

Beschreibung / Dokumentation

MSC I Process

# Inhaltsverzeichnis

1 Definition.....	3
2 Abstract.....	4
3 Abhängigkeiten.....	4
4 Installation.....	4
5 Fehlerbehandlung.....	4
6 Funktionsreferenz MSC Image.....	5
6.1 Struktur.....	5
6.2 Bildtypen.....	5
6.3 Masken.....	5
6.4 Memory Leaks.....	6
6.5 Funktionen.....	7
6.5.1 Grundlegendes.....	7
6.5.2 Interfaces.....	7
6.5.3 Operations.....	9
6.5.4 NUC.....	12
6.5.5 Calibrate Camera.....	13
6.5.6 Optical Measurement.....	18
6.5.7 ROI.....	19
6.5.8 Statistics.....	23
7 Funktionsreferenz MSC Video.....	24
7.1 Struktur.....	24
7.2 Funktionen.....	24
7.2.1 Grundlegendes.....	24
7.2.2 Operations.....	25
8 Funktionsreferenz MSC Video Writer.....	27
8.1 Funktionen.....	27
8.1.1 Grundlegendes.....	27
8.1.2 Operations.....	27
9 ETC.....	28
9.1 Conversion.....	28
9.2 Tools.....	28
10 Demoprogramme.....	30
10.1 MSC Image.....	30
10.2 MSC Video.....	31
11 Anhang.....	32
11.1 Entfernen der Verknüpfung zu IMAQ.....	32




# 1 Definition

## Begriffe:

Begriff	Erklärung
Bibliothek	Die Softwaresammlung dieses Projektes
Klasse	Ein strukturierter Datentyp mit Eigenschaften und Funktionen. Derzeit sind die Klassen „MSC Image“ und „MSC Video“ in dieser Bibliothek verfügbar.
VI	Bezeichnung für einen in LabVIEW realisierte Programmfunktion.
Referenz	„Link“ zu einem Speicherbereich, auf dem sich die eigentlichen Daten befinden.
Memory Leak	Werden nicht alle Referenzen korrekt geschlossen, so existieren die Daten weiterhin im Speicher, obwohl sie nicht mehr gebraucht werden. Dies führt in der Regel zu Programmabstürzen (z.B. out of memory error). Ein Memory Leak ist die Stelle im Programmcode, welcher diese Referenz verliert.

Tabelle 1: Begriffe

## Darstellung:

Anmerkung	
	> Enthält eine Textanmerkung welche zum weiterführenden Verständnis beitragen soll.
Hinweis	
	> Enthält einen Texthinweis, welcher zu beachten ist.
Literaturverweis	
	> Verweis zu weiterführender Literatur

## Programmcode, VI Namen:

Die in der Dokumentation beschriebenen VI's werden kursiv dargestellt. Die VIs können in der LabVIEW Programmumgebung unter dem hier angegebenen Namen gesucht werden.

## 2 Abstract

MSC I Process ermöglicht den einfachen und benutzerfreundlichen Einstieg in die Bildverarbeitung mit LabVIEW. Die Bibliothek basiert teilweise auf OpenCV, was den Zugang und die hohe Performanz einer Vielzahl von Funktionen ermöglicht.

## 3 Abhängigkeiten

Für die Verwendung der Bibliothek werden folgende Softwarekomponenten benötigt bzw. vorausgesetzt.

Benötigte Komponente	zu beziehen von/durch
LabVIEW 2014 oder höher	Installation der Programmierumgebung (National Instruments)
OpenCV 2.4.11	MSC oder von <a href="http://www.opencv.org">www.opencv.org</a> (Bestandteil dieser Bibliothek)
OpenG Toolkit	VI Package Manager
NI IMAQ / NI Vision	National Instruments (optional, nur für ROI Auswahl)
MSC User Library	MSC (Bestandteil dieser Bibliothek)

Tabelle 2: Abhängigkeiten

## 4 Installation

Kopieren Sie den Inhalt des Ordners „user.lib“ in die user.lib Ihrer LabVIEW Installation.  
(z.B. C:\Program Files (x86)\National Instruments\LabVIEW 2014\user.lib)

Beim nächsten Start der Entwicklungsumgebung befinden sich die Vis in der Funktionspalette.

## 5 Fehlerbehandlung

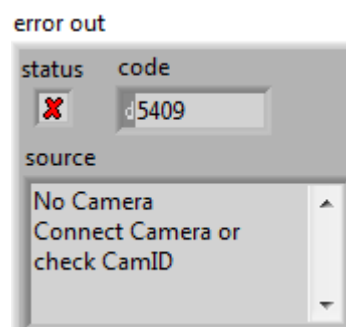


Abbildung 1: Errorcluster

Die hier verwendete Fehlerbehandlung orientiert sich am LabVIEW Standard. Alle Funktionen besitzen einen Fehlerclusterein- und -ausgang. Neben dem Status (Fehler / kein Fehler) gibt es einen

eindeutigen Fehlercode und Fehlerquelle oder Fehlerbeschreibung. Der oben aufgeführte Fehler tritt beispielsweise auf, wenn die ausgewählte Kamera nicht verbunden ist.

Die Fehlercodes reichen von 5401 bis 5499. Damit soll vermieden werden, dass Codes von anderen Programmteilen überschrieben werden. Tritt dennoch eine Überschneidung auf, kann eine Änderung im *VI Error Handling* vorgenommen werden.

## 6 Funktionsreferenz MSC Image

### 6.1 Struktur

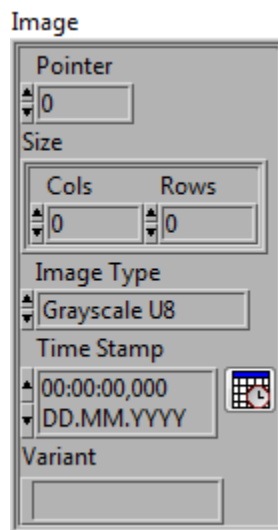


Abbildung 2: Struktur des MSC Image

Der Grundaufbau des MSC Images basiert auf Referenzen. Somit ist gewährleistet, dass sowohl der LabVIEW Code, als auch externer Code ohne unnötiges Kopieren mit den Daten arbeiten kann. Hierbei weist der Pointer direkt auf den Speicherbereich, welcher durch das Bild belegt ist. Anhand des Timestamps lässt sich erkennen, wann das Bild aufgenommen wurde. Da nicht abzusehen ist, welche weiteren Parameter im jeweiligen Projekt benötigt werden, wurde zusätzlich ein Variant mit angefügt. In diesem Platzhalter lassen sich beliebige Daten speichern, beispielsweise Kommentare oder Kameraparameter.

### 6.2 Bildtypen

Derzeit ist es möglich, Graustufenbilder in U8, U16, Single und Double zu erstellen und zu bearbeiten. Der einzige Farbtyp ist RGB mit drei U8 Kanälen.

### 6.3 Masken

Einige der hier verwendeten Funktionen benötigen eine Maske. Eine Maske markiert Bereiche im Bild, welche für die Operation verwendet werden sollen (z.B. *Inpaint* – Restauration von defekten Bildsegmenten). Diese Masken bestehen aus einem U8 Graustufenbild. Jeder Pixelwert ungleich Null gilt als markiert.

## 6.4 Memory Leaks

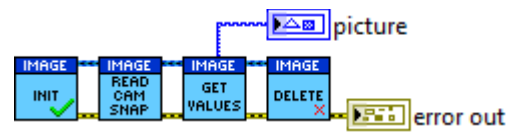


Abbildung 3: Camera Snapshot Demoprogramm

Löschen von Bildern, die nicht mehr gebraucht werden ist essenziell, um Programmastürze zu vermeiden. Vor Allem bei kontinuierlichen Aufnahmen füllt sich der Speicher rapide und der Benutzer bekommt einen „out of memory“ Fehler. Das VI *Delete* ist dafür zuständig, alle mit dem Bild in Verbindung stehenden Daten korrekt zu deallozieren, also quasi aufzuräumen. Logischerweise sind die Daten danach nicht mehr auslesbar.

Im Demoprogramm *Camera Snapshot* ist eine typische Anwendung veranschaulicht. Nach der Bildinitialisierung (*Initialize*) wird ein Frame der Kamera aufgenommen (*Read Camera Snapshot*), und an ein Picture Control ausgegeben (*Get Values*). Wird nach dieser Funktionskette nicht *Delete* ausgeführt, bleiben die Daten des Bildes (~ 1 bis 15 MB, je nach Kamera) im Speicher erhalten bis das komplette Programm beendet wurde.

## 6.5 Funktionen

### 6.5.1 Grundlegendes

#### ***Initialize***



Abbildung 4: Bildinhalt bei Initialisierung (NA.png)

Der Konstruktor eines Bildobjektes wird hier aufgerufen. Standardmäßig wird hier „NA.png“ in den Speicher geladen.

#### ***Delete***

Die Referenz wird zerstört. Außerdem wird der Speicherbereich freigegeben.

#### Hinweis



Wird diese Funktion nicht für jedes erstellte Bild aufgerufen, führt dies zu Memory Leaks.

#### ***Clone***

Kopiert alle Daten in ein neues Bildobjekt. Beide Objekte müssen gelöscht werden, um den Speicherbereich freizugeben (über *Delete*).

### 6.5.2 Interfaces

#### ***Read Camera Snapshot***

Simple Kameraaufnahmefunktion. Über die ID können unterschiedliche Kameras angesprochen werden. Unterstützt werden derzeit folgende Kameratreiber:

MIL, VFW, V4L, V4L2, Fireware, Firewire, IEEE 1394, DC, Stereo TYZX, Quicktime, Unicap, DirectShow, PVAPl, Prosilica GigE, OpenNI, XIMEA Camera

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/highgui/doc/reading\\_and\\_writing\\_images\\_and\\_video.html#video\\_capture](http://docs.opencv.org/modules/highgui/doc/reading_and_writing_images_and_video.html#video_capture)

#### Anmerkung



Die Klasse MSC Video bietet weitere Einstellungsmöglichkeiten, wie z.B. Belichtung und arbeitet deutlich performanter bei kontinuierlicher Aufnahme.

### Get / Set Values

Liest oder schreibt die aktuellen Werte als 2D LabVIEW Array oder Picture. Je nach verwendetem Bildtyp muss hier der korrekte Ausgang ausgewählt werden.

#### Anmerkung



Der RGB Datentyp gibt U32 Werte zurück. Um Kanäle zu extrahieren kann man *Color To RGB* verwenden.

#### Anmerkung



Ist die Aktualisierungsrate des Picture Controls sehr hoch, so empfiehlt es sich das Feature „Erase First“ abzuschalten. (Rechtsklick auf das Picture Control)

### Read / Write IMAQ Image

Wandelt das MSC Image in ein IMAQ Image und umgekehrt.

#### Hinweis



Die Daten werden doppelt abgelegt. Somit muss zusätzlich *IMAQ Dispose* aufgerufen werden, um den Speicher des IMAQ Bildes aufzuräumen.

#### Anmerkung



Der MSC Image ROI Datentyp enthält mehr Informationen als der Typ von IMAQ. Daher sollte die Konvertierung zu IMAQ nur zur Anzeige verwendet werden.

### Read Image

Liest ein gespeichertes Bild. Unterstützte Formate:

- Windows bitmaps: bmp, dib
- JPEG: jpeg, jpg, jpe
- JPEG 2000: jp2
- Portable Network Graphics: png
- Portable image format: pdm, pgm, ppm
- Sun rasters: sr, ras
- TIFF: tiff, tif

Über den Eingang „read image properties“ kann eine vorher abgespeicherte Eigenschaftsdatei auf das eingeladene Bild angewendet werden. (Siehe auch *Write Image*)



## Literaturverweis



OpenCV

[http://docs.opencv.org/modules/highgui/doc/reading\\_and\\_writing\\_images\\_and\\_video.html#imread](http://docs.opencv.org/modules/highgui/doc/reading_and_writing_images_and_video.html#imread)

**Write Image**

Speichert das Bild auf der Festplatte. Es werden die selben Formate wie bei *Read Image* unterstützt.

Zusätzlich kann PNG Komprimierung eingestellt werden. Der Wert 0 steht hierbei für kompressionslos, 9 ist maximale Kompression. Die JPEG Qualität kann von 0 bis 100 definiert werden.

PNG Kompression ist stets verlustfrei, wobei selbst eine JPEG Qualität von 100 bereits Verluste beinhaltet.

Bitte beachten Sie auch, dass nicht jedes Format in einen Dateityp gespeichert werden kann.

## Literaturverweis



OpenCV

[http://docs.opencv.org/modules/highgui/doc/reading\\_and\\_writing\\_images\\_and\\_video.html#imwrite](http://docs.opencv.org/modules/highgui/doc/reading_and_writing_images_and_video.html#imwrite)

Der aktivierte Eingang „write image properties“ erzeugt eine Eigenschaftsdatei, welche Größe, Bildtyp, Zeitstempel und Variant beinhaltet.

## Anmerkung



Die Pfadangabe „image properties path“ ist optional. Wird sie leergelassen, wird die Eigenschaftsdatei mit dem selben Namen der Bilddatei gespeichert.

**Get / Set Image**

Dies ist die Schnittstelle zu beliebiger anderer Software. Über den Pointer, der Bildgröße und den Bildtyp lässt sich das Bild performant austauschen.

**6.5.3 Operations****Arithmetics**

In diesem Ordner befinden sich alle einfachen arithmetischen Operationen für Bild – Bild, Bild – Skalar und Bild – Bild – Skalar. Derzeit ist es möglich zu addieren, multiplizieren, subtrahieren und dividieren.

**Type Cast**

Wandelt den Typ des Bildes um und skaliert optional die Werte.

**Resize**

Ändert die Größe eines Bildes und skaliert dessen Inhalt. Es sind unterschiedliche Interpolationsalgorithmen verfügbar.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/imgproc/doc/geometric\\_transformations.html#resize](http://docs.opencv.org/modules/imgproc/doc/geometric_transformations.html#resize)**Crop**

Schneidet ein Bild auf einen rechteckigen Bereich zu. Liegt dieser Bereich außerhalb des Bildes, so wird er angepasst.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/core/doc/basic\\_structures.html#Mat::Mat%28const%20Mat%26%20m,%20const%20Rect%26%20roi%29](http://docs.opencv.org/modules/core/doc/basic_structures.html#Mat::Mat%28const%20Mat%26%20m,%20const%20Rect%26%20roi%29)**Blur**

Verwischt das Bild über einen Box Filter mit definierter Größe. Je größer der Filterkern, desto stärker der Effekt.

**Literaturverweis**

OpenCV

<http://docs.opencv.org/modules/imgproc/doc/filtering.html#blur>**Canny**

Führt eine Canny Kantendetektion eines U8 Graustufenbildes durch. Vorgeschlagen wird, dass „Threshold 2“ dreimal größer ist als „Threshold 1“.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/imgproc/doc/feature\\_detection.html#canny](http://docs.opencv.org/modules/imgproc/doc/feature_detection.html#canny)

**Overlay Images**

Überlagert zwei Bilder mit einstellbarer Gewichtung. Die Gewichte alpha und beta können von Null bis Eins eingestellt werden.

$\text{image} = (\alpha * \text{image}) + (\beta * \text{image2})$

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/core/doc/operations\\_on\\_arrays.html#addweighted](http://docs.opencv.org/modules/core/doc/operations_on_arrays.html#addweighted)**Equalize Histogram**

Streckt das Histogramm, sodass das Spektrum ideal ausgenutzt wird.

**Invert**

Invertiert das Bild.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/core/doc/operations\\_on\\_arrays.html#invert](http://docs.opencv.org/modules/core/doc/operations_on_arrays.html#invert)**Gaussian Noise**

Füllt das Bild mit gaußschem Rauschen.

**Salt and Pepper Noise**

Füllt das Bild mit Salt and Pepper Noise.

**Dead Pixel Detection**

Erkennt tote Pixel des Bildes durch simple Grenzwertdetektion.

Das Eingabebild sollte eine Aufnahme mit maximaler Belichtung sein, sodass keine Falschdetektion auftritt.

Im Anschluss an die Erkennung können beispielsweise durch *Inpaint* die defekten Pixel ersetzt werden.

**Literaturverweis**

Wikipedia

[http://en.wikipedia.org/wiki/Defective\\_pixel](http://en.wikipedia.org/wiki/Defective_pixel)

**Inpaint**

Inpaint ist ein Verfahren zum Ersetzen von defekten oder gestörten Bildregionen. Anhand eines Radius werden Nachbarpixel ausgewählt und mithilfe eines Algorithmus zur Bildwiederherstellung genutzt.

**Literaturverweis**

OpenCV

<http://docs.opencv.org/modules/photo/doc/inpainting.html#inpaint>**Tint**

Gibt dem Bild einen deutlichen Farbstich einer definierten Farbe. Nützlich ist dies z.B. um Regionen eines Bildes für den Benutzer zu markieren.

**Valid Image**

Überprüft, ob das aktuelle Bild valide und integer ist. Es wird geraten vor Allem nach Pointeroperationen diese Funktion aufzurufen.

**6.5.4 NUC**

Die Non Uniformity Correction wird vor Allem bei Infrarotkameras angewendet.

**NUC Calibration**

In der hier verwendeten Zweipunktkalibrierung werden zwei Bilder benötigt – eines mit hoher und eines mit niedriger Temperatur. Die Temperatur sollte hierbei auf dem ganzen Bild konstant sein und einen idealen schwarzen Körper darstellen.

Abhängig von den Differenzen der Detektoren des Arrays wird eine 2D Tabelle mit linearen Kalibrierdaten gefüllt. Diese können in der *NUC* Funktion verwendet werden, um ein aufgenommenes Bild zu entstören.

Diese Kalibriertabelle besteht aus den Parametern  $b$  und  $m$  für die lineare Funktion  $y = m * x + b$  für jeden Pixelwert.

**Literaturverweis**

Non-uniformity Correction Group

[http://nuc.die.udec.cl/?page\\_id=18](http://nuc.die.udec.cl/?page_id=18)**NUC**

Die Kalibriertabelle von *NUC Calibration* wird hier benötigt, um das aktuelle Bild zu entstören.

### 6.5.5 Calibrate Camera

$$\begin{array}{ll}
 \text{radial} & \begin{aligned} x_{\text{corrected}} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{\text{corrected}} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \\
 \text{tangential} & \begin{aligned} x_{\text{corrected}} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{\text{corrected}} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \\
 \begin{array}{l} \text{unit conversion} \\ f - \text{focal length} \\ c - \text{optical centers} \end{array} & \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}
 \end{array}$$

Abbildung 5: Kalibrierungsfunktionen

Mithilfe einer Kamerakalibrierung lassen sich zwei Matrizen erzeugen, mit deren Hilfe man Verzerrungen aus Bildern entfernen kann.

Die hier entfernbaren Verzerrungen treten durch „Fischaugenlinsen“ und nicht-Parallelität der Kameralinse zum Kamerasensor auf.

Die Kalibrierung muss für jede Kamera nur einmalig durchgeführt werden, solange sich der Sensor und die Linse nicht ändert.

#### **Calibrate Camera / Undistort**

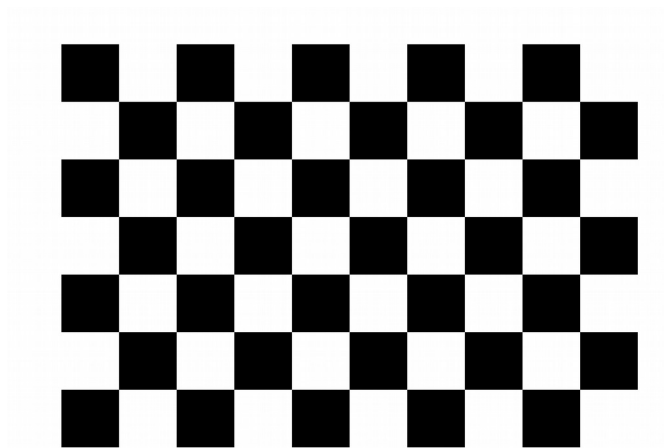


Abbildung 6: Testmuster Chessboard

Um die Kamerakalibrierung durchzuführen, ist es erforderlich ein Testmuster mehrere Male mit der Kamera aufzunehmen.

Das Testmuster sollte in ausreichender Größe auf eine planare Fläche aufgetragen werden. Ein handelsüblicher Drucker und das Befestigen auf einer Holzplatte ist in der Regel ausreichend dafür.

Nachdem die Kamerakalibrierung gestartet wurde, öffnet sich ein neues Fenster mit einem Livestream der Kamera. Theoretisch ist es möglich die Kalibrierung mit zwei Aufnahmen des Testmusters durchzuführen. Da in der Realität jedoch Störungen oder Bewegungsunschärfe die Qualität des Bildes reduzieren, wird eine Aufnahme von 20 Frames empfohlen.

Jede Aufnahme des Testmusters resultiert in einem Gleichungssatz. Um eine möglichst gute Kalibrierung zu erzielen, sollte das Testmuster durch das Bild bewegt werden, um unterschiedliche Winkel und Positionen, und somit auch unterschiedliche Gleichungen zu erhalten.

Die Kalibrierungsroutine hat vier Stufen:

1. Startposition auswählen  
Befindet sich das Testmuster in der gewünschten Position wird die Erfassung durch die „g“-Taste gestartet.
2. Erfassung wird durchgeführt  
Das Testmuster wird erfasst. Bitte bewegen Sie hierfür das Testmuster durch das Bild. In der unteren rechten Ecke ist der Fortschritt angezeigt.
3. Berechnung der Distortion Matrix  
Nach Erfassung aller Frames können diese ausgewertet werden. Eine höhere Frameanzahl resultiert in höherer Berechnungszeit.
4. Betrachtung des Ergebnisses  
Der Livestream zeigt nun das Ergebnis der Kalibrierung. Betätigt man die „u“-Taste, kann man zwischen originalem und entstörtem Livestream umschalten. Außerdem lässt sich die Erfassung durch „g“ neu starten.  
„Esc“ beendet die Kalibrierung, schließt den Stream und kehrt zu LabVIEW mit den ermittelten Matrizen zurück.

Im Stream wird das erkannte Testmuster farbig markiert. Erscheint keine Markierung, so wurde das Muster auch nicht erkannt. Jederzeit kann die Kalibrierung durch die Esc-Taste abgebrochen werden. Ist die Kalibrierung nicht abgeschlossen, so wird ein Error Code erzeugt.

Befinden Sie sich in der Erfassung, aber es werden keine Frames aufgezeichnet, so kann dies unterschiedliche Gründe haben:

- Falsches Testmuster  
Das Testmuster hat den falschen Typ, oder die Größe wurde nicht korrekt angegeben.
- Testmuster nicht oder nur teilweise im Bild  
Das Testmuster muss sich komplett im Bild befinden, um eine eindeutige Erfassung zu ermöglichen.
- Testmuster nicht erkannt  
Die häufigsten Gründe hierfür sind Bewegungsunschärfe und verschmutzte Kameralinsen.

Zudem gibt es weitere Einstellungsmöglichkeiten zur Routine:

- „Tangent Distortion“  
Gibt an, ob der tangentielle Teil der Verzerrungsmatrix verwendet wird, oder nicht.
- „Fixed Aspect Ratio“  
Wenn die Brennweiten  $f_x$  und  $f_y$  identisch sind, so sollte man diesen Wert auf True setzen.
- „Fixed Principal Point“  
Definiert, ob der Hauptpunkt fest ist, oder durch den Algorithmus verändert werden darf.
- „Skip Keys“  
Überspringt die Tastatureingaben.
- „Frames“  
Anzahl der nötigen Frames zur Kalibrierung
- „Frameskip“  
Zeitverzögerung in ms, die mindestens zwischen zwei Frames liegen muss. Dies gibt dem Benutzer Zeit das Testmuster neu zu positionieren.

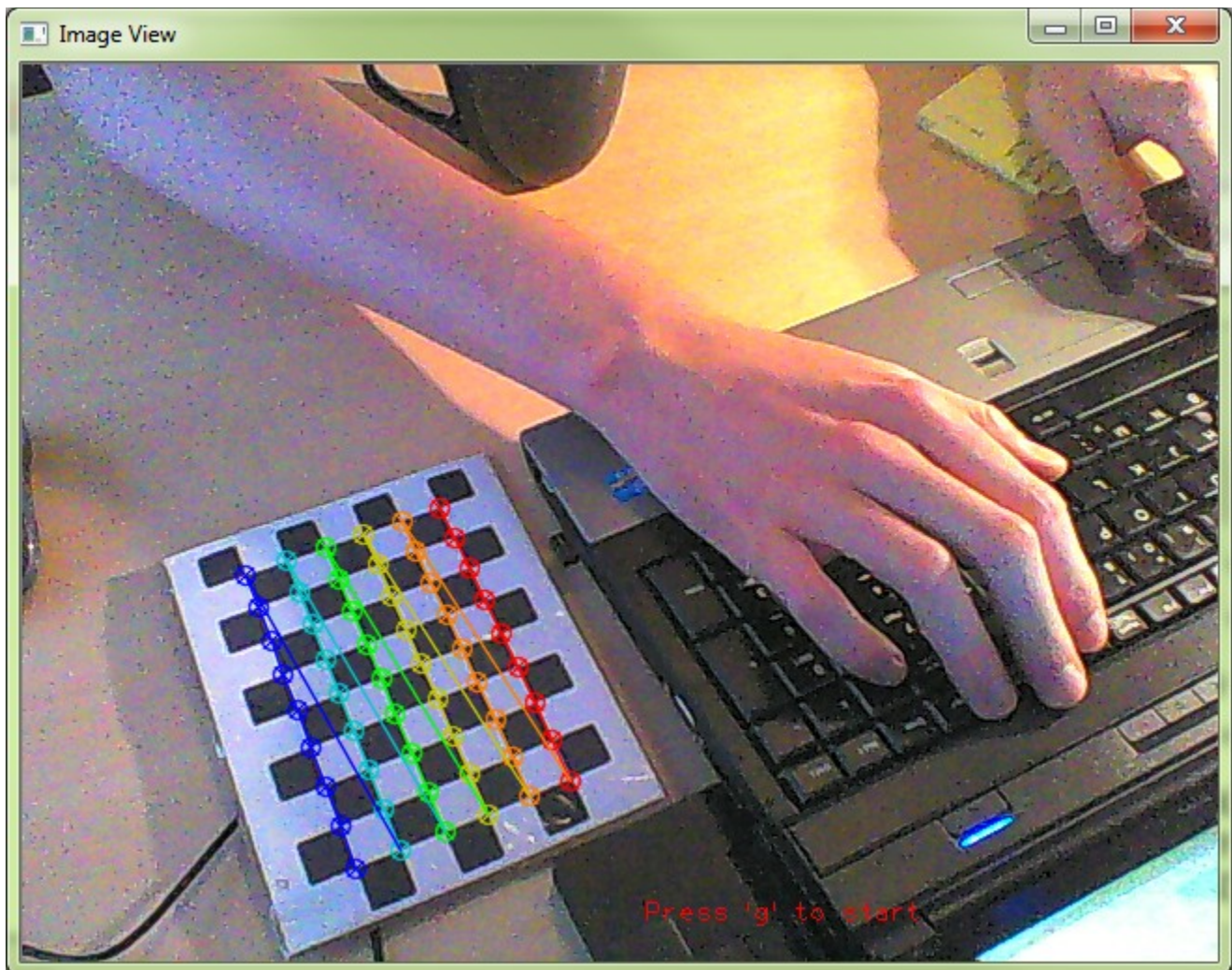


Abbildung 7: Kamerakalibrierung mit farbig markiertem Testmuster





Abbildung 8: Fischaugeneffekt

Um weitere Informationen zu diesem Vorgang zu erhalten, besuchen Sie bitte den Weblink:

#### Literaturverweis



OpenCV

[http://docs.opencv.org/doc/tutorials/calib3d/camera\\_calibration/camera\\_calibration.html](http://docs.opencv.org/doc/tutorials/calib3d/camera_calibration/camera_calibration.html)

### 6.5.6 Optical Measurement

Anhand des mathematischen Modells der dünnen Linse und der Messung der Entfernung der Kamera kann ein Pixelwert in reale Größen umgerechnet werden. Doch zunächst muss die Image Distance berechnet werden:

#### ***Estimate Image Distance***

Bestimmt die Image Distance einer Kamera.

Zunächst wird ein Objekt mit bekannter Größe und bekanntem Abstand zur Kamera parallel zum Kamerasensor platziert. Nachdem man im Bild das Objekt mit zwei Punkten markiert hat, kann die Image Distance ermittelt werden.

Zusätzlich kann, sofern es bekannt ist, die Größe eines Pixels auf dem Sensor mit angegeben werden. Auf diese Art ist es auch möglich, nicht-quadratische Pixelkameras zur Vermessung zu verwenden.

#### ***Optical Line Measurement***

Mit der Information der Image Distance und der Objektentfernung kann durch Markierung des Objektes im Bild dessen Größe ermittelt werden.

Auch hier ist es möglich die Pixelgröße auf dem Sensor einzustellen.

#### ***Extract Line***

Extrahiert ein Linienprofil.

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/core/doc/drawing\\_functions.html#lineiterator](http://docs.opencv.org/modules/core/doc/drawing_functions.html#lineiterator)

### 6.5.7 ROI

ROI

Type  
Point

Parameters  
0 0

Contour  
0  
x 0  
y 0

Bounding Box  
0  
x 0  
y 0

Contour Length ROI Mean ROI Min  
0 0 0

ROI Area ROI StdDev ROI Max  
0 0 0

Hierarchy  
parent ID  
0  
children IDs  
0 0  
level  
0

Variant

Timestamp  
00:00:00,000  
DD.MM.YYYY

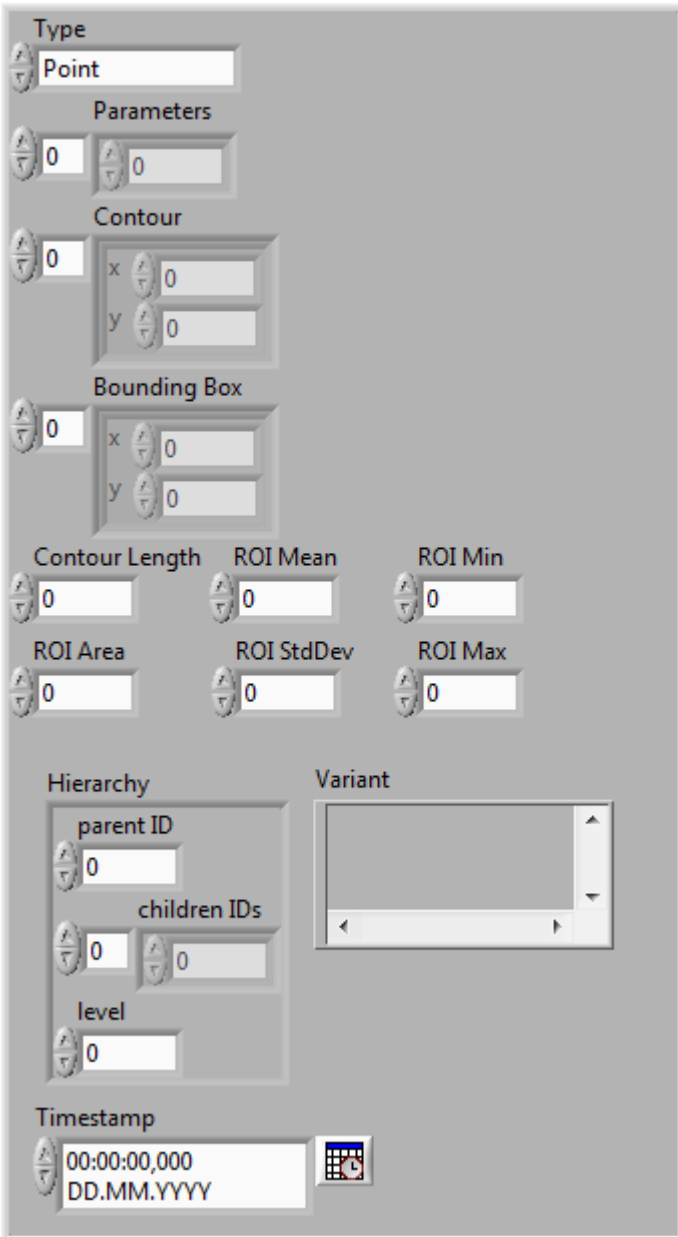


Abbildung 9: Struktur der ROI

Im **Region Of Interest Control** befinden sich alle nötigen Informationen zur beschreibenden Region im Bild.

- „Type“  
Der Typ der Region. Derzeit ist es möglich Punkt, Linie, Rechteck, Oval und Freihand / Polygon zu wählen.
- „Parameters“  
Dies sind die Parameter der IMAQ ROI. Sie werden zur Kompatibilität beibehalten
- „Contour“  
Die umschließende Kontur der Region
- „Bounding Box“  
Umschließendes Rechteck
- „Contour Length“  
Länge der Kontur in Pixeln
- „ROI Area“  
Fläche der Region in Pixel<sup>2</sup>
- „Statistische Werte“  
Mittelwert, Standardabweichung, Minimum und Maximum der untersuchten Region
- „Hierarchy“  
Hierarchie der aktuellen Region zu anderen Regionen
- „Variant“  
Alle zusätzlichen Informationen zur ROI können hier gespeichert werden.
- „Timestamp“  
Zeitstempel der Region

## **ROI**

Die eingegebenen ROIs werden analysiert und berechnet.

Ist „Crop“ auf True, so wird das Ergebnis auf die ausgewählte Region zugeschnitten.

Zusätzlich ist es möglich, eine ID mit anzugeben. Es wird dann lediglich die ROI mit deren Index analysiert. Sinnvoll ist dies beispielsweise mit der Verwendung von *ROI – Fill Tree*, was die ID der ausgewählten ROI zurückgibt.

Sind hierarchische Daten vorhanden, so wird von jedem ROI seine Kind-ROIs abgezogen.

Über *IMAQ ROI to MSC Image ROI* kann ein IMAQ ROI in ein MSC Image ROI umgewandelt werden. So kann man sich die benutzerfreundliche Oberfläche von IMAQ zu Nutze machen.

### Hinweis



Bitte vergessen Sie nicht, auch die hier erstellten Bilder zu dereferenzieren.

### **Apply Mask**

Extrahiert Bilddaten, welche von einem Maskenbild (U8) markiert sind. Alle Pixel ungleich Null werden als markiert betrachtet.

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/core/doc/basic\\_structures.html#void%20Mat::copyTo%28OutputArray%20m,%20InputArray%20mask%29%20const](http://docs.opencv.org/modules/core/doc/basic_structures.html#void%20Mat::copyTo%28OutputArray%20m,%20InputArray%20mask%29%20const)

### **Calculate Contours**

Auf dem Maskenbild (U8) werden Konturen erkannt und als ROIs zurückgegeben. Außerdem wird bei dieser automatischen Bestimmung die Hierarchie beachtet. Somit ist es möglich festzustellen, welche ROI andere ROIs beinhaltet. Die Darstellung über *ROI – Fill Tree* stellt dies über eine Baumansicht dar.

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/imgproc/doc/structural\\_analysis\\_and\\_shape\\_descriptors.html#findContours](http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#findContours)

#### Hinweis



Die Hierarchie arbeitet indexgesteuert. Somit darf die Indizierung des ROI Arrays nicht geändert werden (z.B. Elemente entfernen oder neu ordnen).

### **Threshold**

Führt eine Grenzwertbetrachtung durch und extrahiert die entsprechenden Bildteile

Ist  $Low < High$ , werden Pixel zwischen beiden Grenzen extrahiert.

Ist  $Low > High$ , werden die Pixel betrachtet, die von den Grenzen ausgeschlossen werden.

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/core/doc/operations\\_on\\_arrays.html#inrange](http://docs.opencv.org/modules/core/doc/operations_on_arrays.html#inrange)

**Mask From Threshold**

Erstellt eine Maske anhand von Grenzwerten

Ist  $Low < High$ , werden Pixel zwischen beiden Grenzen extrahiert.

Ist  $Low > High$ , werden die Pixel betrachtet, die von den Grenzen ausgeschlossen werden.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/core/doc/operations\\_on\\_arrays.html#inrange](http://docs.opencv.org/modules/core/doc/operations_on_arrays.html#inrange)

**Mask From Contour**

Erstellt ein Maskenbild (U8) anhand einer Kontur. Alle Pixel innerhalb dieser Kontur werden auf 255 gesetzt.

**Literaturverweis**

OpenCV

[http://docs.opencv.org/modules/imgproc/doc/structural\\_analysis\\_and\\_shape\\_descriptors.html#drawcontours](http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#drawcontours)

### 6.5.8 Statistics

#### ***Get Type***

Liest den Bildtyp aus.

#### ***Get Size***

Liest die Bildgröße aus.

#### ***Get / Set Timestamp***

Liest bzw. schreibt den Timestamp. Wird bei *Set Timestamp* der timestamp nicht verbunden, so wird die aktuelle Systemzeit verwendet.

#### ***Get / Set Variant***

Diese Vis sind der Zugang zu der Variant Funktionalität.

#### ***Histogram***

Berechnet ein Histogramm des aktuellen Bildes für jeden Kanal.

#### Hinweis



Der Bildtyp Double (F64) kann hier zu einem Overflow führen.

#### ***Statistics***

Berechnet statistische Werte eines Bildes mit optionaler Maske. Dieses VI wird beispielsweise zur ROI Berechnung verwendet.

#### Hinweis



Der Bildtyp Double (F64) kann hier zu einem Overflow führen.

## 7 Funktionsreferenz MSC Video

Die Funktion *Read Camera Snapshot* der MSC Image Klasse bietet eine schnell implementierte Kameraaufnahme. Allerdings stößt man hier schnell auf Grenzen: Bei jedem Funktionsaufruf wird der Kamerastream geöffnet, ein Bild extrahiert und der Kamerastream geschlossen. Kontinuierliche Aufnahmen sind so nicht sinnvoll lösbar. Auch ist es nicht möglich manuell einen Kamertreiber zu wählen oder Kameraeinstellungen zu ändern. Abhilfe schafft die MSC Video Klasse.

### 7.1 Struktur

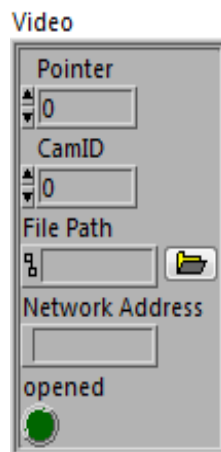


Abbildung 10: Struktur des MSC Videos

Möchte man einen Videostream starten, so benötigt man lediglich die Kamera ID. Der gestartete Stream wird danach über einen Pointer referenziert und lässt sich so auch aus anderer Software ansprechen. Das boolean Element *opened* gibt an, ob der Stream gestartet ist, oder nicht.

### 7.2 Funktionen

#### 7.2.1 Grundlegendes

##### **Initialize**

Initialisiert eine Instanz für weitere Operationen.

##### **Delete**

Schließt den Stream und alle Referenzen zur Videoinstanz.



### 7.2.2 Operations

#### ***Open Camera Stream***

Startet einen Stream zur Kamera mit der ID „camID“.

Es wird automatisch ein Treiber ausgewählt. Sind mehrere verfügbar, so kann der gewünschte über „API“ ausgewählt werden.

#### Literaturverweis



OpenCV

[http://docs.opencv.org/modules/highgui/doc/reading\\_and\\_writing\\_images\\_and\\_video.html#video-capture-videocapture](http://docs.opencv.org/modules/highgui/doc/reading_and_writing_images_and_video.html#video-capture-videocapture)

#### ***Open File Stream***

Startet einen Stream zu einer gespeicherten Videodatei.

#### ***Open Network Stream***

Startet einen Stream zu einer Netzwerkressource. Viele Netzwerkkameras verwenden einen h264 Codec. Um diese ansprechen zu können ist es erforderlich die Datei „opencv\_ffmpeg\*.dll“ entweder in das Applikationsverzeichnis oder das Systemverzeichnis (system32, syswow64) zu kopieren.

#### ***Close Stream***

Schließt den vorher geöffneten Stream.

#### ***Read Image***

Liest ein Bild aus einem geöffneten Stream und gibt es als MSC Image zurück.

#### ***Get / Set Property***

Setzt die Eigenschaft eines geöffneten Kamerastreams. Je nach Kameratyp, Treiber und Typ des Streams sind hier unterschiedliche Eigenschaften setzbar.

**Properties**

Frame Width	640
Frame Height	480
FPS	0
Brightness	0
Contrast	0
Saturation	64
Hue	0
Exposure	-3
White Balance U	2800
Rectification	-1
Gamma	100
Temperature	-1
White Balance V	-1
ISO Speed	-1
Backlight	1
Buffersize	-1

Abbildung 11: Einige verfügbaren Kameraeinstellungen

### ***Get All properties***

Liest alle Parameter aus und gibt sie als 2D Stringarray aus.

### ***Set Resolution***

Setzt die Größe des Videostreams (benutzerfreundlicher als über *Get Property*).

## 8 Funktionsreferenz MSC Video Writer

Um eine Bilderfolge als Video abzuspeichern, kann man sich dem MSC Video Writer bedienen.

### 8.1 Funktionen

#### 8.1.1 Grundlegendes

##### ***Initialize***

Initialisiert das Writer Objekt. Bereits hier muss die Framerate (FPS), Pfad, Größe und Typ ausgewählt werden, um die Datei auf Ihren Inhalt vorzubereiten. Über den FourCC Codec kann eine Komprimierung festgelegt werden. Ist hier „0“ angegeben ist das Video unkomprimiert, eine „-1“ öffnet einen Dialog zur Codecauswahl.

##### Literaturverweis



FourCC Codecs

<http://www.fourcc.org/codecs.php>

##### ***Delete***

*Delete* beendet die Video Writer Operation und löscht offene Referenzen. Diese Funktion löscht nicht die Datei auf der Festplatte.

#### 8.1.2 Operations

##### ***Get Size / Type***

Liest die angegebene Information zurück.

##### ***Write***

Fügt der Videodatei einen Frame hinzu. Der Frame ist im Format des MSC Picture zu übergeben.

## 9 ETC

Einige Funktionen haben nicht direkt etwas mit Bild- oder Videoobjekten zu tun, sind aber hilfreich bei der Programmierung von Bildverarbeitungssoftware.

### 9.1 Conversion

Für die Umrechnung oder Konvertierung verschiedenster Daten findet sich in diesem Ordner eine kleine Auswahl.

Für Sie als Endnutzer sind vermutlich nur zwei Vis dieses Ordners nützlich: *IMAQ ROI to MSC Image ROI* und *MSC Image ROI to IMAQ ROI*. Hier werden die ROIs für die Verwendung umgewandelt.

#### Anmerkung



Der MSC Image ROI Datentyp enthält mehr Informationen als der Typ von IMAQ. Daher sollte die Konvertierung zu IMAQ nur zur Anzeige verwendet werden.

### 9.2 Tools

#### Copyright

Gibt den Copyright Vermerk der .dll Datei zurück.

Bitte beachten Sie, dass sowohl die Programmerweiterung (.dll), als auch die LabVIEW Bibliothek und Klassen geschützt sind. Eine Weitergabe oder Vervielfältigung benötigt schriftliches Einverständnis der Firma MSC.

#### Ellipse Contour Points

Berechnet die Kontur einer Ellipse.

#### Error Handling

Hier finden sich Informationen zu den Error Codes. Dieses VI wird automatisch ausgeführt und kann als Nachschlagewerk für mögliche Fehler genutzt werden.

#### ROI – Fill Tree

Befüllt ein Tree Control mit ROI Daten. Die Wahl fiel auf das Tree Control, da hier die hierarchischen Beziehungen übersichtlich dargestellt werden können.

Zudem gibt dieses VI den Index der ausgewählten ROI zurück, um beispielsweise die Anzeige nur auf diese ROI anzupassen (z.B. in Kombination mit *ROI*).

#### ROI Max Bounding Box

Berechnet die gemeinsame Bounding Box zweier Rechtecke.

#### Scale Intensity Graph

Skaliert einen Intensity Graphen anhand des vorliegenden Bildtyps.

***Spread Histogram***

Spreizt die Werte des Input Arrays um die Grenzen Min und Max ideal auszufüllen.

***U8 Grayscale Color***

Berechnet die RGB Wertpalette eines U8 Graustufenbildes.

***Uneven***

Wandelt die Zahl in eine ungerade Zahl um.

***Zoom Picture***

Setzt die Zoom Parameter eines LabVIEW Picture Controls so, dass das derzeitige Bild maximiert und ohne Verzerrung dargestellt wird.

## 10 Demoprogramme

Diese Programme demonstrieren den Umgang mit verschiedenen Funktionen. Als Einstieg in diese Bibliothek sind diese ideal. Bitte schlagen Sie in der Funktionsreferenz nach, falls Fachbegriffe oder Funktionen unklar sind.

### 10.1 MSC Image

#### **Camera Snapshot**

Die einfachste Demonstration dieser Bibliothek liest ein Bild einer angeschlossenen Kamera und zeigt dieses an.

#### **Read Image**

Liest ein Bild von der Festplatte und zeigt es an.

#### **ROI**

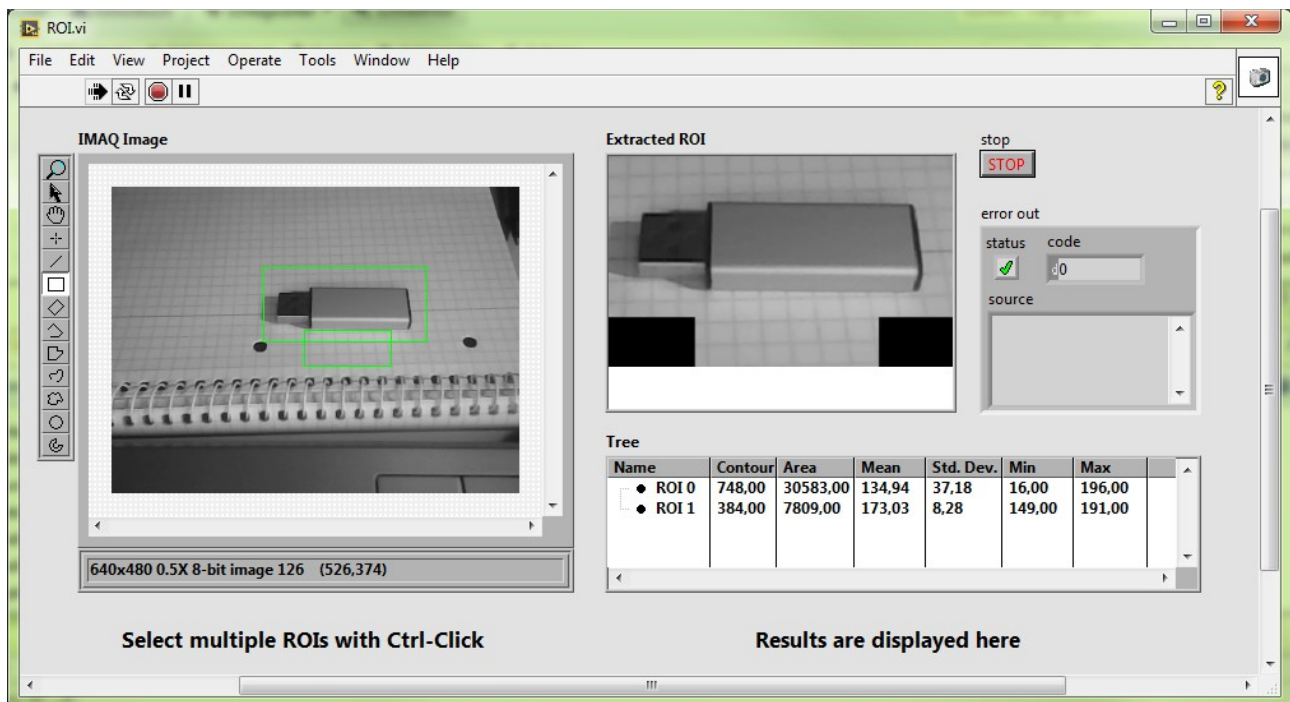


Abbildung 12: Demo ROI

Dieses fortgeschrittene Beispiel beschreibt die Verwendung der ROI Tools. Die auf der linken Seite gewählten ROIs werden auf der rechten Seite analysiert.

## 10.2 MSC Video

### Camera Stream

In dieser Demo wird ein Livestream geöffnet und kontinuierlich Bilder extrahiert.

### Camera Stream Properties

Der Livestream kann über Properties eingestellt werden.

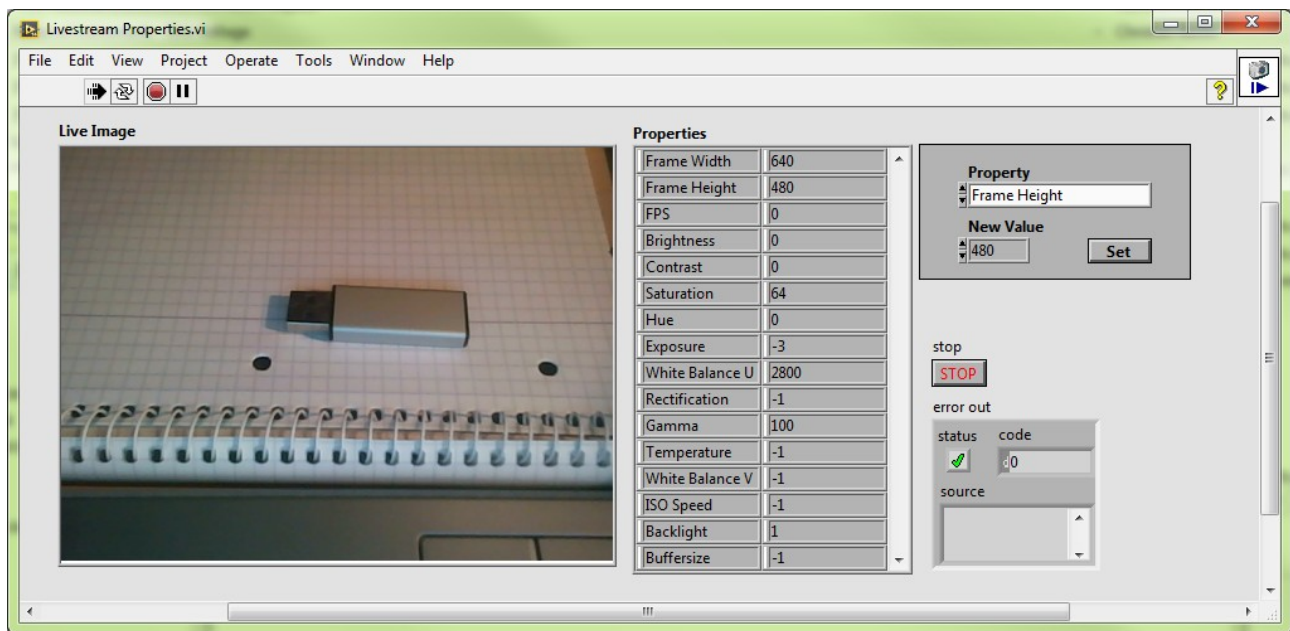


Abbildung 13: Demo Livestream Properties

### Camera Stream Video Writer

Speichert den Livestream als Videodatei auf der Festplatte.

### File Stream

Liest eine Videodatei von der Festplatte.

### Network Stream

Öffnet eine Netzwerkressource und zeigt diese an.

Als Standardwert ist hier eine externe URL angegeben, auf deren Inhalte MSC keinen Einfluss hat. Für die Inhalte der verlinkten Seiten ist stets der jeweilige Anbieter oder Betreiber der Seiten verantwortlich. Für die Inhalte, Verfügbarkeit und die Richtigkeit der Informationen verlinkter Websites fremder Informationsanbieter wird keine Gewähr übernommen.

## 11 Anhang

### 11.1 Entfernen der Verknüpfung zu IMAQ

Wenn Sie auf Ihrem PC NI Vision, bzw. IMAQ nicht installiert haben, kann eine Warnung beim Öffnen auftreten.

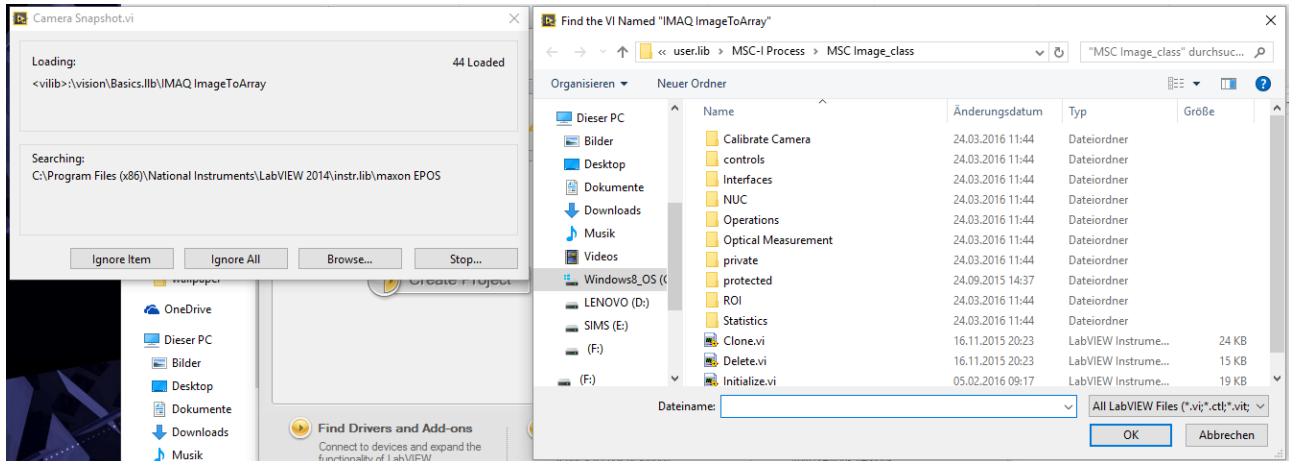


Abbildung 14: Warnung IMAQ

Um diese zu unterbinden, gehen Sie bitte wie folgt vor: Öffnen Sie die Datei user.lib/MSC-I-Process/MSC Image\_class/MSC Image.lvclass mit LabVIEW. Dort finden Sie unter Interfaces die Dateien „Read IMAQ Image.vi“ und „Write IMAQ Image.vi“. Markieren Sie sie und entfernen sie.

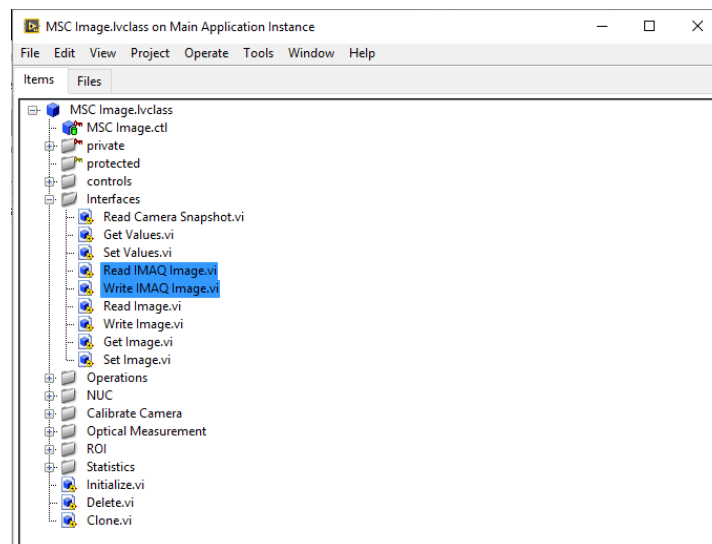


Abbildung 15: Zu entfernende Dateien

Danach wird die geänderte Datei lvclass über File – Save gespeichert. Die Warnung wird nicht mehr auftreten.



## Abbildungsverzeichnis

Abbildung 1: Errorcluster.....	4
Abbildung 1: Errorcluster.....	4
Abbildung 2: Struktur des MSC Image.....	5
Abbildung 3: Camera Snapshot Demoprogramm.....	6
Abbildung 4: Bildinhalt bei Initialisierung (NA.png).....	7
Abbildung 5: Kalibrierungsfunktionen.....	13
Abbildung 6: Testmuster Chessboard.....	13
Abbildung 7: Kamerakalibrierung mit farbig markiertem Testmuster.....	16
Abbildung 8: Fischaugeneffekt.....	17
Abbildung 9: Struktur der ROI.....	19
Abbildung 10: Struktur des MSC Videos.....	24
Abbildung 11: Einige verfügbaren Kameraeinstellungen.....	26
Abbildung 12: Demo ROI.....	30
Abbildung 13: Demo Livestream Properties.....	31
Abbildung 14: Warnung IMAQ.....	32
Abbildung 15: Zu entfernende Dateien.....	32