Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Computer Vision
Übung - Teil 1
Gruppe 1

Aufgabe 1 - Colorizing Images
Detaillierte Vorgehensweise

Meine Implementierung besteht aus 2 Hilfsfunktionen, ‚loadImages' und ‚generatePyramids', und 2 Matlab Klassen, ‚colorize' und ‚bonus'.

loadImages

Diese Funktion wird in beiden Klassen ‚colorize' und ‚bonus' verwendet und regelt das einlesen der channel Bilder aus einem Ordner. Die Funktion returniert 3 Arrays, images_R, images_G und images_B. Dabei enthält images_R alle roten Channel Images, images_G alle grünen Channel Images und images_B alle blauen.

Colorize

Diese Klasse beinhaltet die Implementierung des ersten Assignments – Colorizing Images. Die Images werden zuerst eingelesen, mittels der loadImages Funktion. Nachher wir für eine einzige Image des blauen Channel als fix betrachtet. (Der Code ist jedoch sehr einfach erweiterbar, so dass mehrere bunte Images berechnet werden.)
Dann werden in einem Kernel von 31 x 31 Pixel unterschiedliche Verschiebungen überprüft und letztendlich die beste Verschiebung der roten Channel Image und der Grünen Channel Image ausgewählt. Die 3 ausgewählten Images werden dann zusammengeknüpft (mittels concat) und so entsteht eine farbige Image.
Um die beste Verschiebung zu berechnen, verwende ich die Matlab Funktion circshift, welche eine Image um einen angegebenen k-Wert nach rechts verschoben und um einen angegebenen j-Wert nach unten verschoben. Anschließend berechnet man für jede Verschiebung der Channel die Korrelation zwischen den Channel Bildern, mithilfe der Matlab Funktion corr2. Je ähnlicher sich 2 Bilder sind, desto mehr strebt das Ergebnis von corr2 nach -1 oder +1. 0 bedeutet keine Korrelation.

generatePyramids

Diese Funktion generiert die Gauss-Pyramide einer eingegebenen Image, welche üblicherweise eine Channel Image ist, in meiner Implementierung. Jedoch ist beim Aufruf der Funktion noch nicht klar wie viele Levels die resultierende Pyramide haben wird. Die Funktion generiert jedoch solange neue Levels der Pyramide, bis die Breite der Image auf einem bestimmten Level nicht kleiner als (4 * die Kernelbreite) ist, in der nach der Korrelation der verschobenen Channel Bilder gesucht wird.

bonus

Diese Klasse beinhaltet die Implementierung des Bonus Tasks von Assignment 1, und präsentiert eine Verbesserung der colorize-Klasse, die auch für sehr viel größere Bilder gute Resultate in einer akzeptablen Laufzeit liefert.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Die Verbesserung der Laufzeit besteht darin, dass zuerst Gauss-Pyramiden aus den Channel Images erzeugt werden, mithilfe der Funktion generatePyramids. Außerdem die Kernelsize in beiden Richtungen wurde von [-15, 15] auf [-2, 2] reduziert. Weiterhin fängt die Suche nach der besten Verschiebung in dem größten Level der Pyramide, also im kleinsten Bild an.
Die Koordinaten der besten Verschiebung werden gespeichert und noch bevor der nächste Level bearbeitet wird, wird dieser schon verschoben. Außerdem speichere ich mir in dieser Implementierung auch die besten Koordinaten des vorherigen Levels so dass, sowohl die besten Koordinaten des aktuellen Levels als auch die beste Verschiebung des vorherigen Levels in die Verschiebung des nächsten Levels miteinfließen (Verschiebung um (Koordinaten der aktuellen besten Verschiebung * 2) + (Koordinaten der vorherigen besten Verschiebung * 2)).

3 Kolorierungen vom Assignment 1 mit und ohne Alignment, [-15, 15] Kernel

Die folgenden 6 Bilder repräsentieren kolorierte Bilder aus dem gegebenen urspünglichen Set, mit kleinerer Auflösung.
In der rechten Spalte können die zusammengeknüpften Bilder ohne Verschiebung betrachtet werden, während die linke Spalte kolorierte und korrekt verschobene Bilder veranschaulicht.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

3 Kolorierungen vom Bonus Task, [-2, 2] Kernel, Gaussian Pyramids, Bilder mit Hochauflösung

Die folgenden 3 Bilder repräsentieren kolorierte Bilder aus dem erweiterten Set, mit höherer Auflösung. Ihre Kolorierung wurde durch die verbesserte Pyramiden-Methode ermöglicht.

Um diese Bilder mit hocher Auflösung zu extrahieren, wurden sie zuerst von der folgenden Seite heruntergeladen, die in der Angabe empfohlen wurde :
http://www.loc.gov/pictures/search/?q=Prokudin-Gorski%C4%AD%2C+Serge%C4%AD+Mikha%C4%ADlovich%2C+1863-1944&fi=name&op=EQUAL&st=grid
Dannach wurden sie in dem Fotoverarbeitungsprogramm Photoscape gecroppt und so gut wie möglich angepasst, jedoch bestanden dann immer noch minimale Verschiebungen. Diese wurden dann mithilfe der Implementierung des Bonus Tasks angepasst und zusammengefügt.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Laufzeitvergleich zwischen der ursprünglichen und der verbesserten Implementierung

- Laufzeit der ursprünglichen Implementierung / kleine Bilder: ~**4.4 Sekunden**
- Verbesserte Laufzeit / kleine Bilder: ~**0.32 Sekunden**
- Laufzeit der ursprünglichen Implementierung / hohe Auflösung Bilder: **einige Stunden**
- Verbesserte Laufzeit / hohe Auflösung Bilder: ~**25.4 Sekunden**

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Aufgabe 2 - Image Segmentation by K-means Clustering

Detaillierte Vorgehensweise

Zunächst werden k viele Centroiden zufällig im Bild gewählt. Diese enthalten, im Falle der 3D Version die Farbwerte und bei der 5D Version die Farbwerte + die xy Koordinaten (normalisiert mit der jeweiligen Bildbreite bzw. höhe).

Nun werden mehrere verschachtelte Schleifen durchlaufen. Die Äußerste vergleicht die Differenz der Summe aller Distanzen aller Punkte zu jedem Cluster, mit der Summe des vorherigen Durchlaufes (im ersten Durchlauf ist diese Differenz unendlich). Diese Schleife terminiert wenn diese Differenz kleiner ist als ein angegebener Epsilon-wert.

Die nächste beiden Schleifen durchlaufen die Bildbreite und die Bildhöhe um die Datenpunkte aus dem Bild zu extrahieren.

In der innersten Schleife werden nun die Abstände der Datenpunkte im Bild mit jedem Centroid berechnet (mithilfe der Quadratischen Differenz der Datenvektoren) und der Cetroid bestimmt, zu welchen die jeweiligen Datenpunkte den minimalen Abstand haben.
Wurde die Zuordnung beendet, werden neue Centroiden aus Clustern berechnet:
Hat jeder Centroid mindestens einen Punkt im Bild, welches im zugeordnet ist, wird der Mittelwert aller Werte zum neuen Centroid.
Hat ein Cetroid keinen ihm zugewiesenen Wert (also ist ein Cluster leer), wird der Datenpunkt, der die größte Distanz zu allen Punkten hat, zum neuen Centroiden gewählt.

Am Ende haben wir noch eine Zeichenroutine implementiert, welche die Cluster mit der inversen Farbe umrandet.

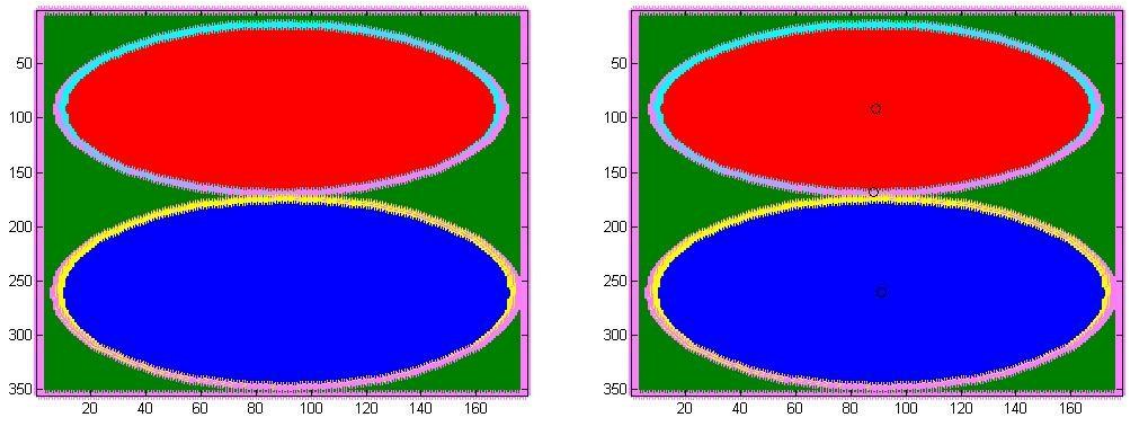
Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Issues to be addressed in the report:

• Show the results for all images in the case of 3D data points as well as 5D data points (using a fixed value of K). Discuss the results. Which data representation is better in your opinion?

Als k wurde 3 gewählt (mehr würde beim Bild „simple“ keinen Sinn machen, da nur 3 Farbwerte enthalten).

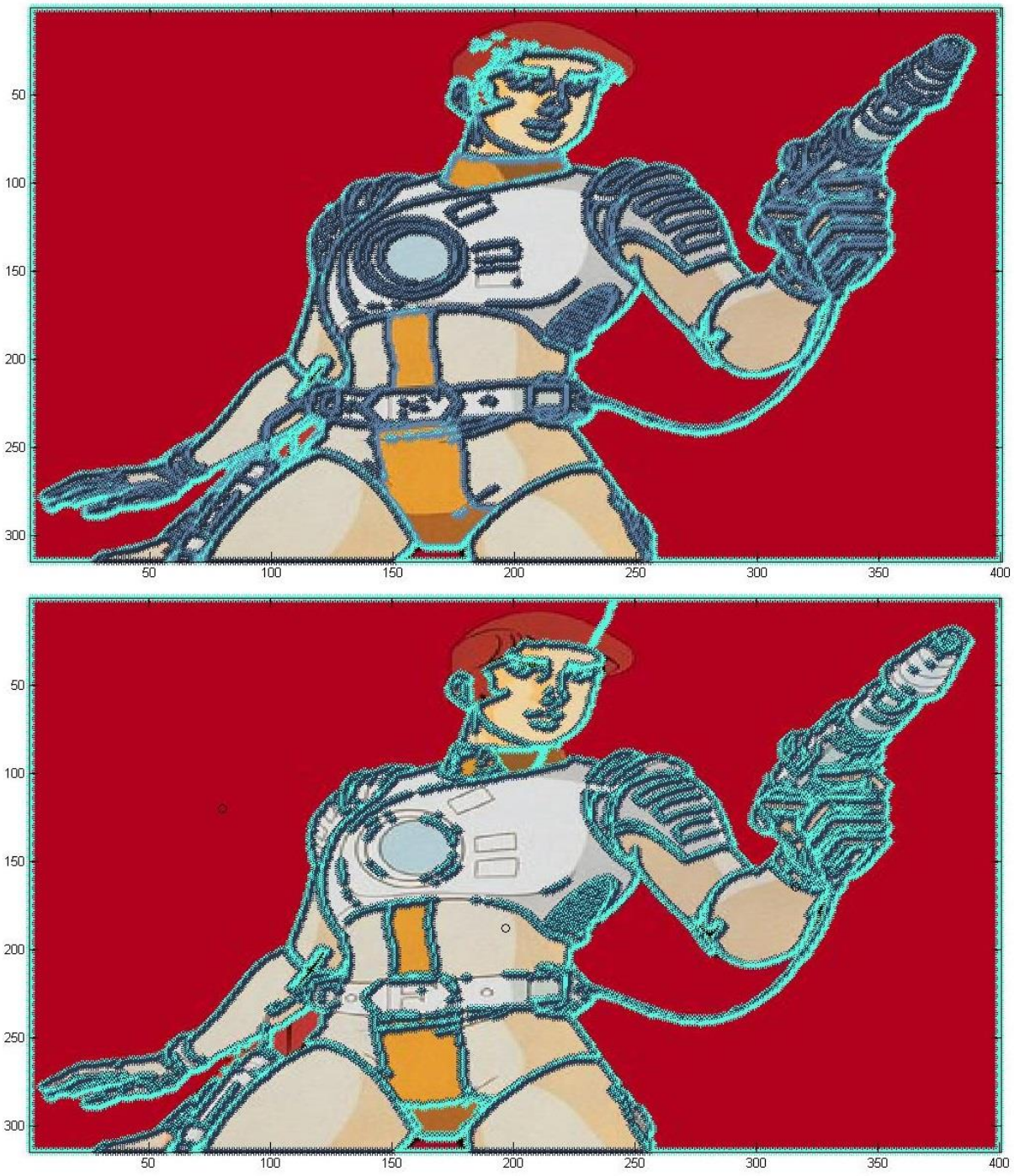
Datei „simple.png“:



(links 3D, rechts 5D (Centroiden bei 5D mit Kreis approximiert))
In diesem Bild gab es keine Unterschiede der beiden Versionen.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

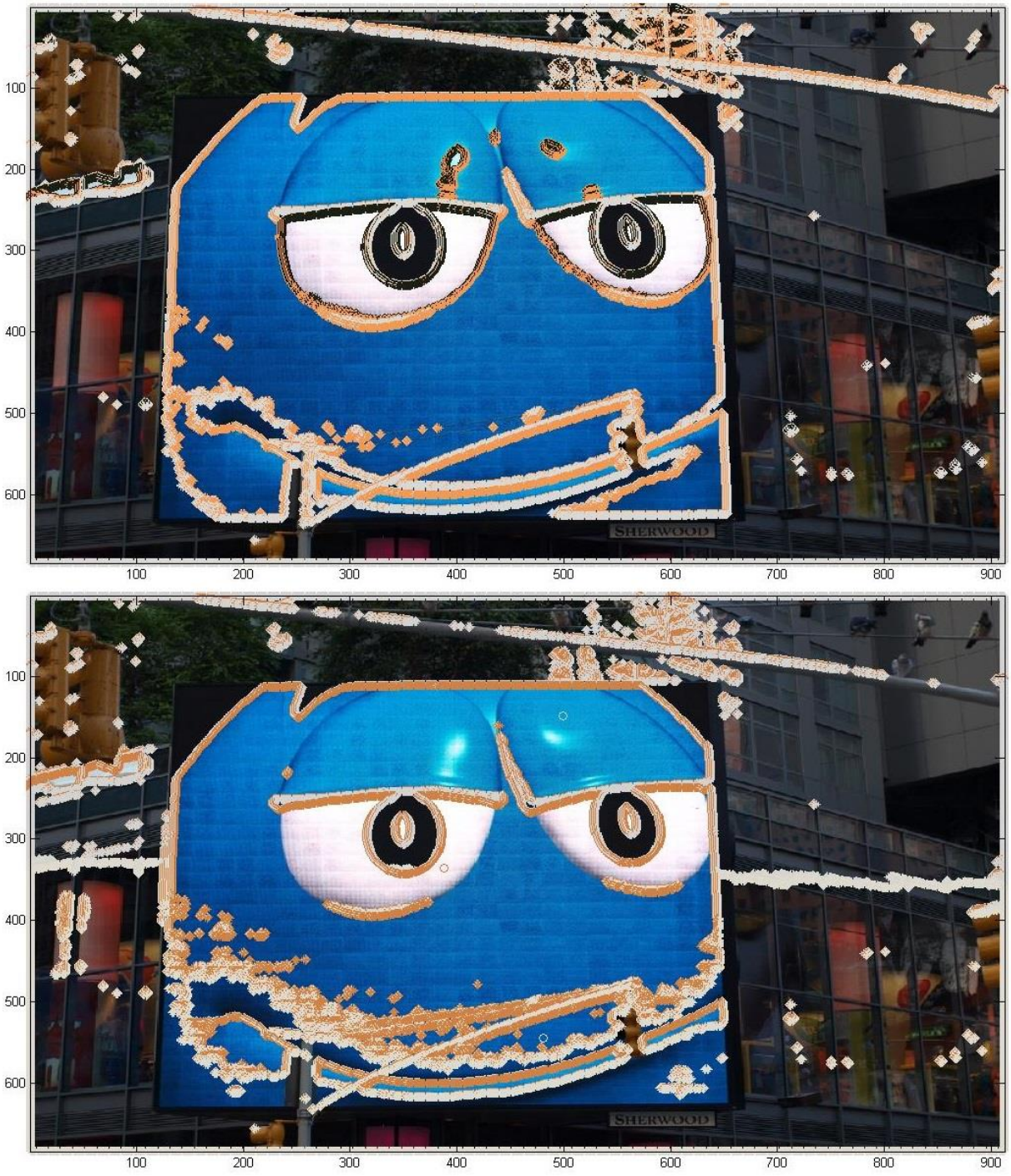
Datei „future.jpg“:



(oben 3D, unten 5D (Centroiden bei 5D mit Kreis approximiert))

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Datei „mm.jpg“:



(oben 3D, unten 5D (Centroiden bei 5D mit Kreis approximiert))

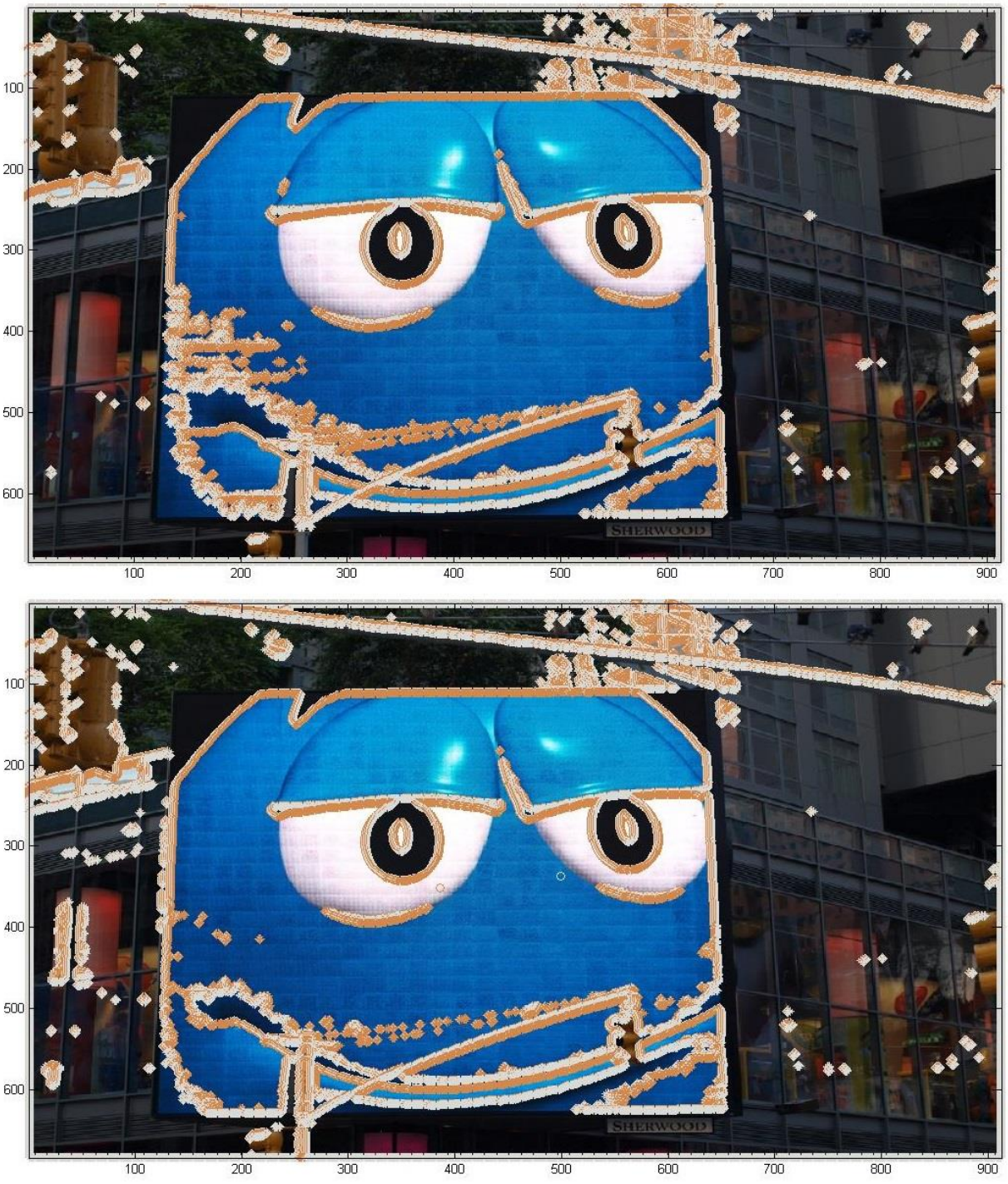
Beim simple Bild sind keine Unterschiede erkennbar. Das ist durchaus denkbar, da es in diesem Bild nur 3 verschiedene Farbwerte existieren.
Im Future Bild sind die Unterschiede am deutlichsten erkennbar. In 3D werden hier mehr kleinere Strukturen erkannt, während diese durch die Position eher zu einen anderen Cluster zugeteilt werden und daher zusammengehörende Strukturen auch eher zusammen bleiben.
Im mm Bild ist dies auch gut an den Augen erkennbar die in 5D zum blauen mm dazugerechnet werden während sie in 3D einem anderen Cluster zugeteilt sind. Andererseits wird der Teil des mm der von der Ampel abgetrennt wird in 5D auch nicht zu diesem dazu gezählt sondern für den Hintergrund gehalten. Dies wird in 3D besser erkannt.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Ausgehend von diesen Datensätzen sind wir der Meinung, dass die 3D Variante mehr Vorteile bietet, da man innere Strukturen, die noch zum Objekt gehören, auch mit einem Test ob ein Objekt ein anderes umschließt in einem weiteren Schritt zu dem Cluster zuweisen kann.

• Apply different values of K to the image mm.jpg and show the results for both 3D and 5D data points. Interpret the results.

K = 2:



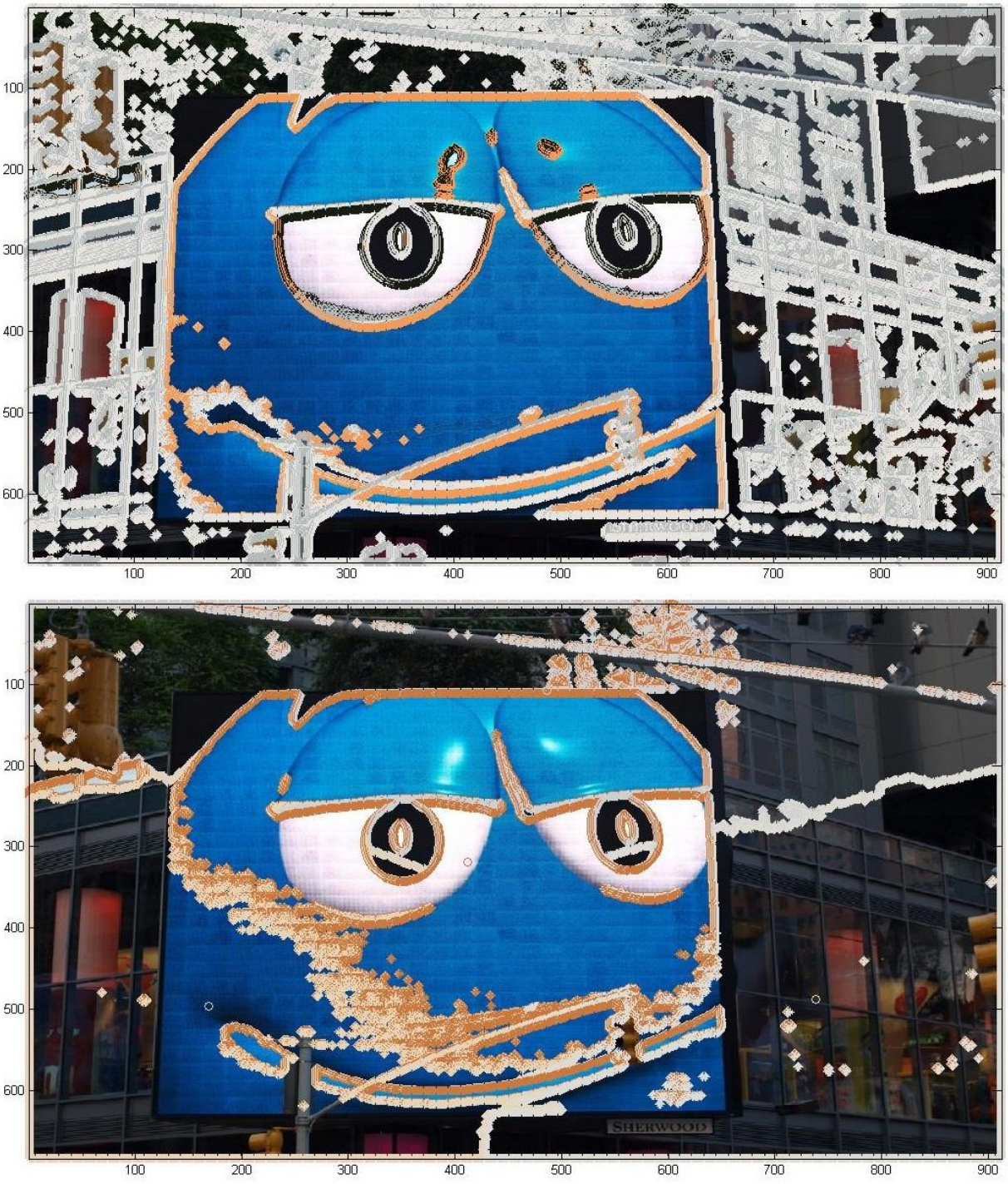
(oben 3D, unten 5D (Centroiden bei 5D mit Kreis approximiert))

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Das Resultat ist hier noch beinahe das gleiche wobei zu sehen ist, dass das blaue mm in 5D hier nun besser erkannt wird (gut sichtbar in der linken unteren Ecke des mm). Es werden aber auch mehr Teile des Hintergrunds dem Cluster zugewiesen (Fenster links in der Mitte).

K = 3:
Sichtbar oben im Beispiel mit dem festen K.
Das blaue mm wird nun in der 3D Version besser erkannt.

K = 4:

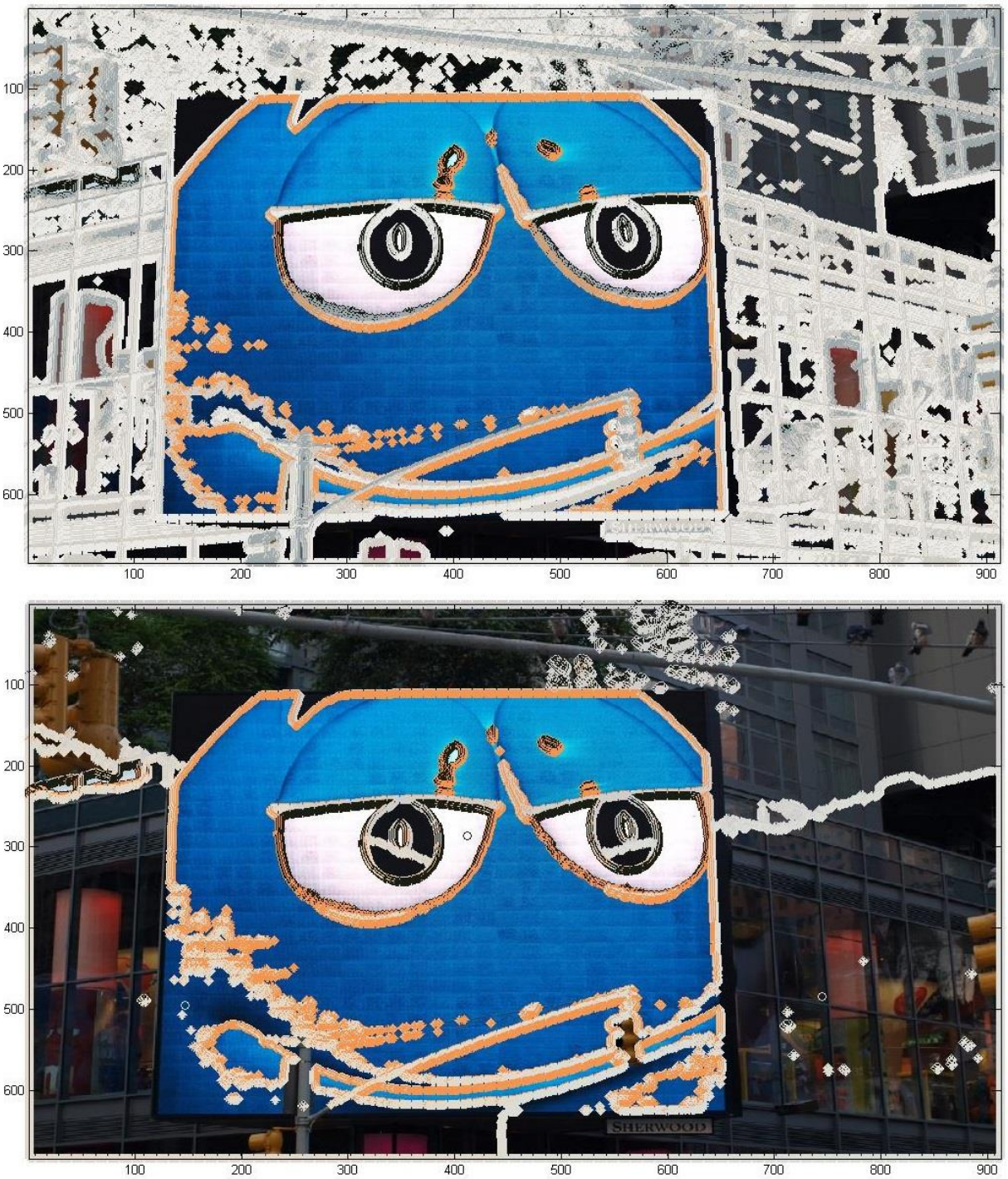


(oben 3D, unten 5D (Centroiden bei 5D mit Kreis approximiert))

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

In 3D werden nun sehr viele Details des Hauses zu den Clustern zugeteilt während sich an den Clustern vom blauen mm sich fast nichts ändert. Es wird weiterhin recht gut erkannt. Anders sieht es im Falle von 5D aus. Hier wird das mm jetzt noch schlechter erkannt während das Haus weiterhin nur in wenige Cluster aufgeteilt wird.

K = 5:



(oben 3D, unten 5D (Centroiden bei 5D mit Kreis approximiert))

3D Version weitgehend unverändert. Es sind noch weitere Cluster im Haus dazu gekommen. Im mm werden auch kleinere Strukturen einem anderen Cluster zugewiesen. Es ist

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470
anzunehmen, dass dieser Trent mit steigendem k noch ansteigen wird, da mit ansteigenden Clustern auch immer mehr Cluster in der Nähe der Farbe blau landen werden.
In 5D sieht man weiterhin recht große Cluster.

• Where do you see - based on your results - the strengths and the weaknesses of the method?

Das erste, dass sofort auffällt ist die lange Laufzeit dieser Methode. Durch die ineinander verschachtelten Schleifen kommt man auf I*m*n*K. Mit I die Anzahl der Iterationen bis die Differenz kleiner Epsilon ist. Der Höhe und Breite des Bildes m und n und der Anzahl der Cluster K. Wenn zu einem Objekt im Bild sehr viele verschiedene Farben vorkommen, wird es von dieser Methode nur sehr schwer bis gar nicht in einem Cluster zusammen gefasst werden.

Positiv ist, dass es eine sehr leicht verständliche Methode ist, die eher leicht zu implementieren ist. Durch Veränderung der Distanzberechnung kann man auch das Resultat beeinflussen. Es können auch Strukturen erkannt werden die nicht zusammenhängend sind.

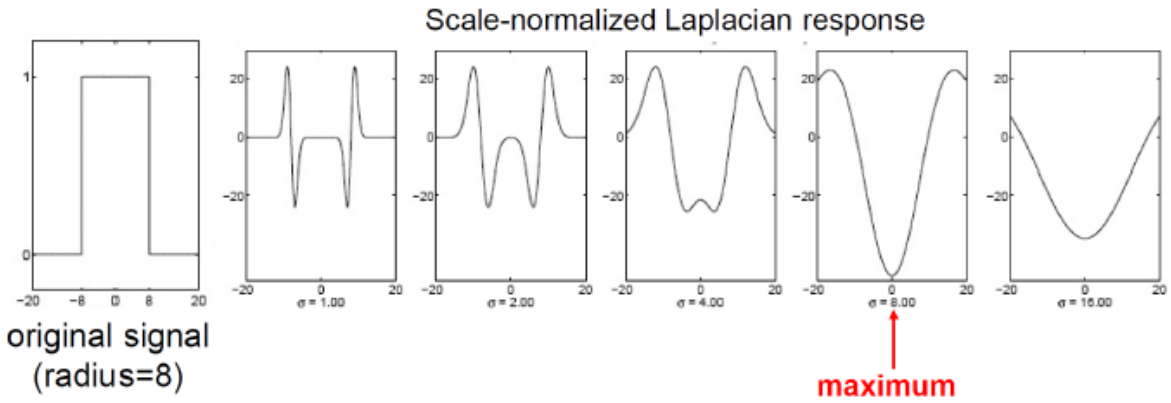
Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Aufgabe 3 - Scale-Invariant Blob Detection

Detaillierte Vorgehensweise

1. Berechnen der LoG Responses

Das Bild wird zunächst mit verschiedenen LoG Kernels gefiltert. Die Response des LoG Filters ist maximal wenn die Blobsize der größe des Filterkernels entspricht.



Für den ersten Filterkernel wird Sigma fixiert und für jede Iterationen mit einem konstanten Faktor multipliziert. Um die Filterantwort über die verschiedenen Kernels zu normalisieren wird mit Sigma² multipliziert. Zu niedrige Werte werden durch einen minimal Threshold ausgefiltert.

2. Non-maxima supression

Alleine mit dem Threshold haben wir jetzt sehr viele Blobkandidaten. Im Grunde wird jeder Pixel im Blob als eigener Blob erkannt, und die Radii der Blobs überlappen stark. Im Idealfall wollen wir aber pro Blob nur das Zentrum + Radius bestimmen.
Dieses Verhalten erreichen wir durch die Non-Maxima Suppression. Ein Blob wird also nur erkannt wenn die Filterantwort im 26 Neighborhood maximal ist. 8 Pixel des aktuellen Levels, und auch 9 Pixel der vor- und nachfolge Levels werden überprüft. Das kann sehr einfach durch Dilation umgesetzt werden, denn die Dilation eines Bildes berechnet für jeden Pixel eines Bildes das Maximum im definierten NHood. Ist der Wert der Filterantwort größer als der Wert im dilatierten Bild ist er ein lokales Maximum.

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Issues to be addressed in the report:
Apply the method to both the original images as well as to half-sized versions of them. Draw the detected blobs as circles with appropriate scale. Is the method able to find blobs in a scale-invariant way? If there are errors, what are the reasons for them?

Ja, die Blobs werden zum größten Teil unabhängig von ihrer Größe erkannt. Offensichtlich werden aber nicht alle Blobs in beiden Bildern erkannt. Die detektierten Blobgrößen sind durch unsere verwendeten Kernels fixiert, ist ein Blob also zu klein oder zu groß in einem Bild kann er nicht mehr erkannt werden.

Full Size Image

Halfsize Image

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Zebra.jpg Fullsize

Zebra.jpg Halfsize

Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

• Pick a detected keypoint and plot the response of the LoG for all scales in both image versions. The outcome should be a 2D plot where the x-axis represents the scale of the filter and the y-axis the filter response at the selected keypoint position. Describe and explain the difference between the two curves.

Die blaue Linie zeigt das originale Bild, die grüne Linie zeigt das gleiche Bild skaliert auf halbe Größe. Auf der Y-Achse ist die Antwort des LoG Filter aufgetragen. Die X-Achse zeigt dabei das Level in der Pyramide. Damit der gezeigte Pixel als Blob in betracht gezogen wird muss die Response des LoG Filters mindestens so groß sein wie der Blob Threshold (in rot eingezeichnet). Achtung! Das übersteigen des Thresholds bedeutet nicht automatisch dass der Pixel in beiden Bildern ein Blob ist, da diese Darstellung die lokalen Maxima nicht berücksichtigt.
Unter einer semantischen Betrachtungsweise gibt uns die Kurve an wie genau der jeweils betrachtete Blob sich an die Blobdetektionsgröße des jeweiligen Filter Kernels anpasst. Durch diesen Zusammenhang kann man vom Maximum der Kurven auf die jeweiligen Blob Radii rückschließen. Die Verschiebung der Kurven zeigt den Größenunterschied der Blobs in beiden Bildern.