UNIVERSITÄT ZU LÜBECK
INSTITUTE OF MATHEMATICS AND
IMAGE COMPUTING

Merkmalsbasierte Bildregistrierung mit SIFT

Feature-Based Image Registration with SIFT

Bachelorarbeit

im Rahmen des Studiengangs
Computational Life Science
der Universität zu Lübeck

vorgelegt von
Christian Strauß

ausgegeben und betreut von
Prof. Dr. rer. nat. Jan Modersitzki
Institute of Mathematics and Image Computing

mit Unterstützung von
Dr. rer. nat. Stefan Heldmann
Institute of Mathematics and Image Computing

Lübeck, den 31. Mai 2013

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 31. Mai 2013

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Im Bereich der Bildverarbeitung stellt die Bildregistrierung ein wichtiges Teilgebiet dar, in dem es darum geht, zwei Bilder des gleichen Objektes aus verschiedenen Blickwinkeln, mit verschiedener Beleuchtung oder aus unterschiedlichen Quellen (zum Beispiel MRT und Röntgen) möglichst gut in Übereinstimmung zu bringen. Das Ziel ist, eins der Bilder so zu transformieren, dass es im besten Fall exakt auf dem anderen liegt.

Gerade im medizinischen Bereich kommt es auf ein höchst genaues Vorgehen an. Daher ist es sinnvoll, auch Verfahren, die für anderen Zwecke entwickelt wurden, für eine mögliche Nutzung in der Registrierung zu testen. In dieser Arbeit wird deshalb das SIFT-Verfahren (Scale Invariant Feature Transform), das ursprünglich zur Objekterkennung in Bildern entwickelt wurde, für die Registrierung angepasst.

1.2 Bildregistrierung

Das Ziel der Registrierung als Teilgebiet der Bildverarbeitung ist, eine Korrespondenz zwischen verschiedenen Bildern vom gleichen Objekt herzustellen. Dabei hat man in der Regel Bilder, die aus verschiedenen Quellen stammen und die darauf abgebildeten Objekte in unterschiedlicher Orientierung oder Beleuchtung vorliegen.

Als Bild bezeichnet man eine Abbildung $(x, y) \mapsto T(x, y)$, die einem Ort (x, y) eine Intensität $T(x, y)$ zuordnet. Bei der Registrierung geht man von einem bekannten Bild, dem Referenzbild, aus; im Folgenden bezeichnet mit R. Dieses Bild ist der Standard, mit dem alle weiteren Bilder verglichen werden, es bleibt dabei stets unverändert. Die anderen Bilder werden als Template T bezeichnet und sollen mit R in Einklang gebracht werden. Im Folgenden betrachten wir ausschließlich zweidimensionale Bilder zur einfacheren Darstellung.

Das Ziel der Registrierung ist es also, eine möglichst optimale zweidimensionale Transformation $\phi : \mathbb{R}^2 \to \mathbb{R}^2$ zu finden, die die Koordinaten der Punkte in T so verändert, dass

$$R(x, y) = T(\phi(x, y))$$

gilt.

Dazu müssen Gemeinsamkeiten in beiden Bildern gefunden werden, die Punkte in T eindeutig mit Punkten in R verknüpfen. Um dies zu erreichen, gibt es verschiedene Ansätze der Registrierung. Einen Überblick dazu findet sich zum Beispiel in [2], [11], [12] und [13].

In dieser Arbeit soll das SIFT-Verfahren darauf getestet werden, ob es die Aufgabe der Registrierung zufrieden stellend bewältigen kann. Im Jahr 1999 entwickelte David G. Lowe, Professor der University of British Columbia, einen neuen Algorithmus zur Objekterkennung in Bildern. Im Gegensatz zur Registrierung, in der Bilder eine feste Auflösung und Größe besitzen, geht es darin um das Problem, ein vorher gespeichertes Objekt in einem anderen Bild wiederzufinden. Es sind also neben der Ausrichtung des Bildes auch die Größe und Auflösung des Objektes im Bild unbekannt. Lowes Algorithmus hatte im Vergleich zu anderen Ansätzen vor allem den Vorteil, dass er weitestgehend unabhängig von der Bildauflösung, der Rotation oder der Beleuchtung war, was ihm den Namen Scale Invariant Feature Transform, kurz SIFT, gab [9].

Um dies zu erreichen, nutzt das SIFT-Verfahren korrespondierende Punkte in R und T. Aus den Bildern werden besonders prägnante Punkte (Schlüsselpunkte) herausgesucht und in Form eines Deskriptors so gespeichert, sodass sie auch nach Änderungen (zum Beispiel von Position und Auflösung) wiedererkannt werden. Die Objekte können anschließend in einem Bild dadurch gefunden werden, dass die Schlüsselpunkte mit denen des Referenzobjektes korrenspondieren.

In den darauffolgenden Jahren wurde das Verfahren von Lowe noch weiter entwickelt, um beispielsweise die Auswahl dieser Schlüsselpunkte zu verbessern. Im Jahr 2004 folgte dann ein neues Paper mit dieser verbesserten Methode, die auch generell den Ablauf des Verfahrens sehr viel detaillierter betrachtete [10].

Darauf aufbauend gab es in den letzten Jahren noch weitere Verbesserungen, hauptsächlich von W. Cheung und G. Hamarneh, die beispielsweise das Verfahren vom 2-dimensionalen Bild ins n-dimensionale erweiterten und auf medizinische Bilderkennung anwendeten [5, 6].

Eine Verbesserung der Methode, vor allem die Laufzeit betreffend, stellt der SURF-Algorithmus (Speeded-Up Robust Features) dar, der 2006 von Herbert Bay et al. vorgestellt wurde [1].

In dieser Arbeit wird Lowes Verfahren modifiziert, um es besser an die Aufgabe der Registrierung anzupassen. Da von weniger möglichen Änderungen zwischen den Bildern als bei der Objektfindung ausgegangen werden kann, können einige Berechnungen vereinfacht werden. Allerdings muss die finale Transformation ergänzt werden, die nach dem Finden der korrespondierenden Punkte das Bild T an das Bild R angleicht.

Dazu wird zunächst in Kapitel 2 der Ablauf des SIFT-Algorithmus in einzelnen Schritten beschrieben. In Kapitel 3 wird dann betrachtet, wie man die durch SIFT bestimmten Merkmale aus zwei Bildern miteinander vergleichen kann und daraus die Transformation zur Registrierung in Kapitel 4 bestimmt wird. Die Funktionalität des Algorithmus wird in Kapitel 5 an verschiedenen Bildern aus unterschiedlichen Quellen getestet. Ein abschließendes Fazit wird in Kapitel 6 gezogen.

2 SIFT

Im Folgenden soll der SIFT-Algorithmus beschrieben werden. Dieser lässt sich grob in mehrere Teilschritte unterteilen:

1. Erstellen einer Difference-of-Gauß-Pyramide, die das Bild in verschiedenen Auflösungen und Bildgrößen darstellt. Dies ist die Grundlage der nächsten Schritte.
2. In diesen Bildern werden lokale Extrema herausgesucht, die als Schlüsselpunkte genutzt werden sollen.
3. Die Anzahl der Schlüsselpunkte wird durch weitere Anforderungen, denen sie genügen müssen, verringert.
4. Aus den verbliebenen Schlüsselpunkten werden Deskriptoren bestimmt. Dabei wird die Umgebung des jeweiligen Punktes in die Berechnung mit einbezogen.

Um zwei Bilder miteinander zu registrieren, müssen diese Schritte für beide Bilder durchlaufen werden.

2.1 Difference-Of-Gauß-Pyramide

Das Hauptmerkmal des SIFT-Verfahrens ist, wie der Name suggeriert, die Skalenunabhängigkeit. Das heißt, dass zwei Bilder unabhängig von ihrer Größe oder Auflösung miteinander verglichen werden können.

Um diese Unabhängigkeit zu erreichen, wird eine Difference-of-Gauß-Pyramide benötigt. Für dessen Bestimmung wird zuerst eine Gauß-Pyramide durch wiederholtes Filtern des Bildes erstellt, sodass man die Bilder in vielen verschiedenen Auflösungen zur Verfügung hat. Danach wird die Difference-of-Gauß-Pyramide als Differenz der verschiedenen Stufen der Pyramide berechnet [4].

Im Gegensatz zur Beschreibung der kontinuierlichen Bilder aus der Einleitung werden hier diskrete zweidimensionale Bilder betrachtet. Nachfolgend verstehen wir unter einem Bild eine Matrix $I \in \mathbb{R}^{m \times n}$ von Bildpunkten (Pixeln), des Weiteren wird ein Pixel an Position (x, y) in I mit $I(x, y)$ bezeichnet.

Beim Filtern wird das Bild mit einer anderen Funktion bzw. Matrix gefaltet; meist mit dem Ziel, Rauschen zu entfernen und / oder Kanten hervorzuheben. Die Faltung $f * g$ von zwei Funktionen ist dabei allgemein definiert als

$$f(x) * g(x) := \int_{\mathbb{R}^n} f(t)g(x-t)dt.$$

Da wir in diesem Fall keine kontinuierliche Funktion nutzen, sondern eine diskrete Bildmatrix, wird eine diskrete Faltung verwendet:

$$G(x,y) * I(x,y) := \sum_k \sum_l G(k,l)I(x-k, y-l).$$

I ist dabei das zu faltende Bild und G der sogenannte Faltungskern. In diesem Fall nutzt man dabei einen Gaußkern, der gut zur Glättung von Kanten geeignet ist. Er ist definiert als

$$G(x,y,k^s\sigma) = \frac{1}{2\pi k^s \sigma^2} e^{-(x^2+y^2)/2k^s\sigma^2}$$

wobei σ die Standardabweichung ist und k^s ein Faktor, über den σ variiert wird (siehe Abb. 2.1).

Der Kern $G(x,y,k^s\sigma)$ wird nun mit zunehmendem s wiederholt mit dem Bild $I(x,y)$ gefaltet, sodass das Bild mit wachsendem s in jeder Faltung immer mehr geglättet wird und gut definierte Kanten hervorgehoben werden (siehe Abb. 2.2).

$$L(x,y,s) = G(x,y,k^s\sigma) * I(x,y)$$

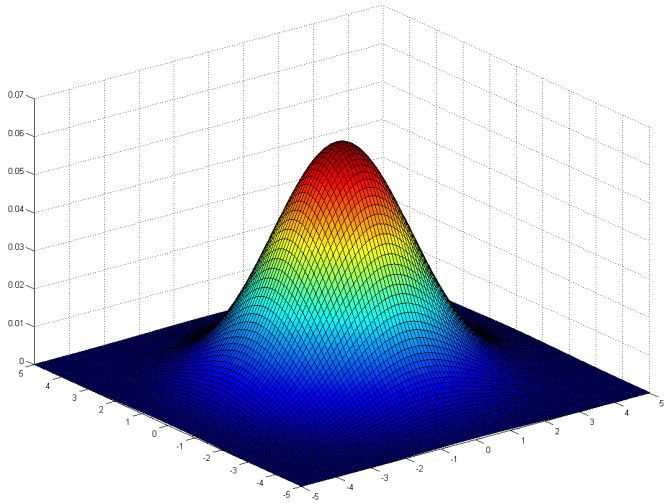


Abbildung 2.1: Visualisierung eines zwei-dimensionalen Gaussfilters

$G * I$ bezeichnet hierbei die Faltung des Gaußkerns G mit dem Bild I. Die Standardabweichung σ sowie der Multiplikator k werden dabei fest gewählt, lediglich $s \in \mathbb{N}$ ändert sich für eine größere Glättung. k^s sorgt dafür, dass die Standardabweichung immer um den gleichen Multiplikator k verändert wird. Die auf diese Art entstandene Folge von gefilterten Bildern wird als Gauß-Pyramide bezeichnet.

Um daraus die Difference-of-Gauß-Pyramide (DoG-Pyramide) zu berechnen, wird die Differenz von je zwei aufeinander folgenden Bildern der Gauß-Pyramide gebildet:

$$\begin{aligned} D(x,y,s) &= (G(x,y,k^s\sigma) - G(x,y,k^{s-1}\sigma)) * I(x,y) \\ &= L(x,y,k^s\sigma) - L(x,y,k^{s-1}\sigma). \end{aligned}$$

Da sich Flächen beim Filtern des Bildes nicht wesentlich ändern, besitzt das Differenzbild an den meisten Stellen sehr kleine Werte. Nur an den Stellen, an denen die Filterung zu einer Änderung führt (also an Stellen, an denen Kanten sind), zeigen sich auch im Differenzbild große Werte. Auf diese Art erhält man Bilder, die auf wenige, jedoch sehr prägnante, Merkmale reduziert sind (siehe Abb. 2.2).

Nachdem das Bild in der Originalgröße mehrfach gefiltert wurde, wird die Bildgröße halbiert. Dazu wird das am schwächsten gefilterte Bild der aktuellen Auflösung gewählt und jeder zweite Pixel entfernt:

$$\begin{aligned} \text{downsample} &: \mathbb{R}^{m\times n} \to \mathbb{R}^{\lfloor m/2\rfloor \times \lfloor n/2\rfloor} \\ \text{downsample}(L)(x,y) &:= L(2x,2y) \quad , \ 1 \le x \le \lfloor m/2\rfloor, 1 \le y \le \lfloor n/2\rfloor. \end{aligned}$$

Die Filterung sorgt dafür, dass die Auflösung entsprechend der Pixelanzahl halbiert wird. In dieser Größe werden alle Schritte wiederholt, also wie vorher DoG-Bilder gebildet und zum Schluss weiter verkleinert. Dies wird spätestens dann abgebrochen, wenn eine Bildseite weniger als 16 Pixel hat, da dies für die späteren Schritte erforderlich ist. Ein Satz von DoG-Bildern mit gleicher Auflösung wird dabei als Oktave bezeichnet.

Der gesamte Prozess der Erstellung der Gauß- und DoG-Pyramiden lässt sich auch übersichtlicher rekursiv darstellen:

$$\begin{aligned} L^{(s,l)} &= G_{k^{s-1}\sigma} * L^{(0,l)} \\ \text{mit } G_{k^{s-1}\sigma} &:= G(x,y,k^{s-1}\sigma) \\ L^{(0,l)} &= \text{downsample}(L^{(1,l-1)}) \\ L^{(0,0)} &= I \\ D^{(s,l)} &= L^{(s+1,l)} - L^{(s,l)} \\ l \in \mathbb{N}_0 &\quad , \quad s \in \mathbb{N} \end{aligned}$$

Dabei bezeichnet der Index s den Exponenten des Multiplikators k^s der Filterung, mit s wächst also die Glättung. l bestimmt die Oktave, $l = 0$ ist die Originalgröße des Bildes I, $l = 1$ die Oktave der einmal halbierte Bildgröße etc.

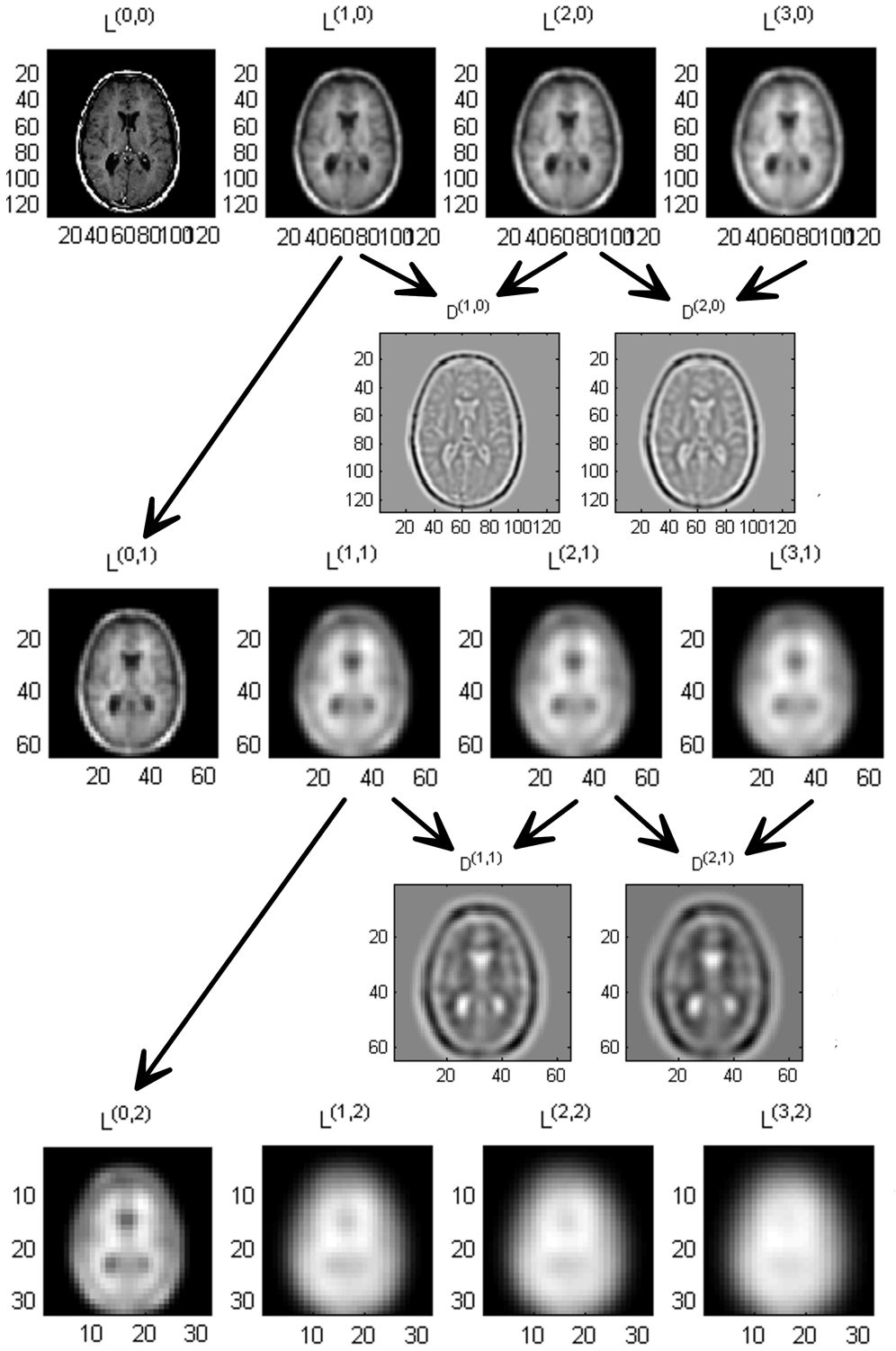


Abbildung 2.2: Beispielentwicklung der Gauß- und Difference-of-Gauß-Pyramiden für drei Oktaven. Es handelt sich um ein MRI-Bild einer Gehirnschicht, die hier und in den folgenden Bildern als Beispiel dient.

Was die Anzahl der DoG-Bilder einer Oktave angeht, so wird von Lowe empfohlen, diese abhängig von der Wahl von k (dem Faktor, um den σ in jeder Faltung erhöht wird) zu machen. Wenn $k = 2^{1/n}$ ist, dann werden $n + 3$ gefilterte Bilder berechnet. Bei stärker gefilterten Bildern sind die Konturen häufig zu verschwommen und ungenau, um sie noch gut auszuwerten.

Die Wahl von σ selbst ist natürlich ebenfalls wichtig. Lowe empfiehlt, $\sigma = 1.6$ zu wählen, dieser Wert hat sich auch in unseren Tests als sehr praktikabel bestätigt. Genauere Erläuterungen zur Wahl von k und σ siehe [10], Seite 94f.

2.2 Kandidaten für Schlüsselpunkte finden

Da alle folgenden Berechnungen immer nur innerhalb einer Oktave stattfinden, lässt sich die Notation vereinfachen, indem ab hier ausschließlich die erste Oktave betrachtet wird. Alle Gleichungen lassen sich analog auf die restlichen Oktaven übertragen:

$$D(x, y, s) = D^{(s,l)}(x, y) \quad \text{für } l \text{ beliebig, aber fest.}$$

Im Folgenden müssen nun einzelne Punkte im Bild gefunden werden, die dieses Bild gut charakterisieren, sogenannte Schlüsselpunkte. Dabei sollte es sich um Punkte handeln, die sich stark von ihrer Umgebung abheben. In der Regel sind das die Stellen im Bild, an denen eine Fläche in eine andere übergeht, also Kanten oder Ecken von Objekten. Oft wird auch das Rauschen in eigentlich einheitlichen Flächen als so ein Übergang erkannt. Deshalb glättet man schon vor dem erstem DoG-Bild einmal, um diese weitestgehend zu entfernen.

Solche Punkte mit starker Änderung der Umgebung findet man in den DoG-Bildern als lokale Extremstellen, also als lokales Maximum oder Minimum. Hierbei vergleicht man aber jeden Punkt nicht nur mit den ihn umgebenen 8 Punkten innerhalb des Bildes, sondern, wenn man auf diese Art ein Extremum gefunden hat, auch noch mit den 9 Punkten des DoG-Bildes auf der Skala davor und danach an derselben Stelle (siehe Abb. 2.3). Das heißt, ein Punkt $\kappa = (x, y, s)$ ist genau dann ein Kandidat für einen Schlüsselpunkt, falls gilt:

$$\begin{aligned} D(x, y, s) &< D(x + \Delta x, y + \Delta y, s + \Delta s) \\ \text{mit } (\Delta x, \Delta y, \Delta s) &\in \{-1, 0, 1\}^3 \setminus (0, 0, 0)\}. \end{aligned}$$

Die Menge aller Schlüsselpunktkandidaten wird im Folgenden als $\mathcal{K}_0$ bezeichnet und sie ist definiert als

$$\begin{aligned} \mathcal{K}_0 = \{\kappa = (x, y, s) \ : \ & D(x, y, s) < D(x + \Delta x, y + \Delta y, s + \Delta s) \\ & \text{mit } (\Delta x, \Delta y, \Delta s) \in \{-1, 0, 1\}^3 \setminus (0, 0, 0)\}. \end{aligned}$$

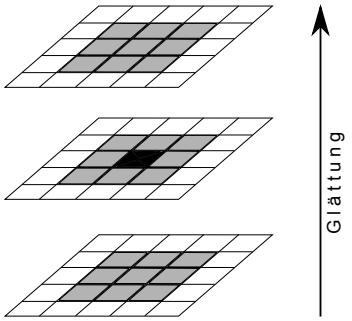


Abbildung 2.3: Skizze zur Veranschaulichung des Extremafindens: Der schwarze Pixel in der Mitte wurde als lokales Extremum ermittelt und muss größer / kleiner sein als alle grauen Pixel.

Erst wenn der beobachte Punkt größer oder kleiner als alle 26 ihn umgebenden Punkte ist, wird er als Kandidat für einen Schlüsselpunkt gemerkt (siehe auch Abb. 2.4 und 2.5).

In unserem Algorithmus wurde darauf verzichtet, im ersten DoG-Bild nach Extrema zu suchen, da hier nur in eine Richtung verglichen werden kann; ebenso im letzten Bild einer Oktave. Dies führte oft zu einer größeren Menge Punkte als in den anderen Oktaven, welche jedoch im folgenden Schritt als ungeeignet aussortiert wurden.

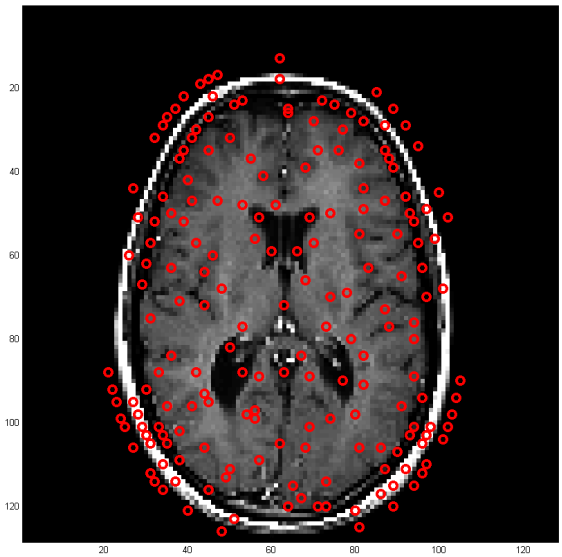


Abbildung 2.4: Realistisches Beispiel des MRI-Bildes. Die roten Kreise markieren die gefundenen Extrema

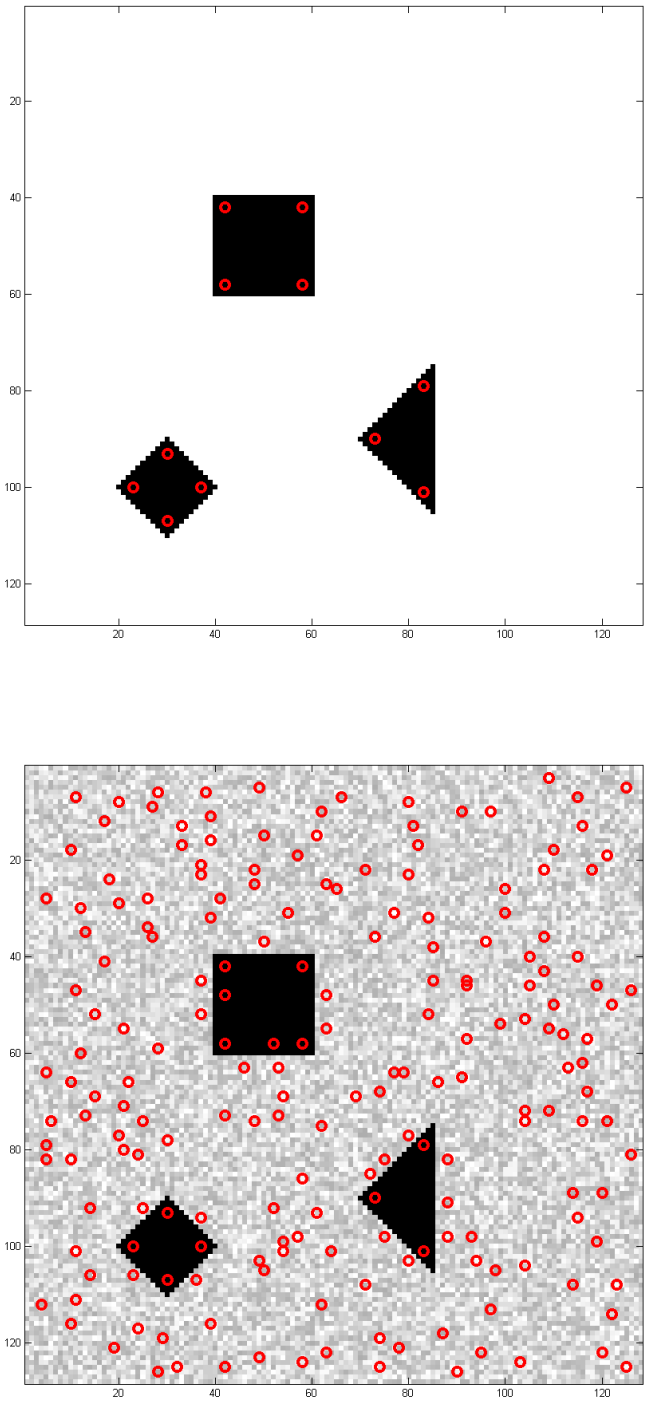


Abbildung 2.5: Oben: Stark vereinfachtes, akademisches Beispiel, um zu zeigen, welche Punkte der Algorithmus auswählt. Unten: Das selbe Bild mit vorher verrauschtem Hintergrund, es werden sehr viel mehr lokale Extrema gefunden.

2.3 Auswahl der finalen Schlüsselpunkte

Wie unter anderem in Abbildung 2.5 sehr gut zu sehen ist, kommt es vor, dass einige der gefundenen Extrema als Schlüsselpunkt ungeeignet sind. Hier liegt das an den lokalen Extrema, die durch das Rauschen im eigentlich weißen Hintergrund entstehen. Um dies zu beheben, werden die zuvor gefundenen Kandidaten für die Schlüsselpunkte einzeln auf weitere Kriterien überprüft: Sie werden genauer mit ihrer Umgebung verglichen, um die Punkte auszusortieren, deren Kontrast zu gering ist (wie die Extrema, die beim Rauschen entstehen) oder die auf einer langen, geraden Kante liegen. Da diese überall gleich aussieht, ist sie als Herausstellungsmerkmal ungeeignet.

Um den Kontrast eines Schlüsselpunktkandidaten zu seinen Nachbarn zu bestimmen, nutzt man die Taylorentwicklung zweiten Grades und bildet sie über alle drei Dimensionen der DoG-Bilder (also auch über die Glättung) mit dem zu untersuchenden Schlüsselpunkt $\kappa_0 = (x_0, y_0, s_0) \in \mathcal{K}_0$ als Entwicklungspunkt. Prinzipiell wäre es möglich, direkt $D(\kappa)$ anzuschauen, aber nach Brown und Lowe [3] ist dieser Ansatz besser geeignet (nähere Erklärung zu dieser Methode siehe auch [10], S.97f):

$$\begin{aligned} \hat{D}(\kappa) &= D(\kappa_0) + \nabla D(\kappa_0)(\kappa - \kappa_0) + \frac{1}{2}(\kappa - \kappa_0)^T \nabla^2 D(\kappa_0)(\kappa - \kappa_0) \\ \kappa &= (x, y, s) \end{aligned}$$

mit

$$\begin{aligned} \nabla D &= \begin{bmatrix} D_x \\ D_y \\ D_s \end{bmatrix} \\ \nabla^2 D &= \begin{bmatrix} D_{xx} & D_{xy} & D_{xs} \\ D_{xy} & D_{yy} & D_{ys} \\ D_{sx} & D_{sy} & D_{ss} \end{bmatrix} \end{aligned}$$

Da D nur diskret vorliegt, werden hier Ableitungen mit finiten Differenzen approximiert:

$$\begin{aligned} D_x &\approx (D(x_0 + 1, y_0, s_0) - D(x_0 - 1, y_0, s_0))/2 \\ D_y &\approx (D(x_0, y_0 + 1, s_0) - D(x_0, y_0 - 1, s_0))/2 \\ D_z &\approx (D(x_0, y_0, s_0 + 1) - D(x_0, y_0, s_0 - 1))/2 \\ D_{xx} &\approx (D(x_0 + 2, y_0, s_0) + D(x_0 - 2, y_0, s_0) - 2D(x_0, y_0, s_0))/4 \\ D_{xy} &\approx (D(x_0 + 1, y_0 + 1, s_0) + D(x_0 - 1, y_0 - 1, s_0) \\ &\quad -D(x_0 + 1, y_0 - 1, s_0) - D(x_0 - 1, y_0 + 1, s_0))/4 \end{aligned}$$

Die restlichen 2. Ableitungen werden analog dazu bestimmt.

Wenn man in dieser Funktion nun nach dem Extremum sucht, erhält man meist eine Abweichung vom ursprünglich bestimmten Extremum, dem Offset h. Sollte der vorher bestimmte Wert exakt stimmen, wäre $h = 0$:

$$\begin{aligned} \hat{D}(\kappa_0 + h) &= D(\kappa_0) + \nabla D(\kappa_0)h + \frac{1}{2}h^T \nabla^2 D(\kappa_0)h \\ \Rightarrow \nabla\hat{D}(\kappa_0 + h) &= \nabla D(\kappa_0) + \nabla^2 D(\kappa_0)h \stackrel{!}{=} 0 \\ \Leftrightarrow h &= -\nabla^2 D(\kappa_0)^{-1} h \cdot \nabla D(\kappa_0). \end{aligned}$$

Durch Einsetzen in die ursprüngliche Taylorentwicklung erhalten wir dann für $\hat{D}(\kappa_0 + h)$:

$$\begin{aligned} \hat{D}(\kappa_0 + h) &= D(\kappa_0) - \nabla D(\kappa_0)^T \nabla^2 D(\kappa_0)^{-1} \nabla D(\kappa_0) \\ &\quad + \frac{1}{2}\nabla D(\kappa_0)^T \nabla^2 D(\kappa_0)^{-1} \nabla D(\kappa_0) \\ &= D(\kappa_0) - \frac{1}{2}\nabla D(\kappa_0)^T \nabla^2 D(\kappa_0)^{-1} \nabla D(\kappa_0) \\ &= D(\kappa_0) + \frac{1}{2}\nabla D(\kappa_0)^T h. \end{aligned}$$

Mit diesem Wert kann man jetzt eine Aussage über den Kontrast um den Entwicklungspunkt treffen: Da die ganze Berechnung auf den DoG-Bildern vorgenommen wurde, bedeutet ein zu kleiner Wert, dass der Kontrast nur sehr gering und das Extremum nicht als Schlüsselpunkt geeignet ist. Lowe gab dafür den Wert 0.03 an bei Bildpunkten im Bereich $[0, 1]$. Da wir im Vergleich zu Lowe auf die Einschränkung der Bildpunkte auf diesen Bereich verzichtet haben, wird ein Wert von 3 % des gesamten Bildbereiches als Threshold genutzt:

$$\textbf{1. Kriterium: } \hat{D}(\kappa_0 + h) > 0.03 \cdot \max\{D(x, y, s_0) : x \in \{1, \ldots, m\}, y \in \{1, \ldots, n\}\}$$

Das zweite Ausschlusskriterium für die Schlüsselpunkte beruht auf dem Problem, dass die Difference-of-Gauß-Funktion nicht nur auf Ecken, sondern auch auf Kanten sehr stark reagiert. Da eine Kante an jeder Stelle im Prinzip gleich aussieht, ist dies kein günstiger Schlüsselpunkt.

Eine Kante ist leicht dadurch erkennbar, dass sie in eine Richtung eine sehr starke Krümmung (der Verlauf des Anstieges) aufweist, aber rechtwinklig dazu nahezu gar keine Krümmung. Nur bei einer starken Krümmung in beiden Richtungen liegt eine Ecke im Bild vor. Die Krümmung kann man über die Berechnung der Hessematrix H ermitteln, in der die 2. Ableitungen, also der Verlauf des Anstieges, angegeben

werden. Die Ableitung wird hierbei erneut über die Differenzen der Bildpunkte gebildet, wie bereits bei der Taylorentwicklung verwendet. Hier werden jedoch im Gegensatz zu vorher ausschließlich die x- und y-Dimension des Bildes betrachtet, eine Berechnung über s ist nicht notwendig.

$$H(\kappa) = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

Das Verhältnis der Eigenwerte der Hessematrix ist hierbei proportional zum Verhältnis der Hauptkrümmungen. Somit ist es nicht nötig, diese exakt auszurechnen, da man nur daran interessiert ist, ob die beiden Werte nah beieinander liegen, oder einer sehr viel größer ist als der andere. Über die Determinante der Hessematrix errechnet man dann das Produkt der beiden Eigenwerte und über die Spur die Summe:

$$\begin{aligned} \mathrm{Tr}(H(\kappa)) &= D_{xx} + D_{yy} = \alpha + \beta \\ \mathrm{Det}(H(\kappa)) &= D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta \end{aligned}$$

Mit r als Verhältnis des größeren Eigenwertes zum kleineren, sodass $\alpha = r\beta$, kann man nun die Spur und die Determinante in eine Beziehung setzen. Daraus folgt, dass die exakten Eigenwerte nicht mehr gebraucht werden:

$$\frac{\mathrm{Tr}(H(\kappa))^2}{\mathrm{Det}(H(\kappa))} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}$$

Das Verhältnis der Eigenwerte ist genau dann sehr klein, wenn die Krümmung am Schlüsselpunkt in beide Richtungen sehr ähnlich ist. Daraus kann man das Ausschlusskriterium bestimmen:

$$\textbf{2. Kriterium: } \frac{\mathrm{Tr}(H(\kappa))^2}{\mathrm{Det}(H(\kappa))} < \frac{(r+1)^2}{r}$$

r ist dabei variabel, aber fest gewählt und bestimmt den Schwellwert. Wir nutzen dafür $r = 10$, sodass sich als Schwellwert 12.1 ergibt. Zusammenfassend erhält man für die Menge der Schlüsselpunkte (vergleiche Abb. 2.6, 2.7 und 2.8):

$$\mathcal{K} = \left\{ \kappa \in \mathcal{K}_0 : \hat{D}(\kappa + h) > 0.03 \wedge \frac{\mathrm{Tr}(H(\kappa))^2}{\mathrm{Det}(H(\kappa))} < \frac{(r+1)^2}{r} \right\}.$$

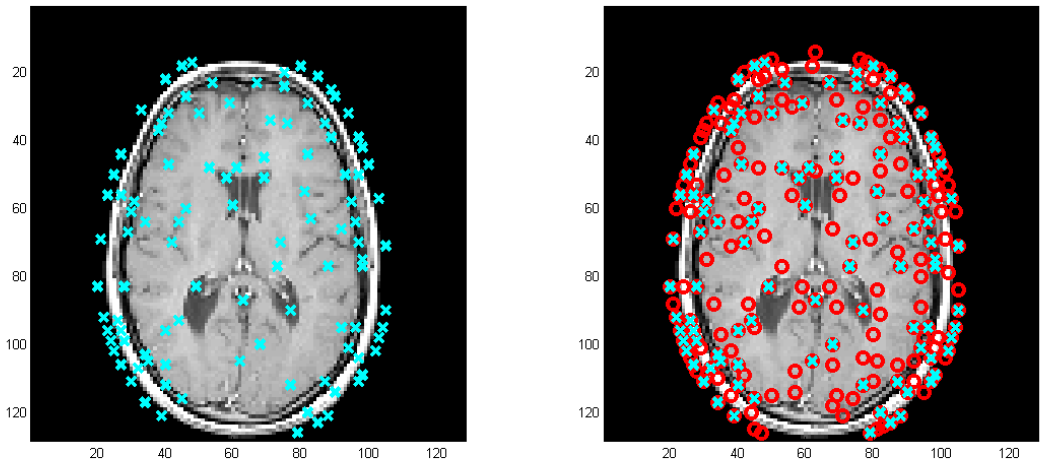


Abbildung 2.6: Im ersten Bild sind die nach den geschilderten Auswahlverfahren übrig bleibenden Extrema zu sehen. Rechts ein Vergleich der ursprünglichen Extrema mit den jetzt übrig gebliebenen. Es ist gut erkennbar, dass vor allem Extrema auf gleich bleibenden Flächen weggefallen sind.

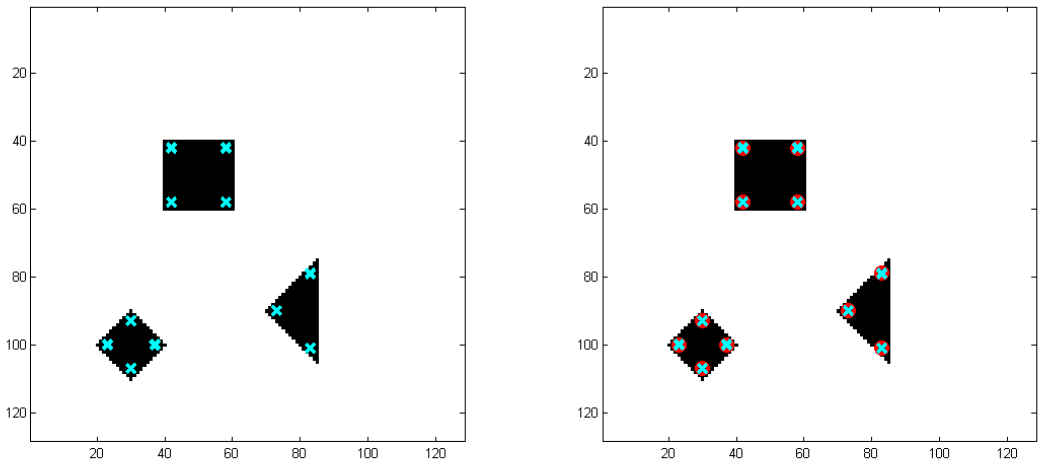


Abbildung 2.7: Der selbe Vergleich im akademischen Beispiel. Da schon vorher nur Eckpunkte ausgewählt waren, werden keine gelöscht. Dafür sieht man, dass die Auswahl keine wichtigen Punkte entfernt, alle lokalen Extrema sind auch jetzt noch Schlüsselpunkte.

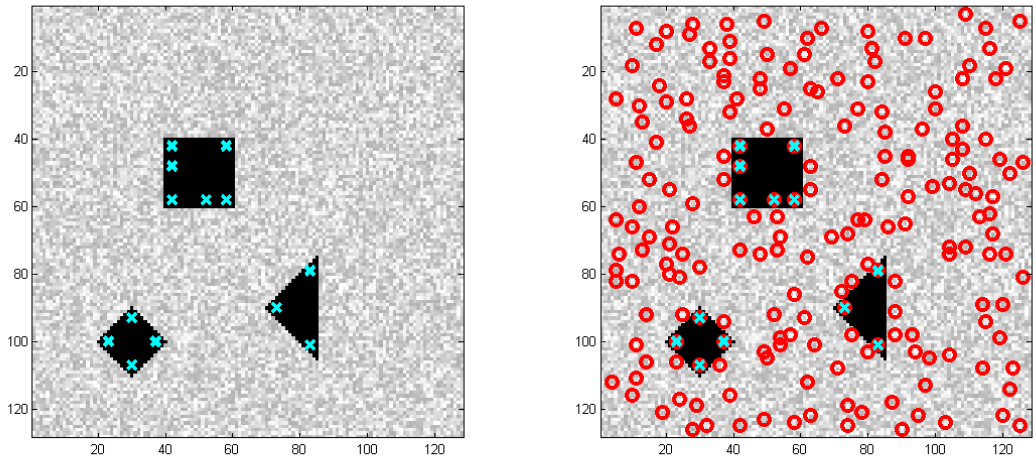


Abbildung 2.8: Im verrauschten Beispiel werden alle Extrema des eigentlich weißen Bereiches entfernt. Es bleiben wie vorher alle Eckpunkte, aber auch einige weitere Schlüsselpunkte, die es im unverrauschten Beispiel nicht gibt. Diese entstehen, da der Bereich um diese Punkte kontrastreicher ist als vorher.

2.4 Bestimmung der Hauptrichtung

Nachdem die Auswahl der Extrempunkte so weit wie möglich optimiert wurde, wird als nächstes jedem Punkt ein Deskriptor zugewiesen, der ein späteres Vergleichen der Punkte möglich macht. Ein Deskriptor beschreibt die lokalen Eigenschaften des Bildes in einer Umgebung um einen Schlüsselpunkt und er sollte invariant bezüglich Rotation sein.

Dafür ist es zunächst notwendig, allen Schlüsselpunkten eine bestimmte Orientierung zuzuweisen. Dabei handelt es sich um die Richtung, in der die meisten Gradienten der Punkte um den Schlüsselpunkt zeigen, die sogenannte Hauptrichtung.

Der Betrag des Gradienten $m(x, y)$ und die Orientierung $\theta(x, y)$ (siehe Abb. 2.9) werden wie folgt berechnet:

$$\begin{aligned} m(x,y) &= \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \\ \theta(x,y) &= \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))). \end{aligned}$$

Dabei ist $L(x, y) := L(x, y, s)\ \forall s$.

Um die Hauptrichtung zu bestimmen, werden die Pixel um den Schlüsselpunkt einzeln durchlaufen und die Richtung des Gradienten in ein Histogramm eingetragen.

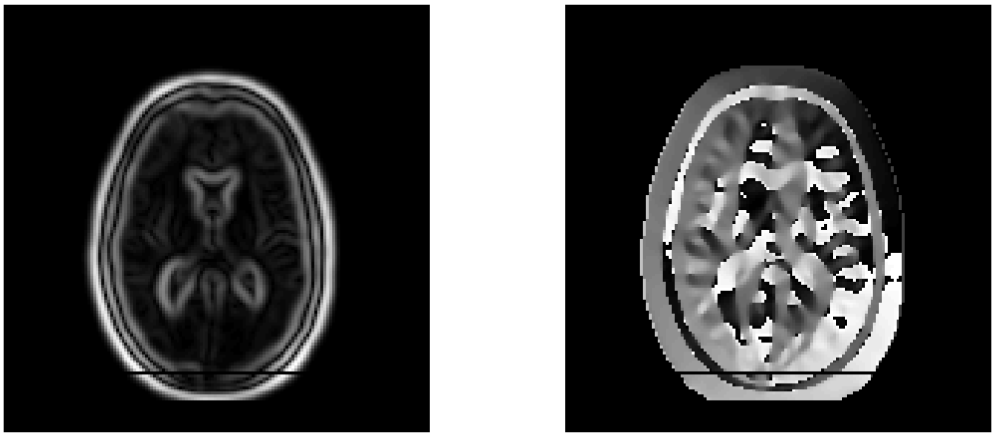


Abbildung 2.9: Im linken Bild ist der Betrag m des Gradienten an jedem Pixel aufgetragen, im rechten Bild die Richtung θ des Gradienten. Ein sehr kleiner Richtungswinkel wird dabei dunkel gefärbt. Beim Sprung von 360° zu 1° kann es dadurch zu einem abruptem Wechsel von Weiß zu Schwarz kommen.

Jede Klasse repräsentiert dabei einen Bereich von 10°. Außerdem wird der Betrag des Gradienten in der Klasse vermerkt, sodass wenige sehr große Gradienten stärker als viele kleine gewichtet werden.

Im Folgenden beziehen wir uns ohne Beschränkung der Allgemeinheit auf einen beliebigen, aber fest gewählten Schlüsselpunkt $\kappa \in \mathcal{K}$.

Sei

$$\mathcal{H}(n) := \{(x,y) \in \text{Neighborhood}(\kappa) : \\ (n \cdot 10^\circ - 5^\circ) \leq \theta(x,y) < (n \cdot 10^\circ + 5^\circ)\}$$

die Menge aller Punkte um den Schlüsselpunkt κ im gewünschten Radius, die sich in der jeweils aktuell betrachteten Histogrammklasse (also dem Winkel) befinden. Die Größe des Radius, der die Nachbarschaft des Punktes bestimmt, ist dabei frei wählbar. In unseren Experimenten wurde ein Radius von 5 Pixeln um den Schlüsselpunkt gewählt:

$$\text{Neighborhood}(\kappa) := \{(x,y) : (x_\kappa - 5, y_\kappa - 5) \leq (x,y) \leq (x_\kappa + 5, y_\kappa + 5)\}$$

Als zusätzliche Gewichtung wird eine Gauß-Verteilung kreisförmig mit dem Schlüsselpunkt als Zentrum über das Bild gelegt. So sind Punkte, die weiter vom Zentrum

entfernt sind, weniger wichtig für die Berechnung der Hauptrichtung als Pixel direkt am Mittelpunkt. Als Standardabweichung σ wurde die Hälfte der Breite des betrachteten Fensters gewählt.

Daraus ergibt sich das gesuchte Histogramm als

$$h(n) := \sum_{(x,y)\in\mathcal{H}(n)} G_\sigma(x - x_\kappa, y - y_\kappa) \cdot m(x,y), \quad n = 0, 1, \ldots, 35.$$

Es ist zu beachten, dass die Winkel von 355° bis 360° zur Histogrammklasse h_0 (also Winkel 0°) gehören.

Die Histogrammklasse, die am Ende den größten Wert hat, repräsentiert die Hauptrichtung des Schlüsselpunktes. Sollte es neben der größten Richtung noch andere Richtungen geben, die im Bereich bis zu 80 % der größten Klasse liegen, werden bei dem Schlüsselpunkt mehrere Hauptrichtungen vermerkt.

2.5 Bestimmung des Deskriptors

Der Deskriptor selbst wird, ähnlich zur Orientierung, ebenfalls über ein Histogramm der Gradientenrichtungen bestimmt. Insgesamt wird hierbei jedoch ein Feld (Patch) von 16×16 Pixeln um den Schlüsselpunkt betrachtet. Diese Patches werden in 16 kleinere Felder von jeweils 4×4 Pixeln eingeteilt (siehe Abb. 2.10). In jedem dieser Patches werden eigene Histogramme über die Richtungen der Gradienten gebildet. In diesem Fall werden diese aber lediglich in 8 Klassen unterschieden, also jeweils in Bereiche von 45° unterteilt. Weniger Richtungen wären zu ungenau für das spätere Matching, mehr Richtungen aber zu anfällig für leichte Änderungen.

Sei

$$\begin{aligned} \mathcal{H}_i(n) &:= \{(x,y) \in \mathrm{Patch}_i(\kappa) : \\ (n \cdot 45° - 22.5°) &\leq \theta(x,y) < (n \cdot 45° + 22.5°)\} \end{aligned}$$

die Menge aller Punkte des i-ten Patches zum Schlüsselpunkt κ, die sich in der jeweils aktuell betrachteten Histogrammklasse (also dem Winkel) befinden. Auch hier werden die Histogrammeinträge mit dem Wert des Gradienten sowie der Gauß-Verteilung gewichtet, mit dem Schlüsselpunkt als Zentrum und σ gleich halbe Fensterbreite. Außerdem wird das Bild vor der Auswertung entsprechend der vorher bestimmten Hauptrichtung gedreht.

Dann ergibt sich das gesuchte Histogramm als

$$h_i(n) := \sum_{(x,y)\in\mathcal{H}_i(n)} G_\sigma(x - x_\kappa, y - y_\kappa) \cdot m(x,y), \quad n = 1, 2, \ldots, 8.$$

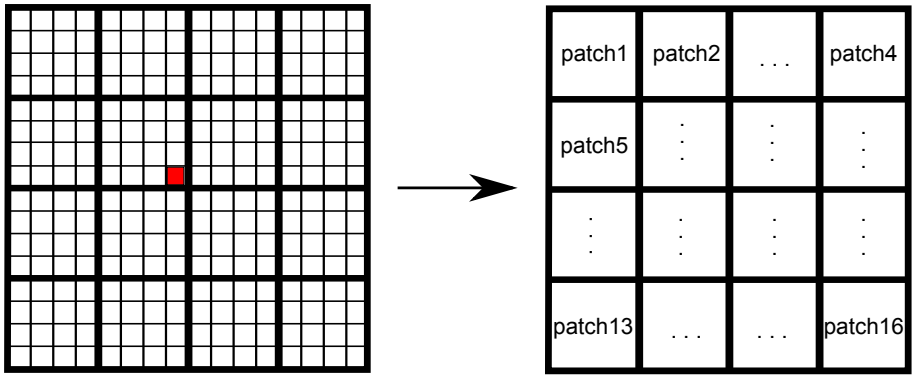


Abbildung 2.10: Aus den 16×16 Pixeln um den Schlüsselpunkt des Bildes werden 4×4 Patches. Der Schlüsselpunkt ist rot markiert.

Dabei hat jeder Patch i ein eigenes Histogramm h_i.

In jedem dieser 16 Felder ergibt sich also ein Histogramm über die 8 Richtungen, mit jeweils einer Gewichtung zu jeder Richtung, die zusammen die Gradientenrichtungen dieses Feldes beschreiben. Insgesamt kommt man so auf $16 \cdot 8 = 128$ Werte, die in einem Vektor, dem Deskriptor, gespeichert werden und damit ein exaktes Bild des Bereiches um den Schlüsselpunkt beschreiben.

Diese Werte werden anschließend zur besseren Vergleichbarkeit auf den Bereich [0,1] normalisiert. Um außerdem zu verhindern, dass einzelne Werte den Deskriptor zu sehr dominieren (diese könnten im Vergleichsbild zum Beispiel verdeckt sein und es könnte somit ein ganz anderer Deskriptor entstehen), werden alle Werte über 0.2 abgeschnitten und danach wieder zurück auf [0,1] normalisiert. Der Deskriptor $\text{desk} \in \mathbb{R}^{128}$ ist definiert als

$$\begin{aligned} \text{desk}(8 \cdot (i-1) + n) &:= 5 \cdot \max\left(\frac{h_i(n)}{h_{max}}, \frac{1}{5}\right) \text{ für } i = 1, \ldots, 16,\ n = 1, \ldots, 8 \\ h_{max} &:= \max\{h_i(n) : i = 1, \ldots, 16, n = 1, \ldots, 8\}. \end{aligned}$$

Nun wurde also für jeden Schlüsselpunkt des Bildes einen 128-dimensionalen Deskriptor erstellt, der diesen beschreibt, und so mit anderen Punkten vergleichbar macht (siehe Abb. 2.11).

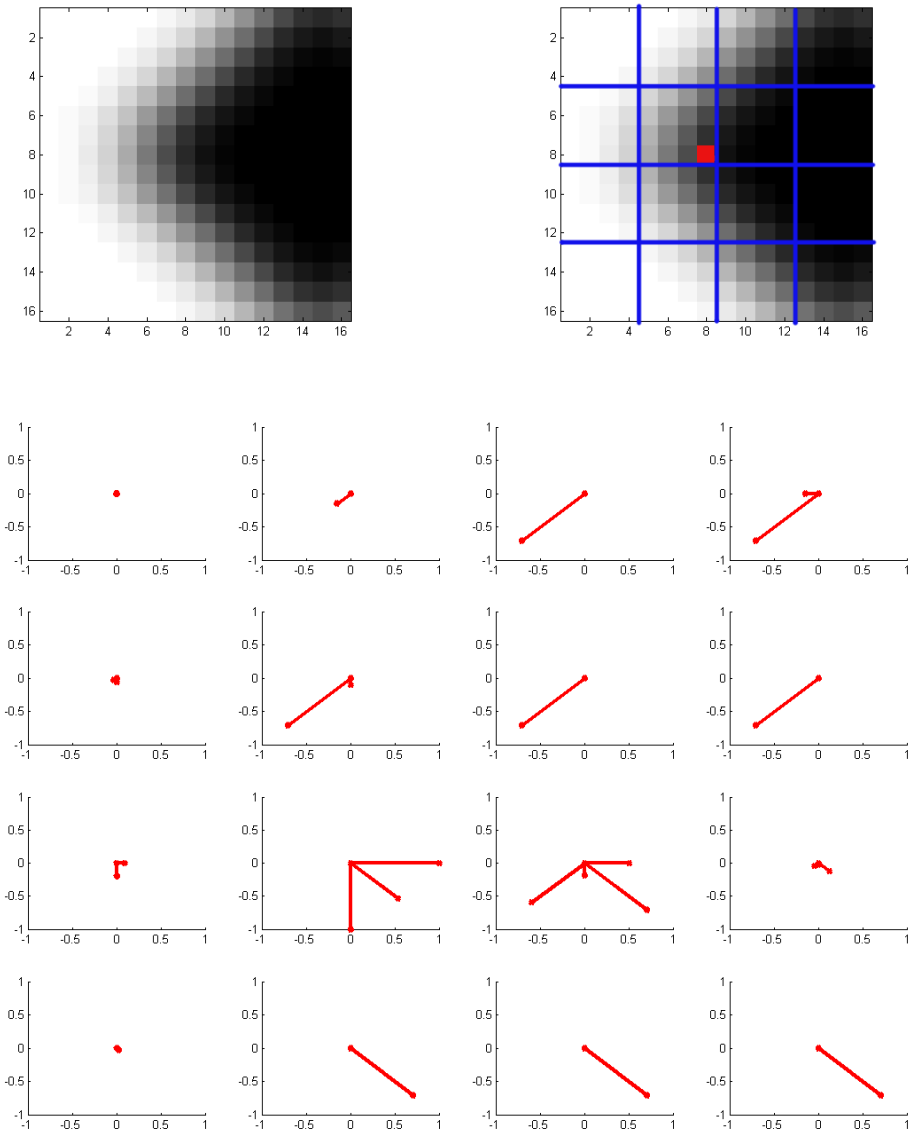


Abbildung 2.11: Die Ecke auf dem ersten Bild wird für den Deskriptor auf die Patches im zweiten Bild eingeteilt, der Schlüsselpunkt ist rot dargestellt. Unten: Der daraus entstandene Deskriptor. Die Ecke ist auch hier klar erkennbar durch die nach unten links gehenden Pfeile in der oberen Hälfte, sowie die nach unten rechts gehenden im unteren Bereich.

3 Matching von Schlüsselpunkten

Das Ziel dieser Arbeit ist die Registrierung auf Basis von korrespondierenden Schlüsselpunkten. Dafür müssen die Schlüsselpunkte von zwei verschiedenen Bildern miteinander „gematched“ werden, das heißt, Korrespondenzen berechnet werden. Dabei fungiert ein Bild als bekanntes Referenzobjekt R und das zweite, neu hinzukommende Bild als Template T, das auf Ähnlichkeit getestet werden soll. Zwei Schlüsselpunkte r und t aus R und T korrespondieren, falls die dazu gehörigen Deskriptoren ähnlich sind.

Seien

$$\mathcal{R} = \{r_1, \ldots, r_m\} \text{ und } \mathcal{T} = \{t_1, \ldots, t_n\}$$

die Deskriptoren der Schlüsselpunkte von Referenzbild (mit m Schlüsselpunkten) und Template (mit n Schlüsselpunkten).

Es wird nun jeder Schlüsselpunkt des Testbildes mit den Schlüsselpunkten des Referenzbildes über ein k-nearest-neighbor-Verfahren verglichen [7]. Dieses Verfahren sucht aus der Menge aller Punkte die k nächsten Punkte (nearest neighbor) aus. Der nearest neighbor ist dabei der Schlüsselpunkt mit dem geringsten Abstand zum aktuellen Schlüsselpunkt des Testbildes. Der Abstand wird über die Deskriptoren mithilfe des euklidischen Abstandes d bestimmt:

$$d(r,t) := \sqrt{\sum_{i=1}^{n} (r_i - t_i)^2}$$

Daraus ergibt sich die Bestimmung des besten Matchings von zwei Schlüsselpunkten. Für die Menge aller Matches $\mathcal{M}_0$ gilt

$$\mathcal{M}_0 \subset \mathcal{R} \times \mathcal{T}.$$

Dann besteht ein Match m aus einem Paar (s_i, t_i), für das gilt

$$d(r_i, t_i) \leq d(r_j, t_i) \text{ für alle } r_j \in \mathcal{R}$$

Weil trotzdem nicht immer der niedrigste Abstand auch wirklich zu einem passenden Punkt im Referenzbild gehört (weil zum Beispiel der gesuchte Punkt gar nicht vorhanden ist), nutzt man diese Eigenschaft auch als Auswahlkriterium anstatt eines

gewöhnlichen Thresholds (Grenzwert). Der Schlüsselpunkt wird nur dann als gültiger Match angenommen, wenn er wenigstens einen 80 % kleineren Abstand hat, als der zweitbeste Match:

$$d(r_i, t_i) \leq 0.8 \cdot d(r_j, t_i) \text{ für alle } r_j \in \mathcal{R} \text{ , } j \neq i$$

Sollten die Abstände näher zusammen liegen, kann man davon ausgehen, dass es eher zwei sehr schlechte Matches zu falschen Punkten gibt, als zwei sehr ähnliche passende Matches. Aus diesem Grund wird auch ein k-nearest-neighbor-Verfahren genutzt, statt eines einfachen nearest-neighbor-Ansatzes.

Was trotz dieser Maßnahme allerdings trotzdem passieren kann, ist, dass zwei Punkte aus dem Testbild mit dem selben Punkt aus dem Referenzbild gematcht werden. Gerade bei sich wiederholenden Mustern im Bild tritt dies häufig auf. Da es keine perfekte Lösung gibt, um dies zu verhindern, diese doppelten Matches aber später hinderlich sind, wurden die Euklidischen Abstände aller doppelt vergebenen Schlüsselpunkte verglichen und nur der geringste unter diesen als passender Match behalten. Die anderen wurden wieder entfernt (siehe Abb 3.1):

$$\mathcal{M} = \{(r,t) \in \mathcal{M}_0 : d(r,t) \leq d(r,\hat{t}) \text{ , } (r,\hat{t}) \in \mathcal{M}_0\} \subseteq \mathcal{M}_0.$$

Das Ziel dieser Arbeit war die Registrierung medizinischer Bilder, deshalb wurden bei der Durchführung nur Bilder der ersten Oktave der DoG-Pyramide genutzt. Es ist der Normalfall, dass zwei Bilder vorlagen, die in der gleichen Größe aufgenommen wurden, aber unterschiedliche Kontraste haben, bei denen es nur um das Finden der exakten Lage des untersuchten Objekte geht. Da es bei Lowe um die Objekterkennung selbst geht, wurden dort alle Oktaven und in der Regel auch mehrere Trainingsdaten des selben Objektes genutzt. Die einzige Änderung im Matching ist dabei, dass der zweitkleinsten Abstand immer auf einem anderen Bild gesucht wird als auf dem mit dem kleinsten Abstand:

> If there are multiple training images of the same object, then we define the second-closest neighbor as being the closest neighbor that is known to come from a different object then the first (...) - [10], S.104

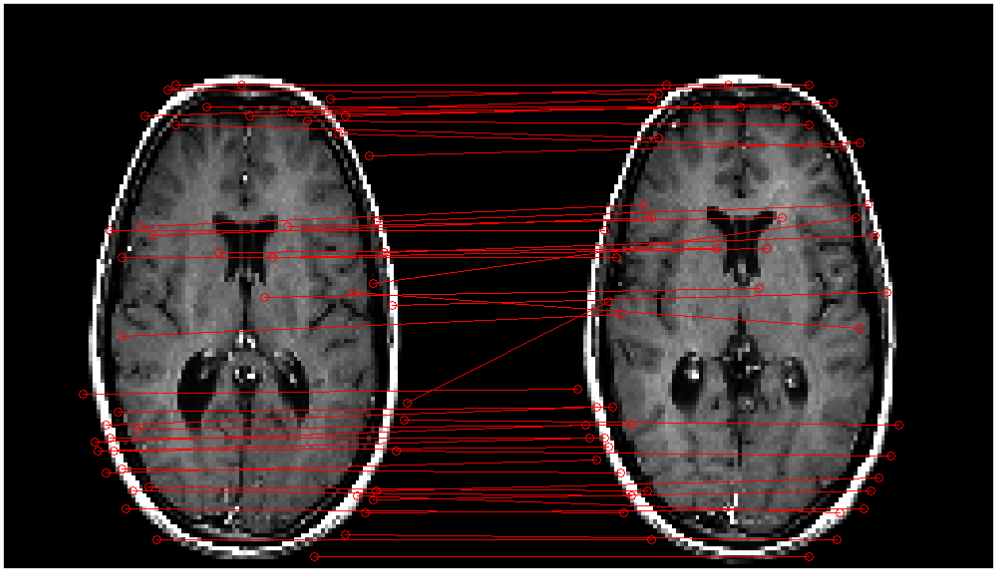


Abbildung 3.1: Ein beispielhaftes Matching von zwei verschiedenen MRI-Schichten. Die große Anzahl paralleler Linien weist auf viele korrekte Matches hin. Im Bereich der Hauptunterschiede zwischen den Bildern kommt es wie erwünscht auch zu keinem Match.

4 Registrierung mit SIFT

Nachdem der komplette SIFT-Algorithmus für Referenzbild und Templatebild durchgeführt wurde und mithilfe des Matchings zusammengehörige Schlüsselpunkte in beiden Bildern bestimmt wurden, muss nun eine Transformation bestimmt werden, damit der Abstand zwischen korrespondierenden Schlüsselpunkten minimal wird. Das Vorgehen ist ähnlich zu Landmarken-basierter Registrierung.

Im Folgenden werden dabei die Koordinaten der Schlüsselpunkte des Referenzbildes R mit $(x_r\ y_r)$ bezeichnet, analog die des Templates T mit $(x_t\ y_t)$. Weiterhin bezeichnen die Koordinaten (x_r^i, y_r^i) und (x_t^i, y_t^i) die Koordinatenpaare für die i-ten Schlüsselpunkte des Matchings.

Nun werden die Koordinaten der Schlüsselpunkte aus R und T, die in $\mathcal{M}$ als Paare vorliegen, miteinander verglichen. Man kann so erkennen, wie die Punkte gegeneinander verschoben sind und so die idealste Möglichkeit berechnen, um diese Verschiebung wieder rückgängig zu machen. In der Regel können dabei nicht alle Punkte exakt aufeinander abgebildet werden, sodass nur eine Lösung mit kleinstmöglichem Fehler gesucht wird.

Lowe benutzte für die Objekterkennung dabei eine affine Transformation [10]. Zum Vergleich wurde hier auch eine rigide Transformation getestet.

4.1 Affine Registrierung

Die affine Transformation eines Punktes aus T in den dazugehörigen Punkt aus R kann beschrieben werden durch die Gleichung

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix} + \begin{bmatrix} m_x \\ m_y \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} x_t & y_t & 0 & 0 & 1 & 0 \\ 0 & 0 & x_t & y_t & 0 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_x \\ m_y \end{bmatrix}$$

Hierbei beschreiben die beiden Parameter m_x und m_y die Translation (Verschiebung) in horizontaler und vertikaler Richtung und die Matrix mit den Parametern m_i die restlichen Transformationen Rotation (Drehung), Streckung und Scherung.

Zur Bestimmung der Parameter ist es nötig, ein lineares Gleichungssystem der Form $Ax = b$ zu lösen:

$$\underbrace{\begin{bmatrix} x_t^1 & y_t^1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_t^1 & y_t^1 & 0 & 1 \\ x_t^2 & y_t^2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_t^2 & y_t^2 & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_x \\ m_y \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} x_r^1 \\ y_r^1 \\ x_r^2 \\ y_r^2 \\ \vdots \end{bmatrix}}_{b}$$

In A sind dabei die Koordinaten des Schlüsselpunktes des Templatebildes gespeichert, in b die Koordinaten des Referenzbildes.

Da die Matrix A im Allgemeinen überbestimmt ist, gibt es keine eindeutige Lösung des Problems. Stattdessen werden die Werte gesucht, für die die Summe der Fehlerquadrate ihr Minimum annimmt:

$$x^* = \arg\min_x \|Ax - b\|^2 \text{ für } x = [m_1, m_2, m_3, m_4, m_x, m_y]^T$$

Dieses Optimierungsproblem wurde in MATLAB mittels QR-Zerlegung mit Householdertransformation gelöst. Hierfür Dwurde der Backslash-Operators ($A\backslash b$) genutzt.

Als weiteres Korrekturverfahren des vorhergehenden Matchings wird nach der Lösung des Gleichungssystems die Probe der ermittelten Parameter für jedes Koordinatenpaar durgeführt. Für eine exakte Lösung würde $Ax - b = 0$ ergeben, daher können alle Punkte mit einem Wert größer als ein bestimmter Schwellwert, in unserem Fall 10, aussortiert werden:

$$\hat{\mathcal{M}} = \left\{ (r,t) \in \mathcal{M} : \left\| \begin{bmatrix} x_t & y_t & 0 & 0 & 1 & 0 \\ 0 & 0 & x_t & y_t & 0 & 1 \end{bmatrix} x - \begin{bmatrix} x_r \\ y_r \end{bmatrix} \right\|_2 \leq 10 \right\} \subseteq \mathcal{M}.$$

Danach wird mit diesen aussortierten Schlüsselpunktpaaren analog zu vorher ein Gleichungssystem aufgestellt und gelöst. Die daraus bestimmten Werte werden für die Transformation genutzt.

4.2 Rigide Registrierung

Bei der rigiden Transformation handelt es sich um eine Untergruppe der affinen Transformation, die nur Änderungen zulässt, die längen-, winkel- und orientierungs-

erhaltend sind, also ausschließlich Rotation, Verschiebung und Spiegelung. Skalierungen und Scherung sind nicht möglich.

Die mathematische Beschreibung kann analog zum affinen Fall Art dargestellt werden. Einziger Unterschied ist, dass die Werte m_1, m_2, m_3 und m_4 alle abhängig vom Drehwinkel α sind:

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix} + \begin{bmatrix} m_x \\ m_y \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} x_t & y_t & 0 & 0 & 1 & 0 \\ 0 & 0 & x_t & y_t & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha \\ -\sin\alpha \\ \sin\alpha \\ \cos\alpha \\ m_x \\ m_y \end{bmatrix}$$

Um dies zu gewährleisten, muss bei der Lösung des Gleichungssystems eine Zusatzbedingung eingeführt werden, durch die alle Werte m_i von α abhängig sind. Gesucht ist

$$x = x(\alpha, m_x, m_y) = \begin{bmatrix} \cos\alpha \\ -\sin\alpha \\ \sin\alpha \\ \cos\alpha \\ m_x \\ m_y \end{bmatrix}, \text{ sodass } \|Ax - b\|^2 \stackrel{!}{=} \min.$$

Da das Gleichungssystem nicht mehr linear ist, kann der Backslash-Operator in diesem Fall nicht genutzt werden. Stattdessen muss eine Methode zur Optimierung von nicht-linearen Gleichungssystemen mit mehreren Parametern genutzt werden, zum Beispiel das Newton-Verfahren. In dieser Arbeit wurde jedoch eine direkte Suche mit *fminsearch* genutzt. Dabei handelt es sich um eine Implementation des Simplex-Algorithmus nach John Nelder und Roger Mead (Downhill-Simplex). Auch hier wurde im Anschluss, wie bei der affinen Registrierung, eine Probe mit den errechneten Werten gemacht, um Schlüsselpunktpaare mit schlechten Werten auszusortieren und das Ergebnis bei einem zweiten Durchlauf exakter zu bestimmen.

5 Experimente

Wie in Kapitel 1 erwähnt, wurde der SIFT-Algorithmus zur Objekterkennung entwickelt. Es gilt nun also zu testen, ob er sich auch für die Bildregistrierung eignet. Es ist zu erwarten, dass die Erkennung gleicher Punkte in beiden Bildern genauso gut funktioniert wie bei der Objekterkennung. Im Folgenden wird zunächst beschrieben, welche Daten zum testen genutzt wurden und im Anschluss die Ergebnisse beschreiben und evaluiert.

5.1 Durchführung

Für diese Experimente wurden Paare von Bildern aus unterschiedlichen Quellen und mit verschiedenen Störungen miteinander verglichen. Dabei wurden einerseits die in Kapitel 2 schon erwähnten medizinischen MRI-Bilder von Gehirnschichten genutzt. Außerdem wurde ein akademisches Bild, das nur aus Quadraten und Dreiecken besteht (siehe Abb. 5.1), sowie zwei detailreiche Fotos eines Objektes, die aus verschiedenen Perspektiven aufgenommen wurden, für die Tests verwendet.

Im akademischen Beispiel wurde als Template T ein Bild genutzt, das durch Transformation von R in horizontaler und vertikaler Richtung verschoben wurde. Außerdem wurde es um seinen Mittelpunkt gedreht. Bei den MRI-Bildern wurde versucht, zwei verschiedene Gehirnschichten aus dem MRI miteinander zu matchen. Das Template wurde jeweils vorher transliert und rotiert (siehe Abb. 5.2).

Die Fotos wurden nicht weiter für die Tests verändert, da es sich bereits um ein Bildpaar eines Objektes aus zwei verschiedenen Blickwinkeln handelt. Außerdem wurden bei den Fotos unterschiedliche Bildgrößen und Verhältnisse gewählt, um dessen Unabhängigkeit für den Algorithmus zu testen (siehe Abb. 5.3). Bei allen Varianten wurde zudem ein Vergleich mit dem Bild selbst als Template vorgenommen, um zu überprüfen, ob die Identität vom Algorithmus korrekt erkannt wird.

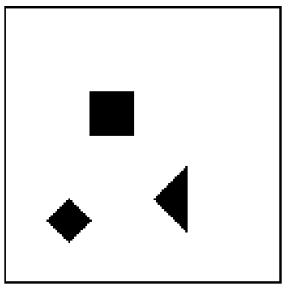


Abbildung 5.1: Ein simples akademisches Beispiel als Testobjekt für SIFT.

Abbildung 5.2: Zwei unterschiedliche Gehirnschnitte aus dem MRI.

Abbildung 5.3: Zwei Bilder des gleichen Objektes aus unterschiedlichen Winkeln.

5.2 Ergebnisse

Um die Wirksamkeit der Auswahlkriterien zu vergleichen, befindet sich in Tabelle 5.1 eine Auflistung der Punkteanzahlen in den verschiedenen Beispielbildern. Es werden die Anzahl der gefundenen Extrema in beiden Bildern, die Anzahl der daraus resultierenden Schlüsselpunkte und die Anzahl der tatsächlichen Matches aufgeführt.

Zunächst wurde getestet, wie gut der SIFT-Algorithmus bei Eingabe zweier identischer Bilder funktioniert. Dabei zeigte sich, dass alle Versuche zum erwarteten Ergebnis führen: Es werden viele Matches gleicher Punkte gefunden und keine Matches von nicht zusammengehörenden Punkten und dementsprechend werden die Bilder als identisch erkannt.

Wenn zwei unterschiedliche MRI-Schichten verglichen werden (Abb. 5.6), resultiert dies dagegen in mehreren falschen Matches, was gut an den nicht parallelen Linien zu erkennen ist. Wenn man die dazu gehörigen Werte in Tabelle 5.1 betrachtet, kann man erkennen, dass aus einer Anzahl von fast 200 Schlüsselpunkten pro Bild nur 52 Matches gebildet werden. Die Auswahlkriterien haben also eine Vielzahl der möglichen Punkte und Matches, die sich daraus bildeten, ignoriert. Wie man an den Transformationen erkennt, fallen die einzelnen falschen Matches auch nicht weiter ins Gewicht. Beide Bilder werden exakt auf einander abgebildet.

Um die Wirkung auf Translation zu beobachten, wurde eine der MRI-Schichten als Template vertikal nach unten verschoben (siehe Abb. 5.7). Es treten keine Fehlmatches auf, alle Punkte werden exakt auf ihre korrespondierenden Punkte im anderen Bild gematcht. Wenn man die dazugehörigen Zahlen betrachtet, sieht man wie zu erwarten im Templatebild eine sehr viel geringere Anzahl an Schlüsselpunkten, aber es wird fast die gesamte Menge dieser Punkte für die Matches genutzt. Nur Schlüsselpunkte am unteren Rand fallen weg, da sich der Deskriptor an dieser Stelle durch die abgeschnittenen Daten stark unterscheidet. Horizontale Verschiebung oder beide Richtungen gemischt führt zu analogen Ergebnissen.

Beim Test des Algorithmus auf Rotation wurde zunächst das akademische Beispiel getestet. Was dabei direkt auffällt, ist die Menge an neuen Extrema, die durch die Drehung von T entstehen (Tabelle 5.1 für 5.4 und 5.5). Durch die Drehung des Bildes für das Template entsteht um das Bild an den Stellen, wo es vorher nicht definiert war, automatisch durch MATLAB ein neuer schwarzer Rahmen, der dann beim Übergang zum weißen Bereich des Bildes zu vielen Extrema führt. Es wird aber weniger als die Hälfte überhaupt als Schlüsselpunkt genutzt und aus diesen dann nur Matches gebildet, die im ursprünglichen Bild liegen. Die Auswahlkriterien der Schlüsselpunkte sowie des Matches erfüllen also ihre Aufgabe.

Bei einer geringen Rotation des Bildes (Abb. 5.4) ist dann auch das Ergebnis entsprechend gut. Es sind nur sehr kleine Abweichungen zwischen R und T nach der

Transformation erkennbar. Bei einer größeren Rotation allerdings sieht das Ergebnis deutlich schlechter aus (Abb. 5.5). Lediglich fünf Matches wurden berechnet, zudem einer davon falsch. Entsprechend verschlechtern die Transformationen sogar T, anstatt es R anzupassen.

Daran lässt sich erkennen, dass einfache Bilder nicht automatisch gute Ergebnisse bedeuten, eher im Gegenteil: Das eigentlich simpel aussehende akademische Beispiel, das nur aus geometischen Figuren in Schwarz auf Weiß besteht, ist für die Anwendung in SIFT ungeeignet. Da die Kanten exakt von schwarz auf weiß wechseln, fehlen die beschreibenden Pixel um den Schlüsselpunkt, die wichtig für die Eindeutigkeit des Deskriptors sind. Damit ist es für den Algorithmus nicht mehr möglich, zwischen den verschiedenen Ecken der Figuren noch eindeutig zu unterscheiden. Ohne Rotation ist dies noch problemlos möglich, da der Algorithmus die einfachste Lösung sucht. Aber je größer der Winkel wird, umso ungenauer wird das Ergebnis, da zum Beispiel die Ecken des einen Quadrates bei einer Rotation von 45° genau den Ecken des anderen Quadrates entsprechen und umgekehrt.

Bei der Rotation der MRI-Bilder zeigt sich, dass auch hier eine sehr viel geringere Menge von Matches im Vergleich zum unrotierten Bild zustande kommt. Trotzdem zeigt sich in Abbildung 5.8, dass der Algorithmus das Template für eine Drehung von 10° richtig transformiert. Die rigide Transformation ist sogar nahezu exakt. Je höher der Drehwinkel wird, um so schlechter wird die Transformation, dabei bleibt die rigide Transformation auch weiterhin besser als die affine (Abb. 5.9). Da das Matching vor allem viele Punkte am Schädel findet, weil diese einen größeren Kontrast als die Punkte im Gehirn besitzen, wird es bei größerem Rotationswinkeln ungenau. Der Schädel ist annährend rund und sieht damit für SIFT in jedem Rotationswinkel gleich aus. Mit größeren Winkel ist es demnach für den Algorithmus schwierig zu unterscheiden, ob eine Stelle vom Schädel gerade „oben“ im Bild ist.

Zuletzt wurde der Algorithmus an zwei Fotos eines Schiffes getestet. Hier lässt sich sehr gut erkennen, wie das Verhältnis von lokalen Extrema zu Schlüsselpunkten und zu den Matches ist. Es werden weit über 1000 Extrema pro Bild gefunden, aber nur ungefähr ein Drittel davon bleibt als Schlüsselpunkt übrig. Die Anzahl der Matches ist wieder nur ein Viertel von dieser Menge. Wie Abbildung 5.10 zeigt, ist die Großzahl dieser Matches richtig, nur sehr wenige wurden falsch berechnet. Besonders hervorzuheben ist hier, dass auch viele der Fenster mit dem exakt korrespondierenden Fenster im anderen Bild gematcht werden, obwohl sie für das menschliche Auge alle gleich aussehen.

Die Transformation der Bilder ist trotz guter Daten nicht ganz exakt, so ist es weder rigide noch affin möglich, ein perfektes Matching zu produzieren. Man kann aber gut erkennen, dass durch die Drehung korrespondierende Kanten in beiden Bildern parallel verlaufen und sich die generelle Position der Objekte im Bild angepasst

hat. Was man außerdem beobachten kann, ist, dass die affine Transformation die exakte Rotation besser ermittelt, dafür die rigide Transformation die Translation fast exakt bestimmt. So sind im einen die Bilder paralleler, während im anderen die grundsätzliche Position besser übereinstimmt. Es war nicht zu erwarten, dass SIFT hier ein exaktes Matching liefern kann, da die Motive der Bilder durch die unterschiedliche Position der Kamera nicht exakt gleich sind. Trotzdem zeigt sich an diesem Beispiel, welche Möglichkeiten der Algorithmus hat, um die Eindeutigkeit der Deskriptoren zu gewährleisten, wenn die Bilddaten detailliert genug sind.

Bild	**Extrema R**	**Extrema T**	**Keypoints R**	**Keypoints T**	**Matches**
5.4	11	108	11	46	9
5.5	11	199	11	89	5
5.6	187	189	186	189	52
5.7	187	100	186	100	94
5.8	187	184	186	182	28
5.9	187	184	187	181	27
5.10	1698	1317	525	404	100

Tabelle 5.1: Auflistung der vom Programm gefundenen Punkte in den folgenden Bildern.

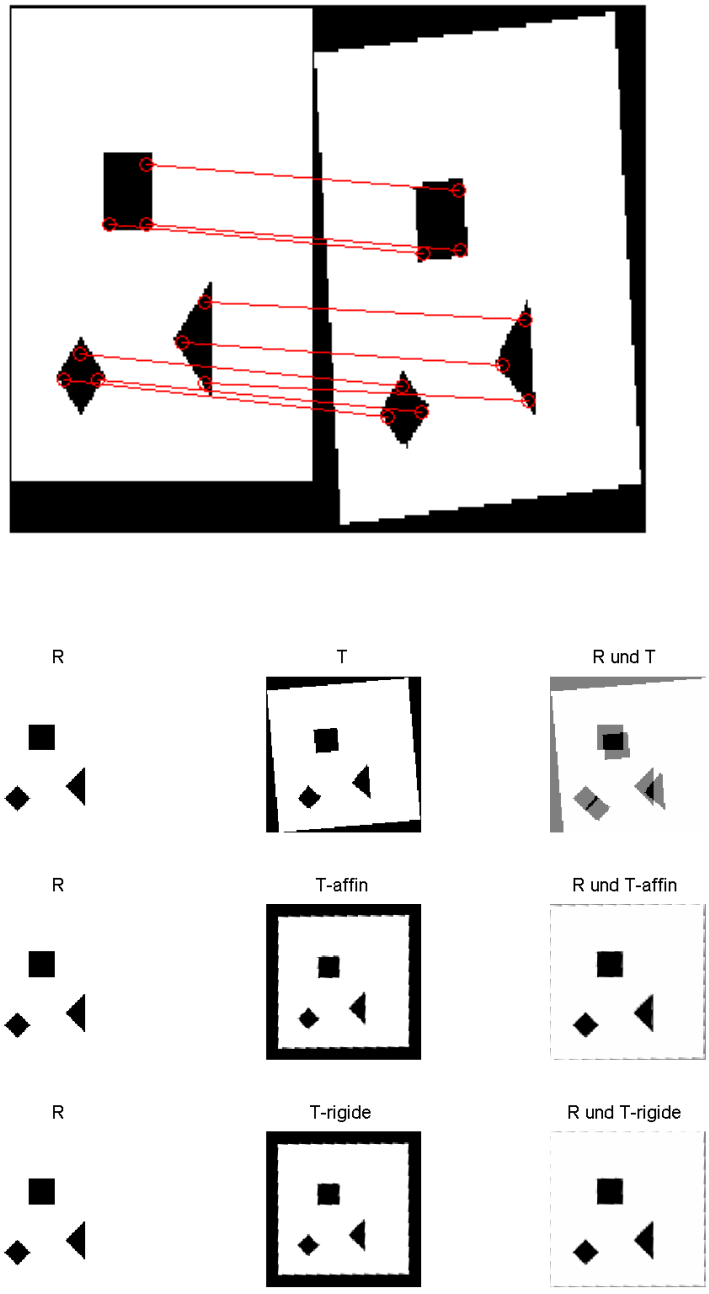


Abbildung 5.4: Vergleich von affiner und rigider Registrierung am akademischen Beispiel mit geringer Rotation. (Hier und im folgenden befinden sich im oberen Bild die Matches, verbunden durch Linien, im unteren Bild die Transformationen. In der ersten Zeile die Originalbilder, in der zweiten Zeile die affine Transformation und in der letzten die rigide. Referenzbild immer links, Template (mit Transformation) in der Mitte, rechts beide übereinander gelegt als Vergleich.)

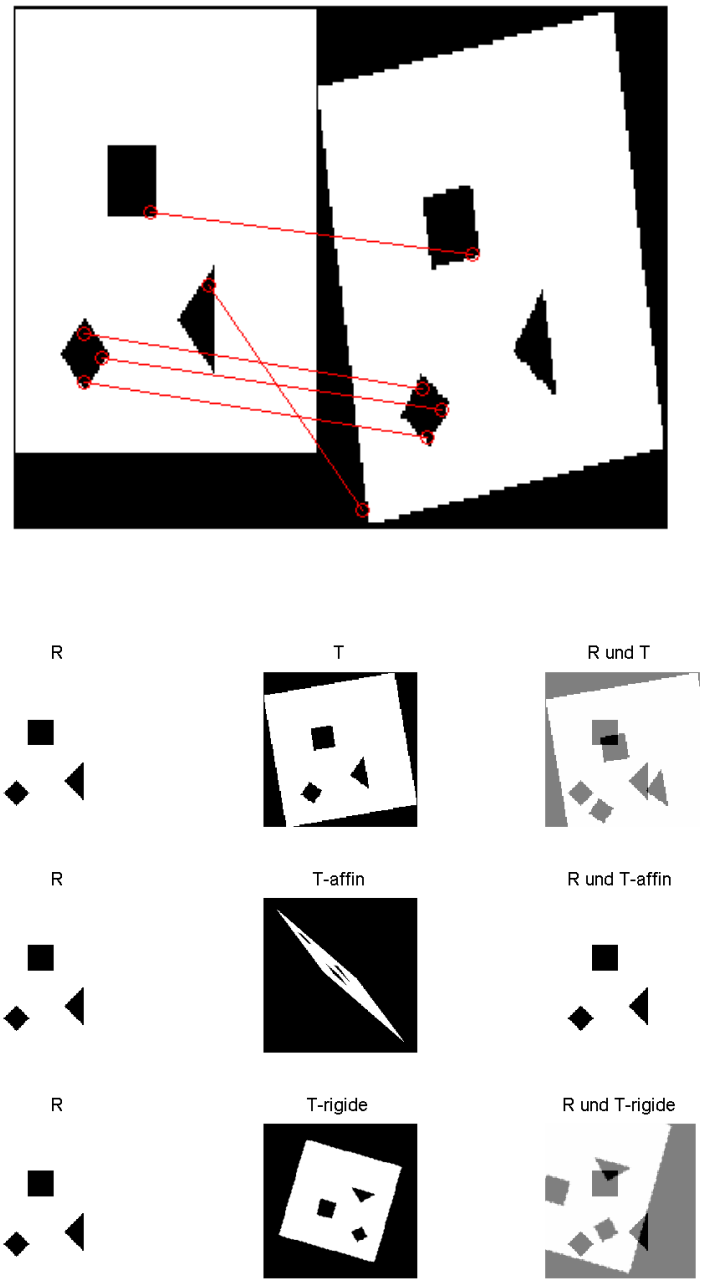
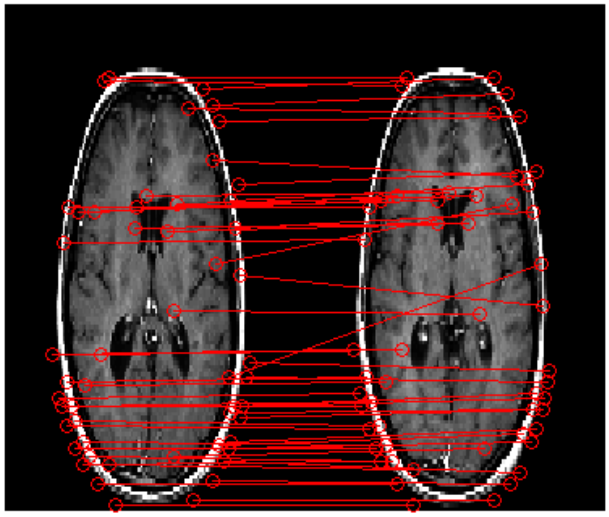


Abbildung 5.5: Akademisches Beispiel mit stärkerer Rotation. Es gibt sehr wenig Matches, darunter zudem falsche. Beide Transformationen resultieren in gleich schlecht Ergebnissen.



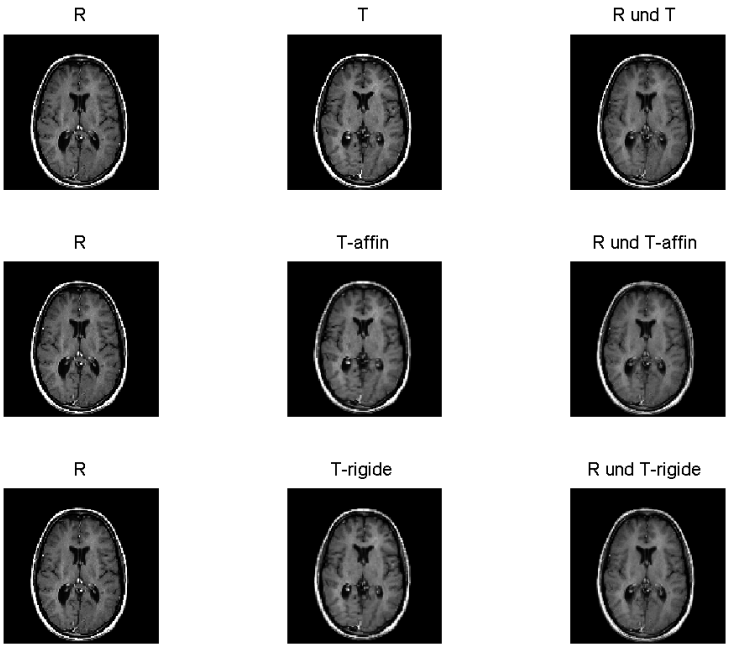
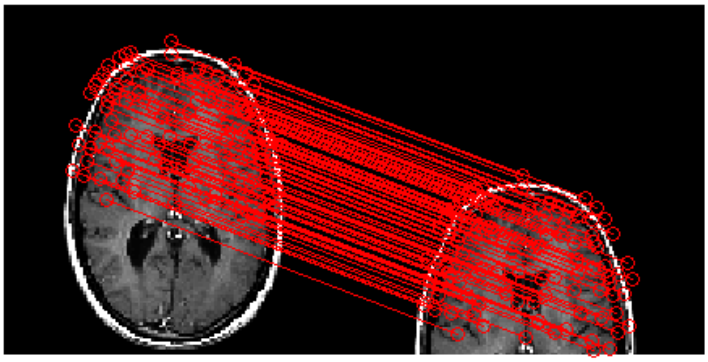


Abbildung 5.6: Registrierung mit zwei Schichten eines MRI ohne weitere Veränderung. Beide Transformationen sind ähnlich gut.



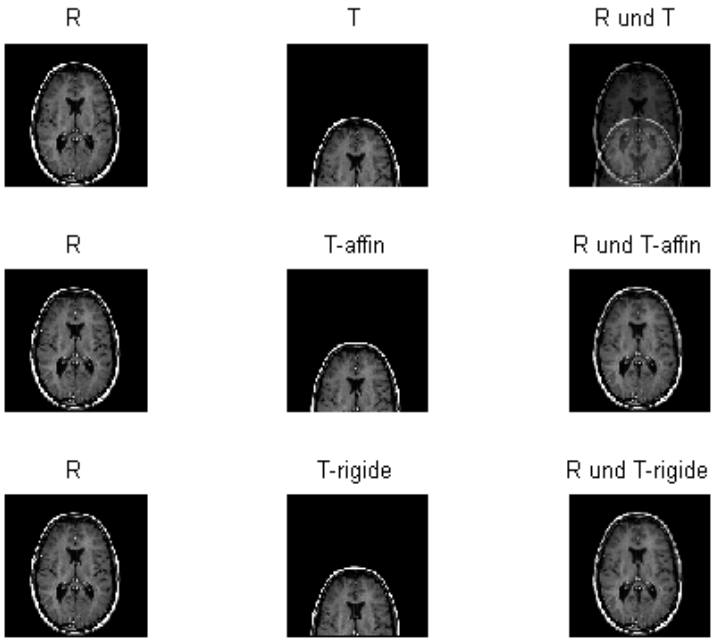


Abbildung 5.7: Registrierung einer Schicht eines MRI, das Template ist nach unten verschoben. Beide Transformationen verschieben T exakt auf das Referenzbild zurück.

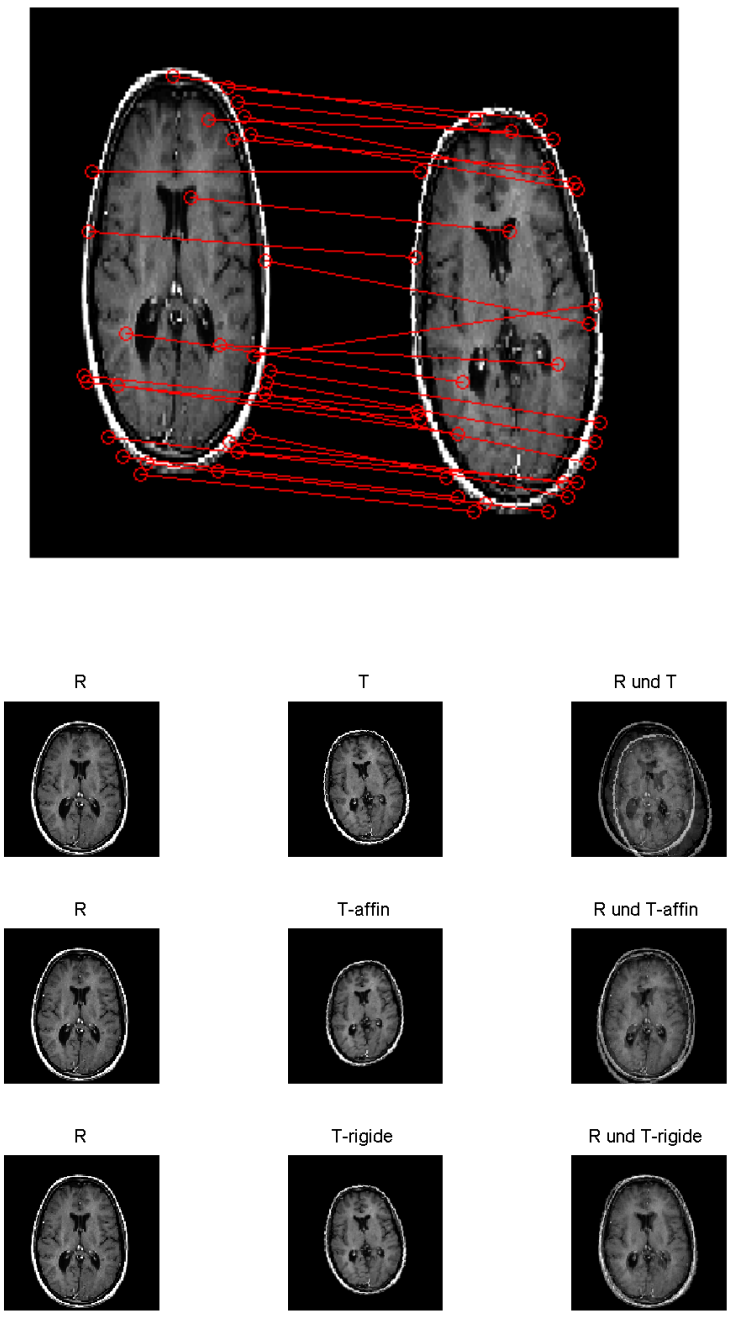


Abbildung 5.8: Registrierung von zwei MRI-Schichten, eine davon gedreht um 10°. Man kann bereits erkennen, dass die rigide Transformation ein exakteres Ergebnis liefert. Die affine Transformation wird durch die Scherung etwas unförmig.

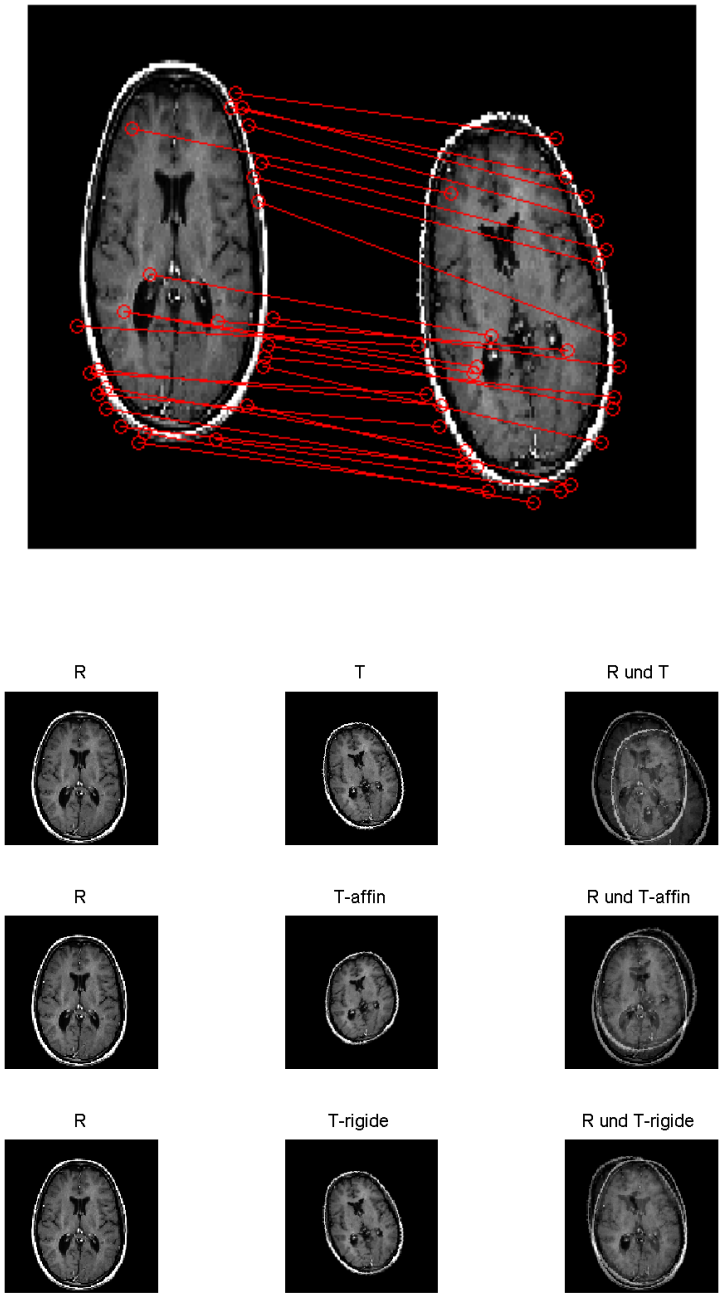


Abbildung 5.9: Registrierung von zwei MRI-Schichten, eine davon gedreht um 15°. Beide Transformationen erhalten keine exakte Lösung mehr, die rigide ist aber deutlich näher dran.

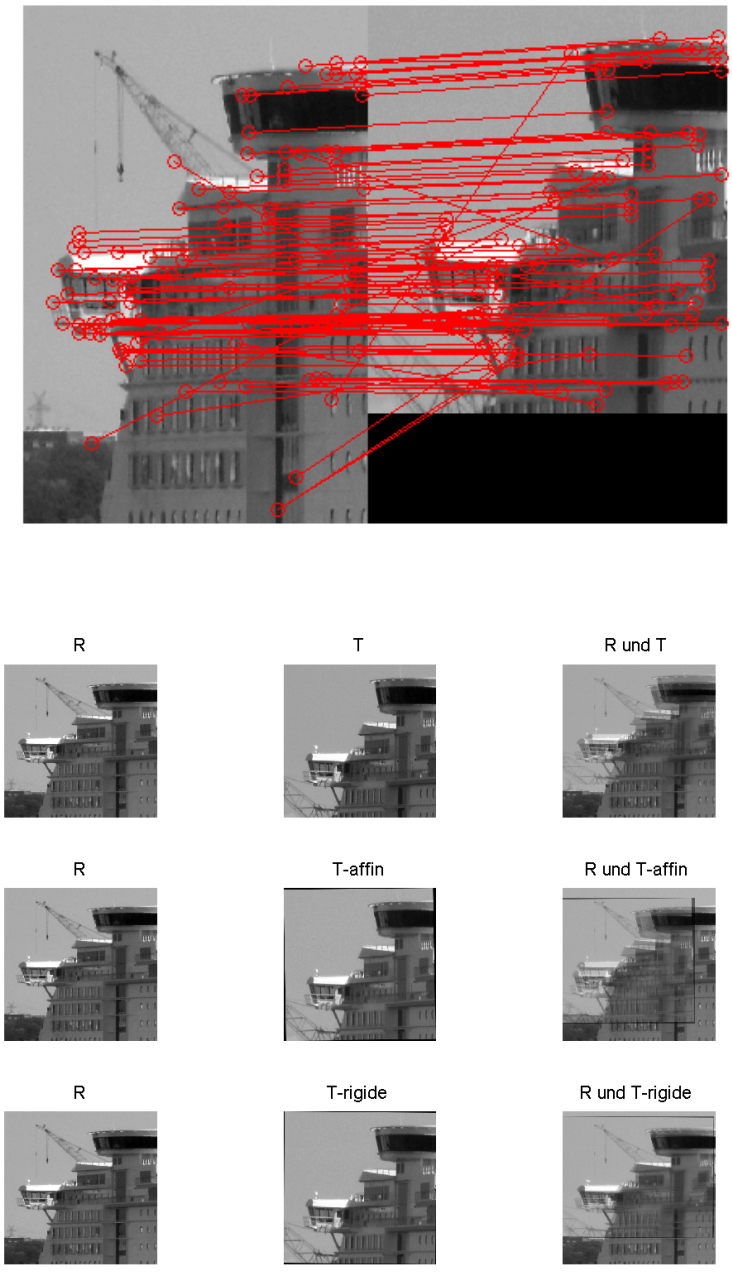


Abbildung 5.10: Registrierung der zwei Schiffbilder. Bei der affinen Transformation stimmt der Winkel besser überein, die rigide ist aber gesamt, vor allem von der Postion her, besser.

6 Fazit

Wie aus den Experimenten ersichtlich wird, ist eine korrekte Registrierung mittels SIFT prinzipiell möglich, scheitert aber teilweise an den exakten Anforderungen. Bei den getesteten MRI-Bildern waren hauptsächlich die äußeren Kanten für das Matching relevant, der Innenraum wurde kaum genutzt. Die äußeren Kanten reichten allerdings nicht aus, um eine mögliche Drehung ausreichend gut zu erkennen. Für eine gute Nutzung in diesem Bereich wäre es also notwendig, das Programm so zu variieren, dass es mehr auf die inneren Merkmale anspricht und den äußeren Bereich ignoriert.

Auf der anderen Seite ist es überraschend, wie exakt die erkannten Matches in anderen Fällen sind. Auf dem realistischen Fotos konnten die mit menschlichem Auge nur schlecht unterscheidbaren Fenster exakt zugeordnet werden. Der Algorithmus ist demnach in der Lage, sehr genau zu arbeiten und auch kleine Unterschiede zwischen Schlüsselpunkten zu erkennen, das entsprechende Ausgangsmaterial vorausgesetzt.

Ein Vorteil für eine praktische Nutzung ist außerdem, dass einzelne Fehlmatches, die sich nie ausschließen lassen, in den meisten Fällen die Berechnung der Transformation nicht beeinträchtigen. Durch die Kontrollmechanismen zu verschiedenen Zeitpunkten des Algorithmus werden sie oft korrigiert. Dies zeigte sich besonders bei der rigiden Transformation, die auch bei einer kleinen Anzahl Matches noch sehr viel exakter blieb, als die affine Transformation.

Es lässt sich also zusammenfassend sagen, dass der SIFT-Algorithmus Potential für die Registrierung hat. Die Probleme, die bei der Rotation der Bilder durch die falsche Priorität der Matches entstand, müssen für eine bessere Berechnung der Transformation beseitigt werden. Wenn dann die Ergebnisse ähnlich exakt wie bei den Fotos sind, lässt sich ein praktischer Nutzen nicht ausschließen.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU) 110*, 3 (2008), 346–359.

[2] Brown, L. G. A survey of image registration techniques. *ACM Computing Surveys (CSUR) 24*, 4 (Dec. 1992), 1–60.

[3] Brown, M., and Lowe, D. Invariant features from interes point groups. In *British Machine Vision Conference* (2002), pp. 656–665.

[4] Burt, P., and Adelson, T. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications 31*, 4 (1983), 532–540.

[5] Cheung, W., and Hamarneh, G. N-sift: N-dimensional scale invariant feature transform for matching medical images. In *Biomedical Imaging: From Nano to Macro, 2007. ISBI 2007. 4th IEEE International Symposium on* (2007), pp. 720–723.

[6] Cheung, W., and Hamarneh, G. n -sift: n -dimensional scale invariant feature transform. *Image Processing, IEEE Transactions on 18*, 9 (2009), 2012–2021.

[7] Cover, T., and Hart, P. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions 13*, 1 (1967), 21–27.

[8] Fischler, M. A., and Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24*, 6 (June 1981), 381–395.

[9] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2* (Washington, DC, USA, 1999), ICCV '99, IEEE Computer Society, pp. 1150–.

[10] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2 (Nov. 2004), 91–110.

[11] Maintz, J. B. A., and Viergever, M. A. A survey of medical image registration. *Medical Image Analysis 2*, 1 (1998), 1–36.

[12] Modersitzki, J. *Numerical Methods for Image Registration*. Numerical Mathematics and Scientific Computation. Oxford University Press, 2004.

[13] Modersitzki, J. *FAIR: Flexible Algorithms for Image Registration*. SIAM, 2009.