

## Projekt Ausarbeitung

---

### Kamerakalibrierung anhand eines Punktgitters

---

geschrieben von

Vera Brockmeyer (Matrikelnr. 11077082)

Artjom Schwabski (Matrikelnr. 11113320)

**Weiterführende Themen der Bildverarbeitung in SS 2017**

**Betreuer:**

Prof. Dr. Dietmar Kunz

Institute for Media- and Phototechnology

---

# Inhaltsverzeichnis

<b>1</b>	<b>Abstrakt</b>	<b>3</b>
<b>2</b>	<b>Einleitung</b>	<b>4</b>
2.1	Motivation . . . . .	4
<b>3</b>	<b>Stand der Wissenschaft</b>	<b>5</b>
3.1	Kamerakalibrierung mit Punktraster . . . . .	5
<b>4</b>	<b>Materialien</b>	<b>6</b>
4.1	Hardware . . . . .	6
4.2	Software . . . . .	6
4.3	<i>ImageJ</i> . . . . .	6
4.4	UnwrapJ . . . . .	7
4.5	<i>Apache Common Math</i> Bibliothek . . . . .	7
4.5.1	Klassen . . . . .	7
<b>5</b>	<b>Methode</b>	<b>9</b>
5.0.1	Levenberg-Marquard-Approximation . . . . .	9
5.0.2	Radial Distance Function (RDF) . . . . .	9
5.0.3	Kamerakalibrierung . . . . .	9
<b>6</b>	<b>Auswertung</b>	<b>10</b>
<b>7</b>	<b>Reflexion</b>	<b>11</b>
<b>8</b>	<b>Zusammenfassung</b>	<b>12</b>

---

# 1 Abstrakt

XXXX

---

## 2 Einleitung

Vera

### 2.1 Motivation

Vera

---

## 3 Stand der Wissenschaft

???

### 3.1 Kamerakalibrierung mit Punktraster

XXXX

## 4 Materialien

XXX

### 4.1 Hardware

During the implementation phase, the application was run on two computers which are described in the following two sections. Both computers needed to be able to deal with the software components described in section ?? . An extract from their data sheet is shown in Table 1 respectively Table 2.

Acer E5-571G	Description
Processor	Intel Core i7 CPU @ 2.40 GHz
RAM	8 GB
Graphic Card	NVIDIA GeForce 840M
Operating System	Windows 10 Education 64 bit

**Tabelle 1:** Extract from the Data Sheet of the Acer E5-571G

Acer Aspire 5820TG	Description
Processor	Intel Core i3 CPU @ 2.40 GHz
RAM	4 GB
Graphic Card 1	AMD Mobilty Radeon HD 5000 Series
Graphic Card 2	Intel(R) HD Graphics
Operating System	Windows 10 Education 64 bit

**Tabelle 2:** Extract from the Data Sheet of the Acer Aspire 5820TG Notebook.

### 4.2 Software

#### 4.3 *ImageJ*

*ImageJ* ist eine plattformunabhängige Open-Source Bildbearbeitungssoftware, die in Java implementiert wurde. Mit *ImageJ* können die gängigen Bildformate, wie TIFF, JPEG oder GIF mit unterschiedlichen Bittiefen angezeigt, analysiert und bearbeitet werden [1]. Es ist weiterführend ein weitverbreitetes Werkzeug zur Entwicklung von Methoden und Algorithmen, die zur Analyse von Bilddaten vorgesehen sind. Die Vielzahl der Funktionen wird stetig von den Nutzern in Form von Plugins erweitert und verbessert, wie es in Open-Source-Plattformen gängige Praxis ist.

## 4.4 UnwrapJ

Artjom

## 4.5 Apache Common Math Bibliothek

Vera

### 4.5.1 Klassen

Artjom

Im folgenden werden die Methoden der einzelnen Klassen erläutert. Die vollständige UML zur besseren Verständlichkeit der Klassenbeziehungen ist der Abb. 1 zu entnehmen.

**point\_grid\_radial\_affin\_distor\_** Hauptklasse der Anwendung. Implementiert das Interface *PluginFilter* um über ImageJ aufgerufen werden zu können.

Die Klasse besitzt folgende Methoden und deren Funktion:

run	Main-Methode des PlugIns in der die Optimierung aufgerufen wird
setup	Konstruktor-Methode des PlugIns in dem die Bildreferenz gespeichert wird
readData	Liest aus einer in ImageJ geöffneten Textdatei Punkt-Paare ein für Start- und Ziel-Koordinaten
computeDrawRadialTransformation	
drawTargets	Zeichnet Punkte an den übergebenen Ziel-Koordinaten in das übergebene Bild
computeDrawAffineTransformation	
computeRadius2Center	Berechnet anhand der Parameter den Abstand zum Gittermittelpunkt
compute_radial_dist_coef	Berechnet mit dem LevenbergMarquadt Optimierer die Koeffizienten der Radialen Verzerrung der übergebenen Punkt und gibt die Koeffizienten zurück

**Tabelle 3:** Methoden der point\_grid\_radial\_affin\_distor\_ Klasse

**SimplePair** Eine Einfache Klasse zum Speichern der Vorgabe- und Ziel Koordinaten und des Abstandes zum Mittelpunkt.

**RadialDistFunction** Klasse zum Erzeugen der Funktionen für den Optimierer.

RadialDistFunction	Konstruktor der Klasse. Es wird ein SimplePoint Array erwartet welcher Koordinaten-Paare für Start- und Ziel-Koordinaten enthält.
realTargetPoints	Gibt ein Array aus welches nur die Ziel-Koordinaten enthält. Dieses wird für den Optimierer benötigt.
retMVF	Funktion zur Modellierung der Radialen Verzerrung für den Optimierer. Berechnet zu den Vorgegeben Koeffizienten und einer Start-Koordinate die Ziel-Koordinate
retMMF	Jacobi-Matrix-Funktion zur Berechnung der Ableitung nach den einzelnen vom Optimierer vorgegebenen Koeffizienten

Tabelle 4: Methoden der RadialDistFunction Klasse

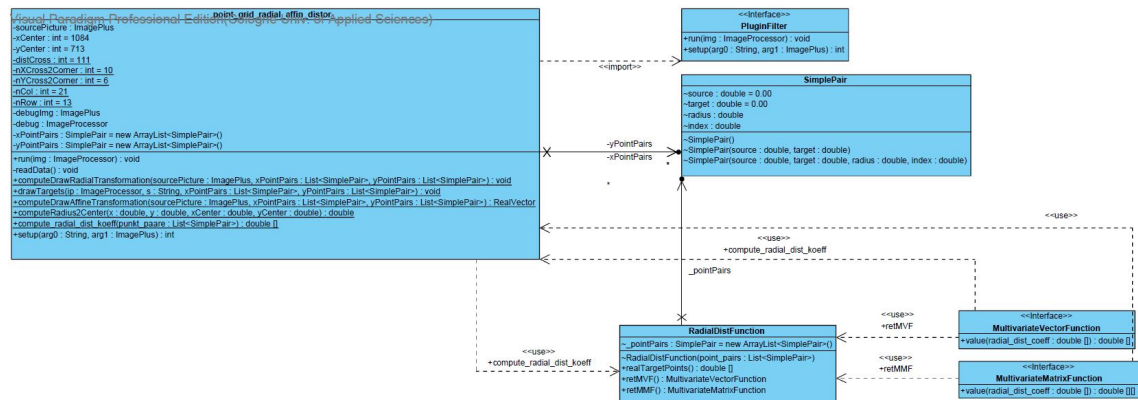


Abbildung 1: UML Klassendiagramm



## 5 Methode

### 5.0.1 Levenberg-Marquard-Approximation

Vera

### 5.0.2 Radial Distance Function (RDF)

Vera

### 5.0.3 Kamerakalibrierung

Artjom

---

## 6 Auswertung

XXXX

## 7 Reflexion

Vera

---

## 8 Zusammenfassung

Vera

## Literatur

- [1] Tony J Collins. Imagej for microscopy. *BioTechniques*, 43(1 Suppl):25—30, July 2007.