UNIVERSITÄT ZU LÜBECK
INSTITUTE OF MATHEMATICS AND
IMAGE COMPUTING

Maschinelles Lernen für Momentum-basierte Bildregistrierung

Machine Learning Methods for Momentum-Based Image Registration

Masterarbeit

im Rahmen des Studiengangs
Mathematik in Medizin und Lebenswissenschaften
der Universität zu Lübeck

vorgelegt von
Stephanie Häger

ausgegeben und betreut von
Prof. Dr. Jan Lellmann
Institute of Mathematics and Image Computing

mit Unterstützung von
Prof. Marc Niethammer, Ph.D.
Department of Computer Science
University of North Carolina

Lübeck, den 30. August 2019

MIC

IM FOCUS DAS LEBEN

Eidesstattliche Erklärung

Ich versichere an Eides statt, die vorliegende Arbeit selbstständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Lübeck, den 30. August 2019 _____

Kurzfassung

In dieser Arbeit wird eine neue Methode des maschinellen Lernens für Momentum-basierte Bildregistrierung vorgestellt. Sie besteht aus einem zweischrittigen Verfahren, das aus einem Prediction Net und einem daran anschließenden Correction Net besteht. Das Gesamtframework sagt das Momentum des SVF (stationary velocity field)-Modells, durch das die gesuchte Transformation der Bildregistrierung parametrisiert ist, patchweise voraus. Durch diese Kombination des maschinellen Lernens mit dem erprobten SVF-Registrierungsmodell kann eine sehr viel schnellere Rechenzeit bei vergleichbarer Registrierungsqualität erreicht werden.
Die vorgestellte Methode wird mit Knieknorpelsegmentierungen des OAI (osteoarthritis initiative)-Datensatzes und Bildern eines synthetischen Datensatzes trainiert und ausgewertet. Sie wird mit der Quicksilver-Methode, auf der die vorgestellte Methode basiert, und dem FlowNet-Ansatz verglichen. Im Gegensatz zum FlowNet kann die vorgestellte Methode aufgrund der patchweisen Voraussagestrategie mit den kleinen zur Verfügung stehenden Trainingsmengen der Datensätze sehr gut umgehen. Nach der Registrierung mit der vorgestellten Methode zeigen die registrierten Bildpaare eine höhere Ähnlichkeit als die mit den beiden Vergleichsmethoden registrierten Bildpaare.

Abstract

In this thesis a new machine learning method for momentum-based image registration is presented. It consists of a two-step overall framework that consists of a Prediction Net and a subsequent Correction Net. This predicts the momentum of the SVF (stationary velocity field) model, which parameterizes the desired transformation of the image registration, in a patchwise fashion. By combining machine learning with the proven SVF registration model, a much faster computing time can be achieved with comparable registration quality.
The presented method is trained and evaluated on knee cartilage segmentations of the OAI (osteoarthritis initiative) data set and images of a synthetic data set. It is compared with the Quicksilver method, on which the presented method is based, and the FlowNet approach. In contrast to FlowNet, the presented method can handle the small amount of training available in the data sets very well due to the patchwise prediction strategy. After registration with the presented method, the aligned image pairs show a higher similarity than the image pairs registered with the two comparison methods.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Arthrose ist die weltweit häufigste Gelenkerkrankung. Sie beginnt mit der Zerstörung des Gelenkknorpels und kann bis zur Freilegung der Knochenoberfläche führen. Dies geht mit Schmerzen und Funktionseinschränkungen einher, die einen deutlichen Verlust an Lebensqualität bedingen. Nach einer Studie des Robert-Koch-Instituts waren 2017 in Deutschland 48,1% der Frauen und 31,2% der Männer in der Gruppe der über 65-Jährigen von Arthrose betroffen. [FKSN17]

Für die Erforschung von Knorpelveränderungen im Knie und die Entwicklung von Vorsorge- und Behandlungsmöglichkeiten wurde von der Osteoarthritis Initiative (OAI) ein Magnetresonanztomographie (MRT)-Bilddatensatz mit Längsschnittbilddaten von mehr als 4.000 Probanden veröffentlicht, die über einen Zeitraum von 8 Jahren aufgenommen wurden. Der Knorpelverlust gilt als dominierender Indikator für das Fortschreiten einer Arthroseerkrankung, bisher sind jedoch nur für etwa 1% der Bilder des OAI Datensatzes Knorpelsegmentierungen öffentlich zugänglich. [PSN08] In Abbildung 1 ist eine 3D-Segmentierung des OAI Datensatzes und eine daraus entnommene 2D-Schicht dargestellt.

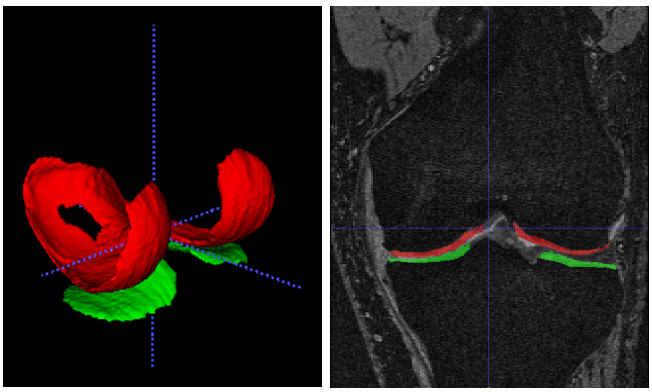


Abbildung 1: Segmentierung des femoralen (rot) und tibialen (grün) Kniegelenkknorpels. **Links:** 3D-Segmentierung. **Rechts:** 2D-Segmentierungsschicht mit Überlagerung des zugehörigen MRT-Bildes.

Mit Hilfe von *Bildregistrierung* kann die Knorpelveränderung eines Patienten zwischen zwei Zeitpunkten analysiert oder eine Populationsanalyse für die Knorpelsegmentierungen mehrerer Patienten durchgeführt werden. Die Bildregistrierung ist eine wichtige Komponente der medizinischen Bildverarbeitung, durch die räumliche Korrespondenzen zwischen zwei Bildern erstellt werden. Das Ziel der Bildregistrierung ist, ein Bild eines Bildpaares plausibel so zu transformieren, dass sich beide Bilder möglichst ähnlich sind. [Mod04]

In den letzten 15 Jahren hat sich das *Large Deformation Diffeomorphic Metric Mapping* (LDDMM)-Modell [BMTY05] als vielseitiges Modell mit vielen Anwendungen im Bereich der Bildregistrierung etabliert. [BMTY05, VRRC12, GFS16] Es wurde entworfen, um diffeomorphe Transformationen zu generieren, die große Deformationen beschreiben können. Diese werden durch die Integration eines zeitabhängigen Geschwindigkeitsfeldes generiert, welches durch das initiale Momentum bestimmt ist. Aufgrund der iterativen, numerischen Berechnungen ist die Registrierung mit dem LDDMM-Modell jedoch mit einem verhältnismäßig hohen Rechenaufwand und Speicherbedarf verbunden. In Abbildung 2 ist das Ergebnis der LDDMM-Registrierung für zwei Quadrate unterschiedlicher Größe dargestellt.

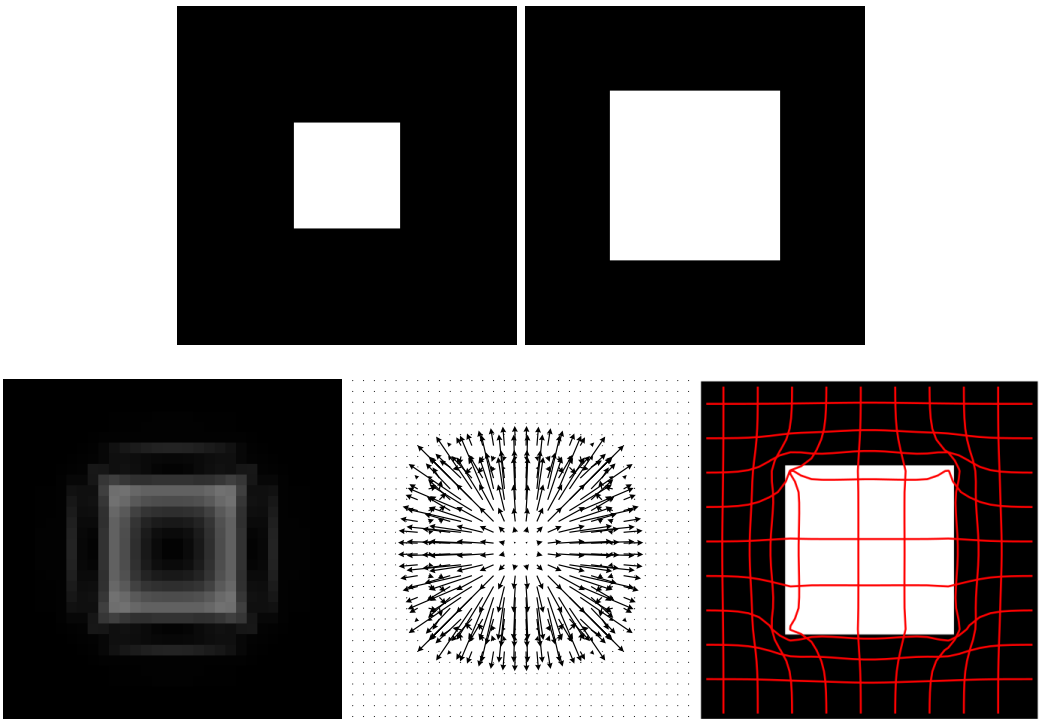


Abbildung 2: Ergebnis der LDDMM-Registrierung mittels numerischer Berechnungen für Bilder der Größe 32×32 Pixel. Die Rechenzeit betrug 29,47 Sekunden. **Oben:** Templatebild (links) und Referenzbild (rechts). **Unten:** Betrag des initialen Momentums (links), Vektorfeld des initialen Momentums (Mitte) und das daraus generierte transformierte Templatebild mit Deformationsfeld (rechts).

Das *stationary velocity field* (SVF)-Modell approximiert diese Methode durch eine Modellierung mit stationären Geschwindigkeitsfeldern. Diese können aus dem stationären Momentum durch Faltung mit einem Differentialglättungsoperator generiert werden. [PSS+16, YLRJ15] Die Vor- und Nachteile des SVF-Modells sind:

+ Es generiert diffeomorphe Transformationen.

+ Es kann große Verformungen erfassen.

+ Es ist bezüglich des Momentums parametrisiert. Dieses muss nicht glatt sein, da das Geschwindigkeitsfeld durch Glättung des Momentums generiert wird.

+ Es erfordert einen geringeren Speicherbedarf und Rechenaufwand als das LDDMM-Modell.

– Die iterativen Berechnungen sind trotz der Einschränkung des Parameterraums für die Approximation immer noch mit einem großen Speicherbedarf und Rechenaufwand verbunden und daher zeitintensiv.

Eine schnellere Alternative bieten *Methoden des maschinellen Lernens* für die Bildregistrierung. Die Entwicklung im Bereich der neuronalen Netze ist in den letzten Jahren stark vorangeschritten. Neuronale Netze finden erfolgreich in verschiedenen Bereichen wie der Spracherkennung [CW08] oder dem autonomen Fahren [BDTD+16] Anwendung. Hierbei haben sich *neuronale Faltungsnetze* (CNNs) als besonders gut für Anwendungen in der Bildverarbeitung, wie Klassifizierung, Segmentierung oder Registrierung erwiesen. [LKB+17] Diese Anwendungen werden während des Trainings durch einen Abgleich der vorausgesagten Werte mit bekannten Referenzwerten erlernt. Für die anschließende Netzauswertung wird nur ein einziger Durchlauf des Netzes benötigt. Die Vor- und Nachteile der Bildregistrierung mit CNNs sind:

+ Die Auswertung ist sehr schnell, da nur ein einziger Durchlauf benötigt wird.

– Es wird eine große Trainingsmenge mit bekannten Referenzwerten benötigt.

In dieser Arbeit wird eine Methode des maschinellen Lernens für die Momentum-basierte Bildregistrierung vorgestellt, die auf der Quicksilver-Methode [YKSN17] basiert und die Vorteile des SVF-Modells mit den Vorteilen eines CNNs verbindet. Hierfür werden der Aufbau der neuen Methode vorgestellt und die Unterschiede zur Quicksilver-Methode [YKSN17] aufgezeigt. Zur Evaluierung wird die in dieser Arbeit vorgestellte Methode auf den Knorpelsegmentierungen des OAI Datensatzes und auf einem synthetischen Datensatz ausgewertet. Zusätzlich wird sie mit der Quicksilver-Methode und dem FlowNet-Ansatz [DFI+15] verglichen.

Auf die Quicksilver-Methode, das FlowNet und weitere verwandte Methoden wird im nächsten Abschnitt genauer eingegangen.

1.2 Verwandte Arbeiten

Es gibt bereits viele Ansätze zur Bildregistrierung mit Methoden des maschinellen Lernens. Sie verfolgen jeweils unterschiedliche Strategien, um Deformationsparameter vorauszusagen. Die wichtigsten Entwicklungen und Ergebnisse der vergangenen Jahre werden in diesem Abschnitt kurz vorgestellt.

Es wurde viel Forschung im Bereich des Erlernens des optischen Flusses [HS80] betrieben. Im Jahr 2015 schätzten Dosovitskiy et al. den optischen Fluss mit einem Enkoder-Dekoder CNN, dem sogenannten *FlowNet* [DFI+15]. Aufgrund seiner U-förmigen Architektur ähnelt es dem UNet [RFB15]. Der Enkoder besteht aus geschichteten Faltungen, die die Merkmalskarten herunterskalieren, während der Dekoder transponierte Faltungen zum hochskalieren der Merkmalskarten verwendet. Verbindungen zwischen dem Enkoder und dem Dekoder verketten die gleichen Auflösungen, um lokale Informationen aller Level aus vorherigen Faltungen zu erhalten. Das Netzwerk sagt den optischen Fluss nach jeder Dekodierungsschicht voraus und verkettet die hochskalierte Version mit der nächsten Ebene.

In der gleichen Publikation wurde ein weiteres Netzwerk *FlowNetC* vorgestellt [DFI+15]. Es berechnet die ersten drei Faltungsebenen getrennt für jedes Bild, anstatt die Bilder vor den Faltungen zu verketten. Die daraus resultierenden Merkmalskarten werden dann durch eine Korrelationsschicht zusammengeführt, welche die Korrelation der Merkmalskarten patchweise berechnet.

In der Arbeit von Wang et al. [WKS+15] werden Deformationen durch Key-Point-Matching vorausgesagt. Dies geschieht unter Verwendung von Sparse Learning gefolgt von einer Generierung dichter Deformationsfelder durch die Interpolation mit radialen Basisfunktionen. Die Leistung der Methode hängt stark von der Genauigkeit bei der Auswahl der Key-Points ab.

In der Publikation von Cao et al. [CSJN15] wird eine semi-gekoppelte Wörterbuch-Lernmethode verwendet, um die Beziehung zwischen den Bildinformationen und den Deformationsparametern des LDDMM-Modells zu modellieren. Es wird jedoch nur ein linearer Zusammenhang zwischen den Bildinformationen und den Deformationsparametern angenommen.

In der Arbeit von Yang et al. [YKSN17] wird ein Framework zur patchweisen Voraussage des initialen Momentums des LDDMM-Modells vorgestellt, die Methode wird *Quicksilver* genannt. Es ist durch zwei aufeinander folgende Enkoder-Dekoder CNNs, dem Prediction Net und dem Correction Net, aufgebaut. Das Prediction Net wird zunächst mit Referenzwerten trainiert, die durch numerische Optimierung des LDDMM-Modells ermittelt wurden. Im Anschluss daran wird das Correction Net

mit den Unterschieden zwischen den Referenzwerten und den durch das Prediction Net vorausgesagten Werten trainiert, um Korrekturen für die Voraussagen des Prediction Nets vorauszusagen.

Die in dieser Arbeit vorgestellte Methode stellt eine Adaption der Quicksilver-Methode für das SVF-Modell dar, die eine andere Vorgehensweise für das Training des Gesamtframeworks verwendet. Zusätzlich wird es in dieser Arbeit nicht für die Registrierung von MRT-Bildern, sondern für die Registrierung der 2D-Knorpelsegmentierungen verwendet.

1.3 Gliederung der Arbeit

Diese Arbeit ist in fünf Kapitel unterteilt. In Kapitel 2 werden zunächst die Grundlagen der Bildregistrierung und des maschinellen Lernens beschrieben. Dabei wird insbesondere auf die Charakteristika der Momentum-basierten Bildregistrierung eingegangen. Darauf folgend wird in Kapitel 3 die neue Methode des maschinellen Lernens für Momentum-basierte Bildregistrierung vorgestellt. Hierbei wird auf die Netzwerkarchitektur des Gesamtframeworks eingegangen und die Vorgehensweise zum Training und zur Auswertung erläutert. Die zur Evaluierung der neuen Methode durchgeführten Experimente und die erzielten Ergebnisse werden im Anschluss in Kapitel 4 beschrieben. Zunächst werden die Ergebnisse der Auswertung der neuen Methode auf dem OAI Datensatz und einem synthetischen Datensatz präsentiert. Im Anschluss daran werden die Ergebnisse der neuen Methode im Vergleich zur Quicksilver-Methode und zum FlowNet vorgestellt. Danach folgt eine Diskussion der experimentellen Ergebnisse. Die Arbeit schließt mit einer Zusammenfassung und einem Ausblick in Kapitel 5.

2 Grundlagen

Im folgenden Kapitel werden die Grundlagen dieser Arbeit vorgestellt. Es ist in drei Abschnitte gegliedert und beginnt mit einer Einführung in die Bildregistrierung basierend auf der Arbeit von J. Modersitzki [Mod04][Mod09] in Abschnitt 2.1. Daraufhin folgt eine Erörterung der Charakteristika der Momentum-basierten Bildregistrierung in Abschnitt 2.2. Das Kapitel schließt mit den Grundlagen des maschinellen Lernens mithilfe künstlicher neuronaler Netze in Abschnitt 2.3.

2.1 Bildregistrierung

2.1.1 Einführung in die Problemstellung

> Das Ziel der (medizinischen) Bildregistrierung ist es, zu einem gegebenen *Referenzbild* $\mathcal{R}$ und einem gegebenen *Templatebild* $\mathcal{T}$ eine sinnvolle *Transformation* zu finden, sodass das transformierte Templatebild dem Referenzbild möglichst ähnlich ist.

Hierfür klären wir zunächst ein paar Grundbegriffe.

Bild

In der Bildregistrierung betrachten wir Bilder, wobei ein *Bild* $\mathcal{I}$ als eine Abbildung eines Bildgebietes $\Omega \subset \mathbb{R}^d$ mit Bilddimension $d \in \mathbb{N}$ in die (mehrdimensionalen) reellen Zahlen $\mathbb{R}^k$ angesehen wird:

$$\mathcal{I}\colon \Omega \subset \mathbb{R}^d \to \mathbb{R}^k. \tag{2.1}$$

Hierbei bezeichnet der Parameter k die Anzahl der Kanäle im Wertebereich des Bildes. Für ein Grauwert- oder Intensitätsbild gilt beispielsweise $k = 1$ und für ein Farbbild, das jeweils einen Kanal für die roten, grünen und blauen Farbanteile besitzt (RGB-Bild), gilt $k = 3$. Der Wertebereich χ kann zusätzlich eingeschränkt werden; typischerweise gilt $\chi = [0,1]^k \subset \mathbb{R}^k$ oder $\chi = \{0,1,\dots 255\}^k \subset \mathbb{N}_0^k$. In dieser Arbeit beschäftigen wir uns mit zweidimensionalen Grauwertbildern, daher gilt im Folgenden $d = 2$, $k = 1$ und $\chi = [0,1]$.

Diskretisierung

Für die numerischen Berechnungen stehen nur endliche Zeit und begrenzte Speicherkapazitäten zur Verfügung, daher erfordern sie diskrete Bilder. Die Diskretisierung der Bilder erfolgt durch ein Abtasten auf einem Gitter. Dafür wird das rechteckige Bildgebiet $\Omega = (\omega_1^1, \omega_2^1) \times (\omega_1^2, \omega_2^2) \subset \mathbb{R}^2$ in ein regelmäßiges Gitter mit $m = \prod_{i=1}^{2} m^i$ Gitterzellen (Pixeln) aufgeteilt, wobei m^i die Anzahl der Gitterzellen

in Richtung x_i, $i \in \{1, \dots, d\}$ bezeichnet. Dann beschreibt $h_i = \frac{\omega_2^1 - \omega_1^1}{m^i}$ die als konstant angenommene Gitterweite (Pixelgröße) in Richtung x_i.

In dieser Arbeit benutzen wir *zellzentrierte Gitter*, die ihre Zellmittelpunkte $x^{(i,j)}$ in der Mitte von Zelle $z_{(i,j)}$ haben für $i \in \{1, \dots, m^1\}$, $j \in \{1, \dots, m^2\}$. Abbildung 3 zeigt ein Beispiel eines zellzentrierten Gitters mit Dimension $d = 2$.

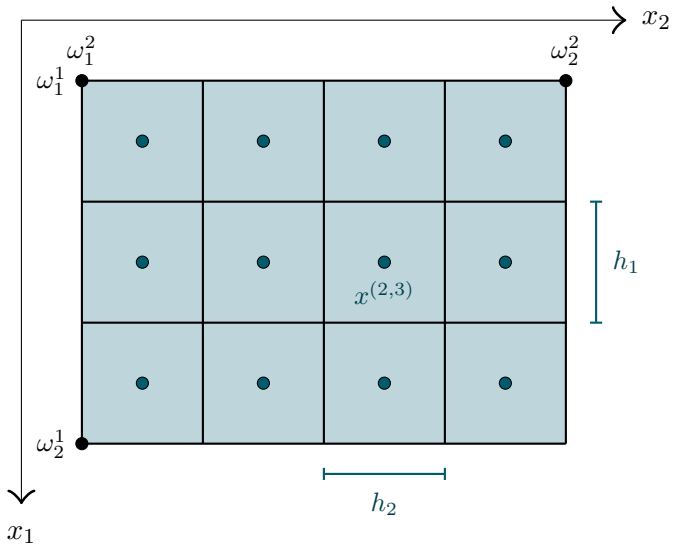


Abbildung 3: Beispiel eines zellzentrierten Gitters mit $m^1 = 3$ und $m^2 = 4$ Zellen auf dem Bildgebiet $\Omega = (\omega_1^1, \omega_2^1) \times (\omega_1^2, \omega_2^2) \subset \mathbb{R}^2$, Gitterweiten h_1 und h_2 sowie dem diskreten Punkt $x^{(2,3)}$.

Zur Unterscheidung bezeichnen kalligraphische Buchstaben, wie $\mathcal{I}$, im Folgenden stetige Bilder sowie deren Maße und normale Buchstaben, wie I, deren Diskretisierungen.

Transformation

Eine *Transformation* oder auch *Deformation* ist eine Funktion $y \colon \Omega \to \mathbb{R}^d$, die das Bildgebiet verändert. Für $y(x) = x + u(x)$ bezeichnet u die Verschiebung der Transformation. Ein transformiertes Bild wird mit $I \circ y$ beschrieben und es gilt $I \circ y(x) = I(x + u(x))$. Dieser Ansatz wird auch *backward Warping* genannt, da jedem Punkt im transformierten Bild eine Position und somit ein Grauwert des ursprünglichen Bildes zugeordnet wird.

Interpolation

Da Bildpunkte des diskreten Templatebildes nach dem Anwenden einer Transformation nicht zwingend wieder auf Gitterpunkten liegen müssen, braucht es Interpolationsmethoden für die Auswertung an Zwischenpunkten. Die in dieser Arbeit vorgestellte Methode verwendet die bilineare Interpolation und die Nächste-Nachbarn-Interpolation.

- Die *bilineare Interpolation* berechnet Bildintensitäten an Zwischenpunkten als gewichtete Summe der Bildintensitäten ihrer vier Nachbarpunkte. Die Gewichte hängen hierbei von der Entfernung der Nachbarpunkte zum neuen Zwischenpunkt ab. Für die Auswertung des Templatebildes an einen Zwischenpunkt $\tilde{x} = (\tilde{x}_1, \tilde{x}_2)$ des zellzentrierten Gitters, wird dieser zunächst in zwei Anteile zerlegt: In einen Anteil $\tilde{x}^G = (\tilde{x}_1^G, \tilde{x}_2^G)$ mit

$$\tilde{x}_d^G = \max\left\{ x_d^{(i,j)} : x_d^{(i,j)} \leq \tilde{x}_d^G, x^{(i,j)} \in \Omega \right\},\ d \in \{1,2\}, \tag{2.2}$$

  der den (auf beide Dimensionen bezogenen) nächstkleineren Gitterpunkt beschreibt, und einen Anteil $\xi = (\xi_1, \xi_2)$ mit

$$\xi_d = \frac{\tilde{x}_d - \tilde{x}_d^G}{h_d},\ 0 \leq \xi_d < 1,\ d \in \{1,2\}, \tag{2.3}$$

  der den normierten Abstand zu den nächstkleineren Gitterpunkten bemisst. Der interpolierte Funktionswert lautet dann

$$\begin{aligned} T^{\text{bilinear}}(\tilde{x}) = T(\tilde{x}_1^G, \tilde{x}_2^G)(1-\xi_1)(1-\xi_2) \ + \ T(\tilde{x}_1^G + h_1, \tilde{x}_2^G)\xi_1(1-\xi_2) \\ + T(\tilde{x}_1^G, \tilde{x}_2^G + h_2)(1-\xi_1)\xi_2 \ + \ T(\tilde{x}_1^G + h_1, \tilde{x}_2^G + h_2)\xi_1\xi_2. \end{aligned} \tag{2.4}$$

- Die *Nächste-Nachbarn-Interpolation* hingegen weist einem Zwischenpunkt die Bildintensität jenes Gitterpunktes zu, der ihm am nächsten liegt:

$$T^{\text{NN}}(\tilde{x}) = \begin{cases} T(\tilde{x}_1^G, \tilde{x}_2^G), & \text{für } \xi_1 \leq \frac{h_1}{2} \text{ und } \xi_2 \leq \frac{h_2}{2}, \\ T(\tilde{x}_1^G + h_1, \tilde{x}_2^G), & \text{für } \xi_1 > \frac{h_1}{2} \text{ und } \xi_2 \leq \frac{h_2}{2}, \\ T(\tilde{x}_1^G, \tilde{x}_2^G + h_2), & \text{für } \xi_1 \leq \frac{h_1}{2} \text{ und } \xi_2 > \frac{h_2}{2}, \\ T(\tilde{x}_1^G + h_1, \tilde{x}_2^G + h_2), & \text{für } \xi_1 > \frac{h_1}{2} \text{ und } \xi_2 > \frac{h_2}{2}. \end{cases} \tag{2.5}$$

Da die Nächste-Nachbarn-Interpolation zu Unstetigkeiten führen kann, wird in dieser Arbeit die bilineare Interpolation zur Berechnung der Transformationen benutzt. Diese ist stückweise differenzierbar und stetig. Für die Anwendung auf Bilder wird hingegen die Nächste-Nachbarn-Interpolation benutzt, um ausschließlich die Bildintensitäten 0 und 1 der Segmentierungen zu erhalten und Grauwerte zu vermeiden. Alternativ könnte auch hier die bilineare Interpolation mit einer anschließenden Rundung durchgeführt werden.

Problemstellung

Zur Erinnerung:

> Das Ziel der (medizinischen) Bildregistrierung ist es, zu einem gegebenen *Referenzbild* $\mathcal{R}$ und einem gegebenen *Templatebild* $\mathcal{T}$ eine sinnvolle *Transformation* ϕ zu finden, sodass das transformierte Templatebild dem Referenzbild möglichst ähnlich ist.

Typischerweise wird das Registrierungsproblem als ein Optimierungsproblem modelliert, da die Distanz zwischen transformiertem Templatebild und Referenzbild minimiert werden soll. Hierfür wird die Distanz durch ein Distanzmaß beschrieben. Allerdings ist die Minimierung der Distanz allein ein schlecht gestelltes Problem [Had02], da sie keine eindeutige Lösung hat. Deshalb wird zusätzlich ein Regularisierer eingeführt, über den Anforderungen an die Plausibilität der Transformation gestellt werden können. Auf diese Weise ergibt sich ein korrekt gestelltes Optimierungsproblem

$$E(\phi) = \mathcal{S}[\phi] + \frac{1}{\sigma^2}\, \mathcal{D}[\mathcal{T} \circ \phi, \mathcal{R}], \tag{2.6}$$

$$\phi^* = \underset{\phi\colon \Omega \to \mathbb{R}^d}{\arg\min}\; E(\phi), \tag{2.7}$$

dessen Lösung die gesuchte Transformation $\phi^*\colon \Omega \to \mathbb{R}^d$ ist. Das *Energiefunktional* E besteht aus einem *Regularisierer* $\mathcal{S}$ und einem *Distanzmaß* $\mathcal{D}$, das durch den Parameter $\sigma > 0$ gewichtet wird. Je größer σ gewählt wird, desto mehr Gewicht wird auf die Plausibilität der Transformation und weniger Gewicht auf die Ähnlichkeit der Bilder gelegt.

Bei der Transformation ϕ unterscheiden wir zwischen parametrischer und nicht-parametrischer Modellierung. Parametrische Modelle nutzen vergleichsweise niedrigdimensionale Parametrisierungen der Transformation. Beispiele sind starre und affine Transformationen. Nicht-parametrische Modelle hingegen parametrisieren die Transformation lokal mit einem Parameter oder einem Parametervektor für jeden Pixel. Ein üblicher Ansatz ist die Parametrisierung durch ein *Verschiebungsfeld*

$$u(x) = \phi(x) - x. \tag{2.8}$$

Die Regularisierung beruht dann auf der Bestrafung von Normen der räumlichen Ableitungen der Verschiebungsvektoren. Beispiele für Regularisierer sind der diffusive Regularisierer $\mathcal{S}^{\text{diff}}[\phi] = \frac{1}{2} \cdot \int_\Omega \sum_{j=1}^d \|\nabla u_j\|^2 \mathrm{d}x$, der die Glattheit der Transformation durch das Bestrafen großer erster Ableitungen fordert, oder der Krümmungsregularisierer $\mathcal{S}^{\text{curv}}[\phi] = \frac{1}{2} \cdot \sum_{j=1}^d \int_\Omega (\Delta u_j)^2 \mathrm{d}x$. Auf die Form von Distanzmaßen gehen wir im folgenden Abschnitt 2.1.2 genauer ein.

2.1.2 Distanzmaße

Ein sehr intuitives Distanzmaß, das die Ähnlichkeit zweier Bilder anhand der punktweisen Differenzen der Bildintensität misst, ist die *Summe der quadrierten Differenzen* (engl. sum of squared differences (SSD)). Es hat die Form

$$\mathcal{D}^{\mathrm{SSD}}[\mathcal{T}, \mathcal{R}] = \frac{1}{2}\|\mathcal{T} - \mathcal{R}\|_{L^2}^2 \tag{2.9}$$

und ist durch den direkten Vergleich der Bildintensitäten besonders gut für Bilder der gleichen Bildmodalität geeignet.

Da in dieser Arbeit ein Datensatz mit Knieknorpelsegmentierungen verwendet wird, der eine große Variation in der Größe der Segmentierungen aufweist, wird für die Registrierung das auf der *normalisierten Kreuzkorrelation* (NCC) basierende Distanzmaß $\mathcal{D}^{\mathrm{NCC}}$ verwendet, das diese gut handhaben kann. Es hat die Form

$$\mathcal{D}^{\mathrm{NCC}}[\mathcal{T}, \mathcal{R}] = 1 - \mathrm{NCC}^2[\mathcal{T}, \mathcal{R}] = 1 - \left(\frac{\langle \mathcal{T} - \overline{\mathcal{T}}, \mathcal{R} - \overline{\mathcal{R}} \rangle}{\|\mathcal{T} - \overline{\mathcal{T}}\| \cdot \|\mathcal{R} - \overline{\mathcal{R}}\|}\right)^2, \tag{2.10}$$

wobei $\overline{\mathcal{T}}$ und $\overline{\mathcal{R}}$ die Mittelwerte der Bildintensitäten von $\mathcal{T}$ und $\mathcal{R}$ bezeichnen und $\|\cdot\| = \sqrt{\langle \cdot, \cdot \rangle}$ die mittelwertfreie Kreuzkorrelation $\langle \mathcal{T} - \overline{\mathcal{T}}, \mathcal{R} - \overline{\mathcal{R}} \rangle$ normiert. Hierfür wird angenommen, dass $\mathcal{T}$ und $\mathcal{R}$ jeweils nicht konstant 0 sind. Durch das Subtrahieren von 1 ist der Wertebereich des Distanzmaßes beschränkt auf das Intervall $[0, 1]$. Ein kleinerer Wert bezeichnet eine höhere Korrelation.

2.1.3 Lösen des Registrierungsproblems

Das Registrierungsproblem (2.7) kann mit iterativen Verfahren der Optimierung gelöst werden. Im Folgenden wird das in dieser Arbeit verwendete *Gradientenabstiegsverfahren* beispielhaft an der Berechnung des Minimierers

$$x^\star = \operatorname*{arg\,min}_{x \in \mathbb{R}^d} f(x)$$

einer beliebigen, stetig differenzierbaren Funktion $f \colon \mathbb{R}^d \to \mathbb{R}$ vorgestellt.

Gradientenabstiegsverfahren

1. Zunächst wird ein Startwert $x^0 \in \mathbb{R}^d$ gewählt.
2. ↻ Bis zuvor festgelegte Abbruchbedingungen erfüllt werden, wird das Update

$$x^k = x^{k-1} - \alpha^k \cdot \nabla f(x^{k-1}) \tag{2.11}$$

iterativ durchgeführt. Hierbei bezeichnet

- $k \in \mathbb{N}_0$ die aktuelle Iteration,
- $-\nabla f(x^{k-1})$ die Richtung des steilsten Abstiegs, in der das Gradientenabstiegsverfahren den Minimierer vermutet, und
- $\alpha^k \in \mathbb{R}_{>0}$ die Schrittweite, die fest gewählt oder zum Beispiel mittels der Armijo-Regel [Arm66] in jeder Iteration neu bestimmt wird. Die Wahl der Schrittweite spielt eine wichtige Rolle, da eine zu große Wahl einen Schritt über den Minimierer hinaus bedeuten kann. Eine zu kleine Wahl kann hingegen dazu führen, dass der Algorithmus in einem lokalen Minimum endet oder mehr Iterationen benötigt und somit einem höheren Rechenaufwand erfordert.

Als Abbruchbedingungen können eine maximale Anzahl an Iterationen $k_{\max}$, ein Schwellwert für die Genauigkeit oder eine Kombination daraus wie zum Beispiel den Gill-Murray-Wright-Abbruchbedingungen [GMW81] dienen.

Alternative Verfahren zur Optimierung sind das stochastische Gradientenabstiegsverfahren, das (Quasi-) Newton-Verfahren oder das (L-)BFGS-Verfahren. Details zu diesen Verfahren finden sich in [NW06].

2.2 Momentum-basierte Bildregistrierung

In diesem Abschnitt erörtern wir die Charakteristika der Momentum-basierten Bildregistrierung und folgen dabei den Ausführungen in [YKSN17, BMTY05, HZN09, Pol18].

2.2.1 Large Deformation Diffeomorphic Metric Mapping

Der Ausdruck *Large Deformation Diffeomorphic Metric Mapping* (LDDMM) wurde von Beg et al. [BMTY05] geprägt und umfasst die wesentlichen Merkmale dieser Registrierungsmethode. LDDMM wurde entwickelt, um diffeomorphe Abbildungen zu erhalten, die große Verformungen beschreiben, und bietet zusätzlich eine Metrik sowohl im Bereich der Diffeomorphismen als auch im Bereich der Bilder. Es folgt eine kompakte Einführung in LDDMM. Mehr Details zum theoretischen Hintergrund liefern [BMTY05, HZN09, DGM98, Pol18].

Für viele Anwendungen ist es wünschenswert, eine invertierbare Transformation ϕ zu finden. Zum Beispiel können statistische Analysen zur Modellierung der anatomischen Variabilität einer Population mit dieser durchgeführt werden [HPO08]. Zusätzlich sollen ϕ und ϕ^{-1} ausreichend glatt sein, damit die Existenz von Eigenschaften wie der Krümmung von Kurven und Oberflächen gesichert ist und die Topografie der anatomischen Strukturen erhalten bleibt. [BMTY05]
Das LDDMM-Modell verfolgt dieses Ziel, indem es sicherstellt, dass die Transformation, die zwischen zwei Bildern berechnet wird, diffeomorph ist. Diffeomorphe Transformationen sind geeignet, da diese stetig differenzierbare, invertierbare Transformationen mit stetig differenzierbaren Inversen sind. Hierzu werden im LDDMM-Modell Elemente der Gruppe der Diffeomorphismen $\mathrm{Diff}(\Omega)$ auf dem Gebiet $\Omega \subset \mathbb{R}^d$ als mögliche Transformation ϕ betrachtet. Das LDDMM-Modell modelliert die Transformation als Endpunkt $\phi = \phi_1$ des Flusses eines zeitabhängigen Geschwindigkeitsfeldes $v_t \colon \Omega \to \mathbb{R}^d$, gegeben durch die gewöhnliche Differentialgleichung (GDGL)

$$\frac{\partial \phi_t^v}{\partial t} = v_t(\phi_t^v). \tag{2.12}$$

Anmerkung: Wenn aus dem Kontext erkenntlich, unterdrücken wir räumliche Abhängigkeiten zur besseren Lesbarkeit der Notation und geben nur die Zeitvariable als Index an, zum Beispiel schreiben wir ϕ_t für $\phi(x,t)$. Das hochgestellte v zeigt die Abhängigkeit einer Transformation ϕ vom zugehörigen Geschwindigkeitsfeld v an.

Die GDGL (2.12) definiert einen Pfad $t \mapsto (\phi_t^v \colon \Omega \to \Omega)$, $t \in [0,1]$, im Raum der diffeomorphen Transformationen, der bei $\phi_0 = \mathrm{Id}$, mit der Identitätstransformation $\mathrm{Id}(x) = x,\ \forall x \in \Omega$, beginnt und beim Endpunkt $t = 1$ des Flusses für die spezielle Transformation $\phi^v = \phi_1^v = \phi_0^v + \int_0^1 v_t(\phi_t^v)\mathrm{d}t$, die die gegebenen Bilder verbindet, endet. Als hinreichende Bedingung für die Korrektgestelltheit der Gleichung (2.12)

wird angenommen, dass v_t in einem *reproducing kernel Hilbert space* (RKHS) V liegt [You10]. Mehr Informationen zu RKHS finden sich in [Sai88], [BRV12].

Typischerweise wird v_t im LDDMM-Modell durch die Minimierung der L^2-Norm eines auf v_t angewandten Differentialoperators F regularisiert. Dieser definiert den reproduzierenden Kern [Sai88] $K^{-1} := L := F^*F$ des Raumes als

$$\int_0^1 \|v_t\|_V \mathrm{d}t = \int_0^1 \|Fv_t\|_{L^2} \mathrm{d}t = \langle Fv_t, Fv_t\rangle_{L^2} = \langle Lv, v\rangle_{L^2}. \tag{2.13}$$

In der Praxis wird typischerweise ein Gaußkern für K gewählt. In dieser Arbeit wird ein multi-Gaußkern [BRV12] G_m verwendet, der durch die gewichtete Summe verschiedener Gaußkerne $G_m = \sum_i w_i G_{\sigma_i}$ generiert wird.

Relaxations-Formulierung

Das zu minimierende Energiefunktional (2.7) bezüglich des zeitabhängigen Geschwindigkeitsfeldes mit SSD-Datenterm lässt sich nun als

$$\begin{aligned} E(v) &= \int_0^1 \|v_t\|_V^2 \mathrm{d}t + \frac{1}{\sigma^2}\|\mathcal{T}\circ\phi_1^{-1} - \mathcal{R}\|_{L^2}^2, \\ \text{s.t.} \quad & \frac{\partial}{\partial t}\phi_t^v = v_t(\phi_t^v), \quad \phi_0 = \mathrm{Id} \end{aligned} \tag{2.14}$$

formulieren. Zur besseren Übersicht wird das Energiefunktional hier mit dem SSD-Distanzmaß $\mathcal{D}^{\mathrm{SSD}}[\mathcal{T},\mathcal{R}] = \|\mathcal{T}-\mathcal{R}\|_{L^2}^2$ dargestellt. In der Implementierung wird jedoch das NCC-Distanzmaß $\mathcal{D}^{\mathrm{NCC}}$ verwendet. Die Differentialgleichungs-Nebenbedingung für ϕ lässt sich in den Euler-Koordinaten, die das zeitabhängige Geschwindigkeitsfeld zu jedem Zeitpunkt in der aktuellen Konfiguration beschreiben und somit von der zuvor erfolgten Transformation abhängen, durch die Gleichung

$$\frac{\partial}{\partial t}(\phi_t^v)^{-1} + (D(\phi_t^v)^{-1})v_t = 0, \tag{2.15}$$

formulieren, wobei D die Jacobi-Matrix bezeichnet. In dieser LDDMM-Formulierung, die auch als *Relaxations-Formulierung* bezeichnet wird, wird die Registrierung durch das volle zeitabhängige Geschwindigkeitsfeld $v(x,t)$ parametrisiert. Würde zum Zeitpunkt $t=0$ eine Menge an Teilchen in das Bild gesetzt, so wird die Transformation durch Verfolgen dieser Teilchen im Geschwindigkeitsfeld über die Zeit erhalten. [YKSN17]

Um über das zeitabhängige Geschwindigkeitsfeld zu optimieren

$$v^\star = \underset{v:\ \frac{\partial\phi_t^v}{\partial t}=v_t(\phi_t)}{\arg\inf} \int_0^1 \|v_t\|_V^2 \mathrm{d}t + \frac{1}{\sigma^2}\|\mathcal{T}\circ\phi_1^{-1} - \mathcal{R}\|_{L^2}^2, \tag{2.16}$$

wird das zugehörige sogenannte adjungierte System (engl. adjoint system) rückwärts in der Zeit gelöst, wobei die finalen Bedingungen des adjungierten Systems durch die aktuelle Bildabweichung bestimmt werden, die mit dem gewählten Distanzmaß gemessen wird [BMTY05]. Dieses adjungierte System kann über einen Optimierungsansatz mit Nebenbedingungen [HZN09] bestimmt werden. Aus der Lösung des adjungierten Systems lassen sich die Gradienten der LDDMM-Energie in Bezug auf das Geschwindigkeitsfeld für jeden Zeitpunkt berechnen. Aufgrund des beschränkten Rahmens dieser Arbeit verweisen wir für Details des adjungierten Systems auf [VRRC12, Pol18, BMTY05] und liefern als Ergebnis die Gradienten der LDDMM-Energie (2.14) bezüglich des Geschwindigkeitsfeldes für den Zeitpunkt $t \in [0,1]$

$$(\nabla_v E_t)_V = 2v_t - K\Big(\frac{2}{\sigma^2}|D\phi^v_{t,1}|\nabla J^0_t(J^0_t - J^1_t)\Big), \tag{2.17}$$

wobei $\phi_{s,t}\colon \Omega \to \Omega$ die Verknüpfung $\phi_{s,t} = \phi_t \circ (\phi_s)^{-1}$ bezeichnet. Der Ausdruck $\phi_{s,t}(y)$ beschreibt also die Position des Teilchens zum Zeitpunkt t, das sich zum Zeitpunkt s an Position y befindet. Zudem gilt $J^0_t = \mathcal{T} \circ \phi_{t,0}$, $J^1_t = \mathcal{R} \circ \phi_{t,1}$.

Mit Hilfe der Gradienten kann das Optimierungsproblem (2.16) numerisch gelöst werden. Hierzu wird in dieser Arbeit das folgenden Gradientenabstiegsverfahren verwendet, das in [BMTY05] vorgestellt wurde:

Gradientenabstiegsverfahren für LDDMM

1. Initialisierung: $k = 0,\ v^k_{t_j} = 0,\ \nabla_{v^k} E_{t_j} = 0, \phi_{t_j,0} = \mathrm{Id},\ \phi_{t_j,1} = \mathrm{Id},$ $\forall t_j \in [0,1],$ mit $j \in \{0,\dots,N-1\},\ N \in \mathbb{N}$
2. Für $k = 0,1,2,\dots,k_{\max}$:
   a) $v^{k+1} = v^k - \alpha \nabla_{v^k} E$
   b) Für $j = N-1 \dots 0$ berechne $\phi^{v^{k+1}}_{t_j,1}$.
   c) Für $j = 0 \dots N-1$ berechne $\phi^{v^{k+1}}_{t_j,0}$.
   d) Für $j = 0 \dots N-1$ berechne $J^0_{t_j} = \mathcal{T} \circ \phi^{v^{k+1}}_{t_j,0}$.
   e) Für $j = N-1 \dots 0$ berechne $J^1_{t_j} = \mathcal{R} \circ \phi^{v^{k+1}}_{t_j,1}$.
   f) Für $j = 0 \dots N-1$ berechne $\nabla J^0_{t_j}$.
   g) Für $j = 0 \dots N-1$ berechne $|D\phi^{v^{k+1}}_{t_j,1}|$.
   h) Für $j = 0 \dots N-1$ berechne $\nabla_{v^{k+1}} E$ mittels (2.17).
   i) Stopp, falls $\|\nabla_{v^{k+1}} E\|$ kleiner als ein festgelegter Schwellwert ϵ.
   j) Berechne die neue Energie $E(v^{k+1})$ mittels (2.14).
   k) Stopp, falls $k = k_{\max}$, sonst $k = k+1$.
3. Return: v^{k+1}

Die Schritte b) und c) werden durch eine Integration der Differentialgleichungs-Nebenbedingung (2.15) vorwärts und rückwärts in der Zeit berechnet. Im adjungierten System hat diese nach dem Wechsel der Variablen durch $\phi_t^v(x) = y$ die Form [BMTY05]

$$\frac{\partial}{\partial t}\phi_{t,0}^v(y) + D\phi_{t,0}^v(y)v_t(y) = 0. \tag{2.18}$$

Bei Konvergenz wird die optimale Lösung die Optimalitätsbedingungen der eingeschränkten LDDMM-Energie von Gleichung (2.14) erfüllen. Eine intuitive Veranschaulichung wird in [YKSN17] gegeben: Sucht man den kürzesten Pfad zwischen zwei Punkten, würde man (im euklidischen Raum) die gerade Linie erhalten, die diese beiden Punkte verbindet. Diese gerade Linie ist der geodätische Pfad im euklidischen Raum.
Durch das LDDMM-Modell wird stattdessen der kürzesten Pfad zwischen zwei Bildern basierend auf dem Minimierer von (2.14) gesucht. Die Optimierung entspricht dem Beginn mit einem möglichen Pfad und der sukzessiven Verbesserung, bis der optimale Pfad gefunden ist. Beim zuvor genannten Beispiel der übereinstimmenden Punkte würde man mit jedem möglichen Pfad beginnen, der die beiden Punkte verbindet, und es dann sukzessive verbessern. Das Ergebnis bei Konvergenz ist der optimale geodätische Weg. Dies schlägt eine alternative Optimierungsformulierung vor, die wir im nächsten Abschnitt erläutern.[YKSN17]

Shooting-Formulierung

Das Beispiel des Pfades zwischen zwei Punkten lässt sich fortführen: Wenn man weiß, dass die optimale Lösung eine gerade Linie (d.h. eine geodätische) sein muss, kann man erwägen, nur über den Raum der Geraden zu optimieren, anstatt über alle möglichen Pfade, die die beiden Punkte verbinden. Dies entspricht dem physikalischen Prinzip der Impulserhaltung. Diese Modellierung reduziert den Parameterraum für die Optimierung, da nur über den y-Achsenabschnitt und die Steigung der Geraden optimiert wird. [YKSN17]

Analog kann für das LDDMM-Modell eine Impulserhaltung formuliert werden. Es ergibt sich die sogenannte *Shooting-Formulierung*, die die Transformation über das initiale *Momentum* $m_0 = m(0)$ und die initiale Transformation $\phi^{-1}(0)$ parametrisiert, aus dem die Transformation ϕ_t für jeden Zeitpunkt berechnet werden kann. Dabei entspricht das initiale Momentum der Steigung der Linie und die initiale Transformation dem y-Achsenabschnitt. [SHJF13]
Die geodätischen Gleichungen, die mit den Optimalitätsbedingungen übereinstimmen, entsprechen der Liniengleichung [YKSN17]. Im Wesentlichen erzwingt die Shooting-Formulierung also diese Optimalitätsbedingungen von (2.14) als Nebenbedin-

gung. Es wird nur über geodätische Pfade gesucht, da diese Optimalitätsbedingungen geodätische Gleichungen sind. Sie können in Bezug auf das Momentum m allein formuliert werden. Insbesondere ist das Momentum über einen positiv-definiten Differentialglättungsoperator K mit dem Geschwindigkeitsfeld v durch

$$v = Km \text{ und } m = Lv \tag{2.19}$$

verbunden, wobei L das Inverse von K und v ein Element im RKHS V ist.
Durch m_0 wird die gesamte räumlich-zeitliche Transformation $\phi(x,t)$ bestimmt. Das zu minimierende Energiefunktional der LDDMM Shooting-Formulierung lautet [SHJF13]

$$\begin{aligned} E(m_0) &= \langle m_0, Km_0 \rangle + \frac{1}{\sigma^2} \|\mathcal{T} \circ \phi_1^{-1} - \mathcal{R}\|_{L^2}^2, \\ \text{s.t.} \quad & \frac{\partial}{\partial t} m + \mathrm{ad}_v^\star m = 0, \\ & m(0) = m_0, \\ & \frac{\partial}{\partial t} \phi^{-1} + D\phi^{-1} v = 0, \\ & \phi^{-1}(0) = \mathrm{Id}, \\ & m - Lv = 0, \end{aligned} \tag{2.20}$$

wobei Id die Identitätstransformation ist. Die Variablen σ, $\mathcal{T}$ und $\mathcal{R}$ sind fest gegeben. Die Variablen m_0, m, v und ϕ werden durch die Optimierung ermittelt, wobei m, v und ϕ vollständig durch m_0 bestimmt sind. Der Operator $\mathrm{ad}^\star$ ist der duale Operator der sogenannten *adjungierten Aktion* (engl. adjoint action) $\mathrm{ad}\colon V \times V \to V$ von V auf sich selbst, die für beliebige aber feste $u, v \in V$ definiert ist als

$$\mathrm{ad}_u(v) \coloneqq \partial_\phi (D\phi \cdot v \circ \phi^{-1})|_{\phi=\mathrm{Id}} u = Du \cdot v - Dv \cdot u. \tag{2.21}$$

Der Optimierungsansatz für die Minimierung von (2.20) ist ähnlich wie bei der Relaxations-Formulierung: es wird das adjungierte System für die Shooting-Formulierung bestimmt und daraus der Gradient in Bezug auf das unbekannte initiale Momentum m_0 ermittelt [SHJF13, VRRC12]. Basierend auf diesem Gradienten kann eine optimale Lösung durch ein Gradientenabstiegsverfahren bestimmt werden.

2.2.2 Stationary Velocity Field

Die Berechnung der zeitabhängigen Differentialgleichung des LDDMM bringt einen hohen Rechenaufwand mit sich. Die Parametrisierung mittels *stationärer Geschwindigkeitsfelder* (engl. stationary velocity field (SVF)) führt hingegen zu effizienteren Berechnungen. Die folgende Zusammenfassung der Parametrisierung mittels SVF folgt den Ausführungen in [PSS+16] und [YLRJ15].

Bei der SVF-Modellierung handelt es sich um eine Approximation des LDDMM-Modells. Hierbei ist das Geschwindigkeitsfeld $v(x,t)$ konstant über die Zeit. Die Pfade, die durch SVFs parametrisiert sind, stellen eine Untergruppe von $\mathrm{Diff}(\Omega)$ dar. Sie unterscheiden sich von den Pfaden der Shooting-Formulierung darin, dass sie keine Metrik haben; der Pfad wird auf einen einzelnen Punkt reduziert. Für den Definitionsbereich Ω vom Referenzbild $\mathcal{R}$ mit räumlichen Positionen $x \in \Omega$ seien $G \subset \mathrm{Diff}(\Omega)$ ein Unterraum, der alle diffeomorphen Transformationen enthält, die durch SVFs parametrisiert werden, und V der Tangentialraum von G in der Identität Id, der alle Geschwindigkeitsfelder v enthält. Dann wird ein Pfad von Diffeomorphismen durch die Gleichung

$$\frac{\partial \phi_t^v}{\partial t} = v(\phi_t^v) \tag{2.22}$$

mit Anfangsbedingung $\phi_0 = \mathrm{Id}$ definiert. Dies entspricht Gleichung (2.12) für zeitabhängige Geschwindigkeitsfelder.

Die finale Transformation kann durch eine Euler-Integration der Differentialgleichung (2.22) approximiert werden, wie in [Ash07] gezeigt. In der vorgestellten Methode wird sie hingegen durch die Runge-Kutta-Integration 4. Ordnung vorwärts und rückwärts in der Zeit berechnet. Eine detaillierte Beschreibung dieser Runge-Kutta-Integration ist in [Pol18] zu finden.

Das Lösen des Optimierungsproblem erfolgt nun also durch Optimierung über $v(x)$ statt $v(x,t)$, beziehungsweise $m(x)$ statt $m(x,t)$, mittels Gradientenabstiegsverfahren. In Abbildung 4 ist der Unterschied zwischen den Strategien zur Berechnung der gesuchten Transformation von der Relaxations- und Shooting-Formulierung des LDDMM-Modells und der Approximation mittels SVF skizziert.

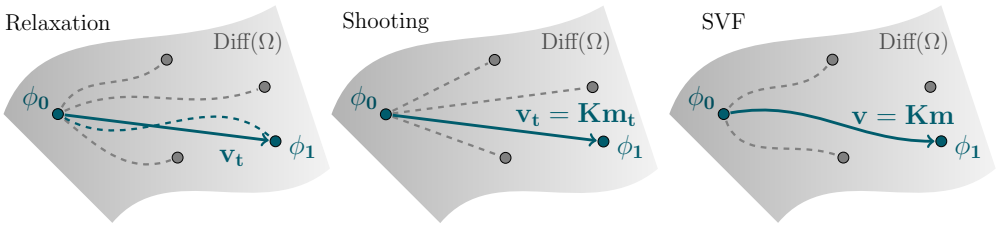


Abbildung 4: Unterschiedliche Strategien zur Berechnung der Deformation ϕ_1. **Relaxation**: Alle Diffeomorphismen können erreicht werden. Verschiedene Geschwindigkeitsfelder können denselben Diffeomorphismus generieren; das Optimum ist die geodätische Linie. **Shooting**: Alle Diffeomorphismen können über eine geodätische Linie erreicht werden. **SVF**: Nicht alle Diffeomorphismen können erreicht werden; die Pfade sind keine geodätischen Linien.

Die Einschränkung des Parameterraums im SVF-Modell führt dazu, dass nur Diffeomorphismen des eingeschränkten Parameterraums generiert werden können. In Experimenten der Registrierung von Hirn-Magnetresonanzbildern konnten mit der stationären Parametrisierung jedoch ähnliche Ergebnisse wie mit zeitabhängigen Parametrisierung erzielt werden. [HPO08] Die Einschränkung des Parameterraums führt zusätzlich zu einer Reduzierung des Speicherbedarfs und des Rechenaufwands und dadurch zu einer Verbesserung der Laufzeit gegenüber den zeitabhängig parametrisierten Formulierungen des LDDMM-Modells. In Abbildung 5 ist das Ergebnis der SVF-Registrierung für zwei Quadrate unterschiedlicher Größe dargestellt. Die numerischen Berechnungen benötigten nur ein Drittel der vom LDDMM-Modell benötigten Rechenzeit.

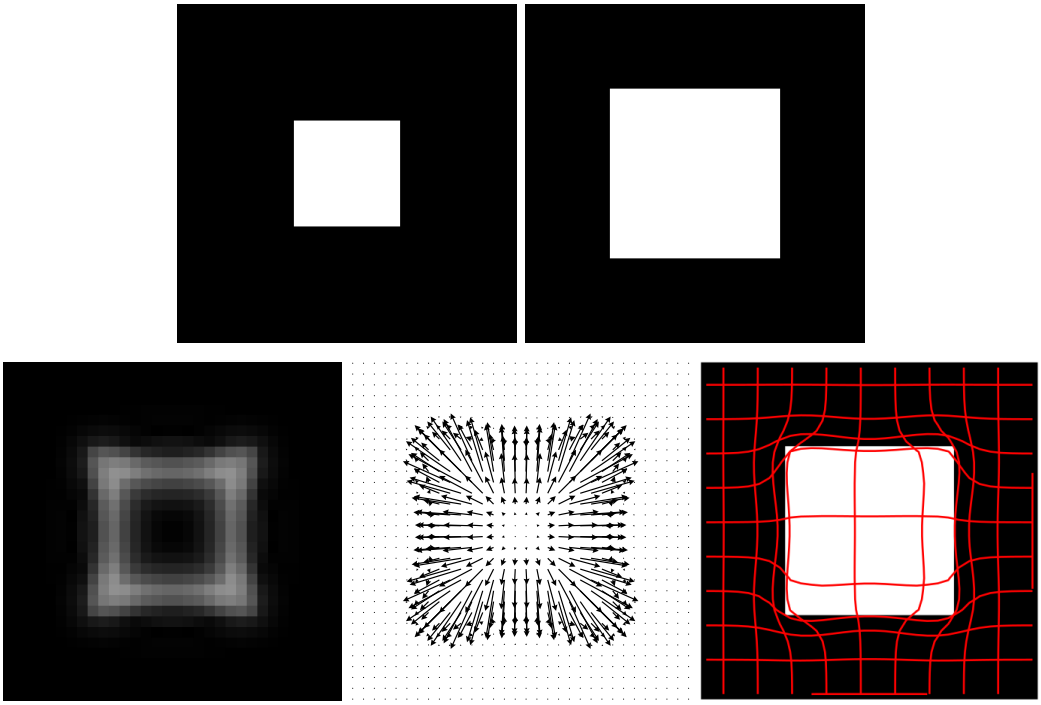


Abbildung 5: Ergebnis der SVF-Registrierung mittels numerischer Berechnungen für Bilder der Größe 32×32 Pixel. Die Rechenzeit betrug 10,42 Sekunden. Dies ist dreimal so schnell wie die LDDMM-Registrierung derselben Bilder (vgl. Abb. 2). **Oben:** Templatebild (links) und Referenzbild (rechts). **Unten:** Betrag des Momentums (links), Vektorfeld des Momentums (Mitte) und das daraus generierte transformierte Templatebild mit Deformationsfeld (rechts).

Noch schnellere Rechenzeiten werden von Methoden des maschinellen Lernens für eine Bildregistrierung erzielt. Auf die Grundlagen gehen wir im nächsten Abschnitt ein.

2.3 Maschinelles Lernen

Im folgenden Abschnitt werden die Grundlagen des maschinellen Lernens mithilfe künstlicher neuronaler Netze für den Einsatz in der Bildverarbeitung erläutert.

2.3.1 Künstliche Neuronale Netze

Die ursprüngliche Idee von *künstlichen neuronalen Netzen* (engl. artificial neural network (ANN)) war es, die Funktionen des menschlichen Gehirns mathematisch zu modellieren. Frühe Modelle wie das Perzeptron [Ros58] legten den Grundstein für die folgende Erforschung ANN-basierter Methoden des maschinellen Lernens (ML) für zahlreiche Anwendungen wie zum Beispiel Mustererkennung und Bilderkennung.

In Anlehnung an das Gehirn besteht ein neuronales Netz aus einer bestimmten Anzahl von sogenannten Neuronen, welche in verschiedenen Schichten angeordnet sind. In seiner Grundform besteht das neuronale Netz aus einer Eingabeschicht mit n Neuronen $x_i,\ i \in \{1, \dots, n\}$, und einem Ausgabeneuron y. Die zugehörigen Gewichte w_i beschreiben jeweils den Einfluss des Neurons x_i auf das Ausgabeneuron y und verbinden auf diese Weise die Eingabeneuronen mit dem Ausgabeneuron. Abbildung 6 (links) zeigt ein Beispiel für ein neuronales Netz in seiner Grundform. Je größer das Gewicht eines Neurons ist, desto stärker ist sein Einfluss auf die Ausgabe, die sich durch Auswertung einer *Aktivierungsfunktion* σ für die gewichtete Summe der Eingaben durch

$$y = \sigma\left(\sum_{i=1}^{n} w_i \cdot x_i + b_i\right) \tag{2.23}$$

berechnet. Um auch nichtlineare Zusammenhänge modellieren zu können, wird meist eine nichtlineare Aktivierungsfunktion σ gewählt, beispielsweise die Sigmoidfunktion. Der negative Schwellwert der Aktivierungsfunktion (auch Bias) wird mit b_i bezeichnet. Dieser Zusammenhang kann allgemein durch die Funktion $f\colon \Theta \times X \to Y$ ausgedrückt werden, die für gegebene Gewichte $\mathbf{w} \in \Theta$ und Eingaben $\mathbf{x} \in X$ die Ausgabe $y \in Y$ bestimmt.

Damit ein Netz bei Bedarf auch in der Lage ist mehrere Werte auszugeben, kann y auch mehrdimensional konstruiert werden. Zusätzlich kann ein Netz neben der Ein- und Ausgabeschicht noch weitere Zwischenschichten enthalten, die zu einer höheren Komplexität des Netzes führen. Enthält ein Netz eine Vielzahl (»10) solcher Zwischenschichten, wird es als *tiefes neuronales Netz* (engl. deep neural network) bezeichnet. Wenn jedes Neuron einer Schicht mit allen Neuronen der folgenden Schicht verbunden ist, wird das Netz als *vollständig verbunden* (engl. fully connected (fc)) bezeichnet. Abbildung 6 (rechts) zeigt ein vollständig verbundenes neuronales Netz mit einer Zwischenschicht.

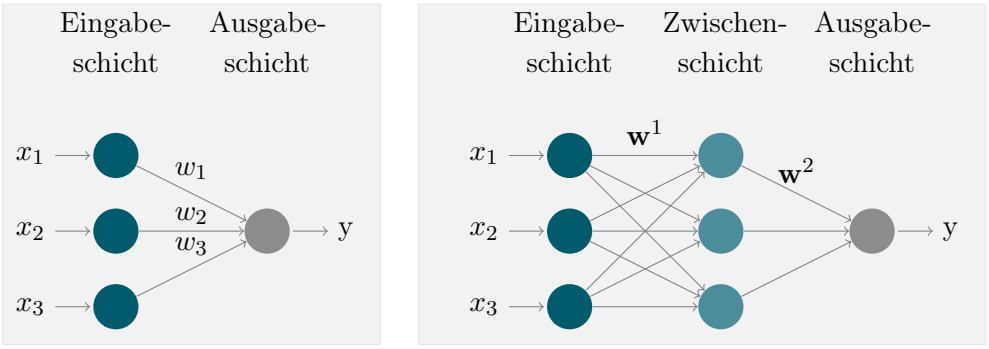


Abbildung 6: Links: Beispiel eines neuronalen Netzes in seiner Grundform mit $n = 3$ Eingabeneuronen x_i, Gewichten w_i und dem Ausgabeneuron y. **Rechts:** Beispiel eines vollständig verbundenen neuronalen Netzes mit einer Zwischenschicht. Die Eingabeneuronen x_i sind hier über die Gewichte $\mathbf{w^1}$ mit allen Neuronen der Zwischenschicht verbunden, die wiederum über die Gewichte $\mathbf{w^2}$ mit dem Ausgabeneuron y verbunden sind.

Das Lernen eines Netzes besteht aus der Anpassung der Gewichte. Hierzu wird die Netz Ausgabe $\mathbf{y}$ für Eingabe $\mathbf{x}$ mit einem bekanntem *Referenzwert* (auch Grundwahrheit) $\mathbf{y}^\star$ in einer *Verlustfunktion* (auch Lossfunktion) $L\colon Y \times Y \to [0, \ldots, \infty)$ verglichen. Je ähnlicher sich der Ausgabe- und Referenzwert sind, desto kleiner wird der *Verlust* (auch Loss).

Daher wird die Lossfunktion

$$L(\mathbf{y}, \mathbf{y}^\star) = L(f(\mathbf{w}, \mathbf{x}), \mathbf{y}^\star) \to \min_{\mathbf{w} \in \Theta} \tag{2.24}$$

unter Verwendung eines Optimierungsverfahrens minimiert. Anschließend wird der Loss zurück propagiert, die sogenannte *Backpropagation*, und die Gewichte auf diese Weise angepasst.

Die in dieser Arbeit vorgestellte Methode verwendet die L1-Norm Lossfunktion

$$L^{\mathrm{L1}}(\mathbf{y}, \mathbf{y}^\star) = \sum_i |y_i - y_i^\star|, \tag{2.25}$$

da sie verhältnismäßig robust gegenüber Ausreißern ist. Die Optimierung wird in dieser Arbeit mit dem Adam-Verfahren [KB14] durchgeführt. Dieses ist in die Familie der stochastischen Gradientenabstiegsverfahren einzuordnen und zeichnet sich dadurch aus, dass es die Schrittweite für das Update der Parameter adaptiv anpasst [Rud16]. Alternative Verfahren sind beispielsweise das AdaGrad-Verfahren [DHS11] und das ASGD-Verfahren [PJ92].

2.3.2 Neuronale Faltungsnetze

Eine spezielle Art von künstlichen neuronalen Netzen sind die neuronalen Faltungsnetzwerke (engl. convolutional neural networks (CNNs)). Beginnend mit der Entwicklung der Methode im Jahr 1989 von Y. LeCun [L+89], wurden sie in den vergangenen Jahrzehnten weiter erforscht. Sie haben sich inzwischen auch für den Einsatz in der Bildverarbeitung etabliert, zum Beispiel für die Mustererkennung und Bildklassifikation [KSH12, HZRS16]. Für diese Anwendungsbereiche ist es wichtig, gleiche Merkmale in zwei Bildern auch dann detektieren zu können, wenn sie sich an unterschiedlichen Positionen befinden. Hierfür werden statt eines vollständig verbundenen neuronalen Netzes, welches n Eingabeneuronen mit l Neuronen der Zwischenschicht über die Gewichtsmatrix $W^{\text{fc}} \in \mathbb{R}^{l \times n}$ verbindet, sogenannte Faltungschichten verwendet, deren lernbare Gewichte die Faltungskerne $W^{\text{conv}} \in \mathbb{R}^{p \times q}$ verschiedener Faltungen sind. Diese Faltungskerne werden mit der Eingabe als Ganzes gefaltet

$$v = \mathbf{x} * W^{\text{conv}} \tag{2.26}$$

und erzeugen eine zweidimensionale Merkmalskarte v. Das Falten mit verschiedenen Faltungskernen führt zu verschiedenen Merkmalskarten, die jeweils einen Kanal in der dreidimensionalen Ausgabe der Faltungsschicht darstellen, welche wiederum als Eingabe für die nächste Faltungsschicht dient. Damit jede Merkmalskarte die gleiche Größe wie die Eingabe hat, wird die Eingabe in jede Richtung um die halbe Faltungskerngröße erweitert und der Dirichlet-Randbedingung entsprechend mit Nullen fortgesetzt. Dies wird als *Padding* bezeichnet. Innerhalb einer Schicht werden die Kerne geteilt, was die Anzahl der lernbaren Parameter und somit auch den Speicherbedarf erheblich senkt. Da die Größe der Filterkerne $p \times q$ geringer ist als die der Eingabe $l \times n$, können lokale Merkmale wie Kanten detektiert werden. Die Kombination mehrerer Faltungsschichten ermöglicht es jedoch auch, Korrespondenzen zwischen weiter entfernten Pixeln zu detektieren und auf diese Weise komplexere Strukturen zu extrahieren [GBC16].

Ein Faltungsnetz besteht typischerweise aus einer Abfolge von Schichten, die verschiedene Funktionen ausüben. Zwei für diese Arbeit relevante Schichten sind die im Folgenden erläuterten PReLU- und Pooling-Schichten. Ein Block in dem Netzwerk, das wir in Kapitel 3.1 vorstellen, besteht jeweils aus der Kombination von Faltungs- und PReLU-Schichten mit einer anschließenden Pooling-Schicht.

PReLU

Eine Kombination von Faltungen kann aufgrund der Linearität von Faltungen nur lineare Funktionen abbilden. Daher wird in der vorgestellten Methode durch die *Parametric Rectified Linear Unit* (PReLU) [HZRS15] eine nichtlineare Aktivierungs-

funktion benutzt, die dem Netz eine zusätzliche Nichtlinearität und folglich eine höhere Komplexität verleiht. Sie hat die Form

$$\mathrm{PReLU}(x) = \begin{cases} x, & \text{für } x > 0, \\ ax, & \text{sonst}, \end{cases} \tag{2.27}$$

wobei a ein Parameter ist, der während des Training des Faltungsnetzes gelernt wird. PReLU ist eine Erweiterung der ReLU Aktivierungsfunktion [NH10] und hat den Vorteil, dass sie nicht nur negative Eingaben löscht, sondern Nullgradienten für negative Eingaben vermeidet, was zu einer Verbesserung der Leistung des Netzes führt.

Pooling

Zur Verminderung des Rechenaufwands und des Speicherbedarfs sowie zur Beschleunigung des Netzes werden in dieser Arbeit sogenannte *Pooling*-Schichten eingesetzt. Die Funktion einer Pooling-Schicht ist es, semantisch ähnliche Merkmale zu einem zu vereinen und so die Größe der Merkmalskarte zu verringern. Eine typische Pooling-Einheit berechnet das Maximum, das Minimum oder den Durchschnitt eines lokalen Patches von Einheiten in einer Merkmalskarte bzw. mehreren Merkmalskarten. Dies ermöglicht dem Netz zusätzlich in tieferen Schichten die Informationen aus Pixeln zu verknüpfen, deren Distanz in den Eingabebildern größer als die Größe des Filterkerns ist. Die Pooling-Schicht ist lokal translationsinvariant. [LBH15]

2.3.3 Software und Hardware

Die Implementierung der in dieser Arbeit vorgestellten Methode wurde in Python 3.6 unter Verwendung der Open-Source-Bibliothek PyTorch 0.4.0. umgesetzt. Diese ermöglicht die effiziente Implementierung von neuronalen Netzen und beschleunigt die Berechnungen durch die Verwendung des Grafikprozessors. Hierfür wurde eine NVIDIA GTX 1080 Ti mit 11 Gigabyte Grafikspeicher verwendet. Als Prozessor stand ein Intel Core i7-4790K zur Verfügung. Für die Implementierung wurde die Mermaid-Toolbox (iMagE Registration via autoMAtIc Differentiation) von Marc Niethammer und der BIAG (biomedical image analysis group) Arbeitsgruppe der University of North Carolina in Chapel Hill verwendet[1]. Sie enthält eine Auswahl an LDDMM und SVF Registrierungsmodellen sowie verschiedene Optimierungsverfahren und kann als Baukasten verwendet werden. Zusätzlich wurde die Python-Implementierung des 2D Quicksilver tiefen Enkoder-Dekoder-Netzes [YKSN17] von Marc Niethammer für diese Arbeit zur Verfügung gestellt. Für den Vergleich der vorgestellten Methode mit dem FlowNet wurde die Python-Implementierung des FlowNetC von Daniel Budelmann, die im Rahmen seiner Masterarbeit [Bud19] erstellt wurde, genutzt.

[1] https://github.com/uncbiag/registration

3 Maschinelles Lernen für Momentum-basierte Bildregistrierung

In diesem Kapitel stellen wir eine neue Methode des maschinellen Lernens für Momentum-basierte Bildregistrierung vor. Hierfür wird zunächst der Methodenaufbau in Abschnitt 3.1 erläutert. Der Unterschied zur Quicksilver-Methode [YKSN17], auf der die vorgestellte Methode basiert, wird anschließend in Abschnitt 3.2 aufgezeigt.

3.1 Methodenaufbau

Wir stellen eine neue Methode des maschinellen Lernens vor, die die Transformationsparameter der Momentum-basierten Bildregistrierung unter Verwendung von Bildinformationen effizient voraussagt. Die Methode besteht aus einem zweischrittigen Framework (im Folgenden als *Gesamtframework* bezeichnet), welches das Momentum m für das SVF-Registrierungsmodell voraussagt. Dies nutzt die Vorteile des SVF-Registrierungsmodells:

- Es kann große Verformungen erfassen.
- Es resultiert in diffeomorphen Transformationen.
- Es hat einen geringeren Rechenaufwand als das LDDMM-Modell mit zeitabhängigen Geschwindigkeitsfeldern.
- Es nutzt das Momentum als Registrierungsparameter, aus dem die gesuchte Transformation berechnet werden kann.

Die Voraussage des Momentums ist zudem geschickt, da es folgende Vorteile hat:

- Das Momentum hat typischerweise einen kompakten Träger rund um Bildkanten, wodurch in konstanten Bildbereichen keine Mehrdeutigkeiten entstehen [YKSN17].
- Es gibt keine Anforderungen an die Differenzierbarkeit des Momentums selbst. Stattdessen werden im SVF-Modell glatte Geschwindigkeitsfelder aus dem Momentum durch anschließende Faltung mit einem Differentialglättungsoperator generiert.
  Dadurch kann das Momentum auch *patchweise* vorausgesagt werden und das gesamte Momentum aus den vorausgesagten Momentumpatches zusammengesetzt werden. Dies führt zu einer einfach parallelisierbaren Voraussage und erlaubt es, ein Netz mit einer relativ kleinen Anzahl an Bildern zu trainieren.

Durch die patchweise Voraussage der Momenta behalten wir somit alle komfortablen mathematischen Eigenschaften von SVF bei und sind gleichzeitig in der Lage,

diffeomorphe Transformationen schnell vorherzusagen. Der Aufbau des hierfür verwendeten Gesamtframeworks wird im Folgenden näher beschrieben.

Die Vorhersage der Momentumpatches geschieht mit einem tiefen Faltungsnetz, das als *Prediction Net* bezeichnet wird. Auf die Form des Prediction Nets werden wir in Abschnitt 3.1.3 näher eingehen. Zunächst müssen Grundwahrheitsdaten für die Trainingsmenge generiert werden, mit denen das Prediction Net trainiert wird.

3.1.1 Erzeugung der Grundwahrheit

Für die Erzeugung der Grundwahrheit werden die Bildpaare der Trainingsmenge, bestehend aus Template- und Referenzbild, durch numerische Optimierung des SVF-Modells (vgl. Kapitel 2.2.1) registriert. Die hierbei erhaltenen Momenta m_{GP} dienen als Grundwahrheitsmomenta für das Training des Prediction Nets.

3.1.2 Patchauswahl

Um das Momentum patchweise voraussagen zu können, werden die Bildpaare der Trainingsmenge und ihre Grundwahrheitsmomenta zunächst in eine Vielzahl von korrespondierenden Patchpaaren mit zugehörigen Momentumpatches aufgeteilt. Dazu wird ein Ansatz mit gleitendem Fenster verwendet. Dieser schneidet an denselben Positionen im Template- und Referenzbild und im Grundwahrheitsmomentum einen durch die Patchgröße festgelegten Bildausschnitt aus und erzeugt so korrespondierende Patchpaare und Momentumpatches. Die Positionen sind dabei um eine festgelegte Schrittweite versetzt.

Bei der patchweisen Voraussage ist die Rechenzeit proportional zu der Anzahl der Patches, die vorausgesagt werden müssen. Wenn eine 1-Pixel Schrittweite für das gleitende Fenster verwendet würde, könnte die Anzahl der vorauszusagenden Patches für ein ganzes Bild beträchtlich sein. Für ein 2D-Bild der Größe 100×160 Pixel müssten für eine Patchgröße von 15×15 Pixeln mehr als 12,000 Patches vorausgesagt werden. Daher werden zwei von [YKSN17] vorgeschlagene Strategien zur Patchreduktion verwendet:

1. Patches, die sowohl im Template- als auch im Referenzbild komplett im Hintergrund liegen (alle Bildintensitäten sind Null), werden ignoriert. Dies wird dadurch gerechtfertigt, dass das Momentum der LDDMM-Theorie zufolge in konstanten Bildbereichen gleich Null ist.

2. Es wird bei der Patchauswahl eine große Schrittweite für das gleitende Fenster benutzt; zum Beispiel eine 14 Pixel Schrittweite für 15×15 Pixel große Patches. Dies ist aufgrund des kompakten Trägers des Momentums und der Translationsinvarianz, die durch die Pooling-Operationen erlangt wird, vertretbar.

Mit Hilfe dieser Patchreduktionsstrategien kann die Anzahl der vorauszusagenden Patches und folglich der Rechenaufwand der Methode drastisch reduziert werden.

3.1.3 Prediction Net

Unser Ziel ist es, eine Voraussagefunktion zu lernen, die zwei Eingabepatches erhält, die an derselben Position vom Template- und Referenzbild entnommen wurden, und den gesuchten vektorwertigen Momentumpatch für die x- und y-Dimension jeweils separat voraussagt. Für Patches der Größe $p \times p$ Pixel, soll also die Voraussagefunktion $f\colon \mathbb{R}^{p^2} \times \mathbb{R}^{p^2} \to \mathbb{R}^{2p^2}$ gelernt werden.
In der vorgestellten Methode wird diese Voraussagefunktion durch ein *Prediction Net*, ein tiefes CNN mit L1-Norm Lossfunktion, implementiert. In Abbildung 7 ist die Architektur des Prediction Nets skizziert, die aus einem *Enkoder* und einem *Dekoder* besteht. Die Architektur ist aus [YKSN17] übernommen und wird im Folgenden näher beschrieben.

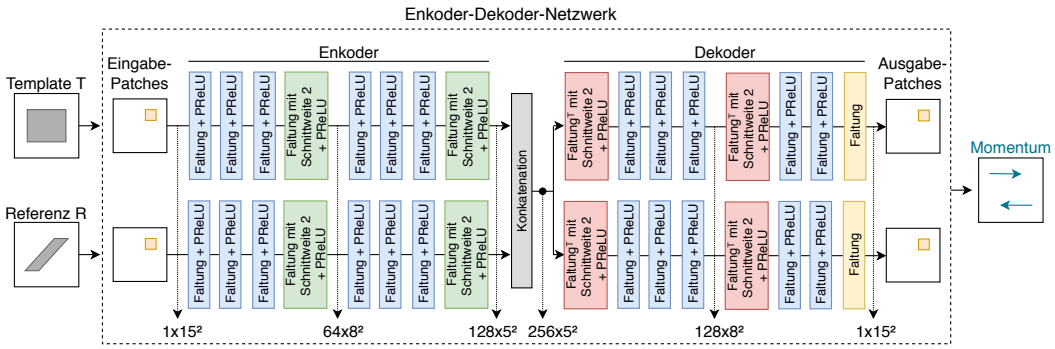


Abbildung 7: 2D Enkoder-Dekoder-Netzwerkarchitektur. Das Netzwerk erhält zwei 2D-Patches vom Template- und Referenzbild als Eingabe und gibt einen vektorwertigen 2D Momentumpatch (für x- und y-Richtung getrennt) aus. Alle Faltungsschichten arbeiten mit 1-Pixel Padding in jede Richtung.
15×15 Pixel Patches, die an derselben Position extrahiert wurden.
2D-Faltungsschicht mit 3×3 Kern, Schrittweite 1 und PReLU-Schicht.
2D-Faltungsschicht mit 2×2 Kern, Schrittweite 2 und PReLU-Schicht.
2D-Faltungsschicht mit 3×3 Kern und Schrittweite 1, kein PReLU.
Transponierte 2D-Faltungsschicht mit 2×2 Kern, Schrittweite 2 und PReLU-Schicht.

Enkoder

Der Enkoderteil besteht aus zwei parallelen Enkodern, die die Merkmale der Eingabepatches aus Template- und Referenzbild unabhängig voneinander lernen. Jeder Enkoder enthält zwei Blöcke, die aus drei 2D-Faltungsschichten mit Faltungskernen der Größe 3×3 Pixel und anschließender PReLU-Schicht bestehen. Jeder Block endet mit einer zusätzlichen 2D-Faltungsschicht mit Faltungskern der Größe 2×2 Pixel und anschließender PReLU-Schicht. Die Faltung wird in dieser Schicht mit Schrittweite 2 ausgeführt, was dazu führt, dass die Größe des Ausgabepatches verringert wird.

Die Anzahl der Merkmale im ersten Block beträgt 64 und erhöht sich auf 128 im zweiten Block. Die gelernten Merkmale der zwei Enkoder werden schließlich konkateniert und an zwei parallele Dekoder übergeben.

Dekoder

Die Struktur eines jeden Dekoders ist invers zu der Struktur der Enkoder. Jedoch ist die Anzahl der Merkmale doppelt so hoch – 256 Merkmale im ersten und 128 Merkmale im zweiten Block – da die Eingabe des Dekoders durch die Konkatenation der Ausgabe beider Enkoder erzielt wird. Zudem wird durch den Einsatz von transponierten Faltungsschichten mit einer Schrittweite 2 ein „Unpooling“ durchgeführt, das dem Inversen der Pooling-Operation des Enkoders entspricht. Außerdem wird die PReLU-Schicht nach der letzten Faltungsschicht weggelassen.
Das Nutzen von Faltungen und transponierten Faltungen zum Lernen der Pooling- bzw. Unpooling-Operation ist für das hier vorgestellte Netzwerk gut geeignet, da die zwei Enkoder das Pooling unabhängig voneinander durchführen. [YKSN17, LSD15]

Die Ausgaben der Dekoder sind der Momentumpatch in x- und y-Richtung. Das gesamte vorausgesagte Momentum wird nun aus allen vorausgesagten Momentumpatches zusammengesetzt. Überlappende Bereiche werden hierfür gemittelt. Die finale Ausgabe des Prediction Nets ist das gesamte vorausgesagte Momentum m_p.

Das Prediction Net hat zwei Schwächen:

1. Es ersetzt den gesamten iterativen numerischen Ansatz durch einen einzigen Inferenzschritt. Es sind daher zunächst ungenauere Ergebnisse zu erwarten.

2. Da Patchpaare an denselben Positionen vom Template- und Referenzbild entnommen werden, können in Fällen von großen Deformationen große Unterschiede zwischen den Patches der Bilder auftreten. Dies kann im Extremfall dazu führen, dass keinerlei korrespondierende Bildinformationen mehr gefunden werden können.

Daher wird ein zweischrittiger Voraussage-Ansatz gewählt, um diese Schwächen zu kompensieren und die Voraussage-Genauigkeit zu verbessern. Der erste Schritt ist das zuvor beschriebene Prediction Net. Der zweite Schritt ist ein sogenanntes *Correction Net*, auf das wir im Folgenden näher eingehen.

3.1.4 Correction Net

Die Aufgabe des *Correction Nets* ist, die Voraussage-Fehler des Prediction Net zu korrigieren. Dafür wird ausgenutzt, dass nach der SVF-Theorie die Transformation ϕ^{-1}, die das Templatebild in den Raum des Referenzbildes abbildet, und ihre Inverse ϕ, die das Referenzbild in den Raum des Templatebildes abbildet, aus dem vorausgesagten Momentum m_p generiert werden können. Nach dem ersten Voraussage-Schritt durch das Prediction Net wird das Referenzbild mittels $\mathcal{R} \circ \phi$ zurück deformiert.

Dann kann das Correction Net mit den Unterschieden zwischen $\mathcal{T}$ und $\mathcal{R} \circ \phi$ trainiert werden, um eine Momentumkorrektur m_c für das vom Prediction Net vorausgesagte Momentum m_p vorauszusagen. Dazu werden wieder Grundwahrheitsdaten für die Trainingsmenge generiert, indem die Bildpaare $(\mathcal{T}, \mathcal{R} \circ \phi)$ durch numerische Optimierung des SVF-Modells registriert werden. Die erhaltenen Momenta m_{GC} dienen als Grundwahrheitsmomenta für das Training des Correction Nets.

Das Correction Net hat dieselbe Architektur wie das Prediction Net (siehe Abbildung 7). Als Eingabe erhält es die Bildpaare $(\mathcal{T}, \mathcal{R} \circ \phi)$ und die korrespondierenden Grundwahrheitsmomenta m_{GC}, aus denen Patchpaare mit korrespondierenden Momentumpatches extrahiert werden. Als Ausgabe liefert es die Momentumkorrektur m_c, die analog zum vorausgesagten Momentum des Prediction Nets aus den einzelnen Momentumkorrekturpatches zusammengesetzt wird.
Da $\mathcal{T}$ und $\mathcal{R} \circ \phi$ in demselben Koordinatensystem liegen, sind die Unterschiede zwischen ihnen klein, solange die vorausgesagte Transformation aus dem Prediction Net angemessen ist. Dies lässt präzisere Voraussagen erwarten.

Zudem wird die Korrektur des vorausgesagten Momentums in dessen Koordinatensystem durchgeführt. Dies ermöglicht es, das finale vorausgesagte Momentum $\widehat{m}$ durch die Addition des vorausgesagten Momentums m_p und der Momentumkorrektur m_c zu generieren:

$$\widehat{m} = m_p + m_c. \tag{3.1}$$

Das finale vorausgesagte Momentum ist die Ausgabe des gesamten Prediction Frameworks. Die Gesamtstruktur wird im nächsten Abschnitt noch einmal zusammengefasst.

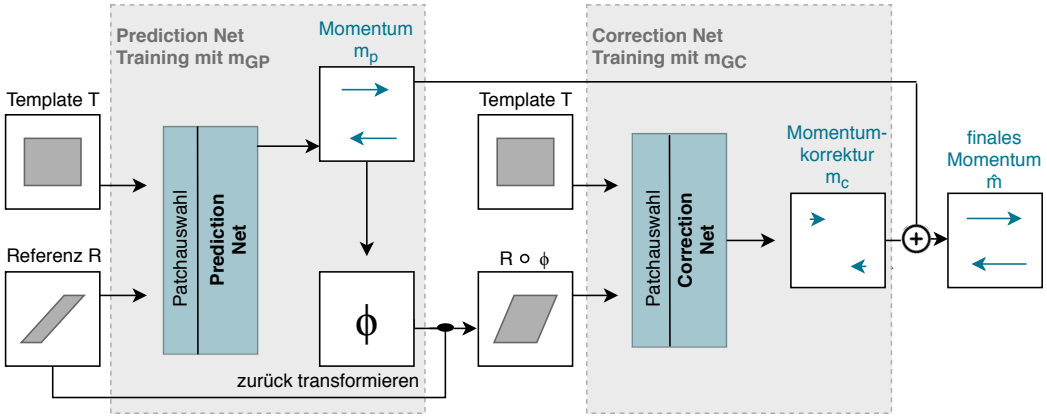


Abbildung 8: Gesamtstruktur des zweischrittigen Gesamtframeworks. Zunächst wird ein grobes Momentum m_p durch das Prediction Net für das Bildpaar $(\mathcal{T}, \mathcal{R})$ patchweise vorausgesagt. Daraus werden die Transformationen ϕ^{-1} und ϕ generiert, sodass das Referenzbild zurück in den Raum des Templatebildes deformiert werden kann. Dann wird eine Momentumkorrektur m_c im Raum des Templatebildes durch das Correction Net für das Bildpaar $(\mathcal{T}, \mathcal{R} \circ \phi^{-1})$ patchweise vorausgesagt. Schließlich wird das finale Momentum $\widehat{m} = m_p + m_c$ als Summe des vorausgesagten Momentums und der Momentumkorrektur berechnet.

3.1.5 Struktur des Gesamtframeworks

Das resultierende zweischrittige Gesamtframework zur Voraussage des Momentums ist in Abbildung 8 dargestellt. Das Correction Net hat dieselbe Struktur wie das Prediction Net. Der Unterschied liegt lediglich in der Eingabe der Netze und den von den Netzen produzierten Ausgaben.

Das Training des Gesamtframeworks wird sequentiell durchgeführt:

1. Generierung der Grundwahrheitsmomenta m_{GP} für das Prediction Net durch numerische Optimierung des SVF-Registrierungsmodells für Trainingsbildpaare $(\mathcal{T}, \mathcal{R})$.

2. Training des Prediction Nets mit Trainingsbildpaaren $(\mathcal{T}, \mathcal{R})$ und Grundwahrheitsmomenta m_{GP}.

3. Aus den vorausgesagten Momenta des Prediction Nets die zugehörigen Transformationen ϕ^{-1}, ϕ generieren, um die Referenzbilder der Trainingsmenge zurück in den Raum der Templatebilder zu deformieren $\mathcal{R} \circ \phi$.

4. Generierung der Grundwahrheitsmomenta m_{GC} für das Correction Net durch numerische Optimierung des SVF-Registrierungsmodells für Trainingsbildpaare $(\mathcal{T}, \mathcal{R} \circ \phi)$.

5. Training des Correction Nets mit Trainingsbildpaaren $(\mathcal{T}, \mathcal{R} \circ \phi)$ und Grundwahrheitsmomenta m_{GC}.

Die Auswertung des Gesamtframeworks verläuft ähnlich zum Training. Die Registrierungsschritte 1.) und 4.) fallen hierbei jedoch weg, da die fertig trainierten Netze für die Auswertung keine Grundwahrheitsmomenta mehr benötigen. Der sequentielle Ablauf der Auswertung hat die Form:

1. Vorausgesagte Momenta m_p durch Auswertung des Prediction Nets für die Testbildpaare $\mathcal{T}, \mathcal{R}$ generieren.

2. Aus den vorausgesagten Momenta des Prediction Nets die zugehörigen Transformationen ϕ^{-1}, ϕ generieren, um die Referenzbilder der Testmenge zurück in den Raum der Templatebilder zu deformieren: $\mathcal{R} \circ \phi$.

3. Momentumkorrekturen m_c durch Auswertung des Correction Nets für die Testbildpaare $(\mathcal{T}, \mathcal{R} \circ \phi)$ generieren.

4. Addition der vorausgesagten Momenta m_p mit den zugehörigen Momentumkorrekturen m_c liefert die finalen vorausgesagten Momenta $\widehat{m} = m_p + m_c$.

3.2 Unterschied zur Quicksilver-Methode

Die vorgestellte Methode zur patchweisen Voraussage des Momentums mit einem zweischrittigen Gesamtframework unterscheidet sich in zwei Punkten von der Quicksilver-Methode [YKSN17], auf der sie basiert.

1. Während die vorgestellte Methode das stationäre Momentum m der SVF-Parametrisierung des LDDMM-Modells voraussagt, sagt die Quicksilver-Methode von [YKSN17] das initiale Momentum m_0 der Shooting-Formulierung des LDDMM-Modells mit zeitabhängigem Momentum voraus. Dies bedeutet einen höheren Rechenaufwand während der Generierung der Grundwahrheitsdaten und während der Generierung der Transformationen ϕ und ϕ^{-1} aus den vorausgesagten Momenta.

2. Die Quicksilver-Methode benutzt zur Generierung der Grundwahrheitsmomenta m_{GC} für das Training des Correction Nets (Punkt 4) nicht die numerische Optimierung des Registrierungsmodells für die Bildpaare $(\mathcal{T}, \mathcal{R} \circ \phi)$. Stattdessen werden Grundwahrheitsmomenta für das Correction Net m_{GC} durch die Differenzen zwischen den Grundwahrheitsmomenta des Prediction Nets m_{GP} und den vom Prediction Net vorausgesagten Momenta m_p generiert:

$$m_{\mathrm{GC}} = m_{\mathrm{GP}} - m_p. \tag{3.2}$$

Hierin ist die anschließende Addition $\widehat{m} = m_p + m_c$ begründet.

Eine Veranschaulichung der Unterschiede zeigt Abbildung 9 . Die Quicksilver-Methode verfolgt das Ziel, Voraussagen für das Momentum zu generieren, die der Grundwahrheit des Prediction Nets m_{GP} so nah wie möglich kommen. Das Quicksilver Correction Net wird daher so trainiert, dass es die Differenzen zwischen den vom Prediction Net vorausgesagten Momenta m_p und den Grundwahrheitsmomenta m_{GP} verringert.

Die in dieser Arbeit vorgestellte Methode verfolgt hingegen das Ziel, dass das aus dem vorausgesagten Momentum generierte transformierte Templatebild dem Referenzbild so nah wie möglich ist und nutzt die Grundwahrheiten des Prediction Nets nur als erste Näherung. Das Correction Net wird daher so trainiert, dass es die Differenz zwischen dem Templatebild und dem durch das vom Prediction Net vorausgesagte Momentum m_p generierte transformierte Referenzbild weiter verringert. Dadurch lässt sich die Addition $\widehat{m} = m_p + m_c$ in der vorgestellten Methode nicht intuitiv erklären, doch die Experimente im folgenden Kapitel 4 zeigen, dass sie auf den verwendeten Datensätzen zu besseren Ergebnissen führt.

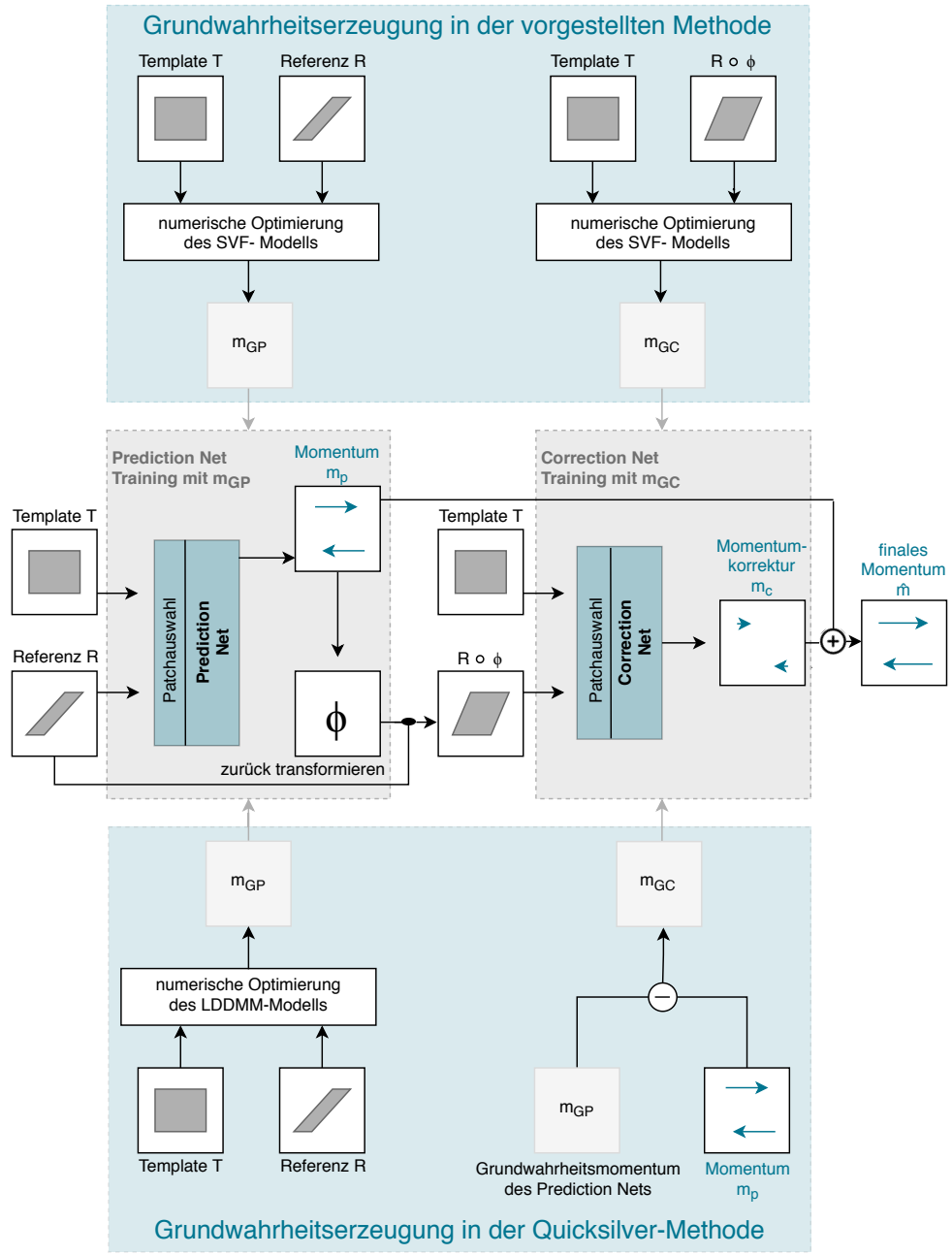


Abbildung 9: Vergleich der Strategien zur Grundwahrheitserzeugung in der vorgestellten Methode (oben) und in der Quicksilver-Methode (unten). Die vorgestellte Methode verwendet die Momenta der SVF-Registrierung von den Bildpaaren $(\mathcal{T}, \mathcal{R})$ als Grundwahrheit für das Prediction Net und die Momenta der SVF-Registrierung von den Bildpaaren $(\mathcal{T}, \mathcal{R} \circ \phi)$ als Grundwahrheit für das Correction Net. Die Quicksilver-Methode verwendet die Momenta der LDDMM-Registrierung von den Bildpaaren $(\mathcal{T}, \mathcal{R})$ als Grundwahrheit für das Prediction Net und die Differenzen $m_{\text{GP}} - m_p$ als Grundwahrheit für das Correction Net.

4 Experimente und Ergebnisse

In diesem Kapitel beschreiben wir die von uns durchgeführten Experimente zur Evaluierung des in Kapitel 3 vorgestellten zweischrittigen Gesamtframeworks. Zunächst werden in Abschnitt 4.1 die für die Experimente verwendeten Datensätze vorgestellt. Abschnitt 4.2 beschreibt die Durchführung und Ergebnisse der Bildregistrierung mit dem vorgestellten Gesamtframework. Im Anschluss vergleichen wir unsere Methode sowohl mit der Quicksilver-Version des Gesamtframeworks für das SVF-Modell als auch mit einer Implementierung des FlowNets von Daniel Budelmann [Bud19]. Das Kapitel schließt mit einer Diskussion der experimentellen Ergebnisse.

4.1 Datensätze

4.1.1 Osteoarthritis Initiative Datensatz

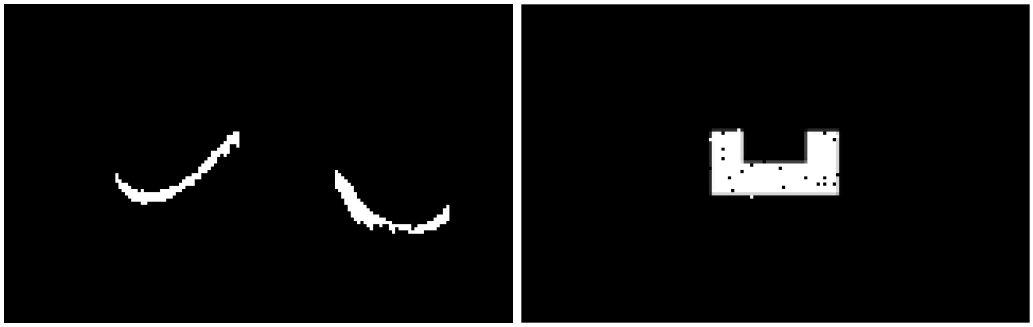
Der verwendete *Osteoarthritis Initiative* (OAI) Datensatz [PSN08] besteht aus 86 Segmentierungen des femoralen Knorpels des linken Knies, die von Ärztinnen und Ärzten in Magnetresonanztomographiebildern annotiert wurden. Die Segmentierungen stammen von 43 Patienten, die zu zwei unterschiedlichen Zeitpunkten untersucht wurden. Die Bildintensität 0 beschreibt den Hintergrund und die Bildintensität 1 den femoralen Knorpel. Die Segmentierungen haben jeweils eine Bildgröße von $384 \times 384 \times 160$ Pixeln, mit einer Pixelgröße von $0.36 \times 0.36 \times 0.7$ mm^3.

Für die Durchführung der Experimente in 2D wurde jeweils die 192. saggitale Schicht entnommen und auf eine Größe von 100×160 Pixel zugeschnitten. Die Aufteilung in Trainings-, Validierungs- und Testmenge erfolgte nach dem Zufallsprinzip:

- Training: 50 Bilder
- Validierung: 18 Bilder
- Test: 18 Bilder

Die Zugehörigkeit zu Patienten wurde hierbei nicht berücksichtigt. Für die Registrierung wurden alle Bilder mit demselben Referenzbild gepaart. Als Referenzbild wurde das Bild 54 gewählt, da es im Mittel den kleinsten quadratischen Abstand zu allen anderen Bildern aufweist. Es ist in Abbildung 10a gezeigt.

In Abbildung 11 sind die anfänglichen $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der Bildpaare des OAI Datensatzes dargestellt. Die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen wurden als Fehlermaß gewählt, da in der folgenden Registrierung das $\mathcal{D}^{\mathrm{NCC}}$-Distanzmaß als Datenterm verwendet wurde (s. Kapitel 4.2.1).



(a) Referenzbild OAI Datensatz **(b)** Referenzbild synthetischer Datensatz

Abbildung 10: Gewählte Referenzbilder der verwendeten Datensätze.

4.1.2 Synthetische Daten

Zusätzlich wurde ein *synthetischer Datensatz* generiert. Hierzu wurde ein Grundbild der Größe 800×1280 Pixel mit einer weißen U-Form (Bildintensität 1) der Größe 160×320 Pixel auf schwarzem Hintergrund (Bildintensität 0) erstellt. Weitere 148 Bilder wurden durch Rotation des Grundbildes um $\pm 5°$ und $\pm 10°$ sowie durch Translation um ± 15, ± 30, ± 45 und ± 60 Pixel in beiden Dimensionen generiert. Anschließend wurden alle Bilder mittels bilinearer Interpolation auf die Größe 100×160 Pixel herunterskaliert und der U-Form-Bereich mit Salt & Pepper Rauschen ($p = 0.05$) belegt.

Die U-Form wurde gewählt, da die Segmentierungen des OAI Datensatzes eine ähnliche Form haben und die U-Form rotationsinvariant ist. Es wurde nur der U-Form-Bereich mit Rauschen belegt, da ein Aussortieren von Patches aus dem Hintergrund sonst nicht mehr möglich ist. Dies würde die Trainingsstrategie des Gesamtframeworks verändern, sodass ein direkter Vergleich der Ergebnisse der vorgestellten Methode für beide Datensätze nicht mehr möglich wäre.
In Abbildung 10b ist das finale Grundbild des synthetischen Datensatzes zu sehen. Es wurde als Referenzbild gewählt und mit allen Bildern zu Bildpaaren für die Registrierung gepaart. Die Aufteilung in Trainings-, Validierungs- und Testmenge erfolgte nach dem Zufallsprinzip:

- Training: 90 Bilder
- Validierung: 30 Bilder
- Test: 29 Bilder

Die anfänglichen $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der Bildpaare des synthetischen Datensatzes sind in Abbildung 11 dargestellt.

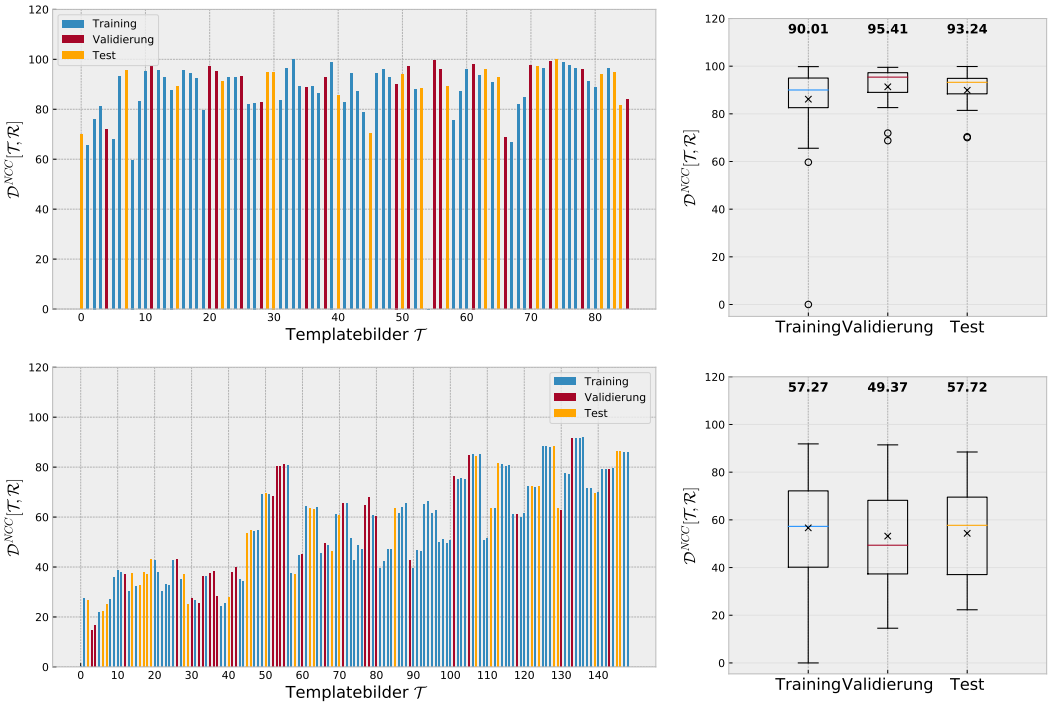


Abbildung 11: Anfängliche $\mathcal{D}^{\text{NCC}}$-Distanzen im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben.

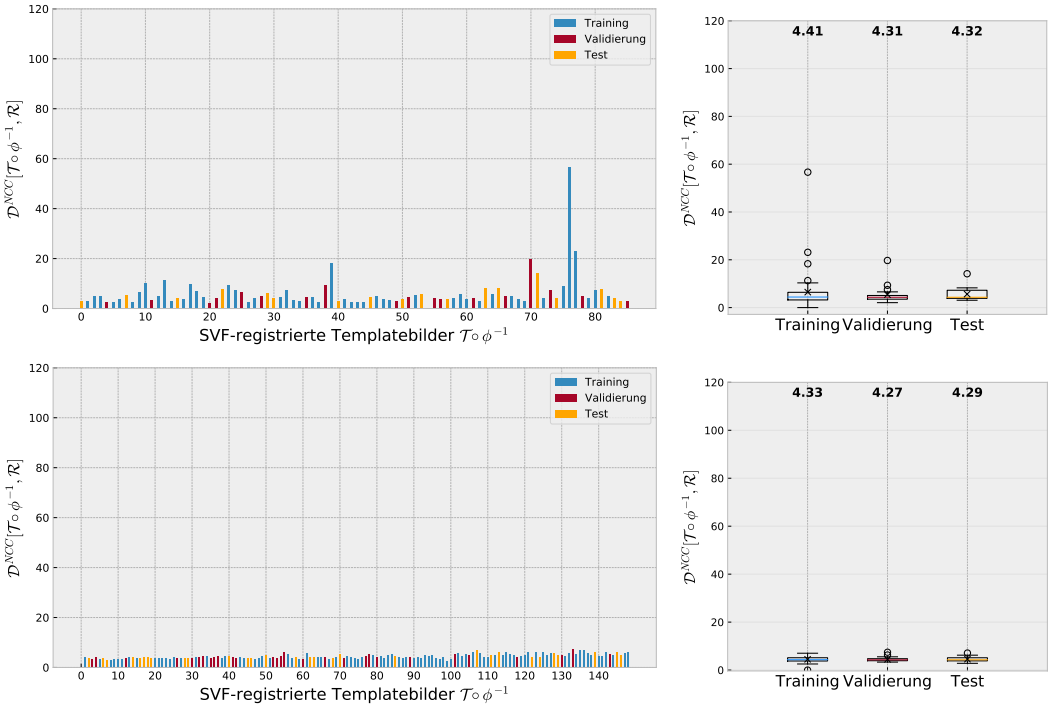


Abbildung 12: $\mathcal{D}^{\text{NCC}}$-Distanzen nach der SVF-Registrierung im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung führt zu einer deutlichen Verringerung des Fehlermaßes. Die Ausreißer für die OAI Bilder 76 und 77 werden durch Löcher im Knieknorpel hervorgerufen (vgl. Abb. 13).

4.2 Bildregistrierung mit dem Gesamtframework

Für die Bildregistrierung mit dem in dieser Arbeit vorgestellten Gesamtframework wurde zunächst eine Grundwahrheit erzeugt und das Gesamtframework mit dieser trainiert. Wir gehen im Folgenden zunächst auf die Parameterwahl für die Erzeugung der Grundwahrheit und danach auf die Parameterwahl für das Training ein, bevor wir anschließend die Ergebnisse der Auswertung des Gesamtframeworks präsentieren.

4.2.1 Erzeugung der Grundwahrheit

Die Erzeugung der Grundwahrheit wurde durch numerische Optimierung des SVF-Modells erzeugt. Dazu wurde die SVF-Registrierung mit $\mathcal{D}^{\mathrm{NCC}}$-Distanzmaß in Python mit der Mermaid-Toolbox in zwei Schritten durchgeführt:

1. Grobe Vorregistrierung des Bildpaares, welches mit einem Gaußfilter mit Standardabweichung 0,01 geglättet wurde. Für den Regularisierer wurde ein multi-Gaußkern G_m mit den Standardabweichungen $[\sigma_1;\ \sigma_2] = [0,02;\ 0,06]$ und den Gewichten $[w_1;\ w_2] = [0,2;\ 0,8]$ gewählt. Der Gewichtungsparameter des Distanzmaßes wurde auf $\sigma = 0,1$ gesetzt. Die Optimierung wurde mit dem Gradientenabstiegsverfahren mit 300 Iterationen und einer Schrittweite von $\alpha = 0,0004$ durchgeführt.

2. Registrierung des ungefilterten Bildpaares mit dem Momentum der groben Vorregistrierung als initialem Momentum. Hierzu wurden für den Regularisierer ein multi-Gaußkern G_m mit den Standardabweichungen $[\sigma_1;\ \sigma_2;\ \sigma_3] = [0,01;\ 0,05;\ 0,1]$ und den Gewichten $[w_1;\ w_2;\ w_3] = [0,2;\ 0,3;\ 0,5]$ gewählt. Der Gewichtungsparameter des Distanzmaßes wurde auf $\sigma = 0,1$ gesetzt. Die Optimierung wurde mit dem Gradientenabstiegsverfahren mit 100 Iterationen und einer Schrittweite von $\alpha = 0,0007$ durchgeführt.

Die durch diese SVF-Registrierung generierten Momenta wurden als Grundwahrheitsmomenta m_{GP} abgespeichert. Die so durchgeführte Registrierung benötigt für den OAI Datensatz im Durchschnitt 55,96 Sekunden pro Bildpaar und für den synthetischen Datensatz im Durchschnitt 61,47 Sekunden pro Bildpaar. Diese Zeiten könnten durch eine effizientere Implementierung, zum Beispiel durch die Parallelisierung der Registrierungen, weiterhin reduziert werden.

Die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der Bildpaare nach der SVF-Registrierung wurden für die Trainings-, Validierungs- und Testmenge berechnet und sind für beide Datensätze in Abbildung 12 dargestellt.
Da der Median im Gegensatz zum Mittelwert robuster gegenüber Ausreißern ist, wird dieser in der Graphik angegeben. Der Median der $\mathcal{D}^{\mathrm{NCC}}$-Distanzen wurde im OAI Datensatz durch die SVF-Registrierung von 90,01 auf 4,41 (Trainingsmenge), von 95,41 auf 4,31 (Validierungsmenge) und von 93,24 auf 4,32 (Testmenge) verringert.

Im synthetischen Datensatz wurde er von von 57,27 auf 4,33 (Trainingsmenge), von 49,37 auf 4,27 (Validierungsmenge) und von 57,72 auf 4,29 (Testmenge) verringert.

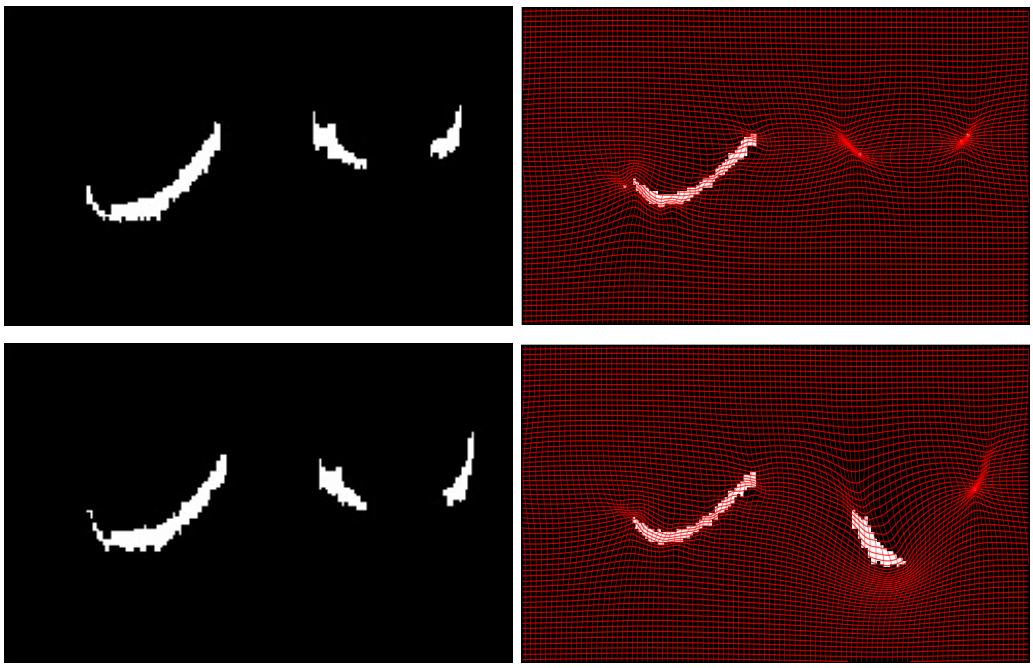


Abbildung 13: Templatebilder des OAI Datensatzes mit den durch SVF-Registrierung generierten Transformationen. **Oben:** Bild 76. **Unten:** Bild 77.

Die Ausreißer für die Bilder 76 und 77 der Trainingsmenge des OAI Datensatzes sind durch die in diesen Bildern auftretenden großen Löcher durch starke Knorpelbeschädigungen bedingt, wie in Abbildung 13 zu sehen ist.

4.2.2 Training des Gesamtframeworks

Patchauswahl

Für das Training des Prediction Net wurden aus den Bildpaaren der Trainings- und Validierungsmenge und den zugehörigen Grundwahrheitsmomenta Patches der Größe 15×15 Pixel mit einer Schrittweite von 14 Pixeln entnommen. Patchpaare des Hintergrundes wurden zur Patchreduktion aussortiert. Dies führte beim OAI Datensatz zu einer Patchreduktion von 795.491 auf 166.923 Patchpaare der Trainingsmenge, beim synthetischen Datensatz zu einer Patchreduktion von 1.434.343 auf 228.419 Patchpaare der Trainingsmenge.

Prediction Net

Das Enkoder-Dekoder-Netzwerk, dessen Implementierung in PyTorch von Marc Niethammer zur Verfügung gestellt wurde, nutzt das Adam-Verfahren zur Optimierung [KB14, YKSN17]. Für die Wahl der Lernrate lr und des Adam weight-decay Parameters wd, wurden die Kombinationen aus $lr \in \{0,01;\ 0,001;\ 0,0001\}$ und $wd \in \{0;\ 0,1;\ 0,5;\ 5\}$ getestet. Eine Übersicht über die Ergebnisse des Prediction Net Trainings mit dem OAI Datensatz ist in Abbildung 14 dargestellt.

Die Kombinationen $(lr = 0,001;\ wd \in \{0;\ 0,1;\ 0,5\})$ und $(lr = 0,0001;\ wd = 0)$ führen zu einem glatten, abfallenden Trainingsloss-Verlauf. Hierbei führt der Adam weight-decay $wd = 0$ zu den kleinsten Trainingsloss-Werten und die Lernrate $lr = 0,0001$ zu dem kleinsten Validierungsloss-Werten. Deshalb wurde $lr = 0,0001$ und $wd = 0$ für das Training des Enkoder-Dekoder-Netzwerkes gewählt.

Der erst abfallende und anschließend wieder leicht steigende Validierungsloss-Verlauf ließen eine Überanpassung (engl. overfitting) vermuten, weshalb zusätzlich der Einsatz eines Dropout-Moduls am Ende eines jeden Enkoder- und Dekoder-Blocks getestet wurde. Das Dropout-Modul setzt Elemente des Eingangstensors mit Wahrscheinlichkeit p (auch Dropoutrate) zufällig auf Null.

In Abbildung 15 sind die Loss-Verläufe für die durchgeführten Prediction Net Trainings mit dem OAI Datensatz unter Verwendung des Dropout-Moduls mit Dropoutraten $p = 0,2$ und $p = 0,5$ dargestellt. Sie zeigen, dass die abfallende und ansteigende Form des Validierungsloss-Verlaufes auch unter Einsatz des Dropout-Moduls erhalten bleibt und der Trainingsloss langsamer abfällt. Deshalb wurde das Dropout-Modul nicht zum Training eingesetzt, also Dropoutrate $p = 0$ gewählt.

Daraufhin wurde zusätzlich getestet, wie gut das Prediction Net generalisieren kann, indem das Netz nach jeder Trainingsepoche auf der Validierungsmenge ausgewertet wurde. Hierbei wurde der durchschnittliche L1-Loss für einen Momentum-

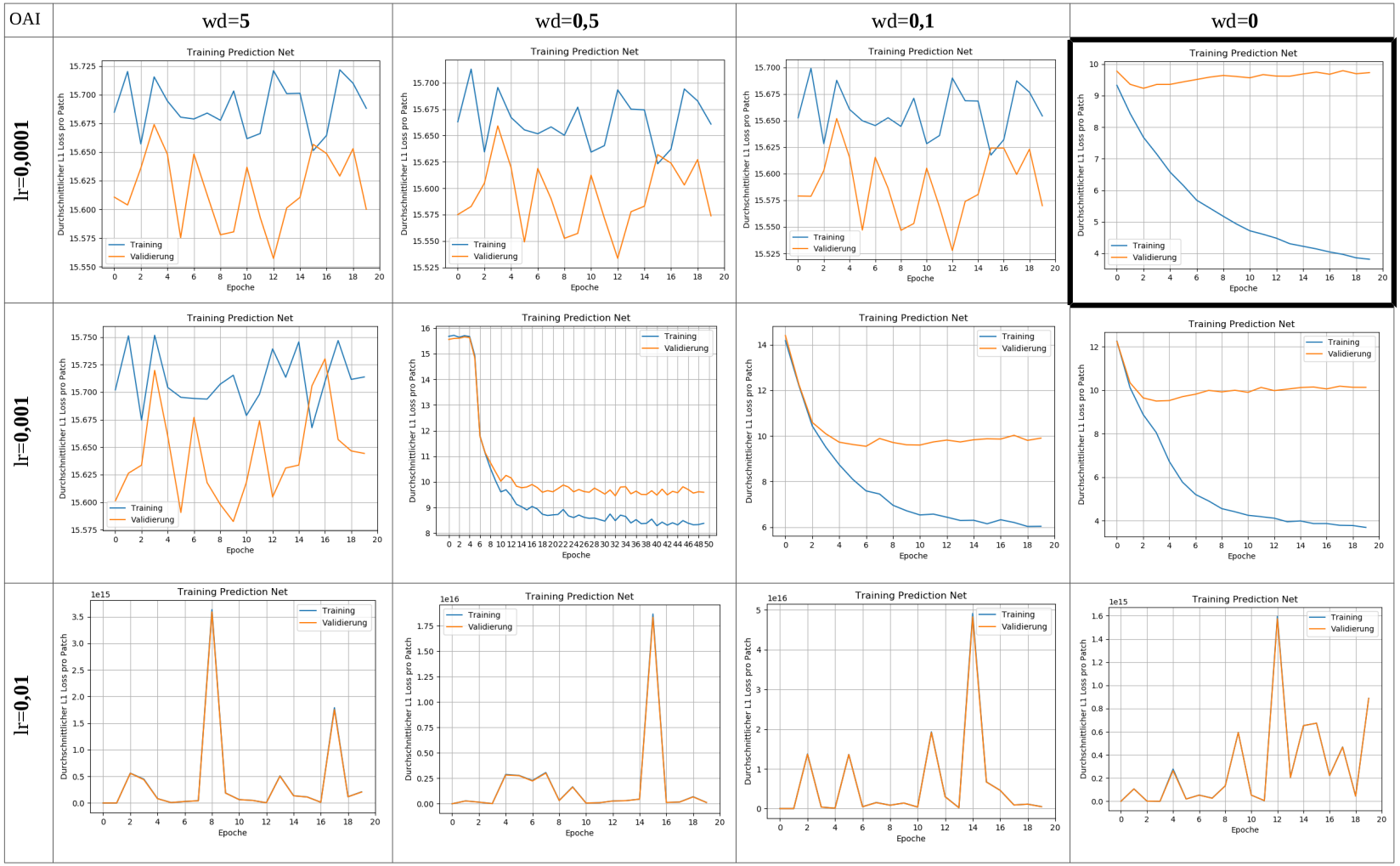
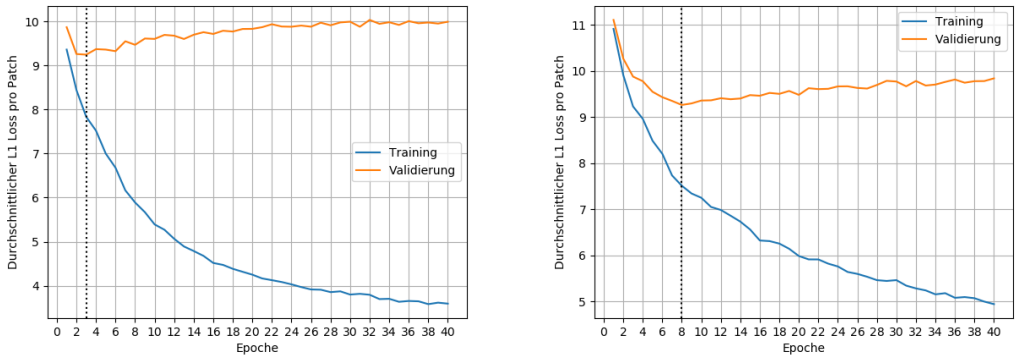


Abbildung 14: Trainingsloss-Verläufe (blau) und Validierungsloss-Verläufe (orange) für das Training des Prediction Net mit dem OAI Datensatz mit unterschiedlichen lr und wd Einstellungen. Nur die Kombinationen ($lr = 0,001$; $wd \in \{0;\ 0,1;\ 0,5\}$) und ($lr = 0,0001$; $wd = 0$) führen zu einem glatten, abfallenden Trainingsloss-Verlauf. Da $wd = 0$ zu den kleinsten Trainingsloss-Werten und $lr = 0,0001$ zu den kleinsten Validierungsloss-Werten führt, wurde diese Kombination gewählt.



(a) $p = 0,2$ **(b)** $p = 0,5$

Abbildung 15: L1-Loss-Verläufe der Prediction Net Trainings mit dem OAI Datensatz unter Verwendung des Dropout-Moduls. Die gestrichelte Linie zeigt die Epoche mit dem geringsten Validierungsloss-Wert an. Die Form des Validierungsloss-Verlaufs wird durch das Dropout-Modul nicht verändert.

patch, für ein zusammengesetztes Momentum und für eine daraus generierte Deformation berechnet. Zusätzlich wurde die $\mathcal{D}^{\mathrm{NCC}}$-Distanz zwischen dem mit der aktuell generierten Deformation transformierten Templatebild und dem durch SVF-Registrierung generierten transformierten Templatebild berechnet.

Die Ergebnisse sind in Abbildung 16 dargestellt. Sie zeigen, dass die durchschnittliche Distanz der transformierten Templatebilder der Validierungsmenge im Gegensatz zum L1-Loss der Momenta während des Trainings abnimmt. Daher wurden die zuvor gewählten Einstellungen $lr = 0,0001$, $wd = 0$ und $p = 0$ beibehalten und das Prediction Net für 20 Epochen trainiert.

Correction Net

Für das Training des Correction Nets wurde zunächst das fertig trainierte Prediction Net für die Bildpaare der Trainings- und Validierungsmenge ausgewertet.
Aus den vorausgesagten Momenta m_p wurden die Transformationen ϕ und ϕ^{-1} durch Auswertung des SVF-Modells generiert und die transformierten Referenzbilder durch Anwenden der Transformationen ϕ generiert. Das Anwenden der Transformation erfolgte mittels Nächste-Nachbarn-Interpolation. Hierdurch werden nur die Intensitätswerte 0 und 1 angenommen und keine Grauwerte in den transformierten Referenzbildern erzeugt, wie es mit bilinearer Interpolation der Fall wäre. Im Anschluss wurden die Grundwahrheitsmomentumkorrekturen m_{GC} durch die in Abschnitt 4.2.1 beschriebene SVF-Registrierung der Bildpaare der Trainings- und Validierungsmenge mit transformierten Referenzbildern generiert.

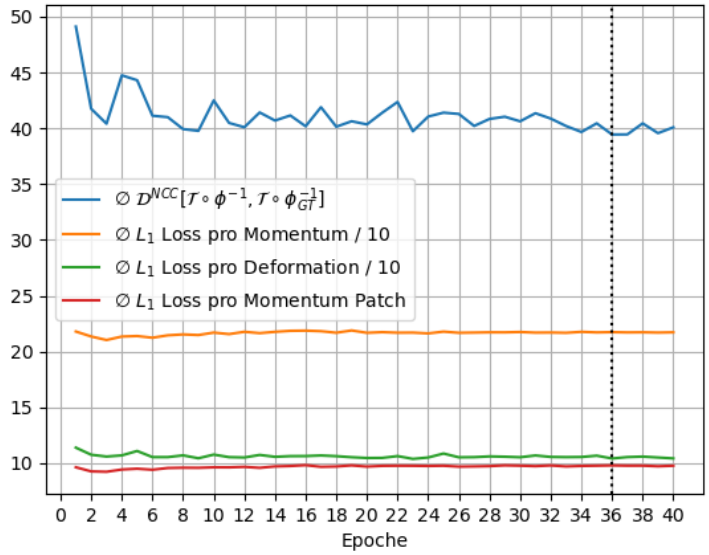


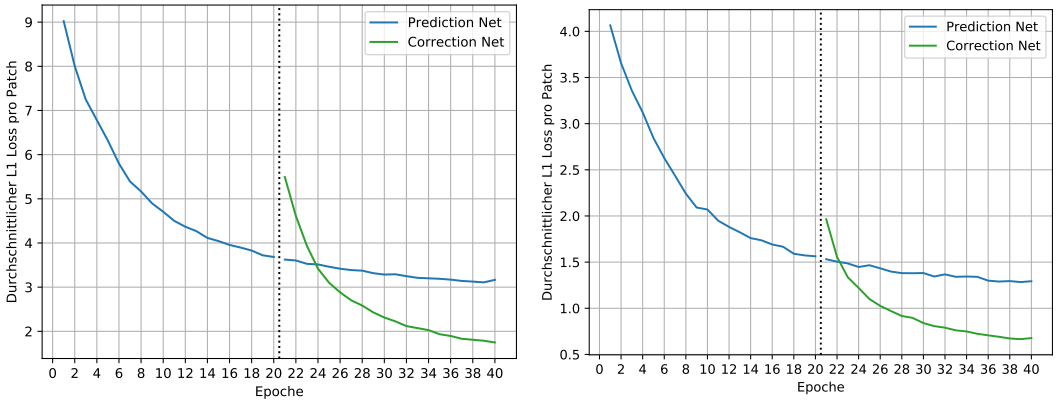
Abbildung 16: Auswertung des Prediction Net Trainings auf der OAI Validierungsmenge. Epoche 36 ist die Epoche mit kleinster durchschnittlicher $\mathcal{D}^{\mathrm{NCC}}$-Distanz.

Analog zur Patchauswahl für das Prediction Net wurden für das Training des Correction Nets Patches der Größe 15×15 Pixel mit einer Schrittweite von 14 Pixeln aus den Bildpaaren der Trainings- und Validierungsmenge entnommen und Hintergrund-Patchpaare aussortiert. Dies führte beim OAI Datensatz zu einer Patchreduktion von 795.010 auf 153.239 Patchpaare der Trainingsmenge, beim synthetischen Datensatz zu einer Patchreduktion von 1.434.343 auf 209.944 Patchpaare der Trainingsmenge.

Für das Training des Correction Nets wurde schließlich das Enkoder-Dekoder-Netzwerk mit exakt denselben Parametereinstellungen wie für das Training des Prediction Nets für 20 Epochen trainiert. Abbildung 17 zeigt den Verlauf des durchschnittlichen L1-Loss-Wertes pro Patch der Trainingsmenge während des Trainings des Gesamtframeworks für beide Datensätze. In beiden Experimenten führt die Verwendung des Correction Nets in Verbindung mit dem Prediction Net zu kleineren Trainingsloss-Werten als das Fortführen des Trainings des Prediction Nets um weitere Epochen.

4.2.3 Auswertung des Gesamtframeworks

Zur Auswertung des Gesamtframeworks wurden die ungesehenen Bildpaare der Testmenge als Eingabe in das Prediction Net gegeben. Es wurde dieselbe Patchauswahl wie in Abschnitt 4.2.2 beschrieben durchgeführt. Aus den durch das Prediction Net vorausgesagten Momentumpatches wurden die Gesamtmomenta m_p zusammenge-



(a) OAI Datensatz **(b)** synthetischer Datensatz

Abbildung 17: Durchschnittlicher L1-Loss pro Patch der Trainingsmenge während des Gesamtframework-Trainings. Die Kombination aus Prediction und Correction Net erzielt kleinere durchschnittliche L1-Loss-Werte pro Patch als ein Training des Prediction Nets für weitere Epochen.

setzt. Aus diesen wurden durch Auswertung des SVF-Modells die Transformationen ϕ und ϕ^{-1} generiert und die Transformation ϕ auf das Referenzbild angewandt.

Die neuen Bildpaare der Testmenge, bestehend aus Templatebild und transformiertem Referenzbild, wurden anschließend als Eingabe in das Correction Net gegeben. Auch hier wurde wieder dieselbe Patchauswahl, wie für das Training des Gesamtframeworks beschrieben, durchgeführt.
Aus den durch das Correction Net vorausgesagten Momentumkorrekturpatches wurden die Gesamtmomentumkorrekturen m_c zusammengesetzt. Diese wurden anschließend mit den vorausgesagten Momenta des Prediction Nets addiert, um die finalen Momenta Voraussagen $\widehat{m} = m_p + m_c$ des Gesamtframeworks zu generieren.

Aus den finalen Momenta $\widehat{m}$ wurden anschließend durch Auswertung des SVF-Modells die Transformationen ϕ und ϕ^{-1} generiert. Die mit ϕ^{-1} transformierten Templatebilder stellen die Ergebnisse der Bildregistrierung mit dem vorgestellten Gesamtframework dar. Die auf diese Weise durchgeführte Registrierung mit dem vorgestellten Gesamtframework wurde im OAI Datensatz im Durchschnitt in 0,183 Sekunden pro Bildpaar und im synthetischen Datensatz im Durchschnitt in 0,142 Sekunden pro Bildpaar durchgeführt. Dies stellt eine eindeutige Verbesserung gegenüber den Laufzeiten im Minutenbereich der SVF-Registrierung mittels numerischer Optimierung dar.

Zur Evaluierung der Ergebnisse wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den transformierten Templatebildern und dem Referenzbild berechnet. Abbildung 18 zeigt die

Übersicht über die Distanzen für beide Datensätze und Abbildung 19 zeigt die Entwicklung der Distanzen für die Ausgangsbilder, registrierten Bilder, vom Prediction Net vorausgesagten Bilder und vom Gesamtframework vorausgesagten Bilder für die Trainings-, Validierungs- und Testmengen beider Datensätze.

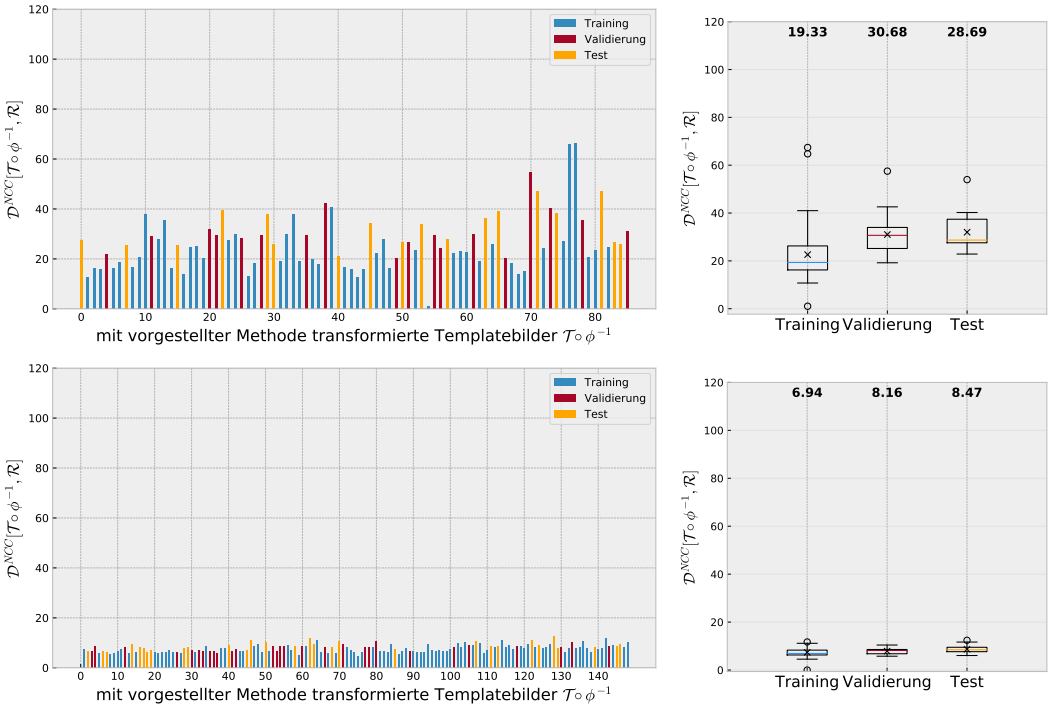


Abbildung 18: $\mathcal{D}^{\mathrm{NCC}}$-Distanzen nach der Registrierung mit dem Gesamtframework im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung führt zu einer deutlichen Verringerung des Fehlermaßes gegenüber den Anfangsdistanzen. Das geringe Fehlermaß nach der SVF-Registrierung wird nicht erreicht. Ausreißer im OAI Datensatz wurden von der SVF-Registrierung übernommen.

Für die OAI Testmenge konnte der Wert des Medians von ursprünglich 93,24 durch das Prediction Net auf 40,38 reduziert werden. Durch das Correction Net wurde dieser Wert weiter auf 28,69 reduziert. Für die Trainings- und Validierungsmenge wurden vergleichbare Reduktionen erzielt, die Werte der Trainingsmenge sind jedoch erwartungsgemäß geringer als die Werte der Validierungs- und Testmenge, da mit ihm die Netze trainiert wurden.

Die Problembilder 76 und 77 der Trainingsmenge führten auch bei der Registrierung mit dem Gesamtframework zu Ausreißern in den Distanzen. Sie fallen im Verhältnis zu den anderen erzielten Distanzen jedoch geringer aus.

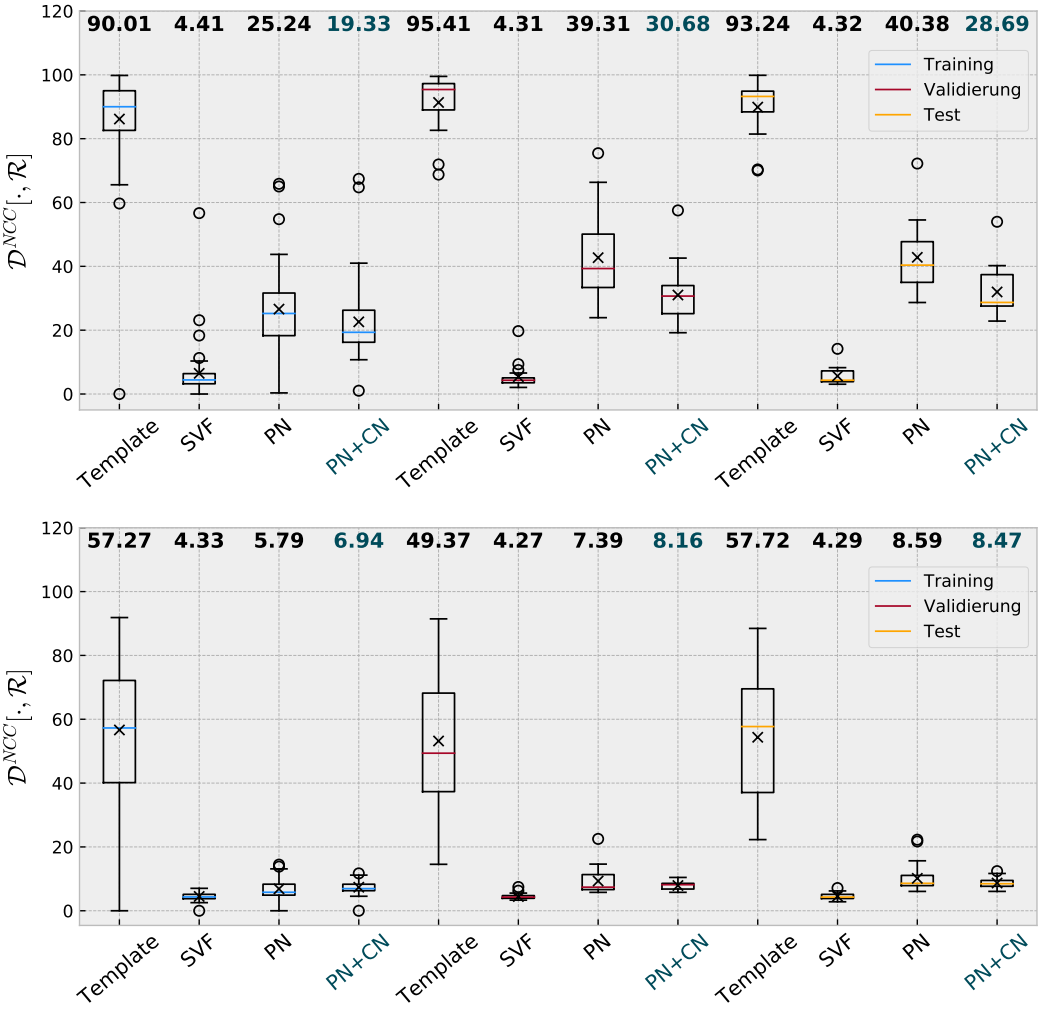


Abbildung 19: Entwicklung der $\mathcal{D}^{\mathrm{NCC}}$-Distanzen im OAI Datensatz (oben) und synthetischen Datensatz (unten). Template bezeichnet die Ausgangsbilder, SVF die durch SVF-Registrierung generierten Bilder, PN die vom Prediction Net generierten Bilder und PN+CN die vom Gesamtframework generierten Bilder. Die Medianwerte sind am oberen Rand gegeben. Die Registrierung mit dem PN+CN verringert das Fehlermaß gegenüber den durch das PN erzielten Werten weiter und reduziert die Varianz der Distanzwerte.

Die Varianz der Distanzen ist nach Anwendung des Correction Nets sowohl für die Test- als auch für die Trainings- und Validierungsmenge geringer als die Varianz der Distanzen nach Anwendung des Prediction Nets.

Für die Testmenge des synthetischen Datensatzes konnte der Wert des Medians von ursprünglich 57,72 durch das Prediction Net auf 8,59 reduziert werden. Durch das Correction Net wurde dieser Wert geringfügig auf 8,47 reduziert. Für die Trainings- und Validierungsmenge wurden vergleichbare Reduktionen durch das Prediction Net erzielt, durch das Correction Net jedoch geringfügige Erhöhungen der Medianwerte herbei geführt.
Die Varianz der Distanzen ist nach Anwendung des Correction Nets jedoch sowohl für die Test- als auch für die Trainings- und Validierungsmenge geringer als die Varianz der Distanzen nach Anwendung des Prediction Nets. Dies spiegelt sich in kleineren Medianwerten wieder.

Zusätzlich wurden zur Evaluierung der durch die Registrierung mit dem Gesamtframework generierten Transformationen ϕ^{-1} die $\mathcal{D}^{\mathrm{SSD}}$-Distanzen zwischen diesen Transformationen und den durch die SVF-Registrierung generierten Transformationen berechnet. Die $\mathcal{D}^{\mathrm{SSD}}$-Distanz wurde als Fehlermaß für die Transformationen gewählt, da sie einen punktweisen Vergleich durchführt und auch sehr kleine Abweichungen erfasst. Eine Übersicht über die Distanzen ist in Abbildung 20 gegeben.

In der Übersicht ist zu erkennen, dass in beiden Datensätzen sehr große Werte von einer kleinen Menge an Ausreißern angenommen werden. Das Ausreißermuster stimmt im OAI Datensatz jedoch nicht mit dem Ausreißermuster der $\mathcal{D}^{\mathrm{NCC}}$-Bilddistanzen überein, im synthetischen Datensatz tauchen erstmals Ausreißer auf.

In den Abbildungen 21a und 21b sind die Ergebnisse für die Bilder 84 und 29 aus der OAI Testmenge und in den Abbildungen 21c und 21d sind die Ergebnisse für die Bilder 7 und 128 aus der Testmenge des synthetischen Datensatzes dargestellt.
Für die Bilder 84 und 7 führte die Registrierung mit dem Gesamtframework zu besonders kleinen $\mathcal{D}^{\mathrm{NCC}}$-Werten. Sie zeigen in den Differenzbildern kaum Unterschiede zum Referenzbild. Für die Bilder 29 und 128 führte die Registrierung mit dem Gesamtframework zu besonders hohen $\mathcal{D}^{\mathrm{NCC}}$-Werten. Die Abweichungen vom Referenzbild sind gut erkennbar.

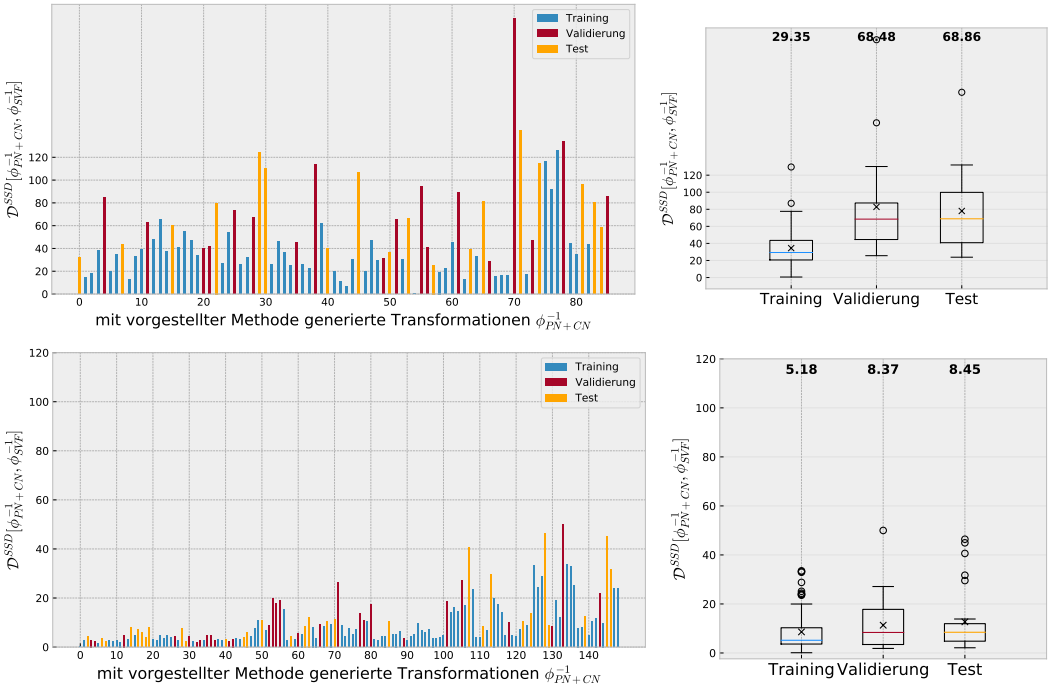
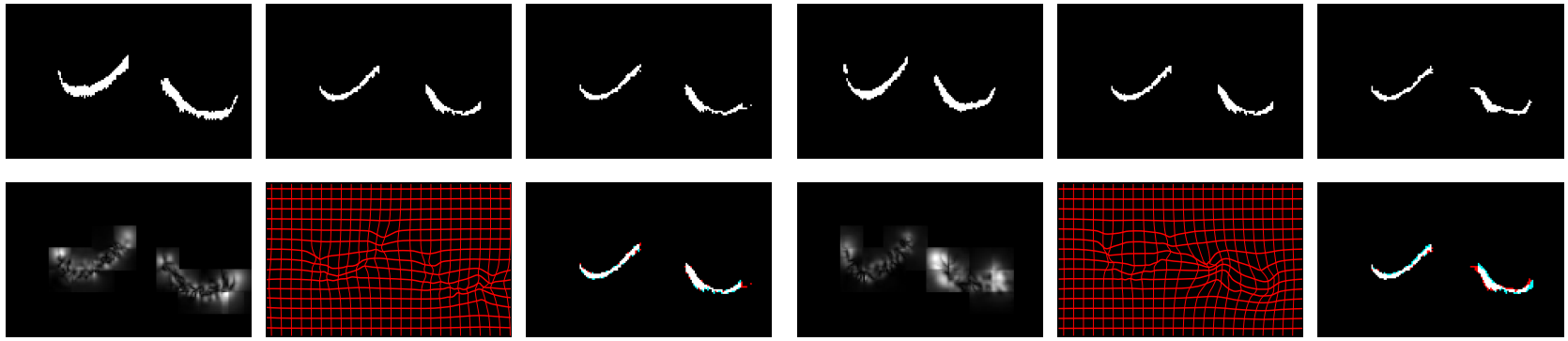
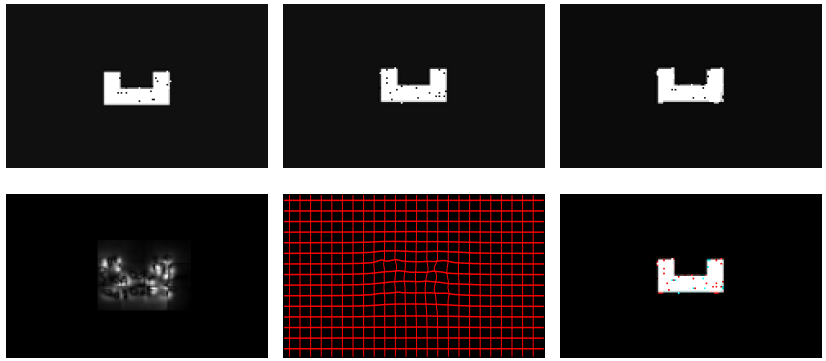


Abbildung 20: $\mathcal{D}^{\mathrm{SSD}}$-Distanzen nach der Registrierung mit dem Gesamtframework im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. In beiden Datensätzen führt eine kleine Menge an Transformationen zu starken Ausreißern. Die Ausreißermuster stimmen nicht mit den Ausreißermustern des $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der zugehörigen transformierten Templatebilder überein (vgl. Abb. 18)

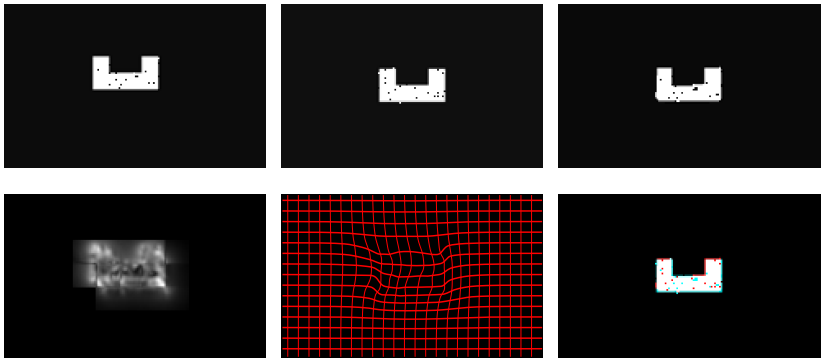


(a) OAI Datensatz – Bild 84, $\mathcal{D}^{\mathrm{NCC}} = 22,85$

(b) OAI Datensatz – Bild 29, $\mathcal{D}^{\mathrm{NCC}} = 39,88$



(c) synthetischer Datensatz – Bild 7, $\mathcal{D}^{\mathrm{NCC}} = 6,06$



(d) synthetischer Datensatz – Bild 128, $\mathcal{D}^{\mathrm{NCC}} = 12,41$

Abbildung 21: Registrierungsergebnisse des Gesamtframeworks für Bilder der Testmengen beider Datensätze. Die Ergebnisse für die Bilder 84 und 7 sind gut, die transformierten Bilder zeigen nur geringe Unterschiede zum Templatebild. Die Ergebnisse für die Bilder 29 und 128 sind im Vergleich schlechter, die transformierten Bilder zeigen deutliche Unterschiede zum Templatebild. Anordnung jeweils:
Oben: Templatebild (links), Referenzbild (Mitte), transformiertes Templatebild (rechts)
Unten: Momentum $|\widehat{m}|$ (links), Transformation (Mitte), Differenzenbild: transformiertes Template rot, Referenz blau (rechts).

4.3 Vergleich mit weiteren Methoden des Maschinellen Lernens

Zur Einordnung der Ergebnisse wurden Experimente zum Vergleich mit der Quicksilver-Methode für das SVF-Modell und mit dem FlowNet durchgeführt. Im Folgenden werden die Experimente beschrieben und die Ergebnisse vorgestellt.

4.3.1 Quicksilver-Methode für das SVF-Modell

Für den Vergleich mit der Quicksilver-Methode wurde eine Adaption dieser Methode für das SVF-Modell in Python unter Verwendung der Mermaid-Toolbox implementiert. Für die Erzeugung der Grundwahrheit für das Prediction Net wurden die in Abschnitt 4.2.2 beschriebene SVF-Registrierung verwendet. Es wurden ebenfalls Patches der Größe 15×15 Pixel mit einer Schrittweite von 14 Pixeln ausgewählt und das Prediction Net ebenfalls mit den Parametereinstellungen $lr = 0,0001$, $wd = 0$ und $p = 0$ für 20 Epochen trainiert. Das Training des Quicksilver Correction Net wurde mit den Momentumdifferenzen $m_{\mathrm{GP}} - m_p$ unter Verwendung derselben Parametereinstellungen durchgeführt.

Die L1-Loss-Verläufe des Trainings von Prediction Net, Correction Net und Quicksilver Correction Net sind in Abbildung 22 für beide Datensätze dargestellt. Der Verlauf des Trainingslosses des Quicksilver Correction Nets ist dem Verlauf des Trainingslosses des Correction Net recht ähnlich. Die Kombination aus Prediction Net und Quicksilver Correction Net führt ebenfalls zu geringeren L1-Loss-Werten als das Fortführen des Trainings des Prediction Nets für weitere Epochen.

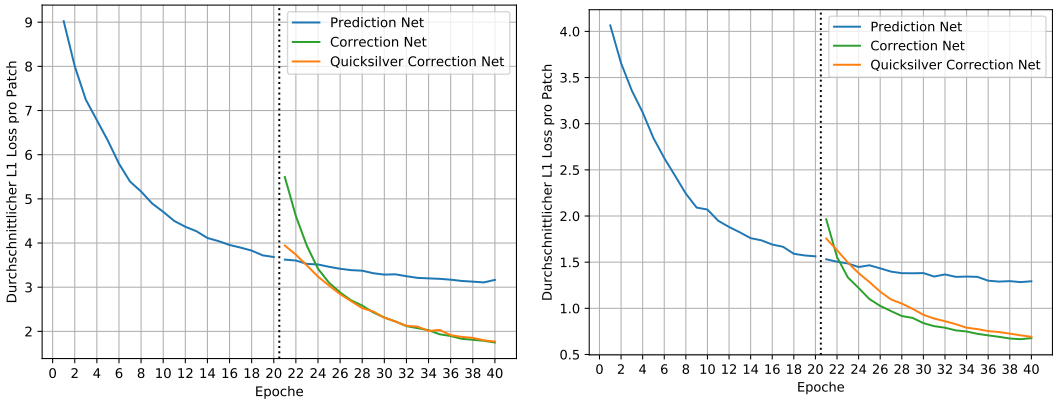


Abbildung 22: Durchschnittlicher L1-Loss pro Patch der Trainingsmenge während des Prediction Net, Correction Net und Quicksilver Correction Net Trainings für den OAI Datensatz (links) und den synthetischen Datensatz (rechts). Die Kombination aus Prediction und (Quicksilver) Correction Net erzielt jeweils kleinere Losswerte pro Patch als ein Training des Prediction Nets für weitere Epochen.

Analog zur Auswertung des in dieser Arbeit vorgestellten Gesamtframeworks wurden die Quicksilver-Methode mit ungesehenen Bildpaaren der Testmenge ausgewertet.

Für den Vergleich mit dem in dieser Arbeit vorgestellten Gesamtframework wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den so generierten transformierten Templatebildern und dem Referenzbild berechnet. Abbildung 23 zeigt die Übersicht über die Distanzen für beide Datensätze. Eine Übersicht über diese Distanzwerte im Vergleich zu den mit der vorgestellten Methode und der SVF-Registrierung erzielten Werten ist für beide Datensätze in Abbildung 24 gegeben.

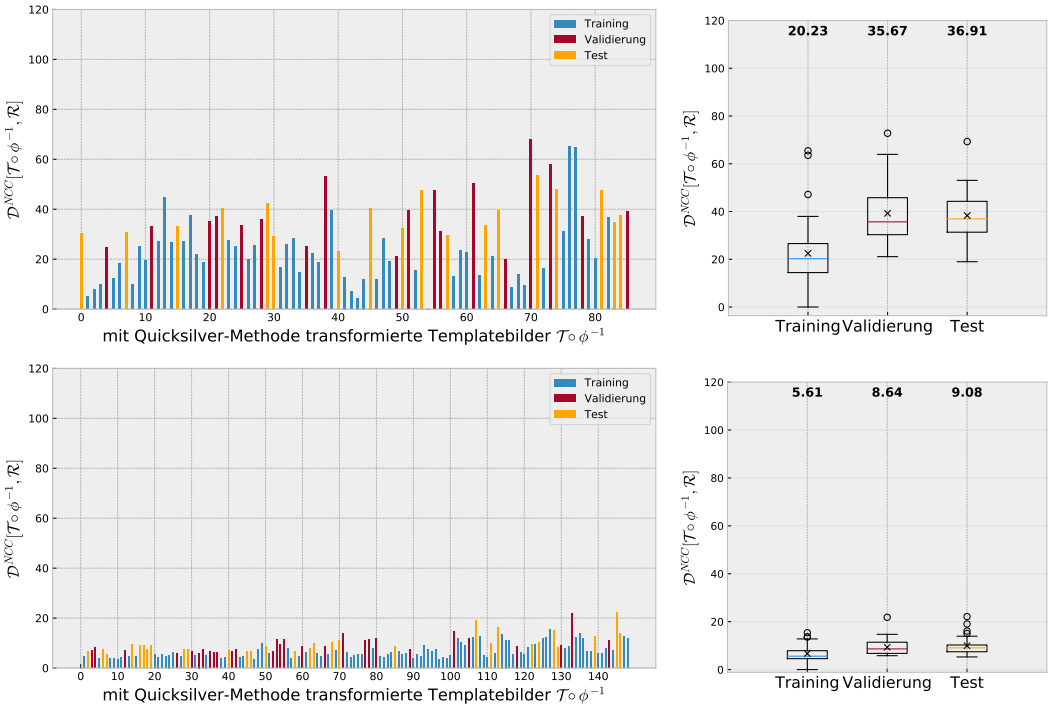


Abbildung 23: $\mathcal{D}^{\mathrm{NCC}}$-Distanzen nach der Registrierung mit der Quicksilver-Methode im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung führt zu einer Verringerung des Distanzmaßes gegenüber den Anfangsdistanzen. Das geringe Fehlermaß nach der SVF-Registrierung wird nicht erreicht und auch das Fehlermaß nach der Registrierung mit dem vorgestellten Gesamtframework ist geringer. Zudem gibt es mehr Ausreißer als nach der Registrierung mit dem Gesamtframework (vgl. Abb. 18).

Für den OAI Datensatz wurden durch die Registrierung mit der Quicksilver-Methode für das SVF-Modell sowohl für die Testmenge als auch für die Trainings- und Validierungsmenge eine Reduktion der Medianwerte allein gegenüber den durch

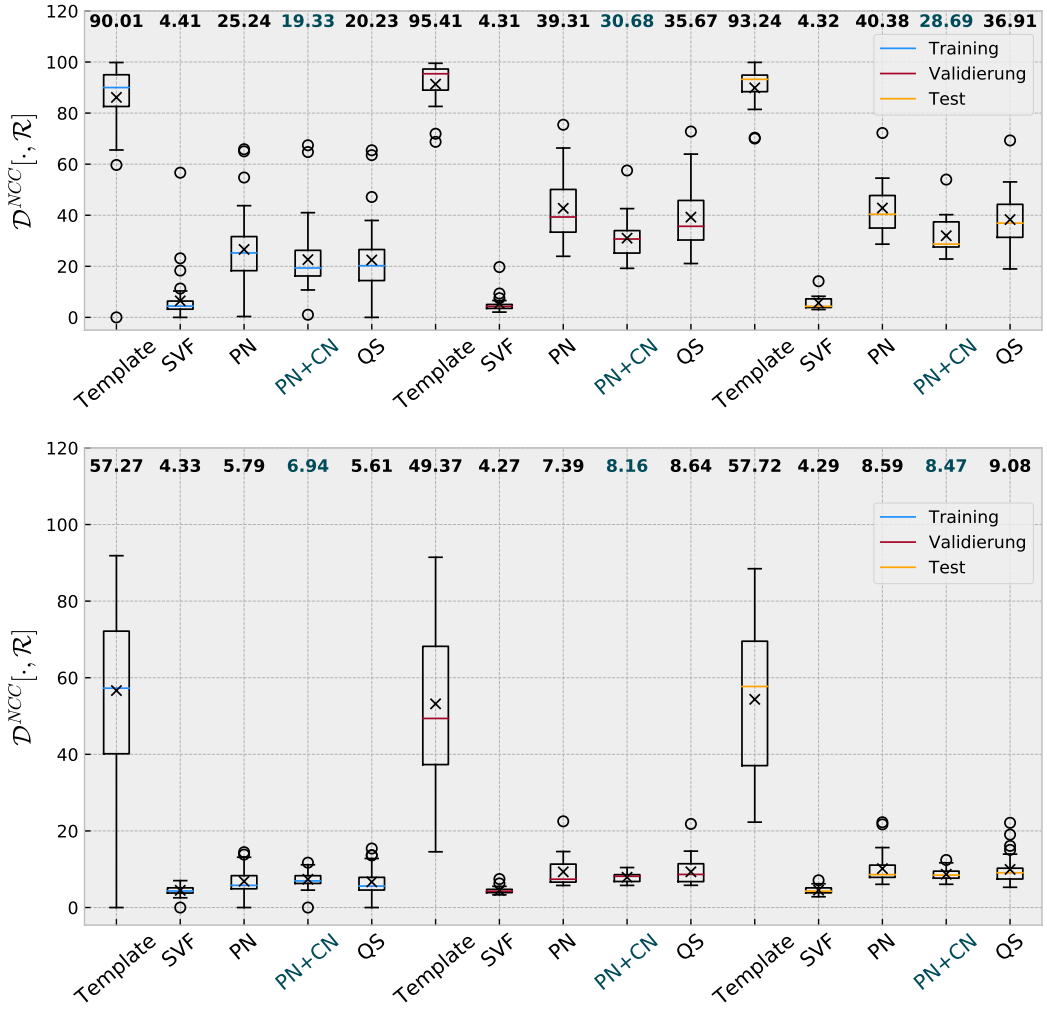


Abbildung 24: Entwicklung der $\mathcal{D}^{\text{NCC}}$-Distanzen im OAI Datensatz (oben) und synthetischen Datensatz (unten). Template bezeichnet die Ausgangsbilder, SVF die durch SVF-Registrierung generierten, PN die vom Prediction Net generierten, PN+CN die vom Gesamtframework generierten und QS die von der Quicksilver-Methode generierten Bilder. Die Medianwerte sind am oberen Rand gegeben. Die Registrierung mit QS verringert das Fehlermaß für den OAI Datensatz gegenüber den durch das PN erzielten Werten weiter, erreicht die durch das PN+CN erzielten niedrigen Distanzwerte aber nicht. Die Varianz der Distanzwerte wird nicht reduziert. Für den synthetischen Datensatz können die Distanzwerte des PN nur minimal verbessert werden, die Varianzen werden nicht reduziert.

das Prediction Net erzielten Werten erreicht. In allen drei Fällen liegt der Medianwert jedoch über dem durch das in dieser Arbeit vorgestellte Gesamtframework erzielten Medianwert. Zudem weisen die $\mathcal{D}^{\text{NCC}}$-Distanzen für die Quicksilver Ergebnisse eine größere Varianz auf.

Für den synthetischen Datensatz konnte durch die Registrierung mit der Quicksilver-Methode für das SVF-Modell sowohl für die Testmenge als auch für die Trainings- und Validierungsmenge keine deutliche Reduktion der Medianwerte allein gegenüber den durch das Prediction Net erzielten Werten erreicht werden. Zudem liegt der Medianwert für die Test- und Validierungsmenge über dem durch das in dieser Arbeit vorgestellte Gesamtframework erzielten Medianwert, für die Trainingsmenge ist der Medianwert geringer. In allen drei Fällen weisen die $\mathcal{D}^{\text{NCC}}$-Distanzen für die Quicksilver Ergebnisse eine größere Varianz auf als die durch das Gesamtframework erzielten Distanzen.

Zusätzlich wurden die $\mathcal{D}^{\text{SSD}}$-Distanzen zwischen den durch die Registrierung mit der Quicksilver-Methode generierten Transformationen ϕ^{-1} und den durch die SVF-Registrierung generierten Transformationen berechnet. Eine Übersicht über die Distanzen ist für beide Datensätze in Abbildung 25 gegeben.

Im Vergleich zu den $\mathcal{D}^{\text{SSD}}$-Distanzen der mit dem Prediction Net generierten Transformationen gibt es in beiden Datensätzen eine größere Anzahl an Ausreißern. Zusätzlich werden auch insgesamt größere Distanzen generiert. Der Medianwert der OAI Trainingsmenge ist von 68,86 für die Differenzen des Prediction Frameworks auf 88,65 für die Differenzen der Quicksilver-Methode angestiegen. Der Medianwert der Trainingsmenge des synthetischen Datensatzes ist von 8,45 für die Differenzen des Gesamtframeworks auf 10,2 für die Differenzen der Quicksilver-Methode angestiegen.

In der Abbildung 26 sind die Ergebnisse der Quicksilver-Methode im direkten Vergleich zu den Ergebnissen des vorgestellten Gesamtframeworks für das Bild 84 des OAI Datensatzes (Abb. 26a) und für das Bild 7 des synthetischen Datensatzes (Abb. 26b) dargestellt. Es werden die Differenzenbilder, die Deformationsfelder und die finalen Momenta verglichen.

Der Vergleich der finalen Momenta zeigt für beide Datensätze höhere Intensitäten für die finalen Momenta die vom vorgestellten Gesamtframework vorausgesagt wurden. Dies spiegelt sich in kleinen Unterschieden in den Transformationsfeldern wieder. Während die Differenzenbilder für Bild 7 für beide Methoden sehr ähnlich sind, lassen sich für Bild 84 größere Differenzen im Differenzenbild der Quicksilver-Methode erkennen als im Differenzenbild des vorgestellten Gesamtframework.

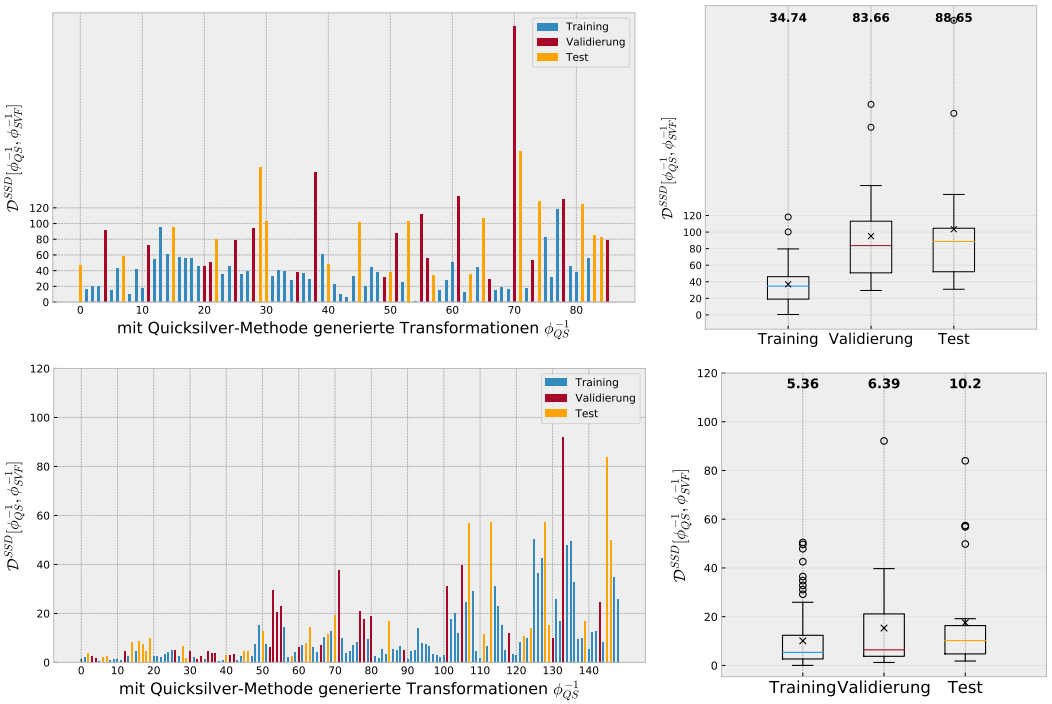
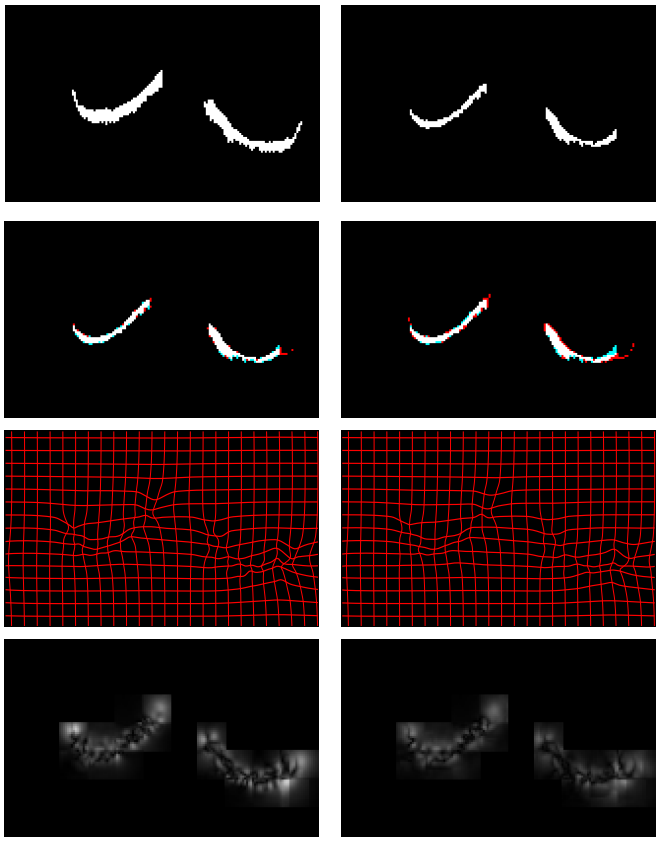
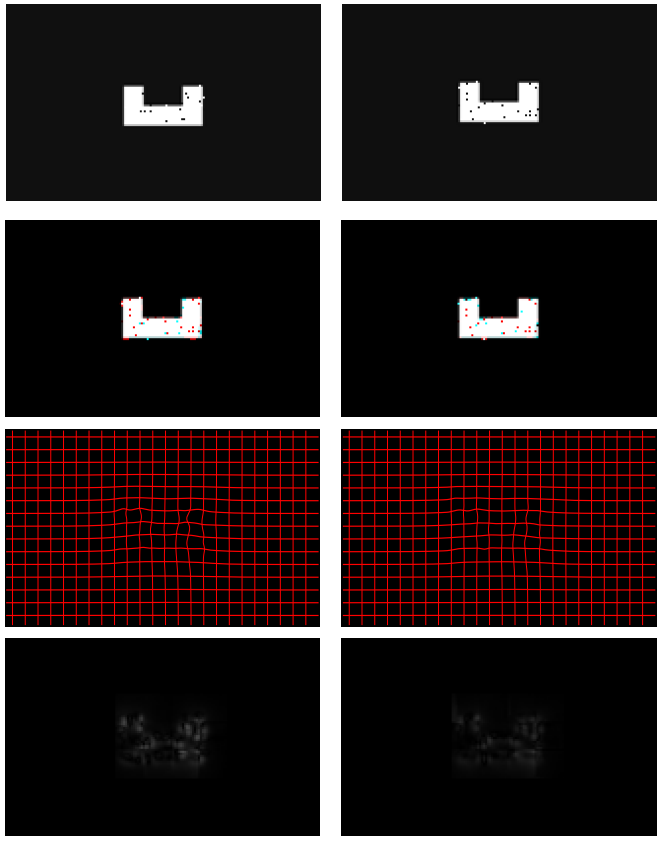


Abbildung 25: $\mathcal{D}^{\mathrm{SSD}}$-Distanzen nach der Registrierung mit der Quicksilver-Methode für SVF im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung liefert ein höheres Fehlermaß als die Registrierung mit dem Gesamtframework (vgl. 20). Das Ausreißermuster ähnelt dem des Gesamtframeworks, die Ausreißer generieren in der Quicksilver Methode jedoch größere Fehlermaße.



(a) OAI Datensatz - Bild 84: PN+CN links - QS rechts



(b) synth. Datensatz - Bild 7: PN+CN links - QS rechts

Abbildung 26: Vergleich der Ergebnisse des vorgestellten Gesamtframeworks (PN+CN) mit den Ergebnissen der Quicksilver-Methode (QS) für Bilder der Testmengen. Das Momentum von Bild 7 weist für PN+CN höhere Intensitäten auf als für QS. Aufgrund der geringen Transformation hat dies keinen Einfluss auf die Differenzenbilder und Transformationen, die für beide Methoden vergleichbar sind. Auch für Bild 84 weist das PN+CN Momentum höhere Intensitäten auf als das QS Momentum. Hier führt dies zu einer größeren Transformation und zu weniger deutlichen Unterschieden im Differenzenbild. Anordnung jeweils:
1. Zeile: Templatebild (l.), Referenzbild (r.), **2. Zeile:** Differenzenbild - transformiertes Template rot, Referenz blau,
3. Zeile: Deformationsfeld, **4. Zeile:** finales Momentum $|\widehat{m}|$.

4.3.2 FlowNet

Für den Vergleich mit dem FlowNet-Ansatz wurde die Python-Implementierung des FlowNetC von Daniel Budelmann im Rahmen seiner Masterarbeit [Bud19] genutzt. Hierfür wurden die mit der SVF-Registrierung generierten Deformationsfelder mittels bilinearer Interpolation auf ein Viertel ihrer Auflösung herunterskaliert, sodass sie eine Größe von $2 \times 25 \times 40$ Pixeln hatten. Mit diesen Deformationsfeldern der Trainingsmenge wurde das FlowNet für 10.000 Iterationen unter Verwendung des Adam-Optimierungsverfahrens mit der Lernrate $lr = 0,001$ trainiert. Die verwendete Lossfunktion basiert auf dem Endpunktfehler (engl. endpoint error (EPE)) und hat die Form

$$L^{\mathrm{EPE}}(y, y^{\star}) = \frac{1}{|y|}\|y - y^{\star}\|_2, \tag{4.1}$$

wobei $|y|$ die Anzahl der Vektoren bezeichnet. Der Verlauf der durchschnittlichen Losswerte pro Bild in der höchsten Auflösung für die Trainings- und Validierungsmenge ist in Abbildung 27 für beide Datensätze dargestellt.

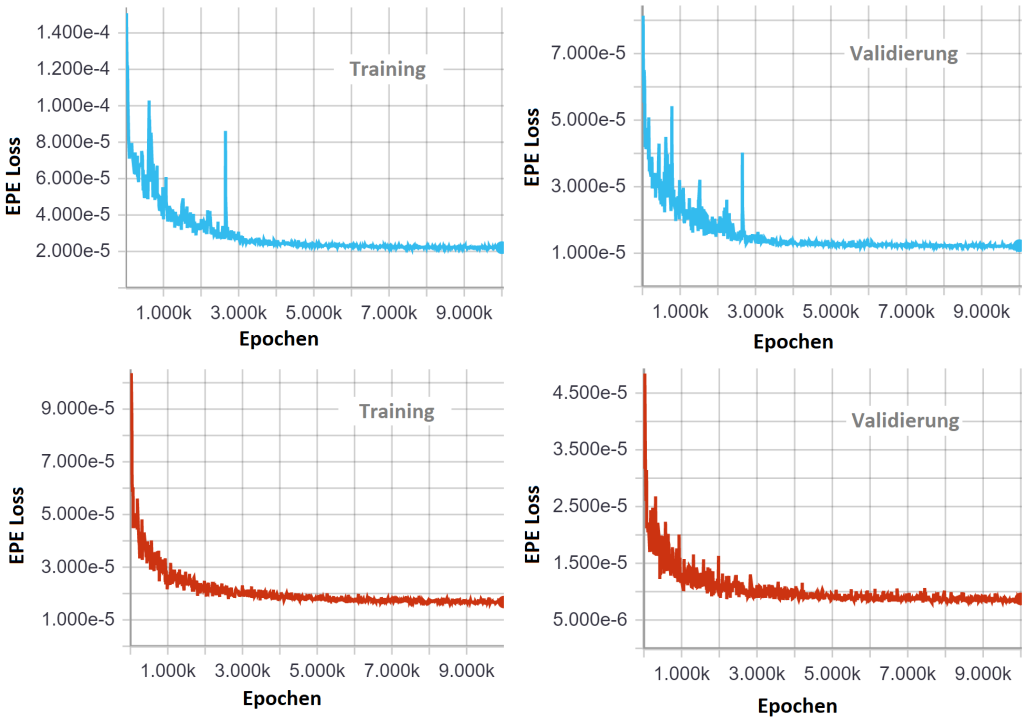


Abbildung 27: Durchschnittlicher EPE Loss-Verlauf pro Bild für das Training des FlowNets für Trainings- und Validierungsmenge des OAI Datensatzes (oben) und des synthetischen Datensatzes (unten). Der Trainingsloss ist mit dem Faktor 0,005 gewichtet. Für beide Datensätze zeigen sowohl der Trainings- als auch der Validierungsloss einen abfallenden Verlauf.

Das fertig trainierte FlowNet wurde anschließend für die Testmenge beider Datensätze ausgewertet. Die generierten Deformationsfelder wurden mittels bilinearer Interpolation auf die ursprüngliche Auflösung von $2 \times 100 \times 160$ Pixeln hochskaliert. Durch Anwenden dieser Transformationen wurden die transformierten Templatebilder generiert.

Für den Vergleich mit dem in dieser Arbeit vorgestellten Gesamtframework wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den so generierten transformierten Templatebildern und dem Referenzbild berechnet. Eine Übersicht über die Distanzen ist für beide Datensätze in Abbildung 28 dargestellt.

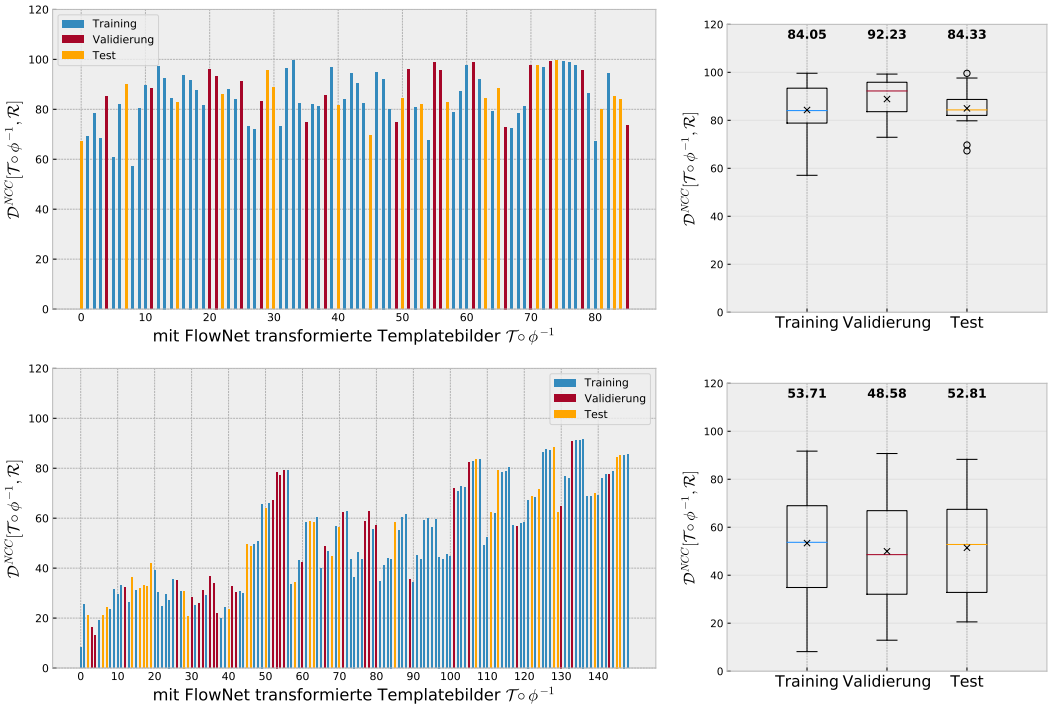


Abbildung 28: $\mathcal{D}^{\mathrm{NCC}}$-Distanzen nach der Registrierung mit der FlowNet-Methode im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung verringert das Fehlermaß kaum. Das Verteilungsmuster ähnelt daher stark dem Verteilungsmuster der Anfangsdistanzen (vgl. Abb 11).

Gegenüber den anfänglichen $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der Templatebilder konnte mit den durch das FlowNet vorausgesagten Transformationen nur geringe Verbesserungen erzielt werden. Dabei wurden die Distanzen für alle Bilder gleichmäßig reduziert, das Verteilungsmuster blieb sowohl im OAI Datensatz als auch im synthetischen Datensatz erhalten.

In Abbildung 29 ist eine Übersicht über diese Distanzwerte im Vergleich zu den mit der vorgestellten Methode erzielten Werten für beide Datensätze gegeben.

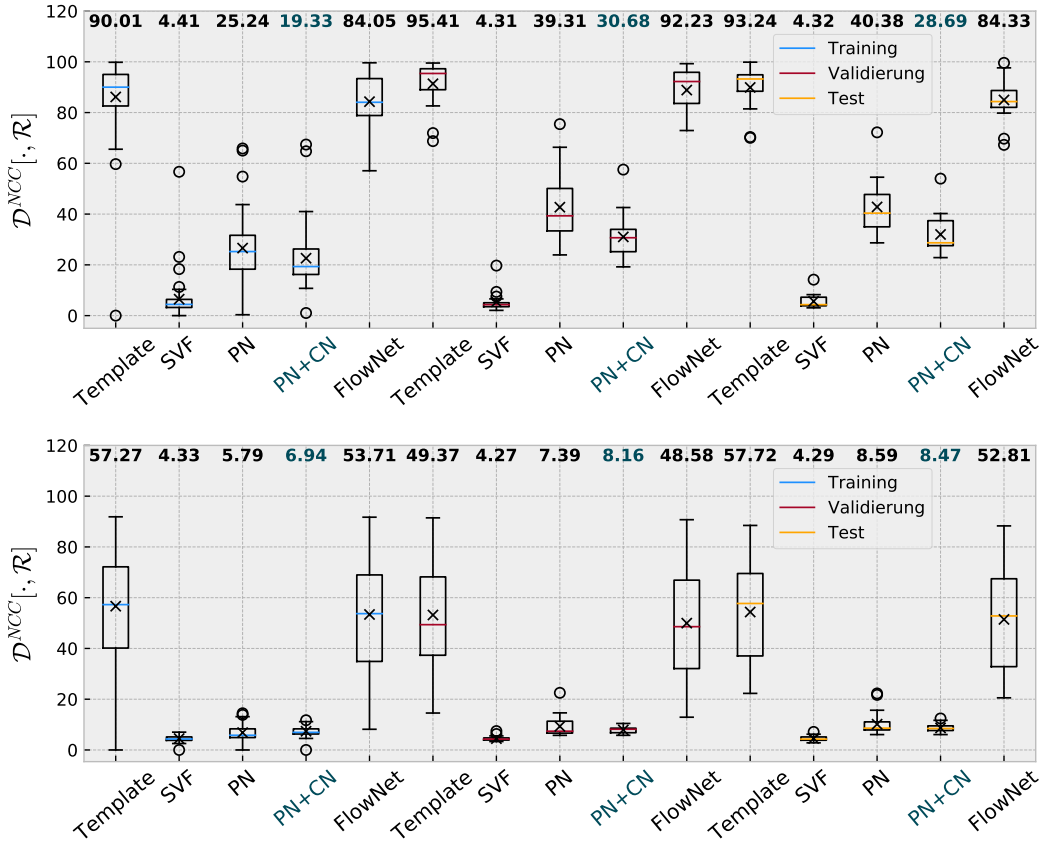


Abbildung 29: Entwicklung der $\mathcal{D}^{\mathrm{NCC}}$-Distanzen im OAI Datensatz (oben) und synthetischen Datensatz (unten). Template bezeichnet die Ausgangsbilder, SVF die durch SVF-Registrierung generierten, PN die vom Prediction Net generierten und PN+CN die von dem vorgestellten Gesamtframework generierten generierten Bilder. Die Medianwerte sind am oberen Rand gegeben. Die Registrierung verringert das Fehlermaß nur minimal, die Varianz kann nicht reduziert werden. Die Ergebnisse der Registrierung sind deutlich schlechter als die von SVF, PN und PN+CN.

Sie zeigt, dass die durch das FlowNet erzielten Distanzen in beiden Datensätzen deutlich größer sind als die durch das Prediction Net und das Gesamtframework erzielten Distanzen.

Zusätzlich wurden die $\mathcal{D}^{\mathrm{SSD}}$-Distanzen zwischen den durch das FlowNetC generierten Transformationen ϕ^{-1} und den durch die SVF-Registrierung generierten Transformationen berechnet. Eine Übersicht über die Distanzen ist in Abbildung 30

gegeben. Aufgrund der sehr großen Werte wurden die Distanzen des synthetischen Datensatzes für die Visualisierung durch 10 geteilt.

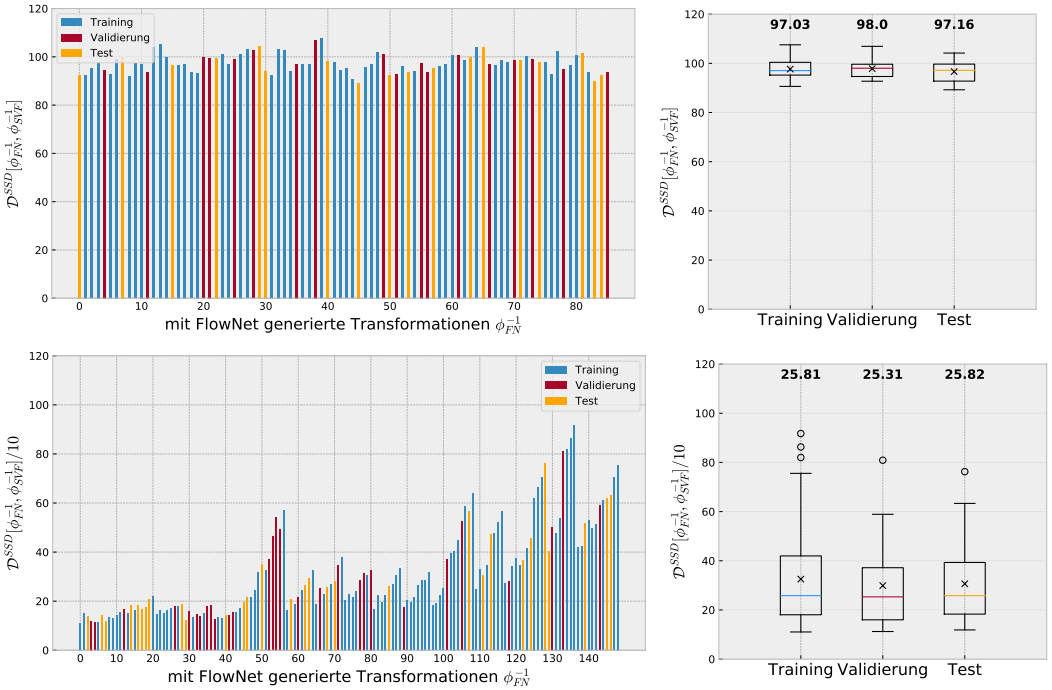


Abbildung 30: $\mathcal{D}^{\mathrm{SSD}}$-Distanzen nach der Registrierung mit der FlowNet-Methode im OAI Datensatz (oben) und synthetischen Datensatz (unten). Die Medianwerte sind am oberen Rand gegeben. Die Registrierung führt zu sehr großen Fehlermaßwerten, die Werte für den synthetischen Datensatz wurden für die Visualisierung durch 10 geteilt. Ähnlich zu den Verteilungen der $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der durch die FlowNet-Methode transformierten Templatebilder weisen die Verteilungsmuster der $\mathcal{D}^{\mathrm{SSD}}$-Distanzen Ähnlichkeit zu den Anfangsdifferenzen der Templatebilder auf (vgl. Abb. 11).

Die Distanzen des OAI Datensatzes weisen keine Ausreißer auf, sondern zeigen durchgehend hohe Werte, was sich in annähernd gleichen Medianwerten für die Trainings-, Test- und Validierungsmenge widerspiegelt. Die Distanzen des synthetischen Datensatzes sind sehr hoch, das Verteilungsmuster ähnelt dem der $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der durch das FlowNet generierten transformierten Bilder.

In der Abbildung 31 sind die Ergebnisse der FlowNet-Methode im Vergleich zur Grundwahrheit für das Bild 84 des OAI Datensatzes und für das Bild 7 des synthetischen Datensatzes dargestellt. Es werden die Differenzenbilder, die Deformationsfelder und die finalen Momenta verglichen.

Die Differenzenbilder zeigen für beide Datensätze deutlich größere Differenzen als

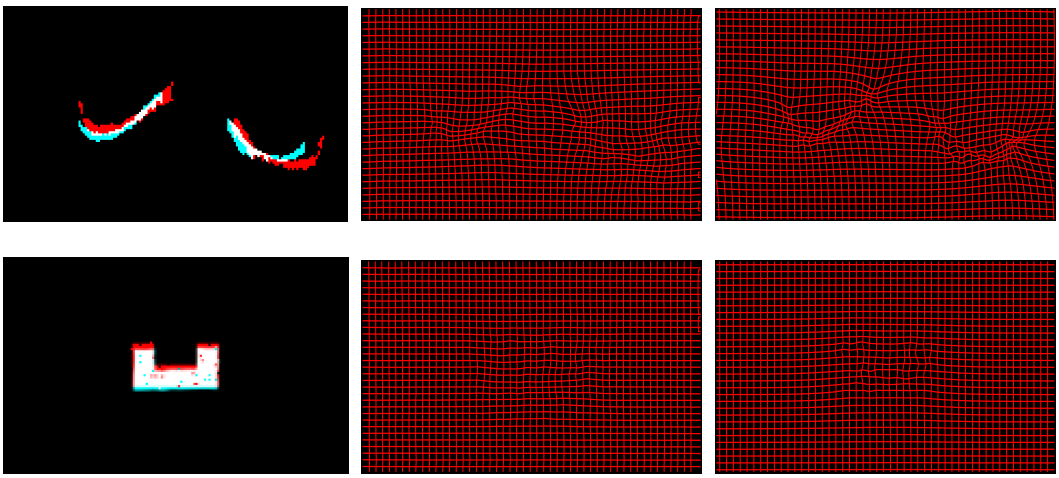


Abbildung 31: Ergebnisse der FlowNet-Methode für das Bild 84 des OAI Datensatzes (oben) und für das Bild 7 des synthetischen Datensatzes (unten). Die Differenzenbilder zeigen für beide Bilder deutlich erkennbare Unterschiede. Die mit der FlowNet-Methode generierten Transformationen zeigen nur eine Grundform der durch SVF-Registrierung generierten Grundwahrheitsdeformation.
Anordnung jeweils: **Links:** Differenzenbild aus transformiertem Templatebild (rot) und Referenzbild (blau). **Mitte:** Mit FlowNet generiertes und hochskaliertes Deformationsfeld. **Rechts:** Mit SVF-Registrierung generiertes Deformationsfeld.

die in Abbildung 26 für das vorgestellte Prediction Net und die Quicksilver-Methode gezeigten Differenzenbilder. Der Vergleich der Deformationsfelder zeigt deutlich kleinere Deformationen in den durch das FlowNet generierten Deformationsfeldern und erklärt die sehr hohen $\mathcal{D}^{\text{SSD}}$-Werte. Während im Deformationsfeld für Bild 84 des OAI Datensatzes zumindest noch die Grundform der starken Grundwahrheitsdeformation erkennbar ist, ist dies im Falle des Deformationsfeldes für Bild 7 des synthetischen Datensatzes, das eine schwächere Grundwahrheitsdeformation besitzt, nicht der Fall.

4.4 Diskussion

Im Folgenden werden die zuvor beobachteten experimentellen Ergebnisse zusammengefasst und diskutiert.

Erzeugung der Grundwahrheit

Für die Erzeugung der Grundwahrheit wurde eine zweischrittige SVF-Registrierung durch numerische Optimierung des SVF-Modells durchgeführt. Die Rechenzeit betrug hierfür im OAI Datensatz im Durchschnitt 55,96 Sekunden pro Bildpaar und im synthetischen Datensatz im Durchschnitt 61,47 Sekunden pro Bildpaar. Es wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den transformierten Templatebildern der Trainings-, Validierungs- und Testmenge und dem Referenzbild berechnet, da das $\mathcal{D}^{\mathrm{NCC}}$-Distanzmaß in der Formulierung der Registrierungsenergie verwendet wurde. Der Vergleich mit den Anfangsdistanzen der Templatebilder zeigt eine deutliche Verringerung dieses Fehlermaßes durch die SVF-Registrierung.

Training des Gesamtframeworks

Während des Trainings vergleichen die CNNs des Gesamtframeworks, das Prediction Net und das Correction Net, ihre vorausgesagten Momentumpatches und Momentumkorrekturpatches mit den durch numerische Optimierung des SVF-Modells generierten Grundwahrheitsmomentumpatches. Hierzu wird der L1-Loss der Form

$$L^{L1} = L^{\mathrm{L1}}(\mathbf{y}, \mathbf{y}^{\star}) = \sum_{i} |y_i - y_i^{\star}| \tag{4.2}$$

berechnet, wobei y den vorausgesagten Momentumpatch und $y^{\star}$ den Grundwahrheitsmomentumpatch bezeichnet.
Die Entwicklung der durchschnittlichen Losswerte der Trainingsmenge zeigt im Training des Prediction Nets einen glatten und abfallenden Verlauf. Der durchschnittliche Losswert der Validierungsmenge fällt hingegen erst ab und steigt dann langsam wieder an. Dieses Verhalten deutet auf eine Überanpassung des CNN an die Trainingsmenge hin. Das CNN verbessert sich dann nur noch in den Voraussagen für die Trainingsmenge, verbessert aber seine Fähigkeit zur Generalisierung nicht oder verschlechtert diese sogar. Versuche dieses Verhalten zu ändern wurden durch die Wahl verschiedener Lernraten lr, die zusätzliche Regularisierung der Netzgewichte durch Erhöhung des Adam weight decay Parameters wd und den Einsatz eines Dropout-Moduls mit unterschiedlichen Dropoutraten p unternommen. Diese veränderten die Form des Losswert-Verlaufes für die Validierungsmenge jedoch nicht.

Zur Untersuchung des Einflusses des Validierungsloss-Verlaufs auf die Registrierungsergebnisse wurde das Prediction Net nach jeder Trainingsepoche für die Validierungsmenge ausgewertet. Das transformierte Templatebild wurde aus der so gene-

rierten Transformation erzeugt und die $\mathcal{D}^{\mathrm{NCC}}$-Distanz zum Referenzbild berechnet, da diese in der numerischen Optimierung des SVF-Modells minimiert wird. Der Verlauf dieser Distanzen zeigt einen Abfall, was der Vermutung einer Überanpassung widerspricht. Vielmehr ist zu vermuten, dass der L1-Loss des vorausgesagten Momentums kein direkter Indikator für die aus dem geglätteten Momentum generierte Transformation ist.

Auswertung des Gesamtframeworks

Für die Evaluierung der Registrierungsergebnisse der vorgestellten Methode wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den transformierten Templatebildern der Trainings-, Validierungs- und Testmenge und dem Referenzbild nach der Auswertung des Prediction Nets und nach der Auswertung des Correction Nets berechnet. Die Ergebnisse beider Datensätze zeigen eine deutliche Verringerung der Distanzen nach der Registrierung mit dem Gesamtframework.

Bei dem OAI Datensatz ist zudem eine deutliche Verbesserung der Ergebnisse des Prediction Nets durch das Correction Net erkennbar. Die Distanzen und die Variation der Distanzen sind nach dem Anwenden des Correction Net geringer als nach dem Anwenden des Prediction Nets. Die wenigen große Ausreißer sind dabei auf Bilder zurückzuführen, die Löcher in den Segmentierungen durch fortgeschrittenen Knorpelverlust enthalten. Diese hatten in der SVF-Registrierung ebenfalls zu Ausreißern geführt, sodass die Grundwahrheiten für das Training der CNNs nicht optimal waren.

Bei dem synthetischen Datensatz wird fast die gesamte Reduzierung der Distanzen, die durch die Registrierung mit dem Gesamtframework erzielt werden, durch das Prediction Net erreicht. Durch das Correction Net werden die Distanzen nur noch leicht verringert. Die Variation der Distanzen wird durch den Einsatz des Correction Nets jedoch reduziert. Bei den Distanzen, die nach der Registrierung mit dem Gesamtframework berechnet werden, gibt es für den synthetischen Datensatz keine Ausreißer.

Zusätzlich wurde für die Evaluation die $\mathcal{D}^{\mathrm{SSD}}$-Distanz zwischen den Transformationen, die durch die vorgestellte Methode erzeugt wurden, und den Transformationen, die durch numerische Optimierung des SVF-Modells generiert wurden, für beide Datensätze berechnet. Sie weisen in beiden Datensätzen Ausreißer auf, die nicht mit den Ausreißern der $\mathcal{D}^{\mathrm{SSD}}$-Distanzen der transformierten Templatebilder übereinstimmen. Dies liegt vermutlich daran, dass das Gesamtframework die SVF-Registrierung zur Erzeugung der Grundwahrheitsmomentumkorrekturen nutzt, anstatt diese als Differenz aus dem Grundwahrheitsmomentum und dem vom Prediction Net vorausgesagten Momentum zu berechnen. Dadurch können Fehler oder Ungenauigkeiten in den Grundwahrheitsmomenta korrigiert werden, sodass nicht zwingend ein Mo-

mentum vorausgesagt wird, das dem Grundwahrheitsmomentum möglichst ähnlich ist (vgl. Abb. 9). Dementsprechend können auch die aus den Momenta generierten Transformationen von der durch SVF-Registrierung generierten Transformation abweichen ohne zu größeren $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der transformierten Templatebilder zu führen.

Das zeitaufwändige Training des Gesamtframeworks kann im Vorfeld auf einem leistungsstarken Computer durchgeführt werden. Die Auswertung des Gesamtframeworks benötigt dann im Gegensatz zur iterativen Registrierung nur einen einzigen Inferenzschritt.
Die Rechenzeit für die Auswertung des Gesamtframeworks beträgt für die Bilder des OAI Datensatzes im Durchschnitt 0,183 Sekunden. Dies ist eine deutliche Verbesserung gegenüber der für die SVF-Registrierung durchschnittlich benötigten Rechenzeit im Minutenbereich.
Die Rechenzeit für die Auswertung des Gesamtframeworks beträgt für die Bilder des synthetischen Datensatzes im Durchschnitt 0,142 Sekunden. Auch dies ist eine deutliche Verbesserung gegenüber der für die SVF-Registrierung durchschnittlich benötigten Rechenzeit im Minutenbereich.

Vergleich mit der Quicksilver-Methode

Für den Vergleich mit der Quicksilver-Methode wurde eine Adaption der Quicksilver-Methode für das SVF-Modell, wie in 4.3.1 beschrieben, mit den Trainingsmengen der beiden Datensätze trainiert. Analog zur Vorgehensweise für die in dieser Arbeit vorgestellte Methode wurden die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen zwischen den transformierten Templatebildern der Trainings-, Validierungs- und Testmenge und dem Referenzbild nach der Auswertung des Prediction Nets und nach der Auswertung des Correction Nets berechnet. Da das Prediction Net in beiden Methoden übereinstimmt, ist die Betrachtung der Ergebnisse der Auswertung der jeweiligen Correction Net Versionen interessant.

Hier zeigt der Vergleich, dass das Quicksilver-Correction Net die durch das Prediction Net erzielten Distanzen weiter reduziert und die Varianz verringert. Die Unterschiede zwischen dem Prediction Net und dem Quicksilver-Correction Net sind jedoch deutlich geringer als im Falle der vorgestellten Methode. Dadurch werden durch die Registrierung mit der Quicksilver-Methode insgesamt höhere Distanzen zwischen den transformierten Templatebildern und dem Referenzbild beider Datensätze erzielt als durch die Registrierung mit der in dieser Arbeit vorgestellten Methode.

Die $\mathcal{D}^{\mathrm{SSD}}$-Distanzen zwischen den durch die Quicksilver-Methode generierten Transformationen und den durch die SVF-Registrierung generierten Transformationen

wurden ebenfalls berechnet. Sie weisen ein ähnliches Verteilungsmuster wie die $\mathcal{D}^{\text{SSD}}$-Distanzen der durch die vorgestellte Methode generierten Transformationen auf. Jedoch sind die Werte insgesamt höher.

Vergleich mit dem FlowNet

Für den Vergleich mit dem FlowNet-Ansatz wurde ein FlowNetC, wie in Kapitel 4.3.2 beschrieben, mit der Trainingsmenge trainiert. Auch hier wurden die $\mathcal{D}^{\text{NCC}}$-Distanzen zwischen den transformierten Templatebildern der Trainings-, Validierungs- und Testmenge und dem Referenzbild nach der Auswertung des FlowNet für beide Datensätze berechnet. Diese Distanzen sind für beide Datensätze nur geringfügig kleiner als die anfänglichen Distanzen der Templatebilder. Die durch das FlowNet generierten Transformationen weisen nur sehr kleine Deformationen auf. Dies spiegelt sich auch in den ebenfalls berechneten $\mathcal{D}^{\text{SSD}}$-Distanzen zwischen den durch das FlowNet generierten Transformationen und den durch die SVF-Registrierung generierten Transformationen wieder. Sie sind für beide Datensätze sehr groß und das Verteilungsmuster ähnelt jeweils dem Verteilungsmuster der Anfangsdistanzen der Templatebilder.

Wir vermuten, dass die schlechten Registrierungsergebnisse des FlowNet auf die sehr kleine Trainingsmenge und die verhältnismäßig kleine Bildgröße zurückzuführen ist. Nach dem Herunterskalieren des Deformationsfeldes hat dieses nur noch eine Größe von $2 \times 25 \times 40$ Pixeln und wird im Laufe des Trainings weiter heruntersklaliert. Für den OAI Datensatz enthält die Trainingsmenge 50, für den synthetischen Datensatz 90 Bildpaare mit zugehörigen Deformationsfeldern.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde eine Methode des maschinellen Lernens für Momentum-basierte Bildregistrierung vorgestellt, die aus einem zweischrittigen Gesamtframework mit vorausgehendem Prediction Net und anschließendem Correction Net besteht. Zur Evaluierung der vorgestellten Methode wurden sowohl Knieknorpelsegmentierungen des OAI-Datensatzes als auch Bilder eines generierten, synthetischen Datensatzes mit der vorgestellten Methode registriert. Zusätzlich wurden die verwendeten Datensätze mit der Quicksilver-Methode und dem FlowNet registriert, um eine Einordnung und den Vergleich der vorgestellten Methode zu ermöglichen.

Die experimentellen Ergebnisse haben gezeigt, dass die vorgestellte Methode die Bildregistrierung zeiteffizient durchführt. Während des Trainings lernt das Gesamtframework die Detektion charakteristischer Merkmale anhand der Trainingsmenge und ist hierdurch in der Lage, zu generalisieren und auch ungesehene Bildpaare der Testmenge in der Auswertung in nur einem Inferenzschritt und ohne iterative Anpassung zu registrieren.
Das zeitaufwendige Training wird im Vorfeld der Registrierung ein einziges Mal durchgeführt. Anschließend kann das trainierte Gesamtframework für die Registrierung ungesehener Bildpaare genutzt werden. Die hierfür benötigte Rechenzeit pro Bildpaar liegt im Dezisekundenbereich und stellt eine deutliche Verbesserung gegenüber der für die iterativen Optimierung des SVF-Modells benötigten Rechenzeit im Minutenbereich dar.

Zudem zeigten die experimentellen Ergebnisse, dass das Fehlermaß durch die Registrierung mit der vorgestellten Methode reduziert wird. Die transformierten Templatebilder beider verwendeten Datensätze weisen eine deutlich kleinere $\mathcal{D}^{\mathrm{NCC}}$-Distanz zum jeweiligen Templatebild auf als die anfänglichen Templatebilder. Hierbei zahlt sich die zweischrittige Struktur des Gesamtframeworks aus. Der Einsatz des Correction Nets verbesserte die Ergebnisse des vorausgegangenen Prediction Nets und reduzierte insbesondere die Varianz des Fehlermaßes.

Der Vergleich mit der State-of-the-Art Quicksilver-Methode hat gezeigt, dass durch die Registrierung mit dem vorgestellten Gesamtframework bessere Registrierungsergebnisse erzielt werden. Die $\mathcal{D}^{\mathrm{NCC}}$-Distanzen der transformierten Templatebilder sind kleiner und ebenso die $\mathcal{D}^{\mathrm{SSD}}$-Distanzen der generierten Transformationen.

Der Vergleich mit dem FlowNet-Ansatz hat gezeigt, dass die Registrierung mit dem FlowNet zu deutlich schlechteren Ergebnissen führt. Es werden nur sehr kleine Transformationen generiert, die nicht die gesamte Registrierung abbilden können. Dies ist auf die verhältnismäßig kleine Menge an Trainingsdaten zurückzuführen, die

in den verwendeten Datensätzen zur Verfügung steht. Diese Limitation des FlowNet-Ansatzes unterstreicht einen weiteren Vorteil des vorgestellten Gesamtframeworks. Durch die patchweise Voraussage des Momentums werden aus jedem Bildpaar der Trainingsmenge eine Vielzahl an Patchpaaren generiert. Daher kann das Gesamtframework bereits mit einer kleinen Trainingsmenge erfolgreich trainiert werden.

Die Ergebnisse zeigten auch, dass die Registrierung mittels numerischer Optimierung des SVF-Modells (abgesehen von der Rechenzeit) die besten Registrierungsergebnisse liefert. Eine Verbesserung dieser Ergebnisse durch die vorgestellte Methode war nicht zu erwarten, da das Training des Gesamtframeworks auf Grundwahrheiten basiert, die durch die numerische Optimierung generiert werden.
Die vorgestellte Methode ist jedoch sehr viel schneller, was für Anwendungen in der Klinik in Echtzeit von großem Vorteil sein kann. So kann eine Aufnahme zum Beispiel direkt während der Behandlung auf die Bilder der zurückliegenden Kontrollen registriert werden.

Ausblick

Im Folgenden werden offen gebliebene Fragestellungen und mögliche Erweiterungen der vorgestellten Methode diskutiert.

Durch die Ergebnisse der in Kapitel 4.2.2 durchgeführten Experimente lässt sich nicht erklären, warum der Validierungsloss-Verlauf während des Trainings nach einem anfänglichen Abfall wieder leicht ansteigt, während die $\mathcal{D}^{\mathrm{NCC}}$-Distanz der transformierten Templatebilder über das gesamte Training abnimmt. Die Frage nach dem Zusammenhang zwischen dem L1-Loss des vorausgesagten Momentums im Training und der $\mathcal{D}^{\mathrm{NCC}}$-Distanz des aus dem Momentum resultierenden transformierten Templatebildes ist trotz der guten Ergebnisse in der Praxis noch ungeklärt. Die Verwendung anderer Lossfunktionen sollte daher ebenfalls untersucht werden.
Intuitiv könnte hier beispielsweise die $\mathcal{D}^{\mathrm{NCC}}$-Distanz zwischen den transformierten Templatebildern und dem Referenzbild gewählt werden. Dies würde jedoch das Voraussagen des gesamten Momentums anstatt der Momentumpatches erfordern, wodurch der Speicherbedarf erhöht wird und der Vorteile der patchweisen Voraussage verloren gehen.

Des Weiteren wurde die Quicksilver-Methode von Yang et al. [YKSN17] für die Voraussage des initialen Momentums des LDDMM-Modells vorgestellt. In dieser Arbeit werden das vorgestellte Gesamtframework und die Quicksilver-Methode hingegen für die Voraussage des stationären Momentums des SVF-Modells verwendet.
Aus diesem Grund wäre eine Evaluierung der vorgestellten Methode für die LDDMM-Registrierung für einen umfangreicheren Vergleich der Methoden ein

sinnvoller nächster Schritt.

Für die verwendeten Datensätze konnten in der vorgestellten Methode durch den Einsatz des Correction Nets vergleichsweise große Verbesserungen der Voraussagen des Prediction Nets erzielt werden. Die Auswirkung des Einsatzes eines weiteren Correction Nets sollte daher untersucht werden. Auch die Ausweitung der vorgestellten Methode auf die gleichzeitige Registrierung der femoralen und tibialen Segmentierungen ist denkbar. Ebenso ist eine Erweiterung des Gesamtframeworks für 3D-Segmentierungen interessant, die durch die getrennten Enkoder für jede Dimension des vorausgesagten Momentums einfach zu realisieren sein sollte.

Abschließend ist eine Kombination denkbar, in der die vorgestellte Methode für die Initialisierung des Momentums in einem anschließenden numerischen Optimierungsverfahren genutzt wird. Auf diese Weise könnte die sehr schnelle Rechenzeit der vorgestellten Methode mit den sehr guten Registrierungsergebnissen der SVF-Registrierung vereint werden.

Literaturverzeichnis

[Arm66] Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

[Ash07] John Ashburner. A fast diffeomorphic image registration algorithm. *Neuroimage*, 38(1):95–113, 2007.

[BDTD+16] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[BMTY05] Mirza Faisal Beg, Michael I Miller, Alain Trouvé, and Laurent Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International journal of computer vision*, 61(2):139–157, 2005.

[BRV12] Martins Bruveris, Laurent Risser, and François-Xavier Vialard. Mixture of kernels and iterated semidirect product of diffeomorphisms groups. *Multiscale Modeling & Simulation*, 10(4):1344–1368, 2012.

[Bud19] Daniel Budelmann. Machine learning for registration in medical optical imaging. Master's thesis, Universität zu Lübeck, July 2019.

[CSJN15] Tian Cao, Nikhil Singh, Vladimir Jojic, and Marc Niethammer. Semi-coupled dictionary learning for deformation prediction. In *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, pages 691–694. IEEE, 2015.

[CW08] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[DFI+15] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[DGM98] Paul Dupuis, Ulf Grenander, and Michael I Miller. Variational problems on flows of diffeomorphisms for image matching. *Quarterly of applied mathematics*, pages 587–600, 1998.

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[FKSN17] Judith Fuchs, Ronny Kuhnert, and Christa Scheidt-Nave. 12-Monats-Prävalenz von Arthrose in Deutschland. In *Journal of Health Monitoring*, volume 2. Robert Koch-Institut, Epidemiologie und Gesundheitsberichterstattung, 2017.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[GFS16] Guido Gerig, James Fishbaugh, and Neda Sadeghi. Longitudinal modeling of appearance and shape and its potential for clinical use, 2016.

[GMW81] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Practical optimization*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, 1981.

[Had02] Jacques Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, pages 49–52, 1902.

[HPO08] Monica Hernandez, Xavier Pennec, and S Olmos. Comparing algorithms for diffeomorphic registration: Stationary lddmm and diffeomorphic demons. *Workshop on Mathematical Foundations of Computational Anatomy, (MICCAI)*, 10 2008.

[HS80] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. Technical report, Cambridge, MA, USA, 1980.

[HZN09] Gabriel L Hart, Christopher Zach, and Marc Niethammer. An optimal control approach for deformable registration. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–16. IEEE, 2009.

[HZRS15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[L+89] Yann LeCun et al. Generalization and network design strategies. In *Connectionism in perspective*, volume 19. Citeseer, 1989.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[LKB+17] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[Mod04] Jan Modersitzki. *Numerical Methods for Image Registration*. Oxford University Press, 2004.

[Mod09] Jan Modersitzki. *FAIR: Flexible Algorithms for Image Registration*. SIAM, Philadelphia, 2009.

[NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[NW06] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[PJ92] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

[Pol18] Thomas Polzin. *Large Deformation Diffeomorphic Metric Mappings – Theory, Numerics, and Applications*. PhD thesis, Universität zu Lübeck, 2018.

[PSN08] Charles G Peterfy, E Schneider, and M Nevitt. The osteoarthritis initiative: report on the design rationale for the magnetic resonance imaging protocol for the knee. *Osteoarthritis and cartilage*, 16(12):1433–1441, 2008.

[PSS+16] Akshay Pai, Stefan Sommer, Lauge Sørensen, Sune Darkner, Jon Sporring, and Mads Nielsen. Kernel bundle diffeomorphic image registration using stationary velocity fields and wendland basis functions. *IEEE Transactions on Medical Imaging*, 35(6):1369–1380, June 2016.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[Ros58] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

[Rud16] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[Sai88] Saburou Saitoh. Theory of reproducing kernels and its applications. *Longman Scientific & Technical*, 1988.

[SHJF13] Nikhil Singh, Jacob Hinkle, Sarang Joshi, and P Thomas Fletcher. A vector momenta formulation of diffeomorphisms for improved geodesic regression and atlas construction. In *2013 IEEE 10th International Symposium on Biomedical Imaging*, pages 1219–1222. IEEE, 2013.

[VRRC12] François-Xavier Vialard, Laurent Risser, Daniel Rueckert, and Colin J Cotter. Diffeomorphic 3d image registration via geodesic shooting using an efficient adjoint calculation. *International Journal of Computer Vision*, 97(2):229–241, 2012.

[WKS+15] Qian Wang, Minjeong Kim, Yonghong Shi, Guorong Wu, Dinggang Shen, Alzheimer's Disease Neuroimaging Initiative, et al. Predict brain mr image registration via sparse learning of appearance and transformation. *Medical image analysis*, 20(1):61–75, 2015.

[YKSN17] Xiao Yang, Roland Kwitt, Martin Styner, and Marc Niethammer. Quicksilver: Fast predictive image registration – a deep learning approach. *NeuroImage*, 158, 07/2017 2017.

[YLRJ15] Xianfeng Yang, Yonghui Li, David Reutens, and Tianzi Jiang. Diffeomorphic metric landmark mapping using stationary velocity field parameterization. *International Journal of Computer Vision*, 115(2):69–86, Nov 2015.

[You10] Laurent Younes. *Shapes and diffeomorphisms*, volume 171. Springer, 2010.