ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ v PRAZE

Fakulta elektrotechnická
Katedra Kybernetiky

Detekce změn v 3D MRI datech pomocí registrace

Diplomová práce

Martin Hrubý

Vedoucí diplomové práce: Dr.Ing. Jan Kybic, Ph.D.
Oponent diplomové práce: Mgr. Václav Krajíček
Studijní program: Biomedicínské inženýrství
Obor: Kybernetika a měření

České vysoké učení technické v Praze - Fakulta elektrotechnická

Katedra kybernetiky ***Školní rok:*** 2006/2007

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Martin Hrubý

Obor: Biomedicínské inženýrství

Název tématu: Detekce změn v 3D MRI datech pomocí registrace

Zásady pro vypracování:

1. Konzultujte s lékaři FN Homolka jejich přesné požadavky a získejte od nich vhodná 3D MRI data.
2. Za použití knihovny ITK navrhněte a naimplementujte program pro čtení těchto dat, odstranění značek stereotaktického rámu pomocí matematické morfologie a registrace vždy dvojice 3D obrazů za účelem zjištění změn mezi nimi.
3. Navrhněte a implementujte též vhodnou vizualizaci pro snazší analýzu výsledků lékaři.

Seznam odborné literatury:

[1] Brown, L.G.: *A Survey of Image Registration Techniques*. Department of Computer Science Columbia University New York, NY 10027 January 12, 1992

[2] Thévenaz, P.; Ruttimann, U.E.; Unser, M.: *A Pyramid Approach to Subpixel Registration Based on Intensity*. IEEE Transactions on Image Processing, vol. 7, no. 1, pp. 27-41, January 1998.

[3] Van den Elsen P.A.; Pol, E.J.D. and Viergever, M.A. : *Medical Image Matching-A Review with Classification*. IEEE Eng. In Medicine and Biology, vol. 12, pp. 26-39, 1993.

[4] Zitova B.; Flusser J.: *Image Registration Methods: a Survey*. Image and Vision Computing, Volume 21, Number 11, October 2003

Vedoucí diplomové práce: Ing. Jan Kybic, Ph.D.

Termín zadání diplomové práce: zimní semestr 2006/2007

Termín odevzdání diplomové práce: leden 2008

L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

prof. Ing. Zbyněk Škvor, CSc.
děkan

V Praze dne 19.02.2007

Prohlášení

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

V Praze dne 21. 1. 2009 ..... Martin Hanf

Podpis

Poděkování

Na tomto místě chci poděkovat všem, kteří mi byli nápomocni při tvorbě této práce. Zvláštní poděkování patří Dr.Ing. Janu Kybicovi, Ph.D., vedoucímu mé diplomové práce, za trpělivost a ochotu. Děkuji také Ing. Jitce Šemnické z pracoviště Stereotaktické radiochirurgie nemocnice Na Homolce za poskytnutá obrazová data pro experimenty. Nemenší dík poté mé rodině a nejbližšímu okolí, jež mi bylo oporou i inspirací.

Anotace

Práce se zabývá dvěma úkoly – odstranění stereotaktického rámu z obrazu a registrace dvou 3D obrazů. Jedná se o obrazy získaných vyšetřením pacienta pomocí MRI a uložených ve formátu DICOM.

První úloha je řešena pomocí prahování, matematické morfologie a rozrůstajícího se prostoru.

Druhá část je řešena metodou víceúrovňové registrace za použití gradientního optimalizátoru s pravidelnou délkou kroku.

Implementace obou algoritmů je provedena v C++ s využitím knihovny ITK.

Klíčová slova

MRI – Registrace – Segmentace - Stereotaktický rám – DICOM - C++ - ITK

Annotation

The diploma thesis is solving two problems – deleting stereotactic frame from image and registration two 3D images. This images are taken by MRI and save as DICOM image.

For the solution of the first part we use tresholding, mathematic morfology a region growing method.

In second task there is used multi-resolution registrations with gradient optimalizer with regular step.

Both algorithms are implemented by C++ with help of image processing library ITK.

Keywords

MRI – Registration – Segmentation – Stereotactic Frame – DICOM – C++ - ITK

Obsah

1. Úkol

Úkolem této práce je navrhnout a vytvořit jednoduchý program pro registraci 3D obrazů získaných z magnetické rezonance. Práce vychází ze zadání oddělení Stereotaktické radiochirurgie nemocnice Na Homolce, kde je často potřeba registrovat vyšetření s časovým odstupem.

Vstupem jsou obrazy z vyšetření magnetickou rezonancí získané v různých časech, v takovém případě není pacient skenován ve stejné poloze. Pro porovnání a zhodnocení léčby je třeba obrazy jednotlivých vyšetření překrýt. Pro zjištění parametrů transformace obrazu z jednoho na druhý je vhodné použít výpočetní techniku.

Nejprve je třeba obrazy předzpracovat, zbavit obraz značek stereotaktického rámu. Tento rám, který je souřadnicovým systémem pro plánování operací pomocí Leksellova gamma nože, je zobrazen jen na některých vyšetřeních. Leksellův gamma nůž má k dispozici oddělení Stereotaktické radiochirurgie nemocnice Na Homolce a umožňuje zničit zhoubné bujení hluboko uvnitř v mozku bez nutnosti nebezpečné operace hlavy.

Po zjištění transformace je možné přenést z jednoho vyšetření do druhého obraz stereotaktického rámu. Tím se pro obsluhu zajistí lepší orientace v obraze a stejný souřadnicový systém v obou porovnávaných obrazech.

Algoritmus by měl najít optimum mezi rychlostí a přesností registrace.

Výsledkem algoritmu registrace je transformace, která transformuje jeden ze vstupních obrazů, aby lícoval s druhým obrazem. Tato transformace obsahuje parametry pro posunutí (translaci), rotaci (otočení) a změnu měřítka.

Implementaci algoritmu provést v jazyce C++ za použití volně dostupných knihoven sloužících ke zpracování a zobrazení obrazů. Pro vlastní zpracování obrazů použít knihovnu ITK, která obsahuje velmi pružný framework pro registraci obrazů a mnoho funkcí pro segmentaci a filtraci. Pro návrh uživatelského rozhraní použít knihovnu FLTK a pro zobrazení dat využít knihovny VTK, případně podobné knihovny.

2. Úvod

2.1. MRI

MRI (Magnetic Resonance Imaging)[8], [9] slouží k získání obrazu vnitřního prostředí pacienta. Nepoužívá škodlivé rentgenové paprsky, ale využívá magnetické a elektromagnetické vlnění v kmitočtové oblasti rádiových vln. Zde je rentgen nahrazen silným magnetem, který vytváří homogenní pole. Tím získáváme vyšetřovací metodu, kde již není pacient vystaven záření s vedlejšími účinky.

Magnetická rezonance je založena na principu měření vychýlení os protonů v atomech vodíku vázaných v molekulách vody. Z toho důvodu je to metoda vhodná pro vyšetření měkkých tkání a zcela nevhodná pro vyšetření kostí.

Omezením je silné magnetické pole, které brání vyšetřit pacienty mající kovové protézy a jiná protetika. Přes výše zmíněné problémy je to dnes jedna z nejvíce používaných zobrazovacích metod. Od 70. let 20. století velmi postoupila dopředu. Roku 2003 dostali Nobelovu cenu P.Lauterbur a P. Mansfield za jejich přínos v oblasti MRI [9].

MRI má mnoho různých aplikací, z nichž například fMRI – funkční magnetická rezonance dovoluje sledovat aktivitu tkání a tak vyhodnocovat například, které konkrétní části mozku obsluhují konkrétní orgány.

Další bouřlivě rozvíjející se část je MRI spektroskopie, která může být převratem v diagnostice.

2.2. Leksellův gamma nůž

Leksellův gamma nůž [1],[7] je lékařský přístroj, který nahrazuje klasickou operaci při odstraňování nitrolebních lézí (nádorů, cévních malformací). Jde o radionuklidový ozařovač, jehož zdrojem gama záření je radionuklid ^{60}Co.

Využitím gamma nože k léčebným účelům se zabýval švédský neurochirurg Lars Leksell v 50. letech. První Leksellův gamma nůž pro experimentální účely byl instalován ve Stockholmu v 60. letech. V roce 1974 byl ve Stockholmu uveden do provozu druhý Leksellův Gama nůž s 201 zdroji záření, určený pro léčbu.

Leksellův gamma nůž se skládá ze tří hlavních jednotek:

- Radiační jednotka (ukrývající 201 zdrojů gama záření sbíhajících se do jednoho ohniska), se čtyřmi vyměnitelnými kolimačními helmicemi a léčebným lůžkem.
- Leksellův stereotaktický rám (Leksellův koordinátový rám), který poskytuje souřadnicový systém pro plánování operace.
- Plánovací systém.

Leksellův gamma nůž je vybaven čtyřmi kolimačními helmicemi, které vytvářejí průměr ohniska 4, 8, 14 a 18 mm.

Neurochirurgové společně s fyziky určí tzv. cílový objem (cílové místo ozáření - nádor a jeho blízké okolí) a terapeutickou dávku radiace. Počítač potom navrhne plán ozáření. Výsledkem plánování je ozařovací protokol, který udává vše potřebné k provedení ozáření. Ten obsahuje:

- počet shotů (kolik bude třeba zásahů do oblasti)
- velikost kolimátorů (jak velký průměr zvolit)
- souřadnice [x, y, z] (kde provést zásahy)

- ozařovací čas (jak dlouho ozařovat)
- počet a umístění wolframových plugů (záslepky na některý z 201 gama zdrojů)

2.3. DICOM

The Digital Imaging and Communications in Medicine, DICOM, je standart pro lékařské aplikace [1]. Umožňuje snadné sdílení dat mezi jednotlivými pracovišti nemocnice i mezi různými zdravotnickými zařízeními, případně výzkumnými ústavy. Zajišťuje, že data z mnoha lékařských zobrazovacích systémů (CT, MRI, Ultrazvuk atd.) lze jednotně skladovat, archivovat a sdílet.

Tento standard obsahuje služby jako skladování, různé druhy potvrzení při správě dat (mazání, zálohování), tisk (DICOM tiskárny vytvářející obrazy na rentgenový film) a v neposlední řadě též stanovuje formát dat v DICOM souborech sloužících k off-line prohlížení a přenosu na optická media.

V těchto souborech je kromě samotného vyšetření (např. obrazu z MRI) uvedeny i informace o pacientovi. Soubor ve formátu DICOM tyto dvě části (vyšetření a informace o pacientovi) nese v jednom souboru na rozdíl od jiného formátu Analyze, který je dělí do dvou souborů (x.img pro obrazovou část a x.hdr pro údaje o pacientovi).

Struktura DICOM souboru je potom taková, že prvních 128bitů je preambule, která je obvykle vyplněna nulami a bez informačního obsahu. Potom následují písmena D, I, C a M. Dále jsou uvedeny informace hlavičky, která mimo údajů o pacientovi obsahuje údaje o samotném vyšetření. Data jsou uspořádána do jednotlivých tagů a jsou organizovány do skupin, které spolu souvisí. Jednotlivé tagy jsou označeny číselně a jménem:

```
0008,0060,Modality=MR
0008,0070,Manufacturer=SIEMENS
0008,0080,Instituon Name=RDG HOMOLKA

0010,0010,Patient's Name=VYMYSLENE.JMENO
0010,0020,Patient ID=000000
0010,0030,Patient Date of Birth=19820815
0010,0040,Patient Sex=M
```

Nakonec hlavička obsahuje řadu informací o následné datové části (nejčastěji obrazové). Mezi tyto údaje patří:

- Použitá barevná paleta
- Styl zobrazení
- Velikost dat
  - Počet snímků
  - Počet sloupců
  - Počet řádků
  - Počet bitů na pixel

Nakonec jsou v souboru obsaženy jednotlivé snímky. V porovnání s dříve uvedeným formátem Analyze [2] mohou být tyto obrázky komprimované i ztrátovou kompresí. Tato data mohou být tedy v různých formátech jako například JPEG, ztrátový JPEG, JPEG 2000 a Run-lenght encoding (podobné jako u TIFF).

2.4. Registrace

Registrací obrazů myslíme zjištění jejich vzájemné korelace. Výsledkem registrace jsou parametry transformace, podle níž můžeme upravit jeden z obrazů tak, aby odpovídal druhému.

Registrace obrazů se používá v mnoha oblastech lidské činnosti [16], [17]. Kromě medicíny našla uplatnění ve zpracování družicových a leteckých snímků, policejních fotografiích nebo v průmyslových aplikacích.

V dnešní době můžeme porovnávat signály různé dimenze [14]. U 1D signálů, které představují časový záznam mnoha lékařských vyšetření (EKG, EEG), zjišťujeme podobnost v čase, případně periodické výskyty některých hodnot. Dále jsou tu obrazy 2D a 3D z různých zobrazovacích systémů. A v dnešní době připadá úvaha i registrace ve 4 dimenzích, kdy máme jako vstupní data časový záznam trojrozměrných obrazů [15].

Algoritmy registrace mohou být založeny na různých principech. Mohou být poloautomatické, tedy vyžadující pomoc uživatele [14]. (např.: Označení korespondujících bodů, jejichž pomocí následně algoritmus nalezne transformaci mezi obrazy.)

Programy hledající danou transformaci na lékařských snímcích hledající pomocí různých vlastností obrazu [17]:

- Intenzity pixelů
- Transformace obrazu (Fourierova)
- Příznaky (značky, povrch, šablony)

Značky mohou být intrinsické [14], [15] založené pouze na informaci obrazu pacienta, nebo mohou být extrinsické, stavějící na uměle dodaných objektech v obraze. Mezi intrinsické patří různé anatomicky významné oblasti. Mezi extrinsické patří hlavové rámy, značky na kůži nebo body dodané obsluhou. Mezi extrinsické [14] mohou být považována radiofarmaka používaná při PET, SPECT nebo kontrastní látky.

Transformace mezi obrazy můžeme rozdělit do několika skupin:

- Rigidní
- Afinní
- Projektivní
- Křivkové

Rigidní transformace obsahují pouze posunutí (translaci), otočení (rotaci) [14], [16]. Afinní zachovávají rovnoběžnost mezi zobrazeními, příkladem mohou být transformace na změnu měřítka či oddělení některé části obrazu. Projektivní zobrazení již nezachovává rovnoběžnost mezi přímkami a zvyšuje tak dimenzi prohledávaného prostoru. Křivkové transformace popisují i různé ohýbání přímek a prostoru.

Pro porovnání obrazů a toho, jak úspěšní jsme při zjišťování parametrů transformace, nám slouží různé metriky. Od jednoduchých a výpočetně nenáročných (např. střední kvadratická chyba) až po metriky, které umožňují navzájem registrovat obrazy různých modalit (např. vzájemná informace) [15], [16].

Velké firmy dodávající zobrazovací systémy do nemocnic zároveň dodávají i softwarové vybavení ke svým přístrojům. To kromě programů ovládajících přímo přístroje a software sloužící k archivaci a zobrazování naskenovaných dat zahrnuje v dnešní době i vlastní registraci obrazů. Mezi takové firmy patří Siemens Medical Solution [4], General Electric Healthcare [5], Philips Medical Systems [6] a další.

Poslední dobou je trend takový, že velké a známé firmy musí čelit konkurenci levnějších značek. Jejich kvalita výrobků je ověřená, přesto mohou některé trhy ztrácet. Většina těchto velkých firem volí strategii nezlepšovat jen hardware, ale zlepšovat analytické nástroje. Jedním takovým nástrojem jsou právě registrační algoritmy. Tato firemní řešení registrace jsou odvozena z již známých algoritmů a vylepšena do komerční podoby. Kdy se hledí především na rychlost a uživatelský příjemné používání.

2.5. Knihovny pro implementaci

Pro implementaci jednotlivých úloh diplomové práce byly zvoleny volně šiřitelné knihovny, které jsou naprogramovány v jazyce C++.

2.5.1. ITK

ITK je knihovna pro zpracování obrazů, která je zejména určena pro zpracování lékařských obrazů[24].

2.5.1.1. Obecně

V roce 1999 US National Library of Medicine při National Institutes of Health se rozhodlo vytvořit open-source projekt pro segmentaci a registraci obrazů. Vývoje se účastní šest společností. Tři jsou komerční společnosti (Kitware, GE Corporate R&D, Insightful) a tři organizace z akademického prostředí (UNC Chapel Hill, University of Utah, University of Pennsylvania).

Vznikla tak sada nástrojů, volně dostupných pro všechny vývojáře, umožňující rychlý vývoj aplikací zpracovávajících obrazová data. Zejména pro jejich filtraci, segmentaci a registraci se zaměřením na medicínské aplikace.

ITK je implementováno v C++. To napomohlo nezávislosti ITK na platformě. A ve spojení s CMake zůstává multiplatformním. Knihovny ITK nemusí být využívány jen programátory C++, může být vytvořeno rozhraní pro některé interpretační jazyky jako je Tcl, Java nebo Python.

Při programování v C++ je díky stylu programování s využitím šablon (templates) dosaženo to, že chyby se většinou odhalí při kompilaci a ne při běhu programu. Zároveň takový kód je přenositelný nejen mezi různými platformami (Windows, Linux, MacOS), ale i mezi různými kompilátory (MSVC, Sun, gcc, Intel a SGI kompilátory)

Využitím „smart pointer“ usnadňují správu referencí a automaticky ruší alokované místo v paměti, pokud počítadlo referencí na danou lokalitu klesne na nulu. Tato strategie usnadňuje programátorovi práci tím, že nemusí hlídat velikost alokovaného prostoru. Při práci s velkým objemy dat jako jsou například 3D obrazová data by se programátorovi a později uživateli nevyplatilo neuvolňovat paměť. Při špatné správě paměti by došlo velmi rychle ke zpomalení celého systému.

2.5.1.2. Podpora přenositelnosti kódu

Z důvodů potřeby vytvářet multiplatformní aplikace ITK obsahuje nástroje obsluhující některé systémové funkce. Od správy souborového systému přes regulární výrazy až po správu registrů a správu procesů a vláken.

Lze toho využít při manipulaci s adresářovou strukturou

- Založení adresáře

```
itksys::SystemTools::MakeDirectory( directory_name )
```

- Změna adresáře

```
itksys::SystemTools::ChangeDirectory(directory_name)
```

Další typickou ukázkou může být vybrání souborů x.dcm k načtení do systému v jednom adresáři pomocí regulárního výrazu:

```
itksys::Directory Adresar;
Adresar.Load( jmeno_adresar );
for( int i = 0; i < Adresar.GetNumberOfFiles(); i++ )
{
    std::string soubor = Adresar.GetPath();
    itksys::RegularExpression re("(\\.dcm)$");
    if ( re.find( Adresar.GetFile( i ) ) )
    {
         soubor = soubor + "/" + Adresar.GetFile( i
);
         obrazy.push_back( NactiSoubor(
soubor.c_str() ) );          }
}
```

Zde je načten adresář, projde se celý jeho obsah a pokud je nalezen soubor, jehož název končí na ‚.dcm', tak je načten pomocí funkce NactiSoubor a přidán do std::vector obsahující jednotlivé obrazy v adresáři.

Tyto různé nástroje a pomůcky umožňují efektivní multiplatformní programování. Bez něj bychom museli použít dva různé kódy pro dva různé kompilátory. Díky ITK může programátor napsat přenositelný kód bez nutnosti řešit rozdílnosti kompilátorů a operačních systémů.

2.5.1.3. Architektura

Díky jazyku C++ je ITK výkonná, flexibilní a také dostupná široké komunitě programátorů. Především využívá šablon a předností objektového návrhu. Celá sada ITK má celkem tři vrstvy[28]:

- Šablonová (Objektová) – předpokládá znalost práce se šablonami a objekty
- Run-time – vzniká automatiky (CableSwig) a umožňuje spojení knihovny ITK s interpretovanými jazyky (Tcl, Python, Java)
- Aplikační – není zahrnuta přímo do ITK, ale obsahuje jednoduché aplikace

Dalším podstatným rysem je, že data jsou oddělena od procesů. Samotná data prochází za sebou řadou filtrů spojených rourami a předávajícími původní či již změněná data.Samotné ITK lze rozdělit na několik částí:

- Jádro (správa pamětí, object factory, správa vláken a procesů
- I/O framework (řeší načítání/ukládání dat z/do různých formátů obrazů)
- Filtrační framework
- Segmentační framework
- Registrační framework

2.5.1.4. Podpora DICOM formátu

ITK může pracovat s mnoha formáty obrazových dat, včetně DICOM formátu. Nástrojem zabývajícím se správou DICOM souborů v ITK je knihovna GDCM [27].

Byla vyvinuta CREATIS Team při INSA-Lyon. A umožňuje načtení a uložení souboru na disk, práci s obrazovými daty a práci s daty obsaženými v hlavičce.

Pro načtení a zápis DICOM v3 a ARC/NEMA slouží třída:

```
itk::GDCMImageIO
```

Ve spojení s třídou

```
itk::MetaDataDictionary
```

umožňuje čtení a editaci tagů v hlavičce DICOM. Třída `itk::MetaDataDictionary` je kontejner obsahující dvojici {klíč, hodnota}.

V této práci se využívá například načtení tagu `slice location` s číselným označením podle standartu `0020|1041` pro seřazení jednotlivých 2D snímků, aby se mohli spojit v jeden 3D obraz.

2.5.2. VTK

The Visualization ToolKit (VTK) [25] je rozsáhlá knihovna k tvorbě počítačové grafiky. Celá knihovna je implementována v jazyku C++ a velmi ovlivněna objektově orientovaným návrhem.

Je stejně jako ITK šířena pod licencí open souce. Je tedy volně k dispozici a dnes je využívána širokou skupinou lidí z výzkumu a vývoje.

V mnoha směrech je stejná jako ITK a to včetně možnosti jejího použití na různých operačních systémech (Windows, Unix, Mac OS). Dále je možnost knihovnu pomocí rozhraní napojit na jiné programovací jazyky než C++ (Tcl/Tk, Python, Java).

2.5.3. FLTK

Fast Light Toolkit (FLTK) [26],[30] slouží k tvorbě GUI v jazyce C++. Opět je použitelná na mnoha systémových platformách (UNIX/Linux, Windows, Mac OS). Je navržena tak, aby byla malá a nenáročná na hardwarové prostředky.

3. Data

3.1. Zdroj

Kompletně všechna data byla dodána nemocnicí Na Homolce [1], oddělením Stereotaktické a radiační neurochirurgie.

Jedná se o snímky hlavy získané pomocí magnetické rezonance.

Data jsou dělena podle pacientů a dále podle jednotlivých vyšetření. Tato vyšetření se neliší jen časem, ale i přístrojem, kterým byly vytvořeny.

Protože pro lékaře je obvykle důležitá jen určitá oblast, která se mapuje několikrát různými sekvencemi MRI, tak tato data z Nemocnice Na Homolce obsahují většinou menší řezy. To je dáno jednak časovou náročností skenování a snahou lékařů zkrátit čas pacienta v „tunelu“, který může u některých vyvolávat pocit stísněnosti, tak je to dáno i finančními hledisky a snahou zlevnit již tak nákladnou léčbu zaměřením se na konkrétní místo, na medicínsky významnou oblast.

Získané snímky byly vytvořeny na dvou přístrojích značky Siemens [3], [4].

- Impact Expert, Siemens 1995 (1T)
- Symphony Maestro, Siemens 2002 (1.5T)

3.2. Data

Jde o celkem devět různých pacientů s různým počtem vyšetření (1 – 4). V těchto případech jde o sekvence T1. Řezy jsou provedeny v rovinách:

- Transverzální, roviny jsou horizontální
- Koronální, neboli frontální rovina je vertikální, ale rozděluje tělo na přední (ventrální, čelní) a zadní (dorzální, zádovou) část.

V různých rozlišeních:

- 0,5mm × 0,5mm × 1mm
- 0,5mm × 0,5mm × 3mm
- 1mm × 1mm × 2mm

Na vyjádření hodnoty jednoho pixelu je vždy určeno 12 bitů.

Velikost obrazu je dána rozlišením pro rozlišení 0,5mm × 0,5mm v rovině řezu je velikost obrazu 512×512 pixelů a pro rozlišení v rovině řezu 1mm × 1mm jsou obrazy ve velikosti 256×256 pixelů. Počet řezů je různorodý, zhruba v rozmezí 3-100.

Některá data obsahují stereotaktický rám, který je upevněn na hlavu pacienta, aby vytvořil souřadnicový systém pro plánování operací na Leksellově gamma noži.

3.3. Umístění

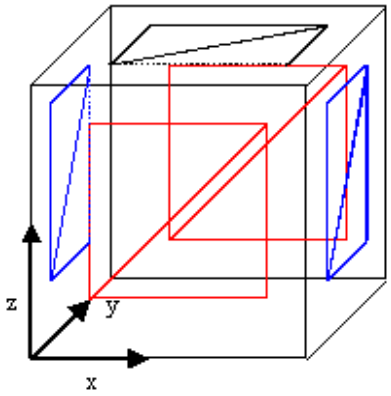
Anonymizovaná data jsou umístěna na přiloženém optickém médiu v adresáři na cestě:

```
. \01_Data\
```

4. Předzpracování obrazu

4.1. Popis problému

Stereotaktický rám se používá jako podpůrný souřadnicový systém ve stereotaktické neurochirurgii, kde slouží k plánování zákroků. Rám je tvořen příčkami umístěnými do pěti rovin (viz Obrázek 1).



Obrázek 1: Schéma stereotaktického rámu v prostoru

Roviny stejných barev jsou rovnoběžné. Roviny s různými barvami jsou na sebe kolmé. V jednotlivých rovinách se poté nachází pět příček rámu, které tvoří čtverec s jednou úhlopříčkou.

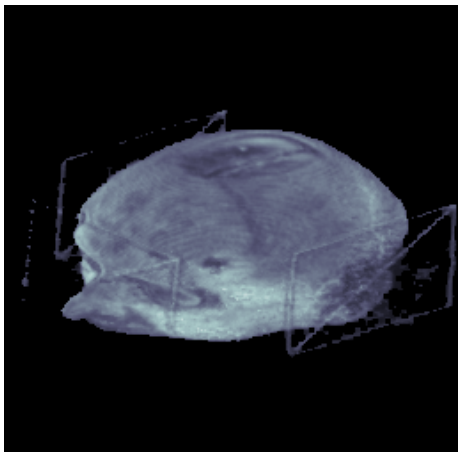
Rám je umístěn na hlavu pacienta a připevněn šrouby k lebce, díky čemuž tvoří fixní systém souřadnic, který se využije při tvorbě ozařovacího plánu. Na obrazech se rám projeví výraznými body s vysokým kontrastem k okolí.

A právě tyto body musíme odstranit z obrazů, jinak komplikují registraci, tedy zjišťování vzájemného posunutí, pootočení nebo změny měřítka dvou obrazů. V praxi totiž občas porovnáváme dva obrazy, které byly získány na různých pracovištích vybavených MRI a jedna data mohou obsahovat značky stereotaktického rámu, zatímco druhá nikoliv. Případně se může stát, že rám je nasazen v odlišné pozici a je tak nutné ho odstranit z obou obrazů pro kvalitnější registraci.

Žádná společnost dodávající softwarové vybavení pro radiodiagnostická oddělení nemocnic neřešila tento problém. A jejich software (Siemens, program „syngo Applications“ [4]) vyžaduje označení značek uživatelem. Zde navržený algoritmus má umožnit plně automatické odstranění značek bez potřeby spolupráce s člověkem.

4.2. Algoritmus řešení

Na začátku jsme ze souboru ve formátu DICOM načetli obraz **O**, který obsahuje 3D MRI obraz.



Obrázek 2: 3D MRI obraz se stereotaktickým rámem, obraz O

V první fázi jsme rozlišili pozadí obrazu pomocí prahování, jehož úroveň jsme stanovili z histogramu obrazu, který jsme spočítali z původního obrazu O (viz Obrázek 2). Šířka binů tohoto histogramu byla určena podle [10] na:

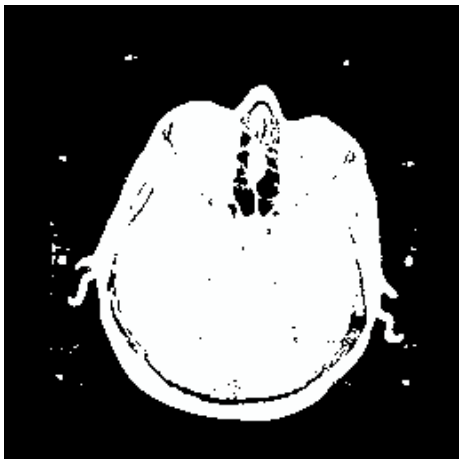
$$W = 3.49 \cdot \sigma \cdot N^{-1/3}, \tag{1}$$

kde W je šířka binu, σ je standardní odchylka a N je počet všech dostupných vzorků. Pro určení prahovací úrovně F_t je použit Otsu filtr [11], [28]. Tato prahovací úroveň pomůže odlišit pozadí od relevantních dat.

Na obraz **O** jsme použili jednoduchý prahovací filtr, který každému pixelu přiřadil hodnotu podle předpisu:

$$M(x,y,z) = \begin{cases} 0 & pro \quad O(x,y,z) \le F_t \\ 1 & pro \quad O(x,y,z) > F_t \end{cases} \tag{2}$$

Obdrželi jsme masku M (Obrázek 3), která pokrývá lékařská data i stereotaktický rám. V další fázi jsme ze získané masky vybrali pouze ty pixely, které jsou na pozicích rámu.

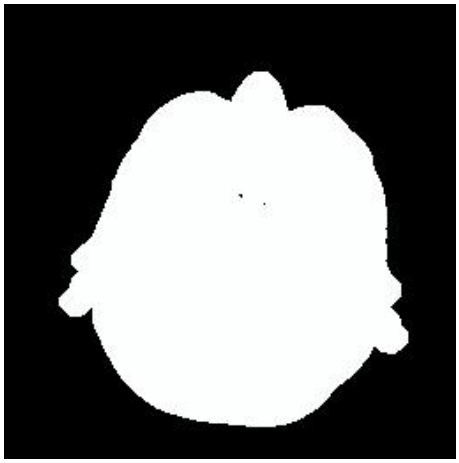


Obrázek 3: Obraz po oprahování, řez obrazem M v rovině XY

To jsme provedli tak, že jsme nechali rozrůstat se prostor do všech tří rozměrů od středu obrazu [28]. Algoritmus pohlcuje všechny pixely, které mají nenulovou hodnotu a sousedí s již algoritmem označeným bodem obrazu:

$$H(x,y,z)=\begin{cases}1, & pro \quad M\left[x+r_x,y+r_y,z\right]=1\\ 0, & pro \quad M\left[x+r_x,y+r_y,z\right]=0\end{cases} \tag{3}$$

Získaný prostor H jsme následně rozšířili dilatací, abychom zamezili nechtěnému mazaní v blízkosti relevantních dat. Velikost masky pro dilataci byla určena z rozměrů daného obrazu a rozlišení, které je obsaženo v hlavičce souborů formátu DICOM. V takto rozšířené oblasti v obraze H' (viz Obrázek 4) se nacházejí data důležitá pro lékaře a registrační algoritmus.



Obrázek 4: Zvětšené segmentované části hlavy, řez obrazem H v rovině XY

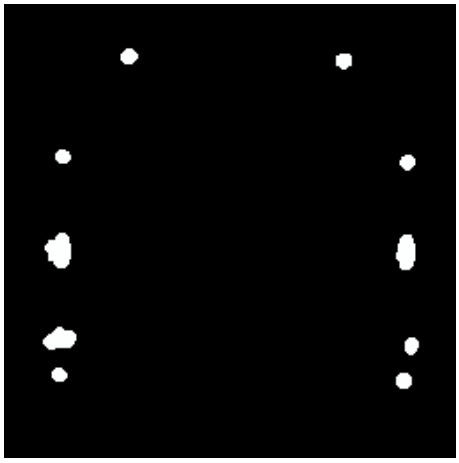
Poté jsme odečetli obraz H' od obrazu masky M:

$$R(x,y,z)=M(x,y,z)-H'(x,y,z), \tag{4}$$

výsledek R obsahoval značky rámu a navíc chyby způsobené prahováním.

Chyby jsme odstranili binárním mediánovým filtrem. Ten díky tomu, že obraz R obsahuje pouze binární data, je optimalizován jen jako čítač počtu ON/OFF sousedních pixelů [28]. Opět byla velikost masky filtru určena z rozměrů a rozlišení obrazu.

Dále jsme opravený obraz R' opět rozšířili dilatací. V tuto chvíli jsme měli v obraze R" (viz Obrázek 5) označeny body stereotaktického rámu a jejich okolí, které může obsahovat stíny rámu vzniklé při snímání MRI obrazů.

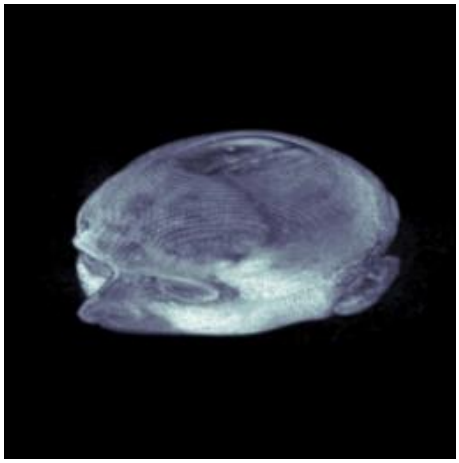


Obrázek 5: Značky stereotaktického rámu po rozšíření dilatací, řez konečným obrazem R" v rovině XY

V konečné fázi algoritmu jsme pak vzali obraz R" a použili ho jako masku k odstranění rámu v původním obraze O. Získali jsme tak výsledný obraz O':

$$O'(x,y,z)=\begin{cases}0, & pro \quad R''(x,y,z)=1\\ O(x,y,z), & pro \quad R''(x,y,z)=0\end{cases}, \tag{5}$$

Tento obraz O' (viz Obrázek 6) již neobsahuje stereotaktický rám a je vhodný k dalšímu zpracování výpočetní technikou. Jsou na něm odstraněny pouze značky stereotaktického rámu, zatímco ostatní důležité informace obrazu zůstaly zachovány.



Obrázek 6: Obraz 3D bez stereotaktického rámu, obraz O'

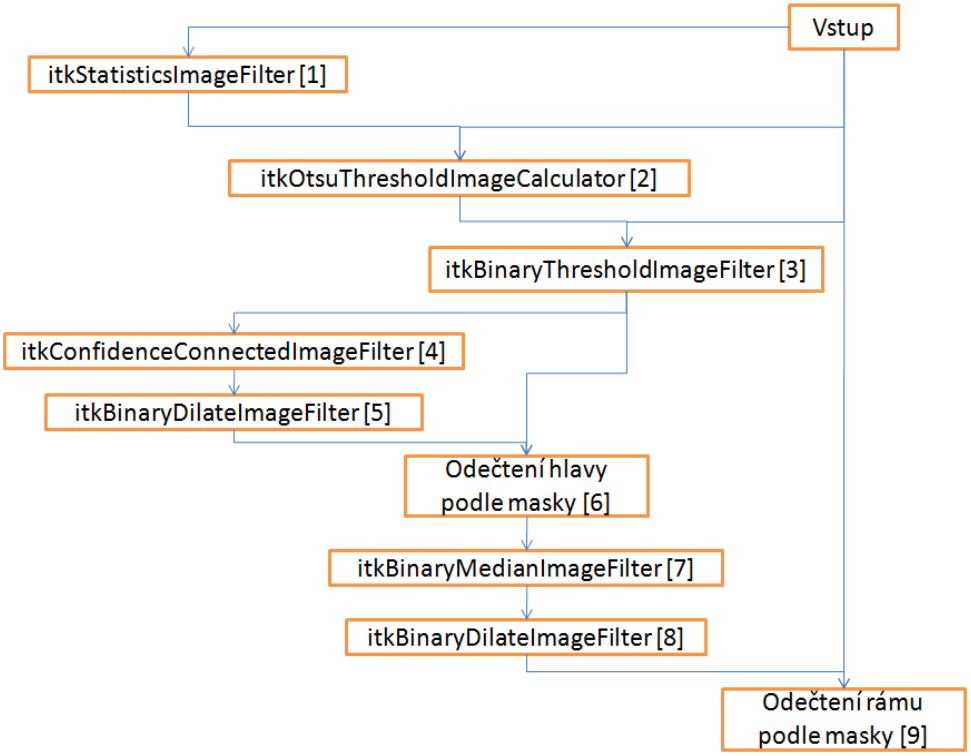
4.3. Implementace

Pro odstranění stereotaktického rámu byla vytvořena vlastní třída `DeleteStereotacticFrame` ve jmenném prostoru itk:

```
template< class TImage >
  class ITK_EXPORT DeleteStereotacticFrame :
    public ImageToImageFilter< TImage, TImage >
```

Tato třída je odvozena od třídy `ImageToImageFilter<TImage, TImage>`, která implementuje obecný filtr, jehož vstupem a výstupem je obraz. Pro třídu `DeleteStereotacticFrame` je vstupem obraz se stereotaktickým rámem a výstupem je obraz zbavený stereotaktické rámu. Bližší popis třídy `ImageToImageFilter` je v [28].

V konstruktoru navržené třídy je kromě inicializace proměnných také vytvořeno propojení jednotlivých filtrů podle vzoru na obrázku (Obrázek 7).



Obrázek 7: Řazení filtrů v třídě DeleteStereotacticFrame

V třídě `StatisticsImageFilter` se počítají statistické údaje např.:

- Maximum a Minimum
- Sigma (směrodatná odchylka)
- Průměr hodnot pixelů
- Součet hodnot pixelů

Filtr `OtsuThresholdImageCalculator` určí hranici prahu pro rozdělení popředí a pozadí obrazu. Určená hranice je vstupem pro filtr, který provede vlastní rozdělení pozadí a popředí obsahující rám a část těla pacienta (hlavy). K tomuto oprahování obrazu slouží ITK třída `BinaryThresholdImageFilter`, které se nastavuje F_t (práh) ve vzorci (2).

V dalším kroku je algoritmem označena část pacienta. K tomu použijeme segmentační metodu rozrůstajícího se prostoru implementovanou ve filtru `ConfidenceConnectedImageFilter`. Předpoklad je, že část pacienta se nachází uprostřed obrazu a proto je inicializace provedena na střed obrazu. Označeny jsou tedy nejprve pixely ve středu a potom všechny pixely, které jsou v sousedství.

Označená část je potom rozšířena pomocí matematické morfologie, dilatace. Na tuto operaci slouží v ITK třída `BinaryDilateImageFilter`.

Následně je odečtením oprahovaného obrazu s obrazem části pacienta získána maska rámu. Ta vstoupí do další funkce a v jednoduchém cyklu je s pomocí masky rámu vymazán z výsledného obrazu celý rám. A zůstává pouze zájmová oblast pro další zpracování. Ta se může uložit do souboru nebo okamžitě vstoupit do dalšího procesu zpracování obrazu.

5. Registrace obrazu

5.1. Popis problému

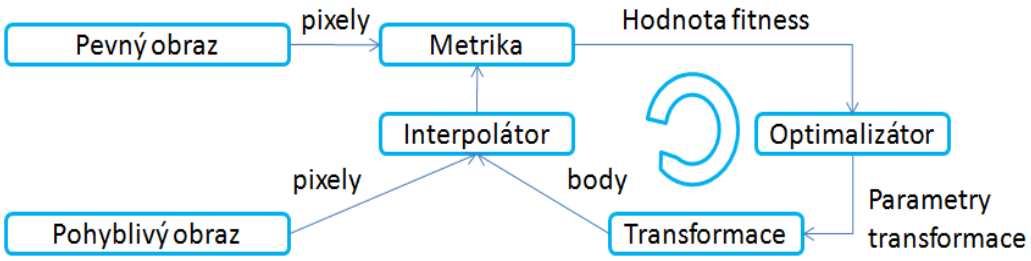
Po předzpracování obrazu je dalším úkolem daný obraz zaregistrovat s obrazem druhým a porovnat tyto dva obrazy odečtením. Z odečtení kvalitně zaregistrovaných obrazů lze například poznat, jak nemoc nebo nádor reaguje na léčbu a postupně se zmenšuje, nebo naopak léčba nepůsobí a nádor se dál rozvíjí.

5.2. Popis prostředků

Popis a vysvětlení základních principů použitých částí pro algoritmus registrace.

5.2.1. Registrace

Jedním z hlavních důvodů vzniku ITK bylo usnadnění programátorům registraci obrazů, proto obsahuje registračních framework, založený na schématu (Obrázek 8)



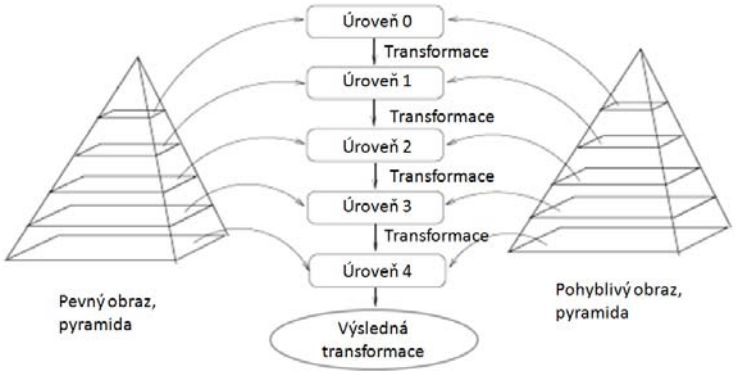
Obrázek 8: Schéma registrace

Vstupem jsou dva obraz, jeden fixní a druhý pohyblivý. Po interpolaci pohyblivého obrazu se vyhodnotí fitness funkce – zhodnotí se, jak moc spolu dané obraz nekorespondují. Optimalizátor rozhodne na základě hodnoty fitness, jak změní parametry transformace. Tedy určí, jak se má pohyblivý obraz posunout, pootočit či deformovat, aby se přiblížil vzoru.

K dispozici je mnoho metod hodnocení rozdílnosti obrazů, optimalizátorů, transformací a interpolací. Jejich různou kombinací můžeme dosáhnout různých přístupů k řešení problému registrace obrazů. Můžeme zvolit jednoduší, výpočetně ne tak náročné, algoritmy, abychom dosáhli porovnání v krátkém čase. Opačnou volbou můžeme dosáhnout robustní registrace s možností registrovat obrazy různých modalit (např.: z CT a MRI nebo MRI a PET).

ITK poskytuje možnost multi-resolutin registrace obrazů [18], [20], [28]. Základní myšlenkou je postupovat od hrubé registrace provedené na menším počtu pixelů a postupně parametry transformace zlepšovat s každým stupněm, ve kterém jsou výchozí parametry brány z předchozího stupně.

To může zlepšit rychlost i přesnost registračních algoritmů [19].



Obrázek 9: Schéma principu metody multi-resolution, převzato z [28]

5.2.2. Transformace

V tomto případě byla použita afinní transformace. Ta je jednou z nejvíce používaných transformací. Její pomocí lze vyjádřit změnu měřítka, posunutí, rotaci a její součástí je i zešikmení. Počet parametrů transformace je roven

$$Pocet = (Dimenze + 1) \cdot Dimenze \quad , \tag{6}$$

V našem případě se dimenze rovná 3 a počet parametrů je po dosazení do vzorce (6) roven dvanácti.

Afinní transformaci pro 3D prostor vyjádříme:

$$\vec{x}' = B\vec{x} + \vec{t} \quad , \tag{7}$$

$$\vec{x}' = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \vec{x} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \tag{8}$$

Nebo

$$\vec{x}' = A\vec{x} \tag{9}$$

$$\vec{x}' = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ a_{31} & a_{32} & a_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \vec{x} \tag{10}$$

Matice A ve vzorcích (8) a (9) v sobě obsahuje změnu měřítka, posunutí a rotaci spolu se zešikmením. Matici A ze vzorce (8) lze tedy vyjádřit součinem jednotlivých transformací:

$$A = M_{posunutí}\ M_{meritko}\ M_{zešikmení}\ M_{rotaceX}\ M_{rotaceY}\ M_{rotaceZ} \quad , \tag{11}$$

Přičemž jednotlivé matice M ze vzorce (10) jsou uvedeny v následující tabulce:

$$M_{posunutí} = \begin{pmatrix} 1 & 0 & 0 & t_1 \\ 0 & 1 & 0 & t_2 \\ 0 & 0 & 1 & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad M_{rotaceX} = \begin{pmatrix} \cos\theta_x & -\sin\theta_x & 0 & 1 \\ \sin\theta_x & \cos\theta_x & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

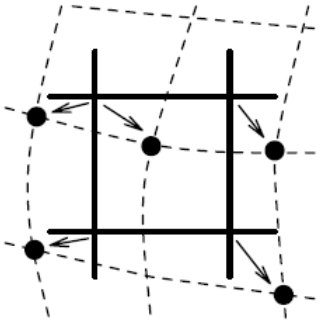
$$M_{meritko} = \begin{pmatrix} m_x & 0 & 0 & 1 \\ 0 & m_y & 0 & 1 \\ 0 & 0 & m_z & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad M_{rotaceY} = \begin{pmatrix} \cos\theta_y & 0 & -\sin\theta_y & 1 \\ 0 & 1 & 0 & 1 \\ \sin\theta_y & 0 & \cos\theta_y & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{zešikmení} = \begin{pmatrix} 1 & z_x & z_x & 1 \\ z_y & 1 & z_y & 1 \\ z_z & z_z & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad M_{rotaceZ} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & \cos\theta_z & -\sin\theta_z & 1 \\ 0 & \sin\theta_z & \cos\theta_z & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

5.2.3. Interpolace

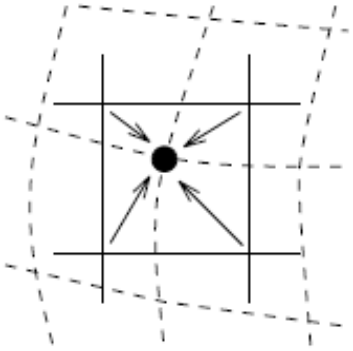
Zvolená metoda interpolace je dalším důležitým bodem registrace. Přímo na ní závisí velikost hodnoty metriky, která je kritériem pro optimalizátor.

Pro lékařské účely se nehodí nejjednodušší možnost interpolace pomocí nejbližšího souseda, která přiřadí novému bodu hodnotu jasu, jež je rovna nejbližšímu sousednímu pixelu v původním obraze. Je velmi rychlá, ale zanáší velkou chybu do výsledného obrazu (Obrázek 10).



Obrázek 10: Interpolace 'Nejbližší soused', převzato z [21]

Pro vlastní registraci je vhodná další metoda, lineární kombinace. Ta přiřazuje hodnotu jasu v novém obraze podle okolních pixelů nacházejícím se v okolí a bere obled na jejich vzdálenost od bodu.



Obrázek 11: Interpolace 'Lineární kombinace', převzato z [21]

V knihovně ITK jsou další, přesnější možnosti interpolace. Například lze použít B-Spline nebo Sinc interpolaci.

Sinc interpolace je pro data v diskrétní mřížce vhodná možnost. Je založena na analýze ve Fourierově spektru. Interpolace se snaží diskrétní vzorkování mřížky převést na spojitou funkci. Její podstatou je oříznutí vysokých frekvencí ve spektru. To může být provedeno nastavením nulových hodnot ve spektru, nebo konvolucí se sinc funkcí:

$$\mathrm{sinc}(x) = \frac{\sin \pi t}{\pi t} \tag{12}$$

Definiční obor této funkce je $D \in (-\infty, \infty)$. Takto zvolený definiční obor nelze naprogramovat, nelze strojově vyjádřit nekonečno. Proto ve skutečnosti je *D* oříznuto a tím vzniká nespojitost ve funkci počítající intenzitu pixelu v daném místě. Tato nespojitost přináší nepřesnost, která se řeší pomocí oknových funkcí. V ITK je standardně nastavena jako oknová funkce Hammingova funkce.

Jádro takové funkce potom vypadá:

$$K(t) = w(t)\mathrm{sinc}(t) \tag{13}$$

$$K(t) = \left(0{,}54 + 0{,}46\cos\frac{\pi x}{m}\right)\frac{\sin \pi t}{\pi t}, \tag{14}$$

Kde *m* představuje velikost okna. Volbou tohoto parametru *m* je určena výpočetní náročnost této interpolace.

Pro registrační algoritmus je lepší použít lineární kombinaci, která je rychlejší než sinc interpolace. Naopak po registraci pro vykreslení obrazu je vhodnější použít právě jemnější sinc interpolaci [12].

5.2.4. Metrika

Metrika slouží k měření kvality transformace jednoho obrazu do druhého a je jednou z nejvýznamnějších částí registrace. Její výběr záleží na problému, který chceme řešit. V našem případě můžeme zvolit jednodušší výpočet, který neumožňuje porovnání obrazů různých modalit, ale je rychlý a vhodný pro porovnání stejných modalit.

5.2.4.1. Střední kvadratická chyba

Počítá střední kvadratickou chybu z rozdílů intenzit pixelů mezi obrazem A a B:

$$MS(A,B) = \frac{1}{N}\sum_{i=1}^{N}(A_i - B_i)^2, \tag{15}$$

Kde N je počet společných pixelů pro oba obrazy.

Pokud se výsledek rovná 0, tak všechny pixely v překrývané části mezi obrazy jsou stejné intensity a obrazy se dokonale překrývají. Naopak pokud hodnoty jsou vysoké, znamená to, že obrazy se nepřekrývají.

Tato metrika je závislá na rozsahu intenzity pixelů, a proto není možné její pomocí porovnávat obrazy různých modalit, kdy se liší intervaly intenzit pixelů.

5.2.4.2. Vzájemná informace

Tato metrika počítá, jak moc je jedna náhodná veličina (intenzita v obraze A) podobná jiné náhodné veličině (intenzitě v obraze B). Výhodou je, že závislost těchto náhodných veličin nemusí být definována a tak se tato metrika dá použít k porovnání podobnosti obrazů z různých modalit [28].

Vzájemná informace je definována pomocí entropie:

$$H(A) = -\int p_A(a)\log(p_A(a))da \quad (16)$$

,

$$H(B) = -\int p_B(b)\log(p_B(b))db \quad (17)$$

.

Sdružená entropie je potom:

$$H(A,B) = \int p_{AB}(a,b)\log(p_{AB}(a,b))dadb \quad (18)$$

.

Pokud jsou jednotlivé náhodné veličiny na sobě nezávislé, pak platí:

$$H(A,B) = H(A) + H(B) \quad (19)$$

.

Naopak při jakékoliv míře závislosti platí:

$$H(A,B) < H(A) + H(B) \quad (20)$$

.

Rozdíl je potom nazýván vzájemnou informací I(A,B), která se vyjádří jako:

$$I(A,B) = H(A) + H(B) - I(A,B) \quad (21)$$

.

V tomto případě hledáme maximum této funkce, které značí maximální překrytí obrazů.

5.2.5. Optimalizátor

Vyhledávání vhodné transformace v prostoru je úkolem optimalizátoru. Ten určuje změnu parametrů transformace podle ohodnocení získané z metriky.

5.2.5.1. Gradientní optimalizátor s pravidelným krokem

Algoritmus s jednoduchou myšlenkou hledat optimální řešení po gradientním spádu:

$$G(x,y,z) = \nabla f(x,y,z) \quad (22)$$

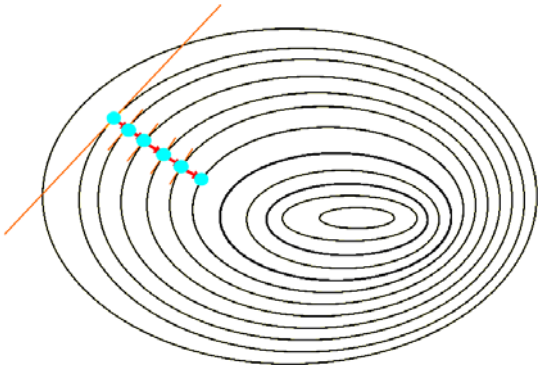
.

Kde $f(x,y,z)$ je funkce, kterou se snažíme zoptimalizovat. Potom velikost následného kroku lze určit vztahem:

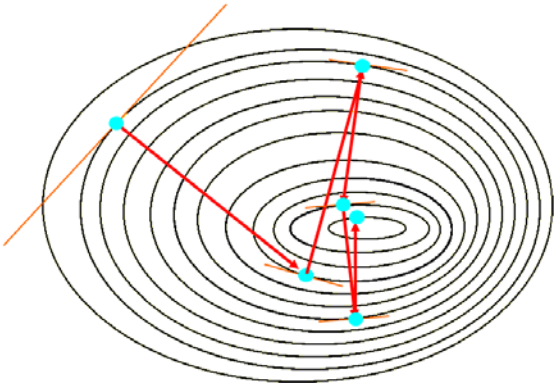
$$S = L \cdot G(x,y,z) \quad (23)$$

,

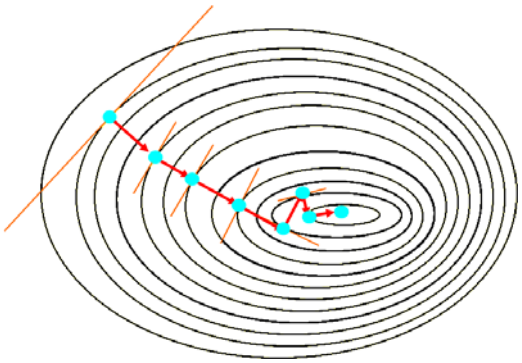
Kde S je velikost následujícího kroku a L je délka kroku. Volba tohoto koeficientu ovlivňuje kvalitu a rychlost optimalizace pomocí gradientní metody (viz Obrázek 14).



Obrázek 12: Příliš malý koeficient učení L, převzato z [29]



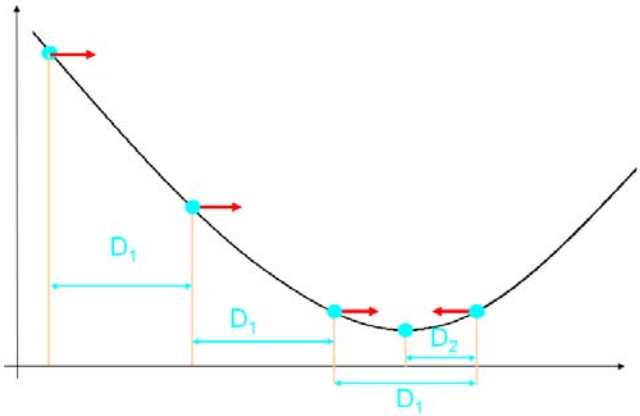
Obrázek 13: Příliš veliký koeficient učení L, převzato z [29]



Obrázek 14: Vhodně zvolený koeficient učení L, převzato z [29]

Zvolením malého koeficientu učení způsobíme, že optimalizační algoritmus postupuje pomalu jako v případě na obrázku 12. Opačný případ je ukázán na obrázku 13.

Pokud chceme zmenšit vliv volby koeficientu učení na optimalizační algoritmus je vhodné zvolit sofistikovanější metodu, gradientní optimalizaci s pravidelnou velikostí kroku. Ta je obměnou předchozí optimalizace. Změna jednotlivých parametrů je stále stejná pokud se algoritmus stále drží stejného trendu postupu. Při výrazné změně trendu se krok zmenší, protože optimalizátor předpokládá, že v tu chvíli se již přibližuje k hledanému extrému funkce. Na obrázku 15 se optimalizátor přibližuje k extrému pravidelným krokem D_1 a při změně směru se krok optimalizace mění na D_2.



Obrázek 15: Gradientní optimalizace s pravidelným krokem, převzato z [29]

5.2.5.2. L-BFGS-B optimalizátor

Tento optimalizátor se snaží o hledání minima nelineární funkce s *n* proměnnými, to lze vyjádřit jako:

$$\min f(\vec{x}), \tag{24}$$

kdy platí

$$l_i \le x_i \le u_i \tag{25}$$

Kde *l* a *u* jsou spodní a horní hranice jednotlivých proměnných vstupujících do nelineární funkce $f : \Re \to \Re$ s n proměnnými [22].

Algoritmus L-BFGS-B (Limited memory Broyden Fletcher Goldfarb Shannon minimalization with simple bounds) nepotřebuje znát druhou derivaci nebo mít znalost vnitřní struktury dané funkce *f*. Díky těmto vlastnostem lze L-BFGS-B použít, když není možné spočítat Hessian.

L-BFGS-B je rozšířením L-BFGS. Na rozdíl od svého předchůdce je L-BFGS-B je schopen se vypořádat s mezemi pro proměnné. Je tak jediným quasi-Newtonovským algoritmem s limitovanou pamětí, jež má tuto schopnost. Jiné algoritmy tuto možnost bohužel nemají.

Algoritmus je podrobně popsán v [22], [23]. Obecně v každé iteraci L-BFGS-B zlepšuje aproximaci Hessianu, který definuje kvadratický model funkce *f*.

Dvěma kroky se zjišťuje směr prohledávání prostoru:

- Identifikuje se sada aktivních proměnných. (To jsou proměnné, které budou drženy ve stanovených mezích *l* a *u*.)
- Kvadratický model aproximace je minimalizován s ohledem na volné proměnné.

Směr prohledávání je potom dán jako vektor změny mezi současnou iterací a touto poslední aproximací.

Tomuto algoritmu jdou nastavit meze pro paměťové nároky. To se děje určením počtu korekcí, které si má L-BFGS-B pamatovat. Algoritmus potřebuje zhruba $(12 + 2m)n$ míst k uložení. Kde *n* je počet proměnných funkce *f* a *m* lze volit v rozmezí $3 \le m \le 20$. A i tyto hodnoty stačí k řešení velkých problémů.

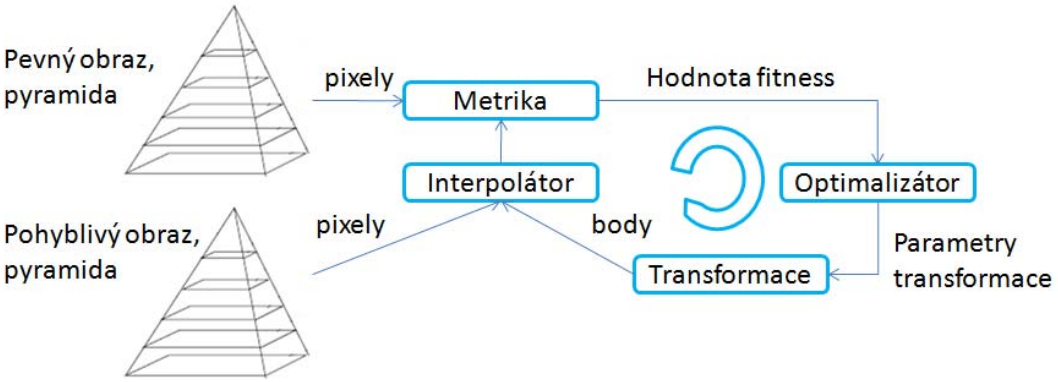
Výpočetní náročnost je proměnná a záleží na počtu proměnných mající určené meze. Pohybuje se v rozmezí od $4mn+n$ (nejsou určeny žádné meze) po $m^2 n$ (všechny proměnné mají meze).

Klady	Zápory
• Jednoduché použití (nemusí být znám Hessián) • Paměťové nároky lze zvolit • Výpočetní náročnost jedné iterace je nízká	• Konverguje pomalu (pro velké problémy je třeba mnoha výpočtů) • Nemůže využít znalost problému k rychlejšímu dokonvergování k cíli

Tabulka 1: Tabulka shrnující klady a zápory optimalizátoru L-BFGS-B

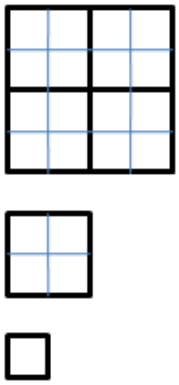
5.3. Algoritmus Řešení

Vstupem jsou dva obrazy, které chceme registrovat. Spojením obrazů z kapitoly 5.2.1 (Obrázek 8 a Obrázek 9) vznikne použité schéma při naší registraci (Obrázek 16).

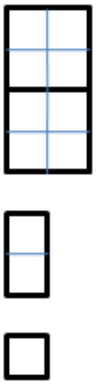


Obrázek 16: Použité schéma registrace

Tyto obrazy vstoupí do filtru, který určí jejich rozdělení do jednotlivých úrovní pyramidy. Rozměry obrazu mezi jednotlivými úrovněmi pyramidy jsou poloviční, pokud je to možné. Daný rozměr obrazu nelze zmenšit, pokud rozměr dosáhl svého minima je ponecháno toto minimum a rozměry obrazu v ostatních osách se dále zmenšují.



Obrázek 17:Pravidelná změna velikosti



Obrázek 18: Nepravidelná změna velikost po dosažení spodní meze

V prvním případě se obraz může zmenšovat pravidelně ve všech osách. Ve druhém případě je v druhém kroku v ose x dosaženo maxima zmenšení a proto se obraz již v této ose x nepůlí, ale zůstává stejný.

Obraz se zmenší podle hodnot uvedených v tabulce 2.

X	Y
1	1
$\frac{1}{2}$	$\frac{1}{2}$
$\frac{1}{4}$	$\frac{1}{4}$

X	Y
1	1
$\frac{1}{2}$	$\frac{1}{2}$
$\frac{1}{2}$	$\frac{1}{4}$

Tabulka 2: Poměry zmenšení obrazu pro různé úrovně.

Algoritmus nejdříve zaregistruje nejmenší obrazy a získanou transformaci použije v inicializaci následné úrovně s dvojnásobně velkými obrazy. Tento počet se provádí stejně pro všechny úrovně, jejich počet lze libovolně zvolit.

Za metriku je zvolena Střední kvadratická chyba (kapitola 5.2.4.1), která je dostatečně rychlá. Metrika postavená na Vzájemná informace (kapitola 5.2.4.2) byla použita pouze pro některé pokusy registrovat obrazy různých modalit a s různými intenzitami pixelů.

Jako transformace je zvolena afinní transformace. Ta umožňuje spočítat všechny požadované parametry:

- Translace
- Rotace
- Změna měřítka

Podle vzorce (6) má tedy ve 3D prostoru celkem 12 parametrů. Ve 2D by měla pouze poloviční počet, tedy 6.

Jako optimalizátor byl zvolen Gradientní optimalizátor s pravidelným krokem (kapitola 5.2.5.1). Optimalizátor L-BGFS-B(kapitola 5.2.5.2) je vhodnější pro hledání složitějších deformačních transformací.

Pro správnou funkčnost je třeba nastavit několik parametrů:

- Význam jednotlivých parametrů: Určuje, které parametry se mají pohybovat ve větším rozsahu a které parametry nemají takový vliv na transformaci.
- Parametry kroku, kterým je prostor prohledáván

5.4. Implementace

Byla navržena třída:

```
template< class TImage >
  class ITK_EXPORT MultiResolutionGRS : public
    Object
```

Ta využívá registrační framework implementovaný v ITK. Lze zvolit několik parametrů:

- Využívanou metriku
- Využívaný optimalizátor
- Počet úrovní pro multi-resolution

Využívá třídu pro multi-resolution registraci:

`itk::MultiResolutionImageRegistrationMethod`

Třída obsahující afinní transformaci:

`itk::AffineTransform`

Třída implementující interpolační funkci:

`itk::LinearInterpolateImageFunction`

Metriky pro porovnání obrazů se nacházejí v třídách:

`itk::MeanSquaresImageToImageMetric`

A optimalizační algoritmy jsou obsaženy

`itk::RegularStepGradientDescentOptimizer`

V konstruktoru třídy jsou propojeny s registrací třídy, jejichž výběr není ovlivněn programátorem. A inicializovány proměnné podle doporučení [32], [33].

Po nastavení vstupů, dvou obrazů, z nichž jeden je pevný a druhý pohyblivý, se v první řadě určí podle počtu úrovní, ve kterých budeme registrovat, jak budou dané obrazy zmenšeny v jednotlivých úrovních registrace. To provedeme mocí funkce

```
ShrinkFactor LevelImageSize(TImage::Pointer Input).
```

Tuto operaci provedeme na oba obrazy, pevný i pohyblivý. Obraz se nesmí zmenšit příliš (velikost v jednotlivých rozměrech nesmí klesnout pod 4), protože potom zmenšování vyhodí výjimku z třídy `itk::DiscreteGaussianImageFilter`.

Transformaci jsou nastaveny počáteční parametry. Pro inicializace je použita transformace neobsahující žádnou rotaci ani změnu měřítka. Počáteční translace je potom nastavena jako rozdíl souřadnic počátků pevného a pohyblivého obrazu.

Počáteční inicializaci lze tedy vyjádřit jako:

$$\vec{x}' = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \vec{x} + \begin{pmatrix} t_{1f} - t_{1m} \\ t_{2f} - t_{2m} \\ t_{3f} - t_{3m} \end{pmatrix} \tag{26}$$

V neposlední řadě jsou nastaveny parametry optimalizátoru. V první řadě je třeba nastavit význam proměnných v transformaci. Lze je rozdělit na tři skupiny:

- Parametry matice (obsahující rotaci a změnu měřítka- vzájemně se překrývají v matici)
- Translace v osách X a Y
- Translace v ose Z

Parametry matice jsou hodnoty, které se mění v malém rozsahu, naopak Translace, posuny obrazu v jednotlivých osách, se mění v poměru k hodnotám matice o mnoho více. Proto je nastaven poměr podle tabulky 3.

Parametry matice (parameters[0] - parameters[8])	1
Translace v osách X a Y (parameters[9] a parameters[10])	$\frac{1}{2\sqrt{x^2+y^2+z^2}}$
Translace v ose Z (parameters[11])	$\frac{1}{\sqrt{x^2+y^2+z^2}}$

Tabulka 3: Význam jednotlivých parametrů pro registraci

Hodnota je odvozena od fyzické velikosti obrazu, protože parametry translace jsou určeny podle úhlopříčky v obraze. V osách X a Y se předpokládá větší posun obrazu než v ose Y, a proto jsou významy parametrů posunutí v těchto osách (X a Y) dvojnásobné oproti posunutí v ose Z.

Dále je třeba nastavit vlastnosti kroku optimalizátoru. Omezit jeho velikost, aby nepřekročil oblast obrazu a dále minimální velikost kroku. Kdyby velikost kroku příliš klesla, tak by registrace mohla trvat mnohem déle a mohla by najít špatné řešení, protože při malém kroku se již nemusí gradient příliš měnit a tak ukončit registraci z důvodů malé změny metriky.

Velikost kroku se také mění mezi jednotlivými úrovněmi registrace. Nejprve je krok velký. To umožňuje rychlé najití první transformace. Postupně, při přecházení úrovněmi je krok nastavován na stále menší, protože na vyšších úrovních se parametry transformace dolaďují podle přibývajících detailů v obraze. Krok se tedy postupně zjemňuje.

Na konci jsou registračnímu algoritmu přiřazeny jednotlivé součásti podle schématu (viz Obrázek 16) a je spuštěna samotná registrace.

Ta nejprve vytvoří obraz dané úrovně (převzorkuje původní obraz do zmenšeného obrazu) a poté je spuštěna registrace obrazu dané úrovně. Když je ukončena použijí se získané parametry pro inicializaci registrace v další úrovni. A tak se postupně algoritmus propracuje k obrazu v plném rozlišení.

6. Experimenty

Všechny statistické výsledky jsou na přiloženém médiu na cestě:

```
.\04_Exper\
```

Obsahuje pouze výsledky ve formě textových logů. S jejich pomocí lze všechna data zrekonstruovat. Jediné, co nelze, je znovuvytvořit stejný náhodný šum přidávaný do obrazů.

Všechna obrazová data, která byla vytvořena během experimentů, byla v řádu desítek GB. Z tohoto důvodu není možné je všechny uchovávat.

V adresáři:

```
.\04_Exper\03_Pic\
```

Jsou uchovány některé z pokusů.

Byly provedeny experimenty pro otestování odebrání stereotaktického rámu z obrazu (viz kapitola 6.1). A dále experimenty pro ověření úspěšnosti registrace obrazů (viz kapitola 6.2).

6.1. Odstranění stereotaktického rámu

Pro experimenty byly použity původní obrazy obsahující stereotaktický rám. Byly vybrány z dodaných dat a byl nad nimi spuštěn algoritmus popsaný v kapitole 4.2. Výsledkem bylo dvacet experimentů, jejichž výsledky shrnuje tabulka v příloze A.

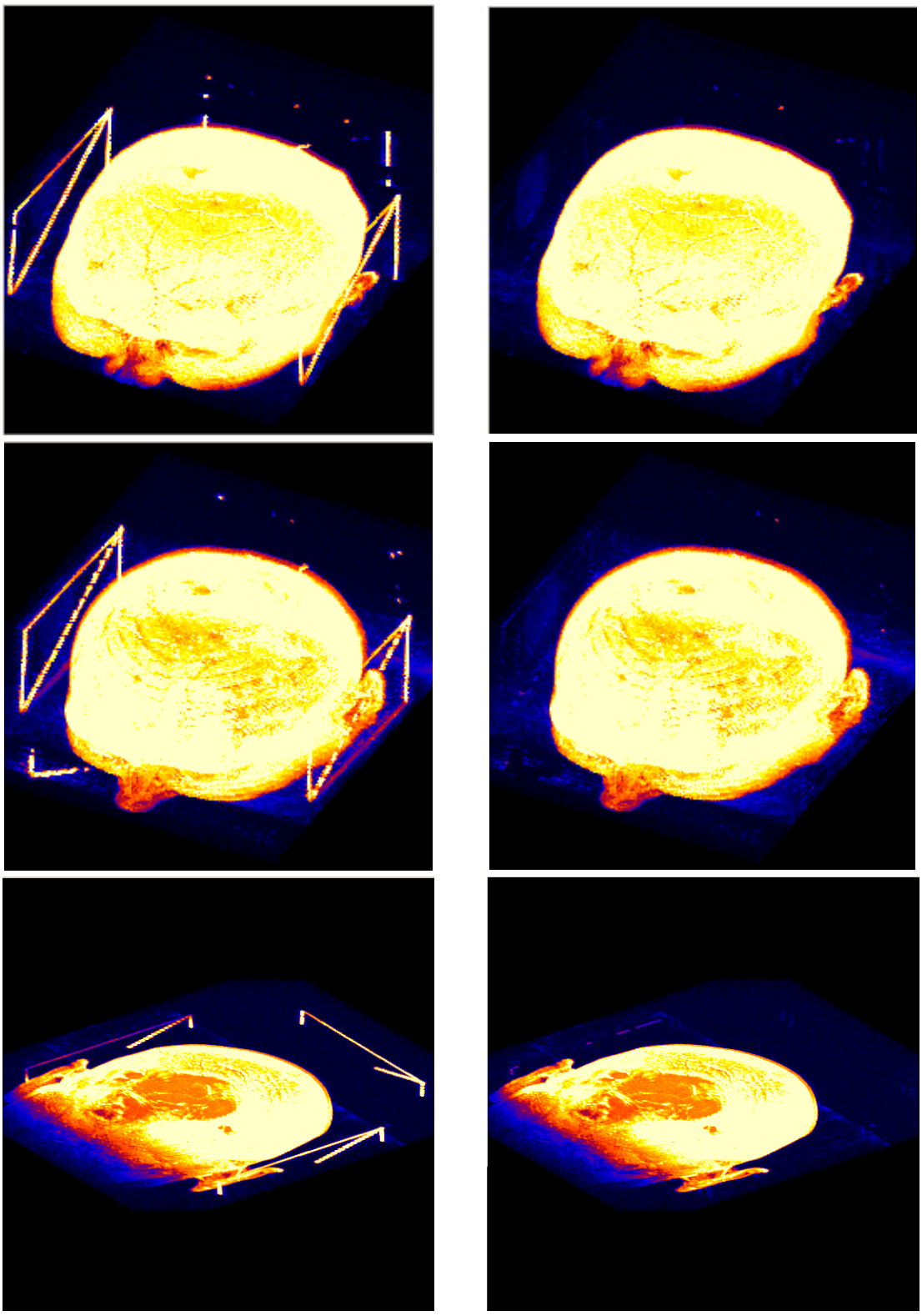
Ve třech případech došlo k tomu, že v obraze zůstala významná část rámu (celá příčka). Ve dvou případech je to způsobeno tím, že příčka rámu je na obraze nevýrazná a stanovený práh neumožnil algoritmu při prahování příčku stereotaktického rámu označit.

V posledním, třetím případě se příčka nacházela v blízkosti hlavy a spadala do oblasti, která byla chráněna před smazáním. To se stalo při kroku, který onačil oblast hlavy, kterou z bezpečnostních důvodů ještě zvětšil.

Rychlost algoritmu je lineárně závislá na velikosti obrazu. S velikostí obrazu roste i časová náročnost na odstranění značek stereotaktického rámu. Tuto skutečnost lze vidět v tabulce v příloze B, která ukazuje čas trvání algoritmu a velikost souboru.

Na šesti obrázcích (Obrázek 19) jsou ukázány případy úspěšného odebrání stereotaktické rámu z 3D MRI obrazů. Pro zvýraznění rámu v obrazech před spuštěním algoritmu je použito obarvení „Hot metal“, která zvýrazní vysoké intenzity pixelů.

Tabulka, která spojuje výše zmíněné experimenty s daty je v příloze C.



a) Obraz se stereotaktickým rámem

b) Obrazy po smazání stereotaktického rámu

Obrázek 19: Porovná obrazů před a po provedení algoritmu na odstranění stereotaktického rámu (použité barevné schéma „Hot metal“

6.2. Registrace

Pro experimenty byly použity uměle vytvořené 3D obrazy získané z jednoho vyšetření, které mělo dostatečnou velikost ve všech dimenzích.

Zdrojem byla data na přiloženém optickém médiu, na cestě:

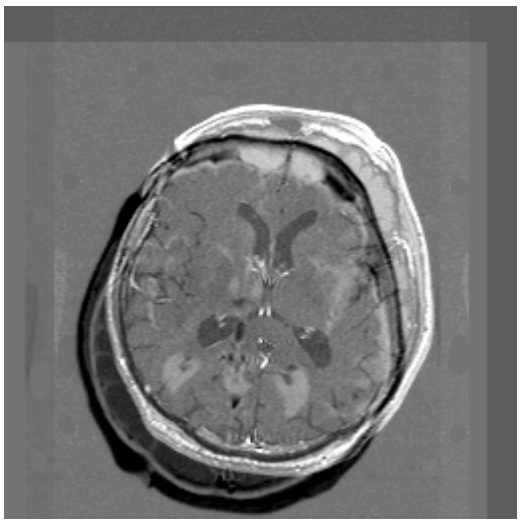
```
D:\diplomka\FinalCD\01_Data\07_Pac\Study__20060802__0
74508.312000\Series__20060802__075203.015000\1.3.12.2
.1107.5.2.6.22428.30000006080120230859300000935\
```

Tato zdrojová data mají rozměry 256 × 256 × 80. S velikostí voxelu 0,98mm × 0,98mm × 2mm. Šedotónová stupnice je v každém voxelu reprezentována 12b.

Při každém experimentu byla ze zdrojového obrazu náhodně vybrána část obrazu v rozměrech 256 × 256 × Z, kde Z bylo náhodně zvolené číslo v intervalu 20 – 60.

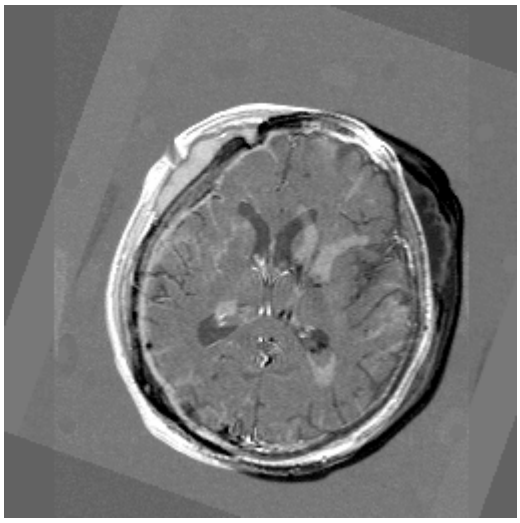
Tento obraz byl potom náhodně transformován. Tato transformace byla tří druhů:

- Pouze translace



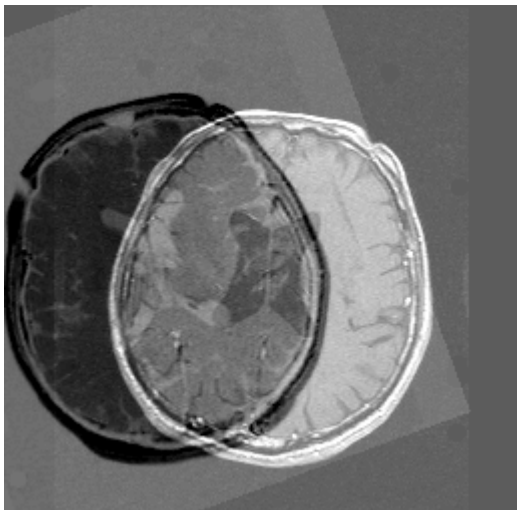
Obrázek 20: Ukázka posunu obrazu (řez rovinou XY)

- Pouze rotace



Obrázek 21: Ukázka otočení obrazu (řez rovinou XY)

- Spojení translace, rotace a změny měřítka



Obrázek 22: Ukázka otočení a posunutí s malou změnou měřítka (řez rovinou XY)

Každý experiment byl proveden pro pět hladin šumu v obraze. Jednalo se o aditivní náhodný šum s normálním rozložením. Směrodatná odchylka **σ** byla zvolena logaritmicky

$$\sigma = \{0,10,20,50,100\}$$

Všechny experimenty byly provedeny registrace s použitím 3 úrovní multi-resolution registrace. Obraz se tedy nejdříve registruje ve čtvrtinovém rozměru, poté v polovičním a nakonec v plné velikosti.

6.2.1. Translace

Posunutí je udáváno ve voxelech. Pro účely použití muselo být převedeno na fyzickou velikost v mm. ITK knihovna neoperuje s pixely a voxely, ale přímo ve fyzických souřadnicích.

Pro účely experimentů byly vybrány intervaly, ve kterých se prováděli jednotlivé série testů:

- 0 - 1 voxel v každém směru
- 1 – 2 voxely v každém směru
- 2 – 5 voxelů v každém směru
- 5 – 10 voxelů v každém směru
- 10 – 50 voxelů v každém směru

Pro každý interval posunutí bylo provedeno 100 pokusů pro všech 5 úrovní šumu.

6.2.1.1. Časová náročnost registrace translace

Tabulka 4 ukazuje průměrný čas potřebný k dokončení registrace obrazu v závislosti na vzdálenosti mezi obrazy a v závislosti na šumu v obraze.

Potřebný čas a počet iterací roste společně se vzdáleností, která dělí registrované obrazy. Dále v průměru roste i čas k registraci více zašuměných obrazů (Tabulka 5).

Při zvětšující se vzdálenosti roste složitost algoritmu a registrace je nucena počítat více iterací pro přiblížení obrazů.

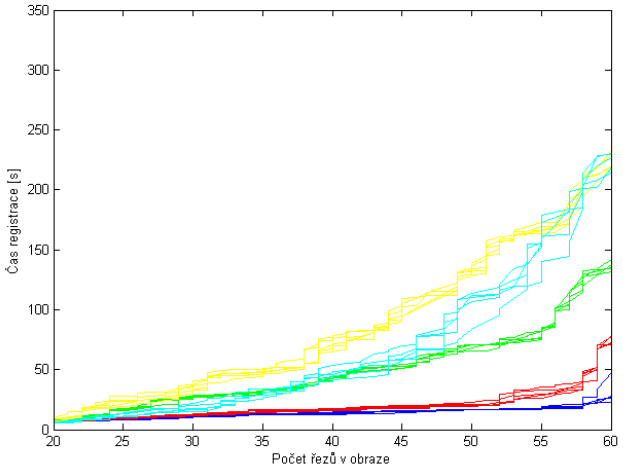
Pásmo posunu[px]	Posun XY[mm]	Posun Z[mm]	σ=0 [s]	σ=10 [s]	σ=20 [s]	σ=50 [s]	σ=100 [s]
0-1	0-0,98	0-2	13,07	13,50	13,74	13,41	14,46
1-2	0,98–1,96	2-4	17,60	17,97	18,27	18,99	19,83
2-5	1,96-4,9	4-10	45,35	45,33	45,60	44,85	44,06
5-10	4,9-9,8	10-20	65,13	57,70	67,30	63,25	65,06
10-50	9,8-49	20-100	75,69	77,57	76,77	74,19	79,04

Tabulka 4: Průměrný čas jedné registrace při translaci pro různé velikosti posunu a úrovně šumu σ

Pásmo posunu[px]	Posun XY[mm]	Posun Z[mm]	σ=0	σ=10	σ=20	σ=50	σ=100
0-1	0-0,98	0-2	19	20	20	20	21
1-2	0,98–1,96	2-4	26	26	26	26	27
2-5	1,96-4,9	4-10	69	69	67	68	70
5-10	4,9-9,8	10-20	136	136	138	139	143
10-50	9,8-49	20-100	272	254	278	278	295

Tabulka 5: Průměrný počet iterací při translaci pro různé velikosti posunu a úrovně šumu σ

Časová náročnost je též závislá na velikosti obrazů, které registrujeme. Na grafu (Obrázek 23) je vidět, jak s přibývajícím počtem řezů (s rostoucí velikostí obrazu) roste i časová náročnost registrace. Na obrázku jsou barevně odlišeny jednotlivé intervaly, ve kterých se prováděla registrace.



Obrázek 23: Graf časová závislost délky registrace na velikosti obrazů

6.2.1.2. Úspěšnost a přesnost algoritmu registrace pro translaci

Úspěšnost registrace transformace shrnuje tabulka 6. Ta zobrazuje procentuální úspěšnost registrace

Je založena na rozdílu známých a zjištěných parametrů transformace. Pokud je rozdíl u některého parametru translace větší než stanovená mez (více jak 3mm), je pokus považován za neúspěšný.

Úspěšnost klesá s velikostí vzdálenosti obrazů. To je způsobeno i tím, že registrované obrazy mizí z prohledávané oblasti.

Pásmo posunu[px]	Posun XY[mm]	Posun Z[mm]	σ=0 [%]	σ=10 [%]	σ=20 [%]	σ=50 [%]	σ=100 [%]
0-1	0-0,98	0-2	99	99	99	99	99
1-2	0,98–1,96	2-4	100	100	100	100	100
2-5	1,96-4,9	4-10	81	81	80	80	80
5-10	4,9-9,8	10-20	31	31	31	31	31
10-50	9,8-49	20-100	5	5	4	4	5

Tabulka 6: Úspěšnost registrace translace translaci pro různé velikosti posunu a úrovně šumu σ

Metrika počítá rozdíl obrazů podle vzorce (15). Z tabulky 7 je vidět, jaký vliv na výsledek registrace má vzdálenost obrazů a míra šumu v obraze.

Šum v obraze zapříčiní vysoký nárůst nepřesnosti. Na každý z obrazů byl použit náhodný aditivní šum, který právě při použití intenzit pixelů k porovnání obrazů způsobí tento nárůst.

Pásmo posunu[px]	Posun XY[mm]	Posun Z[mm]	σ=0	σ=10	σ=20	σ=50	σ=100
0-1	0-0,98	0-2	196,67	249,88	366,85	1019,32	3206,03
1-2	0,98–1,96	2-4	403,56	425,46	530,64	1143,52	3198,80
2-5	1,96-4,9	4-10	1261,24	1304,24	1520,80	2117,41	4192,41
5-10	4,9-9,8	10-20	4225,89	4566,49	4707,75	4697,86	7008,23
10-50	9,8-49	20-100	11320,16	11010,84	14668,19	11126,29	11680,03

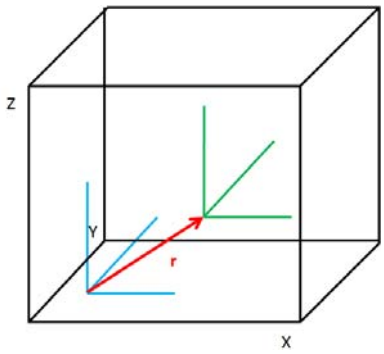
Tabulka 7:Průměrná konečná hodnota metriky při translaci pro různé velikosti posunu a úrovně šumu σ

Lepším ukazatelem úspěšnosti registrace může být velikost vektoru, který vznikl rozdílem vektoru známé translace a translace zjištěné registrací:

$$r = \left\| \vec{t}_z - \vec{t}_r \right\| \tag{27}$$

$$r = \left\| \begin{pmatrix} t_{z1} \\ t_{z2} \\ t_{z3} \end{pmatrix} - \begin{pmatrix} t_{r1} \\ t_{r2} \\ t_{r3} \end{pmatrix} \right\| \tag{28}$$

$$r = \sqrt{(t_{z1} - t_{r1})^2 + (t_{z2} - t_{r2})^2 + (t_{z3} - t_{r3})^2} \tag{29}$$



Obrázek 24: Velikost vektoru r

Průměrné hodnoty těchto rozdílů shrnuje tabulka 8Tabulka 8, která ukazuje, jak pro vyšší hodnoty posunu již registrace není přesná.

To je způsobeno především tím, že obraz se hlavy pacienta se dostává již z velké části mimo alokované pole obrazu. A tak registrace se snaží registrovat obrazy, které se nemohou zcela překrývat.

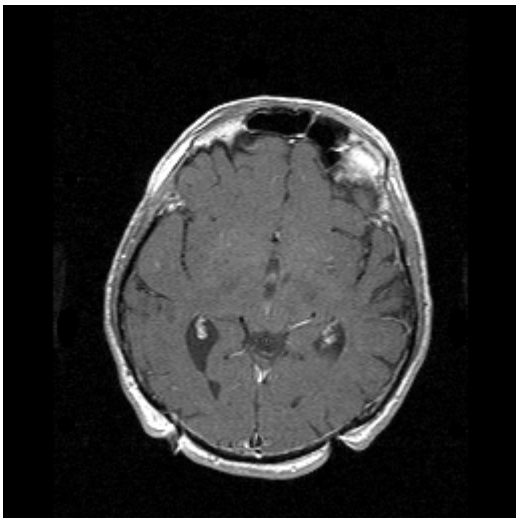
Pásmo posunu[px]	Posun XY[mm]	Posun Z[mm]	σ=0 [mm]	σ=10 [mm]	σ=20 [mm]	σ=50 [mm]	σ=100 [mm]
0-1	0-0,98	0-2	0,72	0,74	0,75	0,74	0,73
1-2	0,98–1,96	2-4	1,78	1,77	1,78	1,77	1,76
2-5	1,96-4,9	4-10	3,37	3,37	3,36	3,58	3,98
5-10	4,9-9,8	10-20	10,06	10,10	9,99	10,79	10,54
10-50	9,8-49	20-100	66,52	68,23	69,38	71,39	71,13

Tabulka 8: Průměrné rozdíly vektorů translace pro různé velikosti posunu a úrovně šumu σ

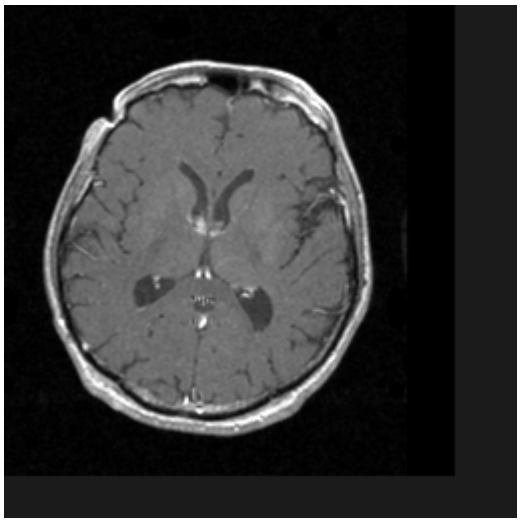
6.2.1.3. Ukázka registrace posunu

Na ukázku je provedena registrace obrazu s maticí transformace:

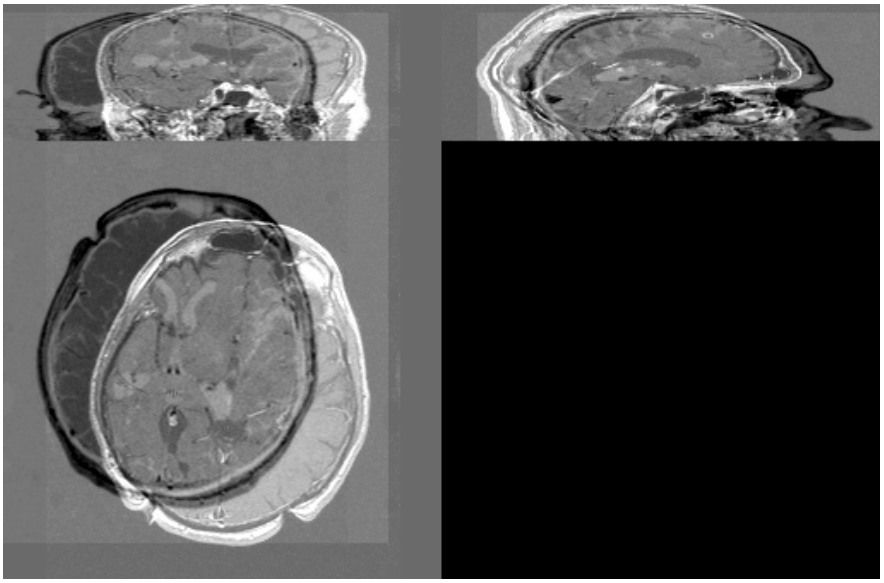
$$A = \begin{pmatrix} 1 & 0 & 0 & 30 \\ 0 & 1 & 0 & 20 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



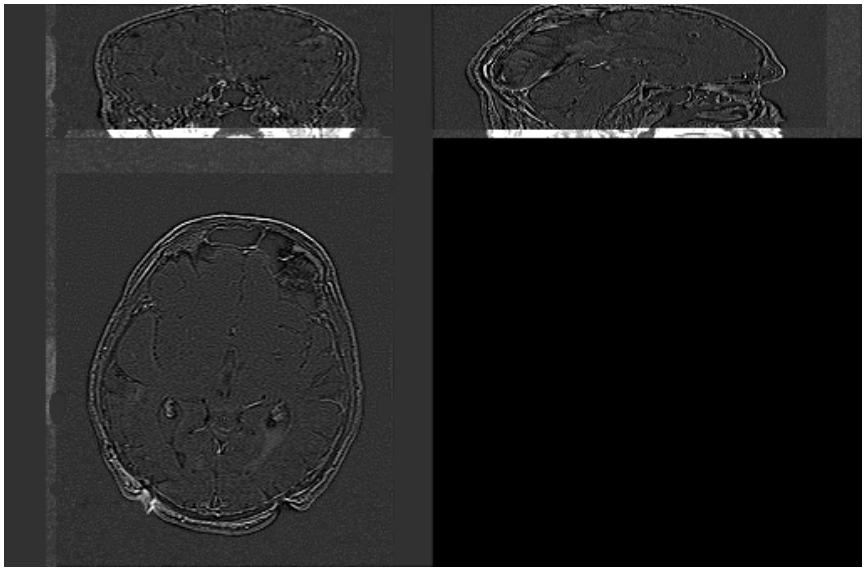
Obrázek 25: Originální obraz



Obrázek 26: Obraz posunutý o [30,20,10]



Obrázek 27: Pohled do všech 3 rovin před registrací



Obrázek 28: Pohled do všech tří rovin po registraci

Výsledkem této registrace byla matice transformace:

$$A = \begin{pmatrix} 0{,}9996 & 0{,}00164 & -0{,}00164 & -29{,}8888 \\ -0{,}0001 & 0{,}99986 & -0{,}00352 & -20{,}2491 \\ -0{,}00202 & 0{,}00638 & -1{,}00379 & -9{,}26714 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

S chybou metriky 723,383 a malou chybu posunu. Nalezená transformace je transformací pootočeného obrazu do původního obrazu.

Při obráceném postupu (prohození vstupů registrace a tak se změní směr registrace) se snažíme znovu nalézt zadanou transformaci, dojdeme k řešení:

$$A = \begin{pmatrix} 1.00069 & -0.00129 & 0.00167 & 30.0831 \\ 0.00125 & 1.00033 & 0.00805 & 20.6866 \\ -0.00825 & -0.0142 & 0.99096 & 7.87202 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ta obsahuje větší chybu v ose Z. A projeví se to i na chybě metriky, která je v tomto případě 835,31.

6.2.2. Rotace

Stejně jako u posunu byly pokusy rozděleny do 5 pásem, tak i rotace je rozdělena do pěti pásem (jednotkou jsou úhlové stupně):

- 0° – 2°
- 2° – 5°
- 5° – 10°
- 10° – 45°
- 45° – 50°

Při rotaci je jednotkou úhlový stupeň, který platí stejně jak v souřadnicovém sytému pixelů, tak fyzickém prostoru.

6.2.2.1. Časová náročnost registrace rotace

Z časového hlediska je rotace náročnější operací než translace. Je to dáno mnoha faktory. Jedním z nich je, že rotace ovlivňuje více parametrů transformace. Dalším důvodem je menší přesnost interpolátoru při rotaci. To ovlivňuje především počet iterací, kdy je algoritmus blízko, ale díky rotaci je lokální extrém nižší než při pouhém posunutí. V tabulkách (Tabulka 9 a Tabulka 10) je vidět stejný efekt jako při translaci:

- Čas potřebný k registraci se zvyšuje ve vztahu ke zvyšujícímu se šumu v obraze.
- Čas a počet iterací se zvyšuje spolu se zvyšující se náročností úlohy. Při velkých úhlech rostou časové nároky.

Pásmo rotace[°]	σ=0 [s]	σ=10 [s]	σ=20 [s]	σ=50 [s]	σ=100[s]
0-2	15,88	17,34	17,06	17,99	17,14
2-5	35,18	34,90	35,69	35,31	36,60
5-10	58,02	59,67	57,12	57,38	53,55
10-45	66,04	63,92	63,54	69,64	63,77
45-90	73,64	80,83	72,88	71,28	78,87

Tabulka 9: Průměrný čas registrace pro různé velikosti rotace a úrovně šumu σ

Pásmo rotace[°]	σ=0	σ=10	σ=20	σ=50	σ=100
0-2	16	18	18	18	18
2-5	36	35	36	36	37
5-10	59	60	58	58	54
10-45	67	64	64	70	64
45-90	74	81	73	72	79

Tabulka 10: Průměrný počet iterací pro různé velikosti rotace a úrovně šumu σ

A stejně jako u rotace je významným faktorem velikost obrazu. Při větším počtu pixelů musí být procházeno a porovnáváno větší množství dat.

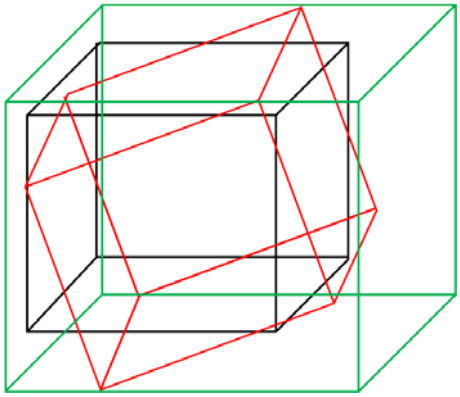
6.2.2.2. Úspěšnost a přesnost algoritmu pro registraci rotace

Úspěšnost registrace rotace shrnuje tabulka 11. Ve větších rotacích ve všech třech osách je algoritmus bohužel zcela neúspěšný.

Kritériem úspěchu jsou v tomto případě absolutní odchylka jednotlivých parametrů:

- Absolutní rozdíl zjištěného parametru matice a známého parametru matice musí být menší jak 0.05
- Absolutní rozdíl zjištěného parametru translace a známého parametru translace musí být menší jak 3mm.

Přesnost algoritmu při hledání rotace je nižší. To vzniklo především jako důsledek toho, že rotace může dostat část dat mimo obraz. Důvodem je, že po rotaci se obraz nerozšíří.



Obrázek 29: Rozšíření obrazu při rotaci obrazu

Na obrázku 29 se černý obraz otáčí do červeného. Pro zachování všech dat by bylo třeba zvětšit původní rozměry obrazu na rozměry zeleně označeného obrazu. Ale takové zvětšení obrazu neúměrně zatíží paměť a zvýší čas potřebný k registraci. Z tohoto důvodu je ponechán obraz v původní velikosti.

To způsobí, že změny o více jak 10° ve všech osách již způsobují to, že algoritmus není schopen najít rozumnou transformaci.

Další chybu přináší velikost voxelu v ose Z. Zde totiž interpolátor musí hodnotu dopočítávat z voxelů, které jsou ve větší vzdálenosti.

Pásmo rotace[°]	σ=0[%]	σ=10[%]	σ=20[%]	σ=50[%]	σ=100[%]
0-2	99	100	100	99	99
2-5	95	95	95	95	95
5-10	86	84	85	85	84
10-45	9	10	8	10	10
45-90	0	0	0	0	0

Tabulka 11: Úspěšnost registrace pro různé velikosti rotace a úrovně šumu σ

Pásmo rotace[°]	**σ=0**	**σ=10**	**σ=20**	**σ=50**	**σ=100**
0-2	507,25	500,52	614,90	1206,30	3055,40
2-5	1098,20	1086,19	1175,03	1737,10	3539,88
5-10	2297,09	2491,23	2560,56	3186,61	4955,97
10-45	23072,01	23239,94	22384,90	23388,40	23524,52
45-90	20755,82	20446,75	20077,94	20441,24	21748,79

Tabulka 12: Průměrná konečná hodnota metriky pro různé velikosti rotace a úrovně šumu σ

Při hledání transformace se algoritmus snaží velkou změnu rotace kompenzovat posunutím a výsledkem je transformace, která neodpovídá skutečné transformaci mezi obrazy.

Pro lepší zhodnocení je dobré použít průměrnou kvadratickou odchylku, kterou spočítáme jako průměr vzdálenosti souřadnic jednotlivých pixelů obrazu transformovaných podle známé transformace od souřadnic pixelů obrazů transformovaných podle zjištěné transformace:

$$e = \frac{1}{XYZ} \sum_{s=1}^{XYZ} \left\| A_{znam}\, \vec{x}_s - A_{zjist}\, \vec{x}_s \right\| \tag{30}$$

Kde X, Y a Z jsou rozměry obrazu v jednotlivých osách.

Výsledkem je tabulka 13 shrnující zobrazující toto kritérium podle pásem rotace a podle šumu v obraze

Pásmo rotace[˚]	σ=0[mm]	σ=10[mm]	σ=20[mm]	σ=50[mm]	σ=100[mm]
0-2	1,70	1,52	1,57	1,74	1,59
2-5	3,37	3,35	3,30	3,27	3,25
5-10	7,07	7,20	7,21	7,76	7,16
10-45	132,82	131,14	135,01	135,28	131,27
45-90	274,30	276,92	279,94	284,22	296,72

Tabulka 13: Průměrná velikost vektoru mezi bodem transformovaným podle známých parametrů a bodem transformovaným podle zjištěných parametrů pro různé úrovně šumu σ

6.2.2.3. Ukázka registrace rotace

Na ukázku byla provedena registrace obrazu, který byl pootočen o 15˚ v roviny XY a o 3˚ v rovině YZ.

Matice transformace potom vypadá:

$$A = \begin{pmatrix} 0.96460 & -0.25881 & 0.05055 & 0 \\ 0.25846 & 0.96592 & -0.01354 & 0 \\ 0.05233 & 0 & 0.99863 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Po provedení registrace Původní obraz vs. Pootočený obraz jsme nalezli tuto matici:

$$A = \begin{pmatrix} 0.96447 & 0.258991 & 0.05324 & 0.10526 \\ -0.25986 & 0.96633 & -0.00181 & -0.09432 \\ -0.05461 & -0.01130 & 0.99421 & -0.12555 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

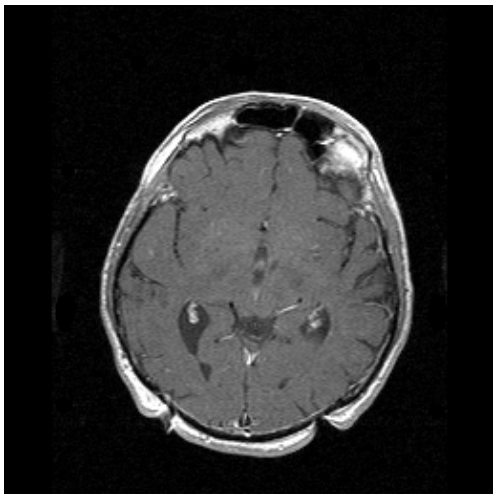
Chyba mezi obrazy určená metrikou je 730,005.

A pokud se provádí registrace ve směru Pootočený obraz vs Původní obraz jsou nalezeny tyto parametry:

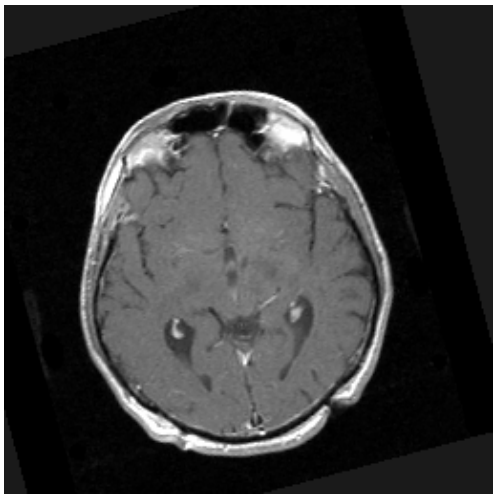
$$A = \begin{pmatrix} 0.96557 & -0.25646 & -0.04866 & 0.25859 \\ 0.25499 & 0.96655 & -0.01281 & 0.04668 \\ 0.05048 & -0.00032 & 0.99909 & -0.01424 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

V tomto případě je chyba pouze 87,4712.

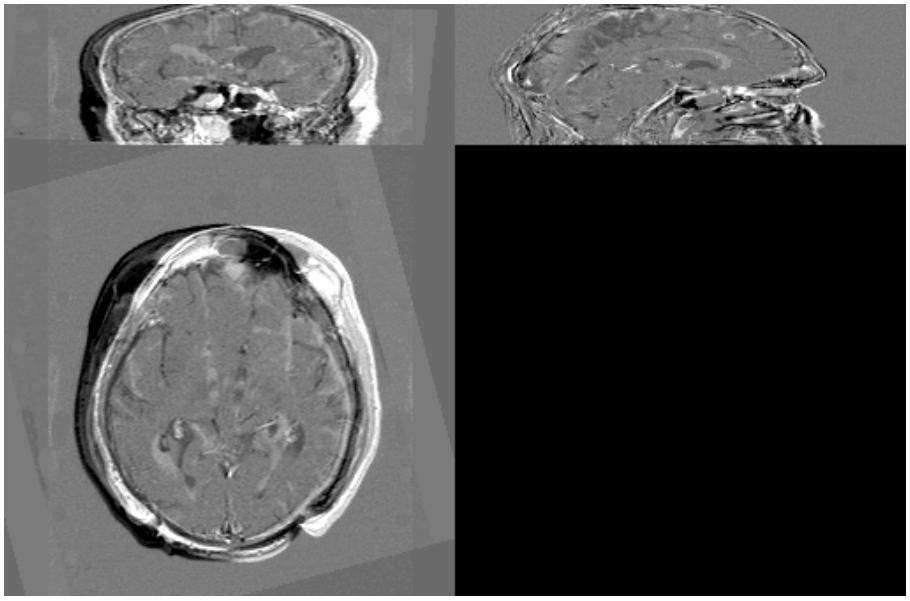
Obrázky vzniklé k této registraci jsou níže uvedené.



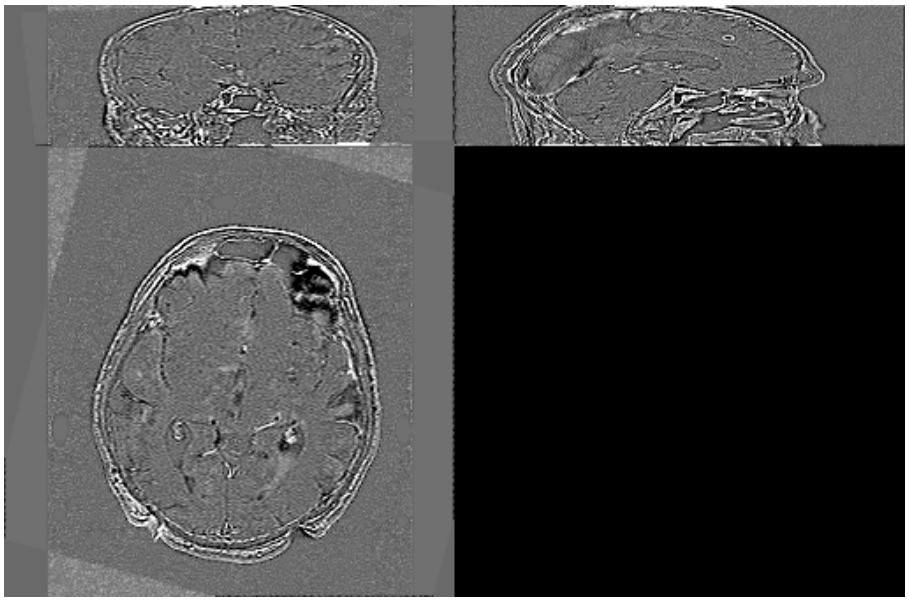
Obrázek 30:Originální obraz



Obrázek 31: Pootočený obraz o 15 a 3 úhlové stupně



Obrázek 32: Porovnání obrazů před registrací



Obrázek 33: Porovnání obrazů po registraci

6.2.3. Posunutí, rotace a změna měřítka

V poslední části experimentů se spojily předchozí dvě možnosti registrace a přidala se jemná změna měřítka.

Pokusy byly opět rozděleny do pěti pásem:

- 0 – 1 px, 0˚ – 2˚
- 1 – 2 px, 2˚ – 5˚
- 2 – 6 px, 5˚ – 10˚
- 5 – 10 px, 10˚ – 45˚
- 10 – 50 px, 45˚ – 50˚

Měřítko *M* bylo náhodně měněno pro všechny pokusy ve stejném intervalu:

$$M \in \langle -3;+3 \rangle$$

Měřítko se mění v každé dimenzi jinak.

6.2.3.1. Časová náročnost registrace

Změna všech parametrů je nejobtížnější a projevuje se i na časové náročnosti hledání řešení.

Překvapivý výsledek, že v posledním pásmu registrace proběhla rychle je způsobeno tím, že algoritmus zcela ztratil hledanou cestu a nic nenašel. To je vidět z následujících tabulek (Tabulka 14 a Tabulka 15), které ukazují, že provede vysoký počet kroků v krátkém čase, kdy se snaží prohledat co nejširší prostor.

Pásmo posunu[px]	Pásmo Rotace[˚]	σ=0 [s]	σ=10 [s]	σ=20 [s]	σ=50 [s]	σ=100 [s]
0-1	0-2	24,48	23,73	23,93	23,61	24,57
1-2	2-5	49,21	47,80	49,11	50,04	47,69
2-6	5-10	75,64	76,84	76,19	79,72	79,93
5-10	10-45	101,76	97,92	96,60	97,59	101,87
10-50	45-90	54,45	59,42	50,81	56,59	54,47

Tabulka 14: Průměrný čas jedné registrace pro různé úrovně šumu σ

Pásmo posunu[px]	Pásmo Rotace[˚]	σ=0	σ=10	σ=20	σ=50	σ=100
0-1	0-2	26,97	26,62	26,62	26,74	27,49
1-2	2-5	63,1	62,87	63,64	64,78	62,76
2-6	5-10	134,38	135,63	134,2	136,79	137,68
5-10	10-45	323,28	299,95	314,78	312,23	330,62
10-50	45-90	308,15	307,31	298,77	310,09	312,51

Tabulka 15: Průměrný počet iterací při registraci pro různé úrovně šumu σ

6.2.3.2. Úspěšnost a přesnost algoritmu registrace

Stejně jako v předchozím případě je úspěch algoritmu závislý na velikosti posunutí a částečně i na šumu v obraze. Stejně jako v předchozích případech, při velkých posunech a rotacích je algoritmus neúčinný.

Metrika počítá rozdíl obrazů podle vzorce (15). Průměrné hodnoty spočítané metrikou po skončení registrace jsou zobrazeny v tabulce (Tabulka 17).

V posledním řádku je vidět nízká hodnota oproti předcházejícímu pásmu. To je způsobeno tím, že v tu chvíli mají velmi málo společných bodů, ze kterých se tato veličina počítá. V tu chvíli lícují z větší části jen okraje obrazu, které jsou podobné a jejichž intenzity pixelů se pohybují v rozmezí 0 – 200.

Pásmo posunu[px]	Pásmo Rotace[°]	σ=0 [%]	σ=10 [%]	σ=20 [%]	σ=50 [%]	σ=100 [%]
0-1	0-2	100	100	100	100	100
1-2	2-5	100	99	100	99	99
2-6	5-10	64	64	64	63	63
5-10	10-45	10	10	10	9	9
10-50	45-90	0	0	0	0	0

Tabulka 16:Úspěšnost registrace pro různé úrovně šumu σ

Kritériem úspěchu pro tabulku 16 je v tomto případě absolutní odchylka jednotlivých parametrů:

- Absolutní rozdíl zjištěného parametru matice a známého parametru matice musí být menší jak 0.05
- Absolutní rozdíl zjištěného parametru translace a známého parametru translace musí být menší jak 3mm.

Pásmo posunu[px]	Pásmo Rotace[°]	σ=0	σ=10	σ=20	σ=50	σ=100
0-1	0-2	618,79	652,17	725,28	1264,32	3118,07
1-2	2-5	1103,58	1072,04	1151,83	1626,86	3526,92
2-6	5-10	2660,53	2841,33	2824,19	3476,95	5214,19
5-10	10-45	22761,64	22930,37	22956,68	23713,07	24146,63
10-50	45-90	18131,95	16932,13	15570,24	15733,44	14673,3

Tabulka 17: Průměrná konečná hodnota metriky pro různé úrovně šumu σ

Stejně jako v předchozích případech je dobré použít kritérium vzdálenosti mezi bodem transformovaným podle známé transformace a bodem transformovaným podle zjištěných parametrů pomocí registrace. Výsledek opět shrnuje

Pásmo posunu[px]	Pásmo Rotace[˚]	σ=0	σ=10	σ=20	σ=50	σ=100
0-1	0-2	2,01	1,99	1,99	2,01	1,99
1-2	2-5	3,85	3,89	3,76	3,61	3,84
2-6	5-10	10,68	10,61	10,35	10,67	9,72
5-10	10-45	128,59	132,21	132,54	136,03	134,16
10-50	45-90	262,89	264,76	264,85	270,71	270,90

Tabulka 18: Průměrná velikost vektoru mezi bodem transformovaným podle známých parametrů a bodem transformovaným podle zjištěných parametrů pro různé úrovně šumu σ

6.2.3.3. Ukázka registrace

Na ukázku byla provedena registrace obrazu, který byl posunut [14; 0; 3], pootočen o 15˚ v roviny XY, o 7˚ v rovině YZ a o 6˚ v rovině XZ. Zmenšen o 2% v ose X a zvětšen v dimenzi osy Z.

Matice transformace potom vypadá:

$$A = \begin{pmatrix} 0.93955 & -0.26970 & -0.09091 & 14 \\ 0.25175 & 0.95733 & -0.13365 & 0 \\ 0.11943 & 0.10374 & 0.99698 & 3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Výsledkem registrace Původní obraz vs. Pootočený obraz je matice

$$A = \begin{pmatrix} 0.97912 & 0.26188 & 0.12166 & -14.3227 \\ -0.2677 & 0.95966 & 0.09632 & 3.08325 \\ -0.09086 & -0.12182 & 0.97380 & -1.0945 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{s}$$

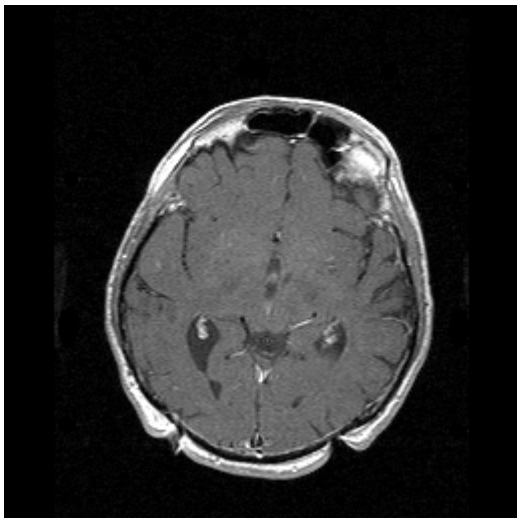
Chyba udávaná metrikou je 1093,8.

A pokud se provádí registrace ve směru Pootočený obraz vs Původní obraz jsou nalezeny tyto parametry:

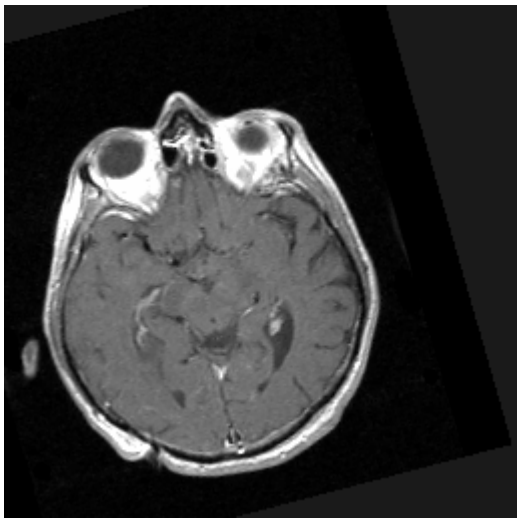
$$A = \begin{pmatrix} 0.94038 & -0.26951 & -0.08945 & 14.1777 \\ 0.25160 & 0.95819 & -0.12843 & 0.38946 \\ 0.11304 & 0.09441 & 0.99199 & 1.91622 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

V tomto případě je chyba 448,698.

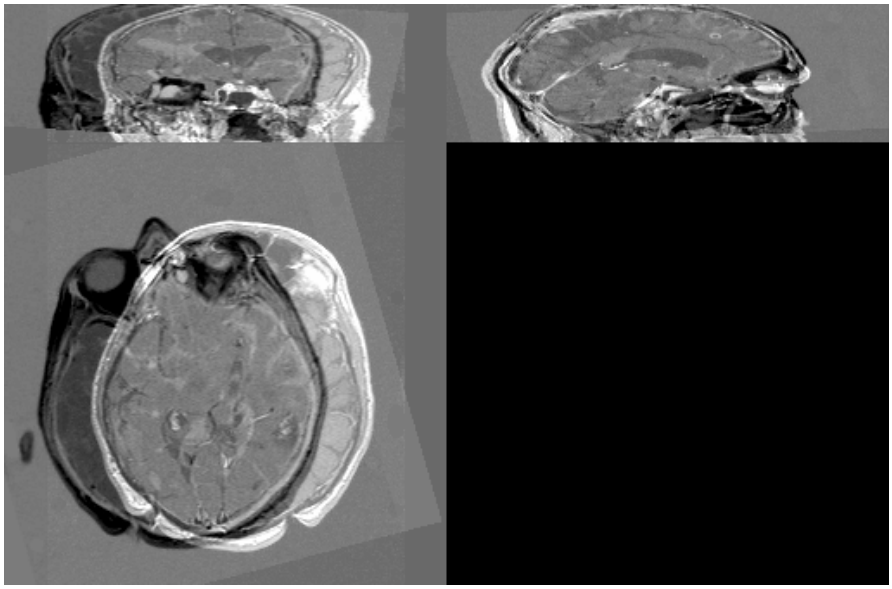
Obrázky vzniklé k této registraci jsou níže uvedené.



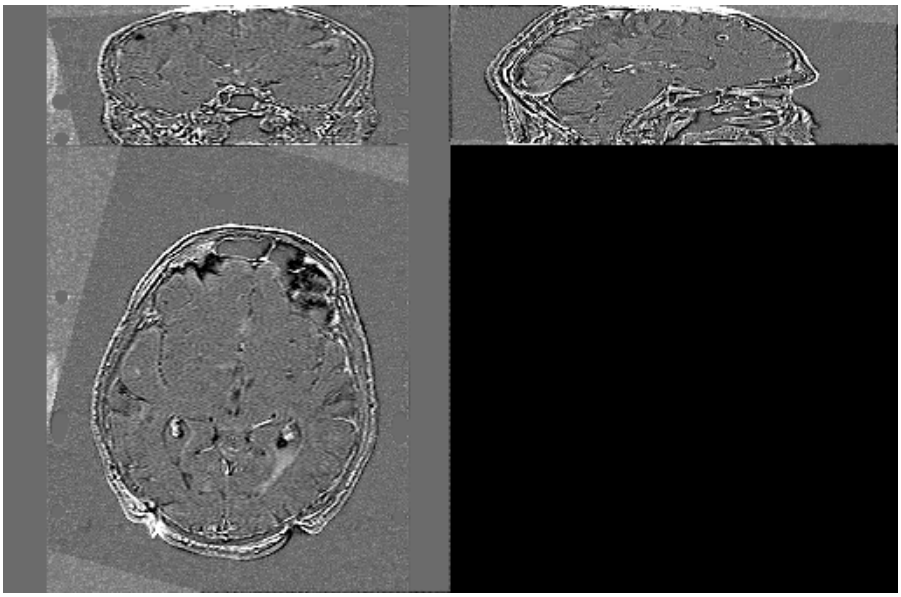
Obrázek 34: Originální obraz



Obrázek 35: Transformovaný obraz



Obrázek 36: Porovnání před registrací



Obrázek 37: Porovnání obrazů po registraci

6.3. Porovnání jednotlivých registrací nad vytvořenými daty

Lze srovnat především časovou náročnost jednotlivých experimentů. Tato data spojuje tabulka 19, kde jsou převzaty jednotlivé tabulky s časy registrací.

Posun XY[mm]	Posun Z[mm]	Rotace [°]	σ=0 [s]	σ=10 [s]	σ=20 [s]	σ=50 [s]	σ=100 [s]
0-0,98	0-2	0	13,07	13,50	13,74	13,41	14,46
0,98–1,96	2-4	0	17,60	17,97	18,27	18,99	19,83
1,96-4,9	4-10	0	45,35	45,33	45,60	44,85	44,06
4,9-9,8	10-20	0	65,13	57,70	67,30	63,25	65,06
9,8-49	20-100	0	75,69	77,57	76,77	74,19	79,04
0	0	0-2	24,48	23,73	23,93	23,61	24,57
0	0	2-5	49,21	47,80	49,11	50,04	47,69
0	0	5-10	75,64	76,84	76,19	79,72	79,93
0	0	10-45	101,76	97,92	96,60	97,59	101,87
0	0	45-90	54,45	59,42	50,81	56,59	54,47
0-0,98	0-2	0-2	24,48	23,73	23,93	23,61	24,57
0,98–1,96	2-4	2-5	49,21	47,80	49,11	50,04	47,69
1,96-4,9	4-10	5-10	75,64	76,84	76,19	79,72	79,93
4,9-9,8	10-20	10-45	101,76	97,92	96,60	97,59	101,87
9,8-49	20-100	45-90	54,45	59,42	50,81	56,59	54,47

Tabulka 19: Spojení jednotlivých tabulek s délkou trvání registrace

V této tabulce 19 je vidět, jak roste časová náročnost s rostoucí hladinou šumu v obraze a s rostoucí složitostí úlohy, která je určena zvětšující se vzdáleností obrazů. A při registraci všech parametrů v největší možná vzdálenosti je čas zkrácen, protože algoritmus brzo zjistí, že není schopen najít odpovídající transformaci.

Dalším kritériem pro srovnání může být úspěšnost registrace v jednotlivých sériích. V přehledu to zobrazuje tabulka 20.

Kritériem úspěchu je v tomto případě absolutní odchylka jednotlivých parametrů:

- Absolutní rozdíl zjištěného parametru matice a známého parametru matice musí být menší jak 0.05
- Absolutní rozdíl zjištěného parametru translace a známého parametru translace musí být menší jak 3mm.

Algoritmus je nejúspěšnější při zjišťování transformací, které obsahují pouze translaci. Jeho úspěšnost je ovlivněna rostoucí vzdáleností registrovaných obrazů, ale šum nemá žádný vliv. To je především důsledek multi-resolution registraci, kdy je obraz zmenšen a vliv šumu se při převzorkování obrazu na menší obraz také zmenší.

Posun XY[mm]	Posun Z[mm]	Rotace [°]	σ=0 [%]	σ=10 [%]	σ=20 [%]	σ=50 [%]	σ=100 [%]
0-0,98	0-2	0	99	99	99	99	99
0,98–1,96	2-4	0	100	100	100	100	100
1,96-4,9	4-10	0	81	81	80	80	80
4,9-9,8	10-20	0	31	31	31	31	31
9,8-49	20-100	0	5	5	4	4	5
0	0	0-2	99	100	100	99	99
0	0	2-5	95	95	95	95	95
0	0	5-10	86	84	85	85	84
0	0	10-45	9	10	8	10	10
0	0	45-90	0	0	0	0	0
0-0,98	0-2	0-2	100	100	100	100	100
0,98–1,96	2-4	2-5	100	99	100	99	99
1,96-4,9	4-10	5-10	64	64	64	63	63
4,9-9,8	10-20	10-45	10	10	10	9	9
9,8-49	20-100	45-90	0	0	0	0	0

Tabulka 20: Spojení jednotlivých tabulek s úspěšností registrace

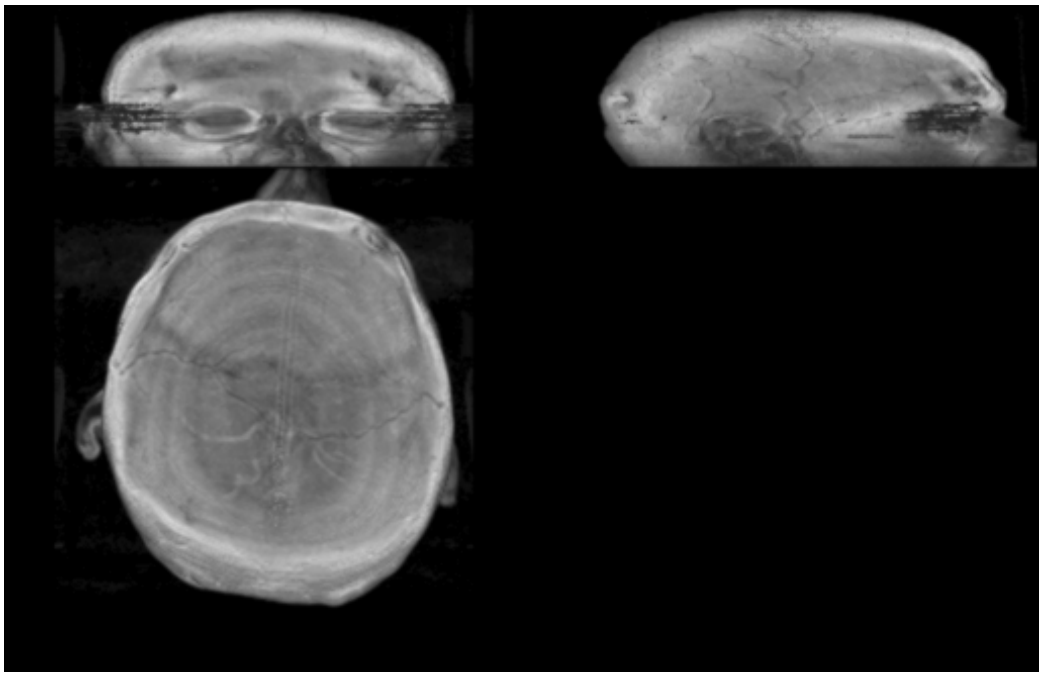
6.4. Reálná data

Program má především sloužit nad reálnými daty. Pro použití je vhodná shodná velikost obrazů. Registrované obrazy by měly mít stejné intenzity pixelů.

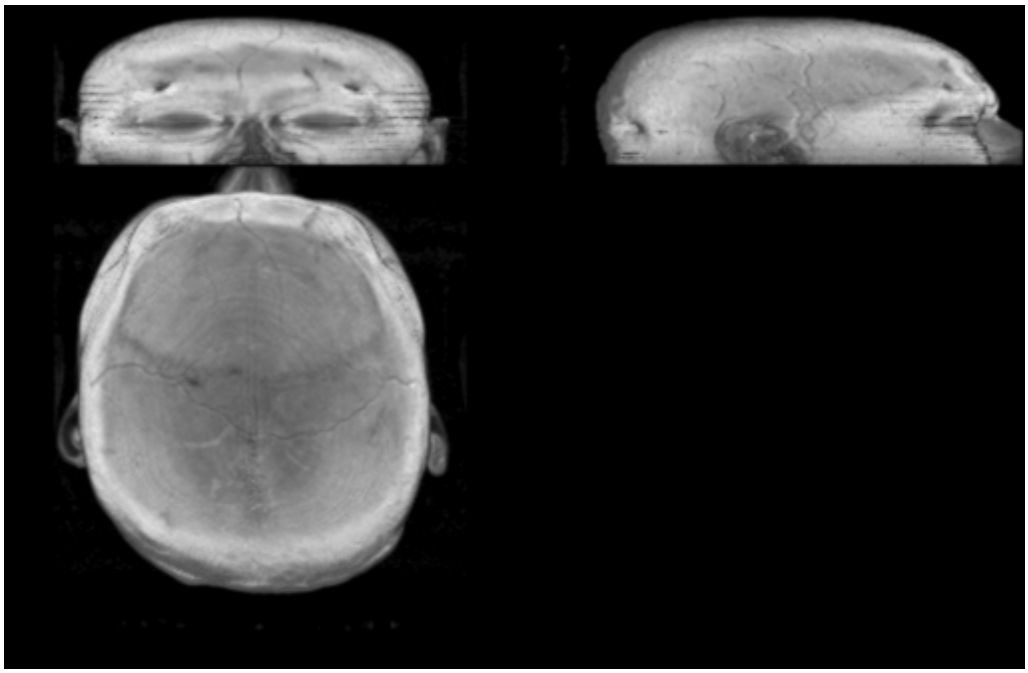
Dodaná data nemocnicí Na Homolce obsahují 9 různých pacientů. Jako příklad je uvedena registrace dvou vyšetření s ročním odstupem.

- `.\01_Data\02_Pac\Study__20060901__072227.906000\Series__20060901__072755.187000\1.3.12.2.1107.5.2.6.22428.30000006083114333151500003305\`
- `.\01_Data\02_Pac\Study__20061113__074215.156000\Series__20061113__074700.562000\1.3.12.2.1107.5.2.6.22428.30000006111315152739000000138\`

Nejprve byla tato data spojena a potom registrována.

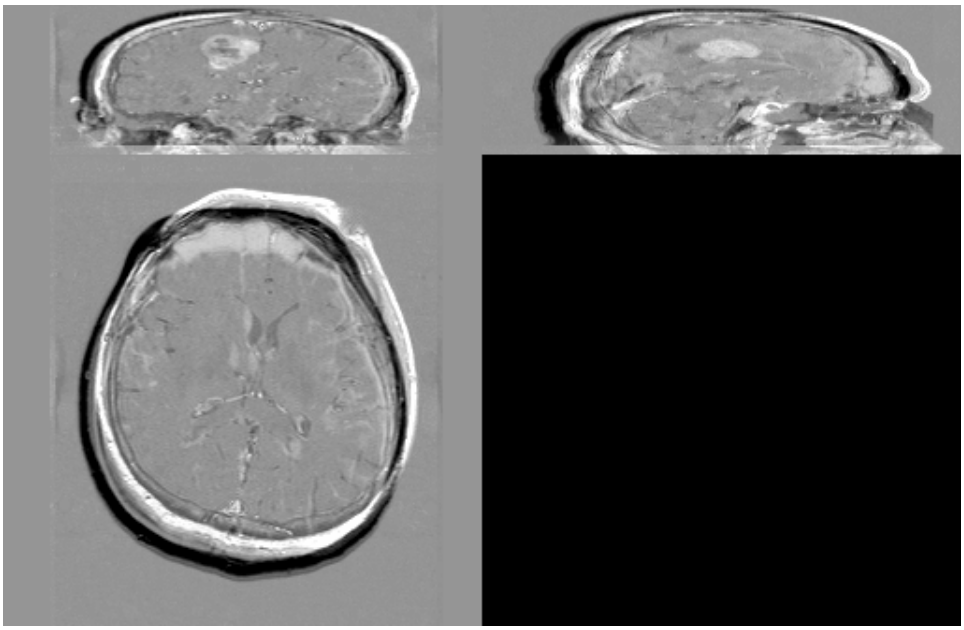


Obrázek 38: Vyšetření 1



Obrázek 39: Vyšetření 2

Pokud se podíváme na rozdílový obraz těchto dvou vyšetření, můžeme si všimnout, že se jedná především o translaci a rotaci.

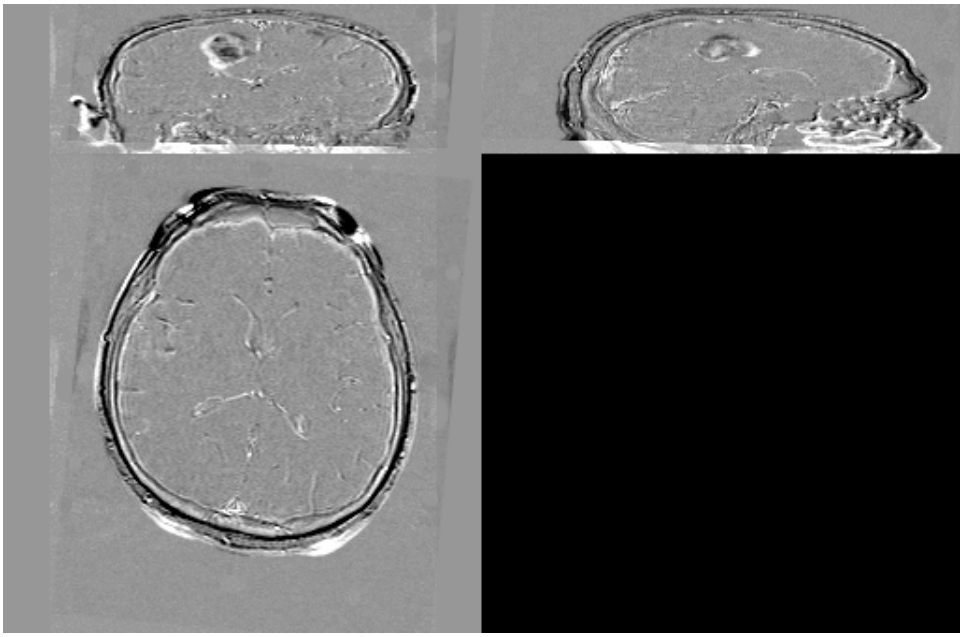


Obrázek 40: Rozdíl Vyšetření 1 a Vyšetření 2 před registrací

Černé části jsou z jednoho obrazu a bíle zdůrazňují druhý. Byly opět provedeny dvě registrace. V první registraci je použit jako pevný obraz Vyšetření 1. Výsledkem je matice:

$$A = \begin{pmatrix} 1.01798 & 0.094094 & -0.03834 & -7.24283 \\ -0.09058 & 1.01138 & 0.07109 & 18.004 \\ 0.03690 & -0.05864 & 1.00185 & -0.81115 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

a chyba metriky 10900.4. Obrazově znázorněný rozdíl obrazů po registraci je na obrázku (viz Obrázek 41).



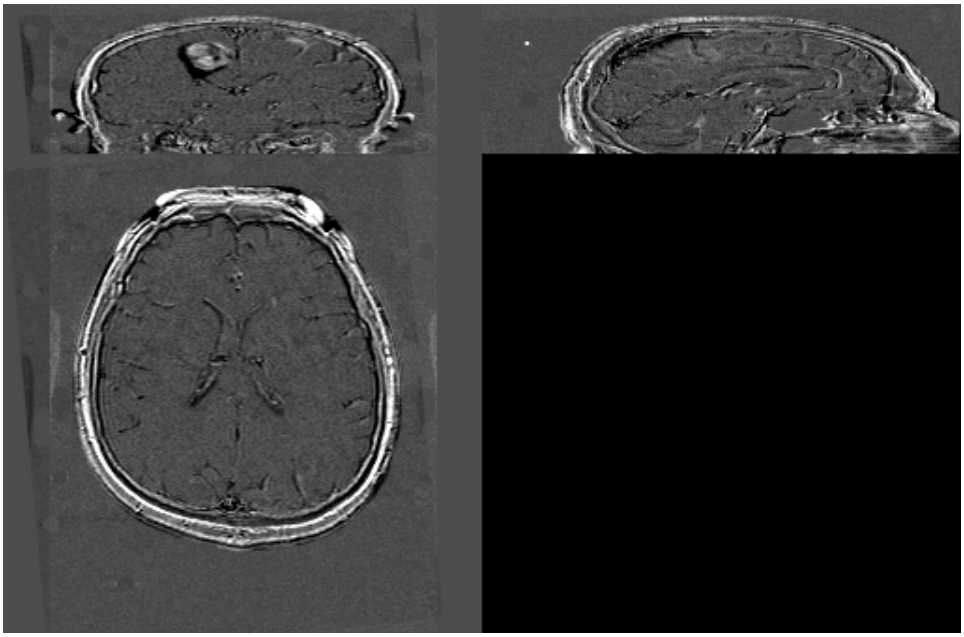
Obrázek 41: Rozdíl Vyšetření 1 a Vyšetření 2 po registraci (pevný obraz -Vyšetření 1)

Další registrací bylo Vyšetření 2 považováno za pevné a Vyšetření 1 se muselo přizpůsobit. Výsledkem byla matice:

$$A = \begin{pmatrix} 0.97312 & -0.08816 & 0.04249 & 8.64997 \\ 0.08972 & 0.97650 & -0.0773 & -18.0266 \\ -0.02920 & 0.071965 & 0.98606 & -0.19541 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

Chyba spočítaná metrikou je tentokrát 11194,9.

Obrazově je to na níže uvedeném obraze (Obrázek 42)



Obrázek 42: Rozdíl Vyšetření 1 a Vyšetření 2 po registraci (pevný obraz -Vyšetření 2)

7. Popis programů

Kapitola obsahuje popis programů na přiloženém optickém médiu. Zdrojové kódy se nacházejí v adresáři na cestě:

`.\03_Prog`

7.1. Obecné prvky

7.1.1. Načítání a ukládání

Data dodaná Nemocnicí Na Homolce byla rozdělena do jednotlivých adresářů, obsahujících soubory formátu DICOM s jednotlivými řezy vyšetření. Bohužel velmi často nesouhlasilo číslo vyšetření, a proto nebylo možno využít třídu:

Itk::GDCMSeriesFileNames

Proto jsme vytvořili jednoduchou třídu

```
template <class TImage>
  class ITK_EXPORT ReadWrite : public Object
```

Ta je schopná načíst všechny soubory adresáře a maskou *.dcm.

V první fázi načte dané soubory do pole, kde si uchovává kromě obrazové informace, také informaci o poloze jednotlivých řezů. K tomu slouží jednoduchá struktura:

```
//strukturka na obraz a polohu(poloha je pro razeni)
typedef typename struct SFileInfo
{
  typename TImage::Pointer PImage;
  float location;
}
```

Poloha řezu `location` se načítá z hlavičky souboru DICOM. Jedná se o tag `slice location`.

Soubory jsou do pole načteny v pořadí podle názvu, což nemusí odpovídat skutečnému pořadí řezů v obrazu. Proto ve druhém kroku se pole setřídí podle polohy, k tomu se použije jednoduché bublinové třídění pole. Pole je malé (řádově desítky prvků) a není třeba optimalizovat tuto úlohu.

Po setřídění jsou jednotlivé řezy překopírovány do výsledného 3D obrazu.

Velikost voxelu je v osách X a Y přebrána z velikosti původních obrazů. Ve směru osy Z je velikost načtena z hlavičky původních souborů (tag `slice thickness`), nebo je spočítána jako rozdíl polohy dvou sousedních řezů.

Počátek obrazu, poloha obrazu ve fyzickém rozměru je odvozena od prvního z řezů. Tedy bodu [0; 0; 0] je přiřazena fyzická poloha bodu [0; 0] z prvního řezu.

7.2. Konsolové aplikace

Miniaplikace vznikly pro jednodušší testování jednotlivých úloh. Navíc mohou sloužit jako pomocné programy.

7.2.1. Odstranění rámu (predzprac)

Jedná se o aplikaci spouštěnou z příkazového řádku. Má především dvě funkce:

- Načíst adresář DICOM a vytvořit z jeho souborů jeden soubor obsahující 3D obraz.
- Odstranit stereotaktický rám z obrazu.

Využívá třídu pro načítání popsanou v kapitole 7.1.1 a následně algoritmus popsaný v kapitole 4.2.

Ovládání programu `predzprac` je pomocí příkazové řádky

Příkaz	Popis
`predzprac -h`	Jednoduchá nápověda
Povinné	
`predzprac -r [soubor]`	Otevře pro následné zpracování soubor na zadané cestě
`predzprac -w [soubor]`	Zapíše na zvolenou cestu originální obraz načtený v paměti
Nepovinné	
`predzprac -r -d [adresář]`	Načte pro následné zpracování celý adresář s DICOM soubory a vytvoří z něj v paměti 3D obraz
`predzprac -w -s [adresář]`	Zapíše do zvoleného adresáře originální obraz 3D načtený v paměti a rozdělený po jednotlivých řezech v rovině obrazu XY
`predzprac -w -rm [soubor]`	Odstraní značky stereotaktického rámu z obrazu

Tabulka 21: Ovládání programu predzprac

Příklad:

```
predzprac -r -d [adresar] -w -rm [soubor]
```

Provede načtení celého adresáře s DICOM obrazy, odstraní značky stereotaktického rámu a zapíše výsledný obraz do výstupního souboru.

7.2.2. Otáčení obrazu (otoc)

Jednoduchý program příkazové řádky, který slouží k posunutí, natočení, změně měřítka a zkosení natáčeného obrazu.

Ovládání je snadné

otoc `zdrojovy_soubor vysledny_soubor soubor_parametry`

Program načte zdrojový soubor a soubor s parametry. Zdrojový soubor je potom transformován podle získaných parametrů.

Soubor s parametry musí mít tento formát:

1. Řádek - obsahuje přepínač mezi zadáním pomocí matice nebo pomocí parametrů.

Pokud je nastaven na 0 napíše se do souboru matice ve tvaru:

A_{11} A_{21} $A3_1$ T_{11}
A_{12} A_{22} A_{32} T_{11}
A_{13} A_{23} A_{33} T_{11}

Pokud je přepínač nastaven na 1, nastavují se jednotlivé parametry pomocí vektorů uvedených v tabulce 22.

2. řádek	Translace, ve všech třech dimenzích [mm]
3. řádek	Rotace, ve všech třech dimenzích [°]
4. řádek	Změna měřítka, ve všech třech • dimenzích o <1zmenšení, o >1zvětšení
5. řádek	Zkosení, ve všech dimenzích

Tabulka 22: Seznam vektorů parametrů pro ovládání programu otoc

Tento program využívá naprogramovanou třídu:

```
template <class TImage>
  class ITK_EXPORT Transform3DImage :
    public ImageToImageFilter<TImage, TImage>
```

Ta umožňuje zadat parametry afinní transformace pomocí matice a vektoru, pomocí vektorů jednotlivých vlastností transformace (translace, rotace, změny měřítka, zkosení) nebo se parametry zadávají vektorem 12 parametrů (prvních 9 čísel je matice, zbylá 3 jsou vektorem translace).

Dále je možné zvolit způsob interpolace bodů. Na výběr jsou dvě možnosti:

- Lineární (pro rychlé operace)
- Sinc (pro přesné aplikace)

7.2.3. Registrace

Program se spouští z příkazové řádky:

registruj `pevny_obraz pohyblivy_obraz počet_urovni vysledkove_soubory`

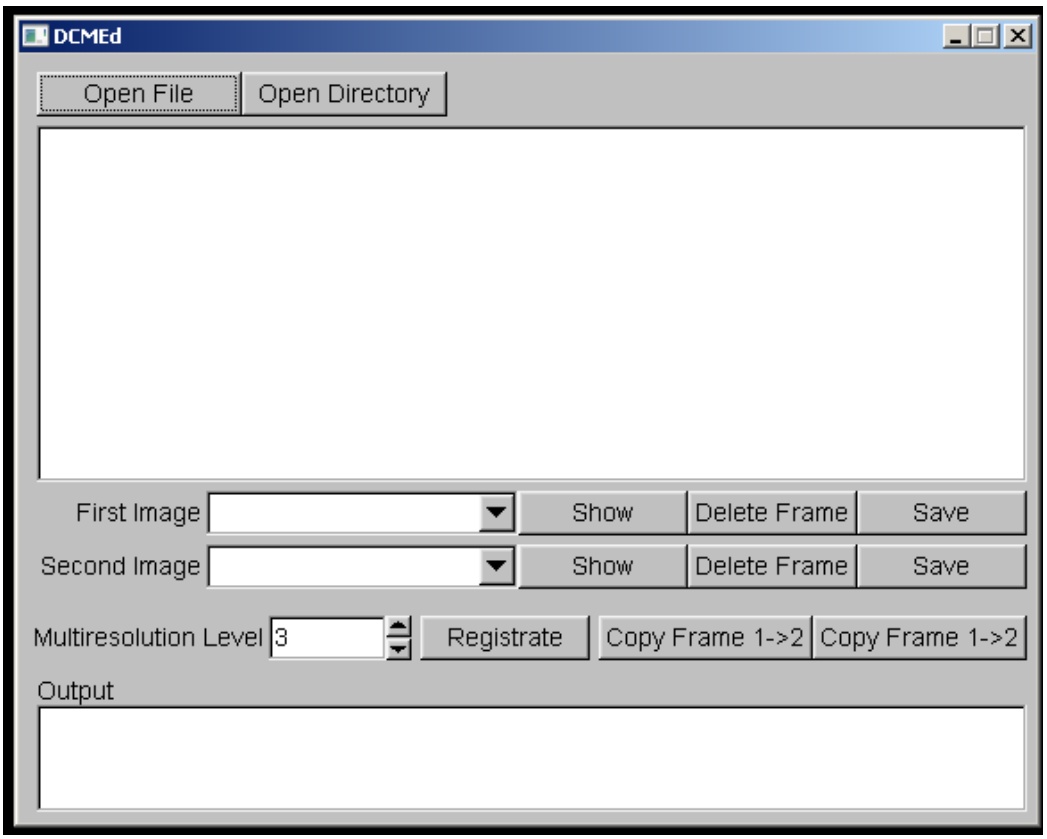
Vstupem jsou dva soubory formátu DICOM obsahující obrazová vyšetření. Výsledkem je výpis nalezené transformace a případně 3 soubory DICOM

- Porovnání vstupních obrazů před registrací
- Pohyblivý obraz natočený podle nalezené transformace
- Porovnání obrazů po registraci

7.3. Grafické prostředí

Pro zobrazení a jednoduchou práci s daty vzniklo malé grafické rozhraní DCMEd. Je napsáno pomocí FLTK a využívá funkce z konzolových aplikací.

Její hlavní okno je zobrazeno na obrázku 43Obrázek 43.



Obrázek 43: Hlavní okno aplikace DCMEd

Lze v něm načítat soubory, které jsou uloženy na disku a následně s nimi pracovat. Načítat je možno jednotlivé soubory DICOM nebo celé adresáře obsahující jednotlivé řezy vyšetření uložených v souborech DICOM.

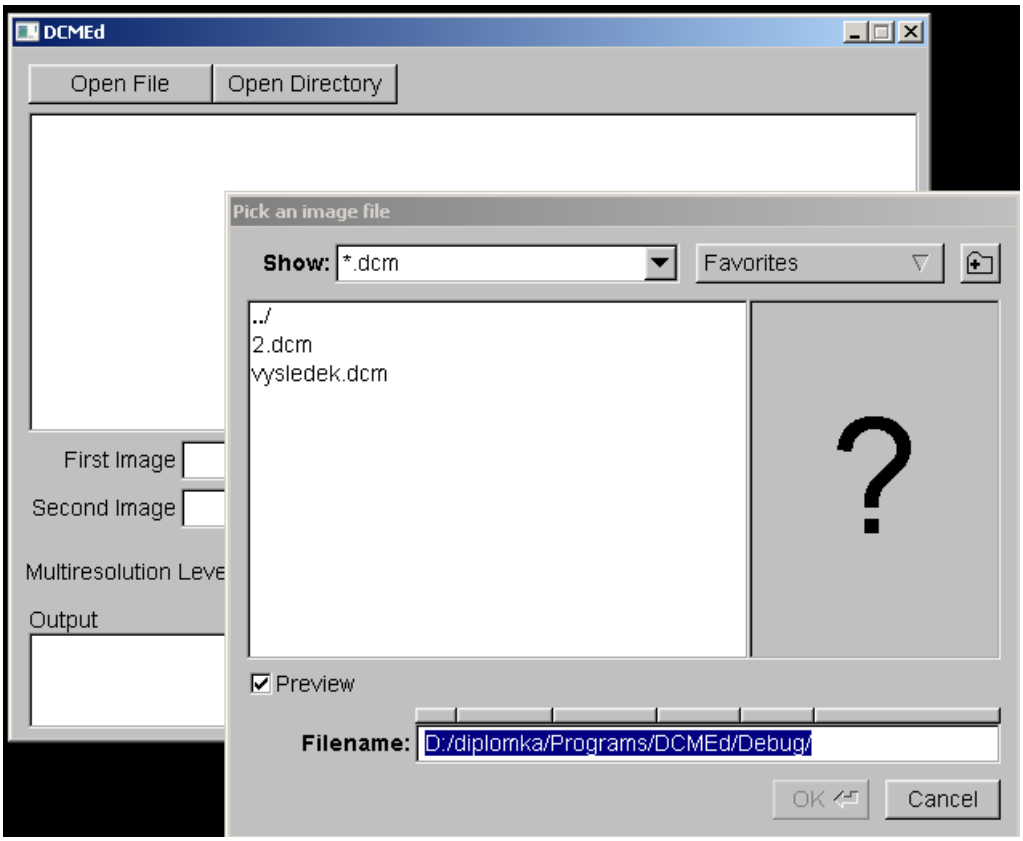
Mezi další akce zpracování načtených obrazů patří hlavně

- Smazání stereotaktického rámu
- Registrace obrazu
- Zkopírování stereotaktického rámu z jednoho obrazu do druhého

Pro registraci lze zvolit počet úrovní, ve kterých má registrace probíhat. Standardně je nastaven počet na 3 úrovně.

Navržené uživatelské prostředí využívá standardní dialogy na načítání a ukládání souborů, které poskytuje knihovna FLTK.

Pro zobrazení obrazových dat z DICOM souborů používá knihovnu `ITKFltkImageViewer`, která je k dispozici v InsightApplications. Tento balík je součástí ITK a dále využívá knihovnu OpenGL.



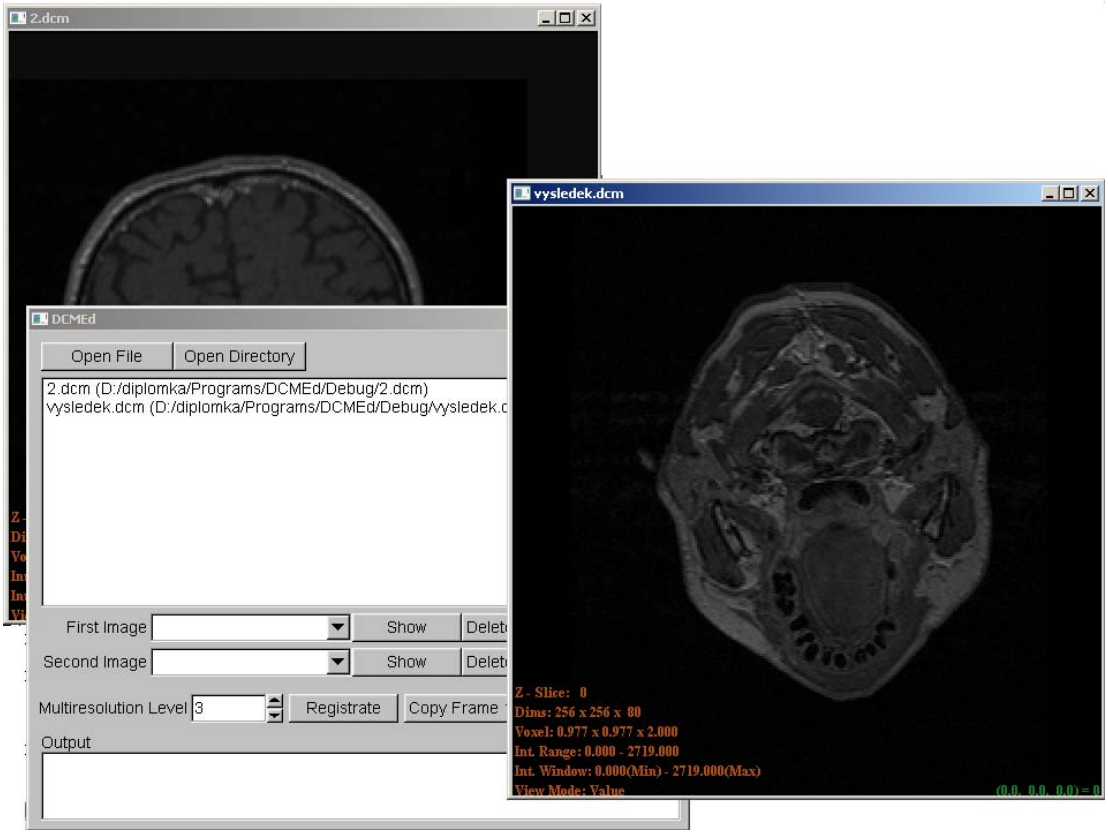
Obrázek 44: Načítání dat v DCMEd

Po načtení dat se v novém nemodálním okně zobrazí vyšetření. V hlavním okně přibude řádek s názvem a cestou k souboru.

Ovládání nového okna je vidět po stisknutí „h“.

Okno pro zobrazování dat je součástí podpůrných aplikací ITK, kde je pro propojení ITK a FLTK třída:

```
template <class ImagePixelType,
          class OverlayPixelType>
  class GLSliceView :
    public SliceView<ImagePixelType>,
    public Fl_Gl_Window
```



Obrázek 45: Zobrazení dat v DCMEd

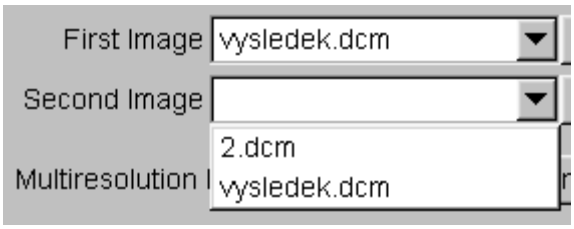
Základní ovládání je zobrazeno v tabulce 19.

Klávesa	***Funkce***
0	Prohlížet řezy po ose X
1	Prohlížet řezy po ose Y
2	Prohlížet řezy po ose Z
<	Procházení řezy dolů
<	Procházení řezy nahoru
h	Nápověda
+	Dvojnásobné zvětšení
-	Dvojnásobné zmenšní

Tabulka 23: Základní ovládání zobrazovacího okna DCMEd

7.3.1. Práce s daty

V hlavním okně si pomocí combo-boxu vybereme data, se kterými chceme pracovat, z nabízeného seznamu.



Obrázek 46: Výběr dat

Potom lze zvolit některým tlačítkem akci, kterou nad nimi chceme provést. Na výběr je celkem několik akcí:

- Zobrazit data (Tlačítko „Show“)
- Odstranit stereotaktický rám z obrazu (Tlačítko „Delete Frame“)
- Uložit na disk (Tlačítko „Save“)
- Obrazy registrovat (Tlačítko „Registrate“)
- Přenést stereotaktický rám z jednoho obrazu do druhého obrazu bez rámu (Tlačítka „Copy Frame 1->2“ a „Copy Frame 2->1“)

V posledním případě je z obrazu se stereotaktickým rámem, tento rám odstraněn, následně je provedena registrace obrazů. V dalším kroku je podle získaných parametrů transformován původní obraz rámu a poté je doplněn do obrazu bez rámu.

Doplnění rámu do obrazu provádí třída:

```
template< class TImage >
  class ITK_EXPORT AddStereotacticFrame :
    public Object
```

Vstupem jsou dva obrazy, které chceme sloučit. V tomto případě se jedná o obraz hlavy pacienta a transformovaný obraz stereotaktického rámu. Výsledkem je jejich sloučený obraz.

7.4. Instalace

V první fázi musíme zkompilovat knihovny ITK a FLTK pomocí programu CMake [31]. Tento nástroj je multiplatformní a podporuje i celou řadu kompilátorů. Lze v něm vytvářet informace pro kompilátory různých OS. A zajišťuje tak přenositelnost kódu mezi platformami. V našem případě zejména mezi Windows (2000, XP, Vista) a Linux (UBuntu).

Použité verze knihoven:

- ITK 3.6.0 (nekompatibilní s GCC 4.3.1, lze použít ITK 3.10.1)
- VTK 5.0.4. (nekompatibilní s GCC 4.3.1, lze použít VTK 5.2.1)
- FLTK 1.3
- InsightApplications 3.6.0 (nekompatibilní s GCC 4.3.1, lze použít ITK 3.10.0)

Doporučené nastavení je **ne**kompilovat dané knihovny s příklady ni s testy. Naopak je lepší nechat vytvořit některé podpůrné knihovny (zlib, jpeg, …). Knihovna VTK požaduje při nastavování v CMake nainstalované OpenGL, stejně tak pro podpůrný plug-in z InsightApplications (ITKFltkImageViewer) je nutné mít nainstalovanou podporu OpenGL.

Při nastavování CMake pro InsightApplications je důležité zatrhnout, že chceme vytvořit soubory i pro knihovny FLTK a VTK.

Jakmile jsou knihovny připraveny pomocí CMake, lze je zkompilovat příslušným kompilátorem jazyka C++.

Potom lze v CMake vytvořit projekty pro jednotlivé programy této práce. To spočívá především v zadání cest ke knihovnám ITK, FLTK, ITK a cesty k adresáři InsightApplications se zdrojovými kódy a adresáři se zkompilovanými soubory (především ke knihovně ITKFltkImageViewer).

Poté lze programy zkompilovat kompilátorem uvedeným do konfigurace CMake.

.

8. Závěr

8.1. Mazání stereotaktického rámu

V první fázi jsme se zaměřili na segmentační úlohu, která měla vyčlenit z obrazu stereotaktický rám. Tento úkol je řešen soustavou filtrů poskytovaných v knihovně ITK. Navržený algoritmus je implementován jako další filtr ITK.

Navržené řešení nemusí být úspěšné, pokud obraz rámu není výrazný nebo pokud je nasazen tak, že se nachází v přílišné blízkosti hlavy pacienta.

Problém nevýrazného rámu v obraze může být způsoben vysokým šumem v obrazu nebo některými fyzikálními jevy při snímání obrazu pomocí MRI. Lze jej řešit kvalitním předzpracováním obrazu, např. vhodná volba filtrace obrazu. Problém defektu při snímání je řešitelný nastavením nižší meze pro prahování. Tuto mez by nemusel nastavovat již algoritmus, ale může být nastavena obsluhou.

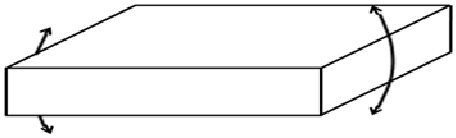
Druhý problém, nasazení rámu v blízkosti hlavy, lze vyřešit opět zásahem uživatele a to tím, že nastaví jinou velikost masky, která rozšiřuje označenou oblast hlavy. Zmenšením této masky se úměrně zmenší i ‚bezpečná' oblast, kde se nic nemaže.

8.2. Registrace

Předzpracování a registrace 3D obrazů není snadná úloha. Přináší problém vhodné volby mezi rychlostí a přesností algoritmu. Tato úloha se dá urychlit právě použitím multi-resolution.

Multi-resolution zkrátí výrazně čas registrace [19], i když na nevzorkování obrazů potřebuje čas. Nezlepšuje jen časové hledisko, ale i spolehlivost algoritmu. Na nižších úrovních algoritmus prohledává rychleji daleko větší prostor a je schopen najít i vzdálenější transformace od své počáteční.

Problémem je registrace malých, plochých obrazů. Kdy jim chybí ‚ukotvení' v jednom z rozměru. A mají tendenci se ‚rozhoupat' jako je to ukázáno na obrázku vpravo (Obrázek 47). To je způsobeno nejen malým počtem řezů, ale navíc je často v tomto směru největší rozměr voxelu. Oba tyto faktory způsobí nepřesnost.



Obrázek 47: Problém plochých obrazů

Nejjednodušším řešením tohoto problému je omezení pomocí nastavení významu parametrů transformace. Snížit význam jednak translace v daném směru a jednak omezit význam parametrů obsahujících rotaci v dané dimenzi.

Navržený algoritmus postačí k registraci obrazů, které nejsou extrémně vzdálené a pokud mají po registraci a transformaci jednoho obrazu ve druhý větší část společnou.

Pro zlepšení algoritmu lze ho plně ozkoušet pro použití vzájemné informace. Potom by byl schopen registrovat i obrazy různých modalit.

9. Použitá literatura

[1] **NEMA.** NEMA. *DICOM website.* [Online] http://medical.nema.org.

[2] **Mayo.** The Analyze data format. *Imaging Wiki.* [Online] http://imaging.mrc-cbu.cam.ac.uk/imaging/FormatAnalyze.

[3] **Nemocnice Na Homolce.** Nemocnice Na Homolce. *Nemocnice Na Homolce.* [Online] http://www.homolka.cz/cz/.

[4] **Siemens.** Siemens. *Magnetic Resonance.* [Online] http://w1.siemens.com/entry/cc/en/.

[5] **General Electric.** GE Healthcare. *General Electric.* [Online] http://www.gehealthcare.com/worldwide.html.

[6] **Philips.** Philips Healthcare. *Philips.* [Online] http://www.medical.philips.com/main/.

[7] **Schmitt, Michal.** Leksell Gamma Knife. *3.Pól.* [Online] 2008. http://www.tretipol.cz/index.asp?clanek&view&164.

[8] **Hornak, Joseph P., Ph.D.** The Basics of MRI. [Online] http://www.cis.rit.edu/htbooks/mri/.

[9] Magnetická rezonance. *Wikipedie, otevřená encyklopedie.* [Online] 2008 http://cs.wikipedia.org/wiki/Magnetická_rezonance.

[10] **Scott, David W.** On optimal and data-based histograms. *Biometrika.* 1979, Sv. 66, stránky 605-610.

[11] **Otsu, N.** A threshold selection method from gray-level histograms. *IEEE Trans. Sys., Man., Cyber.* 1979, Sv. 9, stránky 62–66.

[12] **Interpolation revisited, medical images application.** Thevenaz, P.; Blu, T.; Unser, M. *Medical Imaging, IEEE Transactions, Volume 19, Issue 7.* Červenec 2000, stránky 739 - 758.

[13] **Blu, T., Thevenaz, P. a Unser, M.** Linear interpolation revitalized. *Image Processing, IEEE Transactions.* 13, Květen 2004, Sv. 5, stránky 710 - 719.

[14] **van den Elsen, P.A., Pol, E.-J.D. a Viergever, M.A.** Medical image matching-a review with classification. *Engineering in Medicine and Biology Magazine, Volume 12, Issue 1, IEEE.* Březen 1993, stránky 26 - 39.

[15] **J. B. Antoine Maintz, Max A. Viergever.** A Survay of Medical Image Registration. *Medical Image Analysis.* 2, Březen 1998, Sv. 1, stránky 1-36.

[16] **Zitová, Barbara a Flusser, Jan.** Image registration methods: a survey. *Image and Vision Computing.* 2003, Sv. 21, stránky 977–1000.

[17] **Brown, L.G.** *A Survay of Image Registration Technique.* Department of Computer Science Columbia, New York University. New York, 1992. NY 10027.

[18] **Thévenaz, P., Ruttimann, U.E. a Unser, M.** A Pyramid Approach to Subpixel Registration Based on Intensity. *IEEE Transactions on Image Processing.* Leden 1998, Sv. 7, 1, stránky 27-41.

[19] **Maes, Frederik, Vandermeulen, Dirk a Suetens, Paul.** Comparative evaluation of multiresolution optimization strategies for multimodality image registration by maximization of mutual information. *Medical Image Analysis.* Prosinec 1999, stránky 373-386.

[20] **Kostelec, P. J., Weaver, J. B. a Healy, D. M.** Multiresolution elastic image registration. *Medical Physics.* 1998, Sv. 25, stránky 1593-1604.

[21] **Hlaváč Václav, Kybic Jan.** *Geometrické tranformace.* katedra kybernetiky, Centrum strojového vnímání, Fakulta elektrotechnická CVUT. Praha. přednáška.

[22] **R. H. Byrd, P. Lu and J. Noceda.** A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific and Statistical Computing.* 1995, Sv. 5, 16, stránky 1190-1208.

[23] **C. Zhu, R. H. Byrd and J. Nocedal.** L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software.* 1997, Sv. 23, 4, stránky 550 - 560.

[24] **ITK.** ITK. *Insight Segmentation and Registration Toolkit.* [Online] http://itk.org/.

[25] **VTK.** The Visualization ToolKit. *VTK.* [Online] http://vtk.org/.

[26] **FLTK.** FLTK. *Fast Light Toolkit.* [Online] http://fltk.org/.

[27] **CREATIS-LRMN (Centre de Recherche et d'Applications en Traitement de l'Image et du Signal).** Gdcm Home Page. *CREATIS-LRMN.* [Online] http://www.creatis.insa-lyon.fr/Public/Gdcm/.

[28] **Luis Ibáes, Will Schroeder, Josh Cates and the Insight Software Consortium.** *The ITK Software Guide Second Edition Updated for ITK version 2.4.* 2005.

[29] **Kitware Inc.** *ITK Registration Methods.* [prezentace] : ITK.

[30] **Bill Spitzak, F. Costantini, D. Gibson, M. Melcher.** *FLTK 1.3.0 Programming Manual.* Leden 2009.

[31] **Kitware, Inc.** CMake. *CMake.* [Online] http://www.cmake.org/.

[32] **Prata, Stephen.** *Mistrovství v C++, 3. aktualizované vydání.* Brno : Computer Press, 2007. 978-80-251-1749-1.

[33] **Sutter, Herb a Alexanderescu, Andrei.** *C++ 101 programovacích technik.* Brno : Zoner Press, 2005. 80-86815-28-5.

10. Přílohy

10.1. Příloha A: Obsah CD

Umístění	Popis
01_Data	Data pacientů, obrazy vyšetření
02_Lib	Knihovny programů
fltk-1.3.x-r6131.tar.gz	Knihovna FLTK pro tvorbu GUI
InsightApplications-3.6.0.zip	Aplikace ITK
InsightToolkit-3.6.0.zip	Knihovna ITK pro zpracování obrazů
vtk5.0.4.zip	Knihovna VTK pro zobrazování
03_Prog	Vlastní programy
01_Frame	Program pro odstranění stereotaktického rámu
02_Trans	Program pro transformaci obrazu
03_Reg	Program pro registraci obrazu
04_GUI	Jednoduché GUI, zobrazuje DICOM soubory
05_Test	Program pro testování registrace
04_Exper	Statistika experimentů
05_Text	Text diplomové práce

10.2. Příloha B: Shrnutí výsledků mazání stereotaktického rámu z obrazu

Číslo	Délka běhu algoritmu [s]	Velikost souboru[MB]	Rozměry[px]
1	00:25,0	3,146942	256x256x24
2	01:19,0	10,486976	256x256x80
3	00:27,0	3,146942	256x256x24
4	01:14,0	10,486974	256x256x80
5	01:08,0	10,486976	256x256x80
6	01:13,0	10,486976	256x256x80
7	01:03,0	10,486976	256x256x80
8	00:20,0	2,622652	256x256x20
9	01:12,0	10,486972	512x512x20
10	01:09,0	10,486972	512x512x20
11	00:26,0	4,719806	256x256x36
12	00:56,0	8,651966	256x256x66
13	01:09,0	12,584126	512x512x24
14	00:54,0	10,486974	512x512x20
15	01:07,0	12,584126	512x512x24
16	01:02,0	12,584124	512x512x24
17	01:04,0	10,486976	256x256x80
18	01:01,0	10,486974	256x256x80
19	01:07,0	10,486976	256x256x80
20	00:58,0	10,486974	256x256x80

10.3. Příloha C: Tabulka vstupních dat pro experimenty s mazáním stereotaktického rámu

Číslo	Cesta k datům
1	.\FinalCD\01_Data\01_Pac\Study__20060921__071312.656000\Series__20060921_ _072440.218000\1.3.12.2.1107.5.2.6.22428.30000006091915073131200010872\
2	.\FinalCD\01_Data\01_Pac\Study__20060921__071312.656000\Series__20060921_ _073655.390000\1.3.12.2.1107.5.2.6.22428.30000006091915073131200010898\
3	.\FinalCD\01_Data\01_Pac\Study__20060921__071312.656000\Series__20060921_ _075136.234000\1.3.12.2.1107.5.2.6.22428.30000006091915073131200010980\
4	.\FinalCD\01_Data\01_Pac\Study__20070423__071954.468000\Series__20070423_ _073028.375000\1.3.12.2.1107.5.2.6.22428.30000007041914312115600019940\
5	.\FinalCD\01_Data\02_Pac\Study__20060901__072227.906000\Series__20060901_ _072755.187000\1.3.12.2.1107.5.2.6.22428.30000006083114333151500003305\
6	.\FinalCD\01_Data\02_Pac\Study__20061113__074215.156000\Series__20061113_ _074700.562000\1.3.12.2.1107.5.2.6.22428.30000006111315152739000000138\
7	.\FinalCD\01_Data\02_Pac\Study__20061113__074215.156000\Series__20061113_ _080618.828000\1.3.12.2.1107.5.2.6.22428.30000006111315152739000000225\
8	.\FinalCD\01_Data\02_Pac\Study__20061113__074215.156000\Series__20061113_ _082430.250000\1.3.12.2.1107.5.2.6.22428.30000006111315152739000000307\
9	.\FinalCD\01_Data\03_Pac\Study__20070405__073921.221000\Series__20070405_ _074742.703000\1.3.12.2.1107.5.2.2.9077.20070405074742000004\
10	.\FinalCD\01_Data\03_Pac\Study__20070405__073921.221000\Series__20070405_ _075151.893000\1.3.12.2.1107.5.2.2.9077.20070405075151000005\
11	.\FinalCD\01_Data\04_Pac\Study__20070315__080345.953000\Series__20070315_ _080949.828000\1.3.12.2.1107.5.2.6.22428.30000007031118471809300013495\
12	.\FinalCD\01_Data\05_Pac\Study__20060623__072812.265000\Series__20060623_ _073334.296000\1.3.12.2.1107.5.2.6.22428.30000006062014304034300010209\
13	.\FinalCD\01_Data\06_Pac\Study__20060619__080818.351000\Series__20060619_ _081052.169000\1.3.12.2.1107.5.2.2.9077.20060619081052000002\
14	.\FinalCD\01_Data\06_Pac\Study__20070416__080142.730000\Series__20070416_ _080350.142000\1.3.12.2.1107.5.2.2.9077.20070416080350000002\
15	.\FinalCD\01_Data\07_Pac\Study__20060214__085301.563000\Series__20060214_ _085433.680000\1.3.12.2.1107.5.2.2.9077.20060214085433000002\
16	.\FinalCD\01_Data\07_Pac\Study__20060214__085301.563000\Series__20060214_ _090824.642000\1.3.12.2.1107.5.2.2.9077.20060214090824000005\
17	.\FinalCD\01_Data\07_Pac\Study__20060802__074508.312000\Series__20060802_ _075203.015000\1.3.12.2.1107.5.2.6.22428.30000006080120230859300000935\
18	.\FinalCD\01_Data\08_Pac\Study__20070510__074919.843000\Series__20070510_ _080651.328000\1.3.12.2.1107.5.2.6.22428.30000007050916305425000003465\
19	.\FinalCD\01_Data\08_Pac\Study__20070510__082110.015000\Series__20070510_ _083033.453000\1.3.12.2.1107.5.2.6.22428.30000007050916305425000003550\
20	.\FinalCD\01_Data\09_Pac\Study__20060406__083029.390000\Series__20060406_ _083540.062000\1.3.12.2.1107.5.2.6.22428.30000006040615310334300000004\