Adaptive optics real-time control with the compute and control for adaptive optics (Cacao) software framework*

Olivier Guyon^{a,b,c,d}, Arnaud Sevin^e, Florian Ferreira^e, Hatem Ltaief^f, Jared Males^b, Vincent Deo^{a,e}, Damien Gratadour^{e,g}, Sylvain Cetre^h, Frantz Martinacheⁱ, Julien Lozi^a, Sebastien Vievard^a, Neelay Fruitwala^j, Steven Bos^k, and Nour Skaf^{a,e,l}

^aSubaru Telescope, National Astronomical Observatory of Japan, National Institutes of Natural Sciences (NINS), 650 North A'ohōkū Place, Hilo, HI, 96720, U.S.A.

^bSteward Observatory, University of Arizona, Tucson, AZ, USA

^cJames C. Wyant College of Optical Sciences, University of Arizona, Tucson, AZ, USA

^dAstrobiology Center of NINS, 2-21-1, Osawa, Mitaka, Tokyo, Japan

^eLESIA, Observatoire de Paris, Université PSL, CNRS, Sorbonne Université, Université de Paris, 5 place Jules Janssen, 92195 Meudon, France

^fExtreme Computing Research Center, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

^gResearch School of Astronomy and Astrophysics, Australian National University, Canberra, ACT 2611, Australia

^hW. M. Keck Observatory, 65-1120 Mamalahoa Hwy, Kamuela, HI, USA

ⁱUniversite Cote d'Azur, Observatoire de la Cote d'Azur, CNRS, Laboratoire Lagrange, France

^jDepartment of Physics, University of California, Santa Barbara, 93106, U.S.A.

^kLeiden Observatory, Leiden University, PO Box 9513, 2300 RA Leiden, The Netherlands

^lDepartment of Physics and Astronomy, University College London, London, United Kingdom

ABSTRACT

The Compute and control for adaptive optics (Cacao) is an open source software package providing a flexible framework for deploying real-time adaptive optics control. Cacao leverages CPU and GPU computational resources to meet the demands of modern AO systems with thousands of degrees of freedom running at kHz speed or faster.

Cacao adopts a modular approach, where individual processes operate over a standardized data stream structure. Advanced control loops integrating multiple sensors and DMs are built by assembling multiple such processes. High-level constructs are provided for sensor fusion, where multiple sensors can drive a single physical DM. The common data stream format is at the heart of Cacao, holding data content in shared memory and timing information as semaphores.

Cacao is currently in operation on the general-purpose Subaru AO188 system, the SCExAO and MagAO-X extreme-AO instruments. Its data stream format has been adopted at Keck, within the COMPASS AO simulation tool, and in the COSMIC modular RTC platform.

We describe Cacao's software architecture and toolset, and provide simple examples for users to build a real-time control loop. Advanced features are discussed, including on-sky results and experience with predictive control and sensor fusion. Future development plans will include leveraging machine learning algorithms for real-time PSF calibration and more optimal AO control, for which early on-sky demonstration will be presented

Keywords: Adaptive Optics, Real-time systems, Wavefront control, High-performance computing

* Based on data collected at Subaru Telescope, which is operated by the National Astronomical Observatory of Japan.

Further author information: (Send correspondence to O.G.)

O.G.: E-mail: guyon@naoj.org, Telephone: 1 818 293 8826

1. INTRODUCTION

Adaptive Optics (AO) systems now equip most large astronomical telescopes, providing angular resolution and sensitivity far beyond seeing-limited imaging.^{1,2} AO systems have become considerably more advanced over time, and have branched in multiple directions. While the first AO systems consisted of a single wavefront sensor (WFS) driving a single deformable mirror (DM), current and future systems are comprised of multiple WFSs and DMs operating in complex overlapping control loops. Modern and future AO systems can combine multiple types of WFSs, natural and laser guide stars, and deformable mirrors in order to increase field of view while controlling anisoplanatism, improving the sky coverage,³ or perform high-accuracy wavefront correction for high contrast imaging.⁴

The real-time controller is an essential component of any AO system, processing WFS input data as to compute DM control signals, now typically at frequencies in the 500 Hz – 3 kHz range. The computational requirements of AO systems are thus strongly affected by the curse of dimensionality, and with now systems with thousands of degrees of freedom, the computation of the DM command within the allotted time vastly exceeds the computational capability of a single CPU core. Thus, a strong effort has been driven to deploy AO RTCs onto many-core architectures,^{5,6} or dedicated high-performance coprocessors, such as Xeon Phi,⁷ general purpose graphical units^{8,9} (GPGPUs), or FPGAs.¹⁰

Cacao¹¹ was originally developed to support research and development for extreme-AO (xAO) systems performing high contrast imaging of exoplanets and circumstellar disks. Its goal is to provide a flexible framework for rapid development, implementation and testing of new wavefront sensing and control (WFS/C) concepts. Cacao aims at being both a flexible research tool and a reliable framework for deploying AO control software to be used for routine science operations. This is achieved with a modular design following a common software framework, and by providing tools for managing data and real-time processes. This modular design, based on two fundamental structures of “stream” – the data holding unit – and “process” – the data transfer and processing unit between streams, has proven versatile and successful for the rapid development and deployment of new systems. Cacao is currently in operation on the general-purpose Subaru AO188 system, the SCExAO¹² and MagAO-X¹³ extreme-AO instruments. Its data stream format has been adopted at Keck,¹⁴ within the COMPASS AO simulation tool,^{9,15,16} and in the COSMIC modular RTC platform.^{17,18}

We describe in §2 the framework within which Cacao is implemented, its fundamental building blocks and tools. Cacao’s key features are discussed in §3, and examples leveraging them presented in §4.

2. KEY ELEMENTS AND CONCEPTS

Cacao implements an adaptive optics control loop as a chain of small self-contained processes, each performing a specific task, such as acquiring images from a camera, or multiplying an input vector by a matrix. A simple example is illustrated on fig. 1, showing a simplified process flow for an AO loop from the camera to the DM command, on which we are going to elaborate. The example is structured around six processes, four of which are part of the principal data flow. The execution flow is sequenced through the three streams.

2.1 Processes

The processes are the fundamental computational units within the Cacao framework. They seek their input data in one or several streams (details on the stream structure are given in the next subsection), and publish their output data in other streams. One input stream may be the source of a software clock for the process. For the example given in fig. 1, the framegrabbing process [p0.0] is, at the simplest, triggered by a hardware clock over the cameralink cable or any other protocol. Thus, process [p0.0] sets its own pace for publishing data to stream [s1], as long as the operations performed by process [p0.0] keep up with the framerate of the camera. All other processes shown in fig. 1 have a clocking stream, and their execution, be it the linear AO control, the integrator, or the DM command management, is triggered by the semaphores of their clocking streams. There can also be a number of non-clocking, data holding streams that are not shown in fig. 1: for process [p2.2], the AO integrator, the integrator gain, leak factor, the current integrator buffer, etc, are all fetched from various shared memory streams.

Figure 1. A control loop is implemented as a collection of processes [pX.Y] (grey boxes) linked together by streams [sX] (blue rectangles). Signaling between streams and processes is achieved through POSIX semaphores.

With the same versatility, Cacao encompasses pure-software processes, from stream to stream, such as the matrix-vector multiply or the integrator. The data logging, DM driver, or network sending processes bind Cacao directly to various hardware (or kernel, or non-cacao software) sinkpoints.

In recent months, a strong community-oriented effort has been done to provide easy templates for external collaborators to develop custom processes. The connection to an input trigger stream, a data stream, and an output stream can be summarized in just a few pragmas (here, in non-contractual pseudocode):

```
int main() {
    // Initialization
    in_stream = Register_Clocking_Stream("clock_stream_name");
    data_stream = Register_Input_Stream("in_stream_name");
    output_stream = Register_Output_Stream("out_stream_name");

    process_manager->start(); // Flag the process as running to the overseer

    while(running) {
        // Housekeeping to maintain process manager structure //

        in_stream->wait_for_data(); // Blocking call on a semaphore
        data_stream->get_data(); // Non-blocking call to acquire parameters

        process_manager->start_compute(); // Advertise computation for timings
        // Do computations ... //
```

```

...
// thread safe call to write output
output_stream->publish_data(results_of_this_process);

    process_manager->finish_compute(); // Advertise computation for timings
}
process_manager->terminate(); // Flag this process as finished
}

```

The various calls to `process_manager` in this example above allow the seamless integration in a number of process monitoring tools, some of which are presented in sect. 2.3 thereafter.

2.2 Streams

For all real-time data, Cacao uses a common format, hosted in persistent, shared memory on the RTC. The stream format is based on a open source C implementation that supports all integer and floating point data formats, pieced together with metadata, keywords, and semaphores in a C struct.

- Data: 1, 2 or 3D (cubed data or 2D rolling buffers), integer, float or complex
- Metadata: name, size, type, creation and modification times
- Optional arbitrary number of keywords - accessible and settable at runtime, mostly used to broadcast camera exposure times, etc.
- Pointers to POSIX semaphores (by default, 10 per stream)

The shared memory supports a filesystem representation and thus streams can be inspected by dedicated tools directly in a ramdisk. Cacao provides a number of basic tools to overview all the stream on the systems, as well as to inspect the contents and semaphore timings of any stream.

Streams are not necessarily owned by any system process and can be considered as autonomous interprocess communication (IPC) vectors. However, in normal use they shall only be written by a single process, although their writing is thread-safe, using an atomically set flag directly within the stream header. At the end of the write operation, the upstream process increments all the semaphores of the streams, then releases the *writing* lock.

Each downstream process monitors a single of the 10 stream semaphores. Semaphore allocation is performed dynamically, and a table of semaphore ownership is maintained within the stream structure, allowing to remove dead PIDs and reallocate semaphores for a long duration without compromising the read capability. Upon the increment of its semaphore, downstream processes access the stream data as soon as the OS scheduler will let them - with a real-time Linux kernel, dedicated cpusets and real-time scheduling, we achieved wake up time consistently below 5 μ s on the SCEXAO RTC. There is –voluntarily– no data protection, no thread-safe implementation on the read side of the streams. A downstream process may only eventually detect that it is “running late” when an increment of the trigerring semaphore has already occurred when the computation is finished. By design, read thread-safety is not desired - for no reason we want, in an AO RTC, to enable a downstream process to slow down an upstream process by mean of software locks. It is thus the responsibility of the user, per means of various inspection tools provided in the Cacao suite, to set the framerate of all the “hardware clocked” processes (typically, cameras) to ensure the entire chain of processes and streams can match their speed.

Although it is the core of the design, real-time operation is not necessary; the stream structure is also extensively used as a data passing structure between non-real-time userland processes. This is noticeably enabled by a number of high-level language bindings to the stream data structure, e.g., `pyMilk`[†], that offers very simple python bindings to create, write, and read streams. This enables (non-real-time) interactive stripping on the fly directly on the RTC, e.g., collecting telemetry, processing it and updating AO control structures directly, which is often done in the way of prototyping novel WFS/C algorithms.

[†]www.github.com/milk-org/pyMilk

2.3 Management tools

Of the tools provided in the Cacao suite, two text based interfaces allow the user to monitor all the streams and compliant processes on the machine - the process controller and the stream controller.

2.3.1 Stream Control

Figure 2. Screen capture of the stream control TUI

An example of the stream control TUI is shown on fig. 2, showing one stream “ocam2d”, of data type UINT16, running at 1412 Hz, and its current semaphore values. This tool enables the inspection of all the streams on a system, as well as inspecting the causal dependencies between them through the simultaneous inspection of PIDs registered for read and write. Thus, it enables a user-friendly way to analyze –and troubleshoot– series of dependencies among the number of streams involved in one or several AO loop.

2.3.2 Process Control

Figure 3. Screen captures of the process control TUI. Top: resource allocation, bottom: timing reviews

The second monitoring tool is the process controller. It handles the standardized information registered by the various Cacao processes to the centralized process overseer structure (also a shared memory, the one referred to as **process_manager** in the pseudocode above, and also harvests a number of process monitoring informations from the Linux sysfs. On fig. 3 are shown the two main pages of the process control TUI. On top, the resource allocation overview, showing two processes (lines), one of which has multiple threads (dm00-comb). Columns represent the 36 cores of the machine, 14 of which are left for general usage and 22 reserved (through the kernel option **isolcpus**) for custom real-time processes. Various indicators around the interface display the number of context switches, the cpu load, the idle times, etc. The bottom panel of fig. 3 shows the timing monitor page - again showing 3 processes, dm00-comb (see sect. 3.1, and two network receivers, which grab stream over TCP from matching transmitters on other servers. The timing displays shows the status of each process, the mean and extrema of its execution time and of its looping time, and finally the load on its dedicated CPUs.

3. KEY FEATURES

3.1 Deformable Mirror (DM) channels

Under Cacao, the DM is controlled through an intermediate layer consisting of multiple channels (12 by default). A low-latency process monitors the DM channels for change, and recomputes the sum displacement of all channels whenever a change is detected. This sum displacement is then sent to the physical DM.

Each channel can be owned by a separate process or algorithm, allowing for multiple processes to simultaneously control the DM without collision. This feature is used, for example, to inject simulated turbulence to a DM channel and measure how well the AO control loop compensates for it. One of the DM channels is allocated to response matrix (RM) acquisition, allowing to the RM to be acquired while the AO loop is closed.

3.2 Zero point offloading

Any of the DM channels can be configured as a zero-point offset to the control loop. When configured as such, any update the DM channel is also applied as an offset to the WFS reference, so that the corresponding DM displacement is not corrected by the AO control loop. This is achieved by multiplying the DM channel by the response matrix and adding the resulting WFS vector to the WFS zero point to which the AO loop converges.

This feature, in combination with the DM channels described in §3.1, allows for processes to issue wavefront offsets without having the main AO loop correct them.

Figure 4. Latency measurement, as measured on the SCExAO system with a Pyramid WFS and a MEMS type deformable mirror. The measured hardware latency is $985 \mu\text{s}$. In this setup the WFS camera and DM electronics are connected to separate computers, so WFS frames are transferred over a private TCP link between the data acquisition computer and the real-time control computer.

3.3 Timing calibration

Accurate timing knowledge is essential to many advanced control modes (see §4). Cacao provides a latency measurement function allowing for hardware latency between DM commands and WFS response to be measured accurately.

Latency is measured by sending DM commands and monitoring when the corresponding changes are seen in the incoming WFS stream. The WFS change is measured by differentiating consecutive WFS frames, as shown in Fig. 4. Latency is measured by finding the peak of the WFS change as a function of time delay between DM command and WFS arrival time. In order to fully sample the function, DM pokes are sent at time stamps uniformly distributed within a single WFS exposure. The peak of the function corresponds to a DM motion during the frame transfer of the WFS camera. With a frame transfer CCD and an instantaneous DM motion, the function plotted in Fig. 4 should be a triangle of half-width equal to the WFS single frame exposure time. As shown in Fig. 4, edges are rounded by the finite temporal response of the DM. Additionally, the function is broadened along the time axis by timing jitter and along the vertical axis by readout and photon noise in the WFS.

Figure 5. Response matrix measurement on the SCEXAO system with a Pyramid WFS and a MEMS-type DM. (a) The WFS reference shows the four pupil images characteristic of a pyramid WFS. (b) Single actuator response showing minor temporal bleeding defects. (c) Temporal bleeding removed by temporal swapping and scrambling of the poke sequence.

3.4 Response Matrix Acquisition

Cacao includes routines to measure the linear response of the WFS to DM commands, which is stored as a response matrix (RM). Thanks to accurate timing knowledge (acquired as described in §3.3), the RM is acquired at high speed so that wavefront drifts unrelated to the measurement are mitigated. This is especially important when acquiring on-sky RM, for which high speed modulation can freeze the atmospheric turbulence and minimize interaction with the AO control loop.

When acquiring a RM, both the WFS reference and the linear response to DM pokes are measured. Figure 5 shows, for the SCEXAO pyramid loop, the reference (a) response and the linear response to a single actuator poke ((b) and (c)). RM acquisition can be performed at full modulation speed, for which alternating positive and negative DM pokes are synchronized with the WFS acquisition. In this mode, a full RM for a M -actuator system operating at WFS frequency f requires $2M$ pokes (one positive and one negative for each mode or actuator), which are completed in a $2M/f$ time span. For example, a 2000-actuator system running at 2 kHz requires 2 sec for RM acquisition. Multiple such matrices are collected and averaged to increase signal-to-noise ratio. In practice, full-speed RM acquisition is sensitive to the finite time required for the DM to move, so some temporal bleeding is observed (Fig. 5 (b)).

To eliminate such features from the final averaged RM, poke sequences are temporally swapped and scrambled between individual RM acquisitions, yielding clean high quality RM (Fig 5 (c)). Under this mode, for each poke sequence $+m_k \rightarrow -m_k \rightarrow +m_{k+1} \rightarrow -m_{k+1}$ there is another poke sequence $+m_k \rightarrow -m_k \rightarrow -m_{k+1} \rightarrow +m_{k+1}$, such that the temporal bleeding between modes m_k and m_{k+1} is averaged out. Additional scrambling patterns are also provided to remove hysteresis from the RM measurement.

3.5 Pseudo-open loop real-time reconstruction

Pseudo-open loop (PsOL) telemetry is computed by adding DM commands to WFS residual measurements so that the wavefront state prior to correction is reconstructed. PsOL telemetry is essential to multiple advanced control algorithms (see §4.2 for example) and for tracking atmospheric turbulence conditions and AO control loop performance. PsOL reconstruction requires timing knowledge so that DM commands and WFS measurements can be resampled to the same temporal baseline, and RM knowledge for both WFS and DM streams to be converted to the same modal space.

Figure 6. Experimental validation of real-time pseudo-open loop reconstruction. The same sequence of input wavefront aberrations is injected on a DM channel for the three tests. The open-loop test has no DM correction, so the pseudo-open loop (PsOL) values are entirely computed by the WFS. With 25% loop gain, WFS residuals are small and most of the PsOL values originate from the DM correction. The intermediate gain value has similar DM and WFS contributions to the PsOL values. All three curves are within $\approx 10\%$ of each other, demonstrating that real-time PsOL values accurately reconstruct the incoming pre-correction wavefront independently of AO loop control parameters.

Using the timing and RM calibration acquired by the functions described in §3.3 and §3.4, Cacao computes real-time PsOL telemetry in a modal space. The PsOL computation has been validated by comparing PsOL values for multiple AO loop gains in the lab with the same wavefront errors applied, as shown in Fig. 6.

4. ADVANCED WFS/C ALGORITHMS EXAMPLES

Features described in §3 are the building blocks for implementing advanced WFS/C algorithms. Three examples are provided in this section. Further examples are provided in SPIE paper 11448-255.

4.1 Linear Dark Field Control

Linear Dark Field Control (LDFC) is a wavefront stabilization technique for high contrast imaging (HCI).¹⁹ In HCI systems, the coronagraphic image produces a high contrast focal plane area where starlight has been mostly removed, often referred to as a "dark hole" or dark field (DF). Outside the dark hole, starlight is present in bright speckles, which are generally discarded, and referred to here as the "bright field" (BF). Figure 7 (a) shows such a configuration in an image acquired on the SCEXAO system with a Lyot type coronagraph. The BF may be a spatial area on the detector, or image(s) acquired at wavelength(s) other than the high contrast wavelength range.

LDFC proposes to use the focal plane bright field (BF) as a linear wavefront sensing signal. Small wavefront perturbations will modulate the BF linearly in intensity, so a control loop could lock the BF intensity map to stabilize the wavefront. In Cacao, this is implemented by defining the WFS image as the BF area in the focal plane image. This was done on the SCEXAO system as shown in Fig. 7.

Timing calibration and RM acquisition described in §3 were used to measure the BF linear response to DM perturbations (Fig. 7 (b)). A control matrix was then computed and employed for real-time closed loop correction. Figure 7 shows that an injected speckle is successfully removed by the LDFC loop. Panel (d) demonstrates contrast stabilization with 591 modes controlled.

The LDFC test demonstrates Cacao's flexibility and its applicability to non-conventional wavefront sensing architectures, without requiring new software to be written.

Figure 7. Linear Dark Field Control control loop implemented by Cacao on the SCEXAO testbed. In this configuration, the bright half of a focal plane coronagraphic image (a) is used as the wavefront sensor. The Cacao-measured linear response is shown for 3 DM actuators in panel (b). The LDFC control loop successfully removes an injected speckle (c), and provides wavefront stabilization with 591 control modes (d).

4.2 Predictive Control

Predictive control uses temporal correlations in the incoming wavefront to optimize wavefront correction, eliminating the temporal lag term from the control loop and replacing it by a (hopefully smaller) temporal extrapolation error. Predictive control implementation on Cacao leverages multiple key functions :

- **Real-time pseudo-open loop (PsOL) telemetry** is computed using timing information and response matrix calibration
- **Temporal and spatial** modes of wavefront evolution are computed by Singular Value Decomposition (SVD) of a large data matrix constructed from the PsOL telemetry. Computations for the predictive control follow the Empirical Orthogonal Functions framework.²⁰
- **A 3-dimensional control matrix** is computed. The three axis represent respectively input PsOL mode values, output mode values, and time step.

Figure 8 (a), (b) and (c) show modal 3D control matrices for three stars of varying brightnesses, acquired on the same night. The 3D matrices are unfolded for easy visualization on the figure, and only the first 100 modes are shown (out of 800 modes processed with predictive control). Detailed inspection (d) reveals that

Figure 8. On-sky predictive control with Cacao on the SCExAO system. The modal control matrix has 3 dimensions: input pseudo-open loop mode value, output (prediction) mode value, and time step. Example matrices are shown in panels (a), (b) and (c), with time slices unfolded, and only the first 100 modes shown. A detailed view (d) reveals off-diagonal correlations between different modes. Panel (e) shows on-sky PSFs without (left) and with (right) predictive control.

4.3 Non-common path aberrations correction from focal plane image

Non-common path aberrations (NCPA) are becoming a considerable limiting source of error in HCI. A method is being developed to correct for static, quasi-static and dynamic non-common path aberrations (NCPA) with the Pyramid WFS. The Dr WHO algorithm - Direct Reinforcement Wavefront Heuristic Optimization - aims to better understand what is the *true* reference of the Pyramid WFS, which is the one compensating for NCPAs. Since NCPA keep evolving during observations, it is necessary to continuously update the WFS reference to converge to the *true* (NCPA-compensated) reference. Dr WHO proceeds as follow:

- On a 30 seconds timescale, the algorithm identifies the PSF frames with the best 10% Strehl ratio;
- For these 10% frames, the corresponding WFS measurements are taken;
- The corresponding 10% WFS measurements are averaged, weighted on the value of the Strehl ratio;
- The resulting WFS frame replaces the WFS reference, with an integrator filter;
- Repeat

As the algorithm runs, it is continuously rewarded for better focal plane images, then converges towards a better compensation of the NCPA. Figure 9 presents Dr WHO's preliminary test results on-sky.

Figure 9. Non-common path aberration correction using focal plane image feedback with the VAMPIRES camera. Over about 20 minutes of running time, the algorithm converges towards a PSF with less aberrations.

NCPA compensation with Dr WHO relies on several key flexible features of Cacao. Mainly, one key feature developed on Cacao that is especially powerful for Dr WHO is accurate data synchronization.

Data synchronization

The Pyramid WFS and the fast focal plane camera run at different frequencies and have different hardware latencies, hence the need of synchronising the data streams. A synchronization script is provided by Cacao, where three inputs are required: the two streams to synchronize - which are here the Pyramid WFS frames and the focal plane camera frames -; and the time interval, which is the frequency at which the two streams will be re-sampled. For example, if the focal plane camera runs at 1kHz, and the PWFS camera runs at 2kHz, we can resample both streams to 3kHz. Then, the synchronization script will wait for fits cubes and the timing files to appear on the hard drive, read them and re-sample them to the same time baseline. This script is optimized to handle missed frames, and if the timing isn't continuous. This scripts then provides an other stream of data cubes of the required length (which is set ahead), that are saved on the disk. Choosing the length of the data cubes is linked with the latency needed for Dr WHO.

ACKNOWLEDGMENTS

Developers of the Cacao software package receive funding from the National Science Foundation (MagAO-X project, award #1625441), the Japanese Society for the Promotion of Science (SCEXAO project), the Japanese NINS Astrobiology Center, and the European Union (project Green Flash). LDFC development at SCEXAO is supported by the NASA Strategic Astrophysics Technology (SAT) Program grant 80NSSC19K0121. Development of advanced wavefront control algorithms with Cacao is supported by NASA grant 80NSSC19K0336. The development of SCEXAO was supported by the Japan Society for the Promotion of Science (Grant-in-Aid for Research #23340051, #26220704, #23103002, #19H00703 & #19H00695), the Astrobiology Center of the National Institutes of Natural Sciences, Japan, the Mt Cuba Foundation and the director's contingency fund at Subaru Telescope. The authors wish to recognize and acknowledge the very significant cultural role and reverence that the summit of Maunakea has always had within the indigenous Hawaiian community. We are very fortunate to have the opportunity to conduct observations from this mountain.

REFERENCES

- [1] Beckers, J. M., “Adaptive Optics for Astronomy: Principles, Performance, and Applications,” *The Annual Review of Astronomy and Astrophysics* **31**, 13–62 (Jan. 1993).
- [2] Davies, R. and Kasper, M., “Adaptive Optics for Astronomy,” *The Annual Review of Astronomy and Astrophysics* **50**, 305–351 (Sept. 2012).
- [3] Beckers, J. M., “Increasing the Size of the Isoplanatic Patch with Multiconjugate Adaptive Optics,” in [*Very Large Telescopes and their Instrumentation, Vol. 2*], *European Southern Observatory Conference and Workshop Proceedings* **30**, 693 (Oct. 1988).
- [4] Guyon, O., “Extreme Adaptive Optics,” *The Annual Review of Astronomy and Astrophysics* **56**, 315–355 (Sept. 2018).

- [5] Gratadour, D., Morris, T., Biasi, R., Deneux, H., Bernard, J., Buey, J.-T., Doucet, N., Ferreira, F., Laine, M., Perret, D., Sevin, A., Basden, A., Geng, D., Osborn, J., Staykov, L., Townson, M., Younger, E., Andrighettoni, M., Patauner, C., Pescoller, D., Lemaitre, J., Palazzari, P., Pretet, D., and Rouaud, C., "Prototyping AO RTC using emerging high performance computing technologies with the Green Flash project," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 1070318 (July 2018).
- [6] Younger, E. J., Basden, A. G., Geng, D., Gratadour, D., Morris, T. J., Osborn, J., Perret, D., Sevin, A., Staykov, L., and Townson, M. J., "Middleware evaluation and selection for ELT-scale Adaptive Optics RTCS," in [*Software and Cyberinfrastructure for Astronomy V*], Guzman, J. C. and Ibsen, J., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10707**, 1070718 (July 2018).
- [7] Jenkins, D. R., Basden, A. G., Myers, R. M., Osborn, J., Townson, M. J., Reeves, A. P., Staykov, L., Younger, E. J., Geng, D., Dipper, N. A., Gratadour, D., Sevin, A., and Perret, D., "An ELT scale MCAO real-time control prototype using Xeon Phi technologies," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 1070319 (July 2018).
- [8] Bernard, J., Perret, D., Sevin, A., Laine, M., Buey, T., and Gratadour, D., "Design and performance of a scalable GPU-based AO RTC prototype," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 107034B (July 2018).
- [9] Ferreira, F., Gratadour, D., Sevin, A., Doucet, N., Vidal, F., Deo, V., and Gendron, E., "Real-time end-to-end AO simulations at ELT scale on multiple GPUs with the COMPASS platform," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 1070347 (July 2018).
- [10] Perret, D., Lainé, M., Bernard, J., Gratadour, D., and Sevin, A., "Bridging FPGA and GPU technologies for AO real-time control," in [*Adaptive Optics Systems V*], Marchetti, E., Close, L. M., and Véran, J.-P., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **9909**, 99094M (July 2016).
- [11] Guyon, O., Sevin, A., Gratadour, D., Bernard, J., Ltaief, H., Sukkari, D., Cetre, S., Skaf, N., Lozi, J., Martinache, F., Clergeon, C., Norris, B., Wong, A., and Males, J., "The compute and control for adaptive optics (CACAO) real-time control software package," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 107031E (July 2018).
- [12] Lozi, J., Guyon, O., Jovanovic, N., Goebel, S., Pathak, P., Skaf, N., Sahoo, A., Norris, B., Martinache, F., N'Diaye, M., Mazin, B., Walter, A. B., Tuthill, P., Kudo, T., Kawahara, H., Kotani, T., Ireland, M., Cvetojevic, N., Huby, E., Lacour, S., Vievard, S., Groff, T. D., Chilcote, J. K., Kasdin, J., Knight, J., Snik, F., Doelman, D., Minowa, Y., Clergeon, C., Takato, N., Tamura, M., Currie, T., Takami, H., and Hayashi, M., "SCEXAO, an instrument with a dual purpose: perform cutting-edge science and develop new technologies," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 1070359 (July 2018).
- [13] Males, J. R., Close, L. M., Miller, K., Schatz, L., Doelman, D., Lumbres, J., Snik, F., Rodack, A., Knight, J., Van Gorkom, K., Long, J. D., Hedglen, A., Kautz, M., Jovanovic, N., Morzinski, K., Guyon, O., Douglas, E., Follette, K. B., Lozi, J., Bohlman, C., Durney, O., Gasho, V., Hinz, P., Ireland, M., Jean, M., Keller, C., Kenworthy, M., Mazin, B., Noenickx, J., Alfred, D., Perez, K., Sanchez, A., Sauve, C., Weinberger, A., and Conrad, A., "MagAO-X: project status and first laboratory results," in [*Adaptive Optics Systems VI*], Close, L. M., Schreiber, L., and Schmidt, D., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **10703**, 1070309 (July 2018).
- [14] Biasi, R., "Implementation and initial test results of the new Keck real time controller," *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **11448**, 11448146 (2020).
- [15] Ferreira, F., Gratadour, D., Sevin, A., and Doucet, N., "Compass: An efficient gpu-based simulation software for adaptive optics systems," in [*2018 International Conference on High Performance Computing Simulation (HPCS)*], 180–187 (2018).

- [16] Gratadour, D., Ferreira, F., Sevin, A., Doucet, N., Clénet, Y., Gendron, E., Lainé, M., Vidal, F., Brulé, J., Puech, M., Vérinaud, C., and Carlotti, A., “COMPASS: status update and long term development plan,” in [*Adaptive Optics Systems V*], Marchetti, E., Close, L. M., and Véran, J.-P., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **9909**, 990971 (July 2016).
- [17] Gratadour, D., Ferreira, F., Sevin, A., Doucet, N., Bernard, J., Biasi, R., and Rigaut, F., “MAVIS real-time control system,” *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **11448**, 11448144 (2020).
- [18] Ferreira, F., Sevin, A., Bernard, J. B., Guyon, O., and Gratadour, D., “Hard real-time core software of the AO RTC COSMIC platform: architecture and performance,” *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **11448**, 11448172 (2020).
- [19] Miller, K., Guyon, O., and Males, J., “Spatial linear dark field control: stabilizing deep contrast for exoplanet imaging using bright speckles,” *Journal of Astronomical Telescopes, Instruments, and Systems* **3**, 049002 (Oct 2017).
- [20] Guyon, O. and Males, J., “Adaptive Optics Predictive Control with Empirical Orthogonal Functions (EOFs),” *arXiv e-prints*, arXiv:1707.00570 (July 2017).