

A Fluid Stability

In this section, we provide a detailed discussion of the stability of our fluid simulation model in both the full and reduced spaces.

A.1 Full Space

Our discretization of the fluid equations in §5 is *stable*, meaning that the discrete versions of the partial differential equation do not inherently gain energy. To see why, let us consider the advection, diffusion, and projection steps separately. Energy is given by $E(\mathbf{u}) = \frac{1}{2} \|\mathbf{u}\|_{\mathbf{V}(\mathbf{g})}^2$ and its time derivative is $\dot{E} = \mathbf{u}^T \mathbf{V}(\mathbf{g}) \dot{\mathbf{u}}$. Noting that we use oriented fluxes in (6), it is easy to see that the advection matrix $\mathbf{A} \otimes_2 \mathbf{f}$ is antisymmetric. Substituting for $\dot{\mathbf{u}}$ according to (8), we can see that the advection step exactly conserves energy.

$$\dot{E} = \mathbf{u}^T (\mathbf{A} \otimes_2 \mathbf{f}) \mathbf{u} = 0 \quad (\text{A.1})$$

The graph Laplacian Δ has only positive eigenvalues, so substituting (9) for $\dot{\mathbf{u}}$ gives us

$$\dot{E} = \mathbf{u}^T \left(-\mu \mathbf{V}^{-1}(\mathbf{g}) \Delta \right) \mathbf{u} < 0. \quad (\text{A.2})$$

Finally, in the projection step, we minimize the energy difference between the current velocities and the velocities corresponding to new, divergence free fluxes. Let \mathbf{u} be the velocities before projection, $\mathbf{u}' = \mathbf{V}^{-1}(\mathbf{g})(\mathbf{P} \otimes_2 \mathbf{g})\mathbf{f}$ be the divergence-free velocities after projection, and \mathbf{u}_\perp be their difference: $\mathbf{u} = \mathbf{u}' + \mathbf{u}_\perp$. (This is known as the Helmholtz-Hodge decomposition [Stam 1999].) We use $\|\cdot\|_{\mathbf{V}(\mathbf{g})}$ in our objective function (10), meaning \mathbf{u}' and \mathbf{u}_\perp are orthogonal in *energy space*. So the triangle inequality is tight: $E(\mathbf{u}) = E(\mathbf{u}') + E(\mathbf{u}_\perp)$, which implies $E(\mathbf{u}) \geq E(\mathbf{u}')$, ensuring that the projection never gains energy.

Note that this does not mean that the method is unconditionally stable independent of the time integration method and time step chosen. However, we now use the above arguments to show that for our choice of integrator, the reduced simulation is in fact unconditionally stable.

A.2 Reduced Space

The definition of energy in the full space $E(\mathbf{u}) = \|\mathbf{u}\|_{\mathbf{V}(\mathbf{g})}^2$ leads naturally to a definition of reduced energy $\hat{E}(\hat{\mathbf{u}}) = \|\hat{\mathbf{u}}\|_{\hat{\mathbf{V}}(\hat{\mathbf{g}})}^2$, and all our stability arguments from §A.1 carry over directly with one exception: advection. In order to ensure energy-preserving advection, we must be careful in basis selection. Constructing the reduced equivalent of Eq. A.1, we see that the derivative in energy due to reduced-space advection is given by

$$\dot{\hat{E}} = \hat{\mathbf{u}}^T \mathbf{B}_p^T \left((\mathbf{A} \otimes_2 \mathbf{B}_f) \otimes_2 \hat{\mathbf{f}} \right) \mathbf{B}_u \hat{\mathbf{u}}. \quad (\text{A.3})$$

If we set $\mathbf{B}_u = \mathbf{B}_p$, then $\hat{\mathbf{u}}^T \mathbf{B}_p^T = (\mathbf{B}_u \hat{\mathbf{u}})^T$, and since the full space matrix $(\mathbf{A} \otimes_2 \mathbf{B}_f) \otimes_2 \hat{\mathbf{f}}$ itself is antisymmetric, we once again are in possession of an energy-conserving discretization. Combined with analytic integration using matrix exponentiation, this basis choice results in an unconditionally stable system. To achieve $\mathbf{B}_u = \mathbf{B}_p$, we first compute the momentum and velocity bases independently. We then concatenate them and re-orthogonalize the result. We use this combined basis for both \mathbf{B}_u and \mathbf{B}_p in our fluid simulations, which guarantees that the simulations will preserve energy.

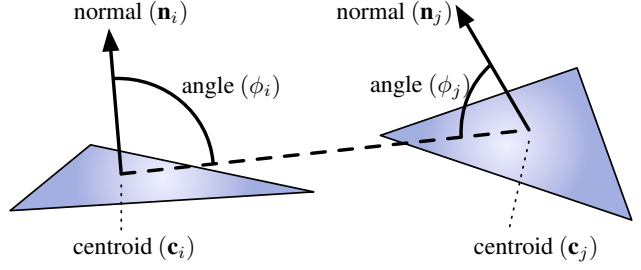


Figure B.1: Geometric layout of the illumination variables on two triangles.

B Radiosity Derivation

They key step in the model reduction of radiosity using our technique is the reduction of the form factor equation (Eq. 15). To model reduce this equation, we follow the process described in §4: we represent it as a collection of tensors, composed using tensor products, matrix inversion, and matrix roots. In this section we provide a detailed derivation of the tensors and sequence of operations shown in Table 2. We arrive at this sequence of operations by decomposing the radiosity equation (Eq. 14), so we will be describing the tensors and operations in Table 2 from bottom to top.

To evaluate the radiosity equation, we need to compute the matrix $\mathbf{I} - \rho \mathbf{F}$. As described in §8.1, we begin by unrolling $\mathbf{F} \in \mathbb{R}^{n \times n}$ into a vector $\mathbf{p} \in \mathbb{R}^{n^2}$ by re-indexing: $p_k = \mathbf{F}_{i,j}$, where $k = ni + j$ and n is the number of faces in the mesh. We also need the transpose index $k^T = nj + i$, so that $p_{k^T} = \mathbf{F}_{j,i}$. The tensor form of the radiosity equation (Eq. 14) is then:

$$\mathbf{b} = (\mathbf{I} - \mathbf{E} \otimes_2 \mathbf{p})^{-1} \mathbf{e}, \quad (\text{B.1})$$

where \mathbf{E} is the 3rd-order tensor that transforms the unrolled form factor vector \mathbf{p} back into the form factor matrix \mathbf{F} and left-multiplies \mathbf{F} by the face albedos ρ .

Our task is now to compute \mathbf{p} , the vector of form factors. At this stage, we choose to separate the factor of area in the denominator of Eq. 15, making this stage a division of *area-free* form factors, denoted by \mathbf{d} , by the square roots of squared areas, where we denote squared areas by \mathbf{a} :

$$p_k = d_k / \sqrt{a_i}. \quad (\text{B.2})$$

We implement this division using a tensor \mathbf{P} :

$$\mathbf{p} = (\mathbf{P} \otimes_2 \mathbf{a})^{-\frac{1}{2}} \mathbf{d}. \quad (\text{B.3})$$

\mathbf{a} is a simple function of the normals \mathbf{n} : $a_k = \mathbf{n}_i \cdot \mathbf{n}_i$. (Recall that \mathbf{n}_i is the normal of the i th face, making it a 3-vector.) We define a tensor \mathbf{N} to implement these dot products:

$$\mathbf{a} = \mathbf{N} \otimes_{1\dots 2} \mathbf{n}. \quad (\text{B.4})$$

Returning to \mathbf{d} , the next factor we separate is visibility, which gives us the expression

$$d_k = c_k \text{Vis}(i, j), \quad (\text{B.5})$$

where \mathbf{c} are *visibility-free* form factors. We define the visibility vector \mathbf{v} such that $v_k = \text{Vis}(i, j)$. We treat $\mathbf{v}(\mathbf{g})$ as a function of geometry directly, and reduce the computation of $\text{Vis}(i, j)$ using a non-Galerkin method which we describe in more detail in §8.3.

Now we can write \mathbf{c} as an element-wise product of two vectors:

$$c_k = \frac{1}{\pi} h_k h_{kT}, \quad (\text{B.6})$$

where

$$h_k = \frac{\mathbf{n}_i \cdot (\mathbf{c}_j - \mathbf{c}_i)}{\|\mathbf{c}_i - \mathbf{c}_j\|^2}. \quad (\text{B.7})$$

We call \mathbf{h} *half form-factors*, and define a tensor \mathbf{C} implementing Eq. B.6:

$$\mathbf{c} = \mathbf{C} \otimes_{1\dots 2} \mathbf{h}. \quad (\text{B.8})$$

Next, we rewrite both parts of the quotient in Eq. B.7:

$$h_k = \frac{s_k}{r_k}, \quad (\text{B.9})$$

where

$$r_k = \|\mathbf{c}_i - \mathbf{c}_j\|^2 \quad (\text{B.10})$$

and

$$s_k = \mathbf{n}_i \cdot (\mathbf{c}_j - \mathbf{c}_i). \quad (\text{B.11})$$

\mathbf{r} is a vector of squared distances between face centroids. \mathbf{s} is a vector of *scaled cosines*: s_k is the cosine of the angle between \mathbf{n}_i and $\mathbf{c}_j - \mathbf{c}_i$, multiplied by face area and the distance between face centroids. We use a tensor \mathbf{H} to construct a diagonal matrix, with \mathbf{r} as its entries, which we invert to find \mathbf{h} :

$$\mathbf{h} = (\mathbf{H} \otimes_2 \mathbf{r})^{-1} \mathbf{s}. \quad (\text{B.12})$$

While \mathbf{s} is polynomial in \mathbf{g} , and so it would be possible to compute it directly as a from \mathbf{g} by contracting a single tensor, we can reduce the polynomial degree of this system from 3 to 2 (and the maximum tensor order from 4 to 3) by decomposing \mathbf{s} one step further. Note that Eq. B.11 is bilinear in \mathbf{n} and \mathbf{c} , the latter of which is linear in \mathbf{g} . Therefore, we can define a tensor \mathbf{S} such that

$$\mathbf{s} = \mathbf{S} \otimes_1 \mathbf{n} \otimes_2 \mathbf{g}. \quad (\text{B.13})$$

This leaves us with only \mathbf{n} to compute. The normals are given by

$$\mathbf{n}_i = (\mathbf{g}_{i,2} - \mathbf{g}_{i,0}) \times (\mathbf{g}_{i,1} - \mathbf{g}_{i,0}), \quad (\text{B.14})$$

where $\mathbf{g}_{i,\ell}$ is the ℓ th vertex of face i and \times is the vector cross product. This expression is a low-degree (quadratic, in fact) polynomial in geometry, and so we can complete the decomposition with a final tensor \mathbf{N} such that

$$\mathbf{n} = \mathbf{N} \otimes_{1\dots 2} \mathbf{g} \quad (\text{B.15})$$

With the tensors defined above, we can now compute radiosity as described in Table 2 given only the scene geometry \mathbf{g} and the incident illumination \mathbf{e} .