BGCE 2018/19

CFD Workflow Acceleration with Machine Learning

Friday, June 7, 2019

TEAM

PEER
BREIER

COME
CFD

KEEFE
HUANG

CSE
Machine Learning
CFD

MORITZ
KRÜGENER

CSE
Machine Learning
HPC / Team Lead

OLEKSANDR
VOLOSHYN

CSE
Machine Learning
HPC / Database

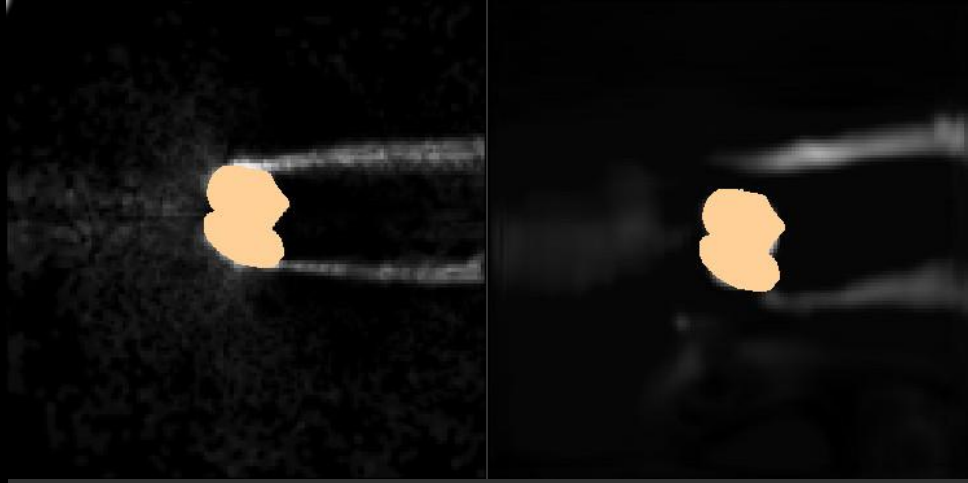
MENGJIE
ZHAO

COME
CFD

PROJECT TIMELINE

Original Goal

Up to Milestone Two
Jun 18 - Dec 18



Use machine learning to predict mesh sensitivities produced by the adjoint solver of StarCCM+

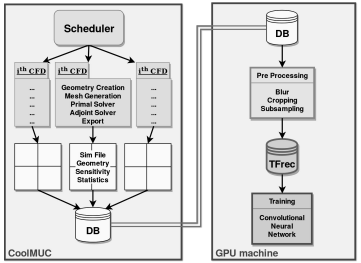
GENERATE RANDOM GEOMETRY

SIMULATE FLOW

LEARN ADJOINT MESH SENSITIVITIES

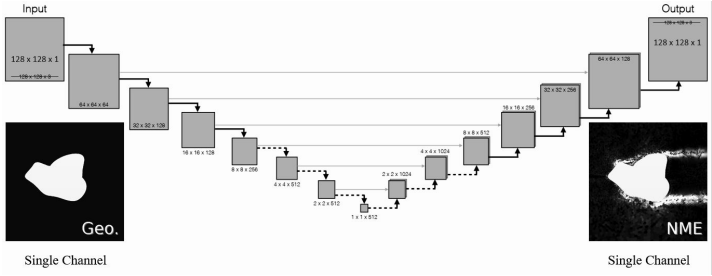
Past Milestones

Goals



RESEARCH
PLANNING
RESOURCES

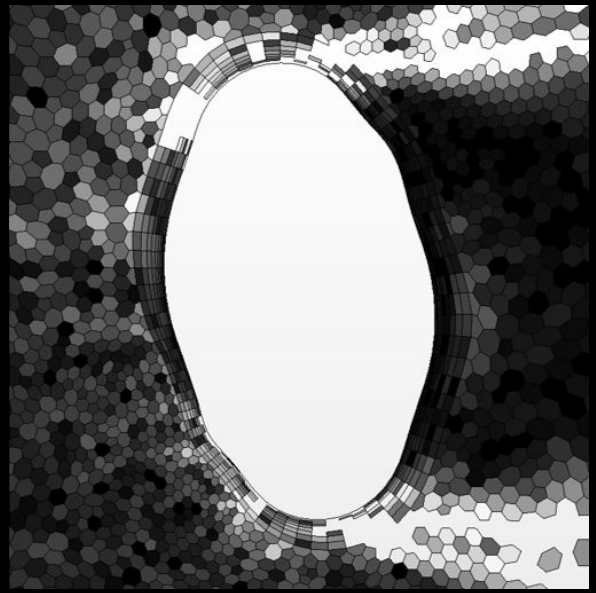
Milestone One



GEOMETRY CREATION
CFD SETUP
CLUSTER PERFORMANCE
PROOF OF CONCEPT NEURAL NETWORK
FULL PIPELINE

Milestone Two

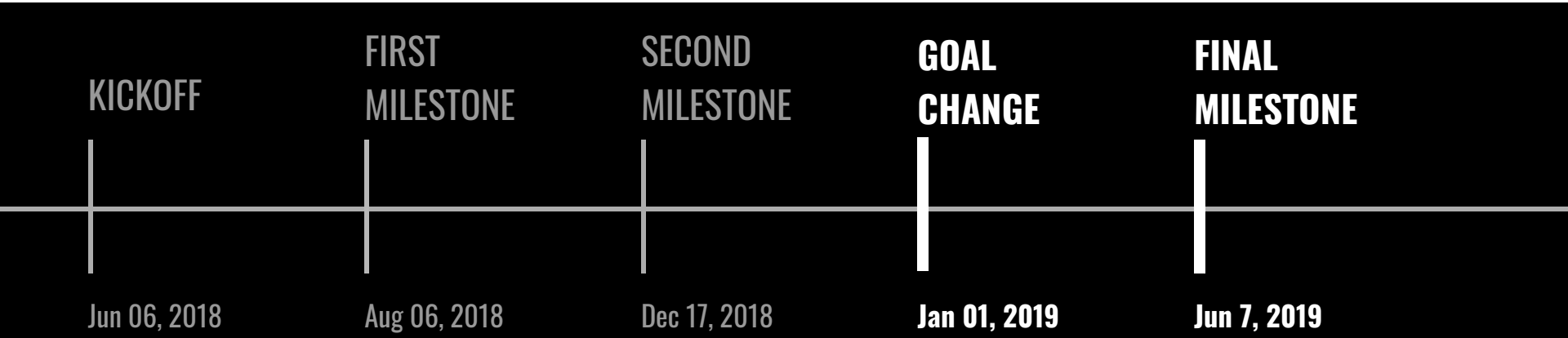
Issues with Original Goal



Thick Prism Layer

Narrow Prism Layer

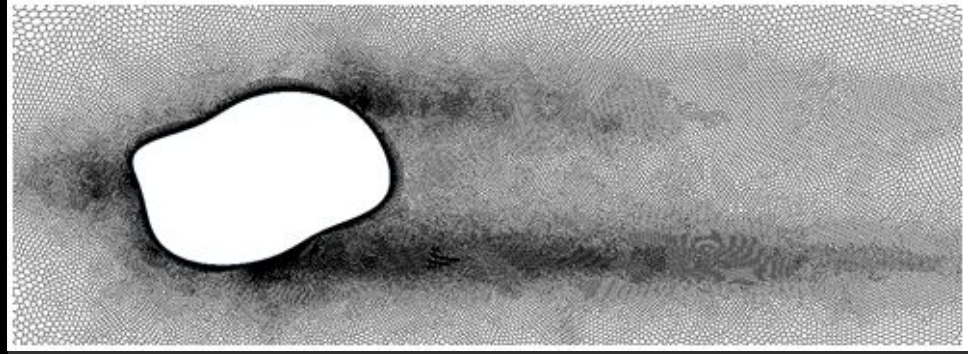
PROJECT TIMELINE



Goal Change

From Milestone Two - Today
Jan 19 - Jun 19

Use machine learning to predict
an optimized mesh
for a random geometry

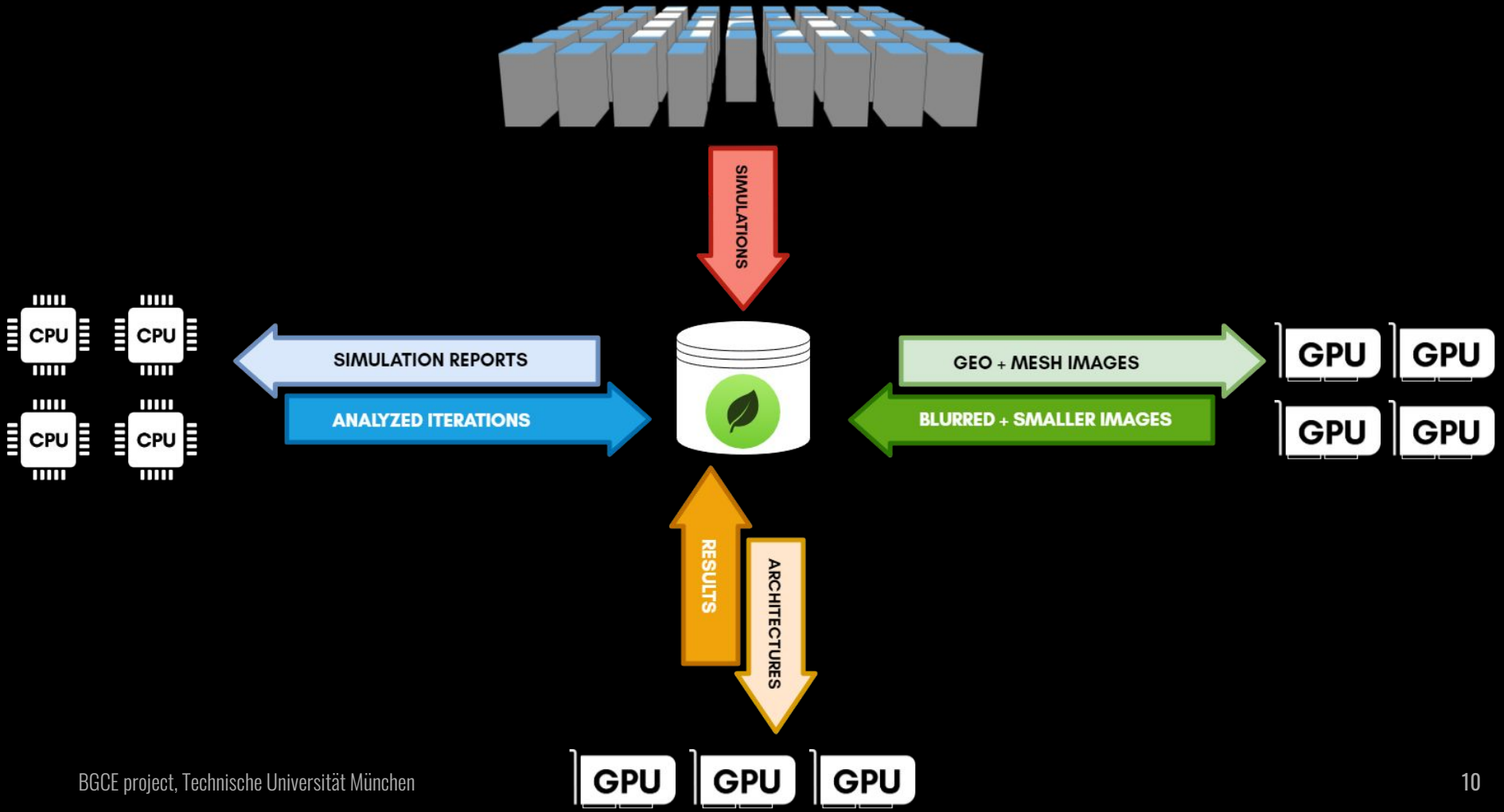


RE - USE CREATED PIPELINE

GENERATE OPTIMIZED MESH

EVALUATE QUALITY

LEARN MESH REFINEMENT MAP

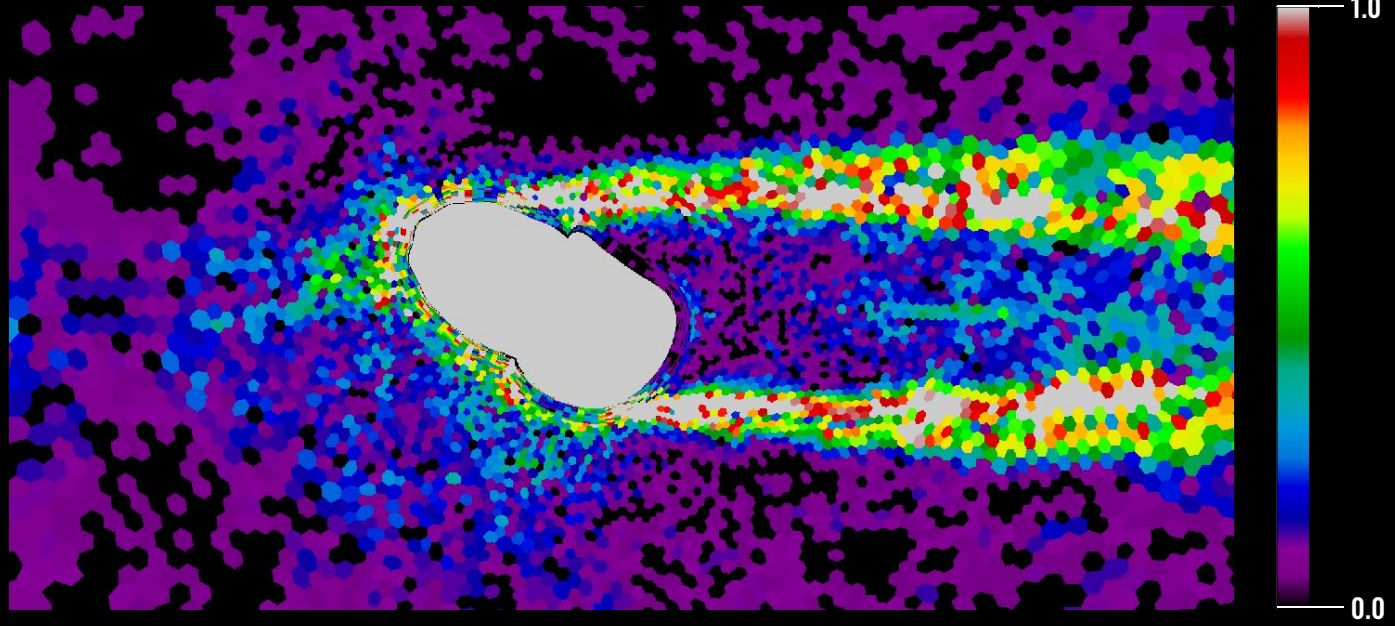

SIMULATIONS
CPU
CPU
CPU
CPU
SIMULATION REPORTS
ANALYZED ITERATIONS
GEO + MESH IMAGES
BLURRED + SMALLER IMAGES
GPU
GPU
GPU
GPU
RESULTS
ARCHITECTURES
GPU
GPU
GPU

Towards the optimized mesh

CFD Pipeline

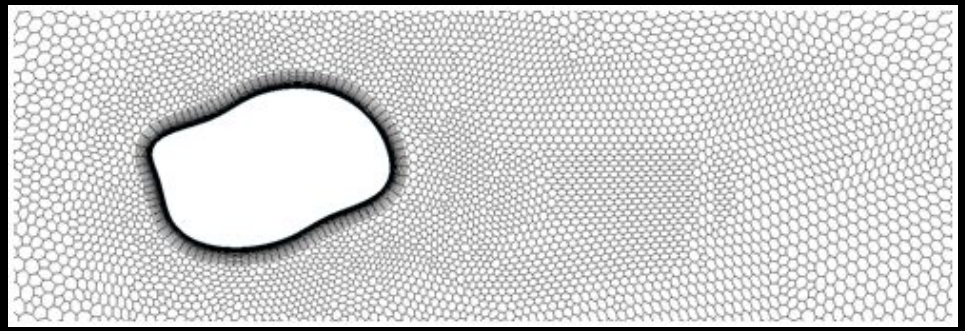
Mesh refinement with adjoint method

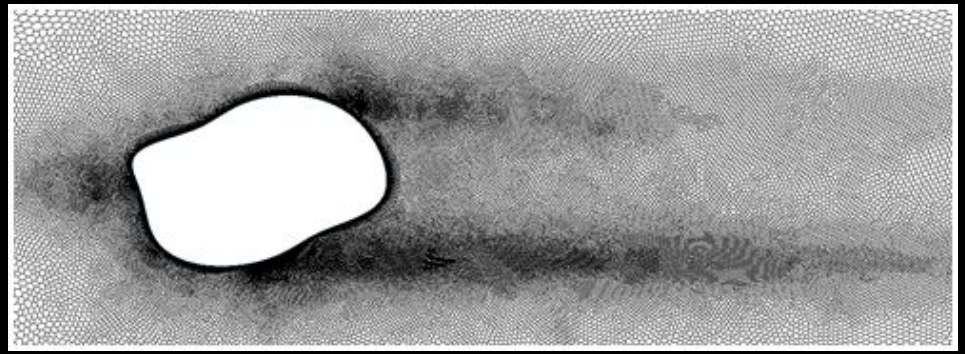
Adjoint error estimation



Adjoint solution estimates the error in each cell

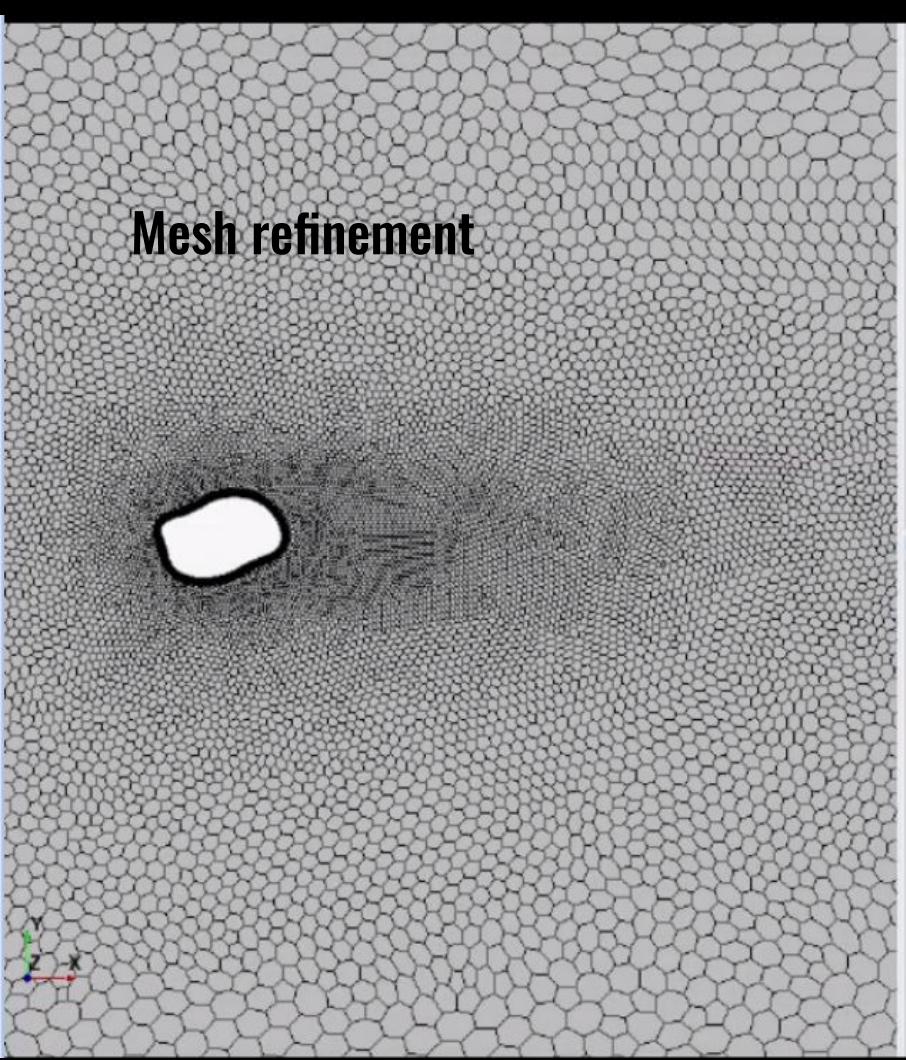
Mesh refinement

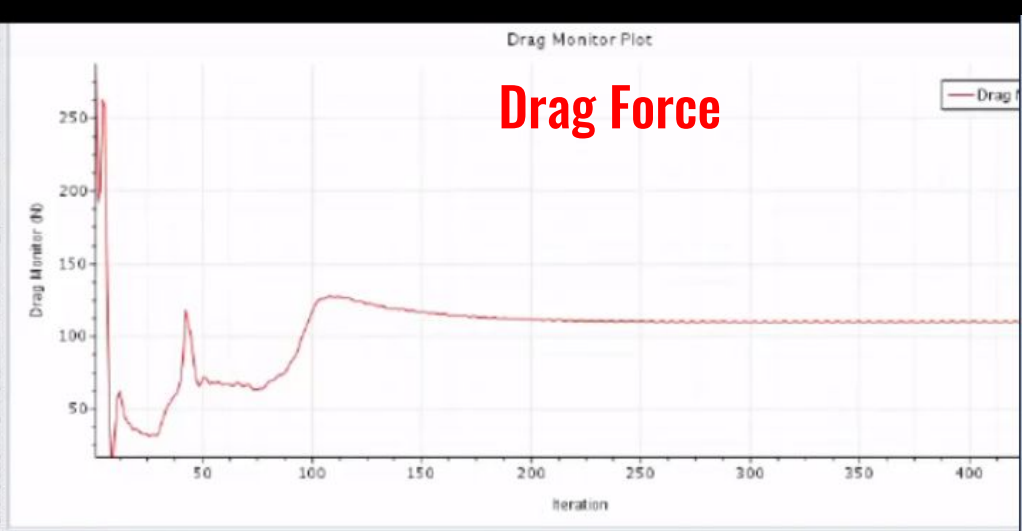


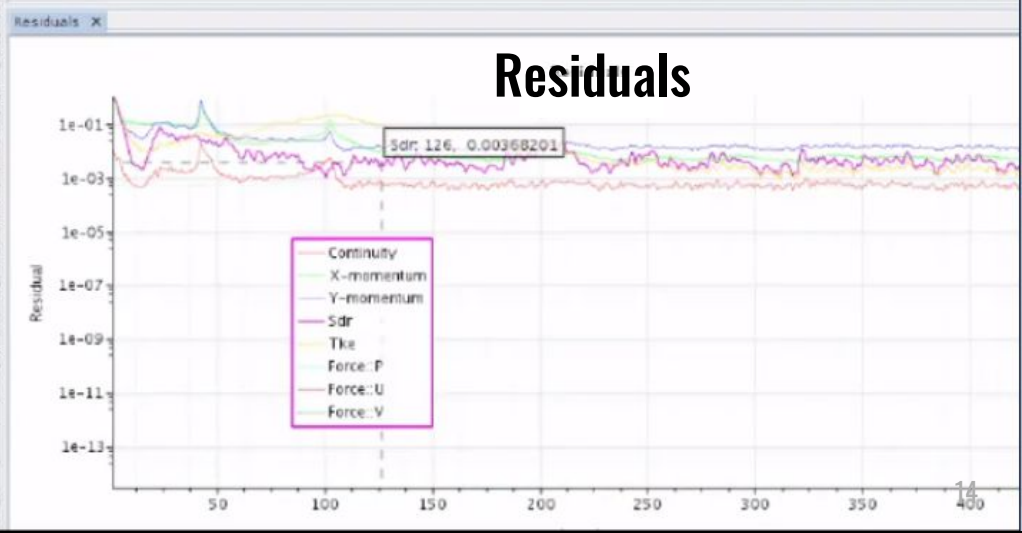


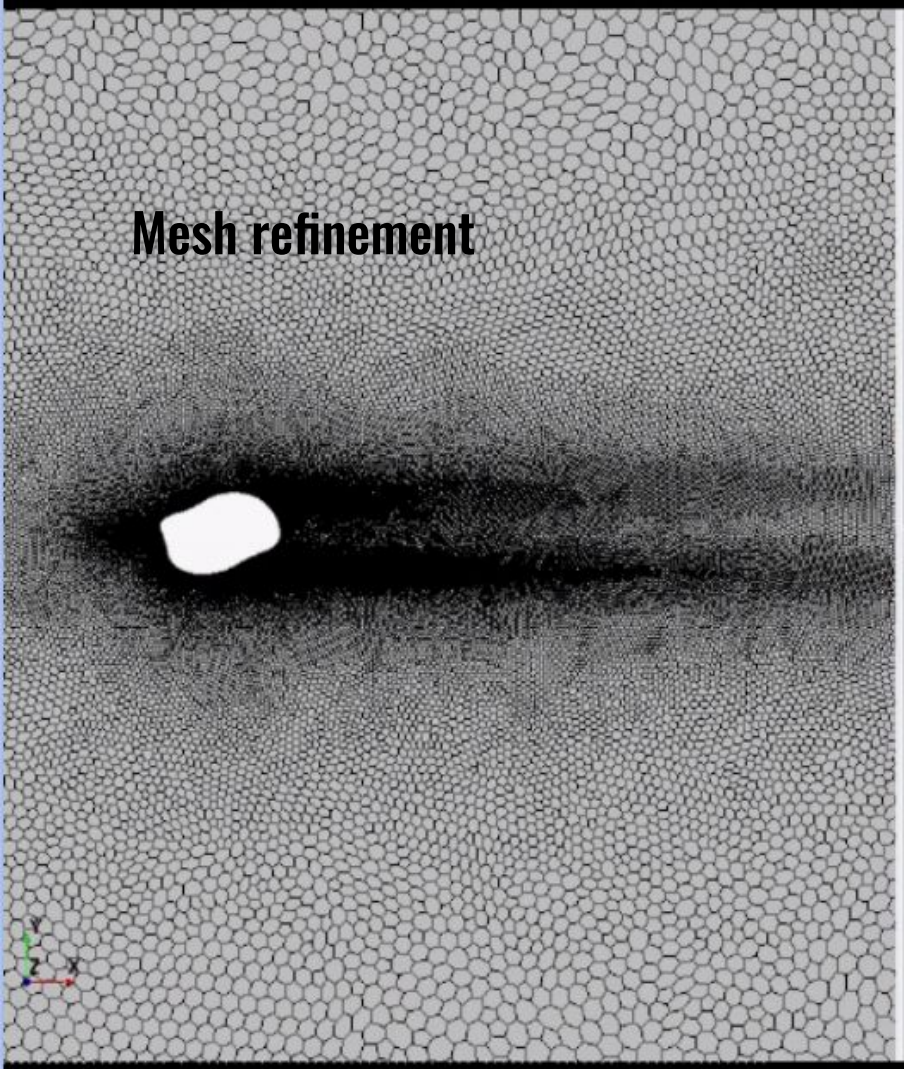
Optimize mesh density at positions of high adjoint error estimate

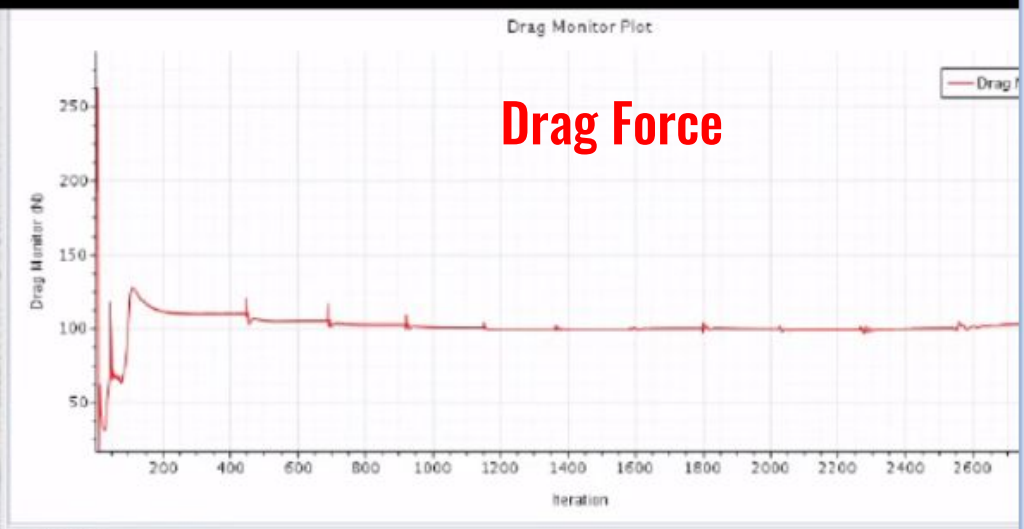
→ Refine cells if **error > threshold**

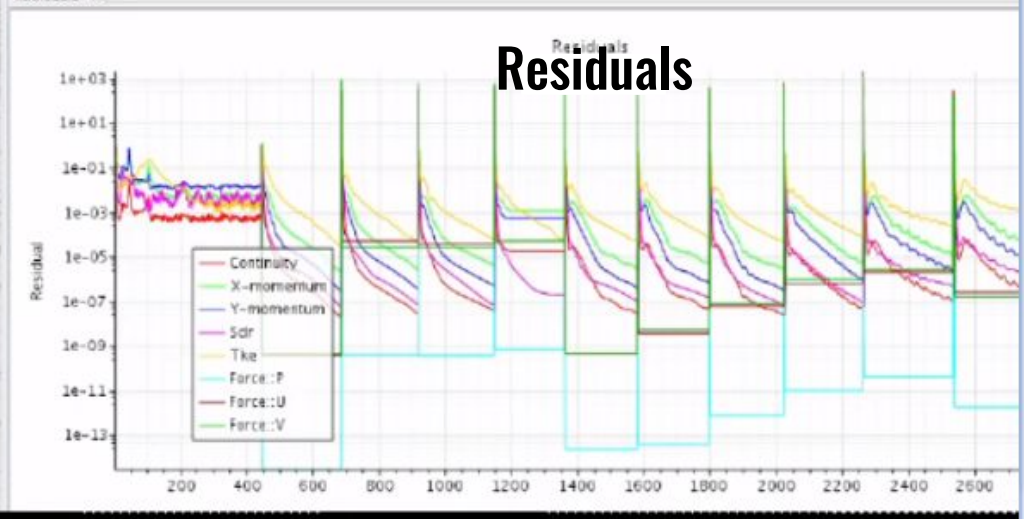

Mesh refinement


Drag Monitor Plot
Drag Force
Drag Monitor (N)
Iteration
50
100
150
200
250
300
350
400


Residuals
Sdr: 126, 0.00368201
Continuity
X-momentum
Y-momentum
Sdr
Tke
Force::P
Force::U
Force::V
Residual
1e-01
1e-03
1e-05
1e-07
1e-09
1e-11
1e-13
50
100
150
200
250
300
350
400

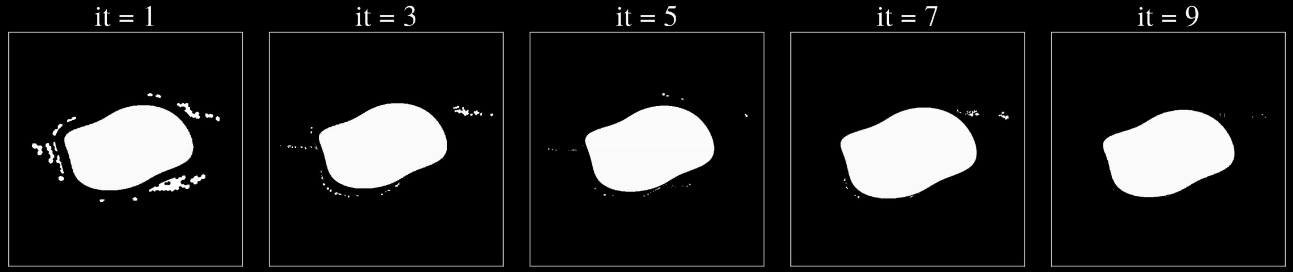

Mesh refinement


Drag Monitor Plot
Drag Force
Drag Monitor (N)
Iteration
250
200
150
100
50
200
400
600
800
1000
1200
1400
1600
1800
2000
2200
2400
2600

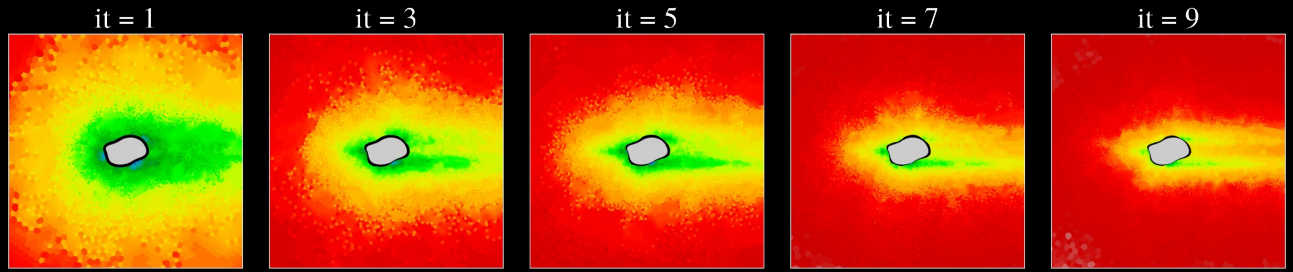

Residuals
Residuals
Residual
1e+03
1e+01
1e-01
1e-03
1e-05
1e-07
1e-09
1e-11
1e-13
Continuity
X-momentum
Y-momentum
Sdr
Tke
Force::P
Force::U
Force::V
200
400
600
800
1000
1200
1400
1600
1800
2000
2200
2400
2600

Iterative mesh refinement

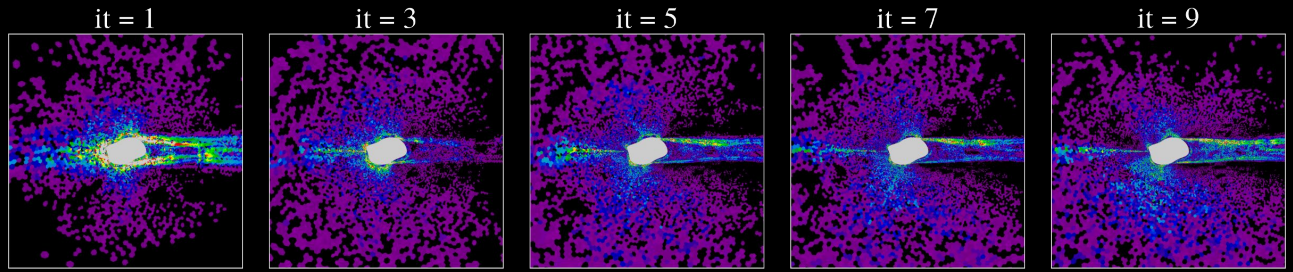
Refinement mask



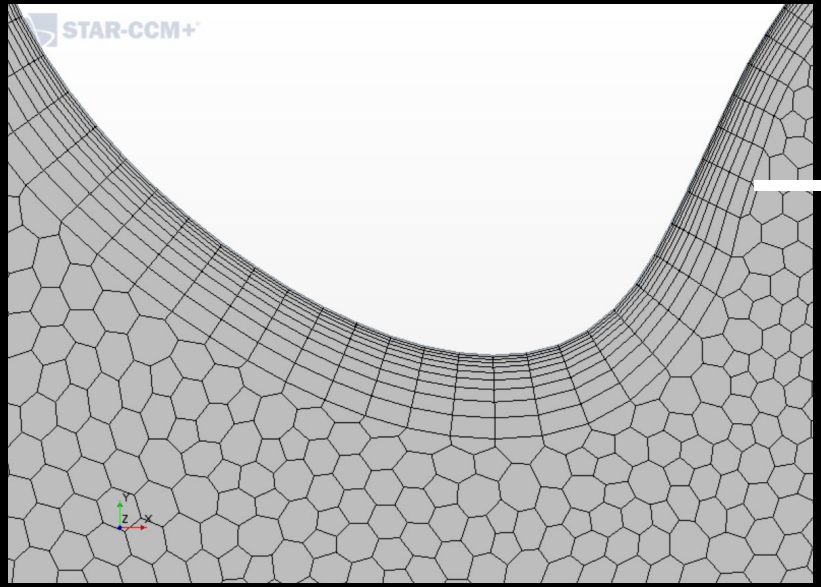
Mesh density (relative)



Adjoint sensitivity (relative)



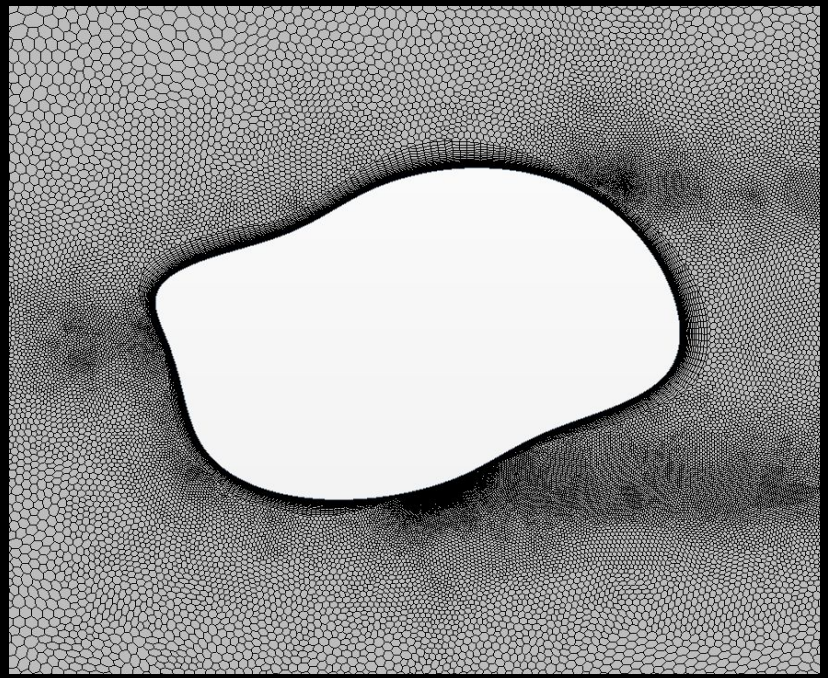
Prism Layer: Purpose



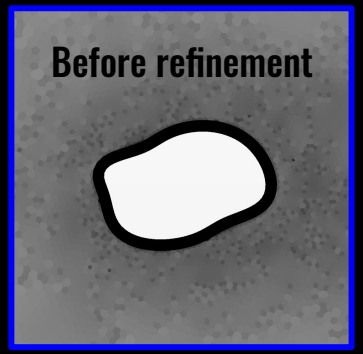
Critical for:

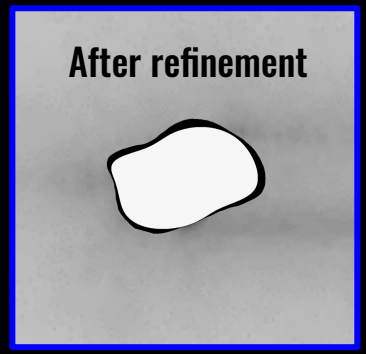
- Wall shear stress
- Separation zone
- Separation location
- Pressure and frictional drag

Prism Layer: Problems

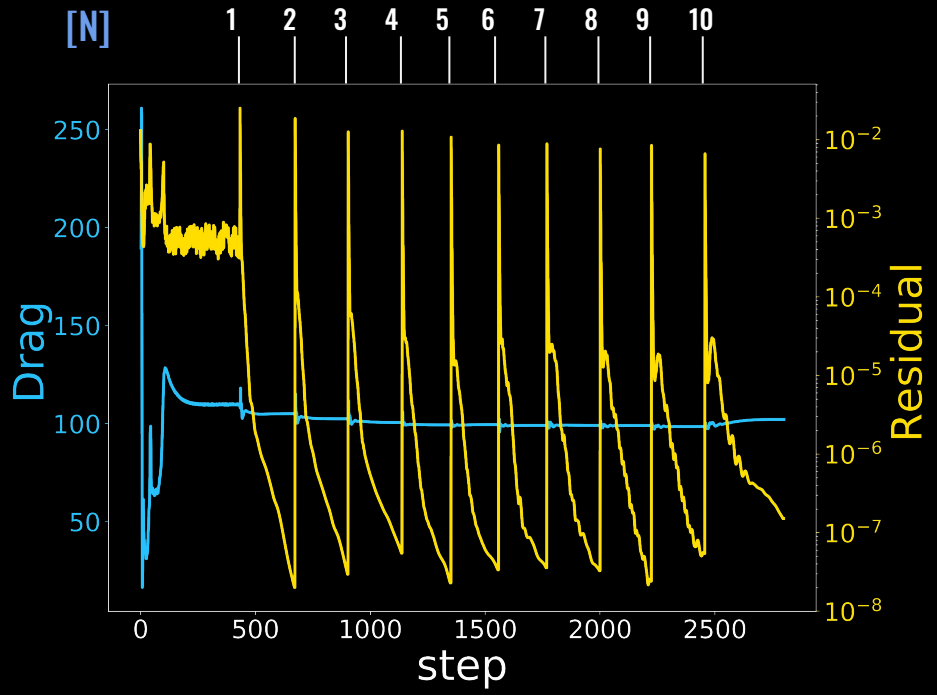


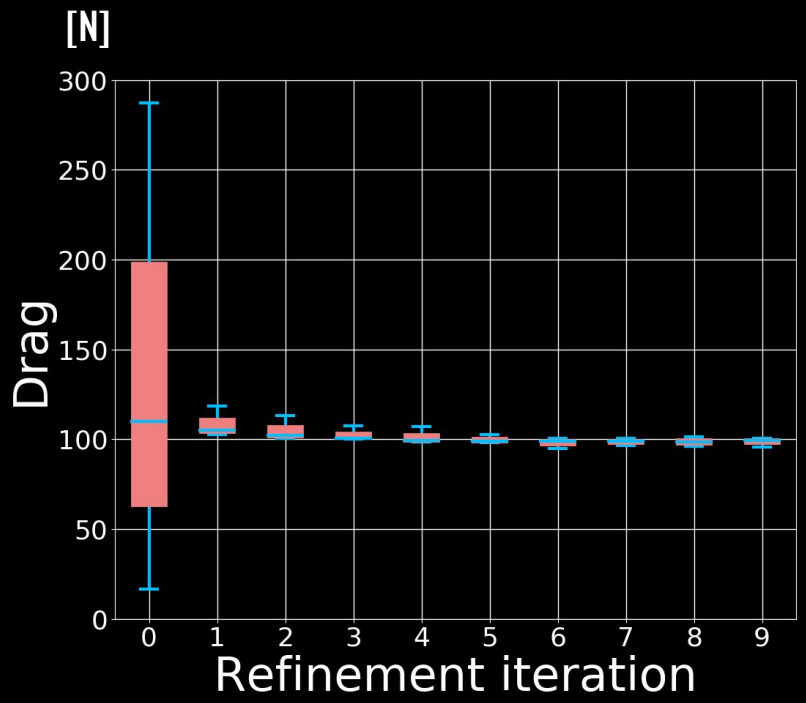
- No function available for prism layer refinement in StarCCM+
- Violation upon refinement
- Increase in total error
- Change of shape after refinement
  → Deterministic shape needed





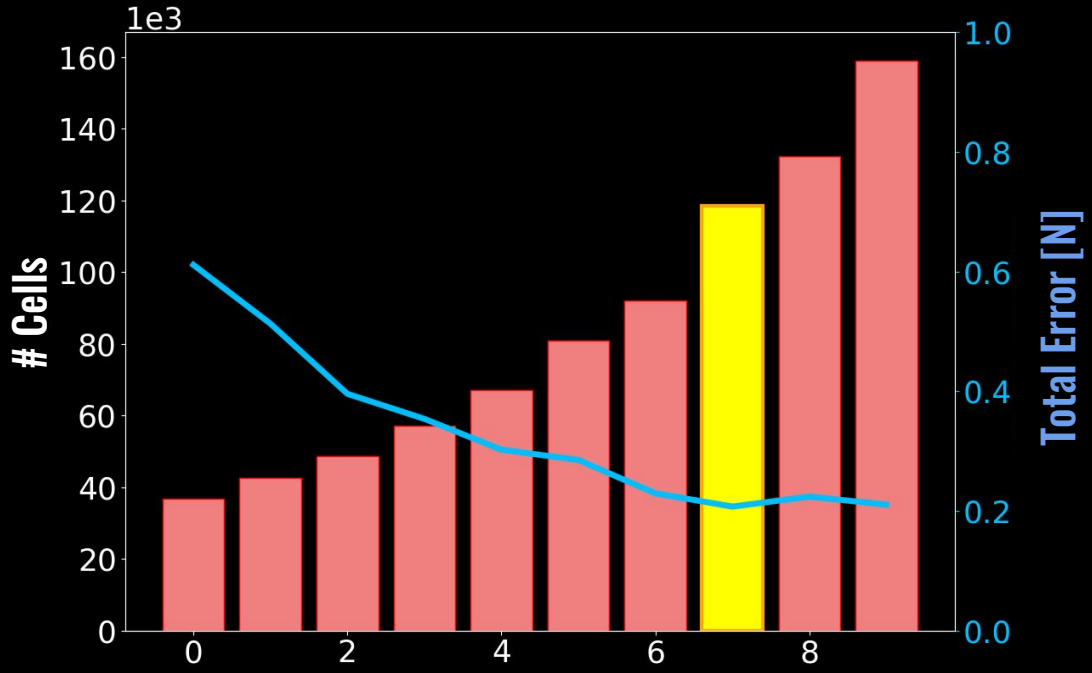
Refinement Iterations





Evaluation of mesh quality

Image Postprocessing



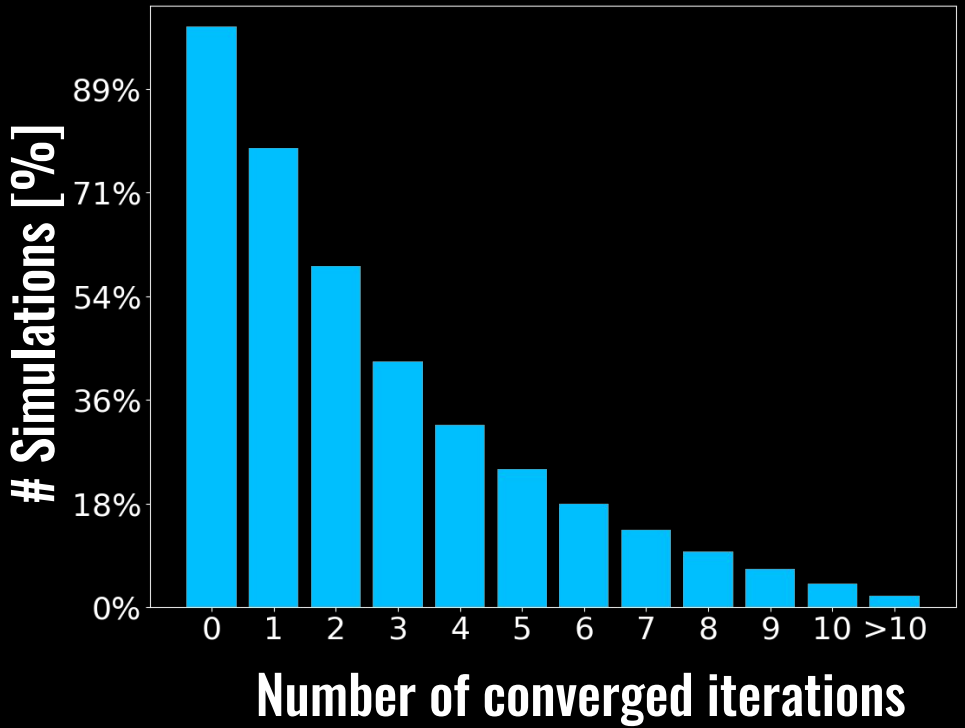
Criteria:

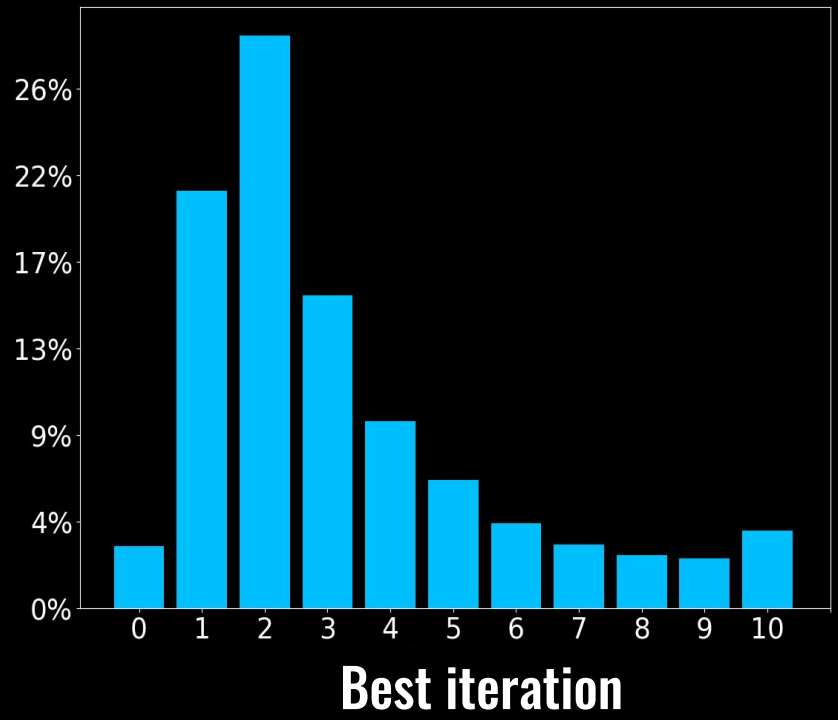
- Convergence of
  - Drag force
  - Primal solution
  - Adjoint solution
- Minimum Error

Global Statistics

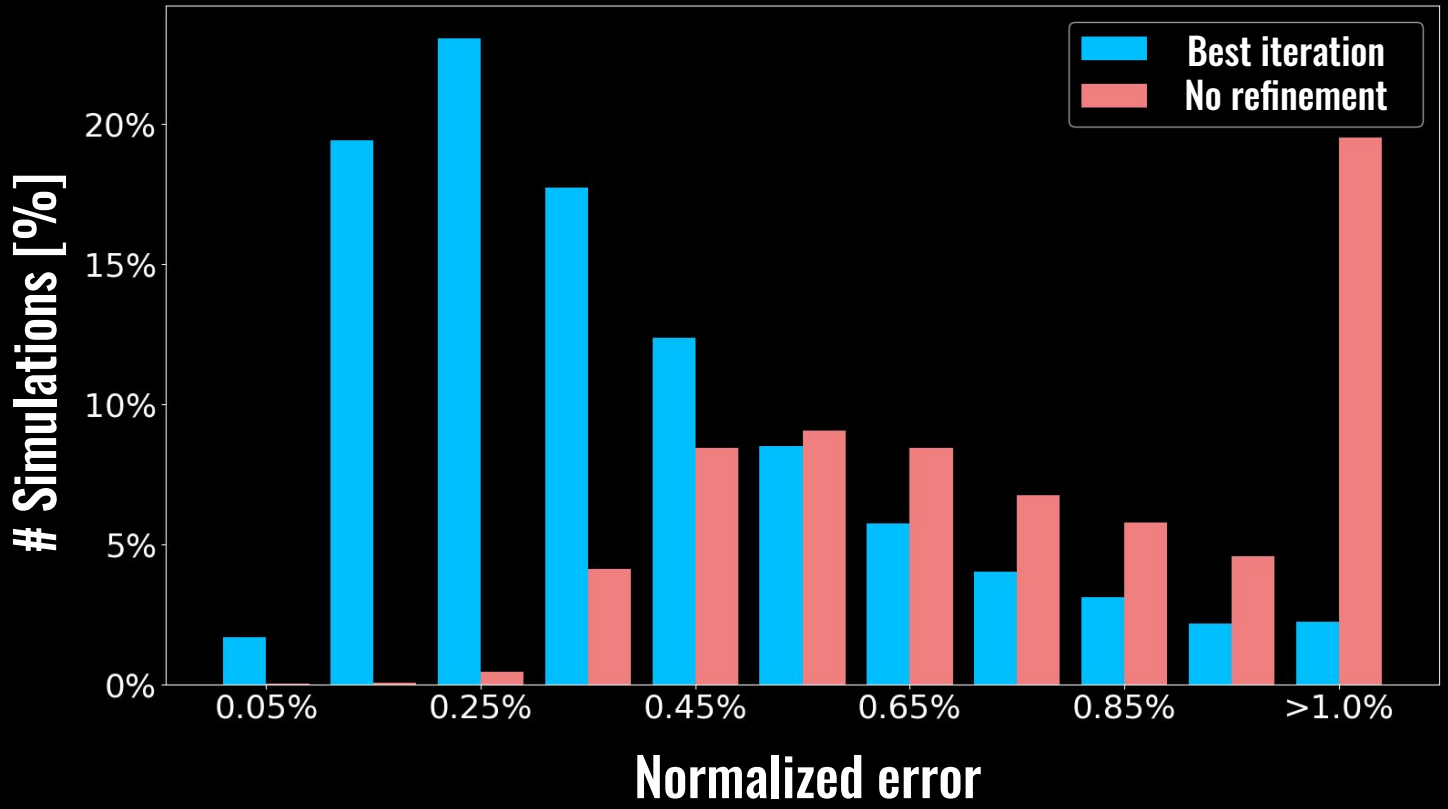
Over 60 simulations

Refinement Iterations





Refinement Iterations



Getting the best out of the data

Data Processing

Text Parsing and Image Augmentation

Managing the data

Cluster level	Up to 50 nodes simultaneously
Node level	10 simulation per node
Simulation level	4.4 iteration per simulation
Iteration level	10 pictures of physical quantities

Total data produced — **6 TB, 4 M files**

MongoDB

NoSQL: Flexible data format (JSON)

Document level concurrency

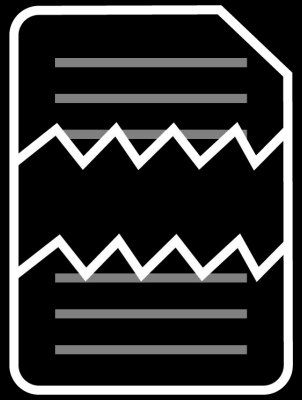
JavaScript - request language

Capable of managing big data

Built in support of replicas

Text Processing

Data Postprocessing



Scaling issues: /tmp

- Macros compilation at runtime
- Temporary log storage
- Other users data

Corrupted data — 40%

Data Postprocessing

Scaling issues: /tmp

- Discard reports with compilation error
- Identify usable iterations
- Select best iteration among available

Corrupted data ——— 40%

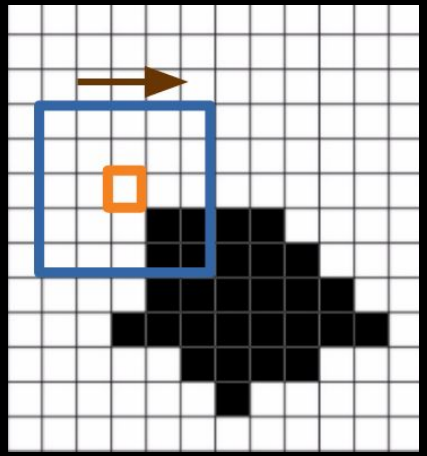
Lost data ——— 10%

Image Augmentation

Blurring

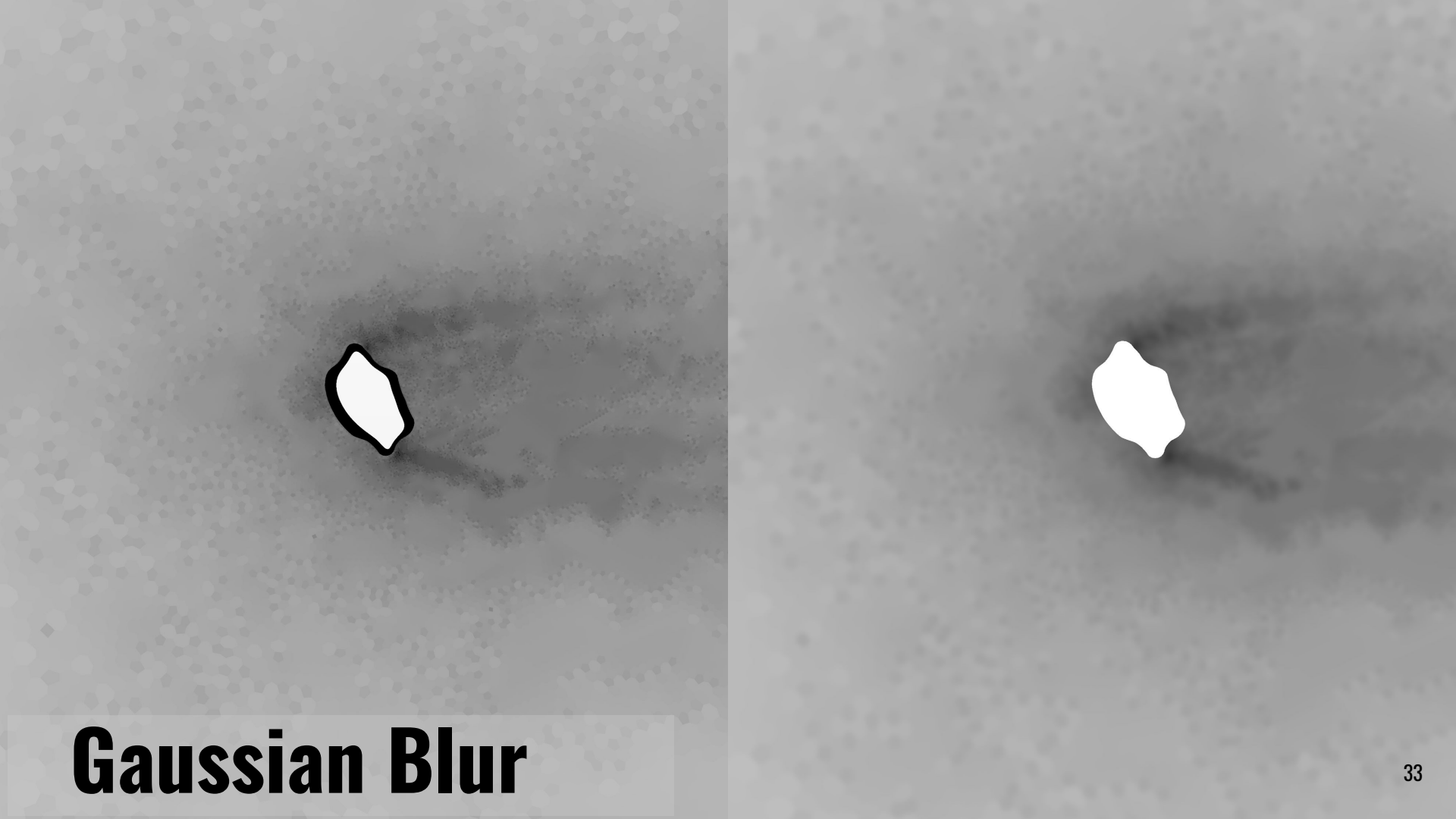
Image Postprocessing

Stride 1



$$\frac{1}{256}\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

$$\frac{1}{231}\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 0 & 0 \\ 1 & 4 & 6 & 0 & 0 \end{pmatrix}$$

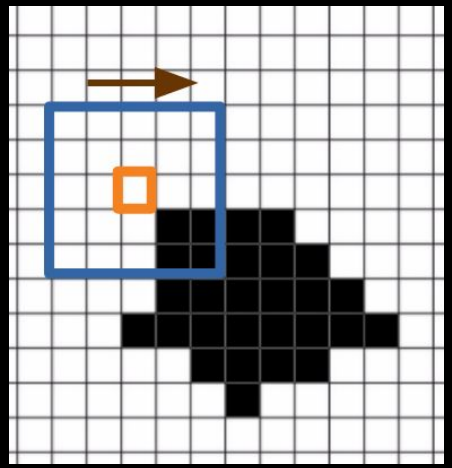


Gaussian Blur

Downsampling

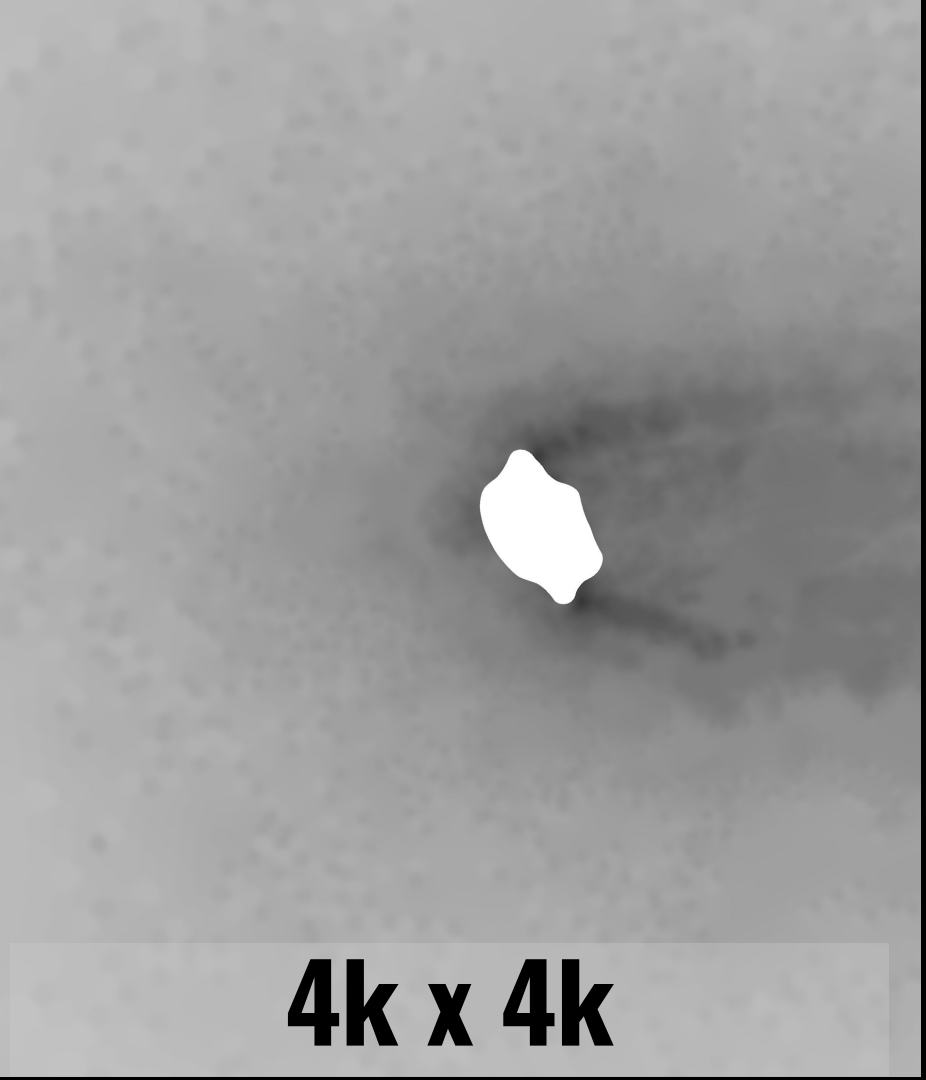
Image Postprocessing

Variable stride

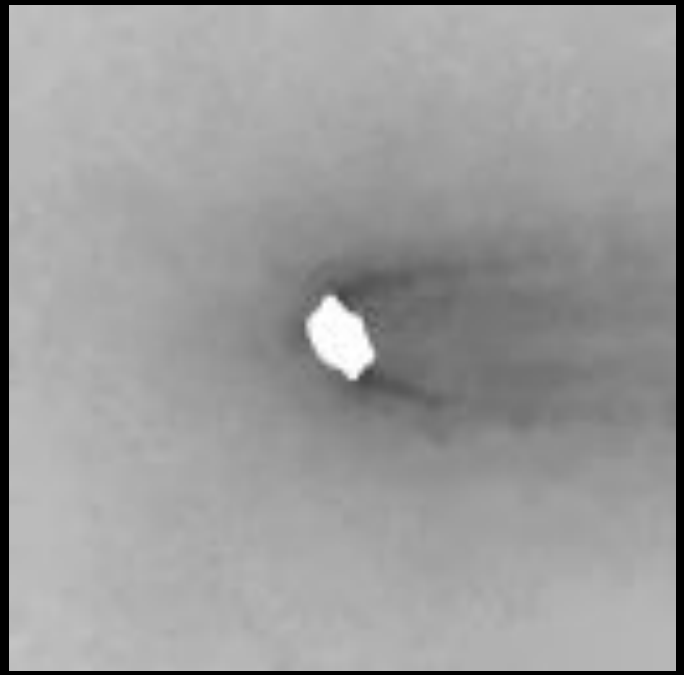


$$\frac{1}{25}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\frac{1}{21}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$



4k x 4k



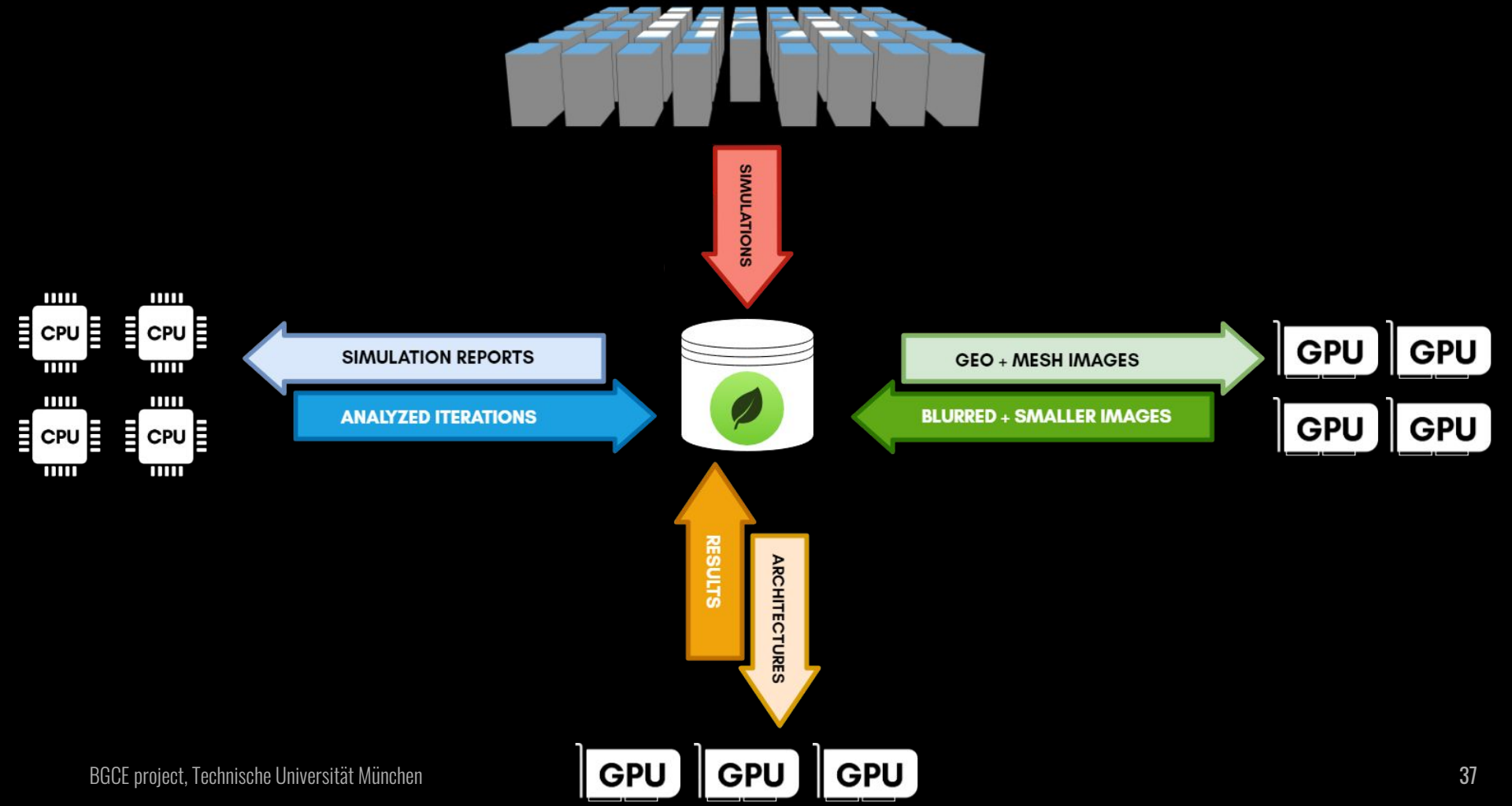
128 x 128

Image Postprocessing

- 3s per simulation
- 2 days for entire dataset

NVIDIA CUDA

- Concurrent execution
- Safe data locking
- Centralized storage accessible via http


SIMULATIONS
CPU
CPU
CPU
CPU
SIMULATION REPORTS
ANALYZED ITERATIONS
GEO + MESH IMAGES
BLURRED + SMALLER IMAGES
GPU
GPU
GPU
GPU
RESULTS
ARCHITECTURES
GPU
GPU
GPU

Learning the Mesh

Machine Learning

Optimizing Neural Network Training for Mesh prediction

Neural Network Selection

3 UNet Architectures

Basic UNet Structure

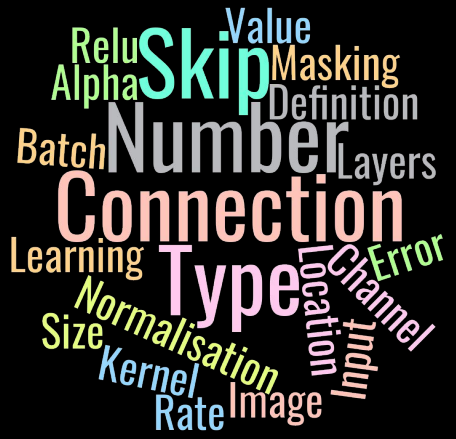
Basic UNet Structure w/ Skip Connections

Staircase Structure w/ Skip Connections

Hyper-Parameter Tuning

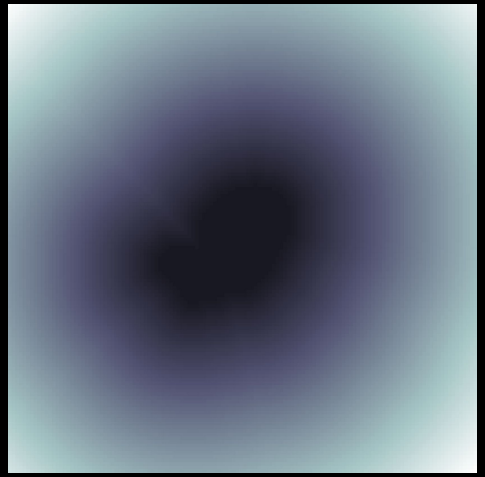
Phase 1 ——— 11 hyper-parameters

61 configurations

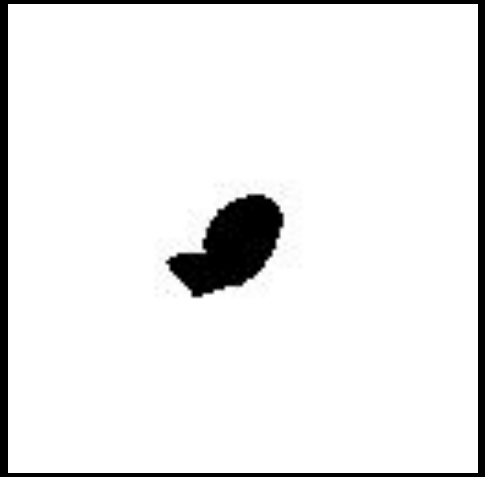


Total Compute Time ——— ½ compute month

UNet Architecture
Input Value

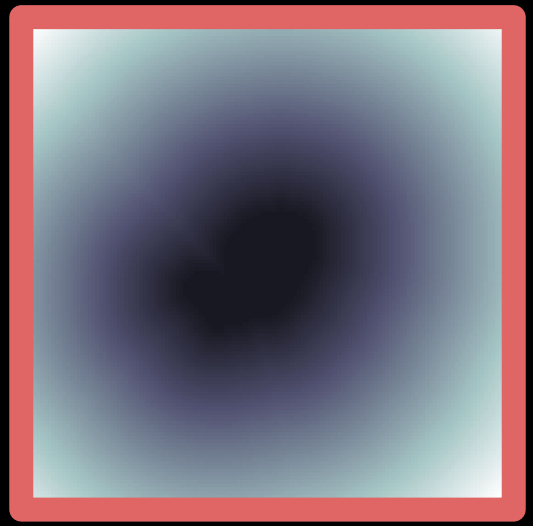


SDF

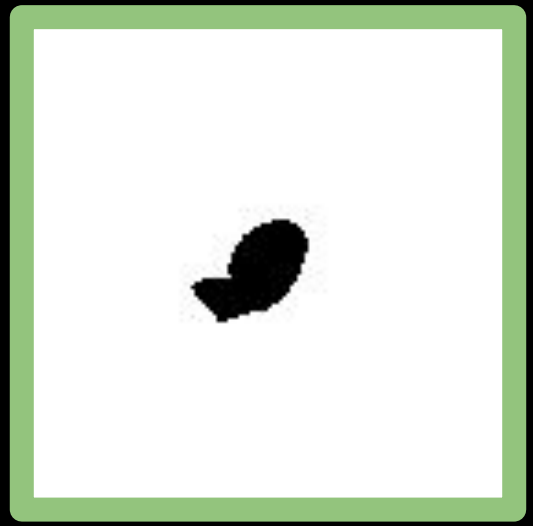


Geometry

UNet Architecture
Input Value



SDF



Geometry

UNet Architecture

Kernel Size & Channels

Sobel Operators

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Vertical Operator

$$\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

Horizontal Operator

UNet Architecture

Kernel Size & Channels

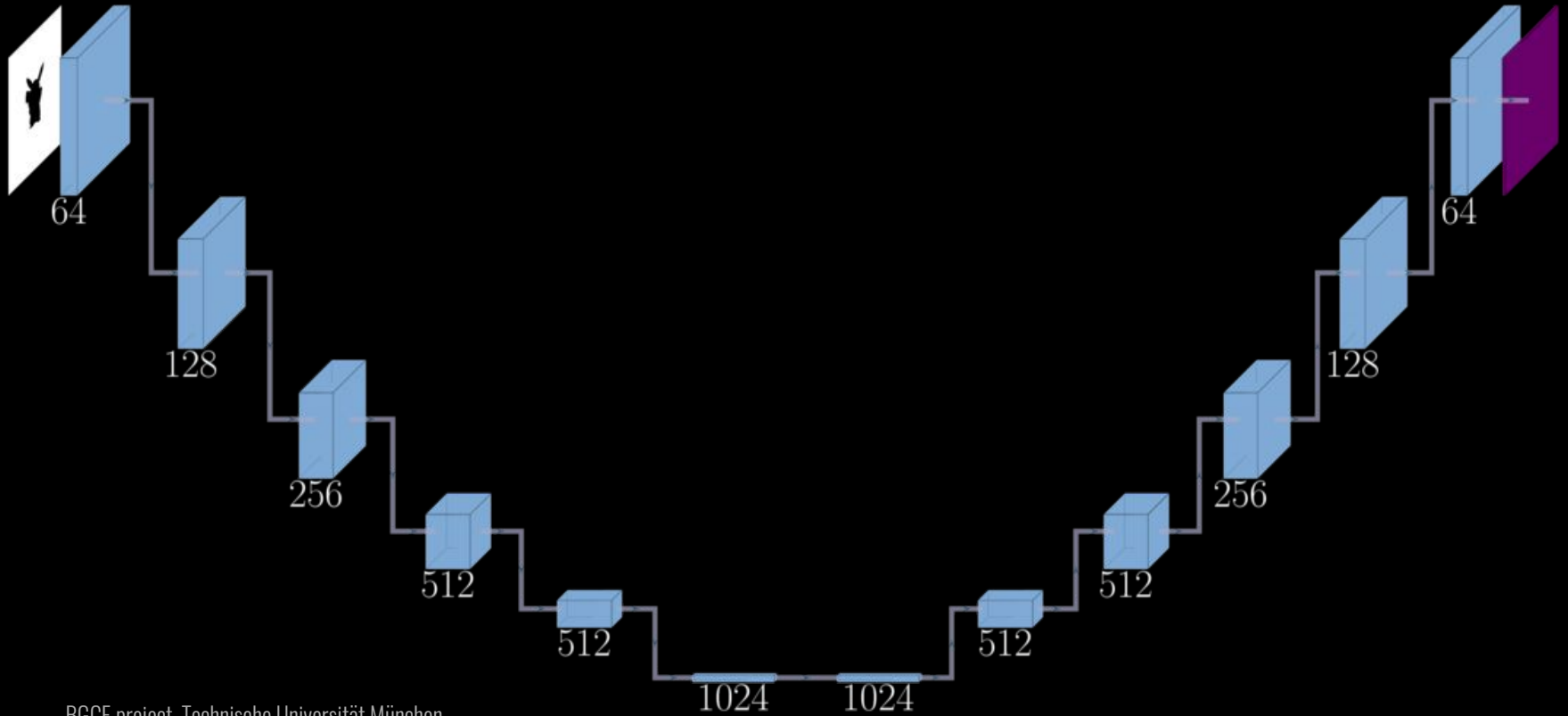
?

Trainable Kernels

$$\begin{pmatrix} x_{11} & x_{12} & \cdots \\ x_{21} & x_{22} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

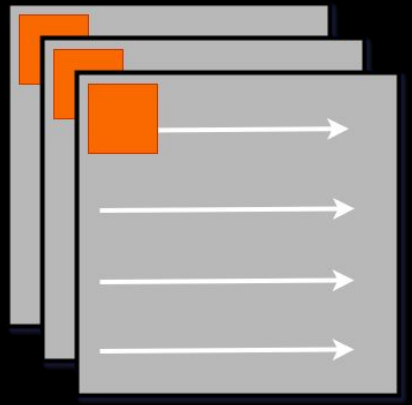
UNet Architecture

Basic UNet



UNet Architecture

Kernel Size & Channels

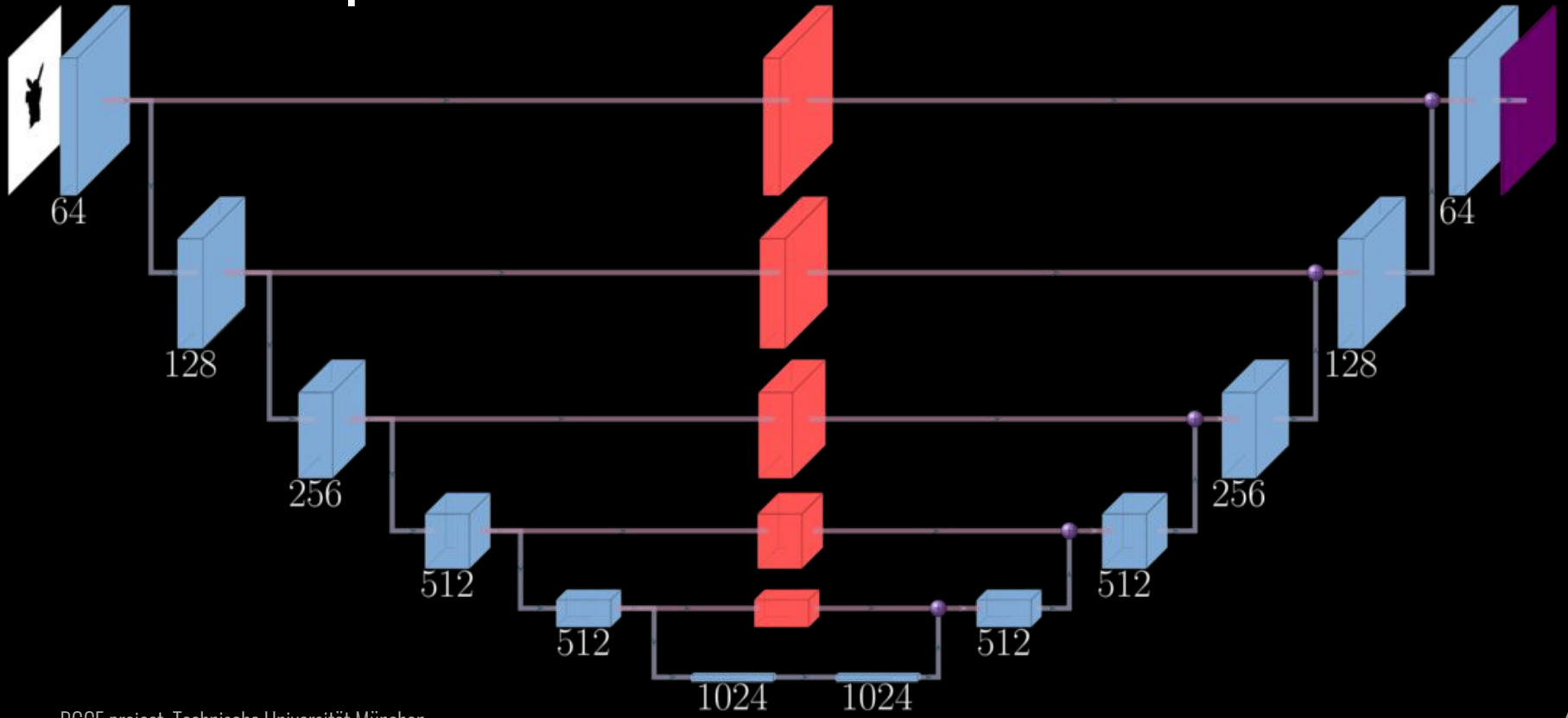


Standard Channel Number
Standard Kernel Size

Increased Channel Number
Increased Kernel Size

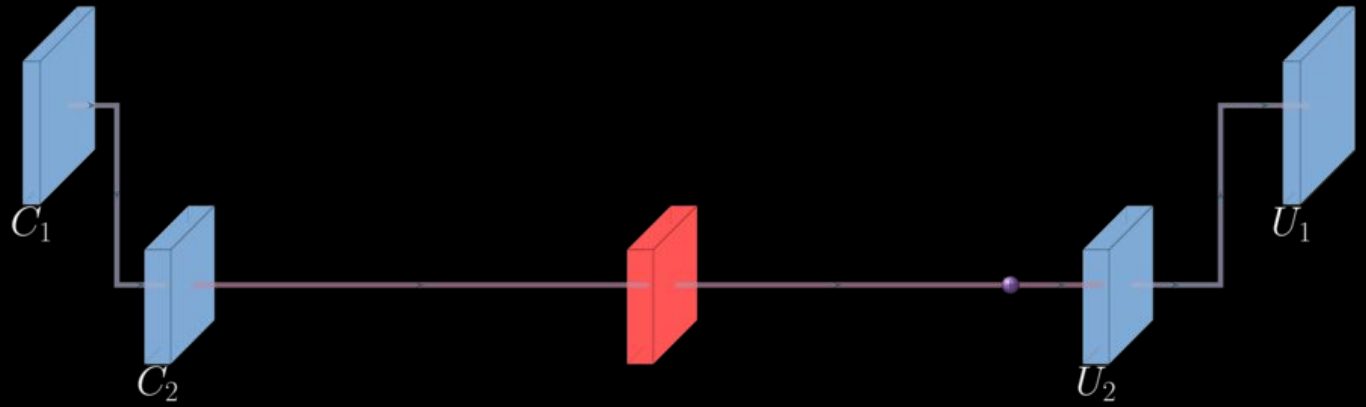
UNet Architecture

Basic UNet w/ Skip Connections



UNet Architecture

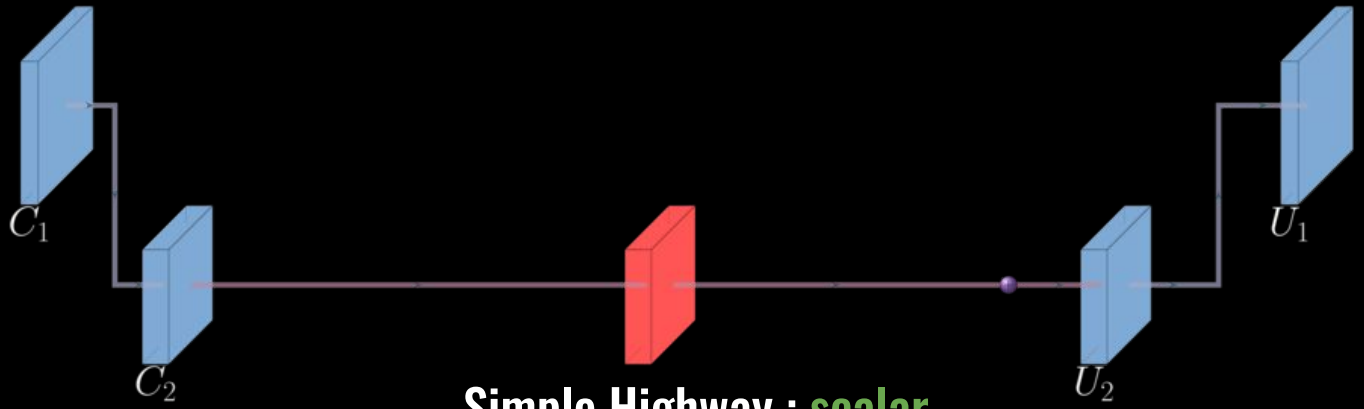
Skip Connections



Skip connections **bypass** layers to preserve gradients of upper layers

UNet Architecture

Skip Connections



Simple Highway : scalar

$$y = tC_2 + (1-t)U_2,\ t \in [0,1]$$

Tensor Highway : tensor

$$y = T \circ C_2 + (1-T) \circ U_2,\ 0 \leq\ t\ \leq 1\ \forall t \in T$$

Hyper-Parameter Training

Phase 1

Outcome:

↑ Kernel Size

↑ Channel Size (# Kernels)

? Skip-Connections

Results

<10% relative error

Hyper-Parameter Training

Phase 1 — 11 hyper-parameters

61 configurations

Phase 2 — 3 hyper-parameters

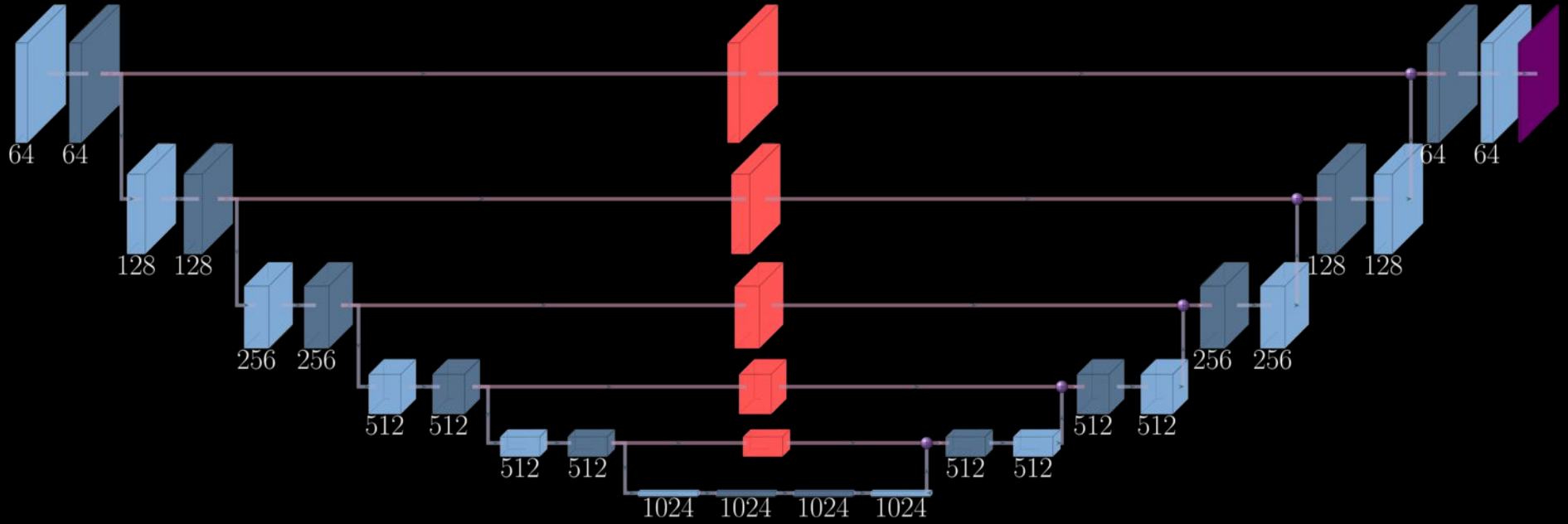
117 configurations

Adam Optimizer Beta Values
Skip Connection Type
Skip Connection Location

Total Compute Time — 1 ½ compute months

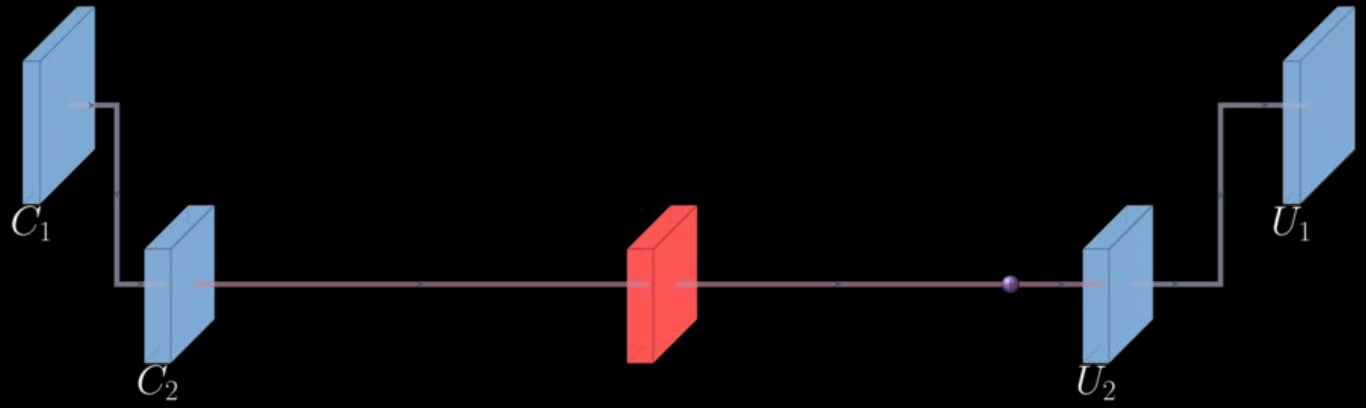
UNet Architecture

Staircase UNet w/ Skip Connections



UNet Architecture

Original Skip Connections

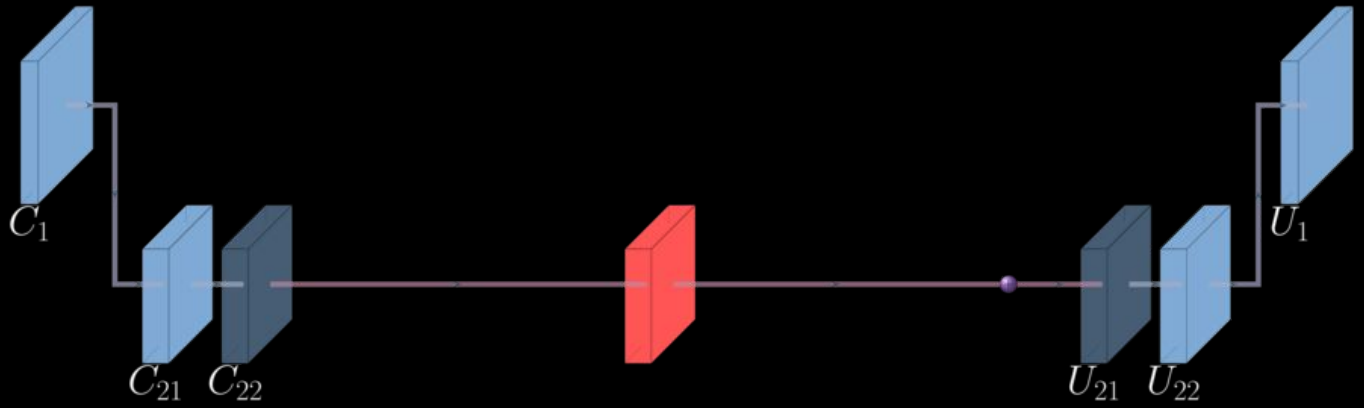


Choice of skip connection layers:

Outer Layer C_2 —————— U_2 Outer Layer

UNet Architecture

Staircase Skip Connections



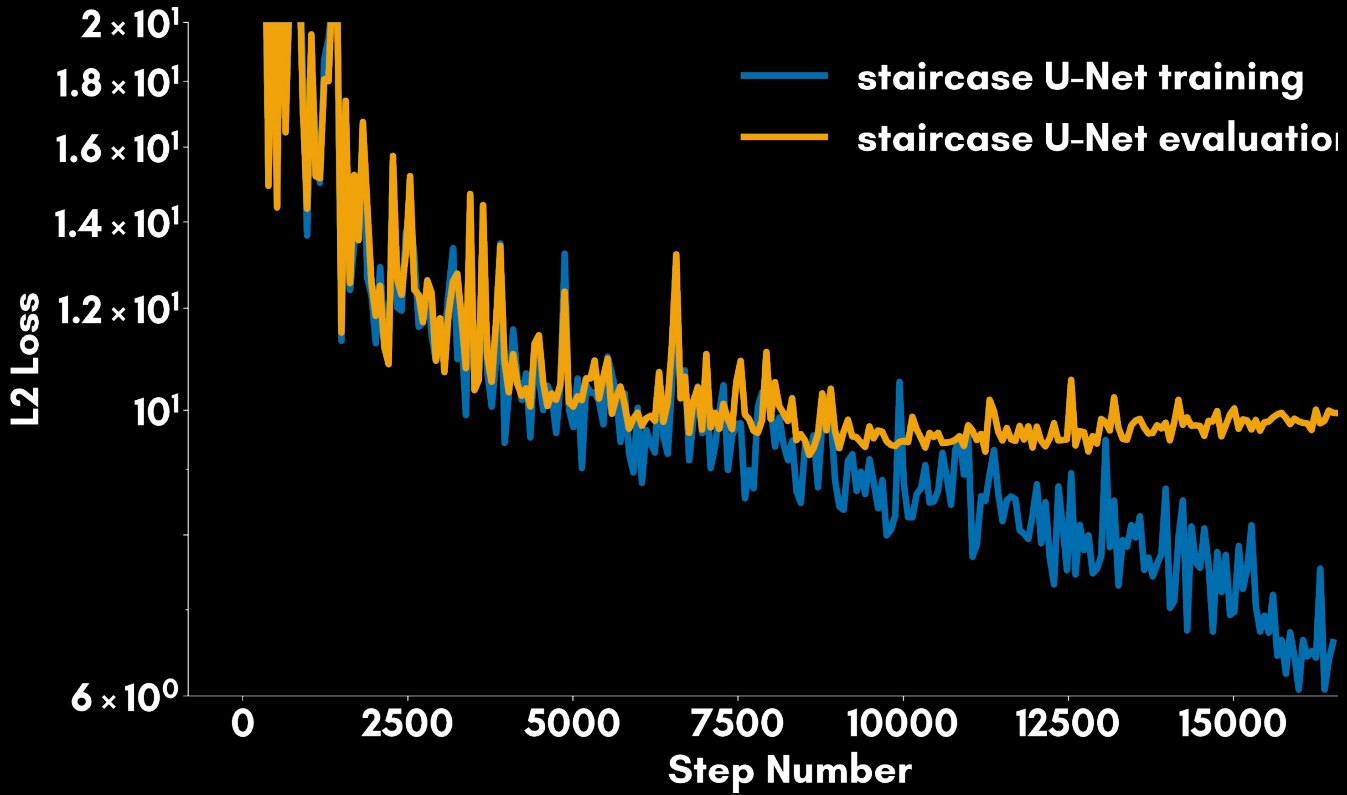
Choice of skip connection layers:

Outer Layer C_{21} —— U_{21} Inner Layer

Inner Layer C_{22} —— U_{21} Inner Layer

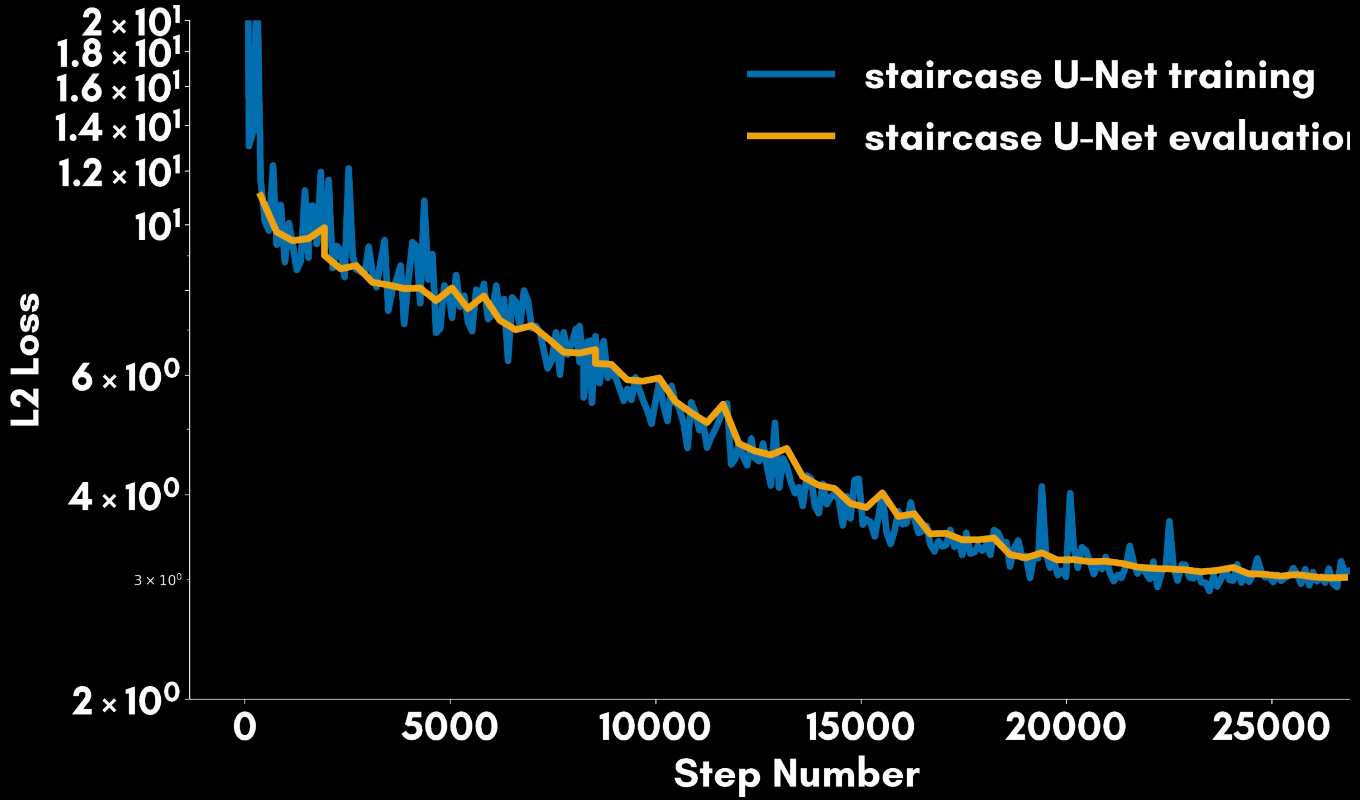
Model Training Behavior

Model 2DB600

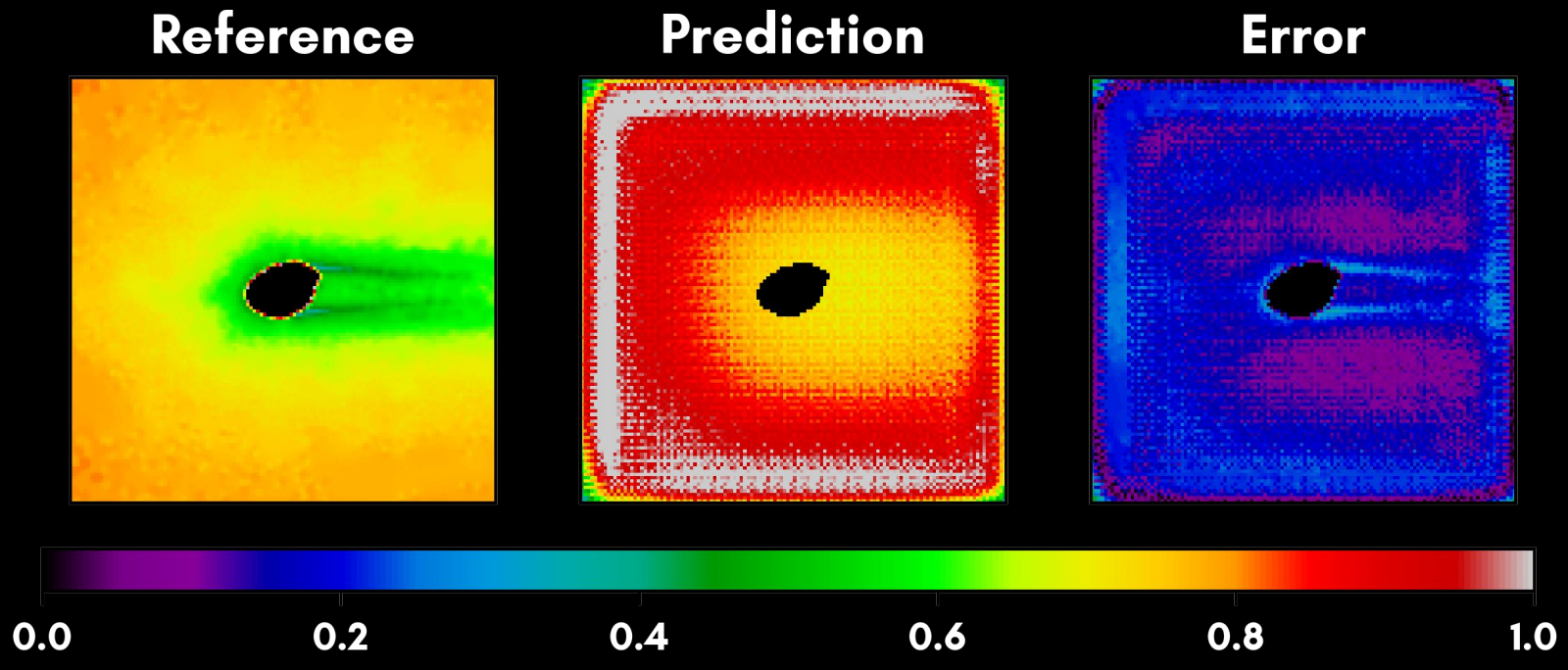


Model Training Behavior

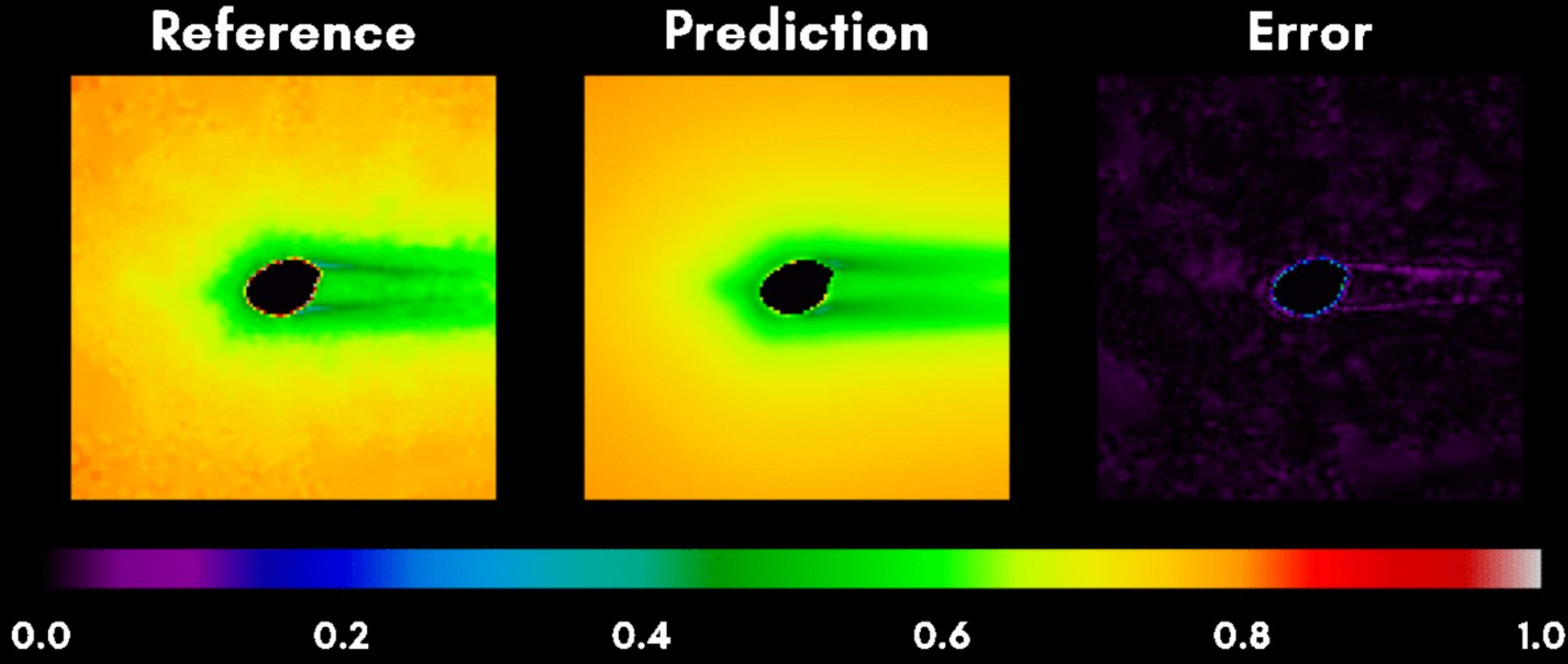
Model 2DBL000



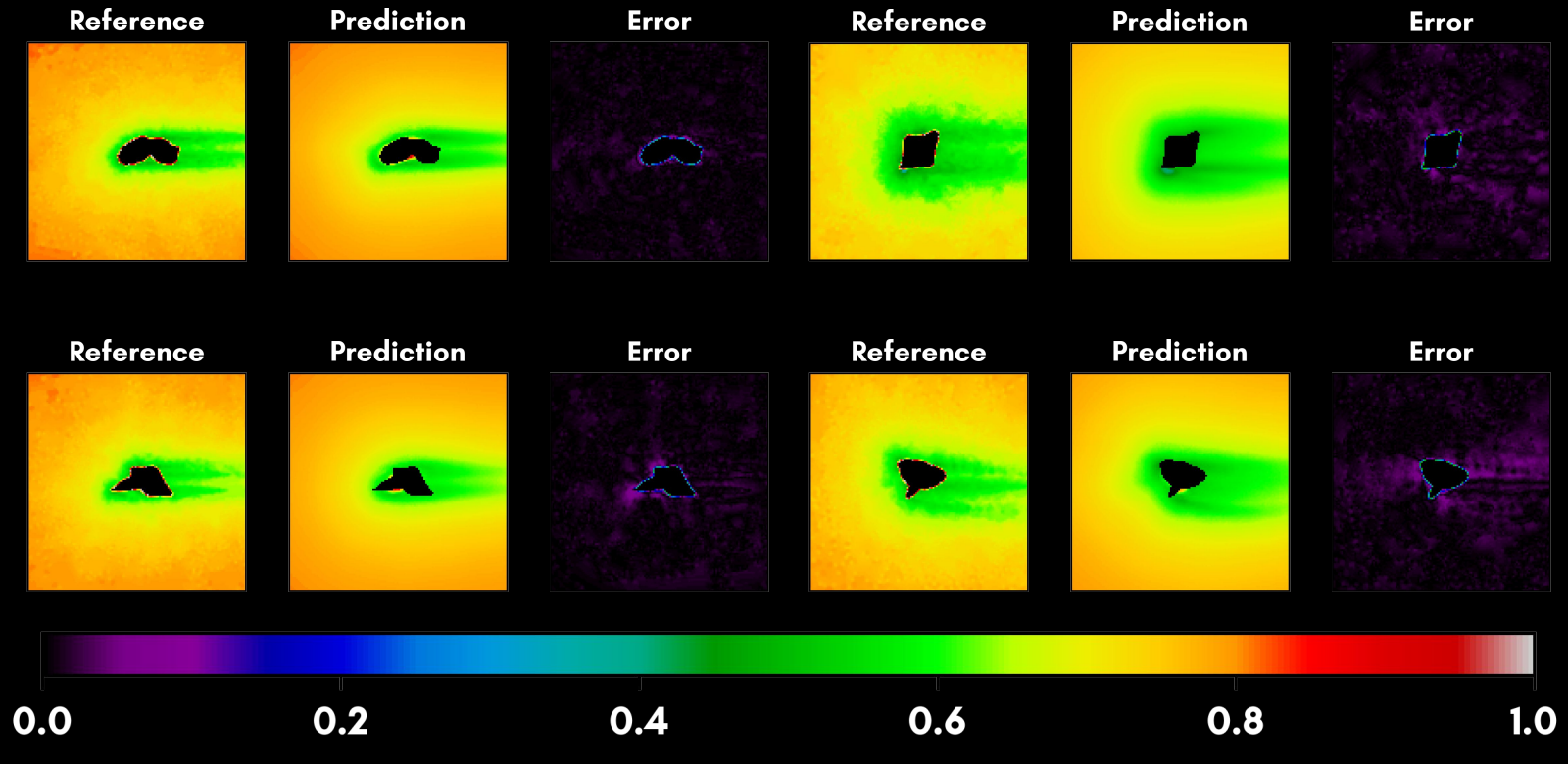
Training Results



Training Results



Training Overview



Machine Learning Results

Best Model Phase 1	Best Model Phase 2
• Model 40: Simple UNet Architecture • 211, 940, 481 DOFs • No skip connections	• 2DBL: Staircase UNet Architecture • 85, 232, 641 DOFs • Tensor Skip Connections
+ 2nd Best Accuracy >98.0%	+ Best Accuracy >98.7% + Best Performance for Size

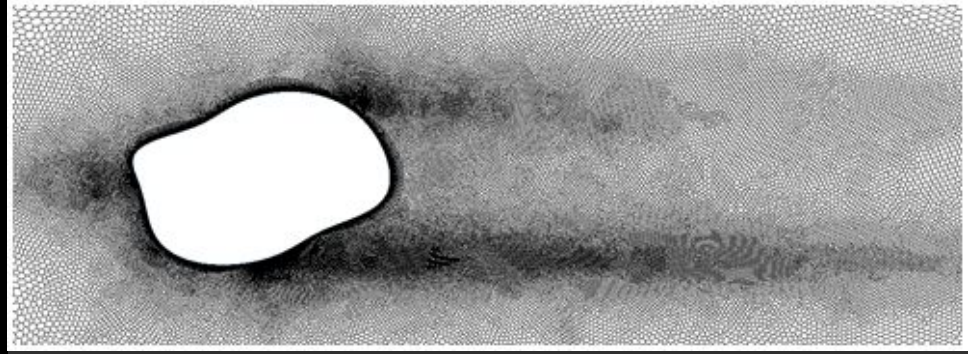
Getting the best out of the data

Conclusion

Going from nothing to a predicted mesh

Final Goal

Use machine learning to predict
an optimized mesh
for a random geometry



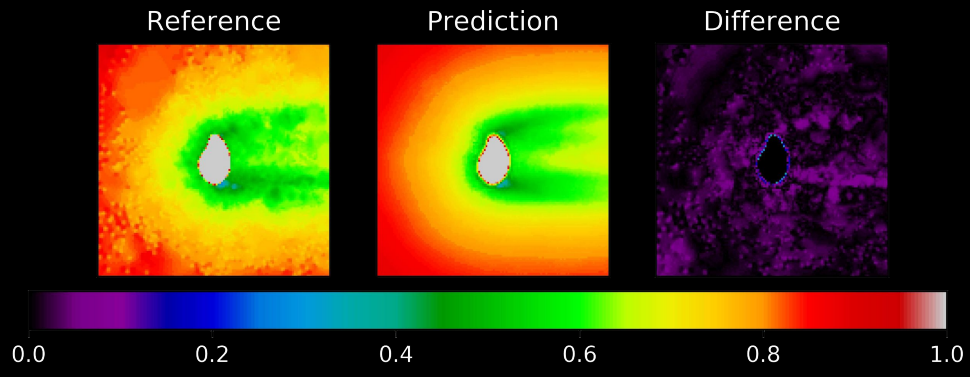
Re - use created pipeline

Generate optimized Mesh

Evaluate Quality

LEARN MESH REFINEMENT MAP

Achievements



Build pipeline to create
and process simulation data
at scale to predict mesh densities
with high accuracy

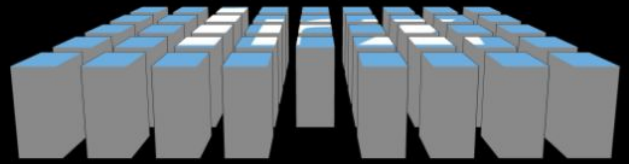
60k Simulations processed

Over 200 Networks trained

Multiuser Processing + Training Pipeline

PREDICT MESH WITH >98.7% ACCURACY

FINAL PRODUCT



1 Uploads per Simulation:

- **Raw data** (residuals, drag etc.)
- **Images** (Geometry, Mesh)
- **Additional quantities** (Velocity / Pressure etc.)

SIMULATIONS

3 CUDA processing per best Iteration:

- **Masked blurring** (custom gaussian blur)
- **Downsample** (4k x 4k --> 128 x 128)

CPU CPU CPU CPU

SIMULATION REPORTS

ANALYZED ITERATIONS

GEO + MESH IMAGES

BLURRED + SMALLER IMAGES

GPU GPU GPU GPU

2 Report processing per Simulation

- **Analyze Iteration** (drag, #cells)
- **Select best** (convergence criterion)

RESULTS

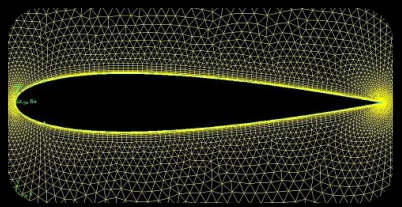
ARCHITECTURES

4 Network Training per config:

- **Train config** on selected dataset
- **Save raw**

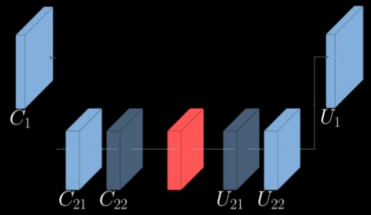
GPU GPU GPU

USE CASES



FIRST GUESS MESH

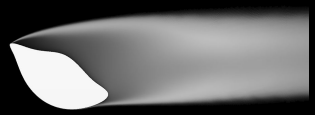
Before ever starting a simulation, get a reasonably good prediction for a mesh without any experience



PACKAGE NEURAL NET

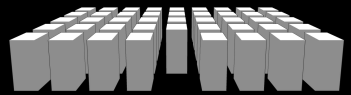
Build in trained Neural Network into e.g. Star-CCM+ as another option for mesh generation or for a first setup

LIMITATIONS



FLOW SETUP

Our dataset is limited to a very specific flow speed and geometry size, thus limiting predictions to similar flows and sizes of objects



GENERALITY

A more generally usable network would need much more diverse data from different regimes

Resources expended

50 YEARS

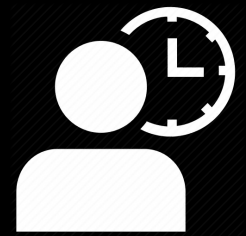
Serial CPU compute time
to produce raw simulation data
on CooIMUC2 + 3 and IvyMUC

2 MONTHS

Single GPU continuous
neural network training time

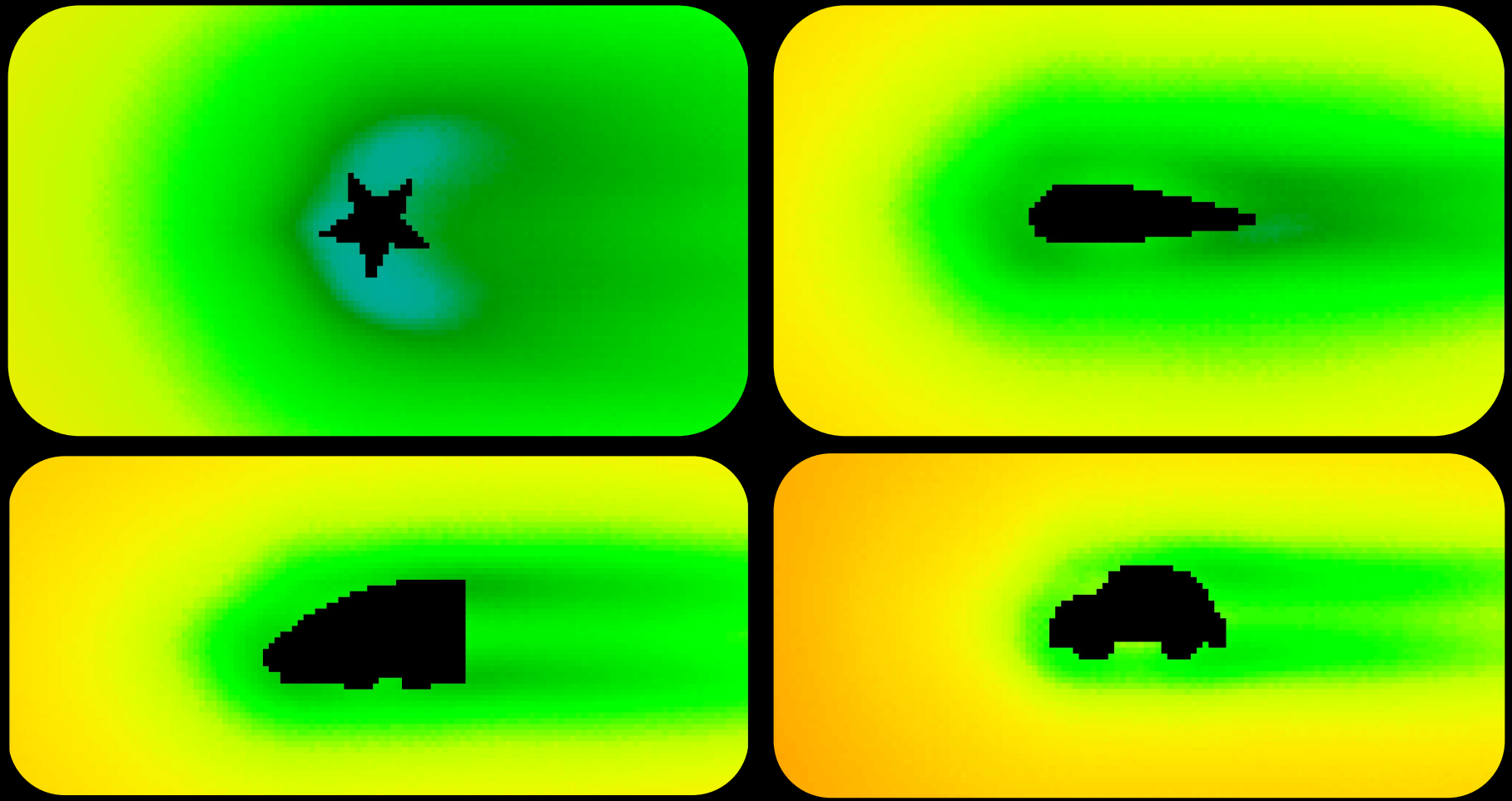
6 TERABYTES

Of data produced and processed.
That's ~60% of all of Wikipedia



2500 WORK HOURS

By our team of 5 students
over a 10 month period



Thank you for your attention