

**Dokumentation Computer Vision Challenge**

Abgabedatum: 12.09.2018

Gruppe: G07

Gruppenmitglieder:

Moritz Eckhoff

Tobias Betz

Victor van Eetvelt

Fabian Uhl

Christian Geiger

Inhaltsverzeichnis

[Aufgabenstellung 3](#_Toc524009489)

[Programmablauf 4](#_Toc524009490)

[Rektifizierung 5](#_Toc524009491)

[Umsetzungsversuch 5](#_Toc524009492)

[Problemlösung 5](#_Toc524009493)

[Disparity Map 6](#_Toc524009494)

[Erstellen virtuelles Bild 8](#_Toc524009495)

[Nachbearbeitung des Bildes 9](#_Toc524009496)

[Zusatzpunkte 9](#_Toc524009497)

[Grafische Benutzeroberfläche 9](#_Toc524009498)

[Laufzeit 9](#_Toc524009499)

[Generierte Ansichten 10](#_Toc524009500)

# Aufgabenstellung

Aus einem Stereo-Bildpaar soll eine dritte virtuelle Ansicht generiert werden. Der Blickwinkel der virtuellen Ansicht soll zwischen den beiden realen Ansichten liegen und durch einen Prozentwert frei bestimmbar sein. Das Programm soll in Matlab ohne die Hilfe spezialisierter Toolboxen erstellt werden. Abbildung 1 zeigt die schematisch den geometrischen Zusammenhang der existierenden Kamerabilder und des virtuell erstellten Bildes.

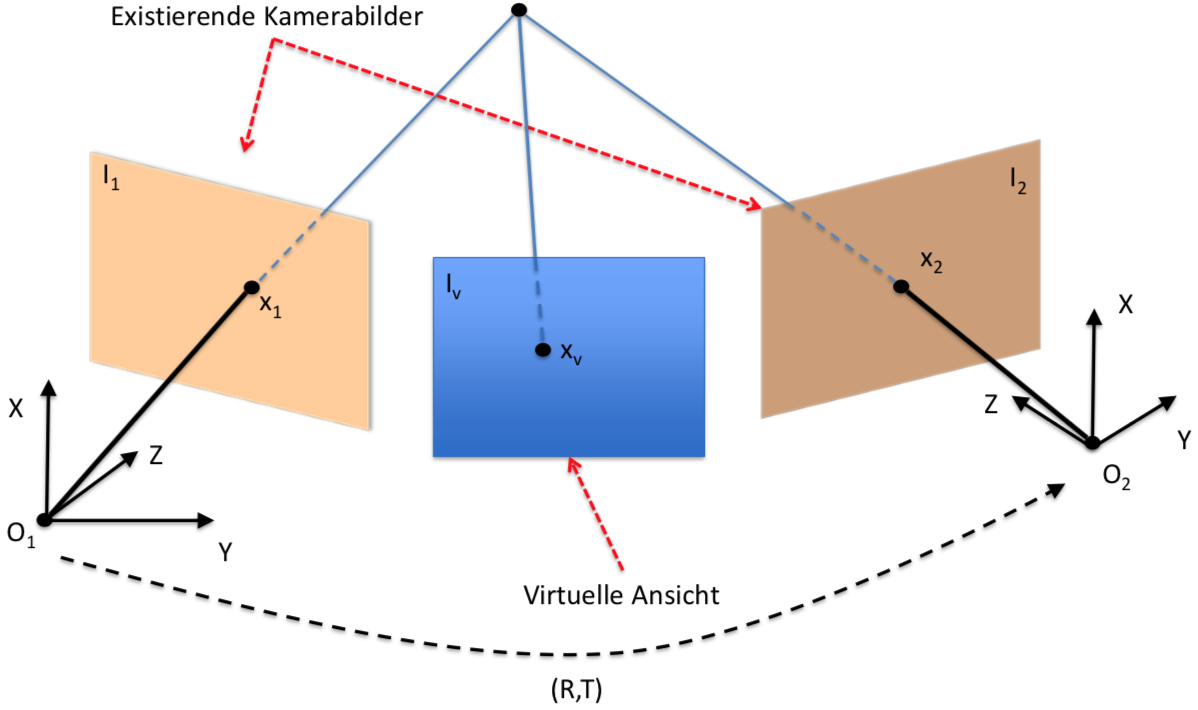


Abbildung 1 - Geometrischer Zusammenhang zwischen den realen Bildern und einer künstlich generierten Ansicht

# Programmablauf

In Abbildung 2 ist der Programmablauf zu sehen. Als erstes werden die beiden existierenden Kamerastereoaufnahmen geladen. Im nächsten Schritt müssen die beiden Bilder rektifiziert werden. Dabei werden die beiden Bilder so verändert, dass die Epipolarlinien parallel verlaufen. Dieser Schritt wird in einem folgenden Kapitel genauer erläutert. Als nächstes wird die sogenannte Disparity Map erstellt. In dieser wird dargestellt, wie sich jeder einzelne Pixel von einem zum anderen Bild verschiebt. Aus diesen Informationen kann anschließend das virtuelle Zwischenbild generiert werden. In einem nachfolgenden Schritt werden nicht berechnete Pixel korrigiert.



Abbildung 2 - Programmablauf Challenge

# Rektifizierung

Die Rektifizierung ist ein wichtiger Schritt bei der Erstellung eines virtuellen Bildes. Das Ziel der Rektifizierung ist, dass die Epipolarlinien der Bilder horizontal zueinander verlaufen. Korrespondenzpunkte von beiden gegebenen Bilder besitzen nach dem Bearbeitungsschritt die gleichen vertikalen Koordinaten. Dadurch kann die Disparity Map einfacher erstellt werden, da nur auf einer horizontalen Linie nach Korrespondenzpunkten gesucht werden muss. Außerdem bringt es Vorteile bei der Rekonstruktion des virtuellen Bildes.

## Umsetzungsversuch

In einem ersten Schritt wurde versucht die Rektifizierung selbst zu implementieren. Dabei ist ein erhebliches Problem aufgetreten. Die Rotationsmatrix und der Translationsvektor können nicht exakt bestimmt werden. Der Grund hierfür ist die ungenaue Schätzung der essentiellen Matrix. Aufgrund des Ransac-Algorithmus, der zufällige Korrespondenzpunkte verwendet ergibt sich eine bei jedem Durchlauf unterschiedliche essentielle Matrix. Die Rotation und Translation wird aus dieser Matrix bestimmt. In der Rektifizierung wird T dafür verwendet, um die X-Achse des neuen Koordinatensystems danach auszurichten und R wird benötigt, um die Rotation vom ersten Bild auf das zweite Bild zu projizieren. Aufgrund dieses Problems ist eine genaue Rektifizierung und die daraus folgende Erstellung des virtuellen Bildes nicht möglich. Der Quellcode ist als m-File unter dem Namen „Rectification.m“ zu finden.

## Problemlösung

Aufgrund des gerade erwähnten Problems wurde eine Vereinfachung getroffen und auf eine vorhandene Toolbox[[1]](#footnote-1) von A. Fusiello von der DPIA Udine zurückgegriffen. Dieser Code liefert ein sehr gutes rektifiziertes Bild, welches von uns verwendet wird. Dabei wird das rektifizierte Bild direkt in unser Programm geladen. In Abbildung 3 ist die Rektifizierung ersichtlich. Das obere Bildpaar zeigt die unrektifizierten Bilder und im unteren Bildpaar sind die rektifizierten Bilder zu sehen.

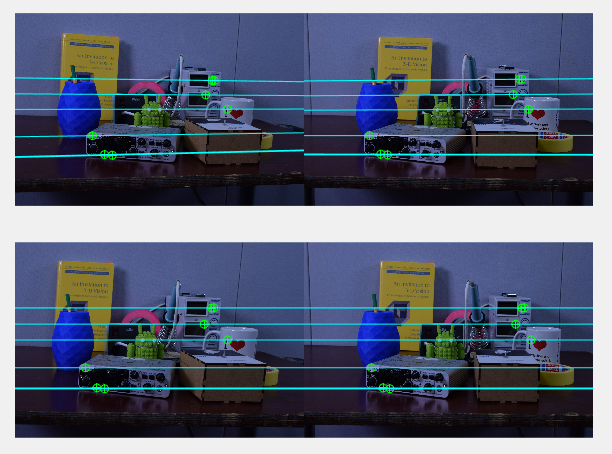


Abbildung 3 – Ergebnis der Rektifizierung

# Disparity Map

In einem ersten Versuch sollte eine Tiefenkarte mit Hilfe von Korrespondenzpunkten erstellt werden. Dabei wurden die in der Vorlesung und in den Hausaufgaben implementierten Funktionen verwendet und so angepasst, dass diese die für die gegebene Aufgabe geforderte Funktion erfüllen. Jedoch ist hier ein weiteres Problem aufgetreten. In den gegebenen Bildern existieren zu viele homogene Flächen, bei welchen die Suche nach Korrespondenzpunkten mit den bereits implementierten Funktionen, nicht möglich ist. In Abbildung 4 ist zu sehen, wie eine Tiefenkarte aussieht, die mit den gefundenen Korrespondenzpunkten erzeugt wurde. Es sind nur die Umrisse der Gegenstände auf dem Tisch und die Kante in der Wand zu erkennen.

Mit Hilfe des Block Matching wurde eine Verschiebungskarte (Disparity Map) erstellt. Dabei wird ein Bildausschnitt mit einer definierten Größe aus dem ersten Bild ausgeschnitten und dessen korrespondierender Bildausschnitt entlang der Epipolarlinie im zweiten Bild gesucht. Die Suche beschränkt sich dabei auf eine festgelegte, bildspezifische Maximalverschiebung (disparity range). Als Verschiebung wird die Entfernung in Pixeln zwischen den korrespondierenden Bildausschnitten in die Disparity Map eingetragen. Um die Blöcke einander zuordnen zu können wird die Sum of Squared differences benutzt. Die Anzahl und Qualität der Korrespondenzpunkte steigt, im Vergleich zu unserer Version mit den Korrespondenzpunktpaaren, da nur auf einem bestimmten Abschnitt der Epipolarlinie gesucht wird. Außerdem wird das Programm dadurch deutlich schneller. Es wurde ein vorhandenes Programm von Chris McCormick verwendet.[[2]](#footnote-2) In Abbildung 5 ist die Disparity Map zu sehen, die mit Hilfe des Block Matching Algorithmus erstellt worden ist. Wie vor allem am Hintergrund zu erkennen ist, gibt es noch einige Fehler, jedoch wird jedem Pixel ein Wert zugeordnet und größtenteils ist dieser Wert auch richtig. Beispielsweise sind die Tischkante, die Box, der Frequenzgenerator und die Android-Figur sehr gut zu erkennen.

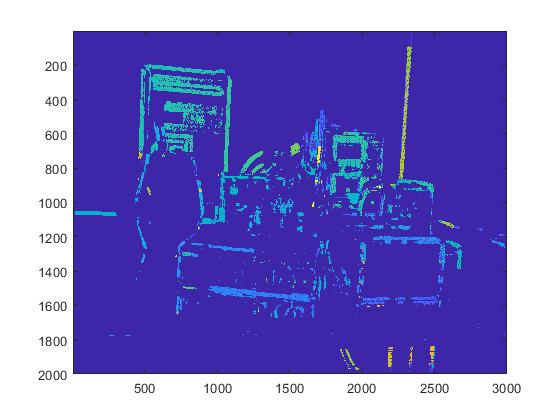


Abbildung 4 - Tiefenkarte aus Korrespondenzpunkten

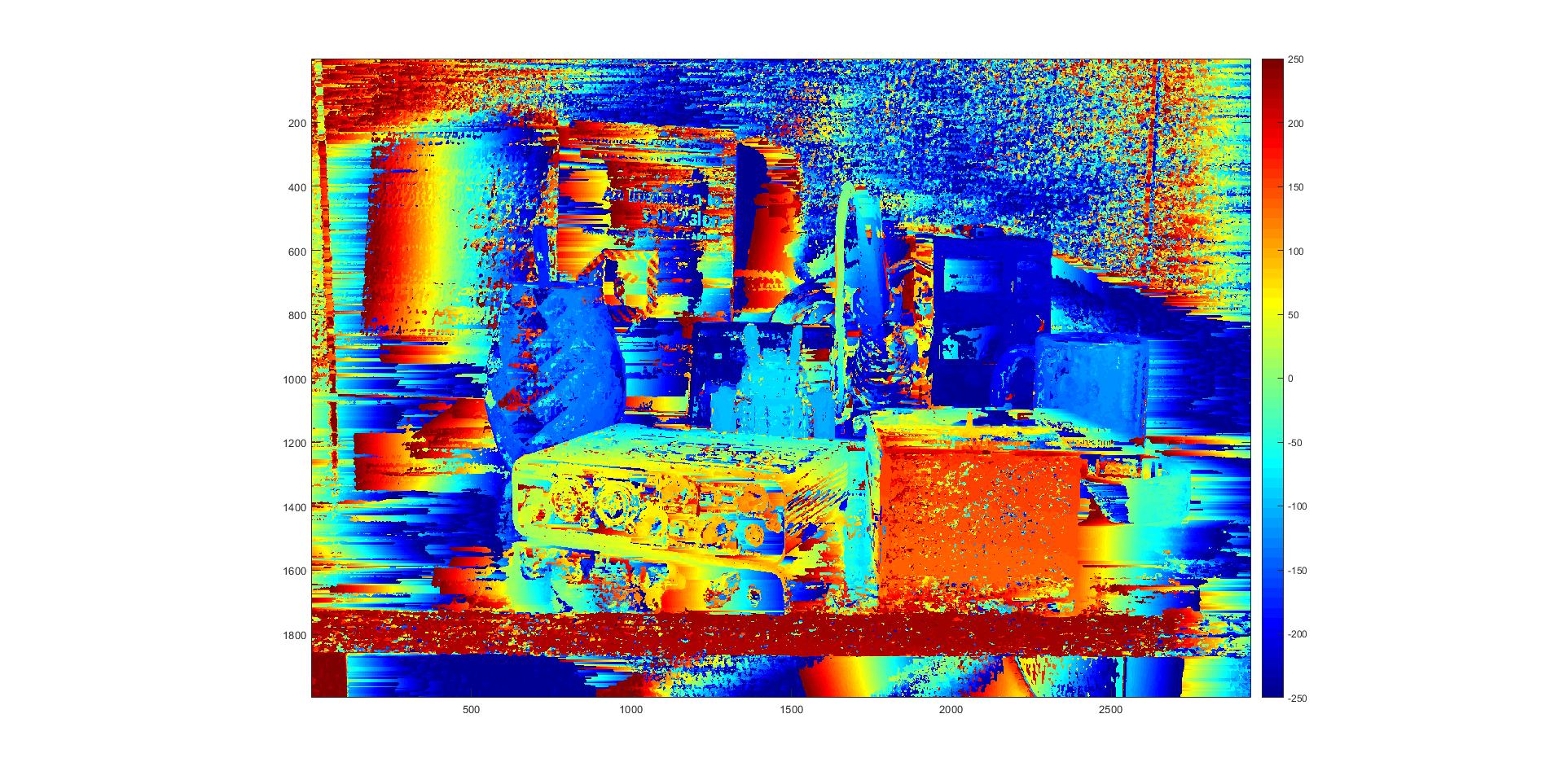


Abbildung 5 - Disparity Map aus Block Matching

# Erstellen virtuelles Bild

In der Funktion „Reconstruction 3D“ wird das virtuelle Bild erstellt. Dabei werden als erstes die Pixel nach der Entfernung zur Kamera sortiert, sodass weiter hinten liegende Gegenstände zuerst konstruiert werden. Das hat den Grund, dass wenn zwei oder mehr

Bildpunkte auf denselben Pixel projiziert werden, der Pixel des weiter vorne liegenden Gegenstandes dargestellt wird. Dadurch soll verhindert werden, dass Hintergrund, welcher im ursprünglichen Bild zu sehen ist, aber im Zwischenbild nicht zu sehen sein soll, den Gegenstand verdrängt welcher dargestellt werden soll. Mit Hilfe der Disparity Map und der vom Nutzer gewünschten relativen Verschiebung, wird die neue x-Koordinate berechnet. Die Formel hierfür sieht wie folgt aus:

Wenn diese innerhalb des Bildes liegt, wird der Farbwerte vom gegebenen Bild in das virtuelle Bild übertragen. Für alle noch schwarzen Pixel wird der gleiche Vorgang mit dem anderen gegebenen Bild wiederholt. In Abbildung 6 ist das virtuelle Bild zu sehen, welches aus den beiden gegebenen Bildern, der Disparity Map und der vom Nutzer gewünschten relativen Verschiebung erzeugt wird. Es sind noch einige schwarze Punkte zu erkennen. Das ist auf die Ungenauigkeit der Disparity Map zurückzuführen.



Abbildung 6 - Virtuelles Bild nicht nachbearbeitet (30 % Verschiebung)

# Nachbearbeitung des Bildes

In diesem Schritt werden die schwarzen Pixel eliminiert. Dafür wird jedes Pixel überprüft und die schwarzen Pixel detektiert. Es wird links und rechts von jedem Pixel das am nächsten liegende Pixel gesucht, welches einen Farbwert besitzt und dieser wird anschließend übernommen. In Abbildung 7 ist das virtuelle Bild zu sehen, nachdem die schwarzen Pixel aufgefüllt wurden.



Abbildung 7 - Virtuelles Bild nachbearbeitet ( 30% Verschiebung)

# Zusatzpunkte

## Grafische Benutzeroberfläche

* p muss in GUI verändert werden können
* Skalierung muss verändert werden können
* Orginalbilder laden
* Rektifizierten Bilder laden
* Half Block Size
* Disparity Range

Wenn möglich mit default werten hinterlegen

Ausgabe in Gui links Orginalbild, mitte generiertes Bild, rechts Orginalbild

## Laufzeit

In der Gui kann mit der Einstellung der Bildskalierung die Laufzeit beeinflusst werden.

# Generierte Ansichten



Abbildung 8 - Virtuelles Bild (20% Verschiebung)



Abbildung 9 - Virtuelles Bild (45%Verschiebung)



Abbildung 10 - Virtuelles Bild (70% Verschiebung)

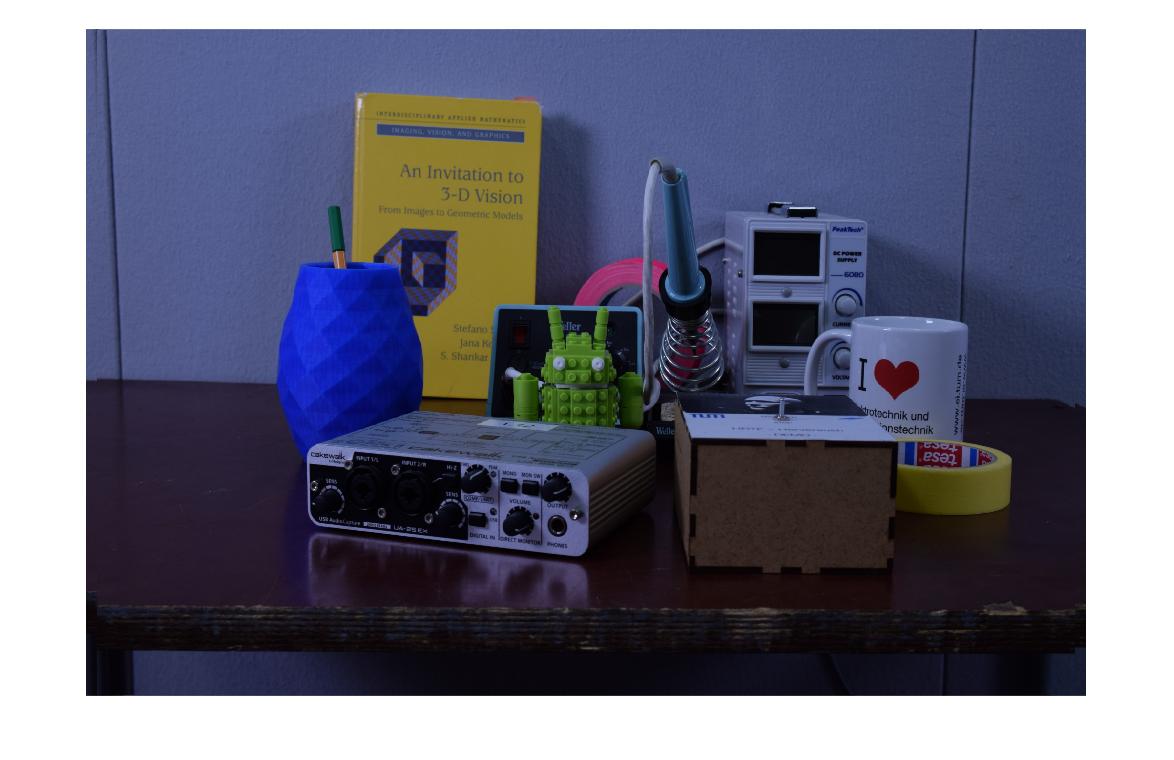


Abbildung 11 - Virtuelles Bild (100% Verschiebung)

1. Fusiello, Andrea,“Epipolar Rectification“, unter: <http://www.diegm.uniud.it/fusiello/demo/rect/> (aufgerufen am 06.09.2018) [↑](#footnote-ref-1)
2. McCormick, Chris,“Stereo Vision Tutorial – Part 1“, unter: http://mccormickml.com/2014/01/10/stereo-vision-tutorial-part-i/ (aufgerufen am 06.09.2018) [↑](#footnote-ref-2)