Dokumentation der Computer Vision Challenge

Gruppe: G10

Mitglieder:

Johann Brenner (ga78zan@mytum.de)

Michael Maas (ga96vur@mytum.de)

Daniel Milz (ga58huv@mytum.de)

Alexander Schmaus (ga96fin@mytum.de)

Die beiden folgenden Bilder werden in dieser Dokumentation zur besseren Veranschaulichung verwendet:



Bild 1

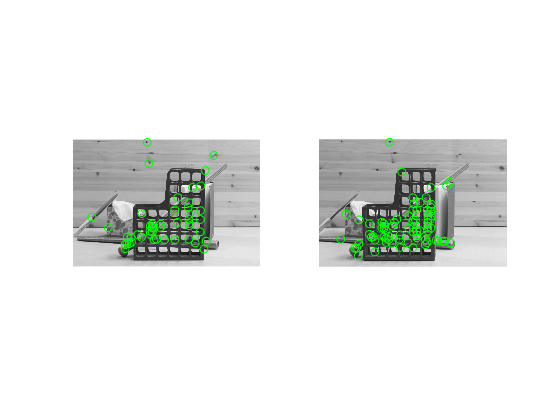


Bild 2

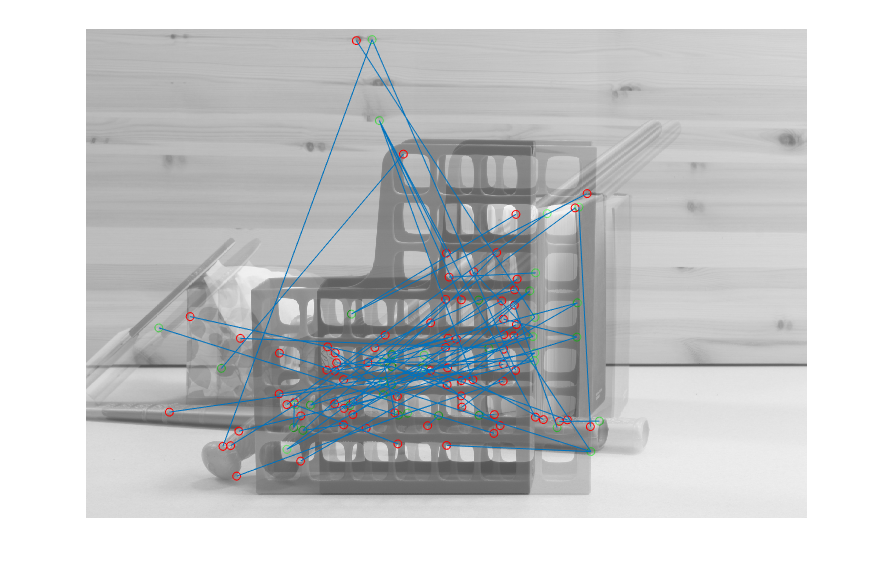
1. Disparity Map

Liegen die Bilder nach dem Einlesen in Matlab als RGB-Bilder vor, werden diese zu Grauwertbildern konvertiert.

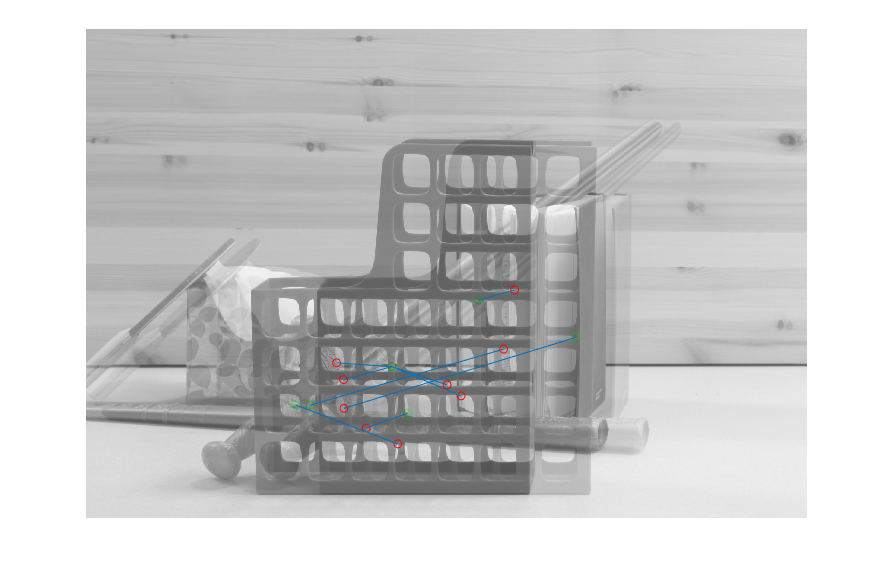
Im ersten Schritt werden die Merkmalspunkte (Ecken, Kanten und homogene Felder) der Bilder mit Hilfe des Harris-Merkmalsextraktors ermittelt. Die Idee des Algorithmus ist es durch Verschiebung des Bildes Änderungen in einzelnen Bildsegmenten zu erkennen. Dafür wird zuerst der Bildgradient eines Bildes ermittelt, daraus anschließend die Harris-Matrix G berechnet und aus dieser abschließend die Harris-Messung H. Anhand der Harris-Messung kann bestimmt werden, ob es sich bei einem Bildsegment um eine Ecke, Kante oder ein homogenes Feld (kein Merkmal) handelt. Für die Implementierung bedeutet dies, dass zuerst die Bildgradienten der beiden Bilder bestimmt werden. Dafür wird der Sobel-Filter verwendet, welcher eine Approximation des Bildgradienten des Bildes bestimmt. Mit Hilfe der Ableitungen wird die Matrix G berechnet. Aus dieser Matrix wird anschließend die Harris-Messung H mittels der Formel berechnet. Aus dieser werden die gewünschten Merkmale extrahiert. Da es sich dabei nur um ungefilterte Werte handeln würde, werden die Ergebnisse anschließend verbessert, indem ein maximaler Abstand zwischen zwei Merkmalen eingeführt wird. Außerdem wird das Bild in mehrere Bereiche aufgeteilt, in denen nur eine bestimmte Anzahl an Merkmalen zulässig ist. Der Algorithmus wird separat für beide Bilder ausgeführt. Danach stehen zwei Matrizen zur Verfügung, die die Koordinaten der Merkmale beinhalten.



Im zweiten Schritt werden Korrespondenzen zwischen den Merkmalen aus Bild 1 und Bild 2 ermittelt. Dafür wird *Normalized Cross Correlation* (NCC)angewandt. Das Verfahren beruht darauf, jedes Bildsegment, das ein Merkmal beinhaltet, zuerst zu normalisieren. Danach werden die Abstände zwischen den einzelnen Korrespondenzen (*Sum of Squared Differences* – SSD) und der NCC berechnet. Die daraus resultierenden Werte zwischen zwei Korrespondenzen liefern eine Aussage darüber, wie stark zwei Korrespondenzen korrelieren. Bei der konkreten Implementierung wurde folgendermaßen vorgegangen: Zuerst werden dem Bildrand zu nahe Merkmale aus den Bildern entfernt und die Bereiche um jeden Merkmalspunkt mit der Formel normalisiert. Mit Hilfe der normalisierten Werte wird anschließend der NCC mit der Formel ermittelt, und anhand dieses Wertes eine Entscheidung darüber getroffen, ob zwei Korrespondenzen korrelieren oder nicht. Korrespondenzen, die stark korrelieren, werden dann zu einem Korrespondenzpunktpaar zusammengefasst.

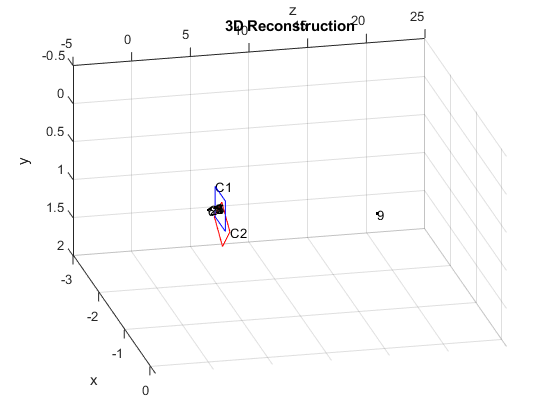


Um aus den Korrespondenzpunktpaaren robuste Korrespondenzpunktpaare zu ermitteln wird auf F-RanSaC Algorithmus zurückgegriffen. Dabei wird die Fundamentalmatrix der Korrespondenzpunktpaare ermittelt, und für jedes Paar die Samson-Distanz berechnet.

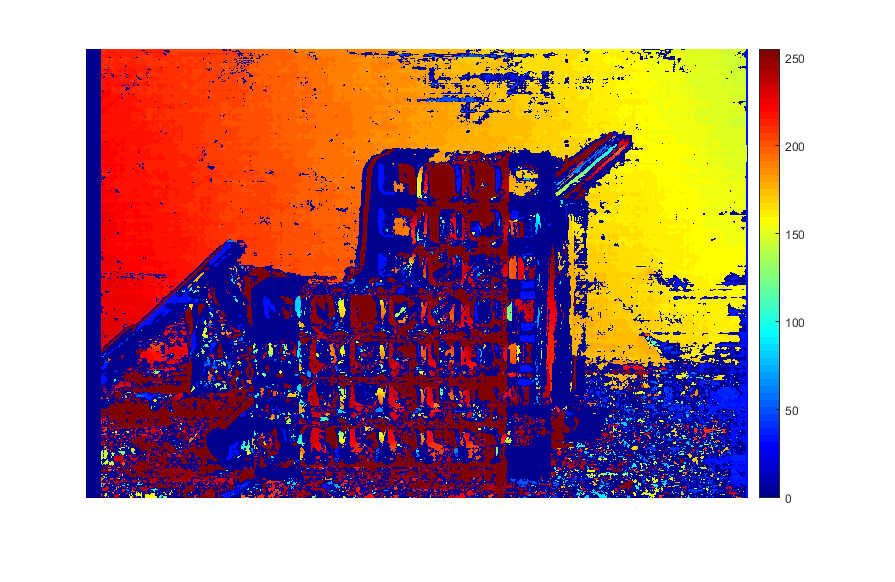


Aus den Korrespondenzpunktpaaren wird in Schritt vier unter Verwendung des Achtpunktealgorithmus die essentielle Matrix E bestimmt. Die essentielle Matrix stellt einen mathematischen Zusammenhang zwischen den Merkmalen von Bild 1 und Bild 2 her. Dafür werden die Koordinaten der Korrespondenzpunktpaare zuerst in homogene Koordinaten umgewandelt. Mit Hilfe des Kronecker-Produktes wird aus diesen Punkten die Matrix A bestimmt. A soll die Eigenschaft erfüllen. ist dabei die vektorisierte essenzielle Matrix. Da dies in der Praxis nicht zu lösen ist, wird anstatt dieser Gleichung das Minimierungsproblem gelöst. Dafür wird die Singulärwertszerlegung berechnet. Daraus ergibt sich als Lösung des Minimierungsproblem die neunte Spalte der Matrix V. Sortiert man diese um, erhält man die 3x3 Matrix G. Da G meist nicht die Bedingungen einer essentiellen Matrix erfüllt, wird auch von G die Singulärwertszerlegung gebildet und mit Hilfe der daraus resultierenden Matrizen U und V die essentielle Matrix erstellt. Bis auf den letzten Singulärwert (gleich 0) sind alle gleich 1. Die Implementierung dieses Schrittes folgt stringent diesem Verfahren.

Aus der essentiellen Matrix werden je zwei mögliche Rotations- und Transformationsmatrizen (R, T) berechnet. Dafür wird die Singulärwertszerlegung von E gebildet, und mit Hilfe einer Hilfsmatrix die Matrizen R1, R2, T1 und T2 berechnet. Für die 3D-Rekonstruktion eines Bildes sind immer genau ein R und ein T notwendig. Aus den je zwei Möglichkeiten ergeben sich damit vier verschiedene Kombinationen, von welchen nur eine geometrisch sinnvoll ist. Um diese Kombination zu ermitteln, wird aus allen Kombinationen die Matrix M berechnet und anschließend das Minimierungsproblem mit Hilfe der Singulärwertszerlegung von M gelöst. Die Kombination, die das Problem lösen könnte, wird anschließend als die Plausibelste angenommen. Damit kann anschließend das Bild rekonstruiert werden.



Abschließend kann dann die Disparity-Map berechnet werden.



2. PSNR

3. Unittest

Zur Absicherung der Funktionen wurde ein Matlab-Unittest implementiert. Dieser beinhaltet drei verschiedene Prüfverfahren:

* Überprüfen auf Toolboxen
* Überprüfen der Variablen in *challenge.m*
* Überprüfen des Ergebnisses von *verify\_dmap.m*

Direkt mit Ausführen des Unittests wird die Datei *challenge.m* aufgerufen. Dadurch stehen die für den zweiten Test benötigten Variablen zur Verfügung.

Im ersten Test wird dann mit Hilfe einer Matlab internen Funktion überprüft, ob in den Skripten/Funktionen *disparity\_map.m, challenge.m, verify\_dmap.m* unerlaubt Funktionen aus Toolboxen verwendet werden.

Im zweiten Test werden die Variablen, die bei der Ausführung von *challenge.m* erstellt werden, überprüft. Dabei darf keine Variable den Wert null haben, und keine Zelle unbefüllt sein.

Im letzten Test wird *verify\_dmap.m* und die Funktion *psnr* aus der Image Processing Toolbox ausgeführt. Die beiden Ergebnisse werden mit einander vergleichen und sollen sich nicht um einen höheren Wert als 0.1 unterscheiden.

4. Challenge (mit Ergebnissen für R,T,p für alle Bilder)

5. Plots

6. GUI

Die GUI wurde mit Hilfe des Matlab App Designer entwickelt und wird durch *start\_gui.m* aufgerufen. Dabei wird zunächst nur ein Button angezeigt, mit dem man den gewünschten Quellordner auswählen kann. Liegen in diesem Quellordner Bilder mit den Namen „Im0.png“ und „Im1.png“, so werden diese angezeigt und der Quellordner ist gültig. Ist der Quellordner gültig, so wird ebenfalls ein Button zugänglich, um die Disparity-Map zu erstellen. Wird dieser gedrückt, so werden im Hintergrund die Funktionen *[D, R, T] = disparity map(scene path)* und *p = verify dmap(D, G)* aufgerufen. Die Ergebnisse dieser Funktionen werden im Anschluss an die Ausführung in einem Plot für die Disparity-Map, Tabellen für R und T sowie einer Zeile für p ausgegeben.