Chapter 11

Shallow Water Equations

Nils Thuerey, Peter Hess

11.1 Introduction

The shallow water equations (SWE) are a simplified version of the more general Navier-Stokes (NS) equations, which are commonly used to describe the motion of fluids. The SWE reduce the problem of a three-dimensional fluid motion to a two-dimensional description with a height-field representation. From now on, we will use the following notation (it is also illustrated in Fig. 11.1):

- $h$ denotes the height of the fluid above zero-level.
- $g$ is the height of the ground below the fluid (above zero-level).
- $\eta$ denotes the height of the fluid above ground, $\eta = h - g$.
- $v$ the velocity of the fluid in the horizontal plane.

A basic version of the SWE can be written as

$$\frac{\partial \eta}{\partial t} + (\nabla \eta) \cdot v = -\eta \nabla \cdot v \quad (11.1)$$
$$\frac{\partial v}{\partial t} + (\nabla v) \cdot v = a_n \nabla h \quad (11.2)$$

where $a_n$ denotes a vertical acceleration of the fluid, e.g., due to gravity. This formulation can be derived from the NS equations by, most importantly, assuming a hydrostatic pressure along the direction of gravity. Interested readers can find a detailed derivation of these equations in Section A.

In the following sections we will first explain how to solve these equations with a basic solver, and then extend this solver with more advanced techniques to handle open boundaries, or free surfaces.
A basic solver of the SWE has to compute the change of the height and velocity values of the water surface over time. According to Eq. (11.2) and Eq. (11.1) the equations for the water height $\eta$ and the velocity $\mathbf{v} = (v_1, v_2)$ can be written as:

$$\begin{align*}
\frac{\partial \eta}{\partial t} + (\nabla \eta) \cdot \mathbf{v} &= -\eta \nabla \cdot \mathbf{v} \\
\frac{\partial v_1}{\partial t} + (\nabla v_1) \cdot \mathbf{v} &= a_n \nabla h \\
\frac{\partial v_2}{\partial t} + (\nabla v_2) \cdot \mathbf{v} &= a_n \nabla h.
\end{align*} \quad (11.3)$$

In this form, the two distinct parts of the equations can be identified: the left side accounts for the advection within the velocity field $\mathbf{v}$, while the right side computes an additional acceleration term. Here, we will use an explicit time integration scheme, as this makes the solution of the SWE significantly more simple. Alternatively, implicit schemes, as described in [LvdP02] could be used. These, however, require a system of linear equations to be solved for the update, and, with simpler methods such as implicit Euler, introduce a significant amount of damping.

To compute a solution for these equations, we first discretize the domain with $n_1$ cells in x-direction, and $n_2$ cells in y-direction. For simplicity, we assume in the following that the cells have a square form with side length $\Delta x$. The gravity force $a_n$ is assumed to act along the z-axis, e.g. $a_n = 10$, and the size of a single time step is given by $\Delta t$. The overall height of the water, and the strength of the gravity will later on determine the speed of the waves travelling on the surface. To represent the three unknowns with this grid we use a staggered grid. This means that the pressure is located in the center of a cell, while the velocity components are located at the center of each edge, as shown in Fig. 11.1. The staggered grid is commonly used for fluid solvers, and prevents instabilities that would result from a discretization on a co-located grid. An update step of the shallow water solver consists of the following parts: first all three fields are advected with the current velocity field. This means that the pressure is located in the center of a cell, while the velocity components are located at the center of each edge, as shown in Fig. 11.1. The staggered grid is commonly used for fluid solvers, and prevents instabilities that would result from a discretization on a co-located grid. An update step of the shallow water solver consists of the following parts: first all three fields are advected with the current velocity field. Afterwards, the acceleration terms are computed for the height and velocity fields. The following pseudo-code illustrates a single step of the simulation loop of a simple shallow water solver:
Shallow-water-step(η, v, g)
(1) \(\eta = \text{Advect}(\eta, v)\)
(2) \(v_1 = \text{Advect}(v_1, v)\)
(3) \(v_2 = \text{Advect}(v_2, v)\)
(4) \(h = \eta' + g\)
(5) \(\text{Update-velocities}(h, v_1, v_2)\)

Note that the three calls of the \textit{Advect(...)} function return a value that is assigned back to the original input grid (e.g., in line 1 \(\eta\) is a parameter of the call, and used in the assignment). This should indicate that the advection requires a temporary array, to which the advected values are written, and which is copied back to the original grid after finishing the advection step.

To compute the advection, we can use the semi-Lagrangian method [Sta99] to compute a solution without having to worry about stability. This algorithm computes the advection on a grid by essentially performing a backward trace of an imaginary particle at each grid location. Given a scalar field \(s\) to be advected, we have to compute a new value for a grid cell at position \(x\). This is done by tracing a particle at this position backward in time, where it had the position \(x^{t-1} = x - \Delta t v(x)\). We now update the value of \(s\) with the value at \(x^{t-1}\), so the new value is given by \(s(x)' = s(x^{t-1})\). Note that although \(x\) is either the center or edge of a cell in our case, \(x'\) can be located anywhere in the grid, and thus usually requires an interpolation to compute the value of \(s\) there. This is typically done with a bi-linear interpolation, to ensure stability. It guarantees that the interpolated value is bounded by its source values from the grid, while any form of higher order interpolation could result in larger or smaller values, and thus cause stability problems. The advection step can be formulated as

\textit{Advect}(s, v)
(1) \textbf{for} \(j = 1 \textbf{ to } n_2 - 1\)
(2) \textbf{for} \(i = 1 \textbf{ to } n_1 - 1\)
(3) \(x = (i \cdot \Delta x, j \cdot \Delta x)\)
(4) \(x' = x - \Delta t \cdot v(x)\)
(5) \(s'(i, j) = \text{interpolate}(s, x')\)
(6) \textbf{endfor}
(7) \textbf{endfor}
(8) \textbf{return}(s')

Note that, due to the staggered grid, the lookup of \(v(x)\) above already might require an averaging of two neighboring velocity components to compute the velocity at the desired position. This is also, why the three advection steps cannot directly performed together - each of them requires slightly different velocity interpolations, and leads to different offsets in the grid for interpolation.

The divergence of the velocity field for the fluid height update can be easily computed with finite differences on the staggered grid. So, according to Eq. (11.3), the height update is given by
Update-height($\eta, v$)

(1) $\text{for } j = 1 \text{ to } n_2 - 1$

(2) $\text{for } i = 1 \text{ to } n_1 - 1$

(3) $\eta(i, j) = \eta(i, j) \cdot \left( \frac{(v_1(i+1, j) - v_1(i, j))}{\Delta x} + \frac{(v_2(i, j+1) - v_2(i, j))}{\Delta x} \right) \Delta t$

(5) $\text{endfor}$

(6) $\text{endfor}$

(7) $\text{return}(\eta')$

In contrast to the advection steps, adding the accelerations can be directly done on the input grids. Similarly, the acceleration term for the velocity update is given by the gradient of the overall fluid height. Note that in this case, the total height above the zero-level is used instead of the fluid height above the ground level. This is necessary, to, e.g., induce an acceleration of the fluid on an inclined plane, even when the fluid height itself is constant (all derivatives of $\eta$ would be zero in this case). The parameter $a$ for the velocity update below is the gravity force.

Update-velocities($h, v_1, v_2, a$)

(1) $\text{for } j = 1 \text{ to } n_2 - 1$

(2) $\text{for } i = 2 \text{ to } n_1 - 1$

(3) $v_1(i, j) = a \left( \frac{h(i-1, j) - h(i, j)}{\Delta x} \right) \Delta t$

(5) $\text{endfor}$

(6) $\text{endfor}$

(7) $\text{for } j = 2 \text{ to } n_2 - 1$

(8) $\text{for } i = 1 \text{ to } n_1 - 1$

(9) $v_2(i, j) = a \left( \frac{h(i, j-1) - h(i, j)}{\Delta x} \right) \Delta t$

(10) $\text{endfor}$

(11) $\text{endfor}$

This concludes a single step of a basic shallow water solver. Note that the steps so far do not update the values at the boundary of the simulation domain, as we cannot compute any derivatives there. Instead, special boundary conditions are required at the border, and can be used to achieve a variety of effects. These will be the topic of the next section.

11.3 Boundary Conditions

In the following, we will describe different types of boundary conditions: reflecting and absorbing boundaries, as well as a form of free surface boundary conditions. The former can be used to model a wall that reflects incoming waves. The second type can be used to give the effect of an open water surface, as waves will simply leave the computational domain. Free surface boundary conditions can be used once the fluid should, e.g., flow through a landscape. Although the boundary conditions will be described to handle the outermost region of the computational domain, they can likewise be used to, e.g., create a
Reflecting Boundaries

In the following, we will, without loss of generality, consider the boundary conditions for cells at the left boundary. Reflecting boundary conditions are achieved by setting the velocities at the boundary to zero (after all, there should be no flux through the wall). In addition, we mirror the height of the fluid in the outermost layer. We thus set:

\[
\begin{align*}
    h(0, j)' &= h(1, j) \\
    v_1(1, j)' &= 0 \\
    v_2(0, j)' &= 0.
\end{align*}
\]  

(11.4)

Note that we do not modify the y-component \( v_2 \) of the velocity field. The fluid is thus allowed to move tangentially to a wall. Theoretically, we could also enforce different behaviors for the tangential velocities, but in practice this does not make a noticeable difference. Also note, that we only set \( v_1(1, j) \), as \( v_1(0, j) \) is usually never accessed during an computation step.
Absorbing Boundaries

Surprisingly, it is more difficult to achieve absorbing boundaries than reflecting ones. The problem of boundaries simulating an infinite domain is already known for a long time (see, e.g., [Dur01] for details). A commonly used method to achieve this, is the perfectly matched layer introduced by [Ber94], requires an additional layer of computations around the actual domain.

This is why we chose to use the Higdon boundary conditions [Hig94] which are less accurate but can be more efficiently computed than PML. Below is the $p^{th}$ order Higdon boundary condition, where the velocities $c_j$ are chosen to span the range of incoming wave velocities.

$$\left(\prod_{j=1}^{p} \left( \frac{\partial}{\partial t} + c_j \frac{\partial}{\partial x} \right) \right) h = 0$$  \hspace{1cm} (11.5)

This boundary condition can be problematic for higher order approximations, but as the wave propagation speed in shallow water is known to be $c = \sqrt{g\eta}$, this allows us to use to use the 1$^{st}$ order boundary condition

$$\left( \frac{\partial}{\partial t} + c \frac{\partial}{\partial x} \right) h = 0 .$$  \hspace{1cm} (11.6)

This boundary condition actually requires temporal derivatives, so we assume the current heightfield is given by $h'$, while the heights of the previous step are stored in $h'^{-1}$. Hence, we can set the boundary values to:

$$h(0, j)' = \frac{\Delta x h(0, j)^{t-1} + \Delta t c(1, j)' h(1, j)'}{\Delta x + \Delta t c(1, j)'}$$

$$v_1(1, j)' = v_1(1, j)^{t-1} - \frac{\Delta h(1, j)' - h(0, j)' \Delta t}{\Delta x}$$

$$v_2(1, j)' = 0$$  \hspace{1cm} (11.7)

Note that the update of $v_1$ is essentially the same acceleration term on the left hand side of Eq. (A.16). To further suppress any residual reflections at the boundary, we can apply a slight damping of the height field in a layer around the boundary.

Fig. 11.3 shows the effect of these boundary conditions compared to reflecting ones.

For boundaries where fluid should flow into or out of the domain, we can reuse the two types above. Inflow boundary conditions can be achieved by specifying reflecting ones, with an additional fixed normal velocity. For outflow boundary conditions, absorbing ones with free normal velocities are more suitable.

Free Surfaces

Often, shallow water simulations assume a completely fluid domain, since this makes solving the SWE quite straightforward. Once applications like a river, or fluid filling an arbitrary terrain are needed, this is not sufficient anymore. Such applications require a distinction between areas filled with fluid, and empty or dry areas. An example can be seen in Fig. 11.4. In the following we will consider this as a problem similar to free surface handling for full fluid simulations. Shallow water simulations naturally have an interface, and thus a free
In addition, we will now prescribe boundary conditions with such a free surface within the simulation plane itself. From the mathematical point of view, a distinction between fluid and dry would not be necessary since the SWE still work if $\eta$ is zero. Distinguishing fluid and dry cells, however, brings some advantages. Foremost, computational time can be saved if large parts of the domain are dry. Therefore, we introduce cell flags $f(i,j)$, that determine the type of each cell, and update them once per time step after updating the heights. This allows us to quickly identify wet and dry cells. Besides the computational advantage, controlling the transition between wet and dry cells also gives us some control over the spreading velocity. Without free surface tracking, the fluid boundary would expand exactly one cell per time step, regardless of cell size and time step length. The height of this advancing boundary would be very small, but this behavior is usually not desired. In addition, we will compute a fill value $r$ for each cell, as this allows us to track a smoothly moving surface line between the fluid and empty cells.

To determine the cell’s flag $f$ we have to compute the minimal and maximal ground level $h_{\min}$ and $h_{\max}$ as well as the maximal fluid depth $\eta_{\max}$ on the cell’s edges.

$$h_{\min}(i,j) = \frac{h(i,j) + \min_{p \in N(i,j)} h(p)}{2} \quad p \in \mathcal{N}(i,j)$$ (11.8)

$$h_{\max}(i,j) = \frac{h(i,j) + \max_{p \in N(i,j)} h(p)}{2} + \varepsilon_H \quad p \in \mathcal{N}(i,j)$$ (11.9)

$$\eta_{\max}(i,j) = \frac{\eta(i,j) + \max_{p \in N(i,j)} \eta(p)}{2} \quad p \in \mathcal{N}(i,j)$$ (11.10)

where $\mathcal{N}(i,j)$ is the set of the four direct neighbors of cell $(i,j)$. Note that we add a small value $\varepsilon_H$ to $h_{\max}$ to prevent $h_{\min}$ to be equal to $h_{\max}$ in flat areas. With these three values we can now determine $f$ as well as the fill ratio $r$ which indicates the cell’s fill level in dependence of the local ground topology $h_{\min}(i,j)$ and $h_{\max}(i,j)$. $r$ can be used to compute
an isoline which defines the border of the rendered fluid surface for rendering the water surface. The following pseudo code shows how $f$ and $r$ are calculated:

\[\text{Compute-flags}(i, j)\]
(1) \textbf{if} \ h(i, j) \leq h_{\text{min}}(i, j) \text{and} \eta_{\text{max}}(i, j) < \epsilon\eta_{\text{max}}
(2) \quad f(i, j) = \text{DRY}
(3) \quad r(i, j) = 0
(4) \quad \textbf{else if} \ h(i, j) > h_{\text{max}}
(5) \quad f(i, j) = \text{FLUID}
(6) \quad r(i, j) = 1
(7) \quad \textbf{else}
(8) \quad f(i, j) = \text{FLUID}
(9) \quad r(i, j) = \left( h(i, j) - h_{\text{min}}(i, j) \right) / \left( h_{\text{max}}(i, j) - h_{\text{min}}(i, j) \right)
(10) \quad \textbf{endif}

A cell is marked as dry if its surface height is not higher than the lowest ground value in the cell and if there is no neighbor cell from which fluid could flow into this cell. The fill
Figure 11.5: A wave flows through an S-shaped river bed, and pushes a large number of rigid bodies along with the flow.

ratio is then set to zero. $\varepsilon_\eta$ can be seen as a threshold which allows inflow from a neighbor cell only if this neighbor has a large enough amount of fluid. This effectively limits the spread of thin layers of fluid. So this could be seen as a simple way of simulating surface tension. A cell is completely filled if its surface height is higher than the ground at any position in the cell. The fill ratio is then set to one. The cell is also marked as fluid if the surface height is only in parts higher than the ground level. In this case however the fill ratio is the ratio between minimal ground level, fluid surface height and maximal ground level.

Note, that with this definition cells may have negative depth values $\eta$ even if they are marked as fluid. There are cases were the cell center itself is dry, so the value of $\eta$ is negative at this point, while the whole cell still contains fluid at the edges of a cell.
Part I

Appendix
Appendix A

Derivation of the Shallow Water Equations

Nils Thuerey, Roland Angst

In this chapter we go through a detailed derivation of a more general form of the shallow water equations (SWE) from the underlying Navier-Stokes (NS) equations. It is based on [Ang07].

A.1 Integral Form

The SWE are a specialized and simplified version of the Navier-Stokes equations. The Navier-Stokes equations describe the dynamics of a fluid much like elasticity theory describes the dynamics of deformable solids. The Navier-Stokes equations relate the dynamics of the density field, the temperature field and the velocity field of a fluid to each other. The full Navier-Stokes equations in three dimensions are a system of partial differential equations with five equations and unknowns (density, velocity and temperature). The Navier-Stokes equations as well as the SWE are derived from three balance equations which prescribe three conservation laws, namely the conservation of mass, momentum and energy. In addition, we will make use of the material specific constitutive relations for the derivation. A common simplification is to assume that the fluid flow is incompressible which means that the fluid density is spatially constant. Likewise, we assume that the viscosity coefficient, and the thermal conductivity are uniform.

This assumption has important implications:

- The density is no longer an unknown of the equation system. Instead, the pressure becomes an unknown.

- The pressure and the velocity field are decoupled from the energy conservation. This means that we can solve for the pressure and the velocity ignoring the temperature field. We thus assume an isothermal fluid.

These first assumptions reduce the Navier-Stokes equations to four equations in four unknowns: three velocity components and the pressure.
Balance Equations

The balance equations describe fundamental physical laws and are valid for any system described with continuum mechanics. Hence, they are the same for the Navier-Stokes equations and for the SWE. Since we assume that the fluid is incompressible and isothermal, the conservation of mass and the conservation of momentum are the only balance equations required for the derivation. The continuity equation or conservation of mass for a domain $\Omega$ with the boundary $\partial \Omega$ with density $\rho$ reads like

$$\frac{d}{dt} \iiint_{\Omega} \rho \, dv + \iint_{\partial \Omega} \rho \mathbf{v} \cdot \mathbf{n} \, dA = 0.$$  \hspace{1cm} (A.1)

It means the temporal change of the fluid mass in the domain, which is computed by the integral of the density over the fluid volume (the first term), is given by the flux along its boundary (the second term). Similarly, the conservation of momentum looks like

$$\frac{d}{dt} \iiint_{\Omega} \rho \mathbf{v} \, dv + \iint_{\partial \Omega} \rho \mathbf{v} \mathbf{v}^T \cdot \mathbf{n} \, dA + \iint_{\partial \Omega} \sigma \cdot \mathbf{n} \, dA = \iiint_{\Omega} \mathbf{f} \, dv$$ \hspace{1cm} (A.2)

where $\mathbf{v}$ is the fluid velocity and $\mathbf{f}$ captures sources and sinks of the fluid. This last term is also known as body force and is a force density, in $[N/m^3]$, and the actual force acting on a given volume $V$ is thus given by the volume integral over this force density. $\sigma$ denotes the stress tensor which is the sum of the pressure $p$ and the viscous stress tensor $\mathbf{T}$. Note that viscous forces are caused by spatial velocity differences in the fluid. We will assume that there are only small velocity variations, so that the viscosity coefficient is small. In addition we will ignore the viscous stress tensor. This assumption justified by observing that water-like fluids have a very small viscosity and that the numerical integration introduces a certain amount of damping, which basically has the same effect as a viscosity larger than zero. Typically, this numerical viscosity is larger than that of water so that we rely on the solver to compute a fluid motion with a viscosity as small as possible. Essentially, we solve for an inviscid fluid.

Projection

The derivation of the SWE now proceeds by making an important assumption: the pressure is assumed to be hydrostatic, i.e.,

$$p = \rho a_n \Delta \eta$$ \hspace{1cm} (A.3)

where $a_n$ is the acceleration in vertical direction and $\Delta \eta$ corresponds to the vertical distance from the bottom of the fluid to its surface. Amongst others, the vertical acceleration takes care of the gravitational acceleration $g = -9.81 \frac{m}{s^2}$. This hydrostatic pressure equation can be derived by assuming that the vertical velocity is much smaller than the horizontal velocity such that the momentum conservation in vertical direction is dominated by the pressure term and hence the advection of the vertical velocity $v_n$ can safely be ignored. This assumption is justified by observing that in areas where the fluid is shallow, the vertical velocity is indeed orders of magnitude smaller than the horizontal velocity components. The implications of the hydrostatic pressure assumption are:
The momentum equation for the vertical component is static, i.e., the vertical velocity is temporally constant and equal to zero.

The velocity is basically a two dimensional tangential vector field which is orthogonal to the vertical (or normal) direction.

The pressure $p$ acts as the sole vertical variable.

The fluid surface can be represented by a heightfield.

The assumption of a hydrostatic pressure thus allows us to further reduce the number of unknowns from four to three - the fluid can now be described by velocities in the horizontal domain, and a height of the fluid.

The tangential velocity can be thought of as averaged over the fluid depth. The momentum conservation equation therefore reads

$$\iiint_{\Omega^3} \left( f_{\tau} \right) dV = \frac{d}{dt} \iiint_{\Omega^3} \rho \begin{pmatrix} v_{\tau} \\ 0 \end{pmatrix} dV + \iiint_{\partial\Omega^3} \rho \begin{pmatrix} v_{\tau} \\ 0 \end{pmatrix}^T n dA + \iiint_{\partial\Omega^3} p I \cdot n dA \tag{A.4}$$

where the subscript $\tau$ indicates horizontal (or tangential) components and the subscript $n$ denotes the vertical (or normal) component. The following notation is introduced and will be used throughout all the chapters of this chapter:

- $H$: Height of fluid between zero-level and ground.
- $h$: Height of fluid above zero-level.
- $\eta$: Total fluid height, i.e. $\eta = H + h$.

To achieve a heightfield representation of the fluid we have to get rid of the vertical dimension in these equations by a projection of the integrals to the two dimensional horizontal domain. Assume the equations are computed over a fluid column with a square base area $\Omega^2_\tau$. Consider first the surface integral in the continuity equation

$$\iint_{\partial\Omega^3} \rho v \cdot ndA = \iint_{\Omega^2_\tau} \rho v \cdot ndA + \iint_{\Omega^2_n} \rho v \cdot ndA$$

where the surface integral was split up into two parts over the surfaces $\Omega^2_n$ vertical to the horizontal projection plane $\Omega^2_\tau$ and the remaining bottom and top surface. There is no mass flux through the bottom because the bottom surface is impermeable and thus this surface integral vanishes. The slope of the fluid surface is assumed to be small. This implies that $v \cdot n$ is small, too and hence, the integral over the top of the fluid column is negligible. The mass flux through the vertical sides $\Omega^2_n$ is reduced to a line integral by projecting the fluid depth onto the boundaries $\partial\Omega^2_n$ of the horizontal projection surface

$$\iint_{\Omega^2_n} \rho v \cdot ndA = \int_{\partial\Omega^2_\tau} \int_{h-H}^{h} \rho v \cdot nd \eta ds = \int_{\partial\Omega^2_\tau} \rho (h + H) v \cdot nds = \int_{\partial\Omega^2_\tau} \rho \eta v \cdot nd. \tag{A.5}$$
Note that the normals of the vertical surfaces $\Omega \times$ of the fluid column are equal to the normals of the boundary curve of the horizontal projection surface. The projection of the momentum advection term is analogous to the projection of the mass flux, i.e., the integrals over the non-vertical areas vanish due to the same reasons.

Consider now the pressure force acting on the fluid column

$$\int_{\Omega^2_1} p \mathbf{l} \cdot \mathbf{n} dA = \int_{\Omega^2_1} \rho \eta \mathbf{n} dA$$

where the surface integral was again split into a horizontal and a vertical component. The surface integral over the top of the fluid surface vanishes because the pressure at this interface is negligible compared to the pressure inside the fluid. The surface integral over the bottom leads to a source term of the momentum, provided the bottom is non-flat. This pressure force makes the fluid flow downwards. This force is computed by integrating the hydrostatic pressure $p = \rho g \Delta \eta$ over the bottom area

$$\int_{\Omega^2_1} p \mathbf{l} \cdot \mathbf{n} dA = \int_{\Omega^2_1} \rho a_n(-\eta) \mathbf{n} dA$$

$$= \int_{\Omega^2_1} \rho a_n(-H-h)(-\nabla H) dA = \int_{\Omega^2_1} \rho a_n(H+h)\nabla H dA. \tag{A.6}$$

Note that the normal $\mathbf{n}$ in this equation corresponds to the normal given by the negative gradient of the bottom heightfield $-\nabla H$ rather than the normal of the horizontal projection plane. Furthermore, the normalization of the gradient and the cosine for the area foreshortening cancel each other out.

The projection of the pressure force due to the vertical sides of the fluid column transforms the surface integral over these sides again into a boundary integral over the curve $\partial \Omega^2_2$, which bounds the horizontal area. Thus, by integrating over the fluid depth, the fluxes through the vertical sides get concentrated at the boundary curve $\partial \Omega^2_2$

$$\int_{\Omega^2_1} p \mathbf{l} \cdot \mathbf{n} dA = \int_{\Omega^2_1} \int_{-H}^h \rho a_n(\eta-h) \mathbf{n} ds$$

$$= -\int_{\partial \Omega^2_2} \frac{1}{2} \rho a_n(-H-h)^2 \mathbf{n} ds = -\int_{\partial \Omega^2_2} \frac{1}{2} \rho a_n \eta^2 \mathbf{n} ds. \tag{A.7}$$

Integrating the vertical component in the volume integrals of Eq. (A.1) and Eq. (A.4) and inserting Eq. (A.5), Eq. (A.6), and Eq. (A.7) the SWE can be restated in the integral form

$$\frac{d}{dt} \int_{\Omega^2} \rho \eta v dA + \int_{\partial \Omega^2} \rho \eta \mathbf{v} \cdot \mathbf{n} dA = 0 \tag{A.8}$$

$$\frac{d}{dt} \int_{\Omega^2} \rho \eta v dA + \int_{\partial \Omega^2} \rho \eta \mathbf{v} \mathbf{v}^T \cdot \mathbf{n} dA - \int_{\partial \Omega^2} \frac{1}{2} \rho a_n \eta^2 \mathbf{n} ds$$

$$= -\int_{\partial \Omega^2} \rho a_n \eta \nabla H dA + \int_{\Omega^2} \eta \mathbf{f}_c dA. \tag{A.9}$$

From now on, the subscript $\tau$ will be omitted in the equations. In the literature, these equations are often given in the so called flux form

$$\frac{d}{dt} \int_{\Omega^2} \mathbf{q} dA + \int_{\partial \Omega^2} \mathbf{F}(\mathbf{q}) \cdot \mathbf{n} dA = \int_{\Omega^2} \psi dA \tag{A.10}$$
where the equations were divided by the constant density and

\[ q = \begin{pmatrix} \eta & \eta u & \eta v \end{pmatrix}^T \]

\[ F(q) = \begin{bmatrix} f(q) \\ g(q) \end{bmatrix} = \begin{bmatrix} \eta u^2 - \frac{1}{2}a_n \eta^2 & \eta \eta v \\ \eta \eta u & \eta^2 - \frac{1}{2}a_n \eta^2 \end{bmatrix} \]

\[ \psi = \begin{pmatrix} 0 \\ -a_n \eta \nabla_H + \frac{\eta}{\rho} f \end{pmatrix}. \]

The two components of the velocity are denoted here as \( \mathbf{v} = (u, v)^T \). This notation clearly shows the non-linearity and the conservative nature of the equations due to the flux across the boundary.

### A.2 Differential Form

We now turn to the derivation of the differential form of the SWE, although the integral form admits more general solutions than the differential form. Solutions to the integral form may exhibit discontinuities (which must still satisfy certain conditions) like for example shocks. Such solutions are also known as weak solutions. The differential form on the other hand is written as a partial differential equation (PDE) which only admits continuous solutions because the derivation requires a continuity assumption. Solutions of the differential form are known as strong (also called pointwise) solutions. But note that the true solution of the SWE with certain boundary conditions might indeed be discontinuous and in such cases, a solution method derived from the differential form fails to converge to the correct solution. However, in computer graphics, accuracy is not the most important factor and thus, a strong solution which is similar to the true weak solution might be adequate enough.

As mentioned before, we will assume in this derivation that the solution to the SWE is continuous and differentiable. The divergence theorem can then be applied to the boundary integral of the integral form A.8 and A.9 and the time derivative can be moved inside the integral. We end up with

\[ \int_{\Omega^2} \mathbf{q}_t \, dA + \int_{\Omega^2} \nabla \cdot F(q) \, dA = \int_{\Omega^2} \psi \, dA. \]

This equation must be valid for any two dimensional domain \( \Omega^2 \) and therefore the following PDE must hold

\[ \mathbf{q}_t + \nabla \cdot F(q) = \mathbf{q}_t + \frac{\partial f(q)}{\partial q} \mathbf{q}_x + \frac{\partial g(q)}{\partial q} \mathbf{q}_y = \psi. \]  

(A.11)

The chain-rule was used to expand the divergence into a product involving the Jacobian of the flux functions \( f \) and \( g \). Note that such a PDE is called hyperbolic if any linear combination of the two Jacobians \( \frac{\partial f}{\partial q} \) and \( \frac{\partial g}{\partial q} \) has real eigenvalues and can be diagonalized. This is indeed the case for the SWE.

Written out in detail, Eq. (A.11) looks like

\[ \eta_t = -\nabla \cdot (\eta \mathbf{v}) \]  

(A.12)

\[ \frac{\partial \eta \mathbf{v}}{\partial t} + \nabla \cdot (\eta \mathbf{v} \mathbf{v}^T - \frac{1}{2}a_n \eta^2) = -a_n \eta \nabla H + \frac{\eta}{\rho} f_T. \]  

(A.13)
This latter equation can be simplified even further. By pushing the differential operator inside the bracketed terms we get

$$
\eta v_t + v \eta_t + v \nabla \cdot (\eta v) + \eta v \cdot \nabla u - a_n \eta \nabla \eta - \frac{1}{2} \eta^2 \nabla a_n = -a_n \eta \nabla H + \frac{\eta}{\rho} f_\tau.
$$

The second and third term on the left hand side correspond to the continuity equation times the velocity and thus is equal to zero. Dividing by $\eta$ and observing that $H - \eta = -h$ finally gives

$$
v_t + v \cdot \nabla v - a_n \nabla h - \frac{1}{2} \eta \nabla a_n = \frac{1}{\rho} f_\tau.
$$

(A.14)

The previous equation clearly highlights the advectional part of the momentum conservation. The differential equation derived from the continuity equation can also be formulated using a material derivative by observing that $\nabla \cdot (\eta v) = v \cdot \nabla \eta + \eta \nabla \cdot v$. The SWE then take the following form

$$
\frac{D\eta}{Dt} = -\eta \nabla \cdot v
$$

(A.15)

$$
\frac{Dv}{Dt} = a_n \nabla h + \frac{1}{2} \eta \nabla a_n + \frac{1}{\rho} f_\tau.
$$

(A.16)

Note that when the balance equations were projected from three dimension to two dimensions, the vertical force component has been dropped. This term can now be reinserted as a vertical acceleration component, for example in addition to a constant acceleration due to gravity $g = -9.81 \text{m/s}^2$:

$$
a_n = g + \frac{f_n}{\rho}
$$

Thanks to these external force terms $f_\tau$ and $f_n$, the SWE can more easily interact with other objects. Note that the SWE are usually derived without these terms.
Bibliography


