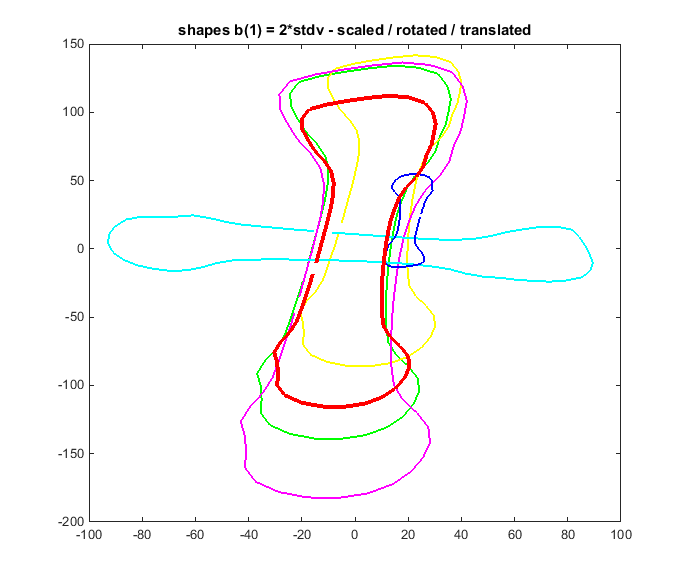
Medizinische Bildverarbeitung

Übung 2 – Particle Filters

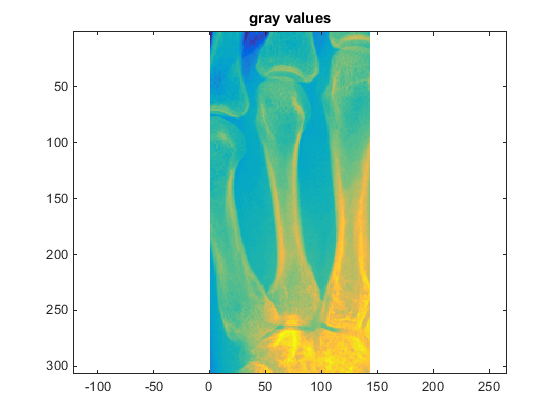
Lucas Dobler, Daniel Gehrer

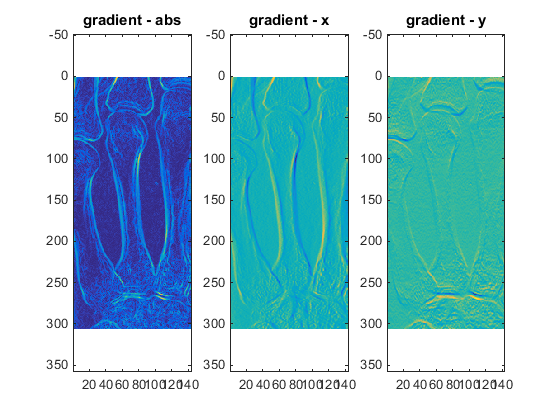
# Shape-Modell

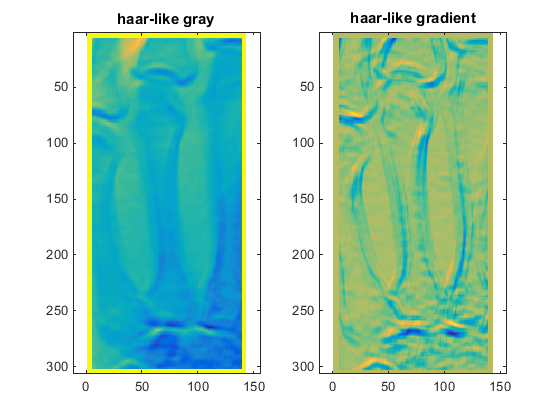


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Farbe | Skalierung | Rotation | x-Translation | y-Translation |
| Rot | originales Shape | | | |
| Blau | 1,2 | 0 | 0 | 0 |
| Grün | 1 | 0 | 10 | 30 |
| Gelb | 1,4 | 0 | 0 | -20 |
| Magenta | 0.8 | π / 2 | 0 | 0 |
| Cyan | 0,3 | π | 20 | 20 |

# Feature-Berechnung





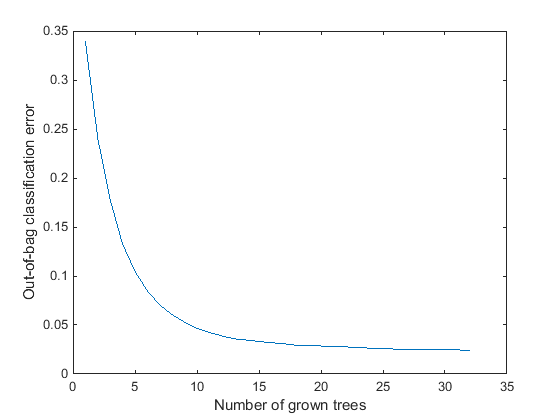


**zusätzliche Features:**

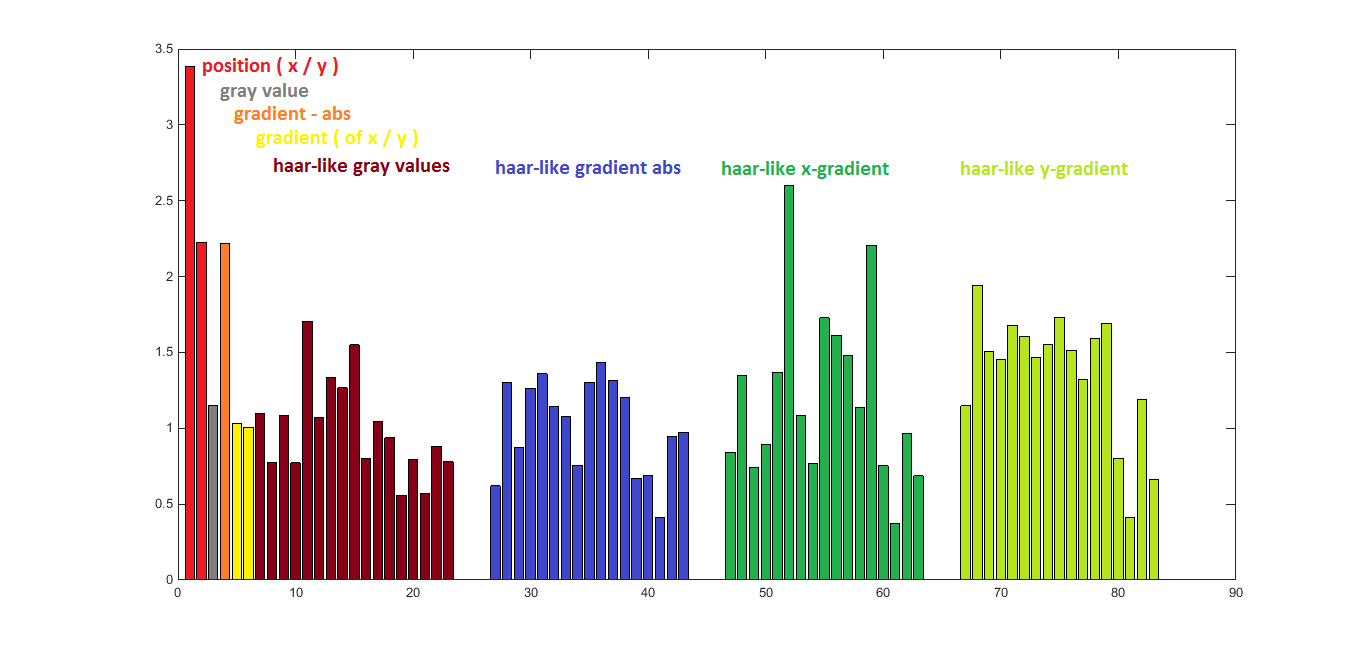
* Haar-like Features des Gradienten in x-Richtung
* Haar-like Features des Gradienten in y-Richtung

# Klassifikation & Feature-Selection

## Out-of-Bag Fehler



Es wurden 32 Bäume generiert. Im Diagramm ist zu erkennen, dass diese Anzahl genügend ist, da sich die Fehlerrate bei ca. 0,03 einpendelt. Aus Effizienzgründen würden bereits 27 Bäume genügen, da sich von dort an der Fehler nur noch marginal verändert.



Das Diagramm gibt das Maß für die Wichtigkeit jedes Features im Random Forrest an. Dabei auffällig ist, dass der x-Wert eine bedeutend größere Rolle spielt, als der y-Wert. Der Grau-Wert hingegen spielt eine signifikant unwichtigere Rolle. Des Weiteren ist auffällig, dass die haar-like features des x- und y-Gradienten bedeutender als die haar-like features der Gradientenstärke und des Grau-Werts sind.

# Shape Particle Filters

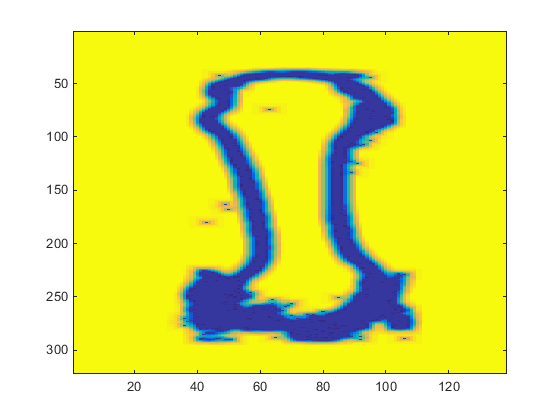
## Kostenfunktionen

### Prediction Surrounding

Das Ergebnis der Prediction des TreeBaggers wird erweitert. Wird ‚Contour‘ vorhergesagt, ist der Wert 1. Wird ‚Background‘ vorhergesagt, ist der Wert 0. Nun wird im Umkreis von allen Feldern mit dem Wert 1 der Wert erhöht.

Im 3x 3-Pixel Umfeld wird der Wert jeweils um 0,18 erhöht.  
Im 5x 5-Pixel Umfeld wird der Wert jeweils um 0,08 erhöht.  
Im 7x 7-Pixel Umfeld wird der Wert jeweils um 0,03 erhöht.  
Im 11x11-Pixel Umfeld wird der Wert jeweils um 0,01 erhöht.

Der maximal mögliche Wert in einem Feld ist jedoch 1. Die Kosten für jedes Pixel ist nun dieser Wert invertiert. Die Kosten für das Shape ist die Summe aus allen Kosten auf denen ein Punkt des Shapes liegt. Liegt mindestens ein Punkt des Shapes außerhalb des Bildbereichs, werden Kosten der Distanz zwischen Bildrand und Punkt mit 1012 multipliziert.

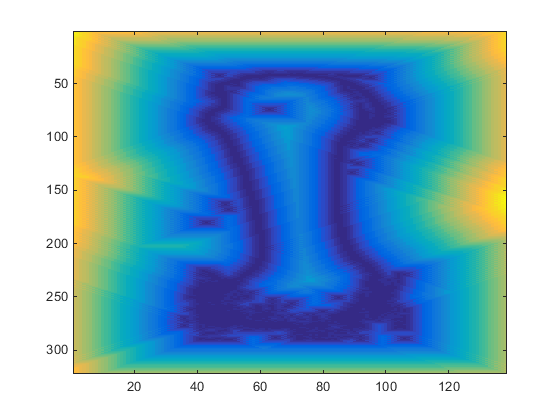
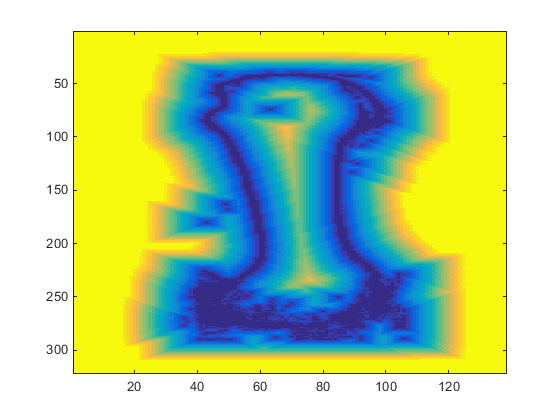


### Grow Prediction

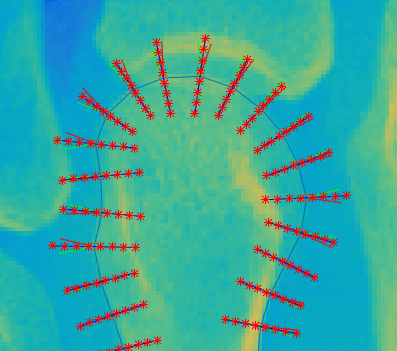
Das Ergebnis der Prediction des TreeBaggers wird erweitert. Wird ‚Contour‘ vorhergesagt, sind die Kosten als 0 definiert. Wird ‚Background‘ vorhergesagt, sind die Kosten als 1000 definiert. Nun wird jedes Pixel (aufsteigend nach den aktuellen Kosten) evaluiert.

Im 3x3-Pixel Umfeld werden die Kosten an die Kosten des Pixels angepasst (abhängig von der angegebenen Auflösung)

Die maximal möglichen Kosten für ein Pixel betragen 1000, Pixel die von TreeBagger vorhergesagt wurden, betragen 0. Die Kosten für das Shape ist die Summe aus allen Kosten auf denen ein Punkt des Shapes liegt. Liegt mindestens ein Punkt des Shapes außerhalb des Bildbereichs, werden Kosten der Distanz zwischen Bildrand und Punkt mit 1012 multipliziert.

 Auflösung: 200 Auflösung: 20

### Landmark Profiles

Diese Methode ist an die Methode in „The Use of Active Shape Models For Locating Structures in Medical Images“ von T.F. Cootes et al. angelehnt.

Für jedes Landmark wird eine Normale berechnet. Die Linie reicht von 10 Pixel außerhalb des Shapes bis 10 Pixel innerhalb des Shapes. Für diese Linie wird ein Profil mit 8 Werten erstellt. Das Profil ergibt sich aus der Differenz der Grauwerte zweier benachbarter Profil-Punkte auf dieser Linie.

Während des Trainings wird für jedes Landmark ein mittleres Profil berechnet und dann die Kovarianz-Matrix über alle diese mittleren Profile bestimmt.

In der Cost-Function werden dann für das aktuelle Shape die Profile bestimmt. Über die Kovarianz-Matrix kann nun für jedes Landmark der Fehler bestimmt werden. Die Kosten für das aktuelle Shape ergeben sich aus der Summe der Absolut Beträge aller Fehler.

### Contour Distance

Es wird die Prediction verwendet und für jede Landmark den Abstand zum nächsten vorausgesagten Konturpunkt berechnet. Die Kosten werden durch die Summe aller Abstände bestimmt. Da die Prediction aber sehr viele Punkte aufweist, ist dieser Algorithmus extrem langsam.

### On Contour

Anstatt die binäre Prediction zu verwenden, verwendet diese Cost Function eine Prediction, die die Wahrscheinlichkeit für jeden Pixel angibt, ob sich dort eine Kontur befindet.

Die Kosten ergeben sich in dem die Wahrscheinlichkeiten, dass eine Landmark auf dem Background (und nicht auf der Kontur liegt), für alle Landmarks aufsummiert werden.

### Test

Kostenfunktion, die lediglich zum Testen der Optimierung verwendet wird. Die Kosten betragen die Summe der Distanzen zwischen den Punkten des berechneten Shape und des idealen Shape. Da diese Funktion die Lösung bereits kennt, funktioniert sie natürlich sehr gut und kann als Referenz für eine bestmögliche Lösung verwendet werden.

## Resultate – Bewertung

Die Bewertung der einzelnen Cost Functions wird anhand von der benötigten Zeit um die richtigen Parameter mit minimalen Kosten zu finden und dem Fehler der resultierenden Landmarks bestimmt.

### Zeit

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bild** | **Prediction Sur.** | **Grow Prediction** | **Landmark Prof.** | **On Contour** | **Test (Referenz)** |
| 31 | 0,98 | 1,02 | 33,60 | 0,90 | 0,82 |
| 32 | 1,00 | 1,02 | 34,22 | 0,83 | 0,86 |
| 33 | 1,01 | 1,02 | 32,92 | 0,90 | 0,82 |
| 34 | 1,02 | 1,06 | 32,67 | 0,84 | 0,84 |
| 35 | 1,06 | 1,02 | 33,51 | 0,82 | 0,83 |
| 36 | 1,05 | 1,03 | 30,27 | 0,89 | 0,85 |
| 37 | 1,03 | 1,04 | 33,13 | 0,91 | 0,89 |
| 38 | 1,00 | 1,06 | 31,27 | 0,84 | 0,86 |
| 39 | 1,03 | 1,04 | 31,28 | 0,81 | 0,82 |
| 40 | 0,95 | 1,01 | 31,69 | 0,85 | 0,87 |
| 41 | 1,06 | 1,05 | 31,50 | 0,88 | 0,82 |
| 42 | 1,07 | 1,00 | 33,83 | 0,82 | 0,86 |
| 43 | 0,96 | 1,04 | 32,71 | 0,90 | 0,81 |
| 44 | 0,94 | 1,01 | 33,89 | 0,85 | 0,86 |
| 45 | 1,00 | 1,02 | 34,86 | 0,90 | 0,84 |
| 46 | 1,00 | 1,04 | 36,63 | 0,89 | 0,84 |
| 47 | 1,01 | 0,95 | 34,52 | 0,83 | 0,83 |
| 48 | 0,95 | 1,01 | 29,15 | 0,92 | 0,81 |
| 49 | 0,99 | 0,99 | 34,34 | 0,80 | 0,80 |
| 50 | 1,02 | 1,02 | 36,36 | 0,81 | 0,83 |
| **Durchschnitt** | ***1,01*** | ***1,02*** | ***33,12*** | ***0,86*** | ***0,84*** |
| **Std. Dev.** | ***0,04*** | ***0,03*** | ***1,84*** | ***0,04*** | ***0,02*** |

An der geringen Standardabweichung ist zu erkennen, dass sich die Laufzeit von Bild zu Bild nicht sonderlich stark unterscheiden. Jede Cost Function benötigt ca. 1 Sekunde um das Shape zuzuordnen. „Landmark Profiles“ ist hier ein Ausreißer, hier werden meist über 30 Sekunden pro Bild benötigt.

### Fehler

Der Fehler wird angegeben als der Durchschnittliche Abstand jedes Landmarks von seiner Idealen Position (Ground Truth).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bild** | **Prediction Sur.** | **Grow Prediction** | **Landmark Prof.** | **On Contour** | **Test (Referenz)** |
| 31 | 1,6423 | 3,0705 | 1,5721 | 2,8562 | 1,2736 |
| 32 | 3,1391 | 2,8950 | 9,8589 | 5,7619 | 2,3998 |
| 33 | 2,7909 | 1,5091 | 1,7587 | 1,7002 | 1,3260 |
| 34 | 3,9415 | 3,0140 | 12,8802 | 7,0639 | 2,1513 |
| 35 | 5,1679 | 5,5565 | 15,2384 | 11,2520 | 3,1960 |
| 36 | 6,3829 | 7,0049 | 19,4793 | 17,1606 | 4,5892 |
| 37 | 6,6307 | 7,4868 | 18,3765 | 16,2661 | 3,9165 |
| 38 | 5,5499 | 5,4467 | 15,1322 | 9,8720 | 3,2599 |
| 39 | 4,0641 | 7,8218 | 7,8960 | 12,3596 | 2,9156 |
| 40 | 2,2539 | 4,2018 | 8,4119 | 6,0939 | 1,7194 |
| 41 | 9,0471 | 9,1292 | 8,0308 | 9,1714 | 2,8054 |
| 42 | 3,3698 | 7,7070 | 8,9332 | 9,8040 | 2,1452 |
| 43 | 2,6093 | 6,9191 | 7,7266 | 9,1997 | 2,5290 |
| 44 | 7,7498 | 7,6795 | 6,2424 | 6,2993 | 1,9480 |
| 45 | 2,6149 | 2,3684 | 2,8103 | 3,2246 | 2,0285 |
| 46 | 1,6912 | 1,6113 | 1,9763 | 2,8536 | 1,4951 |
| 47 | 2,7346 | 2,6368 | 4,5726 | 9,6313 | 1,9536 |
| 48 | 7,8221 | 4,0082 | 2,0591 | 2,9973 | 1,4238 |
| 49 | 2,1699 | 2,2019 | 2,3288 | 9,9223 | 1,9872 |
| 50 | 2,1023 | 1,7660 | 1,6681 | 2,3749 | 1,6518 |
| **Durchschnitt** | ***4,1737*** | ***4,7017*** | ***7,8476*** | ***7,7932*** | ***2,3357*** |
| **Varianz** | ***2,2232*** | ***2,4528*** | ***5,6414*** | ***4,3670*** | ***0,8627*** |
| **Minimum** | ***1,6912*** | ***1,5091*** | ***1,6681*** | ***1,7002*** | ***1,3260*** |
| **Maximum** | ***9,0471*** | ***9,1292*** | ***19,4793*** | ***17,1606*** | ***4,5892*** |

Die Farbkodierung der Fehler ist in dieser Tabelle für jede Cost Function einzeln berechnet um die stärksten und schwächsten Bilder der jeweiligen Cost Function einfach zu erkennen.

Am durchscnittlichen Fehler kann man erkennen, dass „Prediction Surrounding“ durchschnittlich am besten Abschneidet, gleich gefolgt von „Grow Prediction“, was nicht verwunderlich ist, da beide Funktionen eng verwandt sind. Durchschnittlich am schlechtesten ist „Landmark Profiles“, ähnlich wie „On Contour“.

„Landmark Profiles“ hat aber auch stärken, z.B. bei Bild 48 wo „Prediction Surrounding“ um einiges schlechter optimiert. Die zusätzliche Rechenzeit, die „Landmark Profiles“ benötigt, hat aber im Durchschnitt keinen Mehrnutzen. Die einzelnen Cost Functions haben verschiedene Stärken und Schwächen. Beispielsweise hat „Grow Prediction“ bei Bild 33 den kleinsten Fehler überhaupt, allerdings ist es insgesamt nicht die beste Cost Function.

|  |  |
| --- | --- |
| Bei Bild 48 schneidet ausnahmsweise „Landmark Profiles“ besser ab: | |
|  |  |
| Bild 48 mit „Landmark Profiles“ | Bild 48 mit „Prediction Surrounding“ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bild** | **Prediction Sur.** | **Grow Prediction** | **Landmark Prof.** | **On Contour** | **Test (Referenz)** |
| 31 | 1,6423 | 3,0705 | 1,5721 | 2,8562 | 1,2736 |
| 32 | 3,1391 | 2,8950 | 9,8589 | 5,7619 | 2,3998 |
| 33 | 2,7909 | 1,5091 | 1,7587 | 1,7002 | 1,3260 |
| 34 | 3,9415 | 3,0140 | 12,8802 | 7,0639 | 2,1513 |
| 35 | 5,1679 | 5,5565 | 15,2384 | 11,2520 | 3,1960 |
| 36 | 6,3829 | 7,0049 | 19,4793 | 17,1606 | 4,5892 |
| 37 | 6,6307 | 7,4868 | 18,3765 | 16,2661 | 3,9165 |
| 38 | 5,5499 | 5,4467 | 15,1322 | 9,8720 | 3,2599 |
| 39 | 4,0641 | 7,8218 | 7,8960 | 12,3596 | 2,9156 |
| 40 | 2,2539 | 4,2018 | 8,4119 | 6,0939 | 1,7194 |
| 41 | 9,0471 | 9,1292 | 8,0308 | 9,1714 | 2,8054 |
| 42 | 3,3698 | 7,7070 | 8,9332 | 9,8040 | 2,1452 |
| 43 | 2,6093 | 6,9191 | 7,7266 | 9,1997 | 2,5290 |
| 44 | 7,7498 | 7,6795 | 6,2424 | 6,2993 | 1,9480 |
| 45 | 2,6149 | 2,3684 | 2,8103 | 3,2246 | 2,0285 |
| 46 | 1,6912 | 1,6113 | 1,9763 | 2,8536 | 1,4951 |
| 47 | 2,7346 | 2,6368 | 4,5726 | 9,6313 | 1,9536 |
| 48 | 7,8221 | 4,0082 | 2,0591 | 2,9973 | 1,4238 |
| 49 | 2,1699 | 2,2019 | 2,3288 | 9,9223 | 1,9872 |
| 50 | 2,1023 | 1,7660 | 1,6681 | 2,3749 | 1,6518 |
| **Durchschnitt** | ***4,1737*** | ***4,7017*** | ***7,8476*** | ***7,7932*** | ***2,3357*** |
| **Varianz** | ***2,2232*** | ***2,4528*** | ***5,6414*** | ***4,3670*** | ***0,8627*** |
| **Minimum** | ***1,6912*** | ***1,5091*** | ***1,6681*** | ***1,7002*** | ***1,3260*** |
| **Maximum** | ***9,0471*** | ***9,1292*** | ***19,4793*** | ***17,1606*** | ***4,5892*** |

In dieser Tabelle ist die Farbkodierung der Fehler über alle Cost Functions berechnet. Es ist erkennbar, dass „Prediction Surrounding“ die kleinsten Fehler aufweist. Grow Prediction ist ähnlich gut.

|  |
| --- |
|  |
| Einige mit „Prediction Surrounding“ gematchen Bilder. |