Szenenekonstruktion aus stereoskopischen Bildquellen gleicher und verschiedener Auflösungen

Erarbeitet von Studenten und Studentinnen
im Rahmen der Abschlussarbeit

Masterarbeit der Fakultät

Medieninformatik 4.Semester Anja Kretschmer 222222

Betreut von: Prof. Dr. Thomas Schneider

Disclaim here

Fakultät Digitale Medien der Hochschule Furtwangen

Wintersemeter 17/18 - Sommersemeter 18

Inhaltsverzeichnis

Over the last decades computer vision scientist have taken a new approach to vision. They build different computational models of what shoud be computed, what can really be computed, and how these computations can be realized by computer programs, and they use computers to test their models are correct. The result is a better understandung of vision from a different point of view, and at the same time some working artificial vision systems are built that can be used in idustry, medicine, etc. The knowledge obtained on neirophysiology and psychophysics have given hints to and influenced computer vision scientists, helping find solutions to the design of specific algorithms and implementation of vision systems. On the other Hand coputational vision has also given nerophysiologists and psychophysicsist a mathematical framework for modeling vision processes.[1]

1 Einleitung

Die Computer Vision ist ein Fachbereich der Computer Science mit dem Fokus auf die Entwicklung von künstlicher Intelligenz, die ein visuelles Verständnis ihrer Umgebung besitzen. Folglich wird in der Computer Vision der Weg von visuellen Eindrücken oder Bildern aus der Realität in den Rechner beschrieben [2]. Der Mensch ist mit der Fähigkeit ausgestattet, gesehene Bilder zu verarbeiten und kann die ihn umgebene Welt verstehen. Maschinen, die eine ähnliche Fähigkeit besitzen, wären somit ebenfalls in der Lage Entscheidungen auf Grund von visuellen Eindrücken zu fällen. Das entwickeln solcher Maschinen und den damit verbundenen Grundprinzipien und Programmen sind die Forschungsmittelpunkte von aktuellen Anwendungsbereichen wie dem Autonomen Fahren, Motion-Capturing, Bewegungserkennungen oder Service Robotern.

In dieser Masterarbeit wurde ein Algorithmus zur Rekonstruktion einer Szene aus stereoskopischen Bildquellen entwickelt. Das typische Verfahren einer Stereorekonstruktion basiert auf den Grundbausteinen Bildaufnahme und Bildanalyse[2]. In der Bildaufnahme wird eine Szene oder ein Objekt mit Hilfe von Kameras, Sensoren oder Lasern aufgenommen und als digitale zweidimensionale Bilder an den Computer weitergegeben. In der Bildanalyse werden die aufgenommenen Bilder ausgewertet um so die dreidimensionale Szene rekonstruieren zu können. Für die Analyse ist es essentiell die Kameraparameter, wie Position und Auflösung, zu kennen. Sind diese jedoch nicht bekannt, können die Bildquellen genutzt werden um die Kameraparameter abzuschätzen. Eine solche Abschätzung wird als Kamerakalibrierung[3, 4, 5, 1] bezeichnet. Die Position und Rotation einer Kamera im Raum werden als die extrinsischen Kameraparameter bezeichnet. Parameter wie die Auflösung oder Brennweiten, werden als die intrinsischen Kameraparameter bezeichnet[3, 4]. Im Zuge dieser Arbeit ist ein Algorithmus entstanden, welcher unter anderem im Stande ist die Kameras gleicher und unterschiedlicher Auflösung zu kalibrieren und eine 3D-Szenenrekonstruktion durchzuführen. Der vollständige Algorithmus wurde mithilfe eines virtuellen Beispiels verifiziert und auf eine reelle Szenenaufnahme angewandt. Beim Entwickeln von Algorithmen für Computer Vision Applikationen sieht man sich immer wieder mit komplizierten Aufgaben und Herausforderungen konfrontiert. Bei der Aufnahme von Bildern, kann es immer wieder zu unvorhersehbaren Bildfehlern wie beispielsweise Rauschen oder Verzerrungen durch die Kameralinse kommen, was nicht oft zum Verlust von Referenzdaten führt. Im Kapitel Reelle Rekonstruktion wird aufgeführt, wie mit solchen Fehlern umgegangen werden kann.

In der virtuellen Rekonstruktion wird zuerst eine 3D-Szene in zwei voneinander unterschiedlich positionierten, simulierten Kameras projiziert um virtuelle Bilddaten zu generieren. Anhand dieser 2D-Bilddaten wird die Kamerakalibrierung getestet. In der virtuellen Rekonstruktion, werden die Werte für Auflösung und Brennweite, selbst gesetzt. Im Test des Algorithmus mit reellen Bilddaten, wird für dessen Schätzung auf ein bereits existierendes Programm zurückgegriffen. Die intrinsischen Parameter werden mit dem hier entwickelten Algorithmus für die Schätzung der Positionen und Orientierungen der Kameras kombiniert um die Kameras anhand der virtuellen Daten zu kalibrieren. Die durch die Schätzung erhaltenen Kameraparameter können im virtuellen Beispiel so einfach mit den zuvor definierten Parametern verglichen werden, um den Algorithmus zu verifizieren. Diese Kameraparameter werden dann im entwickelten Rekonstruktionsalgorithmus dazu verwendet, die ursprüngliche 3D-Szene wieder herzustellen und die Funktionsweise der Rekonstruktion zu analysieren.

2 Model der Bildaufnahme mit einer Kamera

Um einen Szenenrekonstruktionalgorithmus zu verstehen werden in diesem Abschnitt grundlegende Bedingungen eingeführt um die Bildaufnahme mathematisch zu beschreiben. Ein abbildendes System besteht aus einem Objekt M, einer Kamera C und einer Bildebene I wie in Abbildung 2.1 dargestellt.

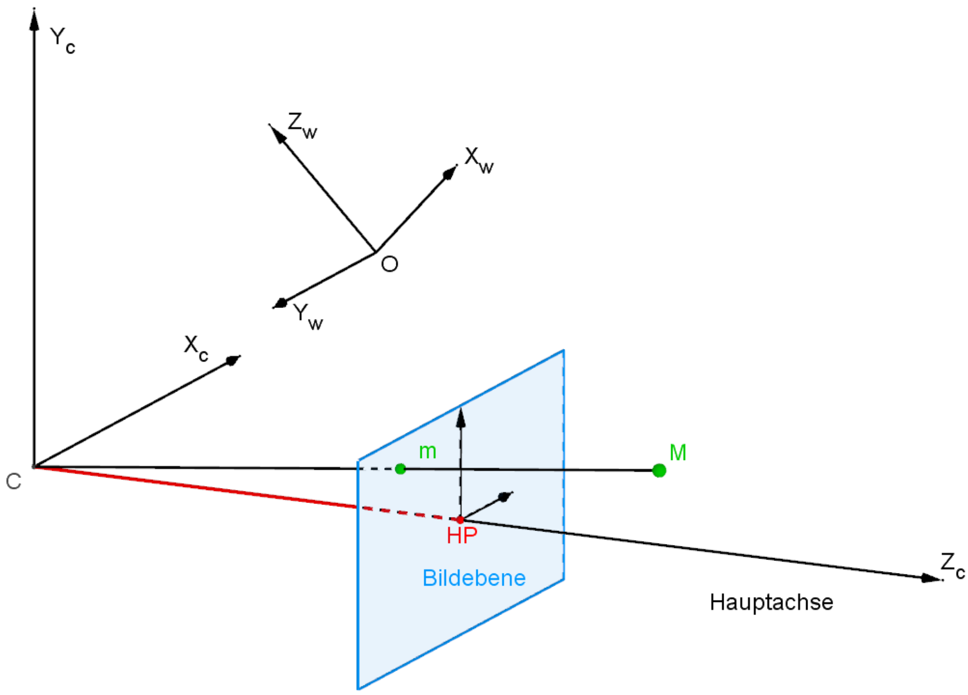


Abbildung 2.1: Schematik eines abbildenden Systems. Ein Punkt M im Weltkoordinatensystem O wird durch eine Kamera C aufgenommen. Diese Aufnahme wird durch eine Projektion, die als Verbindungslinie von M zu C zu sehen ist dargestellt ist und M auf m abbildet, beschrieben.

Ein Punkt M in einem dreidimensionalen Weltkoordinatensystem wird mit Hilfe einer Kamera, die in einem eigenen dreidimensionalen Kamerakoordinatensystem beschrieben wird, auf die Bildebene I projiziert. Die Bildebene I ist durch ein zweidimensionales Bildkoordinatensystem beschrieben. Der projizierte Punkt m kann mit einem Sensor aufgenommen und abgespeichert werden.

Im Folgenden wird zuerst ein Kameramodel eingeführt um die Projektion auf die Bildebene zu beschreiben. Daraufhin werden Koordinatentransformationen beschrieben um abschließend die Aufnahme eines Punktes mit einer willkürlichen Kameraorientierung zu berechnen.

2.1 Lochkameramodell zur Abbildung eines Punktes auf die Bildebene

Mit Hilfe des Lochkameramodells wird die Abbildung eines Objektes auf eine Bildebene beschrieben. Das Modell beruht ausschließlich auf der geometrischen Optik und vernachlässigt physikalische Effekte, wie Beugung oder die Auswirkung der Linse[6]. Das Lochkameramodell besteht aus einem Projektionszentrum C. C beschreibt gleichzeitig die Lage des Kamerazentrums und bildet den Ursprung des Kamerakoordinatensystems.[7, 3]. Die Blickrichtung der Kamera wird als Hauptachse bezeichnet. Die Bildebene steht senkrecht zur Hauptachse. Der Schnittpunkt der Hauptachse mit der Bildebene bildet den Hauptpunkt HP. Der Hauptpunkt ist der Ursprung des Bildebenenkoordinatensystems. Der Abstand vom Projektionszentrum zum Hauptpunkt wird als Brennweite ζ beschrieben[3, 7]. Der Bildpunkt m entsteht am Schnittpunk der Verbindungsgerade von C und M mit der Bildebene I.

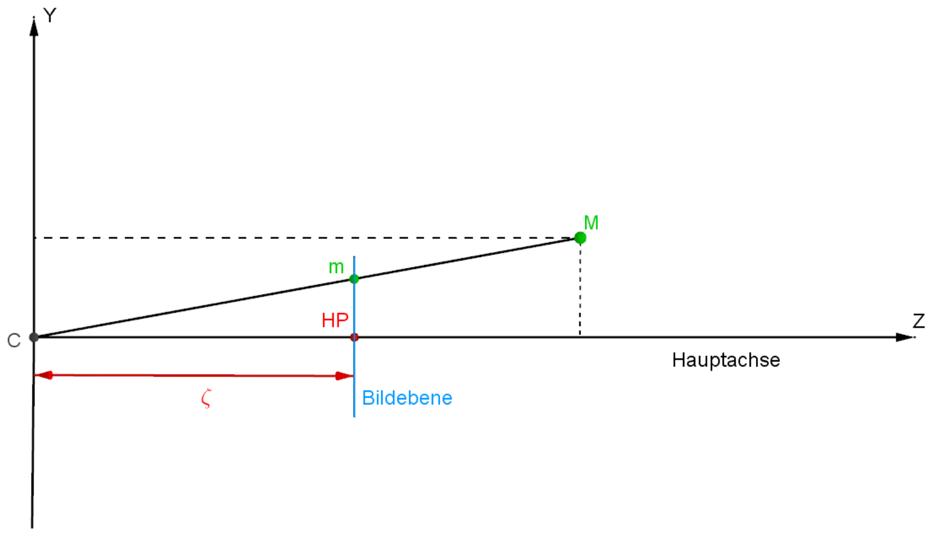


Abbildung 2.2: Die Abbildung zeigt einen Querschnitt des beschriebenen Lochkameramodells. Zu sehen ist das Projektionszentrum C der Kamera. C ist gleichzeitig das Kamerazentrum und bildet den Ursprung für das Kamerakoordinatensystem. ζ beschreibt den Abstand des Projektionszentrums zur Bildebene. Die Hauptachse beschreibt die Blickrichtung der Kamera. Der Punkt an dem die Hauptachse die Bildebene schneidet wird Hauptpunkt genannt und ist gleichzeitig der Ursprung für das Bildebenenkooridnatensystem. Der Bildpunkt m entsteht am Schnittpunk der Verbindungsgerade von C und M mit der der Bildebene I

Die Projektion eines dreidimensionalen Punktes auf eine zweidimensionale Bildebene, wird durch eine 3×3 Kameramatrix K_0 beschrieben.

$$K_0 \cdot M = \begin{bmatrix} \zeta & 0 & 0 \\ 0 & \zeta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} \zeta X \\ \zeta Y \\ Z \end{pmatrix} \mapsto \begin{pmatrix} \zeta \frac{X}{Z} \\ \zeta \frac{Y}{Z} \end{pmatrix} \tag{2.1}$$

Die Koordinaten auf der zweidimensionalen Bildebene werden häufig als homomogene Koordinaten angegeben. Dazu werden die Koordinaten mit Z normiert und somit auf die Ebene $(x, y, 1)^T$ projiziert wird. Zur Vereinfachung wird zuletzt nur die x, y Koordinaten des entstandenen Bildes angegeben. Gleichung 2.1 beschreibt somit die Abbildung eines Punktes auf die Bildebene.

2.2 Koordinatentransformation

Um einen Punkt von einem übergeordneten Weltkoordinatensystem in ein bestimmtes, zum Weltkoordinatensystem rotiertes, Kamerakoordinatensystem zu überführen ist eine Transformation notwendig. Im Folgenden wird der mathematische Weg einer Transformation eines Weltkoordinatensystems (O, δ) mit $\delta = (\hat{d_1}, \hat{d_2}, \hat{d_3}, O)$ in ein Kamerakoordinatensystem (C, β) mit $\beta = (\hat{b_1}, \hat{b_2}, \hat{b_3}, C)$ beschrieben.
Zunächst wird eine Koordinatisierung von Punkten im Weltkoordinatensystem vorgenommen. Ein Punkt P_δ bezüglich des Weltkoordinatensystems wird wie folgt beschrieben:

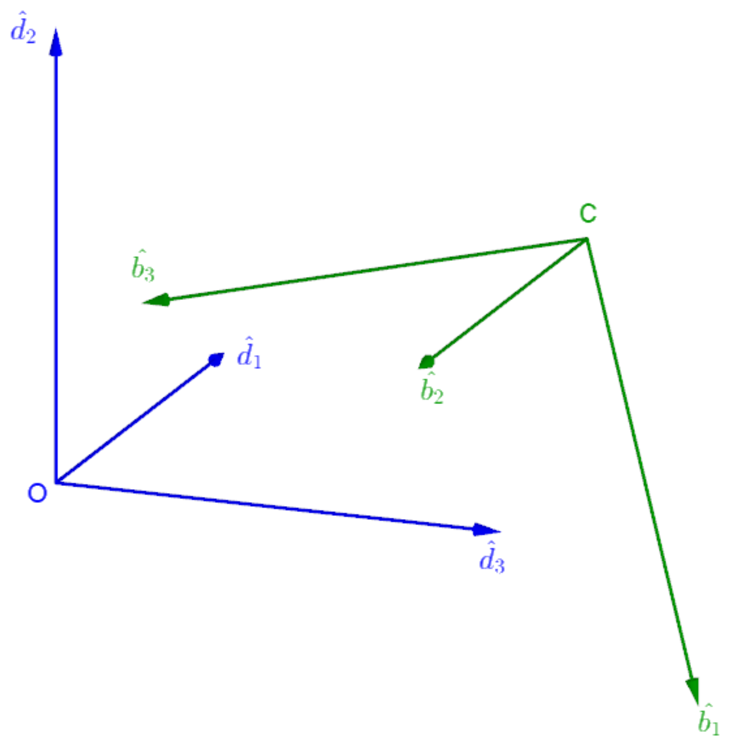


Abbildung 2.3: Ein Weltkoordinatensystem (O, δ) mit $\delta = (\hat{d}_1, \hat{d}_2, \hat{d}_3, O)$ wird zu einem dazu verschobenen und rotiertem Kamerakoordinatensystem (C, β) mit $\beta = (\hat{b}_1, \hat{b}_2, \hat{b}_3, C)$ transformiert.

$$P_\delta = O + p_{1\delta}\hat{d}_1 + p_{2\delta}\hat{d}_2 + p_{3\delta}\hat{d}_3 \tag{2.2}$$

$$\rightsquigarrow P_\delta = (p_{1\delta}, p_{2\delta}, p_{3\delta})^T = \begin{pmatrix} p_{1\delta} \\ p_{2\delta} \\ p_{3\delta} \end{pmatrix}. \tag{2.3}$$

Zwischen den beiden Koordinatensystemen (O, δ) und (C, β) gelten die folgenden Beziehungen:

$$C_\beta = O_\delta + C_{\beta,1}\hat{d}_1 + C_{\beta,2}\hat{d}_2 + C_{\beta,3}\hat{d}_3 \tag{2.4}$$

$$\hat{b}_1 = b_{11}\hat{d}_1 + b_{12}\hat{d}_2 + b_{13}\hat{d}_3 \tag{2.5}$$

$$\hat{b}_2 = b_{21}\hat{d}_1 + b_{22}\hat{d}_2 + b_{23}\hat{d}_3 \tag{2.6}$$

$$\hat{b}_3 = b_{31}\hat{d}_1 + b_{32}\hat{d}_2 + b_{33}\hat{d}_3. \tag{2.7}$$

Diese Beziehungsgleichungen werden in Gleichung 2.2 eingesetzt.

$$\begin{aligned} P_\delta = O &+ (C_{\beta,1} + p_{1\beta}b_{11} + p_{2\beta}b_{21} + p_{3\beta}b_{31}) \cdot \hat{d}_1 \\ &+ (C_{\beta,2} + p_{1\beta}b_{21} + p_{2\beta}b_{22} + p_{3\beta}b_{32}) \cdot \hat{d}_2 \\ &+ (C_{\beta,3} + p_{1\beta}b_{31} + p_{2\beta}b_{23} + p_{3\beta}b_{33}) \cdot \hat{d}_3 \end{aligned} \tag{2.8}$$

Aus Gleichung 2.8 wird ein Gleichungssystem in der Form von Gleichung 2.9 aufgestellt und gelöst.

$$\begin{aligned} p_{1\delta} &= C_{\beta,1} + (C_{\beta,1} + p_{1\beta}b_{11} + p_{2\beta}b_{21} + p_{3\beta}b_{31} \\ \rightsquigarrow\ p_{1\delta} - C_{\beta,1} &= (C_{\beta,1} + p_{1\beta}b_{11} + p_{2\beta}b_{21} + p_{3\beta}b_{31}) \end{aligned} \tag{2.9}$$

Das Gleichungssystem lässt sich in Matrixform darstellen als

$$\begin{bmatrix} b_{11} & b_{21} & b_{31} \\ b_{12} & b_{22} & b_{32} \\ b_{13} & b_{23} & b_{33} \end{bmatrix} \begin{pmatrix} p_{1\beta} \\ p_{2\beta} \\ p_{3\beta} \end{pmatrix} = \begin{pmatrix} p_{1\delta} - C_{\beta,1} \\ p_{2\delta} - C_{\beta,2} \\ p_{3\delta} - C_{\beta,3} \end{pmatrix} \tag{2.10}$$

.

Wenn P_β gegeben ist, erhält man auf diese Weise direkt P_δ. Die inverse Matrix D_β^{-1} kann verwendet werden um P_β aus P_δ zu berechnen.

$$D_\beta^{-1} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \tag{2.11}$$

$$\rightsquigarrow \begin{pmatrix} p_{1\beta} \\ p_{2\beta} \\ p_{3\beta} \end{pmatrix} = D_\beta^{-1} \begin{pmatrix} p_{1\delta} - C_{\beta,1} \\ p_{2\delta} - C_{\beta,2} \\ p_{3\delta} - C_{\beta,3} \end{pmatrix} \tag{2.12}$$

Handelt es sich um ein kartesisches Koordinatensystem, so gilt $D_\beta^{-1} = D_\beta^T$ und die transponierte Matrix kann für die Koordinatentransformation benutzt werden. Für zwei normierte, kartesische Koordinatensysteme ist D und D^T eine Rotationsmatrix R, weshalb im Folgenden, analog zur Literatur [3, 5, 4], $D^T = R$ angenommen wird. Um Gleichung 2.12 in einer kompakten Schreibweise zu formulieren, wird $\vec{p_\beta} = (p_{1\beta}, p_{2\beta}, p_{3\beta})^T$ zu einem vierdimensionalen Vektor mit 1 zu $\vec{p_{4\beta}} = (p_{1\beta}, p_{2\beta}, p_{3\beta}, 1)^T = (\vec{p_\beta}, 1)^T$ erweitert. Damit lässt sich Gleichung 2.12 als eine Matrixmultiplikation ausdrücken

$$\begin{pmatrix} p_{1\beta} \\ p_{2\beta} \\ p_{3\beta} \end{pmatrix} = D_\beta^T \begin{bmatrix} 1 & 0 & 0 & -C_{\beta,1} \\ 0 & 1 & 0 & -C_{\beta,2} \\ 0 & 0 & 1 & -C_{\beta,3} \end{bmatrix} \begin{pmatrix} p_{1\delta} \\ p_{2\delta} \\ p_{3\delta} \\ 1 \end{pmatrix} = R[I - C] \begin{pmatrix} p_{1\delta} \\ p_{2\delta} \\ p_{3\delta} \\ 1 \end{pmatrix} = T \begin{pmatrix} \vec{p_\delta} \\ 1 \end{pmatrix}. \tag{2.13}$$

Die Transformationsmatrix T setzt sich aus der Rotationsmatrix R und der Translationsmatrix $[I|-C]$ zusammen und wirkt auf den neu definierten vierdimensionalen Vektor. Wichtig dabei ist, dass $[I|-C]$ eine symbolische Schreibweise für eine 3×4 Matrix ist.

2.3 Aufname mit einer willkürlichen Kameraorientierung

Ein beliebiger Punkt im Weltkoordinatensystem kann mit der eingeführten Operation auf die Bildebene und schließlich auch auf den Sensor projiziert werden. Es werden insgesamt vier verschiedene Koordinatensysteme definiert. Das Weltkoordinatensystem (O, δ) mit $\delta = (\hat{d_1}, \hat{d_2}, \hat{d_3})$, das Kamerakoordinatensystem (C, β) mit $\beta = (\hat{b_1}, \hat{b_2}, \hat{b_3})$, das Bildebenenkoordinatensystem (I, τ) mit $\tau = (\hat{t_1}, \hat{t_2})$ und als letztes das Sensorkoordinatensystem mit (S, σ) mit $\sigma = (\hat{u}, \hat{v})$. Abbildung 2.4 zeigt die Koordinatensysteme schematisch im Überblick.

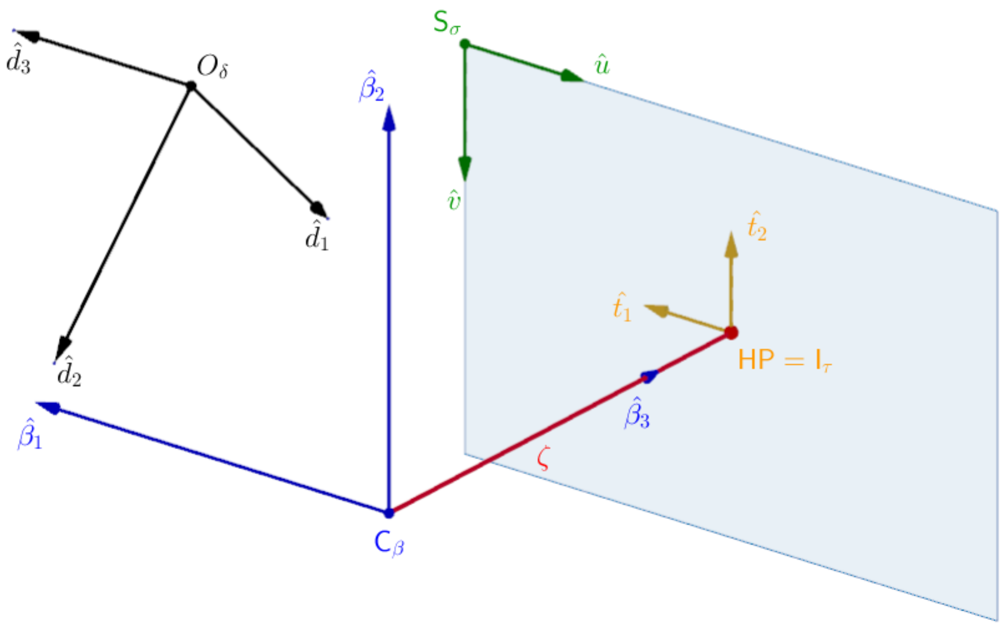


Abbildung 2.4: Das Schaubild zeigt die einzelnen Koordinatensysteme in einem Lochkameramodell. Das Weltkoordinatensystem (O, δ) mit $\delta = (\hat{d_1}, \hat{d_2}, \hat{d_3})$, das Kamerakoordinatensystem (C, β) mit $\beta = (\hat{b_1}, \hat{b_2}, \hat{b_3})$, das Bildebenenkoordinatensystem (I, τ) mit $\tau = (\hat{t_1}, \hat{t_2})$ und das Sensorkoordinatensystem (S, σ) mit $\sigma = (\hat{u}, \hat{v})$.

Für die Projektion eines Punktes $M_\delta = (M_{x\delta}\, M_{y\delta}\, M_{z\delta})^T$ bezüglich des Weltkoordinatensystems in einen Punkt $m_\tau = (m_{x\tau}\, m_{y\tau}\, m_{z\tau})^T$ bezüglich des Bildebenenkoordinatensystems kann eine Projektionsmatrix P definiert werden.

Zuerst muss der Punkt im Weltkoordinatensystem in das Kamerakoordinatensystem transformiert werden. Für die Transformation der Weltkoordinaten in Kamerakoordinaten gilt Gleichung 2.13:

$$\vec{M_\beta} = T \begin{bmatrix} \vec{M_\delta} \\ 1 \end{bmatrix} = R \begin{bmatrix} 1 & 0 & 0 & -C_{\beta,1} \\ 0 & 1 & 0 & -C_{\beta,2} \\ 0 & 0 & 1 & -C_{\beta,3} \end{bmatrix} \begin{pmatrix} M_{x\delta} \\ M_{y\delta} \\ M_{z\delta} \\ 1 \end{pmatrix} = R[I - C] \begin{bmatrix} \vec{M_\delta} \\ 1 \end{bmatrix} \tag{2.14}$$

Nach der Transformation eines Punkts $\vec{M_\delta}$ zu $\vec{M_\beta}$ in das Kamerakoordinatensystem erfolgt die Kameraprojektion von M_β auf m_β wie in Gleichung 2.1 beschrieben.

$$\begin{bmatrix} m_{x\beta} \\ m_{y\beta} \\ m_{z\beta} \end{bmatrix} = \begin{bmatrix} \zeta & 0 & 0 \\ 0 & \zeta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} M_{x\beta} \\ M_{y\beta} \\ M_{z\beta} \end{bmatrix} = K_0 \vec{M_\beta} \tag{2.15}$$

Die Projektion eines Punktes $\vec{M_\delta}$ auf den Bildpunkt $\vec{m_\beta}$ kann durch Gleichung 2.14 und 2.15 zusammengefasst werden.

$$\vec{m_\beta} = K_0 R[I - C] \begin{bmatrix} \vec{M_\delta} \\ 1 \end{bmatrix} \tag{2.16}$$

Um den Bildpunkt $\vec{m_\beta}$ bezüglich eines zweidimensionalen Bildebenenkoordinatensystemes anzugeben, wird die Bildebene mit der Tiefkomponente $m_{z\beta}$ normiert, sodass $m_{z\beta}$ auf den zweidimensionalen Raum der Bildebene gemappt wird. Diese Projektion wird durch folgende Gleichung beschrieben

$$\vec{m_\tau} = \begin{bmatrix} m_{x\beta}/m_{z\beta} \\ m_{y\beta}/m_{z\beta} \\ 1 \end{bmatrix} \tag{2.17}$$

.

Zuletzt folgt die Transformation der Bildebenenkoordinaten auf den Sensorchip. Der Sensorchip besteht aus einer Ansammlung von Sensorelementen. Diese Sensorelemente können verschiedene Formen annehmen. Die meisten Sensorchips bestehen aus rechtwinkligen, rechteckigen Sensorelementen. Aus diesem Grund wird ein rechtwinkliges Sensorelement mit einer Größe $lx\ ly$ angenommen. Diese Sensorelementgröße $lx\ ly$ definiert auch die Pixelgröße und bildet die Längenskalierung des Sensorkoordinatensystems. Neben der unterschiedlichen Skalierung wird der Ursprung des Sensorkoordinatensystems in der Regel an einer Ecke des Sensorchips definiert, sodass die Transformation von Bildebenenkoordinaten in Sensorkoordinaten auch eine Translation $(V_{x\sigma}, V_{y\sigma})$ aufweist[3, 8]. Für einen Punkt $m_\sigma = (u, v, 1)$ auf dem Sensorkoordinatensystem lassen sich die folgenden Bedingungen herleiten

$$u = m_{\tau x} k_x - V_{x\sigma} \tag{2.18}$$
$$v = m_{\tau y} k_y - V_{y\sigma} \tag{2.19}$$
$$1 = 1 \tag{2.20}$$

.

$k_x = 1/lx$ und $k_y = 1/ly$ ist die Pixeldichte in $\frac{pixel}{m}$. Es wird angemerkt, dass in der Bildebene und der Sensorebene die Punkte ausschließlich im zweidimensionalen Raum definiert sind. Aus den normierten Bildkoordinaten lässt sich somit folgende Sensormatrix bilden:

$$\vec{m_\sigma} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} k_x & 0 & V_{\sigma,x} \\ 0 & k_y & V_{\sigma,y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_{x\tau} \\ m_{y\tau} \\ 1 \end{bmatrix} = R_\sigma \vec{m_\tau} \tag{2.21}$$

Mit Hilfe der Transformationsmatrix R_σ kann ein Punkt von der Bildebene auf das Sensorelement projiziert werden.

Der hier skizzierte Lösungsweg beschreibt die Bildaufnahme eines Punktes im Lochkameramodel. Die hier eingeführte Projektionsmatrix $P = K_0 R[I - C]$ gilt für die Abbildung eines Punktes auf den Bildpunkt. Der Bildpunkt wird normiert und in das Sensorkoordinatensystem umgerechnet. In der Literatur wird häufig die Transformation in das Sensorkoordinatensystem bereits in der Kameramatrix zusammengefasst. Damit bildet sich die erweiterte Kameramatrix K

$$K = R_\sigma K_0 = \begin{bmatrix} k_x\zeta & 0 & V_{\sigma,x} \\ 0 & k_y\zeta & V_{\sigma,y} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.22}$$

Mit dieser Kameramatrix wird eine neue Projektionsmatrix mit $P = KR[I - C]$ gebildet, die einen Objektpunkt auf einen Bildpunkt im Sensorkoordinatensystem abbildet. Durch die Normierung dieses Punktes kann auf den direkten Sensorpunkt geschlossen werden. Ein weiterer Vorteil der erweiterten Kameramatrix K ist, dass sie die Pixeldichte k_x, k_y und die Brennweite ζ beinhaltet. Diese Parameter werden im Folgenden als intrinsische Kameraparameter bezeichnet. Die Koordinatentransformationsmatrix R wird im Gegensatz aus den sogenannten extrinsischen Kameraparameter, der Kameraposition und Orientierung, definiert. Sind sowohl die intrinsischen wie auch extrinsischen Kameraparameter vorbestimmt kann somit P bestimmt werden und mit dem hier beschriebenen Lösungsweg das Bild konstruiert werden.

3 Geometrische Beziehungen zwischen Punktekorrespondenzen

Das Ziel dieser Masterarbeit ist eine 3D Szene im dreidimensionalen Raum aus einer stereoskopischen Aufnahme zweier Kameras zu rekonstruieren. Bisher wurde die Bildaufnahme einer einzigen Kamera betrachtet. Jedoch kann eine Kamera allein nicht räumlich sehen. Um dreidimensionale Szenen aus Bilder zu rekonstruieren, müssen mindestens zwei Aufnahmen der gleichen Szene aus unterschiedlichen Blickwinkeln aufgenommen werden. Innerhalb dieser Aufnahmen müssen Punktekorrespondenzen gesucht werden. Korrespondierende Punkte zeichnen sich dadurch aus, dass sie die Abbildungen desselben Ursprungspunktes im Raum sind. Für diese Punkte muss eine gemeinsame Abbildungsvorschrift aufgestellt werden. Die Abbildungsvorschrift wird in einer 3×3-Matrix H ausgedrückt, welche die Transformationsmatrizen sowie die Kameramatrizen zusammenfasst. Die 3×3-Matrix, kann aus gegebenen Punktekorrespondenzen abgeleitet werden. Aus H können Rückschlüssen auf die Kameraparameter der beiden Kameras gezogen werden.

Es seien $m_\tau = (m_{x\tau}, m_{y\tau}, m_{z\tau})^T$ die homogenen Koordinaten eines Punktes auf der Bildebene(I, τ) und $m'_{\tau'} = (m'_{x\tau'}, m'_{y\tau'}, m'_{z\tau'})^T$ der dazu korrespondierende Punkt der Bildebene (I', τ'), vergleiche hierzu Abbildung 3.1. Gesucht wird eine Abbildungsvorschrift welche als Matrix H ausgedrückt wird:

$$m'_{\tau'} = Hm_\tau \tag{3.1}$$

$$Hm_\tau = \begin{bmatrix} {h_1}^T \cdot m_{x\tau} \\ {h_2}^T \cdot m_{y\tau} \\ {h_3}^T \cdot m_{z\tau} \end{bmatrix} \tag{3.2}$$

$$\rightsquigarrow m'_{\tau'} = Hm_\tau = \begin{bmatrix} h_{11}m_{\tau x} + h_{12}m_{\tau y} + h_{13}m_{\tau z} \\ h_{21}m_{\tau x} + h_{22}m_{\tau y} + h_{23}m_{\tau z} \\ h_{31}m_{\tau x} + h_{32}m_{\tau y} + h_{33}m_{\tau z} \end{bmatrix} \tag{3.3}$$

$$\rightsquigarrow H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{3.4}$$

In den folgenden Unterkapiteln werden zwei Herleitungen der Abbildungsvorschriften zwei unterschiedlicher Fälle gesucht. Im ersten Fall wird vorausgesetzt, dass die 3D-Punkte im Raum auf einer Ebene liegen und auf die Bildebenen von zwei zueinander verschobenen und rotierten Kameras abgebildet werden. Im zweiten Fall werden die Punkte eines komplexeren 3D-Objektes auf die beiden Bildebenen abgebildet. Im letzten Abschnitt wird die Herleitungen der entstehenden Matrizen beider Fälle anhand von Punktekorrespondenzen aufgezeigt.

3.1 Korrespondenzen planarer Punktmengen mit Homographien

Eine Abbildungsvorschrift kann in bestimmten Fällen eindeutig bestimmt werden. In diesen Fällen nennt man die Abbildung zwischen beiden zweidimensionalen Bildern Homographie[3, 5, 9]. In diesem Kapitel wird der beispielhafte Fall behandelt, dass Punkte auf der x, y-Ebene im Weltkoordinatensystem auf zwei unterschiedlichen Kameras C und C' abgebildet werden. Dies ist in Abbildung 3.1 dargestellt.

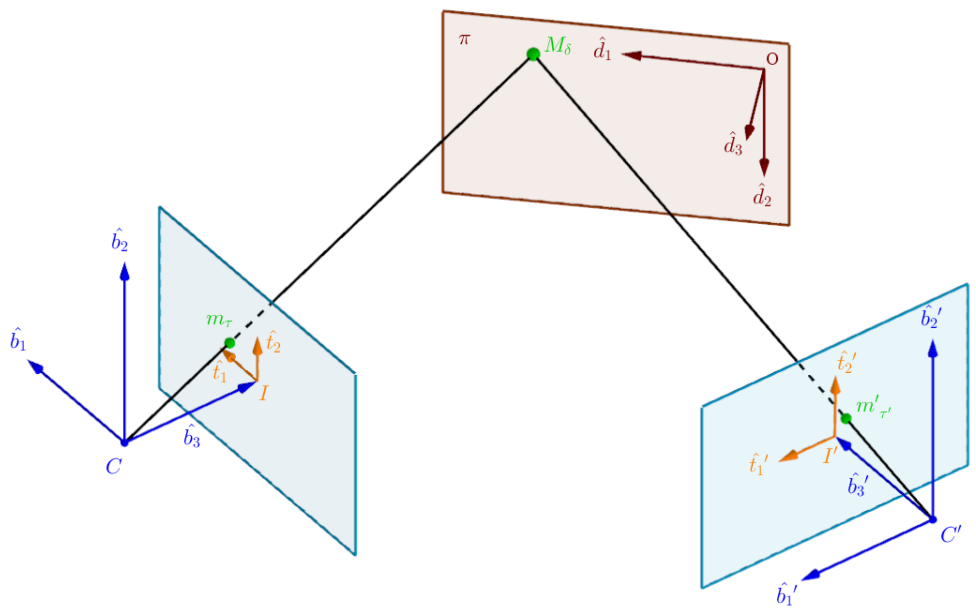


Abbildung 3.1: In der Abblidung sind die beiden Kameras C und C' mit ihren Bildebenen I und I' zu sehen. Ein Objektpunkt M_δ bezüglich eines Weltkoordinatensystems (O, δ) befindet sich auf der Ebene π, welche durch die Achsen $\hat{d_1}$ und $\hat{d_2}$ aufgespannt wird. M_δ wird auf I und I' projiziert. Es entstehen die Bildpunkte m_τ und m'_τ.

Um die Abbildungsvorschrift herzuleiten wird bei Bildpunkt $m_\tau = (m_{\tau,1}, m_{\tau,2}, m'_{\tau,3})^T$ mit $m_{\tau,3} = 1$ begonnen. Während in der Bildaufnahme ein Punkt m_β durch Division mit der $m_{\beta,3}$-Komponenten eindeutig zu m_τ wird ist die Rückrichtung nicht eindeutig. Alle Punkte auf der Geraden von C und m_τ, siehe Abbildung 3.2,werden auf denselben Bildpunkt projiziert. Die Projektion von m_τ auf m_β ist demnach nicht eindeutig. Die Gerade mit allen möglichen Punkten m_β kann als γm_τ mit der freien Variablen $\gamma > 0$ ausgedrückt werden.

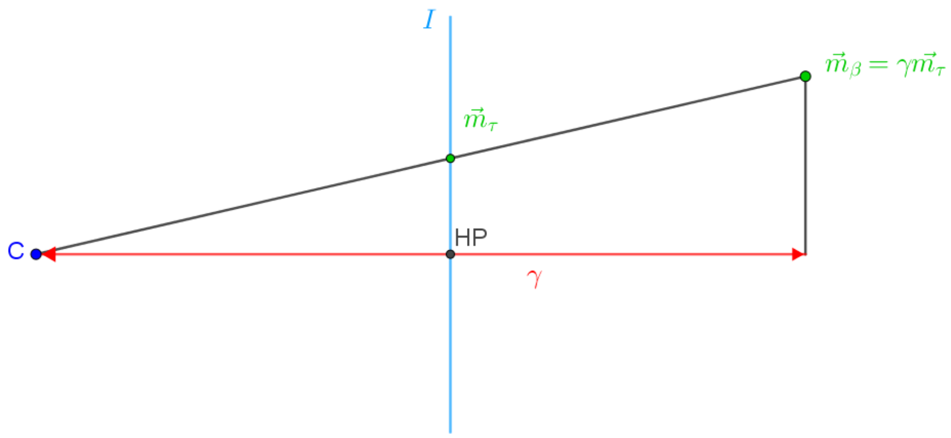


Abbildung 3.2: Auf der Geraden durch C und $\vec{m}_\tau$, befinden sich alle möglichen Punkte für m_β. m_β wird auf Grund seiner Unbestimmtheit als γm_τ bezeichnet

Für zwei korrespondierende Bildpunkte m_τ und $m'_{\tau'}$ kann für alle möglichen Punkte m_β und m'_β die folgende Projektionsvorschrift hergeleitet werden [5].

$$\gamma \vec{m}_\tau = P \begin{bmatrix} \vec{M}_\delta \\ 1 \end{bmatrix} = \begin{bmatrix} KR| - KR\vec{C}_\delta \end{bmatrix} \cdot \begin{bmatrix} \vec{M}\delta \\ 1 \end{bmatrix} = KR(\vec{M}\delta - \vec{C}_\delta) \tag{3.5}$$

$$\gamma' \vec{m'}_{\tau'} = P' \begin{bmatrix} \vec{M'}_\delta \\ 1 \end{bmatrix} = \begin{bmatrix} K'R'| - K'R'\vec{C'}_\delta \end{bmatrix} \cdot \begin{bmatrix} \vec{M}\delta \\ 1 \end{bmatrix} = K'R'(\vec{M}\delta - \vec{C'}_\delta) \tag{3.6}$$

Mit einem Ursprungspunkt $M_\delta = (x_\delta, y_\delta, 0)^T$ auf der x, y- Ebene im Weltkoordinatensystem und einer unbekannten Projektionsmatrix P mit

$$P = \begin{bmatrix} p_{11} & p_{21} & p_{31} & p_{41} \\ p_{12} & p_{22} & p_{32} & p_{42} \\ p_{13} & p_{23} & p_{33} & p_{43} \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix} \tag{3.7}$$

kann die folgende Gleichung aufgestellt werden [5]

$$\gamma m_\tau = P \cdot \begin{bmatrix} M_\delta \\ 1 \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix} \cdot \begin{bmatrix} x_\delta \\ y_\delta \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_4 \end{bmatrix} \cdot \begin{bmatrix} x_\delta \\ y_\delta \\ 1 \end{bmatrix} = G\vec{m}_\delta \tag{3.8}$$

$$\gamma' m'_{\tau'} = P \cdot \begin{bmatrix} M_\delta \\ 1 \end{bmatrix} = \begin{bmatrix} p'_1 & p'_2 & p'_3 & p'_4 \end{bmatrix} \cdot \begin{bmatrix} x_\delta \\ y_\delta \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p'_1 & p'_2 & p'_4 \end{bmatrix} \cdot \begin{bmatrix} x_\delta \\ y_\delta \\ 1 \end{bmatrix} = G'\vec{m}_\delta. \tag{3.9}$$

Aus $\gamma m_\tau = G \cdot m_\delta$ und $\gamma' m'_{\tau'} = G' \cdot m_\delta$ kann dann Folgendes abgeleitet werden[5].

$$\gamma' m'_{\tau'} = G' G^{-1} \gamma m_\tau \tag{3.10}$$

Mit $\lambda = \frac{\gamma'}{\gamma}$, kann Gleichung 3.10 dann wieder umformuliert und in die Bedienungsgleichung der Homographie mit $H = G'G^{-1}$ umgeformt werden:

$$\lambda m'_{\tau'} = H m_\tau \tag{3.11}$$

Die entstandene Homographie H ist somit eine Abbildungsvorschrift welche zwei korrespondierende Punkte in Verbindung setzt. Diese Homographiebedingung stellt ein Gleichungssystem mit neun unbekannten, welche in Kapitel 3.3 gelöst wird[3].

3.2 Korrespondenzanalyse für beliebige Punkte im Raum (Epipolare Geometrie)

Für Bilder von komplexeren, dreidimensionalen Objekten, bei denen die Punkte auf verschiedenen Ebenen liegen können, kann keine Homographiebedingung hergestellt werden um die Kameraparameter zu bestimmen. Jedoch kann auf geometrische Bedingungen zurückgegriffen werden, um die Abbildungsvorschrift zwischen den Bildern auszunutzen und die Kameraparameter beider Kameras zu bestimmen. Ein Ursprungspunkt M_δ wird wieder mit zwei Kameras C und C' aufgenommen. In Abbildung 3.3 ist das stereoskopische System dargestellt.

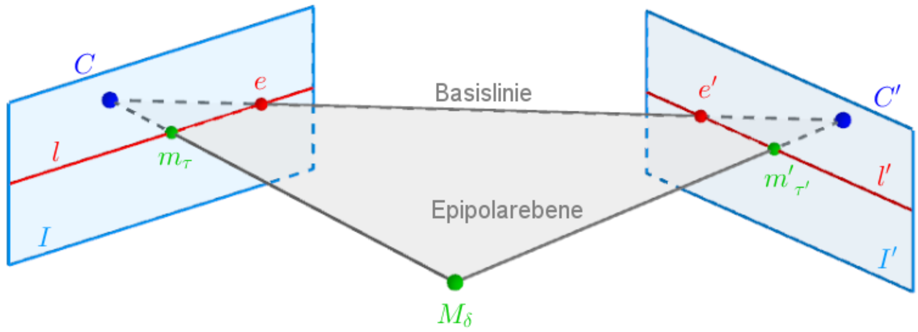


Abbildung 3.3: C und C' sind die Projektionszentren zweier Kameras. Beide Kameras besitzen jeweils eine Bildebene. Die Basislinie verbindet die Projektionszentren der Kameras. Die Punkte an welchen die Basislinie die Bildebenen schneidet werden als Epipole e und e' bezeichnet. Durch einen Epipol verlaufen alle Epipolarlinien des Bildes. M_δ ist der Objektpunkt im 3D-Raum und m_τ und $m'_{\tau'}$ sind die jeweiligen Abbildungen dieses Punktes auf den Bildebenen. Die Verbindungsvektoren zwischen C, C' und M_δ bilden die sogenannte Epipolarebene[10, 11, 3, 1].

Es werden hier einige geometrische Definitionen eingeführt um die danach folgende mathematische Herleitung genauer zu verstehen. Die Vektoren $\overline{CM} = (\vec{M}_\delta - \vec{C}_\delta)$, $\overline{C'M} = (\vec{M}_\delta - \vec{C'}_\delta)$ und $\overline{CC'} = (\vec{C'}_\delta - \vec{C}_\delta)$ definieren die Epipolarebene, die durch das schwarze Dreieck in Abbildung 3.3 gekennzeichnet ist. Die Schnittpunkte der Geraden zwischen C und C' mit der jeweiligen Bildebene I und I' werden als Epipole e und e' bezeichnet. Die Schnittgerade der Epipolarebene mit I und I' bilden die sogenannten Epipolarlinien l und l'[3, 12, 13, 11].

Ein Bildpunkt m_i auf der Bildebene I wird zuerst auf die Gerade, die durch m_i und C geht abgebildet. Die Gerade stellt alle möglichen Ursprungspunkte zu m_i dar. Dies ist durch die drei möglichen Punkte $M1$,$M2$,$M3$ in Abbildung 3.4 dargestellt. Jeder dieser Punkte wird nun wiederum auf I' projiziert. Die so entstandenen Punkte liegen alle auf der Epipolarlinie l'[3].

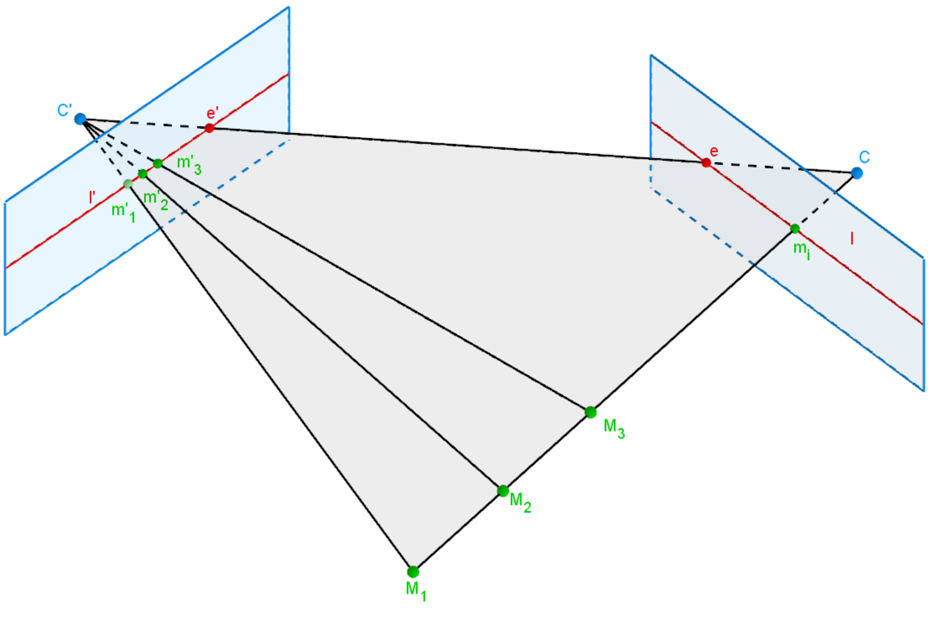


Abbildung 3.4: Die Objektpunkte M_1, M_2 und M_3 werden in I' als m'_1, m'_2 und m'_3 abgebildet, während sie in I immer den selben Bildpunkt m_1 ergeben.

Die hier gezeigte Abbildung von m_i auf l' wird nun genauer betrachtet. Es werden wieder die Gleichungen für m_τ und m'_τ, wie in Gleichung 3.6, aufgestellt

$$\gamma\vec{m}_\tau = P\begin{bmatrix}\vec{M}_\delta\\1\end{bmatrix} = \begin{bmatrix}KR| - KR\vec{C}_\delta\end{bmatrix}\cdot\begin{bmatrix}\vec{M}\delta\\1\end{bmatrix} = KR(\vec{M}\delta - \vec{C}_\delta) \tag{3.12}$$

$$\gamma' \vec{m'_{\tau'}} = P'\begin{bmatrix}\vec{M'}_\delta\\1\end{bmatrix} = \begin{bmatrix}K'R'| - K'R'\vec{C'}_\delta\end{bmatrix}\cdot\begin{bmatrix}\vec{M}\delta\\1\end{bmatrix} = K'R'(\vec{M}\delta - \vec{C'}_\delta) \tag{3.13}$$

Die Gleichungen 3.12 und 3.13 werden nach $(\vec{M} - \vec{C}_\delta)$ und $(\vec{M} - \vec{C'}_\delta)$ aufgelöst.

$$\gamma R^T K^{-1}\vec{m}_\tau = (\vec{M} - \vec{C}_\delta) \tag{3.14}$$

$$\gamma' R'^T K'^{-1}\vec{m'}_{\tau'} = (\vec{M} - \vec{C'}_\delta) \tag{3.15}$$

Wie in Abbildung 3.3 gezeigt bilden die Vektoren $(\vec{M}_\delta - \vec{C}_\delta)$, $(\vec{M}_\delta - \vec{C'}_\delta)$ und $(\vec{C'}_\delta - \vec{C}_\delta)$ ein Dreieck. Für dieses Dreieck kann die folgende Gleichung aufgestellt werden.

$$(\vec{C'}_\delta - \vec{C}_\delta) = (\vec{M}_\delta - \vec{C}_\delta) - (\vec{M}_\delta - \vec{C'}_\delta) \tag{3.16}$$

$(\vec{M} - \vec{C}_\delta)$ und $(\vec{M} - \vec{C'}_\delta)$ können durch die Ausdrücke in den Gleichungen 3.14 und 3.15 ersetzt werden um folgende Gleichung zu erhalten

$$(\vec{C'}_\delta - \vec{C}_\delta) = \gamma' R^T K^{-1}\vec{m}_\tau - \gamma R'^T K'^{-1}\vec{m'}_{\tau'}. \tag{3.17}$$

Durch Vektoridentitäten können γ und γ' eliminiert und folgende Bedingung aus Gleichung 3.17 hergeleitet werden [14]:

$$\vec{m'_\tau}^T K'^{-T} T R' \left[\vec{C'_\delta} - \vec{C}_\delta\right]_\times R^T K^{-1}\vec{m}_\tau = \vec{m'_\tau} F \vec{m}_\tau = 0 \tag{3.18}$$

mit

$$\left[\vec{C'_\delta} - \vec{C}_\delta\right]_\times = \begin{bmatrix} 0 & -(C'_{z\delta} - C_{z\delta}) & C'_{y\delta} - C_{y\delta} \\ C'_{z\delta} - C_{z\delta} & 0 & -(C'_{x\delta} - C_{x\delta}) \\ -(C'_{y\delta} - C_{y\delta}) & C'_{x\delta} - C_{x\delta} & 0 \end{bmatrix}. \tag{3.19}$$

In Gleichung 3.18 wurde die Bedingungsgleichung für die sogenannte Fundamentalmatrix F definiert [10]. Sind die Kameraparameter und dadurch die Kameramatrix K bekannt, so wird die essentielle Matrix E mit $E = R'\left[\vec{C'_\delta} - \vec{C}_\delta\right]_\times R^T$ definiert. Gleichung 3.18 kann zu einer Bedingung für E umgeformt werden.

$$\vec{m'_\tau}^T K'^{-T} E K^{-1}\vec{m}_\tau = 0 \tag{3.20}$$

Die Gleichungen 3.18 und 3.20 definieren den sogenannten *Epipolar-Constraint*[3, 10] und können verwendet werden um die Fundamentalmatrix oder die essentielle Matrix aus bekannten Korrespondierenden Punkten zu bestimmen. Für die essentielle Matrix müssen zuvor noch die Koordinaten in der Form

$$\hat{\vec{m}}'_\tau = \vec{m'_\tau}^T K'^{-T} \tag{3.21}$$

$$\hat{\vec{m}}_\tau = K^{-1}\vec{m}_\tau \tag{3.22}$$

zu normierten Bildebenenkoordinaten umgerechnet werden[3, 15]. Der verwendete Algorithmus zur Bestimmung von F wird in Kapitel 3.3 näher beschrieben.

Wenn die Fundamentalmatrix bekannt ist können auch die Epipole e und e' und die Epipolarlinien l und l' aus Eigenschaften der Fundamentalmatrix bestimmt werden [3, 11, 16, 1, 15]. Um die Epipole e zu erhalten, wird der rechte Kern von F bestimmt und für e' muss der linke Kern von F bestimmt werden[3, 11, 16, 1, 15]. Es gilt also

$$Fe = 0 \tag{3.23}$$

$$F^T e' = 0 \tag{3.24}$$

.
Um die zu m oder m' korrespondierende Epipolarlinie l' oder l zu bestimmten kann die folgende Transformation verwendet werden[3, 11, 16, 1, 15]

$$l' = Fm \tag{3.25}$$

$$l = F^T m' \tag{3.26}$$

.
Die Matrizen F und E sind mit diesen Eigenschaften wichtige Instrumente für die Bestimmung der extrinsischen und intrinsischen Kameraparameter und ihre Eigenschaften werden in den folgenden Kapiteln ausgenutzt um effiziente Rekonstruktionalgorithmen für die Szene zu implementieren.

3.3 Bestimmung von Homographie und Fundamentalmatrix aus Punktekorrespondenzen

Im Folgenden wird gezeigt, wie beispielsweise eine Homographie, Fundamentalmatrix und dementsprechend auch eine essentielle Matrix aus Punktekorrespondenzen gewonnen werden können. Für essentielle Matrizen gilt das selbe Verfahren wie für die Fundamentalmatrizen, nur sind hier die Punkte in der Form wie in Gleichung 3.22 gezeigt dargestellt. Die Herleitung selbst wird am Beispiel der Fundamentalmatrix aufgezeigt.

Es wird davon ausgegangen, dass die Transformationsmatrizen R und R' sowie die Kameramatrizen K und K' nicht bekannt sind. Des Weiteren wird vorausgesetzt, dass zuvor mindestens acht korrespondierende Punkte aus den jeweiligen Bildpaaren detektiert wurden. Im Realfall, werden hierfür bestimmte Detektionsalgorithmen verwendet, wie beispielsweise der SURF-Algorithmus[17], welche markante Bildpunkte in beiden Bilder suchen.

Um eine Homographiematrix mit

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{3.27}$$

zu erhalten werden die Punkte beider Kameras in eine Koeffizientenmatrix A eingetragen, welche sich nach dem folgenden Schema aufstellen lässt[3, 5]. Ausgehend von der Abbildungsvorschrift aus Gleichung 3.11 gilt:

$$Hm_\tau = \lambda m'_{\tau'} \tag{3.28}$$

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} \\ m_\tau \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \lambda m'_{\tau'} \\ \\ \end{bmatrix} \tag{3.29}$$

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} m_{x\tau} \\ m_{y\tau} \\ m_{z\tau} \end{bmatrix} = \begin{bmatrix} \lambda m'_{x\tau'} \\ \lambda m'_{y\tau'} \\ \lambda m'_{z\tau'} \end{bmatrix} \tag{3.30}$$

Aus Gleichung 3.30 lässt sich das folgende Gleichungssystem aufstellen.

$$h_{11}m_{x\tau} + h_{12}m_{y\tau} + h_{13}m_{z\tau} = \lambda m'_{x\tau'} \tag{3.31}$$

$$h_{21}m_{x\tau} + h_{22}m_{y\tau} + h_{23}m_{z\tau} = \lambda m'_{y\tau'} \tag{3.32}$$

$$h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau} = \lambda m'_{z\tau'} \tag{3.33}$$

Da mit zweidimensionalen homogenen Bildkoordinaten gearbeitet wird und somit $m_{z\tau}$ und $m'_{z\tau'} = 1$ ist, ergibt sich für die letzte Zeile $h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau} = \lambda$. Setzt man diesen Ausdruck anstelle von λ in die anderen beiden Gleichungen ein, so ergeben sich:

$$h_{11}m_{x\tau} + h_{12}m_{y\tau} + h_{13}m_{z\tau} = (h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau}) \cdot m'_{x\tau'} \tag{3.34}$$

$$h_{21}m_{x\tau} + h_{22}m_{y\tau} + h_{23}m_{z\tau} = (h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau}) \cdot m'_{y\tau'} \tag{3.35}$$

Für den Aufbau von A werden beide Ausdrücke nach Null aufgelöst, sodass sich pro korrespondierendem Punktepaar zwei Gleichungen nach 3.36 und 3.37 ergeben.

$$h_{11}m_{x\tau} + h_{12}m_{y\tau} + h_{13}m_{z\tau} - (h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau}) \cdot m'_{x\tau'} = 0$$
$$h_{21}m_{x\tau} + h_{22}m_{y\tau} + h_{23}m_{z\tau} - (h_{31}m_{x\tau} + h_{32}m_{y\tau} + h_{33}m_{z\tau}) \cdot m'_{y\tau'} = 0$$

$$\rightsquigarrow h_{11}m_{x\tau} + h_{12}m_{y\tau} + h_{13}m_{z\tau} - h_{31}m_{x\tau} \cdot m'_{x\tau'} - h_{32}m_{y\tau} \cdot m'_{x\tau'} - h_{33}m_{z\tau} \cdot m'_{x\tau'} = 0 \tag{3.36}$$

$$\rightsquigarrow h_{21}m_{x\tau} + h_{22}m_{y\tau} + h_{23}m_{z\tau} - h_{31}m_{x\tau} \cdot m'_{y\tau'} - h_{32}m_{y\tau} \cdot m'_{y\tau'} - h_{33}m_{z\tau}) \cdot m'_{y\tau'} = 0 \tag{3.37}$$

Die entstandenen Gleichungen werden dann nach folgendem Schema in die Koeffizientmatrix A eingetragen.[5, 3, 18, 6]

$$A \cdot x = 0 \tag{3.38}$$

$$\begin{pmatrix} m_{x\tau} & m_{y\tau} & 1 & 0 & 0 & 0 & m_{x\tau}m'_{x\tau'} & m_{y\tau}m'_{x\tau'} & 1 \cdot m'_{x\tau'} \\ 0 & 0 & 0 & m_{x\tau} & m_{y\tau} & 1 & m_{x\tau}m'_{y\tau'} & m_{y\tau}m'_{y\tau'} & 1 \cdot m'_{y\tau'} \\ & & & & & . & & & \\ & & & & & . & & & \\ & & & & & . & & & \\ m_{i,x\tau} & m_{i,y\tau} & 1 & 0 & 0 & 0 & m_{i,x\tau}m'_{i,x\tau'} & m_{i,y\tau}m'_{i,x\tau'} & 1 \cdot m'_{i,x\tau'} \\ 0 & 0 & 0 & m_{i,x\tau} & m_{i,y\tau} & 1 & m_{i,x\tau}m'_{i,y\tau'} & m_{i,y\tau}m'_{i,y\tau'} & 1 \cdot m'_{i,y\tau'} \end{pmatrix} \cdot \begin{pmatrix} h1 \\ h2 \\ . \\ . \\ . \\ hi \end{pmatrix} = 0 \tag{3.39}$$

Gesucht wird ein Vektor $\vec{x}$, für den gilt, dass $A \cdot x = 0$. Besitzt Matrix A einen Rang von 8, so entspricht der gesuchte Vektor $\vec{x}$ dem Kern der Koeffizientenmatrix und ist ein Spaltenvektor mit insgesamt neun Einträgen, welche in die 3x3-Homographiematrix eingetragen werden können[3, 18].

Das Verfahren mit welchem sowohl F als auch E geschätzt werden können ähnelt in seinem Aufbau dem der Bestimmung der Homographiematrix. Das Verfahren wird hier allgemein als der 8-Punkte-Algorithmus bezeichnet[3, 4, 12]. Der 8-Punkte-Algorithmus ist eine lineare Technik, welche angewandt wird, um die Fundamentalmatrix aus $n \geq 8$ Punkten schätzen zu können. Der Algorithmus benötigt $n \geq 8$ Punkte, um ein valides Ergebnis zu liefern [3, 12, 4]. Das Ergebnis und jedes seiner Vielfachen ist eine mögliche Lösung für F. Der Algorithmus wird am Beispiel für die Bestimmung von F veranschaulicht. Zunächst wird eine Koeffizientenmatrix A aus Punktekorrespondenzen gebildet. Hierzu wird sich auf die für F hergeleitete Gleichung 3.18 bezogen.

$$m'^{T}_{\tau'} \cdot F \cdot m_{\tau} = 0$$

$$F = \begin{bmatrix} f_{11} & f_{122} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$$

$$\begin{bmatrix} m'_{x\tau'} & m'_{y\tau'} & 1 \end{bmatrix} \cdot \begin{bmatrix} f_{11} & f_{122} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \cdot \begin{bmatrix} m_{x\tau} \\ m_{y\tau} \\ 1 \end{bmatrix} = 0$$

$$\begin{aligned} f_{11} m_{x\tau} m'_{x\tau'} + f_{12} m_{y\tau} m'_{x\tau'} + f_{13} m'_{x\tau'} + f_{21} m_{x\tau} m'_{y\tau'} \\ + f_{22} m_{y\tau} m_{y'\tau'} + f_{23} m'_{y\tau'} + f_{31} m_{x\tau} + f_{32} m_{y\tau} + f_{33} = 0 \end{aligned} \tag{3.40}$$

$$(m_{x\tau} m'_{x\tau'}, m_{y\tau} m'_{x\tau'}, m'_{x\tau'}, m_{x\tau} m'_{y\tau'}, m_{y\tau} m'_{x\tau'}, m'_{x\tau'}, m_{x\tau}, m_{y\tau}, 1) \cdot f = 0$$

$$\begin{bmatrix} m_{x\tau} m'_{x\tau'} & m_{y\tau} m'_{x\tau'} & m'_{x\tau'} & m_{x\tau} m'_{y\tau'} & m_{y\tau} m'_{y\tau'} & m'_{y\tau'} & m_{x\tau} & m_{y\tau} & 1 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ m_{i,x\tau} m'_{i,x\tau'} & m_{i,y\tau} m'_{i,x\tau'} & m'_{i,x\tau'} & m_{i,x\tau} m'_{i,y\tau'} & m_{i,y\tau} m'_{i,y\tau'} & m'_{i,y\tau'} & m_{i,x\tau} & m_{i,y\tau} & 1 \end{bmatrix} \cdot \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = 0$$

$$A \cdot f = 0$$

Gesucht wird nun ein Vektor $\vec{f}$, für den gilt, dass $A \cdot f = 0$. Besitzt Matrix A einen Rang von 8, so entspricht der gesuchte Vektor $\vec{x}$ auch hier dem Kern von A und ist ein Spaltenvektor mit insgesamt neun Einträgen, welche in die 3x3-Fundamentalmatrix eingetragen werden können[3, 1].

Bei der Homographie, wie auch bei der Fundamentalmatrix, kann es zu überbestimmten Systemen kommen. Ein System gilt als überbestimmt, wenn es durch mehr Gleichungen als Unbekannte beschrieben wird[18, 19]. Für die Koeffizientenmatrix für F und H hätte das zur Folge, dass sie in ihrem Rang steigt. Die Bestimmung des Kerns würde in beiden Fällen kein eindeutiges Ergebnis mehr liefern[3, 18].

Für die Lösung überbestimmter Systeme wird durch ein *Least-Square*-Verfahren, mit Hilfe der Singulärwertszerlegung einer Matrix A, eine Lösung für einen Vektor $\vec{x}$ gesucht, so dass $\parallel A \cdot x \parallel$ minimal wird [3, 19, 18]. Die Singulärwertzerlegung von A ist eine Faktorisierung der Matrix $A \in \mathbb{R}^{m \times n}$ der Form $A = U \cdot \Sigma \cdot V^T$ mit orthogonalen Matrizen $U \in \mathbb{R}^{m \times n}$ und $V \in \mathbb{R}^{m \times n}$ sowie mit einer Diagonalmatrix Σ.

$$S = \begin{pmatrix} s_1 & & ... & 0 & 0 & ... & 0 \\ . & . & & . & . & & . \\ . & & . & . & . & & . \\ . & & & . & . & & . \\ 0 & & ... & s_r & 0 & ... & 0 \\ 0 & & ... & 0 & 0 & ... & 0 \\ . & & & . & . & & . \\ . & & & . & . & & . \\ . & & & . & . & & . \\ 0 & & ... & 0 & 0 & ... & 0 \end{pmatrix} \tag{3.41}$$

Die Diagonalmatrix Σ beinhaltet die Singulärwerte der Matrix. Dabei soll für die diagonalen Singulärwerte in Σ mit s_1 bis s_r gelten, dass $s_1 \geq s_2 \geq ... \geq s_r \geq 0$[19]. Die Spalte der Matrix V^T, welche mit dem kleinsten Singulärwert von Σ korrespondiert, ergibt den Vektor $\vec{x}$, für den $\parallel A \cdot x \parallel$ minimal wird.

4 Synthetische Rekonstruktion

Anhand der erarbeiteten mathematischen Grundlagen ist ein Algorithmus für die Rekonstruktion einer Szene aus einer Stereobildaufnahme entstanden. Der Algorithmus wurde mit dem Ziel der Kamerakalibrierung und der Szenenrekonstruktion aus Bildquellen unterschiedlicher Auflösungen entwickelt, da Stereokalibrierungsverfahren einiger Computer Vision Applikationen keine unterschiedlichen Auflösungen von Kameras berücksichtigen. Der entwickelte Algorithmus ist in der Lage aus einem Stereobildpaar extrinsische Kameraparameter zu bestimmen und anhand dessen die 3D-Szene zu rekonstruieren, jedoch unter der Voraussetzung, dass die intrinsischen Kameraparameter beider Kameras bekannt sind.

Im Folgenden wird der Algorithmus anhand eines virtuellen Beispiels erklärt. Dabei werden die einzelnen Schritte des Aufbaus der virtuellen Szene, der Bestimmung der extrinsischen Kameraparameter und der Rekonstruktion der virtuellen 3D-Szene beschrieben. Abbildung 4.1 fasst den Arbeitsprozess des Szenenrekonstruktionsalgorithmus für das virtuelle Beispiel zusammen.

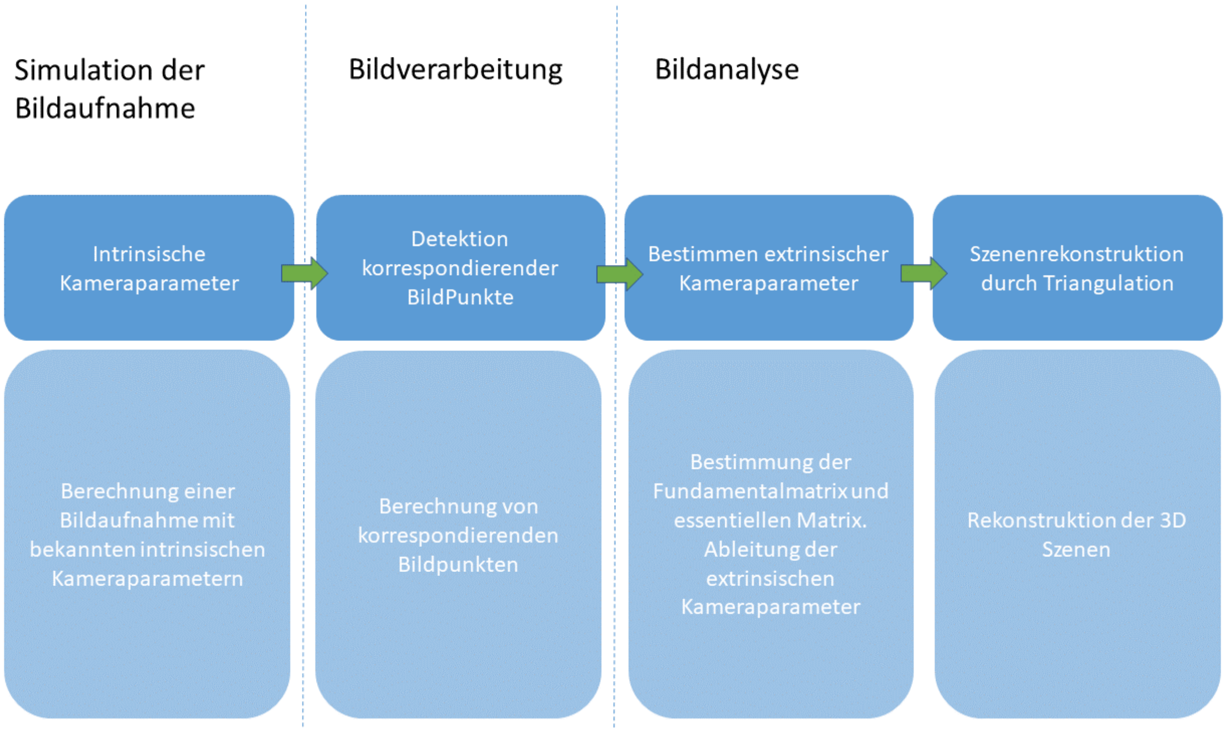


Abbildung 4.1: Ablaufdiagramm für das synthetische Beispiel

4.1 Simulierte Bildaufnahme einer virtuellen Szene

Als 3D-Objekt wurde ein Quader, in ein Weltkoordinatensystem (O, δ) mit $\delta = (\hat{d_1}, \hat{d_2}, \hat{d_3})$ positioniert. Es werden zwei Kameras (C, β) mit $\beta = (\hat{b_1}, \hat{b_2}, \hat{b_3})$ und (C', β') mit $\beta' = (\hat{b'_1}, \hat{b'_2}, \hat{b'_3})$ in (O, δ) platziert. Das Weltkoordinatensystem (O, δ) und das Kamerakoordinatensystem (C, β) sind deckungsgleich. C' ist relativ zu C verschoben und rotiert. Die zwei Bildebenen (I, τ) mit $\tau = (\hat{t_1}, \hat{t_2}, \hat{t_3})$ und (I', τ') mit $\tau' = (\hat{t_1}', \hat{t_2}', \hat{t_3}')$ sind vor C und C' positioniert. Die Sensorkoordinatensysteme (S, σ) und (S', σ') wurden gleich den Bildebenenkoordinatensystemen (I, τ) und (I', τ') gesetzt. Es wird von zwei identischen Kameras ausgegangen und somit werden für den hier diskutierten Fall ausschließlich mit dem vereinfachten Kameramatrizen K_0 gerechnet. Der schematische Aufbau der Szenen ist in den Abbildungen 4.2, 4.3 und 4.4 dargestellt.

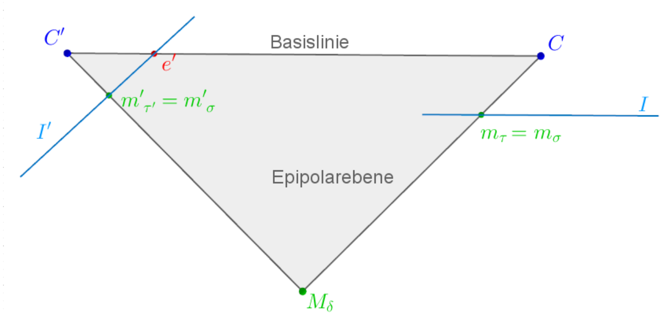


Abbildung 4.2: In der Abbildung ist der vereinfachte Stereoaufbau in einer Top-Down-Ansicht zu sehen

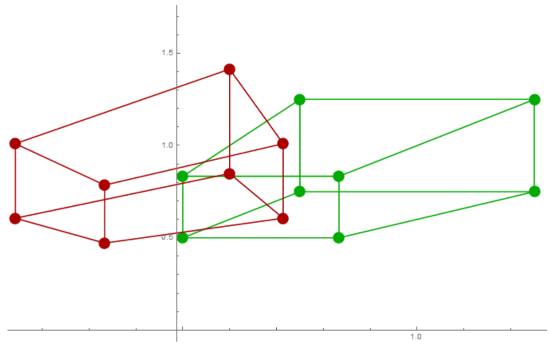


Abbildung 4.3: Simulierite Abbildung des Quaders auf die Kamera C in Grün und auf C' in Rot

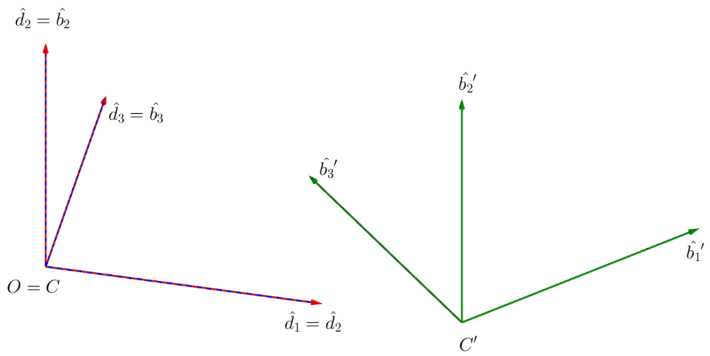


Abbildung 4.4: Abbildung der verschiedenen Kamerakoordinatensysteme, das Weltkoordinatensystem O ist zum Kamerakoordinatensystem C deckungsgleich und C' verschoben und rotiert.

Um die Eckpunkte des Quaders auf die Bildebenen von C und C' abbilden zu können, werden zunächst die Projektionsmatrizen P und P' aufgestellt. C ist deckungsgleich mit dem Weltkoordinatensystem (O, δ). Es gilt also

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.1}$$

$$C = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{4.2}$$

$$T = R[I| - C] \tag{4.3}$$

$$T = \begin{bmatrix} 1 & 0 & 0 & C_1 \\ 0 & 1 & 0 & C_2 \\ 0 & 0 & 1 & C_3 \end{bmatrix} \tag{4.4}$$

.
C' dagegen ist gegenüber C verschoben und rotiert. Als Beispiel wird eine Rotation um die $\hat{b_2}'$ Achse für C' bestimmt. Somit gilt für P':

$$R' = \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix} \tag{4.5}$$

$$\vec{C'} = \begin{pmatrix} 1 & 0 & 0 & -C'_1 \\ 0 & 1 & 0 & -C'_2 \\ 0 & 0 & 1 & -C'_3 \end{pmatrix} \tag{4.6}$$

$$T' = R'[I| - C] \tag{4.7}$$

$$T' = \begin{pmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -C'_1 \\ 0 & 1 & 0 & -C'_2 \\ 0 & 0 & 1 & -C'_3 \end{pmatrix} \tag{4.8}$$

$$T' = \begin{pmatrix} \cos(\alpha) & 0 & -\sin(\alpha) & -C'_1\cos(\alpha) + C'_3\sin(\alpha) \\ 0 & 1 & 0 & -C'_2 \\ \sin(\alpha) & 0 & \cos(\alpha) & -C'_1\sin(\alpha) - C'_3\cos(\alpha) \end{pmatrix} \tag{4.9}$$

Der Quader hat insgesamt acht Punkte, welche auf die Bildebenen der Kameras projiziert werden. Neben den Punkte des Quaders wird noch ein weiterer Punkt außerhalb des Quaders platziert und ebenfalls auf die Bildebenen projiziert. Mit insgesamt neun Punkten bei der Bestimmung der Fundamentalmatrix, wird die Wahrscheinlichkeit ein unterbestimmtes System aus der Koeffizientenmatrix A zu bekommen minimiert. Ist A unterbestimmt so besitzt sie Rang 7 und F kann nicht eindeutig durch einen Sieben-Punkte-Algorithmus bestimmt werden[3, 20]. In dem hier berechneten Beispiel werden deswegen neun Punkte benutzt um sicherzugehen, dass A Rang 8 besitzt und somit eindeutig bestimmt werden kann. Zur Bestimmung von F, wird der in Kapitel 3 aufgeführte Acht-Punkte Algorithmus angewandt.

Um die neun Punkte auf die Bildebenen (I, τ) und (I', τ') zu projizieren, müssen neben den Transformationsmatrizen R und R' noch die Kameramatrizen K_0 und K'_0 festgelegt werden.

$$K_0 = \begin{bmatrix} \zeta_C & 0 & 0 \\ 0 & \zeta_C & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

$$K'_0 = \begin{bmatrix} \zeta_{C'} & 0 & 0 \\ 0 & \zeta_{C'} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

Da zunächst von gleichen Kameraauflösungen ausgegangen wird, gilt $\zeta = \zeta'$. Sind R, R', K_0 und K'_0 bekannt, können die Projektionsmatrizen gebildet und anschließend die Punkte auf die Bildebene projiziert werden. Die entstandenen Bilder sind in Abbildung 4.3 zu sehen.

$$P = K_0 \cdot R \tag{4.12}$$

$$P = \begin{bmatrix} \zeta_C & 0 & 0 \\ 0 & \zeta_C & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

$$P' = K'_0 \cdot R' \tag{4.14}$$

$$P' = \begin{bmatrix} \zeta_{C'} \cos(\alpha) & 0 & \zeta_{C'} \sin(\alpha) & -\zeta_{C'}(v'_1 \cos(\alpha) + v'_3 \sin(\alpha)) \\ 0 & 1 & 0 & \zeta_{C'} - v'_2 \\ \zeta_{C'} \sin(\alpha) & 0 & \zeta_{C'} \cos(\alpha) & -\zeta_{C'}(v'_1 \sin(\alpha) + v'_3 \cos(\alpha)) \end{bmatrix} \tag{4.15}$$

4.2 Bildanalyse

Der Szenenrekonstruktionsalgorithmus für das synthetische Beispiel ist in drei Abschnitte unterteilt. Zuerst wird aus den Punktekorrespondenzen die Fundamentalmatrix und die essentielle Matrix geschätzt. Mit Hilfe der essentiellen Matrix werden die extrinsischen Kameraparameter bestimmt, um so im letzten Schritt die Szenenpunkte durch Rückprojektion der Bildpunkte mit Hilfe der Kameraparameter rekonstruiert.

4.2.1 Bestimmung der extrinsischen Kameraparameter

Zur Bestimmung der extrinsischen Kameraparameter werden in dem hier berechneten Beispiel neun korrespondierende Punkte bestimmt. Mittels des *Epipolar-Constraint* aus Gleichung 3.18 wird der Acht-Punkt-Algorithmus wie in Kapitel 3.3 beschrieben angewandt, um die Fundamentalmatrix zu bestimmen.

$$0 = \vec{m'_\tau}^T K'^{-T} R' \left[\vec{C'_\delta} - \vec{C_\delta}\right]_\times R^T K^{-1} \vec{m_\tau} \tag{4.16}$$

$$= \vec{m'_\tau} F \vec{m_\tau} \tag{4.17}$$

Wie in Kapitel 3.2 hergeleitet, bildet sich die essentielle Matrix aus:

$$\vec{m'_\tau}^T K'^{-T} R' \left[\vec{C'_\delta} - \vec{C_\delta}\right]_\times R^T K^{-1} \vec{m_\tau} = 0 \tag{4.18}$$

$$\rightsquigarrow E = R' \left[\vec{C'_\delta} - \vec{C_\delta}\right]_\times R^T \tag{4.19}$$

Um also E aus F zu bestimmen, gilt:

$$E = K'^T F K \tag{4.20}$$

$$E = K'^T (K'^{-T} R' \left[\vec{C'_\delta} - \vec{C_\delta}\right]_\times R^T K^{-1}) K \tag{4.21}$$

$$E = R' \left[\vec{C'_\delta} - \vec{C_\delta}\right]_\times R^T \tag{4.22}$$

Es wird davon ausgegangen, dass für $T = [R| - RC]$ von C gilt, dass $T = [I| - 0]$ ist. Die aus E zu ermittelnde Matrix T' beschreibt dann die Transformation von C' relativ zu C[3, 4]. Somit kann E umformuliert werden zu:

$$E = R'[\vec{C'_\delta} - \vec{C_\delta}]_\times R^T \tag{4.23}$$

$$E = R'[\vec{C'_\delta} - 0]_\times I^T \tag{4.24}$$

$$E = R'[\vec{C'_\delta}]_\times \tag{4.25}$$

Um R' und $[\vec{C'}_\delta]_\times$ zu bestimmen wird zunächst die essentielle Matrix E, mit Hilfe der Singulärwertszerlegung, in drei Matrizen zerlegt.

$$E = U\Sigma V^T \tag{4.26}$$

Die Singulärwerte $\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)$ der Matrix Σ müssen die Bedingung erfüllen, dass $\Sigma = \mathrm{diag}(1, 1, 0)$[3, 4]. Wenn diese Bedingung nicht erfüllt ist, so wird sie erzwungen. Dazu wird Matrix Σ aus der Sigulärwertszerlegung aus E modifiziert[3, 4].

$$E' = U\mathrm{diag}(1, 1, 0)V^T \tag{4.27}$$

$[C'_\delta]_\times$ ist schiefsymmetrisch und kann in UZU^T zerlegt werden, wobei U eine orthogonale Matrix ist und Z eine block-diagonale Matrix[3]. R' wird in UWV^T und UW^TV^T zerlegt, wobei W eine schiefsymmetrische Matrix ist[4, 3, 15].

$$W = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad Z = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{4.28}$$

Somit lassen sich die folgenden Lösungmöglichkeiten für $[C'_\delta]_\times$ und R_1 und R_2 aufstellen[3, 4].

$$[C'_\delta]_\times = \pm UZU^T \tag{4.29}$$

$$R'_1 = UW^TV^T \quad R'_2 = UWV^T \tag{4.30}$$

$[C'_\delta]_\times$ ist eine schiefsymmetrische Matrix, welche die Information für den noch gesuchten Translationsanteil $v = -R'C'$ beinhaltet. Ohne zusätzliche Informationen kann v nur bis zu einer Skaleninvarianz genau bestimmt werden[3, 4, 15]. Durch die Modifizierung der Singulärwerte von E gilt für $\parallel v \parallel = 1$[3, 4]. Das bedeutet, dass es sich bei dem Translationsvektor v lediglich um den normierten Richtungsvektor zwischen C und C' handelt[21]. Um v aus $[C'_\delta]_\times$ zu extrahieren, wird der Kern von $[C'_\delta]_\times$ bestimmt

$$[C'_\delta]_\times \cdot v = v \times v = 0 \tag{4.31}$$

.

Die Skaleninvarianz bewirkt, dass es bei der Rekonstruktion die Größe der Objekte von ihrer Originalgröße abweichen, da es sich bei v nur um den normierten Richtungsvektor der Ursprünglichen Strecke handelt. Die Abbildungen **??**,**??** und 4.8 zeigen die Auswirkungen von Skaleninvarianz auf die später rekonstruierte Szene.

Letztendlich können für die Rekonstruktion der extrinsischen Kameraparameter vier mögliche Lösungen für T in Form von $T = R[I|-C]$, wie in Gleichung 2.13 in Kapitel 2 definiert, gefunden werden[3, 4, 15]. λv heißt dabei, dass sowohl v also auch alle Vielfache von v Lösungen sein können, was durch die Skaleninvarianz der Resultate bedingt ist[3, 4, 15].

$$T' = [UWV^T| + \lambda v] \quad \text{oder} \quad [UW^TV^T| + \lambda v] \tag{4.32}$$

$$\textit{oder} \quad [UWV^T| - \lambda v] \quad \text{oder} \quad [UW^TV^T| - \lambda v] \tag{4.33}$$

Die Abbildungen 4.5 und 4.6 stellen schematisch die vier verschiedenen Transformationsmöglichkeiten von T' dar. Die richtige Lösung T ist diejenige, bei der das Abbild der Objekte vor den Kameras liegt.

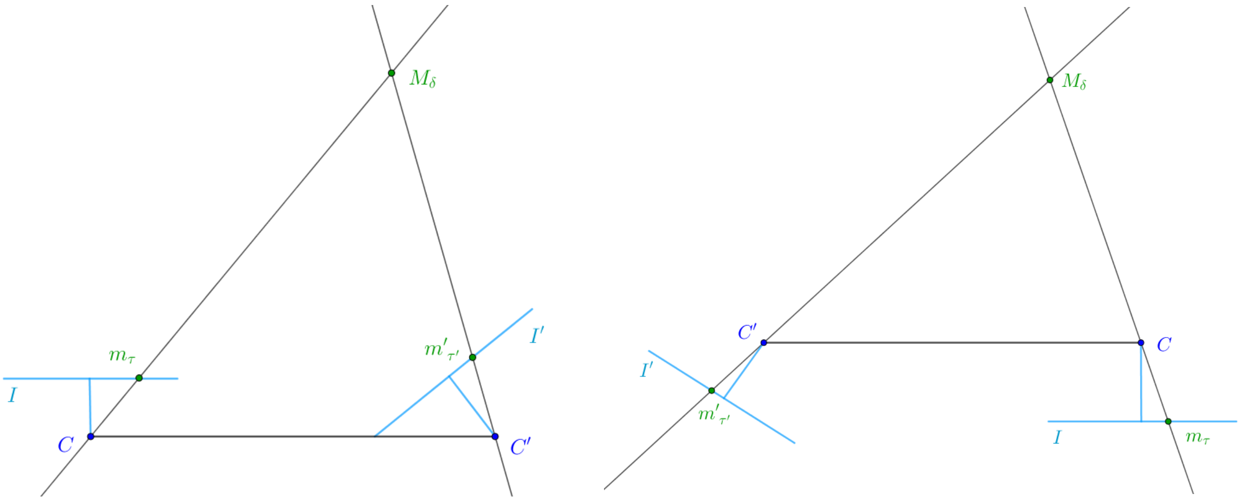


Abbildung 4.5: In den ersten beiden Abbildungen kommt es zu einer Umkehrung der Basisline.

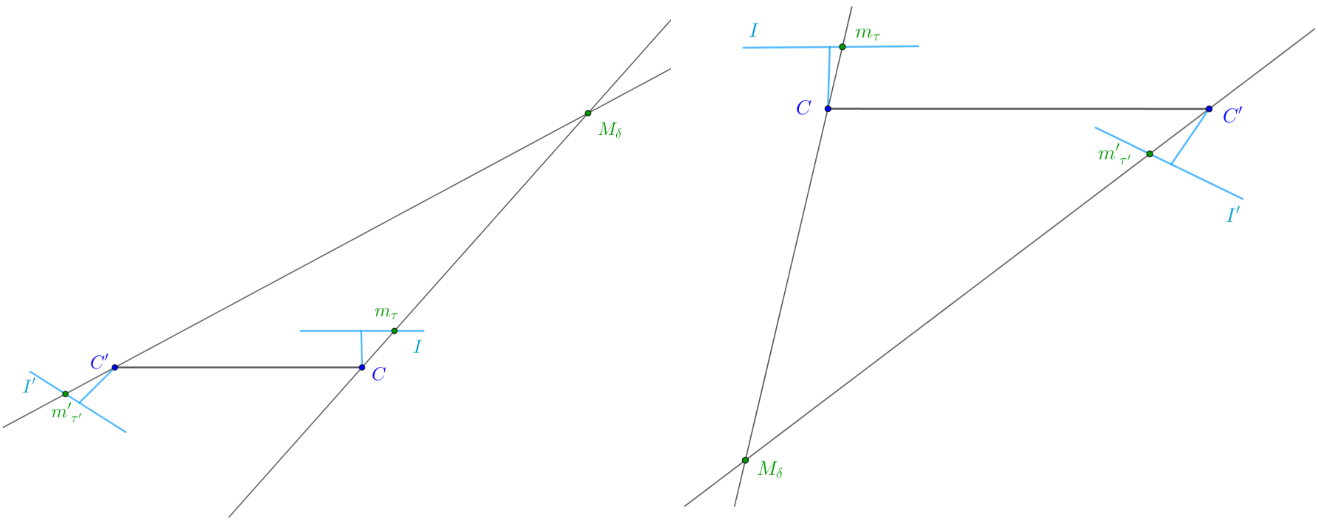


Abbildung 4.6: In den beiden Abbildungen wird C' und 180° gedreht

4.2.2 Szenenrekonstruktion durch Triangulation

Als Triangulierung wird in der Computer Vision die Bestimmung eines 3D-Objektpunktes aus korrespondierenden Bildpunkten bezeichnet. Als Voraussetzung für die Rekonstruktion müssen die jeweiligen korrespondierenden Bildpunkte und die Kameraparameter der einzelnen Kameras bekannt sein. Die Triangulierung funktioniert wie eine umgekehrte Projektion der Bildpunkte auf der Bildebene in einen Objektpunktes im Raum. Zwei Geraden, welche jeweils durch die Projektionszentren und den zu rekonstruierenden Bildpunkten gehen, treffen sich im Raum. Der Schnittpunkt beider Geraden bildet den zu den Bildpunkten gehörenden Ursprungspunkt, wie in Abbildung 4.7 schematisch dargestellt ist.

Im synthetischen Beispiel wird mit reinen Daten gearbeitet. Das heißt, die Bildpunkte wurden mathematisch von ihrem Ursprungspunkt im Raum berechnet und sind somit frei von Verfälschungen durch äußere Einflüsse. Ein zum Bildpunkt m_τ korrespondierender Bildpunkt $m'_{\tau'}$ liegt genau auf der zu m_τ korrespondierenden Epipolarlinien l'. Somit ist garantiert, dass sich die Bildpunkte m_τ und $m'_{\tau'}$ bei einer Rückprojektion in einem Punkt $M_{0,\delta}$ im Raum treffen. Durch die zuvor erwähnte Skaleninvarianz der extrinsischen Kameraparameter handelt es sich bei $M_{0,\delta}$ jedoch noch nicht um den eigentlichen Ursprungspunkt M_δ.

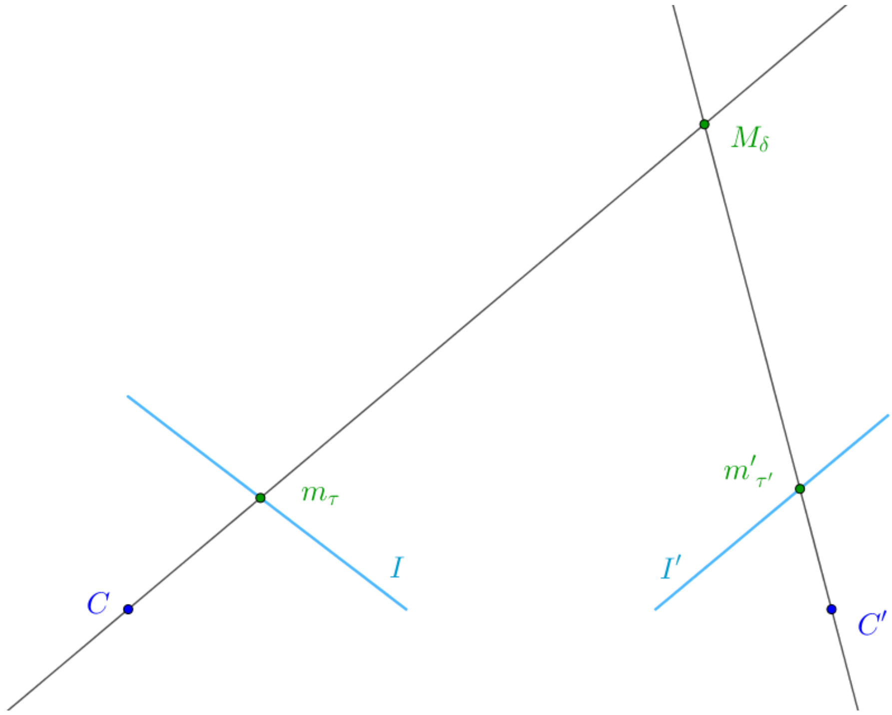


Abbildung 4.7: Optimale Triangulierung: Beide Geraden Treffen sich in einem Punkt im 3D-Raum

Vor der Bestimmung der extrinsischen Kameraparameter wurde festgesetzt, dass $T = [I| - 0]$ die Translationsmatrix von C ist. Somit gilt für die Projektionsmatrix von C, dass $P = K_0T = K_0[I| - 0]$ ist. Die Projektionsmatrix P' für C' setzt sich aus einer der Lösungen von T' und der Kameramatrix K'_0 zusammen, sodass gilt $P' = K'_0T' = K[R'| - RC]$.

Um eine Gerade von den Projektionszentren C und C' durch die jeweiligen Bildpunkte bilden zu können, müssen die Positionen von C und C' bekannt sein. Da die Koordinatensysteme (C, β) und (O, δ) deckungsgleich sind, und $P = K_0[I| - 0]$ ist, gilt $C = (0, 0, 0)^T$. Um C' aus $T' = R'[R'| - R'C']$ zu bestimmen wird der Translationsvektor $-R'C'$ aus T' mit dem transponierten Rotationsmatrix R'^T aus T' multipliziert.

$$C' = R'^T \cdot -R'C' \tag{4.34}$$

Die zweidimensionalen Bildpunkte werden mit den bekannten Brennweiten ζ und ζ' aus K_0 und K'_0 zu einer dreidimensionalen Koordinate erweitert.

$$\begin{pmatrix} m_{\tau x} \\ m_{\tau y} \end{pmatrix} \rightsquigarrow \begin{pmatrix} m_{\tau x} \\ m_{\tau y} \\ \zeta \end{pmatrix} \tag{4.35}$$

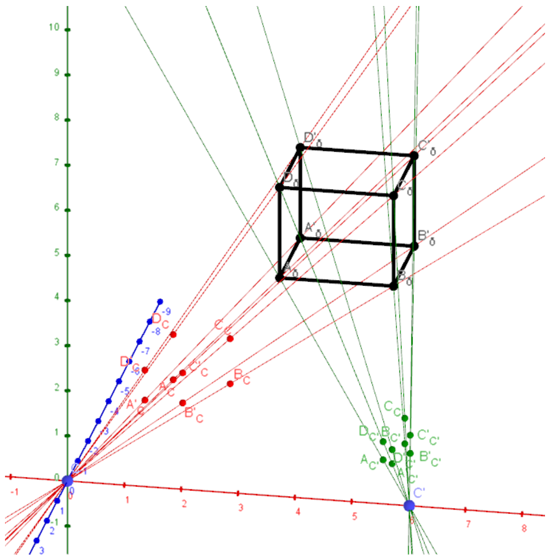
$$\begin{pmatrix} m'_{\tau' x} \\ m'_{\tau' y} \end{pmatrix} \rightsquigarrow \begin{pmatrix} m'_{\tau' x} \\ m'_{\tau' y} \\ \zeta \end{pmatrix} \tag{4.36}$$

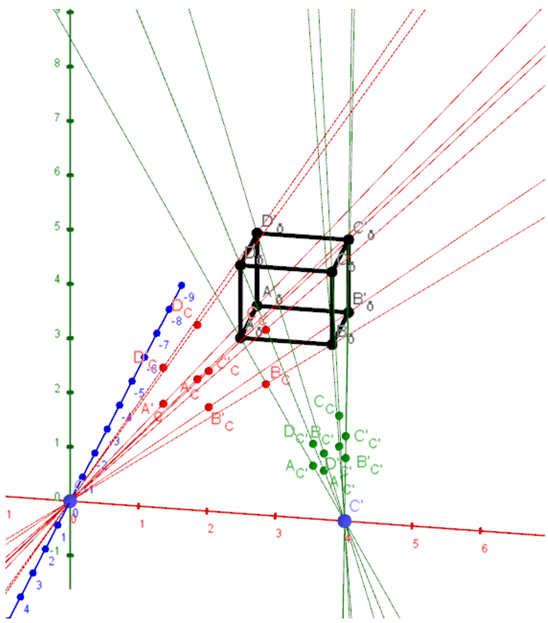
Danach werden für die Rückprojektion zwei Geradengleichungen aufgestellt. Eine Gerade geht durch C und $\begin{pmatrix} m_{\tau x} \\ m_{\tau y} \\ \zeta \end{pmatrix}$ die zweite Gerade geht durch die Punkte C' und $\begin{pmatrix} m'_{\tau' x} \\ m'_{\tau' y} \\ \zeta' \end{pmatrix}$. Anschließend wird aus den zwei Geraden der Schnittpunkt $M_{\delta,0}$ im Raum bestimmt. C und C' sind aus Sicht des Weltkoordinatesystems (O, δ) definiert. $m'_{\tau'}$ wird in Koordinaten bezüglich des Kamerakoordinatesystem (C, β) transformiert mit $m'_\beta = \left[R[I| - C']\right]^{-1} \cdot m'_{\tau'}$.

$$g := \vec{C} + t \cdot \left(\vec{C} - \begin{pmatrix} m_{\beta x} \\ m_{\beta y} \\ \zeta \end{pmatrix} \right) \tag{4.37}$$

$$g' := \vec{C}' + t \cdot \left(\vec{C}' - \begin{pmatrix} m'_{\beta x} \\ m'_{\beta y} \\ \zeta' \end{pmatrix} \right) \tag{4.38}$$

Bei den zuvor ermittelten extrinsischen Kameraparametern ist der Translationsvektor skaleninvariant. Dies führt dazu, dass der rekonstruierte Objektpunkt $M_{\delta,0}$ nach der Szenenrekonstruktion noch nicht dem Ursprünglichen M_δ entsprechen muss. Dementsprechend wird als letzter Schritt die rekonstruierte Szenen anhand einer bekannten Referenzgröße skaliert. Als Referenzgröße kann beispielsweise ein zuvor abgemessener Abstand zwischen zwei Punkten in der Szene dienen. Im synthetischen Aufbau sind beispielsweise die Abstände zwischen den Originalbildpunkten bekannt. Die Abbildung 4.8 zeigt die Szene des Quaders mit unterschiedlichen Skalierungen. Die Abbildungen **??** und **??** zeigt die rekonstruierte Szene des synthetischen Beispiels. Die Abbildungen **??** und **??** zeigen die auf Ursprugsgröße skalierte rekonstruierte Szene des Quaders.





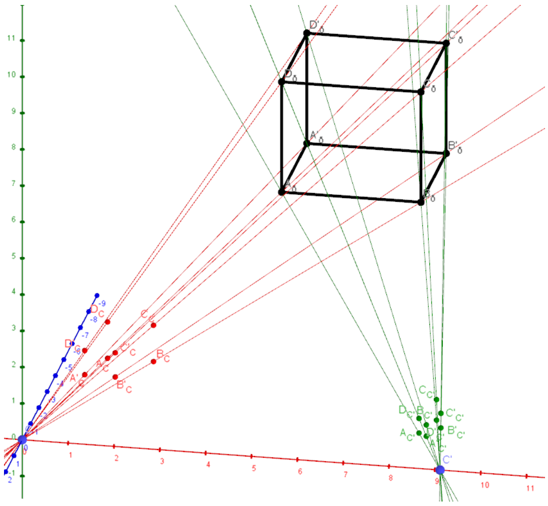


Abbildung 4.8: Veranschaulichung der Skaleninvarianz und dessen Auswirkung auf die geometrische Form und Größe der Objekte

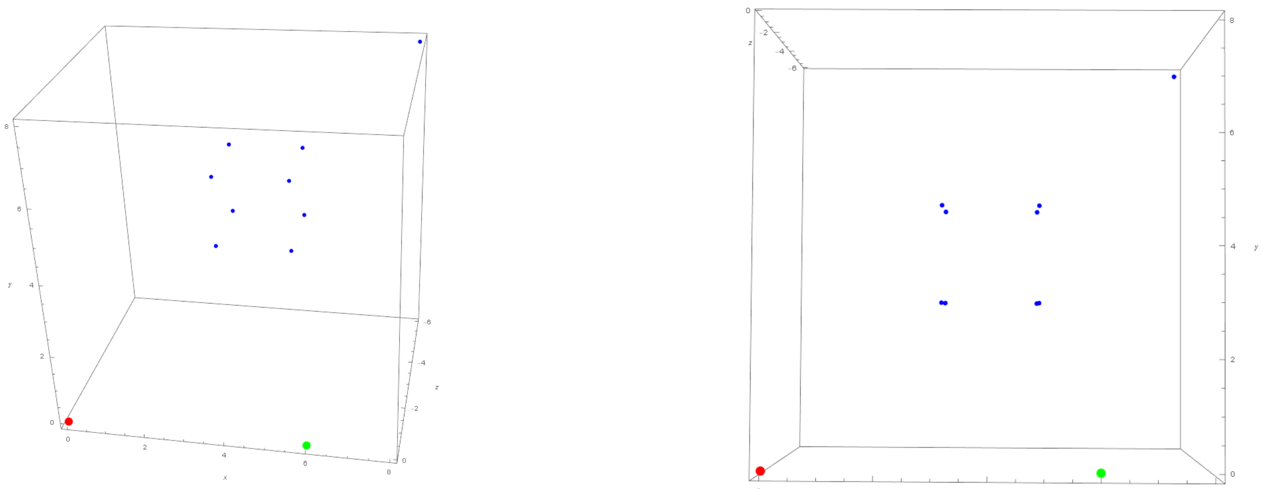


Abbildung 4.9: Der rote Punkt stellt die Postion von C dar, der grüne steht für die Position von C' relativ zu C. Die blauen Punkte stellen den rekonstruierten Quader und den extern platzierten neunten Punkt da. Die Abbildungen entstand aus dem in *Mathematica*[22] implementierten Algorithmus.

4.3 Auswirkungen von unterschiedlichen Kameraauflösungen

Der entstandene Szenenrekonstruktionsalforithmus wurde anhand von Kameras gleicher Auflösung implementiert und bereits validiert. Im folgenden soll nun getestet werden, ob der entwickelte Algorithmus auch im Stande ist für Kameras unterschiedlicher Auflösung eine Szene richtig zu Rekonstruieren. Zunächst wird beschrieben, welche Modifizierungen auf einem Sensor bei veränderung Der Auflösung stattfinden. Anschließend wird analysiert, wie sich die Auflösungsänderung auf das in Kapitel 2 beschriebene Kameramodell ändert und welche Einflüsse sie auf die in Kapitel 3 hergeleiteten Epipolaren Bedingungen und somit auf die Bestimmung der extrinsischen Kameraparameter und der folgenden Szenenrekonstruktion hat. Zum Schluss werden die Ergebnisse der synthetischen Rekonstruktion mit unterschiedlichen Kameraauflösungen präsentiert und validiert.

4.4 Geometrie eines Sensors

Die Geometrie eines Sensors kann als eine $M \times N$ - Matrix, bestehend aus $M \times N$ Sensorelementen dargestellt werden[8]. Die Auflösung eines Sensors hängt von den horizontalen und vertikalen Abständen der Sensorelemente ab. Abbildung 4.10 zeigt den schematischen Aufbau eines Sensors (CMOS).

Ein Sensor hat eine maximale Auflösung. Die maximale Anzahl der Sensorelemente auf einem Sensor beschreit die maximale Auflösung. Verschiedene Kameras können aus diesem Grund verschiedenen Auflösungen besitzen. Die Anzahl und Größe der Einzelnen Sensorelemente variiert mit den Größen der Sensorchips. Je größer ein Sensor ist und je mehr Sensorelemente er besitzt, desto besser ist die Bildqualität[8]. Bei maximaler Auflösung definiert genau ein Sensorelement einen Pixel. Ein Pixel wiederum entspricht einem Bildpunkt[8]. Auch ist es möglich die Auflösung eines Sensors digital zu verändern. Wird eine Auflöung kleiner der maximalen Auflösung eingesteltt, desto geringer wird die Anzahl der Pixel. Der Prozess, welcher hier stattfindet, gehört zu den Nachbarschaftsoperationen. Benachbarte Pixel werden hier zu einem neuen Pixel definiert[8]. Ein neuer Pixel wird aus den benachbarten Pixeln berechnet.

Eine Veränderung der Auflösung kann auch eine Änderung der Seitenverhältnisse mit einschließen. Ändert sich das Seitenverhältnis so wird der Bereich der lichtempfindlichen Fläche auf dem Sensor beschränkt[8]. Dies führt dazu, dass sich die Bildausschnnitte ändern. Abbildung 4.11 stellt schematisch da wie sich die lichtempfindlichen Bereiche auf dem Sensor bei unterschiedlichen Auflösungen ändert.
Eine Veränderung der Auflösung kann auch eine Änderung der Seitenverhältnisse mit einschließen. Ändert sich das Seitenverhältnis so wird der Bereich der lichtempfindlichen Fläche auf dem Sensor beschränkt[8]. Dies führt dazu, dass sich die Bildausschnnitte ändern. Abbildung 4.11 stellt schematisch da wie sich die lichtempfindlichen Bereiche auf dem Sensor bei unterschiedlichen Auflösungen ändert.

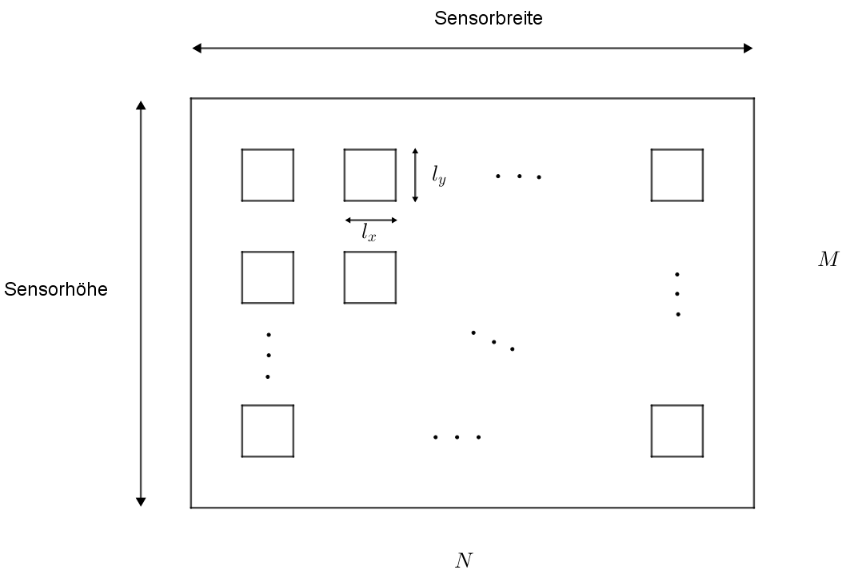


Abbildung 4.10: Rechteckiger Bildsensor mit darauf sich befindendenden quadratischen Sensorelementen. Vergleiche [8]

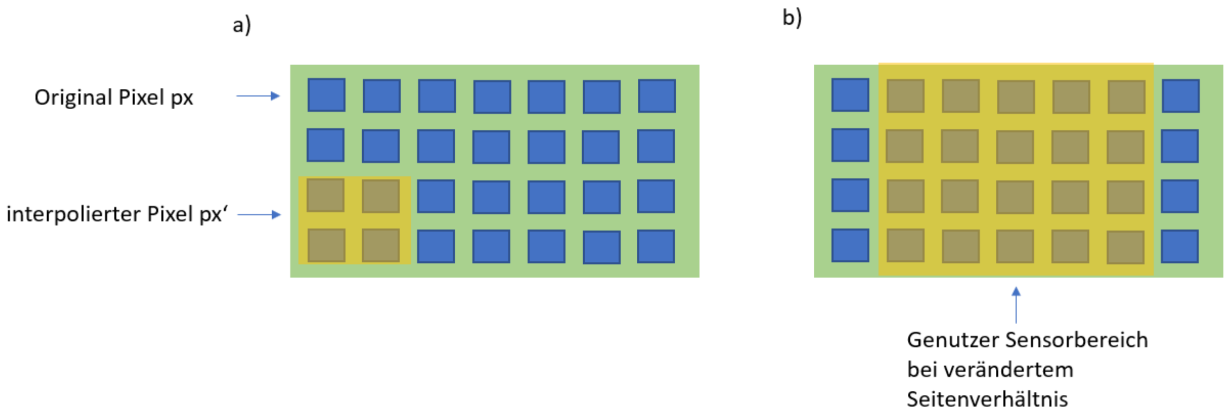


Abbildung 4.11: Bild a) zeigt die den Zusammenschluss mehrerer benachbarter Pixel zu einem neuen Pixel. Bild b) zeigt in gelb markiert, den aktiven lichtempfindlichen Bereich des Sensors, wenn sich das Seitenverhältnis geändert wird und nicht mehr der komplette Sensor genutzt wird.

4.5 Auswirkungen auf die Szenenrekonstruktion

Im folgenden wird zunächst analysiert, welche Änderungen sich im Lochkameramodell bei veränderter Auflösung ergeben und was für Auswirkungen diese auf die in Kapitel 3 aufgestellten Epipolaren Bedingungen hat.

Im Lochkameramodell hat eine Änderung der Kameraauflösung lediglich eine Auswirkung auf die Skalierung der Sensorkoordinatenachsen. Mit der Auflösung, ändern sich die Anzahl und die Größe der Pixel. Die Längeneinheiten des Sensorkoordinatensystems orientieren sich, wie in Kapitel 2 beschrieben, an genau diesen Längenskalierung der Pixelkanten l_x und l_y. Folglich kommt es zu einer Skalierung des Sensorkoordinatensystems.Alle anderen Koordinatensysteme bleiben unverändert. Durch Nachbarschaftsopterationen werden aus mehreren benachbarten Pixel ein neuer, jedoch bleibt der Ort des Pixel der gleiche[23]. Die Abbildungen 4.12 und 4.13 zeigen, dass sich zwar die Projektion von

Bildebenenkoordinatensystem auf das Sensorkoordinatensystem für den Punkt m_σ ändert,

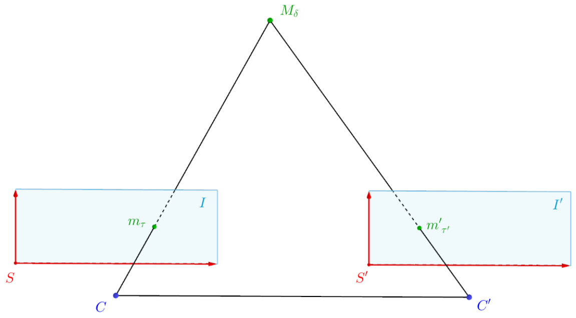


Abbildung 4.12: C und C' haben die selbe Auflösung eingestellt

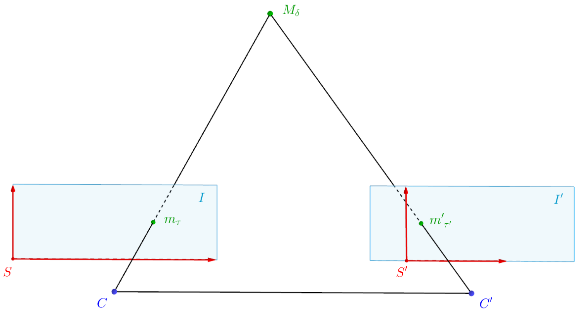


Abbildung 4.13: C und C' haben unterschiedliche Auflösungen eingestellt

Eine Skalierung der Sensorkoordinaten bedeutet, dass sich die Brennweite in Pixeleinheiten gegeben ändert, jedoch ändert sich nicht die effektive Brennweite in Millimeter. Anhand des Aufbau der Kameramatrix aus Kapitel 2 kann das nochmal verdeutlicht werden.

$$K = \begin{bmatrix} k_x\zeta & 0 & V_{\sigma,x} \\ 0 & k_y\zeta & V_{\sigma,y} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.39}$$

Wie bereits bekannt wird kommt es bei der Transformation von Bildebenenkoordinaten m_τ auf Sensorkoordinaten m_σ zu einer Skalierung der Bildebenenkoordinaten in Millimeter auf Sensorkoordinaten in Pixel. ζ_x und ζ_y stehen für die Brennweite. Durch die Multiplikation mit k_x und k_y wird die Brennweite auf Pixeleinheiten skaliert. Die ursprüngliche Brennweite beträgt $\zeta_x = \zeta_y = 1$. Kommt jetzt eine skalierung von $k_x = k_y = 2$ dazu.

$$K_0 = \begin{bmatrix} \zeta & 0 & V_{\tau,x} \\ 0 & \zeta & V_{\tau,y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & V_{\tau,x} \\ 0 & 1 & V_{\tau,y} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.40}$$

$$K = \begin{bmatrix} k_x\zeta & 0 & k_xV_{\tau,x} \\ 0 & k_y\zeta & k_yV_{\tau,y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2\cdot 1 & 0 & 2\cdot V_{\tau,x} \\ 0 & 2\cdot 1 & 2\cdot V_{\tau,y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & V_{\sigma,x} \\ 0 & 2 & V_{\sigma,y} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.41}$$

Die Veränderung der Kameraauflösung hat in diesem Beispiel zur Folge, dass es so wirkt, als wäre die Brennweite verdoppelt worden. Das würde bedeuten, dass sich die Kamera von der Bildebene entfernt hätte, jedoch verändert weder Kamera noch Bildebene ihre Position. Dennoch vergrößert oder verkleinert sich durch die Skalierung der Pixel effektiv die Bildgröße.
Um zu überprüfen. ob die Änderung der Kameraauflösung auch eine Änderung der Epipolaren Bedingungen mit sich führt, werden wieder die Gleichungen der Epipolaren Bedingungen in Kapitel 3 genauer betrachtet.

Die Fundamentalmatrix beinhaltet, sowohl die intrinsischen als auch die extrinsischen Parameter, um von F auf E zu kommen, müssen die intrinsischen Kameraparameter bekannt sein. Mit bekannten K und K' gilt, dass:

$$F = K'^{-T}R'\left[\vec{C}'_\delta - \vec{C}_\delta\right]_\times R^TK^{-1} \tag{4.42}$$

$$E = K'^TFK \tag{4.43}$$

$$E = K'^T(K'^{-T}R'\left[\vec{C}'_\delta - \vec{C}_\delta\right]_\times R^TK^{-1})K \tag{4.44}$$

$$\rightsquigarrow E = R'\left[\vec{C}'_\delta - \vec{C}_\delta\right]_\times R^T \tag{4.45}$$

Da bei der Bestimmung von E aus F die intrinsischen Kameraparameter eliminiert werden, haben unterschiedliche Auflösungen keine Auswirkung auf die essentielle Matrix. Folglich sollte das Ergebnis bei der Bestimmung der extrinsischen Kameraparameter unverändert sein. Um die Aufgestellte Theorie zu überprüfen, wurde im synthetischen Beispiel die Kameramatrix K' von C' modifiziert. Für C wurde $\zeta_x = \zeta_y = 1$ definiert, so das für Kameramatrix K gilt:

$$K = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.46}$$

Für die Kamermatrix K' von C' galt ursprünglich, dass $K = K'$. Die Auflösung von C' wird geändert indem die Skalierungsfaktoren k_x und k_y mit ζ'_x und ζ_y multipliziert werden.
Für das Beispiel wurden drei verschiedenen Auflösungen getestet. K'_1 mit $\zeta'_x \cdot 2$ und $\zeta'_y \cdot 2$, K'_2 mit $\zeta'_x \cdot 3.2$ und $\zeta'_y \cdot 1.2$ und K'_e mit $\zeta'_x \cdot 0.5$ und $\zeta'_y \cdot 4.3$. Da ursprünglich galt, dass $\zeta'_x = \zeta'_y = 1$ ergeben sich die folgenden Kameramatrizen für K'.

$$K'_1 = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.47}$$

$$K'_2 = \begin{bmatrix} 3.2 & 0 & 0 & 0 \\ 0 & 1.2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.48}$$

$$K'_3 = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 4.3 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.49}$$

Die Unterschiede der enstehenden Abbildungen in C' sind in den Abbildungen 4.14, 4.15,4.16 und 4.17 zu sehen.

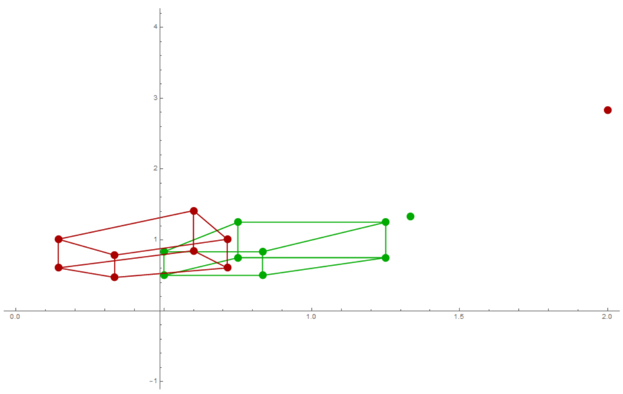


Abbildung 4.14: C und C' haben die selbe Auflösung eingestellt

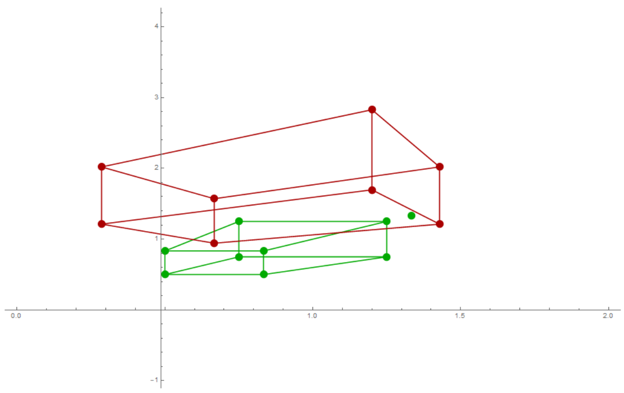


Abbildung 4.15: C und C' haben unterschiedliche Auflösungen eingestellt. C mit K und C' mit K'_1

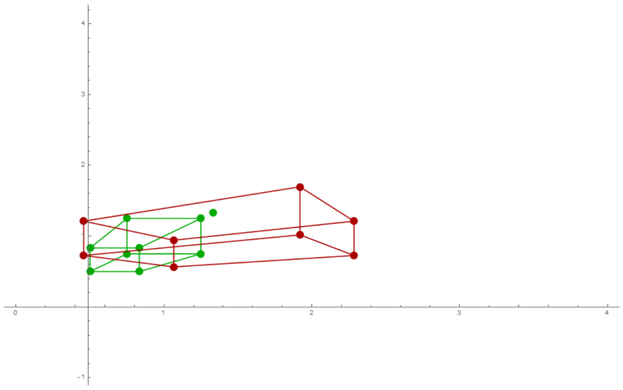


Abbildung 4.16: C mit K und C' mit K'_2

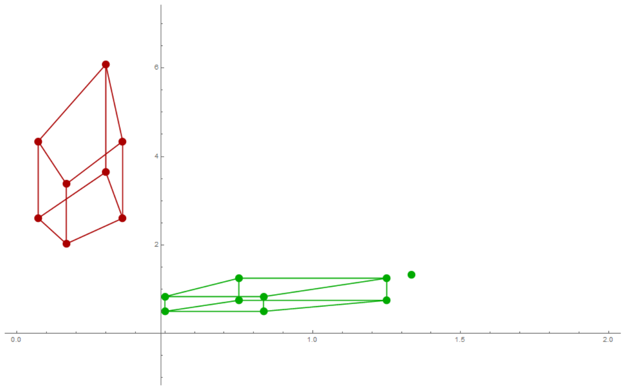


Abbildung 4.17: C mit K und C' mit K'_3

Wird das synthetische Beispiel jeweils mit den drei verschiedenen modifizierten K' durchgerechnet, so ergeben sich für die essentielle Matrix folgende Ergebnisse.

$$\zeta'_x = 1,\ \zeta'_y = 1: \quad E = \begin{pmatrix} 0 & -0.5 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \\ 0 & 0.5 & 0 \end{pmatrix} \mid : 0.5 \rightsquigarrow E = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -\sqrt{2} \\ 0 & 1 & 0 \end{pmatrix}$$

$$\zeta'_x = 2,\ \zeta'_y = 2: \quad E = \begin{pmatrix} 0 & 0.756 & 0 \\ 0 & 0 & 1.069 \\ 0 & -0.756 & 0 \end{pmatrix} \mid : -0.756 \rightsquigarrow E = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -\sqrt{2} \\ 0 & 1 & 0 \end{pmatrix}$$

$$\zeta'_x = 3.2,\ \zeta'_y = 1.2: \quad E = \begin{pmatrix} 0 & 0.634 & 0 \\ 0 & 0 & 1.069 \\ 0 & -0.634 & 0 \end{pmatrix} \mid : -0.634 \rightsquigarrow E = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -\sqrt{2} \\ 0 & 1 & 0 \end{pmatrix}$$

$$\zeta'_x = 0.5,\ \zeta'_y = 4.3: \quad E = \begin{pmatrix} 0 & 0.442 & 0 \\ 0 & 0 & 1.069 \\ 0 & -0.442 & 0 \end{pmatrix} \mid : -0.442 \rightsquigarrow E = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -\sqrt{2} \\ 0 & 1 & 0 \end{pmatrix}$$

Wie beobachtet werden kann, werden, trotz unterschiedlicher Kameraauflösungen für K', immer die selbe essentielle Matrix im Algorithmus bestimmt. Zu Erinnerung, in Kapitel 3 wurde gezeigt, dass jedes Vielfache von E ist eine gültige Lösung ist. Somit gilt die Behauptung, dass die Kameraauflösung keine Auswirkung auf die Bestimmung der extrinsischen Kameraparameter hat, im synthetischen Beispiel, als bestätigt. Als Vergleich kann die Abbildung 4.18, welche das Ergebnis der Rekonstruktion der Szene mit K'_3 als Kameramatrix für C' veranschaulicht, mit der Rekonstruierten Szene in Abbildung **??** und **??** aus dem ersten Beispiel, betrachtet werden. Für die anderen Varianten von K' wurden ebenfals die selbe 3D-Szene rekonstruiert.

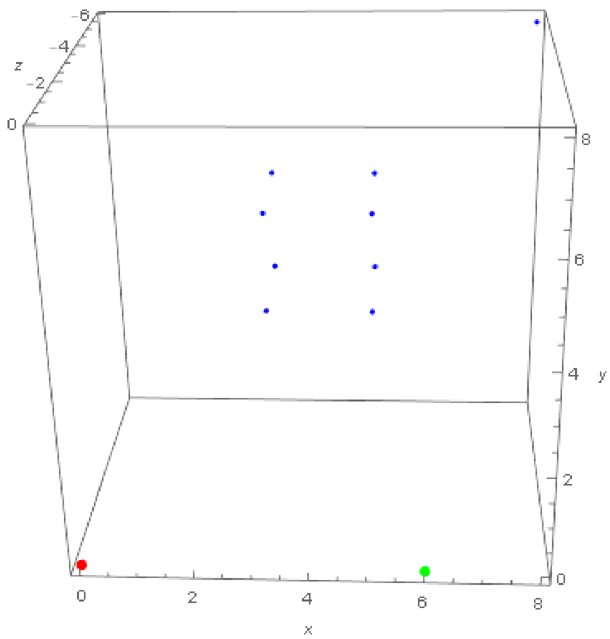


Abbildung 4.18: Die Abbildung zeigt die rekonstruierte Szenen des synthetischen Beispiels mit K_3' als intrisische Parameter für C'.

5 Reelle Rekonstruktion

Der entwickelte Szenenrekonstrukionsalgorithmus wird im folgenden anhand eines realen Stereoaufbaus getestet. Anders als bei den synthetischen Bilddaten im Beispiel zuvor, muss beim arbeiten mit reellen Stereobildern mit einer Fehleranfälligkeit der Bilddaten gerechnet werden. Liegt beispielsweise bei der Bestimmung von korrespondierenden Punkten eine Ecke zwischen zwei Pixel, so kann optisches Rauschen, welches in realen aufnahmen präsent ist, dazu führen dass die Ecke in gleichen Bildern an verschiedenen Pixel erkannt wird. Diese Abweichung der Punktekorrespondenz führt dazu, dass die in Kapitel 3.2 definierten epipolaren Bedingungen nicht mehr erfüllt werden. In diesem Kapitel wird Hauptsächlich darauf eingegangen, welche Auswirkungen diese Ungenauigkeiten auf die Bestimmung der Fundamentalmatrix, der essentiellen Matrix und der anschließenden Rekonstruktion der Szene durch Triangulierng haben und wie man ihnen entgegen wirken kann. In Abbildung 5.1, ist der Arbeitsprozess für den reellen Fall zu sehen. Der Ablauf im allgemeinen bleibt beständig.

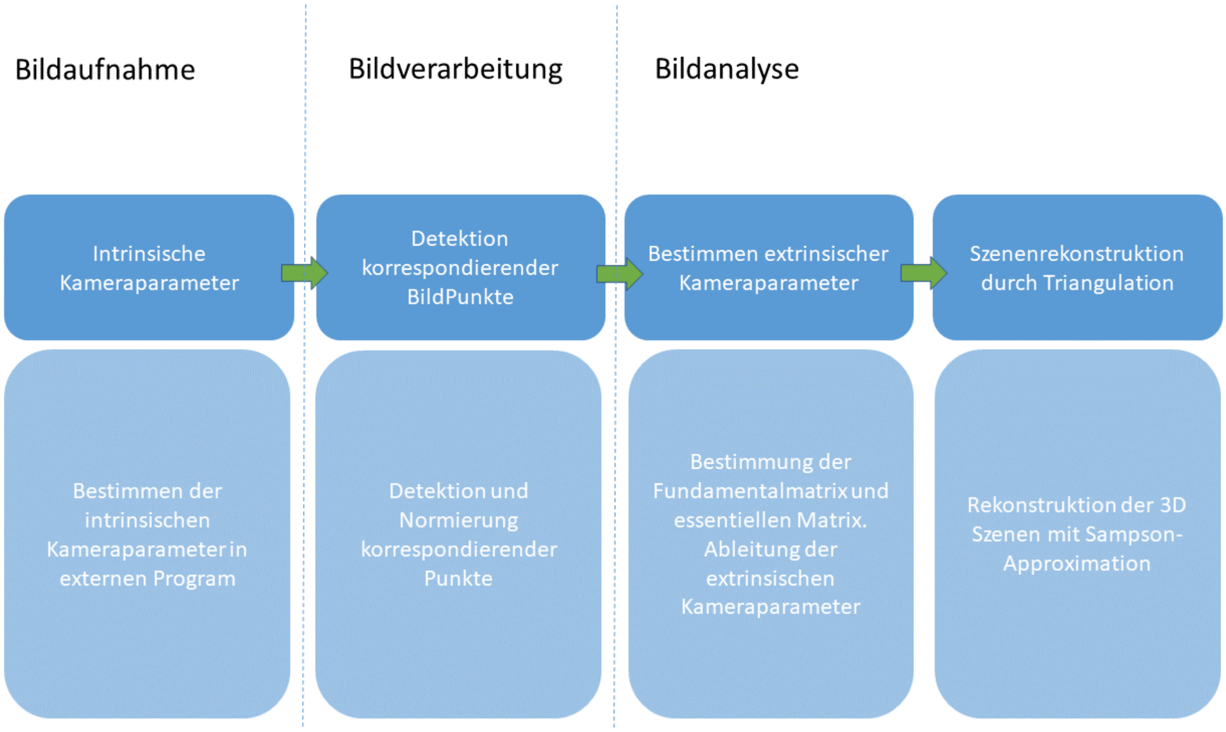


Abbildung 5.1: Ablaufdiagramm für die reelle Rekonstruktion

Zu Beginn wird der Stereoaufbau vorgestellt. Danach folgt die Korrespondenzanalyse, in welcher zwei Möglichkeiten aufgeführt werden, wie die Korrespondierenden Punkte aus den Bildern gewonnen werden können. Der Normierte-Acht-Punkt-Algorithmus, stellt eine für reale Bilddaten leicht veränderte Fassung des bereits bekannten Acht-Punkt-Algorithmus vor. Mit dessen Hilfe ist es möglich die Auswirkungen der Abweichungen in den Punktekorrespondenzen auf ein Minimum zu reduzieren. Nach der Bestimmung der Fundamentalmatrix durch den Normierten-Acht-Punkt-Algorithmus, wird diese auf ihre Gültigkeit kontrolliert. Hierzu werden vor allem die Singulärwerte der Fundamentalmatrix genauer betrachtet und welche Auswirkungen sie auf die Epipolargeometrie haben. Als nächstes wird die aus der Fundamentalmatrix bestimmte essentielle Matrix überprüft. Auch hier wird auf die Eigenschaften, welche die Singulärwerte der essentiellen Matrix haben müssen eingegangen. Danach wird eine Triangulierungsmethode nach [3] vorgestellt, die über ein Näherungsverfahren eine Triangulation zwischen ungenauen korrespondierenden Punkten ermöglicht. Der letzte Abschnitt des Kapitels befasst sich dann mit der reellen Rekonstruktion bei unterschiedlichen Kameraauflösungen.

5.1 Stereoaufbau

Für die Stereobildaufnahme, wurde eine Szene vor zwei nebeneinander platzierten Kameras aufgebaut. Beide Kameras wurden zur Szene gerichtet. Abbildung 5.2 zeigt den Stereoaufbau.

Abbildung 5.2: Kamera eins C befindet sich auf dem Bild links, Kamera zwei C' befindet sich rechts im Bild. Auf dem Tisch zwischen den Kameras ist die in den Abbildungen 5.4 und 5.3 abgebildete Szene zu sehen. Beide Kameras sind auf die Szene gerichtet.

Die auf Abbildung 5.2 zu sehende linke Kamera wurde als Kamera eins definiert. Das Kamerakoordinatensystem (C, β) wurde gleich dem Weltkoordinatensystem (O, δ) gesetzt. Kamera zwei mit (C', β') befindet sich auf der Abbildung rechts. Für beide Kameras wurden in einem externen Programm die intrinsischen Kameraparameter K und K' bestimmt. An den Stereoaufnahmen der Szene wurde dann eine Korrespondenzanalyse durchgeführt.

5.2 Korrespondenzanalyse

Für die Detektion von Punktekorrespondenzen bei Stereoaufnahmen einer dreidimensionalen Szene wurde eine existierende Funktion von Mathematica genutzt[22]. Die Funktion basiert auf dem Prinzip eines SURF-Algoruthmus. SURF ist die Kurzform für *Speeded Up Robust Features* und ist ein Rotations- und Skaleninvarianter Punkte Detektor und Deskriptor[17, 24]. Es werden Punkte an markanten Stellen in beiden Bildern detektiert, wie beispielsweise Eckpunkte oder Kanten. Die Umgebung eines jeden gefundenen Punktes wird durch einen Merkmalsvektor, dem Deskriptor, beschrieben. Die Deskriptoren beider Bilder werden abgeglichen und gleiche Punkte werden als korrespondierende Punkte gekennzeichnet[17, 24]. Die Abbildungen 5.4 und 5.3 zeigen die Ergebnisse nach der Anwendung des SURF-Algorithmus auf das Stereobildpaar. Eine eigens implementierte alternative für die Korrespondenzanalyse zwischen Stereoaufnahmen eines zweidimensionalen Schachbretts wird in Kapitel 7 vorgestellt.

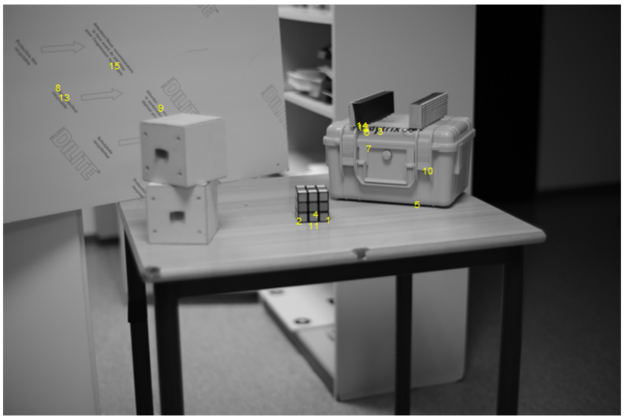


Abbildung 5.3: Aufnahme von Kamera C

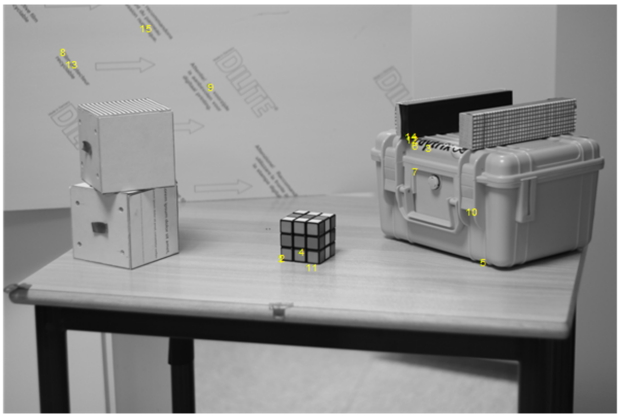


Abbildung 5.4: Aufnahme von Kamera C'

Die mit dem *SURF*-Algorithmus gefundenen Punkte sind mit den gelben Ziffern in den Abbildungen 5.4 und 5.3 gekennzeichnet. Abbildung 5.4 zeigt das Bild von C, die Abbildung 5.3 zeigt das Bild von C'.

Die Detektion von korrespondierenden Punkten mit Detektionsalgorithmen, wie Beispielsweise dem angewendeten SURF-Algorithmus, können immer Fehler und Abweichungen mit sich bringen. Die Ursprünge der Fehler können sowohl durch den Algorithmus als auch durch Fehler, wie Bildrauschen, in den Aufnahmen selbst entstehen. Diese Fehler wirken sich sowohl auf die Bestimmung der Abbildungsvorschriften F und E aus und somit auch auf die Genauigkeit der Szenenrekonstruktion[3]. Im folgenden werden sowohl die Fehler als auch Methoden für deren Minimierung vorgestellt.

5.3 Normierter acht-Punkt-Algorithmus

Trotz das der acht-Punkt-Algorithmus eine einfache Methode zur Bestimmung der Fundamentalmatrix bietet, ist er sehr instabil sobald Fehler wie Ungenauigkeiten in Punktekorrespondenzen auftreten[3, 25].

Die Ausmaße der Fehler lässt sich anhand der Kondition der Koeffizientenmatrix A genauer feststellen. Als Kondition wird die Abhängigkeit der Lösung eines Problems von der Störung der Eingangsdaten beschrieben[16, 26, 27]. Die Kondition lässt sich durch Bestimmung des kleinsten Eigenvektors der Matrixmultiplikation der Koeffizientenmatrix A mit ihrer Transponierten A^T herausfinden. Die Matrix AA^T wird in die Matrizen UDU^T zerlegt, wobei U eine orthogonale und D eine diagonale Matrix ist. Die Diagonaleinträge von D sind in einer nicht ansteigenden Reihenfolge, woraus resultiert, dass der kleinste Singulärwert von D mit der letzten Spalte von U korrespondiert und somit ist die letzte Spalte von U gleich dem kleinsten Eigenvektor von AA^T[16, 26]. Wird angenommen, dass AA^T eine 9 $\times$ 9- Matrix ist, so ergeben die Diagonaleinträge d_1/d_9 den Wert der Kondition. Je größer die Kondition ist, desto größer wirken sich schon kleinste Abweichungen der reinkommenden Bilddaten, auf die aus A bestimmten Matrix F aus.

Um die Kondition möglichst klein zu halten, werden die Bildkoordinaten beider Bilder normiert. Die in Literaturquellen, vorgeschlagene Normierung beinhaltet pro Bild eine Translation und Skalierung, so dass der Schwerpunkt aller Punktekorrespondenzen auf einem Bild im Ursprung des Sensorkoordinatensystems liegt und der durchschnittliche Abstand der Punkte zum Ursprung $\sqrt{2}$ beträgt[3, 4, 25].

Für die Normierung wird pro Bild eine Transformationsmatrix T und T' definiert. Die Matrizen beinhalten sowohl eine Skalierung als auch eine Translation. Die Bestimmung der Matrix T wird im Folgenden aufgezeigt. Zuerst wird der Schwerpunkt s mit $s = \begin{pmatrix} s_x \\ s_y \end{pmatrix}$ der Punktemenge p_n mit

$p_n = \begin{pmatrix} p_{nx} \\ p_{ny} \end{pmatrix}$ berechnet, indem der Mittelwert aller Punkte p_n berechnet wird.

$$\begin{pmatrix} s_x \\ s_y \end{pmatrix} = \frac{1}{n} \sum_{i=1}^{n} \begin{pmatrix} p_i x \\ p_i y \end{pmatrix} \tag{5.1}$$

Danach wird s in den Ursprung verschoben. Die Punkte x_n werden ebenfalls um den Wert von s verschoben $x'_n = x_n - s$. Der Mittelwert aus den um s verschobenen Punkten x'_n ergibt den neuen Schwerpunkt s_0 im Koordinatenursprung. Als nächstes wird die Distanz jedes Punktes von x'_n zu s_0 berechnet und der Mittelwert aller Distanzen, hier mit d bezeichnet, berechnet. Die Matrix T und T' haben dann die folgende Form:

$$T = \begin{pmatrix} \frac{\sqrt{2}}{d} & 0 & -s_x \\ 0 & \frac{\sqrt{2}}{d} & -s_y \\ 0 & 0 & 1 \end{pmatrix} \tag{5.2}$$

$$T' = \begin{pmatrix} \frac{\sqrt{2}}{d'} & 0 & -s'_x \\ 0 & \frac{\sqrt{2}}{d'} & -s'_y \\ 0 & 0 & 1 \end{pmatrix} \tag{5.3}$$

Die originalen Bildpunkte des Stereobildpaares, werden mit den Matrizen T und T' verrechnet. Mit den Normierten Bildkoordinaten wird dann wieder nach dem in Kapitel 3.3 beschriebenen acht-Punkte-Algorithmus eine Fundamentalmatrix $\hat{F}$ bestimmt[3, 16, 4, 25]. Nachdem $\hat{F}$ aus den normierten Koordinaten bestimmt wurde, wird sie mit T und T' wieder denormalisiert.

$$F = T'^T \hat{F} T \tag{5.4}$$

5.3.1 Singularität der Fundamentalmatrix

Die aus den Punktekorrespondenzen bestimmte Fundamentalmatrix muss eine singuläre Matrix mit Rang 2 sein[3, 1, 4]. Die singularität der Fundamentalmatrix sorgt zum einen dafür das ihr rechter und linker Kern jeweils eine eindeutige Lösung für den Epipol des jeweilgen Bildes ergibt und die Epipolarlinien, welche mit $l = Fm_s imga$ und $l' = F^T m'_{\sigma'}$ berechnet werden, auch alle durch diese Epipole verlaufen[3]. Durch Ungenauigkeiten in korrespondierenden Bildpunkten kann es dazu kommen, dass die aus dem Normierten-Acht-Punkt-Algorithmus bestimmte Fundamentalmatrix $\hat{F}$ in ihrem Rang steigt und somit keine singuläre Matrix mehr ist. Matrizen sind singulär, falls sie keine Inverse besitzen, was durch as verschwinden der Determinante nachgewiesen werden kann[28]. Die Determinanten von $\hat{F}$ muss Null sein, damit sie eine singuläre Matrix von Rang 2 ist.

Ist $\hat{F}$ durch den erhöhten Rang keine singuläre Matrix, so ergeben der linke und der Rechte Kern von $\hat{F}$ keine eindeutigen Lösungen mehr für e und e' und die Epipolarlinien in beiden Bildern gehen dem enstprechend auch nicht mehr durch genau einen Punkt, wie man in den Abbildungen 5.5 und 5.6 erkennen kann. Die Abbildungen bilden Epipolarlinien aus dem Stereobildpaar 5.3 und 5.4 ab. Die Ursache des erhöhten Ranges von $\hat{F}$ ist in ihren Singulärwerten von F zu finden.

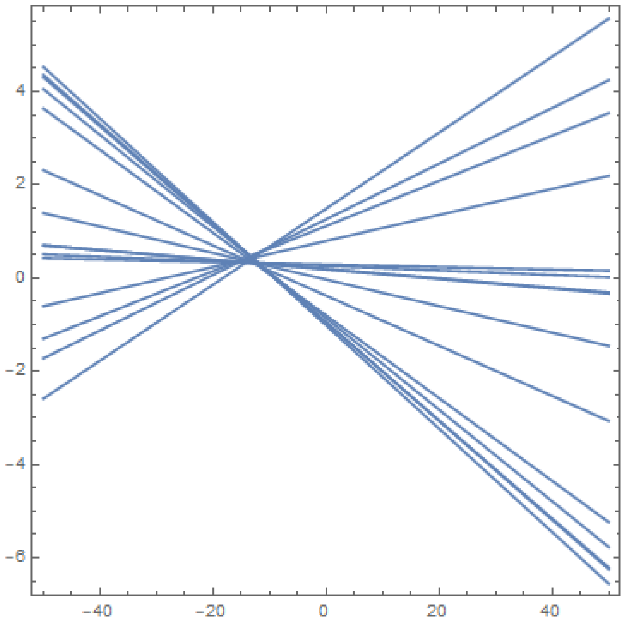


Abbildung 5.5: Epipolarlinien ohne *Epipolar-constraint* im Bild der Canon 6D

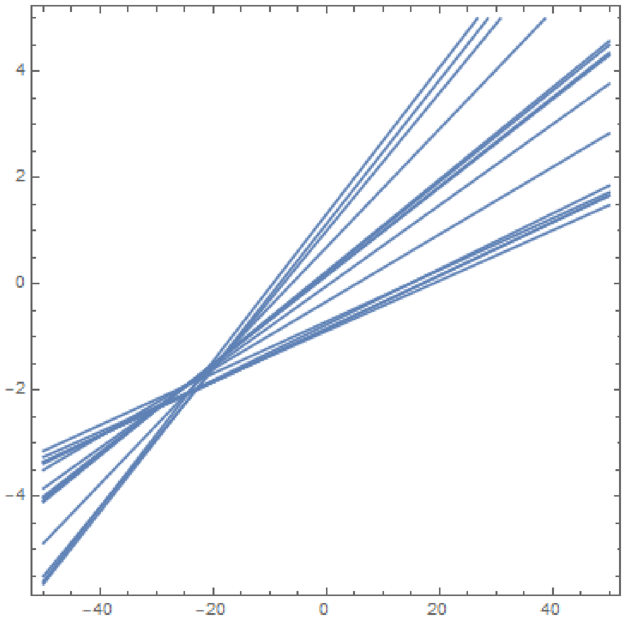


Abbildung 5.6: Epipolarlinien ohne *Epipolar-constraint* im Bild der Canon 60D

Um mit dem Algorithmus weiter verfahren zu können, muss die Singularität in der noch normierten Fundamentalmatrix $\hat{F}$ erzwungen werden[3]. Hierfür wird eine Singulärwertzerlegung an $\hat{F}$ durchgeführt, so dass $\hat{F}$ in $\hat{F} = U\Sigma V^T$ zerlegt wird. Σ beinhaltet in einer Diagonalmatrix die Singulärwerte $D = \text{diag}(r, s, t)$. Die Diagonaleinträge erfüllen die Bedingung, dass $r \geq s \geq t$ gilt. Damit $\hat{F}$ zu einer singulären Matrix wird, muss für die Diagonaleinträge jedoch gelten, dass $\Sigma = \text{diag}(r, s, 0)$ ist. Diese Bedingung wird erzwungen, indem der letzte Eintrag t auf $t = 0$ gesetzt wird. Die so modifizierte Fundamentalmatrix mit $\bar{F} = U\text{diag}(r, s, 0)V^T$ wird dann wieder zusammengesetzt. Die resultierende Fundamentalmatrix $\bar{F}$ besitzt jetzt einen Rang 2 und ist somit singulär[3]. $\bar{F}$ minimiert die Frobenius Norm $\parallel F - \bar{F} \parallel$ und ist somit die nächste zum ursprünglichen F liegende singuläre Matrix von Rang 2[3, 16, 28].

Werden aus $\bar{F}$ der rechte und linke Kern bestimmt, so ergeben sich eindeutige Lösungen für e und e' und die Epipolarlinien l und l' verlaufen jeweils durch ihre entsprechenden Epipole[3]. Die Abbildungen 5.7 und 5.8 zeigen die Auswirkung der erzwungenen Singularität von F auf dem Stereobildpaar 5.3 und 5.4. Die Abbildungen 5.9 und 5.10 die selben Epipolarlinien nur ist $\bar{F}$ mit T und T' denormalisiert worden.

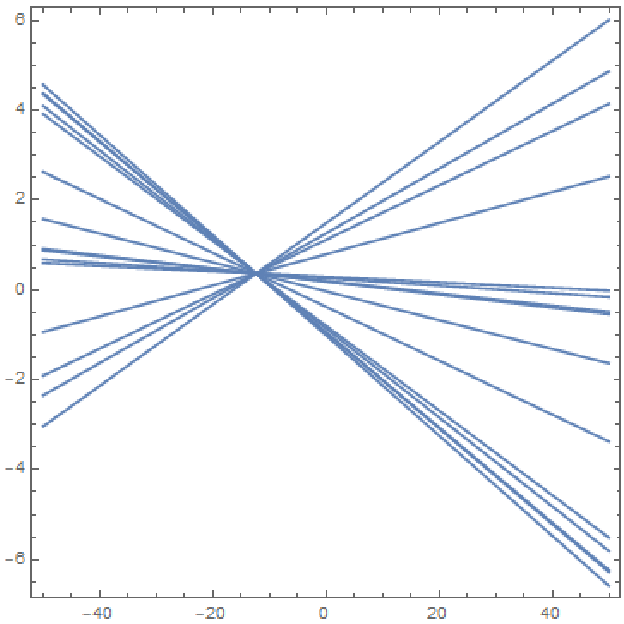


Abbildung 5.7: Die Abbildung zeigt, dass die Epipolarlinien auf der Aufnahme von C, nach dem Erzwingen der singularität in der normierten $\hat{F}$, alle durch den Epipol e verlaufen

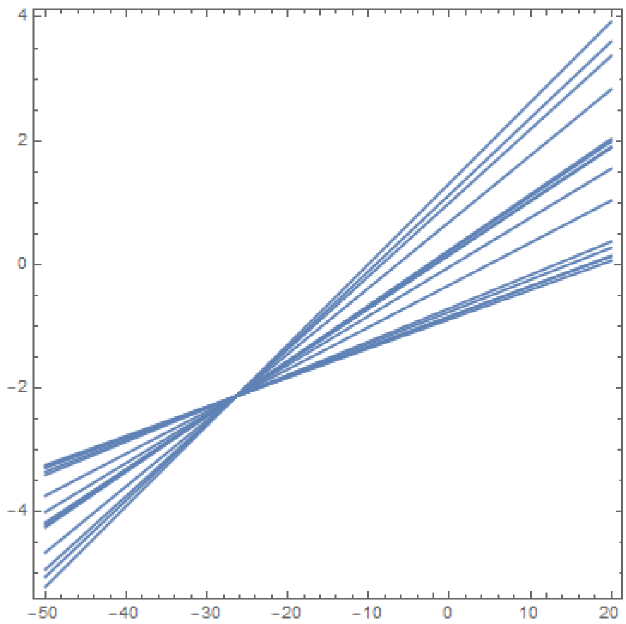


Abbildung 5.8: Die Abbildung zeigt, dass die Epipolarlinien auf der Aufnahme von C', nach dem Erzwingen der singularität in der normierten $\hat{F}$, alle durch den Epipol e' verlaufen

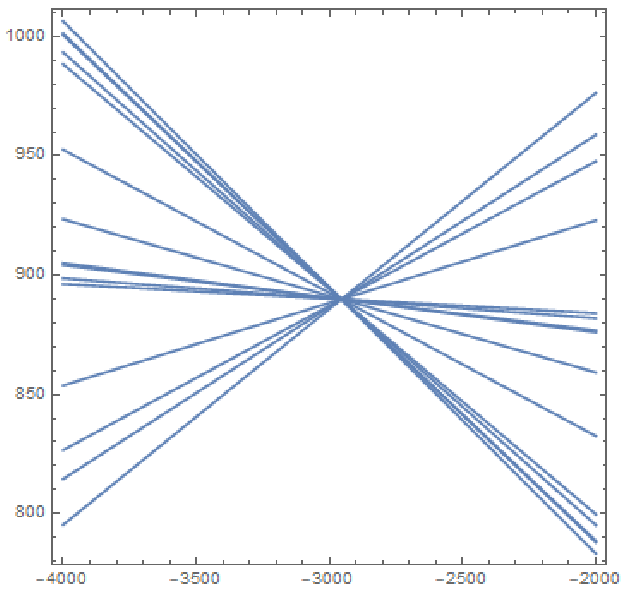


Abbildung 5.9: Die Abbildung zeigt die Epipolarlinien in C nachdem die Fundamentalmatrix $\bar{F}$ mit $F = T'\bar{F}T$ denormalisiert wurde

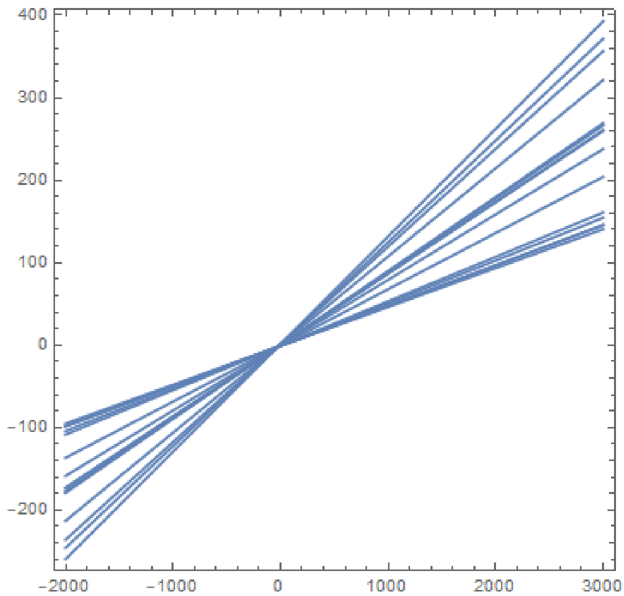


Abbildung 5.10: Die Abbildung zeigt die Epipolarlinien in C' nachdem die Fundamentalmatrix $\bar{F}$ mit $F = T'\bar{F}T$ denormalisiert wurde

5.3.2 Singulärwerte der essentiellen Matrix

Die essentielle Matrix wird im entwickelten Algorithmus aus der Fundamentalmatrix F bestimmt. Da zuvor die singularität von F erzwungen wurde, ist die essentielle Matrix ebenfalls eine Matrix von Rang 2[3]. Im synthetischen Beispiel wurde gezeigt, dass für die Bestimmung der extrinsischen Kameraparameter für E gelten muss, dass für ihre Singulärwerte $\Sigma = \mathrm{diag}(1, 1, 0)$ gelten muss.

Die Ungenauigkeit der Punktekorrespondenzen, kann auf E die Auswirkung haben, dass ihre Singulärwerte die Form $\Sigma = \mathrm{diag}(a, b, c)$ mit $a \geq b \geq c$ annehmen. Eine Matrix gilt nur dann als gültige essentielle Matrix, wenn zwei ihrer Singulärwerte gleich sind ($a = b$) und für den dritten gilt ($c = 0$). Um diese Bedingung zu erzwingen, wird diejenige essentielle Matirix $\hat{E}$ gesucht, welche sich laut der Frobenius Norm am nächsten an der Ursprünglichen E befindet[3, 4]. Diese Matrix lässt sich aus $E = U\Sigma V^T$ bestimmen, indem eine neue essentiellem Matrix $\hat{E}$ mit $\hat{E} = U\hat{\Sigma}V^T$ mit $\hat{\Sigma} = \mathrm{diag}(\frac{a+b}{2}, \frac{a+b}{2}, 0)$[3] bestimmt wird.

Nach erzwingen der Bedingung $\hat{E} = U\hat{\Sigma}V^T$ mit $\hat{\Sigma} = \mathrm{diag}(\frac{a+b}{2}, \frac{a+b}{2}, 0)$, ist E wieder eine gültige essentielle Matrix und der Algorithmus kann mit der Bestimmung der extrinsischen Kameraparameter, wie in Kapitel 4 gezeigt, fortfahren.

Das Erzwingen der singulären Bedingungen für F und E sind Näherungsverfahren, in welchen die Auswirkungen der Fehlerhaften Punktekorrespondenzen minimiert werden. Nur wenn F und E die ihre singulären Bedingungen erfüllen, können die extrinsichen Kameraparameter bestimmt und die Szene rekonstruiert werden[3].

5.4 Szenenrekonstruktion mit Sampson-Approximation

Aufgrund der Ungenauigkeit der korrespondierenden Punkte ist es nicht möglich die 3D-Objektpunkte durch einfache Rückprojektion der Bildpunkte zu rekonstruieren. Liegt der zu m_σ korrespondierende Bildpunkt $m'_{\sigma'}$ nicht ganz genau auf der zu m_σ korrespondierenden Epipolarlinie, so ist die in Kapitel 3 aufgestellte epipolare Bedingung aus Gleichung 3.18 nicht mehr erfüllt. Durch einsetzten der durch den SURF-Alggorithmus detektierten korrespondierenden Punkte m_σ und $m'_{\sigma'}$ in die Gleichung

$$m'^T_{\sigma'} F m_\sigma = 0 \tag{5.5}$$

kommt ein Wert $\neq 0$ heraus. Je weiter der Wert von 0 abweicht, desto ungenauer ist die korrespondenz beider Bildpunkte. Dies führt dazu, dass bei der Rückprojektion der Bildpunkte m_σ und $m'_{\sigma'}$ sich die Linien nicht im Raum treffen, sondern windschief zueinander liegen. Die Abbildungen 5.11 und 5.12 veranschaulichen eine Konsequenz von ungenauen Punktekorrespondenzen.

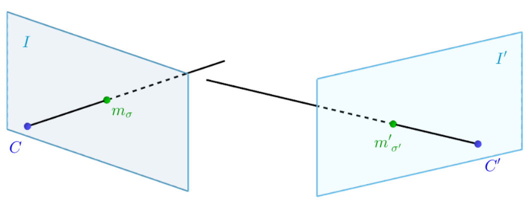


Abbildung 5.11: a) Windschiefe Geraden

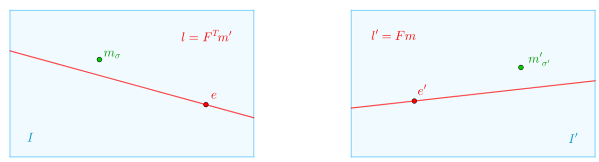


Abbildung 5.12: b) Epipolare Bedingungen werden nicht erfüllt

Abbildung 5.13: a) Die rückprojizierten Strahlen der ungenauen korrespondierenden Punkte m_σ und $m'_{\sigma'}$ sind Windschief zueinander und treffen sich nicht in einem Punkt $M_{\delta,0}$ im Raum. b) Die korrespondierenden Bildpunkte m_σ und $m'_{\sigma'}$ erfüllen nicht die Epipolaren Bedingungen. Die Epipolarlinie $l' = Fm$ ist die korrespondierende Epipolarlinie zu m_σ und $l = F^T m'$ ist die korrespondierende Epipolarlinie zu m'_σ. Da weder m_σ noch $m'_{\sigma'}$ auf der Epipolarlinie zum jeweils korrespondierenden Punkt liegen, kommt es zu keinem Schnittpunkt der rückprojizierten Strahlen

Um trotz der ungenauen korrespondierenden Punkte eine Triangulation zu ermöglichen, wurde ein Verfahren voran geschaltet, welches zwei Punkte $\hat{m}_\sigma$ und $\hat{m}'_{\sigma'}$ sucht, die möglichst nah an den ursprünglichen Punkten m_σ und $m'_{\sigma'}$ liegen und gleichzeitig die epipolare Bedingung $\hat{m}'^T_{\sigma'} F \hat{m}_\sigma = 0$ erfüllt. $\hat{m}_\sigma$ und $\hat{m}'_{\sigma'}$ sollen durch Minimierung einer Funktion C bestimmt werden, welche die Distanz d zwischen m_σ und $\hat{m}_\sigma$ und $m'_{\sigma'}$ und $\hat{m}'_{\sigma'}$ minimiert. Für die Minimierung wird das Verfahren der Sampson-Approximation gewählt[3]. Voraussetzung für die Triangulierung ist, dass die Projektionsmatrizen P und P', sowie die Fundamentalmatrix F bekannt sein müssen[3].

$$C(m, m') = d(m, \hat{m})^2 + d(m', \hat{m'})^2 \tag{5.6}$$

Die optimalen $\hat{m}$ und $\hat{m}'$ liegen auf den korrespondierenden Epipolarlinien $\hat{l}$ und $\hat{l}'$ am Fuße des Lots, welches von den ursprünglich projizierten Punkten m und m' auf die Epipolarlinien $\hat{l}$ und $\hat{l}'$ gefällt wird[3].

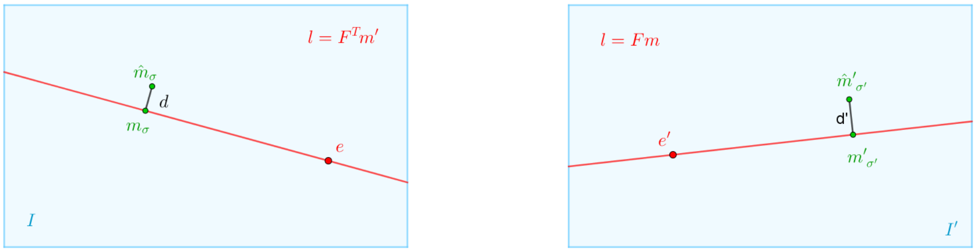


Abbildung 5.14: Die Abbildung zeigt die zwei korrespondierenden Epipolarlinien $\hat{l}$ und $\hat{l}'$ mit den gesuchten Punkten $\hat{m}_\sigma$ und $\hat{m'}_{\sigma'}$

Jedes andere Punktepaar auf $\hat{l}$ und $\hat{l}'$ würde die epipolare Bedingung erfüllen, jedoch minimieren nur $m_{\sigma\perp}$ und $m'_{\sigma'\perp}$ die quadratischen Distanzen $d(m_\sigma, \hat{m}_\sigma)^2$ und $d(m'_{\sigma'}, \hat{m}'_{\sigma'})^2$ in der Funktion C. Gesucht wird also der geringste Abstand von m_σ zu $\hat{l}$ und $m'_{\sigma'}$ zu $\hat{l}'$. Die Funktion C kann dem entsprechend umformuliert werden in

$$C(m, m') = d(\hat{m}_\sigma, \hat{l})^2 + d(\hat{m}'_{\sigma'}, \hat{l}')^2 \tag{5.7}$$

.
Aus allen möglichen Epipolarlinien, welche $\hat{l}$ und $\hat{l}'$ annehmen können, entspricht immer der senkrechte Abstand von m_σ und $m'_{\sigma'}$ zur jeweiligen Epipolarlinie der minimalen Distanz. Es soll jedoch genau das Epipolalinienpaar gewählt werden, welches die Funktion C minimal werden lässt[3].

Im ersten Schritt werden die Epipolarlinien Parametrisiert, so dass eine Linie als $\hat{l}(t)$ geschrieben werden kann. Durch die Parametrisierung der Epipolarlinien, kann die Funktion C als Funktion von t umformuliert werden.

$$C(m_\sigma, m'_{\sigma'}) = d(m_\sigma, \hat{l}(t))^2 + d(m', \hat{l}'(t))^2 \tag{5.8}$$

Um die Minimierung zu vereinfachen, werden zu Beginn die Bildpunkte m_σ und $m'_{\sigma'}$ mit jeweils einer Matrix T und T' in den Ursprung $(0, 0, 1)^T$ verschoben.

$$T = \begin{bmatrix} 1 & 0 & -m_{x\sigma} \\ 0 & 1 & -m_{y\sigma'} \\ 0 & 0 & 1 \end{bmatrix} \rightsquigarrow \bar{m}_{\tau} = T \cdot m_{\tau} \tag{5.9}$$

$$T' = \begin{bmatrix} 1 & 0 & -m'_{x\sigma'} \\ 0 & 1 & -m'_{y\sigma'} \\ 0 & 0 & 1 \end{bmatrix} \rightsquigarrow \bar{m}'_{\tau'} = T' \cdot m'_{\tau'} \tag{5.10}$$

Die Fundamentalmatrix F wird ebenfalls mit T und T' transformiert, sodass sie an die verschobenen Punkte $\bar{m}_{\sigma}$ und $\bar{m}'_{\sigma'}$ angepasst ist.

$$\bar{F} = T'^{-T} F T^{-1} \tag{5.11}$$

Der rechte und linke Kern von $\bar{F}$ ergeben die Epipole $\bar{e}$ und $\bar{e}'$. Angenommen f und f' seinen genau 0, so liegen die Epipole $e = (1, 0, f)^T$ und $e' = (1, 0, f')^T$ im unendlichen. Ist dies der Fall so hat $\bar{F}$ für welche dann gilt, dass $\bar{F}(1, 0, f)^T = (1, 0, f')\bar{F} = 0$ eine spezielle Form[3].

$$\bar{F} = \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix} \tag{5.12}$$

$$\begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ f \end{pmatrix} = \begin{pmatrix} ff'd + (-ff'd) \\ -fb + fb \\ -fd + fd \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.13}$$

$$\begin{pmatrix} 1 & 0 & f' \end{pmatrix} \cdot \begin{pmatrix} ff'd & -f'c & -f'd \\ -fb & a & b \\ -fd & c & d \end{pmatrix} = \begin{pmatrix} ff'd + (-ff'd) \\ -f'c + f'c \\ -f'd + f'd \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} \tag{5.14}$$

Im Realfall sind die Werte der Epipole e und e' nicht genau $e = (1, 0, f)^T$ und $e' = (1, 0, f')^T$, sondern weichen in ihren Richtungen leicht ab. Des Weiteren sind entprechen die $xy-$Komponenten der Epipole noch keinem normierten Richtungsvektor. Also werden sie zunächst einmal so normiert, das $e_1^2 + e_2^2 = 1$ und $e_1'^2 + e_2'^2 = 1$ gilt. Danach werden e und e' mit zwei Rotationsmatrizen R und R' auf $Re = (1, 0, e_3) = (1, 0, f)$ und $R'e' = (1, 0, e_3') = (1, 0, f')$ rotiert.

$$R = \begin{bmatrix} e_1 & e_2 & 0 \\ -e_2 & e_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.15}$$

$$R' = \begin{bmatrix} e_1' & e_2' & 0 \\ -e_2' & e_1' & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.16}$$

$\bar{F}$ wird dann mit $\bar{F}_{Rot} = R' F R^T$ ersetzt. Die Einträge in $\bar{F}_{Rot}$ haben nun die Form wie in Gleichung 5.12, mit $f = e_3$, $f' = e_3'$, $a = \bar{F}_{Rot,22}$, $b = \bar{F}_{Rot,23}$, $c = \bar{F}_{Rot,32}$ und $d = \bar{F}_{Rot,33}$.

$$\bar{F}_{Rot} = \begin{pmatrix} e_3 e_3' \bar{F}_{Rot,33} & -e_3' \bar{F}_{Rot,32} & -e_3' \bar{F}_{Rot,33} \\ -e_3 \bar{F}_{Rot,23} & \bar{F}_{Rot,22} & \bar{F}_{Rot,23} \\ -e_3 \bar{F}_{Rot,33} & \bar{F}_{Rot,32} & \bar{F}_{Rot,33} \end{pmatrix} \tag{5.17}$$

$$\tag{5.18}$$

Verläuft eine Epipolarlinie durch einen Punkt $(0, t, 1)^T$ und dem Epipol $e = (1, 0, f)^T$, wird diese Epipolarlinie mit $l(t)$ bezeichnet. Das Kreuzprodukt dieser beiden Punkte beschreibt die Epipolarlinie $l(t)$[3].

$$\hat{l}(t) = \begin{pmatrix} 0 \\ t \\ 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \\ f \end{pmatrix} = \begin{pmatrix} tf \\ 1 \\ -t \end{pmatrix} \tag{5.19}$$

Die quadratische Distanz dieser Linie zum Ursprung wird dann bezeichnet mit:

$$d(\bar{m_\sigma}, \hat{l}(t))^2 = \frac{t^2}{1 + (tf)^2} \tag{5.20}$$

Laut *Hartley & Zisserman* [3] werden Linien durch Vektoren der Form $\vec{v} = \begin{pmatrix} A \\ B \\ C \end{pmatrix}$ dargestellt. Übersetzt in eine Koordinatengleichung ergibt sich

$$Ax + By + C = 0 \tag{5.21}$$

.

Die Gleichung 5.20 wird anhand der Koordinatengleichung hergeleitet. Für die Herleitung wird die Koordinatenform der Geraden zunächst in Normalform umgeschrieben

$$\vec{n} \cdot (\vec{x} - \vec{p}) = 0 \tag{5.22}$$

$$\begin{pmatrix} A \\ B \end{pmatrix} \cdot (\vec{x} - \begin{pmatrix} 0 \\ \frac{C}{B} \end{pmatrix}) = 0 \tag{5.23}$$

.

Der Abstand $\parallel \vec{v} \parallel$ eines Punktes zur Geraden 5.23 kann folgendermaßen berechnet werden.

$$\vec{v} = \frac{\vec{p} \cdot \vec{n}}{\vec{n} \cdot \vec{n}} \cdot \vec{n} \rightsquigarrow \frac{-C}{A^2 + B^2} \cdot \begin{pmatrix} A \\ B \end{pmatrix} \tag{5.24}$$

$$||\vec{v}|| = \frac{|\vec{p} \cdot \vec{n}|}{||\vec{n}||^2} \cdot ||\vec{n}|| \rightsquigarrow ||\vec{v}|| = \frac{|\vec{p} \cdot \vec{n}|}{||\vec{n}||} \tag{5.25}$$

$$\Rightarrow |C| = |\vec{p} \cdot \vec{n}| \tag{5.26}$$

$$\Rightarrow |\sqrt{A^2 + B^2}| = ||\vec{n}|| \tag{5.27}$$

$$||\vec{v}|| = \frac{|C|}{\sqrt{A^2 + B^2}} \tag{5.28}$$

Werden nun A, B und C mit den Werten der Geraden $(tf, 1, -t)^T$ ersetzt, kann Gleichung 5.20 rekonstruiert werden.

$$A = tf,\ B = 1,\ C = -t,\ \vec{v} = d \tag{5.29}$$

$$d^2 = \frac{t^2}{\sqrt{((tf)^2 + 1^2)^2}} = \frac{t^2}{(tf)^2 + 1^2} = \frac{t^2}{1 + (tf)^2} \tag{5.30}$$

Um die zu $\hat{l}(t)$ korresponiderende Epipolarlinie $\hat{l}'(t)$ zu bestimmen, wird der Beispielpunkt $(0, t, 1)^T$ und die Fundamentalmatrix $\bar{F}_{Rot}$ multipliziert. Für die Übersichtlichkeit der Matrix $\bar{F}_{Rot}$, wird die

Symbolschreibweise aus Gleichung 5.12 genutzt. Für die Einträge gelten die Definitionen aus Gleichung 5.17.

$$l'(t) = \bar{F}_{Rot}(0,t,1)^T = (-f'(ct+d), at+b, ct+d)^T. \tag{5.31}$$

Für die quadratische Distanz $d(m', l'(t))^2$ ergibt sich dann:

$$d(\bar{m_\sigma}, \hat{l}(t))^2 = \frac{(ct+d)^2}{(at+b)^2 + f'^2(ct+d)^2} \tag{5.32}$$

Die Kostenfunktion C kann jetzt in eine Funktion $s(t)$ umformuliert werden.

$$s(t) = \frac{t^2}{1+(tf)^2} + \frac{(ct+d)^2}{(at+b)^2 + f'^2(ct+d)^2} \tag{5.33}$$

Ein Minimum für $s(t)$ kann beispielsweise durch Bestimmung der Minima und Maxima mit $s'(t) = 0$ gefunden werden.

$$s'(t) = \frac{2t}{(1+f^2t^2)^2} - \frac{2(ad-bc)(at+b)(ct+t)}{((at+b)^2 + f'^2(ct+d)^2)^2} \tag{5.34}$$

Werden die beiden Terme in $s'(t)$ auf einen gemeinsamen Nenner gebracht und der Zähler dann gleich Null gesetzt, ergibt sich der folgende Ausdruck $g(t)$[3]

$$g(t) = t((at+b)^2 + f'^2(ct+d)^2)^2 - (ad-bc)(1+f^2t^2)^2(at+b)(ct+d) \tag{5.35}$$

.

Funktion $g(t)$ ist ein Polynom vom Grad 6. Das Minimum für $s(t)$ ergibt sich aus einer der sechs möglichen Lösungen für t aus $g(t)$. Für die Bestimmung des Minimums werden nur die reellen Lösungen in Betracht gezogen. Die reellen Lösungen für t aus $g(t)$, werden dann wieder in $s(t)$ eingesetzt. Das t, welches durch einsetzte in $s(t)$ den kleinsten Wert ergibt, ist das gesuchte Minimum t_{min}.

Mit t_{min} können die Epipolarlinien $\hat{l}(t_{min}) = (t_{min}f, 1, -t)$ und $\hat{l}'(t_{min}) = F(0, t_{min}, 1)^T$ berechnet werden. Danach werden die zwei neuen Punkte $\hat{m}_{\sigma Rot}$ und $\hat{m}'_{\sigma' Rot}$ auf den Epipolarlinien $\hat{l}(t_{min}$ und $\hat{l}'(t_{min})$ bestimmt. $\hat{m}_{\sigma Rot}$ und $\hat{m}'_{\sigma' Rot}$ sind die Punkte auf der Epipolarlinie welche dem Ursprung am nächsten sind. Zur Erinnerung die Bildpunkte m_σ und $m'_{\sigma'}$ wurden zu Beginn in den Ursprung verschoben. Der Punkt, welcher vom Ursprung aus am nächsten auf einer Linien (λ, μ, υ) liegt, kann mit $(-\lambda \cdot \upsilon, -\mu \cdot \upsilon, \lambda^2 + \mu^2)$ berechnet werden[3].

$$\hat{l} = (tf, 1, -t) \ \hat{m}_{\sigma Rot} = (-(tf) \cdot \upsilon, -1 \cdot \upsilon, (tf)^2 \cdot 1^2) \tag{5.36}$$

$$\tag{5.37}$$

Nachdem zu beiden Linien $\hat{l}$ und $\hat{l}'$ der jeweils nächste Punkte $\hat{m}_{Rot}$ und $\hat{m}'_{Rot}$ vom Ursprung aus gefunden wurden, werden diese nun mit T, T', R und R' wieder an ihre Ausgangsposition zurück Transformiert.

$$\hat{m} = T^{-1} R^T \hat{m}_{Rot} \tag{5.38}$$

$$\hat{m}' = T'^{-1} R'^T \hat{m}'_{Rot} \tag{5.39}$$

Für diese neu berechneten korrespondierenden Punkte ist die Epipolare Bedingung aus Gleichung 3.18 erfüllt und es ist gewährleistest, dass sich ihre jeweiligen Rückprojektionen in einem Punkt im Raum treffen.

Für die Rückprojektion der einzelnen Bildpunkte wurde ein lineares Triangulationsverfahren gewählt[3]. Dieses ist für mehrere Punkte rechentechnisch günstiger als das geometrische Verfahren, welches im synthetischen Beispiel in Kapitel 4 genutzt wurde.

Für die Rückprojektion werden pro korrespondierendem Punktepaar zunächst die Projektionsgleichungen $\hat{m}_\sigma = P\hat{M}_\delta$ und $\hat{m}'_{\sigma'} = P'\hat{M}_\delta$ aufgestellt. Diese werden so in eine Koeffizientenmatrix A eingetragen dass gilt $A \cdot x = 0$. Durch die Verwendung des Kreuzproduktes, wird die Homogene Komponente eliminiert[3].

$$\hat{m}_\sigma \times (PM_\delta) = 0 \tag{5.40}$$
$$\hat{m}'_{\sigma'} \times (PM_\delta) = 0 \tag{5.41}$$

Was ausgeschrieben für $\hat{m}$ und $\hat{m}'$ zu den folgenden drei Gleichungen führt. $\hat{m}_\sigma \times (PM_\delta) = 0$ ergibt ausgeschrieben

$$m_{\sigma\, x}(p^{3T} M_\delta) - (p^{1T} M_\delta) = 0 \tag{5.42}$$
$$m_{\sigma\, y}(p^{3T} M_\delta) - (p^{2T} M_\delta) = 0 \tag{5.43}$$
$$m_{\sigma\, x}(p^{2T} M_\delta) - m_{\sigma\, y}(p^{1T} M_\delta) = 0 \tag{5.44}$$

und Für $\hat{m}'_{\sigma'} \times (PM_\delta) = 0$ gelten

$$m'_{\sigma'\, x}(p^{3T} M_\delta) - (p^{1T} M_\delta) = 0 \tag{5.45}$$
$$m'_{\sigma'\, y}(p^{3T} M_\delta) - (p^{2T} M_\delta) = 0 \tag{5.46}$$
$$m'_{\sigma'\, x}(p^{2T} M_\delta) - m'_{\sigma'\, y}(p^{1T} M_\delta) = 0 \tag{5.47}$$

p^{iT} bezeichnet hier jeweils die Reihen der Projektionsmatrix P beziehungsweise P'. Zwei der drei Gleichungen sind linear unabhängig und werden in die Koeffizietenmatrix A geschrieben

$$A = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix} \tag{5.48}$$

$$A = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix} \cdot \begin{pmatrix} X_1 \end{pmatrix} \tag{5.49}$$

Die zwei Wege eine solche Matrix zu lösen wurden in Kapitel 3 vorgestellt. Zum einen kann die inhomogene Methode angewandt werden und der Kern dieser Koeffizientenmatrix bestimmt werden, oder es kann das homogene Verfahren angewandt werden, welches die Methode der Singulärwertzerlegung beinhaltet.

Die rekonstruierten Punkte $M_{\delta 0}$ sind wie in Kapitel 4 auch nur bis auf einen skalierungsfaktor genau bestimmt. Die Abbildungen 5.15 und 5.15 zeigen die aus den Bildern 5.4 und 5.3 rekonstruierten

Punkte im Raum. Der rote Punkt steht für die Position von C, der grüne für die Position von C'. Die blauen Punkte sind die rekonstruierten Punkte, der beiden Bilder 5.4 und 5.3. Somit wurde gezeigt, dass der Algorithmus, welcher für die synthetische Rekonstruktion entwickelt wurde mit gewissen Modifizierungen auch für ein reelles Stereobildpaar angewandt werden kann. Durch die Näherungen, sind gewissen Abweichungen von der Originalszene nicht zu vermeiden.

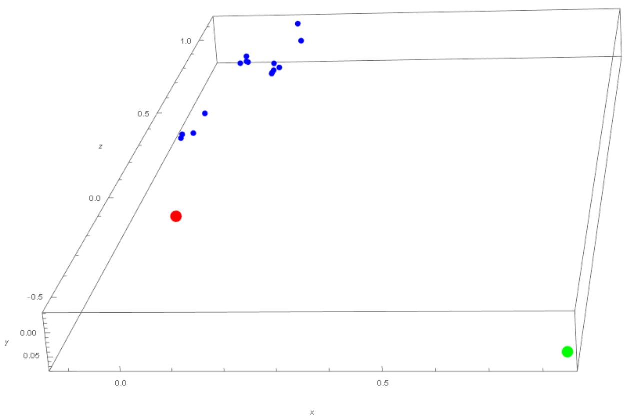


Abbildung 5.15: Rekonstruierte Szene, unskaliert in Pixeleinheiten

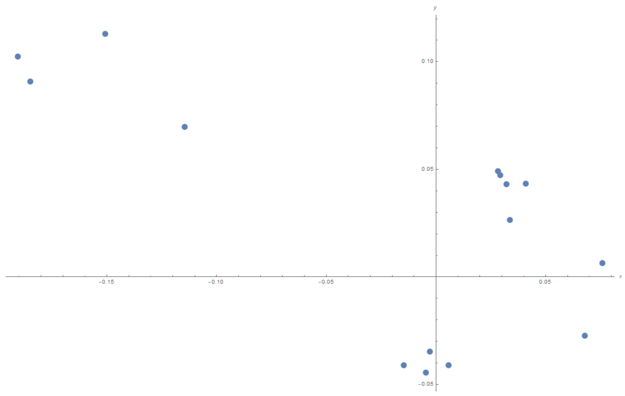


Abbildung 5.16: Rekonstruierte Szene, unskaliert, in Pixeleinheiten und in einem 2D-Plot angezeigt

5.5 Ergebnisse einer Stereoanalyse mit Kameras unterschiedlicher Auflösung

Für den Test, ob Szenerekonstruktion im Realbeispiel auch mit unterschiedlichen Kameraauflösungen funktioniert, wurde die Kameramatrix K' von C' künstlich skaliert. Wie aus Kapitel 2, hat diese die Form

$$K'_{[1:1]} = \begin{bmatrix} k_x\zeta & s & V_{\sigma,x} \\ 0 & k_y\zeta & V_{\sigma,y} \\ 0 & 0 & 1 \end{bmatrix} \tag{5.50}$$

.
Um die Auflösung von K' zu verändern, werden die Matrixeinträge $k_x\zeta$ und $k_y\zeta$ jeweils noch um eine beliebige Skalierung erweitert. Als Beispiel wurde K' drei mal unterschiedlich skaliert und zwar mit den Verhältnissen [5 : 2], [1 : 2] und [1.2 : 2.3].

$$K'_{[5:2]} = \begin{bmatrix} k_x\zeta \cdot 5 & s & V_{\sigma,x} \cdot 5 \\ 0 & k_y\zeta \cdot 2 & V_{\sigma,y} \cdot 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$K'_{[1:2]} = \begin{bmatrix} k_x\zeta \cdot 1 & s & V_{\sigma,x} \cdot 1 \\ 0 & k_y\zeta \cdot 2 & V_{\sigma,y} \cdot 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$K'_{[1.2:2.3]} = \begin{bmatrix} k_x\zeta \cdot 1.2 & s & V_{\sigma,x} \cdot 1.2 \\ 0 & k_y\zeta \cdot 2.3 & V_{\sigma,y} \cdot 2.3 \\ 0 & 0 & 1 \end{bmatrix}$$

Von den detektierten korrespondierenden Bildpunkten des SURF-Algorithmus wurden die Bildunkte des zweiten Bildes von C' auch um die selben Verhältnisse skaliert.

$$m_{\sigma[5:2]} = m_\sigma \cdot \begin{pmatrix} 5 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.51}$$

$$m_{\sigma[1:2]} = m_\sigma \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.52}$$

$$m_{\sigma[1.2:2.3]} = m_\sigma \cdot \begin{pmatrix} 1.2 & 0 & 0 \\ 0 & 2.3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.53}$$

Der Szenenrekonstruktionsalgorithmus wurde für jede Kameraauflösung getestet. Die Abbildungen 5.21, 5.21 und 5.23 zeigen jeweils die vier Lösungen für T', welche sich bei der Bestimmung der extrinsischen Parameter bei den unterschiedlichen Kameraauflösungen ergaben. Zu beobachten ist, dass sich die Lösungen bei unterschiedlichen Auflösungen nicht unterscheiden. Die geringen Abweichungen in den Nachkommastellen sind auf die Ungenauigkeiten der Bilddaten zurückzuführen. Durch die skalierung der Bildkoordinaten in die neuen Sensorkoordinatensysteme werden auch deren Abweichungen mit skaliert.

$$\mathrm{T1} = \begin{pmatrix} 0.991092 & 0.120891 & 0.0558752 & -0.812277 \\ -0.120366 & 0.992649 & -0.0126719 & 0.0389145 \\ -0.0569964 & 0.00583352 & 0.998357 & 0.581973 \end{pmatrix}$$

$$\mathrm{T2} = \begin{pmatrix} 0.991092 & 0.120891 & 0.0558752 & 0.812277 \\ -0.120366 & 0.992649 & -0.0126719 & -0.0389145 \\ -0.0569964 & 0.00583352 & 0.998357 & -0.581973 \end{pmatrix}$$

$$\mathrm{T3} = \begin{pmatrix} 0.378236 & -0.0296342 & -0.925235 & -0.812277 \\ 0.0547644 & -0.997021 & 0.0543212 & 0.0389145 \\ -0.924088 & -0.0712161 & -0.375487 & 0.581973 \end{pmatrix}$$

$$\mathrm{T4} = \begin{pmatrix} 0.378236 & -0.0296342 & -0.925235 & 0.812277 \\ 0.0547644 & -0.997021 & 0.0543212 & -0.0389145 \\ -0.924088 & -0.0712161 & -0.375487 & -0.581973 \end{pmatrix}$$

Abbildung 5.17: Zeigt die Die rekonstruierte Matrix T' bei unveränderter Auflösung. Die Auflösungen von C_δ und C'_δ sind die selben.

$$\mathrm{T1} = \begin{pmatrix} 0.991142 & 0.121017 & 0.0546967 & 0.812113 \\ -0.120498 & 0.992632 & -0.0126975 & -0.0388039 \\ -0.0558303 & 0.00599422 & 0.998422 & -0.582208 \end{pmatrix}$$

$$\mathrm{T2} = \begin{pmatrix} 0.991142 & 0.121017 & 0.0546967 & -0.812113 \\ -0.120498 & 0.992632 & -0.0126975 & 0.0388039 \\ -0.0558303 & 0.00599422 & 0.998422 & 0.582208 \end{pmatrix}$$

$$\mathrm{T3} = \begin{pmatrix} 0.376619 & -0.0296193 & -0.925895 & 0.812113 \\ 0.0551443 & -0.996999 & 0.0543246 & -0.0388039 \\ -0.924725 & -0.0715175 & -0.373856 & -0.582208 \end{pmatrix}$$

$$\mathrm{T4} = \begin{pmatrix} 0.376619 & -0.0296193 & -0.925895 & -0.812113 \\ 0.0551443 & -0.996999 & 0.0543246 & 0.0388039 \\ -0.924725 & -0.0715175 & -0.373856 & 0.582208 \end{pmatrix}$$

Abbildung 5.18: Zeigt die rekonstruierte Matrix T' wenn K' mit einem Verhältnis von [5 : 2] skaliert wurde

$$T1 = \begin{pmatrix} 0.990947 & 0.120272 & 0.0596553 & -0.810768 \\ -0.119751 & 0.992728 & -0.0122401 & 0.0387536 \\ -0.0606937 & 0.0049855 & 0.998144 & 0.584083 \end{pmatrix}$$

$$T2 = \begin{pmatrix} 0.990947 & 0.120272 & 0.0596553 & 0.810768 \\ -0.119751 & 0.992728 & -0.0122401 & -0.0387536 \\ -0.0606937 & 0.0049855 & 0.998144 & -0.584083 \end{pmatrix}$$

$$T3 = \begin{pmatrix} 0.37685 & -0.0292569 & -0.925812 & -0.810768 \\ 0.0543725 & -0.997079 & 0.0536412 & 0.0387536 \\ -0.924677 & -0.0705534 & -0.374158 & 0.584083 \end{pmatrix}$$

$$T4 = \begin{pmatrix} 0.37685 & -0.0292569 & -0.925812 & 0.810768 \\ 0.0543725 & -0.997079 & 0.0536412 & -0.0387536 \\ -0.924677 & -0.0705534 & -0.374158 & -0.584083 \end{pmatrix}$$

Abbildung 5.19: Zeigt die rekonstruierte Matrix T' wenn K' mit einem Verhältnis von [1 : 2] skaliert wurde

$$T1 = \begin{pmatrix} 0.990963 & 0.12034 & 0.0592445 & -0.81095 \\ -0.119818 & 0.99272 & -0.0122887 & 0.0387766 \\ -0.060292 & 0.0050791 & 0.998168 & 0.583829 \end{pmatrix}$$

$$T2 = \begin{pmatrix} 0.990963 & 0.12034 & 0.0592445 & 0.81095 \\ -0.119818 & 0.99272 & -0.0122887 & -0.0387766 \\ -0.060292 & 0.0050791 & 0.998168 & -0.583829 \end{pmatrix}$$

$$T3 = \begin{pmatrix} 0.377058 & -0.0293027 & -0.925726 & -0.81095 \\ 0.0544044 & -0.997073 & 0.0537206 & 0.0387766 \\ -0.92459 & -0.0706194 & -0.37436 & 0.583829 \end{pmatrix}$$

$$T4 = \begin{pmatrix} 0.377058 & -0.0293027 & -0.925726 & 0.81095 \\ 0.0544044 & -0.997073 & 0.0537206 & -0.0387766 \\ -0.92459 & -0.0706194 & -0.37436 & -0.583829 \end{pmatrix}$$

Abbildung 5.20: Zeigt die rekonstruierte Matrix T' wenn K' mit einem Verhältnis von [1.2 : 2.3] skaliert wurde

Die Abbildungen 5.21, 5.22 und 5.23 zeigen einmal die Rekonstruierten Punkte in einem 3D-Plot und daneben den 2D-Plot. Auch hier kann beobachtet werden, dass es immer zu den gleichen rekonstruierten Szenen kommt. In Abbildung 5.22 ist die Rekonstruktion in einem links drehendem Koordinatensystem geplottet, weshalb die Rekonstitution spiegelverkehrt wirkt.

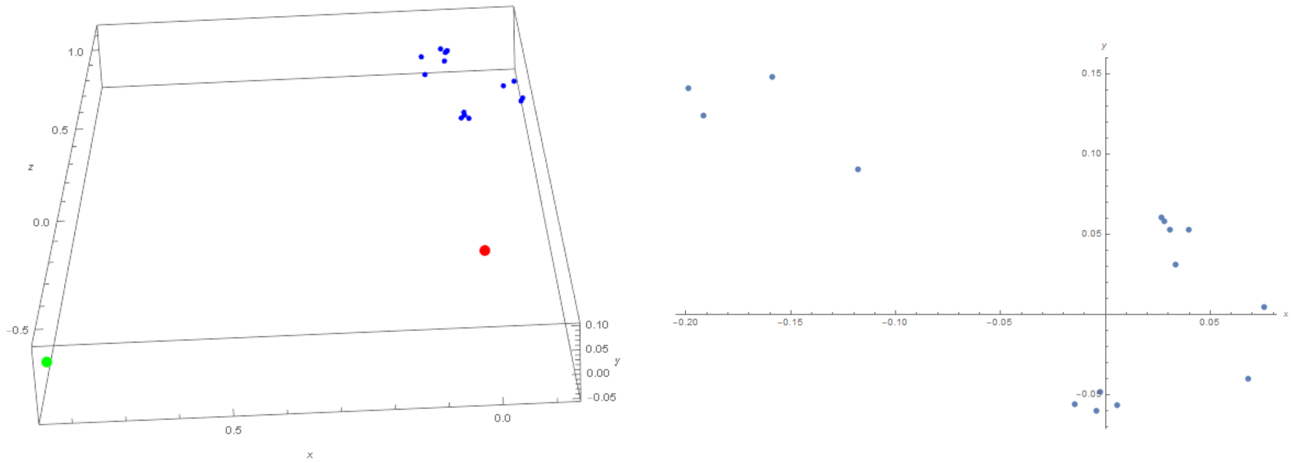


Abbildung 5.21: Rekonstruierte Szene, wenn K' mit einem Verhältnis von [5 : 2] skaliert wurde

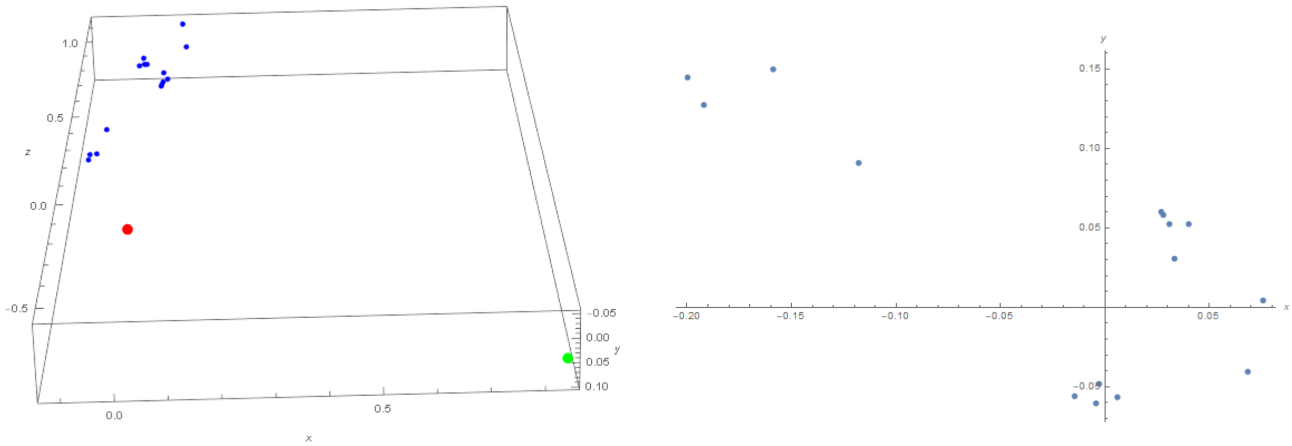


Abbildung 5.22: Rekonstruierte Szene, wenn K' mit einem Verhältnis von $[1:2]$ skaliert wurde

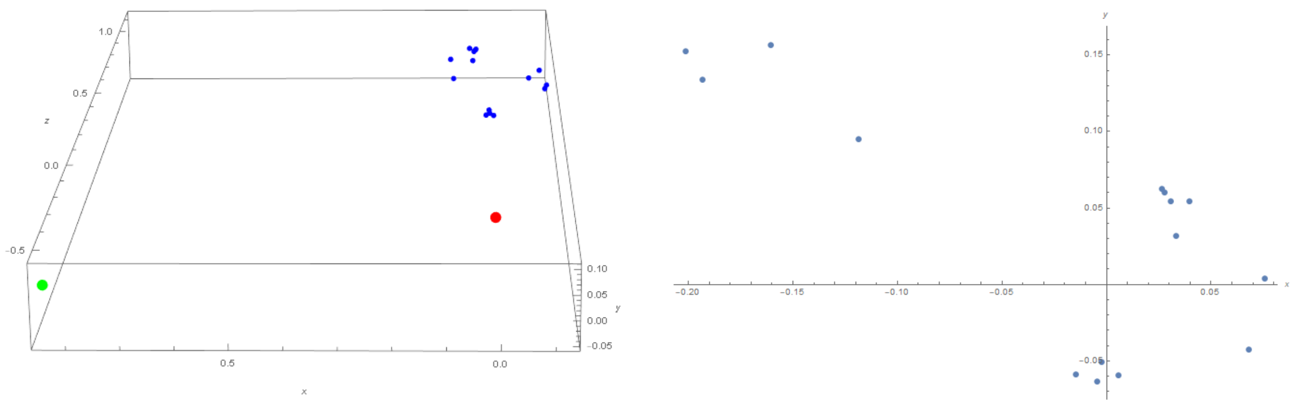


Abbildung 5.23: Rekonstruierte Szene, wenn K' mit einem Verhältnis von $[1.2:2.3]$ skaliert wurde

6 Prinzip Szenenrekonstruktion mit Rektifizierung

Ein verbreiteter Ansatz der Stereobildanalyse basiert auf zuvor Rektifizierten Bildern. Rektifizierte Bilder zeichnen sich durch ins unendlich projizierte Epipole aus. Dies hat zu Folge, dass die jeweiligen Epipolarlinien der Bilder parallel zueinander verlaufen. Anschließend werden die Epipole noch so rotiert, dass die Epipolarlinien in einheitlichen Scanlinien über beide Bilder verlaufen. In Abbildung 6.1 ist das graphisch dargestellt[29, 30, 31, 32, 15].

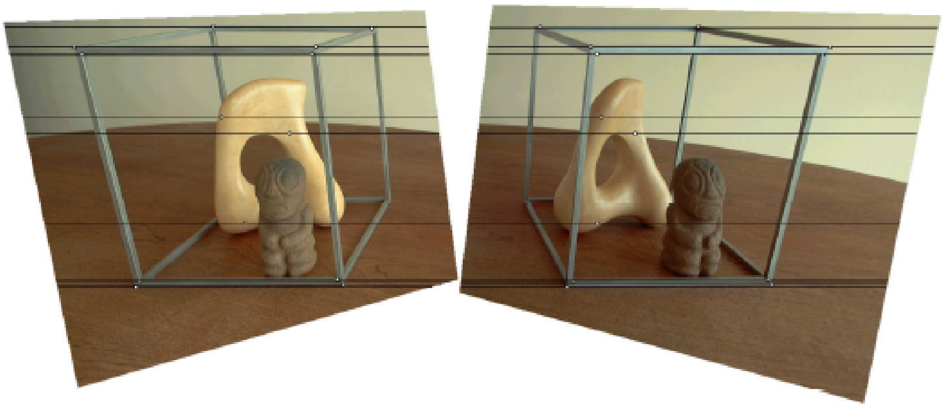


Abbildung 6.1: Beipiel eines rektifizierten Bildes. Die Epipole beider Bilder sind ins unendliche projiziert worden. Des Weiteren wurden die Epipol so rotiert, dass die Epipolarlinen zu einheitlichen Scanlinien über beide Bilder verlaufen. Quelle: [31]

Anhand der so entstandenen Scanlinien, wird die Suche nach weiteren korrespondierenden Punkten auf eine eindimensionale Suche beschränkt. Der korrespondierende Punkt zu einem ausgewählten Punkt auf einem Bild, wird nur noch entlang der entlang der entsprechenden korrespondierenden Epipolarlinie gesucht. Die Rektifizierung ermöglicht eine effiziente Analyse von Punktekorrespondenzen ganzer Bilder mit geringstem Rechenaufwand[31, 29, 30]. Jedoch verlangen viele Rektifizierungsansätze als Voraussetzung, dass das verwendetet Stereobildpaar die selbe Auflösung besitzt.

Im Verlauf des Kapitels wird zunächst der Arbeitsprozess der Stereoanalyse auf Basis von Bildrektifizierungen erläutert. Anschließend wird ein Rektifizierungsalgorithmus nach *Charles Loop & Zhengyou Zhang*[31] vorgestellt, welcher nur durch Vorwissen von neun korrespondierenden Punkten und der daraus resultierenden Fundamentalmatrix eine Rektifizierung zweier Bilder ermöglicht. Zuerst werden zwei Bilder gleicher Auflösung rektifiziert, danach werden zwei Tests mit Bildern unterschiedlicher Auflösung durchgeführt. Die Resultate der rektifizierten Bilder mit unterschiedlichen Auflösungen werden im Anschluss analysiert

6.1 Szenenrekonstruktion mit Rektifizierung

Bestimmte Formen der Rektifizierung benötigen keine vorherige Kalibrierung der Kameras und werden in den meisten gängigen Szenenrekonstruktionen eingesetzt. [29, 30, 33]. Der Arbeitsprozess für die Szenenrekonstruktion auf Basis von rektifizierten Bildern, welcher in Abbildung 6.2 schematisch dargestellt ist, unterscheidet sich etwas zu den bereits bekannten Arbeitsprozessen, welche in den Abbildungen 4.1 und 5.1 zusammengefasst sind.

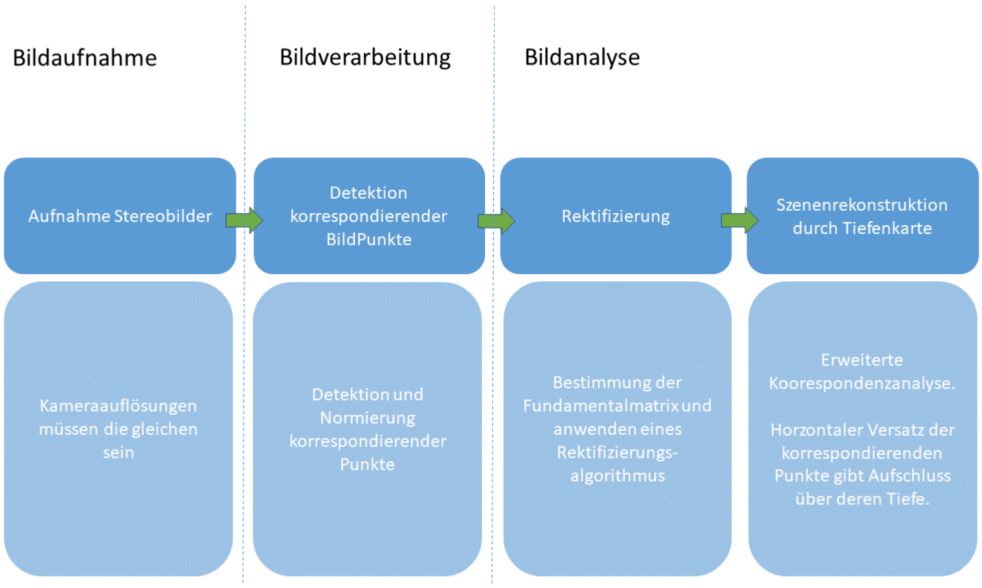


Abbildung 6.2: Ablaufdiagramm der Szenenrekonstruktion basierend auf einem Rektifizierungsansatz

Der entscheidende Unterschied zwischen den beiden Ansätzen liegt im Abschnitt der Bildanalyse. Die Bilder werden vor der Szenenrekonstruktion rektifiziert. Die Rektifizierung benötigt keine extrinsischen oder intrinsischen Kameraparameter und erlaubt eine Rekonstruktion einer Szene mit minimalen vorher bestimmten Informationen[31, 30, 29, 15].

Zunächst werden acht bis neun korrespondierende Punkte in einem Stereobildpaar detektiert. Aus den zuvor noch normierten korrespondierenden Punkten wird eine Fundamentalmatrix F bestimmt, wie in Kapitel 5 beschrieben. Aus den korrespondierenden Punkten und der Fundamentalmatrix F wird ein Rektifizierungsalgorithmus angewandt, welcher zwei Stereobilder transformiert, sodass ihre Epipolarlinien zu horizontal ausgerichteten Scanlinien werden, wie in Abbildung 6.1 zu sehen ist. Mit Hilfe dieser entstandenen Scanlinien, wird die Suche nach weiteren korrespondierenden Punkten von einer zweidimensionalen auf eine eindimensionale Suche vereinfacht.[31, 29, 30].

Nach der Rektifizierung ist zwischen zwei korrespondierenden Punkten ein horizontaler Versatz zu verzeichnen, welcher durch die unterschiedliche Perspektive der Bilder entsteht. Dieser Versatz entsteht durch den Abstand der beiden Kameras zueinander. Der Versatz zwischen den beiden korrespondierenden Punkte wird als Disparität bezeichnet und ist ein Maß für die Tiefe der Punkte in der Szene. Weit von den Kameras entfernte Punkte werden durch die unterschiedliche Perspektive der beiden weniger stark beeinflusst als Punkte welche sich nah an den Kameras befinden. Die Disparität lässt sich als Disparitätskarte oder Tiefenkarte für stereoskopische Bilder veranschaulichen[30]. Der Disparität wird ein Grauwert zugeordnet. In Abbildung 6.3 wurden Punkten mit einer kleinen Disparität ein dunkler Wert und Punkten mit einer großen Disparität ein heller Wert zugeordnet. Die Farbwerte geben somit ein Maß für die Tiefe der Tiefe der Punkte in der Szene[30]. Die Rekonstruktion durch eine Disparitätskarte wird auch als *dense rectification* bezeichnet und bietet eine schnelle und auch effiziente Möglichkeit ein Maß für die Tiefe einer Szene zu generieren.[30, 29, 31]

Abbildung 6.3: In den oberen beiden Bildern sind in blau zwei korrespondierende Epipolarlinien zu sehen, die zusammen eine Scanlinie bilden. Im Bild darunter ist eine aus den Dispritäten zweier Bilder zueinander entstandene Disparitäts beziehungsweise Tiefenkarte zu sehen. Quelle: [30]

Das Standard Rektifizierungsverfahren, welches im folgenden Kapitel eingeführt wird, ist für Bilder gleicher Auflösung definiert. Für den Fall ungleicher Auflösungen wird dieses Verfahren erweitert und anhand von Beispielen die Anwendbarkeit des Rektifizierungsalgorithmus überprüft.

6.2 Rektifizierung mit Homographien

Im Folgenden wird ein Rektifizierungsalgorithmus nach *Zhang*[31] vorgestellt. Diese Art der Rektifizierung zeichnet sich durch minimale Voraussetzungen an die Ursprungsbilder aus. Alle notwendigen Informationen zur Rektifizierung werden aus der Fundamentalmatrix gewonnen. Zusätzlich zur Fundamentalmatrix muss noch die Lage der jeweiligen Epipole bekannt sein[15]. Anhand des entstandenen Algorithmus wird dann getestet, ob Bilder unterschiedlicher Auflösung richtig rektifiziert werden können und was die Ergebnisse für den weiteren Verlauf der Szenenrekonstruktion bedeuten könnten.

Für den Rektifizierungsansatz wird pro Bild eine Homographiematrix H und H' aufgestellt.

$$\bar{m}_\sigma = H m_\sigma \tag{6.1}$$
$$\bar{m}'_{\sigma'} = H' m'_{\sigma'} \tag{6.2}$$

Die Fundamentalmatrix, welche aus den rektifizierten korrespondierenden Punkte resultiert, wird mit $\bar{F}$ bezeichnet[31, 15]:

$$
\begin{aligned}
\bar{m}'^{T}_{\sigma'}\bar{F}\bar{m}_{\sigma} = 0 && (6.3)\\
\rightsquigarrow m'^{T}_{\sigma'}H'^{T}\bar{F}Hm_{\sigma} = 0 && (6.4)\\
\rightsquigarrow F = H'^{T}[i]_{\times}H && (6.5)
\end{aligned}
$$

Die Homographien H und H' werden in die projektiven Komponenten H_p und H'_p und die affinen Komponente H_a und H'_a unterteilt. Die affine Komponenten wird wiederum in zwei weitere Komponenten unterteilt. H_r steht für eine Ähnlichkeitstransformation und H_s bezeichnet eine Scherungstransformation[31, 15].

$$
\begin{aligned}
H = H_aH_p &\quad \rightsquigarrow H = H_sH_rH_p && (6.6)\\
H' = H'_aH'_p &\quad \rightsquigarrow H' = H'_sH'_rH'_p && (6.7)\\
& && (6.8)
\end{aligned}
$$

H_p bezeichnet die projektive Komponenten und H_a steht für die affine Komponente. Die affine Rotation und H_s beinhaltet eine Scherungstransformation. H_p beinhaltet mögliche projektive Transformationen, welche dafür sorgen, dass der Epipol e ins Unendliche projiziert wird und die Epipolarlinien parallel zueinander ausgerichtet sind[31, 15]. H_r ist eine Rotationsmatrix, welche die Epipolarlinien parallel zu horizontalen Achse ausrichtet. H_s ist eine Scherungsmatrix, welche durch Minimierung versucht die durch die vorherigen Transformationen H_p und H_r entstandenen projektiven Verzerrungen bestmöglich auszugleichen[31, 15].

Die Reihen der Homographiematrizen H beschreibt drei Linien u, v und w mit $u = (u_a, u_b, u_c)^T$, $(v_a, v_b, v_c)^T$ und $(w_a, w_b, w_c)^T$, welche jeweils durch den Epipol verlaufen. Selbiges gilt auch für H' mit den Linien u', v' und w'. Die Linien v und v' sowie w und w' sind korrespondierende Epipolarlinien. Es entsteht dadurch eine geometrische Beziehung zwischen den beiden Bildern[31].

$$
H = \begin{bmatrix} u^T \\ v^T \\ w^T \end{bmatrix} = \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & w_c \end{bmatrix} \tag{6.9}
$$

$$
H' = \begin{bmatrix} u'^T \\ v'^T \\ w'^T \end{bmatrix} = \begin{bmatrix} u'_a & u'_b & u'_c \\ v'_a & v'_b & v'_c \\ w'_a & w'_b & w'_c \end{bmatrix} \tag{6.10}
$$

Bevor die Matrizen H und H' in ihre projektiven und affinen Komponenten zerlegt werden, wird die letzte Komponenten w_c und w'_c durch Division eliminiert um somit skaleninvariante Matrizen H und H' zu bekommen[31, 15].

$$
H = \begin{bmatrix} u^T \\ v^T \\ w^T \end{bmatrix} = \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & 1 \end{bmatrix} \tag{6.11}
$$

$$
H' = \begin{bmatrix} u'^T \\ v'^T \\ w'^T \end{bmatrix} = \begin{bmatrix} u'_a & u'_b & u'_c \\ v'_a & v'_b & v'_c \\ w'_a & w'_b & 1 \end{bmatrix} \tag{6.12}
$$

Die Matrizen H_p und H'_p beschreiben den projektiven Teil von H und H'. Sie wirken sich auf den projektiven Teil eines Punktes aus und werden dazu verwendet die Epipole ins unendliche zu projizieren[31, 15].

$$H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{bmatrix} \tag{6.13}$$

Für die affinen Komponeten H_a und H'_a gilt:

$$H_a = H \cdot H_p^{-1} = \begin{bmatrix} u_a - v_c w_b & v_c w_a - v_a & 0 \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} \tag{6.14}$$

.
Des Weiteren gilt für H_a und H'_a, dass jeweils nochmal in eine Rotationsmatrix H_r und H'_r und eine Scherungsmatrix H_s und H'_s zerlegt werden[31, 15].

$$H_a = H_s \cdot H_r \tag{6.15}$$

$$H_r = \begin{bmatrix} v_b - v_c w_b & v_a - v_c w_a & 0 \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} \tag{6.16}$$

$$H_s = \begin{bmatrix} u_a & u_b & u_c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.17}$$

H_r und auch H'_r definieren eine Rotation und auch eine Verschiebung, welche die bereits parallelen Epipolarlinien horizontal ausrichtet. Durch die Verschiebung v_c wird der vertikale Versatz zwischen den Epipolarlinien beider Bilder ausgeglichen. Es entstehen die einheitlichen Scanlinien. Die Matrizen H_s und H'_s definieren eine Scherung. Sie gehören nicht zum eigentlichen Rektifizierungsprozess aber dienen dazu die horizontale Verzerrung der Bilder wieder auszugleichen.

6.2.1 Projektive Transformation

Im Folgenden wird die Herleitung der Matrizen H_p und H'_p beschrieben. Die projektiven Matrizen H_p und H'_p werden von den Linien w und w'. w und w' werden nicht willkürlich gewählt. Definiert werden sie durch eine Richtung $z = \begin{bmatrix} \lambda & \mu & 0 \end{bmatrix}^T$. z soll dabei so gewählt werden, dass die durch die Rektifizierung entstehenden Bildverzerrungen in beiden Bildern minimal bleibt. Die Linien w und w' werden wie folgt definiert.

$$w = [e]_x \cdot z \tag{6.18}$$

$$w' = F \cdot z \tag{6.19}$$

Unter der Minimierung versteht man in diesem Falle, dass versucht wird ein z zu finden, welches $w = (w_a, w_b, w_c)^T$ und $w' = (w'_a, w'_b, w'_c)^T$ so definiert, dass die Eintrage w_a und w_b und auch w'_a und w'_b in H_p und H'_p nahezu null sind. Anders ausgedrückt es wird versucht die projektive Matrizen H_p und H'_p so affin wie möglich zu machen[31].

$$H_p = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ w_a & w_b & 1 \end{bmatrix} \tag{6.20}$$

Sollte es der Fall sein, dass der Epipole e bereits im unendlichen wäre, so wären $w_a = 0$ und $w_b = 0$. In diesem Fall wäre eine projektive Transformation für diesen Epipol nicht mehr nötig.

Für die Minimierung wird die Methode der kleinsten Quadrate angewandt. Die Methode der kleinsten Quadrate ist ein mathematisches Standardverfahren für eine Ausgleichsrechnung. Mit ihrer Hilfe soll aus der Menge der Bildpunkte beider Bilder ein Wert für z ermittelt werden[34]. Für z gilt mit $z = \begin{bmatrix} \lambda & \mu & 0 \end{bmatrix}^T$ bereits die Bedingung, dass es sich um einen Punkt im unendlichen handeln soll. λ und μ, sollen dabei einen Wert annehmen, der sich nah an den Gewichtungen der Punkteansammlungen beider Bilder befindet[31].

Um z zu ermitteln, werden zunächst die Gewichtungen der Punkte beider Bilder benötigt. p_i beinhaltet alle Punkte von Bild eins und p_j beinhaltet alle Punkte von Bild zwei. Ein Punkt $p_{i1} = \begin{bmatrix} p_{i1,u} & p_{i1,v} & 1 \end{bmatrix}^T$ aus Bild eins soll zu einem Punkt $p_{i1} = \begin{bmatrix} \frac{p_{i1,u}}{w_i} & \frac{p_{i1,v}}{w_i} & 1 \end{bmatrix}^T$ mit w_i gleich der Gewichtung

$$w_i = w^T p_i \tag{6.21}$$

transformiert werden. Dasselbe soll auch für die Punkte p_j im zweiten Bild geschehen. Sind die Gewichtungen beider Bilder gleich, so ergibt sich keine projektive Verzerrung und die Epipole sind bereits im unendlichen. Befindet sich einer oder beide Epipole nicht im unendlichen, so können die Gewichtungen der Punkte beider Bilder nicht gleich sein[31].

Das Ziel ist es einen Wert für z zu finden, der die Abweichung der Gewichtungen beider Bilder zueinander so gering wie möglich zu hält. Die Rektifizierung wurde anhand des synthetischen Beispiels in Kapitel 4 implementiert. Dem entsprechend bilden die Abbildungen der Eckpunkte des Quaders auf den Sensoren der virtuellen Kameras, die Punkte in p_i und p_j anhand welcher die Rektifizierung durchgeführt werden soll.

Der Wert p_c mit $p_c = \frac{1}{n}\sum_{i=1}^{n} p_i$ ergibt sich aus dem Durchschnittswert aller verwendeten Punkte eines Bildes und gibt den Bildmittelpunkt an. w_c ist die Gewichtung am Bildmittelpunkt und wird berechnet mit

$$w_c = w^T p_c \tag{6.22}$$

.
Die Abweichung der Gewichtungen wird bezüglich der Gewichtung am Bildzentrum w_c gemessen.

$$\sum_{i-1}^{n} \left[\frac{w_i - w_c}{w_c}\right]^2 \tag{6.23}$$

$$\rightsquigarrow \sum_{i=1}^{n} \left[\frac{w^T(p_i - p_c)}{w^T p_c}\right]^2 \rightsquigarrow \sum_{i=1}^{n} \left[\frac{w^T(p_i - p_c)(p_i - p_c)^T w}{w^T p_c p_c^T w}\right] \tag{6.24}$$

Vereinfacht lässt sich das auch in einer Matrixgleichung in Form von

$$\frac{w^T P P^T w}{w^T p_c p_c^T w} \tag{6.25}$$

angeben, in welcher für P gilt

$$P = \begin{bmatrix} p_{1,u} - p_{c,u} & p_{2,u} - p_{c,u} & \dots & p_{i,u} - p_{c,u} \\ p_{1,v} - p_{c,v} & p_{2,v} - p_{c,v} & \dots & p_{i,v} - p_{c,v} \\ 0 & 0 & \dots & 0 \end{bmatrix} \tag{6.26}$$

.

Für die Punkte p_j in Bild zwei ergibt sich dann entsprechend die Matrixgleichung

$$\frac{w'^T P' P'^T w'}{w'^T p'_c p'^T_c w'} \tag{6.27}$$

.

w und w^T werden nun noch mit ihren Definitionen aus den Gleichungen 6.18 und 6.19 ersetzt und die Gleichungen 6.25 und 6.27 als eine Funktion von z ausgedrückt[31].

$$\frac{z^T [e]_x^T P P^T [e]_x z}{z^T [e]_x^T p_c p_c^T [e]_x z} + \frac{z^T F^T P' P'^T F z}{z^T F^T p'_c p'^T_c F z} \tag{6.28}$$

Gleichung 6.28 wird noch vereinfach mit

$$A = [e]_x^T P P^T [e]_x \tag{6.29}$$
$$B = [e]_x^T p_c p_c^T [e]_x \tag{6.30}$$
$$A' = F^T P' P'^T F \tag{6.31}$$
$$B' = F^T p'_c p'^T_c F \tag{6.32}$$
$$\rightsquigarrow \frac{z^T A z}{z^T B z} + \frac{z^T A' z}{z^T B' F z} \tag{6.33}$$
$$f(z) = \frac{z^T A z}{z^T B z} + \frac{z^T A' z}{z^T B' F z} \tag{6.34}$$

.

Da bereits fest gesetzt ist das die dritte Komponente von z Null ist, wird z im Folgenden als zweidimensionaler Vektor mit $z = \begin{pmatrix} \lambda \\ \mu \end{pmatrix}$ dargestellt. A, B, A' und B' sind 3x3-Matrizen, von denen dann nur noch der erste 2×2- Block wichtig ist.

Für eine nicht lineare Optimierung wird das gesamte Polynom 6.34 aufgeteilt. Als erstes, wird $\frac{z^T A z}{z^T B z}$ minimiert und danach $\frac{z^T A' z}{z^T B' z}$. So entstehen zwei Lösungen $\hat{z_1}$ und $\hat{z_2}$, welche über eine Mittelung eine erste Schätzung für z geben[31].

$$z = \frac{\frac{\hat{z_1}}{\|z_1\|} + \frac{\hat{z_2}}{\|z_2\|}}{2} \tag{6.35}$$

Da es sich um eine nicht lineare Optimierung handelt ist die Minimierung von $\frac{z^T A z}{z^T B z}$ gleichzusetzen mit der Maximierung von $\frac{z^T B z}{z^T A z}$. Beide werden als eine Funktion von $f(z)$ definiert. Matrix A wird mit der Choleskyzerlegung[28] in zwei höhere Dreiecksmatrizen zerlegt $A = D^T D$. Dies geht nur da

A nachweislich eine symmetrische und positiv-definite Matrix ist[35, 28]. Des Weiteren wird definiert, dass $y = Dz$ ist und $f(z)$ wird zu $\hat{f}(y)$[31].

$$A = D^T D \tag{6.36}$$

$$y = Dz \rightsquigarrow z = D^{-1}y \tag{6.37}$$

$$f(z) = \frac{z^T Bz}{z^T Az} \tag{6.38}$$

$$\rightsquigarrow f(z) = \frac{z^T Bz}{z^T D^T Dz} \tag{6.39}$$

$$\hat{f}(y) = \frac{y^T D^{-T} BD^{-1} y}{y^T y} \tag{6.40}$$

Da y bis auf einen Skalierungsfaktor bestimmt ist, kann angenommen werden, dass $\parallel y \parallel = 1$ gilt. $\hat{f}(y)$ ist maximiert, wenn y gleich dem Eigenvektor von $D^{-T}BD^{-1}$ ist, welcher mit dem größten Eigenwert von $D^{-}TBD - 1$ assoziiert wird[31]. Für $\hat{z_1}$ ergibt sich $\hat{z_1} = D^{-1}y$. Exakt das selbe Verfahren wird für die Bestimmung von z_2 mit $\frac{z^T B' z}{z^T A' z}$ angewandt[31].

Die erste Schätzung von z bietet bereits eine akzeptable Lösung. Um das Ergebnis für z zu optimieren können die Werte z_1 und z_2 in Gleichung 6.34 eingesetzt und ein gemeinsames Minimum gesucht werden. das Minimum ergibt den neuen Wert für $z.z$ wird in die Gleichungen 6.18 und 6.19 eingesetzt und w und w' werden berechnet. Die Einträge der Matrizen H_p und H'_p sind mit $w = (w_a, w_b, w_c)^T$ und $w' = (w'_a, w'_b, w'_c)^T$ bestimmt[31].

Im synthetischen Beispiel die Matrizen H_p und H'_p bestimmt und auf die Abbildungen des Quaders angewandt. Abbildung 6.4 zeigt die unrektifizierten Abbildungen der Quader in den Kameras. In grün ist die Abbildung des Quaders in C zu sehen und in rot ist die Abbildung des Quaders in C'. Abbildung 6.5 zeigt in blau die Epipolarlinien beider Abbildungen.

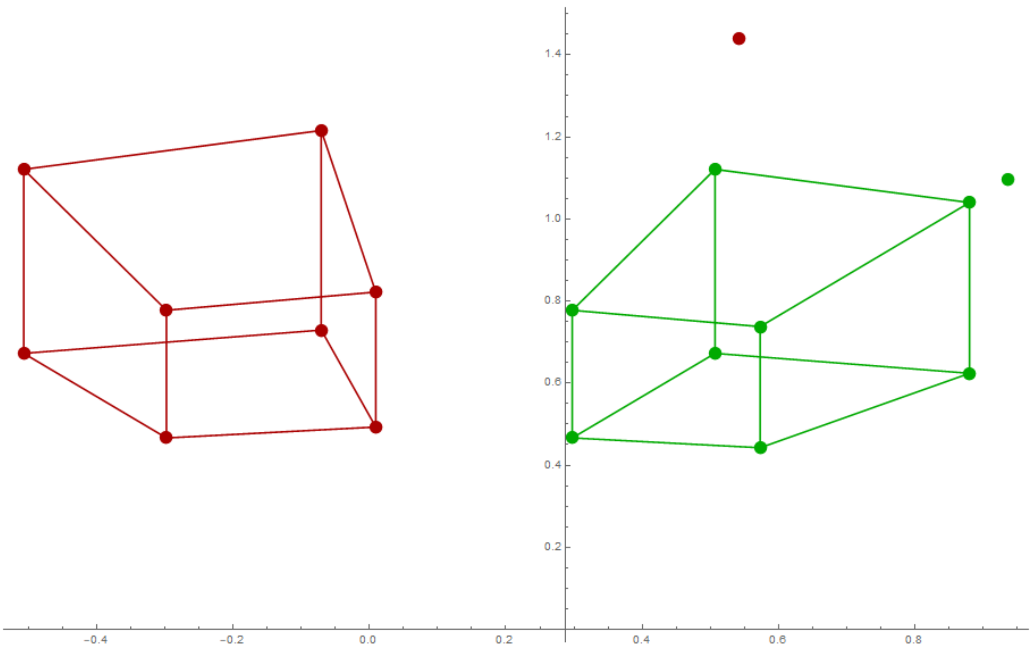


Abbildung 6.4: Aufnahmen zweier Kameras mit den selben Auflösungen, Kamera eins(Grün) und Kamera(rot) zwei gelten jeweils $\zeta = 1$

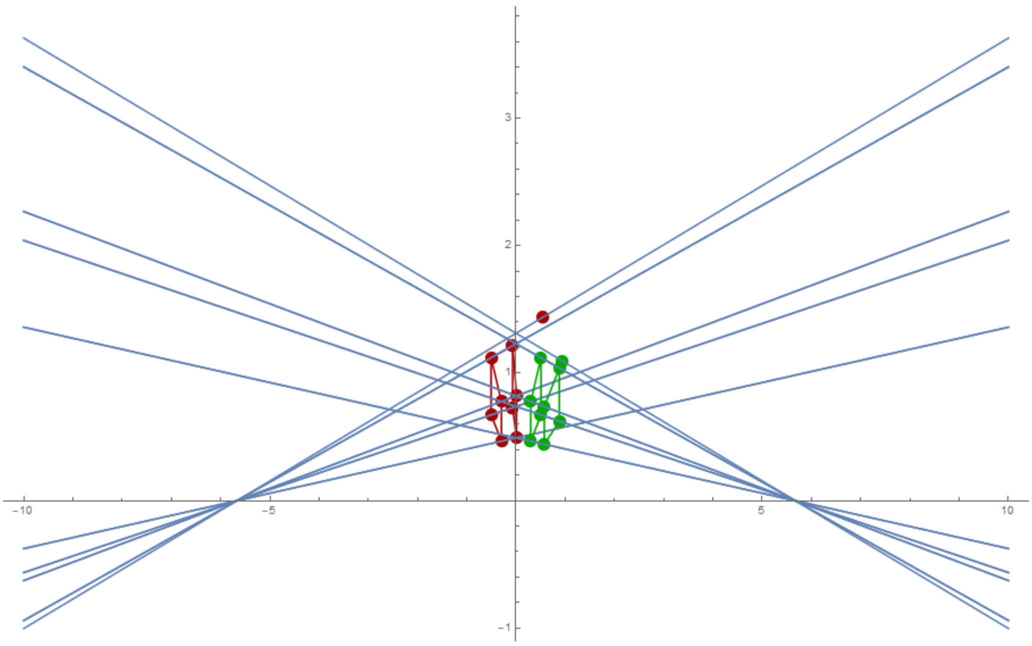


Abbildung 6.5: Epipole für Kamera eins und Kamera zwei vor der Rektifizierung

Nach Transformieren der Abbildungspunkte mit den Matrizen H_p und H'_p wurden die Epipole ins unendliche Transformiert. Die Epipolarlinien verlaufen jetzt parallel zueinander, jedoch noch nicht zwingend parallel zur horizontalen Achse. In den Abbildung en 6.6 und 6.7 ist das Ergebnis der mit H_p und H'_p transformierten Bildpunkte zu sehen.

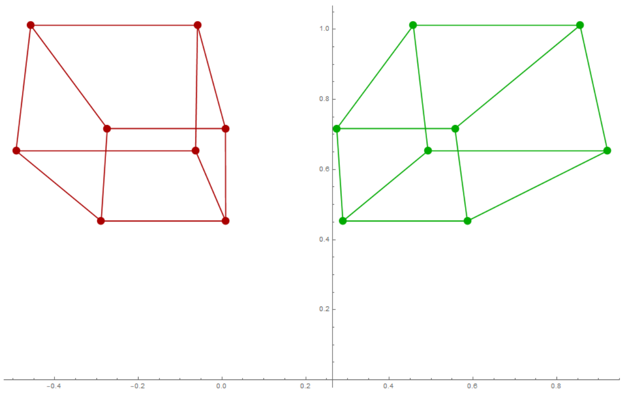


Abbildung 6.6: Abbildungen der Quader nach der Tranfromation mit H_p und H'_p

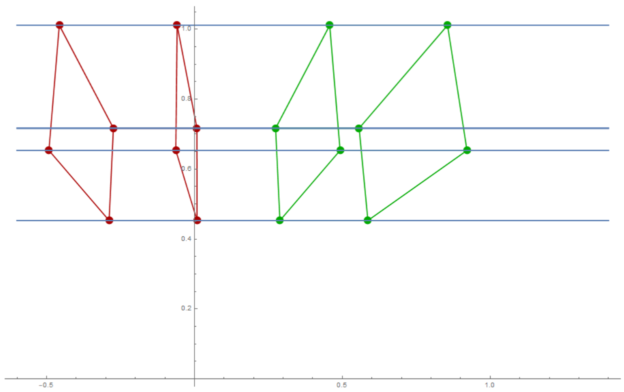


Abbildung 6.7: Abbildungen der Quader nach der Tranfromation mit H_p und H'_p mit eingezeichneten Epipolarlinien

6.2.2 Ähnlichkeitstransformation

Die Epipole der jeweiligen Bilder befinden sich nach der projektiven Transformation im unendlichen. Die daraus resultierenden Epipolarlinien sind, wie in Abbildung 6.7 zu sehen, parallel zueinander angeordnet. Im Folgenden sollen die Epipole rotiert werden, so dass ihre Richtungen $i = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ betragen. H_r und H'_r, welche aus der Zerlegung von H_a und H'_a resultieren, sollen die Rotation ausführen.
w und w' sind bereits bekannt, Mit Hilfe des bekannten F, können v_a und v_b ersetzt werden. Zur Erinnerung eine Voraussetzung für den hier aufgezeigten Rektifizierungsansatz ist, dass sowohl die Bildpunkte als auch die Fundamentalmatrix bekannt sein müssen. Die Epipole wurden ins unendliche projiziert und sollen nach der Rotation die Form $e = [1\ 0\ 0]^T$. Das heißt das nach der Rektifizierung die Fundamentalmatrix $\bar{F}$ die folgende Form haben wird.

$$\bar{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{6.41}$$

Des Weiteren ist bekannt, dass $F = H'^T \bar{F} H$

$$\bar{m}'^T \bar{F} \bar{m} = 0 \rightsquigarrow m'^T H'^T \bar{F} H m = 0 \tag{6.42}$$

$$\rightsquigarrow F = H'^T [i]_\times H \tag{6.43}$$

$$F = \begin{bmatrix} u'_a & v'_a & w'_a \\ u'_b & v'_b & w'_b \\ u'_c & v'_c & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & 1 \end{bmatrix} \tag{6.44}$$

$$F = \begin{bmatrix} v_a w'_a - v'_a w_a & v_b w'_a - v'_a w_b & v_c w'_a - v'_a \\ v_a w'_b - v'_b w_a & v_b w'_b - v'_b w_b & v_c w'_b - v'_b \\ v_a - v'_c w_a & v_b - v'_c w_b & v_c - v'_c \end{bmatrix} \tag{6.45}$$

Bisher sind w, w' und die Fundamentalmatrix F aus den nicht-rektifizierten Punkten bekannt. Um v_a, v_b und v_c zu erhalten, werden jeweils die Einträge der letzten Zeile in F umgeformt. Für v'_a, v'_b und v'_c dient die letzte Spalte von F.

$$v_a = F_{31} + v'_c w_a \tag{6.46}$$

$$v_b = F_{32} + v'_c w_b \tag{6.47}$$

$$v_c = F_{33} + v'_c \tag{6.48}$$

$$v'_a = v_c w'_a - F_{13} \tag{6.49}$$

$$v'_b = v_c w'_b - F_{23} \tag{6.50}$$

$$v'_c = v_c - F_{33} \tag{6.51}$$

F_{31}, F_{32}, F_{33}, F_{13} und F_{23} stehen für die Einträge der Matrix F, welche aus den nicht-rektifizierten Punkten bestimmt wurde. Werden die Gleichungen 6.46 bis 6.51 in H_r und H'_r eingesetzt, so ergeben sich für H_r und H'_r

$$H_r = \begin{bmatrix} v_b - v_c w_b & v_a - v_c w_a & 0 \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} F_{32} - w_b F_{33} & w_a F_{33} - F_{31} & 0 \\ F_{31} - w_a F_{33} & F_{32} - w_b F_{33} & F_{33} + v'_c \\ 0 & 0 & 1 \end{bmatrix} \tag{6.52}$$

$$H'_r = \begin{bmatrix} v'_b - v'_c w'_b & v'_a - v'_c w'_a & 0 \\ v'_a - v'_c w'_a & v'_b - v'_c w'_b & v'_c \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} w'_b F_{33} - F_{23} & F_{13} - w'_a F_{33} & 0 \\ w'_a F_{33} - F_{13} & w'_b F_{33} - F_{23} & v'_c \\ 0 & 0 & 1 \end{bmatrix} \tag{6.53}$$

Die gemeinsame unbekannte Variable v'_c beschreibt die geometrische Verbindung beider Bilder in ihrer Verschiebung entlang ihrer vertikalen Richtung[31]. $F_3 3$ ist somit der horizontale Versatz, welcher benötigt wird, um die horizontalen Epipolarlinien beider Bilder zueinander auszurichten, so dass die gewünschten Scanlinien über beide Bilder entstehen. v' wird dabei so gewählt, dass die kleinste Koordinate der rektifizierten Bilder in vertikaler Achsenrichtung gleich null ist.[31].

Die Abbilder des Quader sehen nach der Transformation mit $H_r H_p$ und $H'_r H'_p$ aus wie in den Abbildungen 6.8 und 6.9 dargestellt. Sollten die Epipolarlinien noch nicht parallel zur horizontalen Achse sein, so sind sie es nach der Transformation mit H_r und H'_r. Des Weiteren wurden beide Bilder durch v'_c so verschoben, dass der in vertikaler Richtung kleinste Punkt auf der Horizontalen Achse liegt.

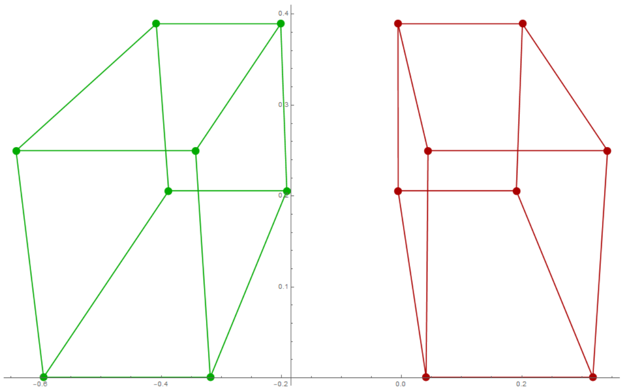


Abbildung 6.8: Abbildungen der Quader nach der Tranfromation mit $H_r H_p$ und $H'_r H'_p$

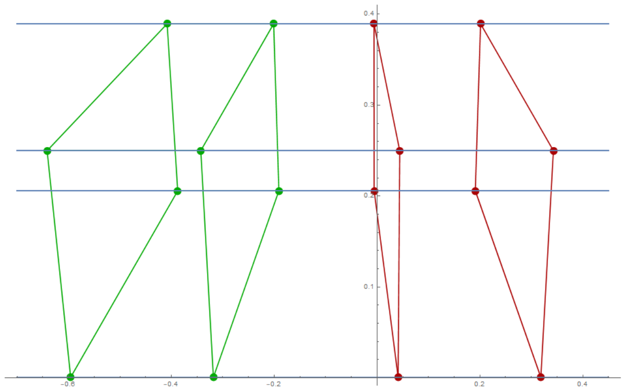


Abbildung 6.9: Abbildungen der Quader nach der Tranfromation mit $H_r H_p$ und $H'_r H'_p$ mit eingezeichneten Epipolarlinien

6.2.3 Scherungstransformation

Im letzten Schritt soll die durch die Transformation mit H_p und H'_p entstandene horizontale Verzerrung in beiden Bilder reduziert werden. Unter der horizontalen Verzerrung versteht man die Verschiebung der gegenüberliegenden Bildkanten zueinander. Die Verbindungslinie der gegenüberliegenden Bildkantenmitten stehen nicht mehr orthogonal aufeinander, was eine Verzerrung des Bildes bewirkt. Abbildung 6.10 stellt diese Verzerrung schematisch dar.

Aus diesem Grund werden die Scherungsmatrizen H_s und H'_s, aus der Zerlegung der affinen Matrix H_a und H'_a benötigt. Die Einträge der ersten zeile in H_s und H'_s haben nur noch Auswirkungen auf die horizontalen Koordinaten der Bildpunkte.

$$H_s = \begin{bmatrix} u_a & u_b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.54}$$

$$H'_s = \begin{bmatrix} u'_a & u'_b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.55}$$

Um die richtigen Werte für u_a, u'_a, u_b und u'_b zu erhalten, werden zunächst Punkte an den jeweiligen gegenüberliegenden Kanten der Bilder definiert. Da die Bilder des Quaders nicht aus mehreren Pixeln bestehen, wie ein reales Bild, sondern nur über dessen Eckpunkte bestimmt ist, wird eine Bildbreite w und w' und eine Bildhöhe h und h' um die Abbildungen des Quaders definiert.

Da es nicht möglich ist die horizontale Verzerrung gänzlich zu reduzieren, wird stattdessen die Orthogonalität der Verbindungslinien $\overline{bd}$ und $\overline{ca}$ wieder hergestellt. In Abbildung 6.10 ist das Ziel grafisch dargestellt.

Für den nächsten Schritt werden im synthetischen Beispiel für die Abbildungen des Quaders jeweils Bildweite w und Bildhöhe h definiert. Da zunächst von gleichen Kameraauflösungen ausgegangen wird, sind die Bildbreiten und Höhen beider Kameras gleich. Danach werden die vier Mittelpunkte a, b, c und d der Bildkanten definiert mit $a = [\frac{w}{2}\ 0\ 1]^T, b = [w\ \frac{h}{2}\ 1]^T, c = [\frac{w}{2}\ h\ 1]^T, d = [0\ \frac{h}{2}\ 1]^T$.

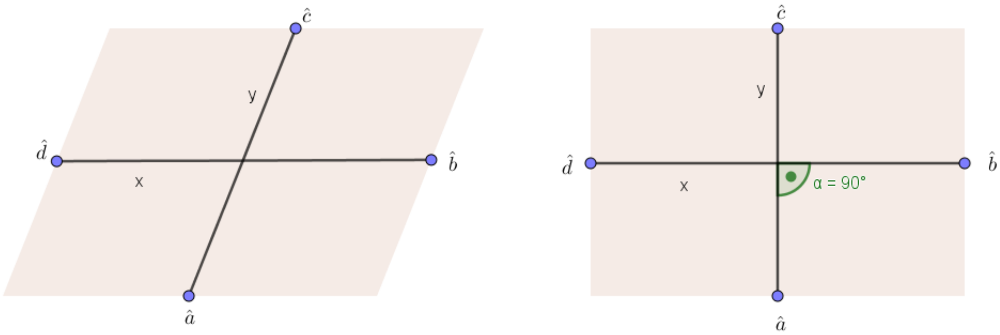


Abbildung 6.10: Die Verbindungslinien $\overline{bd}$ und $\overline{ca}$ sollen so ausgerichtet werden, dass sie orthogonal zueinander stehen.

Die Punkte a, b, c, d und auch a', b', c', d' sind die Mittelpunkte der noch nicht rektifizierten Bildkanten. Diese werden mit den Matrizen $H_r H_p$ und $H'_r H'_p$ transformiert, um so die Position der Kantenmitten nach der Rektifizierung zu erhalten.

$$\hat{a} = H_r \cdot H_p \cdot a$$
$$\hat{b} = H_r \cdot H_p \cdot b$$
$$\hat{c} = H_r \cdot H_p \cdot c$$
$$\hat{d} = H_r \cdot H_p \cdot d$$
$$\hat{a'} = H'_r \cdot H'_p \cdot a'$$
$$\hat{b'} = H'_r \cdot H'_p \cdot b'$$
$$\hat{c'} = H'_r \cdot H'_p \cdot c'$$
$$\hat{d'} = H'_r \cdot H'_p \cdot d'$$

Danach können die Vektoren $\vec{x}$ und $\vec{y}$ der sich ursprünglich gegenüberliegenden Punkte gebildet werden.

$$x = \hat{b} - \hat{d} \tag{6.56}$$
$$y = \hat{c} - \hat{a} \tag{6.57}$$
$$x' = \hat{b}' - \hat{d}' \tag{6.58}$$
$$y' = \hat{c}' - \hat{a}' \tag{6.59}$$

x und y sind Vektoren der euklidischen Bildebene[31]. Das heißt, sie sind genau dann orthogonal zueinander wenn gilt

$$(H_s x)^T (H_s y) = 0 \tag{6.60}$$
$$(H'_s x')^T (H'_s y') = 0 \tag{6.61}$$

.

Für die Erhaltung der Seitenverhältnisse gilt dann:

$$\frac{(H_s x)^T (H_s x)}{(H_s y)^T (H_s y)} = \frac{w^2}{h^2} \tag{6.62}$$
$$\frac{(H'_s x')^T (H'_s x')}{(H'_s y')^T (H'_s y')} = \frac{w'^2}{h'^2} \tag{6.63}$$

Anhand der Gleichungen 6.60, 6.61, 6.62 und 6.63 können die folgenden Gleichungen für die Matrixeinträge u_a und u_b für H_s und H'_s aufgestellt werden[31, 36].

$$u_a = \frac{h^2 x_v^2 + w^2 + y_v^2}{hw(x_v y_u - x_u y_v)} \tag{6.64}$$

$$u_b = \frac{h^2 x_u x_v + w^2 y_u y_v}{hw(x_u y_v - x_v y_u)} \tag{6.65}$$

Die selben Gleichungen werden auch für u'_a und u'_b aufgestellt. Das Ergebnis der gesammten Transformation H mit $H_s H_r H_p$ und H' mit $H'_s H'_r H'_p$ ist in den Abbildungen 6.11 und 6.12 zu sehen.

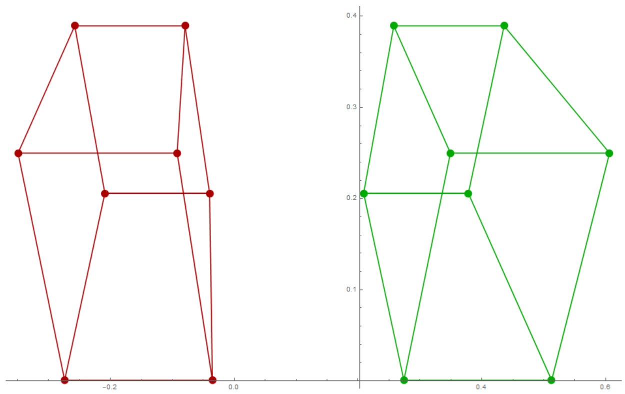


Abbildung 6.11: Abbildungen der Quader nach der Tranfromation mit $H_s H_r H_p$ und $H'_s H'_r H'_p$

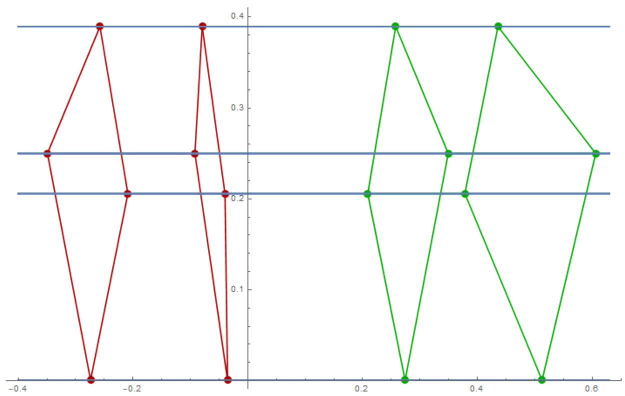


Abbildung 6.12: Abbildungen der Quader nach der Tranfromation mit $H_s H_r H_p$ und $H'_s H'_r H'_p$ mit eingezeichneten Epipolarlinien

6.3 Rektifizierung mit unterschiedlichen Kameraauflösungen

Im Folgenden wird der entstandenen Rektifizierungsalgorithmus auf Bilder unterschiedlicher Auflösung angewandt. Im ersten Beispiel wird für C die Auflösung $\zeta_x = \zeta_y = 1$ gewählt. Die Kameramatrix K lautet wie folgt

$$K = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.66}$$

Für C' wird eine kleinere Auflösung mit $\zeta'_x = \zeta'_y = 0.5$ definiert. Die Kameramatrix K' lautet somit

$$\begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.67}$$

Die entstehenden Bilder des Quaders sind in Abbildung 6.13 zu sehen. Da K' nur eine halb so große Auflösung wie K besitzt, ist das resultierende Bild des roten Quaders auch nur halb so groß. Der grüne Quader zeigt das entstehende Bild von C'.

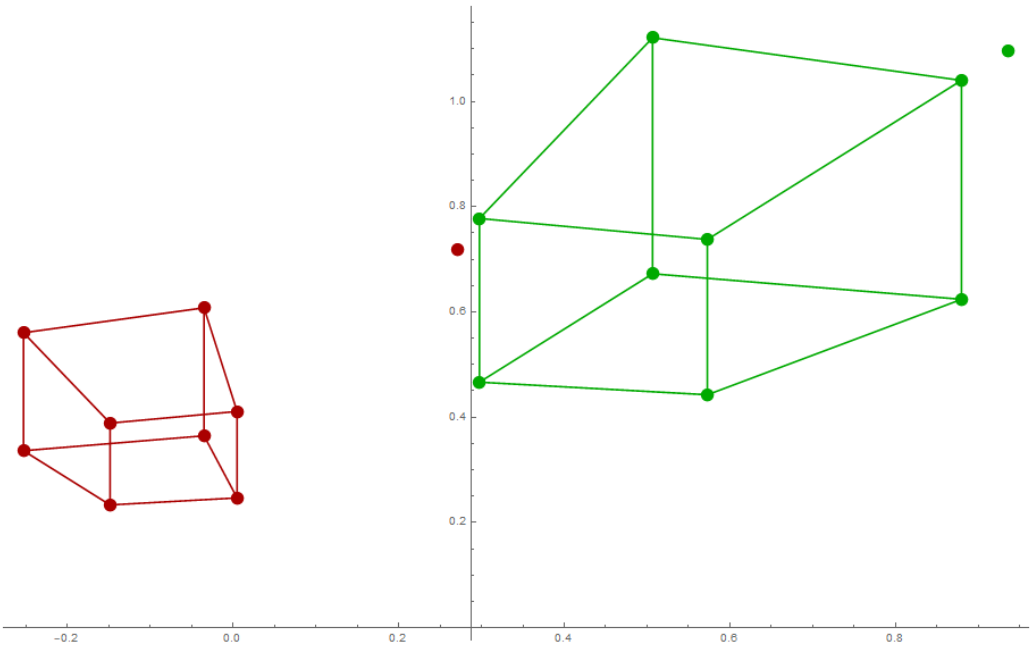


Abbildung 6.13: Aufnahmen zweier Kameras mit unterschiedlichen Auflösungen.Für Kamera eins(Grün) gilt $\zeta_x = \zeta_y = 1$ und für Kamera zwei(rot) gilt $\zeta'_x = \zeta'_y = 2$.

Abbildung 6.14 zeigt die projektive Transformation der beiden Bilder mit H_p und H'_p. Die Epipolarlinien der beiden Bilder sind nach dieser Transformation jeweils parallel zueinander. In Abbildung 6.15 ist das Resultat zu sehen wenn die Transformationen H_r und H'_r dazukommen. Die Epipolarlinien sind jetzt parallel zur horizontalen Achse und die Epipolarlinien von Bild eins und Bild zwei sind so zueinander ausgerichtet, dass sie zu einheitlichen Linien über zwei Bilder werden. Der rote Quader, welches das Bild mit der niedrigeren Auflösung repräsentiert, ist nach dieser Transformation vergrößert.

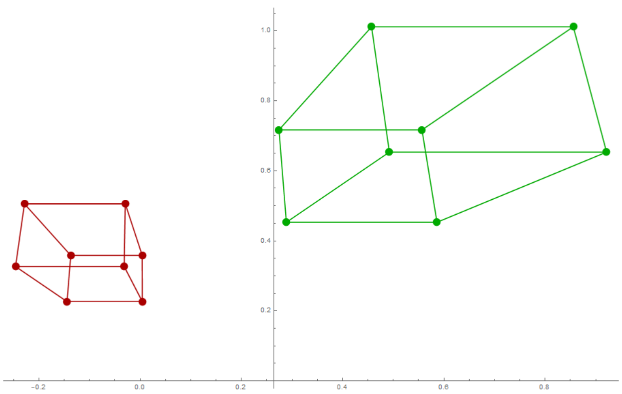


Abbildung 6.14: Transformation H_p und H'_p angewandt auf Bilder unterschiedlicher Auflösungen

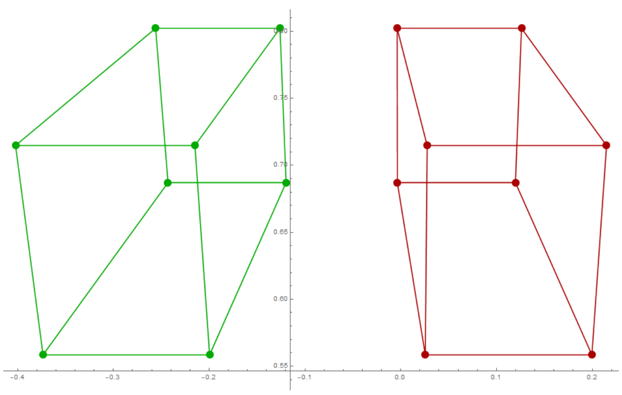


Abbildung 6.15: Transformation $H_r H_p$ und $H'_r H'_p$ angewandt auf Bilder unterschiedlicher Auflösungen

Die Abbildungen 6.16 und 6.17 zeigen die letzte Transformation mit H_s und H'_s. Die Bilder scheinen richtig rektifiziert geworden zu sein. Dies wurde noch mit weiteren Vielfachen der Kameramatrix K ausprobiert. Für alle getesteten Fäller ergab sich dasselbe Ergebnis, wie es in den Abbildungen 6.16 und 6.17 zu sehen ist. Die Transformationen während der Rektifizierung können unterschiedliche Auflösungen ausgleichen und es ist folglich möglich eine Tiefenkarte zu erstellen um die 3D Szene zu rekonstruieren.

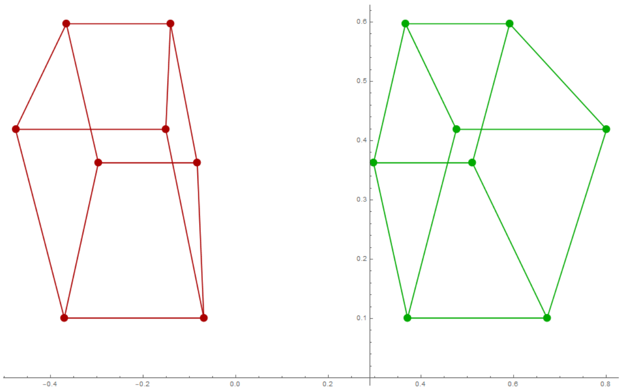


Abbildung 6.16: Transformation $H_s H_r H_p$ und $H'_s H'_r H'_p$ angewandt auf Bilder unterschiedlicher Auflösungen

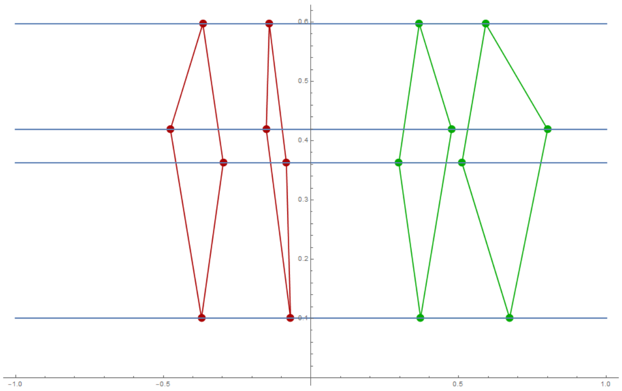


Abbildung 6.17: [Transformation $H_s H_r H_p$ und $H'_s H'_r H'_p$ angewandt auf Bilder unterschiedlicher Auflösungen mit Epipolarlinien

Als nächstes wurden die Auflösung von C' so verändert, dass $\zeta'_x \neq \zeta'_y$ gilt. Es werden $\zeta'_x = 2.3$ und $\zeta'_y = 3.2$ definiert. Somit folgt für K' die folgende Matrix

$$K' = \begin{bmatrix} 2.3 & 0 & 0 \\ 0 & 3.2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.68}$$

.
Für K von C gilt weiterhin $\zeta_x = \zeta_y = 1$. Die entstandenen Bilder sind in Abbildung 6.18 zu sehen. Die horizontale Kantenlänge des roten Quaders ist im Verhältnis kürzer als ihre vertikale Kantenlänge. Wie in Abbildung 6.19 zu sehen ist, bleibt ungleiche Veränderte Seitenverhältnis des roten Quaders nach der Rektifizierung erhalten, weshalb der Rote Quader schmäler ist als der der grüne.

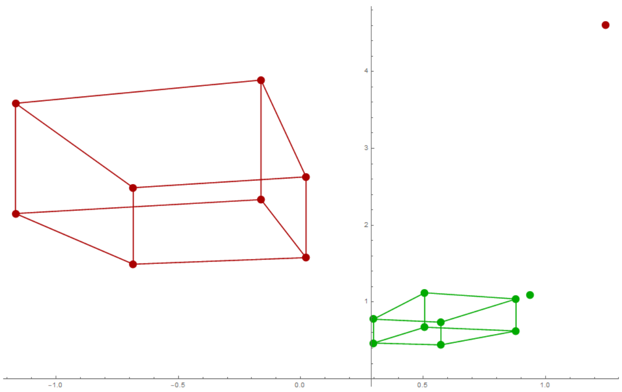


Abbildung 6.18: Aufnahmen zweier Kameras mit unterschiedlichen Auflösungen.Für Kamera eins(grün) gilt $\zeta_x = \zeta_y = 1$ und für Kamera zwei(rot) gilt $\zeta'_x = 2.3$ $\zeta'_y = 3.2$

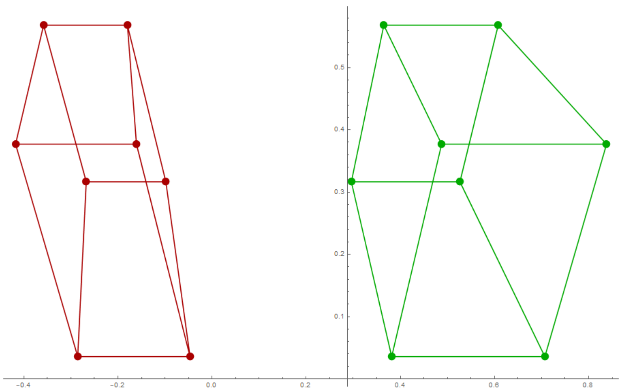


Abbildung 6.19: Nach der Rektifizierung stimmen die horizontalen Koordinaten nicht überein, es würde zu Fehlern in der Tiefenkarte und somit in der gesamten 3D-Rekonstruktion geben.

Die Beispiele zeigen, dass es möglich ist Bilder unterschiedlicher Beispiele zu rektifizieren, jedoch müssen für die nachfolgende Erstellung der Tiefenmaps....(was wäre jetzt ein guter Abschluss??)
Diese Beispiele zeigen, dass das hier vorgestellte Rektifizierungsverfahren auf Bilder verschiedener Auflösungen mit demselben Seitenverhältnis der Pixel anwendbar ist. Jedoch können Bilder mit verschiedenen Seitenverhältnissen mit dem Verfahren nicht rektifiziert werden. Eine mögliche Lösung wäre eine Erweiterung des Models, welche die Bildlängen bestimmt und schließlich die relativen Seitenverhältnisse ausgleicht.

7 Punktesortierung in Schachbrettmustern

Für die Detektion von korrespondierenden Punkten in Stereoskopischen Bildaufnahmen von zweidimensionalen Schachbrettern, ist ein Algorithmus entwickelt worden, welcher die zuvor Detektierten Eckpunkte eines Schachbretts sortiert und eindeutig identifiziert. Jeder Punkt beinhaltet nach der Sortierung die Information in welcher Reihe j und in welcher Spalte i er sich befindet. Jeder Punkt ist somit eindeutig durch die zwei Indizes i und j identifiziert. Dem entstandenen Sorierungsalgorithmus ist ein Algorithmus zu Detektion der Eckpunkte eines Schacbretts voran geschaltet. Werden beide Algorithmen auf die Stereoaufnahme zweier Schachbretter angewandt, so können korrespondierende Punkte anhand dieser Indizes ausgemacht werden. Die Schachbretter können dabei sowohl Kissen- als auch Tonnennverzeichnungen aufweisen und oder perspektivisch verzerrt sein.

Aus den zuvor detektierten Eckpunkten des Schachbretts wird ein Startpunkt gesucht. Der Startpunkt wird so bestimmt, dass es immer die linke unterste Ecke des Schachbretts ist. Somit ist gewährleistet, dass die Indizes der Eckpunkte beider Bilder gleich sind. Ist der Startpunkt bestimmt, bekommt dieser die Indizes $i = 1$ und $j = 1$. i steht für die jeweilige Reihen in welcher sich ein Punkt befindet und j steht für die jeweilige Spalte. Ist der Startpunkt bestimmt, wird der erste Punkt entlang der unteren Schachbrettkante in j-Richtung und der erste Punkt entlang der linken Randkante in i-Richtung gesucht. Anhand dieser Punkte lassen sich vom Startpunkt aus Richtungsvektoren definieren. Entlang des Richtungsvektors wird ein Bereich definiert. In Abbildung 7.1 ist dieser Suchbereich als das blaue Dreieck dargestellt. Ist der nächste Punkt gefunden, so wird der Richtungsvektor anhand des zu vorigen und des neuen Punktes neu ausgerichtet. Der dynamische Suchbereich ermöglicht es somit, dass selbst bei stark verzerrten Schachbrettern, bei denen Punkte nicht mehr auf einer Gerade liegen, Punkte der einzelnen Reihen und Spalten des Schachbretts ausfindig gemacht werden können.

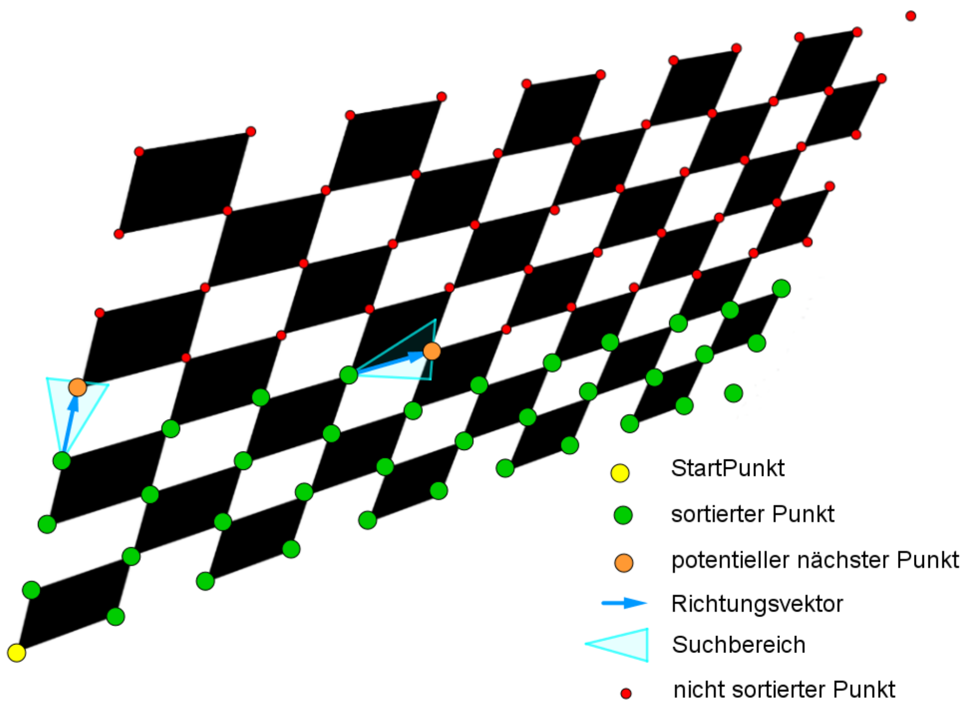


Abbildung 7.1: In der Abbildung ist die Vorgehensweise des Sortierungsalgorithmus schematisch zu sehen.

7.1 Sortierungsalgorithmus

Im Folgenden soll der Ablauf des implementierten Sortierungsalgorithmus beschrieben werden und dessen Funktionsweise an Beipspielen von unterschiedlichen Schachbrettern demonstriert werden.

Der Soritierungsalgorithmus nimmt eine Liste aus den unsortierten Eckpunkten des Schachbretts entgegen. Von den Punkten in dieser Liste wird eine grobe Vorsortierung vorgenommen.
Um das Schachbrett herum wird ein Rahmen definiert. Die Punkte mit der maximalen y-Koordinate und der minimalen y-Koordinate Begrenzen die oberen und unteren Kanten des Rahmens. Die Punkte mit der minimalen x-Koordinate und der Punkt mit der maximalen x-Koordinate begrenzen die vertikalen Kanten des Rahmens. In Abbildung 7.2 sind die roten Begrenzungskanten um das Schachbrett zu sehen.

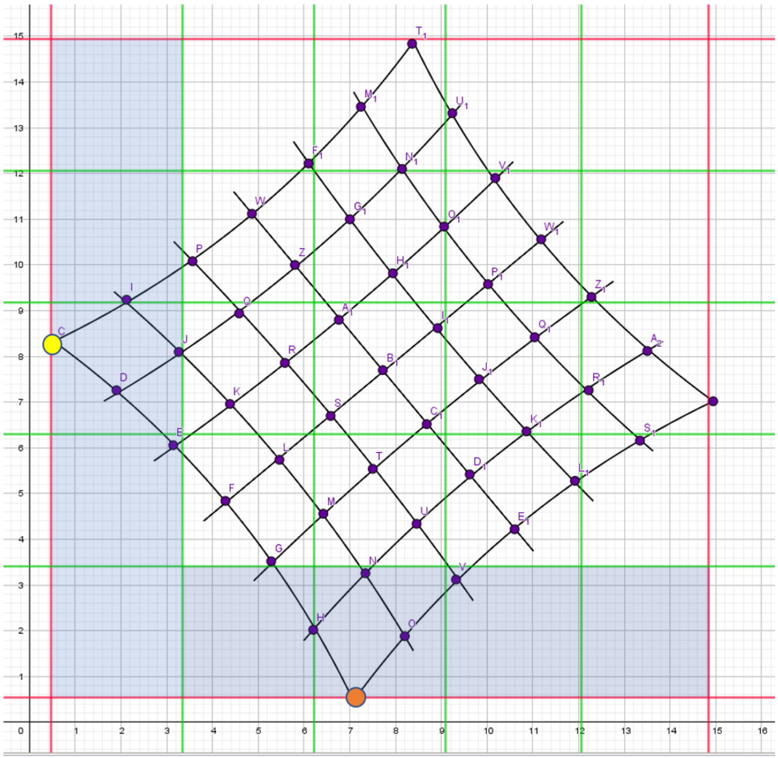


Abbildung 7.2: Die in blau markierten Bereiche beinhalten die möglichen Startpunkte. Der Bereich entlang der horizontalen j- Achse bildet die erste Suchfensterreihe in i-Richtung. Der blaue Bereich entlang der vertikalen i-Achse bildet die erste Suchfensterreihe in j-Richtung. Der gelbe Punkt steht für den Punkt welcher als $VecJ$ bezeichnet wurde und der orange Punkt ist derjenige Punkt, welcher als $VecI$ bestimmt wurde

Der durch den Rahmen begrenzte Bereich, welcher das gesamte Schachbrett einschließt, wird in mehrere Zellen eingeteilt. Diese werden durchgezählt und mit den Indizes i und j eindeutig bestimmt. i beschreibt die Reihennummer der Zelle und j beschreibt die entsprechende Spalte. In Abbildung 7.2 hätte die Zelle links unten im Eck somit die Indizes $i = 1$ und $j = 1$, die zweite rechts daneben die Indizes $i = 1$ und $j = 2$.

In zwei *ConstantArrays* namens *JSplits* und *ISplits* werden die Begrenzungen de Zellen in x- und y-Richtung gespeichert. Die Begrenzungen der Zellen werden berechnet, idem die Distanz zischen den vertikalen und horizontalen Kanten des Rahmens durch die gewünschte Anzahl an Zellen geteilt wird.

Im nächsten Schritt wird überprüft welche Punkte innerhalb welcher Zellen liegen. Für jeden Punkt wird eine Liste Aus *Associations* angelegt. *Associations* können Werten Schlüsselwörter zuweisen. Anhand dieser Schlüssel sind Werte eindeutig identifiziert[22]. Pro Punkt werden die jeweils die x-

und y-Koordinaten und die Indizes Zellen in Schlüssel gespeichert.

$$Punkt = \{<|Coordx \to x_u, Coordy \to y_u, CellI \to i_u, CellJ \to j_u|>\}$$

In Abbildung 7.2 sind die Zellen durch die grünen Linien begrenzt. Die Vorsortierung in Zellen macht es im weiteren Verlauf der Sortierung der Punkte in Zeilen un Spalten einfacher, wenn es darum geht nur bestimmte Bereiche des Schachbretts zu untersuchen.

Nach der Vorsortierung wird der Punkt gesucht, von welchen aus das Schachbrettgitter rekonstruiert werden soll. Alle Punkte innerhalb den Zellen mit den Indizes $i = 1$ und $j \leq j_{max}$ und $i \leq i_{all}$ und $j = 1$ werden als mögliche Startpunkte gekennzeichnet. In Abbildung 7.2 befinden sich die möglichen Startpunkte innerhalb des blau hinterlegten Bereichs.

Um Anhand des Algorithmus später korrespondierende Punkte in zwei Aufnahmen des Schachbretts finden kann, sollte gewährleistet sein, dass der Startpunkt immer an der selben Ecke eines Schachbretts platziert wird.

Die jeweiligen horizontalen Zellen $i = 1$ und $j \leq j_{max}$ werden getrennt von den vertikalen $i \leq i_{all}$ und $j = 1$ untersucht. Innerhalb der horizontalen Zellen wird zunächst nach dem Punkt mit der kleinsten y-Koordinate gesucht und als erster möglicher Startpunkt in $VecI$ gespeichert. Danach wird überprüft, ob es einen Punkt gibt, dessen x-Koordinate kleiner ist als die des momentan gesetzten $VecI$. Ist dies der Fall so wird noch überprüft, ob dessen y-Koordinate kleiner gleich der von $VecI$ plus einem Pufferwert ist. Der Pufferwert wird so definiert, dass die y-Koordinate des neuen möglichen $VecI$ zwar größer als die des momentanen sein kann, jedoch eine bestimmte Schwelle nicht übertreten darf, da es sich sonst um einen Punkt der zweiten i-Reihe handeln könnte. In Abbildung 7.2 ist $VecI$ als oranger Punkt abgebildet.

Innerhalb des vertikalen Suchbereichs bestehend aus den Zellen $i \leq i_{all}$ und $j = 1$ wird derjenige Punkt gesucht, dessen x-Koordinate minimal ist und wird als Punkt $VecJ$ gespeichert. Im nächsten Schritt wird ein Punkt innerhalb des Bereichs gesucht, dessen y-Koordinate kleiner ist als die des momentanen $VecJ$ und dessen x-Koordinaten kleiner gleich der x-Koordinate von $VecJ$ plus einem Pufferwert ist. Ist so ein Punkt gefunden, wird dieser als neues $VecJ$ bestimmt. $VecJ$ ist in Abbildung 7.2 als gelber Punkt dargestellt.

Je nachdem wie das Schachbrett rotiert ist oder welche Art der Verzerrungen es aufweist, kann es sein dass $VecI$ und $VecJ$ bereits den selben Punkt ergeben haben, was den Startpunkt $StartPoint$ eindeutig identifiziert, wie es in den Beispielen 7.12 oder auch 7.16 der Fall ist. Andererseits kann es auch wie in Abbildung 7.2 zu sehen ist, sein, dass $VecI$ und $VecJ$ sich unterscheiden. In solchen Fällen wird $VecJ$ als Startpunkt festgelegt. Damit ist gewährleistet, dass je nach Lage des Schachbretts der linke obere oder die linke untere Ecke des Schachbretts als Startpunkt gesetzt werden

Die $Assiciation$-Liste des Startpunktes wird um zwei weitere Schlüssel erweitert. Die Schlüssel $NeighbourI$ und $NeighbourJ$ speichern die Werte in welcher Reihe i und welcher Spalte j des Schachbrettgitters sich der gefundenen Punkt befindet. Der Startpunkt $StartPoint$ ist dann wie folgt definiert

$$\begin{aligned} StartPoint = \{<|&Coordx \to x-_u,\ Coordy \to y_u, \\ &CellI \to i_u,\ CellJ \to j_u,\ NeighbourI \to 1,\ NeighbourJ \to 1|>\} \end{aligned}$$

Anhand der Schlüssel $NeighbourI$ und $NeighbourJ$, ist die Position des Punktes innerhalb des Schachbrettgitters eindeutig identifiziert. Somit können in einer Stereoskopischen Aufnahme die korresponierenden Eckpunkte von Schachbrettern identifiziert werden. In Abbildung 7.10 und 7.11 wurde in beiden

Bildern alle Punkte abgefragt, deren Schlüssel $NeighbourI = 3$ ist. Die grün gefärbten Punkte liefern das Ergebnis.

$StartPoint$ wird danach in zwei unterschiedliche Listen geschrieben. Die Liste $SortedPoints$ speichert alle fertig sortierten Punkte. Die Liste $CheckList$ speichert eine verkürzte $Associtaion$- Liste der Punkte, welche nur aus den Schlüsseln $Coordx$ und $Coordy$ besteht. Anhand der in $Checklist$ gespeicherten Koordinaten eines Punktes, soll bei der weiteren Suche verhindert werden, dass ein Punkt mehrmals als sortiert wird.

Nachdem der Startpunkt gefunden ist werden die jeweils nächsten Punkte in i- und j-Richtung gesucht. Diese Punkte werden dann in die Variablen $NextI$ und $NextJ$ gespeichert. In Abbildung 7.3 ist $NextI$ der violette Punkte und $NextJ$ ist der rote Punkte.

Für die Bestimmung von $NextI$ wird zu erst initial $NextI$ $=< |Coordx \to 100000, Coordy \to 100000| >$ gesetzt. Für den Punkt $NextI$ der nächsten Reihe i wird die Zelle in welchem sich $StartPoint$ befindet und die jeweiligen Zellen eins darüber und darunter durchsucht. In Abbildung 7.3 ist dieser Bereich um $StartPoint$ gelb hinterlegt.

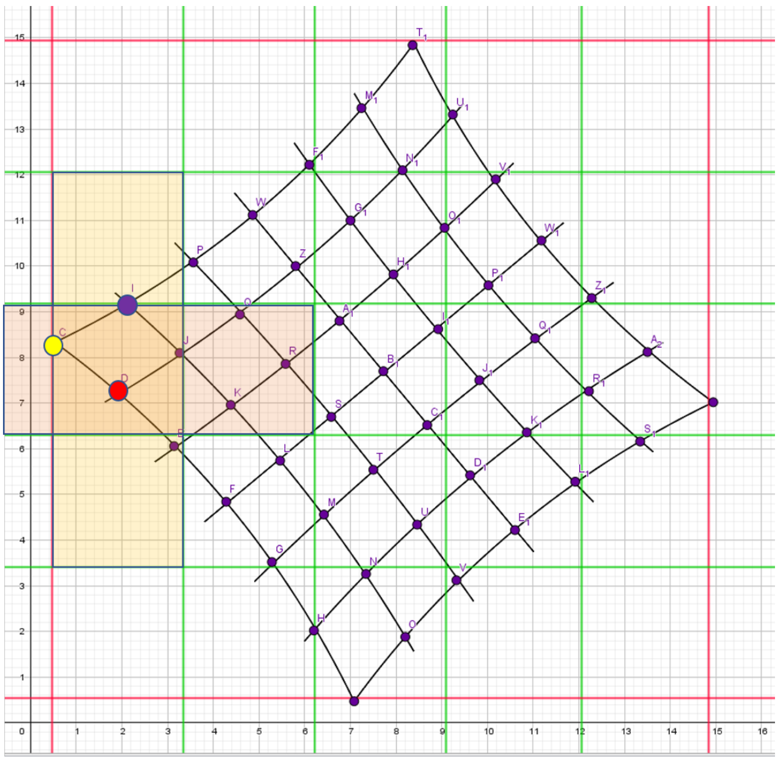


Abbildung 7.3: Der in gelb markierte Bereich beinhaltet die für $NextI$ möglichen Punkte. der orange markierte Bereich beinhaltet die für $NextJ$ möglichen Punkte. Der violett gefärbe Punkt ist der gesuchte $NextI$, der rot gefärbte Punkt ist der gesuchte $NextJ$

Innerhalb des Bereichs wird der Punkt mit der zu $StartPoint$ nächst höheren y-Koordinate ermittelt. Dieser Punkt wird dann als vorläufiges $NextI$ festgelegt. Danach wird geprüft, ob das neu bestimmte $NextI$ bereits das letztendliche $NextI$ ist. Um das zu prüfen wird zunächst überprüft ob es innerhalb des gelben Bereichs noch einen Punkt gibt, dessen x-Koordinatenabstand vom Startpunkt aus kleiner gleich dem x-Koordinatenabstand zwischen dem momentanen $NextI$ zum Startpunkt ist. Gleichzeitig wird geprüft, ob der y-Koordinatenabstand zwischen $StartPoint$ und $NextI$ größer ist als der y-Koordinatenabstand zwischen einem anderen Punkt innerhalb des Suchbereichs und $StartPoint$. Ist dies der Fall, so wird der Punkt auf den beides zutrifft als neues $NextI$ bestimmt. In Abbildung 7.4 ist die Überprüfung on das erste $NextI$ bereits stimmt oder geändert werden muss nochmal grafisch veranschaulicht.

Die Suche nach dem nächsten Punkt in j-Richtung erfolgt nach dem gleichen Prinzip. Für $NextJ$ wird der in Abbildung 7.4 rot hinterlegte Bereich untersucht. Innerhalb des Roten Bereichs, wird derjenige Punkt ermittelt der zu $StartPoint$ den nächst höheren x-Wert besitzt. Dieser wird dann als vorläufiger $NextJ$ gespeichert. Danach wird auch über Abstandsrechnungen überprüft, ob es sich bei $NextJ$ bereits um den finalen Punkt handelt oder ob er nochmal geändert wird.

Die beiden Punkte $NextI$ und $NextJ$ werden dann wie $StartPoint$ um die zwei Schlüssel $NeighbourI$ und $NeighbourJ$ erweitert. Die Punkte sind dann wie folgt definiert und werden in die Liste $SortedPoints$ gespeichert. die auf die Koordinateninformation beschränkte Version der beiden Punkte, wird in die $CheckList$ gespeichert. Damit gelten $NextI$ und $NextJ$ als sortiert.

$$\begin{aligned}
NextI = \{<|&CoordI \to i - \text{Koordinate}, CoordJ \to j - \text{Koordinat},\\
&CellI \to i - \text{Zelle}, CellJ \to j - \text{Zelle}, NeighbourI \to 2, NeighbourJ \to 1| >\}\\
NextJ = \{<|&CoordI \to i - \text{Koordinate}, CoordJ \to j - \text{Koordinat},\\
&CellI \to i - \text{Zelle}, CellJ \to j - \text{Zelle}, NeighbourI \to 1, NeighbourJ \to 2| >\}
\end{aligned}$$

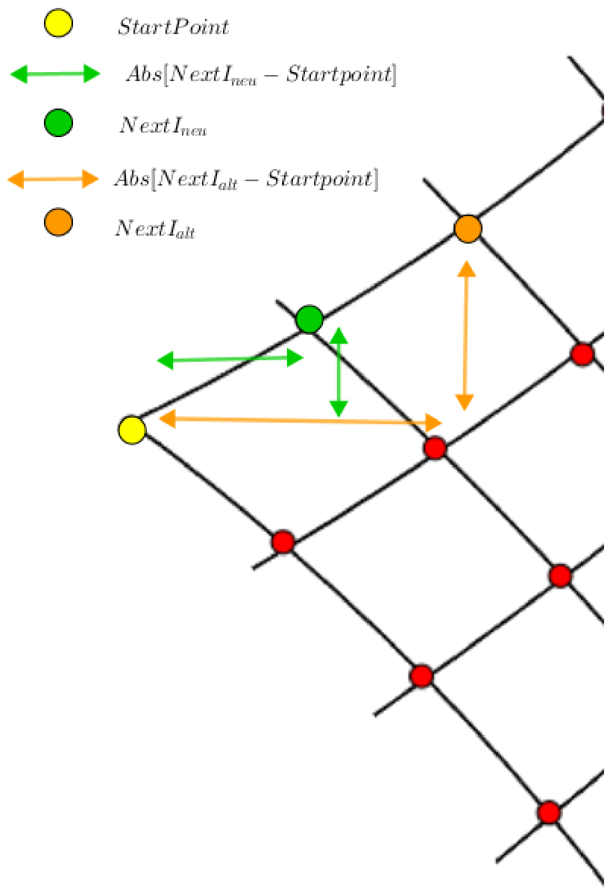


Abbildung 7.4: Um zu überprüfen, ob der momentan $NextI$, in der Abbildung der orange Punkt, wirklich der nächste Punkt ist, wird sein Abstand zum Starpunkt in j- und i- Richtung, mit den anderen noch möglichen Punkten verglichen. Gibt es einen dessen beide Abstände kleiner sind, wie beispielsweise der grüne Punkt, so wird dieser zum neuen $NextI$ bestimmt.

Mit den Punkten $StartPoint$, $NextI$ und $NextJ$ können die ersten Richtungsvektoren in i-Richtung mit $NextIDir = NextI - StatPoint$ und j- Richtung mit $NextJDir = NextJ\, StatPoint$ gebildet werden. Anhand der Richtungsvektoren werden Suchbereiche definiert, in welchen der jeweils nächste Punkt in entsprechender Richtung i oder j gesucht werden soll.

Für beide Richtungen wird die Anzahl der möglichen Punkte begrenzt. Es werden zwei Listen $IList$ und $JList$ angelegt. In $IList$ werden alle Punkte gespeichert welche die Zeleen Schlüssel $CellI_{min}$ bis $CellI_{max}$ und $CellJ_1$ und $CellJ_2$ gespeichert haben. In Abbildung 7.5 entspricht das dem Vertikalen blauen Bereich. In $JList$, kommen alle Punkte deren Zellen Schlüssen $CellJ_{min}$ bis $CellJ_{max}$ und $CellI_1$ und $CellI_2$ gespeichert haben. Was dem horizontalen blauen Bereich in Abbildung 7.5 entspricht.

Der Algorithmus dann nach folgendem Schema vor. Zu erst wird die Untere Schachbrettkante vervollständigt. Sprich alle Punkte de Später die Schlüssen $NeighbourI = 1$ und $NeighbourJ = j_{alle}$ haben sollen.

Neben dem Richtungsvekor $NextJDir$ wird noch ein Pufferwert $ProportionI$ deklariert, welcher den i-Koordinatenabstand zwischen $StartPoint$ und $NextJ$ beinhaltet. Dieser Pufferwert wird in positiver sowie in negativer Richtung um den Richtungsvektor definiert. In Abbildung 7.1 ist dieser Puffer als Blauer Kegel vor den Punkten zu sehen. Dieser Kegel definiert den Suchbereich von einem bereits bekannten Punkt zu einem noch unbekannten Punkt.

$$NextJDir = NextI - StartPoint$$
$$ProportionI = NextI[CoordI] - StartPoint[CoordI]$$

In Abbildung 7.5 auf dem linken Bild wird der Richtungsvektor $NextJDir$ als Pfeil vom gelben $StartPoint$ zum roten $NextJ$ dargestellt, der Richtungsvektor wird noch um einen Pufferwert vergrößert. Die Pfeile welche in i-Richtung von $NextJ$ aus ausgehen, ist der Abstand $ProporionI$, welcher auf die i-Koordinate von $NextJ$ einmal addiert und einmal subtrahiert wird. Der Bereich den die Vektoren aufspannen bildet den Suchbereich für den auf $NextJ$ folgenden Punkt.

Für die weitere Such nach Punkten entlang der Kante wird in einer Schleife $NextJ$ zu neuen $StartPoint$. Der zuvor definierte Suchbereich wird vor den neuen $StatPoint$ definiert und nach einem neuen $NextJ$ gesucht. In Abbildung 7.6 ist dieser Vorgang nochmal veranschaulicht. Wird ein Punkt in diesem Bereich entdeckt, so wird dieser zum neuen $NextJ$, sofern er sich noch nicht in der $CheckList$ befindet. Tritt der Fall ein, dass ein Punkt bereits in der $CheckList$ sich befindet, so wird dieser ignoriert. Der neu entdeckte Punkt wird zu $NextJ$. $NextJ$ wird dann wieder um die zwei Schlüssel $BeighbourI$ und $NeighbourJ$ erweitert und in die $SortedPoints$ Liste gespeichert und dessen verkürzte Fassung wird in die $CheckList$ geschrieben.

Anhand des neuen $StartPoints$ und des neuen $NextJ$ werden der Richtungsvektor $NextJDir$ und $ProportionI$ neu berechnet. $NextJ$ wird wieder zum neuen $StartPoint$ und die Suche wird dementsprechend weiter geführt, bis kein Punkt mehr gefunden wird. Alle Punkte werden als $Asssociation$ mit der entsprechenden Erweiterung der Schlüsselwerte in $SortedList$ und $CheckList$ eingetragen.

Bevor die Suche entlang der beiden außen Kanten als beendet gilt, treten noch zwei Sicherheitsfunktionen in Kraft. Die erste wird als $Saftylist$ bezeichnet. Wird beispielsweise kein weiterer Punkt bei der Detektierung der außen Kanten innerhalb von $JList$ oder $IList$ gefunden, so wird der Suchbereich kurzfristig erweitert. Um den letzten gefundenen Punkt $NextJ$ oder auch $NextI$ werden, sofern vorhanden alle noch nicht abgesuchten Zellen um die Punkte herum abgesucht. Gibt es in einer darum liegenden Zelle noch einen Punkt, der in den Suchbereich fällt, wird dieser noch mit aufgenommen.

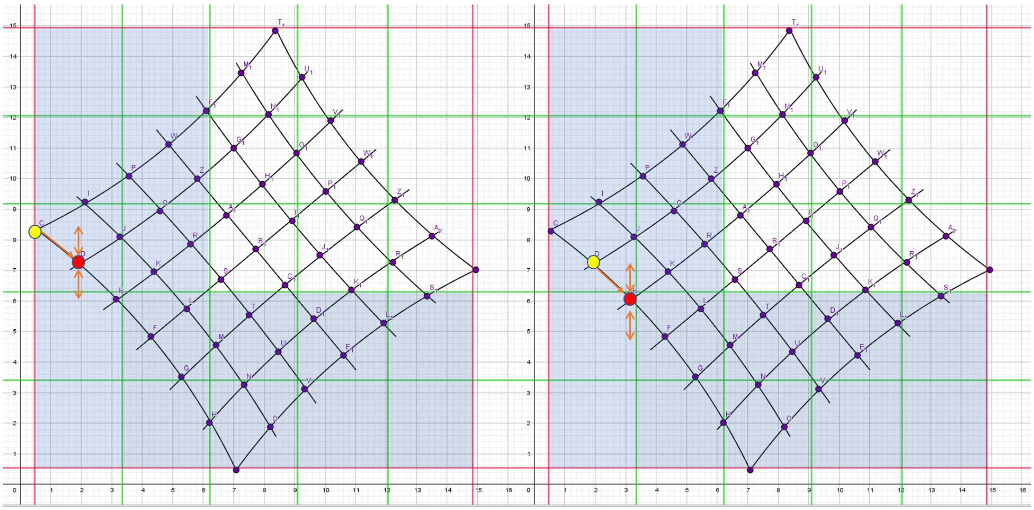


Abbildung 7.5: Im linken Bild ist der Startpunkt in gelb dargestellt und der bereits gefundenen Punt $NextJ$ ist in rot eingefärbt. Anhand der beiden Punkte wird ein Richtungsvektor definiert und zwei Pufferwerte oberhalb und unterhalb. Diese bilden einen Suchbereich. Im rechten Bild wurde $NextJ$ als neuer Startwert festgelegt. Der zuvor definierte Suchbereich wird vor den neuen Startpunkt gesetzt und es wird nach einem potentiellen nächsten $NextJ$ gesucht.

Die Funktion der $SaftyList$ kommt beispielsweise genau dann zum Einsatz, wenn die Punkte eines Schachbretts wie in 7.7 sortiert werden. In der linken Abbildung ist zu sehen, dass der Brreich der $IList$ endet, es aber noch weitere Punkte gibt. Auf der rechten Abbildung ist dann zu sehen, wie der rot hinterlegte Bereich noch hinzugenommen wird und die restlichen Punkte auch noch gefunden werden.

Ist die unterste Kante in j-Richtung bestimmt, so wird die nächste Reihe vervollständigt, beginnend am Punkt $NextI$. $NextI$ ist der neue $StartPoint$ von wo aus die nächste Reihe in j-Richtung vervollständigt wird. Bevor die nächste Reihe vervollstädnigt werden kann, muss zuerset der nächste Punkt $NextI$ auf der äußeren Kante bestimmt werden. Wie in den Abbildung 7.6 und 7.7 zu sehen, basiert die Suche nach $NextI$ auf dem gleichen Verfahren wie die Suche nach $NextJ$.

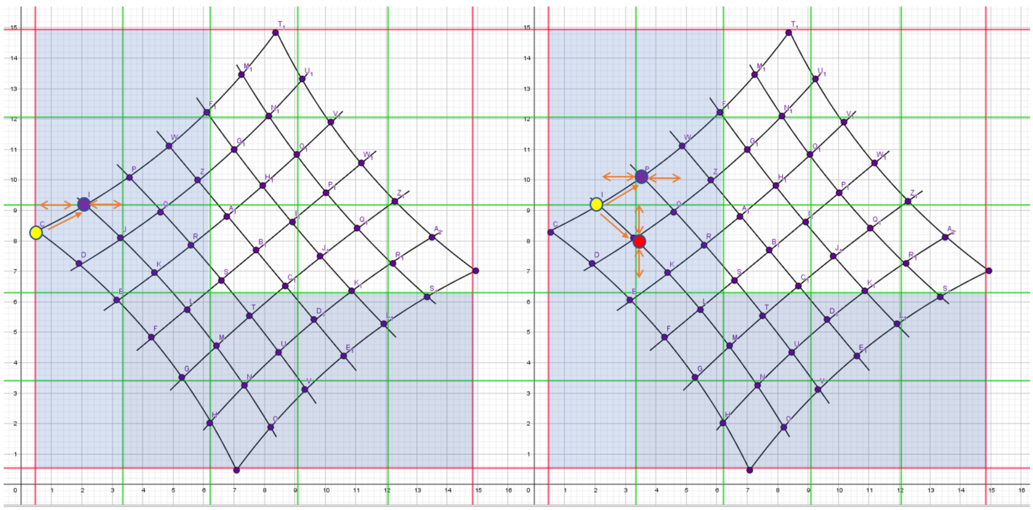


Abbildung 7.6: Die Suche nach dem nächsten $NextI$ basiert auf dem selben Verfahren wie die Suche nach $NextJ$. Sobald ein neuer $NextI$ gefunden wurde, wird die Reihe in j-Richtung vervollständigt, bevor es mit dem nächsten $NextI$ weiter geht

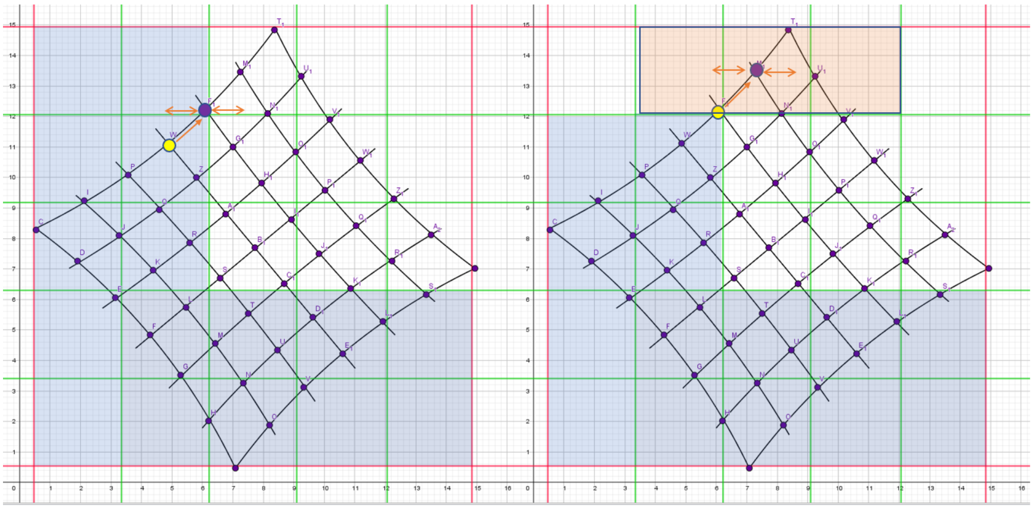


Abbildung 7.7: Wird der kein weiterer Punkt innerhalb von $IList$ oder $JList$ gefunden, Wird ein kleiner Bereich, hier in rot hinterlegt, abgesucht, ob es doch noch potentielle nächste Punkte gibt, die aufgrund der Lage des Schachbretts nich in $IList$ oder $JList$ mit enthalten waren

Zuvor wurde erwähnt, dass es neben $SaftyList$ noch eine andere Sicherheitsfunktion namens gibt. Hierzu sei nochmal erwähnt, dass der hier beschriebene Sortierungsalgorithmus auf einem bereits bestehenden Algorithmus aufbaut. Dieser Algorithmus detektiert Die Eckpunkte eines Schachbretts und speichert die Bildkoordinaten in eine Liste. Bei der Detektion kann es vorkommen, dass Punkte, durch beispielsweise Rauschen oder Bildunschärfen, nicht erkannt werden und somit Lücken im Schachbrettgitter entstehen. Ist dies der Fall würde der Sortierungsalgorithmus wie er bis jetzt ist, bei einer Lücke mit der Suche aufhören, da kein Punkt im definierten Suchbereich gefunden werden kann. Weitere Punkte welche sich je nach dem hinter der Lücke befinden, würde so nicht mehr gefunden und sortiert werden können. Diese Punkte würde nicht in der Liste $SortedList$ landen und der Algorithmus würde dich Nachricht ausgeben, dass die Sortierung unvollständig sei.

Um dem entgegen zu wirken wurde die Sicherheitsfunktion *PlaceSyntheticPoint* entwickelt. Sollte vorerst kein Punkt im Suchbereich entdeckt, werden, so setzt die Sicherheitsfunktion einen synthetischen Punkt und sucht ausgehend von diesem synthetischen weiter nach Punkten. Sollte ein Punkt nach dem synthetischen Punkt gefunden werden, so bleibt der synthetische Punkt bestehen und die Suche wird normal fortgesetzt. Der synthetische Punkt wird nicht in die Liste *SortedList* mit aufgenommen, da er später nicht als möglicher korrespondierender Punkt in betrachtet gezogen werden soll. Er dient lediglich dazu alle real vorhandenen Punkte zu finden und richtig zu sortieren. Wird nach dem setzten des synthetischen Punkt kein weiterer Punkt gefunden, gilt die Suche in der Reihe beziehungsweise Spalte für beendet und der synthetische Punkt wieder wieder gelöscht.

In Abbildung 7.8 und 7.9 wird ein Schachbrett aus zwei unterschiedlichen Blickwinkeln dargestellt. Die roten Punkte markieren die Eckpunkte, deren Koordinaten an den Sortierungsalgorithmus übergeben werden. In den Abbildungen 7.10 und 7.11 werden alle Punkte, welche in der Liste *SortedPoints* gespeichert wurden in einem Plot ausgegeben. Es wird gesagt, dass alle Punkte deren Schlüssel $NeighbourI = 3$ ist, in grün dargestellt werden sollen. Das Ergebnis dieser Abfrage ist ebenfalls in den Abbildungen 7.10 und 7.11 zu sehen. Der entstandene Sortierungalgorithmus ist also in der Lage aus Bildern zweier Schachbretter gleiche Eckpunkte und somit korrespondierende Punkte zu detektieren. Soll ein einzelner Punkt Abgefragt wollen, so für die Schlüssel $NeighbourI$ und $NeighbourJ$ jeweils ein Wert abgefragt werden.

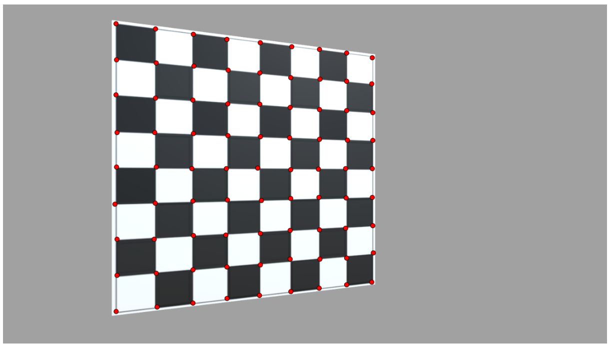


Abbildung 7.8: Die Kamera,welche das Schachbrett abbildet steht links versetzt zum Schachbrett und ist rotiert

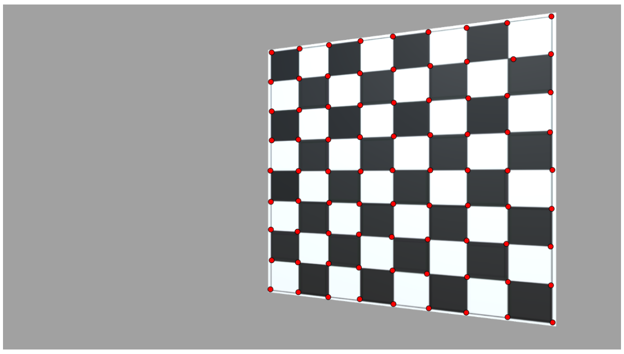


Abbildung 7.9: Die Kamera, welche das Schachbrett abbildet steht rechts versetzt zum Schachbrett und ist rotiert

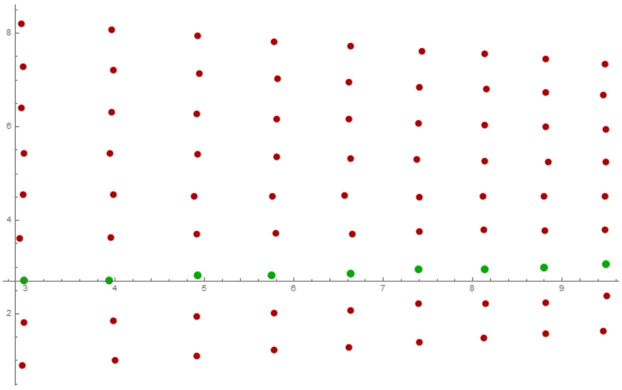


Abbildung 7.10: Die Abbidlung zeigt das Ergebnis der Sortierung des linken Schachbretts. In grün ist die Reihe an Punkten zu sehen, welche den Schlüssel $NeighbourI = 3$ haben.

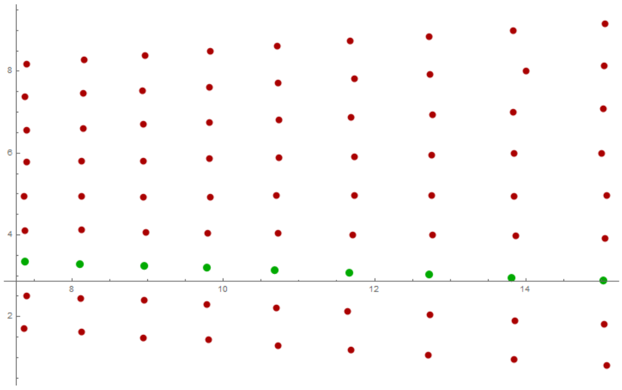


Abbildung 7.11: Die Abbidlung zeigt das Ergebnis der Sortierung des rechten Schachbretts. In grün ist die Reihe an Punkten zu sehen, welche den Schlüssel $NeighbourI = 3$ haben.

7.2 Resultate bei stark verzerrten Schachbrettern

Der entstandenen Sortierungsalgorithmus wird an stark perspektivisch Verzerrten oder durch Bildfehler wie Verzeichnungen betroffenen Schachbrettern getestet. Die Möglichkeit herauszufinden, welche Eckpunkte in einem stark verzerrten Bild eines Schachbretts in eine Reihe oder Spalte gehören, kann bei der mathematischen Entzerrung der Bilder von Nutzen sein. Dieser Vorteil war mit ein Grund warum der Algorithmus auf Grundlage von stark verzerrten Schachbrettbilder entwickelt wurde.

In den folgenden Beispielen sieht man jeweils das Originalbild des Schachbretts und neben dran die Ausgabe des Algorithmus. Der Plot zeigt alle Punkte an, welche in $SortedPoints$ aufgenommen wurden. Alle Punkte, welche als Schlüssel $NeighbourI = 3$ haben, wurden im folgenden grün eingefärbt, die restlichen Punkte sind rot gefärbt.

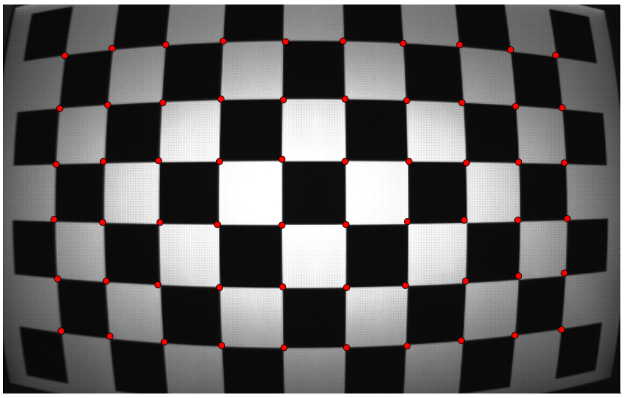


Abbildung 7.12: Schachbrett mit Tonnenverzeichnung

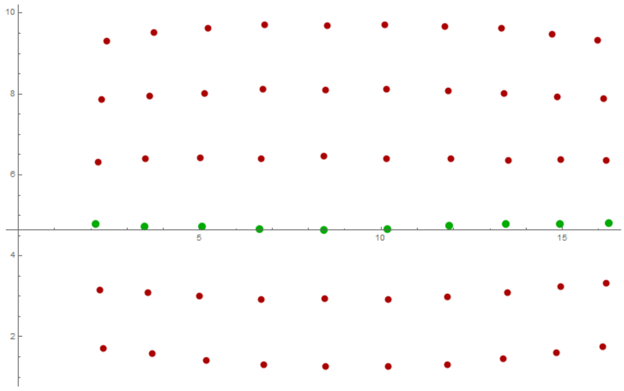


Abbildung 7.13: Ergebnis des Sortierungsalgorithmus.

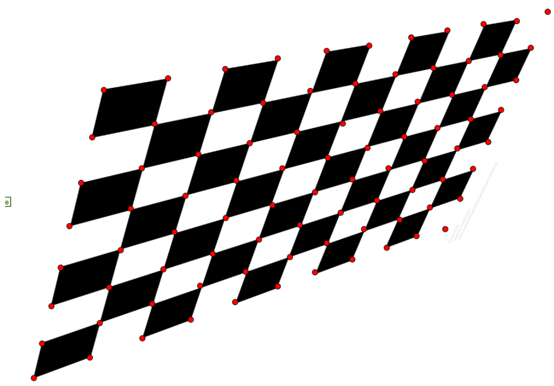


Abbildung 7.14: Perspektivisch stark verzerrtes Schachbrett

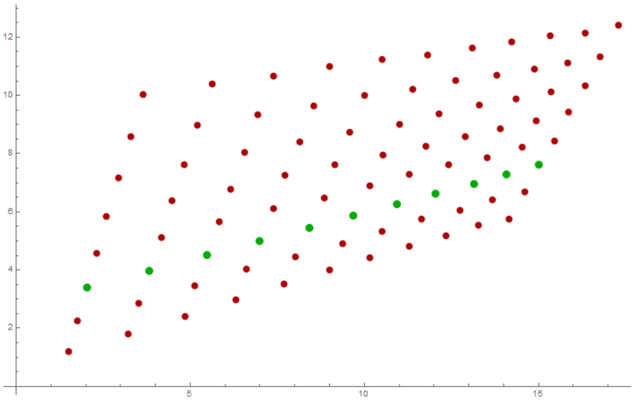


Abbildung 7.15: Ergebnis des Sortierungsalgorithmus.

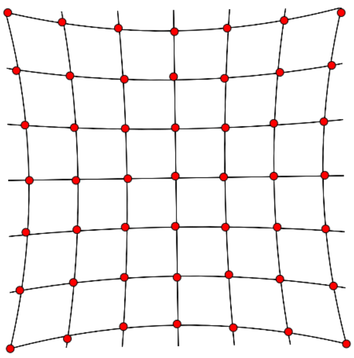


Abbildung 7.16: Schachbrett mit Kissenverzeichnung

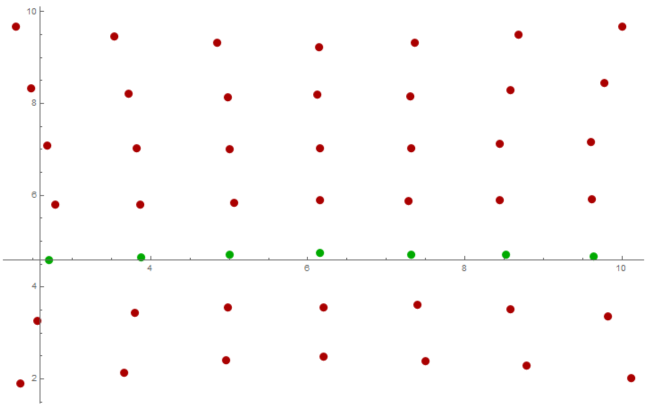


Abbildung 7.17: Ergebnis des Sortierungsalgorithmus.

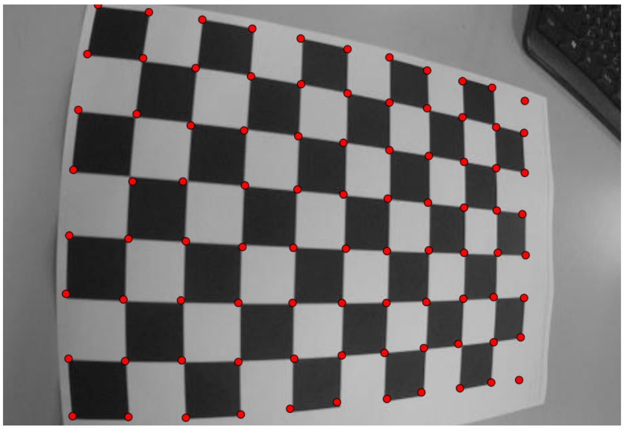


Abbildung 7.18: Perspektivisch verzerrten Schachbrett mit Tonnenverzeichnung

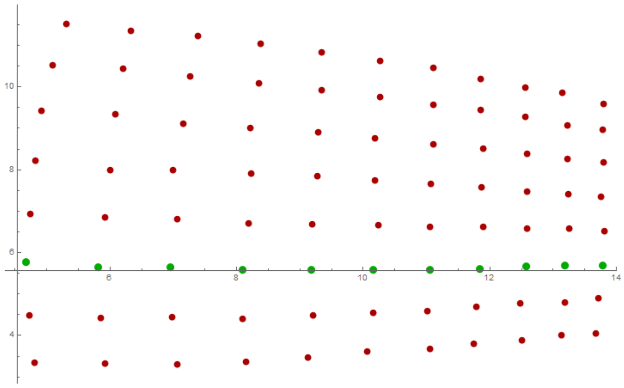


Abbildung 7.19: Ergebnis des Sortierungsalgorithmus.

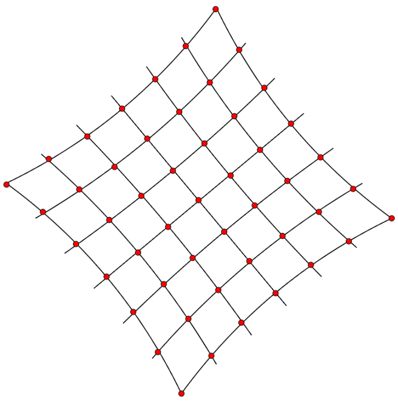


Abbildung 7.20: Bild eines Tonnenförmig verzeichnetem leicht perspektivisch verzerrtem Schachbretts

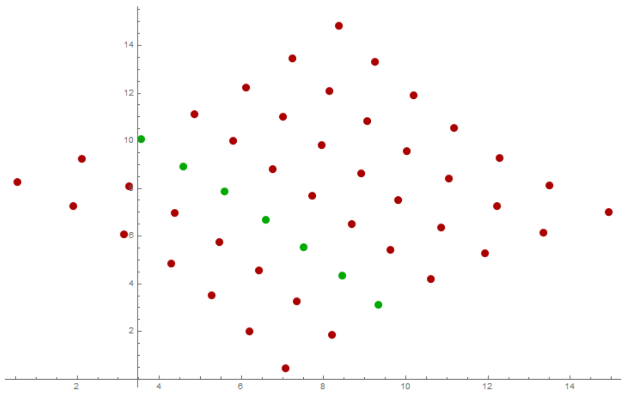


Abbildung 7.21: Algorithmisch detektierte Linie der dritten i-Reihe

8 Fazit

Anhang

Eidesstattliche Erklärung

Hiermit erkläare ich, dass ich die vorliegende Arbeit selbstständig und nur unter Zuhilfenahme der ausgewiesenen Hilfsmittel angefertigt habe. Darüberhinaus erkläre ich, dass diese Abschlussarbeit nicht auch nicht auszugsweise, bereits für andere Prüfungen angefertigt wurde.

Furtwangen, den 29.Mai 2018

Abbildungsverzeichnis

Literaturverzeichnis

[1] Zhengyou Zhang Gang Xu. *Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach.* Springer-Science and Buissness Media, 1996.

[2] Lutz Priese. *Computer Vision, Einführung in die Verarbeitung und Analyse digitaler Bilder.* Springer-Verlag Berlin Heidelberg, 2014.

[3] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in computer vision.* Cambridge, 2004, Second Edition.

[4] Ferid Bajrmovic. *Self- Calibration of Multi- Camera Systems.* Logos Verlag Berlin GmbH, 2010.

[5] Tomas Pajdla. *Elements of Geometry for Computer Vision.* "http://people.ciirc.cvut.cz/ pajdla/", 2013, überarbeitet am 27.2.2017.

[6] Christian Heipke. *Photogrammetrie und Fernerkundung.* 2017 Springer,1. Auflage.

[7] Ramalingam Tardif S.Gasparini J.Barreto R.Sturm, S. Camera models and fundamental concepts used in geometric computer vision. Mitsubishi Electric Research Laboratories, 2011.

[8] Rolf Martin Ekbert Hering. *Photonik, Grundlagen, Technologien und Anwendung.* Springer-Verlag Berlin Heidelberg, 2006.

[9] Dipl.-Ing. Martin Roser. *Modellbasierte und positionsgenaue Erkennung von Regentropfen in Bildfolgen zur Verbesserung von viedeobasierten Fahrerassistenzfunktionen.* 1986, 1994 Springer Basel AG, KIT Scientific Publishing.

[10] Christoph Stiller Thao Dang, Christian Hoffmann. Continuous stereo self-calibration by camera parameter tracking.

[11] Branislav Micusik. *Two-View Geometry of Omnidirectional Cameras.* Dissertation, Technische Universität Prag.

[12] Zhengyou Zhang. *Epipolar Geometry*, pages 247–258. Springer US, Boston, MA, 2014.

[13] Zhengyou Zhang. Determining the epipolar geometry and its uncertainty: A review. Received July 16, 1996; Accepted February 13, 1997.

[14] K.A. Semendjajew I.N. Bronstein. *Taschenbuch der Mathematik*, volume 5. Auflage. First Published 1962.

[15] Sascha jockel. *3-dimensionale Rekonstruktion einer Tischszene aus monokularen Handkamera-Bildsequenzen im Kontext automotiver Serviceroboter.* Dissertation, Fakultät für Mathematik, Informatik und Naturwissenschaften, Universität Hamburg.

[16] Richard I. Hartley. In defence of the 8-point algorithm. GE-Corporate Research and Development, Schenectady, NY, 12309.

[17] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[18] Norbert Köckler Hans Rudolf Schwarz. *Numerische Mathematik.* 2011, Springer Verlag, 8. Auflage.

[19] Daniel Scholz. *Numerik interaktiv, Grundlagen verstehen, Modelle erforschen und Verfahren anwenden mit taramath.* 2016, Sringer Verlag.

[20] LongQuan. *Image Based Modeling*, volume 1. Auflage. Springer US, 2010.

[21] Bernd KITT. *Effiziente Schätzung dichter Bewegungsvektorfelder*, volume Band 027. KIT Scientific Publishing, 2013.

[22] Wolfram Research, Inc. Mathematica, Version 11.1.1. Champaign, IL, 2018.

[23] Olaf Dössel. *Bildgebende Verfahren in der Medizin*, volume 2. Auflage. Springer-Verlag Berlin Heidelberg, 2016.

[24] Emmanuel Habets Nicolas Paparoditis Xiaozhi Qu, Bahman Soheilian. Evaluation of sift and surf for vision based localization. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLI-B3, 2016.

[25] Antin van den Hengel Darren Gawey Wojciech Chojnacki, Michael J. Brooks. Revisiting hartley's normalized eight-point-algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence,Volume: 25, 2003.

[26] Jarosław Bylina Anna Pyzara, Beata Bylina. The influence of a matrix condition number on iterative methods convergence. Proceedings of the Federated Conference on Computer Science and Information Systems, pp. 459–464, 2011.

[27] James Demmel Dinesh Manocha. Algorithms for intersecting parametric and algebraic curves 1. ACM Transactions on Graphics (TOG), Volume 13 Issue 1, Pages 73-100, 1994.

[28] Jürgen Adamy Christian Voigt. *Formelsammlung der Matrizenrechnung.* Oldenburg Verlag München Wien, 2007.

[29] Luca Irsara Andrea Fusiello. Euclidean epipolar rectification of uncalibrated images. Eurac researc, IT.

[30] Carlos Villagrá Arnedor Antonio Javier Gallego Sanchez, Rafael Molina Carmona. *Scene reconstruction and geometrical rectification from stereo images.* Januar 2005,uploaded by Antonio Javier Gallego Sánchez on 21 May 2014, ResearchGate.

[31] Charles Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Vol.1, pages 125–131, June 23-25, 1999. Fort Collins, Colorado, USA, 1999 Errors corrected on June 6, 2001.

[32] MathWorks. Mathworks documentation, rectify stereo images.

[33] Richard I. Hartley. Euclidean reconstruction from uncalibrated views. G.E. CRD, Schenectady, NY, 12301.

[34] Dongqing Li, editor. *Encyclopedia of Microfluidics and Nanofluidics*, pages 999–999. Springer US, Boston, MA, 2008.

[35] William T. Vetterling Brian P. Flannery William H. Press, Saul A. Teukolsky. *Numerical Recipes in Fortran 77 The Art Of Scientific Computing.* Copyright Numerical Recipes Software 1986, 1992, 1997 All Rights Reserved., Reprinted with corrections 1997, Volume 1 of Fortran Numerical Recipes.

[36] James Demmel Dinesh Manocha. Algorithms for interesting parametric and algebraic curves 1: Simple intersections. ACM Transactions of Graphics:73–100, 1994.