

A performance, energy and accuracy aware benchmarking methodology for robot vision

- SLAMBench -

Luigi Nardi Ph.D.
Imperial College London
@Google, March 24th 2015

In collaboration with:

Bruno Bodin, M Zeeshan Zia, John Mawer, Andy Nisbet, Paul H J Kelly, Andrew J Davison, Mikel Luján, Michael F P O'Boyle, Graham Riley, Nigel Topham, Steve Furber



Imperial College
London



My background

- where my biases come from -

- Education:

- 2002: **B.Sc. Computer engineering**

La Sapienza university, Rome

- 2007: **M.Sc. Architecture and distributed systems**

La Sapienza Rome, Universidad Autonoma Madrid and LAAS-CNRS lab Toulouse

- 2011: **Ph.D. Computer/computational science**

Formalisation and automation of YAO, code generation for variational data assimilation, CEDRIC and LOCEAN, Paris

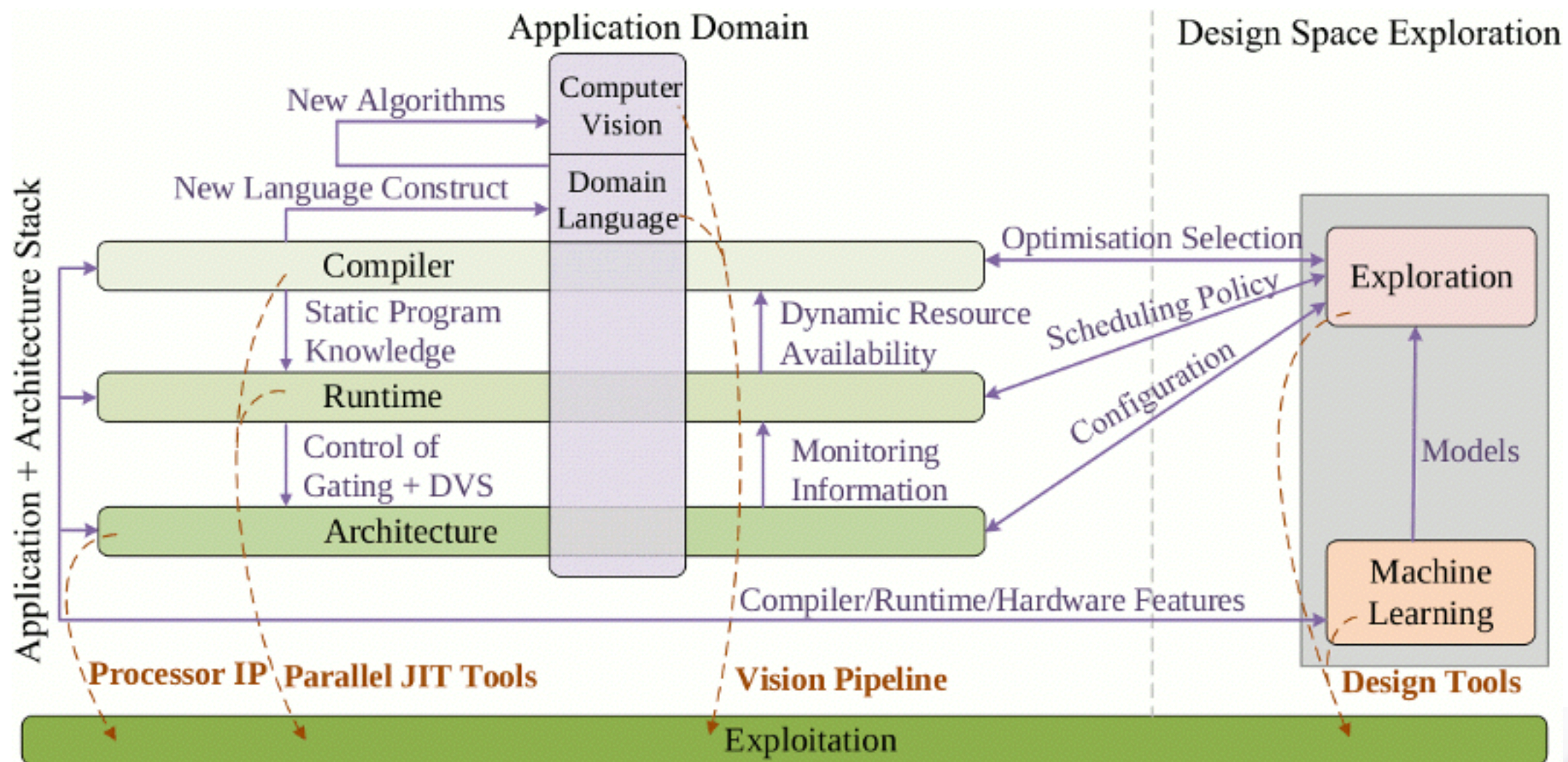
- Research:

- 2011-2013: **Permanent researcher**, R&D at Murex S.A.S., Paris
- From 2014: **Research associate**, Imperial College London

PAMELA project

Panoramic Approach to the Many-core LAndscape -
from application to end-device: a holistic approach

5-year EPSRC grant: Imperial, Manchester and Edinburgh

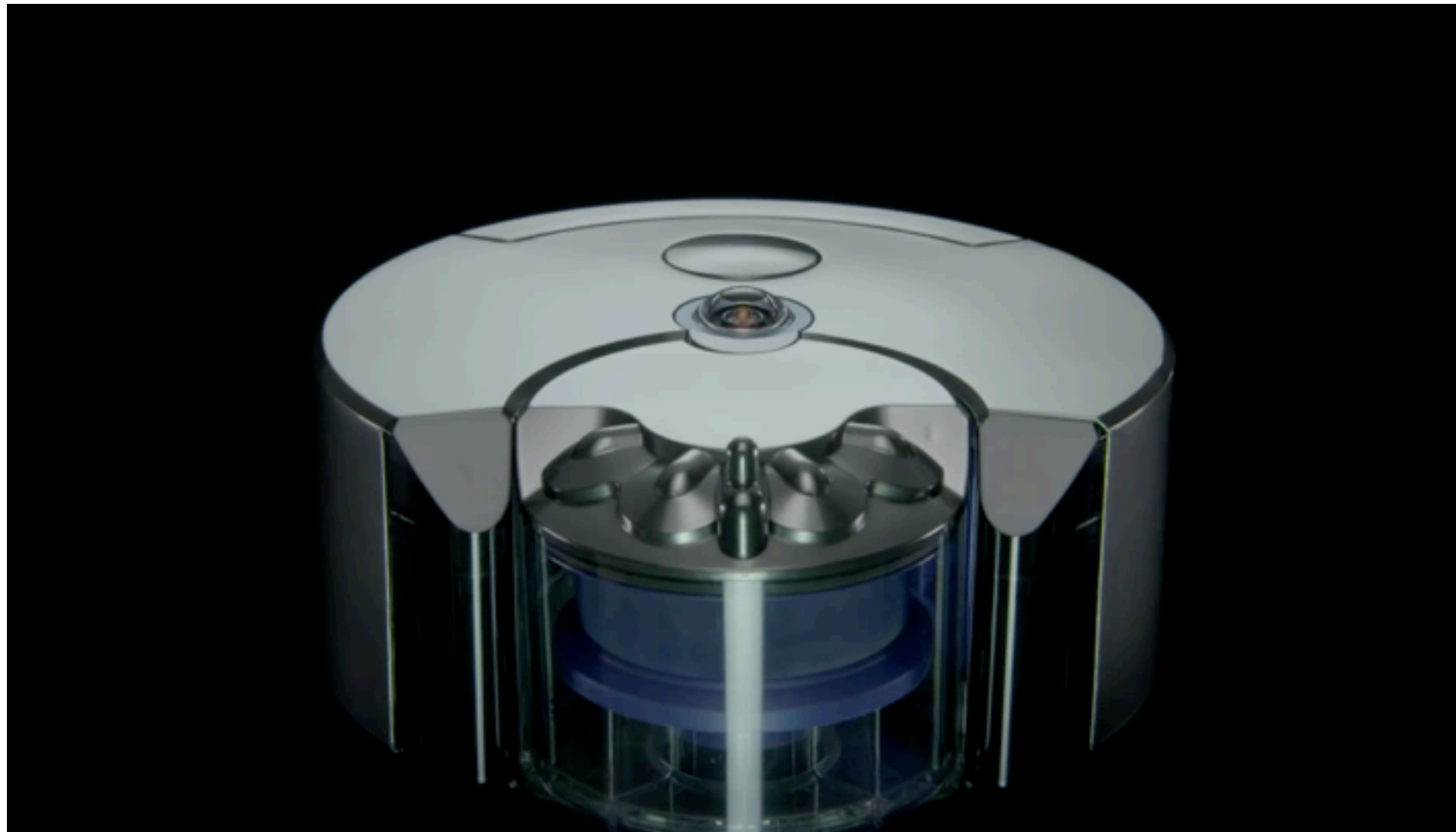
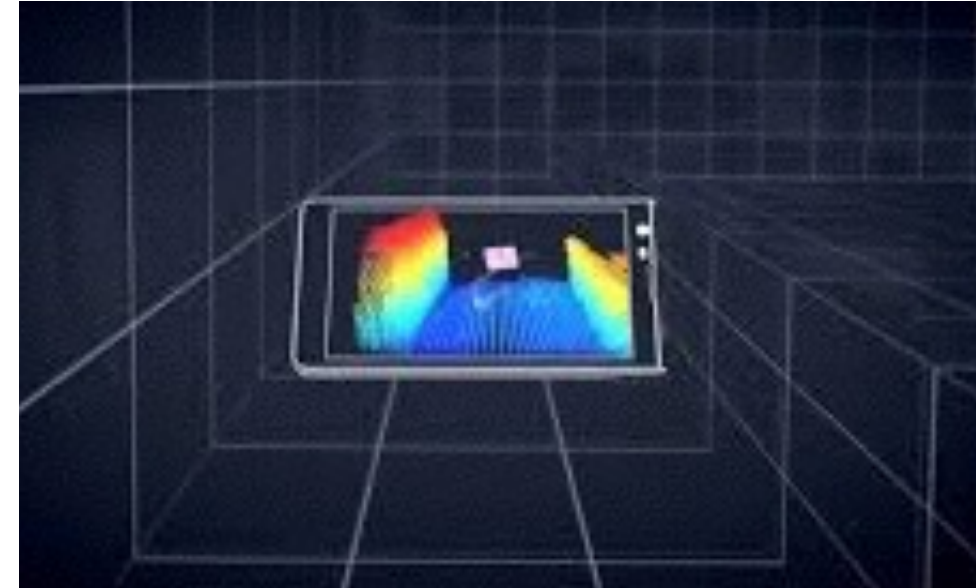


Simultaneous localisation and mapping (SLAM)

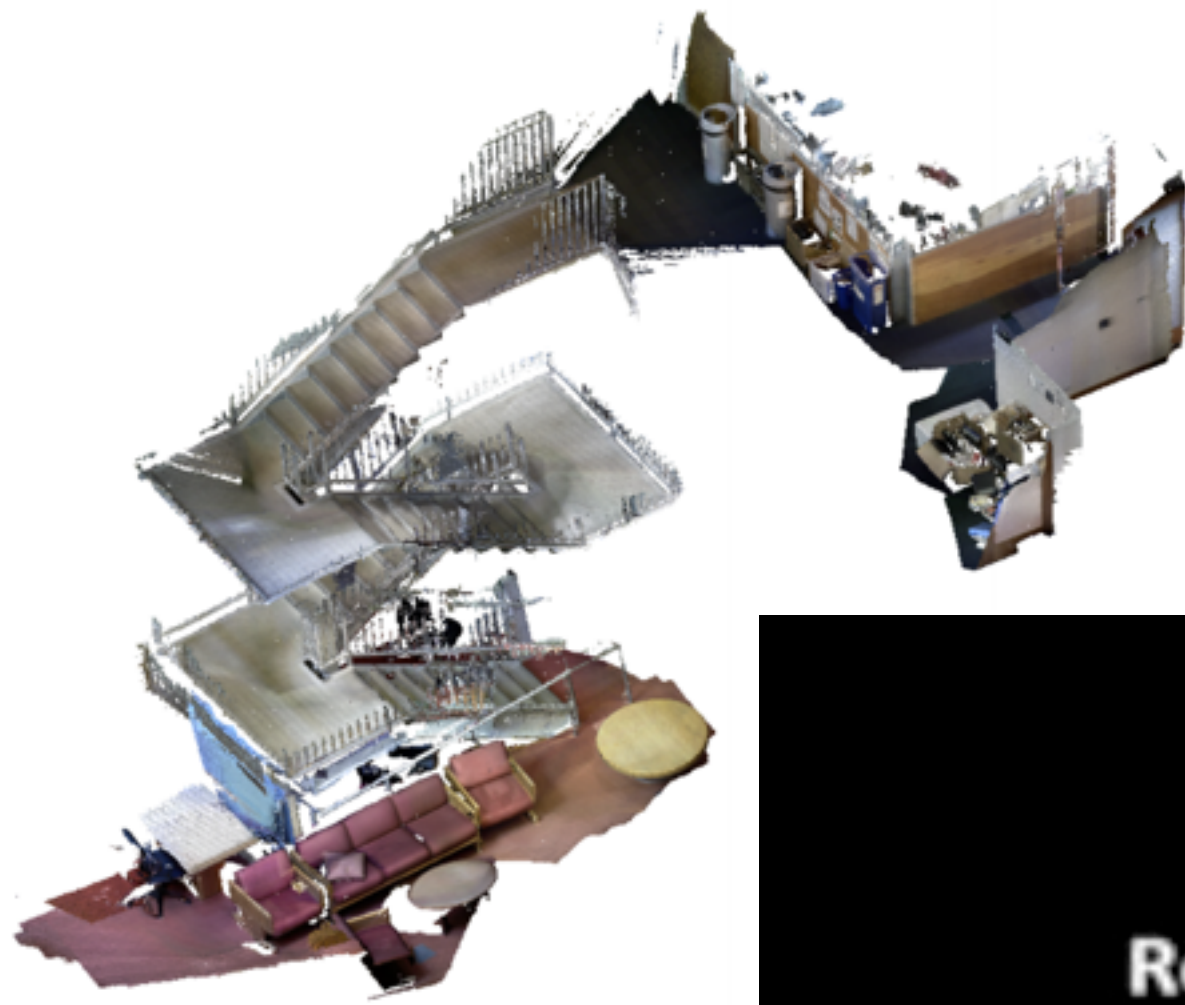
Build a coherent world representation and localise the camera in real-time



Video:
[Dyson 360 Eye](#)



Simultaneous localisation and mapping (SLAM)



[Whelan et al. 2012]



SIGGRAPH Talks 2011 KinectFusion: Real-Time Dynamic 3D Surface Reconstruction and Interaction

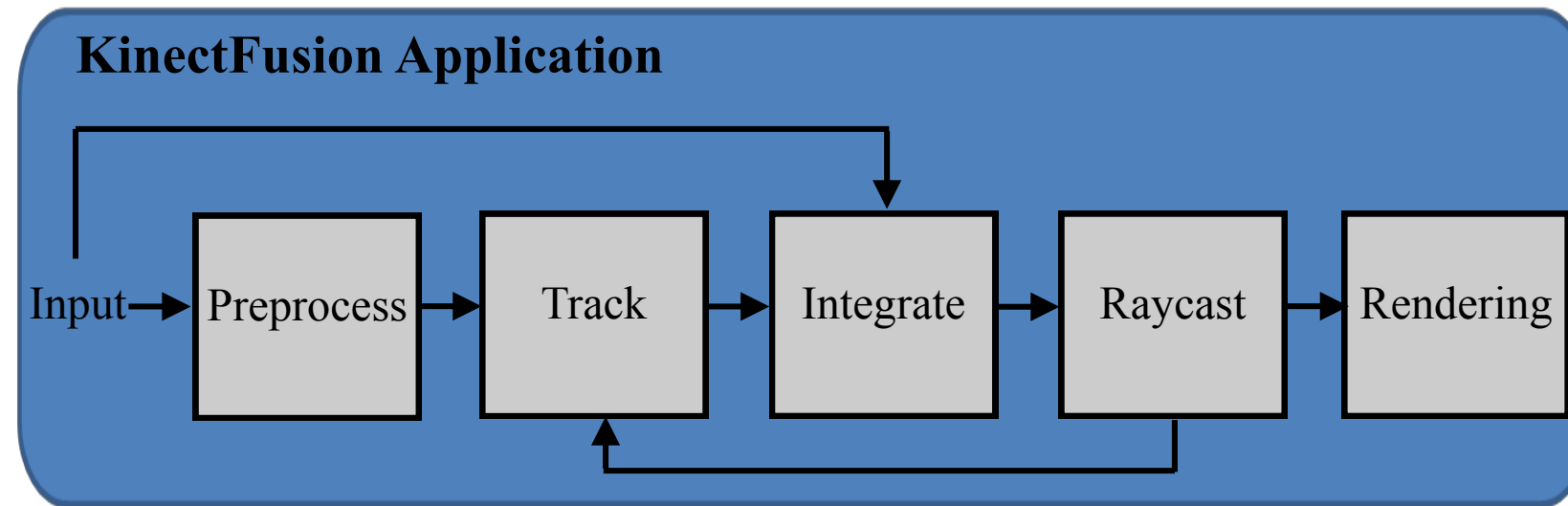
Shahram Izadi 1, Richard Newcombe 2, David Kim 1,3, Otmar Hilliges 1,
David Molyneaux 1,4, Pushmeet Kohli 1, Jamie Shotton 1,
Steve Hodges 1, Dustin Freeman 5, Andrew Davison 2, Andrew Fitzgibbon 1

1 Microsoft Research Cambridge 2 Imperial College London
3 Newcastle University 4 Lancaster University
5 University of Toronto

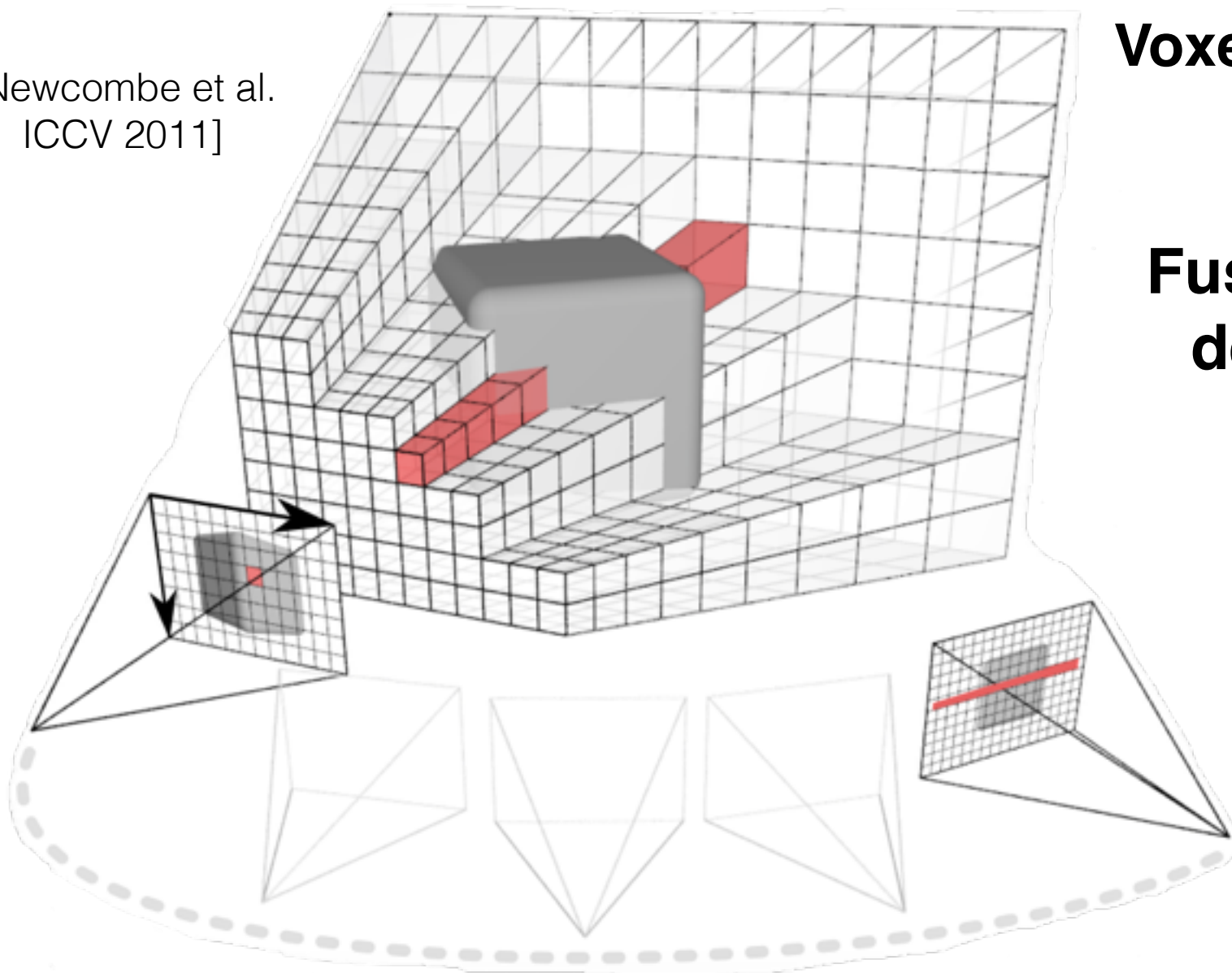
Video:

[Newcombe et al. ISMAR 2011]

KinectFusion SLAM implementation



[Newcombe et al.
ICCV 2011]



Voxel grid represents **3D surfaces**
[Curless and Levoy 1996]

Fuses (or integrate) the stream
depth frames into a **3D map**

3D surfaces recovered
by **raycasting**

Localisation (or tracking):
estimates the camera pose



Outline

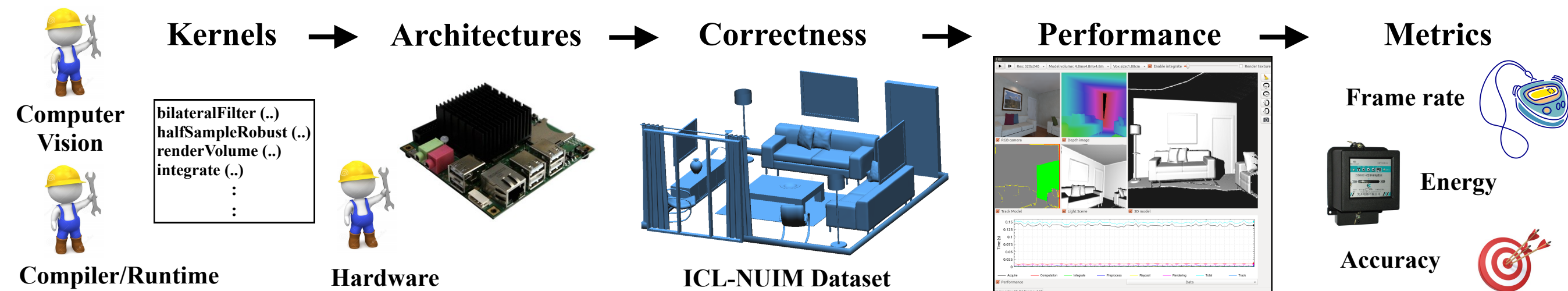
- Simultaneous localisation and mapping (SLAM) application
- **Performance metrics in SLAM**
- From desktop to embedded
- SLAMBench opportunities



Three “Performance” metrics

1. In computer vision (CV) benchmarks target **accuracy**
2. Benchmarks in other fields are about **execution time**
3. Is somebody targeting **energy**?

Holistic approach to SLAM “performance”: SLAMBench



How to measure SLAM “Performance”?

SLAM computation depends on:

- Images acquired
- Way the camera is moved
- Numerical approximations
- Processing frame rate
(depends on hardware capability)



How to measure SLAM “Performance”?

SLAM computation depends on:

- Images acquired
- Way the camera is moved
- Numerical approximations
- Processing frame rate
(depends on hardware capability)

**Need for reproducibility
and accuracy check**

Pre-recorded scenes

Process-every-frame mode

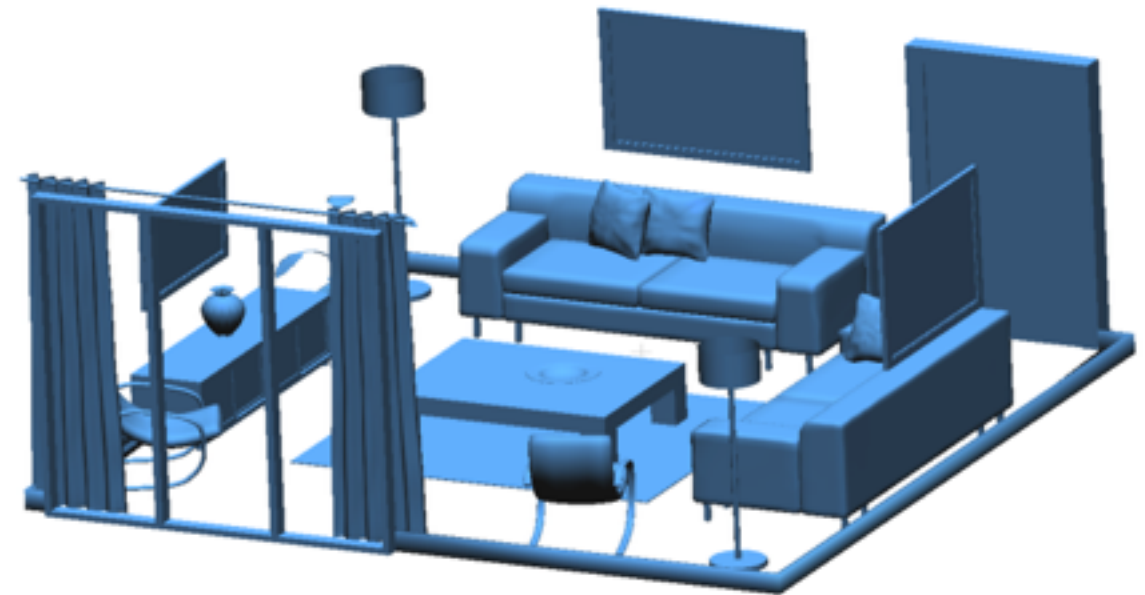
*ICL-NUIM dataset
(frames and ground truth)*



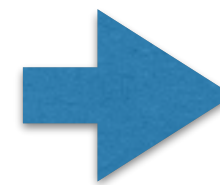
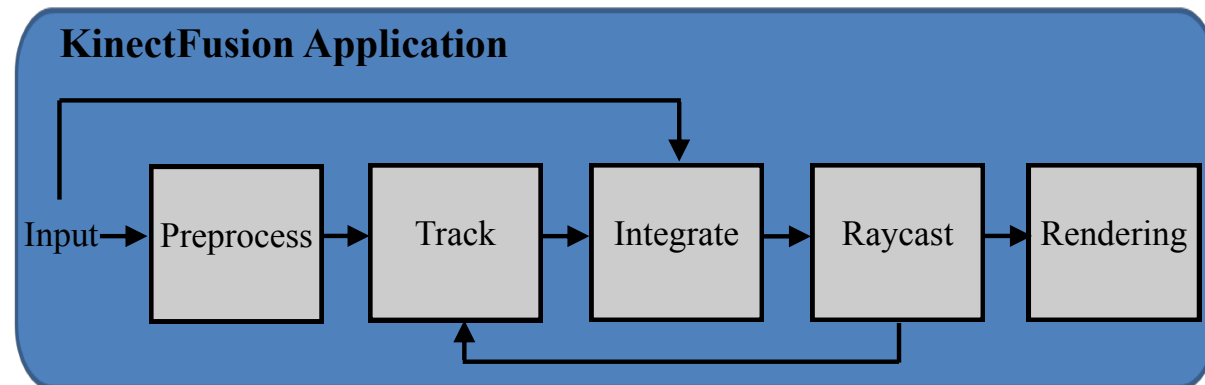
ICL-NUIM dataset



- ICL-NUIM synthetic dataset [Handa et al. 2014]
- 880 RGB-D frames at 30 FPS
- Absolute trajectory error (ATE) based on ground truth

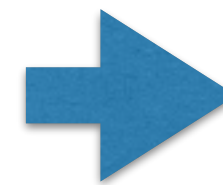
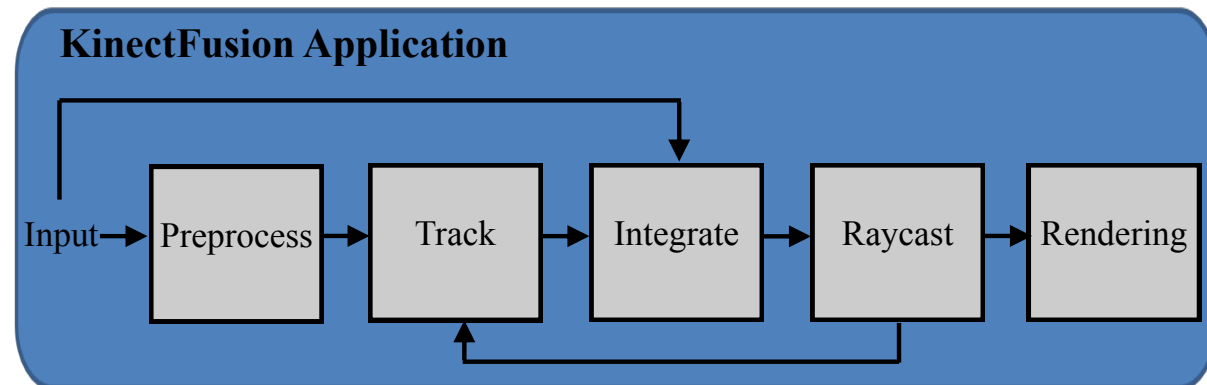


SLAMBench framework

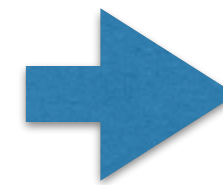
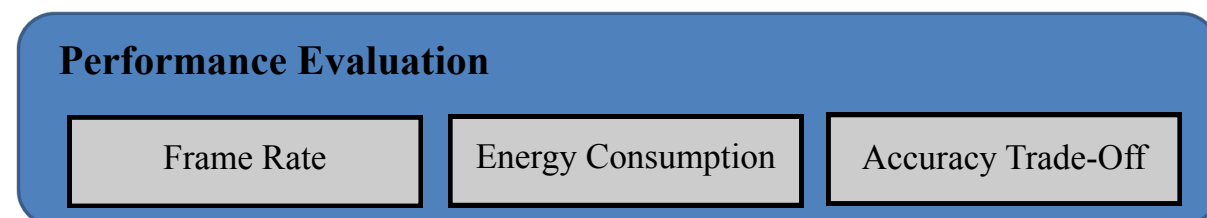


high-level blocks composed
by 14 lower level kernels
based on [Reitmayr 2011]

SLAMBench framework

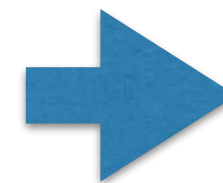
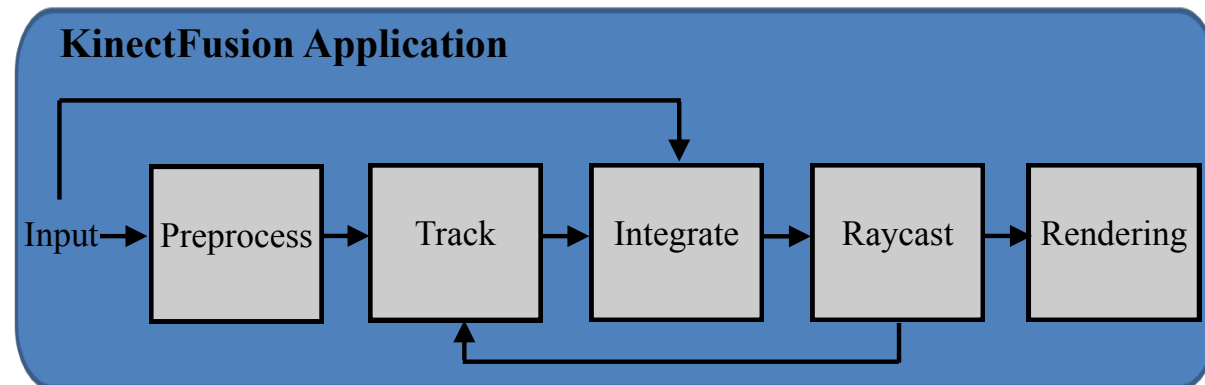


high-level blocks composed
by 14 lower level kernels
based on [Reitmayr 2011]

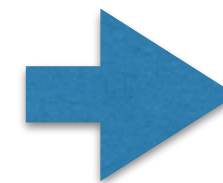
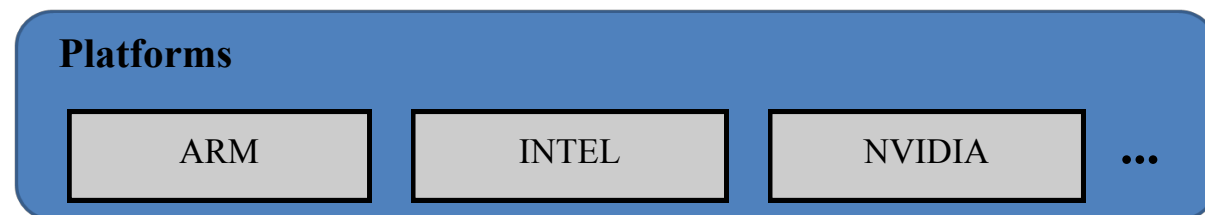


3 metrics:
execution time/energy/accuracy

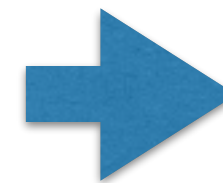
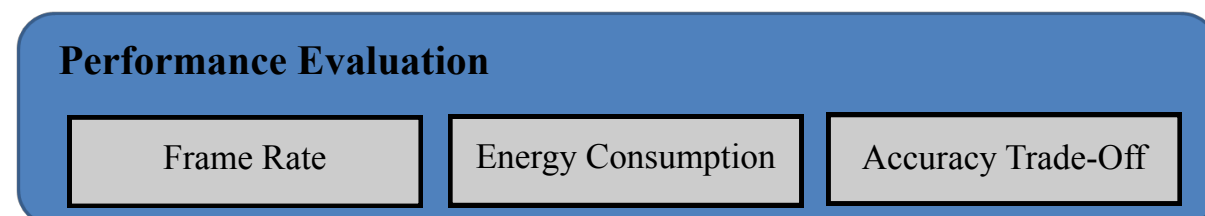
SLAMBench framework



high-level blocks composed
by 14 lower level kernels
based on [Reitmayr 2011]

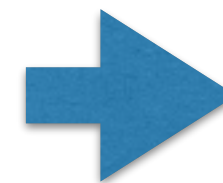
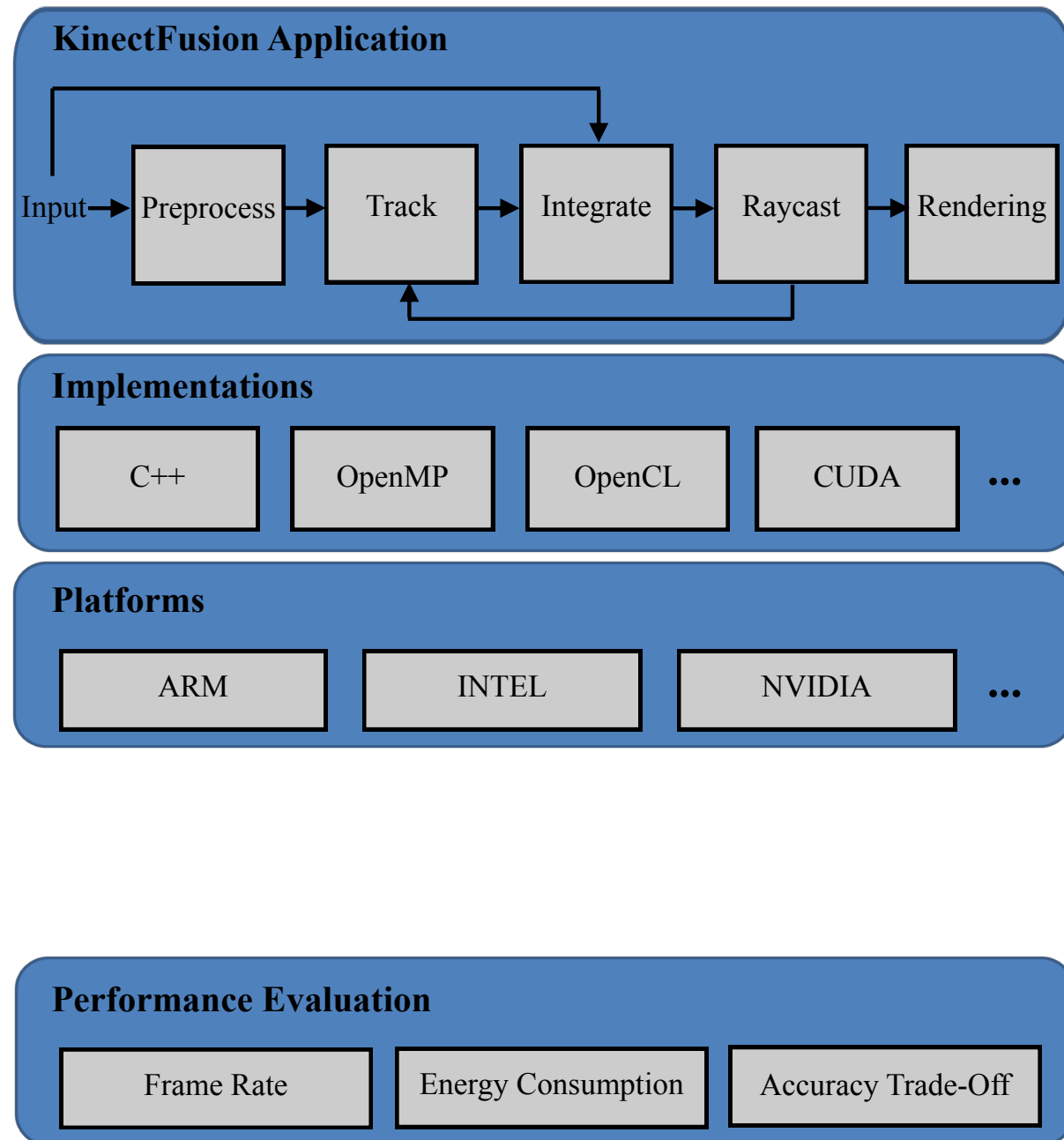


desktop, mobile, embedded:
multi-core and many-core

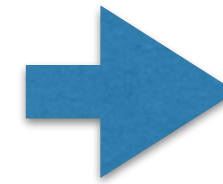


3 metrics:
execution time/energy/accuracy

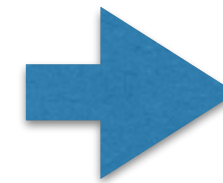
SLAMBench framework



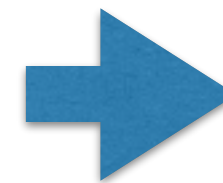
high-level blocks composed
by 14 lower level kernels
based on [Reitmayr 2011]



wide range of languages

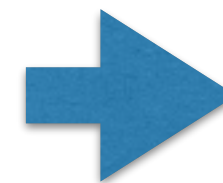
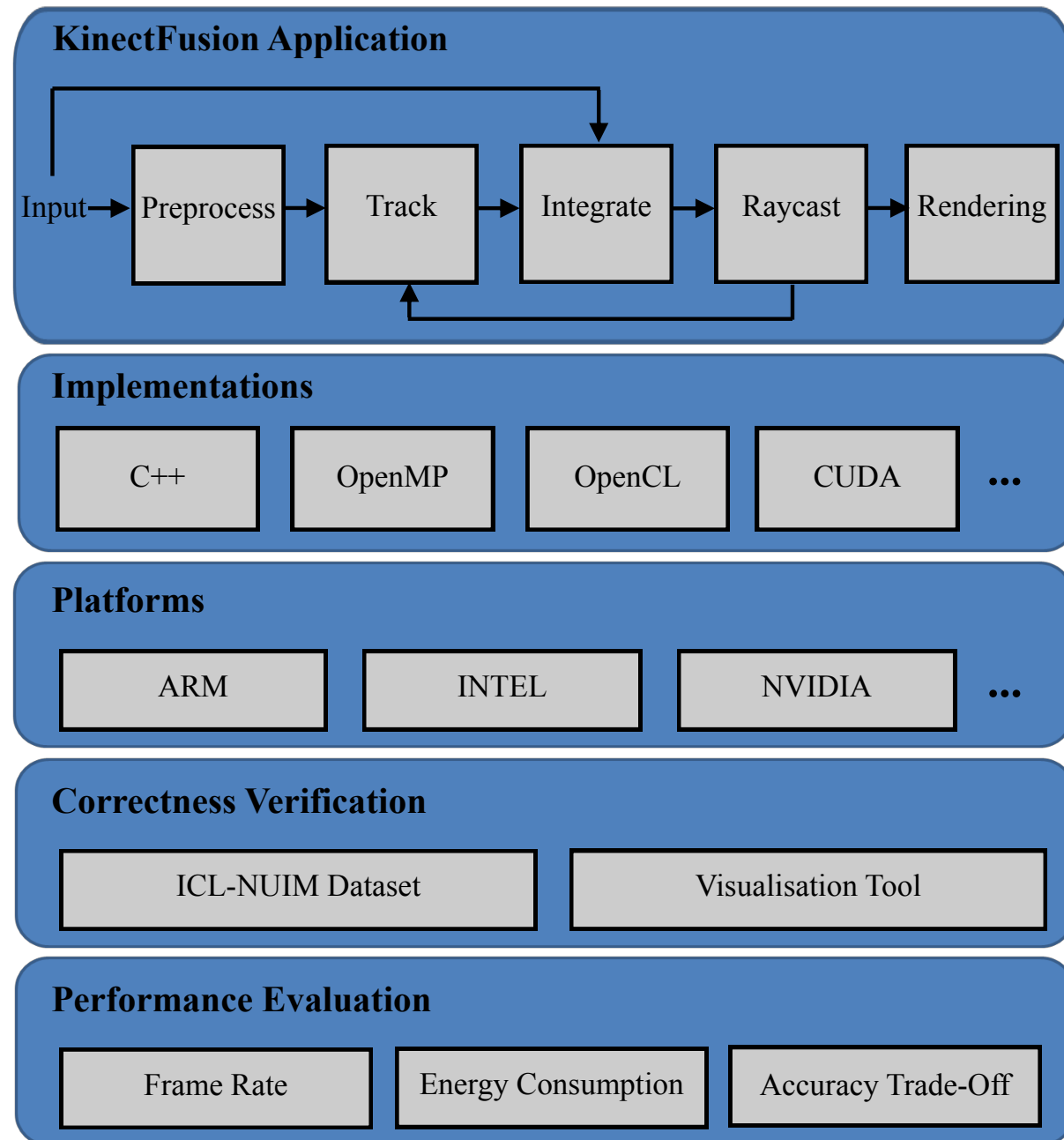


desktop, mobile, embedded:
multi-core and many-core

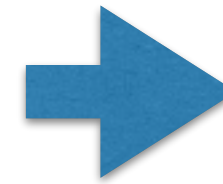


3 metrics:
execution time/energy/accuracy

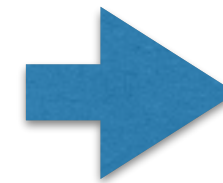
SLAMBench framework



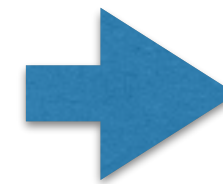
high-level blocks composed
by 14 lower level kernels
based on [Reitmayr 2011]



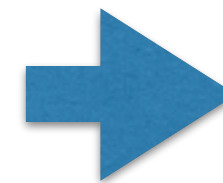
wide range of languages



desktop, mobile, embedded:
multi-core and many-core



integrated
verification/visualisation



3 metrics:
execution time/energy/accuracy

Outline

- Simultaneous localisation and mapping (SLAM) application
- Performance metrics in SLAM
- **From desktop to embedded**
- SLAMBench opportunities



Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60

Platforms



Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



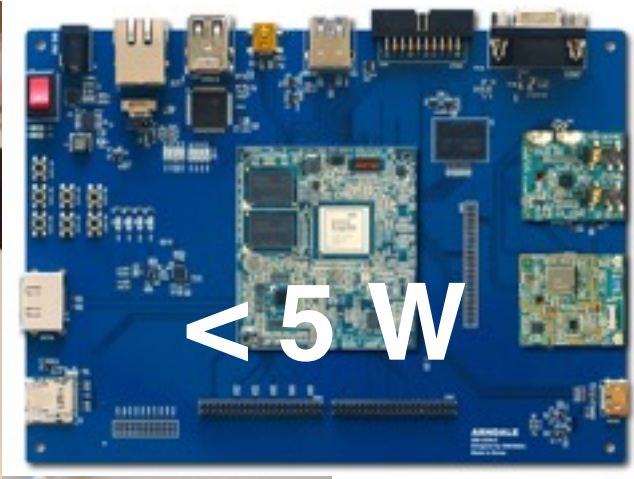
Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



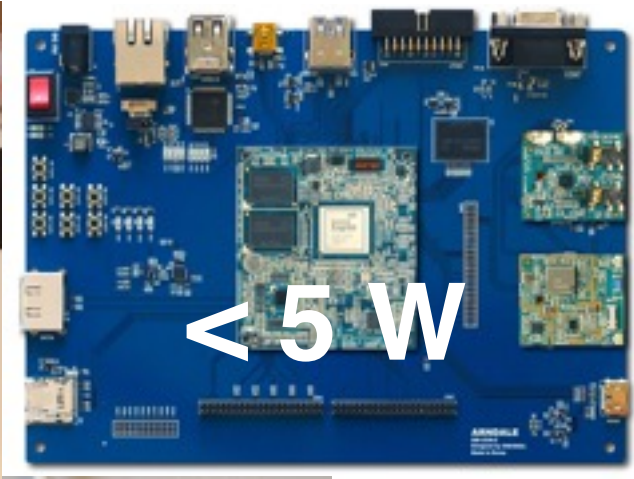
Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



Machines	TITAN	GTX870M	TK1	ODROID	Arndale
CPU	Intel	Intel	ARM	ARM	ARM
CPU name	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU GFLOPS	448	307	74	80	27
CPU cores	4	4	4 + 1	4 + 4	2
GPU	NVIDIA	NVIDIA	NVIDIA	ARM	ARM
GPU name	TITAN	GTX 870M	Tegra K1	Mali-T628	Mali-T604
GPU GFLOPS	4500	2520	330	60 + 30	60



Platforms



“Performance”: accuracy

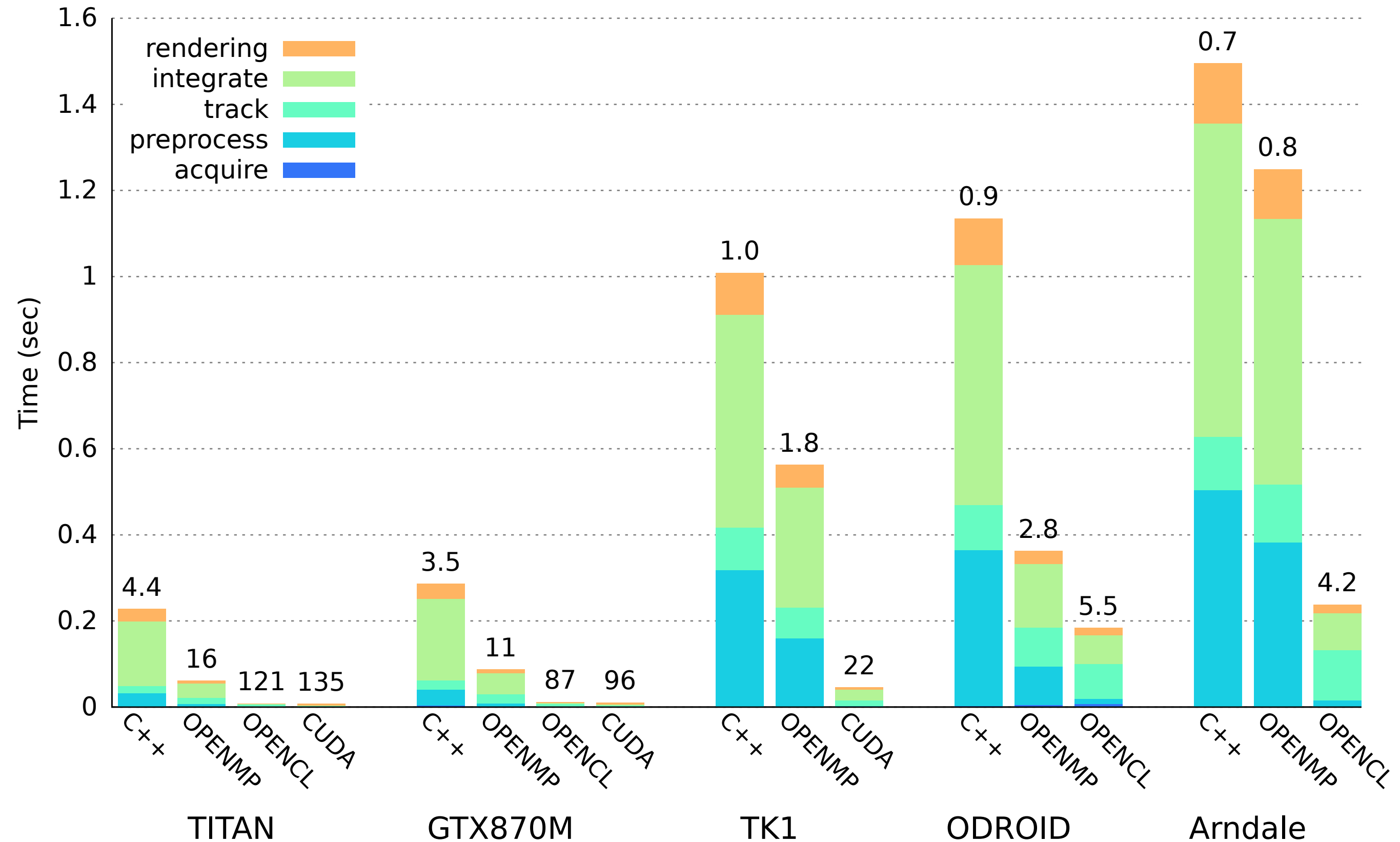
Absolute trajectory error (ATE) in cm - default algorithmic configuration

ATE in cm	TITAN	GTX870M	TK1	ODROID	Arndale
C++	2.07	2.07	2.06	2.06	2.06
OpenMP	2.07	2.07	2.06	2.06	2.06
OpenCL	2.07	2.07	n/a	2.01	2.07
CUDA	2.07	2.07	2.07	n/a	n/a

- ATE easy-to-use tool for non computer vision experts
- Semantic validation instead than bitwise accuracy

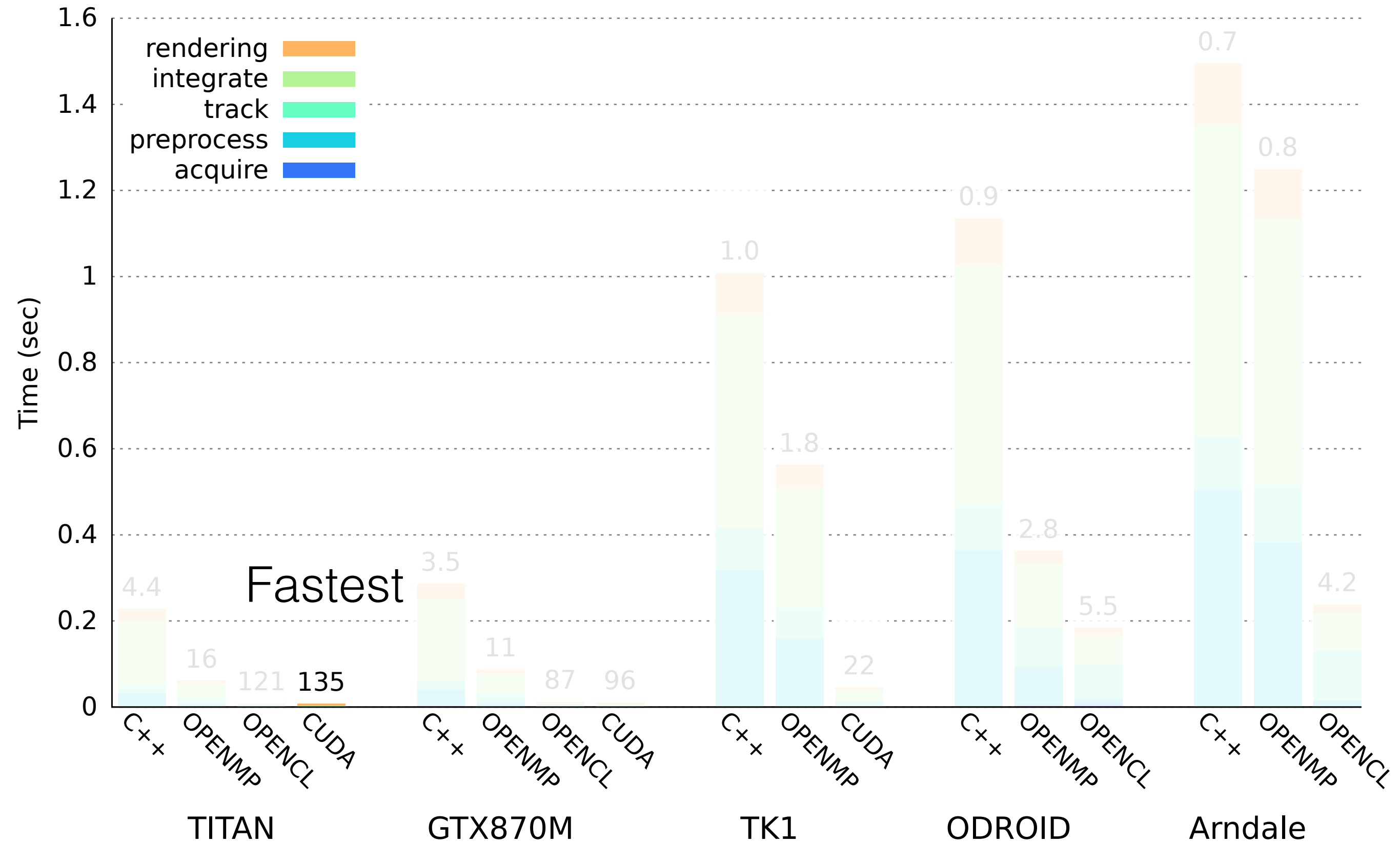
“Performance”: execution time

Mean time per frame (lower is better)



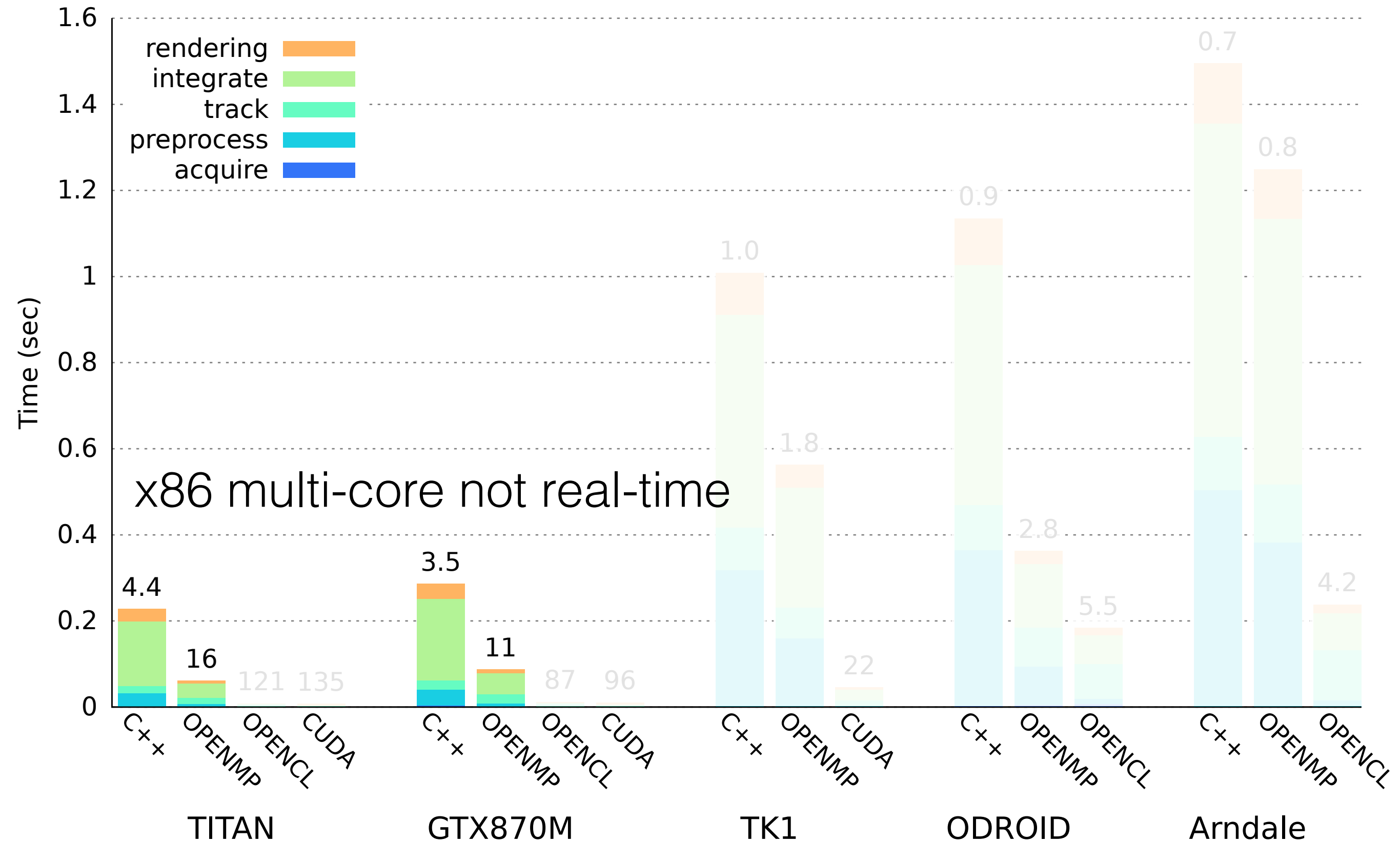
“Performance”: execution time

Mean time per frame (lower is better)



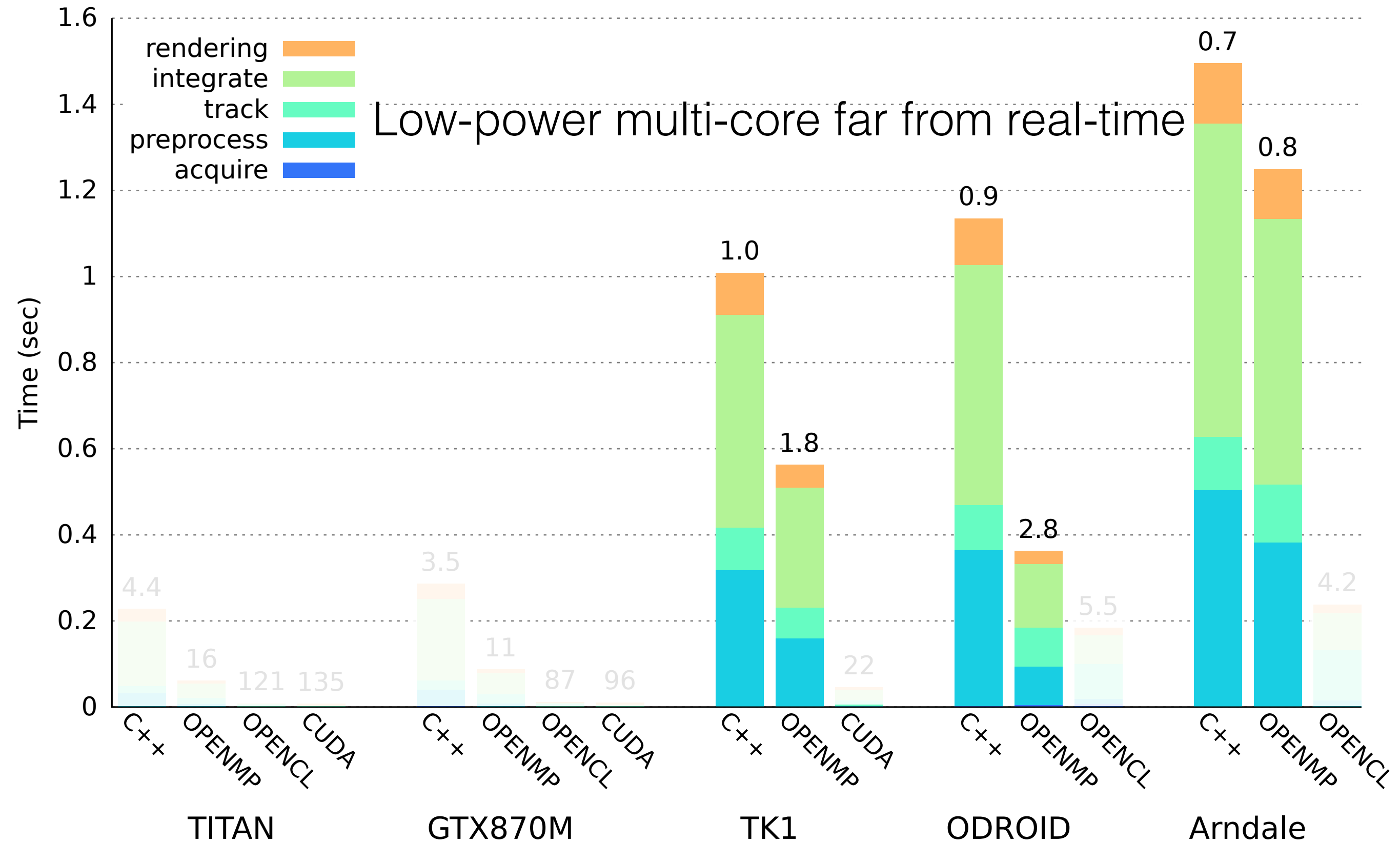
“Performance”: execution time

Mean time per frame (lower is better)



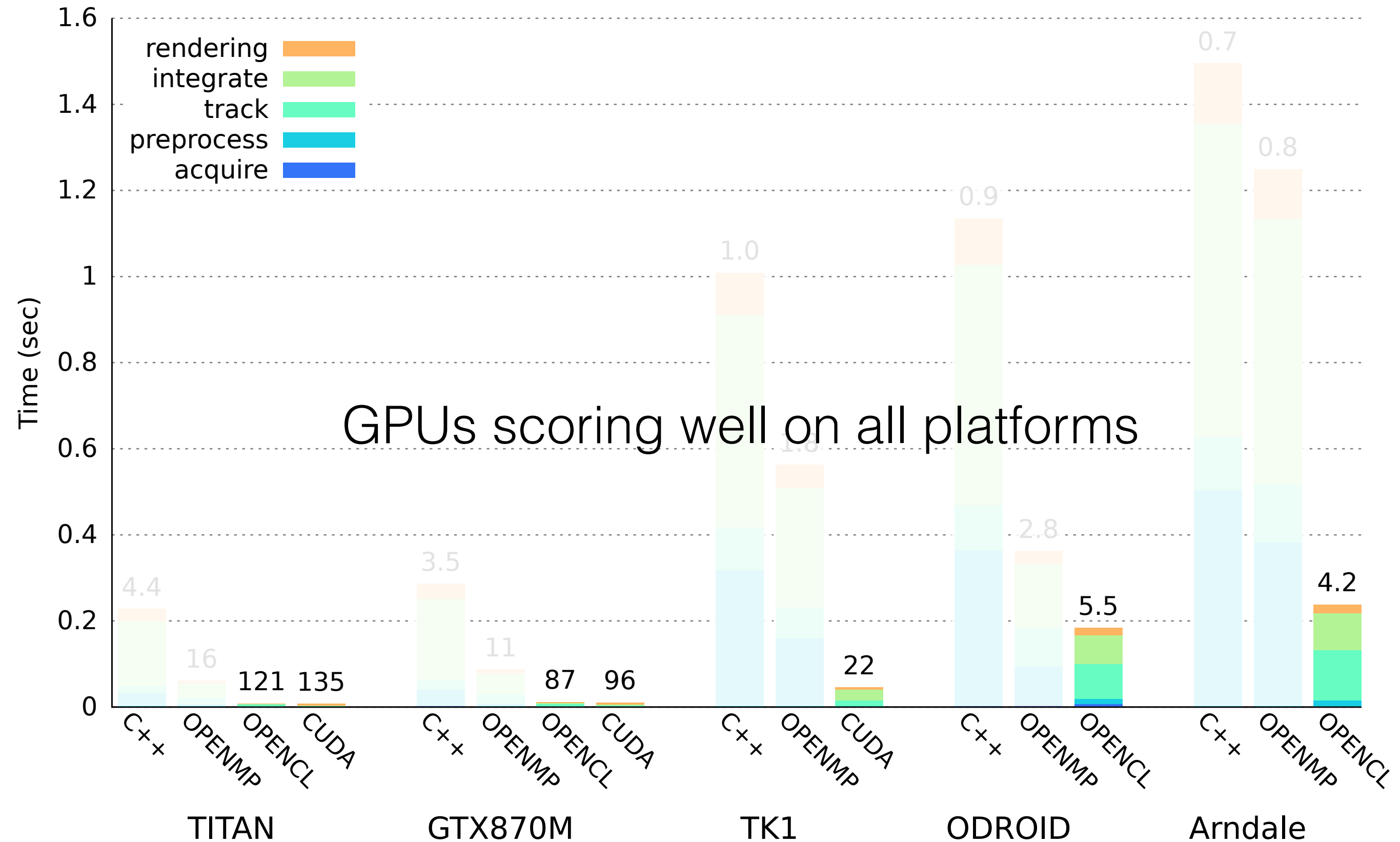
“Performance”: execution time

Mean time per frame (lower is better)



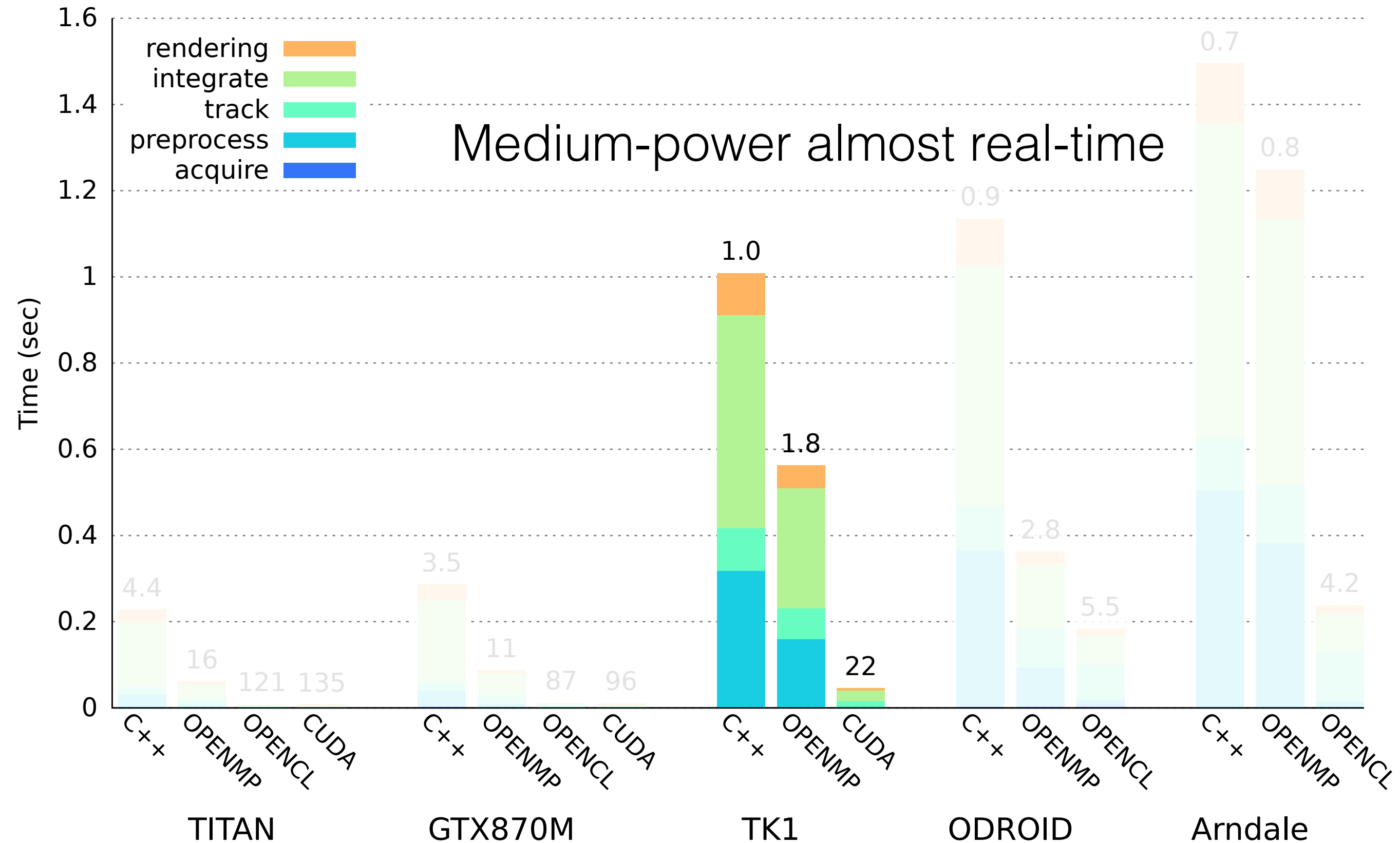
“Performance”: execution time

Mean time per frame (lower is better)



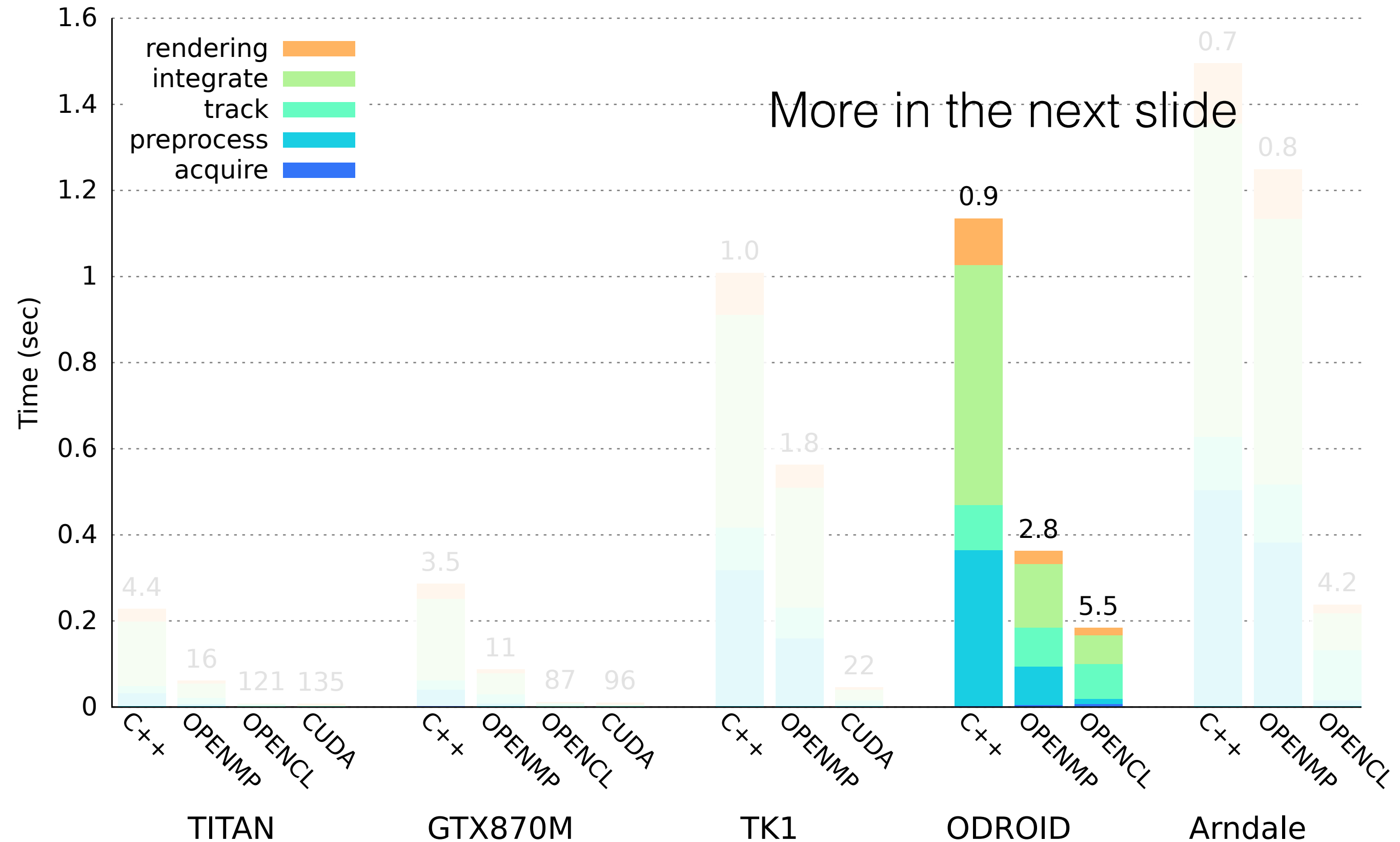
“Performance”: execution time

Mean time per frame (lower is better)



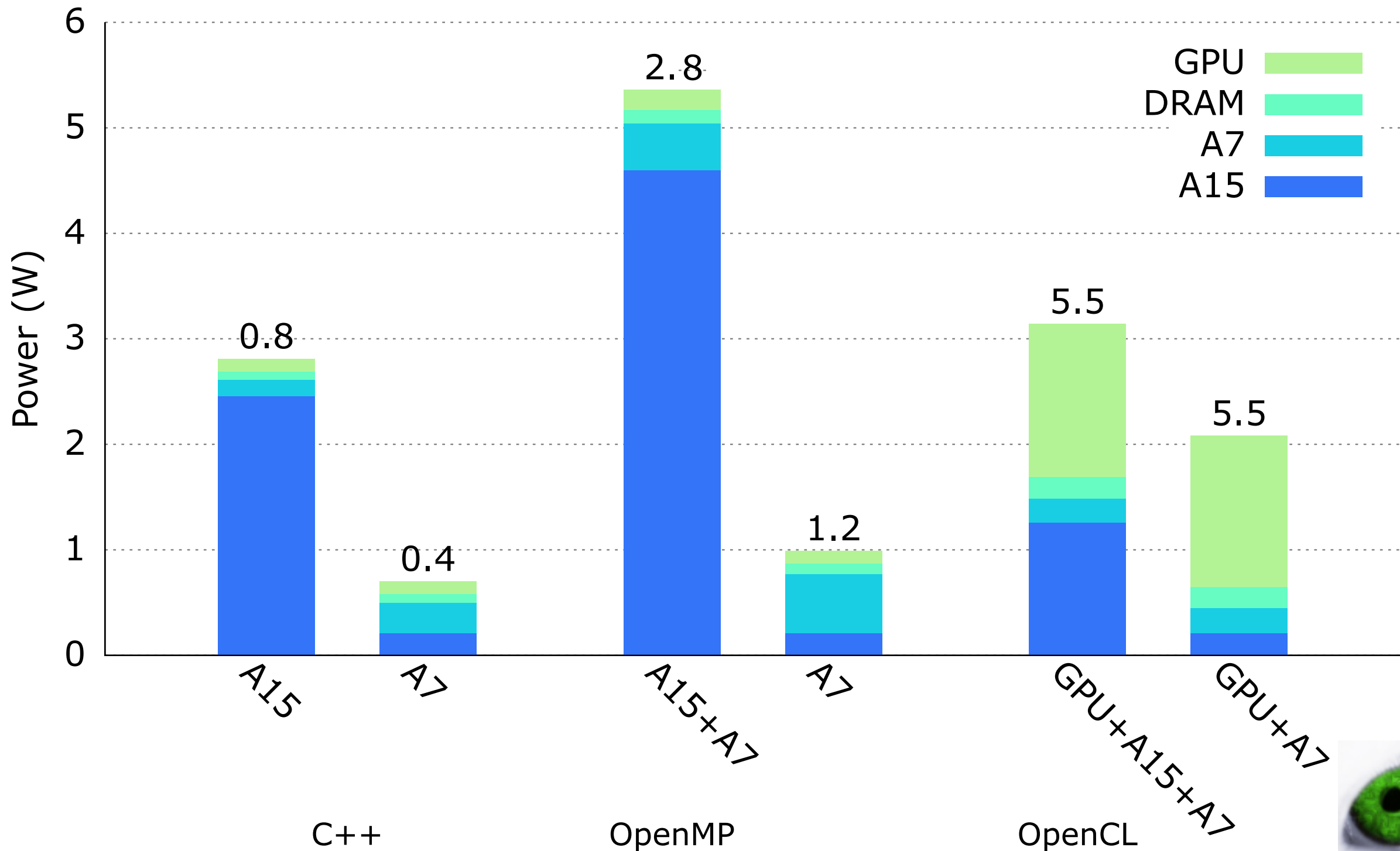
“Performance”: execution time

Mean time per frame (lower is better)



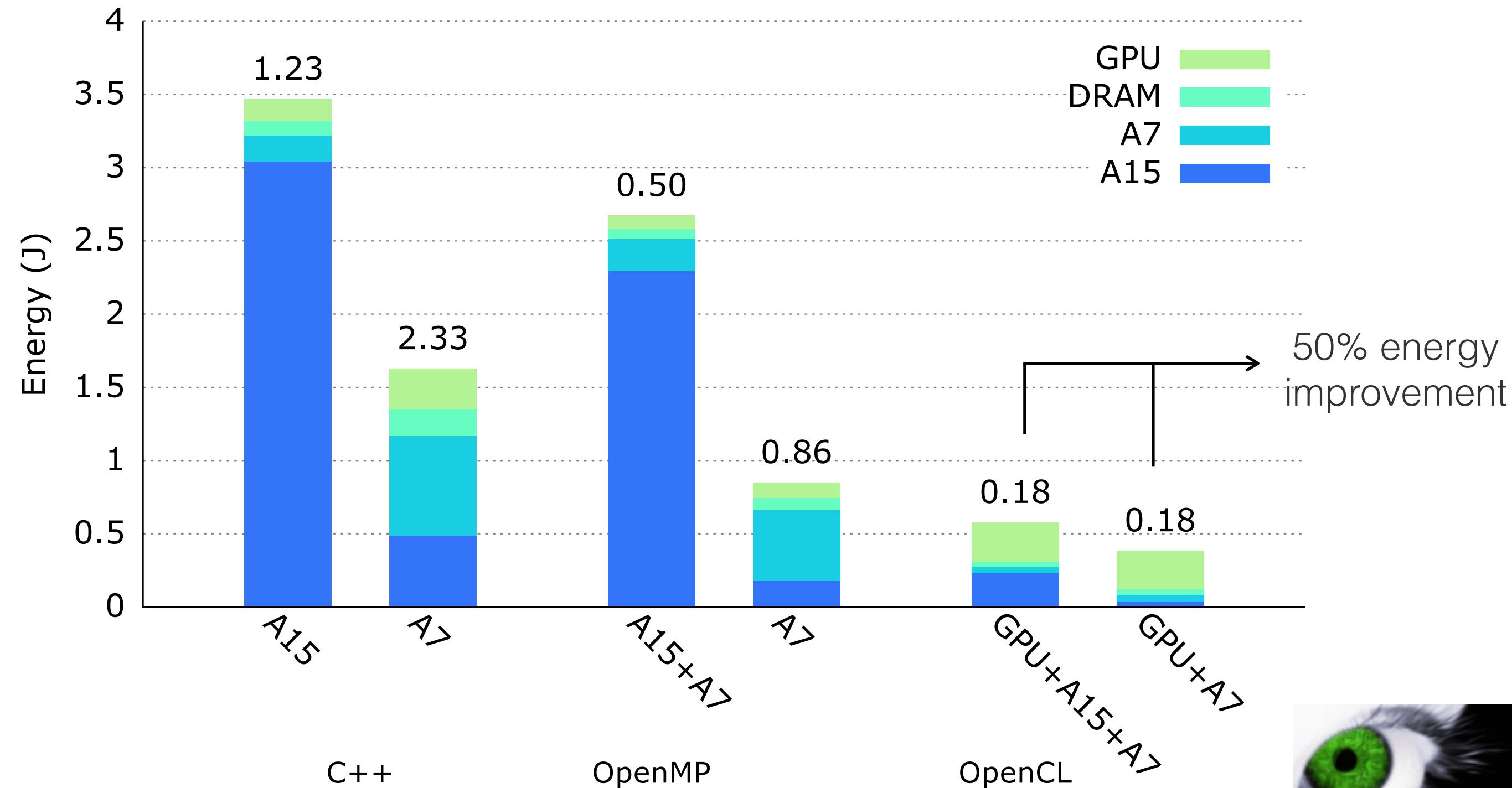
“Performance” power (ODROID-XU3)

On-board voltage/current sensors and split power rails:
power measured individually on big (A15), LITTLE (A7), GPU and DRAM



“Performance”: energy (ODROID-XU3)

Energy usage per frame, mean time as marked



Outline

- Simultaneous localisation and mapping (SLAM) application
- Performance metrics in SLAM
- From desktop to embedded
- **SLAMBench opportunities**



SLAMBench today

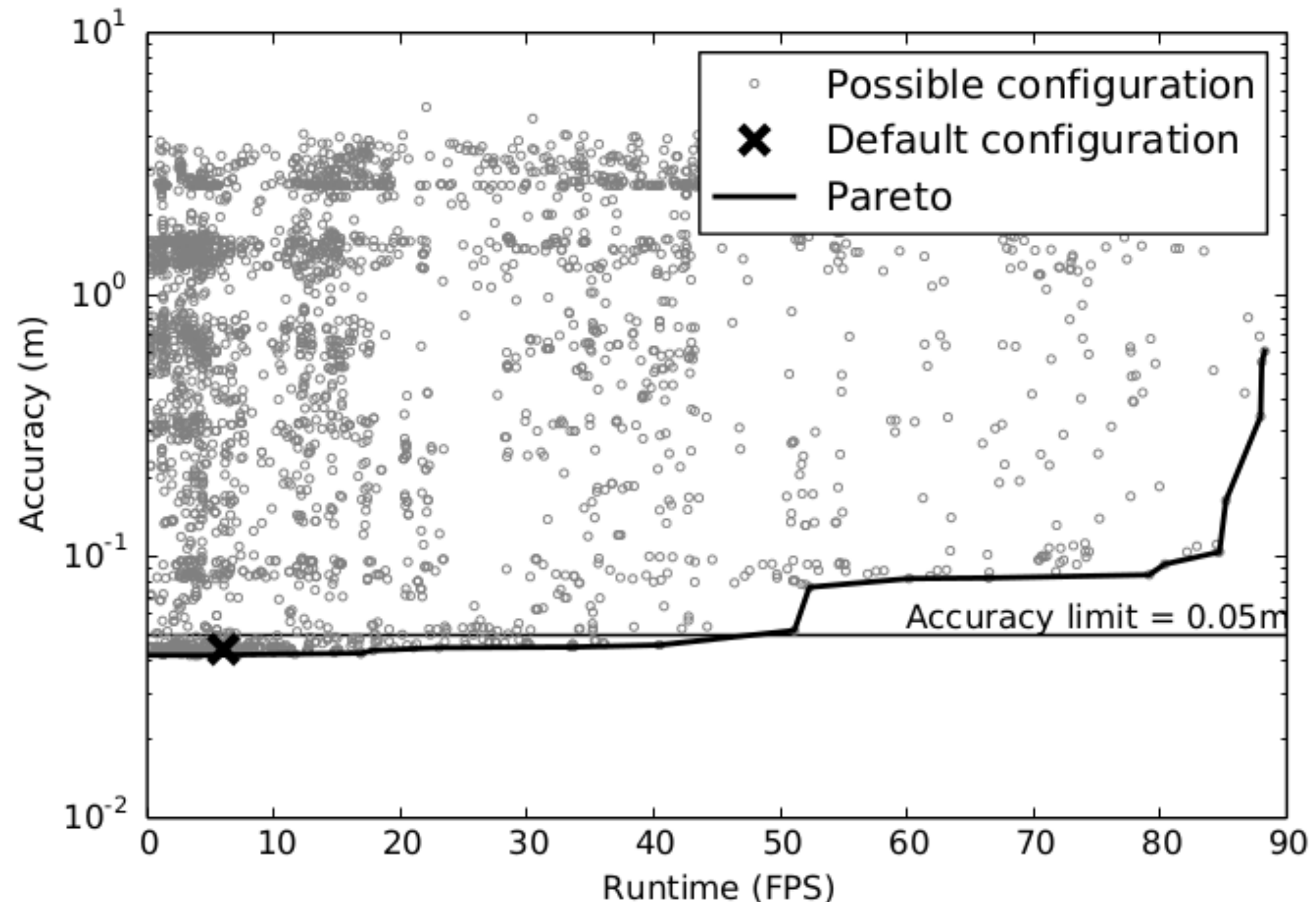
- Publicly released 13/11/2014 (300 downloads today):
<http://apt.cs.manchester.ac.uk/projects/PAMELA/tools/SLAMBench/>
- "*Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM*", Nardi et al., IEEE Int. Conf. on Robotics and Automation (ICRA), May 2015. Available: arxiv.org/abs/1410.2167
- Early adopters (vanity metric):
 - **CV**: Amazon, Microsoft, Toyota, Ocado, Magic Leap, FutureBots, Cnaptic
 - **Compiler/runtime**: ARM, Codeplay and many universities
 - **Architecture**: NVIDIA, Intel, IBM, TI, Movidius, Samsung



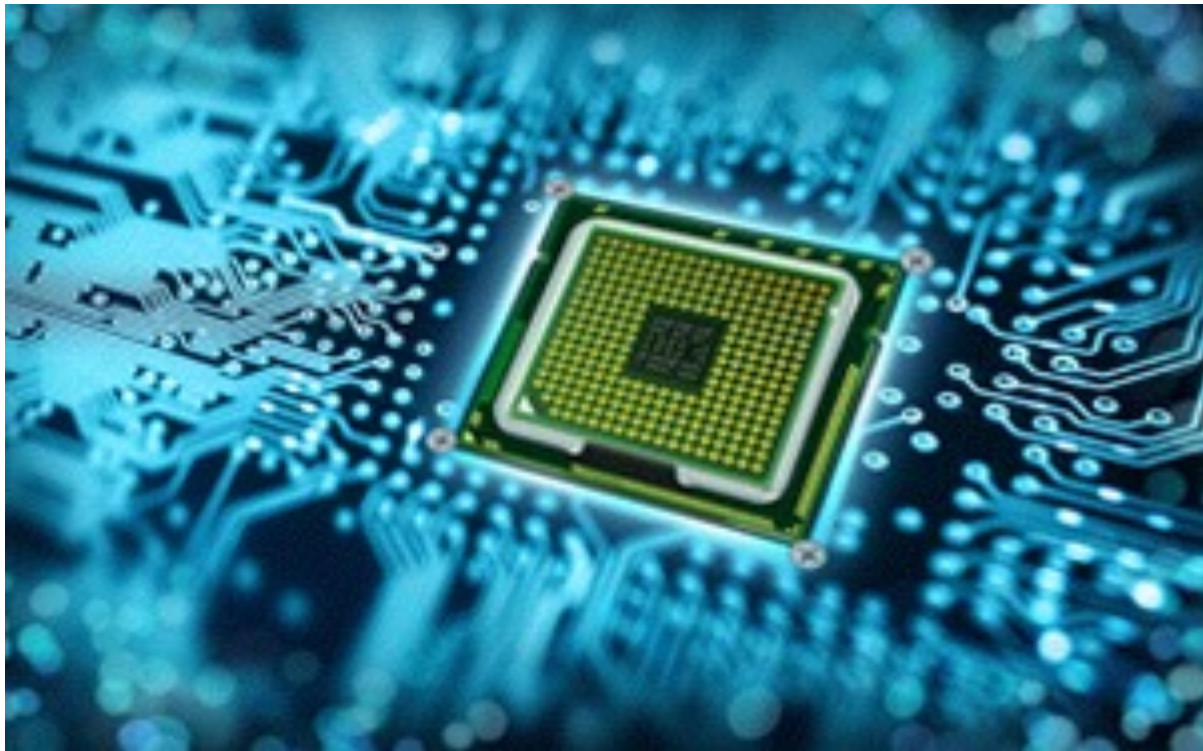
ATE enables aggressive design-space exploration (DSE):

- Algorithmic, compiler and hardware parameters
- Huge space, use machine learning, i.e. active learning

An example:
algorithmic DSE



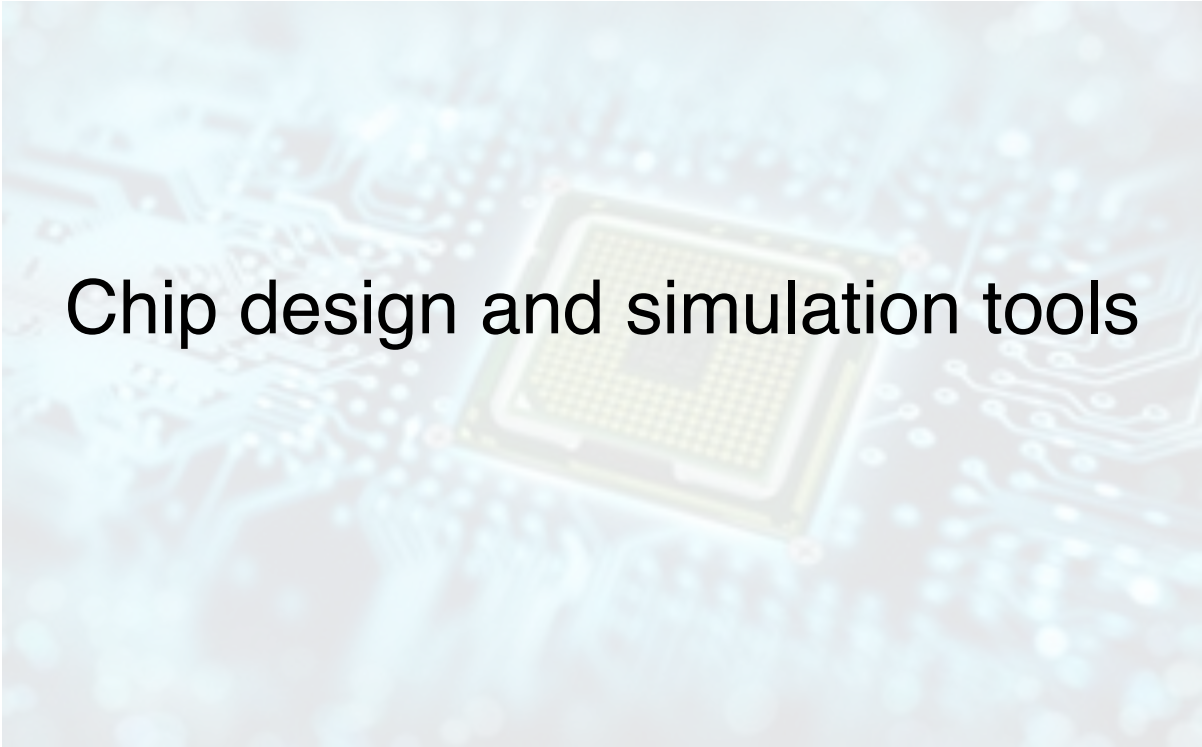
SLAMBench opportunities



Chip design and simulation tools



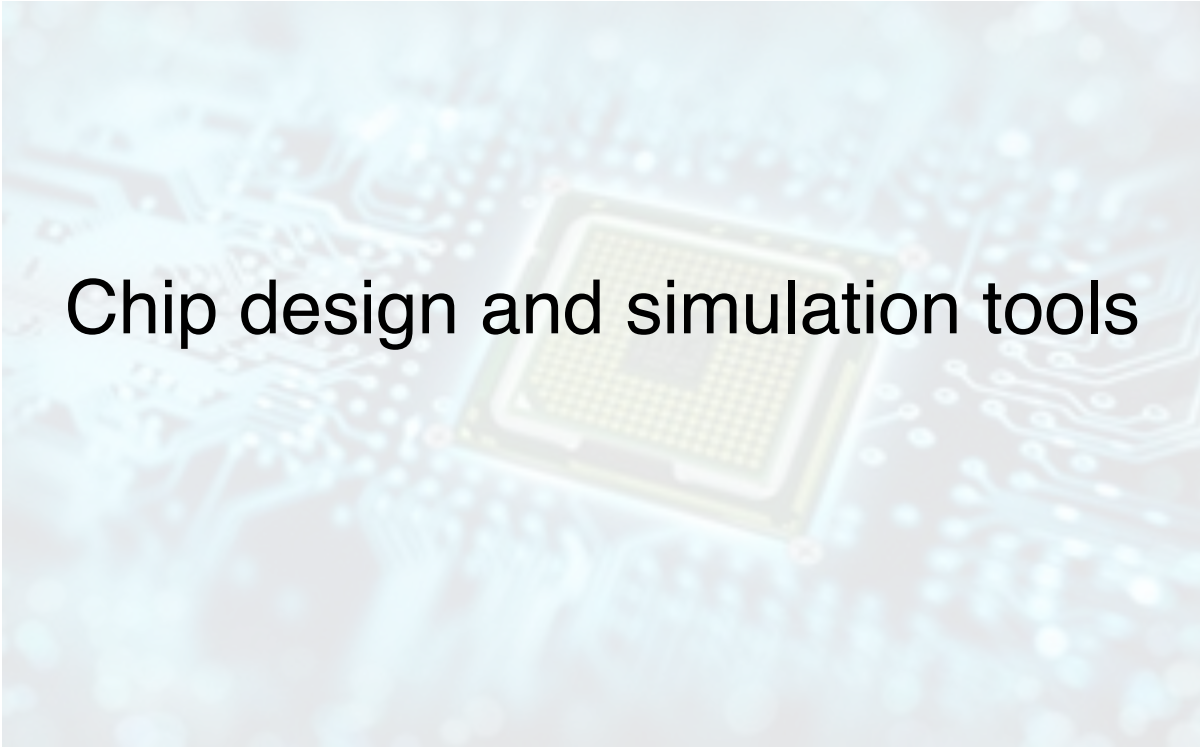
SLAMBench opportunities



Chip design and simulation tools



SLAMBench opportunities



Chip design and simulation tools



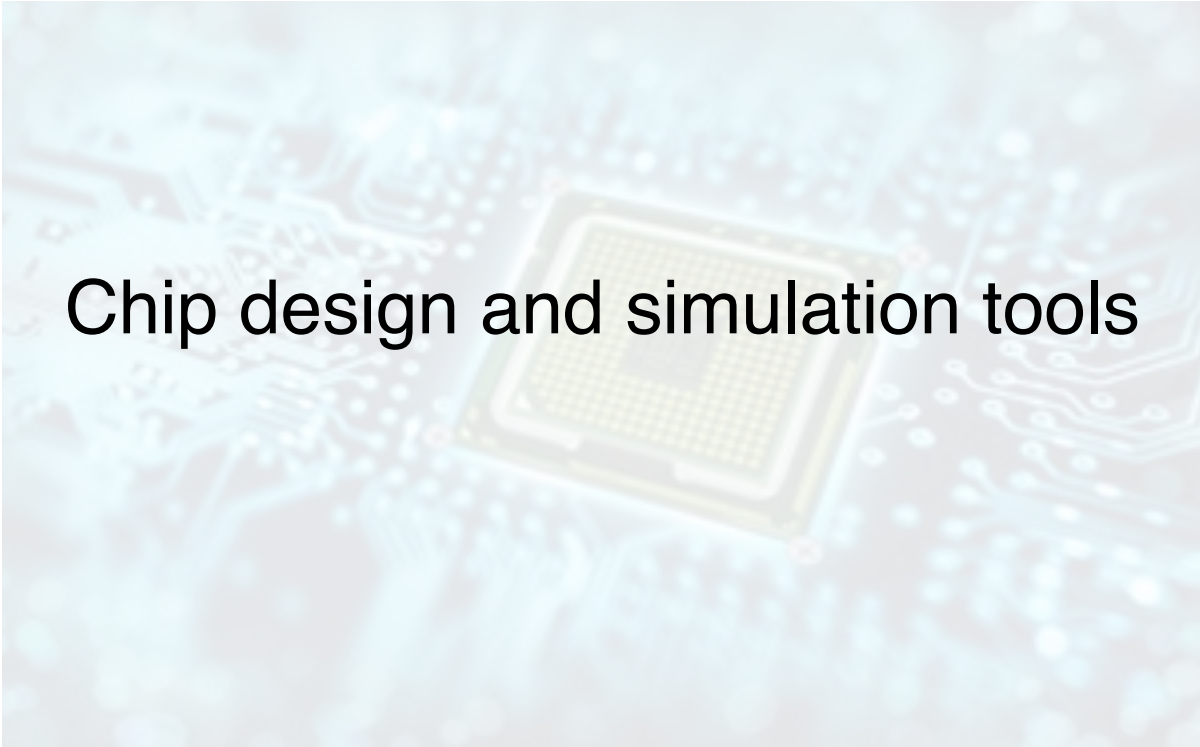
SLAMBench evolution:

- Point fusion
- Octrees
- Semi-dense SLAM
- Feature-based SLAM

Kernels can be improved individually



SLAMBench opportunities



Chip design and simulation tools

SLAMBench evolution:

- Point fusion
- Octrees
- Semi-dense SLAM
- Feature-based SLAM

Kernels can be improved individually



SLAMBench opportunities

Chip design and simulation tools

SLAMBench evolution:

- Point fusion
- Octrees
- Semi-dense SLAM
- Feature-based SLAM

Kernels can be improved individually

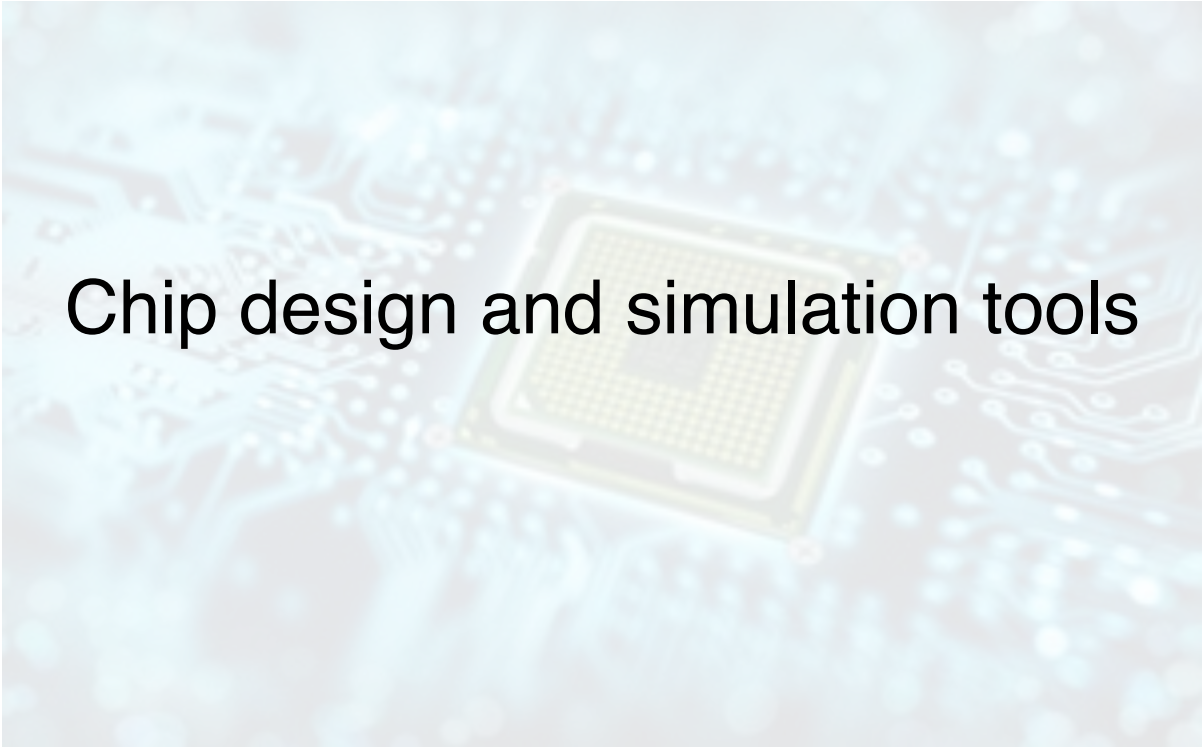
Domain-specific language (DSL) targeting high performance, low-power solutions

Design-space exploration, e.g. algorithmic, compiler and hardware parameters

SLAMBench kernels tuning, e.g. vectorisation, GPU, auto-tuning



SLAMBench opportunities



Chip design and simulation tools

Domain-specific language (DSL) targeting high performance, low-power solutions

Design-space exploration, e.g. algorithmic, compiler and hardware parameters

SLAMBench kernels tuning, e.g. vectorisation, GPU, auto-tuning

SLAMBench evolution:

- Point fusion
- Octrees
- Semi-dense SLAM
- Feature-based SLAM

Kernels can be improved individually



SLAMBench opportunities

Chip design and simulation tools

Domain-specific language (DSL) targeting high performance, low-power solutions

Design-space exploration, e.g. algorithmic, compiler and hardware parameters

SLAMBench kernels tuning, e.g. vectorisation, GPU, auto-tuning

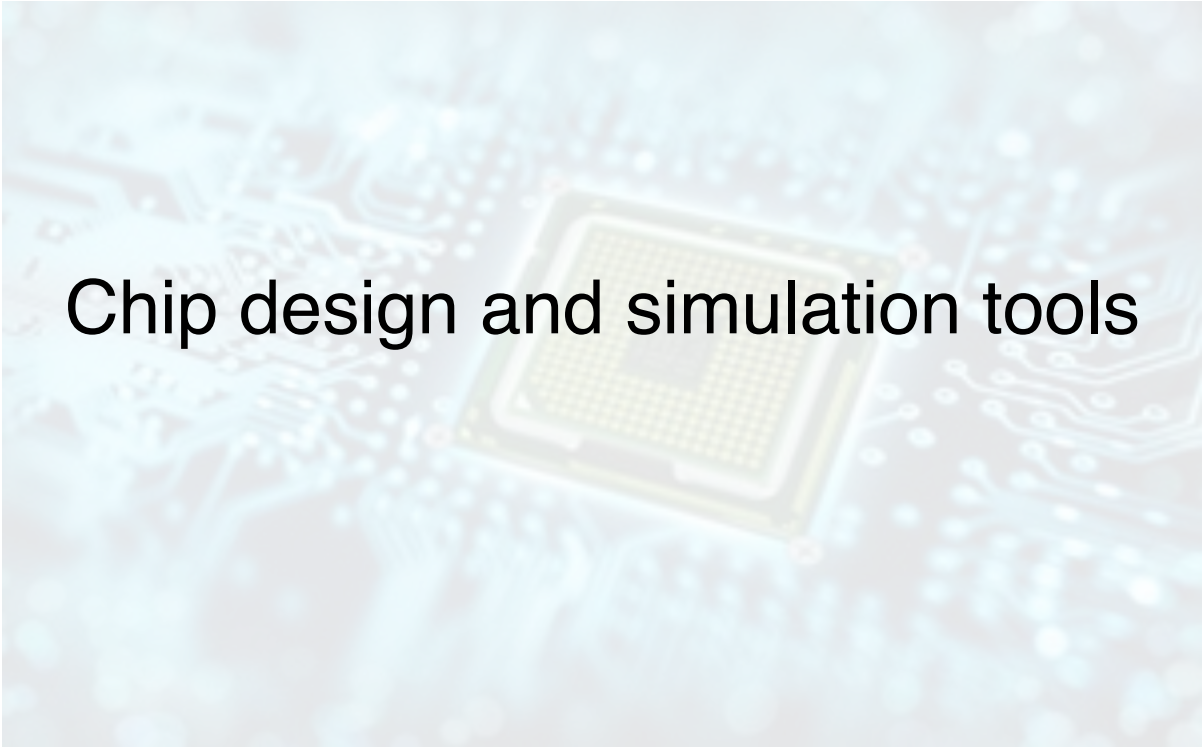
SLAMBench evolution:

- Point fusion
- Octrees
- Semi-dense SLAM
- Feature-based SLAM

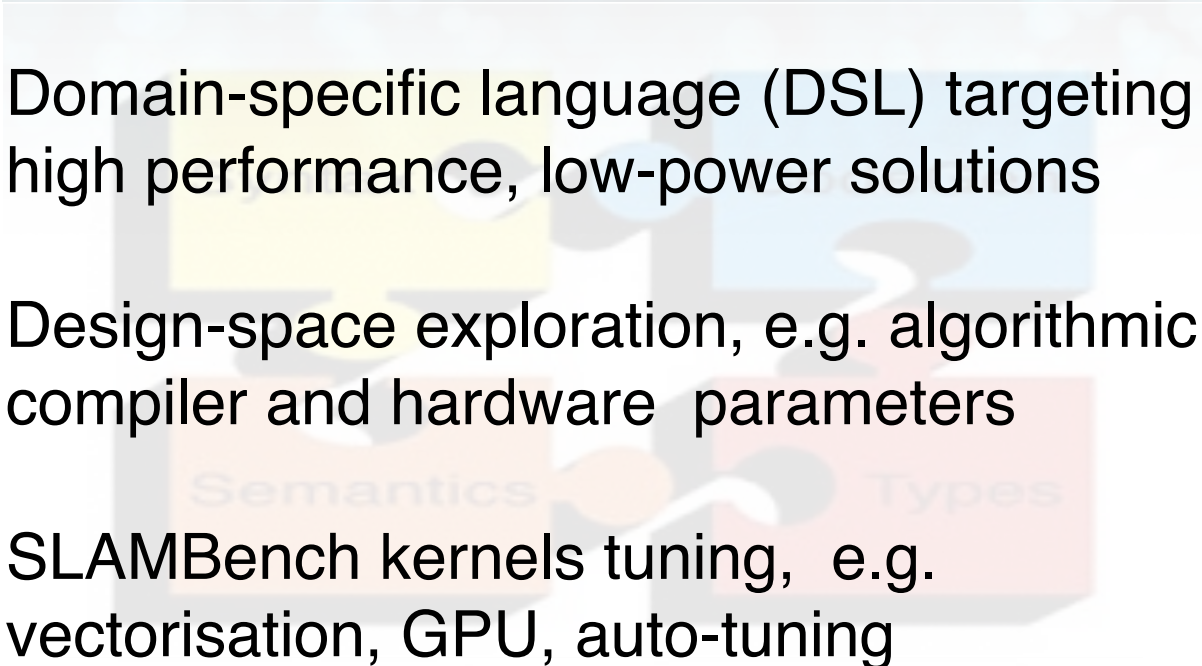
Kernels can be improved individually



SLAMBench opportunities



Chip design and simulation tools

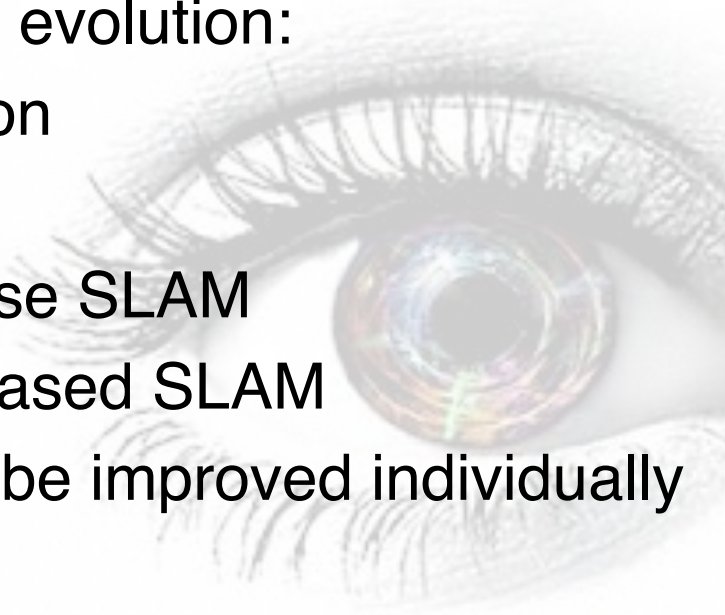


Domain-specific language (DSL) targeting high performance, low-power solutions

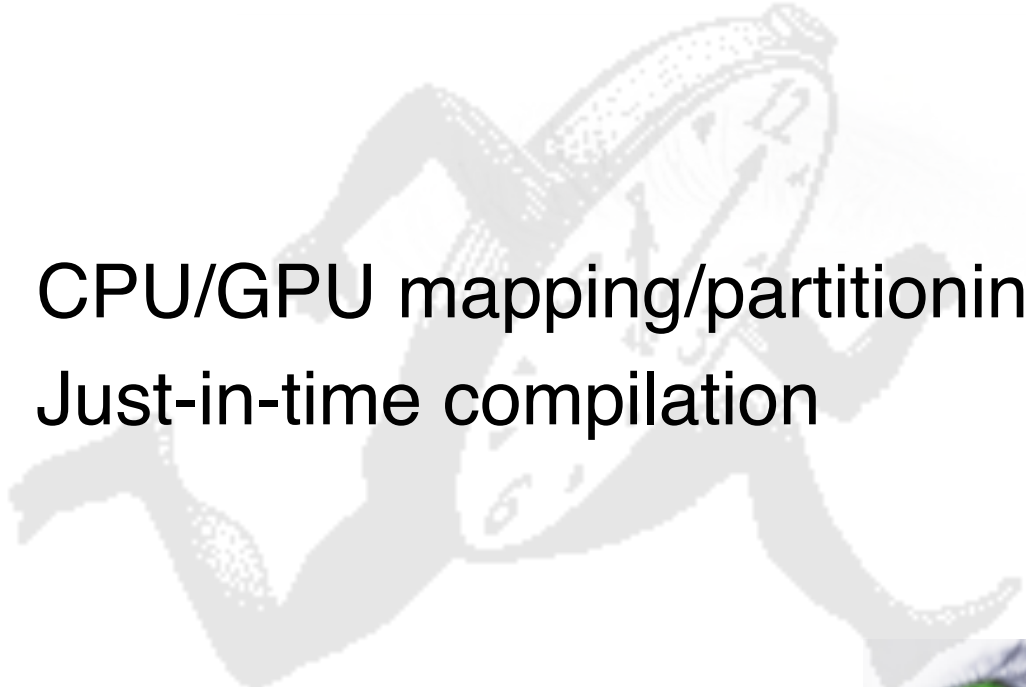
Design-space exploration, e.g. algorithmic, compiler and hardware parameters

SLAMBench kernels tuning, e.g. vectorisation, GPU, auto-tuning

SLAMBench evolution:

- 
- Point fusion
 - Octrees
 - Semi-dense SLAM
 - Feature-based SLAM

Kernels can be improved individually

- 
- CPU/GPU mapping/partitioning
 - Just-in-time compilation



References

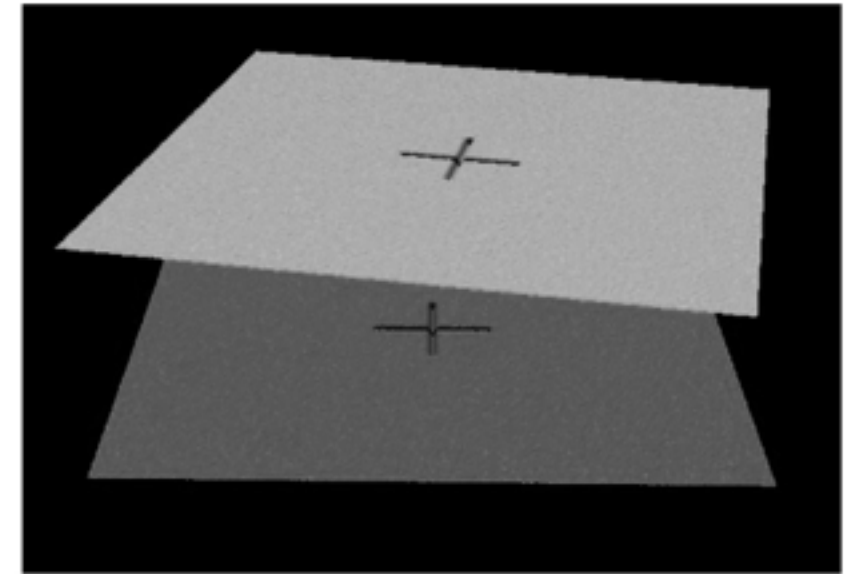
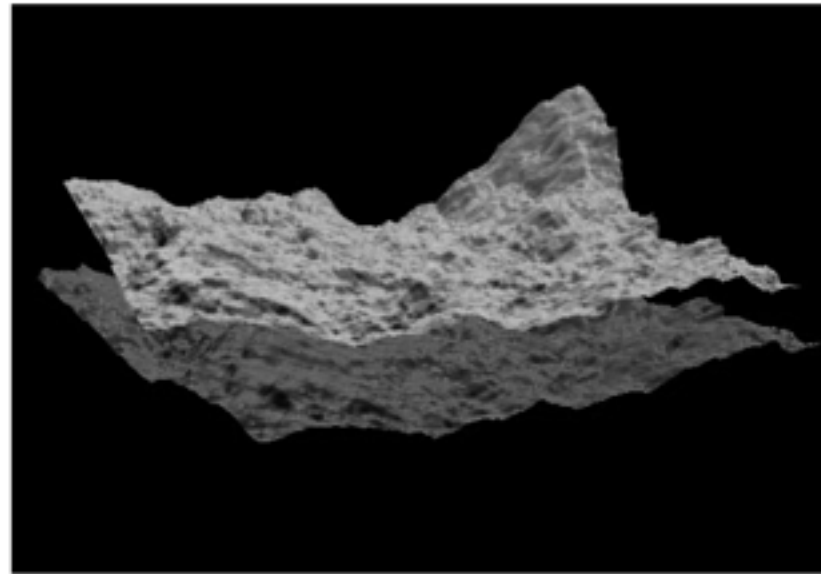
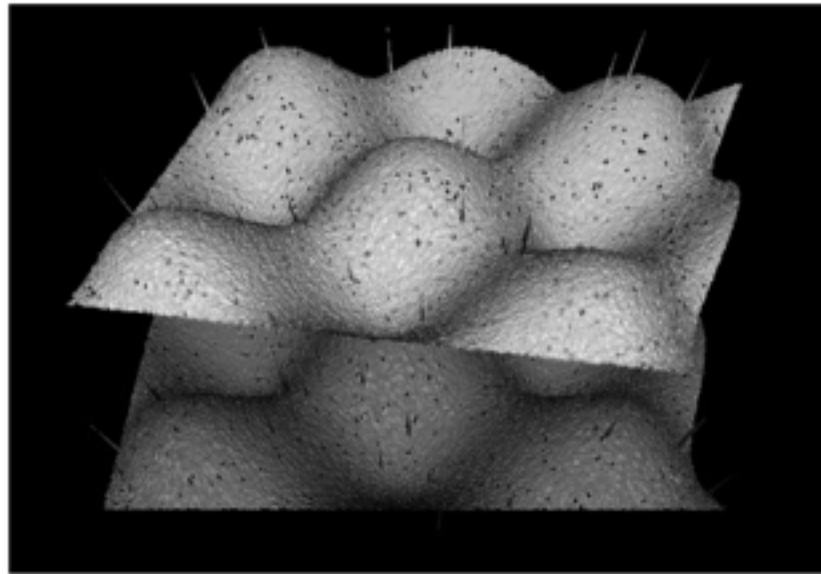
1. [Nardi et al. 2015] L. Nardi, B. Bodin, M. Z. Zia, J. Mawer, A. Nisbet, P. H. J. Kelly, A. J. Davison, M. Luján, M. F. P. O'Boyle, G. Riley, N. Topham, and S. Furber. "Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM." Submitted, arXiv:1410.2167, 2015.
2. [Newcombe et al. ICCV 2011] R. A. Newcombe, S. J. Lovegrove and A. J. Davison. "DTAM: Dense tracking and mapping in real-time." Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011.
3. [Rusinkiewicz and Levoy 2001] S. Rusinkiewicz, and M. Levoy. "Efficient variants of the ICP algorithm." 3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on. IEEE, 2001.
4. [Chen et al. 2013] J. Chen, D. Bautembach, and S. Izadi, Scalable real-time volumetric surface reconstruction, in ACM Trans. Graph., 2013.
5. [Newcombe et al. ISMAR 2011] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. "KinectFusion: Real-time dense surface mapping and tracking." 10th IEEE Int. Symp. on Mixed and augmented reality (ISMAR), 2011.
6. [Handa et al. 2014] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. IEEE Int. Conf. on Robotics and Automation, ICRA 2014.
7. [Reitmayr] G. Reitmayr. KFusion github 2011. <https://github.com/GerhardR/kfusion>
8. [Curless and Levoy 1996] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In Proc. Computer graphics and interactive technique. ACM, 1996.
9. [Whelan et al. 2012] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially extended kinectfusion. 2012.



Backup slides



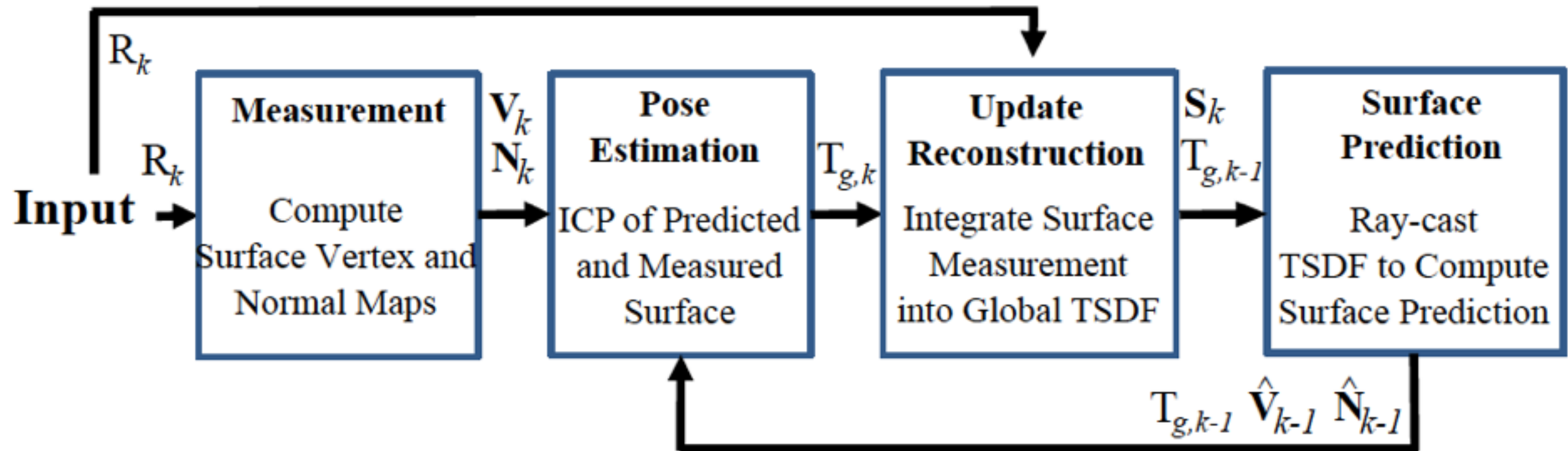
ICP registration



[Rusinkiewicz and Levoy 2001]

- Iterative Closest Point (ICP):
6 DoF rigid body transform from frame $k-1$ to frame k .
- Iterative algorithm computing an energy function minimisation





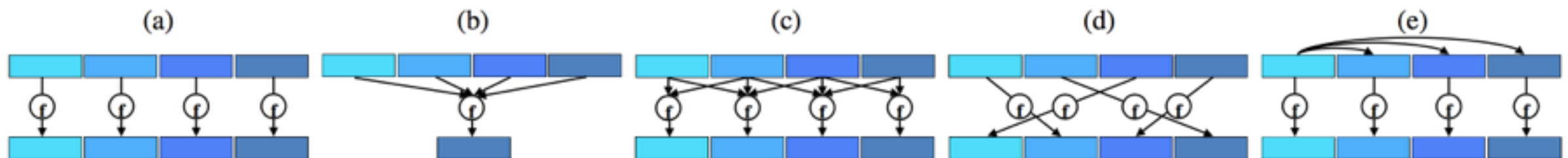
KinectFusion pipeline

- First dense monocular SLAM algorithm [Newcombe et al. ISMAR 2011]
- Adopted as a major building block in more recent SLAM systems
- Implementation based on [Reitmayr 2011] CUDA implementation



SLAMBench kernels

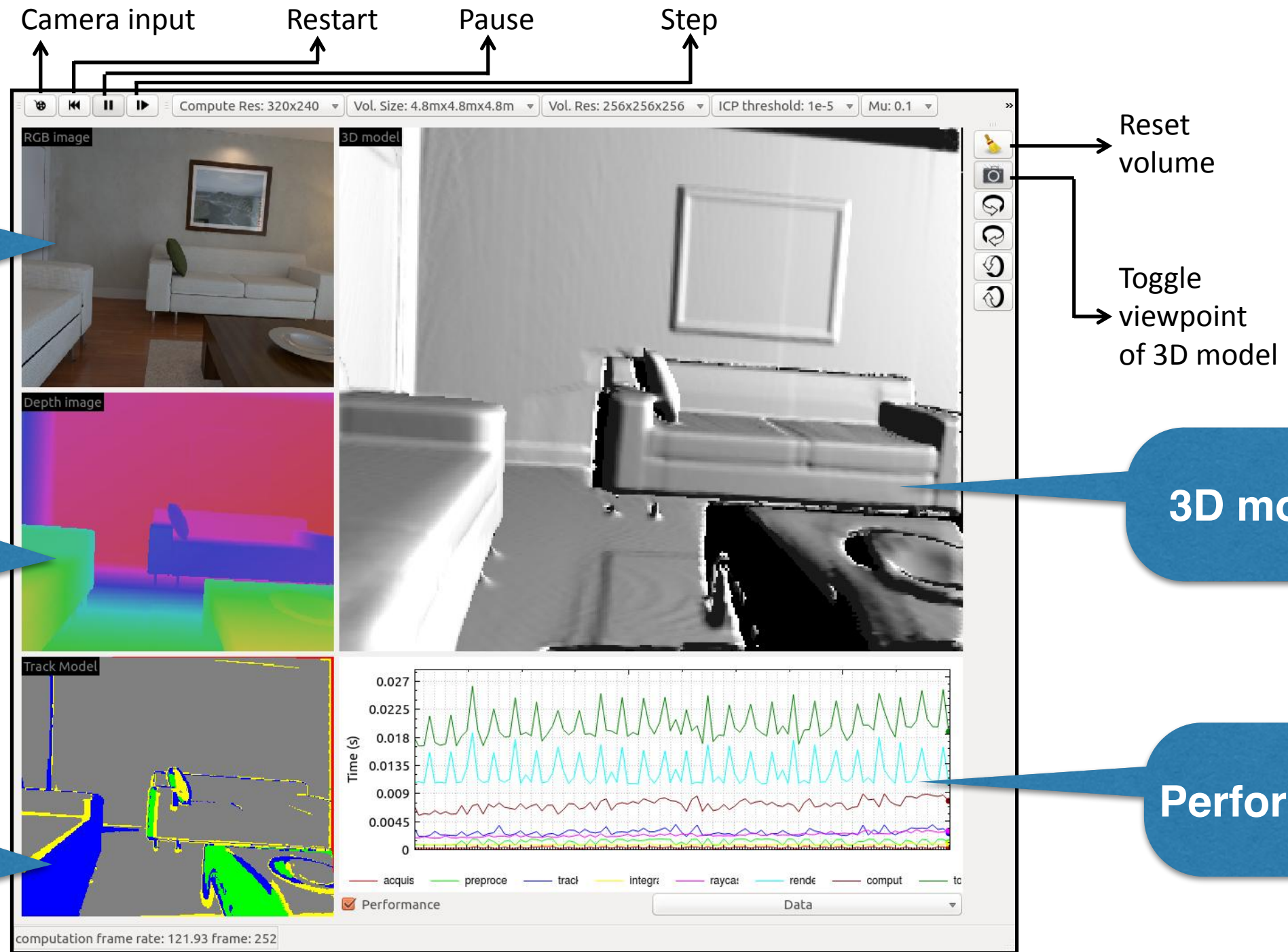
Kernels	Pipeline	Pattern	In	Out	%
acquire	Acquire	n/a	pointer	2D	0.03
mm2meters	Preprocess	Gather	2D	2D	0.06
bilateralFilter	Preprocess	Stencil	2D	2D	33.68
halfSample	Track	Stencil	2D	2D	0.05
depth2vertex	Track	Map	2D	2D	0.11
vertex2normal	Track	Stencil	2D	2D	0.27
track	Track	Map/Gather	2D	2D	4.72
reduce	Track	Reduction	2D	6x6	2.99
solve	Track	Sequential	6x6	6x1	0.02
integrate	Integrate	Map/Gather	2D/3D	3D	12.85
raycast	Raycast	Search/Stencil	2D/3D	2D	35.87
renderDepth	Rendering	Map	2D	2D	0.12
renderTrack	Rendering	Map	2D	2D	0.06
renderVolume	Rendering	Search/Stencil	3D	2D	9.18



Parallel patterns: (a) Map, (b) Reduction, (c) Stencil, (d) Gather and (e) Search.



SLAMBench GUI



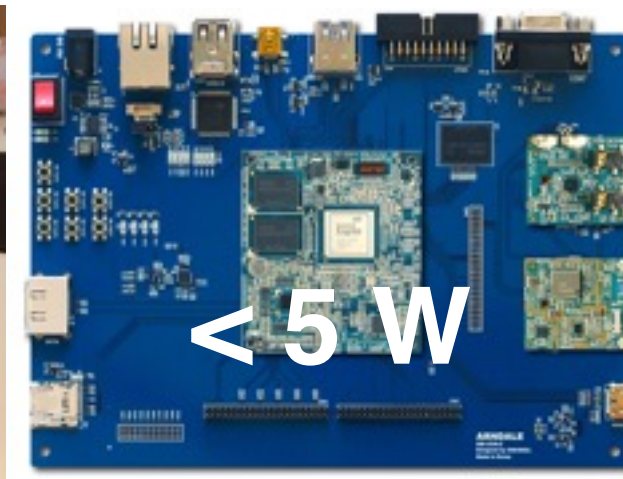
Platforms

Machine names	TITAN	GTX870M	TK1	ODROID (XU3)	Arndale
Machine type	Desktop	Laptop	Embedded	Embedded	Embedded
CPU	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU cores	4	4	4 (Cortex-A15) + 1	4 (Cortex-A15) + 4 (Cortex-A7)	2 (Cortex-A15)
CPU GHz	3.5	2.4	2.3	1.8	1.7
GPU	NVIDIA TITAN	NVIDIA GTX 870M	NVIDIA Tegra K1	ARM Mali-T628-MP6	ARM Mali-T604-MP4
GPU architecture	Kepler	Kepler	Kepler	Midgard 2nd gen.	Midgard 1st gen.
GPU FPU32s	2688	1344	192	60	40
GPU MHz	837	941	852	600	533
GPU GFLOPS (SP)	4500	2520	330	60+30 (72+36)	60 (71)
Language	CUDA/OpenCL/C++	CUDA/OpenCL/C++	CUDA/C++	OpenCL/C++	OpenCL/C++
OpenCL version	1.1	1.1	n/a	1.1	1.1
Toolkit version	CUDA 5.5	CUDA 5.5	CUDA 6.0	Mali SDK1.1.	Mali SDK1.1
Ubuntu OS (kernel)	13.04 (3.8.0)	14.04 (3.13.0)	14.04 (3.10.24)	14.04 (3.10.53)	12.04 (3.11.0)



Platforms

Machine names	TITAN	GTX870M	TK1	ODROID (XU3)	Arndale
Machine type	Desktop	Laptop	Embedded	Embedded	Embedded
CPU	i7 Haswell	i7 Haswell	NVIDIA 4-Plus-1	Exynos 5422	Exynos 5250
CPU cores	4	4	4 (Cortex-A15) + 1	4 (Cortex-A15) + 4 (Cortex-A7)	2 (Cortex-A15)
CPU GHz	3.5	2.4	2.3	1.8	1.7
GPU	NVIDIA TITAN	NVIDIA GTX 870M	NVIDIA Tegra K1	ARM Mali-T628-MP6	ARM Mali-T604-MP4
GPU architecture	Kepler	Kepler	Kepler	Midgard 2nd gen.	Midgard 1st gen.
GPU FPU32s	2688	1344	192	60	40
GPU MHz	837	941	852	600	533
GPU GFLOPS (SP)	4500	2520	330	60+30 (72+36)	60 (71)
Language	CUDA/OpenCL/C++	CUDA/OpenCL/C++	CUDA/C++	OpenCL/C++	OpenCL/C++
OpenCL version	1.1	1.1	n/a	1.1	1.1
Toolkit version	CUDA 5.5	CUDA 5.5	CUDA 6.0	Mali SDK1.1.	Mali SDK1.1
Ubuntu OS (kernel)	13.04 (3.8.0)	14.04 (3.13.0)	14.04 (3.10.24)	14.04 (3.10.53)	12.04 (3.11.0)



Textual User Interface



Easy
run

```
hickory% make 2.openc1.log
LD_PRELOAD=./build/kfusion/thirdparty/liboclwrapper.so ./build/kfusion/kfusion-benchmark-openc1 -s 4.8 -p 0.34,0.5,0.24 -z 4 -c 2 -r 2 -k 481.2,480,320,240 -i living_room_traj2_
enc1.log 2> oclwrapper.2.openc1.log
End of file(garbage found).
cat oclwrapper.2.openc1.log |grep -E ".+ [0-9]+ [0-9]+ [0-9]+" |cut -d" " -f1,4 > kernels.2.openc1.log
./kfusion/thirdparty/checkPos.py benchmark.2.openc1.log livingRoom2.gt.freiburg > resume.2.openc1.log
./kfusion/thirdparty/checkKernels.py kernels.2.openc1.log >> resume.2.openc1.log
hickory%
```

For benchmarking purposes,
SSH and post-processing
tools friendly



Textual User Interface

Easy
run

Raw
data

For benchmarking purposes,
SSH and post-processing
tools friendly



```
hickory% make 2.openc1.log
LD_PRELOAD=./build/kfusion/thirdparty/liboclwrapper.so ./build/kfusion/kfusion-benchmark-openc1 -s 4.8 -p 0.34,0.5,0.24 -z 4 -c 2 -r 2 -k 481.2,480,320,240 -i living_room_traj2_
encl.log 2> oclwrapper.2.openc1.log
End of file(garbage found).
cat oclwrapper.2.openc1.log |grep -E "[0-9]+ [0-9]+ [0-9]+" |cut -d" " -f1,4 > kernels.2.openc1.log
./kfusion/thirdparty/checkPos.py benchmark.2.openc1.log livingRoom2.gt.freiburg > resume.2.openc1.log
./kfusion/thirdparty/checkKernels.py kernels.2.openc1.log >> resume.2.openc1.log
hickory%
```

frame	acquisition	preprocessing	tracking	integration	raycasting	rendering	computation	total	X	Y	Z	tracked	inte
0	0.003564	0.000451	0.001622	0.000461	0.000002	0.001124	0.002536	0.007223	0.0000	0.000000	0.000000	0	1
1	0.002724	0.000352	0.000903	0.000467	0.000001	0.000212	0.001722	0.004658	0.0000	0.000000	0.000000	0	1
2	0.002697	0.000336	0.000904	0.000459	0.000001	0.000212	0.001701	0.004611	0.0000	0.000000	0.000000	0	1
3	0.002663	0.000335	0.000905	0.000467	0.000004	0.000219	0.002311	0.005193	0.0000	0.000000	0.000000	0	1

Textual User Interface

Easy
run

Raw
data

Sum
up

For benchmarking purposes,
SSH and post-processing
tools friendly



```
hickory% make 2.openc1.log
LD_PRELOAD=./build/kfusion/thirdparty/liboclwrapper.so ./build/kfusion/kfusion-benchmark-openc1 -s 4.8 -p 0.34,0.5,0.24 -z 4 -c 2 -r 2 -k 481.2,480,320,240 -i living_room_traj2_
encl.log 2> oclwrapper.2.openc1.log
End of file(garbage found).
cat oclwrapper.2.openc1.log |grep -E "[0-9]+ [0-9]+ [0-9]+" |cut -d" " -f1,4 > kernels.2.openc1.log
./kfusion/thirdparty/checkPos.py benchmark.2.openc1.log livingRoom2.gt.freiburg > resume.2.openc1.log
./kfusion/thirdparty/checkKernels.py kernels.2.openc1.log >> resume.2.openc1.log
hickory%
```

frame	acquisition	preprocessing	tracking	integration	raycasting	rendering	computation	total	X	Y	Z	tracked	inte
grated													
0	0.003564	0.000451	0.001622	0.000461	0.000002	0.001124	0.002536	0.007223	0.0000	0.000000	0.000000	0	1
1	0.002724	0.000352	0.000903	0.000467	0.000001	0.000212	0.001722	0.004658	0.0000	0.000000	0.000000	0	1
2	0.002697	0.000336	0.000904	0.000459	0.000001	0.000212	0.001701	0.004611	0.0000	0.000000	0.000000	0	1
3	0.002663	0.000335	0.000905	0.000467	0.000604	0.000219	0.002311	0.005193	0.0000	0.000000	0.000000	0	1

```
1 Get kfusion output data.
2 Skip kfusion line :
3 Skip nuim line :
4 kfusion file: Valid frames 882 dropped frames: 0
5 kfusion result      : 882 positions.
6 NUIM result        : 880 positions.
7 Working position is : 880
8 Untracked frames: 0
9 Shift kfusion trajectory...
10
11 A detailed statistical analysis is provided.
12 All durations are in seconds and the absolute trajectory error (ATE) is in centimeters.
13 ATE Min : 0.000000 Max : 0.049309 Mean : 0.020662 Total : 18.18239335
14 acquisition Min : 0.000056 Max : 0.009033 Mean : 0.002044 Total : 1.80289800
15 computation Min : 0.001701 Max : 0.009234 Mean : 0.005894 Total : 5.19872500
16 integration Min : 0.000001 Max : 0.000821 Mean : 0.000258 Total : 0.22773500
17 preprocessing Min : 0.000284 Max : 0.001884 Mean : 0.000441 Total : 0.38904600
18 raycasting Min : 0.000001 Max : 0.003313 Mean : 0.001345 Total : 1.18644400
19 rendering Min : 0.000201 Max : 0.003452 Mean : 0.000572 Total : 0.50420900
20 total Min : 0.003882 Max : 0.020054 Mean : 0.008510 Total : 7.50584700
21 tracking Min : 0.000903 Max : 0.006743 Mean : 0.003850 Total : 3.39543600
22 Get SlamBench data.
23 ResetVolume Count : 1 Min : 401056 Max : 401056 Mean : 401056.000000 Total : 401056
24 bilateral_filter Count : 882 Min : 139520 Max : 205632 Mean : 159880.199546 Total : 141014336
25 depth2vertex Count : 2646 Min : 5440 Max : 48384 Mean : 10585.167045 Total : 28008352
26 halfSampleRobust Count : 1764 Min : 6624 Max : 328480 Mean : 9546.557823 Total : 16840128
27 integrate Count : 443 Min : 270880 Max : 793664 Mean : 485648.469526 Total : 215142272
28 mm2meters Count : 882 Min : 6912 Max : 13632 Mean : 8977.306122 Total : 7917984
29 raycast Count : 879 Min : 435584 Max : 3284768 Mean : 1323028.423208 Total : 1162941984
30 reduce Count : 12117 Min : 31488 Max : 523488 Mean : 138305.687546 Total : 1675850016
31 renderDepth Count : 882 Min : 9568 Max : 39904 Mean : 12002.975057 Total : 10586624
32 renderTrack Count : 882 Min : 16608 Max : 33088 Mean : 18716.081633 Total : 16507584
33 renderVolume Count : 221 Min : 479840 Max : 3143200 Mean : 1338055.819005 Total : 295710336
34 track Count : 12117 Min : 13472 Max : 129344 Mean : 51505.231988 Total : 624088896
35 vertex2normal Count : 2646 Min : 8544 Max : 73504 Mean : 22275.362056 Total : 58940608
```


Textual User Interface

Easy
run

Raw
data

Sum
up

High-level blocks statistics

ses,
g



```
hickory% make 2.openc1.log
LD_PRELOAD=./build/kfusion/thirdparty/liboclwrapper.so ./build/kfusion/kfusion-benchmark-openc1 -s 4.8 -p 0.34,0.5,0.24 -z 4 -c 2 -r 2 -k 481.2,480,320,240 -i living_room_traj2_1
encl.log 2> oclwrapper.2.openc1.log
End of file(garbage found).
cat oclwrapper.2.openc1.log |grep -E "[0-9]+ [0-9]+ [0-9]+" |cut -d" " -f1,4 > kernels.2.openc1.log
./kfusion/thirdparty/checkPos.py benchmark.2.openc1.log livingRoom2.gt.freiburg > resume.2.openc1.log
./kfusion/thirdparty/checkKernels.py kernels.2.openc1.log >> resume.2.openc1.log
hickory%
```

frame	acquisition	preprocessing	tracking	integration	raycasting	rendering	computation	total	X	Y	Z	tracked	inte
grated													
0	0.003564	0.000451	0.001622	0.000461	0.000002	0.001124	0.002536	0.007223	0.0000	0.000000	0.000000	0	1
1	0.002724	0.000352	0.000903	0.000467	0.000001	0.000212	0.001722	0.004658	0.0000	0.000000	0.000000	0	1
2	0.002697	0.000336	0.000904	0.000459	0.000001	0.000212	0.001701	0.004611	0.0000	0.000000	0.000000	0	1
3	0.002663	0.000335	0.000905	0.000467	0.000004	0.000219	0.002311	0.005193	0.0000	0.000000	0.000000	0	1

```
1 Get kfusion output data.
2 Skip kfusion line :
3 Skip nuim line :
4 kfusion file: Valid frames 882 dropped frames: 0
5 kfusion result      : 882 positions.
6 NUIM result        : 880 positions.
7 Working position is : 880
8 Untracked frames: 0
9 Shift kfusion trajectory...
10
11 A detailed statistical analysis is provided.
12 All durations are in seconds and the absolute trajectory error (ATE) is in centimeters.
13
14 ATE Min : 0.000000 Max : 0.049309 Mean : 0.020662 Total : 18.18239335
15 acquisition Min : 0.000056 Max : 0.009033 Mean : 0.002044 Total : 1.80289800
16 computation Min : 0.001701 Max : 0.009234 Mean : 0.005894 Total : 5.19872500
17 integration Min : 0.000001 Max : 0.000821 Mean : 0.000258 Total : 0.22773500
18 preprocessing Min : 0.000284 Max : 0.001884 Mean : 0.000441 Total : 0.38904600
19 raycasting Min : 0.000001 Max : 0.003313 Mean : 0.001345 Total : 1.18644400
20 rendering Min : 0.000201 Max : 0.003452 Mean : 0.000572 Total : 0.50420900
21 total Min : 0.003882 Max : 0.020054 Mean : 0.008510 Total : 7.50584700
22 tracking Min : 0.000903 Max : 0.006743 Mean : 0.003850 Total : 3.39543600
23 Get SlamBench data.
24 ResetVolume Count : 1 Min : 401056 Max : 401056 Mean : 401056.000000 Total : 401056
25 bilateral_filter Count : 882 Min : 139520 Max : 205632 Mean : 159880.199546 Total : 141014336
26 depth2vertex Count : 2646 Min : 5440 Max : 48384 Mean : 10585.167045 Total : 28008352
27 halfSampleRobust Count : 1764 Min : 6624 Max : 328480 Mean : 9546.557823 Total : 16840128
28 integrate Count : 443 Min : 270880 Max : 793664 Mean : 485648.469526 Total : 215142272
29 mm2meters Count : 882 Min : 6912 Max : 13632 Mean : 8977.306122 Total : 7917984
30 raycast Count : 879 Min : 435584 Max : 3284768 Mean : 1323028.423208 Total : 1162941984
31 reduce Count : 12117 Min : 31488 Max : 523488 Mean : 138305.687546 Total : 1675850016
32 renderDepth Count : 882 Min : 9568 Max : 39904 Mean : 12002.975057 Total : 10586624
33 renderTrack Count : 882 Min : 16608 Max : 33088 Mean : 18716.081633 Total : 16507584
34 renderVolume Count : 221 Min : 479840 Max : 3143200 Mean : 1338055.819005 Total : 295710336
35 track Count : 12117 Min : 13472 Max : 129344 Mean : 51505.231988 Total : 624088896
36 vertex2normal Count : 2646 Min : 8544 Max : 73504 Mean : 22275.362056 Total : 58940608
```


Textual User Interface

Easy
run

Raw
data

Sum
up

High-level blocks statistics

Kernels statistics

```
hickory% make 2.openc1.log
LD_PRELOAD=./build/kfusion/thirdparty/liboclwrapper.so ./build/kfusion/kfusion-benchmark-openc1 -s 4.8 -p 0.34,0.5,0.24 -z 4 -c 2 -r 2 -k 481.2,480,320,240 -i living_room_traj2_
encl.log 2> oclwrapper.2.openc1.log
End of file(garbage found).
cat oclwrapper.2.openc1.log |grep -E "[0-9]+ [0-9]+ [0-9]+" |cut -d" " -f1,4 > kernels.2.openc1.log
./kfusion/thirdparty/checkPos.py benchmark.2.openc1.log livingRoom2.gt.freiburg > resume.2.openc1.log
./kfusion/thirdparty/checkKernels.py kernels.2.openc1.log >> resume.2.openc1.log
hickory%
```

frame	acquisition	preprocessing	tracking	integration	raycasting	rendering	computation	total	X	Y	Z	tracked	inte
grated													
0	0.003564	0.000451	0.001622	0.000461	0.000002	0.001124	0.002536	0.007223	0.0000	0.000000	0.000000	0	1
1	0.002724	0.000352	0.000903	0.000467	0.000001	0.000212	0.001722	0.004658	0.0000	0.000000	0.000000	0	1
2	0.002697	0.000336	0.000904	0.000459	0.000001	0.000212	0.001701	0.004611	0.0000	0.000000	0.000000	0	1
3	0.002663	0.000335	0.000905	0.000467	0.000004	0.000219	0.002311	0.005193	0.0000	0.000000	0.000000	0	1

```
1 Get kfusion output data.
2 Skip kfusion line :
3 Skip nuim line :
4 kfusion file: Valid frames 882 dropped frames: 0
5 kfusion result      : 882 positions.
6 NUIM result        : 880 positions.
7 Working position is : 880
8 Untracked frames: 0
9 Shift kfusion trajectory...
```

```
10
11 A detailed statistical analysis is provided.
12 All durations are in seconds and the absolute trajectory error (ATE) is in centimeters.
13
14 ATE Min : 0.000000 Max : 0.049309 Mean : 0.020662 Total : 18.18239335
15 acquisition Min : 0.000056 Max : 0.009033 Mean : 0.002044 Total : 1.80289800
16 computation Min : 0.001701 Max : 0.009234 Mean : 0.005894 Total : 5.19872500
17 integration Min : 0.000001 Max : 0.000821 Mean : 0.000258 Total : 0.22773500
18 preprocessing Min : 0.000284 Max : 0.001884 Mean : 0.000441 Total : 0.38904600
19 raycasting Min : 0.000001 Max : 0.003313 Mean : 0.001345 Total : 1.18644400
20 rendering Min : 0.000201 Max : 0.003452 Mean : 0.000572 Total : 0.50420900
21 total Min : 0.003882 Max : 0.020054 Mean : 0.008510 Total : 7.50584700
22 tracking Min : 0.000903 Max : 0.006743 Mean : 0.003850 Total : 3.39543600
```

```
23 ResetVolume Count : 1 Min : 401056 Max : 401056 Mean : 401056.000000 Total : 401056
24 bilateral_filter Count : 882 Min : 139520 Max : 205632 Mean : 159880.199546 Total : 141014336
25 depth2vertex Count : 2646 Min : 5440 Max : 48384 Mean : 10585.167045 Total : 28008352
26 halfSampleRobust Count : 1764 Min : 6624 Max : 328480 Mean : 9546.557823 Total : 16840128
27 integrate Count : 443 Min : 270880 Max : 793664 Mean : 485648.469526 Total : 215142272
28 mm2meters Count : 882 Min : 6912 Max : 13632 Mean : 8977.306122 Total : 7917984
29 raycast Count : 879 Min : 435584 Max : 3284768 Mean : 1323028.423208 Total : 1162941984
30 reduce Count : 12117 Min : 31488 Max : 523488 Mean : 138305.687546 Total : 1675850016
31 renderDepth Count : 882 Min : 9568 Max : 39904 Mean : 12002.975057 Total : 10586624
32 renderTrack Count : 882 Min : 16608 Max : 33088 Mean : 18716.081633 Total : 16507584
33 renderVolume Count : 221 Min : 479840 Max : 3143200 Mean : 1338055.819005 Total : 295710336
34 track Count : 12117 Min : 13472 Max : 129344 Mean : 51505.231988 Total : 624088896
35 vertex2normal Count : 2646 Min : 8544 Max : 73504 Mean : 22275.362056 Total : 58940608
```



Conclusions

- **First vision benchmark** for performance, energy and accuracy



Conclusions

- **First vision benchmark** for performance, energy and accuracy
- **Fair comparison** of accelerators, tools and novel algorithms in SLAM



Conclusions

- **First vision benchmark** for performance, energy and accuracy
- **Fair comparison** of accelerators, tools and novel algorithms in SLAM
- "Performance" results on state-of-the-art desktop, laptop and embedded:
 - 4 configurations super real-time FPS (**135 FPS** on TITAN)
 - Tegra K1 achieves **22 FPS**
 - GPGPU for SLAM leads to high-efficiency
 - ODROID-XU3 achieves **5.5 FPS** for **2.1 Watts**



Conclusions

- **First vision benchmark** for performance, energy and accuracy
- **Fair comparison** of accelerators, tools and novel algorithms in SLAM
- "Performance" results on state-of-the-art desktop, laptop and embedded:
 - 4 configurations super real-time FPS (**135 FPS** on TITAN)
 - Tegra K1 achieves **22 FPS**
 - GPGPU for SLAM leads to high-efficiency
 - ODROID-XU3 achieves **5.5 FPS** for **2.1 Watts**
- This research paves the way for **systematic holistic evaluation**



Copyrights

- Author: unknown. Microsoft Kinect camera. [Image]. Retrieved from <http://channel9.msdn.com/Series/KinectSDKQuickstarts/Understanding-Kinect-Hardware>
- Author: Dyson Ltd. Dyson 360 Eye. [Video]. Retrieved from <https://www.youtube.com/watch?v=OadhulCDAjk>
- Author: Google Inc. Google Tango project. [Image]. Retrieved from <http://blogthinkbig.com/en/project-tango-googles-mobile-kinect/>
- Author: unknown. Audi autonomous car. [Photograph]. Retrieved from <http://www.wired.com/2010/06/audis-robotic-car-looks-hot-in-old-school-livery/>
- Author: ExtremeTech. Google Shaft robot. [Photograph]. Retrieved from <http://www.extremetech.com/extreme/173318-google-wins-darpa-robotics-challenge-wonders-if-it-was-a-good-idea-to-turn-down-future-military-contracts>
- Author: HardKernel. ODROID-XU3 board. [Photograph]. Retrieved from http://www.hardkernel.com/main/products/prdt_info.php?g_code=G135235611947
- Author: PC Specialist Ltd. Vortex series laptop. [Photograph]. Retrieved from <https://www.pcspecialist.co.uk/forums/showthread.php?23366-My-new-beast-15-6-quot-Vortex-III>
- Author: Arndale.org. Arndale board. [Photograph]. Retrieved from http://www.arndaleboard.org/wiki/index.php/Main_Page
- Author: Unknown. Chip. [Image]. Retrieved from <https://cajalesygalileos.wordpress.com/2013/06/23/un-chip-ultrasensible-identifica-15-cepas-de-gripe/>
- Author: Unknown. Eye. [Image]. Retrieved from <http://gallery.digitalculture.asu.edu/?/interactive-environments/computer-vision/>
- Author: Unknown. Compiler. [Image]. Retrieved from <http://d3q6qq2zt8nhwv.cloudfront.net/107/large-icon.png>

