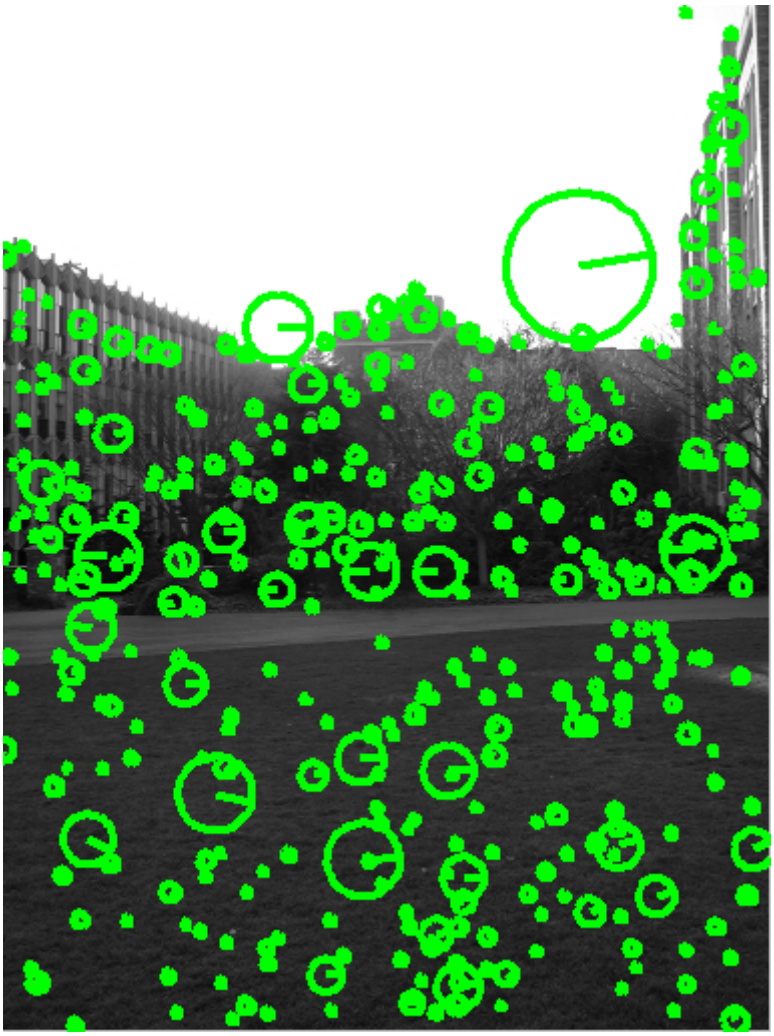
Gruppe 1:
Andreas Gogel, 0801243
Oana-Aurora Moraru, 1108261
Dominik Schörkhuber, 1027470

Computer Vision
Übung – Teil 2
Gruppe 1

Assignment 4: Image Stitching

A. SIFT Interest Point Detection

Im ersten Schritt werden aus den eingelesenen Bildern SIFT Features extrahiert. Dazu werden die Bilder zunächst in Singleprecision Grauwertbilder umgewandelt. Anschließend übernimmt der vl_sift() Befehl die Extraktion der Feature Vektoren. Bild1 zeigt beispielhaft die extrahierten Features des Bildes Campus1. Jedes Feature wird als Kreis dargestellt, wobei das Zentrum des Kreises die x,y **Position** des Features im Bild angibt. Bei der Suche nach Feature Punkten im Bild wird eine DoG (Difference of Gaussians) Pyramide des Bildes erstellt. Als Feature Kandidaten werden dann die Punkte ausgewählt welche in der Pyramide lokale Maxima darstellen. Der **Radius** des Kreises gibt die größe des Features an respektive das Level in der Pyramide auf dem es gefunden wurde. In einem gewissen Radius um das Feature wird für jeden Pixel die Orientierung des Gradienten bestimmt, gewichtet mit einem Gaußkernel werden die Orientierungen der Pixel in einem Histogramm aufgetragen. Die **Orientierung** des Features wird durch den Maximalwert im Histogramm bestimmt und ist im Plot durch eine Linie vom Zentrum zum Rand des Kreises zu erkennen.



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

Bild1 – Features in Campus1

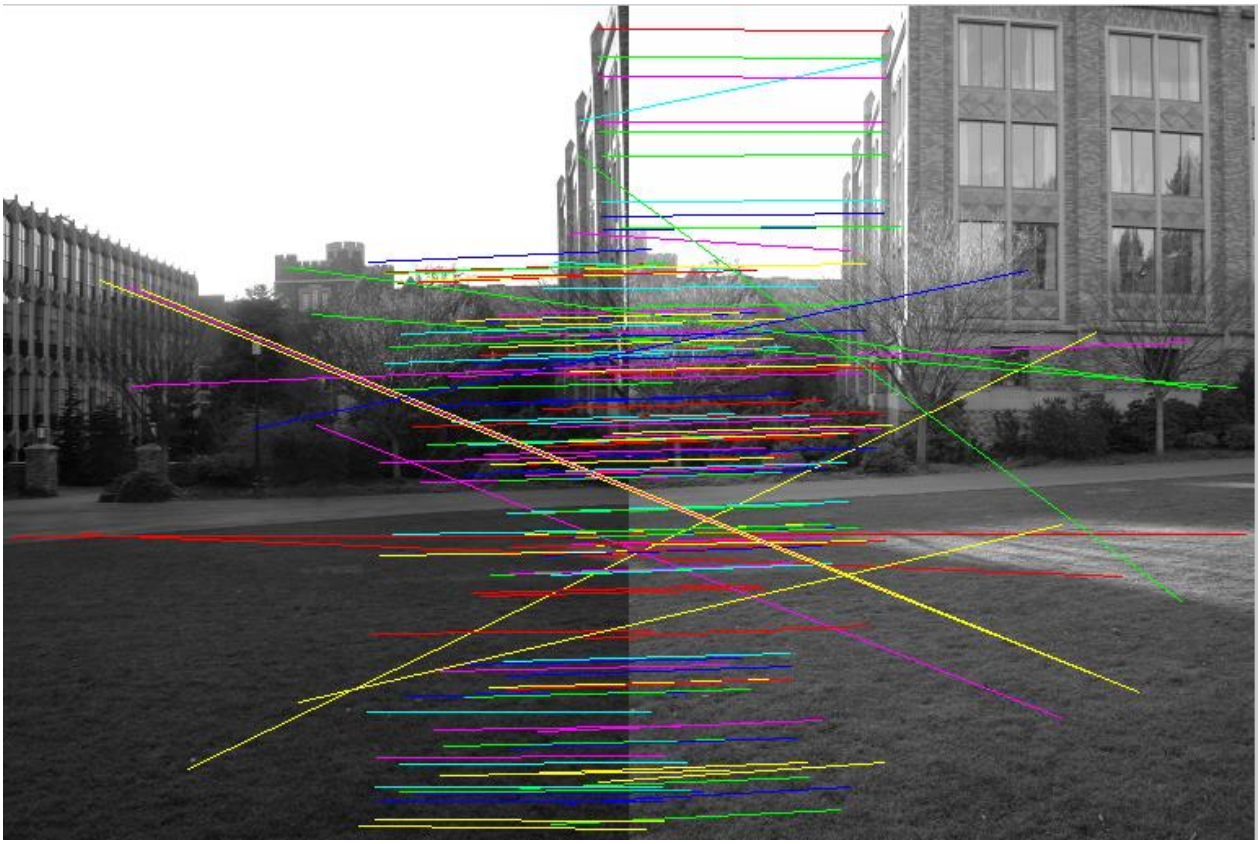


Bild2, Matching der Bilder Campus1 und Campus2

B. Interest Point Matching and Image Registration

Nachdem wir die Features jedes Bildes kennen können wir diese matchen. Also möglichst jedem Feature im linken Bild ein Feature im rechten Bild zuweisen. Bild2 zeigt ein solches Matching. Jede Linie verbindet zwei Features. vl_ubcmatch verwendet eine Nearest Neighbor Suche um einem Feature seinen Match zuzuweisen, der Match wird aber nur akzeptiert wenn die Distanz unter einem wählbarem Grenzwert liegt. Für unsere Implementierung haben wir den Default von 1.5 gewählt.

Als nächstes wollen wir eine Homeographie schätzen. Dazu wählen wir vier Matches, also vier Punkte im ersten und vier Punkte im zweiten Bild aus. Das zweite Bild wird nun so transformiert dass die gewählten Punkte in beiden Bildern übereinander liegen. Um zu schätzen wie gut die gewählte Homeographie wirklich ist werden die Distanzen zwischen jedem Featurepaar bestimmt und gezählt wieviele Featurepaare einen bestimmten Grenzwert unterschreiten. Die Homegraphie welche den höchsten Wert nach (N=1000) Iterationen errreicht wird als Ergebnis akzeptiert. Die Featurepaare welche unter dem Grenzwert liegen werden als Inliers bezeichnet. In Bild3 sehen wir die Inliers für unser Matchingbeispiel. Man erkennt dass nun alle falschen Featurekorrespondenzen ausgefiltert wurden. Mithilfe der Inliers wird nun die Berechnung der Homegraphie für ein exakteres Ergebnis noch einmal durchgeführt.

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

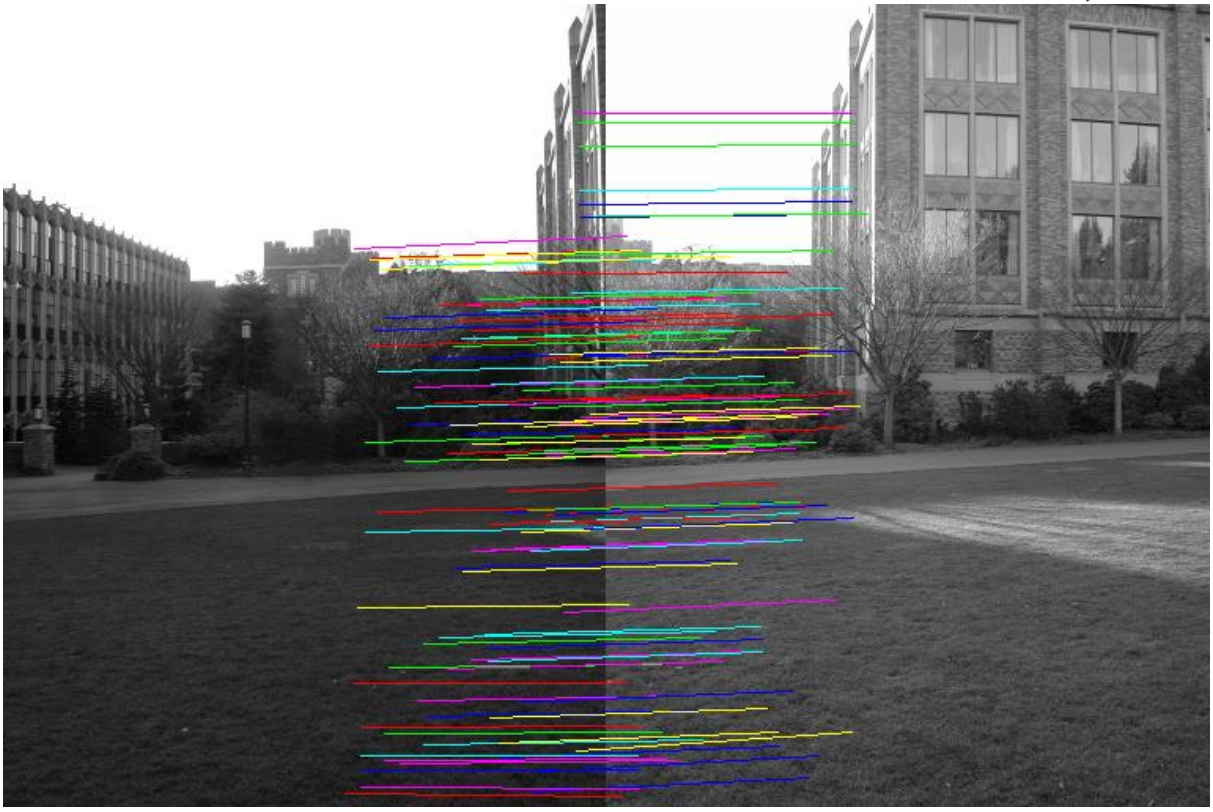
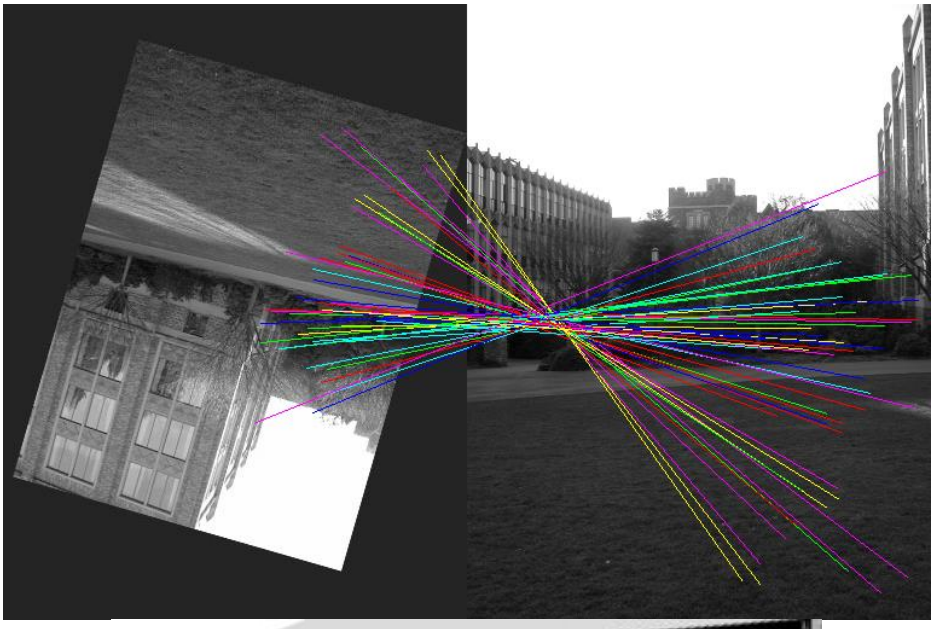


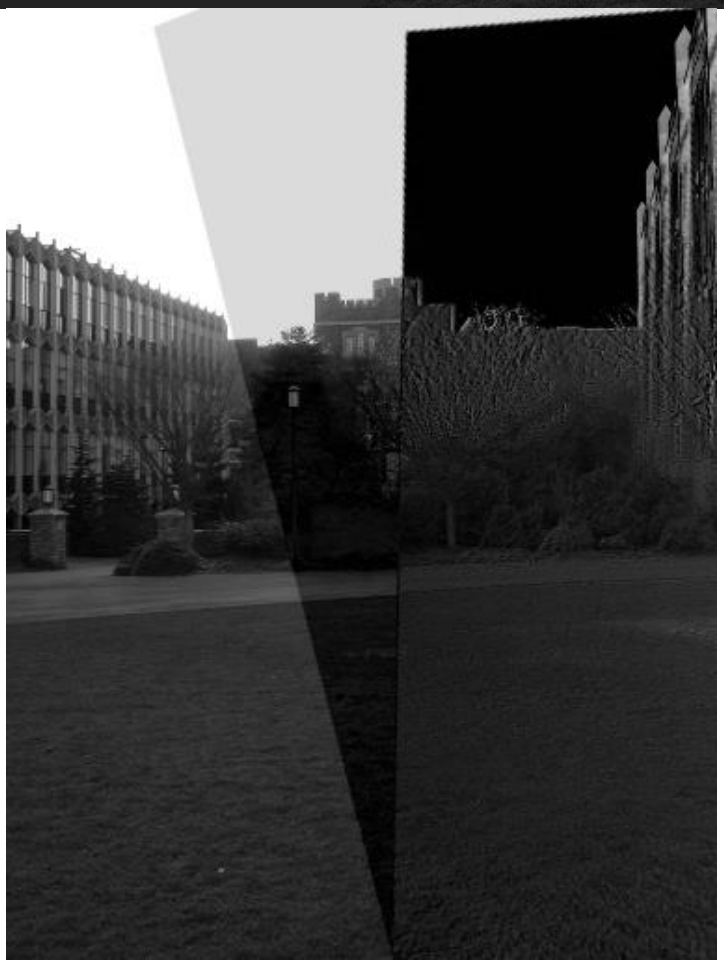
Bild3 Inlier Matches

Bild4 Erfolgreiches Stitching von zwei Bildern

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

Mit der neu berechneten Homeographie wird noch eine finale Transformation des ersten Bildes durchgeführt, und mit dem zweiten Bild überlagert. Das Ergebnis sehen sie auf Bild4. Die beiden letzten Bilder sollen nocheinmal die Rotations- und Skalierungsinvarianz der SIFT Features vor Augen führen. Das linke Bild wurde transformiert, trotzdem wird eine perfekte Homeographie gefunden.





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

C. Image Stitching

Um jetzt alle fünf Bilder zu einem Gemeinsamen zu verbinden müssen einige weitere Schrite getan werden. Zuerst werden die Homeographien zwischen je 2 aufeineinanderfolgende Bilder berechnet und in einer T-Form abegelegt (d.h.: in H12 ist die Homeographi zwischen den Bildern 1 und 2 gespeichert, H23 zwischen 2 und 3,...). Diese werden mithilfe der oben beschriebenen Funktion bestimmt.

Als nächstes wird ein Referenzbild gewählt. Da wir es nachfolgend nur mit Bilderreihen die aus fünf Bilder bestehen zu tun haben, wird dieses immer das mittlere (also das Dritte) sein. Nun werden die Homeographien von jeden Bild auf das Referenzbild bestimmt.

-H13 ergibt sich dann aus H23 * H12
-H23 wurde schon bestimmt
-H3 ist einfach die Identität
-H43 ist die Inverse von H34
-H53 ist die Inverse von H34 * der Inversen von H45

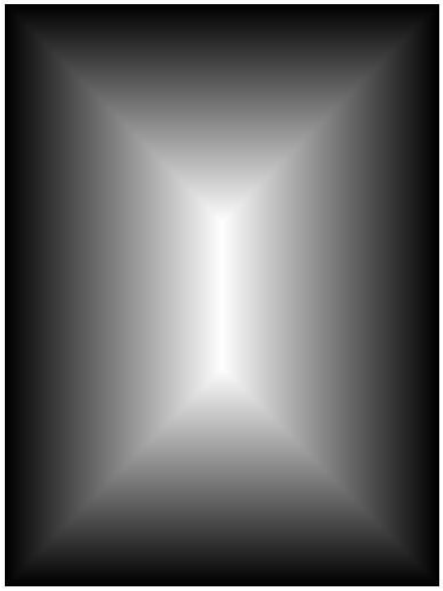
Nun werden die maximalen und minimalen x- bzw. y-Werte benötigt die nach der Transformation vorkommen können, um die Größe den Ausgabebildes bestimmen zu können. Hierfür werden für die Eckpunkte jedes Bildes ihre jeweilige Transformation angewandt und die Ergebnisse gespeichert. Mit den Maximum- bzw. Minimumoperator werden dann daraus die entsprechenden Informationen gewonnen.

Jetzt können die Bilder alle mit ihrer jeweiligen Transformation in das Ausgabebild transformiert werden. Um alle Pixel die nicht zum Bild gehören auf 0 zu setzen wird eine Filtermatrix erzeugt mit der die transformierten Bilder Elementweise multipliziert werden.

Um die überlappenden Bereiche zu behandeln wird das feathering benutzt. Hierfür wird wieder eine Maske benutzt. Diese ist an den Rändern schwarz und an den Stellen mit der größten Distanz zu den Rändern weiß eingefärbt. Die Werte dazwischen werden interpoliert. Hierfür wurde die Funktion bwdist() folgendermaßen verwendet:

-Die Größe der Maske wird auf de Größe des jeweiligen Bildes gesetzt
-zunächst wurde der Rand mit 1en gefüllt
-bwdist wird angewandt wodurch am Rand nun lauter 0en stehen. Leider werden die Zahlen nach innen immer größer und übersteigen die Zahl 1
-Um das zu korrigieren werden alle Werte durch den Maximalwert in der Maske geteilt.

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



Das Bild zeigt ein Beispiel für eine Maske in der Bilderreihe „Campus“. Diese werden nun auch in das Ausgabebild transformiert und mit den jeweiligen Bildern elementweise multipliziert. Für die spätere Normalisierung werden die Transformierten Masken auch noch zusammenaddiert. Die mit der Maske multiplizierten Bilder werden auch zusammenaddiert und das Ergebnis elementeweise durch die Summe der Masken als Normalisierung dividiert.

Das so erhaltene Ergebnisbild ist das gewünschte Ausgabepanorama.

Results discussion

Das “campus” Bild sieht auf den ersten Blick ganz gut aus. Es ist gut an den oberen und den unteren Rand erkennbar, wie die Transformationen angewandt wurden. Leider sind in diesem

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

Bild vor allem die Fenster fehleranfällig was sich durch Doppelungen bemerkbar macht. Aber auch an den Dachkonturen.

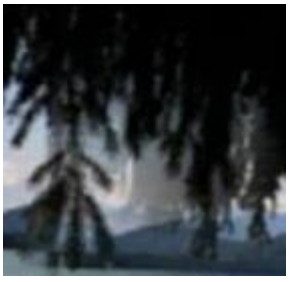
In der Bilderserie “officeview” sind diese Fehler deutlicher zu erkennen.

Im Allgemeinen scheinen sich diese Doppelungen vor allem in der Mitte des Bildes zu zeigen. Womöglich da das Bild 3 untrsansformiert verwendet wird wobei die Bilder rechts und links davon von einer Transformation verändert wurden. Ein weiterer Faktor ist bestimmt auch, dass an dieser Stelle sich mehrere Bilder überlappen. Im unten stehenden Bild werden wieder die gröbsten Fehler fokusiert.

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

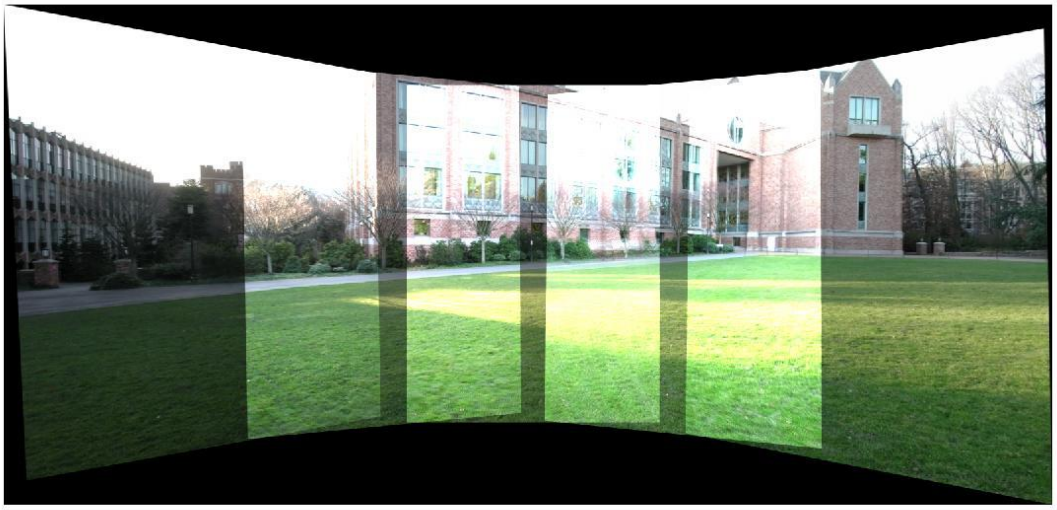
In der Dritten Fotoserie haben wir ein schon vorhandenes Panorama vom Wörthersee zerschnitten um zu testen wie dieses wieder mit der Methode zusammengefügt wird.

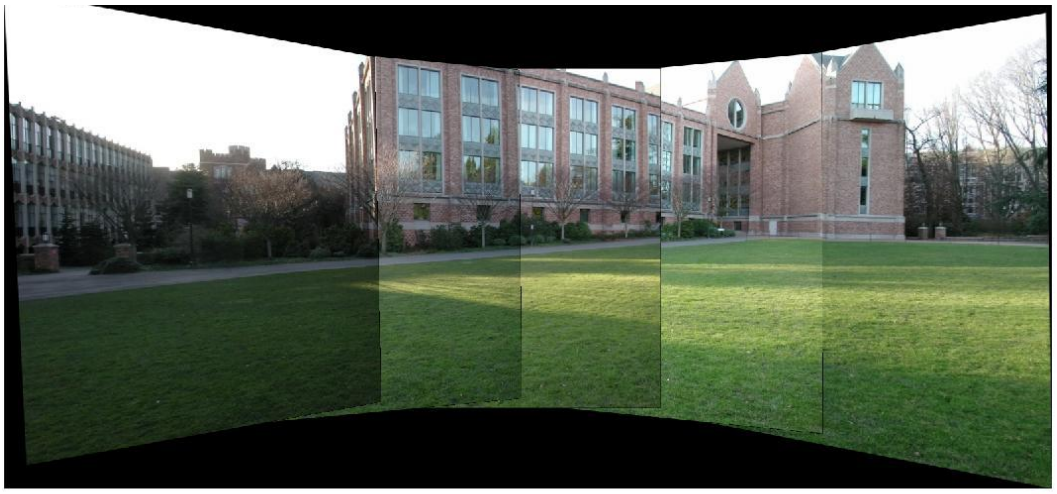
Schön zu sehen ist, dass die Transformationen sich stark in Grenzen halten da die Bilder schon alle dieselbe Fotographiehöhe und Winkel besitzen. Es ist daher beinahe komplett richtig wieder hergestellt. Beim genauen Hinsehen ist erkennbar, dass beim Baumgeäst oben ca. in der Mitte des Bildes ein dunkler Schatten zu sehen welcher nicht ganz richtig ist.



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

Im Weiteren sollte das Ergebnis mit einem Bild ohne Feathering verglichen werden. Hierfür wurden zum einen die Bilder einfach addiert und zum anderen wurden bei Überlappung nur die Farbwerte des jeweils linken Bildes verwendet.





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

Gut erkennbar ist bei der einfachen Addition, dass die überlappenden Teile natürlich vollkommen überbelichtet sind. Zustande kommt dies durch das Addieren der Farbwerte. Diese werden dadurch größer und kommen immer näher an die Farbe Weiß heran.
Im untersten Bild wiederum würden die Farbwerte passen. Durch die unterschiedlichen Lichtverhältnisse ist aber vor allem auf der Wiese ein Sprung der Farbwerte zwischen den Bildern zu erkennen. Passen die Bilder nicht zu 100% nach der Transformation zusammen ist bei dieser Technik zusätzlich noch ein plötzlicher Versatz im Bild zu erkennen.

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

Assignment 5:
Scene Recognition with Bag of Visual Words

Beschreibung Verfahren:

Unter Bag of Words versteht man ein Verfahren, das für Szenenwiedererkennung (Klassifizierung in bestimmte vordefinierte Kategorien von Bildern) verwendet werden kann.
Dazu extrahiert man zuerst SIFT Features aus allen Bildern des Training Sets. Diese Features werden dann in einer vordefinierten Anzahl an Clustern gruppiert. Die Cluster bilden dann die „Wörter“ des „Wörterbuchs“ (words of a dictionary). Dieses Wörterbuch kann dann benutzt werden um Bilder aus dem Test Set zu kategorisieren. Das geschieht indem jedes Bild als ein Histogramm dargestellt wird, welches die Frequenz von bestimmten Wörtern beschreiben. Je nachdem welche Wörter am häufigsten in einem bestimmten Bild erscheinen, kann das Bild zu einer bestimmten Kategorie zugeteilt werden.

Vorgehensweise:

Die Bag of Words Methode wurde implementiert anhand von 3 Matlab Funktionen: BuildVocabulary.m, BuildKNN.m und ClassifyImages.m. Das Script main.m kann eingesetzt werden um alle 3 Funktionen der Reihe nach aufzurufen.

readInFiles.m

Stellt eine Funktion dar, die alle Bilder aus einem Ordner und dessen Unterordner einliest und returniert. Die Funktion wird in allen Matlab Funktionen des Bag of Words verwendet.

Build Vocabulary.m

In dieser Funktion werden jeweils ungefähr 100 SIFT Features aus allen Bilder aus dem Training Set extrahiert. Dann werden alle Features in 50 Cluster gruppiert, welche die 50 Wörter der returnierten Wörterbuchs darstellen.

BuildKNN.m

Diese Funktion führt ein noch dichteres Sampling von SIFT Features durch und erstellt „Histogramme“ für jedes Bild, welche die Frequenz der Vorkommnisse der 50 Wörter in ihnen veranschaulicht. Alle 800 Histogramme werden in einer 800x50 Matrix gespeichert, wobei 800 (Zeilen) die einzelnen Bilder darstellen soll und 50 (Spalten) die Wörter aus dem Wörterbuch.
Die Funktion returniert auch noch einen Spaltenvektor mit 800 Einträgen, der die Klassenzugehörigkeit der 800 Bilder aus dem Training Set speichert.

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

ClassifyImages.m

In dieser Funktion werden wieder die SIFT Features der Test Bilder diesmal extrahiert, die 800x50 Matrix mit den Histrogrammen wird wieder aufgestellt und anhand der Resultate der BuildKNN.m Funktion klassifiziert.
Diese Funktion returniert eine sogenannte Confusion-Matrix, die veranschualicht wie oft ein Bild aus einer Kategorie i zu einer anderen Kategorie j zugeteilt wurde.

Results discussion

Der Bag of Words Ansatz liefert im Allgemeinen ungefähr 60% korrekte Klassifizierungen.
Das Resultat von unserer Bag of Words Implementierung wird in folgender Confusion-Matrix veranschaulicht:

```
conf_matrix =

    25     0    13    29     3    17    10     3
     0    94     0     0     4     0     0     2
     6     0    53    18     0    13     8     2
     7     0    14    46     1    14    13     5
     1     6     0     0    70     2    10    11
     5     0     7    13     0    75     0     0
     5     4     6    24     2     2    47    10
     2     0     3     5     2     1    20    67
```

Dabei beschreibt jedes Element (i,j) wie oft ein Bild aus einer Kategorie i zu einer anderen Kategorie j zugeteilt wurde. Ob ein Bild der richtigen Kategorie zugeordnet wurde, ist sichtbar in der Diagonale der Matrix, nämlich wenn das Element auf der Diagonale den größten Wert der gesamten Zeile repräsentiert, wurde das Bild richtig kategorisiert.
Aber da jede Kategorie (Zeile) 100 Elemente enthält, kann man die Einträge der Matrix direkt als Prozent auslesen.

Die Quote korrekter Klassifikation unserer Implementierung liegt bei 59,625%.

- Die „bedroom“ Klasse wurde generell am schlechtesten erkannt, bloß 25% richtige Treffer. Die „bedroom“-Bilder scheinen sogar häufiger als „living room“ eingeteilt zu werden.
- „Forest“-Bilder wurde mit 94%-iger Korrektheit der richtigen Klasse zugewiesen, schlussfolgernd lassen sie sich besten identifizieren.
- Jedoch „mountain“ und „office“ liefern auch nicht schlechte Ergebnisse, mit 70% und 75%.
- Die restlichen 4 Klassen haben ungefähr 50% erreicht, jedoch wurden sie den richtigen Klassen zugeordnet.

„Forest“ und „Mountain“ lassen sich nicht ein mal als „Kitchen“ oder „Living room“ einteilen.

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

Own test images

Die eigenen Testbilder, die wir gewählt haben gehören in die Klassen: “mountain”, „kitchen“ und „street“.

Fig.1 - Gebirge

Das erste Bild, von einem Gebirge, wurde korrekt der „mountain“ Kategorie zugeteilt.

Fig.2 - Küche

Das zweite Bild von der Küche aus dem Heim, wurde nicht als „kitchen“ erkannt, sondern der Klasse „office“ zugeteilt. Eines der Gründe dafür könnte sein, dass die Form der Mikrowelle und des Ofens mit der Form zweier Bildschirme verwechselt worden sind.

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

Fig.3 - Straße

Dieses Bild der Straße vor dem Heim wurde sogar in der Nacht geschoßen, aber trotzdem richtig der Kategorie “street” zugeteilt.