Vertically-integrated exploration of algorithmic and implementation design spaces in 3D scene understanding

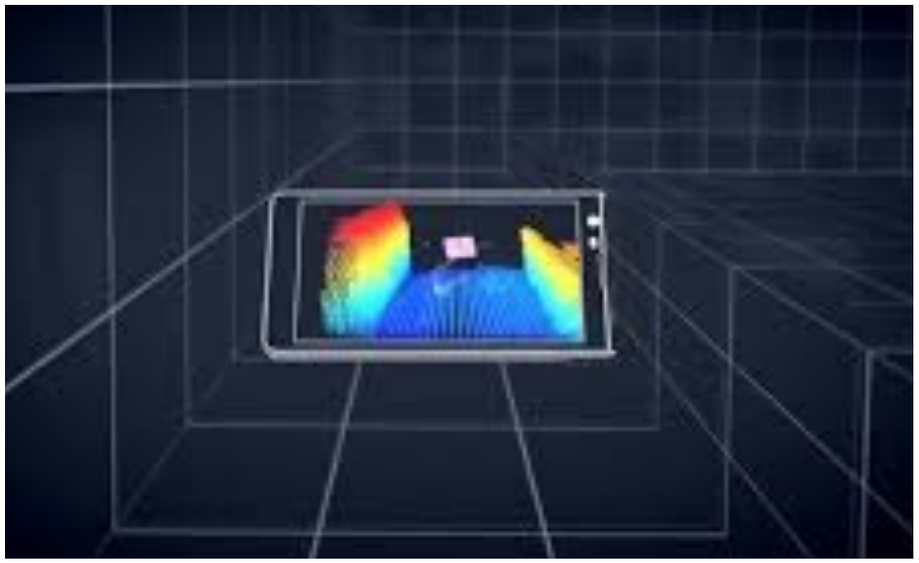
Luigi Nardi, PhD
Software Performance Optimisation group
@IFIP 2.11 Imperial College London
November 11th 2015

In collaboration with:
B. Bodin, M Z. Zia, J. Mawer, E. Vespa, M. K. Emani, A. Nisbet, G. S. Shenoy, M. F. P. O'Boyle, P. H. J. Kelly, B. Franke, C. Kotselidis, M. Luján, A. J. Davison, G. Riley, N. Topham and S. Furber

Luigi Nardi - Imperial College London

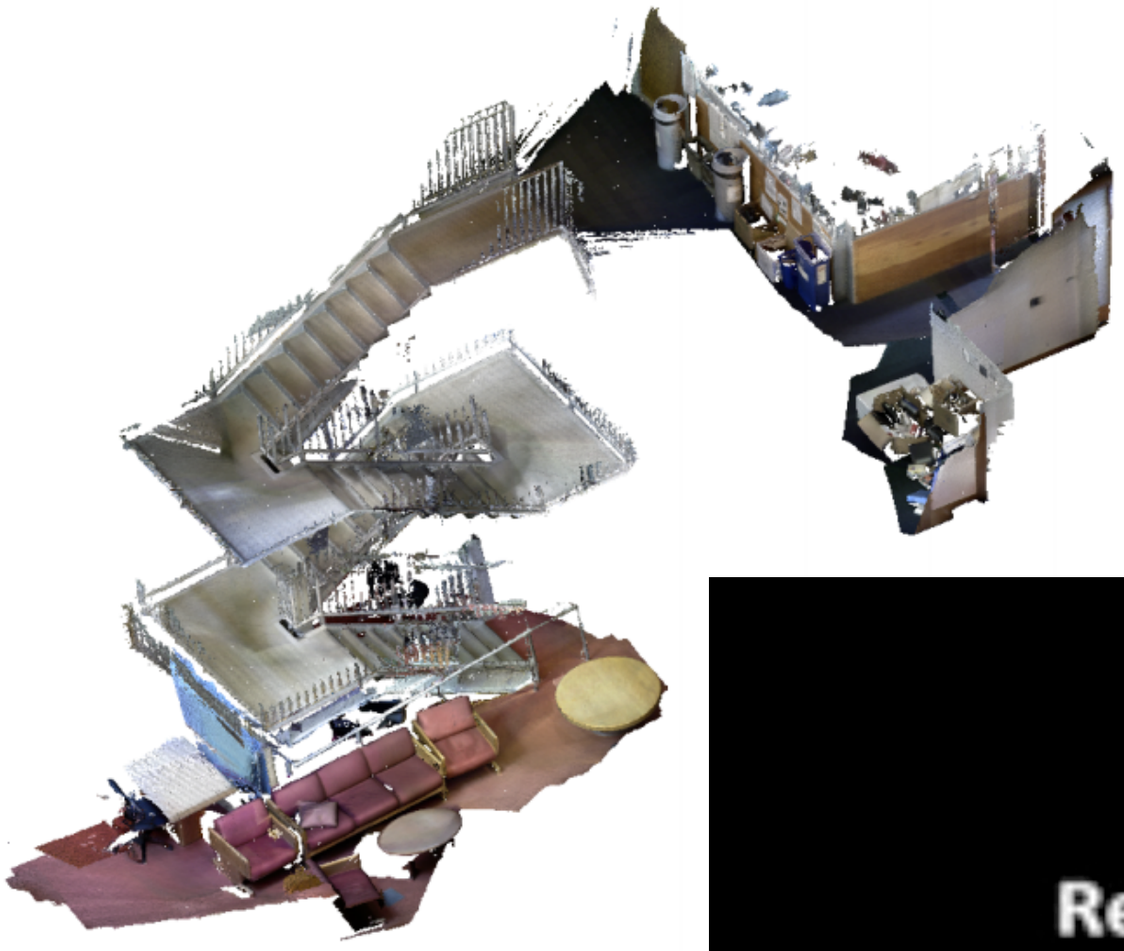
Simultaneous localisation and mapping (SLAM)

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

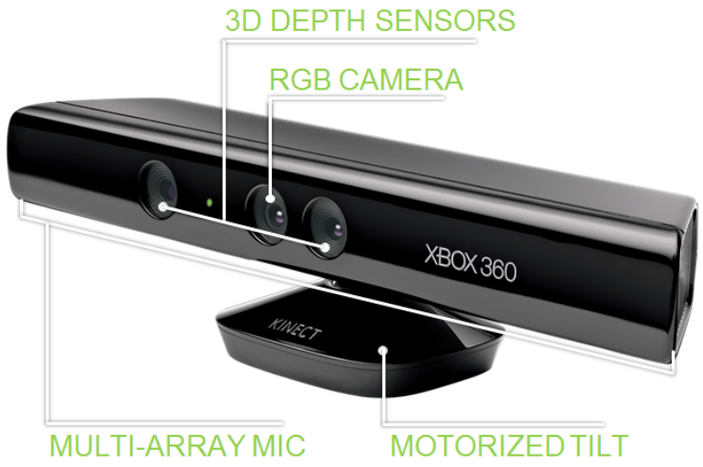


Video:
Dyson 360 Eye

Simultaneous localisation and mapping (SLAM)



[Whelan et al. 2012]



Video:
[Newcombe et al. ISMAR 2011]

Three “Performance” metrics

Holistic approach to SLAM “performance”:

SLAMBench

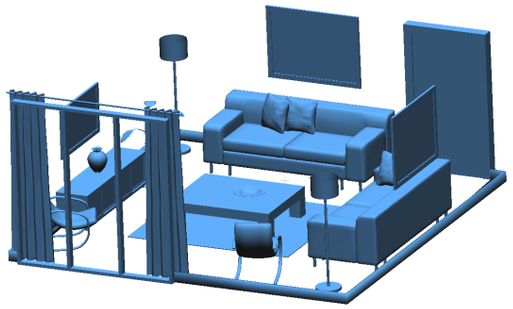
Computer Vision

Compiler/Runtime

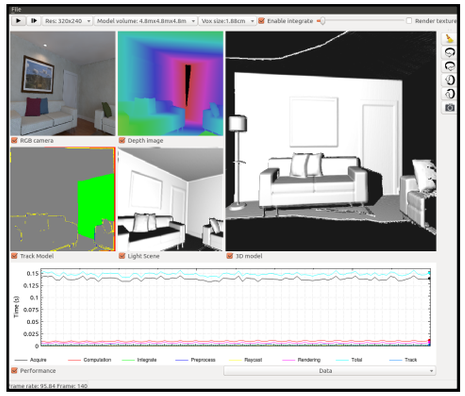
Kernels → **Architectures** → **Correctness** → **Performance** → **Metrics**

```
bilateralFilter (..)
halfSampleRobust (..)
renderVolume (..)
integrate (..)
      :
      :
```

Hardware



ICL-NUIM Dataset

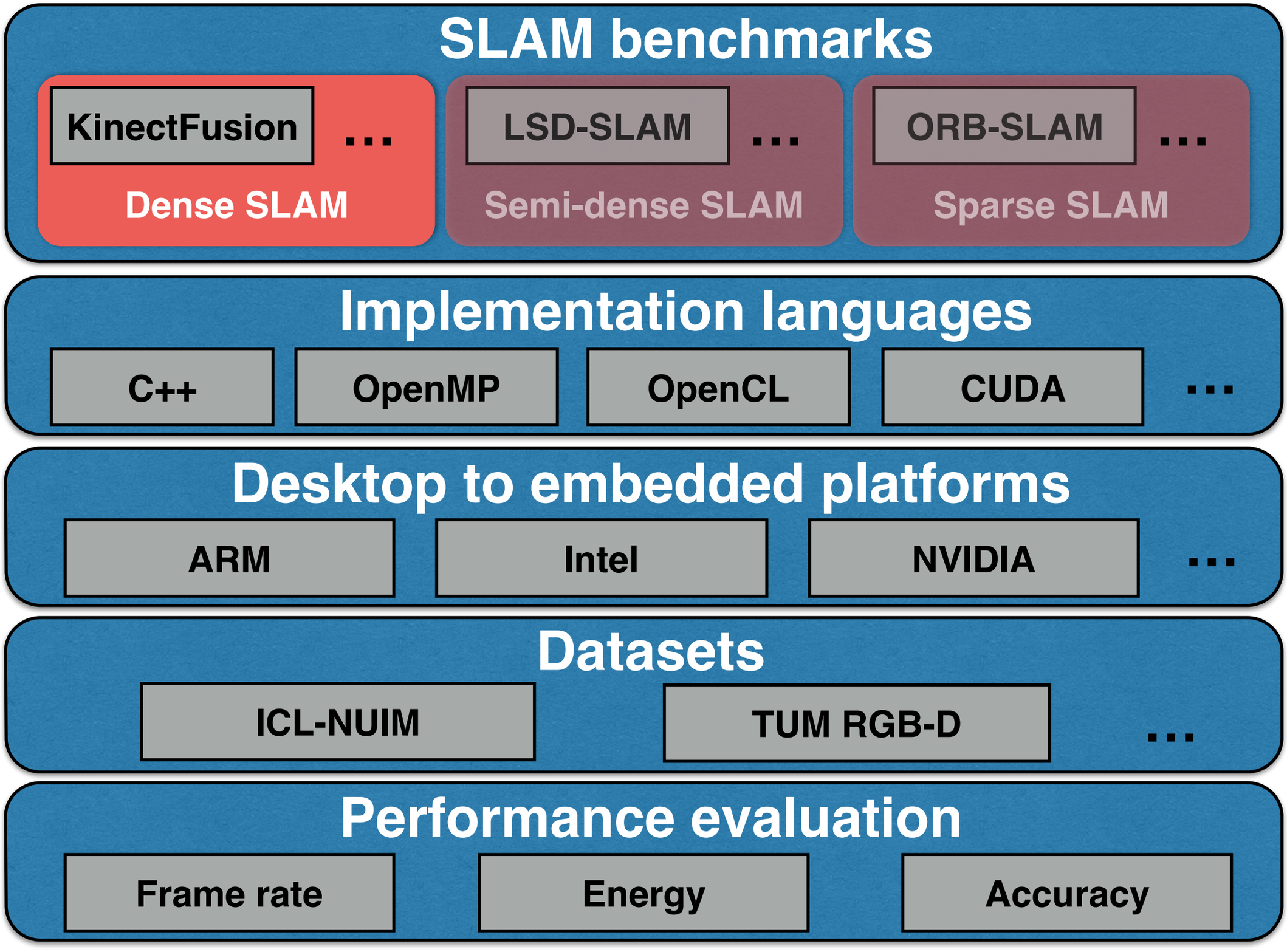


Frame rate

Energy

Accuracy

SLAMBench framework



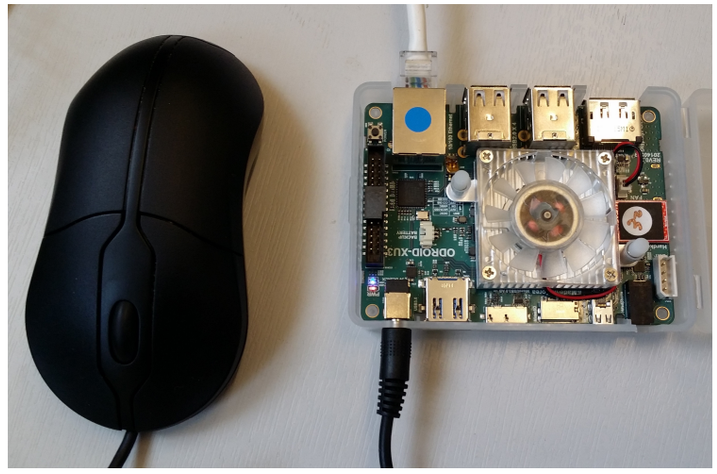
[Nardi et al., 2015]

“Performance” on SLAMBench

- Runtime/energy/accuracy measurements
- Accuracy provided via absolute trajectory error (ATE)

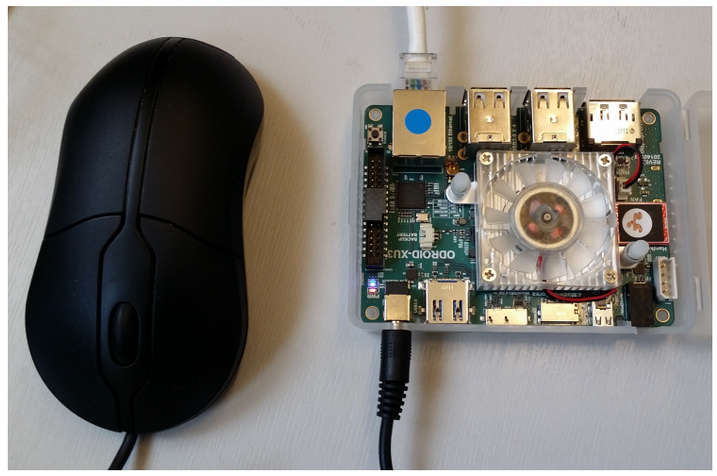
“Performance” on SLAMBench

- Runtime/energy/accuracy measurements
- Accuracy provided via absolute trajectory error (ATE)



Machine	CPU	CPU name	CPU GFLOPS	CPU cores	GPU	GPU name	GPU GFLOPS	TDP Watts
Hardkernel ODROID-XU3	ARM A15 + A7	Exynos 5422	80	4 + 4	ARM	Mali-T628	60 + 30	10

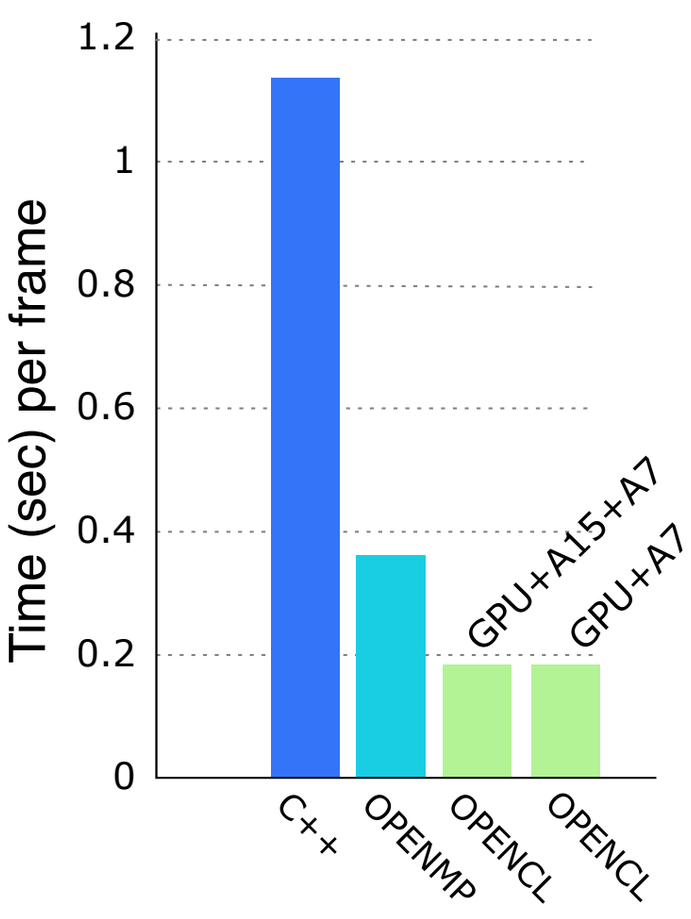
“Performance” on SLAMBench

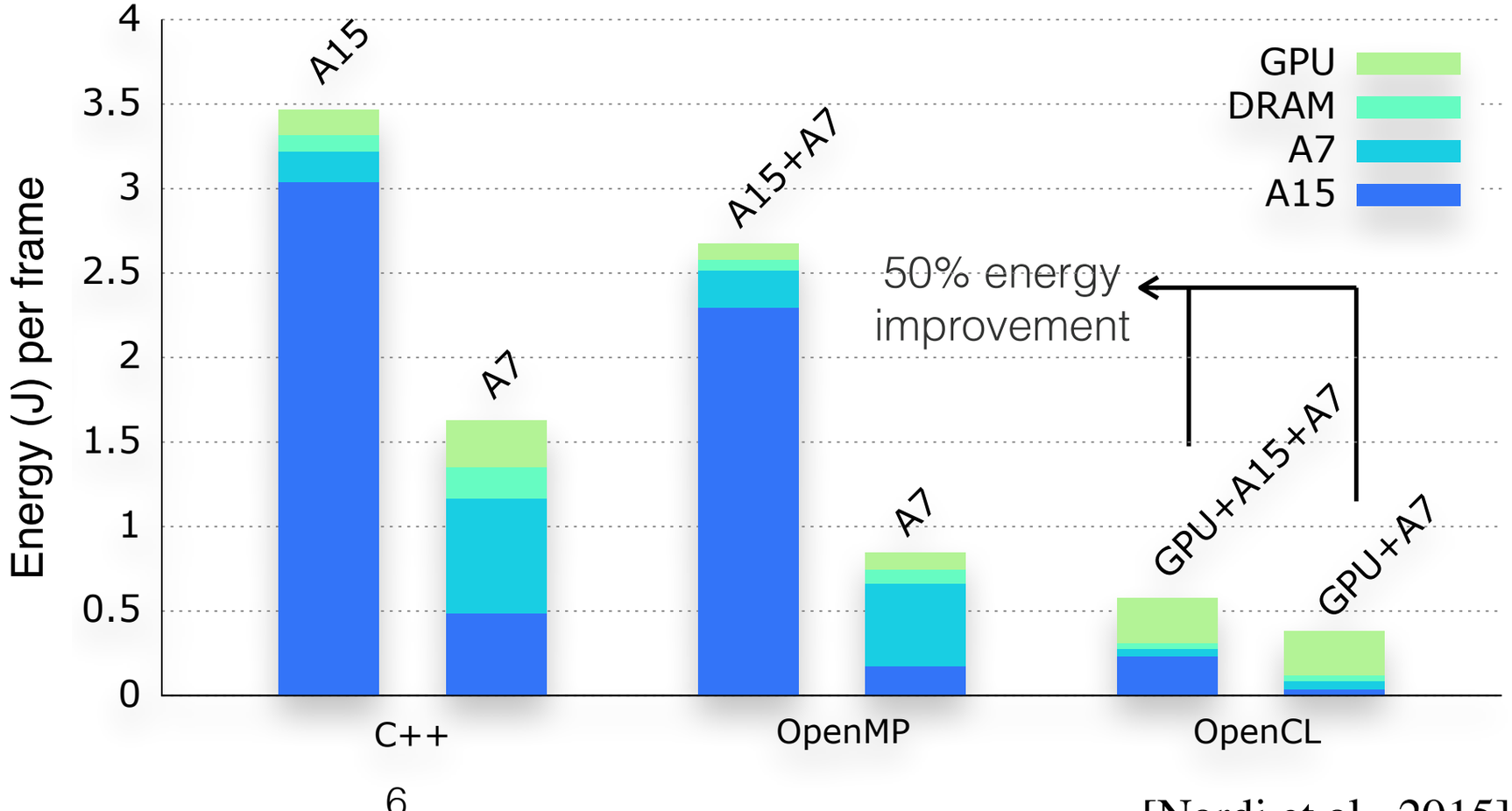


- Runtime/energy/accuracy measurements
- Accuracy provided via absolute trajectory error (ATE)

ATE in cm	
C++	2.06
OpenMP	2.06
OpenCL	2.01

Machine	CPU	CPU name	CPU GFLOPS	CPU cores	GPU	GPU name	GPU GFLOPS	TDP Watts
Hardkernel ODROID-XU3	ARM A15 + A7	Exynos 5422	80	4 + 4	ARM	Mali-T628	60 + 30	10





[Nardi et al., 2015]

SLAMBench today

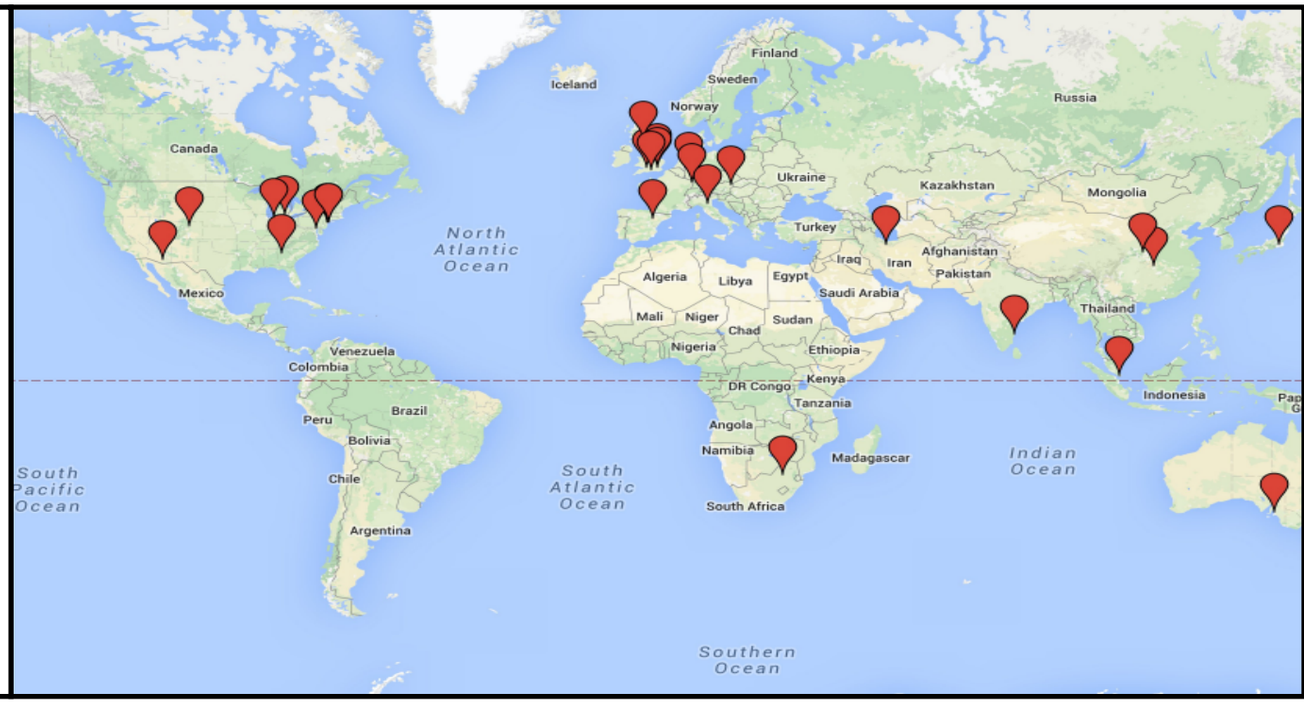
- Publicly released 13/11/2014 (800+ downloads)

- Early adopters:
  - Computer Vision
  - Compiler/runtime
  - Architecture

Company users

AMD, ocado, dyson, ARM, Texas Instruments, Microsoft, amazon, NVIDIA, QUALCOMM, Imagination, IBM, Google, codeplay, Movidius, SAMSUNG, intel, TOYOTA, atheer labs, magic leap

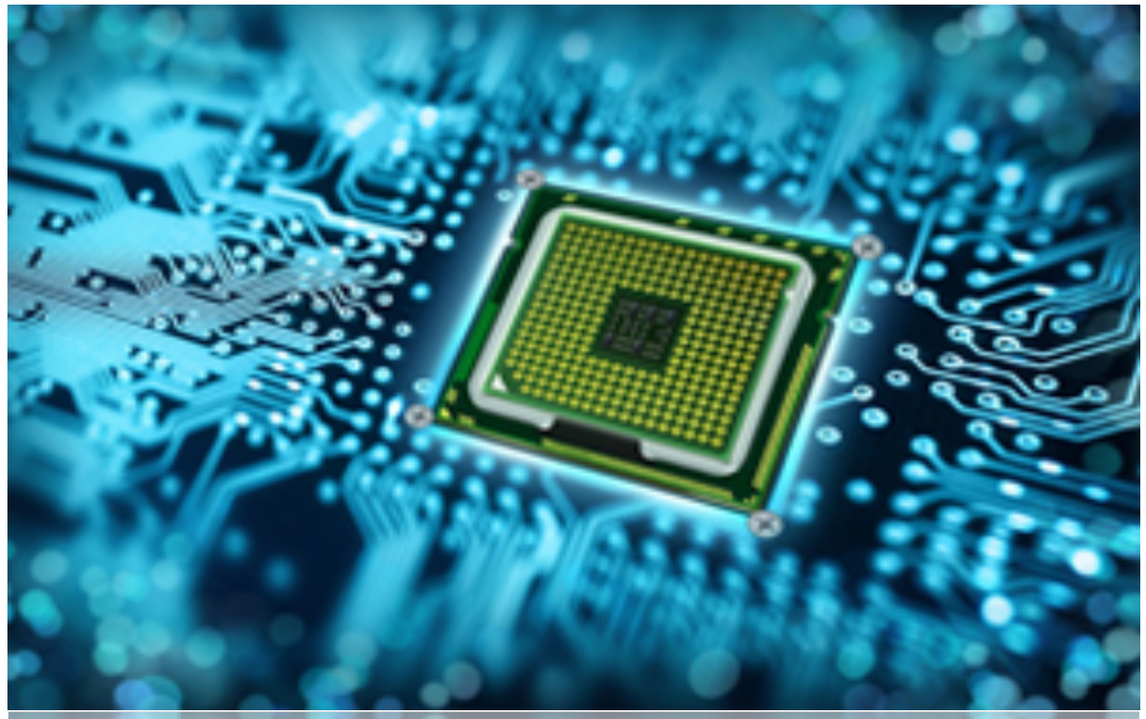
Academic users



Web: apt.cs.manchester.ac.uk/projects/PAMELA/tools/SLAMBench/

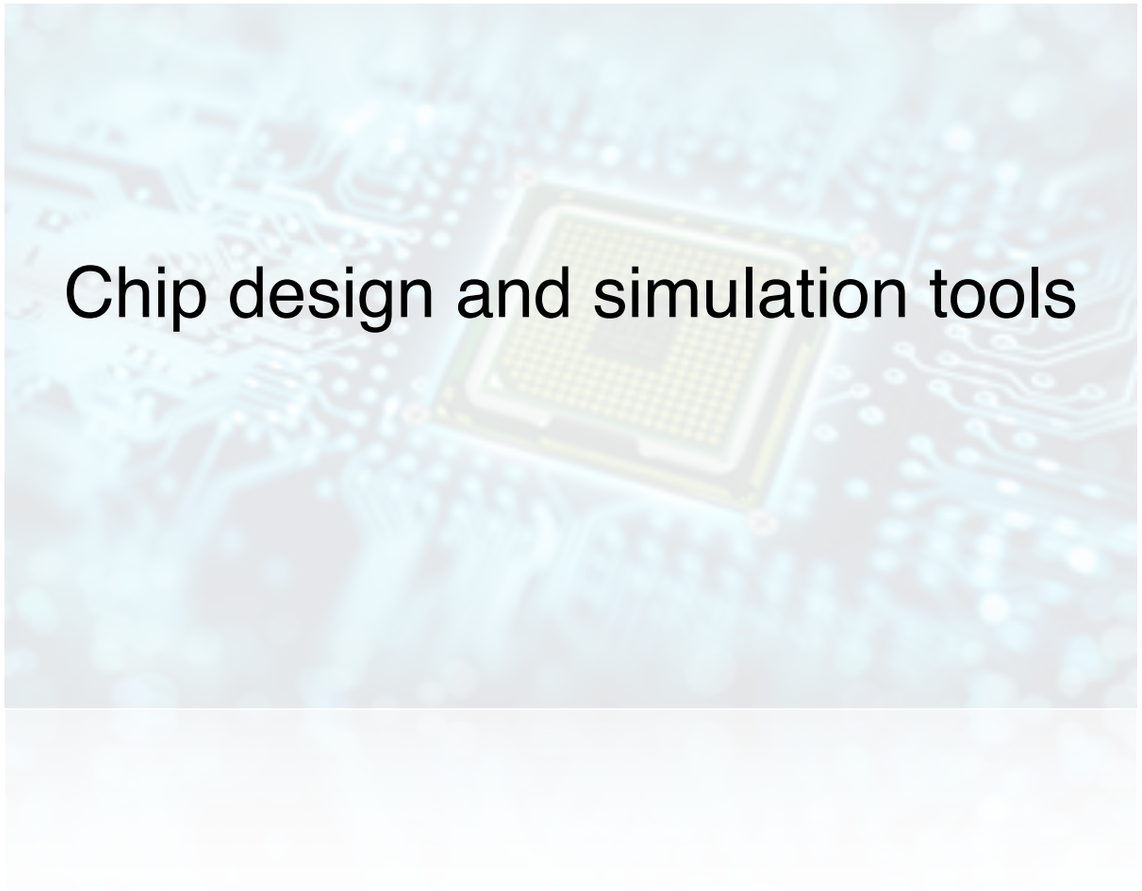
Paper: ***Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM"*, arxiv.org/abs/1410.2167**

SLAMBench opportunities



Chip design and simulation tools

SLAMBench opportunities



SLAMBench opportunities

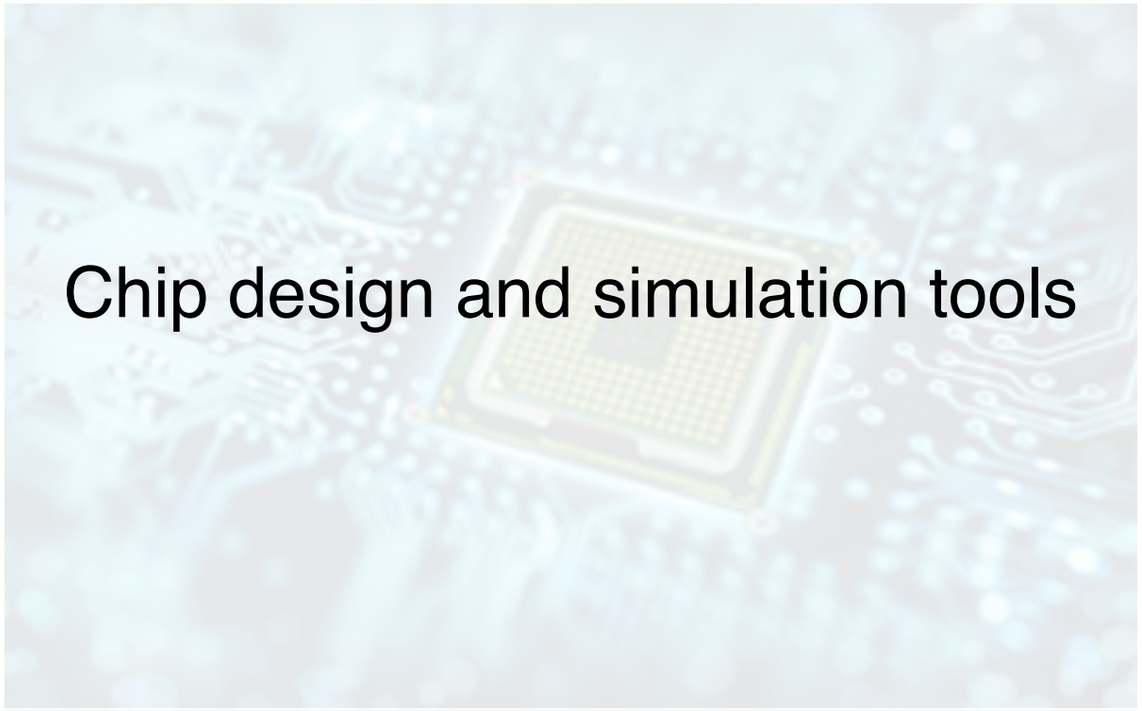
Chip design and simulation tools

SLAMBench evolution:

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

Kernels can be improved individually

SLAMBench opportunities



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

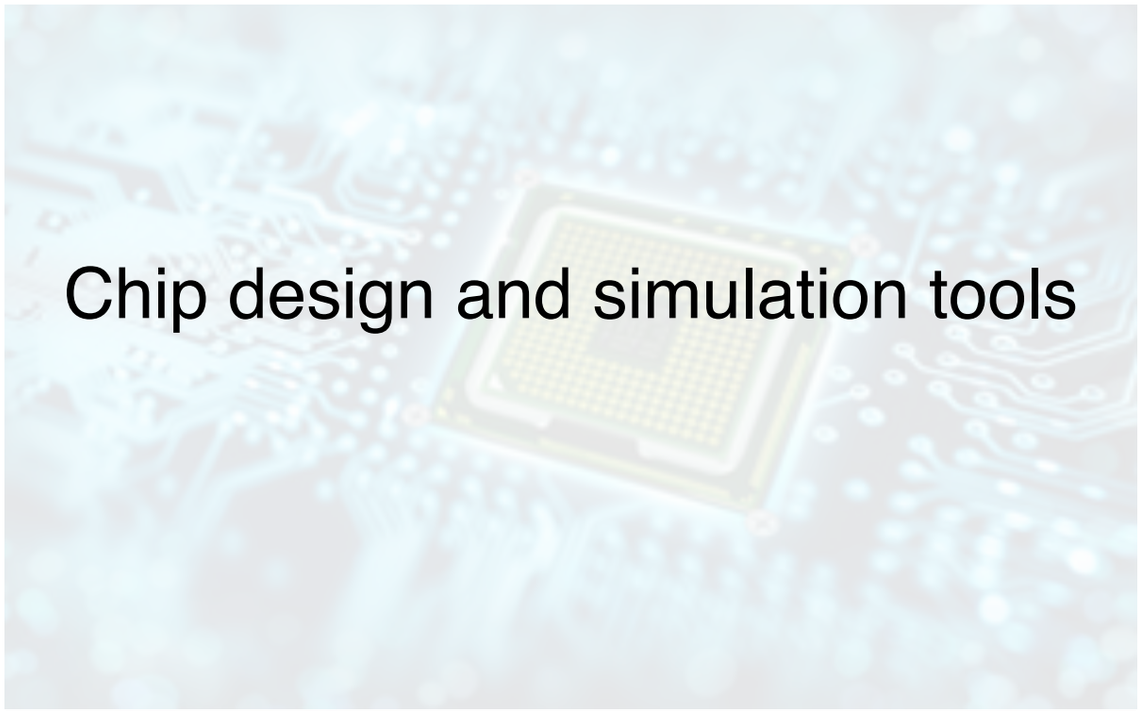
Kernels can be improved individually

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



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

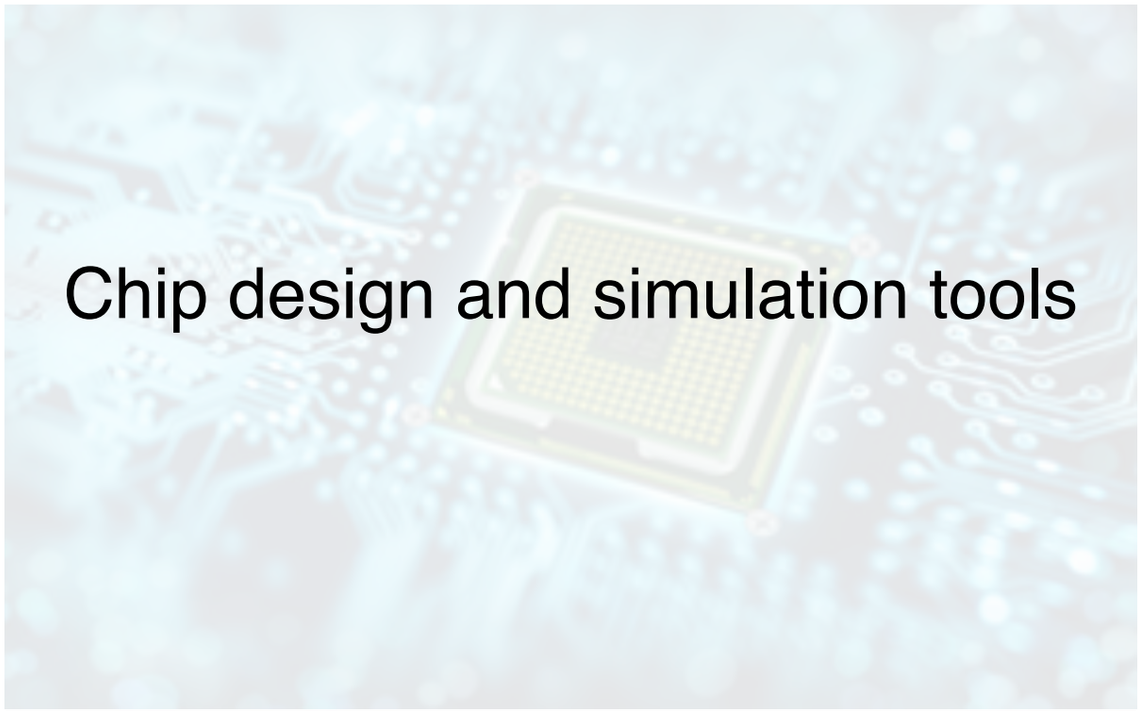
SLAMBench kernels tuning, e.g. vectorisation, GPU, auto-tuning

SLAMBench evolution:

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

Kernels can be improved individually

SLAMBench opportunities



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

What is the optimisation space?

Configuration parameters:

Space 1

1. Algorithmic:
   - Application-specific parameters
   - Minimisation methods
   - Early exit condition values

What is the optimisation space?

Configuration parameters:

Space 1

1. Algorithmic:
   - Application-specific parameters
   - Minimisation methods
   - Early exit condition values

Space 2

2. Compilation:
   - opencl-params: -cl-mad-enable,-cl-fast-relaxed-math, etc.
   - LLVM flags: O1, O2, O3, vectorize-slp-aggressive, etc.
   - Local work group size: 16/32/64/96/112/128/256
   - Vectorisation: width (1/2/4/8), direction (x/y)
   - Thread coarsening: factor (1/2/4/8/16/32), stride (1/2/4/8/16/32), dimension (x/y)

What is the optimisation space?

Configuration parameters:

Space 1

1. Algorithmic:
   - Application-specific parameters
   - Minimisation methods
   - Early exit condition values

Space 2

2. Compilation:
   - opencl-params: -cl-mad-enable,-cl-fast-relaxed-math, etc.
   - LLVM flags: O1, O2, O3, vectorize-slp-aggressive, etc.
   - Local work group size: 16/32/64/96/112/128/256
   - Vectorisation: width (1/2/4/8), direction (x/y)
   - Thread coarsening: factor (1/2/4/8/16/32), stride (1/2/4/8/16/32), dimension (x/y)

Space 3

3. Architecture:
   - GPU frequency: 177/266/350/420/480/543/600/DVFS
   - # of active big cores: 0/1/2/3/4
   - # of active LITTLE cores: 1/2/3/4

What is the optimisation space?

Configuration parameters:

Co-design space

Space 1

1. Algorithmic:
   - Application-specific parameters
   - Minimisation methods
   - Early exit condition values

Space 2

2. Compilation:
   - opencl-params: -cl-mad-enable,-cl-fast-relaxed-math, etc.
   - LLVM flags: O1, O2, O3, vectorize-slp-aggressive, etc.
   - Local work group size: 16/32/64/96/112/128/256
   - Vectorisation: width (1/2/4/8), direction (x/y)
   - Thread coarsening: factor (1/2/4/8/16/32), stride (1/2/4/8/16/32), dimension (x/y)

Space 3

3. Architecture:
   - GPU frequency: 177/266/350/420/480/543/600/DVFS
   - # of active big cores: 0/1/2/3/4
   - # of active LITTLE cores: 1/2/3/4

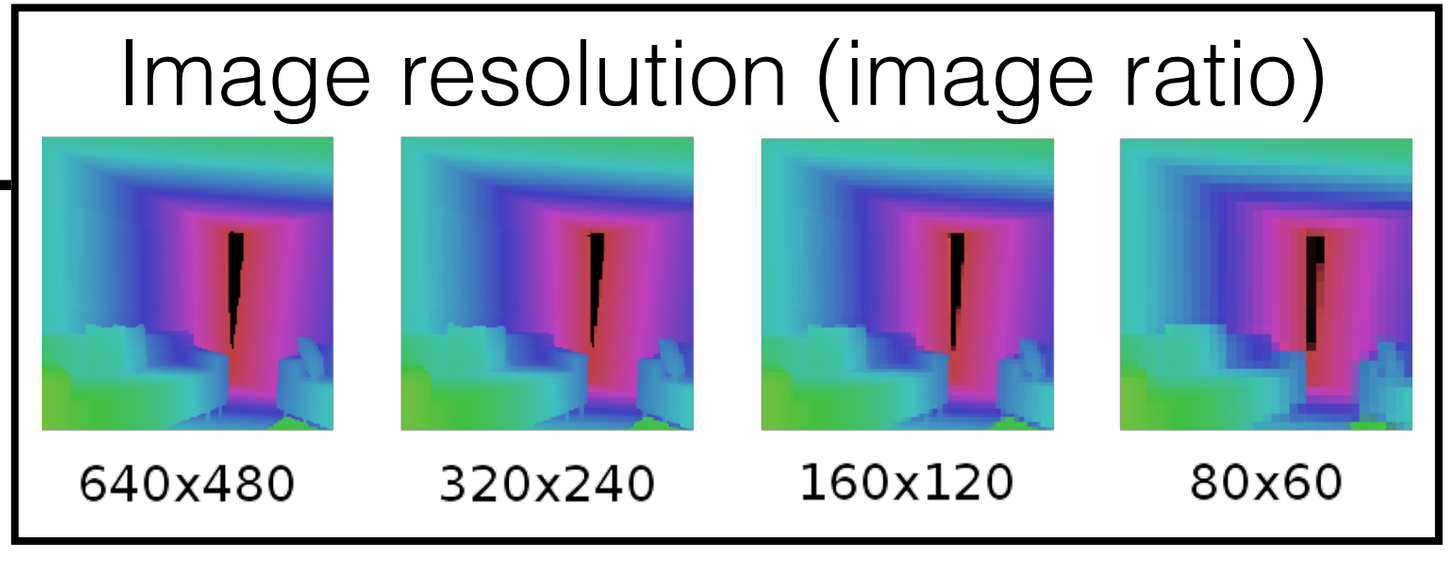
Warning: huge spaces, impossible to run exhaustively

KinectFusion algorithmic features

Features	Ranges
Volume resolution	64x64x64, 128x128x128, 256x256x256, 512x512x512
μ distance	0 .. 0.5
Pyramid level iterations (3 levels)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
Image resolution (image ratio)	1, 2, 4, 8
Tracking rate	1, 2, 3, 4, 5
ICP threshold	10^{-6} .. 10^{2}
Integration rate	1 .. 30

KinectFusion algorithmic features

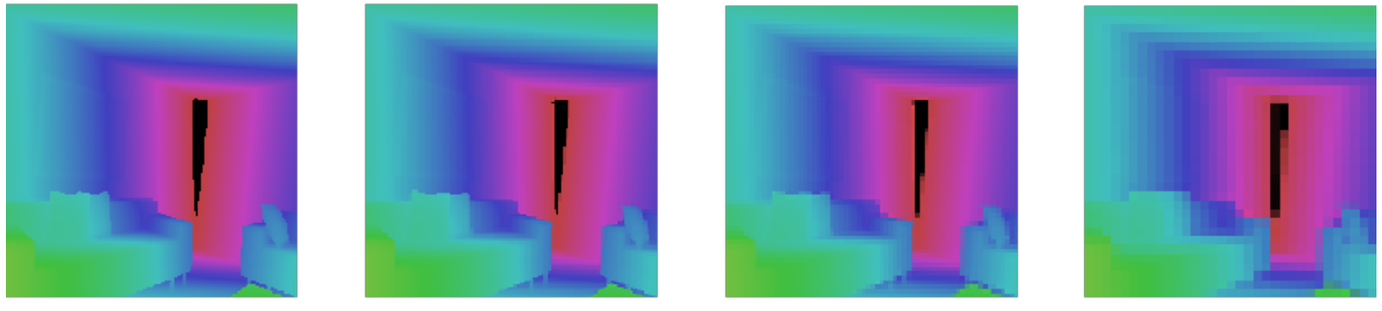
Features	Ranges
Volume resolution	64x64x64, 128x128x128, 256x256x256, 512x512x512
μ distance	0 .. 0.5
Pyramid level iterations (3 levels)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
Image resolution (image ratio)	1, 2, 4, 8
Tracking rate	1, 2, 3, 4, 5
ICP threshold	10^{-6} .. 10^{2}
Integration rate	1 .. 30



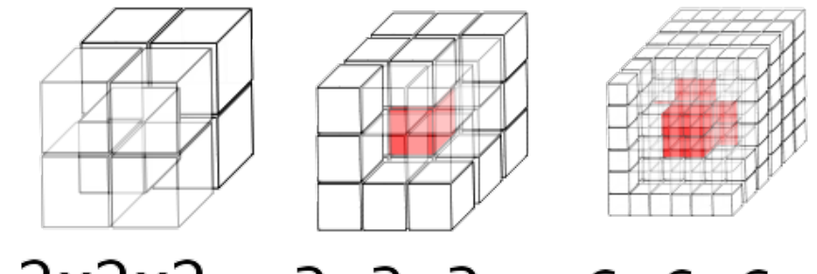
KinectFusion algorithmic features

Features	Ranges
Volume resolution	64x64x64, 128x128x128, 256x256x256, 512x512x512
μ distance	0 .. 0.5
Pyramid level iterations (3 levels)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
Image resolution (image ratio)	1, 2, 4, 8
Tracking rate	1, 2, 3, 4, 5
ICP threshold	10^{-6} .. 10^{2}
Integration rate	1 .. 30

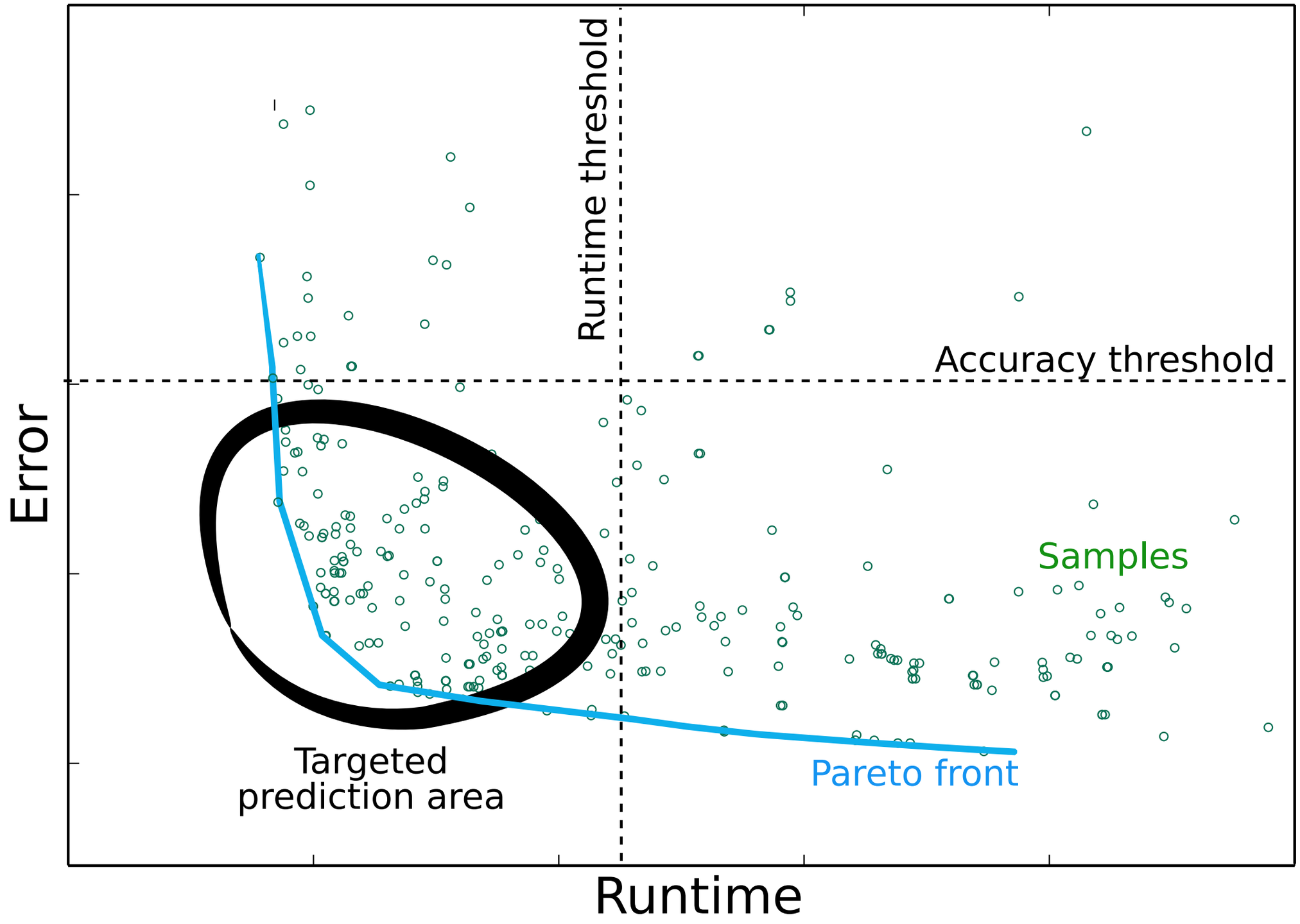
Image resolution (image ratio)



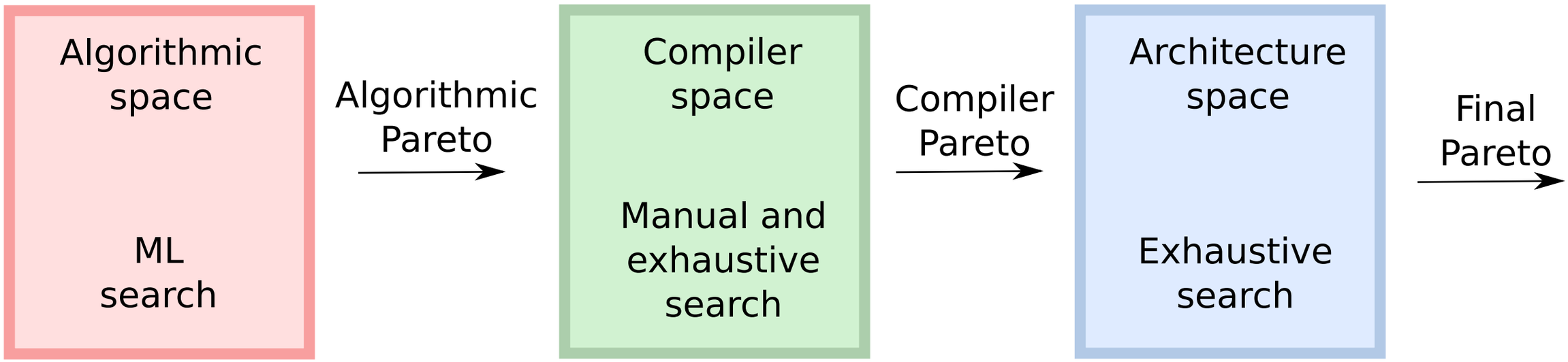
Volume resolution



Exploration goal



Incremental exploration approach

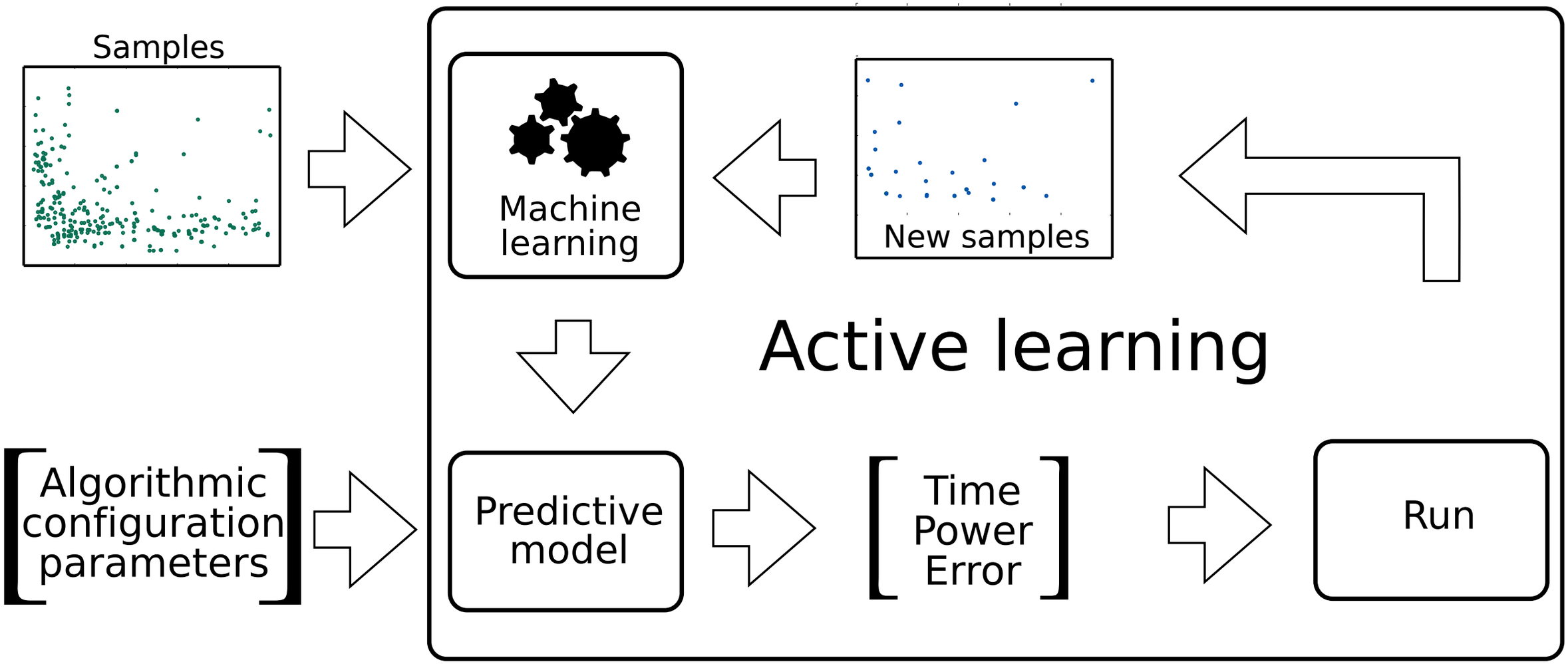


Frame rate
Power
Accuracy

Frame rate
Power
(Accuracy)

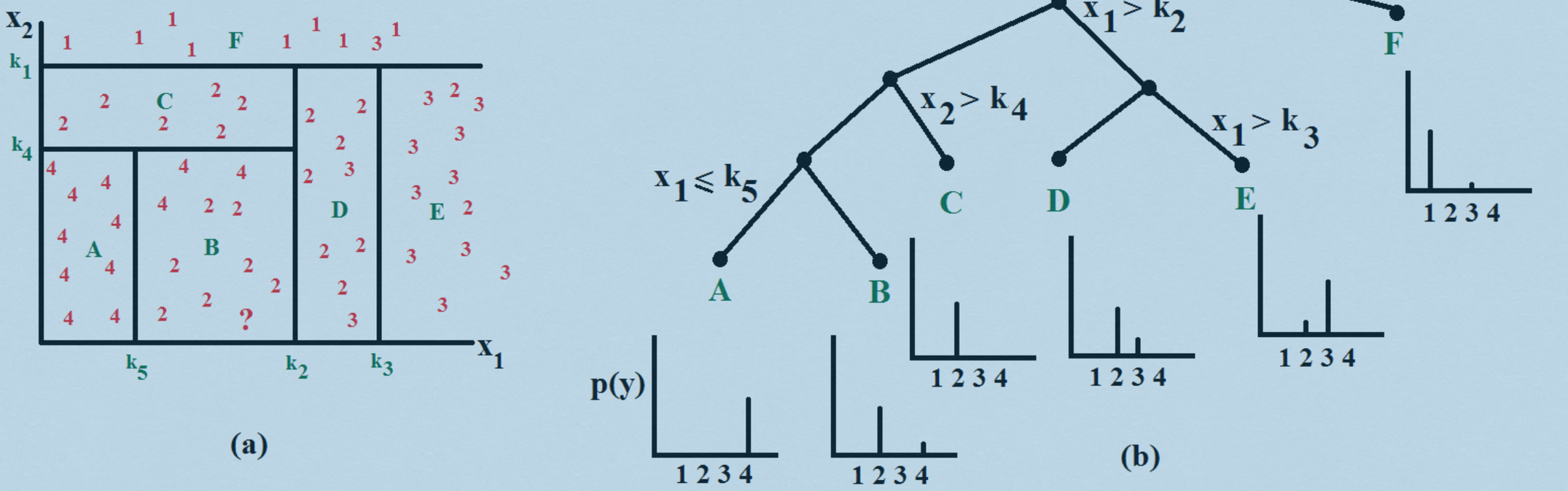
Frame rate
Power

Algo design-space exploration (DSE)

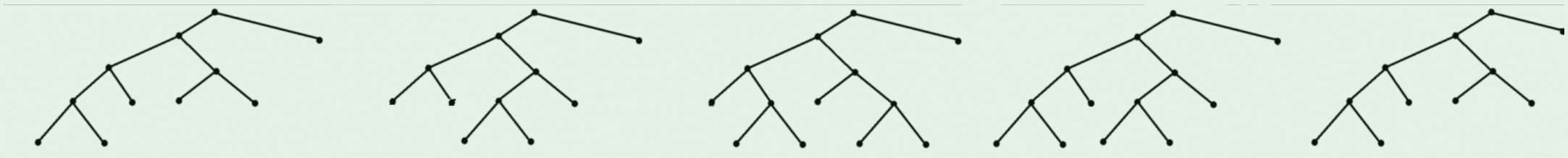


Machine learning methods used

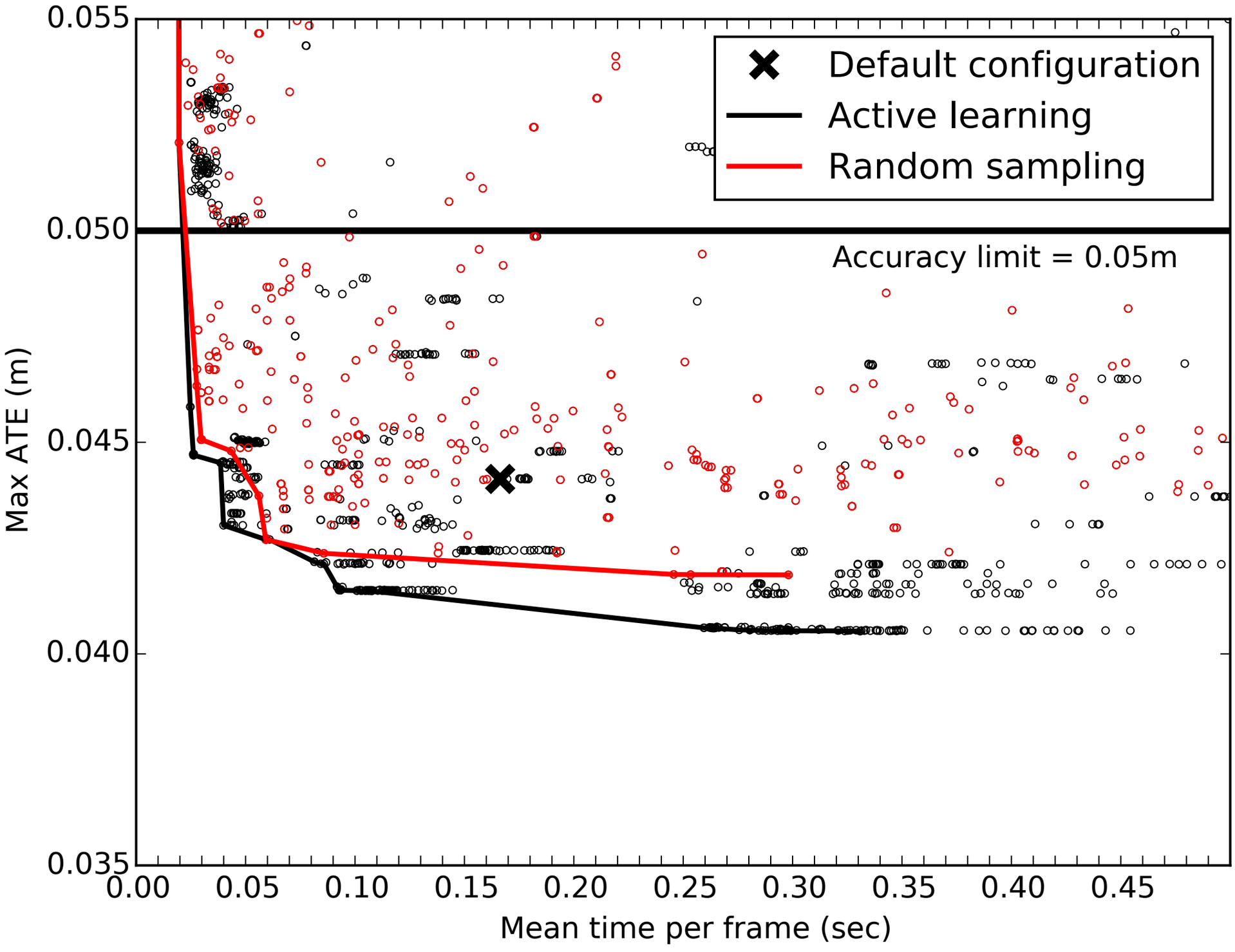
Decision tree



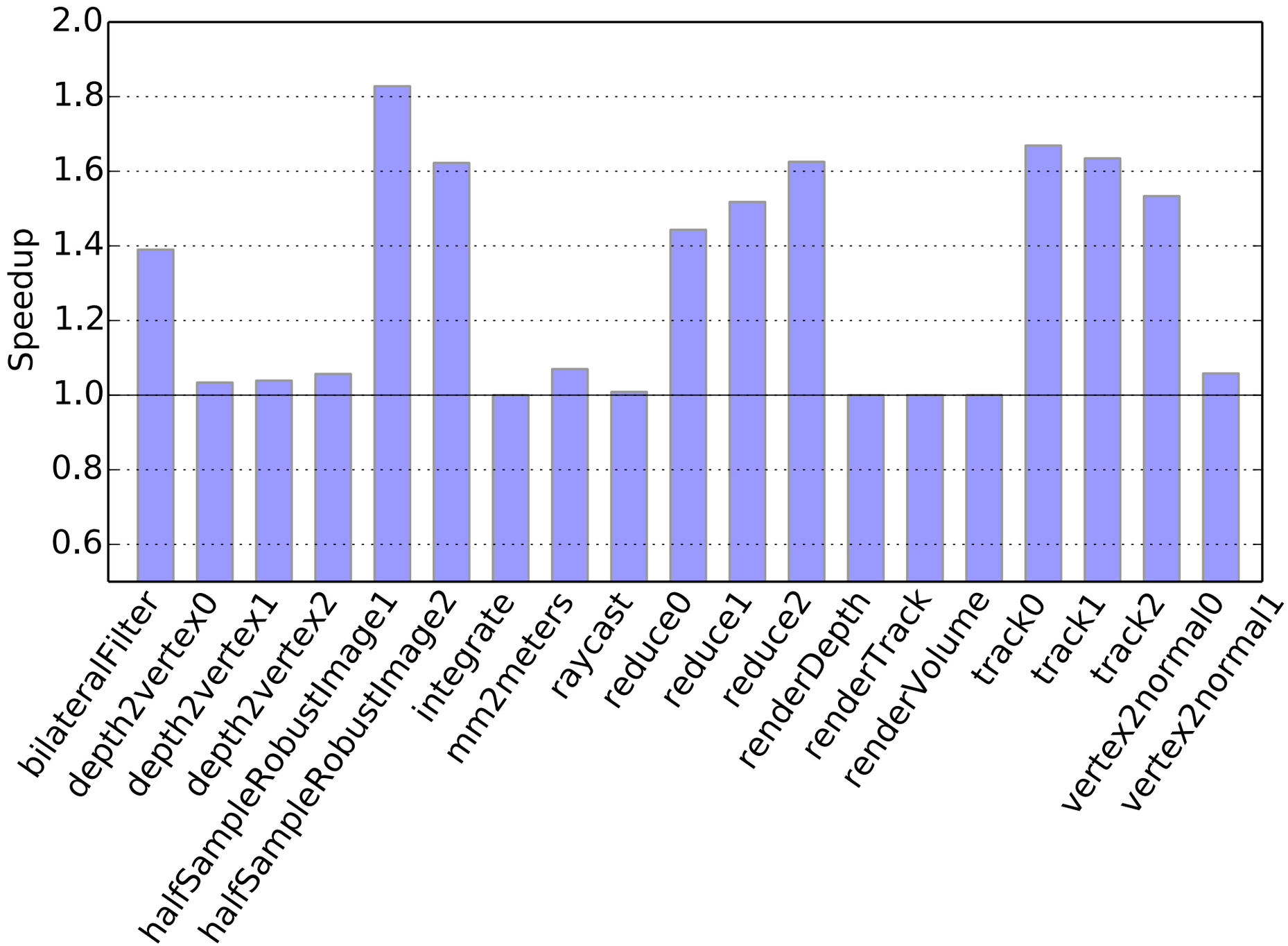
Random forest



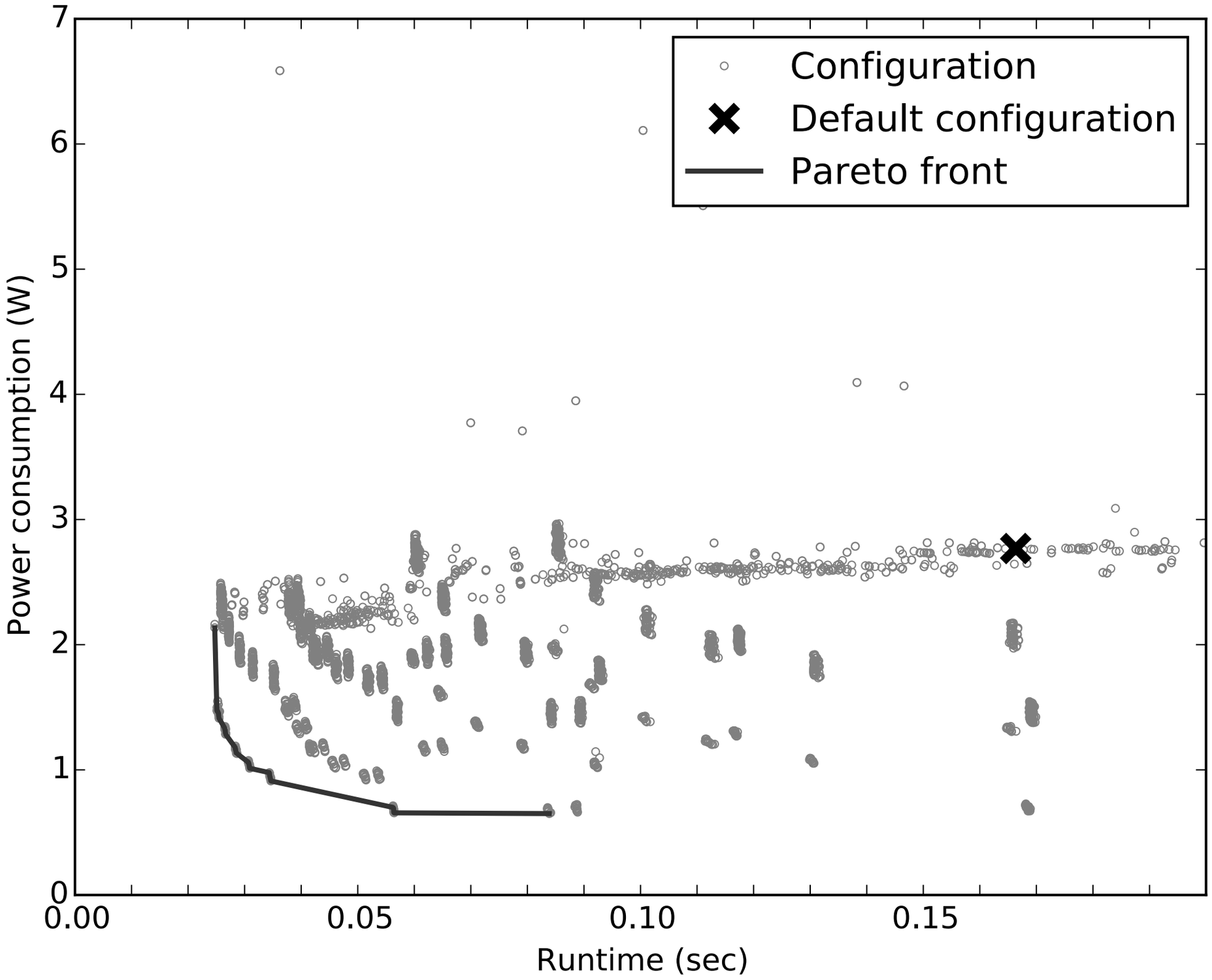
DSE on algorithmic parameters error/runtime



DSE compiler parameters speedup



DSE architecture parameters power/runtime

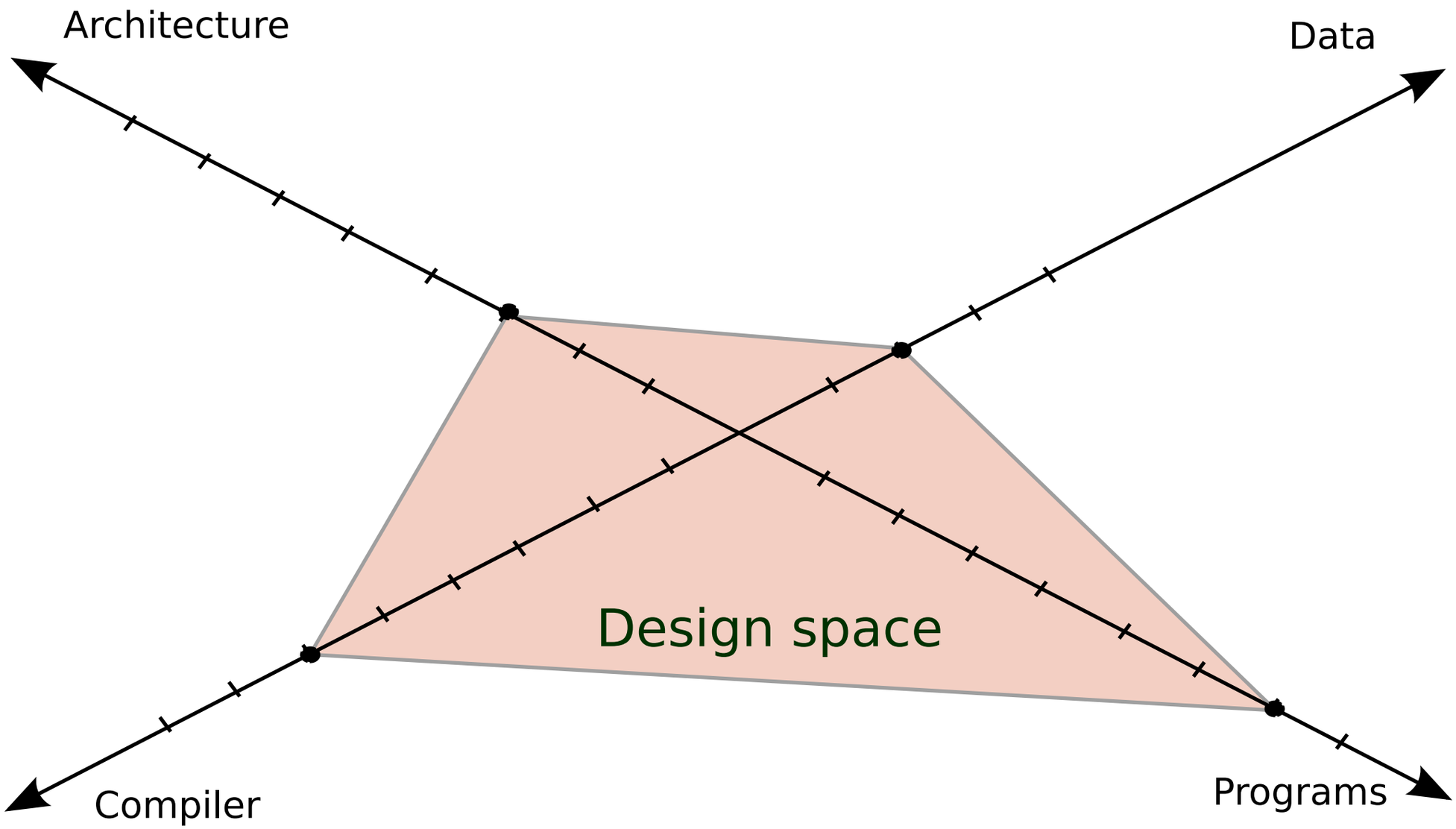


DSE final results and conclusion

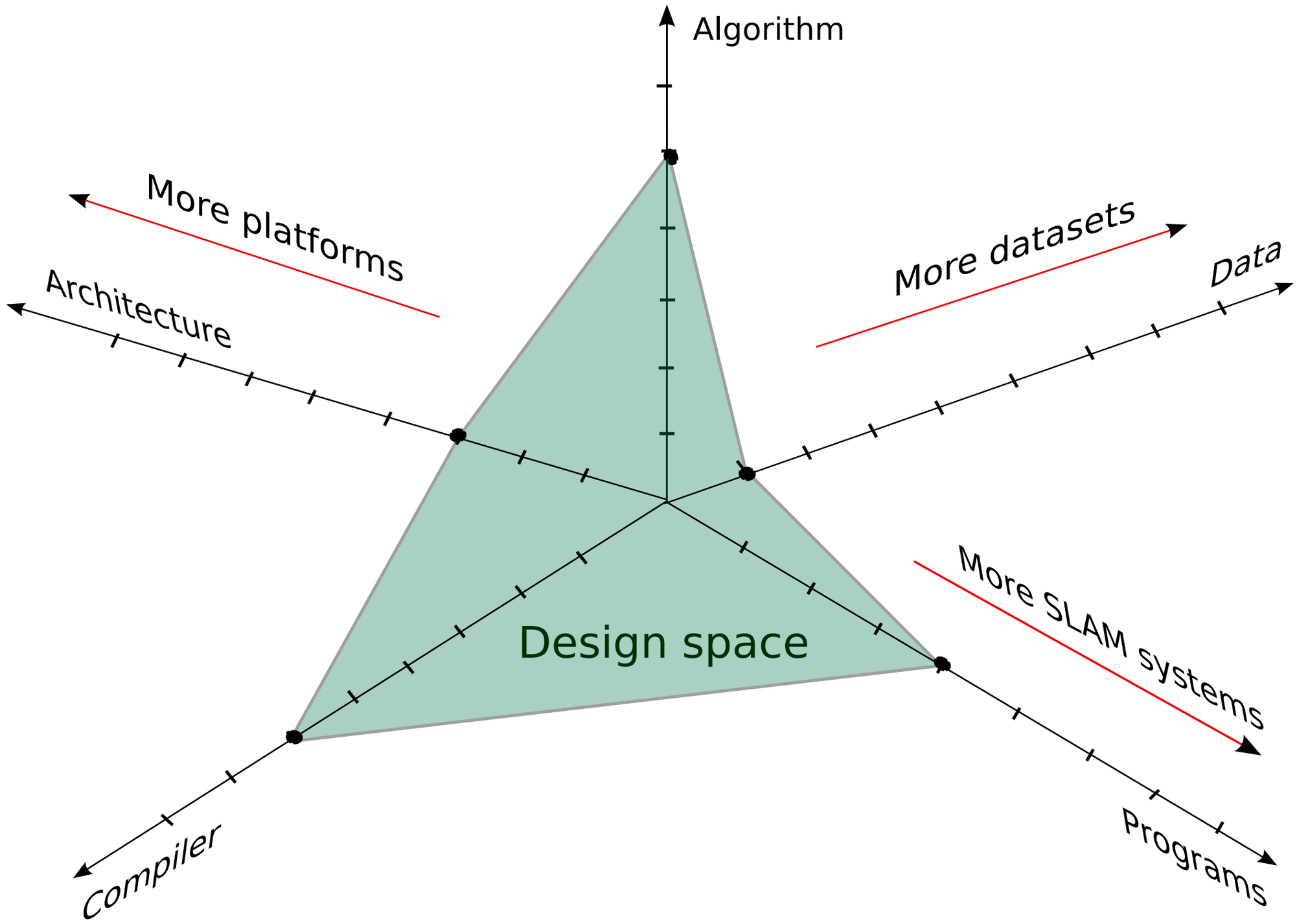
Constraint	Runtime (FPS)	Max ATE (cm)	Power (Watts)
Default	6.03	4.41	2.77
Best runtime	39.85	4.47	1.47
Best accuracy	1.51	3.30	2.38
Best power	11.92	4.45	0.65
Power $<$ 1W	29.09	4.47	0.98
Power $<$ 2W	39.85	4.47	1.47
FPS $>$ 10	11.92	4.45	0.65
FPS $>$ 20	28.87	4.47	0.91
FPS $>$ 30	32.38	4.47	1.01

- Multi-objective optimisation: frame rate/power/accuracy
- Auto-tuning tool to pick interesting points
- KinectFusion real-time on a popular embedded device
- Enabling auto-tuning at the domain-specific language (DSL) level

DSE the big picture I



DSE the big picture II



References I

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.

References II

- [Ogilvie 2014] Ogilvie, William, et al. "Fast automatic heuristic construction using active learning." Proceedings of the Workshop on Languages and Compilers for Parallel Computing (LCPC'14). 2014.
- [Siegmund 2015] Siegmund Norbert et al. "Performance-influence models for highly configurable systems", submitted FSE 2015.
- [Guo 2013] Guo, Jianmei, et al. "Variability-aware performance prediction: A statistical learning approach." Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on. IEEE, 2013.
- [Grewe 2011] Grewe, Dominik et al. "A static task partitioning approach for heterogeneous systems using OpenCL." Compiler Construction. Springer Berlin Heidelberg, 2011.
- [Kurek 2013] Kurek, Maciej, Tianchi Liu, and Wayne Luk. "MULTI-OBJECTIVE SELF-OPTIMIZATION OF RECONFIGURABLE DESIGNS WITH MACHINE LEARNING." 2nd Workshop on Self-Awareness in Reconfigurable Computing Systems (SRCS'13). 2013.
- [Balaprakash 2013] Balaprakash, Prasanna, Robert B. Gramacy, and Stefan M. Wild. "Active-learning-based surrogate models for empirical performance tuning." Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013.