



F. Ferreira, A. Sevin, J. Bernard & D. Gratadour

COMPASS & KRAKEN: GPU-based AO simulation framework for RTC prototyping

Real-Time Control for Adaptive Optics
5th edition

CONTENT

- COMPASS status update: what's new?
- KRAKEN: a RTC system management
- RTC validation & timing
- Half precision for AO RTC

ELT: A NUMERICAL CHALLENGE

For the RTC

- Need to run between 500 Hz and 1 kHz
- High bandwidth
- High throughput

Ex. : ELT MAORY → 1 TMAC/s

For end-to-end simulations

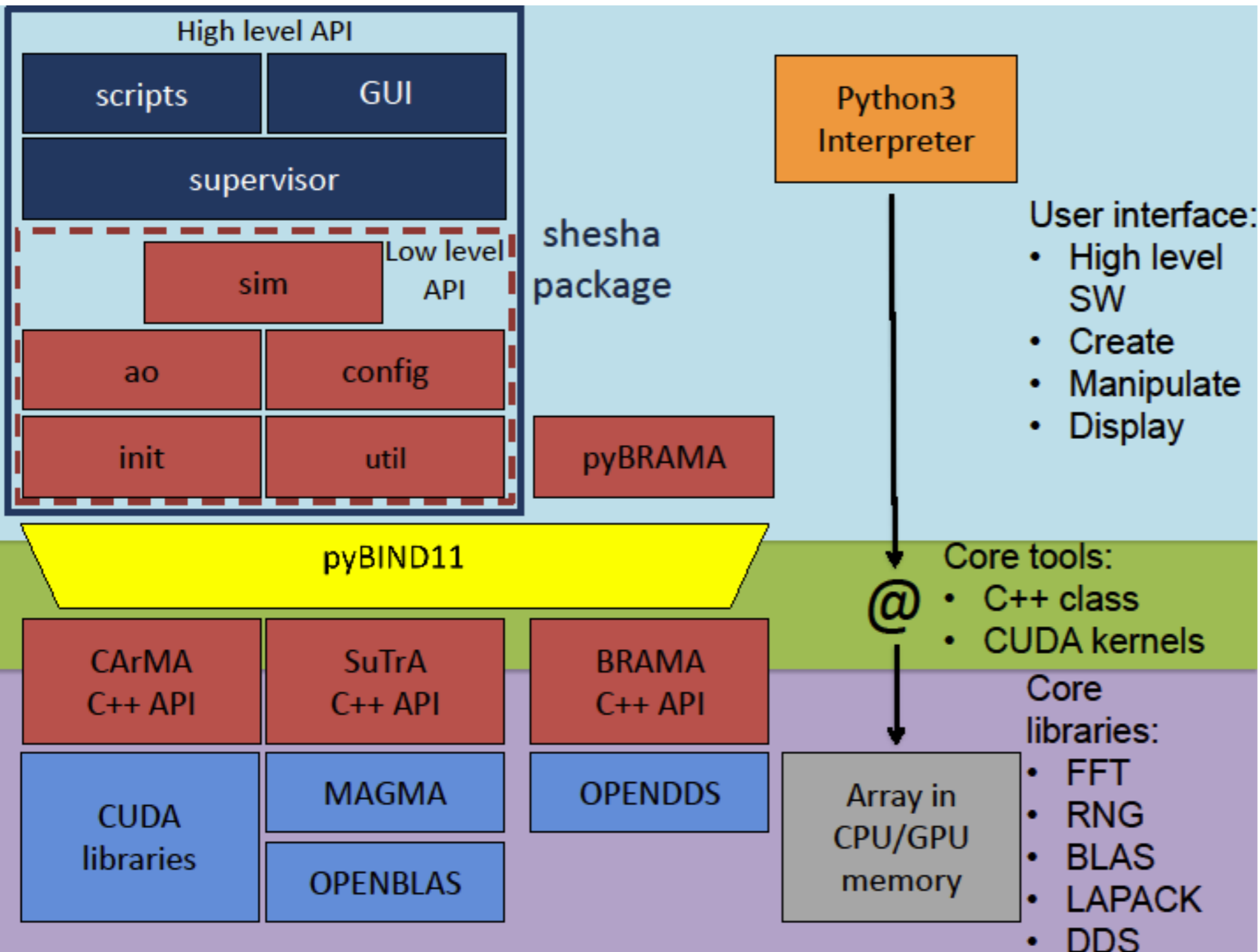
- Instruments design → Requires numerical simulation tool
- Typical simulation run time about hours at ELT scale...
- ... to multiply by the number of runs needed to explore parameters space

Main limitation: memory bandwidth

- ELT scale involves huge data flow
- Computation time mainly limited by memory bandwidth

COMPASS v3.0

WHAT'S NEW?



pyBind11

- Lighter than Cython
- Easy to implement

New high level API

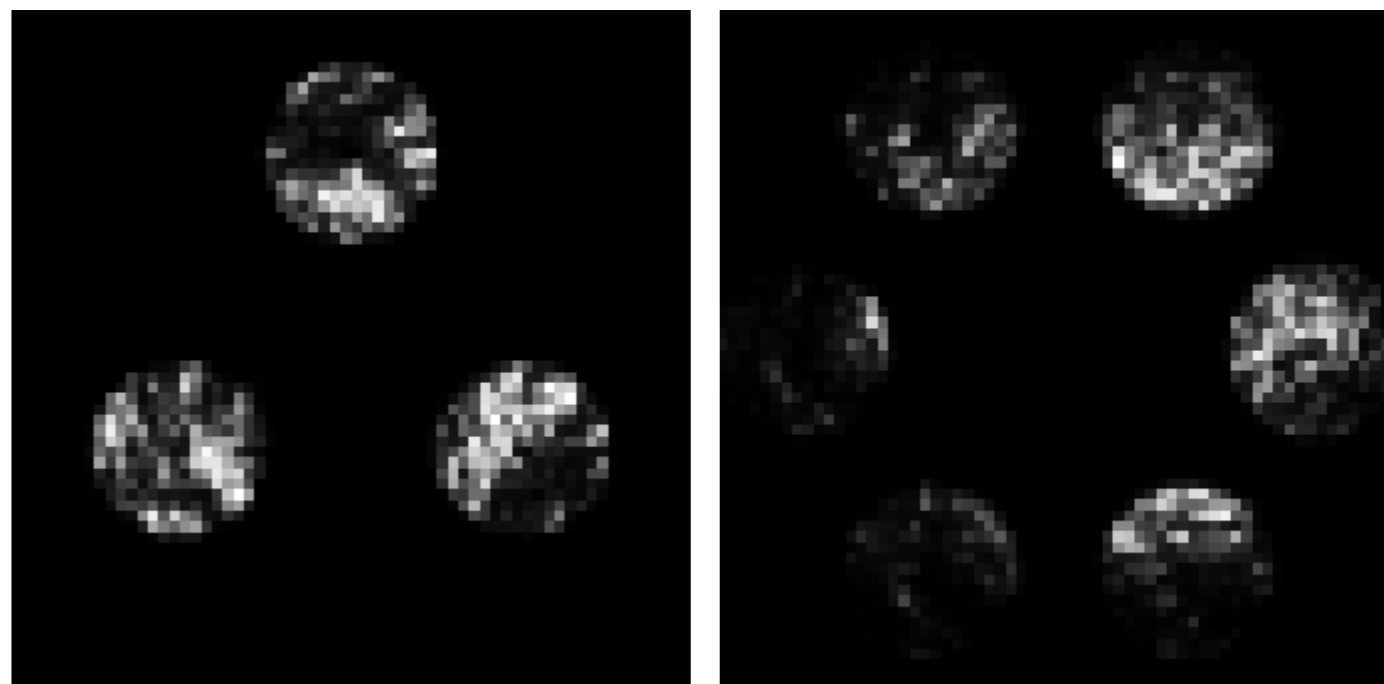
- Supervisor classes
- Easy to use user interface

COMPASS v3.0

NEW FEATURES

Installation test

- *Bash script to check correct installation of COMPASS*



Pyramid WFS

- *N faces pyramid*
- *Control on pixels*

	Test name	Init	SR@100iter
0	test_pyr3_maskedpix.py	True	0.784026
1	test_pyr_base.py	True	0.698999
2	test_pyr_ELTPup.py	True	0.738572
3	test_sh_base.py	True	0.709822
4	test_sh_bpcog.py	True	0.710266
5	test_sh_control_klbasis.py	True	0.716548
6	test_sh_cured.py	True	0.0624456
7	test_sh_ELTPup.py	True	0.666646
8	test_sh_generic.py	True	0.181335
9	test_sh_geo.py	True	0.840501
10	test_sh_influBessel.py	True	0.583418
11	test_sh_influBlacknutt.py	True	0.714818
12	test_sh_influGaussian.py	True	0.719278
13	test_sh_influRadialSchwartz.py	True	0.710639
14	test_sh_influSquareSchwartz.py	True	0.704471
15	test_sh_kl.py	True	0.619334
16	test_sh_lgs_corr.py	True	0.354425
17	test_sh_lgs_wcog.py	True	0.682489
18	test_sh_modopti.py	True	0.28718
19	test_sh_mv.py	True	0.734366
20	test_sh_tcog.py	True	0.711656

COMPASS v3.0

GUARDIANS STACK SOFTWARE

GUARDIANS

ROKET

*Error breakdown
estimation tool*

GROOT

Error breakdown modeling

GAMORA

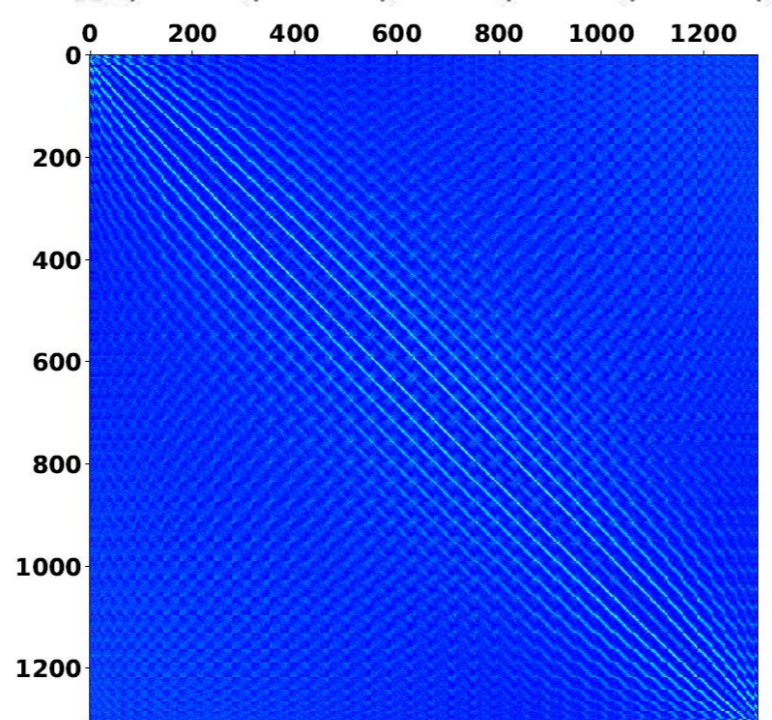
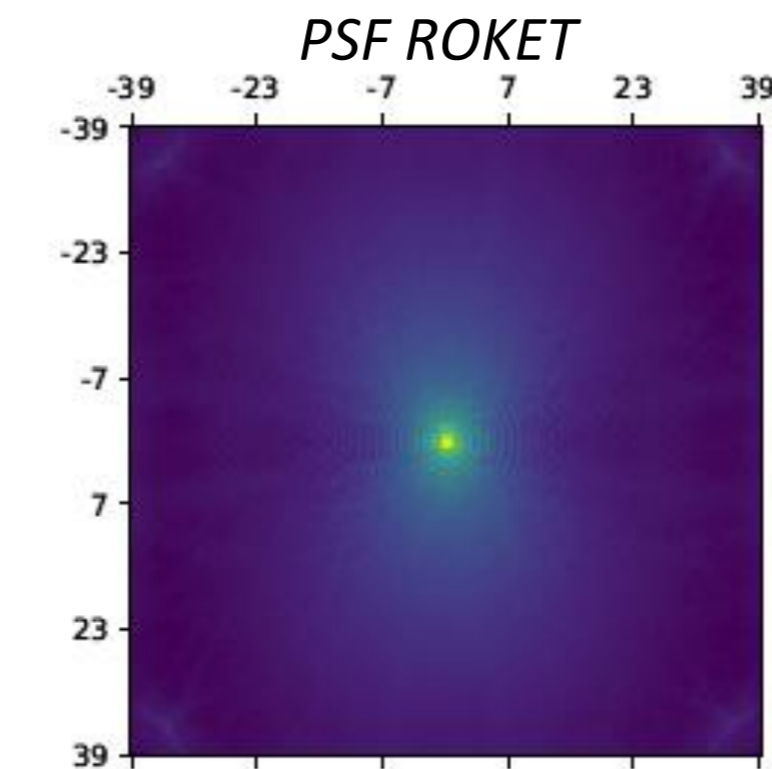
PSF reconstruction

DRAX

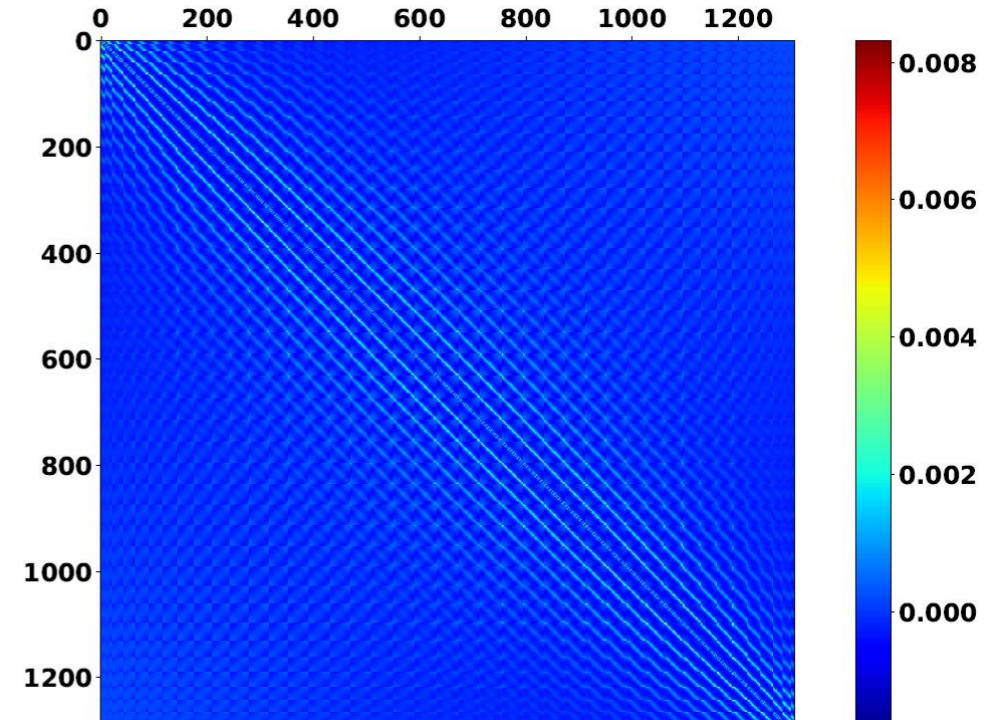
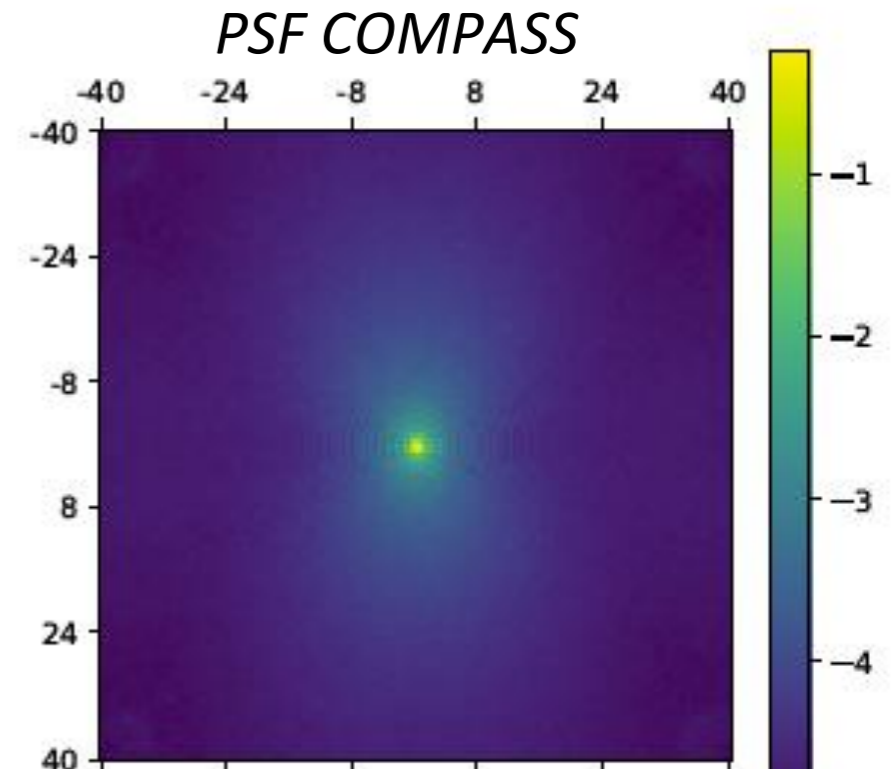
ROKET exploitation tool

STARLORD

*Structure functions
computation*



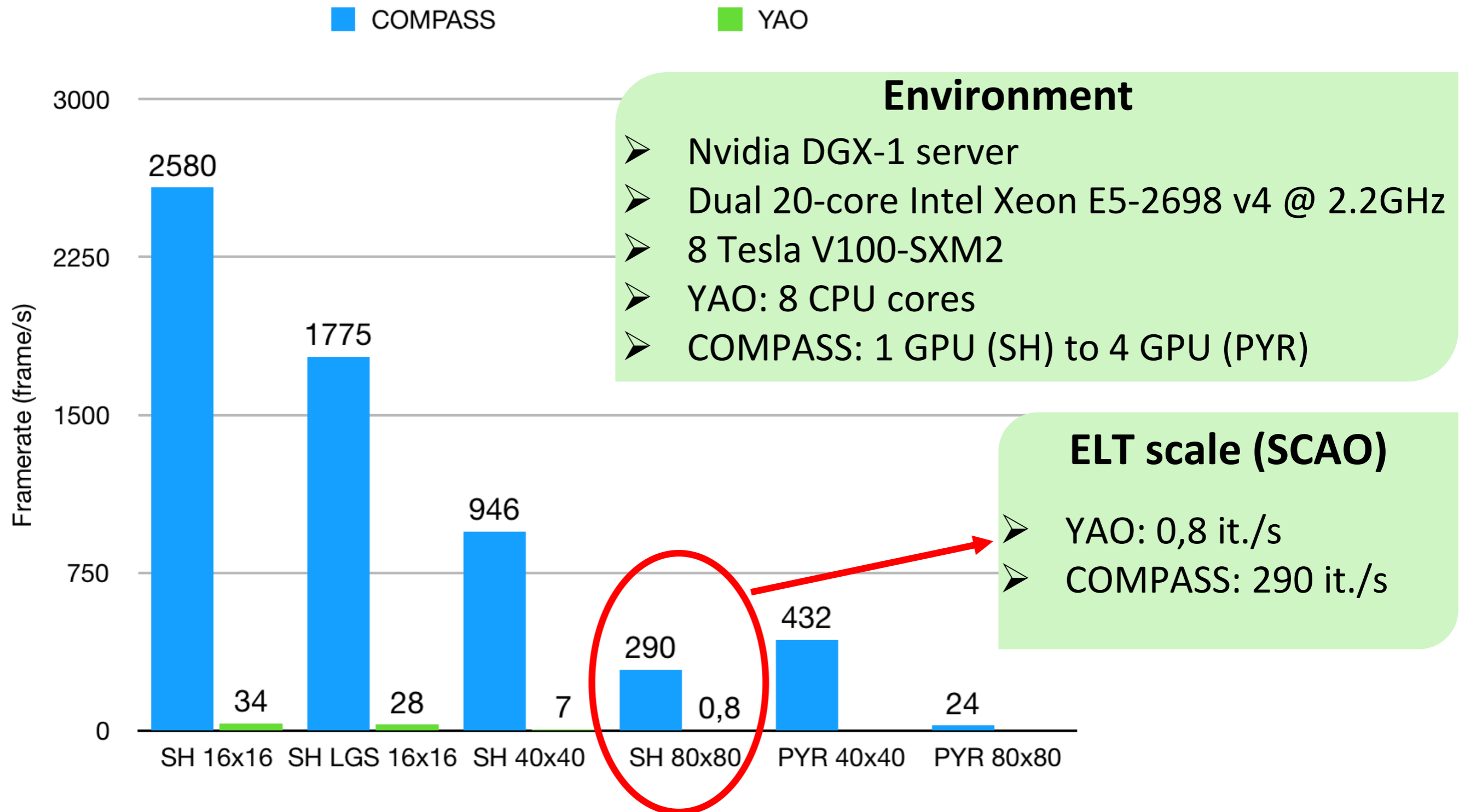
Cov. Mat. GROOT



Cov. Mat. COMPASS

COMPASS PERFORMANCE

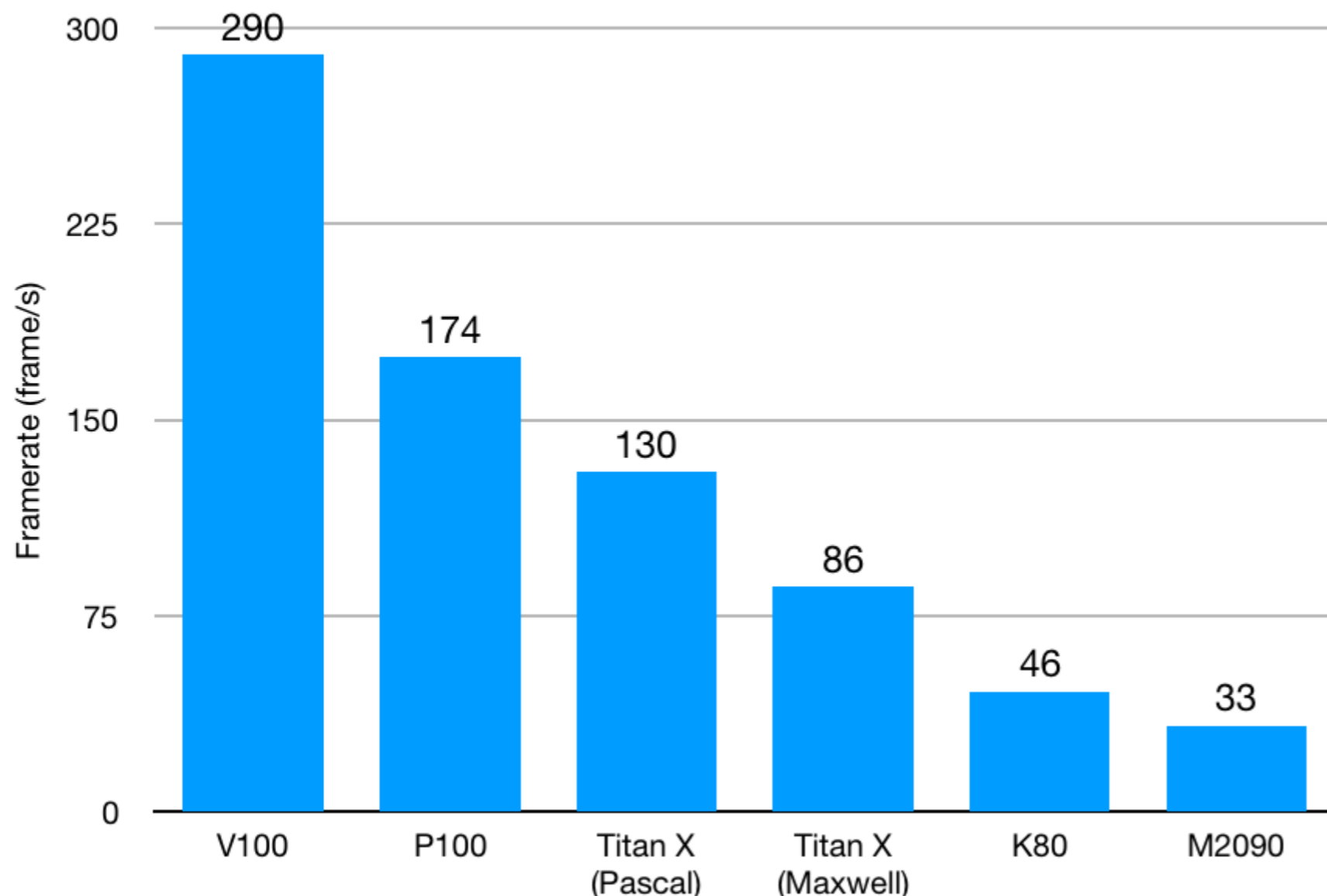
TIME TO SOLUTION



COMPASS PERFORMANCE

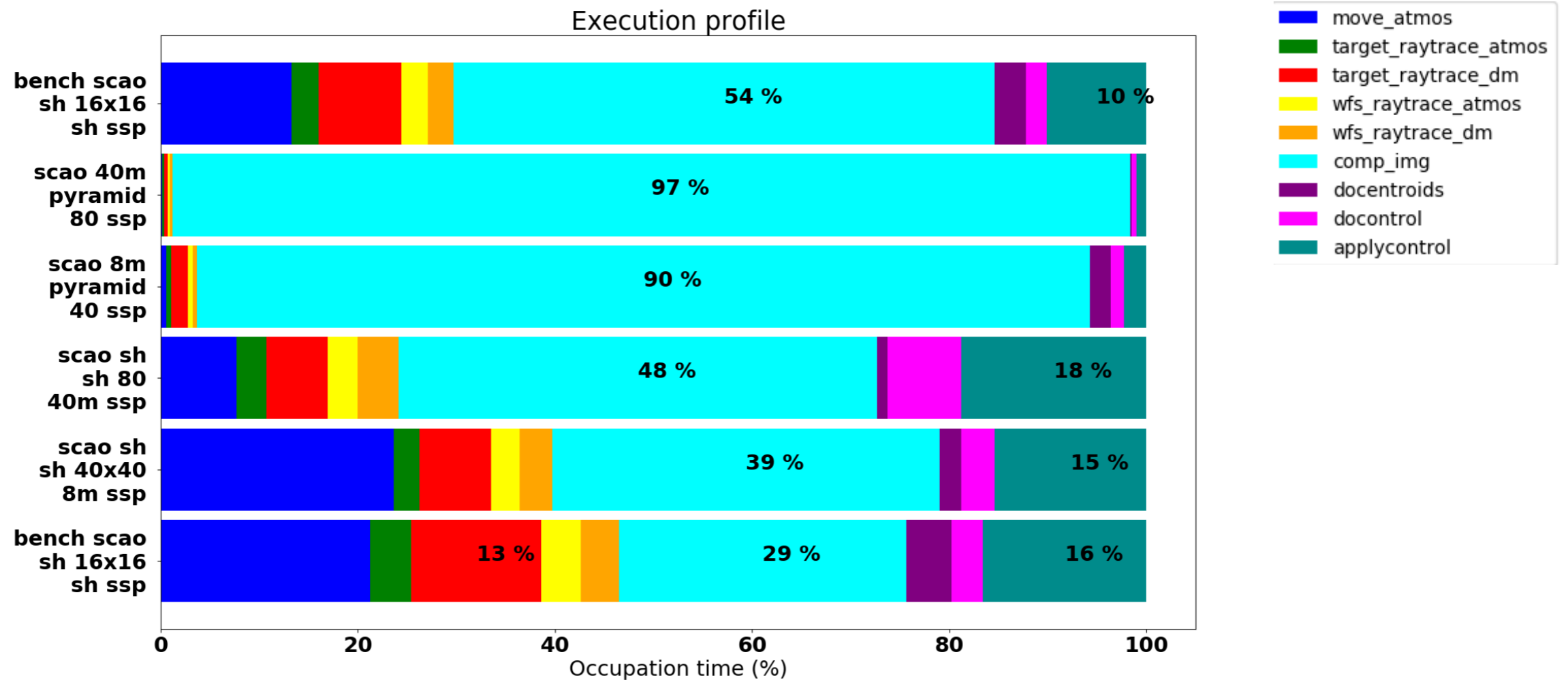
SCALABILITY

	V100	P100	Titan X	Titan X	K80	M2090
Arch.	Volta	Pascal	Pascal	Maxwell	Kepler	Fermi
CUDA cores	5,120	3,584	3,584	3,072	2,496	512
Mem. Band. (GB/s)	760	515	351	255	96	75



- Memory bound
- Architecture independence optimization
- COMPASS performance has been multiplied by a factor 5 within 6 years
- Will continue to follow GPU board performance trends

COMPASS PERFORMANCE PROFILE



- Dominated by WFS image computation
- RTC module represents less than 10 % of the profile

COMPASS & RTC

COMPASS modularity

- User interface & core libraries implemented as independent modules:

Atmosphere
Target
WFS
DM
RTC

Bench & RTC extensions

- COMPASS RTC module will be used to control optical bench
- It could also be used as WFS images & DM simulators to validate RTC

KRAKEN

RTC validation with COMPASS

- Need to develop a common interface RTC – COMPASS
- Need to set an unified environment for RTC developments & validation

KRAKEN

- RTC validation environment
- RTC configuration
- Process management: system health & scheduling
- Communication RTC \leftrightarrow simulation based on standard interfaces:
 - GPU IPC \rightarrow SHM + boost::interprocess
 - CACAO ImageStreamIO \rightarrow SHM + semaphores
 - FPGA \rightarrow GPU direct + memory polling

KRAKEN ARCHITECTURE

Kraken Main

- Allocates arrays on the shared memory

OCTOPUS

Shared memory interface (device and/or host)

Kraken RTC

Slopes
computation

Commands
computation

Kraken Simu (COMPASS)

Turbulence

WFS image

DM shape

KRAKEN

ARCHITECTURE

OCTOPUS

Shared memory

- WFS frame
- DM commands
- Command matrix
- Subap. positions

CACAO

- Semaphores based
- Part of the CACAO framework
- Up to 10 subscribers
- CPU scheduling
- CPU or GPU alloc.

GPU IPC

- Boost::interprocess
- Based on condition variables
- Unlimited subscribers
- CPU scheduling
- GPU alloc. only

KRAKEN

ARCHITECTURE

Kraken simulation

COMPASS supervisorRTC

Initialisation

- Atmosphere
- Target
- WFS
- DM
- Command matrix

SHM interfaces

- fakeWFS
- fakeDM
- Command matrix
- Valid positions

AO loop

Move atmos.

Raytrace

WFS image

fakeWFS.send

fakeDM.recv

DM shape

KRAKEN

ARCHITECTURE

Class KrakenRTC

Attributes

- Rtc
- fakeWFS
- fakeDMS

Methods

- initInterfaces()
- loop()

Abstract methods

- initRTC()
- next()

Subclasses

COMPASS

*COMPASS RTC
module*

Moray

*Python
implementation*

MorayCpp

*C++
implementation*

Orca

*Perp. Kernels
Custom MVM*

KRAKEN IN ACTION

The screenshot shows a terminal window with two panes. The left pane displays the Kraken application's configuration menu, and the right pane shows the terminal output of the application's execution.

Kraken Configuration Menu:

- < Config file: KrakenRTC_sh_16x16_8pix.py
- < Config RTC: Compass
- (X) CPUSHM
- () GPUSHM
- [X] Debug
- [] FastMode
- Iterations: -1
- < Startup
- < Benchmark
- < Open Kraken TMUX
- < Build Kraken
- < Open Kraken Build TMUX
- < Start RTD
- < Cleanup
- < Exit

Terminal Output:

```
malloc semptr 10 entries
image->md[0].sem = 10
shared memory space in CPU RAM = 248 bytes
Creating 10 semaphores
malloc semptr 10 entries
image->md[0].sem = 10
shared memory space in CPU RAM = 248 bytes
Creating 10 semaphores
malloc semptr 10 entries
image->md[0].sem = 10
Dropping into IPython, type %gui qt5 to unlock GUI
KrakenRTC_sh_16x16_8pix.py: 1 Killed
[ferreira@fftbook kraken]$

waiting startup configuration
File /tmp/compass_cmat.im.shm size: 327152
image size = 222 368
atype = 9
atype = FLOAT
1 keywords
10 semaphores detected (image->md[0].sem = 10)
File /tmp/compass_validsubs.im.shm size: 1840
image size = 2 184
atype = 6
atype = INT32
1 keywords
10 semaphores detected (image->md[0].sem = 10)
Waiting Simulation for valid pixels and command matrix
nvalid : 184
nact : 222
gain : 0.400000
controller uses 1 streams
Killed 621.9 544.2
[ferreira@fftbook kraken]$
```

13600	0.743	0.732	0.0	619.8
13700	0.718	0.732	0.0	564.7
13800	0.710	0.732	0.0	578.5
13900	0.718	0.732	0.0	592.3
14000	0.709	0.732	0.0	563.2
14100	0.723	0.732	0.0	618.0
14200	0.779	0.732	0.0	617.8
14300	0.703	0.732	0.0	607.6
14400	0.699	0.732	0.0	555.1
14500	0.725	0.731	0.0	610.1
14600	0.717	0.731	0.0	609.2
14700	0.721	0.731	0.0	591.2
14800	0.772	0.732	0.0	340.5
14900	0.735	0.732	0.0	580.4
15000	0.735	0.732	0.0	621.1
15100	0.720	0.732	0.0	590.5
15200	0.729	0.732	0.0	621.9
15300	0.742	0.732	0.0	625.1

[Killed
[ferrerira@fftbook kraken]\$
[ferrerira@fftbook kraken]\$
[Kraken] 1:MainWindow* ferrerira@fftbook: ~/w* 11:33 22-oct-1

KRAKEN

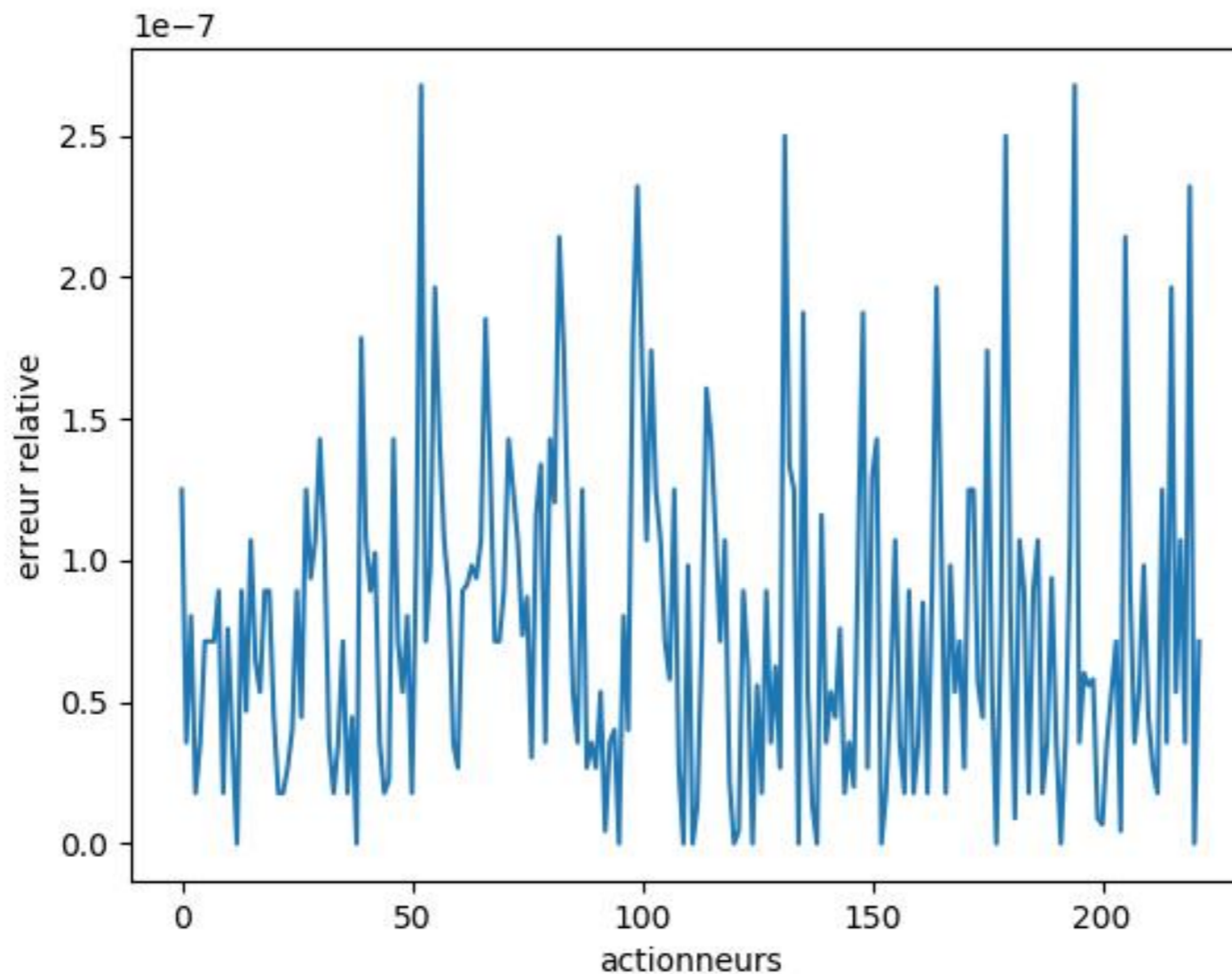
RTC VALIDATION

RTCs cross validation

- Cross validation of COMPASS RTC with Python & C++ implementations
- Classical least square + integrator

```
-----  
iter# | S.E. SR | L.E. S  
-----  
100    0.756  0.756  
200    0.766  0.761  
300    0.728  0.750  
400    0.723  0.743  
500    0.762  0.747  
600    0.748  0.747  
700    0.753  0.748  
800    0.742  0.747  
900    0.752  0.748  
1000   0.762  0.749  
loop execution time: 1.  
.0019953675270080566 (me
```

COMPASS RTC



```
-----  
framerate (Hz)  
-----  
485.2  
612.4  
650.5  
649.6  
612.2  
640.5  
647.2  
654.2  
644.6  
638.7  
7 ( 1000 iterations), 0  
59692839 Hz
```

C++ RTC

on RTC

KRAKEN

RTC BENCHMARK

KrakenTimer

- 3 ways of timing processes:
 - Python module *time*
 - cudaEvents
 - GPU clock counter

```
class carma_timer {
public:
    cudaEvent_t startEv, stopEv;
    double total_time;
    long cc;
    long long int *clockCounter;
    carma_timer() {
        cudaEventCreate(&startEv);
        cudaEventCreate(&stopEv);
        cudaGetDeviceProperties(&cdp, context-
        }get_activeDevice());
        GPUfreq = cdp.clockRate * 1000;
        long dim[1] = {1, 1};
        timeBuffer = cudaEventRecord(cudaEventRecord(
        cudaEventDestroy(stopEv);
        cudaMalloc(&clockCounter, sizeof(long long int));
    }
    void start() { cudaEventRecord(startEv, 0); }

    void reset() { total_time = 0.0; }
    void delete timeBuffer;
    /** stop timer (clock counter); time in seconds */
    void stop() {
        float gpuTime;
        cudaEventRecord(stopEv, 0);
        cudaEventSynchronize(stopEv);
        cudaEventElapsedTime(&gpuTime, startEv, stopEv);
        total_time += gpuTime;
        cudaEventRecord(cudaEventRecord(
        } clockCounter, GPUfreq);
        cc++;
    }
    /** return elapsed time in seconds */
    double elapsed() { return total_time; }
};
```

KRAKEN

RTC BENCHMARK

Benchmark conditions

- ELT SCAO scale (MICADO):
 - SH WFS: 5,070 x 9,792
 - PYR WFS: 4,300 x 23,616
- Framerate obtained by timing the RTC *next()* function over 100k iterations
- DGX-1 server: Tesla V100-SXM2

SH WFS 80x80

Interfaces	GPU IPC		CACAO	
SHM location	CPU	GPU	CPU	GPU
COMPASS	852	1,700	1,071	1,692
ORCA	NIY	NIY	NIY	1,980

PYR WFS 92x92

Interfaces	GPU IPC		CACAO	
SHM location	CPU	GPU	CPU	GPU
COMPASS	478	872	490	870
ORCA	NIY	NIY	NIY	1,070

KRAKEN

TESTING HALF PRECISION FOR AO RTC

AO RTC with FP16

- Easily tested with Moray through *numpy.float16*
- No differences seen in terms of SR over 10k iterations...
- Neither in terms of DM commands: relative error about 10^{-3} (FP16 precision)

Preliminary test with cuBLAS

- *cublasHgemm* → Matrix-matrix multiplication in half precision
- No Hgemv implemented yet → Use Hgemm with one matrix dimension = 1
- MICADO MVM dimensions with PWFS → 5,000 by 24,000
- Test on DGX-1 server with Tesla V100-SXM2

Precision	Execution time [μ s]
FP32	658
FP16	745
FP16 + tensor cores enabled	362

- Of course, Hgemm not optimized for MVM
- Still, 1.8 speed up using tensor cores

CONCLUSION

COMPASS

- ELT scale efficient GPU-based AO E2E simulator
- Performance scales with GPU memory bandwidth
- Modularity allows alternative use as pseudo-RTC or frame simulator
- FP16 RTC module implementation

KRAKEN

- An ecosystem to manage, configure and validate RTC
- Based on standard SHM interfaces
- Ongoing work:
 - Detailed benchmark
 - Frames loss with GPU IPC

THANK YOU !

GitHub : <https://github.com/ANR-COMPASS/shesha>

