Fast End-to-End Multi-Conjugate AO Simulations Using Graphical Processing Units and the MAOS Simulation Code

Lianqi Wang[1a] and Brent Ellerbroek[1]

TMT Observatory Corportaion, 1111 South Arroyo Pkwy Suite 200, Pasadena, CA, 91105

Abstract. The Multi-threaded Adaptive Optics Simulator (MAOS) was developed at TMT to efficiently simulate various kind of AO systems. In particular, it can finish a time step of full end-to-end simulation of an ELT size multi-conjugate AO system in 1 second on 8 contemporary cpu cores. We recently ported it to run on graphical processing units (GPUs) using the Nvidia CUDA technology. A 10 fold speed up is obtained with two GTX 580 GPUs, with each time step taking only 0.1 second. A single GPU can finish 30 iterations of the conjugate gradients (CG) tomography algorithm in 30 ms, or 3 iterations of the fourier domain preconditional CG in 5 ms, which is on the same order as the $\sim$ 1ms requirement.

1 Introduction

The TMT Multi-threaded Adaptive Optics Simulator (MAOS) is a reimplementation of the algorithms and routines in the original, MATLAB-based linear AO simulator (LAOS). The motivation to develop this software was to create an adaptive optics simulator that runs fast, consumes less memory, and does the job without MATLAB, which is proprietary and has a large memory footprint. MAOS is written in C language with a function oriented design. The code is completely configurable through easy-to-read configuration files and command line options, and thus very suitable for exploring a large parameter space. The code tries its best to check the configuration for any apparent errors or conflicts. Performance critical components of MAOS have recently been ported to run on graphical processing units (GPUs) using the Nvidia CUDA technology. A ten fold speed up was obtained with two Nvidia GTX 580 GPUs. The up-to-date version of this documentation and MAOS code can be obtained from
`https://github.com/lianqiw/maos/`

The rest of the paper is organized as follows. Section 2 describes the AO simulations done in MAOS. Section 3 describes the software implementations. Section 4 describes porting MAOS on GPUs. Finally Section 5 gives the conclusion.

2 Adaptive Optics Simulations

Figure 1 shows the block diagram of AO simulations implemented in MAOS. Atmospheric turbulence is represented as one or several discrete screen(s) at different heights that evolve according to frozen flow with given wind velocity. The resulting aberrations, after corrected by deformable mirror(s) are sensed by one or multiple natural guide star (NGS) or laser guide star (LGS) Shack-Hartmann wavefront sensor(s) (WFS). The sensors can be simulated as idealized wavefront gradient sensors, best Zernike fit tilt sensors, or a physical optics WFS using user specified pixel characteristics and a matched filter [2] pixel processing algorithm. The laser guide star wavefront sensing model includes the impact of guide star elongation for a specified sodium layer profile and (optionally) an polar coordinate CCD. The tomographic wavefront reconstruction then estimates the turbulence in one or several different heights from the pseudo open loop gradients measured by the WFS, using one of several different computationally efficient implementation of a minimum variance reconstruction algorithm as described in [1]. These reconstructed turbulence screens are then fit to the actuators on one or several deformable mirrors (DMs) to achieve the best possible correction over a specified field of view (FoV).

---

[a] lianqiw@tmt.org

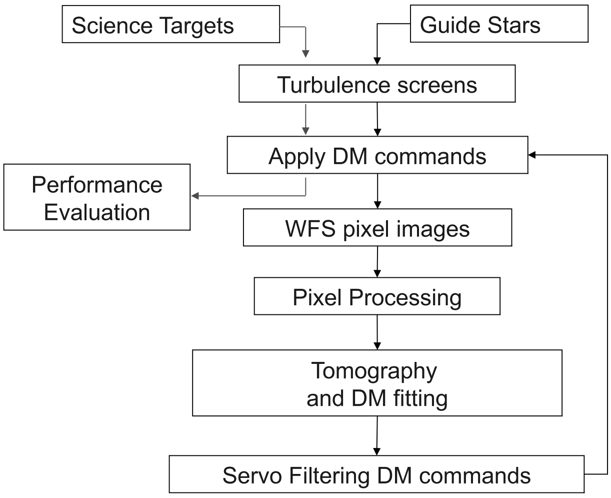

Fig. 1. Diagram of AO simulations in MAOS.

Performance evaluation is done in terms of RMS wavefront error, Strehl ratio, and/or point spread function (PSFs) at a few science objects in the target FoV, which might be different from the FoV used for the fitting step. A range of additional specified features are implemented, such as telescope wavefront aberrations and the “split tomography” control algorithm.

2.1 Simulation capabilities

MAOS can simulation many kind of adaptive optics systems efficiently, including conventional, multi-conjugate, ground layer, laser tomography, and multi-object AO systems. Long simulations (tens of real seconds or even minutes of real time) can be carried out in a reasonable amount of time. Table 1 summarizes the implemented features as of the date this paper was written.

2.2 Possible usages

The typical usage of this software is to assess the performance of an AO system, which can be either single conjugate, multi conjugate, laser tomographic, ground layer, or multi-object AO system. It can compute the open loop and closed loop wavefront error for multiple science targets and their average values with low order Zernike modes removed (optional). It can also compute open and closed loop time averaged point spread functions of the science targets. An incomplete list of possible uses includes

- assess the performance of an AO system under different turbulence conditions and at different zenith angles to build the error budget.
- assess the performance impact of static phase aberrations in the telescope mirrors (e.g., M1, M2, or M3).
- using the closed loop tip/tilt error time series or time-averaged across the science field of view to study the astrometric precision of the system as a function of integration time.
- store PSF time history at an array of NGS locations to do sky coverage post-processing [4].
- study the background tasks such as matched filter gain and offset updating, sodium layer focus tracking, etc by simulating 20+ s real time.

3 Implementation in Software

This software is written in the C language (revision 99), with external dependent libraries of FFTW version 3 (www.fftw.org) and blas/lapack (http://www.netlib.org/lapack/). The code contains a local

Category	Capability
AO Type	– NGS or LGS SCAO mode – NGS or LGS MCAO mode – NGS or LGS GLAO mode – LGS LTAO mode – MOAO with optional SCAO or MCAO woofer
Laser	– Center or side launch with specified LLT location, aperture diameter. – Gaussian beam cut off by the aperture.
Turbulence phase screens	– Multiple Von Karman turbulence screens with frozen flow and given r_0, L_0, C_n^2profile, wind velocity profile.
WFS	– Subaperture partial illumination due to telescope pupil function – Geometric gradients computed as subaperture-averaged gradients – Geometric gradients computed as Zernike best fit tilt (z-tilt) – Physical optics modeling of the WFS using FFT – Optional Polar Coordinate CCD for LGS WFS – Elongated LGS modeling based upon laser beam uplink through the atmosphere and sodium layer vertical profile – Matched filter pixel processing algorithm – Thresholded center of gravity
DM	– DM with linear or bi-cubic spline influence function. – Actuator command histogram computation. – Actuator command clipping – Actuator hysteresis – DM-to-WFS misregistration and pupil distortion
Tomography	– Integrated or split tomography (ad hoc or minimum variance). – Zonal, minimum variance pseudo-open loop reconstructor for the full system or high order only in split tomography. – Solver: Conjugate gradients, Fourier domain preconditioned conjugate gradients or Cholesky back-substitution
Servo type	– Type I control – Type II control using lead filter for low order loop.
Performance evaluation	– Multi-band PSF time history or time averaged (optional tip/tilt removal). – Multi-band Strehl. – rms wavefront error with low Zernike modes removed.
Telescope effects	– Telescope amplitude effects on the primary mirror – Telescope phase effects on the primary, secondary or any other location. – Telescope phase effects on the tilted tertiary mirror on LGS WFS.

Table 1. Capabilities of the MAOS software.

Configuration	Timing (s)
Single thread on Quad core CPU	5.8
Quad core CPU (Core i7 860 @2.8Ghz)	1.67
Dual quad core CPU (W5590)	0.9
Nvidia GTX 580 GPU	0.20
Dual Nvidia GTX 580 GPU	0.11

Table 2. MAOS timing per AO time step

copy Cholmod (http://www.cise.ufl.edu/research/sparse/SuiteSparse/) software for Cholesky matrix factorization. An optimized blas library such ATLAS, GOTO-BLAS, or Intel MKL is necessary to get good performance for Cholesky decompositions of the tomography and/or fitting matrix..

A C99 compliant compiler is required to compile the code. The code has been successfully compiled on 64 bit GNU/Linux using GNU GCC, the Intel C++ compiler ICC (in C mode), or the LLVM clang, and in MAC OS X 10.5 using the GCC compiler. For machines with Intel CPUs, the code generally runs faster when compiled with ICC.

This software also contains a few optional executables (`drawdaemon, monitor, drawdaemon`) for plotting, job monitoring (jobs can be monitored on several different machines), and visualization. These executables require the GTK+ library (version 2 or 3).

The multi-threading is achieved using thread pools with Posix Threads (no MPI is implemented). This parallelism works fairly well in multi-core, shared memory machines. Table 2 shows the timing per AO time step with the TMT Narrow Field Infrared AO system (NFIRAOS) [3] in split tomography mode, which is composed of 7 atmospheric screens each sampled at 1/64 meter, 6 order 60×60 LGS WFS and 3 low order tip/tilt(/focus) NGS WFS, 9 science directions, and wavefront reconstruction using conjugate gradients (CG) with 30 iterations. The code is compiled using Intel C++ compiler. The memory consumption in the NFIRAOS baseline case with physical optics WFS is about 2 G. Table 2 summarizes the wall-clock timing per AO time step using 1, 4 or 8 CPU cores. The scaling with number of cores is consequently fairly good.

4 Running on GPU

The state-of-the-art Nvidia GTX 580 GPU has a total of 512 CUDA cores (or stream processors, equivalent to simplified/dedicated CPU cores) arranged in 16 multiprocessors. It is capable of 1581 GFlops of single precision floating point calculations and 192 GB/s device memory bandwidth. These performance numbers are at least ten times of the state-of-the-art quad core CPUs (i.e., Intel Core i7 2600K). In reality, an average speed up of 10 times is generally observed by porting CPU codes to run on GPUs. Therefore, it is very appealing to run simulations on GPU to improve the simulation speed.

We ported the performance critical portion of MAOS to run on GPUs using the Nvidia CUDA technology. In particular, the following sections were ported

1. Wavefront sensing: including ray tracing from the atmosphere and deformable mirrors to the pupil grid, multiple FFTs per subaperture to get images sampled on detectors, photon and read out noise, gradients calculation from subaperture images, etc.
2. Performance evaluation: including RMS wavefront error computation and point spread function calculation.
3. The wavefront reconstruction: including tomography and deformable mirror fitting for the minimum variance reconstructor.

Significant improvements in speed have been achieved, without losing accuracy due to using single precision floating numbers. Table 3 shows the timing comparison of a time step of a TMT NFIRAOS end-to-end simulation as described in 3. With two GTX 580 GPU and a quad core cpu, the timing per AO time step is reduced further to 0.11 seconds.

Although the performance improvement of ≥10 times is significant, we are still far away from the theoretical limits of computing or memory throughput. The main limiting factors are

Timing (seconds)	CPU (4 threads)	GPU (GTX 580)
Geometric WFS	0.36	0.03
Physical Optics WFS	1.36	0.13
Science RMS WFE	0.47	0.05
Reconstruction (CG30)	0.31	0.0245
Total (Phy. WFS)	2.02	0.20

Table 3. Timing on single GTX 580 GPU plus 2.8 GHz Intel Core i7 quad-core CPU versus using the CPU alone. The total is less than the sum of components because these steps are running in parallel for better resource utilization.

1. Device memory latency: Each memory operation requires 600 cycles or about 0.3 μs .
2. Kernel launch overhead: Each task is accomplished with a kernel launch which takes 2.3 μs for asynchronous launch and 6.5μs for each synchronization.
3. GPU to CPU interface bandwidth and latency: The current PCI-E ×16 interface connecting the graphics card with the mother board has only 8 GB/s throughput and 10μs latency. This single factor makes it impossible to use multiple GPUs for tomography using the conjugate gradient algorithm because the synchronization between GPUs slows down the process.

5 Conclusions

We have described the MAOS software architecture, applications and performance. We reported a 10 fold speed up obtained by using GPU computing. Limitations on GPU computing are described that limits the current performance, especially for wavefront reconstruction. MAOS has become an efficient, easy to use, general purpose adaptive optics simulator for adaptive optics systems, particularly advanced, very high order systems for TMT and other ELTs.

Acknowledgments

This work is supported by the TMT project. The authors gratefully acknowledges the support of the TMT partner institutions. They are the Association of Canadian Universities for Research in Astronomy (ACURA), the California Institute of Technology and the University of California. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the National Research Council of Canada, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Universities for Research in Astronomy (AURA) and the U.S. National Science Foundation.

References

1. Brent L. Ellerbroek. Efficient computation of minimum-variance wave-front reconstructors with sparse matrix techniques. *J. Opt. Soc. Am. A*, 19(9):1803–1816, 2002.
2. Luc Gilles and Brent Ellerbroek. Shack-hartmann wavefront sensing with elongated sodium laser beacons: centroiding versus matched filtering. *Appl. Opt.*, 45(25):6568–6576, 2006.
3. G. Herriot, D. Andersen, J. Atwood, C. Boyer, P. Byrnes, R. Conan, B. Ellerbroek, L. Gilles, P. Hickson, K. Jackson, O. Lardière, J.-P. Véran, and L. Wang. NFIRAOS - first light adaptive optics system for TMT. In *Adaptative Optics for Extremely Large Telescopes*, 2010.
4. Lianqi Wang, Brent Ellerbroek, and Jean Pierre Veran. High fidelity sky coverage analysis via time domain adaptive optics simulations. *Appl. Opt.*, 48(27):5076–5087, Sep 2009.