

# MR-E-2 Development Kit



## Operation Manual

Copyright ©2019 Optotune Switzerland AG. All Rights Reserved. The MR-E-2 development kit hardware and corresponding software Optotune Cockpit shall only be used in connection with the evaluation of the MR-15-30 and MR-10-30 mirror product families. Any other use is not permitted without the prior written authorization of Optotune Switzerland AG.

IN NO EVENT SHALL OPTOTUNE BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS MIRROR DRIVER MR-E-2 DEVELOPMENT KIT AND ITS DOCUMENTATION, EVEN IF OPTOTUNE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

OPTOTUNE SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE MIRROR DRIVER MR-E-2 DEVELOPMENT KIT AND ACCOMPANYING DOCUMENTATION, IF ANY, PROVIDED HEREUNDER IS PROVIDED "AS IS". OPTOTUNE HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS

## Table of Contents

1	Overview .....	5
2	Software and System requirements.....	6
3	Mechanical Details .....	7
3.1	Base Unit.....	7
3.2	Head Unit.....	7
3.3	Heatsink.....	7
4	Hardware Operation .....	9
4.1	Package Contents and Description .....	9
4.2	Power supply specifications .....	9
4.3	Hardware Connections .....	10
5	Mirror Coordinate System .....	12
5.1	Definition of the XY Coordinate System .....	12
5.2	Transformation Between Different Cartesian Projection Coordinates .....	13
5.3	Euler angles .....	16
5.4	Transformation to and from Spherical Coordinates.....	17
6	Amplitude control .....	18
6.1	Motivation .....	18
6.2	Implementation .....	19
6.3	Safety shutdown .....	19
7	Optotune Cockpit.....	20
7.1	Installation of Optotune Cockpit .....	20
7.2	MR-E-2 Connection.....	20
7.3	Mirror Control .....	21
7.4	Factory Reset .....	22
7.5	Examples.....	23
7.5.1	Closed loop static operation .....	23
7.5.2	Mixed Mode Waveform Operation .....	24
8	Simple Serial Communication Mode.....	26
8.1	Installation of a Serial Communication Terminal .....	26
8.2	Step-by-Step procedure.....	26
8.3	List of commands available .....	28
9	Analog Mode.....	30
9.1	Input Pins and Signal Levels.....	30
10	SDKs.....	31
11	SPI Interface .....	31
11.1	Interface .....	31
11.2	Framing.....	32

---

11.3	Timing and synchronization.....	32
11.4	Data Structure .....	32
11.4.1	Master Out, Slave In (MOSI) .....	32
11.4.2	Master In, Slave Out (MISO) .....	33
11.5	Examples.....	33
11.5.1	Setting open loop to both axis and reading back X Y.....	33
11.5.2	Setting closed loop triangular to X axis and open loop sinusoidal to Y axis .....	34
11.5.3	Activate analogue mode .....	35

## 1 Overview

The MR-E-2 is a fully integrated driving solution for the Optotune MR-series beam steering mirrors. It provides access to the full functionality of the mirrors, including open and closed loop control. The user can control the mirror and perform operations in various modes such as waveform operation or static pointing. This document serves as an operation manual for the mirror driver.

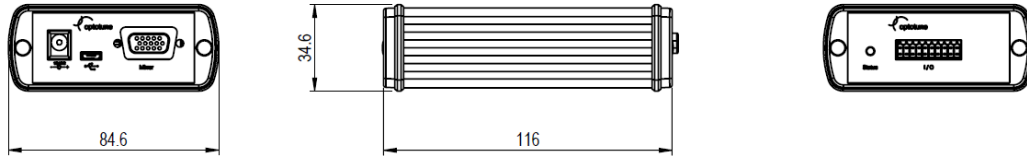
## 2 Software and System requirements

- Windows 10 or later
- USB Driver: USB 2.0 port
- Optotune Cockpit Control Software (<https://www.optotune.com/registration-for-software-download>)
- Optional: Serial communication terminal software such as Windows Hyper-Terminal (only required for simple serial communication mode)

## 3 Mechanical Details

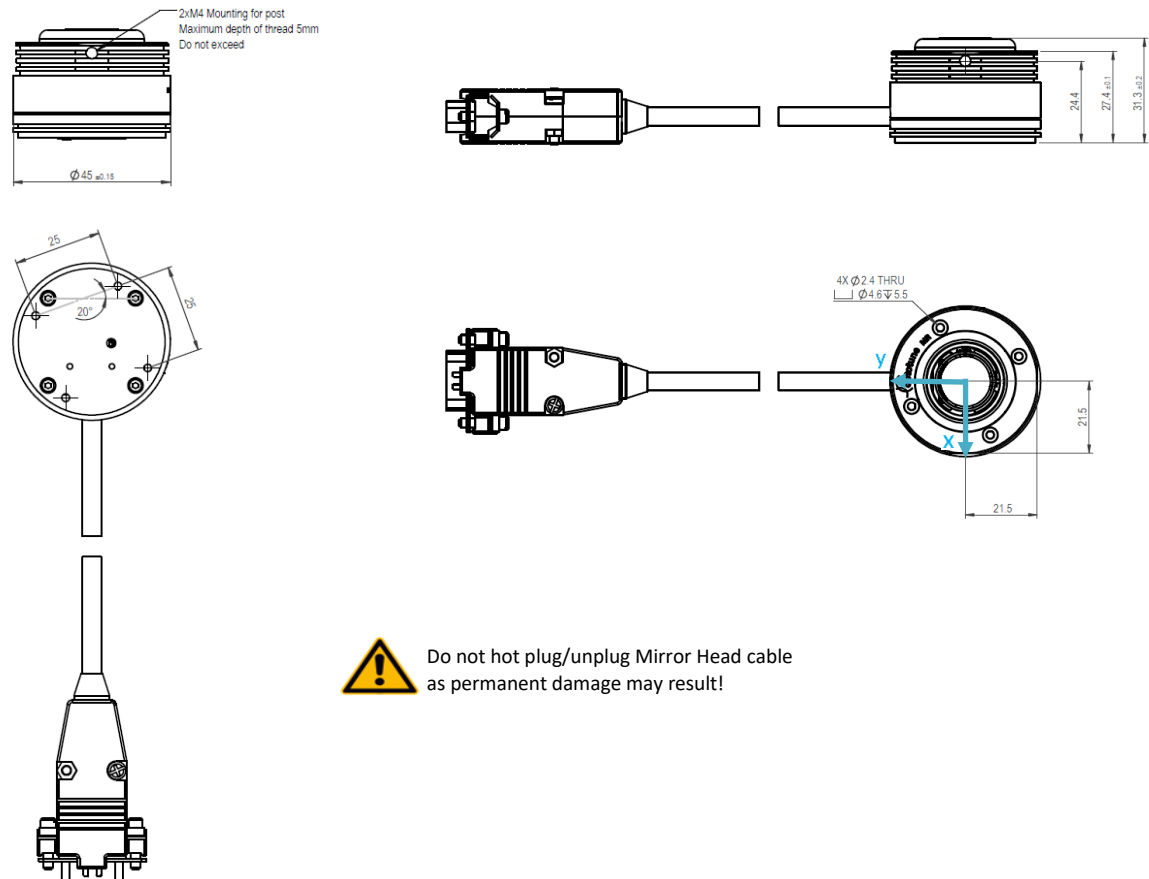
All numbers in the mechanical drawings are in millimeters.

### 3.1 Base Unit



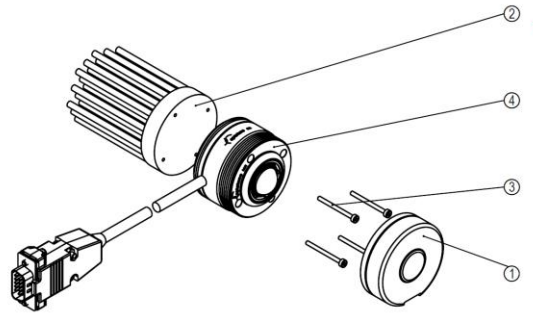
### 3.2 Head Unit

The two mounting holes on the mirror head are M4 tapped.

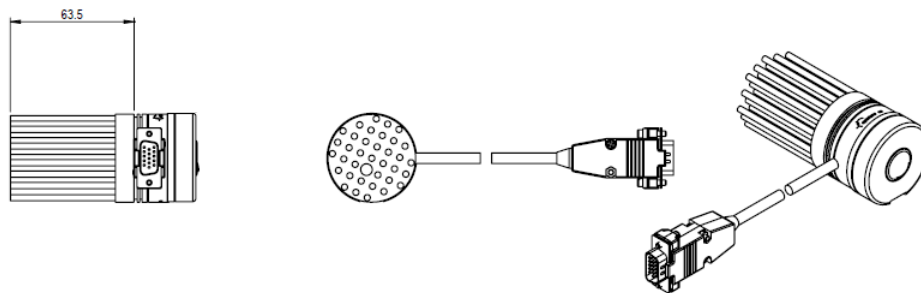


### 3.3 Heatsink

Optotune provides a heatsink with each mirror head. Depending on the operation mode, the mirror head by itself may not be able to dissipate the heat generated by the driver and may shut off due to overheating. In this case, either mount the mirror head flat on a large enough heatsink or attach the heat sink supplied by Optotune.



Attach the heatsink (2) to the mirror head (4) by inserting the M2 mounting screws (3) into the through-holes on the mirror head and tightening them. A protective cap (1) can protect the mirror surface when not in operation.



---

**Warning:** Do not hot plug/unplug cable between Base Unit and Head Unit, as permanent damage may result!

---

---

**Warning:** During operation the mirror head can become hot (up to 85°C). Avoid touching the mirror head during operation and ensure not to place heat-sensitive equipment in the immediate vicinity of the mirror head.

---



## 4 Hardware Operation

### 4.1 Package Contents and Description



The figure above shows all parts contained in the MR-E-2 Dev Kit package. The driver consists of a base unit with control electronics and connectors in a protective enclosure. A cable connects the base unit to the mirror head containing the analog driving electronics. Also included is a power supply and a USB cable. The included push-in type carrier board I/O Connector facilitates wiring for SPI, Analog and UART operation.

**Important:** Depending on how the mirror operates, the mirror head by itself may not be able to fully dissipate the generated heat into the surrounding air and may shut off due to overheating. Ensure proper heat-dissipation by using the heatsink supplied by Optotune or mount the head on a large enough heat-sink with good thermal contact.

**Note:** Strong external magnetic fields may cause offsets in current (in closed-loop) or position (in open-loop).

### 4.2 Power supply specifications

The driver hardware includes a power supply. Specifications for the power supply are:

Specifications	Value	Units
Minimum Output Voltage (Vdc)	15	Vdc
Maximum Output Voltage (Vdc)	28	Vdc
Minimum Output Current (A)	2	A
Minimum Output Power (W)	20	W

The model number of the power supply is PSAA30R-150, a replacement can be purchased directly from Optotune.

## 4.3 Hardware Connections

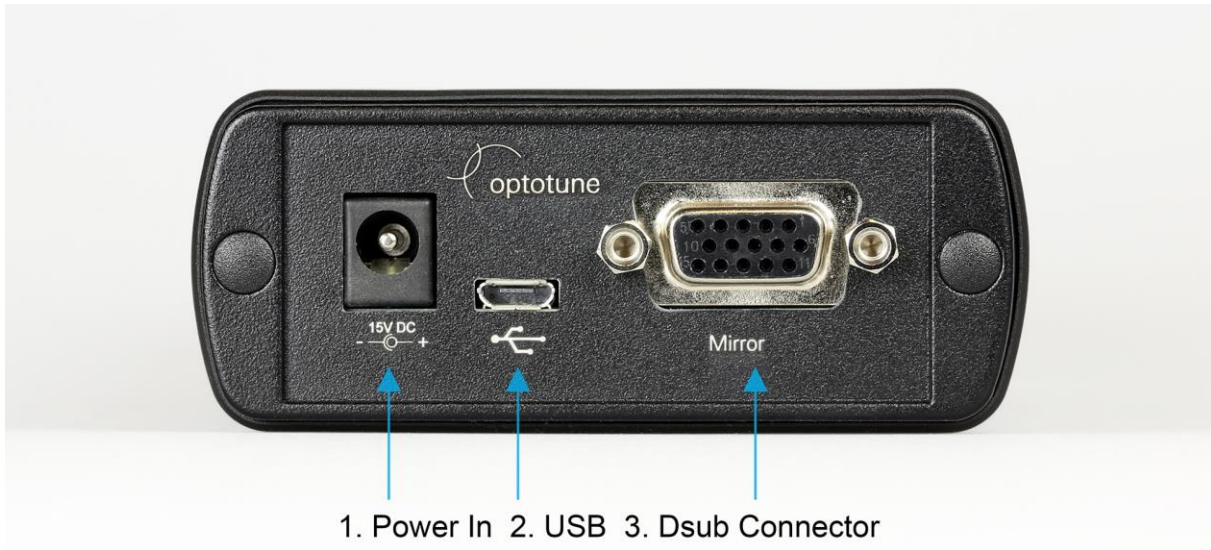


Figure 1: Back panel with connectors for power, USB and mirror head.

The back panel of the MR-E-2 base unit has three connectors. In addition to the power connector, it features a USB type B connector allowing control through Optotune Cockpit software and for simple serial mode operation. The mirror head is connected to the base unit on the D-sub connector labeled **Mirror**. **Use the fixing screws on the D-sub connector to secure the connection.**

**WARNING:** Do not connect or disconnect the mirror head cable while the base unit is connected to power. This will damage the driver and void warranty!  
Use the fixing screws to secure the Dsub connection.

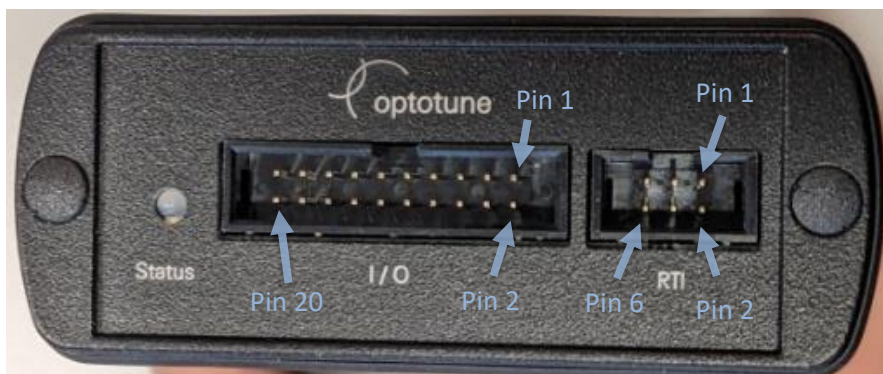


Figure 2: I/O connections and Real-Time-Interface (RTI)

The main connector on the front panel has the I/O connections. The pin-out is given in Table 1: Odd numbered pins are in the top row, even numbered pins in the bottom row. The status LED reflects the state of the driver. Red light indicates an active error or warning. Green light indicates that there is no active error.

The RTI is a Low-Voltage Differential Signaling (LVDS) interface for position readout with low temporal jitter. The interface requires an FPGA to read low-level sensor signals. The RTI is intended for applications requiring dense point clouds at high speed such as in LiDAR.

<i>Pin# in I/O Connector</i>	<i>Signal</i>	<i>Description</i>
1	UART_RX	UART Receive line
2	GND	Circuit ground
3	ERROR	Indicates an active error
4	UART_TX	UART Transmit line
5	-	Reserved
6	TRIGGER	Trigger input <sup>1</sup>
7	-	Reserved
8	STABILITY	Mirror not stable <sup>2</sup>
9	-	Reserved
10	-	Reserved
11	GND	Circuit ground
12	SPI_DATA_NRDY	SPI Data Not Ready
13	SPI_NSS	SPI Negative Slave Select
14	SPI_MISO	SPI Master Input Slave Output
15	SPI_MOSI	SPI Master Output Slave Input
16	SPI_CLK	SPI Clock
17	GND	Circuit ground
18	GND	Circuit ground
19	AI_X	Analog Input X axis
20	AI_Y	Analog Input for Y axis

Table 1: I/O Connector pin-out

<i>Pin# in RTI Connector</i>	<i>Signal</i>	<i>Description</i>
1	-	Reserved
2	CS_SYNC	Position readout trigger
3	SCLK_LVDS	LVDS Clock
4	FSI	Frame start trigger
5	MISO_LVDS	LVDS Master Input Slave Output
6	GND	Circuit ground

Table 2: RTI connector pin-out

<sup>1</sup> Trigger input for 3.3 V CMOS logics to synchronize signal generator or vector pattern unit with external signal.

<sup>2</sup> According to user defined stability criterion. See Firmware manual.

## 5 Mirror Coordinate System

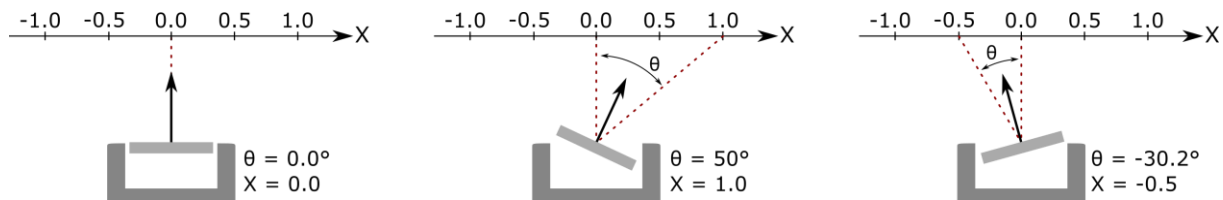
In the following, the mirror coordinate system is introduced and different coordinate transformations are described. An example Python script that implements these transformations can be downloaded [here](#).

### 5.1 Definition of the XY Coordinate System

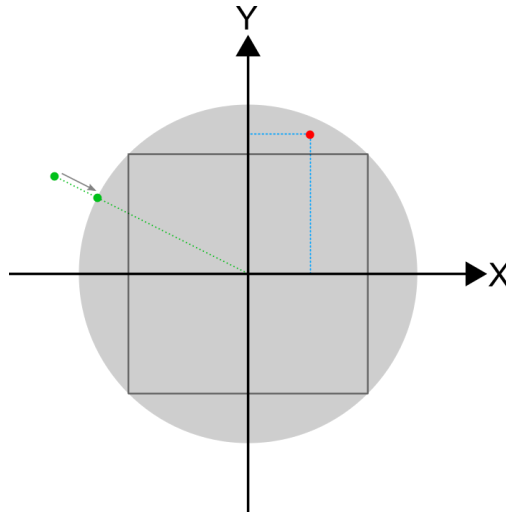
We define a coordinate system to calibrate the internal optical feedback mechanism and relate it with physical mirror position. This coordinate system is a cartesian coordinate system with axes X and Y. The X-axis is perpendicular to the cable protruding from the mirror head and the Y-axis is parallel to this cable. The numerical values on the axes are unitless and defined via the maximum deflection of the mirror in optical angles, i.e. 50°. Along each axis, a deflection of +50° corresponds to a value of +1 and a deflection of -50° corresponds to a value of -1. This corresponds to the projection observed on a screen if the mirror reflects a laser beam travelling along the z-axis, incident on its center. For example, the analytical relationship between deflection angle  $\theta$  and coordinate system value  $x$  along the x-Axis is:

$$x = \frac{\tan(\theta)}{\tan(50^\circ)}$$

The figure below illustrates the relationship between deflection on-axis and coordinate system value using three examples.



The mirror has a maximum deflection of 25° in every direction. Therefore, in the XY coordinate system a unit circle with radius 1 contains all accessible values, as shown in the figure below.

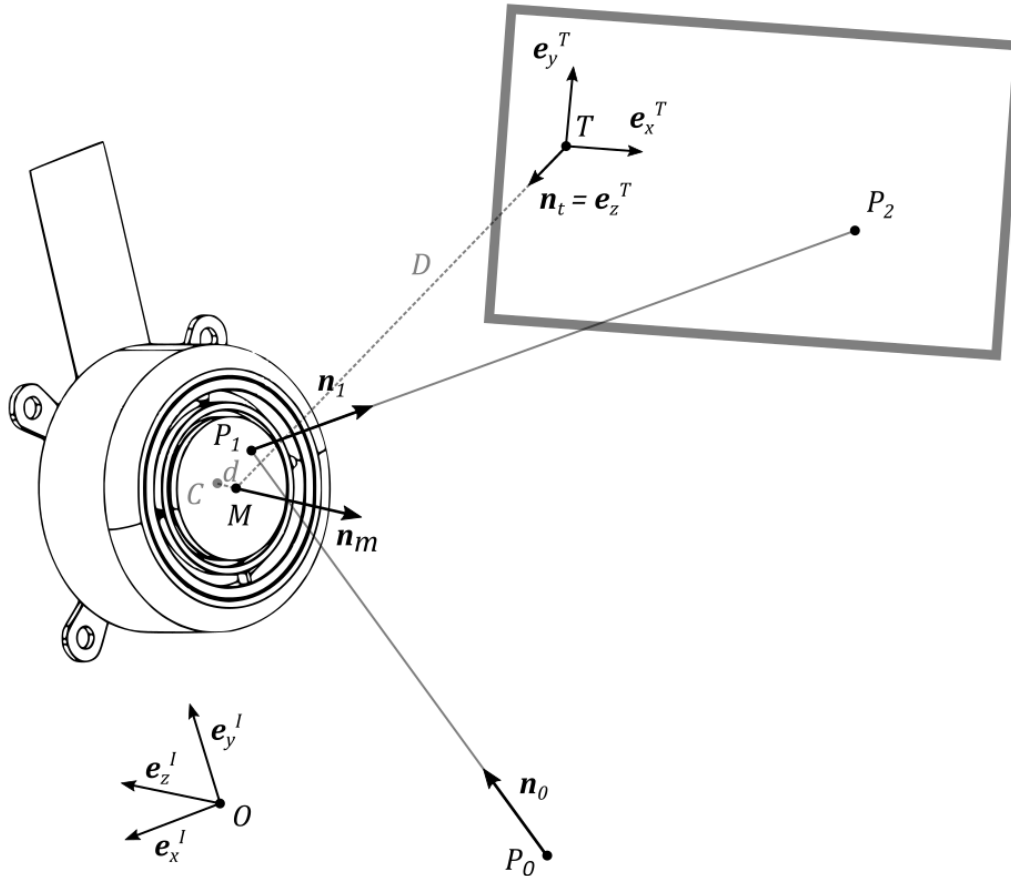


The mirror can access every possible combination of X and Y values for both  $X$  and  $Y < 0.7$  (black square in the figure above). If one of the XY values exceeds 0.7, the other value must decrease, so that at any time  $X^2 + Y^2 \leq 1$  (the red dot in the figure above is an example). The firmware automatically reduces XY positions outside the accessible unit circle by moving them to the nearest edge of the unit circle (green dots in the figure above). This behavior prevents mirror and driver damage.

When driving the mirror in open loop mode, it is possible to reach angles larger than 25°, which correspond to XY values bigger than 1. However, these angles are outside the guaranteed and calibrated range of the mirror and calibration accuracy can vary significantly.

## 5.2 Transformation Between Different Cartesian Projection Coordinates

When using the mirror in a scanning system, typically the arrangement with  $0^\circ$  AOI is not practical, since incident and reflected beam paths would overlap. The figure below shows the general case: The mirror with its middle point  $M$  and normal vector  $\mathbf{n}_m$  rotates around the center of rotation  $C$  and reflects an incoming beam, specified by a point  $P_0$  and a unit vector  $\mathbf{n}_0$ . The reflected beam starts at point  $P_1$  and has the direction of the unit vector  $\mathbf{n}_1$ . Finally, the reflected beam hits a target plane at the point  $P_2$ , where the target plane is located at a distance  $D$  from the undeflected mirror center and has the normal vector  $\mathbf{n}_t$ .



This general arrangement captures distortion effects due to

- AOI of the incoming beam
- Rotated target plane
- Off-centered incoming beam
- Distant center of rotation

The vector analysis to calculate the beam path is straightforward. First, inserting the equation for the incoming beam  $\mathbf{r} = \mathbf{r}_{OP_0} + t \cdot \mathbf{n}_0$ ,  $t \in \mathbb{R}$  into the equation for the mirror plane  $(\mathbf{r} - \mathbf{r}_{OM}) \cdot \mathbf{n}_m = 0$ , yields the intersection point  $P_1$

$$\mathbf{r}_{OP_1} = \mathbf{r}_{OP_0} + t_1 \cdot \mathbf{n}_0, \quad \text{where} \quad t_1 = \frac{(\mathbf{r}_{OM} - \mathbf{r}_{OP_0}) \cdot \mathbf{n}_m}{\mathbf{n}_0 \cdot \mathbf{n}_m}.$$

Note that  $\mathbf{r}_{OM} = \mathbf{r}_{OC} + d \cdot \mathbf{n}_m$ .

Then, the reflected beam is obtained by applying the law of reflection

$$\mathbf{n}_1 = \mathbf{n}_0 - 2 \cdot (\mathbf{n}_0 \cdot \mathbf{n}_m) \cdot \mathbf{n}_m.$$

Finally, we calculate the intersection point with the target plane  $P_2$

$$\mathbf{r}_{OP_2} = \mathbf{r}_{OP_1} + t_2 \cdot \mathbf{n}_1, \quad \text{where} \quad t_2 = \frac{(\mathbf{r}_{OT} - \mathbf{r}_{OP_1}) \cdot \mathbf{n}_t}{\mathbf{n}_1 \cdot \mathbf{n}_t}.$$

We can now express the vector  $\mathbf{r}_{TP_2} = \mathbf{r}_{OP_2} - \mathbf{r}_{OT}$  in target plane coordinates

$${}_T\mathbf{r}_{TP_2} = \mathbf{A}_{TI} \mathbf{r}_{TP_2} = \begin{bmatrix} x_t \\ y_t \\ 0 \end{bmatrix},$$

where  $\mathbf{A}_{TI} = \mathbf{A}_{IT}^{-1} = \mathbf{A}_{IT}^T$  is the orthogonal transformation matrix between the frames of reference  $I$  and  $T$ .

In a simplified case, for an incoming beam hitting the mirror centered and  $d$  assumed to be zero, we have  $P_1 = M = C$  and one can explicitly calculate the mirror orientation from projected coordinates.

$$\mathbf{n}_m = \frac{\mathbf{n}_1 - \mathbf{n}_0}{\|\mathbf{n}_1 - \mathbf{n}_0\|}, \quad \text{where} \quad \mathbf{n}_1 = \frac{\mathbf{r}_{MP_2}}{\|\mathbf{r}_{MP_2}\|} \quad \text{and} \quad \mathbf{r}_{MP_2} = \mathbf{r}_{MT} + \mathbf{r}_{TP_2}.$$

Note that this simplification still captures the distortion introduced by the AOI of the incoming beam, which is by far the most important one to consider.

For convenience, in the following, the origin  $O$  is placed at the undeflected mirror center:  $O = P_1 = M = C$ .

The above equations can be used to transform between normalized mirror coordinates  $(x, y)$  and target plane coordinates:

$(x, y) \rightarrow \mathbf{n}_m \rightarrow (x_t, y_t)$ :

1. By definition of the mirror coordinates set  ${}_I\mathbf{n}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$  and  ${}_I\mathbf{r}_{MP_2} = \begin{bmatrix} x D \tan 50^\circ \\ y D \tan 50^\circ \\ -D \end{bmatrix}$ . So,  ${}_I\mathbf{n}_1$  is obtained by normalizing  $\begin{bmatrix} x \\ y \\ -1/\tan 50^\circ \end{bmatrix}$  and the mirror normal by calculating  $\mathbf{n}_m = \frac{\mathbf{n}_1 - \mathbf{n}_0}{\|\mathbf{n}_1 - \mathbf{n}_0\|}$ .
2. Redefine the actual incoming beam  $\mathbf{n}_0$  and target plane ( $\mathbf{A}_{TI}$  and  $D$ ). Using the known mirror normal  $\mathbf{n}_m$  calculate  ${}_T\mathbf{r}_{TP_2}$  using the above equations and extract  $(x_t, y_t)$ , i.e. the first two components of  ${}_T\mathbf{r}_{TP_2}$ .

$(x_t, y_t) \rightarrow \mathbf{n}_m \rightarrow (x, y)$ :

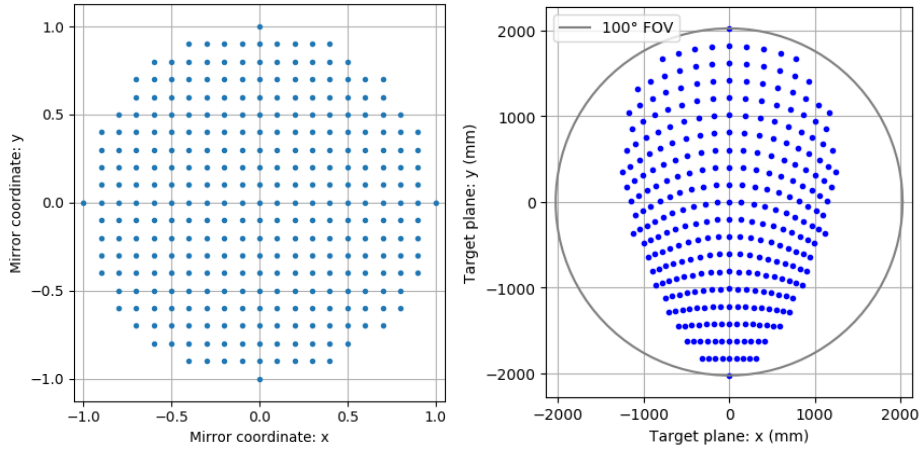
1. Specify the actual incoming beam  $\mathbf{n}_0$  and target plane ( $\mathbf{A}_{TI}$  and  $D$ ). Calculate  ${}_I\mathbf{r}_{MP_2} = \mathbf{A}_{TI}^T \begin{bmatrix} x_t \\ y_t \\ -D \end{bmatrix}$  and normalize to get  ${}_I\mathbf{n}_1$ , then obtain the mirror normal from  $\mathbf{n}_m = \frac{\mathbf{n}_1 - \mathbf{n}_0}{\|\mathbf{n}_1 - \mathbf{n}_0\|}$ .
2. By definition of the mirror coordinates redefine  ${}_I\mathbf{n}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$  and  $\mathbf{A}_{TI} = \mathbf{1}$ . Using the mirror normal  $\mathbf{n}_m$  from the previous step, calculate  ${}_I\mathbf{n}_1 = {}_I\mathbf{n}_0 - 2 \cdot ({}_I\mathbf{n}_0 \cdot {}_I\mathbf{n}_m) \cdot {}_I\mathbf{n}_m$ . Scale this vector with a constant factor to get the vector  $\begin{bmatrix} x \\ y \\ -1/\tan 50^\circ \end{bmatrix}$ , from which  $x$  and  $y$  can be extracted.

As an example, consider the following typical arrangement:

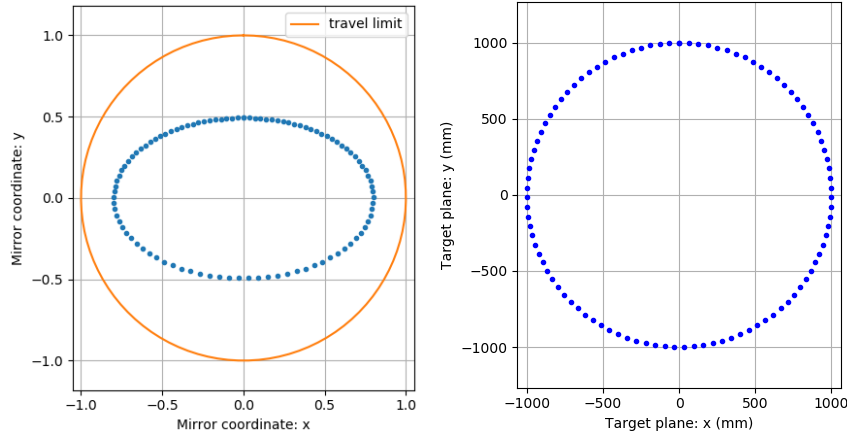
The incoming beam in the  $yz$ -plane hits the mirror centered at a  $45^\circ$  incidence angle. The target plane is placed perpendicular to the reflected beam for an undeflected mirror, i.e. when  $x = y = 0$ .

$${}^I\mathbf{n}_0 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}, {}^I\mathbf{r}_{OP_0} = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}, \mathbf{A}_{TI} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 45^\circ & -\sin 45^\circ \\ 0 & \sin 45^\circ & \cos 45^\circ \end{bmatrix}, D = 1700 \text{ mm}, d = 1.3 \text{ mm}$$

The below plots show the distortions introduced in this case. For reference, also the  $100^\circ$  FOV is shown, which the mirror would enable for  $0^\circ$  AOI.



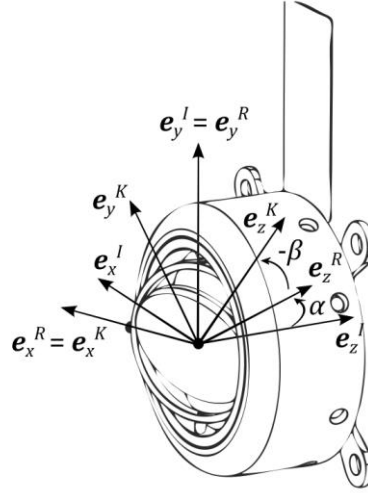
If we want to scan a circle with radius 1 m in the target plane, we can apply the transformation in the other direction to find the necessary mirror coordinates. The fact that the resulting points all lie within the travel limit, shows that this pattern is feasible.





### 5.3 Euler angles

When using Euler angles (sometimes called Cardan angles) to describe the orientation of the mirror, it is necessary to specify the sequence of rotations. One example of such a representation is depicted below. Three different coordinate systems are introduced. The inertial frame of reference  $I$ , the intermediate frame  $R$ , aligned with the outer gimbal ring, and the mirror-fixed frame  $K$ .



The inertial frame of reference  $I$  is rotated around its y-axis by the angle  $\alpha$ . Following that rotation, the resulting intermediate frame  $R$  is rotated around its x-axis by the angle  $-\beta$ . The negative sign accounts for the fact that the rotation shown here acts in the negative direction, as defined by the right-hand rule. Using the two transformation matrices

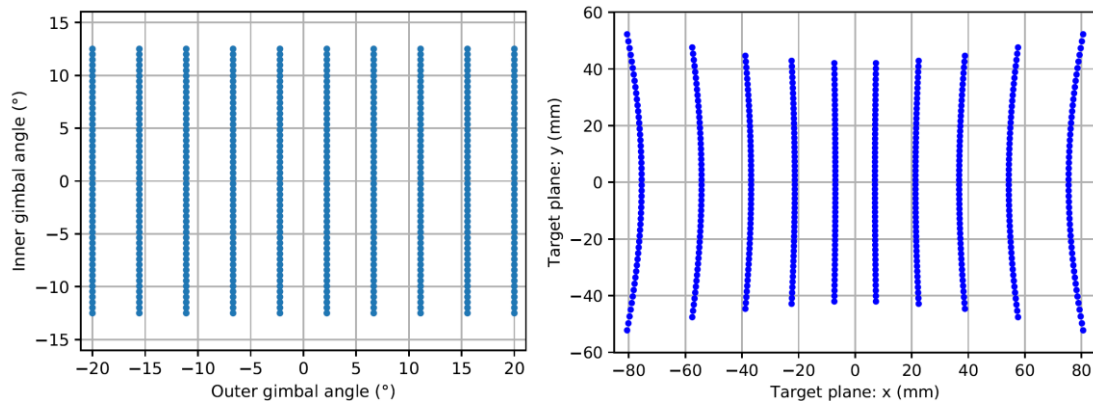
$$A_{IR} = \begin{bmatrix} \cos \alpha & 0 & \sin \alpha \\ 0 & 1 & 0 \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \text{ and } A_{RK} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \beta & -\sin \beta \\ 0 & \sin \beta & \cos \beta \end{bmatrix}$$

the mirror normal vector, expressed in the inertial frame of reference, can then be written as

$${}_I \mathbf{n}_m = -{}_I \mathbf{e}_z^K = -A_{IK} \mathbf{e}_z^K = -A_{IR} A_{RK} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -\sin \alpha \cos \beta \\ \sin \beta \\ -\cos \alpha \cos \beta \end{bmatrix},$$

which can be inserted into the above calculations to obtain projected coordinates.

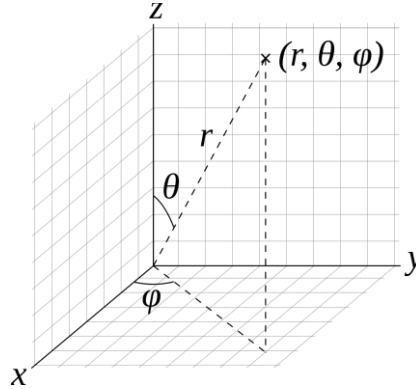
As an example, consider scanning the resonant axis of the MR-10-30 at full angular stroke at fixed outer gimbal angles in the range  $-20^\circ$  to  $20^\circ$ . The plot on right hand side shows the scanned trajectories on a screen placed perpendicular to the incoming beam ( $0^\circ$  AOI):





## 5.4 Transformation to and from Spherical Coordinates

It is also possible to express the mirror position by using spherical coordinates instead of the XY coordinates introduced above. Spherical coordinates are not available in the MR-E-2 firmware and software. In this coordinate system, two angles define the mirror position: A polar angle between the z-axis and the reflected beam, together with an azimuthal angle defined as the angle between the x-axis and the projection of the reflected beam onto the xy-plane.



The following equations transform XY coordinates into spherical coordinates:

$$C = \frac{1}{\tan(50^\circ)}$$

$$\varphi = \text{atan2}(y, x)$$

$$\theta = \arccos\left(\frac{C}{\sqrt{x^2 + y^2 + C^2}}\right)$$

The following equations transform spherical coordinates into XY coordinates:

$$C = \frac{1}{\tan(50^\circ)}$$

$$r = \frac{C}{\cos(\theta)}$$

$$x = r \sin(\theta) \cos(\varphi) = C \tan \theta \cos \varphi$$

$$y = r \sin(\theta) \sin(\varphi) = C \tan \theta \sin \varphi$$

Mechanical vs. optical angle:

If the coordinate system is defined with respect to the mirror's normal vector instead of the reflected beam, the angle  $\theta$  needs to be halved, whereas  $\varphi$  stays the same.

## 6 Amplitude control

This chapter only applies to the [MR-10-30](#) mirror that has a resonant axis. This chapter describes how the resonant axis can be driven reliably and at stable amplitude by an automatic adjustment of the driving frequency of the sinusoidal excitation.

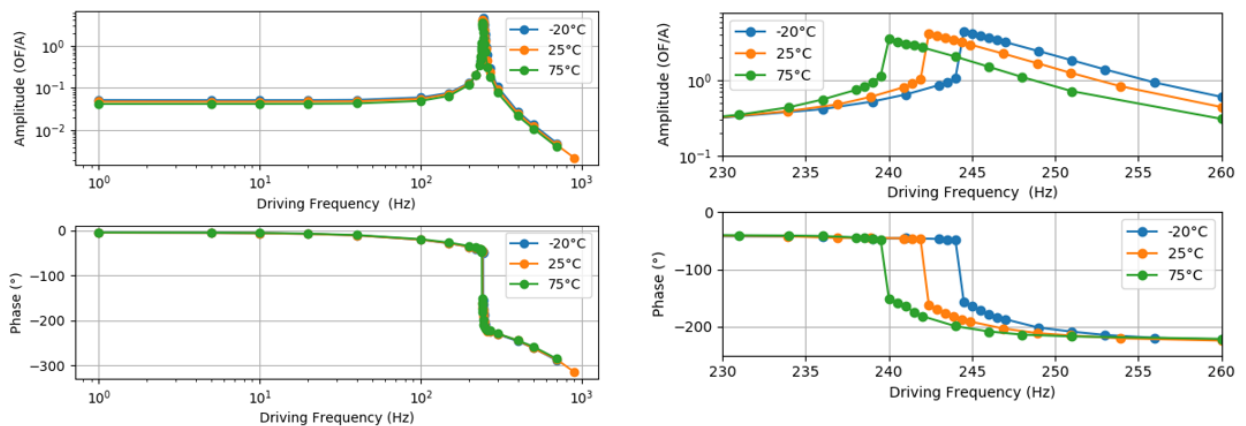
### 6.1 Motivation

Due to its large Q-factor of typically  $Q \approx 400$  the time constant of the resonant axis at  $f_0 = 250$  Hz is significant:

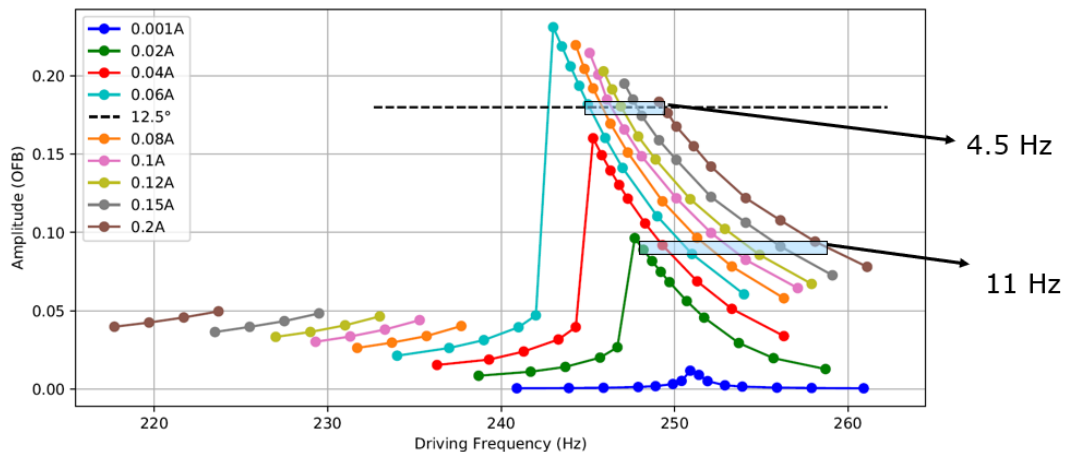
$$\tau = \frac{Q}{\pi f_0} = 0.5 \text{ s}$$

This means that the resonant axis cannot change amplitude quickly. Typically, it is therefore not possible to change the amplitude during a frame that lasts at the order of 100 ms. This means that the resonant axis angle  $\beta$  will be near-constant across a frame, if the frame lasts less than  $\tau$ , which is typically the case. It also means that the curved scan lines depicted in section **Error! Reference source not found.** are fundamental.

The controller should therefore focus on keeping the amplitude stable across different frames and for changing environmental conditions, namely temperature, which shifts resonance by about 5 Hz per 100 K:



There are two ways to compensate a shift in resonance. One is to use a higher nominal excitation current and then to reduce or increase the current as needed. This approach is limited to a few Hz of resonance frequency shift and increases power consumption.



The second approach changes the driving frequency and hence allows to operate always close to resonance with lowest power consumption. This approach is implemented in MR-E-2 Firmware.

## 6.2 Implementation

Amplitude control follows these steps:

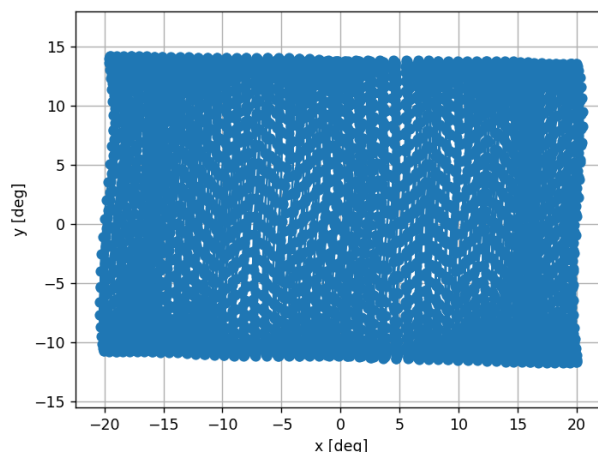
1. Controller measures natural frequency at power up of the controller.
2. User sets desired FOV (or desired mechanical scan amplitude) and activates amplitude control.
3. Suitable driving frequency and driving current are chosen based on the desired FOV and the natural frequency of the MR-10-30.
4. Resonant axis amplitude (Euler angle  $\beta$ ) is periodically measured and the drive frequency is updated according to:

$$f_{\text{drive}} = f_{\text{drive}} - k \cdot \frac{A_{\text{meas}} - A_{\text{set}}}{S}$$

The firmware system 0x35 lists all the parameters for amplitude control in detail. Here a only the

- Enable amplitude control (register 0x3500):  
Activates amplitude control. All parameters need to be configured prior to activation. During operation changing parameters of the algorithm is not possible.
- Desired amplitude or desired FOV (registers 0x3505 and 0x3507):  
Desired amplitude. FOV equals 4x the mechanical amplitude. Only one of these registers needs to be specified.
- Frequency shift, 0x3509:  
Defines starting value for drive frequency above resonance frequency. Use higher value to smoothly ramp-up to the target amplitude. For FOVs close to the maximum this can avoid triggering the safety-shutdown.
- Resonance slope  $S$  and control gain  $k$  (registers 0x350C and 0x350A):  
Tune these to get a good compromise between smoothness of scan and convergence speed to target amplitude.
- Time window, averaging cycles, and control cycles (registers 0x350D, 0x350E, 0x350F), affect the timing of the control update rate:  
control timing = Time window \* Averaging cycles \* Control cycles  
The control timing interval should be larger than 1s.

Example code for amplitude control using the Python SDK can be shared upon request. A resulting scan in Euler angles looks as follows. Note that in the scan below there is a residual offset angle in the resonant axis:



## 6.3 Safety shutdown

The amplitude measurement for the resonant axis is also active when amplitude control is disabled. This makes it possible to zero the output current in case an excessive amplitude was detected, which could overstress or permanently damage the resonant axis. To benefit from this safety feature, users are strongly advised to use firmware version 2.7 or later.

## 7 Optotune Cockpit

This section provides a short overview of the Optotune Cockpit software. It describes the main functions to control the mirror. For more information on Optotune Cockpit, please check the dedicated manual, when downloading the software.

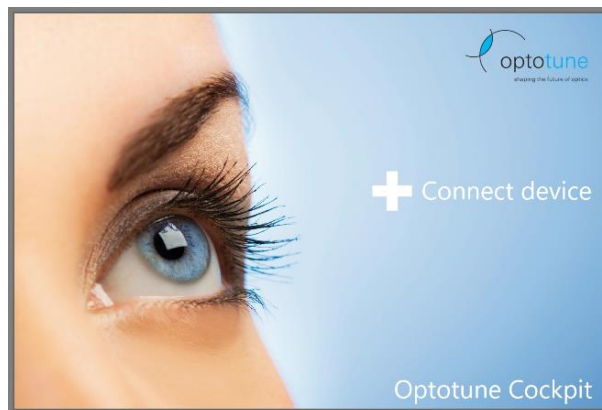
### 7.1 Installation of Optotune Cockpit

Download the Optotune Cockpit Installer from the Optotune website (<https://www.optotune.com/registration-for-software-download>). Open the installer file and follow the instructions on the screen.

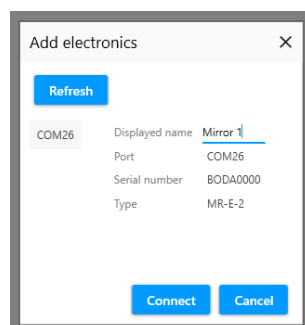


### 7.2 MR-E-2 Connection

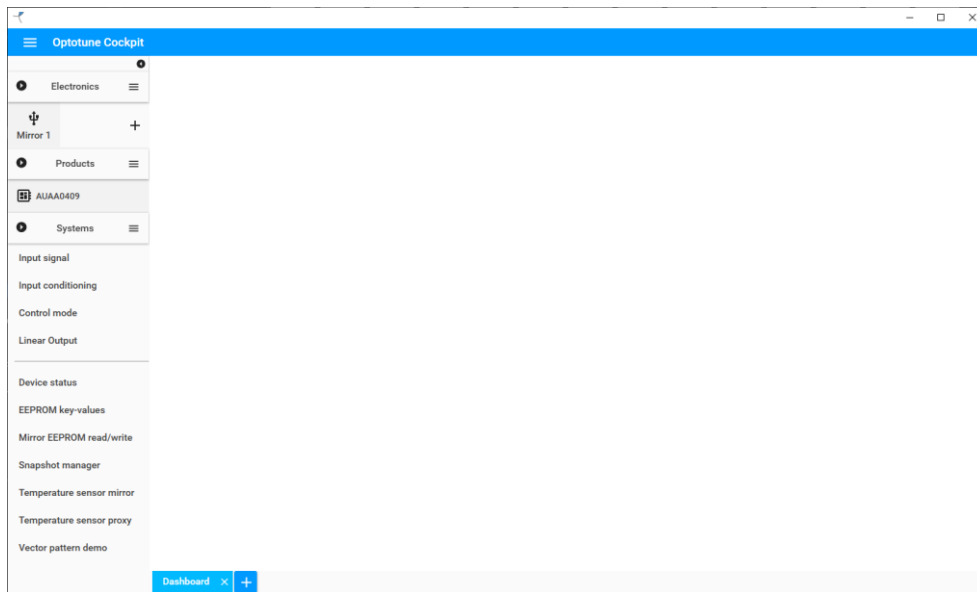
Make sure that the MR-E-2 is connected via USB to the computer and powered. Open Optotune Cockpit and click on *Connect Device*.



A window will appear, allowing you to select the device. You can also change the displayed name of the device for this session. This makes it easy to control several devices with the same computer.

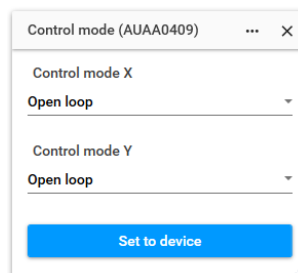


After clicking connect, the device will appear on the left side of the Optotune Cockpit window. You can add additional devices at any time by clicking + and repeating the steps above.

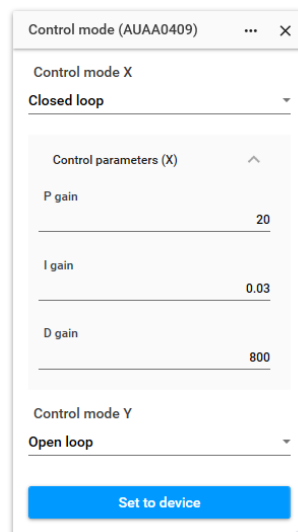


### 7.3 Mirror Control

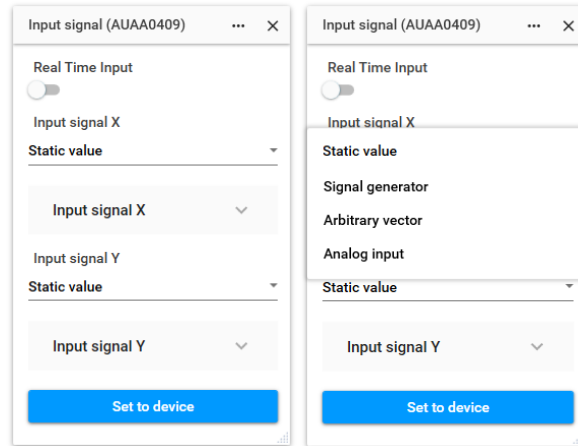
The two main panels for controlling the mirror are *Input Signal* and *Control Mode*. In the *Control Mode* window, you can choose to operate the mirror in closed loop or open loop via the drop-down menus. The X and Y axes are independent and can run with different control modes. After choosing the desired modes, click *Set to device* to send the settings to the driver.



Advanced users can also set the P, I and D gain of the closed-loop PID controller. We recommend keeping the standard values optimized by Optotune, unless for a specific application tuning is necessary. Contact Optotune support if you believe that the standard PID control parameters are not suitable for your application and you are unsure how to tune them.



Having set the control modes for both axes, you can then use the *Input Signal* panel to start controlling the mirror. There are two ways to control the mirror through this Panel, either with *Real Time Input* on or off. If *Real Time Input* is set to on, any changes in the panel are transferred directly to the driver and take effect immediately. If it is turned off, you need to press *Set to device* after making changes for them to be send to the driver.



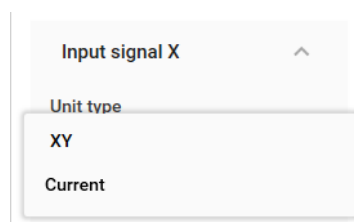
MR-E-2 can be set to four different operational modes:

- **Static value:** Optotune Cockpit sends a single position or current value to the driver.
- **Signal generator:** The internal signal generator within the driver sends a time-dependent stream of position or current values until it is interrupted. Available signal generating functions are Sinusoidal, Rectangular, Sawtooth and Triangular.
- **Arbitrary vector:** The driver reads a set of user-defined positions or currents one after another and sends them to the mirror.
- **Analog input:** The driver switches to Analog input mode (see section on analog control).

---

**Important:** Make sure that the unit type in the Signal Input panel matches the control mode set for the corresponding axis. Closed loop control is only possible with XY unit type. Open loop control is only possible with Current unit type.

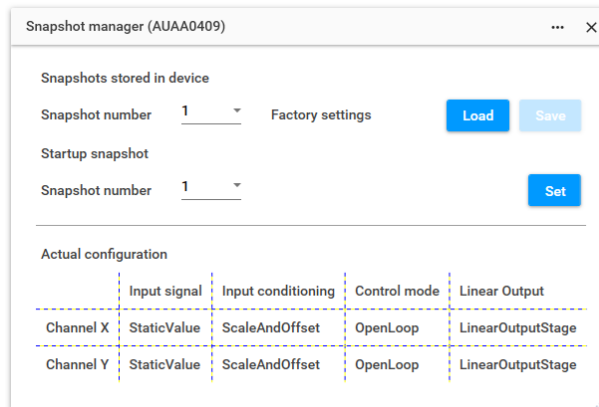
---



## 7.4 Factory Reset

You can always return to factory setting by opening the *Snapshot manager* panel and loading snapshot 1. Snapshot 1 contains the factory settings and cannot be overridden.

Other snapshots can be used to store current settings of the mirror and to automatically apply these directly after power-up by configuring the startup-snapshot.



Snapshot manager (AUAA0409)

Snapshots stored in device

Snapshot number  Factory settings

Startup snapshot

Snapshot number

Actual configuration

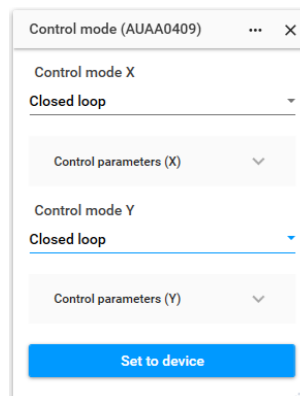
	Input signal	Input conditioning	Control mode	Linear Output
Channel X	StaticValue	ScaleAndOffset	OpenLoop	LinearOutputStage
Channel Y	StaticValue	ScaleAndOffset	OpenLoop	LinearOutputStage

## 7.5 Examples

In this section we show two examples on how to control the mirror with Optotune Cockpit. The first example includes setting both axes to closed loop control and moving the mirror to a specific point. The second example demonstrates how to operate in mixed mode waveform operation.

### 7.5.1 Closed loop static operation

We will move the mirror to the position [0.2, -0.2]. First, set both axes to closed loop control in the *Control Mode* panel and click *Set to device*.



Control mode (AUAA0409)

Control mode X

Closed loop

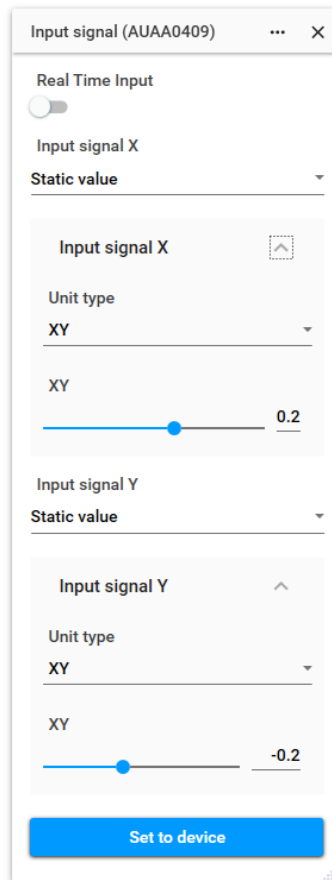
Control parameters (X)

Control mode Y

Closed loop

Control parameters (Y)

Next, open the *Signal input* panel. Under *Input signal X* choose static value. Repeat the same step for the Y-axis. In the *Input signal X* drop-down menu, choose unit type XY and set the value 0.2 either using the slider or typing it in directly. Repeat this for the Y-axis and set a value of -0.2. The panel should now look like this:

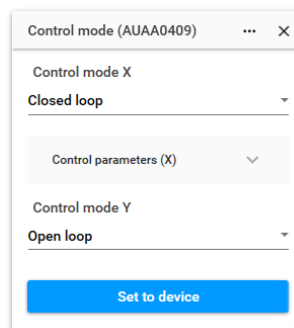


Now click *Set to device* and observe the mirror moving to the [0.2, -0.2] position.

### 7.5.2 Mixed Mode Waveform Operation

We will set the X-axis to closed loop triangular waveform mode with an amplitude of 0.25 and a frequency of 2 Hz. The Y-axis will operate in open loop sinusoidal waveform mode with an amplitude of 0.1 A and a frequency of 30 Hz. We call the situation with one axis operating in closed loop and the other axis operating in open loop mixed mode operation.

First, set the X-axis to closed loop mode and the Y-axis to open loop mode in the *Control Mode* panel. Click *Set to device*.



Next, move to the *Input signal* panel. Set the input signal of the X-axis to *Signal generator* and activate *Toggle running*. Set *Unit type* to XY and *Shape* to Triangular. Set *Frequency* to 2 Hz and *Amplitude* to 0.25. Set the input signal of the Y-axis to *Signal generator* and activate *Toggle running*. Set *Unit type* to Current and *Shape* to Sinusoidal. Set *Frequency* to 30 Hz and *Amplitude* to 0.15. The settings should now look like this:



Input signal X	Input signal Y
Signal generator	Signal generator
<div>Input signal X</div> <div>Toggle running</div> <div>Unit type</div> <div>XY</div> <div>Shape</div> <div>Triangular</div> <div>Frequency [Hz]</div> <div>Amplitude</div> <div>Offset</div> <div>Phase delay [°]</div>	<div>Input signal Y</div> <div>Toggle running</div> <div>Unit type</div> <div>Current</div> <div>Shape</div> <div>Sinusoidal</div> <div>Frequency [Hz]</div> <div>Amplitude [A]</div> <div>Offset [A]</div> <div>Phase delay [°]</div>

Now click *Set to device* and observe the mirror movement.

## 8 Simple Serial Communication Mode

The MR-E-2 firmware provides simple serial mode operation. When activated, the user can send string (ascii) commands via serial communication to control the mirror. It only offers basic control functionality. The complete list of supported commands (see section 8.3) is a small subset of the full functionality offered by the SDKs (Python and C#). This section describes a step-by-step procedure to execute a simple position change of the mirror. Subsequently, we list all the commands available along with their input range and syntax examples. Commands sent via serial are not case sensitive. The maximum message size is 64 bytes.

### 8.1 Installation of a Serial Communication Terminal

The software HTerm is a third-party software available online (<https://www.der-hammer.info/terminal/hterm-windows.zip>). HTerm does not have an installation file. To run it, simply execute the file *HTerm.exe* located within the downloaded files.

A serial communication terminal can be used to control the mirror by simply sending string (ascii) commands to the driver via the USB connection. The driver accepts ascii commands after startup, until Optotune Cockpit establishes a connection. If the user disconnects the driver through Optotune Cockpit, it will accept string commands again.

---

**Important:** Each serial command must be delayed by 1 ms.

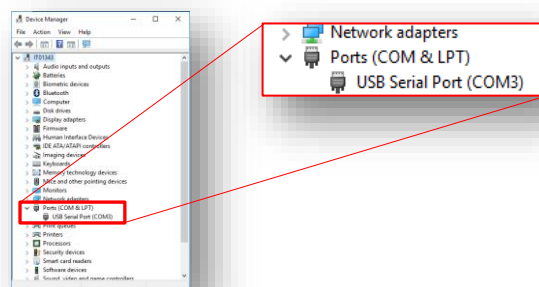
---

### 8.2 Step-by-Step procedure

Make sure that the driver is connected to the computer and powered. Do not have an open connection to the driver with Optotune Cockpit that would block the COM port.

First, find the port corresponding to the driver:

1. On the computer, open the *Run* dialog box by pressing and holding the Windows key, then press the R key.
2. Type *devmgmt.msc* and press Enter.
3. Make sure that you have no other USB devices connected.
4. Observe the *Ports (COM & LPT)* section of the Device Manager dialog window. Note down the COM port corresponding to the driver. In the case of this example the port is COM3 as shown below.



Next, configure a serial communication terminal:

1. Open the serial communication terminal by executing *HTerm.exe* (see icon below).

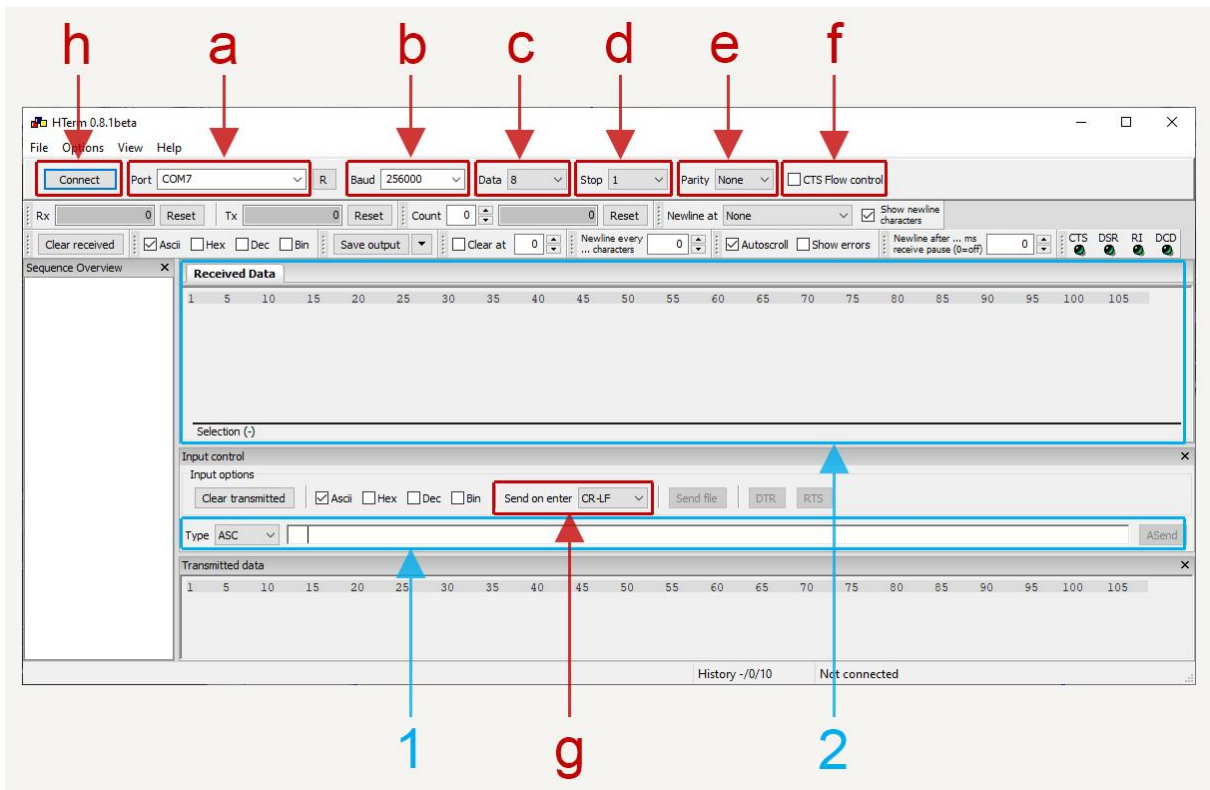


2. Follow the steps below to configure the terminal panel. All steps are indicated by the red marking in the figure.

- a. **Port:** Make sure that the COM port is the one specified in the Device Manager (COM7 in this example)
- b. **Baud:** 256000 bps
- c. **Data:** 8
- d. **Stop Bit:** 1
- e. **Parity Bit:** None
- f. **CTS Flow Control:** None (i.e. in this case unticked)
- g. **Send on enter:** All commands sent to the hardware, need to be terminated with the end-of-line character combination `\r\n`.

For example, when sending the command *Start*, the End-of-Line character combination must be included at the end of the string (i.e. *start\r\n*). By setting the field *Send on enter* to the option CR-LF we ensure that every string (ascii) command you type terminates correctly with an `\r\n`.

- h. **Connect:** Press the button *Connect* to establish a connection between the host PC and the driver.



Now we can start to control the mirror driver via serial commands.

In the *Type* (blue rectangle [1]) field make sure that the option *ASC* is selected from the drop-down menu. Type the following command and hit enter:

```
>> Start
```

Make sure that the *Received Data* (blue rectangle [2]) section of the terminal displays the following:

```
<< OK\r\n
```

This command does not execute any move. It simply works as a handshake between hardware and host PC.

To tilt the mirror in the X-axis, type the following command in the terminal and hit enter:

```
>> x=0.5
```

In the received section of the terminal, the following text should appear:

```
<< OK\r\n
```

In order to reset the mirror to its zero position, type the following text into the terminal:

```
>> xy=0;0
```

The received section of the terminal should display the following text:

```
<< OK\r\n
```

The mirror will then return to the zero-position.

To tilt the mirror in the Y-axis, type the following command in the terminal and hit enter:

```
>> y=0.5
```

In the received section of the terminal should display the following text:

```
<< OK\r\n
```

### 8.3 List of commands available

Command	Range	Explanation	Example
start\r\n	N/A	Used for handshake.	
reset\r\n	N/A	Restarts firmware.	
status\r\n	N/A	Returns the status register in hex format. See Table 5 Status register map	
acknowledge\r\n	N/A	Clears history error flags in the status register.	
getid\r\n	N/A	Gets the firmware serial number	
getsn\r\n	N/A	Gets the driver and mirror serial number	
getversion\r\n	N/A	Gets the firmware version number	
x= X.XXXX\r\n	min: -1.0 max: +1.0	Sets mirror to a calibrated position within the range provided.	x= 0.5\r\n
y= X.XXXX\r\n		Sets mirror to a calibrated position for both channels (x and y) within the range provided.	y= -0.6\r\n
xy= X.XXXX;X.XXXX\r\n			xy=-0.3;0.1\r\n
currentx= XXX.XmA\r\n	min: -500 max: +500	Set current within the range provided.	currentx = 20.2mA\r\n
currenty= XXX.XmA\r\n			currenty = -100.3mA\r\n
gopro\r\n	N/A	Changes to "Pro" mode.	
goprocrc\r\n	N/A	Changes to "Pro" mode with CRC enabled.	

Table 3 List of commands used in "Simple Mode"

For each command the terminal transmits to the driver, there should be a response from the driver. The table below lists all possible responses.

Data Received	Explanation
OK\r\n	the message has been processed with no errors.
ERROR\r\n	driver has an active error. Issue status command to read it
0000000000\r\n	response of status when there is no error
0x00000109\r\n	example response of status command when there is an active error
13816100-00-A\r\n	example response of getid command
1.2.739936\r\n	example response of getversion command
Board: BODA0000, Device: AUAA0346\r\n	example response of getsn command
OU\r\n	out of range upper.
OL\r\n	out of range lower
NO\r\n	the message has not been recognized.

Table 4 List of commands returned by the mirror driver

The table below details the meaning of each bit returned by the `status\r\n` command:

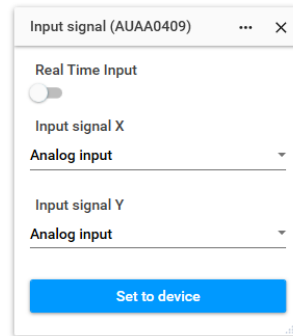
Bit#	Description
0	Proxy not connected
1	Proxy temperature threshold is reached
2	Mirror temperature threshold is reached
3	Mirror EEPROM not valid
4	Mirror not stable

5	Output current limit is reached
6	Output current average limit is reached
7	XY input is trimmed
8	Proxy was disconnected
9	Proxy temperature threshold was reached
10	Mirror temperature threshold was reached
11	Output current limit was reached
12	Output current average limit was reached
13	XY input was trimmed
14..31	Reserved

*Table 5 Status register map*

## 9 Analog Mode

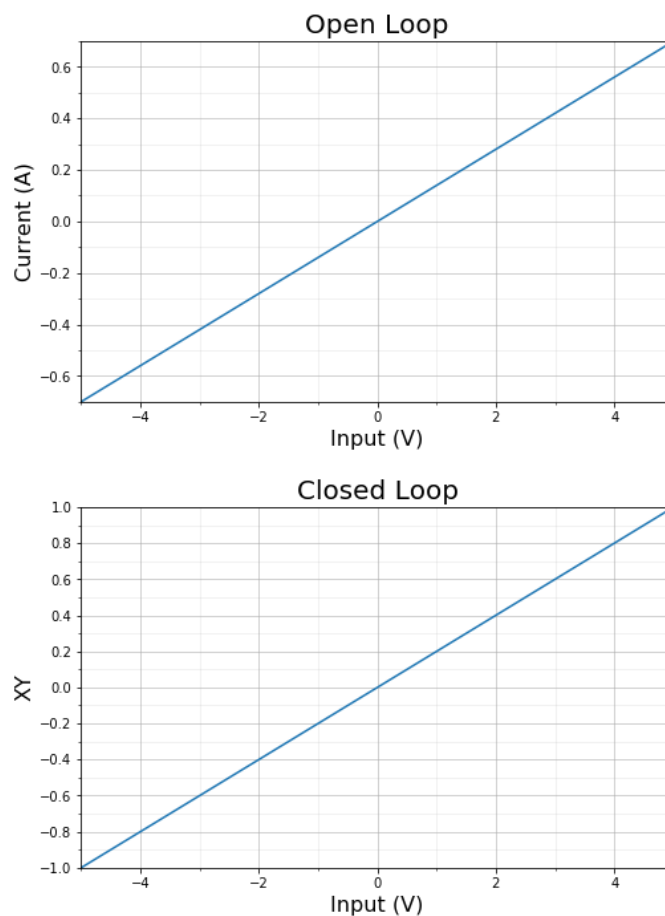
The analog mode must be explicitly activated through Optotune Cockpit or SPI. When analog mode is active, the analog input pins should be connected and not be left floating. Choose analog mode in the *Signal input* panel and click *Set to device*:



### 9.1 Input Pins and Signal Levels

The input pins are located on the I/O connector. Pin 1 controls the X-axis and pin 2 controls the Y-axis. The analog inputs are bipolar single ended signals to ground.

The range of AI\_X and of AI\_Y is from -5V to 5V. The input signals map linearly to this voltage. The figure below shows how the whole voltage range is mapped to the open loop and closed loop input range.

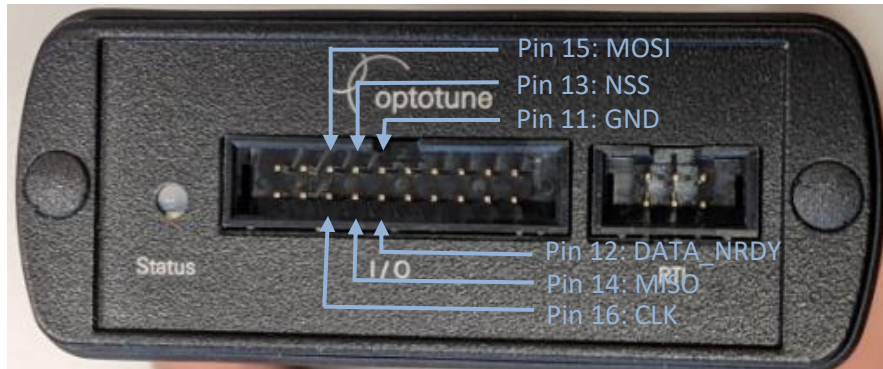


## 10 SDKs

There are both a Python SDK and a C# SDK available in the Software section including separate documentation and example code here: <https://www.optotune.com/downloads>.

## 11 SPI Interface

This section describes how the MR-E-2 operates as SPI slave with the following pin assignments on the I/O connector:

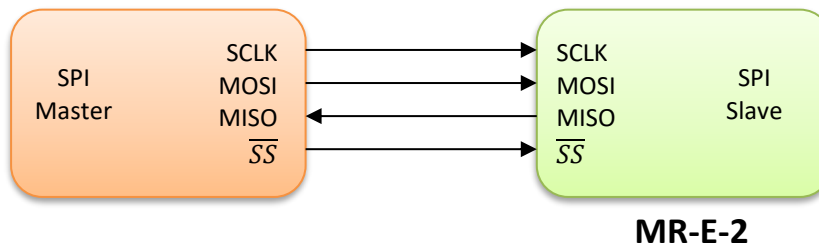


SPI is always active and does not need to be activated or deactivated through Optotune Cockpit. The mirror can be driven through SPI while the master is receiving real time data. The master has full access to all MR-E-2 registers through SPI. The logic level is 3.3V (CMOS).

<i>Signal</i>	<i>Pin on MR-E-2 I/O Connector</i>	<i>IO Direction</i>
SPI_MOSI	15	In
SPI_CLK	16	In
SPI_NSS	13	In
SPI_MISO	14	Out
GND	11	-
SPI_DATA_NRDY	12	Out

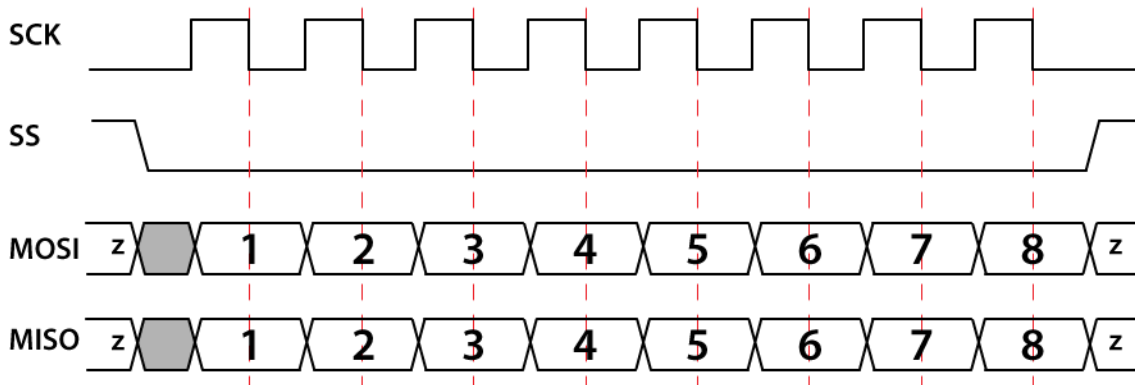
### 11.1 Interface

MR-E-2 driver has a four wire SPI slave interface as shown below:



MR-E-2 uses SPI Mode 1 (CPOL = 0, CPHA = 1). The Interface accepts data on the positive edge of the clock and samples the data at the negative edge. Additionally, MR-E-2 provides a SPI data not ready signal for accurate time synchronization with the master. The following timing diagram shows the 8 bit data frame.

## CPOL=0 CPHA=1



### 11.2 Framing

Each transfer comprises 14 bytes with 8 bits each. A negative edge on the NSS line marks the beginning of a new frame. A frame ends with a positive edge on the NSS line. For more information about the framing refer to Chapter 11.4. Note that MSB comes first in the transmitted data.

### 11.3 Timing and synchronization

- The interface is designed to work up to 4 MHz clock frequency.
- The time required to drive the MISO line after the positive edge of the clock is about 35 ns.
- To ensure a proper operation of the interface, the time between the negative edge of the NSS signal and any edge on the clock signal must be at least 20 ns.

In order to achieve accurate time synchronization between the slave and the master, the MR-E-2 provides a SPI\_DATA\_NRDY signal. When MR-E-2 starts receiving a SPI frame, the SPI\_DATA\_NRDY signal is driven high, signaling that the receiving frame is being processed. Once the frame is processed and the data requested are ready for transmission, the SPI\_DATA\_NRDY is driven low. Master should not send a new SPI frame while the SPI\_DATA\_NRDY is high. MR-E-2 driver updates its registers at a frequency of 10 kHz. Frames received at a higher frequency cannot be processed.

### 11.4 Data Structure

The following sections explain the 14 bytes (seven 16-bit words) for each data transfer (MISO and MOSI). The master can write and read registers in order to control the driver. The master can write up to two registers and read up to three registers with one frame transaction. Regardless of a read or write request, the last eight bytes of the SPI data frame slave response is reserved for reading back registers. For full description of the MR-E-2 register map, please refer to MR-E-2 Firmware documentation (<https://www.optotune.com/registration-for-software-download>).

#### 11.4.1 Master Out, Slave In (MOSI)

A data frame corresponding to a write request contains the following words:

Word	Description
0	Write flag (0x0001)
1	'System ID1' + 'Register ID1' to perform write
2	'System ID2' + 'Register ID2' to perform write
3-4	Data to write in address 'System ID1' + 'Register ID1'
5-6	Data to write in address 'System ID2' + 'Register ID2'



A data frame corresponding to a read request contains the following words:

Word	Description
0	Read flag (0x0000)
1	'System ID' + 'Register ID' to perform read
2-6	0

#### 11.4.2 Master In, Slave Out (MISO)

The slave responds to a write request with the following frame:

Word	Description
0	Write flag (0x0001)
1	'System ID1' + 'Register ID1', by write fail: 0x0000
2	'System ID2' + 'Register ID2', by write fail: 0x0000
3-4	Read back data according to SPI read pointer register 0, by read back fail: 0x7cf0bdc2
5-6	Read back data according to SPI read pointer register 1, by read back fail: 0x7cf0bdc2

The slave responds to a read request with the following frame:

word	Description
0	Read flag (0x0000)
1-2	Read back data of address 'System ID' + 'Register ID' asked from previous read request, by read back fail: 0x7cf0bdc2
3-4	Read back data according to SPI read pointer register 0, by read back fail: 0x7cf0bdc2
5-6	Read back data according to SPI read pointer register 1, by read back fail: 0x7cf0bdc2

For full description of the MR-E-2 controlling and configuring capabilities using SPI, please refer to MR-E-2 Firmware manual.

## 11.5 Examples

Refer to MR-E-2 Firmware documentation for complete register references.

#### 11.5.1 Setting open loop to both axis and reading back X Y

By default, the driver is configured in open loop mode and the read back of the SPI is pointing to the X, Y read registers. Therefore, no initial configuration is needed for this example. The SPI master can directly send a SPI frame to command a current in both channels.

In the following example frame the master commands 0.050 A in X channel and 0.080 A of opposite direction in Y channel.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x5000	0x5100	0x3d4cccd	0xbda3d70a

Explanation:

- Word 0: This example is a write request. 0x0001 is the SPI write flag.
- Word 1: 0x5000 is the register address in the static input system of X axis that sets current
- Word 2: 0x5100 is the register address in the static input system of Y axis that sets current
- Words 3-4: 0x3d4cccd is the single precision floating point representation of 0.050.
- Words 5-6: 0xbda3d70a is the single precision floating point representation of -0.080.

While the master sends the above frame in the MOSI line, the driver will send the X Y readback values in Words 3-4 and Bytes 5-6 in the MISO line, see subsection 11.4.2.

#### 11.5.2 Setting closed loop triangular to X axis and open loop sinusoidal to Y axis

For this example, the signal flow of the driver needs to be configured accordingly. The following frames have to be sent.

1. Select signal generator system as active input for X and Y axis.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x4000	0x4005	0x00000060	0x00000061

Explanation:

- Word 0: This example is a write request. 0x0001 is the SPI write flag.
- Word 1: 0x4000 is the register address for the active input system of X axis.
- Word 2: 0x4005 is the register address for the active input system of Y axis.
- Word 3-4: 0x00000060 is the ID of the signal generator system for X axis.
- Word 5-6: 0x00000061 is the ID of the signal generator system for Y axis.

2. Activate closed loop control for the X axis and open loop control for Y axis.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x4002	0x4007	0x000000c0	0x000000b1

Explanation:

- Word 1: 0x4002 is the register address for the control mode of X axis.
- Word 2: 0x4007 is the register address for the control mode of Y axis.
- Word 3-4: 0x000000c0 is the ID of the closed loop system for X axis.
- Word 5-6: 0x000000b1 is the ID of the open loop system for Y axis.

3. Configure signal unit. XY unit for X axis and current for Y axis.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x6000	0x6100	0x00000002	0x00000000

Explanation:

- Word 1: 0x6000 is the register address for the unit of the signal generator for X axis.
- Word 2: 0x6100 is the register address for the unit of the signal generator for Y axis.
- Words 3-4: 0x00000002 is the ID for XY unit.
- Words 5-6: 0x00000000 is the ID for current unit.

4. Configure signal shape. Triangular for X axis and sinusoidal Y axis.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x6002	0x6102	0x00000001	0x00000000

Explanation:

- Word 1: 0x6002 is the register address for the shape of the signal generator for X axis.
- Word 2: 0x6102 is the register address for the shape of the signal generator for Y axis.
- Words 3-4: 0x00000001 is the ID for triangular shape.
- Words 5-6: 0x00000000 is the ID for sinusoidal shape.

5. Configure signal frequencies (for example 5.0 Hz for X axis and 10.0 Hz for Y axis).

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x6003	0x6103	0x40a00000	0x41200000

Explanation:

- Word 1: 0x6003 is the register address for the frequency of the signal generator for X axis.
- Word 2: 0x6103 is the register address for the frequency of the signal generator for Y axis.
- Words 3-4: 0x40a00000 is the single precision floating point representation of 5.0.
- Words 5-6: 0x41200000 is the single precision floating point representation of 10.0.

6. Configure signal amplitudes (for example 0.6 for X axis and 0.05 A for Y axis).

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x6004	0x6104	0x3f19999a	0x3d4cccd

Explanation:

- Word 1: 0x6004 is the register address for the amplitude of the signal generator for X axis.
- Word 2: 0x6104 is the register address for the amplitude of the signal generator for Y axis.

7. Set the run flag for the signal generators.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x6001	0x6101	0x00000001	0x00000001

Explanation:

- Word 1: 0x6001 is the register address for run flag of the signal generator for X axis.
- Word 2: 0x6101 is the register address for run flag of the signal generator for Y axis.
- Words 3-4: 0x00000001 sets flag to true.

#### 11.5.3 Activate analogue mode

The default active input system of the driver is the static input. Analog input can be explicitly configured through SPI with the frame below.

Word 0	Word 1	Word 2	Words 3-4	Words 5-6
0x0001	0x4000	0x4005	0x00000058	0x00000059

Explanation for X axis:

- Word 0: This example is a write request. 0x0001 is the SPI write flag.
- Word 1: 0x4000 is the register address for the active input system of X axis.
- Word 2: 0x4005 is the register address for the active input system of Y axis.
- Word 3-4: 0x00000058 is the ID of the analogue system for X axis.
- Word 5-6: 0x00000059 is the ID of the analogue system for Y axis.