Medizinische Bildverarbeitung

Ausarbeitung der 2. Übung: Shape Particle Filters

Gruppenmitglieder: Felix Schuller (1025256) und Johannes Riedmann (0926649)

Aufgabe 1: Shape Modell

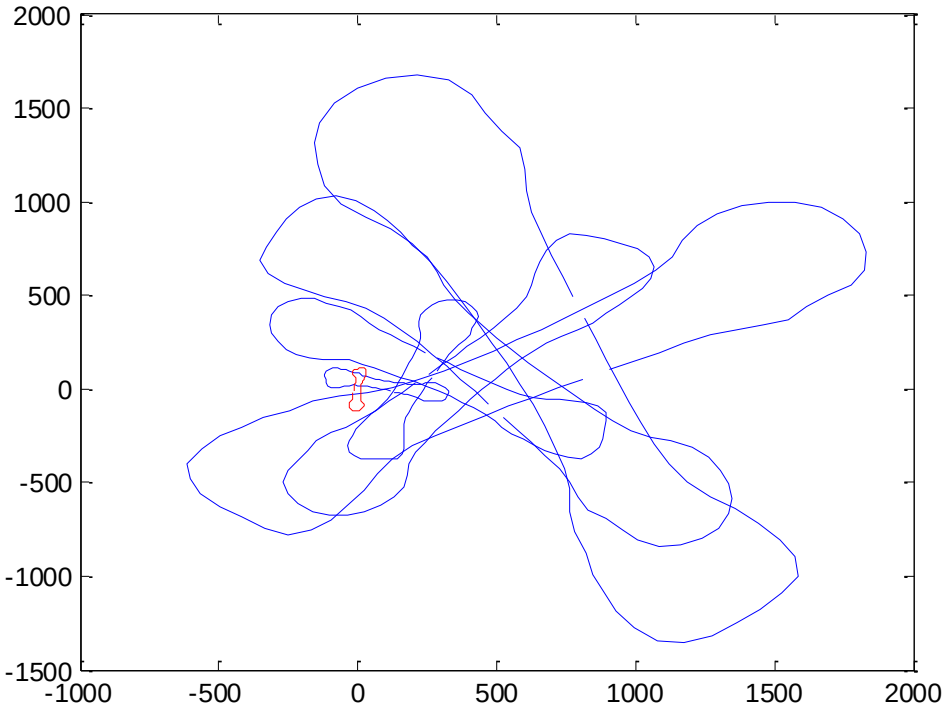


Abbildung 1: Mittels variabler Skalierung/Rotation/Translation generierte Shapes

Das Mean-Shape ist im Plot in Rot dargestellt, ohne Skalierung, Rotation oder Translation. Die blauen Shapes sind in 7 Iterationen mit unterschiedlichen Parametern erzeugt worden. Das in der jeweiligen Iteration erstellte Shape wird immer um die doppelte Größe skaliert, um PI/4 rotiert, um 100 * i (i = Anzahl der Iterationen) in x-Richtung und um 20 * i in y-Richtung verschoben.

Aufgabe 2: Featureberechnung

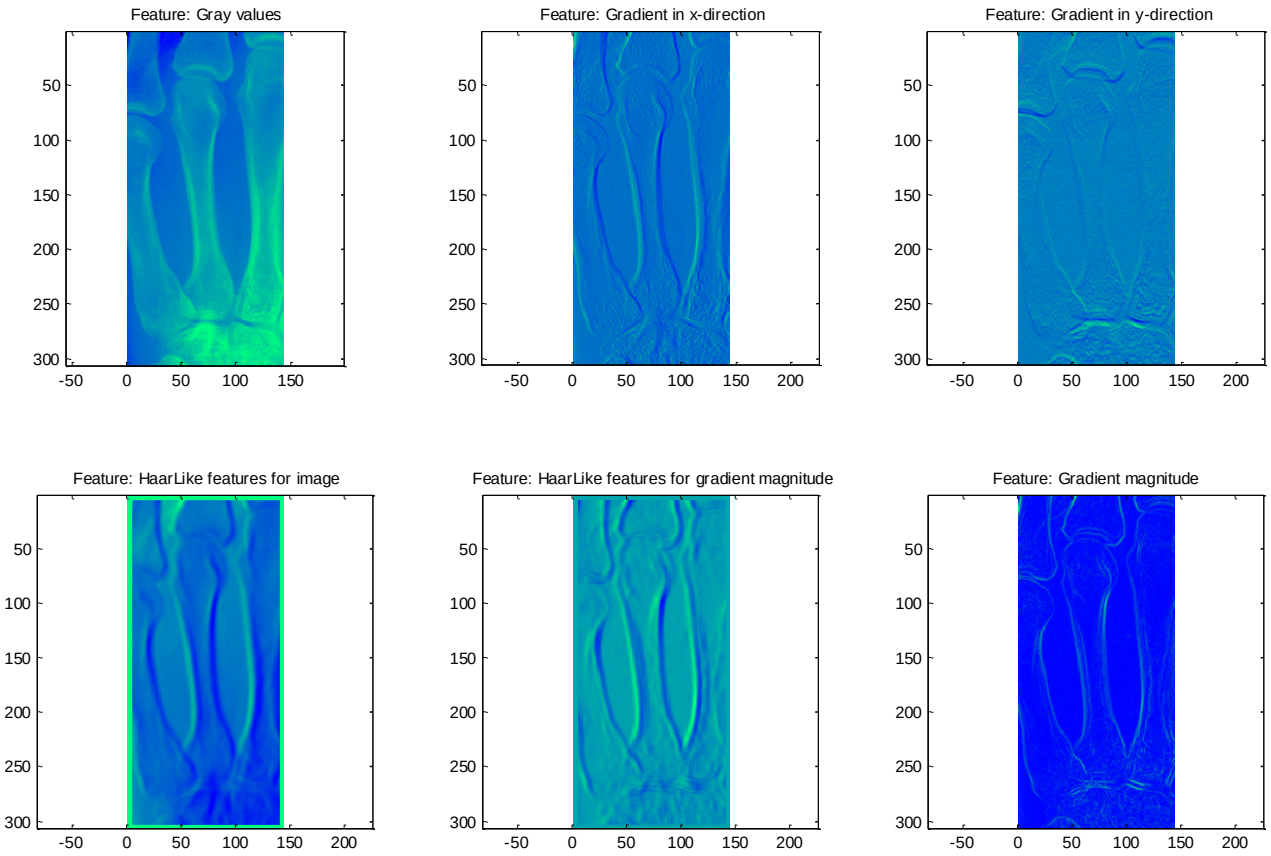


Abbildung 2: Features mit denen Random Forest trainiert wird

Das Feature-Set besteht ausschließlich aus den in der Angabe definierten Features. X- und Y-Koordinaten wurden bewusst nicht geplottet, befinden sich aber auch im Feature-Set. Wir haben zusätzlich mehrere Features getestet (Difference-of-Gaussian, diverse Kantenfilter, v.a. Filter die nicht auf Gradienten basieren da diese bereits im Feature-Set enthalten sind), haben jedoch keine Verbesserung der Prediction des Klassifikators feststellen können.

Aufgabe 3: Klassifikation & Feature Selection

Out-of-bag classification error

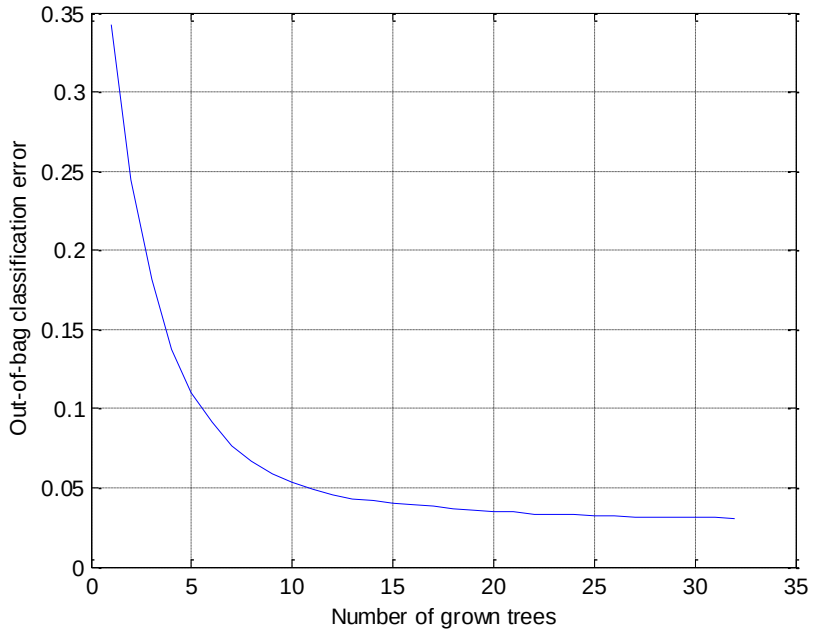


Abbildung 3: OOB Classification error

Der OOBError zeigt, dass der OOB classification error mit steigender Anzahl an Decision Trees stark sinkt. D.h. umso mehr Trees im Random Forest trainiert werden, umso geringer ist die Fehlerrate des Klassifikators. Bei ca. 11 Trees sinkt die Fehlerrate unter 5%, bei 32 Trees ist die Wahrscheinlichkeit für eine Fehlklassifikation etwa bei 3% und sinkt nicht mehr wesentlich ohne unproportional hohen Rechenaufwand für die Berechnung einer wesentlich höheren Anzahl an Trees.

Feature-Signifikanz

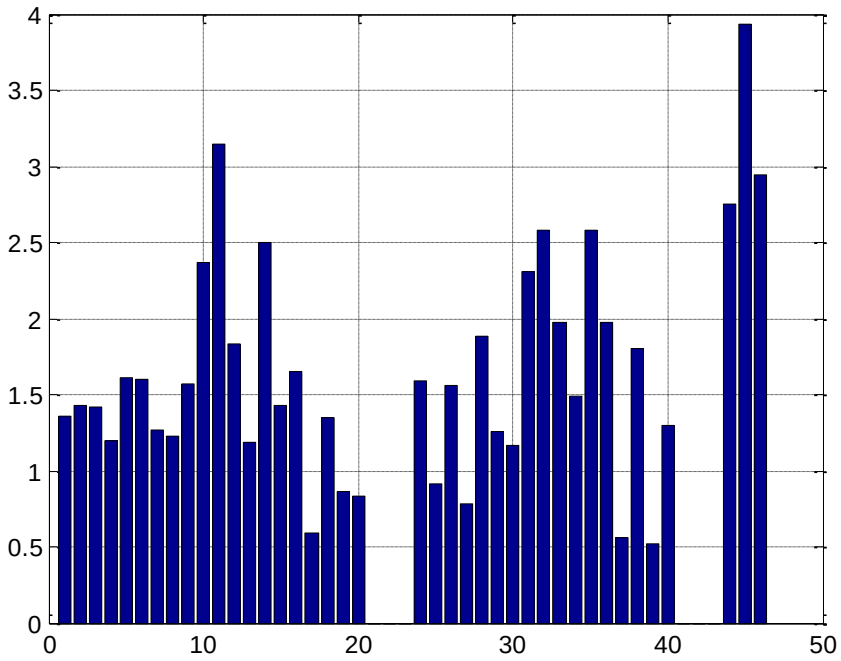


Abbildung 4: Signifikanz der einzelnen Features

Positionen der Features im Diagramm:

x-Wert (oder Wertebereich)	Feature
1	Grauwerte des Bildes
2	Gradient in x-Richtung
3	Gradient in y-Richtung
4 – 23	Haarlike-Features für Grauwert-Bild
24 – 43	Haarlike-Features für Gradientenstärke
44	Gradientenstärke
45	x-Koordinaten des Bildes
46	y-Koordinaten des Bildes

Die Signifikanz der Features für die Klassifikation wird über das OOBPermutedDelteError Property bestimmt. Die X-Koordinaten des Bildes haben mit einem Wert von ca. 3.9 die höchste Signifikanz. Dies liegt daran, dass die Bilder höher sind wie breit (dementsprechend sind weniger x-Werte vorhanden) und die Knochen dementsprechend in einem kleineren X-Bereich vorkommen (kleinere Varianz im X-Wertebereich als im Y-Wertebereich). Die Y-Koordinaten, Gradientenstärke und einige Haarlike-Features, sowohl für das Grauwert-Bild, als auch für die Gradientenstärke sind mit Werten von über 1.5 noch immer wichtig für eine korrekte Klassifikation der Bilder. Jeweils die letzten 3 Haarlike-Feature, sowohl vom Grauwert-Bild als auch von der Gradientenstärke haben absolut keine Relevanz für die Klassifizierung. Wenn diese Features aus dem Featureset entfernt werden, lassen sich keine Unterschiede in der Klassifikationsrate des Random Forests erkennen.

Aufgabe 4: Shape Particle Filters

Untersuchte Varianten für Kostenfunktion

- **Hit-or-miss**
  Dies war die erste, performanteste, simpelste aber auch ungenauste Kostenfunktion die wir getestet haben um die Shape-Parameter zu optimieren. Dabei wird einfach gezählt wie viele Datenpunkte des generierten Shapes auf der Prediction des Klassifikators liegen. Diese Anzahl wird dann invertiert und als Kosten retourniert, die es zu minimieren gilt. Die Ergebnisse mit dieser Variante waren nicht zufriedenstellend.
- **Minimierung der euklidischen Distanz**
  Die Idee dahinter ist, dass man von jedem Datenpunkt im generierten Shape die minimale euklidische Distanz zu jedem Datenpunkt der Prediction berechnet und diese dann aufsummiert. Dies wäre eine sehr gute Variante der Kostenberechnung, jedoch stellte sich heraus, dass die Berechnung dieser Methode aufgrund der hohen Anzahl an Datenpunkten in der Prediction zu langsam und rechenintensiv ist (Berechnung der Distanzen für Anzahl Datenpunkte in Prediction * Anzahl Datenpunkte in generiertem Shape). Die Konvergenz der Methode nimmt so viel Zeit in Anspruch, dass wir nicht einmal Ergebnisse für ein einziges Bild erhalten haben.
- **9er-Nachbarschaft & 25er Nachbarschaft**
  Die Methode für die wir uns schlussendlich entschieden haben, die sowohl performant ist als auch akzeptable Ergebnisse liefert, verwendet die 9er-Nachbarschaft jedes Datenpunkts der im Shape definiert ist. Diese Datenpunkte werden in der Prediction lokalisiert, eine 9er-Nachbarschaft aufgespannt und die Summe innerhalb der Nachbarschaft berechnet. Diese Summe wird dann einer Gesamtsumme hinzuaddiert. Nachdem alle Datenpunkte durchlaufen sind, wird diese Gesamtsumme invertiert, um aus der Anzahl der Hits in der 9er-Nachbarschaft die Kosten zu berechnen. Dieses Verfahren konvergiert nach 100-12000 Aufrufen der Kostenfunktion, abhängig von den initial vom Optimierungsalgorithmus gewählten Parametern.
  Die Variante mit der 25er-Nachbarschaft funktioniert exakt gleich, nur dass sie eben eine größere Nachbarschaft miteinbezieht. Die Ergebnisse waren jedoch kaum unterschiedlich (siehe Boxplots auf der nächsten Seite), deshalb haben wir uns für die 9er-Nachbarschaft entschieden.

Verwendete Prediction für Kostenfunktion

Abbildung 5: Probability prediction des Klassifikators

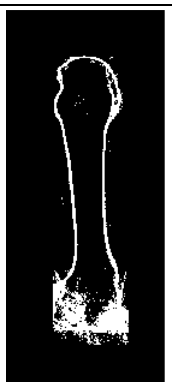


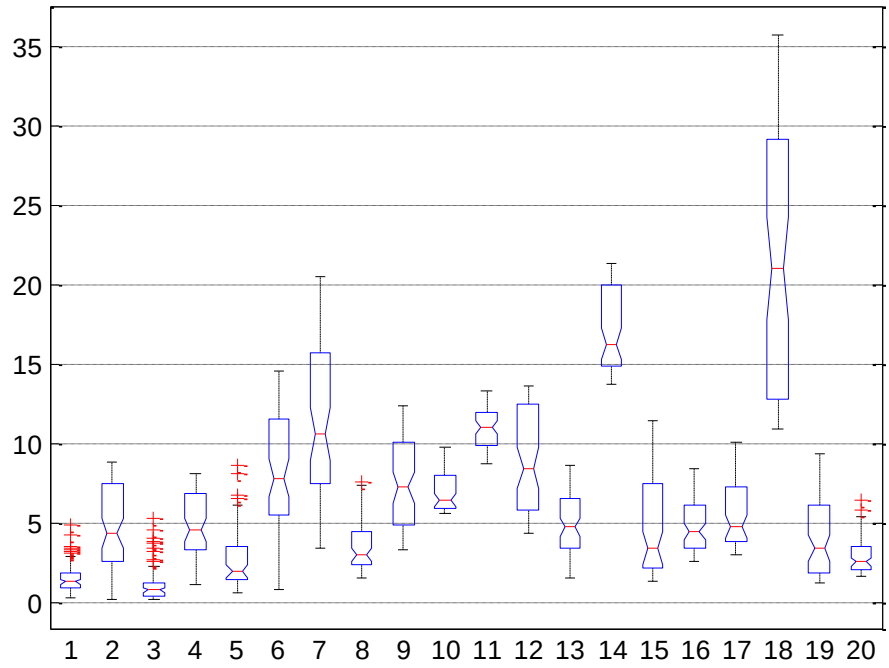
Abbildung 6: Class predictions des Klassifikators

Abbildung 7: Verwendete Prediction (Probability prediction x Class prediction)

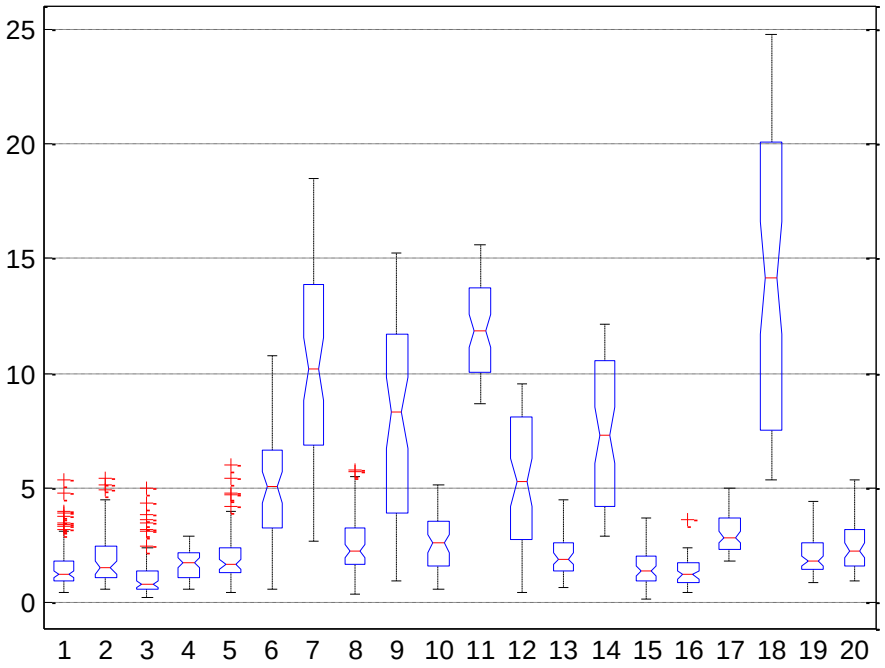
Boxplots für Testdaten aus 20 Testläufen

Die Boxplots zeigen den durchschnittlichen Fehler der einzelnen Datenpunkte jedes berechneten best-fitting Shapes der Testdaten zu den zugehörigen Landmarks in 20 Testläufen für a) 9er-Nachbarschaft und b) 25er-Nachbarschaft.

a) 20 Testläufe mit 9er-Nachbarschaft



b) 20 Testläufe mit 25er-Nachbarschaft



Aus den Boxplots lässt sich ableiten, dass die Shapes 7, 9, 11, 14 und 18 die höchsten durchschnittlichen Fehler pro Datenpunkt aufweisen, was in einer fehlerhaften Segmentierung resultiert, siehe folgende Ausgabebilder, die links das best-fitting Shape über dem Originalbild und rechts das best-fitting Shape über der verwendeten Prediction anzeigen.

Segmentierungen mit hohem durchschnittlichem Fehler

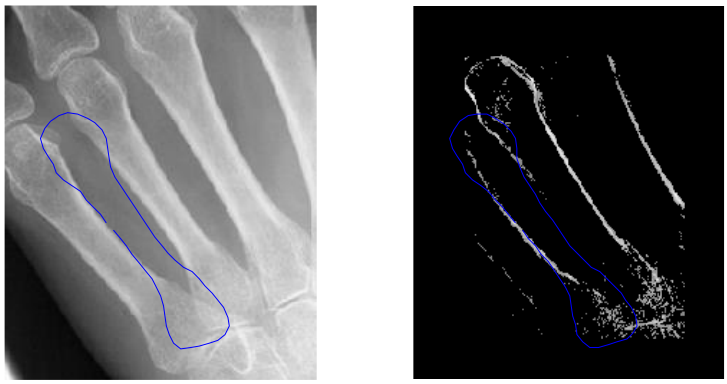


Abbildung 8: Testbild 18 (Datensatz 48)

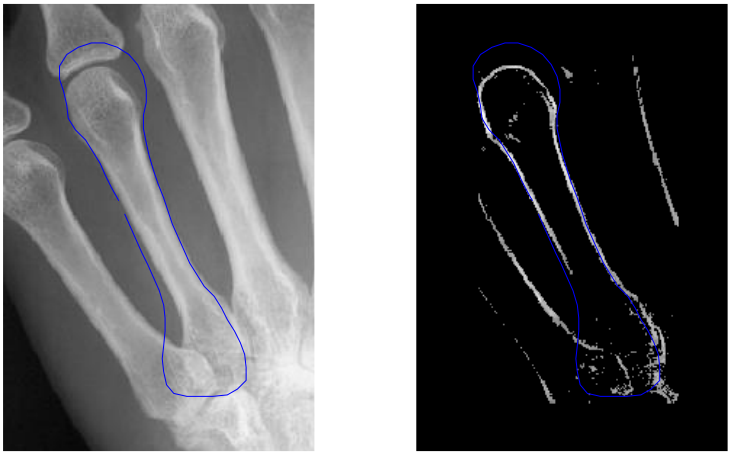


Abbildung 9: Testbild 14 (Datensatz 44)

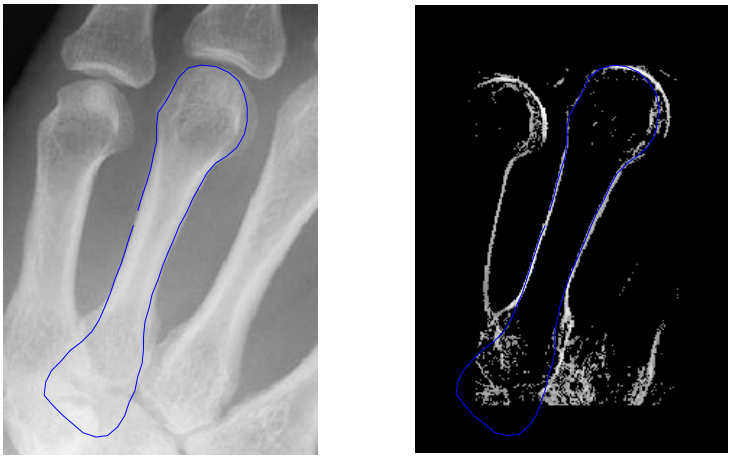
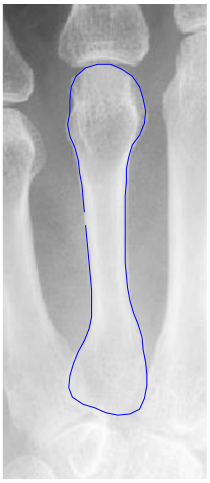


Abbildung 10: Testbild 7 (Datensatz 37)

Segmentierungen mit geringem durchschnittlichem Fehler



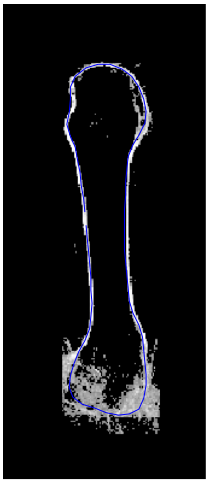


Abbildung 11: Testbild 1 (Datensatz 31)

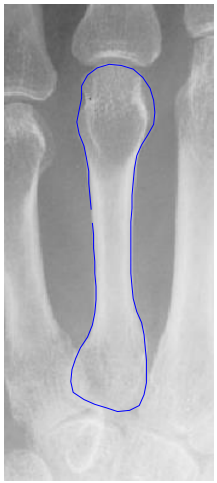


Abbildung 12: Testbild 13 (Datensatz 43)

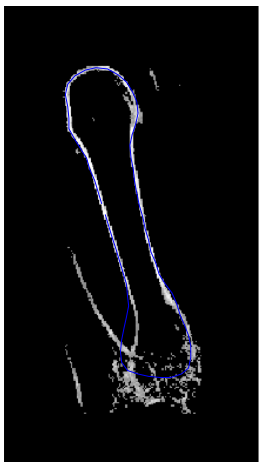


Abbildung 13: Testbild 12 (Datensatz 42)

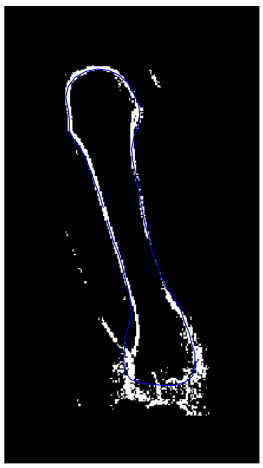


Abbildung 14: Testbild 12 (Datensatz 42)