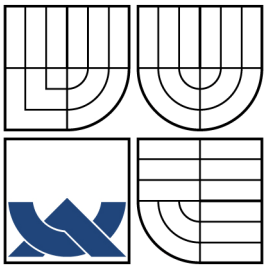
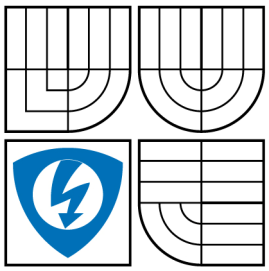


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘÍCÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

ZÍSKÁVÁNÍ DAT Z KAMER

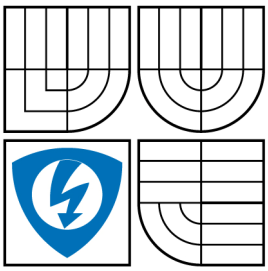
CONTROLLING OF CAMERA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. LADISLAV TYLŠ
AUTHOR

VEDOUCÍ PRÁCE — Ing. MILOSLAV RICHTER, Ph.D.
SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Ladislav Tylš	***ID:***	88525
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Získávání dat z kamer

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte prostředí vhodné pro práci s FireWire kamerami DMK. Vytvořte moduly pro ovládání a nastavení kamer a získávání dat. Soustřeďte se na možnost snímání dat z více kamer a stanovení kvality takto pořízených dat. Vytvořené moduly prezentujte ve vzorové aplikaci (MATLAB, C. Aplikace snímání dat z více kamer s optimalizací parametru snímání).

DOPORUCENÁ LITERATURA:

Kraus K.: Photogrammetrie 1 und 2, Ummler / Bonn, 1996
Žára J., Beneš B., Sochor J., Felkel P.: Moderní počítačová grafika, Computer Press, 1998, ISBN 80-251-0454-0
Hlaváč V., Šonka M.: Počítačové vidění,Grada, Praha 1992, ISBN 80-85424-67-3
Faugeras O.: Three-Dimensional Computer Vision, The MIT Press 1993

Termín zadání:	9.2.2009	***Termín odevzdání:***	25.5.2009

Vedoucí práce: Ing. Miloslav Richter, Ph.D.

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘÍCÍ TECHNIKY

Abstrakt:

Tato práce se zabývá možnostmi vytvoření vhodného prostředí k ovládání a nastavení kamer. V první části jsou popsány základní možnosti připojení kamer a vysvětleny jednotlivé parametry kamer s demonstrací vlivu těchto parametrů na výsledný obraz.

V druhé části je ukázáno, jak je možno vytvořit prostředí, kde bude možno spustit více kamer současně, a které umožní jednoduché nastavení parametrů. K realizaci aplikace je využito toolboxu GUIDE (graphical user interface development environment) programu MATLAB a programu C++BUILDER.

Při vytváření aplikace v programovém prostředí MATLAB bude využito toolboxu image processing a toolboxu image acquisition. Při vytváření aplikace v programovém prostředí C++BUILDER bude využito knihovny opencv, které jsou poskytovány firmou INTEL.

Klíčová slova v českém jazyce:

Parametry kamer, formát obrazu, počet snímků za sekundu, expozice, kontrast, clona, ostrost, zoom, jas, gamma, saturace, barevný tón, vyvážení bílé barvy, kompenzace světla na pozadí, MATLAB GUIDE, image processing toolbox, image acquisition toolbox. C++BUILDER, opencv.

BRNO UNIVERSITY OF TECHNOLOGY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

Abstract:

This thesis describes the principles of making application which is able to set and control camera. The first part describes basic camera connections and it explains definition and specification of camera's features. Features are shown on the acquired image.

The second part of my thesis describes implementation of application, which can use more cameras to image preview, image acquisition and to simply set of camera's features. To implement the applications we can use MATLAB with toolbox GUIDE (graphical user interface development environment) and C++BUILDER. In MATLAB we can use image processing toolbox and image acquisition toolbox. Application which is create in C++BUILDER uses opencv (open source computer vision) libraries developed by INTEL corporation.

English Keywords:

Features of camera, video format, frame per second, exposure, gain, iris, focus, sharpness, zoom, offset, brightness, gamma, saturation, hue, white balance, backlight compensation, tilt, pan, optical filter, shutter, MATLAB GUIDE, image processing toolbox, image acquisition toolbox. C++BUILDER, opencv.

Bibliografická citace VŠKP dle ČSN ISO 690:

TYLŠ, L. *Získávání dat z kamer*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 69 s. Vedoucí diplomové práce Ing. Miloslav Richter, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Získávání dat z kamer jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne: **25. května 2009** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Miloslavu Richterovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **25. května 2009**

……
podpis autora

Obsah

1 ÚVOD 13
2 ZÁKLADNÍ SCHÉMA ZPRACOVÁNÍ OBRAZU 14
2.1 Nižší úroveň zpracování obrazu 14
2.1.1 Osvětlení 14
2.1.2 Objektiv 15
2.1.3 Senzor 15
2.1.4 A/D - digitalizační (grabovací) karta 15
2.1.5 Předzpracování obrazu 15
2.1.6 Segmentace 16
2.2 Vyšší úroveň zpracování obrazu 16
2.2.1 Popis objektů 16
2.2.2 Poslední část měřícího řetězce 16
3 KAMERY PRO POČÍTAČOVÉ VIDĚNÍ 17
3.1 Typy připojení kamer s digitálním výstupem 17
3.1.1 Camera-Link 17
3.1.2 USB (Universal Serial Bus) 18
3.1.3 FireWire 19
3.1.4 Gigabitový ethernet 20
4 KOMUNIKACE S DIGITÁLNÍ KAMEROU 21
4.1 Standard komunikace digitálních kamer pomocí firewire 22
5 PARAMETRY FIREWIRE KAMER DMK 26
5.1 Formát obrazu 26
5.2 Počet snímků za sekundu 27
5.3 Expozice 27
5.3.1 Doba expozice 29
5.4 Kontrast 31
5.5 Clona 32
5.6 Ostrost 33
5.7 Softwarová ostrost 34
5.8 Zoom 35

5.9 Jas 36
5.10 Gamma 37
5.11 Saturace 38
5.12 Barevný tón 38
5.13 Vyvážení bílé barvy 39
5.14 Kompenzace světla na pozadí 40
5.15 Ovládání Spuštění 40
5.16 Ostatní parametry 41
6 POČÍTAČOVÉ VIDĚNÍ V PROSŘEDÍ MATLAB 42
6.1 Image Processing Toolbox 43
6.2 Image Acquisition Toolbox 44
6.3 Realizace v programovém prostředí MATLAB 45
6.3.1 Příklad použitých instrukcí 45
6.3.2 Realizace aplikace v programovém prostředí MATLAB 48
6.3.3 Výsledná aplikace MATLAB 53
7 POČÍTAČOVÉ VIDĚNÍ S OPENCV 55
7.1 Realizace v programovém prostředí C++Builder 56
7.1.1 Implementace knihoven OpenCV do Microsoft Visual C++ 6.0 56
7.1.2 Implementace knihoven OpenCV do C++Builder 57
7.1.3 Příklad použitých instrukcí 58
7.1.4 Realizace aplikace v programovém prostředí C++Builder 60
8 ZÁVĚR 63
9 POUŽITÁ LITERATURA 64
10 SEZNAM PŘÍLOH 66
11 PŘÍLOHY 67

Seznam obrázků

obrázek 2.1: Základní řetězec zpracování obrazu [1] 14
obrázek 3.1: Příklad kamer pro počítačové vidění [6] 17
Obrázek 3.2: Konektor rozhraní Camera-link (26pinů) [2] 18
Obrázek 3.3: Konektor FireWire 800 a FireWire 400 [4] 19
Obrázek 4.4.1: Komunikace aplikace s digitální kamerou 21
Obrázek 4.2: Připojení digitální kamery pomocí protokolu IIDC [13] 24
Obrázek 4.3: Připojení digitálních kamer pomocí firewire [13] 24
Obrázek 5.1: Vliv parametru expozice na snímaný obraz 29
Obrázek 5.2: Příklad obrazu s dlouhou dobou expozice 30
Obrázek 5.3: Příklad obrazu s krátkou dobou expozice [12] 30
Obrázek 5.4: Vliv parametru kontrast na snímaný obraz 31
Obrázek 5.5: Nastavení clony [4] 32
Obrázek 5.6: Správné nastavení zaostřovací vzdálenosti s [5] 33
Obrázek 5.7: Nesprávné nastavení zaostřovací vzdálenosti s [5] 33
Obrázek 5.8: Vliv parametru ostrost 34
Obrázek 5.9: Vliv parametru softwarová ostrost 35
Obrázek 5.10: Vliv parametru zoom 35
Obrázek 5.11: Barevný model HSV 36
Obrázek 5.12: Vliv parametru jasu 36
Obrázek 5.13: Posterizace obrazu 37
Obrázek 5.14: Vliv parametru saturace [11] 38
Obrázek 5.15: Vliv parametru barevný tón [11] 38
Obrázek 5.16 Chromatický digram [1] 39
Obrázek 5.17: Vliv parametru kompenzace světla na pozadí 40
Obrázek 5.18: Spouštění v módu nula 40
Obrázek 5.19: Spouštění v módu jedna 41
Obrázek 5.20: Spouštění v módu dva 41
Obrázek 5.21: Spouštění v módu tři 41
obrázek 6.1: Image processing toolbox [6] 43
obrázek 6.2: Image acquisition toolbox [6] 44

Obrázek 6.3: Vývojový diagram spuštění aplikace 48
Obrázek 6.4: Vývojový diagram běhu aplikace MATLAB 50
Obrázek 6.5: Vývojový diagram callback funkce parametru kamery 52
Obrázek 6.6: Výsledná aplikace v programovém prostředí MATLAB 53
Obrázek 7.1: Výsledná aplikace vytvořená v C++Builder 60
Obrázek 7.2: Okna aplikace se zobrazením videa a zachycených snímků 61
Obrázek 7.3: Zachycené snímky ze dvou současně běžících kamer 61
Obrázek 7.4: Dialogové okno nastavení parametrů kamery 62

Seznam tabulek

Tabulka 5.1: Formáty obrazu 26
Tabulka 5.2: Vzájemný vztah expozice, clony a doby expozice [4] 28

1 ÚVOD

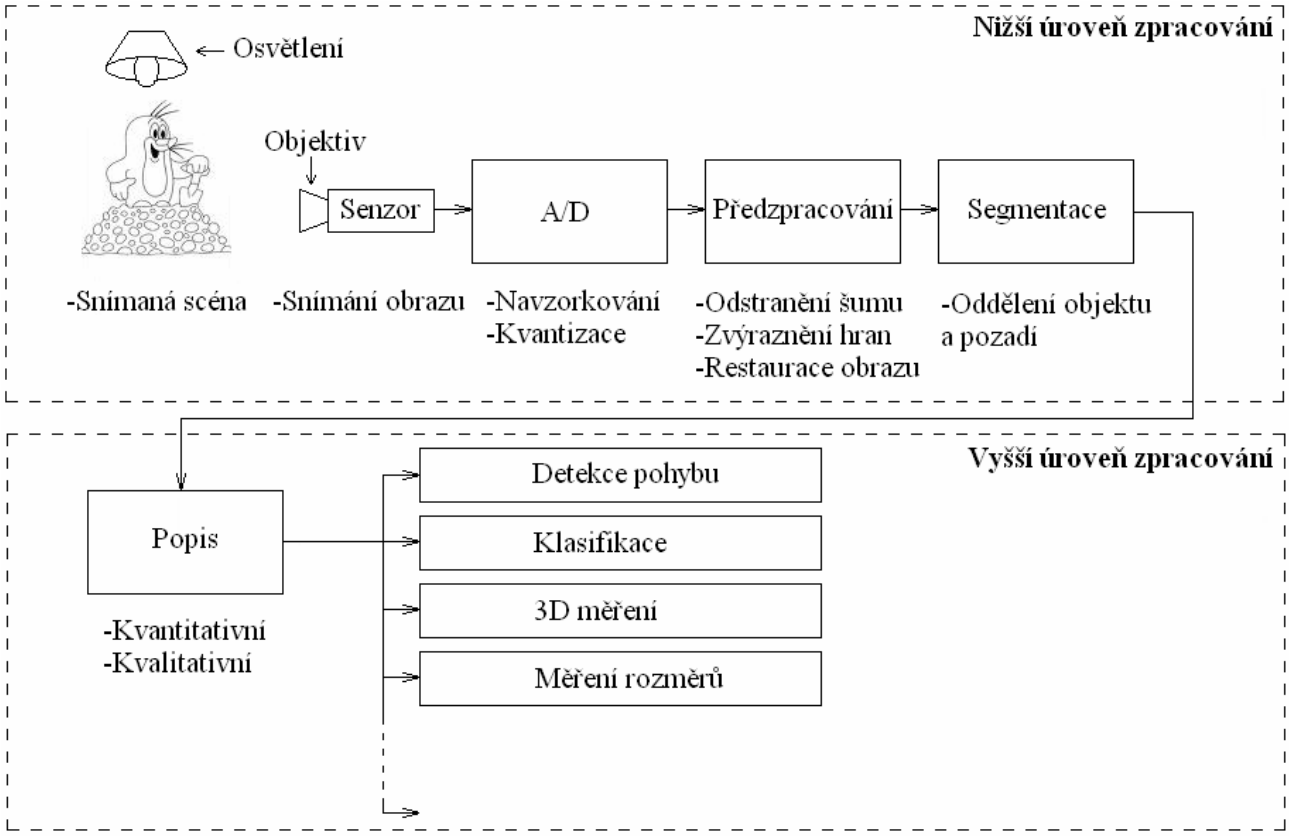
V dnešní době čím dál tím více vzrůstá poptávka po počítačovém vidění a pokud není pořízen kvalitní snímek snímané scény, je problematické další zpracování tohoto obrazu. Proto je zapotřebí optimálně nastavit parametry kamery, aby byl získán co nejkvalitnější snímek. K tomu je zapotřebí mít vhodné prostředí, ve kterém je možno snadno nastavit parametry.

Cílem této práce je seznámit se s parametry FireWire kamer DMK a s principem získávání dat. Bude vytvořeno prostředí, které umožní automatické nastavení parametrů a v tomto prostředí bude zároveň možno spustit více kamer současně. K realizaci aplikace bude využito toolboxu GUIDE (graphical user interface development environment) programu MATLAB a programu C++BUILDER.

Při vytváření aplikace v programovém prostředí MATLAB bude využito toolboxu image processing a toolboxu image acquisition. První z těchto toolboxů poskytuje funkce pro zpracovávání obrazu. Druhý toolbox zajišťuje připojení, inicializaci, nastavení kamer a funkce pro získání obrazu. Výsledná aplikace bude převedena do formátu, který je možno spustit i na počítači, kde není nainstalováno programové prostředí MATLAB.

Při vytváření aplikace v programovém prostředí C++BUILDER bude využito knihovny opencv, které jsou poskytovány firmou INTEL a jedná se o otevřenou knihovnu (pro nekomerční využití). Tato knihovna poskytuje vysokoúrovňové funkce a datové typy, které ulehčují operace spojené se získáváním dat z kamer a dalším zpracováním získaného obrazu.

2 ZÁKLADNÍ SCHÉMA ZPRACOVÁNÍ OBRAZU



obrázek 2.1: Základní řetězec zpracování obrazu [1]

2.1 NIŽŠÍ ÚROVEŇ ZPRACOVÁNÍ OBRAZU

2.1.1 Osvětlení

Osvětlení je jedním z prvních článků řetězce zpracování obrazu. Osvětlení se dá rozdělit podle typu zdroje, provedení, vlnové délky a mnoha dalších parametrů. Typem zdroje osvětlení může být například sluneční světlo, zářivka, žárovka, výbojka, LED dioda, laser. Provedení osvětlení může být bodové, plošné, kruhové, světelný pruh, vzor, přímé, směrové, rovnoběžné, difúzní, zadní a další. Další

rozdělení zdrojů osvětlení je možné podle vlnové délky. Tyto zdroje osvětlení mohou být infračervené, ultrafialové, polarizované, koherentní.

2.1.2 Objektiv

Objektivy se vyrábějí v různých provedeních a dají se rozdělit podle zorného úhlu, zvětšení, rozsahu střeni, hloubky ostrosti, objektivy s polarizačními filtry, bez polarizačních filtrů. Dalšími parametry objektivu jsou světelné číslo (světlo, které propustí na senzor), ohnisková vzdálenost, průměr clony a v různé světelné řady.

2.1.3 Senzor

Nejčastějším rozdělením senzoru je podle použité technologie (CCD, CMOS, progresivní, prokládaný), nebo podle rozměru čipu (1/3“, 1/2“, 2/3“). Další rozdělení je možné podle typu na řádkové, plošné, barevné, černobílé. Další parametry jsou například rozlišení, velikost pixelu, video standart, spektrální citlivost, datový tok, frekvence hodin, spektrální citlivost, expoziční doba, závěrka, interface.

2.1.4 A/D - digitalizační (grabovací) karta

Digitalizační (Grabovací) kartu volíme podle použité kamery. Tato karta slouží k vzorkování, kvantování signálu z čipu a je obvykle implementována v programovatelných hradlových polích, nebo v signálových procesorech. Někdy tyto karty vykonávají i některé operace předzpracování obrazu. Nejčastěji se používá 8 bitová digitalizační karta (256 úrovní), což je dáno citlivostí lidského oka, které je schopno rozlišit maximálně 128 jasových úrovní (7 bitů). Dvounásobek jasových úrovní se volí proto, aby velikost jedné jasové informace byla obsažena v jednom bytu. Výstupní signál digitalizačních karet je pak obvykle ve formátu MPEG.

2.1.5 Předzpracování obrazu

Předzpracování obrazu může obsahovat operace potlačení šumu, operace odstranění zkreslení, nebo potlačení či zvýraznění hran obrazu. Výstup i vstup této operace je obraz a využívá se zde nadbytečnosti údajů v obraze. Nejčastějšími

zástupci předzpracování jsou bodové jasové transformace, geometrické transformace (plošné, jasové), lokální operace předzpracování (filtrace, detekce hran, ostření), restaurace obrazu a matematická morfologie.

2.1.6 Segmentace

Segmentace slouží k rozčlenění obrazu do částí, které souvisejí s detekovanými předměty nebo oblastmi reálného světa. Jedná se tedy o oddělení objektů, které jsou důležité pro danou úlohu, od pozadí obrazu. Problém ovšem při segmentaci tvoří fakt, že pozorovaná data nemusí být jednoznačná. Nejjednodušším příkladem segmentace je segmentace prahováním. Další možné způsoby segmentace jsou například segmentace na základě hran, segmentace narůstáním oblastí, segmentace srovnáním se vzorem, sub-pixelové určení hranice a další.

2.2 VYŠŠÍ ÚROVEŇ ZPRACOVÁNÍ OBRAZU

2.2.1 Popis objektů

Popis objektů tvoří první část ve vyšší úrovni zpracování obrazu a jedná se o operaci, kdy se ze získaných segmentovaných dat snažíme získat příznaky jednotlivých objektů. Mezi největší problémy popisu objektů patří volba příznaků, které nejsou u počítačového vidění jednoznačné pro všechny úlohy.

2.2.2 Poslední část měřícího řetězce

Poslední části měřícího řetězce náleží operace, které již souvisejí přímo s danou úlohou, která je pomocí počítačového vidění zpracovávána. Úlohy počítačového vidění mohou být například měření rozměrů, detekce předmětů, vad, OCR, klasifikace, detekce pohybu a mnoho dalších. Problémem počítačového vidění je obzvláště variantnost úloh, které jsou zpracovávány a tudíž nutnost řešit vícekrát podobné úlohy. Tento řetězec zpracování byl převzat z literatury [1].

3 KAMERY PRO POČÍTAČOVÉ VIDĚNÍ

obrázek 3.1: Příklad kamer pro počítačové vidění [6]

V průmyslových kamerách pro počítačové vidění jsou ve většině případů obsaženy první tři bloky měřícího řetězce. Jedná se tedy o senzor, A/D převodník a předzpracování obrazu. Výstup z průmyslových kamer je tedy obvykle v digitální formě a to ve většině případů ve formátu MPEG. Na trhu se vyskytuje velké množství typu kamer. Mezi nejvýznamnější zástupce můžeme zařadit například kamery monochromatické, barevné, Bayerovy (s aditivním barevným modelem) kamery, kamery s objektivem s proměnlivou ohniskovou vzdáleností (zoom), kamery s připojením přes Camera-Link, USB, FireWire, Gigabitový ethernet a jiné. Dalšími parametry, které jsou u kamer důležité, mohou být například počet snímků za minutu (Fps), rozlišovací schopnost, velikost, technologie použitého čipu a samozřejmě cena.

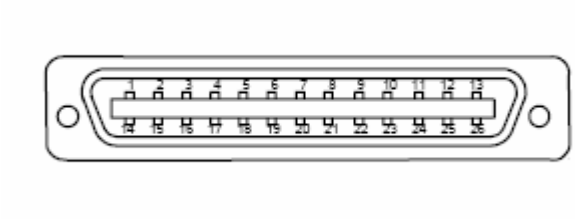
3.1 TYPY PŘIPOJENÍ KAMER S DIGITÁLNÍM VÝSTUPEM

3.1.1 Camera-Link

Camera-Link je sériově paralelní komunikační rozhraní vyvinuté pro aplikace počítačového vidění, jehož licenci vlastní firma Automated Imaging Association (AIA). Rozhraní využívá fyzickou vrstvu, která je založena na LVDS (low voltage

differential signaling). Standard využívá 28 bitů pro reprezentaci dat z jednoho pixelu. Tyto bity obsahují 24 bitů nesoucí informaci o pixelu, 3 bity využity na řídící signály (FVAL, LVAL, DVAL) a poslední bit zatím není ve standardu využíván (SPARE). Těchto 28 bitů je postupně předkládáno na čtyři piny konektoru MDR

Rozhraní Camera-Link využívá 26-ti pinový konektor MDR, který je zobrazen na **Obrázek 3.2,** kde pět párů je využito na přenos dat a časování, čtyři páry na ovládání kamery, dva páry na sériovou komunikaci a dva páry jsou využity na zemnění. Maximální teoretická možná rychlost přenosu je 2,38 Gb/s (297,5 MB/s) při využití jednoho kanálu. Při využití tří kanálu je maximální teoretická rychlost přenosu až 850 MB/s. V praxi je toto rozhraní proto využíváno hlavně pro nejnáročnější aplikace. Maximální délka připojení je 10m. Avšak nevýhodou je vysoká cena.



Obrázek 3.2: Konektor rozhraní Camera-link (26pinů) [2]

3.1.2 USB (Universal Serial Bus)

Univerzální sériové rozhraní USB se v poslední době stalo jedním z nejvíce používaných připojení periférií k počítači. Jedná se o univerzální rozhraní, které dovoluje připojit až 127 zařízení. Existují dvě základní verze USB.

- USB 1.1 – maximální přenosová rychlost 12Mb/s (1,5 MB/s)
- USB 2.0 - maximální přenosová rychlost 480Mb/s (60 MB/s)
- USB 3.0 - maximální přenosová rychlost 5Gb/s (625 MB/s)

USB 2.0 je zpětně kompatibilní s USB 1.1. Maximální délka mezi dvěma zařízeními bez opakovače je pět metrů. Rozhraní obsahuje i stejnosměrné 5V napětí, pomocí kterého můžeme napájet zařízení s odběrem proudu 100mA a v případě potřeby až 500mA (pouze jedno zařízení na sběrnici). Programové vybavení pro ovládání

rozhraní USB odpovídá plně standardu plug & play, což umožňuje připojení zařízení bez nutnosti restartování systému. V USB rozhraní existují čtyři typy konektorů.

- Micro USB
- Mini USB
- USB typ B
- USB typ A

3.1.3 FireWire

FireWire (název od společnosti Apple) je sériové rozhraní pro připojení periférii k počítači, které se někdy označuje jako IEEE 1394, i.LINK(sony), DV (Panasonic). Toto rozhraní bylo vyvinuto speciálně pro přenos obrazových a zvukových dat. Nyní je hojně využíváno zejména pro připojení digitálních kamer. Pomocí tohoto rozhraní lze připojit až 63 zařízení. Maximální délka kabelu je 4,5 metrů. Programové vybavení pro ovládání rozhraní FireWire odpovídá standardu plug & play, což umožňuje připojení zařízení bez nutnosti restartování systému. Existují dvě základní verze FireWire.

Obrázek 3.3: Konektor FireWire 800 a FireWire 400 [4]

FireWire 400 (IEEE 1394a) je starší variantou tohoto rozhraní s rychlostí přenosu dat 400 Mbit/s. Konektor je ukázán na **Obrázek 3.3** vpravo. Pro nedostatečnou přenosovou rychlost byl tento konektor rozšířen o dva vodiče a tato varianta dostala název FireWire 800 (IEEE 1394b). Rychlost přenosu dat je 800 Mbit/s. Výhodou tohoto připojení je jednoduchost a nízká cena, avšak není tak rozšířené jako rozhraní USB.

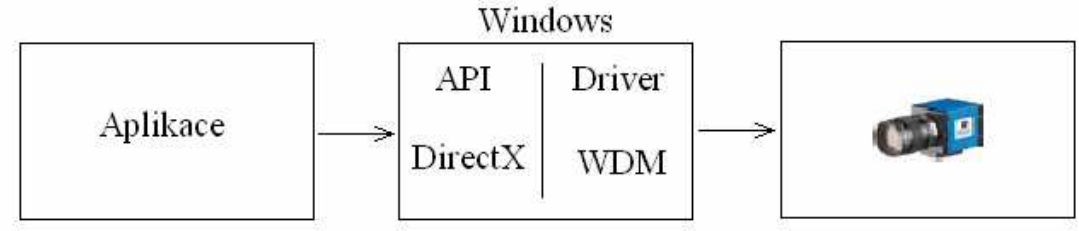
3.1.4 Gigabitový ethernet

Protokol Ethernetu byl vyvinut společností Xerox v 70. letech minulého století. Ethernet je typ lokální sítě, která je převážně realizována na síťové a fyzické vrstvě. Původní protokol byl vyvinut pro potřeby kancelářských aplikací. Postupem času se vyvinulo několik verzí Ethernetu.

- Ethernet – (IEEE 802.3) definován pro koaxiální kabel, kroucenou dvoulinku a optické vlákno s přenosovou rychlostí 10Mb/s.
- Fast Ethernet – Základní verze ethernetu (IEEE 802.3u) s přenosovou rychlostí 100 Mb/s Definována pro kroucenou dvoulinku a optická vlákna.
- Gigabitový Ethernet – Přenosová rychlost 1 Gb/s. Původně byl definován pouze pro optická vlákna (IEEE 802.3z v roce 1998), později byla doplněna i varianta pro kroucenou dvoulinku (IEEE 802.3ab).
- Desetigigabitový Ethernet – Přijat jako IEEE 802.3ae v roce 2003. Definován pro optická vlákna a později i pro kroucenou dvoulinku s přenosovou rychlostí 10 Gb/s.
- Stogigabitový Ethernet – Přijat jako standart IEEE 802.3ba v roce 2007. Standart obsahuje dvě základní přenosové rychlosti 40 Gb/s a 100 Gb/s.

Gigabitový Ethernet postupně vytlačuje klasický Fast Ethernet a dnes je standardem, který předčí mnohá digitální rozhraní. Díky hromadnému nasazení v oblasti lokálních sítí jej výrobci zařadili i do portfolia různých zařízení – kamery, tiskárny, síťová úložiště. Metalickým kabelem 1Gbit třídy cat6e je možno připojovat zařízení až do vzdálenosti 100 metrů. Výhodou je díky masivnímu nasazení nízká cena všech komponent, což obecně platí pro všechny varianty Ethernetu.

4 KOMUNIKACE S DIGITÁLNÍ KAMEROU



Obrázek 4.4.1: Komunikace aplikace s digitální kamerou

Pod operačním systémem Windows probíhá komunikace aplikace s multimediálními zařízeními podle **Obrázek 4.4.1**. Multimediální zařízení se detekuje pomocí API funkcí (Application Programming Interface), které tvoří rozhraní pro komunikaci mezi hardwarem a softwarem. Soubor API funkci pro práci s multimediálními zařízeními je znám pod názvem DirectX, která byla vytvořena firmou Microsoft. Součástí DirectX jsou tyto komponenty:

- **DirectX Graphics**
  - **DirectDraw** – API funkce pro práci s 2D grafikou
  - **Direct3D -** funkce pro práci s 3D grafikou
  - **DXGI** – funkce pro výčet adaptérů a monitorů
- **DirectInput** – funkce pro podporu vstupních zařízení (myš, klávesnice, atd.)
- **DirectPlay** – funkce pro podporu hry více hráčů po síti
- **DirectSound** - funkce pro přehrávání a zaznamenávání zvuku.
  - **DirectSound3D** – funkce pro práci s 3D zvukem
- **DirectMusic** – funkce pro podporu přehrávání a zpracování hudby
- **DirectShow** – funkce pro podporu multimediálních aplikací, přehrávání a zpracování videa a zvuku
- **DirectX Media Objects** – funkce pro podporu streamovacích zařízení
- **DirectSetup** – funkce pro instalaci knihovny DirectX

Nejnovější verzí DirectX je verze DirectX 11, která je funkční pod operačním systémem Windows Vista a Windows 7. Pro operační systém Windows XP je možno využít DirectX 9.0c.

Pro práci s DirectX je možno stáhnout DirectX SDK (Software development kit). DirectX SDK je vývojářská sada, která obsahuje všechny API funkce DirectX a umožňuje pracovat s těmito funkcemi v programovacích jazycích Visual Basic.NET, C/C++ a C#. Jedná se o runtime knihovny v binární formě, které obsahují:

- DirectX hlavičkové soubory a knihovny
- DirectX systémové komponenty
- DirectX API funkce
- Dokumentace
- Ukázky aplikací a zdrojové kódy

Součástí DirectX je tedy i DirectShow a jak již bylo zmíněno dříve, tato komponenta obsahuje funkce pro práci s přenosem digitálního obrazu a zvuku (streamovací zařízení). Pomocí DirectShow lze tedy rozpoznat, jestli není připojeno nějaké multimediální zařízení. Aby bylo možno detekovat mnoho typů zařízení, bylo nutné standardizovat strukturu jejich ovladačů. Pro Windows XP se tento standard nazývá WDM (Win32 Driver Model) a pro Windows Vista WDDM (Windows Vista Display Driver Model) standart.

4.1 STANDARD KOMUNIKACE DIGITÁLNÍCH KAMER POMOCÍ FIREWIRE

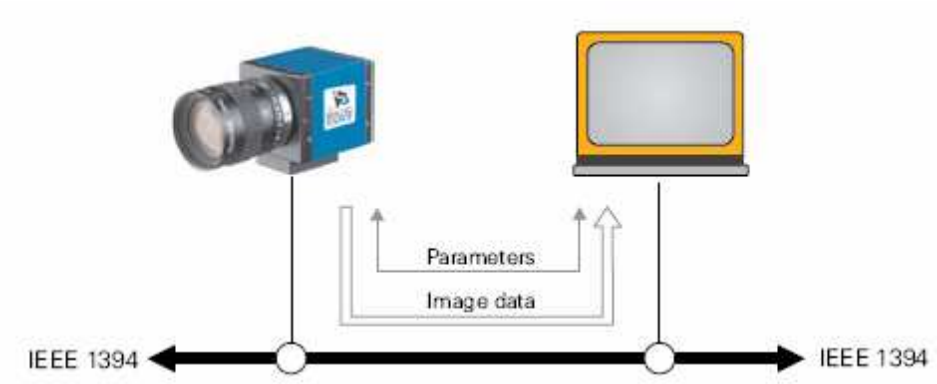
Pro komunikaci počítače s digitální kamerou přes firewire připojení existují různé typy komunikačních protokolů, které definují přenos zvuku, videa a řídících signálů. Mezi základní typy těchto komunikačních protokolů pro firewire patří:

- **SBP-2**
- **AV/C**
- **DCAM/IIDC**

SBP-2 protokol (Seriál Bus Protocol 2) – je definován pomocí normy IEEE 1394 a definuje pouze přenos souboru. Proto se tento protokol využívá pro záznamová zařízení. Původní myšlenka byla přizpůsobit SCSI (Small Computer Systém Interface) pro 1394 (firewire) sériové rozhraní. Později však byl tento protokol zaměřen na výměnu příkazů, dat a statusů mezi zařízeními, které jsou propojeny přes sériovou sběrnici.

AV/C protokol (Audio Video Control) - tento protokol je definován pomocí normy IEEE 1394 (Institute of Electrical and Electronic Engineers) a definuje chování digitálních zařízení, jako jsou například kamery nebo dekodéry. Tento protokol je široce rozšířený na celém světě, protože definuje přenos obrazových dat, zvukových dat, řídících signálů a to i v komprimovaném formátu.

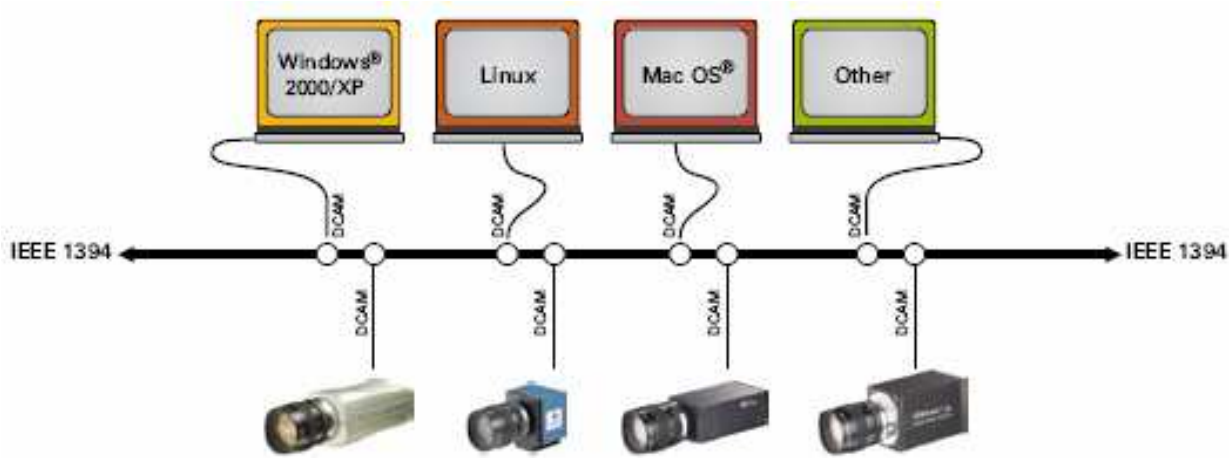
DCAM/IIDC (Digital Camera Specification/Instrumentation and Industrial Control Digital Camera) - tento protokol je opět definován pomocí normy IEEE 1394. DCAM/ IIDC protokol byl iniciován pracovní skupinou IIDC z 1394 Trade Association a je neustále vyvíjen. IIDC (Instrumentation & Industrial Digital Camera) je tedy standard, který definuje přenos nekomprimovaného digitálního obrazu přes asynchronní sériovou sběrnici firewire a také definuje parametry kamer (jas, kontrast,…atd.). Tento standard ovšem nedefinuje přenos audio signálu. Tento protokol se často využívá u průmyslových aplikací, kde jsou požadovány vysoké přenosové rychlosti a nekomprimovaný formát, kvůli realtimovému zpracování a vyhodnocení obrazu. Ukázka připojení je uvedena na **Obrázek 4.2**.



Obrázek 4.2: Připojení digitální kamery pomocí protokolu IIDC [13]

Pro operační systém Windows nabízí společnost Imaging source ovladače, které podporují základní parametry kamer. Pokud je dodržen standard IIDC při vytvoření ovladače, pak je možno digitální kamery ovládat a pracovat s nimi pod jakýmkoliv operačním systémem, jak je patrné z **Obrázek 4.3.** Pod operační systém Linux jsou k dispozici například otevřené knihovny:

- sourceforge.net/projects/unicap
- sourceforge.net/projects/libdc1394
- sourceforge.net/projects/coriander



Obrázek 4.3: Připojení digitálních kamer pomocí firewire [13]

GenICam™ standard - V dnešní době existuje mnoho typů komunikačních protokolů a způsobů připojení, jak již bylo ukázáno v předešlém textu. Proto se výrobci kamer dohodli na standardu GenICam™. Autor tohoto standardu je Europen Machine Vision Association. (EMVA).

Tento standard má za cíl vytvořit obecné programovací rozhraní, pro všechny druhy připojení jako jsou například gigabitový ethernet, camera link, 1394 DCAM, USB a firewire a pro všechny parametry kamer, které jsou v těchto kamerách implementovány. Aplikační programové rozhraní by mělo být pro všechny druhy připojení a parametrů stejné. Standard GenICam™ se tedy dělí na tři základní části:

- **GenApi** – nastavování kamery
- **Standart Feature Naming Convention (SFNC)** – standard zabývající se jednotným pojmenováním parametrů kamer
- **GneTL** – obecný standard transportní vrstvy a grabování obrazu

Mezi členy podílející se na vývoji standardu a které tento standard přijali jsou například firmy:ABS GmbH, Allied Vision Technologies GmbH, COLOUR Control Farbmesstechnik Gmbh, e2v semiconductors (former ATMEL), Fast Vision LLC, GigaLinx Ltd, Matrox Ltd, National Instruments Corp, PixeLink R&D, Silicon Software, University of Bristol, Vieworks Co., Ltd. Bližší informace o tomto standardu je možné získat na stránkách [14].

5 PARAMETRY FIREWIRE KAMER DMK

Kvalita získávaného obrazu je ovlivnitelná mnoha parametry, jako jsou například osvětlení, objektiv, senzor a parametry kamery. Parametry kamery tedy patří do řetězce nižší úrovně zpracování obrazu a lze pomocí jejich nastavení získat optimální obraz, nebo zvýraznit objekty, které jsou pro další zpracování podstatné. V následujícím textu budou tyto parametry popsány a bude ukázán jejich vliv na výsledný obraz.

5.1 FORMÁT OBRAZU

Formát obrazu je první parametr, který je nutno zvolit. Tento parametr určuje, v jakém rozlišení bude snímán obraz sledované scény, v jakém barevném prostoru (RGB, YUV, MONO) a kolik bitů bude využito na reprezentaci jednoho pixelu. V **Tabulka 5.1** je souhrn všech formátů obrazu, které jsou definovány ve standardu DCAM/IIDC [13].

Tabulka 5.1: Formáty obrazu

RGB	YUV	Y
640 X 480 (24bit/pixel)	160 X 120 (24bit/pixel) (4:4:4)	640 X 480 (8bit/pixel)
800 X 600 (24bit/pixel)	320 X 240 (16bit/pixel) (4:2:2)	640 X 480 (16bit/pixel)
1024 X 768 (24bit/pixel)	640 X 480 (12bit/pixel) (4:1:1)	800 X 600 (8bit/pixel)
1280 X 960 (24bit/pixel)	640 X 480 (16bit/pixel) (4:2:2)	800 X 600 (8bit/pixel)
1600 X 1200 (24bit/pixel)	800 X 600 (16bit/pixel) (4:2:2)	1024 X 768 (16bit/pixel)
	1024 X 768 (16bit/pixel) (4:2:2)	1024 X 768 (16bit/pixel)
	1280 X 960 (16bit/pixel) (4:2:2)	1280 X 960 (8bit/pixel)
	1600 X 1200 (16bit/pixel) (4:2:2)	1280 X 960 (8bit/pixel)
		1600 X 1200 (16bit/pixel)
		1600 X 1200 (16bit/pixel)

5.2 POČET SNÍMKŮ ZA SEKUNDU

Počet snímků za sekundu (FPS) udává, kolik snímků sledované scény bude zachyceno do jedné sekundy. Tento parametr závisí na nastaveném formátu obrazu a je obvykle udáván v následujících řadových hodnotách dle [13].

$$FPS = \ldots 1.875, 3.75, 7.5, 15, 30 \ldots \tag{5.1}$$

5.3 EXPOZICE

Expozice (exposure) je závislá na třech hlavních faktorech: doba expozice, clona a kontrast (gain), který se pro filmy vyjadřuje jako ISO citlivost. Pomocí těchto tří parametrů lze nastavovat expozici a je jedno, jaké kombinace těchto tří parametrů se použije pro získání stejné expozice. Výběr kombinace ovšem závisí na rychlosti dané scény. Pro rychlé scény je potřeba volit krátkou dobu expozice, což je ovšem náročnější na zesílení a na optiku systému. Při kratších dobách expozice se tedy více projevuje šum. Vzájemný vztah expozice, doby expozice a clony lze vyjádřit vztahem:

$$EV = \log_2 \frac{F^2}{t} \tag{5.2}$$

kde t je doba expozice [s]

F je clonové číslo a platí pro něj vztah $F = \frac{f}{D}$ (5.3)

kde f je ohnisková vzdálenost [mm] a D je průměr otvoru clony [mm].

Tento vztah souvisí s chováním lidského oka, kdy vnímaný jas (expozice) lidským okem je logaritmicky závislý na intenzitě dopadajícího světla. Přírůstek intenzity dopadajícího světla tedy vnímáme jako velkou změnu při nízké úrovni osvětlení a jako malý přírůstek při vysoké úrovni osvětlení. Tato závislost se v praxi často vyjadřuje jako logaritmus o základu dvou, jak ukazuje vztah (5.2). Oko ovlivňuje intenzitu dopadajícího světla pomocí rozšíření, nebo zúžení zorničky (iris). Snížení expozice o jednu *Ev* hodnotu tedy znamená snížení expozice o jednu

polovinu. Závislost je patrná v **Tabulka 5.2**, kde je zobrazena závislost expoziční doby na expozici a na použité cloně. Parametr expozice u kamer ovšem často neukazuje expozici snímaného obrazu, ale dobu expozice, která s ním souvisí.

.

Tabulka 5.2: Vzájemný vztah expozice, clony a doby expozice [4]

	Clonové číslo						
Ev	**1,0**	**1,4**	**2,0**	**2,8**	**4,0**	**5,6**	**8,0**
-6	60	2m	4m	8m	15m	30m	60m
-5	30	60	2m	4m	8m	15m	30m
-4	15	30	60	2m	4m	8m	15m
-3	8	15	30	60	2m	4m	8m
-2	4	8	15	30	60	2m	4m
-1	2	4	8	15	30	60	2m
0	1	2	4	8	15	30	60
1	1/2	1	2	4	8	15	30
2	1/4	1/2	1	2	4	8	15
3	1/8	1/4	1/2	1	2	4	8
4	1/15	1/8	1/4	1/2	1	2	4
5	1/30	1/15	1/8	1/4	1/2	1	2
6	1/60	1/30	1/15	1/8	1/4	1/2	1
7	1/125	1/60	1/30	1/15	1/8	1/4	1/2
8	1/250	1/125	1/60	1/30	1/15	1/8	1/4
9	1/500	1/250	1/125	1/60	1/30	1/15	1/8
10	1/1000	1/500	1/250	1/125	1/60	1/30	1/15

5.3.1 Doba expozice

Doba expozice (exposure time, shutter) je čas, po kterém je ze senzoru vyčítán elektrický náboj způsobený dopadajícím světelným zářením ze snímané scény. Tato doba se často přepočítává na zaokrouhlené hodnoty mocniny dvou. Někdy se používají ještě mezihodnoty, aby bylo možno nastavit více kombinací.

$$Te \approx 2^n \approx \frac{1}{1000}, \frac{1}{500}, \frac{1}{250}, \frac{1}{125}, \frac{1}{60}, \frac{1}{30}, \frac{1}{15}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8, 15, 30 ... [s] \qquad (5.4)$$

kde $n \in Z$ (parametr expozice)

Tuto dobu lze nastavit na automatické přizpůsobování nebo na manuální, kdy lze dobu expozice nastavit ručně například pomocí slideru. Zvětšení hodnoty parametru expozice (*Te*) tedy znamená nárůst doby expozice na dvojnásobek původní hodnoty. Závislost nastavení doby expozice na získávaném obraze je patrná z **Obrázek 5.1**.

Při získání obrazu vlevo bylo použito módu automatického přizpůsobování expoziční doby, kdy se doba expozice nastavuje na základě odchylky od střední hodnoty šedé získávaného obrazu (při osmibitové reprezentaci barvy hodnota 128). Při získání obrazu uprostřed byla doba expozice nastavena na jednu tisícinu sekundy (obraz podexponován). Na tomto obrazu je patrný šum, který je způsoben optikou a zesilovačem. Na obrazu vpravo je patrná dlouhá doba expozice, která způsobila přeexponování snímku. Doba expozice je tedy volena na základě celkové expozice snímku a na základě dynamiky snímaného děje.

Automatická doba expozice *Te=1/1000s* *Te= 1/2s*

Obrázek 5.1: Vliv parametru expozice na snímaný obraz

Obrázek 5.2: Příklad obrazu s dlouhou dobou expozice

Obrázek 5.3: Příklad obrazu s krátkou dobou expozice [12]

5.4 KONTRAST

Parametr kontrast (gain) udává, jak je zesílen výsledný signál ze senzoru. Čím vyšší bude toto zesílení (čím vyšší bude ISO citlivost pro fotoaparáty), tím se elektronika spokojí se slabším signálem ze senzoru. V praxi nemá smysl normovat vlastní zesílení pro kamery, protože citlivost senzorů na světlo je různá. Normalizace se provádí u fotoaparátů, kde se normuje celková citlivost senzoru se zesilovačem. Citlivost se standardně udává v ISO jednotkách a měla by odpovídat citlivosti klasického filmu. Každá sousední hodnota na ISO stupnici mění citlivost vždy právě 2x. Typická základní stupnice ISO tedy je:

$$ISO = \ldots 50, 100, 200, 400, 800, 1600, \ldots \qquad (5.5)$$

Pokud zvýšíme ISO citlivost 2x (např. z ISO 100 na ISO 200), ke stejné expozici stačí poloviční množství světla (poloviční množství fotonů). Při velké hodnotě kontrastu se zvyšuje v obraze podíl šumu, který je způsoben zesilovačem výstupního signálu. Příklad vlivu parametru kontrast je ukázán na **Obrázek 5.4**.

Auto kontrast | Snížený kontrast | Zvýšený kontrast

Obrázek 5.4: Vliv parametru kontrast na snímaný obraz

5.5 CLONA

Clona (iris) je poslední parametr, kterým lze přímo ovlivnit expozici výsledného snímku. Pomocí tohoto parametru je možné omezit světlo dopadající na citlivou plochu senzoru. Clona je v podstatě mechanická náhražka lidské zornice, která omezuje množství dopadajícího světla na sítnici. Množství dopadajícího světla je tedy úměrné ploše, kterou prochází světelné záření. Pro zvýšení, či snížení se tedy používá clony, jak je ukázáno na **Obrázek 5.5**. Při zdvojnásobení průměru clony ovšem nedojde ke zdvojnásobení expozice, ale ke čtyřnásobení. Je to dáno plochou, kterou prochází světelné záření ($\pi * r^2$). Proto je tedy nutno průměr zvýšit 1,4krát ($\sqrt{2}$), ke zdvojnásobení expozice.

Vliv na snížení množství světla dopadajícího na senzor má ovšem také ohnisková vzdálenost. Při zdvojnásobení této vzdálenosti klesne množství dopadajícího světla čtyřikrát. Je to dáno tím, že se světlo rozprostře na větší plochu. Pro zjednodušení se tedy zavedla tzv. clonová čísla, která vypočítáme z dříve uvedeného vztahu (5.3). Clonové číslo se tady spočítá jako podíl ohniskové vzdálenosti a průměru clony. Clony se vyrábí s clonovými čísly v řadách (5.6), aby se expozice při změně průměru o jednu hodnotu zdvojnásobila. Závislost na expozici je uvedena v **Tabulka 5.2**. Malé clonové číslo tedy značí velké apertury, které se používají především ve hvězdářství a velké clonové číslo se používá u malých apertur.

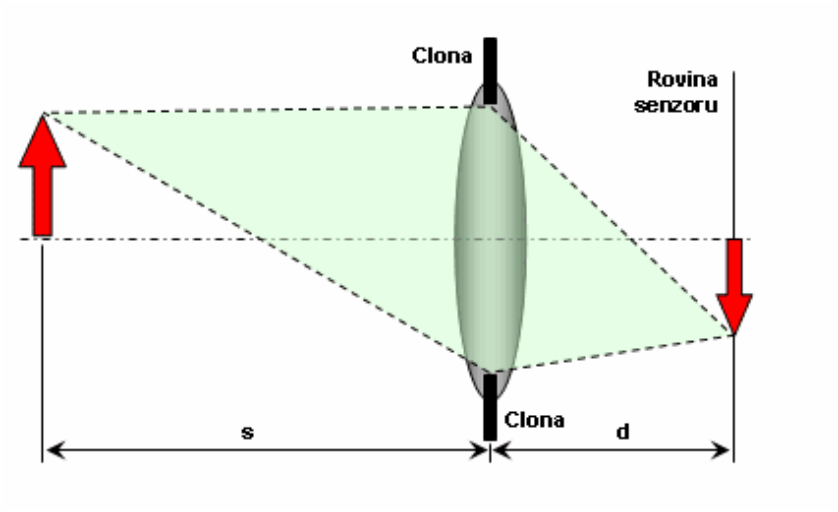
$$F = \sqrt{2^n} = 1.0, 1.4, 2.0, 2.8, 4.0, 5.6, 8, 11, 16, 22, .. \text{ kde } n \in N \qquad (2.6)$$

Obrázek 5.5: Nastavení clony [4]

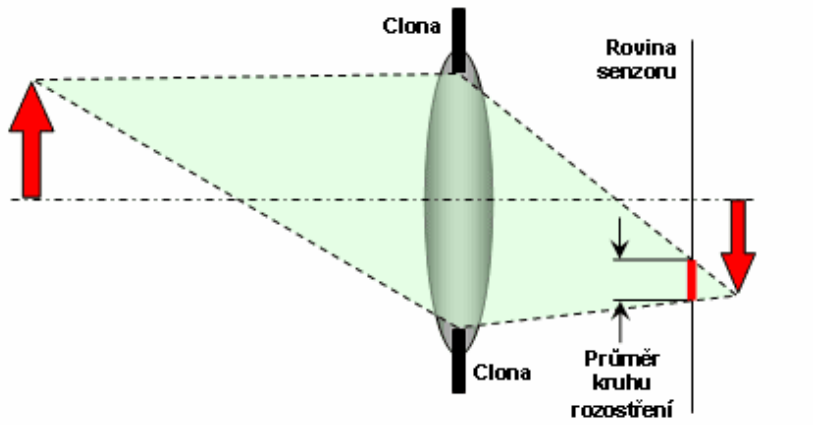
5.6 OSTROST

Vedle expozice je dalším hlavním faktorem ostrost (focus) výsledného obrazu. Ostrost lze ovlivnit mnoha faktory, jako jsou například expoziční čas, objektiv, hloubka ostrosti, velikosti pixelu, cloně, softwarové ostření….atd.

Při zaostřování se pohybují čočky v objektivu tak, aby rovina zaostření byla nastavena na zaostřovací vzdálenost, ve které se nachází sledovaný objekt. Při správném nastavení roviny zaostření je snímaný světelný bod zobrazen opět jako bod na rovině senzoru. To je patrné z **Obrázek 5.6**.



Obrázek 5.6: Správné nastavení zaostřovací vzdálenosti s [5]



Obrázek 5.7: Nesprávné nastavení zaostřovací vzdálenosti s [5]

Při nesprávném určení zaostřovací roviny ovšem dojde k tomu, že se snímaný světelný bod nezobrazí na senzoru jako bod, ale jako kruh rozostření. Tento případ zobrazen na **Obrázek 5.7**.

V praxi se vyskytují dvě základní možnosti, jakými lze dosáhnout automatické optimální ostrosti. Jednou z těchto možností je aktivní autofocus, který vysílá signál (odtud aktivní) a pomocí tohoto signálu se snaží zjistit zaostřovací vzdálenost. Druhou možností je pasivní autofocus, kdy pomocí rozboru získaného obrazu (hrany, fázový posuv signálu) nastavuje optimální zaostřovací vzdálenost.

Správně a nesprávně nastavený parametr ostrosti na získaném obrazu je zobrazen na **Obrázek 5.8**.

Nesprávně nastavená ostrost Správně nastavená ostrost

Obrázek 5.8: Vliv parametru ostrost

5.7 SOFTWAROVÁ OSTROST

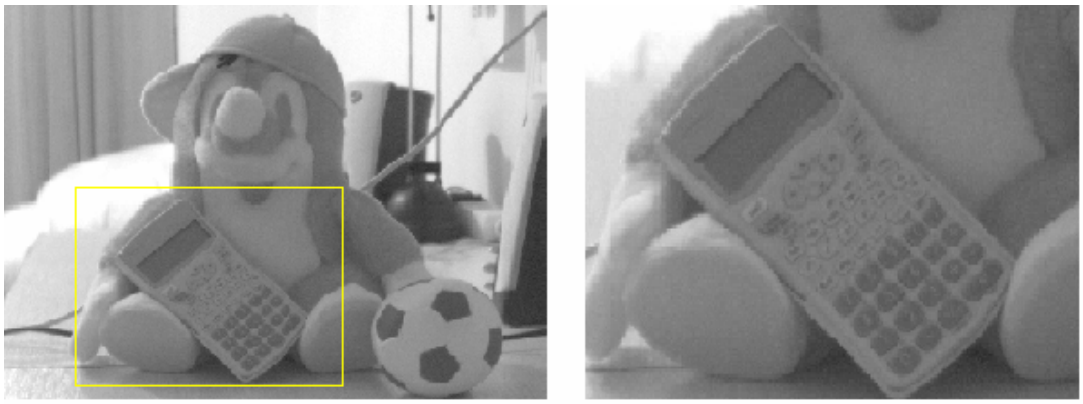
Parametr softwarové ostrosti (sharpness) ovlivňuje ostrost získaného obrazu tím, že přímo v kameře aplikuje na obraz ostřící algoritmus, který využívá nějaký typ ostřícího filtru, který se mění s hodnotou parametru ostrosti. Přílišným softwarovým ostřením ovšem zvýrazňujeme šum a nevýznamné hrany v obraze, které nenesou žádnou důležitou informaci. Vliv tohoto parametru je názorný v **Obrázek 5.9**, kde zvýšením této hodnoty výsledný obraz zaostříme.

Původní obrázek Zvýšená softwarová ostrost

Obrázek 5.9: Vliv parametru softwarová ostrost

5.8 ZOOM

Název zoom vznikl od charakteristického zvuku motorků. Pomocí parametru zoom je možné měnit ohniskovou vzdálenost. Touto změnou ohniskové vzdálenosti se určuje výřez snímané scény. Se zvětšením zoomu se tedy snižuje zorný úhel, pod kterým snímáme sledovanou scénu. Změna ohniskové vzdálenosti ovšem ovlivňuje i mnoho dalších parametrů jako jsou například hloubka ostrosti a expozice. Vliv tohoto parametru na výsledný obraz je ukázán na **Obrázek 5.10**.

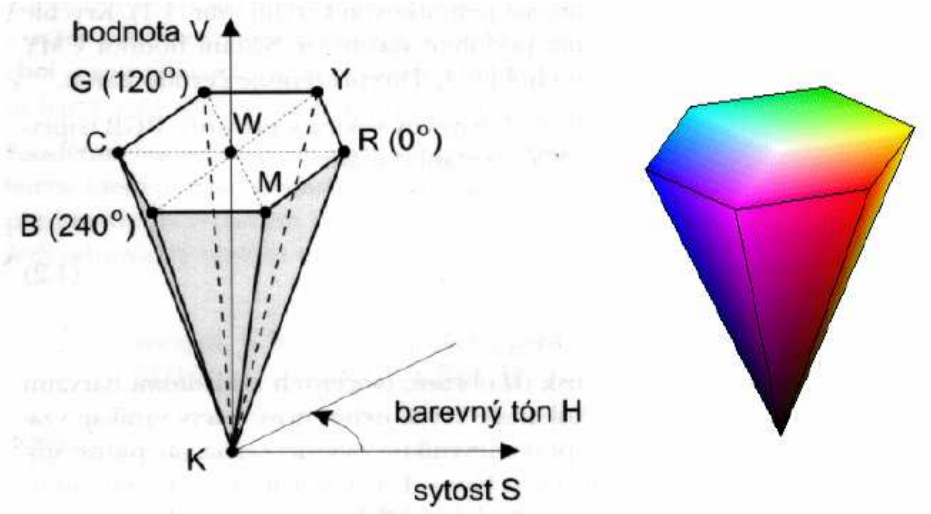


Zoom = 0 Maximální zoom

Obrázek 5.10: Vliv parametru zoom

5.9 JAS

Parametr jasu (offset, brightness) představuje hodnotu offsetu, která je připočítávána k výstupnímu signálu ze senzoru. Tímto parametrem můžeme celý výsledný obraz zesvětlit. K jednotlivým pixelům výstupního obrazu je připočítávána konstanta. Tento parametr je názorný na **Obrázek 5.11**, kde v barevném modelu HSV určuje velikost achromatické složky (hodnota V). Vliv toho parametru je dále ukázán na výsledném obraze (**Obrázek 5.12**), kde nepřímo ovlivňuje expozici výsledného obrazu.



Obrázek 5.11: Barevný model HSV

Původní obrázek Zvýšená hodnota jasu

Obrázek 5.12: Vliv parametru jasu

5.10 GAMMA

Parametr gamma snižuje nebo zvyšuje světlost středních tónů barev v získávaném obraze. Tímto parametrem můžeme například kompenzovat nelineární chování jasu obrazovky. Tohoto se hojně využívalo u CRT monitorů, kde se tímto kompenzovala nelineární závislost mezi napětím na elektrodě a jasem obrazovky, kdy relativní jas je dán vztahem (5.7).

$$L = c * U^{g} \tag{5.7}$$

kde L je relativní jas

c je konstanta

U je napětí na elektrodě [V]

g je právě hodnota gamma.

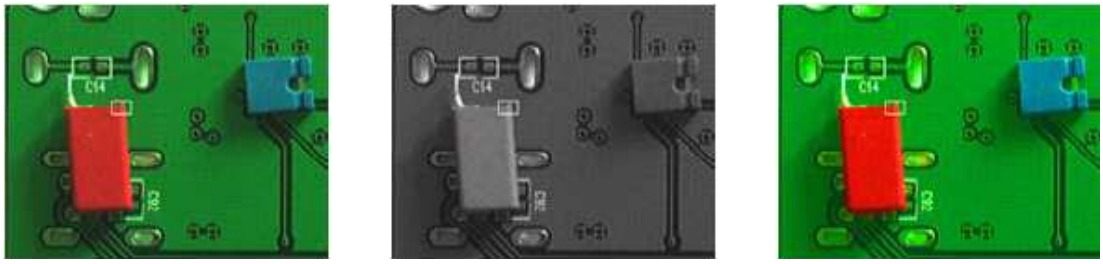
Při nevhodném nastavení tohoto parametru se začíná projevovat tzv. posterizace, která představuje nerovnoměrný přechod mezi jednotlivými odstíny (**Obrázek 5.13**).

Původní obrázek Posterizovaný obrázek

Obrázek 5.13: Posterizace obrazu

5.11 SATURACE

Saturace (saturation) je parametr, kterým je možno ovlivnit míru sytosti získávaného obrazu, což odpovídá ovlivnění vzdálenosti S v modelu HSV (**Obrázek 5.11**). Je tudíž možno z barevné kamery získat šedotónový obraz. Snížením tohoto parametru je tedy snížena sytost získávaného obrazu a zvýšením tohoto parametru je naopak možno získat obraz, který bude mít velikou sytost.

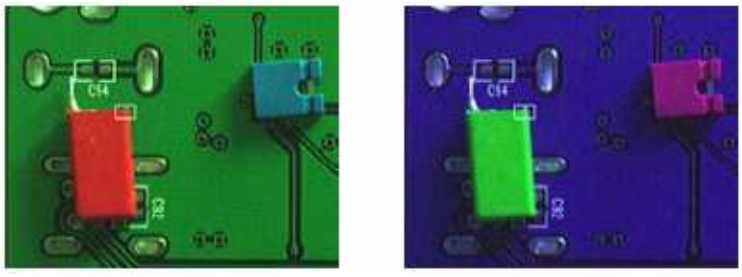


Původní obraz | Saturace = 0 | Maximální saturace

Obrázek 5.14: Vliv parametru saturace [11]

5.12 BAREVNÝ TÓN

Parametrem barevného tónu (hue) lze nastavit přepínání barevných hodnot. Toto přepínání ovšem nemění vztah mezi jednotlivými barevnými tóny v získávaném obraze. Příklad může být ukázán na intuitivním modelu HSV (**Obrázek 5.11**), kde je každý bod v obraze zastoupen třemi parametry: jasem (V), sytostí (S) a barevným tónem (H). Barevný tón zde představuje úhel H. Při změně parametru barevný tón se k tomuto úhlu H přičte konstantní úhel. Tímto se pro každý bod v získávaném obraze změní barevný tón a jejich vzájemný vztah se nezmění.



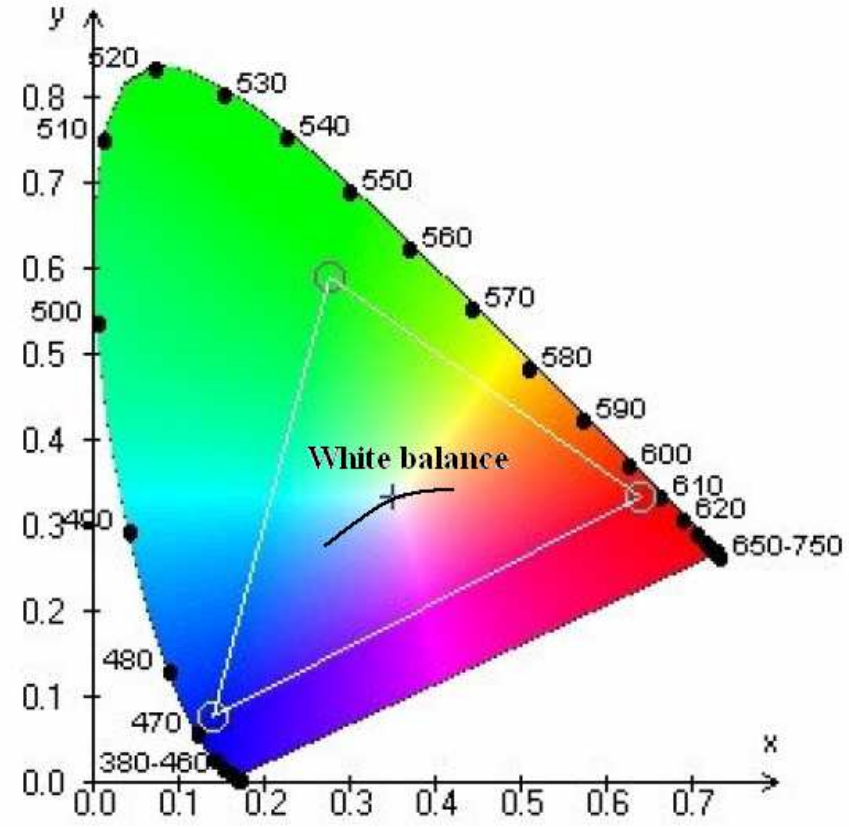
Původní obraz | Posunutý barevný tón

Obrázek 5.15: Vliv parametru barevný tón [11]

5.13 VYVÁŽENÍ BÍLÉ BARVY

Parametr vyvážení bílé (white balance) představuje možnost vyvážení bílé barvy v obraze při naprosto rozdílných světelných podmínkách (umělé osvětlení, denní světlo,….) tak, aby věrohodně představovala snímanou scénu. Vyvážení bílé je možno si představit na gamutu RGB v chromatickém diagramu, kdy se posouvá ve středu tohoto trojúhelníku tak, aby bílá barva věrně představovala bílou ve snímané scéně. Změna tohoto parametru v RGB trojúhelníku ovšem způsobí to, že při posunu k teplejším odstínům (červená) se sníží podíl modré a naopak. Pohyb je možno provést pouze po nějaké trajektorii. Proto kvalitnější kamery mají dva parametry (white balance red a white balance blue), kterými lze nezávisle nastavit stupeň červené a modré.

U fotoaparátů se věrného barevného podání dosahuje výměnou filmu podle daného osvětlení (filmy se vyrábí pro denní/bleskové světlo a na umělé žárovkové osvětlení).



Obrázek 5.16 Chromatický digram [1]

5.14 KOMPENZACE SVĚTLA NA POZADÍ

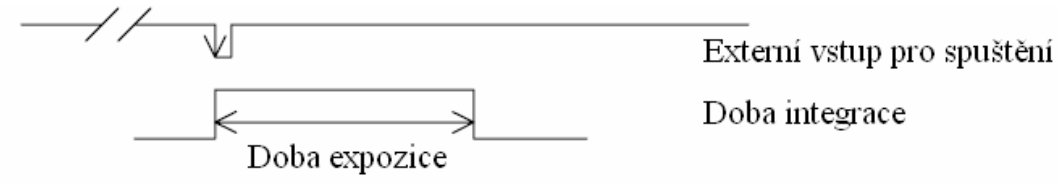
Parametr kompenzace není definován v uvedené literatuře [13], ale jeho vliv na výsledný obraz bude ukázán. Kompenzace světla na pozadí (backlight compenzation) má pouze dva stavy a to zapnuto a vypnuto. Při zapnutí tohoto parametru se zapne algoritmus, který dokáže kompenzovat oblasti s vysokým jasem (lampa, baterka, ….). Při nezapnutí tohoto vlivu by se celkový obraz jevil jako přeexponovaný. Vliv tohoto parametru je ukázána na **Obrázek 5.17**.

Vypnutá kompenzace světla na pozadí Zapnutá kompenzace světla na pozadí

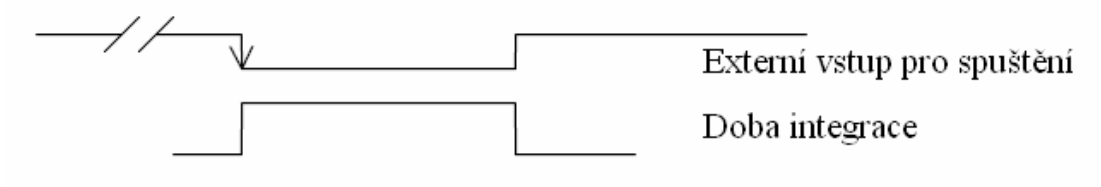
Obrázek 5.17: Vliv parametru kompenzace světla na pozadí

5.15 OVLÁDÁNÍ SPUŠTĚNÍ

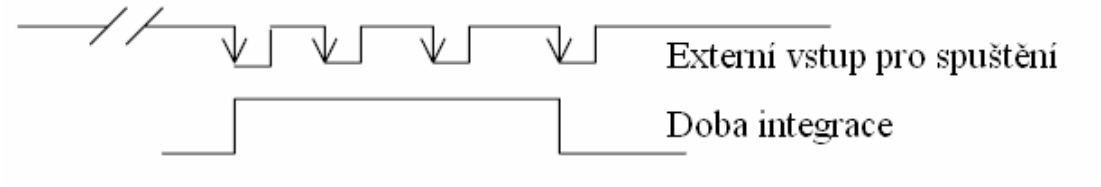
Ovládání spuštění expozice [13] má čtyři módy 0, 1, 2, 3. Mód nula je ukázán na **Obrázek 5.18** a v tomto módu je kamera spuštěna externě a doba integrace je závislá pouze na době, která je nastavená v parametru expozice.



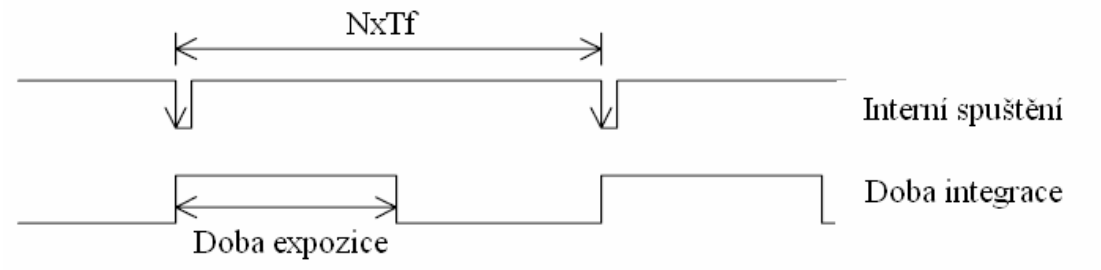
Obrázek 5.18: Spouštění v módu nula



Obrázek 5.19: Spouštění v módu jedna



Obrázek 5.20: Spouštění v módu dva



Obrázek 5.21: Spouštění v módu tři

V módu jedna je doba integrace závislá na externím spouštěcím signálu, který určuje dobu integrace, jak je patrné z **Obrázek 5.19**. V módu dva je doba integrace spuštěna sestupnou hranou externího signálu a po N (volíme) hranách je tato integrace ukončena. V posledním módu je situace podobná, ale s tím rozdílem, že se kamera spouští v závislosti na N periodách nejrychlejší doby počtu snímků za sekundu, doba integrace je závislá na parametru expozice.

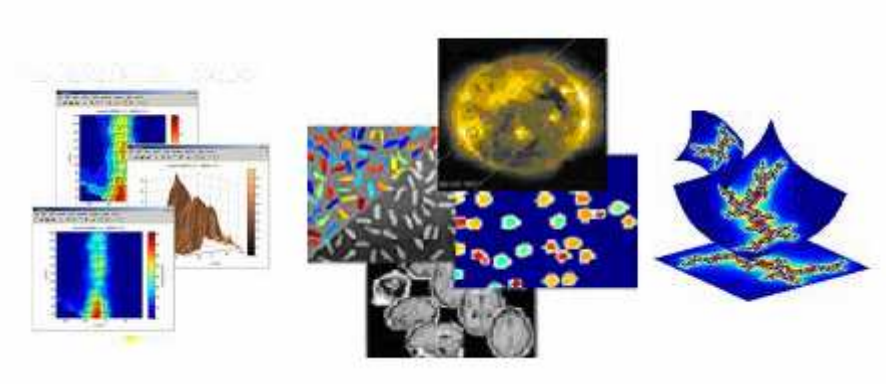
5.16 OSTATNÍ PARAMETRY

Další parametry, které jsou pro kamery definovány ve standardu DCAM/IIDC [13] jsou například aktuální teplota v kameře, horizontální vychýlení kamery (pan), vertikální vychýlení kamery (tilt), které mohou být použity například pro sledování pohyblivého objektu. Dále pak řízení změny optického filtru atd.

6 POČÍTAČOVÉ VIDĚNÍ V PROSŘEDÍ MATLAB

Tento programový systém vyvinula firma MATHWORKS a název MATLAB vznikl z anglického výrazu MATrix LABoratory. MATLAB je velice výkonným jazykem pro vědecké a technické výpočty, zejména v maticových operacích reálných a komplexních čísel, což je výhodné pro operace počítačového vidění. MATLAB byl implementován na všech významných platformách, jako jsou Windows, Linux, Solaris, Mac. Díky své architektuře je MATLAB určen i těm, kteří neznají zcela matematickou podstatu problémů. Je to způsobeno velikou variantností knihoven MATLABU. Knihovny svým rozsahem pokrývají prakticky všechny oblasti lidské činnosti a díky otevřené architektuře je uživateli umožněno vytvářet funkce dle své potřeby. Tyto funkce jsou volány stejně jako vestavěné funkce tohoto prostředí. Vestavěné funkce jsou definovány v knihovnách, které se v prostředí MATLAB nazývají toolboxy. Tyto knihovny se neustále rozšiřují dle vývoje vědních a technických oborů. Další znak MATLABu je také provázanost s jinými programovacími jazyky, jako jsou například Java, C a Fortan, což ještě více rozšířilo pole působnosti tohoto programového prostředí. MATLAB také podporuje tvorbu grafických uživatelských rozhraní, pomocí programové nadstavby GUIDE. GUIDE od verze 7.3 je MATLAB rozšířen o kompilátor. Ten dokáže vytvořit aplikaci, která běží samostatně, tj. bez nutnosti mít nainstalován produkt MATLAB. Nastavení a kompilace je možné obsluhovat pomocí tooboxu deploytool. Nevýhodou této aplikace je ovšem velikost, která je jen u prázdné aplikace zhruba 150 MB. Tato velikost je způsobena přiložením toolboxu reálného času. Další nevýhodou je rychlost aplikace, která se nemůže rovnat klasickým programovacím jazykům. To je ovšem nahrazeno širokou škálou využití a velikou rozmanitostí funkcí, které lze použít. Pro potřeby počítačového vidění jsou v MATLABu implementovány toolboxy Image Processing Toolbox, Image Acquisition Toolbox a Mapping Toolbox. Nebo v nadstavbě SIMULINK Video and Image Processing Blockset.

6.1 IMAGE PROCESSING TOOLBOX



obrázek 6.1: Image processing toolbox [6]

Image Processing toolbox je knihovna funkcí, která poskytuje nástroje pro zpracování, analýzu a vizualizaci obrazu. V tomto toolboxu jsou obsaženy funkce pro základní operace, které jsou potřebné pro počítačové vidění, jako jsou například konvoluce, načítání obrazu, ukládání obrazu, základní filtry a různé 2D transformace. Image Processin toolbox podporuje práci s obrazy generovaných z mnoha zařízení, jako jsou například digitální kamery, digitalizační karty, letecké snímkovací zařízení, mikroskopy nebo lékařské diagnostické přístroje. Velkou výhodou je široká škála datových typů, ve kterých se při práci s obrazy může pracovat (float, signed, unsigned 8-, 16-, 32-bitový int) a také mnoho datových formátů, ve kterých jsou obrazy načítány a ukládány (JPEG, TIFF, PNG, GIF, BMP, HDF, PCX, XWD, ICO, CUR, RAS, PBM, PGM, PPM). Jsou zde i funkce operací předzpracování obrazu, které již byly probrány v předcházející kapitole. Jsou zde funkce pro eliminování šumu, zvýraznění hran i restaurace obrazu. Tyto funkce využívají různé filtry, které si můžeme sami nadefinovat, nebo použít definované filtry v toolboxu. Jsou zde definované hranové filtry (Sobel, Prewitt, Roberts, Canny, Laplacův, Gaussův), ostřící filtry, rozmazávací a mnohé jiné.

V některých případech počítačového vidění je zapotřebí různých typů transformací, jako jsou například FFT (Fast Fourier Transformation), DCT (Discrete cosine transform) a jiné. Tyto transformace jsou základem pro operace s obrazem (komprese, restaurace, analýza). DCT je využito například v kompresi obrazu do formátu JPEG. Samostatně je tato transformace definována v Signal Processing Toolbox. Jak již bylo dříve zmíněno, Image Processing Toolbox umožňuje práci s širokou škálou datových typů a formátu obrazu. Převod formátu obrazu je vyřešen přímo jako funkce toolboxu, které dokáží převádět mezi formáty typu binární obraz, černobílý obraz, barevný obraz, obrazy s různými barevnými modely (YIQ, HSV, YCrCb). Další funkce tohoto toolboxu jsou například automatické prahování, morfologické operace (eroze, dilatace, binární otevření, uzavření), Houghova transformace, operace úpravy jasu obrazu, operace segmentace, rotace obrazu a jiné.

6.2 IMAGE ACQUISITION TOOLBOX

obrázek 6.2: Image acquisition toolbox [6]

Image Acquisition Toolbox je další knihovna, která byla vyvinuta pro počítačové vidění. Tato knihovna slouží k získávání dat obrazu nebo videa přímo ze zařízení. Detekování a nastavení hardwarového zařízení je možné pomocí této knihovny provádět automaticky. Získávat data je možné z webových kamer, grabovacích karet a jiných zařízení.

Pomocí tohoto toolboxu je možné ukládat video přímo do souboru ve formátu AVI. Pomocí cquisition Toolbox je možno automaticky detekovat kompatibilní zařízení a také připojit více zařízení zároveň k PC. Nastavit je možné video formát, rozlišení, zobrazovat oblast zájmu (ROI), počet snímků za sekundu nebo barevný model a jeho parametry (odstín, nasycení, jas, kontrast). Propojením MATLABu, Image Acquisition Toolbox, Image Processing Toolbox a graphical user interfaces (GUIDE) je tedy možné vytvořit aplikaci, která dokáže pokrýt téměř všechny úlohy počítačového vidění.

6.3 REALIZACE V PROGRAMOVÉM PROSTŘEDÍ MATLAB

Pro vytvoření aplikace pro ovládání kamer v programu MATLAB předcházejících dvou tooloxů a toolboxu GUIDE (graphical user interface development environment). Výsledná aplikace je převedena na samostatnou aplikaci, kterou lze spustit na počítači, který nemá nainstalované toto programové prostředí. Pro vytvoření aplikace byla použita verze MATLAB 7.3 (R2006b).

6.3.1 Příklad použitých instrukcí

Na tomto příkladu bude ukázáno obecné nastavení image acqusition a nastavení v jazyku MATLABu. Typické obecné schéma obsahuje čtyři kroky a vypadá následovně:

- Vytvoření objektu vstupních dat
- Nastavení parametrů objektu vstupních dat pro zařízení a otevření okna a zobrazení vstupních dat.
- Uložení videa z objektu vstupních dat, zapisování a zpracování dat
- Vymazání objektu vstupních dat

Vytvoření objektu vstupních dat

Nejprve je nutné zjistit názvy kamer, které jsou nalezeny adaptéry. Adaptéry jsou součástí programového prostředí MATLAB a poskytují komunikaci mezi kamerou a image acqusition toolboxem prostředí MATLAB. Příklad zjištění použitelných adaptérů možné pomocí funkce *imaqhinfo*. Následující dva příkazy tedy uloží strukturu, která obsahuje seznam dostupných adaptérů a následně strukturu s názvy kamer a identifikační čísla, které náleží danému adaptéru.

```
seznam_adaptors=imaqhwinfo;
seznam_kamer_adapteru = imaqhwinfo(název adaptréru);
```

Nyní je možno vytvořit objekt vstupních dat videa s názvem *vid* pomocí funkce *videoinput*. Tomuto objektu přiřadíme jednotlivé hardwarové zařízení pomocí identifikačního čísla. Pomocí příkazu *getselectedsource* je tato kamera vybrána a je možno měnit její parametry, které se dají zjistit pomocí funkce *propinfo*. Příklad vytvoření objektu vstupních dat v jazyku MATLABu může vypadat následovně:

```
vid = videoinput (‘winvideo’,1);
src = getselectedsource(vid);
parametry_vybrane_kamery=propinfo(src);
```

Nastavení parametrů objektu vstupních dat pro zařízení

Dalším krokem je zobrazení videa v grafickém objektu *axes* v grafickém prostředí MATALB a nastavení parametrů. Zde je uveden příklad nastavení jasu a nastavení mohou vypadat následně:

```
hImage=image(zeros(šířka,výška,hloubka), 'Parent', ukazatel na axes objekt);
set (src, ‘Brightness’, 100);
start(vid);
preview (vid,hImage);
```

Uložení videa objektu vstupních dat, zapisování a zpracování dat

Nyní je možno uložit snímek nebo celé sekvence z kamery. Pro vykreslení montáže do objektu *axes* je potřeba nastavit ukazatel na tento grafický objekt třeba pomocí příkazu *gca*, protože tato funkce nemá stejné parametry jako funkce *preview*.

```
snímek = getsnapshot(vid);
sekvence=getgata(vid);
imaqmontage(sekvence);
```

Nyní je možno provádět různé operace se získaným obrazem nebo sekvencí, jako třeba hranování, segmentaci, atd.

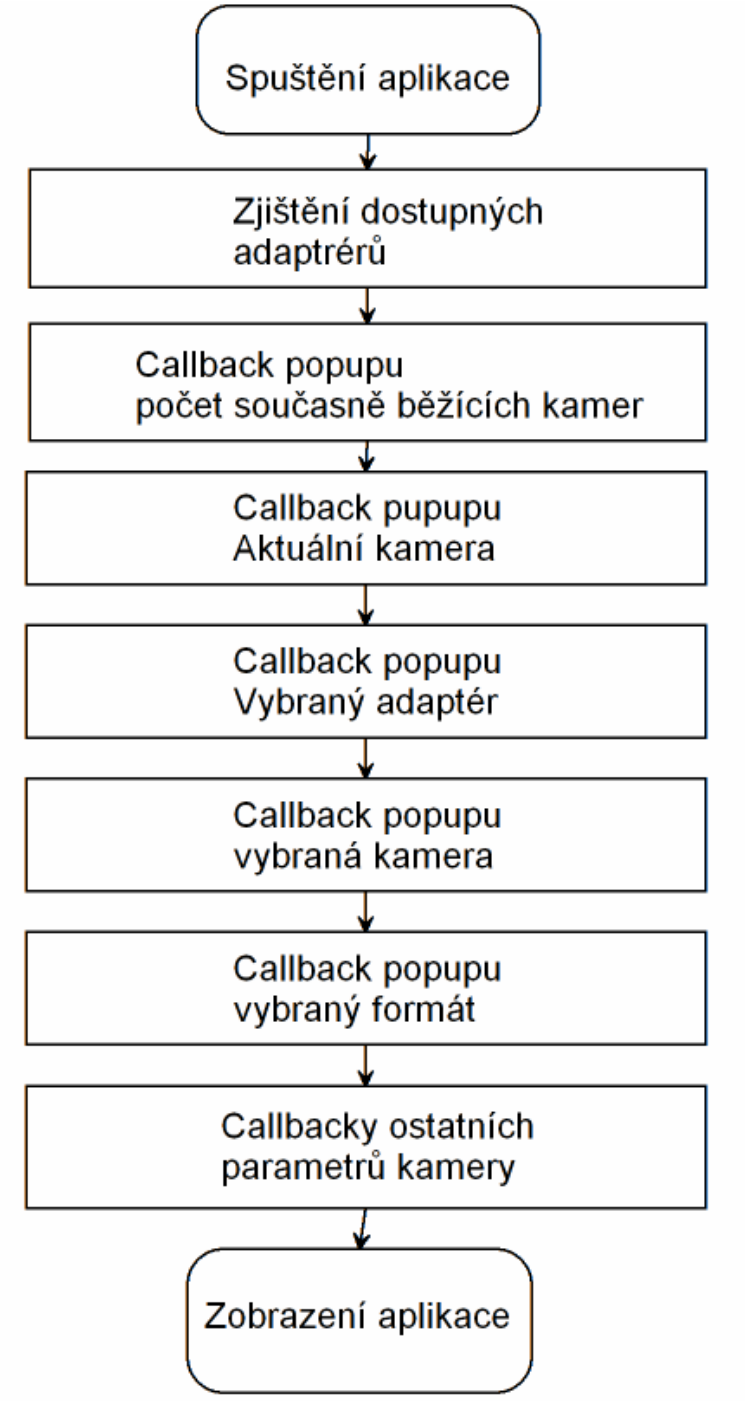
Vymazání objektu vstupních dat

V posledním kroku je nutno zastavit zobrazování videa a smazat objekt vstupních dat u MATLAB Workspace k uvolnění paměti. Tyto funkce vypadají následovně:

```
stop(vid);
delete(vid);
clear vid;
```

Nebo je také vhodné zavolat funkci *imaqreset*, která odpojí a smaže všechny objekty videa.

6.3.2 Realizace aplikace v programovém prostředí MATLAB



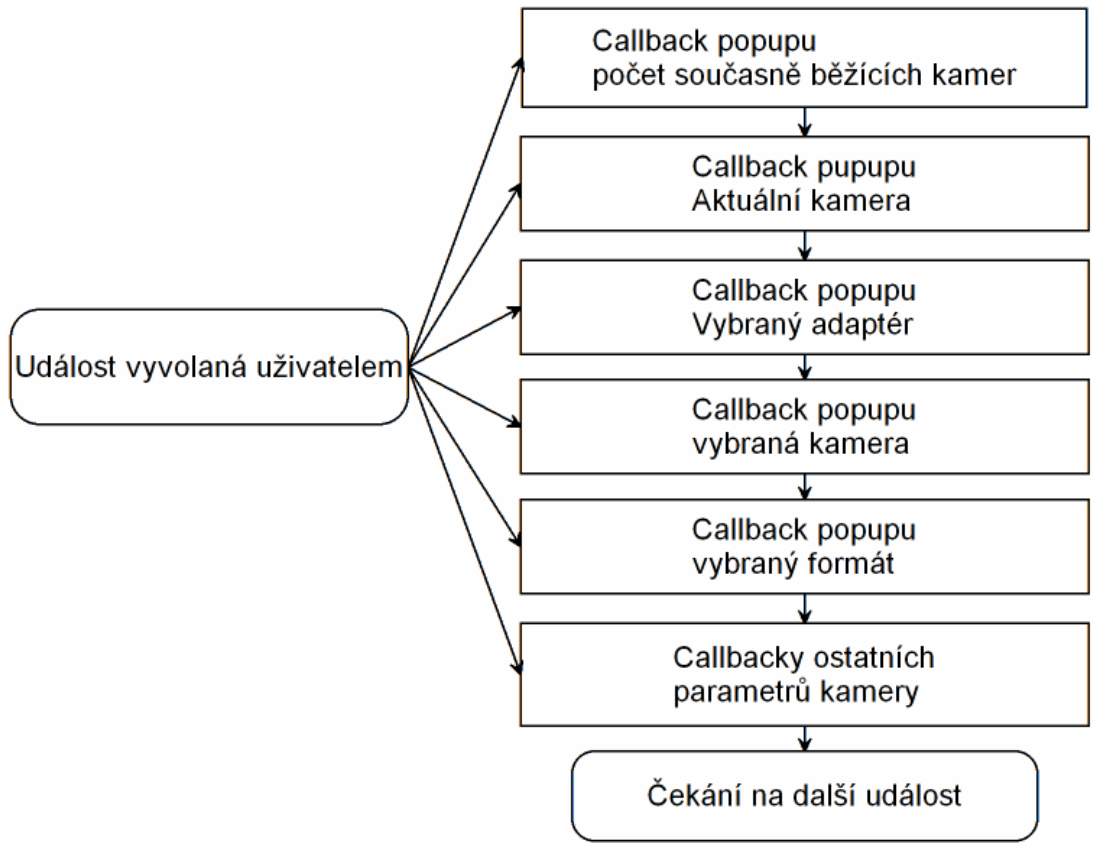
Obrázek 6.3: Vývojový diagram spuštění aplikace

Po spuštění aplikace proběhne program dle vývojového diagramu na **Obrázek 6.3**. Z vývojového diagramu je zřejmé, že celá aplikace se musí hned po zapnutí a ještě před zobrazením sama nastavit, to jest nastavit všechny grafické ovládací prvky.

Nejprve jsou zjištěny všechny dostupné adaptéry. Poté je zavolán callback rozbalovacího menu počtu současně běžících kamer jako klasická funkce. Tento callback při svém prvním spuštění vždy nastaví roletové menu (popup menu) počtu současně spuštěných kamer na první položku, což představuje jednu spuštěnou kameru. Podle této vybrané položky callback upraví menu popupu aktuální kamery. Dále program pokračuje na callback aktuální kamery, kde nastaví v popup menu aktuální kamery ukazatel na první položku. Takto postupuje program přes celý diagram, kdy vždy callback aktuálního roletového menu nastaví položky následujícího až po callback rozbalovacího menu podporované formáty.

Po zavolání callbacku aktuální kamery se tedy nastaví roletové menu adaptéru na první dostupný adaptér a pokud je pod tímto adaptérem nalezeno nějaké zařízení, tak nastaví další roletové menu připojeného zařízení na první nalezenou kameru. Ostatní prvky jsou nastaveny na defaultní hodnotu, pokud kamera tento parametr umožňuje nastavit a pokud kamera neumožňuje nastavit nějaký parametr, pak je ovládací prvek tohoto parametru zneviditelněn, nebo jako v případě popupu objektů, například podporovaného formátu, je nastavena položka na ‘*není nalezen podporovaný formát*‘.

Pokud bylo nalezeno nějaké multimediální zařízení, je k tomuto zařízení vytvořen objekt videa, jak již bylo zmíněno v předcházejícím textu. Po provedení celého diagramu je aplikace zobrazena čeká na událost vyvolanou uživatelem. Následné chování aplikace je podobné jako při spuštění aplikace, pouze s tou výjimkou, že callback může vyvolat i uživatel tím, že změní nějaký grafický objekt a tím následně vyvolá událost, která spouští callback daného objektu. To je zobrazeno na vývojovém diagramu na **Obrázek 6.4**.



Obrázek 6.4: Vývojový diagram běhu aplikace MATLAB

Po počáteční inicializaci čeká aplikace na událost, kterou vyvolá uživatel tím, že změní nějaký grafický objekt. Při změně položky v seznamu popup menu počtu současně běžících kamer se vyvolá callback a aplikace zareaguje tím, že se vykoná zbytek digramu tak, že se všechny prvky nastaví na defaultní hodnotu nebo na první položku v případě rozbalovacího seznamu. Dále tento callback vytvoří v grafickém objektu panel tolik objektů axes, kolik je vybráno současně zobrazených kamer.

Při změně položky v rozbalovacím menu aktuální kamery je popup adaptéru nastaven na minulou položku ze seznamu adaptéru, pokud byl již adapter uživatelem nastaven, nebo na první nalezený adaptér ze seznamu v případě, že ho uživatel ještě nenastavil. Pro daný adaptér se nalezne seznam kamer a z tohoto seznamu jsou odstraněny ty kamery, které jsou již nastaveny pod jinými pozicemi aktuální kamery.

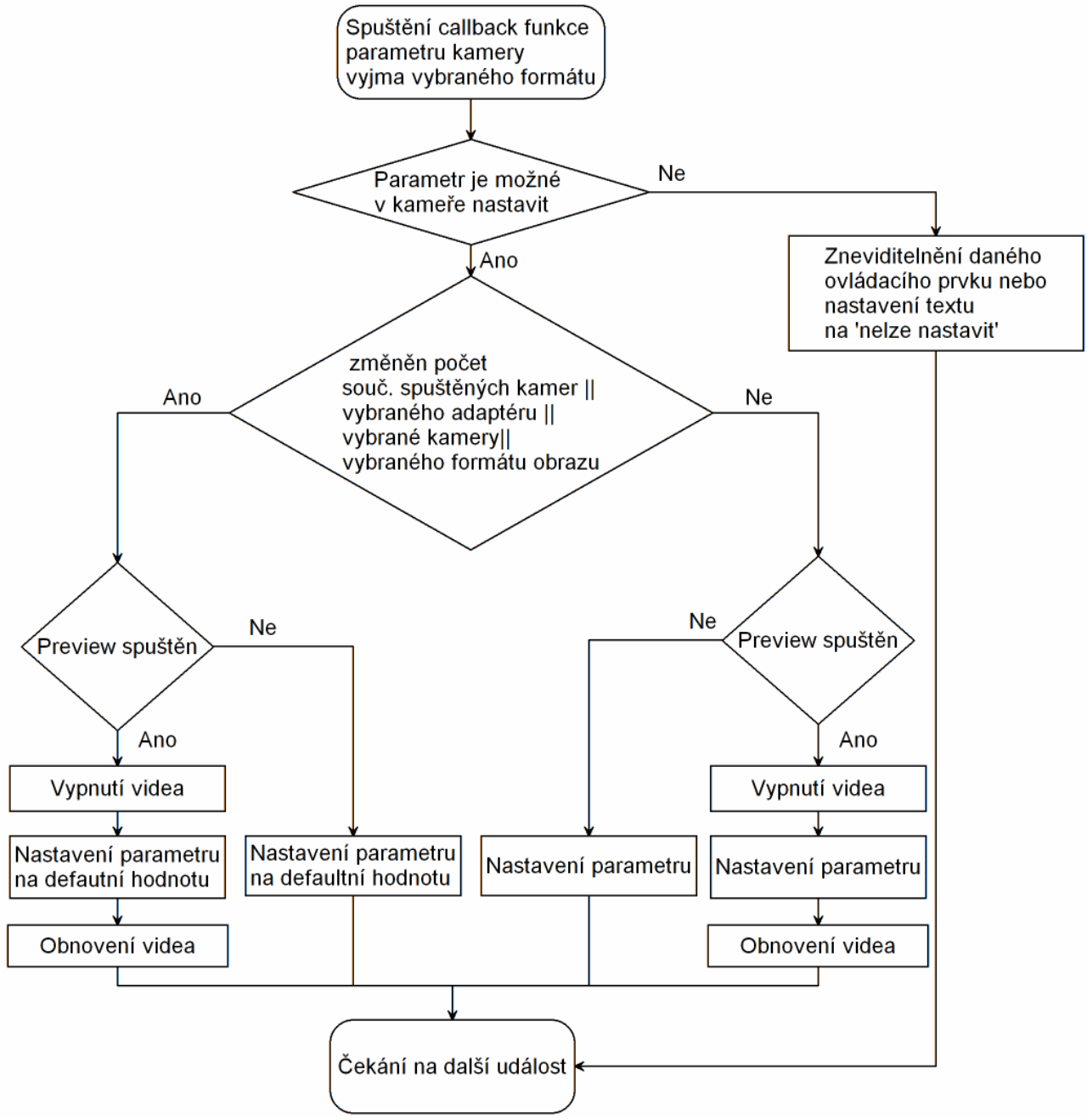
Kamera se nastaví v tomto zredukovaném seznamu kamer opět na minule nastavenou položku v případě, že již byla uživatelem nastavena nebo na první položku, pokud kamera ještě nastavena nebyla. Pokud není nalezena žádná kamera, pak se popup připojeného zařízení nastaví na *'Nenalezeno žádné zařízení'*.

Po nastavení kamery je dalším krokem nastavení popupu podporovaných formátů. Princip je opět stejný, ze seznamu je vybrána ta položka, která byla nastavena uživatelem již dříve a v případě, že uživatel ještě nenastavil žádný formát obrazu, pak je nastaven na první položku ze seznamu formátu obrazu. Pokud nebyla nalezena žádná kamera, pak je roletové menu podporovaného obrazu nastaveno na *'Nenalezen žádný podporovaný formát'*. Poslední roletové menu je počet snímků za sekundu a pokud není tento parametr nalezen je položka nastavena na *'není možno nastavit'*.

Ostatní ovládací prvky parametru kamery jsou nastaveny na defaultní hodnotu v případě, pokud uživatel změnil počet současně zobrazovaných kamer, použitý adaptér, kameru pod pozicí aktuální kamery nebo formát obrazu a pokud daná kamera tento parametr umožňuje nastavit. Tato obsluha je ještě složitější, pokud je spuštěno zobrazování videa. Poté se musí pozastavit tok dat z videa nastavit parametry a poté opět tok obnovit. Vývojový diagram takového nastavení daného ovládacího prvku je zobrazen na **Obrázek 6.5**.

Dalším krokem je vytvoření objektu videa. Objekty videa nemohou být uloženy do MATLABovského pole tak, kdy jsou ostatní prvky pole nulové, takže je nutno tyto objekty vždy uložit tak, aby pole bylo bez mezer (nulových prvků).

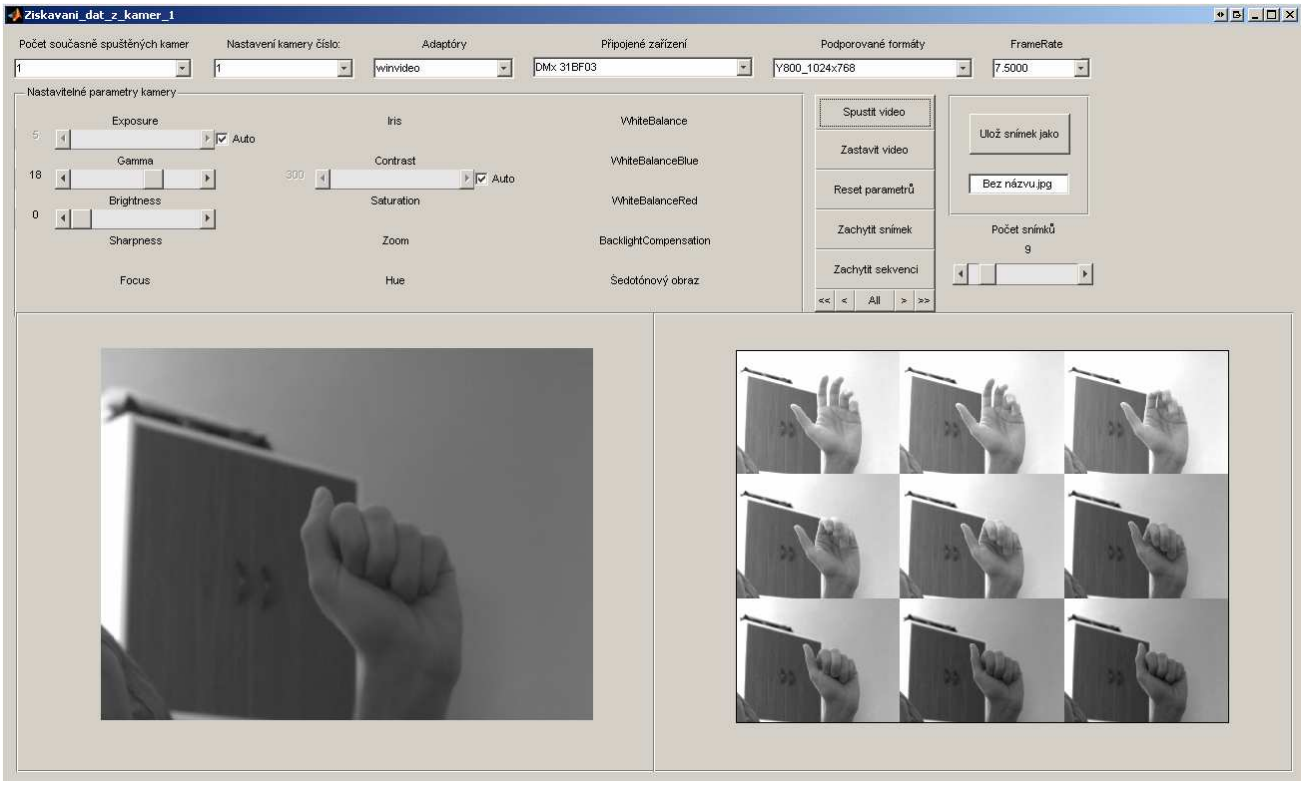
Další problém je se zobrazováním funkce *imaqmontage* a *imshow*, které zobrazují získané obrazy v grafickém objektu axes. Pokud jsou tyto objekty axes vytvořeny jako podobjekty grafického objektu panel za pomocí funkce *subplot*, tak neexistuje žádný způsob, jak těmto funkcím předat ukazatel na tento grafický objekt, proto museli být vytvořeny grafické objekty samostatně a za pomocí funkce *subplot*. Funkce preview předávání ukazatele na tento objekt podporuje, jak ukázáno v kapitole příklady použitých instrukcí.



Obrázek 6.5: Vývojový diagram callback funkce parametru kamery

6.3.3 Výsledná aplikace MATLAB

Vzhled výsledné aplikace v programovém prostředí MATLAB je zobrazen na **Obrázek 6.6**. Celkový vzhled aplikace je také zobrazen v příloha [1] a [2]. Tato aplikace umožňuje uživateli nastavovat jednotlivé parametry dané kamery (Podporovaný formát, FPS, Expozice, atd), nastavit parametry kamery na defaultní hodnotu a současně sledovat video z více kamer. Také tato aplikace poskytuje možnost zachycení snímku z dané kamery a nebo sekvence snímků, které jsou zobrazeny jako montáž a lze se v této sekvenci pohybovat a ukládat tyto snímky na disk. Realtimové zachycení snímků nebo sekvence z více kamer není funkční, protože funkce zajišťující paralelní běh smyček *parfor*, sice relativně zkrátí prodlevy mezi jednotlivými funkcemi, ale nezpracovává tyto smyčky zcela paralelně, což může být způsobeno tím, že procesor v systému, na kterém byla tato aplikace testována, má pouze jedno jádro.



Obrázek 6.6: Výsledná aplikace v programovém prostředí MATLAB

Výsledná aplikace byla také převedena do formy, kterou je možno spustit na jakémkoliv počítači bez nutnosti nainstalování programového prostředí MATLAB. V MATLABU k tomuto účelu slouží příkaz *deploytool*, který spustí průvodce vytvořením samostatně spustitelného projektu. V tomto průvodci se založí nový projekt a v záložce MATLAB *compiler* zvolí *standalone application* a nastaví se jméno projektu. Poté se v nově otevřeném dialogu do *main function* vloží *m-file* soubor a do *other files fig-file* soubor dané aplikace. V nastavení projektu je ještě nutno zaškrtnout *include MATLAB Component Runtime(MCR)*. Poté už je možno projekt vytvořit. Při problému vytvoření projektu je možno využít funkce *mbuild –setup*, který zvolí překladač pro vytváření projektu.

Tímto je vytvořen projekt ve spustitelné formě s příponou exe, ale tato aplikace je spustitelná stále pouze na počítači, kde je MATLAB nainstalován. Proto je nutno v posledním kroku při vytvoření projektu přibalit knihovny MCR pomocí tlačítka *package the components*. Tímto je vytvořena samostatná aplikace spustitelná na počítači, kde není nainstalován MATLAB.

Při problémech se spuštěním aplikace je pak ještě nutno nastavit cestu k MCR knihovnám v *Tento počítač->vlastnosti->upřesnit->proměnné prostředí*

PATH <cesta k MCR knihovnám>\<verze MCR>\runtime\win32

7 POČÍTAČOVÉ VIDĚNÍ S OPENCV

Společnost INTEL zareagovala na rychlý rozvoj v počítačovém vidění vývojem knihoven IPP (Integrated Performance Primitives) a OpenCV (Open Computer Vision).

IPP (Integrated Performance Primitives)

Intel IPP je jedna z knihoven IPL (Intel® Performance Libraries), které využívává i knihovna OpenCV. IPP seskupilo procesory Intel® Pentium®, Itanium® a Intel® Personal Internet Client Architecture (Intel® PCA) do jediného celku se společnými API v rámci všech architektur. IPP poskytuje funkce pro zpracování signálů a počítání s maticemi. Toho se využívá v oblastech kryptografie, zpracování řetězců, audio, video, kódování a rozpoznávání řeči a dokonce i vylepšenou podporu malého otisku. Nejnovější verze IPP je 4.0, která je více popsána v literatuře [9]. Zde jsou popsány všechny funkce a příklady použití.

OpenCV (Open Computer Vision)

OpenCv je otevřená knihovna od společnosti INTEL, vyvíjená touto společností od roku 1999. Tato knihovna jazyka C/C++ obsahuje vysokoúrovňové funkce pro zpracování obrazu a pro počítačové vidění obecně. V této knihovně je definováno mnoho vysokoúrovňových datových typů, mezi které patří například matice, grafy nebo množiny. Tyto funkce vždy začínají písmeny cv. Jsou zde obsaženy funkce geometrie, hranování, Houghovy transformace, morfologie, segmentace, odečtení pozadí, samoprahování, kalibraci kamer, optického toku, histogramu, rozeznávání gest, operace s maticemi, a mnoho dalších. Všechny funkce jsou popsány v literatuře [10]. Tato knihovna se skládá ze šesti hlavních modulů CV (hlavní obrazové funkce pro zpracování obrazu a počítačového vidění), CXCORE (Datové struktury, funkce pro operace lineární algebry, funkce pro kreslení), HIGHGUI (funkce pro zobrazení, načítání obrazu a práce s myší), CVAUX (doplňující funkce a experimentální funkce), CVCAM (funkce pro zpracování dat z kamer a kalibraci kamer) a ML (metody strojového učení). Nejnovější verze

OpenCV je 1.0 a lze ji získat z internetu, protože se jedná o veřejnou knihovnu (pro nekomerční využití). Po instalaci je nutno nastavit cestu do aktuálního adresáře. OpenCV je navrženo pro aplikaci Visual studio. OpenCV je ovšem možno také aplikovat do prostředí C++Builder, ale je nutno překompilovat hlavní knihovny, protože obě aplikace nepoužívají stejný kompilátor.

7.1 REALIZACE V PROGRAMOVÉM PROSTŘEDÍ C++BUILDER

Pro vytváření aplikace pro ovládání kamer v programu C++Builder bude využito již dříve zmíněné knihovny OpenCV, které jsou poskytovány firmou INTEL. Tato knihovna poskytuje vysokoúrovňové funkce a datové typy, které ulehčují operace spojené se získáváním dat z kamer a dalším zpracováním získaného obrazu.

7.1.1 Implementace knihoven OpenCV do Microsoft Visual C++ 6.0

Po nainstalování knihoven OpenCV je nutno nastavit programové prostředí Microsoft Visual C++ 6.0. V programovém prostředí je nutno v záložce hlavního menu *Tools -> Options -> Directories* nastavit Platformu na *Win32*, zvolit *include files* v roletovém menu *Show directories for* a nastavit cestu do adresářů OpenCV, kde jsou nainstalovány hlavičkové soubory k již dříve zmíněným modulům.

<cesta k OpenCV>\ cv\include
< cesta k OpenCV >\ cxcore\include
< cesta k OpenCV >\ cvaux\include
< cesta k OpenCV >\ ml\include
< cesta k OpenCV >\ otherlibs\highgui
< cesta k OpenCV >\ otherlibs\cvcam\include

A také je nutno nastavit cestu ke knihovnám, aby tyto knihovny mohly být přilinkovány k projektu tak, že v roletovém menu *Show directories for* je vybrána položa *library files* a přidána cesta.

<cesta k OpenCV>\ cv\include

Po založení projektu se poté připíší knihovny, které mají být přilinkovány. To je možné provést v *Project -> Settings*, kde se nastaví v roletovém menu *Settings for* na *All Configurations*, v záložce *link* se nastaví *Categhory* na *General* a do editačního okna *Object / library modules* se zapíší následující knihovny:
cv.lib, cvaux.lib, cvcam.lib, cvhaartraining.lib, cxcore.lib, cxts.lib, highgui.lib ,ml.lib.

7.1.2 Implementace knihoven OpenCV do C++Builder

Knihovny OpenCV byly vytvořeny v prostředí Microsoft Visual C++. Binární struktury knihoven v prostředí Mocrosoft Visual C++ (Common Object File Format - COFF) a C++Builder (Object Module Format - OMF) nejsou shodné a tak je nutno tyto knihovny pro použití v C++ Builder převést do správného formátu. Převod je možný pomocí příkazu *coff2omf*. Takto je možné převést všechny hlavní knihovny, které budou v C++Builder využívány. V adresáři OpenCV\lib je založen adresář například *borland* a poté převedeny knihovny pomocí příkazu *coff2omf* z příkazového řádku. Příkaz pro převedení může vypadat následovně:

```
coff2omf cv.lib borland/cv.lib
coff2omf cvaux.lib borland/cvaux.lib
coff2omf cvcam.lib borland/cvcam.lib
coff2omf cxcore.lib borland/cxcore.lib
coff2omf cxts.lib borland/cxts.lib
coff2omf highgui.lib borland/highgui.lib
coff2omf ml.lib borland/ml.lib
```

Takto přeložené knihovny je už možno použít v projektu v programovém prostředí C++Builder. Takto přeložené knihovny je možno použít, ale některé funkce není možno spustit, jako jsou funkce cvGetSize a podobné, které předávají strukturu hodnotou a jejich velikost je větší než 4 byty. V literatuře [15] je uvedena možnost, jak je možné vytvořit vlastní knihovny pomocí *make* souboru, který se nachází v adresáři <cesta k OpenCV>\ _make.

Vytvoření knihoven je možné tak, že v <cesta k OpenCV>otherlibs\highgui\makefile.ms je nutno změnit vfw32 na msvfw32 a do příkazového řádku je zapsán příkaz *make -f make_all_bc.mak.* Pro verzi OpenCV 1.0 se však tyto knihovny nepodařilo implementovat do C++Builder a tak byl zvolen první příklad, kdy jsou nějaké funkce nefunkční, nicméně na vytvoření základní aplikace k ovládání kamer tyto funkce postačí.

Po založení aplikace clx v programovém prostředí C++ Builder je nutno v *Project -> Add to project* přidat knihovny ze souboru <cesta k OpenCV>\lib\borland, které jsme si převedli do formátu OMF, kromě knihovny *cvhaartraining.lib.* A dále nastavit cesty k hlavičkovým souborům a knihovnám v *Project ->Options ->Directories/Conditionals.* V projektu už pouze vložíme ty knihovny, které chceme používat.

7.1.3 Příklad použitých instrukcí

Zjištění počtu a názvy připojených kamer

Nejprve je nutné zjistit názvy kamer, které jsou k počítači připojeny. Jak již bylo zmíněno dříve, k tomuto účelu byla vytvořena knihovna cvcam, která je zahrnuta do OpenCV. Tato knihovna obsahuje základní funkce práci s kamerami a lze s ní pracovat i pod operačním systémem linux, kde se ovšem některé funkce liší a některé nelze využít. Příklad zjištění počtu, názvu kamer a zajištění inicializace knihoven je ukázáno na následujícím příkladě, kde ve *struktura_kamery* je potom možné zjistit název kamery pomocí *struktura_kamery*.DeviceDescription.

```
CameraDescription struktura_kamery;
int pocet_kamer = cvcamGetCameracount();
cvcamGetProperty(int cislo,CVCAM_DESCRIPTION,& struktura_kamery );
```

Vyvolání dialogu pro nastavení parametrů a formátu obrazu

Při vyvolání dialogového okna kamer je nutné tyto kamery nejprve inicializovat. Příklad inicializace a následného vyvolání dialogu nastavení je patrné z následujících příkazů:

```
CvcamSesProperty(int cislo,CVCAM_PROP_ENABLE,CVCAMTRUE);
cvCamInit;
CvcamGetProperty (int cislo,CVCAM_VIDEOFORMAT, NULL);
CvcamGetProperty (int cislo,CVCAM_CAMERAPROPS, NULL);
```

Spuštění kamery a zobrazení videa

Pro zobrazení videa a popřípadě získaného obrazu nám poslouží funkce, které jsou definované v knihovně *highgui*, která je opět součástí OpenCV. Příklad zobrazení videa z jedné kamery může vypadat následovně, kdy je nejprve vytvořeno okno a poté je k tomuto oknu přiřazena kamera. K tomuto obrazu je také navázána callback funkce, která je volána při každém získaném obrazu.

```
CvNamedWindow("Video",CV_WINDOW_AUTOSIZE);
HWND hnwd=(HWND) cvGetWindowHandle("Video");
CvcamSetProperty(int cislo,CVCAM_PROP_ENABLE,CVCAMTRUE);
CvcamSetProperty(int cislo,CVCAM_PROP_RENDER,CVCAMTRUE);
CvcamSetProperty(int cislo,CVCAM_PROP_WINDOW, &hnwd);
CvcamSetProperty(int cislo,CVCAM_PROP_CALBBACK, callback_kam);
cvcamInit();
cvcamStart();
cvWaitKey();
```

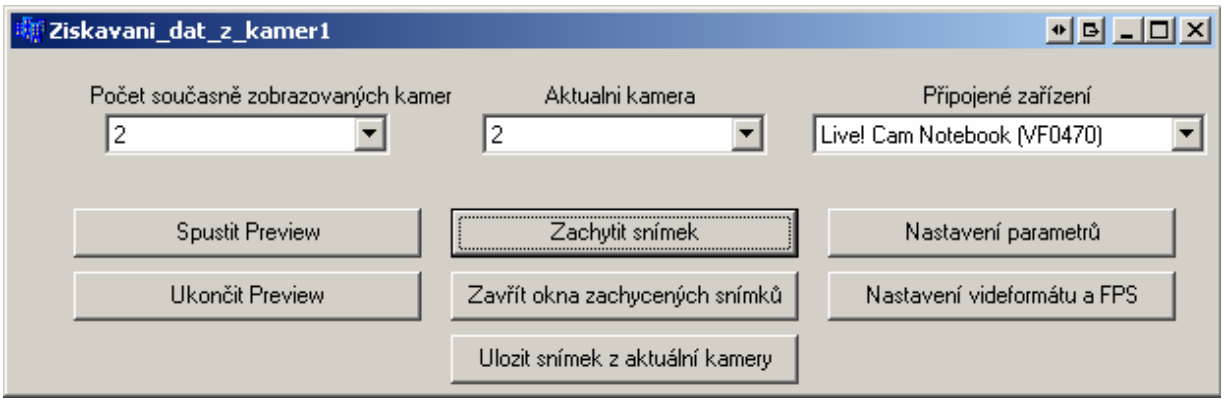
Vypnutí kamery a ukončení videa

```
cvDestroyAllWindows();
cvcamStop();
cvcamExit();
```

7.1.4 Realizace aplikace v programovém prostředí C++Builder

Při spuštěni aplikace a před zobrazením aplikace opět proběhne inicializace aplikace jako v prostředí MATLAB tak, že je zjištěn počet kamer, roletové menu počtu současně spuštěných kamer je nastaveno na jednu kameru a aktuální kamera je nastavena na první. Roletové menu připojených zařízení je nastaveno na první nalezenou kameru z nalezených kamer. Oproti programovému prostředí MATLAB je situace zjednodušena o nastavení parametru a formátu obrazu, které jsou nastaveny pomocí vyvolávaného dialogového okna, jak již bylo dříve zmíněno v přikladu použitých instrukcí.

Vzhled výsledné aplikace v programovém prostředí C++Builder je zobrazen na **Obrázek 7.1**. Uživatel si může vybrat počet současně zobrazených kamer a také má možnost pro aktuální kameru nastavit formát obrazu a parametry kamery. Po spuštění tlačítka *spustit Prewiew* se zobrazí nové okno s názvem vybrané kamery, ve kterém je zobrazováno video. Další funkcí, kterou má uživatel k dispozici je zachycení snímku, kdy se opět zachycený snímek zobrazí v nově vytvořeném okně a tento snímek má uživatel možnost uložit na disk pomocí tlačítka *Uložit snímek z aktuální kamery*. Tyto okna jsou zničeny buďto ovládacími prvky *Ukončit Preview* pro zobrazované video, nebo *zavřít okna zachycených snímků* pro všechny zachycené snímky. Při zavření aplikace se zavřou všechna otevřená okna. Při problémech se zobrazením videa z kamery je nutné odinstalovat program NERO, nebo pomocí programu RadLight Filter Manager 1.5, který je volně k stažení ze [17] a odregistrovat v *General -> DirectShow Filters -> Nero Video* <vše>.



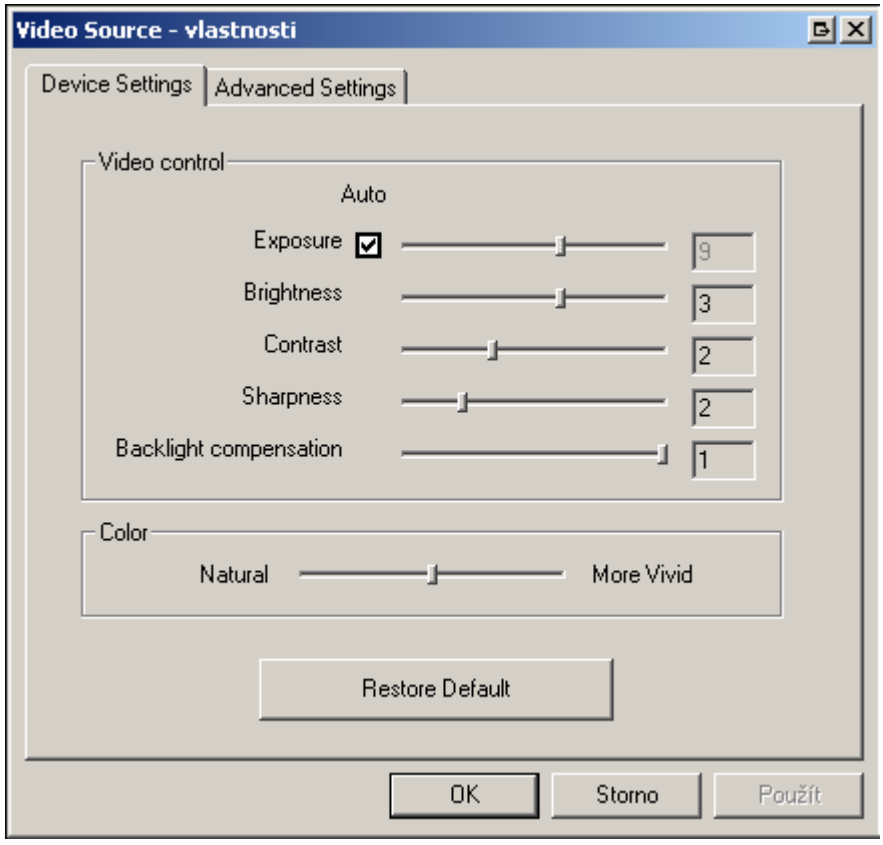
Obrázek 7.1: Výsledná aplikace vytvořená v C++Builder

Obrázek 7.2: Okna aplikace se zobrazením videa a zachycených snímků

Obrázek 7.3: Zachycené snímky ze dvou současně běžících kamer

Pro vytvoření samostatně spustitelné aplikace je nutné v prostředí C++Builder odškrtnout v *Project -> Options -> Linker Use dynamic RTL* a v *Project -> Options -> Packages Build with runtime packages*. K samospustitelné aplikaci je ještě nutno přidat dynamické knihovny OpenCV, které jsou zapotřebí. Pro správnou funkci je zapotřebí zkopírovat tuto aplikaci na disk a spustit soubor *register_all.bat.*

OpenCV se ukázalo být silným nástrojem při vytváření aplikace, která podporuje připojení jak průmyslových firewire kamer tak webových USB kamer. Nastavování parametrů zde nemuselo být nastavováno programově, ale pomocí dialogového okna, jak již byl ukázáno v příkladu použitých instrukcí. Dialogové okno má obdobný charakter jako v prostředí MATLAB, jak je patrné z **Obrázek 7.4.** Volba vývojového prostředí C++Builder je dobrá při vytváření aplikací, nicméně se zde vyskytly problémy s OpenCV kdy některé funkce nefungovaly.



Obrázek 7.4: Dialogové okno nastavení parametrů kamery

8 ZÁVĚR

V dnešní době čím dál tím více vzrůstá poptávka po počítačovém vidění, které ulehčuje, nebo zcela nahrazuje práci, kterou by musela vykonávat obsluha. Pro zpracování a vyhodnocení obrazu je tedy nezbytné, aby pořízený obraz byl co nejkvalitnější, jelikož s nekvalitním obrazem je další zpracování obtížné, nebo zcela nemožné. Pro tento účel je zapotřebí prostředí, které umožňuje ovládání a nastavení parametrů kamery.

Pro vytvoření aplikace bylo použito programového prostředí MATLAB a C++Builder s využitím knihoven OpenCV. Tato prostředí byla vytvořena na počítači s operačním systémem Microsoft Windows XP Professional a sestavou AMD Athlon XP 1700+, 640 MB RAM, kde byla otestována na dvou USB webových kamerách creative live! Cam notebook a firewire kameře DMK31BF03.

V aplikaci vytvořené v programovém prostředí MATLAB je možné najít kamery, které jsou k počítači připojeny a které mají nainstalované příslušné ovladače. Nastavit v nich parametry, které poskytují a získat jednotlivé snímky, sekvenci, nebo zobrazit video z více kamer zároveň. V této aplikaci je možné přehledně a intuitivně nastavovat parametry a sledovat jejich vliv na výsledný obraz. Tato aplikace ovšem není dostatečně rychlá, aby dokázala plně využít hardwarové zařízení, což je způsobeno hlavně počítačem, na kterém byla vytvořena a odzkoušena a univerzálností vývojového prostředí. Vytvořená aplikace je také převedena na samostatně spustitelnou aplikaci, kterou lze spustit i na počítači, na kterém není nainstalovaný MATLAB. Velikost této aplikace je 150 MB, což v dnešní době rychle se vyvíjející výpočetní techniky začíná být bezvýznamné.

Lepší variantou se jeví možnost využití knihoven OpenCV v programovém prostředí C++Builder nebo VisualC++, kde aplikace vytvořená v C++Builder za pomocí těchto knihoven je mnohonásobně rychlejší a je možné opravdu realtimové snímání a zpracování obrazu z více kamer současně. V této aplikaci je opět možné ovládat a nastavit parametry kamer, které jsou připojeny k počítači a které mají nainstalované příslušné ovladače.

9 POUŽITÁ LITERATURA

[1] HORÁK,K.- KALOVÁ,I. - PETYOVSKÝ, P. - RICHTER, M. : Počítačové vidění. Skriptum. VUT v Brně, 2007

[2] Camera link – Specification of the Camera Link interface standart for digital cameras and frame grabbers [online]. Dostupné z: <http://www.imagelabs.com/pdf/CameraLink5.pdf>

[3] Wikipedie- Otevřená encyklopedie [online]. Dostupné z: <http://cs.wikipedie.org>

[4] Wikipedia- The free encyklopedia [online]. Dostupné z: <http://en.wikipedia.org>

[5] Učební texty k předmětu MAPV-Hardware pro pořízení a zpracování obrazu. Skriptum. VUT v Brně. 2008

[6] The MathWorks – MATLAB and Simulink for technical computing [online]. The MathWorks Inc., 2008. Dostupné z: <http://www.mathworks.com>

[7] LANDRÉ, J.: Programming with INTEL IPP and OpenCV under GNU Linux – A begginer's tutorial, Elektronické skriptum, 2003.

[8] TROY,L.:INTEL open source computer vision library version 4.0-Beta installation and getting started guide for Windows, Elektronické skriptum

[9] INTEL Integrated performance primitives for INTEL architecture-Reference manual-Image and video processing, Elektronické skriptum, 2003

[10] Open source computer vision library – reference manual, Elektronické skriptum, 2001

[11] Imaging source – Technology based on standarts [online]. Dostupné z: < http://www.theimagingsource.com>

[12] Fotografování.cz [online]. Dostupné z: < http://www.fotografovani.cz>

[13] 1394 Trade Association - IIDC Based Digital Camera Speciffication, version 1.30 [online]. Dostupné z <http://www.cs.unc.edu/Research/stc/FAQs/1394Firewire/DCAM_Spec_V1_30.pdf>

[14] GenICam^{Tm} standard [online]. Dostupné z

[15] OpenCV Wiki [online]. Dostupné z <http://opencv.willowgarage.com/wiki/C%2B%2BBuilder>

[16] cvcam reference manual [online]. Dostupné z <http://www.cognotics.com/opencv/docs/1.0/cvcam.pdf>

[16] Filter manager [online]. Dostupné z <http://www.free-codecs.com/download/RadLight_Filter_Manager.htm>

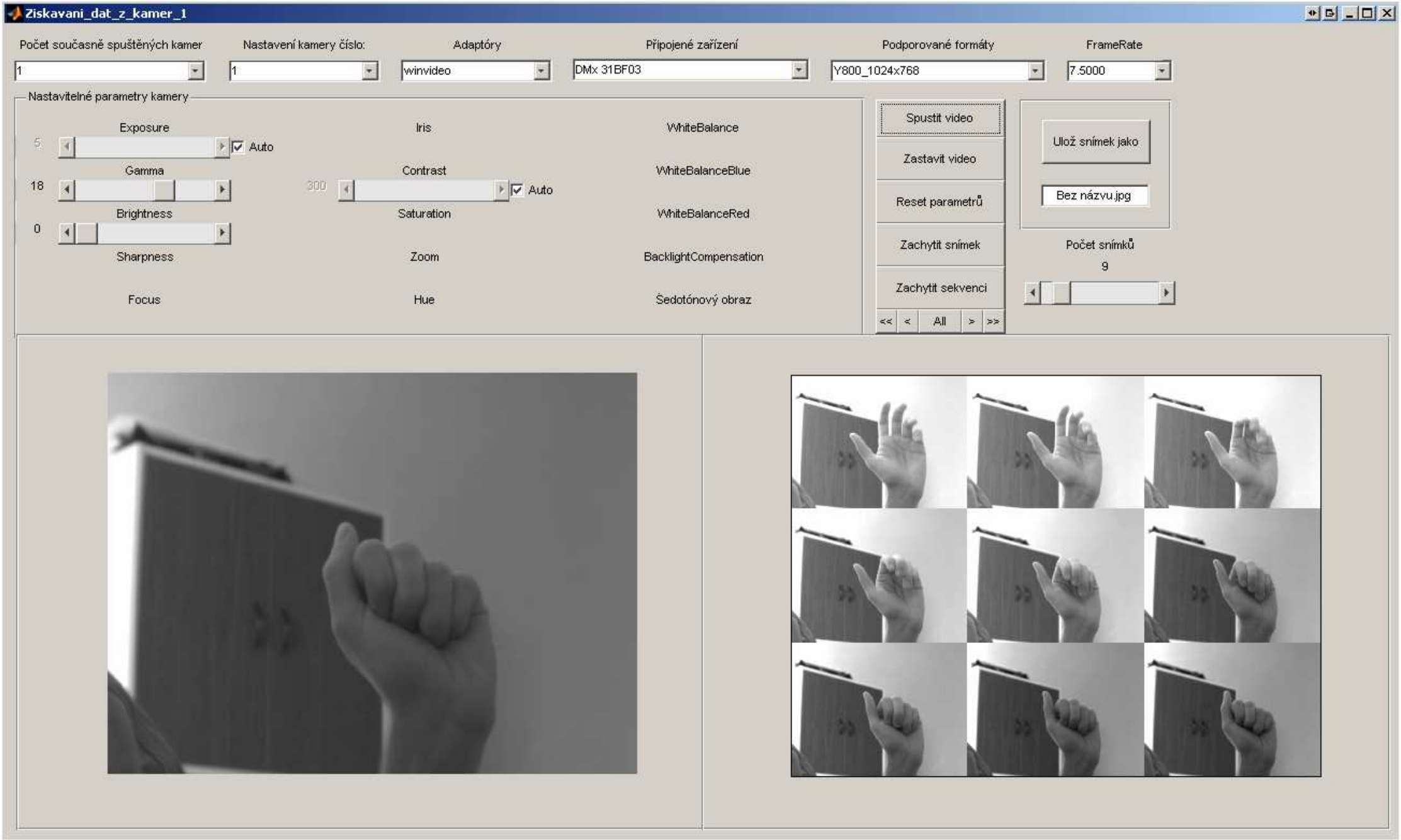
10 SEZNAM PŘÍLOH

[1] Vzhled aplikace se dvěma kamerami v programovém prostředí MATLAB

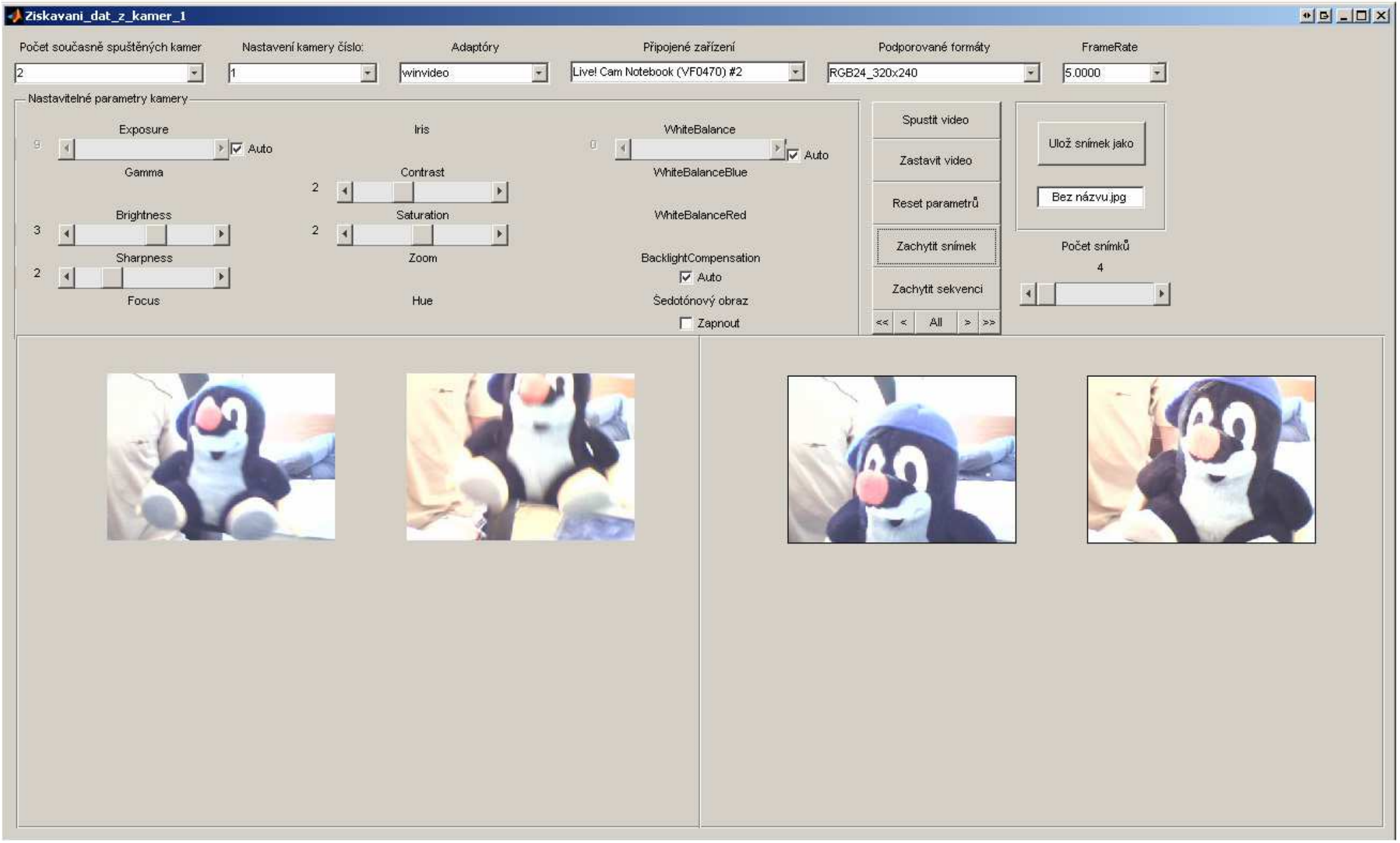
[2] Vzhled aplikace s jednou kamerou v programovém prostředí MATLAB

11 PŘÍLOHY

Příloha č.1


Ziskavani_dat_z_kamer_1
Počet současně spuštěných kamer
1
Nastavení kamery číslo:
1
Adaptóry
winvideo
Připojené zařízení
DMx 31BF03
Podporované formáty
Y800_1024x768
FrameRate
7.5000
Nastavitelné parametry kamery
Exposure
5
Auto
Iris
WhiteBalance
Gamma
18
Contrast
300
Auto
WhiteBalanceBlue
Brightness
0
Saturation
WhiteBalanceRed
Sharpness
Zoom
BacklightCompensation
Focus
Hue
Šedotónový obraz
Spustit video
Zastavit video
Reset parametrů
Zachytit snímek
Zachytit sekvenci
<<
<
All
>
>>
Ulož snímek jako
Bez názvu.jpg
Počet snímků
9

Příloha č.2


Ziskavani_dat_z_kamer_1
Počet současně spuštěných kamer
2
Nastavení kamery číslo:
1
Adaptóry
winvideo
Připojené zařízení
Live! Cam Notebook (VF0470) #2
Podporované formáty
RGB24_320x240
FrameRate
5.0000
Nastavitelné parametry kamery
Exposure
Auto
Iris
WhiteBalance
Auto
Gamma
Contrast
2
WhiteBalanceBlue
Brightness
3
Saturation
2
WhiteBalanceRed
Sharpness
2
Zoom
BacklightCompensation
Auto
Focus
Hue
Šedotónový obraz
Zapnout
Spustit video
Zastavit video
Reset parametrů
Zachytit snímek
Zachytit sekvenci
<< < All > >>
Ulož snímek jako
Bez názvu.jpg
Počet snímků
4