HPEC 2010
High Performance Embedded Computing

Hardware Acceleration of Electromagnetic Field Profile Computation: A Case Study Using the PO-SBR Method

Eric Dunn and **Nathan Smith**, Science Applications International Corporation
Ray Hoare, Concurrent EDA
Huan-Ting Meng and **Jianming Jin**, University of Illinois at Urbana-Champaign

SAIC
From Science to Solutions

About SAIC

www.saic.com

SAIC® From Science to Solutions

SAIC is a FORTUNE 500® scientific, engineering, and technology applications company that uses its deep domain knowledge to solve problems of vital importance to the nation and the world, in national security, energy and the environment, critical infrastructure, and health.

Our Core Values and Purpose

Our Successes

40 years of continuous growth

- $10.8 billion in annual revenues for fiscal year 2010
- FORTUNE 500 company – No. 215

Superb staff of qualified professionals

- Approximately 45,000 personnel worldwide
- 10,000 employees with advanced degrees
- 19,000 employees with security clearances

Work on initiatives of national importance

- Green energy
- Global health
- Cybersecurity

Leading provider of contracted R&D services

All figures are current as of April 2010.

FORTUNE 500 is a registered trademark of Time, Inc. in the U.S. and/or other countries.

SAIC Business Overview

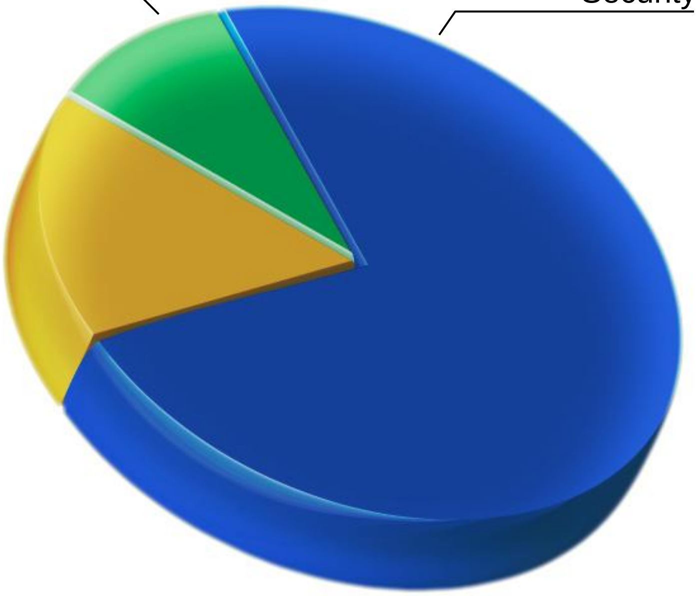
SAIC® From Science to Solutions

Business Areas

- Energy
- Environment
- National security
- Health
- Critical infrastructure

Competitive Strengths

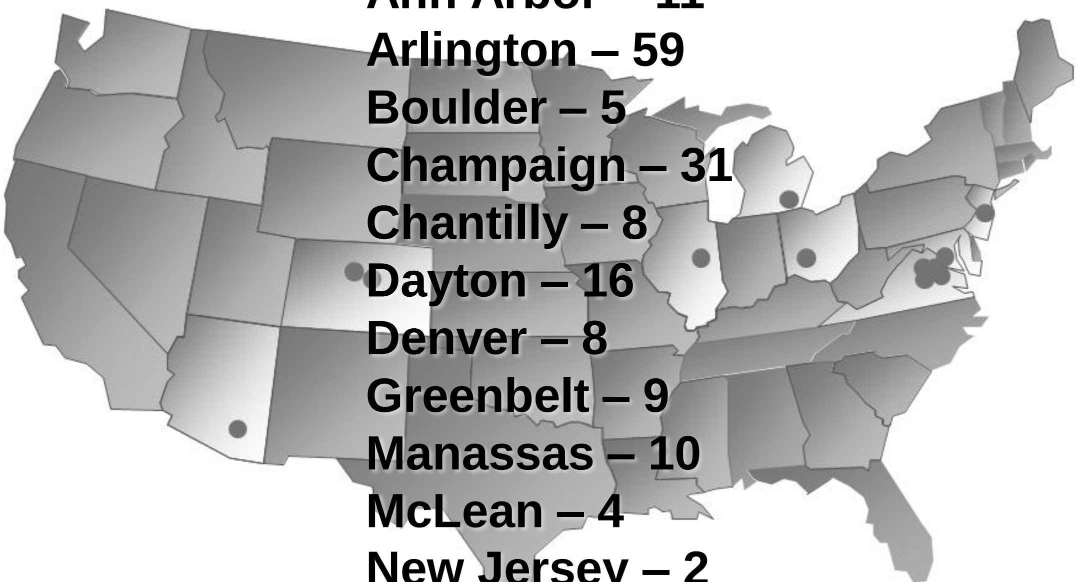
- Innovative applications of research and development
- Customer focus that leads to in-depth understanding of customer missions
- Platform independence
- Reputation for succeeding on the tough jobs
- Breadth and depth of technology and domain expertise
- Proven management track record
- Proven best practices, technologies and systems



$10.8 billion
(Fiscal Year 2010)

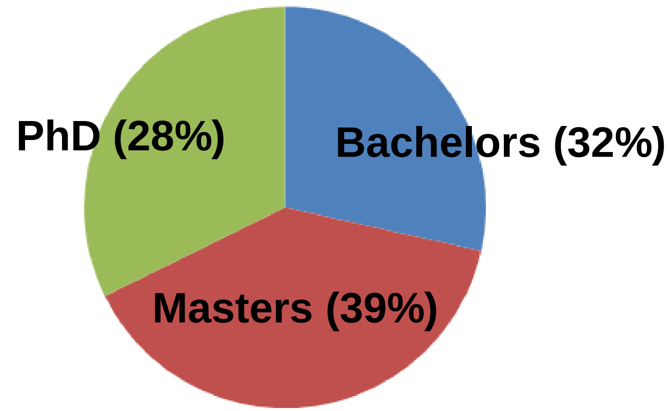
Demographics

Advanced Research and Development Division
195 Team Members

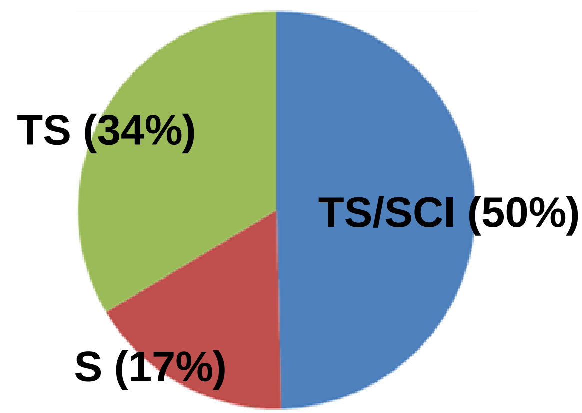


Ann Arbor – 11
Arlington – 59
Boulder – 5
Champaign – 31
Chantilly – 8
Dayton – 16
Denver – 8
Greenbelt – 9
Manassas – 10
McLean – 4
New Jersey – 2
Tucson – 32

Degrees (82%)

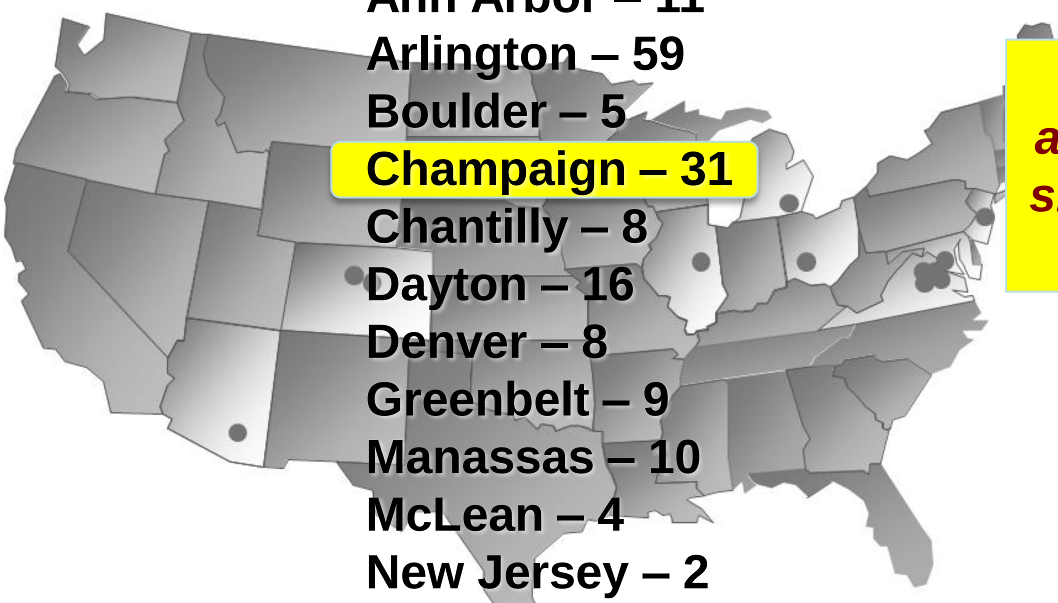


Clearances (73%)



Demographics

Advanced Research and Development Division
195 Team Members



Ann Arbor – 11
Arlington – 59
Boulder – 5
Champaign – 31
Chantilly – 8
Dayton – 16
Denver – 8
Greenbelt – 9
Manassas – 10
McLean – 4
New Jersey – 2
Tucson – 32

We develop products and algorithms for electromagnetic simulations of radar scattering and antenna radiation.

About Concurrent EDA

www.ConcurrentEDA.com

SAIC
From Science to Solutions

Sequential Software Executable | ***Proprietary Design Automation Tools*** | **Analytics in 24 hours** | **Parallel FPGA Cores** | **Integrated Cores**

We transform compiled sequential software into parallel FPGA cores.

Concurrent Analytics: Quantifies FPGA performance in 24 hours
Xilinx, Inc., FPGA cores: FPGA cores from software, pre-built or custom

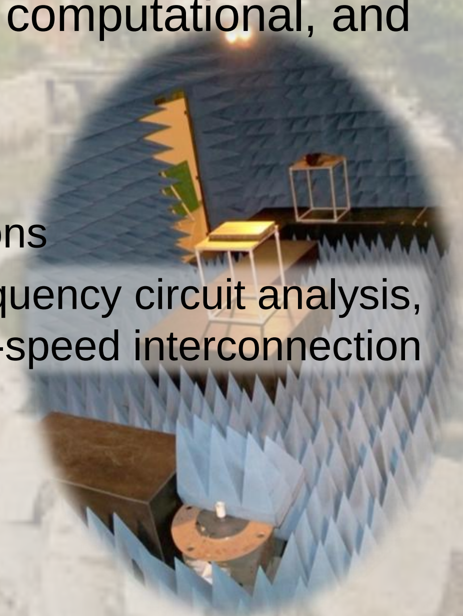
Performance:

- **High data rates:** 1 to 25 Gb/s data rates (video/image, signal, crypto)
- **Extreme processing:** 1 to 100 giga-operations per second
- **High clock rate:** 200 to 350 MHz performance
- **Fast time-to-market**: One to three weeks per core

About CCEML

www.cceml.illinois.edu

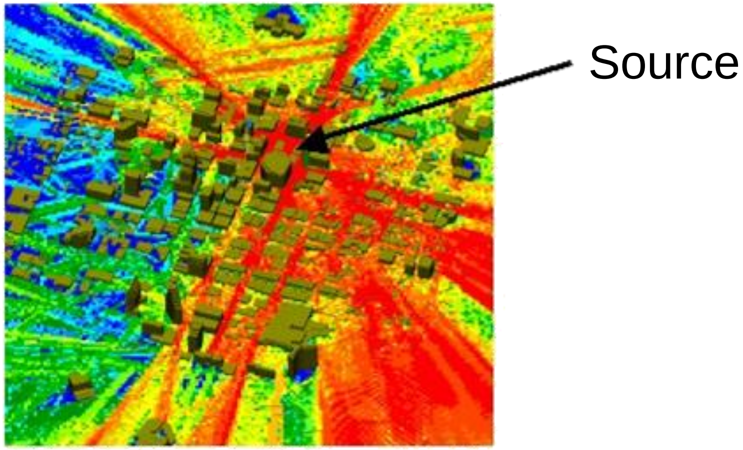
- Center for Computational Electromagnetics and Electromagnetics Laboratory (CCEML)
  - University of Illinois
  - Department of Electrical and Computer Engineering
- **Seven** faculty members and about **50** researchers
- Research activities cover many aspects in theoretical, computational, and experimental electromagnetics
  - Design of smart, reconfigurable antennas
  - Fast algorithms for large-scale electromagnetic simulations
  - Finite element methods for scattering, antenna, high-frequency circuit analysis, bioelectromagnetics, electromagnetic compatibility, high-speed interconnection modeling and electronic packaging
  - Inverse scattering and remote sensing
  - Optoelectronics and integrated optics



Overview

Field profiles have many applications

- Field profiles are color plots showing the electric field strength for many observation points in a scene



- Defense and commercial communication system applications rely on field profiles
  - Prediction of system coverage can help ensure communication links are maintained

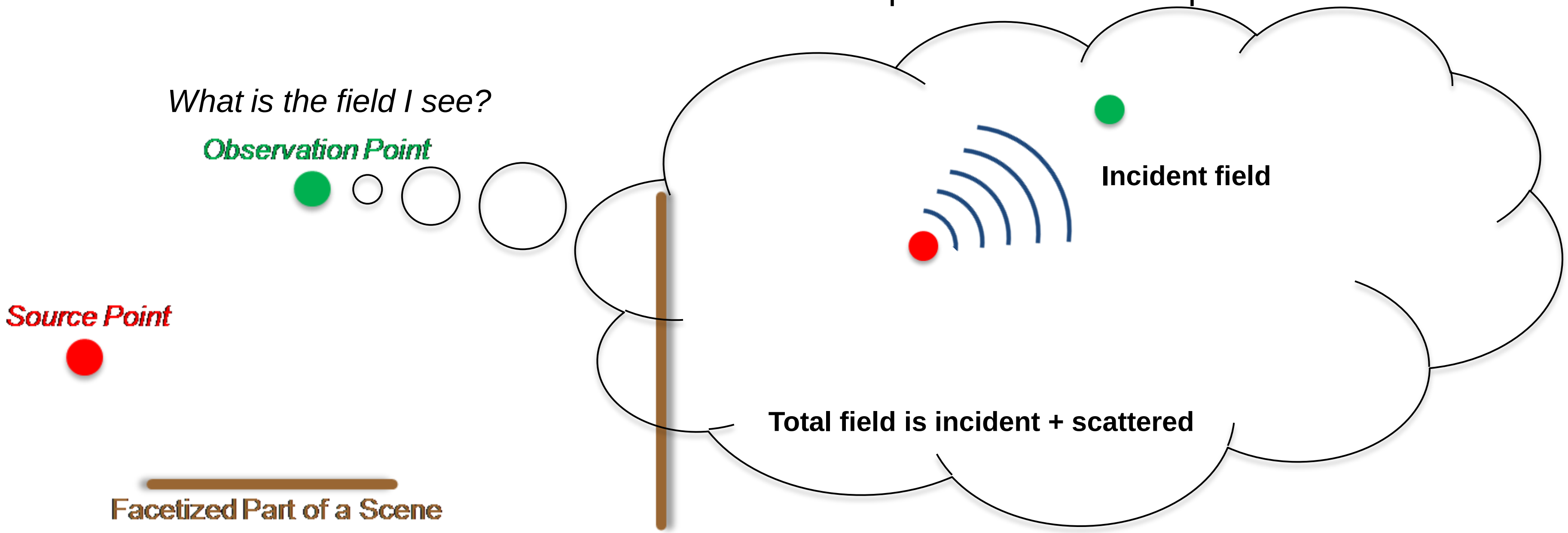
Overview

SAIC
From Science to Solutions

- Calculating field profiles for large scenes requires long simulation times
- Physical Optics - Shooting and Bouncing Ray (PO-SBR) is one of the fastest techniques for generating accurate results
  - Full-wave techniques like Method of Moments (MoM) are generally more accurate, but suffer from intractable memory/run-times
- Geometric Optics - Shooting and Bouncing Ray (GO-SBR) is an alternative
  - Arguably the fastest technique, but suffers from reduced accuracy and discontinuous predictions

Background
Introduction to the PO-SBR method

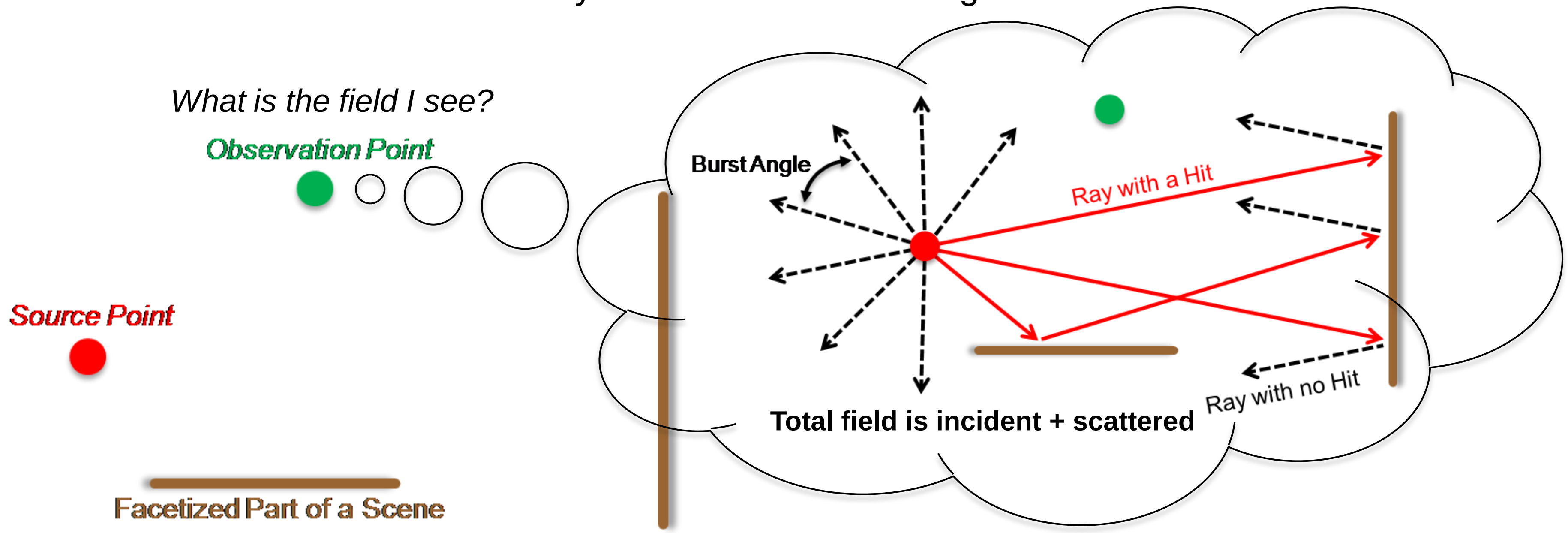
- **Physical optics - shooting and bouncing ray (PO-SBR)**
  - Method for finding the electromagnetic field at an observation point due to a source in the presence of a scene
  - Step 1: **Incident field**
    - Radiate from source to the observation point as if in free-space



Background

Introduction to the PO-SBR method

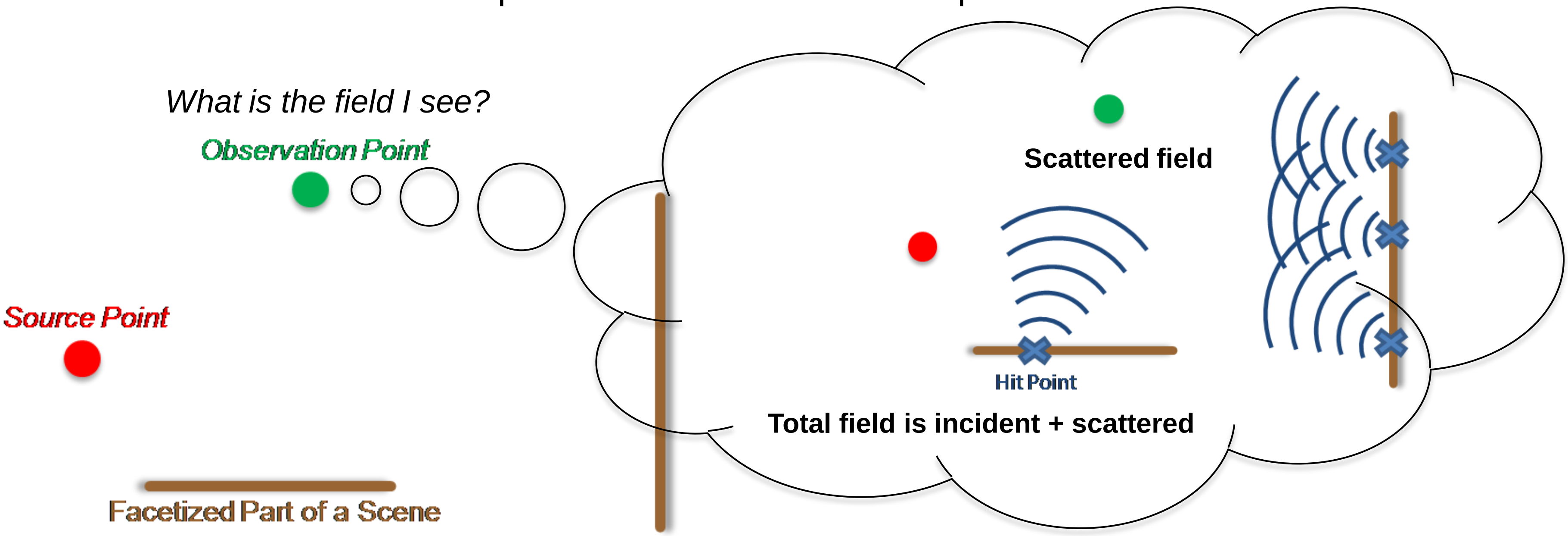
- **Physical optics - shooting and bouncing ray (PO-SBR)**
  - Method for finding the electromagnetic field at an observation point due to a source in the presence of a scene
  - Step 2: **SBR**
    - Launch a burst of rays and trace them through the scene



Background

Introduction to the PO-SBR method

- **Physical optics - shooting and bouncing ray (PO-SBR)**
  - Method for finding the electromagnetic field at an observation point due to a source in the presence of a scene
  - Step 3: **PO**
    - Radiate from equivalent surface currents to produce scattered field



Overview

Two versions under development: GPU and FPGA

- Repetitive process of finding and radiating hit points makes PO-SBR a natural candidate for parallelization

- This study investigated two forms of parallelization to see if PO-SBR could be improved for field profile applications
  - **GPU** implementation
    - Quadro® FX 5800
    - 4 GB
    - 240 cores (30 multiprocessors)
  - **FPGA** estimation

PO = physical optics
SBR = shooting and bouncing ray

The NVIDIA logo and Quadro are registered trademarks of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800

SAIC®
From Science to Solutions

Identify Volley of Rays

CPU
central processing unit

Parallelize Frequencies / Observations (CUDA™)

Parallelize Rays (OptiX™)

PO- --- **GPU** - **GPU** --- **-SBR**

physical optics — graphics processing unit — graphics processing unit — shooting and bouncing ray

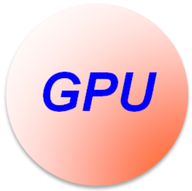
(loops over all rays in volley)

Quadro, CUDA, and OptiX are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800

Calculate Incident Field

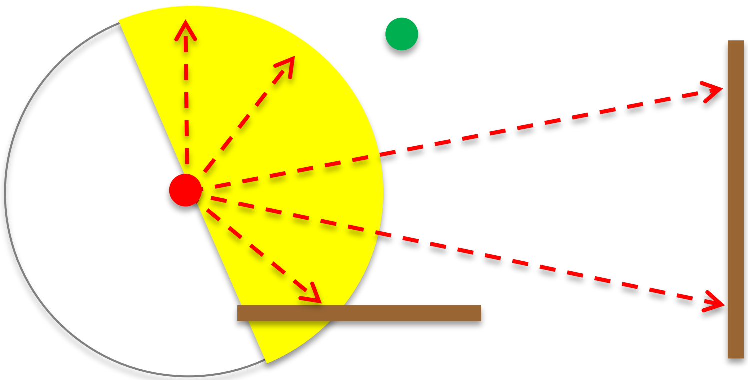


Parallelize Across Frequencies and Observations

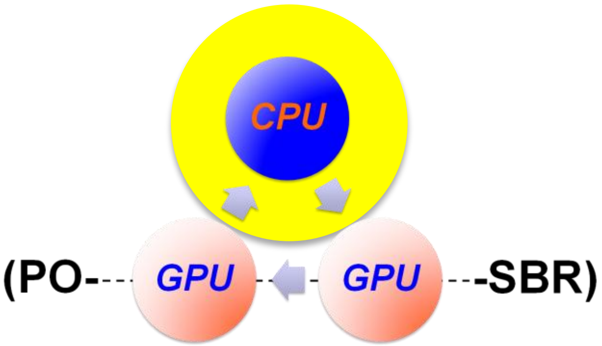
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



First of Two Ray Shoot Sections



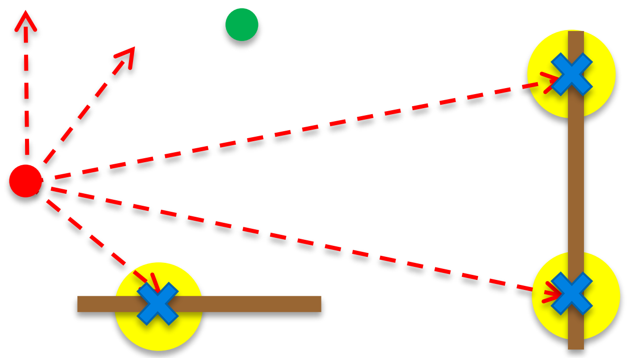
Identify First Volley of Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

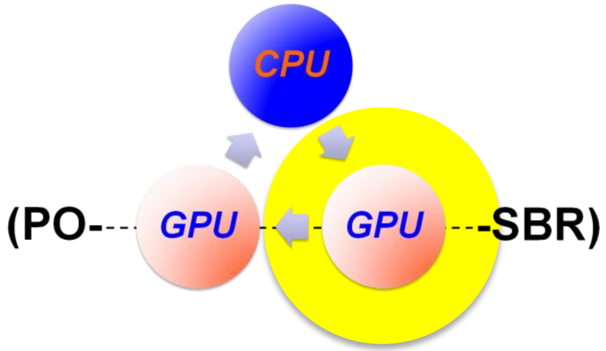
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Hit Points



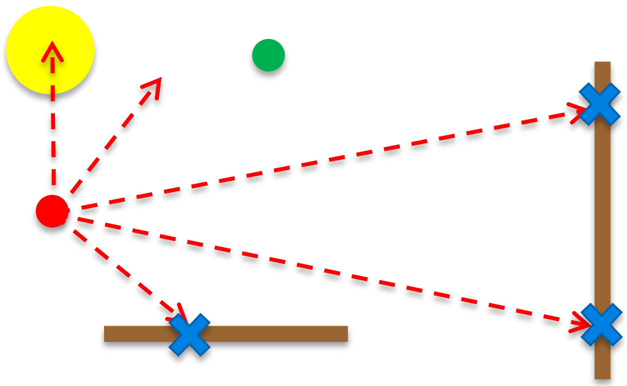
Parallelize Across Rays

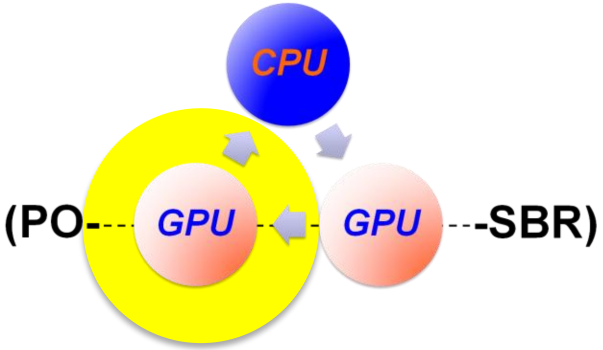
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800





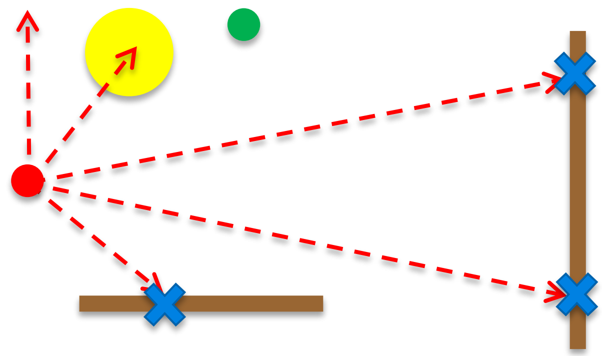
Parallelize Across Frequencies and Observations

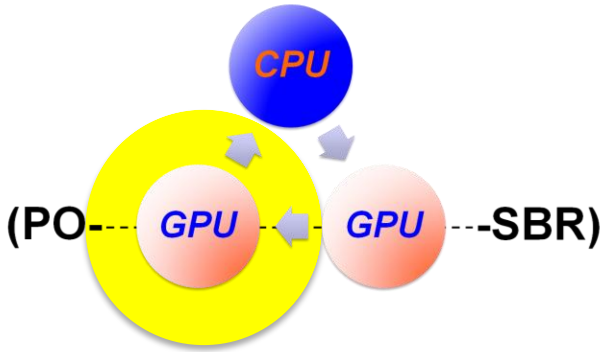
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800





Parallelize Across Frequencies and Observations

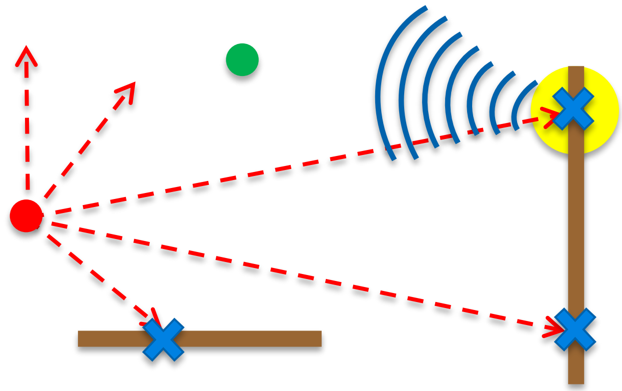
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

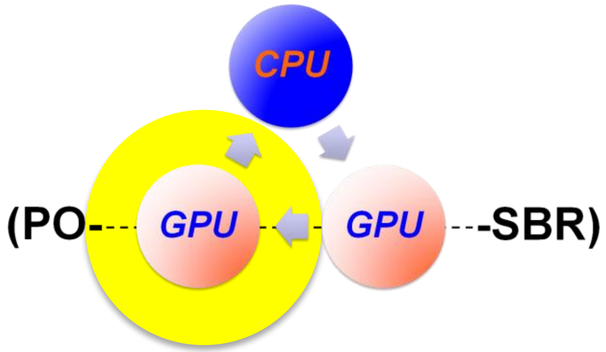
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Scattered Field



Parallelize Across Frequencies and Observations

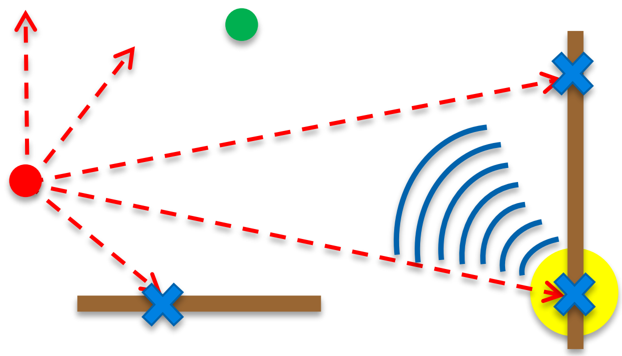
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

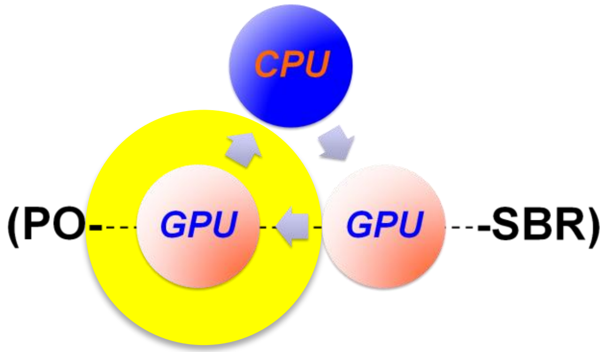
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Scattered Field



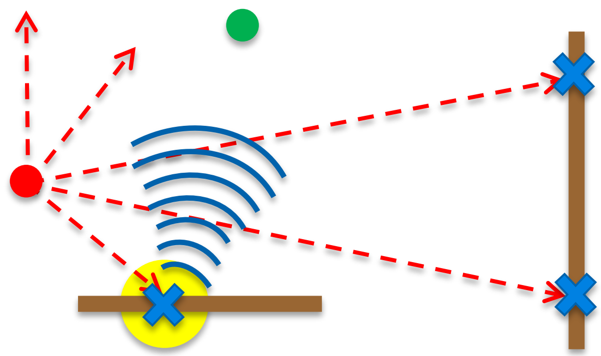
Parallelize Across Frequencies and Observations

(loops over all rays in volley)

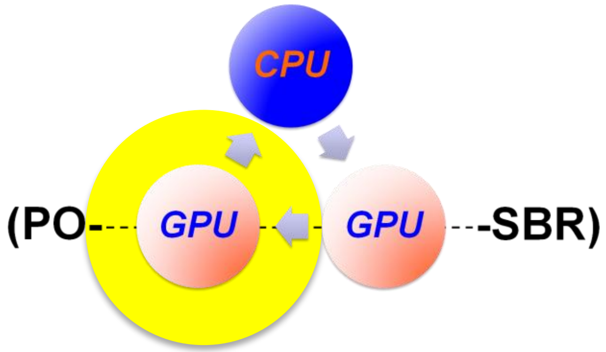
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Scattered Field



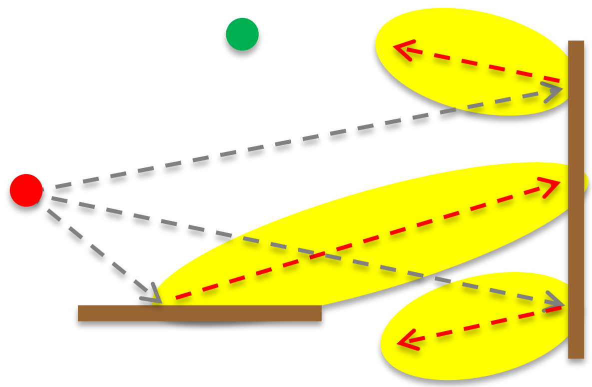
Parallelize Across Frequencies and Observations

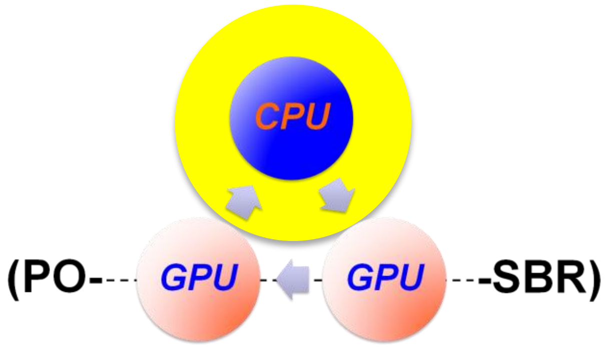
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



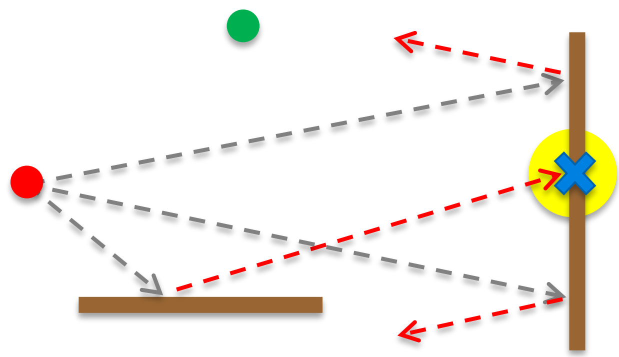


Identify Second Volley of Rays

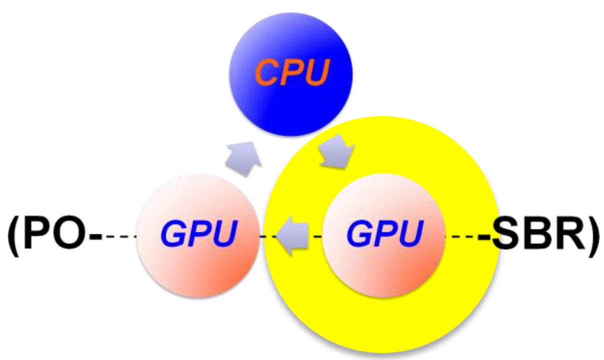
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Hit Points

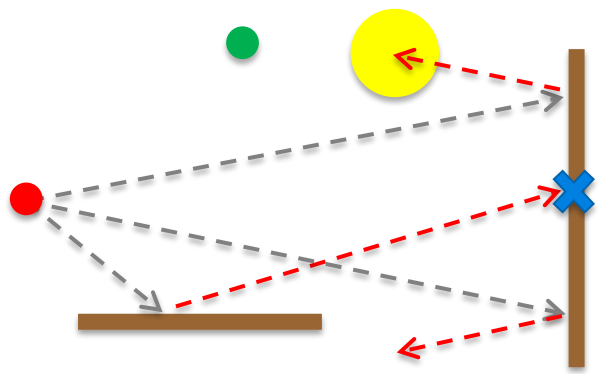


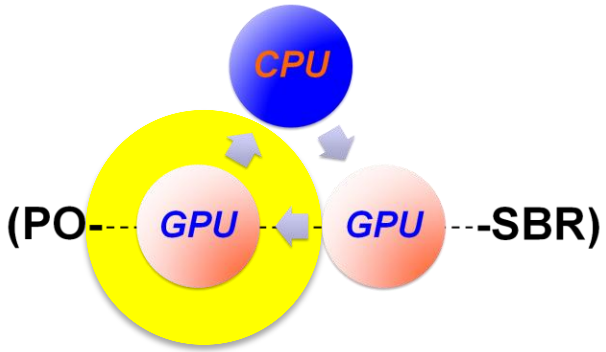
Parallelize Across Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800





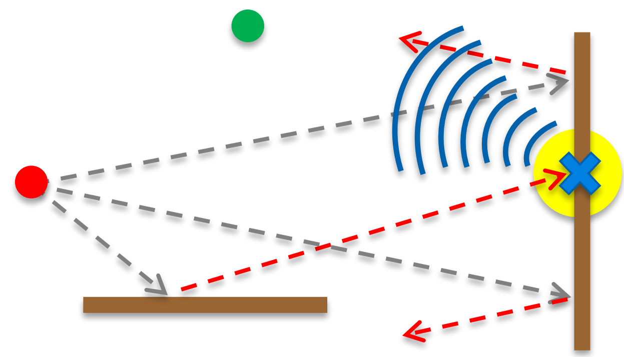
Parallelize Across Frequencies and Observations

(loops over all rays in volley)

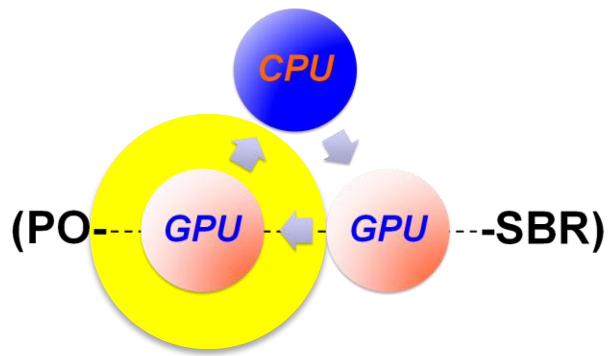
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Calculate Scattered Field



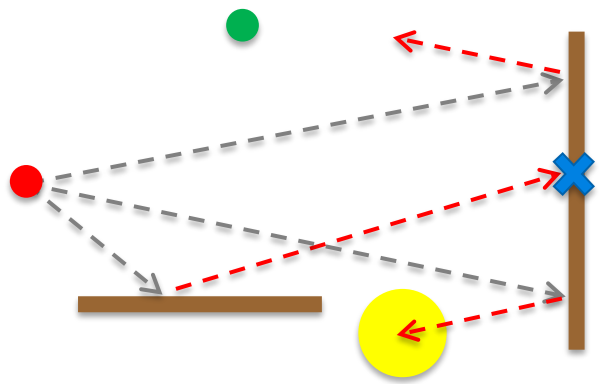
Parallelize Across Frequencies and Observations

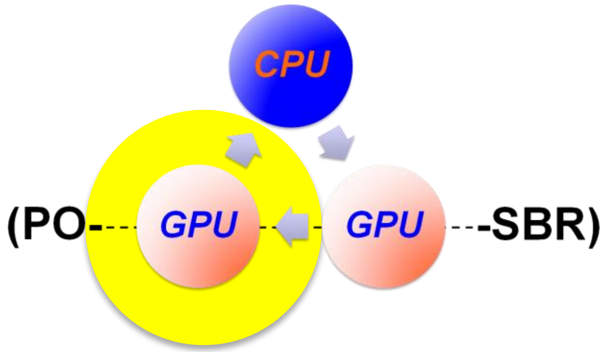
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800





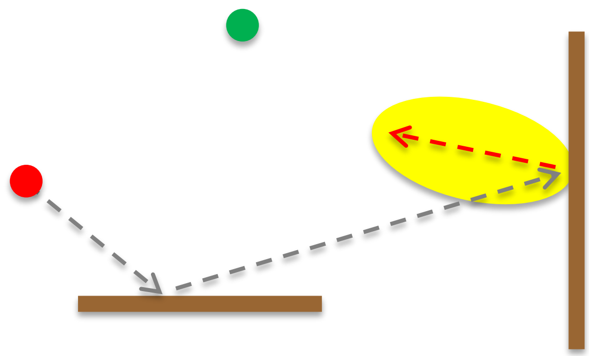
Parallelize Across Frequencies and Observations

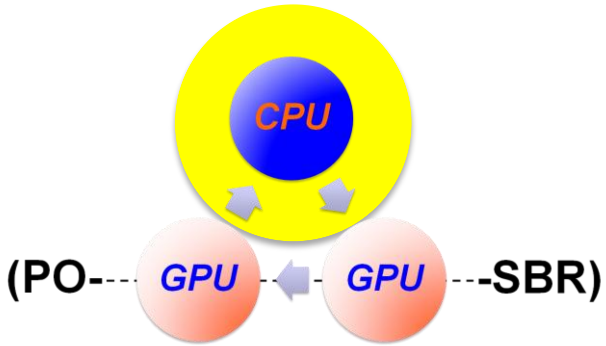
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



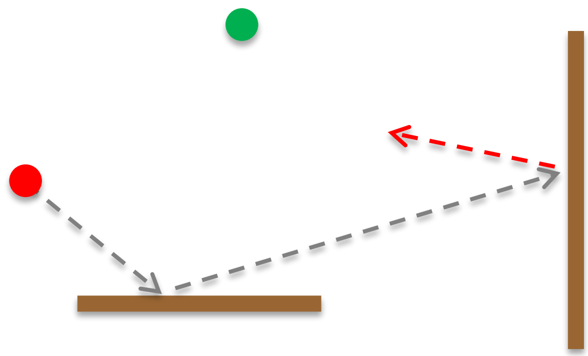


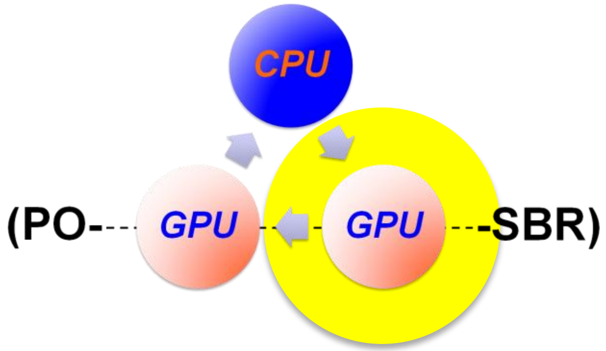
Identify Third Volley of Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800





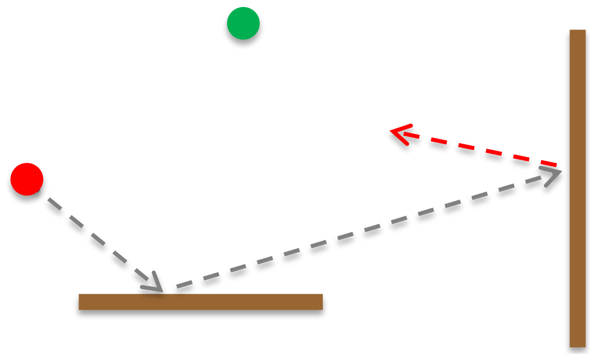
Parallelize Across Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray

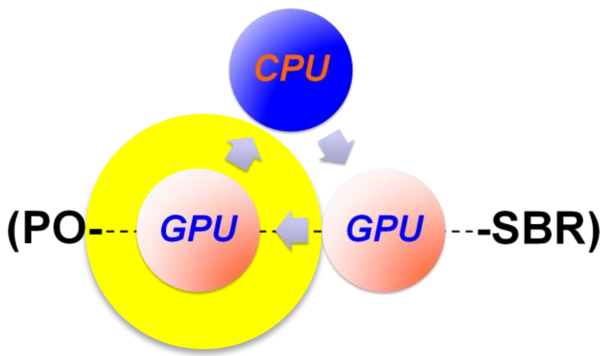
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



SKIP (NO HITS)



Paralleliz... ...s
Freq... ...s and
...rvations

SKIP (NO HITS)

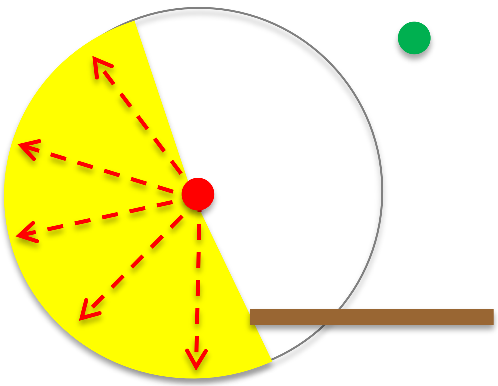
(loops over all rays in volley)

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

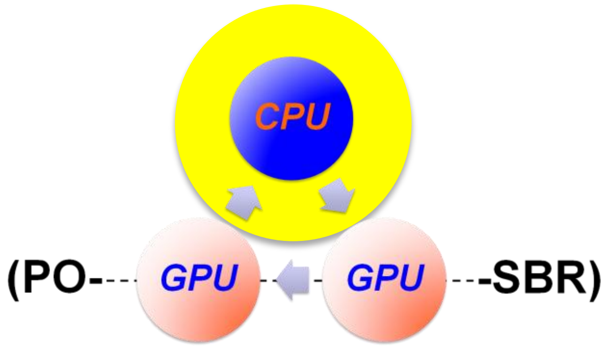
GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800

SAIC® From Science to Solutions



Second of Two Ray Shoot Sections

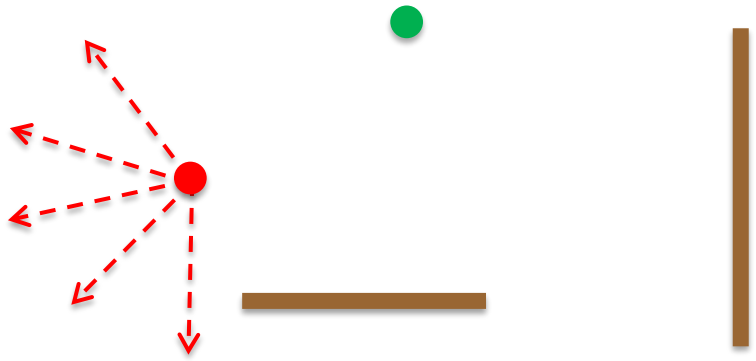


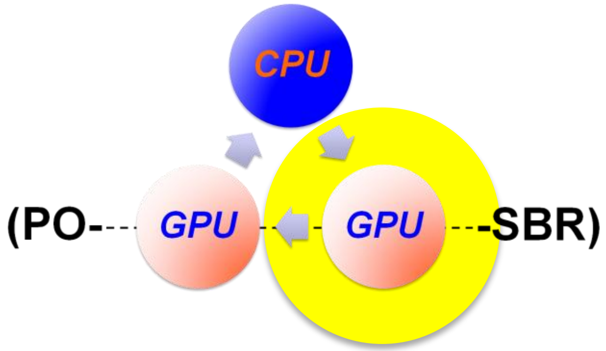
Identify Fourth Volley of Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



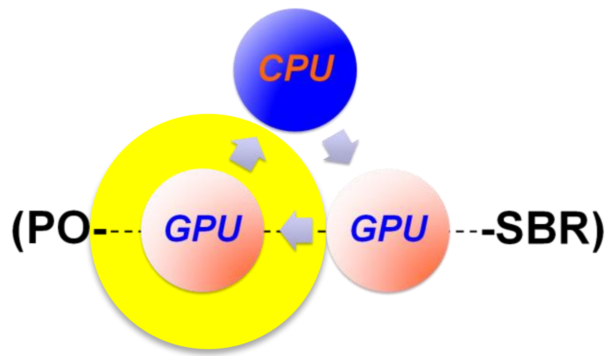


Parallelize Across Rays

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800



Paralleliz… s
Fre… es &
…vations

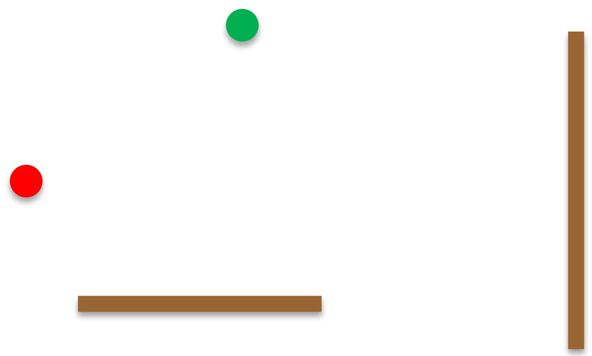
(loops over all rays in volley)

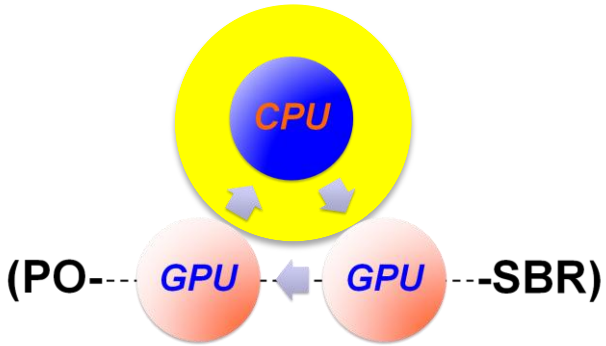
CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800

From Science to Solutions





Visualization of Field Profile

CPU = central processing unit GPU = graphics processing unit PO = physical optics SBR = shooting and bouncing ray
Quadro is a registered trademark of NVIDIA Corporation in the U.S. and other countries.

GPU Implementation

Iterative ray-tracing simulation with Quadro® FX 5800

SAIC®
From Science to Solutions

Identify Volley of Rays

CPU
central processing unit

Parallelize Frequencies / Observations (CUDA™)

Parallelize Rays (OptiX™)

PO- - - - **GPU** - ← - **GPU** - - - **-SBR**

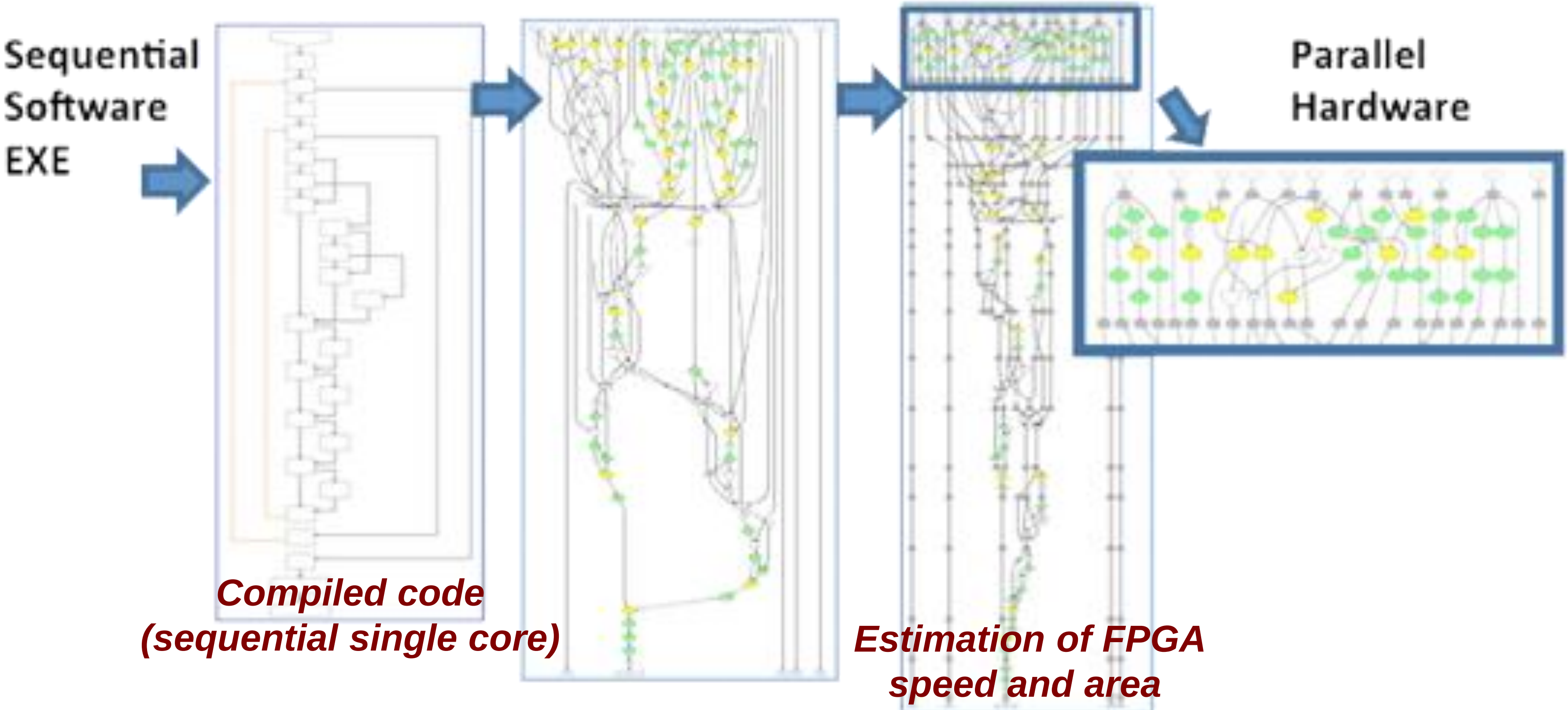
physical optics | graphics processing unit | graphics processing unit | shooting and bouncing ray

(loops over all rays in volley)

Quadro, CUDA, and OptiX are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries.

FPGA Estimation

Concurrent Analytics™ tool used to quantify performance



Compiled code (sequential single core)

Extracts parallelism (transforms loops into hardware pipelines)

Estimation of FPGA speed and area

Concurrent Analytics is a trademark of Concurrent EDA LLC in the U.S. and other countries.

FPGA Estimation

Concurrent Analytics™ tool used to quantify performance

Formula
(millions of sets of $a_{1,1}$ ... $a_{3,3}$)

$$B_{2,2}= (a_{1,1}*m_{11})+(a_{1,2}*m_{12})+(a_{1,3}*m_{13})+ (a_{2,1}*m_{21})+(a_{2,2}*m_{22})+(a_{2,3}*m_{23})+ (a_{3,1}*m_{31})+(a_{3,2}*m_{32})+(a_{3,3}*m_{33})$$

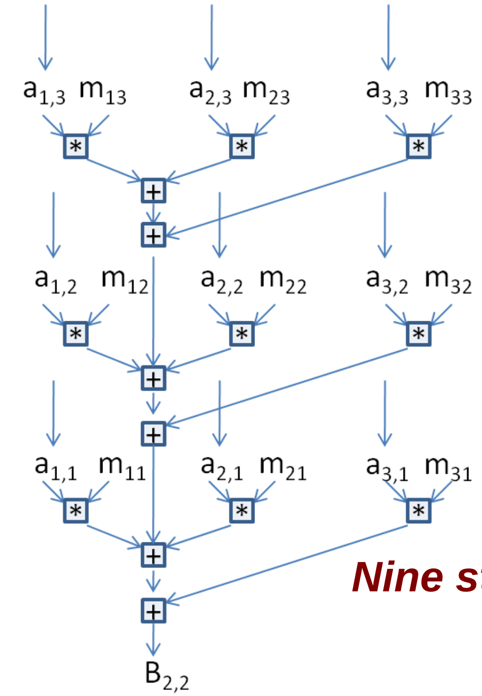
(example from imagery edge extraction
NOT electromagnetic simulation)

Compiled code

$r0=a_{1,1}$
$r1=r0*m_{11}$
$r2= r2 + r1$
$r0=a_{1,2}$
$r1=r0*m_{12}$
$r2= r2 + r1$
$r0=a_{1,3}$
$r1=r0*m_{13}$
$r2= r2 + r1$
$r0=a_{2,1}$
$r1=r0*m_{21}$
$r2= r2 + r1$
$r0=a_{2,2}$
$r1=r0*m_{22}$
$r2= r2 + r1$
$r0=a_{2,3}$
$r1=r0*m_{23}$
$r2= r2 + r1$
$r0=a_{3,1}$
$r1=r0*m_{31}$
$r2= r2 + r1$
$r0=a_{3,2}$
$r1=r0*m_{32}$
$r2= r2 + r1$
$r0=a_{3,3}$
$r1=r0*m_{33}$
$r2= r2 + r1$
$B_{2,2}=r2$

28 steps

Hardware pipeline



Nine stages

Concurrent Analytics is a trademark of Concurrent EDA LLC in the U.S. and other countries.

FPGA Estimation

Concurrent Analytics™ tool used to quantify performance

FPGA Implementation

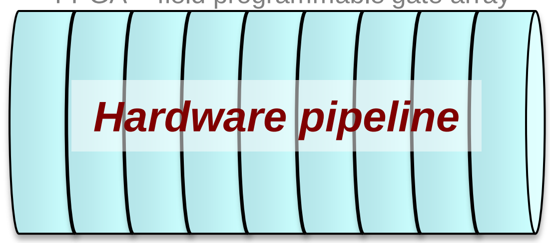
FPGA = field programmable gate array

$a_{1,1}$ . . . $a_{3,3}$ — Set N

...

$a_{1,1}$ . . . $a_{3,3}$ — Set 2

$a_{1,1}$ . . . $a_{3,3}$ — Set 1



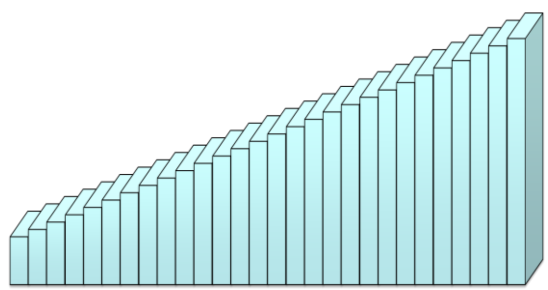
*Nine stages executing **concurrently***
(200 MHz)

Once pipeline filled, new result returned every 5 ns

CPU Implementation

CPU = central processing unit

$a_{1,1}$ . . . $a_{3,3}$



*28 steps executing **sequentially***
(2.4 GHz)

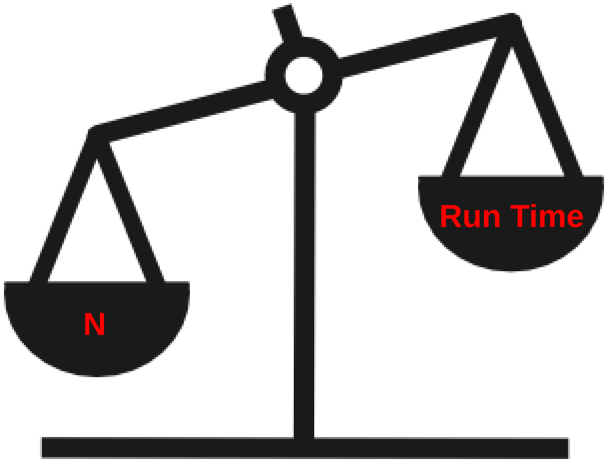
Process must be repeated N times

Concurrent Analytics is a trademark of Concurrent EDA LLC in the U.S. and other countries.

Examples

Evaluating the run time speed-up of GPU / FPGA to CPU

- There are three main parameters that influence the performance
  - The number of observation points (**N_{obs}**)
  - The number of frequencies (**N_{freq}**)
  - The number of hit points (**N_{hit}**)



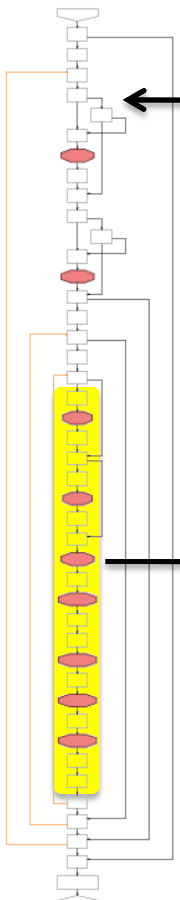
Heavier "N" means longer run time

- Three examples investigated different "**N**"
  - *Urban Scene*
    - Focuses on shooting and bouncing ray (**SBR**) aspect of code (**N_{hit}**)
  - *Four Plate*
    - Focuses on physical optics (**PO**) aspect of code (**N_{obs}**)
  - *Fun Car*
    - Focuses on **PO-SBR** aspect of code (**N_{obs}, N_{hit}**)
- CPU version uses PBRT (www.pbrt.org) code for ray tracing

CPU = central processing unit GPU = graphics processing unit FPGA = field programmable gate array

FPGA Estimation

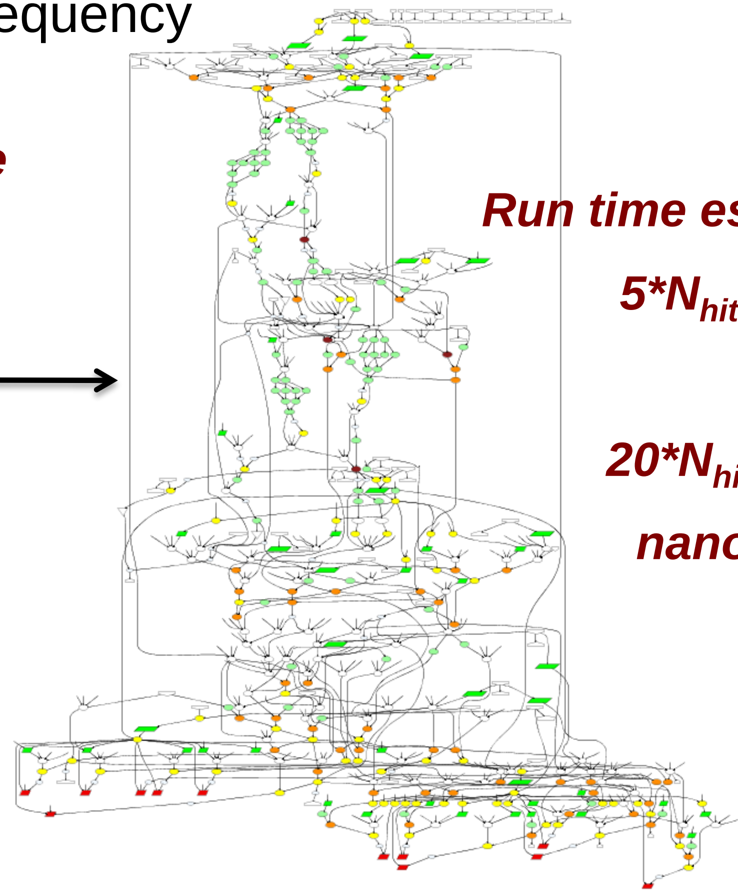
- Each cycle will generate a result for a single combination of hit point, observation point, and frequency



Compiled CPU code for PO calculation

Hardware pipeline (350 cycles to fill)

Triple nested loop N_{hit}, N_{obs}, N_{freq}



Run time estimate between

$5*N_{hit}*N_{obs}*N_{freq}$

and

$20*N_{hit}*N_{obs}*N_{freq}$

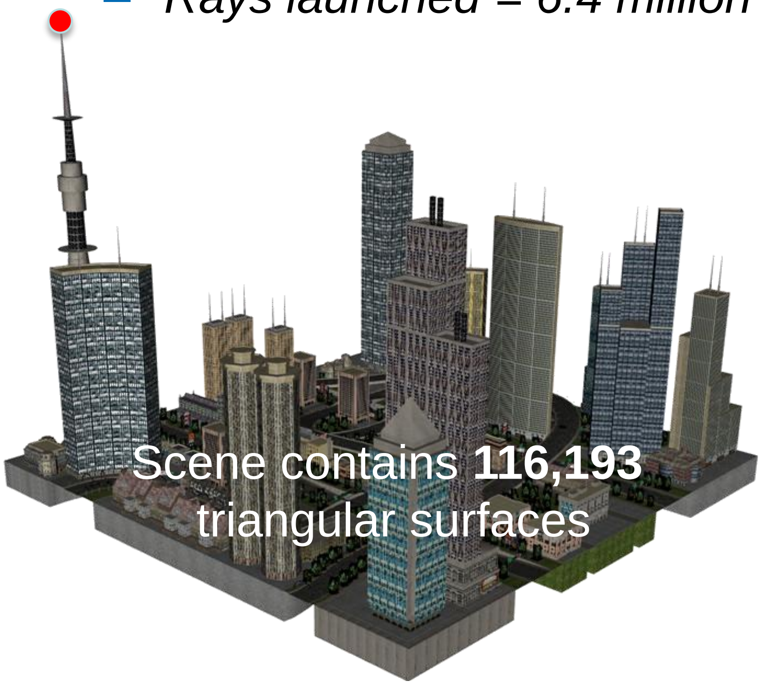
nanoseconds

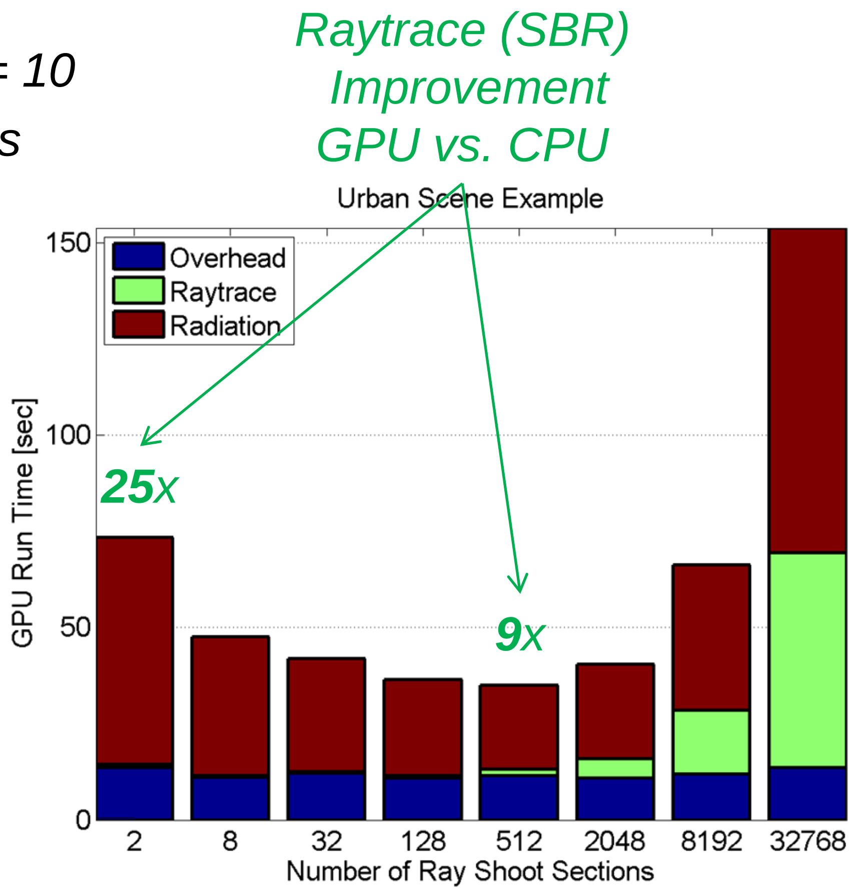
CPU = central processing unit PO = physical optics FPGA = field programmable gate array

Example 1: Urban Scene

Computational emphasis in tracing the rays (SBR)

- Single frequency and single observation point
  - *Maximum number of bounces = 10*
  - *Burst interval size = 0.1 degrees*
  - *Rays launched = 6.4 million*



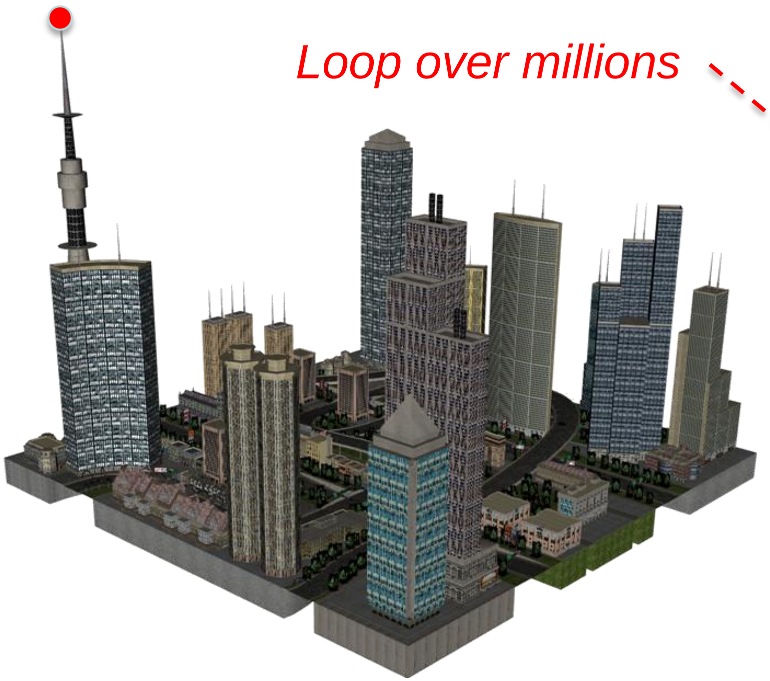


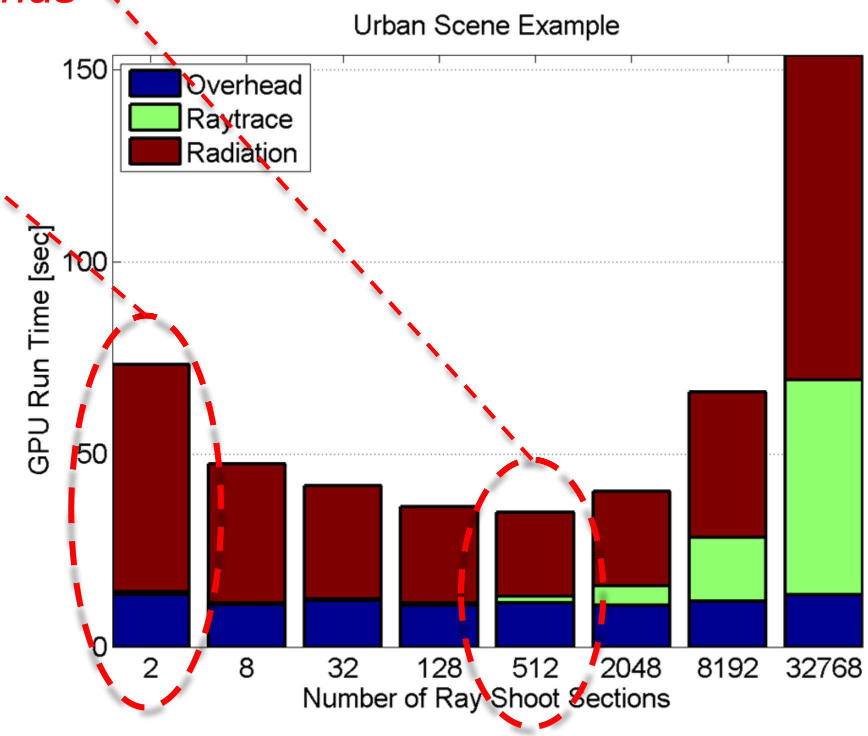
CPU = central processing unit GPU = graphics processing unit SBR = shooting and bouncing ray

Example 1: Urban Scene

Computational emphasis in tracing the rays (SBR)

- Trend of longer total run time for small numbers of sections
  - GPU must check for intersections at each bounce before radiating



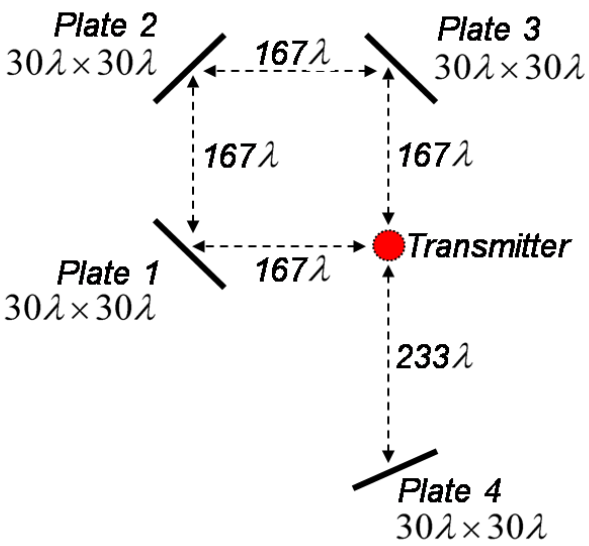


CPU = central processing unit GPU = graphics processing unit SBR = shooting and bouncing ray

Example 2: Four Plate

Computational emphasis in evaluating the fields (PO)

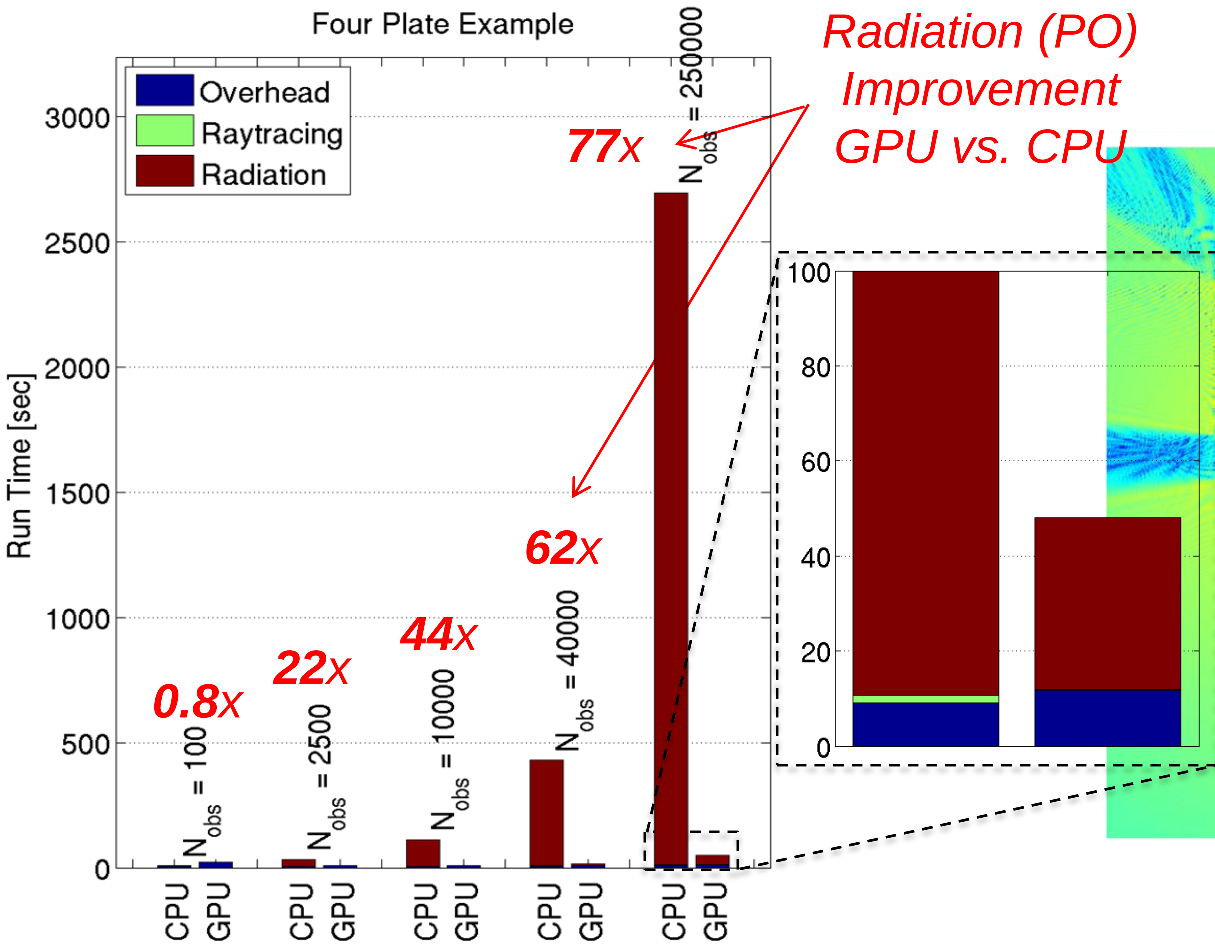
- Single frequency and multiple observation points
  - *Frequency = 10 GHz*
  - *Burst interval size = 0.25 degrees*
  - *Ray surface intersections = 12,746*

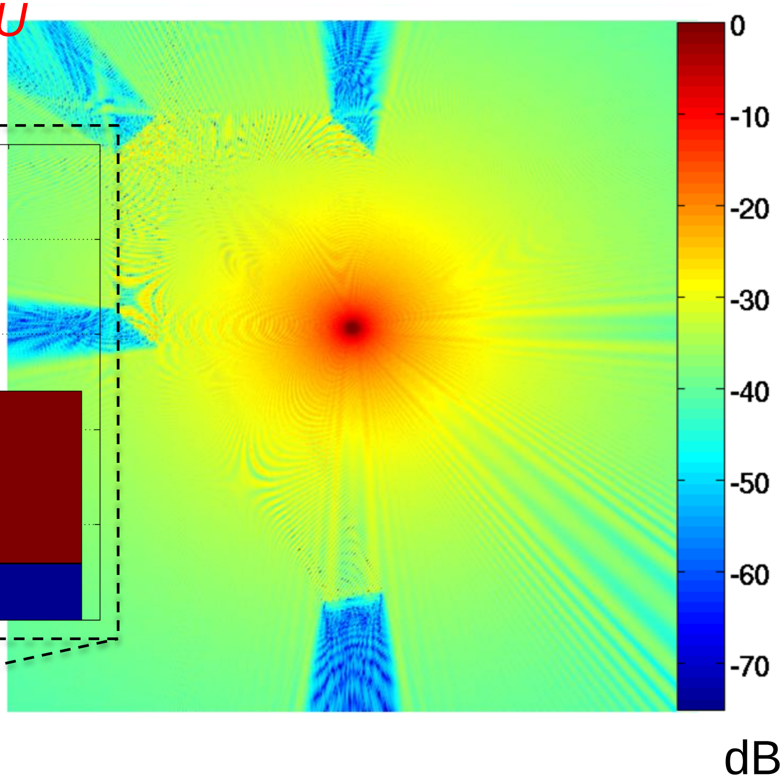


- Compare simulation times for field profiles
  - 100 observation points
  - 2,500
  - 10,000
  - 40,000
  - 250,000

Example 2: Four Plate

Computational emphasis in evaluating the fields (PO)



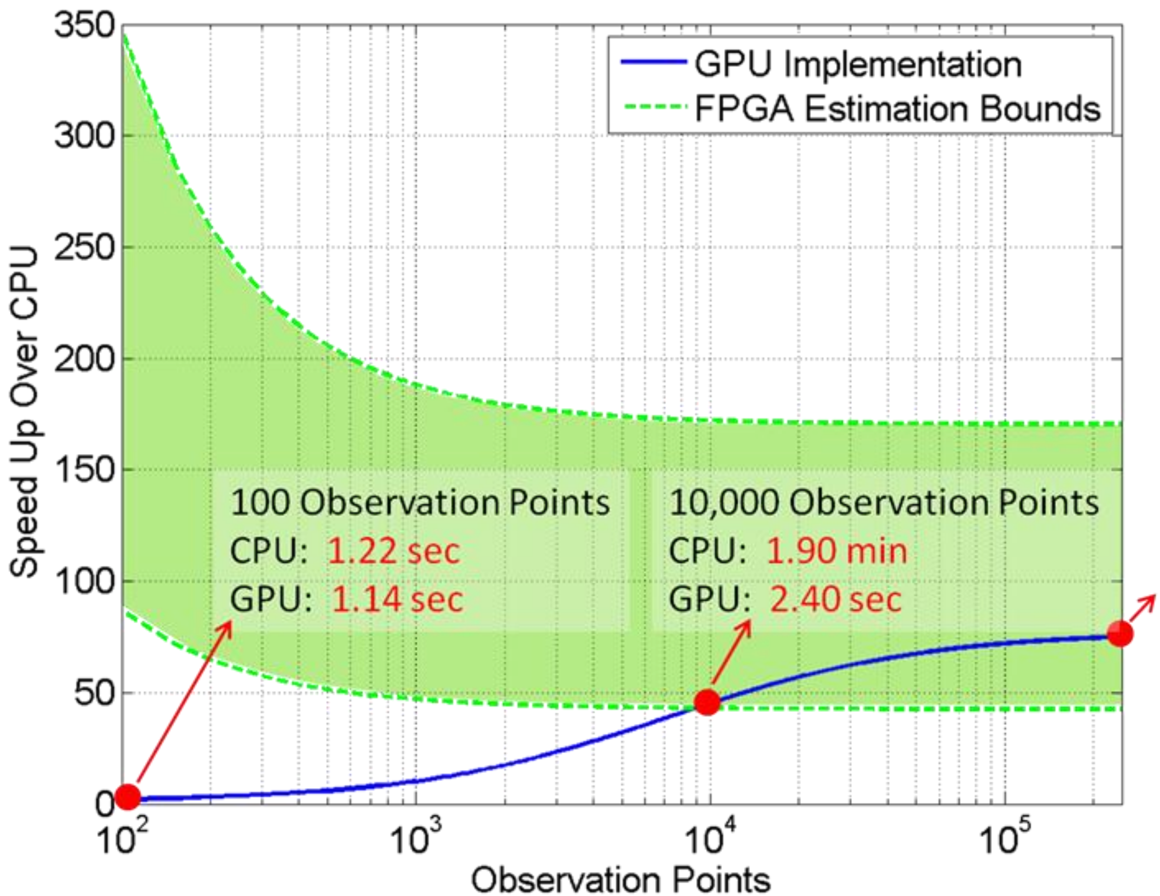


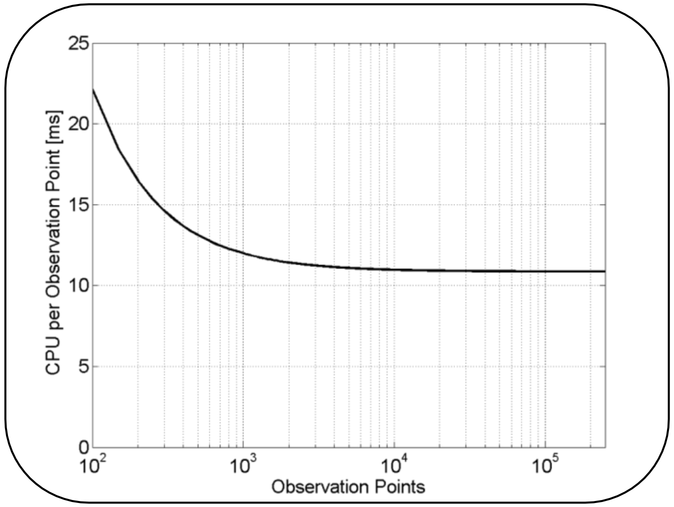
CPU = central processing unit GPU = graphics processing unit PO = physical optics

Energy | Environment | National Security | Health | Critical Infrastructure

Example 2: Four Plate

Computational emphasis in evaluating the fields (PO)



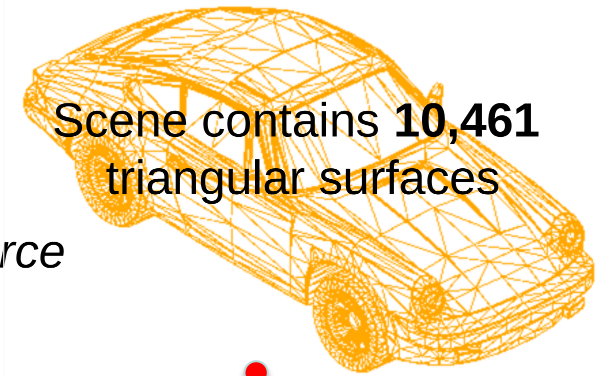


All times are PO time (does not include SBR time)

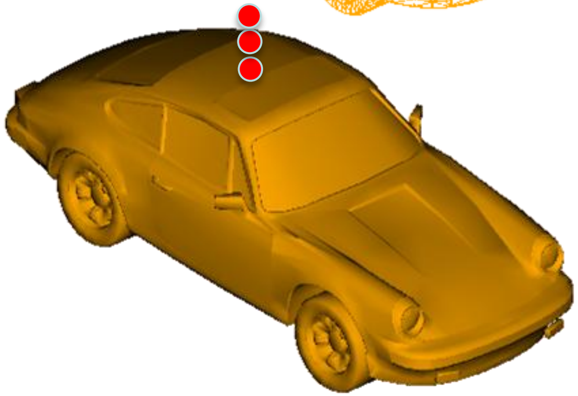
PO = physical optics
SBR = shooting and bouncing ray

Example 3: Fun Car

- Single frequency and multiple observation points
  - *Maximum number of bounces = 1*
  - *Burst interval size = 0.25 degrees*
  - *Ray shoot sections = 1*
  - *Ray surface intersections = 485,406 per source*



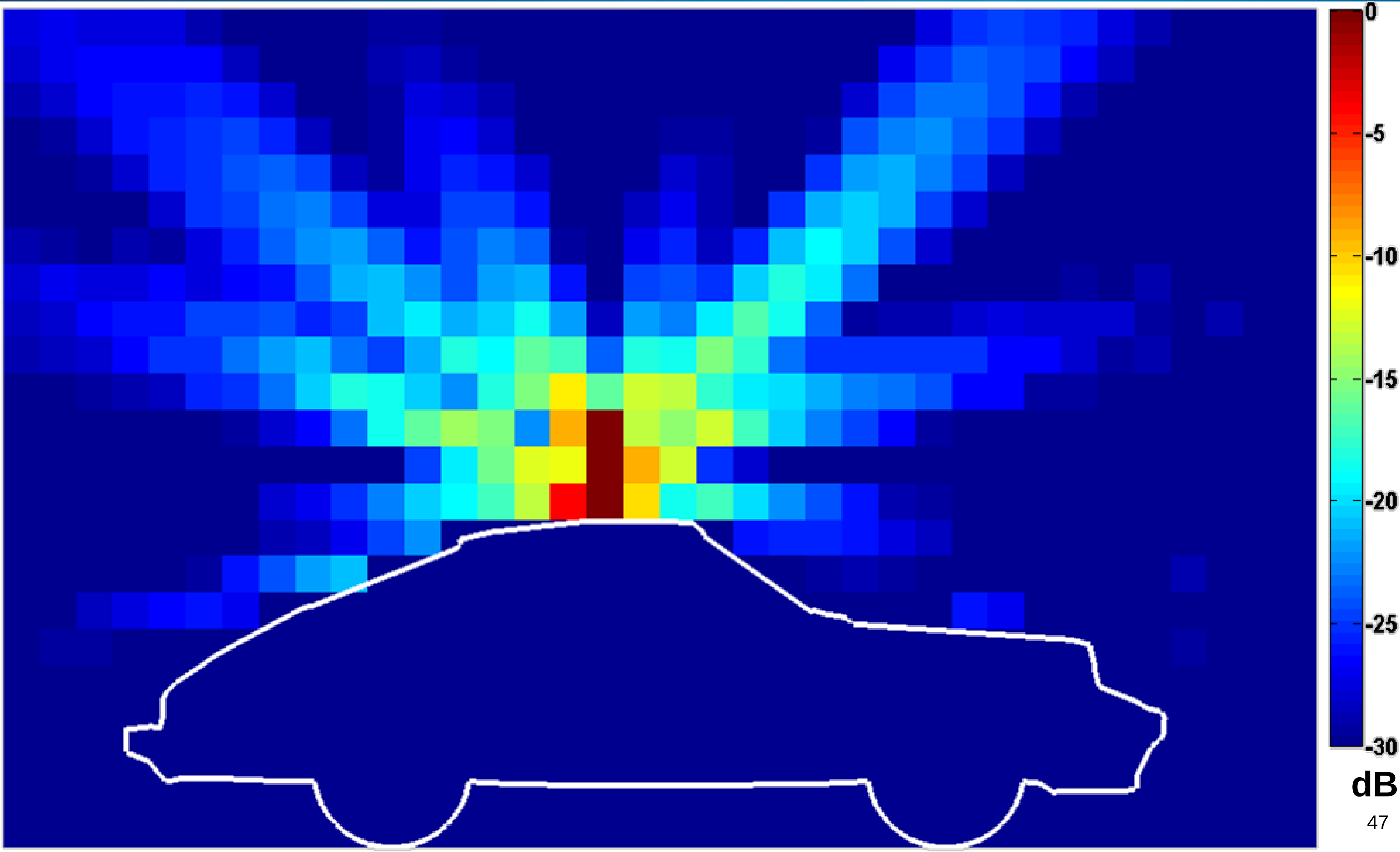
- Compare simulation times for field profiles
  - 15 cm resolution (1 point per wavelength)
  - 6 cm (2.5 points per wavelength)
  - 3 cm (5 points per wavelength)
  - 15 mm (10 points per wavelength)
  - 7.5 mm (20 points per wavelength)



Series of 41 point sources at 2 GHz used to model 12 inch cell phone car antenna

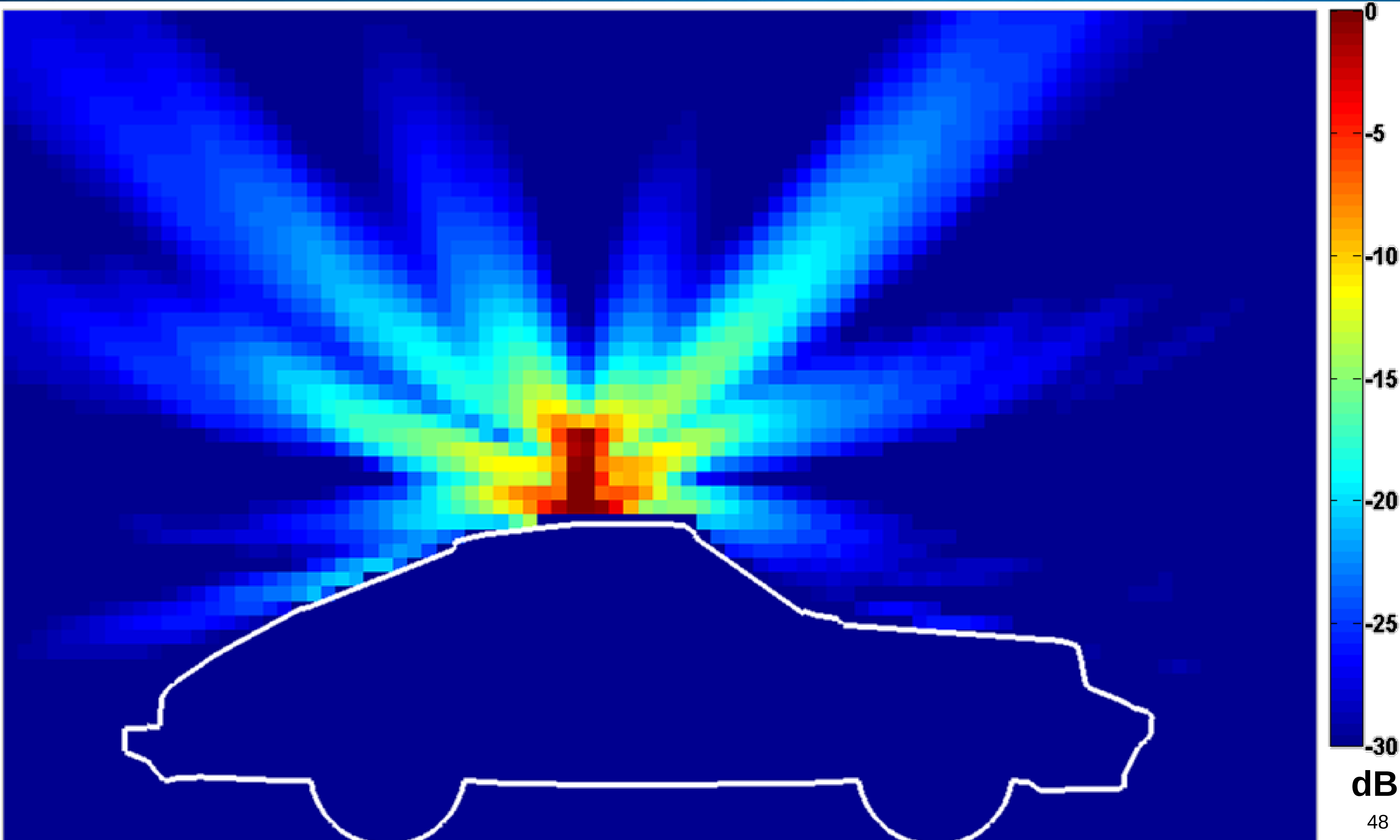
Example 3: Fun Car

15 cm resolution (737 observation points), 3.1 min (GPU version)



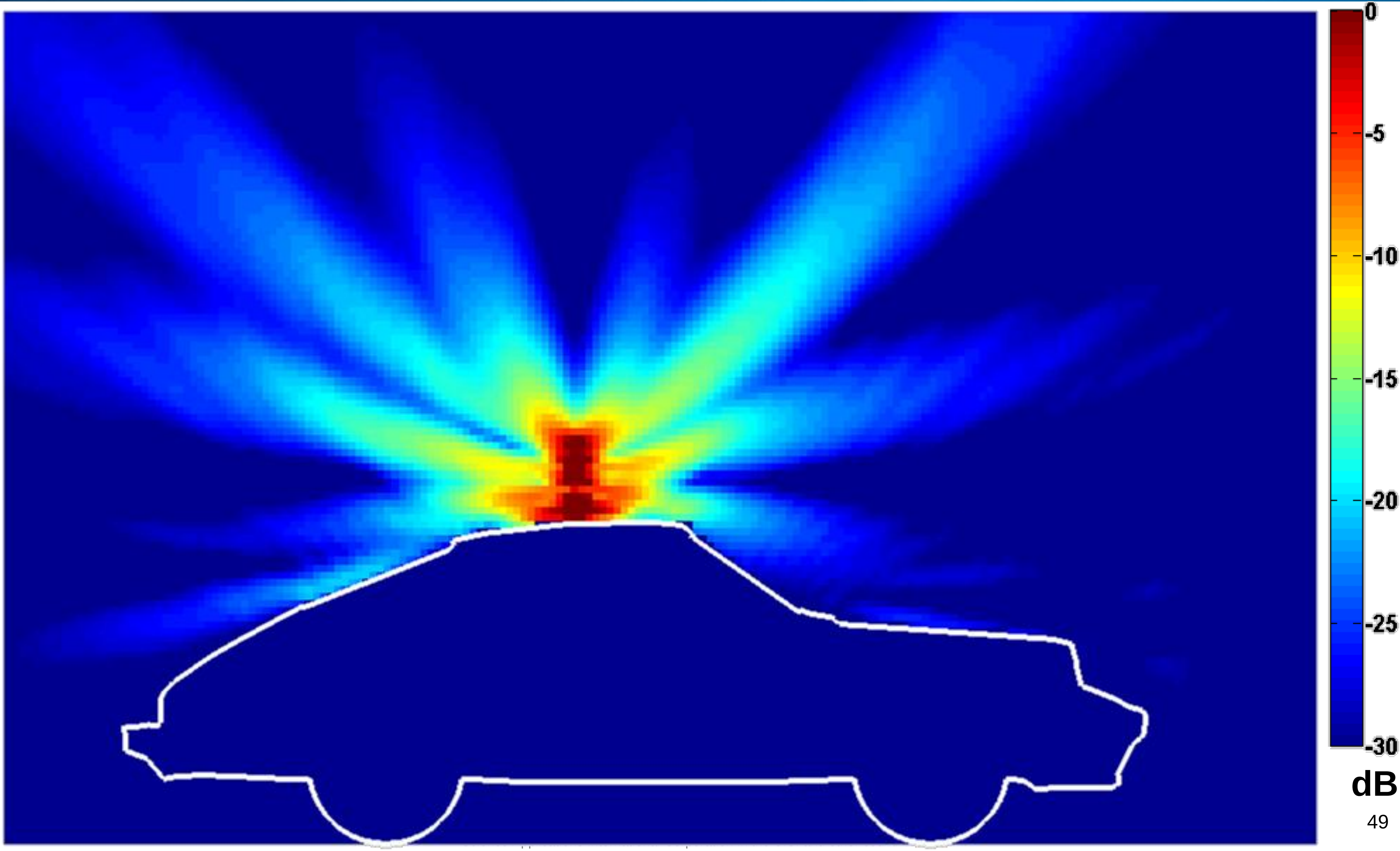
Example 3: Fun Car

6 cm resolution (4,483 observation points), 4.4 min (GPU version)



Example 3: Fun Car

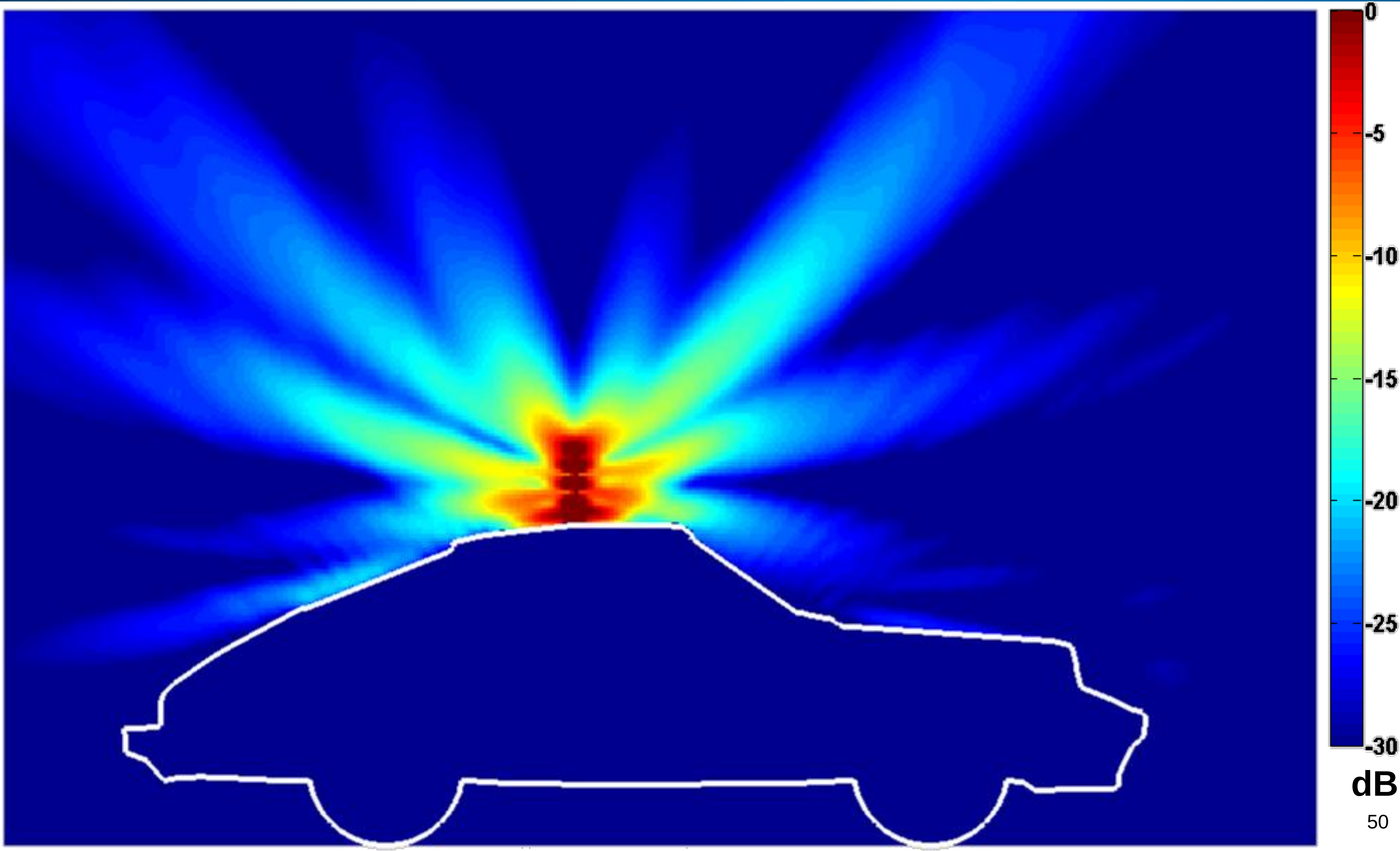
3 cm resolution (17,741 observation points), 15.3 min (GPU version)



Example 3: Fun Car

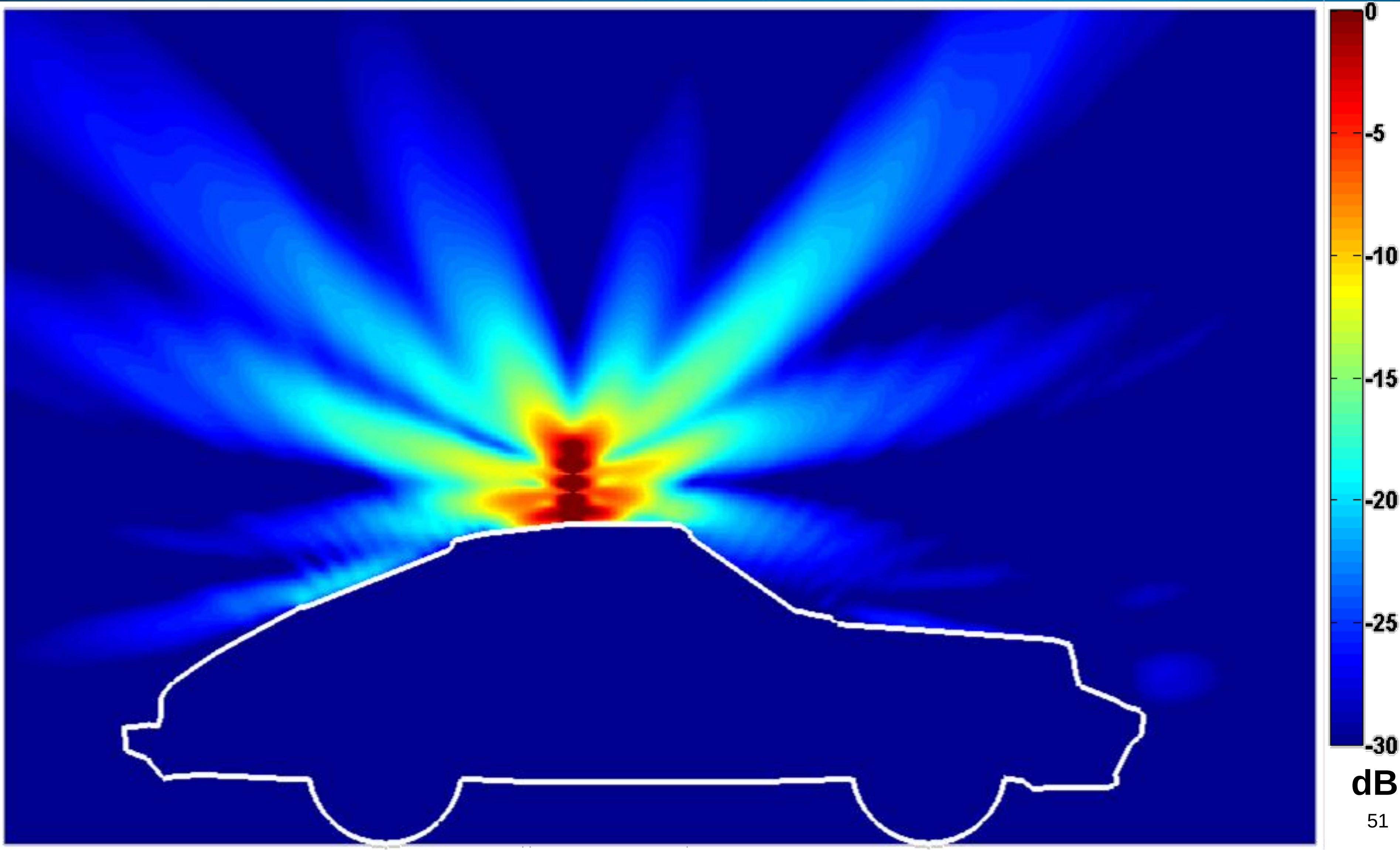
15 mm resolution (70,754 observation points), 0.95 hrs (GPU version)

SAIC® From Science to Solutions

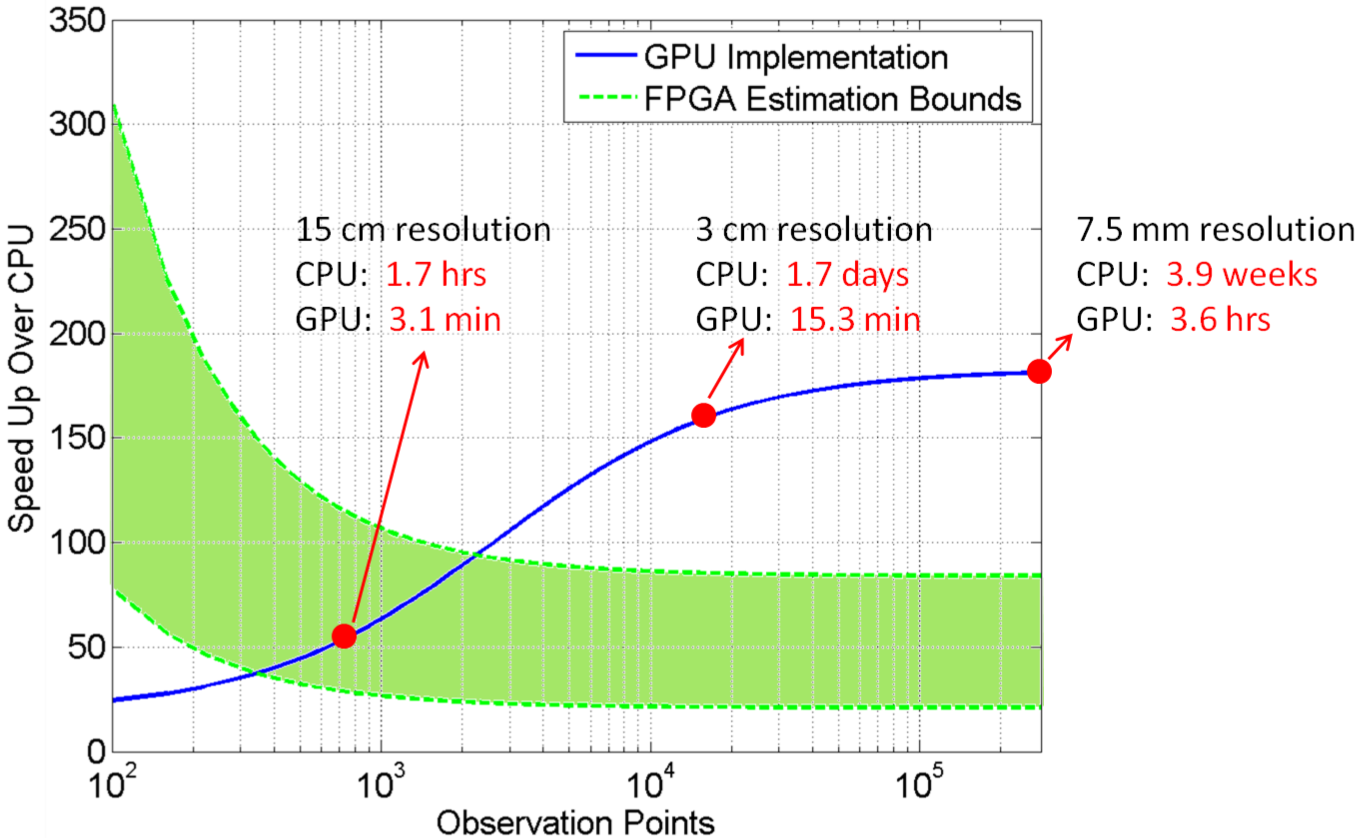


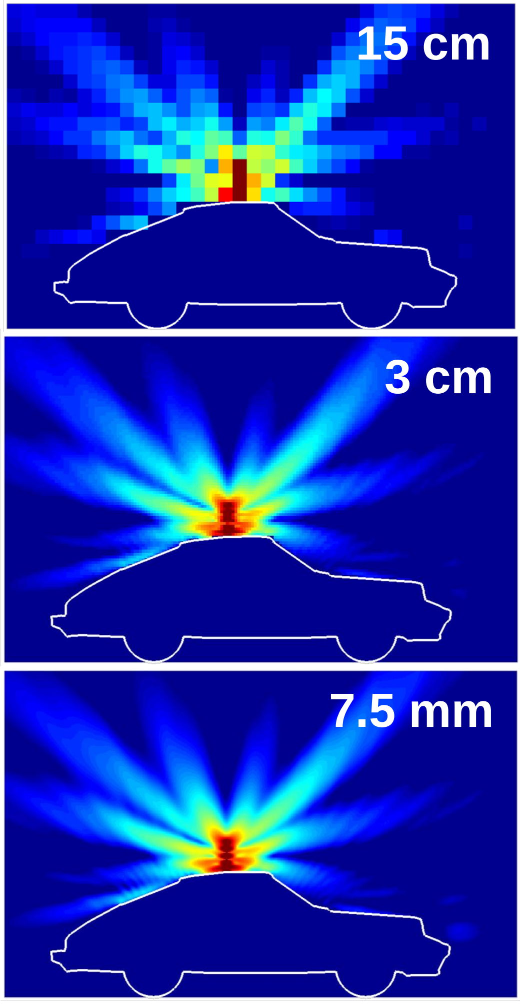
Example 3: Fun Car

7.5 mm resolution (282,242 observation points), 3.6 hrs (GPU version)



Example 3: Fun Car





CPU / FPGA times are PO time (does not include SBR time)
GPU times are total execution time (PO+SBR)
CPU time measured for single source and scaled by 41

PO = physical optics SBR = shooting and bouncing ray

Conclusions

- Parallel hardware like GPUs have been shown to accelerate electromagnetic simulation algorithms
  - Greater than **150x** for fun car cell phone antenna field profile
- The ability to compute high resolution field profiles has many applications
  - Communications planning
  - Signal exploitation
- Pushing simulation algorithms closer to real time will allow deployed systems to maintain operability while being more easily reconfigured based on changes in the environment

Future Studies

The fun has only just begun ...

- Exploit multiple FPGA pipelines / GPU cards
- Implement the FPGA version
  – Estimation is fun, but implementation is truth
- Research ways to combine FPGA, GPU, and multi-core processors
  – Results suggest that in some domains one may perform better than another
- Continue to bring other electromagnetic simulation algorithms closer to real time
  – Acceleration of other methodologies and hybrid techniques

Thank You!