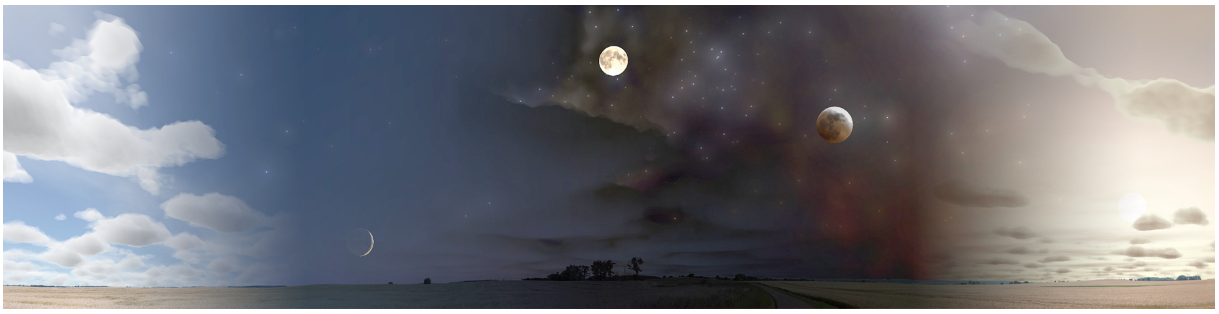
Vision, Modeling, and Visualization (2012)
M. Goesele, T. Grosch, B. Preim, H. Theisel, and K. Toennies (Eds.)

Single-Pass Rendering of Day and Night Sky Phenomena

Daniel Müller   Juri Engel   Jürgen Döllner

Hasso-Plattner-Institut, University of Potsdam, Germany



Abstract

This paper presents astronomical based rendering of skies as seen from low altitudes on earth, in respect to location, date, and time. The technique allows to compose an atmosphere with sun, multiple cloud layers, moon, bright stars, and Milky Way, into a holistic sky with unprecedented high level of detail and diversity. GPU generated, viewpoint-aligned billboards are used to render stars with approximated color, brightness, and scintillations. A similar approach is used to synthesize the moon considering lunar phase, earthshine, shading, and lunar eclipses. Atmosphere and clouds are rendered using existing methods adapted to our needs. Rendering is done in a single pass supporting interactive day-night cycles with low performance impact, and allows for easy integration in existing rendering systems.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—

1. Introduction

Real-time image synthesis targeting photo-realistic representations of arbitrary virtual 3D environments is steadily becoming more important. Improving hardware capabilities allow rendering systems to simulate a growing number of visual subtleties of reality. Applications such as video games and simulators (e.g., flight and vehicle simulators), as well as architectural and historical visualizations demanding spatiotemporal correctness, often benefit from appropriate environment rendering and seamless transitions of day and night skies. Appropriateness begins with an adequate impression of a static blue sky; It ends with immersive, dynamic simulation of environments, aware of date, time, location, lighting, weather, ambient noise, and mood, that are coherent with our daily, visual experience.

There are various approaches on real-time atmosphere and cloud rendering. However, there is a gap concerning seamless day-night transitions and astronomical accurate night sky approximations. In this work, we focus on night sky features and their composition of existing techniques into dynamic, holistic day-night cycles. We propose two techniques, for efficient synthesis of the following phenomena:

- Dynamic moon rendering featuring an astrophysical 3D moon simulated on a viewpoint oriented quad, lit and shaded based on celestial positions. For lunar eclipses, precomputed brightness and color are used.
- Rendering of thousands of individually configurable (e.g., brightness and color), twinkling (scintillations) stars using point-light rendering. For correct background illumination, faint stars that indicate the Milky Way, are added by a textured cube.

The proposed single-pass techniques can be easily integrated into multi-pass rendering techniques and are well suited for arbitrary post-processing. This provides an accurate, dependence-free basis with minimal performance impact. Astronomical accuracy in position is verified numerically, correctness in color, however, depends on a multitude of influences like varying resolution, color calibration of physical output, and subjective agreement that strongly relies on individual expectations and experience. Accuracy tradeoffs in favor of performance are justified for use-cases in, e.g., education, training environments, and entertainment.

1.1. Related Work

A framework for full day-night cycles is described in Jensen et al. [JDD*01], though, no detailed information on their tone-mapping implementation is available. Most aspects of day and night sky rendering have been examined in isolation. A collection of astronomical algorithms is given by Meeus [Mee94].

Rendering of stars has gained little attention so far, and is mostly done with static noise textures [RP05] or high resolution star texture with real positions and colors [JDD*01]. Nadeau et al. [NGN*00] suggested Gaussian spots attenuated with distance to overcome drawbacks of texture based methods, such as rendering artifacts on camera movement due to sampling. Magnor et al. [MSK*10] notes that no solutions covering outscattering and scintillations of stars are available.

The moon is commonly modeled as a separate or aggregated static texture with predefined phase, position, color, and intensity. Much more detail was spent in the research of Jensen et al. [JDD*01], who feature photo-realistic renderings with correct size, positioning, orientation, and shading based on lunar surface scattering. Oberschelp et al. [OHS01] presented a technique for rendering of solar eclipses in commercial 3D animation software. Yapo and Cutler [YC09] recently suggested photon tracing for physically approximated coloring during lunar eclipses. Both approaches however, were not feasible for real-time systems.

Leaving aerial perspective to post-processing, most approaches for atmospheric scattering satisfy our single-pass constraint. Bruneton and Neyret [BN08] presented a sophisticated algorithm that accounts for various phenomena including the earth shadow. For cloud rendering, 2D noise approaches [RP05, HKA05] pretending semi 3D cloud coverage through scattering approximations [Dub05] seem adequate enough, are utilized.

1.2. Overview

The remainder of this paper is structured as follows: The next two sections introduce a new model for moon and lunar eclipse rendering. Section 4 explains rendering of stars efficiently utilizing the GPU. In Section 5, sequential blending of discussed phenomena is briefly shown. Modifications made on atmosphere and cloud rendering are indicated. Section 6 presents and discusses our results including a performance evaluation. Finally, Section 7 summarizes the presented techniques and proposes future work.

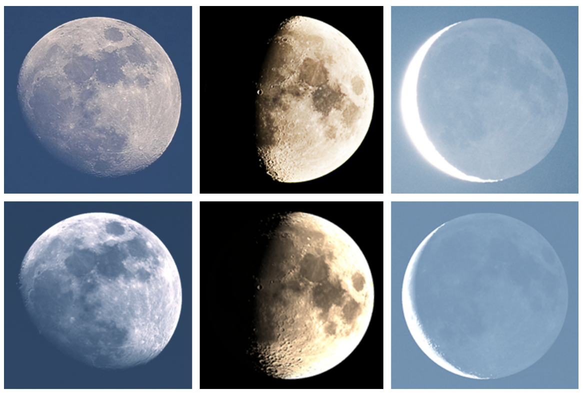


Figure 1: *Various photos of the moon in the top and our renderings in the bottom row, at corresponding day, time, and location. From left to right: nearly full moon, waxing gibbous at night, and nearly new moon with earthshine.*

2. Rendering the Moon

The moon is modeled entirely on the GPU as a viewpoint oriented billboard: a quad, projected onto a unit-sphere's tangent plane, oriented with respect to the world's up direction. On that we simulate the virtual moon-sphere. Figure 1 lists some close-ups of the moon, as seen under clear conditions. The unit-sphere is centered on the topocentric observer and projection is done towards the moon's position. The moon-sphere is extrapolated from intersections between the view ray and the quad. Texture coordinates u and v assigned to the quad, yield this quad intersection in fragment stage. With that, the sphere's selenocentric z-coordinate is given by $z^2 = 1 - u^2 - v^2$. Each virtual moon-sphere fragment is simultaneously defined in relative position and orientation by the normal $n_m = (u, v, z)$. Fragments outside the sphere are discarded. Based on the actual moon distance, the billboard is appropriately scaled and oriented for the topocentric observer. The basic moon model is annotated in Figure 2.

2.1. Apparent Position, Size, and Orientation

For the apparent position and projection direction m of the moon, its position in ecliptic coordinates is retrieved first, then converted to refraction corrected, apparent horizontal coordinates [Mee94]. The refraction of air affects the true altitude and accounts for a displacement towards the local zenith. At sea level, refraction accounts for a displacement of about 36 arc minutes near the horizon, which corresponds to the apparent moon size itself. Finally, horizontal azimuth

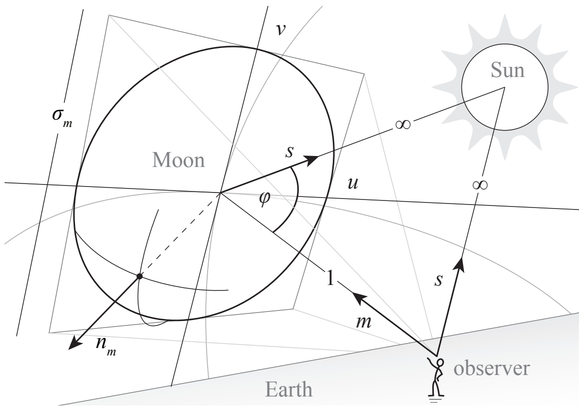


Figure 2: *The virtual moon-sphere. Note that the same s is used for observer and moon. u and v span the tangent plane used for retrieval of the moon's surface normal n_m.*

and altitude are converted to normalized Euclidean coordinates and passed to the GPU.

Refraction actually needs to be applied per fragment, to achieve deviation from circular of the moon near or below the horizon. In favor of a simpler model, we exclusively apply refraction to the moon position. However, shading and lunar eclipses, are correct only without refraction applied.

2.1.1. Distance and Apparent Size

The earth-moon distance is obtained from center to center and varies within lunar perigee and apogee of roughly 363 300 km and 405 500 km. The topocentric observer position and true earth radius at that point are not considered, leading to a maximum error in angular diameter of about 1%. Knowing the distance, the apparent angular moon diameter δ, describing its visible size on earth, can be approximated with basic trigonometry and varies between 0.49 and 0.56 degrees. The virtual moon-sphere diameter σ_m (billboard side length) is expressed as:

$$\sigma_m(\delta) = 2\tan(c_d\, \delta/2), \tag{1}$$

with c_d as artificial scaling factor. Using the correct apparent size in standard field of views (FoV) yields a subjectively small moon, making artificial resizing necessary. Values of c_d between 2 and 3 lead to less irritating moon sizes.

2.1.2. Orientation

The moon is in synchronous rotation with the earth, always revealing the same hemisphere to earthly observers. However, for correct orientation we still have to account for oscillating motions – known as librations – and the observer correlated, parallactic angle.

Our model neglects diurnal and physical librations as unobservable small. The remaining two optical librations amount to an additional, visual surface disclosure of 9% over time however. They are approximated in selenographic coordinates, referring to the mean center of the moon's apparent disk [Mee94]. Libration in latitude b is the angle between the prime meridian of the apparent lunar disk and its rotation axis, alternately revealing north and south pole. Libration in longitude l is the lateral rotation around the perpendicular axis of the lunar ecliptic. The selenocentric orientation matrix for the moon R_m is defined as:

$$R_m = R_x(b)\, R_y(l)\, R_z(p-a). \tag{2}$$

R_x, R_y, and R_z denote counter-clockwise rotation matrices around the principal axes, a is the position angle of axis and p the parallactic angle. Together they describe the orientation around the observer-moon axis, required to account for the observer's topocentric position on a rotating earth.

2.2. Coloring and Shading of the Moon

The illumination of the moon-sphere mainly depends on perspective changes in position of the moon terminator (day-night boundary) and is given by the illuminated fraction of its perceivable disk. Usually to obtain this fraction, one requires to calculate the lunar phase angle φ which is the selenocentric elongation of the earth from the sun. Given that, the position angle of the bright limb can be calculated.

Instead, the lunar phase angle φ is simply derived from horizontal moon and sun position. Using φ as angle between incident and reflected light on the moon surface, yields correct illumination and with that the correct illuminated fraction [JDD*01]. Simplifying the sun position s as directional light source, seen from earth instead of the moon, introduces an indiscernible maximum error in lunar phase angle of about 9 arc minutes. Shading, albedo, and earthshine are approximated as functions of φ. The final surface color I_m, is defined by reflected sun and earth light:

$$I_m(\varphi) = k_\lambda\, a\left(F(\theta_i, \theta_r, \varphi) + \beta_e\, E_{em}(\varphi)\right), \tag{3}$$

with a as surface albedo, β_e E_{em} introducing earthshine, F the Hapke-Lommel-Seeliger BRDF that approximates real moon reflectance [Hap66]. θ_r is the angle between n_m and reflected light, θ_i between n_m and incident light. By multiplying moon-sphere normals with the billboard's orthonormal matrix, they are transformed into required horizontal space. k_λ is used to adjust final color and intensity.

2.2.1. Earthshine

The bluish, faint light of the dark moon fraction known as earthshine, is caused by reflections of earth emitted light. Its intensity E_{em} depends on the phase angle and is most intense during new moon. It is approximated by:

$$E_{em}(\varphi) = -0.0061\,\varphi^3 + 0.0289\,\varphi^2 - 0.0105\,\sin(\varphi), \tag{4}$$

introducing a maximum difference of about 3% to the formula suggested by van de Hulst [vdH80]. For the earthshine color coefficient, $\beta_e = (0.88, 0.96, 1.00)$ is used.

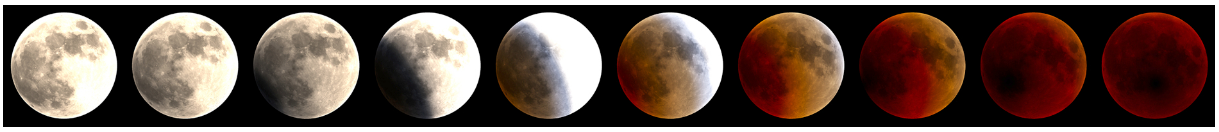


Figure 3: *Renderings of the June 15th 2011 total eclipse, as seen from Mangalore, India, using our technique. The images correlate to universal time in 15 minute intervals, beginning at 18:00 UTC on the left.*

2.2.2. Albedo and Surface Normals

Albedo values and surface normals are encoded in a four channel cubic environment map. The albedo is based on Clementine data, that represents only partially true albedo. The surface was photographed with the sun being always near the cameras longitude, causing static shadows towards the poles. Aggregated imagery taken from earth at full moon would be better suited, but we currently are unaware of any such resource. If librations were not taken into account, a photograph of the full moon would be satisfactory.

The surface normals are based on stereo images taken by the LRO-WAC camera. Slight surface irregularities are important inside the day-night boundary, but even there, they are hard to perceive using correct apparent size. Nevertheless, they are linearly blended with the moon-sphere normals, depending on the desired intensity. We use a texture resolution of 256^2 pixel per cube face to provide generous field of views and closeups.

The dusty moon surface has a slight reddish hue, that is simulated by fitting the average lunar albedo to a measured reflectance spectrum as shown by Yapo and Cutler [YC09]. Using their second-degree polynomial, color channel coefficients $k_\lambda = (0.92, 0.79, 0.64)$ based on representative wavelengths $(680, 550, 440)$nm are obtained [REK*04]. As for stars, atmospheric outscattering is applied too (Sec. 4.3.1).

3. Rendering Lunar Eclipses

We propose a new algorithm for realistic rendering of lunar eclipses in real-time, aiming for photometric resemblance. The magnitude, describing the moon's penetration depth into the earth umbra or penumbra during an eclipse, is easily retrieved within our model. Color and brightness are expressed as two separate multiplier, each correlated to the magnitude, offering control similar to exposure time in photography. Figure 3 shows a half cycle using our technique.

3.1. Modeling Lunar Eclipses

The earth illuminated by the sun, casts a penumbra and within that an umbra as shown in Figure 4. Their diameter ε_u and ε_p are measured in moon radii and can be either simplified to be constant – by assuming average moon and sun distances, ε_u and ε_p could be approximated with 2.65 and 4.65, respectively – or calculated based on actual moon and sun distances d_m and d_s, as well as constant radii:

$$\varepsilon_u = 3.6676 - 397.0001\, d_m\, d_s^{-1}, \tag{5}$$

$$\varepsilon_p = 3.6676 + 404.3354\, d_m\, d_s^{-1}. \tag{6}$$

The horizontal earth-moon system is assumed to be normalized, yielding a moon distance of 1 (Fig. 4). For each visible moon fragment we retrieve the normalized distance between the related moon-sun line g and the earth center, simplifying the sun as directional light again. The position of a moon-sphere fragment a_f is given by $a_f = m - \varepsilon_0\, n_m$, with ε_0 as actual moon radius to moon distance ratio. Using the moon-sun line in the parametric form $g : x \Rightarrow a_f + xs$, the shortest distance D_f in moon radii between a_f and g, henceforth referred to as eclipse phase, is given by $\varepsilon_0\, |a_f \times s|$.

3.2. Coloring of Lunar Eclipses

Color is modeled as a function h of transformed eclipse phase t_f. Since ε_u and ε_p vary, we assign them fixed ranges for easier handling: $0 \le t_f \le 1/2$ for umbral distances and $1/2 \le t_f \le 1$ for penumbral distances. The fragment eclipse phase D_f is linearly transformed into these two ranges. h yields a multiplier per color channel used to modify the re-

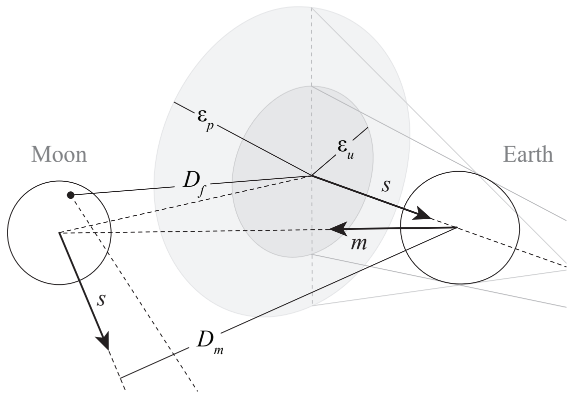


Figure 4: *Lunar eclipse model showing the moon at m, the eclipse phases D_m and D_f between the moon-sun line (dashed in s direction), the earth in the center, and the actual umbra and penumbra radii ε_u and ε_p.*

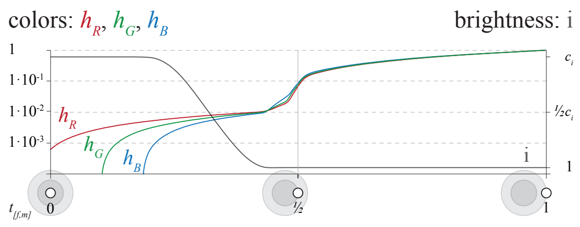


Figure 5: *Left: Multiplier h for lunar eclipses per color channel, shown on a logarithmic scale. Right: Brightness i on a linear scale between 1 and c_i (we used $c_i = 30$). Correlated to the t-axis, lunar eclipse phases are indicated.*

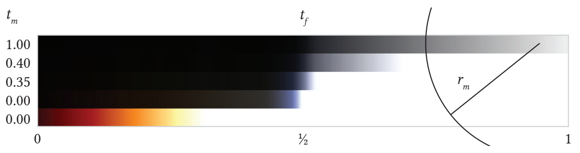


Figure 6: *Fragment colors over t_f applied at various eclipse phases t_m. r_m roughly indicates the moon radius for better sensation of scale. Bottom $t_m = 0$: colors increased tenfold.*

sulting moon fragment color as shown in Figure 5. It accounts for the following four distinctive phenomena:

1. A barely noticeable penumbra, indicated by a soft darkening of the moon towards the umbra.
2. A strongly noticeable, dark umbra, with a soft edge.
3. Strong reddish orange hue in the umbra, becoming darker towards its center. This is due to scattering in the earth's atmosphere, where shorter wavelength are more likely to be outscattered than longer ones. Thus, red light is bent towards the umbra center much stronger than blue light.
4. A soft bluish rim at the umbra edge, which reflects the remaining, less scattered light.

We specify h by a sum of various functions [Mül12], each fitting a phenomena to various photographs. Since h can become very complex, it is precomputed into a one dimensional map for look-ups per fragment. One could also specify this texture by other means, like physical simulations, to meet individual appearance notions and requirements.

3.2.1. Brightness during a Lunar Eclipse

Equivalent to h, i is a function of t_m to describe the brightness during the eclipse cycle. Instead of a specific fragment's eclipse phase D_f, this correlates to the eclipse phase D_m of the moon itself. A reasonable i is shown in Figure 5. Finally, we obtain each fragment color C_f by $C_f(t_f) = i(t_m)\,h(t_f)\,c_f$, with c_f as initial fragment color (Figure 6).

4. Rendering Stars

Rendering stars with random distribution in position and intensity is not plausible, because humans are strongly accustomed to the earthly night sky with its typical constellations. Using photographs or precomputed textures with correct star placement is also insufficient, since stars might appear either bulky and blurred, or small and wobbling during camera movement. Utilizing point-sprites is also not applicable. They are unaffected by camera distortion, which leads to star clustering in the center when using larger FoVs.

Similar to rendering the moon, point light sources are rendered with viewpoint aligned billboards that adapt to the actual output resolution and are prone to distortion. The technique also scales well with increasing number of pixel per inch of modern displays. We obtain actual positions of stars and planets and apply an individual adjusted point spread function for intensity (intensity PSF) [Mai09]. A simple glare (glare PSF) is added to overcome physical intensity constraints and provide a larger range of apparent brightness as common in HDRR. Star color and intensity is approximated based on their temperature and distance. This approach is also applicable for solar planets and satellites and can be adapted for observers within our solar system.

4.1. Modeling Stars

At most, 9500 bright stars and star clusters are perceivable by the naked eye under optimal conditions. These are modeled as points with precomputed position, color, and intensity based on data provided in the Yale Bright Star Catalogue [HW95]. They are passed to the GPU and rotated appropriately to date, time, and location in vertex stage. Here, color and intensity affecting outscattering (extinction caused outside the atmosphere is ignored) and scintillation are applied. Finally, one billboard per star is created and scaled in geometry stage, and finally rendered overlaying the PSFs.

4.1.1. Positioning Stars

For each star or star cluster one vertex is passed to the GPU. Their equatorial right ascension α and declination δ for the J2000.0 equinox, are transformed to Euclidean coordinates $p = (x, y, z)$. To avoid updating all vertex positions on every change in date, time, or location, all vertices are passed only once to the GPU. A single rotation matrix R_s [JDD*01] is required per change, to apply precession P, and convert their equatorial coordinates to horizontal ones. R_s is given by:

$$R_s = R_y(\delta - \pi/2)\, R_z(-LMST)\, P, \tag{7}$$

$$P = R_z(0.01118\,T)\, R_y(-0.00972\,T)\, R_z(0.01118\,T), \tag{8}$$

with T as time in Julian centuries, and $LMST$ the approximated local mean sidereal time. The final vertex position adjusted in vertex stage is pR_s. Unfortunately, this approach does not allow for individual annual proper motions.

4.2. Apparent Magnitude

The brightness of stars is measured on a logarithmic scale in apparent magnitude m: A star with 1 mag is about 100 times brighter than one with 6 mag. This leads to a relative brightness $2.512^{m_1-m_2}$ between two stars with apparent magnitudes m_1 and m_2. $\Delta_m(m) = 2.512^{m_a-m}$ is used for individual star brightness, with m_a as a control magnitude, e.g., representing the observers's current brightness sensitivity. Usually brighter stars also appear larger. However, even the faintest and smallest stars become to bulky on today's screens. We chose not to enlarge the intensity PSF, but use an additional glare PSF on an enlarged billboard, to provide a visual cue of higher intensity. This might be incorrect in photometric terms, but requires neither extra tone mapping nor extra blur passes for glare. Textured glare [SSZ*95] per star or dynamic temporal glare in a post-processing step [RIF*09] could be used to account for star streaks. With a minimum billboard radius of about $\sqrt{2}$ pixel, flickering due to aliasing is eliminated. In screen space, this radius q is obtained by the vertical field of view γ_v and the vertical output resolution res_v in pixel:

$$q = 2\sqrt{2}\tan(\frac{\gamma_v}{2})\, res_v^{-1}. \tag{9}$$

As intensity PSF, $T(l) = 2l^3 - 3l^2 + 1$ is used, with $l \in [0;1]$ as normalized distance to the billboard center. The brightness Δ_m needs to directly correlate to the PSF intensity, which is achieved by scaling with I_T:

$$I_T = V_T^{-1}\Delta_m(m)\, c_q\, q^{-2}, \tag{10}$$

with $c_q \approx 4\times 10^{-7}$, calibrated by comparisons with photos, for $m_a = 4$. Through q, I_T inversely correlates to γ_v, so that on decreasing FoV, fainter stars become more and more visible. Thinking of a PSF as a solid of revolution, its volume V can be interpreted as intensity. Disc-integration yields a volume of $V_T = 1.167$ for T. The calibrated intensity $I_T\, T$ is 1 for $m = m_a$.

For $m < m_a$, a star is lightened with glare of intensity $I_G = \Delta_m(m + (V_T - 1)) - 1$. The glare PSF is arbitrarily defined as $G(l) = \sqrt[64]{l}$, and the billboard radius by $k = \max(q, c_k\sqrt{I_G})$, with a resolution adjusting coefficient c_k. Note when using glares, T needs to be scaled by $k\,q^{-1}$.

4.3. Star Color

To procure star colors, measured B-V values based on the Johnson-Morgan-System [JM53] need to be converted into RGB space. Given a B-V value in mag, a stars temperature can be approximated [JDD*01, Ols98]. Chromaticity coordinates can be obtained by mapping the temperature to the Planckian locus by means of a polynomial fit [KCKH03]. These values are mapped to the CIE tristimulus, and finally converted into sRGB space [Ols98, Kry85].

4.3.1. Scattering

Light reaching an earthly observer at night, is affected by atmospheric scattering (with unnoticeable interspersion though). Because of that, we approximate the optical path

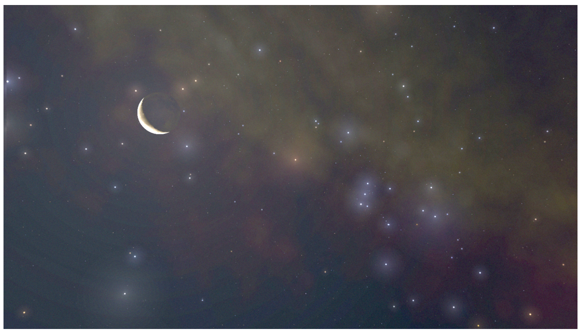


Figure 7: *Night sky with Orion's Belt in the right, and the moon in the left. Note the faint Milky Way and the moon (enlarged) with earthshine, masking the background.*

length Φ of perceived light rays, traveled through a simplified atmosphere [Buc95], relative to the length in zenith direction t_e. Given the mean earth radius $r_e = 6371$ km, the observer altitude h in km, and the light ray's angle to zenith θ, Φ can be expressed by the law of sines:

$$\Phi(\theta) = -\sin\left(\arcsin\left(\frac{r_e+h}{r_e+t_e}\sin\theta\right) - \theta\right)\frac{r_e+t_e}{\sin\theta}, \tag{11}$$

$$O(\theta) = c_r\,(1+\beta_r)\,(t_e-h)^{-1}\,\Phi(\theta), \tag{12}$$

Color and intensity are attenuated by O, with $c_r \approx 6$ and β_r as scattering coefficient for air molecules given by Rayleigh. In our model, it describes the wavelength dependency and we use $\beta_r = (0.16, 0.37, 0.91)$ for $h = 0$ [Buc95]. For t_e values of about 8 km for Rayleigh scattering are common. [NSTN93, PSS99, BN08]. If observer altitude is ignored, the true horizon fixed at $\theta = \pi/2$, simpler approximations [PSS99, YC09] would be sufficient.

4.3.2. Scintillations

Scintillations are fluctuations on a time-scale of milliseconds, causing variations in color and brightness of stars and other perceivable, outer-atmosphere objects. They are caused by atmospheric turbulence and therewith density and refraction differences. Although the appearance of planets, and the moon and sun surface are affected as well, we consider point lights solely.

Similar to scattering, scintillations strongly increase towards the horizon, thus we rely on the optical path length ratio Φ again. For each star we generate a random scintillation basis $n \in [0;1]$ per frame, to simulate fluctuations over time on the smallest available time scale. To distribute the number of simultaneous twinkles, n gets non linearly remapped to $[0;1]$ using an arbitrarily chosen $N = 0.02/n$. Lastly, each star's brightness is attenuated by the scintillation intensity $S = c_s\,\beta_r\,\Phi N$ where c_s is used for adjustment (e.g., $c_s \approx 20$).

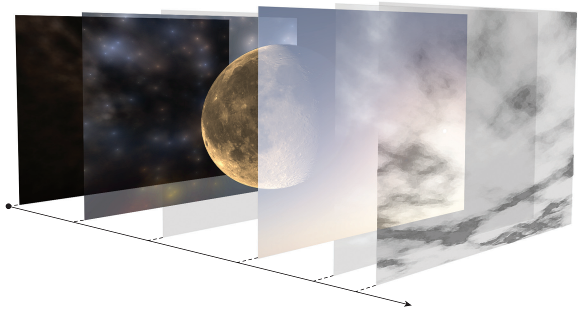


Figure 8: *Exemplary illustration of the composition sequence. Starting at the left: star map, bright stars, the moon, atmosphere with sun, and various cloud layers.*

4.4. Star Map

At clear nights with few artificial light, the faint background of our galaxy can be sensed (Fig. 7). A screen-aligned quad is used to render a textured cube, showing a generated Tycho Catalog Skymap (star map) in equatorial coordinates [BW09] . The star map resolution should be rather low, just indicating the Milky Way instead of further individual stars. Especially for high resolutions, bright stars should be removed though, to avoid start doubling caused by positional differences to individual rendered stars. Correct orientation by means of Equation 7 is applied to the texture lookup. Star map brightness is scaled by $c_s\, \Delta_m(m)\, \sqrt{q}^{-1}$, with c_s depending on output resolution and initial texture intensity. It should be adjusted, so that the Milky Way is hinted with default γ_v at $m_a = 4$. As for bright stars, scattering is applied, and brightness is FoV adaptive again. Scintillations however are inapt.

5. Composition

In order to render within a single-pass, all components have to be rendered in correct sequence, based on their correlated phenomena distance. Rendering starts with the opaque star map or, alternatively, a black background. Bright stars are blended into this, utilizing per pixel intensities in the alpha channel. The moon is opaque again, overlapping stars even when unlit by the sun as seen in Figure 7. Finally, the atmosphere is blended on top of these layers, so all stars and the moon are influenced by its color. Cloud layers can be overlaid afterwards, using appropriate blend modes. The individual layers are illustrated in Figure 8.

Intensity differences over several magnitudes within the day-night cycle, lead to zero visibility of stars and a strongly attenuated moon at day time. Scaling star intensities by $(1+(s_z+1.14)^{32})^{-\frac{1}{2}}$, where s_z is the sun's normalized, euclidean altitude, leads to a smooth transition between full visibility just before, and zero visibility just after sunrise (and vice versa for sunset). Likewise, for moon intensity, a factor of $0.5+(2+2\,(s_z+1.05)^{32})^{-\frac{1}{2}}$ is used.

The atmosphere model suggested by Bruneton and Neyret [BN08] features a good match to CIE Standard General Sky, accounts for earth shadowing, and supports single as well as multiple scattering. Note, that the precomputed textures for rendering the atmosphere, can be used to replace the scattering approximations introduced for moon and star rendering.

An array of precomputed 2D Perlin noise maps, projected onto spherical caps, form the basis for multiple, dynamic cloud layers. Naive scattering [Mül12], inspired by Dubé [Dub05], for lower cloud layers causes a 3D-ish appearance which increases the sky's overall credibility.

6. Results

A composition with clouds and moon at day is shown in Figure 9. A typical night shot in Figure 7. Even though, our results are very compelling, considering the low performance impact, there are few remarks:

- At night, a slight bluish tonemapping due to scotopic viewing should be applied.
- The Moon often feels lost at night. An additional glare, with its intensity linked to the moon's intensity (including variations due to phase and lunar eclipses), fixes this.

Finally, a rendering of a specific lunar eclipse is compared to a result synthesized by Yapo and Cutler [YC09], and two reference photos, in Figure 10.

6.1. Performance

Component	# Vertices	Time in μs
Star Map, $m_a = 6.0$	4	228
Bright Stars, $m_a = 6.0$	$4 \times 9\,129$	56
Moon, $c_d = 3.0$	4	12
Moon, $c_d = 100.0$	4	131
Atmosphere	4	581
Atmosphere with dithering	4	728

Table 1: *Enlisted are the average time differences per frame to an empty scene, measured over a minute. Each component is drawn to a screen-aligned or viewpoint oriented quad. The difference between star map and bright stars is due to discarded stars in geometry stage. Furthermore, the applied glare function strongly influences the performance, since it specifies the amount of fragments to be processed per star. System: Intel Core2 Duo E8400 at 3.0GHz, 8.0GB Memory, NVidia GeForce GTX 460 with 1.0GB memory.*

We use uncached astronomy calculations and not optimized code. Targeting 60 fps, the average performance impact of a cloudless day-night sky is less than 4% on our test system. Precomputation of all atmosphere textures took about 2.0s. Table 1 lists rendering times per phenomena.

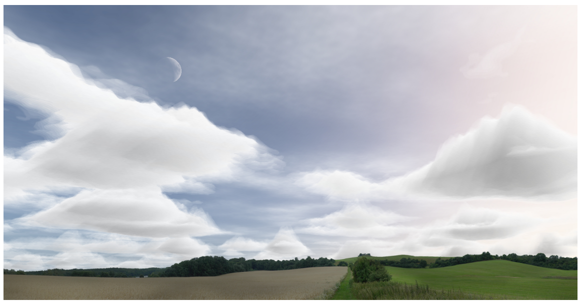


Figure 9: *Photographed landscape combined with a sky at day-time, rendered using our method. It contains two cloud layers at 1km (with scattering) and 7km, the moon with ten times its apparent diameter, and the sun in right direction.*

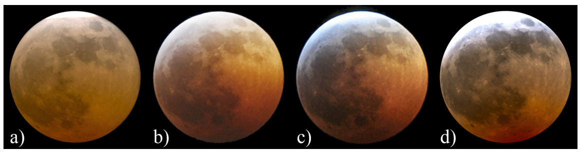


Figure 10: *A comparison of a rendering of the December 21, 2010 lunar eclipse viewed from New York about 7:40 UTC, between a) Yapo and Cutler [YC09], two photographs b) and c), and using the method presented in this paper d).*

7. Conclusion and Future Work

We have shown techniques for efficient, astronomically accurate real-time rendering of the moon and stars featuring a yet unprecedented degree of detail. A third technique was suggested, composing these night phenomena with existing day time techniques, attaining holistic day-night cycles within a single-pass. The methods provide astrophysical pleasing skies and are well suited for on the fly computations of backgrounds and cubemaps, often required for real-time reflections, global illumination, or various post-processing.

Among other issues, blending based on an apparent control magnitude or radiance, modeled for all individual phenomena would be most valuable. This however, asks for a uniform integration of all phenomena within a single model. Finally, we would like to address proper, astrophysical pleasing synthesis of solar eclipses.

References

[BN08] BRUNETON E., NEYRET F.: Precomputed atmospheric scattering. *Comput. Graph. Forum 27*, 4 (2008). 2, 6, 7

[Buc95] BUCHOLTZ A.: Rayleigh-scattering calculations for the terrestrial atmosphere. *Applied Optics 34* (1995). 6

[BW09] BRIDGMAN T., WRIGHT E.: The tycho catalog skymap - version 2.0. svs.gsfc.nasa.gov/goto?3572, 2009. 7

[Dub05] DUBÉ J.-F.: Realistic cloud rendering on modern gpus. In *Game Programming Gems 5*. 2005. 2, 7

[Hap66] HAPKE B.: An improved theoretical lunar photometric function. *Astronomical Journal 71* (1966). 3

[HKA05] HASAN M. M., KARIM M. S., AHMED E.: Generating and rendering procedural clouds in real time on programmable 3d graphics hardware. In *INMIC 2005* (2005), IEEE. 2

[HW95] HOFFLEIT D., WARREN JR. W. H.: Bright star catalogue, 5th revised. *VizieR Online Data Catalog 5050* (1995). 5

[JDD*01] JENSEN H. W., DURAND F., DORSEY J., STARK M. M., SHIRLEY P., PREMOŽE S.: A physically-based night sky model. In *SIGGRAPH 2001* (2001), ACM. 2, 3, 5, 6

[JM53] JOHNSON H. L., MORGAN W. W.: Fundamental stellar photometry for standards of spectral type on the revised system of the yerkes spectral atlas. *Astrophysical Journal 117* (1953). 6

[KCKH03] KIM Y.-S., CHO B.-H., KANG B.-S., HONG D.-I.: Color temperature conversion system and method using the same, 2003. 6

[Kry85] KRYSTEK M. P.: An algorithm to calculate correlated color temperature. *Color Research and Application 10* (1985). 6

[Mai09] MAIWALD C.: Hochwertiges rendern von sternen 2.0. zfx.info/viewtopic.php?f=11&t=8, 2009. 5

[Mee94] MEEUS J.: *Astronomische Algorithmen*. Barth, 1994. 2, 3

[MSK*10] MAGNOR M., SEN P., KNISS J., ANGEL E., WENGER S.: Progress in rendering and modeling for digital planetariums. In *Proc. of Eurographics 2010* (2010). 2

[Mül12] MÜLLER D.: osghimmel - osg lib featuring dynamic, immersive, textured or date-time and location based, procedural skies. osghimmel.googlecode.com, 2012. 5, 7

[NGN*00] NADEAU D. R., GENETTI J. D., NAPEAR S., PAILTHORPE B., EMMART C., WESSELAK E., DAVIDSON D.: Visualizing stars and emission nebulae, 2000. 2

[NSTN93] NISHITA T., SIRAI T., TADAMURA K., NAKAMAE E.: Display of the earth taking into account atmospheric scattering. In *SIGGRAPH 93* (1993), ACM. 6

[OHS01] OBERSCHELP W., HORNUNG A., SAMULOWITZ H.: Visualization of eclipses and planetary conjunction events. the interplay between model coherence, scaling and animation. *The Visual Computer 17*, 5 (2001). 2

[Ols98] OLSON T.: The colors of the stars. In *In IST/SID 6th Color Imaging Conf* (1998). 6

[PSS99] PREETHAM A. J., SHIRLEY P., SMITS B.: A practical analytic model for daylight. In *SIGGRAPH 99* (1999). 6

[REK*04] RILEY K., EBERT D. S., KRAUS M., TESSENDORF J., HANSEN C.: Efficient rendering of atmospheric phenomena. In *EGSR 04* (2004). 4

[RIF*09] RITSCHEL T., IHRKE M., FRISVAD J. R., COPPENS J., MYSZKOWSKI K., SEIDEL H.-P.: Temporal glare: Real-time dynamic simulation of the scattering in the human eye. In *Proceedings Eurographics 2009* (2009). 6

[RP05] RODEN T., PARBERRY I.: Clouds and stars: Efficient real-time procedural sky rendering using 3D hardware. In *Proc. of the 2005 ACM SIGCHI* (2005). 2

[SSZ*95] SPENCER G., SHIRLEY P., ZIMMERMAN K., GREENBERG D. P., INC T.: Physically-based glare effects for digital images. In *SIGGRAPH 95* (1995). 6

[vdH80] VAN DE HULST H. C.: *Multiple Light Scattering: Tables, Formulas, and Applications*. Academic Press, 1980. 3

[YC09] YAPO T. C., CUTLER B.: Rendering lunar eclipses. In *Proc. Graphics Interface* (2009). 2, 4, 6, 7, 8