Desertscape Simulation - Grid Enrichment

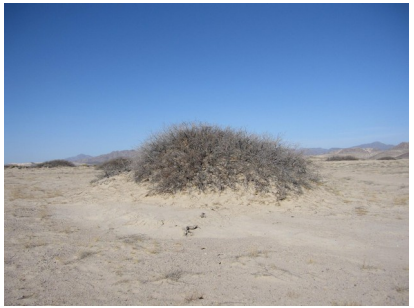
Anant Kumar & Shivam Sood

Agenda

- Background
- Original Paper : Desertscape Simulation
- Our Approach
  - Retracing time steps
  - Quadtrees for Local Refinement
  - Demo
- Future Work

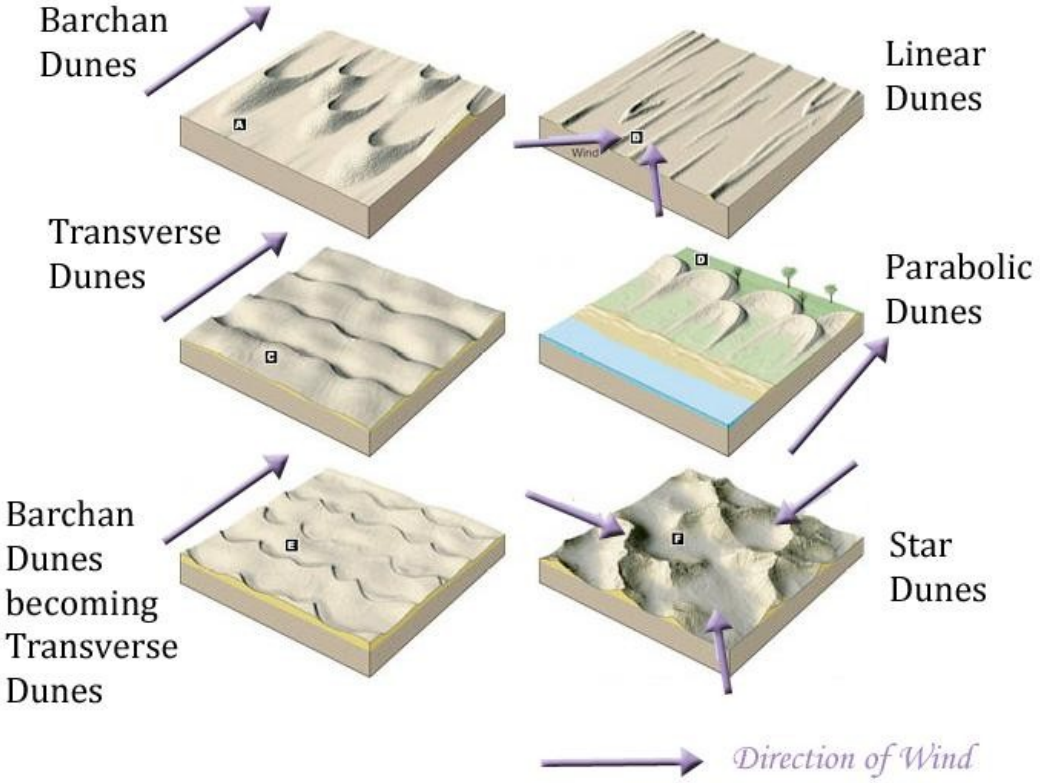
Background

Desert Landscapes

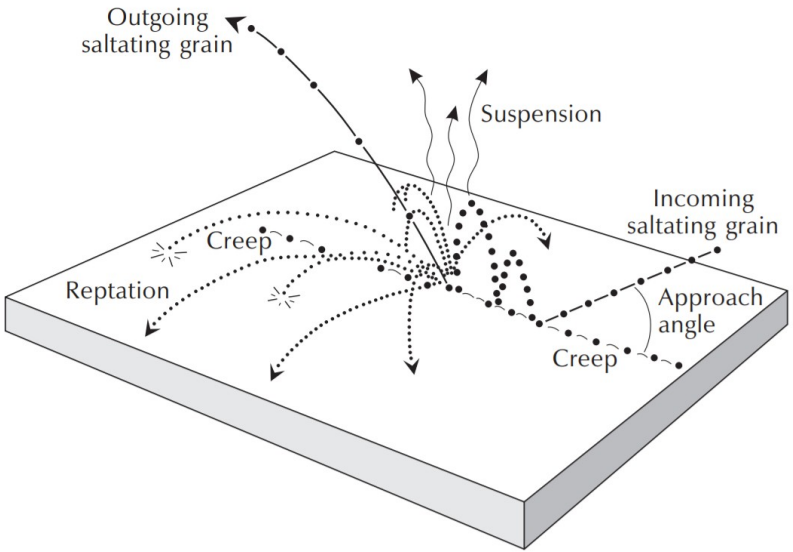


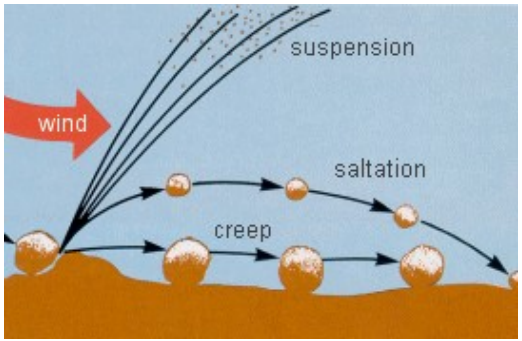
Nabhka

Yardang



Modes of Grain Transport by Wind



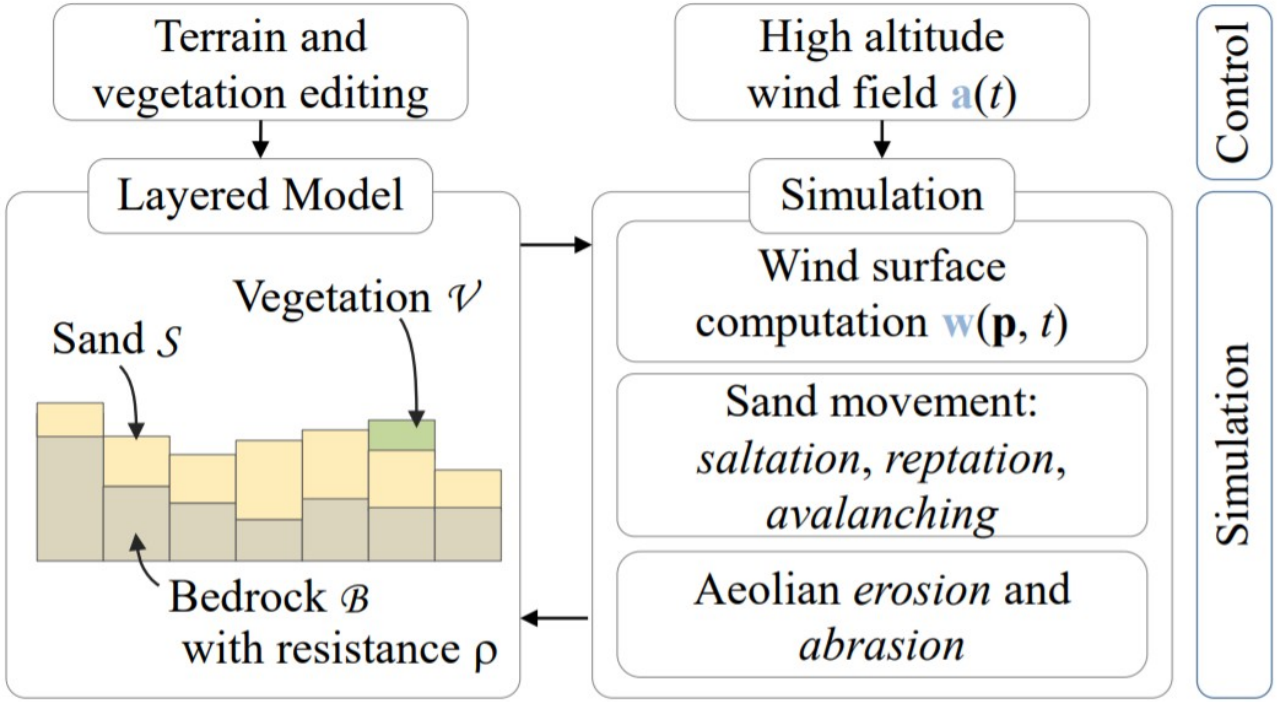


Desertscape Simulation

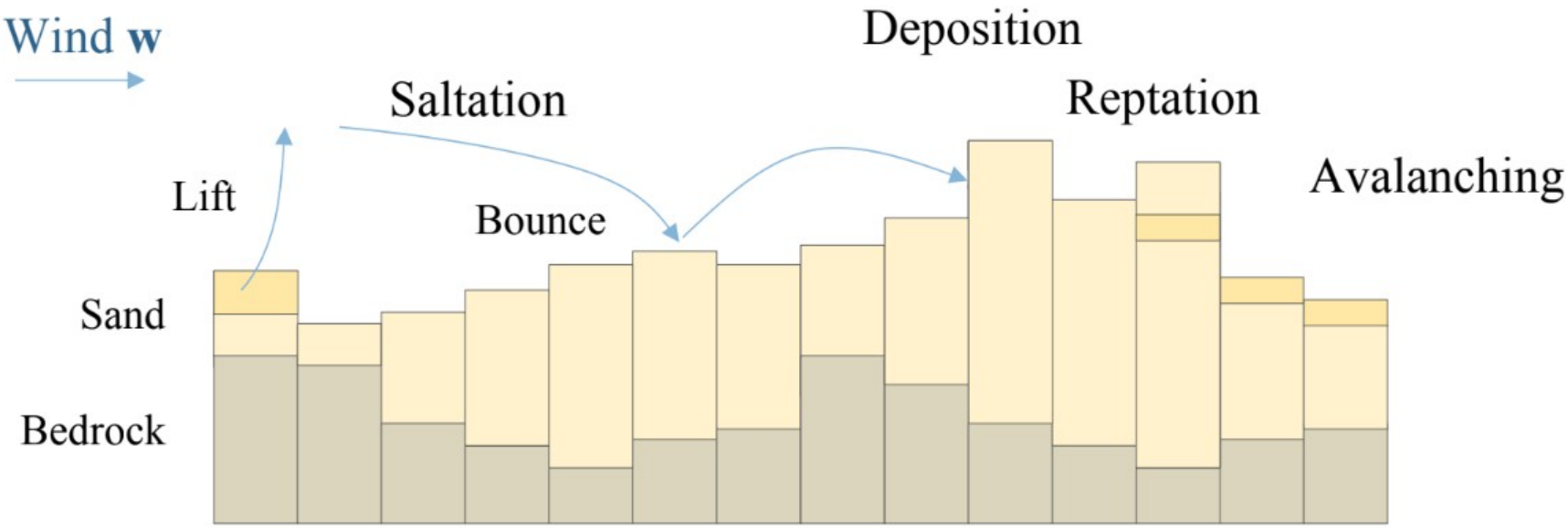
Axel Paris, Adrien Peytavie, Éric Guérin, Oscar Argudo, Éric Galin

https://aparis69.github.io/public_html/projects/paris2019_Deserts.html

Overview of the Approach

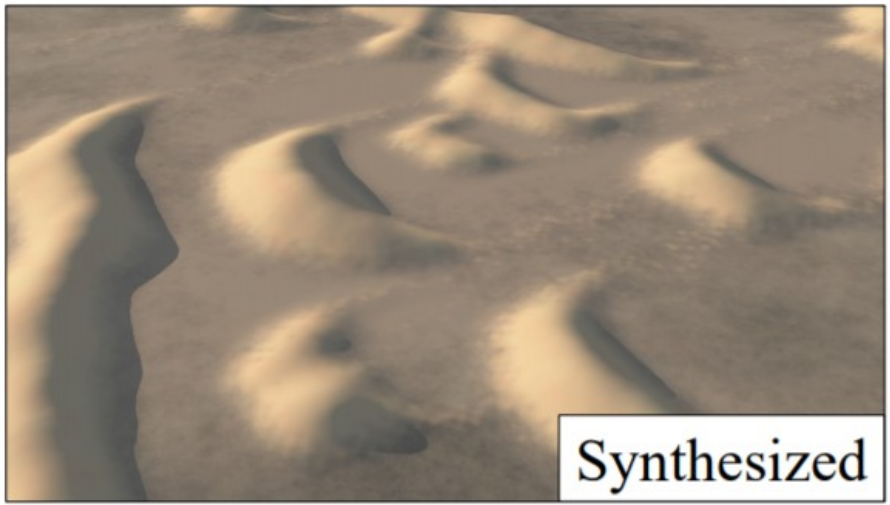


Modelling Transport Phenomena



Limitations

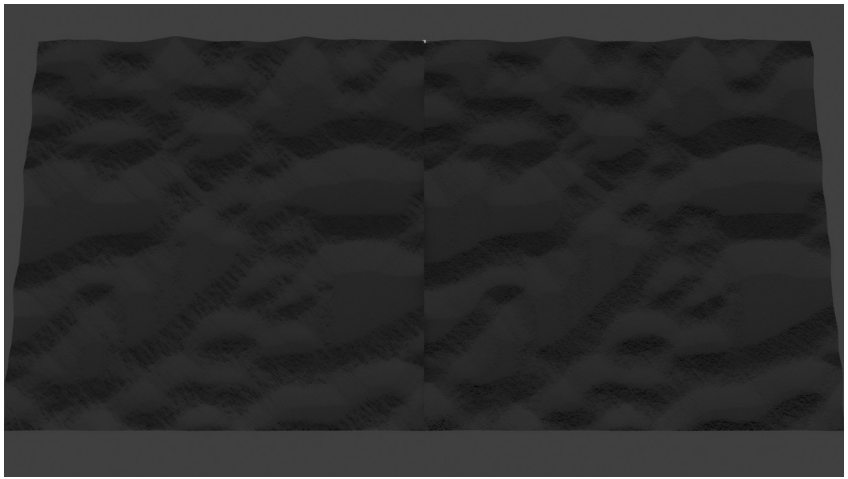
Grid cell ~ 10m

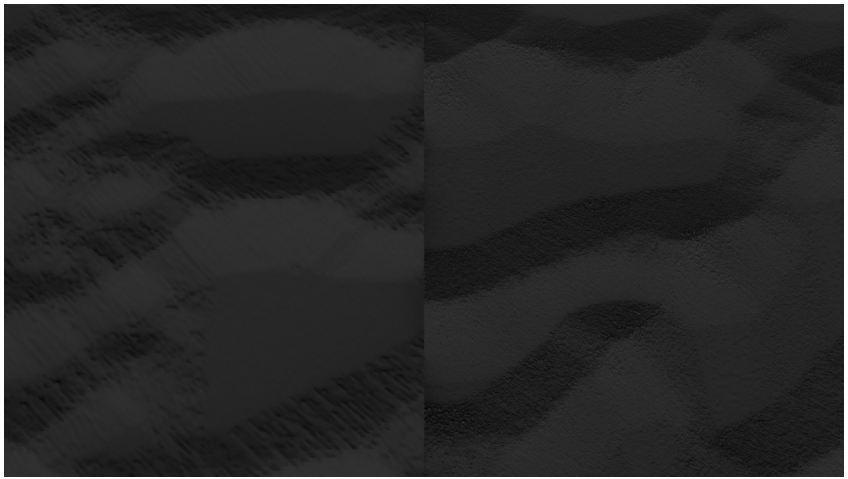


Our Approach

Increased Resolution in Last Steps

Same Coarse Structure with Finer Details:





But what is the last time step?

Tracking Back Snaps

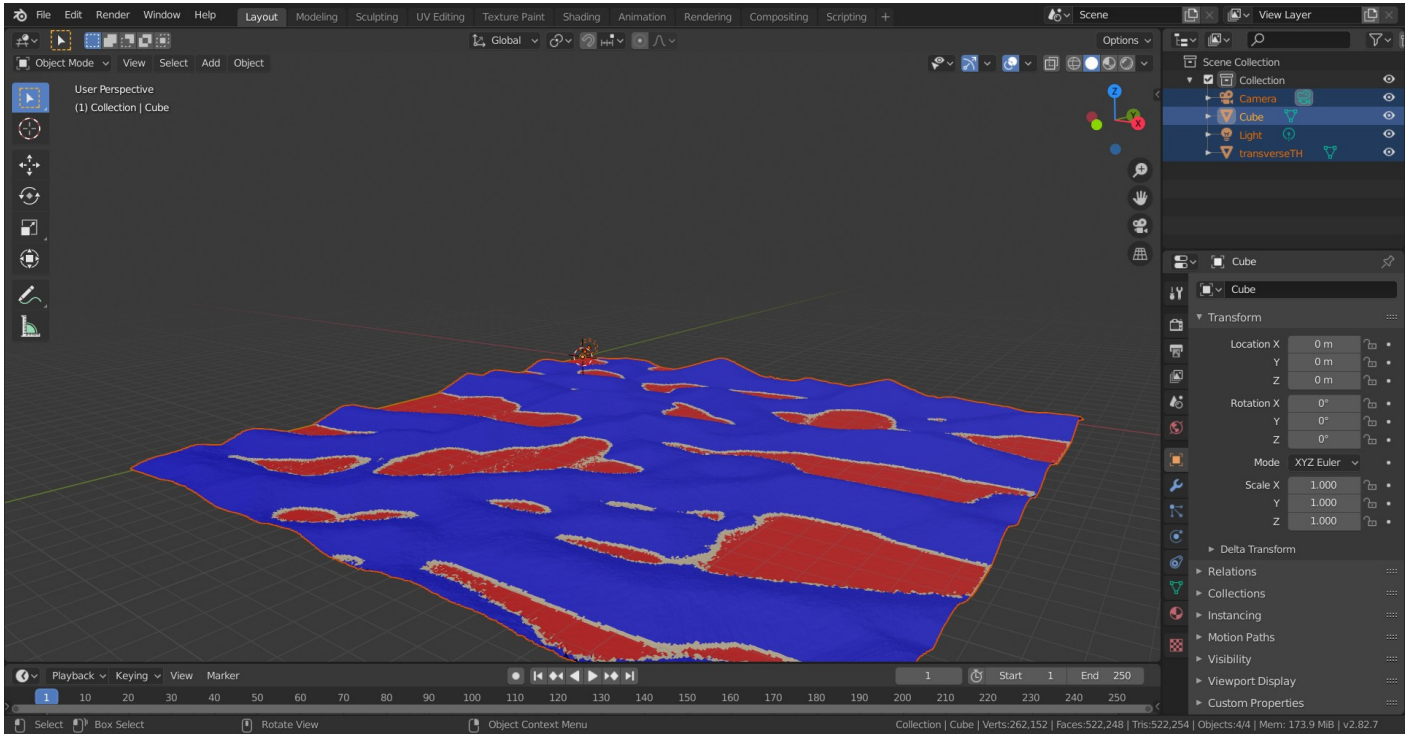
Generate a new dune object from an existing one (snapshot).

Maintain a queue of copies of the dune - FIFO.
(Length decided by user)

User specified simulation end.
Fetch oldest copy from queue.

Copy of dune with Subdividing factor>1 and
Rerun last few steps.

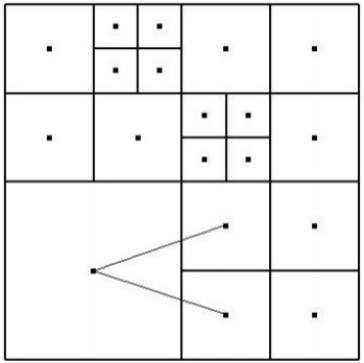
Coloring Mechanism

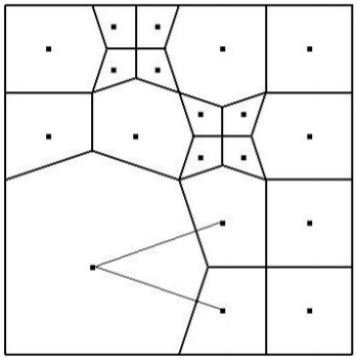


Quad tree(s)

Features	Changes
● Queries for fetching: ○ Leaves ○ Neighbours ○ Bounds ○ CellData corresponding to point ● Separate CellData object - only for a leaf ● roots[nx][ny] for random access	● Gradient Computation ● Mesh Generation ● Saltation, Reptation, Avalanching, abrasion ● Every other function accessing the grid. (Most of the simulation!)

Gradients

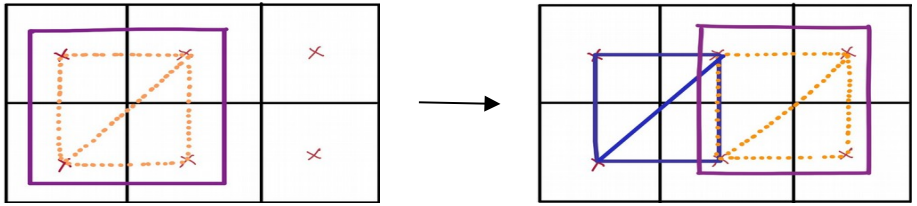


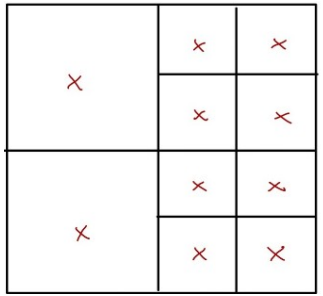


$$(\nabla u)^A = \frac{1}{|p|} \sum_{\sigma \in \mathcal{E}_p} |\sigma|(u_\sigma - u_p)\mathbf{n}_{p\sigma}$$

```
contact = 0;
//diagonal cases should show 0 contact!
if (nei[i]->topRight.x == cBL.x || nei[i]->botLeft.x == cTR.x) {
    //vertical edge is common
    contact = fmin(nei[i]->topRight.y, cTR.y) - fmax(nei[i]->botLeft.y, cBL.y);
    if (nei[i]->topRight.x == cBL.x) { //nei to the left
        ret.x += (contact / perimeter) * (curr->vals[c] - nei[i]->vals[c]);
    }
    else {
        ret.x -= (contact / perimeter) * (curr->vals[c] - nei[i]->vals[c]);
    }
}
else {
    //horizontal edge is common
```

Mesh Generation





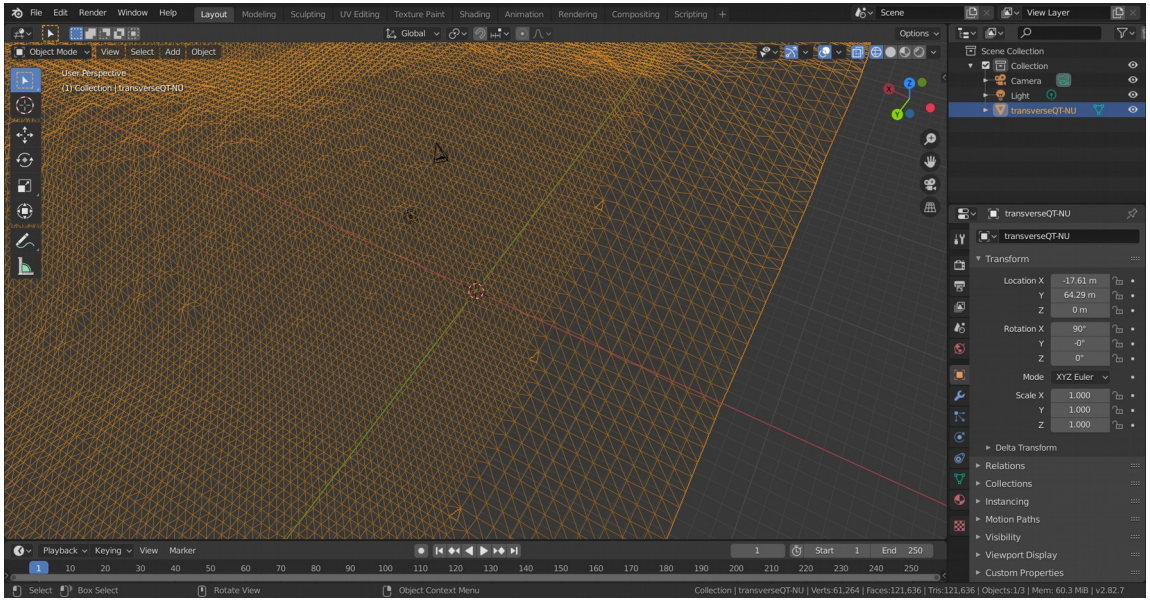
?

```
float fracx = cdList[k]->botLeft.x - float(i);
float fracy = cdList[k]->botLeft.y - float(j);

long basex = int( (std::pow(2, maxlevels-1) * fracx)  + 0.00000001);
basex += (std::pow(2, maxlevels-1) * i);
long basey = int( (std::pow(2, maxlevels-1) * fracy)  + 0.00000001);
basey += (std::pow(2, maxlevels-1) * j);
int pseudoCells = std::pow(2, maxlevels -1 - cdList[k]->qd->level);

for (long p = 0; p < pseudoCells; p++) {
    for (long q = 0; q < pseudoCells; q++) {
        indexMap[basex + p][basey + q] = id;
    }
}
```

```
if ((a - b) * (b - c) * (c - a) != 0)
```

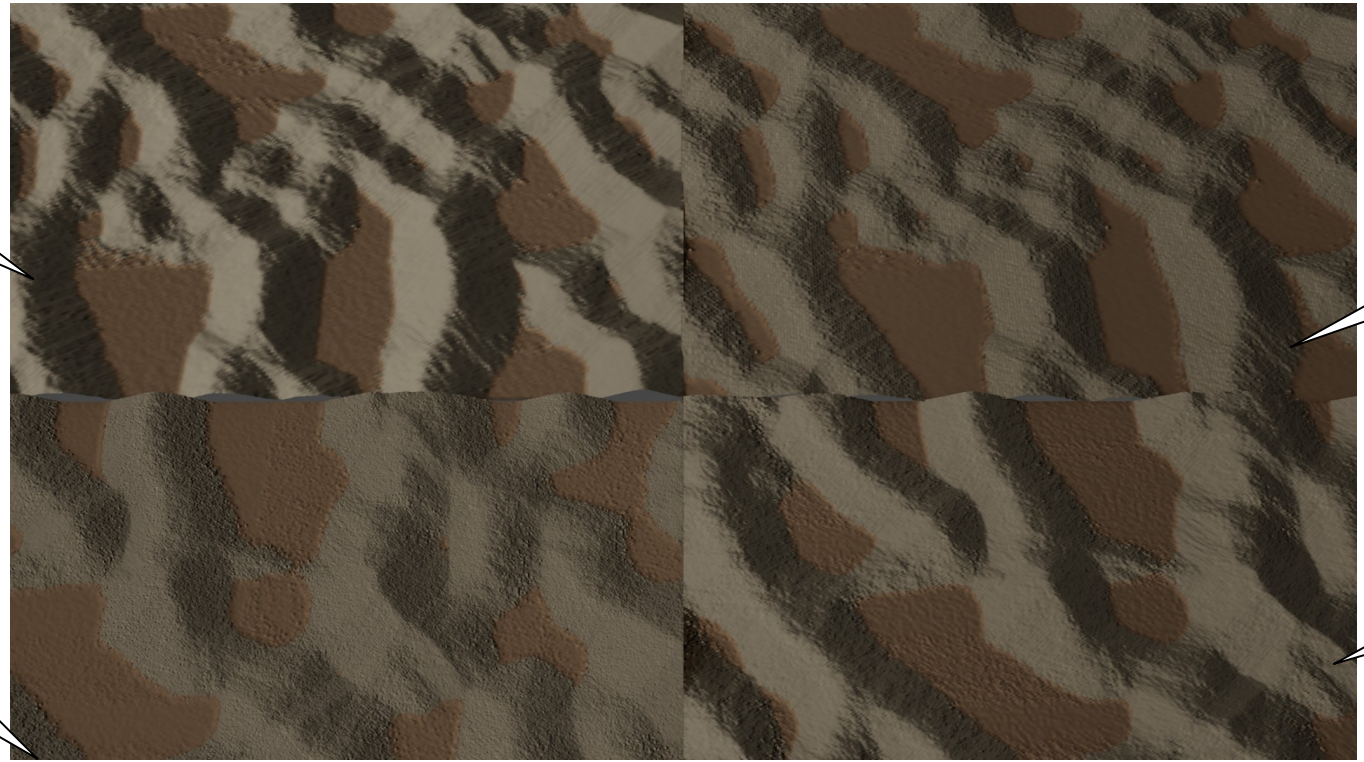
Query Callback for NU Mesh

```
//QT compatible:
DuneSediment(const DuneSediment lowDune, std::function<bool(const Vector2i&, const DuneSediment& )> cellQuery, const int numLevels);
```



Demo

Results



Original
256x256

High Sediment Regions Optimized with Quad-trees

Resolution Doubled + Optimized with Quad-tree

Resolution doubled at last 10 time steps

Future Work

- Impact of Resolution on Thresholds
- Automatic Query Synthesis
- Generalization to other erosion models
- Optimized Implementation
- Other Grid Types

Acknowledgements

Prof. Parag Chaudhuri for providing us with direction and considerate supervision.

Authors: Axel Paris, Adrien Peytavie, Éric Guérin, Oscar Argudo, Éric Galin - for their exceptional work on which we could build upon.

Teachers and fellow students for help that is hard to delineate but impossible to deny.

Thank you!