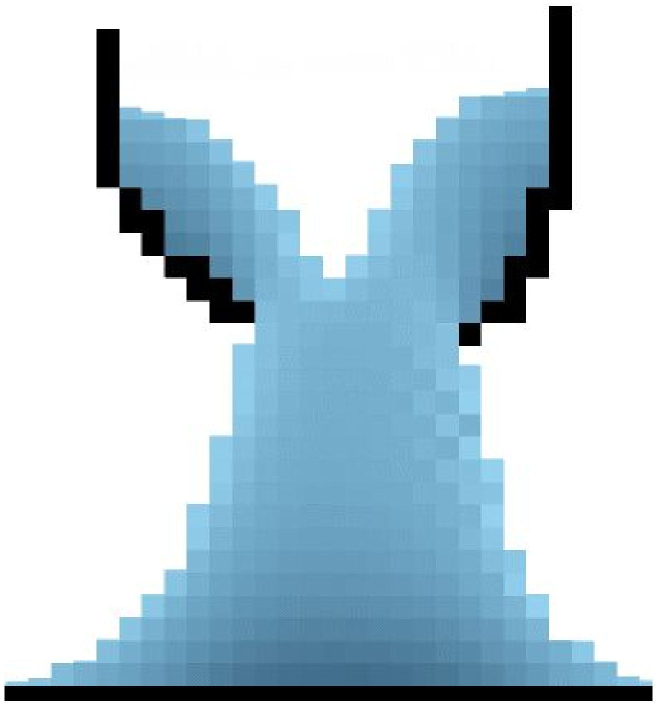
Liquid Physics Using Cellular Automation

One interesting way to represent liquids in a grid based world, is to use a form of cellular automaton. You may have heard of cellular automata from the popular “Conway’s Game of Life”, where cells evolve based on a set of rules that they adhere to.

As the cells evolve, they create some interesting patterns, sometimes even chaotic in nature. For our liquid simulation, we are going to want to enforce some strict rules, so that we have much more control over the movement of our cells.

Take a look at the following gif. Notice how the liquids are all contained inside of grid cells. Each cell is capable of holding a value to indicate how much liquid is contained inside of it.



In this simulation, the cells attempt to disperse their liquid into their neighboring cells. It will do this by following rules that we will set for them. In each iteration of our simulation, every cell that contains liquid will execute the same set of rules.

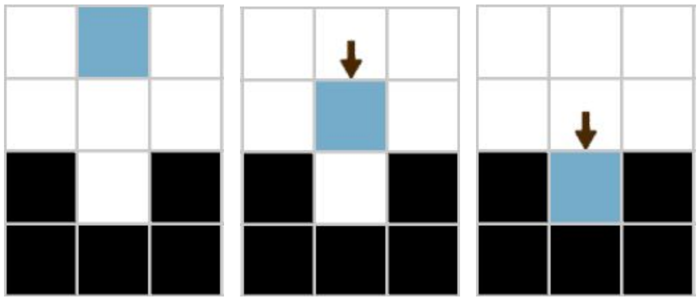
Rule #1: Flowing Into Bottom Neighboring Cell

First, a cell will check to see if it is capable of flowing downwards into its bottom neighbor. This makes sense as we want liquid to fall down as if it was affected by gravity.

The amount of liquid that is allowed to flow into another cell is calculated based on the amount of liquid already in the source and destination cells. Liquid will only flow if the destination cell has less liquid than the source cell.

Once we have calculated the amount of liquid to flow, we make the adjustment to the liquid value of both cells to reflect the change. The images below (from left to right) depict how the liquid behaves, showing one iteration of the simulation per image.

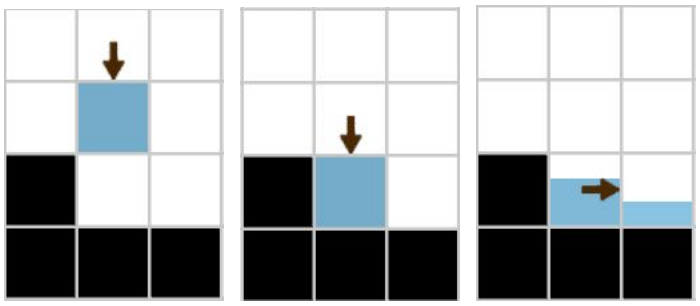
The liquid falls into the cell below in each iteration, until it is unable to fall any more:



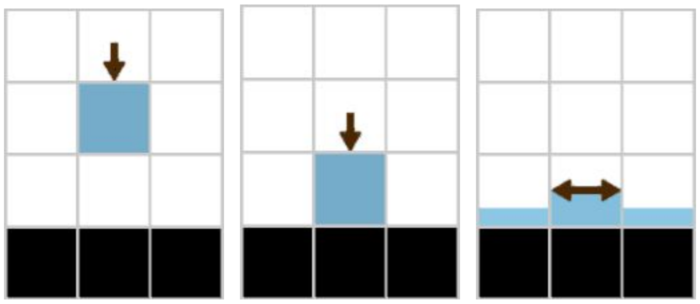
Rule #2: Flowing Into Left and Right Neighboring Cells

Once the cell has finished executing the first rule, it will then attempt to distribute its remaining liquid towards the left or right neighboring cells. Again, the amount of liquid that is allowed to flow is dependent on the current amount of liquid in each cell.

In the following images, you see that the liquid moves down during the first iteration. During the second iteration, it can no longer flow downwards, therefore it flows into the adjacent cell that is available to its right:

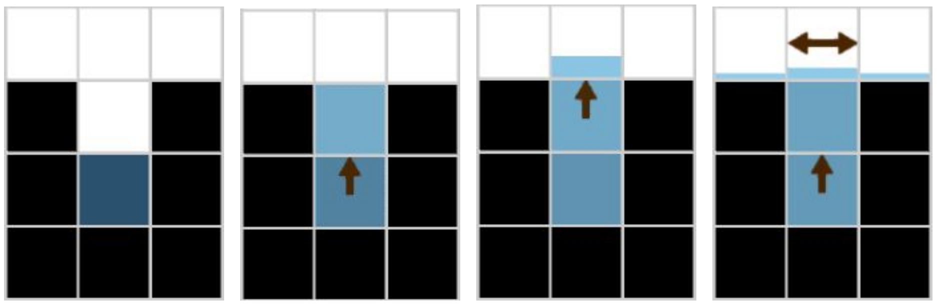


If both neighboring left and right cells are available to accept liquid, then the cell will flow in both directions in the same iteration:



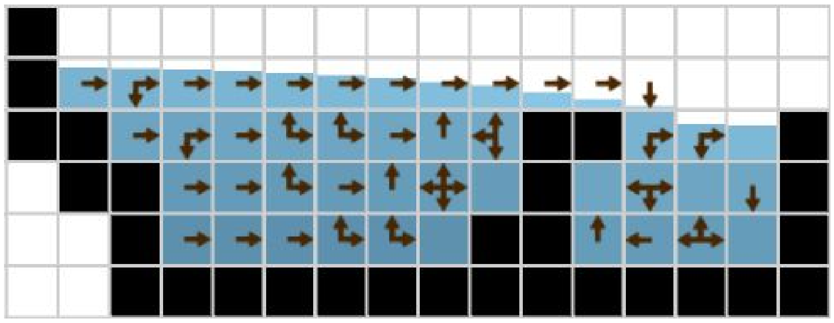
Rule #3: Flowing Upwards with Pressure

After the cell has executed both Rule #1 and Rule #2, it will then verify if it still contains more liquid than the maximum amount allowed. If it does contain more liquid, then the cell becomes pressurized. A pressurized cell will be allowed to flow upwards if possible. This behavior is illustrated below:



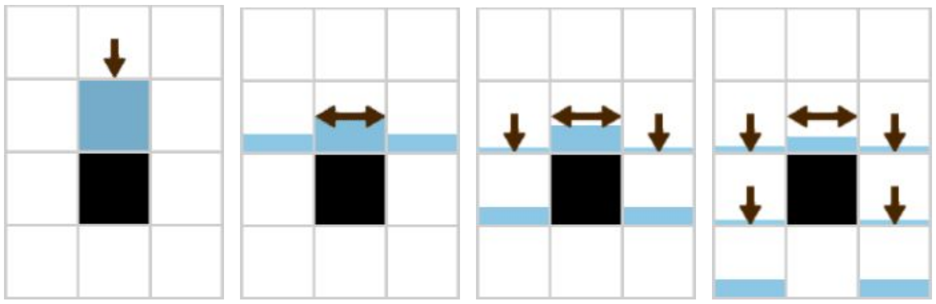
Determining the Flow Direction of each Cell

As we iterate over each of our cells that contain liquid, and apply the above rules, we can also keep track of the last iterations flow direction for each of our cells. Doing so, will allow us to know in what direction each cell is flowing at all times.



Rendering Falling Liquids

Liquids that are flowing downwards do not render very nicely, as they sometimes only contain a small amount of liquid in the cells. Rendering the amount of liquid directly in these cells can look somewhat strange:



If we add a check to see if the liquid flowing downwards, we can render these cells as having a full unit of liquid. This is mainly to make the simulation render a little nicer. Doing so, would render the liquids like this: