Flight Simulation

Advisor:
Hans de Nivelle

Team:
Alisher Shakhiyev, Alen German, Auyez Zhumashev

The project

Components

- 3D physics simulation
- 3D graphics
- 3D terrain (landscape)
- Collision detection

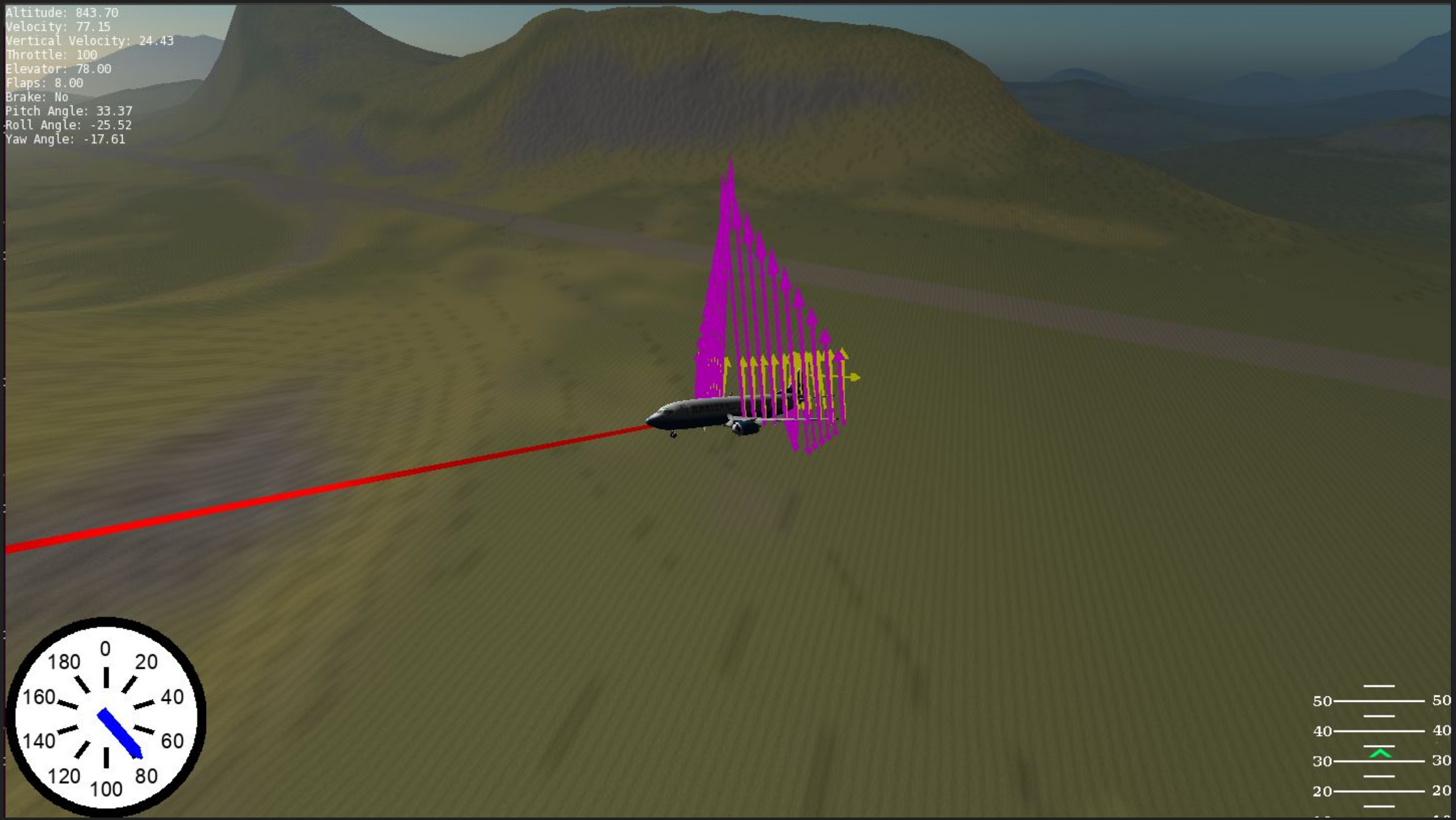
Language & libraries

- C++
- SFML
- OpenGL
- GLM

Purpose

- For the sake of learning

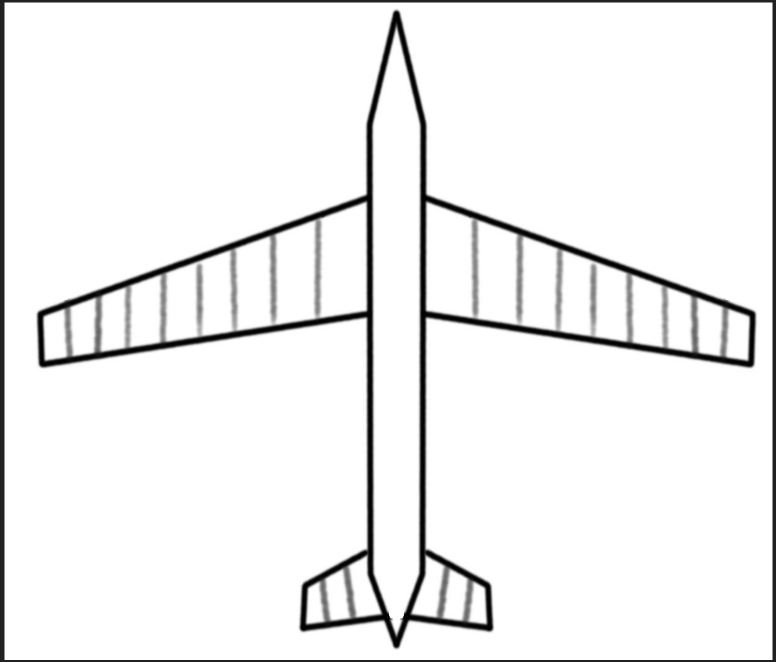
3D Graphics



Altitude: 843.70
Velocity: 77.15
Vertical Velocity: 24.43
Throttle: 100
Elevator: 78.00
Flaps: 8.00
Brake: No
Pitch Angle: 33.37
Roll Angle: -25.52
Yaw Angle: -17.61
0
20
40
60
80
100
120
140
160
180
50
40
30
20

Physics from 2D to 3D, Airfoil Segmentation

- Root and tip sections have different velocities when the plane rotates.



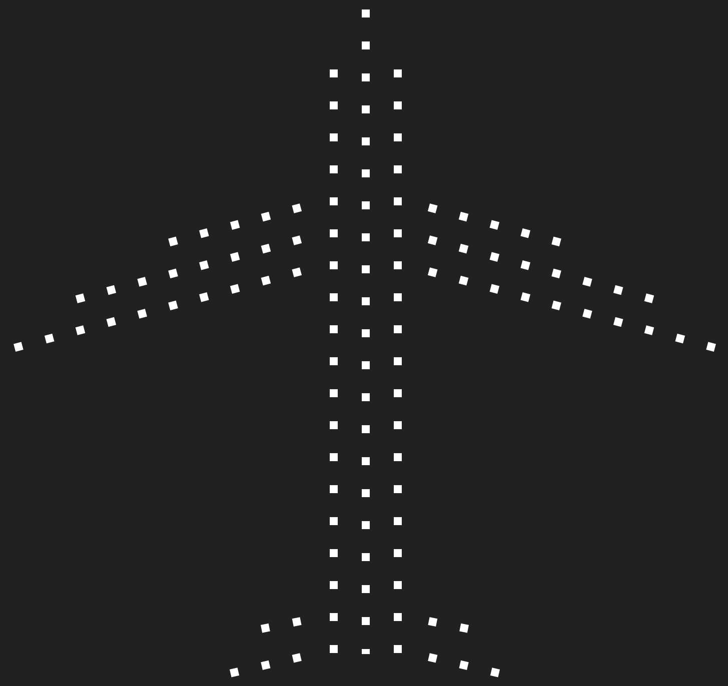
Physics from 2D to 3D, Inertia Matrix

- In 2D inertia is a scalar.
- In 3D inertia is a 3x3 matrix.

$$\mathbf{I} = \begin{bmatrix} I_{11} & I_{12} & I_{13} \\ I_{21} & I_{22} & I_{23} \\ I_{31} & I_{32} & I_{33} \end{bmatrix} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}.$$

[1]

Physics from 2D to 3D, Inertia Matrix



$$I_{11} = I_{xx} = \sum_{k=1}^{N} m_k \left(y_k^2 + z_k^2\right),$$

$$I_{22} = I_{yy} = \sum_{k=1}^{N} m_k \left(x_k^2 + z_k^2\right),$$

$$I_{33} = I_{zz} = \sum_{k=1}^{N} m_k \left(x_k^2 + y_k^2\right),$$

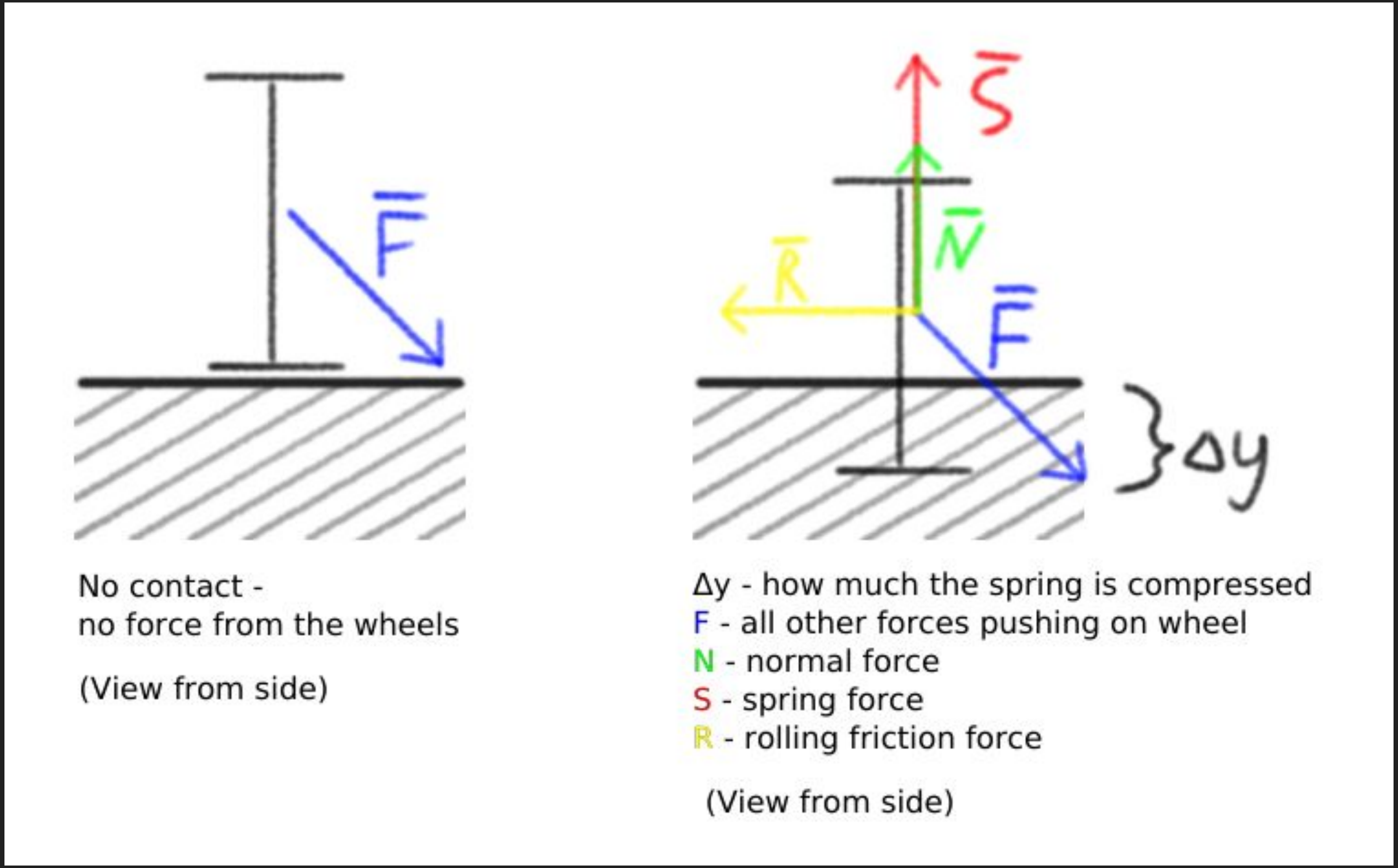
$$I_{12} = I_{21} = I_{xy} = -\sum_{k=1}^{N} m_k x_k y_k,$$

$$I_{13} = I_{31} = I_{xz} = -\sum_{k=1}^{N} m_k x_k z_k,$$

$$I_{23} = I_{32} = I_{yz} = -\sum_{k=1}^{N} m_k y_k z_k.$$

[1]

Wheels in 3D

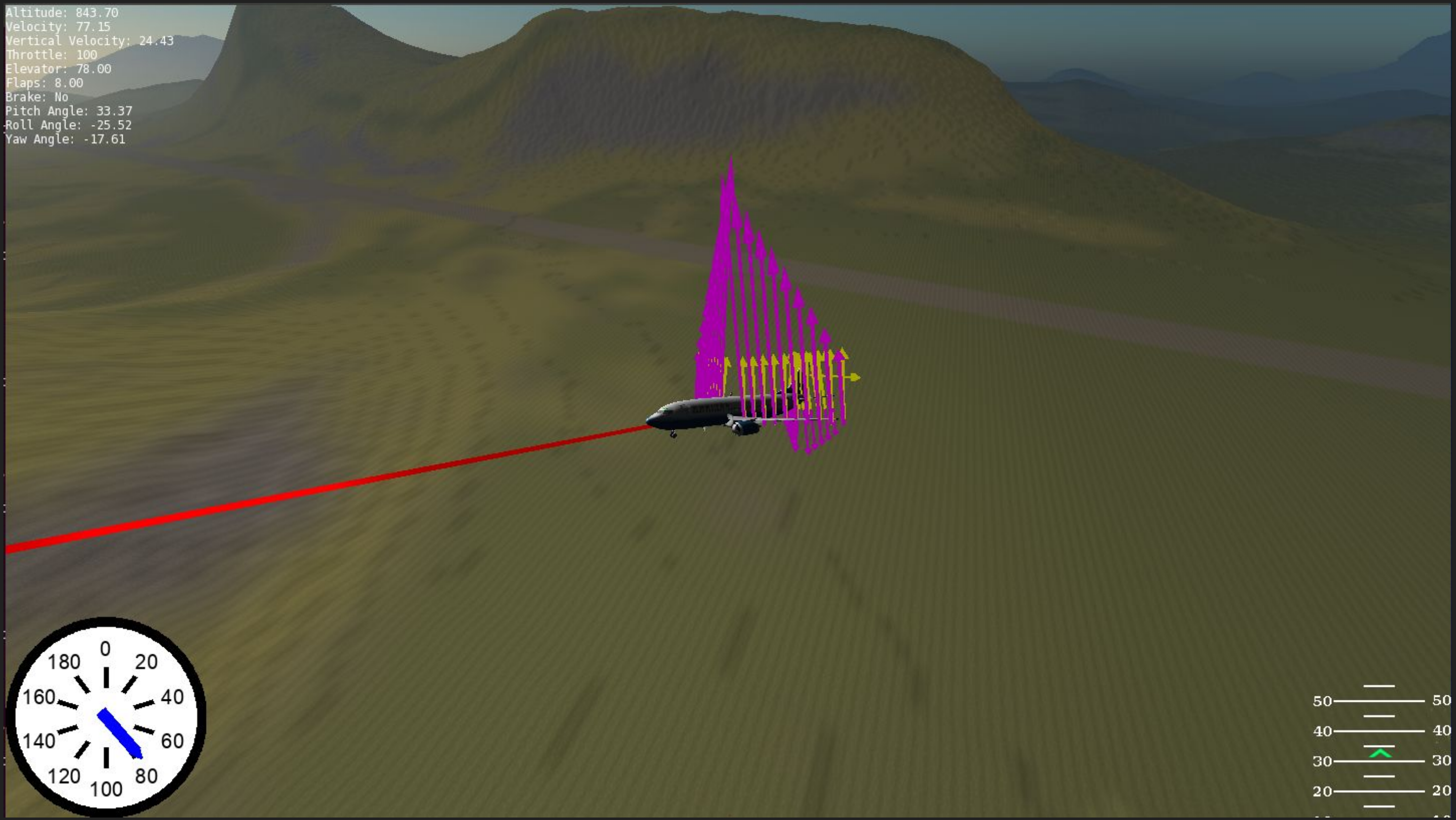

F
S
N
R
F
Δy
No contact -
no force from the wheels
(View from side)
Δy - how much the spring is compressed
F - all other forces pushing on wheel
N - normal force
S - spring force
R - rolling friction force
(View from side)

Wheels in 3D

Cf - friction coefficient when wheel moves forward/backward
Cz - friction coefficient when wheel moves sideways
V - velocity
In this case coefficient will be closer to Cf

(View from top)

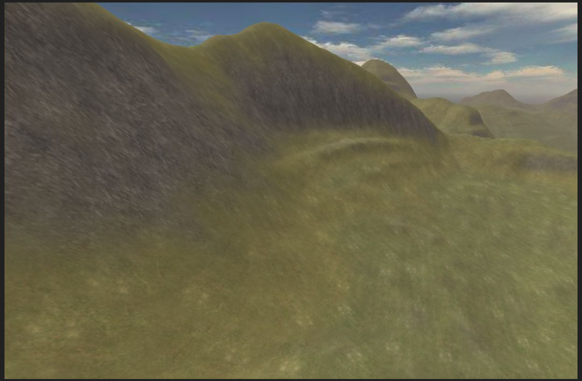
3D Graphics

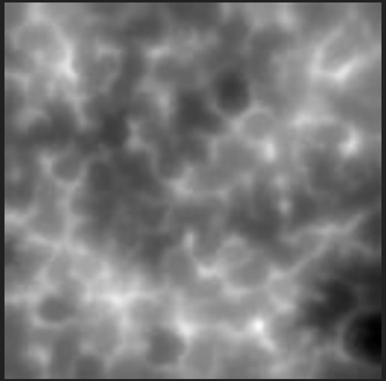


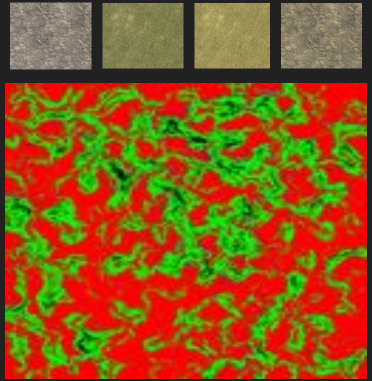
Altitude: 843.70
Velocity: 77.15
Vertical Velocity: 24.43
Throttle: 100
Elevator: 78.00
Flaps: 8.00
Brake: No
Pitch Angle: 33.37
Roll Angle: -25.52
Yaw Angle: -17.61
0
20
40
60
80
100
120
140
160
180
50
40
30
20

3D Graphics

- Terrain is rendered using heightmap
- Different levels of detail based on distance
- Terrain is textured using splatmap
- Deferred rendering
- 3D objects are loaded from AC3D file format

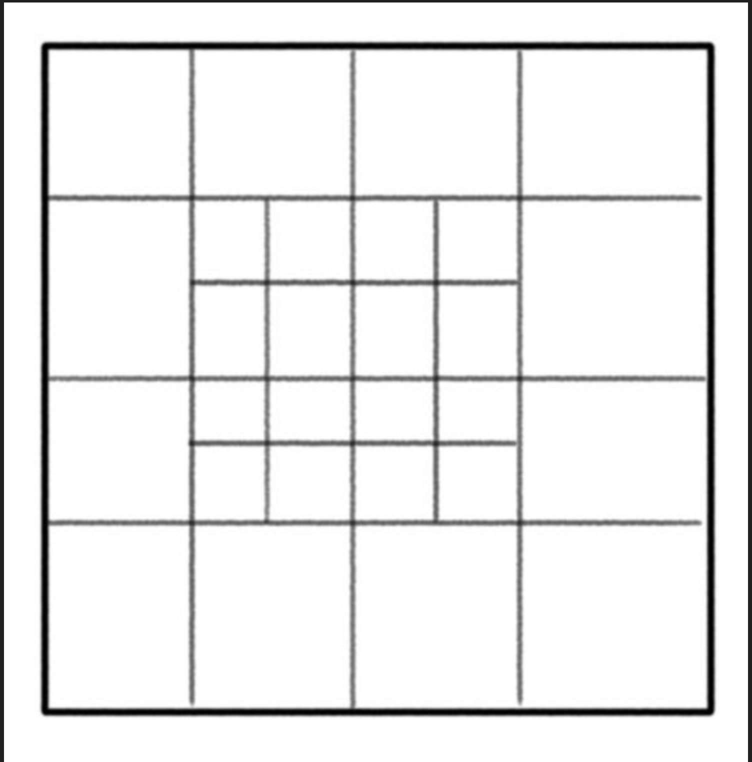






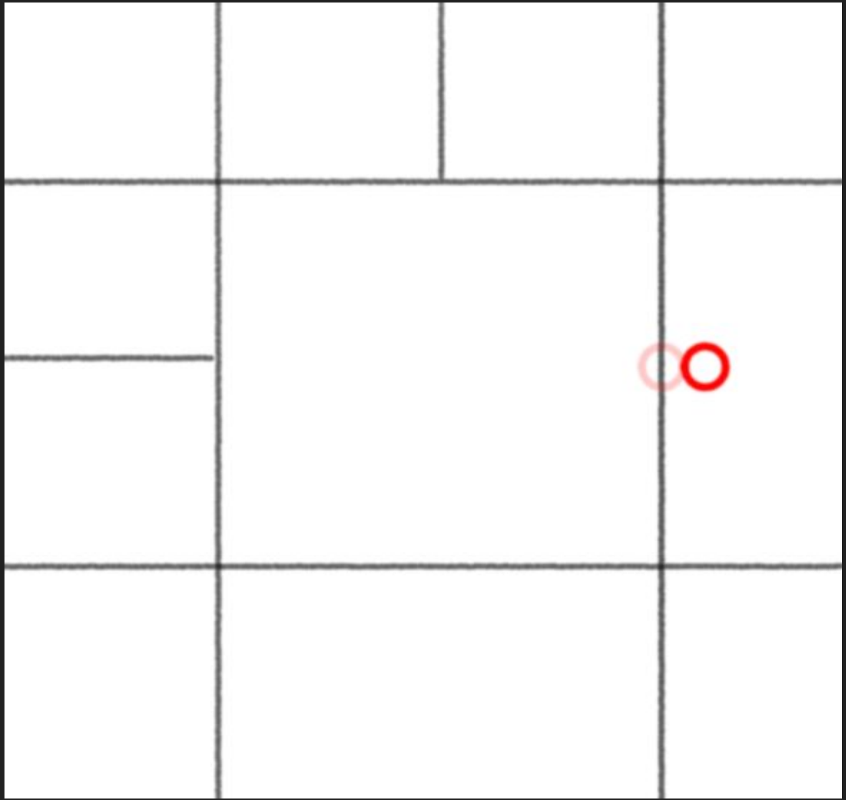
Quadtree neighbor retrieval

- Problem
  - Is a leaf node's neighboring node smaller?
- Assumption
  - Neighbor is either 2 times smaller, 2 times bigger, or of the same size.



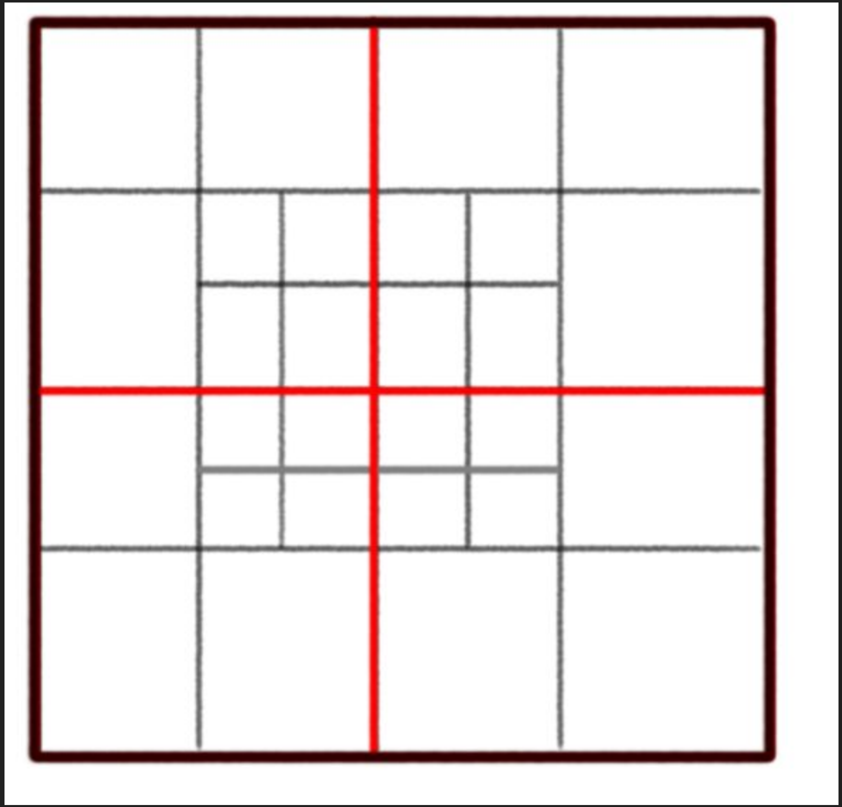
Quadtree neighbor retrieval

- Victorbush's solution [2]
  - Take a pos at the edge of a node
  - Add some ε
  - DFS for a node that contains the position



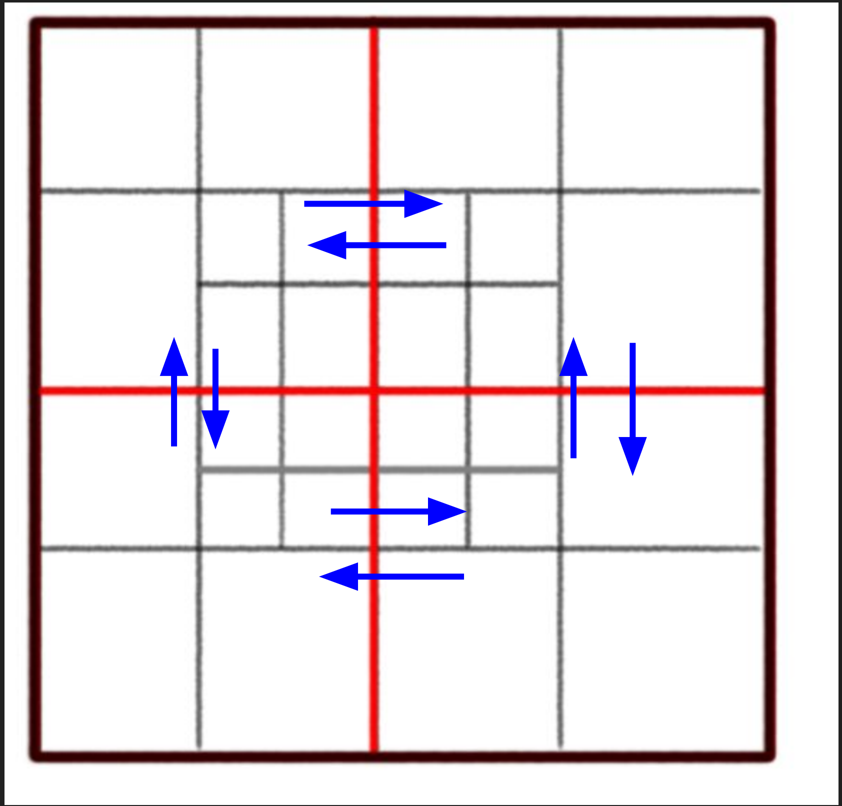
Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.



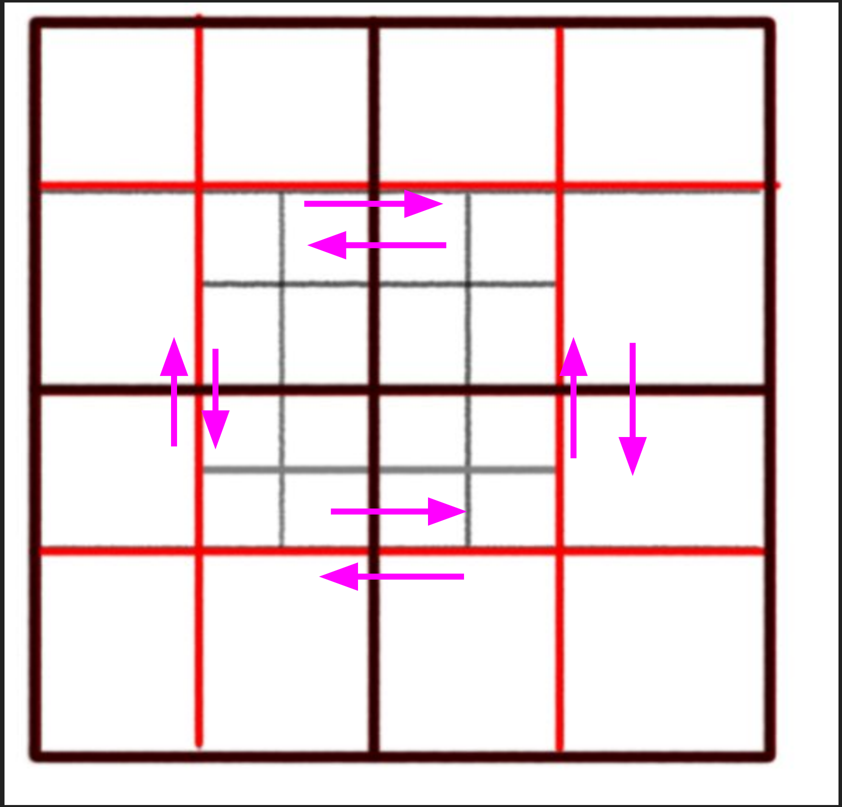
Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.



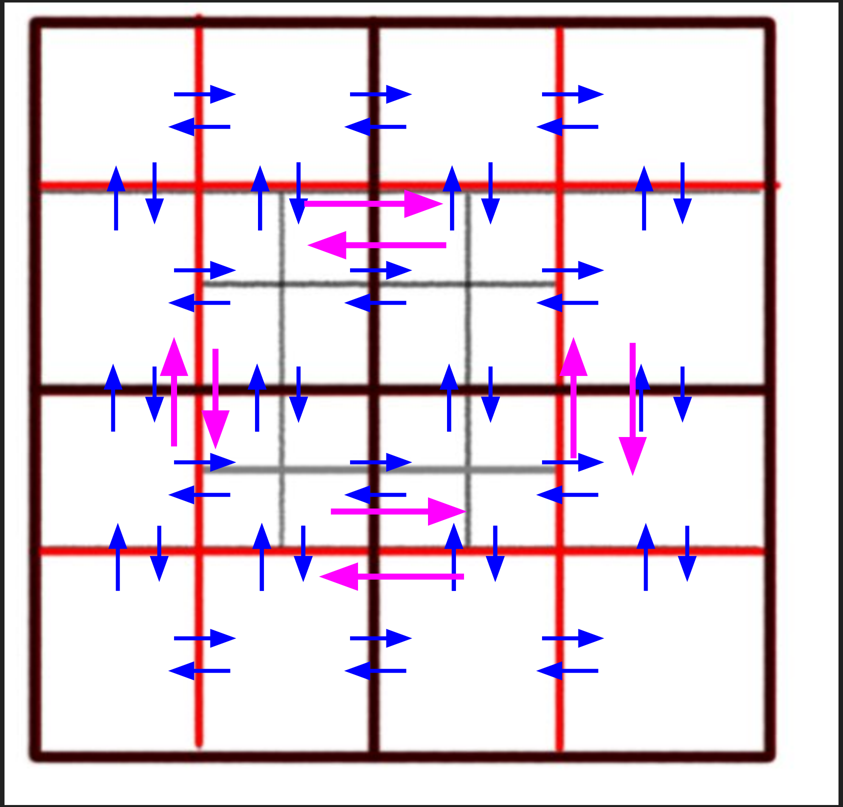
Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.



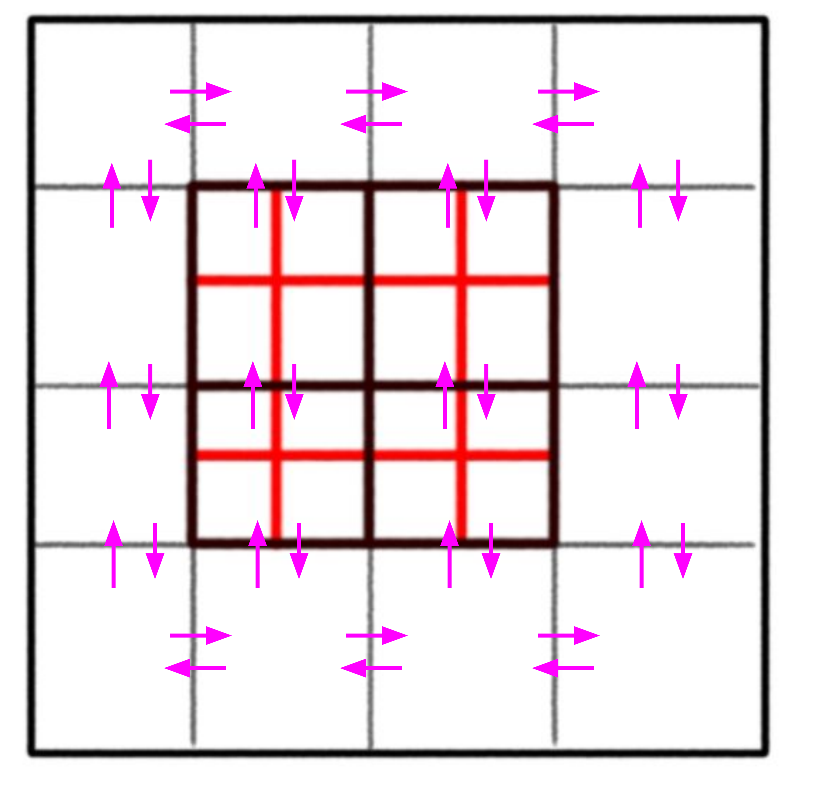
Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.



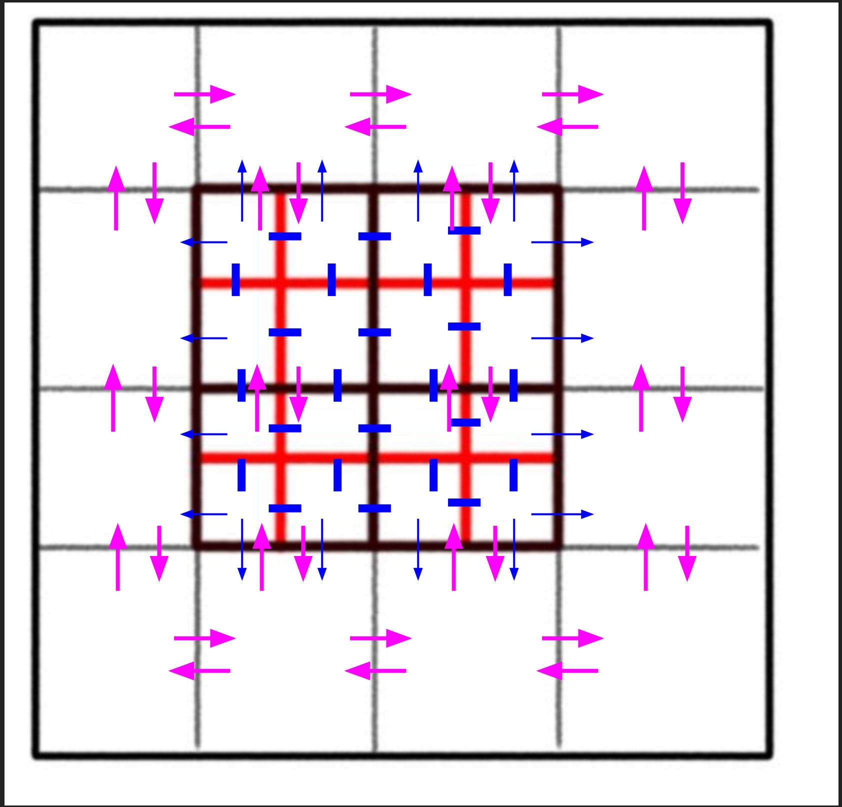
Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.
  - Larger leaf nodes don't know smaller neighbors.
  - Smaller neighbors know larger neighbors.



Quadtree neighbor retrieval

- Our solution
  - Do a breadth first traversal from largest node down to leaf nodes.
  - Set neighbors of children along the way.
  - Larger leaf nodes don't know smaller neighbors.
  - Smaller neighbors know larger neighbors.



Quadtree neighbor retrieval

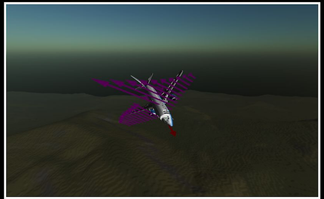
- Victorbush’s solution [2]
  - DFS for every leaf node

- Our solution
  - 1 breadth first traversal

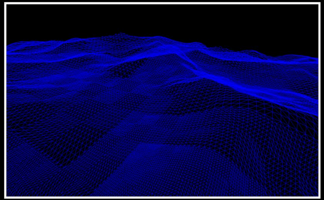
References

1) Moment of inertia wiki page: https://en.wikipedia.org/wiki/Moment_of_inertia
2) Victorbush, Tessellated Terrain Rendering with Dynamic LOD: https://victorbush.com/2015/01/tessellated-terrain/

FLIGHT SIMULATOR



Realistic physics
Atmospheric scattering



Massive terrain with
dynamic level of detail

We wanted to learn about low-level graphics programming and 3D physics simulation. This project is a byproduct of our learning experience. It was done using C++ and OpenGL.

Created by:
- Alisher Shakhiyev
- Alen German
- Auyez Zhumashev

Advisor:
- Hans De Nivelle

Computer Science Department

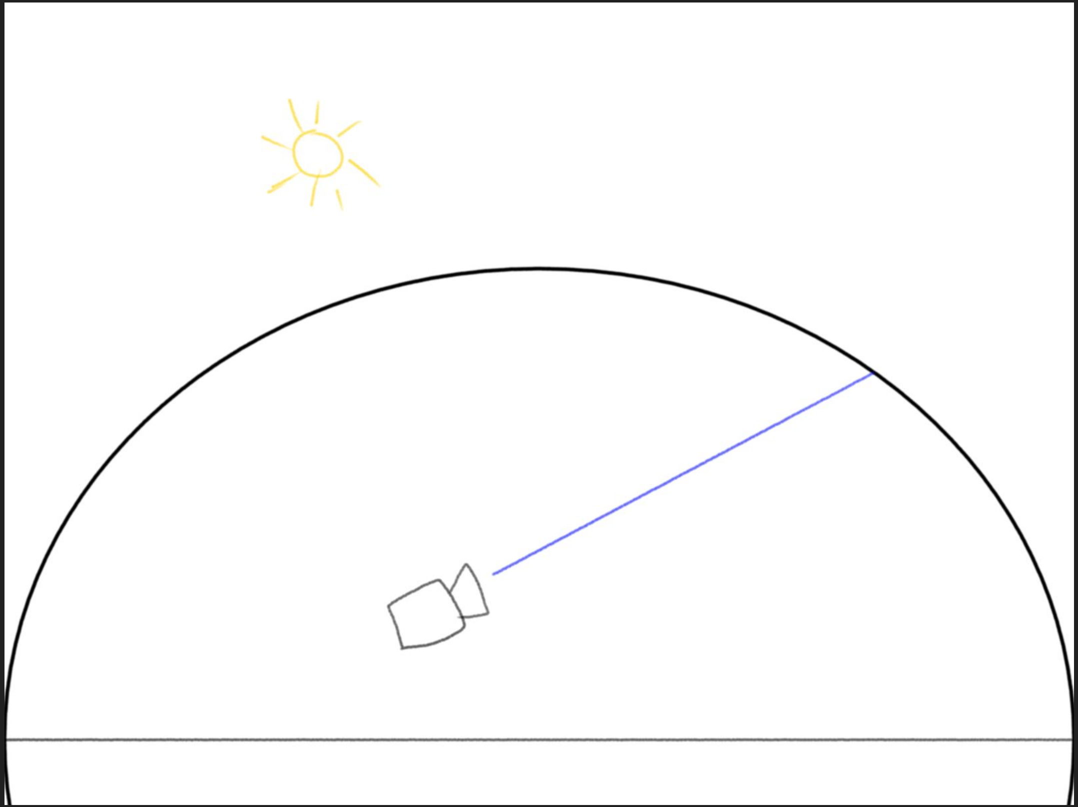
Clouds

Atmospheric Scattering

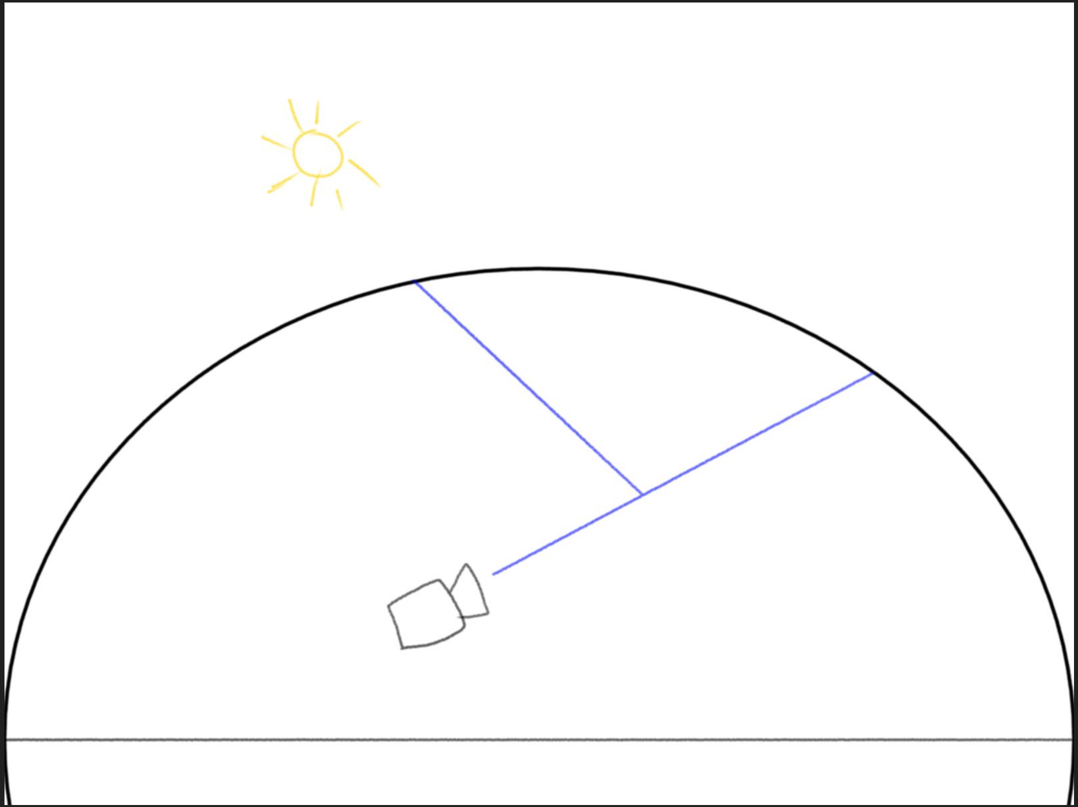
- Transmittance - the ratio of light that reaches the eye.
- **Light_eye = T(extinction_coeff, distance)**

- Transmittance is given by Lambert-Beer law
- **$T(\sigma, d) = \exp(-\sigma * d)$**

Atmospheric Scattering



Atmospheric Scattering



Atmospheric Scattering