УДК 519.6:532.5:533.6.011 doi 10.26089/NumMet.v19r101

ВИЗУАЛИЗАЦИЯ КАПЕЛЬ ЖИДКОСТИ В FLOWVISION

А. А. Федоров[1]

Предложен метод визуализации капель жидкости в программном комплексе FlowVision, который использует локально адаптивную параллелепипедную сетку с подсеточным разрешением геометрии. Функция VoF используется для восстановления поверхности жидкости и капель, которые затем отрисовываются средствами OpenGL. Описываются подходы рендеринга капель жидкости, рассматриваются эффекты преломления и затухания света.

Ключевые слова: визуализация жидкости, рендеринг капель, FlowVision, OpenGL.

1. Введение. Аэродинамическим характеристикам аппаратов, проектируемых для применения в воздушной или жидкостной среде, уделяется большое внимание. Их оценивают путем моделирования различных ситуаций в CFD-пакетах (CFD: Computational Fluid Dynamics). Выполненные расчеты позволяют получить количественные характеристики объекта и оценить их соответствие требованиям. Визуализация данных — важная часть оценочного этапа, потому что зрительное восприятие развито у человека больше, чем остальные виды восприятия. Настоящая статья посвящена визуализации капель жидкости в отечественном CFD-пакете FlowVision.

2. Обзор существующих решений. Жидкость, как правило, прозрачна, поэтому для реалистичной визуализации необходима работа с полупрозрачными объектами. Одна из идей реализации полупрозрачности тел с преломляющими стенками рассмотрена в работе [3]. Суть ее следующая: сначала выполняется отрисовка всех непрозрачных объектов сцены в текстуру фона. Далее рендерятся полупрозрачные объекты, при этом цвет из текстуры фона берется в смещенном пикселе, смещение регулируется картой нормалей мэша и коэффициентом масштабирования. Результат работы такого алгоритма выглядит правдоподобно, однако его минус в том, что не учитывается расстояние до объектов фона — не наблюдается эффект параллакса.

Более серьезный подход к реализации эффекта преломления в реальном времени был представлен для капель в работе [4]. Предполагается, что капли имеют форму полусферы и располагаются на экране. Вводится параметр искажения, используемый для преобразования координат пикселя внутри капли степенной функцией (производится нелинейная гомотетия круга, переводящая круг в самого себя). Здесь не наблюдается эффект параллакса, но контроль преломления ближе к физически корректному, чем в подходе [3].

Физически корректная методика симуляции капель жидкости на лобовом стекле автомобиля предложена в работе [5]. При моделировании формы капель помимо гравитации и ветра в расчетах учитывается гистерезис контактного угла и диссипативные силы. Кроме того, реализован эффект преломления фона в капле. Помимо этого, вычисляются и отрисовываются следы движения капель по стеклу.

Трассировка преломленного луча по карте глубин используется в работе [6] для симуляции эффекта преломления. Этот подход основан на бинарном поиске длины пути луча от места преломления до точки пересечения с фоновой поверхностью, восстанавливаемой по двумерной карте глубин. Он не всегда выдает физически корректные точки пересечений, но дает более реалистичное изображение в реальном времени, чем, например, подход [3]. Более того, в этом подходе учитывается множественное полное внутреннее отражение.

3. Визуализация капель жидкости. Поскольку система визуализации FlowVision основана на OpenGL (используется OpenGL 4.0 compatibility profile, см. спецификацию [8]), то разработанный метод тоже базируется на OpenGL, но в нем не используются какие-либо техники, характерные только для OpenGL, поэтому этот метод может быть реализован с помощью других графических API (Application Programming Interface), например с помощью исполняемой библиотеки DirectX.

Предварительно следует учесть, что в FlowVision используется алгоритм обработки полупрозрачных объектов — depth peeling [7]. Суть его заключается в последовательном получении слоев геометрической сцены и дальнейшем их слитии.

[1] ООО “Вычислительная инженерная платформа”, ул. Юннатов, д. 18, 127083, Москва; программист, e-mail: fedorov@phystech.edu

В FlowVision жидкость представляется в виде функции VoF (Volume of Fluid) [9], заданной на параллелепипедной локально адаптивной сетке с подсеточным разрешением геометрии поверхностей. Поверхность жидкости восстанавливается алгоритмом, основанным на базе marching cubes, однако описание этого процесса выходит за рамки статьи. Работа посвящена отрисовке не разрешаемых расчетной сеткой мелких частиц — капель, занимающих долю ячейки. Наличие капли в ячейке определяется значением функции VoF: если оно больше заданной пороговой величины, а в соседних ячейках меньше, — то капля присутствует. Количество капель может достигать десятков и сотен тысяч, что накладывает жесткие требования на скорость их отрисовки. В то же время необходимо обеспечить приемлемое качество выходного изображения.

Рендеринг большого количества геометрических примитивов упирается в конечную пропускную способность видеокарты. В связи с этим производители графических чипов реализовали возможность генерации геометрии непосредственно на видеокарте. В графическом конвейере за это отвечают стадии *тесселяции* [8, 10].

Следует заранее сказать, что ожидается на выходе: для каждой капли генерируется передняя поверхность сферы. Под передней поверхностью сферы имеется в виду полусфера, которую бы увидел наблюдатель в ортографической проекции, если бы сфера была непрозрачной. По сравнению с рендерингом полной сферы у этого подхода есть следующие преимущества:

— меньше количество генерируемых полигонов и, следовательно, нагрузка на фрагментный шейдер;

— легче рассчитать преломление света внутри частицы.

Массив атрибутов, содержащих центры и радиусы капель, отправляется на видеокарту и обрабатывается тривиальным вершинным шейдером. Шейдер контроля тесселяции оценивает уровень детализации частицы исходя из ее угловых размеров по отношению к наблюдателю. Далее шейдер оценки тесселяции выполняет генерацию вершин передней полусферы, на основе которых задается ее аппроксимация треугольниками. Финальный шаг программируемой части конвейера — фрагментный шейдер, рассчитывающий видимый цвет капли.

3.1. Генерация полусфер с помощью тесселяции.

3.1.1. Определение детализации. Необходимо обеспечить автоматическое определение уровня детализации капли в зависимости от ее угловых размеров относительно наблюдателя.

Пусть на входе дано следующее: R — радиус сферы, d — расстояние от наблюдателя до центра сферы, $w \times h$ — разрешение финального изображения (в пикселях), Δp — допустимая погрешность аппроксимации (в пикселях). Требуется оценить α — угол сектора внешней окружности, разделяющей переднюю и заднюю полусферы.

Максимальное отклонение аппроксимации внешней окружности от аналитической формы выражается так (рис. 1): $\Delta l = R \cdot \left(1 - \cos\dfrac{\alpha}{2}\right)$.

Пусть задана матрица перспективной проекции: $P = \begin{pmatrix} s_x & 0 & 0 & d_x \\ 0 & s_y & 0 & d_y \\ 0 & 0 & s_z & d_z \\ 0 & 0 & -1 & 0 \end{pmatrix}$.

Тогда можно оценить сверху отклонение в пространстве NDC (Normalized Device Coordinates — пространство, в котором трехмерные координаты лежат в диапазоне $[-1; 1]$). Эти координаты в дальнейшем переводятся в пространство экрана: $\Delta l_{\text{ndc}} = \Delta l \cdot \dfrac{\max(s_x, s_y)}{d}$.

Количество различаемых пикселей определяется неравенством

$$\frac{\Delta l_{\text{ndc}}}{2} \cdot \max(w, h) \leqslant \Delta p.$$

Отсюда выводится оценка сверху на размер сектора. Для случая ортографической проекции нужно положить $d = 1$, тем самым исключив перспективное деление. В остальном все расчеты остаются такими же:

$$\alpha \geqslant \alpha_0 = 2 \arccos\left(1 - \frac{2\Delta p \cdot d}{R \cdot \max(s_x, s_y) \cdot \max(w, h)}\right).$$

Количество секторов на внешней окружности $N = \left[\dfrac{2\pi}{\alpha_0}\right]$. Это значение будет использоваться для определения детализации полусферы.

3.1.2. Построение треугольников. Используется режим тесселяции *quads, equal_spacing*: на вход поступают равномерно распределенные абстрактные координаты $(u; v)$ из диапазона $[0; 1]$ (рис. 2):

$$(u'; v') = (2u - 1, 2v - 1), \quad r_{\text{part}} = \frac{1}{\max(u', v')}, \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} = R \cdot \begin{pmatrix} r_{\text{part}} \cdot \dfrac{u'}{\sqrt{u'^2 + v'^2}} \\ r_{\text{part}} \cdot \dfrac{v'}{\sqrt{u'^2 + v'^2}} \\ \sqrt{1 - r_{\text{part}}^2} \end{pmatrix}.$$

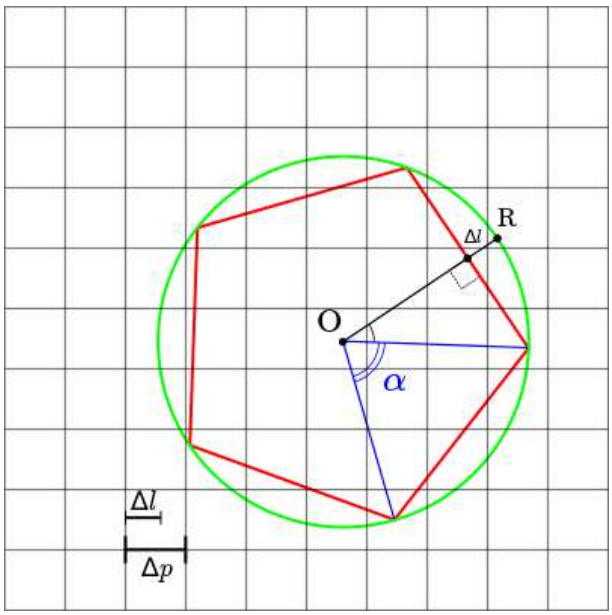


Рис. 1. Определение детализации сферы

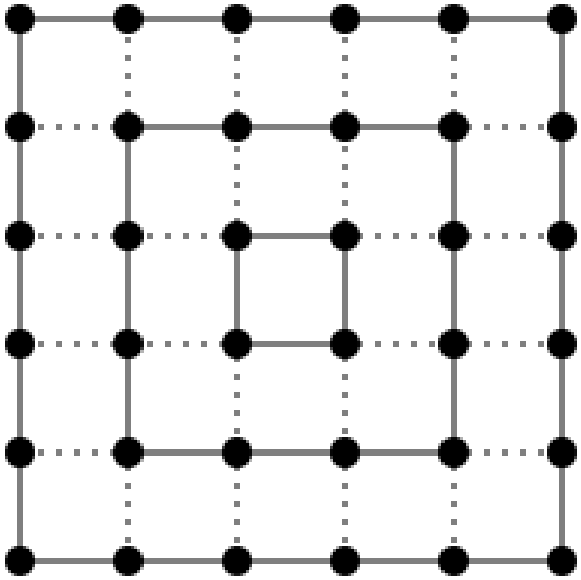


Рис. 2. Разбиение абстрактного квадрата

На выходе $(x; y; z)$ — смещение вершины относительно центра сферы в пространстве наблюдателя. Иллюстрация работы шейдеров тесселяции представлена на рис. 3.

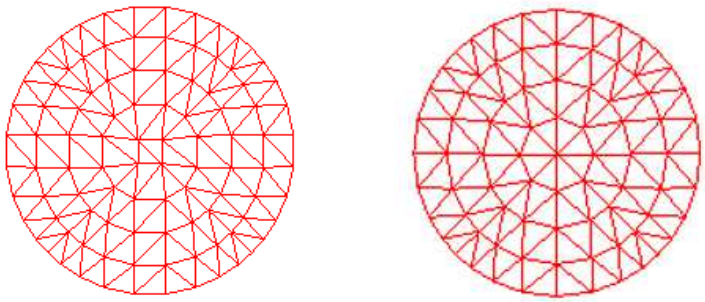


Рис. 3. Триангуляция, генерируемая шейдерами тесселяции

Наблюдается увеличение секторной детализации аппроксимаций окружностей от центра к краю. Итоговое разбиение сферы выглядит равномерным — проецируемые на экран треугольники имеют почти одинаковые площади. Это нужно для поддержания единой точности интерполяции вершинных атрибутов (здесь — нормалей).

3.2. Расчет цвета в фрагментном шейдере.

3.2.1. Преломление луча взгляда. Как было заявлено ранее, необходимо рассчитывать цвет не только передней поверхности сферы, но и учитывать заднюю часть. Точку выхода луча из сферы (она же точка задней поверхности) несложно определить, руководствуясь законом Снеллиуса.

3.2.2. Модель освещения. Рассмотрим освещение отдельной капли источником направленного света. Для начала нужно договориться о модели освещения. Будем рассчитывать диффузную компоненту освещения с помощью модели Ламберта, а бликовую с помощью модели Кука–Торренса. Подробно модель Кука–Торренса изложена в работе [11].

3.2.3. Преломление параллельных лучей света. Как передний, так и задний пиксель могут быть освещены либо прямым лучом света $\boldsymbol{L}$, либо преломленным в шаре. Установить это можно по знаку скалярного произведения луча света на внешнюю нормаль сферы. Например, для передней точки с нормалью $\boldsymbol{n_f}$ поступают так: если $(\boldsymbol{L}, \boldsymbol{n_f}) < 0$, то точка освещена прямым светом; в противном случае — преломленным.

В случае преломленного освещения необходимо найти точку входа луча света. Пусть $\boldsymbol{L}$ — направление распространения луча света; $\boldsymbol{x}$ — направление, перпендикулярное к $\boldsymbol{L}$ и такое, что $(\boldsymbol{x}, \boldsymbol{n_i}) > 0$; α — угол между $\boldsymbol{L}$ и внешней нормалью в точке входа $\boldsymbol{n_i}$; β — угол между преломленным лучом света $\boldsymbol{F}$ и $\boldsymbol{n_i}$; ξ — угол между направлением $\boldsymbol{x}$ и внешней нормалью в точке выхода $\boldsymbol{n_e}$; φ — угол между внешней $\boldsymbol{n_i}$ и $\boldsymbol{x}$.

Угол α поможет определить внешнюю нормаль в точке входа, а следовательно, и саму точку входа света. По этим данным можно будет найти преломленный луч света (рис. 4).

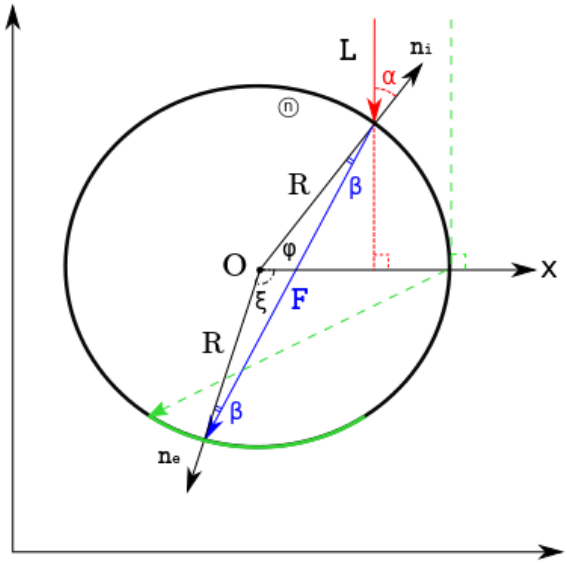


Рис. 4. Преломление луча света на частице

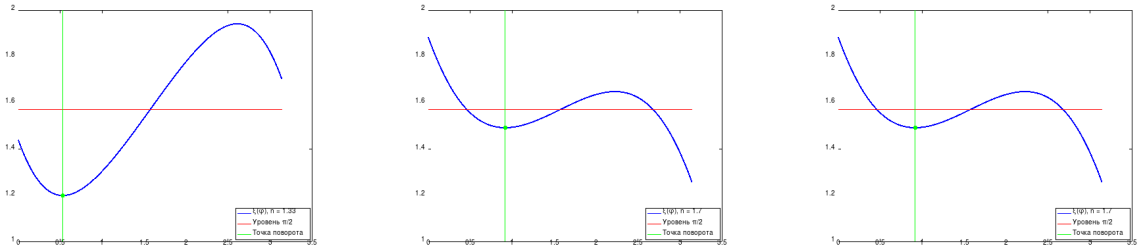
Выполнив ряд математических операций, получим уравнение четвертой степени относительно $t = \sin\beta$:

$$f(t) := 4 \cdot t^4 - 4n\cos\xi \cdot t^3 + \left(n^2 - 4\right) \cdot t^2 + \\ + 2n\cos\xi \cdot t + \cos^2\xi = 0. \tag{1}$$

Заметим, что значение $\cos\xi$ нам на самом деле известно:

$$\cos\left(\frac{\pi}{2} - \xi\right) = (\boldsymbol{L}, \boldsymbol{n_e}) \Rightarrow \cos\xi = \pm\left\|[\boldsymbol{L} \times \boldsymbol{n_e}]\right\|.$$

Решать уравнение (1) будем методом Ньютона. Из цепочки равенств $nt = n\sin\beta = \sin\alpha \leqslant 1$ получаем область поиска $t \in \left[0; \dfrac{1}{n}\right]$. Исследуем количество решений, попадающих в указанных диапазон. Построим график зависимости $\xi(\varphi)$ для нескольких значений показателя преломления n (рис. 5).



$n = 1.33$ $n = 1.7$ $n = 2.1$

Рис. 5. $\xi(\varphi) = \pi - \varphi - 2\arcsin\dfrac{\cos\varphi}{n}$

В действительности уравнение (1) является уравнением относительно $\dfrac{\cos\varphi}{n}$, а значит, относительно φ (поскольку $\varphi \in [0; \pi]$). Иными словами, ставится задача обращения функции $\xi(\varphi)$. Точками поворота называются такие значения φ^*, что $\dfrac{d\xi}{d\varphi}$ меняет знак при переходе через φ^*. Точки поворота этой функции принадлежат множеству решений следующего уравнения:

$$\frac{d\xi}{d\varphi} = -1 + \frac{2\sin\varphi}{\sqrt{n^2 - \cos^2\varphi}} = 0. \tag{2}$$

Решения уравнения (2) в диапазоне $[0;\pi]$ следующие: $\varphi_1 = \arcsin\dfrac{\sqrt{n^2-1}}{\sqrt{3}}$, $\varphi_2 = \pi - \varphi_1$. Для показателя преломления $n > 2$ функция $\xi(\varphi)$ не имеет точек поворота.

Для корректной работы метода Ньютона необходимо произвести разделение области определения функции $\xi(\varphi)$ на подобласти, в которых функция обращается однозначно. Опустим технические подробности этого процесса.

На данной задаче метод хорошо сходится уже на 2–3 итерации. Зная φ, несложно восстановить нормаль в точке входа и, соответственно, найти саму точку входа.

3.2.4. Формулы Френеля. Будем считать свет неполяризованным. Неполяризованный свет можно представить как смесь света с s-поляризацией и p-поляризацией с равными вкладами. Вычислим коэффициент прохождения света через одну поверхность сферы, введя обозначения T — для луча взгляда, T_f — для преломленного луча света, попадающего на точку передней поверхности, T_b — для преломленного луча света, попадающего на точку задней поверхности, и руководствуясь [12].

3.2.5. Формулы Френеля и направленный свет. Следует подробнее остановиться на применении коэффициентов Френеля к освещению источником направленного света. Выбранная нами модель освещения предполагает наличие прямого отраженного от освещаемой поверхности луча (бликовая компонента) и сферической волны (диффузная компонента). Формулы для коэффициентов представлены в таблице, а их обоснование приведено ниже.

Учет коэффициентов Френеля при освещении частицы направленным светом

Поверхность	Попадание луча	Диффузный фактор	Бликовый фактор
Передняя	Прямое	1	$1 - T_f$
Передняя	Преломленное	T_f	$T_f \cdot T_f$
Задняя	Прямое	T	$T_b \cdot T$
Задняя	Преломленное	$T_b \cdot T$	$T_b \cdot (1 - T_b) \cdot T$

В случае освещения передней поверхности прямым светом фактор диффузной компоненты разумно задать единичным, а бликовой — как коэффициент отражения при преломлении луча света на передней поверхности сферы.

Для случая освещения передней поверхности сферы преломленным светом фактор диффузной компоненты имеет смысл задать как T_f, учитывая таким образом прохождение света через заднюю поверхность частицы, а фактор бликовой компоненты как T_f^2, учитывая вдобавок выход луча из частицы (предполагается, что наблюдатель находится снаружи).

Когда речь идет об освещении задней поверхности, то разумно предварительно домножать как диффузный, так и бликовый факторы на коэффициент прохождения луча взгляда T через переднюю поверхность сферы. Бликовую компоненту разумно также домножить на коэффициент прохождения луча света сквозь заднюю поверхность сферы.

Рассмотрим теперь коэффициенты в случае попадания преломленного луча света на заднюю поверхность шара. Диффузный фактор: свет сначала входит в частицу (домножение на T_b), потом рассеивается на задней поверхности, а один из лучей попадает в наблюдателя — отсюда домножение на T (по аналогии с предыдущим случаем). Бликовый фактор: прохождение через поверхность сферы (T_b), отражение от задней поверхности ($1 - T_b$) и попадание в наблюдателя (T). Таким образом были определены правила корректировки интенсивности света для различных типовых случаев следования луча света.

3.2.6. Поглощение света. Согласно закону Бугера [12, гл. VIII, §84], интенсивность света, прошедшего через прозрачную среду, изменяется следующим образом: $I = I_0 \cdot e^{-\chi l}$. Примем на вооружение эту формулу и добавим коэффициент поглощения χ в список входных параметров. Применять ее будем к задней поверхности сферы и к преломленным лучам источника направленного света.

3.3. Удаление невидимых частиц. Часть частиц может быть скрыта за непрозрачными объектами сцены. Обнаружение таких частиц и исключение их из общего массива, отправляемого для отрисовки, может увеличить производительность рендеринга. Для этой цели используется техника Transform Feedback, позволяющая захватывать массивы атрибутов с видеокарты. Создаются специальные шейдеры, проверяющие, перекрывается ли частица фоном или нет.

Проверка видимости частицы происходит в геометрическом шейдере, который определяет координаты ближайшей к наблюдателю точки сферы в экранном пространстве и сравнивает значение ее глубины с значением глубины из z-буфера фона (предварительно отрендеренной непрозрачной части сцены). Ес-

ли проверка проходит успешно (точка имеет меньшую глубину), то считается, что частица видна, и ее атрибуты записываются в выходной поток.

Обычно в таких случаях используется методика Occlusion Culling, основывающаяся на вычислении MIP-уровней карты высот и взятии глубины с такого MIP-уровня, чья детализация грубее размеров проекции габаритного параллелепипеда объекта на экран. В алгоритме Depth Peeling карт глубин столько же, сколько слоев, и вычисление их MIP-уровней может повлиять на производительность. Поэтому было принято решение о реализации описанной выше проверки в ущерб корректности — капли, пересекающиеся с поверхностями, могут исчезать и появляться в зависимости от ракурса.

Выходной поток геометрического шейдера захватывается с помощью Transform Feedback. В результате получается массив такого же формата, который изначально подавался в вершинный шейдер, но меньшего размера — невидимые частицы были отброшены. Далее этот массив отдается основной цепочке шейдеров, описанной в предыдущих разделах.

Удаление невидимых частиц можно производить каскадно: перед отрисовкой очередного полупрозрачного слоя отсечем те частицы, у которых ближайшая к наблюдателю точка имеет глубину больше, чем соответствующий ей пиксель предыдущего слоя. Более того, во время первого отсечения можно оценить уровень детализации частицы и отсечь слишком маленькие.

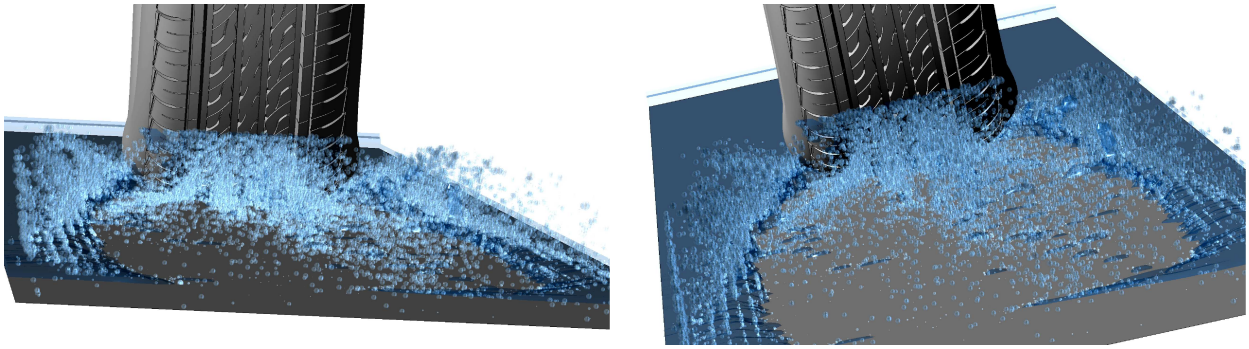


Рис. 6. Новый метод визуализации капель

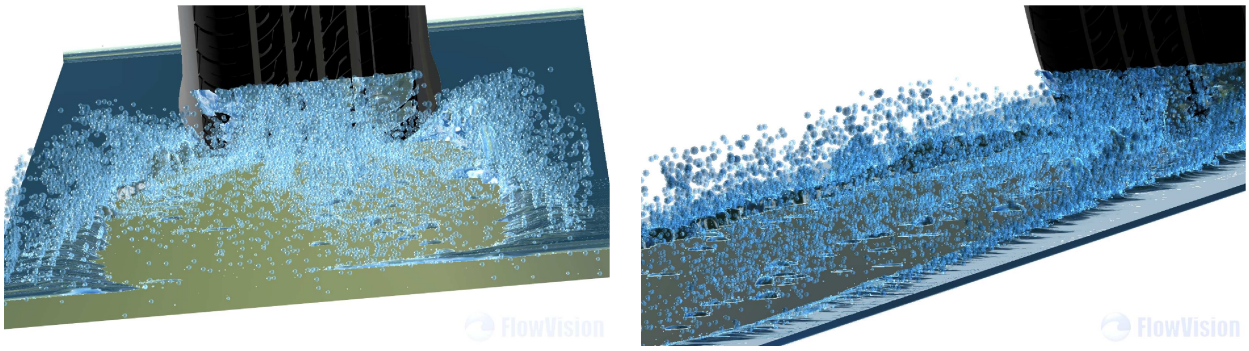


Рис. 7. Новый метод визуализации капель

4. Результаты. Была разработана и реализована методика визуализации капель жидкости для программного комплекса FlowVision. Эта методика была опробована на задаче качения шины. До внедрения этой методики капли визуализировались параллелепипедами сплошного цвета и “тонкими” в плане полупрозрачности стенками. На рис. 6 и 7 приведены изображения, демонстрирующие результаты работы описанной методики.

5. Заключение. В результате выполненной работы была усовершенствована визуализация капель жидкости в программном комплексе FlowVision.

Разработан алгоритм рендеринга сферических частиц с генерацией динамически детализированной геометрии на видеокарте и эффектами объемного поглощения и преломления направленного света, отражения и преломления *cubemap*-текстуры с учетом коэффициентов Френеля. Кроме того, реализовано каскадное отсечение невидимых и уже отрисованных частиц для повышения скорости рендеринга в режиме полупрозрачности Depth Peeling.

Статья рекомендована к публикации Программным комитетом Международной научной конферен-

ции “Суперкомпьютерные дни в России 2017” (http://russianscdays.org).

СПИСОК ЛИТЕРАТУРЫ

1. *Müller-Fischer M.* Fast water simulation for games using height fields. http://matthias-mueller-fischer.ch/talks/GDC2008.pdf. Cited December 27, 2017.
2. *Kryachko Y.* Using vertex texture displacement for realistic water rendering. https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter18.html. Cited December 27, 2017.
3. *Sousa T.* Generic refraction simulation. https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter19.html. Cited December 27, 2017.
4. *Yang Y., Zhu C., Zhang H.* Real-time simulation: water droplets on glass windows // Computing in Science and Engineering. 2004. **6**, Issue 4. 69–73.
5. *Nikata N., Kakimoto M., Nishita T.* Animation of water droplets on a hydrophobic windshield. https://www.semanticscholar.org/paper/Animation-of-Water-Droplets-on-a-Hydrophobic-Winds-Nakata-Kakimoto/0632fe77ecd159c63c1b781eab6e22767adbd9ee. Cited December 27, 2017.
6. *Davis S.T., Wyman C.* Interactive refractions with total internal reflection // Proc. of Graphics Interface. New York: ACM Press, 2007. 185–190.
7. *Everitt C.* Order-independent transparency. http://www.eng.utah.edu// cs5610/lectures/OrderIndependentTransparency.pdf Cited December 27, 2017.
8. *Segal M., Akeley K.* The OpenGL graphics system: a specification (Version 4.0 (Compatibility Profile) — March 11, 2010). https://khronos.org/registry/OpenGL/specs/gl/glspec40.compatibility.pdf. Cited December 27, 2017.
9. *Голов А.В.* Моделирование движения многофазной жидкости в программном комплексе FlowVision. https://mipt.ru/education/chair/computational_mathematics/upload/45a/2013_ApplMath_MSc_731_Golov-arphj5jlf2t.pdf. Cited December 27, 2017.
10. *Боресков А.В.* Тесселяция в современном OpenGL. http://steps3d.narod.ru/tutorials/tesselation-tutorial.html. Cited December 27, 2017.
11. *Cook R.L., Torrance K.E.* A reflectance model for computer graphics // ACM SIGGRAPH Comput. Graph. 1981. **15**, N 3. 307–316.
12. *Сивухин Д.В.* Общий курс физики. Т. IV. Оптика. М.: Физматлит, 2006.

Поступила в редакцию
11.11.2017

Droplet Visualization in FlowVision

A. A. Fedorov [1]

[1] *Limited Liability Company “Numerical Engineering Platform”; ulitsa Yunnatov 18, Moscow, 127083, Russia; Programmer, e-mail: fedorov@phystech.edu*

Received November 11, 2017

Abstract: A method for the droplet visualization in FlowVision is proposed. FlowVision uses a local adaptive parallelepiped computational grid with subgrid geometry resolution. The VOF function is used to reconstruct the fluid and droplet surfaces which are then rendered using OpenGL. A number of approaches to improve the rendering of droplets are described. The effects of light refraction and light attenuation are considered.

Keywords: fluid rendering, droplet rendering, FlowVision, OpenGL.

References

1. M. Müller-Fischer, “Fast Water Simulation for Games Using Height Fields,” http://matthias-mueller-fischer.ch/talks/GDC2008.pdf. Cited December 27, 2017.

2. Y. Kryachko, “Using Vertex Texture Displacement for Realistic Water Rendering,” https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter18.html. Cited December 27, 2017.

3. T. Sousa, “Generic Refraction Simulation,” https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter19.html. Cited December 27, 2017.

4. Y. Yang, C. Zhu, and H. Zhang, "Real-Time Simulation: Water Droplets on Glass Windows," Comput. Sci. Eng. **6** (4), 69–73.

5. N. Nikata, M. Kakimoto, and T. Nishita, "Animation of Water Droplets on a Hydrophobic Windshield," https://www.semanticscholar.org/paper/Animation-of-Water-Droplets-on-a-Hydrophobic-Winds-Nakata-Kakimoto/0632fe77ecd159c63c1b781eab6e22767adbd9ee. Cited December 27, 2017.

6. S. T. Davis and C. Wyman, "Interactive Refractions with Total Internal Reflection," in *Proc. of Graphics Interface, Montreal, Canada, May 28–30, 2007* (ACM Press, New York, 2007), pp. 185–190.

7. C. Everitt, "Order-Independent Transparency,"
http://www.eng.utah.edu// cs5610/lectures/OrderIndependentTransparency.pdf Cited December 27, 2017.

8. M. Segal and K. Akeley, "The OpenGL Graphics System: A Specification (Version 4.0 (Compatibility Profile) — March 11, 2010)," https://khronos.org/registry/OpenGL/specs/gl/glspec40.compatibility.pdf. Cited December 27, 2017.

9. A. V. Golov, *Modeling of Multiphase Fluid Flow in FlowVision.*
https://mipt.ru/education/chair/computational_mathematics/upload/45a/2013_ApplMath_MSc_731_Golov-arphj5jlf2t.pdf. Cited December 27, 2017.

10. A. V. Boreskov, *Tessellation in Modern OpenGL.*
http://steps3d.narod.ru/tutorials/tesselation-tutorial.html. Cited December 27, 2017.

11. R. L. Cook and K. E. Torrance, "A Reflectance Model for Computer Graphics," ACM SIGGRAPH Comput. Graph. **15** (3), 307–316 (1981).

12. D. V. Sivukhin, *General Course of Physics*, Vol. 4: *Optics* (Fizmatlit, Moscow, 2006).