Messung und Auswertung der räumlichen Strahleigenschaften eines Femtosekundenlasers

von
Sommer Daniel, Mat.# 01325734

Diplomarbeit

zur Erlangung des akademischen Grades
Diplomingenieur

eingereicht an der
Technischen Universität Wien

Betreuer
Liedl Gerhard, Ao.Univ.Prof. Dipl.-Ing. Dr.techn.

Mitbetreuer
Fürbacher Roland, Projektass. Dipl.-Ing. BSc

Institut für Fertigungstechnik und Photonische Technologien

Fakultät für Maschinenwesen und Betriebswissenschaften

Wien, April 2022

Abstract

Zusammenfassung

Laser sind komplexe Geräte, die in vielen Bereichen der Technik nicht mehr wegzudenken sind. Egal in welchem Bereich sie verwendet werden ist es notwendig, dass die Eigenschaften des Lasers kontrolliert und für die jeweilige Anwendung optimiert werden. Damit das möglich ist, müssen die Strahleigenschaften des Lasers bekannt sein. Es gibt verschiedene Methoden wie diese bestimmt werden können. Eine davon basiert auf der Methode der zweiten Momente, welche in der DIN EN ISO 11146 beschrieben ist. Diese Methode verwendet die mit einer pixelbasierten Kamera aufgenommenen Strahlprofile um und in der Strahltaille eines Laserstrahls und berechnet daraus die Strahleigenschaften des Lasers. Im Zuge dieser Arbeit und auf Basis der Norm wurde ein Messaufbau und eine automatisierte Mess- und Auswertungssoftware entwickelt, mit der die Eigenschaften von stigmatischen und leicht astigmatischen Strahlen bestimmt werden können. Um die Ergebnisse der automatisierten Messung zu verifizieren wurden die aufgenommenen Strahlprofile mit Brandlöchern verglichen. Dabei konnte festgestellt werden, dass das Verhältnis der Strahlabmessungen des Messsystems und der Brandlöcher proportional zum Verhältnis der Brennweiten der verwendeten Linsen ist und bestätigt, dass sowohl der Messaufbau, als auch die automatisierten Mess- und Auswertungssoftware funktioniert. Für den Messaufbau ist zu beachten, dass die verwendete Kamera, die optischen Elemente und die mechanische Komponenten speziell für einen Ti:Saphir Femtosekundenlaser ausgewählt wurden. Sollte ein anderer Laser das Ziel der Messung sein, muss der Messaufbau mit passenden Komponenten adaptiert werden.

Abstract Englisch

Lasers are complex devices that have become indispensable in many areas of technology. No matter in which field they are used, it is necessary that the properties of the laser are controlled and optimized for the respective application. For this to be possible, the beam properties of the laser must be known. There are several methods how these can be determined. One of them is based on the second moment method, which is described in DIN EN ISO 11146. This method uses the beam profiles around and in the beam waist of a laser beam taken with a pixel-based camera and calculates the beam properties of the laser from them. In the course of this work and based on the standard, a measurement setup and automated measurement and evaluation software were developed to determine the properties of stigmatic and slightly astigmatic beams. To verify the results of the automated measurement, the recorded beam profiles were compared with burn holes. It was found that the ratio of the beam dimensions of the measurement system and the burn holes is proportional to the ratio of the focal lengths of the lenses used, confirming that both the measurement setup and the automated measurement and evaluation software work. For the measurement setup, it should be noted that the camera, optical elements and mechanical components used were specifically selected for a Ti:Sapphire femtosecond laser. If another laser is the target of the measurement, the measurement setup must be adapted with suitable components.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Verschleiß ist in jedem Bereich der Technik ein Thema das nicht ignoriert werden kann. Dadurch ist es notwendig jede Maschine oder Einrichtung regelmäßig zu warten, damit die langfristige Funktionalität sichergestellt ist. Auch bei Lasern ist das eine Notwendigkeit. Sowohl das Lasermedium als auch die notwendigen Peripherie im Laseraufbau verschleißen mit der Zeit und müssen ersetzt oder nachjustiert werden. Die Komplexität des Lasers ist proportional zum Justierungsaufwand, da die optischen Elemente alle voneinander abhängen, wodurch bereits kleine Änderungen ungewollte Schwierigkeiten bereiten können. Um die Justierung zu vereinfachen bzw. die derzeitigen Eigenschaften des Lasers zu bestimmen existiert das Verfahren der Laserstrahlcharakterisierung. Es ist ein Verfahren bei dem die Strahleigenschaften in Ausbreitungsrichtung bestimmt werden. Kennt man diese Eigenschaften, ist es möglich eine Aussage über den derzeitigen Zustand des Lasers zu treffen und auch festzustellen ob die Justierung die gewünschten Verbesserungen bzw. Änderungen gebracht hat.

1.2 Licht

Die Eigenschaften von Licht werden meistens mit vereinfachten Modellen beschrieben. Eine der ersteren Vorstellungen war es, dass Lichtquellen wie z.B. die Sonne Lichtteilchen auf geradlinigen Bahnen aussenden. Bekannt ist diese Theorie als Teilchenmodell des Lichts [1]. In einem einfachen Experiment mit einer Lochblende und einer Lichtquelle wurde versucht einen scharf begrenzten Strahl herzustellen um das Teilchenmodell zu bestätigen. Der Versuch ergab, dass es nach der Lochblende bedingt durch Beugungseffekte des Lichts zu einer Aufweitung des Strahls gegenüber dem Lochdurchmessers kam. Dieses Verhalten kann durch das Wellenmodell des Lichts erklärt werden. Beide Modelle können mit komplexer Mathematik zu einem einheitlichen Modell zusammengefasst werden. In der Realität reicht für die meisten Beobachtungen jeweils eines der Modelle.

1.2.1 Lichtwellen

Im Wellenmodell [2] wird das Licht als transversale elektromagnetische Welle beschrieben, in der die elektrische Feldstärke $\mathbf{E}$ und die magnetische Feldstärke $\mathbf{H}$ mit gleicher Frequenz f schwingen. Die beiden Vektoren stehen stets senkrecht aufeinander siehe Abb. 1.1. Die Wellenlänge λ des Lichts ist dabei der Abstand zweier benachbarter Phasenflächen. Es gilt mit der Lichtgeschwindigkeit im Vakuum $c = 2.998 \times 10^8\,\mathrm{m\,s^{-1}}$ der Zusammenhang

$$c = \lambda \cdot f. \tag{1.1}$$

In einem transparenten Medium breitet sich das Licht mit der reduzierten Lichtgeschwindigkeit c' aus. Diese reduzierte Lichtgeschwindigkeit hängt vom Brechungsindex n ab und kann mit

$$c' = \frac{c}{n} \tag{1.2}$$

bestimmt werden. Der Brechungsindex n kann mit der Dielektrizitätszahl ϵ und der Permeabilität μ über

$$n = \frac{1}{\sqrt{\epsilon\mu}} \tag{1.3}$$

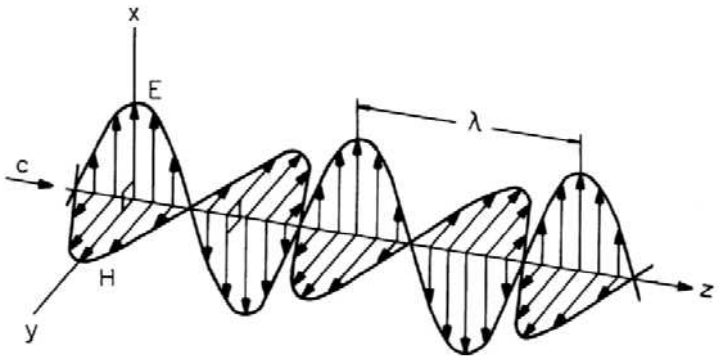


Abbildung 1.1: Verlauf der elektrischen Feldstärke **E** und der magnetischen Feldstärke **H** in Ausbreitungsrichtung einer Lichtwelle [2]

bestimmt werden. Trifft das Licht auf eine Grenzfläche siehe Abb. 1.2 zwischen zwei optischen Medien mit Brechungszahlen n_1 und n_2 gilt das Brechungsgesetz nach Snell

$$n_1 \sin \alpha = n_2 \sin \beta. \tag{1.4}$$

Der Winkel α ist der Eintrittswinkel des Primärstrahls bezogen auf die Grenzflächennormale, α'

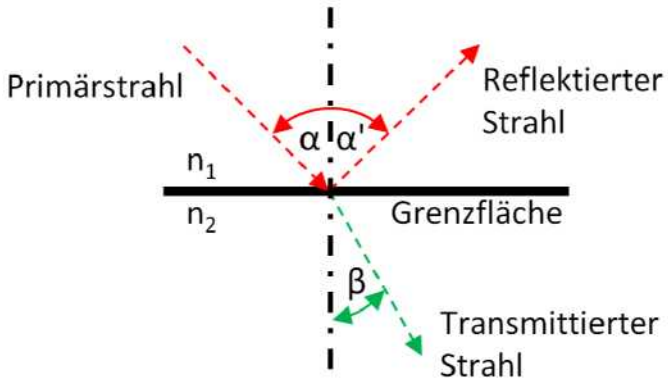


Abbildung 1.2: Brechungsgesetz

der Winkel der reflektierten Strahls wobei gilt

$$\alpha' = -\alpha \tag{1.5}$$

und β der Winkel des transmittierten Strahls. Sowohl der Winkel α, als auch der Winkel β sind bezogen auf die Grenzflächennormale. Beide Winkel werden ausgehend von dieser Normale gemessen. Schließt z.B. α ausgehend von der Normale einen Winkel gegen den Uhrzeigersinn ein, so schließt auch β einen Winkel gegen den Uhrzeigersinn ein.

1.2.2 Polarisation und Farbe des Lichts

Grundsätzlich beschreibt die Polarisation des Lichts die Schwingungsrichtung einer sich transversal ausbreitenden Welle. Man bezieht sich dafür immer auf den Vektor der elektrischen Feldstärke. Ist die Richtung der elektrischen Feldstärke **E** konstant, so spricht man von linear polarisierten Licht. Die meisten Lichtquellen bestehen aus einem Gemisch von Wellen unterschiedlicher Polarisation und werden daher als unpolarisiert bezeichnet. Die Farbe des Lichts ist abhängig von der Frequenz bzw.

der Wellenlänge. Das menschliche Auge reagiert unterschiedlich empfindlich auf die verschiedenen Wellenlängen des Lichts siehe Abb. 1.3. Ist das Auge auf eine helle Umgebung eingestellt $V'(I)$ reagiert es auf Licht mit einer Wellenlänge von 550 nm am empfindlichsten. Ist es auf dunkle Umgebung eingestellt verschiebt sich die Empfindlichkeit weiter nach links zu 500 nm. Der für das

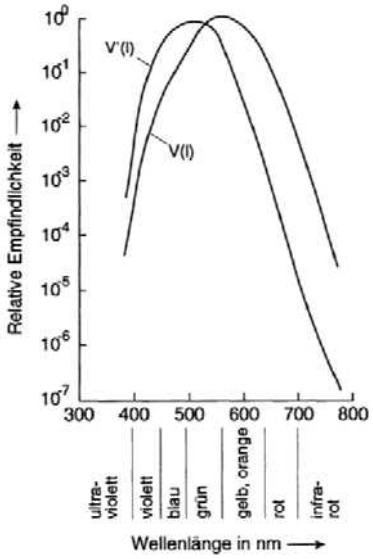


Abbildung 1.3: relative Lichtempfindlichkeit des menschlichen Auges [2]

menschliche Auge sichtbare Bereich liegt zwischen 380 nm und 780 nm.

1.2.3 Absorption von Licht

Beschrieben wird Absorption mithilfe einer ebenen Lichtwelle der Intensität I_0 die auf eine absorbierende Stoffschicht mit Dicke d trifft [2]. Am hinteren Ende der der Stoffschicht beträgt die aus der Schicht austretende Intensität I. Die verminderte Intensität I ist proportional zu I_0, hängt von der Schichtdicke ab und kann mit

$$I = I_0 e^{-\alpha d} \tag{1.6}$$

auch bekannt als Lambert-Beer'sches Gesetz beschrieben werden. Führt man eine Koordinate x ein und nimmt an, dass die Intensität an einer Stelle $I(x)$ um dI abnimmt ergibt sich für die Ableitung des Lambert-Beer'schen Gesetzes

$$dI = -\alpha I x dx. \tag{1.7}$$

Der Faktor α ist ein materialabhängiger Absorptionskoeffizient und beträgt z.B. für Glasfaser $\alpha \approx 1$ bis 10 km oder für Metalle $\alpha \approx 1$ nm.

Der Vorgang der Absorption kann mithilfe diskreten Energiezuständen $(E_1, E_2, E_3, ..., E_n)$ dargestellt werden (siehe Abb .1.4). Im nicht angeregten oder auch ungestörten Zustand nehmen alle Atome den Zustand der niedrigsten Energie E_1 ein. Dieser Zustand wird auch als Grundzustand bezeichnet. Trifft Licht mit einer Frequenz f_{12} auf ein Atom, ist es möglich das das Atom in einen höheren Energiezustand E_2 übergeht, solange die Bohrsche Frequenzbedingung

$$E_2 - E_1 = h f_{12} \tag{1.8}$$

erfüllt wird, wobei $h = 6.626^{-34}$ J s das Plancksche Wirkungsquantum beschreibt. Im Gegenzug bedeutet das, dass dem Licht die Energie hf_{12} entzogen wurde und somit auch die Intensität I

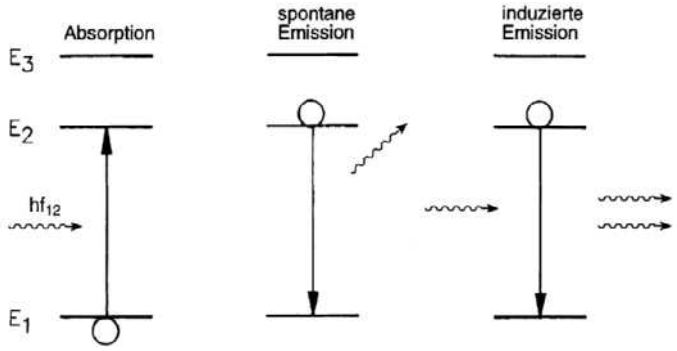


Abbildung 1.4: Schematische Darstellung von Absorption, spontaner Emission und induzierter Emission [2]

vermindert. Aus der Anzahl der Übergänge von Zustand E_1 nach E_2 lässt sich der Absorptionskoeffizient berechnen. Sie kann pro Volumen- und Zeiteinheit definiert werden mit

$$\left|\frac{dN_1}{dt}\right|_a = \sigma_{12} N_1 \Psi, \tag{1.9}$$

wobei N_1 die Dichte der Atome im Grundzustand, Ψ die Stromdichte (Photonen/Zeit·Fläche) und σ_{12} die Wirkfläche der Absorption beschreibt. Der Index a symbolisiert das die Übergänge durch Absorption entstehen. Sie ist gleich der Änderung der Photonendichte (Photonen/Volumen) $d\Phi/dt$. Mit der Stromdichte Ψ und der Änderung der Schichtdicke über der Zeit kann die Photonendichte ausgedrückt werden mit

$$\Phi = \Psi \frac{dt}{dx}. \tag{1.10}$$

Daraus ergibt sich der Zusammenhang

$$\left|\frac{dN_1}{dt}\right|_a = -\frac{d\Psi}{dx}. \tag{1.11}$$

Da die Photonendichte abnimmt ist es mit einem negativen Vorzeichen versehen. Mit der Definition der Intensität

$$I = h f_{12} \cdot \Psi, \tag{1.12}$$

und Gl. 1.11 ergibt sich für die Abnahme der Intensität durch Absorption

$$\left|\frac{dI}{dx}\right|_a = -\sigma_{12} N_1 I. \tag{1.13}$$

Durch Vergleich mit Gl. 1.7 kann der Absorptionskoeffizient gefunden werden mit

$$\alpha = \sigma_{12} N_1. \tag{1.14}$$

Aus dieser Beziehung lässt sich erkennen, dass der Absorptionskoeffizient proportional zur Dichte der absorbierten Atome ist.

1.2.4 Spontane Emission

Angeregte Atome gehen nach einer gewissen Zeit wieder in ihren Zustand der niederen Energie über [2]. Dabei kann Energie in Form eines Lichtquants in eine beliebige Richtung abgestrahlt werden. Dieser Prozess ist allgemein bekannt als spontane Emission. Durch spontane Emission

nimmt die Atomdichte der Atome im höheren Energiezustand ab. Mithilfe der Lebensdauer für spontane Emission τ und der Dichte der im höheren Energiezustand vorhandenen Atome N_2 kann diese Änderung beschrieben werden mit

$$\left|\frac{dN_2}{dt}\right|_{sp} = \frac{N_2}{\tau}. \tag{1.15}$$

1.2.5 Induzierte Emission

Der dritte wichtige Mechanismus für Laser ist die induzierte Emission [2]. Bei dieser Art der Emission erfolgt der Übergang der Atome vom höheren zum niedrigeren Energiezustand nicht spontan, sondern mithilfe einer Lichtwelle, welche die Bohrsche Frequenzbedingung erfüllt. Die Beziehungen dieses Prozesses sind analog zu denen der Absorption und ergeben sich zu

$$\left|\frac{dN_2}{dt}\right|_{i} = \sigma_{21} N_2 \Psi \tag{1.16}$$

und

$$\left|\frac{dI}{dx}\right|_{i} = \sigma_{21} N_2 I. \tag{1.17}$$

In diesen Beziehungen ist N_2 wie bei der spontanen Emission die Dichte der Atome im höheren Energiezustand und σ_{21} der Wirkungsquerschnitt der induzierten Emission. Im Gegensatz zur spontanen Emission, bei der ein Lichtquant in eine beliebige Richtung abgestrahlt wird, wird bei der induzierten Emission das Lichtquant in Ausbreitungsrichtung der Lichtwelle abgestrahlt. Die Phase und Frequenz dieser induzierten Welle ist identisch zur einfallenden Welle und verstärkt damit die einfallende Welle. Aus thermodynamischen und quantenmechanischen Überlegungen kann gezeigt werden, dass der Wirkungsquerschnitt der Absorption σ_{12} gleich groß ist wie der der induzierten Emission σ_{21}, sofern die Energieniveaus gleiche statistische Wahrscheinlichkeit haben.

$$\sigma_{12} = \sigma_{21} = \sigma \tag{1.18}$$

Gibt es in den Energieniveaus Unterzustände gilt die Beziehung

$$g_1 \sigma_{12} = g_2 \sigma_{21} = \sigma, \tag{1.19}$$

wobei g_1 und g_2 die Anzahl der Unterzustände des jeweiligen Energieniveaus beschreiben. Die Verstärkung der eingestrahlten Lichtwelle durch die induzierte Emission wirkt der Absorption entgegen. Dadurch ergibt sich die Gesamtänderung der Intensität der Lichtwelle zu

$$dI = -|dI|_a + |dI|_i \tag{1.20}$$

Für Energieniveaus die keine Unterzustände besitzen ($g_1 = g_2 = g = 1$) gilt daher

$$\frac{dN_2}{dt} = -\sigma N_1 I + \sigma N_2 I = -\sigma (N_1 - N_2) I \tag{1.21}$$

Integriert man diese Beziehung erhält man das verallgemeinerte Lambert-Beer'sche Gesetz

$$\frac{I}{I_0} = e^{\sigma (N_1 - N_2) d} = G. \tag{1.22}$$

Das Verhältnis von I/I_0 wird auch Verstärkungsfaktor G genannt. Mit einer He-Ne Gaslaserentladung einer Länge von 1 m lässt sich zum Beispiel im kontinuierlichen Betrieb ein Verstärkungsfaktor von $G = 1.1$ erreichen.

Damit eine Verstärkung auftreten kann, muss Besetzungsinversion ($N_2 > N_1$) der Energieniveaus gegeben sein. Bewerkstelligt wird das durch das sogenannte Pumpen. Pumpen beschreibt den Prozess die Atome des Lasermediums z.B He-Ne Gas auf ein höheres Energieniveau zu heben. Der Pumpmechanismus der verwendet wird hängt im allgemeinen vom Lasertyp ab. Die wichtigsten Laserobertypen sind Gaslaser, Festkörperlaser, Farbstofflaser und Halbleiterlaser.

1.3 Aufbau von Lasern

Ein Laser ist im Grunde ein Oszillator (Verstärker) für Signale mit optischen Frequenzen [3]. Der Frequenzbereich streckt sich dabei vom Infrarotbereich bis zum ultravioletten bzw. dem Bereich der Röntgenstrahlung. Der Grundaufbau eines Lasers beinhaltet dabei immer einen Verstärker (Oszillator). Dieser ist von der Funktionsweise her vergleichbar mit einem elektrischen Verstärker. Eine typische Verstärkerschaltung ist in Abb. 1.5 dargestellt. Die Eingangsspannung V_i wird mit der Rückkopplung βV_o summiert und anschließend über das Verstärkerbauteil mit Verstärkung A_0 verstärkt.

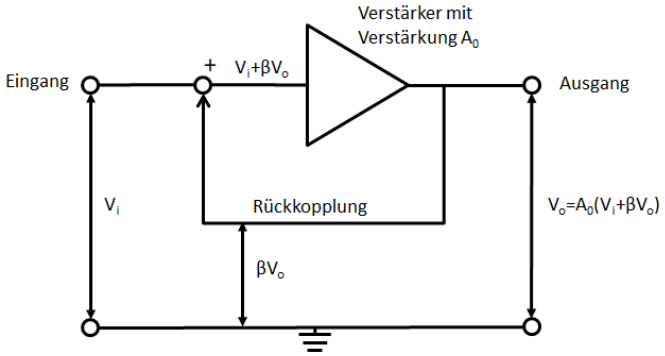


Abbildung 1.5: Schaltdiagramm eines Verstärkers mit positiver Rückkopplung [3]

Die Verstärkung A_0 ohne Rückkopplung ist dabei gegeben durch

$$A_0 = \frac{V_o}{V_i} \tag{1.23}$$

dem Verhältnis der Ausgangsspannung V_o zur Eingangsspannung V_i. Um die Gesamtverstärkung der Schaltung zu bestimmen muss die Rückkopplung berücksichtigt werden. Der Rückkopplungsfaktor β ist dabei im allgemeinen eine komplexe Zahl und kann als komplexer Vektor der Form

$$\beta = |\beta| e^{j\varphi} \tag{1.24}$$

dargestellt werden. Er besitzt einer Amplitude $|\beta| \leq 1$ und eine Phase φ. Die Ausgangsspannung V_o ergibt sich mit der Rückkopplung zu

$$V_o = A_o(V_i + \beta V_o) \tag{1.25}$$

und die Gesamtverstärkung A zu

$$A = \frac{A_o V_i}{1 + \beta A_0}. \tag{1.26}$$

Theoretisch ist damit eine unendlich große Verstärkung des Eingangssignals möglich wenn βA_0 genau $+1$ erreicht. Das würde bedeuten, dass ohne ein Eingangssignal ein finites Ausgangssignal

generiert wird. In der Praxis gibt es in jeder elektrischen Schaltung einen gewissen Rauschanteil, wodurch das Eingangssignal nie wirklich 0 sein kann. Der Faktor βA_0 ist frequenzabhängig und im allgemeinen nur bei einer bestimmten Frequenz genau +1. Bei dieser Frequenz schwingt die Schaltung indem das Rauschsignal verstärkt wird. Theoretisch sollte der Ausgang unendlich groß werden. Das passiert in der Realität allerdings nicht, denn umso größer das Signal wird, umso kleiner wird die Verstärkung A_0. Dieser Effekt ist bekannt als Sättigung und stellt ein fundamentales Phänomen für alle Verstärker dar. Der schematische Aufbau eines optischen Verstärkers ist in Abb. 1.6 abgebildet. Er besteht aus zwei gegenüberliegend angeordneten Spiegeln mit Abstand L und einem optischen Frequenzverstärker in der Mitte zwischen beiden Spiegeln.

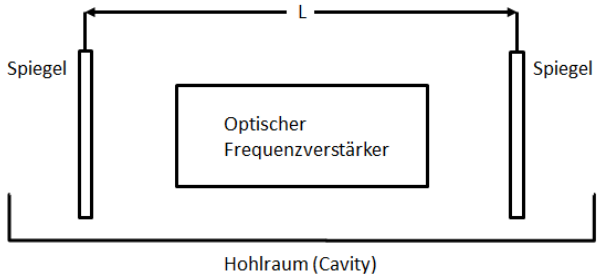


Abbildung 1.6: Schematische Darstellung eines optischen Verstärkers [3]

1.4 Laserparameter

Die Laserparameter [2] können in Laserkenndaten und Laserstrahlparameter unterteilt werden. Zu den Laserkenndaten gehören die Wellenlänge, die Frequenz, die Leistung, die Energie und der Betriebsart. Zu den Laserstrahlparametern gehören das Strahlprofil, die transversale Modenstruktur, die Strahldivergenz, die Fokussierbarkeit, die Kohärenz und die Polarisation.

1.4.1 Wellenlänge λ, Frequenz f

Die Wellenlänge λ beschreibt den kleinsten Abstand zweier Punkte gleicher Phase einer periodischen Welle. Sie kann mithilfe der Lichtgeschwindigkeit c und der Frequenz f der Welle mit

$$\lambda = \frac{c}{f} \tag{1.27}$$

bestimmt werden.

1.4.2 Leistung P, Pulsenergie E, Pulsdauer τ

und Abstand zweier Pulse T Die Leistung P [2] eines Lasers beschreibt die Ausgangsleistung eines Dauerstrichlasers in Watt. Für gepulste Laser wird die Ausgangsleistung charakterisiert über die Pulsenergie E in Joule, die Pulsdauer τ in Sekunden und der zeitliche Abstand zweier Pulse T in Sekunden. Die Pulsspitzenleistung P_{max} ist gegeben durch die Pulsenergie über der Pulsdauer τ mit

$$P_{max} = \frac{E}{\tau} \tag{1.28}$$

und die mittlere Leistung P über

$$P = \frac{E}{T} \tag{1.29}$$

1.4.3 Polarisation

Die Polarisation [2] definiert die Orientierung des elektrischen Feldes einer Lichtwelle. Laserlicht ist meistens linear polarisiert. Linear polarisiert bedeutet, dass der Vektor der elektrischen Feldstärke immer in einer definierten Ebene schwingt.

1.4.4 Divergenz θ

Die Divergenz [4] beschreibt die Verbreiterung eines Strahls mit zunehmender Entfernung im Bezug auf die Strahltaille des Lasers durch Beugung. Sie wird definiert über den Halbwinkel des Lasers und ist bei Gaußstrahlen definiert als

$$\theta = \frac{\lambda}{\pi w_0}, \tag{1.30}$$

wobei w_0 der Strahltaillenradius und λ die Wellenlänge des Laserstrahl beschreibt.

1.4.5 Beugungsmaßzahl M^2

Das Strahlprofil eines Laserstrahls beschreibt die Intensitätsverteilung des Laserstrahlquerschnitts. Die gebräuchlichsten Strahlprofile haben entweder eine gaußsche oder eine Flat-Top Verteilung. Ein gaußsches Profil besitzt die geringste Divergenz und ist daher für viele Anwendungen ideal. Die Beugungsmaßzahl M^2 [4] beschreibt die Abweichung des Divergenzwinkels eines realen Strahls von einem idealen Gaußstrahl und ist somit definiert als

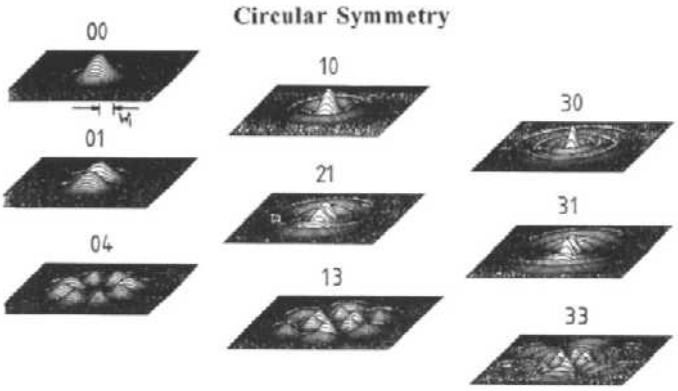
$$M^2 = \frac{\varphi_{real}}{\varphi_{gauss}} \tag{1.31}$$

1.4.6 Longitudinal und Transversalelektromagnetische (TEM) Modenstruktur

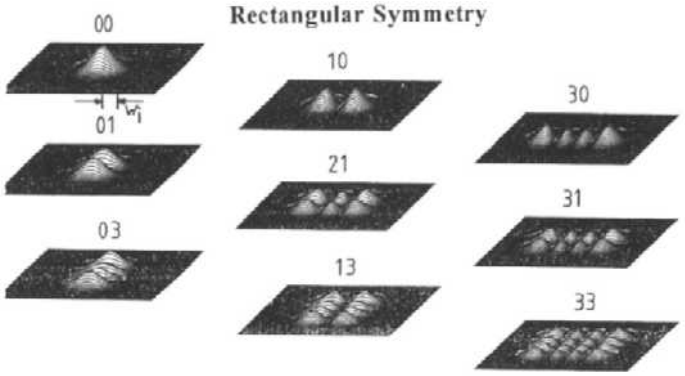
Jede Welle [5] besteht aus unterschiedlichen Schwingungsformen (Moden). Die Moden in Ausbreitungsrichtung der Strahlung werden Longitudinal Moden und senkrecht zur Ausbreitungsrichtung als Transversal Moden bezeichnet. In longitudinaler Richtung können grundsätzlich alle Wellenlängen, die in den Resonator passen und einen Schwingungsknoten auf den Resonatorspiegeln aufweisen, anschwingen. Besteht es nur aus rein einer optischen Frequenz spricht man von monochromatischen Licht ansonsten von polychromatischem Licht. Welche Moden in transversaler Richtung schwingen wird vom Aufbau des Lasers beziehungsweise vom Resonator bestimmt. Der TEM Grundmode entspricht einem idealen Gaußprofil und wird auch als TEM_{00} bezeichnet. Bei den TEM höherer Ordnung kann zwischen radialsymmetrischen Moden (siehe Abb. 1.7a) und rechteckig symmetrischen Moden (siehe Abb. 1.7b) unterschieden werden. Welcher Mode sich ausbildet kann unter anderen durch Blenden beeinflusst werden. Ist im Resonator keine Blende verbaut, können viele Moden gleichzeitig schwingen. Wird jedoch eine Blende verbaut, ist es möglich die Anzahl der schwingenden Moden zu reduzieren, indem der Durchmesser der Blende verkleinert wird. Möchte man das Strahlprofil nur auf den TEM_{00} Grundmode beschränken, muss der Durchmesser der Blende nahe dem Durchmesser des TEM_{00} Profils gewählt werden. Dadurch werden für alle anderen Moden hohe Verluste an der Blende generiert, wodurch diese nicht schwingen können. Erhöht man den Durchmesser der Blende können auch Moden höherer Ordnung die "durch die Blende passen" schwingen. Niedrige Moden können auch über die Spiegelform und der Resonatorgeometrie, d.h. dem Verhältnis Spiegeldurchmesser und Resonatorlänge selektiert werden.

Dasselbe Prinzip gilt für die Erzeugung anderer Moden. Möchte man einen speziellen Mode generieren müssen für die anderen Moden Verluste verursacht werden. Eine Möglichkeit das zu Bewerkstelligen ist es zwei gekreuzte Drähte im Resonator zu platzieren (siehe Abb. 1.8). Die Drähte verursachen entlang ihrer Länge Verluste. Dadurch werden Moden bevorzugt die Symmetrien um

Abbildung 1.7: TEM Moden mit unterschiedlicher Symmetrie



(a) Berechnete TEM Moden mit radialer Symmetrie[5]



(b) Berechnete TEM Moden mit rechteckiger Symmetrie [5]

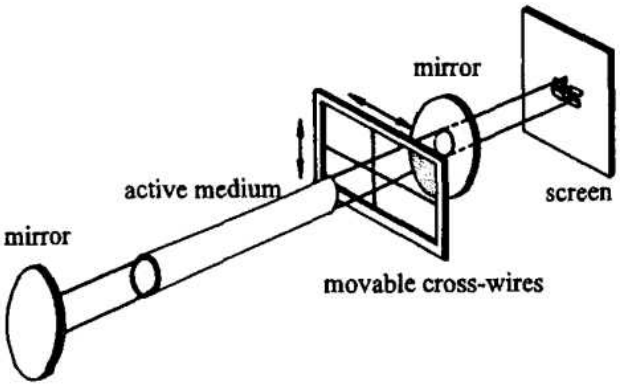


Abbildung 1.8: Aufbau zur Beobachtung von individuellen TEM Moden [5]

die Achsen der Drähte besitzen, da diese weniger beeinflusst werden. Wendet man das Prinzip auf einen He-Ne Laser an, können rechteckig symmetrische Moden unterschiedlicher Ordnung erzeugt werden. Abb. 1.9 zeigt TEM Moden die mithilfe so eines Aufbaus erzeugt werden können. Die Drähte wurden an die Stellen verschoben an der der jeweilige Mode seine Symmetrie besitzt und wird dadurch bevorzugt. Der Winkel der Drähte beeinflusst dabei nicht welcher Mode generell schwingt, sondern nur um welche Achse sich der Mode symmetrisch ausbildet.

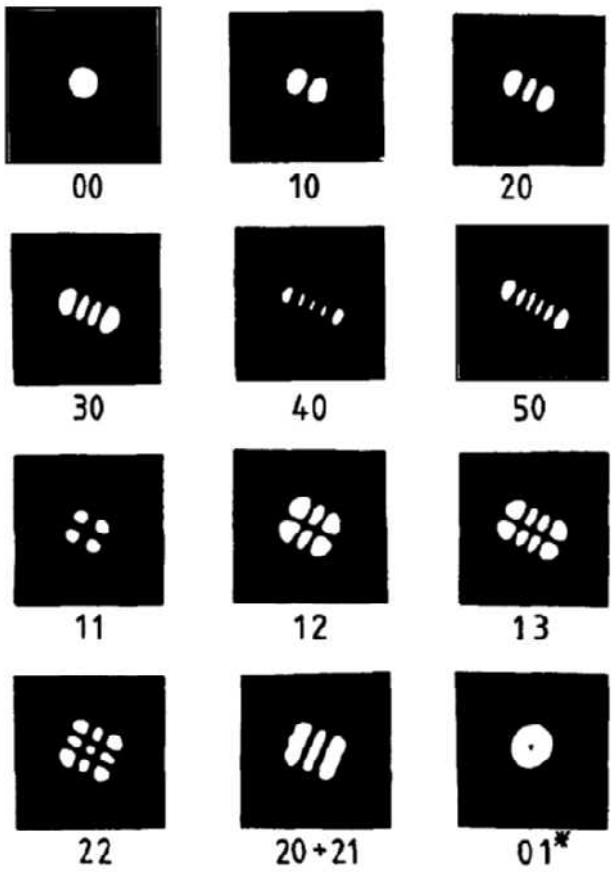


Abbildung 1.9: Aufgenommene Strahlprofile für verschiedene TEM Moden eines He-Ne Lasers [5]

1.5 Laserstrahlcharakterisierung

Laserstrahlcharakterisierung ist eine Methode um die Strahleigenschaften von Lasern zu ermitteln. Die wichtigsten Parameter sind dabei die Beugungsmaßzahl M^2, der Ort der Strahltaille d_{z_0}, der Strahldurchmesser d_σ und der Divergenzwinkel Θ . Kennt man diese Parameter, ist es möglich den Laserstrahl für jede Anwendungen optimal zu nutzen, bzw. während dem Justieren des Lasers die Einflüsse des Justierens selbst auf den resultierenden Laserstrahl zu beobachten. Besonders wichtig ist die Kenntnis dieser Parameter bei Lasern mit komplexem, internen Aufbau, da bei solchen Lasern regelmäßige Justagearbeiten notwendig sind, um die maximale Arbeitsleistung zu erreichen.

1.6 Ti:Saphir Femtosekundenlaser

Es ist nicht Inhalt dieser Arbeit auf alle Einzelheiten, der physikalischen Effekte für die Erzeugung von Femtosekundenpulsen einzugehen [6]. Ein Überblick über den Aufbau und die Funktionsweise ist aber insofern wichtig, um die Notwendigkeit einer regelmäßigen Überprüfung des Strahlprofils zu erklären. Das FEMTOPOWER COMPACT PRO Lasersystem mit dem Pulse einer Pulsdauer von $< 30fs$ erzeugt werden können besteht aus zwei separaten Lasern (siehe Abb. 1.10): Einem Ti:Saphir Oszillator-laser gepumpt von einem CW Diodenlaser, einer Verstärkerstufe ebenfalls mit einem Ti:Saphir Kristall und einem Nd:YLF Pumplaser. Der Oszillatorlaser liefert Pulse mit einer Wiederholfrequenz von $74-78MHz$, einer Pulsdauer von $< 12fs$ und einer Pulsenergie von $> 2\ nJ$ (siehe Tab. 1.1). Diese spektral breitbandigen Pulse werden über den Effekt der sogenannten Group-

Tabelle 1.1: Datenblatt FEMTOPOWER COMPACT PRO

Output	amplifier	oscillator
Pulse duration (FHWM)	$< 30\ fs$	$< 12\ fs$
Spectral width (FHWM) @800nm	$> 40\ nm$	$> 100\ nm$
Outtput energy	$> 800\ \mu J$	$> 2\ nJ$
Pulse repetition rate	$1\ kHz$	74 - $78\ MHz$
Peak power	$> 30\ GW$	$> 500\ kW$
Beam diameter $(1/e^2)$	$15\ mm$ (nominal)	$>2\ mm$
B-integral of amplified pulses	< 1	-
Spatial mode	$TEM_{00}(M^2 < 2)$	$TEM_{00}(M^2 < 1.3)$
Contrast ratio	$> 10^7 : 1 / > 10^4 : 1$	-
Polarizations	linear, horizontal	linear, horizontal
Pulse-to-pulse energy stability	$< 1.5\ \%\ rms$	-
Beam divergence	$< 3\ mrad$	$< 2\ mrad$

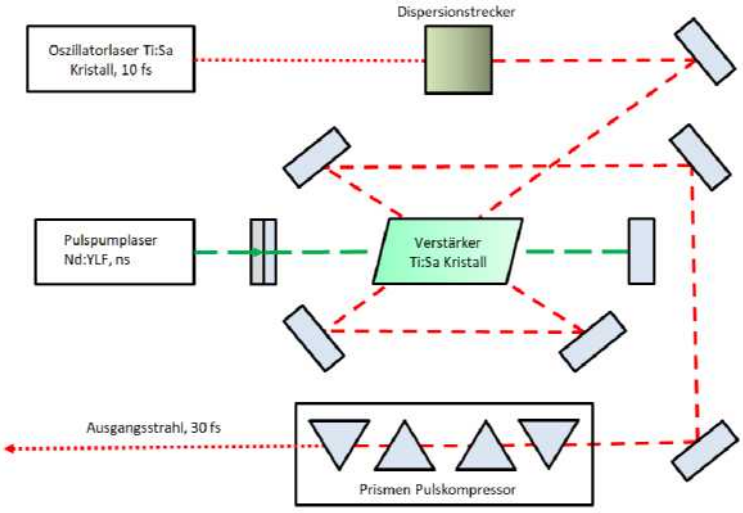


Abbildung 1.10: Vereinfachte Schemaskizze eines Ti:Saphir Femtosekunden Lasers

Delay-Dispersion (GDD) mit einem positiven chirp zeitlich um einen Faktor von ca. 10^5 aufgeweitet um eine weitere Verstärkung zu ermöglichen. Würden die Oszillatorpulse nicht aufgeweitet werden, würde aufgrund der Selbstfokussierung im Ti:Saphir Kristall, die Intensität der Strahlung den Schadengrenzwert des Kristalls überschreiten und ihn beschädigen. Um eine höhere Verstärkung zu erreichen wird der Oszillator Laserstrahl mehrmals durch den Verstärkerkristall geführt. Durch das mehrmalige "durchfädeln" des Oszillatorlaserstrahls durch den Verstärker Ti:Saphir Kristall wird die Energie der Laserpulse immer weiter erhöht. Nach der Verstärkung wird der aufgeweitete Puls über einen Prismenkompressor bestehend aus 4 Prismen komprimiert um den zuvor eingeführten

positiven chirp wieder auszugleichen. Die einzelnen Prismen des Prismenkompressors sind verschieb- bzw. verdrehbar und das Einstellen dieser Prismen ist kein simples Unterfangen. Ändert sich die mittlere Wellenlänge des Laserpulses [7], müssen alle 4 Prismen um denselben Winkel verdreht werden um diese Veränderung auszugleichen. Zusätzlich muss die GDD durch verschieben der Prismen eingestellt werden, damit am Ende ein möglichst kurzer Puls zur Verfügung steht. Der Abstand zwischen dem ersten und zweiten sowie dem dritten und vierten Prisma ermöglicht eine Grobkorrektur der GDD. Die Feinkorrektur der GDD kann durch verschieben des zweiten sowie dritten Prismas weiter in oder aus dem Laserstrahl eingestellt werden. Der resultierende gepulste Laserstrahl hat eine Pulsfrequenz von 1 kHz, mit einer Pulsdauer von $< 30\ fs$ und einer Pulsenergie von $< 800\ \mu J$.

2 Bestimmung der Strahleigenschaften

Die Bestimmung der Strahleigenschaften von Lasern ist mit der DIN EN ISO 11146 [8] genormt. Aufgeteilt ist die Norm in drei Teile. Der erste Teil, die DIN EN ISO 11146-1 beschreibt die Prüfverfahren und Bestimmung der Strahlparameter für stigmatisch und leicht astigmatische Strahlen, die DIN EN ISO 11146-2 die Prüfverfahren und Bestimmung der Strahlparameter für astigmatische Strahlen und die ISO/TR 11146-3 [9] dient als unterstützende Norm, in der unter anderem Hintergrundkorrekturverfahren beschrieben sind. Die Normen dienen als Grundlage für die in diesem Kapitel beschriebenen Auswertung.

2.1 Hintergrundkorrekturverfahren

Die mit einem pixelbasierten Sensor aufgenommene Leistungsdichteverteilung $E_{meas}(x, y)$ kann in zwei Teile aufgeteilt werden. In die "wahre"Leistungsdichteverteilung $E(x, y)$ und in eine Hintergrundleistungsdichteverteilung E_H die von anderen Lichtquellen (z.B. Raumbeleuchtung) verursacht wird:

$$E_{meas}(x, y) = E(x, y) + E_H(x, y) \tag{2.1}$$

Diese Hintergrundleistungsdichteverteilung $E_H(x, y)$ kann weiter aufgeteilt werden in einen homogenen Anteil $E_{H,offset}(x, y)$ der den für die gesamte Verteilung konstanten Versatz beschreibt, einen inhomogenen Anteil $E_{H,inh}(x, y)$ der die Neigung des Versatzes beschreibt und einem hochfrequenten Rauschanteil $E_{H,noise}(x, y)$.

$$E_H(x, y) = E_{H,offset}(x, y) + E_{H,inh}(x, y) + E_{H,noise}(x, y). \tag{2.2}$$

Der hochfrequente Rauschanteil $E_{H,noise}(x, y)$ kann und muss normalerweise nicht korrigiert werden, da Rauschkorrekturen das Bild glätten. Glätten bedeutet, dass jeder einzelnen Pixel mithilfe der umgebenden Pixeln einen neuen Wert zugewiesen bekommt. Dadurch kann Rauschen zwar entfernt werden, aber die Intensitätsverteilung wird ebenfalls als gesamtes beeinflusst. Vor allem die Ränder der Intensitätsverteilung würden auf diese Weise nach außen verschoben werden, wodurch die Intensitätsverteilung künstlich vergrößert wird. Dafür reichen im Randbereich bereits kleine Änderungen aus. Der homogene und inhomogene Anteil der durch zusätzliche Lichtquellen verursacht wird kann allerdings nicht ignoriert werden, da Strahldurchmesser und Strahlmittelpunkt aus Integralen der Leistungsdichteverteilung bestimmt werden. Kompensiert man diese Anteile nicht, führt es dazu, dass die Strahldurchmesser größer berechnet werden als sie in Wirklichkeit sind. Es gibt mehrere Methoden die Hintergrundkorrektur durchzuführen. Im Zuge dieser Arbeit wurden 4 in der Norm beschriebene Methoden miteinander verglichen. Die 4 Verfahren lassen sich in Grobkorrektur und Feinkorrektur unterteilen. Zu den Grobkorrekturverfahren gehören "Coarse correction by background map subtraction"(CCBMS) bzw. "Coarse correction by average background subtraction"(CCBAM) und zu den Feinkorrekturverfahren "Fine correction by statistical method"(FCBSM) bzw. "Fine correction by approximation method"(FCBAM).

2.2 Grobkorrektur (Coarse correction)

Um eine Grobkorrektur des Hintergrunds durchzuführen sind Aufnahmen der Intensitätsverteilung aufgrund von externen Lichtquellen notwendig. Diese Aufnahmen werden auch ”schwarze Bilder” genannt. Beide Grobkorrekturverfahren (CCBMS & CCBAM) bestimmen aus diesen ”schwarze

Bildern" den notwendigen Korrekturwert der auf jedes Pixel angewendet wird. Bestimmt wird dieser Korrekturwert grundsätzlich bei beiden Verfahren durch Mittlung von Pixelwerten. Aus diesem Grund ist ein Einzelbild des Hintergrunds nicht ausreichend. Es sollten laut Norm zumindest $n > 10$ Bilder an jedem Messpunkt aufgenommen werden. Die Messung des Hintergrunds sollte möglichst zeitnah vor oder nach der Lasermessung durchgeführt werden, damit Änderungen des Hintergrundverhältnisse auf ein Minimum zu reduzieren. Sind die Hintergrundverhältnisse konstant ist es möglich erst den Hintergrund zu messen und danach den Laserstrahl.

2.2.1 Coarse correction by background map subtraction (CCBMS)

Bei CCBMS wird jeder einzelne Pixel der "schwarzen Bilder" gemittelt und daraus eine sogenannte Hintergrundkorrekturkarte(background map) erstellt. Diese Hintergrundkarte ist im Grunde nichts anderes als ein Bild bestehend aus den gemittelten Pixel der "schwarzen Bilder". Bestimmt werden diese gemittelten Pixel mit

$$\overline{E_H(x,y)} = \frac{1}{n}\sum_{i=1}^{n} E_H(x,y)_i. \tag{2.3}$$

Mit dieser Hintergrundkarte bestehend aus den gemittelten Pixeln kann anschließend die korrigierte Leistungsdichteverteilung mit

$$E(x,y) = E_{meas}(x,y) - \overline{E_H(x,y)} \tag{2.4}$$

berechnet werden. Da jeder Pixel einzeln gemittelt wird ist es mit dieser Grobkorrektur möglich Einflüsse die nur Teile des Sensors betreffen zu kompensieren. Ein Beispiel dafür ist eine Strahlungsquelle die den Sensor nur in der unteren Hälfte bestrahlt.

2.2.2 Coarse correction by average background subtraction (CCBAM)

Ist der Einfluss der Umgebung über den gesamten Sensor konstant, kann CCBAM verwendet werden. Im Gegensatz zur CCBMS werden bei diesem Verfahren alle Pixel über alle "schwarzen Bilder" gemittelt und daraus ein konstanter Versatzwert mit

$$\overline{E_{H,Offset}(x,y)} = \frac{1}{n \cdot m}\sum_{i=1}^{n}\sum_{x,y=1}^{m} E_H(x,y)_i \tag{2.5}$$

bestimmt. Dabei ist m die Gesamtanzahl an (x,y) Datenpunkten auf dem Sensor und n die Anzahl der Einzelmessungen. Mit diesem Versatzwert, wird die korrigierte Leistungsdichteverteilung mit

$$E(x,y) = E_{meas}(x,y) - \overline{E_{H,Offset}(x,y)} \tag{2.6}$$

bestimmt.

2.3 Feinkorrektur (Fine correction)

Feinkorrektur stellt die zweite Stufe der Hintergrundkorrektur dar. Sie ist erforderlich wenn die Grobkorrektur alleine nicht ausreicht Hintergrundeinflüsse zu kompensieren.

2.3.1 Fine correction by statistical method (FCBSM)

Bei diesem Verfahren wird die Hintergrundverteilung $E_{H,Offset}(x,y)$ durch einen durchschnittlichen Versatz $E_{H,Offset}$ und einer Standardabweichung $E_{H,\sigma}$ beschrieben. Im ersten Schritt ist es bei diesem Verfahren notwendig den Versatz und die Standardabweichung zu schätzen. Eine Methode

um diese Schätzwerte zu generieren ist es sie aus "schwarzen Bildern" zu bestimmen. Für die n "schwarzen Bilder" kann mit Gl. 2.3 $E_{H,Offset}$ abgeschätzt werden. Die Standardabweichung kann aus den n Bildern mit

$$E_{H,\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(E_{H,Offset,i} - E_{H,Offset})^2} \tag{2.7}$$

bestimmt werden. Hat nun ein Pixel einen höheren Graustufenwert als

$$E_{i,j} > E_{H,Offset} + n_T E_{B\sigma} \tag{2.8}$$

wird dieser als beleuchtet angesehen und für die Strahldurchmesser und Strahlmittelpunktbestimmung herangezogen. Der Faktor n_T sollte dabei im Bereich $2 < n_T > 4$ liegen. Er wird zum Feinjustieren des Kriteriums für beleuchtete Pixel 2.8 verwendet. Jeder Pixel der Gl. 2.8 nicht erfüllt wird zur Bestimmung des Versatzes herangezogen und ergibt sich aus dem Mittelwert der "dunklen"Pixel. Angewendet wird dieses Verfahren allerdings nicht direkt auf das gemessene Bild sondern auf eine verschmierte Version davon. Eine einfache Methode ein Bild zu verschmieren ist die 2D-Faltung. Bei der 2D-Faltung wird ein zentraler Pixel mit seinen umgebenden Pixel summiert, wobei jeder Pixel eine eigene Gewichtung besitzt. Die Gewichtung jedes einzelnen Pixel liegt zwischen 0 - 1 und hängt davon ab welcher Effekt mit der Faltung erreicht werden soll. Festgelegt wird diese mit der sogenannten Kernelmatrix. Das sind $n \times m$ Matrizen in der die Gewichtungen jedes einzelnen Pixels festgelegt wird. Sie haben die Form

$$K = \begin{pmatrix} a_{1,1} & a_{1,1} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,m}. \end{pmatrix}$$

Um nun ein Bild zu verschmieren benötigt man eine Kernelmatrix in der jeder Pixel dieselbe Gewichtung besitzt. Die Koeffizienten ergeben sich dann zu

$$a_{n,m} = a = \frac{1}{n \cdot m} \tag{2.9}$$

und die Kernelmatrix zu

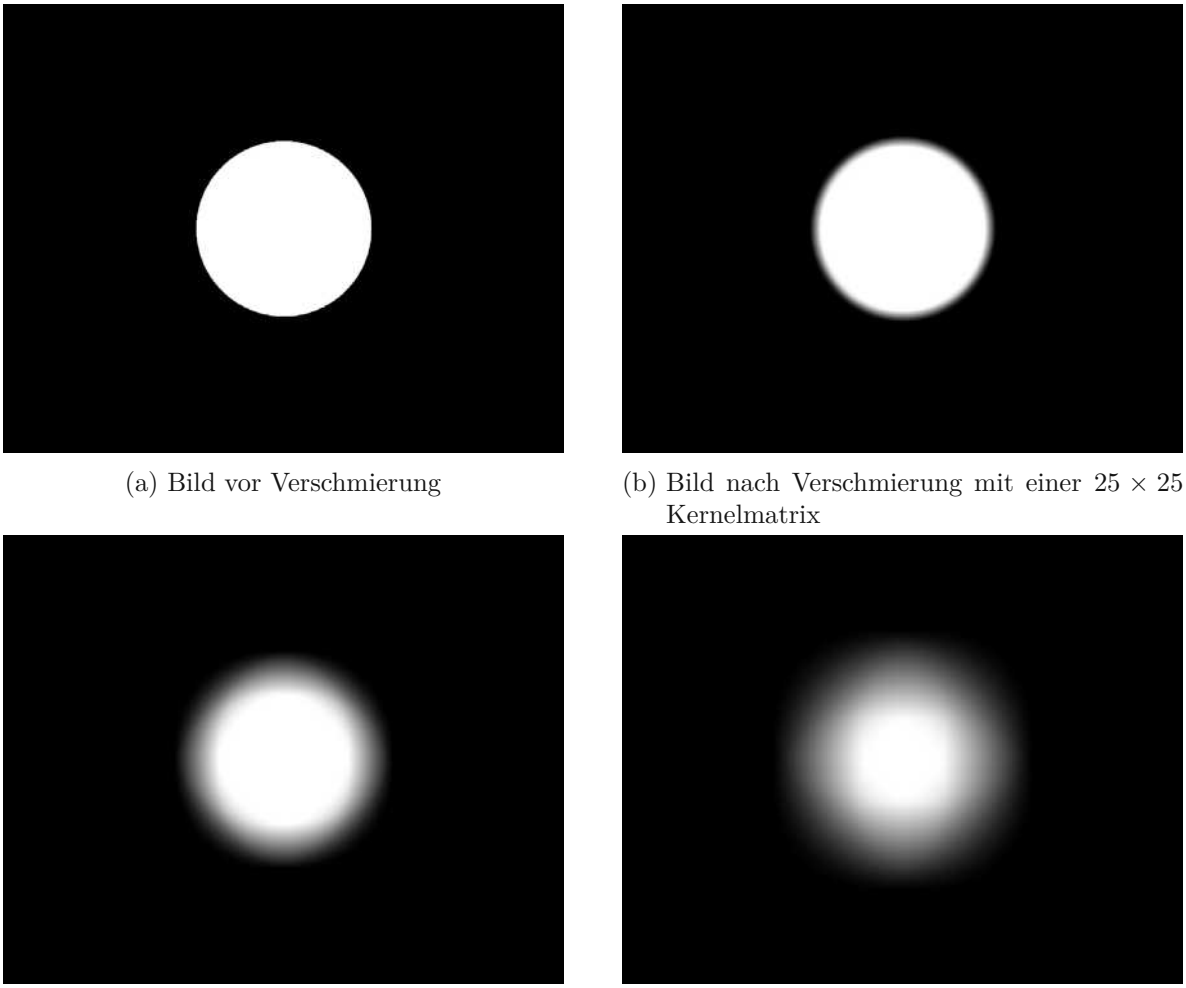
$$K = \frac{1}{n \cdot m}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1. \end{pmatrix} \tag{2.10}$$

Diese Form der Kernelmatrix wird auch als Verwisch-filter bezeichnet. Jeder einzelne Pixelwert des verschmierten Bildes lässt sich mathematisch beschreiben mit

$$\tilde{E_{i,j}} = \frac{1}{(n+1)\cdot(m+1)} \sum_{k=i-\frac{n}{2}}^{k=i+\frac{n}{2}} \sum_{l=j-\frac{m}{2}}^{l=j+\frac{m}{2}} E_{k,l}. \tag{2.11}$$

Wie stark verschmiert wird hängt von der Größe der Kernelmatrix ab. Verwendet man eine 25×25 Kernelmatrix wird aus dem weißen Kreis in Abb. 2.1a ein leicht verwischter. Durch Erhöhung der Anzahl der Elemente die gemittelt werden, kann das Bild weiter verwischt werden (siehe Abb. 2.1c). Um den Einfluss der Kernelmatrix auf das verschmierte Bild zu zeigen wurde das Bild mit einer 25×25 Kernelmatrix, einer 100×100 Kernelmatrix und einer 200×200 Kernelmatrix verschmiert. Der Durchmesser des Kreises wird wächst mit der Größe der Kernelmatrix, wobei die Intensität jedes

Abbildung 2.1: Einfluss der Größe einer Kernelmatrix auf die Bildverschmierung



(a) Bild vor Verschmierung

(b) Bild nach Verschmierung mit einer 25×25 Kernelmatrix

(c) Bild nach Verschmierung mit einer 100×100 Kernelmatrix

(d) Bild nach Verschmierung mit einer 200×200 Kernelmatrix

Pixel gleichzeitig abnimmt. Die Größe der gewählten Kernelmatrix ist daher ein wichtiger Faktor. In der Norm ist als Richtwert für die Größe der Kernelmatrix $2\,\% - 5\,\%$ der Sensorabmessungen angegeben. Auf das verschmierte Bild kann nun zur Bestimmung der unbeleuchteten Pixel die Bedingung

$$\tilde{E_{i,j}} < E_{H,Offset} + n_T \frac{E_{B\sigma}}{\sqrt{(n+1)(m+1)}} \tag{2.12}$$

herangezogen werden. Den tatsächlichen Versatz erhalten wir indem alle Pixel die die Bedingung Gl.2.12 erfüllen(die unbeleuchteten Pixel) mitteln.

$$\overline{E_{H,Offset,tats}} = \frac{\sum E_{i,j}}{\text{Anzahl der Pixel}} \tag{2.13}$$

Das korrigierte Bild wird anschließend mit

$$E(x,y) = E_{meas}(x,y) - \overline{E_{H,Offset,tats}(x,y)} \tag{2.14}$$

bestimmt.

2.3.2 Fine correction by approximation method (FCBAM)

Bei der FCBAM wird der Versatz direkt aus dem gemessenen Bild ermittelt. Eine Voraussetzung für Anwendung dieses Verfahren ist, dass der Durchmesser des Laserstrahls nicht größer ist als das 0.5 fache der Sensordimensionen. Das liegt daran, dass es nicht beleuchtete Bereiche auf dem Sensor geben muss aus denen der Versatz berechnet werden kann. Die Ecken des Sensors eignen sich dafür besonders gut, da diese Bereiche im Normalfall nicht beleuchtet sind. Für die Bestimmung des Versatzes werden N nicht beleuchtete Pixel in einem Bereich $n \times m$ der 4 Ecken gemittelt. Die Größe des Bereichs kann mit $2\,\% - 5\,\%$ der Sensordimensionen abgeschätzt werden. Die Voraussetzung der Laserstrahlgröße allein reicht aber noch nicht aus um sicher zu stellen,dass diese Methode zulässig ist. Es muss zusätzlich die Bedingung

$$\frac{|E_{H,Offset,meas} - E_{H,Offset}|\sqrt{N}}{E_{H\sigma}} < n_T, \tag{2.15}$$

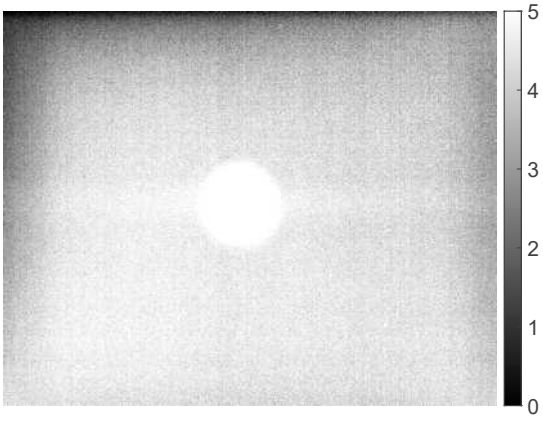
erfüllt werden. Dabei ist $E_{H,Offset,meas}$ der Versatz bestimmt aus den Ecken, $E_{H,Offset}$ ist der Versatz bestimmt aus den "schwarzen Bilder" mit Gl. 2.3 und n_T wie bei FCBSM zwischen $2 < n_T > 4$ liegen. Die Funktion ist dieselbe wie in FCBSM.

2.4 Anwendung der Hintergrundkorrekturverfahren

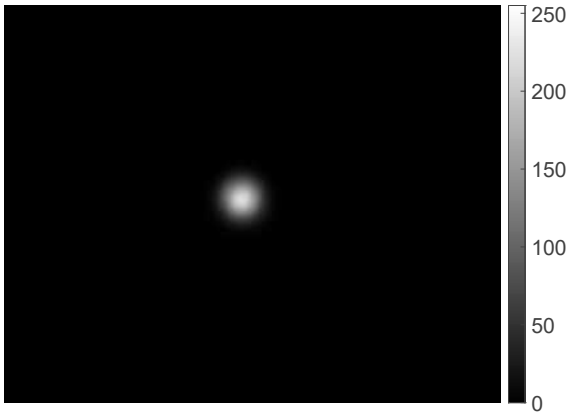
Der Einfluss der Hintergrundkorrekturverfahren auf das gemessene Strahlprofil ist in den folgenden Abbildungen dargestellt. Aufgenommen wurde das Strahlprofil eines He-Ne Lasers. Verwendet wurde dafür eine monochromatische Kamera mit einem Wertebereich von $0-255$ in einem verdunkelten Raum. Der Raum konnte allerdings nicht vollständig verdunkelt werden. Die Verwendung des gesamten Wertebereichs für die Skalierung der Pixel führte zu Bildern die auf ersten Blick völlig hintergrundfrei waren. Um die mitgemessenen Hintergrundeinflüsse sichtbar zu machen, war es notwendig den Wertbereich der Graustufenskalierung von $0-255$ auf $0-5$ zu verringern. Mit dieser neuen Skalierung hoben sich Pixel mit Graustufen > 0 von den schwarzen Pixeln deutlich ab. Abb. 2.2a zeigt ein Messergebnis mit einer Graustufenskalierung von $0-255$ was dem gesamten möglichen Wertebereich entspricht und Abb. 2.2b dasselbe Messergebnis allerdings mit einer reduzierten Graustufenskalierung von $0-5$. Der Unterschied zwischen den beiden Skalierungen war klar zu erkennen. In Abb. 2.2a war es eine scheinbar hintergrundfreie Aufnahme eines Laserstrahls mit TEM_{00} Profil. Mit dem verkleinerten Wertebereich in Abb. 2.2b wurde der Hintergrund sichtbar und das Strahlprofil war nur mehr schwer zu erkennen. Abb. 2.2c und Abb. 2.2d zeigen dasselbe Profil, nachdem CCBMS auf das Bild angewendet wurde. Eine Gegenüberstellung der beiden verschiedenen Graustufenskalierungen lieferte in der $0-255$ Graustufenskalierung keinen sichtbaren Unterschied zum unkorrigierten Profil in Abb. 2.2a. In Abb. 2.2d wird der Effekt den die Grobkorrektur auf das unkorrigierte Bild hatte sehr klar, da nun das Strahlprofil eindeutig erkennbar war. Betrachtet man Abb. 2.2d im Detail fällt auf, dass die Grobkorrektur nicht alles kompensieren konnte. Aus diesem Grund wurde nun noch eine Feinkorrektur nach der Grobkorrektur durchgeführt. Angewendet wurde die Feinkorrektur anhand einer statistischen Methode (FCBSM). Dargestellt ist das Ergebnis in Abb. 2.2e und Abb. 2.2f wieder mit beiden Graustufenskalierungen. Auch im fein korrigierten Bild mit der Farbskalierung $0-255$ ist erwartungsgemäß kein Unterschied zum unkorrigierten Bild Abb.2.2a sichtbar. Mit der reduzierten Graustufenskalierung ist eine klare Verbesserung zur Grobkorrektur in Abb. 2.2d erkennbar. Im Idealfall erhält man nach Grob- und Feinkorrektur ein völlig hintergrundfreies Bild auf dem nur das Laserstrahlprofil zu sehen ist. Realistisch bleibt aber ein gewisses Restrauschen übrig, das aufgrund seiner Zufälligkeit nicht gut korrigiert werden kann. Es gibt wie bereits in Kap. 2.1 erwähnt verschiedene Möglichkeiten diese Korrektur durchzuführen. CCBAM als Grobkorrektur kompensierte einen Großteil des Hintergrundes, allerdings nicht so gut wie CCBMS. Aus den Ergebnissen geht hervor, dass die Grobkorrektur mit CCBMS das Rauschen

Abbildung 2.2: Einfluss der Graustufenskalierung und der Korrekturverfahren

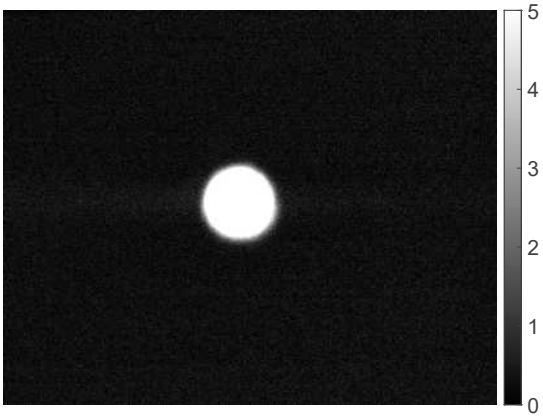
(a) Keine Korrektur Graustufenskalierung $0 - 255$



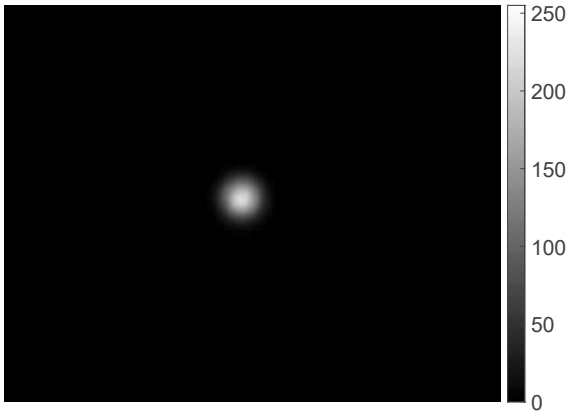
(b) Keine Korrektur Graustufenskalierung $0 - 5$



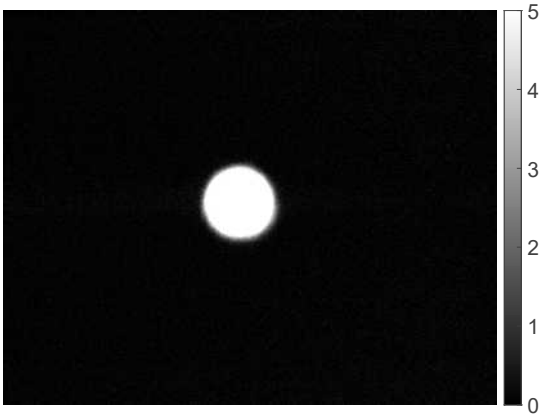
(c) Nach Background Map Substraction Graustufenskalierung $0 - 255$



(d) Nach Background Map Substraction Graustufenskalierung $0 - 5$

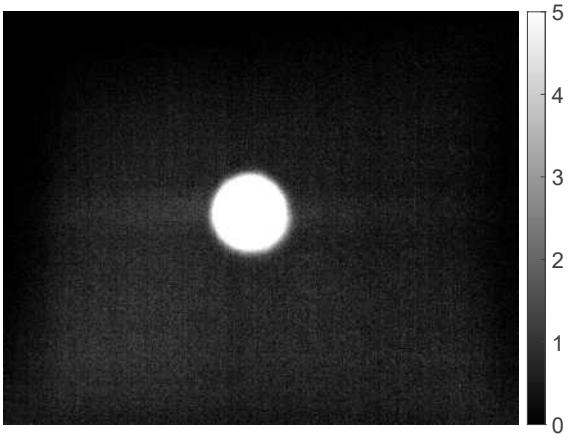


(e) Nach Statistical Feinkorrektur Graustufenskalierung $0 - 255$



(f) Nach Statistical Feinkorrektur Graustufenskalierung $0 - 5$

besser filtert als CCBAM. Der Grund dafür ist in Abb.2.2g zu sehen. Das Bild zeigt eine inhomogene Verteilung des Rauschens. Ein häufiger Grund dafür ist, dass die Hintergrundstrahlung mit einem Eintrittswinkel (Angle of incidence AOI) $\neq 0$ auf den Kamerasensor trifft wodurch der Sensor inhomogen beleuchtet wird. CCBAM eignet sich wie bereits beschrieben nur für die Kompensation



(g) Nach Average Value Substraction Graustufenskalierung $0-5$

(h) Nach Statistical Feinkorrektur Graustufenskalierung $0-5$

von jenen Einflüssen, die den gesamten Kamerasensor betreffen.

2.5 Bestimmung des Strahldurchmessers aus einer gegebenen Leistungsdichteverteilung

2.5.1 Stigmatische und leicht astigmatische Strahlen

Als Grundlage für die Bestimmung der Laserstrahleigenschaften von stigmatischen bzw. leicht astigmatischen Strahlen dient die DIN EN ISO 11146-1. In ihr sind die Berechnungsverfahren beschrieben um aus einer gegebenen Leistungsdichteverteilung eines Laserstrahls die Laserstrahleigenschaften zu bestimmen. Verwendet wird dafür die Momente erster- bzw- zweiter- Ordnung.

Bestimmung der Momente erster Ordnung einer Leistungsdichteverteilung $\overline{x}$,$\overline{y}$

Die Momente erster Ordnung einer Leistungsdichteverteilung beschreiben die Position des Strahlzentrums. In den allgemeinen Gleichungen gehen die Grenzen von $-\infty$ bis ∞, wir ersetzen diese Grenzen mit unseren finiten Integrationsbereich beschränkt durch die Größe des Kamerasensors. Die Momente erster Ordnung können bestimmt werden mit

$$\overline{x}(z) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z)x \; dx \; dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z) \; dx \; dy} \tag{2.16}$$

und

$$\overline{y}(z) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z)y \; dx \; dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z) \; dx \; dy} \tag{2.17}$$

$\overline{x}$ entspricht dem Abstand des Strahlzentrums von der x-Achse und $\overline{y}$ dem Abstand des Strahlzentrums von der y-Achse des gewählten Integrationsbereichs. $E(x,y,z)$ ist dabei der Intensitätswert eines Pixels und x bzw. y der Abstand des Pixels zum Koordinatensystem des gewählten Integrationsbereichs.

Bestimmung der Momente zweiter Ordnung einer Leistungsdichteverteilung σ_x^2, σ_y^2, σ_{xx}^2

Die Momente zweiter Ordnung einer Leistungsdichteverteilung beschreiben die geometrische Ausdehnung eines Strahls und sind gegeben durch

$$\sigma_x^2(z) = \left\langle x^2 \right\rangle = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z)(x-\overline{x})^2 \; dx \; dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z) \; dx \; dy} \tag{2.18}$$

für das Moment zweiter Ordnung in x,

$$\sigma_y^2(z) = \left\langle y^2 \right\rangle = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z)(y-\overline{y})^2 \; dx \; dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z) \; dx \; dy} \tag{2.19}$$

für das Moment zweiter Ordnung in y und

$$\sigma_{xy}^2(z) = \left\langle xy \right\rangle = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z)(x-\overline{x})(y-\overline{y}) \; dx \; dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x,y,z) \; dx \; dy} \tag{2.20}$$

für das Mischmoment zweiter Ordnung.

Bestimmung des Azimutwinkels φ

Der Azimutwinkel φ beschreibt den Winkel zwischen der x-Achse der Kamera und der Hauptachse des Strahlprofils die am nächsten dazu liegt. Der Winkel der Kamera zum Laser wird durch die Montage der Kamera festgelegt. Bei einfach astigmatischen Strahlen kann der Azimutwinkel bestimmt werden mit

$$\varphi(z) = \frac{1}{2}\arctan\left(\frac{2\sigma_{xy}^2}{\sigma_x^2 - \sigma_y^2}\right) \tag{2.21}$$

für Leistungsdichteverteilungen in denen $\sigma_x^2 \neq \sigma_y^2$ ist. Begrenzt ist der Azimutwinkel in diesem Fall auf $-\frac{\pi}{4} < \varphi < \frac{\pi}{4}$. Für den Fall das $\sigma_x^2 = \sigma_y^2$ ist, kann der Azimutwinkel bestimmt werden mit

$$\varphi = \mathrm{sgn}(\sigma_{xy}^2)\frac{\pi}{4} \tag{2.22}$$

und $\mathrm{sgn}(\sigma_{xy}^2)$ mit

$$\mathrm{sgn}(\sigma_{xy}^2) = \frac{\sigma_{xy}^2}{|\sigma_{xy}^2|}. \tag{2.23}$$

Der Azimutwinkel ist in diesem Fall entweder $-\frac{\pi}{4}$ oder $\frac{\pi}{4}$.

Bestimmung der Strahlabmessungen in Hauptachsenrichtung $d_{\sigma x}$,$d_{\sigma y}$

Die Bestimmung der Strahlabmessungen erfolgt in Richtung der Hauptachsenrichtung. Für Leistungsdichteverteilungen in denen $\sigma_x^2 \neq \sigma_y^2$ können die Laserstrahlabmessungen bestimmt werden mit

$$d_{\sigma x}(z) = 2\sqrt{2}\left\{\left(\sigma_x^2 + \sigma_y^2\right) + \gamma\left[\left(\sigma_x^2 - \sigma_y^2\right)^2 + 4\left(\sigma_{xy}^2\right)^2\right]^{\frac{1}{2}}\right\}^{\frac{1}{2}} \tag{2.24}$$

und

$$d_{\sigma y}(z) = 2\sqrt{2}\left\{\left(\sigma_x^2 + \sigma_y^2\right) - \gamma\left[\left(\sigma_x^2 - \sigma_y^2\right)^2 + 4\left(\sigma_{xy}^2)^2\right)\right]^{\frac{1}{2}}\right\}^{\frac{1}{2}}. \tag{2.25}$$

Dabei ist

$$\gamma = \mathrm{sgn}(\sigma_x^2 - \sigma_y^2) = \frac{\sigma_x^2 - \sigma_y^2}{|\sigma_x^2 - \sigma_y^2|} \tag{2.26}$$

Ist $\sigma_x^2 = \sigma_y^2$ vereinfachen sich die Gleichungen zu

$$d_{\sigma x}(z) = 2\sqrt{2}\left(\sigma_x^2 + \sigma_y^2 + |2\sigma_{xy}^2|\right)^{\frac{1}{2}} \tag{2.27}$$

und

$$d_{\sigma y}(z) = 2\sqrt{2}\left(\sigma_x^2 + \sigma_y^2 - |2\sigma_{xy}^2|\right)^{\frac{1}{2}}. \tag{2.28}$$

Ist die Elliptizität ε

$$\varepsilon = \frac{d_{\sigma,max}}{d_{\sigma,min}} \tag{2.29}$$

größer als 0.87 darf der Laserstrahlquerschnitt als symmetrisch betrachtet werden und kann mit

$$d_{\sigma}(z) = 2\sqrt{2}\left(\sigma_x^2 + \sigma_y^2\right)^{\frac{1}{2}} \tag{2.30}$$

berechnet werden.

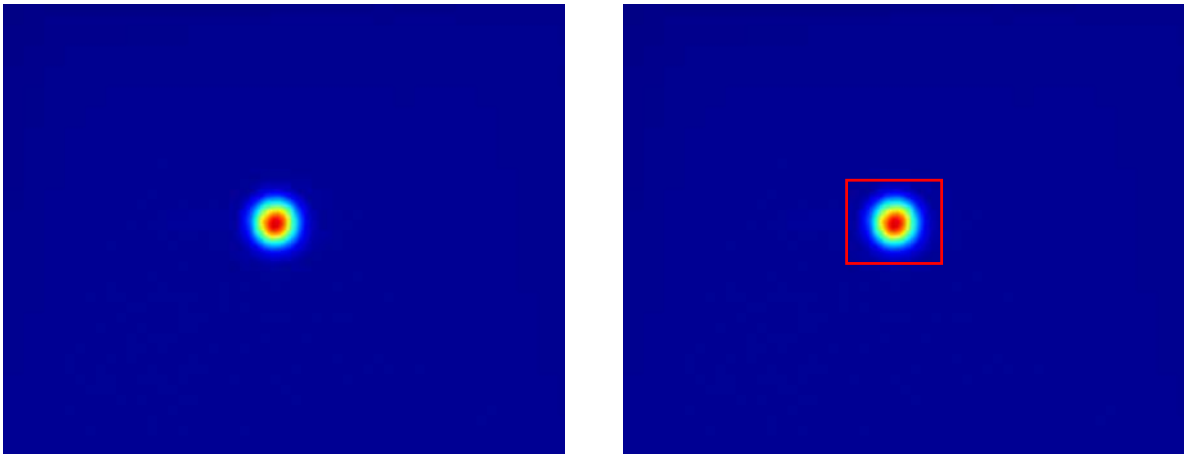
2.6 Anwendung auf Einzelaufnahme eines Strahlprofils mit pixelbasierten Sensor

Um nun aus einer Einzelaufnahme eines Strahlprofils alle bisher beschriebenen Parameter zu berechnen, ist ein geeigneter Integrationsbereich notwendig. Festgelegt ist dafür in der Norm, dass der Integrationsbereich dreimal so groß sein sollte wie die Strahlabmessungen. Aus diesem Grund ist auch die Hintergrundkorrektur essentiell, da alle Pixel mit Werten $\neq 0$ in die Momentenberechnung mit einfließen. Gleichzeitig ist diese Anforderung an den Integrationsbereich eine Limitierung des mit einer bestimmten Kamera messbaren maximalen Strahlabmessungen. Wird zum Beispiel eine Kamera mit einem 6 mm × 6 mm Sensor verwendet darf der Strahldurchmesser an keiner Stelle > 2 mm sein. Mit einer Aufnahme eines Strahlprofils die die Bedingung erfüllt siehe Abb. 2.3a muss noch ein passender Integrationsbereich gewählt werden. Dafür wird im ersten Schritt der Integrationsbereich händisch geschätzt. Die mit diesem Integrationsbereich berechneten Strahlabmessungen können aber, wenn man nicht zufällig den idealen Integrationsbereich auswählt, ungenau sein. Die Lösung dafür ist die Strahlabmessungen iterativ zu bestimmen. Wir legen einen ersten Schätzwert für die Position des Strahls und dessen Strahlabmessungen fest siehe (Abb. 2.3b). Mit diesem geschätzten Integrationsbereich werden die Strahlabmessungen berechnet, anschließend werden aufgrund dieser Strahlabmessungen ein neuer Integrationsbereich festgelegt und wieder die Strahlabmessungen berechnet. Dieser Schritt wird wiederholt bis das Konvergenzkriterium

$$\left.\begin{aligned} |d_{\sigma,x,alt} - d_{\sigma,x,neu}| \\ |d_{\sigma,y,alt} - d_{\sigma,y,neu}| \\ |\overline{x}_{alt} - \overline{x}_{neu}| \\ |\overline{y}_{alt} - \overline{y}_{neu}| \end{aligned}\right\} \leq k_{convergence} \tag{2.31}$$

erfüllt ist.

Abbildung 2.3: Auswahl des geschätzten Integrationsbereichs



(a) Strahlprofil aufgenommen mit einem pixelbasierten Sensor

(b) Strahlprofil mit geschätzten Integrationsbereich

2.7 Bestimmung von Taillenlagen, Divergenzwinkeln und Beugungsmaßzahlen

Für die Bestimmung von Taillenlagen, Divergenzwinkeln und Beugungsmaßzahlen sind mehrere Messpunkte entlang der Ausbreitungsrichtung (z-Achse) des Strahls notwendig. Vorgegeben wird in der Norm, dass die Hälfte der Messpunkte innerhalb einer Rayleighlänge der Strahltaille und die andere Hälfte jenseits von zwei Rayleighlängen liegen sollten. Die Anzahl der Messpunkte sollte ≥ 10 sein. Bestimmt werden die Parameter indem eine hyperbolische Kurve an die quadrierten Strahlabmessungen der Form

$$d^2_{\sigma x,y}(z) = a + bz + cz^2. \tag{2.32}$$

angepasst wird siehe Abb. 2.4. Aus den Koeffizienten a, b und c (bzw. a_x, a_y, b_x, b_y, c_x, c_y) der

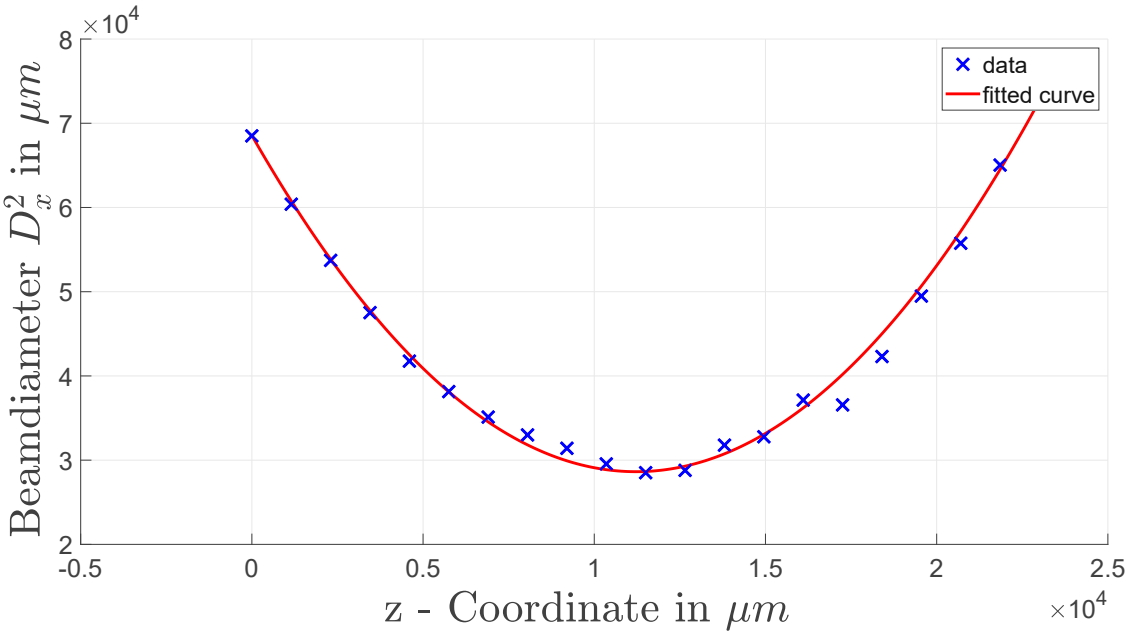


Abbildung 2.4: An die Strahldurchmesser in x-Richtung angepasste hyperbolische Kurve

angepassten Hyperbel kann die Taillenlage mit

$$z_{0x,0y} = \frac{-b_{x,y}}{2c_{x,y}}, \tag{2.33}$$

die Strahlabmessungen der Strahltaille mit

$$d_{\sigma 0x,\sigma 0y} = \frac{1}{2\sqrt{c_{x,y}}}\sqrt{4a_{x,y}c_{x,y} - b_{x,y}^2}, \tag{2.34}$$

die Divergenzwinkel mit

$$\Theta_{\sigma x,\sigma y} = \sqrt{c_{x,y}}, \tag{2.35}$$

die Rayleigh Länge mit

$$z_{Rx,Ry} = \frac{1}{2c_{x,y}}\sqrt{4a_{x,y}c_{x,y} - b_{x,y}^2} \tag{2.36}$$

und die Beugungsmaßzahl mit

$$M_{x,y}^2 = \frac{\pi}{8\lambda}\sqrt{4a_{x,y}c_{x,y} - b_{x,y}^2} \tag{2.37}$$

bestimmt werden.

3 Messaufbau

Die verfügbare Fläche vor dem FSL ist begrenzt durch opto-mechanische und mechanische Komponenten, die für Oberflächenstrukturierung von Metallen verwendet werden (siehe Abb. 3.1). Aus

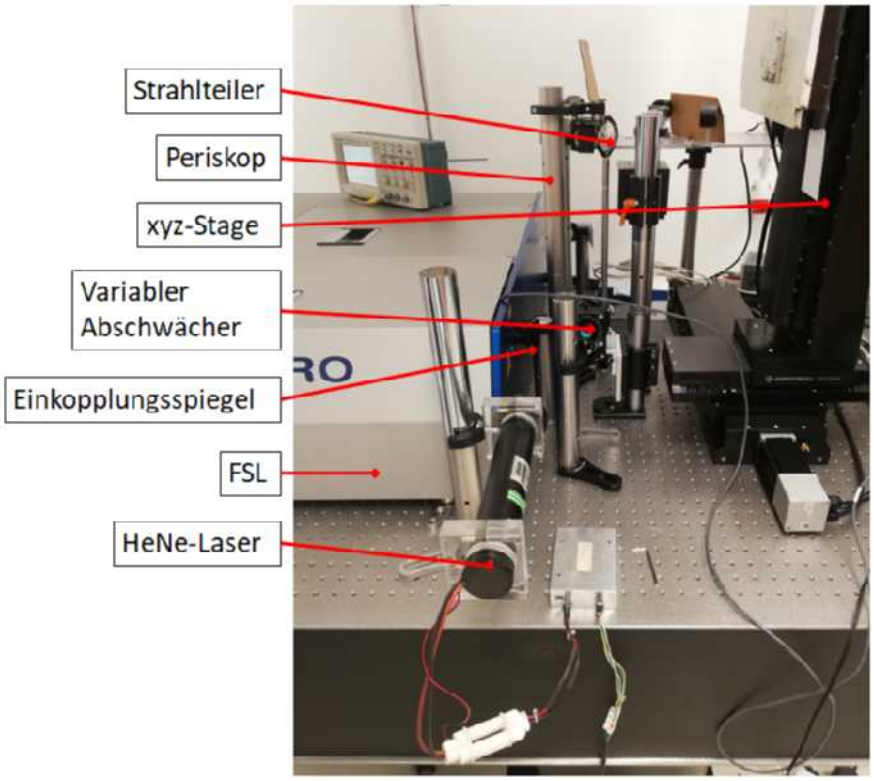


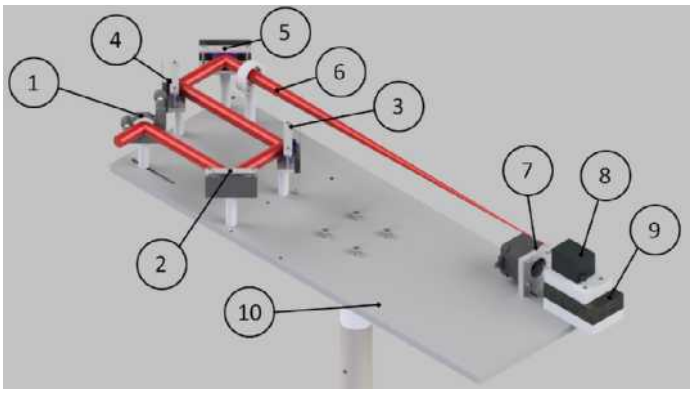
Abbildung 3.1: Anordnung opto-mechanischer Elemente vor dem FSL

diesem Grund war es notwendig den Messaufbau so kompakt wie möglich zu gestalten. Entworfen wurde der Messaufbau mithilfe von CAD, wodurch sehr schnell und effizient Änderungen vorgenommen werden konnten. Der entworfene Messaufbau wurde in einem Vorversuch mit einem He-Ne Laser getestet und mit der gewonnenen Erfahrung überarbeitet. Auf diese Weise wurde der Messaufbau zweimal überarbeitet.

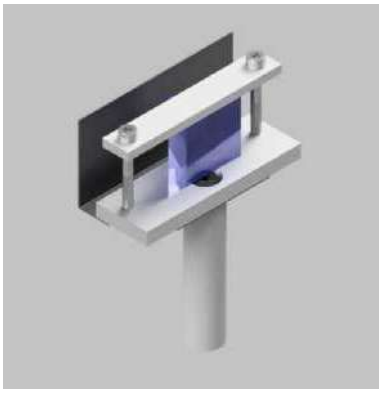
3.1 Messaufbau 1

Messaufbau 1 besteht aus einer Aluminium Grundplatte (10) die auf einem vertikalen Steher aus Stahl montiert wurde, einem silberbeschichteten Einkopplungsspiegel (1), 4 rechtwinkligen Prismen (2) - (5), einer Fokussierlinse mit Brennweite 500 mm (6), einer Kamera mit CMOS Sensor (8) und einem Linearschlitten mit maximalen Verfahrweg 23 mm (9) angetrieben mithilfe einea Schrittmotors (7). Der Laserstrahl durchläuft den Messaufbau beginnend mit dem silberbeschichteten Spiegel (1). Er übernimmt das Einkoppeln des Laserstrahls in das Messsystem und wurde auf einem justierbaren Spiegelhalter montiert. Nach dem Einkopplungsspiegel wurde der Laserstrahl durch Reflexion, an den Prismen (2) - (5) , umgelenkt und gleichzeitig aufgrund der Reflexions- bzw. Transmissionseigenschaften der Prismen abgeschwächt. Bestimmt werden die Reflexions- und

Abbildung 3.2: Messaufbau Version 1



(a) Übersicht Messaufbau Version 1



(b) Aufbau eines Prismahalters

Transmissionskurven vom Werkstoff der Prismen. Für die Bandbreite des FSL von 40 nm mit einer zentralen Wellenlänge von 800 nm und dem He-Ne Laser mit einer Wellenlänge von 632.8 nm sind Prismen aus BK7 geeignet, da sie in diesem Bereich nahezu konstante Eigenschaften haben (siehe Abb . 3.9). Jede Prisma reflektiert $\sim 10\,\%$ der auftreffenden Strahlung. Dadurch ergibt sich bei serieller Reflexion an vier Prismen ein Abschwächungsfaktor von 10^{-4}. Die Prismen wurden in eigens dafür konstruierten Prismahaltern montiert. Sie bestehen aus zwei Platten die über zwei Schrauben miteinander verbunden sind siehe Abb. 3.2b. Das Prisma befindet sich zwischen den beiden Platten und kann über die Schrauben fixiert werden. Auf der Rückseite des Prismahalters sind zusätzlich matt schwarz lackierte Strahlabfangbleche montiert, die transmittierte Strahlung abfangen sollten. Das letzte optische Element bildete eine Fokussierlinse ⑥ aus BK7 die den Laserstrahl auf die Kamera ⑦ fokussierte. Die Fokussierung war notwendig, da der Rohstrahl des FSL größer ist als der verbaute Sensor in der Kamera (siehe Kap. 3.4.6. In der Norm wird diese durch die Fokussierlinse erzeugte Strahltaille als künstliche Strahltaille bezeichnet. Die Kamera wurde auf einem Linearschlitten montiert der von einem 24 V Schrittmotor angetrieben wurde. Der Verfahrweg pro Schritt des Schrittmotors ist aus in einem Versuch ermittelt worden. Dabei wurde die Position z_0 des Linearschlittensschlitten zu einer Referenzkante (Hinterkante der Linearschlitten) gemessen. Nach separaten Verfahren des Linearschlittens um 1000 Schritte, wurde der Abstand z_{1000} zur Referenzkante gemessen. Der Verfahrweg pro Schritt ergibt sich zu

$$\Delta z_i = \frac{|z_0 - z_{1000}|}{1000}. \tag{3.1}$$

Die Messung wurde 20 Mal wiederholt und aus den Einzelergebnissen der Mittelwert mit

$$\Delta z = \frac{1}{n}\sum_{i=1}^{n} \Delta z_i \tag{3.2}$$

gebildet. Daraus ergab sich ein Verfahrweg pro Schritt von $1.8 \mu m$. Gesteuert wird der Schrittmotor über ein Arduino UNO Rev3 Microcontroller-Board. Als Treiberstufe wird ein Arduino Motor Shield Rev3 verwendet. Die Grundplatte bildet eine unabhängige Messebene, wodurch der Messaufbau ohne großen Aufwand zum Vermessen anderer Laserquellen verwendet werden kann. Die Höhe der Grundplatte bezogen auf die Montageoberfläche des Stehers ist von 375 mm bis 415 mm einstellbar.

3.1.1 Vorversuch mit einem He-Ne Laser

Bevor das Messsystem am FSL getestet wurde, ist ein Vorversuch mit einem He-Ne Laser durchgeführt worden. Der Laser hat eine Zentral-Wellenlänge von 632.8 nmnm, einen $1/e^2$ Durchmesser

von 1.02 mm, und eine Ausgangleistung von 7 mW. Für den Versuch wurde der Laserstrahl wie in Abb. 3.2a dargestellt über den Einkopplungsspiegel ① ins Messsystem eingekoppelt und mit der Kamera ⑦ das Strahlprofil aufgenommen. Die Ziele des Vorversuchs waren es das Messprogramm zu testen, die Funktionalität des Messaufbaus sicher zu stellen und die optischen Elemente so genau wie möglich auszurichten. Die Ausrichtung jedes einzelnen optischen Elements war einfach durchzuführen, aber alle Komponenten im Zusammenspiel miteinander auszurichten war sehr arbeitsintensiv. Ausgerichtet wurden die Komponenten in der Reihenfolge wie der Laserstrahl sie durchläuft ① - ⑥. Der Prozess wurde solange wiederholt bis der Laserstrahl den Kamera Sensor traf.

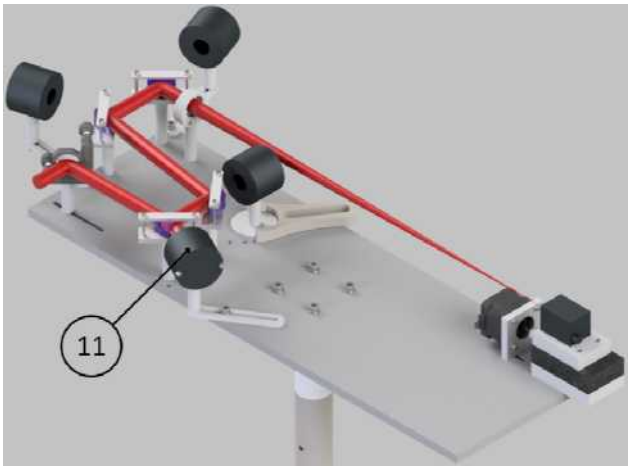
3.1.2 Ergebnisse des Vorversuchs mit Messaufbau 1

Die Ergebnisse des Vorversuchs zeigten, dass es durch die geometrische Anordnung der Prismen und Bleche zu Reflexionen an den Strahlabfangblechen kam. Die Reflexionen überlagerten sich mit dem Messignal, wodurch die Messergebnisse unbrauchbar wurden.

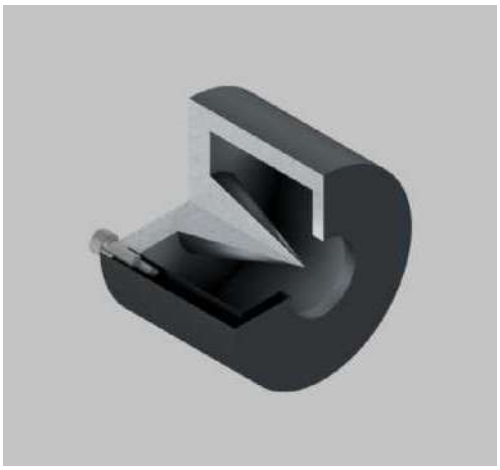
3.2 Messaufbau 2

Messaufbau 2 hatte denselben Grundaufbau wie Messaufbau 1, allerdings wurden die Strahlabfangbleche entfernt und durch Strahlabsorber ⑪ ersetzt siehe Abb. 3.3a. Strahlabsorber sind Körper

Abbildung 3.3: Messaufbau Version 2 und Strahlfalle



(a) Übersicht Messaufbau Version 2



(b) Dreiviertelschnitt einer Strahfalle

die Laserstrahlen durch Absorption in Wärme umwandeln sollen. Die einfachste Ausführung für Absorber sind passiv oder aktiv gekühlte Platten. Eine weitere Möglichkeit ist es die Strahlabsorber als "Fallen" zu konzipieren in die der Laserstrahl durch eine Öffnung eintreten, aber durch geschickt gewählte Geometrien im inneren der Falle nicht mehr austreten bevor er vollständig absorbiert wird. Wie die Plattenabsorber können auch die Strahlfallen passiv oder aktiv gekühlt ausgeführt werden. Die Strahlabsorber für Messaufbau 2 sind als passiv gekühlte Strahlfallen konzipiert worden (siehe Abb. 3.3b). Sie bestehen aus zwei Teilen. Einem zylindrischen Absorbtionskörper mit der Eintrittsbohrung und einer Abschlussplatte mit Spitzkegel. Der Spitzkegel absorbiert einen Teil der Strahlung und reflektiert den restlichen Anteil in Richtung der Absorptionskörperwände, die wiederum Strahlung absorbieren und reflektieren. Auf diese Art wird der Strahl vielfach im Körper reflektiert bis er vollständig absorbiert ist. Um die Absorptionsfähigkeit der Oberflächen zu steigern wurden diese zusätzlich matt schwarz lackiert.

3.2.1 Ergebnisse des Vorversuchs mit Messaufbau 2

Die Ergebnisse des Vorversuchs mit Messaufbau 2 zeigten, dass die transmittierte Strahlung innerhalb des Prismas reflektiert wurde. Das führte dazu, dass der an der Rückseite des Prismas reflektierte Laserstrahl das Prisma an der Frontfläche als Sekundärstrahl seitlich verschoben aber parallel zum Primärstrahl austrat siehe Abb .3.4. Dieser parallelen Sekundärstrahlen traten bei jedem Prisma auf, wodurch an der Fokussierlinse mehrere örtlich verschobene aber annähernd parallele Strahlen auftrafen. Diese wurden von der Linse auf die Kamera fokussiert, wodurch es zu Interferenzen

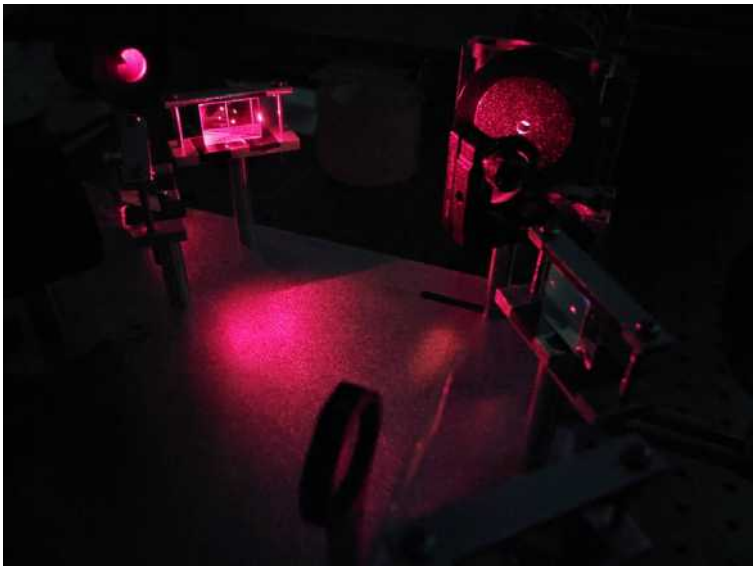
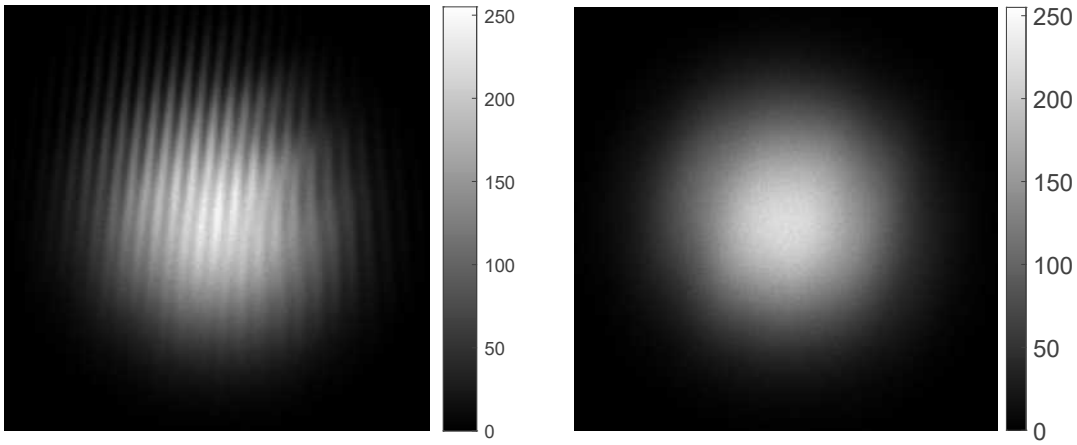


Abbildung 3.4: Reflexionen im Prisma

im gemessenen Strahlprofil kam. Die ungewünschten Laserstrahlen konnten im Vorversuch mittels Schablone abgeblockt werden was zu einer Unterdrückung der Interferenzen führte. Zu sehen ist der Einfluss dieser sekundären Reflexionen in Abb. 3.5a. Abb. 3.5b zeigt das Messergebnis mit unterdrückten Sekundärstrahlen. Eine weitere Möglichkeit der Veranschaulichung von Interferenzen ist

Abbildung 3.5: Laserstrahlprofil

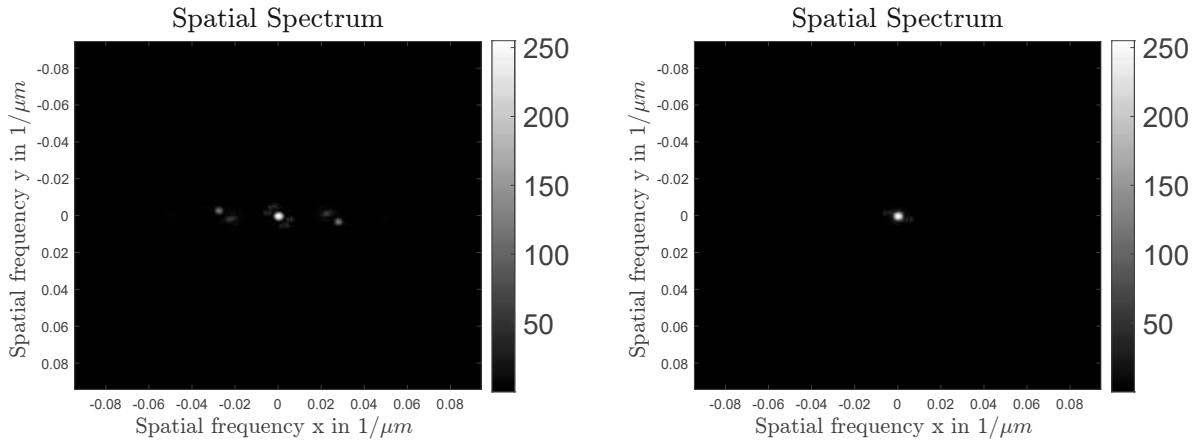


(a) Mit Interferenzen

(b) Ohne Interferenzen

die Anwendung eine 2D FastFourierTransformation(FFT) wodurch ein Bild in seine Spektralanteile zerlegt wird siehe Abb. 3.6a und Abb. 3.6b. In den Spektren wurden die Intensitäten logarithmisch

Abbildung 3.6: Ortsfrequenzspektrum eines Laserstrahlprofils



(a) Mit Interferenzen

(b) Ohne Interferenzen

skaliert als Grauwerte im Bereich von $0 - 255$ und in x- und y-Richtung die Ortsfrequenzen aufgetragen. Durch die logarithmische Skalierung wurden die Hauptanteile eindeutig sichtbar. Abb. 3.6b der Messung ohne Interferenzen zeigte, dass sich das Strahlprofil hauptsächlich aus niedrigen Ortsfrequenzen zusammensetzt. Verglichen mit Abb. 3.6a dem Strahlprofil mit Interferenzen fiel auf, dass die zentralen Teile beider Spektren beinahe identisch waren. Sie unterschieden sich nur über zusätzliche Anteile mit höhere Ortsfrequenz in x-Richtung. Im Strahlprofil siehe Abb. 3.5a waren diese als Streifen in y-Richtung erkennbar. Damit die Sekundärstrahlen in Messaufbau 2 mit einer Schablone unterdrückt werden konnten mussten die Strahlabmessungen klein sein. Der He-Ne Laser hat einen Strahldurchmesser von 1.02 mm wodurch die Sekundärstrahlen unterdrückt werden konnten. Im Gegensatz dazu hat der FSL laut Datenblatt einen Durchmesser von 15 mm, wodurch der Sekundärstrahl sich bereits vor der Fokussierlinse mit dem Primärstrahl überlagerte und eine Schablone keine Option darstellte.

3.3 Messaufbau 3

Um die Interferenzerscheinungen zu eliminieren wurde Messaufbau 3 neu konzipiert. Das bisher verwendete Konzept bei dem die Abschwächung des Laserstrahls durch Reflexion an Prismen erreicht wurde ist verworfen worden. Ersetzt wurden sie mit Neutraldichte Filter (ND-Filter) (12), die deren

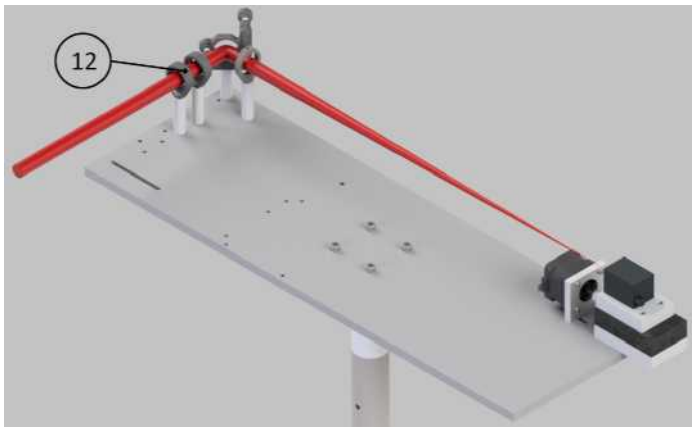


Abbildung 3.7: Messaufbau Version 3

Funktion übernahmen siehe Abb. 3.7. Für die ND-Filter wurden Filter mit einer optischen Dichte von 5.0 bzw. 4.0 verwendet. Der zuvor am Anfang des Laserstrahls platzierte Einkopplungsspiegel wurde ans Ende verschoben und dazu verwendet den Laserstrahl auf die Kamera auszurichten.

3.3.1 Ergebnisse aus dem Vorversuch mit Messaufbau 3

Die Ergebnisse des Vorversuchs mit Messaufbau 3 zeigten ein Interferenzfreies Strahlprofil.

3.4 Verwendete Hardware und Software

3.4.1 Unbeschichtetes rechtwinkliges Prisma (Thorlabs PS908) [10]

Das Prisma PS908 von Thorlabs ist ein unbeschichtetes rechtwinkliges Prisma aus BK7 mit einer Kantenlänge von 20 mm und einer Höhe von 20 mm. Es besitzt im Wellenlängenbereich von 600 nm

Abbildung 3.8: Rechtwinkliges Prisma PS908 [11]

- 1000 nm nahezu konstante Transmissionseigenschaften (siehe Abb .3.9), wodurch es sich sowohl für den Vorversuch mit dem He-Ne Laser, als auch den Versuch am FSL eignet.

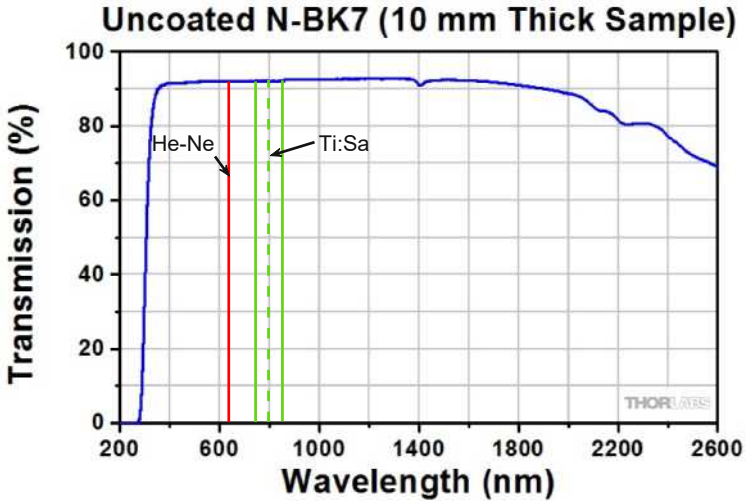


Abbildung 3.9: Transmissionskurve von unbeschichtetem BK7 [12]

3.4.2 Neutraldichtefilter ND-Filter (Thorlabs NE40B-B, NE50B-B) [13]

Verwendet wurden OD 4.0 und OD 5.0 Filter vom Typ NE40B-B und NE50B-B der Firma Thorlabs. Es sind Antireflektionsbeschichtete Filter die für einen Wellenlängen Bereich von 650 nm - 1050 nm nahezu konstante Transmissionseigenschaften besitzt. Aufgrund der Beschichtung ist bei der Handhabung und Reinigung besondere Vorsicht geboten um Beschädigungen der Beschichtung

Abbildung 3.10: ND Filter NE40B-B, NE50B-B [14]

zu vermeiden. Die Transmissionskurven für OD 2.0 - OD 6.0 sind in Abb. 3.11 abgebildet. Relevant für diese Arbeit sind die grüne(OD 4.0) und die gelbe (OD 5.0) Kurve. Das OD 4.0 Filter hat bei einer mittleren Wellenlänge von 800 nm eine Transmissionsrate von ca. 0.3 % und das OD 5.0 Filter ca. 0.09 %. Verwendet wurden die Filter um die Fluenz des Lasers soweit abzuschwächen das die Pixel der Kamera weder zerstört noch gesättigt werden.

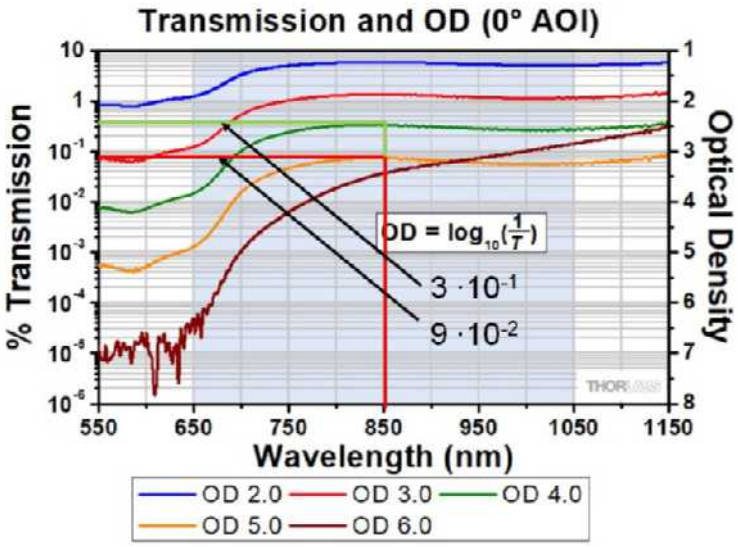


Abbildung 3.11: ND-Filter Transmissionskurven für OD 2.0 - OD 6.0 [15]

3.4.3 Schrittmotor (Sanyo Denki 103H5205-5240) [16]

Der zweipolige Schrittmotor vom Typ 103H5205-5240 Sanyo Denki besitzt bei einem Nennstrom von 1 A/*Phase* ein Haltemoment von 0.265 N m. Wie bei allen E-Motoren hängt das verfügbare

Abbildung 3.12: Schrittmotor Sanyo Denki 103H5205-5240 [17]

Drehmoment von der Drehzahl ab. Bis zu einer Drehzahl von 100 min^{-1} ist er in der Lage das maximale Drehmoment bereitzustellen siehe Abb. 3.13. Er wurde verwendet um die Linearschlitten anzutreiben. Gesteuert wurde er über ein Arduino UNO Rev3 Microcontroller-Board mit zusätzlichem Arduino motor shield Rev3 Treiberboard direkt aus MATLAB.

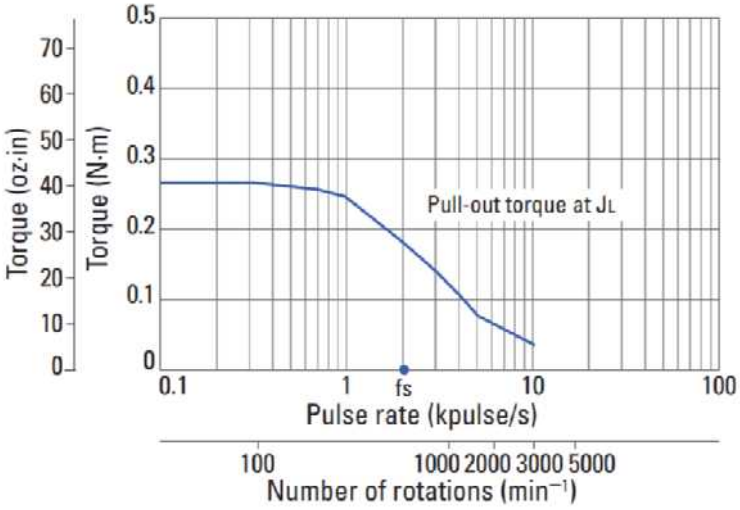


Abbildung 3.13: Momentenverlauf Sanyo Denki 103H5205-5240 [18]

3.4.4 Steuerboard (Arduino UNO Rev3) [19]

Das Arduino UNO Rev3 board ist ein sehr einfach zu programmierendes Board das viele Möglichkeiten in seiner Verwendung bietet. Da das zu erstellende Messprogramm in MATLAB geschrieben

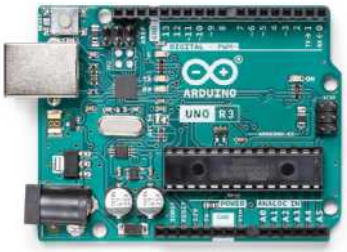


Abbildung 3.14: Arduino UNO Rev3 [20]

wurde eignete sich dieses Board in Verbindung mit einem Arduino Motor Shield Rev3 ideal, weil es bereits vorgefertigte Toolboxen gibt um eine Verbindung mit dem Board und Periphere Hardware wie zum Beispiel einen Motor herzustellen. Es wurde auch für die Überwachung eines Endschalters verwendet der sicher stellt, dass der Linearschlitten nicht den Rand erreicht.

3.4.5 Treiberboard (Arduino Motor Shield Rev3) [21]

Mit dem Arduino Motor Shield Rev3 können sowohl Schrittmotoren als auch Gleichstrommotoren angesteuert werden. Es ist direkt kompatibel mit dem Arduino UNO Rev3 und bietet die Möglichkeit direkt aus MATLAB heraus über das Arduino UNO Rev3 board verwendet zu werden. Es wurde verwendet um den Schrittmotor anzutreiben.

Abbildung 3.15: Arduino Motor Shield Rev3 [22]

3.4.6 Kamera (Basler ace GigE acA1300-60gmNIR) [23]

Die Basler ace GigE acA1300-60gmNIR ist eine NIR Kamera mit 1.2 MP und einer maximalen Bildrate von 60 Bildern pro Sekunde. In ihr verbaut ist ein e2v EV76C661 mono CMOS Sensor mit einer Auflösung von 1282 x 1026 (BxH Pixel), einer Pixelgröße von 5.3 μm x 5.3 μm und sowohl global als auch progressive shutter. Global shutter bedeutet das alle Pixel zum selben Zeitpunkt ausgelesen werden und progressive shutter, dass eine Pixelreihe nach der anderen ausgelesen wird. Die Verbindung zur Kamera besteht über Ethernet. Ausgelöst wird die Kamera entweder über einen Software Trigger oder einen externen Hardware Trigger. Bei der Inbetriebnahme sollten immer die vom Hersteller zur Verfügung gestellten Anleitungen verwendet werden um Verbindungsproblemen mit der Kamera vorzubeugen.

Abbildung 3.16: Basler ace GigE acA1300-60gmNIR [24]

3.4.7 MATLAB R2020b

MATLAB ist eine Software zur Analyse und Visualisierung von Daten, entwickeln von Algorithmen, erstellen von Benutzeroberflächen und Steuern und auslesen von Hardware. Es bietet sehr viele Toolboxen mit vorgefertigten Funktionen an die immer weiterentwickelt bzw. um neue Funktionen ergänzt werden. Jeder Entwickler hat zusätzlich die Möglichkeit seine Programme oder Toolboxen mit anderen zu Teilen wodurch die Auswahl an Toolboxen stetig steigt. Mit ihm wurde die gesamte Software entwickelt. Das umfasst das Steuern der Hardware, das Auswerten der Daten, das Visualisieren der Daten, und der Bereitstellung einer Benutzeroberfläche mit der jede Funktion bedient werden kann.

4 Messprogramm

Im Vordergrund der Arbeit stand das Erstellen eines Messprogramms mit dem automatisiert die Strahlparameter auf Basis der DIN EN ISO-11146 gemessen, ausgewertet und angezeigt werden können. Eine weitere Anforderung an das Prüfprogramm war die Implementierung einer Livemessung, um einen möglichen Einfluss der Laserjustage auf das Strahlprofil sichtbar zu machen. Zur Eingabe der benötigten Prüfparameter musste eine anwenderfreundliche Benutzeroberfläche geschaffen werden. Um alle diese Anforderungen umzusetzen wurde dabei die Programmierumgebung MATLAB (MathWorks) genutzt. Es bietet zusätzlich zu seiner Grundfunktionalität prozedural mathematische Berechnungen durchzuführen und Benutzeroberflächen zu erstellen, vorgefertigten Bibliotheken für Bildverarbeitung und steuern bzw. auslesen von externer Hardware.

4.1 Klassenaufbau

Die Benutzeroberfläche und deren Elemente, im englischen auch "Frontend" genannt, wurden vollständig von der Auswertungsfunktionalität die im Hintergrund arbeitet, im englischen "Backend" genannt, getrennt. Eine solche Trennung bietet den Vorteil, dass es möglich ist Änderungen oder zusätzliche Funktionalität nachträglich einzubauen ohne gegenseitige Beeinflussung von Frontend und Backend. MATLAB arbeitet grundsätzlich prozedural, unterstützt aber auch eine eigene Form der objektorientierten Programmierung die ich im Backend dazu nutzte eigene Datenelemente als Klassen zu definieren. Die Klassen sind hierarchisch aufeinander aufgebaut, wobei jede Klasse auch

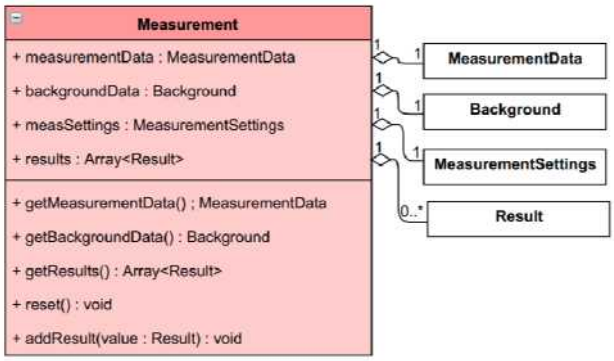


Abbildung 4.1: Klassendiagramm Measurement

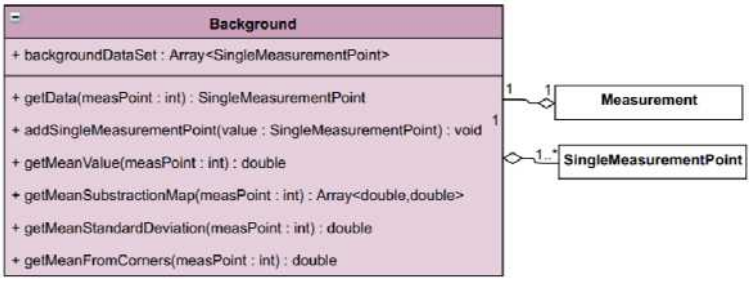


Abbildung 4.2: Klassendiagramm Background

alleinstehend verwendet werden kann. Die oberste Klasse bildet die Klasse *Measurement*. Sie beinhaltet alle Daten einer Messung und besteht aus vier untergeordneten Klassen (siehe Abb.4.1). Die vier untergeordneten Klassen sind die Klasse *MeasurementData* in der die Messdaten enthalten sind, die Klasse *Background* in der die Hintergrunddaten enthalten sind, die Klasse *MeasurementSettings* in der die Messparameter gespeichert sind und die Klasse *Result* in der ausgewertete Messdaten enthalten sind.

Abbildung 4.3: Klassendiagramm MeasurementData

Die Klasse *Background* (siehe Abb.4.2) besteht nur aus einer Liste von *SingleMeasurementPoint*

Objekten, wobei jedes Listenelement einen der Messpunkte darstellt. Auf die Klasse *SingleMeasurementPoint* wird noch im Detail eingegangen. Die Methoden der Klasse *Background*, dienen der Datenbereitstellung für die Hintergrundkorrektur, die ein wichtiger Bestandteil der Auswertung ist.

Die Klasse *MeasurementData* ist das Herzstück der Messung (siehe Abb.4.3). Es enthält, wie bereits die Klasse *Background*, ebenfalls eine Liste von *SingleMeasurementPoint* Objekten, aber zusätzlich dazu noch alle übergreifenden Strahlparameter die für ein Messergebnis bestimmt werden müssen. Die Methoden dieser Klasse beinhalten sowohl die wichtigsten Berechnungsoperationen der Hintergrundkorrektur und der Parameterbestimmung laut DIN EN ISO 11146, als auch die notwendigen GET Funktionen um alle Daten abzurufen.

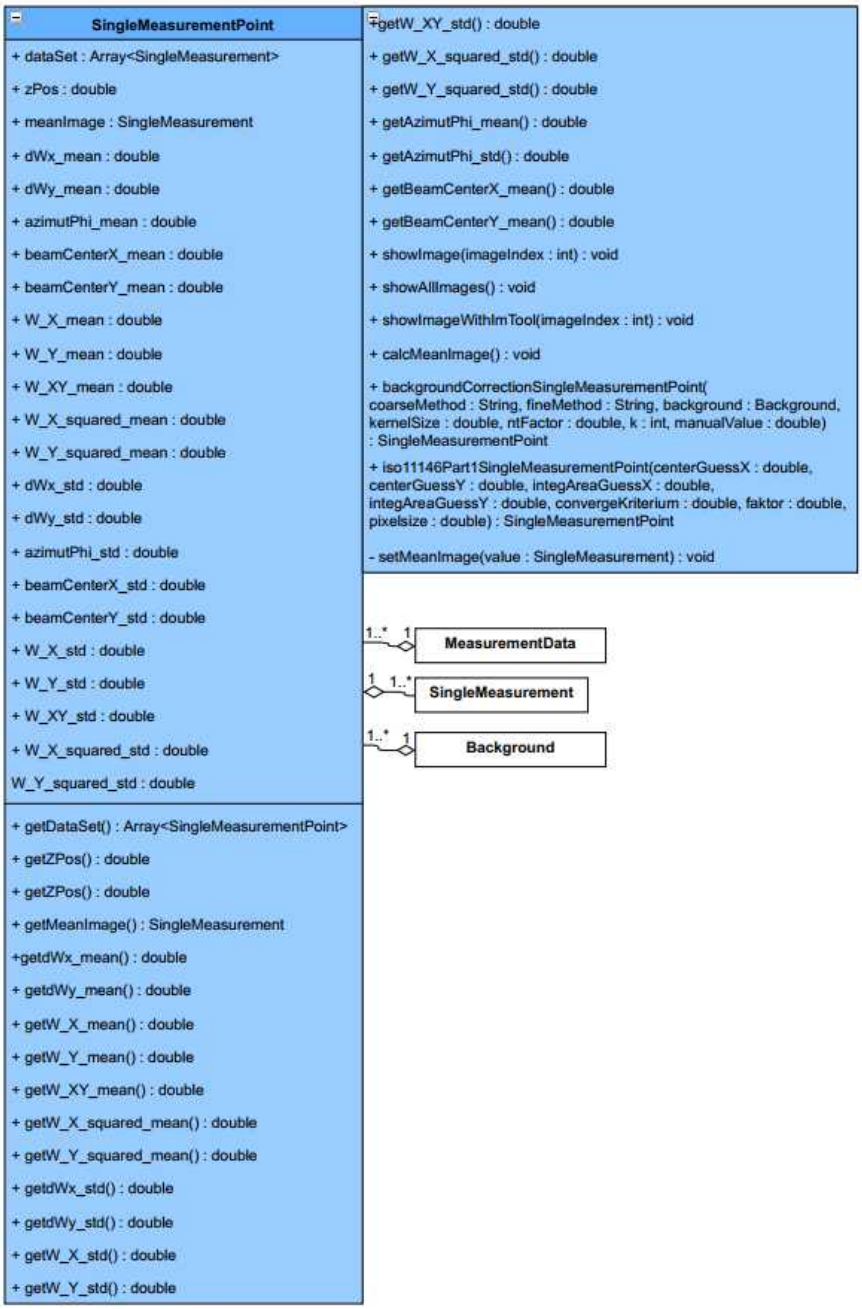


Abbildung 4.4: Klassendiagramm SingleMeasurementPoint

Die Klasse *SingleMeasurementPoint* (siehe Abb.4.4) repräsentiert einen einzelnen Messpunkt und besitzt einen sehr ähnlichen Aufbau wie die Klasse *MeasurementData*. Sie besteht aus einer Liste von *SingleMeasurement* Objekten die eine Einzelmessung beschreiben und den aus den Einzelmessungen bestimmten gemittelten Strahlparametern.

Die Klasse *SingleMeasurement* (siehe Abb.4.5) ist das unterste Element des hierarchischen Aufbaus und stellt eine Einzelmessung dar. Darin enthalten ist ein aufgenommenes Bild gespeichert in Form einer $n \times m$ Matrix in der die Graustufenwerte jedes einzelnen Pixels hinterlegt sind und den berechnete Strahlparametern einer Einzelmessung.

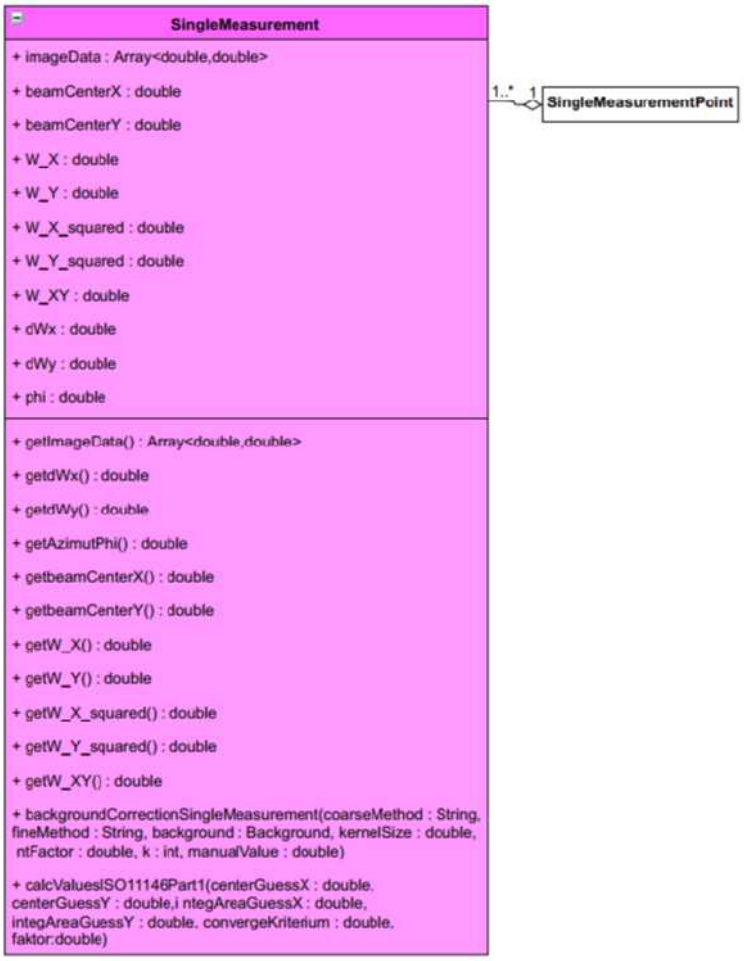


Abbildung 4.5: Klassendiagramm SingleMeasurement

4.2 Programmabläufe und GUI

Bisher wurden nur die Klassen und ihre Beziehungen zueinander beschrieben. Als nächstes werden jetzt die Abläufe in dem umgesetzten Messprogramm anhand von Ablaufdiagrammen beschrieben. Eine der geforderten Funktionen ist die Automatisierte Messung der Strahlcharakteristik eines Laserstrahls. Der Ablauf einer Messung wurde auf mehrere Schritte aufgeteilt (siehe Abb.4.8).

Der erste Schritt einer Messung ist das Auslesen der Messparameter aus einer grafischen Benutze-

roberfläche (GUI). Die Messparameter, die dafür festgelegt werden müssen, sind das Messgitter, die Anzahl der Messungen pro Messpunkt und die Belichtungszeit (ExposureTime) der Kamerapixel.

4.2.1 Definition des Messgitters

Für die Festlegung des Messgitters in Ausbreitungsrichtung (Z-Achse) des Laserstrahls (siehe Abb. 4.6) wurden zwei Konfigurationsmethoden gefordert. Eine Konfigurationsmethode die das Messgitter symmetrisch um einen zentralen Punkt verteilt und eine zweite die das Messgitter ausgehend vom Nullpunkt des Linearschlittens auf einen einstellbaren Bereich verteilt. Die Methode der Verteilung um einen zentralen Punkt ist am besten geeignet, wenn die Position der Strahltaille (Fokuspunkt) des Laserstrahls bekannt ist. Die Methode der Verteilung ausgehend vom Nullpunkt des Linearschlittens ist am besten dafür geeignet die Position der Strahltaille initial zu bestimmen. Die Methode zur Verteilung des Messgitters um einen zentralen Punkt (siehe Abb. 4.7a) benötigt

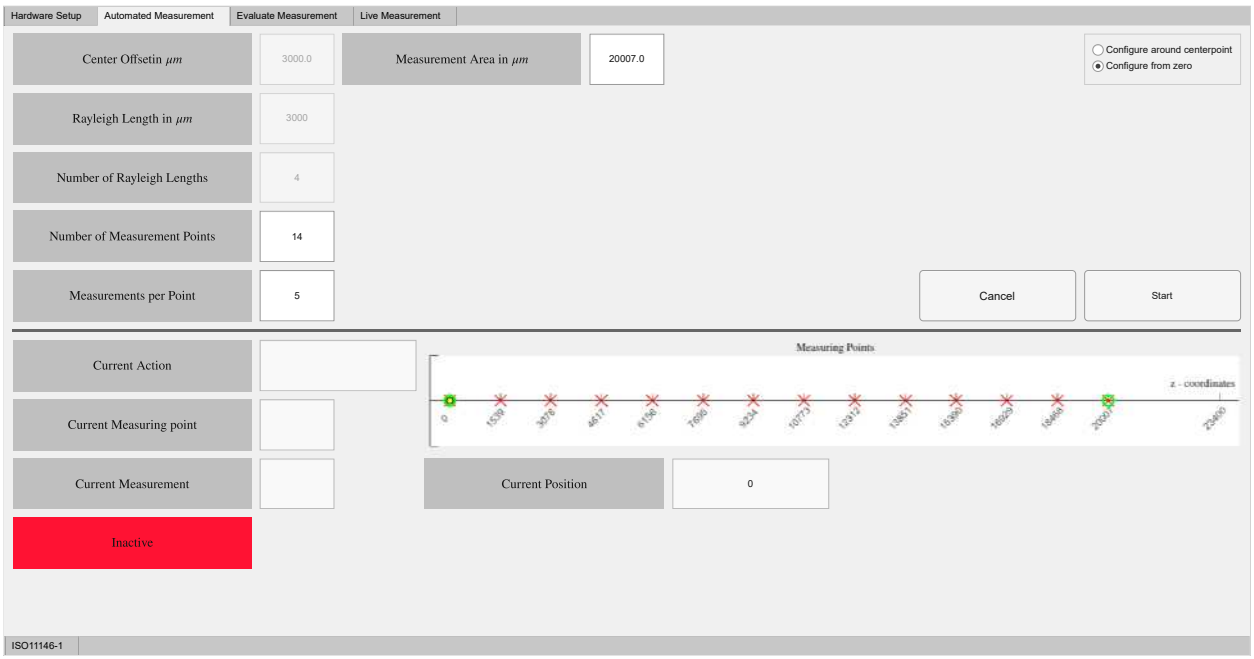
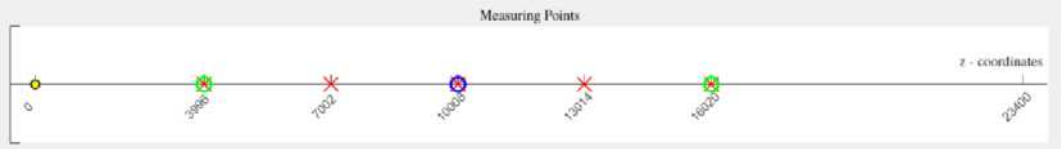


Abbildung 4.6: Messgitterkonfiguration

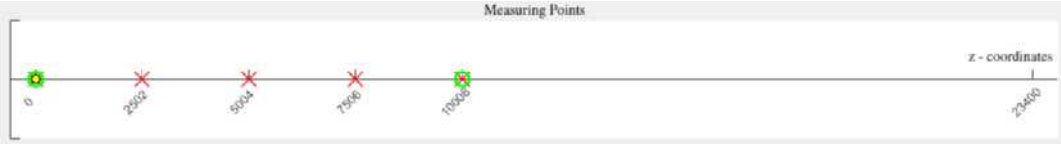
vier Parameter um das Messgitter festzulegen. Die Position des zentralen Punktes ausgehend vom Nullpunkt des Linearschlittens, die Rayleighlänge des zu vermessenden Laserstrahls, der zusammen mit einem einstellbaren Multiplikator den Messbereich in beide Richtungen vom zentralen Punkt festlegt und die Anzahl der Messpunkte. Damit werden die Messpunkte auf den festgelegten Messbereich verteilt. Sollte es nicht möglich sein ein homogenes Messgitter auf den Messbereich zu verteilen, bei dem jeder Messpunkt genau angefahren werden kann (limitiert durch die Schrittweite des Motors), wird die Anzahl der eingestellten Messpunkte automatisch angepasst, sodass jeder Messpunkt angefahren werden kann.

Bei der zweiten Konfigurationsmethode für die Verteilung der Messpunkte wird das Messgitter ausgehend vom Nullpunkt des Linearschlittens (siehe Abb.4.7b) auf eine einstellbare Länge verteilt. Benötigt werden dazu nur zwei Parameter. Die Länge des Messbereichs und die Anzahl der Messpunkte. Das Messgitter wird gleich zur Methode um einen zentralen Punkt auf den Messbereich aufgeteilt und die Anzahl der Messpunkt angepasst, sollte es notwendig sein.

Abbildung 4.7: Messgitter Konfigurationsmethoden



(a) Konfiguration um einen zentralen Punkt



(b) Konfiguration ausgehend vom Nullpunkt des Linearschlittens

4.2.2 Messzyklus

Nach dem Erstellen des Messgitters beginnt ein Messzyklus. Zuerst wird dafür ein neues Objekt *Measurement* erstellt in dem die Messdaten abgespeichert werden sollen. Der Messzyklus besteht aus einer Hintergrundmessung und einer Hauptmessung. Für die Hintergrundmessung muss der Laserstrahl abgeblockt werden, damit rein der Hintergrund gemessen werden kann. Ist das sichergestellt läuft die Hintergrundmessung wie in Abb.4.10 dargestellt ab. Zu Beginn wird ein neues Objekt *Background* erstellt in dem die Hintergrunddaten abgespeichert werden. Danach werden die Messpunkte nacheinander angefahren, wobei für jeden Messpunkt ein Objekt *SingleMeasurementPoint* erstellt wird in dem die Messdaten eines Messpunktes abgespeichert werden. Anschließend wird für jeden Messpunkt die zuvor eingestellte Anzahl der Messungen durchgeführt. Für jede Einzelmessung wird die Kamera über einen Softwaretrigger ausgelöst, das Bild ausgelesen, und mit dem Bild ein Objekt *SingleMeasurement* erstellt. Die *SingleMeasurement* Objekte werden immer dem jeweiligen *SingleMeasurementPoint* hinzugefügt und der *SingleMeasurementPoint* dem *Background*. Das Endergebnis ist eine wie im Klassenaufbau beschriebene hierarchische Struktur. Ist die Hintergrundmessung abgeschlossen kann der Laserstrahl wieder freigegeben werden und die Hauptmessung gestartet. Der Ablauf der Hauptmessung siehe Abb.4.9 ist identisch zur Hintergrundmessung, nur wird anstelle eines Objektes *Background* ein Objekt *MeasurementData* erstellt in dem die Messdaten abgelegt werden.

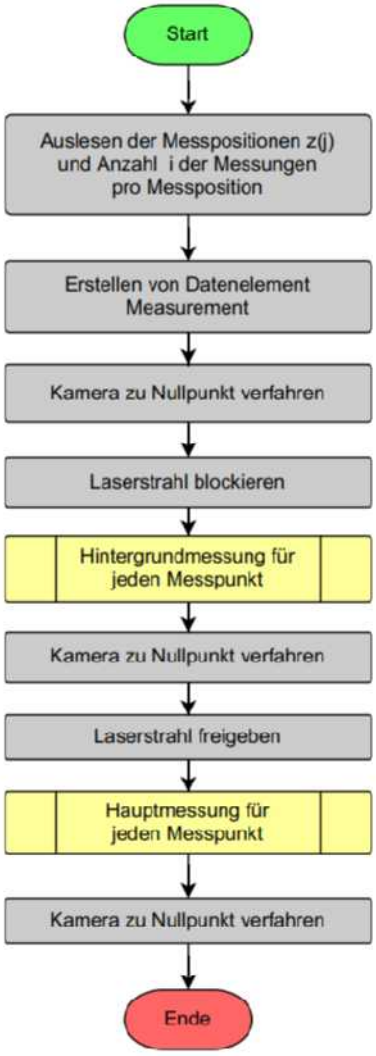


Abbildung 4.8: Ablaufplan Automatisierte Messung

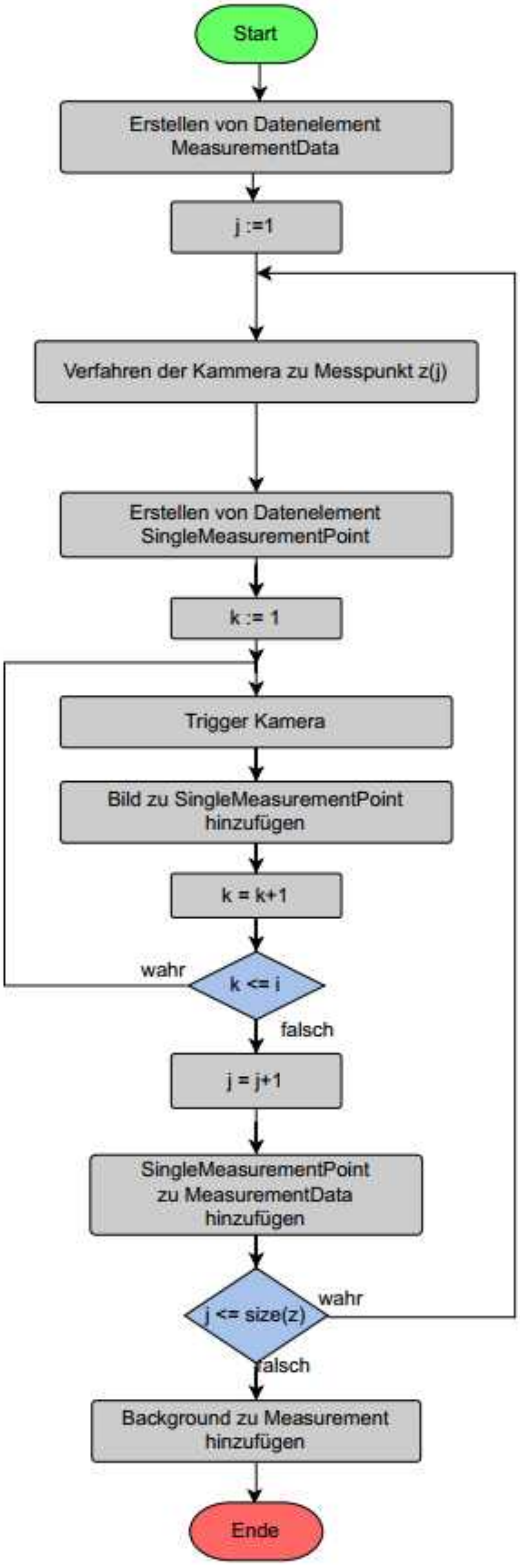


Abbildung 4.9: Ablaufplan Hauptmessung

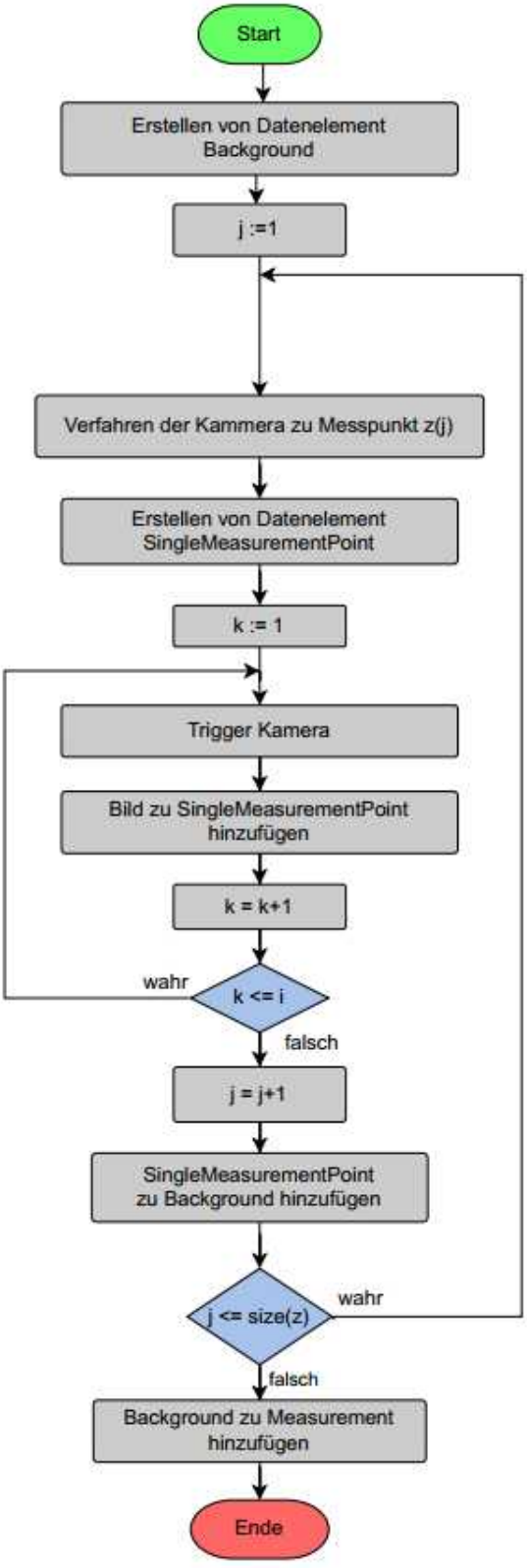


Abbildung 4.10: Ablaufplan Hintergrundmessung

4.2.3 Auswertung

Mit dem Messprogramm aufgenommenen Messdaten werden nach einem abgeschlossenen Messzyklus direkt an den Auswertungstab in der GUI übergeben und angezeigt (siehe Abb.4.11). Bevor die

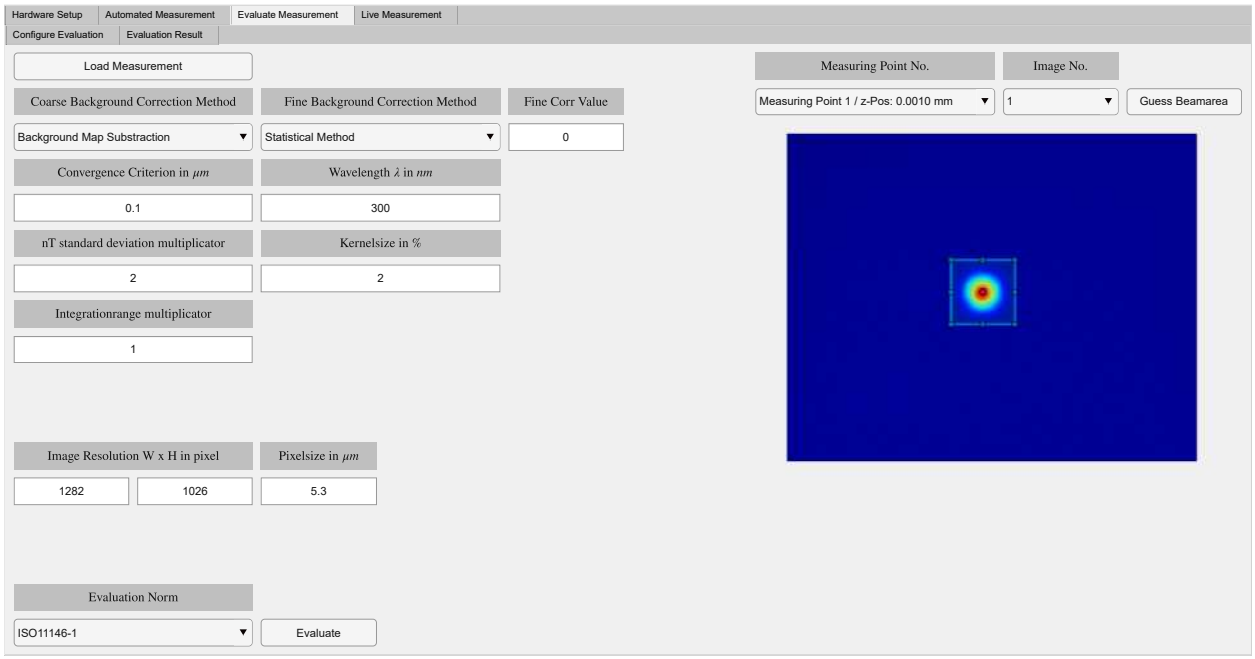


Abbildung 4.11: Konfiguration der Auswertungsparameter

Auswertung gestartet werden kann, müssen zuerst die Auswertungsparameter eingestellt werden. Die notwendigen Parameter können in Hintergrundkorrekturparameter und ISO-11146 Parameter unterteilt werden. Die Hintergrundkorrekturparameter umfassen alle Parameter die für die Hinter-

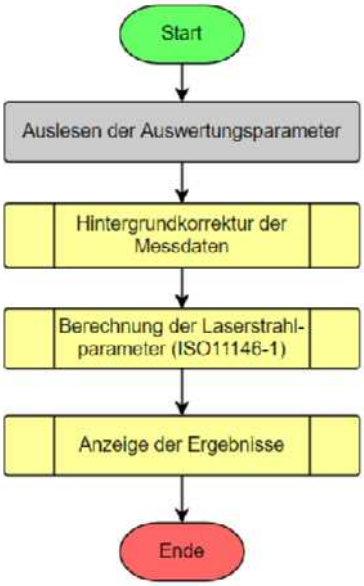


Abbildung 4.12: Ablaufplan der Auswertung

grundkorrektur notwendig sind (Grobkorrekturmethode, Feinkorrekturmethode, n_T und Kernelgröße) und die ISO 11146 Parameter. Davon all jene, die zur Bestimmung der Laserstrahlparameter

notwendig sind (Wellenlänge, Integrationsbereichsmultiplikator, Konvergenzkriterium und Schätzung des Integrationsbereichs).

Der Ablauf der Auswertung ist in Abb.4.12 dargestellt. Zuerst werden die Auswertungsparameter aus der GUI ausgelesen. Anschließend erfolgt die Hintergrundkorrektur (siehe Abb. 4.15) eines Messdatensatzes in der Form eines *MeasurementData* Objekts. Zu Beginn wird ein neues *MeasurementData* Objekt erstellt, indem die korrigierten Messdaten abgespeichert werden. Danach wird in einer Schleife über die Messpunkte iteriert, wobei für jeden *SingleMeasurementPoint* ein neues Objekt vom selben Typ erstellt wird. In einer zweiten Schleife werden alle SingleMeasurement Objekte des *SingleMeasurementPoint* hintergrundkorrigiert und dem neuen *SingleMeasurementPoint* hinzugefügt. Die korrigierten *SingleMeasurementPoint* Objekte werden dann dem *MeasurementData* Objekt hinzugefügt. Der Einfluss und Ablauf der Hintergrundkorrektur eines *SingleMeasuerement* Objekts (Einzelbild) wurde in Kap. 2.1 beschrieben. Sind die Messdaten hintergrundkorrigiert können die Parameter wie in Kap. 2.5 beschrieben bestimmt werden.

In Abb. 4.14 ist der Ablauf der Parameterbestimmung dargestellt. Sehr ähnlich zur Hintergrundkorrektur wird über die *SingleMeasurementPoint* Objekte und die SingleMeasurement Objekte iteriert. Die Parameter werden zuerst in der innersten Schleife für die *SingleMeasurement* Objekte, in der äußeren Schleife für die *SingleMeasurementPoint* Objekte und am Ende für das *MeasurementData* Objekt bestimmt. Die ausgewerteten Daten werden dann in der GUI angezeigt (siehe Abb. 4.13). Gegliedert ist die Anzeige der ausgewerteten Messdaten in drei Bereiche. Einem in dem die Parameter für die gesamte Messung angezeigt werden (links), einem für die gemittelten Messdaten eines Messpunktes (Mitte) und einem in dem die einzelnen Messungen und Parameter angezeigt werden (rechts). Im rechten Bereich wird zusätzlich zu den ausgewerteten Parametern immer auch das Bild des Strahlprofil angezeigt. Das Farbschema und die Limits für die Farbskalierung können darunter angepasst werden. Im linken Bereich befinden sich zusätzlich zu den Parametern die Plotoptionen der Messdaten.

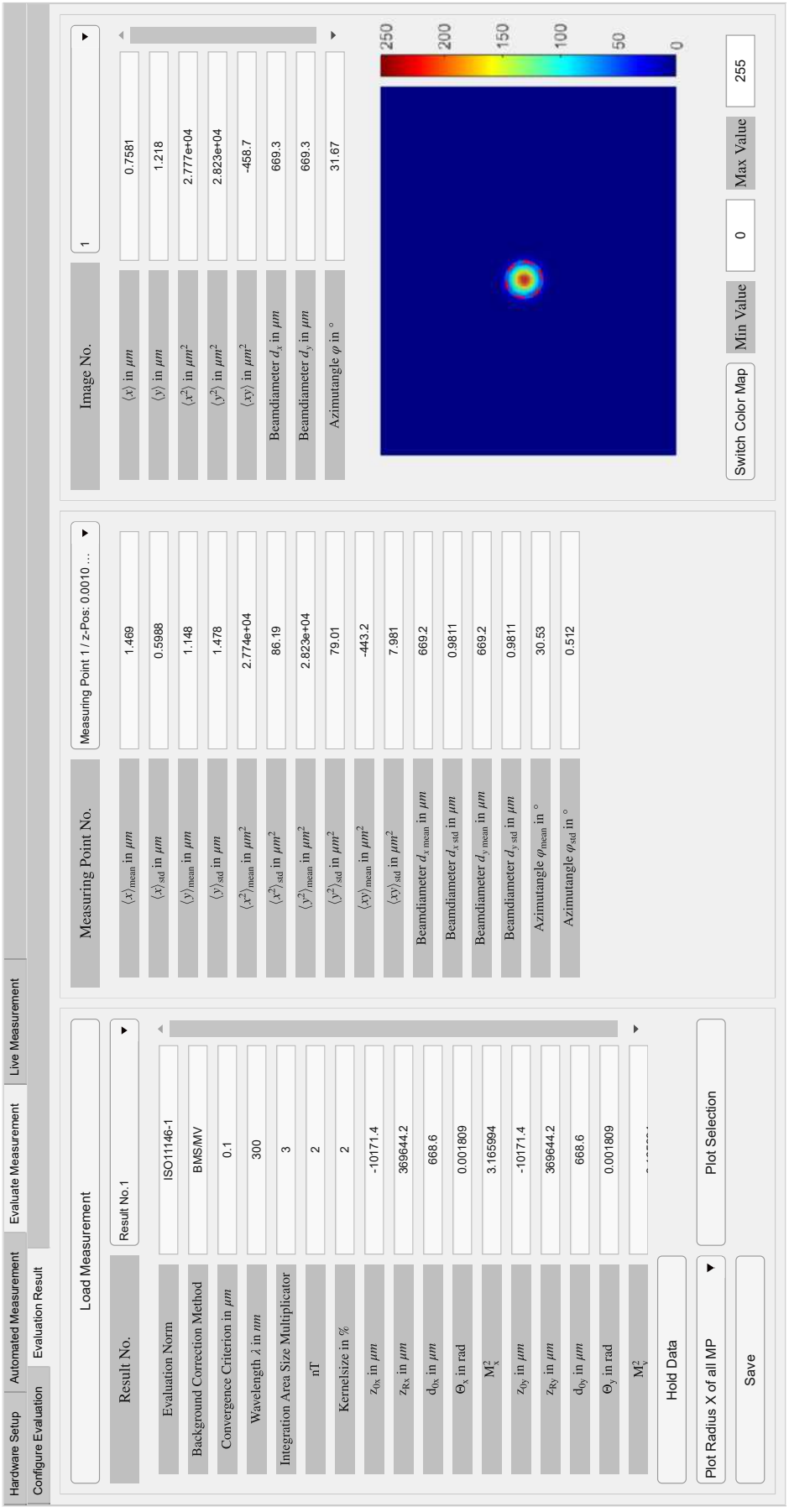


Abbildung 4.13: Darstellung der Ergebnisse mit Farbschema jet

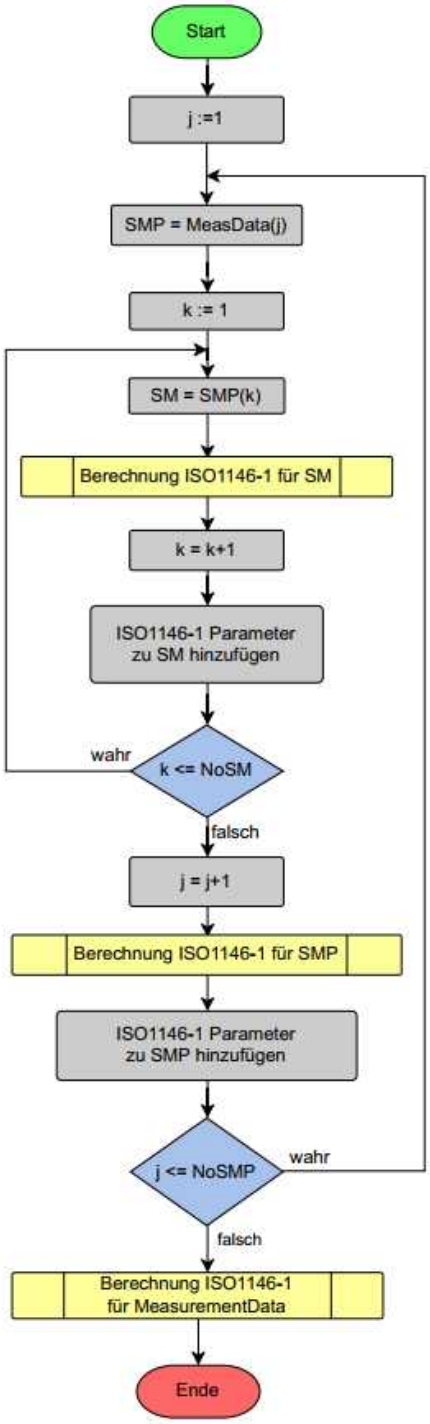


Abbildung 4.14: Ablaufplan Parameterberechnung

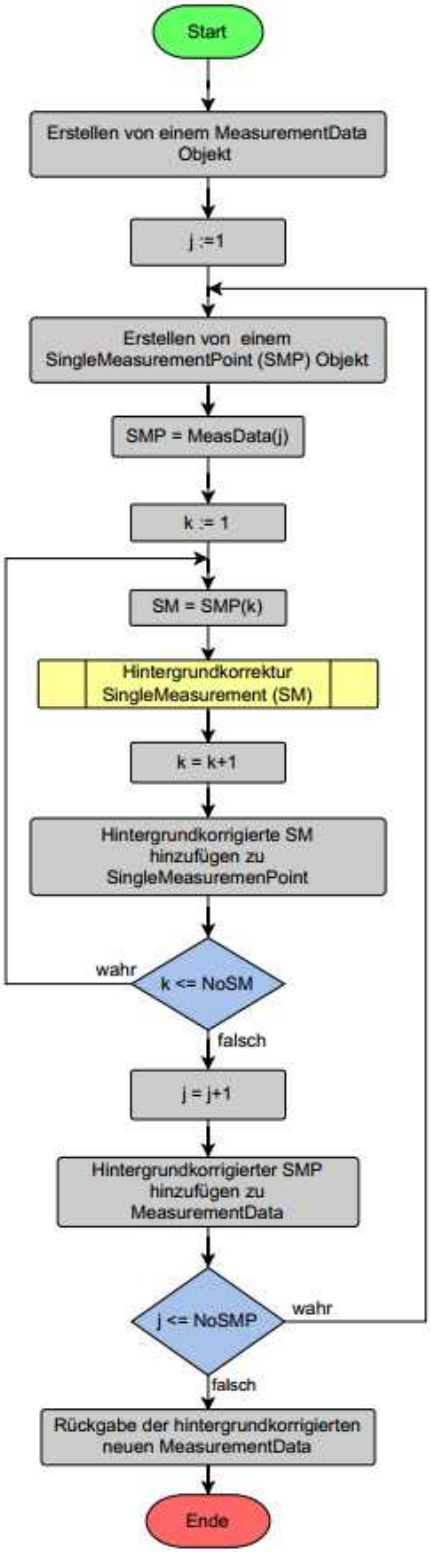


Abbildung 4.15: Ablaufplan Hintergrundkorrektur

4.3 Livemessung

Der Grundaufbau und die notwendigen Parameter der Livemessung sind identisch zur Auswertung (siehe Abb. 4.17). Im Gegensatz zur Auswertung steht hier aber noch kein Messdatensatz und damit auch keine Hintergrunddaten zur Verfügung. Daher ist es notwendig vor der Livemessung selbst Hintergrunddaten am gewünschten Messpunkt aufzunehmen. Die Anzahl der Bilder die vom Hintergrund gemacht werden ist frei wählbar, aber es sollten nie weniger als 15 Bilder eingestellt werden. Erst wenn Hintergrunddaten vorhanden sind, der initiale Integrationsbereich geschätzt und die Auswertungsparameter gesetzt wurden, kann die Livemessung gestartet werden. Das Ergebnis eines Mess- und Auswertungszyklus wird dann im rechten unteren Bereich (siehe Abb. 4.17) angezeigt. Angezeigt werden sowohl das Strahlprofil, als auch die Durchmesser in den Hauptachsenrichtungen und der dazugehörige Azimutwinkel. Das Ergebnis wird periodisch aktualisiert, wobei ein neues Bild erst aufgenommen wird sobald das aktuelle Bild fertig ausgewertet und der *live_toggle* „1“ gesetzt ist.

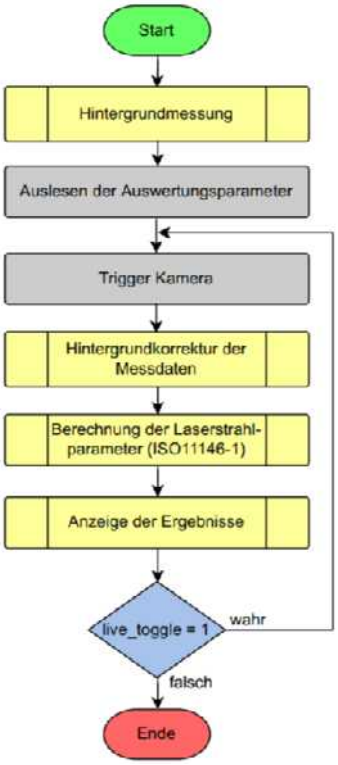


Abbildung 4.16: Ablaufplan Livemessung

Abbildung 4.17: Livemessung

5 Messergebnisse

Zusammen mit Messaufbau 3 und dem erstellten Messprogramm wurden zwei Messungen am FSL zu verschiedenen Zeitpunkten durchgeführt. Zum Zeitpunkt der ersten Messung war der im FSL verbaute Pumplaser schon weit degradiert und nur durch sehr hohe Stromzufuhr betreibbar, weshalb der Pumplaser zeitnah nach der Messung ausgetauscht wurde.

Tabelle 5.1: Messparameter

Parameter	**Vor Tausch**	**Nach Tausch**
Temperatur	20.7 °C	20.7 °C
Luftfeuchtigkeit	47.0 %	47.4 %
Anzahl der Messpunkte	21	21
Anzahl der Messungen pro Messpunkt	20	20
Messbereich	23 mm	20 mm
Belichtungszeit	2 ms	2 ms

Tabelle 5.2: Zusammenfassung der Messergebnisse

	Vor Tausch	**Nach Tausch**
Hintergrundkorrektur grob	Backgroundmap-substraction	Backgroundmap-substraction
Hintergrundkorrektur fein	Manual Value: 2	Manual Value: 2.2
Konvergenzkriterium in μm	0.1	0.1
Wellenlänge	800 nm	800 nm
Integrationsbereichsmultiplikator	3	3
Rayleighlänge z_{Rx}	9.500 mm	14.80 mm
Ort der Strahltaillie z_{0x}	11.217 mm	6.46 mm
Strahldurchmesser d_{0x}	169.2 μm	215.2 μm
Divergenzwinkel Θ_x (Vollwinkel)	17.8 mrad	14.5 mrad
Beugungsmaßzahl M_x^2	2.96	3.07
Rayleighlänge z_{Ry}	5.523 mm	4.220 mm
Ort der Strahltaillie z_{0y}	9.724 mm	6.44 mm
Strahldurchmesser d_{0y}	106.1 μm	67.6 μm
Divergenzwinkel Θ_y (Vollwinkel)	19.2 mrad	16.0 mrad
Beugungsmaßzahl M_y^2	2	1.06
Azimuthwinkel ϕ	0.67 deg	−1.09 deg
Beugungsmaßzahl M_{eff}^2	2.43	1.80

Die zweite Messung wurde durchgeführt nachdem der Pumplaser getauscht und alle optischen Elemente des FSL neu justiert waren. Für beide Messungen wurden 21 Messpunkte mit jeweils

20 Bildern pro Messpunkt aufgenommen. Die Messpunkte wurden auf den maximal zugänglichen Messbereich des Linearschlitten verteilt. Im Vergleich zu vor dem Pumplasertausch ist der zugängliche Messbereich kleiner geworden, da zwischen den Messungen ein Endschalter montiert wurde. Die Belichtungszeit der Kamera wurde auf 2 ms eingestellt, weil der Laser mit einer Wiederholrate von 1 kHz betrieben wird und damit sichergestellt wurde, dass pro Aufnahme zumindest ein Laserpuls detektiert werden konnte.

5.1 Messergebnisse vor dem Pumplasertausch

Abb. 5.1 und Abb. 5.2 zeigen den Verlauf der Strahlradien in Ausbreitungsrichtung des Laserstrahls. Auf der x-Achse wurde die z-Koordinate des Messpunkts ausgehend vom Nullpunkt des Linearschlittens entgegen der Ausbreitungsrichtung des Laserstrahls und auf der y-Achse der Strahlradius $R_{x,y}$ aufgetragen. Beide Verläufe waren erwartungsgemäß hyperbolisch, wobei der Ort des kleinsten Strahlradius sich in x- und y-Richtung voneinander unterschied. Der Strahlradius R_x hatte sein Minimum von 169.2 μm bei 11.217 mm und in y-Richtung 106.1 μm bei 9.724 mm.

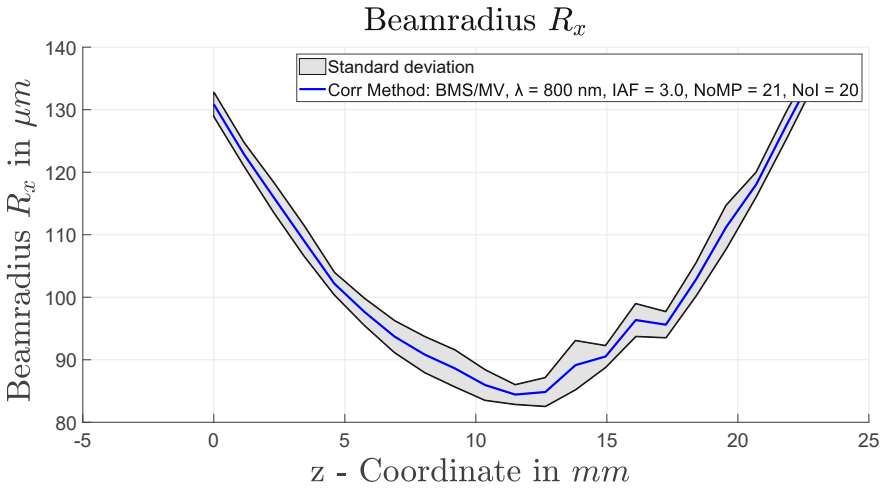


Abbildung 5.1: Laserstahlradius R_x in Ausbreitungsrichtung vor dem Tausch

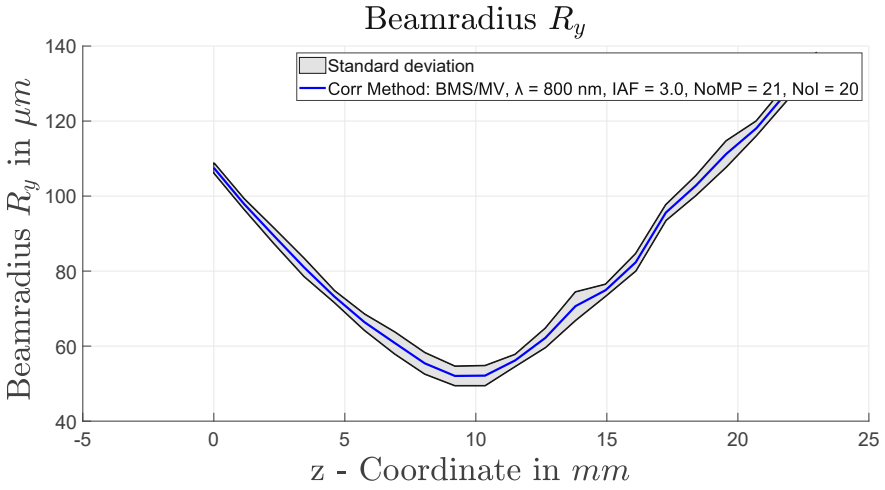


Abbildung 5.2: Laserstahlradius R_y in Ausbreitungsrichtung vor Tausch

Abb .5.3 und Abb. 5.4 zeigen das Strahlprofil des nächstliegenden Messpunktes zur Strahltaille in Z-Richtung. Klar erkennbar ist, dass das Strahlprofil innerhalb eines elliptischen Bereichs eine konstante Intensität ähnlich einem "Top Hat" Strahlprofil hatte. Der FSL hat aber eigentlich ein Gaußverteiltes Strahlprofil. Der Unterschied zwischen dem gemessenen Strahlprofil und dem erwarteten Strahlprofil kommt dadurch zustande, dass der Laserstrahl nicht ausreichend abgeschwächt wurde, wodurch es zur Sättigung der Kamerapixel kam.

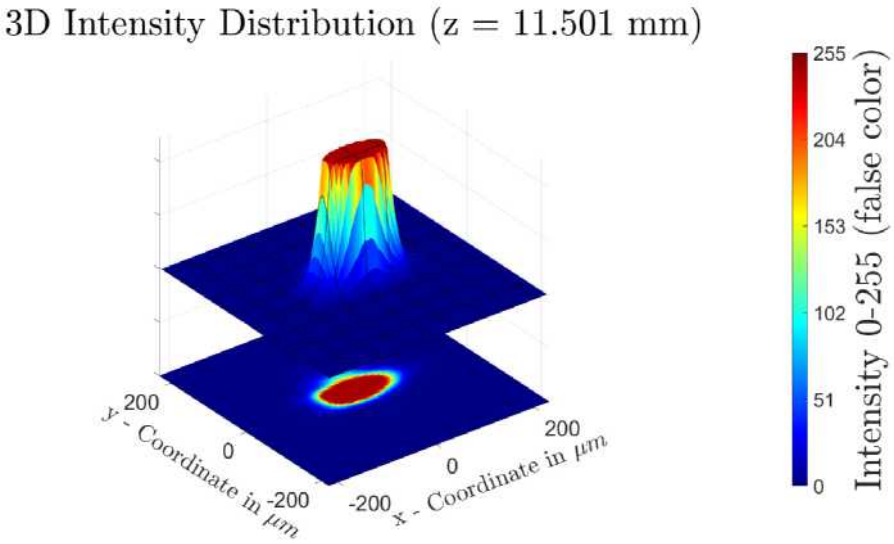


Abbildung 5.3: 3D Intensitätsverteilung vor dem Tausch

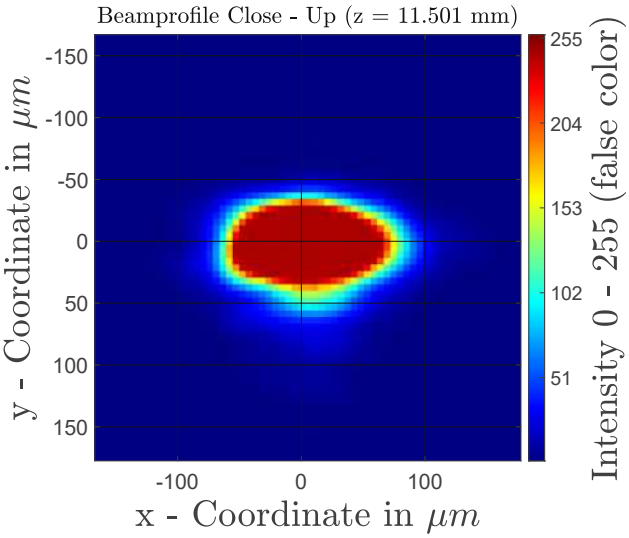


Abbildung 5.4: Nahaufnahme des Laserstrahlprofils vor dem Tausch

Durch die Übersättigung der Kamerapixel kam es zu einer lokalen Überbelichtung, dem sogenannten "Blooming" Effekt. Zustande kommt dieser Effekt, wenn ein Pixel mehr Ladung in Form von Photonen ausgesetzt ist als er aufnehmen kann. Das führt in weiterer Folge dazu, dass der Pixel die überschüssige Ladung an umliegende Pixel abgibt, wodurch überbelichtete Bereiche im aufgenommenen Bild entstehen. Das gemessene Strahlprofil wird dadurch künstlich vergrößert.

In Abb. 5.5 und Abb. 5.6 wird die Intensitätsverteilungen entlang der Hauptachsen des Strahlprofils dargestellt. Auf der x-Achse wurde der Strahlradius $R_{x,y}$ und in y-Achse die Intensität als Wert im Bereich von $0 - 255$ aufgetragen. Zusätzlich wurden der FHWM Durchmesser (magenta strichlierte Linien), der $D4\sigma$ Durchmesser (schwarz strichlierte Linien) und eine an die Intensitätsverteilung angepasste ideale Gaußkurve (rot strichlierte Linie) eingezeichnet.

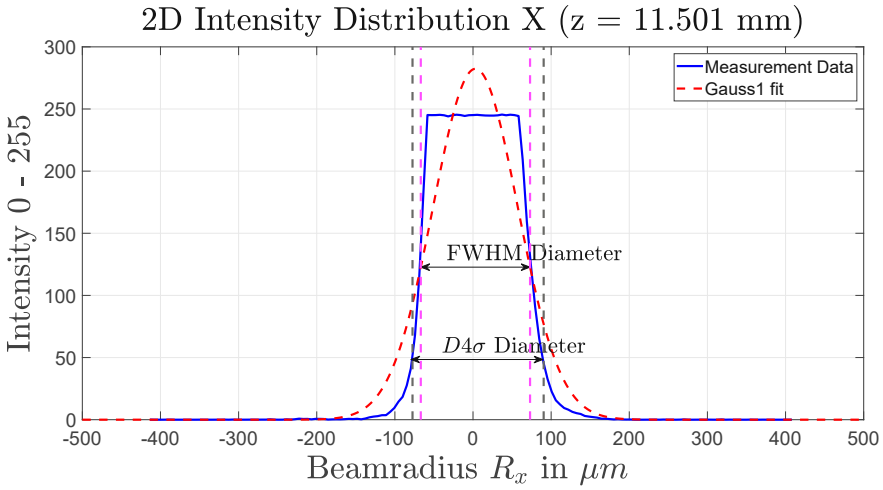


Abbildung 5.5: 2D Intensitätsverteilung in x-Richtung vor dem Tausch

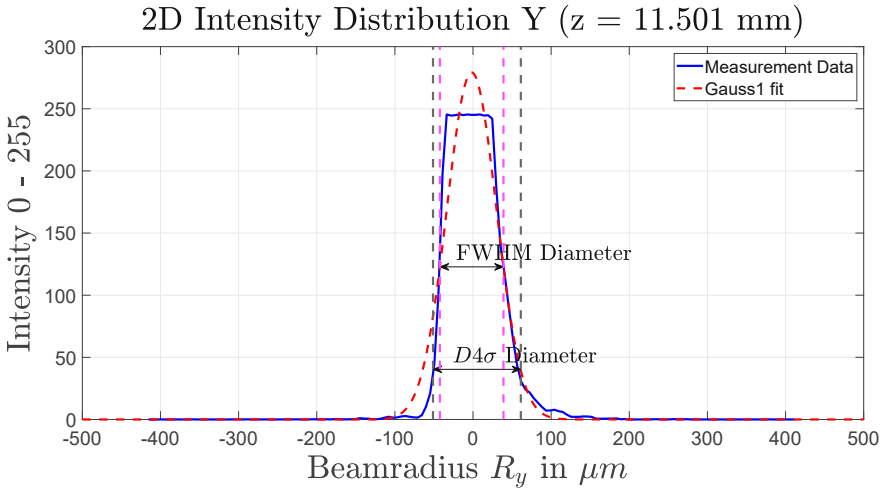


Abbildung 5.6: 2D Intensitätsverteilung im Fokuspunkt in y-Richtung vor dem Tausch

Die 2D FFT vom Strahlprofil (siehe Abb. 5.7) zeigt die logarithmisch skalierten Amplituden der Ortsfrequenzen. Diese Darstellung hat den Vorteil, dass nur relevante Amplituden im Spektrum sichtbar werden. Es zeigt den erwarteten niederfrequenten Anteil im Zentrum umgeben von symmetrischen Ringen deren Amplitude mit steigender Frequenz sinkt. Diese Ringe sind ein Produkt der überbelichteten Pixel und der FFT. Die FFT zerlegt das Intensitätsprofil in seine Frequenzanteile. Ist das zu analysierende Signal ein Rechtecksignal, was recht gut dem gemessenen Intensitätsprofil entspricht, siehe Abb. 5.5 oder Abb. 5.6, ergeben sich zusätzliche Frequenzanteile die notwendig sind um das Signal nachzubauen. Die Ortsfrequenzanteile in den Hauptachsenrichtungen sind in Abb. 5.8 dargestellt. Die Interpretation ist dabei identisch zur 2D FFT. Es gibt niederfrequente Anteile und die durch die Pixelsättigung bedingten zusätzlichen Frequenzanteile.

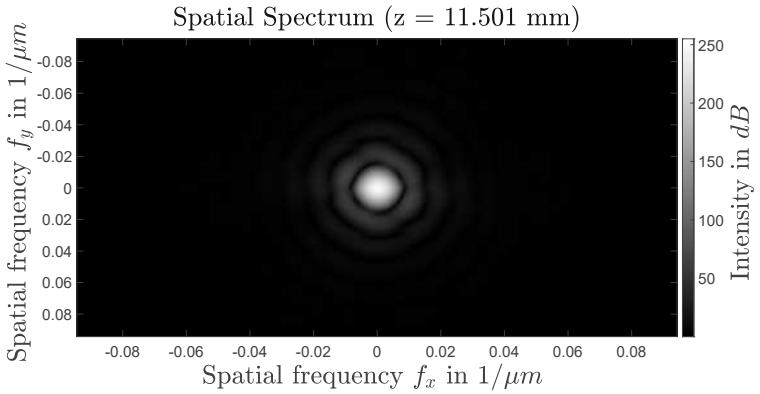


Abbildung 5.7: 2D FFT des Laserstrahlprofils vor dem Tausch

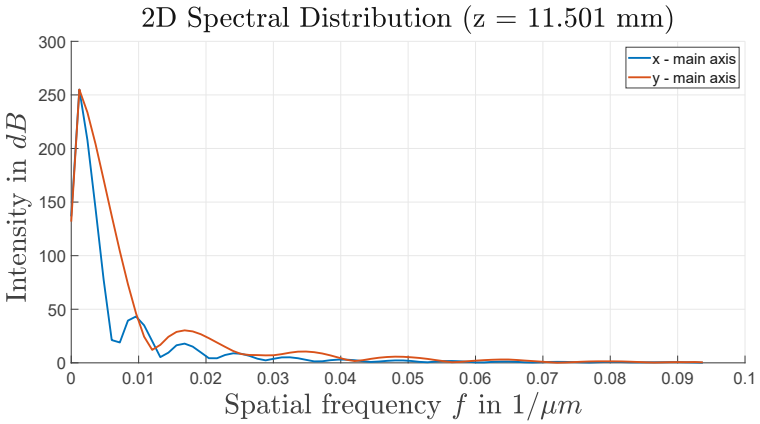


Abbildung 5.8: 1D FFT des Laserstrahlprofils vor dem Tausch in Hauptachsenrichtungen

5.2 Messergebnisse nach dem Pumplasertausch

Um die Pixelsättigung (siehe S. 51) zu verhindern wurde ein zusätzliches Filterelement zur Abschwächung des Laserstrahls verbaut. Nach dem Tausch des Pumplasers sind beide Verläufe ebenfalls hyperbolisch, wobei der Ort des kleinsten Strahlradius in x- und y-Richtung fast identisch war.

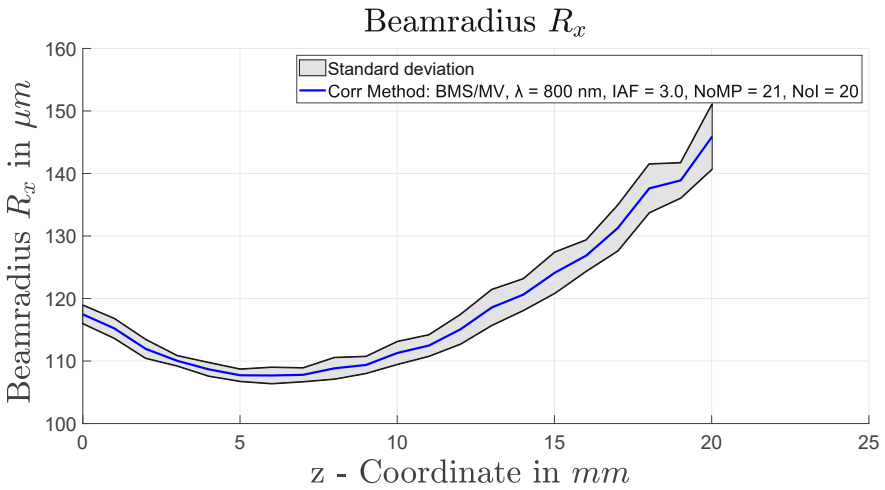


Abbildung 5.9: Laserstahlradius in x-Richtung nach dem Tausch

Der Strahlradius R_x hatte sein Minimum von 215.2 μm bei 6.46 mm und in y-Richtung 67.6 μm bei 6.44 mm.

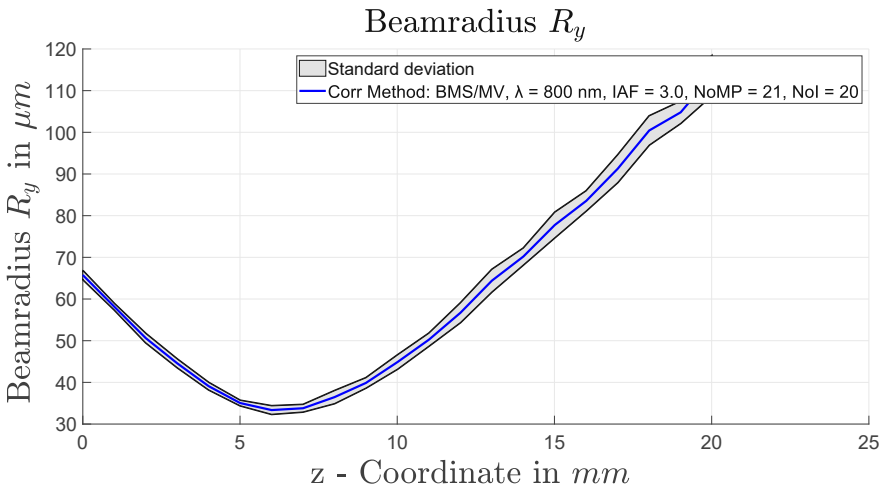


Abbildung 5.10: Laserstahlradius in y-Richtung nach dem Tausch

Die Strahlprofile (siehe Abb. 5.11 und Abb. 5.12) sind weiterhin elliptisch, aber nun hat das Strahlprofil im Zentrum ein klares Maximum, dass nach außen hin abnimmt.

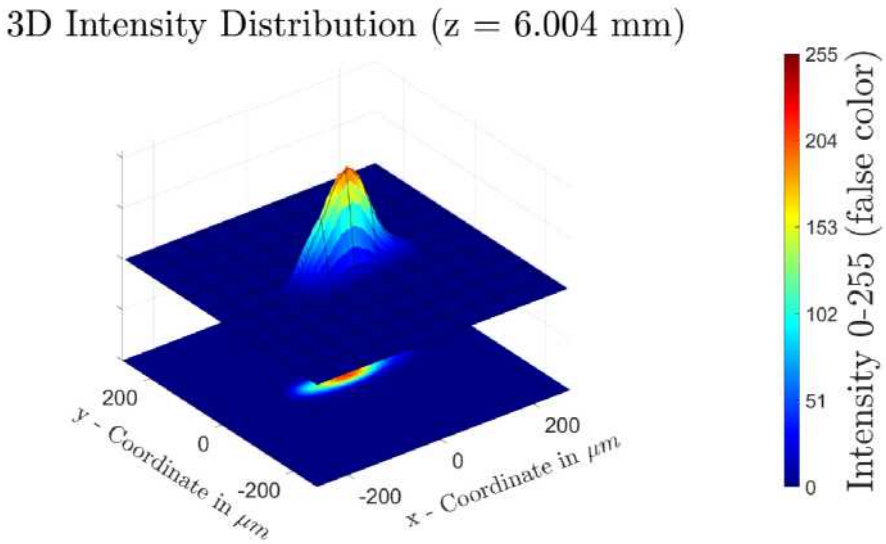


Abbildung 5.11: 3D Intensitätsverteilung nach dem Tausch

Mögliche Gründe für das elliptische Strahlprofil:

- Das Zentrum der höchsten Intensität des Strahlprofils ist elliptisch und wurde erst durch Abschwächung sichtbar.
- Da der Laserstrahl des FSL aus einem breiten Spektrum an Wellenlängen besteht, die verwendeten ND-Filter allerdings keine konstante Transmission über diesen Bereich besitzen, könnte durch die inhomogene Abschwächung das Strahlprofil beeinflusst worden sein.
- Die Messebene der Kamera könnte nicht orthogonal auf die Ausbreitungsrichtung des Laserstrahls stehen, wodurch das Strahlprofil eine elliptische Form bekommt.

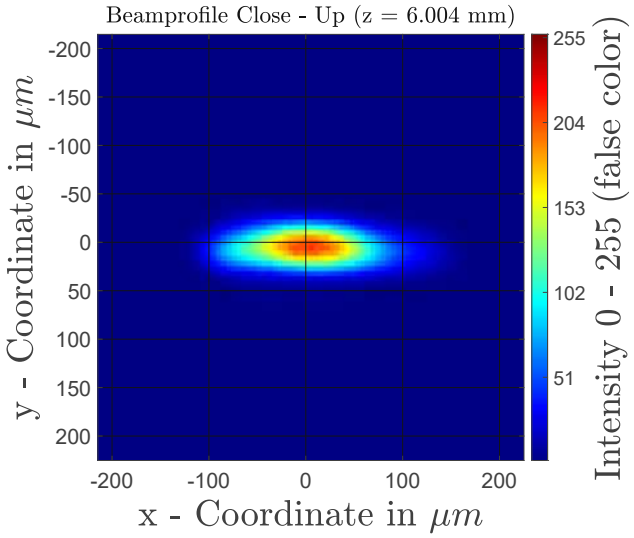


Abbildung 5.12: Nahaufnahme des Laserstrahlprofils nach dem Tausch

Das unfokussierte Strahlprofil wurde auf eine weiße Oberfläche projiziert. Dabei konnte mit freiem Auge festgestellt werden, dass der Rohstrahl ein gaußsches Profil besitzt. Das würde dafür sprechen, dass durch die signifikante Abschwächung des Laserstrahls nurmehr der Bereich mit der höchsten Intensität messbar bleibt. Ob und welchen Einfluss die inhomogenen Transmissionseigenschaften

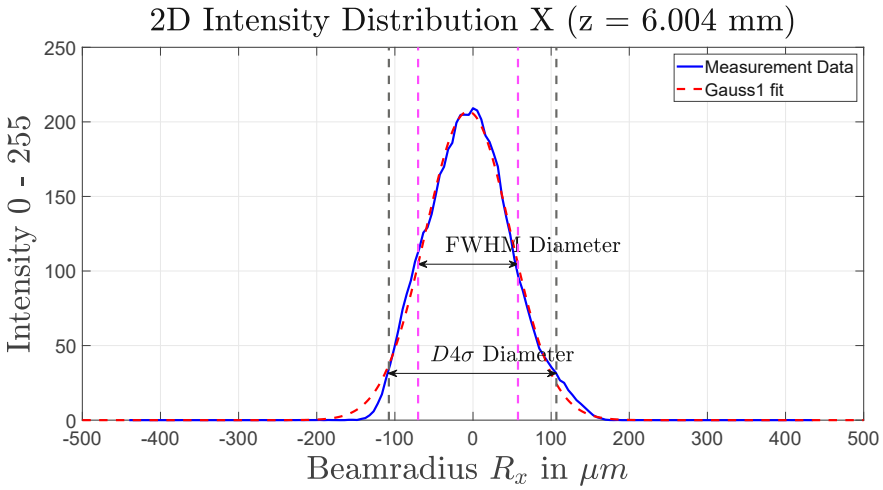


Abbildung 5.13: 2D Intensitätsverteilung nach dem Tausch

haben, ließe sich nur durch Kenntnis der im inhomogenen räumlichen Verteilung der Bandbreite im Strahlquerschnitt feststellen. Die Orthogonalität der Messebene der Kamera auf die Ausbreitungsrichtung des Laserstrahls wurde während dem Einrichtprozess des Messsystems geprüft. Bevor aber die Winkelverschiebung der Messebene überprüft werden kann muss zuerst überprüft werden, ob der Laserstrahl koaxial zur Verfahrachse des Schlittens ist. Geprüft wurde es, indem der Laserstrahl zuerst auf das Zentrum des Messsensors ausgerichtet wurde und anschließend die Kamera über den Linearschlitten in Ausbreitungsrichtung verfahren wurde. Das gemessene Strahlprofil und dessen Position auf der Kamera wurde währenddessen beobachtet, um zu überprüfen ob der Spot über den Sensor wandert. Stellt man bei verfahren des Linearschlittens fest, dass das gemessene Strahlprofil nach oben oder unten wandert, deutet das darauf hin, dass eine Winkelabweichung zur Horizontalen vorliegt. Wandert er nach links oder rechts deutet das darauf hin, dass der Linearschlitten eine Win-

kelverschiebung zur Ausbreitungsrichtung des Laserstrahl aufweist. Wenn man diese beiden Fälle ausgeschlossen hat, kann mithilfe des verstellbaren Spiegels der Laserstrahl auf das Zentrum der Kamera ausgerichtet werden. Die 2D Intensitätsverteilungen in Hauptachsenrichtungen (siehe Abb.

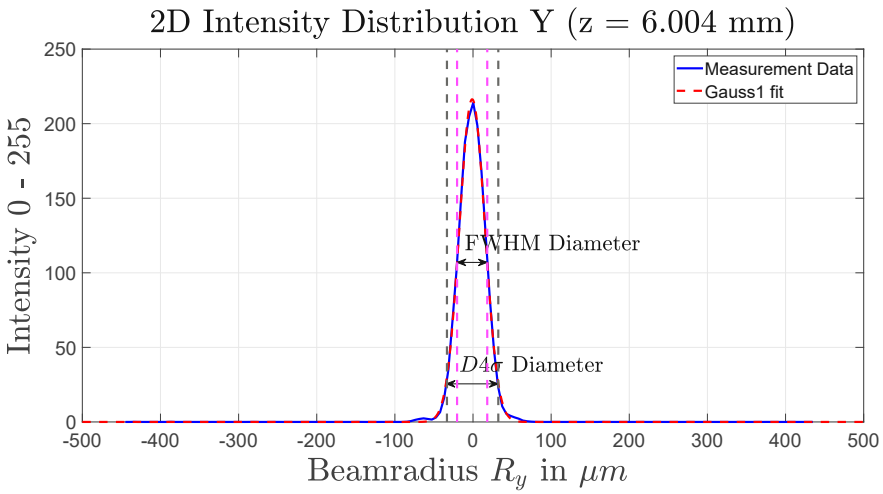


Abbildung 5.14: 2D Intensitätsverteilung nach dem Tausch

5.13 und Abb. 5.14) decken sich gut mit einer idealen Gaußverteilung. Es gibt ein klares Maximum und die Randbereiche der Verteilung sind nicht abgeschnitten (erkennbar durch den kontinuierlichen Verlauf gegen 0), wodurch Überkompensation durch die Hintergrundkorrektur ausgeschlossen werden kann. In der 2D FFT (siehe Abb. 5.15) sind hauptsächlich niederfrequente Anteile vorhanden.

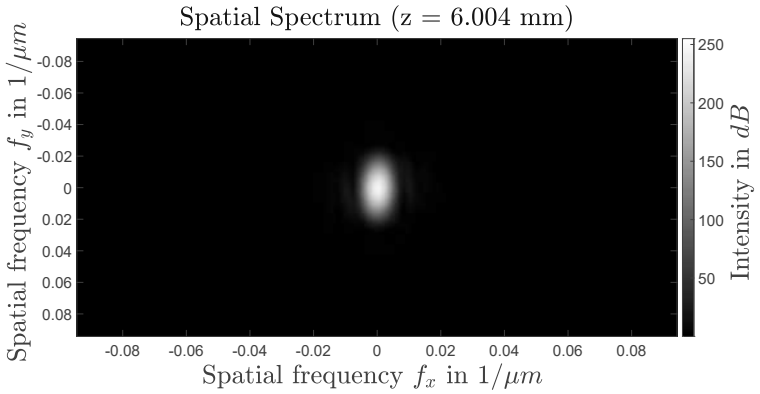


Abbildung 5.15: 2D FFT des Laserstrahlprofils nach dem Wechsel

Die vor dem Tausch des Pumplasers beobachteten periodischen Ringe, aufgrund der Pixelsättigung um den niederfrequenten zentralen Teil des Spektrums, sind hier nicht mehr vorhanden. Die FFT der 2D Intensitätsverteilungen (siehe Abb. 5.16)zeigt in x-Richtung ein schmaleres Frequenzband als in y-Richtung.

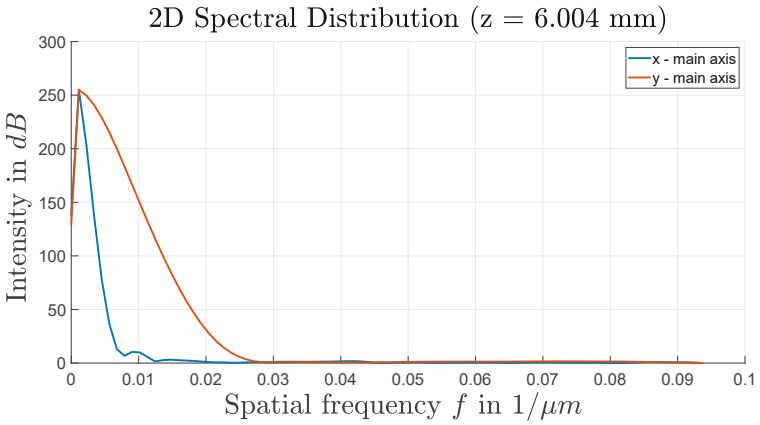


Abbildung 5.16: 1D FFT des Laserstrahlprofils nach dem Tausch in Hauptachsenrichtungen

5.3 Vergleich theoretische und reale Fokusdurchmesser

Jede optische Komponente im Aufbau eines Lasers sind aufgrund von Imperfektionen fehlerbehaftet. Es lässt sich daraus schließen, dass auch die theoretisch erreichbaren Werte von den realen abweichen. Das gilt auch für den Fokusdurchmesser der durch ein fokussierendes optisches Element erreicht werden kann. Der Durchmesser eines fokussierten Strahls [2] kann mit der Brennweite f, der Wellenlänge λ, der Strahlgüte $k = 1/M^2$ und dem Strahltaillenradius w_L am fokussierenden Element über

$$d_f = \frac{2 \cdot f \cdot \lambda}{\pi \cdot k \cdot w_L} \tag{5.1}$$

berechnet werden. Verglichen wurden die theoretischen und gemessenen Werte bei einer Brennweite von 100 mm und 500 mm. Für die 100 mm Brennweite wurden die Strahlabmessungen des fokussierten Strahls visuell anhand eines Einschusses mikroskopisch bestimmt (siehe Abb. 5.17). Für die Brennweite von 500 mm konnten die zuvor über das Messsystem bestimmten Strahlabmes-

Abbildung 5.17: Beispiel eines Einschusslochs bei einer Brennweite von 100 mm

sungen herangezogen werden. In Tab. 5.3 sind die Ergebnisse zusammengefasst. Bestimmt wurden die theoretischen Strahldurchmesser für die aus den Messungen bestimmten effektiven Beugungsmaßzahlen und der vom Hersteller angegebenen Beugungsmaßzahl. Es fällt direkt auf, dass sowohl die vermessenen Brandlöcher als auch die mit dem Messsystem gemessenen Strahlprofile elliptisch

sind. Die Differenz der gemessenen Strahlabmessungen zu den theoretischen Abmessungen beträgt vor dem Tausch bei einer Brennweite von 100 mm in x-Richtung 16.8 µm und nach dem Tausch 30.93 µm. In y-Richtung beträgt die Differenz vor dem Tausch 4.9 µm bzw. nach dem Tausch 2.6 µm. Bei der 500 mm Brennweite beträgt die Differenz vor dem Tausch in x-Richtung 76.7 µm und nach dem Tausch 154.1 µm. In y-Richtung beträgt die Differenz vor dem Tausch 13.6 µm und nach dem Tausch 6.6 µm.

Tabelle 5.3: Vergleich theoretische und reale Fokusdurchmesser

Parameter	**Hersteller**	**Vor Tausch**	**Nach Tausch**
Beugungsmaßzahl M^2	2	2.43	1.80
Strahlradius $1/e^2$ w_L	7.5 mm	7.5 mm	7.5 mm
theoretischer Fokusdurchmesser $1/e^2$ d @100mm	13.6 µm	16.5 µm	12.2 µm
gemessener Fokusdurchmesser $1/e^2$ d_x @100mm	-	33.3 µm	43.13 µm
Differenz Δd_x @100mm	-	16.8 µm	30.93 µm
gemessener Fokusdurchmesser $1/e^2$ d_y @100mm	-	21.3 µm	14.8 µm
Differenz Δd_y @100mm	-	4.9 µm	2.6 µm
theoretischer Fokusdurchmesser $1/e^2$ d @500mm	67.9 µm	92.5 µm	61.1 µm
gemessener Fokusdurchmesser $1/e^2$ d_x @500mm	-	169.2 µm	215.2 µm
Differenz Δd_x @500mm	-	76.7 µm	154.4 µm
gemessener Fokusdurchmesser $1/e^2$ d_y @500mm	-	106.1 µm	67.6 µm
Differenz Δd_y @500mm	-	13.6 µm	6.6 µm
Verhältnis $d_x@500mm/d_x@100mm$	-	5.1	4.99
Verhältnis $d_y@500mm/d_y@100mm$	-	4.98	4.35

Die Brennweiten der Messungen stehen in einem Verhältnis von 5 zueinander. Bildet man die Verhältnisse der gemessenen Durchmesser bei 500 mm und 100 mm ergibt sich vor dem Tausch des Pumplasers in x-Richtung ein Verhältnis von 4.98 und in y-Richtung 4.35. Nach dem Pumplasertausch ist das Verhältnis in x-Richtung 4.98 und in y-Richtung 4.35. Das spricht dafür, dass die mit dem Messsystem gemessenen Strahlprofile und damit einhergehend die berechneten Strahleigenschaften den tatsächlichen Eigenschaften des Laserstrahls entsprechen.

6 Zusammenfassung und Ausblick

Der Messaufbau und das Messprogramm die im Zuge dieser Arbeit gebaut und programmiert wurden, sind zwar für den Einsatz am Ti:Sa-Femtosekundenlaser entwickelt, können bei Bedarf aber auch für alternative Laserquellen genutzt werden, solange sie ein stigmatisches oder leicht astigmatisches Strahlprofil besitzen. Für die Messung von alternativen Lasern ist zu beachten, dass die verwendete Hardware an den Leistungs- und Frequenzbereich des zu vermessenden Lasers angepasst werden muss. Ansonsten erlaubt der Aufbau des Messsystems auf einer vom Laser unabhängigen Plattform eine einfache Portabilität für die Montage bei anderen Laserquellen. Für astigmatische Laserstrahlen wäre die ISO 11146-2 anzuwenden. Die Bestimmung der dafür notwendigen Parameter würde zusätzlich zur bestehenden Messung, zwei weitere Messungen voraussetzen, für die der Messaufbau und das Messprogramm umgebaut bzw. erweitert werden müssten. Der bestehende Aufbau liefert acht der zehn Momente zweiter Ordnung die für die Bestimmung der Parameter von astigmatischen Strahlen notwendig sind. Um die letzten zwei Momente zweiter Ordnung zu bestimmen, müsste die bestehende Linse nach einem Messzyklus durch eine Zylinderlinse ersetzt und der Messzyklus insgesamt zweimal wiederholt werden (einmal mit vertikal montierter und einmal mit horizontal montierter Zylinderlinse). In weiterer Folge muss auch das Messprogramm um neue Datenelemente und Funktionen erweitert werden, um die zusätzlichen Anforderungen an den Messzyklus und der Auswertung zu erfüllen. Nach der Bestimmung aller Momente zweiter Ordnung mit dem erweiterten Messaufbau und Messprogramm bestimmt, kann die Strahlmatrix gebildet und damit die Strahleigenschaften berechnet werden.

Im Verlauf dieser Arbeit wurden zwei unterschiedliche Ansätze zur Abschwächung der Laserstrahlen für die anschließende Messung der Strahleigenschaften mittels NIR Kamera untersucht. In Messaufbau Version 1 wurde versucht den Laserstrahl durch Reflexion an Prismen abzuschwächen. Im Zuge der ersten Tests wurde festgestellt, dass "Ghosting" Strahlen resultierend aus sekundären Reflexionen in den Prismen zu Interferenzen mit dem Primärstrahl führten. Die Abschwächung des Laserstrahls mittels Reflexion hat einen Messtechnischen Vorteil gegenüber der Abschwächung mittels Transmission durch Filterelemente. Der Vorteil ist, dass die Reflexionseigenschaften von rechtwinkligen Glasprismen oder Glaswedges sehr konstant über einen breiten Wellenlängenbereich sind. Dadurch kann ein Einfluss der frequenzabhängigen Abschwächung auf das Strahlprofil ausgeschlossen werden. Der große Nachteil dieser Methode ist, dass für große Abschwächungen viele Stufen notwendig sind und zwischen den Stufen genug Abstand vorgesehen werden muss, um die "Ghosting" Strahlen aus dem Messsystem auszukoppeln. Verwendet man wie in Messaufbau Version 3 Filter zur Abschwächung, kann das Messsystem wesentlich kompakter gebaut werden, es muss jedoch die Frequenzabhängigkeit der Transmission durch die Filter berücksichtigt werden. Die Abschwächung des Laserstrahls ist vor allem für die Vermessung von Hochleistungslasern notwendig, da der auf die Kamera fokussierte Laserstrahl ansonsten entweder die Kamerapixel zerstören oder sättigen würde. Gesättigte Pixel führen zu einer künstlichen Vergrößerung des gemessenen Strahlprofil durch den "Blooming" Effekt. Es gibt weitere Anforderungen an das Strahlprofil und die Kamera die beachtet werden müssen. Der Durchmesser des Strahlprofils darf nicht größer sein als ein Drittel der verfügbaren Sensorfläche, da in der Norm festgelegt ist, dass der Auswertungsbereich (Integrationsbereich) dreimal so groß sein sollte wie die zu bestimmenden Strahlabmessungen. Weiters sollte der Strahldurchmesser laut Norm mit mindestens 20 Kamerapixel aufgelöst werden. Mit der in dieser Arbeit verwendeten Kamera betrug die Auflösung des Strahldurchmesser bei einer Brennweite von 500 mm 10 Kamerapixel. Um die Auflösung von 20 Kamerapixel zu erreichen

hätte man entweder eine Kamera mit höherer Auflösung verwenden müssen oder die Brennweite des fokussierenden Elements vergrößern. Aufgrund des beschränkten Platzes war die Brennweite allerdings bereits mit 500 mm am Limit des möglichen und eine andere Kamera stand nicht zur Verfügung. Für die Messung von stigmatischen Strahlen sind 20 Messpunkte im Bereich von zumindest zwei Rayleighlängen in beiden Richtungen ausgehend von der Strahltaille erforderlich um ein normkonformes Messergebnis zu erhalten. Daraus ergibt sich, dass der Linearschlitten einen Verfahrweg von mindestens 5 Rayleighlängen und eine passende Ganggenauigkeit haben muss.

Zusammenfassend gibt es viele Feinheiten die bei der Messung und Auswertung der räumlichen Strahleigenschaften eines Lasers beachtet werden müssen. Jede Komponente im Messaufbau kann das Messergebnis beeinflussen und muss daher sorgfältig ausgewählt werden um ein möglichst großes Spektrum an unterschiedlichen Lasern abdecken zu können. Als nächsten Schritt wäre es wünschenswert einen Messaufbau aufzubauen der den Laserstrahl mittels Reflexion abschwächt. Dann könnten die Vor- und Nachteile der beiden Ansätze zur Abschwächung direkt miteinander verglichen werden.

Abbildungsverzeichnis

Tabellenverzeichnis

7 Anhang

Anhang 1: Quellcode

7.1: Class Measurement

```
1  classdef Measurement < handle
2      %MEASUREMENT Class used to perform, store and analyze a
3      %Laserbeammeasurement.
4
5      properties(Access = public)
6          measurementData MeasurementData = MeasurementData();
7          backgroundData Background
8          measSettings MeasurementSettings;
9          results Result;
10     end
11
12     methods
13         function obj = Measurement(rayleighLength,timesReyleighLength,noMP,
               measPerPoint)
14             obj.measSettings = MeasurementSettings(rayleighLength,
```

```
                timesReyleighLength,noMP,measPerPoint);
            obj.backgroundData = Background();
            obj.measurementData = MeasurementData();
        end
        function reset(obj)
            obj.measurementData.reset();
            obj.backgroundData.reset();
        end
        function value = getMeasurementData(obj)
            % Get stored measurment data.
            value = obj.measurementData;
        end
        function value = getBackgroundData(obj)
            % Get stored background data.
            value = obj.backgroundData;
        end
        function value = getResults(obj)
            % Get stored measurment data.
            value = obj.results;
        end
        addResult(obj,value);

    end
end
```

7.2: Function addResult

```
function addResult(obj,value)
%addResult Adds new Result to the existing Measurement result dataset.
obj.results(end+1) = value;
end
```

7.3: Class MeasurementData

```
classdef MeasurementData < handle
    %MEASUREMENTDATA  Class containing the measurement data of all measurementpoints
        .

    properties (Access = public)
        measurementDataSet SingleMeasurementPoint;

        W_X_theta_X double;
        W_Y_theta_Y double;
        W_theta_X_squared double;
        W_theta_Y_squared double;
        W_theta_X_theta_Y double;


        fx cfit;
        z0x double;
```

```
        dSigma0x double;
        thetaSigmaX double;
        zRx double;
        MsquaredX double;

        fy cfit;
        z0y double;
        dSigma0y double;
        thetaSigmaY double;
        zRy double;
        MsquaredY double;

        zPos double;
        dWx double;
        dWxStd double;
        dWy double;
        dWyStd double;
    end

    methods
        %         function delete(obj)
        %             delete(obj.measurementDataSet);
        %         end
        function reset(obj)
            obj.measurementDataSet = SingleMeasurementPoint.empty;
        end
        function measurementData(obj)
            obj.measurementDataSet = SingleMeasurementPoint.empty;
        end
        function measurementDataSet = getMeasurementDataSet(obj)
            % Get stored measurment dataSet.
            measurementDataSet = obj.measurementDataSet;
        end
        function value = getZ0X(obj)
            % Get stored measurment dataSet.
            value = obj.z0x;
        end
        function value = getZ0Y(obj)
            % Get stored measurment dataSet.
            value = obj.z0y;
        end
        function value = getDSigma0x(obj)
            % Get stored measurment dataSet.
            value = obj.dSigma0x;
        end
        function value = getDSigma0y(obj)
            % Get stored measurment dataSet.
            value = obj.dSigma0y;
        end
```

```
        function value = getThetaSigmaX(obj)
            % Get stored measurment dataSet.
            value = obj.thetaSigmaX;
        end
        function value = getThetaSigmaY(obj)
            % Get stored measurment dataSet.
            value = obj.thetaSigmaY;
        end
        function value = getZrX(obj)
            % Get stored measurment dataSet.
            value = obj.zRx;
        end
        function value = getZrY(obj)
            % Get stored measurment dataSet.
            value = obj.zRy;
        end
        function value = getMsquaredX(obj)
            % Get stored measurment dataSet.
            value = obj.MsquaredX;
        end
        function value = getMsquaredY(obj)
            % Get stored measurment dataSet.
            value = obj.MsquaredY;
        end
        function value = getfxFit(obj)
            % Get stored measurment dataSet.
            value = obj.fx;
        end
        function value = getfyFit(obj)
            % Get stored measurment dataSet.
            value = obj.fy;
        end
        function value = getzPos(obj)
            % Get stored measurment dataSet.
            value = obj.zPos;
        end
        function value = getdWx(obj)
            % Get stored measurment dataSet.
            value = obj.dWx;
        end
        function value = getdWxStd(obj)
            % Get stored measurment dataSet.
            value = obj.dWxStd;
        end
        function value = getdWyStd(obj)
            % Get stored measurment dataSet.
            value = obj.dWyStd;
        end
        function value = getdWy(obj)
```

```
            % Get stored measurment dataSet.
            value = obj.dWy;
        end
        addSingleMeasurementPoint(obj,value);
        value = backgroundCorrectionMeasurementData(obj,coarseMethod,fineMethod,
            background,kernelSize,ntFactor,integAreaFactor,k,manualValue);
        %         value = correction(obj,method,backgroundData,varargin);
        value = iso11146Part1MeasurementData(obj,centerGuessX,centerGuessY,
            integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,lambda,resWidth
            ,resHeight,pixelsize);
        value = iso11146Part2MeasurementData(obj,centerGuessX,centerGuessY,
            integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,lambda);
    end
end
```

7.4: Function addSingleMeasurementPoint

```
function addSingleMeasurementPoint(obj,value)
%addSingleMeasurementPoint Add a element of type SingleMeasurementPoint to
%dataSet.
obj.measurementDataSet(end+1) = value;
end
```

7.5: Function backgroundCorrectionMeasurementData

```
function value = backgroundCorrectionMeasurementData(obj,coarseMethod,fineMethod,
    background,kernelSize,ntFactor,manualValue)
%backgroundCorrectionMeasurementData Initiates backgroundcorrection for the
%whole measurementdata.

%   Correct measurment data.
if not(isa( background, 'Background' ))
    error('Invalid Classtype of input parameter background.');
end
measurementDataSet = obj.getMeasurementDataSet();
[Useless,NoMP] = size(measurementDataSet);
value = MeasurementData();
for k = 1 : NoMP
    fprintf('Performing background correction for measuring point %i.\n',k);
    corrMeasPoint = measurementDataSet(k).backgroundCorrectionSingleMeasurementPoint
        (coarseMethod,fineMethod,background,kernelSize,ntFactor,k,manualValue);
    value.addSingleMeasurementPoint(corrMeasPoint);
end
end
```

7.6: Function iso11146Part1MeasurementData

```
function value = iso11146Part1MeasurementData(obj,centerGuessX,centerGuessY,
    integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,lambda,pixelsize)
%iso11146Part1MeasurementData calc LB parameters using ISO11146-1 for
%background corrected Measurementdata
```

```
4
5  measurementDataSet = obj.getMeasurementDataSet();
6  [Useless,NoMP] = size(measurementDataSet);
7  value = MeasurementData();
8  zPos = [];
9  dWx = [];
10 dWxStd = [];
11 dWy = [];
12 dWyStd = [];
13 for k = 1 : NoMP
14     fprintf('Computing ISO11146-1 parameters for measuring point %i.\n',k);
15     corrMeasPoint = measurementDataSet(k).iso11146Part1SingleMeasurementPoint(
           centerGuessX,centerGuessY,integAreaGuessX,integAreaGuessY,convergeKriterium,
           faktor,pixelsize);
16     value.addSingleMeasurementPoint(corrMeasPoint);
17
18     zPos = [zPos,corrMeasPoint.getZPos()];
19     dWx = [dWx,corrMeasPoint.getdWx_mean()];
20     dWxStd = [dWxStd,corrMeasPoint.getdWx_std];
21     dWy = [dWy,corrMeasPoint.getdWy_mean()];
22     dWyStd = [dWyStd,corrMeasPoint.getdWy_std];
23
24 end
25 offset_weight = 1;
26 weight = 0.1*ones(1,size(dWx,2));
27
28 [V,I] = min(dWx);
29 for i=1: size(dWx,2)
30     if (i >= I-offset_weight &&  i <= I+offset_weight) || i <= offset_weight || i>=
           size(dWx,2)-offset_weight
31         weight(i) = 1;
32     end
33 end
34
35 fx=fit(zPos',dWx.^2','poly2','Robust','Bisquare','Weight',weight);
36 value.fx = fx;
37
38 cx = fx.p1;
39 bx = fx.p2;
40 ax = fx.p3;
41
42 value.z0x = -bx/(2*cx);
43 value.dSigma0x = (1/(2*sqrt(cx)))*sqrt(4*ax*cx-bx^2);
44 value.thetaSigmaX = sqrt(cx);
45 value.zRx = 1/(2*cx)*sqrt(4*ax*cx-bx^2);
46 value.MsquaredX = pi/(8*lambda/1e+3)*sqrt(4*ax*cx-bx^2);
47 value.fx = fx;
48
49
```

```
weight = 0.1*ones(1,size(dWy,2));
[V,I] = min(dWy);
for i=1: size(dWy,2)
    if (i >= I-offset_weight &&  i <= I+offset_weight) || i <= offset_weight || i>=
        size(dWy,2)-offset_weight
        weight(i) = 1;
    end
end

fy = fit(zPos',dWy.^2','poly2','Robust','Bisquare','Weight',weight);
value.fy = fy;

cy = fy.p1;
by = fy.p2;
ay = fy.p3;

value.z0y = -by/(2*cy);
value.dSigma0y = (1/(2*sqrt(cy)))*sqrt(4*ay*cy-by^2);
value.thetaSigmaY = sqrt(cy);
value.zRy = 1/(2*cy)*sqrt(4*ay*cy-by^2);
value.MsquaredY = pi/(8*lambda/1000)*sqrt(4*ay*cy-by^2);
value.fy = fy;

value.zPos = zPos;
value.dWx = dWx;
value.dWxStd = dWxStd;
value.dWy = dWy;
value.dWyStd = dWyStd;


end
```

7.7: Class Background

```
classdef Background < handle
    %BACKGROUNDMEASUREMENTDATA Background data class containing background
    %Measuring points and functions to evaluate background data.

    properties(Access = public)
        backgroundDataSet SingleMeasurementPoint;
    end

    methods
        %         function delete(obj)
        %             delete(obj.backgroundDataSet);
        %         end
        function reset(obj)
            obj.backgroundDataSet = SingleMeasurementPoint.empty;
        end
        function value = getData(obj,measPoint)
```

```
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            value = obj.backgroundDataSet(measPoint);
        end
        addSingleMeasurementPoint(obj,value);
        value = getMeanValue(obj,measPoint);
        value = getMeanSubstractionMap(obj,measPoint);
        value = getMeanStandardDeviation(obj,measPoint);
        value = getMeanFromCorners(obj,measPoint,n,m);
    end
end
```

7.8: Function addSingleMeasurementPoint

```
function addSingleMeasurementPoint(obj,value)
%addSingleMeasurementPoint Adds a new Singlemeasurementpoint to the
%existing dataset
obj.backgroundDataSet(end+1) = value;
end
```

7.9: Function getMeanFromCorners

```
function value = getMeanFromCorners(obj,measPoint,n,m)
%GETMEANFROMCORNERS Returns the mean pixel value taken from corner areas.
%The averaged area is defined by n and m.
backgroundDataPoint = obj.getData(measPoint);
[Useless,NoMP] = size(backgroundDataPoint);
for k = 1 : NoMP
    backgroundDataPointSet = backgroundDataPoint(k).getDataSet();
    [Useless,NoI] = size(backgroundDataPointSet);
    for f = 1 : NoI
        backgroundDataPointImageData = backgroundDataPointSet(f).getImageData();

        [rows,cols] = size(backgroundDataPointImageData);
        temp = 0;
        counter = 0;
        for i=1:rows
            for j=1:cols
                if (i <= n && j <= m) || (i <= n && j >= cols-m+1) || (i >= rows-n+1
                    && j <= m) || (i >= rows-n+1 && j >= cols-m+1)
                    temp = temp + double(backgroundDataPointImageData(i,j));
                    counter = counter + 1;
                end

            end
        end
    end
end
value = temp/counter;
end
```

7.10: Function getMeanStandardDeviation

```
function value = getMeanStandardDeviation(obj,measPoint)
%GETMEANSTANDARDDEVIATION Returns the Standard Deviation of a Measuring
%Point. All Images belonging to Measuring Point are used.
backgroundDataPoint = obj.getData(measPoint);
backgroundDataPoint.calcMeanImage();
meanBack=backgroundDataPoint.getMeanImage().getImageData();
value = std(double(reshape(meanBack,[],1)));
end
```

7.11: Function getMeanSubstractionMap

```
function value = getMeanSubstractionMap(obj,measPoint);
%GETSUBSTRACTIONMAP Returns the mean Backgroundsubstraction map created
%using all Images belonging to the Measuring Point.

backgroundDataPoint = obj.getData(measPoint);
backgroundDataPointSet = backgroundDataPoint.getDataSet();
[Useless,NoI] = size(backgroundDataPointSet);
[rows,cols] = size(backgroundDataPointSet(1));
temp=zeros(rows,cols,'double');
for k = 1 : NoI
    temp = temp+double(backgroundDataPointSet(k).getImageData());
end
% value = cast(ceil(temp./NoI),'uint8');
value = cast(temp./NoI,'uint8');
end
```

7.12: Function getMeanSubstractionMapDouble

```
function value = getMeanSubstractionMapDouble(obj,measPoint);
%GETSUBSTRACTIONMAP Returns the mean Backgroundsubstraction map created
%using all Images belonging to the Measuring Point.

backgroundDataPoint = obj.getData(measPoint);
backgroundDataPointSet = backgroundDataPoint.getDataSet();
[Useless,NoI] = size(backgroundDataPointSet);
[rows,cols] = size(backgroundDataPointSet(1));
temp=zeros(rows,cols,'double');
for k = 1 : NoI
    temp = temp+double(backgroundDataPointSet(k).getImageData());
end
% value = cast(ceil(temp./NoI),'uint8');
value = temp./NoI;
end
```

7.13: Function getMeanValue

```
function value = getMeanValue(obj,measPoint)
%GETMEANVALUE Returns the mean pixel value of all images belonging to the
%Measuring point.
```

```
4
5  backgroundDataPoint = obj.getData(measPoint);
6  backgroundDataPointSet = backgroundDataPoint.getDataSet();
7  [Useless,NoI] = size(backgroundDataPointSet);
8  [rows,cols] = size(backgroundDataPointSet(end).getImageData());
9  temp=0;
10 for k = 1 : NoI
11     imageData=backgroundDataPointSet(k).getImageData();
12     temp = temp + mean( double(imageData) , 'all' );
13 end
14 value = temp/NoI;
15 end
```

7.14: Class MeasurementSettings

```
1  classdef MeasurementSettings
2      %MEASUREMENTSETTINGS data class containing information about the
3      %measurement settings
4
5      properties
6          measurementMethod string;
7          rayleighLength double;
8          timesReyleighLength double;
9          noMP double;
10         measPerPoint double;
11         measArea double;
12     end
13
14     methods
15         function obj = MeasurementSettings(rayleighLength,timesReyleighLength,noMP,
               measPerPoint)
16             %MEASUREMENTSETTINGS Construct an instance of this class
17             %   Detailed explanation goes here
18             obj.rayleighLength = rayleighLength;
19             obj.timesReyleighLength = timesReyleighLength;
20             obj.noMP = noMP;
21             obj.measPerPoint = measPerPoint;
22         end
23     end
24 end
```

7.15: Class SingleMeasurementPoint

```
1  classdef SingleMeasurementPoint < handle
2      %SINGLEMEASUREMENTPOINT Class containing the data of a single measurementpoint.
3
4      properties (Access = public)
5          dataSet SingleMeasurement;
6          zPos double;
7          meanImage SingleMeasurement;
```

```
8
9           dWx_mean double;
10          dWy_mean double;
11          azimutPhi_mean double;
12          beamCenterX_mean double;
13          beamCenterY_mean double
14          W_X_mean double;
15          W_Y_mean double;
16          W_XY_mean double;
17          W_X_squared_mean double;
18          W_Y_squared_mean double;
19
20          dWx_std double;
21          dWy_std double;
22          azimutPhi_std double;
23          beamCenterX_std double;
24          beamCenterY_std double;
25          W_X_std double;
26          W_Y_std double;
27          W_XY_std double;
28          W_X_squared_std double;
29          W_Y_squared_std double;
30
31      end
32
33      methods
34  % function delete(obj)
35  %             delete(obj.dataSet);
36  %         end
37          function obj = SingleMeasurementPoint(value)
38              %SINGLEMEASUREMENTPOINT Construct an instance of this class
39              %   Detailed explanation goes here
40              obj.zPos = value;
41          end
42          function value = getDataSet(obj)
43              % Get stored measurementpoint dataSet.
44              value = obj.dataSet;
45          end
46          function value = getZPos(obj)
47              % Get xCoordinate of measurementpoint.
48              value = obj.zPos;
49          end
50          addMeasurement(obj,value);
51
52          function value = getMeanImage(obj)
53              value = obj.meanImage;
54          end
55           function value = getdWx_mean(obj)
56              % Get xCoordinate of measurementpoint.
```

```
57              value = obj.dWx_mean;
58          end
59          function value = getdWy_mean(obj)
60              % Get xCoordinate of measurementpoint.
61              value = obj.dWy_mean;
62          end
63          function value = getW_X_mean(obj)
64              % Get xCoordinate of measurementpoint.
65              value = obj.W_X_mean;
66          end
67          function value = getW_Y_mean(obj)
68              % Get xCoordinate of measurementpoint.
69              value = obj.W_Y_mean;
70          end
71          function value = getW_XY_mean(obj)
72              % Get xCoordinate of measurementpoint.
73              value = obj.W_XY_mean;
74          end
75          function value = getW_X_squared_mean(obj)
76              % Get xCoordinate of measurementpoint.
77              value = obj.W_X_squared_mean;
78          end
79         function value = getW_Y_squared_mean(obj)
80              % Get xCoordinate of measurementpoint.
81              value = obj.W_Y_squared_mean;
82         end
83
84                  function value = getdWx_std(obj)
85              % Get xCoordinate of measurementpoint.
86              value = obj.dWx_std;
87          end
88          function value = getdWy_std(obj)
89              % Get xCoordinate of measurementpoint.
90              value = obj.dWy_std;
91          end
92          function value = getW_X_std(obj)
93              % Get xCoordinate of measurementpoint.
94              value = obj.W_X_std;
95          end
96          function value = getW_Y_std(obj)
97              % Get xCoordinate of measurementpoint.
98              value = obj.W_Y_std;
99          end
100         function value = getW_XY_std(obj)
101             % Get xCoordinate of measurementpoint.
102             value = obj.W_XY_std;
103         end
104         function value = getW_X_squared_std(obj)
105             % Get xCoordinate of measurementpoint.
```

```
            value = obj.W_X_squared_std;
         end
        function value = getW_Y_squared_std(obj)
            % Get xCoordinate of measurementpoint.
            value = obj.W_Y_squared_std;
        end

        function value = getAzimutPhi_mean(obj)
            % Get xCoordinate of measurementpoint.
            value = obj.azimutPhi_mean;
        end
        function value = getAzimutPhi_std(obj)
            % Get xCoordinate of measurementpoint.
            value = obj.azimutPhi_std;
        end
        function value = getBeamCenterX_mean(obj)
            % Get xCoordinate of measurementpoint.
            value = obj.beamCenterX_mean;
        end
         function value = getBeamCenterY_mean(obj)
            % Get xCoordinate of measurementpoint.
            value = obj.beamCenterY_mean;
        end

        showImage(obj,imageIndex);
        [centers,radii] = showImageWithCircleDetection(obj,imageIndex,dMin,dMax,
            objectPolarity,circleSensitivity,edgeThreshold);
        showAllImages(obj);
        showImageWithImTool(obj,imageIndex);
        calcMeanImage(obj);
        value = backgroundCorrectionSingleMeasurementPoint(obj,coarseMethod,
            fineMethod,background,kernelSize,ntFactor,k,manualValue);
        value = iso11146Part1SingleMeasurementPoint(obj,centerGuessX,centerGuessY,
            integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,pixelsize);
    end
    methods (Access = private)
        function setMeanImage(obj,value)
            obj.meanImage = value;
        end
    end
end
```

7.16: Function addMeasurement

```
function addMeasurement(obj,value)
%addMeasurement Adds a single measurement to the Measuring point dataset
obj.dataSet(end+1) = value;
end
```

7.17: Function backgroundCorrectionSingleMeasurementPoint

```
function value = backgroundCorrectionSingleMeasurementPoint(obj,coarseMethod,
    fineMethod,background,kernelSize,ntFactor,k,manualValue)
%BACKGROUNDCORRECTIONSINGLEMEASUREMENTPOINT perform background correction
%for a Measuring point

value = SingleMeasurementPoint(obj.getZPos());
measPointDataSet = obj.getDataSet();
[Useless,NoI] = size(measPointDataSet);
for j = 1 : NoI
    fprintf('Performing background correction for image %i.\n',j);
    corrMeas = measPointDataSet(j).backgroundCorrectionSingleMeasurement(
        coarseMethod,fineMethod,background,kernelSize,ntFactor,k,manualValue);
    value.addMeasurement(corrMeas);
end
end
```

7.18: Function backgroundCorrectionSingleMeasurementPoint

```
function value = backgroundCorrectionSingleMeasurementPoint(obj,coarseMethod,
    fineMethod,background,kernelSize,ntFactor,k,manualValue)
%BACKGROUNDCORRECTIONSINGLEMEASUREMENTPOINT perform background correction
%for a Measuring point

value = SingleMeasurementPoint(obj.getZPos());
measPointDataSet = obj.getDataSet();
[Useless,NoI] = size(measPointDataSet);
for j = 1 : NoI
    fprintf('Performing background correction for image %i.\n',j);
    corrMeas = measPointDataSet(j).backgroundCorrectionSingleMeasurement(
        coarseMethod,fineMethod,background,kernelSize,ntFactor,k,manualValue);
    value.addMeasurement(corrMeas);
end
end
```

7.19: Function iso11146Part1SingleMeasurementPoint

```
function value = iso11146Part1SingleMeasurementPoint(obj,centerGuessX,centerGuessY,
    integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,pixelsize)
%BACKGROUNDCORRECTIONSINGLEMEASUREMENTPOINT perform ISO11146-1 for a
%Measuring point.

value = SingleMeasurementPoint(obj.getZPos());
measPointDataSet = obj.getDataSet();
[Useless,NoI] = size(measPointDataSet);
dWx = [];
dWy = [];
azimutPhi = [];
beamCenterX = [];
beamCenterY = [];
W_X = [];
```

```
W_Y = [];
W_XY = [];
W_X_squared = [];
W_Y_squared = [];
for j = 1 : NoI
    fprintf('Computing ISO11146-1 parameters for image %i.\n',j);
    processedMeas = measPointDataSet(j).iso11146Part1SingleMeasurement(centerGuessX,
        centerGuessY,integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,
        pixelsize);
    dWx = [dWx;processedMeas.getdWx()];
    dWy = [dWy;processedMeas.getdWy()];
    azimutPhi = [azimutPhi;processedMeas.getAzimutPhi()];
    beamCenterX = [beamCenterX;processedMeas.getbeamCenterX()];
    beamCenterY = [beamCenterY;processedMeas.getbeamCenterY()];
    W_X = [W_X;processedMeas.getW_X()];
    W_Y = [W_Y;processedMeas.getW_Y()];
    W_X_squared = [W_X_squared;processedMeas.getW_X_squared()];
    W_Y_squared = [W_Y_squared;processedMeas.getW_Y_squared()];
    W_XY = [W_XY;processedMeas.getW_XY()];
    value.addMeasurement(processedMeas);
end
value.dWx_mean = mean(dWx,'all');
value.dWy_mean = mean(dWy,'all');
value.azimutPhi_mean = mean(azimutPhi,'all');
value.beamCenterX_mean = mean(beamCenterX,'all');
value.beamCenterY_mean = mean(beamCenterY,'all');
value.W_X_mean = mean(W_X,'all');
value.W_Y_mean = mean(W_Y,'all');
value.W_X_squared_mean = mean(W_X_squared,'all');
value.W_Y_squared_mean = mean(W_Y_squared,'all');
value.W_XY_mean = mean(W_XY,'all');

value.dWx_std = std(dWx);
value.dWy_std = std(dWy);
value.azimutPhi_std = std(azimutPhi);
value.beamCenterX_std = std(beamCenterX);
value.beamCenterY_std = std(beamCenterY);
value.W_X_std = std(W_X);
value.W_Y_std = std(W_Y);
value.W_X_squared_std = std(W_X_squared);
value.W_Y_squared_std = std(W_Y_squared);
value.W_XY_std = std(W_XY);
end
```

7.20: Function showAllImages

```
function showAllImages(obj)
%SHOWALLEIMAGES show all Images for a Measuring point.
[Useless NoI] = size(obj.dataSet);
cols = 5;
```

```
rows = fix(NoI/cols)+1;
restCols = rem(NoI,cols);

temp =[];
figure;
for j=1 : NoI
    temp = [temp obj.dataSet(j).getImageData()];
end
montage(temp);


% imageIndex = 1;
% c = gray;
% figure
%
%     for j=1 : NoI
%         subplot(rows,cols,j), imshow(obj.dataSet(imageIndex).getImage())
%     end
% end
```

7.21: Function showImage

```
function showImage(obj,imageIndex)
%SHOWIMAGE show a specified image of the measuring point.
figure;
imshow(obj.dataSet(imageIndex).imageData);
end
```

7.22: Function showImageWithCircleDetection

```
function [centers,radii] = showImageWithCircleDetection(obj,imageIndex,dMin,dMax,
    objectPolarity,circleSensitivity,edgeThreshold)
%SHOWIMAGEWITHCIRCLEDETECTION legacy code show image with matlab circle
%detection. Not usable for our purpose but fun anyway.
imageToShow = obj.dataSet(imageIndex).imageData;
figure;
imshow(imageToShow);
[centers,radii] = imfindcircles(imageToShow,[dMin/2 dMax/2],'ObjectPolarity',
    objectPolarity, ...
     'Sensitivity',circleSensitivity,'EdgeThreshold',edgeThreshold);
viscircles(centers, radii,'Color','r');
end
```

7.23: Function showImageWithImTool

```
function showImageWithImTool(obj,imageIndex)
%SHOWIMAGEWITHIMTOOL show a specified image in the imtool.
imtool(obj.dataSet(imageIndex).imageData);
end
```

7.24: Class SingleMeasurement

```
classdef SingleMeasurement
    %MEASURMENTDATA Class containing one single measurement.


    properties (Access = public)
        imageData (1026,1282);
        beamCenterX double;
        beamCenterY double;
        W_X double;
        W_Y double;
        W_X_squared double;
        W_Y_squared double;
        W_XY double;
        dWx double;
        dWy double;
        phi double;
    end

    methods
%         function delete(obj)
%             delete(obj.imageData);
%         end
        function obj = SingleMeasurement(value)
            %MEASURMENTDATA Construct an instance of this class
            %   Detailed explanation goes here
            obj.imageData = value;
        end

        function value = getImageData(obj)
            % Get stored measurment image data.
            value = obj.imageData;
        end
        function value = getdWx(obj)
            % Get stored measurment image data.
            value = obj.dWx;
        end
        function value = getdWy(obj)
            % Get stored measurment image data.
            value = obj.dWy;
        end
        function value = getAzimutPhi(obj)
            % Get stored measurment image data.
            value = obj.phi;
        end
        function value = getbeamCenterX(obj)
            % Get stored measurment image data.
            value = obj.beamCenterX;
        end
```

```
        function value = getbeamCenterY(obj)
            % Get stored measurment image data.
            value = obj.beamCenterY;
        end
        function value = getW_X(obj)
            % Get stored measurment image data.
            value = obj.W_X;
        end
        function value = getW_Y(obj)
            % Get stored measurment image data.
            value = obj.W_Y;
        end
        function value = getW_X_squared(obj)
            % Get stored measurment image data.
            value = obj.W_X_squared;
        end
        function value = getW_Y_squared(obj)
            % Get stored measurment image data.
            value = obj.W_Y_squared;
        end
        function value = getW_XY(obj)
            % Get stored measurment image data.
            value = obj.W_XY;
        end
        value = backgroundCorrectionSingleMeasurement(obj,coarseMethod,fineMethod,
            background,kernelSize,ntFactor,k,manualValue);
        value = calcValuesISO11146Part1(obj,centerGuessX,centerGuessY,
            integAreaGuessX,integAreaGuessY,convergeKriterium,faktor);
         value = calcValuesISO11146Part2(obj,centerGuessX,centerGuessY,
             integAreaGuessX,integAreaGuessY,convergeKriterium,faktor);
%
    end
end
```

7.25: Function backgroundCorrectionSingleMeasurement

```
function value = backgroundCorrectionSingleMeasurement(obj,coarseMethod,...
    fineMethod,background,kernelSize,ntFactor,k,manualValue)
plotBool=false;
singleMeasurementImageData = obj.getImageData();
[rows,cols] = size(singleMeasurementImageData);
%  imtool(double(singleMeasurementImageData));
if plotBool
    imtool(singleMeasurementImageData);
    caxis([0 255])
    colorbar('FontSize',20);
end
switch coarseMethod
    case 'Background Map Substraction'
        backgroundSubstractionMap = background.getMeanSubstractionMapDouble(k);
```

```
15          singleMeasurementImageData = imsubtract(double(singleMeasurementImageData),
                backgroundSubstractionMap);
16          for i=1:rows
17              for j=1:cols
18                  if singleMeasurementImageData(i,j)<0
19                      singleMeasurementImageData(i,j)=0;
20                  end
21              end
22          end
23          %         imtool(double(singleMeasurementImageData));
24          if plotBool
25              imtool(singleMeasurementImageData);
26              caxis([0 255])
27              colorbar('FontSize',20);
28          end
29          value = SingleMeasurement(singleMeasurementImageData);
30      case 'Average Background Substraction'
31          E_b_offset = double(background.getMeanValue(k));
32          singleMeasurementImageData = imsubtract(double(singleMeasurementImageData),
                ones(rows,cols)*E_b_offset);
33          %         imtool(double(singleMeasurementImageData));
34          for i=1:rows
35              for j=1:cols
36                  if singleMeasurementImageData(i,j)<0
37                      singleMeasurementImageData(i,j)=0;
38                  end
39              end
40          end
41          if plotBool
42              imtool(singleMeasurementImageData);
43              caxis([0 255])
44              colorbar('FontSize',20);
45          end
46          value = SingleMeasurement(singleMeasurementImageData);
47  end
48  [height,width] = size(singleMeasurementImageData);
49  kernel = [round(height*kernelSize/100) round(width*kernelSize/100)];
50  switch fineMethod
51      case 'Statistical Method'
52          %% calc background guess for E_b_offset
53          E_b_offsetGuess = background.getMeanValue(k);
54          %% calc background Guess for standard deviation
55          E_b_SigmaGuess = background.getMeanStandardDeviation(k);
56          %% calc 2-D averaging subarray convolution
57          avgFilt = fspecial('average',kernel);
58          castedConvImage = imfilter(singleMeasurementImageData,  avgFilt);
59          E_b_Sigma_Tilde = std(double(reshape(castedConvImage,[],1)));
60          temp = 0;
61          counter = 0;
```

```
        for i=1:rows
            for j=1:cols
                if double(castedConvImage(i,j)) < double(E_b_offsetGuess)+ntFactor*
                    E_b_Sigma_Tilde/(sqrt((kernel(1)+1)*(kernel(2)+1)))
                    if singleMeasurementImageData(i,j) >0
                        temp = temp + double(singleMeasurementImageData(i,j));
                        counter = counter+1;
                    end
                end
            end
        end
        E_b_offset = temp/counter;
        singleMeasurementImageData = imsubtract(singleMeasurementImageData,ones(rows
            ,cols)*E_b_offset);
        for i=1:rows
            for j=1:cols
                if singleMeasurementImageData(i,j)<0
                    singleMeasurementImageData(i,j)=0;
                end
            end
        end
        %         imtool(double(singleMeasurementImageData));
        if plotBool
            imtool(singleMeasurementImageData);
            caxis([0 255])
            colorbar('FontSize',20);
        end
        value = SingleMeasurement(singleMeasurementImageData);

    case 'Approximation Method'
        E_b_offsetDark = background.getMeanFromCorners(k,kernel,kernel);
        %% calc background Guess for standard deviation
        E_b_SigmaDark=background.getMeanStandardDeviation(k);
        %% calc E_b_offsetMeas
        original=obj.getImageData();
        temp = 0;
        counter = 0;
        for i=1:rows
            for j=1:cols
                if (i <= kernel(1) && j <= kernel(2)) ...
                        || (i <= kernel(1) && j >= cols-kernel(2)+1) ...
                        || (i >= rows-kernel(1)+1 && j <= kernel(2)) ...
                        || (i >= rows-kernel(1)+1 && j >= cols-kernel(2)+1)
                    temp = temp + double(original(i,j));
                    counter = counter + 1;
                end
            end
        end
        E_b_offsetMeas = temp/counter;
```

```
        if abs(double(E_b_offsetMeas) - double(E_b_offsetDark))*sqrt(counter)/double
            (E_b_SigmaDark) < ntFactor
            singleMeasurementImageData = imsubtract(singleMeasurementImageData,cast(
                ones(rows,cols)*E_b_offsetMeas,'uint8'));
            for i=1:rows
                for j=1:cols
                    if singleMeasurementImageData(i,j)<0
                        singleMeasurementImageData(i,j)=0;
                    end
                end
            end
            %            imtool(double(singleMeasurementImageData));
            if plotBool
                imtool(singleMeasurementImageData);
                caxis([0 255])
                colorbar('FontSize',20);
            end

            value = SingleMeasurement(singleMeasurementImageData);
        else
            value = SingleMeasurement(zeros(height,width));
            fprintf('\nDifference between dark backgroundimage and measrumentimage
                is too big.\nLaserbeam might be too big for Sensor\n');
        end
        %     case 'Median Filter 3x3'
        %         singleMeasurementImageData = medfilt2(singleMeasurementImageData
            ,[3 3]);
        %         value = SingleMeasurement(singleMeasurementImageData);
        %     case 'Median Filter 5x5'
        %         singleMeasurementImageData = medfilt2(singleMeasurementImageData
            ,[5 5]);
        %         value = SingleMeasurement(singleMeasurementImageData);
        %
    case 'Manual Value'
        E_b_offset = manualValue;
        singleMeasurementImageData = imsubtract(singleMeasurementImageData,ones(rows
            ,cols)*E_b_offset);
        for i=1:rows
            for j=1:cols
                if singleMeasurementImageData(i,j)<0
                    singleMeasurementImageData(i,j)=0;
                end
            end
        end
        if plotBool
            imtool(singleMeasurementImageData);
            caxis([0 255])
            colorbar('FontSize',20);
        end
```

```
        value = SingleMeasurement(singleMeasurementImageData);
end

end
```

7.26: Function iso11146Part1SingleMeasurement

```
function value = iso11146Part1SingleMeasurement(obj,centerGuessX,centerGuessY,
    integAreaGuessX,integAreaGuessY,convergeKriterium,faktor,pixelsize)
%CALCVALUESISO11146PART1 calc LB parameters using 11146-1 for single
%Measurement

%% set initial Values
oldCenterX = 0;
oldCenterY = 0;
old_dSigmaX = 0;
old_dSigmaY = 0;

newCenterX = centerGuessX;
newCenterY = centerGuessY;
new_dWx = integAreaGuessX;
new_dWy = integAreaGuessY;
originalImage = obj.getImageData();
centerInPixelX = round(newCenterX/pixelsize);
centerInPixelY = round(newCenterY/pixelsize);
%% start IterationProcess
while abs(newCenterX-oldCenterX) > convergeKriterium ||...
        abs(newCenterY-oldCenterY) > convergeKriterium ||...
        abs(new_dWx-old_dSigmaX) > convergeKriterium ||...
        abs(new_dWy-old_dSigmaY) > convergeKriterium

    if mod(fix(faktor*new_dWx/pixelsize),2) == 0
        intAreaInPixelX = fix(faktor*new_dWx/pixelsize+1);
    else
        intAreaInPixelX = fix(faktor*new_dWx/pixelsize);
    end

    if mod(fix(faktor*new_dWy/pixelsize),2) == 0
        intAreaInPixelY = fix(faktor*new_dWy/pixelsize+1);
    else
        intAreaInPixelY = fix(faktor*new_dWy/pixelsize);
    end

    %% create Subimage used for calc
    subImage = originalImage(centerInPixelY-floor(intAreaInPixelY/2):centerInPixelY+
        floor(intAreaInPixelY/2),...
        centerInPixelX-floor(intAreaInPixelX/2):centerInPixelX+floor(intAreaInPixelX
            /2));

    %% create Mesh for integration
```

```
    [rows,cols]=size(subImage);
    x = -floor(cols/2)*pixelsize:pixelsize:floor(cols/2)*pixelsize;
    y = floor(rows/2)*pixelsize:-pixelsize:-floor(rows/2)*pixelsize;
    [X,Y] = meshgrid(x,y);
    %% calc  newCenter
    F = double(subImage).*X;
    I = trapz(y,trapz(x,F,2));
    I2 = trapz(y,trapz(x,double(subImage),2));
    oldCenterX = newCenterX;
    W_X = I/I2;

    F = double(subImage).*Y;
    I = trapz(y,trapz(x,F,2));
    I2 = trapz(y,trapz(x,double(subImage),2));
    oldCenterY = newCenterY;
    W_Y = I/I2;

    centerInPixelX = centerInPixelX+round(W_X/pixelsize);
    centerInPixelY = centerInPixelY-round(W_Y/pixelsize);
    newCenterY = centerInPixelY*pixelsize;
    newCenterX = centerInPixelX*pixelsize;

    %% calc Sigma_X_squared
    F = double(subImage).*((X-W_X).^2);
    I = trapz(y,trapz(x,F,2));
    I2 = trapz(y,trapz(x,double(subImage),2));
    W_X_squared = I/I2;
    %% calc Sigma_Y_squared
    F = double(subImage).*((Y-W_Y).^2);
    I = trapz(y,trapz(x,F,2));
    I2 = trapz(y,trapz(x,double(subImage),2));
    W_Y_squared = I/I2;
    %% calc Sigma_XY_squared
    F = double(subImage).*((X-W_X).*(Y-W_Y));
    I = trapz(y,trapz(x,F,2));
    I2 = trapz(y,trapz(x,double(subImage),2));

    W_XY = I/I2;
    %% different equations used depending if Sigma_X_squared == Sigma_Y_squared
    if abs(W_X_squared-W_Y_squared) < 1e8*eps(min(abs(W_X_squared),abs(W_Y_squared))
        )
        %% calc Azimutwinkel
        if W_XY == 0
            azimutSign = 1;
        else
            azimutSign = W_XY/abs(W_XY);
        end
        phi = azimutSign*45; % or pi/4
        %% calc new_dSigmaX
```

```
        old_dSigmaX = new_dWx;
        new_dWx = 2*sqrt(2)*(W_X_squared+W_Y_squared+2*abs(W_XY))^(1/2);
        %% calc new_dSigmaY
        old_dSigmaY = new_dWy;
        new_dWy = 2*sqrt(2)*(W_X_squared+W_Y_squared-2*abs(W_XY))^(1/2);
    else
        phi = 0.5*atand(2*W_XY/(W_X_squared-W_Y_squared));
        %% calc gamma is 1 or -1
        gamma = (W_X_squared-W_Y_squared) / abs(W_X_squared-W_Y_squared);
        %% calc new_dSigmaX
        old_dSigmaX = new_dWx;
        new_dWx=2*sqrt(2)*((W_X_squared+W_Y_squared)+gamma*((W_X_squared-W_Y_squared
            )^2+4*W_XY^2)^(1/2))^(1/2);
        %% calc new_dSigmaY
        old_dSigmaY = new_dWy;
        new_dWy = 2*sqrt(2)*((W_X_squared+W_Y_squared)-gamma*((W_X_squared-
            W_Y_squared)^2+4*W_XY^2)^(1/2))^(1/2);

    end
    epsilon = min([new_dWx,new_dWy])/max([new_dWx,new_dWy]);
    %% check if elipticity is bigger than 0.87 and if it is beamdiameter can be
        assumed to be a circle
    if epsilon > 0.87
        new_dWx = 2*sqrt(2)*(W_X_squared+W_Y_squared)^(1/2);
        new_dWy = 2*sqrt(2)*(W_X_squared+W_Y_squared)^(1/2);
    end
end
obj.beamCenterX = newCenterX;
obj.beamCenterY = newCenterY;
obj.W_X = W_X;
obj.W_Y = W_Y;
obj.W_X_squared = W_X_squared;
obj.W_Y_squared = W_Y_squared;
obj.W_XY = W_XY;
obj.dWx = new_dWx;
obj.dWy = new_dWy;
obj.phi = phi;
value = obj;
end
```

7.27: Class Result

```
classdef Result
    %RESULT Measurement evaluation result containing processed data and
    %calced parameters


    properties
        backgroundCorrMethod string;
        evalNorm string;
```

```
        lambda double;
        convCriterion double;
        nT double
        integAreaFactor double
        evaluatedMeasurementData MeasurementData = MeasurementData();
        kernelSizeInPercent double;
    end

    methods
        function obj = Result(coarseBackCorrMethod,fineBackgCorrMethod,evalNorm,
            lambda,convCriterion,nT,integAreaFactor,kernelSizeInPercent,
            evaluatedMeasurementData)
            %RESULT Construct an instance of this class
            %   Detailed explanation goes here
            switch coarseBackCorrMethod
                case 'Background Map Substraction'
                    p1 = 'BMS';
                case 'Average Background Substraction'
                    p1 = 'ABS';
            end
            switch fineBackgCorrMethod
                case 'Statistical Method'
                    p2 = 'SM';
                case 'Manual Value'
                    p2 = 'MV';
                case 'Off'
                    p2 = 'Off';
            end
            obj.backgroundCorrMethod = sprintf('%s/%s',p1,p2);
            obj.lambda = lambda;
            obj.evalNorm = evalNorm;
            obj.convCriterion = convCriterion;
            obj.integAreaFactor = integAreaFactor;
            obj.nT = nT;
            obj.evaluatedMeasurementData = evaluatedMeasurementData;
            obj.kernelSizeInPercent = kernelSizeInPercent;
        end
        function value = getEvaluatedMeasurementData(obj)
            % Get stored measurment data.
            value = obj.evaluatedMeasurementData;
        end
        function value = getBackgroundCorrMethod(obj)
            % Get stored measurment data.
            value = obj.backgroundCorrMethod;
        end
        function value = getLambda(obj)
            % Get stored measurment data.
            value = obj.lambda;
        end
```

```
        function value = getConvCriterion(obj)
            % Get stored measurment data.
            value = obj.convCriterion;
        end
        function value = getIntegAreaFactor(obj)
            % Get stored measurment data.
            value = obj.integAreaFactor;
        end
        function value = getNt(obj)
            % Get stored measurment data.
            value = obj.nT;
        end
        function value = getEvalNorm(obj)
            % Get stored measurment data.
            value = obj.evalNorm;
        end
        function value = getKernelSizeInPercent(obj)
            % Get stored measurment data.
            value = obj.kernelSizeInPercent;
        end
    end
end
```

7.28: Function lbcUI_Launcher

```
addpath('../backend');
fig = lbcUI;
```

7.29: Function CalculatePacketDelay

```
function [delay] = CalculatePacketDelay(vid, fps)
% CalculatePacketDelay Calculates recommended packet delay for a GigE Vision camera.
%
%    DELAY = CalculatePacketDelay(VID, FPS) calculates recommended packet
%    delay DELAY for a GigE Vision camera, represented by an image
%    acquisition videoinput or gigecam object VID.
%    Camera framerate FPS (frames per second) is required and is specific
%    to the camera configuration. The companion CalculateFrameRate function
%    can be used to measure the camera framerate, if unknown.
%
%    Example with videoinput vid:
%       vid = videoinput('gige', 1);
%       src = vid.Source;
%       src.PacketSize = 9000;
%       fps = CalculateFrameRate(vid, 20);
%       delay = CalculatePacketDelay(vid, fps);
%       src.PacketDelay = delay;
%
%    Example with gigecam g:
%       g = gigecam;
```

```
%       g.GevSCPSPacketSize = 9000;
%       fps = CalculateFrameRate(g, 20);
%       delay = CalculatePacketDelay(g, fps);
%       g.GevSCPD = delay;

%   Copyright (c) 2013 – 2015 The MathWorks, Inc.

if nargin < 2
    error('CalculatePacketDelay(vid, fps) videoinput/gigecam object and framerate
        are required arguments to run this utility.');
else
    if ~isnumeric(fps)
        error('CalculatePacketDelay(vid, fps) fps must be a number');
    end
    if ~(isscalar(fps) && isfinite(fps) && (fps > 0))
        error('CalculatePacketDelay(vid, fps) fps must finite and > 0 to find a
            frame rate');
    end
end

if ~(isa(vid, 'videoinput') || isa(vid, 'gigecam')) || ~isvalid(vid) || ~isscalar(
    vid)
    error('CalculatePacketDelay(vid, fps) vid must be a valid videoinput or gigecam
        object');
end

% get  – packetSize (GevSCPSPacketSize)
%      – TickFreq (GevTimestampTickFrequency)
%      – video format (PixelFormat)
%      – frame height
%      – frame width

switch class(vid)
    case 'videoinput'
        src = getselectedsource(vid);

        % get packet size (depending on MATLAB release PacketSize is int32 or char)
        if isnumeric(src.PacketSize)
            packetSize = double(src.PacketSize);
        else
            packetSize = str2double(src.PacketSize);
        end

        try
            TickFreq = src.TimestampTickFrequency;
        catch e
            if strcmp(e.identifier,'MATLAB:noSuchMethodOrField') || strcmp(e.
                identifier,'testmeas:getset:invalidProperty')
                 error('Packet delay is not supported on the device');
```

```
            end
        end
        pixelFormat = vid.videoFormat;
        roi = vid.ROIPosition;
        height = roi(1,4);
        width = roi(1,3);
    case 'gigecam'
        packetSize = double(vid.GevSCPSPacketSize);
        TickFreq = double(vid.GevTimestampTickFrequency);
        pixelFormat = vid.PixelFormat;
        height = double(vid.Height);
        width = double(vid.Width);
   otherwise
        % unexpected case
        error('CalculatePacketDelay(vid, fps) vid must be a valid videoinput or
            gigecam object');
end

switch (pixelFormat)
    case {'Mono8' ,'BayerGR8','BayerRG8','BayerGB8','BayerBG8'}
        BytesPerPixel = 1;
    case {'Mono10Packed' ,'YUV411Packed'}
        BytesPerPixel = 1.5;
    case {'Mono10','Mono12','Mono14' ,'Mono16','BayerGR10','BayerRG10','BayerGB10','
        BayerBG10','BayerGR12','BayerRG12','BayerGB12','BayerBG12','BayerGR16','
        BayerRG16','BayerGB16','BayerBG16','YUV422Packed',''}
        BytesPerPixel =2;
    case {'RGB8Packed','BGR8Packed','YUV444Packed','RGB8Planar',}
        BytesPerPixel=3;
    case {'RGBA8Packed','BGRA8Packed'}
        BytesPerPixel = 4;
    case {'RGB10Packed','BGR10Packed','RGB12Packed','BGR12Packed','RGB10Planar','
        RGB12Planar','RGB16Planar'}
        BytesPerPixel = 6;
end


% calculate packet delay
fprintf('Calculating packet delay for:\n frame rate = %d fps,\tPacketSize (
    GevSCPSPacketSize) = %d,\tframe height = %d,\tframe width = %d,\tpixel format =
    %s, \n',...
    fps, packetSize, height, width, pixelFormat);
fprintf(1, 'camera time stamp tick frequency (ticks/s) = %.1f \n', TickFreq);

numOfBytes_EthernetHeader = 14;
numOfBytes_IPHeader  = 20;
numOfBytes_UDPHeader = 8;
numOfBytes_GVSPHeader = 8;
numOfBytes_EthernetFooter = 2;
```

```
108
109 numOfBytes_overheadPerPacket = numOfBytes_EthernetHeader+numOfBytes_IPHeader+
        numOfBytes_UDPHeader+numOfBytes_GVSPHeader;
110 numOfBytes_GVSP_Leader = numOfBytes_overheadPerPacket+36;
111 numOfBytes_GVSP_Trailer = numOfBytes_overheadPerPacket+numOfBytes_EthernetFooter+
        numOfBytes_GVSPHeader; %Ethernet Footer+GVSP Header
112
113 actual_packetSize = packetSize-numOfBytes_overheadPerPacket;
114
115 numOfBytes_perFrame = height  *  width  *  BytesPerPixel;
116
117 numOfPackets_perFrame = ceil(numOfBytes_perFrame/actual_packetSize)+2 ;
118
119 total_OverHead_Bytes = ceil(numOfPackets_perFrame) * numOfBytes_overheadPerPacket;
120
121 full_packets = floor(numOfPackets_perFrame ) * actual_packetSize;
122
123 numOfBytes_perImage = numOfBytes_GVSP_Leader+(floor(numOfPackets_perFrame ) *
        actual_packetSize)+(numOfBytes_perFrame-(full_packets))+total_OverHead_Bytes +
        numOfBytes_GVSP_Trailer;
124
125 numOfBytes_perSec  = fps * numOfBytes_perImage;
126
127 bits_perSecond = numOfBytes_perSec  * 8;
128
129 ratio_GigeUtilized= bits_perSecond/10^9;
130
131 ratio_GigeNotUtilized = 1 - ratio_GigeUtilized;
132
133 pauseTime_between_Packets = ratio_GigeNotUtilized/(numOfPackets_perFrame * fps);
134
135 PacketDelay = TickFreq * pauseTime_between_Packets;
136
137 delay = round(0.9 * PacketDelay);
138 fprintf(1, '\nUsed gigabit bandwitdh: %.1f %%\n', 100 * ratio_GigeUtilized);
139 fprintf(1, 'PacketDelay (GevSCPD): %d (ticks)\n', delay);
140
141
142 end
```

7.30: Class lbcUI

```
1 classdef lbcUI < matlab.apps.AppBase
2     % Properties that correspond to app components
3     properties (Access = public)
4         UIFigure                matlab.ui.Figure
5         TabGroup                matlab.ui.container.TabGroup
6         HardwareSetupTab        matlab.ui.container.Tab
7         HSGridLayout            matlab.ui.container.GridLayout
8         %% define Hardware Setup Configuration Properties
```

```
        HSCamConfPanel           matlab.ui.container.Panel
        HSCamConfGridLayout      matlab.ui.container.GridLayout
        HSCamConfLabel       matlab.ui.control.Label
        HSCamConfConnectButton matlab.ui.control.Button
        HSCamConfDisconnectButton matlab.ui.control.Button
        HSCamConfCamUIAxes  matlab.ui.control.UIAxes
        HSCamConfCamTestButton matlab.ui.control.Button
        HSCamConfExposureTimeLabel   matlab.ui.control.Label
        HSCamConfExposureTimeEditField  matlab.ui.control.NumericEditField
        HSCamConfApplySettingButton matlab.ui.control.Button
        HSCamConfCamPreviewButton matlab.ui.control.StateButton
        HSCamConfConnectionLabel      matlab.ui.control.Label


        HSStepmotorConfPanel     matlab.ui.container.Panel
        HSStepmotorConfGridLayout   matlab.ui.container.GridLayout
        HSStepmotorConfLabel       matlab.ui.control.Label
        HSStepmotorConfComPortLabel   matlab.ui.control.Label
        HSStepmotorConfComPortRefreshButton matlab.ui.control.Button
        HSStepmotorConfComPortDropDown    matlab.ui.control.DropDown
        HSStepmotorConfConnectButton matlab.ui.control.Button
        HSStepmotorConfDisconnectButton matlab.ui.control.Button
        HSStepmotorConfStepsPerRevolutionLabel   matlab.ui.control.Label
        HSStepmotorConfStepsPerRevolutionEditField   matlab.ui.control.
            NumericEditField
        HSStepmotorConfRpmLabel   matlab.ui.control.Label
        HSStepmotorConfRpmEditField   matlab.ui.control.NumericEditField
        HSStepmotorConfApplySettingsButton matlab.ui.control.Button
        HSStepmotorConfConnectionLabel      matlab.ui.control.Label

        HSTestHardwarePanel     matlab.ui.container.Panel
        HSTestHardwareGridLayout   matlab.ui.container.GridLayout
        HSTestHardwareReferenceStageButton matlab.ui.control.Button
        HSTestHardwareGoToZeroButton matlab.ui.control.Button
        HSTestHardwareMotorStatusLabel      matlab.ui.control.Label
        HSTestHardwareGoToPositionLabel   matlab.ui.control.Label
        HSTestHardwareGoToPositionEditField   matlab.ui.control.NumericEditField
        HSTestHardwareGoToPositionButton matlab.ui.control.Button
        HSTestHardwarePositionUIAxes  matlab.ui.control.UIAxes
        HSTestHardwareCurrPosLabel      matlab.ui.control.Label
        HSTestHardwareCurrPosEditField matlab.ui.control.NumericEditField


        %% define Automated Measurement Properties
        AutoMeasTab  matlab.ui.container.Tab
        AutoMeasTabGroup             matlab.ui.container.TabGroup
        AutoMeasISO111146Part1Tab              matlab.ui.container.Tab
        AutoMeasISO111146Part1TabGridLayout    matlab.ui.container.GridLayout

```

```
        AutoMeasISO111146DistanceHolderLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1MethodRBGroup      matlab.ui.container.ButtonGroup
        AutoMeasISO111146Part1FromCenterRB      matlab.ui.control.RadioButton
        AutoMeasISO111146Part1FromZeroRB      matlab.ui.control.RadioButton
        AutoMeasISO111146Part1OffsetLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1OffseEditField matlab.ui.control.NumericEditField
        AutoMeasISO111146Part1RayleighLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1RayleighEditField matlab.ui.control.NumericEditField
        AutoMeasISO111146Part1TimesRayleighLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1TimesRayleighEditField matlab.ui.control.
            NumericEditField
        AutoMeasISO111146Part1NoMeasPointsLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1NoMeasPointsEditField matlab.ui.control.
            NumericEditField
        AutoMeasISO111146Part1MeasPerPointLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1MeasPerPointEditField matlab.ui.control.
            NumericEditField
        AutoMeasISO111146Part1MeasAreaLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1MeasAreaEditField matlab.ui.control.NumericEditField
        AutoMeasISO111146Part1StartMeasButton   matlab.ui.control.Button
        AutoMeasISO111146Part1PreviewMeasPointsUIAxes  matlab.ui.control.UIAxes
        AutoMeasISO111146Part1ProgressCurrActionLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1ProgressCurrActionEditField matlab.ui.control.
            EditField
        AutoMeasISO111146Part1ProgressCurrMeasPointLabel      matlab.ui.control.
            Label
        AutoMeasISO111146Part1ProgressCurrMeasPointEditField matlab.ui.control.
            EditField
        AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel      matlab.ui.control.
            Label
        AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField matlab.ui.control.
            EditField
        AutoMeasISO111146Part1ProgressSaveMeasButton   matlab.ui.control.Button
        AutoMeasISO111146Part1ProgressCurrPositionUIAxes  matlab.ui.control.UIAxes
        AutoMeasISO111146Part1StatusLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1CurrPosLabel      matlab.ui.control.Label
        AutoMeasISO111146Part1CurrPosEditField matlab.ui.control.NumericEditField
        AutoMeasCancelButton matlab.ui.control.StateButton

        %% define Evaluate Measurement Properties
        EvaluateMeasurementTab   matlab.ui.container.Tab
        EvalTabGroup             matlab.ui.container.TabGroup
        ConfEvalTab              matlab.ui.container.Tab
        ConfEvalTabGridLayout    matlab.ui.container.GridLayout
        ConfEvalLoadMeasButton   matlab.ui.control.Button
        ConfEvalSmpDropDown      matlab.ui.control.DropDown
        ConfEvalSmpLabel         matlab.ui.control.Label
        ConfEvalSmpImDropDown    matlab.ui.control.DropDown
        ConfEvalSmpImLabel       matlab.ui.control.Label
```

```
        ConfEvalIntegAreaFactorSlider matlab.ui.control.Slider
        ConfEvalIntegAreaEditField matlab.ui.control.NumericEditField
        ConfEvalIntegAreaFactorLabel      matlab.ui.control.Label
        ConfEvalNtFactorSlider matlab.ui.control.Slider
        ConfEvalNtEditField matlab.ui.control.NumericEditField
        ConfEvalNtFactorLabel      matlab.ui.control.Label
        ConfEvalWavelengthSlider matlab.ui.control.Slider
        ConfEvalWavelengthEditField matlab.ui.control.NumericEditField
        ConfEvalWavelengthLabel      matlab.ui.control.Label
        ConfEvalConvergeCriterionSlider matlab.ui.control.Slider
        ConfEvalConvergeCriterionEditField matlab.ui.control.NumericEditField
        ConfEvalConvergeCriterionLabel      matlab.ui.control.Label
        ConfEvalResLabel      matlab.ui.control.Label
        ConfEvalResWidthEditField matlab.ui.control.NumericEditField
        ConfEvalResHeightEditField matlab.ui.control.NumericEditField
        ConfEvalPixelsizeLabel      matlab.ui.control.Label
        ConfEvalPixelsizeEditField matlab.ui.control.NumericEditField
        ConfEvalKernelSizeSlider matlab.ui.control.Slider
        ConfEvalKernelSizeEditField matlab.ui.control.NumericEditField
        ConfEvalKernelSizeLabel      matlab.ui.control.Label
        ConfEvalDrawRectangleButton   matlab.ui.control.Button
        ConfEvalMethodDropDown   matlab.ui.control.DropDown
        ConfEvalMethodLabel      matlab.ui.control.Label
        ConfEvalBackCorrMethodDropDown matlab.ui.control.DropDown
        ConfEvalBackCorrMethodLabel matlab.ui.control.Label
        ConfEvalFineBackCorrMethodDropDown matlab.ui.control.DropDown
        ConfEvalFineBackCorrMethodLabel matlab.ui.control.Label
        ConfEvalEvaluationButton matlab.ui.control.Button
        ConfEvalEvaluationStatusLabel matlab.ui.control.Label
        ConfEvalPreviewImUIAxes  matlab.ui.control.UIAxes
        ConfEvalDrawRectButton   matlab.ui.control.Button
        ConfEvalFineCorrValueEditField matlab.ui.control.NumericEditField
        ConfEvalFineCorrValueLabel      matlab.ui.control.Label
        %% LiveMeasurementTab
        LiveMeasTab       matlab.ui.container.Tab
        LiveMeasTabGridLayout    matlab.ui.container.GridLayout
        LiveMeasIntegAreaEditField matlab.ui.control.NumericEditField
        LiveMeasIntegAreaFactorLabel      matlab.ui.control.Label
        LiveMeasNtEditField matlab.ui.control.NumericEditField
        LiveMeasNtFactorLabel      matlab.ui.control.Label
        LiveMeasConvergeCriterionEditField matlab.ui.control.NumericEditField
        LiveMeasConvergeCriterionLabel      matlab.ui.control.Label
        LiveMeasResLabel      matlab.ui.control.Label
        LiveMeasResWidthEditField matlab.ui.control.NumericEditField
        LiveMeasResHeightEditField matlab.ui.control.NumericEditField
        LiveMeasPixelsizeLabel      matlab.ui.control.Label
        LiveMeasPixelsizeEditField matlab.ui.control.NumericEditField
        LiveMeasKernelSizeEditField matlab.ui.control.NumericEditField
        LiveMeasKernelSizeLabel      matlab.ui.control.Label
```

```
        LiveMeasMethodDropDown   matlab.ui.control.DropDown
        LiveMeasMethodLabel      matlab.ui.control.Label
        LiveMeasBackCorrMethodDropDown matlab.ui.control.DropDown
        LiveMeasBackCorrMethodLabel matlab.ui.control.Label
        LiveMeasFineBackCorrMethodDropDown matlab.ui.control.DropDown
        LiveMeasFineBackCorrMethodLabel matlab.ui.control.Label
        LiveMeasBackgroundMeasButton matlab.ui.control.Button
        LiveMeasPreviewImUIAxes  matlab.ui.control.UIAxes
        LiveMeasResultImUIAxes  matlab.ui.control.UIAxes
        LiveMeasStartStopButton matlab.ui.control.StateButton
        LiveMeasStatusLabel     matlab.ui.control.Label
        LiveMeasImData_dWx matlab.ui.control.NumericEditField
        LiveMeasImData_dWx_Label matlab.ui.control.Label
        LiveMeasImData_dWy matlab.ui.control.NumericEditField
        LiveMeasImData_dWy_Label matlab.ui.control.Label
        LiveMeasImData_phi matlab.ui.control.NumericEditField
        LiveMeasImData_phi_Label matlab.ui.control.Label
        LiveMeasTriggerImageButton matlab.ui.control.Button
        LiveMeasDrawRectButton matlab.ui.control.Button
        LiveMeasFineCorrValueEditField matlab.ui.control.NumericEditField
        LiveMeasFineCorrValueLabel      matlab.ui.control.Label
        LiveMeasChangeColorMapButton matlab.ui.control.StateButton
        LiveMeasColorMinEditField matlab.ui.control.NumericEditField
        LiveMeasColorMinLabel matlab.ui.control.Label
        LiveMeasColorMaxEditField matlab.ui.control.NumericEditField
        LiveMeasColorMaxLabel matlab.ui.control.Label
        LiveMeasNoOfImEditField matlab.ui.control.NumericEditField
        LiveMeasNoOfImLabel      matlab.ui.control.Label

        %% define Evalutation Result Properties
        EvalResTab   matlab.ui.container.Tab
        EvalResTabGridLayout matlab.ui.container.GridLayout
        EvalResMeasDataPanel           matlab.ui.container.Panel
        EvalResMeasDataGridLayout matlab.ui.container.GridLayout
        EvalResMeasDataParamsGridLayout matlab.ui.container.GridLayout

        EvalResMeasData_EvalNorm matlab.ui.control.EditField
        EvalResMeasData_EvalNorm_Label matlab.ui.control.Label
        EvalResMeasData_BackCorrMethod matlab.ui.control.EditField
        EvalResMeasData_BackCorrMethod_Label matlab.ui.control.Label
        EvalResMeasData_Lambda matlab.ui.control.NumericEditField
        EvalResMeasData_Lambda_Label matlab.ui.control.Label
        EvalResMeasData_ConvCrit matlab.ui.control.NumericEditField
        EvalResMeasData_ConvCrit_Label matlab.ui.control.Label
        EvalResMeasData_IntegAreaFactor matlab.ui.control.NumericEditField
        EvalResMeasData_IntegAreaFactor_Label matlab.ui.control.Label
        EvalResMeasData_Nt matlab.ui.control.NumericEditField
        EvalResMeasData_Nt_Label matlab.ui.control.Label
        EvalResMeasData_KernelSizeInPercent matlab.ui.control.NumericEditField
```

```
196         EvalResMeasData_KernelSizeInPercent_Label matlab.ui.control.Label
197
198         EvalResMeasData_z_0_X matlab.ui.control.NumericEditField
199         EvalResMeasData_z_0_X_Label matlab.ui.control.Label
200         EvalResMeasData_d_0_X matlab.ui.control.NumericEditField
201         EvalResMeasData_d_0_X_Label matlab.ui.control.Label
202         EvalResMeasData_theta_X matlab.ui.control.NumericEditField
203         EvalResMeasData_theta_X_Label matlab.ui.control.Label
204         EvalResMeasData_z_R_X matlab.ui.control.NumericEditField
205         EvalResMeasData_z_R_X_Label matlab.ui.control.Label
206         EvalResMeasData_M_squared_X matlab.ui.control.NumericEditField
207         EvalResMeasData_M_squared_X_Label matlab.ui.control.Label
208
209         EvalResMeasData_z_0_Y matlab.ui.control.NumericEditField
210         EvalResMeasData_z_0_Y_Label matlab.ui.control.Label
211         EvalResMeasData_d_0_Y matlab.ui.control.NumericEditField
212         EvalResMeasData_d_0_Y_Label matlab.ui.control.Label
213         EvalResMeasData_theta_Y matlab.ui.control.NumericEditField
214         EvalResMeasData_theta_Y_Label matlab.ui.control.Label
215         EvalResMeasData_z_R_Y matlab.ui.control.NumericEditField
216         EvalResMeasData_z_R_Y_Label matlab.ui.control.Label
217         EvalResMeasData_M_squared_Y matlab.ui.control.NumericEditField
218         EvalResMeasData_M_squared_Y_Label matlab.ui.control.Label
219         EvalResMeasData_M_squared_eff matlab.ui.control.NumericEditField
220         EvalResMeasData_M_squared_eff_Label matlab.ui.control.Label
221
222         EvalResMeasLoadButton   matlab.ui.control.Button
223         EvalResMeasDataResultDropDown_Label matlab.ui.control.Label
224         EvalResMeasDataResultDropDown    matlab.ui.control.DropDown
225
226         EvalResSaveMeasButton   matlab.ui.control.Button
227         EvalResExportMeasButton   matlab.ui.control.Button
228         EvalResMeasPlotButton   matlab.ui.control.Button
229         EvalResHoldPlotButton   matlab.ui.control.StateButton
230         EvalResPlotChooserDropDown    matlab.ui.control.DropDown
231
232         EvalResSmpDataPanel           matlab.ui.container.Panel
233         EvalResSmpDataGridLayout matlab.ui.container.GridLayout
234         EvalResSmpDataParamsGridLayout matlab.ui.container.GridLayout
235         EvalResSmpData_W_X matlab.ui.control.NumericEditField
236         EvalResSmpData_W_X_Label matlab.ui.control.Label
237         EvalResSmpData_W_X_std matlab.ui.control.NumericEditField
238         EvalResSmpData_W_X_std_Label matlab.ui.control.Label
239         EvalResSmpData_W_Y matlab.ui.control.NumericEditField
240         EvalResSmpData_W_Y_Label matlab.ui.control.Label
241         EvalResSmpData_W_Y_std matlab.ui.control.NumericEditField
242         EvalResSmpData_W_Y_std_Label matlab.ui.control.Label
243         EvalResSmpData_W_X_squared matlab.ui.control.NumericEditField
244         EvalResSmpData_W_X_squared_Label matlab.ui.control.Label
```

```
        EvalResSmpData_W_X_squared_std matlab.ui.control.NumericEditField
        EvalResSmpData_W_X_squared_std_Label matlab.ui.control.Label
        EvalResSmpData_W_Y_squared matlab.ui.control.NumericEditField
        EvalResSmpData_W_Y_squared_Label matlab.ui.control.Label
        EvalResSmpData_W_Y_squared_std matlab.ui.control.NumericEditField
        EvalResSmpData_W_Y_squared_std_Label matlab.ui.control.Label
        EvalResSmpData_W_XY matlab.ui.control.NumericEditField
        EvalResSmpData_W_XY_Label matlab.ui.control.Label
        EvalResSmpData_W_XY_std matlab.ui.control.NumericEditField
        EvalResSmpData_W_XY_std_Label matlab.ui.control.Label
        EvalResSmpData_dWx matlab.ui.control.NumericEditField
        EvalResSmpData_dWx_Label matlab.ui.control.Label
        EvalResSmpData_dWx_std matlab.ui.control.NumericEditField
        EvalResSmpData_dWx_std_Label matlab.ui.control.Label
        EvalResSmpData_dWy matlab.ui.control.NumericEditField
        EvalResSmpData_dWy_Label matlab.ui.control.Label
        EvalResSmpData_dWy_std matlab.ui.control.NumericEditField
        EvalResSmpData_dWy_std_Label matlab.ui.control.Label
        EvalResSmpData_phi matlab.ui.control.NumericEditField
        EvalResSmpData_phi_Label matlab.ui.control.Label
        EvalResSmpData_phi_std matlab.ui.control.NumericEditField
        EvalResSmpData_phi_std_Label matlab.ui.control.Label
        EvalResSmpDataDropDown_Label matlab.ui.control.Label
        EvalResSmpDataDropDown    matlab.ui.control.DropDown
        EvalResSmpDataUIAxes  matlab.ui.control.UIAxes
        EvalResSmpInfoTextArea matlab.ui.control.TextArea

        EvalResImDataPanel           matlab.ui.container.Panel
        EvalResImDataGridLayout matlab.ui.container.GridLayout
        EvalResImDataParamsGridLayout matlab.ui.container.GridLayout
        EvalResImData_W_X matlab.ui.control.NumericEditField
        EvalResImData_W_X_Label matlab.ui.control.Label
        EvalResImData_W_Y matlab.ui.control.NumericEditField
        EvalResImData_W_Y_Label matlab.ui.control.Label
        EvalResImData_W_X_squared matlab.ui.control.NumericEditField;
        EvalResImData_W_X_squared_Label matlab.ui.control.Label
        EvalResImData_W_Y_squared matlab.ui.control.NumericEditField;
        EvalResImData_W_Y_squared_Label matlab.ui.control.Label
        EvalResImData_W_XY matlab.ui.control.NumericEditField
        EvalResImData_W_XY_Label matlab.ui.control.Label
        EvalResImData_dWx matlab.ui.control.NumericEditField
        EvalResImData_dWx_Label matlab.ui.control.Label
        EvalResImData_dWy matlab.ui.control.NumericEditField
        EvalResImData_dWy_Label matlab.ui.control.Label
        EvalResImData_phi matlab.ui.control.NumericEditField
        EvalResImData_phi_Label matlab.ui.control.Label
        EvalResImDataDropDown_Label matlab.ui.control.Label
        EvalResImDataDropDown    matlab.ui.control.DropDown
        EvalResImDataUIAxes  matlab.ui.control.UIAxes
```

```
        EvalResImDataColorGridLayout matlab.ui.container.GridLayout
        EvalResImDataChangeColorMapButton matlab.ui.control.StateButton
        EvalResColorMinEditField matlab.ui.control.NumericEditField
        EvalResColorMinLabel matlab.ui.control.Label
        EvalResColorMaxEditField matlab.ui.control.NumericEditField
        EvalResColorMaxLabel matlab.ui.control.Label
        %% define additional necessery Properties and variables needed for GUI
        % Rectangle Object
        ConfEvalRectRoi;
        % Point Object
        ConfEvalPointRoi;
        % Rectangle Object
        LiveMeasRectRoi;
        % Point Object
        LiveMeasPointRoi;
        loadedMeas Measurement;
        ConfEvalSelectedArea;
        LiveMeasSelectedArea
        fontsize14 double = 14;
        %Arduino Stepmotor
        arduinoUNO;
        arduinoUNOShield;
        arduinoUNOShieldStepmotor;
        currPos double;
        currPosRoi;
        currPosRoiAutoMeas;
        maxValueStage;
        StepInkrement double;
        measurementPoints;
        previewMarkers;
        centerMarker;
        endPointMarkers;
        evaluationInProgress logical;
        LiveMeas Measurement;
        hLinkCurrPos;
        %Camera class videoinput
        vid
        LiveMeasColorMap;
        EvalResColorMap;
    end
    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)
            backGroundColorLabel = [0.75 0.75 0.75];
            app.LiveMeasColorMap = [0 255];
            app.EvalResColorMap = [0 255];
            %% Create UIFigure and hide until all components are created
```

```
343             app.UIFigure = uifigure('Visible', 'off');
344             %%           app.UIFigure.Position = [size(1) size(2) size(3) size(4)];
345             app.UIFigure.Name = 'lbc App';
346             app.UIFigure.WindowState = 'maximized';
347             app.UIFigure.Visible = 'on';
348             %%              app.UIFigure.Resize = 'off';
349             app.UIFigure.AutoResizeChildren = 'on';
350             % Setting Stage Variables
351             app.StepInkrement = 1.8;
352             app.maxValueStage = 23400;
353             app.currPos = [0 0];
354
355             %% Create TabGroup
356             app.TabGroup = uitabgroup(app.UIFigure);
357             app.TabGroup.Units = 'normalized';
358             app.TabGroup.Position = [0 0 1 1];
359
360             %% Create HardwareSetupTab
361             app.HardwareSetupTab = uitab(app.TabGroup);
362             app.HardwareSetupTab.Title = 'Hardware Setup';
363             app.HardwareSetupTab.Scrollable = 'on';
364
365             %% Create HSGridLayout
366             app.HSGridLayout = uigridlayout(app.HardwareSetupTab);
367             app.HSGridLayout.RowHeight = {'1x'};
368             app.HSGridLayout.ColumnWidth = {'1x','1x','1x'};
369
370             %% Create HSCamConfPanel
371             app.HSCamConfPanel = uipanel(app.HSGridLayout);
372             app.HSCamConfPanel.Layout.Row = 1;
373             app.HSCamConfPanel.Layout.Column = 1;
374
375             %% Create HSCamConfGridLayout
376             app.HSCamConfGridLayout = uigridlayout(app.HSCamConfPanel);
377             app.HSCamConfGridLayout.RowHeight = {'1x','1x','1x','1x','1x','1x','1x',
                    '1x','1x','1x','1x'};
378             app.HSCamConfGridLayout.ColumnWidth = {'1x'};
379
380             %% Create HSCamConfLabel
381             app.HSCamConfLabel = uilabel(app.HSCamConfGridLayout);
382             app.HSCamConfLabel.Layout.Row = 1;
383             app.HSCamConfLabel.Layout.Column = [1 3];
384             app.HSCamConfLabel.Text = 'Configure Camera';
385             app.HSCamConfLabel.FontSize = app.fontsize14;
386             app.HSCamConfLabel.Interpreter = 'latex';
387             app.HSCamConfLabel.BackgroundColor = backGroundColorLabel;
388             app.HSCamConfLabel.HorizontalAlignment = 'center';
389
390             %% Create HSCamConfConnectButton
```

```
            app.HSCamConfConnectButton = uibutton(app.HSCamConfGridLayout);
            app.HSCamConfConnectButton.Text = 'Connect';
            app.HSCamConfConnectButton.Layout.Row = 2;
            app.HSCamConfConnectButton.Layout.Column = 2;
            app.HSCamConfConnectButton.HorizontalAlignment = 'center';
            app.HSCamConfConnectButton.FontSize = app.fontsize14;
            app.HSCamConfConnectButton.ButtonPushedFcn = @(src, event)
                HSCamConfConnectButton_ButtonPushedFcn(app, src, event);
            %% Create HSCamConfDisconnectButton
            app.HSCamConfDisconnectButton = uibutton(app.HSCamConfGridLayout);
            app.HSCamConfDisconnectButton.Text = 'Disconnect';
            app.HSCamConfDisconnectButton.Layout.Row = 2;
            app.HSCamConfDisconnectButton.Layout.Column = 3;
            app.HSCamConfDisconnectButton.HorizontalAlignment = 'center';
            app.HSCamConfDisconnectButton.FontSize = app.fontsize14;
            app.HSCamConfDisconnectButton.Enable = 'off';
            app.HSCamConfDisconnectButton.ButtonPushedFcn = @(src, event)
                HSCamConfDisconnectButton_ButtonPushedFcn(app, src, event);
            %% Create HSCamConfConnectionLabel
            app.HSCamConfConnectionLabel = uilabel(app.HSCamConfGridLayout);
            app.HSCamConfConnectionLabel.Layout.Row = 2;
            app.HSCamConfConnectionLabel.Layout.Column = 1;
            app.HSCamConfConnectionLabel.FontSize = app.fontsize14;
            app.HSCamConfConnectionLabel.Interpreter = 'latex';
            app.HSCamConfConnectionLabel.BackgroundColor = [1 0.07 0.2];
            app.HSCamConfConnectionLabel.HorizontalAlignment = 'center';
            app.HSCamConfConnectionLabel.Text = 'Disconnected';
            %% Create HSCamConfCamUIAxes
            app.HSCamConfCamUIAxes = uiaxes(app.HSCamConfGridLayout);
            app.HSCamConfCamUIAxes.Layout.Row = [5 10];
            app.HSCamConfCamUIAxes.Layout.Column = [1 3];
            app.HSCamConfCamUIAxes.Visible = 'off';

            %% Create HSCamConfCamTestButton
            app.HSCamConfCamTestButton = uibutton(app.HSCamConfGridLayout);
            app.HSCamConfCamTestButton.Text = 'Trigger Image';
            app.HSCamConfCamTestButton.HorizontalAlignment = 'center';
            app.HSCamConfCamTestButton.FontSize = app.fontsize14;
            app.HSCamConfCamTestButton.Layout.Row = 4;
            app.HSCamConfCamTestButton.Layout.Column = 1;
            app.HSCamConfCamTestButton.ButtonPushedFcn = @(src, event)
                HSCamConfCamTestButton_ButtonPushedFcn(app, src, event);
            app.HSCamConfCamTestButton.Enable = 'off';

            %% Create HSCamConfExposureTimeLabel
            app.HSCamConfExposureTimeLabel = uilabel(app.HSCamConfGridLayout);
            app.HSCamConfExposureTimeLabel.Layout.Row = 3;
            app.HSCamConfExposureTimeLabel.Layout.Column = 1;
            app.HSCamConfExposureTimeLabel.Text = '$\textrm{Exposure Time in } \mu s
```

```
            $';
437         app.HSCamConfExposureTimeLabel.FontSize = app.fontsize14;
438         app.HSCamConfExposureTimeLabel.Interpreter = 'latex';
439         app.HSCamConfExposureTimeLabel.BackgroundColor = backGroundColorLabel;
440         app.HSCamConfExposureTimeLabel.HorizontalAlignment = 'center';
441
442         %% Create HSCamConfExposureTimeEditField
443         app.HSCamConfExposureTimeEditField = uieditfield(app.HSCamConfGridLayout
                ,'numeric');
444         app.HSCamConfExposureTimeEditField.Layout.Row = 3;
445         app.HSCamConfExposureTimeEditField.Layout.Column = 2;
446         app.HSCamConfExposureTimeEditField.HorizontalAlignment = 'center';
447         app.HSCamConfExposureTimeEditField.FontSize = app.fontsize14;
448         app.HSCamConfExposureTimeEditField.Enable = 'off';
449         %% Create HSCamConfApplySettingButton
450         app.HSCamConfApplySettingButton = uibutton(app.HSCamConfGridLayout);
451         app.HSCamConfApplySettingButton.Text = 'Apply';
452         app.HSCamConfApplySettingButton.HorizontalAlignment = 'center';
453         app.HSCamConfApplySettingButton.FontSize = app.fontsize14;
454         app.HSCamConfApplySettingButton.Layout.Row = 3;
455         app.HSCamConfApplySettingButton.Layout.Column = 3;
456         app.HSCamConfApplySettingButton.ButtonPushedFcn = @(src, event)
                HSCamConfApplySettingButton_ButtonPushedFcn(app, src, event);
457         app.HSCamConfApplySettingButton.Enable = 'off';
458         %% Create HSCamConfCamPreviewButton
459         app.HSCamConfCamPreviewButton = uibutton(app.HSCamConfGridLayout,'state'
                );
460         app.HSCamConfCamPreviewButton.Text = 'Toggle Preview';
461         app.HSCamConfCamPreviewButton.HorizontalAlignment = 'center';
462         app.HSCamConfCamPreviewButton.FontSize = app.fontsize14;
463         app.HSCamConfCamPreviewButton.Layout.Row = 4;
464         app.HSCamConfCamPreviewButton.Layout.Column = 2;
465         app.HSCamConfCamPreviewButton.ValueChangedFcn = @(src, event)
                HSCamConfCamPreviewButton_ValueChangedFcn(app, src, event);
466         app.HSCamConfCamPreviewButton.Enable = 'off';
467
468         %% Create HSStepmotorConfPanel
469         app.HSStepmotorConfPanel = uipanel(app.HSGridLayout);
470         app.HSStepmotorConfPanel.Layout.Row = 1;
471         app.HSStepmotorConfPanel.Layout.Column = 2;
472         %% Create HSStepmotorConfGridLayout
473         app.HSStepmotorConfGridLayout = uigridlayout(app.HSStepmotorConfPanel);
474         app.HSStepmotorConfGridLayout.RowHeight = {'1x','1x','1x','1x','1x','1x'
                ,'1x','1x','1x','1x'};
475         app.HSStepmotorConfGridLayout.ColumnWidth = {'1x','1x','1x'};
476         %% Create HSStepmotorConfLabel
477         app.HSStepmotorConfLabel = uilabel(app.HSStepmotorConfGridLayout);
478         app.HSStepmotorConfLabel.Layout.Row = 1;
479         app.HSStepmotorConfLabel.Layout.Column = [1 3];
```

```
480             app.HSStepmotorConfLabel.Text = 'Configure Stepmotor';
481             app.HSStepmotorConfLabel.FontSize = app.fontsize14;
482             app.HSStepmotorConfLabel.Interpreter = 'latex';
483             app.HSStepmotorConfLabel.BackgroundColor = backGroundColorLabel;
484             app.HSStepmotorConfLabel.HorizontalAlignment = 'center';
485             %% Create HSStepmotorConfComPortLabel
486             app.HSStepmotorConfComPortLabel = uilabel(app.HSStepmotorConfGridLayout)
                    ;
487             app.HSStepmotorConfComPortLabel.Layout.Row = 2;
488             app.HSStepmotorConfComPortLabel.Layout.Column = 1;
489             app.HSStepmotorConfComPortLabel.Text = 'Select COM-Port';
490             app.HSStepmotorConfComPortLabel.FontSize = app.fontsize14;
491             app.HSStepmotorConfComPortLabel.Interpreter = 'latex';
492             app.HSStepmotorConfComPortLabel.BackgroundColor = backGroundColorLabel;
493             app.HSStepmotorConfComPortLabel.HorizontalAlignment = 'center';
494             %% Create HSStepmotorConfComPortRefreshButton
495             app.HSStepmotorConfComPortRefreshButton = uibutton(app.
                    HSStepmotorConfGridLayout);
496             app.HSStepmotorConfComPortRefreshButton.Text = 'Refresh';
497             app.HSStepmotorConfComPortRefreshButton.HorizontalAlignment = 'center';
498             app.HSStepmotorConfComPortRefreshButton.FontSize = app.fontsize14;
499             app.HSStepmotorConfComPortRefreshButton.Layout.Row = 2;
500             app.HSStepmotorConfComPortRefreshButton.Layout.Column = 2;
501             app.HSStepmotorConfComPortRefreshButton.ButtonPushedFcn = @(src, event)
                    HSStepmotorConfComPortRefreshButton_ButtonPushedFcn(app, src, event)
                    ;
502             %% Create HSStepmotorConfComPortDropDown
503             app.HSStepmotorConfComPortDropDown = uidropdown(app.
                    HSStepmotorConfGridLayout);
504             app.HSStepmotorConfComPortDropDown.Layout.Row = 2;
505             app.HSStepmotorConfComPortDropDown.Layout.Column = 3;
506             app.HSStepmotorConfComPortDropDown.Items = {};
507             app.HSStepmotorConfComPortDropDown.FontSize = app.fontsize14;
508             %% Create HSStepmotorConfConnectButton
509             app.HSStepmotorConfConnectButton = uibutton(app.
                    HSStepmotorConfGridLayout);
510             app.HSStepmotorConfConnectButton.Text = 'Connect';
511             app.HSStepmotorConfConnectButton.HorizontalAlignment = 'center';
512             app.HSStepmotorConfConnectButton.FontSize = app.fontsize14;
513             app.HSStepmotorConfConnectButton.Layout.Row = 3;
514             app.HSStepmotorConfConnectButton.Layout.Column = 2;
515             app.HSStepmotorConfConnectButton.ButtonPushedFcn = @(src, event)
                    HSStepmotorConfConnectButton_ButtonPushedFcn(app, src, event);
516             %% Create HSStepmotorConfDisconnectButton
517             app.HSStepmotorConfDisconnectButton = uibutton(app.
                    HSStepmotorConfGridLayout);
518             app.HSStepmotorConfDisconnectButton.Text = 'Disconnect';
519             app.HSStepmotorConfDisconnectButton.Layout.Row = 3;
520             app.HSStepmotorConfDisconnectButton.Layout.Column = 3;
```

```
521             app.HSStepmotorConfDisconnectButton.HorizontalAlignment = 'center';
522             app.HSStepmotorConfDisconnectButton.FontSize = app.fontsize14;
523             app.HSStepmotorConfDisconnectButton.Enable = 'off';
524             app.HSStepmotorConfDisconnectButton.ButtonPushedFcn = @(src, event)
                    HSStepmotorConfDisconnectButton_ButtonPushedFcn(app, src, event);
525             %% Create HSStepmotorConfConnectionLabel
526             app.HSStepmotorConfConnectionLabel = uilabel(app.
                    HSStepmotorConfGridLayout);
527             app.HSStepmotorConfConnectionLabel.Layout.Row = 3;
528             app.HSStepmotorConfConnectionLabel.Layout.Column = 1;
529             app.HSStepmotorConfConnectionLabel.FontSize = app.fontsize14;
530             app.HSStepmotorConfConnectionLabel.Interpreter = 'latex';
531             app.HSStepmotorConfConnectionLabel.BackgroundColor = [1 0.07 0.2];
532             app.HSStepmotorConfConnectionLabel.HorizontalAlignment = 'center';
533             app.HSStepmotorConfConnectionLabel.Text = 'Disconnected';
534             %% Create HSStepmotorConfStepsPerRevolutionLabel
535             app.HSStepmotorConfStepsPerRevolutionLabel = uilabel(app.
                    HSStepmotorConfGridLayout);
536             app.HSStepmotorConfStepsPerRevolutionLabel.Layout.Row = 4;
537             app.HSStepmotorConfStepsPerRevolutionLabel.Layout.Column = [1 2];
538             app.HSStepmotorConfStepsPerRevolutionLabel.Text = 'Steps Per Revolution'
                    ;
539             app.HSStepmotorConfStepsPerRevolutionLabel.FontSize = app.fontsize14;
540             app.HSStepmotorConfStepsPerRevolutionLabel.Interpreter = 'latex';
541             app.HSStepmotorConfStepsPerRevolutionLabel.BackgroundColor =
                    backGroundColorLabel;
542             app.HSStepmotorConfStepsPerRevolutionLabel.HorizontalAlignment = 'center
                    ';
543             %% Create HSStepmotorConfStepsPerRevolutionEditField
544             app.HSStepmotorConfStepsPerRevolutionEditField = uieditfield(app.
                    HSStepmotorConfGridLayout,'numeric');
545             app.HSStepmotorConfStepsPerRevolutionEditField.Layout.Row = 4;
546             app.HSStepmotorConfStepsPerRevolutionEditField.Layout.Column = 3;
547             app.HSStepmotorConfStepsPerRevolutionEditField.HorizontalAlignment = '
                    center';
548             app.HSStepmotorConfStepsPerRevolutionEditField.FontSize = app.fontsize14
                    ;
549             app.HSStepmotorConfStepsPerRevolutionEditField.Value = 200;
550             app.HSStepmotorConfStepsPerRevolutionEditField.Enable = 'off';
551             %% Create HSStepmotorConfRpmLabel
552             app.HSStepmotorConfRpmLabel = uilabel(app.HSStepmotorConfGridLayout);
553             app.HSStepmotorConfRpmLabel.Layout.Row = 5;
554             app.HSStepmotorConfRpmLabel.Layout.Column = [1 2];
555             app.HSStepmotorConfRpmLabel.Text = 'Revolutions Per Minute (RPM)';
556             app.HSStepmotorConfRpmLabel.FontSize = app.fontsize14;
557             app.HSStepmotorConfRpmLabel.Interpreter = 'latex';
558             app.HSStepmotorConfRpmLabel.BackgroundColor = backGroundColorLabel;
559             app.HSStepmotorConfRpmLabel.HorizontalAlignment = 'center';
560             %% Create HSStepmotorConfRpmEditField
```

```
561             app.HSStepmotorConfRpmEditField = uieditfield(app.
                    HSStepmotorConfGridLayout,'numeric');
562             app.HSStepmotorConfRpmEditField.Layout.Row = 5;
563             app.HSStepmotorConfRpmEditField.Layout.Column = 3;
564             app.HSStepmotorConfRpmEditField.HorizontalAlignment = 'center';
565             app.HSStepmotorConfRpmEditField.FontSize = app.fontsize14;
566             app.HSStepmotorConfRpmEditField.Value = 10;
567             app.HSStepmotorConfRpmEditField.Enable = 'off';
568             %% Create HSStepmotorConfApplySettingsButton
569             app.HSStepmotorConfApplySettingsButton = uibutton(app.
                    HSStepmotorConfGridLayout);
570             app.HSStepmotorConfApplySettingsButton.Text = 'Apply Settings';
571             app.HSStepmotorConfApplySettingsButton.HorizontalAlignment = 'center';
572             app.HSStepmotorConfApplySettingsButton.FontSize = app.fontsize14;
573             app.HSStepmotorConfApplySettingsButton.Layout.Row = 6;
574             app.HSStepmotorConfApplySettingsButton.Layout.Column = [2 3];
575             app.HSStepmotorConfApplySettingsButton.ButtonPushedFcn = @(src, event)
                    HSStepmotorConfApplySettingsButton_ButtonPushedFcn(app, src, event);
576             app.HSStepmotorConfApplySettingsButton.Enable = 'off';
577             %% Create HSTestHardwarePanel
578             app.HSTestHardwarePanel = uipanel(app.HSGridLayout);
579             app.HSTestHardwarePanel.Layout.Row = 1;
580             app.HSTestHardwarePanel.Layout.Column = 3;
581             %% Create HSTestHardwareGridLayout
582             app.HSTestHardwareGridLayout = uigridlayout(app.HSTestHardwarePanel);
583             app.HSTestHardwareGridLayout.RowHeight = {'1x','1x','1x','1x','1x','1x',
                    '1x','1x','1x','1x'};
584             app.HSTestHardwareGridLayout.ColumnWidth = {'1x','1x','1x'};
585             %% Create HSTestHardwareReferenceStageButton
586             app.HSTestHardwareReferenceStageButton = uibutton(app.
                    HSTestHardwareGridLayout);
587             app.HSTestHardwareReferenceStageButton.Text = 'Reference Stage';
588             app.HSTestHardwareReferenceStageButton.HorizontalAlignment = 'center';
589             app.HSTestHardwareReferenceStageButton.FontSize = app.fontsize14;
590             app.HSTestHardwareReferenceStageButton.Layout.Row = 1;
591             app.HSTestHardwareReferenceStageButton.Layout.Column = [1 3];
592             app.HSTestHardwareReferenceStageButton.ButtonPushedFcn = @(src, event)
                    HSTestHardwareReferenceStageButton_ButtonPushedFcn(app, src, event);
593             %% Create HSTestHardwareGoToZeroButton
594             app.HSTestHardwareGoToZeroButton = uibutton(app.HSTestHardwareGridLayout
                    );
595             app.HSTestHardwareGoToZeroButton.Text = 'Go to Zero';
596             app.HSTestHardwareGoToZeroButton.HorizontalAlignment = 'center';
597             app.HSTestHardwareGoToZeroButton.FontSize = app.fontsize14;
598             app.HSTestHardwareGoToZeroButton.Layout.Row = 2;
599             app.HSTestHardwareGoToZeroButton.Layout.Column = 3;
600             app.HSTestHardwareGoToZeroButton.ButtonPushedFcn = @(src, event)
                    HSTestHardwareGoToZeroButton_ButtonPushedFcn(app, src, event);
601             %% Create HSTestHardwareGoToPositionLabel
```

```
602             app.HSTestHardwareGoToPositionLabel = uilabel(app.
                    HSTestHardwareGridLayout);
603             app.HSTestHardwareGoToPositionLabel.Layout.Row = 4;
604             app.HSTestHardwareGoToPositionLabel.Layout.Column = 1;
605             app.HSTestHardwareGoToPositionLabel.Text = 'Go To Position';
606             app.HSTestHardwareGoToPositionLabel.FontSize = app.fontsize14;
607             app.HSTestHardwareGoToPositionLabel.Interpreter = 'latex';
608             app.HSTestHardwareGoToPositionLabel.BackgroundColor =
                    backGroundColorLabel;
609             app.HSTestHardwareGoToPositionLabel.HorizontalAlignment = 'center';
610             %% Create HSTestHardwareGoToPositionEditField
611             app.HSTestHardwareGoToPositionEditField = uieditfield(app.
                    HSTestHardwareGridLayout,'numeric');
612             app.HSTestHardwareGoToPositionEditField.Layout.Row = 4;
613             app.HSTestHardwareGoToPositionEditField.Layout.Column = 2;
614             %% app.HSTestHardwareGoToPositionEditField.ValueChangedFcn = @(src,
                    event) ConfEvalWavelengthSlider_AllEvents(app, src, event);
615             app.HSTestHardwareGoToPositionEditField.HorizontalAlignment = 'center';
616             app.HSTestHardwareGoToPositionEditField.FontSize = app.fontsize14;
617             app.HSTestHardwareGoToPositionEditField.Value = 10;
618             %% Create HSTestHardwareGoToPositionButton
619             app.HSTestHardwareGoToPositionButton = uibutton(app.
                    HSTestHardwareGridLayout);
620             app.HSTestHardwareGoToPositionButton.Text = 'GO';
621             app.HSTestHardwareGoToPositionButton.HorizontalAlignment = 'center';
622             app.HSTestHardwareGoToPositionButton.FontSize = app.fontsize14;
623             app.HSTestHardwareGoToPositionButton.Layout.Row = 4;
624             app.HSTestHardwareGoToPositionButton.Layout.Column = 3;
625             app.HSTestHardwareGoToPositionButton.ButtonPushedFcn = @(src, event)
                    HSTestHardwareGoToPositionButton_ButtonPushedFcn(app, src, event);
626              %% Create AutoMeasISO111146Part1StatusLabel
627             app.HSTestHardwareMotorStatusLabel = uilabel(app.
                    HSTestHardwareGridLayout);
628             app.HSTestHardwareMotorStatusLabel.Layout.Row = 3;
629             app.HSTestHardwareMotorStatusLabel.Layout.Column = 1;
630             app.HSTestHardwareMotorStatusLabel.FontSize = app.fontsize14;
631             app.HSTestHardwareMotorStatusLabel.Interpreter = 'latex';
632             app.HSTestHardwareMotorStatusLabel.BackgroundColor = [1 0.07 0.2];
633             app.HSTestHardwareMotorStatusLabel.HorizontalAlignment = 'center';
634             app.HSTestHardwareMotorStatusLabel.Text = 'Motor Off';
635             %% Create HSTestHardwarePositionUIAxes
636             app.HSTestHardwarePositionUIAxes = uiaxes(app.HSTestHardwareGridLayout);
637             app.HSTestHardwarePositionUIAxes.Layout.Row = [5 7];
638             app.HSTestHardwarePositionUIAxes.Layout.Column = [1 3];
639             app.HSTestHardwarePositionUIAxes.XLim = [0—250 app.maxValueStage+250];
640             app.HSTestHardwarePositionUIAxes.YLim = [—0.5 0.5];
641             app.HSTestHardwarePositionUIAxes.XAxisLocation = 'origin';
642             app.HSTestHardwarePositionUIAxes.Interactions = [];
643             app.HSTestHardwarePositionUIAxes.Toolbar.Visible = 'off';
```

```
644             app.HSTestHardwarePositionUIAxes.YTickLabel = [];
645             app.HSTestHardwarePositionUIAxes.Title.String = 'Camera z — Position';
646             app.HSTestHardwarePositionUIAxes.Title.Interpreter = 'latex';
647             xtickangle(app.HSTestHardwarePositionUIAxes,45);
648             drawnow;
649             %% Create currPosRoi
650             app.currPosRoi = drawpoint(app.HSTestHardwarePositionUIAxes,'Position',
                    app.currPos,'Color','yellow');
651             app.currPosRoi.InteractionsAllowed = 'none';
652             %% Create HSTestHardwareCurrPosLabel
653             app.HSTestHardwareCurrPosLabel = uilabel(app.HSTestHardwareGridLayout);
654             app.HSTestHardwareCurrPosLabel.Layout.Row = 8;
655             app.HSTestHardwareCurrPosLabel.Layout.Column = 1;
656             app.HSTestHardwareCurrPosLabel.FontSize = app.fontsize14;
657             app.HSTestHardwareCurrPosLabel.Interpreter = 'latex';
658             app.HSTestHardwareCurrPosLabel.BackgroundColor = backGroundColorLabel;
659             app.HSTestHardwareCurrPosLabel.HorizontalAlignment = 'center';
660             app.HSTestHardwareCurrPosLabel.Text = 'Current Position';
661             %% Create HSTestHardwareCurrPosEditField
662             app.HSTestHardwareCurrPosEditField = uieditfield(app.
                    HSTestHardwareGridLayout,'numeric');
663             app.HSTestHardwareCurrPosEditField.Layout.Row = 8;
664             app.HSTestHardwareCurrPosEditField.Layout.Column = 2;
665             app.HSTestHardwareCurrPosEditField.HorizontalAlignment = 'center';
666             app.HSTestHardwareCurrPosEditField.Editable = 'off';
667             %% Create AutomatedMeasTab
668             app.AutoMeasTab = uitab(app.TabGroup);
669             app.AutoMeasTab.Title = 'Automated Measurement';
670             app.AutoMeasTab.Scrollable = 'on';
671             %% Create AutoMeasTabGroup
672             app.AutoMeasTabGroup = uitabgroup(app.AutoMeasTab);
673             app.AutoMeasTabGroup.Units = 'normalized';
674             app.AutoMeasTabGroup.Position = [0 0 1 1];
675             app.AutoMeasTabGroup.TabLocation = 'bottom';
676             %% Create AutoMeasISO111146Part1Tab
677             app.AutoMeasISO111146Part1Tab = uitab(app.AutoMeasTabGroup);
678             app.AutoMeasISO111146Part1Tab.Title = 'ISO11146—1';
679             app.AutoMeasISO111146Part1Tab.Scrollable = 'on';
680             %% Create AutoMeasISO111146Part1TabGridLayout
681             app.AutoMeasISO111146Part1TabGridLayout = uigridlayout(app.
                    AutoMeasISO111146Part1Tab);
682             app.AutoMeasISO111146Part1TabGridLayout.RowHeight = {'1x','1x','1x','1x'
                    ,'1x',3,'1x','1x','1x','1x','1x'};
683             app.AutoMeasISO111146Part1TabGridLayout.ColumnWidth = {'1x','1x','1x','1
                    x','1x','1x','1x','1x','1x','1x','1x','1x','1x','1x','1x','1x'};
684
685             %% Create AutoMeasISO111146Part1MethodRBGroup
686             app.AutoMeasISO111146Part1MethodRBGroup = uibuttongroup(app.
                    AutoMeasISO111146Part1TabGridLayout);
```

```
687             app.AutoMeasISO111146Part1MethodRBGroup.Layout.Row = 1;
688             app.AutoMeasISO111146Part1MethodRBGroup.Layout.Column = [14 15];
689             app.AutoMeasISO111146Part1MethodRBGroup.SelectionChangedFcn = @(src,
                    event) AutoMeasISO111146Part1MethodRBGroup_SelectionChangedFcn(app,
                    src, event);
690             %% Create AutoMeasISO111146Part1FromCenterRB
691             app.AutoMeasISO111146Part1FromCenterRB =  uiradiobutton(app.
                    AutoMeasISO111146Part1MethodRBGroup,'Position',[10 35 200 15]);
692             app.AutoMeasISO111146Part1FromCenterRB.Text = 'Configure around
                    centerpoint';
693             %% Create AutoMeasISO111146Part1FromZeroRB
694             app.AutoMeasISO111146Part1FromZeroRB =  uiradiobutton(app.
                    AutoMeasISO111146Part1MethodRBGroup,'Position',[10 15 200 15]);
695             app.AutoMeasISO111146Part1FromZeroRB.Text = 'Configure from zero';
696
697             %% Create AutoMeasISO111146Part1OffsetLabel
698             app.AutoMeasISO111146Part1OffsetLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
699             app.AutoMeasISO111146Part1OffsetLabel.Layout.Row = 1;
700             app.AutoMeasISO111146Part1OffsetLabel.Layout.Column = [1 3];
701             app.AutoMeasISO111146Part1OffsetLabel.FontSize = app.fontsize14;
702             app.AutoMeasISO111146Part1OffsetLabel.Interpreter = 'latex';
703             app.AutoMeasISO111146Part1OffsetLabel.BackgroundColor =
                    backGroundColorLabel;
704             app.AutoMeasISO111146Part1OffsetLabel.HorizontalAlignment = 'center';
705             app.AutoMeasISO111146Part1OffsetLabel.Text = '$\textrm{Center Offsetin }
                     \mu m$';
706             %% Create AutoMeasISO111146Part1OffseEditField
707             app.AutoMeasISO111146Part1OffseEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout,'numeric','ValueDisplayFormat','
                    %.1f');
708             app.AutoMeasISO111146Part1OffseEditField.Layout.Row = 1;
709             app.AutoMeasISO111146Part1OffseEditField.Layout.Column = 4;
710             app.AutoMeasISO111146Part1OffseEditField.HorizontalAlignment = 'center';
711             app.AutoMeasISO111146Part1OffseEditField.ValueChangedFcn = @(src, event)
                     UpdatePreviewAxes(app, src, event);
712             app.AutoMeasISO111146Part1OffseEditField.Value = 3000;
713             %% Create AutoMeasISO111146Part1RayleighLabel
714             app.AutoMeasISO111146Part1RayleighLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
715             app.AutoMeasISO111146Part1RayleighLabel.Layout.Row = 2;
716             app.AutoMeasISO111146Part1RayleighLabel.Layout.Column = [1 3];
717             app.AutoMeasISO111146Part1RayleighLabel.FontSize = app.fontsize14;
718             app.AutoMeasISO111146Part1RayleighLabel.Interpreter = 'latex';
719             app.AutoMeasISO111146Part1RayleighLabel.BackgroundColor =
                    backGroundColorLabel;
720             app.AutoMeasISO111146Part1RayleighLabel.HorizontalAlignment = 'center';
721             app.AutoMeasISO111146Part1RayleighLabel.Text = '$\textrm{Rayleigh Length
                     in } \mu m$';
```

```
722             %% Create AutoMeasISO111146Part1RayleighEditField
723             app.AutoMeasISO111146Part1RayleighEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout,'numeric');
724             app.AutoMeasISO111146Part1RayleighEditField.Layout.Row = 2;
725             app.AutoMeasISO111146Part1RayleighEditField.Layout.Column = 4;
726             app.AutoMeasISO111146Part1RayleighEditField.HorizontalAlignment = '
                    center';
727             app.AutoMeasISO111146Part1RayleighEditField.ValueChangedFcn = @(src,
                    event) UpdatePreviewAxes(app, src, event);
728             app.AutoMeasISO111146Part1RayleighEditField.Value = 3000;
729             %% Create AutoMeasISO111146Part1TimesRayleighLabel
730             app.AutoMeasISO111146Part1TimesRayleighLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
731             app.AutoMeasISO111146Part1TimesRayleighLabel.Layout.Row = 3;
732             app.AutoMeasISO111146Part1TimesRayleighLabel.Layout.Column = [1 3];
733             app.AutoMeasISO111146Part1TimesRayleighLabel.FontSize = app.fontsize14;
734             app.AutoMeasISO111146Part1TimesRayleighLabel.Interpreter = 'latex';
735             app.AutoMeasISO111146Part1TimesRayleighLabel.BackgroundColor =
                    backGroundColorLabel;
736             app.AutoMeasISO111146Part1TimesRayleighLabel.HorizontalAlignment = '
                    center';
737             app.AutoMeasISO111146Part1TimesRayleighLabel.Text = 'Number of Rayleigh
                    Lengths$';
738             %% Create AutoMeasISO111146Part1TimesRayleighEditField
739             app.AutoMeasISO111146Part1TimesRayleighEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout,'numeric');
740             app.AutoMeasISO111146Part1TimesRayleighEditField.Layout.Row = 3;
741             app.AutoMeasISO111146Part1TimesRayleighEditField.Layout.Column = 4;
742             app.AutoMeasISO111146Part1TimesRayleighEditField.HorizontalAlignment = '
                    center';
743             app.AutoMeasISO111146Part1TimesRayleighEditField.ValueChangedFcn = @(src
                    , event) UpdatePreviewAxes(app, src, event);
744             app.AutoMeasISO111146Part1TimesRayleighEditField.Value = 4;
745             %% Create AutoMeasISO111146Part1NoMeasPointsLabel
746             app.AutoMeasISO111146Part1NoMeasPointsLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
747             app.AutoMeasISO111146Part1NoMeasPointsLabel.Layout.Row = 4;
748             app.AutoMeasISO111146Part1NoMeasPointsLabel.Layout.Column = [1 3];
749             app.AutoMeasISO111146Part1NoMeasPointsLabel.FontSize = app.fontsize14;
750             app.AutoMeasISO111146Part1NoMeasPointsLabel.Interpreter = 'latex';
751             app.AutoMeasISO111146Part1NoMeasPointsLabel.BackgroundColor =
                    backGroundColorLabel;
752             app.AutoMeasISO111146Part1NoMeasPointsLabel.HorizontalAlignment = '
                    center';
753             app.AutoMeasISO111146Part1NoMeasPointsLabel.Text = 'Number of
                    Measurement Points';
754             %% Create AutoMeasISO111146Part1NoMeasPointsEditField
755             app.AutoMeasISO111146Part1NoMeasPointsEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout,'numeric');
```

```
            app.AutoMeasISO111146Part1NoMeasPointsEditField.Layout.Row = 4;
            app.AutoMeasISO111146Part1NoMeasPointsEditField.Layout.Column = 4;
            app.AutoMeasISO111146Part1NoMeasPointsEditField.HorizontalAlignment = '
                center';
            app.AutoMeasISO111146Part1NoMeasPointsEditField.ValueChangedFcn = @(src,
                 event) UpdatePreviewAxes(app, src, event);
            app.AutoMeasISO111146Part1NoMeasPointsEditField.Value = 3;
            %% Create AutoMeasISO111146Part1MeasPerPointLabel
            app.AutoMeasISO111146Part1MeasPerPointLabel = uilabel(app.
                AutoMeasISO111146Part1TabGridLayout);
            app.AutoMeasISO111146Part1MeasPerPointLabel.Layout.Row = 5;
            app.AutoMeasISO111146Part1MeasPerPointLabel.Layout.Column = [1 3];
            app.AutoMeasISO111146Part1MeasPerPointLabel.FontSize = app.fontsize14;
            app.AutoMeasISO111146Part1MeasPerPointLabel.Interpreter = 'latex';
            app.AutoMeasISO111146Part1MeasPerPointLabel.BackgroundColor =
                backGroundColorLabel;
            app.AutoMeasISO111146Part1MeasPerPointLabel.HorizontalAlignment = '
                center';
            app.AutoMeasISO111146Part1MeasPerPointLabel.Text = 'Measurements per
                Point';
            %% Create AutoMeasISO111146Part1MeasPerPointEditField
            app.AutoMeasISO111146Part1MeasPerPointEditField = uieditfield(app.
                AutoMeasISO111146Part1TabGridLayout,'numeric');
            app.AutoMeasISO111146Part1MeasPerPointEditField.Layout.Row = 5;
            app.AutoMeasISO111146Part1MeasPerPointEditField.Layout.Column = 4;
            app.AutoMeasISO111146Part1MeasPerPointEditField.HorizontalAlignment = '
                center';
            app.AutoMeasISO111146Part1MeasPerPointEditField.Value = 5;
            %% Create AutoMeasISO111146Part1MeasAreaLabel
            app.AutoMeasISO111146Part1MeasAreaLabel = uilabel(app.
                AutoMeasISO111146Part1TabGridLayout);
            app.AutoMeasISO111146Part1MeasAreaLabel.Layout.Row = 1;
            app.AutoMeasISO111146Part1MeasAreaLabel.Layout.Column = [5 7];
            app.AutoMeasISO111146Part1MeasAreaLabel.FontSize = app.fontsize14;
            app.AutoMeasISO111146Part1MeasAreaLabel.Interpreter = 'latex';
            app.AutoMeasISO111146Part1MeasAreaLabel.BackgroundColor =
                backGroundColorLabel;
            app.AutoMeasISO111146Part1MeasAreaLabel.HorizontalAlignment = 'center';
            app.AutoMeasISO111146Part1MeasAreaLabel.Text = '$\textrm{Measurement
                Area in } \mu m$';
            %% Create AutoMeasISO111146Part1MeasAreaEditField
            app.AutoMeasISO111146Part1MeasAreaEditField = uieditfield(app.
                AutoMeasISO111146Part1TabGridLayout,'numeric','ValueDisplayFormat','
                %.1f');
            app.AutoMeasISO111146Part1MeasAreaEditField.Layout.Row = 1;
            app.AutoMeasISO111146Part1MeasAreaEditField.Layout.Column = 8;
            app.AutoMeasISO111146Part1MeasAreaEditField.HorizontalAlignment = '
                center';
            app.AutoMeasISO111146Part1MeasAreaEditField.Value = 10000;
```

```
            app.AutoMeasISO111146Part1MeasAreaEditField.Enable = 'off';
            app.AutoMeasISO111146Part1MeasAreaEditField.ValueChangedFcn = @(src,
                event) UpdatePreviewAxes(app, src, event);
            %% Create AutoMeasISO111146Part1StartMeasButton
            app.AutoMeasISO111146Part1StartMeasButton = uibutton(app.
                AutoMeasISO111146Part1TabGridLayout);
            app.AutoMeasISO111146Part1StartMeasButton.Layout.Row = 5;
            app.AutoMeasISO111146Part1StartMeasButton.Layout.Column = [14 15];
            app.AutoMeasISO111146Part1StartMeasButton.Text = 'Start';
            app.AutoMeasISO111146Part1StartMeasButton.ButtonPushedFcn = @(src, event
                ) AutoMeasISO111146Part1StartMeasButton_ButtonPushedFcn(app, src,
                event);
            app.AutoMeasISO111146Part1StartMeasButton.Enable = 'off';
            %% Create AutoMeasISO111146Part1PreviewMeasPointsUIAxes
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes = uiaxes(app.
                AutoMeasISO111146Part1TabGridLayout);
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Layout.Row = [7 8];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Layout.Column = [6
                15];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XLim = [0-250 app.
                maxValueStage+250];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.YLim = [-0.5 0.5];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XAxisLocation = '
                origin';
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Interactions = [];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Toolbar.Visible = 'off
                ';
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.YTickLabel = [];
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Title.Interpreter = '
                latex';
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.Title.String = '
                Measuring Points';
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XLabel.Interpreter = '
                latex';
            app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XLabel.String = 'z -
                coordinates';
            xtickangle(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes,45);
            %% Create currPosRoi
            app.currPosRoiAutoMeas = drawpoint(app.
                AutoMeasISO111146Part1PreviewMeasPointsUIAxes,'Position',app.
                currPosRoi.Position,'Color','yellow');
            app.currPosRoiAutoMeas.InteractionsAllowed = 'none';
            %% Create AutoMeasISO111146Part1CurrPosLabel
            app.AutoMeasISO111146Part1CurrPosLabel = uilabel(app.
                AutoMeasISO111146Part1TabGridLayout);
            app.AutoMeasISO111146Part1CurrPosLabel.Layout.Row = 9;
            app.AutoMeasISO111146Part1CurrPosLabel.Layout.Column = [6 8];
            app.AutoMeasISO111146Part1CurrPosLabel.FontSize = app.fontsize14;
            app.AutoMeasISO111146Part1CurrPosLabel.Interpreter = 'latex';
```

```
824             app.AutoMeasISO111146Part1CurrPosLabel.BackgroundColor =
                    backGroundColorLabel;
825             app.AutoMeasISO111146Part1CurrPosLabel.HorizontalAlignment = 'center';
826             app.AutoMeasISO111146Part1CurrPosLabel.Text = 'Current Position';
827             %% Create AutoMeasISO111146Part1CurrPosEditField
828             app.AutoMeasISO111146Part1CurrPosEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout,'numeric');
829             app.AutoMeasISO111146Part1CurrPosEditField.Layout.Row = 9;
830             app.AutoMeasISO111146Part1CurrPosEditField.Layout.Column = [9 10];
831             app.AutoMeasISO111146Part1CurrPosEditField.HorizontalAlignment = 'center
                    ';
832             app.AutoMeasISO111146Part1CurrPosEditField.Value = app.currPos(1);
833             app.AutoMeasISO111146Part1CurrPosEditField.Editable = 'off';
834             %% Create AutoMeasISO111146DistanceHolderLabel
835             app.AutoMeasISO111146DistanceHolderLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
836             app.AutoMeasISO111146DistanceHolderLabel.Layout.Row = 6;
837             app.AutoMeasISO111146DistanceHolderLabel.Layout.Column = [1 15];
838             app.AutoMeasISO111146DistanceHolderLabel.FontSize = app.fontsize14;
839             app.AutoMeasISO111146DistanceHolderLabel.Interpreter = 'latex';
840             app.AutoMeasISO111146DistanceHolderLabel.BackgroundColor = [0.4 0.4
                    0.4];
841             app.AutoMeasISO111146DistanceHolderLabel.HorizontalAlignment = 'center';
842             app.AutoMeasISO111146DistanceHolderLabel.Text = '';
843             %% Create AutoMeasISO111146Part1ProgressCurrActionLabel
844             app.AutoMeasISO111146Part1ProgressCurrActionLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
845             app.AutoMeasISO111146Part1ProgressCurrActionLabel.Layout.Row = 7;
846             app.AutoMeasISO111146Part1ProgressCurrActionLabel.Layout.Column = [1 3];
847             app.AutoMeasISO111146Part1ProgressCurrActionLabel.FontSize = app.
                    fontsize14;
848             app.AutoMeasISO111146Part1ProgressCurrActionLabel.Interpreter = 'latex';
849             app.AutoMeasISO111146Part1ProgressCurrActionLabel.BackgroundColor =
                    backGroundColorLabel;
850             app.AutoMeasISO111146Part1ProgressCurrActionLabel.HorizontalAlignment =
                    'center';
851             app.AutoMeasISO111146Part1ProgressCurrActionLabel.Text = 'Current Action
                    ';
852             %% Create AutoMeasISO111146Part1ProgressCurrActionEditField
853             app.AutoMeasISO111146Part1ProgressCurrActionEditField = uieditfield(app.
                    AutoMeasISO111146Part1TabGridLayout);
854             app.AutoMeasISO111146Part1ProgressCurrActionEditField.Layout.Row = 7;
855             app.AutoMeasISO111146Part1ProgressCurrActionEditField.Layout.Column = [4
                     5];
856             app.AutoMeasISO111146Part1ProgressCurrActionEditField.
                    HorizontalAlignment = 'center';
857             app.AutoMeasISO111146Part1ProgressCurrActionEditField.Editable = 'off';
858             %% Create AutoMeasISO111146Part1ProgressCurrMeasPointLabel
859             app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel = uilabel(app.
```

```
        AutoMeasISO111146Part1TabGridLayout);
860     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.Layout.Row = 8;
861     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.Layout.Column = [1
        3];
862     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.FontSize = app.
        fontsize14;
863     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.Interpreter = '
        latex';
864     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.BackgroundColor =
        backGroundColorLabel;
865     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.HorizontalAlignment
         = 'center';
866     app.AutoMeasISO111146Part1ProgressCurrMeasPointLabel.Text = 'Current
        Measuring point';
867     %% Create AutoMeasISO111146Part1ProgressCurrMeasPointEditField
868     app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField = uieditfield(
        app.AutoMeasISO111146Part1TabGridLayout);
869     app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Layout.Row = 8;
870     app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Layout.Column =
         4;
871     app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.
        HorizontalAlignment = 'center';
872     app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Editable = 'off
        ';
873     %% Create AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel
874     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel = uilabel(app.
        AutoMeasISO111146Part1TabGridLayout);
875     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.Layout.Row = 9;
876     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.Layout.Column =
         [1 3];
877     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.FontSize = app.
        fontsize14;
878     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.Interpreter = '
        latex';
879     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.BackgroundColor
         = backGroundColorLabel;
880     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.
        HorizontalAlignment = 'center';
881     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasLabel.Text = 'Current
         Measurement';
882     %% Create AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField
883     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField =
        uieditfield(app.AutoMeasISO111146Part1TabGridLayout);
884     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Layout.Row
        = 9;
885     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Layout.
        Column = 4;
886     app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.
        HorizontalAlignment = 'center';
```

```
887             app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Editable =
                    'off';
888             %% Create AutoMeasISO111146Part1StatusLabel
889             app.AutoMeasISO111146Part1StatusLabel = uilabel(app.
                    AutoMeasISO111146Part1TabGridLayout);
890             app.AutoMeasISO111146Part1StatusLabel.Layout.Row = 10;
891             app.AutoMeasISO111146Part1StatusLabel.Layout.Column = [1 3];
892             app.AutoMeasISO111146Part1StatusLabel.FontSize = app.fontsize14;
893             app.AutoMeasISO111146Part1StatusLabel.Interpreter = 'latex';
894             app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [1 0.07 0.2];
895             app.AutoMeasISO111146Part1StatusLabel.HorizontalAlignment = 'center';
896             app.AutoMeasISO111146Part1StatusLabel.Text = 'Inactive';
897             %% Create AutoMeasCancelButton
898             app.AutoMeasCancelButton = uibutton(app.
                    AutoMeasISO111146Part1TabGridLayout,'state');
899             app.AutoMeasCancelButton.Text = 'Cancel';
900             app.AutoMeasCancelButton.HorizontalAlignment = 'center';
901             app.AutoMeasCancelButton.FontSize = app.fontsize14;
902             app.AutoMeasCancelButton.Layout.Row = 5;
903             app.AutoMeasCancelButton.Layout.Column = [12 13];
904 %               app.AutoMeasCancelButton.ValueChangedFcn = @(src, event)
    HSCamConfCamPreviewButton_ValueChangedFcn(app, src, event);
905
906             %% Create EvaluateMeasurementTab
907             app.EvaluateMeasurementTab = uitab(app.TabGroup);
908             app.EvaluateMeasurementTab.Title = 'Evaluate Measurement';
909             app.EvaluateMeasurementTab.Scrollable = 'on';
910
911             %% Create TabGroup
912             app.EvalTabGroup = uitabgroup(app.EvaluateMeasurementTab);
913             app.EvalTabGroup.Units = 'normalized';
914             app.EvalTabGroup.Position = [0 0 1 1];
915             app.EvalTabGroup.TabLocation = 'top';
916             % app.EvalTabGroup.
917
918             %% Create ConfEvalTab
919             app.ConfEvalTab = uitab(app.EvalTabGroup);
920             app.ConfEvalTab.Title = 'Configure Evaluation';
921             %app.ConfEvalTab.BackgroundColor = [0.75 0.75 0.75];
922             app.ConfEvalTab.Scrollable = 'on';
923             %% Create ConfEvalTabGridLayout
924             app.ConfEvalTabGridLayout = uigridlayout(app.ConfEvalTab);
925             app.ConfEvalTabGridLayout.RowHeight = {'1x','1x','1x','1x','1x','1x','1x
                    ','1x','1x','1x','1x','1x','1x','1x','1x','1x'};
926             app.ConfEvalTabGridLayout.ColumnWidth = {'1x','1x','1x','1x','1x','1x','
                    1x','1x','1x'};
927
928             %% Create ConfEvalLoadMeasButton
929             app.ConfEvalLoadMeasButton = uibutton(app.ConfEvalTabGridLayout);
```

```
930             app.ConfEvalLoadMeasButton.Text = 'Load Measurement';
931             app.ConfEvalLoadMeasButton.Layout.Row = 1;
932             app.ConfEvalLoadMeasButton.Layout.Column = [1 2];
933             app.ConfEvalLoadMeasButton.ButtonPushedFcn = @(src, event)
                    ConfEvalLoadMeasButton_ButtonPushedFcn(app, src, event);
934             app.ConfEvalLoadMeasButton.FontSize = app.fontsize14;
935
936             %% Create ConfEvalBackCorrMethodLabel
937             app.ConfEvalBackCorrMethodLabel = uilabel(app.ConfEvalTabGridLayout);
938             app.ConfEvalBackCorrMethodLabel.Text = 'Coarse Background Correction
                    Method';
939             app.ConfEvalBackCorrMethodLabel.Layout.Row = 2;
940             app.ConfEvalBackCorrMethodLabel.Layout.Column = [1 2];
941             app.ConfEvalBackCorrMethodLabel.FontSize = app.fontsize14;
942             app.ConfEvalBackCorrMethodLabel.Interpreter = 'latex';
943             app.ConfEvalBackCorrMethodLabel.BackgroundColor = backGroundColorLabel;
944             app.ConfEvalBackCorrMethodLabel.HorizontalAlignment = 'center';
945             %% Create ConfEvalBackCorrMethodDropDown
946             app.ConfEvalBackCorrMethodDropDown = uidropdown(app.
                    ConfEvalTabGridLayout);
947             app.ConfEvalBackCorrMethodDropDown.Layout.Row = 3;
948             app.ConfEvalBackCorrMethodDropDown.Layout.Column = [1 2];
949             app.ConfEvalBackCorrMethodDropDown.Items = {'Background Map Substraction
                    ','Average Background Substraction'};
950             %               app.ConfEvalBackCorrMethodDropDown.ValueChangedFcn = @(src
                    , event) ConfEvalBackCorrMethodDropDown_ValueChangedFcn(app, src,
                    event);
951             app.ConfEvalBackCorrMethodDropDown.FontSize = app.fontsize14;
952             %% Create ConfEvalFineBackCorrMethodLabel
953             app.ConfEvalFineBackCorrMethodLabel = uilabel(app.ConfEvalTabGridLayout)
                    ;
954             app.ConfEvalFineBackCorrMethodLabel.Text = 'Fine Background Correction
                    Method';
955             app.ConfEvalFineBackCorrMethodLabel.Layout.Row = 2;
956             app.ConfEvalFineBackCorrMethodLabel.Layout.Column = [3 4];
957             app.ConfEvalFineBackCorrMethodLabel.FontSize = app.fontsize14;
958             app.ConfEvalFineBackCorrMethodLabel.Interpreter = 'latex';
959             app.ConfEvalFineBackCorrMethodLabel.BackgroundColor =
                    backGroundColorLabel;
960             app.ConfEvalFineBackCorrMethodLabel.HorizontalAlignment = 'center';
961
962             %% Create ConfEvalFineBackCorrMethodDropDown
963             app.ConfEvalFineBackCorrMethodDropDown = uidropdown(app.
                    ConfEvalTabGridLayout);
964             app.ConfEvalFineBackCorrMethodDropDown.Layout.Row = 3;
965             app.ConfEvalFineBackCorrMethodDropDown.Layout.Column = [3 4];
966             app.ConfEvalFineBackCorrMethodDropDown.Items = {'Statistical Method','
                    Manual Value','Off'};
967             %               app.ConfEvalFineBackCorrMethodDropDown.ValueChangedFcn =
```

```
            @(src, event) ConfEvalBackCorrMethodDropDown_ValueChangedFcn(app,
            src, event);
968         app.ConfEvalFineBackCorrMethodDropDown.FontSize = app.fontsize14;
969         % Create ConfEvalFineCorrValueLabel
970         app.ConfEvalFineCorrValueLabel = uilabel(app.ConfEvalTabGridLayout);
971         app.ConfEvalFineCorrValueLabel.Text = 'Fine Corr Value';
972         app.ConfEvalFineCorrValueLabel.Layout.Row = 2;
973         app.ConfEvalFineCorrValueLabel.Layout.Column = 5;
974         app.ConfEvalFineCorrValueLabel.FontSize = app.fontsize14;
975         app.ConfEvalFineCorrValueLabel.Interpreter = 'latex';
976         app.ConfEvalFineCorrValueLabel.BackgroundColor = backGroundColorLabel;
977         app.ConfEvalFineCorrValueLabel.HorizontalAlignment = 'center';
978         %% Create ConfEvalFineCorrValueEditField
979         app.ConfEvalFineCorrValueEditField = uieditfield(app.
            ConfEvalTabGridLayout,'numeric');
980         app.ConfEvalFineCorrValueEditField.Layout.Row = 3;
981         app.ConfEvalFineCorrValueEditField.Layout.Column = 5;
982         app.ConfEvalFineCorrValueEditField.Limits = [0 10];
983         %             app.ConfEvalNtEditField.ValueChangedFcn = @(src, event)
            ConfEvalNtFactorSlider_AllEvents(app, src, event);
984         app.ConfEvalFineCorrValueEditField.HorizontalAlignment = 'center';
985         app.ConfEvalFineCorrValueEditField.FontSize = app.fontsize14;
986         app.ConfEvalFineCorrValueEditField.Value = 0;
987         %% Create ConfEvalConvergeCriterionLabel
988         app.ConfEvalConvergeCriterionLabel = uilabel(app.ConfEvalTabGridLayout);
989         app.ConfEvalConvergeCriterionLabel.Text = '$\textrm{Convergence
            Criterion} \textrm{ in } \mu m$';
990         app.ConfEvalConvergeCriterionLabel.Layout.Row = 4;
991         app.ConfEvalConvergeCriterionLabel.Layout.Column = [1 2];
992         app.ConfEvalConvergeCriterionLabel.FontSize = app.fontsize14;
993         app.ConfEvalConvergeCriterionLabel.Interpreter = 'latex';
994         app.ConfEvalConvergeCriterionLabel.BackgroundColor =
            backGroundColorLabel;
995         app.ConfEvalConvergeCriterionLabel.HorizontalAlignment = 'center';
996         %% Create ConfEvalConvergeCriterionEditField
997         app.ConfEvalConvergeCriterionEditField = uieditfield(app.
            ConfEvalTabGridLayout,'numeric');
998         app.ConfEvalConvergeCriterionEditField.Layout.Row = 5;
999         app.ConfEvalConvergeCriterionEditField.Layout.Column = [1 2];
1000        app.ConfEvalConvergeCriterionEditField.Limits = [0.1 10];
1001        %             app.ConfEvalConvergeCriterionEditField.ValueChangedFcn =
            @(src, event) ConfEvalConvergeCriterionSlider_AllEvents(app, src,
            event);
1002        app.ConfEvalConvergeCriterionEditField.HorizontalAlignment = 'center';
1003        app.ConfEvalConvergeCriterionEditField.FontSize = app.fontsize14;
1004        %% Create ConfEvalWavelengthLabel
1005        app.ConfEvalWavelengthLabel = uilabel(app.ConfEvalTabGridLayout);
1006        app.ConfEvalWavelengthLabel.Text = '$\textrm{Wavelength }\lambda \textrm
            { in } nm $';
```

```
            app.ConfEvalWavelengthLabel.Layout.Row = 4;
            app.ConfEvalWavelengthLabel.Layout.Column = [3 4];
            app.ConfEvalWavelengthLabel.FontSize = app.fontsize14;
            app.ConfEvalWavelengthLabel.Interpreter = 'latex';
            app.ConfEvalWavelengthLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalWavelengthLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalWavelengthEditField
            app.ConfEvalWavelengthEditField = uieditfield(app.ConfEvalTabGridLayout,
                'numeric');
            app.ConfEvalWavelengthEditField.Layout.Row = 5;
            app.ConfEvalWavelengthEditField.Layout.Column = [3 4];
            app.ConfEvalWavelengthEditField.Limits = [300 1100];
            %             app.ConfEvalWavelengthEditField.ValueChangedFcn = @(src,
                event) ConfEvalWavelengthSlider_AllEvents(app, src, event);
            app.ConfEvalWavelengthEditField.HorizontalAlignment = 'center';
            app.ConfEvalWavelengthEditField.FontSize = app.fontsize14;
            %% Create ConfEvalNtFactorLabel
            app.ConfEvalNtFactorLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalNtFactorLabel.Text = 'nT standard deviation multiplicator';
            app.ConfEvalNtFactorLabel.Layout.Row = 6;
            app.ConfEvalNtFactorLabel.Layout.Column = [1 2];
            app.ConfEvalNtFactorLabel.FontSize = app.fontsize14;
            app.ConfEvalNtFactorLabel.Interpreter = 'latex';
            app.ConfEvalNtFactorLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalNtFactorLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalNtEditField
            app.ConfEvalNtEditField = uieditfield(app.ConfEvalTabGridLayout,'numeric
                ');
            app.ConfEvalNtEditField.Layout.Row = 7;
            app.ConfEvalNtEditField.Layout.Column = [1 2];
            app.ConfEvalNtEditField.Limits = [2 4];
            %             app.ConfEvalNtEditField.ValueChangedFcn = @(src, event)
                ConfEvalNtFactorSlider_AllEvents(app, src, event);
            app.ConfEvalNtEditField.HorizontalAlignment = 'center';
            app.ConfEvalNtEditField.FontSize = app.fontsize14;
            %% Create ConfEvalIntegAreaFactorLabel
            app.ConfEvalIntegAreaFactorLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalIntegAreaFactorLabel.Text = 'Integrationrange multiplicator'
                ;
            app.ConfEvalIntegAreaFactorLabel.Layout.Row = 8;
            app.ConfEvalIntegAreaFactorLabel.Layout.Column = [1 2];
            app.ConfEvalIntegAreaFactorLabel.FontSize = app.fontsize14;
            app.ConfEvalIntegAreaFactorLabel.Interpreter = 'latex';
            app.ConfEvalIntegAreaFactorLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalIntegAreaFactorLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalIntegAreaEditField
            app.ConfEvalIntegAreaEditField = uieditfield(app.ConfEvalTabGridLayout,'
                numeric');
            app.ConfEvalIntegAreaEditField.Layout.Row = 9;
```

```
            app.ConfEvalIntegAreaEditField.Layout.Column = [1 2];
            app.ConfEvalIntegAreaEditField.Limits = [1 3];
            app.ConfEvalIntegAreaEditField.HorizontalAlignment = 'center';
            app.ConfEvalIntegAreaEditField.FontSize = app.fontsize14;
            %% Create ConfEvalResLabel
            app.ConfEvalResLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalResLabel.Text = 'Image Resolution W x H in pixel';
            app.ConfEvalResLabel.Layout.Row = 12;
            app.ConfEvalResLabel.Layout.Column = [1 2];
            app.ConfEvalResLabel.FontSize = app.fontsize14;
            app.ConfEvalResLabel.Interpreter = 'latex';
            app.ConfEvalResLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalResLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalResWidthEditField
            app.ConfEvalResWidthEditField = uieditfield(app.ConfEvalTabGridLayout,'
                numeric');
            app.ConfEvalResWidthEditField.Layout.Row = 13;
            app.ConfEvalResWidthEditField.Layout.Column = 1;
            app.ConfEvalResWidthEditField.HorizontalAlignment = 'center';
            app.ConfEvalResWidthEditField.FontSize = app.fontsize14;
            app.ConfEvalResWidthEditField.Value = 1282;
            %% Create ConfEvalResHeightEditField
            app.ConfEvalResHeightEditField = uieditfield(app.ConfEvalTabGridLayout,'
                numeric');
            app.ConfEvalResHeightEditField.Layout.Row = 13;
            app.ConfEvalResHeightEditField.Layout.Column = 2;
            app.ConfEvalResHeightEditField.HorizontalAlignment = 'center';
            app.ConfEvalResHeightEditField.FontSize = app.fontsize14;
            app.ConfEvalResHeightEditField.Value = 1026;
            %% Create ConfEvalPixelsizeLabel
            app.ConfEvalPixelsizeLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalPixelsizeLabel.Text = '$\textrm{Pixelsize} \textrm{ in } \mu
                m$';
            app.ConfEvalPixelsizeLabel.Layout.Row = 12;
            app.ConfEvalPixelsizeLabel.Layout.Column = 3;
            app.ConfEvalPixelsizeLabel.FontSize = app.fontsize14;
            app.ConfEvalPixelsizeLabel.Interpreter = 'latex';
            app.ConfEvalPixelsizeLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalPixelsizeLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalPixelsizeEditField
            app.ConfEvalPixelsizeEditField = uieditfield(app.ConfEvalTabGridLayout,'
                numeric');
            app.ConfEvalPixelsizeEditField.Layout.Row = 13;
            app.ConfEvalPixelsizeEditField.Layout.Column = 3;
            app.ConfEvalPixelsizeEditField.HorizontalAlignment = 'center';
            app.ConfEvalPixelsizeEditField.FontSize = app.fontsize14;
            app.ConfEvalPixelsizeEditField.Value = 5.3;
            %% Create ConfEvalKernelSizeLabel
            app.ConfEvalKernelSizeLabel = uilabel(app.ConfEvalTabGridLayout);
```

```
            app.ConfEvalKernelSizeLabel.Text = 'Kernelsize in \%';
            app.ConfEvalKernelSizeLabel.Layout.Row = 6;
            app.ConfEvalKernelSizeLabel.Layout.Column = [3 4];
            app.ConfEvalKernelSizeLabel.FontSize = app.fontsize14;
            app.ConfEvalKernelSizeLabel.Interpreter = 'latex';
            app.ConfEvalKernelSizeLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalKernelSizeLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalKernelSizeEditField
            app.ConfEvalKernelSizeEditField = uieditfield(app.ConfEvalTabGridLayout,
                'numeric');
            app.ConfEvalKernelSizeEditField.Layout.Row = 7;
            app.ConfEvalKernelSizeEditField.Layout.Column = [3 4];
            app.ConfEvalKernelSizeEditField.Limits = [2 5];
            app.ConfEvalKernelSizeEditField.HorizontalAlignment = 'center';
            app.ConfEvalKernelSizeEditField.FontSize = app.fontsize14;
            %% Create ConfEvalMethodLabel
            app.ConfEvalMethodLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalMethodLabel.Text = 'Evaluation Norm';
            app.ConfEvalMethodLabel.Layout.Row = 16;
            app.ConfEvalMethodLabel.Layout.Column = [1 2];
            app.ConfEvalMethodLabel.FontSize = app.fontsize14;
            app.ConfEvalMethodLabel.Interpreter = 'latex';
            app.ConfEvalMethodLabel.BackgroundColor = backGroundColorLabel;
            app.ConfEvalMethodLabel.HorizontalAlignment = 'center';
            %% Create ConfEvalMethodDropDown
            app.ConfEvalMethodDropDown = uidropdown(app.ConfEvalTabGridLayout);
            app.ConfEvalMethodDropDown.Layout.Row = 17;
            app.ConfEvalMethodDropDown.Layout.Column = [1 2];
            app.ConfEvalMethodDropDown.Items = {'ISO11146-1'};
            app.ConfEvalMethodDropDown.FontSize = app.fontsize14;
            %% Create ConfEvalEvaluationButton
            app.ConfEvalEvaluationButton = uibutton(app.ConfEvalTabGridLayout);
            app.ConfEvalEvaluationButton.Text = 'Evaluate';
            app.ConfEvalEvaluationButton.Layout.Row = 17;
            app.ConfEvalEvaluationButton.Layout.Column = 3;
            app.ConfEvalEvaluationButton.ButtonPushedFcn = @(src, event)
                ConfEvalEvaluationButton_ButtonPushedFcn(app, src, event);
            app.ConfEvalEvaluationButton.FontSize = app.fontsize14;
            %% Create ConfEvalEvaluationStatusLabel
            app.ConfEvalEvaluationStatusLabel = uilabel(app.ConfEvalTabGridLayout);
            app.ConfEvalEvaluationStatusLabel.Layout.Row = 16;
            app.ConfEvalEvaluationStatusLabel.Layout.Column = 3;
            app.ConfEvalEvaluationStatusLabel.FontSize = app.fontsize14;
            app.ConfEvalEvaluationStatusLabel.Interpreter = 'latex';
            app.ConfEvalEvaluationStatusLabel.BackgroundColor = [1 0.07 0.2];
            app.ConfEvalEvaluationStatusLabel.HorizontalAlignment = 'center';
            app.ConfEvalEvaluationStatusLabel.Text = 'Standby';
            %% Create ConfEvalSmpLabel
            app.ConfEvalSmpLabel = uilabel(app.ConfEvalTabGridLayout);
```

```
1142            app.ConfEvalSmpLabel.Text = 'Measuring Point No.';
1143            app.ConfEvalSmpLabel.Layout.Row = 1;
1144            app.ConfEvalSmpLabel.Layout.Column = [7 8];
1145            app.ConfEvalSmpLabel.FontSize = app.fontsize14;
1146            app.ConfEvalSmpLabel.Interpreter = 'latex';
1147            app.ConfEvalSmpLabel.HorizontalAlignment = 'center';
1148            app.ConfEvalSmpLabel.BackgroundColor = backGroundColorLabel;
1149            %% Create ConfEvalSmpDropDown
1150            app.ConfEvalSmpDropDown = uidropdown(app.ConfEvalTabGridLayout);
1151            app.ConfEvalSmpDropDown.Layout.Row = 2;
1152            app.ConfEvalSmpDropDown.Layout.Column = [7 8];
1153            app.ConfEvalSmpDropDown.ValueChangedFcn = @(src, event)
                    ConfEvalSmpDropDown_ValueChangedFcn(app, src, event);
1154            app.ConfEvalSmpDropDown.Items = {};
1155            app.ConfEvalSmpDropDown.Enable = 'off';
1156            app.ConfEvalSmpDropDown.FontSize = app.fontsize14;
1157            %% Create ConfEvalSmpImLabel
1158            app.ConfEvalSmpImLabel = uilabel(app.ConfEvalTabGridLayout);
1159            app.ConfEvalSmpImLabel.Text = 'Image No.';
1160            app.ConfEvalSmpImLabel.Layout.Row = 1;
1161            app.ConfEvalSmpImLabel.Layout.Column = 9 ;
1162            app.ConfEvalSmpImLabel.FontSize = app.fontsize14;
1163            app.ConfEvalSmpImLabel.Interpreter = 'latex';
1164            app.ConfEvalSmpImLabel.HorizontalAlignment = 'center';
1165            app.ConfEvalSmpImLabel.BackgroundColor = backGroundColorLabel;
1166            %% Create ConfEvalSmpImDropDown
1167            app.ConfEvalSmpImDropDown = uidropdown(app.ConfEvalTabGridLayout);
1168            app.ConfEvalSmpImDropDown.Layout.Row = 2;
1169            app.ConfEvalSmpImDropDown.Layout.Column = 9;
1170            app.ConfEvalSmpImDropDown.ValueChangedFcn = @(src, event)
                    ConfEvalSmpImDropDown_ValueChangedFcn(app, src, event);
1171            app.ConfEvalSmpImDropDown.Items = {};
1172            app.ConfEvalSmpImDropDown.Enable = 'off';
1173            app.ConfEvalSmpImDropDown.FontSize = app.fontsize14;
1174            %% Create ConfEvalDrawRectButton
1175            app.ConfEvalDrawRectButton = uibutton(app.ConfEvalTabGridLayout);
1176            app.ConfEvalDrawRectButton.Text = 'Guess Beamarea';
1177            app.ConfEvalDrawRectButton.Layout.Row = 2;
1178            app.ConfEvalDrawRectButton.Layout.Column = 10;
1179            app.ConfEvalDrawRectButton.ButtonPushedFcn = @(src, event)
                    ConfEvalDrawRectButton_ButtonPushedFcn(app, src, event);
1180            app.ConfEvalDrawRectButton.FontSize = app.fontsize14;
1181            %% Create ConfEvalPreviewImUIAxes
1182            app.ConfEvalPreviewImUIAxes = uiaxes(app.ConfEvalTabGridLayout);
1183            app.ConfEvalPreviewImUIAxes.Layout.Row = [3 14];
1184            app.ConfEvalPreviewImUIAxes.Layout.Column = [7 10];
1185            %%Create LiveMeasTab
1186            app.LiveMeasTab = uitab(app.TabGroup);
1187            app.LiveMeasTab.Title = 'Live Measurement';
```

```
1188            app.LiveMeasTab.Scrollable = 'on';
1189            app.LiveMeasTabGridLayout = uigridlayout(app.LiveMeasTab);
1190            app.LiveMeasTabGridLayout.RowHeight = {'1x','1x','1x','1x','1x','1x','1x
                    ','1x','1x','1x','1x','1x','1x','1x','1x','1x'};
1191            app.LiveMeasTabGridLayout.ColumnWidth = {'1x','1x','1x','1x','1x','1x','
                    1x','1x','1x'};
1192            %% Create LiveMeasBackCorrMethodLabel
1193            app.LiveMeasBackCorrMethodLabel = uilabel(app.LiveMeasTabGridLayout);
1194            app.LiveMeasBackCorrMethodLabel.Text = 'Coarse Background Correction
                    Method';
1195            app.LiveMeasBackCorrMethodLabel.Layout.Row = 2;
1196            app.LiveMeasBackCorrMethodLabel.Layout.Column = [1 2];
1197            app.LiveMeasBackCorrMethodLabel.FontSize = app.fontsize14;
1198            app.LiveMeasBackCorrMethodLabel.Interpreter = 'latex';
1199            app.LiveMeasBackCorrMethodLabel.BackgroundColor = backGroundColorLabel;
1200            app.LiveMeasBackCorrMethodLabel.HorizontalAlignment = 'center';
1201            %% Create LiveMeasBackCorrMethodDropDown
1202            app.LiveMeasBackCorrMethodDropDown = uidropdown(app.
                    LiveMeasTabGridLayout);
1203            app.LiveMeasBackCorrMethodDropDown.Layout.Row = 3;
1204            app.LiveMeasBackCorrMethodDropDown.Layout.Column = [1 2];
1205            %               app.LiveMeasBackCorrMethodDropDown.Items = {'Background
                    Map Substraction','Average Background Substraction'};
1206            app.LiveMeasBackCorrMethodDropDown.FontSize = app.fontsize14;
1207            linkprop([app.ConfEvalBackCorrMethodDropDown app.
                    LiveMeasBackCorrMethodDropDown],'Items');
1208            %% Create LiveMeasFineBackCorrMethodLabel
1209            app.LiveMeasFineBackCorrMethodLabel = uilabel(app.LiveMeasTabGridLayout)
                    ;
1210            app.LiveMeasFineBackCorrMethodLabel.Text = 'Fine Background Correction
                    Method';
1211            app.LiveMeasFineBackCorrMethodLabel.Layout.Row = 2;
1212            app.LiveMeasFineBackCorrMethodLabel.Layout.Column = [3 4];
1213            app.LiveMeasFineBackCorrMethodLabel.FontSize = app.fontsize14;
1214            app.LiveMeasFineBackCorrMethodLabel.Interpreter = 'latex';
1215            app.LiveMeasFineBackCorrMethodLabel.BackgroundColor =
                    backGroundColorLabel;
1216            app.LiveMeasFineBackCorrMethodLabel.HorizontalAlignment = 'center';
1217            %% Create LiveMeasFineBackCorrMethodDropDown
1218            app.LiveMeasFineBackCorrMethodDropDown = uidropdown(app.
                    LiveMeasTabGridLayout);
1219            app.LiveMeasFineBackCorrMethodDropDown.Layout.Row = 3;
1220            app.LiveMeasFineBackCorrMethodDropDown.Layout.Column = [3 4];
1221            linkprop([app.ConfEvalFineBackCorrMethodDropDown app.
                    LiveMeasFineBackCorrMethodDropDown],'Items');
1222            app.LiveMeasFineBackCorrMethodDropDown.FontSize = app.fontsize14;
1223            % Create LiveMeasFineCorrValueLabel
1224            app.LiveMeasFineCorrValueLabel = uilabel(app.LiveMeasTabGridLayout);
1225            app.LiveMeasFineCorrValueLabel.Text = 'Fine Corr Value';
```

```
1226            app.LiveMeasFineCorrValueLabel.Layout.Row = 2;
1227            app.LiveMeasFineCorrValueLabel.Layout.Column = 5;
1228            app.LiveMeasFineCorrValueLabel.FontSize = app.fontsize14;
1229            app.LiveMeasFineCorrValueLabel.Interpreter = 'latex';
1230            app.LiveMeasFineCorrValueLabel.BackgroundColor = backGroundColorLabel;
1231            app.LiveMeasFineCorrValueLabel.HorizontalAlignment = 'center';
1232            %% Create LiveMeasFineCorrValueEditField
1233            app.LiveMeasFineCorrValueEditField = uieditfield(app.
                    LiveMeasTabGridLayout,'numeric');
1234            app.LiveMeasFineCorrValueEditField.Layout.Row = 3;
1235            app.LiveMeasFineCorrValueEditField.Layout.Column = 5;
1236            app.LiveMeasFineCorrValueEditField.Limits = [0 10];
1237            %               app.ConfEvalNtEditField.ValueChangedFcn = @(src, event)
                    ConfEvalNtFactorSlider_AllEvents(app, src, event);
1238            app.LiveMeasFineCorrValueEditField.HorizontalAlignment = 'center';
1239            app.LiveMeasFineCorrValueEditField.FontSize = app.fontsize14;
1240            app.LiveMeasFineCorrValueEditField.Value = 0;
1241            %% Create LiveMeasConvergeCriterionLabel
1242            app.LiveMeasConvergeCriterionLabel = uilabel(app.LiveMeasTabGridLayout);
1243            app.LiveMeasConvergeCriterionLabel.Text = '$\textrm{Convergence
                    Criterion} \textrm{ in } \mu m$';
1244            app.LiveMeasConvergeCriterionLabel.Layout.Row = 4;
1245            app.LiveMeasConvergeCriterionLabel.Layout.Column = [1 2];
1246            app.LiveMeasConvergeCriterionLabel.FontSize = app.fontsize14;
1247            app.LiveMeasConvergeCriterionLabel.Interpreter = 'latex';
1248            app.LiveMeasConvergeCriterionLabel.BackgroundColor =
                    backGroundColorLabel;
1249            app.LiveMeasConvergeCriterionLabel.HorizontalAlignment = 'center';
1250            %% Create LiveMeasConvergeCriterionEditField
1251            app.LiveMeasConvergeCriterionEditField = uieditfield(app.
                    LiveMeasTabGridLayout,'numeric');
1252            app.LiveMeasConvergeCriterionEditField.Layout.Row = 5;
1253            app.LiveMeasConvergeCriterionEditField.Layout.Column = [1 2];
1254            app.LiveMeasConvergeCriterionEditField.Limits = [0.1 10];
1255            app.LiveMeasConvergeCriterionEditField.HorizontalAlignment = 'center';
1256            app.LiveMeasConvergeCriterionEditField.FontSize = app.fontsize14;
1257            %% Create LiveMeasNtFactorLabel
1258            app.LiveMeasNtFactorLabel = uilabel(app.LiveMeasTabGridLayout);
1259            app.LiveMeasNtFactorLabel.Text = 'nT standard deviation multiplicator';
1260            app.LiveMeasNtFactorLabel.Layout.Row = 6;
1261            app.LiveMeasNtFactorLabel.Layout.Column = [1 2];
1262            app.LiveMeasNtFactorLabel.FontSize = app.fontsize14;
1263            app.LiveMeasNtFactorLabel.Interpreter = 'latex';
1264            app.LiveMeasNtFactorLabel.BackgroundColor = backGroundColorLabel;
1265            app.LiveMeasNtFactorLabel.HorizontalAlignment = 'center';
1266            %% Create LiveMeasNtEditField
1267            app.LiveMeasNtEditField = uieditfield(app.LiveMeasTabGridLayout,'numeric
                    ');
1268            app.LiveMeasNtEditField.Layout.Row = 7;
```

```
            app.LiveMeasNtEditField.Layout.Column = [1 2];
            app.LiveMeasNtEditField.Limits = [2 4];
            app.LiveMeasNtEditField.HorizontalAlignment = 'center';
            app.LiveMeasNtEditField.FontSize = app.fontsize14;
            %% Create LiveMeasIntegAreaFactorLabel
            app.LiveMeasIntegAreaFactorLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasIntegAreaFactorLabel.Text = 'Integrationrange multiplicator'
                ;
            app.LiveMeasIntegAreaFactorLabel.Layout.Row = 8;
            app.LiveMeasIntegAreaFactorLabel.Layout.Column = [1 2];
            app.LiveMeasIntegAreaFactorLabel.FontSize = app.fontsize14;
            app.LiveMeasIntegAreaFactorLabel.Interpreter = 'latex';
            app.LiveMeasIntegAreaFactorLabel.BackgroundColor = backGroundColorLabel;
            app.LiveMeasIntegAreaFactorLabel.HorizontalAlignment = 'center';
            %% Create LiveMeasIntegAreaEditField
            app.LiveMeasIntegAreaEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasIntegAreaEditField.Layout.Row = 9;
            app.LiveMeasIntegAreaEditField.Layout.Column = [1 2];
            app.LiveMeasIntegAreaEditField.Limits = [1 3];
            app.LiveMeasIntegAreaEditField.HorizontalAlignment = 'center';
            app.LiveMeasIntegAreaEditField.FontSize = app.fontsize14;
            %% Create LiveMeasResLabel
            app.LiveMeasResLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasResLabel.Text = 'Image Resolution W x H in pixel';
            app.LiveMeasResLabel.Layout.Row = 12;
            app.LiveMeasResLabel.Layout.Column = [1 2];
            app.LiveMeasResLabel.FontSize = app.fontsize14;
            app.LiveMeasResLabel.Interpreter = 'latex';
            app.LiveMeasResLabel.BackgroundColor = backGroundColorLabel;
            app.LiveMeasResLabel.HorizontalAlignment = 'center';
            %% Create LiveMeasResWidthEditField
            app.LiveMeasResWidthEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasResWidthEditField.Layout.Row = 13;
            app.LiveMeasResWidthEditField.Layout.Column = 1;
            app.LiveMeasResWidthEditField.HorizontalAlignment = 'center';
            app.LiveMeasResWidthEditField.FontSize = app.fontsize14;
            app.LiveMeasResWidthEditField.Value = 1282;
            %% Create LiveMeasResHeightEditField
            app.LiveMeasResHeightEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasResHeightEditField.Layout.Row = 13;
            app.LiveMeasResHeightEditField.Layout.Column = 2;
            app.LiveMeasResHeightEditField.HorizontalAlignment = 'center';
            app.LiveMeasResHeightEditField.FontSize = app.fontsize14;
            app.LiveMeasResHeightEditField.Value = 1026;
            %% Create LiveMeasPixelsizeLabel
            app.LiveMeasPixelsizeLabel = uilabel(app.LiveMeasTabGridLayout);
```

```
1314            app.LiveMeasPixelsizeLabel.Text = '$\textrm{Pixelsize} \textrm{ in } \mu
                    m$';
1315            app.LiveMeasPixelsizeLabel.Layout.Row = 12;
1316            app.LiveMeasPixelsizeLabel.Layout.Column = 3;
1317            app.LiveMeasPixelsizeLabel.FontSize = app.fontsize14;
1318            app.LiveMeasPixelsizeLabel.Interpreter = 'latex';
1319            app.LiveMeasPixelsizeLabel.BackgroundColor = backGroundColorLabel;
1320            app.LiveMeasPixelsizeLabel.HorizontalAlignment = 'center';
1321            %% Create LiveMeasPixelsizeEditField
1322            app.LiveMeasPixelsizeEditField = uieditfield(app.LiveMeasTabGridLayout,'
                    numeric');
1323            app.LiveMeasPixelsizeEditField.Layout.Row = 13;
1324            app.LiveMeasPixelsizeEditField.Layout.Column = 3;
1325            app.LiveMeasPixelsizeEditField.HorizontalAlignment = 'center';
1326            app.LiveMeasPixelsizeEditField.FontSize = app.fontsize14;
1327            app.LiveMeasPixelsizeEditField.Value = 5.3;
1328            %% Create LiveMeasKernelSizeLabel
1329            app.LiveMeasKernelSizeLabel = uilabel(app.LiveMeasTabGridLayout);
1330            app.LiveMeasKernelSizeLabel.Text = 'Kernelsize in \%';
1331            app.LiveMeasKernelSizeLabel.Layout.Row = 6;
1332            app.LiveMeasKernelSizeLabel.Layout.Column = [3 4];
1333            app.LiveMeasKernelSizeLabel.FontSize = app.fontsize14;
1334            app.LiveMeasKernelSizeLabel.Interpreter = 'latex';
1335            app.LiveMeasKernelSizeLabel.BackgroundColor = backGroundColorLabel;
1336            app.LiveMeasKernelSizeLabel.HorizontalAlignment = 'center';
1337            %% Create LiveMeasKernelSizeEditField
1338            app.LiveMeasKernelSizeEditField = uieditfield(app.LiveMeasTabGridLayout,
                    'numeric');
1339            app.LiveMeasKernelSizeEditField.Layout.Row = 7;
1340            app.LiveMeasKernelSizeEditField.Layout.Column = [3 4];
1341            app.LiveMeasKernelSizeEditField.Limits = [2 5];
1342            app.LiveMeasKernelSizeEditField.HorizontalAlignment = 'center';
1343            app.LiveMeasKernelSizeEditField.FontSize = app.fontsize14;
1344            %% Create LiveMeasMethodLabel
1345            app.LiveMeasMethodLabel = uilabel(app.LiveMeasTabGridLayout);
1346            app.LiveMeasMethodLabel.Text = 'Evaluation Norm';
1347            app.LiveMeasMethodLabel.Layout.Row = 16;
1348            app.LiveMeasMethodLabel.Layout.Column = [1 2];
1349            app.LiveMeasMethodLabel.FontSize = app.fontsize14;
1350            app.LiveMeasMethodLabel.Interpreter = 'latex';
1351            app.LiveMeasMethodLabel.BackgroundColor = backGroundColorLabel;
1352            app.LiveMeasMethodLabel.HorizontalAlignment = 'center';
1353            %% Create LiveMeasMethodDropDown
1354            app.LiveMeasMethodDropDown = uidropdown(app.LiveMeasTabGridLayout);
1355            app.LiveMeasMethodDropDown.Layout.Row = 17;
1356            app.LiveMeasMethodDropDown.Layout.Column = [1 2];
1357            app.LiveMeasMethodDropDown.Items = {'ISO11146-1'};
1358            app.LiveMeasMethodDropDown.FontSize = app.fontsize14;
1359            %% Create LiveMeasPreviewImUIAxes
```

```
            app.LiveMeasPreviewImUIAxes = uiaxes(app.LiveMeasTabGridLayout);
            app.LiveMeasPreviewImUIAxes.Layout.Row = [1 8];
            app.LiveMeasPreviewImUIAxes.Layout.Column = [7 10];
            %% Create LiveMeasResultImUIAxes
            app.LiveMeasResultImUIAxes = uiaxes(app.LiveMeasTabGridLayout);
            app.LiveMeasResultImUIAxes.Layout.Row = [9 16];
            app.LiveMeasResultImUIAxes.Layout.Column = [7 10];
            %% Create LiveMeasStartStopButton
            app.LiveMeasStartStopButton = uibutton(app.LiveMeasTabGridLayout,'state'
                );
            app.LiveMeasStartStopButton.Text = 'Start/Stop Live Measurement';
            app.LiveMeasStartStopButton.Layout.Row = 17;
            app.LiveMeasStartStopButton.Layout.Column = [3 4];
            app.LiveMeasStartStopButton.ValueChangedFcn = @(src, event)
                LiveMeasStartStopButton_ValueChangedFcn(app, src, event);
            app.LiveMeasStartStopButton.FontSize = app.fontsize14;
            app.LiveMeasStartStopButton.Enable = 'off';
            %% Create LiveMeasBackgroundMeasButton
            app.LiveMeasBackgroundMeasButton = uibutton(app.LiveMeasTabGridLayout);
            app.LiveMeasBackgroundMeasButton.Text = 'Measure Background';
            app.LiveMeasBackgroundMeasButton.Layout.Row = 16;
            app.LiveMeasBackgroundMeasButton.Layout.Column = [3 4];
            app.LiveMeasBackgroundMeasButton.ButtonPushedFcn = @(src, event)
                LiveMeasBackgroundMeasButton_ButtonPushedFcn(app, src, event);
            app.LiveMeasBackgroundMeasButton.FontSize = app.fontsize14;
            %% Create AutoMeasISO111146Part1StatusLabel
            app.LiveMeasStatusLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasStatusLabel.Layout.Row = 17;
            app.LiveMeasStatusLabel.Layout.Column = [5 6];
            app.LiveMeasStatusLabel.FontSize = app.fontsize14;
            app.LiveMeasStatusLabel.Interpreter = 'latex';
            app.LiveMeasStatusLabel.BackgroundColor = [1 0.07 0.2];
            app.LiveMeasStatusLabel.HorizontalAlignment = 'center';
            app.LiveMeasStatusLabel.Text = 'Inactive';
            %% Create LiveMeasImData_dWx_Label
            app.LiveMeasImData_dWx_Label = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasImData_dWx_Label.Text = '$\textrm{Beamdiameter } d_{x} \
                textrm{ in } \mu m$';
            app.LiveMeasImData_dWx_Label.Interpreter = 'latex';
            app.LiveMeasImData_dWx_Label.HorizontalAlignment = 'center';
            app.LiveMeasImData_dWx_Label.BackgroundColor = backGroundColorLabel;
            app.LiveMeasImData_dWx_Label.Layout.Row = 9;
            app.LiveMeasImData_dWx_Label.Layout.Column = [4 5];
            %% Create LiveMeasImData_dWx
            app.LiveMeasImData_dWx = uieditfield(app.LiveMeasTabGridLayout,'numeric'
                );
            app.LiveMeasImData_dWx.HorizontalAlignment = 'center';
            app.LiveMeasImData_dWx.Editable = 'off';
            app.LiveMeasImData_dWx.Layout.Row = 9;
```

```
1404            app.LiveMeasImData_dWx.Layout.Column = 6;
1405            %% Create LiveMeasImData_dWy_Label
1406            app.LiveMeasImData_dWy_Label = uilabel(app.LiveMeasTabGridLayout);
1407            app.LiveMeasImData_dWy_Label.Text = '$\textrm{Beamdiameter } d_{y} \
                    textrm{ in } \mu m$';
1408            app.LiveMeasImData_dWy_Label.Interpreter = 'latex';
1409            app.LiveMeasImData_dWy_Label.HorizontalAlignment = 'center';
1410            app.LiveMeasImData_dWy_Label.BackgroundColor = backGroundColorLabel;
1411            app.LiveMeasImData_dWy_Label.Layout.Row = 10;
1412            app.LiveMeasImData_dWy_Label.Layout.Column = [4 5];
1413            %% Create LiveMeasImData_dWy
1414            app.LiveMeasImData_dWy = uieditfield(app.LiveMeasTabGridLayout,'numeric'
                    );
1415            app.LiveMeasImData_dWy.HorizontalAlignment = 'center';
1416            app.LiveMeasImData_dWy.Editable = 'off';
1417            app.LiveMeasImData_dWy.Layout.Row = 10;
1418            app.LiveMeasImData_dWy.Layout.Column = 6;
1419            %% Create LiveMeasImData_phi_Label
1420            app.LiveMeasImData_phi_Label = uilabel(app.LiveMeasTabGridLayout);
1421            app.LiveMeasImData_phi_Label.Text = '$\textrm{Azimutangle } \varphi \
                    textrm{ in } Â°$';
1422            app.LiveMeasImData_phi_Label.Interpreter = 'latex';
1423            app.LiveMeasImData_phi_Label.HorizontalAlignment = 'center';
1424            app.LiveMeasImData_phi_Label.BackgroundColor = backGroundColorLabel;
1425            app.LiveMeasImData_phi_Label.Layout.Row = 11;
1426            app.LiveMeasImData_phi_Label.Layout.Column = [4 5];
1427            %% Create LiveMeasImData_phi
1428            app.LiveMeasImData_phi = uieditfield(app.LiveMeasTabGridLayout,'numeric'
                    );
1429            app.LiveMeasImData_phi.HorizontalAlignment = 'center';
1430            app.LiveMeasImData_phi.Editable = 'off';
1431            app.LiveMeasImData_phi.Layout.Row = 11;
1432            app.LiveMeasImData_phi.Layout.Column = 6;
1433            %% Create LiveMeasTriggerImageButton
1434            app.LiveMeasTriggerImageButton = uibutton(app.LiveMeasTabGridLayout);
1435            app.LiveMeasTriggerImageButton.Text = 'Trigger Image';
1436            app.LiveMeasTriggerImageButton.Layout.Row = 2;
1437            app.LiveMeasTriggerImageButton.Layout.Column = 6;
1438            app.LiveMeasTriggerImageButton.ButtonPushedFcn = @(src, event)
                    LiveMeasTriggerImageButton_ButtonPushedFcn(app, src, event);
1439            app.LiveMeasTriggerImageButton.FontSize = app.fontsize14;
1440            %% Create LiveMeasDrawRectButton
1441            app.LiveMeasDrawRectButton = uibutton(app.LiveMeasTabGridLayout);
1442            app.LiveMeasDrawRectButton.Text = 'Guess Beamarea';
1443            app.LiveMeasDrawRectButton.Layout.Row = 3;
1444            app.LiveMeasDrawRectButton.Layout.Column = 6;
1445            app.LiveMeasDrawRectButton.ButtonPushedFcn = @(src, event)
                    LiveMeasDrawRectButton_ButtonPushedFcn(app, src, event);
1446            app.LiveMeasDrawRectButton.FontSize = app.fontsize14;
```

```
            %% Create LiveMeasChangeColorMapButton
            app.LiveMeasChangeColorMapButton = uibutton(app.LiveMeasTabGridLayout,'
                state');
            app.LiveMeasChangeColorMapButton.Text = 'Switch Color Map';
            app.LiveMeasChangeColorMapButton.Layout.Row = 13;
            app.LiveMeasChangeColorMapButton.Layout.Column = 6;
            app.LiveMeasChangeColorMapButton.FontSize = app.fontsize14;
            %% Create LiveMeasColorMinLabel
            app.LiveMeasColorMinLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasColorMinLabel.Text = 'Min Value';
            app.LiveMeasColorMinLabel.Layout.Row = 14;
            app.LiveMeasColorMinLabel.Layout.Column = 5;
            app.LiveMeasColorMinLabel.FontSize = app.fontsize14;
            app.LiveMeasColorMinLabel.Interpreter = 'latex';
            app.LiveMeasColorMinLabel.BackgroundColor = backGroundColorLabel;
            app.LiveMeasColorMinLabel.HorizontalAlignment = 'center';
            %% Create LiveMeasColorMinEditField
            app.LiveMeasColorMinEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasColorMinEditField.Layout.Row = 14;
            app.LiveMeasColorMinEditField.Layout.Column = 6;
            app.LiveMeasColorMinEditField.Limits = [0 255];
            app.LiveMeasColorMinEditField.HorizontalAlignment = 'center';
            app.LiveMeasColorMinEditField.FontSize = app.fontsize14;
            app.LiveMeasColorMinEditField.Value = app.LiveMeasColorMap(1);
            %% Create LiveMeasColorMaxLabel
            app.LiveMeasColorMaxLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasColorMaxLabel.Text = 'Max Value';
            app.LiveMeasColorMaxLabel.Layout.Row = 15;
            app.LiveMeasColorMaxLabel.Layout.Column = 5;
            app.LiveMeasColorMaxLabel.FontSize = app.fontsize14;
            app.LiveMeasColorMaxLabel.Interpreter = 'latex';
            app.LiveMeasColorMaxLabel.BackgroundColor = backGroundColorLabel;
            app.LiveMeasColorMaxLabel.HorizontalAlignment = 'center';
            %% Create LiveMeasColorMinEditField
            app.LiveMeasColorMaxEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasColorMaxEditField.Layout.Row = 15;
            app.LiveMeasColorMaxEditField.Layout.Column = 6;
            app.LiveMeasColorMaxEditField.Limits = [0 255];
            app.LiveMeasColorMaxEditField.HorizontalAlignment = 'center';
            app.LiveMeasColorMaxEditField.FontSize = app.fontsize14;
            app.LiveMeasColorMaxEditField.Value = app.LiveMeasColorMap(2);
            %% Create LiveMeasNoOfImLabel
            app.LiveMeasNoOfImLabel = uilabel(app.LiveMeasTabGridLayout);
            app.LiveMeasNoOfImLabel.Text = 'No Of Background Images';
            app.LiveMeasNoOfImLabel.Layout.Row = 14;
            app.LiveMeasNoOfImLabel.Layout.Column = [3 4];
            app.LiveMeasNoOfImLabel.FontSize = app.fontsize14;
```

```
            app.LiveMeasNoOfImLabel.Interpreter = 'latex';
            app.LiveMeasNoOfImLabel.BackgroundColor = backGroundColorLabel;
            app.LiveMeasNoOfImLabel.HorizontalAlignment = 'center';
            %% Create LiveMeasNoOfImEditField
            app.LiveMeasNoOfImEditField = uieditfield(app.LiveMeasTabGridLayout,'
                numeric');
            app.LiveMeasNoOfImEditField.Layout.Row = 15;
            app.LiveMeasNoOfImEditField.Layout.Column = [3 4];
            app.LiveMeasNoOfImEditField.Limits = [10 9999];
            app.LiveMeasNoOfImEditField.HorizontalAlignment = 'center';
            app.LiveMeasNoOfImEditField.FontSize = app.fontsize14;
            app.LiveMeasNoOfImEditField.Value = 20;
            %% Create EvalResTab
            app.EvalResTab = uitab(app.EvalTabGroup);
            app.EvalResTab.Title = 'Evaluation Result';
            app.EvalResTab.Scrollable = 'on';
            %% Create EvalResTabGridLayout
            app.EvalResTabGridLayout = uigridlayout(app.EvalResTab);
            app.EvalResTabGridLayout.RowHeight = {'1x'};
            app.EvalResTabGridLayout.ColumnWidth = {'1x','1x','1x'};
            %% Create EvalResMeasDataPanel
            app.EvalResMeasDataPanel = uipanel(app.EvalResTabGridLayout);
            %% Create EvalResMeasDataGridLayout
            app.EvalResMeasDataGridLayout = uigridlayout(app.EvalResMeasDataPanel);
            app.EvalResMeasDataGridLayout.RowHeight = {30,30,'1x',30,30,30};
            app.EvalResMeasDataGridLayout.ColumnWidth = {'1x','1x'};
            %% Create EvalResMeasLoadButton
            app.EvalResMeasLoadButton = uibutton(app.EvalResMeasDataGridLayout);
            app.EvalResMeasLoadButton.Layout.Row = 1;
            app.EvalResMeasLoadButton.Layout.Column = [1 2];
            app.EvalResMeasLoadButton.Text = 'Load Measurement';
            app.EvalResMeasLoadButton.FontSize = app.fontsize14;
            app.EvalResMeasLoadButton.ButtonPushedFcn = @(src, event)
                ConfEvalLoadMeasButton_ButtonPushedFcn(app, src, event);
            %% Create EvalResMeasDataResultDropDown_Label
            app.EvalResMeasDataResultDropDown_Label = uilabel(app.
                EvalResMeasDataGridLayout);
            app.EvalResMeasDataResultDropDown_Label.Layout.Row = 2;
            app.EvalResMeasDataResultDropDown_Label.Layout.Column = 1;
            app.EvalResMeasDataResultDropDown_Label.Text = 'Result No.';
            app.EvalResMeasDataResultDropDown_Label.FontSize = app.fontsize14;
            app.EvalResMeasDataResultDropDown_Label.HorizontalAlignment = 'center';
            app.EvalResMeasDataResultDropDown_Label.BackgroundColor =
                backGroundColorLabel;
            app.EvalResMeasDataResultDropDown_Label.Interpreter = 'latex';
            %% Create EvalResMeasDataResultDropDown
            app.EvalResMeasDataResultDropDown = uidropdown(app.
                EvalResMeasDataGridLayout);
            app.EvalResMeasDataResultDropDown.Layout.Row = 2;
```

```
            app.EvalResMeasDataResultDropDown.Layout.Column = 2;
            app.EvalResMeasDataResultDropDown.ValueChangedFcn = @(src, event)
                EvalResMeasDataResultDropDown_ValueChangedFcn(app, src, event);
            app.EvalResMeasDataResultDropDown.Items = {};
            app.EvalResMeasDataResultDropDown.ItemsData = [];
            %% Create EvalResMeasDataParamsGridLayout
            app.EvalResMeasDataParamsGridLayout = uigridlayout(app.
                EvalResMeasDataGridLayout);
            app.EvalResMeasDataParamsGridLayout.Layout.Row = 3;
            app.EvalResMeasDataParamsGridLayout.Layout.Column = [1 2];
            app.EvalResMeasDataParamsGridLayout.RowHeight =
                {20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20};
            app.EvalResMeasDataParamsGridLayout.ColumnWidth = {'1x','1x'};
            app.EvalResMeasDataParamsGridLayout.Scrollable = 'on';
            %% Create EvalResMeasData_EvalNorm_Label
            app.EvalResMeasData_EvalNorm_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_EvalNorm_Label.Text = 'Evaluation Norm';
            app.EvalResMeasData_EvalNorm_Label.Interpreter = 'latex';
            app.EvalResMeasData_EvalNorm_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_EvalNorm_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResMeasData_EvalNorm
            app.EvalResMeasData_EvalNorm = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'text');
            app.EvalResMeasData_EvalNorm.HorizontalAlignment = 'center';
            app.EvalResMeasData_EvalNorm.Editable = 'off';
            %% Create EvalResMeasData_BackCorrMethod_Label
            app.EvalResMeasData_BackCorrMethod_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_BackCorrMethod_Label.Text = 'Background Correction
                Method';
            app.EvalResMeasData_BackCorrMethod_Label.Interpreter = 'latex';
            app.EvalResMeasData_BackCorrMethod_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_BackCorrMethod_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResMeasData_BackCorrMethod
            app.EvalResMeasData_BackCorrMethod = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'text');
            app.EvalResMeasData_BackCorrMethod.HorizontalAlignment = 'center';
            app.EvalResMeasData_BackCorrMethod.Editable = 'off';
            %% Create EvalResMeasData_ConvCrit_Label
            app.EvalResMeasData_ConvCrit_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_ConvCrit_Label.Text = '$\textrm{Convergence
                Criterion} \textrm{ in } \mu m$';
            app.EvalResMeasData_ConvCrit_Label.Interpreter = 'latex';
            app.EvalResMeasData_ConvCrit_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_ConvCrit_Label.BackgroundColor =
```

```
            backGroundColorLabel;
1574        %% Create EvalResMeasData_ConvCrit
1575        app.EvalResMeasData_ConvCrit = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric');
1576        app.EvalResMeasData_ConvCrit.HorizontalAlignment = 'center';
1577        app.EvalResMeasData_ConvCrit.Editable = 'off';
1578        %% Create EvalResMeasData_Lambda_Label
1579        app.EvalResMeasData_Lambda_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
1580        app.EvalResMeasData_Lambda_Label.Text = '$\textrm{Wavelength }\lambda \
                textrm{ in } nm $';
1581        app.EvalResMeasData_Lambda_Label.Interpreter = 'latex';
1582        app.EvalResMeasData_Lambda_Label.HorizontalAlignment = 'center';
1583        app.EvalResMeasData_Lambda_Label.BackgroundColor = backGroundColorLabel;
1584        %% Create EvalResMeasData_Lambda
1585        app.EvalResMeasData_Lambda = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric');
1586        app.EvalResMeasData_Lambda.HorizontalAlignment = 'center';
1587        app.EvalResMeasData_Lambda.Editable = 'off';
1588        %% Create EvalResMeasData_IntegAreaFactor_Label
1589        app.EvalResMeasData_IntegAreaFactor_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
1590        app.EvalResMeasData_IntegAreaFactor_Label.Text = 'Integration Area Size
                Multiplicator';
1591        app.EvalResMeasData_IntegAreaFactor_Label.Interpreter = 'latex';
1592        app.EvalResMeasData_IntegAreaFactor_Label.HorizontalAlignment = 'center'
                ;
1593        app.EvalResMeasData_IntegAreaFactor_Label.BackgroundColor =
                backGroundColorLabel;
1594        %% Create EvalResMeasData_IntegAreaFactor
1595        app.EvalResMeasData_IntegAreaFactor = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric');
1596        app.EvalResMeasData_IntegAreaFactor.HorizontalAlignment = 'center';
1597        app.EvalResMeasData_IntegAreaFactor.Editable = 'off';
1598        %% Create EvalResMeasData_Nt_Label
1599        app.EvalResMeasData_Nt_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
1600        app.EvalResMeasData_Nt_Label.Text = 'nT';
1601        app.EvalResMeasData_Nt_Label.Interpreter = 'latex';
1602        app.EvalResMeasData_Nt_Label.HorizontalAlignment = 'center';
1603        app.EvalResMeasData_Nt_Label.BackgroundColor = backGroundColorLabel;
1604        %% Create EvalResMeasData_Nt
1605        app.EvalResMeasData_Nt = uieditfield(app.EvalResMeasDataParamsGridLayout
                ,'numeric');
1606        app.EvalResMeasData_Nt.HorizontalAlignment = 'center';
1607        app.EvalResMeasData_Nt.Editable = 'off';
1608        %% Create EvalResMeasData_KernelSizeInPercent_Label
1609        app.EvalResMeasData_KernelSizeInPercent_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
```

```
1610            app.EvalResMeasData_KernelSizeInPercent_Label.Text = 'Kernelsize in \%';
1611            app.EvalResMeasData_KernelSizeInPercent_Label.Interpreter = 'latex';
1612            app.EvalResMeasData_KernelSizeInPercent_Label.HorizontalAlignment = '
                    center';
1613            app.EvalResMeasData_KernelSizeInPercent_Label.BackgroundColor =
                    backGroundColorLabel;
1614            %% Create EvalResMeasData_KernelSizeInPercent
1615            app.EvalResMeasData_KernelSizeInPercent = uieditfield(app.
                    EvalResMeasDataParamsGridLayout,'numeric');
1616            app.EvalResMeasData_KernelSizeInPercent.HorizontalAlignment = 'center';
1617            app.EvalResMeasData_KernelSizeInPercent.Editable = 'off';
1618            %% Create EvalResMeasData_z_0_X_Label
1619            app.EvalResMeasData_z_0_X_Label = uilabel(app.
                    EvalResMeasDataParamsGridLayout);
1620            app.EvalResMeasData_z_0_X_Label.Text = '$\textrm{z}_{\textrm{0x}}\textrm
                    { in } \mu m$';
1621            app.EvalResMeasData_z_0_X_Label.Interpreter = 'latex';
1622            app.EvalResMeasData_z_0_X_Label.HorizontalAlignment = 'center';
1623            app.EvalResMeasData_z_0_X_Label.BackgroundColor = backGroundColorLabel;
1624            %% Create EvalResMeasData_z_0_X
1625            app.EvalResMeasData_z_0_X = uieditfield(app.
                    EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
                    ');
1626            app.EvalResMeasData_z_0_X.HorizontalAlignment = 'center';
1627            app.EvalResMeasData_z_0_X.Editable = 'off';
1628            %% Create EvalResMeasData_z_R_X_Label
1629            app.EvalResMeasData_z_R_X_Label = uilabel(app.
                    EvalResMeasDataParamsGridLayout);
1630            app.EvalResMeasData_z_R_X_Label.Text = '$\textrm{z}_{\textrm{Rx}}\textrm
                    { in } \mu m$';
1631            app.EvalResMeasData_z_R_X_Label.Interpreter = 'latex';
1632            app.EvalResMeasData_z_R_X_Label.HorizontalAlignment = 'center';
1633            app.EvalResMeasData_z_R_X_Label.BackgroundColor = backGroundColorLabel;
1634            %% Create EvalResMeasData_z_R_X
1635            app.EvalResMeasData_z_R_X = uieditfield(app.
                    EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
                    ');
1636            app.EvalResMeasData_z_R_X.HorizontalAlignment = 'center';
1637            app.EvalResMeasData_z_R_X.Editable = 'off';
1638            %% Create EvalResMeasData_d_0_X_Label
1639            app.EvalResMeasData_d_0_X_Label = uilabel(app.
                    EvalResMeasDataParamsGridLayout);
1640            app.EvalResMeasData_d_0_X_Label.Text = '$\textrm{d}_{\textrm{0x}}\textrm
                    { in } \mu m$';
1641            app.EvalResMeasData_d_0_X_Label.Interpreter = 'latex';
1642            app.EvalResMeasData_d_0_X_Label.HorizontalAlignment = 'center';
1643            app.EvalResMeasData_d_0_X_Label.BackgroundColor = backGroundColorLabel;
1644            %% Create EvalResMeasData_d_0_X
1645            app.EvalResMeasData_d_0_X = uieditfield(app.
```

```
            EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
            ');
1646        app.EvalResMeasData_d_0_X.HorizontalAlignment = 'center';
1647        app.EvalResMeasData_d_0_X.Editable = 'off';
1648        %% Create EvalResMeasData_theta_X_Label
1649        app.EvalResMeasData_theta_X_Label = uilabel(app.
            EvalResMeasDataParamsGridLayout);
1650        app.EvalResMeasData_theta_X_Label.Text = '$\Theta_{\textrm{x}}\textrm{
            in rad}$';
1651        app.EvalResMeasData_theta_X_Label.Interpreter = 'latex';
1652        app.EvalResMeasData_theta_X_Label.HorizontalAlignment = 'center';
1653        app.EvalResMeasData_theta_X_Label.BackgroundColor = backGroundColorLabel
            ;
1654        %% Create EvalResMeasData_theta_X
1655        app.EvalResMeasData_theta_X = uieditfield(app.
            EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.6f
            ');
1656        app.EvalResMeasData_theta_X.HorizontalAlignment = 'center';
1657        app.EvalResMeasData_theta_X.Editable = 'off';
1658        %% Create EvalResMeasData_M_squared_X_Label
1659        app.EvalResMeasData_M_squared_X_Label = uilabel(app.
            EvalResMeasDataParamsGridLayout);
1660        app.EvalResMeasData_M_squared_X_Label.Text = '$\textrm{M}^{2}_{\textrm{x
            }}$';
1661        app.EvalResMeasData_M_squared_X_Label.Interpreter = 'latex';
1662        app.EvalResMeasData_M_squared_X_Label.HorizontalAlignment = 'center';
1663        app.EvalResMeasData_M_squared_X_Label.BackgroundColor =
            backGroundColorLabel;
1664        %% Create EvalResMeasData_M_squared_X
1665        app.EvalResMeasData_M_squared_X = uieditfield(app.
            EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.6f
            ');
1666        app.EvalResMeasData_M_squared_X.HorizontalAlignment = 'center';
1667        app.EvalResMeasData_M_squared_X.Editable = 'off';
1668        %% Create EvalResMeasData_z_0_Y_Label
1669        app.EvalResMeasData_z_0_Y_Label = uilabel(app.
            EvalResMeasDataParamsGridLayout);
1670        app.EvalResMeasData_z_0_Y_Label.Text = '$\textrm{z}_{\textrm{0y}}\textrm
            { in } \mu m$';
1671        app.EvalResMeasData_z_0_Y_Label.Interpreter = 'latex';
1672        app.EvalResMeasData_z_0_Y_Label.HorizontalAlignment = 'center';
1673        app.EvalResMeasData_z_0_Y_Label.BackgroundColor = backGroundColorLabel;
1674        %% Create EvalResMeasData_z_0_Y
1675        app.EvalResMeasData_z_0_Y = uieditfield(app.
            EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
            ');
1676        app.EvalResMeasData_z_0_Y.HorizontalAlignment = 'center';
1677        app.EvalResMeasData_z_0_Y.Editable = 'off';
1678        %% Create EvalResMeasData_z_R_Y_Label
```

```
            app.EvalResMeasData_z_R_Y_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_z_R_Y_Label.Text = '$\textrm{z}_{\textrm{Ry}}\textrm
                { in } \mu m$';
            app.EvalResMeasData_z_R_Y_Label.Interpreter = 'latex';
            app.EvalResMeasData_z_R_Y_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_z_R_Y_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResMeasData_z_R_Y
            app.EvalResMeasData_z_R_Y = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
                ');
            app.EvalResMeasData_z_R_Y.HorizontalAlignment = 'center';
            app.EvalResMeasData_z_R_Y.Editable = 'off';
            %% Create EvalResMeasData_d_0_Y_Label
            app.EvalResMeasData_d_0_Y_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_d_0_Y_Label.Text = '$\textrm{d}_{\textrm{0y}}\textrm
                { in } \mu m$';
            app.EvalResMeasData_d_0_Y_Label.Interpreter = 'latex';
            app.EvalResMeasData_d_0_Y_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_d_0_Y_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResMeasData_d_0_Y
            app.EvalResMeasData_d_0_Y = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.1f
                ');
            app.EvalResMeasData_d_0_Y.HorizontalAlignment = 'center';
            app.EvalResMeasData_d_0_Y.Editable = 'off';
            %% Create EvalResMeasData_theta_Y_Label
            app.EvalResMeasData_theta_Y_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_theta_Y_Label.Text = '$\Theta_{\textrm{y}}\textrm{
                in rad}$';
            app.EvalResMeasData_theta_Y_Label.Interpreter = 'latex';
            app.EvalResMeasData_theta_Y_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_theta_Y_Label.BackgroundColor = backGroundColorLabel
                ;
            %% Create EvalResMeasData_theta_Y
            app.EvalResMeasData_theta_Y = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.6f
                ');
            app.EvalResMeasData_theta_Y.HorizontalAlignment = 'center';
            app.EvalResMeasData_theta_Y.Editable = 'off';
            %% Create EvalResMeasData_M_squared_Y_Label
            app.EvalResMeasData_M_squared_Y_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_M_squared_Y_Label.Text = '$\textrm{M}^{2}_{\textrm{y
                }}$';
            app.EvalResMeasData_M_squared_Y_Label.Interpreter = 'latex';
            app.EvalResMeasData_M_squared_Y_Label.HorizontalAlignment = 'center';
```

```
            app.EvalResMeasData_M_squared_Y_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResMeasData_M_squared_Y
            app.EvalResMeasData_M_squared_Y = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.6f
                ');
            app.EvalResMeasData_M_squared_Y.HorizontalAlignment = 'center';
            app.EvalResMeasData_M_squared_Y.Editable = 'off';
                        %% Create EvalResMeasData_M_squared_eff_Label
            app.EvalResMeasData_M_squared_eff_Label = uilabel(app.
                EvalResMeasDataParamsGridLayout);
            app.EvalResMeasData_M_squared_eff_Label.Text = '$\textrm{M}^{2}_{\textrm
                {eff}}$';
            app.EvalResMeasData_M_squared_eff_Label.Interpreter = 'latex';
            app.EvalResMeasData_M_squared_eff_Label.HorizontalAlignment = 'center';
            app.EvalResMeasData_M_squared_eff_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResMeasData_M_squared_eff
            app.EvalResMeasData_M_squared_eff = uieditfield(app.
                EvalResMeasDataParamsGridLayout,'numeric','ValueDisplayFormat','%.6f
                ');
            app.EvalResMeasData_M_squared_eff.HorizontalAlignment = 'center';
            app.EvalResMeasData_M_squared_eff.Editable = 'off';

            %% Create EvalResMeasDataResultDropDown
            app.EvalResPlotChooserDropDown = uidropdown(app.
                EvalResMeasDataGridLayout);
            app.EvalResPlotChooserDropDown.Layout.Row = 5;
            app.EvalResPlotChooserDropDown.Layout.Column = 1;
            app.EvalResPlotChooserDropDown.FontSize = app.fontsize14;
            app.EvalResPlotChooserDropDown.Items = {'Plot Radius X of all MP'...
                ,'Plot Radius Y of all MP'...
                ,'Plot 2D Intensity Distribution X of selected MP'...
                ,'Plot 2D Intensity Distribution Y of selected MP'...
                ,'Plot 3D Intensity Distribution of selected MP'...
                ,'Plot Spatial Spectrum of selected MP'...
                ,'Plot 2D Spectral Distribution X of selected MP'...
                ,'Plot 2D Spectral Distribution Y of selected MP'...
                ,'Plot Close Up of Beamprofile'...
                ,'Plot Diameter X squared Datapoints with fitted curve'...
                ,'Plot Diameter Y squared Datapoints with fitted curve'};
             %% Create EvalResMeasPlotDivYButton
            app.EvalResMeasPlotButton = uibutton(app.EvalResMeasDataGridLayout);
            app.EvalResMeasPlotButton.Layout.Row = 5;
            app.EvalResMeasPlotButton.Layout.Column = 2;
            app.EvalResMeasPlotButton.Text = 'Plot Selection';
            app.EvalResMeasPlotButton.FontSize = app.fontsize14;
            app.EvalResMeasPlotButton.ButtonPushedFcn = @(src, event)
                EvalResMeasPlotButton_ButtonPushedFcn(app, src, event);
```

```
            %% Create EvalResSaveMeasButton
            app.EvalResSaveMeasButton = uibutton(app.EvalResMeasDataGridLayout);
            app.EvalResSaveMeasButton.Layout.Row = 6;
            app.EvalResSaveMeasButton.Layout.Column = 1;
            app.EvalResSaveMeasButton.Text = 'Save';
            app.EvalResSaveMeasButton.FontSize = app.fontsize14;
            app.EvalResSaveMeasButton.ButtonPushedFcn = @(src,event)
                EvalResSaveMeasButton_ButtonPushedFcn(app,src,event);
            %% Create EvalResExportMeasButton
            app.EvalResExportMeasButton = uibutton(app.EvalResMeasDataGridLayout);
            app.EvalResExportMeasButton.Layout.Row = 6;
            app.EvalResExportMeasButton.Layout.Column = 2;
            app.EvalResExportMeasButton.Text = 'Export Report';
            app.EvalResExportMeasButton.FontSize = app.fontsize14;
            app.EvalResExportMeasButton.Visible = 'off';
            %% Create EvalResHoldPlotButton
            app.EvalResHoldPlotButton = uibutton(app.EvalResMeasDataGridLayout,'
                state');
            app.EvalResHoldPlotButton.Layout.Row = 4;
            app.EvalResHoldPlotButton.Layout.Column = 1;
            app.EvalResHoldPlotButton.Text = 'Hold Data';
            app.EvalResHoldPlotButton.FontSize = app.fontsize14;
            %% Create EvalResSmpDataPanel
            app.EvalResSmpDataPanel = uipanel(app.EvalResTabGridLayout);
            app.EvalResSmpDataGridLayout = uigridlayout(app.EvalResSmpDataPanel);
            app.EvalResSmpDataGridLayout.RowHeight = {30,'1x',60};
            app.EvalResSmpDataGridLayout.ColumnWidth = {'1x','1x'};
            %% Create EvalResSmpDataDropDown_Label
            app.EvalResSmpDataDropDown_Label = uilabel(app.EvalResSmpDataGridLayout)
                ;
            app.EvalResSmpDataDropDown_Label.Layout.Row = 1;
            app.EvalResSmpDataDropDown_Label.Layout.Column = 1;
            app.EvalResSmpDataDropDown_Label.Text = 'Measuring Point No.';
            app.EvalResSmpDataDropDown_Label.BackgroundColor = backGroundColorLabel;
            app.EvalResSmpDataDropDown_Label.FontSize = app.fontsize14;
            app.EvalResSmpDataDropDown_Label.HorizontalAlignment = 'center';
            app.EvalResSmpDataDropDown_Label.Interpreter = 'latex';
            %% Create EvalResSmpDataDropDown
            app.EvalResSmpDataDropDown = uidropdown(app.EvalResSmpDataGridLayout);
            app.EvalResSmpDataDropDown.Layout.Row = 1;
            app.EvalResSmpDataDropDown.Layout.Column = 2;
            app.EvalResSmpDataDropDown.Items = {};
            app.EvalResSmpDataDropDown.ValueChangedFcn = @(src, event)
                EvalResSmpDataDropDown_ValueChangedFcn(app, src, event);
            %% Create EvalResSmpDataParamsGridLayout
            app.EvalResSmpDataParamsGridLayout = uigridlayout(app.
                EvalResSmpDataGridLayout);
            app.EvalResSmpDataParamsGridLayout.Layout.Row = 2;
            app.EvalResSmpDataParamsGridLayout.Layout.Column = [1 2];
```

```
            app.EvalResSmpDataParamsGridLayout.RowHeight =
                {20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20};
            app.EvalResSmpDataParamsGridLayout.ColumnWidth = {'1x','1x'};
            app.EvalResSmpDataParamsGridLayout.Scrollable = 'on';
            %% Create EvalResSmpData_W_X_Label
            app.EvalResSmpData_W_X_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_X_Label.Text = '$\langle x \rangle_{\textrm{mean}}\
                textrm{ in } \mu m$';
            app.EvalResSmpData_W_X_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_X_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_X_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResSmpData_W_X
            app.EvalResSmpData_W_X = uieditfield(app.EvalResSmpDataParamsGridLayout,
                'numeric');
            app.EvalResSmpData_W_X.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_X.Editable = 'off';
            %% Create EvalResSmpData_W_X_std_Label
            app.EvalResSmpData_W_X_std_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_X_std_Label.Text = '$\langle x \rangle_{\textrm{std
                }}\textrm{ in } \mu m$';
            app.EvalResSmpData_W_X_std_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_X_std_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_X_std_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResSmpData_W_X_std
            app.EvalResSmpData_W_X_std = uieditfield(app.
                EvalResSmpDataParamsGridLayout,'numeric');
            app.EvalResSmpData_W_X_std.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_X_std.Editable = 'off';
            %% Create EvalResSmpData_W_Y_Label
            app.EvalResSmpData_W_Y_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_Y_Label.Text = '$\langle y \rangle_{\textrm{mean}}\
                textrm{ in } \mu m$';
            app.EvalResSmpData_W_Y_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_Y_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_Y_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResSmpData_W_Y
            app.EvalResSmpData_W_Y = uieditfield(app.EvalResSmpDataParamsGridLayout,
                'numeric');
            app.EvalResSmpData_W_Y.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_Y.Editable = 'off';
            %% Create EvalResSmpData_W_Y_std_Label
            app.EvalResSmpData_W_Y_std_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_Y_std_Label.Text = '$\langle y \rangle_{\textrm{std
                }}\textrm{ in } \mu m$';
            app.EvalResSmpData_W_Y_std_Label.Interpreter = 'latex';
```

```
1833            app.EvalResSmpData_W_Y_std_Label.HorizontalAlignment = 'center';
1834            app.EvalResSmpData_W_Y_std_Label.BackgroundColor = backGroundColorLabel;
1835            %% Create EvalResSmpData_W_Y_std
1836            app.EvalResSmpData_W_Y_std = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1837            app.EvalResSmpData_W_Y_std.HorizontalAlignment = 'center';
1838            app.EvalResSmpData_W_Y_std.Editable = 'off';
1839            %% Create EvalResSmpData_W_X_squared_Label
1840            app.EvalResSmpData_W_X_squared_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1841            app.EvalResSmpData_W_X_squared_Label.Text = '$\langle x^{2} \rangle_{\
                    textrm{mean}}\textrm{ in } \mu m^{2}$';
1842            app.EvalResSmpData_W_X_squared_Label.Interpreter = 'latex';
1843            app.EvalResSmpData_W_X_squared_Label.HorizontalAlignment = 'center';
1844            app.EvalResSmpData_W_X_squared_Label.BackgroundColor =
                    backGroundColorLabel;
1845            %% Create EvalResSmpData_W_X_squared
1846            app.EvalResSmpData_W_X_squared = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1847            app.EvalResSmpData_W_X_squared.HorizontalAlignment = 'center';
1848            app.EvalResSmpData_W_X_squared.Editable = 'off';
1849            %% Create EvalResSmpData_W_X_squared_std_Label
1850            app.EvalResSmpData_W_X_squared_std_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1851            app.EvalResSmpData_W_X_squared_std_Label.Text = '$\langle x^{2} \rangle_
                    {\textrm{std}}\textrm{ in } \mu m^{2}$';
1852            app.EvalResSmpData_W_X_squared_std_Label.Interpreter = 'latex';
1853            app.EvalResSmpData_W_X_squared_std_Label.HorizontalAlignment = 'center';
1854            app.EvalResSmpData_W_X_squared_std_Label.BackgroundColor =
                    backGroundColorLabel;
1855            %% Create EvalResSmpData_W_X_squared_std
1856            app.EvalResSmpData_W_X_squared_std = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1857            app.EvalResSmpData_W_X_squared_std.HorizontalAlignment = 'center';
1858            app.EvalResSmpData_W_X_squared_std.Editable = 'off';
1859            %% Create EvalResSmpData_W_Y_squared_Label
1860            app.EvalResSmpData_W_Y_squared_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1861            app.EvalResSmpData_W_Y_squared_Label.Text = '$\langle y^{2} \rangle_{\
                    textrm{mean}}\textrm{ in } \mu m^{2}$';
1862            app.EvalResSmpData_W_Y_squared_Label.Interpreter = 'latex';
1863            app.EvalResSmpData_W_Y_squared_Label.HorizontalAlignment = 'center';
1864            app.EvalResSmpData_W_Y_squared_Label.BackgroundColor =
                    backGroundColorLabel;
1865            %% Create EvalResSmpData_W_Y_squared
1866            app.EvalResSmpData_W_Y_squared = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1867            app.EvalResSmpData_W_Y_squared.HorizontalAlignment = 'center';
1868            app.EvalResSmpData_W_Y_squared.Editable = 'off';
```

```
            %% Create EvalResSmpData_W_Y_squared_std_Label
            app.EvalResSmpData_W_Y_squared_std_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_Y_squared_std_Label.Text = '$\langle y^{2} \rangle_
                {\textrm{std}}\textrm{ in } \mu m^{2}$';
            app.EvalResSmpData_W_Y_squared_std_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_Y_squared_std_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_Y_squared_std_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResSmpData_W_Y_squared_std
            app.EvalResSmpData_W_Y_squared_std = uieditfield(app.
                EvalResSmpDataParamsGridLayout,'numeric');
            app.EvalResSmpData_W_Y_squared_std.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_Y_squared_std.Editable = 'off';
            %% Create EvalResSmpData_W_XY_Label
            app.EvalResSmpData_W_XY_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_XY_Label.Text = '$\langle xy \rangle_{\textrm{mean
                }}\textrm{ in } \mu m^{2}$';
            app.EvalResSmpData_W_XY_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_XY_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_XY_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResSmpData_W_XY
            app.EvalResSmpData_W_XY = uieditfield(app.EvalResSmpDataParamsGridLayout
                ,'numeric');
            app.EvalResSmpData_W_XY.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_XY.Editable = 'off';
            %% Create EvalResSmpData_W_XY_std_Label
            app.EvalResSmpData_W_XY_std_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_W_XY_std_Label.Text = '$\langle xy \rangle_{\textrm{
                std}}\textrm{ in } \mu m^{2}$';
            app.EvalResSmpData_W_XY_std_Label.Interpreter = 'latex';
            app.EvalResSmpData_W_XY_std_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_XY_std_Label.BackgroundColor = backGroundColorLabel
                ;
            %% Create EvalResSmpData_W_XY_std
            app.EvalResSmpData_W_XY_std = uieditfield(app.
                EvalResSmpDataParamsGridLayout,'numeric');
            app.EvalResSmpData_W_XY_std.HorizontalAlignment = 'center';
            app.EvalResSmpData_W_XY_std.Editable = 'off';
            %% Create EvalResSmpData_dWx_Label
            app.EvalResSmpData_dWx_Label = uilabel(app.
                EvalResSmpDataParamsGridLayout);
            app.EvalResSmpData_dWx_Label.Text = '$\textrm{Beamdiameter } d_{x\textrm
                { mean}} \textrm{ in } \mu m$';
            app.EvalResSmpData_dWx_Label.Interpreter = 'latex';
            app.EvalResSmpData_dWx_Label.HorizontalAlignment = 'center';
            app.EvalResSmpData_dWx_Label.BackgroundColor = backGroundColorLabel;
```

```
1905            %% Create EvalResSmpData_dWx
1906            app.EvalResSmpData_dWx = uieditfield(app.EvalResSmpDataParamsGridLayout,
                    'numeric');
1907            app.EvalResSmpData_dWx.HorizontalAlignment = 'center';
1908            app.EvalResSmpData_dWx.Editable = 'off';
1909            %% Create EvalResSmpData_dWx_std_Label
1910            app.EvalResSmpData_dWx_std_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1911            app.EvalResSmpData_dWx_std_Label.Text = '$\textrm{Beamdiameter } d_{x\
                    textrm{ std}} \textrm{ in } \mu m$';
1912            app.EvalResSmpData_dWx_std_Label.Interpreter = 'latex';
1913            app.EvalResSmpData_dWx_std_Label.HorizontalAlignment = 'center';
1914            app.EvalResSmpData_dWx_std_Label.BackgroundColor = backGroundColorLabel;
1915            %% Create EvalResSmpData_dWx_std
1916            app.EvalResSmpData_dWx_std = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1917            app.EvalResSmpData_dWx_std.HorizontalAlignment = 'center';
1918            app.EvalResSmpData_dWx_std.Editable = 'off';
1919            %% Create EvalResSmpData_dWy_Label
1920            app.EvalResSmpData_dWy_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1921            app.EvalResSmpData_dWy_Label.Text = '$\textrm{Beamdiameter } d_{y\textrm
                    { mean}} \textrm{ in } \mu m$';
1922            app.EvalResSmpData_dWy_Label.Interpreter = 'latex';
1923            app.EvalResSmpData_dWy_Label.HorizontalAlignment = 'center';
1924            app.EvalResSmpData_dWy_Label.BackgroundColor = backGroundColorLabel;
1925            %% Create EvalResSmpData_dWy
1926            app.EvalResSmpData_dWy = uieditfield(app.EvalResSmpDataParamsGridLayout,
                    'numeric');
1927            app.EvalResSmpData_dWy.HorizontalAlignment = 'center';
1928            app.EvalResSmpData_dWy.Editable = 'off';
1929            %% Create EvalResSmpData_dWy_std_Label
1930            app.EvalResSmpData_dWy_std_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1931            app.EvalResSmpData_dWy_std_Label.Text = '$\textrm{Beamdiameter } d_{y\
                    textrm{ std}} \textrm{ in } \mu m$';
1932            app.EvalResSmpData_dWy_std_Label.Interpreter = 'latex';
1933            app.EvalResSmpData_dWy_std_Label.HorizontalAlignment = 'center';
1934            app.EvalResSmpData_dWy_std_Label.BackgroundColor = backGroundColorLabel;
1935            %% Create EvalResSmpData_dWy_std
1936            app.EvalResSmpData_dWy_std = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1937            app.EvalResSmpData_dWy_std.HorizontalAlignment = 'center';
1938            app.EvalResSmpData_dWy_std.Editable = 'off';
1939            %% Create EvalResSmpData_phi_Label
1940            app.EvalResSmpData_phi_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1941            app.EvalResSmpData_phi_Label.Text = '$\textrm{Azimutangle } \varphi_{\
                    textrm{mean}} \textrm{ in } Â°$';
```

```
1942            app.EvalResSmpData_phi_Label.Interpreter = 'latex';
1943            app.EvalResSmpData_phi_Label.HorizontalAlignment = 'center';
1944            app.EvalResSmpData_phi_Label.BackgroundColor = backGroundColorLabel;
1945            %% Create EvalResSmpData_phi
1946            app.EvalResSmpData_phi = uieditfield(app.EvalResSmpDataParamsGridLayout,
                    'numeric');
1947            app.EvalResSmpData_phi.HorizontalAlignment = 'center';
1948            app.EvalResSmpData_phi.Editable = 'off';
1949            %% Create EvalResSmpData_phi_std_Label
1950            app.EvalResSmpData_phi_std_Label = uilabel(app.
                    EvalResSmpDataParamsGridLayout);
1951            app.EvalResSmpData_phi_std_Label.Text = '$\textrm{Azimutangle } \varphi_
                    {\textrm{std}} \textrm{ in } Â°$';
1952            app.EvalResSmpData_phi_std_Label.Interpreter = 'latex';
1953            app.EvalResSmpData_phi_std_Label.HorizontalAlignment = 'center';
1954            app.EvalResSmpData_phi_std_Label.BackgroundColor = backGroundColorLabel;
1955            %% Create EvalResSmpData_phi_std
1956            app.EvalResSmpData_phi_std = uieditfield(app.
                    EvalResSmpDataParamsGridLayout,'numeric');
1957            app.EvalResSmpData_phi_std.HorizontalAlignment = 'center';
1958            app.EvalResSmpData_phi_std.Editable = 'off';
1959            %% Create EvalResSmpData_phi_std
1960            app.EvalResSmpInfoTextArea = uitextarea(app.EvalResSmpDataGridLayout);
1961            app.EvalResSmpInfoTextArea.HorizontalAlignment = 'center';
1962            app.EvalResSmpInfoTextArea.Editable = 'off';
1963            app.EvalResSmpInfoTextArea.Layout.Row = 3;
1964            app.EvalResSmpInfoTextArea.Layout.Column = [1 2];
1965            app.EvalResSmpInfoTextArea.Value ='Two or more measuring points have an
                    azimuth angle difference of 10Â° or more. Could be badly conditioned
                     evaluation or a astigmatic laserbeam. Either way the results cannot
                     be trusted.';
1966            app.EvalResSmpInfoTextArea.Visible = 'off';
1967            app.EvalResSmpInfoTextArea.FontSize = 14;
1968            app.EvalResSmpInfoTextArea.FontColor = 'red';
1969            app.EvalResSmpInfoTextArea.FontWeight = 'bold';
1970            %% Create EvalResImDataPanel
1971            app.EvalResImDataPanel = uipanel(app.EvalResTabGridLayout);
1972            app.EvalResImDataGridLayout = uigridlayout(app.EvalResImDataPanel);
1973            app.EvalResImDataGridLayout.RowHeight = {30,240,'1x',50};
1974            app.EvalResImDataGridLayout.ColumnWidth = {'1x','1x'};
1975            %% Create EvalResImDataDropDown_Label
1976            app.EvalResImDataDropDown_Label = uilabel(app.EvalResImDataGridLayout);
1977            app.EvalResImDataDropDown_Label.Layout.Row = 1;
1978            app.EvalResImDataDropDown_Label.Layout.Column = 1;
1979            app.EvalResImDataDropDown_Label.Text = 'Image No.';
1980            app.EvalResImDataDropDown_Label.BackgroundColor = backGroundColorLabel;
1981            app.EvalResImDataDropDown_Label.FontSize = app.fontsize14;
1982            app.EvalResImDataDropDown_Label.HorizontalAlignment = 'center';
1983            app.EvalResImDataDropDown_Label.Interpreter = 'latex';
```

```
1984            %% Create EvalResImDataDropDown
1985            app.EvalResImDataDropDown    = uidropdown(app.EvalResImDataGridLayout);
1986            app.EvalResImDataDropDown.Layout.Row = 1;
1987            app.EvalResImDataDropDown.Layout.Column = 2;
1988            app.EvalResImDataDropDown.Items = {};
1989            app.EvalResImDataDropDown.ValueChangedFcn = @(src, event)
                    EvalResImDataDropDown_ValueChangedFcn(app, src, event);
1990
1991            %% Create EvalResImDataParamsGridLayout
1992            app.EvalResImDataParamsGridLayout = uigridlayout(app.
                    EvalResImDataGridLayout);
1993            app.EvalResImDataParamsGridLayout.Layout.Row = 2;
1994            app.EvalResImDataParamsGridLayout.Layout.Column = [1 2];
1995            app.EvalResImDataParamsGridLayout.RowHeight = {20,20,20,20,20,20,20,20};
1996            app.EvalResImDataParamsGridLayout.ColumnWidth = {'1x','1x'};
1997            app.EvalResImDataParamsGridLayout.Scrollable = 'on';
1998            %% Create EvalResImData_W_X_Label
1999            app.EvalResImData_W_X_Label = uilabel(app.EvalResImDataParamsGridLayout)
                    ;
2000            app.EvalResImData_W_X_Label.Text = '$\langle x \rangle\textrm{ in } \mu
                    m$';
2001            app.EvalResImData_W_X_Label.Interpreter = 'latex';
2002            app.EvalResImData_W_X_Label.HorizontalAlignment = 'center';
2003            app.EvalResImData_W_X_Label.BackgroundColor = backGroundColorLabel;
2004            %% Create EvalResImData_W_X
2005            app.EvalResImData_W_X = uieditfield(app.EvalResImDataParamsGridLayout,'
                    numeric');
2006            app.EvalResImData_W_X.HorizontalAlignment = 'center';
2007            app.EvalResImData_W_X.Editable = 'off';
2008            %% Create EvalResImData_W_Y_Label
2009            app.EvalResImData_W_Y_Label = uilabel(app.EvalResImDataParamsGridLayout)
                    ;
2010            app.EvalResImData_W_Y_Label.Text = '$\langle y \rangle\textrm{ in } \mu
                    m$';
2011            app.EvalResImData_W_Y_Label.Interpreter = 'latex';
2012            app.EvalResImData_W_Y_Label.HorizontalAlignment = 'center';
2013            app.EvalResImData_W_Y_Label.BackgroundColor = backGroundColorLabel;
2014            %% Create EvalResImData_W_Y
2015            app.EvalResImData_W_Y = uieditfield(app.EvalResImDataParamsGridLayout,'
                    numeric');
2016            app.EvalResImData_W_Y.HorizontalAlignment = 'center';
2017            app.EvalResImData_W_Y.Editable = 'off';
2018            %% Create EvalResImData_W_X_squared_Label
2019            app.EvalResImData_W_X_squared_Label = uilabel(app.
                    EvalResImDataParamsGridLayout);
2020            app.EvalResImData_W_X_squared_Label.Text = '$\langle x^{2} \rangle\
                    textrm{ in } \mu m^{2}$';
2021            app.EvalResImData_W_X_squared_Label.Interpreter = 'latex';
2022            app.EvalResImData_W_X_squared_Label.HorizontalAlignment = 'center';
```

```
            app.EvalResImData_W_X_squared_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResImData_W_X_squared
            app.EvalResImData_W_X_squared = uieditfield(app.
                EvalResImDataParamsGridLayout,'numeric');
            app.EvalResImData_W_X_squared.HorizontalAlignment = 'center';
            app.EvalResImData_W_X_squared.Editable = 'off';
            %% Create EvalResImData_W_Y_squared_Label
            app.EvalResImData_W_Y_squared_Label = uilabel(app.
                EvalResImDataParamsGridLayout);
            app.EvalResImData_W_Y_squared_Label.Text = '$\langle y^{2} \rangle\
                textrm{ in } \mu m^{2}$';
            app.EvalResImData_W_Y_squared_Label.Interpreter = 'latex';
            app.EvalResImData_W_Y_squared_Label.HorizontalAlignment = 'center';
            app.EvalResImData_W_Y_squared_Label.BackgroundColor =
                backGroundColorLabel;
            %% Create EvalResImData_W_Y_squared
            app.EvalResImData_W_Y_squared = uieditfield(app.
                EvalResImDataParamsGridLayout,'numeric');
            app.EvalResImData_W_Y_squared.HorizontalAlignment = 'center';
            app.EvalResImData_W_Y_squared.Editable = 'off';
            %% Create EvalResImData_W_XY_Label
            app.EvalResImData_W_XY_Label = uilabel(app.EvalResImDataParamsGridLayout
                );
            app.EvalResImData_W_XY_Label.Text = '$\langle xy \rangle\textrm{ in } \
                mu m^{2}$';
            app.EvalResImData_W_XY_Label.Interpreter = 'latex';
            app.EvalResImData_W_XY_Label.HorizontalAlignment = 'center';
            app.EvalResImData_W_XY_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResImData_W_XY
            app.EvalResImData_W_XY = uieditfield(app.EvalResImDataParamsGridLayout,'
                numeric');
            app.EvalResImData_W_XY.HorizontalAlignment = 'center';
            app.EvalResImData_W_XY.Editable = 'off';
            %% Create EvalResImData_dWx_Label
            app.EvalResImData_dWx_Label = uilabel(app.EvalResImDataParamsGridLayout)
                ;
            app.EvalResImData_dWx_Label.Text = '$\textrm{Beamdiameter } d_{x} \
                textrm{ in } \mu m$';
            app.EvalResImData_dWx_Label.Interpreter = 'latex';
            app.EvalResImData_dWx_Label.HorizontalAlignment = 'center';
            app.EvalResImData_dWx_Label.BackgroundColor = backGroundColorLabel;
            %% Create EvalResImData_dWx
            app.EvalResImData_dWx = uieditfield(app.EvalResImDataParamsGridLayout,'
                numeric');
            app.EvalResImData_dWx.HorizontalAlignment = 'center';
            app.EvalResImData_dWx.Editable = 'off';
            %% Create EvalResImData_dWy_Label
            app.EvalResImData_dWy_Label = uilabel(app.EvalResImDataParamsGridLayout)
```

```
            ;
app.EvalResImData_dWy_Label.Text = '$\textrm{Beamdiameter } d_{y} \
    textrm{ in } \mu m$';
app.EvalResImData_dWy_Label.Interpreter = 'latex';
app.EvalResImData_dWy_Label.HorizontalAlignment = 'center';
app.EvalResImData_dWy_Label.BackgroundColor = backGroundColorLabel;
%% Create EvalResImData_dWy
app.EvalResImData_dWy = uieditfield(app.EvalResImDataParamsGridLayout,'
    numeric');
app.EvalResImData_dWy.HorizontalAlignment = 'center';
app.EvalResImData_dWy.Editable = 'off';
%% Create EvalResImData_phi_Label
app.EvalResImData_phi_Label = uilabel(app.EvalResImDataParamsGridLayout)
    ;
app.EvalResImData_phi_Label.Text = '$\textrm{Azimutangle } \varphi \
    textrm{ in } Â°$';
app.EvalResImData_phi_Label.Interpreter = 'latex';
app.EvalResImData_phi_Label.HorizontalAlignment = 'center';
app.EvalResImData_phi_Label.BackgroundColor = backGroundColorLabel;
%% Create EvalResImData_phi
app.EvalResImData_phi = uieditfield(app.EvalResImDataParamsGridLayout,'
    numeric');
app.EvalResImData_phi.HorizontalAlignment = 'center';
app.EvalResImData_phi.Editable = 'off';
%% Create EvalResImDataUIAxes
app.EvalResImDataUIAxes  = uiaxes(app.EvalResImDataGridLayout);
app.EvalResImDataUIAxes.Layout.Row = 3;
app.EvalResImDataUIAxes.Layout.Column = [1 2];
%% Create EvalResImDataColorGridLayout
app.EvalResImDataColorGridLayout = uigridlayout(app.
    EvalResImDataGridLayout);
app.EvalResImDataColorGridLayout.Layout.Row = 4;
app.EvalResImDataColorGridLayout.Layout.Column = [1 2];
app.EvalResImDataColorGridLayout.RowHeight = {'1x'};
app.EvalResImDataColorGridLayout.ColumnWidth = {120,'1x','1x','1x','1x'
    };
app.EvalResImDataColorGridLayout.Scrollable = 'on';
%% Create EvalResImDataChangeColorMapButton
app.EvalResImDataChangeColorMapButton = uibutton(app.
    EvalResImDataColorGridLayout,'state');
app.EvalResImDataChangeColorMapButton.Text = 'Switch Color Map';
app.EvalResImDataChangeColorMapButton.Layout.Row = 1;
app.EvalResImDataChangeColorMapButton.Layout.Column = 1;
app.EvalResImDataChangeColorMapButton.ValueChangedFcn = @(src, event)
    EvalResImDataChangeColorMapButton_ValueChangedFcn(app, src, event);
app.EvalResImDataChangeColorMapButton.FontSize = app.fontsize14;
%% Create EvalResColorMinLabel
app.EvalResColorMinLabel = uilabel(app.EvalResImDataColorGridLayout);
app.EvalResColorMinLabel.Text = 'Min Value';
```

```
2099            app.EvalResColorMinLabel.Layout.Row = 1;
2100            app.EvalResColorMinLabel.Layout.Column = 2;
2101            app.EvalResColorMinLabel.FontSize = app.fontsize14;
2102            app.EvalResColorMinLabel.Interpreter = 'latex';
2103            app.EvalResColorMinLabel.BackgroundColor = backGroundColorLabel;
2104            app.EvalResColorMinLabel.HorizontalAlignment = 'center';
2105            %% Create EvalResColorMinEditField
2106            app.EvalResColorMinEditField = uieditfield(app.
                    EvalResImDataColorGridLayout,'numeric');
2107            app.EvalResColorMinEditField.Layout.Row = 1;
2108            app.EvalResColorMinEditField.Layout.Column = 3;
2109            app.EvalResColorMinEditField.Limits = [0 255];
2110            app.EvalResColorMinEditField.HorizontalAlignment = 'center';
2111            app.EvalResColorMinEditField.FontSize = app.fontsize14;
2112            app.EvalResColorMinEditField.Value = app.EvalResColorMap(1);
2113            app.EvalResColorMinEditField.ValueChangedFcn = @(src, event)
                    EvalResImChangeColorMap(app, src, event);
2114
2115            %% Create LiveMeasColorMaxLabel
2116            app.EvalResColorMaxLabel = uilabel(app.EvalResImDataColorGridLayout);
2117            app.EvalResColorMaxLabel.Text = 'Max Value';
2118            app.EvalResColorMaxLabel.Layout.Row = 1;
2119            app.EvalResColorMaxLabel.Layout.Column = 4;
2120            app.EvalResColorMaxLabel.FontSize = app.fontsize14;
2121            app.EvalResColorMaxLabel.Interpreter = 'latex';
2122            app.EvalResColorMaxLabel.BackgroundColor = backGroundColorLabel;
2123            app.EvalResColorMaxLabel.HorizontalAlignment = 'center';
2124            %% Create LiveMeasColorMinEditField
2125            app.EvalResColorMaxEditField = uieditfield(app.
                    EvalResImDataColorGridLayout,'numeric');
2126            app.EvalResColorMaxEditField.Layout.Row = 1;
2127            app.EvalResColorMaxEditField.Layout.Column = 5;
2128            app.EvalResColorMaxEditField.Limits = [0 255];
2129            app.EvalResColorMaxEditField.HorizontalAlignment = 'center';
2130            app.EvalResColorMaxEditField.FontSize = app.fontsize14;
2131            app.EvalResColorMaxEditField.Value = app.EvalResColorMap(2);
2132            app.EvalResColorMaxEditField.ValueChangedFcn = @(src, event)
                    EvalResImChangeColorMap(app, src, event);
2133            %% Show the figure after all components are created
2134            app.UIFigure.Visible = 'on';
2135        end
2136    end
2137
2138    % App creation and deletion
2139    methods (Access = public)
2140
2141        % Construct app
2142        function app = lbcUI
2143
```

```
            % Create UIFigure and components
            createComponents(app)


            % Register the app with App Designer
            registerApp(app, app.UIFigure);
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            imaqreset()
            delete(app.UIFigure)

        end
    end
end
```

7.31: Function AutoMeasISO111146Part1MethodRBGroup_SelectionChangedFcn

```
function AutoMeasISO111146Part1MethodRBGroup_SelectionChangedFcn(app, src, event)
%AutoMeasISO111146Part1MethodRBGroup_SelectionChangedFcn ValueChangedFcn listening
    to changes of the selected Automated
%Measurement Configuration RadioButtonGroup.
%   Enables and Disables UI Elements depending on the selected
%   Configuration Method and triggers an update of the Measuring Point
%   preview axes.
newSelection = event.NewValue;

if strcmp('Configure around centerpoint', newSelection.Text)
    app.AutoMeasISO111146Part1MeasAreaEditField.Enable = 'off';
    app.AutoMeasISO111146Part1OffseEditField.Enable = 'on';
    app.AutoMeasISO111146Part1RayleighEditField.Enable = 'on';
    app.AutoMeasISO111146Part1TimesRayleighEditField.Enable = 'on';
else
    app.AutoMeasISO111146Part1MeasAreaEditField.Enable = 'on';
    app.AutoMeasISO111146Part1OffseEditField.Enable = 'off';
    app.AutoMeasISO111146Part1RayleighEditField.Enable = 'off';
    app.AutoMeasISO111146Part1TimesRayleighEditField.Enable = 'off';
end
UpdatePreviewAxes(app, src, event);
end
```

7.32: Function AutoMeasISO111146Part1StartMeasButton_ButtonPushedFcn

```
function AutoMeasISO111146Part1StartMeasButton_ButtonPushedFcn(app,src,event)
%AutoMeasISO111146Part1StartMeasButton_ButtonPushedFcn ButtonPushedFcn listening to
    the Start Button of the Automated
```

```
% Measurement.
%   Starts the automated measuerement using the parameters given in the
%   EditFields of the UI.
try
    app.AutoMeasISO111146Part1StartMeasButton.Enable = 'off';
    if app.AutoMeasCancelButton.Value
        app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
        app.AutoMeasCancelButton.Value = 0;
        app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [1 0.07 0.2];
        app.AutoMeasISO111146Part1StatusLabel.Text = 'Inactive';
        uialert(app.UIFigure,{'Measurement canceled!'},'Measurement Stopped
            Information','Icon','success','CloseFcn','uiresume(gcbf)');
        uiwait(gcbf)
        return;
    end
    measurement = Measurement(app.AutoMeasISO111146Part1RayleighEditField.Value,...
        app.AutoMeasISO111146Part1TimesRayleighEditField.Value,...
        app.AutoMeasISO111146Part1NoMeasPointsEditField.Value,...
        app.AutoMeasISO111146Part1MeasPerPointEditField.Value);

    % Reset old Measurementdata
    measurement.reset();

    app.AutoMeasISO111146Part1StatusLabel.Text = 'Active';
    app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [0.2 1 0.3];

    app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Go to zero';
    drawnow;
    moveToZpos(app,0);
    pause(1);
    uialert(app.UIFigure,{'Cover laser!'},'Next Step Information','Icon','info','
        CloseFcn','uiresume(gcbf)');
    uiwait(gcbf)
    app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Background
        Measurement';
    for i = 1 : size(app.measurementPoints,2)
        app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = '—';
        app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value = '—';
        drawnow;
        moveToZpos(app,app.measurementPoints(i));
        pause(1);
        app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = int2str(i);

        MeasurementPoint = SingleMeasurementPoint(app.currPos(1));
        for k = 1 : app.AutoMeasISO111146Part1MeasPerPointEditField.Value
            if app.AutoMeasCancelButton.Value
                app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
                app.AutoMeasCancelButton.Value = 0;
                app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [1 0.07
```

```
                    0.2];
                app.AutoMeasISO111146Part1StatusLabel.Text = 'Inactive';
                uialert(app.UIFigure,{'Measurement canceled!'},'Measurement Stopped
                    Information','Icon','success','CloseFcn','uiresume(gcbf)');
                uiwait(gcbf)
                return;
            end
            app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value =
                int2str(k);
            MeasurementPoint.addMeasurement(SingleMeasurement(TriggerCameraImage(app
                )));
            pause(1);
        end
        measurement.getBackgroundData().addSingleMeasurementPoint(MeasurementPoint);
    end
    app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Go to zero';
    app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = '—';
    app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value = '—';
    drawnow;
    moveToZpos(app,0);
    pause(1);
    uialert(app.UIFigure,{'Remove laser cover!'},'Next Step Information','Icon','
        info','CloseFcn','uiresume(gcbf)');
    uiwait(gcbf)
    app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Laser Measurement
        ';
    for i = 1 : size(app.measurementPoints,2)
        app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = '—';
        app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value = '—';
        drawnow;
        moveToZpos(app,app.measurementPoints(i));
        pause(1);
        app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = int2str(i);
        MeasurementPoint = SingleMeasurementPoint(app.currPos(1));
        for k = 1 : app.AutoMeasISO111146Part1MeasPerPointEditField.Value
            if app.AutoMeasCancelButton.Value
                app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
                app.AutoMeasCancelButton.Value = 0;
                app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [1 0.07
                    0.2];
                app.AutoMeasISO111146Part1StatusLabel.Text = 'Inactive';
                uialert(app.UIFigure,{'Measurement canceled!'},'Measurement Stopped
                    Information','Icon','success','CloseFcn','uiresume(gcbf)');
                uiwait(gcbf)
                return;
            end
            app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value =
                int2str(k);
            MeasurementPoint.addMeasurement(SingleMeasurement(TriggerCameraImage(app
```

```
                )));
89              pause(1);
90          end
91          measurement.getMeasurementData().addSingleMeasurementPoint(MeasurementPoint)
                ;
92      end
93      app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Go to zero';
94      app.AutoMeasISO111146Part1ProgressCurrMeasPointEditField.Value = '-';
95      app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value = '-';
96      drawnow;
97      moveToZpos(app,0);
98      pause(1)
99      app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Saving
            Measurement';
100     drawnow;
101     uisave({'measurement'});
102     app.AutoMeasISO111146Part1ProgressCurrActionEditField.Value = 'Finished';
103     drawnow;
104
105     app.AutoMeasISO111146Part1StatusLabel.BackgroundColor = [1 0.07 0.2];
106     app.AutoMeasISO111146Part1StatusLabel.Text = 'Inactive';
107
108     uialert(app.UIFigure,{'Measurement finished successfull!'},'Measurement
            Successfull Information','Icon','success','CloseFcn','uiresume(gcbf)');
109     uiwait(gcbf)
110
111     app.loadedMeas = measurement;
112     UpdateGUIConfEvalTab(app);
113
114     if not(isempty(app.loadedMeas.getResults()))
115         UpdateGUIEvalResTab(app);
116     end
117
118 catch ME
119     app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
120     rethrow(ME);
121 end
122 app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
123 end
```

7.33: Function CalcNumOfSteps

```
1  function steps = CalcNumOfSteps(app,targetPos)
2  %CALCNUMOFSTEPS Calculates the number of Steps the Steppermotor has to do.
3  %   Nothing to add here.
4  steps=cast((targetPos-app.currPos(1))/app.StepInkrement,'int32');
5  end
```

7.34: Function ClearGUIEvalResTab

```
function ClearGUIEvalResTab(app)
%ClearGUIEvalResTab Clears all Evaluation Result UI elements of their
%content.
app.EvalResMeasDataResultDropDown.Items = {};
app.EvalResMeasDataResultDropDown.ItemsData = [];

app.EvalResMeasData_BackCorrMethod.Value = '';
app.EvalResMeasData_Lambda.Value = 0;
app.EvalResMeasData_ConvCrit.Value = 0;
app.EvalResMeasData_IntegAreaFactor.Value = 0;
app.EvalResMeasData_Nt.Value = 0;
app.EvalResMeasData_EvalNorm.Value = '';

app.EvalResMeasData_z_0_X.Value = 0;
app.EvalResMeasData_z_0_Y.Value = 0;

 app.EvalResMeasData_z_R_X.Value = 0;
app.EvalResMeasData_z_R_Y.Value = 0;

app.EvalResMeasData_d_0_X.Value = 0;
app.EvalResMeasData_d_0_Y.Value = 0;

 app.EvalResMeasData_theta_X.Value = 0;
 app.EvalResMeasData_theta_Y.Value = 0;

 app.EvalResMeasData_M_squared_X.Value = 0;
 app.EvalResMeasData_M_squared_Y.Value = 0;
 app.EvalResMeasData_M_squared_eff.Value = 0;

app.EvalResSmpDataDropDown.Items = {};
app.EvalResSmpDataDropDown.ItemsData = [];

app.EvalResSmpData_W_X.Value = 0;
app.EvalResSmpData_W_X_std.Value = 0;
app.EvalResSmpData_W_Y.Value = 0;
app.EvalResSmpData_W_Y_std.Value = 0;
app.EvalResSmpData_W_X_squared.Value = 0;
app.EvalResSmpData_W_X_squared_std.Value = 0;
app.EvalResSmpData_W_Y_squared.Value = 0;
app.EvalResSmpData_W_Y_squared_std.Value = 0;
app.EvalResSmpData_W_XY.Value = 0;
app.EvalResSmpData_W_XY_std.Value = 0;
app.EvalResSmpData_dWx.Value = 0;
app.EvalResSmpData_dWx_std.Value = 0;
app.EvalResSmpData_dWy.Value = 0;
app.EvalResSmpData_dWy_std.Value = 0;
app.EvalResSmpData_phi.Value = 0;
app.EvalResSmpData_phi_std.Value = 0;

```

```
app.EvalResImData_W_X.Value = 0;
app.EvalResImData_W_Y.Value = 0;

app.EvalResImData_W_X_squared.Value = 0;
app.EvalResImData_W_Y_squared.Value = 0;

app.EvalResImData_W_XY.Value = 0;
app.EvalResImData_dWx.Value = 0;
app.EvalResImData_dWy.Value = 0;
app.EvalResImData_phi.Value = 0;

cla(app.EvalResImDataUIAxes,'reset');

end
```

7.35: Function ConfEvalBackCorrMethodDropDown_ValueChangedFcn

```
function ConfEvalBackCorrMethodDropDown_ValueChangedFcn(app, src, event)
%ConfEvalBackCorrMethodDropDown_ValueChangedFcn ValueChangedFcn listening
%to changes of the BackgroundCorrectionMethod Dropdown Menu
newCorrMethod = event.Value;
if strcmp('FCBAM',newCorrMethod)
    app.ConfEvalKernelSizeSlider.Enable = 'on';
    app.ConfEvalKernelSizeEditField.Enable = 'on';
    return;
end
app.ConfEvalKernelSizeSlider.Enable = 'off';
app.ConfEvalKernelSizeEditField.Enable = 'off';
end
```

7.36: Function ConfEvalConvergeCriterionSlider_AllEvents

```
function ConfEvalConvergeCriterionSlider_AllEvents(app, src, event)
%ConfEvalConvergeCriterionSlider_AllEvents listening to all events of the
%ConvergeCriterion UI Elements.
eventName = event.EventName;
switch(eventName)
    case {'ValueChanging'}
        app.ConfEvalConvergeCriterionEditField.Value = event.Value;
    case {'ValueChanged'}
        app.ConfEvalConvergeCriterionSlider.Value = event.Value;

end

end
```

7.37: Function ConfEvalDrawRectButton_ButtonPushedFcn

```
function ConfEvalDrawRectButton_ButtonPushedFcn(app, src, event)
%ConfEvalDrawRectButton_ButtonPushedFcn listening to guessBeamarea button
%to draw the first guess of the beam area and beamcenter.
```

```
if not(isa(app.ConfEvalRectRoi,'images.roi.Rectangle')) || not(isvalid(app.
    ConfEvalRectRoi))
    app.ConfEvalRectRoi = drawrectangle(app.ConfEvalPreviewImUIAxes,'LabelVisible','
        hover',...
        'Label','beamareaGuess','Position',[1 1 200 200]);
    addlistener(app.ConfEvalRectRoi,'MovingROI',@(src, evt)
        drawConfEvalRectAllevents(app,src,evt));
    addlistener(app.ConfEvalRectRoi,'ROIMoved',@(src, evt) drawConfEvalRectAllevents
        (app,src,evt));
    centerRectangleX = app.ConfEvalRectRoi.Position(1)+app.ConfEvalRectRoi.Position
        (3)/2;
    centerRectangleY = app.ConfEvalRectRoi.Position(2)+app.ConfEvalRectRoi.Position
        (4)/2;
    app.ConfEvalPointRoi = drawpoint(app.ConfEvalPreviewImUIAxes,...
        'Position',[centerRectangleX centerRectangleY],'Color','r',...
        'InteractionsAllowed','none');
end
end
```

7.38: Function ConfEvalEvaluationButton_ButtonPushedFcn

```
function ConfEvalEvaluationButton_ButtonPushedFcn(app, src, event)
%ConfEvalEvaluationButton_ButtonPushedFcn listening to the Evaluate Button
%and starting a evaluation using the specified parameters in the UI.
try
    app.ConfEvalEvaluationButton.Enable = 'off';
    app.ConfEvalEvaluationStatusLabel.BackgroundColor = [0.2 1 0.3];
    app.ConfEvalEvaluationStatusLabel.Text = 'Computing';
    drawnow;
    coarseBackCorrMethod = app.ConfEvalBackCorrMethodDropDown.Value;
    fineBackCorrMethod = app.ConfEvalFineBackCorrMethodDropDown.Value;
    convCrit = app.ConfEvalConvergeCriterionEditField.Value;
    lambda = app.ConfEvalWavelengthEditField.Value;
    nT = app.ConfEvalNtEditField.Value;
    integAreaFactor = app.ConfEvalIntegAreaEditField.Value;
    evalNorm = app.ConfEvalMethodDropDown.Value;
    pixelsize = app.ConfEvalPixelsizeEditField.Value;
    kernelSize = app.ConfEvalKernelSizeEditField.Value;
    centerRectangleX = app.ConfEvalRectRoi.Position(1)+app.ConfEvalRectRoi.Position
        (3)/2;
    centerRectangleY = app.ConfEvalRectRoi.Position(2)+app.ConfEvalRectRoi.Position
        (4)/2;
    sizeRectangleX = app.ConfEvalRectRoi.Position(3);
    sizeRectangleY = app.ConfEvalRectRoi.Position(4);
    manualValue = app.ConfEvalFineCorrValueEditField.Value;

    corrData = app.loadedMeas.getMeasurementData().
        backgroundCorrectionMeasurementData(...
        coarseBackCorrMethod,fineBackCorrMethod,app.loadedMeas.getBackgroundData(),
            ...
```

```
        kernelSize,nT,manualValue);
    switch(evalNorm)
        case 'ISO11146-1'
            resData = corrData.iso11146Part1MeasurementData(centerRectangleX*
                pixelsize,centerRectangleY*pixelsize,sizeRectangleX*pixelsize,
                sizeRectangleY*pixelsize,convCrit,integAreaFactor,lambda,pixelsize);
            res = Result(coarseBackCorrMethod,fineBackCorrMethod,evalNorm,lambda,
                convCrit,nT,integAreaFactor,kernelSize,resData);
            app.loadedMeas.addResult(res);
            UpdateGUIEvalResTab(app);
        case 'ISO11146-2'
    end
    uialert(app.UIFigure,{'Evaluation finished!'},'Next Step Information','Icon','
        success','CloseFcn','uiresume(gcbf)');
    uiwait(gcbf)
catch ME
    uialert(app.UIFigure,{'Evaluation failed! Check Evaluation Parameters!'},'Next
        Step Information','Icon','warning','CloseFcn','uiresume(gcbf)');
    uiwait(gcbf)
    app.ConfEvalEvaluationButton.Enable = 'on';
    app.ConfEvalEvaluationStatusLabel.BackgroundColor = [1 0.07 0.2];
    app.ConfEvalEvaluationStatusLabel.Text = 'Standby';
    drawnow;
    rethrow(ME)
end
app.ConfEvalEvaluationButton.Enable = 'on';

    app.ConfEvalEvaluationStatusLabel.BackgroundColor = [1 0.07 0.2];
    app.ConfEvalEvaluationStatusLabel.Text = 'Standby';
    drawnow;
end
```

7.39: Function ConfEvalLoadMeasButton_ButtonPushedFcn

```
function ConfEvalLoadMeasButton_ButtonPushedFcn(app, src, event)
%ConfEvalLoadMeasButton_ButtonPushedFcn listening to the Load Measurement
%button and loading a already performed Measurement with or without already
%existing results.
[baseFileName, folder] = uigetfile('*.mat');
fullFileName = fullfile(folder, baseFileName);
if exist(fullFileName, 'file')
    % Normal situation - they picked an existing file.
    temp = struct2cell(load(fullFileName));
    app.loadedMeas = temp{1};
    % Now do something with storedStructure, like extract fields into new variables
        or whatever you want.
else
    % Error: Would only get here if they typed in a name of a non-existant file
    % instead of picking one from the folder.
    warningMessage = sprintf('Warning: mat file does not exist:\n%s', fullFileName);
```

```
    uiwait(errordlg(warningMessage));
    return;
end
UpdateGUIConfEvalTab(app);

if not(isempty(app.loadedMeas.getResults()))
    UpdateGUIEvalResTab(app);
else
    ClearGUIEvalResTab(app);
end


end
```

7.40: Function ConfEvalSmpDropDown_ValueChangedFcn

```
function ConfEvalSmpDropDown_ValueChangedFcn(app, src, event)
%ConfEvalSmpDropDown_ValueChangedFcn listengin to changes of the selected
%Measuring Point and adjusting the shown data in the UI.
app.ConfEvalEvaluationButton.Enable = 'on';
newValueDataSet = event.Value.getDataSet();
[Useless NoI] = size(newValueDataSet);
cla(app.ConfEvalPreviewImUIAxes,'reset');
app.ConfEvalSmpImDropDown.Items = {};
app.ConfEvalSmpImDropDown.ItemsData = [];
for j=1:NoI
    app.ConfEvalSmpImDropDown.Items = [app.ConfEvalSmpImDropDown.Items sprintf('%i',
        j)];
    app.ConfEvalSmpImDropDown.ItemsData = [app.ConfEvalSmpImDropDown.ItemsData
        newValueDataSet(j)];
end
% Display image and stretch to fill axes
imshow(newValueDataSet(1).getImageData(),...
    'Parent', app.ConfEvalPreviewImUIAxes);
app.ConfEvalPreviewImUIAxes.Toolbar.Visible = 'off';
colormap(app.ConfEvalPreviewImUIAxes, jet(256));
app.ConfEvalPreviewImUIAxes.Interactions = [];
if not(isempty(app.ConfEvalSelectedArea))
    app.ConfEvalRectRoi = drawrectangle(app.ConfEvalPreviewImUIAxes,'LabelVisible','
        hover','Label','OuterRectangle','Position',app.ConfEvalSelectedArea);
    addlistener(app.ConfEvalRectRoi,'MovingROI',@(src, evt) drawRectAllevents(app,
        src,evt));
    addlistener(app.ConfEvalRectRoi,'ROIMoved',@(src, evt) drawRectAllevents(app,src
        ,evt));

    centerRectangleX = app.ConfEvalRectRoi.Position(1)+app.ConfEvalRectRoi.Position
        (3)/2;
    centerRectangleY = app.ConfEvalRectRoi.Position(2)+app.ConfEvalRectRoi.Position
        (4)/2;
    app.ConfEvalPointRoi = drawpoint(app.ConfEvalPreviewImUIAxes,...
```

```
        'Position',[centerRectangleX centerRectangleY],'Color','r',...
        'InteractionsAllowed','none');
end
drawnow;
drawnow;
end
```

7.41: Function ConfEvalSmpImDropDown_ValueChangedFcn

```
function ConfEvalSmpImDropDown_ValueChangedFcn(app, src, event)
%ConfEvalSmpImDropDown_ValueChangedFcn listening to changes to the selected
%image and adjusting shown elements in the UI.
cla(app.ConfEvalPreviewImUIAxes,'reset');
imshow(event.Value.getImageData(), 'Parent', app.ConfEvalPreviewImUIAxes);
app.ConfEvalPreviewImUIAxes.Toolbar.Visible = 'off';
app.ConfEvalPreviewImUIAxes.Interactions = [];
colormap(app.ConfEvalPreviewImUIAxes, jet(256));
if not(isempty(app.ConfEvalSelectedArea))
    app.ConfEvalRectRoi = drawrectangle(app.ConfEvalPreviewImUIAxes,'LabelVisible','
        hover','Label','OuterRectangle','Position',app.ConfEvalSelectedArea);
    addlistener(app.ConfEvalRectRoi,'MovingROI',@(src, evt)
        drawConfEvalRectAllevents(app,src,evt));
    addlistener(app.ConfEvalRectRoi,'ROIMoved',@(src, evt) drawConfEvalRectAllevents
        (app,src,evt));

    centerRectangleX = app.ConfEvalRectRoi.Position(1)+app.ConfEvalRectRoi.Position
        (3)/2;
    centerRectangleY = app.ConfEvalRectRoi.Position(2)+app.ConfEvalRectRoi.Position
        (4)/2;
    app.ConfEvalPointRoi = drawpoint(app.ConfEvalPreviewImUIAxes,...
        'Position',[centerRectangleX centerRectangleY],'Color','r',...
        'InteractionsAllowed','none');
end
drawnow;
drawnow;
%     cla(app.LiveMeasResultImUIAxes,'reset');
%     imshow(event.Value.getImageData(), 'Parent', app.LiveMeasResultImUIAxes);
%     el = drawellipse(app.LiveMeasResultImUIAxes,'Center',...
%         [1282/2 1026/2],...
%         'SemiAxes',[100 100],...
%         'StripeColor','r','Linewidth',1.75);
%     el.InteractionsAllowed = 'none';
end
```

7.42: Function drawConfEvalRectAllevents

```
function drawConfEvalRectAllevents(app,src,evt)
%drawRectAllevents listening to the Rectangle object and resolving all
%events happening to it.
evname = evt.EventName;
```

```
switch(evname)
    case{'MovingROI'}
        disp(['ROI moving previous position: ' mat2str(evt.PreviousPosition)]);
        disp(['ROI moving current position: ' mat2str(evt.CurrentPosition)]);
    case{'ROIMoved'}
        disp(['ROI moved previous position: ' mat2str(evt.PreviousPosition)]);
        disp(['ROI moved current position: ' mat2str(evt.CurrentPosition)]);
end

app.ConfEvalSelectedArea = evt.CurrentPosition;
centerRectangleX = app.ConfEvalRectRoi.Position(1)+app.ConfEvalRectRoi.Position(3)
    /2;
centerRectangleY = app.ConfEvalRectRoi.Position(2)+app.ConfEvalRectRoi.Position(4)
    /2;
app.ConfEvalPointRoi.Position(1) = centerRectangleX;
app.ConfEvalPointRoi.Position(2) = centerRectangleY;

end
```

7.43: Function drawLiveMeasRectAllevents

```
function drawLiveMeasRectAllevents(app,src,evt)
%drawRectAllevents listening to the Rectangle object and resolving all
%events happening to it.
evname = evt.EventName;
switch(evname)
    case{'MovingROI'}
        disp(['ROI moving previous position: ' mat2str(evt.PreviousPosition)]);
        disp(['ROI moving current position: ' mat2str(evt.CurrentPosition)]);
    case{'ROIMoved'}
        disp(['ROI moved previous position: ' mat2str(evt.PreviousPosition)]);
        disp(['ROI moved current position: ' mat2str(evt.CurrentPosition)]);
end

app.LiveMeasSelectedArea = evt.CurrentPosition;
centerRectangleX = app.LiveMeasRectRoi.Position(1)+app.LiveMeasRectRoi.Position(3)
    /2;
centerRectangleY = app.LiveMeasRectRoi.Position(2)+app.LiveMeasRectRoi.Position(4)
    /2;
app.LiveMeasPointRoi.Position(1) = centerRectangleX;
app.LiveMeasPointRoi.Position(2) = centerRectangleY;

end
```

7.44: Function EvalResImChangeColorMap

```
function EvalResImChangeColorMap(app, src, event)
%changes the colormap limits of the EvalResImDataUIAxes plot
app.EvalResColorMap(1) = app.EvalResColorMinEditField.Value
app.EvalResColorMap(2) = app.EvalResColorMaxEditField.Value
```

```
caxis(app.EvalResImDataUIAxes,app.EvalResColorMap);
end
```

7.45: Function EvalResImDataChangeColorMapButton_ValueChangedFcn

```
function EvalResImDataChangeColorMapButton_ValueChangedFcn(app, src, event)
%HSCamConfCamPreviewButton_ValueChangedFcn ValueChangedFcn listening to the
%state button to enable and disable the preview of the camera.

if src.Value
    colormap(app.EvalResImDataUIAxes, gray(256));
    caxis(app.EvalResImDataUIAxes,app.EvalResColorMap);
    return;
end
colormap(app.EvalResImDataUIAxes, jet(256));
caxis(app.EvalResImDataUIAxes,app.EvalResColorMap);
end
```

7.46: Function EvalResImDataDropDown_ValueChangedFcn

```
function EvalResImDataDropDown_ValueChangedFcn(app, src, event)
%EvalResImDataDropDown_ValueChangedFcn listening to changes of the selected
%image and ajusting data shown in ui to the new selection
SetEvalResImDataParams(app,event.Value);
end
```

7.47: Function EvalResMeasDataResultDropDown_ValueChangedFcn

```
function EvalResMeasDataResultDropDown_ValueChangedFcn(app, src, event)
%EvalResMeasDataResultDropDown_ValueChangedFcn listening to the result
%dropdown menu adjusting shown data in the ui for the new selection.
SetEvalResMeasDataParams(app,event.Value);
end
```

7.48: Function EvalResMeasPlotButton_ButtonPushedFcn

```
function EvalResMeasPlotButton_ButtonPushedFcn(app, src, event)
%EvalResMeasPlotButton_ButtonPushedFcn listening to Plot Selection button
%to plot the chosen Plotoption.
fontsizeLabels = 38;
fontsizeLegend = 20;
fontsizeArrowLabel = 26;
fontsizeTickLabels = 22;

offsetHleft = -5;
offsetHright = -5;
offsetArrowLeft = -5;
offsetArrowHeight = -22;

lineWidth = 2.5;

```

```
plotSelection = app.EvalResPlotChooserDropDown.Value;

Mp = app.EvalResSmpDataDropDown.Value;
Mp.calcMeanImage();
im = Mp.getMeanImage().getImageData();
offset = 125;

if app.EvalResHoldPlotButton.Value
    hold on;
    ax=gca;
else
    fig=figure;
    ax=axes;
end

switch plotSelection
    case 'Plot Radius X of all MP'
        f = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
            getfxFit();
        dW = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
            getdWx();
        NoMP = size(app.EvalResMeasDataResultDropDown.Value.
            getEvaluatedMeasurementData().getMeasurementDataSet(),2);
        temp = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
            .getMeasurementDataSet();
        NoI = size(temp(1).getDataSet(),2);
        dWStd = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData
            ().getdWxStd();
        zPos = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
            .getzPos();
        corrMethod = app.EvalResMeasDataResultDropDown.Value.getBackgroundCorrMethod
            ();
        lambda = app.EvalResMeasDataResultDropDown.Value.getLambda();
        integAreaFact = app.EvalResMeasDataResultDropDown.Value.getIntegAreaFactor;
        legendText = sprintf('Corr Method: %s, \x03bb = %i nm, IAF = %.1f, NoMP = %i
            , NoI = %i',corrMethod,lambda,integAreaFact,NoMP,NoI);

        %         zPos = zPos(1:end-2);
        %         dW = dW(1:end-2);
        %         dWStd = dWStd(1:end-2);

        dW_min = dW-dWStd;
        dW_max = dW+dWStd;
        hold on;

        stdLowerLimit = line(zPos/1e+3,dW_min/2,'Color','black','LineWidth',
            lineWidth);
        stdLowerLimit.Annotation.LegendInformation.IconDisplayStyle = 'off';
        stdUpperLimit = line(zPos/1e+3,dW_max/2,'Color','black','LineWidth',
```

```
            lineWidth);
56          stdUpperLimit.Annotation.LegendInformation.IconDisplayStyle = 'off';
57
58          zPos2 = [zPos/1e+3, fliplr(zPos/1e+3)];
59          inBetween = [dW_min/2, fliplr(dW_max/2)];
60          stdArea = fill(zPos2, inBetween, [0.85 0.85 0.85],'DisplayName','Standard
                deviation');
61          %         stdArea.Annotation.LegendInformation.IconDisplayStyle = 'off';
62
63          plot(zPos/1e+3,dW/2,'b','LineWidth', lineWidth,'DisplayName',legendText);
64          set(gca,'FontSize',fontsizeTickLabels);
65          legend('FontSize',fontsizeLegend);
66
67          title('$\textrm{Beamradius } R_{x}$','Interpreter','latex','FontSize',
                fontsizeLabels)
68          ylabel('$\textrm{Beamradius } R_{x} \textrm{ in } \mu m$ ','Interpreter','
                latex','FontSize',fontsizeLabels);
69          xlabel('$\textrm{z - Coordinate in } mm$ ','Interpreter','latex','FontSize',
                fontsizeLabels);
70      case 'Plot Radius Y of all MP'
71          f = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getfyFit();
72          dW = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getdWy();
73          NoMP = size(app.EvalResMeasDataResultDropDown.Value.
                getEvaluatedMeasurementData().getMeasurementDataSet(),2);
74          temp = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getMeasurementDataSet();
75          NoI = size(temp(1).getDataSet(),2);
76          dWStd = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData
                ().getdWyStd();
77          zPos = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getzPos();
78
79          corrMethod = app.EvalResMeasDataResultDropDown.Value.getBackgroundCorrMethod
                ();
80          lambda = app.EvalResMeasDataResultDropDown.Value.getLambda();
81          integAreaFact = app.EvalResMeasDataResultDropDown.Value.getIntegAreaFactor;
82          legendText = sprintf('Corr Method: %s, \x03bb = %i nm, IAF = %.1f, NoMP = %i
                , NoI = %i',corrMethod,lambda,integAreaFact,NoMP,NoI);
83
84          dW_min = dW-dWStd;
85          dW_max = dW+dWStd;
86          hold on;
87
88          stdLowerLimit = line(zPos/1e+3,dW_min/2,'Color','black','LineWidth',
                lineWidth);
89          stdLowerLimit.Annotation.LegendInformation.IconDisplayStyle = 'off';
90          stdUpperLimit = line(zPos/1e+3,dW_max/2,'Color','black','LineWidth',
```

```
            lineWidth);
        stdUpperLimit.Annotation.LegendInformation.IconDisplayStyle = 'off';

        zPos2 = [zPos/1e+3, fliplr(zPos/1e+3)];
        inBetween = [dW_min/2, fliplr(dW_max/2)];
        stdArea = fill(zPos2, inBetween, [0.85 0.85 0.85],'DisplayName','Standard
            deviation');
        %         stdArea.Annotation.LegendInformation.IconDisplayStyle = 'off';

        plot(zPos/1e+3,dW/2,'b','LineWidth', lineWidth,'DisplayName',legendText);
        set(gca,'FontSize',fontsizeTickLabels);
        legend('FontSize',fontsizeLegend);

        title('$\textrm{Beamradius } R_{y}$','Interpreter','latex','FontSize',
            fontsizeLabels)
        ylabel('$\textrm{Beamradius } R_{y} \textrm{ in } \mu m$ ','Interpreter','
            latex','FontSize',fontsizeLabels);
        xlabel('$\textrm{z - Coordinate in } mm$ ','Interpreter','latex','FontSize',
            fontsizeLabels);

    case 'Plot 2D Intensity Distribution X of selected MP'
        %Get Data from current MP
        xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        zPos = Mp.getZPos()/1e+3;
        dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        x=xCenterInPixel-round(dWxInPixel/2):1:xCenterInPixel+round(dWxInPixel/2);
        y=ones(size(x))*yCenterInPixel;
        v=[x;y];
        %Rotate line into Mainaxis using the azimuth angle
        x_center = xCenterInPixel;
        y_center = yCenterInPixel;
        center = repmat([x_center; y_center], 1, length(x));
        theta = Mp.getAzimutPhi_mean();
        R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
        s = v - center;     % shift points in the plane so that the center of
            rotation is at the origin
        so = R*s;           % apply the rotation about the origin
        vo = so + center;   % shift again so the origin goes back to the desired
            center of rotation
        % this can be done in one line as:
        % vo = R*(v - center) + center
        % pick out the vectors of rotated x- and y-data
        x_rotated = vo(1,:);
        y_rotated = vo(2,:);
        % plot(x, y, 'k-', x_rotated, y_rotated, 'r-', x_center, y_center, 'bo');
```

```
131         rounded_rotated_x=floor(x_rotated);
132         rounded_rotated_y=floor(y_rotated);
133
134         %creating data vektor on rotated line
135         z=[];
136         for i=1:size(rounded_rotated_x,2)
137             z = [z,im(rounded_rotated_y(i),rounded_rotated_x(i))];
138         end
139
140
141
142         rounded_rotated_x = (rounded_rotated_x-xCenterInPixel)*app.
                ConfEvalPixelsizeEditField.Value;
143
144         %fitting gaus to data
145         f = fit(rounded_rotated_x.',z.','gauss1');
146         % plotting gaus plus data
147         plot(ax,rounded_rotated_x,z,'b','LineWidth',lineWidth);
148         set(gca,'FontSize',fontsizeTickLabels);
149         hold on;
150         h=plot(f,'r--');
151         set(h, 'LineWidth',lineWidth);
152         legend('Measurement Data','Gauss1 fit','FontSize',fontsizeLegend);
153
154         % hold on because we need to draw quite some things
155         hold on;
156         %Draw 1/e^2 Diameter lines and centerline
157         hLeft = xline(-(dWxInPixel-offset)*app.ConfEvalPixelsizeEditField.Value/2-
                offsetHleft, 'k--', 'LineWidth', lineWidth);
158         hLeft.Annotation.LegendInformation.IconDisplayStyle = 'off';
159         hRight = xline((dWxInPixel-offset)*app.ConfEvalPixelsizeEditField.Value/2-
                offsetHright, 'k--', 'LineWidth', lineWidth);
160         hRight.Annotation.LegendInformation.IconDisplayStyle = 'off';
161         %         hCenter = xline(yCenterInPixel-yCenterInPixel, 'g--', 'LineWidth',
                 1);
162         %         hCenter.Annotation.LegendInformation.IconDisplayStyle = 'off';
163
164         xLeft = ones(1,size(z,2))*(-(dWxInPixel-offset)*app.
                ConfEvalPixelsizeEditField.Value/2);
165         yLeft = 0:max(z)/(size(z,2)-1):max(z);
166         [xiLeft,yiLeft] = polyxpoly(rounded_rotated_x,z,xLeft,yLeft);
167         xRight = ones(1,size(z,2))*((dWxInPixel-offset)*app.
                ConfEvalPixelsizeEditField.Value/2);
168         yRight = 0:max(z)/(size(z,2)-1):max(z);
169         [xiRight,yiRight] = polyxpoly(rounded_rotated_x,z,xRight,yRight);
170
171         % Draw Doublearrow plus textbox for 1/e^2 Diameter
172         xArrow = [xiLeft-offsetArrowLeft xiRight-offsetArrowLeft];
173         yArrow = [yiLeft-offsetArrowHeight yiLeft-offsetArrowHeight];
```

```
annoArrow=annotation('doublearrow');
annoArrow.Parent = ax;
annoArrow.X = xArrow;
annoArrow.Y = yArrow;
annoArrow.LineWidth = 1;

annoText=annotation('textbox');
annoText.Parent = ax;
annoText.Position = [(xArrow(1))/2 yArrow(1)+20 1 1];
annoText.FitBoxToText = 'on';
annoText.LineWidth = 1;
annoText.String = '$D4\sigma$ Diameter';
annoText.FontSize = fontsizeArrowLabel;
annoText.Interpreter = 'latex';
annoText.Position(1) = annoText.Position(1)-annoText.Position(3)/2;
annoText.LineStyle = 'none';

% Calc FWHM Diameter location
xFHWM = min(rounded_rotated_x):0.001:max(rounded_rotated_x);
gausFitFunction = feval(f,xFHWM);
indexOfFWHM=find(abs(gausFitFunction-max(z)/2) < 0.01);
indexOfFWHMLeft=indexOfFWHM(1);

indexOfFWHMRight=indexOfFWHM(end);

%Draw FWHM Diameter lines
hLeftFWHM = xline(xFHWM(indexOfFWHMLeft), 'm--', 'LineWidth', lineWidth);
hLeftFWHM.Annotation.LegendInformation.IconDisplayStyle = 'off';
hRightFHWM = xline((xFHWM(indexOfFWHMRight)), 'm--', 'LineWidth', lineWidth)
    ;
hRightFHWM.Annotation.LegendInformation.IconDisplayStyle = 'off';


xiRight = xFHWM(indexOfFWHMRight);
xiLeft = xFHWM(indexOfFWHMLeft);

% Draw Doublearrow plus textbox for FWHM Diameter
yiLeft = gausFitFunction(indexOfFWHMLeft);
xArrow = [xiLeft xiRight];
yArrow = [yiLeft yiLeft];
annoArrow=annotation('doublearrow');
annoArrow.Parent = ax;
annoArrow.X = xArrow;
annoArrow.Y = yArrow;
annoArrow.LineWidth = 1;

annoText=annotation('textbox');
annoText.Parent = ax;
annoText.Position = [(xArrow(1))/2 yArrow(1)+20 1 1];
```

```
        annoText.FitBoxToText = 'on';
        annoText.LineWidth = 1;
        annoText.String = 'FWHM Diameter';
        annoText.FontSize = fontsizeArrowLabel;
        annoText.Interpreter = 'latex';
        annoText.Position(1) = annoText.Position(1)-annoText.Position(3)/2;
        annoText.LineStyle = 'none';
        titleText = sprintf('2D Intensity Distribution X (z = %.3f mm)',zPos);
        title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
        ylabel('Intensity 0 - 255','Interpreter','latex','FontSize',fontsizeLabels);
        xlabel('$\textrm{Beamradius } R_{x} \textrm{ in } \mu m$ ','Interpreter','
            latex','FontSize',fontsizeLabels);

        hold off;
    case 'Plot 2D Intensity Distribution Y of selected MP'
        %Get Data from current MP
        xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        zPos = Mp.getZPos()/1e+3;
        dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        x=xCenterInPixel-round(dWxInPixel/2):1:xCenterInPixel+round(dWxInPixel/2);
        y=ones(size(x))*yCenterInPixel;
        v=[x;y];
        %Rotate line into Mainaxis using the azimuth angle
        x_center = xCenterInPixel;
        y_center = yCenterInPixel;
        center = repmat([x_center; y_center], 1, length(x));
        theta = Mp.getAzimutPhi_mean()+90;
        R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
        s = v - center;     % shift points in the plane so that the center of
            rotation is at the origin
        so = R*s;           % apply the rotation about the origin
        vo = so + center;   % shift again so the origin goes back to the desired
            center of rotation
        % this can be done in one line as:
        % vo = R*(v - center) + center
        % pick out the vectors of rotated x- and y-data
        x_rotated = vo(1,:);
        y_rotated = vo(2,:);
        % plot(x, y, 'k-', x_rotated, y_rotated, 'r-', x_center, y_center, 'bo');
        rounded_rotated_x=floor(x_rotated);
        rounded_rotated_y=floor(y_rotated);

        %creating data vektor on rotated line
        z=[];
        for i=1:size(rounded_rotated_x,2)
```

```
266             z = [z,im(rounded_rotated_y(i),rounded_rotated_x(i))];
267         end
268
269
270         rounded_rotated_y = (rounded_rotated_y-yCenterInPixel)*app.
                ConfEvalPixelsizeEditField.Value;
271         %fitting gaus to data
272         f = fit(rounded_rotated_y.',z.','gauss1');
273         % plotting gaus plus data
274         plot(ax,rounded_rotated_y,z,'b','LineWidth',lineWidth);
275         set(gca,'FontSize',fontsizeTickLabels);
276         hold on;
277         h=plot(f,'r--');
278         set(h, 'LineWidth',lineWidth);
279         legend('Measurement Data','Gauss1 fit','FontSize',fontsizeLegend);
280
281         % hold on because we need to draw quite some things
282         hold on;
283         %Draw 1/e^2 Diameter lines and centerline
284         hLeft = xline(-(dWyInPixel-offset)*app.ConfEvalPixelsizeEditField.Value/2-
                offsetHleft, 'k--', 'LineWidth', lineWidth);
285         hLeft.Annotation.LegendInformation.IconDisplayStyle = 'off';
286         hRight = xline((dWyInPixel-offset)*app.ConfEvalPixelsizeEditField.Value/2-
                offsetHright, 'k--', 'LineWidth', lineWidth);
287         hRight.Annotation.LegendInformation.IconDisplayStyle = 'off';
288         %         hCenter = xline(yCenterInPixel-yCenterInPixel, 'g--', 'LineWidth',
                 1);
289         %         hCenter.Annotation.LegendInformation.IconDisplayStyle = 'off';
290
291         xLeft = ones(1,size(z,2))*(-(dWyInPixel-offset)*app.
                ConfEvalPixelsizeEditField.Value/2);
292         yLeft = 0:max(z)/(size(z,2)-1):max(z);
293         [xiLeft,yiLeft] = polyxpoly(rounded_rotated_y,z,xLeft,yLeft);
294         xRight = ones(1,size(z,2))*((dWyInPixel-offset)*app.
                ConfEvalPixelsizeEditField.Value/2);
295         yRight = 0:max(z)/(size(z,2)-1):max(z);
296         [xiRight,yiRight] = polyxpoly(rounded_rotated_y,z,xRight,yRight);
297
298         % Draw Doublearrow plus textbox for 1/e^2 Diameter
299         xArrow = [xiLeft-offsetArrowLeft xiRight-offsetArrowLeft];
300         yArrow = [yiLeft-offsetArrowHeight yiLeft-offsetArrowHeight];
301         annoArrow=annotation('doublearrow');
302         annoArrow.Parent = ax;
303         annoArrow.X = xArrow;
304         annoArrow.Y = yArrow;
305         annoArrow.LineWidth = 1;
306
307         annoText=annotation('textbox');
308         annoText.Parent = ax;
```

```
annoText.Position = [(xArrow(1))/2 yArrow(1)+20 1 1];
annoText.FitBoxToText = 'on';
annoText.LineWidth = 1;
annoText.String = '$D4\sigma$ Diameter';
annoText.FontSize = fontsizeArrowLabel;
annoText.Interpreter = 'latex';
annoText.Position(1) = annoText.Position(1)-annoText.Position(3)/2;
annoText.LineStyle = 'none';

% Calc FWHM Diameter location
xFHWM = min(rounded_rotated_y):0.001:max(rounded_rotated_y);
gausFitFunction = feval(f,xFHWM);
indexOfFWHM=find(abs(gausFitFunction-max(z)/2) < 0.01);
indexOfFWHMLeft=indexOfFWHM(1);

indexOfFWHMRight=indexOfFWHM(end);

%Draw FWHM Diameter lines
hLeftFWHM = xline(xFHWM(indexOfFWHMLeft), 'm--', 'LineWidth', lineWidth);
hLeftFWHM.Annotation.LegendInformation.IconDisplayStyle = 'off';
hRightFHWM = xline((xFHWM(indexOfFWHMRight)), 'm--', 'LineWidth', lineWidth)
    ;
hRightFHWM.Annotation.LegendInformation.IconDisplayStyle = 'off';


xiRight = xFHWM(indexOfFWHMRight);
xiLeft = xFHWM(indexOfFWHMLeft);

% Draw Doublearrow plus textbox for FWHM Diameter
yiLeft = gausFitFunction(indexOfFWHMLeft);
xArrow = [xiLeft xiRight];
yArrow = [yiLeft yiLeft];
annoArrow=annotation('doublearrow');
annoArrow.Parent = ax;
annoArrow.X = xArrow;
annoArrow.Y = yArrow;
annoArrow.LineWidth = 1;

annoText=annotation('textbox');
annoText.Parent = ax;
annoText.Position = [(xArrow(1))/2 yArrow(1)+20 1 1];
annoText.FitBoxToText = 'on';
annoText.LineWidth = 1;
annoText.String = 'FWHM Diameter';
annoText.FontSize = fontsizeArrowLabel;
annoText.Interpreter = 'latex';
annoText.Position(1) = annoText.Position(1)-annoText.Position(3)/2;
annoText.LineStyle = 'none';
titleText = sprintf('2D Intensity Distribution Y (z = %.3f mm)',zPos);
```

```
        title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
        ylabel('Intensity 0 - 255','Interpreter','latex','FontSize',fontsizeLabels);
        xlabel('$\textrm{Beamradius } R_{y} \textrm{ in } \mu m$ ','Interpreter','
            latex','FontSize',fontsizeLabels);

        hold off;
    case 'Plot 3D Intensity Distribution of selected MP'
        xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        zPos = Mp.getZPos()/1e+3;
        dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+10;
        dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+10;
        titleText = sprintf('3D Intensity Distribution (z = %.3f mm)',zPos);

        if dWxInPixel > dWyInPixel
            dWsym = dWxInPixel;
        else
            dWsym = dWyInPixel;
        end
        offset=200;
        im = im+offset;
        centerY = (round(yCenterInPixel+dWsym)-round(yCenterInPixel-dWsym))/2;
        centerX = (round(xCenterInPixel+dWsym)-round(xCenterInPixel-dWsym))/2;
        s = surf(ax,im(round(yCenterInPixel-dWsym):round(yCenterInPixel+dWsym),...
            round(xCenterInPixel-dWsym):round(xCenterInPixel+dWsym)));
        s.XData = (s.XData-centerX)*app.ConfEvalPixelsizeEditField.Value;
        s.YData = (s.YData-centerY)*app.ConfEvalPixelsizeEditField.Value;
        set(gca,'xlim',[s.XData(1) s.XData(end)],'ylim',[s.YData(1) s.YData(end)]);
        s.EdgeColor = 'none';
        set(gca,'FontSize',fontsizeTickLabels);
        X = s.XData';
        zFFT = s.YData;
        ZZ = s.ZData;
        [XX YY] = meshgrid(X,zFFT);
        hold on
        ims = imagesc(im(round(yCenterInPixel-dWsym):round(yCenterInPixel+dWsym),...
            round(xCenterInPixel-dWsym):round(xCenterInPixel+dWsym)));
        ims.XData = (ims.XData-centerX)*app.ConfEvalPixelsizeEditField.Value;
        ims.YData = (ims.YData-centerY)*app.ConfEvalPixelsizeEditField.Value;
        spacing = 10;  % play around so it fits the size of your data set
        for i = 1 : spacing : length(XX(:,1))
            plot3(XX(:,i), YY(:,i), ZZ(:,i),'-k');
            plot3(XX(i,:), YY(i,:), ZZ(i,:),'-k');
        end
        set(gca,'ZTickLabel',[]);
        title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
        %         zlabel('$\textrm{Intensity}$ ','Interpreter','latex','FontSize
```

```
        ',16);
403     xlabel('$\textrm{x - Coordinate in } \mu m$','Interpreter','latex','FontSize
        ',25);
404     ylabel('$\textrm{y - Coordinate in } \mu m$','Interpreter','latex','FontSize
        ',25);
405
406     xh = get(gca,'XLabel'); % Handle of the x label
407     set(xh, 'Units', 'Normalized')
408     pos = get(xh, 'Position');
409     set(xh, 'Position',pos.*[1.2,-0.3,1],'Rotation',21)
410     yh = get(gca,'YLabel'); % Handle of the y label
411     set(yh, 'Units', 'Normalized')
412     pos = get(yh, 'Position');
413     set(yh, 'Position',pos.*[0.6,-0.2,1],'Rotation',-33)
414     axis image;
415
416     a = colorbar('FontSize',20);
417     a.Ticks = linspace(200, 455, 6);
418     a.TickLabels = linspace(0, 255, 6);
419     ylabel(a,'Intensity 0-255 (false color)','Interpreter','latex','FontSize',
        fontsizeLabels);
420     colormap(jet(256));
421     caxis([0+offset 255+offset]);
422
423
424 case 'Plot Spatial Spectrum of selected MP'
425     dx=app.ConfEvalPixelsizeEditField.Value;
426     dy=app.ConfEvalPixelsizeEditField.Value;
427     zPos = Mp.getZPos()/1e+3;
428     Fs_x = 1/dx;     % centimeters per pixel
429     Fs_y = 1/dy;
430
431     [ M, N, ~ ] = size(im);      % pixels
432     x = dx*(0:N-1)';               % centimeters
433     y = dy*(0:M-1)';
434     dFx = Fs_x/N;              % cycles per centimeter
435     dFy = Fs_y/M;
436     Fx = (-Fs_x/2:dFx:Fs_x/2-dFx)';     % cycles per centimeter
437     Fy = (-Fs_y/2:dFy:Fs_y/2-dFy)';
438
439     zFFT = fft2(im);
440     absY=abs(fftshift(zFFT));
441     imagesc(Fx,Fy,absY);
442     set(gca,'FontSize',fontsizeTickLabels);
443     colormap(gray);
444     c = colorbar('FontSize',fontsizeTickLabels);
445     ylabel(c,'$\textrm{Intensity in Greyscale}$ ','Interpreter','latex','
        FontSize',fontsizeLabels);
446
```

```
447         title('Spatial Spectrum Linear Amplitude','Interpreter','latex','FontSize',
                fontsizeLabels)
448         xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
449         ylabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
450
451         figure;
452         absYlog=10*log10(absY./max(max(absY)));
453         imagesc(Fx,Fy,absYlog);
454         set(gca,'FontSize',fontsizeTickLabels);
455         colormap(gray);
456         c = colorbar('FontSize',fontsizeTickLabels);
457         ylabel(c,'$\textrm{Intensity in }dB$ ','Interpreter','latex','FontSize',
                fontsizeLabels);
458         title('Spatial Spectrum Logarithmic Amplitude','Interpreter','latex','
                FontSize',fontsizeLabels)
459         xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
460         ylabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
461
462
463         k=0.0001;
464         scaledP1 = absY.*k;
465         c=255/log10(1+max(max(scaledP1)));
466         absYlog2 = c*log10(1+scaledP1);
467
468         figure;
469         imagesc(Fx,Fy,absYlog2);
470         set(gca,'FontSize',fontsizeTickLabels);
471         colormap(gray);
472         c = colorbar('FontSize',fontsizeTickLabels);
473         ylabel(c,'$\textrm{Intensity in }dB$ ','Interpreter','latex','FontSize',
                fontsizeLabels);
474         titleText = sprintf('Spatial Spectrum (z = %.3f mm)',zPos);
475         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
476         xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
477         ylabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
478         grid on;
479     case 'Plot 2D Spectral Distribution X of selected MP'
480         %Get Data from current MP
481         xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value
                ();
482         yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value
                ();
483         zPos = Mp.getZPos()/1e+3;
```

```
484         dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
485         dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
486         x=xCenterInPixel-round(dWxInPixel/2):1:xCenterInPixel+round(dWxInPixel/2);
487         y=ones(size(x))*yCenterInPixel;
488         v=[x;y];
489         %Rotate line into Mainaxis using the azimuth angle
490         x_center = xCenterInPixel;
491         y_center = yCenterInPixel;
492         center = repmat([x_center; y_center], 1, length(x));
493         theta = Mp.getAzimutPhi_mean();
494         R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
495         s = v - center;     % shift points in the plane so that the center of
                rotation is at the origin
496         so = R*s;           % apply the rotation about the origin
497         vo = so + center;   % shift again so the origin goes back to the desired
                center of rotation
498         % this can be done in one line as:
499         % vo = R*(v - center) + center
500         % pick out the vectors of rotated x- and y-data
501         x_rotated = vo(1,:);
502         y_rotated = vo(2,:);
503         % plot(x, y, 'k-', x_rotated, y_rotated, 'r-', x_center, y_center, 'bo');
504         rounded_rotated_x=floor(x_rotated);
505         rounded_rotated_y=floor(y_rotated);
506
507         %creating data vektor on rotated line
508         z=[];
509         for i=1:size(rounded_rotated_x,2)
510             z = [z,im(rounded_rotated_y(i),rounded_rotated_x(i))];
511         end
512         dx=app.ConfEvalPixelsizeEditField.Value;
513         dy=app.ConfEvalPixelsizeEditField.Value;
514
515         Fs_x = 1/dx;     % centimeters per pixel
516         Fs_y = 1/dy;
517
518         [M, N] = size(z);      % pixels
519         x = dx*(0:N-1)';               % centimeters
520
521         dFx = Fs_x/N;              % cycles per centimeter
522
523         Fx = (-Fs_x/2:dFx:Fs_x/2-dFx)';     % cycles per centimeter
524
525         zFFT = fft(z);
526         P2 = abs(zFFT/N);
527         P1 = P2(1:floor(N/2)+1);
528         P1(2:end-1) = 2*P1(2:end-1);
529         f = Fs_x*(0:(N/2))/N;
530 %         plot(gca,f,P1,'LineWidth',lineWidth);
```

```
%         set(gca,'FontSize',fontsizeTickLabels);
%         titleText = sprintf('2D Spectral Distribution X (z = %.3f mm)',zPos);
%         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
%         ylabel('$\textrm{Intensity}\textrm{ in Greyscale}$ ','Interpreter','latex','FontSize',fontsizeLabels);
%         xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','Interpreter','latex','FontSize',fontsizeLabels);
%         grid on;
%
%         figure;
%         absP1log=10*log10(P1./max(P1));
%         plot(gca,f,absP1log,'LineWidth',lineWidth);
%         set(gca,'FontSize',fontsizeTickLabels);
%         titleText = sprintf('2D Spectral Distribution X (z = %.3f mm)',zPos);
%         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
%         ylabel('$\textrm{Intensity}\textrm{ in }dB$ ','Interpreter','latex','FontSize',fontsizeLabels);
%         xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','Interpreter','latex','FontSize',fontsizeLabels);
%         grid on;
%
%         figure;
        k=0.0001;
        scaledP1 = P1.*k;
        c=255/log10(1+max(max(scaledP1)));
        absP1log = c*log10(1+scaledP1);

        plot(ax,f,absP1log,'LineWidth',lineWidth);
        set(gca,'FontSize',fontsizeTickLabels);
        titleText = sprintf('2D Spectral Distribution X (z = %.3f mm)',zPos);
        title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
        ylabel('$\textrm{Intensity}\textrm{ in }dB$ ','Interpreter','latex','FontSize',fontsizeLabels);
        xlabel('$\textrm{Spatial frequency } f_{x} \textrm{ in } 1/\mu m$ ','Interpreter','latex','FontSize',fontsizeLabels);
        grid on;

        hold off;
    case 'Plot 2D Spectral Distribution Y of selected MP'
        %Get Data from current MP
        xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value();
        yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value();
        zPos = Mp.getZPos()/1e+3;
        dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+offset;
        x=xCenterInPixel-round(dWxInPixel/2):1:xCenterInPixel+round(dWxInPixel/2);
        y=ones(size(x))*yCenterInPixel;
```

```
572         v=[x;y];
573         %Rotate line into Mainaxis using the azimuth angle
574         x_center = xCenterInPixel;
575         y_center = yCenterInPixel;
576         center = repmat([x_center; y_center], 1, length(x));
577         theta = Mp.getAzimutPhi_mean()+90;
578         R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
579         s = v - center;     % shift points in the plane so that the center of
                rotation is at the origin
580         so = R*s;           % apply the rotation about the origin
581         vo = so + center;   % shift again so the origin goes back to the desired
                center of rotation
582         % this can be done in one line as:
583         % vo = R*(v - center) + center
584         % pick out the vectors of rotated x- and y-data
585         x_rotated = vo(1,:);
586         y_rotated = vo(2,:);
587         % plot(x, y, 'k-', x_rotated, y_rotated, 'r-', x_center, y_center, 'bo');
588         rounded_rotated_x=floor(x_rotated);
589         rounded_rotated_y=floor(y_rotated);
590
591         %creating data vektor on rotated line
592         z=[];
593         for i=1:size(rounded_rotated_x,2)
594             z = [z,im(rounded_rotated_y(i),rounded_rotated_x(i))];
595         end
596
597         dx=app.ConfEvalPixelsizeEditField.Value;
598         dy=app.ConfEvalPixelsizeEditField.Value;
599
600         Fs_x = 1/dy;     % centimeters per pixel
601
602         [M, N] = size(z);      % pixels
603         x = dx*(0:N-1)';               % centimeters
604
605         dFx = Fs_x/N;              % cycles per centimeter
606
607         Fx = (-Fs_x/2:dFx:Fs_x/2-dFx)';     % cycles per centimeter
608
609         zFFT = fft(z);
610         P2 = abs(zFFT/N);
611         P1 = P2(1:floor(N/2)+1);
612         P1(2:end-1) = 2*P1(2:end-1);
613         f = Fs_x*(0:(N/2))/N;
614 %         plot(gca,f,P1,'LineWidth',lineWidth);
615 %         set(gca,'FontSize',fontsizeTickLabels);
616 %         titleText = sprintf('2D Spectral Distribution Y (z = %.3f mm)',zPos);
617 %         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
618 %         ylabel('$\textrm{Intensity}\textrm{ in Greyscale}$ ','Interpreter','latex
```

```
    ','FontSize',fontsizeLabels);
%         xlabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
    Interpreter','latex','FontSize',fontsizeLabels);
%         grid on;
%
%         figure;
%         absP1log=10*log10(P1./max(P1));
%         plot(gca,f,absP1log,'LineWidth',lineWidth);
%         set(gca,'FontSize',fontsizeTickLabels);
%         titleText = sprintf('2D Spectral Distribution Y (z = %.3f mm)',zPos);
%         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
%         ylabel('$\textrm{Intensity}\textrm{ in }dB$ ','Interpreter','latex','
    FontSize',fontsizeLabels);
%         xlabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
    Interpreter','latex','FontSize',fontsizeLabels);
%         grid on;

%         figure;
        k=0.0001;
        scaledP1 = P1.*k;
        c=255/log10(1+max(max(scaledP1)));
        absP1log = c*log10(1+scaledP1);

        plot(gca,f,absP1log,'LineWidth',lineWidth);
        set(gca,'FontSize',fontsizeTickLabels);
        titleText = sprintf('2D Spectral Distribution Y (z = %.3f mm)',zPos);
        title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
        ylabel('$\textrm{Intensity}\textrm{ in }dB$ ','Interpreter','latex','
            FontSize',fontsizeLabels);
        xlabel('$\textrm{Spatial frequency } f_{y} \textrm{ in } 1/\mu m$ ','
            Interpreter','latex','FontSize',fontsizeLabels);
        hold off;
    case 'Plot Close Up of Beamprofile'
        xCenterInPixel = Mp.getBeamCenterX_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        yCenterInPixel = Mp.getBeamCenterY_mean/app.ConfEvalPixelsizeEditField.Value
            ();
        zPos = Mp.getZPos()/1e+3;
        dWxInPixel = Mp.getdWx_mean/app.ConfEvalPixelsizeEditField.Value()+0;
        dWyInPixel= Mp.getdWy_mean/app.ConfEvalPixelsizeEditField.Value()+0;
        titleText = sprintf('Beamprofile Close - Up (z = %.3f mm)',zPos);

        if dWxInPixel > dWyInPixel
            dWsym = dWxInPixel;
        else
            dWsym = dWyInPixel;
        end
        centerY = (round(yCenterInPixel+dWsym)-round(yCenterInPixel-dWsym))/2;
        centerX = (round(xCenterInPixel+dWsym)-round(xCenterInPixel-dWsym))/2;
```

```
660         set(gca,'FontSize',fontsizeTickLabels);
661         ims = image(im(round(yCenterInPixel-dWsym):round(yCenterInPixel+dWsym),...
662             round(xCenterInPixel-dWsym):round(xCenterInPixel+dWsym)));
663         axis image;
664         ims.XData = (ims.XData-centerX)*app.ConfEvalPixelsizeEditField.Value;
665         ims.YData = (ims.YData-centerY)*app.ConfEvalPixelsizeEditField.Value;
666         title(titleText,'Interpreter','latex','FontSize',fontsizeLabels)
667         set(gca,'FontSize',fontsizeTickLabels,'GridLineStyle','-','GridAlpha',1,'
                GridColor','k');
668         %         zlabel('$\textrm{Intensity}$ ','Interpreter','latex','FontSize
                ',16);
669         xlabel('$\textrm{x - Coordinate in } \mu m$','Interpreter','latex','FontSize
                ',fontsizeLabels);
670         ylabel('$\textrm{y - Coordinate in } \mu m$','Interpreter','latex','FontSize
                ',fontsizeLabels);
671         a = colorbar('FontSize',20);
672         a.Ticks = linspace(0, 255, 6);
673         a.TickLabels = linspace(0, 255, 6);
674         ylabel(a,'Intensity 0 - 255 (false color)','Interpreter','latex','FontSize',
                fontsizeLabels);
675         colormap(jet(256));
676     case 'Plot Diameter X squared Datapoints with fitted curve'
677         f = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getfxFit();
678         dW = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getdWx();
679         NoMP = size(app.EvalResMeasDataResultDropDown.Value.
                getEvaluatedMeasurementData().getMeasurementDataSet(),2);
680         temp = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getMeasurementDataSet();
681         NoI = size(temp(1).getDataSet(),2);
682         dWStd = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData
                ().getdWxStd();
683         zPos = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getzPos();
684         corrMethod = app.EvalResMeasDataResultDropDown.Value.getBackgroundCorrMethod
                ();
685         lambda = app.EvalResMeasDataResultDropDown.Value.getLambda();
686         integAreaFact = app.EvalResMeasDataResultDropDown.Value.getIntegAreaFactor;
687         legendText = sprintf('Corr Method: %s, \x03bb = %i nm, IAF = %.1f, NoMP = %i
                , NoI = %i',corrMethod,lambda,integAreaFact,NoMP,NoI);
688         hold on;
689
690         j = plot(zPos,dW.^2,'bx');
691         set(j,'LineWidth', 0.75*lineWidth,'MarkerSize',15);
692         h = plot(f,'r-',zPos,dW.^2,'bx');
693         set(h,'LineWidth', lineWidth,'MarkerSize',15);
694         uistack(j,'top')
695         set(gca,'FontSize',fontsizeTickLabels);
```

```
696         legend('FontSize',fontsizeLegend);
697
698         title('','Interpreter','latex','FontSize',fontsizeLabels)
699         ylabel('$\textrm{Beamdiameter } D_{x}^{2} \textrm{ in } \mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
700         xlabel('$\textrm{z — Coordinate in }\mu m$ ','Interpreter','latex','FontSize
                ',fontsizeLabels);
701     case 'Plot Diameter Y squared Datapoints with fitted curve'
702         f = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getfyFit();
703         dW = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData().
                getdWy();
704         NoMP = size(app.EvalResMeasDataResultDropDown.Value.
                getEvaluatedMeasurementData().getMeasurementDataSet(),2);
705         temp = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getMeasurementDataSet();
706         NoI = size(temp(1).getDataSet(),2);
707         dWStd = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData
                ().getdWyStd();
708         zPos = app.EvalResMeasDataResultDropDown.Value.getEvaluatedMeasurementData()
                .getzPos();
709
710         corrMethod = app.EvalResMeasDataResultDropDown.Value.getBackgroundCorrMethod
                ();
711         lambda = app.EvalResMeasDataResultDropDown.Value.getLambda();
712         integAreaFact = app.EvalResMeasDataResultDropDown.Value.getIntegAreaFactor;
713         legendText = sprintf('Corr Method: %s, \x03bb = %i nm, IAF = %.1f, NoMP = %i
                , NoI = %i',corrMethod,lambda,integAreaFact,NoMP,NoI);
714         hold on;
715
716         j = plot(zPos,dW.^2,'bx');
717         set(j,'LineWidth', 0.75*lineWidth,'MarkerSize',15);
718         h = plot(f,'r—',zPos,dW.^2,'bx');
719         set(h,'LineWidth', lineWidth,'MarkerSize',15);
720         uistack(j,'top')
721         set(gca,'FontSize',fontsizeTickLabels);
722         legend('FontSize',fontsizeLegend);
723
724         title('','Interpreter','latex','FontSize',fontsizeLabels)
725         ylabel('$\textrm{Beamdiameter } D_{y}^{2} \textrm{ in } \mu m$ ','
                Interpreter','latex','FontSize',fontsizeLabels);
726         xlabel('$\textrm{z — Coordinate in }\mu m$ ','Interpreter','latex','FontSize
                ',fontsizeLabels);
727 end
728 ax.XGrid = 'on';
729 ax.YGrid = 'on';
730 end
```

7.49: Function EvalResSaveMeasButton_ButtonPushedFcn

```
function EvalResSaveMeasButton_ButtonPushedFcn(app, src, event)
%EvalResSaveMeasButton_ButtonPushedFcn ButtonPushedFcn listening to the
%Save Button.
loadedMeas = app.loadedMeas;
uisave({'loadedMeas'});
end
```

7.50: Function EvalResSmpDataDropDown_ValueChangedFcn

```
function EvalResSmpDataDropDown_ValueChangedFcn(app, src, event)
%EvalResSmpDataDropDown_ValueChangedFcn ValueChangedFcn listening to
%changes to the currently shown Measuring Point in the Evaluation Results
%Tab and triggering the function to update Measuring Point data.
SetEvalResSmpDataParams(app,event.Value);
end
```

7.51: Function HSCamConfApplySettingButton_ButtonPushedFcn

```
function HSCamConfApplySettingButton_ButtonPushedFcn(app,src,event)
%HSCamConfApplySettingButton_ButtonPushedFcn ButtonPushedFcn listening to
%the Apply Button to adjust the ExposureTime of the Camera.
src = getselectedsource(app.vid);
src.ExposureTimeAbs = app.HSCamConfExposureTimeEditField.Value;
src.ExposureTimeRaw = app.HSCamConfExposureTimeEditField.Value;
```

7.52: Function HSCamConfCamPreviewButton_ValueChangedFcn

```
function HSCamConfCamPreviewButton_ValueChangedFcn(app, src, event)
%HSCamConfCamPreviewButton_ValueChangedFcn ValueChangedFcn listening to the
%state button to enable and disable the preview of the camera.
if src.Value
app.HSCamConfCamUIAxes.Visible = 'on';
app.vid.FramesPerTrigger = 1;
vidsrc = getselectedsource(app.vid);
delay = CalculatePacketDelay(app.vid, 30);
vidsrc.PacketDelay = delay;
vidRes = app.vid.VideoResolution;
nBands = app.vid.NumberOfBands;
hImage = image( zeros(vidRes(2), vidRes(1),nBands),'Parent', app.HSCamConfCamUIAxes)
    ;
drawnow;
preview(app.vid,hImage);
wait(app.vid);
else
stoppreview(app.vid);
wait(app.vid);
app.HSCamConfCamUIAxes.Visible = 'off';
vidsrc = getselectedsource(app.vid);
delay = CalculatePacketDelay(app.vid, 1);
vidsrc.PacketDelay = delay;
drawnow;
```

```
end
end
```

7.53: Function HSCamConfCamTestButton_ButtonPushedFcn

```
function HSCamConfCamTestButton_ButtonPushedFcn(app, src, event)
%HSCamConfCamTestButton_ButtonPushedFcn ButtonPushedFcn listening to the
%Trigger Image button to create Testimages with the current ExposureTime
%Setting.

% Initiate the acquisition.
start(app.vid);

% Trigger the acquisition.
trigger(app.vid)

% Wait for the acquisition to end.
wait(app.vid, 10);

% Determine the number frames acquired.
frameslogged = app.vid.FramesAcquired;

data = getdata(app.vid);
imshow(data);
colormap(gca, jet(256));
uisave('data');

end
```

7.54: Function HSCamConfConnectButton_ButtonPushedFcn

```
function HSCamConfConnectButton_ButtonPushedFcn(app, src, event)
%HSCamConfConnectButton_ButtonPushedFcn listening to the Connect Button for
%the camera.
% Creates the camera connection and sets a few properties.
imaqreset();
pause(2);
app.vid = videoinput('gige', 1, 'Mono8');
src = getselectedsource(app.vid);

app.vid.FramesPerTrigger = 1;
% framesPerSecond = CalculateFrameRate(app.vid, app.vid.FramesPerTrigger)
delay = CalculatePacketDelay(app.vid, 1)
src.PacketDelay = delay;
src.ExposureTimeAbs = 130;
src.ExposureTimeRaw = 130;

% Configure the trigger type.
triggerconfig(app.vid, 'manual');

```

```
app.HSCamConfExposureTimeEditField.Value = src.ExposureTimeAbs;

vidRes = app.vid.VideoResolution;
app.ConfEvalResWidthEditField.Value = vidRes(1);
app.ConfEvalResHeightEditField.Value = vidRes(2);
app.LiveMeasResWidthEditField.Value = vidRes(1);
app.LiveMeasResHeightEditField.Value = vidRes(2);

app.ConfEvalResWidthEditField.Enable = 'off';
app.ConfEvalResHeightEditField.Enable = 'off';
app.LiveMeasResWidthEditField.Enable = 'off';
app.LiveMeasResHeightEditField.Enable = 'off';

app.HSCamConfConnectionLabel.BackgroundColor = [0.2 1 0.3];
app.HSCamConfConnectionLabel.Text = 'Connected';
drawnow;

app.HSCamConfConnectButton.Enable = 'off';
app.HSCamConfDisconnectButton.Enable = 'on';
app.HSCamConfCamTestButton.Enable = 'on';
app.HSCamConfExposureTimeEditField.Enable = 'on';
app.HSCamConfApplySettingButton.Enable = 'on';
app.HSCamConfCamPreviewButton.Enable = 'on';
end
```

7.55: Function HSCamConfDisconnectButton_ButtonPushedFcn

```
function HSCamConfDisconnectButton_ButtonPushedFcn(app, src, event)
%HSCamConfDisconnectButton_ButtonPushedFcn listening to the Disconnect Button for
%the camera.
% Deletes the camera connection and reseset the connection.
imaqreset();

app.HSCamConfExposureTimeEditField.Enable = 'off';

app.HSCamConfConnectionLabel.BackgroundColor = [1 0.07 0.2];
app.HSCamConfConnectionLabel.Text = 'Diconnected';
drawnow;

app.HSCamConfConnectButton.Enable = 'on';
app.HSCamConfDisconnectButton.Enable = 'off';
app.HSCamConfCamTestButton.Enable = 'off';
app.HSCamConfExposureTimeEditField.Enable = 'off';
app.HSCamConfApplySettingButton.Enable = 'off';
app.HSCamConfCamPreviewButton.Enable = 'off';
end
```

7.56: Function HSStepmotorConfApplySettingsButton_ButtonPushedFcn

```
function HSStepmotorConfApplySettingsButton_ButtonPushedFcn(app,src,event)
```

```
%HSStepmotorConfApplySettingsButton_ButtonPushedFcn listening to the Aplly Button
%for the Stepmotor Settings.
%   Creates the Stepper object and applies the stepper properties.
if isempty(app.arduinoUNOShieldStepmotor)
app.arduinoUNOShieldStepmotor = stepper(app.arduinoUNOShield,1,app.
    HSStepmotorConfStepsPerRevolutionEditField.Value);
app.HSStepmotorConfStepsPerRevolutionEditField.Enable = 'off';
end
app.arduinoUNOShieldStepmotor.RPM = app.HSStepmotorConfRpmEditField.Value;
end
```

7.57: Function HSStepmotorConfComPortRefreshButton_ButtonPushedFcn

```
function HSStepmotorConfComPortRefreshButton_ButtonPushedFcn(app,src,event)
%HSStepmotorConfComPortRefreshButton_ButtonPushedFcn listening to refresh
%button to check for avaiable COM Ports.

app.HSStepmotorConfComPortDropDown.Items = seriallist;
app.HSStepmotorConfComPortDropDown.ItemsData = seriallist;
end
```

7.58: Function HSStepmotorConfConnectButton_ButtonPushedFcn

```
function HSStepmotorConfConnectButton_ButtonPushedFcn(app,src,event)
%HSStepmotorConfConnectButton_ButtonPushedFcn listening to the connect
%button of the Steppermotor and creating a arduino connnectiom element.

selComPort=app.HSStepmotorConfComPortDropDown.Value;
if isempty(app.arduinoUNO)
app.arduinoUNO = arduino(selComPort,'Uno','Libraries','Adafruit\MotorShieldV2');
configurePin(app.arduinoUNO,'D3','pullup');
app.arduinoUNOShield = addon(app.arduinoUNO,'Adafruit\MotorShieldV2');

app.currPos = 0;

app.HSStepmotorConfConnectionLabel.BackgroundColor = [0.2 1 0.3];
app.HSStepmotorConfConnectionLabel.Text = 'Connected';
drawnow;

app.HSStepmotorConfConnectButton.Enable = 'off';
app.HSStepmotorConfDisconnectButton.Enable = 'on';
app.HSStepmotorConfRpmEditField.Enable = 'on';
app.HSStepmotorConfApplySettingsButton.Enable = 'on';
app.HSStepmotorConfStepsPerRevolutionEditField.Enable = 'on';
end
end
```

7.59: Function HSTestHardwareGoToPositionButton_ButtonPushedFcn

```
function HSTestHardwareGoToPositionButton_ButtonPushedFcn(app,src,event)
%HSTestHardwareGoToPositionButton_ButtonPushedFcn listening to the Go
```

```
%Button of the HardwareSettingTest area. Makes the Steppermotor go to a
%specified position
if app.HSTestHardwareGoToPositionEditField.Value < 0
    uialert(app.UIFigure,{'YOU FOOL! One does not simply walk into mordor ... ahm i
        mean zero!'},'You Fool Information','Icon','warning','CloseFcn','uiresume(
        gcbf)');
    uiwait(gcbf)
    return;
end
    moveToZpos(app,app.HSTestHardwareGoToPositionEditField.Value);
end
```

7.60: Function HSTestHardwareGoToZeroButton_ButtonPushedFcn

```
function HSTestHardwareGoToZeroButton_ButtonPushedFcn(app,src,event)
%HSTestHardwareGoToZeroButton_ButtonPushedFcn listening to the GoToZero
%button and making the stage go to the zero position.
moveToZpos(app,0);
end
```

7.61: Function HSTestHardwareReferenceStageButton_ButtonPushedFcn

```
function HSTestHardwareReferenceStageButton_ButtonPushedFcn(app,src,event)
%HSTestHardwareReferenceStageButton_ButtonPushedFcn listening to the
%Reference Stage button starting the referencing process of the stage.
disp('Started Stage referencing.');
app.HSTestHardwareMotorStatusLabel.BackgroundColor = [0.2 1 0.3];
app.HSTestHardwareMotorStatusLabel.Text = 'Motor On';
drawnow;
limitSwitchStatus = readDigitalPin(app.arduinoUNO,'D3');
while limitSwitchStatus
    move(app.arduinoUNOShieldStepmotor,-1);
    limitSwitchStatus = readDigitalPin(app.arduinoUNO,'D3');

end
app.currPos(1) = 0;
disp('Stage referencing is finished.');
app.AutoMeasISO111146Part1CurrPosEditField.Value = app.currPos(1);
app.HSTestHardwareCurrPosEditField.Value = app.currPos(1);
app.currPosRoi.Position = [app.currPos(1) 0];
app.currPosRoiAutoMeas.Position = [app.currPos(1) 0];
app.HSTestHardwareMotorStatusLabel.BackgroundColor = [1 0.07 0.2];
app.HSTestHardwareMotorStatusLabel.Text = 'Motor Off';
drawnow;
end
```

7.62: Function LiveMeasBackgroundMeasButton_ButtonPushedFcn

```
function LiveMeasBackgroundMeasButton_ButtonPushedFcn(app, src, event)
%LiveMeasBackgroundMeasButton_ButtonPushedFcn listening to the Measure
%Background button and starting background measurement at the current
```

```
%position.
app.LiveMeasStartStopButton.Enable = 'off';
app.LiveMeasBackgroundMeasButton.Enable = 'off';
app.LiveMeas = Measurement(999999,...
    999999,...
    999999,...
    999999);

% Reset old Measurementdata
app.LiveMeas.reset();

MeasurementPoint = SingleMeasurementPoint(app.currPos(1));
for k = 1 : app.LiveMeasNoOfImEditField.Value
    app.AutoMeasISO111146Part1ProgressCurrMeasPointMeasEditField.Value = int2str(k);
    MeasurementPoint.addMeasurement(SingleMeasurement(TriggerCameraImage(app)));
    pause(1);
end
app.LiveMeas.getBackgroundData().addSingleMeasurementPoint(MeasurementPoint);
app.LiveMeasStartStopButton.Enable = 'on';
app.LiveMeasBackgroundMeasButton.Enable = 'on';
end
```

7.63: Function LiveMeasDrawRectButton_ButtonPushedFcn

```
function LiveMeasDrawRectButton_ButtonPushedFcn(app, src, event)
%LiveMeasDrawRectButton_ButtonPushedFcn listening to guessBeamarea button
%to draw the first guess of the beam area and beamcenter.
if not(isa(app.LiveMeasRectRoi,'images.roi.Rectangle')) || not(isvalid(app.
    LiveMeasRectRoi))
    app.LiveMeasRectRoi = drawrectangle(app.LiveMeasPreviewImUIAxes,'LabelVisible','
        hover',...
        'Label','beamareaGuess','Position',[1 1 200 200]);
    addlistener(app.LiveMeasRectRoi,'MovingROI',@(src, evt)
        drawLiveMeasRectAllevents(app,src,evt));
    addlistener(app.LiveMeasRectRoi,'ROIMoved',@(src, evt) drawLiveMeasRectAllevents
        (app,src,evt));
    LiveMeascenterRectangleX = app.LiveMeasRectRoi.Position(1)+app.LiveMeasRectRoi.
        Position(3)/2;
    LiveMeascenterRectangleY = app.LiveMeasRectRoi.Position(2)+app.LiveMeasRectRoi.
        Position(4)/2;
    app.LiveMeasPointRoi = drawpoint(app.LiveMeasPreviewImUIAxes,...
        'Position',[LiveMeascenterRectangleX LiveMeascenterRectangleY],'Color','r',
            ...
        'InteractionsAllowed','none');
end
end
```

7.64: Function LiveMeasStartStopButton_ValueChangedFcn

```
function LiveMeasStartStopButton_ValueChangedFcn(app, src, event)
```

```
%LiveMeasStartStopButton_ValueChangedFcn ValueChangedFcn listening to the
%state button to start and stop the livemeasurement.
if src.Value
    app.LiveMeasStatusLabel.Text = 'Active';
    app.LiveMeasStatusLabel.BackgroundColor = [0.2 1 0.3];
    app.LiveMeasBackgroundMeasButton.Enable = 'off';
else
    app.LiveMeasStatusLabel.BackgroundColor = [1 0.07 0.2];
    app.LiveMeasStatusLabel.Text = 'Inactive';
    app.LiveMeasBackgroundMeasButton.Enable = 'on';
end
coarseMethod = app.LiveMeasBackCorrMethodDropDown.Value;
fineMethod = app.LiveMeasFineBackCorrMethodDropDown.Value;
background = app.LiveMeas.getBackgroundData();
kernelSize = app.LiveMeasKernelSizeEditField.Value;
ntFactor = app.LiveMeasNtEditField.Value;
%Messpunkt Nummer hier immer 1
k = 1;
centerRectangleX = app.LiveMeasRectRoi.Position(1)+app.LiveMeasRectRoi.Position(3)
    /2;
centerRectangleY = app.LiveMeasRectRoi.Position(2)+app.LiveMeasRectRoi.Position(4)
    /2;
sizeRectangleX = app.LiveMeasRectRoi.Position(3);
sizeRectangleY = app.LiveMeasRectRoi.Position(4);
convergeKriterium = app.LiveMeasConvergeCriterionEditField.Value;
faktor = app.LiveMeasIntegAreaEditField.Value;
pixelsize = app.LiveMeasPixelsizeEditField.Value;
manualValue=app.LiveMeasFineCorrValueEditField.Value;
app.vid.FramesPerTrigger = 1;
vidsrc = getselectedsource(app.vid);
delay = CalculatePacketDelay(app.vid, 1);
vidsrc.PacketDelay = delay;

while src.Value
    app.LiveMeasColorMap(2) = app.LiveMeasColorMaxEditField.Value;
    app.LiveMeasColorMap(1) = app.LiveMeasColorMinEditField.Value;
    singleMeasurement = SingleMeasurement(TriggerCameraImage(app));
    corrMeas = singleMeasurement.backgroundCorrectionSingleMeasurement(coarseMethod,
        fineMethod,background,kernelSize,ntFactor,k,manualValue);

    processedMeas = corrMeas.iso11146Part1SingleMeasurement(centerRectangleX*
        pixelsize,centerRectangleY*pixelsize,sizeRectangleX*pixelsize,sizeRectangleY
        *pixelsize,convergeKriterium,faktor,pixelsize);

    cla(app.LiveMeasResultImUIAxes,'reset');
    imshow(processedMeas.getImageData(), 'Parent', app.LiveMeasResultImUIAxes);
    if app.LiveMeasChangeColorMapButton.Value
        colormap(app.LiveMeasResultImUIAxes, gray(256));
        caxis(app.LiveMeasResultImUIAxes,app.LiveMeasColorMap);
```

```
    else
        colormap(app.LiveMeasResultImUIAxes, jet(256));
        caxis(app.LiveMeasResultImUIAxes,app.LiveMeasColorMap);
    end
    colorbar(app.LiveMeasResultImUIAxes,'FontSize',20);
    el = drawellipse(app.LiveMeasResultImUIAxes,'Center',...
        [processedMeas.getbeamCenterX()/pixelsize processedMeas.getbeamCenterY()/
           pixelsize],...
        'SemiAxes',[processedMeas.getdWx()/2/pixelsize processedMeas.getdWy()/2/
           pixelsize],...
        'StripeColor','r','Linewidth',1.75);
    el.InteractionsAllowed = 'none';
    app.LiveMeasImData_dWx.Value = processedMeas.getdWx();
    app.LiveMeasImData_dWy.Value = processedMeas.getdWy();
    app.LiveMeasImData_phi.Value = processedMeas.getAzimutPhi();

end

end
```

7.65: Function LiveMeasTriggerImageButton_ButtonPushedFcn

```
function LiveMeasTriggerImageButton_ButtonPushedFcn(app, src, event)
%LiveMeasTriggerImageButton_ButtonPushedFcn ButtonPushedFcn listening to the
%Trigger Image button to create Testimages with the current ExposureTime
%Setting.

% Initiate the acquisition.
app.vid.FramesPerTrigger = 1;
vidsrc = getselectedsource(app.vid);
delay = CalculatePacketDelay(app.vid, 1);
vidsrc.PacketDelay = delay;
start(app.vid);

% Trigger the acquisition.
trigger(app.vid)

% Wait for the acquisition to end.
wait(app.vid, 10);

% Determine the number frames acquired.
frameslogged = app.vid.FramesAcquired;

hImage = getdata(app.vid);

cla(app.LiveMeasPreviewImUIAxes,'reset');
imshow(hImage, 'Parent', app.LiveMeasPreviewImUIAxes);
colormap(app.LiveMeasPreviewImUIAxes, jet(256));

end
```

7.66: Function moveToZpos

```
function  moveToZpos(app,zPos)
%moveToZpos moves the stage to a specified location
steps = CalcNumOfSteps(app,zPos);

app.HSTestHardwareMotorStatusLabel.BackgroundColor = [0.2 1 0.3];
app.HSTestHardwareMotorStatusLabel.Text = 'Motor On';
drawnow;

move(app.arduinoUNOShieldStepmotor,steps);

app.currPos(1) = app.currPos(1)+steps*app.StepInkrement;
app.AutoMeasISO111146Part1CurrPosEditField.Value = app.currPos(1);
app.HSTestHardwareCurrPosEditField.Value = app.currPos(1);
app.currPosRoi.Position = [app.currPos(1) 0];
app.currPosRoiAutoMeas.Position = [app.currPos(1) 0];

app.HSTestHardwareMotorStatusLabel.BackgroundColor = [1 0.07 0.2];
app.HSTestHardwareMotorStatusLabel.Text = 'Motor Off';
drawnow;
end
```

7.67: Function SetEvalResImDataParams

```
function SetEvalResImDataParams(app,Mp)
%SetEvalResImDataParams sets the Image Data Parameters in the Evaluation
%Result Tab.

app.EvalResImData_W_X.Value = Mp.getW_X();
app.EvalResImData_W_Y.Value = Mp.getW_Y();

app.EvalResImData_W_X_squared.Value = Mp.getW_X_squared();
app.EvalResImData_W_Y_squared.Value = Mp.getW_Y_squared();

app.EvalResImData_W_XY.Value = Mp.getW_XY();
app.EvalResImData_dWx.Value = Mp.getdWx();
app.EvalResImData_dWy.Value = Mp.getdWy();
app.EvalResImData_phi.Value = Mp.getAzimutPhi();

cla(app.EvalResImDataUIAxes,'reset');
imshow(Mp.getImageData(), 'Parent', app.EvalResImDataUIAxes);
if app.EvalResImDataChangeColorMapButton.Value
colormap(app.EvalResImDataUIAxes,gray(256));
caxis(app.EvalResImDataUIAxes,app.EvalResColorMap);
else
    colormap(app.EvalResImDataUIAxes,jet(256));
caxis(app.EvalResImDataUIAxes,app.EvalResColorMap);
end
colorbar(app.EvalResImDataUIAxes,'FontSize',12);
el = drawellipse(app.EvalResImDataUIAxes,'Center',...
```

```
27      [Mp.getbeamCenterX()/app.ConfEvalPixelsizeEditField.Value Mp.getbeamCenterY()/
            app.ConfEvalPixelsizeEditField.Value],...
28      'SemiAxes',[Mp.getdWx()/2/app.ConfEvalPixelsizeEditField.Value Mp.getdWy()/2/app
            .ConfEvalPixelsizeEditField.Value],...
29      'StripeColor','r','Linewidth',1.75);
30  el.InteractionsAllowed = 'none';
31  end
```

7.68: Function SetEvalResMeasDataParams

```
1   function SetEvalResMeasDataParams(app,result)
2   %SetEvalResMeasDataParams sets the Result Parameters in the Evaluation
3   %Result Tab.
4   app.EvalResSmpDataDropDown.Items = {};
5   app.EvalResSmpDataDropDown.ItemsData = [];
6
7
8
9   app.EvalResMeasData_BackCorrMethod.Value = result.getBackgroundCorrMethod();
10  app.EvalResMeasData_Lambda.Value = result.getLambda();
11  app.EvalResMeasData_ConvCrit.Value = result.getConvCriterion();
12  app.EvalResMeasData_IntegAreaFactor.Value = result.getIntegAreaFactor();
13  app.EvalResMeasData_Nt.Value = result.getNt();
14  app.EvalResMeasData_EvalNorm.Value = result.getEvalNorm();
15  app.EvalResMeasData_KernelSizeInPercent.Value = result.getKernelSizeInPercent();
16
17  measData=result.getEvaluatedMeasurementData();
18
19  app.EvalResMeasData_z_0_X.Value = measData.getZ0X();
20  app.EvalResMeasData_z_0_Y.Value = measData.getZ0Y();
21
22  if isreal(measData.getZrX())
23      app.EvalResMeasData_z_R_X.Value = measData.getZrX();
24  else
25      app.EvalResMeasData_z_R_X.Value = 99999999;
26  end
27
28  if isreal(measData.getZrY())
29      app.EvalResMeasData_z_R_Y.Value = measData.getZrY();
30  else
31      app.EvalResMeasData_z_R_Y.Value = 99999999;
32  end
33
34  if isreal(measData.getDSigma0x())
35     app.EvalResMeasData_d_0_X.Value = measData.getDSigma0x();
36  else
37      app.EvalResMeasData_d_0_X.Value = 99999999;
38  end
39
40  if isreal(measData.getDSigma0y())
```

```
   app.EvalResMeasData_d_0_Y.Value = measData.getDSigma0y();
else
    app.EvalResMeasData_d_0_Y.Value = 99999999;
end

if isreal(measData.getThetaSigmaX())
    app.EvalResMeasData_theta_X.Value = measData.getThetaSigmaX();
else
    app.EvalResMeasData_theta_X.Value = 99999999;
end

if isreal(measData.getThetaSigmaY())
    app.EvalResMeasData_theta_Y.Value = measData.getThetaSigmaY();
else
    app.EvalResMeasData_theta_Y.Value = 99999999;
end

if isreal(measData.getMsquaredX())
    app.EvalResMeasData_M_squared_X.Value = measData.getMsquaredX();
else
    app.EvalResMeasData_M_squared_X.Value = 99999999;
end
if isreal(measData.getMsquaredY())
    app.EvalResMeasData_M_squared_Y.Value = measData.getMsquaredY();
else
    app.EvalResMeasData_M_squared_Y.Value = 99999999;
end
app.EvalResMeasData_M_squared_eff.Value = sqrt(app.EvalResMeasData_M_squared_X.Value
    *app.EvalResMeasData_M_squared_Y.Value);

dataSet=result.getEvaluatedMeasurementData().getMeasurementDataSet();
[Useless NoMP] = size(dataSet);

for k = 1:NoMP
    SmP = dataSet(k);
    tempString=sprintf('Measuring Point %i / z-Pos: %.4f mm',k,SmP.getZPos()/1000);
    app.EvalResSmpDataDropDown.Items = [app.EvalResSmpDataDropDown.Items tempString
        ];
    app.EvalResSmpDataDropDown.ItemsData = [app.EvalResSmpDataDropDown.ItemsData SmP
        ];
end
for k = 1:NoMP-1
    for j = k+1:NoMP
    end
    SmP = dataSet(k);
    SmP2 = dataSet(j);
    if abs(SmP.getAzimutPhi_mean() - SmP2.getAzimutPhi_mean()) > 10
        app.EvalResSmpInfoTextArea.Visible = 'on';
    else
```

```
87          app.EvalResSmpInfoTextArea.Visible = 'off';
88      end
89  end
90
91  SetEvalResSmpDataParams(app,dataSet(1))
92
93  end
```

7.69: Function SetEvalResSmpDataParams

```
1   function SetEvalResSmpDataParams(app,SmP)
2   %SetEvalResSmpDataParams sets the MeasuringPoint Parameters in the Evaluation
3   %Result Tab.
4   app.EvalResImDataDropDown.Items = {};
5   app.EvalResImDataDropDown.ItemsData = [];
6
7   app.EvalResSmpData_W_X.Value = SmP.getW_X_mean();
8   app.EvalResSmpData_W_X_std.Value = SmP.getW_X_std();
9   app.EvalResSmpData_W_Y.Value = SmP.getW_Y_mean();
10  app.EvalResSmpData_W_Y_std.Value = SmP.getW_Y_std();
11  app.EvalResSmpData_W_X_squared.Value = SmP.getW_X_squared_mean();
12  app.EvalResSmpData_W_X_squared_std.Value = SmP.getW_X_squared_std();
13  app.EvalResSmpData_W_Y_squared.Value = SmP.getW_Y_squared_mean();
14  app.EvalResSmpData_W_Y_squared_std.Value = SmP.getW_Y_squared_std();
15  app.EvalResSmpData_W_XY.Value = SmP.getW_XY_mean();
16  app.EvalResSmpData_W_XY_std.Value = SmP.getW_XY_std();
17  app.EvalResSmpData_dWx.Value = SmP.getdWx_mean();
18  app.EvalResSmpData_dWx_std.Value = SmP.getdWx_std();
19  app.EvalResSmpData_dWy.Value = SmP.getdWy_mean();
20  app.EvalResSmpData_dWy_std.Value = SmP.getdWy_std();
21  app.EvalResSmpData_phi.Value = SmP.getAzimutPhi_mean();
22  app.EvalResSmpData_phi_std.Value = SmP.getAzimutPhi_std();
23  SmPData = SmP.getDataSet();
24
25  [Useless NoI] = size(SmPData);
26  for j=1:NoI
27      app.EvalResImDataDropDown.Items = [app.EvalResImDataDropDown.Items sprintf('%i',
            j)];
28      app.EvalResImDataDropDown.ItemsData = [app.EvalResImDataDropDown.ItemsData
            SmPData(j)];
29  end
30
31  SetEvalResImDataParams(app,SmPData(1));
32  end
```

7.70: Function TriggerCameraImage

```
1   function image = TriggerCameraImage(app)
2   %triggerCameraImage Triggers image aquisition of the camera
3   for i=1 : 10
```

```
4
5       try
6           start(app.vid);
7
8           % Trigger the acquisition.
9           trigger(app.vid)
10
11          % Wait for the acquisition to end.
12          wait(app.vid, 10);
13
14          % Determine the number frames acquired.
15          frameslogged = app.vid.FramesAcquired;
16
17          image = getdata(app.vid);
18          if ismatrix(image)
19              break;
20          end
21      catch ME
22          disp('HAD TO RESET CAM');
23          imaqreset();
24          pause(2);
25          app.vid = videoinput('gige', 1, 'Mono8');
26          src = getselectedsource(app.vid);
27
28          app.vid.FramesPerTrigger = 1;
29          src.ExposureTimeAbs = 130;
30          src.ExposureTimeRaw = 130;
31
32          % Configure the trigger type.
33          triggerconfig(app.vid, 'manual');
34          app.HSCamConfCamTestButton.Enable = 'on';
35
36  %         start(app.vid);
37  %
38  %         % Trigger the acquisition.
39  %         trigger(app.vid)
40  %
41  %         % Wait for the acquisition to end.
42  %         wait(app.vid, 2);
43  %
44  %         % Determine the number frames acquired.
45  %         frameslogged = app.vid.FramesAcquired;
46  %
47  %         image = getdata(app.vid);
48      end
49  end
50
51  end
```

7.71: Function UpdateGUIConfEvalTab

```
function UpdateGUIConfEvalTab(app)
%UpdateGUIConfEvalTab Updates the whole ConfigureEvaluation Tab and its
%elements.
dataSet=app.loadedMeas.getMeasurementData().getMeasurementDataSet();
[Useless NoMP] = size(dataSet);
app.ConfEvalSmpDropDown.Items = {};
app.ConfEvalSmpDropDown.ItemsData = [];
app.ConfEvalSmpImDropDown.Items = {};
app.ConfEvalSmpImDropDown.ItemsData = [];
for i = 1:NoMP
    SmP = dataSet(i);

    tempString=sprintf('Measuring Point %i / z-Pos: %.4f mm',i,SmP.getZPos()/1000);
    app.ConfEvalSmpDropDown.Items = [app.ConfEvalSmpDropDown.Items tempString];
    app.ConfEvalSmpDropDown.ItemsData = [app.ConfEvalSmpDropDown.ItemsData SmP];
end
SmP = dataSet(1);
SmPData = SmP.getDataSet();
[Useless NoI] = size(SmPData);
for j=1:NoI
    app.ConfEvalSmpImDropDown.Items = [app.ConfEvalSmpImDropDown.Items sprintf('%i',
        j)];
    app.ConfEvalSmpImDropDown.ItemsData = [app.ConfEvalSmpImDropDown.ItemsData
        SmPData(j)];
end
app.ConfEvalPreviewImUIAxes.Visible = 'on';
app.ConfEvalDrawRectButton.Enable = 'on';
app.ConfEvalSmpDropDown.Enable = 'on';
app.ConfEvalSmpImDropDown.Enable = 'on';
imshow(SmPData(1).getImageData(), 'Parent', app.ConfEvalPreviewImUIAxes);
colormap(app.ConfEvalPreviewImUIAxes, jet(256));
app.ConfEvalPreviewImUIAxes.Toolbar.Visible = 'off';
app.ConfEvalPreviewImUIAxes.Interactions = [];
end
```

7.72: Function UpdateGUIEvalResTab

```
function UpdateGUIEvalResTab(app)
%UpdateGUIEvalResTab Updates the whole EvaluationResult Tab and its
%elements.
resultDataSet=app.loadedMeas.getResults();
[Useless NoR] = size(resultDataSet);
app.EvalResMeasDataResultDropDown.Items = {};
app.EvalResMeasDataResultDropDown.ItemsData = [];
for i = 1:NoR
    result = resultDataSet(i);
    tempString=sprintf('Result No.%i',i);
    app.EvalResMeasDataResultDropDown.Items = [app.EvalResMeasDataResultDropDown.
        Items tempString];
```

```
    app.EvalResMeasDataResultDropDown.ItemsData = [app.EvalResMeasDataResultDropDown
        .ItemsData result];
end
SetEvalResMeasDataParams(app,resultDataSet(1));

end
```

7.73: Function UpdatePreviewAxes

```
function UpdatePreviewAxes(app, src, event)
%UpdatePreviewAxes Updates the Measuring Point Axes in the Automated Measurement Tab
    .

if strcmp('on',app.AutoMeasISO111146Part1OffseEditField.Enable)
    offset = app.AutoMeasISO111146Part1OffseEditField.Value;
    while mod(offset,app.StepInkrement) ~= 0
        offset = offset+1;
    end
    app.AutoMeasISO111146Part1OffseEditField.Value = offset;

    timesRayLength = app.AutoMeasISO111146Part1TimesRayleighEditField.Value;
    rayLength = app.AutoMeasISO111146Part1RayleighEditField.Value;
    while mod(rayLength,app.StepInkrement) ~= 0
        rayLength = rayLength+1;
    end
    app.AutoMeasISO111146Part1RayleighEditField.Value = rayLength;
    NoMP = app.AutoMeasISO111146Part1NoMeasPointsEditField.Value;

    xTicks = rayLength*2*timesRayLength/(NoMP-1);
    while mod(xTicks,app.StepInkrement) ~= 0
        NoMP = NoMP + 1;
        xTicks = rayLength*2*timesRayLength/(NoMP-1);
    end
    app.AutoMeasISO111146Part1NoMeasPointsEditField.Value = NoMP;
    app.measurementPoints = (offset-xTicks*(NoMP-1)/2:xTicks:offset+xTicks*(NoMP-1)
        /2);

    if isempty(app.centerMarker) || not(ishandle(app.centerMarker))
        app.centerMarker = line(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes,
            offset,0,'Color','blue','Marker','o','MarkerSize',10,'LineStyle','none',
            'LineWidth',2);
    else
        app.centerMarker.XData = offset;
        app.centerMarker.YData = 0;
    end
    app.centerMarker.Visible = 'on';
else
    area = app.AutoMeasISO111146Part1MeasAreaEditField.Value;
    while mod(area,app.StepInkrement) ~= 0
        area = area+1;
```

```
    end
    app.AutoMeasISO111146Part1MeasAreaEditField.Value = area;
    NoMP = app.AutoMeasISO111146Part1NoMeasPointsEditField.Value;

    xTicks = area/(NoMP-1);
    while mod(xTicks,app.StepInkrement) ~= 0
        NoMP = NoMP + 1;
        xTicks = area/(NoMP-1);
    end
    app.AutoMeasISO111146Part1NoMeasPointsEditField.Value = NoMP;
    app.measurementPoints = (0:xTicks:xTicks*(NoMP-1));

    app.centerMarker.Visible = 'off';
end

 y = zeros(1,size(app.measurementPoints,2));
    app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XLim = [0-600 app.
        maxValueStage+600];

    XTICK = unique([0,app.measurementPoints,app.maxValueStage]);
    app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XTick = XTICK;
    app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes.XTickLabel = XTICK;

    hold(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes,'on')
    if isempty(app.previewMarkers)
        app.previewMarkers = line(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes,
            app.measurementPoints,y,'Color','red','Marker','x','MarkerSize',15,'
            LineStyle','none','LineWidth',1);
    else
        app.previewMarkers.XData = app.measurementPoints;
        app.previewMarkers.YData = y;
    end

    if isempty(app.endPointMarkers)
        app.endPointMarkers = line(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes
            ,[app.measurementPoints(1),app.measurementPoints(end)],[0,0],'Color','
            green','Marker','o','MarkerSize',10,'LineStyle','none','LineWidth',2);
    else
        app.endPointMarkers.XData = [app.measurementPoints(1),app.measurementPoints(
            end)];
        app.endPointMarkers.YData = [0,0];
    end

    hold(app.AutoMeasISO111146Part1PreviewMeasPointsUIAxes,'off')

if app.measurementPoints(1)<0 || app.measurementPoints(end)>app.maxValueStage
    app.AutoMeasISO111146Part1StartMeasButton.Enable = 'off';
    return;
end
```

```
81 app.AutoMeasISO111146Part1StartMeasButton.Enable = 'on';
82
83 end
```

Anhang 2: Installationsanleitung

Damit das Messprogramm verwendbar ist, müssen Hardware und Software korrekt eingerichtet werden. Die notwendigen Schritte sind:

1. Installieren des Arduino Board Treibers CH341SER.EXE (notwendig damit der Rechner das Arduino Board im Geräte Manager erkennt)
2. Installieren von MATLAB Version R2020b
3. Installieren von Basler Pylon Viewer Version 6.1.1.19832
4. Verbinden der Basler Kamera über Ethernet
5. Den Schritten in Anhang 3 folgen um den Ethernet Adapter richtig zu konfigurieren
6. Anhang 4 folgen um MATLAB für die Verwendung der Basler Kamera einzurichten
7. Installieren der Arduino Library für MATLAB, damit direkt mit dem Board kommuniziert werden kann
8. Installieren der notwendigen MATLAB Packages:
   - Curve Fitting Toolbox Version 3.5.12 (R2020b)
   - Image Acquisition Toolbox Version 6.3 (R2020b)
   - Image Processing Toolbox Version 11.2 (R2020b)
   - Image Acquisition Toolbox Support Package for GigE Vision Hardware Version 20.2.1 (R2020b)
   - MATLAB Support Package for Arduino Hardware Version 20.2.0 (R2020b)

Sind alle Schritte abgearbeitet sollte es keine Probleme beim öffnen des Programms geben. Geöffnet wird das Programm über *lbcUI_Launcher.m*. Der Startbildschirm der geöffnet werden sollte ist in Abb. 7.1 dargestellt. Die weitere Bedienung ist bereits in der Arbeit selbst beschrieben.

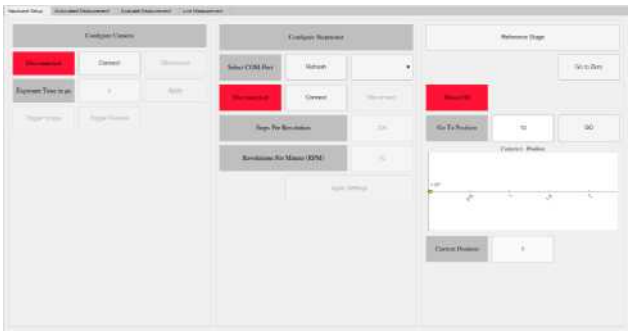


Abbildung 7.1: Startbildschirm des Messprogramms

1

GigE Vision Quick Start Configuration Guide

Configure GigE Vision Image Acquisition on Windows

Acquiring images from a GigE Vision camera in Image Acquisition Toolbox™ using the GigE Vision Hardware support package functionality (either the `videoinput` object using the `gige` adaptor or the `gigecam` object) requires specific configuration and setup for the Ethernet network adapter and network connection outside of MATLAB®. This guide describes the configuration steps to help you get started using a GigE Vision camera with the Image Acquisition Toolbox on a Windows® computer.

The configuration instructions and screen shots included are for Windows 7, but similar configuration steps are applicable for Windows 8 or Windows 10.

Recommended Hardware Setup

The recommended getting-started configuration when using the Image Acquisition Toolbox to acquire images from a single GigE Vision camera consists of a system with:

- GigE Vision compliant camera.
- Computer with a camera-dedicated Gigabit Ethernet network adapter (for example a PCI Express Gigabit Ethernet network interface card), which supports jumbo frames (4k or 9k bytes are common jumbo frame sizes).
- Direct connection between the camera and dedicated Gigabit Ethernet adapter.
- For any additional computer network connection (such as an Internet connection or your organization's local area network) it is recommended to use a separate network adapter.

Although other hardware setups are possible, the above configuration ensures that:

- Maximum bandwidth is available for streaming images from the camera.
- The camera-dedicated private network connection can have network/firewall settings different than the Internet or domain network connection.

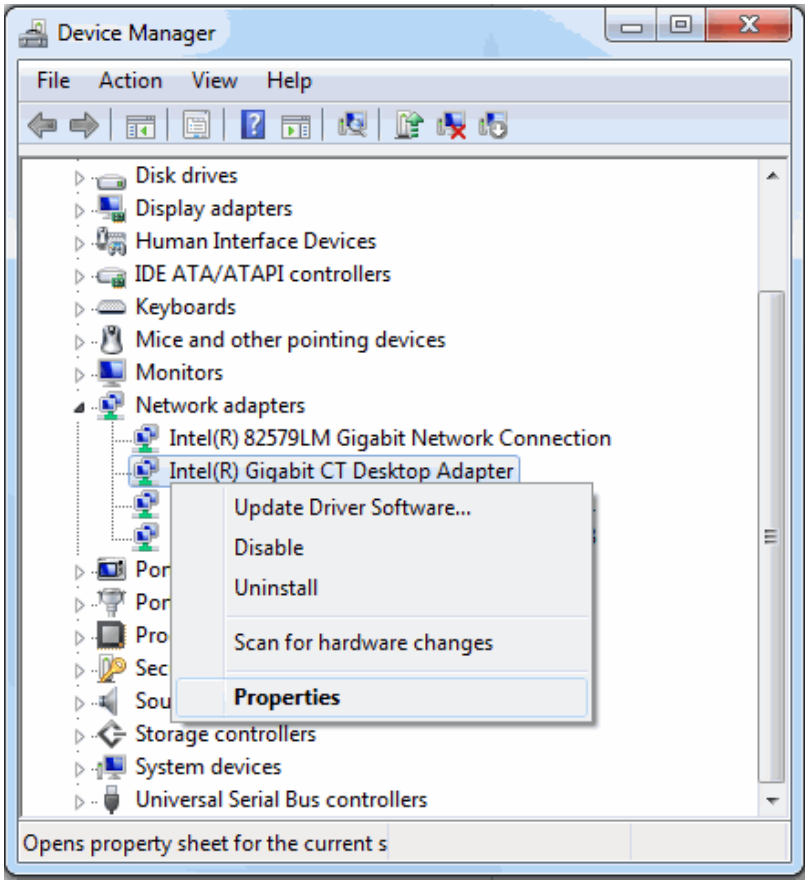
Gigabit Ethernet Network Adapter Configuration

The following Ethernet adapter configuration settings are recommended for proper operation and optimum performance when acquiring images from a GigE Vision camera.

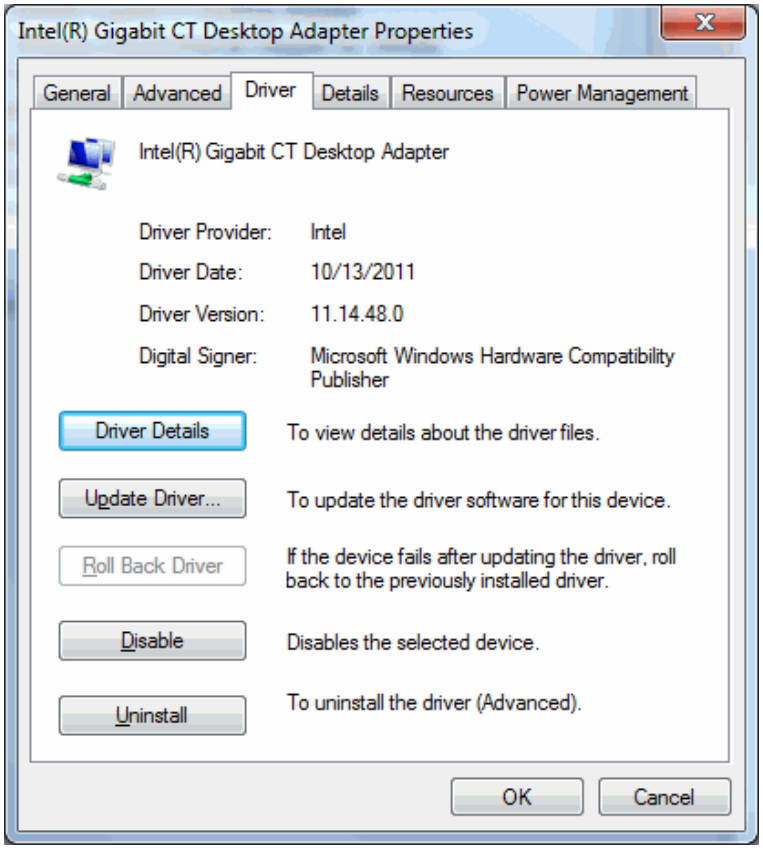
Gigabit Ethernet Adapter Driver

Verify that an appropriate Ethernet network adapter driver (provided by the network adapter manufacturer) is installed and working properly. Custom high-performance drivers installed for use with a third-party imaging application will not work with the Image Acquisition Toolbox `videoinput` and `gigecam` interfaces.

1. Open Windows Device Manager, and click on **Network adapters**.
2. Right-click on the Gigabit Ethernet adapter entry dedicated to the camera's network connection under **Network adapters**, and click **Properties** in the context menu.

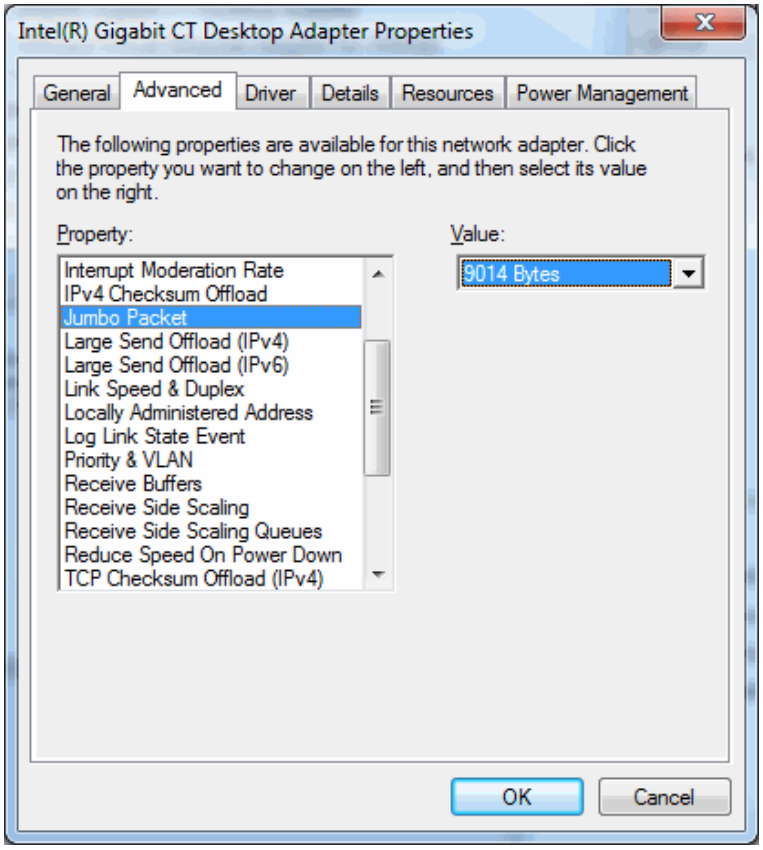


3 In the Adapter Properties, click the **Driver** tab to verify the driver.

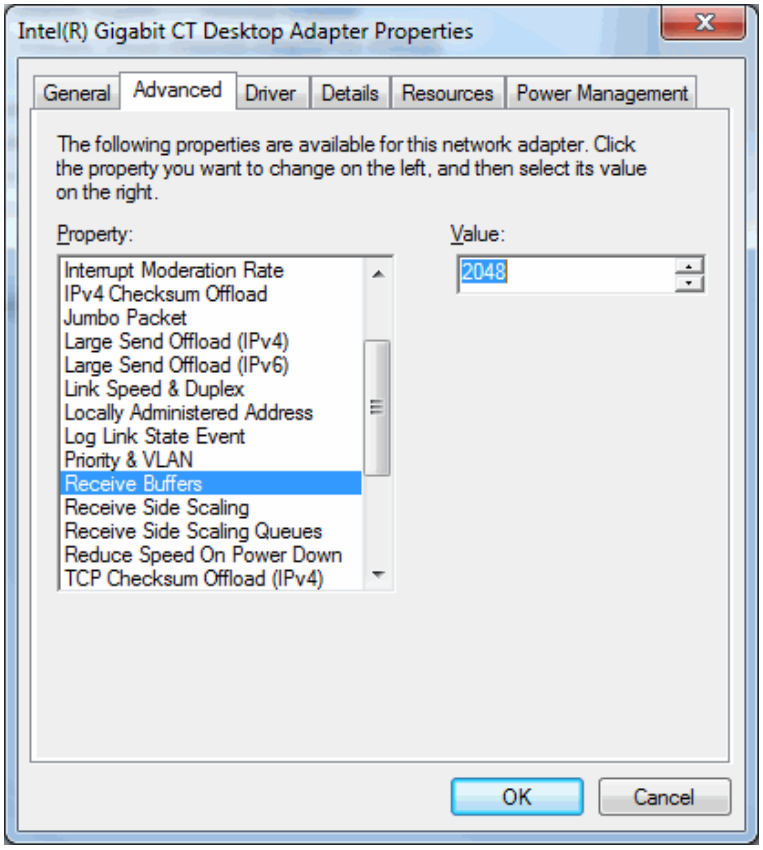


For optimum GigE Vision streaming performance, i.e. reduced CPU load and smaller likelihood of dropped frames, the following settings for the Gigabit Ethernet adapter jumbo packet and receive buffers are recommended:

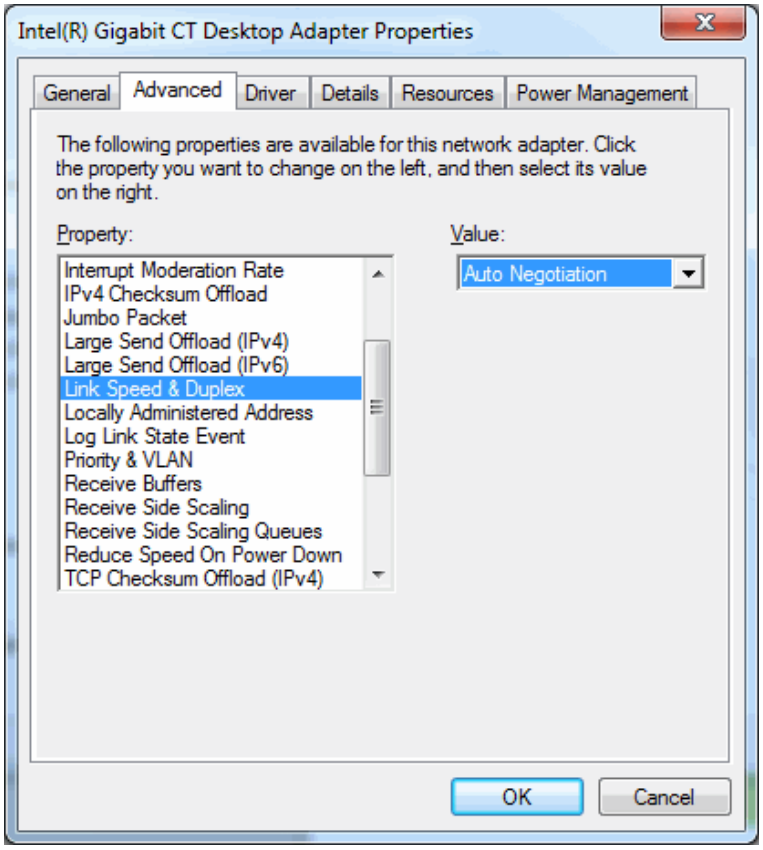
1. In the network Adapter Properties (described in the previous section), click the **Advanced** tab.
2. Set **Jumbo Frame** (or **Jumbo Packet**) to the maximum supported value (for example 9014 bytes). Gigabit Ethernet controllers that support jumbo frames can transfer packet sizes larger than the standard Ethernet frame size (1500 bytes).



3 Set **Receive Buffers** (or **Receive Descriptors**) to the maximum supported value (for example 2048). For some Ethernet controller drivers this setting is grouped under **Performance Options**.



4 Confirm that **Link Speed & Duplex** is set to **Auto Negotiation** (or **Auto Detect**).

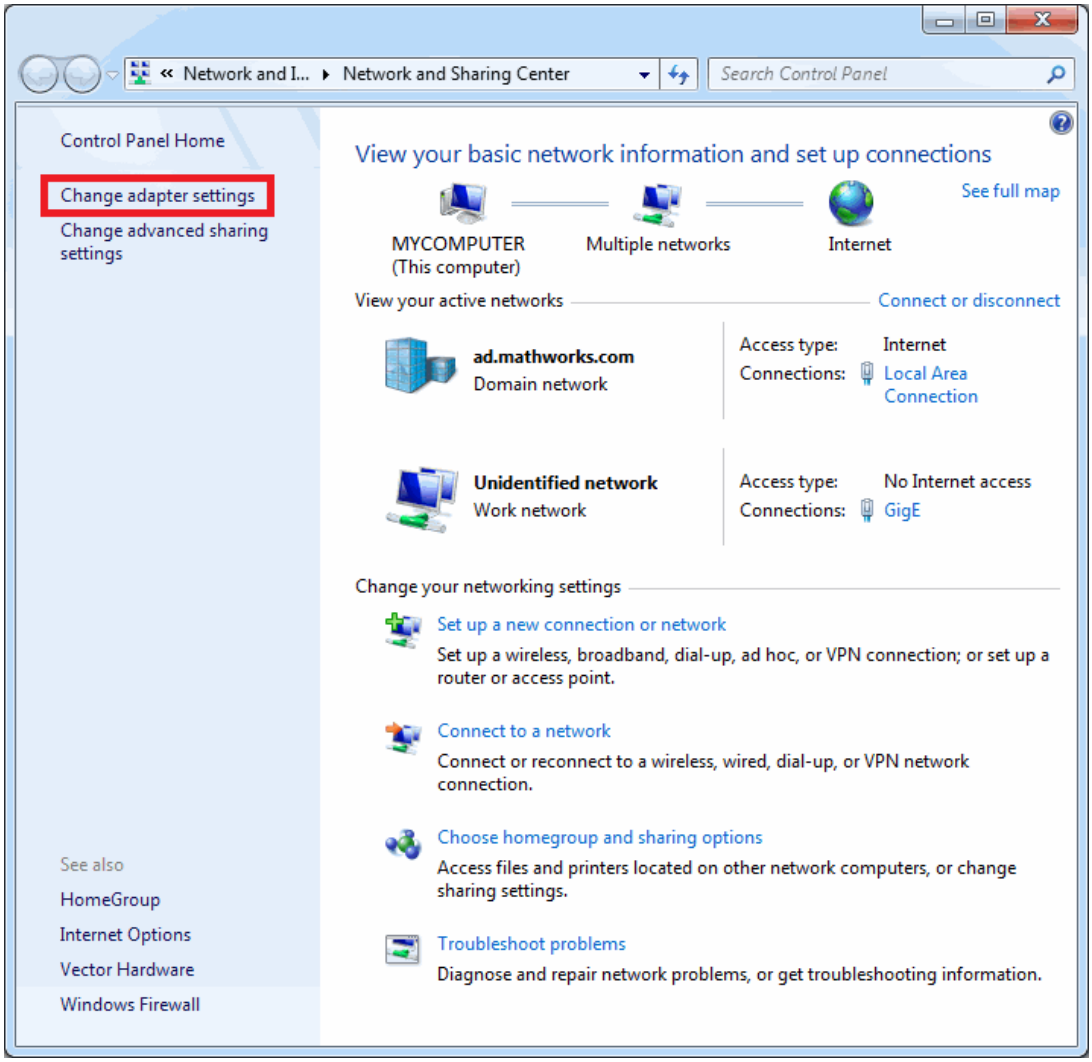


Network Settings for the Camera's Network Connection

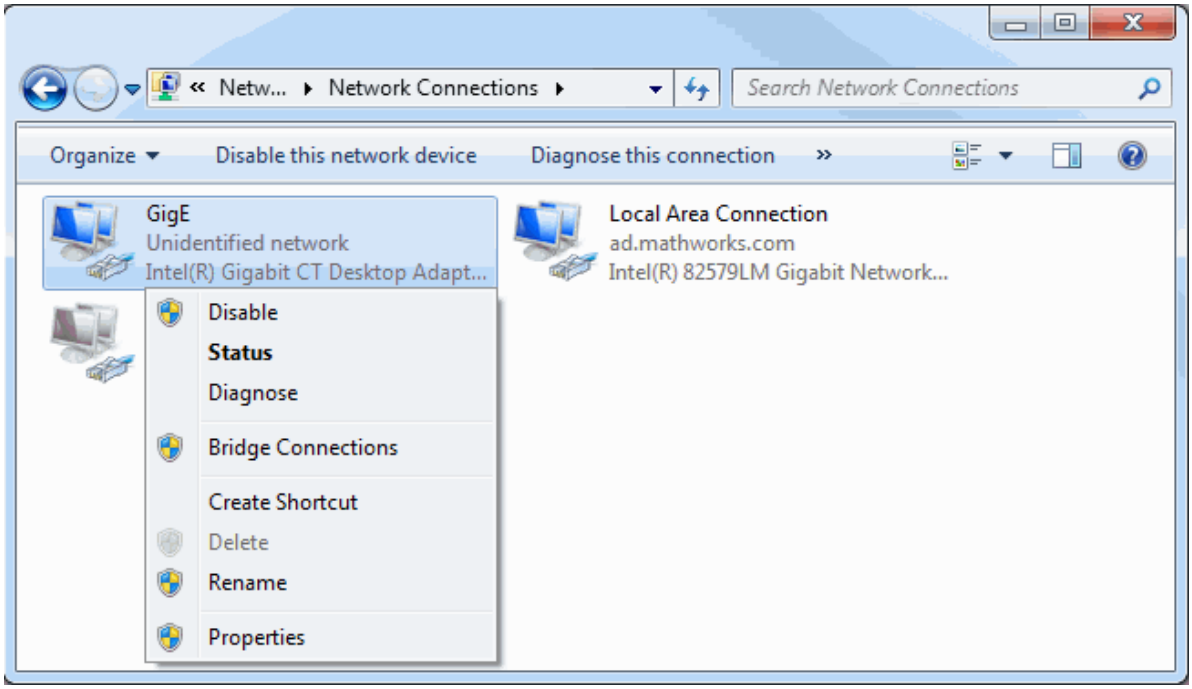
The following settings for the camera's dedicated network connection are recommended for proper operation and optimum performance when acquiring images from a GigE Vision camera.

1 In Windows Control Panel, open the **Network and Sharing Center**.

2 Click on **Change adapter settings**.

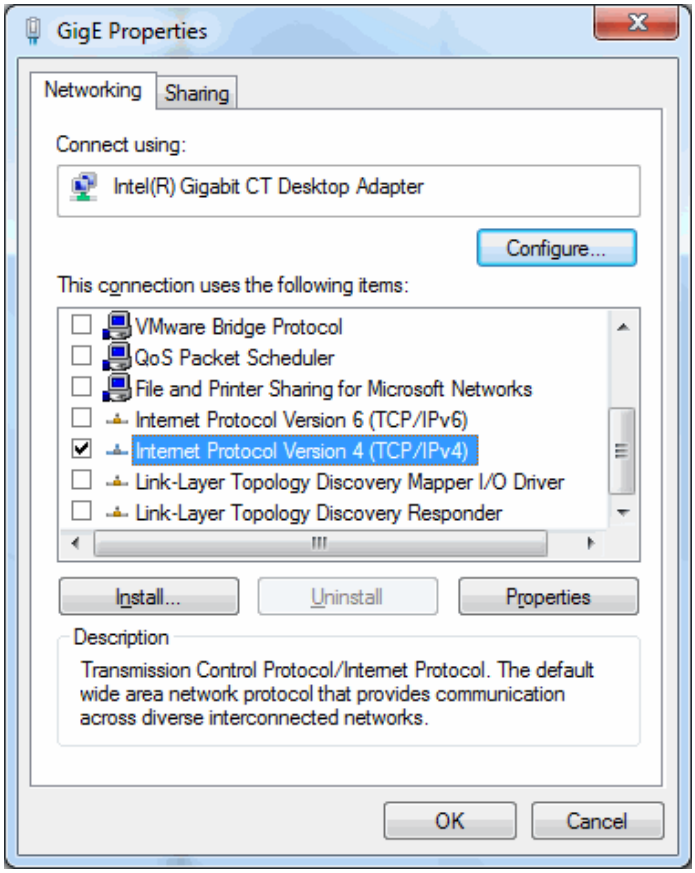


3 Right-click on the camera's network connection and click **Properties**.



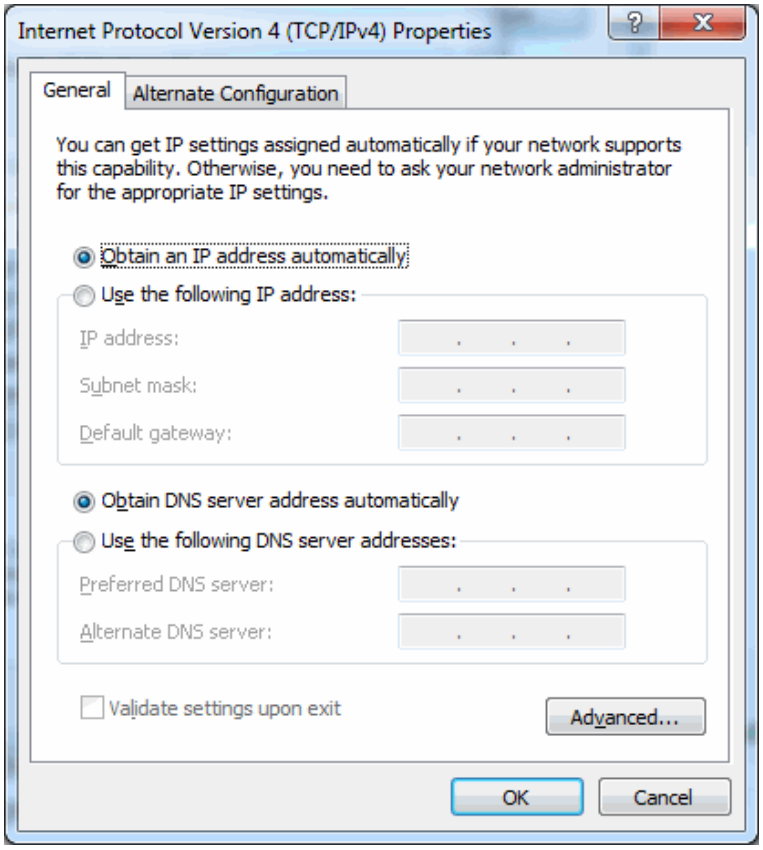
4 In the **Networking** tab, confirm that **Internet Protocol Version 4 (TCP/IPv4)** is enabled/checked, and uncheck all other options.

When a dedicated camera network connection is used, other protocols, clients, or services can be disabled/unchecked, as they are not necessary for GigE Vision control and streaming.



5 To configure the IP address, select **Internet Protocol Version 4 (TCP/IPv4)** and right-click **Properties**.

To configure automatic IP address assignment for the camera network connection, on the **General** tab, select **Obtain an IP address automatically** and **Obtain DNS server address automatically**, and click **OK**.



1-13

Firewalls and the GigE Vision Camera Network Connection

Firewalls can block UDP packets used for image data transfer by the GigE Vision Streaming Protocol. The ports used for image data transfer by the camera and Gigabit network adapter are dynamic, and setting up the required firewall rules is an advanced operation. Firewall processing of the image data stream can also reduce performance.

A convenient getting-started configuration is to turn off the firewall for the camera's dedicated network connection, while keeping the firewall enabled for the other network connections (such as the Internet connection or your organization's local area network connection).

Windows uses network location profiles to group firewall settings for different types of connections, and turning off Windows firewall selectively for an individual network connection is not directly possible. Refer to the following online article for possible network and firewall configurations: http://www.mathworks.com/matlabcentral/answers/232356.

Configure Camera GigE Vision Streaming Parameters in MATLAB

Each image acquired by the camera is transferred to the computer as a data block of UDP packets, which are processed by MATLAB. For a lower CPU load during image acquisition and to prevent dropped frames, you can configure the camera GigE Vision streaming parameters (packet size and packet delay) in MATLAB.

Packet Size

- The toolbox configures the packet size automatically once the connection between MATLAB and the camera is established.
- The packet size value can also be configured manually by setting the `PacketSize` property of the `videoinput` source or `gigecam` objects.
- The packet size value should be set to the largest size that the network adapter and camera can handle (not larger than the Ethernet adapter jumbo packet size configured in step 1).

Packet Delay

- To prevent dropped frames, you can configure the camera to introduce a time delay between the image data packets by setting the `PacketDelay` property of the `videoinput` source or `gigecam` objects.
- Optimum values for the packet delay depend on the packet size, acquired frame size (image height and width), specified pixel format (for example `'Mono8'`), camera frame rate, and other camera-specific settings.

The following online article provides details about determining the recommended packet delay value, and setting the packet size and packet delay values in MATLAB: http://www.mathworks.com/matlabcentral/answers/91834.

Basler pylon

APPLICATION NOTE

How to use the Basler pylon GenTL Producers for Basler GigE and USB 3.0 Cameras with MathWorks MATLAB

Document Number: AW001343
Version: 03 Language: 000 (English)
Release Date: 6 November 2017

BASLER
the power of sight

Contacting Basler Support Worldwide

Europe, Middle East, Africa

Basler AG
An der Strusbek 60–62
22926 Ahrensburg
Germany

Tel. +49 4102 463 515
Fax +49 4102 463 599

support.europe@baslerweb.com

The Americas

Basler, Inc.
855 Springdale Drive, Suite 203
Exton, PA 19341
USA

Tel. +1 610 280 0171
Fax +1 610 280 7608

support.usa@baslerweb.com

Asia-Pacific

Basler Asia Pte. Ltd.
35 Marsiling Industrial Estate Road 3
#05–06
Singapore 739257

Tel. +65 6367 1355
Fax +65 6367 1255

support.asia@baslerweb.com

www.baslerweb.com

Table of Contents

1 Introduction

The Basler pylon GigE and USB GenTL producers enable you to operate Basler GigE Vision and USB3 Vision cameras with MATLAB (www.mathworks.com) or any software libraries that support the GenICam GenTL standard (http://www.emva.org) and implement their own GenTL consumer.

This document describes how to configure MATLAB so that Basler GigE and USB 3.0 cameras are detected in MATLAB. Once the cameras are displayed in MATLAB, you can configure them as desired and acquire images.

The Basler pylon GigE and U3V GenTL producers are included in the pylon Camera Software Suite 5.0.11 or higher which is available for Windows, Linux x86, and Linux ARM. For more information about the pylon Camera Software Suite and to download the latest version go to www.baslerweb.com.

2 Requirements

The procedures described in this document assume that the following software and hardware is present:

- MATLAB R2017a (9.2.0.556344) 64-bit with the Image Acquisition Toolbox version 5.2 (R2017a)
- Image Acquisition Toolbox Support Package for GenICam Interface version 17.2.0.0, which enables you to acquire video and images from GenTL-compliant cameras
- Image Acquisition Toolbox Support Package for GigE Vision Hardware version 17.2.0.0, which is required for advanced IP address configuration and troubleshooting of GigE cameras
- Basler pylon Camera Software Suite 5.0.11 including the pylon GigE and USB GenTL 64-bit producers
- Basler GigE and/or Basler USB 3.0 cameras
- GigE network card and/or USB 3.0 host controller card recommended by Basler
- GigE and or USB 3.0 cables recommended by Basler

The Basler pylon GenTL producers are compliant with GenTL version 1.5.

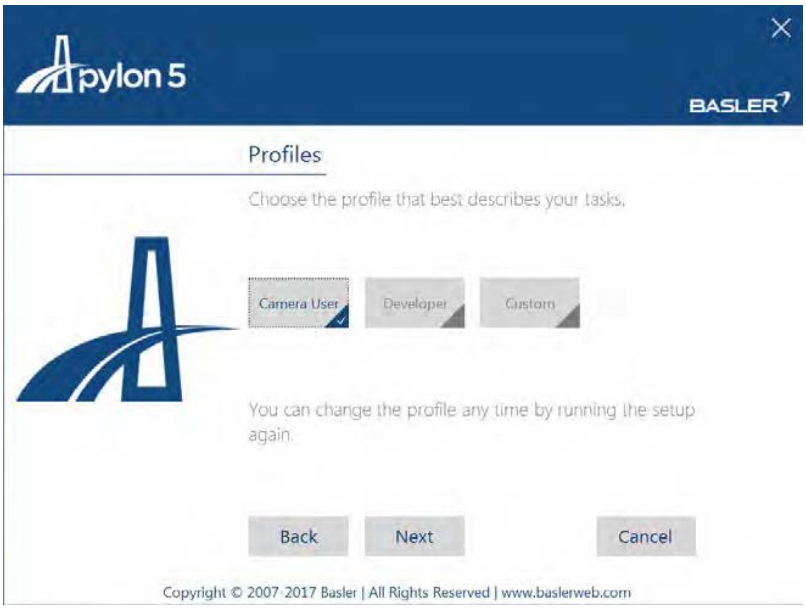
For more information about recommended accessories for Basler GigE and Basler USB 3.0 cameras, visit to the Basler website (www.baslerweb.com) or contact your local Basler Customer Service team.

3 Installation

3.1 Installing the Basler pylon GigE and USB GenTL Producers

To install the Basler pylon GigE and USB GenTL producers:

1. Run Basler pylon 5.0.11.xxx.exe.
2. On the **Profiles** page, choose either the **Camera User** or the **Developer** profile.
   If you intend to develop applications based on the pylon Camera Software Suite APIs, choose the **Developer** profile.



3. Click **Next**.
4. On the **Interfaces** page, select the **Interfaces** for which you want to install the necessary drivers, runtime environment, and GenTL producers, e.g., USB and GigE.

5. Click **Next**.
6. Click **Install** to install the selected components now.

7. After the installation has completed, log off from your computer and then log in again. This is necessary for the changed system environment variables to take effect.

3.2 Installing and Configuring MATLAB

To install and configure MATLAB:

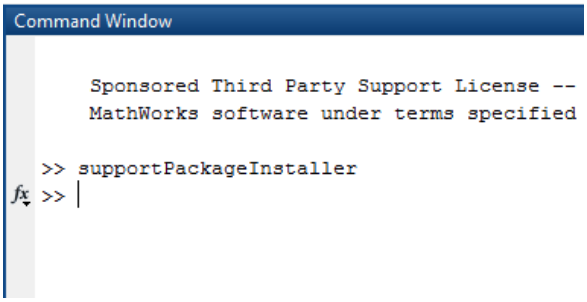
1. In MATLAB, go to the **APPS** tab and make sure that the **Image Acquisition** app is installed.

If the Image Acquisition app has not been installed yet, install it by running the Support Package Installer in one of the following ways:

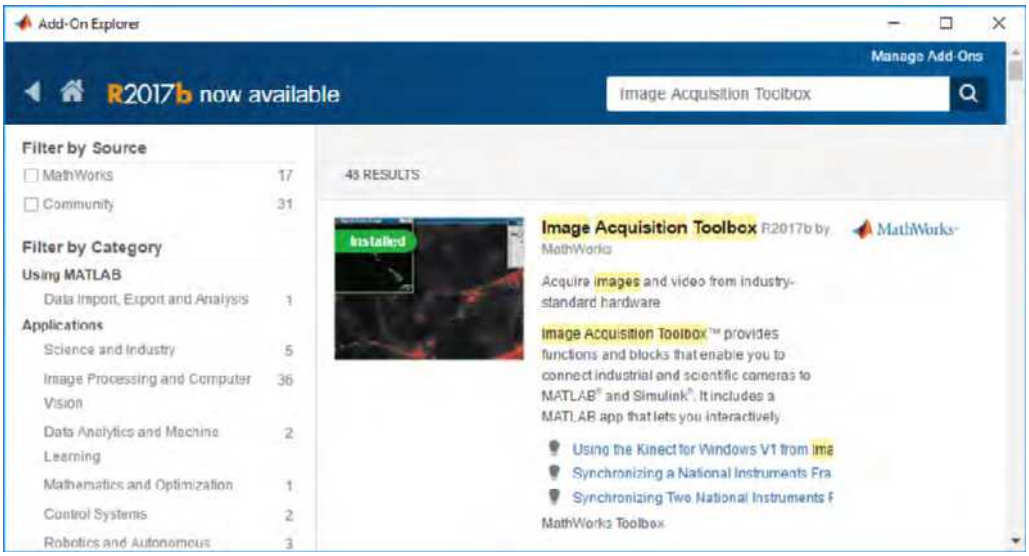
- On the **HOME** tab, click **Add-Ons** > **Get Hardware Support Packages**.

- In the MATLAB **Command Window**, enter:

```
supportPackageInstaller
```



a. In the Add-On Explorer, remove the **Clear Filter** field and enter '**Image Acquisition Toolbox**'.



b. Select the Image Acquisition Toolbox and install it.

2. Check if the MATLAB GenTL consumer is available. In the MATLAB **Command Window**, enter:

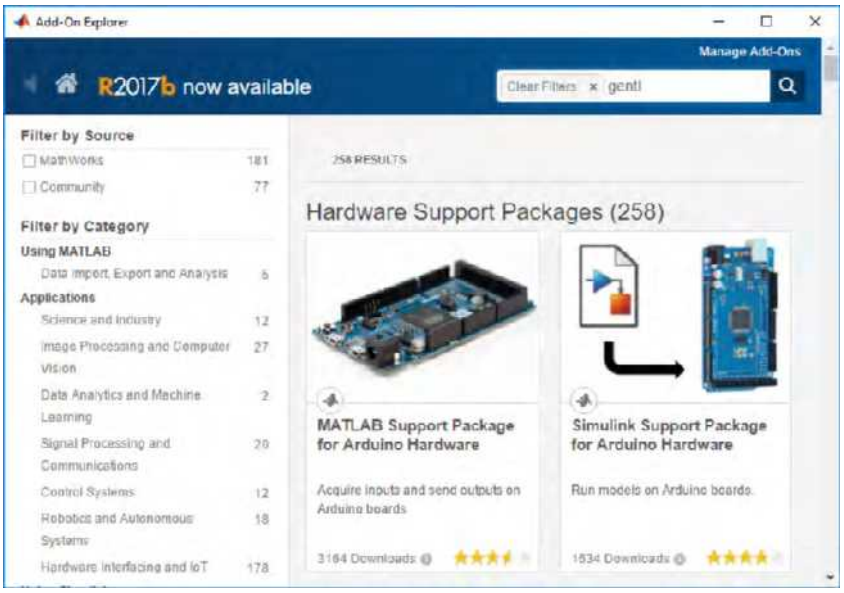
```
imaqhwinfo
```

If '**gentl**' is not listed under **InstalledAdaptors**, install the MATLAB GenTL Consumer in one of the following ways:

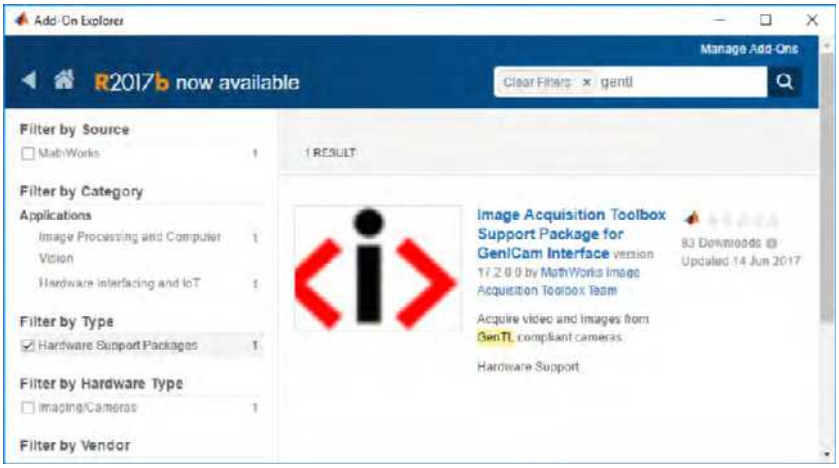
- On the **HOME** tab, click **Add-Ons** > **Get Hardware Support Packages**.
- In the MATLAB **Command Window**, enter:

```
supportPackageInstaller
```

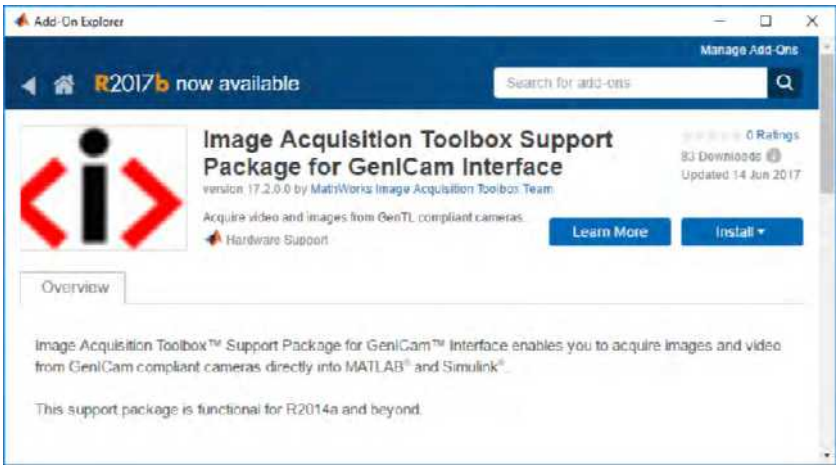
3. In the Add-On Explorer enter '**gentl**' in the search field.



4. Select the Image Acquisition Toolbox Support Package for GenICam Interface.



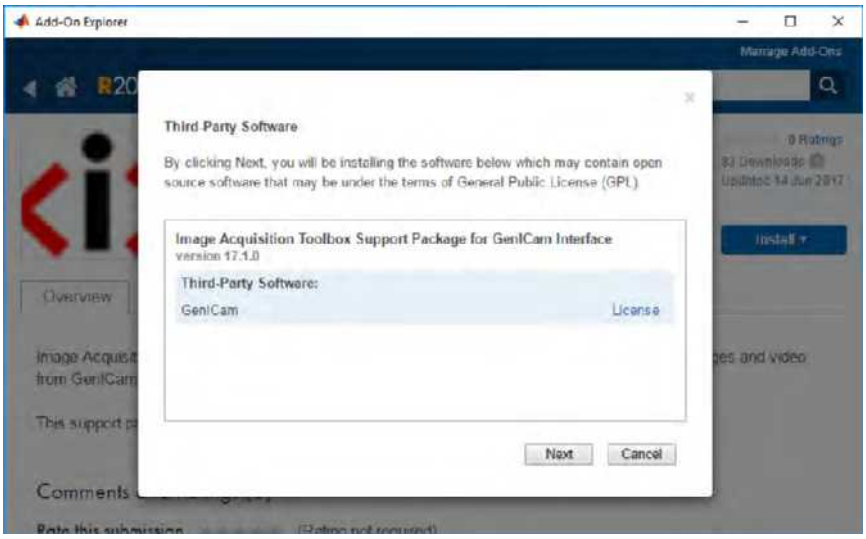
5. Click Install to install the Support Package for GenICam Interface.



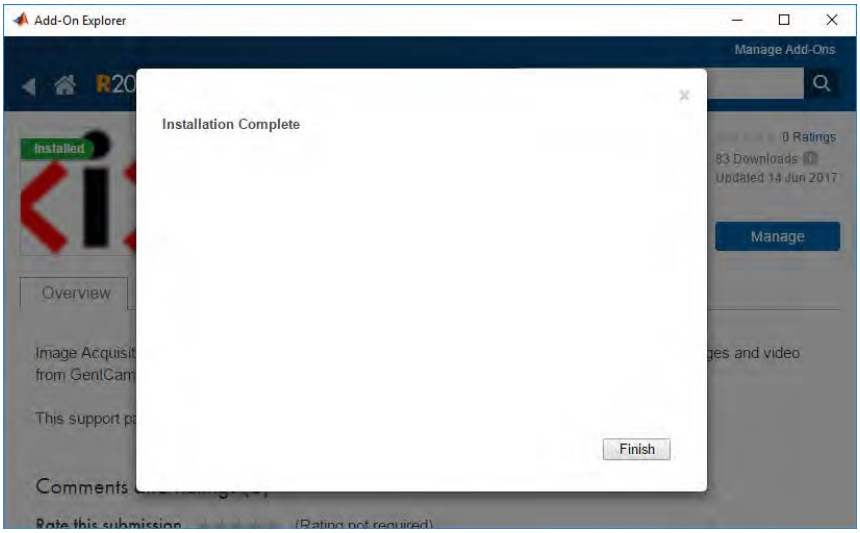
A window with licensing information will be displayed.

6. Accept the MathWorks Auxiliary Software License Agreement and the Genicam License Agreement.

7. Click **Next** to install the Support Package for GenICam Interface.



8. When the installation is complete, click **Finish**.



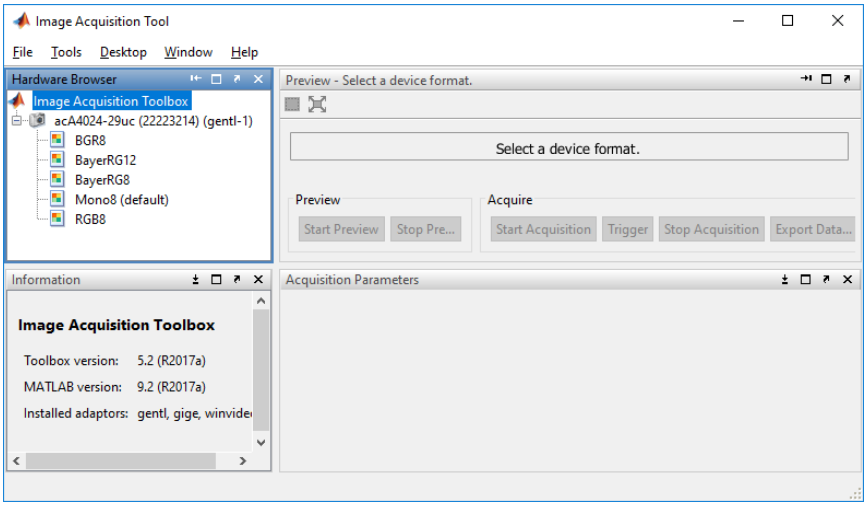
4 Operating a Basler USB3 Vision Camera with MATLAB

This section explains how to access a Basler ace USB 3.0 camera in MATLAB and how to configure the camera's features.

To access a Basler ace USB 3.0 camera in MATLAB:

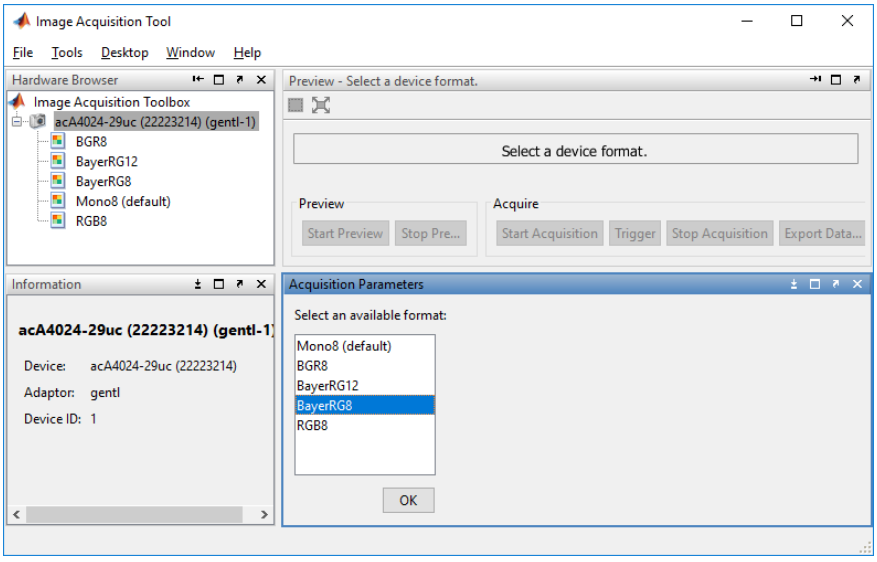
1. Connect your Basler ace USB 3.0 camera to a port of a USB 3.0 host controller card recommended by Basler.
2. In MATLAB, click **APPS** > **Image Acquisition**.

   The **Image Acquisition Tool** opens in a new window. In this example, a Basler ace acA4024-29uc camera is used. It is listed in the **Hardware Browser** pane.

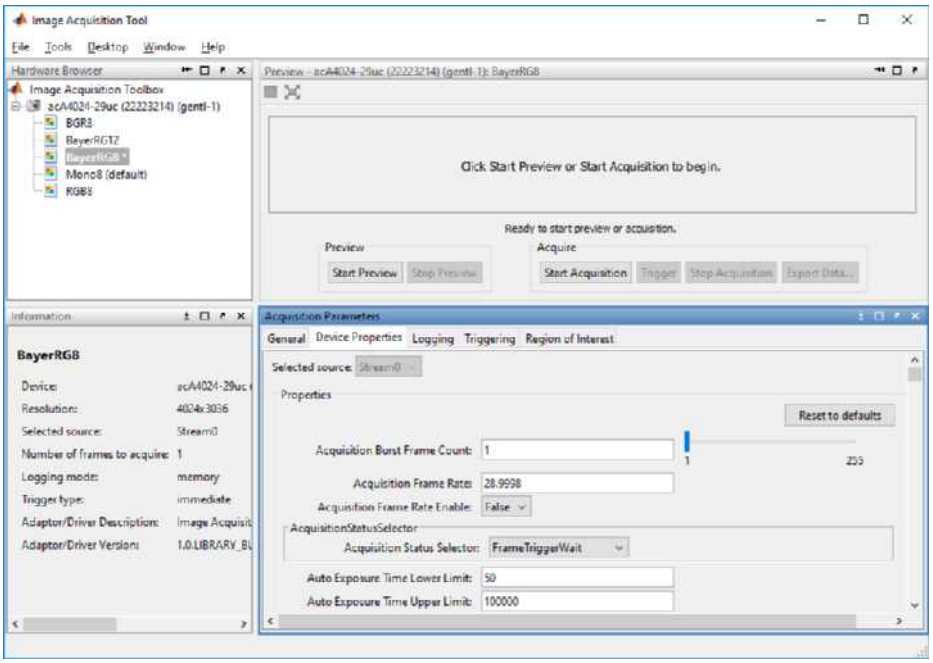


3. When you select the camera in the **Hardware Browser** pane, the available pixel formats will be listed in the **Acquisition Parameters** pane.
4. Select the desired pixel format and click **OK**.

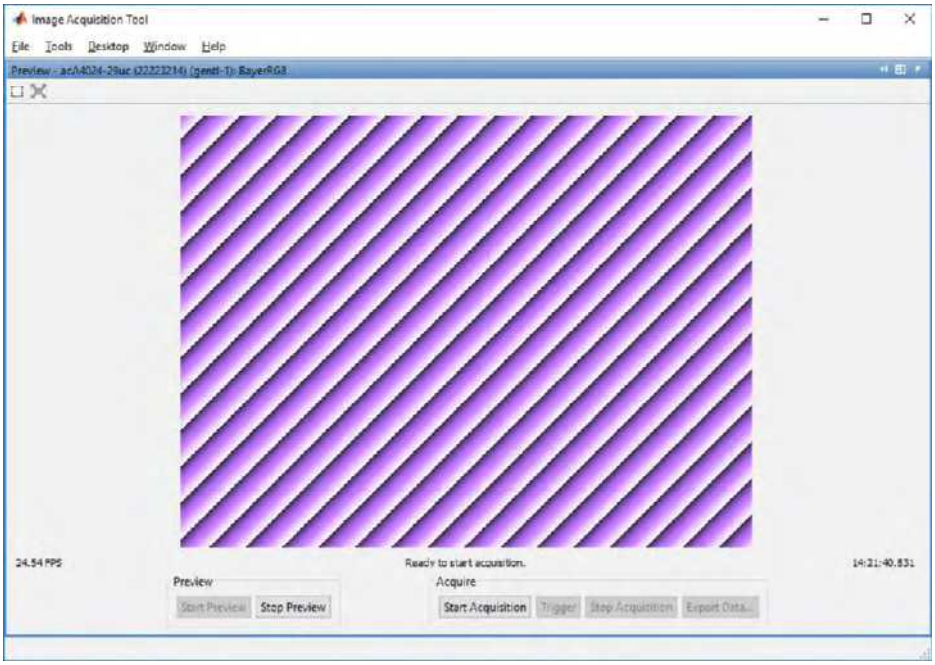
   The camera has been opened and is ready for use now.



5. In **Acquisition Parameters** pane, go to the **Device Properties** tab to access all currently supported camera features.



6. In the **Preview** pane, click **Start Preview** in order to get a live image from the camera.



5 Operating a Basler GigE Camera with MATLAB

This section explains how to access a Basler ace GigE Vision camera in MATLAB and how to configure the camera's features.

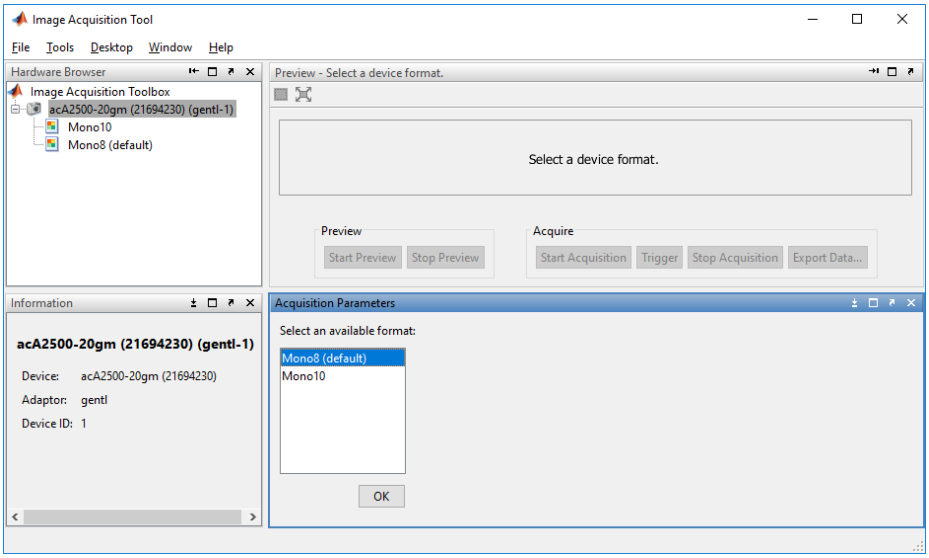
To access a Basler ace GigE camera in MATLAB:

1. Connect your Basler ace GigE camera to a GigE network card recommended by Basler.
2. In MATLAB, click **APPS** > **Image Acquisition**.

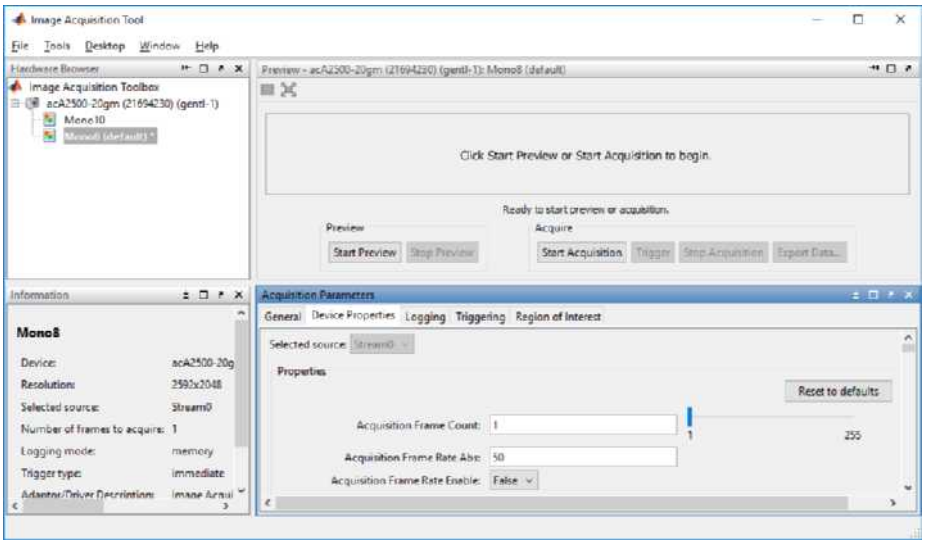
The **Image Acquisition Tool** opens in a new window. In this example, a Basler ace acA2500-20gm camera is used. It is listed in the **Hardware Browser** pane.

3. When you select the camera in the **Hardware Browser** pane, the available pixel formats will be listed in the **Acquisition Parameters** pane. Select the desired pixel format and click **OK**.

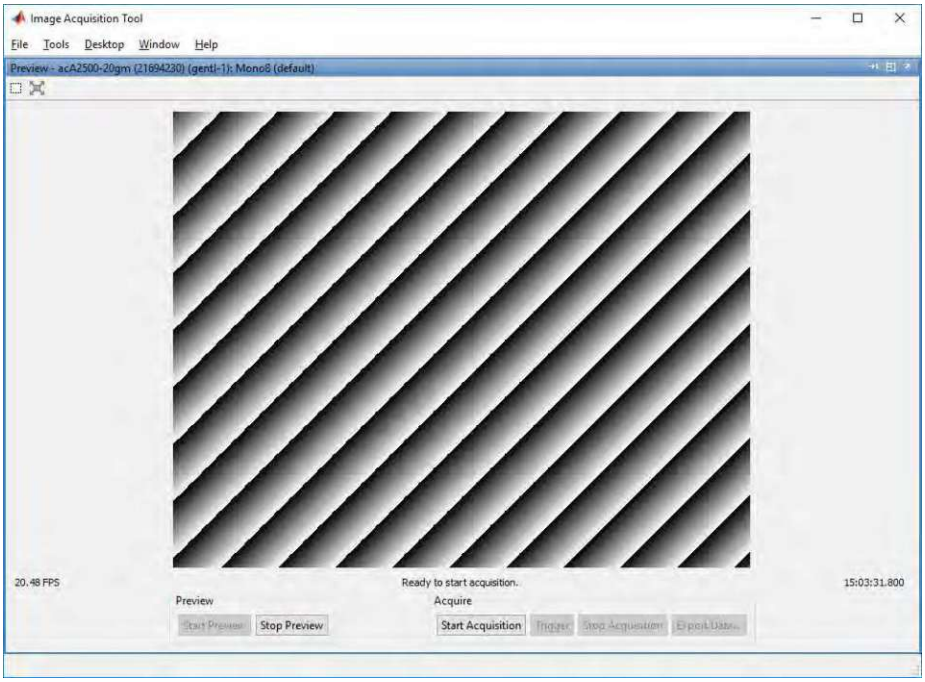
   The camera has been opened and is ready for use now.



4. In the **Acquisition Parameters** > **Device Properties** tab you can access all currently supported camera features:



5. In the **Preview** pane, click **Start Preview** in order to get a live image from the camera:



6 Troubleshooting

6.1 Troubleshooting USB 3.0 Cameras

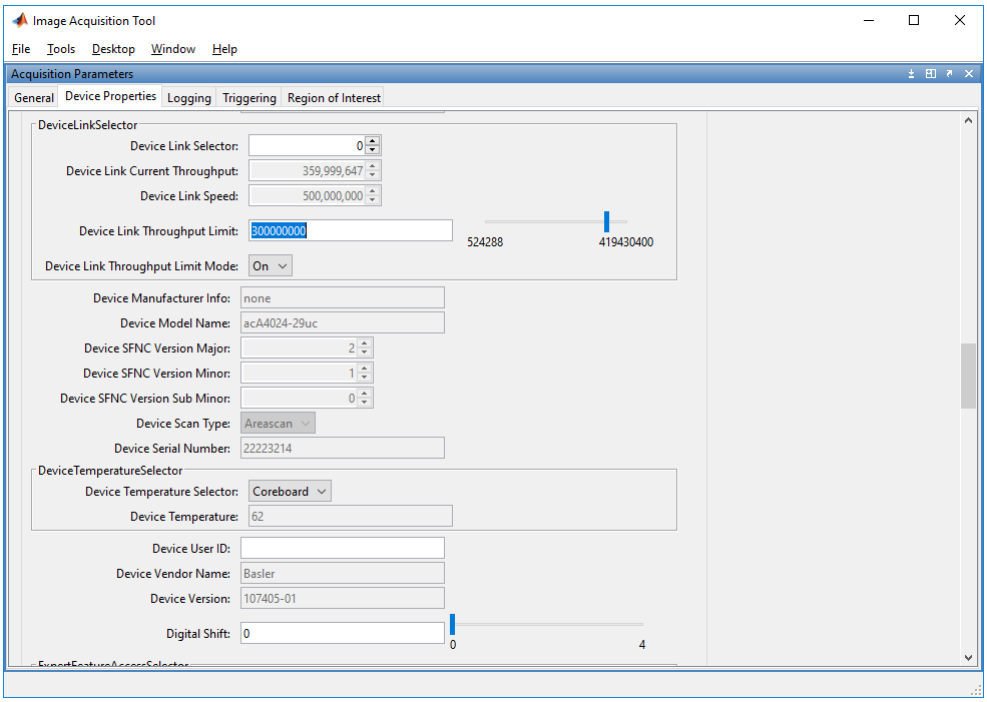
6.1.1 Troubleshooting Using MATLAB

If the images you acquire are corrupt, try decreasing the bandwidth currently used by the camera in the MATLAB Image Acquisition Tool. You can do this in two different ways:

- Image Acquisition Tool:

  In the Image Acquisition Tool, go to **Acquisition Parameters** > **Device Properties** and make the following changes:

  - Set Device Link Throughput Limit Mode to On.
  - **Device Link Throughput Limit**: Change the value (in Bytes/s) until MATLAB stops acquiring corrupt images, e.g., 300 000 000 Bytes (i.e., 300 Mbytes).



- Command Window:

  In MATLAB, click **HOME** > **Command Window** and enter the following:

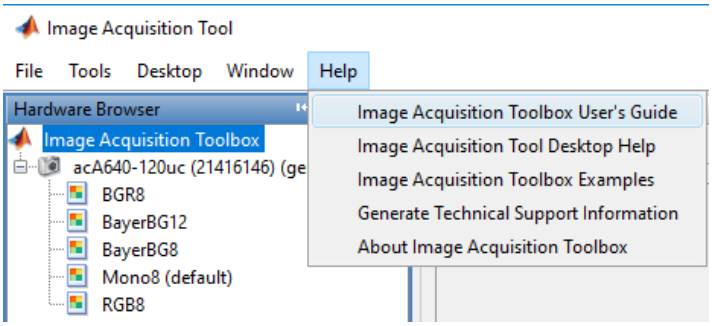
```
vid= videoinput('gentl', 1, 'BayerRG8')
src= getselectedsource(vid);
src.DeviceLinkThroughputLimitMode = 'On'
src.DeviceLinkThroughputLimit = 300000000
```

```
Command Window

    Sponsored Third Party Support License -- for use only to support products interfaced to
    MathWorks software under terms specified in your company's restricted use license agreement.

>> vid= videoinput('gentl', 1,'BayerRG8')

Summary of Video Input Object Using 'acA4024-29uc (22223214)'.

   Acquisition Source(s):  Stream0 is available.

  Acquisition Parameters:  'Stream0' is the current selected source.
                           10 frames per trigger using the selected source.
                           'BayerRG8' video data to be logged upon START.
                           Grabbing first of every 1 frame(s).
                           Log data to 'memory' on trigger.

      Trigger Parameters:  1 'immediate' trigger(s) on START.

                  Status:  Waiting for START.
                           0 frames acquired since starting.
                           0 frames available for GETDATA.

>> src= getselectedsource(vid);
>> src.DeviceLinkThroughputLimitMode = 'On'
```

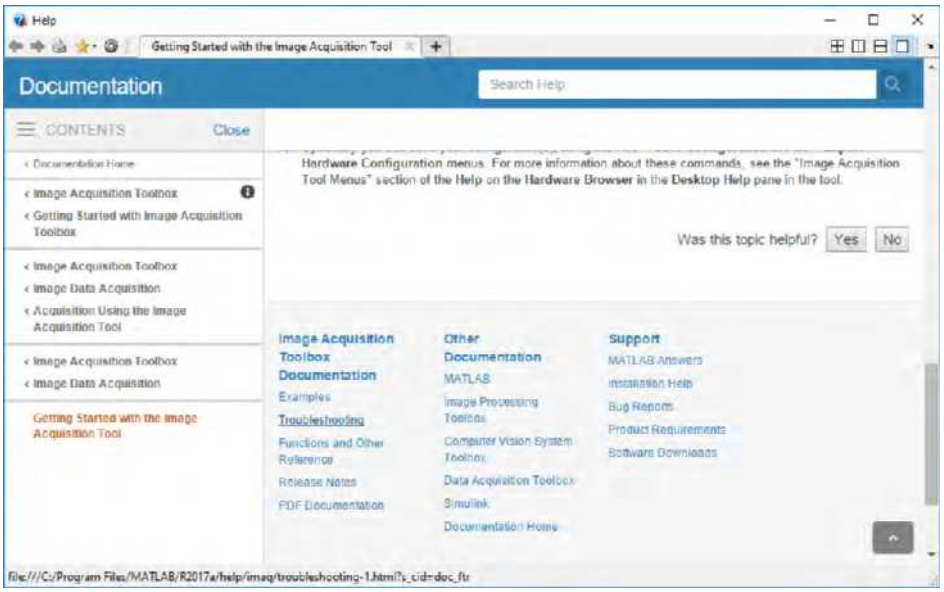
If decreasing the value of the **Device Link Throughput Limit** parameter doesn't improve the image acquisition issue, you may need further debugging with the Basler pylon Viewer, which is part of the Basler pylon Camera Software Suite.

For further MATLAB troubleshooting information, you have the following options:

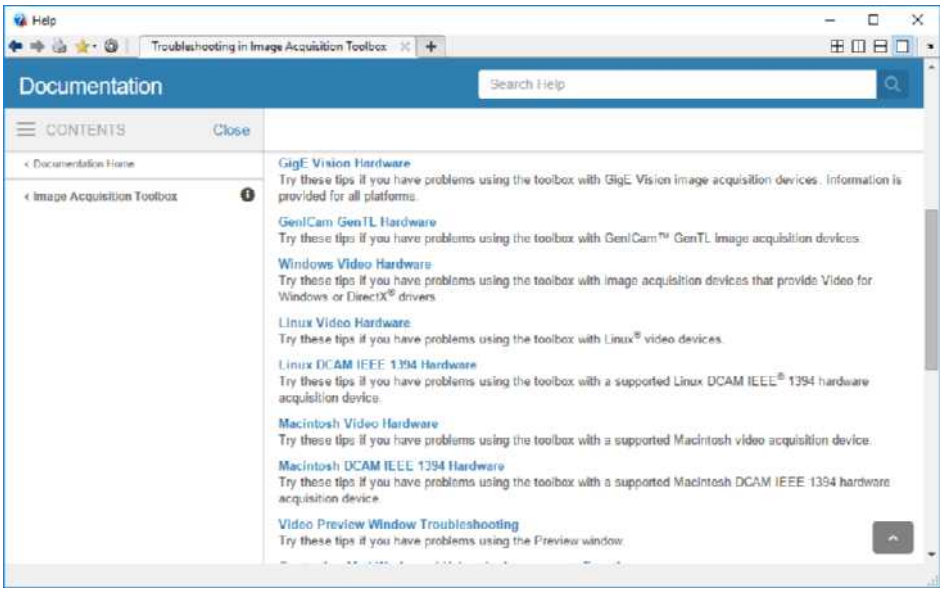
- Image Acquisition Toolbox User's Guide:
  1. In MATLAB, click **APPS** > **Image Acquisition** > **Help** > **Image Acquisition Toolbox User's Guide**.



  2. Scroll down and select **Troubleshooting**.



3. Here, you will find troubleshooting tips related to the GenICam GenTL Hardware:



- Contact the MATLAB technical support team.

6.1.2 Troubleshooting Using the Basler pylon Camera Software Suite

The Basler pylon Camera Software Suite offers the following tools: pylon Viewer, pylon USB Bandwidth Manager and pylon USB Configurator.

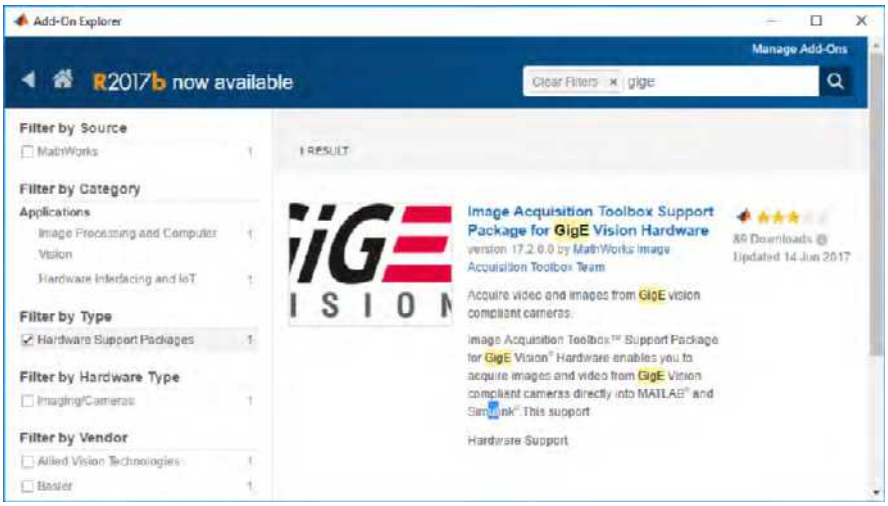
If you need help using these tools, contact your local Basler Customer Service Team (www.baslerweb.com).

6.2 Troubleshooting GigE Vision Cameras

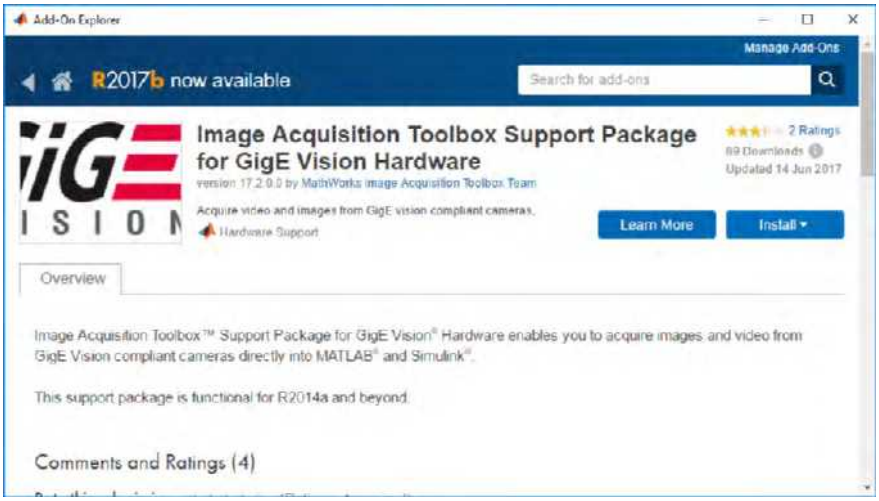
6.2.1 Troubleshooting Using MATLAB

If your GigE Vision camera is not shown in the MATLAB Image Acquisition Toolbox or the images you acquire are corrupt, you have to install the **Image Acquisition Toolbox Support Package for GigE Vision Hardware**. To do so, follow these steps:

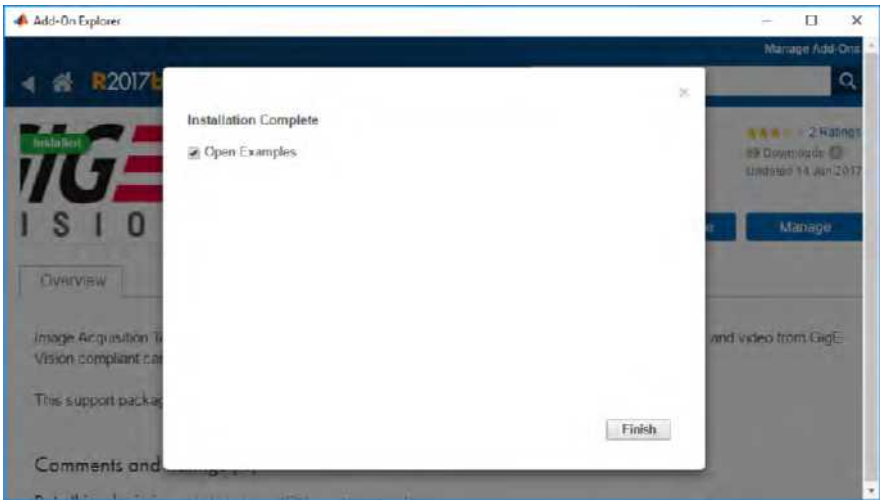
1. On the **HOME** tab, click **Add-Ons** > **Get Hardware Support Packages**.
2. In the Add-On Explorer enter '**gige**' in the search field:



3. Select the **Image Acquisition Toolbox Support Package for GigE Vision Hardware** and click **Install** to install it.

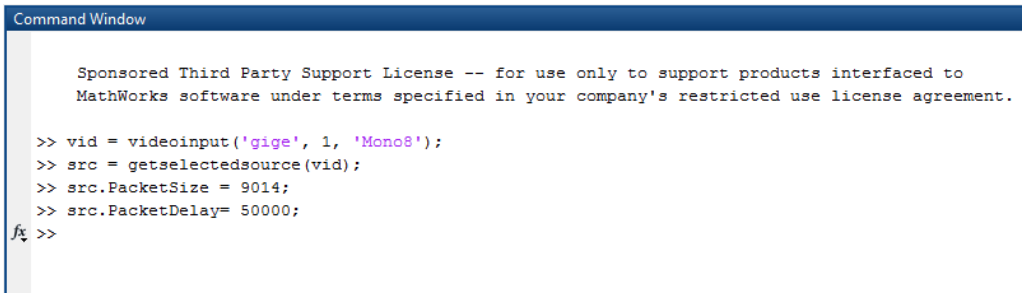


4. When the installation process has completed, click **Finish**.



5. In MATLAB, click **HOME** > **Command Window** enter the following to optimize the **Packet Size** and the **Packet Delay** parameters.

```
vid= videoinput('gige', 1, 'Mono8')
src= getselectedsource(vid);
src.PacketSize = 9014;
src.PacketDelay= 5000;
```



When using jumbo frames for the **Packet Size** camera parameter, i.e., values above 1500 Byte, make sure that your network adapter supports that value. Otherwise, image acquisition may fail.

Increasing the **Packet Delay** camera parameter, may reduce the camera's resulting maximum acquisition frame rate.

For more information about troubleshooting GigE Vision cameras in MATLAB, refer to the following documents:

- MATLAB GigE Vision Quick Start Configuration Guide: https://www.mathworks.com/matlabcentral/answers/uploaded_files/41167/GigEVisionQuickStart.pdf
- MATLAB FAQ: https://de.mathworks.com/matlabcentral/answers/91834-how-do-i-calculate-the-packet-delay-for-a-gige-vision-camera-to-prevent-dropped-frames

6.2.2 Troubleshooting Using the Basler pylon Camera Software Suite

The Basler pylon Camera Software Suite offers the following tools: pylon Viewer, pylon GigE Bandwidth Manager and pylon IP Configurator.

If you need help using these tools, contact your local Basler Customer Service Team (www.baslerweb.com).

Revision History

Document Number	Date	Changes
AW00134301000	4 Feb 2015	Initial release version of this document.
AW00134302000	6 Nov 2017	Updated the document to reflect the use of MATLAB R2017a. Updated the document to reflect the use of pylon 5.0.11. Added Chapter 5 to reflect the operation of GigE Vision cameras with MATLAB. Added Chapter 6.2 to reflect the troubleshooting of GigE Vision cameras.
AW00134303000	6 Nov 2017	Corrected a typo.

Anhang 5: Bedienungsanleitung

Schritte zum durchführen einer automatisierten Messung:

1. Im ”Hardware Setup” Tab verbinden und testen der Hardware
2. Referenzieren des Linearschlittens
3. Im ”Automated Measurement” Tab einstellen des Messgitters
4. Nach starten der automatisierten Messung dem Anweisungen des Programms folgen
5. Messdaten abspeichern (Messdaten werden automatisch im ”Configure Evaluation” Tab geladen)
6. Einstellen der Auswertungsparameter
7. Das Zentrum des Strahls mit dem Guess Beamarea Button markieren
8. Starten der Auswertung
9. Sollte die Auswertung abbrechen ist meistens die Hintergrundkorrektur der Grund
10. Ist die Auswertung abgeschlossen werden die ausgewerteten Messdaten automatisch in den ”Evaluation Result” Tab geladen

Schritte zum durchführen einer Livemessung:

1. Im ”Hardware Setup” Tab verbinden und testen der Hardware
2. Referenzieren des Linearschlittens
3. Über die Kontrollfelder im ”Hardware Setup” Tab Linearschlitten an gewünschte Position verfahren
4. Im ”Live Measurement” Tab ein Probebild des Strahlprofils machen (Trigger Image Button)
5. Das Zentrum des Strahls mit dem Guess Beamarea Button markieren
6. Einstellen der Auswertungsparameter
7. Hintergrund messen mit dem Measure Background Button (dafür muss der Laserstrahl abgeblockt werden)
8. Starten der Livemessung über den Toggle Button Start/Stop Live Measurement

Literatur

[1] Joachim Grehn, Gerd Harbeck, Peter Wessels. *PSSC Physik*. 2. Aufl. Friedr. Vieweg + Sohn . Braunschweig, 1974.

[2] Jürgen Eichler, Hans Joachim Eichler. *Laser, Bauformen,Strahlführung,Anwendungen*. 3. Aufl. Springer-Verlag, 1998.

[3] Chunlei Guo, Subhash Chandra Singh. *Handbook of Laser Technology and Applications*. 2. Aufl. CRC Press, 2021.

[4] Peter Adams. „Die Strahlqualität von Lasern“. In: *Laser Technik Journal* 1.2 (Okt. 2004), S. 63–66.

[5] Norman Hodgson, Horst Weber. *Laser Resonators and Beam Propagation*. 2. Aufl. Springer-Verlag, 2005.

[6] Claude Rullière. *Femtosecond Laser Pulses*. 2. Aufl. Springer Science+Business Media, Inc, 2003.

[7] *Pulse-Compression*. 2022. URL: http://www.swampoptics.com/pulse-compression.html. zugegriffen am: 23.02.2022.

[8] *Laser und Laseranlagen – Prüfverfahren für Laserstrahlabmessungen, Divergenzwinkel und Beugungsmaßzahlen – Teil 1: Stigmatische und einfach astigmatische Strahlen (ISO 11146-1:2005); Deutsche Fassung EN ISO 11146-1:2005*. Norm. Apr. 2005.

[9] *Laser und Laseranlagen - Prüfverfahren für Laserstrahlabmessungen, Divergenzwinkel und Beugungsmaßzahlen - Teil 3: Intrinsische und geometrische Laserstrahlklassifizierung und -propagation sowie Details der Prüfverfahren*. Norm. Feb. 2004.

[10] *Datenblatt Thorlabs PS908*. 2021. URL: https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=142. zugegriffen am: 20.10.2021.

[11] *Unbeschichtetes rechtwinkliges Prisma Thorlabs PS908*. 2021. URL: https://www.thorlabs.com/images/GuideImages/3765_PS908_SG.jpg. zugegriffen am: 12.10.2021.

[12] *Übertragungskurve von unbeschichteten BK7*. 2021. URL: https://www.thorlabs.com/images/TabImages/Uncoated_N-BK7_Transmission_780.gif. zugegriffen am: 12.10.2021.

[13] *Datenblatt ND-Filter (Thorlabs NE40B-B, NE50B-B)*. 2021. URL: https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=6273. zugegriffen am: 12.10.2021.

[14] *Bild ND-Filter (Thorlabs NE40B-B, NE50B-B)*. 2021. URL: https://www.thorlabs.com/images/large/24280-lrg.jpg. zugegriffen am: 12.10.2021.

[15] *Übertragungskurven für OD 2.0 - OD 6.0*. 2021. URL: https://www.thorlabs.com/images/TabImages/B_ND_Filters_Transmission_26_G1-350.gif. zugegriffen am: 12.10.2021.

[16] *Datenblatt Schrittmotor Sanyo Denki 103H5205-5240*. 2021. URL: https://docs.rs-online.com/9f7b/0900766b813b30f6.pdf. zugegriffen am: 20.10.2021.

[17] *Bild Schrittmotor Sanyo Denki 103H5205-5240*. 2021. URL: https://docs.rs-online.com/9f7b/0900766b813b30f6.pdf. zugegriffen am: 20.10.2021.

[18] *Momentenverlauf Sanyo Denki 103H5205-5240*. 2021. URL: https://docs.rs-online.com/9f7b/0900766b813b30f6.pdf. zugegriffen am: 20.10.2021.

[19] *Arduino UNO Rev3*. 2021. URL: https://store.arduino.cc/products/arduino-uno-rev3?selectedStore=eu. zugegriffen am: 20.10.2021.

[20] *Bild Arduino UNO Rev3*. 2021. URL: https://cdn.shopify.com/s/files/1/0506/1689/3647/products/A000079_03.front_710x467.jpg?v=1615307638. zugegriffen am: 20.10.2021.

[21] *Arduino UNO Rev3*. 2021. URL: https://store.arduino.cc/products/arduino-motor-shield-rev3?selectedStore=eu. zugegriffen am: 20.10.2021.

[22] *Bild Arduino UNO Rev3*. 2021. URL: https://cdn.shopify.com/s/files/1/0506/1689/3647/products/A000079_03.front_710x467.jpg?v=1615307638. zugegriffen am: 20.10.2021.

[23] *Datenblatt Basler ace GigE acA1300-60gmNIR*. 2021. URL: https://docs.baslerweb.com/aca1300-60gmnir. zugegriffen am: 20.10.2021.

[24] *Bild Basler ace GigE acA1300-60gmNIR*. 2021. URL: https://docs.baslerweb.com/images/image-ace-gige-mono.jpg. zugegriffen am: 12.10.2021.