



**Westfälische
Hochschule**

Gelsenkirchen Bocholt Recklinghausen
University of Applied Sciences



Fraunhofer

IAIS

Westfälische Hochschule Gelsenkirchen

Fachbereich Informatik und Kommunikation
Angewandte Informatik
Lehr- und Forschungsgebiet Autonome Systeme

Fraunhofer-Institut für Intelligente Analyse- und Informationssysteme

Abt. Adaptive Reflective Teams

Bachelorarbeit

**Fusion von 3D-Laserscanner-Daten mit 2D-
Thermal-Bilddaten**

**Tom-Marvin Liebelt
200923681**

Erstprüfer:
Zweitprüfer:

Prof. Dr.-Ing. Dipl. Inform. Hartmut Surmann
Dipl.-Inform. Rainer Worst

Datum und Unterschrift des Erstprüfers

Hiermit versichere ich, die Arbeit selbstständig angefertigt und keine anderen als die angegebenen und bei Zitaten kenntlich gemachte Quellen und Hilfsmittel benutzt zu haben.

Datum und Unterschrift des Verfassers:

Diese Bachelorthesis ist ein Prüfungsdokument. Eine Verwendung zu einem anderen Zweck ist mit dem Einverständnis von Verfasser und Prüfern erlaubt.

Abstract

Wärmebildkameras finden mehr und mehr Einzug in die Welt der Robotik, doch ein Wärmebild allein ist selten aussagekräftig, weshalb die Fusion mit weiteren Sensordaten gebraucht wird. In dieser Arbeit werden mehrere Aufnahmen einer Wärmebildkamera mit der Punktwolke eines 3D-Laserscanners kombiniert, um eine 3-dimensionale Wärmebildaufnahme zu erzeugen. Eine Kalibrierung der Wärmebildkamera mithilfe eines eigens dafür entwickelten Kalibrierungsmusters sorgt für ein verzerrungsfreies Ausgangsbild und liefert die wichtigen Kameraparameter. Weiterhin werden über eine, leicht durch den Nutzer durchzuführende, extrinsische (äußere) Kalibrierung die Aufnahmepositionen bestimmt um zusammen mit den Kameraparametern eine Passgenaue Fusion von Wärmebildern und Punktwolke zu ermöglichen. Das Ergebnis der Fusion ist eine Punktwolke mit den Falschfarben einer Wärmebildkamera und eine Verbindung von den 3D-Punkten der Wolke mit Temperaturwerten in °C.

Schlüsselworte: Wärmebildkamera, 3D-Laserscanner, Kalibrierungsmuster, 3D-Wärmebildaufnahmen

Diese Arbeit wurde von der Europäischen Kommission unter der Nummer **FP7-247870-NIFTI** finanziert.

Inhaltsverzeichnis

Abstract	1
Inhaltsverzeichnis	3
1 Einleitung	6
1.1 Motivation	6
1.2 Problemstellung	8
1.2.1 Kernprobleme	8
1.2.2 Teilprobleme	8
1.2.3 Nebenprobleme	8
1.3 Aufbau der Arbeit	9
2 Stand der Technik	11
2.1 Allgemein	11
2.2 Intrinsische Kamerakalibrierung	12
2.3 Extrinsische Kamerakalibrierung	13
2.4 Kalibrierung mit einer Thermalkamera	14
2.5 ThermalMapper	14
3 Systemdesign	15
3.1 Der 3D-Laserscanner und die Thermalkamera	15
3.2 Das Kalibrierungsmuster	16
3.2.1 Erwärmen eines gedruckten Musters	16
3.2.2 Aufbau des Musters mit Leuchtmitteln	17
3.2.3 Aufbau des Musters mit Widerstandsdrähten	19
3.2.4 Ausgewähltes Verfahren: Muster mit Widerstandsdrähten	19
3.3 Hardwareaufbau	20
3.4 Systemaufbau	21
3.5 Entwicklung	23
3.5.1 Betriebssysteme	23
3.5.2 Frameworks und Tools	23
4 Thermalkamera	24
4.1 Infrarotstrahlung	24
4.2 Thermalkamera	25
4.3 thermoIMAGER TIM 160 unter ROS	26
4.3.1 Erweiterungen	27
5 Intrinsische Kamerakalibrierung	28

5.1 Intrinsische Parameter	28
Exkurs: Die Verzerrung (Verzeichnung) einer Kamera.....	29
5.2 Intrinsische Kalibrierung.....	30
5.3 Korrektur eines Thermalbildes mit den intr. Parametern	33
5.4 Intrinsische Parameter in ROS	34
6 Extrinsische Kalibrierung.....	36
6.1 Extrinsische Parameter	36
6.2 Laserscanner und Punktwolken	38
6.3 Extrinsische Kalibrierung	39
6.3.1 Bestimmung der Kameraausrichtung	41
6.3.2 Aufnehmen von Kamerabildern alle x°	43
6.3.3 Markieren des Kalibrierungsmusters in der Punktwolke	44
6.3.4 Markieren des Kalibrierungsmusters auf einem Kamerabild.....	45
6.3.5 Abbilden der 3D-Punkte auf die Bildebene.....	46
6.3.6 Berechnung eines Fehlerwertes	47
6.3.7 Erstellung der launch-Datei.....	49
7 Datenfusion	50
7.1 Ablauf der Fusion	50
7.2 Beschreibung der Teilaufgaben	51
7.2.1 Sammeln der Daten (Vorarbeit für (1) und (2)).....	51
7.2.2 Thermalaufnahme ermitteln (3.1).....	52
7.2.3 Gegenstück des 3D-Punktes im Kamerabild finden ((3.1) bis (3.4)).....	54
7.2.4 3D-Punkte entsprechend der Voreinstellung einfärben (3.5)	55
7.2.5 Punktwolke und Thermaldaten veröffentlichen (Punkt 5).....	57
8 Darstellung der Thermal-Punktwolke	58
9 Evaluierung	59
9.1 Bekannte Probleme	59
9.1.1 Rotationseinheit	59
9.1.2 Punktwolken-Erzeugung	60
9.1.3 Fusion der Thermalbilder mit der Punktwolke.....	61
9.2 Passgenauigkeit der Fusion	62
10 Zusammenfassung und Ausblick.....	63
10.1 Zusammenfassung	63
10.2 Verbesserungsmöglichkeiten	64
10.3 Ausblick	65
11 Anhang	67
11.1 Einrichtung für den ersten Start.....	67

11.2 Hinweise für die Benutzung der Software.....	68
11.3 QT-Beispiel.....	69
11.4 Aufwandsanalyse	70
12 Bildquellen.....	71

1 Einleitung

1.1 Motivation

Sehen ist für uns Menschen der wichtigste Sinn, wenn es darum geht sich auf diesem Planeten zurechtzufinden. Viele der Reize, die täglich auf uns einwirken, sind visueller Natur. Aber der für uns sichtbare Teil der elektromagnetischen Strahlung ist relativ klein, bedenkt man welcher Strahlung wir im Alltag ausgesetzt sind. Mit elektromagnetischer Strahlung haben wir ständig zu tun ohne dass wir sie bemerken. Darunter ist auch eine Strahlung, die wir zwar nicht sehen dafür aber spüren können: die Wärmestrahlung.

Wärmestrahlung können wir ohne technische Hilfsmittel wahrnehmen, jedoch ist unser Wärmeempfinden oft nur sehr vage, wenn wir die Position einer Wärmequelle bestimmen möchten. Oft ist es aber nötig Wärmequellen genau zu orten und auch kleinste Temperaturunterschiede sichtbar zu machen und da ist kein Hilfsmittel besser geeignet als die Wärmebildkamera.



Wärmebildaufnahme einer ungedämmten Hauswand.¹

Eine Wärmebildkamera, ausgenommen spezieller Modelle, ist sehr universell einsetzbar. Zum Beispiel die Untersuchung von Gebäudefassaden auf Fehler in der Wärmedämmung wäre ohne eine Wärmebildkamera nur mit großem Aufwand möglich. Ein anderes Beispiel ist die Verwendung von Wärmebildern zur Auffindung von Verschütteten, denn hier kann jede Minute, in der die Rettungskräfte auf der Suche nach Opfern sind, entscheidend sein. Um nur zwei Beispiele zu nennen bei denen man die Wärmebildkamera nicht mehr missen möchte.

¹ Wikipedia-Artikel zu Wärmebildkamera: <http://de.wikipedia.org/wiki/W%C3%A4rmebildkamera> (27.03.2013)



Einsatz von Wärmebildkameras durch Rettungskräfte um die im Gebäude eingeschlossenen Verletzten zu finden und zu bergen.²

Eine Wärmebildkamera nimmt ihre Umgebung in 2D auf, die Interpretation der Aufnahme kann aber nur dann ein sinnvolles Ergebnis liefern, wenn bekannt ist auf was die Kamera gerichtet war. In der Regel wird eine Wärmebildkamera von einem Menschen bedient, der die Umgebung auch mit seinen eigenen Augen sehen kann. Problematisch wird es wenn die Umgebung nicht bekannt ist, die Aufnahme also aus der Ferne ausgelöst wird. Allein aus der Thermalaufnahme heraus lässt sich nur schwer bestimmen was aufgenommen wurde. Ein Mensch, der hinter der Kamera steht verwendet hauptsächlich die Wahrnehmung der räumlichen Tiefe zur Interpretation des Wärmebildes, also ist es nur logisch eine 3D-Aufnahme mit der Thermalaufnahme zu kombinieren.

Um noch einmal auf das Beispiel mit den Rettungskräften zurückzukommen, in Katastrophengebieten gibt es viele Situationen in denen sich Rettungskräfte bereits bei der Gebietserkundung in Lebensgefahr begeben. Die 3D-Thermalaufnahmen machen eine ferngesteuerte oder automatische Erkundung durch Roboter möglich, da die Aufnahmen für sich genommen schon aussagekräftig genug für eine Einschätzung der Situation sind.

² Artikel "Wärmebildkamera bewährt sich" der Feuerwehr Fleisbach: <http://feuerwehr-fleisbach.de/uploads/presse/W%C3%A4rmebildkamera%20bew%C3%A4hrt%20sich.pdf> (27.03.2013)

1.2 Problemstellung

1.2.1 Kernprobleme

Es müssen mehrere 2D-Thermalaufnahmen mit den 3D-Informationen eines Laserscanners kombiniert werden um eine 3-dimensionale Thermalaufnahme zu erhalten. Die Zusammenführung der Daten soll, ausgenommen einmaliger Konfigurationen, automatisch ablaufen. Um eine einfache weitere Verwendung der Aufnahmen zu gewährleisten, müssen so weit wie möglich die vorgegebenen hardwareunabhängigen Datentypen verwendet werden.

1.2.2 Teilprobleme

Für die Realisierung des Systems sind mehrere Teilproblematiken zu lösen:

- Erweiterung des Thermalkamera-Treibers
- Durchführung einer intrinsischen Kalibrierung der Kamera
- Implementierung einer extrinsischen Kalibrierung von Kamera ↔ Punktwolke
- Realisierung der Datenfusion
- Erstellung eines Visualisierungsprogramms

1.2.3 Nebenprobleme

Für diese Arbeit sind der Erwerb und die Vertiefung einiger Kenntnisse notwendig:

- Einarbeitung in den Aufbau der Point Cloud Library für den Umgang mit den Punktwolken
- Aneignung von Wissen über QT zur Erstellung der GUIs
- Einarbeitung in die Funktionsweise von tf zur Anwendung der Transformationen
- Sammlung von Erkenntnissen im Bereich der intrinsischen und extrinsischen Parameter

1.3 Aufbau der Arbeit

Einleitung

Dieses Kapitel soll einen allgemeinen Einstieg in die Thematik dieser Arbeit geben.

Stand der Technik

Dieses Kapitel beschreibt den aktuellen Stand der Technik zum Bearbeitungszeitpunkt dieser Arbeit.

Systemdesign

In diesem Kapitel wird auf das allgemeine Design der Systemarchitektur, der verwendeten Hardware, sowie auf die für die Entwicklung benutzten Tools und Betriebssysteme eingegangen. Zudem werden unterschiedliche Möglichkeiten eines Kalibrierungsmusters für Thermalkameras vorgestellt, welche die intrinsische Kalibrierung möglich machen.

Thermalkamera

Dieses Kapitel beschreibt zuerst allgemein die Infrarotstrahlung um den Einstieg in den darauf folgenden Abschnitt über die Thermografie zu erleichtern. Im zweiten Teil wird konkret auf die verwendete Thermalkamera eingegangen, wobei der Schwerpunkt auf die Integration in ROS gelegt wird. Abschließend werden die Erweiterungen des Thermalkamera-Treibers genannt, die für die weitere Arbeit nötig waren.

Intrinsische Kamerakalibrierung

Dieser Abschnitt beschreibt die Merkmale der intrinsischen Parameter und zeigt den Ablauf der intrinsischen Kalibrierung mithilfe der MATLAB Camera Calibration Toolbox. Zudem wird auf die Verwendung der intrinsischen Parameter in ROS eingegangen.

Extrinsische Kalibrierung

In diesem Kapitel werden die nötigen Schritte für eine erfolgreiche extrinsische Kalibrierung vorgestellt, jedoch nicht ohne zuvor den Sinn der extrinsischen Parameter zu nennen.

Datenfusion

Dieses Kapitel zeigt den Ablauf der Fusion von Thermalbildern mit der Punktwolke mitsamt den Einstellungsmöglichkeiten.

Darstellung der Thermal-Punktwolke

Ein kurzer Abschnitt über das entstandene Visualisierungsprogramm für Punktwolke und Thermaldaten.

Evaluierung

In diesem Abschnitt werden die Ergebnisse der Experimente vorgestellt, die die Eigenschaften und Funktionen des Systems zeigen.

Zusammenfassung und Ausblick

Das letzte Kapitel fasst den Fortschritt dieser Arbeit zusammen und bewertet ihn. Ein abschließender Ausblick gibt Anregungen für weitere Arbeiten bzw. Projekte.

Anhang

2 Stand der Technik

2.1 Allgemein

3D-Thermalbilder können auf verschiedene Arten und mit den unterschiedlichsten Techniken aufgenommen werden. Zum Beispiel können zwei Thermalkameras³ oder eine 3D-Kamera und eine Thermalkamera⁴ benutzt werden. In dieser Arbeit geht es aber um die Erstellung von 3D-Thermalaufnahmen mithilfe eines 3D-Laserscanners und einer Thermalkamera, deshalb wird auf die anderen Techniken nicht weiter eingegangen. Die Messung eines Objekts oder einer Umgebung mit einem Laserscanner ist eine präzise Möglichkeit die Struktur in Erfahrung zu bringen und außerdem sind 3D-Laserscans in der Robotik schon länger in Benutzung, d.h. man kann auf eine sehr große Anzahl an Arbeiten und fertigen Programmen zurückgreifen um ein Projekt auf diesem Gebiet zu realisieren.

Eine Fusion von Thermalbildern mit 3D-Laserscanner-Daten ist hingegen ein ziemlich neues Verfahren und deshalb ist die Anzahl der zur Verfügung stehenden wissenschaftlichen Arbeiten stark beschränkt. Es gibt zwar Arbeiten, in denen 3D-Laserscanner-Daten mit 2D-Bilddaten kombiniert werden⁵ und es gibt auch Arbeiten, die sich mit der Kombination von 3D-Modellen mit Thermaldaten beschäftigen⁶. Doch eine automatische Aufnahme der gesamten Umgebung inklusive Thermaldaten bietet derzeit nur der ThermalMapper⁷. Ein 3D-Aufnahme der Umgebung mit Thermalinformationen, wie sie bei dem ThermalMapper zu sehen ist, ist das Ziel dieser Bachelorarbeit. Da der Quellcode des ThermalMapper nicht zur Verfügung steht und nicht die gleiche Hardware benutzt werden kann, sind Eigenentwicklungen sowie einige Anpassungen an die vorhandene Hardware erforderlich. Zu diesem Zweck wird nachfolgend der Stand der dafür nötigen Technik beschrieben.

³ Surya Prakash, Pei Yean Lee und Antonio Robles-Kelly: "Stereo techniques for 3D mapping of object surface emperatures", QIRT Journal Volume 4 - 1/2007

⁴ Mathew Price, Jeremy Green und John Dickens: "Creating Three-Dimensional Thermal Maps", ROBMECH 2011 - 11/2011

⁵ Richard Hanten: "Fusion von 3D-Laserscanner-Daten mit 2D-Bilddaten", 08/2011

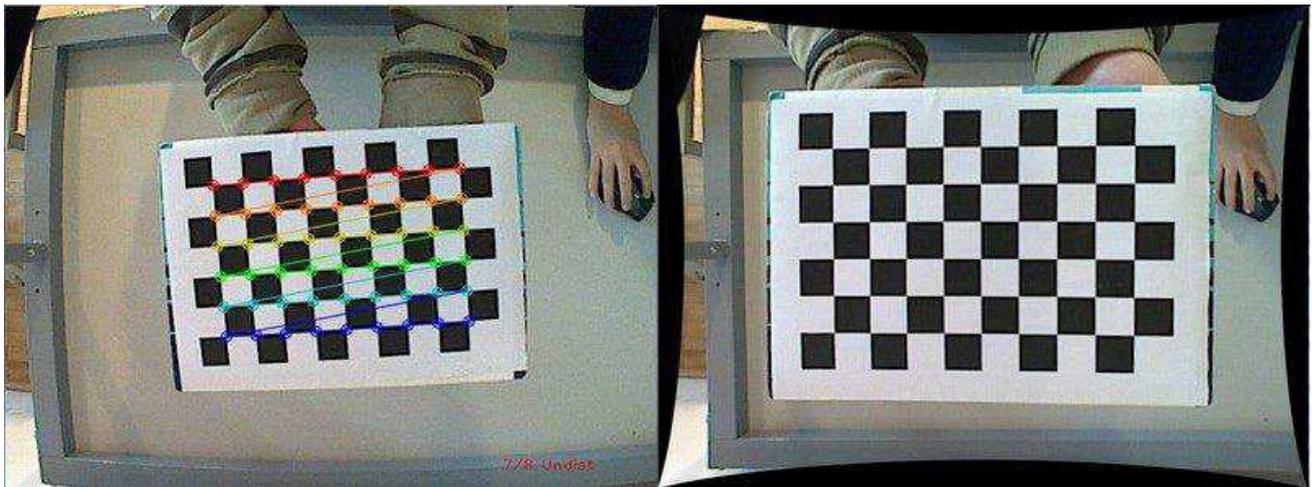
⁶ M. Cabrelles, S. Galcerá, S. Navarro, J. L. Lerma, T. Akasheh und N. Haddad: "INTEGRATION OF 3D LASER SCANNING, PHOTOGRAMMETRY AND THERMOGRAPHY TO RECORD ARCHITECTURAL MONUMENTS", 22nd CIPA Symposium - 10/2009

⁷ Dorit Borrmann, Andreas Nüchter, Marija Dakulovic, Ivan Maurovic, Ivan Petrovic, Dinko Osmankovic und Jasmin Velagic: "The Project ThermalMapper - Thermal 3D Mapping of Indoor Environments for Saving Energy", 2011

2.2 Intrinsische Kamerakalibrierung

Man muss die spezifischen Parameter einer Kamera kennen, wenn man einem Pixel eine 3D-Position zuordnen oder, wie es bei dieser Arbeit der Fall ist, zu einem Punkt aus der Punktwolke das passende Pixel finden will.

Die bekannteste automatische Kamerakalibrierung ist die mit einem Schachbrettmuster. Diese Kalibrierungsmethode ist unter anderem in OpenCV⁸ realisiert, dabei werden die Quadratecken in der Kameraaufnahme eines Schachbrettmusters für die Kalibrierung herangezogen. OpenCV kann über eine weitere Funktion noch exakter die Position der Eckpunkte bestimmen, da hierfür die Eigenschaften solcher, aus Quadraten bestehenden, Muster genutzt werden. Sind die Punkte erst bestimmt können die Verzerrung, Brennweite und andere Kameraparameter über eine weitere Funktion bestimmt werden, der man zusätzlich zu den ermittelten Punkten, die Soll-Koordinaten der Punkte übergibt.



Auf der linken Seite sieht man das noch verzerrte Bild, die Verzerrung erkennt man am besten am Tischrahmen. OpenCV hat bereits die Ecken der Quadrate markiert, die im nächsten Schritt mit den Sollpositionen der Eckpunkte verglichen werden, um die Verzerrung zu bestimmen. Das rechte Bild ist das Ergebnis einer Entzerrung durch OpenCV⁹.

In der Regel wird die Kalibrierung mit einem Muster durchgeführt, das auf irgendeine Weise aus Quadraten besteht, doch es können auch Kreise benutzt werden. Die Erkennung von Kreisen ist ebenfalls in OpenCV vorhanden, doch können bei einem Kreis-Muster die für die Kalibrierung nötigen Mittelpunkte nicht so präzise bestimmt werden, wie bei den Quadrat-Mustern die Eckpunkte.

⁸ Gary Bradski und Adrian Kaehler: "Learning OpenCV", 2008

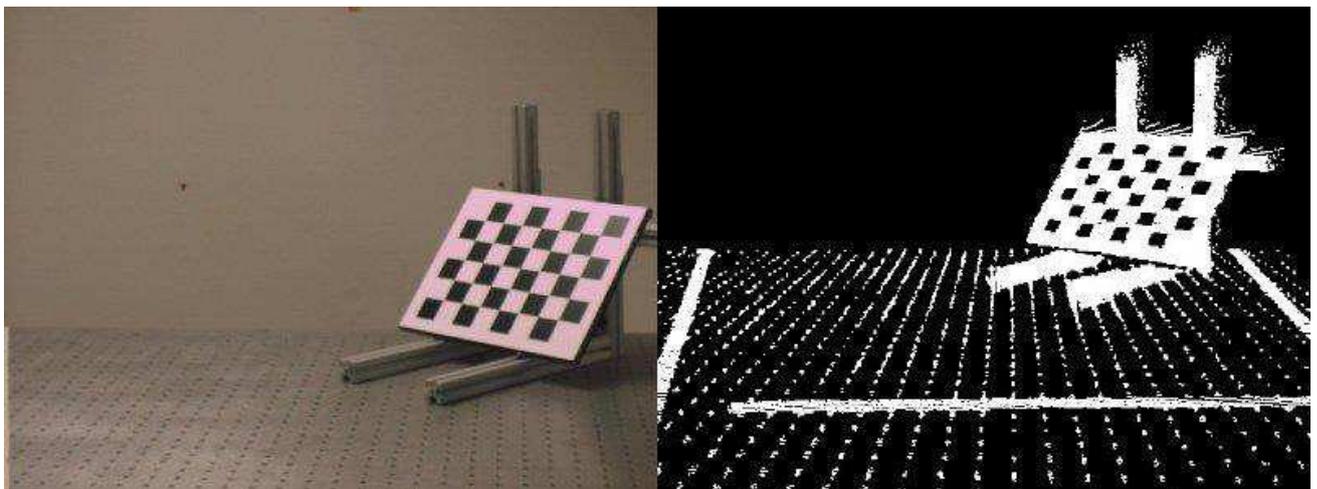
⁹ OpenCV Dokumentation 2.4.9:

http://docs.opencv.org/trunk/doc/tutorials/calib3d/camera_calibration/camera_calibration.html, 07.01.2013

Es muss aber nicht zwingend eine automatische Kalibrierung sein, es reicht auch wenn die Eckpunkte oder Kreismittelpunkte per Hand auf dem Kamerabild angegeben werden, da man eine Kalibrierung nur einmal für eine Kamera benötigt ist der Aufwand noch vertretbar, die restlichen Schritte können dann wieder automatisch durchgeführt werden.

2.3 Extrinsische Kamerakalibrierung

Um eine Kamera mit einem Laserscanner kombinieren zu können benötigt man die genaue Ausrichtung der Kamera zum Laserscanner. Dazu benötigt man ein Objekt, das man im Kamerabild und in der Punktwolke des Laserscanners klar erkennen kann. Oft wird das Kalibrierungsmuster von der intrinsischen Kalibrierung genutzt¹⁰, dies setzt aber eine hohe Auflösung des Laserscanners voraus wenn die Quadrate oder Kreise sicher gefunden werden sollen.



Die Linke Abbildung stammt von einer normalen Digitalkamera und die rechte ist ein Ausschnitt aus der mit einem 3D-Laserscanner aufgenommenen Punktwolke. Dank der hohen Auflösung des Laserscanners sind alle Quadratecken gut sichtbar und können deshalb für die Kalibrierung verwendet werden.

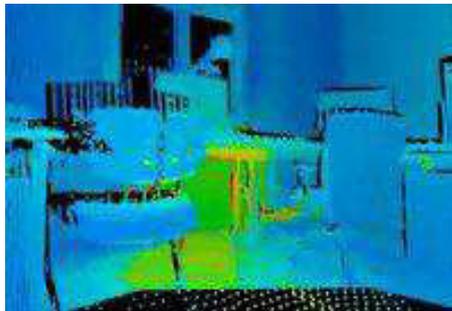
Eine extrinsische Kalibrierung kann mit jedem beliebigen Objekt als Referenz durchgeführt werden, wichtig ist nur, dass markante Punkte oder Linien zugeordnet werden können. Diese Zuordnung muss nicht automatisch stattfinden, aber in vielen Fällen sorgt eine automatische Erkennung für gleichbleibende Ergebnisse. Ob diese Ergebnisse insgesamt besser sind hängt natürlich von der Güte des Algorithmus ab.

¹⁰ Ranjith Unnikrishnan und Martial Hebert: "Fast Extrinsic Calibration of a Laser Rangefinder to a Camera", Robotics Institute. Paper 339. - 2005

2.4 Kalibrierung mit einer Thermalkamera

Bei den Kalibrierungsverfahren wurde eine Besonderheit der Thermalkamera noch nicht genannt, die gedruckten Kalibrierungsmuster können ohne weiteres Zutun nicht erkannt werden kann. Thermalkameras mit hoher Auflösung kommen mit einem Kalibrierungsmuster zurecht, welches mit einer Wärmelampe oder ähnliches erwärmt wird¹¹, doch mit geringer auflösenden Kameras kommt dieses Vorgehen nicht in Frage. Ein Kalibrierungsmuster bestehend aus Glühlampen kann hier die Lösung sein¹², diese Glühlampen können z.B. als Kreise in OpenCV interpretiert werden.

2.5 ThermalMapper



Punktwolke mit Thermaldaten vom Projekt ThermalMapper¹³.

Auf den ThermalMapper, der wie oben bereits erwähnt als Anhaltspunkt für diese Arbeit genutzt wird, wird nun kurz eingegangen. Das Projekt ThermalMapper kombiniert eine Punktwolke eines 3D-Laserscanners mit den Temperaturinformationen aus einer Thermalkamera um 3D-Thermal-Karten von Innenräumen zu erstellen. Der 3D-Laserscanner ist eine präzise und robuste Möglichkeit Entfernungsinformationen einer Umgebung zu sammeln. Die Thermalkamera befindet sich auf dem Laserscanner und dreht sich deshalb wie der Scanner ständig um die vertikale Achse(Z-Achse). Der ThermalMapper erkundet außerdem selbstständig seine Umgebung und fertigt eine 3D-Thermal-Karte an um Einsparpotenzial durch bessere Dämmung oder Entfernung von Energieverschwendern erkennen zu können. Die selbstständige Erkundung der Umgebung wird nicht Teil dieser Arbeit sein.

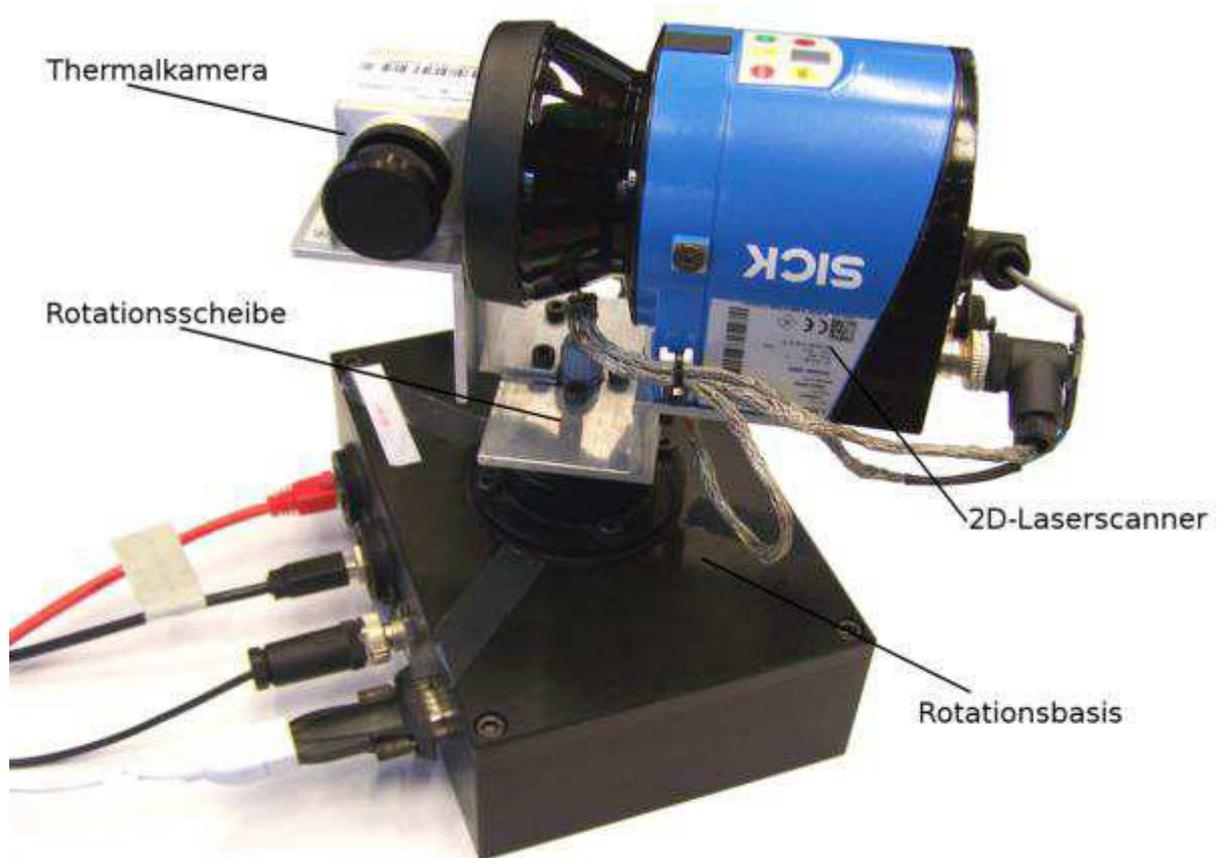
¹¹ Surya Prakash, Pei Yean Lee und Antonio Robles-Kelly: "Stereo techniques for 3D mapping of object surface emperatures", QIRT Journal Volume 4 - 1/2007

¹² Dorit Borrmann, Andreas Nüchter, Marija Dakulovic, Ivan Maurovic, Ivan Petrovic, Dinko Osmankovic und Jasmin Velagic: "The Project ThermalMapper - Thermal 3D Mapping of Indoor Environments for Saving Energy", 2011

¹³ Project Thermalmapper: <http://www.faculty.jacobs-university.de/anuechter/thermalmapper.html> , 07.01.2013

3 Systemdesign

3.1 Der 3D-Laserscanner und die Thermalkamera



Die zur Fusion von 3D-Laserscanner-Daten mit 2D-Thermalbild-Daten verwendete Hardware.

Für die Bachelorarbeit steht ein Fraunhofer 3DLS¹⁴ zur Verfügung. Dieser 3D-Laserscanner besteht im Wesentlichen aus zwei Komponenten, ein SICK LMS100¹⁵ 2D-Laserscanner und eine eigens dafür angefertigte Rotationseinrichtung (Rotationsscheibe und -basis). Der 2D-Laserscanner ist so befestigt, dass die 0°-Position eines 2D-Scans senkrecht nach oben zeigt, es handelt sich also um einen Nick-Winkel (engl. pitch) von -90°. Die Rotationseinrichtung dreht den 2D-Laserscanner um seine X-Achse (die Z-Achse im Weltkoordinatensystem), d.h. mit einer 180° Drehung kann eine 360° Punktwolke der Umgebung erstellt werden. Zur Auflösung der Punktwolke ist noch zu sagen, dass der Laserscanner 541 Punkte bei einem Öffnungswinkel von 270° aufnimmt und die Anzahl der

¹⁴ Fraunhofer IAIS 3D-Laserscanner (3DSL): <http://www.3d-scanner.net/index.html> (18.02.2013)

¹⁵ Datenblatt des SICK LMS100 Indoor: <https://www.mysick.com/PDF/Create.aspx?ProductID=33753&Culture=de-DE> (18.02.2013)

Laserscans über die Geschwindigkeit des Rotationsmechanismus eingestellt werden kann. Der SICK Laserscanner kann außerdem die Remissionswerte für die gemessenen Punkte zurückgeben, diese Grauwerte sind für die Erkennung des Kalibrierungsmusters von essenzieller Bedeutung.

Als Thermalkamera wird die Micro-Epsilon thermoIMAGER TIM 160¹⁶ verwendet. Der Temperatur-Messbereich reicht von -20°C bis 100°C bei einer thermischen Empfindlichkeit von 0,08K. Sie besitzt einen ungekühlten Infrarotdetektor, mehr dazu im Kapitel "Thermalkamera", und kann direkt an einen USB-Port angeschlossen werden. Ein Nachteil dieser Kamera ist die geringe optische Auflösung von nur 160 x 120 Pixel bei einem Öffnungswinkel von 64°. Die Thermalkamera ist genau wie der Laserscanner auf der Rotationsscheibe befestigt aber so gedreht, dass die Z-Achse der Kamera (die Blickrichtung) auf der Scanebene liegt. Dadurch sind für einen Teil der durch den 2D-Laserscanner ausgenommenen Punkte sofort die Thermalinformationen verfügbar. Kleine Bereiche könnten somit schnell erfasst werden, da keine größere Drehung abgewartet werden muss.

3.2 Das Kalibrierungsmuster

Im Kapitel "Stand der Technik" wurde erwähnt, dass ein für die „normalen“ optischen Kameras hergestelltes Kalibrierungsmuster auf einem Thermalbild nicht ohne weiteres Zutun zu sehen ist. Es gibt verschiedene Verfahren ein Muster im Thermalbild sichtbar zu machen, im Folgenden werden von dreien die Vor- und Nachteile diskutiert und abschließend wird das in dieser Arbeit verwendete Verfahren genauer vorgestellt.

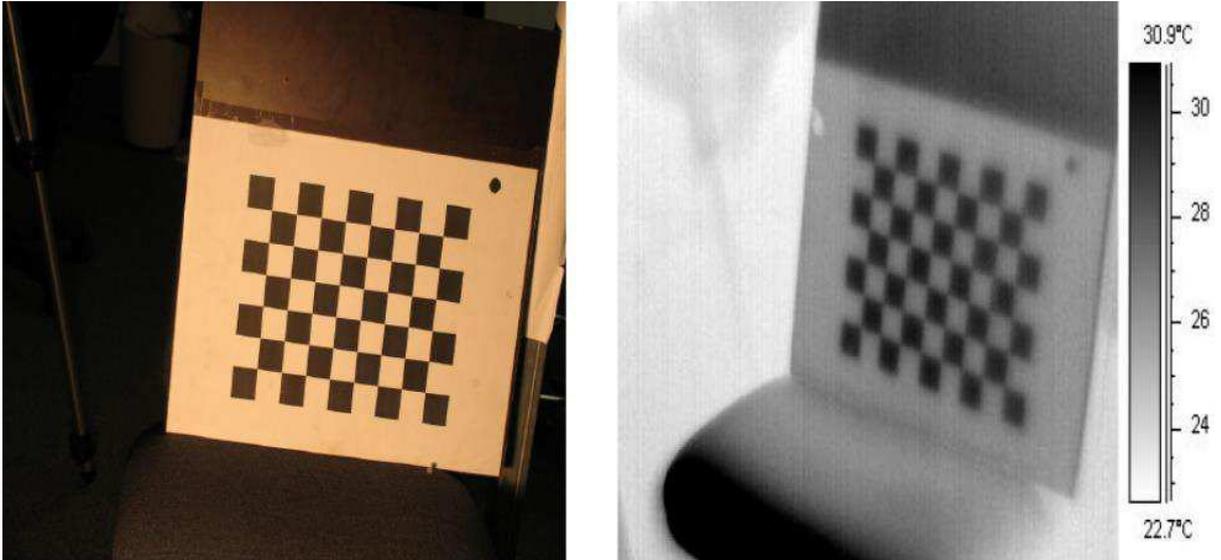
3.2.1 Erwärmen eines gedruckten Musters

Der Ausgangspunkt dieses Verfahrens ist ein Kalibrierungsmuster, wie man es aus verschiedenen Arbeiten bereits kennt. Es ist ein auf Papier gedrucktes Muster, das aus schwarzen Quadraten oder Kreisen besteht. Die Anordnung der Quadrate bzw. Kreise sollte einem sich wiederholenden Muster folgen um eine automatische Erkennung und Kalibrierung zu vereinfachen. Dieses klassische Kalibrierungsmuster kann nun mit einer Wärmelampe sichtbar gemacht werden¹⁷. Die schwarzen Flächen werden sich unter

¹⁶ Herstellerseite des thermoIMAGER TIM 160: http://www.micro-epsilon.de/temperature-sensors/thermoIMAGER/thermoIMAGER_160/index.html (18.02.2013)

¹⁷ Surya Prakash, Pei Yean Lee und Antonio Robles-Kelly: "Stereo techniques for 3D mapping of object surface

der Wärmelampe stärker aufwärmen als die weißen und so können sie im Thermalbild unterschieden werden.



Das linke Bild wurde mit einer normalen optischen Kamera gemacht und rechts sieht man dasselbe Muster im Thermalbild nach der Bestrahlung mit einer Wärmelampe.

Vorteile:

- geringer Aufwand beim Erstellen des Musters
- sehr geringe Kosten
- es können die bereits verfügbaren Programme zur Kalibrierung genutzt werden.

Nachteile:

- die Thermalkamera muss eine hohe optische Auflösung besitzen um die Kanten der Quadrate klar anzeigen zu können
- die Temperaturdifferenz ist gering
- eine gleichmäßige Erwärmung ist schwer zu realisieren

3.2.2 Aufbau des Musters mit Leuchtmitteln

Da eine Thermalkamera in der Regel nur die mittlere Infrarotstrahlung aufnimmt (siehe Kapitel "Thermalkamera"), hat es wenig Sinn ein Kalibrierungsmuster aus Infrarot-LEDs aufzubauen. Doch auch wenn diese LEDs nur nahes Infrarot ausstrahlen, können sie, wie auch alle anderen LEDs, auf Thermalbilder erkannt werden. Der Grund dafür ist, dass jede LED warm wird. Für ein Kalibrierungsmuster werden aber eher Leuchtmittel bevorzugt, die mehr Wärme abstrahlen als LEDs und dadurch deutlicher

wahrzunehmen sind, z.B. konventionelle Glühlampen¹⁸.



Auch hier zeigt das linke Bild das Muster durch eine normale optische Kamera und das rechte Bild dasselbe Muster im Thermalbild.

Vorteile:

- die Wärmequellen heben sich deutlicher von der Umgebung ab als die schwarzen Flächen der o.g. Variante
- alle Glühlampen auf dem Muster geben etwa gleich viel Wärme ab

Nachteile:

- hoher Verkabelungsaufwand
- Glühlampen erwärmen deutlich ihre Umgebung, d.h. die Positionen der Lampen sind nicht präzise zu bestimmen
- Standardprogramme zu Kalibrierung können nicht ohne größere Anpassungen verwendet werden

¹⁸ Dorit Borrmann, Hassan Afzal, Jan Elseberg und Andreas Nüchter: "Mutual Calibration for 3D Thermal Mapping", 2011

3.2.3 Aufbau des Musters mit Widerstandsdrähten

Widerstandsdrähte erwärmen sich bei durchfließendem Strom sehr gleichmäßig, außerdem kann die Temperatur über den Strom genau eingestellt werden. Es bietet sich also an, die Kanten der Quadrate eines klassischen Kalibrierungsmusters mit solchen Widerstandsdrähten sichtbar zu machen. Um nur die Eckpunkte, die bei einer Schachbrettkalibrierung Verwendung finden, im Thermalbild sichtbar zu machen genügt es bereits die inneren schwarzen Quadrate mit dem Draht zu umspannen.

Vorteile:

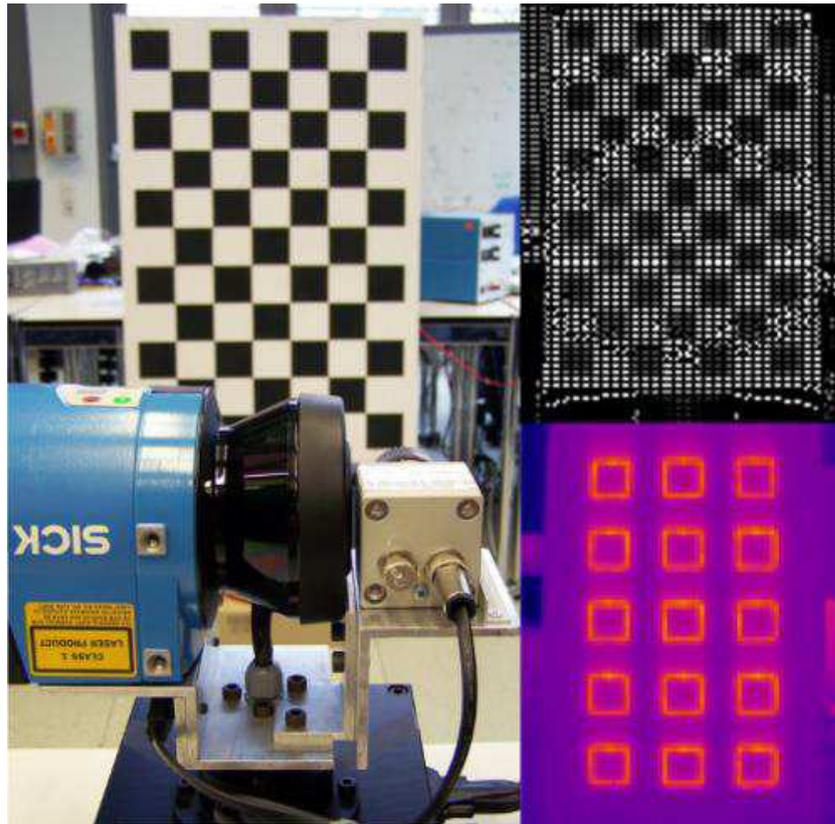
- die umspannten Quadrate sind deutlich sichtbar
- geringerer Aufwand als mit Leuchtmitteln
- bei geringem Strom sind die Quadrate scharf umrissen
- die Umgebung wärmt sich nicht so stark auf wie mit den Leuchtmitteln

Nachteil:

- nur wenige Standardkalibrierungsprogramme können dieses Muster erkennen

3.2.4 Ausgewähltes Verfahren: Muster mit Widerstandsdrähten

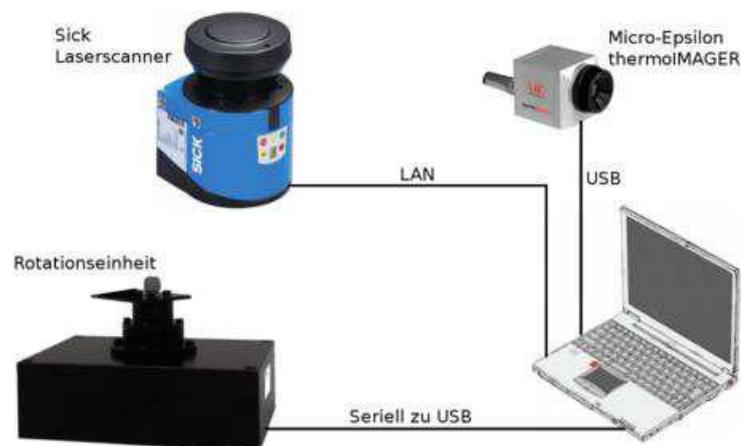
Die erste Variante scheidet sofort aus, da die zur Verfügung stehende Thermalkamera keine ausreichende optische Auflösung dafür bietet. Zwischen der zweiten und dritten Variante fiel die Entscheidung schon etwas schwerer. Glühlampen wurden bereits erfolgreich zur Kalibrierung von Thermalkameras genutzt, bei der letzten Variante hingegen kann man nicht auf die Erfahrungen und Erkenntnisse anderer zurückgreifen. Dennoch fiel die Entscheidung auf die letzte Variante, da ein einziges Muster für die intrinsische und extrinsische Kalibrierung ausreicht. Die Widerstandsdrähte werden einfach über ein Schachbrettmuster gelegt, das dank der Remissionswerte in der Punktwolke sichtbar ist. Die somit sichtbaren Konturen im Thermalbild können eindeutig zu den schwarzen Quadraten in der Punktwolke zugeordnet werden.



Links sieht man das Kalibrierungsmuster durch eine normale optische Kamera. Das rechte obere Bild stammt aus der Punktwolke des Laserscanner und rechts unten ist das Bild der Thermalkamera. Alle drei Aufnahmen zeigen dasselbe Kalibrierungsmuster. Durch den Vergleich der Bilder ist deutlich erkennbar welche Quadrate mit dem Widerstandsdraht umspannt worden sind und welche nicht.

3.3 Hardwareaufbau

Der Aufbau der Hardware ist relativ simpel, weswegen diese Übersichtsgrafik recht übersichtlich ausfällt.

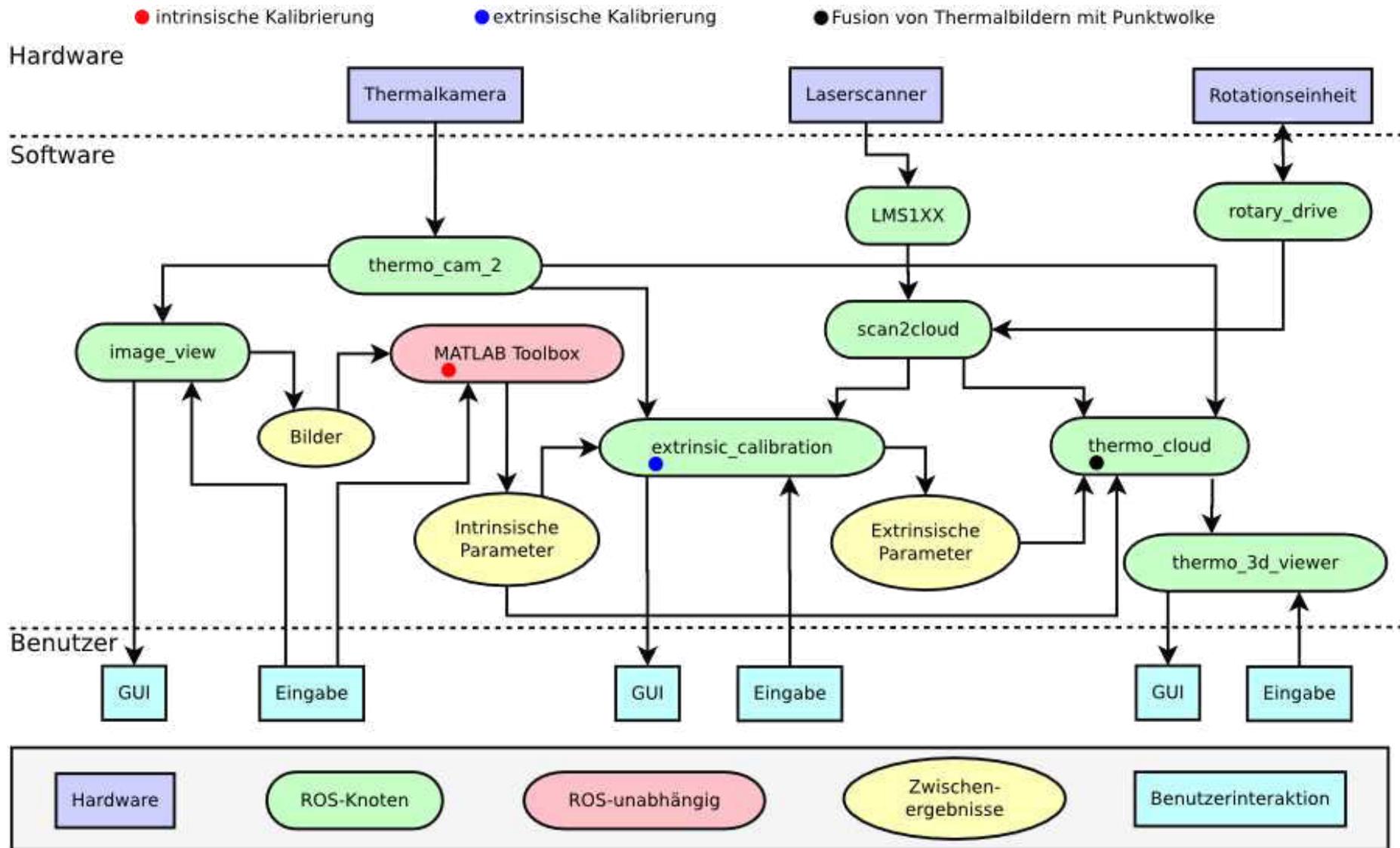


Der Hardwareaufbau mit den drei Hauptkomponenten und den Verbindungen zum Notebook

3.4 Systemaufbau

Der Systemaufbau gibt einen Überblick über den gesamten Aufbau der Software mit den Verbindungen zur Hardware. Der folgende Abschnitt gibt zusätzlich zur unten stehenden Grafik weitere Informationen zu den ROS-Knoten.

- **thermo_cam_2** (Erstellte/Modifizierte Software), ein Kamera-Treiber, der auf dem thermo_cam-Knoten von Herrn Wilkes basiert wurde für diese Arbeit erweitert, mehr dazu im Kapitel "Thermalkamera".
- **LMS1xx** (Verwendete Software), Laserscanner-Treiber
- **rotary_drive** (Verwendete Software), Rotationsscheiben-Treiber, ein Teil des 3DLSKdrivers
- **image_view** (Verwendete Software), Darstellungsprogramm für Kamerabilder
- **scan2cloud** (Verwendete Software), Erstellt aus 2D-Scans 3D-Punktwolken, ein Teil des 3DLSKdrivers
- **extrinsic_calibration** (Erstellte Software), bestimmt die extrinsischen Parameter, mehr dazu im Kapitel "Extrinsische Kalibrierung".
- **thermo_cloud** (Erstellte Software), kombiniert die Thermalbilder mit den Punktwolken, mehr dazu im Kapitel "Datenfusion".
- **thermo_3d_viewer** (Erstellte Software), stellt die Punktwolken dar und kann bestimmte Temperaturbereiche hervorheben, mehr dazu im Kapitel "Darstellung der Thermal-Punktwolke".



Gesamte Softwarearchitektur dieser Arbeit

3.5 Entwicklung

3.5.1 Betriebssysteme

- Ubuntu 12.04 LTS (64-bit): Das Entwicklungssystem
- Windows 7 (64-bit): Zur Erstellung der druckbaren Version (PDF-Version) dieser Arbeit

3.5.2 Frameworks und Tools

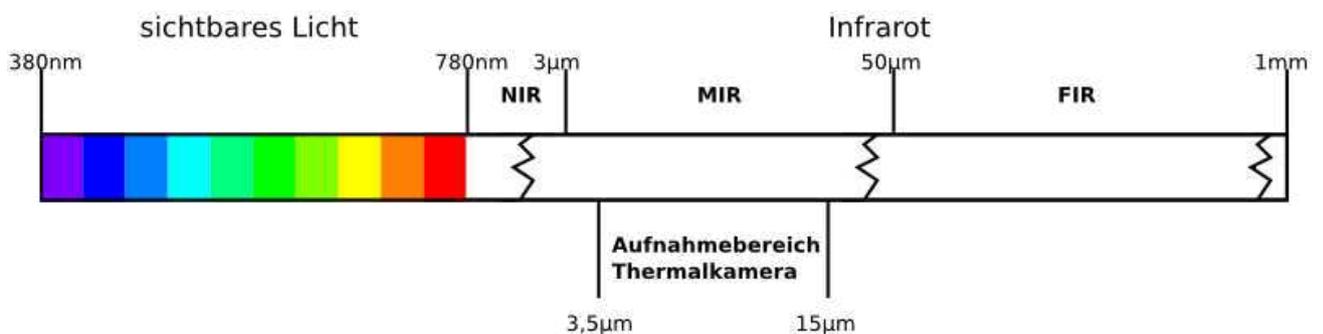
- Eclipse Indigo (Version: 3.7.2) mit C++ Plugin als Entwicklungsumgebung
- ROS Fuerte (Point Cloud Library 1.5)
- QT 4 für die GUI-Programmierung
- Inkscape 0.48 und Gimp 2.6 für die Abbildungen in dieser Arbeit
- MS Word 2010 für die Erstellung der druckbaren Version

4 Thermalkamera

4.1 Infrarotstrahlung

Die Infrarotstrahlung¹⁹ ist langwelliger als das sichtbare Licht und beginnt bei einer Wellenlänge von 780 Nanometer und endet bei einem Millimeter. Die Einteilung des Spektralbereichs der Infrarotstrahlung wird nach spezifischen Anwendungen oder physikalischen Phänomene vorgenommen und ist nicht eindeutig. Eine mögliche Einteilung ist in nahem, mittlerem, und fernem Infrarot²⁰. Das nahe Infrarot (NIR) ist kurzwellige IR-Strahlung von 780nm bis 3 μ m, die sich direkt an den sichtbaren Bereich anschließt und z.B. zur Signalübertragung in optischen Glasfasern benutzt wird. Mittleres Infrarot (MIR) mit Wellenlängen von 3 μ m bis 50 μ m ist der Bereich thermischer Strahlung bei irdischen Temperaturen ist. Die ferne Infrarotstrahlung (FIR) reicht von MIR (50 μ m) bis zur Terahertzstrahlung (1mm) und wird für astronomische Beobachtungen genutzt um sehr kalte Materie aufzuspüren.

Die Wärmestrahlung²¹, die eine Thermalkamera aufnimmt, kann von jedem Objekt stammen, das eine Temperatur oberhalb des absoluten Nullpunkts besitzt. Die bekannten optischen Prinzipien wie Beugung, Reflektion und Fokussierung mit Linsen gelten auch für die Wärmestrahlung.



Das sichtbare Licht und die sich daran anschließende Infrarotstrahlung.

¹⁹ Welt der Physik, Artikel über Infrarotstrahlung: <http://www.weltderphysik.de/gebiete/atome/forschung-mit-licht/elektromagnetisches-spektrum/infrarotstrahlung/> (18.02.2013)

²⁰ Wikipedia-Artikel über Infrarotstrahlung: <http://de.wikipedia.org/wiki/Infrarotstrahlung> (18.02.2013)

²¹ Artikel über Wärmestrahlung der Uni Heidelberg: <http://klimt.iwr.uni-heidelberg.de/PublicFG/ProjectB/CFT/dipluschimpf/node5.html> (15.03.2013)

4.2 Thermalkamera

Eine Thermalkamera, oder auch Wärmebildkamera²², macht Infrarotstrahlung sichtbar, die wir sonst nur als Wärme spüren können. Wärmebildkameras nutzen von der gesamten Infrarotstrahlung nur mittleres Infrarot und davon, aufgrund der typischen Emissionswellenlängen (siehe dazu das Wiensche Verschiebungsgesetz²³), meistens nur den Spektralbereich von 3,5 bis 15µm. Grundsätzlich erzeugt eine Infrarotkamera nur Graustufen-Bilder, da es sich aber oft um mehr Graustufen handelt als das menschliche Auge unterscheiden kann, sind Falschfarben-Darstellungen sinnvoll. Die Grauabstufungen von Thermalbildern, die auf unterschiedliche Temperaturen hinweisen, entsprechen in einer Falschfarben-Darstellung Farbabstufungen von z.B. Blau über Rot bis Weiß. Es gibt zwei Arten von Thermografie-Kameras: Systeme mit gekühlten Infrarotbilddetektoren und mit ungekühlten Detektoren.

Die **gekühlten** Detektoren bestehen aus einem Array aus Fotoempfängern, die sich in einem vakuumversiegelten Gehäuse befinden und kryogenisch gekühlt werden. Somit sind die Infrarotdetektoren in den meisten Fällen deutlich kälter als die zu beobachtenden Objekte. Die thermische Empfindlichkeit des Systems ist deshalb viel höher als bei einem System mit ungekühltem Detektor. Die Nachteile des gekühlten Systems sind die lange Anlaufzeit, bis der Detektor herunter gekühlt ist und die erhöhten Anschaffungs- und Betriebskosten. Sollte die Kühlung ausfallen kann nichts aufgenommen werden.

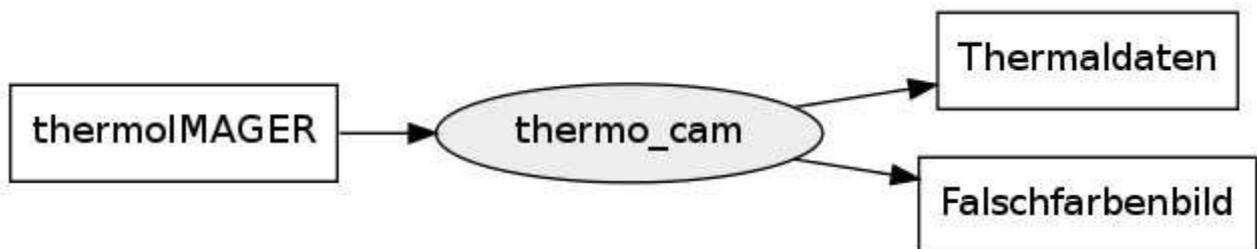
Die **ungekühlten** Systeme nutzen Detektoren, die bei Umgebungstemperatur arbeiten. Die Infrarotstrahlung heizt den Detektor auf und verändert so den Widerstand, die Spannung oder die Stromstärke. Die veränderten Werte werden dann mit den Werten des Detektors bei Betriebstemperatur verglichen. Die Betriebstemperatur wird durch konstante Temperierung über thermoelektrische Kühler gehalten und verringert den Signaldrift der Empfänger-Elemente. Jede ungekühlte Thermalkamera benötigt einen mechanischen Chopper oder eine periodische Abschaltung des Bildsensors um ein Dunkelbild zu gewinnen, welches als sensorspezifische Referenz vom aufgenommenen Bild Pixel für Pixel abgezogen wird.

²² Wikipedia-Artikel über Wärmebildkameras: <http://de.wikipedia.org/wiki/Wärmebildkamera> (18.02.2013)

²³ Wikipedia-Artikel über das Wiensche Verschiebungsgesetz: http://de.wikipedia.org/wiki/Wiensches_Verschiebungsgesetz (18.02.2013)

4.3 thermoIMAGER TIM 160 unter ROS

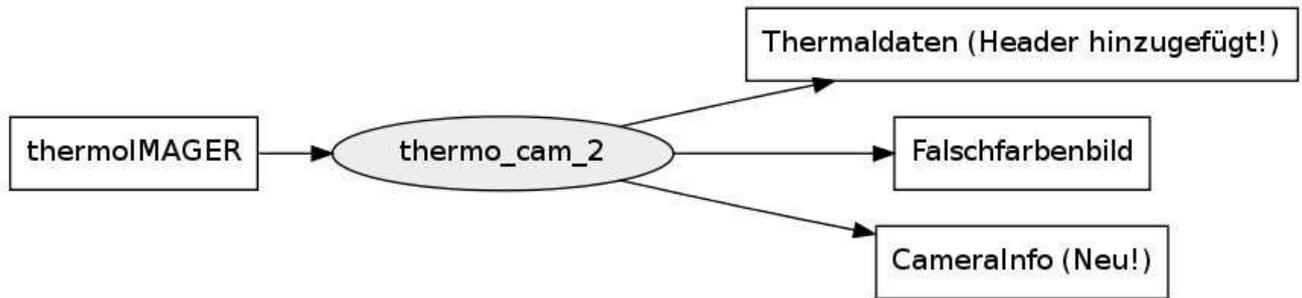
Im Kapitel "Systemdesign" fand diese Thermalkamera bereits Erwähnung. Die wichtigsten Kenndaten wurden genannt und deshalb folgt nun die Beschreibung der Software. Der Hersteller bietet für Windows-Systeme eine umfangreiche Software an, die es leider so nicht für Linux-Systeme gibt. Für Linux existiert, zum Zeitpunkt dieser Arbeit, nur der Kameratreiber und ein einfaches Anzeigeprogramm. Der Kameratreiber wurde von Herrn Wilkes von einem ROS-Knoten (`thermo_cam`) umhüllt und kann dadurch wie eine normale Kamera in ROS verwendet werden.



Die Thermalkamera gibt ein Graustufenbild zurück, wobei jedem Grauwert eine Temperatur zugeordnet werden kann. Im ROS-Knoten `thermo_cam` werden diese Graustufen in Falschfarben umgesetzt. So kann aus dem geringen Unterschied im Grauton zwischen 20°C und 25°C ein deutlicher Farbunterschied von blau zu orange werden. Welche Farben genau eingesetzt werden entscheidet eine Farbpalette, die individuell angepasst werden kann. Die Umwandlung in Falschfarben geschieht im `thermo_cam`-Knoten flexibel, d.h. die minimale und maximale Temperatur werden ermittelt um einen Temperaturbereich anzugeben in dem die verfügbare Farben gleichmäßig verteilt werden. Eine Farbe repräsentiert demnach nicht immer dieselbe Temperatur und kann außerdem gleich für mehrere Temperaturen stehen, wenn der Temperaturbereich größer als die Farbpalette ist.

Das erzeugte Falschfarbenbild wird als `sensor_msgs/Image` veröffentlicht, auch jeder andere Kamera-Knoten benutzt für sein Kamerabild diesen Nachrichtentypen. Zusätzlich zu den Falschfarbenbildern werden auch die Temperaturwerte in $^{\circ}\text{C}$ veröffentlicht. Der `thermo_cam`-Knoten beinhaltet zu diesem Zweck einen eigenen Thermaldaten-Nachrichtentyp (`thermo_cam/ThermoData`), der einen ähnlichen Aufbau wie das `sensor_msgs/Image` hat.

4.3.1 Erweiterungen



Folgende Erweiterungen waren für den thermo_cam-Knoten nötig, damit er in dieser Arbeit eingesetzt werden konnte:

- Die Thermaldaten wurden wie oben genannt als ThermoData-Nachricht veröffentlicht, doch dieser Nachrichtentyp beinhaltete keinen Header. Der Header bietet unter anderem den Zeitstempel und ohne einen Zeitstempel kann keine Zuordnung von Thermaldaten zu einem Falschfarbenbild gemacht werden. Es war wichtig, dass die Thermaldaten und das Falschfarbenbild denselben Zeitstempel erhielten, wenn sie von ein und derselben Kameraaufnahme stammten. Der Header wurde hinzugefügt und mit demselben Zeitstempel versehen, den auch das Falschfarbenbild erhielt.
- Die automatische Anpassung der Farbzurordnung führt bei der Zusammenführung von mehreren Aufnahmen mit der Punktwolke dazu, dass eine Farbe bereits innerhalb einer Punktwolke völlig unterschiedliche Temperaturen repräsentiert. Deshalb wurde die Möglichkeit geschaffen den Temperaturbereich über die launch-Datei festzulegen. Temperaturen die außerhalb des definierten Bereichs liegen erhalten dann die Farbe des minimalen bzw. des maximalen Temperaturwertes.
- Die intrinsischen Parameter werden in ROS in der Regel vom Kameraknoten selbst veröffentlicht. Diese Funktionalität muss nicht für jeden Kameraknoten neu implementiert werden. Der CameraInfoManager stellt alles Nötige zur Verfügung und muss nur eingebunden werden. Die intrinsischen Parameter können aus einer Datei geladen oder über einen Service an den Knoten gesendet werden. Da der thermo_cam-Knoten noch nichts dergleichen bot, wurde der CameraInfoManager eingebunden.

5 Intrinsische Kamerakalibrierung

5.1 Intrinsische Parameter

Die intrinsischen Parameter stellen den Zusammenhang zwischen dem Kamera- und dem Bildkoordinatensystem her und werden zusammengenommen als Kameramodell bezeichnet. Das Kameramodell ist unabhängig von der Orientierung und Position der Kamera im Weltkoordinatensystem. Zur Ermittlung der intrinsischen Parameter wurde die MATLAB Camera Calibration Toolbox²⁴ genutzt. Ein Kameramodell dieser Toolbox umfasst die folgenden Parameter:

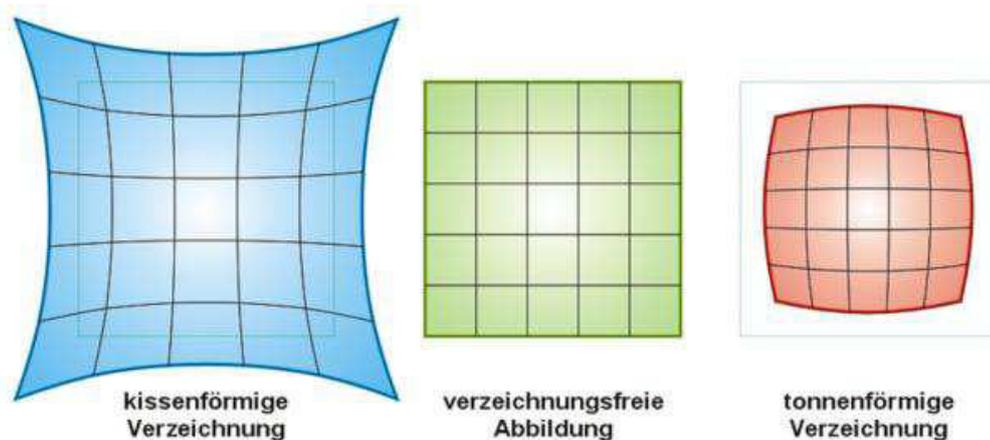
- Die Brennweite f_c (focal length),
- das optische Zentrum in der Bildebene c_c (principal point),
- der Drehungskoeffizient α_c (skew coefficient) und
- die Verzerrung k_c (distortions).

Die Brennweite ist wie man es erwartet die Entfernung der Bildebene zum Projektionszentrum, doch das optische Zentrum in der Bildebene ist nicht mit der Bildmitte gleichzusetzen. Das optische Zentrum c_c ist der Punkt in dem die optische Achse die Bildebene schneidet, dieser Punkt stimmt bei einer realen Kamera, wegen Ungenauigkeiten in der Produktion, selten mit der Bildmitte überein. Der Drehungskoeffizient gibt Auskunft über den Winkel zwischen der X- und Y-Achse des Bildkoordinatensystems, doch bei fast allen Kameras ist der Winkel 90° und deshalb kann dieser Parameter vernachlässigt werden. Eine Kamera verzerrt die Bildpunkte, manche Kameras stärker als andere aber es gibt keine Kamera ohne Verzerrung. Da man also eine Verzerrung nicht vermeiden kann muss auch sie exakt bestimmt werden, um sie im Anschluss aus dem Kamerabild heraus rechnen zu können.

²⁴ Dokumentation der MATLAB Camera Calibration Toolbox: http://www.vision.caltech.edu/bouguetj/calib_doc/ (18.02.2013)

Exkurs: Die Verzerrung (Verzeichnung) einer Kamera

Es gibt zwei wichtige Arten der geometrischen Verzerrung: radiale und tangentielle Verzerrung²⁵. Die radiale Verzerrung skaliert den Abstand des Bildpunktes zum Fokus, dem Zentrum der Verzerrung. Je weiter man sich vom optischen Zentrum entfernt desto stärker ist die radiale Verzerrung. Auf Bildern fällt dem Betrachter nur die radiale Verzerrung auf, da sie generell stärker ausfällt als die tangentielle Verzerrung. Die tangentielle Verzerrung tritt tangential zu dem Vektor von dem Zentrum des Bildes auf und resultiert hauptsächlich aus der Dezentralisierung der Linsen der Objektive. Der Einfluss der tangentialen Verzerrung ist aber so gering, dass sich eine Bestimmung nur bei hohen Genauigkeitsanforderungen lohnt. Die Bestimmung der Verzerrung beschränkt sich daher oft auf die radiale Verzerrung der Kamera.



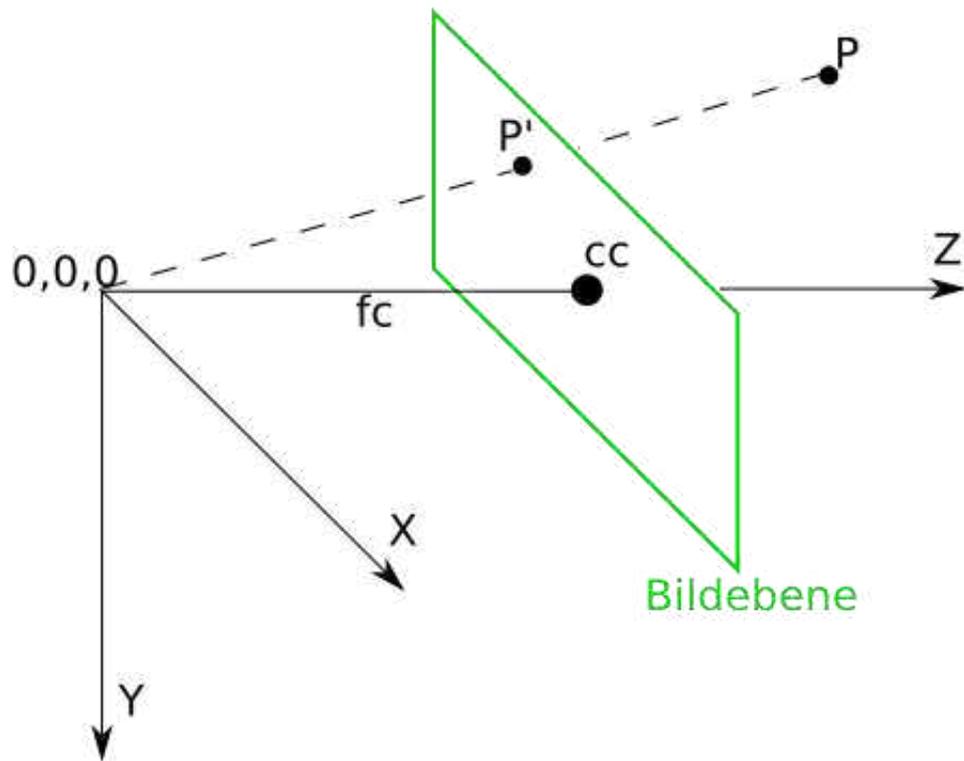
Beispiele für radiale Verzerrungen einer Kamera. Bildquelle²⁶

Mit dem Kameramodell ist die Abbildung zwischen dem 3D-Kamera- und dem 2D-Bildkoordinatensystem definiert, d.h. ein 3D-Punkt im Kamerakoordinatensystem kann auf die Bildebene der Kamera projiziert werden. Die intrinsischen Parameter allein machen den umgekehrten Weg, also von einem 2D-Punkt auf einen 3D-Punkt, nicht möglich. Denn einem 2D-Punkt auf der Bildebene können beliebig viele 3D-Punkte auf dem zugehörigen Sehstrahl (die Gerade, die den Ursprung des Kamerakoordinatensystems und die Bildebene bei dem o.g. 2D-Punkt schneidet) zugeordnet werden.

²⁵ Kamera Kalibrierung, Hauptseminar Augmented Reality TUM-Lehrstuhl Klinker/Navab, WS2004:

<http://campar.in.tum.de/twiki/pub/Chair/TeachingWs04Tracking/05CameraCalibration.pdf>

²⁶ Wikipedia-Artikel zu Verzeichnungen: <http://de.wikipedia.org/wiki/Verzeichnung> (25.02.2013)



Das Kamerakoordinatensystem mit der Bildebene der Kamera. Hier wurde der 3D-Punkt P auf die Bildebene projiziert und zeigt sich somit im Bildkoordinatensystem als 2D-Punkt P' . Die gestrichelte Linie ist der bereits erwähnte Sehstrahl.²⁷

5.2 Intrinsische Kalibrierung

Die Kalibrierung mit der MATLAB Camera Calibration Toolbox basiert auf der in "A Four-step Camera Calibration Procedure with Implicit Image Correction"²⁸ beschriebenen Prozedur.

Es handelt sich um eine 4-Schritte Kalibrierung. Damit man eine grobe Vorstellung vom Ablauf erhält, werden die einzelnen Schritte kurz vorgestellt.

1. Lineare Parameterschätzung

Mit der DLT-Methode (direct linear transformation), die auf dem Lochkameramodell basiert und die nichtlineare radiale und tangentiale Verzerrung ignoriert, gelangt man zu den Parametern.

²⁷ Kamerakoordinatensystem von den CameraInfo-Nachrichten in ROS: http://www.ros.org/wiki/image_pipeline/CameraInfo (18.02.2013)

²⁸ Janne Heikkilä und Olli Silvén: "A Four-step Camera Calibration Procedure with Implicit Image Correction", 1997, http://www.vision.caltech.edu/bouguetj/calib_doc/papers/heikkila97.pdf

2. Nichtlineare Schätzung

Es wird angenommen, dass es sich beim Messfehler um weißes gaußsches Rauschen handelt. Für die Schätzung der Kameraparameter wird dann die Abweichung zwischen dem Modell und N Beobachtungen minimiert, wobei das Ergebnis der linearen Parameterschätzung als Ausgangspunkt genommen wird

3. Korrektur für die asymmetrische Projektion

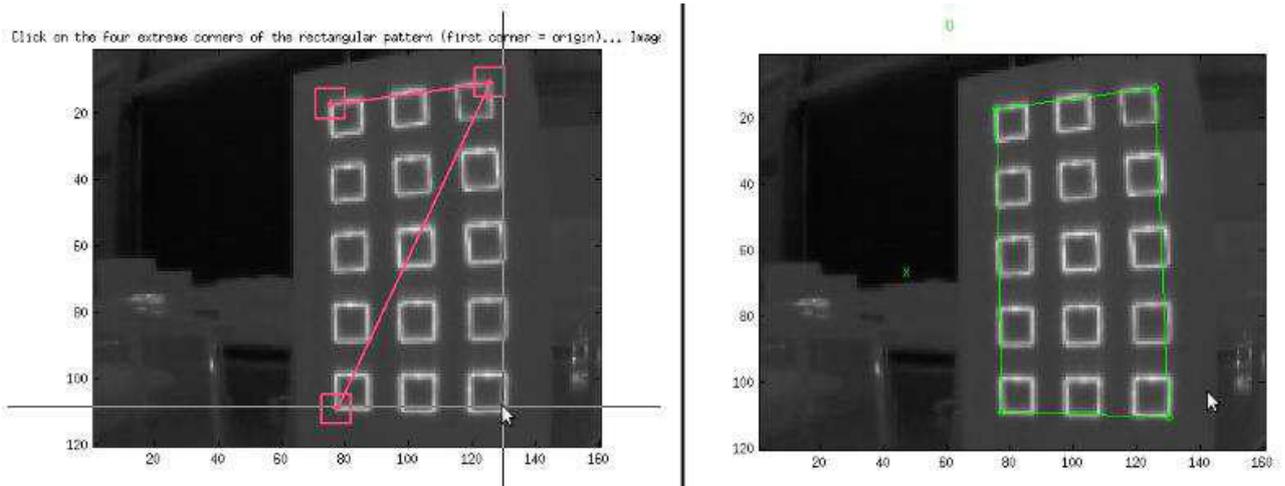
Für den Fall, dass die Projektionen der Kontrollpunkte die Pixelgröße übersteigen muss mit den Auswirkungen perspektivischer Verzerrungen entsprechend umgegangen werden.

4. Bildkorrektur

Im letzten Schritt wird die Bildkorrektur über die Interpolation der korrekten Bildpunkte auf Basis der Kameraparameter aus den vorherigen Schritten erreicht.

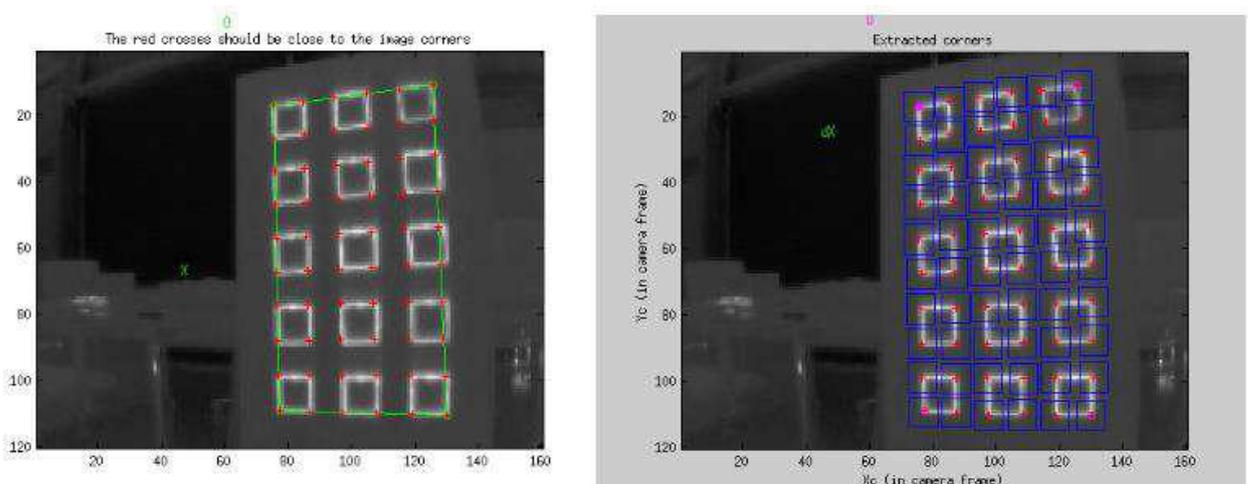
Die Toolbox war zum Zeitpunkt dieser Arbeit das einzige Kalibrierungsprogramm, welches die Eckpunkte des Musters aus Widerstandsdrähten zuverlässig und ohne Anpassungen erkennen konnte. Die Erkennung der Eckpunkte geschieht nicht gänzlich automatisch, es sind einige Angaben nötig, die nun beschrieben werden.

- Nachdem man die Bilder entsprechend der Anleitung in die Toolbox geladen hat, wird für die Extraktion der Eckpunkte die Angabe einer Fenstergröße erwartet. Ein Fenster gibt den Bereich vor, in dem zu der Sollposition einer Ecke die tatsächliche Position gesucht werden darf. Für die Thermalkamera haben die Werte $wintx = 4$ und $winty = 4$ die besten Ergebnisse geliefert, wenn dabei das Kalibrierungsmuster ca. 1,5m entfernt war. Größere Entfernungen sind aufgrund der geringen optischen Kameraauflösung nicht empfehlenswert. Die nachfolgenden Bilder verdeutlichen den Sinn der Fenster.
- Der nächste Schritt ist die manuelle Markierung der äußeren Eckpunkte. Die Reihenfolge ist zu Beginn beliebig, doch die Eckpunkte in weiteren Bildern müssen in der gleichen Reihenfolge markiert werden wie im ersten Bild.



Das linke Bild zeigt die GUI der MATLAB Camera Calibration Toolbox, in der drei Eckpunkte bereits markiert wurden. Die roten Quadrate sind die Fenster in denen nach der tatsächlichen Position der Eckpunkte gesucht wird. Auf dem rechten Bild ist die gleiche GUI nach dem Klick auf die 4. Ecke zu sehen.

- Nach der Markierung der äußeren Eckpunkte wird die Angabe der Anzahl der Quadrate auf dem Kalibrierungsmuster notwendig, da das automatische Zählen bei diesem speziellen Muster fehlschlägt. In X-Richtung ist das 9 und in Y-Richtung 5 oder umgekehrt, falls das Muster gedreht ist.
- Die Größe der Quadrate ist bei dem Muster, das auf den Bildern zu sehen ist, 90x90mm und muss ebenfalls angegeben werden.
- Eine gute Kalibrierung benötigt mehr als ein Bild, zudem müssen weitere Bilder das Kalibrierungsmuster aus anderen Perspektiven zeigen. Wenn nun alle Eckpunkte extrahiert wurden, kann die Kalibrierung ausgeführt werden.



Nachdem die Eckpunkte markiert wurden hat die Toolbox die Soll-Position in die Kameraaufnahme eingezeichnet (links). Die Ist-Positionen konnten erfolgreich innerhalb der Fenster gefunden werden (rechts).

Das Ergebnis der Thermalkamera-Kalibrierung ist ein Kameramodell wie dieses hier:

$fc = [182.1995 \ 180.93691] \pm [5.53195 \ 5.68313]$

$cc = [77.65186 \ 59.84703] \pm [4.48596 \ 6.01754]$

$kc = [-0.10092 \ 0.75153 \ -0.00254 \ 0.00702 \ 0.0] \pm [0.10092 \ 0.48147 \ 0.00647 \ 0.00622 \ 0.0]$

$err = [0.52111 \ 0.33560]$

Die Komponenten werden in den folgenden Abschnitten so bezeichnet:

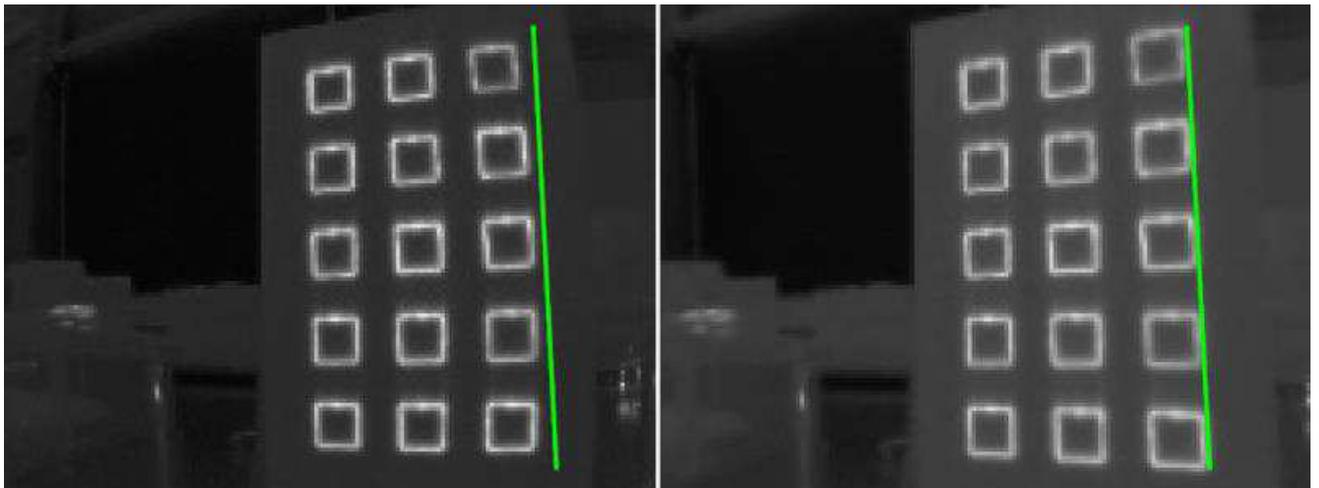
$fc = [fc1 \ fc2]$

$cc = [cc1 \ cc2]$

$kc = [kc1 \ kc2 \ kc3 \ kc4 \ kc5]$

5.3 Korrektur eines Thermalbildes mit den intr. Parametern

Das von der Toolbox ausgegebene Kameramodell ist nicht zwangsläufig korrekt. Bei einer starken Verzerrung können Eckpunkte außerhalb der angegebenen Fenster liegen und werden deshalb nicht gefunden. Doch diese Eckpunkte werden in der Toolbox nicht hervorgehoben. Aus diesem Grund werden, gerade wenn viele Eckpunkte für die Kalibrierung genutzt werden, die nicht richtig markierten Eckpunkte vom Nutzer leicht übersehen. Zur Kontrolle werden deshalb die Thermalbilder mit der Toolbox oder auch mit OpenCV entzerrt.



Auf der linken Seite ist das Originalbild vom thermo_cam-Knoten. Die andere Seite zeigt das gleiche Bild nach der Entzerrung durch die Toolbox. Die grüne Linie ist in beiden Bildern an derselben Position und auch die Ausrichtung ist identisch. Man kann erkennen, dass Teile des Bildes nach der Entzerrung verloren gegangen sind, da die Bildgröße beibehalten wurde.

Bei der verwendeten Thermalkamera tritt eine tonnenförmige Verzeichnung auf, d.h. durch die Entzerrung wird das Bild größer und ragt über die Bildgrenzen hinaus. Man müsste dem entzerrten Bild eine größere Fläche zur Verfügung stellen, wenn man alle Pixel benutzen möchte. Aber ein entzerrtes Bild, welches alle verfügbaren Pixel anzeigt, beinhaltet auch viele freie Flächen, die durch die ungleichmäßige Ausdehnung des Bildes entstanden sind. Es erleichtert die weitere Verwendung des Thermalbildes ungemein, wenn das Ausgangsbild dieselbe Größe hat wie das Eingangsbild.

5.4 Intrinsische Parameter in ROS

In ROS ist es üblich die intrinsischen Kameraparameter in der Form von CameraInfo-Messages weiterzugeben. Diese Nachrichten werden in der Regel vom Kameraknoten zeitgleich mit dem Kamerabild ausgegeben. Der Knoten für die Thermalkamera wurde dementsprechend erweitert und lädt die zu sendenden Parameter aus einer Konfigurationsdatei, in die der Nutzer die von der MATLAB Toolbox ermittelten Werte einträgt. Alternativ können die Parameter auch über den ROS-Service „set_camera_info“ von einem Kalibrierungsknoten an den Kameraknoten gesendet werden und der Kameraknoten speichert diese Werte wiederum in die Konfigurationsdatei. Der Aufbau dieser Konfigurationsdatei ist durch den camera_info_manager von ROS vorgegeben und weicht etwas von der MATLAB Toolbox Ausgabe ab, deshalb werden nachfolgend die Bedeutungen der Einträge erläutert.

Die Konfigurationsdatei findet man im ROS-Paket "thermo_cam_2" im Unterordner "/cali" unter dem Namen "cali.yaml". Der Inhalt könnte z.B. so aussehen:

```
image_width: 160
image_height: 120
camera_name: thermo_cam
camera_matrix:
  rows: 3
  cols: 3
  data: [182.19955, 0, 77.65186, 0, 180.93691, 59.84703, 0, 0, 1]
distortion_model: plumb_bob
distortion_coefficients:
  rows: 1
  cols: 5
  data: [-0.58172, 0.75153, -0.00254, 0.00702, 0]
rectification_matrix:
  rows: 3
  cols: 3
  data: [1, 0, 0, 0, 1, 0, 0, 0, 1]
projection_matrix:
  rows: 3
  cols: 4
  data: [182.19955, 0, 77.65186, 0, 0, 180.93691, 59.84703, 0, 0, 0, 1, 0]
```

Die ersten drei Zeilen sind verständlicherweise bei jeder Kalibrierung mit dieser Thermalkamera identisch. In der darauffolgenden Kameramatrix (`camera_matrix`) werden die Brennweite fc und das optische Zentrum cc vereint gespeichert. Die Kameramatrix K ist wie

folgt aufgebaut:
$$K = \begin{pmatrix} fc_1 & 0 & cc_1 \\ 0 & fc_2 & cc_2 \\ 0 & 0 & 1 \end{pmatrix}$$
. In die Konfigurationsdatei werden Matrizen immer zeilenweise hintereinander geschrieben. Das verwendete Verzerrungsmodell (`distortion_model`) heißt "Plump Bob" und wurde erstmals in "Decentering Distortion of Lenses"²⁹ beschrieben. Die Verzerrungskoeffizienten (`distortion_coefficients`) D werden in der gleichen Reihenfolge angegeben wie die Toolbox sie ausgibt: $D = (kc_1 \ kc_2 \ kc_3 \ kc_4 \ kc_5)$.

Die Korrekturmatrix (`rectification_matrix`) R ist nur für Stereo-Kameras gedacht und kann deshalb einfach mit der Einheitsmatrix belegt werden. Auch die Projektionsmatrix (`projection_matrix`) P ist abhängig von der Kameraart. Bei einer normalen "einäugigen"

Kamera ist P zum Großteil identisch mit K :
$$P = \begin{pmatrix} fc_1 & 0 & cc_1 & 0 \\ 0 & fc_2 & cc_2 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$
.

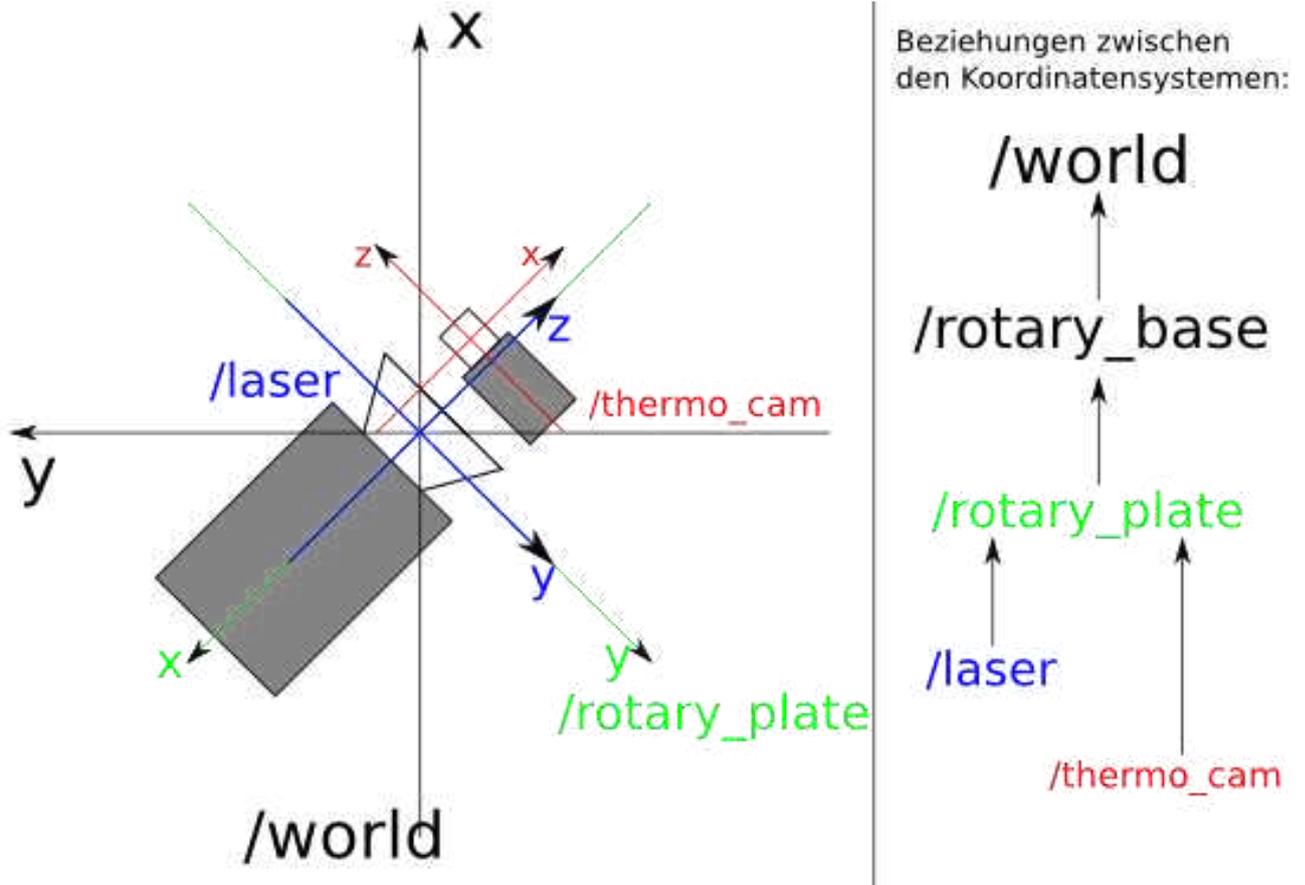
²⁹ Brown: "Decentering Distortion of Lenses", 1966

6 Extrinsische Kalibrierung

6.1 Extrinsische Parameter

Extrinsische (äußere) Parameter beschreiben die Position und die Orientierung eines Koordinatensystems zu einem anderen. Für die Fusion von 3D-Laserscanner- mit 2D-Thermalbilddaten muss das Kamera- mit dem Weltkoordinatensystem in Zusammenhang gebracht werden. Die extrinsischen Parameter können zusammen mit den intrinsischen Parametern verwendet werden, um zum Beispiel einem 2D-Punkt aus dem Bildkoordinatensystem einem 3D-Punkt im Weltkoordinatensystem zuzuordnen. In vielen Fällen gibt es aber keine direkte Transformation vom Kamera- zum Weltkoordinatensystem. Auch hier ist das der Fall, da die Orientierung und Position der Kamera relativ zu der Rotationsscheibe angegeben wird. Die Rotationsscheibe ist wiederum mit der Rotationsbasis verbunden und erst die Position und Ausrichtung der Basis wird in Weltkoordinaten angegeben. Die extrinsische Kalibrierung wird also nicht direkt die Orientierung sowie Position der Kamera in Weltkoordinaten zurückgeben. Nur die Transformation zwischen Kamera- und Rotationsscheiben-Koordinatensystem ist zu bestimmen. Die Transformationen von der Rotationsscheibe bis zur Welt sind bereits durch den ROS-Knoten des 3D-Laserscanners bekannt. Die Verwaltung der Koordinatensysteme mit ihren Transformationen geschieht in ROS durch `tf`³⁰.

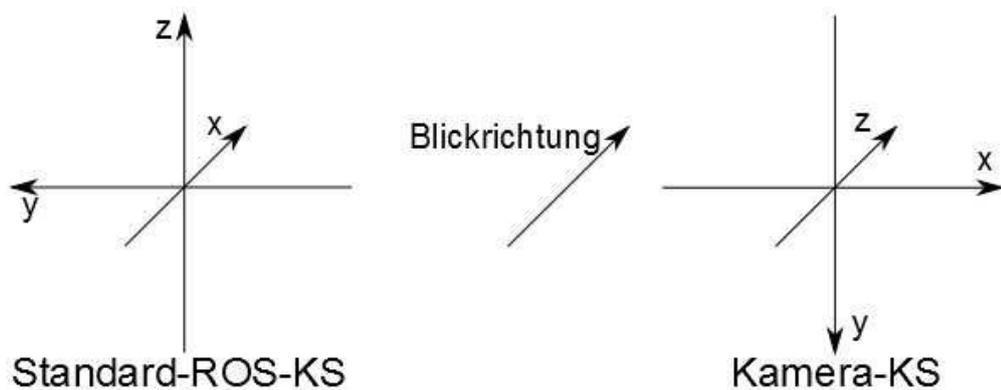
³⁰ ROS-Seite zu `tf`: <http://www.ros.org/wiki/tf> (25.02.2013)



Die Draufsicht auf den 2D-Laserscanner und die Thermalkamera mit ihren Koordinatensystemen. Von der Rotationsscheibe wurde nur das Koordinatensystem eingezeichnet und das KS der Rotationsbasis fehlt, da es mit dem Welt-KS identisch ist. Die Beschriftungen der Koordinatensysteme entsprechen den Bezeichnungen in tf.

Die Abhängigkeiten der Koordinatensysteme werden in tf als Eltern-Kind-Beziehungen verwaltet, wobei das Kind-Koordinatensystem als Objekt innerhalb des Eltern-Koordinatensystems angesehen wird. Es gibt zwei Möglichkeiten die Rotation eines Koordinatensystems innerhalb eines übergeordneten Systems anzugeben. Die Roll-Pitch-Yaw-Drehungen um feste Achsen und die Drehungen mit Eulerwinkeln. Die Eulerwinkel geben die Drehungen um die Achsen des Objekts bzw. des Kindkoordinatensystems unabhängig von dessen Orientierung im Weltkoordinatensystem an. Bei den Rotationen um feste Achsen hingegen wird immer um die Achsen des Elternkoordinatensystems gedreht. Die Roll-Pitch-Yaw-Drehungen um feste Achsen wurden wegen der einfacher erkennbaren Endlage des Koordinatensystems in dieser Arbeit bevorzugt. In ROS zeigt die X-Achse nach vorne, die Y-Achse nach links und die Z-Achse nach oben. Das Koordinatensystem einer

Kamera ist ein Sonderfall, denn es ist anders ausgerichtet als das Standard-ROS-KS³¹. Damit die Kamera in die richtige Richtung zeigt, muss das Kamera-KS so gedreht werden, dass die Z-Achse in die gewünschte Blickrichtung und die X-Achse nach rechts zeigt. Davon unberührt dreht ein Yaw das Kind-Koordinatensystem um die Z-Achse des Eltern-Koordinatensystem, Pitch um die Y-Achse und Roll um die X-Achse. Die Drehungen werden immer in der Reihenfolge Roll-Pitch-Yaw durchgeführt. Die Position des Kind-KS im Eltern-KS ist unabhängig von der angegebenen Drehung, weil immer zuerst das KS gedreht wird bevor es verschoben wird.



Das ROS-Koordinatensystem, das für die meisten Objekte genutzt wird, unterscheidet sich in der Ausrichtung von den Kamerakoordinatensystemen.

6.2 Laserscanner und Punktwolken

In dieser Bachelorarbeit wird nicht direkt auf die Distanz- und Remissionswerte des Laserscanners zugegriffen. Der bereits existente ROS-Stack 3DLSKdriver übernimmt diese Arbeit und liefert die benötigten 3D-Aufnahmen der Umgebung als Punktwolken. Deshalb wird auf die Funktionsweise eines Laserscanners nicht eingegangen. Einen groben Überblick liefert der Wikipedia-Artikel³² zu Laserscanning und Informationen zur Erstellung von Punktwolken kann man aus der Arbeit "Kognitive Robotik 1"³³ (sowie aus darauf aufbauende Arbeiten) gewinnen. Für die Arbeit mit Punktwolken wird die in ROS zur Verfügung stehende Point Cloud Library (PCL) genutzt. Die neben den Strukturen zur Verwaltung von Punktwolken auch viele Möglichkeiten der Manipulation und Visualisierung bietet.

³¹ ROS-Seite zur CameraInfo, besonders zum Kamera-KS: http://www.ros.org/wiki/image_pipeline/CameraInfo (25.02.2013)

³² Wikipedia-Artikel zu Laserscanning : <http://de.wikipedia.org/wiki/Laserscanning> (08.03.2013)

³³ B. Möller, Kognitive Robotik 1, http://roblab.informatik.fh-gelsenkirchen.de/mediawiki/index.php/Kognitive_Robotik_1

6.3 Extrinsische Kalibrierung

Für die Ermittlung der extrinsischen Parameter muss eine Verbindung zwischen dem Kalibrierungsmuster auf dem Kamerabild und demselben Muster in der Punktwolke des 3D-Laserscanners hergestellt werden. Hat man die markanten Punkte beider Aufnahmen des Musters gefunden kann die Position und Ausrichtung der Kamera zum Zeitpunkt der Aufnahmen bestimmt werden. Da die Kamera und er 2D-Laserscanner starr miteinander verbunden sind, ist eine einzige Aufnahme des Kalibrierungsmusters ausreichend. Doch bevor an die Umsetzung gedacht werden konnte, mussten zunächst ein paar grundlegende Fragen beantwortet werden:

- **Was für Daten müssen aus dem Kamerabild und der Punktwolke des Laserscanners gewonnen werden?**

Wir nehmen also an, dass die Kamera frei im Raum positioniert werden kann, d.h. es müssen mindestens vier Punkte des Kalibrierungsmusters im Kamerabild und in der Punktwolke bestimmt werden. Mit den vier Punkten kann jede Drehung und auch jede Position der Kamera im Raum bestimmt werden, sofern man davon ausgeht, dass das Kalibrierungsmuster nur von einer Seite betrachtet werden kann. Alternativ ist eine Kalibrierung auch mit Linien statt mit Punkten denkbar.

- **Welche Informationen über Position und Drehung der Kamera liegen bereits vor?**

Die Kamera befindet sich zusammen mit dem 2D-Laserscanner auf der Rotationsscheibe, von der die Drehung bekannt ist. Über die Position der Kamera dagegen gibt es keine Informationen, weil von der Gehäuseposition nicht auf die Position der Bildebene oder des Brennpunktes geschlossen werden kann.

- **Können Annahmen für das Kamera-KS getroffen werden um die Kalibrierung zu vereinfachen?**

Die Drehung der Kamera auf der Rotationsscheibe kann als bekannt angenommen werden, da man davon ausgehen darf, dass die Z-Achse der Kamera längs durch das Gehäuse verläuft. Eine genaue Positionierung des Gehäuses wiederum ist problemlos möglich. Durch diese Annahme beschränkt man die Kalibrierung auf 3 Freiheitsgrade und spart somit auch noch einen Punkt ein. Demnach reichen drei Punkte des Musters zur Feststellung der Kameraposition.

- **Wie kann das Ergebnis der Kalibrierung an den ROS-Knoten für die Fusion der Daten weitergegeben werden?**

Die Lösung für die letzte Frage ist eine eigene launch-Datei zu erstellen. Das Ergebnis der Kalibrierung ist eine statische Transformation von einem Koordinatensystem in ein anderes und solche Transformationen können in einer launch-Datei von ROS definiert werden. Der Aufruf dieser launch-Datei kann eigenständig oder von einer anderen launch-Datei aus erfolgen und ist nicht an einen bestimmten ROS-Knoten gebunden. Der Aufbau der launch-Datei ist im Unterpunkt "Erstellung der launch-Datei" dieses Kapitels beschrieben.

Aus diesen Überlegungen ergab sich folgender Ablauf:

1. Bestimme die aktuelle Ausrichtung der Kamera zur Rotationsbasis.
2. Nehme alle x° Kamerabildern auf.
3. Nehme eine Punktwolke auf.
4. Markierung dreier Eckpunkte des Kalibrierungsmusters in dem Kamerabild und in der Punktwolke durch den Nutzer.
5. Führe die nachfolgenden Schritte für alle drei Achsen durch:
 1. Projiziere die markierten Punkte aus der Punktwolke auf die Bildebene der Kamera.
 2. Berechne einen Fehlerwert, der Auskunft über den Versatz beider Punktesätze zueinander gibt.

3. Wiederhole die nachfolgenden Schritte solange bis der Fehler ein Minimum erreicht hat:
 1. Verschiebe das Kamerakoordinatensystem entlang einer Achse.
 2. Projiziere erneut die markierten Punkte auf die Bildebene der Kamera.
 3. Berechne einen neuen Fehlerwert und vergleiche ihn mit dem alten Wert.
 4. Wenn der neue Fehler größer als der alte ist, dann ändere einmalig die Richtung in die geschoben wird.
 5. Ansonsten ersetze den alten Fehlerwert durch den neuen.
6. Ermittle die Transformation von der Rotationsscheibe zur Kamera aus den gesammelten Daten und speichere sie in die launch-Datei.

6.3.1 Bestimmung der Kameraausrichtung

Drehungen werden oft, so auch in tf, als Quaternionen verwaltet. Ein Quaternion kann eindeutig eine Drehung im Raum beschreiben und besteht aus vier Komponenten. Wenn Drehungen durch Yaw-Pitch-Roll-Winkel angegeben werden, müssen weitere Angaben folgen um die Drehung auch korrekt durchführen zu können, z.B. die Abarbeitungsreihenfolge oder die Zuordnung der Drehungen zu bestimmten Achsen. Die Orientierung eines Objekts wird, der Einfachheit halber, oft in Yaw-Pitch-Roll-Winkel angegeben, deshalb gibt es in tf Funktionen zur Konvertierung in Quaternionen. Für den umgekehrten Weg existieren ebenfalls entsprechende Funktionen, doch nicht immer liefern sie die gewünschten Werte.

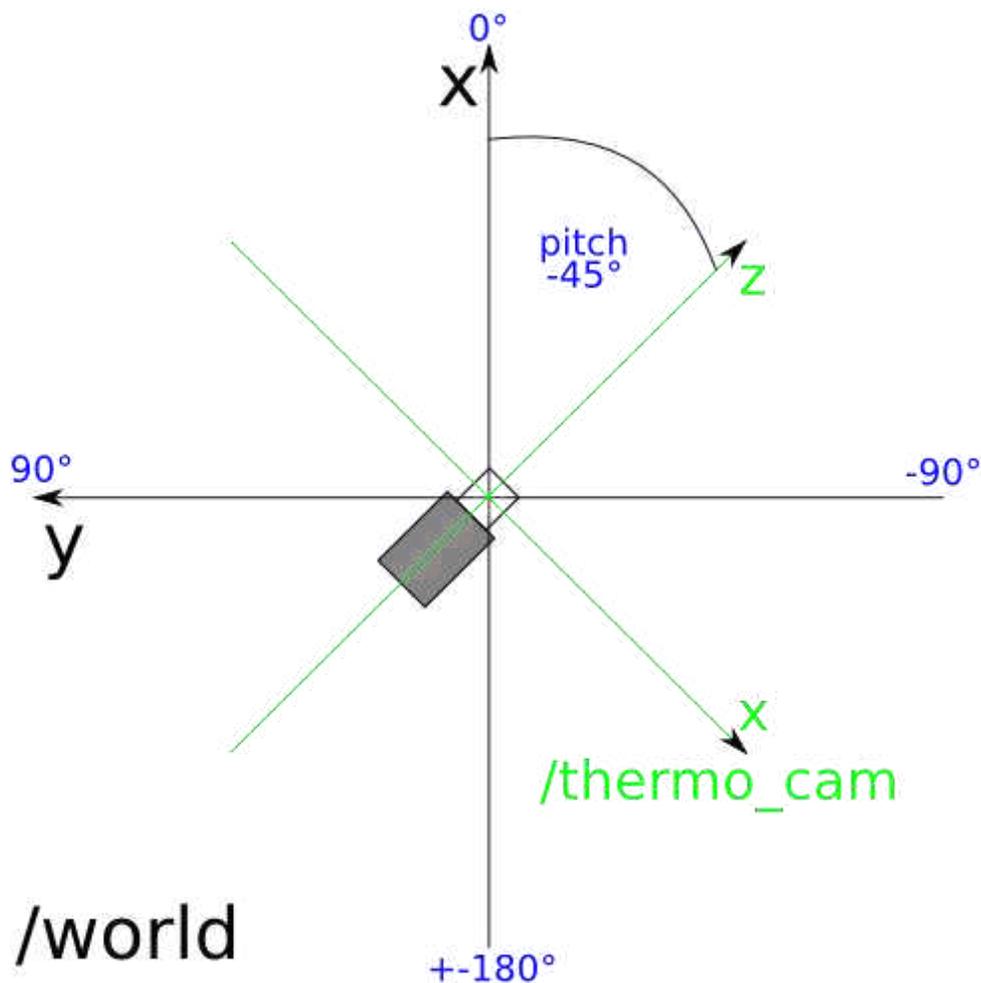
Die Kamera befindet sich nach der oben getroffenen Annahme vorerst zentral auf der Rotationsscheibe, diese Position wird nachfolgend als Initialposition bezeichnet. In der Initialposition entspricht die Ausrichtung der Kamera dem Drehwinkel um die Y-Achse des Kamerakoordinatensystems. Eine Drehung um Y ist nach der ROS-Konvention ein Pitch und kann über diese Gleichung aus einem Quaternion q berechnet werden:

$$q = (x \ y \ z \ w)^T \quad pitch = \arcsin(2(x * z - w * y))$$

Der *arcsin* gibt jedoch nur Werte zwischen $-\pi / 2$ und $\pi / 2$ zurück, alle möglichen Drehungen können trotz dieser Einschränkung angegeben werden, wenn Yaw und Roll von $-\pi$ bis π reichen³⁴. Über den Pitch-Winkel allein muss aber für diesen Fall eindeutig die Kameraausrichtung angegeben werden können. Das Ergebnis der Winkelberechnung sollte daher ein Winkel zwischen $-\pi$ und π sein. Recherchen ergaben³⁵, dass der Pitch-Winkel auch über den *atan2* berechnet werden kann.

Berechnung des Pitch-Winkels in pseudo-code:

```
pitch = 2 * atan2(y, w) + pi / 2;
if (pitch > pi)
    pitch -= 2 * pi;
else if (pitch < -pi)
    pitch += 2 * pi;
```



Die Kameraausrichtung für den einfachen Fall, d.h. die Kamera ist im Mittelpunkt der Rotationscheibe.

Diese Berechnung stützt sich auf einen Sonderfall und kann nicht allgemein eingesetzt

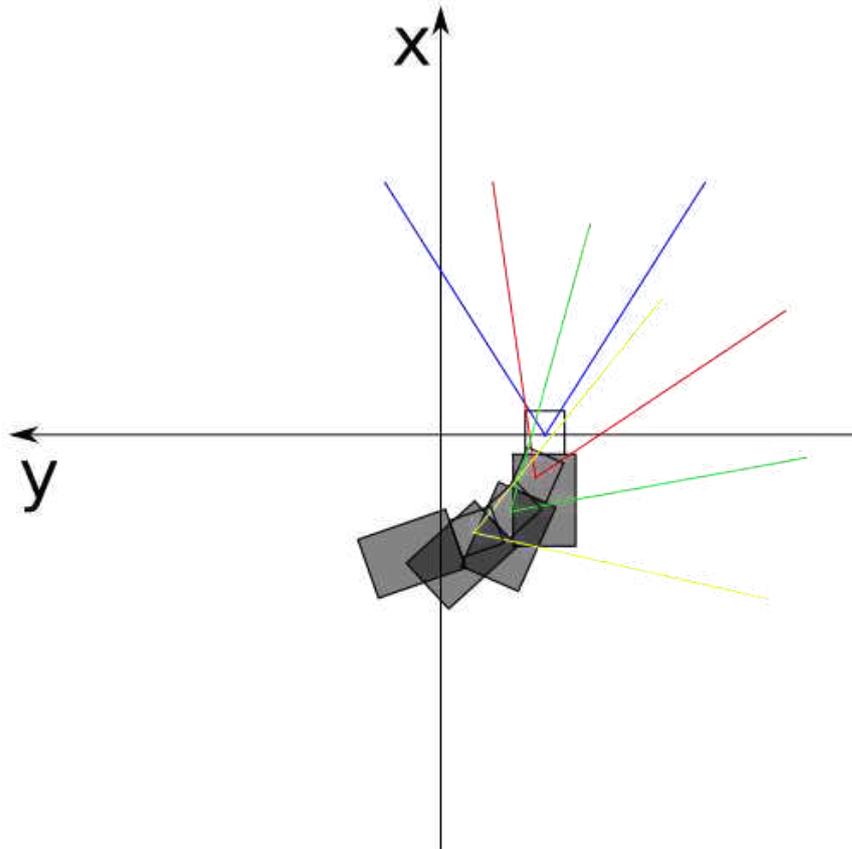
³⁴ Wikipedia-Artikel zur Umwandlung von Quaternion zu Yaw-Pitch-Roll:
http://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles (25.02.2013)

³⁵ EuclideanSpace-Unterseite über die Umwandlung von Quaternion zu Euler:
<http://www.euclideanSpace.com/math/geometry/rotations/conversions/quaternionToEuler/> (25.02.2013)

werden um den Winkel um die Y-Achse zu ermitteln. Das Resultat ist ein Winkelwert zwischen $-\pi$ und π . Zudem wurde auf den Pitch-Winkel $\pi / 2$ addiert, damit die 0° -Position der Kamera, der 0° -Position der Punktwolke entspricht. Dadurch kann der Winkel eines Punktes aus der Wolke leichter einer Kameraorientierung zugeordnet werden.

6.3.2 Aufnahmen von Kamerabildern alle x°

Der Winkel der Kamera kann nunmehr aus einem Quaternion berechnet werden. Ein passendes Quaternion wird aus der Transformation gewonnen, die aus der Multiplikation der zwei tf-Transformationen Welt-zu-Rotationsscheibe und Rotationsscheibe-zu-Thermalkamera erzeugt wird. Zur Erinnerung: Die letztere Transformation ist eine vorläufige Transformation und beinhaltet die Orientierungsänderung durch die Rotationsscheibe und die Initialposition der Kamera. Die ROS-Knoten des 3DSLKdriver-Pakets veröffentlichen regelmäßig die Transformationen zwischen Welt und Rotationsbasis, Rotationsbasis und Rotationsscheibe und auch zwischen Rotationsscheibe und Laserscanner. Die vorläufige Transformation zwischen Rotationsscheibe und Kamera bleibt innerhalb des Kalibrierungsknotens. Der Kalibrierungsknoten kann direkt die Transformation von der Welt zur Rotationsscheibe aus tf beziehen. Der 3DSLKdriver liefert nicht für jede Kameraorientierung, bzw. für jede Rotationsscheibenposition, eine Transformation, d.h. eine gewisse Toleranz muss für die Kameraaufnahme eingeplant werden. Beispielsweise kann eine Kameraaufnahme, die für 90° vorgesehen ist erst bei 91° gemacht werden, da erst dann die Transformation verfügbar ist. Dieses Verhalten fällt nicht besonders ins Gewicht, da die Kameraaufnahmen immer mit deutlicher Überlappung aufgenommen werden.



Die Grafik zeigt vier Aufnahmepositionen aus einer Gesamtdrehung mit 16 Aufnahmepositionen. Es wird zwar zu Beginn angenommen, dass die Kamera im Mittelpunkt der Rotationsscheibe liegt, trotzdem soll mit dieser Grafik gezeigt werden was tatsächlich aufgenommen wird. Mehr dazu im Kapitel "Datenfusion".

6.3.3 Markieren des Kalibrierungsmusters in der Punktwolke

Der 3DSLKdriver liefert nach jeder 180° Drehung der Rotationsscheibe eine Punktwolke mit Remissionswerten. In dieser Punktwolke befindet sich jetzt das Kalibrierungsmuster und muss durch den Nutzer markiert werden. Die Markierung könnte man zwar direkt in der Punktwolke über eine GUI vornehmen, doch dies erfordert eine aufwändige Programmierung und ist zudem fehleranfällig, wenn die Blickrichtung beliebig verändert werden kann.

Dazu ein Beispiel:

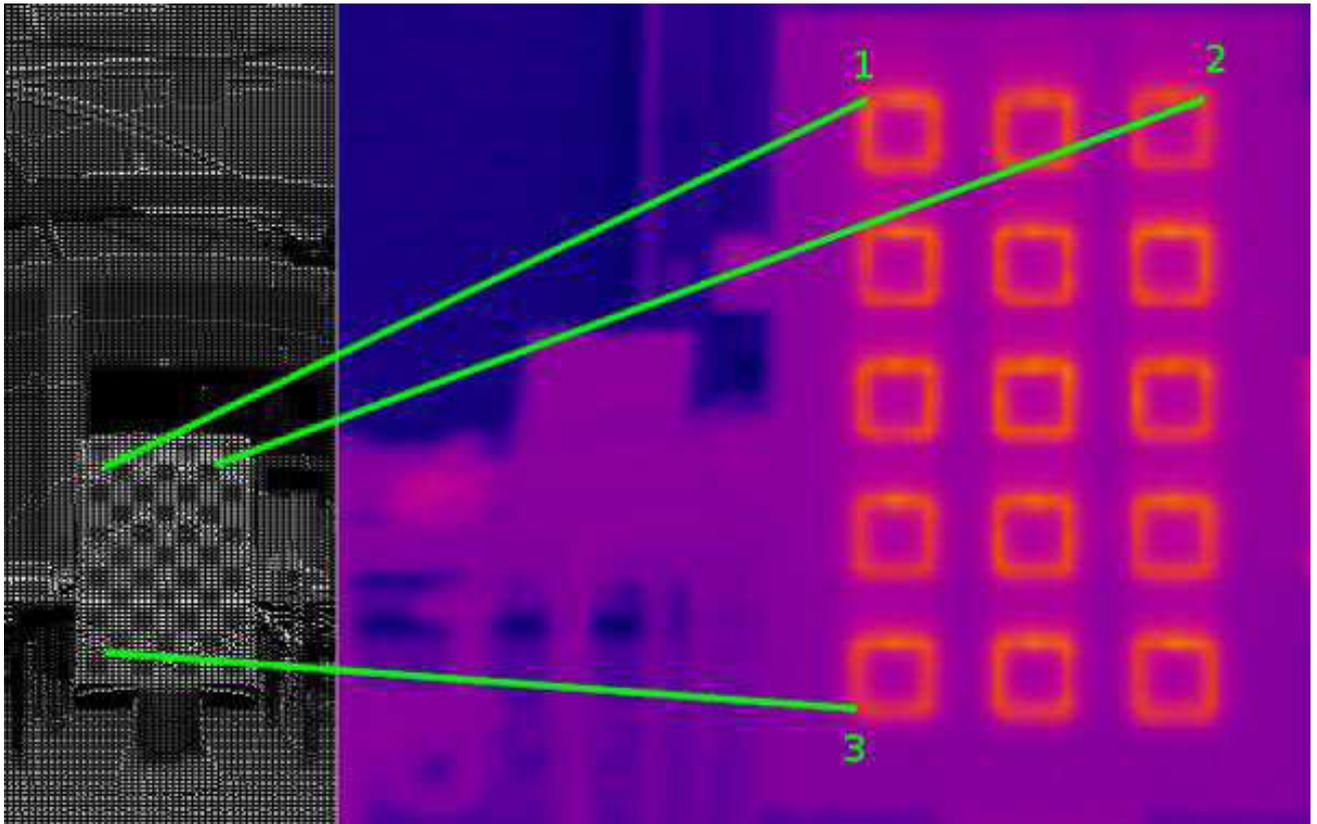
Dem Nutzer wird eine 2D-Ansicht der Punktwolke angeboten, in der das Kalibrierungsmuster zuerst gesucht werden muss. Dabei verändert der Nutzer die Perspektive und klickt, nachdem er das Muster gefunden hat, auf Punkt A. Er beabsichtigte jedoch den Klick auf B. A und B sind aus der aktuellen Perspektive für den Nutzer direkt nebeneinander, doch in der Realität liegen sie 20cm auseinander.

Das Beispiel zeigt, dass die Markierung am besten immer aus der Perspektive des Laserscanners bzw. vom Zentrum der Punktwolke aus erfolgt. Der Nutzer sollte auch nicht lange nach dem Muster suchen müssen. Die Kriterien können am einfachsten erfüllt werden, wenn die gesamte Punktwolke in ein 2D-Bild umgewandelt wird. Das Ergebnis einer solchen Umwandlung ist ein Bild, das von links nach rechts eine 360° Drehung in der Punktwolke darstellt. Nun können die Eckpunkte, die auch auf den Thermalbildern zu sehen sind, schnell und einfach in der GUI markiert werden. Im Kalibrierungsprogramm wird davon ausgegangen, dass die Punkte in der Reihenfolge: links oben, rechts oben, links unten markiert werden, siehe dazu das Bild im nächsten Abschnitt.

6.3.4 Markieren des Kalibrierungsmusters auf einem Kamerabild

Pro Umdrehung werden in der Grundeinstellung 16 Bilder aufgenommen, die alle mithilfe der intrinsischen Kameraparameter und der `image_geometry`³⁶ von ROS entzerrt werden. In der GUI klickt der Nutzer sich durch die Bilder, bis er ein Bild findet, auf dem das Kalibrierungsmuster mittig abgebildet ist. Anschließend markiert der Nutzer die gleichen Eckpunkte, die er zuvor schon in der Punktwolke markiert hat. Auch diesmal muss die gleiche Reihenfolge für die Markierungen wie bei der Punktwolke eingehalten werden. Weitere Angaben müssen nicht gemacht werden, da die Bestimmung des Versatzes beider Punktsätze automatisch abläuft.

³⁶ ROS-Seite zur `Image_Geometry`: http://www.ros.org/wiki/image_geometry (25.02.2013)



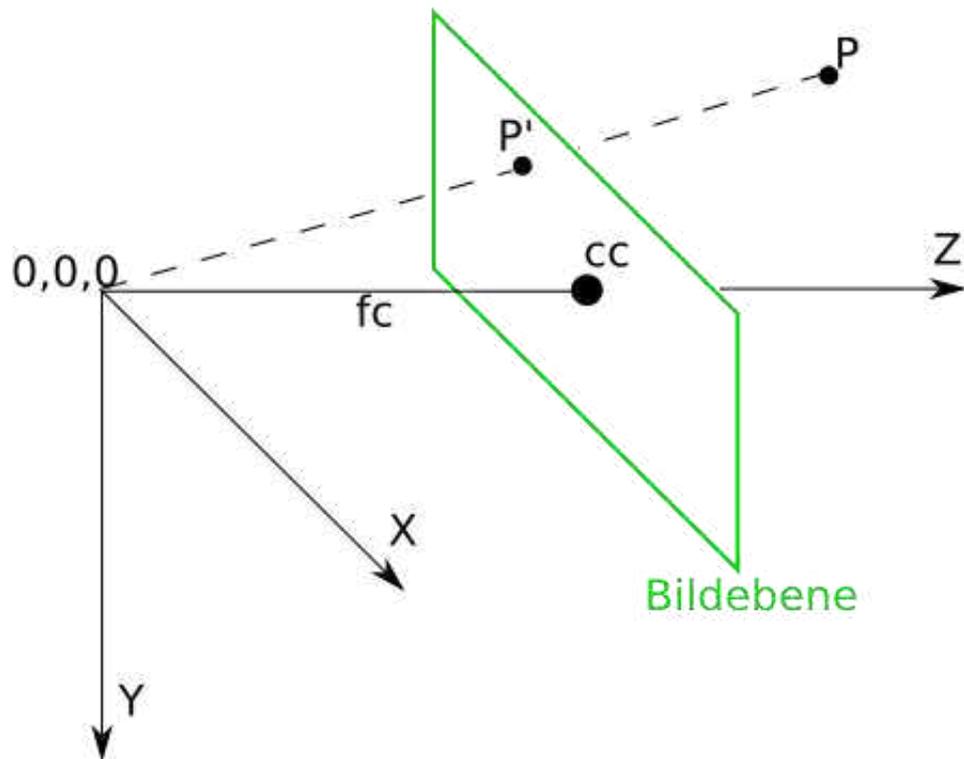
Hier sind die Eckpunkte, die für die Kalibrierung genutzt werden, von 1 bis 3 nummeriert. Es können auch andere Punkte genommen werden, doch die erste Markierung muss immer oben links, die 2. oben rechts und die 3. unten links liegen. Die Linien zeigen die zur Thermalbildaufnahme zugehörigen Eckpunkte in der Punktwolke.

6.3.5 Abbilden der 3D-Punkte auf die Bildebene

Zu jedem gespeichertem Thermalbild wurde auch die Transformation von der Welt zur Rotationsscheibe ($T_{Rot.\leftarrow Welt}$) gespeichert. Der Nutzer hatte auf eines dieser Bilder drei Eckpunkte des Kalibrierungsmusters markiert. Als Ausgangspunkt wird auch hier wieder die vorläufige Kameratransformation zusammen mit der zum ausgewählten Bild gespeicherten Transformation genutzt: ($T_{Kamera\leftarrow Rot.}$). Es kann folgendermaßen ein 3D-Punkt (P_{Welt}) aus der Wolke in das Kamerakoordinatensystem transformiert werden:

$$P_{Kamera} = T_{Kamera\leftarrow Rot.} * T_{Rot.\leftarrow Welt} * P_{Welt}$$

Der transformierte 3D-Punkt ist dabei ein Eckpunkt des Kalibrierungsmusters, und muss anschließend auf die Bildebene der Kamera projiziert werden um ihn mit seinem Pendant im Thermalbild vergleichen zu können.

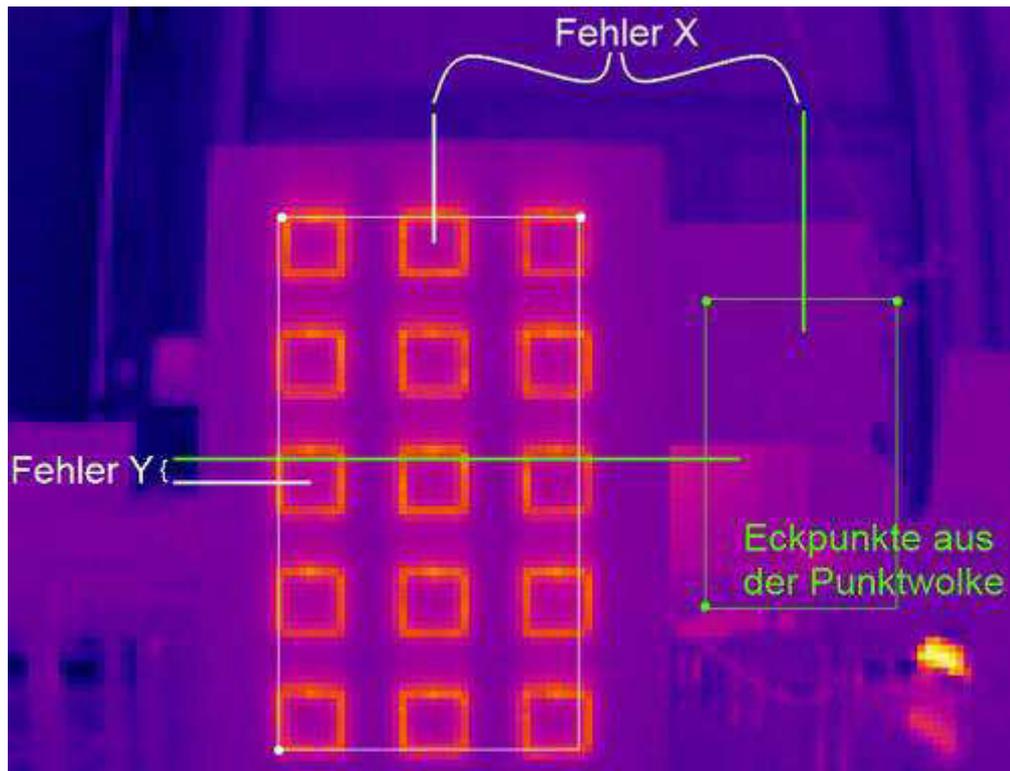


Hier nochmal zur Veranschaulichung das Kamerakoordinatensystem mit der Bildebene der Kamera. Der Punkt P entspricht nun dem Punkt P_{Kamera} und wird jetzt auf die Bildebene projiziert (Punkt P')

An dieser Stelle benötigt man wieder die intrinsischen Kameraparameter, die zusammen mit der `image_geometry` eingesetzt werden um die Projektion eines 3D-Punktes auf ein entzerrtes Kamerabild durchzuführen. Auch hier gibt es bereits Funktionen in ROS, die einem diese Arbeit abnehmen. Sind alle drei Kalibrierungspunkte aus der Punktwolke auf die Bildebene abgebildet, kann ein Fehlerwert berechnet werden.

6.3.6 Berechnung eines Fehlerwertes

Der Fehler zeigt an wie weit die Punkte beider Sensoren auseinander liegen. Die Kameraposition wird solange verändert, bis der Fehler minimal ist. Das Problem dabei ist die Gefahr bei der Minimierung in ein lokales Minimum zu laufen. Die ideale Kameraposition kann dadurch um mehrere Zentimeter verfehlt werden. Deshalb wurden für die Bewegungsrichtungen vor/zurück, links/rechts und hoch/runter jeweils eigene Fehlerwerte berechnet.



Die markierten Eckpunkte aus der Punktwolke wurden zur Fehlerberechnung auf die Bildebene projiziert. Der Fehler für die Bewegungsrichtung vor/zurück entspricht dem Größenunterschied beider Rechtecke. Die Erklärungen zu den Fehlern X und Y befinden sich weiter oben.

- **Fehlerberechnung für die Bewegungsrichtung links/rechts (Fehler X):**

Die ideale Position wird als X-Wert definiert, der in der Mitte der beiden oberen markierten Eckpunkte des Musters im Kamerabild liegt. Als tatsächliche Position wird der X-Wert genommen, der sich in der Mitte der neu auf die Bildebene projizierten oberen Eckpunkte aus der Punktwolke befindet. Der Fehlerwert ist einfach der Abstand zwischen der idealen und tatsächlichen Position.

- **Fehlerberechnung für die Bewegungsrichtung hoch/runter (Fehler Y):**

Die ideale Position wird als Y-Wert definiert, der in der Mitte der beiden linken markierten Eckpunkte des Musters im Kamerabild liegt. Als tatsächliche Position wird der Y-Wert genommen, der sich in der Mitte der neu auf die Bildebene projizierten linken Eckpunkte aus der Punktwolke befindet. Der Fehlerwert ist wieder nur der Abstand zwischen der idealen und tatsächlichen Position.

- **Fehlerberechnung für die Bewegungsrichtung vor/zurück:**

Eine Vorwärts oder Rückwärtsbewegung verkleinert oder vergrößert das auf die Bildebene projizierte Kalibrierungsmuster, deshalb sind die zur Fehlerberechnung genutzten Werte Längen. Die hierfür benutzte Länge ist der Abstand vom rechten oberen zum linken unteren Eckpunkt. Die ideale Länge wird zwischen den zuvor genannten Eckpunkten des Kalibrierungsmusters im Kamerabild gemessen und die tatsächliche Länge ist der Abstand der beiden Eckpunkte zueinander, die von der Punktwolke neu auf die Bildebene projiziert werden.

6.3.7 Erstellung der launch-Datei

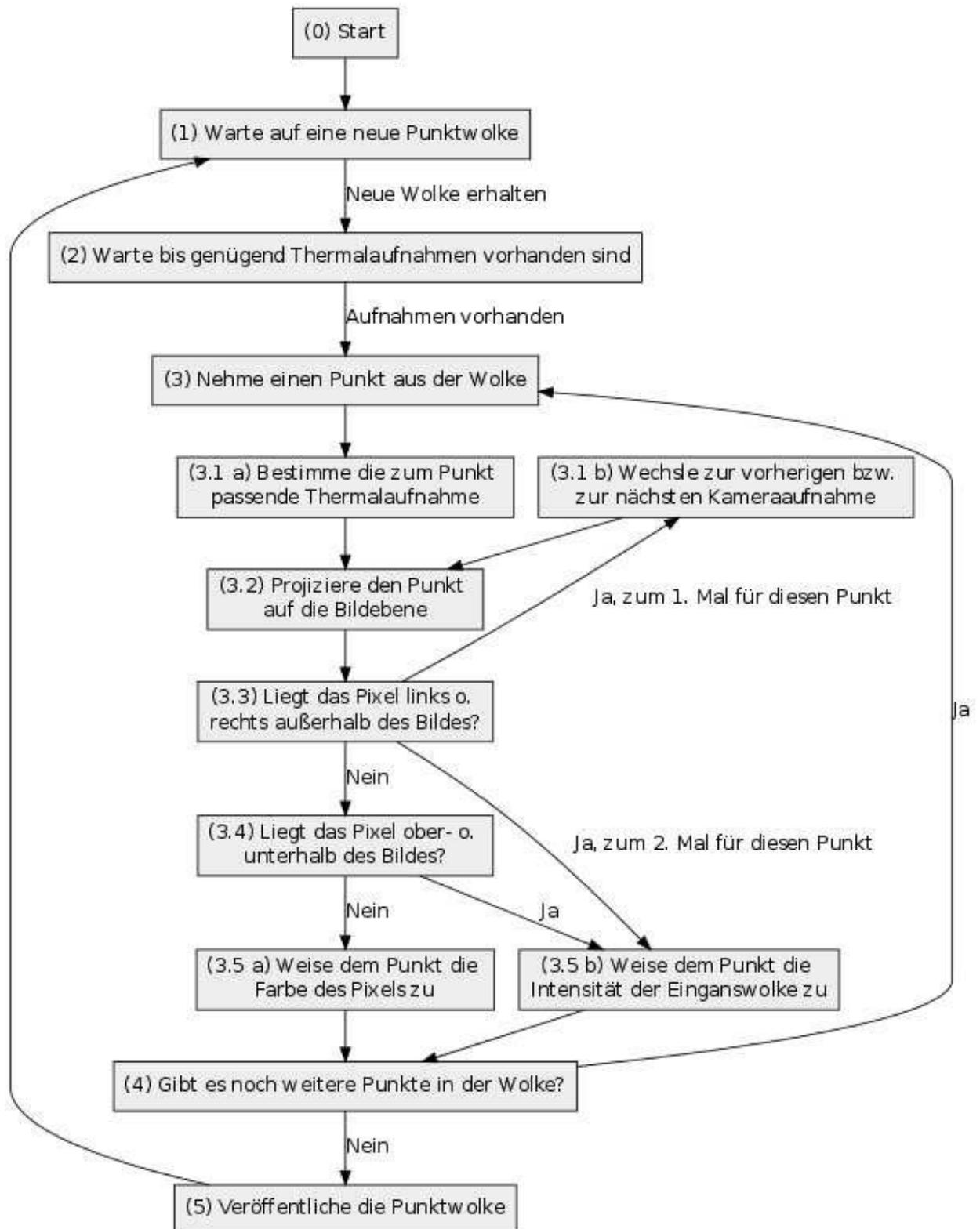
Die ROS-Komponente tf bietet einen static_transform_publisher an, der aus einer launch-Datei heraus gestartet werden kann. Eine statische Transformation wie das Ergebnis der extrinsischen Kalibrierung kann nach folgendem Muster in die launch-Datei eingetragen werden:

```
<launch>
<node pkg="tf"
      type="static_transform_publisher"
      name="cam_link_broadcaster"
      args="x y z yaw pitch roll /rotary_plate /thermo_cam 10" />
</launch>
```

Die pkg- und type-Einträge erklären sich von selbst und der darauf folgende Name ist beliebig, sollte jedoch nicht ein zweites Mal in ROS vorkommen. Die Platzhalter x,y und z sind die aus der Kalibrierung gewonnenen Positionsangaben und yaw, pitch, roll ist die als bekannt angenommene Ausrichtung der Kamera. Danach folgen der Name des Elternkoordinatensystem und dann der Name des Kindkoordinatensystems. Die „10“ gibt die Häufigkeit der Veröffentlichungen der Transformation in Millisekunden an.

7 Datenfusion

7.1 Ablauf der Fusion



Die Abbildung zeigt die Erzeugung einer Ausgangspunktwolke nur für eine bestimmte Voreinstellung. Die Erzeugung der Punktwolke ist der Hauptteil der Datenfusion und um den gezeigten Ablauf übersichtlich zu halten wurde auf die weiteren Einstellungen und Teilaufgaben verzichtet. Die nicht gezeigten Teile und die möglichen Voreinstellungen der Datenfusion finden sich jedoch in der Detailbeschreibung wieder.

7.2 Beschreibung der Teilaufgaben

7.2.1 Sammeln der Daten (Vorarbeit für (1) und (2))

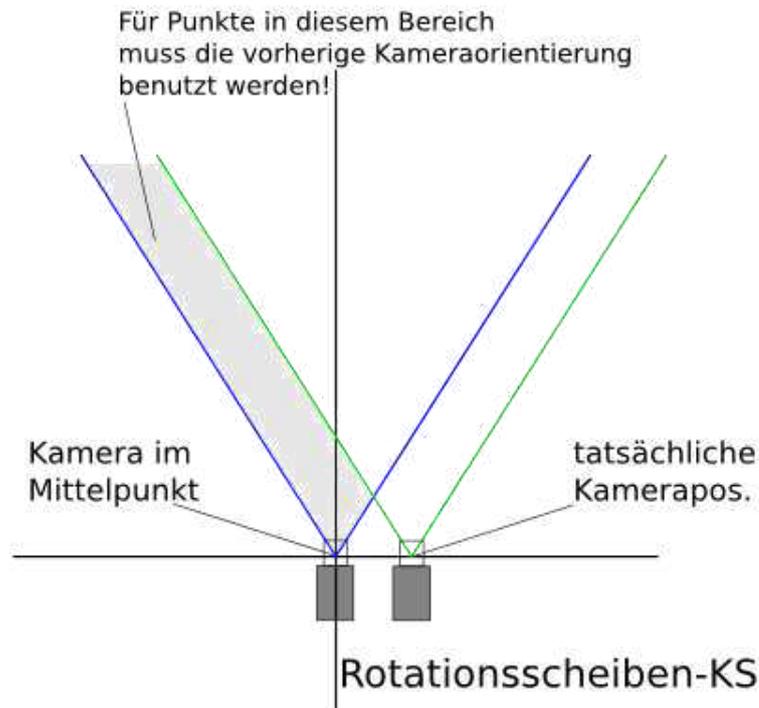
Die Kameraaufnahmen werden wie bei der extrinsischen Kalibrierung in gleichmäßigen Abständen gemacht. Zu Beginn der extrinsischen Kalibrierung traf man die Annahme, dass die Kamera im Mittelpunkt der Rotationsscheibe befestigt ist und bestimmte die Blickrichtung über den Winkel um die Drehachse der Rotationsscheibe. An dieser Stelle ist jedoch die Transformation vom Welt- zum Kamerakoordiantensystem bereits bekannt. Die Kamera befindet sich folglich nicht im Mittelpunkt der Rotationsscheibe, trotzdem kann dieselbe Winkelberechnung benutzt werden um einen immer gleichen Abstand zwischen zwei Aufnahmen einhalten zu können. Zu diesem Zeitpunkt muss nicht bestimmt werden was genau die Kamera aufnimmt. Das kann später mithilfe der tf-Transformation und der CameraInfo-Nachricht gemacht werden, die zusammen mit der Kameraaufnahme (Falschfarbenbild und Thermalwerte) vom Kamera-Knoten stammen.

Eine andere Aufgabe, die bereits hier erledigt wird, ist das Entzerren der Kameraaufnahme. Das Falschfarbenbild kann direkt mit einer Funktion aus der `image_geometry` von ROS entzerrt werden, wenn einfach das Kameramodell mit angegeben wird. Zur Erinnerung: das Kameramodell kann aus der CameraInfo-Nachricht erzeugt werden. Die Thermalwerte müssen ebenfalls korrigiert werden, da sie aus derselben verzerrten Kameraaufnahme gewonnen wurden wie das Falschfarbenbild. Erwartungsgemäß kann die `image_geometry` nur mit Bildern umgehen, d.h. die Thermaldaten müssen in ein Format gebracht werden, welches als Bild interpretiert werden kann. Die Thermaldaten-Nachricht gleicht in großen Teilen einer Bild-Nachricht, auch die Reihenfolge der Daten ist gleich. Mit Daten sind die Pixelwerte gemeint, die bei der Bild-Nachricht aus Farbwerten und bei der Thermaldaten-Nachricht aus Temperaturwerten bestehen. Die Umwandlung in ein Bildformat ist also nur ein Kopieren der Temperaturwerte und kann mit geringem Aufwand durchgeführt werden.

Neben den Kameraaufnahmen wird eine Punktwolke benötigt. Der 3DLSKdriver erstellt aus einer Vielzahl von 2D-Laserscans eine 3D-Punktwolke, die nur noch gespeichert werden muss. Die Punktwolke ist unorganisiert, d.h. die Reihenfolge der Punkte entspricht nur der Aufnahmereihenfolge und keiner besonderen Struktur. Zwischen zwei Punkten, die sich in der Realität nebeneinander befinden, liegen in der Reihenfolge der Punktwolke im schlechtesten Fall tausende andere Punkte. Im nachfolgenden Ablauf wurden deswegen auch keine Annahmen bezüglich der Reihenfolge getroffen.

7.2.2 Thermalaufnahme ermitteln (3.1)

Eine Reihe von Thermalaufnahmen wurde abgespeichert, welche sich zusammengenommen zu einer 360°-Aufnahme ergänzen. Nun wird ein 3D-Punkt aus der Wolke genommen, dabei wird immer mit dem ersten in der Wolke begonnen. Für die nachfolgenden Schritte muss für den 3D-Punkt eine Thermalaufnahme ausgewählt werden. Idealerweise beinhaltet die gewählte Aufnahme eine Abbildung des 3D-Punktes. Da sich die Kamera aber nicht im Mittelpunkt der Rotationsscheibe befindet, benötigt die genaue Bestimmung des Aufnahmebereichs ein paar Berechnungen mehr. Die zusätzlichen Berechnungen bedeuten für einen einzelnen Punkt nur einen geringen Mehraufwand, doch bei tausenden von Punkten wird der Aufwand spürbar. Der Vorgang wird beschleunigt indem nur noch die Drehung der Kamera (Kamerawinkel) für eine "ungefähre" Ermittlung des Aufnahmebereichs benutzt wird. Da die Kamera nur wenige Zentimeter vom Mittelpunkt der Rotationsscheibe entfernt befestigt ist, wird für den Großteil der Punkte die richtige Kameraaufnahme ausgewählt.



Der blaue Sichtbereich ist der geschätzte Aufnahmebereich. Nur da wo er sich mit dem grünen Bereich, dem tatsächlichen Aufnahmebereich, überschneidet kann ein 3D-Punkt auf das Thermalbild projiziert werden.

Die gewählte Aufnahme verfehlt somit den 3D-Punkt maximal um eine Aufnahmeposition. Im schlechtesten Fall muss in den anschließenden Schritten die zum 3D-Punkt zugehörige Pixelposition zweimal bestimmt werden. Hätte man komplett auf diese Schätzung verzichtet, wäre im schlechtesten Fall die zugehörige Pixelposition halb so oft bestimmt worden wie es Aufnahmen gibt. Denn nach jeder Pixelposition, die links oder rechts außerhalb lag, hätten die gleichen Berechnungen mit der nächsten bzw. vorherigen Aufnahme ein weiteres Mal durchgeführt werden müssen.

Die Bestimmung der Kameradrehung ist mit der Winkelbestimmung aus der extrinsischen Kalibrierung identisch. Bei der extrinsischen Kalibrierung wurde die richtige Aufnahme jedoch manuell bestimmt, bei der Datenfusion muss sie automatisch gefunden werden. Deshalb ist die Berechnung der Punktwinkel erforderlich. Ein Punktwinkel wird von der gleichen Position aus gemessen wie der Kamerawinkel, so kann der Punkt direkt einer Aufnahme zugeordnet werden. Für einen 3D-Punkt p kann folgender Winkel α bestimmt werden:

$\alpha = \text{atan2}(p_y, p_x)$. Der Winkel der Kameraaufnahme β und der Öffnungswinkel der Kamera γ werden benutzt um die Zugehörigkeit von Punkt zu Aufnahme festzustellen:

Überprüfung der Koordinaten:

- **Der 2D-Punkt liegt links oder rechts außerhalb des Bildes**

Falls dies zum ersten Mal für diesen Punkt der Fall ist, dann projiziere den Punkt auf die Bildebene der vorherigen bzw. nächsten Kameraaufnahme. Liegt der aktuelle Punkt bereits ein zweites Mal außerhalb, dann gibt es kein passendes Pixel für ihn. Die Kameraaufnahme kann max. um eine Aufnahmeposition daneben liegen, d.h. ein Pixel muss spätestens nach dem ersten Wechsel gefunden werden. Weitere Wechsel zwischen den Kameraaufnahmen werden verhindert, um Endlosschleifen zu vermeiden.

- **Der 2D-Punkt liegt oberhalb oder unterhalb des Bildes**

Für den Punkt gibt es kein Pixel in den Kameraaufnahmen, da die Kamera nicht nach oben oder unten geschwenkt werden kann.

Die Projektion des 3D-Punktes wird immer nur auf das Falschfarbenbild durchgeführt. Die Thermaldaten haben, wie schon früher erwähnt, den gleichen Datenaufbau. Wenn also eine Zuordnung von 3D-Punkt zu Pixel im Falschfarbenbild erfolgreich war, ist auch der zugehörige Thermalwert bekannt. Die Thermalwerte werden in derselben Reihenfolge wie die neu erzeugten 3D-Punkte (nächster Abschnitt) der Wolke gespeichert. Auch 3D-Punkte, die nicht einem Pixel aus dem Falschfarbenbild zugeordnet werden konnten, erhalten einen Thermalwert. Nur dieser Thermalwert ist dann kein gültiger Wert und liegt daher unter dem absoluten Nullpunkt um es deutlich zu machen. Bei einer späteren Verwendung von Punktwolke und Thermaldaten kann der Zusammenhang direkt über den Index hergestellt werden, d.h. ein 3D-Punkt an der Position i gehört zum Thermalwert an der Position i .

7.2.4 3D-Punkte entsprechend der Voreinstellung einfärben (3.5)

Der letzte Schritt bei der Datenfusion ist die Festlegung einer Farbe für den 3D-Punkt. Der Farbwert hängt von der Voreinstellung ab, die über eine launch-Datei vorgegeben werden kann.

Es gibt fünf Einstellungsmöglichkeiten für das Aussehen der Punktwolke:

- 1. Nur die Remissionswerte der Eingangswolke werden in die Ausgangswolke übernommen.**
- 2. Die Falschfarben der Kameraaufnahme ersetzen immer die Remissionswerte der Eingangspunktwolke.**

Wenn für einen 3D-Punkt ein Pixel existiert wird die Farbe aus der Kameraaufnahme übernommen, ansonsten wird der Remissionswert der Eingangspunktwolke als RGB-Wert in die Ausgangswolke übernommen. Diese Einstellung wurde in der Abbildung am Anfang dieses Kapitels gezeigt.

- 3. Die Falschfarben überlagern die Remissionswerte der Eingangswolke.**

Wenn für einen 3D-Punkt ein Pixel existiert wird die Farbe, unter Berücksichtigung einer einstellbaren Transparenz, über den Remissionswert gelegt, ansonsten wird wie oben der Remissionswert in die Ausgangswolke übernommen.

Das Überlagern von Farben heißt Alpha Blending und wird folgendermaßen durchgeführt:

$$C = \alpha_A A + (1 - \alpha_A) B$$

A : die erste Farbe

B : die zweite Farbe

α_A : die einstellbare Transparenz der Farbe A

C : das Resultat aus der Überlagerung von A und B

- 4. Nur die Falschfarben der Kameraaufnahme werden in die Ausgangswolke übernommen.**

Die Punkte die sich nicht auf einer Kameraaufnahme befinden werden schwarz.

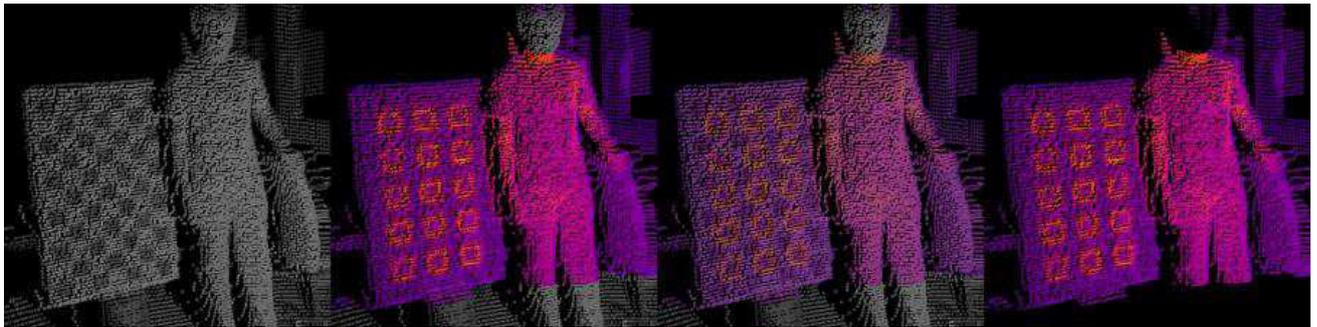
- 5. Alle Punkte werden in der Ausgangswolke schwarz.**

Remissionswerte aus Punktwolken sind Ganzzahlen von 0 bis 4000. Gegenstände aus dem alltäglichen Leben unterscheiden sich meist nur durch ein paar hundert Remissionswerte. Da ein Grauwert nur von 0 bis 255 angegeben werden kann ist ein minimaler und maximaler Wert für die Remissionswerte sinnvoll.

Bei den Voreinstellungen, die Remissionswerte benutzen, ist zusätzlich die Angabe eines Wertebereichs $[R_{min}, R_{max}]$ notwendig.

Umrechnung des Remissionswerts in einen Grauwert:

$$\text{Grauwert} = \frac{(\max(R_{min}, \min(R_{max}, \text{Remissionswert}))) - R_{min}}{(R_{max} - R_{min}) * 255}$$



Die Bilder zeigen die Einstellungen 1, 2, 3, 4 (v.l.n.r.). Die Transparenz steht für die Einstellung 2 auf 50%, am besten ist die Überlagerung von Falschfarbe und Remissionswert auf dem Kalibrierungsmuster zu sehen.

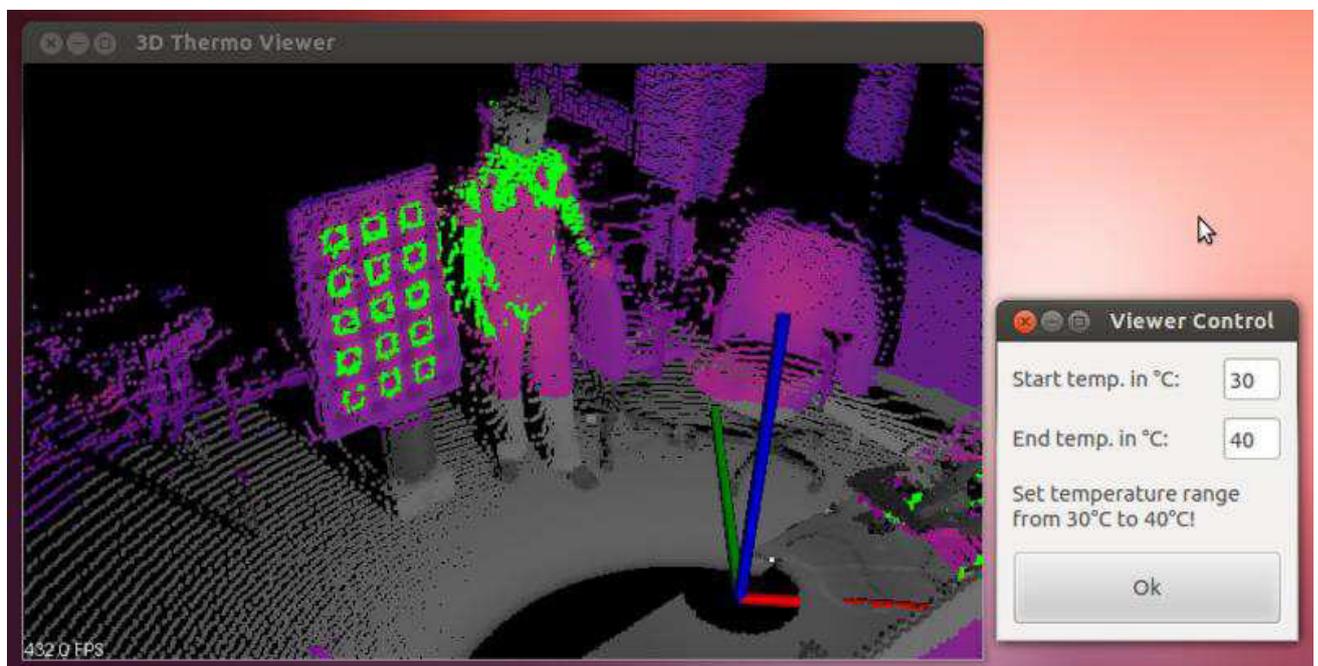
7.2.5 Punktwolke und Thermaldaten veröffentlichen (Punkt 5)

Nachdem über alle Punkte der Eingangswolke iteriert wurde, ist die Ausgangspunktwolke komplett. Neben der Ausgangswolke mit den Farbwerten existiert auch eine Ansammlung von Thermalwerten in derselben Reihenfolge wie die 3D-Punkte. Beides muss zum Abschluss der Datenfusion nur noch in ROS veröffentlicht werden. Die Punktwolke wird in ein von ROS bereitgestelltes Nachrichtenformat umgewandelt und ausgegeben. Ein einheitliches Format für Thermaldaten existiert jedoch nicht und deshalb wurde eine eigene Nachricht definiert, die sich von der ThermoData-Nachricht des thermo_cam-Knotens nur durch einen größeren Wertebereich abhebt. Die Thermaldaten werden zeitgleich mit der Punktwolke veröffentlicht und beide Nachrichten besitzen denselben Zeitstempel um eine problemlose Zuordnung möglich zu machen.

Wenn einmal ein kompletter Satz an Kameraaufnahmen (eine 360°-Drehung) und eine Eingangspunktwolke vorliegen werden die Ausgangswolke und Thermaldaten bei jeder neuen Eingangswolke erstellt/aktualisiert und veröffentlicht. Die Thermalkamera zeigt nur nach vorne während der Laserscanner nach vorne und nach hinten messen kann, das hat zur Folge, dass die Falschfarben in der Ausgangswolke und die Thermaldaten nur in einem kleinen Bereich die Aktualität der 3D-Werte besitzen.

8 Darstellung der Thermal-Punktwolke

Die Punktwolke mit den Falschfarben der Thermalkamera kann natürlich von jedem Programm dargestellt werden, das PCL-Punktwolken darstellen kann. Ein Beispiel ist das in ROS integrierte Visualisierungsprogramm rviz³⁷. Doch rviz kann nicht mit selbst erstellten Nachrichtentypen, wie dem ThermoData-Typ, umgehen. Um also die zeitgleich zur Punktwolke veröffentlichten Thermodaten auf ihre Richtigkeit überprüfen zu können war ein eigenes Visualisierungsprogramm nötig. Die Darstellung der Punktwolke geschieht mit dem PCLVisualizer und gleicht somit der Darstellung in rviz. Der Unterschied zu rviz ist die GUI, die es erlaubt einen Temperaturbereich anzugeben. Die Punkte, welche nach den Daten der ThermoData-Nachricht eine Temperatur innerhalb dieses Bereiches aufweisen, werden grün hervorgehoben. Diese Visualisierung ist nicht nur hilfreich beim Debuggen, sie zeigt auch eine mögliche Anwendung des Ergebnisses dieser Arbeit. Das Hervorheben einzelner Temperaturbereiche kann bei der Analyse der Thermaltaufnahmen eine wichtige Rolle spielen. Die Analyse von Thermaltaufnahmen war jedoch nicht Teil dieser Arbeit, weshalb das entstandene Visualisierungsprogramm nicht mehr als ein einfaches Beispiel für die Verwendung der erzeugten Daten ist.



Das Visualisierungsprogramm mit dem eingestellten Temperaturbereich von 30 bis 40°C.

³⁷ Visualisierungsprogramm rviz von ROS: <http://www.ros.org/wiki/rviz> (15.03.2013)

9 Evaluierung

9.1 Bekannte Probleme

Diese Fehler haben ihre Ursache nicht in der im Laufe dieser Bachelorarbeit entstandenen Software. Die Ursache liegt in der Fremdsoftware, die zum Teil nicht optimal auf die verwendete Hardware abgestimmt war. Für die fehlerhafte Software gibt es derzeit keine Alternativen, deshalb muss mit den nachfolgend genannten Problemen gerechnet werden.

9.1.1 Rotationseinheit

Die Angabe von Geschwindigkeit und Drehrichtung für die Rotationseinheit geschieht im rotary_drive-Knoten (ein Teil des 3DLSKdriver) mithilfe eines Software-Motorcontrollers. Dieser Motorcontroller kommuniziert über eine serielle Schnittstelle mit der Positioniersteuerung des Motors. An dieser Stelle tritt das Problem auf, der Motorcontroller sendet anscheinend in unregelmäßigen Abständen fehlerhafte Telegramme an die Positioniersteuerung. Damit kann die Positioniersteuerung nichts anfangen und man erhält einen CRC-Fehler, der in der Konsole ausgegeben wird:

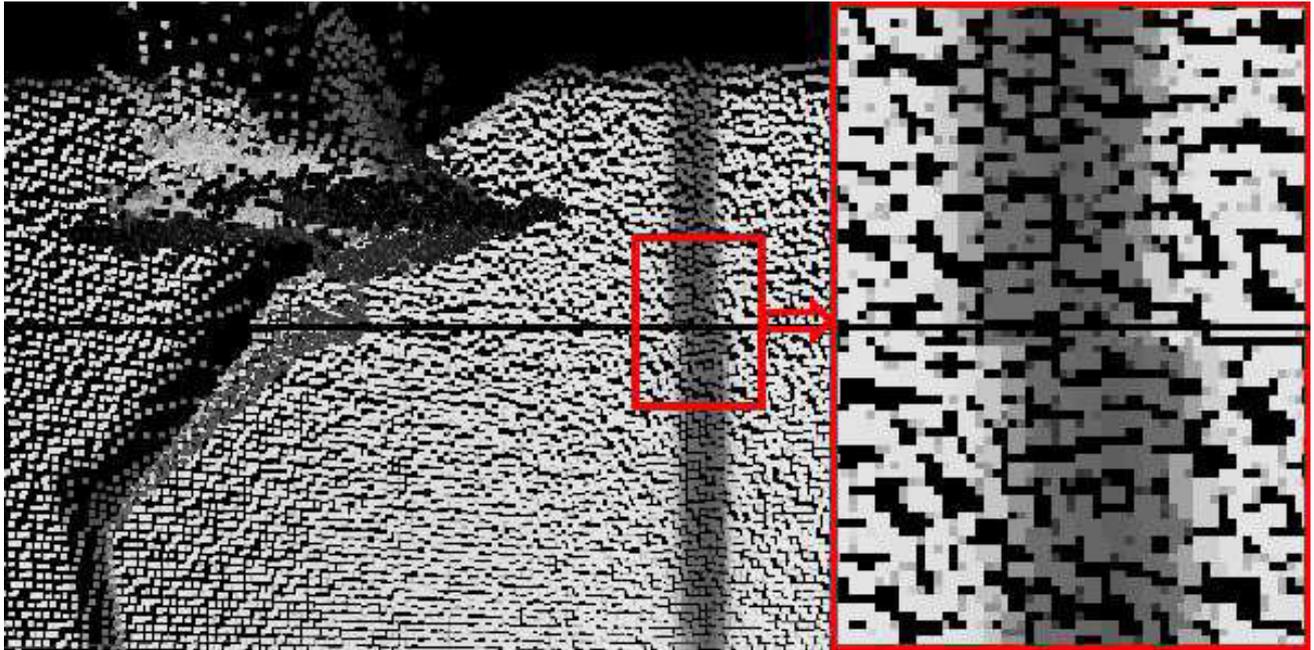
```
CRC test FAI LED!
*** get Actual Position: ReadObject() returned -1 **
EPOS not ready, reply was: 0xffffffff86
*** ReadObject: problems with sendCom(), return value was -1 ***
*** get Actual Position: ReadObject() returned -1 **
readBYTE: Invalid argument
EPOS says: 0x01 | CRCerror!
*** ReadObject: problems with sendCom(), return value was -1 ***
*** get Actual Position: ReadObject() returned -1 **
readBYTE() returned -1 at sendCom line 3254
ERROR: no reply from EPOS received, is it online?
EPOS not ready, reply was: 0x77
*** ReadObject: problems with sendCom(), return value was -1 ***
*** get Actual Position: ReadObject() returned -1 **
```

Der Fehler ist auf unterschiedlichen Computern reproduzierbar, jedoch tritt er bei manchen Computern nur selten oder sogar überhaupt nicht auf. Auch scheint der CRC-Fehler unabhängig von der benutzten Ubuntu-Version aufzutreten. Für diesen Fehler existiert noch keine Lösung (07.03.2013), es gibt lediglich einen Workaround, der die weitere Benutzung der Rotationseinheit nach einem Fehler erlaubt. Dafür wurde ein "exit()" aus dem Quellcode entfernt, das nach einem Fehler das Programm beenden hätte. Trotzdem kann der rotary_drive-Knoten, der in regelmäßigen Abständen die aktuelle Position des Motors über den Motorcontroller bezieht, während eines Fehlers die Motorposition nicht bestimmen.

Infolgedessen wird auch keine tf-Transformation und auch keine ROS-twist-Nachricht in dieser Zeit veröffentlicht. Der 3DLSKdriver mit dem Workaround befindet sich auf dem svn-Server zusammen mit den anderen Knoten aus dieser Arbeit.

9.1.2 Punktwolken-Erzeugung

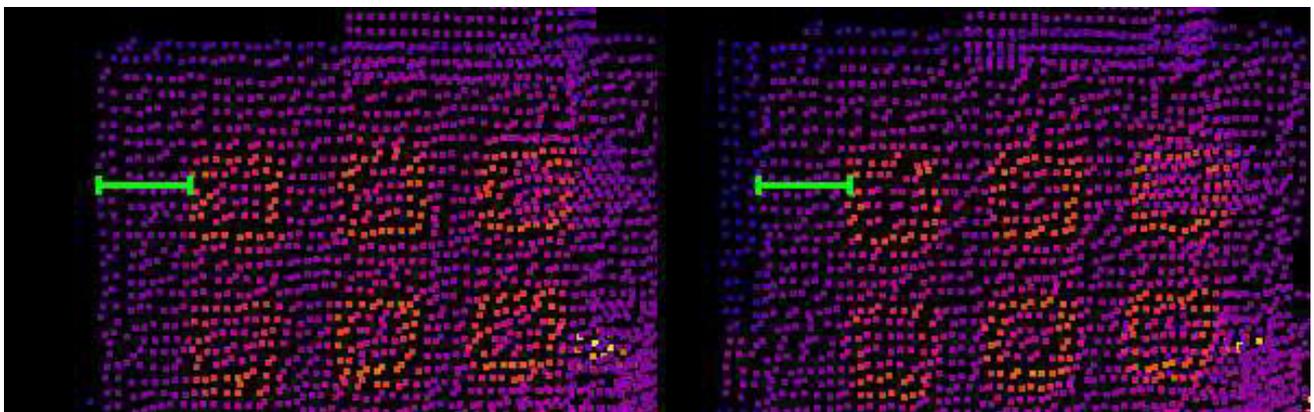
Bei der Erzeugung von Punktwolken durch den scan2cloud-Knoten (ebenfalls ein Teil vom 3DLSKdriver) werden die Positionen der einzelnen 2D-Scans von einem Startpunkt aus erzeugt. Der Startpunkt wechselt nach jeder erzeugten Punktwolke seine Position, weil für eine 360°-Punktwolke nur eine 180°-Drehung erforderlich ist. Im Idealfall liegt eine Startposition bei 0°/360° oder bei 180° und nach einer 360°-Drehung sollten zwei gleichgroße Punktwolken vorliegen. In der Realität sind die beiden erzeugten Wolken jedoch nicht gleich groß. Genauer gesagt, die Aufnahmedauer der ersten Punktwolke ist um den Teil kleiner wie sie bei der darauf folgenden Punktwolke größer ist. Der Startpunkt wird immer neu an das Ende einer Aufnahme gesetzt. Alle 3D-Punkte verschieben sich nach jeder Wolke, entsprechend der Startposition, erst ein Stück nach links bzw. rechts bevor sie in der nächsten Punktwolke wieder auf der Ausgangsposition landen. Deshalb macht es den Anschein, dass sich die aufgenommenen Objekte bewegen obwohl keine Veränderung in der Wirklichkeit stattgefunden hat.



Ein Beispiel für den Versatz zwischen zwei Punktwolken. Oben ist die 1. Punktwolke und darunter ist die danach aufgenommene Wolke. Der rechts vergrößerte Bereich ist die Strebe einer Stellwand, an der man deutlich die Verschiebung erkennen kann.

9.1.3 Fusion der Thermalbilder mit der Punktwolke

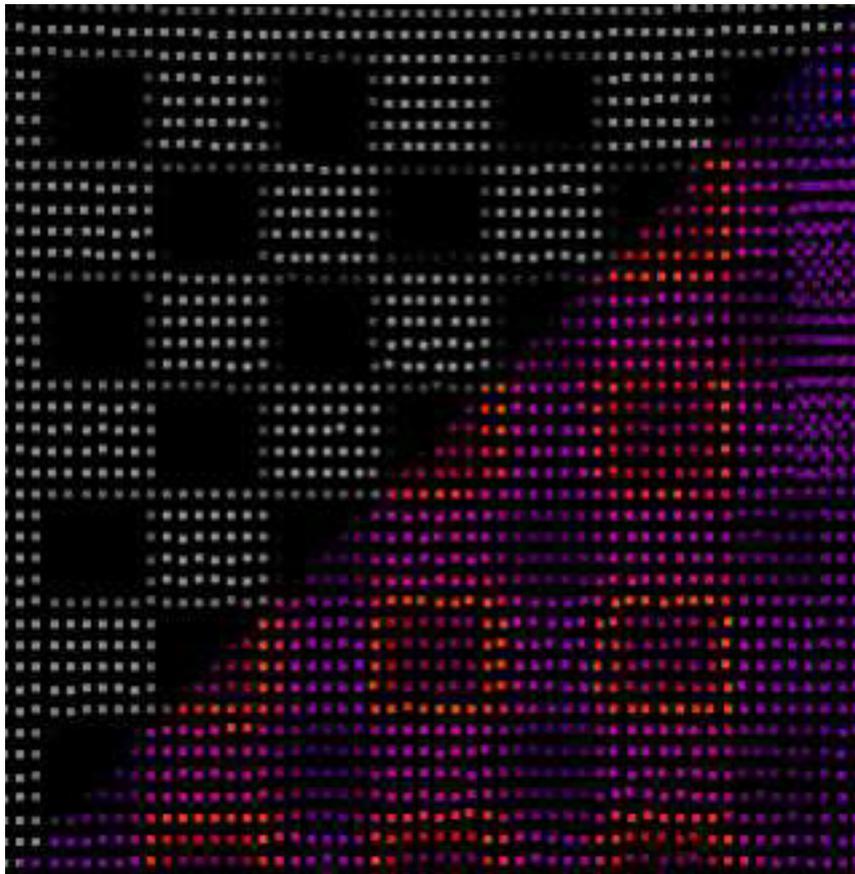
Jede der erzeugten Punktwolken für sich genommen ist trotz des o.g. Fehlers, eine korrekte Umgebungsaufnahme. Eine extrinsische Kalibrierung wird jedoch nur für eine Punktwolke durchgeführt, d.h. nur bei jeder zweiten Punktwolke stimmen Thermalwerte mit den 3D-Punkten überein. Anders ausgedrückt heißt das, wenn sich die Punktwolke verschiebt bleiben die Kamerapositionen konstant. Dieser Effekt ist auf dem folgenden Bild hervorgehoben.



Die Punktwolke hat sich verschoben, aber die Falschfarben nicht. Anhand der grünen Markierung sieht man, dass sich der Abstand zum Rand des Kalibrierungsmusters vergrößert hat.

9.2 Passgenauigkeit der Fusion

Abgesehen von den oben beschriebenen Fehlern hängt die Passgenauigkeit der Thermaldaten zur Punktwolke von der Genauigkeit der extrinsischen Kalibrierung ab. Der Nutzer wählt je drei Punkte aus der Punktwolke und aus dem Thermalbild, worauf die Kalibrierung durchgeführt wird. Es ist unwahrscheinlich, dass der Punkt aus der Punktwolke an derselben Stelle auf dem Kalibrierungsmuster liegt wie das gewählte Pixel im Thermalbild. Sollte der Nutzer jedoch entgegen der Erwartung eine perfekte Markierung des Musters vorgenommen haben, können die Thermaldaten trotzdem abweichen. Der Grund dafür liegt darin, dass es zwar für jede Markierung im Thermalbild ein Pixel gibt, die Markierung in der Punktwolke jedoch zwischen zwei Punkten liegen kann. Für die Kalibrierung wird aus der Punktwolke immer der Punkt genommen, der der Markierung am nächsten liegt. Die Temperaturwerte und die Falschfarben liegen hingegen immer korrekt übereinander, da sie mit denselben Parametern entzerrt werden. Anders gesagt heißt das, wenn die Falschfarben nicht zur Punktwolke passen, dann passen die Temperaturwerte ebenfalls nicht.



Ein Beispiel für eine gute Kalibrierung. Die Aufnahme mit den Falschfarben lässt zu 40% die Remissionswerte durchscheinen, dadurch lässt die die Passgenauigkeit besser erkennen.

10 Zusammenfassung und Ausblick

10.1 Zusammenfassung

In dieser Arbeit wurde eine Methode entwickelt um Thermalbilder mit den Punktwolken eines Laserscanners zu kombinieren. Mit einer Kombination oder Fusion ist die Zuordnung von Wärmeinformationen (Falschfarbe und Temperaturangabe in °C) zu einem 3D-Punkt aus der Wolke gemeint. Dafür musste zunächst eine Möglichkeit gefunden werden eine Thermalkamera so zu kalibrieren wie es von optischen Kameras bekannt ist. Mehrere Arten von Kalibrierungsmustern standen zur Auswahl und wurden gegeneinander abgewogen. Schlussendlich wurde ein Muster aus Widerstandsdrähten erstellt, wie es so in der Literatur noch nicht beschrieben wurde. Das Muster erfüllt gleich zwei Eigenschaften, zum einen kann es problemlos mit niedrig auflösenden Thermalkameras erkannt werden und zum anderen ist es im Remissionsbild des Laserscanners sichtbar. Die Kamerakalibrierung (intrinsische Kalibrierung) selbst wurde mithilfe der MATLAB Camera Calibration Toolbox durchgeführt, da diese sich schon in anderen Arbeiten³⁸ bewährt hat und zudem ohne Anpassungen mit dem Kalibrierungsmuster umgehen konnte.

Neben der Ermittlung von intrinsischen Parametern der Kamera war die Bestimmung der Kameraposition (extrinsische Kalibrierung) ein wichtiger Schritt auf dem Weg zur Fusion von Thermalbildern und Punktwolken. Es wurde eine extrinsische Kalibrierung erstellt, die aus den Markierungen des Nutzers im Kamerabild und in der Punktwolke die Position ohne großen Aufwand bestimmen kann. Da jedoch ein paar Annahmen in Bezug auf die verwendete Hardware gemacht worden sind, ist das entstandene Programm nicht universell einsetzbar. Trotz den gemachten Annahmen kann das Kalibrierungsprogramm ohne große Änderungen mit den meisten rotierenden Thermalkamera-Laserscanner-Kombinationen eingesetzt werden.

Vor der Arbeit an der Zusammenführung der Daten wurde der, ursprünglich von Herrn Wilkes entwickelte, Thermalkameratreiber erweitert. Die Funktionserweiterung sorgt dafür, dass die Unterschiede zu den anderen ROS-Kameratreibern kleiner werden ohne dass die zusätzlichen Funktionen der Thermalkamera eingeschränkt werden. Zu den gemachten Änderungen zählen auch kleiner Verbesserungen, die speziell für diese Arbeit sinnvoll oder

³⁸ Richard Hanten: "Fusion von 3D-Laserscanner-Daten mit 2D-Bilddaten", 08/2011

auch zwingend notwendig waren.

Der Hauptteil dieser Arbeit bildet die Fusion von Thermalbildern mit Punktwolken. Mehrere Thermalaufnahmen, die zusammen eine Rundumsicht bilden, werden auf die Punktwolke des Laserscanners gelegt. Das Programm wurde so gestaltet, dass festgelegt werden kann ob und wie die Falschfarben der Thermalkamera in die Punktwolke aufgenommen werden sollen. Die Temperaturwerte werden aber auf jeden Fall mit 3D-Punkten in Zusammenhang gebracht. Die Punktwolke und die Thermalinformationen können von jedem beliebigen 3D-Sensor bzw. von jeder beliebigen Thermalkamera stammen, die einzige Einschränkung ist, dass sich die Thermalkamera um die Z-Achse des Punktwolken-Koordinatensystems drehen muss.

Zu guter Letzt ist noch ein Anzeigeprogramm für die Punktwolken und Thermalwerte entstanden. Die Punktwolke wird, wie man es aus anderen Visualisierungsprogrammen gewohnt ist, mit der schematischen Darstellung ihres Koordinatensystems angezeigt. Über die vorhandene GUI kann der Nutzer einen Temperaturbereich vorgeben, der hervorgehoben werden soll. Dieses Anzeigeprogramm soll gleichzeitig ein Beispiel für die Verwendung der in dieser Arbeit erstellten Daten sein

10.2 Verbesserungsmöglichkeiten

Intrinsische Kamerakalibrierung

Ein eigener ROS-Knoten für die Kamerakalibrierung würde den Umgang erleichtern und die derzeit noch nötige "Handarbeit" auf ein Minimum reduzieren. Außerdem steht nicht jedem, der eine Thermalkamera kalibrieren will, MATLAB zur Verfügung. Möglicherweise reicht es hierfür einen der verfügbaren Kalibrierungsknoten anzupassen anstatt ganz von vorne zu beginnen.

Extrinsische Kalibrierung

Für die extrinsische Kalibrierung muss der Nutzer die Position des Kalibrierungsmusters manuell angeben. Dies kann automatisiert werden und wurde nur deshalb nicht automatisiert, weil die Kalibrierung nur ein einziges Mal durchgeführt werden muss. Wenn jedoch bereits ein Kalibrierungsknoten wie oben beschrieben erzeugt wurde, ist damit ein Teil dieser Aufgabe schon getan. Der nächste Schritt, die Erkennung des Musters in der Punktwolke, fällt infolgedessen vermutlich leichter.

Eine weitere Verbesserung wäre die selbstständige Bestimmung der Kameraausrichtung auf der Rotationsscheibe, denn noch wird sie als bekannt angenommen und muss vor der Kalibrierung angegeben werden.

Aufnahmebereich der Thermalkamera

Die Thermalkamera ist fest mit der Rotationsscheibe verbunden und kann nicht bewegt werden. Wenn die Kamera auf eine Nick-Vorrichtung befestigt wird, kann ein ähnlich hoher Aufnahmebereich erreicht werden wie beim Laserscanner. Somit können für alle 3D-Punkte in der Punktwolke Thermaldaten erfasst werden. Eine komplette Aufnahme der Umgebung würde dann jedoch mehrerer Umdrehungen bedürfen.

Weitere Kamera

Eine optische Kamera, die neben der Thermalkamera angebracht wird, kann hilfreich für eine Objekterkennung sein. Die Kombination von optischen Kamerabild und Punktwolke unterscheidet sich nur geringfügig von der Variante mit Thermalbild, deshalb kann ähnlich vorgegangen werden.

10.3 Ausblick

In diesem Abschnitt werden Anregungen für Arbeiten oder Projekte gemacht, die auf diese Arbeit aufbauen.

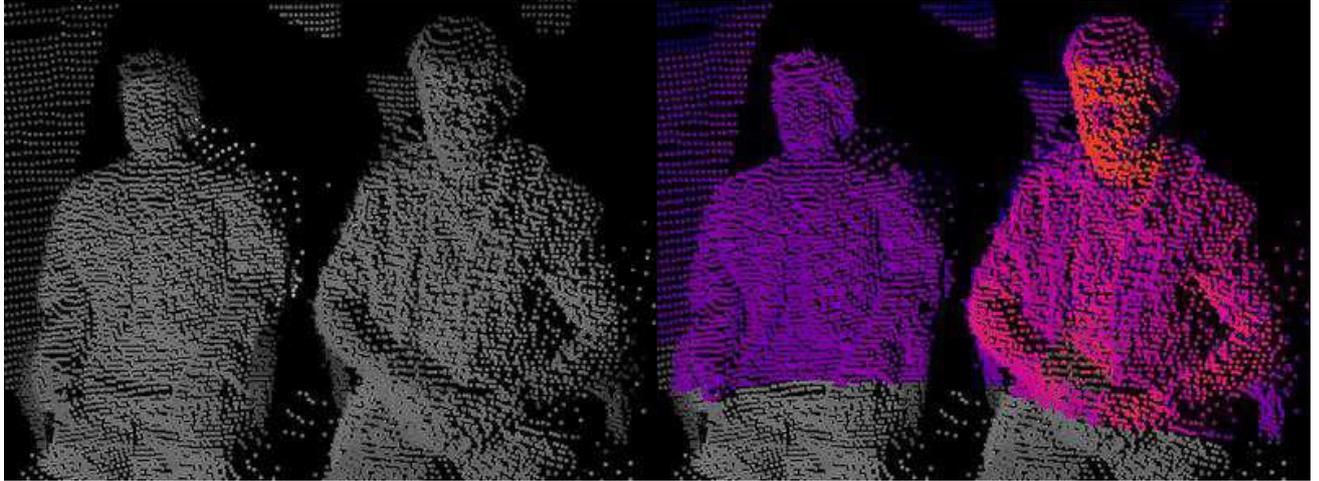
Nachteile ausgleichen

Aus einer Kombination von Thermalbildern und Punktwolken können verschiedenste Informationen gewonnen werden. Manche Gegenstände sind weder auf einem Thermalbild noch in der Punktwolke eindeutig zu identifizieren, z.B. Objekte aus Glas. Eine Glasscheibe erscheint auf dem Thermalbild als undurchsichtige Wand, während dieselbe Scheibe in der Punktwolke oft überhaupt nicht zu finden ist. Erst die Kombination beider Informationen ermöglicht die Erkennung als Glasscheibe.

Erkennung von Menschen

Ein Mensch kann innerhalb einer Punktwolke relativ gut erkannt werden, sofern er steht oder sitzt. Wenn der Mensch jedoch am Boden liegt, z.B. wegen eines Sturzes, gestaltet sich die Erkennung deutlich schwieriger. Die Extremitäten sind zum Teil verdeckt und können dadurch nicht mehr erkannt werden.

Ein Thermalbild kann Abhilfe schaffen wenn sich die Körpertemperatur von der Umgebungstemperatur unterscheidet. Über die Fusion mit der Punktwolke kann die genaue Position und Größe bzw. Umrise des Menschen festgestellt werden.



Der Vergleich von Dummy und Mensch als Anregung für den Einsatz der 3D-Thermal-Aufnahme. Hier sieht man, dass erst die Fusion von Thermaldaten und Punktwolke eine Unterscheidung möglich macht.

11 Anhang

11.1 Einrichtung für den ersten Start

- Installation des Betriebssystems Ubuntu 12.04
- Installation des Roboter Betriebssystems ROS Fuerte
- Installation des **thermo_cam_2**-Knotens:
 - Den ROS-Knoten auschecken: https://fileserv.informatik.fh-gelsenkirchen.de:3711/roblab/TLiebelt/BA/thermo_cam_2
 - Das Paket libudev-dev mit "rosdep install thermo_cam" oder mit "sudo apt-get install libudev-dev" installieren.
 - Den thermo_cam_2-Knoten mit "rosmake" kompilieren.
 - Den Ordner /imager aus dem Ordner /Config nach /usr/lib/imager/ kopieren und darauf achten, dass man danach Leserechte für den Ordner /usr/lib/imager/ und alle Unterordner/Dateien besitzt.
 - Die Datei 88-optrisimager-usb.rules aus dem Ordner /Config in den Ordner /etc/udev/rules.d kopieren.
 - Den aktuellen Nutzer der Gruppe video hinzufügen: "sudo usermod -aG video <<username>>"
- Auschecken und Kompilieren des Quellcodes(möglichst in dieser Reihenfolge)
 - die **RCPRG_laser_drivers** (https://github.com/RCPRG-ros-pkg/RCPRG_laser_drivers)
 - und die während dieser Arbeit erstellten bzw. bearbeiteten ROS-Pakete (<https://fileserv.informatik.fh-gelsenkirchen.de:3711/roblab/TLiebelt/BA>):
3DLSKdriver, extrinsic_calibration, thermo_cloud, thermo_3d_viewer
- Installation von Matlab 5.x oder höher
- Einrichtung des Matlab Camera Calibration Toolbox entsprechend der Anleitung (http://www.vision.caltech.edu/bouguetj/calib_doc/index.html#start)

11.2 Hinweise für die Benutzung der Software

- **Intrinsische Kalibrierung**

Für die intrinsische Kalibrierung müssen mehrere Thermalaufnahmen gespeichert werden, dafür kann der thermo_cam_2-Knoten zusammen mit dem image_view-Knoten (standardmäßig in ROS vorhanden) benutzt werden. Der Start beider Knoten geschieht über die Launch-Datei "thermo_cam_w_viewer.launch" innerhalb des thermo_cam_2-Pakets. Die Thermalaufnahmen werden bei einem Klick auf das angezeigte Bild in den Ordner gespeichert, der in der Launch-Datei angegeben wird:

```
...
<node pkg="image_view" type="image_view" name="viewer">
  <param name="filename_format" value="/home/USER/images/frame%04i.jpg"/>
  <remap from="image" to="thermo_cam/image_raw"/>
</node>
```

- **Extrinsische Kalibrierung**

Für die extrinsische Kalibrierung müssen zuvor folgende Knoten gestartet werden: thermo_cam_2, LMS100, 3DLSKdriver_rotary_drive und 3DLSKdriver_scan2cloud. Zusätzlich muss der static_transform_publisher eingesetzt werden um die Transformationen vorzugeben. Das alles übernimmt die Launch-Datei "extrinsic_calibration.launch" aus dem extrinsic_calibration-Paket.

Wenn beim Einsatz des extrinsic_calibration-Knoten selbst nach mehreren Umdrehungen kein Fenster für die Markierung der Punkte im Kamerabild aufgeht, ist die Aufnahmetoleranz auf einen höheren Wert einzustellen. Die Toleranz gibt an wie weit die Aufnahmeposition von der Vorgabe abweichen darf, sie hat jedoch keinen Einfluss auf die Passgenauigkeit der Fusion.

- **Fusion der Daten**

Auch für die Datenfusion existiert eine entsprechende Launch-Datei, die im Paket thermo_cloud unter dem Namen "thermo_cloud.launch" zu finden ist. Die Ausgabe von Thermalpunktewolken und Temperaturwerten erfolgt frühestens nach einer 360°-Drehung und sollte spätestens nach der 2. Umdrehung vorliegen. Erfolgte danach immer noch keine Ausgabe gilt der gleiche Hinweis zur Toleranzeinstellung wie bei der extrinsischen Kalibrierung, besonders wenn die Rotationsgeschwindigkeit erhöht wurde.

- **Anzeigen der Punktwolken**

Der thermo_3d_viewer wird direkt mit rosrund gestartet und bedarf keiner Konfiguration.

11.3 QT-Beispiel

Die grafischen Oberflächen dieser Arbeit wurden mit QT und Eclipse erstellt und während der Einarbeitungsphase ist dafür ein Beispielprogramm entstanden. Das Beispielprogramm ist ein einfacher ROS-Knoten, der mit QT Kamerabilder auf den Bildschirm anzeigt. Von der ROS-Seite wird das Kamerabild in die QT-GUI gesendet wenn über einen Button in der GUI ein neues Bild angefordert wurde, dadurch wird auf einfache Weise die Verwendung des Signal-Slot-Konzepts in ROS gezeigt.

Bei der Verwendung von QT in ROS ist zu beachten, dass viele ROS-Komponenten Boost einsetzen und sich die Schlüsselworte von QT und Boost überschneiden. Wenn man Boost parallel zu QT benutzen will muss in der CMakeList folgende Zeile hinzufügen:

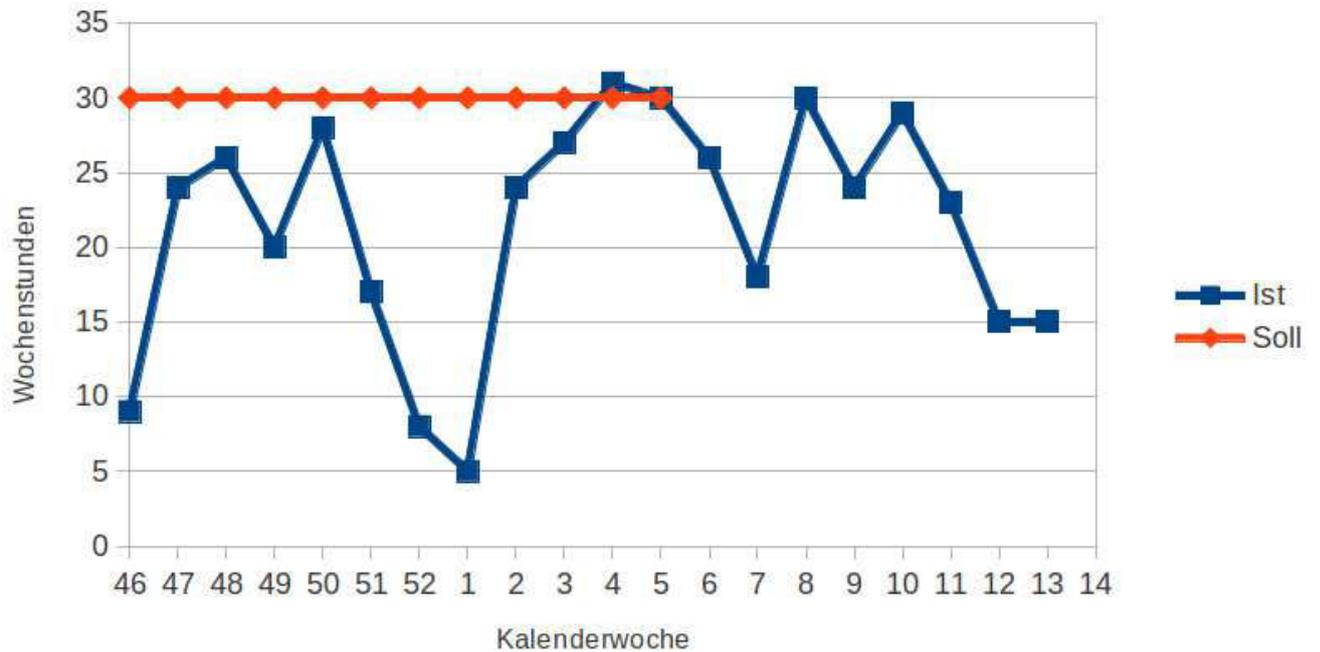
```
ADD_DEFINITIONS(-DQT_NO_KEYWORDS)
```

Damit stehen die QT-Schlüsselworte wie "emit" oder "signals" nicht mehr zur Verfügung und müssen durch "Q_EMIT", "Q_SIGNALS", etc. ersetzt werden. Außerdem ist es möglich die grafische Oberfläche im QT-Creator zu erstellen und die UI-Datei in der CMakeList anzugeben um sie im Programm nutzbar zu machen. Dafür sind diese Zeilen nötig:

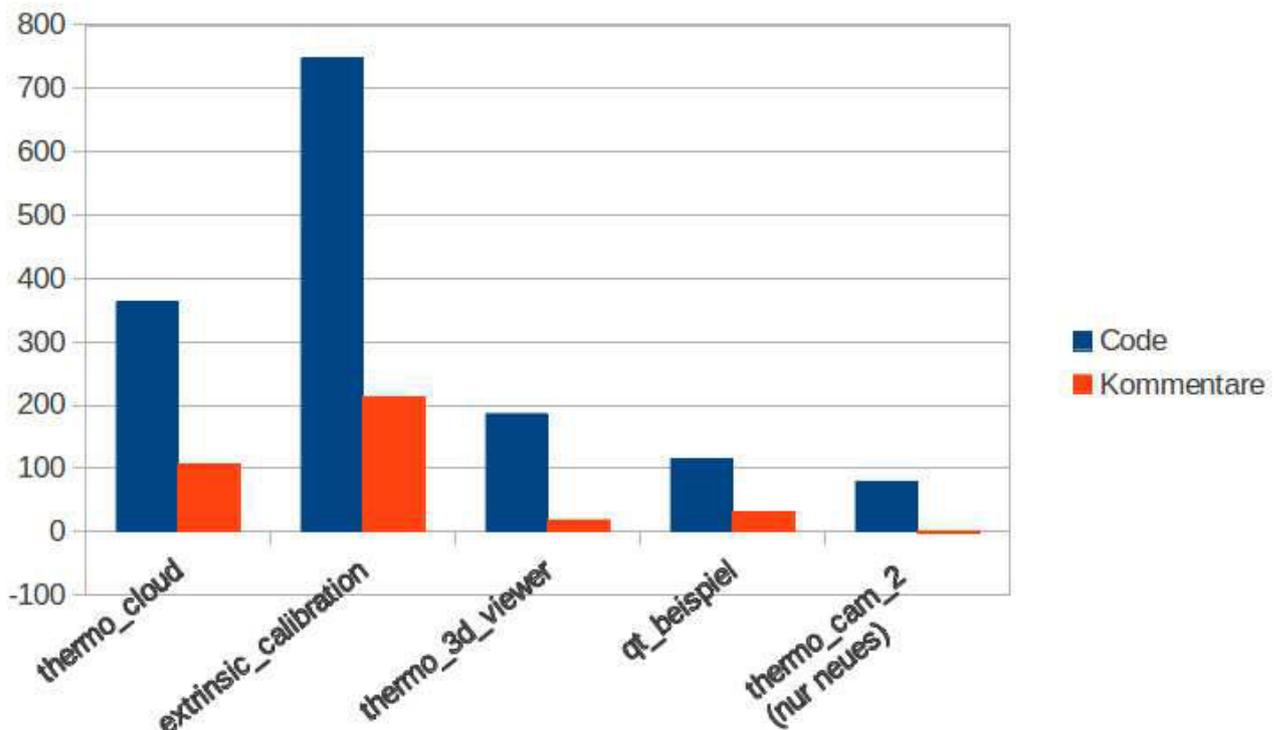
```
QT4_WRAP_UI(uis_h gui/beispielGUI.ui)  
include_directories(${CMAKE_CURRENT_BINARY_DIR})
```

Der QT-Beispielknoten befindet sich auf dem SVN: <https://fileserv.informatik.fhgelsenkirchen.de:3711/roblab/TLiebelt/BA>

11.4 Aufwandsanalyse



Zeitaufwand. Insgesamt ca. 430 Stunden. (Ohne Kolloquium)



Programmiertechnischer Aufwand. Die Anzahl der Code-/Kommentarzeilen.

12 Bildquellen

Wikipedia-Artikel zu Wärmebildkamera:

<http://de.wikipedia.org/wiki/W%C3%A4rmebildkamera> (27.03.2013)

Artikel "Wärmebildkamera bewährt sich" der Feuerwehr Fleisbach: <http://feuerwehr-fleisbach.de/uploads/presse/W%C3%A4rmebildkamera%20bew%C3%A4hrt%20sich.pdf> (27.03.2013)

OpenCV Dokumentation 2.4.9:

http://docs.opencv.org/trunk/doc/tutorials/calib3d/camera_calibration/camera_calibration.html, 07.01.2013

Ranjith Unnikrishnan und Martial Hebert: "Fast Extrinsic Calibration of a Laser Rangefinder to a Camera", Robotics Institute. Paper 339. - 2005

Project Thermalmapper: <http://www.faculty.jacobs-university.de/anuechter/thermalmapper.html> , 07.01.2013

Surya Prakash, Pei Yean Lee und Antonio Robles-Kelly: "Stereo techniques for 3D mapping of object surface temperatures", QIRT Journal Volume 4 - 1/2007

Dorit Borrmann, Hassan Afzal, Jan Elseberg und Andreas Nüchter: "Mutual Calibration for 3D Thermal Mapping", 2011

Wikipedia-Artikel zu Verzeichnungen: <http://de.wikipedia.org/wiki/Verzeichnung> (25.02.2013)