filtru přímo souvisí s mediánem, který vyjadřuje hodnotu rozdělující skupinu prvků seřazených dle velikosti jejich hodnot na dvě části o stejném množství prvků. V případě skupiny o počtu prvků $2k + 1$, který při aplikaci filtru zkoumaná oblast nabývá, se hodnota mediánu nachází na místě $k + 1$ (Petrou a Petrou, 2010).

Použití filtru spočívá v nahrazení hodnoty vybraného pixelu mediánem hodnot z jeho okolí, což je aplikováno pro každý pixel obrazu pomocí pomyslné masky filtru. Nejprve dochází k výběru sousedních pixelů vybraného obrazového bodu, tedy pixelů z okolí o velikosti $n \times m$, poté je z nich vytvořen vektor, jehož prvky jsou následně seřazeny dle hodnoty od nejmenší po největší. Dále je vybrána hodnota nacházející se ve středu vektoru, obsahující seřazené prvky, což je medián. Nakonec je hodnota vybraného pixelu nahrazena hodnotou mediánu, tedy vložena do výstupního obrazu, a maska výběru se posouvá k dalšímu pixelu, celý proces je opakován pro všechny body obrazu (Russ a Russ, 2008).



Obrázek 1.13: Schématické znázornění základního principu mediánového filtru

Hlavní myšlenkou mediánového filtru je upravit hodnotu pixelů s výrazně odlišnou intenzitou vůči okolním pixelům tak, aby velké odchylky hodnot byly eliminovány. V obrazu se mohou vyskytovat izolované shluky bodů o malé ploše, které jsou velice světlé nebo tmavé v porovnání se sousedními body, na které má mediánový filtr značný vliv, větší shluky jsou ovlivněny tímto filtrem podstatně méně (Win et al., 2019). Velmi dobrý účinek má mediánový filtr na potlačení impulsního šumu označovaného také jako šum typu sůl a pepř (Dobeš, 2008; Gonzales a Woods, 2002). Dle Nixon a Aguado (2002) je aplikace mediánového filtru se čtvercovou maskou výběru výpočetně náročná, je tedy možné využít i jiné tvary masky. K běžným alternativním tvarům masky se řadí kříž a vertikální nebo horizontální linie, jejichž střed je zaměřen na vybraný bod obrazu, počet bodů spadajících do vybraného okolí je nižší než v případě výběru dle čtvercové masky.



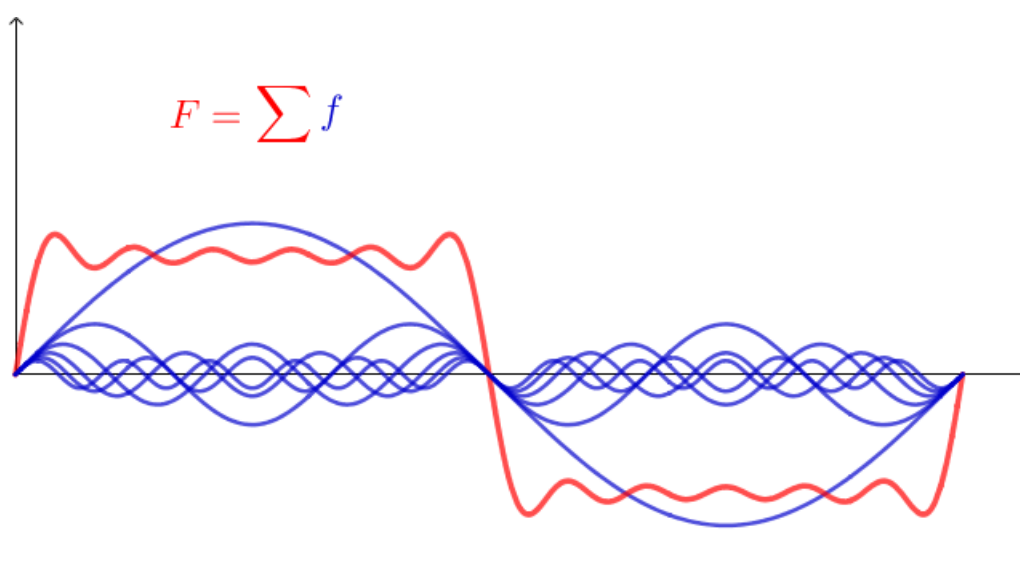
Obrázek 1.14: Obraz po aplikaci mediánového filtru a) vstupní obraz s aditivním šumem typu sůl a pepř, b) medián z oblasti okolních bodů o velikosti 5×5 , c) medián z oblasti okolních bodů 7×7 , d) medián z oblasti okolních bodů 15×15

2 Úpravy obrazu ve frekvenční doméně

Digitální obraz lze zpracovat kromě prostorové či pixelové oblasti také v oblasti frekvenční, ve které je možné provádět velké množství stejných operací, ale s výhodami jako je potřeba znatelně nižšího výpočetního výkonu, rychlost a minimální ztráta obrazové kvality při převodu do této oblasti a zpětně do domény prostorové (Russ, 2002). Pro zpracování obrazu ve frekvenční oblasti je třeba nejprve obraz převést lineární integrální transformací, k čemuž je možné využít Fourierovu či jinou transformaci (Hlaváč a Sedláček, 2009; Petrou a Petrou, 2010; Smith, 1999), například Hamardovu (Devi a Singh, 2020), Hammingovu, Haarovu (Hou et al, 2019), vlnkovou nebo diskrétní kosinovou.

2.1 Fourierova transformace

Fourierova transformace je založena na myšlence, že každá funkce může být vyjádřena jako součet řad funkcí sinus a kosinus (obr. 2.1), které jsou ovlivněny třemi parametry, amplitudou, frekvencí a fází (Russ a Russ, 2008).



Obrázek 2.1: Znázornění funkce složené ze součtu řad funkcí sinus a kosinus

Klíč et al. (2012) uvádí, že se jedná o zobrazení, které každé funkci přiřadí jinou funkci, z jejichž vlastností lze vyčíst informace o funkci původní, stručná vizualizace je znázorněná na obrázku 2.2. Vztah mezi funkcí $f(x)$ reálné proměnné x , reprezentující pozici, v případě spojitých signálů jde o čas, a Fourierovým obrazem nazvaným komplexní funkcí $F(u)$ reálné proměnné u , reprezentující frekvenci, je vyjádřen rovnicí

$$F(u) = \int_{-\infty}^{\infty} f(x)e^{-2\pi iux} dx, \quad (2.1)$$

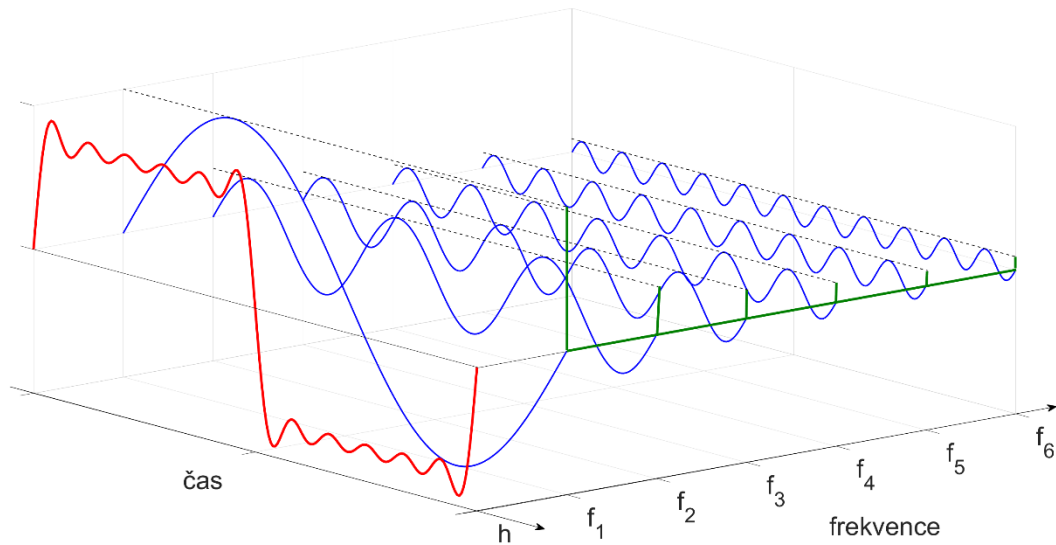
kde $i = \sqrt{-1}$. K získání funkce $f(x)$ z dané funkce $F(u)$ slouží inverzní Fourierova transformace daná vztahem

$$f(x) = \int_{-\infty}^{\infty} F(u)e^{2\pi iux} dx. \quad (2.2)$$

Exponenciální zápis lze vyjádřit pomocí Eulerova vzorce jako

$$e^{-2\pi iux} = \cos(2\pi ux) + i \cdot \sin(2\pi ux), \quad (2.3)$$

ze kterého je jasně patrné, že transformovaná funkce $F(u)$ je komplexní, skládá se tedy ze součtu reálné a imaginární části značené R a I .



Obrázek 2.2: 3D vizualizace Fourierovi transformace

Dvoudimenzionální Fourierovu transformaci se zápisem 2.4 získáme rozšířením rovnice 2.1

$$F(u, v) = \iint_{-\infty}^{\infty} f(x, y)e^{-2\pi i(ux+vy)} dx dy \quad (2.4)$$

a rozšířením rovnice 2.2 inverzní transformaci

$$f(x, y) = \iint_{-\infty}^{\infty} F(u, v) e^{2\pi i(ux+vy)} dx dy . \quad (2.5)$$

2.2 Diskrétní Fourierova transformace

Pro transformaci nespojitých konečných signálů, což jsou například i obrazová data, se aplikuje Diskrétní Fourierova transformace, která vychází ze vzorce 2.6. Většina autorů jako Russ a Russ (2008), Hlaváč a Sedláček (2009), Gonzales et al. (2009), Petrou a Petrou (2010), Smith (1999) uvádí předpis DFT pro diskrétní funkci $f(x)$, kde $x = 0, 1, 2 \dots M - 1$

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) e^{-\frac{i2\pi ux}{M}}, \quad (2.6)$$

$u = 0, 1, 2 \dots M - 1$. V případě, že je dána funkce $F(u)$, je možné získat původní funkci $f(x)$ pomocí inverzní DFT

$$f(x) = \sum_{u=0}^{M-1} F(u) e^{\frac{i2\pi ux}{M}}. \quad (2.7)$$

Činitel $1/M$ nacházející se před Fourierovou transformací je v některých publikacích, například Sojka (2000), nahrazen $1/\sqrt{M}$ či v případě dvourozměrné transformace $1/\sqrt{MN}$ a je součástí rovnice DFT i inverzní DFT. Také je možné se v literatuře setkat i se zápisem, kde je výše zmíněný činitel součástí rovnice 2.7 místo 2.6 (Gonzales et al., 2009; Smith, 2007). Tento způsob vyjádření je aplikován ve výpočetním softwaru MATLAB, který byl v této práci využit pro implementaci vybraných filtrů do vyvíjeného algoritmu.

2.2.1 Maticové vyjádření diskrétní Fourierovy transformace

Diskrétní Fourierovu transformaci je také možné vyjádřit pomocí maticového zápisu (Smith, 2007; Chu a George, 1998; Petrou a Petrou, 2010). Rovnici 2.6 lze přepsat do tvaru

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} W_M^{ux} f(x), \quad (2.8)$$

kde

$$W_M^{ux} = e^{-\frac{i2\pi}{M}ux}, \quad (2.9)$$

poté

$$F = \frac{1}{M} W_{MM} f \quad (2.10)$$

a $F = [F(0), F(1), F(2), \dots, F(M-1)]$ a $f = [f(0), f(1), f(2), \dots, f(M-1)]$ jsou sloupcové vektory s počtem M členů, W_{MM} je matice o velikosti $M \times M$ skládající se z členů W_M^{ux} , $x, u = 0, 1, 2 \dots M-1$, tedy

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ \vdots \\ F(M-1) \end{bmatrix} = \frac{1}{M} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & W_M^1 & W_M^2 & \dots & W_M^{M-1} \\ 1 & W_M^2 & W_M^4 & \dots & W_M^{2(M-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W_M^{M-1} & W_M^{(M-1)2} & \dots & W_M^{(M-1)(M-1)} \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ f(M-1) \end{bmatrix}. \quad (2.11)$$

Pro $N = 4$ je maticový zápis rovnice a jeho následná úprava

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix}, \quad (2.12)$$

W_N lze upravit dle vztahu $W_M^k = W_M^{k+jM}$, $k, j \in \mathbb{Z}$, který plyne z dosazení W_M^{k+jM} do rovnice 2.9, tedy

$$W_M^{k+jM} = e^{-\frac{i2\pi(k+jM)}{M}} = e^{-\frac{i2\pi k + i2\pi jM}{M}} = e^{-\frac{i2\pi k}{M}} e^{-i2\pi j} = W_M^k \cdot 1 = W_M^k, \quad (2.13)$$

protože

$$e^{-i2\pi j} = e^{-i2\pi j} = [\cos(2\pi) - i\sin(2\pi)]^j = 1^j = 1, \quad (2.14)$$

a tak rovnici 2.12 upravíme do tvaru

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^{-1} \\ 1 & W_4^2 & W_4^0 & W_4^2 \\ 1 & W_4^{-1} & W_4^2 & W_4^1 \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix}. \quad (2.15)$$

Dosazením rovnice 2.9 získáme

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & e^{-\frac{i2\pi 1}{4}} & e^{-\frac{i2\pi 2}{4}} & e^{-\frac{i2\pi(-1)}{4}} \\ 1 & e^{\frac{i2\pi 2}{4}} & e^{\frac{i2\pi 0}{4}} & e^{\frac{i2\pi 2}{4}} \\ 1 & e^{-\frac{i2\pi(-1)}{4}} & e^{-\frac{i2\pi 2}{4}} & e^{-\frac{i2\pi 1}{4}} \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix} \quad (2.16)$$

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix}. \quad (2.17)$$

Po maticovém násobení budou hodnoty vektoru F , tedy hodnoty DFT

$$\begin{aligned} F(0) &= \frac{1}{4}(f(0) + f(1) + f(2) + f(3)) \\ F(1) &= \frac{1}{4}(f(0) - if(1) - f(2) + if(3)) \\ F(2) &= \frac{1}{4}(f(0) - f(1) + f(2) - f(3)) \\ F(3) &= \frac{1}{4}(f(0) + if(1) - f(2) - if(3)) \end{aligned} \quad (2.18)$$

2.2.2 Rychlá Fourierova transformace

Transformace diskrétní funkce $f(x)$ o N prvcích vyžaduje při využití diskrétní Fourierovy transformace alespoň N^2 operací, se zvyšujícím se počtem prvků tedy exponenciálně roste výpočetní náročnost i čas. Vývoj algoritmů rychlé Fourierovy transformace (FFT) započal již v 60. letech 20. století. Jedním ze základních a nejznámějších algoritmů z této skupiny je Cooley-Tukey algoritmus, který autoři představili roku 1965, založený na opakovaném půlení a dvojnásobení dané množiny hodnot, jenž umožňuje snížit výpočetní náročnost na $N \log_2 N$ operací, což je při počtu prvků $N = 1024$ více než stonásobný rozdíl (Hlaváč a Sedláček, 2009; Klíč et al., 2012; Russ a Russ, 2008). Chu a George (1998) ve své knize uvádí, že základní myšlenkou FFT je paradigma rozdělení a panuj, které rozděluje do tří hlavních kroků. Při prvním kroku dochází k rozdělení skupiny do dvou či více podskupin o menší velikosti, při druhém kroku je každá podskupina řešena rekurzivně stejným algoritmem, kdy jsou stanoveny mezní podmínky k zavedení rekurze. V posledním kroku je kombinací jednotlivých řešení podskupin získáno řešení původní skupiny. Rozdělení 2^r , $r \in \mathbb{N}$

prvků do dvou polovičních podskupin je využito například v rámci algoritmu s decimací v čase, DIT FFT, či s decimací ve frekvenci, DIF FFT. Podrobné rozpracování metod rychlé Fourierovi transformace je uvedeno ve výše zmíněné literatuře.

2.3 Dvourozměrná diskretní Fourierova transformace

Pro převod dvourozměrné diskretní funkce $f(x, y)$, tedy i obrazu, o velikosti $M \times N$, $x = 0, 1, \dots, M - 1$, $y = 0, 1, \dots, N - 1$, z prostorové oblasti do oblasti frekvenční je možné použít dvoudimenzionální diskretní Fourierovu transformaci, jenž je přímým rozšířením jednodimenzionální DFT s předpisem daným rovnicí 2.19

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-i2\pi(\frac{ux}{M} + \frac{vy}{N})}, \quad (2.19)$$

kde $u = 0, 1, \dots, M - 1$, $v = 0, 1, \dots, N - 1$ (Petrou a Petrou, 2010). Obraz daný funkcí $f(x, y)$ můžeme znázornit jako matici $A(M, N)$, kde M je počet řádků, N je počet sloupců, x odpovídá číslu řádku zmenšenému o 1 a y číslu sloupce zmenšenému o 1

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MN} \end{bmatrix} \leftrightarrow \quad (2.20)$$

$$f = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0, N-1) \\ f(1,0) & f(1,1) & \cdots & f(1, N-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1, N-1) \end{bmatrix}.$$

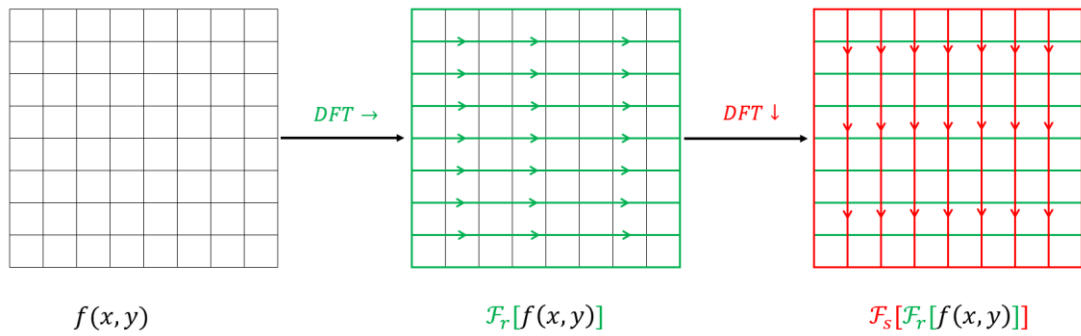
Předpis inverzní dvoudimenzionální DFT je dán rovnicí

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{i2\pi(\frac{ux}{M} + \frac{vy}{N})}, \quad (2.21)$$

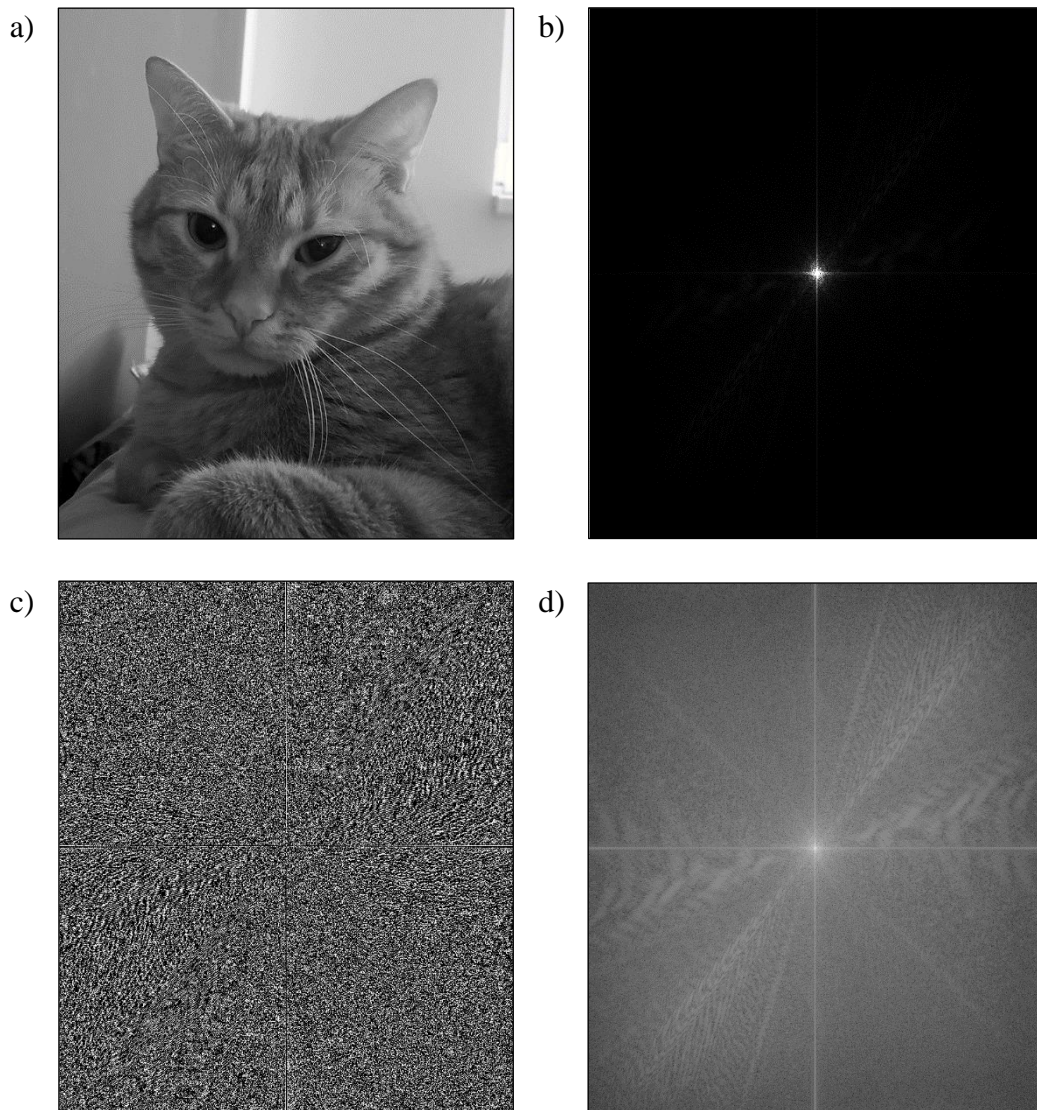
kde $x = 0, 1, \dots, M - 1$; $y = 0, 1, \dots, N - 1$. Rovnici 2.19 lze také upravit do tvaru

$$F(u, v) = \frac{1}{M} \sum_{x=0}^{M-1} \left[\frac{1}{N} \sum_{y=0}^{N-1} e^{-i2\pi(\frac{vy}{N})} f(x, y) \right] e^{-i2\pi(\frac{ux}{M})}, \quad (2.22)$$

$u = 0, 1, \dots, M - 1$; $v = 0, 1, \dots, N - 1$, kde výraz v hranatých závorkách odpovídá jednorozměrné DFT pro jednotlivé řádky. Při 2D DFT obrazu lze tedy nejprve nahradit všechny řádky obrazu jejich jednorozměrnou DFT a poté provést jednorozměrnou DFT po sloupcích (Yoo, 2004).



Obrázek 2.3: Znárodnění realizace dvourozměrné diskretní Fourierovy transformace postupnou aplikací jednorozměrných DFT



Obrázek 2.4: Dvourozměrná DFT obrazu, a) vstupní obraz, b) výkonové spektrum, c) fázové spektrum, d) výkonové spektrum po logaritmické transformaci hodnot

Funkce $F(u, v)$ vzniklá Fourierovou transformací je komplexní, skládá se z reálné části, $R(u, v)$, a imaginární části, $I(u, v)$. Velikost funkce je vyjádřena jako její absolutní hodnota, tedy odmocnina součtu druhých mocnin obou částí funkce, viz vztah 2.24, a je nazvána jako amplitudové frekvenční spektrum obrazu, jeho druhou

mocninou je vyjádřeno výkonové spektrum $P(u, v)$. V tomto spektru je často zobrazováno znázornění DFT obrazu, na které pro větší přehlednost někteří autoři aplikují logaritmickou transformaci ke snížení velkého rozmezí hodnot, jak je ilustrováno na obrázku 2.4d). S funkcí $F(u, v)$ souvisí také fázové spektrum $\phi(u, v)$ definované jako arkus tangens podílu imaginární a reálné části (Russ a Russ, 2008; Gonzales et al, 2009).

$$F(u, v) = R(u, v) + iI(u, v) \quad (2.23)$$

$$|F(u, v)| = \sqrt{R(u, v)^2 + I(u, v)^2} \quad (2.24)$$

$$\phi(u, v) = \tan^{-1} \frac{I(u, v)}{R(u, v)} \quad (2.25)$$

$$P(u, v) = |F(u, v)|^2 = R(u, v)^2 + I(u, v)^2 \quad (2.26)$$

2.3.1 Maticové vyjádření dvoudimenzionální diskretní Fourierovy transformace

Dvoudimenzionální DFT lze stejně jako jednodimenzionální DFT vyjádřit maticovým zápisem. Rovnici 2.19 přepíšeme do tvaru

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} W_M^{ux} f(x, y) W_N^{vy}, \quad (2.27)$$

kde

$$W_M^{ux} = e^{-\frac{i2\pi}{M}ux}, \quad (2.28)$$

$$W_N^{vy} = e^{-\frac{i2\pi}{N}vy}, \quad (2.29)$$

poté

$$F = \frac{1}{MN} W_{MM} f W_{NN}, \quad (2.30)$$

F a f jsou matice o rozměru $M \times N$, W_{MM} je matice o velikosti $M \times M$ skládající se z členů W_M^{ux} , $x, u = 0, 1, 2 \dots M - 1$ a W_{NN} je matice o velikosti $N \times N$ skládající se z členů W_N^{vy} , $y, v = 0, 1, 2 \dots N - 1$ (Hlaváč a Sedláček, 2009; Wang, 2010). Například maticový zápis 2D DFT obrazu s počtem bodů na sloupec a řádek $M = 4$ a $N = 3$ bude

$$F = \frac{1}{12} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) \\ f(1,0) & f(1,1) & f(1,2) \\ f(2,0) & f(2,1) & f(2,2) \\ f(3,0) & f(3,1) & f(3,2) \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & W_3^1 & W_3^2 \\ 1 & W_3^2 & W_3^4 \end{bmatrix} \quad (2.31)$$

kde

$$F = \begin{bmatrix} F(0,0) & F(0,1) & F(0,2) \\ F(1,0) & F(1,1) & F(1,2) \\ F(2,0) & F(2,1) & F(2,2) \\ F(3,0) & F(3,1) & F(3,2) \end{bmatrix}. \quad (2.32)$$

2.4 Filtrace obrazu ve frekvenční doméně

Předzpracování obrazu ve frekvenční doméně má několik velmi užitečných vlastností, z nichž zásadní pro filtraci obrazu je, že násobení odpovídá konvoluci v prostorové doméně, tedy aplikace filtrů je početně jednodušší (Smith, 1999). V prostorové oblasti vzniká filtrací upravený obraz $h(x, y)$ konvolucí obrazu daného diskretní funkcí $f(x, y)$ s kernelem $g(x, y)$

$$h(x, y) = f(x, y) * g(x, y), \quad (2.33)$$

ve frekvenční oblasti je výstup $H(u, v)$ výsledkem násobením po prvcích transformované funkce $F(u, v)$ a filtrem $G(u, v)$, což jsou Fourierovy obrazy $f(x, y)$ a $g(x, y)$, platí

$$H(u, v) = F(u, v)G(u, v), \quad (2.34)$$

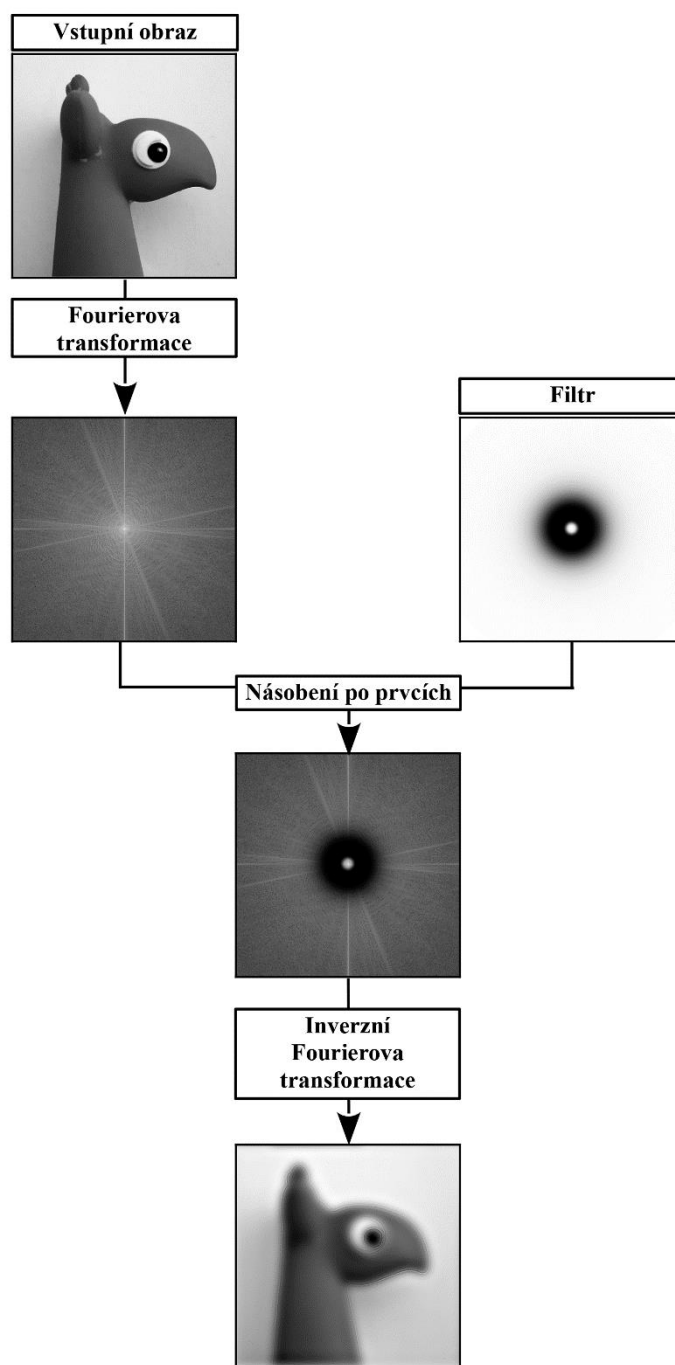
$H(u, v)$ je Fourierovým obrazem $h(x, y)$, tudíž konvoluce v prostorové doméně je ekvivalentní s násobením v doméně frekvenční

$$f(x, y) * g(x, y) \Leftrightarrow F(u, v)G(u, v). \quad (2.35)$$

V rámci praktické aplikace lze mezi těmito dvěma operacemi najít i drobné rozdíly, při konvoluci v prostorové doméně je třeba rozšířit obraz o okrajový rám, z důvodu zpracování hodnoty krajních pixelů, tedy aby kernel nezasahoval do oblastí bez obrazových informací. Při konvoluci násobením ve frekvenční doméně není provedení této úpravy nutné, avšak na místě okrajových bodů mohou vznikat obrazové artefakty (Russ, 2002). Postup úpravy obrazu se skládá z několika po sobě jdoucích operací, nejprve je obraz převeden Fourierovou transformací, obvykle nějakým FFT algoritmem, z prostorové do frekvenční domény, poté dochází k aplikaci frekvenčního filtru

vynásobením filtru s transformovaným obrazem, nakonec je výpočtem inverzní transformace, IFFT, výsledek převeden zpět do prostorové domény (Hlaváč a Sedláček, 2009; Jensen, 2015). Celý postup můžeme zapsat jako

$$\begin{aligned} f(x, y) &\rightarrow FFT[f(x, y)] \rightarrow F(u, v) \rightarrow F(u, v)G(u, v) \rightarrow \\ &\rightarrow H(u, v) \rightarrow IFFT[H(u, v)] \rightarrow h(x, y). \end{aligned} \quad (2.36)$$



Obrázek 2.5: Schematické znázornění postupu při filtraci obrazu ve frekvenční doméně za využití Fourierovy transformace

Frekvenční filtry lze kategorizovat dle potlačovaných frekvencí na filtry dolní propusti, označované také jako low-pass filtry, které slouží k odstranění vysokofrekvenčních informací jako obrazový šum a zachování nízkých frekvencí z požadovaného rozsahu neboli šířky pásma. Filtry horní propusti, high-pass filtry, fungující na opačném principu, tedy vysokofrekvenční informace ponechávají, a filtry pásmové propusti, band-pass filtry, používané pro zachování frekvencí ze specifického pásma (Gonzales a Woods, 2002; Nixon a Aguado, 2002).

2.4.1 Low-pass filtry

Filtry tohoto typu jsou obdobou vyhlazovacích filtrů prostorové oblasti, slouží ke stejnému účelu, a to k odstranění šumu či vyhlazení velkých lokálních nespojitostí, s čímž se pojí i ztráta drobných detailů a rozmazání hran, které nabývají vysokých frekvencí. Za hlavní zástupce jsou považovány Ideální (ILPF), Butterworthův (BLPF) a Gaussův low-pass filtr (GLPF), jenž pokrývají rozsah v rámci rozdělení frekvencí od velmi ostrých filtrů po velice plynulé (Gonzales et al., 2009). S ohledem na podobný základ a stejný princip budou dále filtry popsány společně a navzájem porovnány. Filtry jsou definovány následujícími přenosovými funkcemi.

Ideální low-pass filtr

$$G(u, v) = \begin{cases} 1 & D(u, v) \leq D_0 \\ 0 & D(u, v) > D_0 \end{cases}, \quad (2.37)$$

Butterworthův low-pass filtr

$$G(u, v) = \frac{1}{1 + \left[\frac{D(u, v)}{D_0} \right]^{2n}}, \quad (2.38)$$

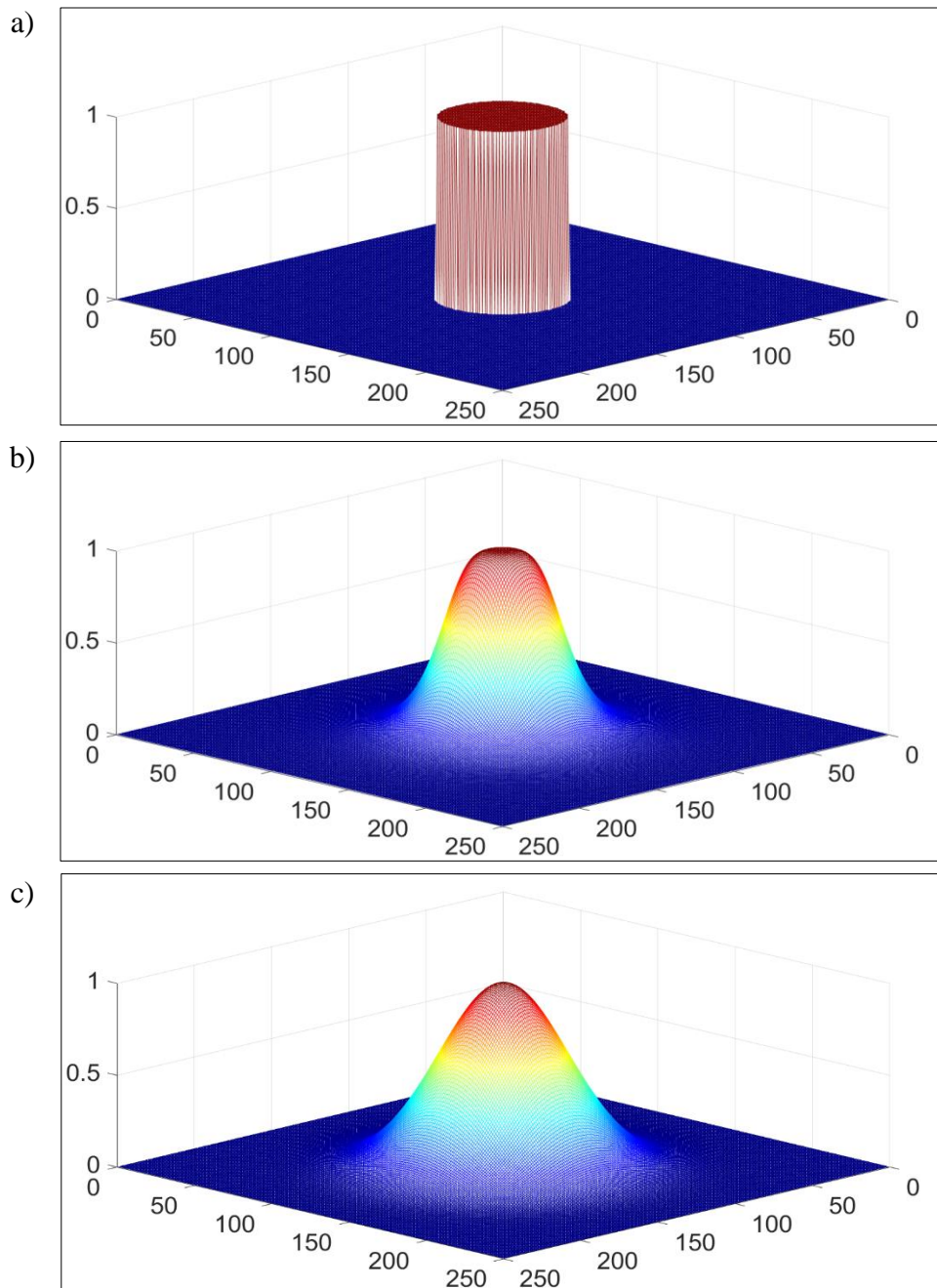
Gaussův low-pass filtr

$$G(u, v) = e^{-\frac{D^2(u, v)}{2D_0^2}}, \quad (2.39)$$

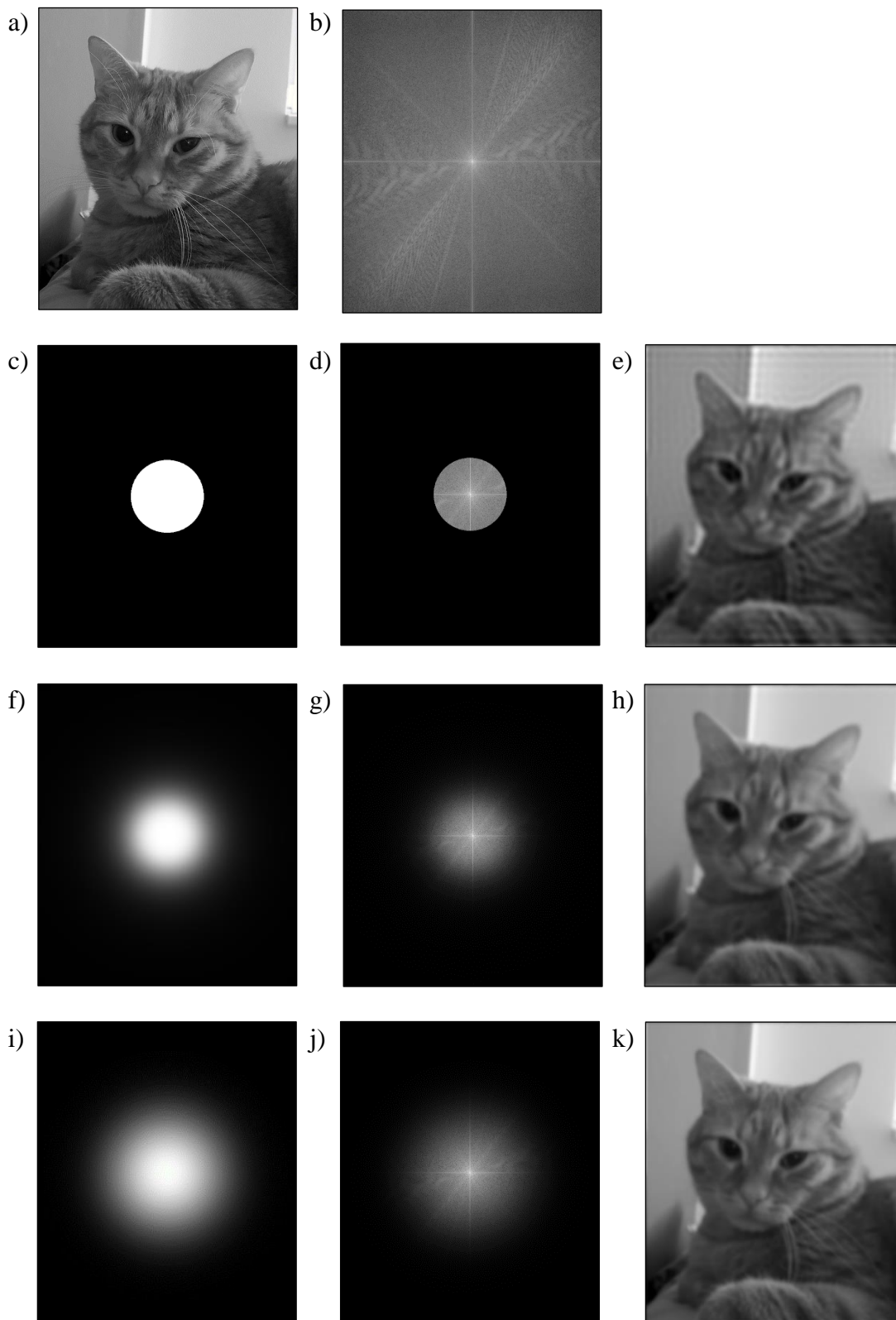
kde $D(u, v)$ představuje vzdálenost mezi bodem (u, v) a středem frekvenční plochy filtru o stejné velikosti jako obraz, $M \times N$, označovaného za počátek transformace, která je dána vztahem

$$D(u, v) = \sqrt{\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2}, \quad (2.40)$$

hodnoty $M/2$ a $N/2$ jsou souřadnice středu. Parametr D_0 stanovuje konkrétní vzdálenost od středu a ovlivňuje, či v případě ideálního filtru přímo vymezuje, mezní frekvence filtrace, s jehož rostoucí hodnotou se zvětšuje šířka propustného pásma a snižuje efekt rozmazání obrazu. Na podobu, zvláště na strmost, Butterworthova filtru má vliv také parametr n , který při malé hodnotě odpovídá za plynulý přechod k hraniční frekvenci, naopak u velkých hodnot parametru je přechod ostrý, přibližující se ideálnímu filtru (Dogra a Bhalla, 2014; Shaikh et al., 2016, Zhang, 2021).



Obrázek 2.6: Grafy low-pass filtrů pro $D_0=30$ a) Ideální low-pass filtr, b) Butterworthův low-pass filtr, $n=2$, c) Gaussův low-pass filtr



Obrázek 2.7: Aplikace low-pass filtrů, a) vstupní obraz, b) výkonové spektrum vstupního obrazu, c) znázornění ILPF, $D_0=100$, d) výsledek aplikace ILPF na výkonové spektrum obrazu, e) výsledný obraz po inverzní DFT, f) znázornění BLPF, $D_0=100$, $n=2$, g) výsledek aplikace BLPF na výkonové spektrum obrazu, h) výsledný obraz po inverzní DFT, i) znázornění GLPF, $D_0=100$, j) výsledek aplikace GLPF na výkonové spektrum obrazu, k) výsledný obraz po inverzní DFT

2.4.2 High-pass filtry

Pro eliminaci součástí nízkých frekvencí a zachování frekvencí vysokých slouží high-pass filtry, nazývané také jako filtry horní propusti. Jejich zásadní funkcí je zvýšení ostrosti obrazu, zvýraznění bodů, křivek a hran, jež jsou součástí přechodů hodnot jasu, prudkých změn lokálních hodnot pixelů. Vzhledem k low-pass filtrům se jedná o obrácený přístup k filtraci, a tak transferové funkce mohou být vyjádřeny vztahem

$$G_{HP}(u, v) = 1 - G_{LP}(u, v), \quad (2.41)$$

kde $G_{LP}(u, v)$ je funkce odpovídajícího low-pass filtru (Dewangan a Sharma, 2017; Zawaideh et al., 2017). Nejpoužívanější filtry této skupiny jsou filtry vzniklé analogicky vůči hlavním filtrům dolní propusti, tedy Ideální (IHPF), Butterworthův (BHPF) a Gaussův high-pass filtr (GHPF), jejichž transferové funkce uvedené dále jsou odvozeny ze vztahu 2.41.

Ideální low-pass filtr

$$G(u, v) = \begin{cases} 1 & D(u, v) > D_0 \\ 0 & D(u, v) \leq D_0 \end{cases}, \quad (2.42)$$

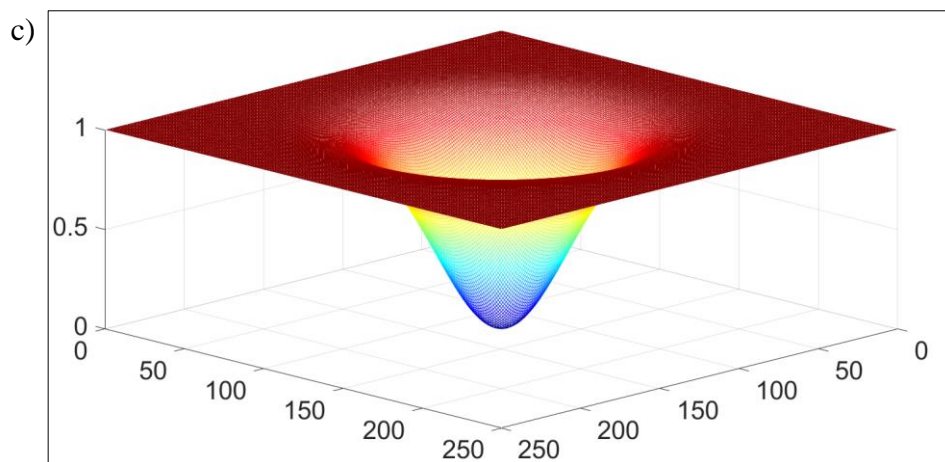
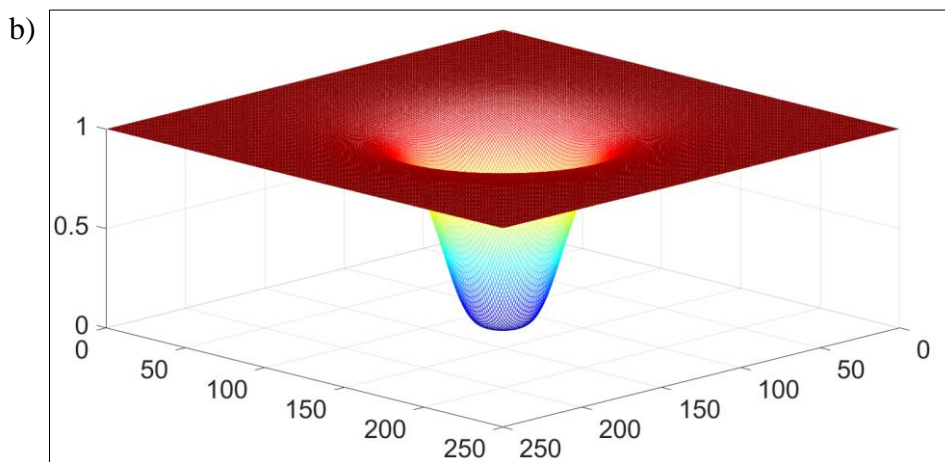
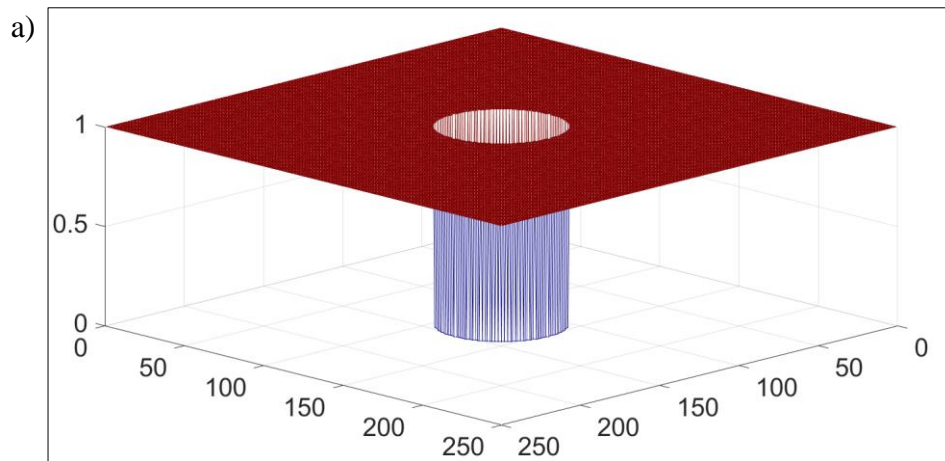
Butterworthův low-pass filtr

$$G(u, v) = \frac{1}{1 + \left[\frac{D_0}{D(u, v)} \right]^{2n}}, \quad (2.43)$$

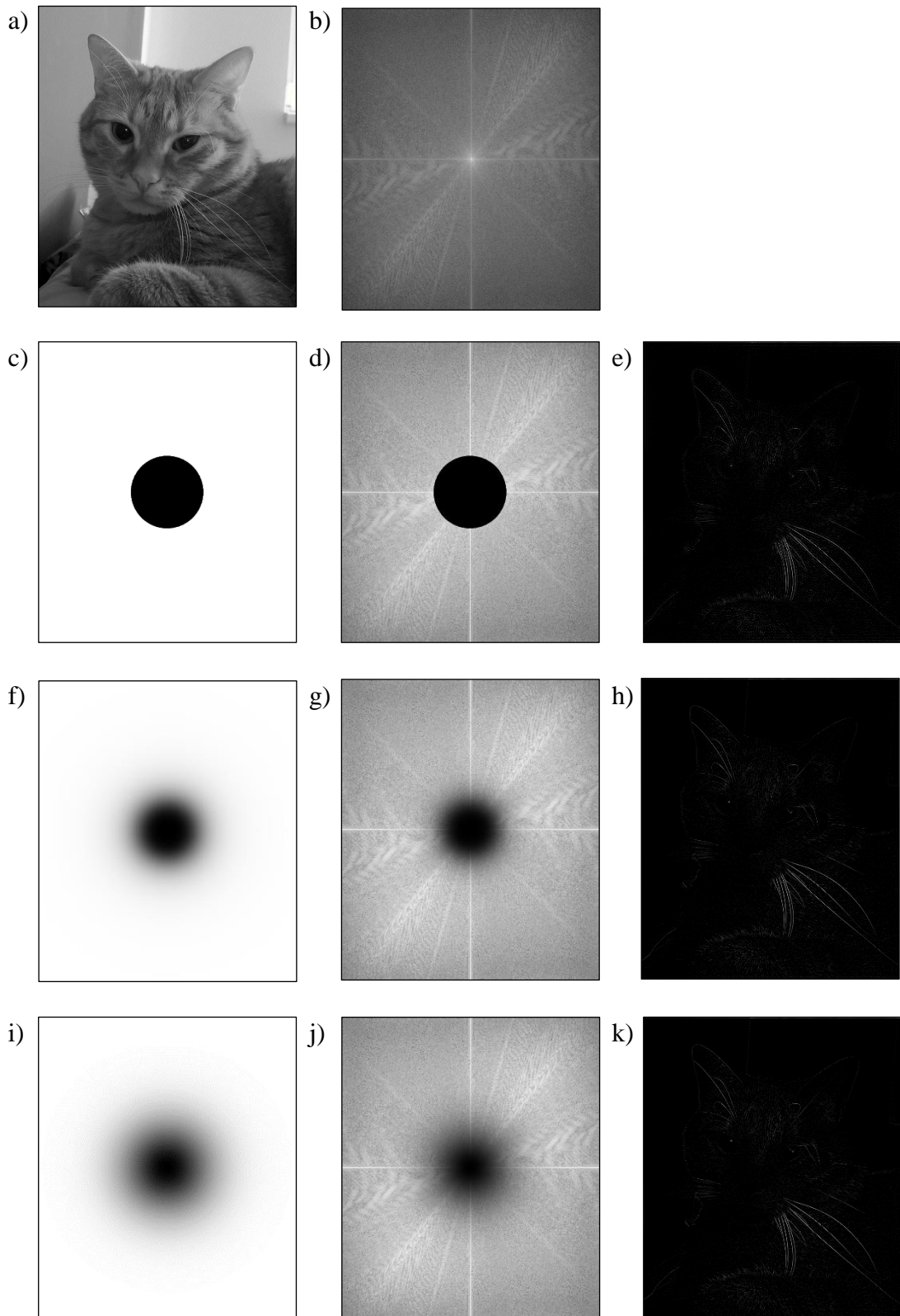
Gaussův low-pass filtr

$$G(u, v) = 1 - e^{-\frac{D^2(u, v)}{2D_0^2}}. \quad (2.44)$$

Vzdálenost $D(u, v)$ bodu (u, v) od středu roviny frekvencí má stejný předpis jako u low-pass filtrů, koeficient D_0 ovlivňuje rozsah potlačených frekvencí, s jeho klesající velikostí roste zvýraznění hran. Parametr n má výrazný dopad na strmost Butterworthova high-pass filtru v tentýž stylu jako na jeho verzi z oblasti dolní propusti (Gonzales a Woods, 2002; de Ruijter et al., 2020; Shaikh et al., 2016).



Obrázek 2.8: Grafy high-pass filtrů pro $D_0=30$ a) Ideální high-pass filtr, b) Butterworthův high-pass filtr, $n=2$, c) Gaussův high-pass filtr



Obrázek 2.9: Aplikace high-pass filtrů, a) vstupní obraz, b) výkonové spektrum vstupního obrazu, c) znázornění IHPF, $D_0=100$, d) výsledek aplikace IHPF na výkonové spektrum obrazu, e) výsledný obraz po inverzní DFT, f) znázornění BHPF, $D_0=100$, $n=2$, g) výsledek aplikace BHPF na výkonové spektrum obrazu, h) výsledný obraz po inverzní DFT, i) znázornění GHPF, $D_0=100$, j) výsledek aplikace GHPF na výkonové spektrum obrazu, k) výsledný obraz po inverzní DFT

2.4.3 Band-pass filtry

Band-pass filtry umožní průchod pouze frekvencím z určitých oblastí vybraných pomocí mezikružší, při aplikaci radiálních filtrů, či několika kruhových oblastí párově souměrných dle středu frekvenční plochy, v případě notch filtrů. Využití těchto filtrů se skýtá hlavně v odstranění periodického šumu nebo vzorů překrývajících obraz. Stejně jako u filtrů dolní a horní propusti vychází základní zástupci z Ideálního, Butterworthova a Gaussova filtru (Hlaváč a Sedláček, 2009; Zhang, 2021). Filtry jsou dané následujícími transformačními funkcemi.

Ideální notch filtr

$$G(u, v) = \begin{cases} 1 & D_1(u, v) > D_0 \wedge D_2(u, v) > D_0, \\ 0 & \text{jinak} \end{cases}, \quad (2.45)$$

Butterworthův notch filtr

$$G(u, v) = \frac{1}{1 + \left[\frac{D_0}{D_1(u, v)D_2(u, v)} \right]^n}, \quad (2.46)$$

Gaussův notch filtr

$$G(u, v) = 1 - e^{-\frac{1}{2} \left[\frac{D_1(u, v)D_2(u, v)}{D_0^2} \right]}, \quad (2.47)$$

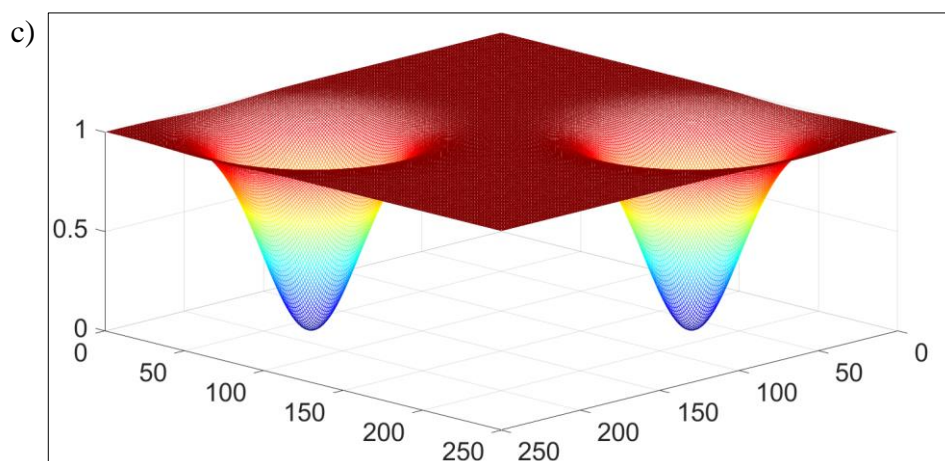
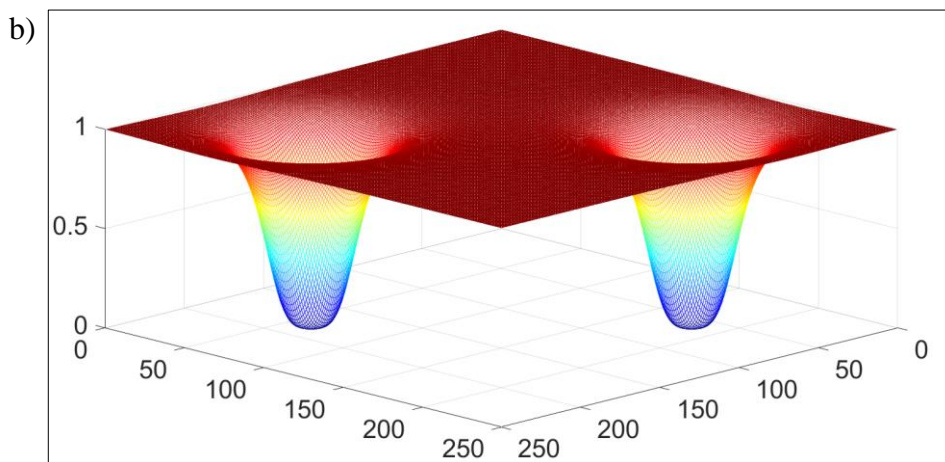
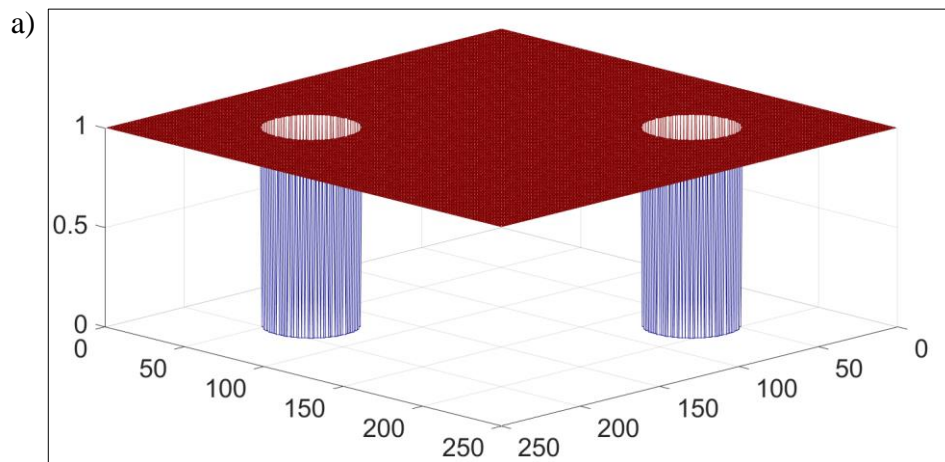
Kde

$$D_1(u, v) = \sqrt{\left(u - \frac{M}{2} - u_0\right)^2 + \left(v - \frac{N}{2} - v_0\right)^2} \quad (2.48)$$

a

$$D_2(u, v) = \sqrt{\left(u - \frac{M}{2} + u_0\right)^2 + \left(v - \frac{N}{2} + v_0\right)^2} \quad (2.49)$$

Body (u_0, v_0) a dle bodové symetrie $(-u_0, -v_0)$ jsou středy kruhových oblastí filtrovaných frekvencí s poloměrem D_0 . D_1 a D_2 představují vzdálenosti jednotlivých bodů filtru od center filtrovaných oblastí, $M/2$ a $N/2$ jsou souřadnice středu frekvenční plochy filtru.



Obrázek 2.10: Grafy notch filtrů pro $D_0=22$ a) Ideální notch filtr, b) Butterworthův notch filtr, $n=2$, c) Gaussův notch filtr

Ideální radiální band-pass filtr

$$G(u, v) = \begin{cases} 1 & D_0 - \frac{W}{2} \leq D(u, v) \leq D_0 + \frac{W}{2}, \\ 0 & \text{jinak} \end{cases} \quad (2.50)$$

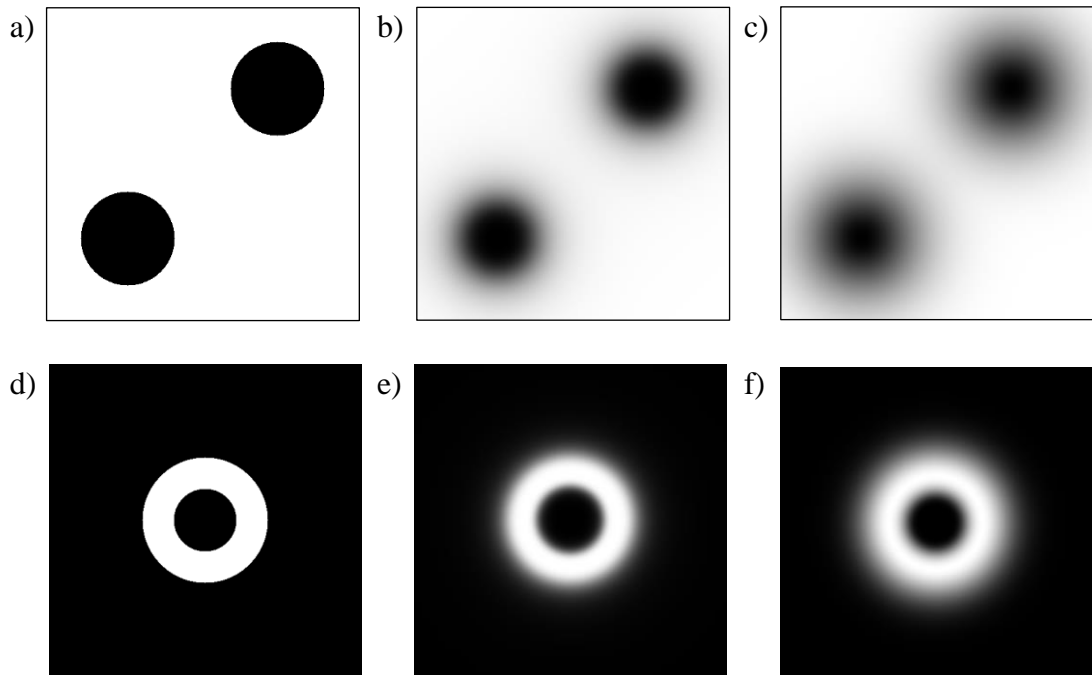
Butterworthův radiální band-pass filtr

$$G(u, v) = \frac{1}{1 + \left[\frac{D^2(u, v) - D_0^2}{D(u, v)W} \right]^n}, \quad (2.51)$$

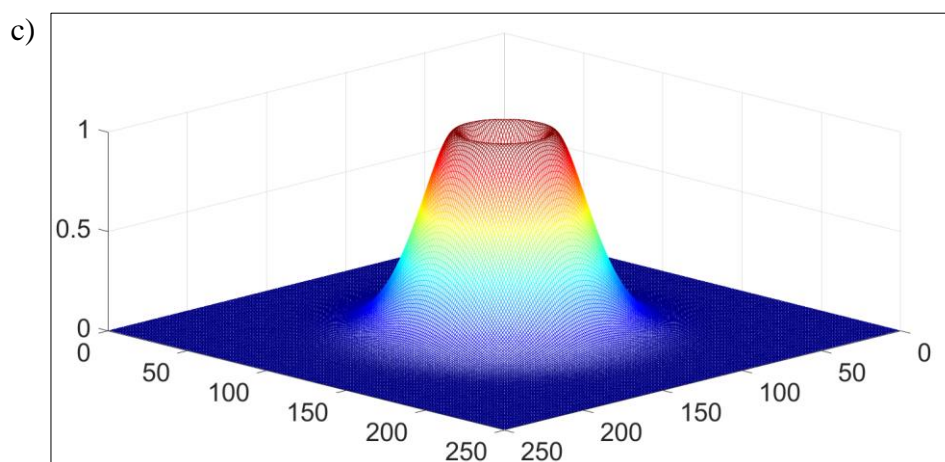
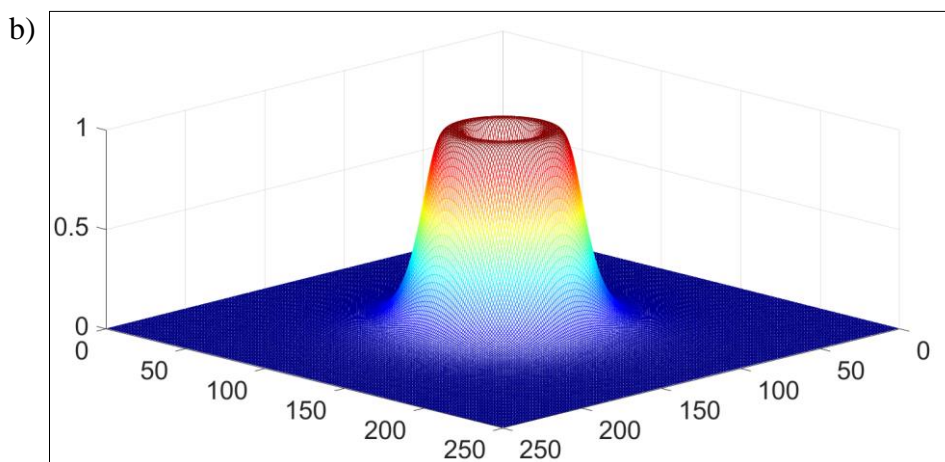
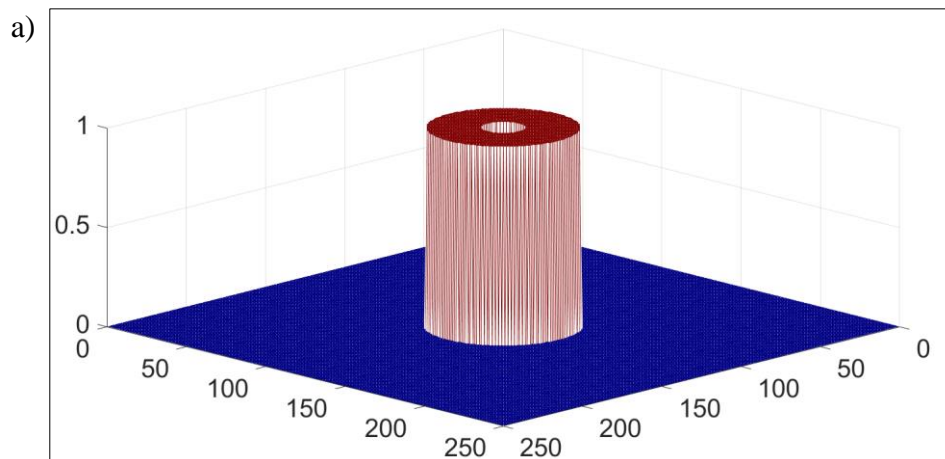
Gaussův radiální band-pass filtr

$$G(u, v) = e^{-\frac{1}{2} \left[\frac{D^2(u, v) - D_0^2}{D(u, v)W} \right]^2}. \quad (2.52)$$

Šířka pásma propusti, tedy vzdálenost mezi horní a dolní mezní frekvencí je zastoupena parametrem W , D_0 je poloměr prstencového pásma zachovaných frekvencí, vzdálenost bodů (u, v) je určena proměnnou $D(u, v)$ (Gonzales a Woods, 2002; Gonzales et al., 2009; Zhang, 2021).



Obrázek 2.11: Znárodnění notch a band-pass filtrů $D_0=22$, $w=25$, a) Ideální notch filtr, b) Butterworthův notch filtr, $n=2$, c) Gaussův notch filtr, d) Ideální radiální band-pass filtr, e) Butterworthův radiální band-pass filtr, $n=2$, f) Gaussův radiální band-pass filtr



Obrázek 2.12: Grafy radiálních band-pass filtrů pro $D_0=22$, $w=25$, a) Ideální radiální band-pass filtr, b) Butterworthův radiální band-pass filtr, $n=2$, c) Gaussův radiální band-pass filtr

3 Segmentace obrazu

Segmentace je velice důležitá operace při zpracování obrazu za účelem získání obsažených informací, během které dochází k rozdělení obrazu do oblastí s určitou pravděpodobností odpovídajících strukturním jednotkám scény obrazu, rozdělení na souvislé oblasti složené ze základních prvků s podobnými vlastnostmi, rozpoznání objektů zájmu a oddělení objektů od nezajímavého pozadí. Segmentace je občas popisována analogicky k procesu vidění jako odlišení popředí a pozadí, kdy lidské vidění vykládá obraz na základě vztahů mezi prvky vytvořenými nevědomě v nižších úrovních zrakového systému, je proto vhodné i v systémech počítačové analýzy obrazu postupovat od prvků z nízkých úrovní vzestupně, počáteční kritéria vytvářet na úrovni pixelů (Russ, 2002; Yoo, 2004).

Zhang (2021) uvádí formální definici segmentace obrazu, kdy R představuje oblast vstupního obrazu, která je segmentací rozdělená na několik neprázdných podoblastí R_i , $i = 1, 2, \dots, n$, které splňují následující podmínky:

- 1) $\bigcup_{i=1}^n R_i = R$.
- 2) Pro všechny i a j , kdy $i \neq j$ platí, že $R_i \cap R_j = \emptyset$.
- 3) Pro $i = 1, 2, \dots, n$ je $P(R_i) = \text{PRAVDA}$.
- 4) Pro $i \neq j$ je $P(R_i \cup R_j) = \text{NEPRAVDA}$.
- 5) Pro $i = 1, 2, \dots, n$ je R_i propojená oblast,

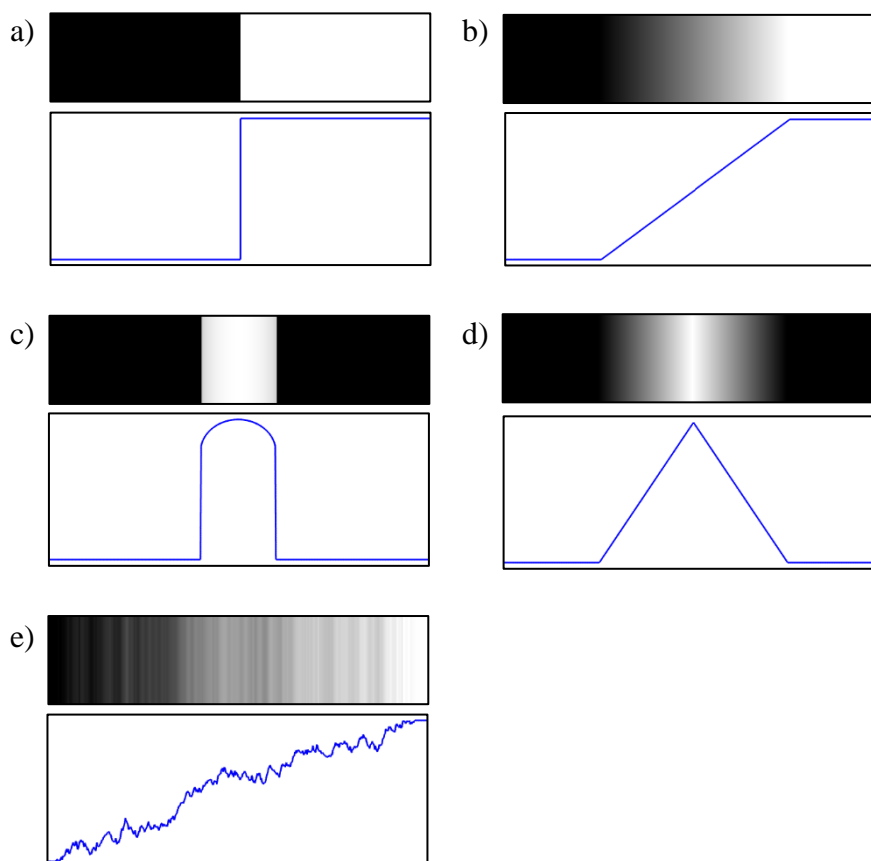
přičemž $P(R_i)$ vyjadřuje jednotný predikát, což je speciální logická funkce jedné proměnné, jejíž výstup PRAVDA nebo NEPRAVDA závisí pouze na vlastnostech bodů náležících do posuzované množiny, pro všechny členy množiny R_i , \emptyset značí prázdnou množinu.

Základem algoritmů pro segmentaci obrazu jsou obecně vzato dvě elementární vlastnosti hodnot intenzity jasu, což jsou nespojitost a podobnost. Do první kategorie spadají přístupy rozdělení obrazu na základě prudkých změn jasových hodnot, jako jsou hrany v obrazu, tedy metody spočívající v detekci kontur, jež označují hranice jednotlivých objektů (Gonzales a Woods, 2002; Sojka et al., 2011). Metody tohoto typu budou dále v práci blíže popsány. Hlavní přístupy druhé kategorie jsou založeny na rozdělení obrazu na oblasti s podobností dle stanovených kritérií. K těmto přístupům patří například prahování, kdy dochází k extrakci obrazu na základě stanovené prahové hodnoty, jejíž velikost může být určena různými způsoby, například přímým výběrem (Bloch et al., 2019) nebo nalezením průsečíku aproximací hodnot histogramu

funkcemi normálního rozdělení s vrcholy odpovídajícími dvěma nejvyšším hodnotám histogramu (Asdrubali et al., 2018). Další velmi používanou metodou segmentace obrazu na základě podobných hodnot bodů je K-means segmentační algoritmus, jehož aplikací dochází k rozčlenění obrazu na shluky neboli klastry reprezentující vybrané kategorie. Na počátku segmentace jsou jednotlivé body přiřazeny do shluků náhodně, poté jsou shluky stále aktualizovány a body jsou přerazovány mezi shluky tak, aby se uvnitř stejného shluku podobnost hodnot bodů zvyšovala a mezi rozdílnými shluky snižovala (Akram et al., 2019; Wen et al., 2020; Yoo, 2004).

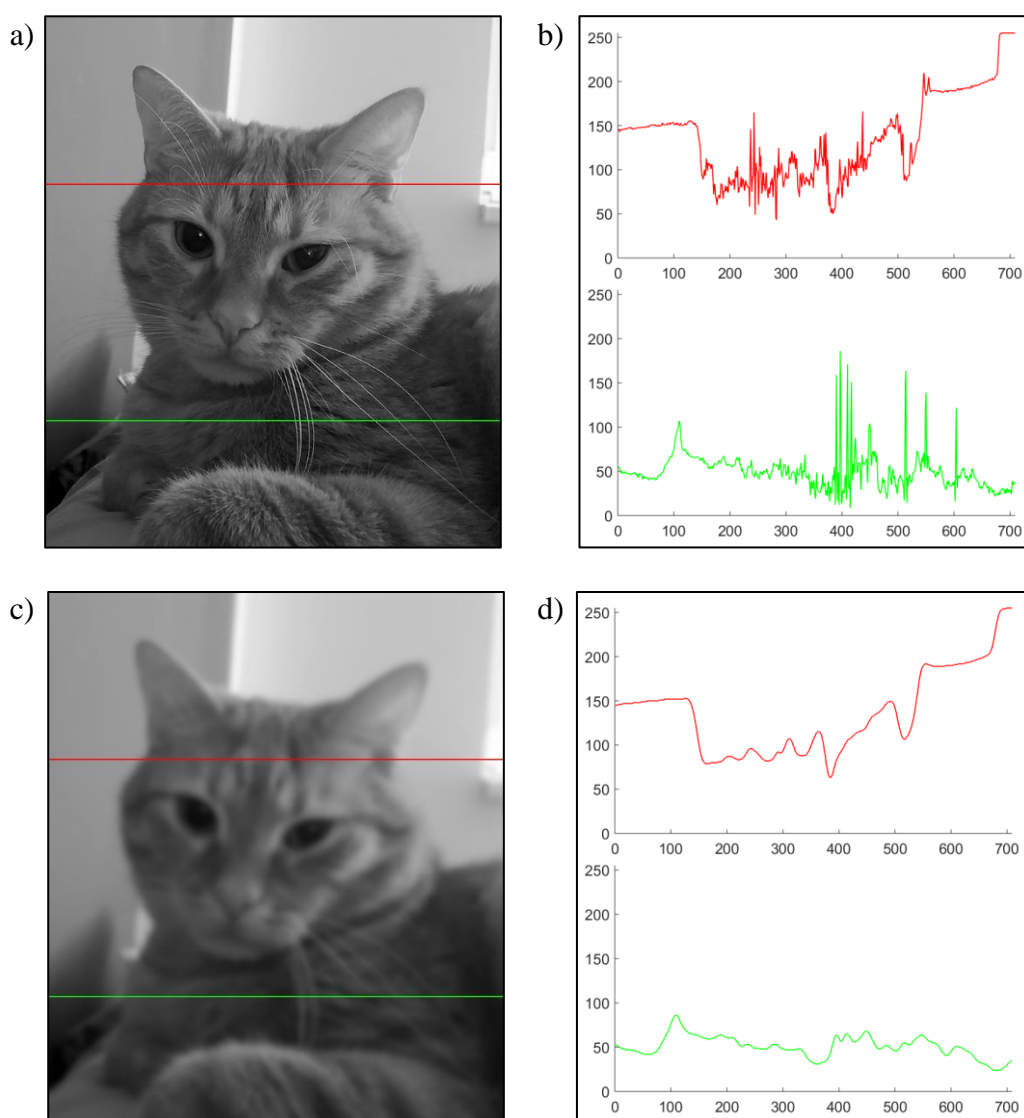
3.1 Detekce hran

Jedním ze základních příznaků vyskytujících se v obrazu a poskytujících důležité informace pro jeho interpretaci jsou hrany, které lze popsat jako výraznou lokální změnu intenzity jasu obrazových bodů nebo také přechod mezi tmavými a světlými pixely odpovídající lokálnímu obrysu objektu (Gonzales a Woods, 2002; Magnier et al., 2018).



Obrázek 3.1: Znárodnění typů hran, a) skoková hrana, b) šikmá hrana; c) liniová hrana, d) střechová hrana, e) zašuměná hrana

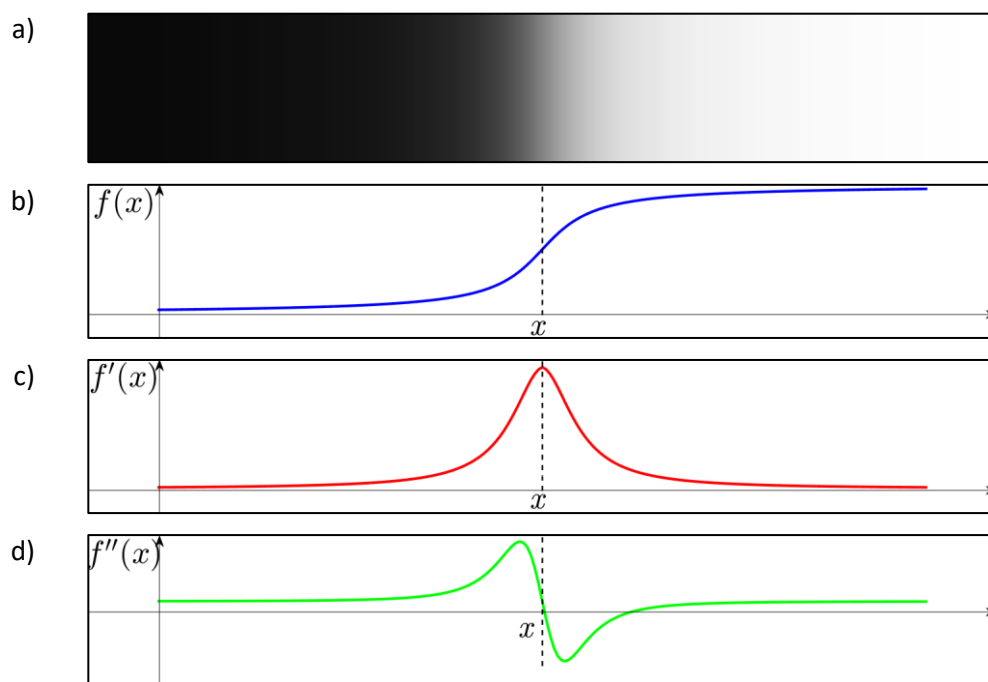
Tyto nespojitosti jasu jsou kategorizovány dle jejich jednorozměrného profilu na a) skokové, kdy dochází k ostrým, náhlým změnám hodnot sousedních pixelů, b) šikmé, označované také jako ramp edge, které tvoří plynulý přechod, jejichž sklon nepřímo úměrně souvisí s rozmazáním hrany, se snižujícím se rozmazáním se sklon zvyšuje, c) liniové hrany se vyznačující skokovou změnu hodnot a strmým návratem k hodnotám původním, d) střeškové hrany charakterizované plynulým změnou hodnot a jejich plynulým návratem zpět, e) zašuměné hrany, jenž obsahují výkyvy hodnot v rámci průběhu hrany. V reálných obrazech se skokové a liniové hrany vyskytují pouze zřídka, nejčastěji se lze setkat s hranami zašuměnými (Hlaváč a Sedláček, 2009; Jain et al., 1995).



Obrázek 3.2: a) Vzorový obraz s vyznačenými jednorozměrnými průřezy, b) grafy průběhu jasu vybraných průřezů obrazu, c) obraz po aplikaci Gaussova filtru s parametrem $\sigma=3$, d) grafy průběhu jasu vybraných průřezů obrazu po aplikaci Gaussova filtru

Množství hran vyskytujících se v obrazu je ovlivněno jeho členitostí a také šumem. Na obrázku 3.2b) jsou znázorněny profily jednorozměrných průřezů obrazu 3.2a), ze kterých je patrné, že malému počtu členitostí okrajů průřezu odpovídá i nízký výskyt výrazných změn hodnot jasu v jeho profilu, méně výrazný šum se v grafu projevuje drobnými výkyvy hodnot. Naopak ve střední, velice členité, části průřezu je přítomnost hran vysoká, což je zřetelné v průběhu odpovídající oblasti grafu. Snížení počtu a zvýraznění nejdůležitějších hran lze uskutečnit aplikací vyhlazovacího filtru, viz obr. 3.2d), kde jsou uvedeny grafy průřezů obrazu po použití Gaussova filtru s parametrem σ o velikosti 3.

K detekci hran se využívají hranové filtry, které se dělí do několika skupin. Základní skupiny tvoří gradientní metody hledání hran, označované také jako metody první derivace, což jsou detektory založené na aproximaci maximálních hodnot první derivace obrazové funkce, znázorněné na obr. 3.3c), a metody druhé derivace spočívající v hledání průchodů druhé derivace nulou, viz obr. 3.3d) (Sojka et al., 2011; Russ a Russ, 2008). Další skupinu tvoří postupy na bázi evolučních algoritmů (Kavitha a Rajeswari, 2014; Mousavi et al., 2019).



Obrázek 3.3: a) Ilustrace přechodu jasu, b) znázornění obrazové funkce přechodu jasu, c) první derivace obrazové funkce, d) druhá derivace obrazové funkce

3.2 Hledání hran na základě první derivace

Gradientní operátory využívané k detekci hran spočívají v aproximaci výsledků první derivace obrazové funkce, kdy jejich hodnoty v oblasti hran jsou vysoké. Gradient vyjadřující změnu hodnot obrazové funkce $f(x, y)$ je definován jako vektor

$$\nabla f(x, y) = \begin{bmatrix} e_x(x, y) \\ e_y(x, y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{bmatrix}. \quad (3.1)$$

Důležitou vlastností gradientu, zásadní pro hledání hran, je jeho velikost, která udává nejvyšší míru nárůstu funkce na jednotku vzdálenosti ve směru gradientu, a je určena vztahem

$$E(x, y) = |\nabla f(x, y)| = \sqrt{e_x(x, y)^2 + e_y(x, y)^2}. \quad (3.2)$$

Velikost gradientu lze také aproximovat součtem absolutních hodnot jeho prvků či jejich maximální hodnotou, a tak platí

$$E(x, y) \approx |e_x(x, y)| + |e_y(x, y)| \quad (3.3)$$

$$E(x, y) \approx \max(|e_x(x, y)|, |e_y(x, y)|). \quad (3.4)$$

Směr gradientu, jenž je kolmý na směr hrany, je dán jako

$$\alpha(x, y) = \arctan\left(\frac{e_y(x, y)}{e_x(x, y)}\right), \quad (3.5)$$

úhel α je brán jako úhel mezi směrem gradientu a osou x (Nixon a Aguado, 2002; Jain et al., 1995). Vzhledem k tomu, že je obrazová funkce $f(x, y)$, $x = 0, 1, \dots, M - 1$, $y = 0, 1, \dots, N - 1$, reálného obrazu nespojitá, změny v intenzitě jasu mohou být vypočítání diferencí sousedních bodů, takže

$$e_x(x, y) = f(x + 1, y) - f(x, y), \quad (3.6)$$

$$e_y(x, y) = f(x, y + 1) - f(x, y). \quad (3.7)$$

Výpočet hodnot e_x a e_y pro všechny body obrazu může být realizován pomocí konvoluce obrazu s kernely $g_x = [1 \quad -1]$ ve směru osy x a $g_y = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ ve směru osy y , tedy

$$e_x(x, y) = f(x, y) * g_x, \quad (3.8)$$

$$e_y(x, y) = f(x, y) * g_y. \quad (3.9)$$

K potlačení méně výrazných hran lze stanovit prahovou hodnotu, výsledný pixel bude zobrazen, pokud je jeho hodnota větší než prahová hodnota, v případě menší hodnoty pixelu mu je přidělena hodnota 0, dochází k jeho nahrazení černou barvou. K základním operátorům detekce hran se řadí Robertsův, Sobelův, Robinsonův, Kirschův operátor a operátor Prewittové (Petrou a Petrou, 2010; Sojka, 2000; Sonka et al., 2008).

3.2.1 Robertsův operátor

Robertsův operátor je nejstarším hranovým detektorem. K detekci hran využívá dvou konvolučních masek, jejichž konvoluce s obrazem je ekvivalentní k diferenci ve dvou na sebe kolmých směrech (Gonzales a Woods, 2002).

$$\begin{array}{cc} \text{a)} & \text{b)} \\ g_x = \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array} & g_y = \begin{array}{|c|c|} \hline 0 & -1 \\ \hline 1 & 0 \\ \hline \end{array} \end{array}$$

Obrázek 3.4: Hodnoty konvolučních masek Robertsova operátoru pro a) detekci horizontálních hran, b) detekci vertikálních hran

Velikost gradientu je určena sumou absolutních hodnot prvků gradientu (Jain et al., 1995; Hlaváč a Sedláček, 2009)

$$E(x, y) = |e_x(x, y)| + |e_y(x, y)|, \quad (3.10)$$

po dosazení hodnot $e_x(x, y)$ a $e_y(x, y)$ z rovnic 3.8 a 3.9 získáme

$$E(x, y) = |f(x, y) * g_x| + |f(x, y) * g_y|, \quad (3.11)$$

konvoluci s uvedenými konvolučními maskami můžeme dále rozepsat jako

$$E(x, y) = |f(x, y) - f(x + 1, y + 1)| + |f(x, y + 1) - f(x + 1, y)|. \quad (3.12)$$

Dle Nixon a Aguado (2002) lze velikost gradientu při aplikaci Robertsova operátoru stanovit jako maximální hodnotu prvku gradientu

$$E(x, y) = \max\{|e_x(x, y)|, |e_y(x, y)|\}. \quad (3.13)$$

3.2.2 Operátor Prewittové

Tento operátor funguje na základě aproximace první derivace pro okolí obrazového bodu o velikosti 3×3 . Gradient lze odhadovat v osmi směrech, tudíž je třeba provést konvoluci obrazu s osmi kernely a poté je pro každý bod vybrán výsledek konvoluce s nejvyšší hodnotou, avšak k detekci vodorovných a svislých hran se běžně používají pouze dvě konvoluční masky korespondující se směrem osy x a osy y (Hlaváč a Sedláček, 2009; Karasulu, 2012).

$$\begin{array}{cc} \text{a)} & \text{b)} \\ g_x = & g_y = \\ \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array} & \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \end{array}$$

Obrázek 3.5: Hodnoty konvolučních masek operátoru Prewittové pro a) detekci horizontálních hran, b) detekci vertikálních hran

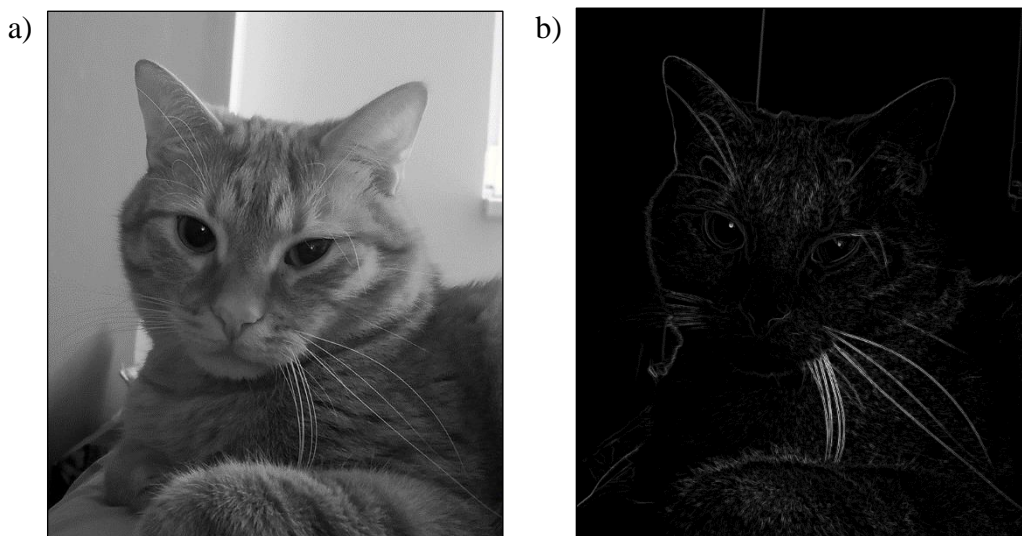
Velikost je v případě dvou kernelů stanovena jako

$$E(x, y) = \sqrt{e_x(x, y)^2 + e_y(x, y)^2}, \quad (3.14)$$

směr gradientu je vypočítán ze vzorce

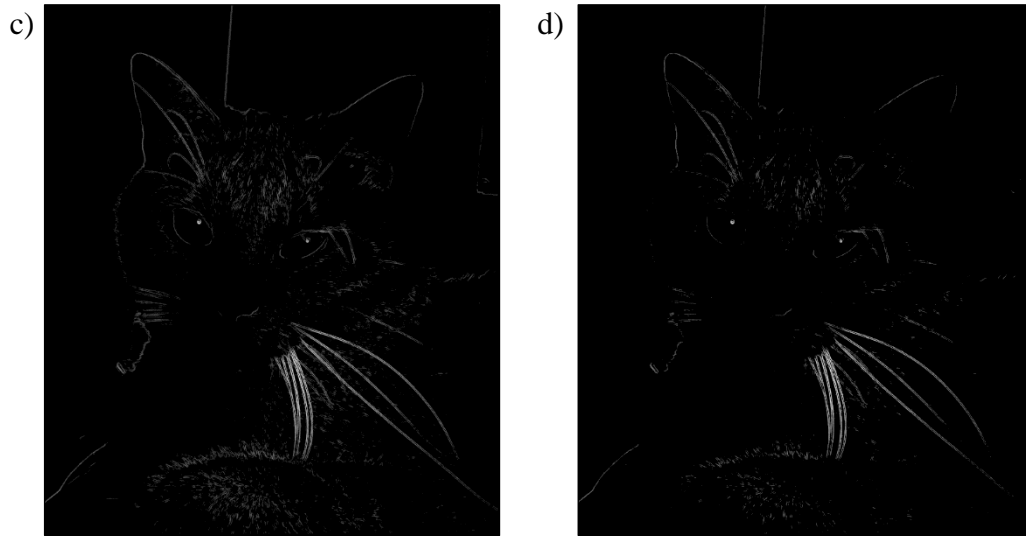
$$\alpha(x, y) = \arctan\left(\frac{e_y(x, y)}{e_x(x, y)}\right), \quad (3.15)$$

v případě osmi kernelů je směr gradientu určen dle směru kernelu, jehož konvolucí s obrazem vznikla nejvyšší hodnota (Nema a Saxena, 2013).



pokračování obrázku na další stránce

pokračování obrázku z předchozí stránky



Obrázek 3.6: Obraz po aplikaci operátoru Prewittové a) vstupní obraz, b) výsledný obraz pro práh o hodnotě 10, c) výsledný obraz pro práh o hodnotě 30, d) výsledný obraz pro práh o hodnotě 50

3.2.3 Sobelův operátor

Jedním z nejznámějších operátorů pro hledání hran je Sobelův operátor, který má značnou podobnost s operátorem Prewittové, vychází také z aproximace první derivace realizované pomocí konvoluce kernelu o velikosti 3×3 s obrazem, velikost i směr gradientu jsou určeny dle stejných vzorců, také ho lze aplikovat k detekci gradientu v osmi směrech, stejně tak se nejčastěji používá pro nalezení vodorovných a svislých hran, avšak nabízí lepší detekční schopnosti (Nixon a Aguado, 2002; Russ, 2002). Na obrázku 3.7 jsou znázorněny konvoluční masky pro detekci vodorovných a svislých hran.

$$\begin{array}{cc}
 \text{a)} & \text{b)} \\
 g_x = & g_y = \\
 \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} & \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}
 \end{array}$$

Obrázek 3.7: Hodnoty konvolučních masek Sobelova operátoru pro a) detekci horizontálních hran, b) detekci vertikálních hran

Jain et al. (1995) a Sojka et al. (2011) uvádějí, že parciální derivace v poloze každého pixelu s okolím 3×3 pixelů lze vypočítat ze vztahu

$$e_x(x, y) = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7), \quad (3.16)$$

$$e_y(x, y) = (z_1 + 2z_2 + z_3) - (z_7 + 2z_8 + z_9), \quad (3.17)$$

parametry z_1, \dots, z_9 představují polohu okolních pixelů vůči vybranému pixelu, viz obrázek 3.8.

z_1	z_2	z_3
z_4	$f(x, y)$	z_6
z_7	z_8	z_9

Obrázek 3.8: Znázornění značení okolních bodů vzhledem k vybranému bodu



Obrázek 3.9: Obraz po aplikaci Sobelova operátoru, a) vstupní obraz, b) výsledný obraz pro práh o hodnotě 10, c) výsledný obraz pro práh o hodnotě 30, d) výsledný obraz pro práh o hodnotě 50

3.2.4 Robinsonův operátor

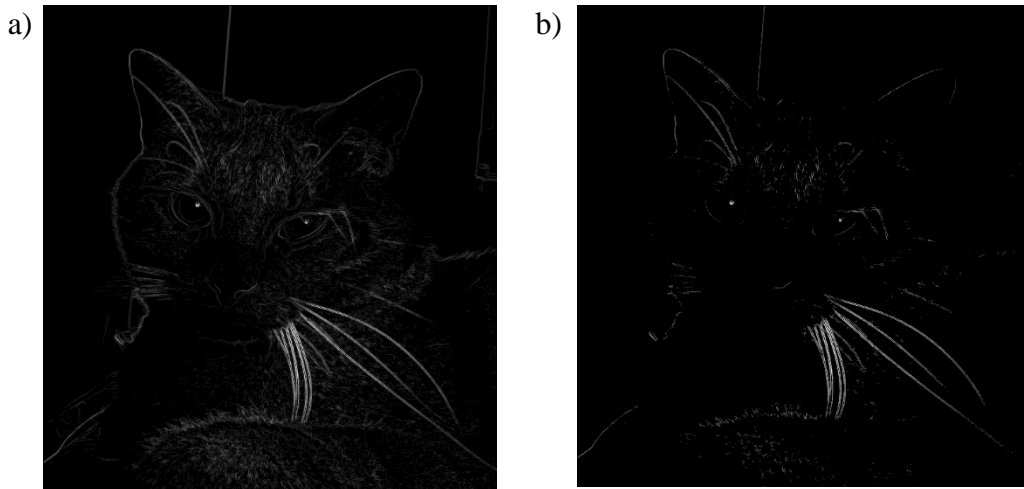
Robinsonův operátor vychází z použití konvolučních masek o velikosti 3×3 orientovaných do osmi směrů, které jsou také označovány jako kompasové masky, směry masek jsou pojmenovány dle světových stran, tedy sever, severovýchod, východ, jihovýchod, jih, jihozápad, západ a severozápad. Jednotlivé masky vznikly rotací o 45° kolem jejich středového bodu (Sonka et al., 2008).

a)	b)	c)	d)																																				
$g_1 =$	$g_2 =$	$g_3 =$	$g_4 =$																																				
<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>-2</td><td>1</td></tr><tr><td>-1</td><td>-1</td><td>-1</td></tr></table>	1	1	1	1	-2	1	-1	-1	-1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>-1</td><td>-2</td><td>1</td></tr><tr><td>-1</td><td>-1</td><td>1</td></tr></table>	1	1	1	-1	-2	1	-1	-1	1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>-1</td><td>1</td><td>1</td></tr><tr><td>-1</td><td>-2</td><td>1</td></tr><tr><td>-1</td><td>1</td><td>1</td></tr></table>	-1	1	1	-1	-2	1	-1	1	1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>-1</td><td>-1</td><td>1</td></tr><tr><td>-1</td><td>-2</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	-1	-1	1	-1	-2	1	1	1	1
1	1	1																																					
1	-2	1																																					
-1	-1	-1																																					
1	1	1																																					
-1	-2	1																																					
-1	-1	1																																					
-1	1	1																																					
-1	-2	1																																					
-1	1	1																																					
-1	-1	1																																					
-1	-2	1																																					
1	1	1																																					
e)	f)	g)	h)																																				
$g_5 =$	$g_6 =$	$g_7 =$	$g_8 =$																																				
<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>-1</td><td>-1</td><td>-1</td></tr><tr><td>1</td><td>-2</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	-1	-1	-1	1	-2	1	1	1	1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>1</td><td>-1</td><td>-1</td></tr><tr><td>1</td><td>-2</td><td>-1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	-1	-1	1	-2	-1	1	1	1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>1</td><td>1</td><td>-1</td></tr><tr><td>1</td><td>-2</td><td>-1</td></tr><tr><td>1</td><td>1</td><td>-1</td></tr></table>	1	1	-1	1	-2	-1	1	1	-1	<table border="1" style="display: inline-table; border-collapse: collapse;"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>-2</td><td>-1</td></tr><tr><td>1</td><td>-1</td><td>-1</td></tr></table>	1	1	1	1	-2	-1	1	-1	-1
-1	-1	-1																																					
1	-2	1																																					
1	1	1																																					
1	-1	-1																																					
1	-2	-1																																					
1	1	1																																					
1	1	-1																																					
1	-2	-1																																					
1	1	-1																																					
1	1	1																																					
1	-2	-1																																					
1	-1	-1																																					

Obrázek 3.10: Hodnoty konvolučních masek Robinsonova operátoru pro a) směr sever, b) severovýchod, c) východ, d) jihovýchod, e) jih, f) jihozápad, g) západ, h) severozápad

Velikost gradientu je rovna nejvyšší hodnotě vzniklé při konvoluci každé z osmy konvolučních masek s obrazem, směr je určen orientací masky s nejvyšší výslednou hodnotou (Burnham et al., 1997; Zhang, 2021).

$$E(x, y) = \max\{|e_1(x, y)|, \dots, |e_8(x, y)|\} \quad (3.18)$$



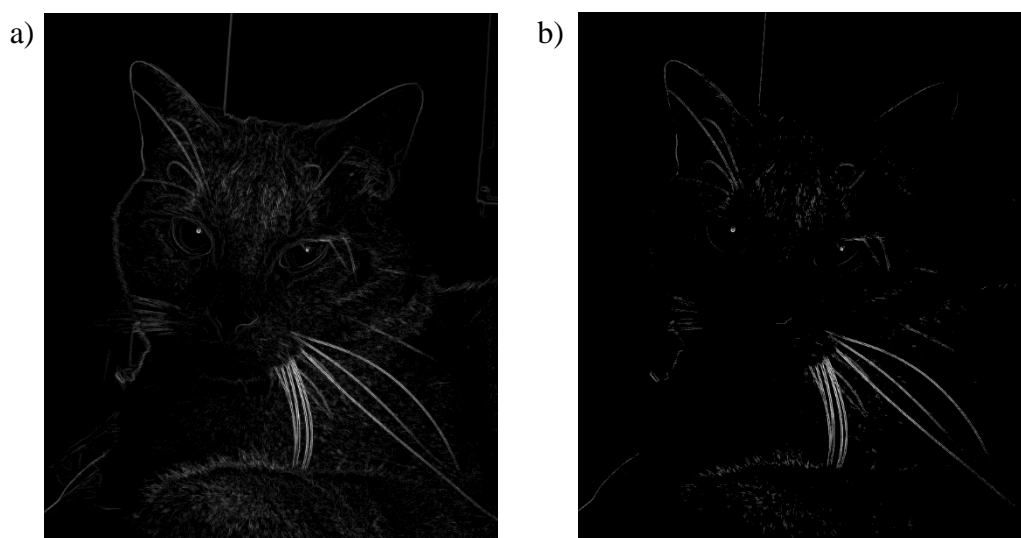
Obrázek 3.11: Obraz po aplikaci Robinsonova operátoru, a) výsledný obraz pro práh o hodnotě 10, b) výsledný obraz pro práh o hodnotě 50

3.2.5 Kirschův operátor

Tato metoda se zakládá na stejném principu jako Robinsonův operátor, využívá tedy konvoluce obrazu s rotačními maskami o osmi směrech odpovídajících rotaci o 45°, viz obr. 3.12, určení velikosti gradientu a jeho směru je také totožné (Kasulu, 2013; Russ, 2002).

$$\begin{array}{cccc}
 \text{a)} & & \text{b)} & & \text{c)} & & \text{d)} \\
 g_1 = & \begin{array}{|c|c|c|} \hline -3 & -3 & -3 \\ \hline -3 & 0 & -3 \\ \hline 5 & 5 & 5 \\ \hline \end{array} & g_2 = & \begin{array}{|c|c|c|} \hline -3 & -3 & -3 \\ \hline 5 & 0 & -3 \\ \hline 5 & 5 & -3 \\ \hline \end{array} & g_3 = & \begin{array}{|c|c|c|} \hline 5 & -3 & -3 \\ \hline 5 & 0 & -3 \\ \hline 5 & -3 & -3 \\ \hline \end{array} & g_4 = & \begin{array}{|c|c|c|} \hline 5 & 5 & -3 \\ \hline 5 & 0 & -3 \\ \hline -3 & -3 & -3 \\ \hline \end{array} \\
 \\
 \text{e)} & & \text{f)} & & \text{g)} & & \text{h)} \\
 g_5 = & \begin{array}{|c|c|c|} \hline 5 & 5 & 5 \\ \hline -3 & 0 & -3 \\ \hline -3 & -3 & -3 \\ \hline \end{array} & g_6 = & \begin{array}{|c|c|c|} \hline -3 & 5 & 5 \\ \hline -3 & 0 & 5 \\ \hline -3 & -3 & -3 \\ \hline \end{array} & g_7 = & \begin{array}{|c|c|c|} \hline -3 & -3 & 5 \\ \hline -3 & 0 & 5 \\ \hline -3 & -3 & 5 \\ \hline \end{array} & g_8 = & \begin{array}{|c|c|c|} \hline -3 & -3 & -3 \\ \hline -3 & 0 & 5 \\ \hline -3 & 5 & 5 \\ \hline \end{array}
 \end{array}$$

Obrázek 3.12: Hodnoty konvolučních masek Kirschova operátoru pro a) směr sever, b) severovýchod, c) východ, d) jihovýchod, e) jih, f) jihozápad, g) západ, h) severozápad



Obrázek 3.13: Obraz po aplikaci Kirschova operátoru, a) výsledný obraz pro práh o hodnotě 10, b) výsledný obraz pro práh o hodnotě 50

3.2.6 Cannyho hranový detektor

Jedním z pokročilých hranových operátorů fungujících na základě první derivace je Cannyho hranový detektor, který aproximuje funkci optimalizující poměr mezi signálem a šumem a zajišťující přesnou lokalizaci hran. Návrh detektoru vychází ze tří kritérií, prvním je detekční kritérium požadující optimální detekci hran s minimem chybných výsledků, druhé kritérium, lokalizační, klade důraz na maximální přesnost

polohy nalezené hrany vzhledem k její reálné pozici, třetí podmínkou je kritérium jedné odezvy, v rámci kterého se jedná o eliminaci vícenásobné odezvy detektoru na jednu hranu vedoucí k jednoznačnou identifikaci hrany (Hlaváč a Sedláček, 2009; Nixon a Aguado, 2002). Hlavní myšlenkou je vytvoření operátoru představujícího derivaci Gaussiánu ve směru gradientu. V případě hran s orientací ve směru osy x a ve směru osy y lze popsat první derivace konvoluce Gaussovy funkce s obrazem (Petrou a Petrou, 2010; Sojka et al., 2011) jako

$$\nabla h(x, y) = \nabla [f(x, y) * G(x, y)], \quad (3.19)$$

dochází tedy k derivaci vyhlazené obrazu s potlačeným šumem, jejíž výsledné hodnoty budou

$$e_x(x, y) = \frac{\partial [f(x, y) * G(x, y)]}{\partial x}, \quad (3.20)$$

$$e_y(x, y) = \frac{\partial [f(x, y) * G(x, y)]}{\partial y}. \quad (3.21)$$

K výpočtu hodnot derivace nespojitě funkce, kterou je i obraz, lze provést metodou konečných diferencí (Jain et al., 1995). Liu et al. (2018) a Tong et al. (2018) uvádí, že je tento krok možné realizovat konvolucí se Sobelovým operátorem, tedy pokud za S_x označíme konvoluční masku pro detekci vodorovných hran a za S_y kernel k detekci svislých hran, tak složky vektoru gradientu můžeme určit dle vztahu

$$e_x(x, y) = \frac{\partial [f(x, y) * G(x, y)]}{\partial x}, \quad (3.22)$$

$$e_y(x, y) = \frac{\partial [f(x, y) * G(x, y)]}{\partial y}. \quad (3.23)$$

velikost gradientu je dána vztahem

$$E(x, y) = \sqrt{e_x(x, y)^2 + e_y(x, y)^2} \quad (3.24)$$

a směr gradientu je určen

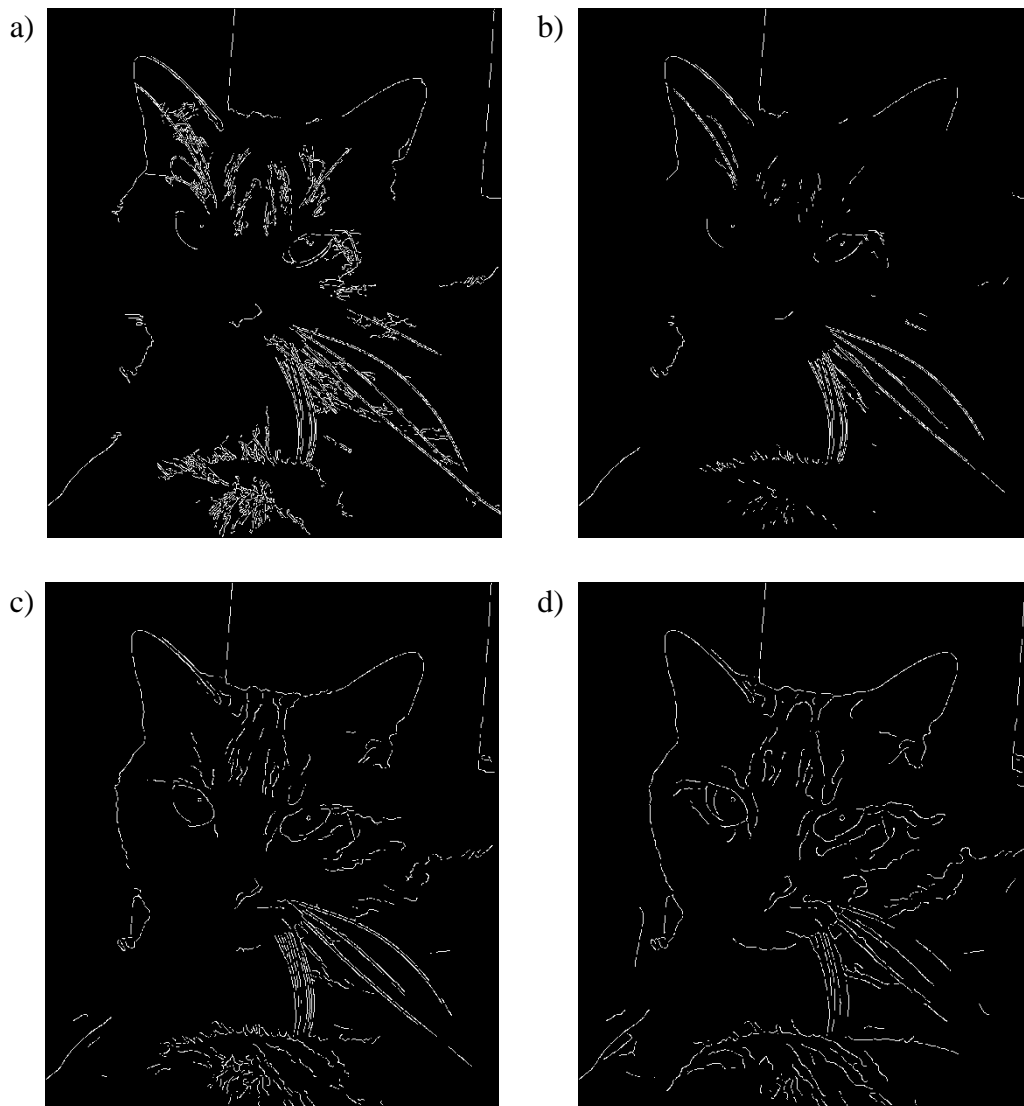
$$\alpha(x, y) = \arctan\left(\frac{e_y(x, y)}{e_x(x, y)}\right). \quad (3.25)$$

Dalším krokem hledání hran je potlačení odezev mimo maxima, při kterém dochází k zúžení hran a nespojitostí (Russ a Russ, 2008). Hodnota gradientu ve všech bodech je posuzována vzhledem hodnotám bodů z osmiokolí dle směru gradientu. Směry gradientů v jednotlivých bodech jsou rozřazeny dle následujících intervalů zastupujících čtyři základní směry, což jsou horizontální, vertikální a dva diagonální, které určují výběr sousedních bodů k porovnání hodnot. Posuzovaný bod zůstává nezměněn, je-li jeho hodnota větší než hodnota sousedních bodů v určeném směru, v případě nesplnění této podmínky nabývá bod hodnoty 0 (Bugarinović et al., 2020; Sonka et al., 2008; Tabassum et al., 2013).

$$\begin{aligned}
 \alpha(x, y) \in \langle 0^\circ; 22,5^\circ \rangle \cup \langle 135^\circ; 180^\circ \rangle &\Rightarrow \begin{aligned} E(x-1, y) &< E(x, y) \\ E(x+1, y) &< E(x, y) \end{aligned} \\
 \alpha(x, y) \in \langle 22,5^\circ; 45^\circ \rangle &\Rightarrow \begin{aligned} E(x+1, y+1) &< E(x, y) \\ E(x-1, y-1) &< E(x, y) \end{aligned} \\
 \alpha(x, y) \in \langle 45^\circ; 90^\circ \rangle &\Rightarrow \begin{aligned} E(x, y+1) &< E(x, y) \\ E(x, y-1) &< E(x, y) \end{aligned} \\
 \alpha(x, y) \in \langle 90^\circ; 135^\circ \rangle &\Rightarrow \begin{aligned} E(x-1, y+1) &< E(x, y) \\ E(x+1, y-1) &< E(x, y) \end{aligned}
 \end{aligned}$$

Potlačení odezev mimo maxima je také možné realizovat s porovnáním bodů ležících přesně ve směru gradientu, jejichž hodnoty jsou vypočítány interpolací z hodnot dvojice jejich sousedních bodů (Sojka et al., 2011; Zhang, 2021).

V poslední části postupu je provedeno prahování s hysterezí, při kterém dochází k výběru dvou odlišných prahových hodnot, kdy je vyšší hodnota určena jako maximální práh T_{\max} a nižší jako minimální práh T_{\min} . Všechny body obrazu jsou s prahovými hodnotami porovnány, pokud je jejich hodnota vyšší nebo rovna T_{\max} , jsou uznány jako bod hrany, body jejichž hodnota je v rozmezí mezi T_{\min} a T_{\max} a zároveň sousedí s bodem hrany, jsou také přiřazeny do skupiny hranových bodů. Body, které předchozí podmínky nesplní, jsou ve výstupním obrazu nahrazeny hodnotou 0. Prahování s hysterezí se v rámci hledání hran může vícekrát opakovat (Liu et al., 2018; Zhang, 2021).



Obrázek 3.14: Obrázek 3.15: Obrázek po aplikaci Cannyho operátoru, a) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=1$, b) výsledný obraz pro dolní a horní práh o hodnotách 50 a 80, $\sigma=1$ c) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=2$, d) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=3$

3.3 Hledání hran na základě druhé derivace

Standardní gradientní operátory popsané výše se potýkají s jistou citlivostí na šum. Využití druhé derivace ke hledání hran je příhodnější díky vyšší spolehlivosti, jež je dána strmostí druhé derivace funkce v okolí jejího průchodu nulou. Základní operátor využívající druhou derivaci, Laplacián, popsaný v kapitole 1, se však potýká se šumem způsobujícím detekci falešných hran v ještě větší míře. Dalším operátorem uplatňujícím druhou derivaci je Laplacián Gaussiánu umožňující zmíněnou slabinu minimalizovat (Hlaváč a Sedláček, 2009; Sojka et al., 2011).

3.3.1 LoG filtr

V roce 1980 představily Marr a Hildreth postup, se kterým lze negativní vliv šumu potlačit implementací Gaussova vyhlazovacího filtru, tedy Laplacián Gausiánu, zkratka LoG je z anglického Laplacian of Gaussian (Gironés a Julià, 2020; Marr a Hildreth, 1980; Zhang 2021). Operátor vychází z druhé derivace konvoluce obrazu s Gaussovým filtrem, tedy

$$\nabla^2 h(x, y) = \nabla^2 [f(x, y) * G(x, y)], \quad (3.26)$$

po záměně pořadí aplikace druhé derivace a konvoluce, kterou lze provést z důvodu linearit obou operací, vznikne

$$\nabla^2 h(x, y) = [\nabla^2 G(x, y)] * f(x, y). \quad (3.27)$$

Gaussův dvourozměrný filtr je definován jako

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \quad (3.28)$$

a předpis Laplaceova operátoru je

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \quad (3.29)$$

K získání vztahu pro LoG operátor je třeba provést součet druhých derivací funkce Gaussova filtru dle x a y

$$\nabla^2 G(x, y) = \frac{\partial^2 G(x, y)}{\partial x^2} + \frac{\partial^2 G(x, y)}{\partial y^2}, \quad (3.30)$$

$$\nabla^2 G(x, y) = \frac{x^2 - \sigma^2}{2\pi\sigma^6} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} + \frac{y^2 - \sigma^2}{2\pi\sigma^6} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}, \quad (3.31)$$

odtud získáme

$$\nabla^2 G(x, y) = \frac{x^2 + y^2 - 2\sigma^2}{2\pi\sigma^6} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}, \quad (3.32)$$

po úpravě dostaneme výslednou rovnici

$$\nabla^2 G(x, y) = \left(-\frac{1}{\pi\sigma^4}\right) \left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}. \quad (3.33)$$

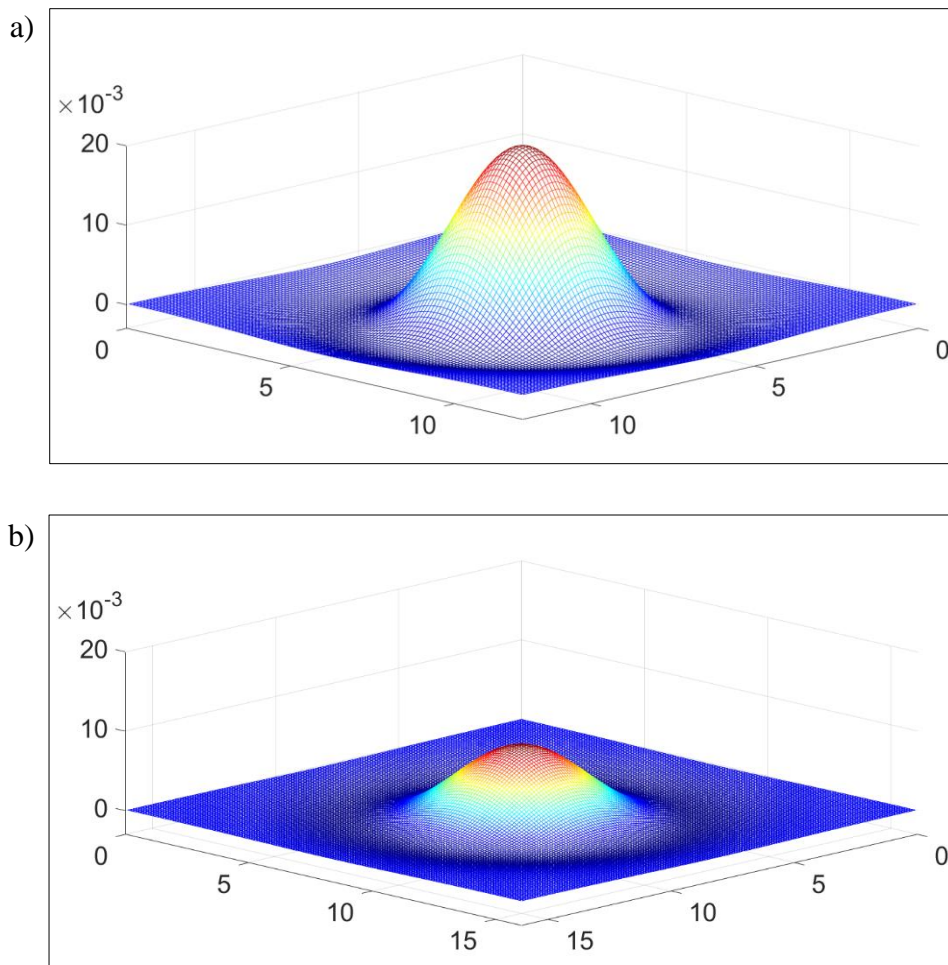
V literatuře se také vyskytuje vyjádření LoG pomocí vztahu

$$\nabla^2 G(x, y) = \frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}, \quad (3.34)$$

rozdíl mezi vyjádřeními LoG dle rovnic 3.33 a 3.34 pramení z využití Gaussovy funkce ve tvaru

$$G(x, y) = e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \quad (3.35)$$

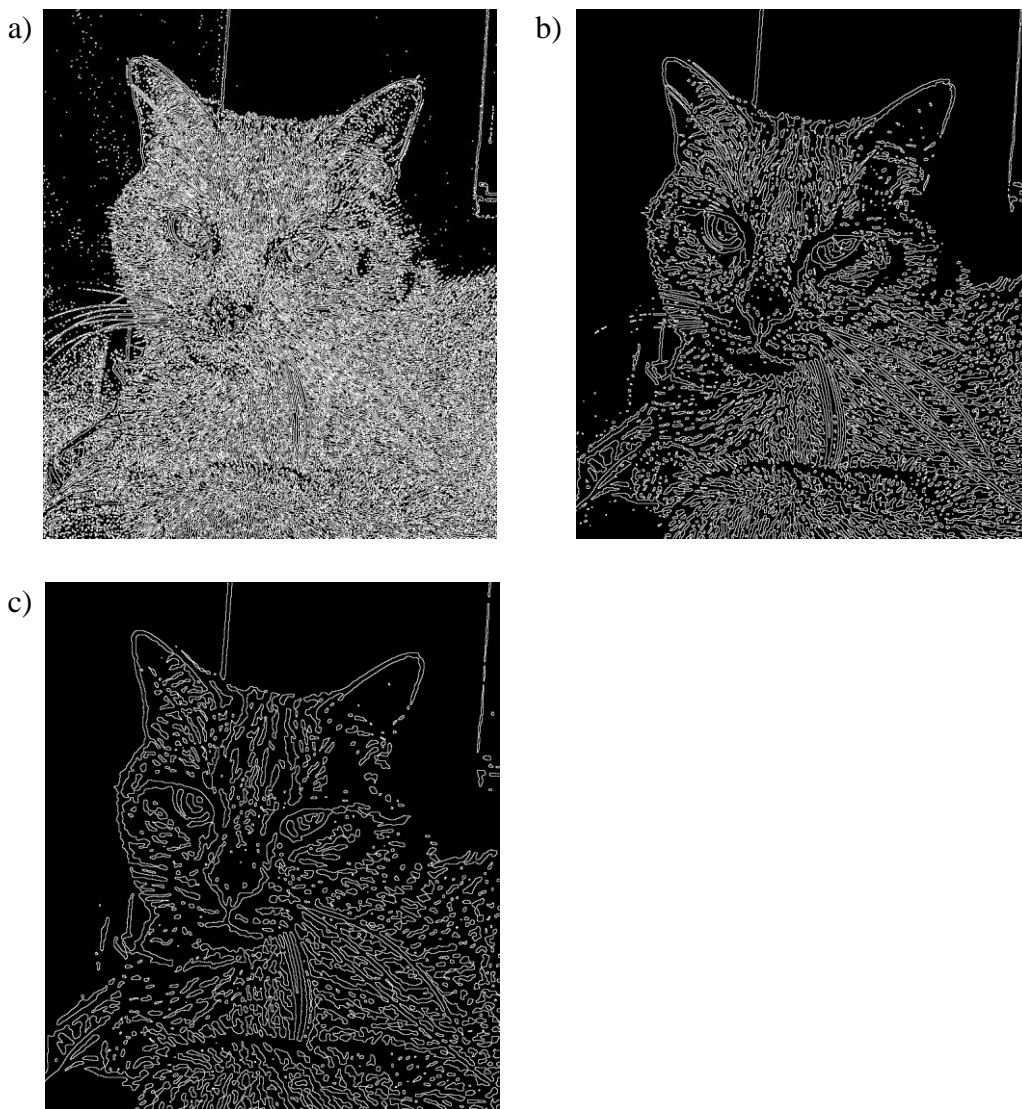
případě vztahu 3.34 místo předpisu Gaussova normálního rozdělení (Gonzales et al., 2009; Jain et al., 1995; Nixon a Aguado, 2002).



Obrázek 3.16: Graf konvoluční masky LoG filtru, kdy a) $\sigma=2$, b) $\sigma=2,5$

Po provedení konvoluce obrazu s kernelem operátoru LoG je třeba provést hledání průchodu nulou, protože pixel náleží hraně, pokud mezi ním a sousedními pixely dojde ke změně hodnot z kladných na záporné či naopak. Detektor průchodu nulou může mít například podobu masky o rozměrech 2×2 , kdy je zvolený reprezentativní bod s neměnnou polohou vůči masce, k detekci hrany dochází při změně znaménka uvnitř

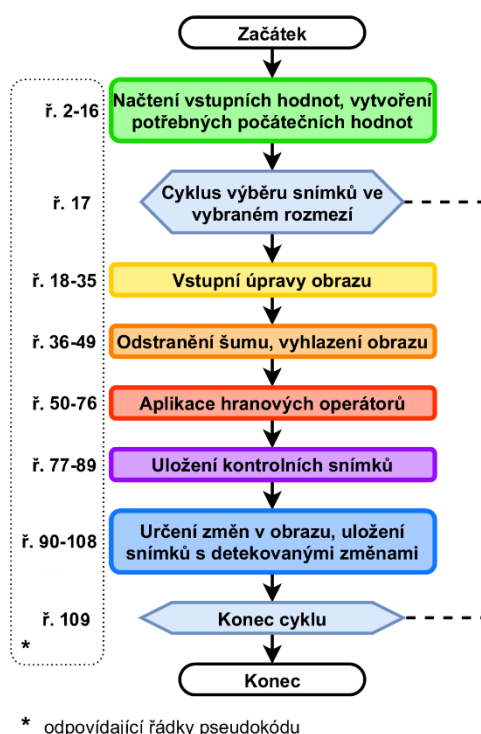
masky vůči reprezentativnímu bodu (Sonka et al., 2008). Marr a Hildreth (1980) před detekcí průchodu nulou vytvoří z výsledku konvoluce binární obraz přidělením bílé barvy, tedy hodnoty 255, kladným hodnotám a černé barvy hodnotám záporným, hodnotám nulové se nijak nemění, poté je třeba detekovat mezní pixely, v jejichž bezprostřední blízkosti dochází ke změně hodnot. Petrou a Petrou (2010) popisují postup hledání průchodu nulou, který je rozdělen na hledání ve směru osy x , po řádcích, a osy y , po sloupcích, kdy se pro každý bod (x, y) z intervalu $x \in \langle 1; M - 1 \rangle$, $y \in \langle 1; N - 1 \rangle$ výsledného obrazu o velikosti $M \times N$ provádí součin s bodem následujícím dle směru, tedy $(x + 1, y)$ nebo $(x, y + 1)$. Když je výsledek součinu menší nebo rovno nule, tak bod (x, y) náleží hraně a do nového výstupního obrazu je mu dána hodnota 0, jinak získá hodnotu 255. Při standardní prezentaci nalezených hran, kdy hrany mají bílou barvu a ostatní pixely černou, jsou hodnoty přiděleny naopak.



Obrázek 3.17: Obraz po aplikaci operátoru LoG, a) výsledný obraz pro $\sigma=1$, b) výsledný obraz pro $\sigma=2$, c) výsledný obraz pro $\sigma=3$

4 Návrh algoritmu a implementace vybraných operátorů

Algoritmus pro rozpoznávání změn v obrazu se skládá z několika hlavních částí, z nichž první je zaměřena na získání vstupních hodnot a vygenerování parametrů či prázdných množin potřebných pro jeho správné fungování. Činnost dalších úseků orientovaných na úpravy a porovnání jednotlivých snímků videosouboru je cyklická, počet jejich opakování se odvíjí z celkového počtu snímků videa, snímkové frekvenci a zvoleném časovém rozestupu mezi jednotlivými snímky. Vstupní úpravy obrazu zahrnují načtení snímku s daným pořadím, jeho úpravu z barevného prostoru RGB na úrovně šedi a případně zvolené zmenšení obrazu. Následuje možnost odstranění šumu vyhlazením obrazu pomocí Gaussova filtru s velikostí masky určenou automaticky dle hodnoty σ , či s její velikostí přesně definovanou uživatelem. V dalším kroku dochází k hledání hran aplikací jednoho ze šesti v algoritmu dostupných operátorů. V rámci navazujícího oddílu jsou ukládány kontrolní snímky v původní nebo upravené podobě do složky vytvořené ve zvoleném adresáři během první fáze. Finální etapa se zabývá určením změn v obrazu na základě porovnání dvou snímků s rozmezím od 1 do 20 snímků či aktuálního snímku s prvním snímkem videa, kdy je vyhodnocen relativní rozdíl jejich hodnot, přičemž změna je detekována v případě překročení předem definované meze, již je vhodné zkalibrovat na krátkém úseku zkoumaného videa.



Obrázek 4.1: Základní schéma algoritmu na detekci změn v obrazu

Základní zjednodušené schéma je znázorněno na obrázku 4.1, samostatné etapy, řazené v posloupnosti dle průběhu programu, jsou odlišeny barvami odpovídajícími zvýrazněným celkům ve vývojovém diagramu obsahujícím podrobný rozpis algoritmu za využití příkazů a syntaxe jazyka programu MATLAB, v jehož prostředí byly skripty vyvíjeny. Čísla řádků uvedená u každé části korespondují s náležitými částmi uvedeného pseudokódu. Silně vyznačené podprogramy ve vývojovém diagramu na obrázku 4.2 a 4.3, což jsou funkce obrazových filtrů, jejichž ekvivalenty v podobě nativních MATLAB funkcí jsou dostupné pouze v rozšiřující knihovně zaměřené na zpracování obrazu, budou dále přiblíženy pomocí pseudokódu a stručně popsány.

Do algoritmu nebyly začleněny filtry z frekvenční domény z důvodu vyšší časové náročnosti oproti konvoluci při filtraci obrazu v programu MATLAB. Pro konvoluci i rychlou Fourierovu transformaci jsou v jeho základní verzi implementovány funkce *conv2(obraz, kernel)* a *fft2(obraz)*. Porovnání času potřebného pro realizaci konvoluce obrazů o pěti rozdílných rozlišeních s konvolučními maskami o velikostech 3×3 až 21×21 a úprav v rámci frekvenční domény za využití FFT na stejné obrazy je uvedeno v sekundách v tabulce 4.1. K zajištění relevantnosti časových údajů bylo každé měření provedené stokrát, uvedený výsledný čas byl určen aritmetickým průměrem naměřených hodnot.

Tabulka 4.1: Porovnání časové náročnosti filtrace v prostorové a frekvenční doměně [s]

Velikost konvoluční masky	Rozlišení obrazu				
	256×256	512×512	1024×1024	2048×2048	4096×4096
3×3	0,0004	0,0012	0,0026	0,0093	0,0335
5×5	0,0003	0,0012	0,0027	0,0098	0,0358
7×7	0,0004	0,0013	0,0029	0,0105	0,0386
9×9	0,0005	0,0014	0,0030	0,0110	0,0432
11×11	0,0006	0,0013	0,0037	0,0122	0,0504
13×13	0,0008	0,0016	0,0042	0,0143	0,0599
15×15	0,0010	0,0019	0,0045	0,0168	0,0700
17×17	0,0012	0,0022	0,0052	0,0192	0,0823
19×19	0,0014	0,0025	0,0048	0,0231	0,0971
21×21	0,0017	0,0026	0,0065	0,0261	0,1137
FFT	0,0024	0,0069	0,0274	0,0935	0,3471

ALGORITMUS PRO NALEZENÍ ZMĚNY V OBRAZU

```

1  začátek
2  čti (cesta_otevrit, cesta_ulozit, nazev, t, zm, p, lin_filt,
3     a, a_aut, σ, det, typ_d, prah, prah_H, r0, k_s_u0, k_s_u1) ;
4  OTEVRI_ADRESAR (cesta_otevrit) ;
5  VYTVOR_ADRESAR ('kontrolni_snimky_', nazev) ;
6  vid ← NACTI_VIDEO (cesta_otevrit) ;
7  fr ← SNIMKOVA_FREKVENCE (vid) ;
8  numfr ← POČET_SNIMKU (vid) ;
9  [sloupce, radky] ← ROZLIŠENI (vid) ;
10 ppix ← radky · sloupce ;
11 s ← t · fr ;
12 jestliže p ≥ 1 pak
13   začátek
14    $B \leftarrow O_{\left\lfloor \frac{radky}{z_m} \right\rfloor \times \left\lfloor \frac{sloupce}{z_m} \right\rfloor \times (p+1)}$  ;
15   jinak
16    $B \leftarrow O_{\left\lfloor \frac{radky}{z_m} \right\rfloor \times \left\lfloor \frac{sloupce}{z_m} \right\rfloor \times 2}$  ;
17   konec
18   pro i od 1 po s do numfr
19   začátek
20    $j \leftarrow \frac{i-1}{s} + 1$  ;
21   jestliže p = 0 ∧ i = 2 pak
22   začátek
23    $B[:, :, 1] \leftarrow I_z$  ;
24   konec
25   jestliže p ≥ 1 ∧ i > 1 pak
26   začátek
27    $B[:, :, 1:p] \leftarrow B[:, :, 2:p + 1]$ 
28   konec
29   snimek ← PRECTI_SNIMEK_VIDEA (vid, i) ;
30   jestliže zm > 1 pak
31   začátek
32    $I_z \leftarrow ZMENSENI (snimek, z_m)$  ;
33   jinak
34    $I_z \leftarrow snimek$  ;
35   konec
36    $I_{z0} \leftarrow I_z$  ;
37   jestliže lin_filt = 1 pak
38   začátek
39   jestliže det = 1 ∧ (typ_d = 5 ∨ typ_d = 6) pak
40   začátek
41    $I_z \leftarrow I_z$  ;

```

```

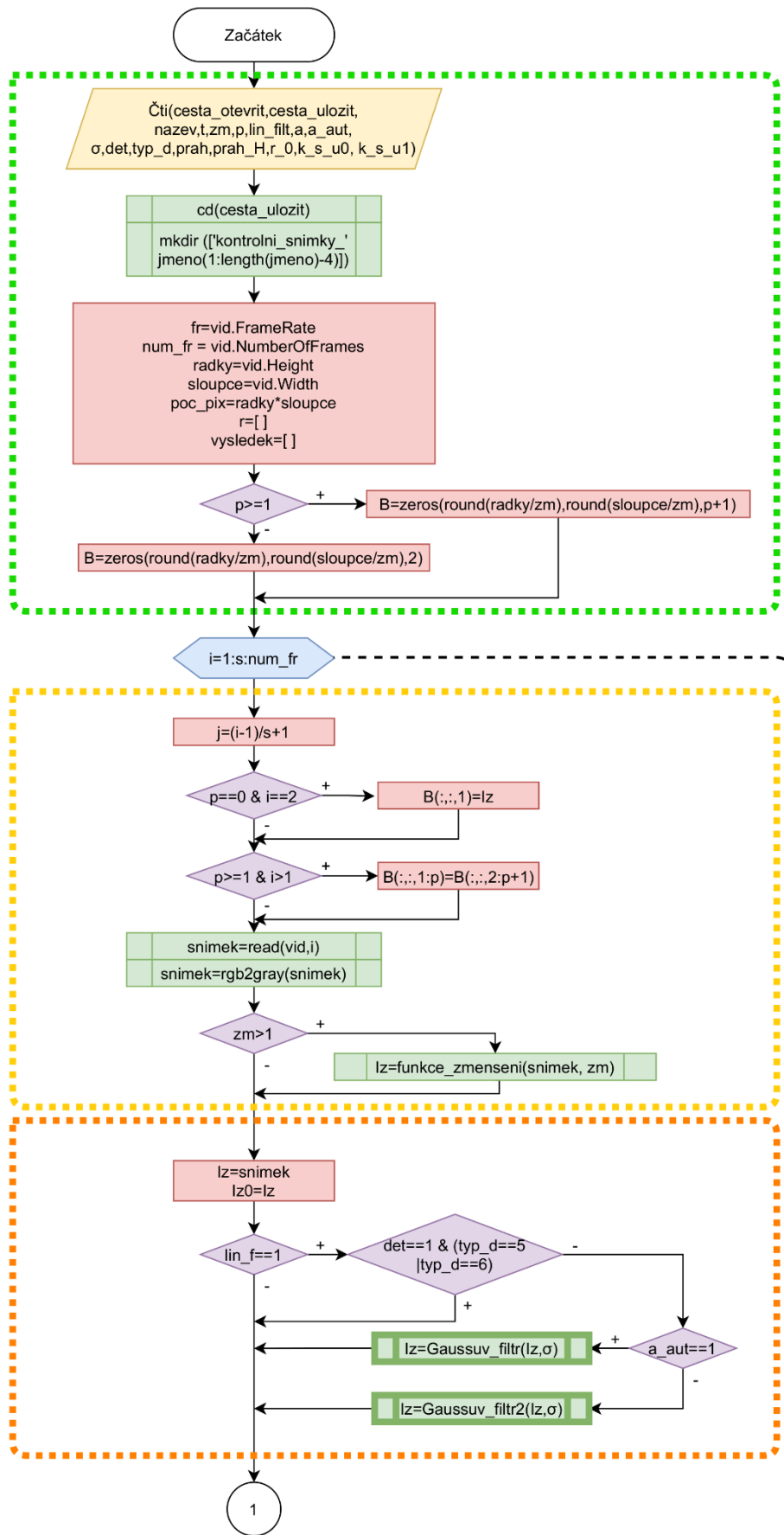
41      jinak
42          jestliže  $a_{aut} = 1$  pak
43          začátek
44           $I_z \leftarrow \text{GAUSSUV\_FILTR}(I_z, \sigma)$ ;
45      jinak
46           $I_z \leftarrow \text{GAUSSUV\_FILTR\_M}(I_z, a, \sigma)$ ;
47      konec
48  konec
49  konec
50  jestliže  $det = 1$  pak
51  začátek
52      jestliže  $typ\_d = 1$  pak
53      začátek
54           $I_z \leftarrow \text{SOBELUV\_OPERATOR}(I_z, prah)$ ;
55      konec
56      jestliže  $typ\_d = 2$  pak
57      začátek
58           $I_z \leftarrow \text{OPERATOR\_PREWITTOVE}(I_z, prah)$ ;
59      konec
60      jestliže  $typ\_d = 3$  pak
61      začátek
62           $I_z \leftarrow \text{ROBINSONUV\_OPERATOR}(I_z, prah)$ ;
63      konec
64      jestliže  $typ\_d = 4$  pak
65      začátek
66           $I_z \leftarrow \text{KIRSCHUV\_OPERATOR}(I_z, prah)$ ;
67      konec
68      jestliže  $typ\_d = 5$  pak
69      začátek
70           $I_z \leftarrow \text{LoG\_OPERATOR}(I_z, \sigma)$ ;
71      konec
72      jestliže  $typ\_d = 6$  pak
73      začátek
74           $I_z \leftarrow \text{CANNYHO\_OPERATOR}(I_z, \sigma, prah, prah\_H)$ ;
75      konec
76  konec
77  jestliže  $k\_s\_u0 = 0 \wedge k\_s\_u1 = 1$  pak
78  začátek
79      ZAPIS_OBRAZ( $I_z$ , [cesta_ulozit '\kontrolni_snimky\_nazev
80                          '\Obraz\_PREVOD\_NA\_RETEZEC(j)'.png']
81  konec
82  jestliže  $k\_s\_u0 = 1 \wedge k\_s\_u1 = 0$  pak
83  začátek

```

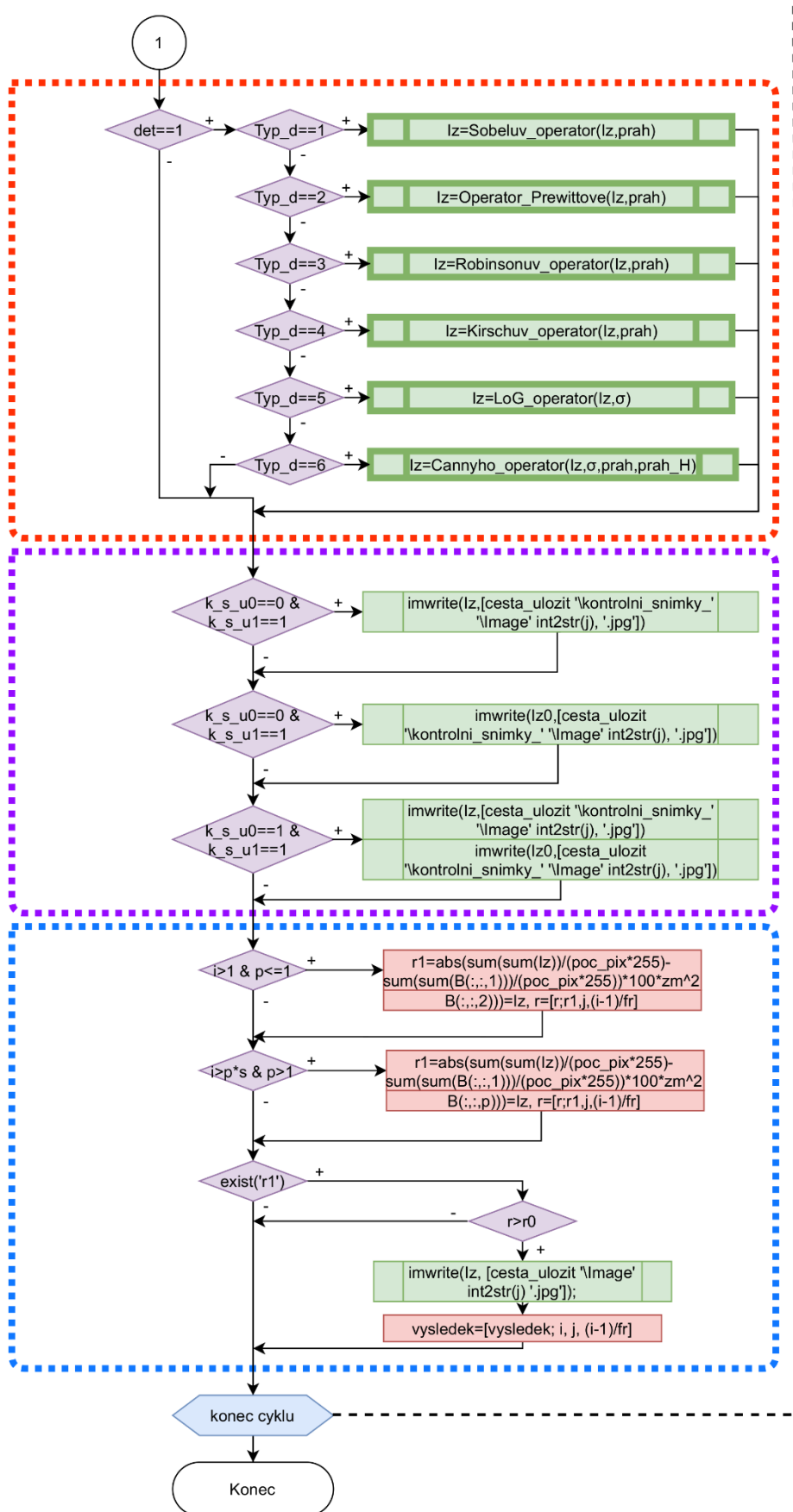
```

83     ZAPIS_OBRAZ ( $I_{z0}$ , [cesta_ulozit '\kontrolni_snimky_' nazev
84     konec
85     jestliže  $k_{s\_u0} = 1 \wedge k_{s\_u1} = 1$  pak
86     začátek
87     ZAPIS_OBRAZ ( $I_z$ , [cesta_ulozit '\kontrolni_snimky_' nazev
88     ZAPIS_OBRAZ ( $I_{z0}$ , [cesta_ulozit '\kontrolni_snimky_' nazev
89     konec
90     jestliže  $i > 1 \wedge p \leq 1$  pak
91     začátek
92      $r_1 \leftarrow abs\left(\frac{\sum \sum I_z}{255p_{pix}} - \frac{\sum \sum B[:, : 1]}{255p_{pix}}\right) \cdot 100z_m^2 ;$ 
93      $B[:, : , 2] \leftarrow I_z$ 
94      $R \leftarrow [R; r_1, j, \frac{i-1}{fr}] ;$ 
95     konec
96     jestliže  $i > p \cdot s \wedge p > 1$  pak
97     začátek
98      $r_1 \leftarrow abs\left(\frac{\sum \sum I_z}{255p_{pix}} - \frac{\sum \sum B[:, : 1]}{255p_{pix}}\right) \cdot 100 ;$ 
99      $B[:, : , p] \leftarrow I_z$ 
100     $R \leftarrow [R; r_1, j, \frac{i-1}{fr}] ;$ 
101    konec
102    jestliže  $\exists r_1$  pak
103    jestliže  $r_1 \geq r_0$  pak
104    začátek
105    ZAPIS_OBRAZ ( $I_z$ , [cesta_ulozit '\Obraz'
106    PREVOD_NA_RETEZEC ( $j$ ) '.png' ]
107     $V_f = [V_f; i, j, \frac{i-1}{fr}] ;$ 
108    konec
109    konec
110   konec

```



Obrázek 4.2: Vývojový diagram algoritmu na detekci změn v obrazu – část 1



Obrázek 4.3: Vývojový diagram algoritmu na detekci změn v obrazu—část 2

4.1 Implementace Gaussova filtru

GAUSSŮV FILTR

```
1  začátek
2  čti (obraz,  $\sigma$ );
3  jestliže VELIKOST (obraz, 3)  $\neq$  1 pak
4  začátek
5      obraz  $\leftarrow$  PREVOD_DO_UROVNI_SEDI (obraz) ;
6  konec
7   $a \leftarrow 2 \cdot \lceil (3\sigma) \rceil + 1$  ;
8   $v \leftarrow (a - 1)/2$  ;
9   $e \leftarrow \exp(1)$  ;
10  $A \leftarrow$  DOUBLE (obraz) ;
11  $A_1 \leftarrow$  ZRCADLOVE_ROZSIRENI ( $A$ ,  $v$ ) ;
12  $K \leftarrow O_{a \times a}$  ;
13 pro  $i$  od  $a$  po  $-1$  do 1
14 začátek
15     pro  $j$  od  $a$  po  $-1$  do 1
16     začátek
17          $u \leftarrow [i - b; j - b]$ ;
18          $K(i, j) \leftarrow \frac{1}{2\pi\sigma^2} e^{-\frac{u(1)^2 + u(2)^2}{2\sigma^2}}$  ;
19     konec
20 konec
21  $K_1 \leftarrow \frac{K}{\sum_{i=1}^a \sum_{j=1}^a K}$  ;
22  $A_{out} \leftarrow$  2D_KONVOLUCE ( $A_1$ ,  $K_1$ ) ;
23 konec
```

Zpočátku je načten obraz, který je brán jako třírozměrná matce prvků, a parametr σ . Pokud je obraz barevný, skládá se ze tří vrstev, z nichž každá je vyhrazena pro jeden barevný kanál nesoucí informaci o červené, zelené či modré barvě, třetí rozměr matice je tak roven třem, pro snadnější úpravu obrazu je třeba převést obraz do úrovně šedi, kdy obsahuje již jen jednu vrstvu s informacemi ekvivalentními k hodnotám celkového jasu obrazu v rozmezí od 0 do 255. K této operaci slouží funkce *rgb2gray*, jenž je dostupná v základní verzi programu MATLAB, stejně jako další funkce, které budou v průběhu popisu pseudokódu zmíněny. Příkazem *size* lze určit velikost rozměrů matice, kdy jako první je uveden počet řádků, tedy výška obrazu, druhým údajem je počet sloupců, šířka obrazu, a třetí hodnota udává velikost třetího rozměru matice. Po určení hodnoty parametru a představujícího velikost konvoluční masky, následuje převedení obrazu z 8-bitového formátu uint8, jež má rozsah hodnot $x \in (0; 2^8 - 1)$,

$x \in \mathbb{Z}$, na datový formát double, zrcadlovému rozšíření obrazu vygenerováním okolního rámu o šířce v a vytvoření nulové matice K o rozměrech $a \times a$ použitím příkazu *zeros* určenou k záznamu hodnot kernelu je v rámci dvou cyklů realizován proces doplnění příslušných hodnot do matice K dle vzorce 1.14. V předposledním kroku je provedena normalizace hodnot konvolučního kernelu vydělením každého jeho prvku součtem všech jeho prvků, nakonec je uskutečněna konvoluce normalizovaného kernelu s rozšířeným obrazem. Aplikační knihovna Image processing toolbox umožňuje pro aplikaci Gaussova filtru využít funkci *imgaussfilt*.

4.2 Implementace konvoluce a zrcadlového rozšíření obrazu

2D KONVOLUCE

```

1  začátek
2  čti (obraz, kernel);
3  radky ← VELIKOST (obraz, 1);
4  sloupce ← VELIKOST (obraz, 2);
5   $v_1 \leftarrow \lfloor \text{VELIKOST}(\textit{kernel}, 1) - 1 \rfloor / 2$ 
6   $v_2 \leftarrow \lfloor \text{VELIKOST}(\textit{kernel}, 2) - 1 \rfloor / 2$ 
7   $A \leftarrow O_{(\textit{radky} - 2v_1) \times (\textit{sloupce} - 2v_2)}$ ;
8  pro i od  $v_1$  do radky -  $v_1$ 
9  začátek
10  pro j od  $v_2$  do sloupce -  $v_2$ 
11  začátek
12   $A(i, j) \leftarrow \sum_{c=-v_1}^{v_1} \sum_{d=-v_2}^{v_2} \textit{obraz}(i + c, j + d) \cdot$ 
    $\textit{kernel}(v_1 + c, v_2 + d)$ 
13  konec
14  konec
15  konec
```

Vstupními hodnotami pro konvoluci jsou obraz, optimálně v rozšířené formě a formátu double, a konvoluční kernel. Parametru *radky* je přiřazena hodnota celkového počtu řádků, parametr *sloupce* vyjadřuje analogicky počet sloupců matice obrazových hodnot, v_1 a v_2 odpovídají polovičnímu množství sloupců a řádků nacházejících se v okolí centrálního bodu konvoluční masky, při využití standardních čtvercových kernelů jsou jejich hodnoty stejné. Dále je vygenerována nulová matice o velikosti obrazu před rozšířením, tedy matice $A[\textit{radky} - 2v_1, \textit{sloupce} - 2v_2]$, protože při samotné konvoluci, jejíž hlavní část tvořící její podstatu je uvedena na řádku 12, jsou zpracovány hodnoty ve vzdálenosti větší než v_1 od horního a dolního a větší než v_2 od levého a pravého kraje obrazu. Konvoluci je také možné provést aplikací funkce *conv2*.

ZRCADLOVÉ ROZŠÍŘENÍ OBRAZU

```
1 začátek
2   čti (obraz, v);
3   radky ← VELIKOST (obraz, 1);
4   sloupce ← VELIKOST (obraz, 2);
5    $A_1 \leftarrow O_{(radky+2v) \times (sloupce+2v)}$ ;
6    $A_1(\text{od } v + 1 \text{ do } radky + v, \text{od } v + 1 \text{ do } sloupce + v) \leftarrow obraz$ ;
7   pro i od 1 do v
8   začátek
9      $A_1(\text{od } v + 1 \text{ do } radky + v, i) \leftarrow obraz(:, v + 1 - i)$ ;
10     $A_1(\text{od } v + 1 \text{ do } radky + v, sloupce + v + i) \leftarrow$ 
11       $\leftarrow obraz(:, sloupce - i)$ ;
12   konec
13   pro j od 1 do v
14   začátek
15      $A_1(j, :) \leftarrow A_1(2v + 1 - j, :)$ ;
16      $A_1(radky + v + j, :) \leftarrow A_1(radky + v + 1 - j, :)$ ;
17   konec
18 konec
```

Zrcadlové rozšíření obrazu slouží k vytvoření rámu kolem obrazu s hodnotami shodnými s hodnotami obrazu dle os souměrnosti umístěných na okrajích obrazu. Kromě vstupních hodnot v podobě obrazu a parametru v , což je polovina počtu řádků/sloupců v okolí středového bodu konvoluční masky, je důležitou součástí nulová matice A_1 o velikosti $(radky + 2v) \times (sloupce + 2v)$, která je nejprve doplněna o hodnoty bodů obrazu tak, že do vzdálenosti v jsou všechny krajní hodnoty stále rovny 0, prvky ve sloupcích od $v + 1$ do $sloupce + v$ a v řádcích od $v + 1$ do $radky + v$ mají hodnotu bodu vstupního obrazu. Poté doplněny hodnoty od $v + 1$ do $radky + v$ prvních v sloupců levého okraje matice A_1 hodnotami s prvních v sloupců vstupního obrazu v opačném pořadí, i -tému sloupci budou přiřazeny hodnoty sloupce obrazu s pořadím $v + 1 - i$. Například pokud bude $v = 3$, tak první sloupec A_1 získá hodnoty třetího sloupce vstupního obrazu, druhý sloupec A_1 hodnoty druhého sloupce obrazu a třetí sloupec A_1 hodnoty prvního sloupce obrazu. Stejným způsobem jsou doplněny i hodnoty v sloupců pravého okraje. Hodnoty prvních v řádků horní části a dolní části jsou také doplňovány na totožném principu, avšak nejsou již brány z původního obrazu ale z upravované matice A_1 .

4.3 Implementace Sobelova operátoru a operátoru Prewittové

SOBELŮV OPERÁTOR

```
1 začátek
2  čti (obraz, prah);
3  jestliže VELIKOST (obraz,3) ≠ 1 pak
4  začátek
5    obraz ← PREVOD_DO_UROVNI_SEDI (obraz) ;
6  konec
7  A ← DOUBLE (obraz) ;
8  A1 ← ZRCADLOVE_ROZSIRENI (A, 1) ;
9  Gx ← [-1, 0, 1; -2, 0, 2; -1, 0, 1] ;
10 Gy ← [-1, -2, -1; 0, 0, 0; 1, 2, 1] ;
11 Ex ← 2D_KONVOLUCE (A1, Gx) ;
12 Ey ← 2D_KONVOLUCE (A1, Gy) ;
13 E ←  $\sqrt{E_x^2 + E_y^2}$  ;
14 Mθ ←  $\tan^{-1} \frac{E_y}{E_x}$  ;
15 E ←  $\frac{E}{\max(E)} \cdot 255$  ;
16 E (E < prah) ← 0 ;
17 konec
```

Sobelův operátor a operátor Prewittové mají velice podobnou až téměř shodnou skladbu, rozdíl mezi nimi se nalézá pouze v podobě konvolučních kernelů. Jednotlivé kroky prvotní části zahrnující řádky 1 až 8 jsou vysvětleny v rámci Gaussova filtru a tak nebudou více popisovány. Druhá část algoritmu je zaměřena na konvoluci rozšířeného obrazu s konvolučními maskami Sobelova operátoru pro detekci hran v horizontálním a vertikálním směru, které jsou zaneseny do matice E_x a E_y , ze kterých jsou následovně vypočítány a zaneseny do matice E velikosti nalezených hran a úhly jejich gradientu náležící matici M_θ . Následuje normalizace velikostí hran, při níž jsou všechny prvky matice E vyděleny maximální hodnotou v ní se vyskytující a vynásobeny číslem 255, tímto je zaručeno, že nejvyšší hodnota matice nepřekročí hodnotu 255, což je důležité pro zobrazení obrazu v 8-bitových barvách, protože se jedná o nejvyšší možnou hodnotu ve standardním datovém typu používaném pro obraz. Posledním úkonem je prahování, kdy jsou zachovány pouze hrany s velikostí převyšující stanovenou prahovou hodnotu, ostatní jsou nahrazeny nulou. V rámci Image processing toolboxu je možné tyto operátory aplikovat příkazem `edge(obraz,'Sobel')` a `edge(obraz, 'Prewitt')`.

OPERÁTOR PREWITTOVÉ

```
1 začátek
2  čti (obraz, prah);
3  jestliže VELIKOST (obraz,3) ≠ 1 pak
4  začátek
5     obraz ← PREVOD_DO_UROVNI_SEDI (obraz) ;
6  konec
7  A ← DOUBLE (obraz) ;
8  A1 ← ZRCADLOVE_ROZSIRENI (A, 1) ;
9  Gx ← [-1, 0, 1; -1, 0, 1; -1, 0, 1] ;
10 Gy ← [-1, -1, -1; 0, 0, 0; 1, 1, 1] ;
11 Ex ← 2D_KONVOLUCE (A1, Gx) ;
12 Ey ← 2D_KONVOLUCE (A1, Gy) ;
13 E ←  $\sqrt{E_x^2 + E_y^2}$  ;
14 Mθ ←  $\tan^{-1} \frac{E_y}{E_x}$  ;
15 E ←  $\frac{E}{\max(E)} \cdot 255$  ;
16 E(E < prah) ← 0 ;
17 konec
```

4.4 Implementace Robinsonova a Kirschova operátoru

ROBINSONŮV OPERÁTOR

```
1 začátek
2  čti (obraz, prah);
3  jestliže VELIKOST (obraz,3) ≠ 1 pak
4  začátek
5     obraz ← PREVOD_DO_UROVNI_SEDI (obraz) ;
6  konec
7  A ← DOUBLE (obraz) ;
8  A1 ← ZRCADLOVE_ROZSIRENI (A, 1) ;
9  G1 ← [1, 1, 1; 1, -2, 1; -1, -1, -1] ;
10 G2 ← [1, 1, 1; -1, -2, 1; -1, -1, 1] ;
11 G3 ← [-1, 1, 1; -1, -2, 1; -1, 1, 1] ;
12 G4 ← [-1, -1, 1; -1, -2, 1; 1, 1, 1] ;
13 G5 ← [-1, -1, -1; 1, -2, 1; 1, 1, 1] ;
14 G6 ← [1, -1, -1; 1, -2, -1; 1, 1, 1] ;
15 G7 ← [1, 1, -1; 1, -2, -1; 1, 1, -1] ;
16 G8 ← [1, 1, 1; 1, -2, -1; 1, -1, -1] ;
17 E1 ← 2D_KONVOLUCE (A1, G1) ;
```

```

18   $E_2 \leftarrow 2D\_KONVOLUCE (A_1, G_2) ;$ 
19   $E_3 \leftarrow 2D\_KONVOLUCE (A_1, G_3) ;$ 
20   $E_4 \leftarrow 2D\_KONVOLUCE (A_1, G_4) ;$ 
21   $E_5 \leftarrow 2D\_KONVOLUCE (A_1, G_5) ;$ 
22   $E_6 \leftarrow 2D\_KONVOLUCE (A_1, G_6) ;$ 
23   $E_7 \leftarrow 2D\_KONVOLUCE (A_1, G_7) ;$ 
24   $E_8 \leftarrow 2D\_KONVOLUCE (A_1, G_8) ;$ 
25   $radky \leftarrow VELIKOST (A, 1) ;$ 
26   $sloupce \leftarrow VELIKOST (A, 2) ;$ 
27   $E \leftarrow O_{radky \times sloupce} ;$ 
28  pro i od 1 do radky
29  začátek
30    pro j od 1 do sloupce
31    začátek
32       $E(i, j) \leftarrow \max\{E_1(i, j), E_2(i, j), E_3(i, j), E_4(i, j),$ 
33         $E_5(i, j), E_6(i, j), E_7(i, j), E_8(i, j)\} ;$ 
34    konec
35  konec
36   $E \leftarrow \frac{E}{\max(E)} \cdot 255 ;$ 
37   $E(E < prah) \leftarrow 0$ 
38  konec

```

U Robinsonova a Kirschova operátoru platí stejné tvrzení ohledně podobnosti jako předchozí dvojice hranových detektorů, odlišnost mezi nimi je opět pouze v podobě konvolučních masek. Základem těchto operátorů je osm konvolucí obrazu s kernely určených ke hledání hran v osmi směrech. Výsledná velikost hrany v určitém bodě je určena maximem ze skupiny prvků vybraných z každé matice E_1, \dots, E_8 , vzniklých z konvoluce, s polohou odpovídající tomuto bodu. Jak je patrné z řádků 28 až 34, tato operace je provedena pro všechny prvky matic. Finální fáze zahrnuje normalizaci a prahování. Robinsonův ani Kirschův operátor nejsou v knihovně pro zpracování obrazu zahrnuti.

KIRSCHŮV OPERÁTOR

```

1  začátek
2  čti (obraz, prah);
3  jestliže VELIKOST (obraz,3)  $\neq$  1 pak
4  začátek
5     $obraz \leftarrow PREVOD\_DO\_UROVNI\_SEDI (obraz) ;$ 
6  konec
7   $A \leftarrow DOUBLE (obraz) ;$ 

```

```

8    $A_1 \leftarrow \text{ZRCADLOVE\_ROZSIRENI}(A, 1)$ ;
9    $G_1 = [-3, -3, -3; -3, 0, -3; 5, 5, 5]$ ;
10   $G_2 = [-3, -3, -3; 5, 0, -3; 5, 5, -3]$ ;
11   $G_3 = [5, -3, -3; 5, 0, -3; 5, -3, -3]$ ;
12   $G_4 = [5, 5, -3; 5, 0, -3; -3, -3, -3]$ ;
13   $G_5 = [5, 5, 5; -3, 0, -3; -3, -3, -3]$ ;
14   $G_6 = [-3, 5, 5; -3, 0, 5; -3, -3, -3]$ ;
15   $G_7 = [-3, -3, 5; -3, 0, 5; -3, -3, 5]$ ;
16   $G_8 = [-3, -3, -3; -3, 0, 5; -3, 5, 5]$ ;
17   $E_1 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_1)$ ;
18   $E_2 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_2)$ ;
19   $E_3 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_3)$ ;
20   $E_4 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_4)$ ;
21   $E_5 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_5)$ ;
22   $E_6 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_6)$ ;
23   $E_7 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_7)$ ;
24   $E_8 \leftarrow \text{2D\_KONVOLUCE}(A_1, G_8)$ ;
25   $\text{radky} \leftarrow \text{VELIKOST}(A, 1)$ ;
26   $\text{sloupce} \leftarrow \text{VELIKOST}(A, 2)$ ;
27   $E \leftarrow O_{\text{radky} \times \text{sloupce}}$ ;
28  pro  $i$  od 1 do  $\text{radky}$ 
29  začátek
30  pro  $j$  od 1 do  $\text{sloupce}$ 
31  začátek
32   $E(i, j) \leftarrow \max\{E_1(i, j), E_2(i, j), E_3(i, j), E_4(i, j),$ 
33   $E_5(i, j), E_6(i, j), E_7(i, j), E_8(i, j)\}$ ;
34  konec
35  konec
36   $E \leftarrow \frac{E}{\max(E)} \cdot 255$ ;
37   $E(E < \text{prah}) \leftarrow 0$ 
38  konec

```

4.5 Implementace LoG operátoru

LOG OPERÁTOR

```

1  začátek
2  čti ( $\text{obraz}, \text{sigma}$ );
3  jestliže  $\text{VELIKOST}(\text{obraz}, 3) \neq 1$  pak
4  začátek
5   $\text{obraz} \leftarrow \text{PREVOD\_DO\_UROVNI\_SEDI}(\text{obraz})$ ;
6  konec
7   $a \leftarrow 2 \cdot [(3 \cdot \text{sigma})] + 1$ ;

```

```

8   v ← (a - 1)/2 ;
9   e ← exp(1) ;
10  A ← DOUBLE (obraz) ;
11  A1 ← ZRCADLOVE_ROZSIRENI (A, v) ;
12  K ← Oa×a ;
13  pro i od a po -1 do 1
14  začátek
15    pro j od a po -1 do 1
16    začátek
17      u ← [i - b; j - b];
18       $K(i, j) \leftarrow \left( \frac{u(1)^2 + u(2)^2 - 2\sigma^2}{2\pi\sigma^2} \right) \cdot e^{(-1) \cdot \frac{u(1)^2 + u(2)^2}{2\sigma^2}} ;$ 
19    konec
20  konec
21  radky ← VELIKOST (A, 1) ;
22  sloupce ← VELIKOST (A, 2) ;
23  E ← 2D_KONVOLUCE (A, K) ;
24  E(E < 0) ← 0 ;
25  E(E > 0) ← 255 ;
26  Eout ← Oradky×sloupce ;
27  pro i od 1 do radky
28  začátek
29    pro j od 1 do sloupce
30    začátek
31      jestliže |E(i, j) - E(i + 1, j)| > 0 ∨
                 |E(i, j) - E(i, j + 1)| > 0 ∨
                 |E(i, j) - E(i + 1, j + 1)| > 0 pak
32      začátek
33        Eout(i, j) ← 255 ;
34      jinak
35        Eout(i, j) ← 0 ;
36      konec
37      jestliže |E(radky, j) - E(radky - 1, j)| > 0 ∨
                 |E(radky, j) - E(radky, j + 1)| > 0 ∨
                 |E(radky, j) - E(radky - 1, j + 1)| > 0 pak
38      začátek
39        Eout(radky, j) ← 255 ;
40      jinak
41        Eout(radky, j) ← 0 ;
42      konec
43      jestliže |E(i, sloupce) - E(i, sloupce - 1)| > 0 ∨
                 |E(i, sloupce) - E(i + 1, sloupce)| > 0 ∨
                 |E(i, sloupce) - E(i + 1, sloupce - 1)| > 0 pak

```

```

44     začátek
45          $E_{out}(i, sloupce) \leftarrow 255 ;$ 
46     jinak
47          $E_{out}(i, sloupce) \leftarrow 0 ;$ 
48     konec
49 konec
50 konec
51 konec

```

LoG operátor má v prvních dvaceti řádcích téměř totožnou formu jako Gaussův filtr, což je způsobeno tím, že z něj prakticky vychází, jen vztah, dle kterého jsou počítány hodnoty konvolučního kernelu filtru, se liší, je dán rovnicí 3.32. První část hledání hran spočívá v konvoluci obrazu s vytvořenou maskou, nato jsou hodnoty výsledné matice E menší než nula nahrazeny nulou a hodnoty větší než nula nabývají 255. Nulová matice E_{out} slouží k záznamu hodnot průchodu nulou, kdy je pro každý prvek matice E ověřováno, zda má prvek v sousedním řádku, sloupci či sousední úhlopříčce stejnou nebo odlišnou hodnotu. V podstatě je prvek porovnáván s hodnotami ze čtvercové oblasti o velikosti 2×2 , kde jeho poloha odpovídá pravému hornímu rohu. V případě shody hodnot je prvku o stejné pozici v matici E_{out} dána hodnota 0, pokud se hodnoty různí, absolutní hodnota jejich rozdílu je větší než nula, tak získá hodnotu 255. Image processing toolbox zahrnuje pro aplikaci LoG operátoru funkci `edge(obraz, 'log')`.

4.6 Implementace Cannyho operátoru

CANNYHO OPERÁTOR

```

1  začátek
2  čti (obraz,  $\sigma$ , prahD, prahH) ;
3   $p \leftarrow pocet\_opakovani ;$ 
4  jestliže VELIKOST (obraz,3)  $\neq 1$  pak
5  začátek
6   $obraz \leftarrow PREVOD\_DO\_UROVNI\_SEDI (obraz) ;$ 
7  konec
8   $A \leftarrow GAUSSŮV\_FILTR (obraz, \sigma) ;$ 
9   $[A_s, M_\theta] \leftarrow SOBELŮV\_OPERÁTOR (A, prah\_D)$ 
10  $A_\theta \leftarrow N-M\_SUPPRESSION (A_s, M_\theta) ;$ 
11  $A_{out} \leftarrow PRAHOVÁNÍ\_S\_HYSTEREZÍ (A, p) ;$ 
12 konec

```

Cannyho operátor obsahuje nejvíce vstupních hodnot ze zmíněných operátorů, mimo obrazu jimi jsou σ , a dolní a horní práh. Po případném převedení obrazu do úrovní šedi, je obraz vyhlazen Gaussovým filtrem, dále dochází k detekci hran Sobelovým operátorem. Nalezené hrany jsou zúženy potlačením odezev mimo maxima, označovaným také jako non-maximal suppression, během kterého jsou velikosti hranových bodů porovnávány se dvěma sousedními body ve směru gradientu a pokud má posuzovaný prvek vyšší hodnotu než oba prvky, se kterými je porovnáván, tak je jeho hodnota přenesena do nulové matice na stejnou pozici, v opačném případě k přenosu hodnoty prvku nedochází. Proces působení Cannyho operátoru uzavírá prahování s hysterezí, jehož se týká parametr p , ovlivňující počet opakování procesu, při němž jsou prvky s hodnotou spadající mezi dolní a horní práh uznány jako bod hrany pouze tehdy, když se v jejich okolí o velikosti 3×3 nachází prvek s hodnotou větší než horní práh.

N-M SUPPRESSION

```

1  začátek
2  čti ( $A_s, M_\theta$ )
3   $radky \leftarrow \text{VELIKOST}(M_\theta, 1)$ ;
4   $sloupce \leftarrow \text{VELIKOST}(M_\theta, 2)$ ;
5   $A_\theta \leftarrow O_{radky \times sloupce}$ ;
6   $M_\theta \leftarrow -M_\theta$ ;
7   $M_\theta(A_s = 0) \leftarrow 255$ ;
8   $M_\theta(M_\theta < 0) \leftarrow M_\theta + 90^\circ$ ;
9  pro  $i$  od 2 do  $radky - 1$ 
10 začátek
11   pro  $j$  od 2 do  $sloupce - 1$ 
12   začátek
13     jestliže [ $M_\theta(i, j) \geq 0^\circ \wedge M_\theta(i, j) \leq 22,5^\circ$ ] v
14       [ $M_\theta(i, j) > 157,5^\circ \wedge M_\theta(i, j) \leq 180^\circ$ ] pak
15     začátek
16       jestliže  $A_s(i, j - 1) \leq A_s(i, j) \wedge A_s(i, j) \geq A_s(i, j + 1)$  pak
17       začátek
18          $A_\theta(i, j) \leftarrow 255$ ;
19       jinak
20          $A_\theta(i, j) \leftarrow 0$ ;
21       konec
22     jestliže  $M_\theta(i, j) > 22,5^\circ \wedge M_\theta(i, j) \leq 67,5^\circ$  pak
23     začátek
24       jestliže  $A_s(i + 1, j - 1) \leq A_s(i, j) \wedge A_s(i, j) \geq A_s(i - 1, j + 1)$ 

```

pak

25 **začátek**

26 $A_{\theta}(i, j) \leftarrow 255 ;$

27 **jinak**

28 $A_{\theta}(i, j) \leftarrow 0 ;$

29 **konec**

30 **konec**

31 **jestliže $M_{\theta}(i, j) > 67,5^{\circ} \wedge M_{\theta}(i, j) \leq 112,5^{\circ}$ pak**

32 **začátek**

33 **jestliže $A_s(i - 1, j) \leq A_s(i, j) \wedge A_s(i, j) \geq A_s(i + 1, j)$ pak**

34 **začátek**

35 $A_{\theta}(i, j) \leftarrow 255 ;$

36 **jinak**

37 $A_{\theta}(i, j) \leftarrow 0 ;$

38 **konec**

39 **jestliže $M_{\theta}(i, j) > 112,5^{\circ} \wedge M_{\theta}(i, j) \leq 157,5^{\circ}$ pak**

40 **začátek**

41 **jestliže $A_s(i - 1, j - 1) \leq A_s(i, j) \wedge A_s(i, j) \geq A_s(i + 1, j + 1)$**

42 **pak**

42 **začátek**

43 $A_{\theta}(i, j) \leftarrow 255 ;$

44 **jinak**

45 $A_{\theta}(i, j) \leftarrow 0 ;$

46 **konec**

47 **konec**

48 **konec**

49 **konec**

50 **konec**

PRAHOVÁNÍ S HYSTEREZÍ

1 **začátek**

2 čti (A_{θ}, p)

3 $radky \leftarrow \text{VELIKOST}(A_{\theta}, 1) ;$

4 $sloupce \leftarrow \text{VELIKOST}(A_{\theta}, 2) ;$

5 **pro i od 1 do $radky$**

6 **začátek**

7 **pro j od 1 do $sloupce$**

8 **začátek**

9 **jestliže $A_{\theta} \geq prah_H$ pak**

10 **začátek**

11 $A_{out}(i, j) \leftarrow 255 ;$

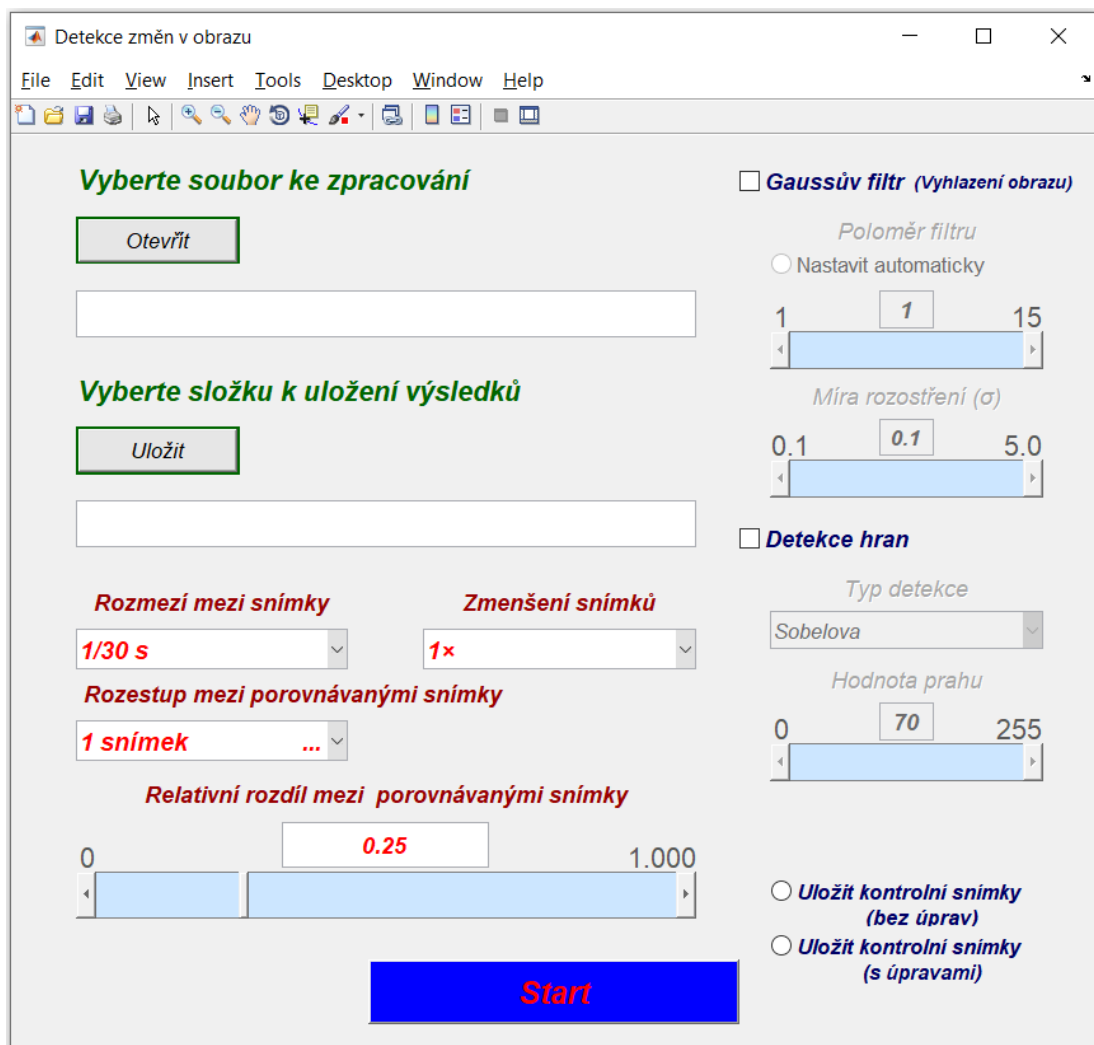
12 **konec**

13 **konec**

14 **konec**
15 **pro** o **od** 1 **do** p
16 **začátek**
17 **pro** i **od** 2 **do** *radky* $- 1$
18 **začátek**
19 **začátek**
20 **pro** j **od** 2 **do** *sloupce* $- 1$
21 **začátek**
22 **jestliže** $A_{\theta}(i, j) \geq \text{prah}_D \wedge A_{\theta}(i, j) < \text{prah}_H$ **pak**
23 **začátek**
24 $B \leftarrow \sum_{b=-1}^1 A_{out}(i + b, j + b) ;$
25 **jestliže** $B \geq 255$ **pak**
26 **začátek**
27 $A_{out}(i, j) = 255 ;$
28 **konec**
29 **konec**
30 **konec**
31 **konec**
32 **konec**
33 **konec**

4.7 Grafické uživatelské prostředí

K algoritmu byla vytvořena i jednoduchá grafická nadstavba, která umožní uživatelsky přívětivější vkládání dat a volbu parametrů úprav obrazu. V MATLABu lze grafické uživatelské prostředí, GUI, vytvořit v systému Handle Graphics, což je rozšíření implementované v základní verzi softwaru, umožňující pracovat s grafikou použitím definovaných příkazů, tímto způsobem bylo sestaveno grafické prostředí vyvinutého algoritmu, jenž je uvedeno na obrázku 4.4. Druhou možností jak vytvořit GUI je prostředí GUIDE, které obsahuje soubor nástrojů pro interaktivní tvorbu grafických objektů, kdy je uživatelské prostředí formováno jejich výběrem a umístěním pomocí kurzoru, editace parametrů vybraných objektů je realizována v okně pro úpravu vlastností zvaném Property Inspector, výsledný kód je generován automaticky.



Obrázek 4.4: Grafické uživatelské prostředí programu pro detekci změn v obrazu

Základním objektem GUI je Figure, který představuje hlavní okno programu, do něhož jsou umístěny podřízené grafické objekty Uicontrol, Uimenu a Uicontextmenu.

Ke každému objektu musí být přiřazena proměnná, tzv. Handle, dle které je grafický objekt jednoznačně identifikován. Při spuštění programu je příkazem `get(0, 'ScreenSize')`, kde 0 označuje hierarchicky nejvyšší objekt, Root, totožný s obrazovkou monitoru, a `'ScreenSize'` zastupuje parametr určující jeho rozlišení, získáno rozlišení obrazovky monitoru, dle kterého jsou nastaveny rozměry a pozice hlavního okna. Horizontální pozice je definována jako třetina horizontálního rozlišení monitoru, stejně tak i šířka okna, analogicky je stanovena i vertikální pozice, výška okna je rovna polovině vertikálního rozlišení monitoru. Vybrané hodnoty pozice a rozměrů je třeba zadat do parametru `'Position'`, za jehož jednotky byly zvoleny pixely.

Veškeré prvky zobrazené uvnitř hlavního okna jsou objekty `Uicontrol` s různými styly, mezi které se řadí statický text, jednostavové tlačítko `'Push button'`, rám `'Frame'`, sloužící ke zvýraznění ostatních objektů, editovací pole `'Edit text'` umožňující vkládat textové řetězce, seznam položek `'Popup menu'`, posuvník `'Slider'`, zaškrťovací políčko `'Check box'` či dvoustavové tlačítko `'Radio button'`. Objekty `Uimenu` a `Uicontextmenu` nejsou ve zhotoveném GUI použity.

a)

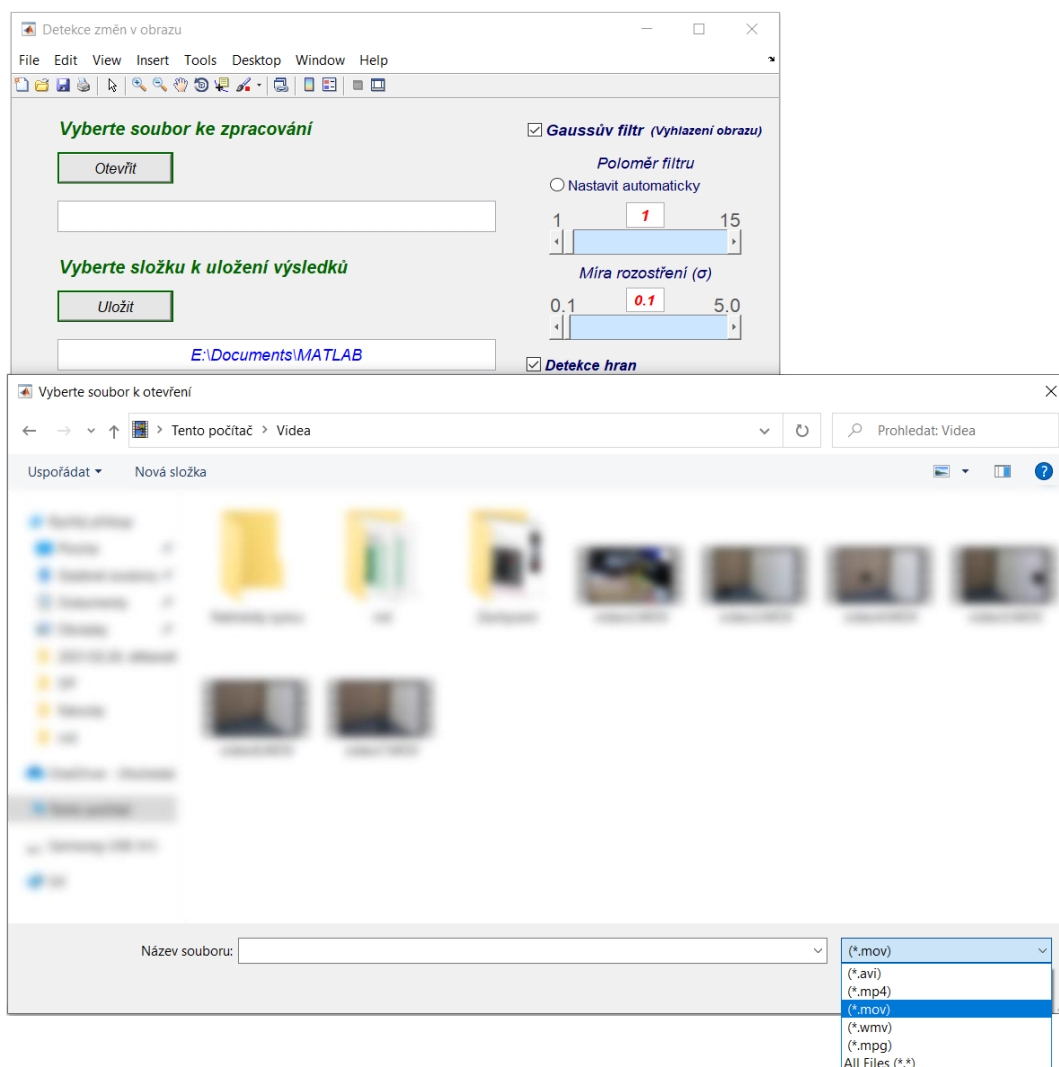
```
g1_2=uicontrol('Style', 'Pushbutton',...
              'Units', 'normalized',...
              'Position', [0.063 0.862 0.145 0.046],...
              'Tag', 'otevrit',...
              'BackgroundColor', [0.9 0.9 0.9],...
              'ForegroundColor', 'black',...
              'FontUnits', 'normalized',...
              'FontSize', 0.5,...
              'FontWeight', 'normal',...
              'Fontangle', 'italic',...
              'string', 'Otevřít',...
              'callback', 'programDP otevrit');
```

b)

```
h1_2=uicontrol('Style', 'Popupmenu',...
              'Units', 'normalized',...
              'Position', [0.06 0.41 0.25 0.05],...
              'Tag', 'cas_rozmezi',...
              'BackgroundColor', 'white',...
              'ForegroundColor', 'red',...
              'FontUnits', 'normalized',...
              'FontSize', 0.5,...
              'FontWeight', 'bold',...
              'Fontangle', 'italic',...
              'String', '1/30s|1/25s|0.1s|0.2s|0.5s|1s|2s|5s',...
              'callback', 'programDP cas');
```

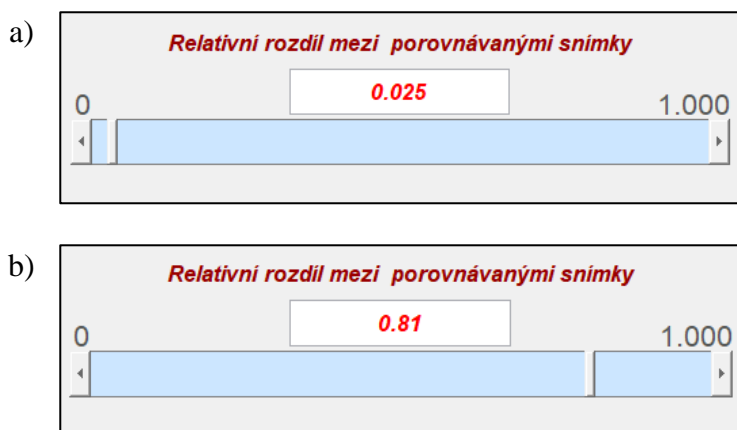
Obrázek 4.5: Ukázka nastavení vybraných parametrů pro a) jednostavové tlačítko Otevřít, b) popup menu pro výběr časového rozmezí mezi zpracovávanými snímky videa

Jednotlivé objekty obsahují několik editovatelných parametrů, z nichž byly často využity úpravy pozice včetně velikosti s normalizovanými jednotkami, jenž je vztažena vůči hlavnímu oknu nikoliv k obrazovce monitoru, dále parametru *'Tag'*, sloužícího jako značka k vyhledání objektu, a parametru *'CallBack'*, který zajišťuje vykonání činnosti dle zdrojového kódu spojené s reakcí vybraného objektu na jeho aktivaci nebo změnu. U většiny objektů došlo také k editaci barev pozadí případně písma daných parametry *'BackgroundColor'* a *'ForegroundColor'* a vlastností písma pomocí parametrů *'FontSize'*, *'FontWeight'*, *'FontAngle'*. Veškerý text v hlavním okně, tedy popisky tlačítek, seznam položek u *'Popup menu'*, vyplněné hodnoty v editovacích polích a statický text, je vypsán za parametr *'String'* u vybraných objektů.



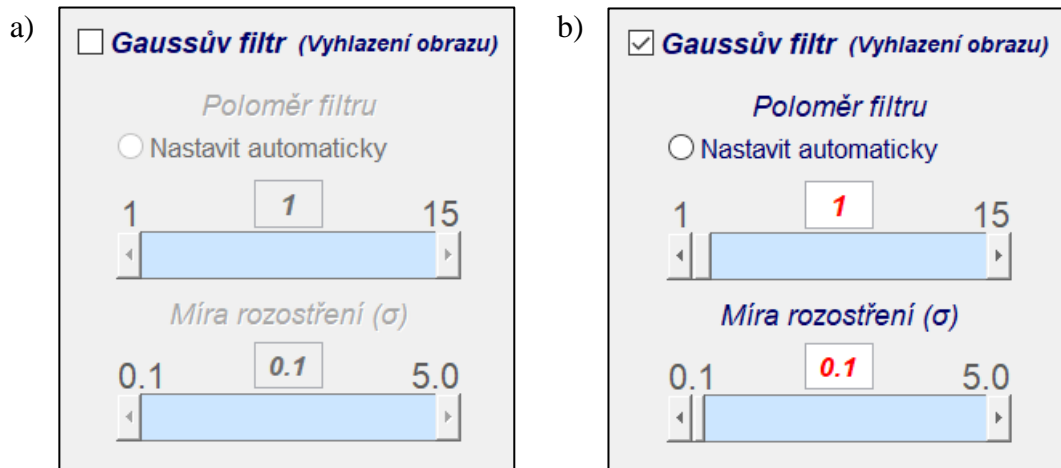
Obrázek 4.6: Dialogové okno pro výběr videosouboru podporovaného formátu

V první části tvorby GUI jsou tedy nadefinovány zobrazované objekty a jejich vlastnosti, druhá část začínající spuštěním přepínače příkazem *switch*, stanovuje jejich činnost, kdy úkon při aktivitě objektu je vyvolán příkazem *case('příklad')*, přičemž '*příklad*' je řetězec náležící k parametru '*CallBack*'. Po stisknutí tlačítka ,Otevřít' dojde k otevření dialogového okna pro výběr vstupního videosouboru, ve kterém lze zobrazit soubory formátu *.avi, *.mp4, *.mov, *.wmv a *.mpg, do editovacího pole je automaticky vepsána cesta k vybranému souboru. Pro přímé zadání cesty k požadovanému souboru je možné využít pouze editovací pole. Tlačítkem ,Uložit' je také otevřeno dialogové okno sloužící k výběru složky pro uložení výsledků, provázanost s editovacím polem pod tlačítkem je stejná jako u výběru vstupního souboru. Na obrázku 4.6 je zachyceno dialogové okno pro výběr videosouboru, cesta k vybrané složce k uložení výsledků je již vyplněna v náležitém editovacím okně.



Obrázek 4.7: Znázornění vzájemné vazby posuvníku a příslušného editovacího pole

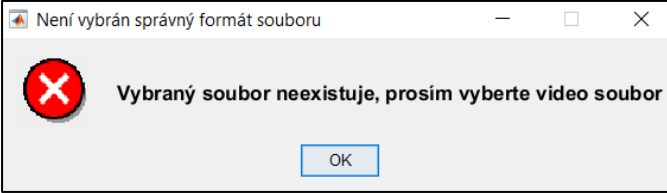
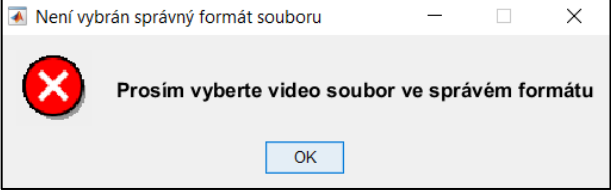
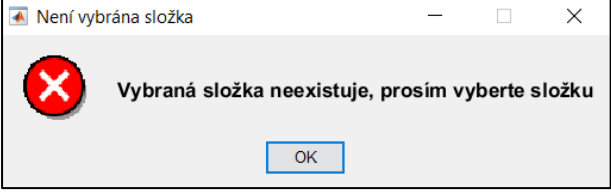
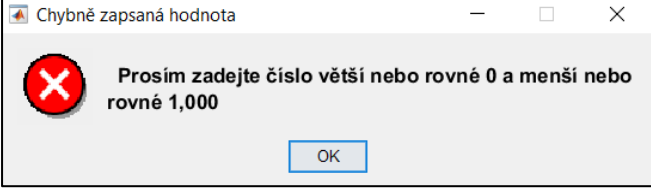
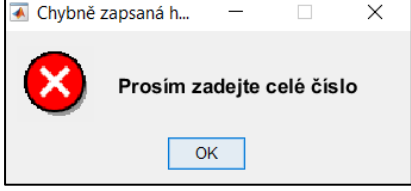
Popup menu ,Rozmezí mezi snímky', ,Zmenšení snímků' a ,Rozestup mezi porovnávanými snímky' jsou určeny pouze k získávání vstupních parametrů ke zpracování videosouboru, jejich aktivita není příkazem *case* nijak postoupena. Posuvníky a editovací pole umístěné nad každým z nich jsou vzájemně propojeny, při změně hodnoty na posuvníku se mění i hodnota v editovacím poli, zanesením jisté hodnoty do editovacího pole dochází také k úpravě polohy jezdce posuvníku, viz obr. 4.7. Zaškrtačací pole ,Gaussov filtr' a ,Detekce hran' mají úlohu zpřístupnit objekty nalézající se pod nimi, pokud jsou označeny, pokud označeny nejsou, tak zpřístupnění nenastává. Označením tlačítka typu radio button s popisem ,Nastavit automaticky' je nastavena velikost konvolučního kernelu Gaussova filtru a deaktivován posuvník i editační pole pro zadání příslušných hodnot.



Obrázek 4.8: Znázornění funkce zaškrtačacího pole v prostředí programu a) neoznačené pole s výslednou deaktivací náležitých objektů, b) označené pole zapříčiňující zobrazení příslušných objektů

4.7.1 Opatření GUI chybovými hláškami

Do grafického prostředí byly implementovány také chybové hlášky, které jsou důležité pro informování uživatele o chybném kroku, díky němuž by funkce programu mohla být nerelevantní nebo by jeho činnost nemusela proběhnout vůbec. Tímto způsobem je třeba ošetřit všechny objekty, do nichž lze hodnoty zadávat ručně, tudíž v případě vytvořeného programu se to týká editovacích polí. Pokud není vybrán vstupní soubor nebo zadaná cesta nevede k žádnému videosouboru, zobrazí se hláška 1 informující o jeho neexistenci, když je vybrán soubor neodpovídajícího formátu, dojde k zobrazení hlášky 2, která žádá o výběr souboru ve správném formátu. Chybová hláška 3 označuje neexistenci složky při zadání adresáře k ukládání výsledků, jež se na vybraném paměťovém médiu nenachází, hláška 4 je vztažena k editačnímu poli relativního rozdílu mezi porovnávanými snímky a vyzývá uživatele k zadání z rozmezí od 0 do 1,000. U položek Poloměr filtru, Míra rozostření a Hodnota prahu se při zadání hodnoty mimo intervaly vyznačené šedými čísly nad posuvníky nebo nečíslného výrazu ukáží hlášky upozorňující uživatele ve stejném stylu jako hláška 4, tedy požadují vložení hodnot z příslušného rozmezí. Jestliže zadané hodnoty u poloměru filtru či prahu nejsou celá čísla, objeví se hláška 5 sdělující prosbu o zadání celého čísla. Jakmile se uživatel pokusí o spuštění programu tlačítkem „Start“ i přes špatně zadané hodnoty v editovacích polích, chybové hlášky program zastaví a navedou uživatele k vložení validních hodnot.

- a) 
- b) 
- c) 
- d) 
- e) 

Obrázek 4.9: Chybové hlášky implementované do GUI a) hláška 1, b) hláška 2, c) hláška 3, d) hláška 4, e) hláška 5

5 Výsledky a diskuze

Navržený program lze hodnotit dle více kritérií, výsledky této práce jsou zaměřeny na posouzení rychlosti vybraných algoritmů a spolehlivosti detekce změn v obrazu. Vyhodnocení rychlosti programu bylo provedeno na základě časového intervalu potřebného pro zpracování FullHD videosouboru o délce 60 s se snímkovou frekvencí 25 fps. Tabulky 5.1–5.4 prezentují hodnoty časové náročnosti, v sekundách, v závislosti na časovém rozmezí mezi zpracovávanými snímky videa a velikosti konvoluční masky použitých filtrů, jež se odvíjí od velikosti hodnoty σ , nastavení ostatních parametrů jako je hodnota prahu či relativnímu rozdílu mezi snímky jsou z hlediska časové závislosti nevýznamné, jelikož počet provedených operací neovlivňují. Měření pro každou dvojici parametrů bylo provedeno desetkrát, výsledné hodnoty jsou jejich aritmetickým průměrem. Počítačová sestava, na které byl program provozován, zahrnuje čtrnáctijádrový procesor Intel Core i9 10940X doplněný 64 GB operační paměti, vytížení procesoru se při činnosti programu pohybovalo pouze v rozmezí 6–7 %.

Tabulka 5.1: Časová náročnost na zpracování videosouboru o délce 60 s za využití Sobelova filtru [s]

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=0	112,599	42,861	28,554	16,413	16,475	17,707
K=3 ($\sigma=0,3$)	141,982	52,827	34,494	18,527	17,242	17,181
K=5 ($\sigma=0,6$)	142,021	53,052	34,613	18,815	17,749	17,600
K=7 ($\sigma=1$)	142,676	53,100	34,606	18,737	17,615	18,037
K=9 ($\sigma=1,3$)	144,983	53,855	35,156	19,129	17,699	17,784
K=11 ($\sigma=1,6$)	146,542	54,475	35,388	18,968	17,308	17,927
K=13 ($\sigma=2$)	149,214	55,241	35,984	19,455	17,829	17,172
K=15 ($\sigma=2,3$)	152,221	56,256	36,485	19,559	17,666	17,691
K=17 ($\sigma=2,6$)	155,669	57,336	36,917	19,606	17,748	17,906
K=19 ($\sigma=3$)	158,918	58,049	37,431	19,876	18,471	17,580
K=21 ($\sigma=3,3$)	161,814	58,941	37,920	20,049	18,255	18,213
K=21 ($\sigma=3,3$)¹	136,225	50,007	34,101	18,403	13,340	12,537

¹ Hodnoty získané aplikací Sobelova filtru implementovaného v rámci Image processing toolbox

Tabulka 5.2: Časová náročnost na zpracování videosouboru o délce 60 s za využití Kirschova filtru [s]

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=0	640,560	222,175	135,990	61,788	39,412	27,949
K=3 ($\sigma =0,3$)	673,761	232,669	141,111	63,680	40,302	28,177
K=5 ($\sigma =0,6$)	672,990	232,182	141,681	63,799	40,400	28,098
K=7 ($\sigma =1$)	673,965	231,254	140,826	63,480	40,709	28,695
K=9 ($\sigma =1,3$)	674,665	232,156	141,778	64,092	40,192	27,655
K=11 ($\sigma =1,6$)	675,545	232,691	141,850	63,755	40,535	28,031
K=13 ($\sigma =2$)	679,474	233,296	142,011	63,813	40,497	28,454
K=15 ($\sigma =2,3$)	680,783	234,013	142,526	63,999	40,722	28,222
K=17 ($\sigma =2,6$)	682,755	235,261	143,476	64,189	40,857	28,512
K=19 ($\sigma =3$)	688,249	236,756	143,954	64,486	40,924	28,457
K=21 ($\sigma =3,3$)	692,725	238,114	145,368	65,148	41,006	29,332

Tabulka 5.3: Časová náročnost na zpracování videosouboru o délce 60 s za využití LoG filtru [s]

Velikost konvoluční masky operátoru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	235,856	90,888	58,688	30,168	21,594	17,059
K=5 ($\sigma =0,6$)	246,571	90,827	58,416	29,982	21,560	17,085
K=7 ($\sigma =1$)	226,999	83,782	54,162	28,223	20,664	16,566
K=9 ($\sigma =1,3$)	228,984	83,185	52,913	27,850	21,441	19,023
K=11 ($\sigma =1,6$)	233,895	85,743	55,522	29,034	20,886	16,843
K=13 ($\sigma =2$)	254,012	92,673	59,667	30,492	21,741	17,013
K=15 ($\sigma =2,3$)	257,320	93,816	60,051	30,599	21,854	17,079
K=17 ($\sigma =2,6$)	261,024	94,900	60,476	30,743	21,711	17,083
K=19 ($\sigma =3$)	264,613	95,831	60,896	30,968	22,025	17,440
K=21 ($\sigma =3,3$)	269,964	97,019	61,801	31,120	22,051	17,232
K=21 ($\sigma =3,3$)²	221,438	79,247	50,040	26,713	16,969	14,585

² Hodnoty získané aplikací LoG filtru implementovaného v rámci Image processing toolbox

Tabulka 5.4: Časová náročnost na zpracování videosouboru o délce 60 s za využití Cannyho filtru [s]

Velikost konvoluční masky operátoru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma=0,3$)	451,493	156,987	97,055	45,880	31,163	23,878
K=5 ($\sigma=0,6$)	457,683	159,367	98,702	46,384	31,365	23,678
K=7 ($\sigma=1$)	456,153	159,241	98,417	46,384	31,533	24,152
K=9 ($\sigma=1,3$)	457,473	159,789	98,428	46,360	31,292	24,305
K=11 ($\sigma=1,6$)	461,036	160,715	99,319	46,698	31,550	23,895
K=13 ($\sigma=2$)	464,174	161,679	99,871	46,818	31,572	23,954
K=15 ($\sigma=2,3$)	468,296	163,234	101,556	47,084	31,572	23,334
K=17 ($\sigma=2,6$)	470,957	163,847	101,028	47,372	31,992	24,002
K=19 ($\sigma=3$)	475,543	165,115	102,144	47,327	31,919	23,646
K=21 ($\sigma=3,3$)	478,126	165,994	102,164	47,594	32,357	23,843
K=21 ($\sigma=3,3$)³	197,028	70,372	45,323	24,383	16,841	14,082

Z naměřených dat jednoznačně vyplývá, že časová náročnost na zpracování videa roste se snižujícím se časovým rozdílem mezi jednotlivými snímky, tedy se zvyšujícím se počtem zpracovávaných snímků. Velikost konvoluční masky použitých filtrů má při vyšším časovém rozestupu mezi snímky zanedbatelný vliv na rychlost programu, více se její účinek projevuje u hodnot menších než 0,5 s, což přímo souvisí s větším množstvím vyhodnocených snímků. U Sobelova a Kirschova operátoru je doba potřebná k provedení požadované činnosti při hodnotě $K = 0$ výrazně nižší, než u ostatních velikostí konvolučních masek, protože nedochází k použití Gaussova filtru. Rozdíl časové náročnosti mezi hodnotami $K = 3$ a $K = 21$ při zpracování každého snímku videa, za rozmezí 1/25 s, je u Sobelova operátoru 13,97 %, Kirschova operátoru 2,81 %, LoG operátoru 14,46 % a u Cannyho operátoru 5,90 %. V rámci porovnání rychlosti vybraných hranových detektorů vychází nejlépe Sobelův operátor s průměrnou dobou 149,6 s u rozmezí mezi snímky 1/25 s, a nejhůře Kirschův operátor, který v průměru potřeboval 679,5 s, průměrný časový úsek LoG operátoru je při stejných hodnotách 247,9 s, u Cannyho operátoru 464,1 s. Z uvedených hodnot je také patrné, že žádný z vybraných operátorů nelze využít pro vyhodnocení videosouboru

³ Hodnoty získané aplikací Cannyho filtru implementovaného v rámci Image processing toolbox

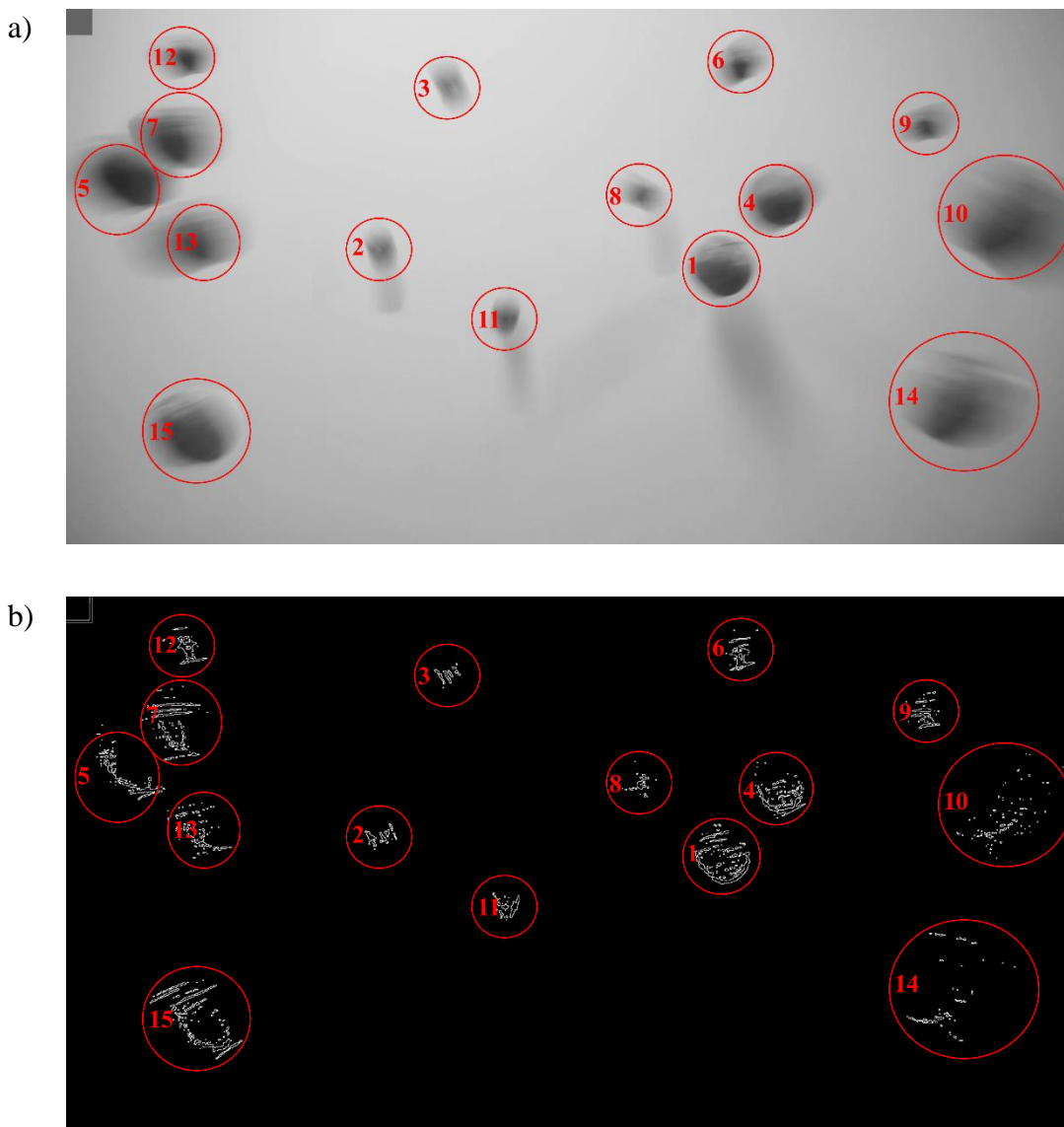
v reálném čase, pokud bude zpracován každý jeho snímek, Sobelův operátor by mohl být pro tuto činnost využit od rozmezí 0,1 s, Kirschův operátor až od rozmezí 1 s, LoG operátor od 0,2 s a Cannyho operátor od 0,5 s. Rychlost operátoru Prewittové nebyla měřena, protože provádí stejný počet operací jako Sobelův operátor, a tak rozdíl časové náročnosti mezi nimi by byl zanedbatelný, stejná situace je mezi Kirschovým a Robinsonovým filtrem. Odpovídající operátory implementované v rámci rozšiřující knihovny Image processing toolbox díky lepší optimalizaci vyžadují na provedení určených operací výrazně menší časové úseky, při jejich aplikaci došlo k vytížení procesoru na 16–18 %, což je téměř třikrát více, než při aplikaci operátorů vytvořených v rámci práce. Kromě optimálnějšího využití vícejádrového výkonu procesoru je vyšší rychlost činnosti operátorů také pravděpodobně podpořena nižší latencí přístupu k operační paměti. Uvedené důvody jsou však pouze domněnky, které nejsou podloženy reálnými daty.

Spolehlivost detekce změn v obrazu závisí na více faktorech, než je velikost konvoluční masky a rozmezí mezi zpracovávanými snímky, prahová hodnota operátorů i hodnota relativního rozdílu mezi porovnávanými snímky jsou rovněž velice důležité parametry, jejichž velikost je třeba stanovit experimentálně vzhledem k podobě vyhodnocovaných scén a vlastnostem objektů zapříčínující jejich změny. Hodnota vyjadřující spolehlivost vyhodnocení snímků je určena jako

$$s_p = \frac{I_s}{I_o + I_n}, \quad (5.1)$$

kde I_s je počet správně vyhodnocených snímků, I_o je počet snímků, ve kterých by měla být definována změna, a I_n vyznačuje počet nesprávně identifikovaných snímků. Při využití navrženého softwaru pro zpracování experimentálně získaných dat z projektu TRIO FV30234 nedošlo k získání relevantních výsledků z důvodu nevhodně navrženého pokusu, kdy v testované lokalitě nedocházelo k dostatečnému počtu střetů ptačtva se skleněnou překážkou, a tak byla vytvořena testovací videa určená ke stanovení spolehlivosti programu za daných podmínek simulují situaci, kdy na poměrně homogenní pozadí bez výrazných členitostí dopadají kulové objekty o průměru 5 a 10 cm, video č. 1 obsahuje objekty černé barvy z důvodu dosažení vysokého kontrastu vůči světlému pozadí, v testovacím videu č. 2 jsou šedé objekty, kontrast vzhledem k pozadí je tedy nižší. Pro plnou úspěšnost je třeba detekovat všech 15 objektů, které se postupně ve snímcích objevují, a zároveň nevyhodnotit jako správné i snímky,

kde se objekty nenachází nebo jsou objekty na nich zobrazené již detekovány. Na obrázku 5.1 jsou znázorněny polohy všech objektů dopadajících na pozadí v testovacím videu č. 2, u každého objektu je zaznamenáno pořadí, ve kterém se objekt ve videu vyskytl. Na snímcích je v levém horním rohu umístěn šedý čtverec, který zastává funkci referenční hrany, jenž je důležitá při normalizaci jednotlivých snímků probíhající při výstupu hranových operátorů. Při absenci referenční hrany dochází ke zvýraznění jinak nevýznamných hran, protože normalizace se odvíjí vždy od nejvyšší hodnoty gradientu v daném obraze, tedy v případě snímku obsahujícího pouze pozadí jsou detekovány hrany znázorňující nepatrné přechody mezi odstíny pozadí i přes jejich nízkou hodnotu gradientu oproti hodnotám gradientu na snímcích s výskytem objektů.



Obrázek 5.1: Znázornění polohy všech objektů vyskytujících se v testovacím videu č. 2 a) složení snímku bez úprav, b) složení snímku upravených vytvořenými algoritmy

Testovací videa byla zpracována pro časové rozmezí mezi jednotlivými snímky od 1/25 s do 2 s a velikosti konvoluční masky Gaussova filtru od 3 do 21. Parametr relativní rozdíl mezi snímky byl nastaven na hodnotu 0,01 pro všechny hranové detektory, stanovená prahová hodnota je 10 pro Sobelův, Kirschův i LoG operátor, pro Cannyho operátor byl určen dolní a horní práh o hodnotě 10 a 30.

Tabulka 5.5: Spolehlivost detekce změn v obrazu za využití Sobelova operátoru – video č. 1

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	71,4%	100,0%	100,0%	100,0%	80,0%	40,0%
K=5 ($\sigma =0,6$)	57,7%	100,0%	100,0%	100,0%	86,7%	40,0%
K=7 ($\sigma =1$)	57,7%	100,0%	100,0%	100,0%	86,7%	40,0%
K=9 ($\sigma =1,3$)	55,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=11 ($\sigma =1,6$)	55,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=13 ($\sigma =2$)	53,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=15 ($\sigma =2,3$)	57,7%	100,0%	100,0%	73,3%	46,7%	26,7%
K=17 ($\sigma =2,6$)	71,4%	93,3%	86,7%	53,3%	40,0%	20,0%
K=19 ($\sigma =3$)	78,9%	80,0%	66,7%	40,0%	33,3%	13,3%
K=21 ($\sigma =3,3$)	53,6%	100,0%	87,5%	66,7%	53,3%	20,0%

Tabulka 5.6: Spolehlivost detekce změn v obrazu za využití Kirschova operátoru – video č. 1

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	75,0%	100,0%	100,0%	93,3%	80,0%	40,0%
K=5 ($\sigma =0,6$)	71,4%	100,0%	100,0%	100,0%	80,0%	40,0%
K=7 ($\sigma =1$)	57,7%	100,0%	100,0%	100,0%	86,7%	40,0%
K=9 ($\sigma =1,3$)	55,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=11 ($\sigma =1,6$)	55,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=13 ($\sigma =2$)	51,7%	93,8%	93,8%	93,8%	86,7%	40,0%
K=15 ($\sigma =2,3$)	78,9%	86,7%	73,3%	53,3%	46,7%	26,7%
K=17 ($\sigma =2,6$)	62,5%	87,5%	93,3%	66,7%	40,0%	20,0%
K=19 ($\sigma =3$)	82,4%	66,7%	53,3%	40,0%	33,3%	13,3%
K=21 ($\sigma =3,3$)	53,6%	100,0%	93,8%	66,7%	53,3%	20,0%

Tabulka 5.7 Spolehlivost detekce změn v obrazu za využití LoG operátoru – video č. 1

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%
K=5 ($\sigma =0,6$)	93,3%	93,3%	80,0%	66,7%	53,3%	26,7%
K=7 ($\sigma =1$)	9,9%	11,1%	16,3%	12,1%	20,3%	15,4%
K=9 ($\sigma =1,3$)	60,0%	100,0%	100,0%	100,0%	86,7%	40,0%
K=11 ($\sigma =1,6$)	68,2%	100,0%	100,0%	100,0%	80,0%	40,0%
K=13 ($\sigma =2$)	88,2%	100,0%	100,0%	80,0%	80,0%	40,0%
K=15 ($\sigma =2,3$)	83,3%	100,0%	93,3%	80,0%	80,0%	40,0%
K=17 ($\sigma =2,6$)	83,3%	100,0%	93,3%	86,7%	80,0%	40,0%
K=19 ($\sigma =3$)	88,2%	100,0%	93,3%	80,0%	73,3%	33,3%
K=21 ($\sigma =3,3$)	83,3%	100,0%	93,3%	86,7%	80,0%	40,0%

Tabulka 5.8: Spolehlivost detekce změn v obrazu za využití Cannyho operátoru – video č. 1

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	88,2%	100,0%	100,0%	80,0%	80,0%	40,0%
K=5 ($\sigma =0,6$)	88,2%	100,0%	100,0%	80,0%	80,0%	40,0%
K=7 ($\sigma =1$)	78,9%	100,0%	100,0%	100,0%	80,0%	40,0%
K=9 ($\sigma =1,3$)	68,2%	100,0%	100,0%	100,0%	86,7%	40,0%
K=11 ($\sigma =1,6$)	68,2%	100,0%	100,0%	80,0%	80,0%	40,0%
K=13 ($\sigma =2$)	88,2%	100,0%	100,0%	80,0%	80,0%	40,0%
K=15 ($\sigma =2,3$)	57,7%	100,0%	100,0%	100,0%	86,7%	40,0%
K=17 ($\sigma =2,6$)	55,6%	100,0%	100,0%	100,0%	86,7%	40,0%
K=19 ($\sigma =3$)	51,7%	100,0%	100,0%	100,0%	86,7%	40,0%
K=21 ($\sigma =3,3$)	51,7%	100,0%	100,0%	100,0%	86,7%	40,0%

Výstup programu při daných hodnotách u vybraného videa lze hodnotit jako poměrně spolehlivý při časovém rozmezí mezi zpracovávanými snímky 0,1, 0,2 a 0,5 s, kdy u vybraných hranových detektorů byly změny obrazu vyhodnoceny naprosto správně nejméně v deseti případech u každého z nich v závislosti na velikosti konvoluční masky Gaussova filtru ovlivňující vliv vyhlazení obrazu. Spolehlivost výsledků

za časového rozmezí 1/25 s je u většiny vybraných filtrů nízká i přes správnou detekci všech objektů z důvodu nesprávného vyhodnocení změn ve větším množství snímků, například u Sobelova operátoru při hodnotě $K = 13$ byla změna detekována u 28 snímků místo 15, extrémní výkyv nastal u LoG operátoru při hodnotě $K = 7$, kdy došlo k detekci změn u 140 snímků. Jednou z hlavních příčin je s vysokou pravděpodobností vysoký počet po sobě jdoucích snímků, ve kterých se vyskytuje stejný předmět s odlišnou polohou plynoucí z jeho pohybu. Dalším zdrojem těchto nesrovnalostí může být nízká nastavená prahová hodnota, u které nedochází k odfiltrování méně výrazných hran vznikajících rozmazáním objektů jejich rychlým pohybem. U časového rozmezí 2 s je také přesnost správného vyhodnocení nízká, celkem vyhodnocených snímků je malé množství, což je způsobeno velikostí časových prodlev mezi snímky, na zpracovávaných snímcích se tak objekt k detekci velmi často neobjevuje, jelikož snímky na kterých se objekt nachází, mohou spadat do časové mezery. Při zprůměrování všech zjištěných hodnot u jednotlivých operátorů vychází jako nejspolehlivější Cannyho operátor s průměrnou spolehlivostí 80,8 %, jako nejméně spolehlivý se jeví LoG operátor se spolehlivostí 64,6 %, avšak tato hodnota je zkreslena nulovou úspěšností při velikosti $K = 3$, kdy hodnoty konvoluční masky LoG nabývají velmi vysokého rozsahu, a tak nejsou nalezeny žádné hrany ve všech snímcích, tedy není detekována žádná změna obrazu. Průměrná úspěšnost Sobelova a Kirschova operátoru činí 72,8 % a 71,7 %.

Ověření hodnot spolehlivosti proběhlo stejným způsobem také pro video č. 2, ve kterém se náhodně objevují objekty šedé barvy a tak kontrast mezi nimi a pozadím je nižší než u videa č. 1, výskyt objektů z hlediska jejich množství je stejný jako u předchozího testovacího videa, avšak liší se v čase a pořadí. Výsledky v tabulkách 5.9–5.12 s jistou podobností korespondují s hodnotami získaných evaluací změn v obrazu u objektů s vysokým kontrastem, s tím rozdílem, že nižší spolehlivost se projevuje již od rozdílu mezi snímky 0,5 s. Tuto skutečnost lze připisovat jinému načasování vpádu jednotlivých objektů do záběru, dalším vysvětlením může být jejich pohyb, kdy pomyslná stopa, vzniklá rozmazáním objektů při změně pozice v čase, nabývá světlejších odstínů šedi než stopa tmavších objektů, a tak i kontrast mezi ní a pozadím je nižší. Operátorem vykazujícím nejvyšší spolehlivost při určení změn je Kirschův operátor s průměrnou hodnotou 70,1 %, s minimálním odstupem se za něj řadí Sobelův a Cannyho operátor s průměrnými hodnotami 70,0 % a 69,7 %, nejhorší výsledek

podal LoG operátor s průměrnou spolehlivostí 49,7 %, který však v nižších časových rozmezích a výraznějších vyhlazení obrazu poskytuje velmi dobré výsledky.

Tabulka 5.9: Spolehlivost detekce změn v obrazu za využití Sobelova operátoru – video č. 2

Velikost konvo- luční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	83,3%	100,0%	100,0%	53,3%	46,7%	26,7%
K=5 ($\sigma =0,6$)	75,0%	100,0%	100,0%	60,0%	60,0%	33,3%
K=7 ($\sigma =1$)	71,4%	100,0%	100,0%	60,0%	66,7%	40,0%
K=9 ($\sigma =1,3$)	55,6%	100,0%	100,0%	80,0%	66,7%	40,0%
K=11 ($\sigma =1,6$)	55,6%	100,0%	100,0%	80,0%	66,7%	40,0%
K=13 ($\sigma =2$)	51,7%	100,0%	100,0%	86,7%	66,7%	40,0%
K=15 ($\sigma =2,3$)	71,4%	100,0%	93,3%	46,7%	33,3%	20,0%
K=17 ($\sigma =2,6$)	83,3%	100,0%	100,0%	33,3%	53,3%	33,3%
K=19 ($\sigma =3$)	83,3%	100,0%	100,0%	33,3%	60,0%	33,3%
K=21 ($\sigma =3,3$)	62,5%	100,0%	100,0%	60,0%	60,0%	33,3%

Tabulka 5.10: Spolehlivost detekce změn v obrazu za využití Kirschova operátoru – video č. 2

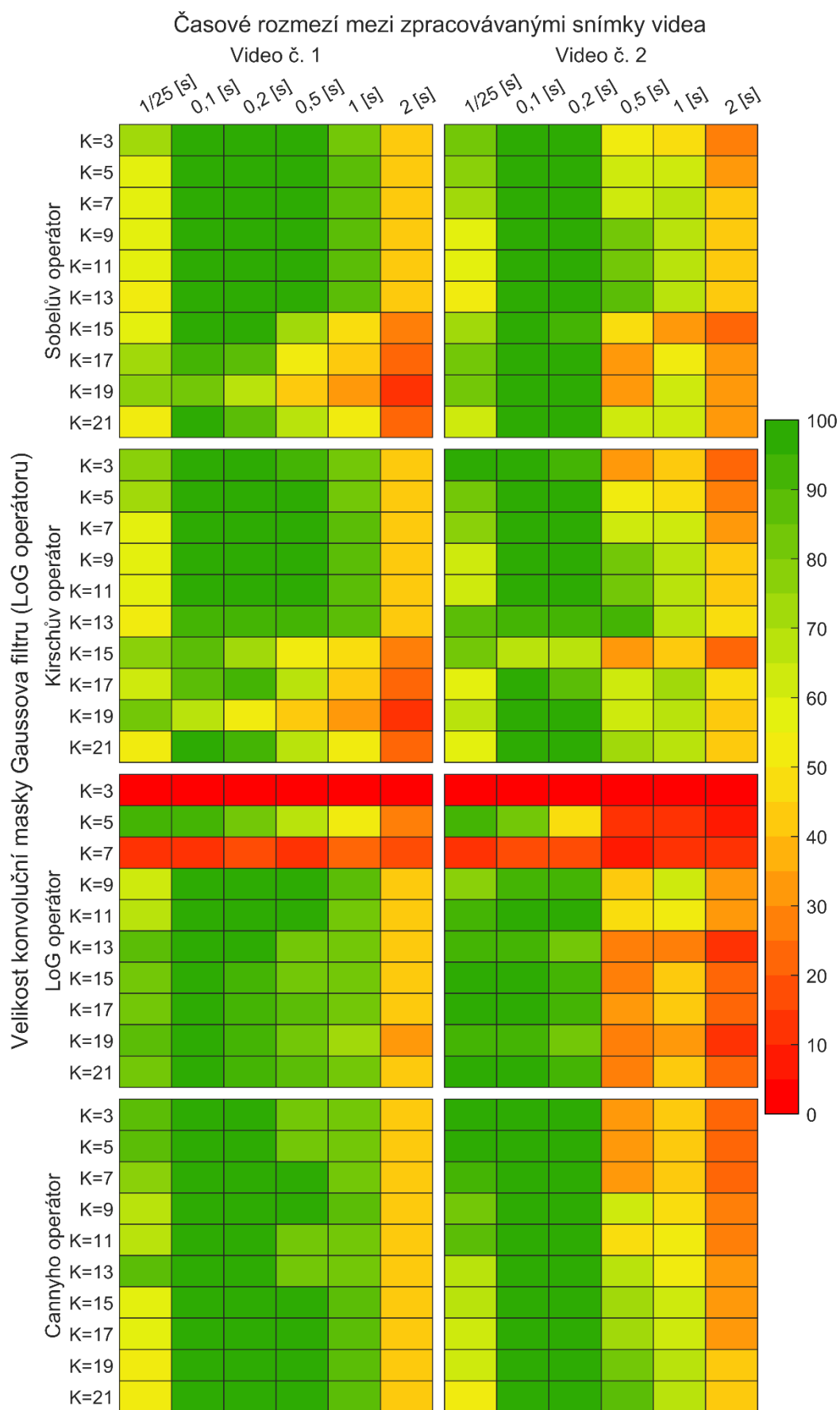
Velikost konvo- luční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	100,0%	100,0%	93,3%	33,3%	40,0%	20,0%
K=5 ($\sigma =0,6$)	83,3%	100,0%	100,0%	53,3%	46,7%	26,7%
K=7 ($\sigma =1$)	75,0%	100,0%	100,0%	60,0%	60,0%	33,3%
K=9 ($\sigma =1,3$)	62,5%	100,0%	100,0%	80,0%	66,7%	40,0%
K=11 ($\sigma =1,6$)	62,5%	100,0%	100,0%	80,0%	66,7%	40,0%
K=13 ($\sigma =2$)	86,2%	93,8%	93,8%	93,3%	66,7%	46,7%
K=15 ($\sigma =2,3$)	81,3%	66,7%	66,7%	33,3%	40,0%	20,0%
K=17 ($\sigma =2,6$)	55,6%	100,0%	88,2%	60,0%	73,3%	46,7%
K=19 ($\sigma =3$)	68,2%	100,0%	100,0%	60,0%	66,7%	40,0%
K=21 ($\sigma =3,3$)	57,7%	100,0%	100,0%	73,3%	66,7%	40,0%

Tabulka 5.11: Spolehlivost detekce změn v obrazu za využití LoG operátoru – video č. 2

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%
K=5 ($\sigma =0,6$)	93,3%	80,0%	46,7%	13,3%	13,3%	6,7%
K=7 ($\sigma =1$)	10,7%	17,2%	19,5%	7,7%	13,2%	12,2%
K=9 ($\sigma =1,3$)	78,9%	93,8%	93,8%	43,8%	60,0%	33,3%
K=11 ($\sigma =1,6$)	93,8%	100,0%	100,0%	46,7%	53,3%	33,3%
K=13 ($\sigma =2$)	93,3%	93,3%	80,0%	26,7%	26,7%	13,3%
K=15 ($\sigma =2,3$)	100,0%	100,0%	93,3%	26,7%	40,0%	20,0%
K=17 ($\sigma =2,6$)	100,0%	100,0%	93,3%	33,3%	40,0%	20,0%
K=19 ($\sigma =3$)	93,3%	93,3%	80,0%	26,7%	33,3%	13,3%
K=21 ($\sigma =3,3$)	100,0%	100,0%	93,3%	26,7%	40,0%	20,0%

Tabulka 5.12: Spolehlivost detekce změn v obrazu za využití Cannyho operátoru – video č. 2

Velikost konvoluční masky Gaussova filtru	Časové rozmezí mezi zpracovávanými snímky videa					
	1/25 [s]	0,1 [s]	0,2 [s]	0,5 [s]	1 [s]	2 [s]
K=3 ($\sigma =0,3$)	100,0%	100,0%	100,0%	33,3%	40,0%	20,0%
K=5 ($\sigma =0,6$)	100,0%	100,0%	100,0%	33,3%	40,0%	20,0%
K=7 ($\sigma =1$)	93,8%	100,0%	100,0%	33,3%	40,0%	20,0%
K=9 ($\sigma =1,3$)	83,3%	100,0%	100,0%	60,0%	46,7%	26,7%
K=11 ($\sigma =1,6$)	88,2%	100,0%	100,0%	46,7%	53,3%	26,7%
K=13 ($\sigma =2$)	65,2%	100,0%	100,0%	66,7%	53,3%	33,3%
K=15 ($\sigma =2,3$)	68,2%	100,0%	100,0%	73,3%	60,0%	33,3%
K=17 ($\sigma =2,6$)	62,5%	100,0%	100,0%	73,3%	60,0%	33,3%
K=19 ($\sigma =3$)	60,0%	100,0%	100,0%	80,0%	66,7%	40,0%
K=21 ($\sigma =3,3$)	53,6%	100,0%	100,0%	86,7%	66,7%	40,0%



Obrázek 5.2: Grafické znázornění výsledných procentuálních hodnot spolehlivosti vytvořeného programu pro detekci změn v obraze při aplikaci na vybrané testovací videosoubory

Závěr

Vyvinutý software využívá vybrané základní metody zpracování a segmentace obrazu, jenž jsou pro prvotní práci s obrazovými daty velmi důležité, avšak postupů určeným k jejich úpravám a následnému získání požadovaných informací je široké spektrum, z nichž se diplomová práce zabývá pouze zlomkem z nich. K dalšímu rozpracování a implementaci do programu lze navrhnout například metody zakládající na úpravách histogramu, adaptivní prahování, K-means clusterizaci či jiné pokročilejší techniky segmentace, případně aplikovat některý z příznakových detektorů. Z hlediska využitelnosti se vytvořený program jeví jako obstojný nástroj k detekci změn v obrazu, ale je třeba vzít ohled na relevantní nastavení daných parametrů vzhledem k vlastnostem videosouboru i detekovaných objektů.

Seznam použité literatury

Akram, M. W. et al. (2019). Improved outdoor thermography and processing of infrared images for defect detection in PV modules. *Solar Energy*, 190(4):549–560.

Asdrubali, F. et al. (2018). Detection of thermal bridges from thermographic images by means of image processing approximation algorithms. *Applied Mathematics and Computation*, 317:160–171.

Bloch, V. et al. (2020). Automatic broiler temperature measuring by thermal camera. *Biosystems Engineering*, 199:127–134.

Bruno, L. et al. (2012). Improved Traffic Signal Detection a Classification via Image Processing Algorithms. *Procedia - Social a Behavioral Sciences*, 53:810–820

Bugarinović, Ž. et al. (2020). On the introduction of canny operator in an advanced imaging algorithm for real-time detection of hyperbolas in ground-penetrating radar data. *Electronics*, 9(3).

Burnham, J. et al. (1997). A Comparison of the Roberts, Sobel, Robinson, Canny, a Hough Image Detection Algorithms. In: *MS State DSP Conference*, Mississippi, USA, pp. 13.

Devi, H. S. a Singh, K. M. (2020). Red-cyan anaglyph image watermarking using DWT, Hadamard transform a singular value decomposition for copyright protection. *Journal of Information Security a Applications*, 50:1–7.

Dewangan, S. a Sharma, K. A. (2017). Image Smoothing a Sharpening using Frequency Domain Filtering Technique. *International Journal of Emerging Technologies in Engineering Research*, 5(4):169–174.

Dobeš, M. (2008). *Zpracování obrazu a algoritmy v C#*. Ben, Praha. ISBN: 978-80-7300-233-6.

Dogra, A. a Bhalla, P. (2014). Image sharpening by Gaussian a Butterworth high pass filter. *Biomedical a Pharmacology Journal*, 7(2):707–713.

Flusser, J. et al. (2016). Recognition of images degraded by Gaussian blur. *IEEE Transactions on Image Processing*, 25(2):790–806.

Forsyth, A. D. a Ponce, J. (2012). *Computer Vision: A Modern Approach*. Second edition. Pearson, New Jersey. ISBN: 978-0-13-608592-8.

Gasteratos, A. a Andreadis, I. (2000). Non-linear image processing in hardware. *Pattern Recognition*, 33(6):1013–1021.

Gironés, X. a Julià, C. (2020). Real-time localization of multi-oriented text in natural scene images using a linear spatial filter. *Journal of Real-Time Image Processing*, 17(5):1505–1525.

Gonzales, R. C. a Woods R. E. (2002). *Digital Image Processing*. Second Edition. Prentice Hall, New Jersey. ISBN 0-201-18075-8.

Gonzales, R. C. et al. (2009). *Digital Image Proicessing using MATLAB*. Second Edition. Gatesmark Publishing, Knoxville. ISBN: 978-0-9820854-0-0.

Heras, V. *et al.* (2019). Urban heritage monitoring, using image processing techniques a data collection with terrestrial laser scanner (tls), case study cuenca-Ecuador. *ISPRS Annals of the Photogrammetry, Remote Sensing a Spatial Information Sciences*, 42(2):609–613.

Hlaváč, V. a Sedláček, M. (2009). *Zpracování signálů a obrazů*. Vyd. 3. ČVUT, Praha. ISBN 978-80-01-04442-1.

Hou, Y. et al. (2019). Block-Extraction a Haar Transform Based Linear Singularity Representation for Image Enhancement. *Mathematical Problems in Engineering*, 2019.

Chen, W. *et al.* (2020). Lane departure warning systems a lane line detection methods based on image processing a semantic segmentation: A review. *Journal of Traffic a Transportation Engineering (English Edition)*, 7(6):748–774.

Chu, E. a George, A. (1998). FFT algorithms a their adaptation to parallel processing. *Linear Algebra a Its Applications*, 284:95–124.

Iqbal, Z. et al. (2018) An automated detection a classification of citrus plant diseases using image processing techniques: A review. *Computers a Electronics in Agriculture*. Elsevier, 153(8):12–32.

Jain, R. et al. (1995). *Machine Vision*. McGraw-Hill, New York. ISBN: 0-07-032018-7.

Jensen, J. R. (2015). *Introductory Digital Image Processing: A Remote Sensing Perspective*. 4th Edition. Pearson, Londýn. ISBN-13: 978-0-13-405816-0.

Jeong, S. et al. (2021). Multi-Regime Analysis for Computer Vision- Based Traffic Surveillance Using a Change-Point Detection Algorithm, 9:40980–40995.

Jha, R. K. a Swami, P. D. (2020). Intelligent fault diagnosis of rolling bearing a gear system under fluctuating load conditions using image processing technique. *Journal of Mechanical Science a Technology*, 34(10):4107–4115.

Karasulu, B. (2012). Automatic Extraction of Retinal Blood Vessels: a Software Implementation. *European Scientific Journal*, 8(30):47–57.

Kavitha, S. a Rajeswari, R. (2014). Evolutionary Algorithms for Edge Detection. *IJERT*, 2(5):137–141.

Klíč, A. et al. (2012). *Fourierova transformace s příklady z infračervené spektroskopie*. VŠČHT, Praha. ISBN 978-80-7080-478-0.

Kuo, C. F. J. et al. (2020). Applied image processing techniques in video laryngoscope for occult tumor detection. *Biomedical Signal Processing a Control*, 55.

Kurka, P. R. G. a Salazar, A. A. D. (2019). Applications of image processing in robotics a instrumentation. *Mechanical Systems a Signal Processing*, 124:142–169.

Li, G. et al. (2020). Analysis of feeding a drinking behaviors of group-reared broilers via image processing. *Computers a Electronics in Agriculture*. Elsevier, 175(5).

Liu, L. et al. (2018). Ship infrared image edge detection based on an improved adaptive Canny algorithm. *International Journal of Distributed Sensor Networks*, 14(3).

Magnier, B. et al. (2018). A review of supervised edge detection evaluation methods a an objective comparison of filtering gradient computations using hysteresis thresholds. *Journal of Imaging*, 4(6).

Marr, D. a Hildreth, E. (1980). Theory of edge detection. *Proc. R. Soc. Lond. B.*, 207(1167):187–217.

Monteiro, R. de C. M. *et al.* (2020). Image processing to identify damage to soybean seeds. *Ciencia Rural*, 51(2):1–8.

Mousavi, S. M. H. *et al.* (2019). Analysis of a robust edge detection system in different color spaces using color a depth images. *Computer Optics*, 43(4):632–646.

Nema, R. a Saxena, A. K. (2013). Edge Detection Operators on Digital Image. *IJESRT*, 2(6):1596–1601.

Nixon, M. a Aguado, A. (2002). *Feature Extraction and Image Processing*. Newnes, Oxford. ISBN: 0-7506-5078-8.

Padmavathi, G. *et al.* (2009). Performance analysis of non linear filtering algorithms for underwater images. *International Journal of Computer Science and Information Security*, 6(2):179–184.

Pathmanabhan, A. a Dinesh, S. (2007). The effect of Gaussian blurring on the extraction of peaks a pits from digital elevation models. *Discrete Dynamics in Nature a Society*, 2007(1).

Petrou, M. M. P. a Petrou C. (2010). *Image Processing: The Fundamentals*. Second Edition. Wiley, Chichester. ISBN: 978-0-470-74586-1.

Ramani, R. G. a Shanthamalar, J. J. (2020). Improved image processing techniques for optic disc segmentation in retinal fundus images. *Biomedical Signal Processing a Control*, 58.

de Ruijter, J. *et al.* (2020). Automated 3D geometry segmentation of the healthy a diseased carotid artery in free-ha, probe tracked ultrasound images. *Medical Physics*, 47(3):1034–1047.

Russ, J. C. (2002). *The Image Processing Handbook*. Fourth Edition. CRC Press, Boca Raton. ISBN: 0-8493-1142-X.

Russ, J. C. a Russ J. Ch. (2008). *Image Processing and Analysis*. CRC Press, Boca Raton. ISBN: 978-0-8493-7073-1

Sangnoree, A. a Chamnongthai, K. (2017). Thermal-image processing and statistical analysis for vehicle category in nighttime traffic. *Journal of Visual Communication and Image Representation*, 48:88-109.

Shaikh, S. et al. (2016). Analysis of Digital Image Filters in Frequency Domain. *International Journal of Computer Applications*, 140(6):12–19.

Smith, O. J. (2007). *Mathematics of the Discrete Fourier Transform (DFT), with Audio Applications*. Second Edition. W3K Publishing, Stanford. ISBN 978-0-9745607-4-8.

Smith, S. W. (1999). *The Scientist and Engineer's Guide to Digital Signal Processing*. Second Edition. California Technical Publishing, San Diego. ISBN: 0-9660176-7-6.

Sojka, E. et al. (2011). *Matematické základy digitálního zpracování obrazu*. VŠB-TU Ostrava, Ostrava.

Sojka, E. (2000). *Digitální zpracování a analýza obrazů, učební texty*. VŠB-TU Ostrava, Ostrava. ISBN: 80-7078-746-5.

Sonka, M. et al. (2008). *Image Processing. Analysis. and Machine Vision*. Third Edition. *International Student Edition*. Thomson Learning, Toronto. ISBN: 978-0-495-24428-7.

Tabassum, A. et al. (2013). Implementation of Canny Edge Detection Algorithm on FPGA a displaying Image through VGA Interface. *IJARIE*, 2(5):139–143.

Tong, J. et al. (2018). Skewness correction a quality evaluation of plug seedling images based on Canny operator a Hough transform. *Computers a Electronics in Agriculture*, 155(9):461–472.

Vishnoi, V. K. et al. (2021) Plant disease detection using computational intelligence a image processing. *Journal of Plant Diseases a Protection*, 128:19–53.

Wang, Z. et al. (2017). Review of Plant Identification Based on Image Processing. *Archives of Computational Methods in Engineering*, 24(3):637–654.

Wen, D. M. et al. (2019) Use of thermal imaging and Fourier transform infrared spectroscopy for the pre-symptomatic detection of cucumber downy mildew. *European Journal of Plant Pathology*, 155(2):405–416.

Win, N. N. *et al.* (2019). Image Noise Reduction Using Linear a Nonlinear Filtering Techniques. *International Journal of Scientific a Research Publications (IJSRP)*, 9(8):816–821.

Yoo, T. S. (2004). *Insight into Images: Principles and Practice for Segmentation, Registration, and Image Analysis*. AK Peters Ltd, Wellesey. ISBN:978-1-56881-217-5

Zawaideh, F. H., et al. (2017). Comparison between Butterworth a Gaussian High-pass Filters using an Enhanced Method. *IJCSNS International Journal of Computer Science a Network Security*, 17(7):113–117.

Zhang, Y-J. (2021). *Handbook of Image Engineering*. Springer, Singapore. ISBN 978-981-15-5873-3.

Zhao, K. et al. (2018). Automatic lameness detection in dairy cattle based on leg swing analysis with an image processing technique. *Computers a Electronics in Agriculture*, 148(4):226–236.

Zhu, J. et al. (2021). Automobile tire life prediction based on image processing a machine learning technology. *Advances in Mechanical Engineering*, 13(3):1–13.

Seznam obrázků

Obrázek 1.1: Znázornění průběhu konvoluce	10
Obrázek 1.2: Rozšíření vstupního obrazu a) vytvořením černého rámu, b) zrcadlovým prodloužením, c) periodickým prodloužením obrazu, d) replikací hodnot krajních pixelů	11
Obrázek 1.3: Konvoluční masky filtru o velikosti a) 3×3 a b) 5×5	12
Obrázek 1.4: Konvoluční masky filtru váženého průměru, a) zápis dle Jain et al. (1995), b) zápis dle Gonzales a Woods (2002).....	13
Obrázek 1.5: Obraz po aplikaci filtrace průměrováním a) vstupní obraz bez vyhlazení, b) kernel o velikosti 5×5 , c) kernel 15×15 , d) kernel 25×25	13
Obrázek 1.6: Grafy Gaussova normálního rozdělení pro σ o velikosti 1 a 2.....	14
Obrázek 1.7: Graf kernelu Gaussova filtru, kdy a) $\sigma=3$, b) $\sigma=5$	16
Obrázek 1.8: Hodnoty kernelu Gaussova filtru pro $\sigma=0,5$ a) vypočítané dle vztahu 1.14, b) vypočítané dle vztahu 1.19, c) hodnoty kernelu a) po jeho normalizování, d) hodnoty kernelu b) po jeho normalizování	17
Obrázek 1.9: Obraz po aplikaci Gaussova filtru a) vstupní obraz bez vyhlazení, b) $\sigma=0,5$, c) $\sigma=5$, d) $\sigma=9$	18
Obrázek 1.10: Hodnoty kernelu Laplaceova operátoru pro a) základní kernel pro čtyřokolí, b) opačné hodnoty základního kernelu, c) rozšířený kernel pro osmiokolí	20
Obrázek 1.11: Upravená konvoluční maska Laplaceova operátoru pro a) čtyřokolí, b) osmiokolí.....	21
Obrázek 1.12: Obraz po aplikaci Laplaceova operátoru zostření, a) vstupní obraz bez vyhlazení, b) upravená konvoluční maska pro čtyřokolí, c) upravená konvoluční maska pro osmiokolí	21
Obrázek 1.13: Schématické znázornění základního principu mediánového filtru.....	23
Obrázek 1.14: Obraz po aplikaci mediánového filtru a) vstupní obraz s aditivním šumem typu sůl a pepř, b) medián z oblasti okolních bodů o velikosti 5×5 , c) medián z oblasti okolních bodů 7×7 , d) medián z oblasti okolních bodů 15×15	24

Obrázek 2.1: Znázornění funkce složené ze součtu řad funkcí sinus a kosinus	25
Obrázek 2.2: 3D vizualizace Fourierovi transformace	26
Obrázek 2.3: Znázornění realizace dvourozměrné diskrétní Fourierovy transformace postupnou aplikací jednorozměrných DFT.....	31
Obrázek 2.4: Dvourozměrná DFT obrazu, a) vstupní obraz, b) výkonové spektrum, c) fázové spektrum, d) výkonové spektrum po logaritmické transformaci hodnot.....	31
Obrázek 2.5: Schematické znázornění postupu při filtraci obrazu ve frekvenční doméně za využití Fourierovy transformace	34
Obrázek 2.6: Grafy low-pass filtrů pro $D_0=30$ a) Ideální low-pass filtr, b) Butterworthův low-pass filtr, $n=2$, c) Gaussův low-pass filtr	36
Obrázek 2.7: Aplikace low-pass filtrů, a) vstupní obraz, b) výkonové spektrum vstupního obrazu, c) znázornění ILPF, $D_0=100$, d) výsledek aplikace ILPF na výkonové spektrum obrazu, e) výsledný obraz po inverzní DFT, f) znázornění BLPF, $D_0=100$, $n=2$, g) výsledek aplikace BLPF na výkonové spektrum obrazu, h) výsledný obraz po inverzní DFT, i) znázornění GLPF, $D_0=100$, j) výsledek aplikace GLPF na výkonové spektrum obrazu, k) výsledný obraz po inverzní DFT	37
Obrázek 2.8: Grafy high-pass filtrů pro $D_0=30$ a) Ideální high-pass filtr, b) Butterworthův high-pass filtr, $n=2$, c) Gaussův high-pass filtr	39
Obrázek 2.9: Aplikace high-pass filtrů, a) vstupní obraz, b) výkonové spektrum vstupního obrazu, c) znázornění IHPF, $D_0=100$, d) výsledek aplikace IHPF na výkonové spektrum obrazu, e) výsledný obraz po inverzní DFT, f) znázornění BHPF, $D_0=100$, $n=2$, g) výsledek aplikace BHPF na výkonové spektrum obrazu, h) výsledný obraz po inverzní DFT, i) znázornění GHPF, $D_0=100$, j) výsledek aplikace GHPF na výkonové spektrum obrazu, k) výsledný obraz po inverzní DFT	40
Obrázek 2.10: Grafy notch filtrů pro $D_0=22$ a) Ideální notch filtr, b) Butterworthův notch filtr, $n=2$, c) Gaussův notch filtr.....	42
Obrázek 2.11: Znázornění notch a band-pass filtrů $D_0=22$, $w=25$, a) Ideální notch filtr, b) Butterworthův notch filtr, $n=2$, c) Gaussův notch filtr,	

d) Ideální radiální band-pass filtr, e) Butterworthův radiální band-pass filtr, n=2, f) Gaussův radiální band-pass filtr.....	43
Obrázek 2.12: Grafy radiálních band-pass filtrů pro $D_0=22$, $w=25$, a) Ideální radiální band-pass filtr, b) Butterworthův radiální band-pass filtr, n=2, c) Gaussův radiální band-pass filtr	44
Obrázek 3.1: Znárodnění typů hran, a) skoková hrana, b) šikmá hrana; c) liniiová hrana, d) střechová hrana, e) zašuměná hrana	46
Obrázek 3.2: a) Vzorový obraz s vyznačenými jednorozměrnými průřezy, b) grafy průběhu jasu vybraných průřezů obrazu, c) obraz po aplikaci Gaussova filtru s parametrem $\sigma=3$, d) grafy průběhu jasu vybraných průřezů obrazu po aplikaci Gaussova filtru	47
Obrázek 3.3: a) Ilustrace přechodu jasu, b) znárodnění obrazové funkce přechodu jasu, c) první derivace obrazové funkce, d) druhá derivace obrazové funkce.....	48
Obrázek 3.4: Hodnoty konvolučních masek Robertsova operátoru pro a) detekci horizontálních hran, b) detekci vertikálních hran	50
Obrázek 3.5: Hodnoty konvolučních masek operátoru Prewittové pro a) detekci horizontálních hran, b) detekci vertikálních hran	51
Obrázek 3.6: Obraz po aplikaci operátoru Prewittové a) vstupní obraz, b) výsledný obraz pro práh o hodnotě 10, c) výsledný obraz pro práh o hodnotě 30, d) výsledný obraz pro práh o hodnotě 50	52
Obrázek 3.7: Hodnoty konvolučních masek Sobelova operátoru pro a) detekci horizontálních hran, b) detekci vertikálních hran	52
Obrázek 3.8: Znárodnění značení okolních bodů vzhledem k vybranému bodu	53
Obrázek 3.9: Obraz po aplikaci Sobelova operátoru, a) vstupní obraz, b) výsledný obraz pro práh o hodnotě 10, c) výsledný obraz pro práh o hodnotě 30, d) výsledný obraz pro práh o hodnotě 50	53
Obrázek 3.10: Hodnoty konvolučních masek Robinsonova operátoru pro a) směr sever, b) severovýchod, c) východ, d) jihovýchod, e) jih, f) jihozápad, g) západ, h) severozápad	54
Obrázek 3.11: Obraz po aplikaci Robinsonova operátoru, a) výsledný obraz pro práh o hodnotě 10, b) výsledný obraz pro práh o hodnotě 50	54

Obrázek 3.12: Hodnoty konvolučních masek Kirschova operátoru pro a) směr sever, b) severovýchod, c) východ, d) jihovýchod, e) jih, f) jihozápad, g) západ, h) severozápad	55
Obrázek 3.13: Obraz po aplikaci Kirschova operátoru, a) výsledný obraz pro práh o hodnotě 10, b) výsledný obraz pro práh o hodnotě 50	55
Obrázek 3.14: Obrázek 3.15: Obraz po aplikaci Cannyho operátoru, a) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=1$, b) výsledný obraz pro dolní a horní práh o hodnotách 50 a 80, $\sigma=1$ c) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=2$, d) výsledný obraz pro dolní a horní práh o hodnotách 10 a 80, $\sigma=3$	58
Obrázek 3.16: Graf konvoluční masky LoG filtru, kdy a) $\sigma=2$, b) $\sigma=2,5$	60
Obrázek 3.17: Obraz po aplikaci operátoru LoG, a) výsledný obraz pro $\sigma=1$, b) výsledný obraz pro $\sigma=2$, c) výsledný obraz pro $\sigma=3$	61
Obrázek 4.1: Základní schéma algoritmu na detekci změn v obrazu	62
Obrázek 4.2: Vývojový diagram algoritmu na detekci změn v obrazu–část 1	67
Obrázek 4.3: Vývojový diagram algoritmu na detekci změn v obrazu–část 2	68
Obrázek 4.4: Grafické uživatelské prostředí programu pro detekci změn v obrazu ..	81
Obrázek 4.5: Ukázka nastavení vybraných parametrů pro a) jednostavové tlačítko Otevřít, b) popup menu pro výběr časového rozmezí mezi zpracovávanými snímky videa.....	82
Obrázek 4.6: Dialogové okno pro výběr videosouboru podporovaného formátu.....	83
Obrázek 4.7: Znázornění vzájemné vazby posuvníku a příslušného editovacího pole.....	84
Obrázek 4.8: Znázornění funkce zaškrtačovacího pole v prostředí programu a) neoznačené pole s výslednou deaktivací náležitých objektů, b) označené pole zapříčiňující zobrazení příslušných objektů	85
Obrázek 4.9: Chybové hlášky implementované do GUI a) hláška 1, b) hláška 2, c) hláška 3, d) hláška 4, e) hláška 5	86
Obrázek 5.1: Znázornění polohy všech objektů vyskytujících se v testovacím videu č. 2 a) složení snímků bez úprav, b) složení snímků upravených vytvořenými algoritmy	91
Obrázek 5.2: Grafické znázornění výsledných procentuálních hodnot spolehlivosti vytvořeného programu pro detekci změn v obraze při aplikaci na vybrané testovací videosoubory	97

Seznam tabulek

Tabulka 4.1: Porovnání časové náročnosti filtrace v prostorové a frekvenční doméně.....	63
Tabulka 5.1: Časová náročnost na zpracování videosouboru o délce 60 s za využití Sobelova filtru	87
Tabulka 5.2: Časová náročnost na zpracování videosouboru o délce 60 s za využití Kirschova filtru	88
Tabulka 5.3: Časová náročnost na zpracování videosouboru o délce 60 s za využití LoG filtru	88
Tabulka 5.4: Časová náročnost na zpracování videosouboru o délce 60 s za využití Cannyho filtru	89
Tabulka 5.5: Spolehlivost detekce změn v obrazu za využití Sobelova operátoru – video č. 1	92
Tabulka 5.6: Spolehlivost detekce změn v obrazu za využití Kirschova operátoru – video č. 1	92
Tabulka 5.7 Spolehlivost detekce změn v obrazu za využití LoG operátoru – video č. 1	93
Tabulka 5.8: Spolehlivost detekce změn v obrazu za využití Cannyho operátoru – video č. 1	93
Tabulka 5.9: Spolehlivost detekce změn v obrazu za využití Sobelova operátoru – video č. 2	95
Tabulka 5.10: Spolehlivost detekce změn v obrazu za využití Kirschova operátoru – video č. 2	95
Tabulka 5.11: Spolehlivost detekce změn v obrazu za využití LoG operátoru – video č. 2	96
Tabulka 5.12: Spolehlivost detekce změn v obrazu za využití Cannyho operátoru – video č. 2	96
