# Image Stitching

**EDBV WS 2014/2015: AG_B4**

J. Sebastian Kirchner (0926076)
Hanna Huber (0925230)
Patrick Wahrmann (1327120)
Ernad Sehic (1227865)
Nikolaus Leopold (1327344)

6. Januar 2015

## 1 Gewählte Problemstellung

(1-1,5 Seiten)
entspricht dem (aktualisierten) Konzept

### 1.1 Ziel

Das Ziel ist es zwei Bilder mit überlappenden Bildbereichen (also gleiche Szene mit horizontal verschobener Kamera) anhand von in beiden Bildern vorhandenen Bildmerkmalen (Interest Points) zu einem Bildmosaik zusammenzufügen, sodass die zusammengehörenden Interest Points übereinanderliegen.

### 1.2 Eingabe

Als Eingabedaten werden zwei Bilddateien im Format JPEG oder PNG, die die unten beschriebenen Voraussetzungen erfüllen, verwendet.

### 1.3 Ausgabe

Das Ergebnis ist ein Bildmosaik im selben Format wie die Eingabebilder, das aus Transformation der Einzelbilder entsteht, sodass die gemeinsamen Merkmale übereinstimmen.

## 1.4 Voraussetzungen und Bedingungen

Der Einfachheit halber werden folgende Eigenschaften der Eingabebilder vorausgesetzt:

- Die Bilder sind Abbildungen der gleichen Szene, die nur horizontal versetzt sind
- Es müssen überlappende Bereiche in den Bildern vorhanden sein, die kontrastreiche Interest Points aufweisen
- Die Belichtungsverhältnisse müssen in beiden Bildern möglichst gleich sein (ähnliche Zeit, Ausrichtung bezüglich der Lichtquelle, gleiche Kamera!)
- Perspektivische Verzerrungen sollten möglichst vermieden werden, indem entfernte Motive verwendet werden und die Brennweite möglichst hoch gewählt wird.
- Möglichst anorganische Strukturen mit klaren Umrissen als zentrales Motiv (Bäume etc. nur am Rande)
- Keine beweglichen Objekte, durch die Interests Points verdeckt werden könnten.

## 1.5 Methodik

Der folgende Verfahrensablauf wurde implementiert:

- Die Bilder werden eingelesen.
- Dann wird eine Bildpyramide (Difference of Gaussian, kurz DoG) für SIFT aufgebaut, mittels wiederholt ausgeführtem Gauss-Filter und Downsampling.
- Mithilfe der Bildpyramide werden Extrempunkte gefunden (Minima und Maxima der DoG Bilder).
- Unpassende Extrempunkte (geringer Kontrast, Kanten) werden entfernt, die verbleibenden Extrema sind die gesuchten Keypoints.
- Für Rotationsinvarianz werden die Keypoint-Umgebungsorientierungen bestimmt.
- In der Folge werden mittels SIFT die Keypoint-Deskriptoren erstellt.

- Korrespondierende Keypoints werden gefunden und zu Merkmalspaaren zusammengefasst.
- Mittels RANSAC (Random Sample Consensus) wird die homographische Transformationsmatrix ermittelt, mit der die Bilder so überandergelegt werden, dass die korrespondierenden Keypoints übereinstir

Weiters wurde ein GUI implementiert.

## 1.6 Evaluierungsfragen

- Werden sinnvolle Merkmale gefunden?
- Werden übereinstimmende Bildpaare gefunden?
- Werden die Bilder korrekt zusammengefügt?
- Hat die Auflösung der Eingabebilder einen merklichen Einfluss auf das Ergebnis?
- Welche Bilddaten / Szenen sind ungünstig für den Erfolg des Verfahrens?
- Ist das GUI intuitiv aufgebaut?

## 2 Arbeitsteilung

(0,5 Seiten)
Wer hat welche Aufgaben übernommen (MATLAB-Funktionen, Abschnitt im Bericht, Evaluierung, Datenerfassung, etc.)?

Evaluierung, Datenerfassung, sowie Teile der Funktionalität und des Berichts wurden gemeinsam erarbeitet. Die folgende Tabelle zeigt die Haupt-Aufgabenbereiche, d.h. zu den genannten Themen wurden die entsprechenden Matlab-Funktionen und Abschnitte im Bericht erstellt.

| Name (alphabetisch) | Tätigkeiten |
|---|---|
| Hanna Huber | Keypoint-Matching (RANSAC)<br>Stitching der Bilder<br>Multi-Resolution-Spline für weiche Bildverläufe |
| Sebastian Kirchner | Entfernung von Keypoints die Kanten darstellen<br>Anzeige von Keypoints und Matching<br>Scale-Space-Erzeugung (DoGs) |
| Nikolaus Leopold | Bestimmung der Keypoint-Umgebungsorientierungen<br>Erstellung der SIFT-Deskriptoren |
| Ernad Sehic | Scale-Space-Erzeugung (DoGs) |
| Patrick Wahrmann | Bestimmung der Scale-Space-Extrempunkte<br>Entfernung von Keypoints mit geringem Kontrast<br>GUI |

# 3 Methodik

Ein wesentlicher Teil des Image Stitching ist die Korrespondenzanalyse. Dabei werden für den überlappenden Bereich der Bilder jene Bildpunkte in beiden Bildern gesucht, die dasselbe Objekt darstellen. Dieses Problem wird mithilfe von merkmalbasiertem Matching gelöst, das gegenüber regionenbasiertem Matching den Vorteil hat, dass homogene und daher irrelevante Bildbereiche nicht in die Berechnung mit einfließen. Der Aufwand wird dadurch erheblich reduziert. [5]
Die Merkmale - Interest Points oder Keypoints - werden mittels Scale Invariant Feature Transform (kurz: SIFT) [4] gefunden. Dieser Algorithmus hat den Vorteil, dass das Ergebnis unabhängig von der Skalierung oder Orientierung der Interest Points im jeweiligen Bild ist [5].

## 3.1 SIFT

Der SIFT-Algorithmus ist in mehrere Schritte unterteilt, in denen Skalierung, Position und Orientierung der Interest Points ermittelt wird. Abschließend muss jeder Punkt eindeutig beschrieben werden, um die Korrespondenzanalyse zu ermöglichen.

### 3.1.1 Bildpyramiden

Um die Skalierungsinvarianz zu garantieren, ist eine Multiskalenanalyse der Bilder notwendig. Im Zuge dessen wird eine Bildpyramide aufgebaut, bestehend aus 4 Oktaven mit jeweils 5 Frequenzstufen. Von Oktave zu Oktave wird die Auflösung des Bildes halbiert und innerhalb der Oktaven wird iterativ gefiltert.
Ausgehend von dieser Bildpyramide werden nun pro Oktave 4 DoG-Bilder erstellt indem sukzessive die Differenz der gefilterten Bilder berechnet wird.

### 3.1.2 Position der Interest Points

Interest Points stellen besonders markante, in einer lokalen Umgebung möglichst einzigartige Bildpunkte dar. Punkte, an denen sich im DoG-Bild ein lokales Minimum oder Maximum befindet, sind somit gute Kandidaten für Interest Points. Die Extrema werden in jeder Oktave der DoG-Bildpyramide gesucht. Für jedes Pixel werden dafür die umliegenden Nachbarn derselben, sowie der darüber und darunter liegenden Ebene betrachtet.

Um die Position des Extremums noch exakter - also im Subpixelbereich - zu bestimmen, wird danach noch eine Taylorapproximation durchgeführt. Extrempunkte die wenig Kontrast aufweisen oder eine Kante darstellen, werden an dieser Stelle wieder verworfen.

### 3.1.3 Umgebungsorientierung der Keypoints

Um Rotationsinvarianz der Keypoint-Deskriptoren zu gewährleisten, wird zusätzlich zur Keypoint-Position die Richtung maximaler Änderung, also der Gradient, der Umgebung um den Keypoint gespeichert. Die einzelnen Bestandteile des Deskriptors umfassen auch einige Orientierungen, welche nun relativ zur Grundorientierung der Umgebung definiert werden können.
Um die dominante Umgebungsorientierung zu beschreiben werden innerhalb eines Fensters um den Keypoint für alle Pixel die Gradienten (als Vektoren definiert durch Magnitude und Winkel) ihrer unmittelbaren 4er-Nachbarschaft bestimmt. Diese werden dann je nach ihrer Orientierung klassifiziert (hier in 36 Teile der vollen Umdrehung von 2pi radian) und die Magnituden aller Gradienten einer Klasse (bin) aufsummiert, es wird also ein Histogramm erstellt. Dabei werden die Beiträge der Magnituden nach Gauss'scher Gewichtung um den Keypoint verteilt, sodass weiter entfernte Gradienten weniger ins Gewicht fallen. Die nach Magnitude dominante Klasse definiert nun die Orientierung der Umgebung.[4]

### 3.1.4 Keypoint-Deskriptoren

Um die Korrespondenzanalyse zweier Keypoints unter hunderten weiteren und unter Umständen sehr ähnlichen Keypoints zu ermöglichen, muss jeder Keypoint möglichst eindeutig beschrieben sein. Die Deskriptoren durch welche die Keypoints beschrieben werden müssen aber auch eine gewisse Redundanz aufweisen, da sich korrespondierende Keypoints in anderen Bildern (hier in Aufnahmen derselben Szene mit anderen Einstellungen) in der Regel in Nuancen voneinander unterscheiden (Helligkeitsunterschiede, Verdeckungen, Verzerrungen, etc.). Neben einer gewissen Toleranz in der Korrespondenzanalyse (Matching) ist es daher wichtig, die Definition der für den Vergleich relevanten Grundstruktur in mehrere Teile aufzuspalten, die jeweils einen Beitrag zur Akzeptanzwahrscheinlichkeit des Keypoints liefern, sodass die Grundstruktur auch bei Abweichungen in einzelnen Teilen noch erkannt werden kann.
Bei SIFT werden daher die Deskriptoren durch Histogramme der Gradientenmagnituden nach 8 Orientierungsklassen (also ähnlich wie bei der

Bestimmung der Umgebungsorientierung) aus 16 4*4-Fenstern um den Keypoint herum definiert. Das heißt für 16 Fenster gibt es jeweils 8 Klassen in denen Magnituden aufsummiert werden, die Definition eines SIFT-Deskriptors umfasst somit 128 Elemente.[4]

## 3.2 Matching

Nun müssen Keypoints, die denselben Interest Point beschreiben, einander zugewiesen werden. Wie bei Lowe[4] wird dazu der Nearest-Neighbor-Ansatz gewählt.
Allerdings wird statt der euklidischen Distanz der Winkel zwischen den Deskriptor-Vektoren der entsprechenden Keypoints zwischen den Vektoren minimiert. Dieser lässt sich leicht aus dem Skalarprodukt der beiden Vektoren ermitteln. Für jeden Deskriptor des ersten Bildes können die entsprechenden Skalarprodukte für alle Deskriptoren des zweiten Bildes mittels einer einzigen Vektor-Matrix-Multiplikation berechnet werden.
Um die Eindeutigkeit der Zuweisung zu garantieren, wird der kleinste Winkel mit dem zweitkleinsten verglichen[4]. Nur wenn deren Verhältnis unter einem Schwellwert liegt, wird die Zuweisung akzeptiert.
Zusätzlich wird überprüft, ob der Deskriptor mit minimalem Winkelabstand bereits einem zuvor betrachteten Deskriptor zugewiesen wurde. In diesem Fall wird dieser durch den aktuellen Deskriptor ersetzt.

## 3.3 Homographische Transformation

Um die beiden Bilder zu einem Bild zusammenzuführen, muss eine Homographie-Matrix $H$ ermittelt werden, die die Koordinaten der Bildpunkte eines Bildes in entsprechende Koordinaten im Koordinatensystem des anderen Bildes umwandelt. Dies gilt insbesondere für Keypoint-Paare, also $X_2 = H \cdot X_1$ für ein Keypoint-Paar $(X_1, X_2)$. $H$ wird mithilfe des Random Sample Consensus-Algorithmus[2] berechnet.
Die Information über die Koordinaten der Keypoint-Paare wird verwendet, um ein lineares Gleichungssystem aufzustellen, dessen Lösung die Koeffizienten der Matrix $H$ liefert. Dafür werden vier Keypoint-Paare benötigt.[3]
Der RANSAC-Algorithmus wählt diese in jedem Iterationsschritt - deren Anzahl wird zuvor festgelegt - zufällig aus und berechnet die zugehörige Matrix $H$. Daraufhin wird die Homographie auf alle zu einem Keypoint-Paar $(X_1, X_2)$ gehörigen Punkte $X_1$ angewendet. Liegt der transformierte Punkt innerhalb eines Toleranzbereichs um den Punkt $X_2$, wird er als

*Inlier* bezeichnet. In jedem Iterationsschritt, also für jedes $H$, wird die Anzahl der Inlier berechnet. Am Ende wird die Matrix mit den meisten Inliers als Homographie-Matrix gewählt.
Da eine korrekte Lösung des oben beschriebenen Gleichungssystems stark von Ursprung und Skalierung des Koordinatensystems der Bilder abhängt, werden die Koordinaten der Keypoint-Paare zuvor normalisiert. Um die endgültige Homographie-Matrix zu erhalten, wird die Matrix $H$ am Ende noch mit den entsprechenden Transformationsmatrizen multipliziert.[2]

## 3.4 Image Stitching

Das Bildmosaik wird erstellt, indem das rechte Bild, $imB$, mithilfe der Homographie ins Koordinatensystem des linken Bildes, $imA$, transformiert wird. Um durch gerundete Daten entstehende Löcher zu vermeiden, wird dafür zunächst die Lage des transformierten Bildes im neuen Koordinatensystem ermittelt, indem die Ecken von $imB$ transformiert werden. Damit kann die Größe des Bildmosaiks berechnet werden.
Nun wird $imA$ entsprechend erweitert und eine Maske von derselben Größe erstellt, die die Region für das transformierte $imB$ enthält. Für diese Region werden anschließend mithilfe der inversen Homographie die entsprechenden Bildwerte berechnet.
Für das naive Splining wird schließlich das Bildmosaik mittels $I_{Mosaik} = (1 - I_{Maske}) \cdot I_{A,erweitert} + I_{Maske} \cdot I_{B,tranformiert+erweitert}$ erstellt.
Für das Multiresolution Splining[1] werden von beiden (erweiterten bzw transformierten) Bildern sowie der Maske Laplacepyramiden erstellt.Dadurch wird der Frequenzbereich in einzelne Bandbreiten von einer Oktave aufgeteilt. Nun wird auf jeder Ebene wie beim naiven Splining ein Bildmosaik zusammengesetzt. Am Ende werden die Bilder der Mosaikpyramie schließlich zu einem Mosaik zusammengefügt.

# 4 Implementierung

(1-X Seiten)
Hier gebt ihr einen Überblick über eure Implementierung:
Wie habt ihr die im vorhergehnden Abschnitt vorgestellte Methodik praktisch umgesetzt? Wie werden die einzelnen Methoden kombiniert (zB. Implementierungspipeline)?
Hier ist Platz für Implementierungsdetails wie zB. gewählte Parameter.
Wie startet der User das Programm? Welche Parameter hat der User zu setzen?
Auch in diesem Abschnitt können Referenzen und Zitate notwendig sein.

Die in Abschnitt 3 (Methodik) beschriebene Funktionalität ist in Matlab-Funktionen gegliedert und in main.m zur kompletten Image-Stitching-Pipeline zusammengefügt. Der Prozess wird durch main('Pfad zum linken Bild', 'Pfad zum rechten Bild') gestartet. Optional kann eine grafische Oberfläche mittels Aufruf von GUI.m verwendet werden. Alle notwendigen Schritte sind in dieser erklärt.
Um vor dem Endresult, also dem gestitchten Bild, die gefundenen Keypoints oder das Matching der Keypoints beider Bilder ausgeben zu lassen, können die Funktionen showKeypoints.m und showMatches.m. In main.m können dazu einfach die Zeilen mit 'showKeypoints' und 'showMatches' dekommentiert werden.

Für ein reibungsloses Zusammenspielen der einzelnen Komponenten sind in den Funktions-Headern die Schnittstellen definiert, also das Format der Parameter und Rückgabewerte beschrieben. Für genauere Information zu den einzelnen Funktionen sind daher diese heranzuziehen.

TODO

# 5 Evaluierung

(2-X Seiten)
Hier stellt ihr euren Datensatz vor und beantwortet Evaluierungsfragen: z.B. Fakten zum Datensatz: Anzahl der Bilder, Größe der Bilder, Quelle des Datensatzes (falls selbst aufgenommen: Aufnahmegerät, Einstellungen,... / falls nicht selbst erstellt: Datenbank vostellen...)
Diskussion der Evaluierungsfragen: Beantwortung der Fragen, Diskussion anhand von Beispielen, Diskussion von Grenzfällen: für welche Bilder funktioniert die Implementierung, für welche nicht? Worin unterscheiden sich diese Bilder? etc.

# 6 Schlusswort

(max. 1 Seite)
Hier fasst ihr Ergebnisse eures Projekt zusammen:
Welche Schlussfolgerung lässt sich ziehen? Gibt es offene Probleme? Wie lässt sich eure Lösung noch verbessern? etc.

## Literatur


[1] Peter J. Burt and Edward H. Adelson. A Multiresolution Spline with Application to Image Mosaics. *ACM Trans. Graph.*, 2(4):217–236, October 1983.

[2] Elan Dubrovsky. Homography Estimation. Master's thesis, The University of British Columbia, 2009.

[3] David Kriegman. Homography Estimation, 2007.

[4] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[5] Robert Sablatnig and Werner Purgathofer. Einführung in Visual Computing: Skriptum zur VU, 2014.