Velocity-Based Monte Carlo Fluids - Supplemental Note

Ryusuke Sugimoto
University of Waterloo
Waterloo, Ontario, Canada
rsugimot@uwaterloo.ca

Christopher Batty
University of Waterloo
Waterloo, Ontario, Canada
christopher.batty@uwaterloo.ca

Toshiya Hachisuka
University of Waterloo
Waterloo, Ontario, Canada
toshiya.hachisuka@uwaterloo.ca

A WALK-ON-BOUNDARY FOR PROJECTION

For Eq. 16 and Eq. 17, we defined the normal directions as pointing outward from the fluid domain to describe both interior and exterior problems with a single pair of equations, whereas the previous work [Sabelfeld and Simonov 1994; Sugimoto et al. 2023] had two separate pairs of equations for interior and exterior problems. Hence, our equations for exterior problems differ from Sabelfeld and Simonov [1994] and Sugimoto et al. [2023] in their signs.

Monte Carlo Estimation. Based on Eq. 16 and Eq. 17, we use a walk-on-boundary Monte Carlo estimator [Sabelfeld and Simonov 1994; Sugimoto et al. 2023]. Note that Eq. 17 contains the unknown density function μ itself on the left-hand side of the equation and in one of the integrands on the right-hand side, making it a recursive integral equation similar to the rendering equation [Kajiya 1986] used in path tracing for light transport simulation rather than a simple non-recursive one as in Section 2.3.1. Following Sugimoto et al. [2023], we truncate the recursion after M steps and multiply the contribution of the longest path by 0.5 to design a biased estimator.

We can consider various sampling strategies to estimate the solution to the truncated recursive integral equation with Monte Carlo methods. Following Sugimoto et al. [2023], we use a forward estimator to solve this in our implementation: to sample a path consisting of multiple boundary points, we randomly sample a boundary point first, take a random walk on randomly sampled boundary points, and finally connect them to each individual evaluation point. In the below, for $P_U(\mathbf{x}_0)$, we use uniform sampling to sample a point $\mathbf{x}_0$ on the solid boundary $\partial\Omega$. Then, for $P_R(\mathbf{x}_{m+1}|\mathbf{x}_m)$, we uniformly randomly sample a direction from a unit hemisphere for a line that goes through the point $\mathbf{x}_m$ and uniformly randomly sample one of the intersection points between the line and the solid boundary $\partial\Omega$ to sample the next point $\mathbf{x}_{m+1}$. Recursively applying this sampling strategy lets us sample a series of points $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{M+1}$ to form a path, given previous points in the path. We use this strategy because the PDF associated with this specific forward sampling strategy is known to be proportional to the integral kernel $\frac{\partial G}{\partial \mathbf{n}_\mathbf{x}}$ we have in Eq. 17.

Compared to the simplest formulation for the Laplace equation [Sugimoto et al. 2023], we have some additional non-recursive terms in Eq. 16 and Eq. 17 as a result of the transformations we discussed in the main paper, and we need to consider how to sample such terms, too. We choose to sample the non-recursive contributions on $\partial\Omega$ using the forward estimator as well for the purpose of importance sampling while estimating the other terms similarly to Section 2.3.1. To do so, we first define two estimators for length m contributions, $\langle\mu_m^1(\mathbf{x})\rangle$ and $\langle\mu_m^2(\mathbf{x})\rangle$, recursively, based on Eq. 17: we define the base case for length 1 contributions as

$$\langle\mu_1^1(\mathbf{x}_1)\rangle = \frac{2\frac{\partial G}{\partial \mathbf{n}_\mathbf{x}}(\mathbf{x}_1, \mathbf{x}_0)}{P_R(\mathbf{x}_1|\mathbf{x}_0)}\mathbf{n}(\mathbf{x}_0)\cdot\{\mathbf{u}_3(\mathbf{x}_0) - \mathbf{u}_3(\mathbf{x}_1)\}, \tag{20}$$

$$\langle\mu_1^2(\mathbf{x}_0)\rangle = 2\mathbf{n}(\mathbf{x}_0)\cdot\{-\langle E_V(\mathbf{x}_0)\rangle - \langle E_A(\mathbf{x}_0)\rangle + \mathbf{u}_3(\mathbf{x}_0) - \mathbf{u}_s(\mathbf{x}_0)\}, \tag{21}$$

where $\langle\mu_1^1(\mathbf{x}_1)\rangle$ corresponds to the non-recursive contribution from the first integral in Eq. 17 and $\langle\mu_2^1(\mathbf{x}_0)\rangle$corresponds to the rest of the non-recursive contributions in Eq. 17. Then, we define the contributions from longer paths as

$$\langle\mu_{m+1}^1(\mathbf{x}_{m+1})\rangle = \frac{-2\frac{\partial G}{\partial \mathbf{n}_\mathbf{x}}(\mathbf{x}_{m+1}, \mathbf{x}_m)}{P_R(\mathbf{x}_{m+1}|\mathbf{x}_m)}\langle\mu_m^1(\mathbf{x}_m)\rangle, \tag{22}$$

$$\langle\mu_{m+1}^2(\mathbf{x}_m)\rangle = \frac{-2\frac{\partial G}{\partial \mathbf{n}_\mathbf{x}}(\mathbf{x}_m, \mathbf{x}_{m-1})}{P_R(\mathbf{x}_m|\mathbf{x}_{m-1})}\langle\mu_m^2(\mathbf{x}_{m-1})\rangle. \tag{23}$$

Note in particular, when we use the above-mentioned uniform line intersection sampling for $P_R(\mathbf{y}|\mathbf{x})$,

$$\frac{2\frac{\partial G}{\partial \mathbf{n}_\mathbf{x}}(\mathbf{x}, \mathbf{y})}{P_R(\mathbf{x}|\mathbf{y})} = \kappa(\mathbf{y}, \mathbf{x})\cdot\mathrm{sgn}(\mathbf{n}(\mathbf{x})\cdot(\mathbf{x}-\mathbf{y})), \tag{24}$$

where $\kappa(\mathbf{y}, \mathbf{x})$ is the number of intersection points that the line that goes through the two points has, excluding point $\mathbf{y}$, and sgn is the sign function. Then, based on Eq. 16, we can estimate the pressure gradient by taking N_P sample paths in addition to the terms in Eq. 12:

$$\begin{aligned}
&\langle\nabla_\mathbf{x}\overline{p}(\mathbf{x})\rangle \\
&= \langle E_V(\mathbf{x})\rangle + \langle E_A(\mathbf{x})\rangle \\
&\quad + \frac{1}{N_P}\sum_{k=1}^{N_P}\left[-\frac{\nabla_\mathbf{x}G(\mathbf{x}, \mathbf{x}_0^k)}{2P_U(\mathbf{x}_0^k)}\mathbf{n}(\mathbf{x}_0^k)\cdot\left\{\mathbf{u}_3(\mathbf{x}_0^k) - \mathbf{u}_3(\mathbf{x})\right\}\right. \\
&\quad + \frac{\nabla_\mathbf{x}G(\mathbf{x}, \mathbf{x}_{M+1}^k)}{2P_U(\mathbf{x}_0^k)}\langle\mu_M^1(\mathbf{x}_{M+1}^k)\rangle + \frac{\nabla_\mathbf{x}G(\mathbf{x}, \mathbf{x}_M^k)}{2P_U(\mathbf{x}_0^k)}\langle\mu_M^2(\mathbf{x}_M^k)\rangle \\
&\quad \left. + \sum_{m=1}^{M-1}\frac{\nabla_\mathbf{x}G(\mathbf{x}, \mathbf{x}_{m+1}^k)}{P_U(\mathbf{x}_0^k)}\langle\mu_m^1(\mathbf{x}_{m+1}^k\rangle + \frac{\nabla_\mathbf{x}G(\mathbf{x}, \mathbf{x}_m^k)}{P_U(\mathbf{x}_0^k)}\langle\mu_m^2(\mathbf{x}_m^k)\rangle\right].
\end{aligned} \tag{25}$$

On the right-hand side, the first line estimates the last two integrals in Eq. 16, the second line estimates the non-recursive terms in the first integral, the third line estimates the longest path contributions, and the last line estimates the shorter path contributions. Note that while we do not explicitly indicate so, the contributions from paths of any length $\langle\mu_m^1(\mathbf{x}_{m+1}^k)\rangle$ and $\langle\mu_m^2(\mathbf{x}_m^k)\rangle$ implicitly depend on the sampled boundary point $\mathbf{x}_0^k$ due to the sampling strategy we employ.

To use the estimator Eq. 25, the most naive approach would be to generate sample paths for each individual evaluation point, which has a high computational cost. Instead, we use a boundary

value caching strategy similar to the virtual point light method in rendering [Keller 1997]. We first compute the contributions of N_P subpaths up to the point before we connect them to the evaluation points and cache them at $M+1$ boundary points per path. We then connect all of them to all evaluation points. This significantly increases the effective number of sample paths per evaluation point without increasing the computational cost too much while introducing a correlation of estimates between evaluation points. Even this correlation can be preferable for our application because it guarantees that the contributions from these paths changes smoothly across evaluation points.

Sugimoto et al. [2023, Figure 9] apply a similar caching technique to their Neumann problem walk-on-boundary solver based on the same single-layer boundary integral formulation as well, but they use a backward estimator, which works only with a less efficient resampled importance sampling strategy. In contrast, our method can utilize a forward estimator with more efficient line intersection importance sampling. This is at the cost of increasing storage per cache point by a small constant multiplication factor and increasing computation per evaluation point when we sum up the contributions from all cache points.

Similar to Section 2.3.1, for the estimation of volume integrals $\langle E_V(\mathbf{x})\rangle$ in Eq. 21 and Eq. 25, we use the importance sampling strategy so PDF P_V is proportional to $1/r^{(d-1)}$, and let $\mathbf{S}(\mathbf{x},\mathbf{y})=\mathbf{0}$ for all $\mathbf{y}$ outside the simulation domain. Additionally, we need to set $\mathbf{S}(\mathbf{x},\mathbf{y})=\mathbf{0}$ for all $\mathbf{y}$ inside of solid obstacles, too. We perform this insidedness test by casting a ray from point $\mathbf{y}$ in a random direction and taking a sum of its intersection signs. This strategy can be considered a Monte Carlo estimator for the generalized winding number [Jacobson et al. 2013]. We use uniform sampling for $\langle E_A(\mathbf{x})\rangle$.

For all 2D examples in the paper except Fig. 7, we use $N_V = N_A = 5\cdot 10^5$ for direct contributions to each evaluation point. For indirect (path) contributions, we use $N_P = 5\cdot 10^5$ paths with path length 4. For each path, the volume and area term sample counts for the pseudo boundary are $N_V = N_A = 10$.

B WALK-ON-BOUNDARY FOR DIFFUSION

To get scalar diffusion equations from Eq. 18, we first absorb the viscosity coefficient ν into the variable to define $\mathbf{w}(\mathbf{x},s)=\nu\overline{\mathbf{u}}(\mathbf{x},s/\nu)$ and redefine the range of time s accordingly:

$$\begin{aligned}
\frac{\partial \mathbf{w}(\mathbf{x},s)}{\partial s} &= \nabla^2\mathbf{w}(\mathbf{x},s) && \text{for } \mathbf{x}\in\Omega, s\in(0,\nu\Delta t)\\
\mathbf{w}(\mathbf{x},s) &= \mathbf{w}^*(\mathbf{x},s)=\overline{\mathbf{u}}(\mathbf{x},s/\nu) && \text{for } \mathbf{x}\in\partial\Omega, s\in(0,\nu\Delta t), \text{and}\\
\mathbf{w}(\mathbf{x},0) &= \mathbf{w}^*(\mathbf{x},0)=\overline{\mathbf{u}}(\mathbf{x},0) && \text{for } \mathbf{x}\in\Omega,
\end{aligned} \tag{26}$$

where the asterisk in $\mathbf{w}^*$ indicates that it is a given function. This is still a vector-valued equation, but because we only need to deal with simple Dirichlet boundary conditions, we can solve the vector diffusion equation Eq. 26 component-wise. Thus, we describe the scalar diffusion equation solver for one scalar component of $\mathbf{w}$, w below.

We will summarize the diffusion walk-on-boundary solver from the book by Sabelfeld and Simonov [1994, Chapter 4] here, focusing on its use in our context. For more general cases not described here, such as Neumann problems and nonhomogeneous problems, readers should refer to the book. The idea of the diffusion walk-on-boundary method is very similar to the one for the Poisson equation: we convert the partial differential equation into an integral equation and solve it using a ray-tracing-style solver. However, with the additional time dependency, we need to additionally consider the time variation in the diffusion equation and take walks in the space-time domain. Note that this solver does not discretize the time domain within each time step, unlike most traditional methods that discretize the time with a finite difference.

First, we define the fundamental solution for the diffusion equation Eq. 26, also known as the heat kernel,

$$Z(\mathbf{x},s;\,\mathbf{y},\tau)=\Theta(t')(4\pi t')^{-d/2}e^{-r^2/4t'}, \tag{27}$$

where $t'=s-\tau$ and $\Theta(\cdot)$ is the Heaviside step function.

Using the fundamental solution, we can write the solution to the diffusion equation in the form of the double layer potential and an additional initial condition term for $\mathbf{x}\in\Omega$ and $s\in[0,\nu\Delta t]$:

$$\begin{aligned}
w(\mathbf{x},s) = &-\int_0^s\int_{\partial\Omega}\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x},s;\,\mathbf{y},\tau)\phi(\mathbf{y},\tau)\,\mathrm{dA}(\mathbf{y})\,\mathrm{d}\tau\\
&+\int_\Omega Z(\mathbf{x},s;\mathbf{y},0)w^*(\mathbf{x},0)\,\mathrm{dV}(\mathbf{y}),
\end{aligned} \tag{28}$$

where $\phi(\cdot,\cdot)$ is an unknown density function defined for $\mathbf{x}\in\partial\Omega$ and $s\in[0,\nu\Delta t]$. We can also derive a recursive boundary integral equation for $\phi(\cdot,\cdot)$ by taking the limit $\mathbf{x}\to\partial\Omega$ in Eq. 28:

$$\begin{aligned}
\phi(\mathbf{x},s) = &\int_0^s\int_{\partial\Omega}2\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x},s;\,\mathbf{y},\tau)\phi(\mathbf{y},\tau)\,\mathrm{dA}(\mathbf{y})\,\mathrm{d}\tau\\
&+2w^*(\mathbf{x},s)-2\int_\Omega Z(\mathbf{x},s;\mathbf{y},0)w^*(\mathbf{x},0)\,\mathrm{dV}(\mathbf{y}).
\end{aligned} \tag{29}$$

Monte Carlo Estimation. We design a Monte Carlo estimator based on Eq. 28 and Eq. 29. First, we define an estimator for the initial condition term with N_I samples using PDF P_I as

$$\langle E_I(\mathbf{x},s)\rangle = \frac{1}{N_I}\sum_{i=1}^{N_I}\frac{\mathbf{Z}(\mathbf{x},s;\mathbf{y}^i,0)}{P_I(\mathbf{y}^i|\mathbf{x},s)}w^*(\mathbf{y}^i,0). \tag{30}$$

Using this estimator, we can define the estimator for Eq. 28 with N_D path samples using PDF P_D as

$$\langle w(\mathbf{x},s)\rangle = \langle E_I(\mathbf{x},s)\rangle + \frac{1}{N_D}\sum_{j=1}^{N_D}-\frac{\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x},s;\,\mathbf{y}^j,\tau^j)}{P_D(\mathbf{y}^j,\tau^j|\mathbf{x},s)}\langle\phi(\mathbf{y}^j,\tau^j)\rangle \tag{31}$$

and the one for Eq. 29 with 1 sample with PDF P_E as

$$\langle\phi(\mathbf{x},s)\rangle = 2\frac{\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x},s;\,\mathbf{y},\tau)}{P_E(\mathbf{y},\tau|\mathbf{x},s)}\langle\phi(\mathbf{y},\tau)\rangle + 2w^*(\mathbf{x},s) - 2\langle E_I(\mathbf{x},s)\rangle, \tag{32}$$

which is a recursive estimator: $\langle\phi(\mathbf{y},\tau)\rangle$ included on the right-hand side should be estimated recursively. We use a backward estimator and start generating paths for this recursive estimator from each evaluation point $(\mathbf{x},\nu\Delta t)$ toward time 0. Note that P_D and P_E only need to sample points with time $\tau<s$ because the integral kernel $\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}$ is zero for $\tau\geq s$. Intuitively, this is because the solution to the heat equation depends only on previous times. Using this property, we sample points in the space-time domain so that times for the sampled sequence of points strictly decrease. We terminate the recursion when the sampled time τ is negative because the original

integral domain does not contain the negative part. While the walk-on-boundary method for the Poisson equation is biased, the walk-on-boundary method for the diffusion equation gives an unbiased solution estimate.

As for the specific sampling strategy, for the initial condition sampling $P_I(\mathbf{y}^i|\mathbf{x}, s)$, we sample the points by $\mathbf{y}_i \leftarrow \mathbf{x} + \sqrt{t\gamma_{d/2}}\omega$, where $\gamma_{d/2}$ is a sample drawn from the Gamma distribution with shape parameter $d/2$ and scale parameter 1, and ω is a uniformly random direction sampled on a unit sphere. With this sampling strategy, we get

$$\frac{\mathbf{Z}(\mathbf{x}, s; \mathbf{y}^i, 0)}{P_I(\mathbf{y}^i|\mathbf{x}, s)} = 1. \tag{33}$$

Similar to Section 2.3.2, we set $\mathbf{Z}(\mathbf{x}, s; \mathbf{y}^i, 0) = 0$ for all $\mathbf{y}$ inside of solid obstacles in Eq. 30.

For $P_D(\mathbf{y}, \tau|\mathbf{x}, s)$ and $P_E(\mathbf{y}, \tau|\mathbf{x}, s)$, we sample $\mathbf{y}$ using the uniform line intersection sampling as in Section 2.3.2 and sample time τ, given t, by $\tau \leftarrow s - \frac{\|\mathbf{y}-\mathbf{x}\|}{4\gamma_{d/2}}$ to get

$$\begin{aligned} \frac{\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x}, s;\ \mathbf{y}, \tau)}{P_D(\mathbf{y}, \tau|\mathbf{x}, s)} &= -\kappa(\mathbf{x}, \mathbf{y}) \cdot \mathrm{sgn}(\mathbf{n}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x})), \quad \text{and} \\ 2\frac{\frac{\partial Z}{\partial \mathbf{n}_\mathbf{y}}(\mathbf{x}, s;\ \mathbf{y}, \tau)}{P_E(\mathbf{y}, \tau|\mathbf{x}, s)} &= -\kappa(\mathbf{x}, \mathbf{y}) \cdot \mathrm{sgn}(\mathbf{n}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x})). \end{aligned} \tag{34}$$

In the above, the left-hand side scaling factors of the two expressions differ by 2 because the PDF P_D and P_E differ even though we use the same strategy: point $\mathbf{x}$ for P_D lies within the domain, whereas point $\mathbf{x}$ for P_E lies on a boundary.

To draw samples from the Gamma distribution, we use $\gamma_1 \leftarrow -\ln(\alpha)$ for 2D, where α is a uniformly random sample in the range $(0, 1)$, and $\gamma_{3/2} \leftarrow \gamma_1 + \xi^2/2$ for 3D, where ξ is a standard normal sample.

For problems without boundaries, we drop the second term in Eq. 31, and the remaining first term estimates the convolution of the input field and a Gaussian function.

In our implementation, we use $N_I = 5 \cdot 10^5$ initial conditions samples at each evaluation point, with $N_D = 5 \cdot 10^5$ paths with $N_I = 10$ initial condition samples per bounce. Unlike the walk-on-boundary method for the projection step, we do not share the subpaths among evaluation points as we found that the diffusion walk-on-boundary is typically cheaper, and there was not much need to improve its efficiency relative to the projection.

REFERENCES

Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. 2013. Robust Inside-Outside Segmentation Using Generalized Winding Numbers. *ACM Trans. Graph.* 32, 4, Article 33 (jul 2013), 12 pages. https://doi.org/10.1145/2461912.2461916

James T. Kajiya. 1986. The Rendering Equation. *SIGGRAPH Comput. Graph.* 20, 4 (aug 1986), 143–150. https://doi.org/10.1145/15886.15902

Alexander Keller. 1997. Instant Radiosity. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., USA, 49–56. https://doi.org/10.1145/258734.258769

Karl K. Sabelfeld and Nikolai A. Simonov. 1994. *Random Walks on Boundary for Solving PDEs*. De Gruyter, Berlin. https://doi.org/10.1515/9783110942026

Ryusuke Sugimoto, Terry Chen, Yiti Jiang, Christopher Batty, and Toshiya Hachisuka. 2023. A Practical Walk-on-Boundary Method for Boundary Value Problems. *ACM Trans. Graph.* 42, 4, Article 81 (jul 2023), 16 pages. https://doi.org/10.1145/3592109