Evaluation of Graphics-based General Purpose Computation Solutions for Safety Critical Systems: An Avionics Case Study

Marc Benito Bermúdez, Matina Maria Trompouki, Leonidas Kosmidis
Juan David Garcia, Sergio Carretero, Ken Wenger
{marc.benito, leonidas.Kosmidis}@bsc.es

Motivation

Embedded GPUs can provide the required performance

• Safety Critical Systems require higher performance to support new advanced functionalities

Characteristics of Safety Critical Systems:

- Certification: Need to comply with safety standards: ISO26262 / DO178
- Very conservative in terms of hardware and software: simple processors, mainly single core

• Massively parallel architectures, high computational power and high energy efficiency, in thermally limited systems

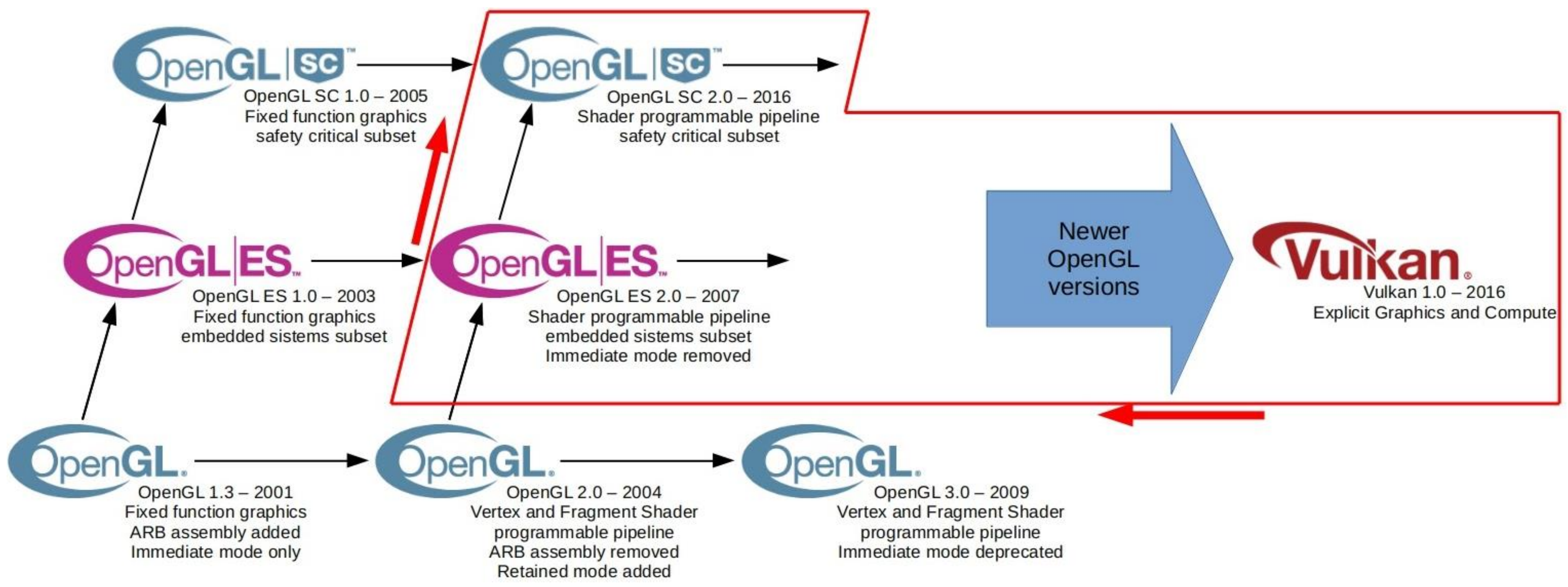
• OpenCL and CUDA dominate the market of GPGPU programming in HPC

- Easily programmable APIs
- Cannot be used in safety critical systems because of **pointers and dynamic memory allocation**

• In this Bachelor's thesis [1] awarded with a **Technology Transfer Award** we:
• Analyze their differences compared to desktop graphics APIs
• Demonstrate how a safety-critical application written in a non-certifiable programming model can be converted to use safety-critical APIs.
• Evaluate performance and programmability trade-offs

OpenGL and Vulkan versions diagram

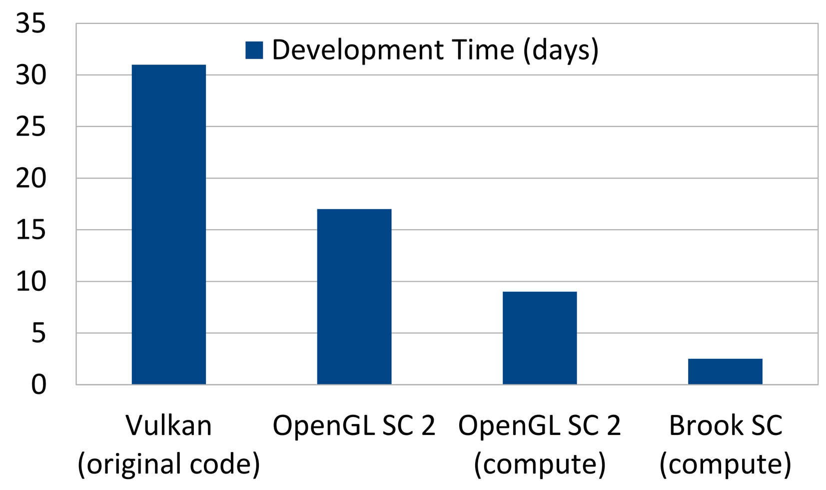
Initial prototype GPU-based avionics application, written in Vulkan and ported to OpenGL SC 2 following the guidelines of [2][3].

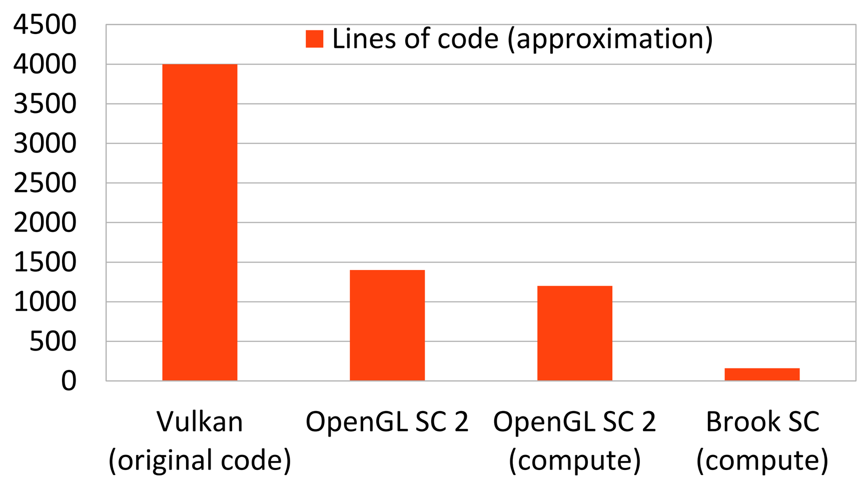


Brook SC Porting and Comparison

Brook SC [4][5][6] generates automatically OpenGL SC 2 code from a CUDA-like language. Comparison with the handwritten version:

• Porting completed in few days with no previous knowledge
• Very high productivity
• An order of magnitude reduction in the amount of code
• Negligible impact in performance

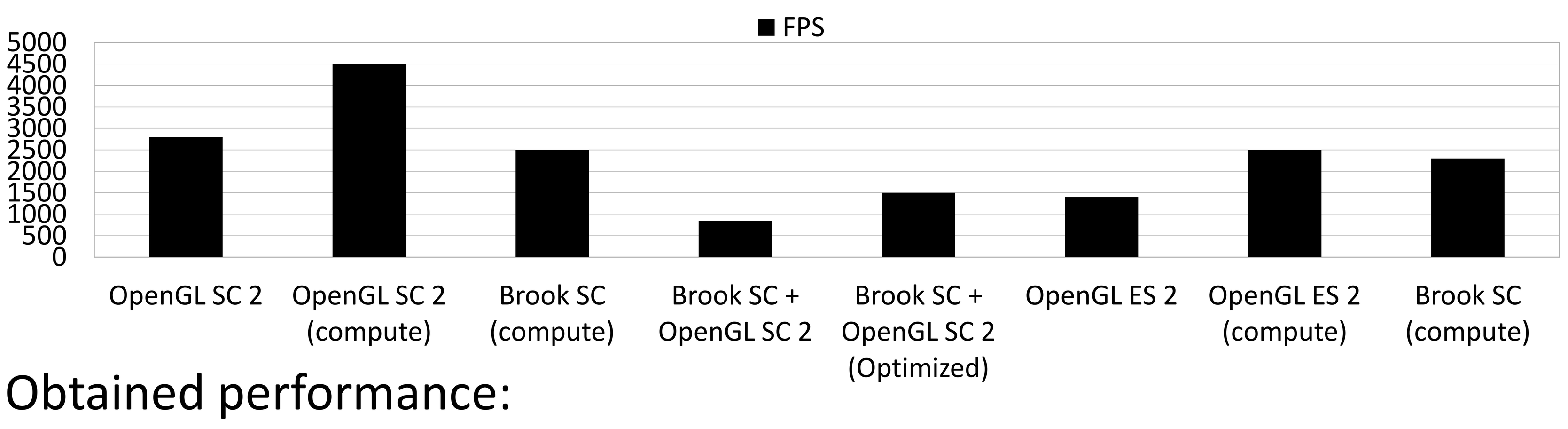




Programming language	Vulkan (original code)	OpenGL SC 2	OpenGL SC 2 (general-purpose compute)	Brook SC (general-purpose compute)
Development time (days)	31	17	9	2.5
Lines of code (approx)	4000	1400	1200	160

Performance Evaluation on an avionics grade AMD E8860 GPU

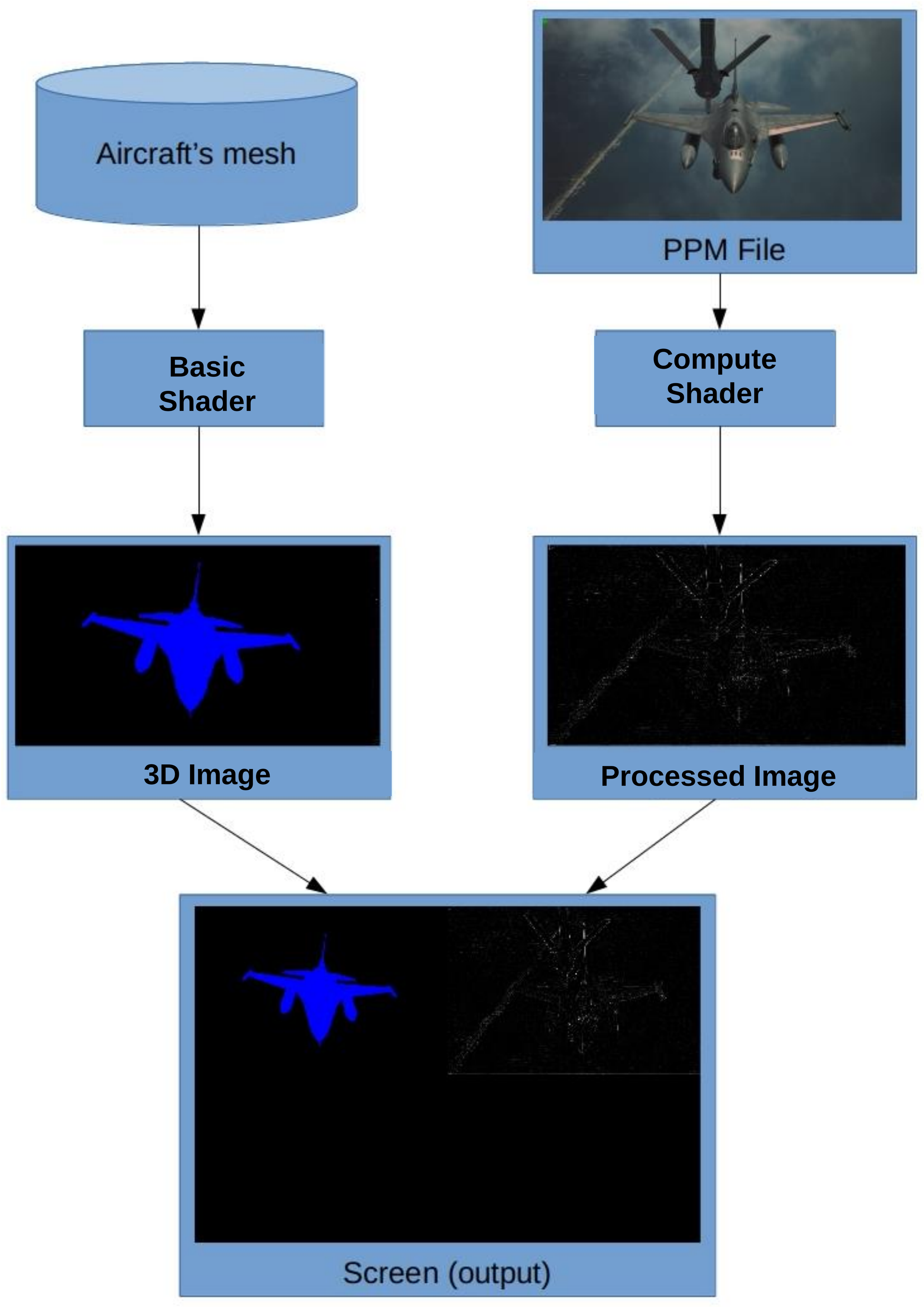
• CoreAVI OpenGL SC 2.0 driver, Open source AMD OpenGL ES 2.0 driver



Obtained performance:

• All variants meet the target refresh rate of the avionics display (60 FPS)
• Higher performance on OpenGL SC 2.0 driver than on OpenGL ES 2.0
• Brook SC compute performance is close to the Native OpenGL ES 2.0
• Texture sharing optimization between compute and graphics contexts doubles performance, by eliminating unnecessary texture copies
• Future Brook SC+OpenGL SC 2.0 optimization using FBO compositing is expected to increase performance further, by eliminating an extra fragment shader with respect to the OpenGL SC 2.0 implementation

Visual output of the Avionics Application



The display is divided in four regions

• The application uses both graphics and general purpose computations
• The **first region is the upper left zone** of the screen with a rotating 3D model of a plane. We load the mesh, then apply a basic shader and finally we draw it in a frame buffer.
• The **second region is the upper right zone** of the screen with a plane image obtained from a camera. The image is processed with general purpose computations and the result is written to the framebuffer.
• Finally, we draw **the framebuffer to the output screen**.

Acknowledgements

• Airbus Defence and Space Getafe (Madrid), Spain, provided the prototype GPU-based avionics application in Vulkan
• CoreAVI provided the certified OpenGL SC 2 driver and an avionics-grade AMD E8860 GPU

References

[1] Benito, M., Analysis and Evaluation of Embedded Graphics Solutions for Critical Systems, Bachelor's Thesis, Faculty of Informatics, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
[2] Trompouki et al, Towards General Purpose Computations on Low-End Mobile GPUs. DATE'16
[3] Trompouki et al, Optimisation Opportunities and Evaluation for GPGPU applications on Low-End Mobile GPUs. DATE'17
[4] Trompouki et al, Brook Auto: High-level Certification-friendly Programming for GPU powered automotive systems, DAC'18
[5] Trompouki et al, BRASIL: A High-Integrity Compiler for Automotive Systems. ICCD'19
[6] Kosmidis et. al, Brook SC, https://github.com/lkosmid/brook

This work has been partially supported by the HiPEAC Network of Excellence with a Technology Transfer Award 2019. Leonidas Kosmidis is also funded by the Spanish Ministry of Economy and Competitiveness (MINECO) under a Juan de la Cierva Formación postdoctoral fellowship (FJCI-2017-34095).