VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

DEPARTMENT OF BIOMEDICAL ENGINEERING

REGISTRACE OBRAZŮ S VYUŽITÍM OPTIMALIZACE POMOCÍ AUTOMATICKÉ DIFERENCIACE

IMAGE REGISTRATION USING OPTIMIZATION WITH AUTOMATIC DIFFERENTIATION

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. David Slavíček

AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Vičar, Ph.D.

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Biomedicínské inženýrství a bioinformatika**

Ústav biomedicínského inženýrství

Student: Bc. David Slavíček ***ID:*** 211212

Ročník: 2 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Registrace obrazů s využitím optimalizace pomocí automatické diferenciace

POKYNY PRO VYPRACOVÁNÍ:

1) Prostudujte metody pro automatickou registraci obrazových dat se zaměřením na optimalizaci transformačních parametrů pomocí metod využívajících gradient. 2) Proveďte rešerši existujících knihoven pro registraci obrazů pomocí metod využívajících gradient, které využívají automatickou diferenciaci knihoven pro hluboké učení (PyTorch, Tensorflow, atd.). 3) Navrhněte a implementujte vlastní jednoduchou pro afinní registraci s optimalizací využívající automatické diferenciace a jeho funkčnost ověřte na několika synteticky deformovaných obrázcích. 4) Vytvořte vhodné databáze pro testovaní úspěšnosti registrace. 5) Zdokonalte a rozšiřte vytvořenou metodu pro registraci a jednotlivý modifikace vyhodnoťte na vytvořených databázích. 6) Výsledky vhodně diskutujte a srovnejte s dostupnými metodami registrace.

DOPORUČENÁ LITERATURA:

[1] ZITOVÁ, Barbara a Jan FLUSSER. Image registration methods: a survey. Image and Vision Computing. 2003, 21(11), 977-1000. ISSN 02628856.

[2] MATTES, David, David R. HAYNOR, Hubert VESSELLE, Thomas K. LEWELLYN, William EUBANK, Milan SONKA a Kenneth M. HANSON. Nonrigid multimodality image registration. 2001-7-3, 1609-1620.

[3] NAN, Abhishek, Matthew TENNANT, Uriel RUBIN a Nilanjan RAY. DRMIME: Differentiable Mutual Information and Matrix Exponential for Multi-Resolution Image Registration. In: Proceedings of Machine Learning Research. PMLR, 2020, s. 527-543. ISSN 2640-3498.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 22.5.2023

Vedoucí práce: Ing. Tomáš Vičar, Ph.D.

prof. Ing. Valentine Provazník, Ph.D.
předseda rady studijního programu

ABSTRAKT

Práce se zabývá využitím automatické diferenciace v registraci obrazů. V teoretické části byly popsány kroky registrace obrazů a možné způsoby jejich realizace. V praktické části byl navržen algoritmus pro registraci afinně transformovaných obrazů s využitím automatické diferenciace. Algoritmus byl implementován. Byly navrhnuty dvě modifikace algoritmu. Dále byla sestavena testovací databáze sestávající z RTG snímků hrudníku. Na ní byl algoritmus otestován.

KLÍČOVÁ SLOVA

registrace obrazů, automatická diferenciace, optimalizace, zpracování obrazu

ABSTRACT

This thesis deals with image registration using optimization with automatic differentiation. In the teoretical part were decribed steps of image registration and possibilities of their realization. The practical part consists of design of algorithm for registration of afinely transformed images and implementation of it. Two modifications of the algorith were proposed. Database of xray images was designed. The algorithm was tested on this database.

KEYWORDS

image registration, automatic differentiation, optimalization, image processing

SLAVÍČEK, David. *Registrace obrazů s využitím optimalizace pomocí automatické diferenciace*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav biomedicínského inženýrství, 2022, 40 s. Diplomová práce. Vedoucí práce: Ing. Tomáš Vičar, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. David Slavíček
VUT ID autora:	211212
Typ práce:	Diplomová práce
Akademický rok:	2022/23
Téma závěrečné práce:	Registrace obrazů s využitím optimalizace pomocí automatické diferenciace

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Tomáši Vičarovi Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Registrace obrazů nachází uplatnění v mnoha aplikacích. V medicíně umožňuje lékařům lépe porovnat snímky jednoho pacienta v čase, nebo vytvořit zfúzovaný obraz z několika zobrazovacích modalit. [1, 2]

Cílem této práce je sestavit algoritmus pro registraci obrazů s využitím automatické diferenciace, navrhnout modifikace, algoritmus otestovat a porovnat s dostupnými algoritm. Od programu je vyžadováno, aby byl schopen registrovat různé medicínské obrazy s dobrou přesností a nízkým výpočetním časem.

1 Registrace obrazu

Registrace obrazů je úloha při které jsou zadány dva nebo více obrazů a je třeba jeden nebo několik z nich transformovat tak, aby pixely výsledných obrazů odpovídaly stejným, nebo co nejbližším místům v prostoru. [3] Mattes a kol. definují pro účely registrace medicínských dat v definici místo prostoru "anatomický prostor". [2] Registraci obrazů můžeme rozdělit na dva základní druhy - monomodální a multimodální. Monomodální registrace zarovnává obrazy ze stejné modality (například dva rentgenové snímky), multimodální obrazy z různých modalit (například CT a MRI data).[4]

Obvykle jeden z obrazů považujeme za nedeformovaný a tedy jej netransformujeme. Označme nedeformovaný obraz B a obraz, který k němu chceme registrovat A. Dále označme X_A zorné pole obrazu A, tedy soubor pozičních vektorů $\mathbf{x}_A$ a X_B zorné pole obrazu B, tedy soubor pozičních vektorů $\mathbf{x}_B$. Poziční vektory jsou zobrazeny na odpovídající hodnoty intenzity $A(\mathbf{x}_A)$, $B(\mathbf{x}_B)$.

$$X_A = \{\mathbf{x}_A\}, X_B = \{\mathbf{x}_B\}; X_A \rightarrow \{A(\mathbf{x}_A)\}, X_B \rightarrow \{B(\mathbf{x}_B)\} \tag{1.1}$$

Při registraci využíváme zvolenou transformaci T_α, kde α je vektor parametrů transformace. Transformací transformujeme obraz A na A', který pak prostorově odpovídá obrazu B.

$$T_\alpha : \mathbf{x}_{A'} = T_\alpha(\mathbf{x}_A), A'(\mathbf{x}_{A'}) = A(\mathbf{x}_A), \mathbf{x}_{A'} = \mathbf{x}_B \tag{1.2}$$

Protože si zorná pole neodpovídají, je porovnání obrazů možné dělat jen v průniku zorných polí $X_{A'}$ a X_B, který označíme Ω_α. Průnik Ω_α závisí na vektoru transformačních parametrů α.

$$\Omega_\alpha = X_{A'} \cap X_B, X_{A'} = \{T_\alpha(\mathbf{x}_A)\}, X_B = \{\mathbf{x}_B\} \tag{1.3}$$

Cílem registrace je nalézt vektor parametrů α, tak aby byla optimalizována kriteriální funkce $c(B(\mathbf{x}_B), A'(T_\alpha(\mathbf{x}_A)))$.

$$\alpha_0 = arg\,max_\alpha c(B(\mathbf{x}_B), \mathbf{x}_B, T_\alpha(\mathbf{x}_A) \in \Omega_\alpha \tag{1.4}$$

[3]

Před provedením registrace může být vhodné data předzpracovat, zejména odstraněním šumu. Poté je třeba rozhodnout které transformaci odpovídá deformace obrazu. Dále je třeba stanovit vhodnou kriteriální funkci, pomocí které je průběžně odhadováno, jak kvalitně jsou obrazy zarovnány. Při použití metod založených na extrakci charakteristických rysů je třeba zvolit, které rysy budou extrahovány a vytvořit postup jejich extrakce. Následuje volba optimalizačního algoritmu. Poté optimalizační algoritmus iterativně hledá nejvhodnější zarovnání. [4]

1.1 Kriteriální funkce

Kriteriální funkce slouží ke kvantifikování kvality zarovnání obrazů. Při registraci lze využít gradientu kriteriální funkce k hledání optimálního zarovnání.

1.1.1 Střední kvadratická chyba (MSE)

Jednoduchá a často využívaná metrika[1].

$$MSE = \sum_{i}(x_i - y_i)^2$$

x_i ve vzorci značí pixel na i-té pozici v zarovnávaném obraze, y_i značí pixel na i-té pozici v referenčním obraze

1.1.2 Střední absolutní chyba (MAE)

Jednoduchá metrika, méně využívaná, než *střední kvadratická chyba* (MSE).

$$MAE = \sum_{i}|x_i - y_i|$$

x_i ve vzorci značí pixel na i-té pozici v zarovnávaném obraze, y_i značí pixel na i-té pozici v referenčním obraze

1.1.3 Vzájemná informace

Tato metoda je vhodná pro multimodální registraci obrazů. Vzájemná informace je původně definována jako míra závislosti dvou náhodných proměnných. [2, 4, 5] Můžeme ji vyjádřit pomocí vzorce:

$$MI = H_A + H_B - H_AB$$

Kde H_A, H_B a H_AB jsou dány vztahy:

$$H_A = -\sum_{k=0}^{q-1} P(k)logP(k)$$

$$H_B = -\sum_{k=0}^{r-1} P(k)logP(k)$$

$$H_A = -\sum_{k=0}^{q-1}\sum_{l=0}^{r-1} P(k,l)logP(k,l)$$

kde q-1 a r-1 jsou maximální intenzity v jednotlivých obrazech a P(x) pravděpodobnost výskytu dané intenzity v obraze. [3]

1.1.4 Normalizovaná křížová korelace

1.2 Metody založené na ploše a metody založené na charakteristických rysech

Při registraci obrazu můžeme využít metody založené na ploše, nebo metody založené na extrakci charakteristických rysů. Metody založené na extrakci charakteristických rysů poskytují výhodu při registraci multimodálních obrazů. [1]

1.3 Transformace

1.3.1 Afinní transformace

Afinní transformace sestávají z posunutí (translace), otočení, zvětšení nebo zmenšení v jednotlivých osách a zkosení. Afinní transformaci lze vyjádřit transformační maticí. Pokud jsou známy souřadnice bodu před transformací a je třeba znát souřadnice bodu po transformaci, lze využít následující postup. Souřadnice bodu před transformací jsou zapsány jako sloupcový vektor. K vektoru je zdola přidán jeden řádek obsahující hodnotu jedna. Vektor je vynásoben transformační maticí. Výsledkem jsou souřadnice bodu po afinní transformaci.

1.3.2 Transformace s využitím kontrolních bodů

Další možností transformace je určit kontrolní body, popsat na kterou pozici se transformuje každý kontrolní bod a definovat pravidlo pro transformaci ostatních bodů v závislosti na transformaci kontrolních bodů v jejich okolí. Mattes a spol využívají tohoto principu a body, které nejsou kontrolními dopočítávají pomocí kubických B-splinů. Tento postup je vhodný pokud není deformace rovnoměrná po celém obraze. Například u tomografických snímků kde hraje roli pohyb bránice při dýchání. [2]

1.4 Optimalizace

K optimalizaci lze využít gradientních metod nebo evolučních algoritmů. V práci je využit gradientní sestup, implementovaný pomocí autommatické diferenciace.

1.5 Algoritmy pro registraci obrazů

1.5.1 DRMIME

Algoritmus DRMIME slouží k registraci monomodálních i multimodálních dat. Využívá metriky mutual information(MI) a je plně diferencovatelný. Metriku MI počítá metodou MINE, založenou na neuronových sítích. [4]

Využívá pyramidového přístupu, který je však upraven, tak aby nepracoval od nejhrubšího rozlišení k nejjemnějšímu, ale se všemi úrovni pyramidy současně. Kvůli drobnému posunu struktur v obraze při rozmazání a podvzorkování, používá algoritmus mírně upravené transformační matice pro transformaci nejjemnější rozlišení v obrazové pyramidě. [4]

K ušetření místa v paměti grafické karty algoritmus vybere v každé iteraci jen část pixelů, které se účastní výpočtu kriteriální funkce. Autoři algoritmus implementovali ve dvou verzích, kdy jedna vybírá k výpočtu náhodně 10% pixelů, druhá najde hrany Cannyho detektorem a používá pouze pixely v okolí hran. [4]

Algoritmus byl testován na databázi FIRE[6] obsahující monomodální data a na databázi ANHIR[7] obsahující multimodální data. [4]

1.5.2 Nerigidní multimodální registrace obrazů - Mattes a kolektiv

Algoritmus slouží k registraci 3D snímků hrudníku z PET a CT. Motivací pro registraci těchto snímků je vysoké rozlišení obrazů z CT a fyziologický význam dat z PET. [2]

Při PET vyšetření je těsně před snímáním radiofarmaka pořízen prozařovací snímek. Získaný snímek je velmi podobný CT, ale je získán zářením o vyšší intenzitě, což způsobuje horší kontrast tkání a má oproti CT snímkům horší rozlišení. Algoritmus tedy neregistruje CT snímky k obrazu získanému vyzařováním radiofarmaka, ale k prozařovacím snímkům, získaným těsně před snímáním radiofarmaka. [2]

Algoritmus využívá kontrolních bodů umístěných v pravidelné mřížce a interpolaci B-spliny. Kriteriální funkcí je vzájemná informace v upravené spojité podobě. K upravení vzájemné informace do spojité podoby jsou histogramy, ze kterých je vzájemná informace počítána vytvořeny s použitím Parzenova okna. Derivace kriteriální funkce jsou vyjádřeny analyticky. Optimalizace probíhá na základě gradientu. [2]

K urychlení výpočtu a snížení pravděpodobnosti uvíznutí v lokálním minimu je využita registrace v několika fázích, kde se mění rozmazání vstupních obrazů Gaussovým konvolučním jádrem, počet kontrolních bodů a další parametry. Jedná se o postup podobný pyramidovému přístupu. [2]

1.5.3 DIRNet

Algoritmus DIRNet využívá k registraci neuronové sítě. Tvoří jej tři hlavní části, kterými jsou neuronová síť, prostorový transformátor a resampler. Neuronová síť je učena bez učitele, po naučení se je schopna registrovat obrazy na jeden průchod. [8]

Referenční a pohyblivý obraz nejprve vstupují do konvoluční neuronové sítě. Neuronová síť analyzuje výřezy z referenčního a pohyblivého obrazu a vypočítá lokální deformační parametry pro prostorový transformátor. Prostorový transformátor vygeneruje pole vektorů posunu. To vstupuje do resampleru, který pak transformuje pohyblivý obraz. Ve fázi učení je dále vypočítána metrika podobnosti a jsou upraveny váhy neuronové sítě. [8]

Algoritmus byl implementován s použitím knihoven Theano a Lasagne [9]. Testován byl úspěšně na datasetu ručně psaných číslic MNIST a na MRI snímcích srdce. [8]

1.6 Software pro registraci obrazů

1.6.1 Elastix

Elastix je software pro rigidní a nonrigidní transformaci obrazů, zejména medicínských. Využívá knihovnu Insight Toolkit (ITK)[10]. Elastix umožňuje uživateli vytvořit registrační algoritmus pomocí parametrového souboru. Z příkazové řádky pak uživatel program spustí s parametrovým souborem a danými obrazy a ten provede registraci. [11]

Elastix podporuje různé transformace, například rigidní, afinní nebo transformaci s využitím kontrolních bodů. Nabízí mnoho kriteriálních funkcí jako například MSE, křížovou korelaci nebo vzájemnou informaci. Program podporuje využití pyramidového přístupu a to ve dvou formách - rozmazání Gaussovým konvolučním jádrem s podvzorkováním a bez podvzorkování. K urychlení výpočtu umožňuje program využití pouze některých pixelů k výpočtu kriteriální funkce. Pixely mohou být zvoleny například v okolí hran v referenčním obraze, nebo náhodně. [11]

1.6.2 Simple Elastix

Simple Elastix je toolkit pro Python, Javu, C# a další jazyky, obsahující knihovnu Elastix. Navíc nabízí předkonfigurované metody registrace, které mohou sloužit jako výchozí bod při návrhu registračního algoritmu. [12]

1.6.3 Kornia

Kornia je diferencovatelná knihovna pro programovací jazyk Python umožňující snadnou integraci metod klasického strojového učení do modelů pro hluboké učení. Sestává z několika modulů, které slouží například pro práci s barvami, kriteriální funkce nebo extrakci znaků. [13]

Kornia je silně inspirována knihovnou OpenCV. Hlavními výhodami oproti OpenCV je diferencovatelnost a kompatibilita s PyTorch. Nevýhodou je nižší optimalizovanost pro použití v produkci a embedded zařízeních. Záměrem autorů tedy je, aby Kornia sloužila k trénování neuronových sítí, které pak budou nasazeny pomocí jiných knihoven například OpenCV. [13]

Knihovna Kornia je strukturována do několika modulů podle různých témat z oblasti počítačového vidění. Vlastnosti jednotlivých modulů jsou shrnuty níže. [13]

Modul *kornia.augumentation* obsahuje funkce k augumentaci dat pro trénování neuronových sítí. Podporuje práci s dávkami (batch). Příklady funkcionalit jsou: náhodné rotace, afinní a perspektivní transformace, různé náhodné transformace intenzit barev, přidání šumu a rozmazání pohybem. [13]

Modul *kornia.color* umožňuje konverze mezi barevnými prostory. Funkce modulu jsou podobné jako některé funkce knihoven OpenCV a Scikit-image, ale Kornia zavádí modifikace umožňující podporovat výpočty s plovoucí desetinnou čárkou. Umožňuje přechod například mezi následujícími barevnými prostory: šedotónový, RGB, RGBA, BGR, HSV. Dále umožňuje výpočet barevných histogramů, úpravu kontrastu, gamma korekci a jiné. [13]

Modul *kornia.features* obsahuje funkce pro detekci charakteristických rysů (features). Modul obsahuje například tyto funkce: Harrisův rohový detektor, Shi-Tomasiho rohový detektor, detekci charakteristických rysů metodou SIFT a detekci chrakteristických rysů deeplearningovými metodami HardNet a SOSNet. Funkce jsou implementovány diferencovatelně. [13]

Modul *kornia.filters* zahrnuje různé lineární i nelineární filtry. Obsahuje například funkce pro rozmazání pomocí Gaussiánu, nebo obdelníkovým oknem, mediánový filtr, nebo Sobelův hranový detektor. Všechny zmíněné funkce jsou diferencovatelné. [13]

Dalšími moduly jsou moduly *kornia.geometry*, *kornia.losses* a *kornia.contrib*. První zmíněný model umožňuje geometrické transformace včetně afinních transformací. Druhý zmíněný model nabízí různé kriteriální funkce. Poslední zmíněný model obsahuje různé experimentální operátory. [13]

1.6.4 AIRLab

AIRLab je knihovna pro programovací jazyk Python zaměřená na registraci medicínských obrazů. AIRLab využívá knihovny PyTorch a umožňuje výpočty na GPU. Pro práci se vstupy a výstupy využívá AIRLab knihovny SimpleITK[14]. Autoři uvádí záměr vytvořit knihovnu umožnující rychlé prototypování registračních algoritmů. [15]

Knihovna klade poměrně velký důraz na regularizaci algoritmu pro registraci. Nabízí čtyři různé regularizátory - difůzní regularizátor, regularizátor využívající celkové variace anizotropní i izotropní a sparsity regularizátor, penalizující nenulové koeficienty. [15]

Poslední příspěvek do této knihovny na GitHubu[16] je z 8. října 2020, připomínky (issues) na GitHubu zůstávají bez reakce. Z toho lze usuzovat, že knihovna již pravděpodobně není dále rozšiřována a vyvíjena.

2 Navržený algoritmus

2.1 Popis algoritmu

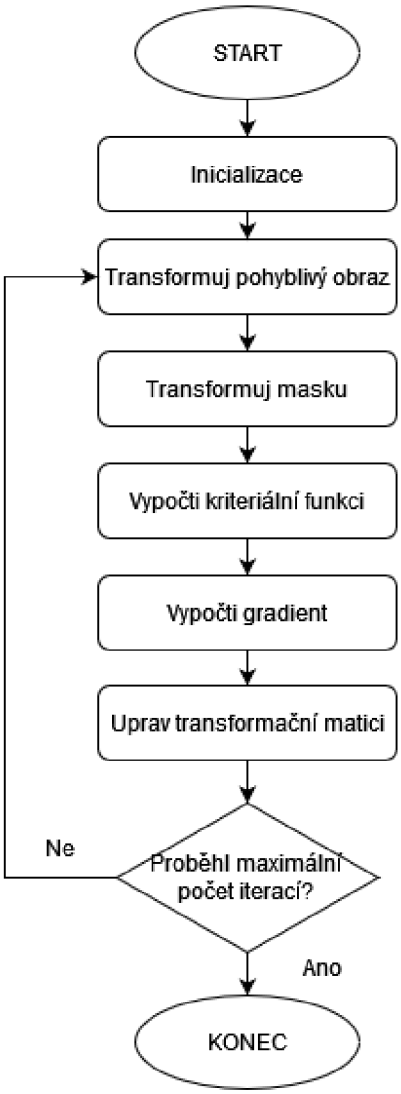
Algoritmus iterativně transformuje pohyblivý obraz a masku pomocí aktuální transformační matice. Poté vypočte z referenčního obrazu, pohyblivého obrazu a masky hodnotu kriteriální funkce. Následně vypočte gradient kriteriální funkce pomocí automatické diferenciace. Dále upraví podle vypočteného gradientu transformační parametry a vypočte novou transformační matici. Pokud proběhl požadovaný počet iterací algoritmus končí, pokud ne proběhne další iterace. Blokové schéma algoritmu je na obrázku 2.1.

2.2 Pyramidový přístup

Algoritmus byl vylepšen s využitím pyramidového přístupu. Před vykonáním algoritmu je třeba zvolit koeficienty, podle kterých budou obrazy podvzorkovány. Na začátku algoritmu jsou referenční obraz, pohyblivý obraz i maska podvzorkovány nejhrubším koeficientem. Poté proběhne předem daný počet iterací algoritmu s podvzorkovanými obrazy. Následně jsou obrazy a maska podvzorkovány jemnějším koeficientem a opět proběhne daný počet iterací algoritmu. Obdobě se opakuje podvzorkování všemi zvolenými koeficienty, nakonec proběhne algoritmus pro nepodvzorkované obrazy. Pyramidový přístup omezuje pravděpodobnost uvíznutí v lokálním minimu a snižuje potřebný výkon pro registraci. Schéma modifikace s využitím pyramidového přístupu je znázorněno na obrázku 2.2

2.3 Změna kroku učení pomocí scheduleru

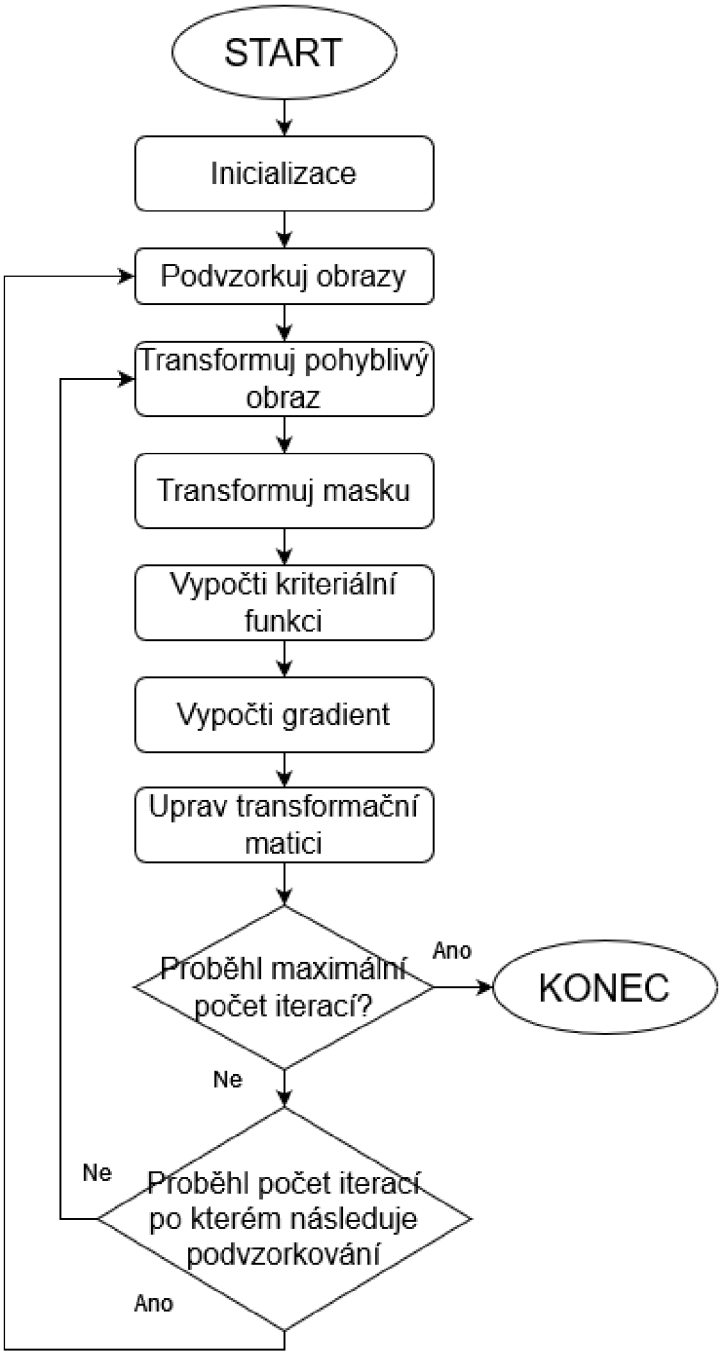
Na začátku registrace je výhodnější, když algoritmus modifikuje transformační parametry ve větších krocích. Je tím dosaženo rychlejšího přiblížení k optimálnímu řešení a omezení uvíznutí v lokálním minimu. V závěru registrace naopak bývá výhodnější postupovat po malých krocích, aby se algoritmus nevzdaloval od optima a nalezl jej s co nejvyšší přesností. Proto můžeme v průběhu registrace měnit krok učení. Scheduler použitý v této práci mění krok učení tak, že jej vždy po předem daném počtu iterací vynýsobí koeficientem gamma. Schedulerem lze rozšířit jak základní verzi algoritmu, tak jej zkombinovat s pyramidovým přístupem.



Obr. 2.1: Blokové schéma navrženého algoritmu v základní verzi

2.4 Implementace navrženého algoritmu

Navržený algoritmus využívá knihoven Scikit image[17], Numpy, [18] a PyTorch [19]. Nejprve jsou vytvořeny proměnné pro translaci ve směrech x a y, zvětšení ve směrech x a y, zkosení ve směrech x, y a rotaci. Zvětšení jsou inicializovány hodnotou jedna, ostatní parametry hodnotou nula. Dále je v každé iteraci ze zmíněných proměnných vytvořena transformační matice. Pomocí funkce z knihovny PyTorch je registrovaný obraz transformován. Následně je vypočten gradient a s využitím optimizeru Adam jsou vypočteny nové hodnoty posunů a rotace. Poté se cyklus opakuje až do dosažení požadovaného počtu iterací.

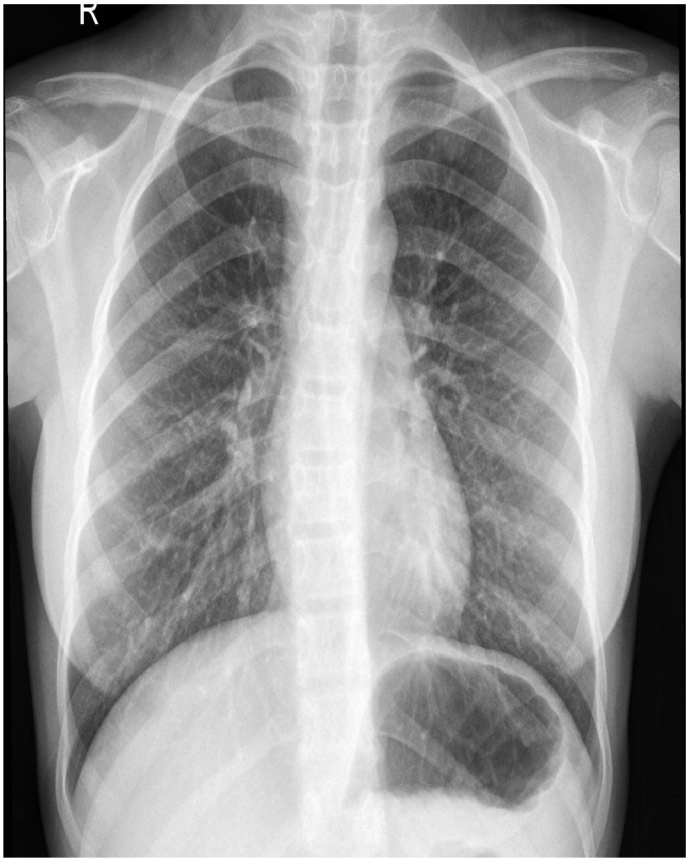


Obr. 2.2: Blokové schéma navrženého algoritmu v modifikaci s využitím pyramidového přístupu

3 Testovací databáze

3.1 Rentgenové snímky hrudníku

Testovací sada obsahuje 100 8bitových šedotónových rentgenových snímků hrudníku o výšce 1280 pixelů a šířce 1024 pixelů. Snímky byly získány z databáze[20] a upraveny, aby měly všechny stejný rozměr. Příklad vstupu z testovací databáze si můžete prohlédnout na obrázku 3.1. Snímky byly transformovány počítačem, aby byly vytvořeny páry obrazů k registraci. Díky tomu známe přesně hledané transformační matice. Hlavní testovací sada obsahovala afinně transformované obrazy s posuny v x a y v rozsahu +-5% šířky a výšky obrazu, otočením +-12°, zvětšením 90% až 110% a koeficienty zkosení v rozmezí -0,1 až 0,1. Transformační parametry byly generovány rovnoměrným rozložením. Dále byly vytvořeny databáze pro dílčí transformace (například pouze posun v X).



Obr. 3.1: Příklad obrázku z testovací databáze

4 Testování hyperparametrů

4.1 Metrika testování

Testování proběhlo s využitím metriky Target registration error. Tato metrika využívá definovaných bodů v referenčním a posuvném obraze a chybu registrace pak stanovuje jako průměrnou eukleidovskou vzdálenost, mezi místem na které měl být daný bod být registrován a místem na které registrován byl.

Protože jsou data generována synteticky, není třeba detekovat významné body v referenčním a posuvném obraze. Jako významné body jsou určeny body na souřadnicích [0,5;0,5], [0,5;-0,5], [-0,5;0,5] a [-0,5;-0,5] v referenčním obraze. Využíváme zde souřadnic normalizovaných na rozsah <-1,1> v obou dimenzích.

Protože je známa jak deformační matice, kterou byl vytvořen pohyblivý obraz, tak matice získaná z registrace, lze pak body jednoduše transformovat postupně pomocí obou transformačních matic. Při dokonalé registraci by se body promítly zpět samy na sebe. Chyba je vypočtena jako vzdálenost průměrná eukleidovská vzdálenost původního bodu a bodu získaného jeho postupnou transformací pomocí obou matic.

4.2 Testování hyperparametrů základní verze algoritmu

V základní verzi algoritmu bylo potřeba otestovat dva hyperparametry, kterými jsou krok učení a počet iterací. Počet iterací je třeba volit dostatečně vysoký, aby se algoritmus stihl dostatečně přesně přiblížit správnému řešení a zároveň ne zbytečně velký, aby byla omezena doba běhu programu a spotřeba výpočetního výkonu. U kroku učení je třeba, aby byl dostatečně velký k tomu, aby se algoritmus blížil požadovanému řešení rychle a neuvízl v lokálním minimu. Příliš velký krok učení však omezuje přesnost s jakou jsme schopni dosáhnout výsledku. Tyto hyperparametry se navzájem ovlivňují. Optimální krok učení pro jeden počet iterací nemusí být optimální pro druhý a naopak.

4.2.1 Testování kroku učení v základní verzi algoritmu

Algoritmus byl otestován s kroky učení 0,001, 0,003, 0,005 a 0,007. Testování probíhalo pro 120 iterací. Výsledky testování jsou zaznamenány v tabulce 4.1. Jako optimální hodnota kroku učení bylo zvoleno 0,003.

Tab. 4.1: Testování různých kroků učení

krok učení	TRE	kriteriální funkce
0,001	0,0391	8,00E-4
0,003	0,0203	2,92E-5
0,005	0.0210	1,41E-5
0,007	0.0245	1,30E-5

4.2.2 Testování počtu iterací v základní verzi algoritmu

Vhodnou hodnotu počtu iterací lze odhadnout z grafu hodnoty kriteriální funkce po jednotlivých iteracích algoritmu. Pokud hodnota kriteriální funkce významně neklesá, pravděpodobně již není třeba dále zvyšovat počet iterací.

V tabulce 4.2 je zanesen testovaný počet iterací, chyba TRE, kriteriální funkce a doba běhu programu na základní testovací databáze se 100 obrazy. Na grafu A.1 je patrný průběh kriteriální funkce pro 300 iterací algoritmu.

Jako optimální hodnota počtu iterací bylo stanoveno 120. Při zvoleném počtu iterací je doba běhu algoritmu asi jedna minuta. Inspirací pro tuto volbu byl i článek [4], ve kterém nejrychlejší algoritmy dosahovaly na databázi s obdobným počtem obrazů času běhu rovněž přibližně jedné minuty.

Tab. 4.2: Testování optimálního počtu iterací

počet iterací	TRE	kriteriální funkce	doba běhu programu
50	0,0286	8,17E-4	26,94
100	0,0211	6,38E-5	52,40
120	0,0203	2,92E-5	62,84
140	0,0199	1,94E-5	72,66
160	0,0196	1,34E-5	83,08
180	0,0194	1,00E-5	93,35
200	0,0193	8,67E-6	103,60
300	0,0189	5,03E-6	155,08

4.3 Testování hyperparametrů algoritmu modifikovaného s využitím pyramidového přístupu

U modifikace s využitím pyramidového přístupu je mimo dříve testovaných hyperparametrů třeba otestovat i počet úrovní podvzorkovací pyramidy.

4.3.1 Testování počtu úrovní pyramidy

Podvzorkovací pyramida byla vždy konstruována tak, aby předchozí patro bylo podvzorkováno dvakrát více než předchozí a poslední patro nebylo podvzorkováno vůbec. Vhodný počet úrovní pyramidy byl testován pro 30 iterací na každé úrovni podvzorkování. Výsledky testování jsou zobrazeny v tabulce 4.3. Nejlepšího výsledku bylo dosaženo pro 6 úrovní pyramidy.

Tab. 4.3: Testování různých úrovní podvzorkování

počet úrovní pyramidy	TRE	kriteriální funkce	doba běhu programu
1	0,0378	3,02E-3	16,49
2	0,0183	4,81E-4	20,40
3	0,0089	9,67E-5	20,98
4	0,0062	3,84E-5	21,10
5	0,0048	1,93E-5	21,36
6	0,0040	1,25E-5	21,61
7	0,0051	1,33E-5	22,77

4.3.2 Testování počtu iterací

Při použití pyramidového přístupu je optimální počet iterací jiný, než když není použit. Proto bylo třeba optimální počet iterací hledat znovu. Počet iterací byl optimalizován pro 6 úrovní pyramidy a krok učení 0,003. Bylo zjištěno, že optimální počet iterací je 30. Výsledky testování jsou v tabulce 4.4. Pro vyšší počty iterací docházelo k poklesu kriteriální funkce, ale chyba TRE se naopak zvyšovala. Uvedený počet iterací probíhá v každé úrovni pyramidy, celkem tedy proběhne iterací šestinásobek, byť většina z nich je s výrazně podvzorkovanými obrazy.

4.3.3 Testování kroku učení

Protože použitím pyramidového přístupu došlo k výrazné změně chování algoritmu, bylo třeba ověřit, zda je stále vhodné využití korku učení 0,003.

Tab. 4.4: Testování různých počtů iterací

počet iterací	TRE	kriteriální funkce	doba běhu programu
10	0,0130	4,45E-3	07,32
20	0,0045	7,07E-4	14,09
30	0,0040	6,22E-5	21,21
40	0,0051	1,25E-5	28,30

Tab. 4.5: Testování různých kroků učení

krok učení	TRE	kriteriální funkce
0,001	0,0125	2,00E-4
0,001	0,0046	3,95E-5
0,003	0,0040	1,25E-5
0,005	0.0086	2,07E-5
0,008	0.0127	6,59E-5
0,01	0.0145	1,12E-4

4.4 Testování hyperparametrů algoritmu modifikovaného s využitím pyramidového přístupu a scheduleru

S použitím scheduleru je třeba otestovat parametr gamma, který udává jak se mění krok učení. V tabulce 4.6 jsou výsledky testování tohoto parametru. Testování probíhalo pro krok učení 0,005, se šesti úrovněmi pyramidy a maximálním počtem iterací 200 v každé úrovni pyramidy. Optimální hodnota parametru byla stanovena na 0,4. Doba běhu programu byla 204 sekund, nezávisle na velikosti parametru gamma.

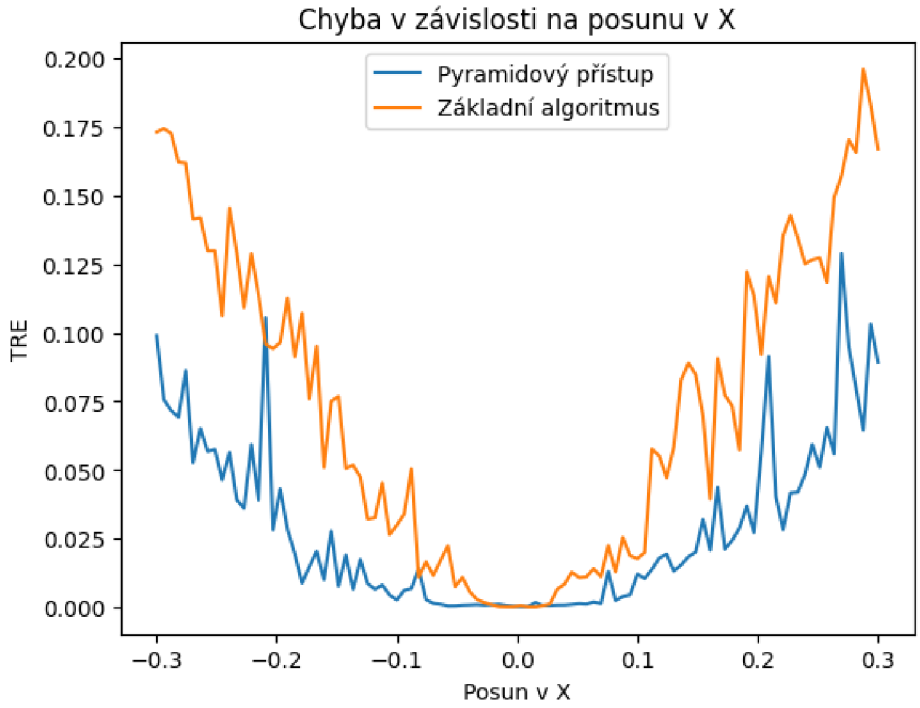
Tab. 4.6: Testování parametru gamma

gamma	TRE	kriteriální funkce
0,2	0,0085	1,05E-5
0,3	0,0041	6,84E-6
0,4	0,0038	3,95E-6
0,5	0,0040	1,25E-6

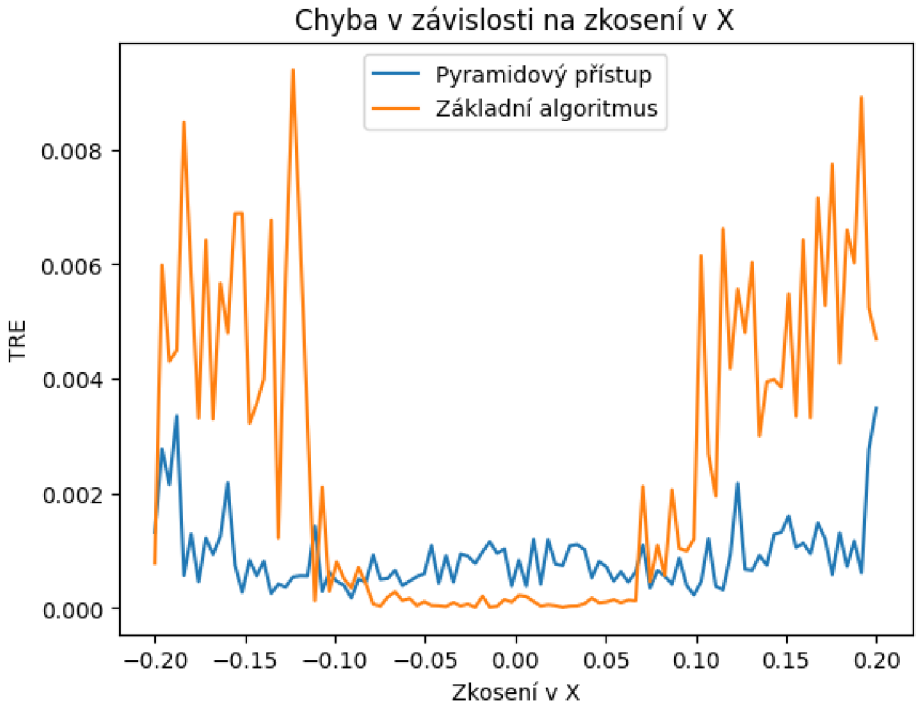
5 Testování algoritmů pro jednotlivé dílčí transformace

Algoritmus byl otestován na rentgenových snímcích upravených vždy jen jednou dílčí transformací. Testování umožnilo zjistit které typy deformací zvládá algoritmus hůře a které lépe a v jakém rozsahu deformací poskytuje dobrou přesnost registrace.

Testován byl v základní verzi a ve verzi s pyramidovým přístupem. Použití pyramidového přístupu se ukázalo jako výhodné u všech typů deformací s výjimkou zkosení. Ukázka testování pro posun v x a zkosení v x je v grafech 5.1 a A.6. Další grafy z testování jsou v přílohách A.2 až A.5.



Obr. 5.1: Testování posunu v X



Obr. 5.2: Testování zkosení

Závěr

V teoretické části diplomové práce byly popsány vybrané metody pro automatickou registraci obrazových dat. Dále byly popsány knihovny pro registraci obrazů, využívající automatickou diferenciaci pro výpočet gradientu.

V praktické části byl navržen algoritmus pro registraci obrazů s využitím automatické diferenciace. Navržený algoritmus byl naprogramován v programovacím jazyce Python, s využitím knihovny PyTorch [19]. Byly vytvořeny modifikace navrženého algoritmu s využitím pyramidového přístupu a s použitím scheduleru. Testováním bylo zjištěno, že použití pyramidového přístupu i scheduler mohou zlepšit přesnost navrženého algoritmu. U scheduleru bylo však zvýšení přesnosti jen velmi malé a to za cenu významného prodloužení doby běhu programu.

Literatura

[1] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003.

[2] David Mattes, David R Haynor, Hubert Vesselle, Thomas K Lewellyn, and William Eubank. Nonrigid multimodality image registration. In *Proceedings of SPIE*, volume 4322, pages 1609–1620. SPIE, 2001.

[3] Jiří Jan. *Medical image processing, reconstruction and analysis : cocepts and methods*. CRC Press, London, second edition edition, 2020.

[4] Abhishek Nan, Matthew Tennant, Uriel Rubin, and Nilanjan Ray. Drmime: Differentiable mutual information and matrix exponential for multi-resolution image registration. 2020.

[5] Jorge R. Vergara and Pablo A. Estévez. A review of feature selection methods based on mutual information. *Neural computing applications*, 24(1):175–186, 2014.

[6] Carlos Hernandez-Matas, Xenophon Zabulis, Areti Triantafyllou, Panagiota Anyfanti, Stella Douma, and Antonis A Argyros. Fire: Fundus image registration dataset. *Modeling and Artificial Intelligence in Ophthalmology*, 1(4):16–28, 2017.

[7] Jiří Borovec, Jan Kybic, Ignacio Arganda-Carreras, Dmitry Sorokin, Gloria Bueno, Alexander Khvostikov, Spyridon Bakas, Eric Chang, Stefan Heldmann, Kimmo Kartasalo, Leena Latonen, Johannes Lotz, Michelle Noga, Sarthak Pati, Kumaradevan Punithakumar, Pekka Ruusuvuori, Andrzej Skalski, Nazanin Tahmasebi, Masi Valkonen, and Arrate Muñoz-Barrutia. Anhir: Automatic non-rigid histological image registration challenge. *IEEE Transactions on Medical Imaging*, PP:1–1, 04 2020. doi:10.1109/TMI.2020.2986331.

[8] Bob D. de Vos, Floris F. Berendsen, Max A. Viergever, Marius Staring, and Ivana Išgum. End-to-end unsupervised deformable image registration with a convolutional neural network. In M. Jorge Cardoso, Tal Arbel, Gustavo Carneiro, Tanveer Syeda-Mahmood, João Manuel R.S. Tavares, Mehdi Moradi, Andrew Bradley, Hayit Greenspan, João Paulo Papa, Anant Madabhushi, Jacinto C. Nascimento, Jaime S. Cardoso, Vasileios Belagiannis, and Zhi Lu, editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 204–212, Cham, 2017. Springer International Publishing.

[9] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, Jack Kelly, Jeffrey De Fauw, Michael Heilman, diogo149, Brian McFee, Hendrik Weideman, takacsg84, peterderivaz, Jon, instagibbs, Dr. Kashif Rasul, CongLiu, Britefury, and Jonas Degrave. Lasagne: First release., August 2015. doi:10.5281/zenodo.27878.

[10] Matthew McCormick, Xiaoxiao Liu, Luis Ibanez, Julien Jomier, and Charles Marion. Itk: enabling reproducible research and open science. *Frontiers in Neuroinformatics*, 8, 2014. URL: https://www.frontiersin.org/articles/10.3389/fninf.2014.00013, doi:10.3389/fninf.2014.00013.

[11] Stefan Klein, Marius Staring, Keelin Murphy, Max A. Viergever, and Josien P. W. Pluim. elastix: A toolbox for intensity-based medical image registration. *IEEE Trans. Med. Imaging*, 29(1):196–205, 2010. URL: http://dblp.uni-trier.de/db/journals/tmi/tmi29.html#KleinSMVP10.

[12] Kasper Marstal, Floris Berendsen, Marius Staring, and Stefan Klein. Simpleelastix: A user-friendly, multi-lingual library for medical image registration. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 574–582. IEEE, 2016.

[13] J. Shi D. Ponsa F. Moreno-Noguer E. Riba, D. Mishkin and G. Bradski. A survey on kornia: an open source differentiable computer vision library for pytorch. 2020.

[14] Simple itk, 2020. URL: https://simpleitk.org/.

[15] Robin Sandkühler, Christoph Jud, Simon Andermatt, and Philippe C Cattin. Airlab: Autograd image registration laboratory. 2018.

[16] Airlab. URL: https://github.com/airlab-unibas/airlab.

[17] Stéfan Van Der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. Scikit-image: Image processing in python. *PeerJ (San Francisco, CA)*, 2014(1):e453–e453, 2014.

[18] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy,

Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with numpy. *Nature (London)*, 585(7825):357–362, 2020.

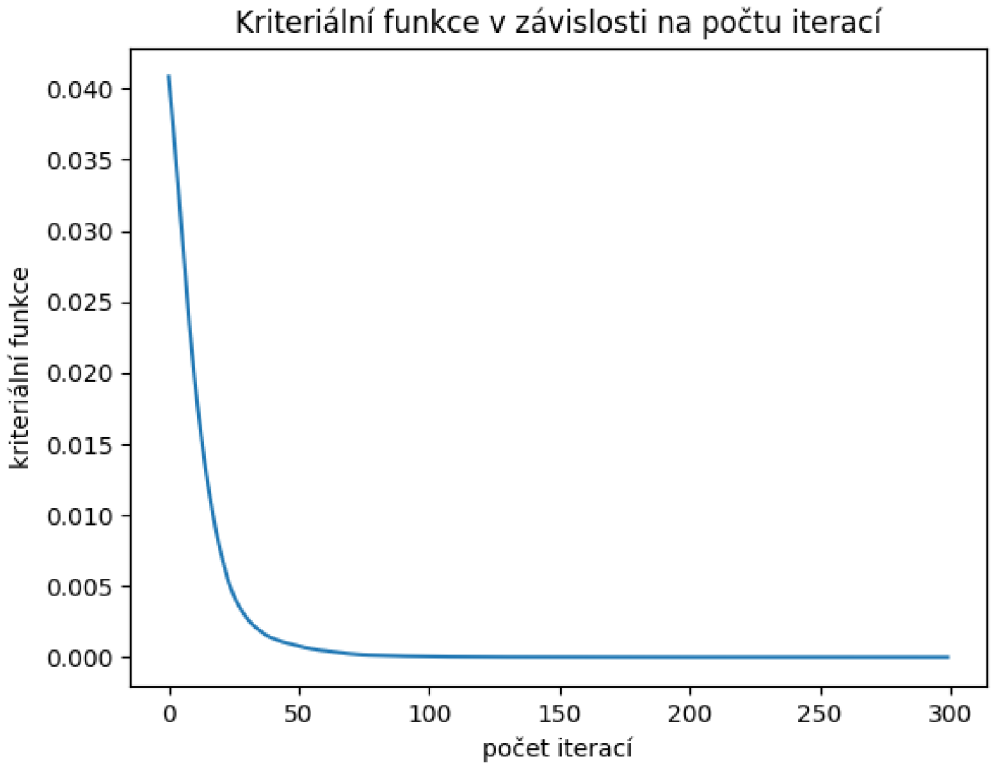
[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 2019.

[20] Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, Carolina C.S. Valentim, Huiying Liang, Sally L. Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, Justin Dong, Made K. Prasadha, Jacqueline Pei, Magdalene Y.L. Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A.N. Huu, Cindy Wen, Edward D. Zhang, Charlotte L. Zhang, Oulan Li, Xiaobo Wang, Michael A. Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M. Anthony Lewis, Huimin Xia, and Kang Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131.e9, 2018.

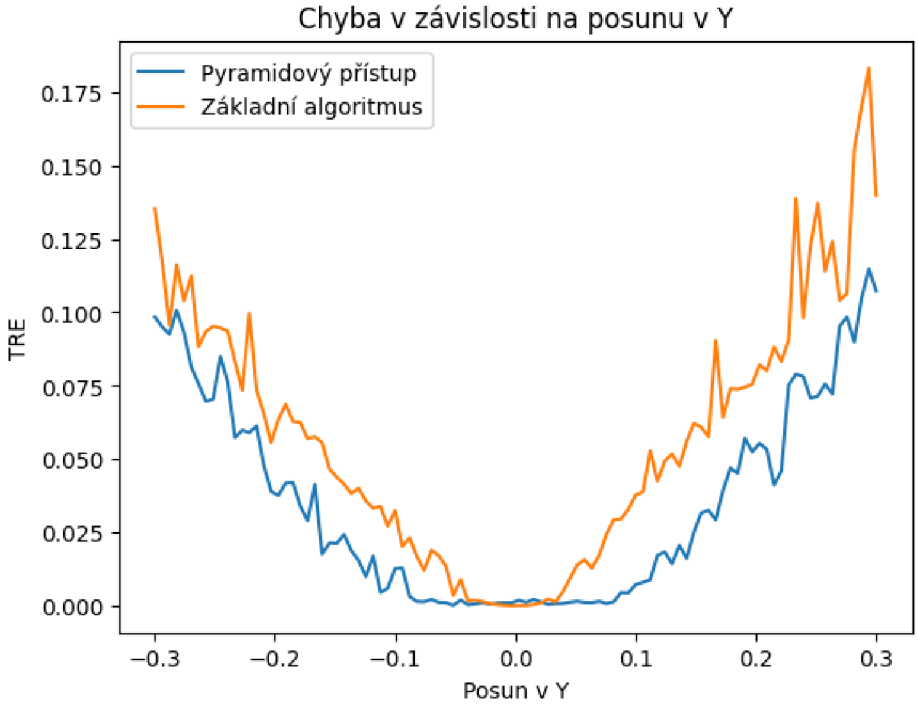
Seznam symbolů a zkratek

CT	počítačová tomografie
MAE	střední absolutní chyba
MSE	střední kvadratická chyba
MRI	magnetická rezonance

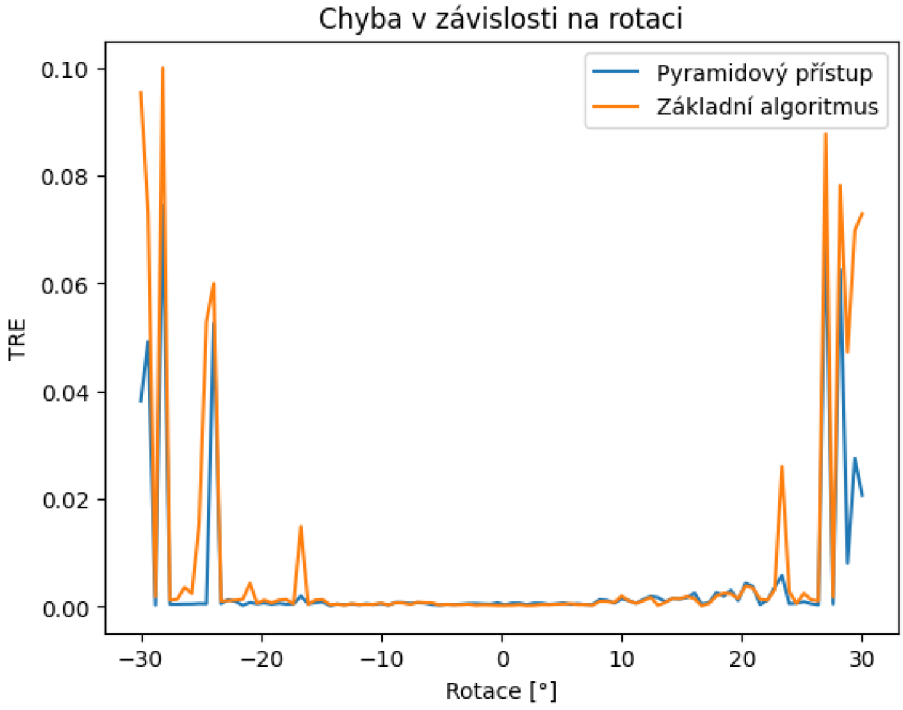
A Grafy testování algortimu a jeho modifikací



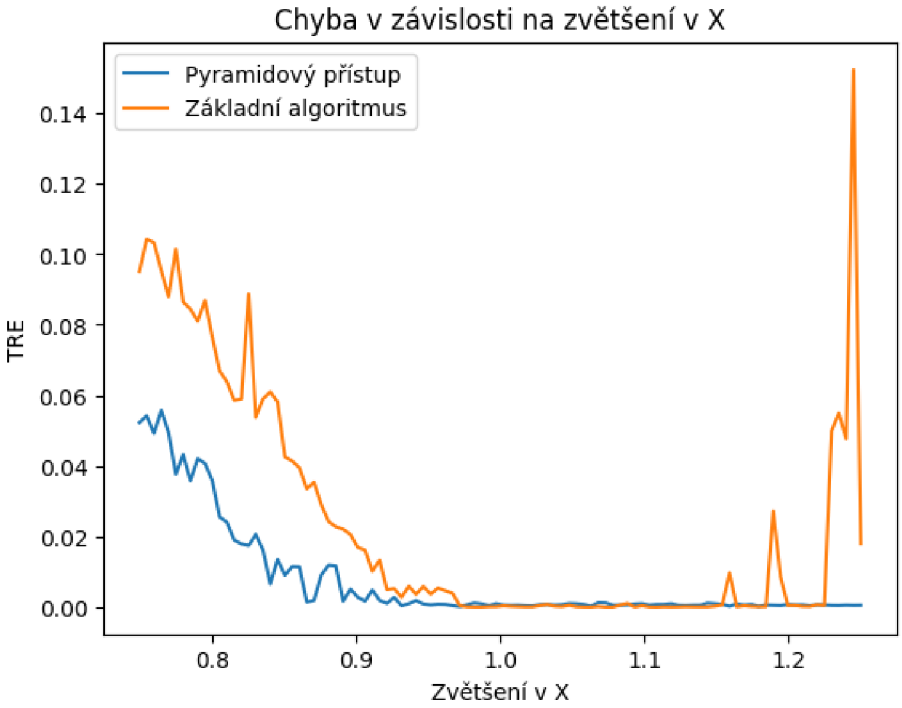
Obr. A.1: Hodnota kriteriální funkce v závislosti na proběhlém počtu iterací pro základní verzi algoritmu



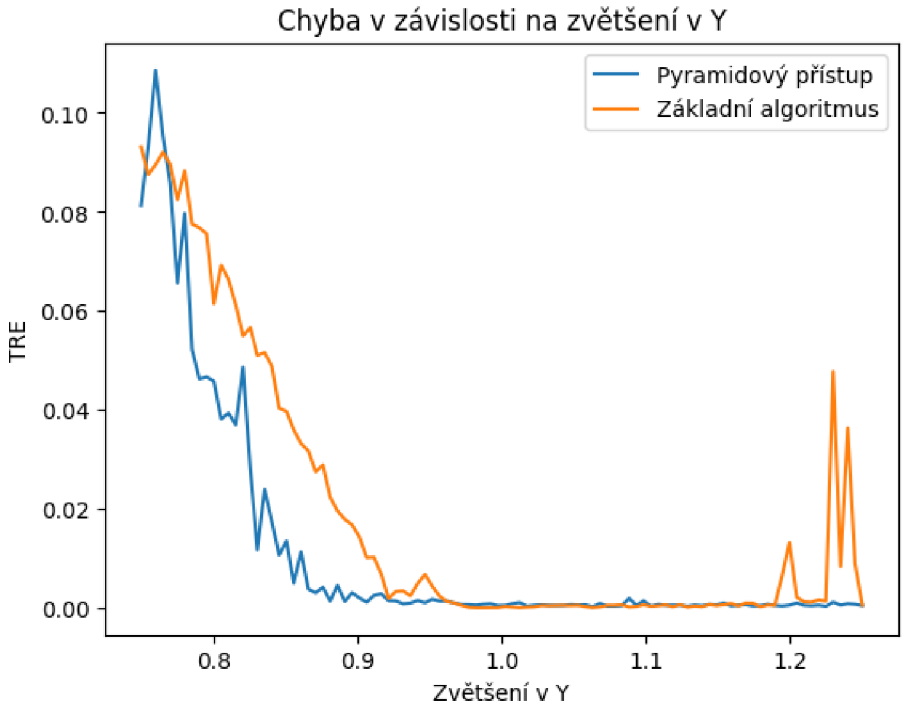
Obr. A.2: Testování posunu v Y



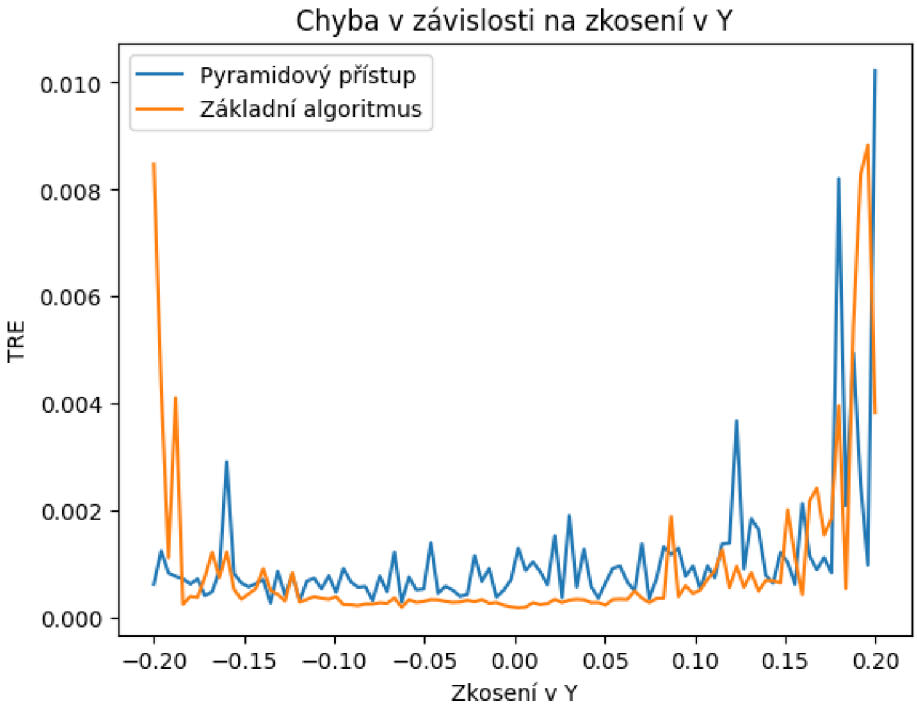
Obr. A.3: Testování rotace



Obr. A.4: Testování zvětšení v X



Obr. A.5: Testování zvětšení v Y



Obr. A.6: Testování zkosení v Y

B Zdrojový kód

Výpis B.1: Funkce vykonávající základní verzi algoritmu pro registraci

```
def affine_registration(ref, sample, max_iters=120, mu=0.02, datatype=to
    # this function performs registration of two 4D pytorch tensors
    # inputs - reference tensor, tensor of moving images, max. number of
    # datatype, verbose mode
    # output - dictionary containing tensor of registered images, shifts
    # loss function
    if verbose:
        print("Begining registration")
    if ref.shape != sample.shape:
        print("ERROR: reference tensor and sample tensor must have same
        print("refrerence dimensions: " + str(ref.shape))
        print("sample dimensions: " + str(sample.shape))
        return 1

    batch_size = ref.shape[0]
    height = ref.shape[2]
    width = ref.shape[3]
    registered_tens = None

    ref = ref.to(used_device)
    sample = sample.to(used_device)

    mask = torch.ones(sample.shape, dtype=datatype, device=used_device)

    x_shift = torch.zeros(batch_size, dtype=datatype, requires_grad=True
    y_shift = torch.zeros(batch_size, dtype=datatype, requires_grad=True
    angle_rad = torch.zeros(batch_size, dtype=datatype, requires_grad=Tr

    x_scale = torch.ones(batch_size, dtype=datatype, requires_grad=True,
    y_scale = torch.ones(batch_size, dtype=datatype, requires_grad=True,

    shear_x = torch.zeros(batch_size, dtype=datatype, requires_grad=True
    shear_y = torch.zeros(batch_size, dtype=datatype, requires_grad=True

    optimizer = torch.optim.Adam([x_shift, y_shift, angle_rad, x_scale,
    losses = []
    for i in range(max_iters):
        if verbose and (i == 0 or i % 5 == 4 or i == max_iters - 1):
```

```
            print("iteration", i + 1, "␣out␣of␣", max_iters)
        # prepare transformation matrix
        T = translation_mat(x_shift, y_shift, datatype, used_device)
        R = rotation_mat(angle_rad, datatype, used_device)
        S = scale_mat(x_scale, y_scale, datatype, used_device)
        SH = shear_mat(shear_x, shear_y, datatype, used_device)
        t_mat = T @ R @ S @ SH
        t_mat = t_mat[:, 0:2, :]


        # transform image
        grid = F.affine_grid(t_mat, sample.shape, align_corners=False)
        registered_tens = F.grid_sample(sample, grid, padding_mode="zero
                                        align_corners=False)
        # transform mask
        with torch.no_grad():
            mask = F.grid_sample(mask, grid, padding_mode="zeros",
                                 align_corners=False)
        # calculate loss function
        diff = (registered_tens - ref) ** 2
        diff = diff * mask
        loss = diff.mean()

        losses.append(loss.detach().cpu().numpy())
        # adjust transformation parameters
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
    if verbose:
        print("Registration␣complete.")
        print("batchsize:␣", batch_size, "height:␣", height, "width:␣",

    output_dict = {
        "x_shifts": x_shift.detach().cpu().numpy(),
        "y_shifts": y_shift.detach().cpu().numpy(),
        "angles_rad": angle_rad.detach().cpu().numpy(),
        "losses": losses,
        "registered_tens": registered_tens.detach().cpu(),
        "transformation_matrices": t_mat.detach().cpu()
    }
    return output_dict
```