\*\*Introduction\*\*

SenAOReFoc is a complete closed-loop sensorbased adaptive optics (AO) and remote focusing software that works with a deformable mirror (DM) and a Shack-Hartmann wavefront sensor (SHWS). When a large stroke DM is used, such as an Alpao-69 or Alpao-97, both the closed-loop AO correction and remote focusing functionalities can be exploited. If the DM exhibits insufficient stroke for remote focusing, the remote focusing unit can be simply ignored, leaving the closed-loop AO correction functionalities fully functional without additional modifications to the software. The graphic user interface (GUI) exhibits modular widget arrangements and interactive commands are available in the message box for user guidance.

SenAOReFoc is developed under PySide2, which is a Python binding of the Qt application framework, with the GUI built using Qt Designer. Main processes are executed as separate worker threads from 'app.py' to allocate and recycle resources. Communication of events during processes are then handled using Qt's signals and slots mechanism. Common events between different threads, such as displaying an image or error message, reuse the same slots.

Closed-loop AO correction can be performed using both the zonal method, which updates DM control voltages in terms of the raw slope values; and the modal method, which updates DM control voltages in terms of orthogonal Zernike polynomials. There are four sub-modes tagged to each of the two methods: 1) standard closed-loop AO correction; 2) closed-loop AO correction with consideration of obscured search blocks; 3) closed-loop AO correction ignoring defocus; 4) closed-loop AO correction with consideration of obscured search blocks and ignoring defocus.

The software has been validated on a reflectance confocal microscope. However, it should also be fully compatible with fluorescence microscopes designed in a closed-loop configuration.

\*\*Statement of need\*\*

\*\*Scripts included in this folder and general descriptions\*\*

* app.py - Sets up GUI and event handlers, displays the GUI, instantiates devices such as DM and SHWS
* AO\_slopes.py - Performs closed-loop AO correction via direct slopes (zonal correction)
* AO\_zernikes.py - Performs closed-loop AO correction via Zernikes (modal correction)
* calibration\_RF.py - Performs remote focusing calibration using closed-loop AO correction
* calibration\_zern.py - Generates DM control matrix
* calibration.py - Performs DM calibration
* centroid\_acquisition.py - Methods for centroiding of SH spots
* centroiding.py - Performs system aberration calibration
* config.py - Sets configuration file path
* config.yaml - Configuration file incorporating all system and static parameters **(please change and add as needed according to the specific system)**
* conversion.py - Performs slope - Zernike conversion
* data\_collection.py - Performs automated characterisation of microscope AO performance
* HDF5\_dset.py - Methods for generating and appending data to HDF5 datasets
* image\_acquisition.py - Methods for image acquisition
* log.py - Method to set up event logs
* mirror.py - Performs DM initialisation
* SB\_geometry.py - Performs initialisation of search block geometry
* SB\_position.py - Performs repositioning of search blocks
* sensor.py - Performs SHWS initialisation
* SH\_acquisition.py - Performs SHWS image acquisition
* zernike.py - Generates Zernike polynomials and derivatives

\*\*Installation instructions\*\*

You will need the following open-source software installed:

* Python 3.6 +: <https://www.python.org/downloads/>
* Git: <https://www.python.org/downloads/>
* A source-code editor (Visual Studio Code): <https://code.visualstudio.com/>
* HDFView: <https://portal.hdfgroup.org/display/support/Download+HDFView>

Once these are installed, in the command prompt clone the source-code from the Git repository:

git clone -b cleaned\_v1 <https://github.com/jiahecui/SensorbasedAO.git>

Then set up a virtual environment (optional but recommended):

python –m venv venv

And activate the virtual environment:

venv\Scripts\activate

Then install all package dependencies:

pip install –r requirements.txt

These include:

* numpy
* PySide2==5.13.2
* qimage2ndarray
* Click
* scipy==1.5.4
* h5py==2.9.0
* pyyaml

And install the project in editable mode:

pip install –e .

Finally, SenAOReFoc can be run in **standard mode** in the availability of hardware devices (DM and SHWS:

python sensorbasedAO/app.py

As well as **debug mode** in the absence of hardware devices, which then sets up dummy devices for the purpose of testing software functionality:

python sensorbasedAO/app.py -d

\*\*Getting started\*\*

Before running SenAOReFoc on the microscope system in **standard mode**, please check the ‘config.yaml’ file which incorporates all system and static parameters required to correctly function the software. Modifications should be made according to the specific system and new parameters can be added as needed. All parameters have been commented. The current example works with an Alpao-69 DM and custom SHWS with a Ximea CMOS camera. In particular, parameters that need to be modified for basic closed-loop AO correction include:

camera:

  SN: "26883050"

  exposure: 40000  # us

  frame\_rate: 20.0  # Hz

  sensor\_width: 2048  # pixels Ximea - 2048

  sensor\_height: 2048  # pixels Ximea - 2048

  sensor\_diam: 11.26  # mm Ximea - 11.26

  bin\_factor: 2  # Ximea - 2

  pixel\_size: 5.5  # um Ximea - 5.5

DM:

  SN: "HSDM69-15-014"

  exercise: 0  # Flag for whether to exercise DM upon initialisation

  vol\_min: -0.1  # V Negative voltage to apply on DM during calibration

  vol\_max: 0.1  # V Positive voltage to apply on DM during calibration

   vol\_bias: 0  # Neutral voltage of DM actuators

  settling\_time: 0.001  # DM membrane settling time

DM0:

  actuator\_num: 69  # Alpao

  pitch: 1500  # um

  aperture: 10.5  # mm

relay:

  mirror\_odd: 0  # Flag for whether there is an odd number of mirrors in between DM and lenslet

  relay\_odd: 0  # Flag for whether there is an odd number of relays in between DM and lenslet

lenslet:

  lenslet\_pitch: 150  # um

  lenslet\_focal\_length: 5200  # um

search\_block:

  pupil\_diam\_0: 2.2  # mm Diameter of beam incident on SHWS

sys\_calib:

  sys\_calib\_mode: 1  # Mode flag for system aberration calibration method, 0 – load previous system aberration calibration profile, 1 – perform new system aberration correction

AO:

  zern\_gen: 0  # Flag for generating zernike modes on DM in AO\_zernikes.py and AO\_slopes.py, 0 - off, 1 - iterative generation

  loop\_max\_gen: 15  # Maximum number of loops for closed-loop control during generation of zernike modes

  recon\_coeff\_num: 69  # Number of zernike modes to use during wavefront reconstruction

  control\_coeff\_num: 20  # Number of zernike modes to control during AO correction  # 3rd - 14, 4th - 20, 5th - 27, 6th - 35, 7th - 44, 8th - 54, 9th - 65, 10th - 77

Note that the ‘DM exercise’ option may need to be set to 1 for electrostatic mirrors that observe substantial thermal effects [], which will start a process of sending random voltages within [-0.5, 0.5] to all actuators for ~5 mins upon initialisation of search blocks. After above parameters have been modified, the procedures below can be performed for basic closed-loop AO correction. Note that for a microscope using reflectance contrast, this should be performed while scanning over a small region on scattering samples to avoid specular reflections that lead to the double-pass effect [], which causes errors to the aberration measurements. For fluorescence microscopes, a fluorescent bead can be used as the sample with a static beam.

**DM calibration:**

On the GUI press [Initialise SB] -> [Position SB] (follow instructions in message box to reposition search block area) -> [Calibrate-S] -> [S-Z Conv] -> [Calibrate-Z]

**System aberration calibration:**

To perform system aberration calibration anew, make sure the corresponding mode flag is left as default in ‘config.yaml’.

sys\_calib:

  sys\_calib\_mode: 1  # Mode flag for system aberration calibration method, 0 – load previous system aberration calibration profile, 1 – perform new system aberration correction, 2 – no system aberration correction

Then on the GUI press [Calibrate-Sys]. This will run closed-loop AO correction (zonal control) until the Strehl ratio is above the given tolerance factor (default Maréchal criterion 0.81), save final DM control voltages (DM system flat file) to the HDF5 file, and halt with the DM system flat file applied such that separate imaging can be performed.

To load a previous DM system flat file, change the corresponding mode flag in ‘config.yaml’.

sys\_calib:

  sys\_calib\_mode: 0  # Mode flag for system aberration calibration method, 0 – load previous system aberration calibration profile, 1 – perform new system aberration correction, 2 – no system aberration correction

Then on the GUI press [Calibrate-Sys]. This will automatically load the DM system flat file from last calibration session and halt with the DM system flat file applied.

To ignore system aberration correction for all subsequent closed-loop AO correction processes, change the corresponding mode flag in ‘config.yaml’.

sys\_calib:

  sys\_calib\_mode: 2  # Mode flag for system aberration calibration method, 0 – load previous system aberration calibration profile, 1 – perform new system aberration correction, 2 – no system aberration correction

Then on the GUI press [Calibrate-Sys]. This will save the current centroid positions as the reference point for convergence during closed-loop AO correction.

**Closed-loop AO correction:**

Closed-loop AO correction can be performed without first generating Zernike modes to the DM by setting the corresponding mode flag to its default value in ‘config.yaml’.

AO:

  zern\_gen: 0  # Flag for generating zernike modes on DM in AO\_zernikes.py and AO\_slopes.py, 0 - off, 1 - iterative generation

Then, closed-loop AO correction can be performed using 2 different approaches, each with 4 different sub-modes, as explained in the functionality documentation below.

Alternatively, for characterisation purposes, Zernike modes can be first applied to the DM before performing closed-loop AO correction by changing the corresponding mode flag in ‘config.yaml’.

AO:

  zern\_gen: 1  # Flag for generating zernike modes on DM in AO\_zernikes.py and AO\_slopes.py, 0 - off, 1 - iterative generation

\*\*Functionality documentation\*\*

SenAOReFoc consists of 5 main units, the SHWS initialisation and DM calibration unit, the Zernike aberration input unit, the AO control and data collection unit, the miscellaneous control unit, and the remote focusing unit. There are also 2 auxiliary units, a message box which informs the user of required inputs and the current task progress, and a software termination module. Options are available to perform both direct slope (zonal) and Zernike (modal) control.

\*SHWS initialisation and DM calibration unit\*

[Initialise SB] determines the search block (SB) geometry given user informed parameters of the lenslet pitch (the diameter of one lenslet), the dimension and pixel size of the camera sensor, as well as the incident pupil diameter. Using this information, the software first calculates the number of pixels across one SB and the number of SBs across the camera sensor, before determining the number and geometrical arrangement of the SBs within the pupil region of the incident beam. Then, their reference centroid coordinates with regard to the camera sensor is calculated, before the full SB region is displayed on the GUI. This button also exercises the DM membrane by generating a user specified number of random patterns on the DM, which helps to remove hysteresis and creep of the membrane for magnetic mirrors upon initialisation.

[Position SB] repositions the SB region on the camera sensor such that it is concentric with the incident beam to allow for maximum dynamic range. This can be achieved in two ways depending on whether or not the DM requires recalibration. User input is needed at this stage by entering 'y' or 'n' on the keyboard. Upon entering 'y', the user can then reposition the SB region by entering '↑', '↓', '←', '→' arrows on the keyboard, which will shift the entire SB region by one pixel at a time, and display it at its new location. When the central SH spot(s) is centred within its corresponding SB, by pressing 'Enter' on the keyboard, the user can confirm the new SB position and trigger the process of automatically calculating new SB reference centroid coordinates according to the number of pixels shifted in each direction. Upon entering 'n', SB and DM calibration related parameters from the last session will be automatically loaded from the HDF5 file.

[Calibrate-Sys] performs system aberration calibration in three modes. Mode 1 loads records centroid coordinates of the SH spots within each SB using the CoG algorithm with either a DM flat file or a DM system flat file applied. These will be used as the actual reference centroids for aberration analysis. Due to the presence of system aberrations, coordinates recorded in the former case are usually different to those of SB reference centroids (geometrical centre of SB).

[Calibrate-S] performs DM calibration to retrieve the control matrix (CM) for direct slopes (zonal) control. The positive and negative bias control voltages provided by the user are sequentially applied to each DM actuator to retrieve the 'x' and 'y' slope values, which are then used to calculate the DM influence function (IF) matrix and slope CM by computing its pseudo-inverse.

[S-Z Conv] calculates the conversion matrix between raw slope values and Zernike coefficients. This can be used to calculate the Strehl ratio (SR) using for evaluation of the image quality during direct slope (zonal) control, or to further acquire the DM IF matrix and CM for Zernike (modal) control.

[Calibrate-Z] retrieves the CM for Zernike (modal) control by first calculating the DM IF matrix using raw slope values acquired in [Calibrate-S] and slope-Zernike conversion matrix calculated in [S-Z Conv]. Before performing pseudo-inverse of the DM IF matrix, the results after singular value decomposition are first evaluated to identify a cut-off value for small singular values. This is usually selected as the value after which a significant drop is seen. System modes below this value are removed during pseudo-inverse for acquisition of a stable Zernike CM.

\*Zernike aberration input unit\*

The Zernike aberration input unit is mainly used for thorough characterisation of the system AO performance by specifying certain amounts of Zernike modes to generate on the DM membrane. Zernike modes are ordered according to the OSA/ANSI standard and piston is removed such that mode 1 starts with tip. [Zernike mode spin box] allows the user to specify a single mode number, the amplitude of which is given in [Zernike value spin box] in the unit of microns. Mode combinations can be specified in [Zernike array edit box] by providing an array of coefficients in the correct mode order. Irrelevant modes can be set as zero and only modes up to the maximum mode number controlled during AO correction (user-defined) can be generated.

\*AO control and data collection unit\*

This is the core software unit responsible for closed-loop AO control and automated data collection. Both Zernike (modal) and direct slopes (zonal) control can be performed in 4 different modes, which will be discussed below.

[Zernike AO 1] performs basic closed-loop Zernike (modal) AO control by updating DM control voltages with the Zernike CM obtained in [Calibrate-Z]. Termination of the control loop is realised after the SR satisfies the Maréchal criterion (SR > 0.8) or the number of iterations exceeds the maximum limit (user-defined). The same criteria is used for subsequent AO control modes. There are 2 sub-modes under this button that can be specified using mode flags in 'config.yaml'. Sub-mode 1 starts AO correction from a DM system flat file. Sub-mode 2 generates aberrations specified in the Zernike aberration input unit on the DM membrane before performing closed-loop AO correction. The latter sub-mode is especially useful for characterisation of the system AO performance.

[Zernike AO 2] is a modified version of [Zernike AO 1] that takes into account missing or ill-formed spots (obscured SBs) detected by the SHWS due to arbitrarily shaped pupils [Dong2018OpticsExpress,Ye2015OpticsExpress,cui\_j\_2020\_3885508]. In this case, Zernike polynomials which are defined over a unit circle lose orthogonality and original calibration matrices lead to unreliable interpretations of the wavefront. Therefore, a new set of orthogonal modes is established and an updatable control method is required to dynamically modify the CM during closed-loop AO correction.

[Zernike AO 3] is a modified version of [Zernike AO 1] that neglects the correction of Zernike defocus such that refocusing of the focal spot is minimised. For Zernike (modal) control, this is simply achieved by setting the detected value of Zernike mode 4 to zero before updating the DM control voltages using the original Zernike CM obtained in [Calibrate-Z].

[Slope AO 1] performs basic closed-loop direct slopes (zonal) AO control by updating DM control voltages with the direct slope CM obtained in [Calibrate-S]. The 2 sub-modes described in [Zernike AO 1] also apply.

[Slope AO 2] is a modified version of [Slope AO 1] fulfilling the same purpose as [Zernike AO 2].

[Slope AO 3] is a modified version of [Slope AO 1] fulfilling the same purpose as [Zernike AO 3].

[Zernike Full] and [Slope Full] perform closed-loop AO control taking into account both obscured SBs and suppression of Zernike defocus at the same time.

[Data Collection] is designed to perform automated AO performance characterisations of the microscope system, though any data collection process that benefits from automation can be written beneath the hood. Each operation mode is identified by a unique mode flag in 'config.yaml', which should be determined before software initialisation. Example execution of integrated modes on a real microscope system can be found in the 'examples' folder.

\*Miscellaneous control unit\*

This unit controls independent image acquisition when no other background process thread is running, as well as miscellaneous functions that are used by multiple units.

Independent image acquisition can be performed in 3 modes. [Live Acq] continuously acquires images and displays them on the GUI at the user-defined frame rate and exposure time until terminated by clicking the button a second time. [Burst Acq] acquires a user-defined number of frames in burst mode before automatically terminating. [Single Acq] only acquires one image when clicked.

[Reset DM] resets the actuators to their neutral position. [Camera exposure time spin box] controls the exposure time of the SHWS given in units of microseconds. [Maximum loop no. spin box] determines the maximum number of iterations before termination of the AO control loop.

\*Remote focusing unit\*

The remote focusing unit controls all parameters and procedures relevant to the calibration and execution of remote focusing using a DM.

[CALIBRATE] triggers the calibration of control voltages that enable DM actuators to deform the membrane for fine axial refocusing. For details of the RF calibration process, please refer to [Cui2021BiophotonicsCongress, Cui2021OpticsExpress]. The software performs this as a 2-step process that proceeds along the negative direction before the positive. Messages are displayed in the message box to inform users of the current calibration direction and step number, as well as to provide users with options to: 1) confirm that the sample has been moved to the next calibration step; and 2) save or discard calibrated voltages and exit the thread. Commands can then be entered via the keyboard. A typical example of the interactive command message is: Move sample to positive position. Press [y] to confirm. Press [s] to save and exit. Press [d] to discard and exit. The software automatically saves calibrated voltages to the HDF5 file after reaching the maximum user-defined step number, or does so on request at an arbitrary step. It also saves results of the slope values, Zernike coefficients, and SR acquired before and after performing closed-loop AO correction at each calibration step for further analysis of the wavefront. After interpolation has been performed between voltages acquired at each calibration step to acquire those for finer axial steps, all RF voltages can then be loaded upon software initialisation.

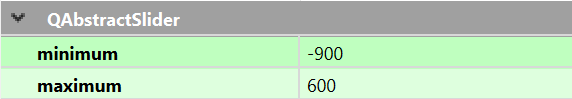
[Scan Focus] controls whether remote focusing is to be performed at only one depth or at multiple depths. When unticked, only the first spin box on the left-hand-side panel is accessible while others are disabled. When ticked, the otherwise is true.

[MOVE] performs remote focusing at only one depth by first retrieving control voltages corresponding to that specified in [Focus Depth (microns)], and then applying those voltages on the DM to move the focus relative to the natural focal plane. Negative values move towards -z-axis and positive ones +z-axis.

[SCAN] performs remote focusing at multiple depths sequentially by applying different sets of voltages to the DM membrane and scanning the focus depth-wise. [Step Increment (microns)] controls the displacement interval between consecutive remote focusing planes, [Step Number] the number of planes to be accessed, [Start Depth (microns)] the depth at which to start the remote focusing process relative to the natural focal plane, and [Depth Pause Time (s)] the time for which the focal spot remains stable at each remote focusing plane. Entering a negative value for [Step Increment (microns)] permits RF in the -z-direction.

[AO Correction Type] allows the user to choose whether or not to apply AO correction at each remote focusing plane and which type to apply. Four options are available from the drop box [Zernike AO 3], [Zernike Full], [Slope AO 3], and [Slope Full], the features of which have been discussed above. All options are applicable to the remote focusing process behind both [MOVE] and [SCAN], default is set to [None].

[Remote focusing toggle bar] permits random access of different remote focusing planes during imaging by simply dragging the bar to a desired depth. Default is set to the natural focal plane and max/min limits are provided by the user according to RF calibration results by changing the corresponding minimum and maximum value of ‘RFslider’ widget in Qt Designer. The values are calculated as [depth (um) / step increment (um)].



\*\*Example usage\*\*

Three examples are given for automated AO performance characterisations of the microscope system.

**Example 1**: The dynamic range of the SHWS for the first *N* Zernike modes can be characterised by generating and correcting for them in closed-loop using mode 0/1 of [Data Collection]. Parameters in ‘config.yaml’ under ‘data\_collect’ can be set as follows.

data\_collect:

  data\_collect\_mode: 1  # Mode flag for Data Collection button, detailed function descriptions in app.py

  loop\_max\_gen: 15  # Maximum number of loops during closed-loop generation of zernike modes in mode 0/1/2/3

  incre\_num: 15  # Number of zernike mode amplitudes to generate in mode 0/1

  incre\_amp: 0.02  # Increment amplitude between zernike modes in mode 0/1

  run\_num: 5  # Number of times to run mode 0/1/2/3

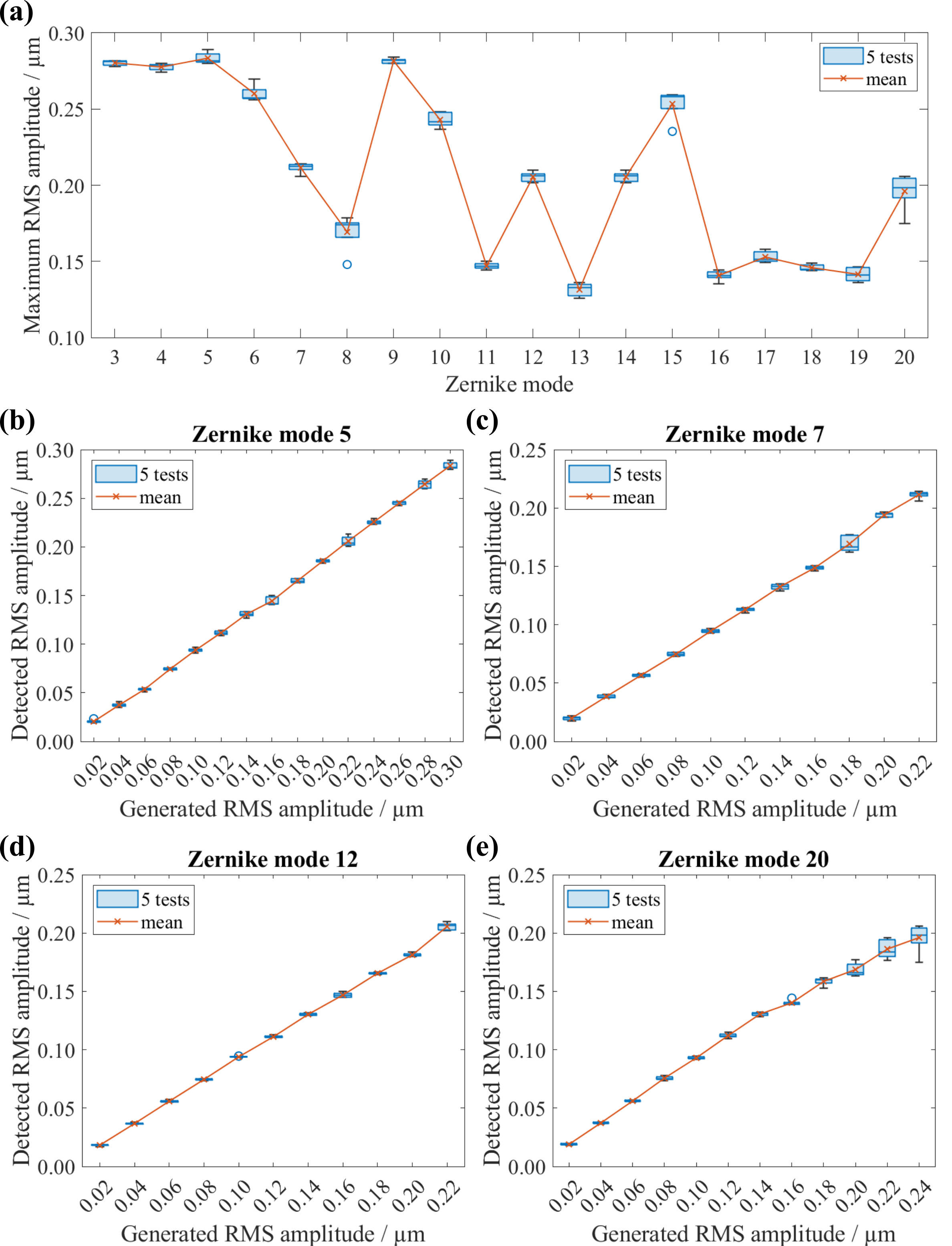


Figure 1. Characterisation results of system AO correction performance. (a) Dynamic range of the SHWS for Zernike modes 3∼20 (excl. tip/tilt). (b)-(e) Generated and detected RMS amplitudes of odd/even Zernike modes (b) 5, (c) 7, (d) 12, and (e) 20, in increments of 0.02 μm. 5 tests were performed for each measurement.

**Example 2**: The degree of Zernike mode coupling upon detection at the SHWS can be characterised by individually generating the same amount of each mode on the DM and constructing a heatmap of the detected Zernike coefficients using mode 0/1 of [Data Collection]. Parameters in ‘config.yaml’ under ‘data\_collect’ can be set as follows.

data\_collect:

  data\_collect\_mode: 1  # Mode flag for Data Collection button, detailed function descriptions in app.py

  loop\_max\_gen: 15  # Maximum number of loops during closed-loop generation of zernike modes in mode 0/1/2/3

  incre\_num: 1  # Number of zernike mode amplitudes to generate in mode 0/1

  incre\_amp: 0.1  # Increment amplitude between zernike modes in mode 0/1

  run\_num: 5  # Number of times to run mode 0/1/2/3

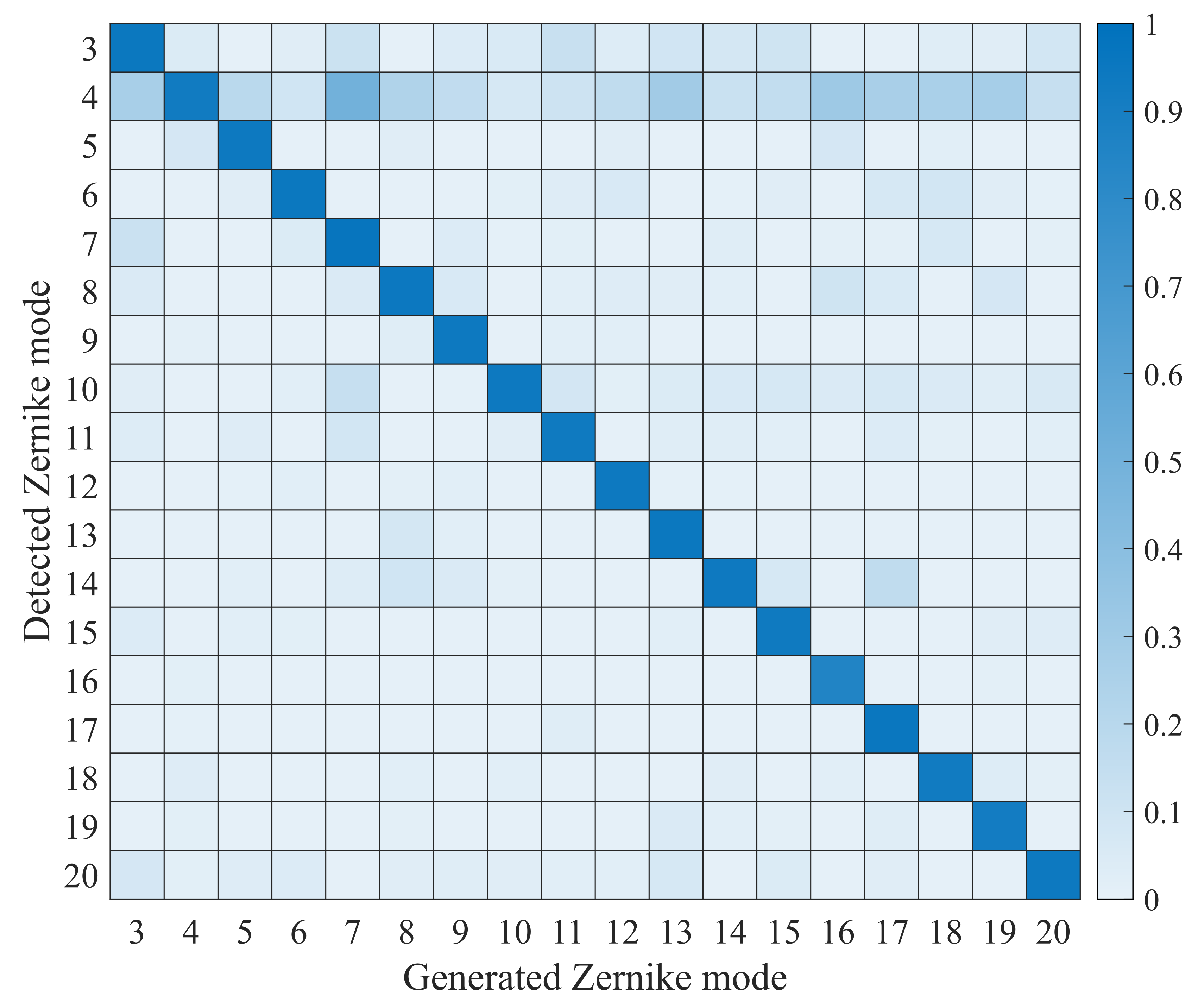


Figure 2. Heatmap of correlation coefficients between detected and generated mode values for 0.1 μm of Zernike modes 3∼20 (excl. tip/tilt). 5 tests were performed for each measurement.

**Example 3**: To ensure the system could correct for multiple Zernike modes with good stability and minimal mode coupling, different combinations of odd and even Zernike modes can be generated and corrected for in closed-loop using mode 2/3 of [Data Collection]. Parameters in ‘config.yaml’ under ‘data\_collect’ can be set as follows.

data\_collect:

  data\_collect\_mode: 3  # Mode flag for Data Collection button, detailed function descriptions in app.py

  loop\_max\_gen: 15  # Maximum number of loops during closed-loop generation of zernike modes in mode 0/1/2/3

  run\_num: 5  # Number of times to run mode 0/1/2/3

And [Zernike array edit box] can be set to [0 0 0 0 **0.1** 0 **0.1** 0 0 0 0 **0.1** 0 0 0 0 0 0 0 **0.1**].

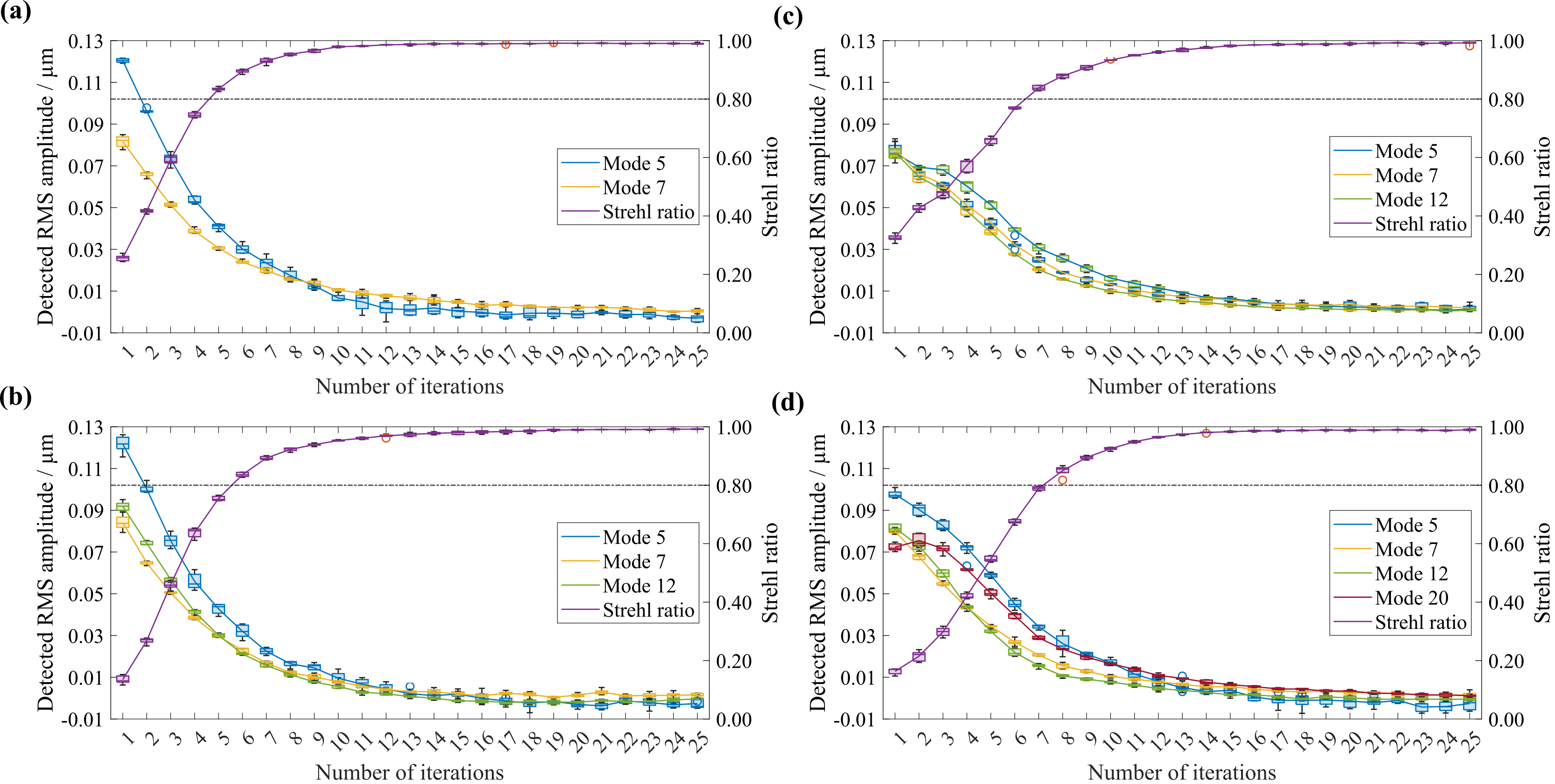


Figure 3. Detected amplitudes of generated odd and even Zernike mode combinations and Strehl ratio calculated using first 69 Zernike modes (excl. tip/tilt) at each iteration of closed-loop AO correction.

\*\*Automated tests\*\*

\*\*Notes for development\*\*

1. The current software assumes using an electromagnetic DM, where the actuator displacement is linearly proportional to the applied control voltage []. In this case, the device can be driven linearly in both the negative and positive directions by applying a normalised control voltage of , where is the maximum control voltage. However, if an electrostatic DM is to be used, the actuator displacement would be proportional to the square of the applied control voltage. In such a case, in order to drive the device linearly, the normalised control voltage should be given as .

An HDF5\_dset.py script is included as an example of how to store data using an HDF5 file. The centroid acquisition

process in centroid\_acquisition.py needs to load images from the HDF5 file for fast access rather than directly passing

them into the method.

Urban Bitenc, "Software compensation method for achieving high stability of Alpao deformable mirrors," Opt. Express 25, 4368-4381 (2017).

Pablo Artal, Susana Marcos, Rafael Navarro, and David R. Williams, "Odd aberrations and double-pass measurements of retinal image quality," *J. Opt. Soc. Am. A* 12, 195-201 (1995).

1. Bing Dong and Martin J. Booth, "Wavefront control in adaptive microscopy using Shack-Hartmann sensors with arbitrarily

shaped pupils," Opt. Express 26, 1655-1669 (2018). https://doi.org/10.1364/OE.26.001655.

2. Jingfei Ye, Wei Wang, Zhishan Gao, Zhiying Liu, Shuai Wang, Pablo Benítez, Juan C. Miñano, and Qun Yuan,

"Modal wavefront estimation from its slopes by numerical orthogonal transformation method over general shaped aperture,"

Opt. Express 23, 26208-26220 (2015). https://doi.org/10.1364/OE.23.026208.

3. Jiahe Cui, Bing Dong, and Martin J. Booth, "Shack-Hartmann sensing with arbitrarily shaped pupil (1.0),"

Zenodo (2020). https://doi.org/10.5281/zenodo.3885508.

4. Jiahe Cui, Raphaël Turcotte, Karen Hampson, Nigel J. Emptage, and Martin J. Booth, "Remote-Focussing for Volumetric

Imaging in a Contactless and Label-Free Neurosurgical Microscope," in Biophotonics Congress 2021, C. Boudoux,

K. Maitland, C. Hendon, M. Wojtkowski, K. Quinn, M. Schanne-Klein, N. Durr, D. Elson, F. Cichos, L. Oddershede, V. Emiliani,

O. Maragò, S. Nic Chormaic, N. Pégard, S. Gibbs, S. Vinogradov, M. Niedre, K. Samkoe, A. Devor, D. Peterka, P. Blinder,

and E. Buckley, eds., OSA Technical Digest (Optical Society of America, 2021), paper DTh2A.2.

5. Jiahe Cui, Raphaël Turcotte, Nigel J. Emptage, and Martin J. Booth, "Extended range and aberration-free autofocusing via

remote focusing and sequence-dependent learning," Opt. Express 29, 36660-36674 (2021). <https://doi.org/10.1364/OE.442025>.

Urban Bitenc, Nazim A. Bharmal, Timothy J. Morris, and Richard M. Myers, "Assessing the stability of an ALPAO deformable mirror for feed-forward operation," *Opt. Express* 22, 12438-12451 (2014).