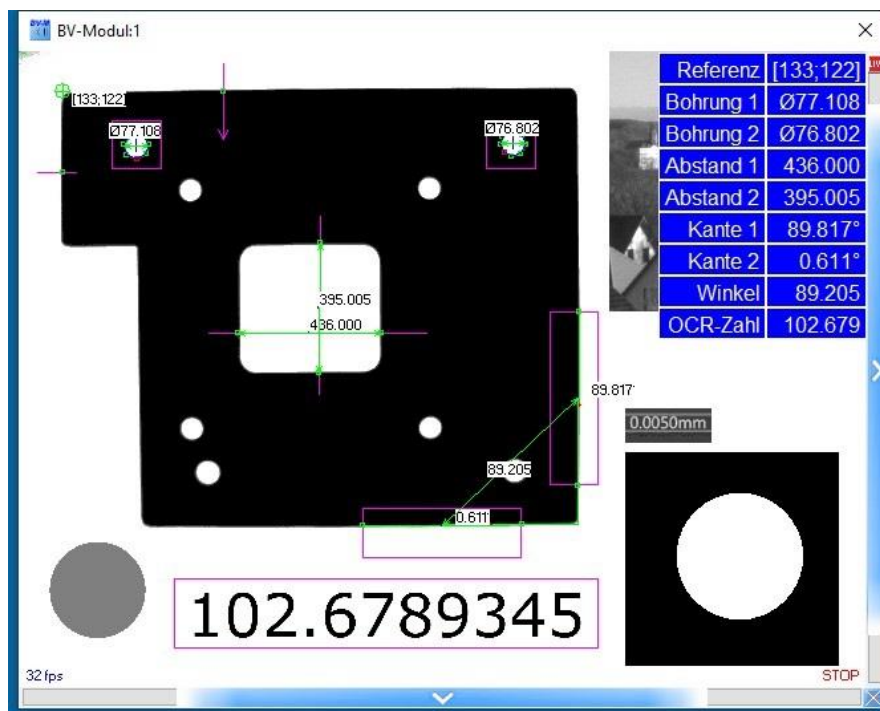


# *Bedienungsanleitung*

# **BVMODUL**

**Version: 2.3 (01.11.24)**



- messende Bildverarbeitung als eigenständiges Programm mit externen Datentransfer
- als DLL-Modul in eigene Anwendungen integrierbar (optimal für LabVIEW, C...)
- unterstützt verschiedene Industriekameras
- parallele Messung mit bis zu 10 Kameras
- innerhalb weniger Minuten ist das fertige Messprogramm erzeugt und einsatzbereit
- USB Stick mit Arbeitsplatzlizenz zur unbegrenzten Nutzung in beliebig vielen Anwendungen - keine Jahres- oder Abo-Gebühren
- Ergänzung von speziellen Funktionen, weiteren Kamerateypen... auf Anfrage möglich.

© Copyright 2024  
Messtechnik Klüger

---

## Installation:

Mit der Ausführung des Installationsprogramms wird die Software auf dem PC installiert. Unter Umständen benötigen sie Administratorrechte dazu.

Im Programmverzeichnis wird mit der Installation der Ordner "BVMODUL" erzeugt. Darin befinden sich alle notwendigen Dateien.

- [MTK\\_BVM.exe](#)

Eigenständiges Windowsprogramm mit einem Kamerabild, das direkt ausgeführt werden kann. Wird das Programm ein weiteres Mal gestartet, wird damit eine weitere Instanz geöffnet. D.h. es gibt nun 2 unabhängige Programmfenster. Bis zu 10 parallel arbeitende Fenster sind möglich. Jedes Fenster hat einen festgelegten Index von 1..10 und seine eigene Konfiguration.

**Wichtig: eine Kamera kann immer nur von einer Programminstanz verwendet werden.**

- [MTK\\_BVM.lib](#) (für einfache direkte Einbindung in C++ Anwendungen mit MFC)

- [MTK\\_BVM.dll](#) (für allgemeine Einbindung in Anwendungen)

- [mtk\\_bvm.h](#)

Zur Implementierung als Modul in eigene Programmanwendungen. Es muss lediglich die Headerdatei und die DLL/LIB in die eigene Anwendung integriert werden. In der Headerdatei findet man die Beschreibung welche Funktionen verfügbar sind. Diese sind sehr einfach aufgebaut, ermöglichen das Öffnen und Schließen von Modulfenstern, die Triggerung von Messungen und das Abfragen von Messwerten aus den Modulfenstern.

- Beispielanwendung: [Example\\_BVMODUL.exe](#)

Einfaches Beispiel für die Einbindung der DLL in eine eigene Anwendungen. Der Quellcode (für VC2022) findet man im Installationsordner.

- Beispielanwendung: [BV\\_MODUL2.vi](#)

Einfaches Beispiel wie man die DLL mit LabVIEW verwendet. Die Kamerafenster werden per LabView-Befehl ferngesteuert. Es werden Messwerte übernommen und verarbeitet.

Jedes Modulfenster verwendet eine Konfigurationsdatei (\*.cfg) die alle Einstellungen und auch den Prüfplan enthält.

Optional kann man die "exe"-Version mit 2 Parametern starten. Der erste Parameter ist die gewünschte Modulnummer (1..10), als 2.Parameter kann die gewünschte Konfigurationsdatei (cfg) vorgegeben werden.

Ohne Vorgabe der Modulnummer: Es wird jede neu geöffnete Programminstanz mit fortlaufender Modulnummer gestartet.

Ohne Vorgabe der Konfigurationsdatei: Es wird die Standardvorgabedatei (bvmodul\_1.cfg) verwendet bzw. die individuell ausgewählte cfg-Datei im Konfigurationsdialog vom BV-Modul.

Verwendet man die DLL in der eigenen Anwendung kann man beliebige Konfigurationsdateien verwenden und damit zwischen beliebig vielen Prüfplänen umschalten. Wird der Dateiname für die cfg-Datei ohne Pfadangabe vorgegeben, wird die Datei im Systemordner erwartet.

*Hinweise:*

*Das Installationsprogramm erzeugt automatisch den Systemordner "C:\BVMODUL". Sollte dies vom System verhindert worden sein, ist ein beliebiger Ordner auf dem PC mit den notwendigen Systemdateien anzulegen. Beim Start des BV\_MODULS muss der Ordner vom Anwender einmalig ausgewählt werden. Blockiert der Vieren-Scanner die Installation ist die Blockierung durch eine "Ausnahmebestätigung" aufzuheben.*

## Externe Programme und Treiber:

Nach der Installation arbeitet das BV-Modul zunächst mit einer Bilddatei (bmp). Für die reale Messung mit einer Kamera müssen vorab alle notwendigen Treiber und Testprogramme des Kameraherstellers installiert werden.

z.Z. werden folgende Quellen unterstützt:

- **IDS-Imaging:** alle UEYE - Kameramodelle (mit Grauwert)
- **Daheng Imaging:** alle GX / Mercury Kameramodelle (mit Grauwert)
- **Bitmap** als Grauwertdatei (wird mit jedem Bildtrigger aktualisiert)
- **Desktop** Screenshot Fenster- /Bildschirmbereiche (Farben werden in Grauwert umgewandelt, Bild wird laufend aktualisiert)

Eine Auswahlbox ermöglicht die spezifischen Wahl eines "Fensters" (aus allen geöffneten Desktop-Programmen). Wählt man so ein Fenster funktioniert die Bildverarbeitung unabhängig von Position, Größe und Lage des Fensters.

Für den Datentransfer der Messdaten nach EXCEL wird eine aktuelle Excel-Installation auf dem PC benötigt.

## Lizenzierung:

Das Programm wird durch einen Dongle vor illegaler Nutzung geschützt. Eventuell ist dieser Dongle nur für einen vorgegebenen Testzeitraum voll funktionsfähig. Ist diese Zeit abgelaufen oder fehlt der Dongle arbeitet das Programm mit Einschränkungen bzw. zeitlich beschränkt.

Geht der Dongle verloren, verliert man damit auch alle erworbenen Programmrechte.

## Datenschutz:

Beachten sie -beim Einsatz des Programms- unbedingt die gültigen gesetzlichen Vorschriften zum Datenschutz. Das Programm bietet technisch die Möglichkeit Kamerabilder und aktuelle Monitorinhalte (screenshots) unbemerkt auszuwerten und zu speichern.

Diese Möglichkeit liegt aber alleinig im Verantwortungsbereich des Anwenders. Grundsätzlich sammelt, sendet oder empfängt das Programm KEINE Daten an den Entwickler oder externe Dritte (siehe Anhang).

## aktuell verfügbare BV-BASIS-Funktionen:

Nr.	Bezeichnung	Parameter/Varianten	Ergebnis
0	Referenz	Punkt/Gerade/Schnittpunkte....	Referenzpunktkoordinaten [x/y] Winkelkorrektur
1	Helligkeit	Einzelpixel/Bildbereich	Grauwert (0...255)
2	Punkt auf Suchstrahl	einseitige Kantensuche / hell / dunkel	Punktkoordinaten [x1;y1]
3	Punkt/Punkt auf Suchstrahl	doppelseitige Kantensuche / hell / dunkel	Punktkoordinaten [x1;y1]; [x2;y2]
4	Gerade mit 2 Suchstrahlen	Objektkante approximiert durch 2 Kantenpunkte (hell / dunkel)	Kantenwinkel, Mittelpunktlage x/y
5	Gerade mit Suchbereich	Objektkante im Suchfenster / hell/dunkel/ Antastrichtung	Winkel, Linearität, Mittelpunktlage
6	Kreis mit Suchbereich	Kreis im Suchfenster / hell / dunkel / Antastrichtung	Durchmesser, Formabweichung, Mittelpunktlage, Geschwindigkeit
7	Kreisbogen mit 3 Suchstrahlen	Kreisbogen / hell / dunkel / 3 beliebige Suchstrahlen	Radius, Mittelpunktlage
8	Kantenabstand mit Suchbereich	Abstand von zwei (etwa) parallelen Kanten (hell/dunkel) verschiedene Rasterabstände wählbar (1/5/10/20/30) verschiedene Störungsfilter wählbar (unebene Kanten glätten)	Abstand /max./min./mittel/dif. Kantenlänge
18	Markerposition	exakte Lage von kreisförmigen Objekt finden /hell/dunkel + freie Bezugskoordinaten (x/y) im Bild	Mittelpunktlage [x;y] Referenzpunktabweichung [dx;dy] Durchmesser Geschwindigkeit
19	Zentrumsuche	ultra-exakte Mittelpunktberechnung von kreisf. Objekt /hell/dunkel Ziel-Bezugskoordinaten (x/y)	Mittelpunktabweichung (x/y/abs)
20	Einzelziffer OCR	Zahlenerkennung: grafisch / 7Segment/ hell / dunkel /kursiv	Ziffernwert (numerisch)

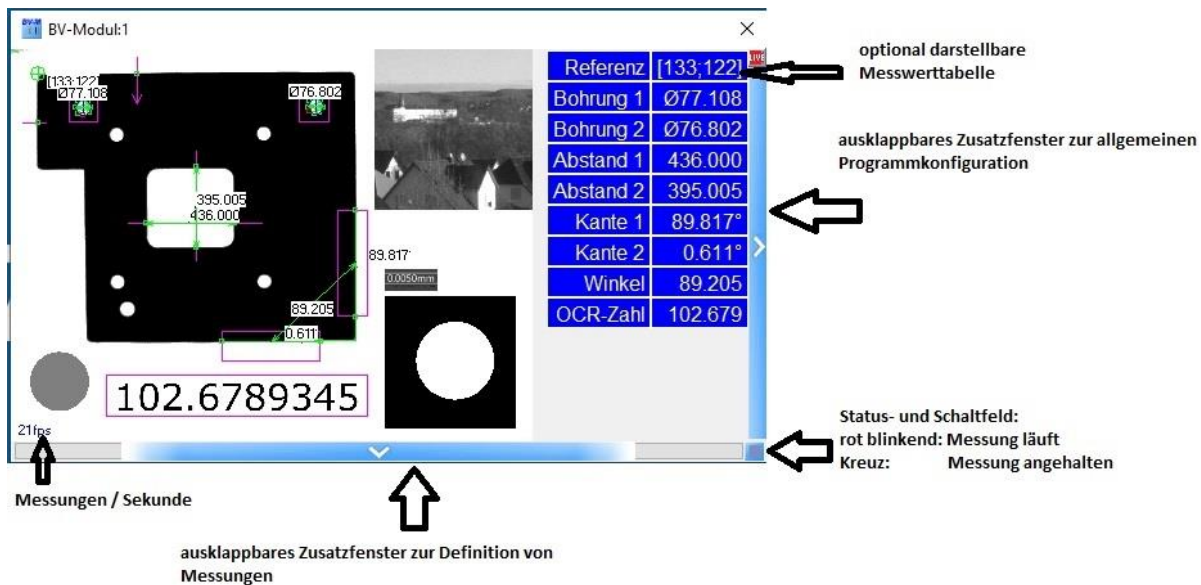
21	Zahl OCR	Zahlenerkennung: grafisch / 7Segment/ hell / dunkel /kursiv	Zahlenwert (numerisch)
22	Einzelziffer Mustervergleich	Erkennung mit 13 (teach) Vergleichsmustern	Ziffernwert (numerisch)
23	Zahl Mustervergleich	Erkennung mit 13 (teach) Vergleichsmustern	Zahlenwert (numerisch)
24			
25	Bild speichern	bmp-Einzeldatei / fortlaufende bmp-Dateien	BMP-Datei(en)
	<a href="#">neue Basisfunktionen auf Anfrage</a>		

### aktuell verfügbare Kombinationsberechnungen:

Nr.	Bezeichnung	Parameter/Varianten	Ergebnis
1	Abstand PUNKT/PUNKT	2 beliebige Ergebnisskoordinaten [x/y] [x/y]	Abstandswert
2	Lotabstand PUNKT / GERADE	beliebige "Gerade" +Ergebniskoordinate [x/y]	Abstandswert
3	Winkel GERADE/GERADE	2 x beliebige "Gerade"	
11 ... 16	Mittelwert, Max., Min., Unterschied, Addition, Subtraktion	beliebige 2 numerische Ergebnisse	numerischer Wert

## Los gehts.....mit den wichtigsten Programmkonfigurationen:

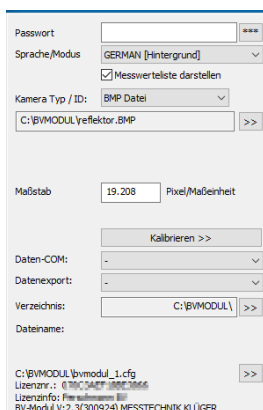
Starten sie das Programm mit der Datei: [MTK\\_BVM.exe](#)



**WICHTIG:** Rechts/unten befindet sich ein Schaltfeld - damit Startet oder Stoppt man alle Messungen! Rechts/oben kann man mit dem Schaltfeld "LIVE" das aktuelle Kamerabild zum Standbild machen.

Nach der Erstinstallation wird als Bildquelle ein Bitmap verwendet, der Prüfplan beinhaltet bereits einige Messungen. Positionieren sie das Fenster auf gewünschte Position und Größe, diese Einstellung wird abgespeichert.

Wir beginnen mit der "allgemeinen Programmkonfiguration" durch Öffnen des rechten Zusatzfensters. Je nach Konfiguration werden nur notwendige Fensterelemente angezeigt.



### Passwort:

Die gesamte Konfiguration kann durch Vorgabe eines individuellen Passworts vor unberechtigtem Zugriff geschützt werden. Ohne Vorgabe sind alle Eigenschaften frei editierbar. (fest definiertes Administratorpasswort: *mtkadmin*)

### Sprache/Modus:

Wählen sie hier die gewünschte Programmsprache sowie den Darstellungsmodus. Das Programmfenster kann sowohl im Vordergrund als auch im Hintergrund zu anderen Programmen und Fenstern laufen. Wird es von (einer externen Anwendung) als DLL ausgeführt ist auch ein "unsichtbarer" Modus möglich.

### Kamera Typ:

Je nach Kamera-/Quellenauswahl hat das Fenster unterschiedliche Auswahlfelder. Bei der Bitmap-Datei kann die Quelldatei ausgewählt werden. Bei Kameras entsprechende Typen, ID Nummern

sowie weitere Bildparameter.

Maßstab:

Jedes Bild besteht aus Pixeln mit festen Abständen. Entsprechend der optischen Abbildung ergibt sich ein Verhältnis von Pixelabstand zur realen Objektgröße. Dieser Wert ist allerdings mit verschiedenen Fehlerfaktoren in der optischen Abbildung behaftet. Geben sie hier zunächst einen ungefähren Wert vor. Mit dem Maßstab 1,000 entsprechen die Einheiten im Programm dem Pixelwerten. Mit dem Ausführen der automatischen Kalibrierfunktion wird auch der Maßstab automatisch neu berechnet.

Kalibrieren:

Benötigt wird eine Kalibrierkarte mit definiertem Punkte-Raster. Objektiv- und Abbildungsfehler können durch Vergleich von theoretischer und realer Rasterpunktkoordinate berechnet und bei der Messung korrigiert werden.

Datenschnittstelle:

Das BVMODUL kann sich so verhalten, als wäre es ein reales, tastendes Messgerät mit USB/RS232 Schnittstelle (z.B. ein Messschieber oder eine Messuhr). Das Programm überwacht die vorgegebenen Schnittstellen und sendet den aktuellen Wert über diese Schnittstelle an eine externe Auswertesoftware.

Datenexport:

Das BVMODUL kann alle Messwerte in eine vorgegebene CSV/TXT-Datei oder direkt in eine EXCEL-Vorlage schreiben. Dabei werden grundsätzlich alle aktiven Messwerte in eine Zeile kopiert. Man kann entscheiden ob die Zeile jeweils überschrieben wird oder fortlaufende Zeilen erzeugt werden.

Bei gleichzeitigen Zugriff von mehreren PC's auf dieses Datenfile wird der direkte EXCEL-Export nicht empfohlen. Ist eine Datei von einem externen Nutzer bereits geöffnet kommt es automatisch zum Blockieren der Schreibrechte und es kann keine neue Messung zugefügt werden! Verwenden sie hier besser csv-Dateien und verlinken diese automatisch mit einer Excel-Vorlage. Über das Schaltfeld "Daten aktualisieren" kann man bei Bedarf im Excel immer die aktuellen Messwerte sehen (ohne Blockierung).

Verzeichnis:

Datenverzeichnis für Excel, csv oder vom Programm erzeugte Bilddateien (bmp).

Zusatzinfos:

Die wichtigsten Infos zum Programm - wie die aktuell verwendete Konfigurationsdatei, Lizenzinfos und Programmversion werden hier angezeigt.

Auswahl der cfg-Datei:

Jedes BV-MODUL arbeitet mit jeweils einer Konfigurationsdatei (cfg). Diese enthält sämtliche Informationen zur Konfiguration sowie alle Prüf- und Messeinstellungen. Mit diesem Schaltfeld kann eine individuelle cfg-Datei vorgewählt werden.

**Wichtig:** Das gewählte Verzeichnis für die cfg-Datei bestimmt gleichzeitig den "Systemordner" der weitere unverzichtbare Dateien enthält - speichern sie neue cfg-Dateien grundsätzlich in einem gültigen Systemordner!

Die gewählte cfg-Datei wird beim nächsten Programmstart nur dann wieder vorgewählt, wenn keine prioritäre Programmfunktion darüber steht (z.B. cfg-Vorgabe über Startparameter oder DLL-Funktion).

Während das Konfigurationsfenster geöffnet ist erfolgt KEINE laufende Messung. Einige Änderungen erfordern das erneute Schließen und Öffnen des Konfigurationsfenster, eventuell auch einen Programmneustart.

## weiter gehts.....mit der ersten Bildverarbeitung

öffnen sie jetzt das untere Zusatzfenster zur "Definition von Messungen".....

### Grundlagen:

Das Vermessen bzw. Verarbeiten von Bildern erfolgt durch Suchen von Helligkeitsunterschieden - sogenannten Kanten- im Bild. Man kann auch die Helligkeit von einzelnen Pixeln oder Pixelbereichen verwenden, um damit Rückschlüsse auf das Vorhandensein oder die Qualität von Objekten zu ziehen. Für die Messgenauigkeit ist ein scharfes, kontrastreiches Bild mit eindeutig erkennbaren Kanten entscheidend. Hier spielt die Beleuchtung eine entscheidende Rolle. Auf eine Bildvorverarbeitung (spezielle Filter) wurde in diesem Programm bewusst verzichtet. Schwerpunkt liegt auf der möglichst exakten Messung und diese wird durch jegliche Art von digitaler Vorverarbeitung verschlechtert. Bei sehr guter Bildqualität kann man mit einer Messgenauigkeit besser  $\pm 1$  Pixel rechnen.

Dieses Programm bietet die Möglichkeit von maximal 50 Messungen im Bild. Die Messung mit dem Index[0] ist immer für einen Referenzpunkt reserviert. Die Reihenfolge aller übrigen Messungen und deren Bezeichnung kann frei gewählt werden.

Während das Konfigurationsfenster geöffnet ist, wird die aktuell gewählte Messung fortlaufend ausgeführt. Sämtliche Änderungen werden unmittelbar gültig und können nicht rückgängig gemacht werden.

### Messung / Bezeichnung:

Wählen sie zunächst die gewünschte Messung über den Index[] in der Auswahlbox.

Durch Eingabe einer Bezeichnung erkennt das Programm, dass es diese Messung tatsächlich gibt. Damit erscheint diese Bezeichnung auch in der "Messwertliste" im Hauptfenster. Eine sehr komfortable, schnelle Methode zur Indexauswahl ist es direkt in dieser Liste eine

Messung anzuklicken. Der vorgestellte Index[] muss nicht eingegeben werden und wird automatisch immer ergänzt.

Eine bereits mit Namen versehene Messung kann "aktiv" oder "inaktiv" sein. Inaktive Messungen werden nicht ausgeführt und als hellgrauer Wert in der Liste dargestellt.



### Messmodus:

Eine Messung kann sowohl eine unabhängige "Basismessungen" (wie Kantenpositionen, Kreise, Geraden, Helligkeiten....), aber auch eine Zusatzberechnungen oder mathematische Kombinationen aus bereits vorhandenen Messungen sein. Zunächst betrachten wir Basismessungen, weil diese die Grundlage für die anderen Typen darstellen.

Alle BV-Operationen werten bestimmte Bildbereiche aus. Zur Festlegung der Suchbereiche (Größe und Position) wird die Maus verwendet. Unter ungünstigen Umständen kann es vorkommen, dass ein Bereich nicht mehr mit der Maus fassbar ist - mit dem Schaltfeld "RESET Suchbereich" wird der Suchbereich in jedem Fall wieder sichtbar und kann nun wie gewünscht positioniert werden.

### Objekttyp / Suchparameter....:

Es sind unterschiedliche, direkt messbare Objekttypen wählbar. Dazu zählen Kanten, Geraden, Kreise, Kreissegmente..... sowie Spezialobjekte. Je nach gewählten Objekt müssen unterschiedliche Suchbereiche für dieses Objekt definiert werden. Bei der Kantensuche wird z.B. ein Suchstrahl (mit Anfangs- und Endpunkt), bei Kreisen und Geraden ein Suchrechteck definiert. Oft sind weitere Suchparameter notwendig.

Bei Kanten -zum Beispiel- kann es sich um eine dunkle oder helle Kante handeln, bei Kreisen muss man unbedingt die Antastrichtung auswählen. Da Kanten nur über Helligkeitsunterschiede gefunden werden, muss der relevante Helligkeitsgrenzwert vorgegeben werden. Der Helligkeitswert eines Pixels liegt im Bereich von 0 (schwarz) bis 255 (weiß) und wird über den Schieberegler ausprobiert.

### Hauptmesswert:

Ein Basisobjekt hat i.a. verschiedene Eigenschaften, die mit der Bildverarbeitung erfasst werden. Für einen Kreis wäre dies:

- Durchmesser
- Zentrumsordinate (X)
- Zentrumsordinate (Y)
- Abstand vom Zentrum zum Referenzpunkt
- Kreisformabweichung
- Geschwindigkeit

Je nach Bedarf wählt man hier den gewünschten Wert aus. Grundsätzlich werden alle Werte berechnet - aber es kann immer nur ein **Hauptmesswert** verarbeitet und in der Liste dargestellt werden!

*Möchte man einen weiteren Wert darstellen ist es nicht notwendig das gleiche Objekt nochmal zu messen. Es genügt, wenn man im Messmodus für eine weitere Messung "Zusatzmesswert" auswählt und hat damit Zugriff auf weitere Kennwerte von dieser Basismessung.*



Jeder **Hauptmesswert** kann individuell mit Faktor/Offset verrechnet werden. Zur exakten Kalibrierung mit einem bekannten Musterteil steht eine eigene Funktion zur Verfügung, die den exakten Faktor automatisch berechnet.

### Darstellung:

Während der Messung können Suchbereich, Maßpfeil und Messwert als Overlay im Kamerabild eingeblendet werden. Die Formatierungsvorgabe bezieht sich auf die Darstellung des Hauptmesswertes im Overlay sowie in der Messwerttabelle. Der Datenexport von Messwerten erfolgt -davon unabhängig- immer mit 6 Nachkommastellen.

### Messwerttrigger und Mehrfachmessung:

Jede Messung muss durch einen Messwerttrigger ausgelöst werden. Das Triggersignal kann durch einen Timer, eine Keyboardtaste oder ein externes Signal erfolgen. In der DLL Version kann das externe Programm einen Trigger auslösen. Wird das Schaltfeld "laufend" aktiviert, dann wird diese Messung sofort nach jeder neuen Bildaufnahme ausgeführt. Ist die "laufende" Triggerung aktiviert sind die anderen Optionen praktisch sinnlos, da diese immer mit höchste Messfrequenz ausgeführt wird.

Wichtig: Jeder einzelne Basismesswert hat seinen eigenen Trigger und kann damit zu unterschiedlichen Zeitpunkten ausgelöst werden. Ein Messwert ohne Trigger wird nie ausgeführt (Ausnahme ist eine mögliche Triggerung über die DLL Schnittstelle).

### Triggervarianten:

- **laufend** (Messung erfolgt mit jeder neuen Bilderfassung)
- **Timer** (Messung erfolgt im Abstand der vorgegebenen Sekunden)
- **Taste** (Messung erfolgt beim Drücken der vorgegebenen Taste - Keyboard oder Maustasten stehen zur Wahl)
- **COM-Trigger** (Messung erfolgt wenn eine Messwertabfrage über eine RS232/USB Schnittstelle erfolgt)
- **Bildsignal** (Messung erfolgt wenn ein anderer Messwert  $>/<$  als ein Vorgabewert ist, es kann zwischen einmaliger Flankenerkennung oder laufenden Triggersignal beim Erreichen der Bedingung gewählt werden)

Solange keine neuer Trigger für einen Basiswert ausgelöst wird bleibt der zuletzt gemessene Wert in der Anzeige und beim Export erhalten (eingefroren).

Da Messwerte von Bildverarbeitungsfunktionen immer mit einer gewissen zufälligen Abweichung versehen sind, macht es Sinn diese durch eine Mittelwertbildung zu stabilisieren. Nach einem Triggersignal werden dazu bis zu 99 folgende, "laufende" Werte miteinander verrechnet. Diese Funktion ist auch geeignet, um Maximum- oder Minimumwert zu erhalten. Der Zeitbedarf bis ein angeforderter Wert tatsächlich zur Verfügung steht verlängert sich entsprechend.

**Datenexport:**

Ist die Option "Datenexport" zu EXCEL oder CSV/TXT Datei aktiviert, kann mit diesem Schaltfeld festgelegt werden, ob mit dem

Trigger auch ein Datenexport auslösen soll.

**Verwendung von Zusatzmesswerten:**

Jedes Messobjekt kann nur einen Hauptmesswert besitzen. Manchmal sind aber auch weitere Kennwerte auszuwerten. An einem Kreis kann man z.B. neben dem Durchmesser auch noch die Formabweichung sowie die Position des Kreismittelpunktes berechnen.

Man muss diesen Kreis jetzt nicht mehrfach messen sondern kann weitere Messungen mit dem Messmodus

"ZUSATZMESSWERT" erzeugen. In der Liste der Objekttypen erscheinen nun alle zur Verfügung stehende Basismessungen mit ihren Index-Bezeichnungen.

Zusatzwerte benötigen keinen Trigger und auch keine Suchkriterien da die Messung bereits mit der Basismessung erfolgt ist.

**Verwendung von Kombinationsberechnungen:**

2 Basismesswerte können mathematisch zu einem Kombinationswert verrechnet werden. Typische Anwendungsfälle sind:

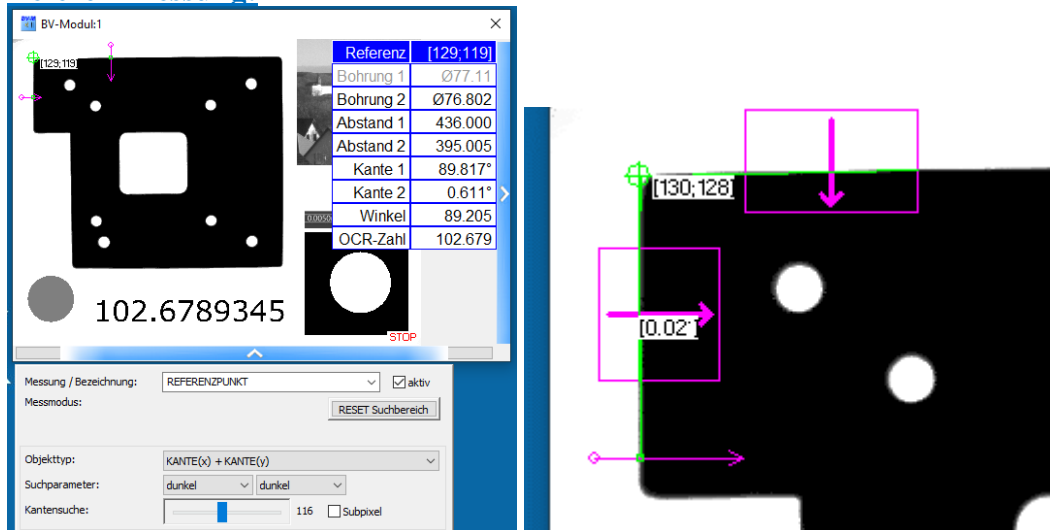
- Abstand von 2 Punkten
- Lotabstand eines Punktes auf eine Gerade
- Winkel zwischen 2 Geraden
- Mittelwert
- Größtwert

- Kleinstwert
- Unterschied
- Addition
- Subtraktion

Mit der Vorwahl einer Kombinationsberechnung werden zwei Listen zur Auswahl der möglichen Index-Bezeichnungen angezeigt. Gibt es mehrere mögliche Kennwerte erscheinen weitere Auswahl-listen.

## Spezialfunktionen:

### Referenzmessung:



In vielen Anwendungsfällen liegen die Messobjekte genau genug, damit die eingelernten Suchbereiche alle Basisobjekte sicher erkennen - dann braucht man die Referenzmessung nicht unbedingt. Anders sieht es aus, wenn absolute X/Y Koordinaten benötigt werden oder die Messobjekte etwas verschoben und gedreht sind. Kleinste Lageunterschiede von Messobjekt oder Kamera können dazu führen, dass Positionskoodinaten und Winkellagen falsch berechnet oder Suchbereiche außerhalb vom Messobjekt liegen.

Durch die Bestimmung eines Referenz- bzw. Bezugspunktes werden geringe Lageunterschiede berechnet und alle Suchfenster in X und Y-Richtung verschoben. Gehört zu den Messaufgaben auch die Messung der absoluten Winkellage einer Kante ist eine Korrektur von diesem Wert unbedingt notwendig. Zum Referenzpunkt wird dafür eine zusätzliche Referenzkante benötigt.

Vorgefertigte Referenzpunktobjekte sind:

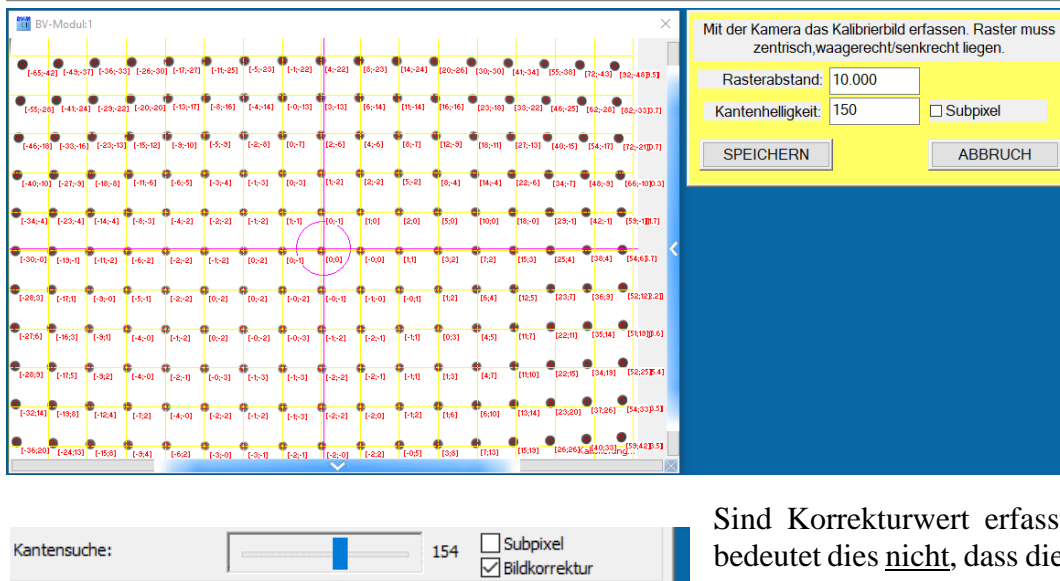
- Bohrungen (Kreise)
- theoretischer Schnittpunkt von zwei Kanten (x,y)
- theoretischer Schnittpunkt von zwei Geraden

**Wichtig:** Die Referenzmessung kann nur kleine Abweichungen von Position und Winkelabweichungen korrigieren. Sie ist nicht geeignet um willkürlich liegende Objekte zu vermessen!

### Kalibrierung mit Rasterbild

Die Funktion ermöglicht Abbildungs- und Objektivfehler (in gewissen Grenzen) zu kompensieren. Verwenden sie ein geeignete Kalibrierbild mit möglichst exakt liegenden Rasterpunkten (maximal 50x50) an genau der Stelle, wo sich das Messobjekt normalerweise befindet. Zu jedem Rasterpunkt wird die Abweichung zu den theoretisch errechneten Koordinaten berechnet.

Ist der "Rasterabstand" bekannt, wird gleichzeitig der Maßstab berechnet und im Konfigurationsdialog verwendet.

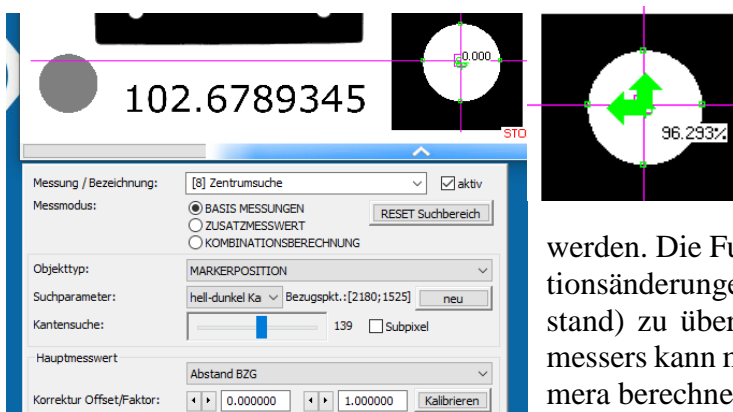


Sind Korrekturwert erfasst und gespeichert bedeutet dies nicht, dass die komplette Kamerabildddarstellung damit korrigiert wird. Für

jeder Messwert kann festgelegt werden, ob eine Korrektur erfolgen soll. Diese erfolgt dann ohne messbaren Performanceverlust.

Mit einem speziellen Algorithmus werden alle Objektkoordinaten optimal auf das eingelernte Koordinatensystem transformiert. Die Korrekturgenauigkeit kann allerdings nur so gut, wie die Genauigkeit der Rasterpunktvorlage sein.

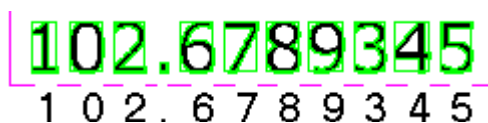
## MARKERPOSITION und ZENTRUMSUCHE



Die Spezialfunktion "Markerüberwachung" ermöglicht die Lage und Größe eines "Markers / Kreises" genau zu berechnen. Als Besonderheit kann die Lageabweichung gegenüber einer eingelernten Ausgangsposition berechnet werden. Die Funktion ist damit bestens geeignet um Positionsänderungen von Markern (x, y und absoluter Abstand) zu überwachen. Mit der Berechnung des Durchmessers kann man sogar den Abstand des Markers zur Kamera berechnen/abschätzen.

Die Funktion: "Zentrumsuche" wurde speziell für die hochgenaue Positionierung von Reflektoren oder Markern an Objekten entwickelt. Die Berechnung ist extrem genau und schnell. Richtung (x/y) und Betrag der Abweichung werden durch Pfeilgröße und -Pfeilfarbe visualisiert. Verwenden sie den "Korrekturfaktor" für die Empfindlichkeit der Pfeildarstellung.

## OCR - Zahlenerkennung



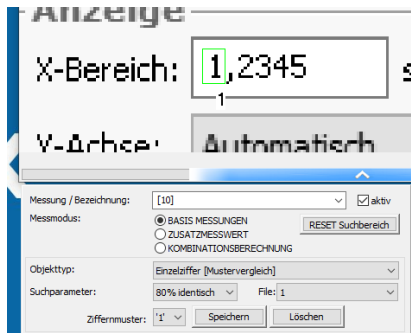
Bei der Spezialfunktion "OCR-Zahl" handelt es sich um eine Mustererkennungssoftware. Sie ist ausschließlich für die Erkennung von grafischen und 7-Segment Zahlenformaten verwendbar. Die

grafische Erkennung erfolgt durch Konturverfolgungen und die Bewertung nach bestimmten Kriterien. Sie ist damit weitgehend unabhängig von Größe und Schriftart. Voraussetzung ist aber eine Mindestgröße (in Pixeln) und gute Qualität damit die Konturverfolgung die Außenkanten einer Ziffer richtig erfassen kann!

Die 7-Segment Zahlenerkennung erfordert die genaue Positionierung der Suchbereiche auf die gesuchten Ziffern. Die Ziffern müssen waagrecht im Suchbereich liegen.

### Einzelziffer und Zahlen durch Mustervergleich

Die OCR-Zahlenerkennung versagt, wenn die Ziffern sehr klein sind. Das ist z.B. der Fall wenn man als Datenquelle einen Bildschirmarschnitt (Screenshot vom Desktop) verwenden möchte.



Beispiel: Ein komplexes Messsystem ermöglicht lediglich die Ausgabe von Messwerten auf dem Monitor (keine API-Funktion). Um diese Werte mit eigener Software zu verarbeiten scannt das BV-Modul permanent den Bildschirmbereich mit dem Messwert. Jede Ziffer (0...9) wird 1x eingelernt. Durch Mustervergleiche erkennt das BV-MODUL den aktuellen Messwert.

Für Kameraanwendungen ist der Mustervergleich schwierig - durch Bildschwankungen ist eine 100% Übereinstimmung von Bildausschnitt und eingelernten Muster sehr unsicher. Zur Mustererkennung muss man zunächst jede einzelne Ziffer und die Sonderzeichen

(,+-) mit dem Objekttyp "Einzelziffer (Mustervergleich)" einlernen. Dazu den Suchbereich auf eine Ziffer sehr genau positionieren .... den Ziffernmustertyp "1,2,3,..." auswählen und "Speichern". Ab jetzt wird diese Ziffer erkannt, wenn sie sich im Suchbereich befindet.

Die Muster der einzelnen Ziffern werden in einer Datei gespeichert. Sollen in einem Bild mehrere unterschiedliche Ziffernmuster erkannt werden, besteht die Möglichkeit mehrere Datendateien zu verwenden.

Sind alle Ziffernmuster abgespeichert kann man zum Objekttyp "Zahl (Mustervergleich)" wechseln. Hier wählt man die Musterdatei aus und positioniert den Suchrahmen großzügig um die gesamte Zahl.

### Messung von Geschwindigkeiten

Bei einigen Basisobjekten besteht die Möglichkeit der "Geschwindigkeitsmessung". Vorausgesetzt wird, dass das Messobjekt mit mindestens 2 Mehrfachmessungen erfasst wird! Das Programm vergleicht die Koordinatenunterschiede mit dem Zeitindex der Bildaufnahmen und errechnet daraus die Geschwindigkeit pro Sekunde. Leider sind (PC bedingt) die Zeitindexe nicht sehr genau. Durch Vorgabe eines möglichst großen Wertes für die Mehrfachmessung kann man den Berechnungszeitraum vervielfachen und damit die Genauigkeit erheblich verbessern.

### Bild speichern

Der Objekttyp "Bild speichern" nimmt eine Sonderstellung ein. Das aktuelle Kamerabild wird dabei nicht vermessen sondern als Bitmap-Datei abgespeichert. Es wird das vorgegebene Verzeichnis für den Datenexport verwendet.

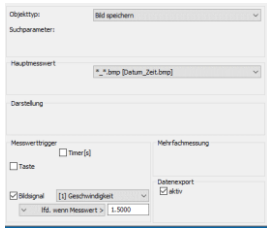
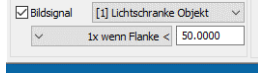
Zur Wahl steht:

- Abspeichern / Überschreiben der Datei: **"image.bmp"**.
- Erzeugen von neuen Dateien jeweils mit dem Dateinamen aus **"Datum+Zeit.bmp"**.

**Achtung: Dabei können sehr schnell enorme Datenmengen auf dem PC entstehen wenn der Trigger ungünstig gewählt wird!!!**

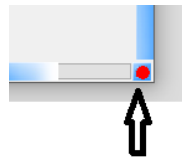
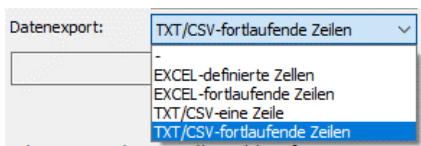
### Trigger mit Bildsignalen

**Beispiel:** In einer laufenden Produktion kommen die Messobjekte unbestimmt ins Bildfeld. Wenn das Objekt stabil liegt, soll die Messung 1x erfolgen. Dazu definiert man sich zunächst eine Basis-messung [1] die "laufend" ausgeführt wird (z.B. "Kantensuche" oder "Helligkeit"). Alle anderen Messungen werden mit dem Trigger: "Bildsignal" ausgelöst. Damit das Objekt auch tatsächlich stabil liegt verwendet man die Möglichkeiten zur Mehrfachmessung + Mittelwertbildung für den auslösenden Messwert([1]). Damit verzögert man den Trigger und unterdrückt Störungen.



**Beispiel:** Es wird laufend die Geschwindigkeit eines Objektes [1] gemessen. Wenn diese einen Grenzwert erreicht, soll der Wert (in csv-Datei) und das aktuelle Kamerabild (als bmp) abgespeichert werden. In dieser Konfiguration werden fortlaufend Trigger/Messungen ausgelöst solange die Geschwindigkeit größer "1.5" ist.

### **Hinweise zum Datenexport:**



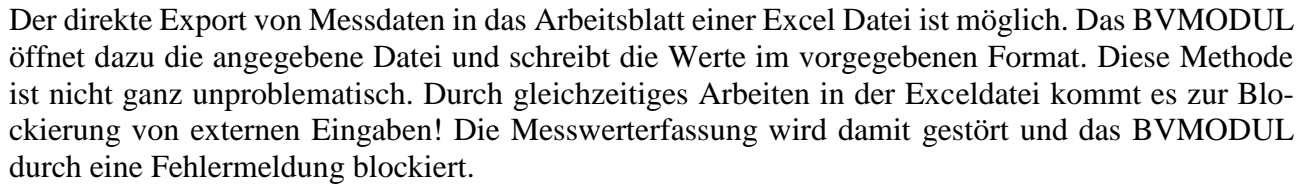
Die fortlaufende, schnelle Datenübertragung -insbesondere nach Excel- verbraucht sehr viel Rechenkapazität. Unter Umständen wird die Bedienung des PC's stark eingeschränkt. Stoppen sie in diesem Fall zunächst alle laufenden Messungen mit dem blinkenden Schaltfeld (rechts unten im Hauptfenster). Damit stoppt auch die Datenübertragung.

### EXCEL und CSV/TXT Datei:

Das Dateiformat ist fest definiert, die Werte sind durch Semikolons bzw. Spalten getrennt.

- Datum
- Uhrzeit
- Referenzposition X
- Referenzposition Y
- Referenzwinkel
- Hauptmesswert für Messindex [1]
- Hauptmesswert für Messindex [2]....
- Hauptmesswert für Messindex [50]

Es werden nur die Messwerte in die Zeile übernommen, die definiert und aktiviert sind. Für nicht aktivierte Messungen wird dennoch ein Semikolon eingefügt - damit bleibt die Spaltenzuordnung unabhängig von den aktivierten Messungen. Handelt es sich beim Messwert um den Objekttyp "Bild speichern" wird anstelle des Hauptmesswertes die kodierte Dateibezeichnung exportiert.



CSV/TXT Dateien können direkt mit einem Editor oder Excel geöffnet werden. Der aktuelle Stand entspricht dem Zeitpunkt des Öffnens. Es erfolgt keine Blockierung für weitere Schreiboperationen.



## Hinweise zum Einbinden der DLL in eigene Anwendungsprogramme:

Die Einbindung der DLL in eigene Programme erfolgt über die allgemeine DLL oder eine LIB (nur für VC Anwendungen verwendbar). Die notwendigen Bibliotheken (MFC-Dateien sind bereits statisch in die DLL integriert). Da das BVMODUL aber dynamisch gelinkte weitere DLL's von den Kameraherstellern benötigt, müssen sich diese Dateien ebenfalls im Programmverzeichnis befinden!

Auf dem Datenträger finden man ein Beispiel (für VisualStudio 2022 C++), das die LIB-Datei verwendet.

Folgender Beispiel-Ablauf wird empfohlen:

### 1. Einmalige Initialisierung der benötigten Module / Kameras:

```
//hier kann (bei Bedarf) die Sprache für alle Module vorgewählt werden...
SetLanguage_bvmodul(int mode);

//Starte das Modul 1:
start_bvmodul(1);

//hier kann (bei Bedarf) die Fenstergröße, Position und Darstellung vorgewählt werden
ShowWindow_bvmodul(1, 1);
SetWindowPos_bvmodul(1, 100, 100, 500, 400);
SetWindowStyle_bvmodul(1, 2);

//Modul 1 aktivieren:
run_bvmodul(1);
```

### 2. Schleife mit laufender Abfrage der gewünschten Messwerte in den jeweiligen Modulen.

Bei der externen Abfrage von Messwerten muss man sich bewusst sein, dass ein Kennwert nur dann gemessen wird, wenn vorab ein Triggersignal erfolgt ist. Wird dieses Triggersignal bereits im Modul (intern) erzeugt, wird bei der externen Messwertabfrage ein -unter Umständen erheblich verzögerter Messwert empfangen. Damit alle Messwerte synchron und schnell getriggert werden, ist es von Vorteil ein gemeinsames externes Triggersignal (über die DLL-Funktion) zu verwenden.

Theoretisch können Triggersignale auch parallel ausgelöst werden (intern + extern). Dies verbraucht aber unnötig Rechenkapazität und führt zu einer schlechteren Synchronisierung.

Deaktivieren Sie deshalb vorab die internen Trigger mit der DLL-Funktion.

Diese Funktion wirkt nur auf interne Trigger vom Typ "laufend" und "Timer" und deaktiviert sich selbst wieder wenn das Konfigurationsmenü gestartet wird.

Beispiel:

In der Messschleife löst das Master-Programm zunächst einen DLL-Trigger für alle Messungen aus. Damit werden alle Bildverarbeitungen einmalig und zum gleichen Zeitpunkt ausgeführt. Nach dem Triggersignal werden die gewünschten Messwerte einzeln ausgelesen.

```
set_trigger_off_bvmodul(1, 0);
do
{
    trigger_bvmodul(1, 0, 1);
    read_main_value_bvmodul(1, 1, wert1);
    read_main_value_bvmodul(1, 2, wert2);
    read_main_value_bvmodul(1, 3, wert3);
    ....
}while(...);
set_trigger_on_bvmodul(1, 0);
close_bvmodul(0);
```

## Übersicht der DLL-Funktionen:

```
//verschiedene Varianten um ein BV-Modul zu starten/öffnen
//ohne Dateivorgabe wird "bvmodul_1.cfg" (.... "bvmodul_10.cfg") verwendet
//Rückgabewert:
// 1 = Modul erfolgreich gestartet
// -1 = Modul existiert bereits (kann nicht 2x gestartet werden!)
// -2 = ungültiger Modulindex
extern "C" MTKBVLIBRARY_API int start_bvmodul(int modulindex);
//Modulindex + Konfigurationsdateiname (Unicode)
extern "C" MTKBVLIBRARY_API int start_bvmodulW(int modulindex, wchar_t file[256]);
//mit Modulindex + Konfigurationsdateiname (ascii)
extern "C" MTKBVLIBRARY_API int start_bvmodulA(int modulindex, char file[256]);
//schließen/beenden
//0=alle schließen oder mit gültigem Modulindex (1..10)
extern "C" MTKBVLIBRARY_API void close_bvmodul(int modulindex);

//Spezialfunktionen:
//Abfragefunktion: Modul bereits vorhanden? 0=irgendein Modul 1..10 Prüfe auf dieses Modul
//0 = Rückgabewert ist der Index vom ersten gefundenen Modul (1..10) oder "0" wenn Kein Modul gefunden
//Rückgabewert:
// 0 = Modul ist nicht gestartet
// 1 = Modul ist gestartet
```

---

```

extern "C" MTKBVLIBRARY_API int find_bvmodul(int modulindex);

// Sprache voreinstellen (Diese Funktion hat Priorität gegenüber der Sprachauswahl im Konfigurationsmenü)
// SetLanguage_bvmodul() vor start_bvmodul() aufrufen, um die Programmsprache für alle Module vorab zu definieren
// 0=Deutsch 1=Englisch
extern "C" MTKBVLIBRARY_API void SetLanguage_bvmodul(int mode);

//Nach dem "START" eines BV-Moduls befindet dieses noch im "STOP" Zustand! D.h. das Livebild wird angezeigt, es erfolgt aber keine Reaktion auf Triggersignale!
//Zwischen "STOP" und "RUN" Zustand kann im Dialogfenster oder über die folgenden Befehle gewechselt werden:
//Rückgabewert:
// 1 = OK
// -2 = ungültiger Modulindex
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int run_bvmodul(int modulindex);
extern "C" MTKBVLIBRARY_API int stop_bvmodul(int modulindex);

//Kamerabild wird zunächst immer "live" dargestellt, kann über diese Kommandos "eingefroren/angehalten" werden (nicht verfügbar für Modus: BMP und SCREEN).
//Rückgabewert:
// 1 = OK
// -2 = ungültiger Modulindex
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int live_bvmodul(int modulindex);
extern "C" MTKBVLIBRARY_API int freeze_bvmodul(int modulindex);
////////////////////////////////////
////////////////////////////////////
//Darstellung vom BV-Modul festlegen (unsichtbar/sichtbar)
//0=unsichtbar (HIDE) 1=sichtbar (RESTORE)
extern "C" MTKBVLIBRARY_API void ShowWindow_bvmodul(int modulindex, int mode);
Fenstergröße und -position ändern/festlegen (links/oben/Breite/Höhe)
extern "C" MTKBVLIBRARY_API void SetWindowPos_bvmodul(int modulindex, int x, int y, int cx, int cy);
//Darstellung vom BV-Modul festlegen (Anordnung im Hintergrund, Vordergrund...)
// 0 = im Hintergrund (nicht aktiviert) 1=Überlappend 2=immer im Vordergrund (topmost)
extern "C" MTKBVLIBRARY_API void SetWindowStyle_bvmodul(int modulindex, int mode);

// BV-Modul Trigger für eine Messwerterfassung auslösen -> warte bis Messung fertig ja/nein
// Übergabewerte:
// modulindex= 1,2...10 Kamera/Modulnummer
// mssindex= 0 //es wird ein Trigger für die Messung von "allen" aktivierten Messungen ausgelöst
// mssindex= 1,2....50 //es wird ein Trigger für eine spezifische Messung ausgelöst
// wait= true // Funktion wird erst beendet, wenn Messung fertig
// wait= false // Funktion wird sofort beendet
//Rückgabewert:
// 0 = kein aktivierter Messwert vorhanden
// 1 = OK
// -1 = allg. Fehler
// -2 = timeout Meldung
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int trigger_bvmodul(int modulindex, int mssindex, bool wait);

////////////////////////////////////
////////////////////////////////////
// BV-Modul programmierte interne Trigger deaktivieren/aktivieren
// Wird das BV-Modul von extern getriggert (siehe Funktion: trigger_bvmodul()) sind interne Trigger evtl. ungünstig,
// weil schlechte Synchronisierung und unnötig hohe Rechenlast vom Modul.
// Betrifft NUR die Triggertypen: Echtzeit (real) und Timer
//
// Übergabewerte:
// modulindex= 1,2...10 //Kamera/Modulnummer
// mssindex= 0 //es werden alle internen real und timer - Trigger on/off gesetzt
// mssindex= 1,2....50 //es werden die Trigger für eine spezifische Messung on/off gesetzt
//Rückgabewert:
// 0 = Fehler
// 1 = OK
// -1 = Kennwert ist ungültig
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int set_trigger_off_bvmodul(int modulindex, int mssindex);
extern "C" MTKBVLIBRARY_API int set_trigger_on_bvmodul(int modulindex, int mssindex);

//Abfragen eines HAUPT-Messwertes aus einem Modul Vorgabe: Modulindex & Messwertindex...
//Rückgabe: double Messwert
//Rückgabewert:
// 0 = Messung ist aktiviert/ getriggert aber noch kein Wert vorhanden. Eventuell ist im BV-Modul die Messung angehalten?
// 1 = OK
// -1 = Kennwert ist nicht aktiviert
// -2 = Fehler bei Messung
// -3 = sonstiger Fehler
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int read_main_value_bvmodul(int modulindex, int mssindex, double* value);

```

---

---

```

//Abfragen von "n" Sub-Ergebnissen (im Bereich von n= 1...49) von einer Messung aus einem Modul Vorgabe: Modulindex &
Messwertindex und Anzahl der zu lesenden double-Datenwerte
//Rückgabe: double Datenfeld[50]
//Rückgabewert:
// 0 = Messung ist aktiviert/ getriggert aber noch kein Wert vorhanden. Eventuell ist im BV-Modul die Messung ange-
halten?
// 1 = OK
//-1 = Kennwert ist nicht aktiviert
//-2 = Fehler bei Messung
//-3 = sonstiger Fehler
//-6 = Modulfehler
extern "C" MTKBVLIBRARY_API int read_sub_values_bvmodul(int modulindex, int mssindex, double values[], int n);

//Abfragen von Zyklus-Ergebnissen von einer Messung aus einem Modul Vorgabe: Modulindex & Messwertindex
//Rückgabe: Zeiger auf double Datenfeld[100]
//Rückgabewert:
//>=1...100 = Messung erfolgreich, Wert (value) kann verarbeitet werden. Der Rückgabewert entspricht Anzahl der gül-
tigen Werte im Datenfeld.
// 0 = Messung ist aktiviert/ getriggert aber noch kein Wert vorhanden. Eventuell ist im BV-Modul die
Messung angehalten?
// -1 = Programm befindet sich im Einrichtungsmodus
// -2 = Programm befindet sich im Konfigurationsmodus
// -3 = Messung ist nicht aktiviert
// -4 = Fehler bei der Messung
// -5 = un spez. Fehler
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int read_zyklus_values_bvmodul(int modulindex, int mssindex, double values[]);

//Abfragen Referenzpunkt aus einem Modul Vorgabe: Modulindex
//Rückgabe: 3 double Werte (x,y,Drehwinkel)
//Rückgabewert:
// 0 = Messung ist aktiviert/ getriggert aber noch kein Wert vorhanden. Eventuell ist im BV-Modul die Messung ange-
halten?
// 1 = OK
//-1 = Kennwert ist nicht aktiviert
//-2 = Fehler bei Messung
//-3 = sonstiger Fehler
//-6 = Modulfehler
extern "C" MTKBVLIBRARY_API int read_ref_values_bvmodul(int modulindex, double* ref_x, double* ref_y, double* ref_w);

//Abfragen Status von einer Messung (aktiviert?) sowie die interne Bezeichnung für diese Messung
//Rückgabewert:
// 0 = Messung ist deaktiviert
// 1 = Messung ist aktiviert
//-6 = Modulfehler
extern "C" MTKBVLIBRARY_API int read_mss_status_bvmodul(int modulindex, int mssindex, wchar_t txt[]);

// Das aktuelle Bild wird unter den vorgegebenen Namen/Pfad als bitmap abgespeichert
// - vorhandene Dateien werden ersetzt, noch nicht vorhandene Dateien erzeugt (Pfad muss vorhanden sein!)
// - bei Dateiname (ohne Pfadangabe) wird die Bilddatei in das vordefinierte Bildverzeichnis abgelegt
// Übergabewerte:
// modulindex= 1,2...10 Kamera/Modulnummer
// wait=true Die Funktion wird erst dann beendet, wenn fertig ausgeführt.
//Rückgabewert:
// 1 = OK
// -1 = allg. Fehler
// -6 = Modulfehler
extern "C" MTKBVLIBRARY_API int save_bmp_bvmodulW(int modulindex, wchar_t file[2048], bool wait); //UNICODE Version
extern "C" MTKBVLIBRARY_API int save_bmp_bvmodulA(int modulindex, char file[2048], bool wait); //ASCII Version

```

---