

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ
KATEDRA ŘÍDICÍ TECHNIKY



DIPLOMOVÁ PRÁCE

Orientace v prostoru



Praha, 2008

Autor: Bc. Ondřej Ton

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v příloženém seznamu.

V Praze dne 14. 1. 2009

A handwritten signature in dark ink, appearing to be 'Tou' or similar, written above a horizontal line.

podpis

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu diplomové práce Doc. Ing. Jiřímu Bayerovi, CSc. za vstřícnost a cenné rady, které mi v této práci pomohly. Dále bych rád poděkoval všem ostatním, kteří mi byli ochotni poradit a pomoci při tvorbě této diplomové práce.

V neposlední řadě bych chtěl poděkovat také celé své rodině a přátelům, kteří mi jsou oporou po celou dobu mého studia.

Abstrakt

Cílem této diplomové práce bylo navrhnout systém detekce překážek pro orientaci v prostoru, určený pro modely mobilních zařízení na Katedře řídicí techniky ČVUT. Jsou zde shrnuty základní metody pro měření hloubkové mapy scény, z nichž je vybrána metoda stereovidění. Tato metoda je dále rozebrána a aplikována na zvolený signálový DSP procesor Blackfin ADSP-BF61 na vybrané HW platformě vývojového kitu s rozšiřujícím modulem doplněným dvěma CMOS kamerami.

Algoritmizace jednoduchého a rychlého algoritmu stereovidění je provedena nejdříve v prostředí Matlab a později aplikována na samotný signálový procesor. Abychom mohli efektivně vyhledávat korespondence mezi stereoskopickými snímky okolního světa, pořízenými z CMOS kamer, je realizována rektifikace a kalibrace stereoskopického páru snímků. V závěru práce jsou shrnuty výsledky realizované metody a návrh možných vylepšení.

Abstract

The aim of this diploma thesis was to design an obstacle detection system for space navigation focused on the mobile devices models developed at the Department of Control Engineering CTU. It summarized the basic methods of measuring the depth map of the scene, of which chosen method is stereovision. This method has been analysed and applied to the selected signal DSP processor Blackfin ADSP-BF61 using a specified hardware platform of development kit complemented by an expansion module with two CMOS cameras.

Firstly, the algorithmization of a simple and fast stereovision algorithm was implemented in Matlab. Later on, it was applied to the chosen signal processor. An effective correspondence search between stereoscopic images of outside world acquired by 2 CMOS cameras is being carried out via rectification and calibration of a stereoscopic image pair. In conclusion, the work summarizes the results of implemented methods and possible improvements to the proposal.

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Ondřej Ton**

Studijní program: Elektrotechnika a informatika (magisterský), strukturovaný
Obor: Kybernetika a měření, blok KM1 - Řídicí technika

Název tématu: **Orientace v prostoru**

Pokyny pro vypracování:


1. Navrhněte strukturu systému pro orientaci mobilního zařízení (vzducholod', vznášedlo) v neznámém prostředí s překážkami.
2. Navrhněte způsob získání mapy prostředí (miniaturní kamera, laserové odměřování, PMD a pod.) a bezdrátový přenos získaných dat z mobilního zařízení do stacionárního PC.
3. Z rozboru datového toku mezi stacionárním a palubním počítačem a výpočetními možnostmi palubního počítače rozhodněte o distribuci činnosti obou z nich při získání a využití mapy prostředí tak, aby mobilní zařízení bylo do jisté míry ve své pohybové funkci autonomní.
4. Navrhněte a realizujte HW vloženého systému získání a využití mapy prostředí a jeho interface pro připojení na řídicí mikropočítač mobilního zařízení.
5. Navrhněte a realizujte programové vybavení PC pro zpracování mapy prostředí, její vizualizaci a následné sofistikované řízení pohybu mobilního zařízení.

Seznam odborné literatury:

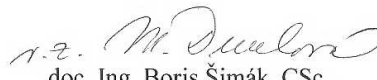
Dodá vedoucí práce

Vedoucí: doc. Ing. Jiří Bayer, CSc.

Platnost zadání: do konce zimního semestru 2008/2009


prof. Ing. Michael Šebek, DrSc.
vedoucí katedry




doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 10. 9. 2007

Obsah

Seznam obrázků	ix
Seznam tabulek	xii
1 Úvod	1
2 Měření polohy bodů v prostoru	3
2.1 Stereovidění	5
2.1.1 Úvod do reprezentace obrazu a 3D vidění	6
2.1.2 Kalibrace kamery	9
2.1.3 Epipolární geometrie 2 kamer	10
2.1.4 Výpočet vzdálenosti bodu v prostoru - kanonická konfigurace . . .	12
2.1.5 Geometrie dvou kamer - fundamentální matice	14
2.1.6 Problém okluze	16
2.2 Triangulační metoda využívající čárový laser a kameru	17
2.3 Získání hloubkové mapy ze snímků se známou ohniskovou vzdáleností . . .	19
2.4 PMD technologie a PMD kamery	20
2.4.1 PMD technologie	20
2.4.2 PMD kamery	22
2.4.2.1 PMD[vision]® 1k-S, 3k-S a 19k	23
2.4.2.2 SwissRanger™ SR3000 a SR4000	24
2.4.2.3 Canesta	26
3 Výběr HW a SW pro zpracování daných úloh	28
3.1 Výběr vhodného prostředku pro zpracování obrazových dat	28
3.2 Přehled DSP procesorů a vývojových kitů	30
3.3 Informace o Blackfin procesoru ADSP-BF561	32
3.4 Struktura signálového procesoru ADSP-BF561	33

3.4.1	Popis jádra procesoru ADSP-BF561	33
3.4.2	Architektura paměti procesoru ADSP-BF561	34
4	Vývojový kit pro ADSP-BF561	37
4.1	Vývojová deska DEV-BF5xxDA-Lite s integrovaným Debug Agentem . .	38
4.2	Modul procesorové jednotky CM-BF561	40
4.3	Rozšiřující modul EXT-BF5xx-CAM	43
4.4	CMOS kamera OV2640FSL	44
4.5	Vývojové prostředí Visual DSP++ 5.0	45
4.6	Nastavení a instalace vývojového kitu	46
4.6.1	Nastavení vývojové desky DEV-BF5xxDA-Lite	46
4.6.2	Nastavení rozšiřujícího modulu EXT-BF5xx-CAM	46
4.6.3	Ověření funkčnosti celého kitu a komunikace přes UART	47
4.6.4	Instalace ovladačů USB Debug Agentu	48
5	Algoritmizace metody stereovidění	50
5.1	Nastavení projektu ve Visual DSP++ 5.0	51
5.2	Nastavení procesoru	52
5.3	Nastavení rozhraní UART, kamer a paralelních sběrnic PPI	53
5.4	Kalibrace a rektifikace kamer - MATLAB Camera Calibration Toolbox .	55
5.4.1	Kalibrace kamer	55
5.4.2	Rektifikace snímků	57
5.5	Algoritmizace stereovidění	59
5.5.1	Zpracování snímků z kamer	59
5.5.2	Získání disparitní mapy	61
5.6	Výpočet hloubkové mapy	65
5.7	Optimalizace algoritmu stereovidění a výsledky	67
6	Závěr	71
	Literatura	72
A	Přílohy	I
B	Obsah přiloženého CD	XII

Seznam obrázků

1.1	Vývojový kit použitý pro aplikaci stereovidění.	2
2.1	Intenzitní obraz a hloubková/disparitní mapa	4
2.2	Lidský zrak - znázornění stereovidění.	5
2.3	Geometrie perspektivního zobrazení bodu ve 3D scéně.	6
2.4	Geometrie lineární perspektivní kamery.	7
2.5	Výpočet souřadnic bodu vzniklého projekcí.	9
2.6	Příklady zkreslení obrazu	10
2.7	Epipolární geometrie prostorového vidění.	11
2.8	Kanonická stereo konfigurace dvou kamer.	12
2.9	Základní geometrie stereovidění v kanonické formě.	13
2.10	Obecná geometrie dvou kamer.	14
2.11	Problém okluze	16
2.12	Znázornění triangulační metody využívající čárový laser a kameru.	17
2.13	Výpočet vzdálenosti z podobných trojúhelníků.	18
2.14	Obrázky s různou ostřicí vzdáleností a výsledná hloubková mapa	19
2.15	Princip metody měření TOF využívající fotodiodu na straně přijímače.	20
2.16	Struktura PMD pixelu.	21
2.17	Použití PMD prvku místo fotodiody na straně přijímače.	22
2.18	PMD kamery z nabídky PMDTechnologies GmbH.	23
2.19	Ukázka výstupu kamery PMDTechnologies GmbH	24
2.20	PMD kamery z nabídky Mesa Imaging AG	25
2.21	Ukázka z dodávaného GUI kamery SwissRanger™ SR3000	26
2.22	Development kit DP300 a DP200 od firmy Canesta, Inc.	26
3.1	Funkční blokový diagram součástí ADSP-BF561.	31
3.2	Struktura jádra signálového procesoru ADSP-BF561.	34
3.3	Architektura paměti ADSP-BF561.	35

4.1	Vývojová deska DEV-BF5xxDA-Lite.	38
4.2	Přehled součástí vývojové desky DEV-BF5xxDA-Lite.	39
4.3	Procesorová jednotka CM-BF561.	40
4.4	Blokový diagram procesorové jednotky CM-BF561.	41
4.5	Popis vývodů procesorové jednotky CM-BF561.	42
4.6	Rozšiřující kamerová deska EXT-BF5xx-CAM.	43
4.7	CMOS kamery OV2640FSL	44
4.8	Uživatelské prostředí Visual DSP++ 5.0	45
4.9	Obrazovka utility DEV-BF5xxDA-Lite Installer.	48
4.10	Visual DSP++ konfiguratör - nastavení platformy.	49
4.11	Visual DSP++ konfiguratör - výběr platformy.	49
5.1	Blokové schéma použité fronty zpráv pro komunikaci jader.	53
5.2	Ukázka hyperterminálové obrazovky komunikující s ADSP-BF561	54
5.3	Nasnímaný kalibrační vzor.	56
5.4	Vnější parametry kamer	57
5.5	Blokový diagram algoritmizace stereoidění	60
5.6	Grafické zobrazení kroků výpočtu disparitní mapy	63
5.7	Disparitní mapa pro známý dataset „tsukuba“.	64
5.8	Konvolučního jádro průměrovacího filtru	64
5.9	Disparitní a hloubková mapa	65
5.10	Hloubková mapa vykreslená ve 3D prostoru	66
5.11	Závislost vzdálenosti bodu v prostoru na disparitě	67
5.12	Princip zrychleného výpočtu hodnot průměrovacího filtru.	68
5.13	Struktura formátu dat typu <i>fract16</i>	70
A.1	Disparitní mapa pro dataset „Midd1“.	I
A.2	Disparitní mapa scény č. 1	I
A.3	Disparitní mapa scény č. 2	II
A.4	Disparitní mapa scény č. 3	II
A.5	Disparitní mapa scény č. 4	II
A.6	Disparitní mapa pro dataset „Wood1“.	II
A.7	Disparitní mapa pro dataset „Plastic“.	III
A.8	Disparitní mapa pro scénu „bílé“ zdi.	III
A.9	Disparitní mapa pro uměle vytvořené černé obrázky	III
A.10	Rozmístění a očíslování konektorů DEV-BF5xxDA-Lite	IV

A.11 Rozmístění a očíslování konektorů EXT-BF5xx-CAM	IV
A.12 BDTImark2000 TM - poměr cena/výkon	X
A.13 Kalibrační a rektifikační parametry kamer	XI

Seznam tabulek

2.1	Specifikace PMD kamer z nabídky PMDTechnologies GmbH.	24
2.2	Specifikace PMD kamer z nabídky Mesa Imaging AG.	25
2.3	Specifikace PMD kamer z nabídky Canesta, Inc.	27
3.1	Srovnávací tabulka DSP procesorů	30
4.1	Typická spotřeba procesorové jednotky CM-BF561.	41
5.1	Přehled rychlostí funkcí pro průměrovací filtr	69
5.2	Přehled rychlostí dalších použitých funkcí	70
A.1	Vývody konektoru X1 procesorové jednotky CM-BF561.	V
A.2	Vývody konektoru X1 procesorové jednotky CM-BF561.	VI
A.3	Vývody konektoru X2 procesorové jednotky CM-BF561.	VII
A.4	Vývody konektoru X2 procesorové jednotky CM-BF561.	VIII
A.5	Tabulka připojení kamery č. 1 a č. 2	IX
A.6	Funkce jednotlivých LED diod na vývojové desce DEV-BF5xxDA-Lite. .	X

Kapitola 1

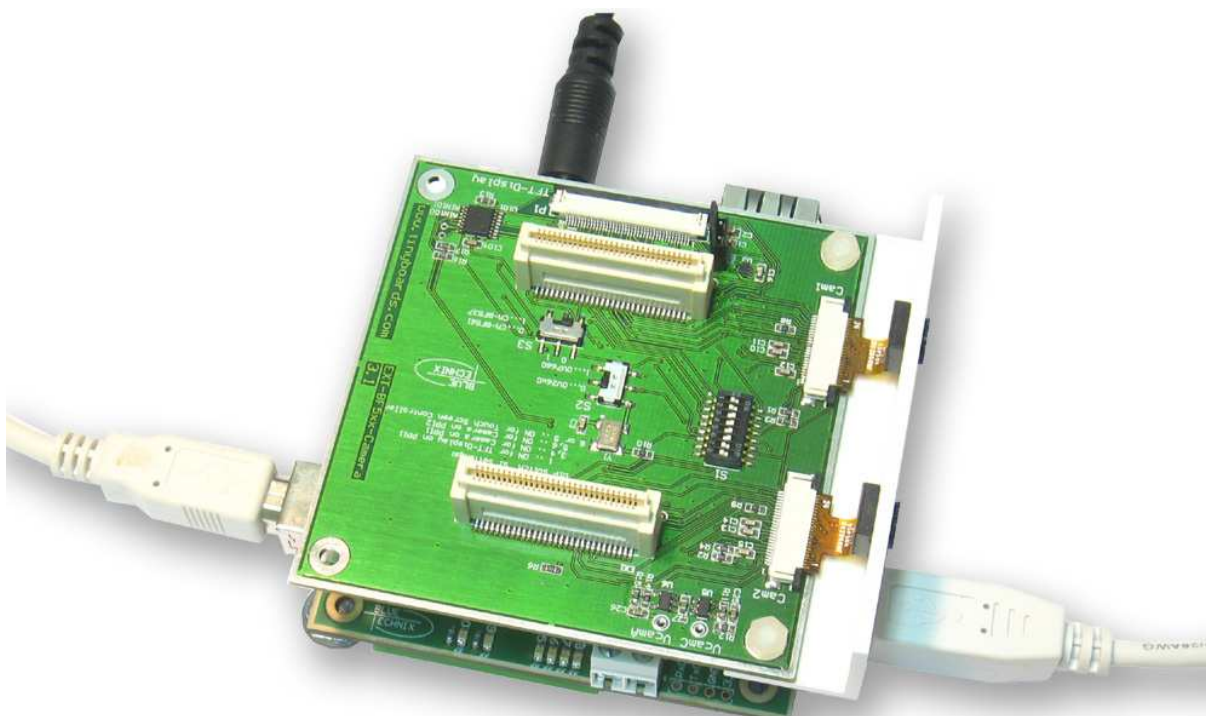
Úvod

Stejně tak jako člověk, tak i stroje se potřebují orientovat v okolním prostoru, vyhnout se překážce, určit vzdálenost, naplánovat směr pohybu, reagovat na vnější podmínky. Člověk disponuje svými smysly, především zrakem, které mu v této orientaci napomáhají.

Tato diplomová práce se zabývá návrhem systému získání hloubkové mapy (anglicky *depth map*) pro detekci překážek. Takto navržený systém by měl být dále využitelný pro orientaci v prostoru se zaměřením na modely umístěné na Katedře řídicí techniky ČVUT, např. model vzducholodi nebo model vznášedla, a to s ohledem na jejich vlastnosti a parametry [4]. V kapitole 2 je stručný výčet metod, které lze použít k získání hloubkové mapy. Z těchto metod je vybrána metoda stereovidění využívající vlastností dvou stereoskopických snímků a principu triangulace. Tato metoda je podrobněji rozebrána v kapitole 2.1 a aplikována pro navržený systém.

Pro požadavky zpracování úlohy stereovidění bylo nutné vybrat vhodný výpočetní prostředek zpracování obrazových dat. Byl vybrán dvoujádrový signálový procesor od firmy Analog Devices ADSP-BF561, popsáný v kapitole 3.1. Pro vývoj a implementaci navržených algoritmů na tomto signálovém procesoru byla zvolena univerzální vývojová deska DEV-BF5xxDA-Lite s integrovaným „Debug Agentem“ od firmy Bluetechnix. Tato vývojová deska je osazená procesorovou jednotkou CM-BF561, která obsahuje zmiňovaný dvoujádrový signálový procesor ADSP-BF561. Aby bylo možné zpracovat obrazy z okolní scény, je celý modulový vývojový kit na obrázku obr. 1.1 doplněn o rozšiřující kamerovou desku EXT-BF5xx-CAM se dvěma barevnými CMOS kamerami. Vlastnosti a parametry jednotlivých částí vývojového kitu jsou rozebrány v kapitole 4.

Jednoduchý algoritmus stereovidění pro výpočet hloubkové mapy byl nejdříve navržen v programovacím prostředí *Matlab verze 7.1* společnosti *MathWorks* a později implementován ve vybraném vývojovém prostředí pro daný signálový procesor.



Obrázek 1.1: Vývojový kit použitý pro aplikaci stereovidění.

V závěru práce jsou shrnuty a zhodnoceny vlastnosti, parametry a výsledky otestovaného algoritmu stereovidění použitého na HW platformě firmy Bluetechnix s dvoujádrovým signálovým procesorem ADSP-BF561.

Kapitola 2

Měření polohy bodů v prostoru

V této kapitole jsou popsány základní bezkontaktní metody měření vzdálenosti a získání hloubkové mapy (podrobnější popis je uveden v literatře [3]). Jsou zde shrnuty výhody a nevýhody použitelnosti konkrétních metod pro vlastnosti mobilního zařízení.

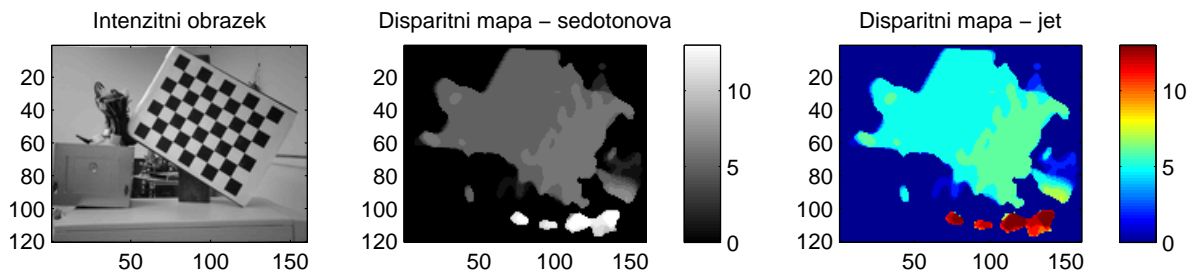
Měření vzdáleností můžeme provádět kontaktně nebo bezkontaktně. U bezkontaktních metod se měřicí aparatura nedostane do přímého styku s měřeným objektem. Mezi obecně známé bezkontaktní metody patří např. sonary využívající ultrazvuku a radary využívající mikrovlnných pásem. Ty pracují na principu měření zpoždění vyslaného vlnění, tj. dobu nutnou k překonání vzdálenosti od vysílače k překážce a zpět k přijímači. Z nekontaktních metod jsou pro tuto diplomovou práci důležité především metody operující ve spektru viditelného světla (a/nebo blízko viditelného spektra), které lze zpracovat pomocí kamer. Metody operující ve spektru viditelného světla (a/nebo blízko viditelného spektra) lze dále rozdělit na aktivní a pasivní.

Mezi aktivní metody patří např. promítání jistého vzoru (bodů, čar nebo složitějšího vzoru) do scény. Ze znalosti umístění zdroje promítaného světelného vzoru v prostoru, umístění snímacího prvku v prostoru (v našem případě je snímacím prvkem kamera) a jejich vzájemného natočení, lze triangulací vypočítat vzdálenosti těch bodů ve scéně, jenž jsou světelným vzorem pokryty. Takováto aktivní metoda, využívající čárový laser a kameru, je popsána v kapitole 2.2. Odlišnou aktivní metodou, jenž primárně nezpracovává obrazovou informaci, je metoda využívající PMD technologii (kapitola 2.4). Ta nahrazuje měření doby letu světelného paprsku měřením fázového posuvu mezi vyslaným a přijatým modulovaným světelným signálem.

Pasivní metodou je např. stereovidění, kdy využíváme pouze okolního světla. Zde hledáme korespondence obrazových bodů nebo jejich okolí mezi dvěma snímky, pořízenými z kamer se známým umístěním v prostoru (podrobný popis této metody je uveden v kapi-

tole 2.1). Další pasivní metodou je získání hloubkové mapy hledáním kontrastních oblastí ve snímku se známou ohniskovou vzdáleností (kapitola 2.3).

Pro přehled bych zde rád uvedl pojmy *pixel*, *voxel*, *hloubková mapa*, *disparitní mapa* a *2D intenzitní obraz*. Pixel (*picture element*) je jednotkový a dále již nedělitelný prvek intenzitního obrazu, jež vyjadřuje množství světelné energie přijaté ze scény kamerovým senzorem. Voxel (*volume pixel*) je odvozen od pixelu a značí jednotkový prvek, kterým namísto množství světelné energie u pixelu vyjádříme vzdálenost v prostoru. Poloha skupiny všech bodů v 3D prostoru scény je vyjádřena hloubkovou mapou, tj. maticí vzdáleností (voxelů), která souřadnicově odpovídá 2D intenzitnímu obrázku snímané scény. Stejný význam jako hloubková mapa má i pojem disparitní mapa s tím rozdílem, že disparitní mapa nevyjadřuje přímo vzdálenosti skupiny bodů v prostoru, ale vzdálenost odpovídajících si pixelů mezi dvěma stereoskopickými snímky. Vzdálenost je uvedena v pixelech a skutečná vzdálenost se pak přepočítá pomocí vzorce (5.6). 2D intenzitní obraz je vyjádřen maticí světelných intenzit (pixelů).



Obrázek 2.1: Intenzitní obraz je vlevo, vpravo je vyjádření disparitní mapy (šedotónová a „jet“ paleta).

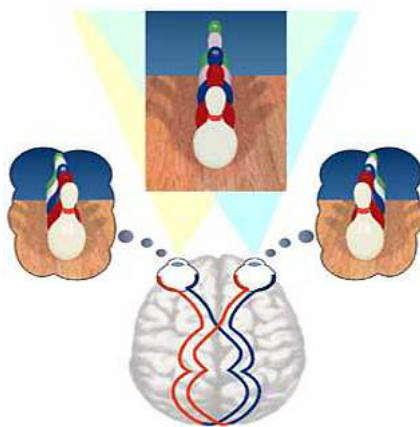
Příklad intenzitního vyjádření obrazu je vidět vlevo na obrázku obr. 2.1 a vyjádření disparitní mapy ve dvou variantách vpravo (šedotónová a „jet“ paleta). Černá barva v šedotónovém vyjádření disparitní mapy odpovídá nulové vzdálenosti korespondujících si bodů a bílá jejich maximální vzdálenosti. Podobně je tomu i u hloubkové mapy typu „jet“, pouze se změnila barevná paleta odpovídajících hodnot.

Podobné zobrazení je možné provést i pro hloubkovou mapu. Jednotkou bude místo disparity vzdálenost bodu v prostoru. Hodnoty v šedotónovém vyjádření hloubkové mapy odpovídají odstínům šedi - bílá je nejbližší pozorovateli, černá naopak nejdále. Samozřejmě je možné disparitní/hloubkovou mapu vyjádřit i v jiných barevných paletách.

2.1 Stereovidění

Metoda stereovidění je jednou z metod, jenž lze využít k získání vzdáleností bodů ve scéně nebo k vytvoření kompletní hloubkové mapy. Princip stereoskopických snímků mimo jiné využívá také lidský smysl - zrak. Složitost zpracování mozkiem je však mnohem složitější a stereovidění v lidském mozku je jen část komplexního zpracování, které je ještě v mnohém neprozkoumané.

Analogicky můžeme lidské oči připodobnit ke dvěma kamerám a mozek k procesoru, který tyto dva stereoskopické snímky, dvojrozměrné (2D) intenzitní obrazy zpracovává a triangulací vypočítá vzdálenost jednotlivých bodů. Obdobným způsobem k této úloze přistupuje i počítačové vidění.



Obrázek 2.2: Lidský zrak - znázornění stereovidění.

V kapitole 2.1.1 je popsán princip geometrické perspektivní projekce bodů z prostoru, jehož výsledkem je dvojrozměrný obraz získaný nelineárními vztahy (2.1) a (2.2). Opětovná geometrická rekonstrukce vzdálenosti promítnutého bodu z jednoho získaného intenzitního obrazu je nedostatečně podmíněná. Nabízejícím se řešením je získat dva či více snímků z různých míst, kde známe vzájemné rozmístění a natočení kamer v prostoru. Způsob takového řešení získání hloubkové mapy scény komplikují jednostranná nebo oboustranná okluze (kapitola 2.1.6), přítomnost šumu v získaných snímcích, různé radiometrické¹ vlastnosti scény v jednotlivých snímcích, problém hledání vzájemných korespondencí a v neposlední řadě časová/výpočetní náročnost hledání těchto korespondencí mezi snímky.

¹Radiometrie je část optiky, která se zabývá měřením elektromagnetického záření, včetně světla. Radiometrie se zabývá absolutními veličinami, zatímco fotometrie studuje obdobné veličiny, avšak z hlediska jejich působení na lidské oko.

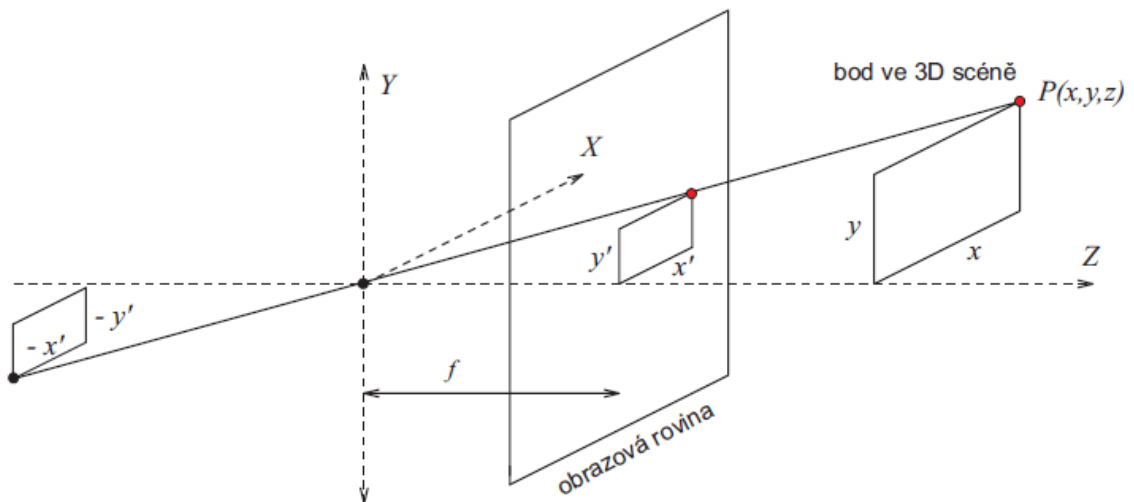
Princip získání hloubkové mapy metodou stereovidění lze v základním podání popsat pomocí těchto tří kroků:

1. Kalibrace obou kamer použitých pro stereovidění
2. Nalezení korespondencí bodů mezi levým a pravým snímkem
3. Výpočet třetího rozměru bodu v prostoru v závislosti na vzdálenosti odpovídajících si nalezených korespondencí v levém a pravém snímku

V následujícím textu je uvedena teorie potřebná pro navržení algoritmů uvedených v kapitole 5.

2.1.1 Úvod do reprezentace obrazu a 3D vidění

Reálný svět, který všichni dobře známe, jenž nás obklopuje a my jako lidé se v tomto světě pohybujeme a žijeme, je trojrozměrný (3D). Pojem trojrozměrný (tj. trojdimenzionální) znamená, že body okolního prostředí interpretujeme pomocí tří proměnných x , y , z a tyto proměnné jsou souřadnice reprezentující umístění bodu v prostoru. Pokud použijeme kartézskou soustavu souřadnic (tzn. soustavu souřadnic, u které jsou souřadné osy vzájemně kolmé, protínají se v počátku soustavy souřadnic a souřadnice polohy bodu je možno získat jako kolmé průměty polohy k jednotlivým osám), můžeme graficky znázornit umístění bodu v prostoru tak, jak je to vyobrazeno na obrázku obr. 2.3.



Obrázek 2.3: Geometrie perspektivního zobrazení bodu ve 3D scéně.

Kameru s tenkou čočkou lze dobře aproximovat modelem dírkové komory. Na obrázku obr. 2.4 je vidět geometrie takovéto kamery, kde se bod \mathbf{X} ze scény reálného světa promítá přes optický střed kamery na obrazovou rovinu do bodu \mathbf{U} . Vzdálenost f optického středu \mathbf{C} od obrazové roviny je známá jako ohnisková vzdálenost. Kamera provádí lineární transformaci 3D projektivního prostoru P^3 do 2D projektivního prostoru P^2 . Bod \mathbf{X} umístěný ve scéně vyjádříme v souřadnicovém systému kamery (\mathbf{X}_c) pomocí vztahu (2.3) s použitím translace \mathbf{t} a rotace R .

$$\mathbf{X}_c = \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = R(\mathbf{X}_w - \mathbf{t}) \quad (2.3)$$

Výpočet bodu \mathbf{U}_c vzniklého projekcí bodu \mathbf{X} do obrazové roviny je dán vztahem (2.4), definovaným podobnými trojúhelníky v obrázku obr. 2.5.

$$\mathbf{U}_c = \left(\frac{-fx_c}{z_c}, \frac{-fy_c}{z_c}, -f \right)^T \quad (2.4)$$

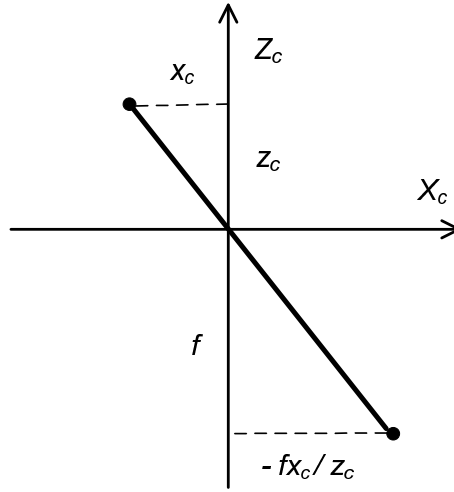
$$\mathbf{u} = \begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} a & b & -u_0 \\ 0 & c & -v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{-fx_c}{z_c} \\ \frac{-fy_c}{z_c} \\ 1 \end{pmatrix} = \begin{pmatrix} -fa & -fb & -u_0 \\ 0 & fc & -v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{-x_c}{z_c} \\ \frac{-y_c}{z_c} \\ 1 \end{pmatrix} \quad (2.5)$$

Bod promítnutý do obrazové roviny můžeme vyjádřit v homogenních³ souřadnicích jako $\tilde{\mathbf{u}} = (U, V, W)^T$ nebo ve 2D euklidovských souřadnicích $\mathbf{u} = (u, v)^T = (U/W, V/W)^T$. Při použití homogenních souřadnic můžeme rovnici (2.5) násobit nenulovou konstantou (z_c) a získáme

$$z_c \tilde{\mathbf{u}} = z_c \begin{pmatrix} -fa & -fb & -u_0 \\ 0 & fc & -v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{-x_c}{z_c} \\ \frac{-y_c}{z_c} \\ 1 \end{pmatrix} = \begin{pmatrix} -fa & -fb & -u_0 \\ 0 & fc & -v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -x_c \\ -y_c \\ 1 \end{pmatrix} = \quad (2.6)$$

$$= \begin{pmatrix} -fa & -fb & -u_0 \\ 0 & fc & -v_0 \\ 0 & 0 & 1 \end{pmatrix} R(\mathbf{X}_w - \mathbf{t}) = KR(\mathbf{X}_w - \mathbf{t}) \quad (2.7)$$

³více v literatuře [2]



Obrázek 2.5: Výpočet souřadnic bodu vzniklého projekcí.

Těmito operacemi jsme se dostali ke kalibrační matici K . Tato matice je horní trojúhelníková a její koeficienty se nazývají vnitřní parametry (anglicky *intrinsic parameters*) a jsou nezbytné pro udání vztahu mezi souřadnicemi v obrazové rovině a souřadným systémem kamery. Vnější parametry kamery (anglicky *extrinsic parameters*) definují polohu a orientaci souřadného systému kamery vzhledem ke známému obecnému souřadnému systému.

Podrobnější popis projektivní geometrie pro 3D vidění je uveden v literatuře [2].

2.1.2 Kalibrace kamery

Obraz z kamery je obvykle deformován zkreslením, které lidským okem nemusíme vůbec odhalit, ale pro použití v počítačovém zpracování obrazu je toto zkreslení nezanedbatelné, bývá zpravidla několik pixelů. Kalibrace kamery je postup, kde výstupem je kalibrační matice.

Pro aplikaci stereovidění je z hlediska rychlosti zpracování stereoskopických snímků tuto korekci nutné provést. Rovnice korekcí souřadnic (2.8) a (2.9) můžeme zapsat ve tvaru, kde u, v jsou korigované souřadnice, \tilde{u}, \tilde{v} jsou souřadnice v původním obraze a $\delta u, \delta v$ jsou korekce.

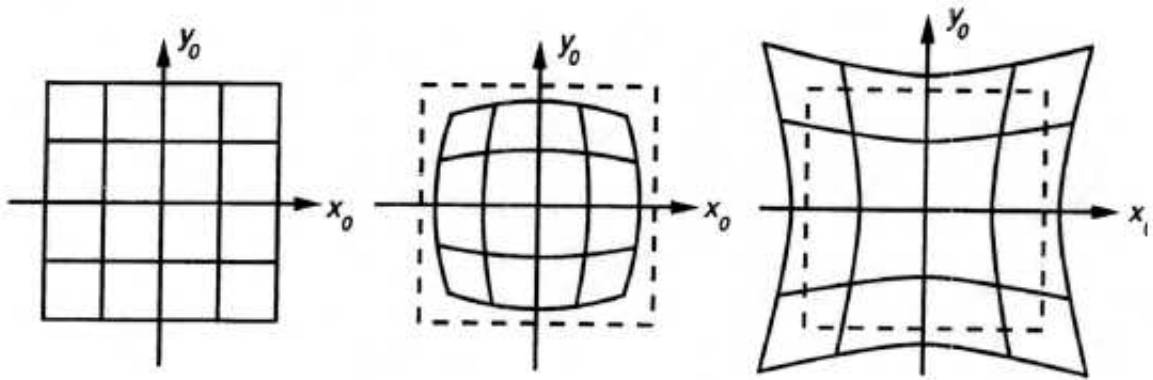
$$u = \tilde{u} + \delta u \quad (2.8)$$

$$v = \tilde{v} + \delta v \quad (2.9)$$

Typické bývá radiální zkreslení a zkreslení, kdy je hlavní bod umístěn mimo optickou osu. Informaci o běžných zkresleních výrobci v katalogu neuvádějí, a tak parametry zkreslení získáme kalibrací kamery. Radiální zkreslení je popsáno rovnicemi druhého řádu (2.10) a (2.11) a dáno optickou soustavou. Typickým představitelem je soudkovitý (anglicky *barrel*) nebo poduškovitý (anglicky *pincushion*) tvar zkreslení (obr. 2.6). Zkreslení může být samozřejmě složitějšího charakteru.

$$u = \tilde{u} [1 \pm \kappa_1 (\tilde{u}_2^2 + \tilde{v}_2^2)] \quad (2.10)$$

$$v = \tilde{v} [1 \pm \kappa_1 (\tilde{u}_2^2 + \tilde{v}_2^2)] \quad (2.11)$$



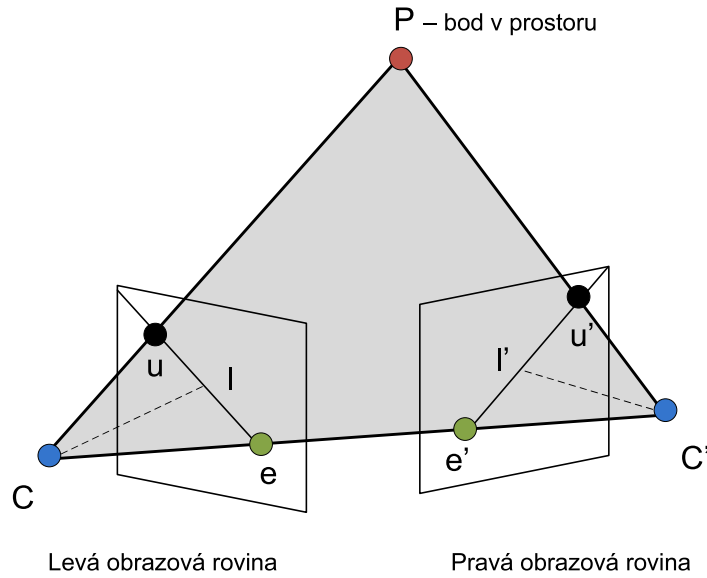
Obrázek 2.6: Příklady zkreslení obrazu - obraz bez zkreslení (vlevo), soudkovité (uprostřed), poduškovité (vpravo).

Aby hledání korespondencí mezi stereoskopickými snímky bylo co nejjednodušší a nejrychlejší, je potřeba převést epipolární geometrii snímků do kanonického tvaru, kde jsou epipolární linie paralelní a tím pádem je možné převést úlohu 2D hledání korespondencí na úlohu 1D prohledávání - hledání korespondencí v jednom řádku (viz. kapitola 2.1.3). Tím si ušetříme složitost algoritmizace a zrychlíme celý proces získání hloubkové mapy. Kalibrace dvojice kamer pro stereovidění je popsána v kapitole 5.4

2.1.3 Epipolární geometrie 2 kamer

V případě, že známe souřadný systém zkalibrované kamery, můžeme jednoznačně určit projektivní paprsek spojující snímaný bod s jeho promítnutím v obraze (senzoru kamery).

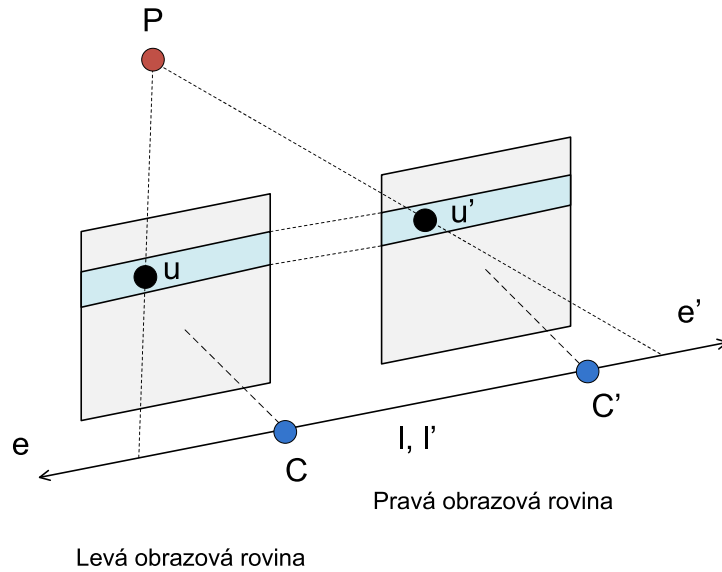
Stejnou úvahu provedeme i pro druhou kameru. Pokud tyto dvě kamery snímají stejný bod \mathbf{P} ve scéně, pak z průsečíku projektivních paprsků obou kamer určíme prostorové souřadnice pozorovaného bodu \mathbf{P} . Geometrie takového systému je zobrazena na obrázku obr. 2.7 a nazývá se epipolární geometrie.



Obrázek 2.7: Epipolární geometrie prostorového vidění.

Spojnice mezi body optických středů \mathbf{C} a \mathbf{C}' se nazývá základna (anglicky *baseline*). Základna, spojnice mezi optickým středem \mathbf{C} a bodem \mathbf{P} a spojnice mezi optickým středem \mathbf{C}' a bodem \mathbf{P} vymezují epipolární rovinu. Průsečnice epipolární roviny s levou, resp. pravou obrazovou rovinou se nazývá epipolární linie \mathbf{l} , resp. \mathbf{l}' . V případě, že se bod v prostoru pohybuje, pak všechny epipolární linie \mathbf{l} , resp. \mathbf{l}' vždy procházejí bodem \mathbf{e} , resp. \mathbf{e}' . Body \mathbf{e} a \mathbf{e}' se nazývají epipóly. Jsou tvořeny průsečíkem základny s levou a pravou obrazovou rovinou. Bod \mathbf{u} je projekcí bodu \mathbf{P} do levé obrazové roviny a bod \mathbf{u}' je projekcí bodu \mathbf{P} do pravé obrazové roviny. V případě, že bod bude v prostoru měnit pouze vzdálenost (zvětšuje se nebo zmenšuje z -tová souřadnice), pak se spojnice optického středu \mathbf{C} a bodu \mathbf{P} , reprezentující veškeré možné vzdálenosti bodu \mathbf{P} ve scéně, zároveň promítá na epipolární linii \mathbf{l}' pravé obrazové roviny. Z toho plyne, že korespondence bodu \mathbf{u}' z levé obrazové roviny bude vždy ležet na epipolární linii \mathbf{l}' náležící pravé obrazové rovině. Tento poznatek je pro návrh algoritmů velice důležitý. Poskytuje omezení hledání korespondence projekce \mathbf{u} bodu \mathbf{P} pouze na epipolární linii \mathbf{l}' . Jak již bylo řečeno, zjednodušuje problém vyhledávání korespondence z 2D na 1D prohledávání. Tuto úvahu lze přenést i na hledání korespondence bodu \mathbf{u}' z pravé obrazové roviny na epipolární linii \mathbf{l} ležící v levé obrazové rovině.

Dalším důležitým zjednodušením a možností zrychlit základní algoritmus stereovidění je uspořádání kamer do tzv. kanonické konfigurace (obrázek obr. 2.8). Princip spočívá v umístění kamer tak, aby si obrazové řádky senzorů kamer odpovídaly - byly zarovnaný horizontálně. Optické osy kamer jsou tímto uzpůsobením kamer paralelní, epipóly se přesunou do nekonečna a epipolární linie v obrazových rovinách se stanou také paralelní. Zarovnání kamer může být také horizontální, ale pro jednoduchost zpracování a principu snímání obrazu kamerou po řádcích, je lepší hledat korespondence v řádku.



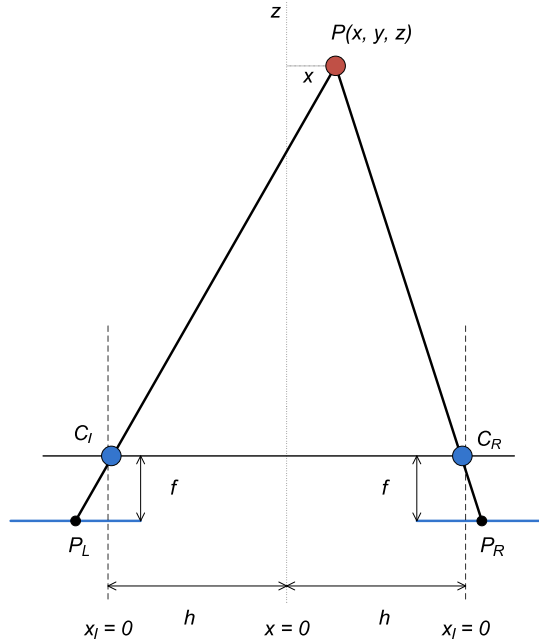
Obrázek 2.8: Kanonická stereo konfigurace dvou kamer.

Ačkoliv obě kamery umístíme a mechanicky nastavíme tak, abychom dosáhli co nejpřesnějšího možného horizontálního zarovnání, je nutné provést geometrickou transformaci zvanou rektifikace (anglicky *rectification*). Bližší informace o rektifikaci je možné najít v literatuře [2] nebo [10].

2.1.4 Výpočet vzdálenosti bodu v prostoru pro kanonickou konfiguraci kamer

V tomto textu je popsán princip výpočtu vzdálenosti bodu umístěného v prostoru z kanonického uspořádání kamer. Na obrázku obr. 2.9 je zobrazen pohled shora na dvě kamery s optickými středy ve vzdálenosti $b = 2h$. Na obrázku si dále můžeme všimnout bodu P , umístěného v prostoru na souřadnicích x, y, z . Bod P je projektivní transformací promítnut do levé obrazové roviny do bodu P_l a zároveň do bodu P_r v pravé

obrazové rovině. Osa z představuje vzdálenost bodu \mathbf{P} od kamer. Kamery jsou umístěny ve vzdálenosti $z = 0$, v počátku. Osa x vyjadřuje horizontální vzdálenost ze středu umístění mezi kamerami a navíc má každá ze dvou kamer svůj souřadný systém x_l a x_r se středy v $x_l = 0$ a $x_r = 0$. Rozdíl mezi hodnotami x_l a x_r , resp. $|P_l - P_r| > 0$ je nazýván disparitou (anglicky *disparity*). Spojnice bodů P_l , C_l a P , C_l jsou přepony podobných pravoúhlých trojúhelníků. Pomocí základní geometrie dokážeme odvodit vztah (2.15) pro výpočet vzdálenosti bodu \mathbf{P} od kamer.



Obrázek 2.9: Základní geometrie stereovidění v kanonické formě.

Použitím známého nelineárního vztahu (2.1) pro souřadnice na ose x , do něhož dosadíme pro levou obrazovou rovinu $x' = h + x$, získáme (2.12).

$$-P_l = \frac{(h + x)f}{z} \quad (2.12)$$

Opětovným použitím vztahu (2.1) dosadíme pro pravou obrazovou rovinu $x' = h - x$ a získáme (2.13).

$$P_r = \frac{(h - x)f}{z} \quad (2.13)$$

Ze vztahů (2.12) a (2.13) dostaneme eliminací proměnné x vztah (2.14) a upravíme do tvaru (2.15).

$$z(P_r - P_l) = 2hf \quad (2.14)$$

$$z = \frac{2hf}{P_r - P_l} \quad (2.15)$$

Rozdíl $P_r - P_l$ je výsledná hodnota disparity pozorovaného bodu \mathbf{P} a pokud je $P_r - P_l = 0$, pak značí, že vzdálenost bodu je teoreticky v nekonečnu. Prakticky nulová disparita značí mez rozlišení největší možné měřené vzdálenosti.

2.1.5 Geometrie dvou kamer - fundamentální matice

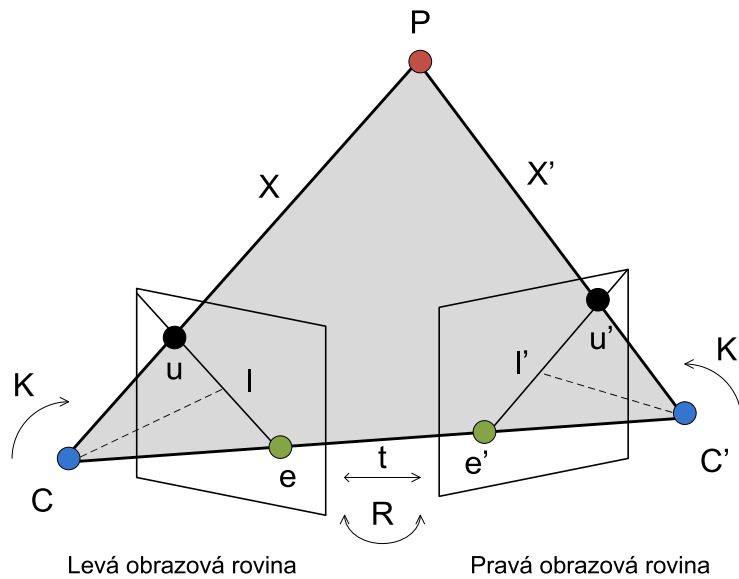
Algoritmy v kapitole 5 jsou navrženy pro kanonickou konfiguraci kamer a rektifikované snímky. Zde je stručně uveden matematický popis obecného modelu dvou kamer, které nejsou v kanonické konfiguraci. Více informací lze opět najít v literatuře [2].

Souřadný systém levé obrazové roviny může být transformován do pravé obrazové roviny z bodu \mathbf{C} optického středu levé kamery do optického středu pravé kamery \mathbf{C}' pomocí translace \mathbf{t} a matice rotace R . Souřadný systém levé kamery použijeme jako vztažný. Projekce bodu v prostoru do levé obrazové roviny je popsána rovnicí (2.16), kde K je kalibrační matice levé kamery.

$$u \cong K\mathbf{X} \quad (2.16)$$

Symbol \cong značí, že není známo měřítko. Transformace souřadného systému pravé obrazové roviny do souřadného systému levé obrazové roviny je provedena v rovnici (2.17) výrazem $(R\mathbf{X} - R\mathbf{t})$, kde K' je kalibrační matice pravé kamery.

$$u' \cong K'(R\mathbf{X} - R\mathbf{t}) = K'\mathbf{X}' \quad (2.17)$$



Obrázek 2.10: Obecná geometrie dvou kamer.

Vektory \mathbf{X} , \mathbf{X}' a \mathbf{t} jsou koplanární⁴. Vektor \mathbf{X}' vůči pravé kameře označíme jako $\mathbf{X}'_{\mathbf{R}}$ a vůči levé kameře jako $\mathbf{X}'_{\mathbf{L}}$. Rotace souřadných systému vyjádříme vztahy (2.18) a (2.19). Rovnici vyjadřující koplanaritu zapíšeme jako (2.20).

$$\mathbf{X}'_{\mathbf{R}} = R\mathbf{X}'_{\mathbf{L}} \quad (2.18)$$

$$\mathbf{X}'_{\mathbf{L}} = R^{-1}\mathbf{X}'_{\mathbf{R}} \quad (2.19)$$

$$\mathbf{X}'_{\mathbf{L}}^T(\mathbf{t} \times \mathbf{X}'_{\mathbf{L}}) = 0 \quad (2.20)$$

Rovnici (2.16) vyjádříme ve tvaru $\mathbf{X}_{\mathbf{L}} = K^{-1}\mathbf{u}$, rovnici (2.17) jako $\mathbf{X}'_{\mathbf{R}} = (K')^{-1}\mathbf{u}'$, do rovnice (2.19) dosadíme $\mathbf{X}'_{\mathbf{R}}$ a získáme vztah (2.21).

$$\mathbf{X}'_{\mathbf{L}} = R^{-1}(K')^{-1}\mathbf{u}' \quad (2.21)$$

Dosazením (2.21) do (2.20) získáme

$$(K^{-1}\mathbf{u})^T(\mathbf{t} \times R^{-1}(K')^{-1}\mathbf{u}') = 0 \quad (2.22)$$

Tato rovnice je vzhledem k \mathbf{t} homogenní, tudíž není určeno měřítko a absolutní měřítko nezískáme, pokud nebudeme vědět alespoň jednu vzdálenost dvou bodů ve scéně.

Pro zjednodušení nahradíme vektorový součin násobením matic (2.23). Z vektoru translace $\mathbf{t} = (t_x, t_y, t_z)^T$, za podmínky $\mathbf{t} \neq \mathbf{0}$, vytvoříme antisymetrickou matici $S(\mathbf{t})$.

$$S(\mathbf{t}) = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}$$

$$\mathbf{t} \times A = S(\mathbf{t})A \quad (2.23)$$

Rovnici (2.22) přepíšeme do tvaru (2.26) a střední část (2.25) je tzv. fundamentální matice F nesoucí informaci o geometrii kamer a jejich vzájemné poloze.

$$\mathbf{u}^T(K^{-1})^T S(\mathbf{t}) R^{-1}(K')^{-1}\mathbf{u}' = 0 \quad (2.24)$$

$$F = (K^{-1})^T S(\mathbf{t}) R^{-1}(K')^{-1} \quad (2.25)$$

⁴Leží v dané rovině.

Pro každé dva korespondující body $u = (u, v, 1)^T$ a $u' = (u', v', 1)^T$ zapsané v homogenních souřadnicích platí podmínka

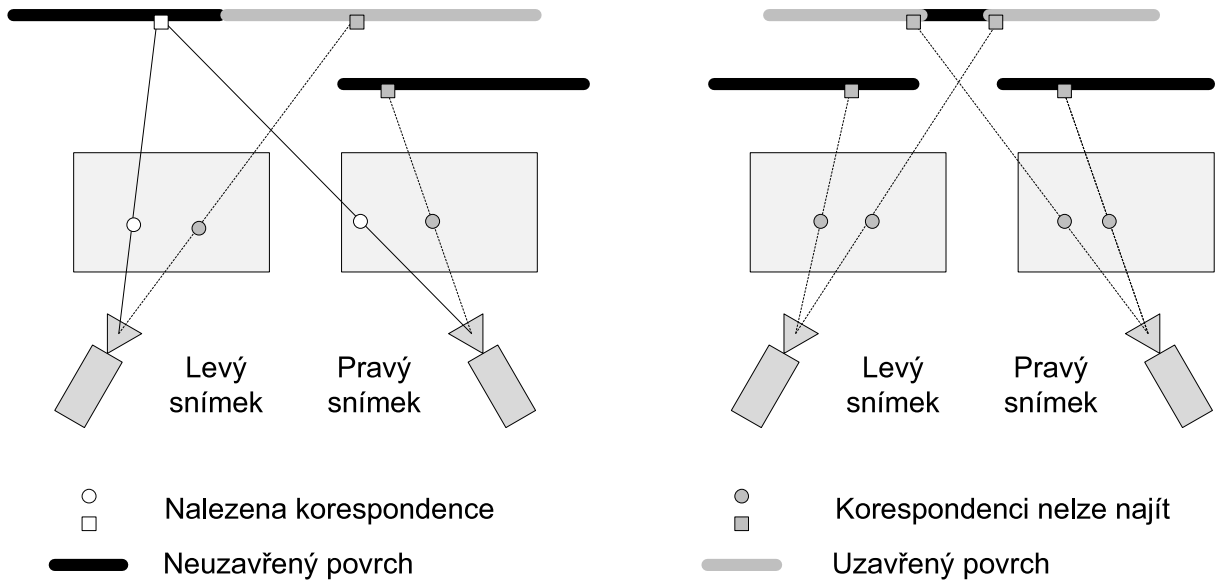
$$\mathbf{u}^T F \mathbf{u}' = 0. \quad (2.26)$$

Fundamentální matice F v sobě obsahuje veškeré informace, které mohou být odvozeny z dvojice snímků, pokud vyřešíme problém korespondence.

2.1.6 Problém okluze

Při hledání korespondencí mezi dvěma snímky lze narazit na problém jednostranné či oboustranné okluze (anglicky *occlusion*). Pokud nenastane žádná okluze je bod snímáný ve scéně na neuzavřeném povrchu vidět oběma kamerami. V případě jednostranné okluze je konkrétní bod na neuzavřeném povrchu vidět pouze jednou kamerou a druhá kamera vidí bod na uzavřeném povrchu. Jednostranná okluze je naznačena na obrázku obr. 2.11 vlevo. V případě oboustranné okluze jsou body viditelné oběma kamerami umístěné na uzavřeném povrchu. Oboustrannou okluzi ukazuje obrázek obr. 2.11 vpravo.

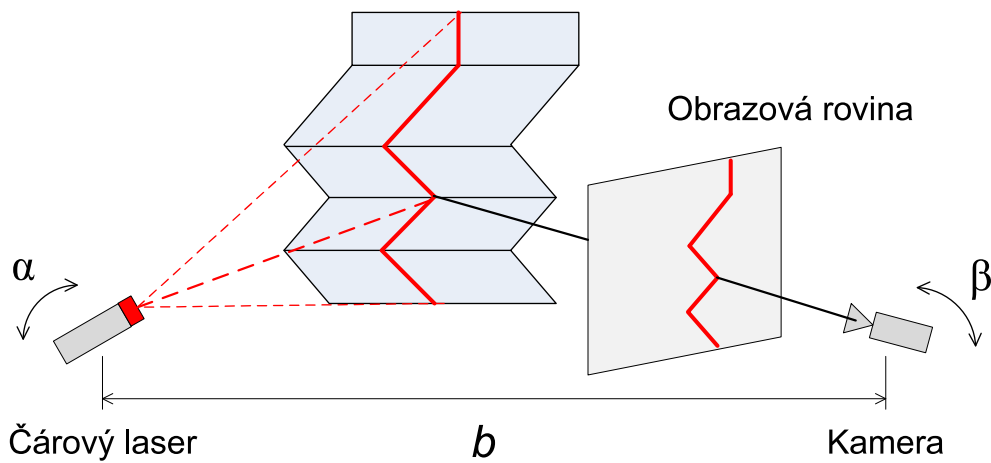
V reálných scénách není okluze příliš častým jevem, který by zásadně ovlivnil výslednou disparitní mapu. U algoritmů uvedených v kapitole 5 je použito filtrování korespondencí, které chybně nalezené korespondence z disparitní mapy vyřadí.



Obrázek 2.11: Jednostranná okluze vlevo, oboustranná okluze vpravo.

2.2 Triangulační metoda využívající čárový laser a kameru

V této kapitole je stručně popsána triangulační metoda využívající čárový laser jako aktivní zdroj světla a kameru jako senzor. Vzdálenost a vzájemné natočení kamery a čárového laseru je opět známé, stejně jako v případě stereovidění. Pokud použijeme bodové strukturované světlo, pak triangulaci můžeme vypočítat vzdálenost jediného bodu v prostoru. Strukturované světlo čárového laseru (úsečka - dále v textu obecně uváděna jako čára) umožňuje tzv. 2D triangulaci, tj. získání vzdáleností bodů v prostoru v dané rovině promítaného vzoru (čáry). Pokud budeme do scény promítat plošné strukturované světlo (v jednom okamžiku je osvětlena celá plocha scény⁵), pak lze mluvit i o případě stereovidění doplněném externí texturou. Jakmile budeme laserový paprsek tvaru bodu rozmítat v ose x a y , lze postupnou 1D triangulací získat kompletní hloubkovou mapu. Kompletní hloubkovou mapu získáme rovněž rozmítáním laserového paprsku tvaru čára, tentokrát v jedné ose a 2D triangulací.

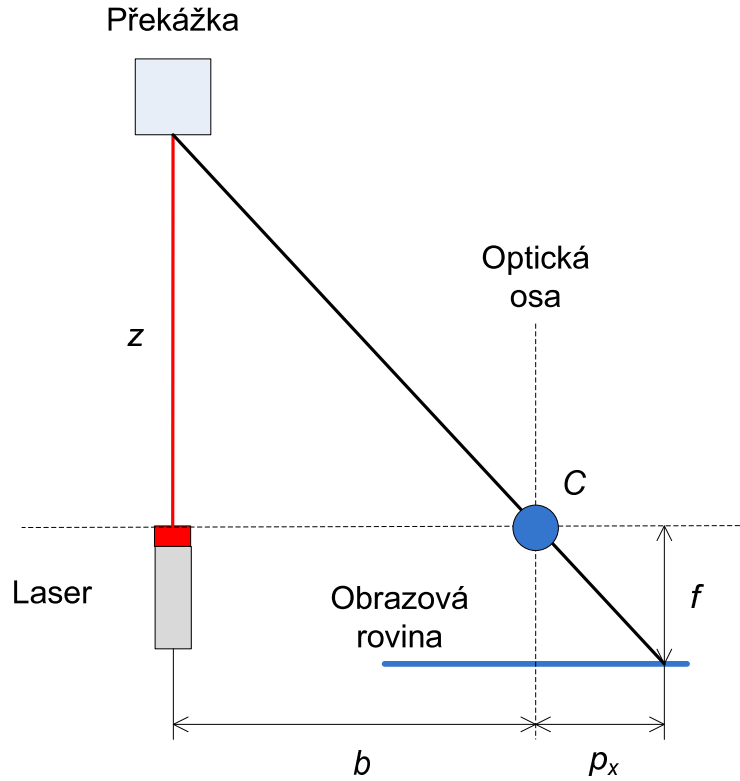


Obrázek 2.12: Znázornění triangulační metody využívající čárový laser a kameru.

Na obrázku obr. 2.12 je vidět případ použití čárového laseru a kamery. Kamera snímá okolní scénu, kam dopadá laserový paprsek ve tvaru čáry. Laserový paprsek vyváří ve scéně tvar, který se promítne na obrazovou rovinu. Hledáním maximální intenzity v řádku obrazu získáme souřadnici p_x . Soustava kamery, zdroje aktivního světla a osvětleného

⁵V tomto případě nemusí být osvětlený každý bod ve scéně, ale promítaný vzor musí vytvářet ve scéně vhodnou texturu.

bodu ve scéně na zkoumaném objektu tvoří triangulační trojúhelník (obr. 2.13). Vzdálenost světelného zdroje a kamery se nazývá triangulační báží b . Na straně zdroje je úhel svíraný s triangulační báží neměnný. Naopak na straně snímáče je úhel určen proměnnou pozicí 3D bodu v prostoru. Ze známých informací o trojúhelníku (úhly nebo velikosti stran) a souřadnicích promítnutého bodu na senzor kamery lze určit vzdálenost bodu v prostoru.



Obrázek 2.13: Výpočet vzdálenosti z podobných trojúhelníků.

V nejjednodušší konfiguraci kamery a laseru na obrázku obr. 2.13, lze vzdálenost překážky z určit pomocí vztahu (2.28) z rovnosti poměru odvěsen (2.27), kde b je vzdálenost kamery a laseru, f ohnisková vzdálenost a p_x je vzdálenost nalezeného pixelu od optické osy.

$$\frac{z}{b} = \frac{f}{p_x} \quad (2.27)$$

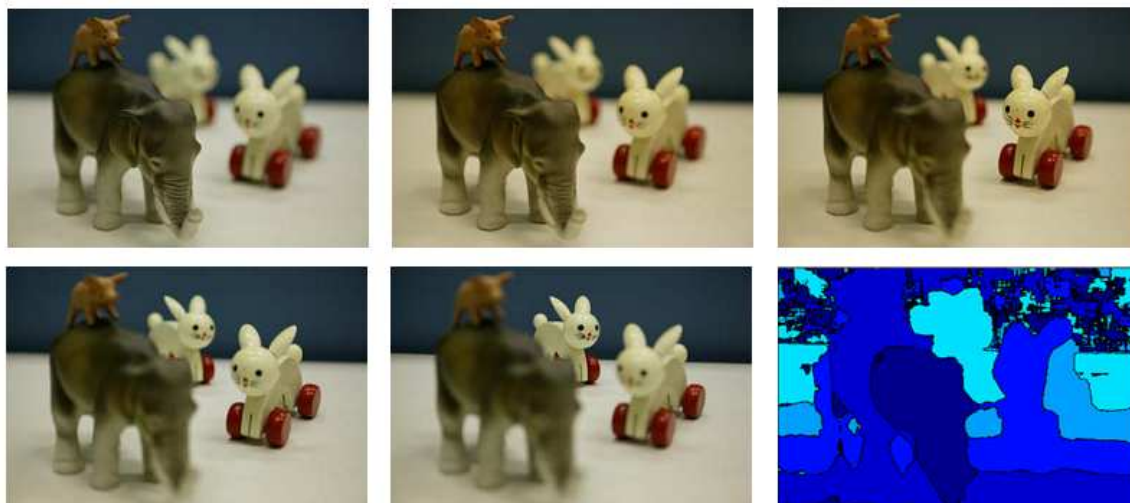
$$z = b \frac{f}{p_x} \quad (2.28)$$

2.3 Získání hloubkové mapy ze snímků se známou ohniskovou vzdáleností

Základním principem pro změření hloubkové mapy prostředí, při znalosti známých ostřících vzdáleností, je získání setu několika snímků, kdy je každý snímek pořízen v jiné (avšak známé) ostřící vzdálenosti. Pro nalezení kontrastní oblasti hledáme ve snímku vysoké frekvence (hrany). Vysoké frekvence detekujeme frekvenčně citlivými, směrově nezávislými operátory. Tímto operátorem je např. Laplaceův operátor. Tento operátor je nazýván Laplacián $\nabla^2 f(x, y)$ a tvoří ho suma druhých parciálních derivací.

Požadavkem pro úspěšné zpracování jsou dobré světelné podmínky a snímek musí obsahovat dostatek detailů pro určení kontrastu. Obdobně je tomu tak i v případě stereo-vidění.

Máme tedy soubor snímků z různých známých ostřících vzdáleností, kde každý snímek obsahuje ostré a rozostřené oblasti (viz. snímky 1-5 na obr. 2.14). Zaostřené oblasti odpovídají příslušným ostřícím vzdálenostem. Z celého souboru snímků tak lze poskládat hloubkovou mapu a zobrazit ji v příslušné paletě barev (obrázek obr. 2.14 vpravo dole).



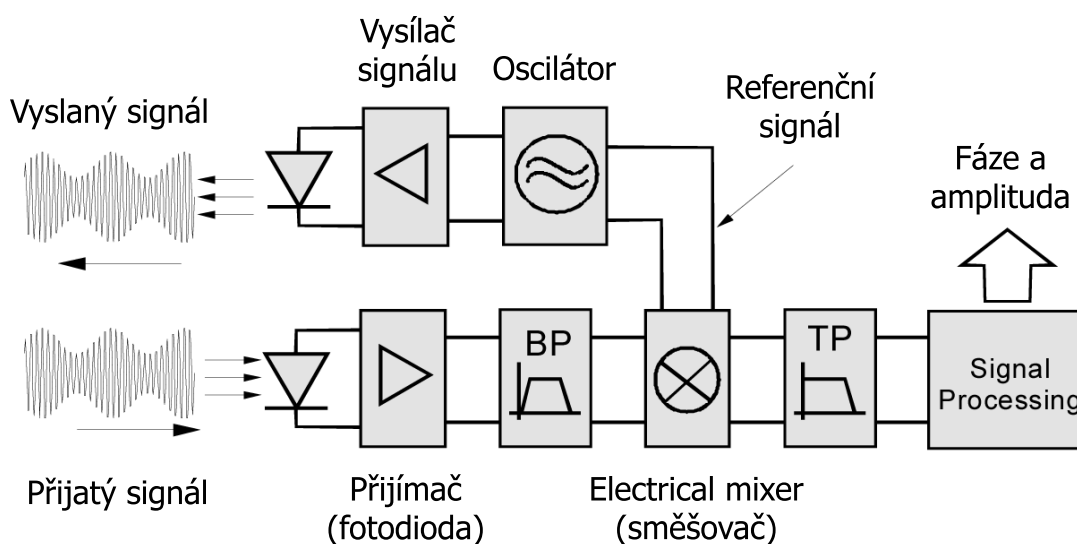
Obrázek 2.14: Obrázky s různou ostřící vzdáleností a výsledná hloubková mapa

2.4 PMD technologie a PMD kamery

Další možností, jak změřit vzdálenost ve scéně, je změřením doby letu světelného paprsku (Time Of Flight - TOF metody). Měří se doba průchodu paprsku od vysílače k přijímači, resp. doba za kterou se z vysílače dostane paprsek odrazem k přijímači. Rychlost světla je konstantní, známá a doba průchodu paprsku je tedy úměrná vzdálenosti. Změřené časy jsou příliš malé na to, aby se daly běžně měřit a z tohoto důvodu se měření doby letu nahrazuje měřením fázového posuvu mezi vyslaným a přijatým světelným signálem. Na tomto principu je založena PMD (Photonic Mixed Device) technologie, resp. PMD kamery.

2.4.1 PMD technologie

Pokud chceme měřit změnu fáze vyslaného a přijatého signálu, musíme nejdříve vyslaný signál namodulovat určitým referenčním signálem. Můžeme použít téměř libovolný zdroj světla (LED, laser) s libovolnou vlnovou délkou. Vysláním a zpětným přijetím signálu získáme posuv fáze úměrný době zpoždění t_d , kterou hledáme. Porovnání fází se provádí ve směšovači, tzv. 2D-EOM (Electro-Optical Mixer). Realizaci 2D-EOM lze provést použitím fotodiody nebo přímo PMD prvkem.

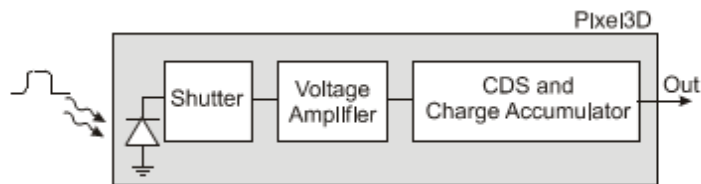


Obrázek 2.15: Princip metody měření TOF využívající fotodiodu na straně přijímače.

Realizace pomocí fotodiody používá fotodiodu jako převodník světla na elektrický signál a je vidět na obrázku obr. 2.15. Takto přijatý signál se dále frekvenčně omezuje pásmovým filtrem *BP*, kvůli omezení okolního rušení. Po odfiltrování nežádoucích složek se zjišťuje posuv fáze ve zmíněném směšovači (Electrical Mixer).

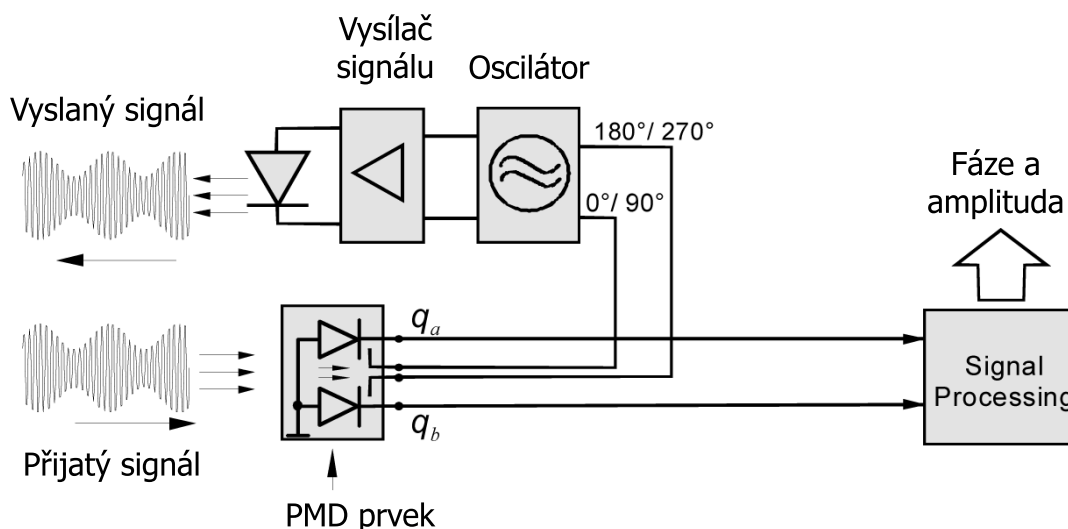
PMD (prvek) senzor provádí funkci směšovače již na úrovni dopadajícího světla. Převádí světlo na elektrický signál, ale i zároveň vyhodnocuje změnu fáze. Princip lze popsat následujícím způsobem. Na fotocitlivé elektrody se přivádí referenční modulační napěťový signál. Tímto signálem byl modulovaný i vyslaný světelný paprsek, přičemž na jednu elektrodu se přivádí signál fázově posunutý o 180° proti vedlejší elektrodě. Poté se postupnou střídavou změnou potenciálu na obou elektrodách se odvádí nahromaděný náboj z dopadajícího světla buďto na levou nebo na pravou sběrací elektrodu. Tam je náboj snímán čtecí elektronikou. Pokud je dopadající světelný paprsek ve fázi s referenčním signálem na elektrodách, tj. má stejný průběh, je náboj odváděn pravou sběrnou elektrodou. Pokud je světlo fázově posunuto o 180° je náboj odváděn na levou sběrnou elektrodu. V případě, že by byl fázově posunut o 90° proti referenčnímu signálu, je náboj stejnoměrně odváděn na levou i pravou sběrací elektrodu. Ze vzniklých proudů i_1 a i_2 z obou elektrod získáme jejich součtem $i_1 + i_2$ intenzitu dopadajícího záření a rozdílem $i_1 - i_2$ získáme fázový posuv mezi vyslaným a přijatým signálem.

Struktura PMD prvku (PMD pixelu) je znázorněn na obrázku obr. 2.16. Tímto způsobem na výstupu PMD senzoru dostaneme elektrický signál s informací o změně fáze a intenzitě dopadajícího záření.



Obrázek 2.16: Struktura PMD pixelu.

PMD senzor je založen na technologii CMOS a výrazně zjednodušuje konstrukci zařízení pro měření vzdáleností ve 3D prostoru scény. Použitím PMD senzoru odpadají všechny diskrétní části a prvky od příjmu světla až po obvody zpracování (*Signal Processing*). Realizace s PMD prvkem je na obrázku obr. 2.17.



Obrázek 2.17: Použití PMD prvku místo fotodiody na straně přijímače.

2.4.2 PMD kamery

PMD kamery jsou v této době velice diskutovaným tématem a v rámci např. automobilové bezpečnosti, se jimi intenzivně zabývají mnohé firmy. Současně je využitích těchto kamer vhodné komplexní řešení i pro robotické účely a tudíž také pro modely mobilních zařízení (vzducholodě, vznášedla, ...). Obecně tyto kamery dokáží zpracovat scénu v reálném čase (framerate⁶ je 30 fps⁷ a více) nebo jemu blízkém a to pro rozlišení v rozsahu od 64 x 16 až 176 x 144 pixelů. V některých případech umí kamery velice dobře potlačit vlivy okolního světla. Dokáží si poradit i se složitými podmínkami a vlastnostmi okolní scény, ale samozřejmě jsou závislé především na odrazivosti materiálu scény, členitosti a dalších vlastnostech. Parametry, které dále ovlivňují kvalitu a vlastnosti 3D obrazu, jsou modulační frekvence (výrobce bývá standardně nastavena na hodnotu, pro níž je kamera zkalibrována) a dále také doba integrace. Doba integrace je časová perioda, po kterou se akumulují přijaté fotony na jednom pixelu pro jeden měřící cyklus, ve kterém se určí posuv fáze, resp. vzdálenost. Pokud je tedy integrační doba příliš malá, pak amplituda přijatého signálu bude nízká pro správné určení vzdálenosti. Napoak při vysoké době integrace dojde k saturaci a měření je také nesprávné. Omezujícím parametrem PMD

⁶Framerate vyjadřuje počet zpracovaných snímků za vteřinu. Jednotkou je fps.

⁷„frame per second“ - počet snímků za vteřinu

kamer je rozsah měřených vzdáleností, jehož horní hranicí je tzv. opakovací vzdálenost⁸. Opakovací vzdálenost se odvíjí od modulační frekvence.

Z hlediska aplikace pro orientaci v prostředí pro mobilní zařízení je nevýhodou těchto kamer jejich vysoká cena a spotřeba elektrické energie, ačkoliv vývoj 3D kamer je velmi intenzivní a jak cena, tak spotřeba se zdatelně snižují a parametry se zlepšují. Dále uvádím základní přehled několika profesionálně vyráběných PMD kamer firem PMDTechnologies GmbH, Mesa Imaging AG a Canesta, Inc. Informace o produktech těchto firem lze najít v literatuře [5].

2.4.2.1 PMD[vision]® 1k-S, 3k-S a 19k

Jednou z firem zabývající se vývojem PMD technologie a výrobou 3D PMD kamer je německá PMDTechnologies GmbH. PMD kamery se dodávají jako kompletní set kamery i se světelným zdrojem.

V nabídce se v současné době objevují tyto modely PMD kamer: *PMD[vision]® 1k-S*, *PMD[vision]® 3k-S*, *PMD[vision]® 19k*, a nově *PMD[vision]® A-sample receiver* s dosahem až 40m. Všechny kamery obsahují *Suppression of Background Illumination (SBI)*, čili potlačení vlivu okolního osvětlení již na bázi samotných pixelů a zpracování zajišťuje 32-bitový procesor AMD Elan SC520 s operačním systémem eLinOS. Pro svoji funkčnost využívají vlnovou délku blízkou infračervenému světlu (NIR⁹). Na výstupu pak pro každý pixel získáme informaci o vzdálenosti a intenzitní obraz.



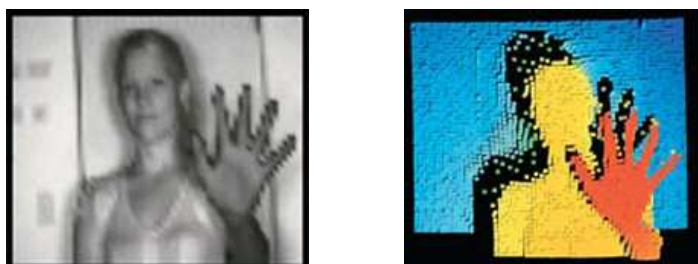
Obrázek 2.18: PMD kamery z nabídky PMDTechnologies GmbH.

⁸Vzdálenost, která je daná modulační frekvencí. Pokud je tato vzdálenost překročena a předmět je umístěn za maximální měřitelnou vzdáleností, pak již nelze jednoznačně určit, v jaké vzdálenosti se předmět nachází. Tj. puls, který byl vyslán v kroku n dorazí k přijímači až po vyslání $n + 1$ pulsu a nelze mezi nimi rozlišit.

⁹„near-infra-red“

Typ kamery (PMD[vision]®)	1k-S	3k-S	19k
Rozlišení	64 x 16 pixelů	64 x 48 pixelů	160 x 120 pixelů
Poměr stran	3:1	4:3	4:3
Modulační frekvence	standartně 20 MHz		
Opakovací vzdálenost	7,5m		
Rozlišení vzdálenosti	6mm		
Framerate	až 50 fps	až 25 fps	až 15 fps
Vlnová délka	870nm		
Optický výkon	3W		
ADC rozlišení	12 bitů		
Výstup	IEEE 1394a a Ethernet		
Váha	1,4kg		
Cena	na dotaz		

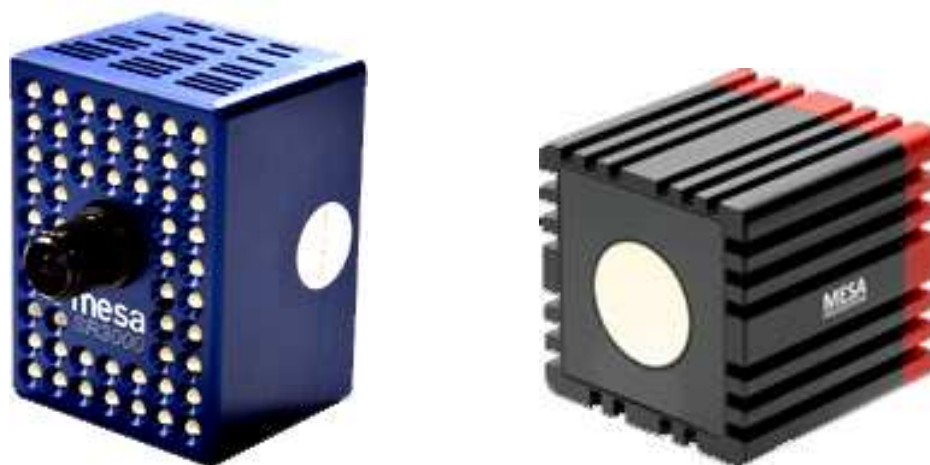
Tabulka 2.1: Specifikace PMD kamer z nabídky PMDTechnologies GmbH.



Obrázek 2.19: Ukázka výstupu - intenzitní obrázek (vlevo) a hloubková mapa (vpravo).

2.4.2.2 SwissRanger™ SR3000 a SR4000

Další z firem zabývajících se vývojem PMD technologií a prodejem PMD kamer je švýcarská firma Mesa Imaging AG. V současné době jsou k dispozici dva modely. Jsou to SwissRanger™ SR3000 a SR4000. Kamery v sobě mají zabudovaný světelný zdroj a v reálném čase poskytují data hloubkové mapy ve stejné rozlišení, tj. 176 x 144 pixelů. Pro svoji funkčnost využívají opět vlnovou délku blízkou infračervenému světlu a na výstupu pro každý pixel získáme také informaci o vzdálenosti a intenzitní obraz. Starší model SwissRanger™ SR3000 oproti jeho nástupci neobsahuje embedded DSP procesor a se svým světelným zdrojem s 55 LED diodami má přibližně dvojnásobnou spotřebu.

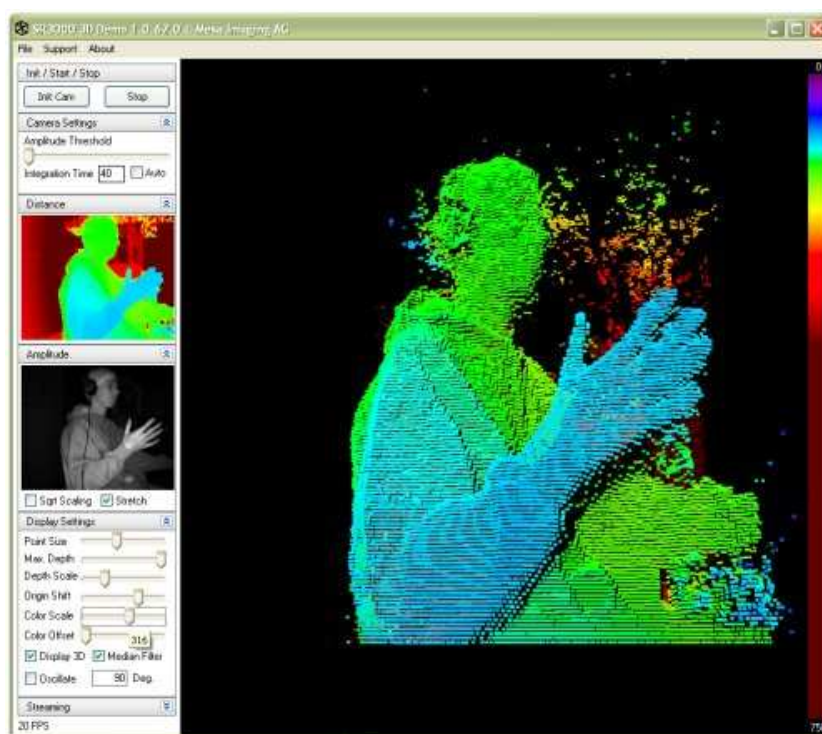


Obrázek 2.20: PMD kamery z nabídky Mesa Imaging AG SwissRanger™ SR3000 (vlevo) a SR4000 (vpravo).

Ve srovnání s produkty firmy PMD Technologies GmbH je řada SwissRanger™ určena spíše pro indoor aplikace (u SwissRanger™ SR4000 se pracuje na lepším potlačení vlivu okolního světla) a obecně mají nižší optický výkon a tím pádem i menší dosah ($\approx 5\text{m}$). Kamery používají kódovou modulaci, která dovoluje provoz několika kamer ve stejném prostoru. Kamery jsou dodávány se SW balíkem, který mimo jiné obsahuje také user interface pro *Matlab* a drivery pro různé OS.

Typ kamery (SwissRanger™)	SR3000	SR4000
Rozlišení	176 x 144 pixelů (QCIF)	
Poměr stran	4:3	
Modulační frekvence	standartně 20 MHz	standartně 30 Mhz
Opakovací vzdálenost	7,5m	5m (standartní nastavení)
Rozlišení vzdálenosti	1% z rozsahu	5mm
Framerate	až 25 fps	až 54 fps
Vlnová délka	850nm	
Optický výkon	1W	0,5W
Výstup	USB2.0	USB2.0, Ethernet
Spotřeba	typicky 12W	typicky 7W
Váha	-	0,47kg
Cena	nabídka na 5750 Euro	na dotaz

Tabulka 2.2: Specifikace PMD kamer z nabídky Mesa Imaging AG.



Obrázek 2.21: Ukázka z dodávaného GUI kamery SwissRanger™ SR3000

2.4.2.3 Canesta

Poslední zde uvedenou firmou zabývající se PMD technologiemi a výrobou 3D PMD kamer je americká firma Canesta, Inc. Tato firma nabízí vývojové kity PMD kamer DP2xx a nástupce DP3xx (obrázek obr. 2.22), včetně SW. Svoje firemní zájmy však nyní soustřeďuje především na průmyslové aplikace a k dispozici již nejsou volně ke stažení informace o jejich produktech. Proto je dále uvedena pouze specifikace starší řady vývojových kitů PMD kamer - DP2xx.



Obrázek 2.22: Development kit DP300 a DP200 od firmy Canesta, Inc.

Typ kamery	DP203	DP205	DP208
FOV	30°	55°	80°
Rozlišení	64 x 64 pixelů		
Poměr stran	1:1		
Modulační frekvence	13 - 104 MHz		
Opakovací vzdálenost	20 m	12 m	6 m
Rozlišení vzdálenosti	30 cm		
Framerate	až 30 fps		
Vlnová délka	785nm		
Optický výkon	0.1 - 1W		
Výstup	USB 1.1		
Spotřeba	10W		
Cena	na dotaz		

Tabulka 2.3: Specifikace PMD kamer z nabídky Canesta, Inc.

Kapitola 3

Výběr HW a SW pro zpracování daných úloh

V této kapitole jsou shrnuty informace o vybraném HW a programovém vybavení, použitém pro zpracování obrazových dat v úlohách získání disparitní a hloubkové mapy.

3.1 Výběr vhodného prostředku pro zpracování obrazových dat z digitálních kamer

Pro tuto diplomovou práci bylo v první řadě nutné zvolit vhodný prostředek pro zpracování obrazových informací z dvou digitálních kamer, ze kterých je možné vhodnou metodou extrahovat informace o vzdálenostech obrazových bodů ve scéně nebo přímo celou hloubkovou mapu scény. Tímto prostředkem by mohl být například osobní počítač nebo jeho mobilní modifikace, ale jeho architektura a rozměry, váha či spotřeba nevyhovují požadavkům pro modely mobilních zařízení, zmíněných v úvodní kapitole (1).

Dalšími vhodnými prostředky mohou být obvody typu:

- FPGA (*Field-programmable gate array*)
- ASIC/ASIP (*Application-specific integrated circuit/instruction-set processor*)
- CPLD (*Complex programmable logic device*)
- VHDL (*Very High Speed Integrated Circuit Hardware Description Language*)
- DSP (*Digital signal processor*) a další ...

Bohužel, většina těchto obvodů (FPGA, ASIC, ...) má malou flexibilitu a vyžaduje specifický, dosti časově náročný a složitý návrh. Zároveň jsou pro vývoj cenově nakladné. Pro bližší seznámení s těmito obvody je vhodné přečíst si článek [7] nebo příslušnou odbornou literaturu věnující se tomuto tématu.

Jako vhodný prostředek pro rychlý, energeticky nenáročný, cenově příznivý a flexibilní vývoj systému pro stereovidění (resp. řešení praktických úloh zpracování obrazů pro orientaci v prostředí a detekci překážek) byl vybrán obvod typu signálový procesor (DSP). Pro takovéto procesory jsou od výrobců většinou k dispozici vývojová prostředí a překladače, jenž umožňují vývoj SW jak v jazyku assembler, tak i v jazycích vyšších - např. jazyku C. Programování ve vyšším programovacím jazyku dovoluje rychlejší a flexibilnější návrh algoritmů řešení, naopak jazyk assembler umožňuje lepší a efektivnější využití možností DSP. Zároveň je od výrobce k dispozici optimalizovaná knihovna funkcí a ukázkových příkladů, jenž usnadňují programátorovi práci a dovolují efektivně využít tyto SW prostředky.

DSP signálové procesory lze dále přibližně dělit do následujících tří skupin:

1. low-cost fixed-point DSPs
2. high-performance fixed-point DSPs
3. floating-point DSPs

Low-cost fixed-point DSP signálové procesory jsou založeny na tradiční DSP architektuře z dřívějších let, pracují s pevnou řádovou čárkou, typicky na frekvencích nižších než 350 MHz, obvykle mají jednu MAC¹ jednotku a jejich cena je nízká.

Zcela jinou kategorií jsou floating-point DSP signálové procesory pracující s pohyblivou řádovou čárkou a mající komplexní obvodovou základnu, velmi vysoký výkon i vysokou přesnost při širokém dynamickém rozsahu. Jejich nevýhoda spočívá ve větší energetické náročnosti a vyšší ceně.

Poslední jmenovanou skupinou signálových procesorů jsou nové high-performance fixed-point DSP procesory. Většinou podporují několikanásobné vykonání instrukce během jednoho programového cyklu použitím techniky VLIW (*very long instruction words*) a také obsahují více MAC jednotek. Tyto procesory jsou kompromisem mezi dvěma výše zmíněnými skupinami. Zaručují vlastnosti flexibilního, rychlého a výkonného DSP s pevnou řádovou čárkou, který má zároveň nízké nároky na spotřebu, při zachování jednoduchosti programování a práce s ním. Nezanedbatelnou výhodou jsou i nízké náklady.

¹jednotka *multiply-accumulate*

Dělení DSP procesorů do těchto tří skupin se často překrývá a proto některé DSP procesory mají vlastnosti, které jsou typické pro různé skupiny.

3.2 Přehled DSP procesorů a vývojových kitů

Při výběru DSP procesoru a následně i vývojového kitu jsem se zaměřil na skupinu high-performance fixed-point DSP procesorů a pokusil najít vhodného kandidáta, který svými vlastnostmi nejlépe vyhovuje potřebám mobilních zařízení pro aplikace s rychlým zpracováním (multimediálních) dat. Kromě výkonu, spotřeby, rozměrů a váhy samotného DSP procesoru byly brány v potaz také dostupnost vhodného vývojového kitu, vývojového programového vybavení (včetně knihoven a příkladů) a možnost připojení CMOS kamer na vhodné periferie. Mezi vedoucí firmy nejenom na poli DSP procesorů lze jistě řadit:

- Texas Instrument (TI)
- Analog Devices (ADI)
- Freescale Semiconductor

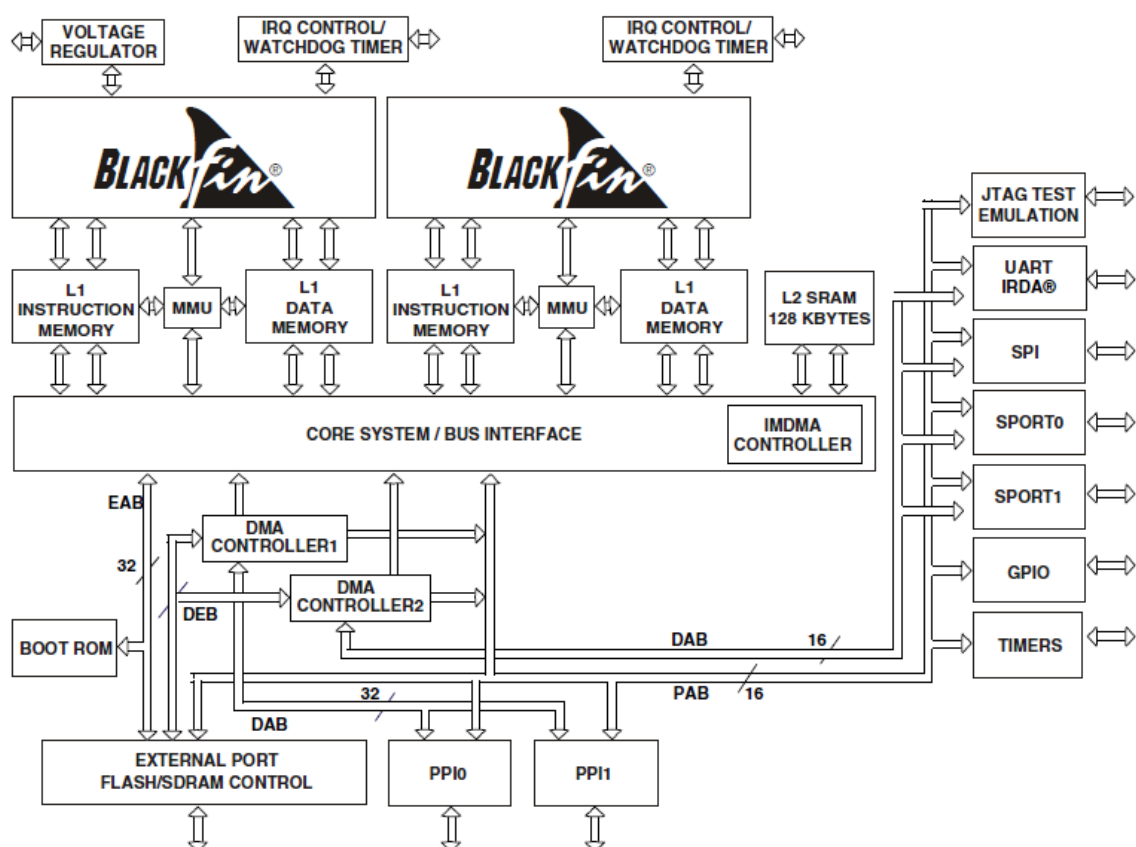
DSP procesor	BlackFin® ADSP-BF561	TMS320DM642-600™
Výrobce	Analog Devices (ADI)	Texas Instrument (TI)
Clock	600MHz (1 jádro)	600MHz
MMACs	max. 2400	max. 4800
L1 a L2	328 kB	288 kB
Přip. kamery	2 PPI (ITU-656 comp.)	3 videoporty (ITU-656 comp.)
Typ. spotřeba	1100 mW	1700mW
Cena (>10ks)	20 - 30 USD	40 - 60 USD
Package	297 ball PBGA	FCBGA 548-Pin

Tabulka 3.1: Srovnávací tabulka DSP procesorů BlackFin® ADSP-BF561 [21] a TMS320DM642-600™ [20].

Z nabídky těchto firem (rok 2007/2008) byl vždy vybrán jeden kandidát vhodného high-performance fixed-point DSP procesoru. Z produktů firmy Freescale byl vybrán DSP procesor i.MX21 (MC9328MX21), který však neměl možnost připojení obou kamer pro případ stereovidění. Mezi zbylé kandidáty patří DSP procesor BlackFin® ADSP-BF561

od Analog Devices a TMS320DM642-600TM od Texas Instrument. Podle výše uvedených kritérií byl zvolen dvoujádrový DSP procesor BlackFin® ADSP-BF561.

Na závěr je třeba říci, že všechny zde uvedené DSP procesory mají své pro i proti a velice si konkurují. Výběr DSP procesoru BlackFin® ADSP-BF561 zohledňuje především dostupnost vhodného kompaktního vývojového kitu (popsaného v kapitole 4) s rozšiřující kartou pro připojení kamer, velikost on-chip paměti a spotřeba procesoru. V současné době vývoj signálových procesorů pokračuje mílovými kroky vpřed a např. v nabídce firmy Texas Instrument je možné najít například DSP procesor TMS320C647xTM se třemi C64x+TM jádry, každé běžící na frekvenci 1 GHz a s až 24000 MMAC². Nabídka nových procesorů umožňuje při použití stejných algoritmů vyšší rychlost zpracování a nižší spotřebu.



Obrázek 3.1: Funkční blokový diagram součástí ADSP-BF561.

²Million Multiply ACcumulates per Second

3.3 Informace o Blackfin procesoru ADSP-BF561

Signálový procesor s označením ADSP-BF561 (nebo stručněji BF561) je z rodiny procesorů Blackfin, které jsou zaměřeny především na multimediální aplikace. Procesor nabízí dvě nezávislá jádra založená na architektuře Blackfin procesorů. Procesory této řady nabízejí vysoký výkon, velice nízkou spotřebu a jednoduchost použití a kódování. Právě tyto vlastnosti jsou klíčové pro aplikace výpočetně náročných obrazových operací pro mobilní zařízení, které je závislé na bateriovém napájení a tudíž na nízké spotřebě elektrické energie.

Architektura signálového procesoru BF561 kombinuje duální MAC jednotku, ortogonální³ RISC⁴ instrukční set s flexibilní single-instruction multiple-data (SIMD) možnostmi a také single-instruction-set architekturu některých multimediálních funkcí.

Signálové procesory řady Blackfin dále nabízejí možnost změny jak napětí jádra, tak nastavení frekvence jádra a tím optimalizaci spotřeby elektrické energie a možnost přizpůsobit aplikaci daným potřebám i v tomto ohledu.

ADSP-BF561 obsahuje následující periferie:

- Rychlé paralelní sběrnice - Parallel Peripheral Interfaces (PPI)
- Sériové porty - Serial Ports (SPORT)
- Rozhraní SPI - Serial Peripheral Interface
- Časovače - General-Purpose Timers, Watchdog Timers
- Rozhraní UART - Universal Asynchronous Receiver Transmitter
- Obecné vstupy/výstupy - General Purpose I/O (Programmable Flags)

Všechny výše uvedené periferie mimo general-purpose I/O a časovačů jsou podporovány flexibilní strukturou DMA⁵ řadičů (DMA1 a DMA2) a také řadiči vnitřní paměti (IMDMA). Každý z DMA1 a DMA2 má 12 programovatelných kanálů a dva oddělené DMA streamy určené pro přenos dat mezi pamětmi DSP a také externí SDRAM a asynchronní pamětí. Sběrnice jsou zároveň dostatečně dimenzované svojí propustností, aby dokázaly obsloužit veškerý provoz jak periférií, tak on-chip pamětí.

³Architektura procesoru schopná rozdělit při zpracování instrukci na více úkonů a ty (díky vhodnému návrhu jeho ALU nebo jiných integrovaných obvodů) provést v jednom strojovém cyklu.

⁴*Reduced Instruction Set Computer* - Počítač s redukovanou instrukční sadou.

⁵*Direct Memory Access* - umožňuje hardwarovému subsystému přímý přístup do operační paměti bez účasti procesoru.

3.4 Struktura signálového procesoru ADSP-BF561

Tato kapitola popisuje vnitřní strukturu signálového procesoru ADSP-BF561. Jsou zde popsány jádra procesoru a uspořádání paměťového systému.

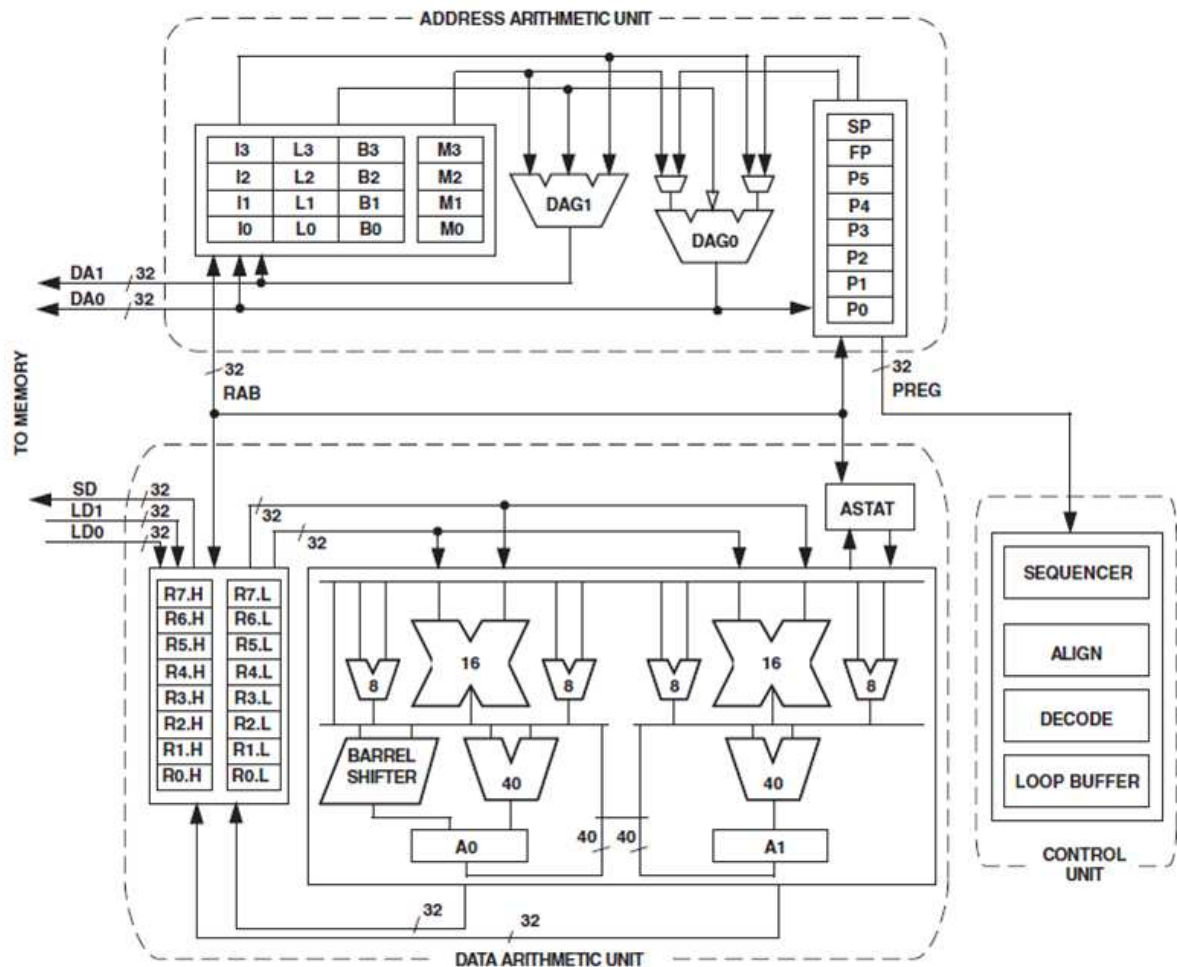
3.4.1 Popis jádra procesoru ADSP-BF561

ADSP-BF561 má dvě identická Blackfin jádra. Každé z těchto jader obsahuje dvě 16-bitové násobičky, dvě 40-bitové sčítačky, dvě 40-bitové aritmeticko-logické jednotky (ALU), čtyři 8-bitové video aritmeticko-logické jednotky a 40-bitový shifter (vykonávající posun, rotaci, normalizaci, vyjmutí). Každá MAC jednotka vykonává v jednom cyklu 16 x 16-bitové násobení s akumulací jako 40-bitový výsledek. Výpočetní jednotka pracuje s 8-bitovými, 16-bitovými a také 32-bitovými daty. Jádro procesoru obsahuje 8-vstupový 32-bitový registr pro výpočetní jednotky. ALU jednotky obsahují běžný set aritmetických a logických operací, které pracují s 16-bitovými a 32-bitovými daty. Pro některé úlohy procesor obsahuje speciální instrukce. Jedná se např. videoinstrukce bytového zarovnání, 8-bitové operace sčítání absolutních odchylek - SAA (*subtract-absolute value-accumulate*), 8-bitové operace průměrování nebo 16-bitové a 8-bitové sčítání. Pro některé instrukce mohou být dvě 16-bitové ALU operace vykonány současně na dvojici registrů. V případě, že použijeme obě ALU pak je možné provést čtyři 16-bitové operace - *quad* 16-bitová operace.

Programový sequencer řídí tok instrukcí, jejich zarovnání a dekodování. Řadič podporuje PC-relative a nepřímé podmíněné skoky (se statickou podmínkovou predikcí) a volání subrutin. Architektura procesoru je navržena proti vzniku uváznutí při pipeline zpracování⁶. Adresová aritmetická jednotka umožňuje adresování pro duální přístup k datům.

Instrukce řady Blackfin jsou optimalizovány pro 16-bitové operační kódy, reprezentující nejvíce používané instrukce. Komplexní DSP instrukce jsou kódovány do 32-bitových operačních kódů jako multifunkční instrukce. To dovoluje programátorovi vhodné využití zdrojů v jednom instrukčním cyklu. Architektura procesoru je optimalizovaná pro použití s překladačem jazyka C.

⁶Pipelining neboli zřetěžené zpracování, či překrývání instrukcí. Základní myšlenkou je rozdělení zpracování jedné instrukce mezi různé části procesoru a tím i dosažení možnosti zpracovávat více instrukcí najednou.



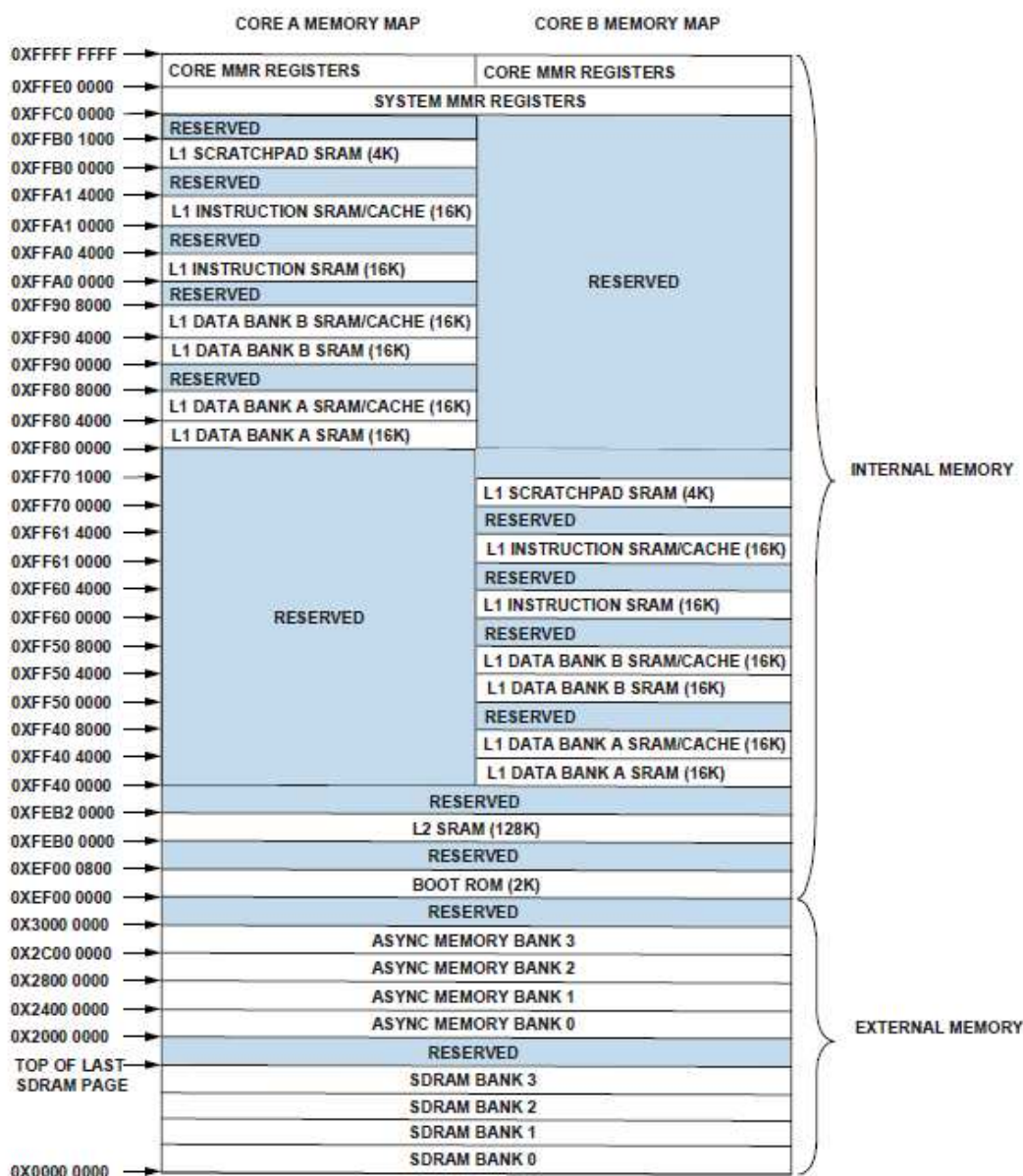
Obrázek 3.2: Struktura jádra signálového procesoru ADSP-BF561.

3.4.2 Architektura paměti procesoru ADSP-BF561

Hierarchicky uspořádaná struktura paměťového systému je pojata jako společný 4GB adresní prostor využívající 32-bitové adresování. Všechny paměťové zdroje (interní paměť, externí paměť, vstupně/výstupní registry) používají oddělené sekce společného adresovacího prostoru, který je uspořádán ve struktuře spojující vlastnosti a výkon on-chip pamětí s pomalejšími externími paměťmi.

Procesory řady Blackfin podporují modifikovanou Harvardskou strukturu. Level 1 (L1) paměť pracuje přibližně na stejné rychlosti jako samotné jádro procesoru s malým nebo

téměř nulovým zpožděním. L1 paměť obsahuje oddělené části pro instrukce, data a na tzv. scratchpad části paměti jsou uloženy informace o lokálních proměnných. K dispozici je několik L1 paměťových bloků, konfigurovatelných jako SRAM nebo cache. *Memory Management Unit* (MMU) poskytuje ochranu paměti pro individuální úlohy a chrání systémové registry proti neúmyslným přístupům.



Obrázek 3.3: Architektura paměti ADSP-BF561.

Obě jádra procesoru ADSP-BF561 sdílejí tzv. „on-chip“ L2 paměť, dovolující rychlý

přístup k paměti typu SRAM, avšak o něco pomalejší než je tomu v případě L1 paměti. Paměť typu L2 podporuje Von Neumannovskou strukturu. L2 je tedy společná jak pro instrukce, tak pro data a pracuje přibližně na poloviční frekvenci samotného jádra. Její velikost je 128 kB.

Rychlý širokopásmový blokový přenos dat nebo kódu mezi rychlou interní (L1, L2) pamětí a pomalou externí (L3) pamětí umožňuje DMA řadič. Externí (tzv. „off-chip“) paměť je přístupná přes External Bus Interface Unit (EBIU), což je 32-bitový interface poskytující připojení až 4 bank synchronní (SDRAM) paměti a až 4 bank asynchronní paměti (flash paměť, EPROM, ROM, SRAM). Řadič SDRAM (vyhovující standartu PC133) může přistupovat až k 512 MB SDRAM paměti.

Kapitola 4

Vývojový kit pro ADSP-BF561

Při výběru vhodného HW řešení úloh zpracování obrazu a získání hloubkové mapy byl kladen důraz také na dostupnost vhodného vývojového kitu pro vybraný DSP procesor, který by byl zároveň univerzálně využitelný pro případné testování na reálných modelech.

Pro oba vybrané kandidáty DSP procesorů BlackFin® ADSP-BF561 od Analog Devices a TMS320DM642-600™ od Texas Instrument nabízejí výrobci originální vývojové kity (např. ADSP-BF561 EZ-KIT Lite®¹ pro ADSP-BF561 a DM648 Digital Video Development Platform² pro TMS320DM642-600™). Tyto kity jsou svými periferiemi uzpůsobeny nejen pro vývoj video aplikací, ale i audio a jiných aplikací. Svými rozměry, hmotností a nadbytečnými periferiemi nevyhovují univerzální využitelnosti.

Mimo vlastností a možností ADSP-BF561, které byly shrnuty v předchozí kapitole, byl ADSP-BF561 vybrán pro realizaci řešení také díky dostupnosti vhodného vývojového kitu od firmy Bluetechnix [19], specializující se na DSP procesory řady BlackFin®. Základem vývojových kitů firmy Bluetechnix je vývojová deska s vybranou procesorovou jednotkou (*Core Module*). Vývojová deska dále disponuje 60-pinovými rozšiřujícími konektory, na které se stavebnicově připojují rozšiřující moduly podle potřeby. Vývojový kit vybraný pro stereovidění se skládá z následujících modulů:

- modul procesorové jednotky CM-BF561
- modul vývojové desky DEV-BF5xxDA-Lite s integrovaným USB Debug Agentem
- modul rozšiřující kamerové desky EXT-BF5xx-CAM (se dvěma 2 MPix CMOS kamerami OV2640)

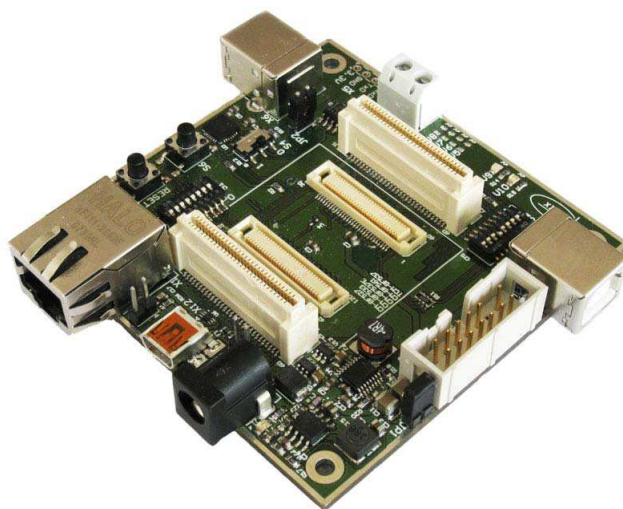
¹viz. webové stránky www.analog.com [21]

²viz. webové stránky www.ti.com [20]

Váha vývojové desky DEV-BF5xxDA-Lite je 60 g, procesorové jednotky CM-BF561 5 g, modulu rozšiřující kamerové desky EXT-BF5xx-CAM i se dvěma kamerami 40 g. Celý kit tedy váží 105 g. Rozměry vývojové desky i rozšiřujícího kamerového modulu jsou 75 x 75 mm. Odběr celého vývojového kitu, měřený na svorce JP1, byl 220 mA při režimu „nečinnosti“, 250 mA při běhu programu a 270 mA při snímání obrazů z kamer (napájecí napětí 5V).

4.1 Vývojová deska DEV-BF5xxDA-Lite s integrovaným Debug Agentem

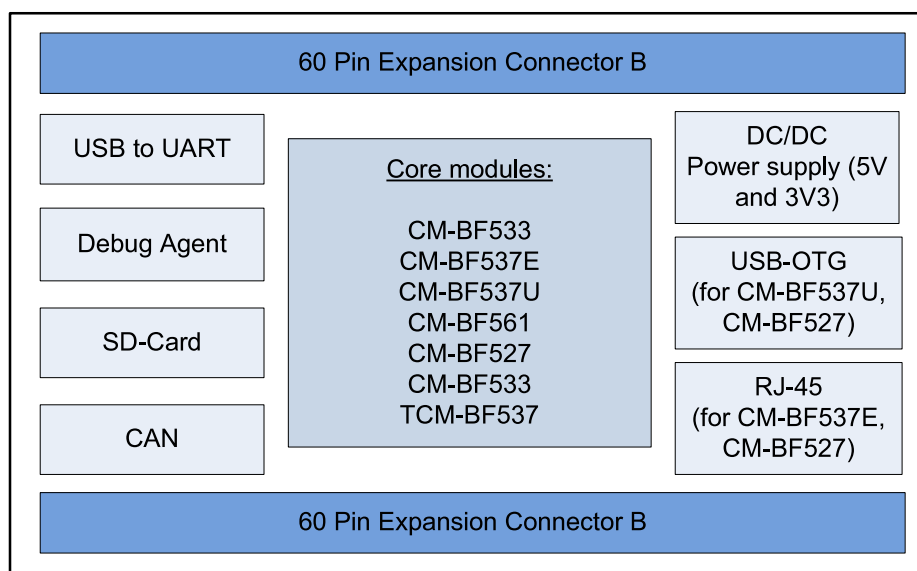
Vývojová deska DEV-BF5xxDA-Lite s integrovaným USB Debug Agentem (ve verzi 5.1) je určena pro programování a ladění aplikací pro procesory řady Blackfin. V tomto konkrétním případě pro procesor ADSP-BF561, který je srdcem procesorové jednotky CM-BF561 (kapitola 4.3). USB Debug Agent, umístěný na spodní straně vývojové desky, je kompatibilní s vývojovým prostředím *Visual DSP++ 5.0* od Analog Devices, které bylo při vývoji aplikací použito.



Obrázek 4.1: Vývojová deska DEV-BF5xxDA-Lite.

Vývojová deska DEV-BF5xxDA-Lite umožňuje připojit procesorovou jednotku a dále disponuje dvěma rozšiřujícími 60-pinovými konektory (X7 a X8), JTAG konektorem (X3)

dovolující přemostění JTAG USB Debug Agenta (X4), slotem pro SD karty, několika obecně použitelnými LED diodami, univerzálním tlačítkem, tlačítkem RESET a napájecím konektorem. Dále vývojová deska disponuje konektory pro periférie závislé na použitém DSP procesoru: USB konektorem (X6) pro výstup USB-UART převodíku (samotný výstup UART je k dispozici na konektoru X5), Ethernet konektorem (X1), konektorem (X9) pro OTG USB 2.0 rozhraní, CAN konektor (X10). Pro procesor ADSP-BF561 je k dispozici pouze USB-UART převodník. Jako zdroj napětí slouží dodávaný zdroj 12 V/2 A, který se připojuje do konektoru X11 a dále jsou na vývojové desce použity napěťové regulátory pro napětí 5 V/2 A a 3,3 V/1,5 A. Rozmístění a očíslování konektorů na desce plošných spojů je vidět na obrázku obr. A.10. Rozšiřující 60-pinové konektory v sobě pro procesorovou CM-BF561 mimo jiné sdružují připojení ke dvěma paralelním PPI sběrnicím.



Obrázek 4.2: Přehled součástí vývojové desky DEV-BF5xxDA-Lite.

Rozšiřující 60-pinový konektor č.1 přenáší signály pro

- sériový port SPORT0
- JTAG
- sériovou sběrnici SPI
- asynchronní sériovou sběrnici UART
- paralelní sběrnici PPI1
- vstupy/výstupy pro obecné použití

Rozšiřující 60-pinový konektor č.2 přenáší signály pro

- datovou sběrnici
- adresovou sběrnici
- řídicí signály pro paměť
- paralelní sběrnici PPI2
- napájecí napětí

4.2 Modul procesorové jednotky CM-BF561

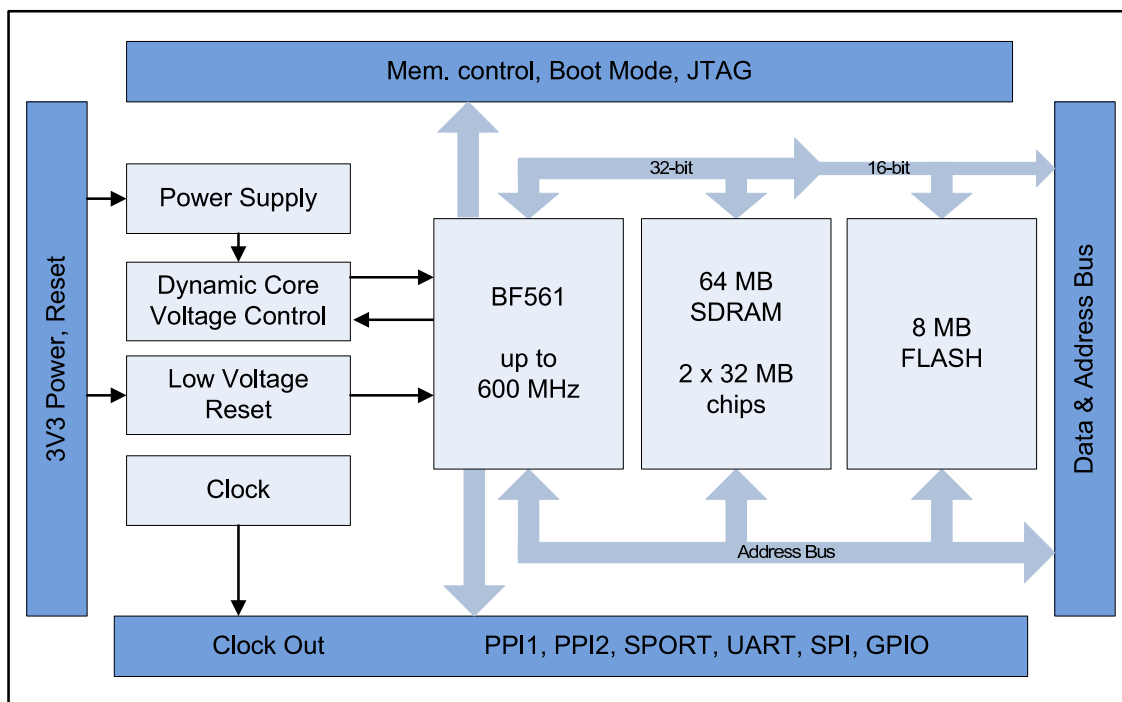
Vybraná procesorová jednotka CM-BF561 ve verzi 2.0, založená na dvoujádrovém procesoru ADSP-BF561 (ADSP-BF561SKBCZ600), poskytuje vysoký výkon fixed-point DSP procesoru a zaručuje nízkou spotřebu díky možnosti dynamicky změnit napětí jádra a jeho frekvenci. Při svých rozměrech 31 x 36 mm a váze pouhých 5 gramů jde o ideální variantu pro mobilní aplikace.



Obrázek 4.3: Procesorová jednotka CM-BF561.

Procesorová jednotka CM-BF561 dále obsahuje 25 MHz krystal jako zdroj frekvence, napěťový regulátor pro jádra procesoru (0,8V - 1,2V), 64 MB 32-bitové SDRAM paměti (2x MT48LC16M16A2BG-75IT) s možností pracovat na frekvenci až 133 MHz a 8 MB Flash paměti (Intel P30 8MB). Připojení k vývojové desce DEV-BF5xxDA-Lite tvoří opět dva 60-pinové konektory, na které jsou připojeny rozhraní UART, SPI, SPORT a paralelní sběrnice PPI1, PPI2. Přes tyto konektory je procesorová jednotka napájena napájecím

napětím 3,3V. Blokový diagram procesorové jednotky CM-BF561 je na obrázku obr. 4.4.



Obrázek 4.4: Blokový diagram procesorové jednotky CM-BF561.

8 MB Flash paměť je adresována od 0 x 20000000 do 0 x 207FFFFFFF. Adresovací prostor paměti SDRAM je 0 x 00000000 až 0 x 03FFFFFFF. Maximální dodávaný proud pro procesorovou jednotku může být 550 mA při napájecím napětí 3,3 V a napětí jádra 1,2 V. Následující tabulka tabulka 4.1 udává typickou spotřebu procesorové jednotky CM-BF561 pro dané podmínky ($t = 25^{\circ}\text{C}$).

Frekvence jádra	2 x 600 MHz	2 x 600 MHz	1 x 600 MHz
Napětí jádra	1,21 V	0,8 V	1,16 V
Vytížení SDRAM paměti	20%		
Spotřeba	470 mA	160 mA	330 mA

Tabulka 4.1: Typická spotřeba procesorové jednotky CM-BF561.

Značení uvedené na schématické značce CM-BF561 (obr. 4.5) odpovídá značení z katalogového listu Analog Devices pro ADSP-BF561 [12], [13]. V příloze jsou k dispozici tabulky tabulka A.1, tabulka A.2 a tabulka A.3, tabulka A.4 se seznamem jednotlivých

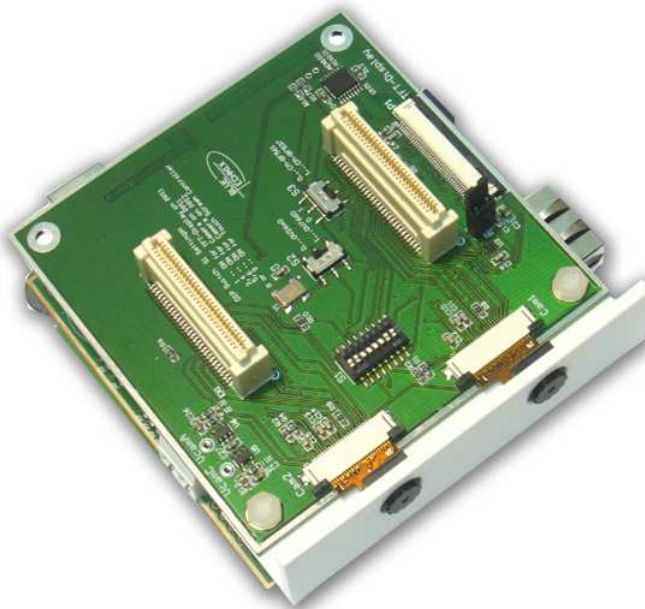
vývodů pro oba konektory procesorové jednotky CM-BF561. Volitelný konektor X3 pro rozšíření počtu obecně použitelných vstupů/výstupů nebyl v tomto případě použit.

1	RSCLK0 / PF28	ABE3 / SDQM3	61
60	RFS0 / PF19	A2	120
2	DR0PRI	A3	62
59	DR0SEC / PF20	A4	119
3	TSCLK0 / PF29	A5	63
58	TFS0 / PF16	A6	118
4	DT0PRI / PF18	A7	64
57	DT0SEC / PF17	A8	117
5	PF11 (CLK-out)	A9	65
56	PF10	A10	116
6	PF9	A11	66
55	PF8	A12	115
7	PF7 / SPISEL7 / TMR7	A13	67
54	PF6 / SPISEL6 / TMR6	A14	114
8	PF5 / SPISEL5 / TMR5	A15	68
53	PF4 / SPISEL4 / TMR4	PPI2Clk	113
9	Vin 3V3	PPI2Sy1	69
52	GND	PPI2Sy3	112
10	Vin 3V3	PPI2Sy2	70
51	GND	PPI2D0	111
11	PPI1D0	PPI2D1	71
50	PPI1D1	PPI2D2	110
12	PPI1D2	PPI2D3	72
49	PPI1D3	PPI2D4	109
13	PPI1D4	PPI2D5	73
48	PPI1D5	PPI2D6	108
14	PPI1D6	PPI2D7	74
47	PPI1D7	PF32 / PPI2D8	107
15	PPI1D8 / PF40	PF33 / PPI2D9	75
46	PPI1D9 / PF41	PF34 / PPI2D10	106
16	PPI1D10 / PF42	PF35 / PPI2D11	76
45	PPI1D11 / PF43	PF36 / PPI2D12	105
17	PPI1D12 / PF44	PF37 / PPI2D13	77
44	PPI1D13 / PF45	PF38 / PPI2D14	104
18	PPI1D14 / PF46	PF39 / PPI2D15	78
43	PPI1D15 / PF47	N.C.	103
19	PPI1Sy3	GND	79
42	PPI1Sy2 / TMR9	AMS2	102
20	PPI1Sy1 / TMR8	AMS1	80
41	PPI1Clk	ARE	101
21	PF3 / SPISEL3 / TMR3	AWE	81
40	PF2 / SPISEL2 / TMR2	AOE	100
22	PF1 / SPISEL1 / TMR1	NMI 0	82
39	PF0 / SPISS / TMR0	RESET	99
23	RX / PF27	D0	83
38	TX / PF26	D1	98
24	MOSI	D2	84
37	MISO	D3	97
25	SCK	D4	85
36	ABE0 / SDQM0	D5	96
26	ABE2 / SDQM2	D6	86
35	ABE1 / SDQM1	D7	95
27	ARDY	D8	87
34	AMS3	D9	94
28	TCK	D10	88
33	TDO	D11	93
29	TDI	D12	89
32	TMS	D13	92
30	TRST	D14	90
31	EMU	D15	91

Obrázek 4.5: Popis vývodů procesorové jednotky CM-BF561.

4.3 Rozšiřující modul EXT-BF5xx-CAM

Samotná vývojová deska neumožňuje připojení dvojice kamer. Z tohoto důvodu byl přidán rozšiřující stavebnicový modul EXT-BF5xx-CAM, který je na obrázku obr. 4.6. Tento kamerový modul se k vývojové desce připojuje pomocí 60-pinových rozšiřujících konektorů X7 a X8, které umožňují jeho napájení a veškerou komunikaci s vývojovou deskou, resp. s procesorem ADSP-BF561, včetně získání obrazových dat. Tyto konektory jsou na kamerovém modulu EXT-BF5xx-CAM „průchozí“ a na vrchní straně je možné připojit další stavebnicový rozšiřující modul podle požadavků aplikace. Pro tuto diplomovou práci byl použit pouze jeden rozšiřující modul.



Obrázek 4.6: Rozšiřující kamerová deska EXT-BF5xx-CAM.

Rozšiřující modul EXT-BF5xx-CAM podporuje připojení dvojice kamer³ s obrazovým senzorem Omnivision OV7760, OV2630 a OV2640. Pokud je připojená pouze kamera č. 2 (nebo žádná), je možné připojit ke kamerovému modulu také jeden z Hitachi displejů (TX09D50VM1CCA, TX09D70VM1CDA) nebo nějaký z kompatibilních displejů. Nastavení modulu pro 2 kamery OV2640 je popsáno v kapitole 4.6.2.

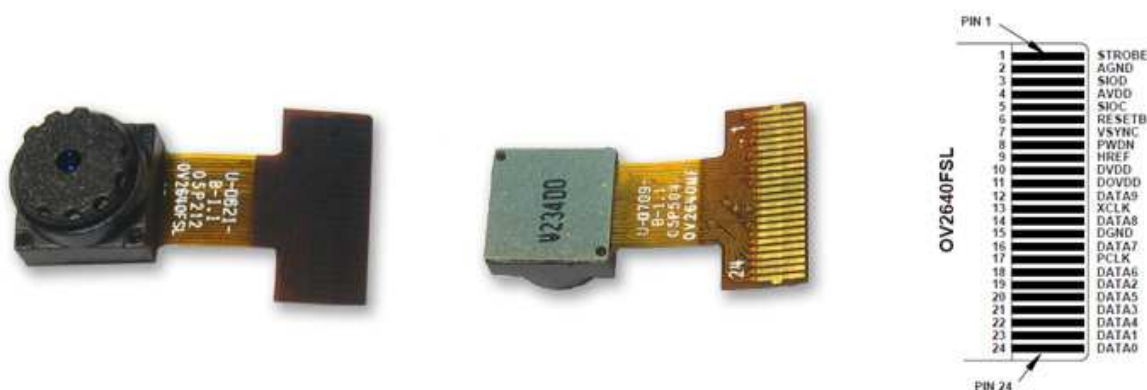
³Dvojici kamer nebo kameru společně displejem je možné připojit pouze při použití procesorové jednotky CM-BF561, resp. procesoru ADSP-BF561.

4.4 CMOS kamery OV2640FSL

Jak již bylo řečeno, rozšiřující modul EXT-BF5xx-CAM byl pro metodu stereovidění dodán společně se dvěma barevnými kamerami OV2640FSL (obrázek obr. 4.7). Tyto kamery jsou založeny na CMOS technologii a mají maximální rozlišení 1600 x 1200 pixelů, tj. 2 MPix (UXGA⁴). Podporují YUV422(420)/YCbCr422 a RGB565/555 8-bitový formát výstupu, 8/10-bitový RAW RGB výstup⁵ a také komprimovaný obraz. Maximální počet snímků za vteřinu, který je kamera schopna pořídit, je 15 pro rozlišení UXGA, 30 pro SVGA a až 60 pro CIF a nižší.

Optická soustátka má plastovou čočku. Diagonální zorný úhlep je 60° a ohnisková vzdálenost $f = 3,78$ mm. Kamera má pevně daný ostřicí rozsah 60 cm $\rightarrow \infty$. Citlivost použitého senzoru OV2640 je 0,6 V/Lux-sec a velikost jednoho pixelu 2,2 x 2,2 μm .

Kamera vyžaduje stejnosměrné napájení jádra o velikosti 1,3 V a napájení pro analogové obvody v rozsahu 2,5 - 3 V. Spotřeba je 125 mW⁶.



Obrázek 4.7: CMOS kamery OV2640FSL (pohled zepředu, zezadu a popis pinů konektoru kamery).

Rozměry pouzdra jsou 8 x 8 x 6 mm. Použitý konektor pro napájení a data je typu „flex cable“.

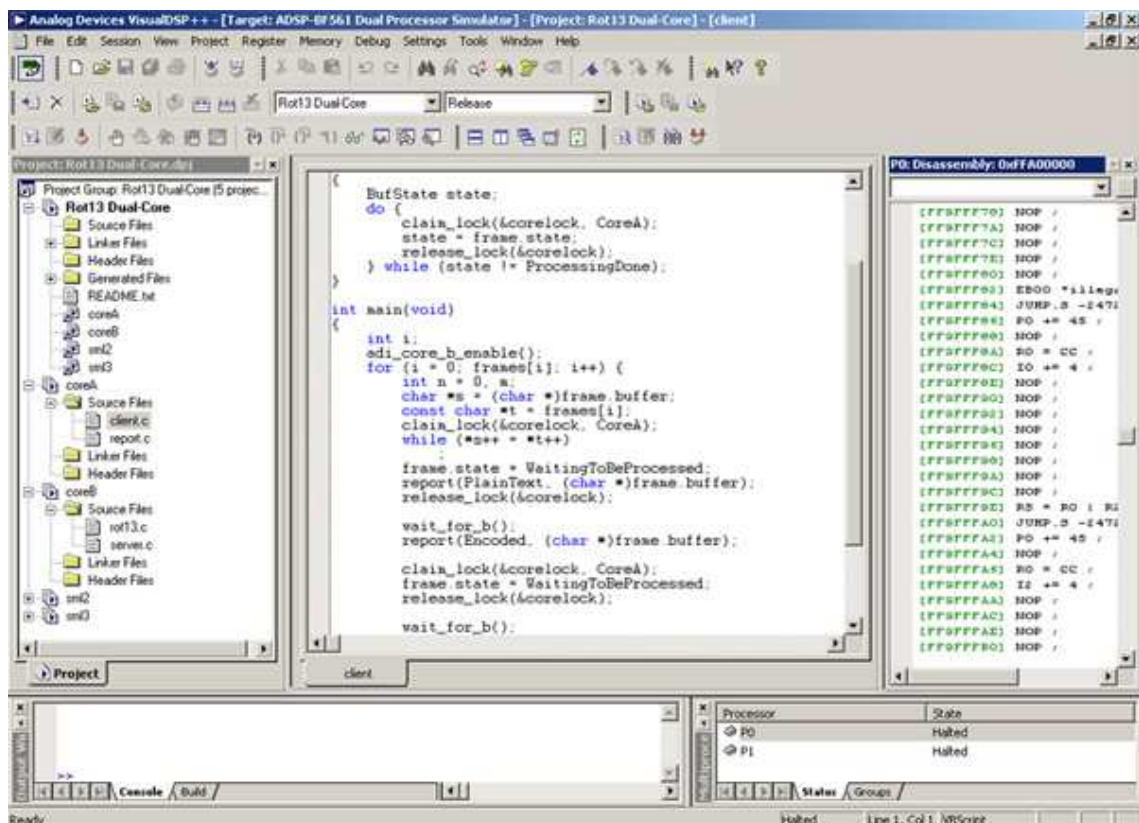
⁴Velikosti rozlišení: QVGA 320 x 240, (CIF 400 x 292), VGA 640 x 480, SVGA 800 x 600 ... UXGA 1600 x 1200 pixelů.

⁵RAW (soubor obsahující minimálně zpracovaná data ze snímače), YUV (barevný model, kde Y je jasová složka a U a V jsou barevné složky), RGB (aditivní způsob míchání barev červené, zelené a modré)

⁶rozlišení UXGA, YUV mód, 15 fps

4.5 Vývojové prostředí Visual DSP++ 5.0

Jako vývojové prostředí pro programování procesoru ADSP-BF561 byl použit SW od firmy Analog Devices *Visual DSP++ 5.0*. Toto prostředí podporuje USB Debug Agent dodávaný s vývojovou deskou DEV-BF5xxDA-Lite. Vytvoření a nastavení tzv. „Session“ je popsáno kapitole 4.6.4. Ta slouží k nastavení rozhraní mezi *Visual DSP++ 5.0* a konkrétním HW.



Obrázek 4.8: Uživatelské prostředí Visual DSP++ 5.0

Vývojové prostředí *Visual DSP++ 5.0* poskytuje plně integrované uživatelské prostředí s možností správy projektů (podporující i projekty pro vícejádrové procesory), ladění programu, simulátor různých procesorů včetně ADSP-BF561, podporuje jazyky C/C++ a assembler, umožňuje správu paměti na vývojovém kitu pomocí programu *expert linker*, analýzu běhu programu a časové náročnosti jednotlivých funkcí, grafické znázornění dat procesoru a nástroje pro optimalizaci.

4.6 Nastavení a instalace vývojového kitu DEV-BF5xxDA-Lite pro CM-BF561

V této kapitole jsou shrnuty základní kroky pro zapojení a nastavení vývojové desky DEV-BF5xxDA-Lite a rozšiřujícího kamerového modulu EXT-BF5xx-CAM. Nastavení je provedeno pro dvojici CMOS kamer OV2640FSL a procesor ADSP-BF561. Dále je popsána instalace driverů pro OS Microsoft WinXP a nastavení vývojového prostředí *Visual DSP++ 5.0* pro práci s HW Bluetechnix.

4.6.1 Nastavení vývojové desky DEV-BF5xxDA-Lite

Nastavení vývojové desky DEV-BF5xxDA-Lite provedeme pomocí DIP přepínačů S a jumperů JP. DIP přepínač S1 slouží k připojení konektoru Ethernetu k jádru. Pro procesor ADSP-BF561 nelze Ethernet použít, a tak jsou všechny přepínače na S1 v poloze „Off“. DIP přepínačem S2 volíme použitou procesorovou jednotku a bootovací nastavení. CM-BF561 má omezený počet volných pinů a z tohoto důvodu nelze nastavit bootovací mód - přepínače 1 - 4 jsou v poloze Off. Nastavení přepínačů 5 - 8 procesorové jednotky CM-BF561 je 5 - Off, 6 - On, 7 - Off, 8 - Off. DIP přepínačem S4 přepínáme mezi výstupem portu UART na konektor X5 (přepínač v poloze 1) nebo na USB-UART převodník, připojený na USB konektor X6 (přepínač v poloze 0). Jumper JP1 připojuje napájecí napětí na vývojové desce a lze také použít k měření odběru proudu. JP2 připojuje V_{dd} RTC na napětí 3,3 V procesorové jednotky. Alternativně lze jumper vyjmout a V_{dd} RTC⁷ napájet z bateriového zdroje. Jumpery JP3 a JP6 jsou při použití CM-BF561 rozpojené.

Vývojová deska dále obsahuje dvě tlačítka a několik informačních LED diod. Tlačítko S3 je resetovací a tlačítko S6 slouží k obecnému použití a na CM-BF561 je připojené na vývod PF46. V tabulce tabulka A.6 je seznam LED diod s popisem funkce. LED diody V5 - V8 indikují stav USB Debug Agentu a diody V9 a V10 slouží k obecnému použití.

4.6.2 Nastavení rozšiřujícího modulu EXT-BF5xx-CAM

Na obrázku obr. A.11 v příloze je vidět rozmístění konektorů a přepínačů rozšiřující kamerové desky EXT-BF5xx-CAM. Konektory Cam1 a Cam2 slouží k připojení CMOS kamer. EX1 a EX2 je v tomto případě označení „průchozích“ 60-pinových rozšiřujících

⁷Real-Time Clock

konektorů pro připojení dalšího modulu. Na desce EXT-BF5xx-CAM jsou umístěny také tři přepínače S1, S2 a S3. DIP řepínač S1 umožňuje vybrat připojený HW (diplej, kamery). Pro 2 kamery nastavíme jednotlivé přepínače S1 do pozic (1, 2, 8, 9 - Off a 3, 4, 5, 6 - On). Pro typ kamer OV26x0 nastavíme přepínač S2 do pozice 0. Přepínačem S3 vybereme typ procesorové jednotky - pozice 0 pro CM-BF561. Kamery s obrazovým senzorem Omnivision OV7760, OV2630 a OV2640 vyžadují různé analogové napájecí napětí V_{cc_A} a různé napětí pro své jádro V_{cc_C} . Konkrétní použitá kamera OV2640FSL vyžaduje $V_{cc_A} = 1,3 \text{ V}$ a $V_{cc_C} = 2,7 \text{ V}$. Napětí V_{cc_A} se nastaví rezistory $R_{11} = 10 \text{ k}\Omega$ a $R_{12} = 120 \text{ k}\Omega$. Napětí V_{cc_C} rezistory $R_{14} = 14 \text{ k}\Omega$ a $R_{15} = 10 \text{ k}\Omega$. Jumper JP1 na kamerové desce EXT-BF5xx-CAM připojuje napájecí napětí a je možné ho opět použít pro měření odběru proudu. Konektor P1 slouží k připojení displeje a je nevyužitý.

4.6.3 Ověření funkčnosti celého kitu a komunikace přes UART

Abychom mohli s vývojovou deskou DEV-BF5xxDA-Lite komunikovat přes rozhraní UART, je nutné nejdříve nainstalovat ovladače pro převodník USB-UART. Do konektorů desky DEV-BF5xxDA-Lite zapojíme procesorovou desku CM-BF561 a do konektoru X11 připojíme napájecí napětí z dodávaného 12 V adaptéru. Propojovací USB kabel zapojíme konektorem typu B do konektoru X6 a konektor typu A do PC. OS Windows XP by nyní měl sám rozeznat zařízení a dotázat se na instalaci ovladačů. Zvolíme výběr pro specifikaci umístění ovladače. Na CD Bluetechnix se nacházejí potřebné soubory ovladačů, případně je možné ovladač stáhnout z webových stránek [19]. Tuto proceduru je nutné zopakovat dvakrát. Jednou pro samotné USB a podruhé pro UART bridge. Soubory ovladače jsou pokaždé stejné. Ve správci zařízení OS poté zkontrolujeme, zda-li se objevilo nové zařízení - nový COM port „CP2101 USB to UART bridge controller“ (nejspíše COM4 nebo podobně). Pokud ano, pak komunikaci vyzkoušíme přes terminálový program (např. Hyperterminál obsažený v OS Windows XP). Otevřeme terminál a nastavíme tyto parametry:

- rychlost 115200 Baud
- počet datových bitů 8
- parita žádná
- počet stop-bitů 1, řízení toku žádné

Pokud vše proběhlo korektně, pak se stiskem tlačítka S3 (RESET) objeví v okně Hyperterminálu bootovací obrazovka systému BLACKSheep, který je firmou Bluetechnix defaultně nahrán v procesoru.

4.6.4 Instalace ovladačů USB Debug Agent a vývojového prostředí Visual DSP++ 5.0

Nejdříve provedeme instalaci a registraci vývojového prostředí *Visual DSP++ 5.0*. Instalace a registrace je popsána v přiloženém manuálu a je intuitivní. Instalaci je vhodné doplnit o poslední dostupný update [21].

Pro nahrání a ladění programů procesoru ADSP-BF561 je ještě potřeba nastavit tzv. Session, která zprostředkovává komunikaci s USB Debug Agentem umístěným na vývojové desce DEV-BF5xxDA-Lite. Jakmile dokončíme instalaci *Visual DSP++ 5.0* spustíme utilitu *DEV-BF5xxDA-Lite Installer* (obr. 4.9), která nakonfiguruje vývojové prostředí pro podporu Bluetechnix vývojové desky. Z nabídky vybereme procesorovou desku CM-BF561 a nainstalujeme.

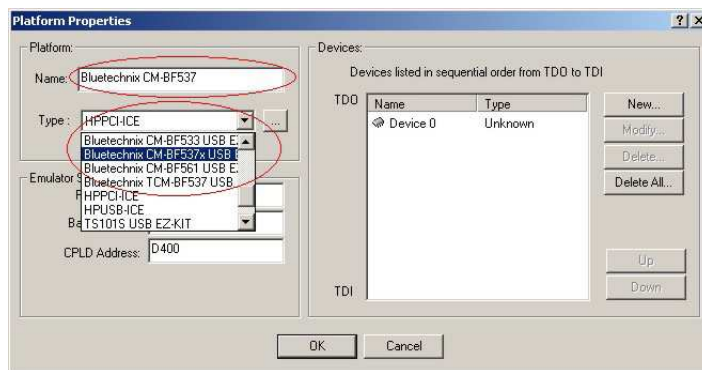


Obrázek 4.9: Obrazovka utility DEV-BF5xxDA-Lite Installer.

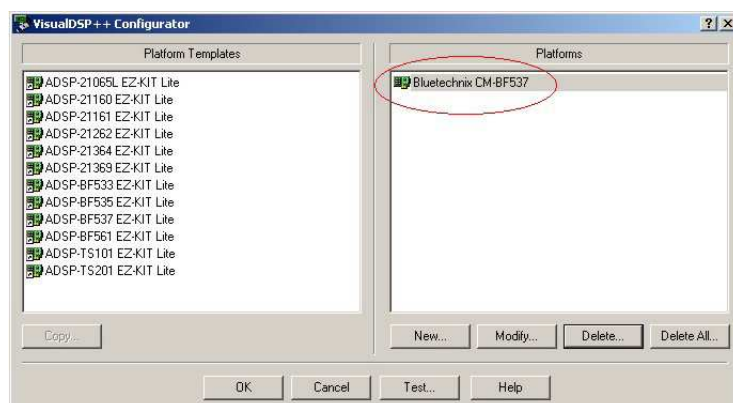
Připojíme další USB kabel, obdobně jako pro USB-UART převodík (viz. výše), tentokrát do konektoru X4. Ovladače nainstalujeme pomocí automatické instalace. Spustíme Visual DSP++ Configurator, který se nám nainstaloval spolu s vývojovým prostředím. Zadáme název a vybereme správný typ platformy *Bluetechnix CM-BF561 EZ-Kit*, jak je tomu naznačeno na obrázcích obr. 4.10 a obr. 4.11.

Zavřeme Visual DSP++ Configurator a ve vývojovém prostředí VisualDSP++ 5.0 vybereme z menu položku *Session->New Session*, vybereme procesor ADSP-BF561, stiskneme tlačítko *Next*, vybereme *EZ-Kit Lite connection type* a ukončíme stiskem tlačítka

Finish. Těmito kroky jsme nastavili komunikaci vývojového prostředí *Visual DSP++ 5.0* s USB Debug Agentem.



Obrázek 4.10: Visual DSP++ konfigurátor - nastavení platformy.



Obrázek 4.11: Visual DSP++ konfigurátor - výběr platformy.

Kapitola 5

Algoritmizace metody stereovidění na procesoru ADSP-BF561

Úloha stereovidění a problémy s ní spojené jsou v současné době intenzivně řešeným problémem počítačového vidění. K dispozici je několik přístupů a metod řešení problému stereovidění, resp. hledání korespondencí. V zásadě je můžeme shrnout do těchto dvou zběžných kategorií: tzv. *area-based* přístup (využíváme určitého výřezu jasových úrovní typu okno z jednoho obrazu k porovnání/korelaci s pohybujícím se oknem stejné velikosti v dalším obrazu) nebo tzv. *feature-based* přístup (korelaci provádíme mezi jistými objekty nalezenými v obraze - může jít o hrany, čáry, složitější objekty, ap.). Tyto kategorie se prolínají a lze nalézt také hybridní přístupy, spojující jejich vlastnosti.

Hlavním problémem hledání korespondencí je obecně velké množství obrazových dat, které je nutné prohledávat a častá nejednoznačnost nalezených korespondencí, především v *area-based* přístupu při nepřiliš strukturované scéně s nízkým kontrastem, bez dostatečné textury. Řešení složitějších přístupů k stereovidění přesahuje rámec této diplomové práce.

Mobilní zařízení, které se chce pohybovat a orientovat v prostoru, detekovat překážky, vyžaduje dostatečně rychlé zpracování disparitní/hloubkové mapy, aby na tyto vnější podněty stačilo reagovat. Pokud chceme stereovidění použít k základní detekci překážek, je nutné vybrat takový přístup zpracování obrazových dat, který bude vybraný HW schopen zpracovat v přiměřené rychlosti. Z těchto důvodů bylo zvoleno nízké rozlišení zpracovaných obrazových snímků z kamer, které je možné uložit do rychlých vnitřních pamětí samotného procesoru. Vybraný postup hledání korespondencí, jejich filtrace a získání disparitní mapy vychází z článku [11]. Tento algoritmus byl nejdříve otestován ve vývojovém prostředí *Matlab verze 7.1*. Pro zpracování na samotném DSP procesoru byl algoritmus upraven a je popsán v kapitole 5.5. Kódován a odladěn je ve vývojové prostředí

Visual DSP++ 5.0. Prostředí *Matlab* bylo taktéž použito pro tvorbu a čtení datových souborů přenášených z DSP procesoru do PC a naopak. Před samotným hledáním korespondencí, je nezbytné provést zarovnání získaných stereoskopických snímků procedurou nazvanou rektifikace, zahrnující i odstranění zkreslení kamer (popsanou v kapitole 5.4.1 o kalibraci kamer). Rektifikační a kalibrační parametry stereoskopického uspořádání kamer byly vypočteny použitím *Matlab Camera Calibration Toolboxu*.

Vybraný DSP procesor ADSP-BF561 je před samotným zpracováním algoritmu nutné uvést do požadovaného pracovního stavu, nastavit periferie pro získání obrazových informací a komunikaci s PC. Této části se věnuje kapitola 5.1 až 5.3. Vývojové prostředí a způsob algoritmizace dovolují optimalizaci výkonu výsledného programu stereovidění, čímž se zabývá závěrečná kapitola 5.7.

5.1 Nastavení projektu ve Visual DSP++ 5.0

Vývojové prostředí *Visual DSP++ 5.0* nabízí vytvoření hned několika odlišných typů projektů obsahující zdrojové kódy pro dvoujádrový procesor ADSP-BF561. Jsou to tyto tři základní přístupy:

- aplikace běžící pouze na jednom jádře (druhé jádro je uvedeno do nečinného stavu *Idle*) - tzv. *Single-Core Application*
- jádra se chovají jako nezávislé procesory, aplikace jsou zkompileovány zvlášť a každá běží na svém jádře (sdílené prostředky jsou obsluhovány vývojářem) - tzv. *One Application Per Core*
- jedna aplikace je zkompileována pro obě jádra a sdílené prostředky jsou obsluhovány linkerem - tzv. *Single Application/Dual Core*

Při algoritmizaci metody stereovidění byl použit nejdříve první přístup aplikace běžící pouze na jednom jádře a později třetí přístup, kdy byla aplikace rozdělena na dvě části běžící paralelně na obou jádrech. V případě prvního přístupu obsahuje tzv. *Project group*¹ dva projekty. Projekt s aplikací pro první jádro a defaultní „nečinný“ projekt pro jádro druhé. Projekty jsou rozděleny na složky zdrojových kódů, hlavičkových a linkovacích souborů, souborů vygenerovaných vývojovým prostředím a závislostmi mezi projekty. Třetí přístup k programování dvoujádrového procesoru se skládá ze speciální struktury

¹Pracovní skupina, ve které jsou v definované struktuře uloženy jednotlivé projekty.

5 projektů (projekt pro jádro A, projekt pro jádro B, sdílený kód a data paměti L2, sdílený kód a data paměti L3 a hlavní projekt). Hlavní projekt obsahuje závislosti na ostatní projekty a linkovací LDF² soubor. Aplikace projektu pro jádro A nebo B může využít kód ze své L1 paměti nebo sdílené paměti L2, L3. Přeložením této pětičky projektů získáme dva spustitelné soubory, jeden pro každé jádro.

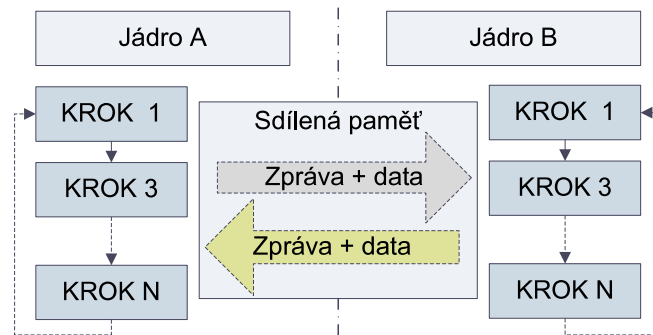
5.2 Nastavení procesoru

Před samotným zpracováním dat se pro procesor ADSP-BF561 musí spustit běh druhého jádra pomocí funkce *adi_core_b_enable()*. V této funkci se nepředává žádný parametr. Pomocí systémových power management funkcí s prefixem „*ADI_PWR_*“ lze nastavit napětí jádra a/nebo jeho frekvenci. V případě, že nastavíme pouze frekvenci, napětí jádra se nastaví na nejmenší možnou hodnotu, kvůli spotřebě. Pokud nastavíme pouze napětí jádra, nastaví se max. možná pracovní frekvence jádra. Také můžeme oba parametry kombinovat nebo nastavit vlastnosti jádra přímo pomocí systémových registrů. Dále je nutné nastavit frekvenci sběrnice externí paměti. Jako typické pracovní nastavení ADSP-BF561 byla použita frekvence jádra 500 MHz a frekvence sběrnice externí SDRAM paměti 125 MHz. Podrobné informace o power managementu procesoru ADSP-BF561 lze najít v literatuře [14], případně o dynamickém power managementu v literatuře [13]. Po nastavení parametrů jader inicializujeme systém přerušení - *ADI interrupt manager* a definujeme paměťové buffery pro zpracování obrazových dat.

Abychom mohli činnost aplikace rozdělit na dvě jádra procesoru, musíme zabezpečit vhodnou komunikaci. Komplexní komunikaci a sdílení prostředků může zajišťovat VDK³ vývojového prostředí *Visual DSP++ 5.0*, ale pro potřeby navržené aplikace byla zbytečně komplikovaná. Z tohoto důvodu byla naprogramována jednoduchá fronta zpráv. Využívá systémové funkce zamykání *claim_lock()* a odemykání *release_lock()* proměnných, umístěných v L2 (L3) paměti, která vzájemně předává stavy procesů jader a potřebná data. Blokové schéma této fronty zpráv s jednotlivými stavy procesorů je na obrázku obr. 5.1.

²Linker Description File - umístění kódu, dat v paměti

³VisualDSP++ Kernel



Obrázek 5.1: Blokové schéma použité fronty zpráv pro komunikaci jader.

5.3 Nastavení rozhraní UART, kamer a paralelních sběrnic PPI

V následujícím kroku programu jsou inicializovány a nastaveny obecné vstupy a výstupy vývojové desky DEV-BF5xxDA-Lite, komunikační rozhraní UART, obě paralelní PPI sběrnice, komunikace s kamerami prostřednictvím protokolu SCCB⁴ a registry kamer. Průběh inicializace a stav aplikace je průběžně zobrazován na obrazovce hyperterminálového programu prostřednictvím komunikace přes rozhraní UART.

Nastavení komunikace rozhraní UART se provede pomocí 2 funkcí `uart_open(...)` a `uart_setMode(...)`.

```
T_UART_HANDLE hUART = uart_open(0, sclk, 1024, 1024, 0);
uart_setMode(hUART, UART_BAUD_RATE, UART_NONE, 8, 1);
```

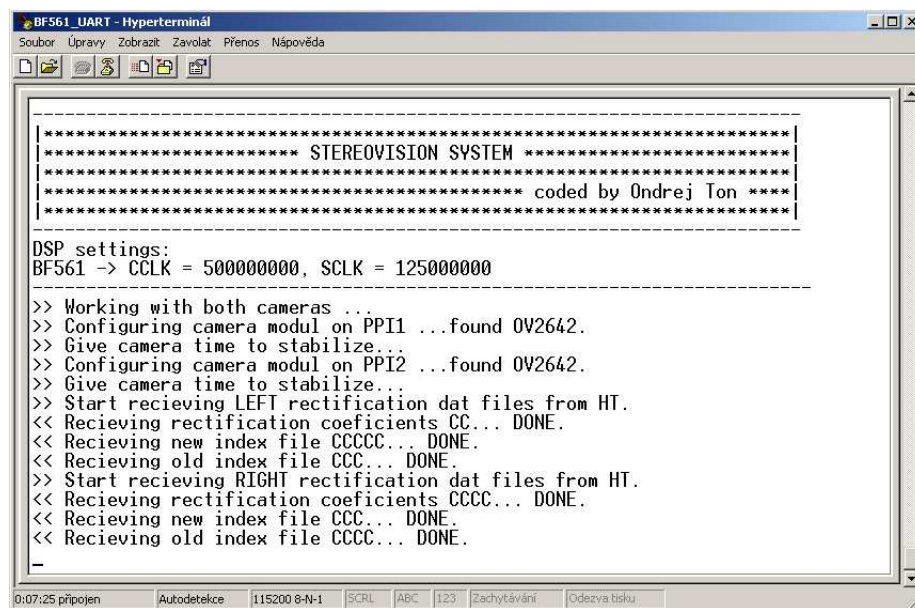
Funkcí `uart_open(...)` získáme UART handler `hUART`. Parametry funkce jsou číslo UART portu, hodnota systémových hodin, velikost vstupního a výstupního bufferu. Posledním parametrem je `callback` funkce. Funkce `uart_setMode(...)` pro handler `hUART` nastaví stejné parametry jako v kapitole 4.6.3 (rychlost, žádné řízení toku, 8 datových bitů a 1 stop-bit). Výpis textu na obrazovku obstarává funkce `uart_writeString(...)`. Pro nahrání datových souborů z PC do procesoru a naopak jsou využity funkce `XM_ReceiveFile` a `XM_TransmitFile` používající protokol X-Modem.

```
uart_writeString(hUART, sMessage);
XM_ReceiveFile(hUART, datFile);
XM_TransmitFile(hUART, datFile, size);
```

⁴Serial Control Camera Bus

Nastavení registrů kamer (parametr *pCfgParam*) je provedeno funkcí *cam_setup(...)*. *phCamera* je ukazatel na handler kamery. *tSIOCf* je hodinový signál, *tSIODf* datový signál pro SCCB komunikaci s kamerou a *tPwDn* signál snížení spotřeby kamery. Adresa v parametru *CAM_ADDRESS* je 0x60 pro kamery OV2640. Parametr *cclk* předává informaci o frekvenci jádra.

```
T_ERROR_CODE erResult = cam_setup(&phCamera, tSIOCf, tSIODf, tPwDn, CAM_ADDRESS, pCfgParam, cclk, 0);
```



Obrázek 5.2: Ukázka hyperterminálové obrazovky komunikující s procesorem ADSP-BF561, na kterém běží program stereovidění.

Jakmile je veškerá inicializace hotova, hyperterminálová obrazovka vyzve k nahrání setu kalibračních a rektifikačních dat. Můžeme se také rozhodnout, zda chceme pracovat se snímky z kamer nebo se snímky nahranými přes rozhraní UART (tato možnost volby se mění přímo ve zdrojovém kódu).

Abychom získali snímky z kamer, nakonfigurujeme sběrnici PPI pro DMA přenos, vybraný mód obrazu YUV a rozlišení $nXsize \times nYsize$ funkcí *ppi_setup_gp(...)*, kde $cPPI^5$ je číslo PPI sběrnice, YUVbuffer je buffer pro uložení snímku. Ostatní parametry funkce nastavují registry PPI sběrnice a vlastnosti DMA přenosu dat pro požadovaný formát

⁵ $PPI1 = 0, PPI2 = 1$

obrazových dat (blíže jsou popsány v hlavičkovém souboru funkce). Posledním parametrem je *callback* funkce pro dokončení DMA přenosu.

```
ppi_setup_gp(cPPI, YUVbuffer, 0x002c, nYsize, (nXsize * 2) - 1, 0, 0x00b2,
            nXsize * 2, 1, nYsize, 1, 0, (T_PPI_CALLBACK)PPIHandler);
```

Snímky z kamer získáme DMA přenosem, aktivovaným funkcí *ppi_enable(cPPI)*. Pokud během daného časového intervalu nezískáme požadovaný snímek, objeví se na hyperterminálové obrazovce chybová hláška. Pokud je přenos obrazových dat z kamer do paměti úspěšný pokračuje program ve vykonávání programu dále popsaného v kapitole 5.5.

Veškeré zde popisované kroky jsou také vidět na blokovém schématu na obrázku obr. 5.5.

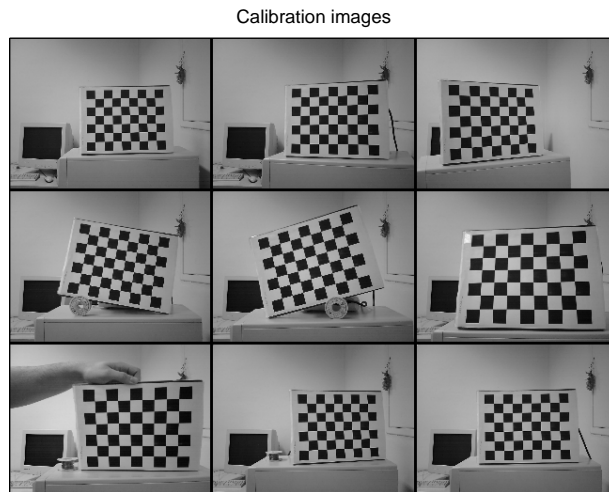
5.4 Kalibrace a rektifikace kamer - MATLAB Camera Calibration Toolbox

Jak bylo řečeno v kapitole 2.1.2, snímky z kamer jsou zatíženy zkreslením a kamerové řádky nejsou obecně zcela horizontálně zarovnané. Proto před samotným zpracováním dvojice stereoskopických snímků pro disparitní mapu odstraníme zkreslení a provedeme rektifikaci. Tím zjednodušíme problém 2D prohledávání na prohledávání korespondencí v jednom kamerovém řádku. Pro kalibraci kamer a rektifikaci stereoskopických snímků byl použit *Camera Calibration Toolbox* v prostředí *MATLAB 7.1*. Tento *Matlab* toolbox je intuitivní a výborně popsáný v literatuře [10], včetně praktických ukázek použití. Výpočet kalibračních matic a parametrů rektifikace se provede pouze jednou („*offline*“). Tato vlastnost je velice důležitá, protože pro aplikaci na DSP procesoru stačí vypočtené parametry „pouze“ aplikovat na sejmuté snímky.

5.4.1 Kalibrace kamer

Abychom mohli kameru (kamery) zkalibrovat pomocí *Camera Calibration Toolboxu*, potřebujeme ve scéně nasnímat kalibrační vzor. V tomto případě „šachovnice“ se známými parametry. Kalibrační vzor „šachovnice“ byl vytištěn na papír formátu A4 a obsahoval 7 x 9 polí o velikosti 27,5 x 27,5 mm. Vzorek byl umístěn na pevný podklad a nasnímán v dostatečném počtu (cca 10) snímků v různých vzdálenostech a pod různým úhlem

oběma kamerami. Snímání snímků pravou i levou kamerou zároveň je důležité pro rektifikaci. Nyní v *Camera Calibration Toolboxu* spustíme příkazem *calib_gui* nabídku funkcí, které toolbox nabízí pro kalibraci kamer. Vybereme snímky kalibrované kamery, které chceme použít a načteme je do paměti.



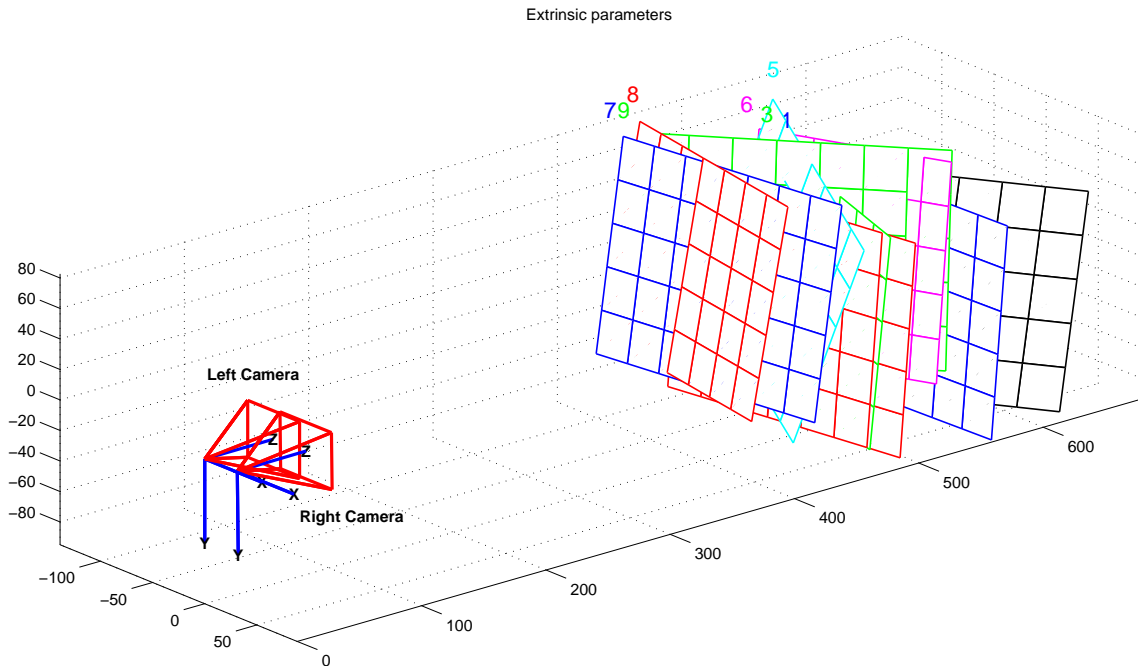
Obrázek 5.3: Nasnímaný kalibrační vzor.

Jakmile jsou všechny snímky k dispozici, provedeme ruční zadání referenčních bodů, v případě „šachovnice“ zadání souřadnic 4 obvodových rohů pro všechny načtené snímky. Po zadání 4 obvodových rohů extrahuje *Camera Calibration Toolbox* v každém snímku všechny rohy polí ve snímku unvitř obdélníku vymezeného obvodovými rohy. Z těchto dat *Camera Calibration Toolbox* provede kalibraci kamer - vypočte vnitřní a vnější parametry užitím řešení uzavřeného tvaru (anglicky *closed-form solution*), vypočte koeficienty radiální distorze řešením lineární regrese a veškeré parametry zpřesní minimalizací. Poté si můžeme zobrazit vnější parametry kamery a přehledné vykreslení jednotlivých pozic kalibračního vzoru v prostoru. K dispozici je také graf s chybami. Podle tohoto grafu se můžeme rozhodnout, jestli vyřadíme nějaký snímek, který způsobuje velkou chybu nebo jestli chceme znovu definovat obvodové rohy. Tím můžeme chybu kalibrace snížit. Po dokončení kalibrace můžeme ze snímků odstranit zkreslení přímo v toolboxu. Výsledky kalibrace pro levou a pravou kameru uložíme do *.mat souborů, které později použijeme pro výpočet rektifikačních parametrů. Parametry kamer jsou k dispozici v příloze na obrázku obr. A.13.

5.4.2 Rektifikace snímků

Data získané kalibrací dvojice kamer použijeme pro kalibraci a rektifikaci stereoskopických snímků. V *Camera Calibration Toolboxu* zadáme příkaz *stereo_gui*. Otevře se nám opět intuitivní menu, tentokrát pro stereo kalibraci a rektifikaci. Vybereme vypočtené kalibrační soubory pro obě kamery a spustíme stereo kalibraci. Poté si můžeme zobrazit vnitřní parametry kamer, zobrazit ve 3D prostoru vnější parametry obou kamer a pozice nasnímaných kalibračních vzorů (viz. obr. 5.4) a uložit vypočtené parametry do souboru. Data vygenerované toolboxem jsou nyní uloženy v těchto datových polích obsahujících:

- indexy pixelu v novém snímku (*ind_new_left* a *ind_new_right*)
- indexy pixelů původního snímku pro lineární interpolaci (*ind_1_left* - *ind_4_left* a *ind_1_right* - *ind_4_right*)
- interpolační koeficienty pro jednotlivé indexy starého snímku (*a_1_left* - *a_4_left* a *a_1_right* - *a_4_right*)



Obrázek 5.4: Zobrazení vnějších parametrů kamer v 3D prostoru a nasnímaných kalibračních vzorů.

Tyto data je nutné nahrát do paměti ADSP-BF561, abychom mohli snímky rektifikovat. Soubory s indexy jsou datového typu *unsigned short* (16 bitů). V závislosti na počtu použitých indexů je velikost souboru s indexy pixelů v novém snímku jedné kamery přibližně 37 kB⁶, souboru s indexy pixelů původního snímku jedné kamery přibližně 148 kB⁷. Interpolační koeficienty indexů původního snímku jsou uloženy jako desetinná čísla. Z důvodu úspory jak paměťového místa, tak zrychlení algoritmu snížením přístupů do paměti, byly provedeny následující kroky. Interpolační koeficienty⁸ byly z desetinných čísel převedeny na kladnou celočíselnou reprezentaci bez znaménkového bitu vlastního datového typu *fract8* (8 bitů) pomocí výrazu (5.1). Jak již bylo řečeno, pro výpočet hodnoty intenzity nového pixelu potřebujeme 4 interpolační koeficienty k_{1x} , k_{2x} , k_{3x} a k_{4x} . Ty jsou uloženy v této posloupnosti: k_{11} , k_{21} , k_{31} , k_{41} , k_{12} , k_{22} , k_{32} , k_{42} , ..., k_{1n} , k_{2n} , k_{3n} , k_{4n} . Velikost souboru s interpolačními koeficienty pro jednu kameru je tedy přibližně 74 kB⁹.

$$N = \text{round}(256 * x) \quad (5.1)$$

Datová struktura polí indexů původního snímku v *Camera Calibration Toolboxu* je uložena tak, že každému interpolačnímu koeficientu je přiřazen jeden index. Tato struktura je však redundantní, jelikož nový pixel se interpoluje vždy ze 4 nejbližších sousedů. Pokud si tedy uložíme pouze soubor *ind_1_left* pro levou kameru, ostatní indexy uložené v datových polích *ind_2_left* - *ind_4_left* lze spočítat podle rovnic (5.2), kde *yWidth* je výška snímku.

$$\begin{aligned} \text{ind_2_left} &= \text{ind_1_left} + yWidth \\ \text{ind_3_left} &= \text{ind_1_left} + 1 \\ \text{ind_4_left} &= \text{ind_1_left} + yWidth + 1 \end{aligned} \quad (5.2)$$

Tímto zjednodušením jsme původní velikosti souboru snížili ze 148 kB na 37 kB. Celkový paměťový prostor potřebný pro rektifikační soubory je přibližně 148 kB pro každou kameru, 296 kB celkem.

Při spuštění systému stereovidění nejdříve nahrajeme do systému všechny soubory potřebné pro rektifikaci obou kamer (prostřednictvím Hyperterminálu komunikujícího přes USB port PC s převodníkem USB-UART na vývojové desce). Pořadí v jakém se soubory mají nahrát vypisuje obrazovka Hyperterminálu.

⁶Výpočet pro 18706 indexů, uložených jako data typu *unsigned short*.

⁷Výpočet pro 4*18706 indexů, uložených jako data typu *unsigned short*.

⁸Koeficienty nabývají hodnot $\langle 0, 1 \rangle$ a součet těchto 4 koeficientů je roven 1.

⁹Výpočet pro 4*18706 indexů, uložených jako data typu *fract8*, resp. *unsigned char*.

5.5 Algoritmizace stereovidění

Z teoretického úvodu v kapitole 2.1 jsme se dozvěděli o základní geometrii dvou kamer, jak je převést do kanonického uspořádání a vypočítat vzdálenost korespondujících si bodů od kamerové základny. V této kapitole je popsána algoritmizace jednotlivých úloh pro získání výsledné disparitní mapy z dvou stereoskopických snímků.

5.5.1 Zpracování snímků z kamer

Snímky z kamer získáme postupem popsaným v kapitole 5.3. Pokud během daného časového intervalu nezískáme požadovaný snímek, objeví se chybová hláška na hyperterminálové obrazovce. Pokud je přenos obrazových dat z kamer do paměti úspěšný následuje převod barevného YUV formátu na intenzitní „černobílý“ obraz funkcí

```
imageConvertYUV2GS (YUVbuffer, imBuffer, nXsize, nYsize);
```

a snížení rozlišení na požadovaných 160 x 120 pixelů funkcí

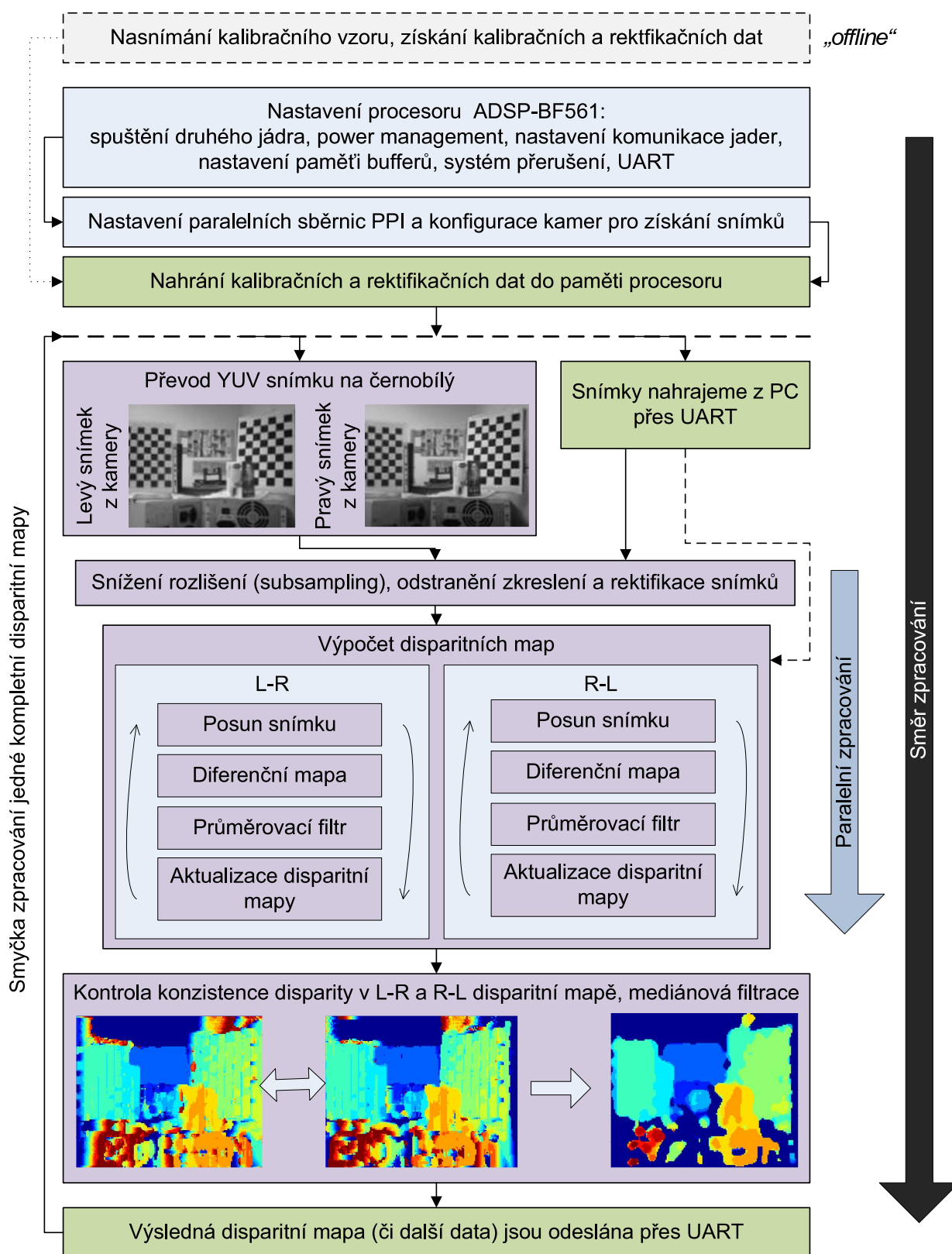
```
imageScalingDownBy2(imBuffer, imBufferSmall, nXsize, nYsize);
```

Parametry těchto funkcí jsou *YUVbuffer*, který se převede na intenzitní obraz uložený v *imBuffer* a ten se poté zmenší na potřebné rozlišení do *imBufferSmall*. *nXsize* a *nYsize* jsou rozměry původního snímku. YUV (správné označení pro užití v digitálních systémech je YCbCr) formát dat byl zvolen z důvodu jeho jednoduché konverze na intenzitní obraz. YCbCr je barevný prostor reprezentovaný složkou jasu (luminescence) Y a komponentami Cb a Cr, jež jsou modrou a červenou chromatickou komponentou. Pro získání intenzitního obrazu tedy stačí z přijatých dat uložit pouze jasovou složku Y. Popis tohoto barevného prostoru, grafické znázornění složek a převod do barevného prostoru RGB je uveden v literatuře [23].

Pokud je to potřeba, snímek je dále rektifikován pomocí funkce

```
imageRectification(original_image, rectified_image, new_index, old_index, coefficients, nIndexes);
```

Vstupem funkce je původní snímek *original_image* a výstupem rektifikovaný snímek *rectified_image*. Ostatní parametry (viz. kapitola 5.4.2) jsou koeficienty lineární transformace, indexy původního a rektifikovaného obrázku a jejich počet. Z těchto upravených snímků je možné vypočítat disparitní mapu.



Obrázek 5.5: Blokový diagram algoritmizace stereovidění.

5.5.2 Získání disparitní mapy

Snímky jsou rektifikované a zmenšené na požadovanou velikost. Samotný výpočet disparitní mapy sestává z cyklicky prováděných kroků:

- posun levého/pravého snímku o jeden krok (pixel) vlevo/vpravo
- vytvoření diferenční mapy odečtením pravého posunutého snímku od levého/levého posunutého snímku od pravého
- filtrace diferenční mapy průměrovacím filtrem
- aktualizace disparitní mapy v závislosti na nově vypočtených hodnotách filtrované diferenční mapy

Tyto kroky se provádějí dvakrát - jednou se posouvá pravý snímek vpravo a odečítá se s levým a podruhé posouvá levý snímek vlevo a odečítá s pravým. Tento zdvojený výpočet (tzv. L-R a R-L¹⁰ disparitní mapy) slouží k vyřazení těch hodnot disparit mezi L-R a R-L mapami, jejichž rozdíl je větší než povolená tolerance a bereme je jako špatné.

Prvním krokem cyklu výpočtu disparitní mapy sejmuté scény je posun snímku. Tato operace je provedna buďto funkcí *imageShiftLeft(...)* nebo funkce *imageShiftRight(...)*, kde vstupem je levý/pravý snímek *left_image/right_image*. Rozměry daného snímku jsou *inImgXSize* x *inImgYSize* a výstupem je obrázek *shifted_image* posunutý o počet pixelů hodnoty proměnné uložené v *shift*.

```
imageShiftRight(right_image,shifted_image,inImgXSize,inImgYSize,shift);
```

```
imageShiftLeft(left_image,shifted_image,inImgXSize,inImgYSize,shift);
```

Diferenční mapa *diff_map* je získána funkcí *imageDiff(...)*, jejíž prvními dvěma parametry jsou levý (pravý) posunutý obrázek a pravý (levý) původní rektifikovaný snímek.

```
imageDiff(shifted_image,image,diff_map,inImgXSize,inImgYSize);
```

Diferenční mapa je poté filtrována funkcí *imageSmoothing(...)* průměrovacího filtru daného jádrem o velikosti $n = 2*AVG_FILTER_SIZE+1$. Filtrace je cyklicky prováděna jako 2D konvoluce (5.3), kde h je konvoluční jádro (5.4) rozměru $n \times n$ a f jsou vstupní data. Jinak řečeno, průměrovací filtr provede součet vstupních hodnot a poté je vydělí jejich počtem (n^2).

$$g(x, y) = \sum_{k=0}^{n-1} \left(\sum_{l=0}^{n-1} ((h(x-k, y-l)f(k, l)) \right) \quad (5.3)$$

¹⁰Left to Right, Right to Left

$$h = \frac{1}{n^2} \begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix} \quad (5.4)$$

```
imageSmoothing(diff_map_filtered,diff_map,AVG_FILTER_SIZE,inImgXSize,inImgYSize);
```

Aktualizaci L-R nebo R-L disparitní mapy provedeme porovnáním hodnot aktuálně vypočítané diferenční mapy a diferenční mapy z předchozího kroku. K tomu slouží funkce *updateDisparityMap()*. *diff_map_filtered* je aktuální a *diff_map_tmp* předchozí diferenční mapa.

```
updateDisparityMap(diff_map_tmp,diff_map_filtered,disparity,inImgXSize,inImgYSize,shift);
```

Pro ty hodnoty, které budou menší nebo rovny než hodnoty diferenční mapy z předchozího kroku, aktualizujeme souřadnicově odpovídající hodnotu disparitní mapy na hodnotu aktuálního posunu *shift*. Nejvyšší možná hodnota disparity ve zpracované disparitní mapě je daná konstantou *MAX_DISP*, kterou si zvolíme podle požadovaného rozsahu měřených vzdáleností (např. pro *MAX_DISP* = 15 je interval měřených vzdáleností $\langle 34, 4; \infty \rangle$ cm, pro parametry uvedené v kapitole 5.6.

Zde uvedený programový cyklus provádí výpočet L-R disparitní mapy (R-L výpočet je analogický). Oproti zdrojovému kódu obsahuje jenom funkce důležité pro pochopení principu. Procedura rektifikace pravého, levého snímku a výpočet L-R, R-L disparitní mapy je jednoduše dekomponovatelný na dvě části, z nichž každá může běžet na jednom jádře.

```
// L-R disparity map
for (p = 0; p<MAX_DISP; p++) {

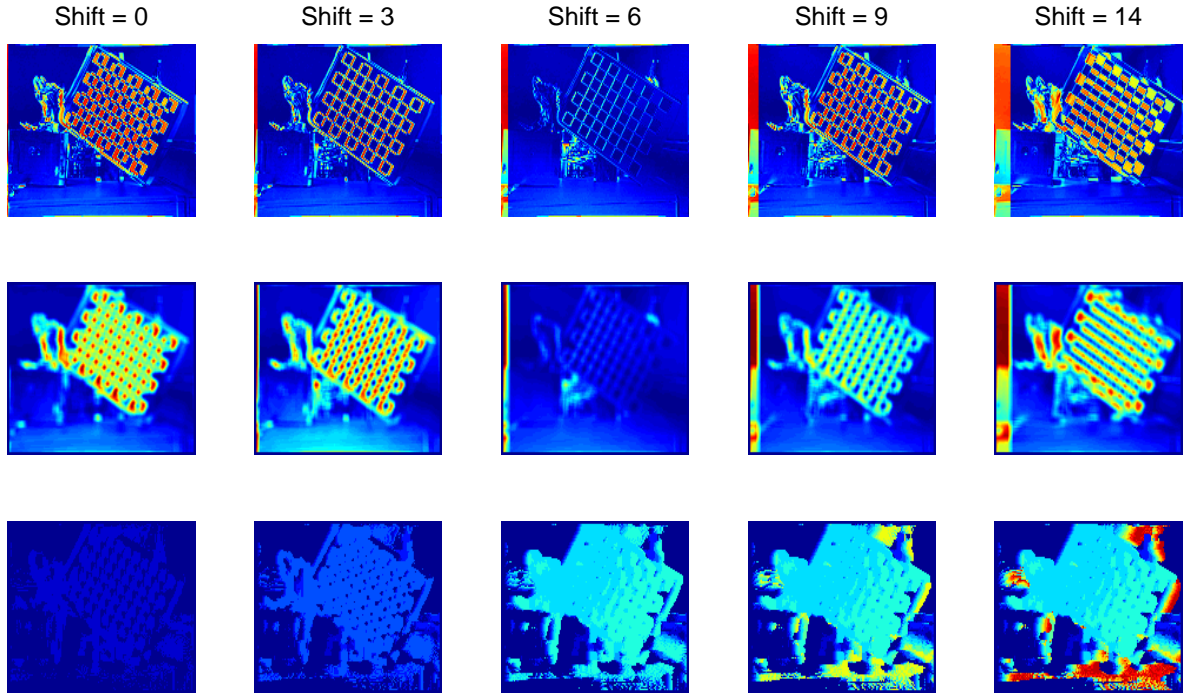
    imageShiftRight(right_image,shifted_image,inImgXSize,inImgYSize,shift);
    imageDiff(shifted_image,image,diff_map,inImgXSize,inImgYSize);
    imageSmoothing(diff_map_filtered,diff_map,AVG_FILTER_SIZE,inImgXSize,inImgYSize);

    updateDisparityMap(diff_map_tmp,diff_map_filtered,disparity,inImgXSize,inImgYSize,shift);

}
```

Na obrázku obr. 5.6 jsou v horní části je vyobrazeny výsledky funkce *imageDiff(...)* pro posun *shift* = [0, 3, 6, 9, 14] pixelů, uprostřed výsledky průměrovacího filtru a ve spodní části aktualizovaná disparitní mapa pro jednotlivé posuny. Horní dva řádky obrázků

odpovídají rozsahu 0 - tmavě modrá až 255 - tmavě červená. Aktualizace disparitních map je v rozsahu 0 - tmavě modrá až 15 - tmavě červená.

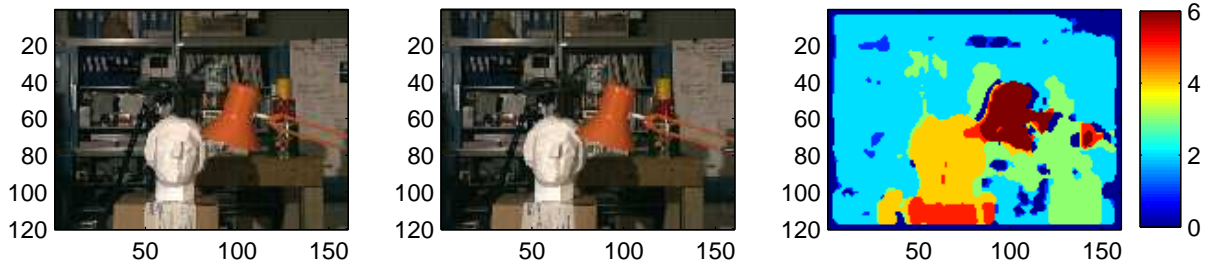


Obrázek 5.6: Grafické zobrazení kroků výpočtu disparitní mapy.

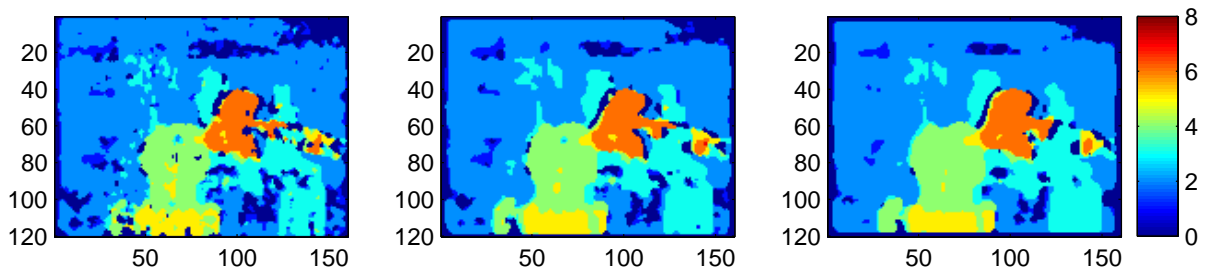
Jakmile získáme L-R i R-L disparitní mapu, použijeme funkci *findBestCandidates(...)*, která mezi těmito mapami vyhledá kandidáty „správné“ disparity. Správní kandidáti vyhovují podmínce (5.5). Abychom ještě více potlačili chyby způsobené nesprávně nalezenými korespondencemi, filtrujeme mapu „správných“ kandidátů mediánovým filtrem. Parametr *disparity_map*, použitý ve funkci *_median(...)*, obsahuje data výsledné disparitní mapy.

$$|dispm_{L-R} - dispm_{R-L}| < TOLERANCE \quad (5.5)$$

Vliv rozměru konvolučního jádra ($n \times n$) průměrovacího filtru na výslednou disparitní mapu (obr. 5.7) je vidět na obrázku obr. 5.8. Čím je jeho rozměr větší, tím více se stávají plochy disparitní mapy spojitě. Nevýhoda spočívá v rozmazávání zpracovaných dat („hranic objektů“) a nemožnosti rozlišit malé předměty. Pro úlohu detekce překážek to však nemá zásadní význam. V ostatních disparitních mapách bylo použito konvoluční jádro 5×5 .



Obrázek 5.7: Disparitní mapa pro známý dataset „tsukuba“.



Obrázek 5.8: Vliv konvolučního jádra průměrovacího filtru na výslednou disparitní mapu. Vlevo je disparitní mapa pro rozměr konvolučního jádra 3 x 3, uprostřed 5 x 5 a vpravo 7 x 7.

```
findBestCandidates(disparityL_R,disparityR_L,disparity,inImgXSize,inImgYSize,TOLERANCE);

_median(disparity,inImgXSize,inImgYSize,disparity_map);
```

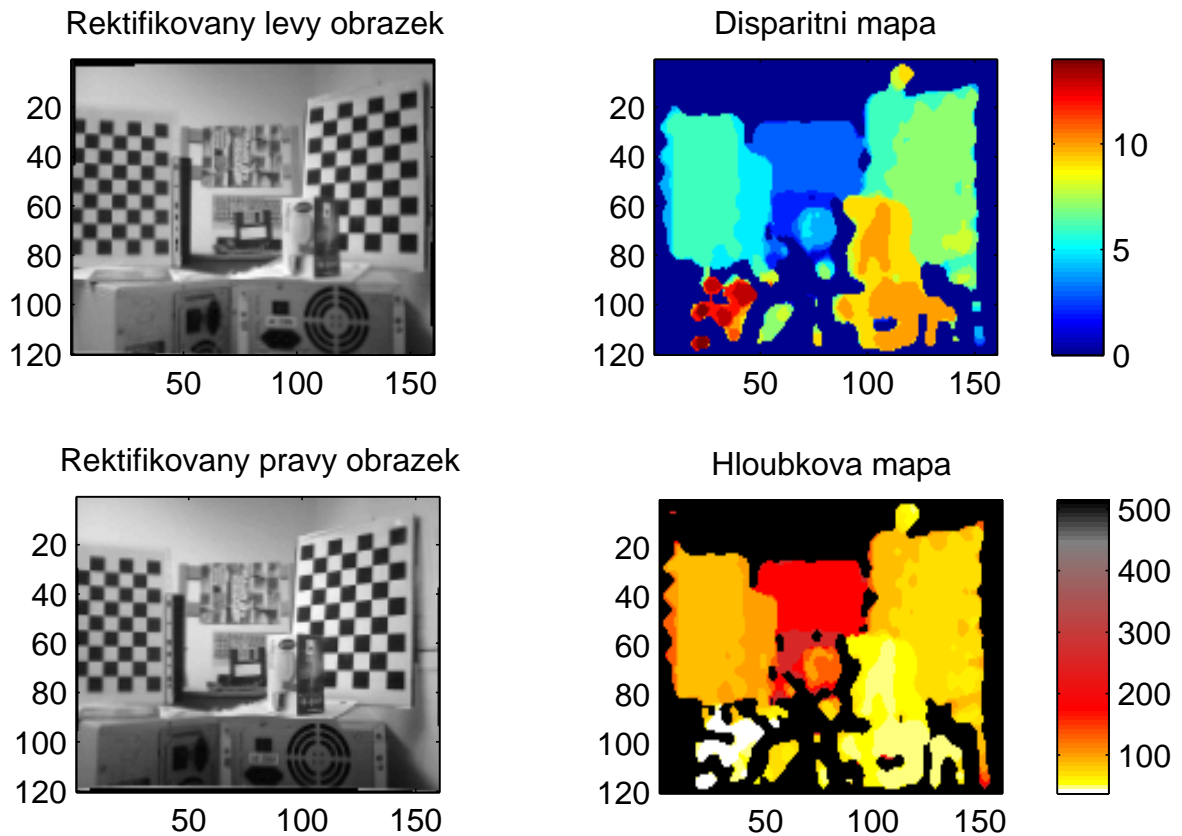
V příloze A jsou uvedeny výsledky vypočtených disparitních map různých scén zde popsaným algoritmem. U obrázku obr. A.8 nebo obr. A.1 je dobře vidět problém vzniku falešných korespondencí. Ty mohou vznikat např. v důsledku různých radiometrických vlastností dvou snímků. Na obrázku obr. A.9 jsou dva uměle vytvořené černé snímky. Z těchto snímků je patrné, že nemůže být nalezena žádná korespondence, ale díky filtraci nevznikají ani falešné korespondence. Málo texturovaný je také obrázek obr. A.7. Ačkoliv lidským okem rozeznáme objekty a jejich přibližnou vzdálenost, použitý algoritmus má s těmito snímky problémy. Na obrázku obr. A.6 můžeme dobře pozorovat vliv problému okluze, kdy dřevěná přepážka uprostřed snímku „tvoří“ uzavřený povrch pro pravou kameru a neuzavřený povrch pro kameru levou. Levý pruh nulových disparitních hodnot směrem od dřevěné přepážky je tvořen právě okluzí.

5.6 Výpočet hloubkové mapy

V kapitole 2 byly popsány pojmy disparitní mapa a hloubková mapa. Disparitní mapu, vyjadřující pixelovou vzdálenost dvou korespondujících si bodů v levém a pravém snímku, kterou jsme získali algoritmy popsanými v předchozím textu, chceme přepočítat na mapu hloubkovou. Hloubkovou mapu, vyjadřující vzdálenosti, vypočteme podle vztahu (5.6).

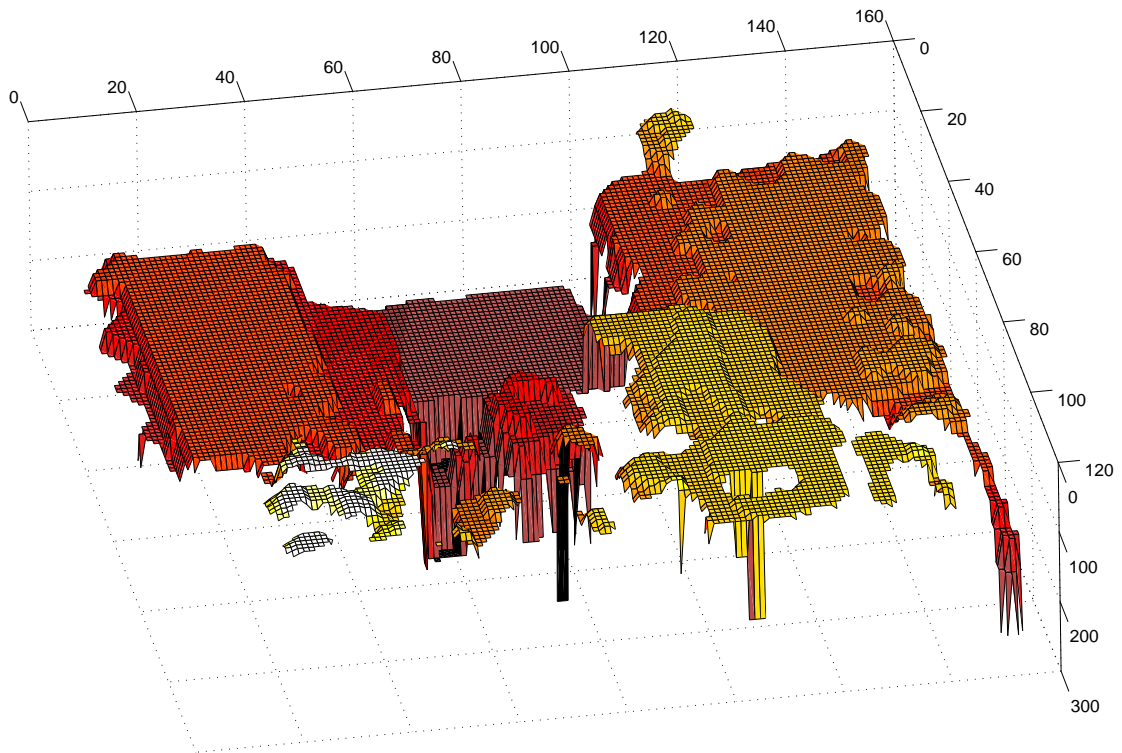
$$D = \frac{bf}{d_{disparity}p_{size}}, \quad (5.6)$$

kde D je vzdálenost bodu v prostoru, b základna (vzdálenost optických středů kamer), f ohnisková vzdálenost, p_{size} šířka jednoho pixelu kamerového senzoru a $d_{disparity}$ je jedna hodnota z disparitní mapy. Na obrázku je zobrazena disparitní mapa (vpravo nahoře) a z ní vypočítaná hloubková mapa v odlišné barevné paletě (vpravo dole). Jednotky hloubkové mapy jsou uvedeny v cm.

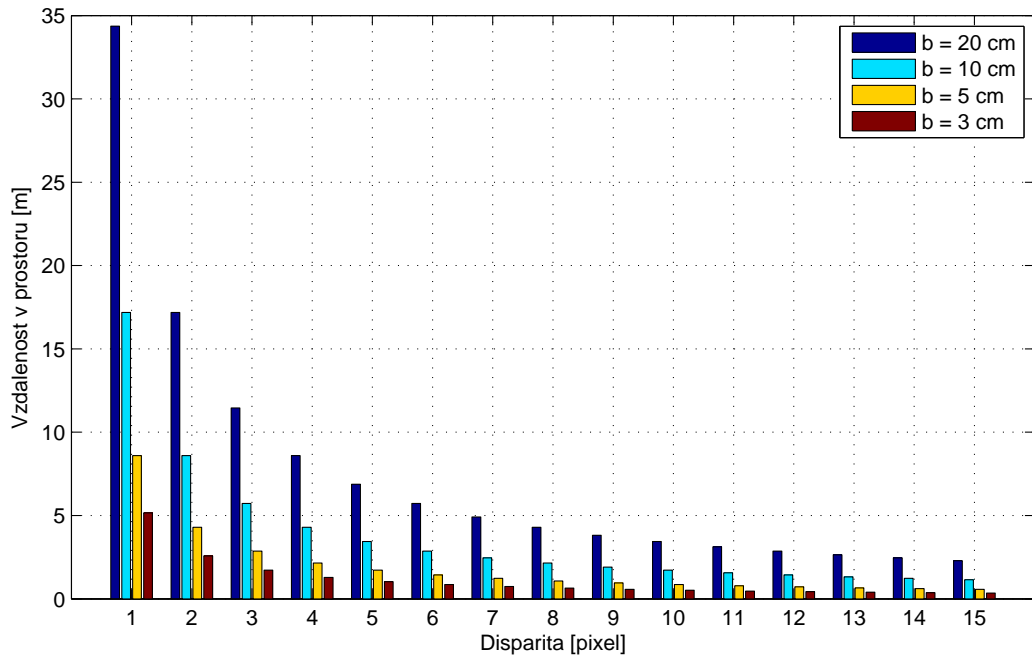


Obrázek 5.9: V levé části jsou rektifikované stereoskopické snímky a vpravo je výsledná disparitní a hloubková mapa (hodnoty jsou uvedeny v cm).

Rozměry vývojového kitu jsou relativně malé a tudíž i vzdálenost optických středů kamer (základna b) je tomu způsobena a činí 3 cm. Na obrázku obr. 5.11 vidíme graf znázorňující závislost vzdálenosti bodu v prostoru na hodnotě disparity a velikosti základny b . Z tohoto grafu je zřejmé, že pro velikost základny 3cm a parametry kamer uvedené v kapitole 4.4 odpovídá disparitě $d_{disparity} = 1$ vzdálenost bodu v prostoru 5 m. Závislost vzdálenosti na disparitě je hyperbolická a vzdálenost 5 m (pro $b = 3\text{cm}$) je mezní hodnotou. Zároveň disparita $d_{disparity} = 1$ je hodnota, která způsobuje velkou chybu. Modely (vzducholod', vznášedlo, ...) potřebují pro svoji orientaci v prostoru spíše větší rozsah vzdáleností (do 10 m) a proto by lépe vyhovovala základna b o velikost přibližně 10 až 20 cm. Tato velikost základny by, při hodnotě disparity $d_{disparity} = 1$, dovolovala měřit vzdálenosti až do 17,2 m (pro $b = 10\text{cm}$) nebo 34,4 m (pro $b = 20\text{cm}$). Především by se však zjemnil krok měřených vzdáleností v rozsahu do 10 m. Bohužel, vývojový kit nemá možnost měnit šířku základny b .



Obrázek 5.10: Hloubková mapa odpovídající předchozímu obrázku, vykreslená ve 3D prostoru.



Obrázek 5.11: Závislost vzdálenosti bodu v prostoru na disparitě pro různé hodnoty základny b .

5.7 Optimalizace algoritmu stereovidění a výsledky

Z hlediska rychlosti zpracování jsou v tomto algoritmu klíčové funkce, jenž probíhají v cyklu aktualizace disparitní mapy:

- posun snímku (*imageShiftLeft()* nebo *imageShiftRight()*)
- rozdíl snímků (*imageDiff()*)
- filtrace diferenční mapy průměrovacím filtrem (*imageSmoothing()*)
- aktualizace disparitní mapy (*updateDisparityMap()*)

U filtrace diferenční mapy je možné provést zrychlení s využitím vypočítaných dat v předešlých krocích. Nejprve vypočteme sumu n řádkových hodnot $p(y, x)$ příslušného sloupce. Hodnoty odpovídající jednotlivým sloupcům výšky n si uložíme do pomocného 1D pole o stejné velikosti jako šířka datového pole. Z pomocného pole sečteme n prvních hodnot a vydělíme počtem prvků průměrovacího filtru n^2 . Získáme první hodnotu $nValue$

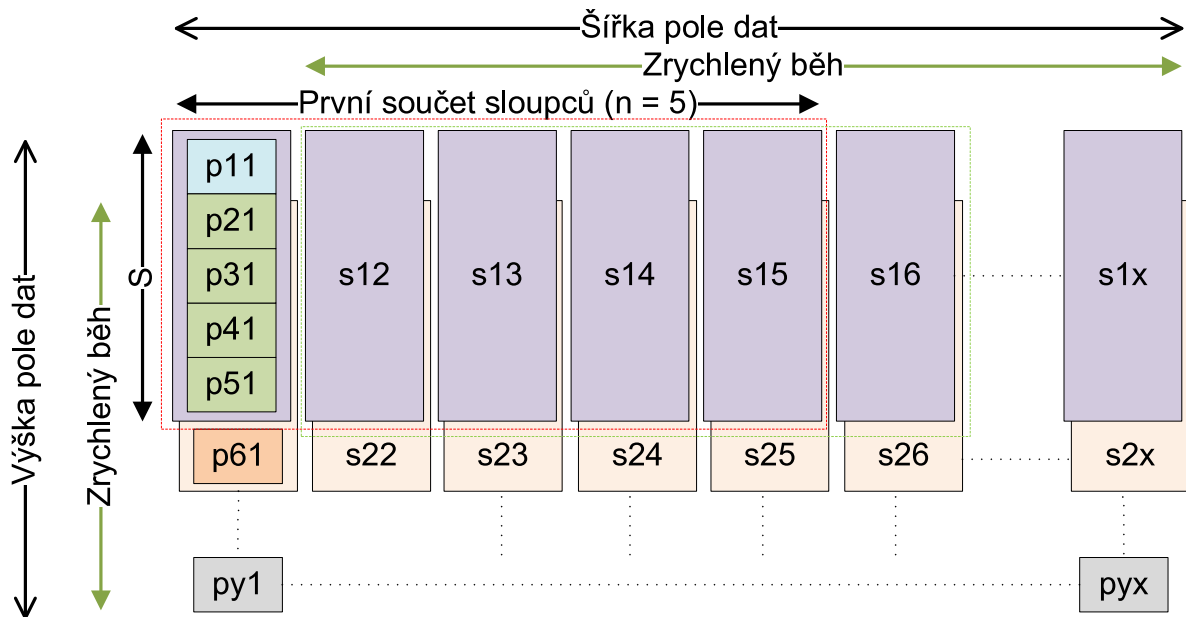
filtrovaných dat (viz. červený rámeček na obrázku obr. 5.12). Až do tohoto kroku je výpočet stejný jako při použití původního algoritmu průměrovacího filtru (resp. toto platí pro všechny hodnoty prvního sloupce vypočítané průměrovacím filtrem). Ostatní hodnoty v prvním řádku získáme tímto cyklem:

```
for (j = 0; j < xWidth - n; j++) {
    nValue += (row[j + n] - row[j]);
    *memIndexSmooth++ = (unsigned char)((nValue / avgCoef) & 0xff);
}
```

Od předchozího celkového součtu ($nValue$) je odečtena hodnota jeho prvního sloupce a zároveň se přičte hodnota sloupce následujícího (např. $nValue - s_{11} + s_{16}$ pro $n = 5$, viz. zelený rámeček na obrázku obr. 5.12). Vydělením výsledku počtem průměrovaných hodnot $avgCoef = n^2$ získáme ostatní řádkové hodnoty průměrovacího filtru. Podobná úvaha byla provedena i pro aktualizaci sloupců pomocného pole. Aktualizace se provede v tomto cyklu:

```
for (k = 0; k < xWidth; k++) {
    row[k] += *(memIndexOrig + (i + n)*xWidth + k) - *(memIndexOrig + i*xWidth + k);
}
```

Proměnná i vyjadřuje aktuální řádek. Od předchozí hodnoty sloupce se odečte hodnota prvního sloupcového prvku a přičte se hodnota následujícího sloupcového prvku (např. $s_{21} = s_{11} - p_{11} + p_{61}$ pro $n = 5$, viz. obr. 5.12). Opakováním uvedených cyklů získáme kompletní filtrované datové pole průměrovacím filtrem.



Obrázek 5.12: Princip zrychleného výpočtu hodnot průměrovacího filtru.

Funkce posunu snímků, výpočtu diferenční mapy, aktualizace disparitní mapy, rektifikace snímků, ap. byly optimalizovány především s využitím rychlých vnitřních pamětí procesoru a vlastností překladače.

Výrazného urychlení zpracování stereoskopických snímků dosáhneme využitím rychlé vnitřní paměti L1. Tato paměť pracuje na stejné frekvenci jako jádra procesoru. Rozlišení snímků scény bylo zvoleno na 160 x 120 pixelů (pro 8-bitové obrazové data v tomto rozlišení potřebujeme 19,2 kB paměti). L1 paměť jednoho jádra nám svojí velikostí 64 kB rozdělenou do 2 bank (A a B) dovoluje vytvořit dva „obrazové“ buffery. Pro paralelní zpracování disparitní mapy má druhé jádro stejně velkou a stejně koncipovanou L1 paměť, takže zpracování probíhá stejnou rychlostí a nezávisle. V tabulce tabulka 5.1 a tabulka 5.2 je uveden přehled některých testovaných funkcí, pro které byly použita data uloženy v paměti L1, L2 nebo L3¹¹.

Algoritmus stereovidění byl nejdříve testován na jednom jádru bez optimalizace kódu překladačem, s neoptimalizovanými funkcemi výpočtu disparitní mapy a využíval paměť typu L3. Doba zpracování byla přibližně 1,25 s pro jednu kompletní disparitní mapu. Optimalizovaný algoritmus dosahuje na obou jádrech rychlosti zpracování necelých 10 snímků za vteřinu. Doba zpracování jedné kompletní disparitní mapy je 109,3 ms. Optimalizací bylo dosaženo více než 10-ti násobného zrychlení.

Paměť	vylepšená <i>imageSmoothing()</i>	<i>imageSmoothing()</i>	<i>_conv2d5x5_gen()</i>
Bez optimalizace kódu pomocí překladače			
L1	4,255 ms	12,86 ms	0,954 ms
L2	7,259 ms	19,125 ms	7,856 ms
Optimalizace kódu pomocí překladače			
L1	0,513 ms	1,627 ms	0,954 ms
L2	1,037 ms	7,859 ms	0,954 ms

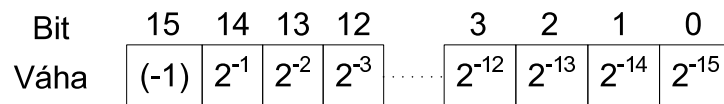
Tabulka 5.1: Přehled rychlostí funkcí pro průměrovací filtr (data 160 x 120 pixelů).

¹¹Nastavení DSP procesoru: frekvence CPU 500 MHz, frekvence externí paměti 125 MHz. Průměrovací filtr: $n = 5$. $MAX_DISP = 15$.

Paměť	<i>imageShiftLeft()</i>	<i>imageDiff()</i>	<i>updateDisparityMap()</i>
Bez optimalizace kódu pomocí překladače			
L1 a L2	0,935 ms	2,124 ms	2,69 ms
Optimalizace kódu pomocí překladače			
L1 a L2	0,020 ms	1,686 ms	0,694 ms
L1 a L3	0,036 ms	1,786 ms	1,619 ms

Tabulka 5.2: Přehled rychlostí dalších použitých funkcí v cyklu získání disparitní mapy (data 160 x 120 pixelů).

Funkce 2D konvoluce *conv2d()* nebo speciální případ konvoluce s jádrem velikosti 5 x 5 *_conv2d5x5_gen()* je k dispozici v knihovně *Visual DSP++ 5.0*. Tyto funkce jsou optimalizovány pro daný procesor v jazyku assembler, ale vstupní a výstupní data jsou ve formátu *fract16* (případně *fract32*). Formát *fract16* je znázorněn na obrázku obr. 5.13. Nevýhoda tohoto formátu dat je pro navržený algoritmus stereovidění v tom, že jsou 16-bitová a koncipovaná jako desetinné číslo v intervalu $(-1; 1)$, ve tvaru „1.15“ (MSB je znaménkový bit a ostatní bity jsou váhové). Obrázek se tedy musí nejdříve do tohoto formátu konvertovat a vyžaduje 2-krát více paměti (pro rozlišení 160 x 120 již nelze vytvořit takto velké buffery v L1 paměti). Funkce *_conv2d5x5_gen()*, optimalizována pro obecnou 2D konvoluci, je v konečném důsledku pomalejší, než vylepšená *imageSmoothing()*. Z těchto důvodů není v řešení použita.



Obrázek 5.13: Struktura formátu dat typu *fract16*.

Ještě vyšší optimalizace kódu C je možné dosáhnout možnostmi překladače. Ten ve vlastnostech překládaného projektu umožňuje měnit poměr mezi velikostí zkompilevaného souboru a optimalizací rychlosti. Pro algoritmus stereovidění byla zvolena optimalizace rychlosti.

Kapitola 6

Závěr

V úvodu této diplomové práce byly popsány vybrané metody měření hloubkové mapy pro orientaci v prostoru, resp. detekci překážek, se zaměřením na metodu stereovidění a PMD kamery. Jako vhodná byla pro modely mobilních zařízení zvolena metoda stereovidění. Princip spočívá v získání dvou stereoskopických snímků se známými parametry, z nichž lze vhodným postupem získat disparitní a hloubkovou mapu.

Modely mobilních zařízení jsou závislé na spotřebě elektrické energie a hmotnosti palubní elektroniky, kterou nesou. Zároveň je nutné brát ohled na dostatečnou rychlost zpracování dané metody, aby modely měly možnost včas reagovat na podmínky v okolním prostoru. Tyto požadavky byly brány v úvahu při výběru HW části. Pro řešení úlohy stereovidění a její algoritmizaci byl vybrán signálový dvoujádrový procesor s pevnou řádovou čárkou a nízkou spotřebou ADSP-BF561 od firmy Analog Devices. ADSP-BF561 obsahuje dvě identická jádra, každé pracuje až na frekvenci 600 MHz a disponuje 100 kB vnitřní L1 paměti pro rychlé zpracování dat. Dále obsahuje 128 kB L2 paměti sdílené oběma jádry a periferie dovolující komunikovat s okolním světem (rozhraní UART, SPI) a CMOS kamerami (dvě paralelní sběrnice PPI a GPIO). K tomuto signálovému procesoru byl dále vybrán vhodný vývojový kit od firmy Bluetechnix, skládající se z vývojové desky DEV-BF5xxDA-Lite, procesorové jednotky CM-BF561 s procesorem ADSP-BF561 a rozšiřujícího kamerového modulu EXT-BF5xx-CAM. Tento modul byl navíc doplněn 2 kamerami OV2640FSL.

Prvotní řešení jednoduchých algoritmů založených na rozdílu stereoskopických snímků, filtraci takto vzniklé diferenční mapy a kontroly konzistence L-R a R-L disparitních map bylo úspěšně provedeno a ověřeno v prostředí *Matlab 7.1*. Stejně tak výpočet kalibračních koeficientů jednotlivých kamer a rektifikačních koeficientů stereoskopických snímků byl proveden v prostředí *Matalab* pomocí *Camera Calibration Toolboxu*. Výsledné řešení

stereovidění bylo pro ADSP-BF561 kódováno ve vývojovém prostředí *Visual DSP++ 5.0*. Byly zvládnuty kroky získání obrazů z obou použitých kamer, jejich převod z YUV formátu na intenzitní obraz, rektifikace pro následné optimalizované hledání korespondencí mezi snímky a výpočet disparitní mapy. Optimalizací použitých funkcí, využitím rychlých L1 (L2) pamětí procesoru ADSP-BF561 a optimalizačních metod nabízených překladačem vývojového prostředí *Visual DSP++ 5.0*, bylo dosaženo rychlosti 6 snímků za vteřinu (6 fps) pro jednojádrovou aplikaci a necelých 10 fps pro dvoujádrovou aplikaci v rozlišení 160 x 120 pixelů a pro maximální hodnotu prohledávané disparity 15 pixelů.

Výsledky výpočtů disparitních map použitého algoritmu jsou závislé na množství textury ve scéně a v tomto ohledu nejsou příliš robustní, při malém množství textury nemusejí odpovídat realitě (viz. obrázky v příloze A). Vylepšení robustnosti by mohlo být provedeno využitím varianční mapy intenzitního obrázku a variační mapy výsledné disparitní mapy pro odstranění špatných hodnot disparit málo kontrastních ploch. Dalším řešením by jistě bylo i užití sofistikovanějších algoritmů stereovidění, které by však potřebovaly více výpočetního výkonu/času. Možností je také promítání „aktivní textury“ do scény, z čehož však vyplývá vyšší spotřeba.

Literatura

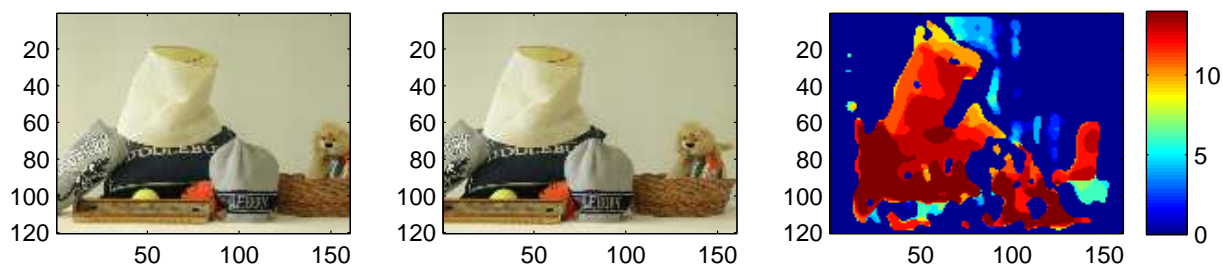
- [1] HLAVÁČ, V., SEDLÁČEK, M. (2003). *Zpracování signálů a obrazů (2. přepracované vydání)*. Vydavatelství ČVUT 2003
- [2] SONKA, M., HLAVÁČ, V., BOYLE, R. (1999). *Image Processing, Analysis, and Machine Vision, 2nd edition*. Thomson Learning, Toronto, April 2007, 821 p, ISBN 049508252X (2nd edition Brooks/Cole, Pacific Grove, CA, 1999, 1st edition Chapman & Hall, London 1993,).
- [3] TON, O. (2006). *Orientace v prostředí*. Bakalářská práce, Praha: ČVUT, Katedra řídicí techniky.
- [4] WEBOVÉ STRÁNKY KATEDRY ŘÍDICÍ TECHNIKY ČVUT-FEL PRAHA (2008). *Prezentace projektů a témat DP, BP, PMI* (<http://dce.felk.cvut.cz/prezentace/>)
- [5] WEBOVÉ STRÁNKY VÝROBCŮ PMD KAMER - (<http://www.pmdtec.com>), (<http://www.swissranger.ch>), (<http://www.canesta.com>)
- [6] HW.CZ (2007). *PMD senzor a 3D měření vzdálenosti - 1. a 2. část*. (<http://www.hw.cz/externi/1642/>)
- [7] WIKIPEDIA.ORG (2008). *Programmable logic device*. Category: Gate arrays (http://en.wikipedia.org/wiki/Programmable_logic_device)
- [8] BTDI - WWW.DSPDESIGNLINE.COM (2007). *A Survey of Mainstream DSP Processors* (<http://www.dspdesignline.com/howto/198800216>)
- [9] WEBOVÉ STRÁNKY BDTI (2007). *BDTI DSP Kernel BenchmarksTM Results* (<http://www.bdti.com/bdtimark/BDTImark2000.htm>)
- [10] JEAN-YVES BOUGUET (2008). *Camera Calibration Toolbox for Matlab*. (http://www.vision.caltech.edu/bouguetj/calib_doc/)

- [11] SHAWN LANKTON (2008). *3D Stereo Disparity*.
⟨<http://www.mathworks.de/matlabcentral/fileexchange/19406>⟩
- [12] ANALOG DEVICE (2007). *Blackfin Embedded Symmetric Multiprocessor ADSP-BF561 - Rev. C* ⟨<http://www.analog.com>⟩
- [13] ANALOG DEVICE (2007). *ADSP-BF561 Blackfin® Processor Hardware Reference - Revision 1.1* ⟨<http://www.analog.com>⟩
- [14] ANALOG DEVICE (2008). *VisualDSP++ 5.0 Device Drivers and System Services Manual for Blackfin Processors - Revision 3.2* ⟨<http://www.analog.com>⟩
- [15] ANALOG DEVICE (2008). *VisualDSP++ 5.0 C/C++ Compiler and Library Manual for Blackfin Processors - Revision 5.1* ⟨<http://www.analog.com>⟩
- [16] ANALOG DEVICE (2008). *Blackfin Processor Programming Reference - Revision 1.3* ⟨<http://www.analog.com>⟩
- [17] ANALOG DEVICE (2007). *VisualDSP++ 5.0 Users Guide - Revision 3.0* ⟨<http://www.analog.com>⟩
- [18] ANALOG DEVICE (2008). *VisualDSP++ 5.0 Linker and Utilities Manual - Revision 3.1* ⟨<http://www.analog.com>⟩
- [19] WEBOVÉ STRÁNKY FIRMY BLUETECHNIX (2008). *Blackfin family tools* ⟨<http://www.bluetechnix.com>⟩
- [20] WEBOVÉ STRÁNKY TEXAS INSTRUMENT, INC. (2007/2008). *DSP Design Support* ⟨<http://www.ti.com>⟩
- [21] WEBOVÉ STRÁNKY ANALOG DEVICES, INC. (2008). *Embedded processing and DSP* ⟨<http://www.analog.com/embedded-processing-dsp/processors/en/index.html>⟩
- [22] H. HIRSCHMÜLLER AND D. SCHARSTEIN. (2007). *Evaluation of cost functions for stereo matching..* In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), Minneapolis, MN, June 2007. ⟨<http://vision.middlebury.edu/stereo/data/>⟩
- [23] WIKIPEDIA.ORG (2008). *YCbCr*. Kategorie: Barevné prostory ⟨<http://cs.wikipedia.org/wiki/YCbCr>⟩

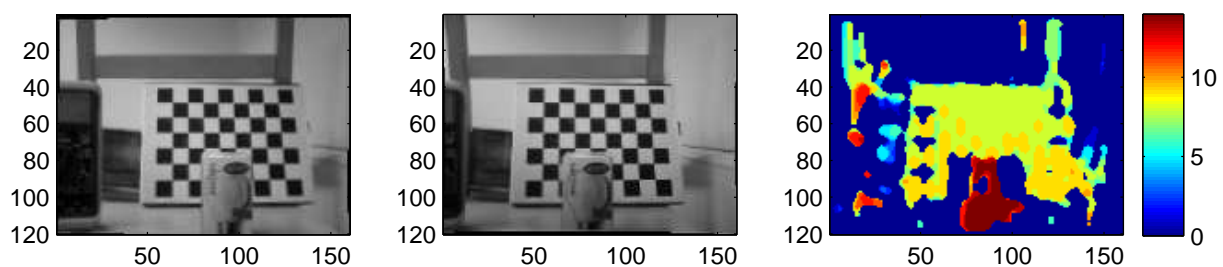
Příloha A

Přílohy

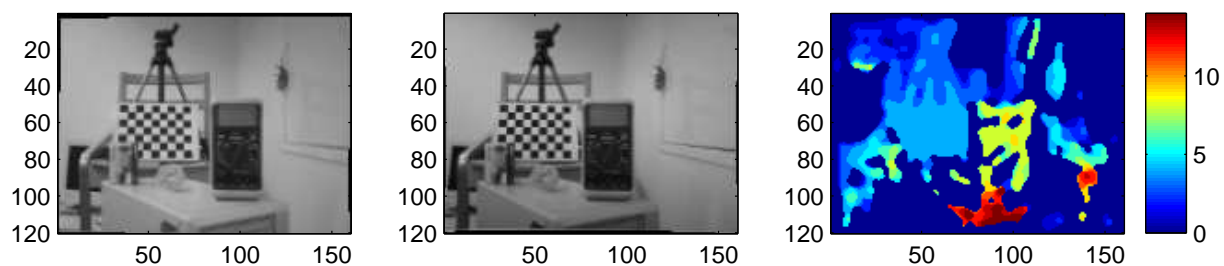
Na začátku této přílohy jsou zobrazeny výsledky disparitních map použitého algoritmu stereoidění. Barevné snímky jsou použity z knihovny rektifikovaných obrázků [22]. „Černobílé“ snímky jsou reálné scény pořízené kamerami vývojového kitu.



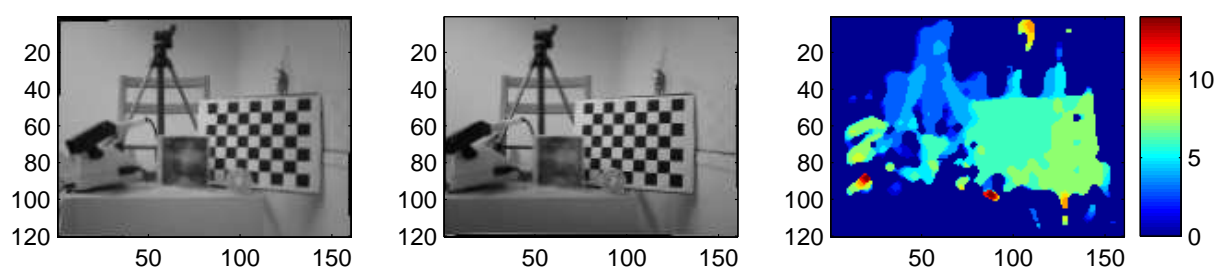
Obrázek A.1: Disparitní mapa pro dataset „Midd1“.



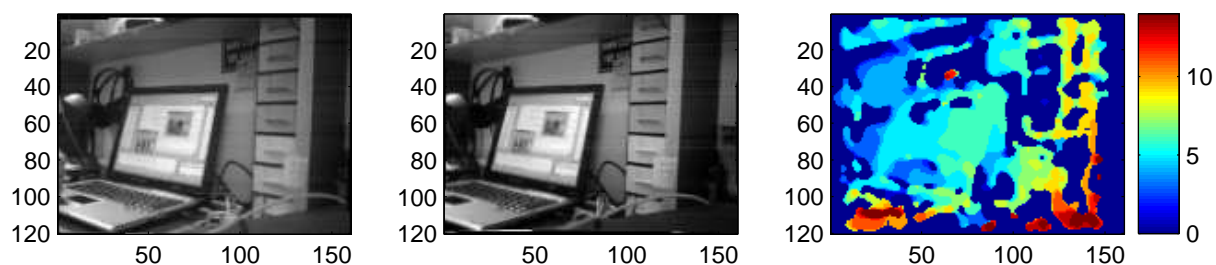
Obrázek A.2: Disparitní mapa scény č. 1 pořízené z kamer.



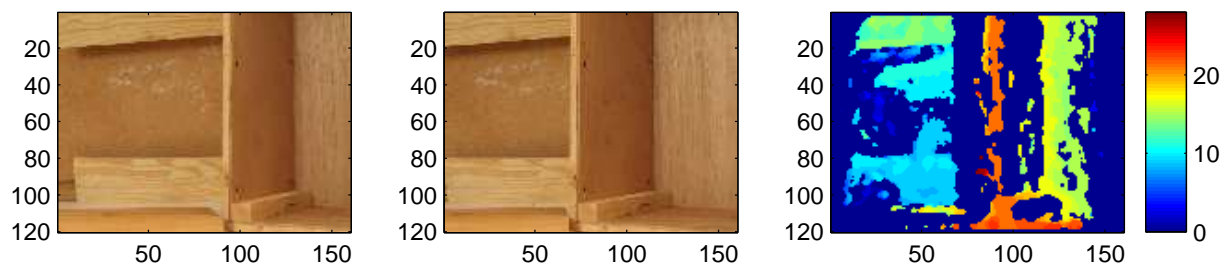
Obrázek A.3: Disparitní mapa scény č. 2 pořízené z kamer.



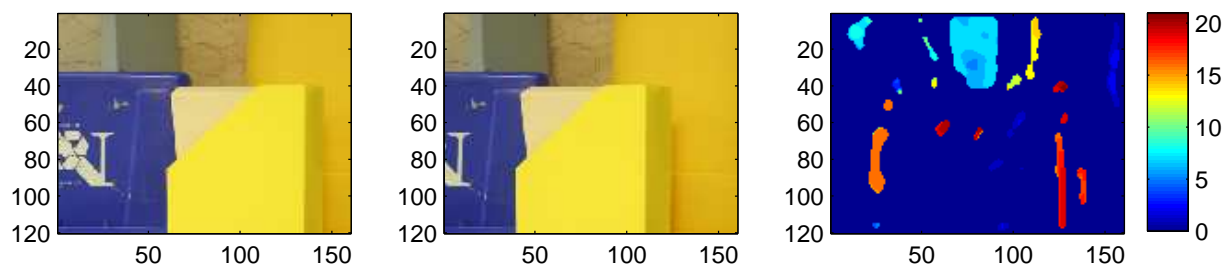
Obrázek A.4: Disparitní mapa scény č. 3 pořízené z kamer.



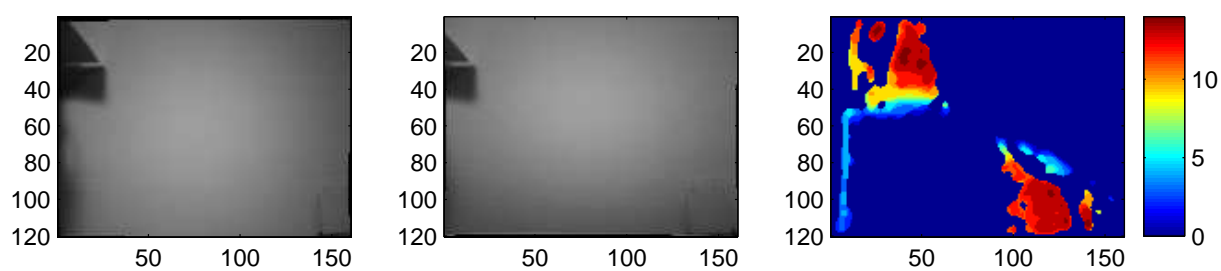
Obrázek A.5: Disparitní mapa scény pracovního stolu s notebookem.



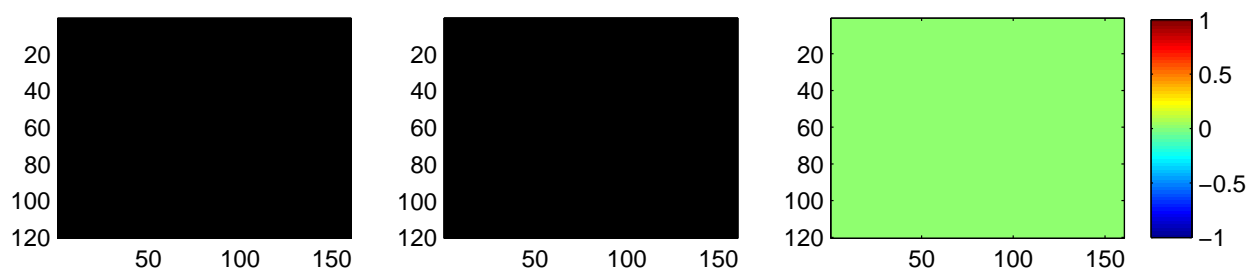
Obrázek A.6: Disparitní mapa pro dataset „Wood1“.



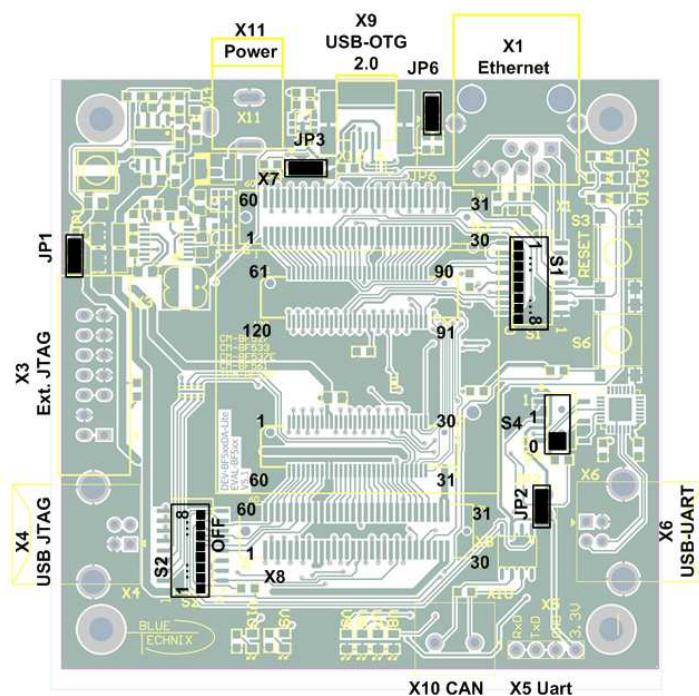
Obrázek A.7: Disparitní mapa pro dataset „Plastic“.



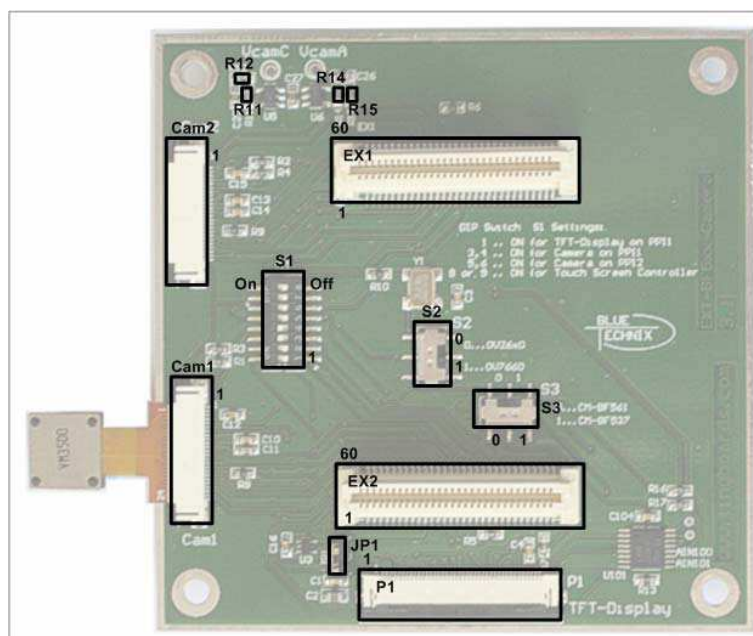
Obrázek A.8: Disparitní mapa pro scénu „bílá“ zdi.



Obrázek A.9: Disparitní mapa pro uměle vytvořené černé obrázky.



Obrázek A.10: Rozmístění a očíslování konektorů na desce plošných spojů DEV-BF5xxDA-Lite.



Obrázek A.11: Rozmístění a očíslování konektorů na rozšiřující desce EXT-BF5xx-CAM.

Číslo pinu	Název signálu	Vstup(I)/Výstup(O)/napájení
1	RSCLK0 / PF28	I/O
2	DR0PRI	I
3	TSCLK0 / PF29	I/O
4	DT0PRI / PF18	I/O
5	PF11(Clk_out) *	I/O
6	PF9	I/O
7	PF7 / SPISEL7 / TMR7	I/O
8	PF5 / SPISEL5 / TMR5	I/O
9	Vin 3V3	napájení
10	Vin 3V3	napájení
11	PPI1D0	I/O
12	PPI1D2	I/O
13	PPI1D4	I/O
14	PPI1D6	I/O
15	PPI1D8 / PF40	I/O
16	PPI1D10 / PF42	I/O
17	PPI1D12 / PF44	I/O
18	PPI1D14 / PF46	I/O
19	PPI1SYNC3	I/O
20	PPI1SYNC1 / TMR8	I/O
21	PF3 / SSEL3 / TM3	I/O
22	PF1 / SPISEL1 / TMR1	I/O
23	RX / PF27	I/O
24	MOSI	I/O
25	SCK	I/O
26	nABE2 / SDQM2	O
27	ARDY	I
28	TCK	I
29	TDI	I
30	$\overline{\text{TRST}}$	I

Tabulka A.1: Vývody konektoru X1 procesorové jednotky CM-BF561.

Číslo pinu	Název signálu	Vstup(I)/Výstup(O)/napájení
31	$\overline{\text{EMU}}$	O
32	TMS	I
33	TDO	O
34	$\overline{\text{AMS3}}$	O
35	$\overline{\text{ABE1}}/\text{SDQM1}$	O
36	$\overline{\text{ABE0}}/\text{SDQM0}$	O
37	MISO	I/O
38	TX / PF26	I/O
39	PF0/SPISS/TMR0	I/O
40	PF2/SSEL2/TMR2	I/O
41	PPI1CLK	I
42	PPI1SYNC2 / TMR9	I/O
43	PPI1D15 / PF47	I/O
44	PPI1D13 / PF45	I/O
45	PPI1D11 / PF43	I/O
46	PPI1D9 / PF41	I/O
47	PPI1D7	I/O
48	PPI1D5	I/O
49	PPI1D3	I/O
50	PPI1D1	I/O
51	GND	napájení
52	GND	napájení
53	PF4/SPISEL4/TMR4	I/O
54	PF6/SPISEL6/TMR6	I/O
55	PF8	I/O
56	PF10	I/O
57	DT0SEC / PF17	O
58	TFS0 / PF16	I/O
59	DR0SEC / PF20	I
60	RFS0 / PF19	I/O

Tabulka A.2: Vývody konektoru X1 procesorové jednotky CM-BF561.

Číslo pinu	Název signálu	Vstup(I)/Výstup(O)/napájení
61	$\overline{\text{ABE3}}/\text{SDQM3}$	O
62	A3	O
63	A5	O
64	A7	O
65	A9	O
66	A11	O
67	A13	O
68	A15	O
69	PPI2SYNC1	I/O
70	PPI2SYNC2	I/O
71	PPI2D1	I/O
72	PPI2D3	I/O
73	PPI2D5	I/O
74	PPI2D7	I/O
75	PPI2D9 / PF33	I/O
76	PPI2D11 / PF35	I/O
77	PPI2D13 / PF37	I/O
78	PPI2D15 / PF39	I/O
79	GND	napájení
80	$\overline{\text{AMS1}}$	O
81	$\overline{\text{AWE}}$	O
82	NMI_0	I
83	D0	I/O
84	D2	I/O
85	D4	I/O
86	D6	I/O
87	D8	I/O
88	D10	I/O
89	D12	I/O
90	D14	I/O

Tabulka A.3: Vývody konektoru X2 procesorové jednotky CM-BF561.

Číslo pinu	Název signálu	Vstup(I)/Výstup(O)/napájení
91	D15	I/O
92	D13	I/O
93	D11	I/O
94	D9	I/O
95	D7	I/O
96	D5	I/O
97	D3	I/O
98	D1	I/O
99	$\overline{\text{RESET}}$	I
100	$\overline{\text{AOE}}$	O
101	$\overline{\text{ARE}}$	O
102	$\overline{\text{AMS2}}$	O
103	N.C	-
104	PPI2D14 / PF38	I/O
105	PPI2D12 / PF36	I/O
106	PPI2D10 / PF34	I/O
107	PPI2D8 / PF32	I/O
108	PPI2D6	I/O
109	PPI2D4	I/O
110	PPI2D2	I/O
111	PPI2D0	I/O
112	PPI2SYNC3	I/O
113	PPI2CLK	I
114	A14	O
115	A12	O
116	A10	O
117	A8	O
118	A6	O
119	A4	O
120	A2	O

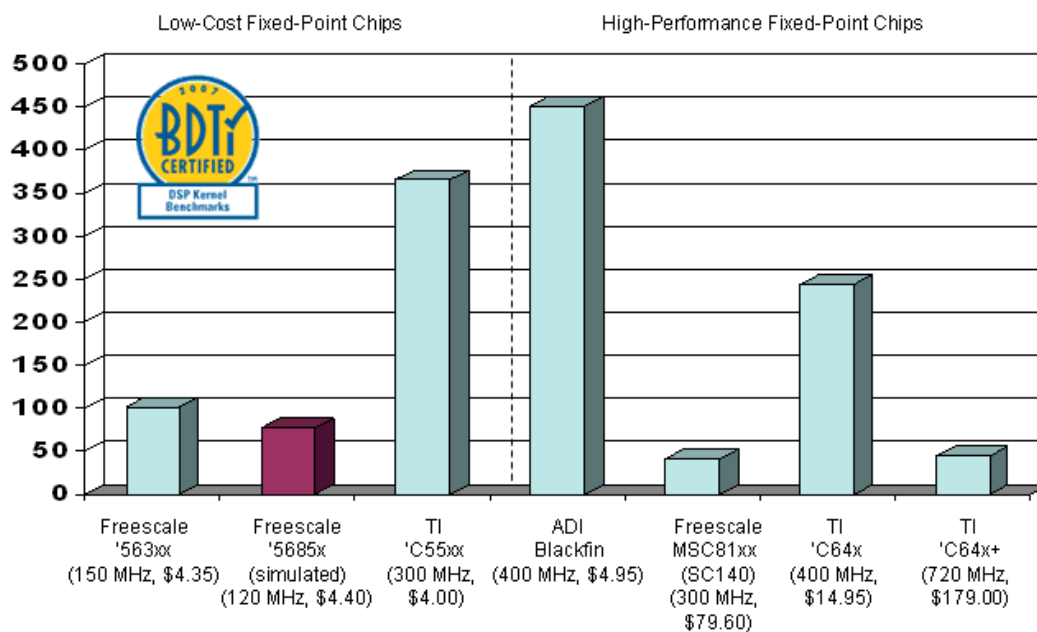
Tabulka A.4: Vývody konektoru X2 procesorové jednotky CM-BF561.

Č. pinu	Signál kamery	Kamera č. 1	Kamera č. 2
1	not connected	not connected	not connected
2	AGND	not connected	not connected
3	SIO_D	PPI1D12	PPI2D11
4	AVDD	not connected	not connected
5	SIO_C	PPI1D15	PPI2D10
6	RESET	not connected	not connected
7	VSYNC	PPI1Sy2	PPI2Sy2
8	PWDN	not connected	not connected
9	HREF	PPI1Sy1	PPI2Sy1
10	DVDD	not connected	not connected
11	DOVDD	not connected	not connected
12	D7	PPI1D7	PPI2D7
13	CamClk	not connected	not connected
14	D6	PPI1D6	PPI2D6
15	DGND	not connected	not connected
16	D5	PPI1D5	PPI2D5
17	PCLK	PPI1Clk	PPI2Clk
18	D4	PPI1D4	PPI2D4
19	D0	PPI1D0	PPI2D0
20	D3	PPI1D3	PPI2D3
21	D1	PPI1D1	PPI2D1
22	D2	PPI1D2	PPI2D2
23	D8	PPI1D8	PPI2D8
24	D9	PPI1D9	PPI2D9

Tabulka A.5: Tabulka připojení kamery č. 1 a č. 2 ke sběrnicím PPI1 a PPI2 CM-BF561.

Označení	Barva	Funkce
V1	Červená	Full Duplex (Ethernet)
V2	Zelená	Activity (Ethernet)
V3	Zelená	100MB Speed LED (Ethernet)
V5	Zelená	Flag0 (Debug Agent)
V6	Zelená	Flag1 (Debug Agent)
V7	Červená	Monitor (Debug Agent)
V8	Zelená	Pr. done (Debug Agent)
V9	Zelená	GPIO (PF43)
V10	Červená	GPIO (PF42)
V14	Žlutá	napájení

Tabulka A.6: Funkce jednotlivých LED diod na vývojové desce DEV-BF5xxDA-Lite.



Obrázek A.12: Výsledky fixed-point DSP procesorů z dané řady v testu BDTmark2000TM[9] v závislosti na poměru cena/výkon (cost-effective). Vyšší výsledek je lepší.

Vnitřní (intrinsic) parametry levé kamery (název/proměnná v Matlabu):

Ohnisková vzdálenost:

$$fc_left = [175.61085 \quad 175.20868] \pm [2.10832 \quad 1.99907]$$

Hlavní bod:

$$cc_left = [77.88728 \quad 59.86497] \pm [2.68535 \quad 2.30244]$$

Zkosení os (v tomto případě se neuplatňuje):

$$\alpha_c_left = [0.00000] \pm [0.00000] \quad 90^\circ \pm 0$$

Zkreslení:

$$kc_left = [0.08348 \quad -0.19002 \quad -0.00265 \quad -0.00253 \quad 0.00000] \\ \pm [0.07091 \quad 0.62399 \quad 0.00528 \quad 0.00670 \quad 0.00000]$$

Vnitřní (intrinsic) parametry pravé kamery (název/proměnná v Matlabu):

Ohnisková vzdálenost:

$$fc_right = [175.67880 \quad 174.95999] \pm [2.11958 \quad 1.98107]$$

Hlavní bod:

$$cc_right = [79.61816 \quad 60.10274] \pm [2.51903 \quad 2.01111]$$

Zkosení os (v tomto případě se neuplatňuje):

$$\alpha_c_right = [0.00000] \pm [0.00000] \quad 90^\circ \pm 0$$

Zkreslení:

$$kc_right = [0.14756 \quad -0.67056 \quad 0.00169 \quad 0.00268 \quad 0.00000] \\ \pm [0.05882 \quad 0.29416 \quad 0.00386 \quad 0.00731 \quad 0.00000]$$

Vnější (extrinsic) parametry – pozice levé kamery vůči pravé:

Vektor rotace:

$$om = [-0.00977 \quad -0.00756 \quad 0.00642] \pm [0.01478 \quad 0.01911 \quad 0.00095]$$

(rotační matici 3 x 3 získáme příkazem $R = \text{rodrigues}(om)$ v Matlabu)

Vektor translace:

$$T = [-30.27065 \quad 0.01818 \quad -0.98434] \pm [0.50370 \quad 0.45854 \quad 3.04514]$$

Obrázek A.13: Kalibrační a rektifikační parametry kamer.

Příloha B

Obsah přiloženého CD

K této práci je přiloženo CD, na kterém jsou uloženy zdrojové kódy, katalogové listy, manuály, doplňkový SW a tato diplomová práce ve formátu PDF.

Seznam hlavních adresářů:

- *BF561_code*: zdrojový kód pro DSP procesor ADSP-BF561
- *BF561_documents*: manuály a katalogové listy k DSP procesoru ADSP-BF561
- *Bluetechnix*: manuály, katalogové listy, výkresy k vývojovému kitu a kamerám
- *Diploma_thesis*: text této DP v PDF formátu
- *DSP_processors*: informace o DSP procesorech
- *Matlab*: zdrojové kódy a data pro prostředí Matlab, Camera Calibration Toolbox
- *VisualDSP*: manuály a update k vývojovému prostředí Visual DSP++ 5.0