Моделирование реалистичных атмосферных эффектов в летном симуляторе

Сапронов Роман
Казанский национальный исследовательский технический
университет им. А.Н. Туполева – КАИ, Казань, Россия
lurky@rambler.ru

Аннотация

Предложены способы динамического формирования изображения ряда атмосферных эффектов, таких, как облака, туман, дождь, снег на основе объемных моделей с использованием процедуры трассировки лучей в экранном пространстве. Предложены эффективные способы реализации указанных алгоритмов пригодные для использования не только в системе визуализации летного симулятора, но также и других приложениях реального времени.

Ключевые слова: *Визуализация, Атмосферные эффекты, Облака, Погодные эффекты, Летный симулятор, Реальное время, Модель освещения.*

1. ВВЕДЕНИЕ

Задача моделирования атмосферных эффектов находит широкое применение в различных приложениях компьютерной графики. Несмотря на то, что в ряде приложений реального времени удается достичь баланса между реалистичностью получаемого изображения и общей производительностью приложения, за счет, например, использования проекций фотографических изображений неба, в случае летного симулятора на первый план выходит необходимость моделирования атмосферы как объемной среды, что в необходимой мере соответствует ее природе, а также природе соответствующих эффектов. Визуализация атмосферных эффектов является чрезвычайно важной задачей при создании летного симулятора. Однако, в интерактивных приложениях, временной бюджет, выделяемый на их отображение существенно ограничен (как правило, не более 10 мс.). В этой связи, широко распространен подход, при котором небесная сфера представляется в виде кубической карты (Cubemap), спроецированной на внутреннюю поверхность сферы или куба достаточно большого размера. При этом, затрудняется имитация смены времени суток, анимация облаков (в подобных случаях кубическая карта получается путем обработки панорамной фотографии), поскольку, в этом случае, необходима плавная интерполяция между кубическими картами, полученными с максимальным временным разрешением.

Подходом, вызывающим наибольший практический интерес, является моделирование атмосферы, как слоя, окружающего сферическую поверхность планеты. Несмотря на очевидные преимущества данного подхода с точки зрения физической корректности, необходимо отметить, что отведенный временной бюджет при этом будет скорее всего превышен, поскольку для визуализации неба необходимо его представление в виде участка сферической полигональной сетки довольно высокой плотности; метод требует отдельного подхода для визуализации облаков; необходимо представление земной и морской поверхности в виде сферических сеток большого радиуса, что, в совокупности с требованием высокой детализации может привести к накоплению ошибок вычислений; не будут работать большинство методов визуализации, в связи с наличием кривизны поверхности (а также, необходимости представления ее в виде полигональной сетки и наличия схемы ее тесселяции). Кроме того, указанный метод имеет практический смысл при проектировании космических симуляторов, и оказывается бесполезным для симуляторов летательных аппаратов, функционирующих в нижних слоях атмосферы (0 – 20 км.).

2. ВИЗУАЛИЗАЦИЯ НЕБА

В соответствии с общепринятой методикой, небо моделируется в виде полусферической полигональной сетки, с нормалями, направленными к центру указанной полусферы. Простейший метод определения цвета неба основан на использовании градиента двух базовых цветов, выбираемых эмпирически. Однако, наиболее правдоподобные модели учитывают эффекты рассеяния света на молекулах (релеевское рассеяние), и аэрозолях (рассеяние Ми). Так, рассеянием коротковолнового излучения на неоднородностях объясняется голубой цвет неба, красный цвет на закате является следствием увеличения пути, требуемого для прохождения света до наблюдателя, в этом случае, большая часть коротковолнового излучения многократно рассеивается в разных направлениях и поглощается, не доходя до наблюдателя.

Рассеивание Ми вызвано присутствием частиц пыли и других примесей (также называемых аэрозолями), является однородным для большинства длин волн, что выражается в серых оттенках неба в туманную погоду. Указанный тип рассеяния может быть также применен для симуляции других эффектов, например, радуги.

Количество света, поступившее в одном, и рассеянное в другом направлении определяется при помощи фазовой функции, предложенной Хеньи-Гринштейном [1,2]:

$$F(\theta, g) = \frac{3(1-g^2)}{2(2+g^2)} \frac{(1+\cos^2\theta)}{(1+g^2-2g\cos\theta)^{1.5}}, \quad (1)$$

которая выражает зависимость угла между входящим и исходящим направлениями θ и параметром g. Нетрудно

заметить, что при нулевом значении указанного параметра получаем функцию, выражающую рассеяние Релея, отрицательные значения параметра определяют функцию, рассеивающую большую часть энергии в направлении к наблюдателю (положительные – в обратном). Для воспроизведения характера рассеивания на частицах принимают $g \in (-0.75, -0.99)$.

Для каждой видимой точки полусферы цвет рассчитывается по формуле, приведенной в работах Т. Нишиты и И. Добаши [3,4]:

$$I_{\upsilon}(\lambda) = I_s(\lambda)K(\lambda)F(\theta, g)\int_{P_a}^{P_b} e^{\frac{-h}{H_0}} e^{\left(-t(PP_c,\lambda)-t(PP_a,\lambda)\right)} ds\,, \quad (2)$$

где $I_{\upsilon}(\lambda)$ - значение яркости, рассчитываемое по каждой из компонент цветовой системы RGB; $I_s(\lambda)$ - яркость солнечного излучения; $K(\lambda)$ - постоянная рассеяния, определяемая в соответствии с используемой моделью рассеяния (Ми или Релея); $F(\theta, g)$ - фазовая функция, определенная выше; P_a и P_b - точки выхода и вхождения луча в атмосферу; h - относительная высота текущей точки над уровнем моря (0 – уровень моря, 1 – максимальная высота видимого слоя атмосферы); H_0 - относительная высота, соответствующая усредненному значению атмосферной плотности (в большинстве реализаций принято значение ¼); функции t выражают ***оптическую толщину*** на отрезках PP_c и PP_a, и определяются следующим образом:

$$t(P_aP_b, \lambda) = 4\pi K(\lambda)\int_{P_a}^{P_b} e^{\frac{-h}{H_0}} ds\,. \quad (3)$$

При численном решении уравнения 3 интегралы заменяются конечными суммами.

На практике удалось перенести генерирование двумерных таблиц рассеяния Ми и Релея на GPU, с обеспечением частоты, соответствующей приложениям, выполняющимся в режиме реального времени, однако, в связи с высокой интенсивностью вычислений в целом, и специфики разрабатываемого приложения, были внесены следующие дополнительные изменения:

1. Небо моделируется полусферой, центр которой совпадает с текущим положением видовой камеры, в связи с этим, цвета по-прежнему рассчитываются при пересечении лучей от наблюдателя до верхнего края атмосферы, но проецируются на локальный сферический меш сферы;
2. Земная поверхность представлена высотной картой, проецируемой на плоскость;
3. Выход за пределы земной атмосферы не рассматривается, ввиду особенности моделируемых летательных аппаратов, что уменьшает количество возможных вариантов, и соответственно шейдерных программ.

Кроме того,

1. Проведено кэширование значений яркостей для суточной траектории движения Солнца в ряд таблиц, с возможностью имитации смены его положения путем линейной интерполяции значений соседних таблиц, что упрощает вычисления до двух чтений двумерной текстуры, что превосходит по производительности и используемым ресурсам видеопамяти ряд методов, использующих трехмерные текстуры [5,6], при этом отсутствие множественно рассеяния может быть скомпенсирован рядом подходов, среди которых снижение насыщенности при помощи эмпирической функции, использование процедуры трассировки лучей;
2. Устранены ресурсоемкие вычисления, связанные с вычислением двойных интегралов в уравнении (2) (при этом возможно вычисление значений следующей таблицы в параллельном потоке средствами центрального процессора);

2.1 ВИЗУАЛИЗАЦИЯ ОБЛАКОВ

При визуализации атмосферы, неизбежно возникает задача реалистичного моделирования облаков. В этой связи необходимо отметить несколько наиболее распространенных подходов:

1. Представление облаков в виде анимированных во времени двумерных текстур, проецируемых на поверхность небесной сферы (либо кубических карт, созданных из фотографий, содержащих в себе изображения облаков), к этой категории можно отнести множество компьютерных игр (серии Quake, Doom, Fear, Serious Sam, Painkiller, и т.д.);
2. Моделирование при помощи полигональной сетку [7];
3. Представление облаков в виде одиночного или нескольких слоев – отдельного полигона, либо группы полигонов, параллельных земной поверхности [8];
4. Моделирование облаков в виде системы полигональных частиц, обращенных к наблюдателю (импосторов) [9,10];
5. Представление облаков в виде элементов объемного изображения, вокселов (при этом, физически, вокселы могут быть интерпретированы с использованием всех вышеуказанных методов, включая преобразования объемной структуры данных в полигональную сетку, а также, рассмотренных ранее методов трассировки луча [11–13].

Приведем наиболее известные приемы, используемые при анимации облаков:

1. Системы, основанные на физической симуляции динамики жидкостей [3]
2. Алгоритмы, основанные на конечных автоматах, выдающие одни из наиболее правдоподобных результатов, требующие при этом, тщательной настройки и балансировки для достижения необходимой стабильности [14,15];
3. Алгоритмы, основанные на одной из шумовых моделей [11];
4. Подходы, использующие некий трехмерный объем, заполняемый случайным образом расположенными импосторами [10,16,17].

Так, в большинстве симуляторов полета (включая известные серии MSFS, серию X-Plane, Flight Gear и т.д.), и многих компьютерных играх серии игр Far Cry, Crysis, Uncharted, и т.д.), используется система частиц, размещаемых и анимируемых вручную. При таком подходе возможен полный контроль формы и поведения моделируемых объектов, увеличивая, вместе с тем, трудозатраты, связанные с ручной настройкой всех необходимых параметров. Большинство алгоритмов, позволяющих получить наиболее реалистичное изображение, зачастую слишком медленны (хотя и выполняются отдельно от других компонентов системы в интерактивное время), либо требуют расширенной аппаратной поддержки для достижения максимального качества.

В соответствии с разработанной моделью, функция плотности облака математически представляется в виде суммы нескольких октав трехмерного шума:

$$q(\mathbf{p},t_i)=\sum_{k=0}^{N_{oct}} w_k N_k(\mathbf{p},\mathbf{v_k},a_k,b_k,c_k,t_i), \quad (4)$$

где $w_k < w_{k-1}$; $a_k > a_{k-1}$; параметры a_k, b_k, c_k, w_k, количество октав N выбираются эмпирически, $\mathbf{v_k}$ - вектор скорости смещения k -ой октавы, N_{oct} -количество октав шума во времени. В ходе анализа фотографических изображений различного типа облаков выведено следующее соотношение для $N_k(\mathbf{p},\mathbf{v_k},a_k,b_k,c_k,t_i)$:

$$N_k(\mathbf{p},\mathbf{v_k},a_k,b_k,c_k,t_i)=r_k-\left|o_k-(N(\mathbf{p}a_k+\mathbf{v_k}t_i)b_k+c_k)\right|. \quad (5)$$

Выбор одной из моделей шума осуществляется исходя из наблюдаемых особенностей конкретного типа облаков, для получения кучевых облаков используется вторая модель, для получения слоистых – первая. При моделирования перистых и перисто-слоистых облаков координаты вектора $\mathbf{p}$ смещаются на значения, определяющиеся той же шумовой функцией:

$$\mathbf{p}_k=\mathbf{p}_{k-1}+N_{k-1}\mathbf{s}_k, \quad (6)$$

где $\mathbf{s}_k$ - вектор масштабирования, компоненты которого характеризуют величину искажений по каждой из осей.

В качества шумовой функции используется виртуальная трехмерная текстура, получающаяся из двумерной шумовой текстуры смещением каждого слоя на константу, текстурные координаты при этом определяются следующим образом:

$$\begin{aligned}\mathbf{a}_{xy}&=\mathbf{p}_{xz}+\lfloor\mathbf{p}\rfloor_y k_{pr}\\ \mathbf{b}_{xy}&=\mathbf{p}_{xz}+(\lfloor\mathbf{p}\rfloor_y+1)k_{pr}\end{aligned}, \quad (7)$$

результат выборок из текстуры по указанным координатам смешивается с коэффициентом $\{\mathbf{p}\}_y$. Преимущества указанного подхода над выборкой из трехмерного объема очевидны: при выборке из трехмерного объема 1024^3 потребуется до 4 Гб памяти для размещения одной текстуры, в случае же двумерной текстуры объем памяти не превысит 4 Мб. Указанной особенностью объясняется низкое качество облаков в большинстве недавних компьютерных игр (Horizon: Zero Dawn [18], Reset и т.д.). Кроме того, в случае использования указанного подхода возможно задавать уникальные коэффициенты смещения $\mathbf{v_k}$ для каждого из слоев, что затруднено в случае использования трехмерной текстуры. Однако, количество выборок из двумерной текстуры оказывается существенно выше, чем в случае трехмерной текстуры, в связи с чем, семплы по координатам $\mathbf{a}_{xy}$ и $\mathbf{b}_{xy}$ записываются в разные каналы одной текстуры (также возможно записать значения следующей октавы в оставшиеся два канала), что существенно повышает производительность в 3 – 4 раза) по сравнению с использованием первоначального подхода.

Приведем основные шаги разработанного алгоритма освещения:

1. Найти пересечения видового луча с плоскостями визуализируемого объема, представленных соответствующими уравнениями плоскости (в общем случае возможно нахождение луча, соединяющего позицию наблюдателя и трехмерную координату, определяемую соответствующим значением глубины, однако использование дополнительного пересечения хотя бы с одной из плоскостей может уменьшить количество итераций, необходимых для построения указанного объема).
2. Используя найденные ранее значения начальной и конечной точек, полученных в результате пересечения видового луча с указанными выше плоскостями виртуального объема (либо граничные точки, представленные позицией наблюдателя и значением координаты, полученной из G-Buffer'a), задаваясь величиной шага трассировки, найдем соответствующий вектор приращения.
3. В цикле, двигаясь по лучу, будем находить очередное значение цвета, смешивая его с полученным на предыдущем шаге по формуле $C_i^{\Sigma}=lerp(C_{i-1}^{\Sigma},C_i,a_i\mu\Delta p)$, где накапливаемое значение цвета C_i^{Σ} на текущем шаге, вычисляется посредством линейной интерполяцией ($lerp$) значения C_{i-1}^{Σ}, полученного на предыдущем шаге, а также значения цвета C_i, вычисленного на текущем шаге, a_i - значение прозрачности, а $\Delta p=\|\mathbf{p}_i-\mathbf{p}_{i-1}\|$ - расстояние между текущим и предыдущим положениями точки на луче, μ - константа, определяющая удельное значение прозрачности.

При этом, значения цвета и прозрачности на текущем шаге - C_i и a_i, соответственно, могут быть получены в ходе следующей последовательности действий:

1. На основании координаты точки $\mathbf{p}_i$ получить значение плотности $N_{\Sigma}(\mathbf{p}_i,t_i)$ путем сложения функций нескольких октав трехмерной шумовой функции.
2. Получить значение прозрачности a_i используя найденное ранее значение плотности, $a_i=N_{\Sigma}(\mathbf{p}_i,t_i)\cdot g((p_y-h_{cloud})/\delta_{cloud})$, где p_y - высота

точки $\mathbf{p}_i$, $h_{cloud} = (h_{\min} + h_{\max})/2$ - высота слоя, δ_{cloud} - толщина указанного слоя, $g(x)$ - линейная функция вида: $g(x) = ax + b$.

3. На основании оригинального значения плотности $N_\Sigma(\mathbf{p}_i, t_i)$ и значения, смещенного по направлению к источнику $N_\Sigma(\mathbf{p}_i + \mathbf{l}m, t_i)$ освещения получить направленную компоненту освещения C_{sc} (в большинстве известных подходов [18] для получения компоненты направленного освещения используется 6 и более выборок текстуры по направлению к источнику освещения, что в совокупности с большим количеством шагов по лучу приводит к существенному падению производительности);
4. Найти цвет в данной точке как сумму количества света, полученного от источника C_d, отраженного земной поверхностью C_{in} и рассеянного освещения C_a: $C_i = lerp(C_a, C_d, df) + C_i$, где $C_a = lerp(L_{sky}, L_{sun}, \mathrm{gr})$; L_{sky}, L_{sun}, gr - цвет неба, Солнца и высотный градиент, соответственно; $C_d = L_{sun} \cdot f\left[1 - \left(\hat{\mathbf{h}}, \hat{\mathbf{v}}\right)\right]$;

$\hat{\mathbf{h}} = (\mathbf{l} + \mathbf{v}) / \|\mathbf{l} + \mathbf{v}\|$ - вектор половинного угла, $\hat{\mathbf{l}}$, $\hat{\mathbf{v}}$ - нормализованные векторы направления на источник освещения и к наблюдателю, f - функция вида $f(x) = kx^n$; $C_{in} = L_{in}(1 - \mathrm{gr})^m$; L_{in} характеризует количество освещения, отраженного земной поверхностью, k и n, m - некоторые константы, выбираемые эмпирически;

$df = \gamma(C_{cs})$, где $\gamma(x) = (kx + m)^{n_i}$.

2.2 ВИЗУАЛИЗАЦИЯ ПОГОДНЫХ ЭФФЕКТОВ

При реализации погодных эффектов, возникает необходимость реалистичного отображения таких явления, как дождь и снег. Традиционно, существует большое количество техник использующих системы частиц, моделирующих отдельные капли дождя или хлопья снега. Однако, ввиду того, что указанные эффекты обладают объемной природой, и необходимости учета взаимодействия указанных эффектов и окружающих поверхностей (например, намокания участков поверхности), а также с целью унификации средств конфигурирования указанных эффектов был избран процедурный подход, аналогичный примененному для моделирования облаков. Здесь одна и та же шумовая текстура задает как представление отдельных капель, так и внешний вид поверхности, подвергшейся воздействию одного из указанных эффектов. При этом, текстурные координаты для представления отдельных капель определяются следующим соотношением:

$$uv = \left[\frac{1}{Th} + t\mathbf{v}_v, \frac{1}{L} + t\mathbf{v}_h\right]\mathbf{M}. \qquad (8)$$

Здесь, Th характеризует толщину капель, $L-$ длина капель (при моделировании снежных хлопьев выбираем $L \sim Th$, для имитации дождевых капель $L \gg Th$), t -переменная времени, $\mathbf{v}_v$ и $\mathbf{v}_h$ -вертикальная и горизонтальная компоненты вектора скорости соответственно, $\mathbf{M}$ - двумерная матрица ориентации эффекта относительно мировой системы координат. Для имитации эффекта объема используется несколько анимируемых плоскостей, расположенных перпендикулярно вектору наблюдения.

С целью моделирования эффекта намокания (а также снежного покрытия) формируется карта, аналогичная карте теней, которая маркирует участки мокрой (занесенной снегом) поверхности, на основе которой строятся отражения окружающей сцены, с использованием метода трассировки лучей в экранном пространстве. В случае дождя, применяется дополнительная анимированная карта смещений для симуляции следов капель, где высоты волн, вызываемые падающими капель в каждый момент времени, задаются уравнением:

$$\frac{\partial^2 z}{\partial t^2} = v^2\left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2}\right). \qquad (9)$$

Необходимо отметить, что метод трассировки лучей также может быть применен и для определения участков намокания, что, однако, подвержено артефактам, присущим большинству локальных методов. Эффект неоднородного тумана (дымки) реализован методом, аналогичным приведенному для визуализации облаков. Результаты визуализации облаков представлены на рис. 1 и рис. 2, результаты визуализации дождя и тумана – на рис. 3. и рис. 4. Все результаты получены в реальном времени (50 кадров/сек.) на мобильной видеокарте и процессоре (Intel Core i7-Q720M, ATI Radeon HD5870).

Рис.1. Визуализация облаков и атмосферы при положении наблюдателя ниже уровня облаков

Рис. 2. Визуализация облаков при положении наблюдателя выше уровня облаков

Рис. 3. Визуализация дождя, для лучшего восприятия эффект увеличен в два раза

Рис. 4. Имитация эффекта объемного тумана над водной поверхностью

3. ЗАКЛЮЧЕНИЕ

В статье предложены методы динамического формирования изображения ряда атмосферных эффектов, таких, как облака, туман, дождь, снег на основе объемных моделей с использованием процедуры трассировки лучей в экранном пространстве. Предложены эффективные способы реализации указанных алгоритмов.

4. БЛАГОДАРНОСТИ

Прикладные научные исследования проведены при финансовой поддержке государства в лице 1) Фонда содействия развитию малых форм предприятий в научно-технической сфере (договор № ГУ2/2015) 2) Минобрнауки России (уникальный идентификатор прикладных научных исследований RFMEFI57414X0105).

5. ССЫЛКИ

1. O'Neil S. Accurate atmospheric scattering // GPU Gems. 2005. Vol. 2. P. 253–268.
2. Nielsen R.S. Real Time Rendering of| Atmospheric Scattering Effects for Flight Simulators. Technical University of Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark, 2003.
3. Miyazaki R., Dobashi Y., Nishita T. Simulation of cumuliform clouds based on computational fluid dynamics // Proc. Eurographics 2002 Short Present. 2002. P. 405–410.
4. Dobashi Y., Nishita T., Yamamoto T. Interactive rendering of atmospheric scattering effects using graphics hardware // Graph. Hardware'. 2002. Vol. 02pages. P. 99–108.
5. Bruneton E., Neyret F. Precomputed atmospheric scattering // Comput. Graph. Forum. 2008. Vol. 27, № 4. P. 1079–1086.
6. Elek O. Rendering Parametrizable Planetary Atmospheres with Multiple Scattering in Real-Time // Cescg. 2009.
7. Bouthors A. et al. Interactive multiple anisotropic scattering in clouds // Proc. 2008 Symp. Interact. 3D Graph. games - SI3D '08. 2008. № c. P. 173.
8. Bouthors A., Neyret F., Lefebvre S. Real-time realistic illumination and shading of stratiform clouds // Nat. Phenom. 2006. P. 41–50.
9. Wang N. Realistic and Fast Cloud Rendering // J. Graph. Tools. 2004. Vol. 9. P. 21–40.
10. Harris M.J., Lastra A. Real-Time Cloud Rendering // Comput. Graph. Forum. 2001. Vol. 20. P. 76–85.
11. Kajiya J.T., Von Herzen B.P. Ray tracing volume densities // ACM SIGGRAPH Comput. Graph. 1984. Vol. 18. P. 165–174.
12. Wrenninge M., Zafar N. Bin. Production Volume Rendering Fundamentals // SIGGRAPH 2011 Course Notes. 2011. P. 71.
13. Engel K. et al. Real-time volume graphics // Proc. Conf. SIGGRAPH 2004 course notes - GRAPH '04. 2004. 29-es p.
14. Dobashi Y. et al. A simple, efficient method for realistic animation of clouds // SIGGRAPH. 2000. P. 19–28.
15. Wang H., Meng F. Real-time Simulation of Dynamic Clouds Based On Cellular Automata // Int. J. Hybrid Inf. Technol. 2013. Vol. 6, № 5. P. 171–182.
16. Lagae A., Dutré P. An alternative for Wang tiles: colored edges versus colored corners // ACM Trans. Graph. 2006. Vol. 25, № 4. P. 1442–1459.
17. Yusov E. High-Performance Rendering of Realistic Cumulus Clouds Using Pre-computed Lighting. 2014.
18. Schneider A. Real-Time Volumetric Cloudscapes // GPU Pro 7 Adv. Render. Tech. CRC Press, 2016. P. 97.

Об авторах

Роман Сапронов – н.с. КНИТУ-КАИ имени А.Н. Туполева, магистр. E-mail: lurky@rambler.ru.

Modeling of realistic atmosphere effects in the flying simulator.

Abstract. Rendering techniques of various atmosphere effects like clouds, fog, rain and snow based on volumetric models using screen-space ray-tracing technique are proposed. Approaches to implementing all of these techniques are described, allowing them to be used in the broad range of real-time applications are proposed.

Ключевые слова: *Rendering, Atmosphere Effects, Clouds, Weather Effects, Flying Simulator, Real-Time, Lighting Model*

Roman Sapronov – KNRTU-KAI named after A.N. Tupolev

.