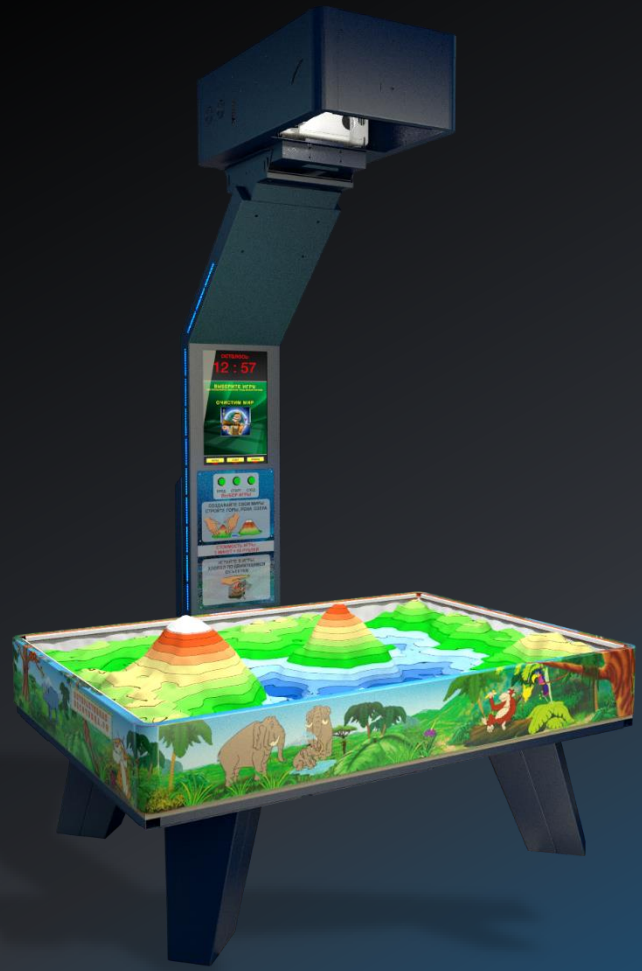
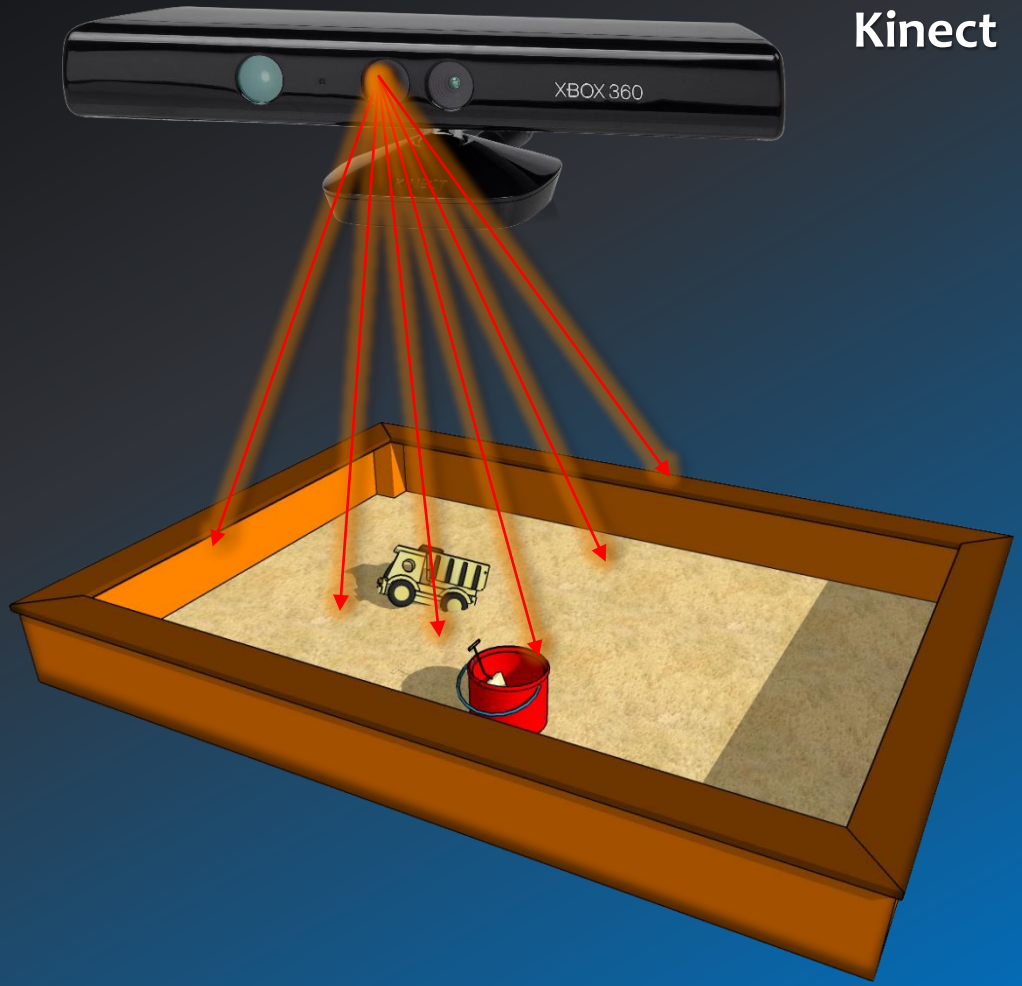
Interactive sandbox

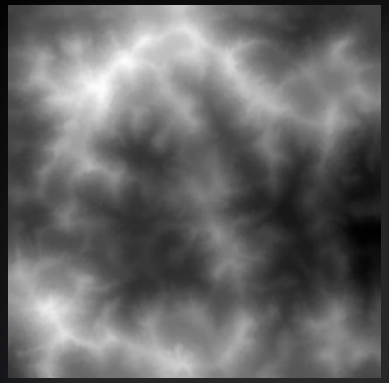
Simulation of nature phenomena on hand-made landscape

Maxim Rud
Tomsk Polytechnic University



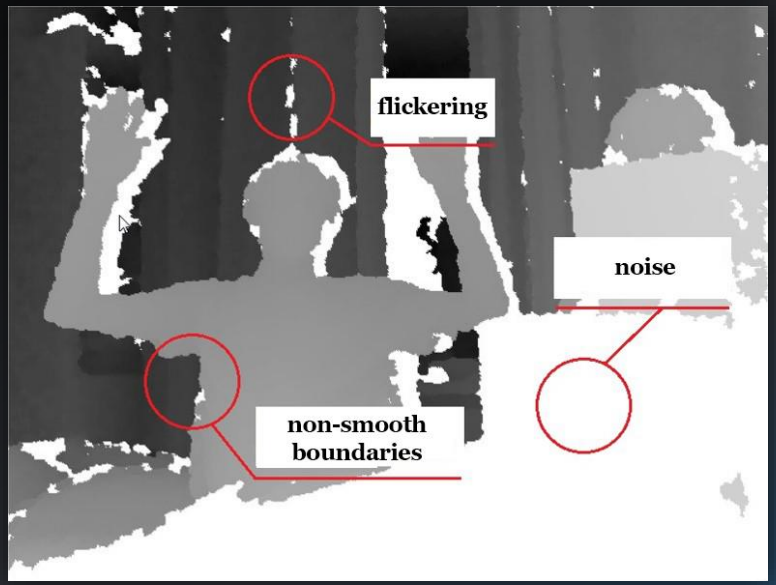


Kinect

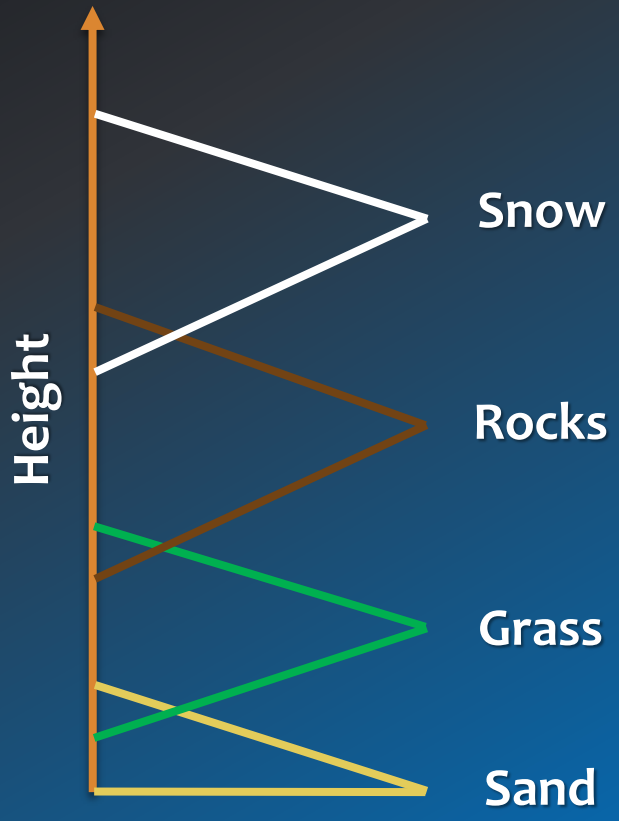


Depth map

Filter Kinect data



Implement multitexturing



From entertainment to research

Basic visualization

New software (OpenGL + GLSL)

Physical Simulation (GPGPU)

How to animate terrain?

Water flow

Volcanos

Ice melting

Fluid simulation

Choosing a method

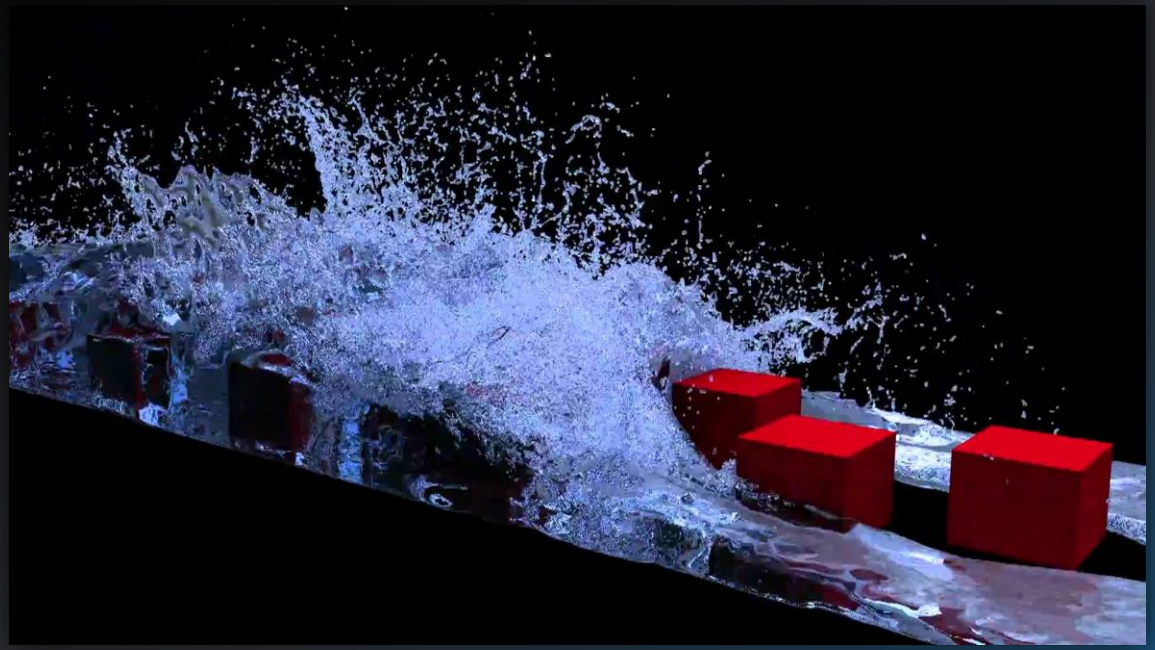
Ocean simulation

Water is the main character

No interacting with terrain

Simulating only waves, but not flow

Choosing a method



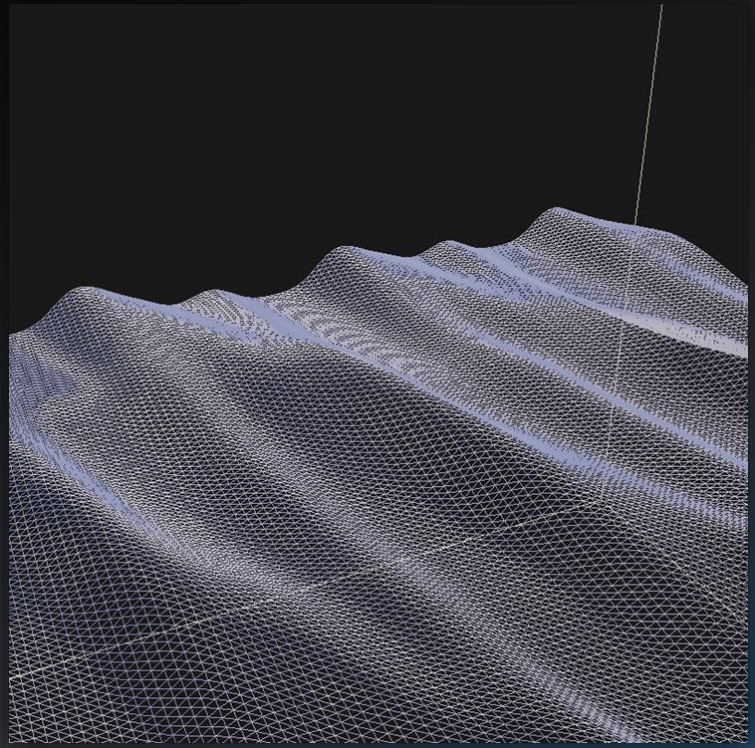
Smooth particle hydrodynamics

Computationally expensive

Impractical, because we have only 2D-projection

Performance depends on quantity of water

Choosing a method



Height field

Water is represented by a grid

No breaking waves

Provides realistic appearance of water

Interacting with bottom

Shallow water equations

$$U = \begin{pmatrix} w \\ hu \\ hv \end{pmatrix}$$

Numerical scheme

Well-balanced

$$u=0,\ h + B = const$$

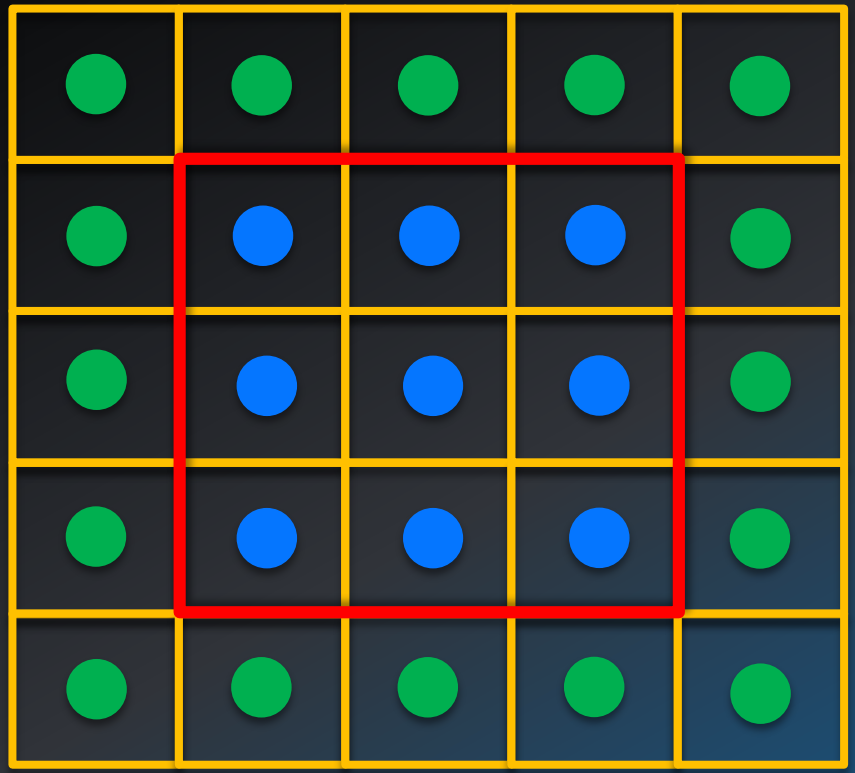
Positivity preserving

Must handle dry (near-dry) zones

$$u \pm \sqrt{gh}$$

A. Kurganov and G. Petrova, “A second-order well-balanced positivity preserving central-upwind scheme for the Saint-Venant system”

Numerical scheme



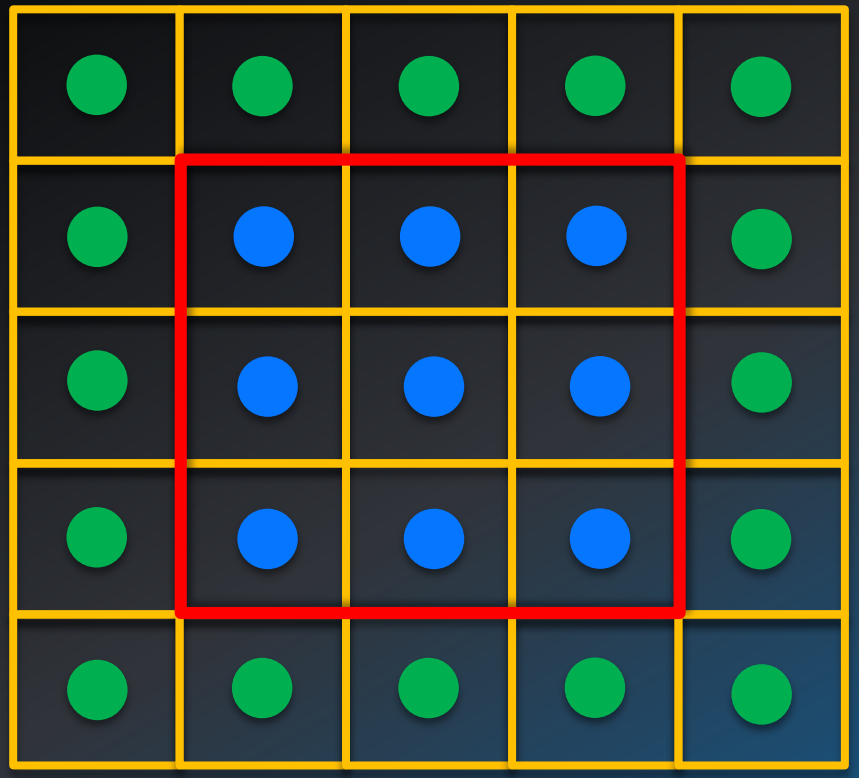
Initial data

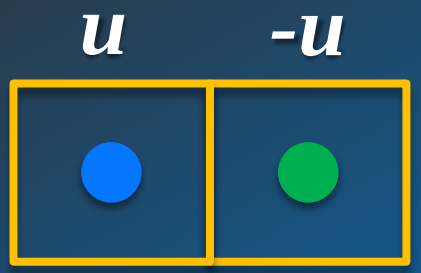
Kurganov – Petrova

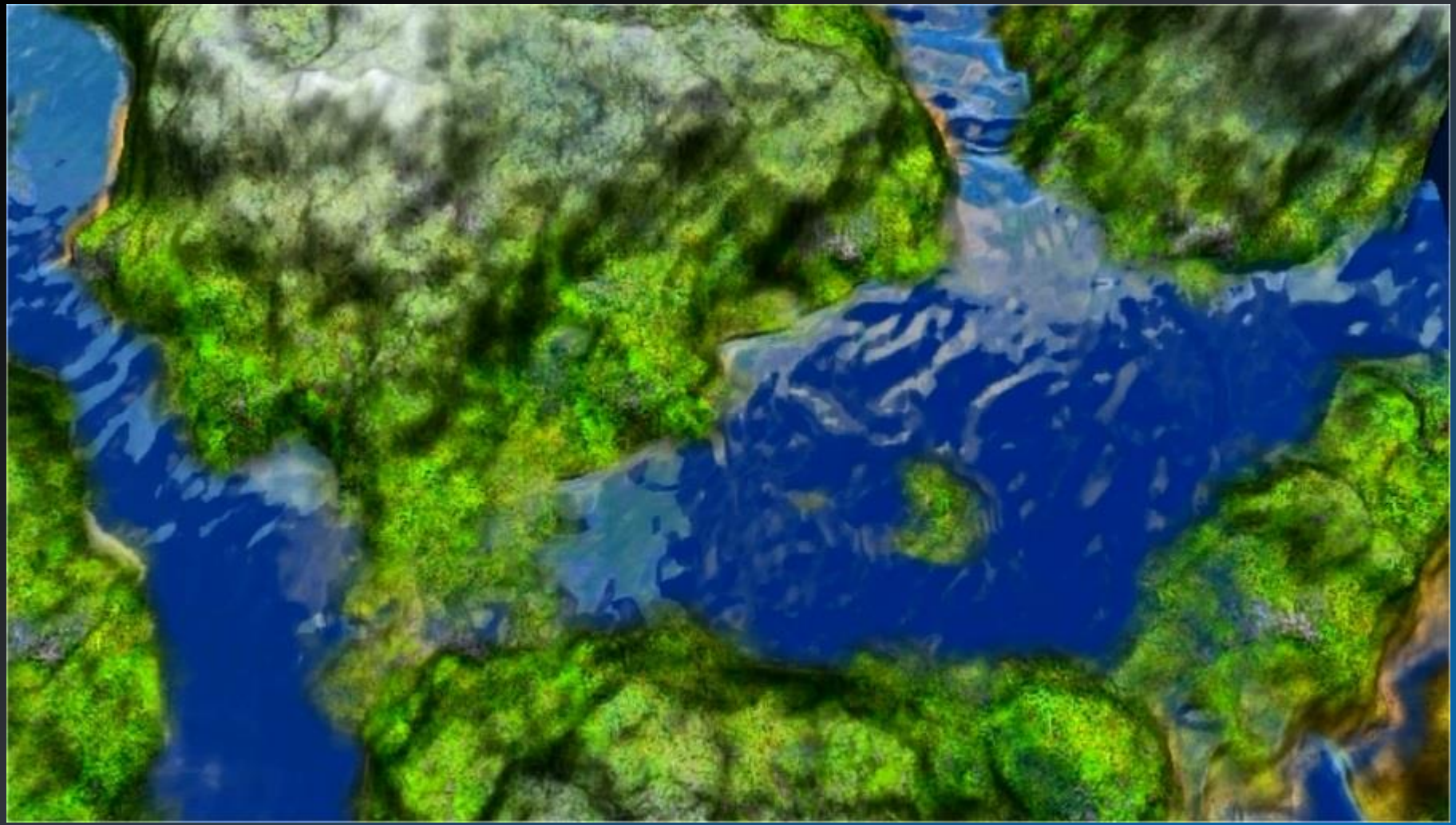
Timestep

Runge - Kutta

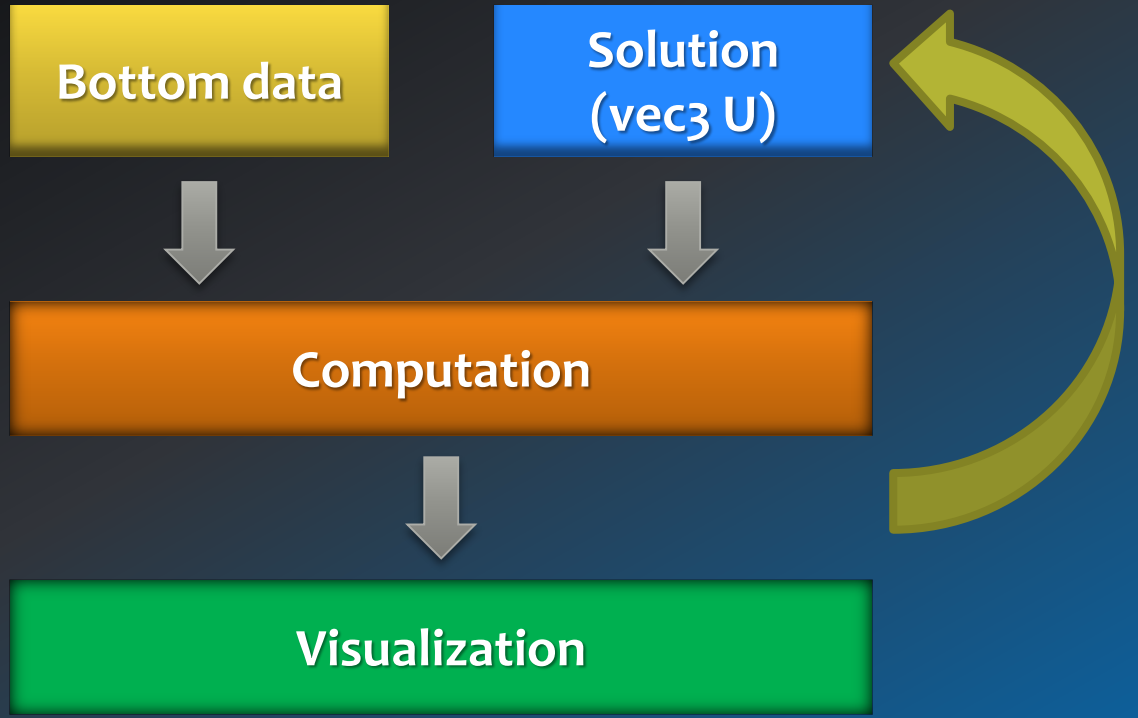
Boundary conditions





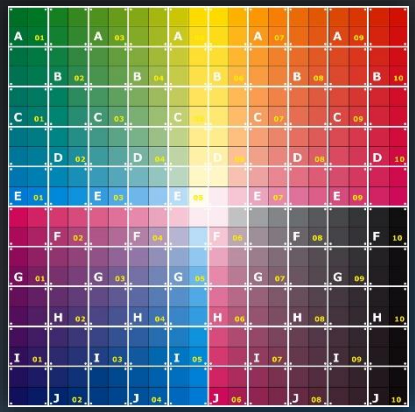


Implementation on GPU using OpenGL



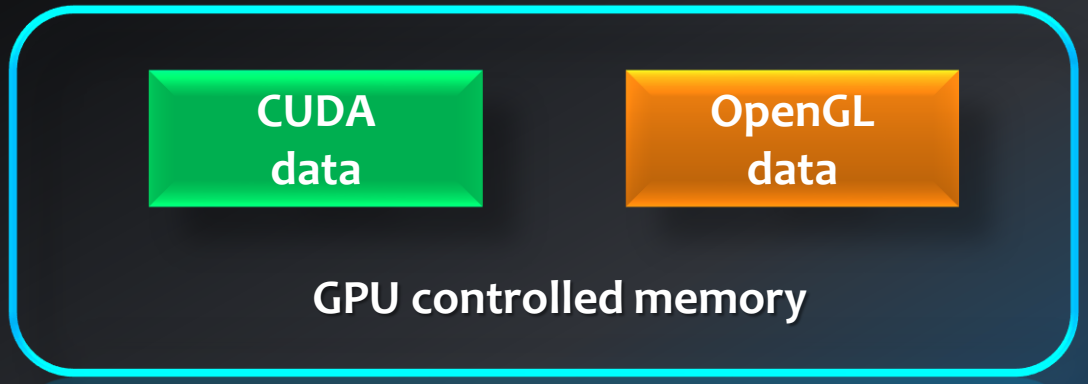
Implementation on GPU using OpenGL

Vertex input data → Texture → Fragment shader → Compute values → Render to texture → To the next stage



glFramebuffer extention

Implementation on GPU using CUDA



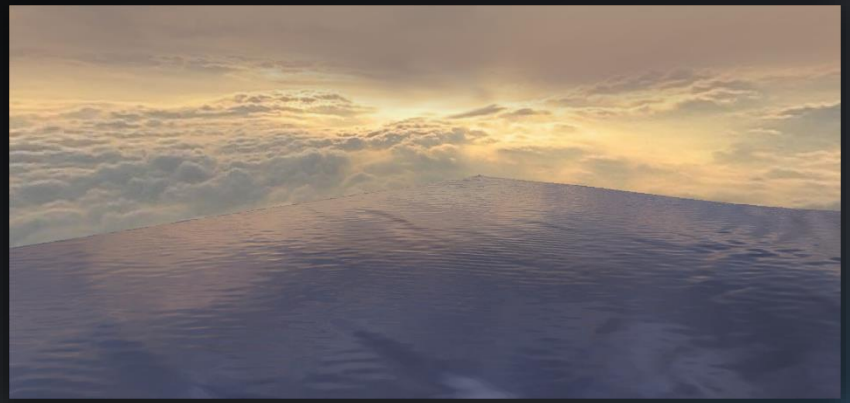
Computational kernel (CUDA)

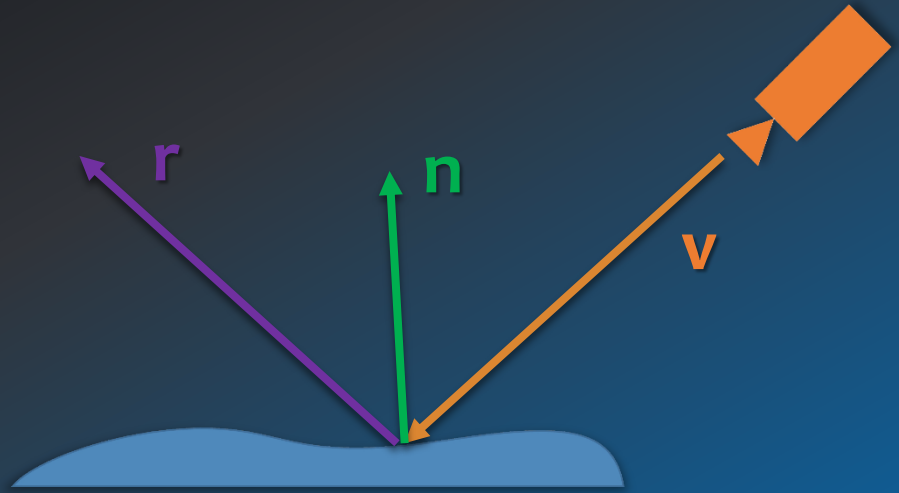
Synchronizing threads

Timestep – reduction algorithm

Visualization kernel (OpenGL)

Environmental reflection





r = reflect(-v,n)

ReflectColor = texture(cubeMap, r)

Terrain reflection

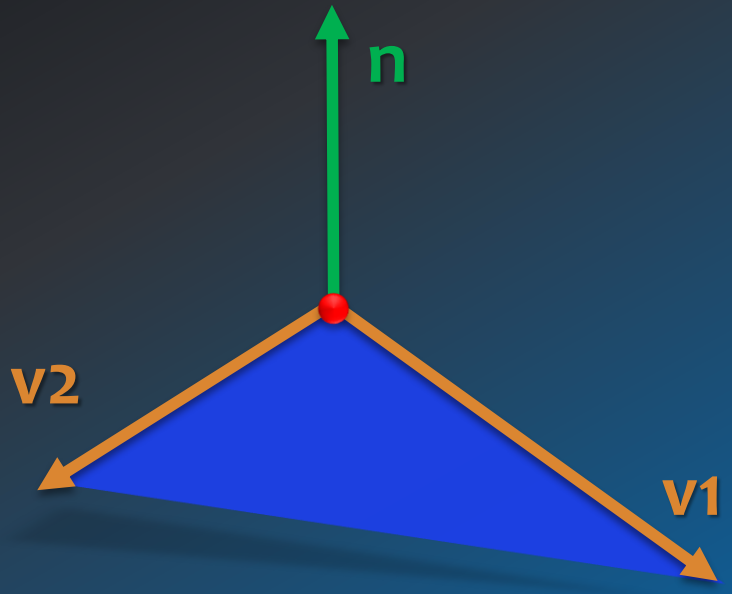
mirrored terrain

texture

water

$$\text{transparency} = 1.0 - \exp(-\alpha \cdot \text{height})$$

Computing normals on GPU



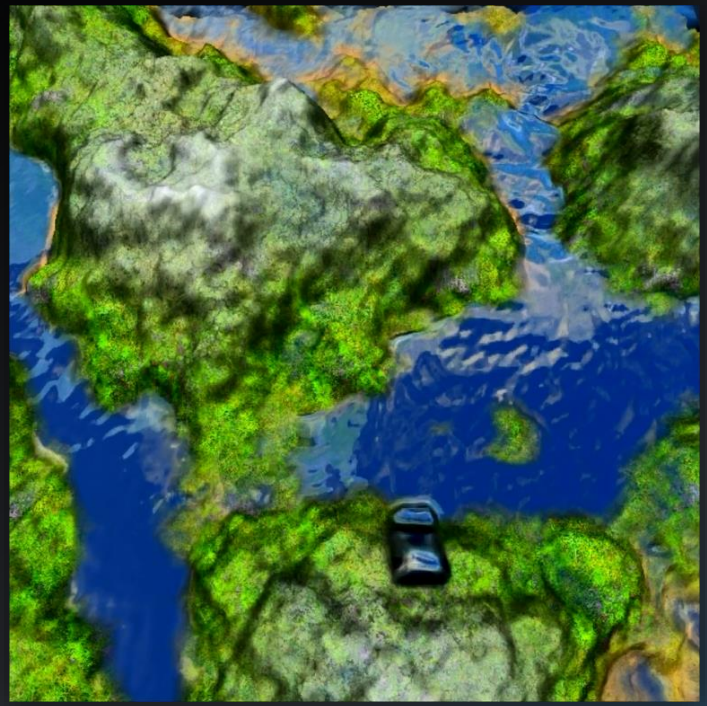
n = dot product (v1, v2)

Geometry shader is not sutable

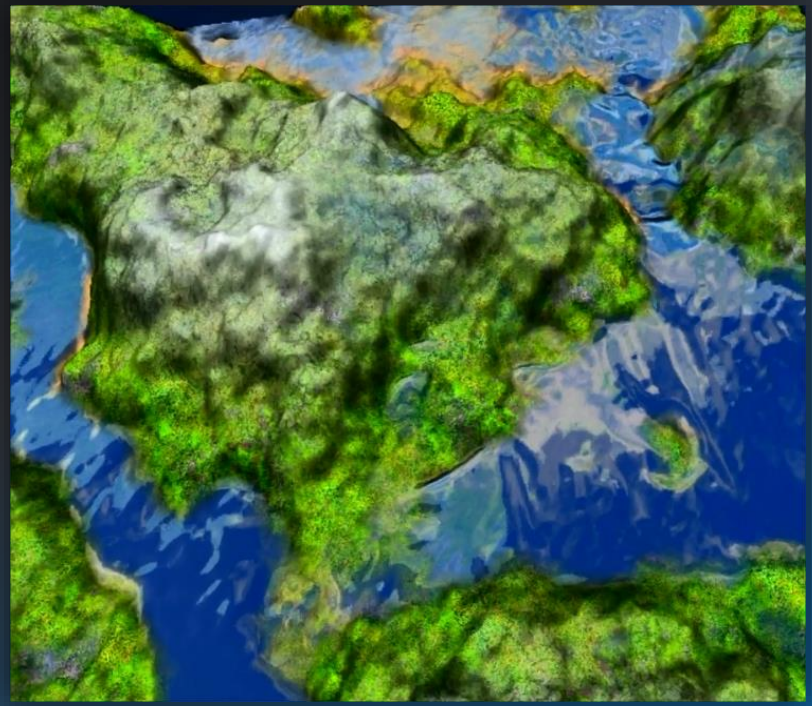
(We need acsess to more geometry)

Render geometry to texture

Update only *z* coordinate

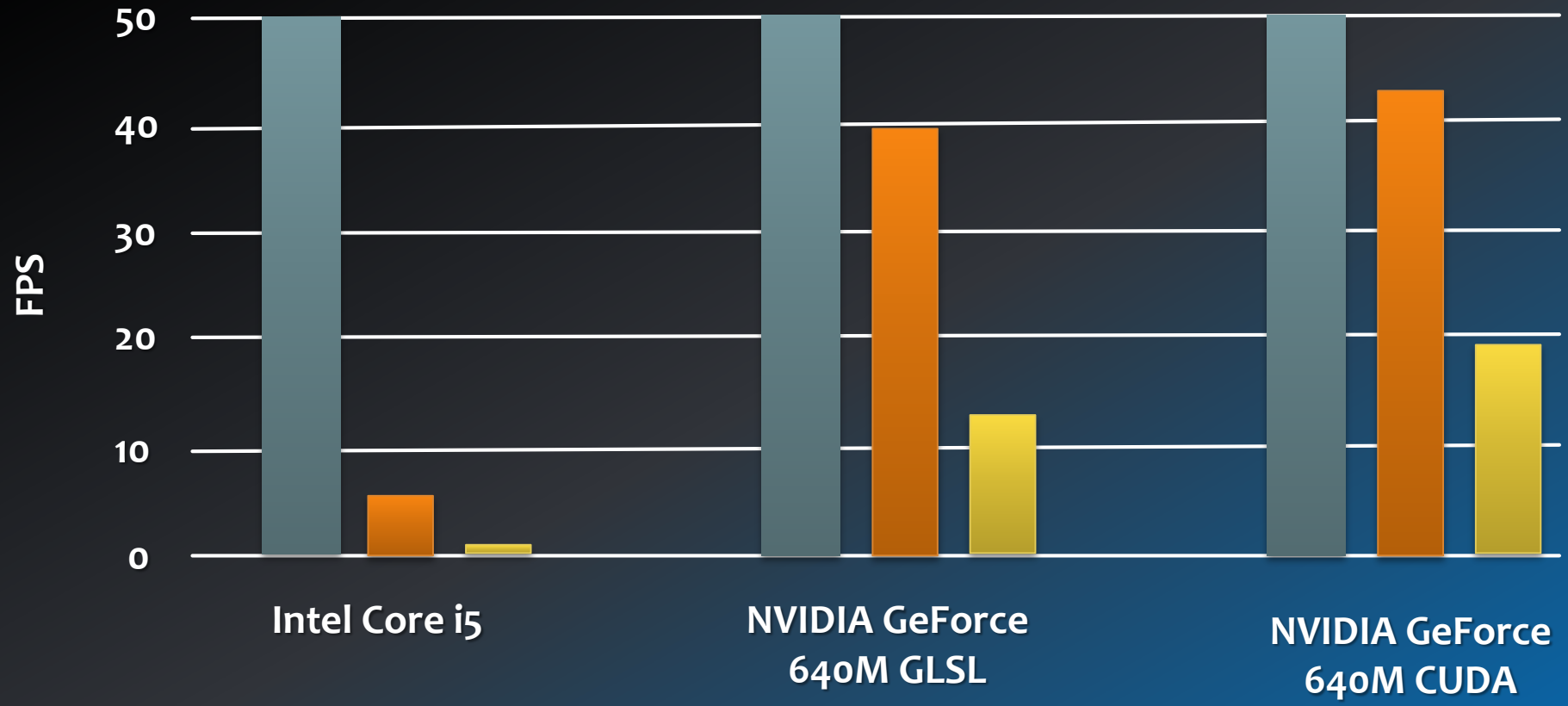


On CPU



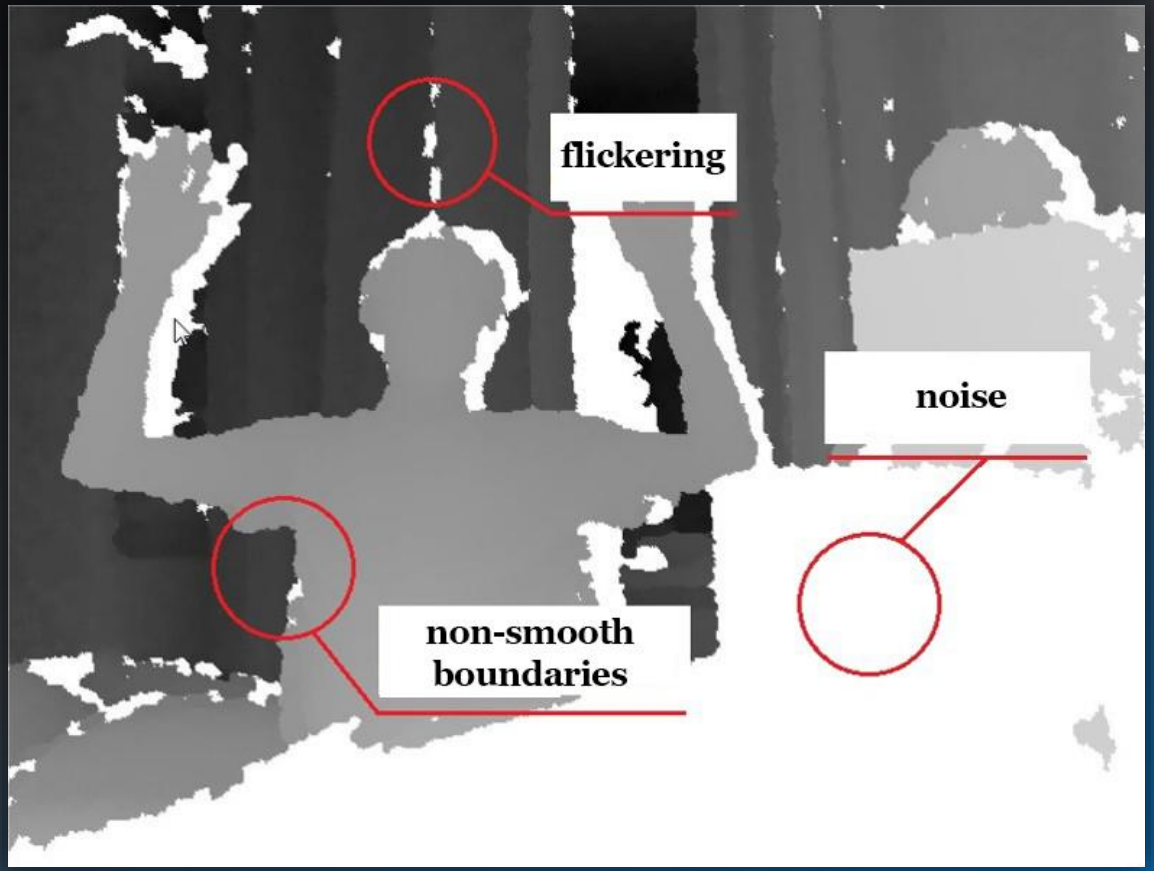
On GPU

+ 8-10 FPS


50
40
30
20
10
0
FPS
Intel Core i5
NVIDIA GeForce 640M GLSL
NVIDIA GeForce 640M CUDA
50 x 50
320 x 240
500 x 500

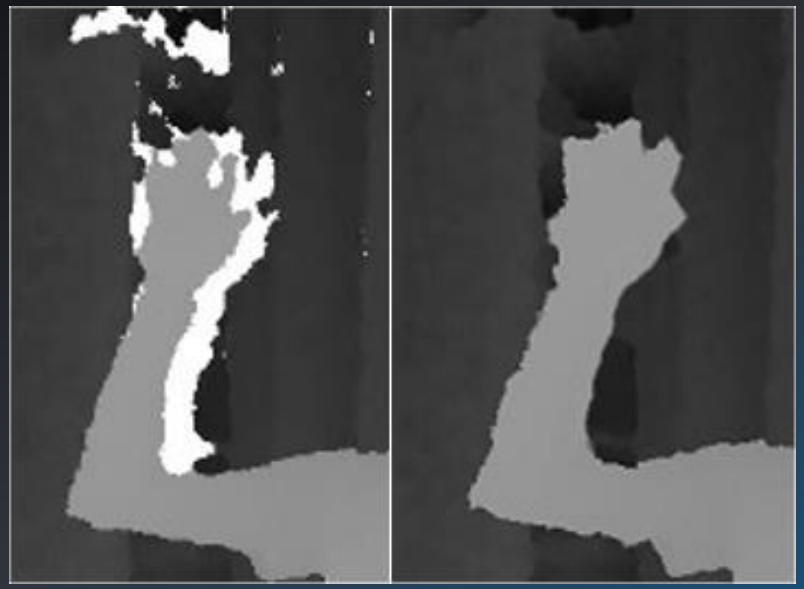
Kinect data filtering

Kinect data problem



Filter

Fast marshing method

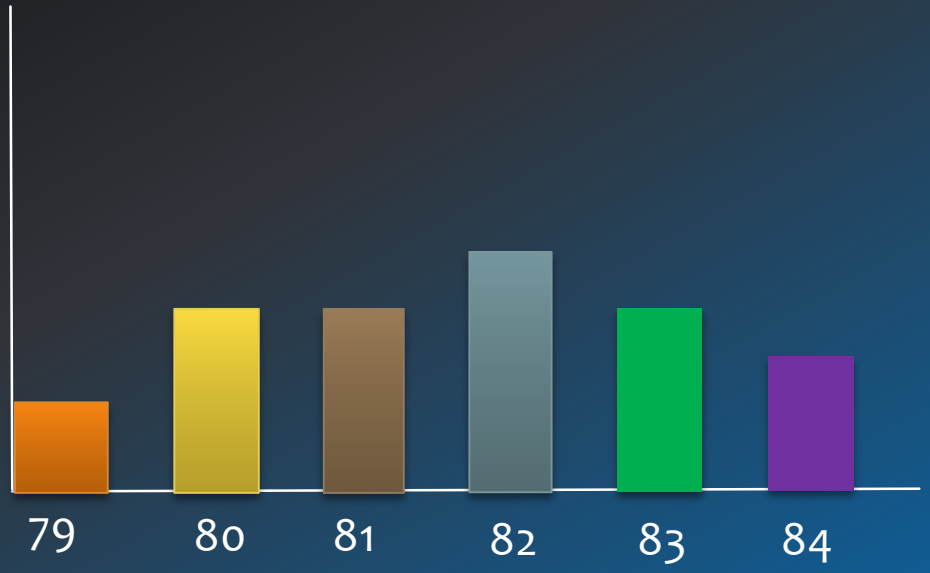


A. Telea “An image inpainting technique based on Fast marshing method”

Median filter

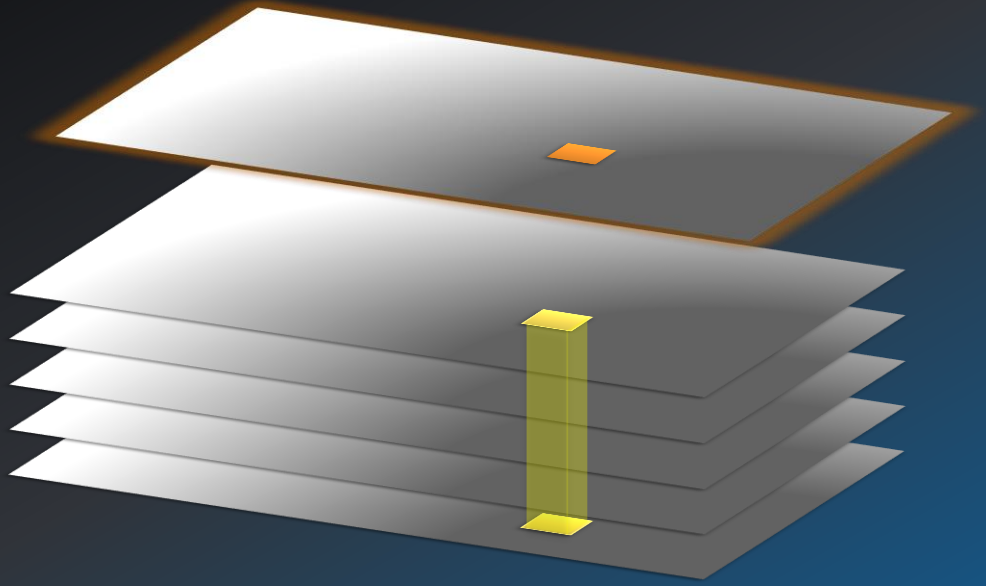
Median filter

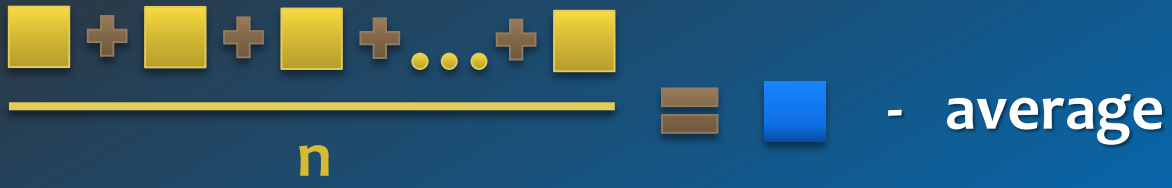
80	82	81	84	82
83	80	84	82	81
83	79	**82**	80	81
84	83	80	84	82
83	80	79	82	81



Statistic filter

Update only if





- average

FMM

Median Filter

Statistic Filter

The next stage

Basic visualization

New software (GLSL)

Physical Simulation (GPGPU)

Photorealistic rendering

StudyAndPlay

www.studyandplay.ru

Maxim Rud
rudmax13@gmail.com
+7-923-412-4948