The Compute and Control for Adaptive Optics (cacao) real-time control software package

Olivier Guyon[a,b,c], Arnaud Sevin[d], Damien Gratadour[d], Julien Bernard[d], Hatem Ltaief[e], Dalal Sukkari[e], Sylvain Cetre[f], Nour Skaf[c], Julien Lozi[c], Frantz Martinache[g], Christophe Clergeon[c], Barnaby Norris[h], Alison Wong[h], Jared Males[b]

[a]Astrobiology Center; National Institutes of Natural Sciences, Tokyo, JAPAN
[b]Steward Observatory, University of Arizona, Tucson, AZ 85721, USA
[c]National Astronomical Observatory of Japan, Subaru Telescope, National Institutes of Natural Sciences, Hilo, HI 96720, USA
[d]Observatoire de Paris, France
[e]King Abdullah University of Science and Technology, SAUDI ARABIA
[f]Keck Observatories, USA
[g]Observatoire de la Cote d'Azur, FRANCE
[h]University of Sydney, AUSTRALIA

ABSTRACT

The compute and control for adaptive optics (cacao) package is an open-source modular software environment for real-time control of modern adaptive optics system. By leveraging many-core CPU and GPU hardware, it can scale up to meet the demanding computing requirements of current and future high frame rate, high actuator count adaptive optics (AO) systems. cacao's modular design enables both simple/barebone operation, and complex full-featured AO control systems.

cacao's design is centered on data streams that hold real-time data in shared memory along with a synchronization mechanism for computing processes. Users and programmers can add additional features by coding modules that interact with cacao's data stream format.

We describe cacao's architecture and its design approach. We show that accurate timing knowledge is key to many of cacao's advanced operation modes. We discuss current and future development priorities, including support for machine learning to provide real-time optimization of complex AO systems.

Keywords: Adaptive Optics, High Performance Computing, Wavefront Control

1. INTRODUCTION

Adaptive Optics systems are now in operation on most current large telescopes,[1] and will be a key part of the next generation of large ground-based telescopes. Thanks to advances in computing power, low-noise high speed cameras, and deformable mirror technologies, AO systems are becoming more capable and meeting a broader set of requirements. The first generation of AO systems consisted of a single natural guide star wavefront sensor (WFS) driving a single deformable mirror (DM). New and future AO systems are considerably more complex and diverse: they often include multiple WFSs and DMs to achieve lower residual wavefront error, and/or to perform AO correction over a wider field of view.

The cacao package is aimed at providing a flexible, modular toolkit for adaptive optics control. Its source code, in C language, is open-source and available online[2] under a GPLv3 license. The package includes a command-line interpreter for low-level user input, and scripts & ASCII GUIs for high-level control. cacao's goal is to provide a modular, expandable toolkit for high performance AO control. cacao provides a framework for easy implementation of new wavefront control algorithms, while maintaining high performance and throughput.

Further author information: (Send correspondence to O.G.)
O.G.: E-mail: oliv.guyon@gmail.com, Telephone: 1 818 293 8826

In addition to CPU-based computations, support for GPGPU-based computation is provided through the CUDA library for NVIDIA hardware.

We describe the cacao software architecture in §2, and its capabilities / operation modes in §3. We describe in §4 how precision timing is instrumental to AO calibration and many of cacao's advanced features. Some of cacao's advanced operation modes are discussed in §5: multiple control loops operation, and machine learning for predictive control and sensor fusion.

2. SOFTWARE ARCHITECTURE

2.1 Data Streams

cacao uses, for all real-time data, a common **data stream** format, which stores all real-time data arrays and images (1D, 2D or 3D; integer or floating point precisions). The data stream format includes:

- metadata: image name, size, type, creation time, etc..
- A pointer to the data, held in shared memory.
- Pointers to semaphores (by default, 10 semaphores per data stream).
- Optional keywords

Holding the data in shared memory allows low-latency interprocess communication: several computing process can transparently share the same data stream, and read/write to the same memory space with little overhead. Synchronization between processes is handled by the semaphores, which are used as shared memory counters according to the following rules:

- Each data stream is written by a single process. When the write operation is complete, the writing process shall post (= increment) all semaphores in the data stream. The POSIX function ***sem_post()*** performs the post operation.
- An arbitrary number of processes may start computation as soon as the data stream is ready (write is complete). To do so, each of these processes waits on one of the semaphores using the POSIX function ***sem_wait()***. Each process will be blocked while the corresponding semaphore value is at zero; once the semaphore is posted, the process will immediately proceed and decrease the semaphore value (usually back to zero, such that the process only processes the frame once and then waits for the next frame again).

2.2 Inter-Process Communication

A conventional single-WFS, single-DM AO control loop implementation with cacao is shown in Figure 1. In this example, the cacao AO loop uses three data streams (blue rectangles: WFS image, incremental DM displacement, and total DM displacement) and six processes (gray rounded rectangular boxes: Frame grabber acquisition, data logger, link to other computer, multiply by control matrix, gain multiply and add, and DM electronics driver). All data streams are created and processes launched prior to the loop running. Since each of the process is waiting on a semaphore, and all semaphores are at zero, the processes are initially blocked: they will only start processing once semaphores are posted.

The algorithm steps are:

1. **Frame grabber camera acquisition** process: The WFS image is acquired from the camera and written to the WFS image data stream shared memory.
2. **Frame grabber camera acquisition** process: Upon completion of the WFS image acquisition, the WFS image data stream semaphores are posted: their values is incremented from 0 to 1.

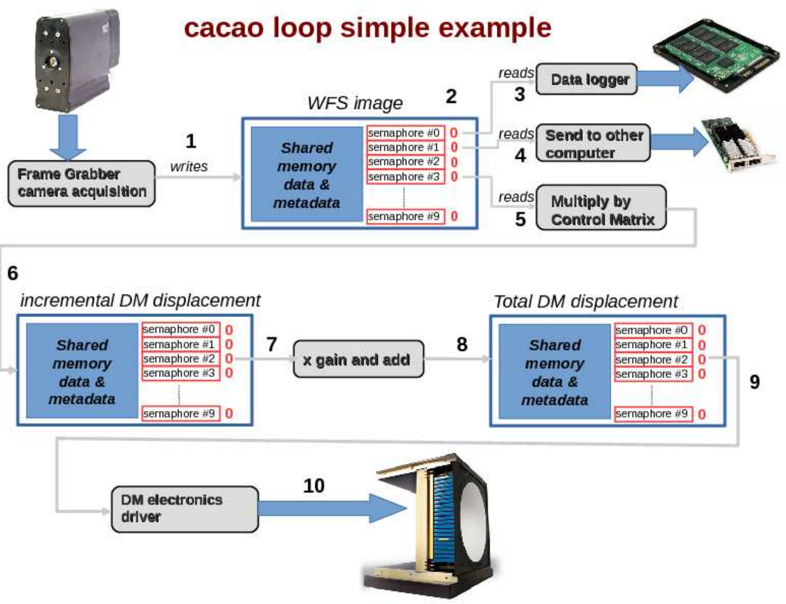
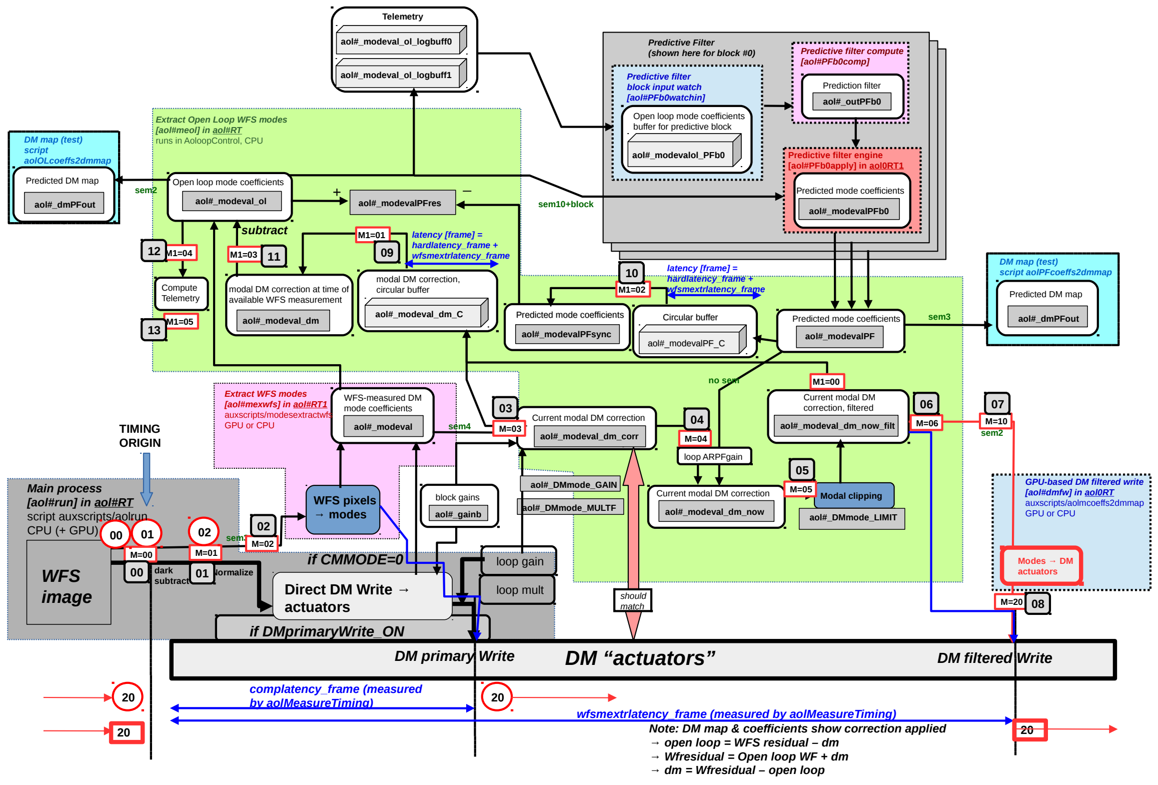


Figure 1. Modular implementation of an AO control loop with cacao. In this example, the control loop is constructed using three data streams.



Telemetry
aol#_modeval_ol_logbuff0
aol#_modeval_ol_logbuff1
Predictive Filter (shown here for block #0)
Predictive filter compute [aol#PFb0comp]
Prediction filter
aol#_outPFb0
Predictive filter block input watch [aol#PFb0watchin]
Open loop mode coefficients buffer for predictive block
aol#_modevalol_PFb0
Predictive filter engine [aol#PFb0apply] in aol0RT1
Predicted mode coefficients
aol#_modevalPFb0
DM map (test) script aolOLcoeffs2dmmap
Predicted DM map
aol#_dmPFout
Extract Open Loop WFS modes [aol#meol] in aol#RT runs in AoloopControl, CPU
Open loop mode coefficients
aol#_modeval_ol
aol#_modevalPFres
sem10+block
subtract
sem2
Compute Telemetry
modal DM correction at time of available WFS measurement
aol#_modeval_dm
modal DM correction, circular buffer
aol#_modeval_dm_C
latency [frame] = hardlatency_frame + wfsmextrlatency_frame
Predicted mode coefficients
aol#_modevalPFsync
Circular buffer
aol#_modevalPF_C
Predicted mode coefficients
aol#_modevalPF
sem3
DM map (test) script aolPFcoeffs2dmmap
Predicted DM map
aol#_dmPFout
no sem
Extract WFS modes [aol#mexwfs] in aol#RT1 auxscripts/modesextractwfs GPU or CPU
WFS-measured DM mode coefficients
aol#_modeval
sem4
Current modal DM correction
aol#_modeval_dm_corr
loop ARPFgain
Current modal DM correction, filtered
aol#_modeval_dm_now_filt
sem2
TIMING ORIGIN
block gains
aol#_gainb
aol#_DMmode_GAIN
aol#_DMmode_MULTF
Current modal DM correction
aol#_modeval_dm_now
Modal clipping
aol#_DMmode_LIMIT
GPU-based DM filtered write [aol#dmfw] in aol0RT auxscripts/aolmcoeffs2dmmap GPU or CPU
Modes → DM actuators
Main process [aol#run] in aol#RT script auxscripts/aolrun CPU (+ GPU)
WFS image
dark subtract
normalize
WFS pixels → modes
if CMMODE=0
Direct DM Write → actuators
loop gain
loop mult
should match
if DMprimaryWrite_ON
DM primary Write
DM "actuators"
DM filtered Write
complatency_frame (measured by aolMeasureTiming)
wfsmextrlatency_frame (measured by aolMeasureTiming)
Note: DM map & coefficients show correction applied
→ open loop = WFS residual – dm
→ Wfresidual = Open loop WF + dm
→ dm = Wfresidual – open loop

Figure 2. Advanced modal control with cacao. Each small gray rectangular box is a data stream. Modal coefficients are first computed from the WFS image (lower left gray box and pink box). The modal coefficients are then filtered, pseudo-open loop coefficients are computed, and predictive control is applied (large green box). Predictive control processed are at the top of this figure. The output control modes are converted to DM actuator displacements (blue/gray box in lower right).

3. The **Data logger** process is unblocked, as semaphore #0 of WFS image is equal to 1. The process starts data logging of the frame and decrements semaphore #0 back to 0. Upon completion, the process returns to the wait for semaphore #0 state.

4. The **Send to other computer** process is unblocked, and follows the same steps as the data logger process.

5. The **Multiply by Control Matrix** process is unblocked and starts computing the DM solution. It decrements semaphore #3 back to zero as soon as it starts computation.

6. The **Multiply by Control Matrix** process completes, and posts all semaphores in the incremental DM displacement data stream.

7. The **Multiply by gain and add** process in unblocked by semaphore #2: it decrements semaphore #2 back to zero and starts computing.

8. The **Multiply by gain and add** process completes: it writes the Total DM displacement shared memory and posts its semaphores.

9. The **DM electronics driver** process is unblocked: it decrements semaphore #2 of the Total DM displacement data stream, and starts its computation.

10. Commands are sent to the DM.

Note that computation steps can overlap, so that the total time to complete all steps can be longer than the WFS frame rate: step #9 may start while the next WFS arrives (step #1).

This simple example illustrates cacao's modular approach to AO control control. More advanced AO loop constructions follow the same principle, where data streams are usually written by a single process and trigger multiple computations. A block diagram for the more advanced modal control employed on the Subaru Coronagraphic Extreme Adaptive Optics (SCExAO) system[3, 4] is shown in Figure 2. Additionally, multiple control loops may be linked together on the same system, as described in §5.2.

The user/programmer must carefully allocate resources to processes: processes that may overlap should be deployed on independent CPU cores or GPUs to avoid excessive latency, and processes sharing data streams should use cores on the same physical CPU to optimize data locality.

2.3 Deformable Mirror Input Channels

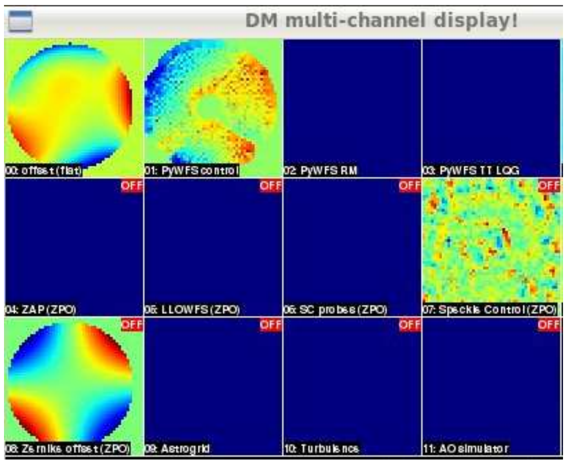


Figure 3. Multi-DM display window. Twelve DM channels are created by cacao for a single physical DM. When any of the channels is modified, their displacements are added into a single total dispacement map that is sent to the physical DM.

cacao's processes do not write directly on the DM; instead, a number of DM channels (default: 12) are created, and a low-latency process monitors all channels for change. When one of the channels is updated, the displacements are summed across all channels, and the total displacement is sent to the DM. This interface enables multiple processes to address the same physical DM without having to handle complex timing issues and race conditions. This feature is widely used by cacao, and enables, for example, the following operation modes:

- Testing: Wavefront errors (such as simulated turbulence) may be added to one of the DM channel while cacao's main control writes on another channel to test AO loop performance.
- Astrometric calibration: An astrometric grid[5] can be written on one of the channels.
- Acquiring a response matrix while the AO loop is running

This feature is augmented by configuring some of the DM channels as ***offset channels***: in addition to their displacements being sent to the DM, they are multiplied by the system response matrix to offset the WFS reference, so that the corresponding displacement will not be corrected by the AO loop.

3. CAPABILITIES AND OPERATION MODES

3.1 Zonal vs. Modal control

cacao supports both zonal and modal control.

In zonal control mode, as shown in Figure 1, the control matrix transforms WFS signals directly into DM actuator displacements. The computation steps are: (1) normalize WFS image and subtract reference, (2) multiply the resulting WFS signal by the control matrix and (3) Multiply solution by control gain and subtract to current DM state. This is the simplest and fastest AO control scheme, computationally dominated by the matrix-vector multiply (MVM) operation. The user can allocate multiple GPUs for this MVM, so high frame rate and large actuator counts can be supported.

The modal control scheme is computationally more demanding and generally slower, but far more capable, as shown in Figure 2. The first MVM transforms WFS signal into wavefront mode coefficients. The coefficients are then filtered (gain, clipping) before being converted into DM actuator displacements. This provides real-time access to modal coefficients, and many of cacao's advanced control features such as real-time pseudo-open loop modal reconstruction and predictive control.

3.2 Computation speed and computing hardware

In both zonal and modal control schemes, the computation load is dominated by MVMs. They can be deployed on CPU or GPUs, and cacao's interface allows multiple GPUs to share the load of a single MVM. Computation times for the full-featured modal control on the SCExAO system are given in Table 1, along with the hardware latency to be discussed in §4.2.

3.3 Real-time pseudo-open loop reconstruction

cacao's modal control scheme includes real-time reconstruction of pseudo-open loop wavefronts: the measured residual wavefront error is added to the DM correction to estimate the wavefront at the entrance of the AO system prior to correction. The WFS and DM signals need to be co-aligned in time, as the WFS latency is usually larger than the DM latency, so the last WFS measurement available is older and the DM signals should be shifted back in time.

The pseudo-open loop reconstruction is essential for AO control loop optimization, and is the basis for cacao's predictive control optimization, as detailed in §5.3.

Table 1. Timing measurements in full-featured modal control mode (14,400 sensors, 2000 actuators, 3.5 kHz frame rate)

Operation	Hardware	Time [μs]	Frame(s)	Notes
COMPUTE MODAL COEFFICIENTS: 14,400 sensors $->$ 1096 modes				
LOAD WFS IMAGE	CPU	3.406	1.19 %	
DARK SUBTRACT	CPU	11.009	3.85 %	Can be multithreaded
COMPUTE WFS IMAGE TOTAL	CPU	1.174	0.41 %	
NORMALIZE WFS IMAGE	CPU	4.980	1.74 %	uses previous frame total
SUBTRACT WFS REFERENCE	CPU	10.747	3.76 %	Can be multithreaded
CONTROL MATRIX MVM SETUP	CPU	0.417	0.15 %	
TRANSFER WFS TO GPU	CPU-GPU	10.050	3.52 %	Over PCIe3.0x16 bus
MVM COMPUTE	GPU(2)	143.810	50.33 %	2x NVIDIA GTX 980Ti
TRANSFER TO CPU THREADS	GPU-CPU	5.970	2.09 %	Over PCIe3.0x16 bus
ADD THREADS RESULTS	CPU	2.017	0.71 %	
MISC	CPU	14.590	5.01 %	
TOTAL	ALL	208.170	72.85 %	
IDLE TIME / WAIT FOR WFS	CPU	77.566	27.15 %	
PROCESS MODAL COEFFICIENTS (incl. PREDICTIVE CONTROL), 1096 modes				
MODAL FILTERING	CPU	19.662	6.88 %	
MODAL COEFFICIENTS $->$ ACTUATORS: 1096 modes $->$ 2000 actuators				
MVM COMPUTE DM ACTUATORS	GPU	76.714	26.85 %	NVIDIA GTX 980Ti
SUMMARY				
TOTAL HARDWARE LATENCY	WFS+DM	875.000	3.062 frame	
TOTAL COMPUTE LATENCY	CPU+GPU	304.546	1.066 frame	
TOTAL LATENCY	ALL	1179.546	4.128 frame	

3.4 Data logging

cacao provides full speed data logging to disk. Telemetry is stored as FITS data cubes, with the 3rd axis corresponding to time. A separate timing file is written for each FITS file, so that each slice of the data cube is accurately timed. The time logged is the time at which the corresponding stream write is completed.

On the SCExAO system main loop, the data rate when all streams are saved is $\approx 2TB/hr$, dominated by the WFS images.

4. CALIBRATION AND TIMING SYNCHRONIZATION

4.1 Timing Requirements

Accurate timing knowledge is essential to high performance AO control. The very first step in setting up cacao on a new system is to run cacao's hardware latency measurement script, as described in §4.2.

For the pseudo-open loop modal reconstruction (described in §3.3) to provide accurate results, timing errors must well below the loop period (approximately $< 1/3$ of a frame). This is essential for the predictive control algorithm, at it seeks to identify linear temporal relationships from the system telemetry. The most stringent timing requirements come from the high speed response matrix acquisition described in §4.3 for which DM commands must be precisely aligned with the camera readout, so the timing errors should be below $\approx 5\%$ of a frame.

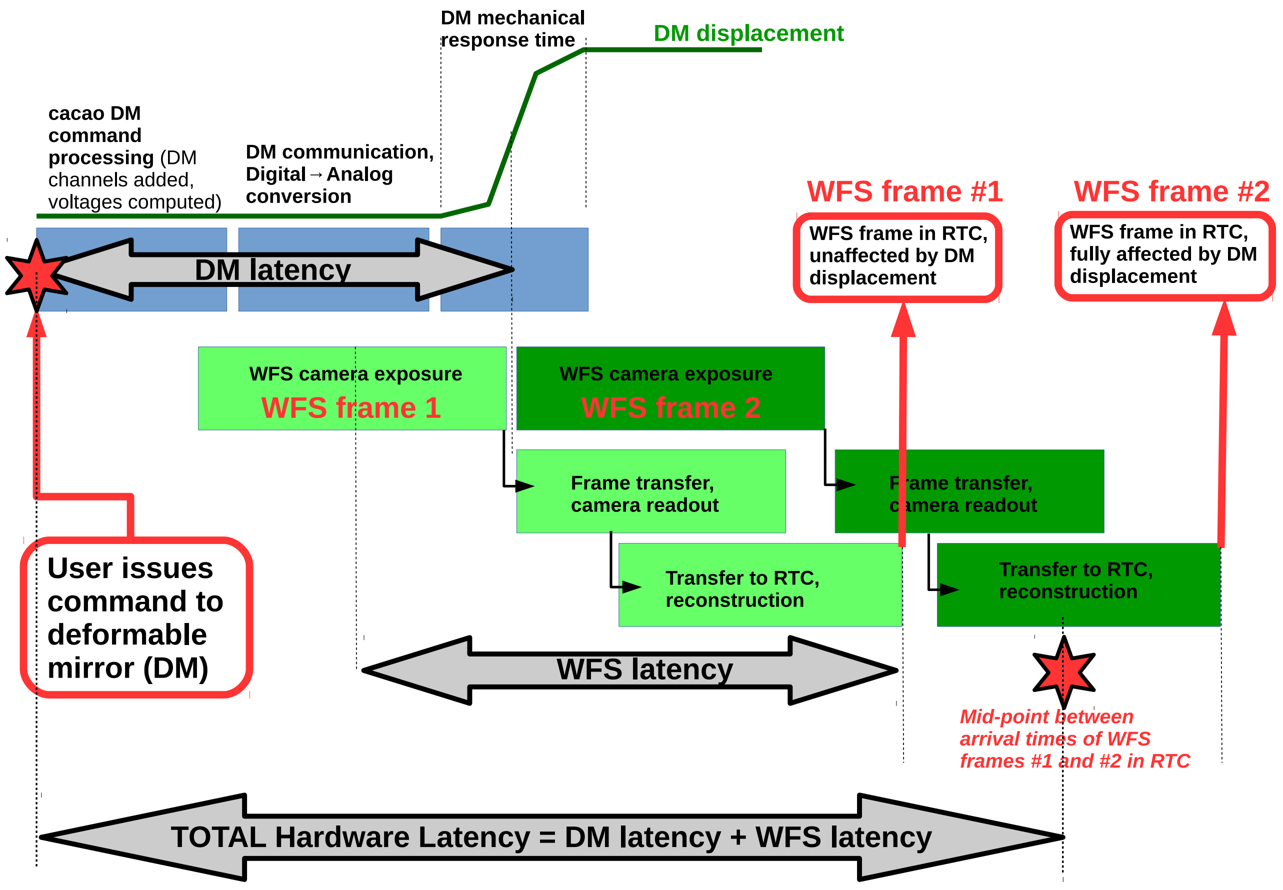

DM mechanical response time
DM displacement
cacao DM command processing (DM channels added, voltages computed)
DM communication, Digital→Analog conversion
DM latency
WFS frame #1
WFS frame in RTC, unaffected by DM displacement
WFS frame #2
WFS frame in RTC, fully affected by DM displacement
WFS camera exposure
WFS frame 1
WFS camera exposure
WFS frame 2
Frame transfer, camera readout
Frame transfer, camera readout
Transfer to RTC, reconstruction
Transfer to RTC, reconstruction
User issues command to deformable mirror (DM)
WFS latency
Mid-point between arrival times of WFS frames #1 and #2 in RTC
TOTAL Hardware Latency = DM latency + WFS latency

Figure 4. Cacao hardware latency definition.

4.2 Hardware Latency Measurement

The hardware latency, as seen from cacao, is the time delay between the instant at which a DM command is issued within cacao and the time at which the WFS data stream accessible within cacao contains the corresponding signal. It includes software communication overheads as well as actual hardware delays. The exact definition of cacao's hardware latency is shown in Figure 4: when a DM command is issued at a time such that the corresponding DM physical motion occurs exactly between two WFS frames, then the hardware latency is the time offset between the instant at which the DM command is issued by cacao, and the mid-point between the arrival times (in cacao shared memory) of the last frame unaffected by the DM motion and the first frame affected by the DM motion.

To measure latency, cacao issues a DM command and computes the sum square difference of consecutive WFS frames for a time interval following the DM command. This measurement is repeated many times, ensuring that the DM command is issued at varying fractional WFS periods. The measurement points (root sum-squared differences) can then be plotted against the time since the DM command was issued. This should produce a triangular peak, centered on the hardware latency as defined by cacao: the peak value (top of the triangle) corresponds to the DM motion falling between two frames. The triangle width should be twice the WFS period.

Figure 5 shows the measurement points for different WFSs on the SCExAO system. The points follow the expected triangular shapes, with some measurable curving due to the finite DM mechanical speed. The measured latency vary from 1715 μs for the Pyramid WFS camera running at 500 Hz, to 551 μs for the focal plane WFS camera running at 6.4 kHz.

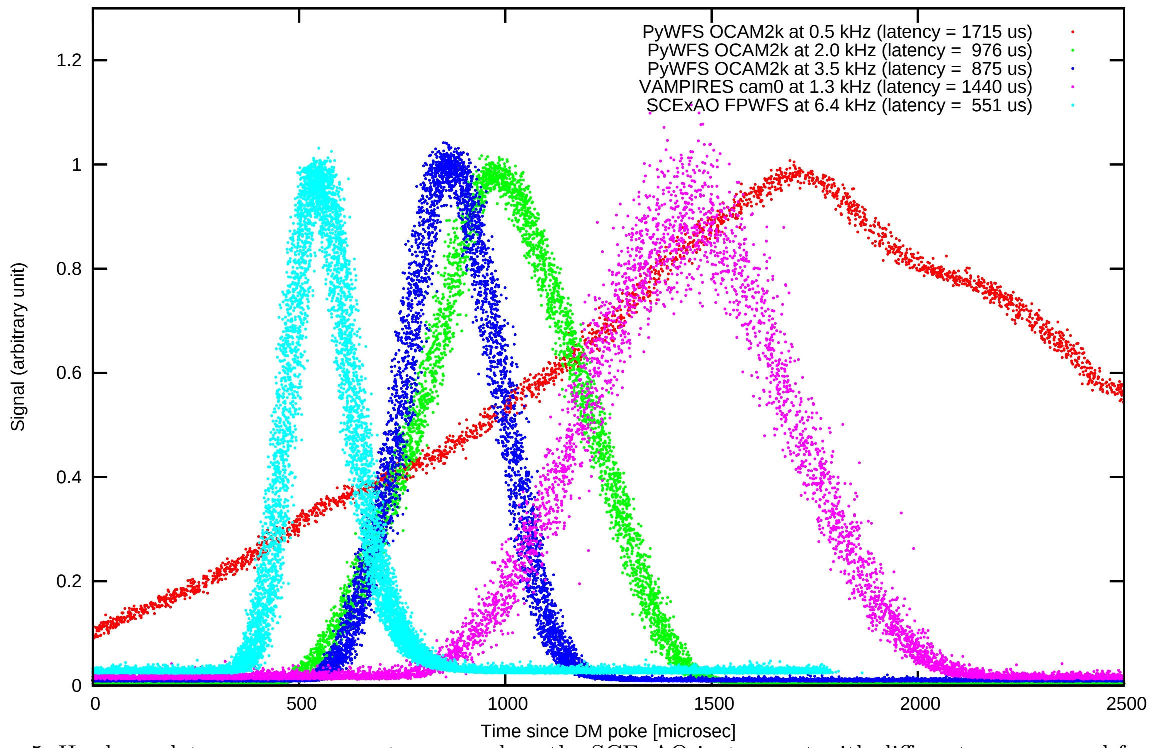


Figure 5. Hardware latency measurement measured on the SCExAO instrument with different cameras and frame rates.

4.3 Response Matrix Acquisition

Acquiring a high quality response matrix (RM) is essential for AO control loop operation, but can often be a time-consuming task. cacao's RM acquisition process acquire high SNR RMs in a short time thanks to precision timing and optimal choice of poke modes.

Thanks to accurate knowledge of hardware latency, cacao can schedule the DM pokes such that they occur between WFS frames. This enables high speed acquisition where sensing N modes is performed in $2 \times N$ WFS frames. For example, on the SCExAO system running at 3.5 kHz, the 2000-modes RM requires 4000 DM pokes (plus and minus pokes for each mode) that can be completed in $4000/3500 = 1.142$ second. Because the response is measured by differentiating WFS frames that are 286 μs apart, the high speed RM acquisition ensures that the underlying WFS (separately from the poke) has not had time to change significantly, so the measurement is robust and largely insensitive to wavefront instabilities. This becomes invaluable for acquiring RMs on-sky, as the underlying atmospheric wavefront is continuously changing. High speed RM acquisition, combined with the DM channel splitting described in §2.3, enables on-sky RM acquisition while the control loop is running: by modulating the DM at the full WFS acquisition frequency, we ensure that the closed loop correction does not have sufficient temporal bandwidth to alter the RM pokes.

The RM quality (SNR) is maximized by increasing the poke RMS amplitude. The poke peak amplitude must however be kept below the WFS linearity limit, so the optimal poke pattern for the RM acquisition is the one that simultaneously maximizes RMS displacement and minimizes the peak displacement. A zonal poke sequence (moving one actuator at a time) is the worst possible choice, and the best choice is to poke, at any given time, half of the actuators to the same positive displacement and the other half of the actuators to the opposite negative displacement. cacao adopts a Hadamard-encoded poke sequence to achieve optimal sensitivity. As shown in Figure 6 (top left), each raw Hadamard poke contains signal across the whole sensor image. The set of Hadamard pokes is decoded in a zonal RM (one slice shown in bottom left). When poking at high speed, the DM mechanical response time becomes measurable at the % level, as the time for actuators to move is no longer negligible compared to the WFS frame rate. This effect is visible in laboratory RM in the center, bottom of Figure 6: the reconstructed RM slice shows a faint ghost due to the previous pokes. This unwanted feature

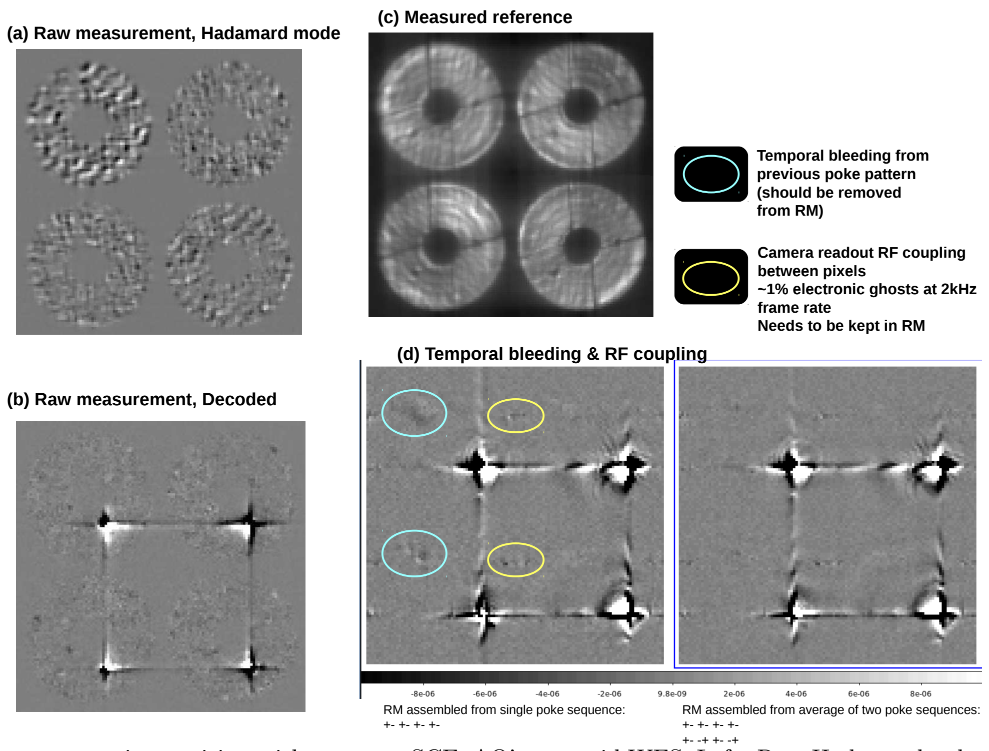

(a) Raw measurement, Hadamard mode
(c) Measured reference
Temporal bleeding from previous poke pattern (should be removed from RM)
Camera readout RF coupling between pixels
~1% electronic ghosts at 2kHz frame rate
Needs to be kept in RM
(b) Raw measurement, Decoded
(d) Temporal bleeding & RF coupling
-8e-06
-6e-06
-4e-06
-2e-06
9.8e-09
2e-06
4e-06
6e-06
8e-06
RM assembled from single poke sequence:
+- +- +- +-
RM assembled from average of two poke sequences:
+- +- +- +-
+- -+ +- -+

Figure 6. Response matrix acquision with cacao on SCExAO's pyramid WFS. Left: Raw Hadamard poke (top) and zonal-decoded poke (bottom) responses acquired on-sky at 2 kHz while the AO loop is running. Top center: WFS reference. Bottom right: laboratory RM showing temporal bleeding due to finite DM response time (left), and temporally balanced RM free of the temporal bleeding effect (right).

should not be part of the RM, and can be removed by acquiring two RMs with opposite temporal dependencies and averaging them to cancel the temporal bleeding effect. The resulting laboratory RM (bottom right) is nearly perfect, and reveals a small but measurable capacitive coupling between readout lines of the EMCCD camera (this effect is and should be part of the RM).

5. ADVANCED FEATURES

5.1 Focal plane wavefront sensing

cacao can drive any linear AO control loop, and it will calibrate linear relationships between the WFS and the DM to then run the loop. In some cases, a linear signal can be extracted from the focal plane image. In high contrast imaging, when wavefront perturbations are small, the bright parts of a focal plane PSF respond linearly to wavefront errors. Figure 7 shows a results from a cacao-run control loop using the bright half of a PSF to control the opposite dark side.[6,7] A speckle is injected (frame (b)) and successfully removed (frame (c)) by the control loop. This test was performed by pointing cacao's WFS input data stream to the bright half of the focal plane image. Sample slices from the response matrix are shown in the lower right of the figure.

5.2 Zero Point Offsetting and Linking Multiple AO loops

cacao allows for multiple WFSs to drive the same DM, and provides high-level hooks for such links to be transparent.

The linking relies on the ability to configure any of the DM channels as a zero point offset. When the user writes a displacement on a DM channel configured as a zero point offset, the displacement map is automatically multiplied by the response matrix of each WFS in the system, and the resulting WFS maps are applied as offsets to the WFS references. This process moves the convergence point of each WFS so that it will track, rather than

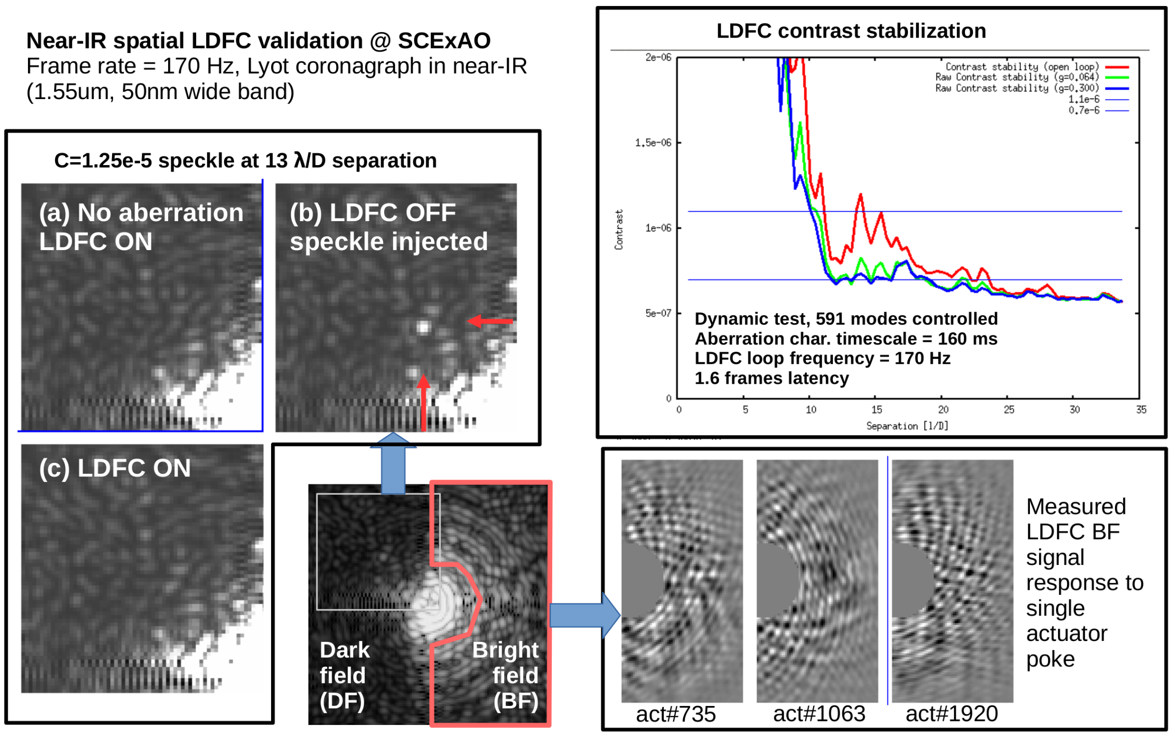


Figure 7. Laboratory focal plane control with cacao. Test performed on the SCExAO system with its internal source (off-sky). Here, cacao uses the bright half of a focal plane PSF to control the high contrast dark half, following the Linear Dark Field Control (LDFC) principle.

remove, the DM displacement added to the zero point offset DM channel. Through this process, cacao allows for any control loop to offset the convergence point of another loop: the new loop, instead of writing on a physical DM, writes on a channel of the primary loop DM pre-configured as a zero point offset.

On the SCExAO system, this feature is used for focal plane wavefront control (speckle sensing and control) as well as for the coronagraphic low-order wavefront sensor.

5.3 Predictive Control and Sensor Fusion

cacao includes a linear machine learning (LML) engine that can be used for sensor fusion and/or predictive control. The engine works by finding the least square solution to the linear relationship between input and output entries selected from the system telemetry. When configured for predictive control the LML engine input is the set of N consecutive pseudo-open loop mode coefficients vectors, and the output is the corresponding future measured pseudo-open loop mode coefficients vector.[8] If the system latency is 4 frames, then the output is 4 frames after the most recent of the N input vectors. The program scans the recent telemetry (typically the last 100 sec) to assemble a training set for the supervised machine learning engine. For example, if the loop runs at 2kHz and control M modes, then 200,000 sample of input vectors (each $N \times M$ long) and the corresponding output vectors (each M long) will be assembled. The supervised LML will then find the linear relationship (a matrix of size NM by N) that, in the least square sense, best links the input to the output. Once this relation ship has been found, it can be exercised in real-time to estimate, based on the most recent N pseudo-open loop measurements, what will be the pseudo-open loop measurement in the near future. The same framework can also be used to combine multiple sensors (sensor fusion) to better estimate the state of the system: there is no algorithmic difference between sensor fusion and prediction. Figure 8 shows on-sky results obtained with the predictive control using the LML engine. A significant improvement in both PSF surface brightness and stability

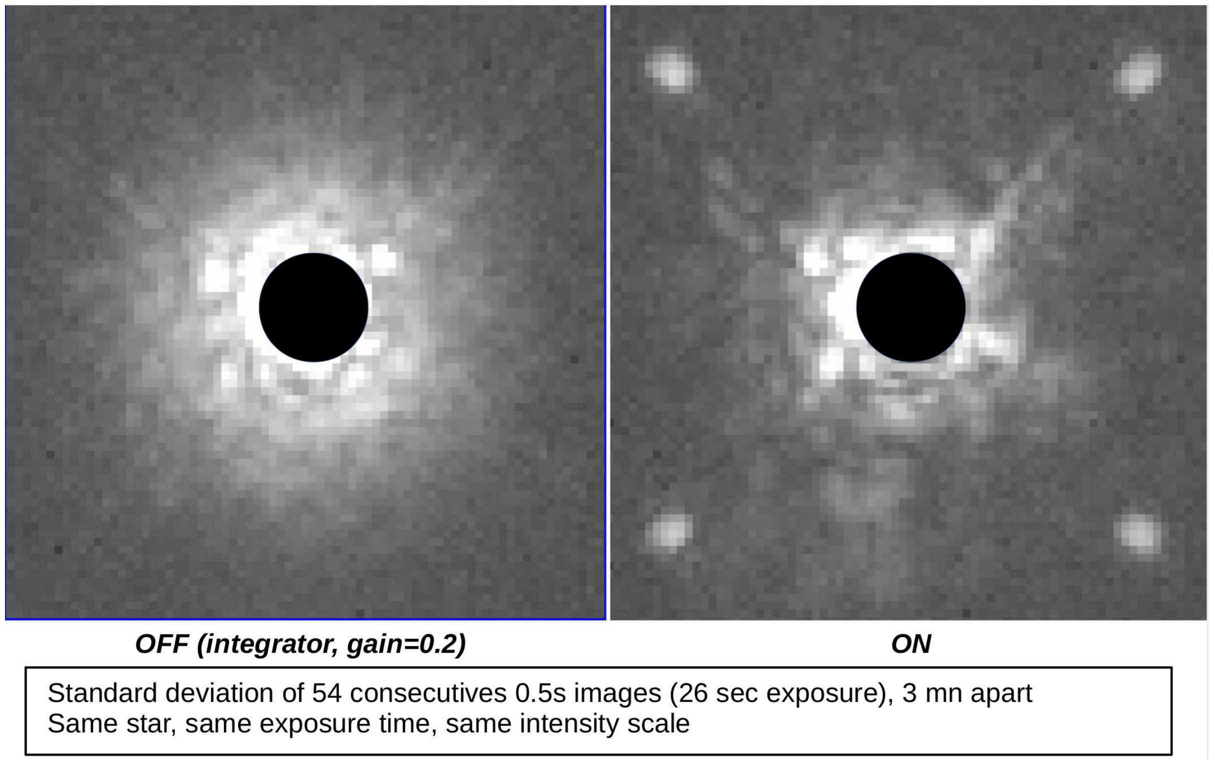

OFF (integrator, gain=0.2)
ON
Standard deviation of 54 consecutives 0.5s images (26 sec exposure), 3 mn apart
Same star, same exposure time, same intensity scale

Figure 8. On-sky predictive control demonstration with cacao on the SCExAO system. PSF standard deviation without (left) and with (right) predictive control.

was observed. We note that the contrast gain is not as large as predicted,[8,9] which may be due to non-stationary in atmosphere turbulence statistics.

6. CONCLUSIONS, FUTURE WORK

cacao provides an open-source modular framework for AO loop control. It is scalable in both compute performance and its ability to tackle complex AO challenges. It can be configured to run a simple AO control loop with a single DM and WFS, or drive multiple DMs from multiple WFSs with complex relationships. Optional modules offer advanced operation modes and enhanced performance such as predictive control.

cacao is still under heavy development, and much of its development and on-sky validation has been done for the Subaru Coronagraphic Extreme AO (SCExAO) system. Additional current and future AO system are planning on-sky use of the cacao framework.[10,11] To support wider adoption the cacao development team is improving its user interface and expanding its capabilities by providing new high performance compute libraries,[12] better integration with GPGPU hardware, and exploring FPGA support.[13] cacao is being integrated with AO simulations tools developed as part of the Green Flash project.[14,15]

As an open-source project written by and for the AO community, the project welcomes new programmers and users.

ACKNOWLEDGMENTS

Developers of the cacao software package receive funding from the National Science Foundation (MagAO-X project), the Japanese Society for the Promotion of Science (SCExAO project), the Japanese NINS Astrobiology Center, and the European Union (project Green Flash).

REFERENCES

1. R. Davies and M. Kasper, “Adaptive optics for astronomy,” *Annual Review of Astronomy and Astrophysics* **50**, pp. 305–351, 2012.
2. cacao development team, “cacao: Compute and control for adaptive optics.” https://github.com/cacao-org/cacao, 2017–2018.
3. A. Sahoo, O. Guyon, C. Clergeon, N. Skaf, Y. Minowa, and J. Lozi, “Subaru Coronagraphic Extreme-AO (SCExAO) wavefront control: current status and ongoing developments,” *Proc. SPIE* **10703**(187), 2018.
4. J. Lozi, O. Guyon, N. Jovanovic, S. Goebel, P. Pathak, N. Skaf, A. Sahoo, B. Norris, F. Martinache, M. NDiaye, B. Mazin, A. Walter, P. Tuthill, T. Kudo, H. Kawahara, T. Kotani, M. Ireland, N. Cvetojevic, E. Huby, S. Lacour, S. Vievard, T. D. Groff, J. K. Chilcote, J. Kasdin, J. Knight, F. Snik, D. Doelman, Y. Minowa, C. Clergeon, N. Takato, M. Tamura, T. Currie, H. Takami, and M. Hayashi, “SCExAO, an instrument with a dual purpose: perform cutting-edge science and develop new technologies,” *Proc. SPIE* **10703**(270), 2018.
5. N. Jovanovic, O. Guyon, F. Martinache, P. Pathak, J. Hagelberg, and T. Kudo, “Artificial Incoherent Speckles Enable Precision Astrometry and Photometry in High-contrast Imaging,” *ApJ* **813**, p. L24, Nov. 2015.
6. O. Guyon, K. Miller, J. Males, R. Belikov, and B. Kern, “Spectral Linear Dark Field Control: Stabilizing Deep Contrast for Exoplanet Imaging Using out-of-band Speckle Field,” *ArXiv e-prints* , June 2017.
7. K. Miller, O. Guyon, and J. Males, “Spatial linear dark field control: stabilizing deep contrast for exoplanet imaging using bright speckles,” *Journal of Astronomical Telescopes, Instruments, and Systems* **3**, p. 049002, Oct. 2017.
8. O. Guyon and J. Males, “Adaptive Optics Predictive Control with Empirical Orthogonal Functions (EOFs),” *ArXiv e-prints* , July 2017.
9. J. R. Males and O. Guyon, “Ground-based adaptive optics coronagraphic performance under closed-loop predictive control,” *Journal of Astronomical Telescopes, Instruments, and Systems* **4**, p. 019001, Jan. 2018.
10. J. R. Males, L. M. Close, K. Miller, L. Schatz, D. Doelman, J. Lumbres, F. Snik, A. Rodack, J. Knight, K. Van Gorkom, J. D. Long, A. Hedglen, M. Kautz, N. Jovanovic, K. Morzinski, O. Guyon, E. Douglas, K. B. Follette, J. Lozi, C. Bohlman, O. Durney, V. Gasho, P. Hinz, M. Ireland, M. Jean, C. Keller, M. Kenworthy, B. Mazin, J. Noenickx, D. Alfred, K. Perez, A. Sanchez, C. Sauve, A. Weinberger, and A. Conrad, “MagAO-X: project status and first laboratory results,” *Proc. SPIE* **10703**(9), 2018.
11. D. Mawet, C. Bond, J.-R. Delorme, N. Jovanovic, M. Chun, D. Hall, S. Cetre, S. Lilley, O. Guyon, J. Wallace, and P. Wizinovich, “Keck Planet Imager and Characterizer: status update,” *Proc. SPIE* **10703**(6), 2018.
12. H. Ltaief, D. Sukkari, O. Guyon, and D. E. Keyes, “Extreme computing for extreme adaptive optics: The key to finding life outside our solar system,” *PASC ’18: Proceedings of the Platform for Advanced Scientific Computing Conference*, 2018.
13. D. Perret, M. Lainé, J. Bernard, D. Gratadour, and A. Sevin, “Bridging FPGA and GPU technologies for AO real-time control,” in *Adaptive Optics Systems V*, *Proc. SPIE* **9909**, p. 99094M, July 2016.
14. D. Gratadour, N. Dipper, R. Biasi, H. Deneux, J. Bernard, J. Brule, R. Dembet, N. Doucet, F. Ferreira, E. Gendron, M. Laine, D. Perret, G. Rousset, A. Sevin, U. Bitenc, D. Geng, E. Younger, M. Andrighettoni, G. Angerer, C. Patauner, D. Pescoller, F. Porta, G. Dufourcq, A. Flaischer, J.-B. Leclere, A. Nai, P. Palazzari, D. Pretet, and C. Rouaud, “Green FLASH: energy efficient real-time control for AO,” in *Adaptive Optics Systems V*, *Proc. SPIE* **9909**, p. 99094I, July 2016.
15. D. Gratadour, F. Ferreira, A. Sevin, N. Doucet, Y. Clénet, E. Gendron, M. Lainé, F. Vidal, J. Brulé, M. Puech, C. Vérinaud, and A. Carlotti, “COMPASS: status update and long term development plan,” in *Adaptive Optics Systems V*, *Proc. SPIE* **9909**, p. 990971, July 2016.