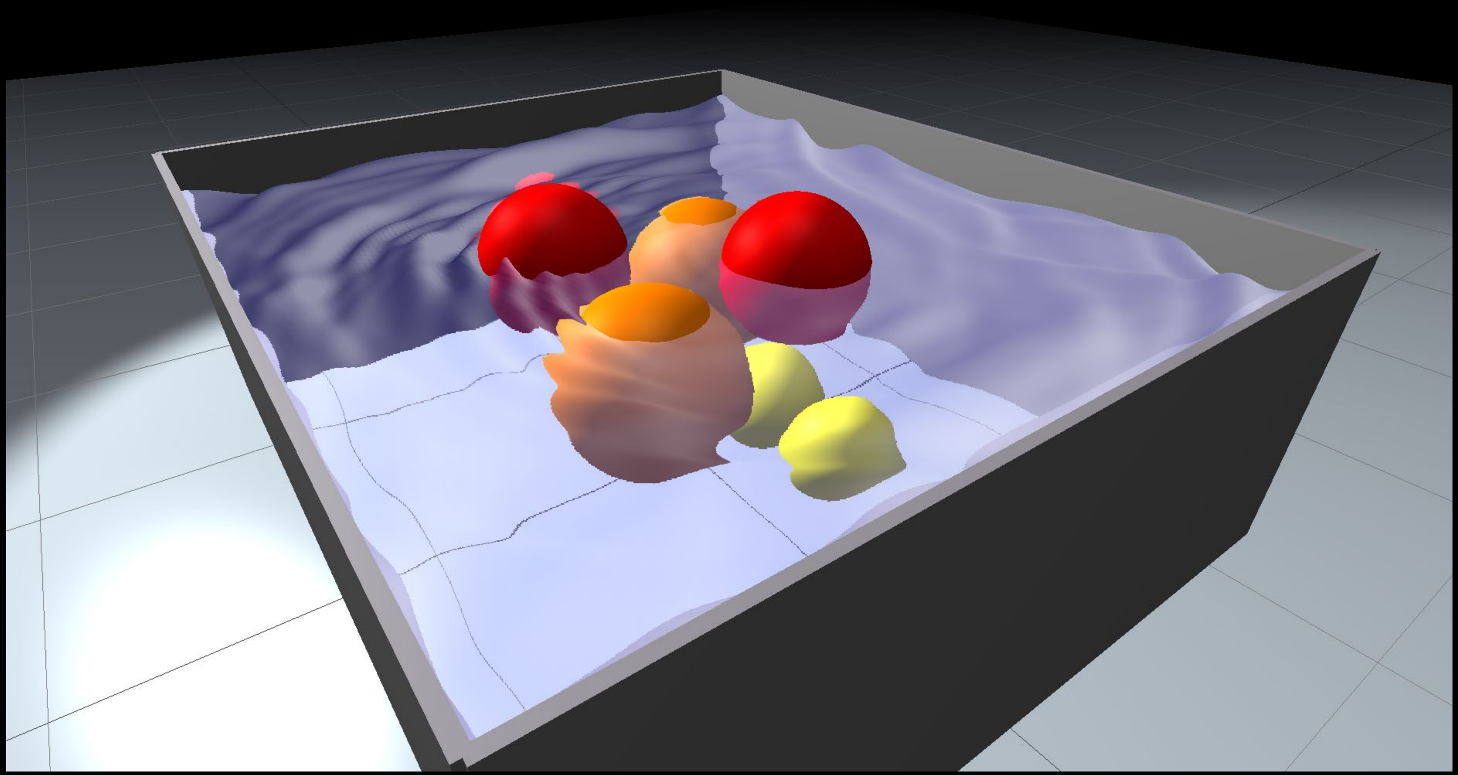


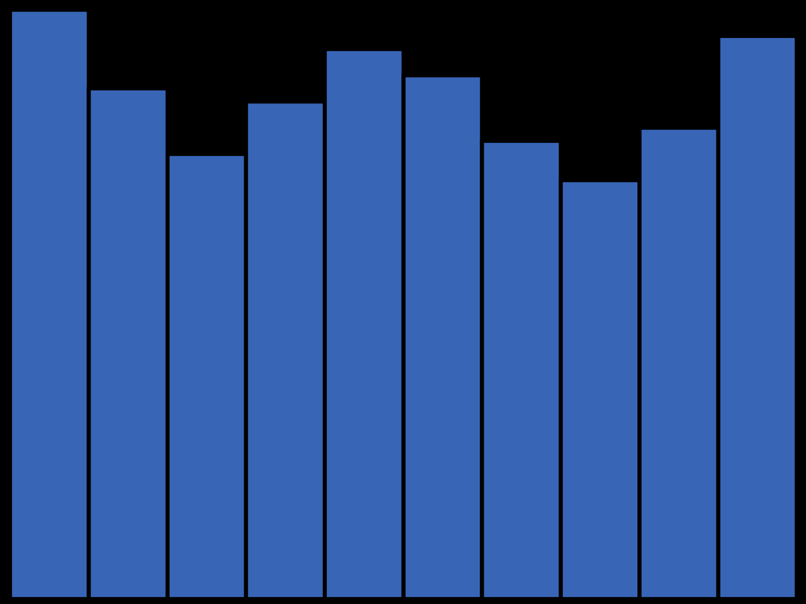
How to write a Water Simulator

Matthias Müller, Ten Minute Physics

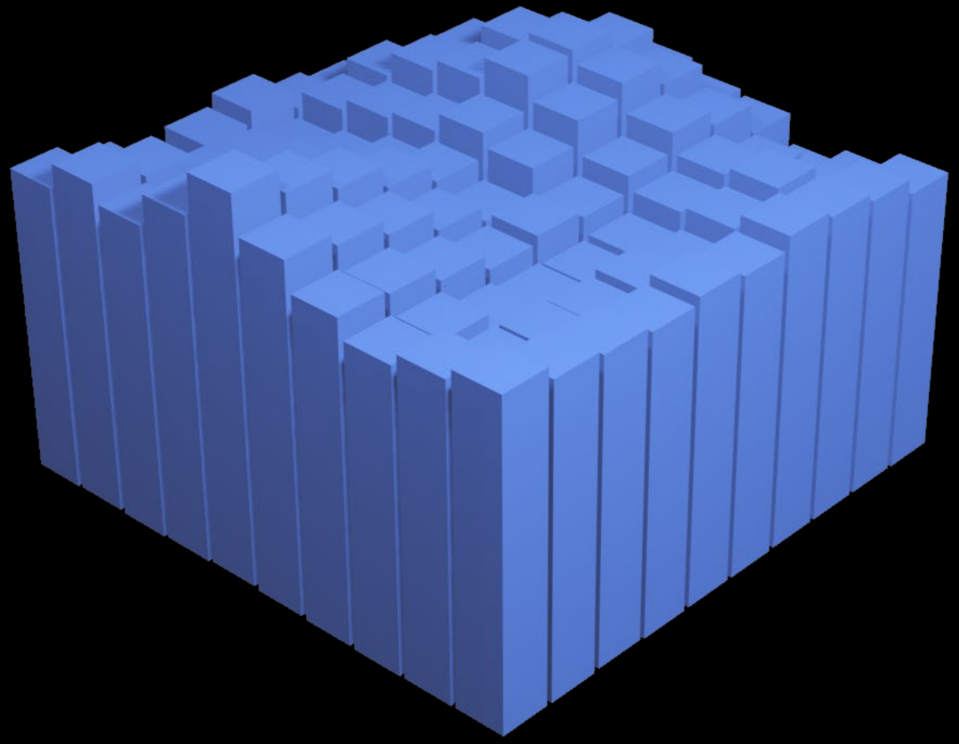
For the code and the demo see:

www.matthiasmueller.info/tenMinutePhysics

Water as an Array of Columns

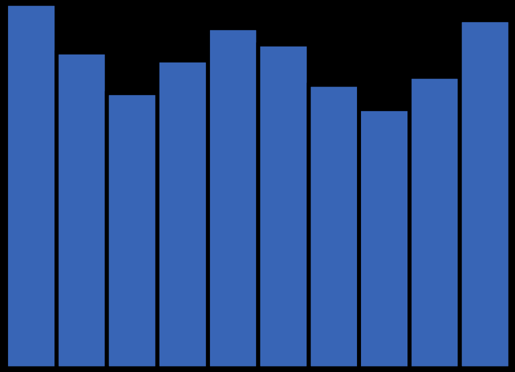


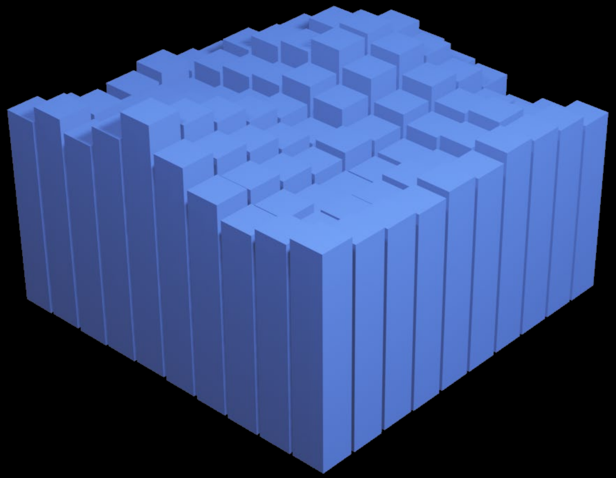
1.5 d



2.5 d

Water as an Array of Columns





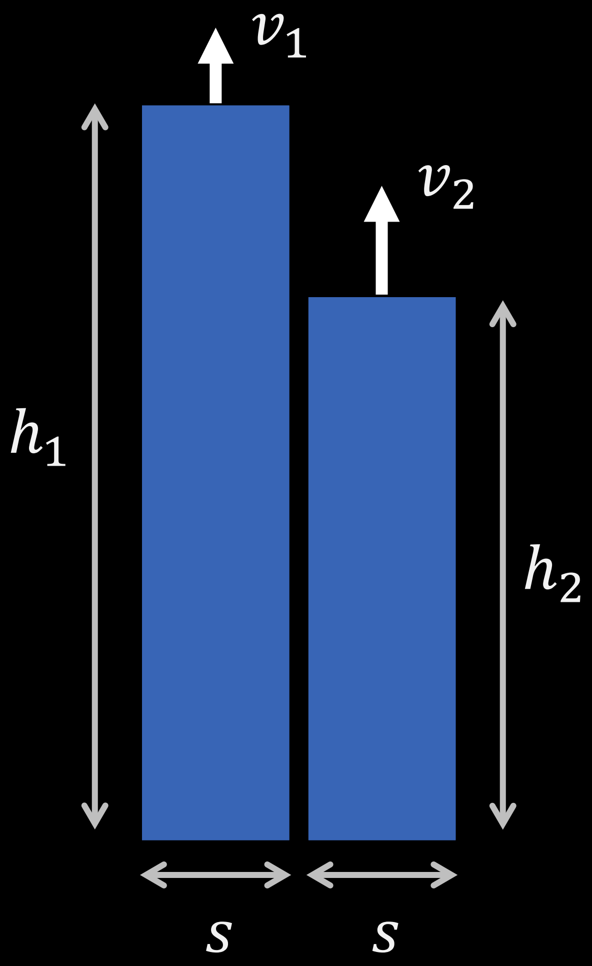
Pros

- Simple to simulate
- Fast
- Easy to extract the surface

Cons

- No overturning waves
- No splashes (add particles)

Grid Setup



- Each column stores a height h and a velocity v
- Both needed for a dynamic simulation
- The width s is the same for all columns

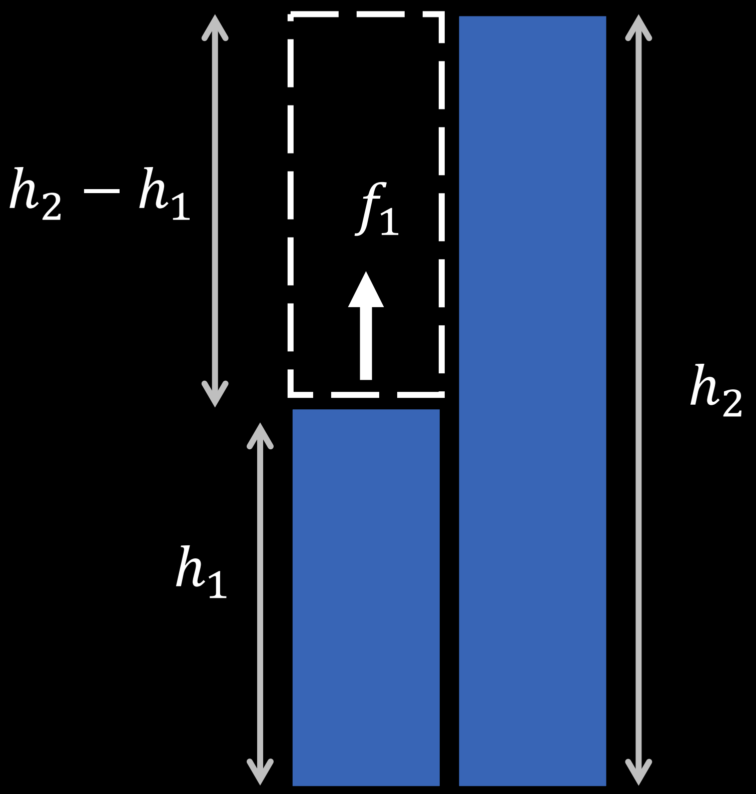
Simulation

Archimedes' principle

Eureka!

Any object, totally or partially immersed in a fluid or liquid,
is buoyed up by a force equal to the weight of the fluid displaced by the object.

Simulation



- From Archimedes:

  $f_1 \sim h_2 - h_1$

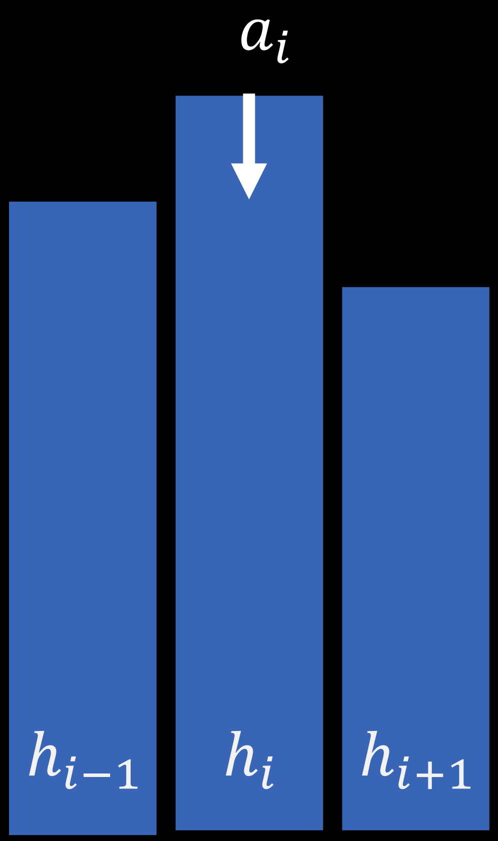
- From Newton’s second law:

  $a_1 \sim f_1$

- From conservation of volume:

  $a_2 = -a_1$

In the 1.5d Grid



- Contribution from both neighbors:

$$a_i \sim (h_{i+1} - h_i) - (h_i - h_{i-1})$$

$$a_i \sim h_{i-1} + h_{i+1} - 2h_i$$

$$a_i = k\,(h_{i-1} + h_{i+1} - 2h_{i,})$$

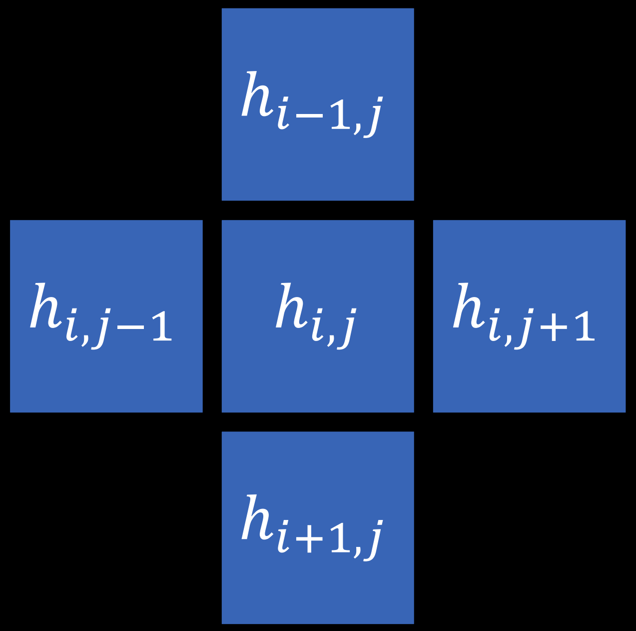
- From the discretization of the wave equation:

$$k = \frac{c^2}{s^2}$$

- The constant c is the wave speed, s the column width

In the 2.5d Grid

top view:



- With four neighbors:

  $$a_{i,j} \sim h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1} - 4h_{i,j}$$

- With the constant of proportionality:

$$a_{i,j} = \frac{c^2}{s^2}\left(h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1} - 4h_{i,j}\right)$$

- Reflecting boundary condition:

  If the neighbor $h_{i\pm1,j\pm1}$ lays outside the domain replace it with $h_{i,j}$

The Simulation Algorithm:

- For all cells i, j in the domain:

$$a_{i,j} \leftarrow \frac{c^2}{s^2}(h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1} - 4h_{i,j})$$

$$v_{i,j} \leftarrow v_{i,j} + \Delta t\, a_{i,j}$$

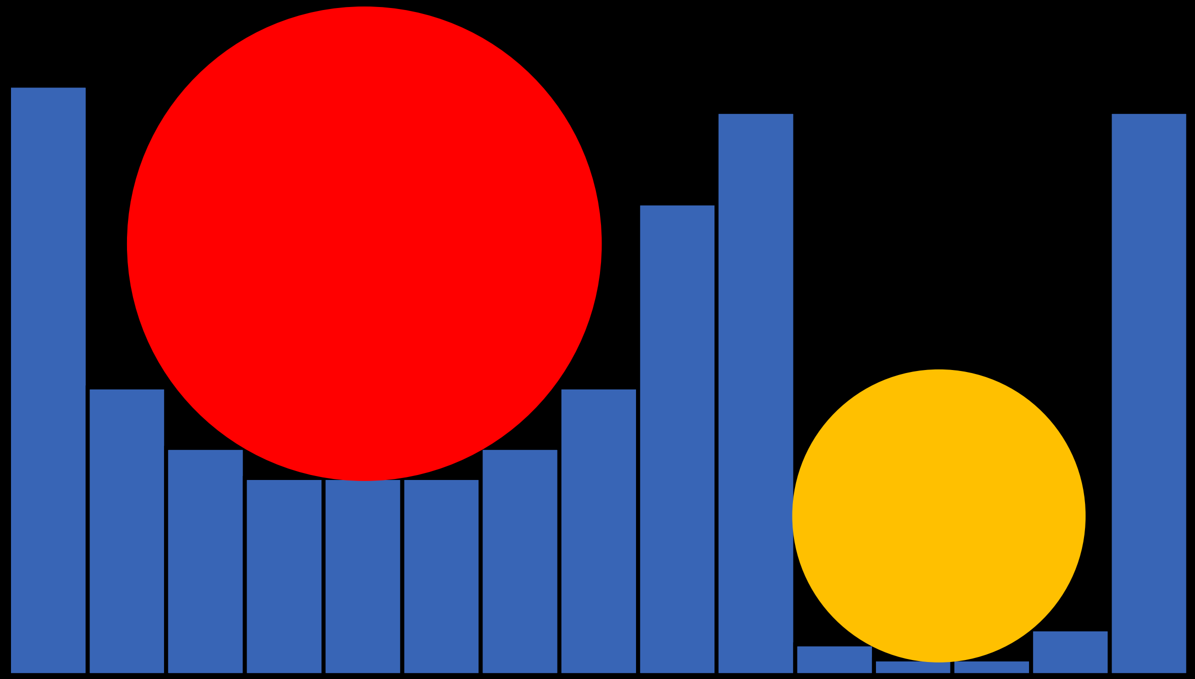
- For all cells i, j in the domain:

$$h_{i,j} \leftarrow h_{i,j} + \Delta t\, v_{i,j}$$

- Semi implicit Euler integration with time step size Δt
- Stability criterion (CFD): $\Delta t\, c < s$

Object Interaction

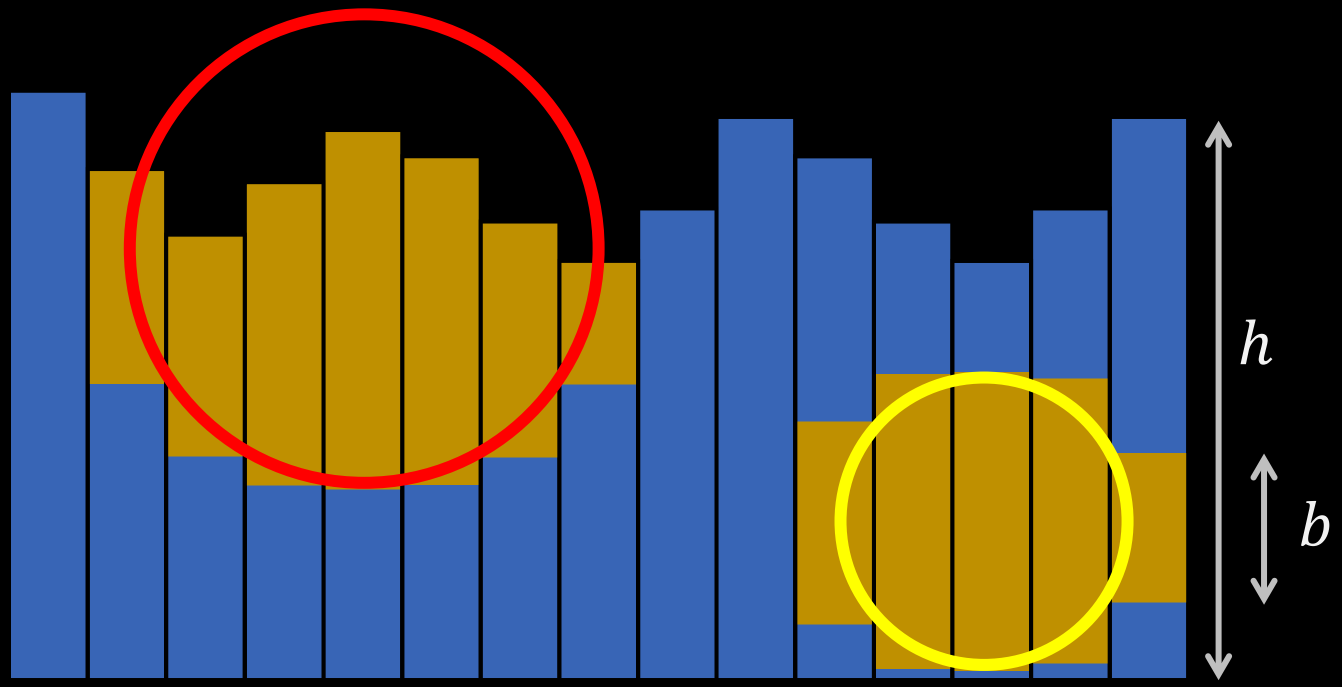
Object To Water First Try



Push columns down

- Volume loss
- Does not work
  for submerged bodies

Object To Water – my Solution



Use an additional field $b_{i,j}$

$h_{i,j}$ stores the total height of the column

$b_{i,j}$ stores the height covered by objects

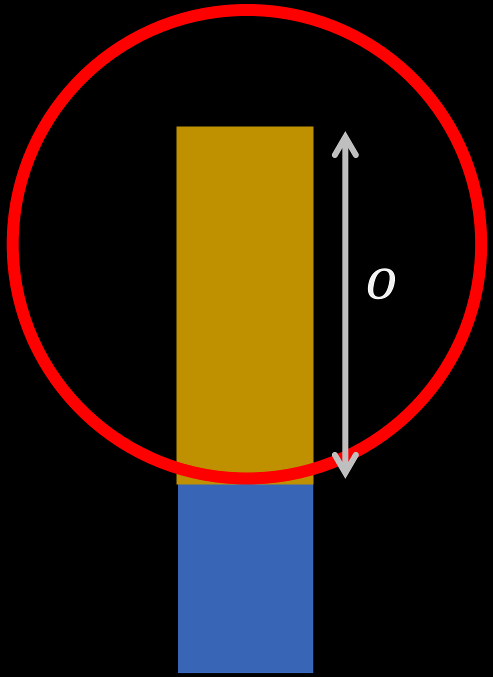
Water Update

- For all cells i, j in the domain:

$$h_{i,j} \leftarrow h_{i,j} + \alpha \left(b_{i,j} - b_{i,j}^{\text{prev}} \right)$$

- Add the change of $b_{i,j}$ to the heights
- Can be positive or negative, no bias, volume conservation
- The parameter $0 \leq \alpha \leq 1$ defines the intensity of the effect
- Smooth $b_{i,j}$ to prevent spikes and instabilities

Water to Object



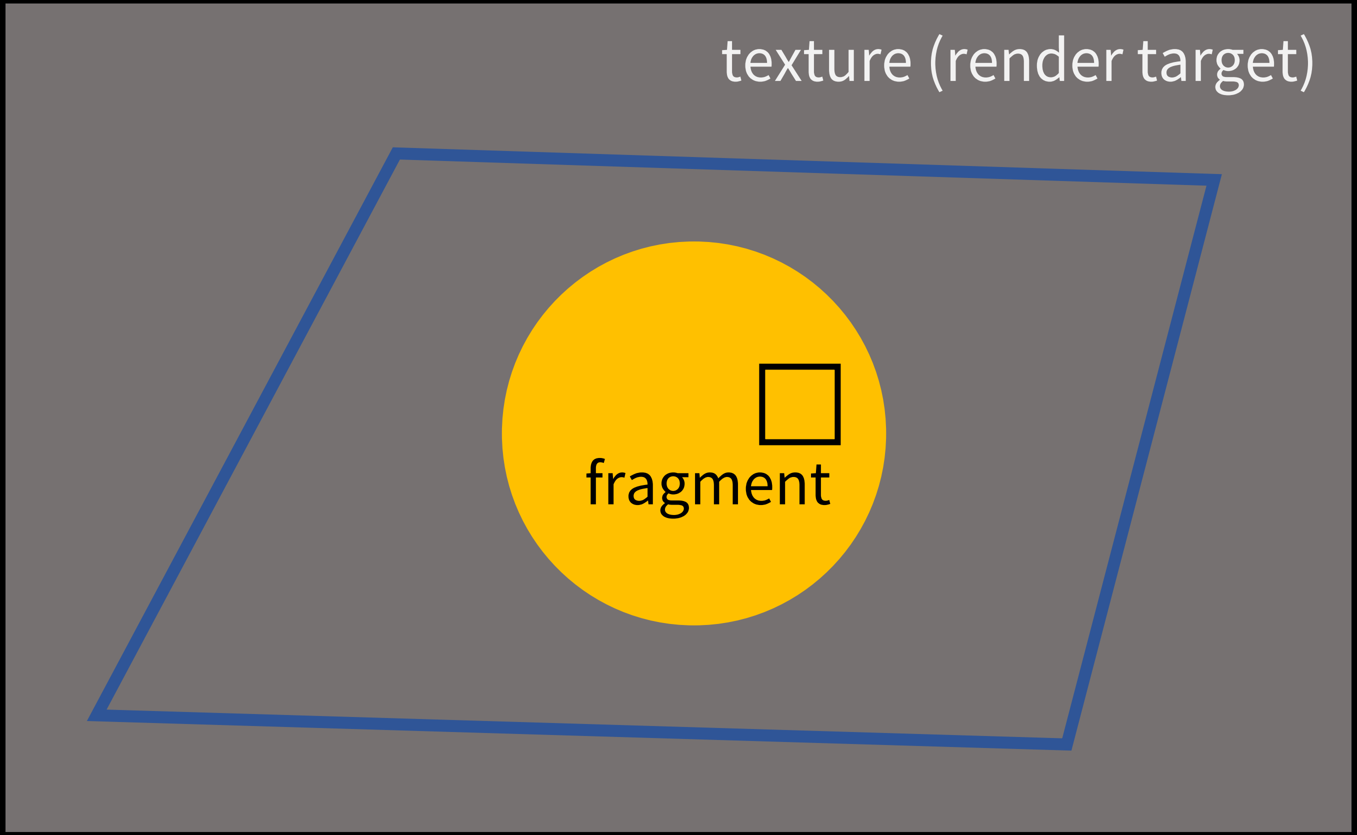
For each overlap of an object with a water column:

apply the force $f = m\,g = \rho_{\text{water}}\,o\,s^2 g$

to the object at the position of the column,
where g is the gravitational acceleration.

Rendering

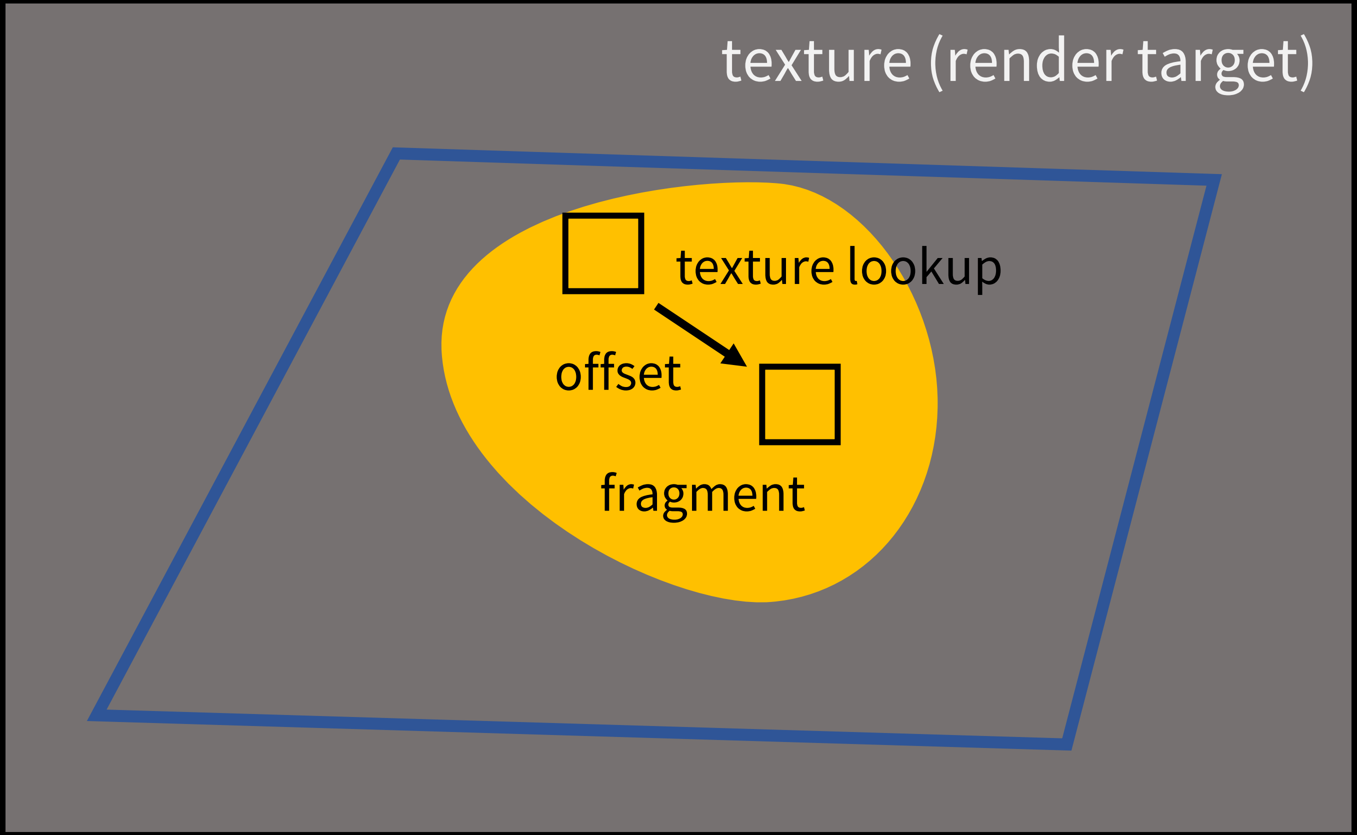
Render a Transparent Plane



- Render the scene behind the plane
  to the texture
  using the current camera

- Use the screen coordinates of the fragment to locate the color in the texture

Add Refraction Effect



- Use the screen coordinates of the fragment plus an offset to locate the color in the texture.
- Make direction of the offset dependent on the surface normal.
- Make length of the offset dependent the distance to the camera

Let’s have a look at the code…