Radiometry

In the course (and CG generally): study light as a particle, not a wave (simpler). One exception: polarisation (important for Fresnel reflections). Luckily, easily encapsulated in particle-based (geometric) models.
Basic properties of geometric optics: linearity and energy conservation

Basic Quantities:

Radiant Flux or Power (Φ): Energy flowing through a surface per unit time. Appropriate for the energy emitted by a light source. (J/s) or (W)
Irradiance (E): Area density of incoming flux. Appropriate for energy received by a surface. For an isotropic light source, at distance r, $E = \Phi/4\pi r^2$. (W/m²)
Intensity (I): Flux density per solid angle. Convenient for point light sources. $I = d\Phi/d\omega$. (W/sr)
Radiance (L): Radiant flux density per unit area per unit solid angle. $L = d^2\Phi/(dA \cos\theta\, d\omega)$. Radiance remains constant with distance (unlike irradiance). (W/m²sr)
Lambert's Law: Irradiance E is proportional to the cosine of the angle between light direction l and surface normal n. $E = \Phi\cos\theta/A$
Incident Radiance: $L_i(x, \omega)$, due to light arriving from a source
Exitant Radiance: $L_e(x, \omega)$ or $L_o(x, \omega)$, due to reflection from a surface
Almost always, exitant radiance is less than incident radiance (perfect reflectors are physically near-impossible)
Bidirectional Reflectance Distribution Function (BRDF): Formalises the reflection of light at a surface. (1/sr)
$f_r(x, \omega_i, \omega_e) = dL_e(x, \omega_e)/dE_i(x, \omega_i) = dL_e(x, \omega_e)/(L_i(x, \omega_i) \cos\theta\, d\omega_i)$

Camera Settings:

Shutter Speed: Time the sensor is exposed to light for (inverse relationship with ratio between actual and measured radiance).
Aperture: Radius of opening in the lens (small f-values represent larger radii, so squared relationship with actual radiance).
Gain: Digital signal boost (ISO 100 is the zero-point). Also measured in dB (actual radiance is proportional to $(10^{(1/20)})^{\text{decibels}}$). Accentuates the signal, but also accentuates noise.
Neutral Density (ND) Filters: Blocking light with filters. Effectiveness given in terms of in "density" 0.1-4.0 (log10 scale). 0.3, 0.6 and 0.9 correspond to 1, 2 and 3 *stops* respectively.
Stops: Ratio of actual radiance to measured radiance, log2. i.e. an image taken at x stops will appear 2^x times darker than an image at 0 stops.
Taking HDR Images: As opposed to single-exposure photography, we can take multiple exposures of the same scene (an "exposure stack") and fuse them into a single image. To generate the exposure stack, we can vary shutter speed or aperture; varying shutter speed can have problems with motion, while varying apertures can have vignetting/glare issues due to using different parts of the lens as aperture changes size. Wider apertures also have very shallow depth of field, which might be undesirable.
Gamma Correction: Raising pixel values (normalised to the range 0-1) to the power of 1/2.2 to correct for non-linear response of the display. Hold-over from CRTs (LCDs just replicate this curve to stay consistent).
Image Acquisition Pipeline: Scene radiance L passes through the lens onto the sensor, which measures irradiance E. This is then integrated over time based on the shutter speed to produce a final exposure value.
Modern cameras aim for linear response. On cameras which do not achieve this, we can recover the response curve by analysing how measured pixel brightnesses change across an exposure stack.
Image Formation Model: $Z_{ij} = f(E_i \Delta t_j)$ where Z_{ij} is the measured camera pixel, f is the camera response and $E_i \Delta t_j$ is the product of irradiance and exposure. Need to account for f and Δt_j when assembling HDR images. We also need to weight values in the middle of the range higher than those at the edges to account for clipping.
HDR Image Formats: Portable FloatMap (.pfm) - 12 bytes per pixel (4 per channel) with plaintext header to store metadata, .hdr - entire HDR pixel in 4 bytes (8 bits for each R, G and B mantissa plus one shared 8-bit exponent), OpenEXR - 16-bit floating point per-channel plus .jpg-like wavelet compression.
Tone Mapping: Displaying HDR images on an actual display (i.e. converting back to LDR). Linearly mapping doesn't work well most of the time; better - histogram equalisation.
Global vs Local: Global tone mapping effectively compresses the global dynamic range. This reduces local contrast. Local mapping combines with global mapping, aiming to preserve edges/detail, mimicking human visual perception.
Inverse Tone Mapping: Going from LDR to HDR. Application of *deep learning*. Can use masking to apply the NN only to crushed/blown-out parts of the image (limiting the impact of "hallucinations").

Image Based Lighting with HDR Imaging

RT alone isn't enough for real lighting. With HDR Imaging, we capture the full surrounding env (a 2D cross-section of the spherical light probe) around us as an image based rep of a 360 degree covering of the light source. Because of the large light dyn. range, IBL needs multiple exposure photography. We usually capture the 360 deg env map via photographing a light probe (e.g a mirror ball). The photo captures each pixel in the panoramic image, and those can be used in the rendering simulation. More images = better quality reconst. Not used for real time rendering, as there's *millions* of pixel light sources - Spherical Harmonic Lighting is instead.
Even the highest quality mirrors are only 90% reflective. So you must calibrate the reflectivity of the probe: 1) Take a known diffuse reflectance 2) Take the mirror ball and place them nearby 3) Take a photo 4) Compute the intensity of the diffuse object's reflectance to the camera (e.g: 0.58), and the reflectance of diffuse object's reflectance in the mirrorball (e.g: 0.34). 0.34/0.58 = 59%; the mirror ball thus has 59% reflectance.

Constructing a High Quality Light probe:

- You can't simply just take a single photo of a mirrorball - this is not sufficient for a 360 degree panorama. Theoretically one mirror ball photo does fully sample all directions (as the surface normal angle at the poles is 90°; and due to the reflectance formula this means that light is 180° away)
- But practically, the *area density of measurement* means that one photo will have amazing sampling of the frontal hemisphere, and poor of the back
- More than 2/3rds of the area of the mirror ball will reflect the frontal hemisphere, the back hemisphere will be squished around the edges.
- We use two orthogonal views and merge them into one full 360° panorama
- Two orth? two ppp from front and back: has correspondence (overlapping info); the cameraman is not in the image in one view at high res so you can copy pixels from there (to remove them) & we get good pixels everywhere
- Removing the cameraman isn't so important technically but rather aesthetically (as their pixels will be quite dark; won't affect relighting) because the light probe wants to be used as a background for rendering.

Using the High Quality Light probe:

- You model the scene with geometry. The camera traces out a ray which intersects with the geometry. Depending on the material property of the material intersected, you ray trace out in different directions to sample the illumination, because everywhere contains a light source.
- In the late 90s, we used cube-maps rather than spherical maps as only those were supported in the GPU.
- We rasterize every single pixel by tracing a ray:
  - CASE 1: If the ray does out into the scene and doesn't intersect with any geometry, then it keeps on going and the camera direction is used to directly index a pixel in the env map which is used as the colour
  - This lets us rasterise the background & the foreground at same time
  - CASE 2: The ray intersects with the geometry (the sphere) - in this case we reflect the ray off the sphere normal and where it ends up informs us of the pixels to index as our colour: Reflection: R = 2(N.L)N-L
    - N = sphere surface normal, L = Ray direction.
    - We also have an R'' representing the refraction into the sphere; so we shoot two rays both of which do an indexing to inform the colour.
    - In glass objects, the background env map tends to be inverted as the refrac rather than reflect is the dominant indexing quantity (Fresnel)
  - CASE 3: Glossy materials. They have both a diffuse and specular reflec. Expensive to render as we shoot rays in multiple directions to sample, highlight them & combine them. We see soft shadows on the table if some rays are occluded / don't make it back to the camera (soft shadows - many light sources).
- Remember to use Gamma to naturalise the lighting.
- We often project our mirror balls down onto a lat-long environment map
- +x: top row, -ve Y: bottom row; the left-right extremes are -Z (behind mirror ball). Centre is positive Z, -X is left between -Z and Z+; X+ is right between +Z and -Z. ie: it's all how the world light would map onto the ball.

Acquiring Light Probes: We can acquire light probes in multiple other ways: 1) Panoramic (fisheye) lenses
2) High-Res photos of many angles and stitch them together into a panorama
3) Panoramic lens on robotic mounts. Very high res, full sphere in one scan, good dyn range; expensive & takes a while.
Usually we use 5-6 fisheye photos; higher quality than mirror ball; don't need thousands of high res pics to stitch.

However, the intensity capture capabilities in fisheye photos fall off when far from the center - at aperture f/16 this happens past 1000 pixels, at f/8 quite drastic past 500 pix. To combat this you need to learn the falloff curve and invert it for every pixel to properly receive the intensity.

Direct Sunlight:

- Causes light probes to become inaccurate as the light is clipped to 1. Multiple exposures won't help, it's just that bright. ND filters help but change the colour tone and can bump the lens (ruining the capture).
- Solution: Do reasonable HDR imaging - The sun (and a few pixels) remains saturated. We approx the clipped light energy of the sun as a light source and place it into the scene as an extra light source (by adding it back in)
- We use a diffuse sphere captures the unclipped sun's brightness. By comparing a rendered sphere (using an estimated sun intensity) to the image of the diffuse sphere, we determine a scaling factor (alpha) to restore the sun's correct energy in the clipped probe.
- We can measure natural illumination in one shot! Mirror ball for sky, diffuse ball for unsaturated sun, and black ball for accurate sun position.

UNet can be used to extrapolate the dynamic range of an LDR light probe to an HDR light probe.

Relighting

Lighting: For synthetic objects, Relighting: For "real" objects
Can use a light stage to generate a 4D reflectance field (lat/long of incident light plus x/y pixel coordinates into images of the subject).
From an environment map and a reflectance field, relighting is very simple mathematically: merely a dot product between the two. When using latitude-longitude representation, we need to remember to normalise - * sin(lat).
Convention: X = cosφ sinθ, Y = cosθ, Z = -sinφ cosθ where θ is lat and φ is long. If we also account for the position of the light source, we get an 8D generalised reflectance field or a BSRDF (Bidirectional Scattering Reflectance Distribution Function) but this is usually too many dimensions to be practical. If the scene is small w.r.t. the size of the light stage, the positional dependence on the incoming illumination can be ignored. If we also fix the camera position, the angular variation of the outgoing reflectance is removed. This results in a 4D Reflectance Field.
Relighting relies on the linearity of light transport: the radiance of the final scene can be linearly decomposed into the contribution from each source.
Light Stage Designs:
Single Light Source: A single light source moved around the scene. Inexpensive and supports any resolution but slow and hard to automate.
Light Arc: A single arc of lights that stays fixed in place while the scene rotates. High resolution and faster to capture, but restricted subject size.
Light Dome: A dome of lights surrounding the scene. Very fast and fully automatable, but expensive, complex and limited resolution.
Free-form: We can get away with non-uniform sampling of the reflectance field by using computer vision approaches (e.g. add some reference object to a scene, like a diffuse sphere) to calibrate for the incident light dir. We can use angular Voronoi diagram to partition samples to different angular cells.
Reflectance Function: How a single pixel varies across incoming light directions in a reflectance field.
Nyquist-Shannon Theorem: If a function f contains no frequencies higher than B, then it is completely determined by its ordinates at a series of points spaced 1/2B apart. i.e. We need to sample at twice the highest frequency. Applied to light stages: some materials (specular) might have much higher frequency reflectance functions than others, which requires much higher resolution sampling to measure accurately.
Neural Networks for Relighting: NNs can attempt to interpolate specular reflectance functions from a small number of samples, seemingly overcoming the limit imposed by Nyquist theory.
Lighting Reproduction: Directly light the subject with the illumination of a virtual background (e.g. use a light dome with RGB lights). Need to isolate subject from the background, which can be done with infrared illumination/camera. RGB LED illumination can also have gaps in the spectrum compared to the original scene (LED spectra are very "spiky"). Having additional LEDs of different colours (e.g. cyan, amber) helps.

Reciprocity and Light Fields

Recall the 4D Reflectance Field: composed of position: (x,y) and the angle the light is coming from: with regards to the planes (xy, yz). However, due to the discreteness of having n number of lights, light fields have problems:

- For a glass object with very sharp specularities, transparities and high frequency caustics, a regular discretized light stage doesn't work very well (there's black light gaps - aliasing), even with a very high density of lights.
- Shadows don't move continuously with the motion of the lights

Helmholtz Reciprocity: for BRDFs, swapping the light and camera positions results in the same reflected intensity. Most objects have this characteristic. This can be generalised to the entire light transport between the source, scene and camera: swap camera position and light position and the same magnitude of light transport happens. This principle inspired a new light stage design.
Dual Light Stage:

- We have a high speed camera with a fisheye lens above the scene, a hollow diffuse gray closed sphere surrounding the object and a laser and galvanometer (at the same point) pointing at the object through a hole.
- This configuration continuous dome prevents aliasing by allowing the camera to record a continually reflectance field (barring the gap for the laser beam and the gap for the camera). This works because in this setup the camera doesn't record the object and how it responds to light but rather it looks at the pixels it sees on the dome wall from the laser obj refl.
- Diffuse sphere because it lets the light scatter in all directions
- We move the laser very finely to rasterize each pixel by pinging the object with the galvanometer.
- Need high speed camera as we need MANY pictures. Each ping of the beam = 1 pixel, Even for a small 200x200 image, that's 40000 pictures.
- in the images produced by Dual Light Stages, there's a lot of grey bg near the high frequency streaks. This is the bounce light, as the diffuse wall is enclosed (and as a result there's a lot of scattering). We can measure the average ambient grey level over the images and subtract it to solve this.
- Our light at point u,v represents the reflectance function for u,v. Once we get the full pixel reflectance function, we reparametrize the reflect function (OR the light probe) to be in the same coordinate system with the same parameterization. Then we can just dot product it per pixel with the desired lighting (e.g: Grace Cathedral EM Map) to do the full relighting.

Dual Photography:

- Primal Configuration: With a camera capturing the scene at viewpoint c' and a projector illuminating from p', the captured image is: c' = Tp' where T is the light transport matrix. By Helmholtz reciprocity, we can compute the dual image (what the projector would see if it were a camera) using: $p' = T^T c'$. This allows viewpoint swapping and virtual relighting without moving the camera or projector.
- You can find the Light Transport Matrix one column at a time illuminating one pixel using the projector at a time (with a long exposure photo) - remember that the LTM T[i,j] states how much light from projector pixel i reaches j (e.g: for each point how much light from all others)
- If you did this then you can create a synthetic image from the point of view of the projector, lit by a virtual projector that illuminates the scene, but all from the point of the camera! (with the transpose mentioned above)
- To avoid needing so many big long exposure photos, we can replace the camera with a photodiode (a one pixel camera that measures avg intensity)
  - *Adaptive Measurements*: First you illuminate the entire image (level 1) and measure. Then sequentially subdivide into 4 quadrants, and measure the light from each quadrant sequentially (4 more photos). We check the to see if any part of the image was illuminated by more than one quadrant from the four photographs.
    - If yes then those two quadrants are in conflict, and their subdivision measurements can only be done sequentially
    - If not then they're not in conflict and we can measure them in parallel
    - We continue this until we have figured out all of the pixels that can be recorded in parallel, and all of the ones that have to be done separately. While we still have to go all the way down to each pixel we ideally manage to learn which pixels must be measured sequentially in just log n steps.
    - Then we can measure all of the parallelisable ones at once, and then the sequential ones individually.
- Applications: reading hidden images (like cards) from their reflections.W
  - We can also extend a 4D reflectance field to a 6D field by using an array of mirrors and looking at all the mirrors a high resolution camera. These mirrors can be pointed to the scene lit by the projector. If we then invert the LTM, it's as if the camera was viewing the scene lit by several light sources (or rather projectors). We've extended the 4D reflectance field to a 6D as there's now the 2D location of the virtual light source / camera

Light Fields: L(u, v, s, t)

- (u,v) = coords of point on sensor, (s,t) = coords of point the light ray is from
- A light field describes the total amount of light traveling in every direction at every point in a space. Instead of just capturing a 2D image from a single viewpoint, a light field captures how light rays vary across different angles and positions. Encodes both spatial and angular information about a scene.
- If we think of light instead as a quantity travelling in free space between two plains then we can reduce it to 4 dimensions.
- Radiance required 5Ds: in 3D is function of position (3D) and direction (2D). In the general case it is a 5D plenoptic function on a surface.
- An image is a 2D slice of a light field

Capturing the Light Field:

- We can't reproduce reality as there's a limit to the focal and viewing plane.
- We place cameras at multiple locations on the viewing plane and they all record rays emerging from the focal plane (which has the scene we care abt)
- This measures the light field.
- In the original paper they used a camera on a plane that can rotate up, left right and down (pitch, yaw) and can move a little bit
- More modern setups: Stanford Camera Array (100 cameras in a 10x10 matrix). It records a 4D image because it has 2 dimensions of 3 dimensions
- Applications: looking through foliage - to see through cover with many cams
  - We can synthetical refocus on the,

Within cameras we can embed a camera array ahead of the first lens but between the outer lens. This allows us to take a picture then make synthetic images afterwards with a changed focal plane (refocusing the camera)

  - This is done by resampling all of the rays to construct rays that are focused at different depths.
  - We have a main sensor - e.g 4000 x 4000 res. Then we have a lens-let array e.g: 200 x 200 (effective res) which sets the effective resolution of the virtual image we can make. The lens let arrays sample 20 by 20 rays in different angles. Trades off of resolution for synthetic refocusing.

By looking at the whole image and a matrix of images you're looking at the (s,t) plane from different points on the (u,v) plane.
Reflectance Functions:

- The reflectance field representation and reflectance function representation are duals of each other.
- From each image we have (views of (s, t) from the (u, v) plane), we can extract how light is reflected from all incoming directions at each surface point, allowing us to reconstruct the appearance of the scene from a particular viewpoint under varying illumination.
- Looking at individual pixels of the reflectance function we can see the diffuse, specular reflections, subsurface scattering, interreflection, and translucency.

Light Stage 2 - the object sat inside as the light stage gantry rotates
Light Stage 5 - geodesic discretized sphere with lights
Light Stage 6 - combined both light field capture and reflectance field capture; a very large light stage 6m in diameter with LEDs everywhere for full body movement capture. They also have a backscreen for green lighting (to cutout the subject) and turntable rotation.
Using three camera sand the rotation they're able to capture all views, and capture the lightfield so that they can synthetically visualise any different viewpoint that exists. You do this by creating correspondence between views e.g: with optical flow. Optical flow fails if one image has something occluded that the other doesn't.

- If you have the views from the camera at the equator and the neighbouring cameras in the vertical / horizontal you interpolate to synthesize views b/w them. This isn't artefact free, it can fail, but does usually make good results.

There's also lightfields with cameras going outwards that rotate. We capture a 4D lightfield (a series of panoramas) rather than a 2D panorama. These fields are used to power VR (as movement needs affect, & L/R eye need diff views). Good for static scenes, but for dynamic scenes we need a camera array that captures rays coming from multiple directions simultaneously.
NERF: Neural Radiance Fields help us view the lightfields from novel viewpoints. Classically we'd use optical flow but that has problems as we mentioned. i.e: we're trying to reason about 3D parallax with 2 dimensions (errors). Modern approach: NNs for NERF from a FCN which think about the scenes as volumetric 3D quantities. Input: (x,y,z) of the voxel we want to learn, and (theta, phi) of the camera angle. Output: (RGB, alpha). If you have enough images of the scene from many viewpoints, the model performs very well to interpolate the vowels of the scene. We can then render the produced volume with raycasting which gives us great results (no artifacts)

Gaussian Splatting:

- Fit 3D Gaussians with mean and variance to points in 3D space reconstructed with structure-from-motion (SfM) (multiview capture).
- Fast training to fit anisotropic gaussians using differentiable optimization.
- Real-time rendering with rasterization (splatting) instead of 3D volume rendering employed by NeRF.
- Achieves higher quality than NeRFs on thin/high-frequency features.
- Pipeline: take points (with SFM) -> point cloud -> init gaussians -> given all the images, fit gaussians (backprop) -> render with splatting

Radiometry, and Fresnel Reflectance

Remember - we model light as a particle with polarization as the exception, and remember the properties (section 1). Assumptions 1) the electromagnetic field is unpolarized, 2) no fluorescence (when light is absorbed and re-emitted immediately); wavelength independence) 3) steady state; no phosphorescence (when light is absorbed and re-emitted over time). We also do not model wave diffraction or interference.
Recall *Lambert's Law* (section 1): E = dΦ/dA; hence E1 = Φ/A (along the surface normal), E2 = Φcosθ/A (general case).
Radiometric Integrals: Irradiance at surfaces is computed over a hemisphere of directions: $\int_\Omega L_i(x,\omega) \cos\theta\, d\omega$.

- We take the integral over Ω (the set of all directions) representing the hemisphere of incoming directions above the surface at a given point x.
- $L_i(x,\omega)$ is the incoming radiance at point x from direction ω.
- θ is the angle between the incoming light direction ω and the surface normal at x. cosθ does foreshortening - light arriving at a steep angle spreads over a larger area and is less effective at illuminating the surface.
- If this was not a solid surface, but a smoke particle, or fire, then the domain of integration would span the whole sphere instead of hemisphere

Spherical Coordinates: A 3D cartesian direction vector (x,y,z) is related to 2D spherical coordinates (θ, ϕ) by x = sinθcosϕ, y=sinθsinϕ, z = cosθ
Computing the Hemispherical Integral in Practice: We compute:
$E(x, n) = \int_{[0 \text{ to } 2\pi]} \int_{[0 \text{ to } \pi/2]} L_i(x, \theta, \phi) \cos\theta \sin\theta\, d\theta\, d\phi$.

- θ is as before and ϕ is the azimuthal angle (rotation around the sf normal)
- a full spherical integral would have the inner integral be from 0 to Π
- cosθ is Lambert's Law.
- sinθ accounts for the fact that we're doing non uniform sampling - we have way more samples at the poles than the equator. This rescales it to counter

Area Integrals: We often have to swap between Area and Angle Integrals. Well, we can do so with the following equality between the two concepts:
$E(x, n) = \int_\Omega L\cos\theta_i d\omega = \int_A L\cos\theta_i \cos\theta_o dA / r^2$.

- A: area of the light-emitting surface (e.g., a patch of an area light source).
- L: as before, $\cos\theta_i$: Lambert's law
- cosθo: The cosine of the angle between the outgoing light direction and the normal of the receiving surface. r²: inverse square law.

This concept improves efficiency if we have just one area light source - instead of taking millions of samples in the surface upper hemisphere, we can take a thousand samples on the area light source!
Bidirectional Reflectance Distribution Function:
Ratio of reflec radiance to incident irradiance: $f(x, w_r, w_i) = dL_r(x, w_r)/dE_i(x, w_i)$

- Units: inverse steradian (1/sr). They have 2 important properties:
  - Helmholtz Reciprocity: $f_r(x, w_r, w_i) = f_r(x, w_i, w_r)$
  - Energy Conservation: $\int_\Omega f_r(x, w_r, w_i) \cos\theta_i\, d\omega_i \leq 1$ for all w_r in Ω
    - i.e: the integral over all incoming directions for that fixed viewing direction * the cosine modulated angle is less than or equal to 1.

The Rendering Equation: $L_r(x, w_r) = \int_\Omega f_r(x, w_r, w_i) L_i(x, \omega_i) \cos\theta_i\, d\omega_i$
Snell's Law:

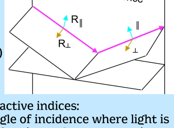
- Perfect specular reflection: $\theta_i = \theta_r$; the light reflects off (for material the light cannot pass into).
- Specular transmission (which depends on the refraction indexes & happens for dielectric materials): $\eta_i \sin\theta_i = \eta_t \sin\theta_t$
  - η_i is refractive index of the outside material, η_t is refractive index of material light enters, θ_i is angle the ray makes with the surface normal, and θ_t is angle between the inverted surface normal and the bent ray.

Dispersion: Refraction indices are wavelength dependent! i.e: light disperses when refraction happens because the different waves are bent in different ways (and this causes rainbows).
Polarization: Light is a transverse EM wave.

- It oscillates in a vertical direction - the electric field, and a horizontal direction (perpendicular to the electric field) - the magnetic field.
- Each photon has its own electric field going in a random direction. If you have lots of light coming at you, you have many electric fields coming to you. The effect of this is unpolarized light.
  - When you have a polarizing filter (sunglasses or on a light stage) then the light becomes *linearly polarized* as only light oscillating in the right direction can pass through

Fresnel Reflectance:

- Reflection from a surface is view dependent - i.e. both reflection and refraction, but amount of reflection depends on the angle of incidence.
  - Normal incidence: low specular reflection
  - At oblique (high/sharp) angles of incidence: strong specular reflection
- *Fresnel equations:*
  - For dielectric materials (insulators), reflection and transmission:



    - $R_\parallel = (\eta_t \cos\theta_i - \eta_i \cos\theta_t)^2 / (\eta_t \cos\theta_i + \eta_i \cos\theta_t)^2$ (parallel polarised reflectance)
    - $R_\perp = (\eta_i\cos\theta_i - \eta_t \cos\theta_t)^2 / (\eta_i\cos\theta_i + \eta_t \cos\theta_t)^2$ (perp. polarised reflectance)
    - Unpolarized reflectance = $F_r = ½(R_\parallel + R_\perp)$
    - Transmittance = $T_r = 1 - F_r$
    - θ_i = angle of incidence,
    - θ_t = angle of refraction
    - When entering objects with higher refractive indices:
      - *Brewster's Angle:* $\tan(\theta_B) = (n_2/n_1)$: angle of incidence where light is perfectly transmitted with no reflection. (n_2 = what you enter)
        - Even though the incident light was unpolarized, it becomes polarized now and perpendicular to the angle of reflection.
      - At surface normal angle, 0, only about 10% of light reflect/refrac.
      - At the grazing angle (90), 100% of light does.
      - Between this, $R_\perp$ increases slowly at greater rates; $R_\parallel$ dips and rises
    - When entering objects with lower refractive indices:
      - *Critical Angle:* $\sin(\theta_C) = (n_2/n_1)$: angle from which $R_\parallel = R_\perp = 1$
    - *Schlick's Approximation*: Tries to approximate F_r as: $F_r(\cos\theta) = R_0 + (1 - R_0)(1 - \cos\theta)^5$, where R_0 = the reflectance at normal incidence, and cosθ = h . v = h . l; cosθ is the angle between the half vector and the light source (or half vector - the vector between the incidence and reflection ray; not always the sf norm, and view vector).
      - Pretty accurate, and don't need to know refractive indices
  - For conductors (metals), we only have reflection and some absorption k:
    - $R_\parallel = ((\eta^2 + k^2) \cos^2\theta_i - 2\eta \cos\theta_i + 1) / ((\eta^2 + k^2) \cos^2\theta_i + 2\eta \cos\theta_i + 1)$
    - $R_\perp = ((\eta^2 + k^2) - 2\eta \cos\theta_i + \cos^2\theta_i) / ((\eta^2 + k^2) + 2\eta \cos\theta_i + \cos^2\theta_i)$
    - Sadly, for metals, there's little data for values of η, k. We approximate one by fixing the other. ie, if we set k = 0, and $\cos\theta_i = 1$ (normal incidence) then the equations simplify a lot. If we invert them we get $\eta = (1+\sqrt{R_0})/(1-\sqrt{R_0})$, so if we know the reflectance at normal incidence we can learn what η is.

Bidirectional Reflecance Functions (BRDFs)

Glossiness is a measure of how much the material spreads light around the reflection direction (broad vs shiny). *Retro-reflectivity*: light is preferentially reflected back in the direction it came from.
Phenomenological BRDF Models describe surfaces' qualitative behaviour

- e.g: matte, glossy or plastic, roughness
  a. *Lambertian Diffuse Reflection*: light is equally reflected in all directions: $f_r(\omega_r, \omega_i) = \rho_d / \pi$ is the diffuse reflection coefficient [0,1]. $\pi = \int_\Omega \cos\theta\, d\omega$ is the normalization constant.
  b. *Phong Model:* The BRDF is the sum of the diffuse and specular BRDFs: (ω_o is the dir vec between observed point & view; ω_i b/w light & point) $f_r(\omega_o, \omega_i) = \rho_d / \pi + \rho_s (\omega_r \cdot \omega_o)^s / (n \cdot \omega_i) = \rho_d / \pi + \rho_s (\cos\theta)^s / (n \cdot \omega_i)$. ρ_s is the specular reflection coefficient [0,1], S controls the specular lobe width
  c. *Blinn-Phong Model:* Variation of Phong, more accurate to reality: $f_r(\omega_o, \omega_i) = \rho_d / \pi + \rho_s (n \cdot \omega_h)^s / (n \cdot \omega_i)$; ω_h is the half angle b/w. +Efcnt
  d. *Lafortune Generalized Cosine Lobe*: $f_r(\omega_r, \omega_i) = \rho_d / \pi + \Sigma_j [C_{x,j} (\omega_{i.x} \cdot \omega_{r.x}) + C_{y,j} (\omega_{i.y} \cdot \omega_{r.y}) + C_{z,j} (\omega_{i.z} \cdot \omega_{r.z})]^{(s, j)}$. Suitable for off-specularity (if peak brightness isn't quite at the reflection direction), retro-reflection, anisotropy. Well suited for measured data! Not physics based, just an arb. function with enough DOF to fit the ingoing data.
  e. *Ward Anisotropic Model:* $f_r(\omega_i, \omega_r) = \rho_d /\pi + \rho_s*(1/\sqrt{(\cos\theta_i \cos\theta_r)})*(\exp[-\tan^2\delta(\cos^2\phi/\alpha_x^2 + \sin^2\phi/\alpha_y^2)] / (4\pi\alpha_x\alpha_y))$. Like before we have a diffuse BRDF and a specular coefficient, but replacing the Cosine Lobe model with an exponential model (the trig functions are for reciprocity, the bottom is for normalization). We have two params of anisotropy: δ (the angle between sf normal and half vector), and ϕ - the projection of ω_h onto the y plane, as for anisotropic materials the surface reflectance depends on the direction you look at it from The elliptic Gaussians α_x and α_y control the std dev in x & y.
  f. *Ashikhmin-Shirley Phong Model:* Takes the og Phonh model and makes it anisotropic and adds Fresnel Reflectance using Schlick: $f_r(\omega_i, \omega_r) = (\sqrt{(n_u + 1)(n_v + 1)} / 8\pi) * ((n \cdot \omega_h)^s/(\omega_i \cdot \omega_h)\max((n \cdot \omega_i), (n \cdot \omega_r)) * F_r((\omega_i \cdot \omega_h))$, where $F_r((\omega_i \cdot \omega_h))$ is the fresnel reflectance term

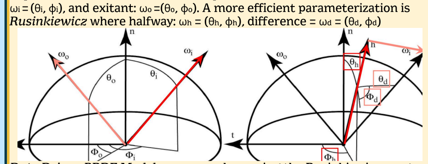
Physically Based BRDF Models: Based on the low level geometric structure of surfaces.
Microfacet Models: rough surfaces are modelled as a collection of microfacets. Each face is a perfect specular reflector (has sf norm). Smooth surface; not much bend in norm, rough = high bend. We describe the face distribution statistically. With V-shaped grooves, geometric effects start to take effect in more oblique angles. This causes 1) masking: light reaches a groove but another groove blocks the reflection from being scene 2) shadowing: the camera can see the groove, but light is blocked from reaching it by another 3) interreflections: the camera doesn't see the direct reflection but the interreflection from another groove bouncing into the camera. These grooves are so small we statistically average all 3 effects.
Torrance-Sparrow Model: $f_r(\omega_r, \omega_i) = D(\omega_h)G(\omega_r, \omega_i)F_r(\omega_h)/4(n \cdot \omega_i)(n \cdot \omega_r)$ - for rough surfaces. D term describes distribution of facets, G describes 3 efx, F = fresn; can be Maxwell Equations (the $R_\parallel$ etc.) or the Schlick Approx.

- *Beckman's Distribution:* $D(\omega_h) = \exp\text{-}[\tan\delta/m]^2 / \pi m^2 \cos^4 \delta$. δ is as before, m is the RMS slope of microfacets; low for rough mats, high for shiny. $G(\omega_r, \omega_i) = \min\{1, 2 (n \cdot \omega_h) (n \cdot \omega_r)/(\omega_r \cdot \omega_h), 2(n \cdot \omega_h)(n \cdot \omega_i)/(\omega_r \cdot \omega_h)\}$.
- *Blinn Microfacet Distribution*: $D(\omega_h)$ can have a cosine lobe: $(s+2)/2\pi * (n \cdot \omega_h)^s$. 1st term is normalizing
- *GGX Microfacet Distribution:* $D(\omega_h) = a^2 / \pi\cos^4\delta(a^2 + \tan^2\delta)^2$. δ as before, a = width of distribution. Better fit to measured data, with a longer tail and sharper peak than. More accurate for real surfaces.

Poulin-Fournier Model: We instead have cylindrical grooves. d & h control anisotropy. Suitable for fabrics such as satin and velvet.

- For satin: $p(h) = c * \exp(- \tan^2\theta(\cos^2 \phi/\sigma_x^2 + \sin^2 \phi/\sigma_y^2))$
- For velvet: $p(h) = c * \exp(- \cot^2 \theta/\sigma^2)$

Reparameterization: The standard parameterization has incident: $\omega_i = (\theta_i, \phi_i)$, and exitant: $\omega_o = (\theta_o, \phi_o)$. A more efficient parameterization is *Rusinkiewicz* where halfway: $\omega_h = (\theta_h, \phi_h)$, difference = $\omega_d = (\theta_d, \phi_d)$



Data Driven BRDF Models: more modern using the Rusinkiewicz prmtrs. $\rho_M(\theta_h, \theta_d, \phi_d) = \rho_d + \rho_s (D(\theta_h) F(\theta_d) G(\theta_i) G(\theta_o) / (\cos \theta_i \cos \theta_o))$.
This is the product of 3 separate 1D factors: D, F and G. This allows for massive compression with very high acc. 3*90 factors + 2 albedo param
Neural BRDFs: with h, d as input, we can approximate $F_r(h, d)$ with an MLP. We use ReLU activ, and exp output. Size: 6*21*21*3 weights. This is very accurate.

Scattering in Skin and Hair

To measure facial subsurface scattering, can use an LED probe. Faster and more accurate measurements can be made using a light stage with polarised lights - both specular and single-bounce scattered light retains polarisation while multiple scattering depolarises light. Specifically, can take both parallel and cross polarisation photos (where camera and light polarisers are in same and opposite orientation respectively). Specular and single-bounce scatter can be separated by first fitting specular as best as possible, and then fitting single-bounce scatter to what remains. Can also distinguish "shallow" and "deep" multiple scattering in skin by using a similar technique to "direct-indirect separation" - a high-frequency (e.g. sinusoidal) light and it's complement is shone on the subject. In direct-indirect separation, the measured reflectance in the areas not directly lit can be assumed to be a good estimate of half the full indirect lighting in the scene, and similarly, the not-directly-lit skin can be assumed to be lit entirely through "deep" scattering.
Scattering translucency can be fitted by blocking small circles of light and aiming to match the appearance with a Monte Carlo rendering.
Spectral 2 Layer Model: Model for skin scattering using bio-physical parameters: melanin type and melanin/hemoglobin concentrations. Can be estimated using inverse rendering against images (two pictures, one in white and one in combined white + blue light turns out to be enough).
Rendering Hair: Necessarily anisotropic (specular highlights along hair direction). Instead of surface normals, surface tangent (direction of hair strand) is used to parameterise reflections. i.e. instead of cos(N, H) = N· H we use $\sin(T, H) = \sqrt{1 - (T \cdot H)^2}$.
Also need single scattering for dark hair and multiple scattering for light-coloured hair. In fur, the medulla spreads all scattering distributions much more.

Monte Carlo

Monte Carlo algorithms are unbiased (mean converges to true solution), but there is always variance.

Rendering Equation: $L_r(x, \omega_r) = \int_\Omega f_r(x, \omega_i, \omega_r)\, L(x, \omega_i) \cos\theta_i\, V(\omega_i)\, d\omega_i$

On top of the previous formulation, we have added in a $V(\omega_i)$ for visibility (either zero or one). Note this still only describes direct illumination. We will account for indirect illumination later…

For a small number of point lights, integrating is easy; but for area lights, or worse, environment maps with potentially millions of pixels, brute force is way too expensive, hence the need for stochastic approximation.

Observation behind Monte Carlo: Integrals compute averages.

Let $I(f) = \int_S f(x)\, p(x)\, dx$ where p is a probability density function.

Monte Carlo estimate: $I_N(f) = 1/N \sum_{i=1 \text{ to } N} f(x_i)$ where each x_i is an independently-drawn random sample from p. Converges as N approaches ∞.

We also denote $I(f)$ as $E(x)$ - the expected value

Variance: $var(x) = E((x - E(x))^2) = E(x^2) - E(x)^2$. In image synthesis, manifests as noise.

$var(I_N(f)) = 1/N^2\, var(\sum_{i=1 \text{ to } N} f(x_i)) = 1/N\, var(f(x_i))$. Actual visible error behaves like standard deviation (square root of variance), so image quality really depends on $\sqrt{N}$ (we have to quadruple N to halve the error).

For rendering, we don't know how to sample from the entire equation.

Importance Sampling: $I(f) \approx \sum_{i=1 \text{ to } N} f(x_i) / Np(x_i)$ where $p(x_i)$ is some PDF we do know how to sample from (the proposal distribution) and the x_is are drawn from p. For this to work well, f/p needs to have low variance (i.e. f and p should have similar shape).

For direct illumination, we can choose p according to the distribution of illumination or the BRDF (i.e. either of two major contributors).

PDF Properties: $p(x) \geq 0$, $\int_{[0,1]} p(x)\, dx = 1$.

We can sample by function inversion using the CDF, $C(x) = p(X \leq x)$ $= \int_{[0,x]} p(y)\, dy$. i.e. we pick a uniform variate $u_i \in [0, 1]$ and transform u_i by C^{-1} to obtain $x_i \sim p$ ($x_i = C^{-1}(u_i)$).

Uniformly Sampling a Disk: $p(r, \theta) = 1/\Pi R^2$, $C = \int_{[0,r]} \int_{[0,\theta]} 1/\Pi R^2\, r\, dr\, d\theta$ (the extra r is the Jacobian) $= \theta r^2 / 2\Pi R^2$, as C is 2d, we define our inverse on $[0, 1]^2$, i.e. $C^{-1}(u_1, u_2) = (R\sqrt{u_1}, 2\Pi u_2)$

Cosine lobe, Gaussian lobe, GGX etc… BRDF models can be sampled analytically. Phong Lobe: $p(\theta, \phi) = (n+1)/2\Pi \cos^n\theta$. Inverse CDF becomes $(\theta, \phi) = (\arccos((1-u_1)^{\wedge}(1/(n+1))), 2\Pi u_2)$. Note isotropic so ϕ uniformly distributed. Note also the sample direction about local +Z - the samples need to be rotated to be about the global reflection vector ω_r.

IMPORTANT: $p(…) \sim …$ in exams means we can treat the RHS as the CDF.

Ward BRDF, Isotropic Gaussian: $p(\omega_h) \sim \exp(-\tan^2\theta_h / \alpha^2)$, so inverse is $\theta_h = \arctan(\alpha \sqrt{-\log u_1})$, and ϕ_h is again uniformly distributed $= 2\Pi u_2$

GGX BRDF: $p(\omega_h) \sim \alpha^2 / (\alpha^2 + \tan^2\theta_h)^2$, so inverse is $\theta_h = \arctan(\alpha\sqrt{u_1} / \sqrt{1-u_1})$, $\phi_h = 2\Pi u_2$

Ward BRDF, Anisotropic Gaussian:

$p(\omega_h) \sim \exp(-\tan^2\theta_h (\cos^2\phi_h / \alpha_x^2 + \sin^2\phi_h / \alpha_y^2))$

We first sample $\phi_h = \arctan((\alpha_y/\alpha_x)\tan(2\Pi u_2))\, u_2$ and then sample $\theta_h = \arctan(\sqrt{-\log u_1} / (\cos^2\phi_h / \alpha_x^2 + \sin^2\phi_h / \alpha_y^2))$

Ashikhmin-Shirley, Anisotropic Phong: $p(\omega_h) \sim \sqrt{(n_u+1)(n_v+1)} \cos^{e}\theta_h / 2\Pi$ where $s = \cos^2\phi_h\, n_u + \sin^2\phi_h\, n_v$. We first sample $\phi_h = \arctan(\sqrt{(n_u+1)/(n_v+1)} \tan(2\Pi u_1))$ and then sample $\theta_h = \arccos((1 - u_2)^{\wedge}(1/(s+1)))$ (note dependence on ϕ_h comes from s).

In general for anisotropic functions, we first sample ϕ_h and then use that sample plus another uniform variate to sample θ_h.

BRDF models parameterised by the half vector, ω_h, (e.g. the microfacet models above) cause us to generate samples about the half vector, and we need to handle this by computing $\omega_i = 2(\omega_h \cdot \omega_r)\omega_h - \omega_r$ and $p(\omega_r) = p(\omega_h)/4(\omega_h \cdot \omega_r)$ - reflection and reweighting.

Numerical CDF Inversion: For e.g. environment maps, we need to invert the CDF numerically to sample. We can do this just by uniformly sampling the CDF and using these samples as a lookup-table. For an environment map, we build a PDF of the average energies of each scanline and a PDF per scanline, so that we can first sample a scanline and then a specific pixel along that row. Of course, we need to compute inverted CDFs for each of these PDFs to sample. As usual, energy must be multiplied by the Jacobian ($\sin\theta$).

Direct Illumination

MC Estimate for Direct Illumination Integral:

$L_{r,N}(\omega_r) = 1/N \sum_{i=0 \text{ to } N} f_r(\omega_r, \omega_i) \cos\theta_i\, L_i(\omega_i)\, V(\omega_i) / q(\omega_i)$ where q is the proposal distribution for sampling (i.e. EM or BRDF importance).

If we sample from EM, $q_i(\omega_i) = L_i(\omega_i) / \int_\Omega L_i(\omega)\, d\omega$, so MC estimate simplifies to: $(\int_\Omega L_i(\omega)\, d\omega)/N \sum_{j=0 \text{ to } N} f_r(\omega_r, \omega_j) \cos\theta_j\, V(\omega_j)$, and variance simplifies to: $(\int L^2 / N)\, var(f_r(\omega_r, \omega_j) \cos\theta_j\, V(\omega_j))$

Sampling from EM causes variance to depend on BRDF (and visibility), sampling from BRDF causes variance to depend on EM (and visibility)! We should sample from the higher-frequency function if possible.

Multiple Importance Sampling: If both the BRDF and EM are high frequency then sampling from either will still have high variance. We can therefore get better results by sampling from both.

$I_{MIS} = 1/(M+N) (\sum_{i=1 \text{ to } M} f(x_i)\, W_m(x_i))/q_m(x_i) + (\sum_{j=1 \text{ to } N} f(x_j)\, W_n(x_j))/q_n(x_j)$

Where q_m, q_n are the proposal distributions and w_m, w_n are the weighting functions (equal weighting works reasonably well in practice).

Bidirectional Importance: The optimal distribution to sample from really is the product of BRDF and illumination. Impossible to do directly, but can use a two-step approach: First generate samples from one distribution, p, (the initial distribution) and then adjust the samples to be proportional to the product, pq (redistribution). Redistribution can be done with rejection sampling or sampling importance re-sampling (SIR):

Rejection Sampling: For every initial distribution sample, $x_i \sim p(x)$, take another uniform random sample from $[0, p(x_i) \max(q)]$. If this sample is below $p(x_i)\, q(x_i)$, accept and use the sample, otherwise reject. The relative size of the accept region is larger when the initial distribution is the one of frequency.

SIR: First compute the value of the product distribution at each sample of the initial distribution. Then use these samples of the product distribution to estimate the PDF of that distribution and take a new (smaller) set of samples to actually raytrace with (the idea being that computing the product distribution of BRDF and EM is much faster than raytracing, which also has to account for visibility).

Deterministic Sampling: Avoid variance (i.e. no noise), at the cost of producing a biased solution. e.g. instead of sampling randomly from the EM, we could pick a set number of lights. Aim is a consistent solution: with enough samples, still converges to the correct solution.

Median Cut: Recursively partitioning of the EM into regions of equal energy. Light sources can then be placed at the centroids (w.r.t. energy) of every region.

Variance Minimisation: Variation on median cut where we aim for equal-variance partitions (rather than equal energy). Turns out to create more correct shadows by avoiding an edge case in median cut where a partitions are made directly on light sources, and the centroids end up awkwardly between bright spots.

Spherical Harmonic Lighting

- Heavily used in video games as it allows for efficient and fast rendering
- IBL requires with RT: trace ray, hit geom, bounce, sample ports. Depends on the BRDF of the objects. For glossy objs, preferred dir of reflection is reflec vec and specular lobe is tight around it. For specular we need MANY samples, as not tight. Even if using Monte Carlo Sampling

Spherical Harmonics:

- Spherical Harmonics have efficient encoding and evaluation (no RT!) as the integral is computable using the SH basis function. There's in error though
- These are a 3D analog to fourier basis functions. We have diff frequencies and represent the increasing freqs with higher order polynomials.
- EMs and BRDFs ideal to represent functions over a sphere of directions
- L - order (frequency) of the harmonic; L = 0 = constant. L = 1st order change; 1 oscil from north to south. m - degree.
- (x, y, z) = ($\sin\theta\cos\phi$, $\sin\theta\sin\phi$, $\cos\theta$) (Spherical Harmonics are Cartesian)
- $Y_l^m(\theta, \phi) = K_l^m e^{(im\phi)} P_l^{|m|}(\cos\theta)$, $l \in N$, $-l \leq m \leq l$
- $K_l^m = \sqrt{((2l+1)(l-|m|)! / 4\Pi(l+|m|)!)}$

$$y_l^m = \begin{cases} \sqrt{2}\, \mathrm{Re}(Y_l^m), & m > 0 \\ \sqrt{2}\, \mathrm{Im}(Y_l^m), & m < 0 \\ Y_l^0, & m = 0 \end{cases} = \begin{cases} \sqrt{2} K_l^m \cos(m\varphi) P_l^m(\cos\theta), & m > 0 \\ \sqrt{2} K_l^m \sin(-m\varphi) P_l^{-m}(\cos\theta), & m < 0 \\ K_l^0 P_l^0(\cos\theta), & m = 0 \end{cases}$$

- Where P_l^m is the Legendre polynomial of order l and degree m
- We have 3 SH Basis Functions above parameterizable by l and m.

1. We Compute the SH Coefficients:
   - $L_{lm} = \int_{\theta=0}^{\Pi} \int_{\phi=0}^{2\Pi} L(\theta, \phi)\, Y_l^m \sin\theta\, d\theta\, d\phi$
     - L_{lm} - SH coefficient (order l and degree m) of Lighting (EM)
     - Y_{lm} - SH basis function of order l and degree m
   - We take the env map - e.g: Grace Cathedral and integrate w.r.t each basis function, ending with multiple SH coefficients. Lets say we used 16 (basis function, l, m) combos (we compute for each RGB independently). We reduce our env map from ~1 mil pixels to 16 coeffs!!
     - More terms = less error; 16 will incur still a high error
2. Then we (at render-time) reconstruct the EM using the SH Coefficients:
   - $L(\theta, \phi) = \sum_{lm} L_{lm}\, Y_{lm}(\theta, \phi)$ Reconstructs as a sum of products for specific angles more coeff = approxes higher freqs.

Features get more localized. Seeing the low freq rings = Gibbs Artefacting. SH Basis popular as we usually represent low freq funcs anyway, so don't need that high an order (and thus not THAT many coefficients).

The General Case:

- SH Projection: $C_i^m = \int_S f(s)\, y_i(s)\, ds$
- SH Reconstruction: $\tilde{f}(s) = \sum_{l=0 \text{ to } n-1} \sum_{m=-l \text{ to } l} C_l^m y_l^m(s) = \sum_{i=0 \text{ to } n^2} c_i y_i(s)$

We gen the coeffs by integrating over SH basis func y over the domain s: $C_l^m = \int_S f(s)\, y_l^m(s)\, ds$

We do that for EACH basis.

Applications:

1. Irradiance:
   - Spherical Harmonics efficiently compute diffuse reflections which used to be computationally expen with environmental illuminations
     - as for each pixel in diffuse you need to sample entire upper hemisphere and integrate all of the light. BUT the shading IS VERY smooth - low freq. Spherical Harmonics kill this!
     - $E(n) = \int_{\Omega(n)} L(\omega)(n \cdot \omega)\, d\omega$ is the normal irradiance we compute - the integral of illumination * cos (Lambert's Law)
     - As a SH: $E(\theta, \phi) = \sum_{lm} E_{lm} Y_{lm}(\theta, \phi)$. If we expand, $A = (n \cdot \omega) = \max(\cos\theta, 0) = \sum_l A_l Y_{l0}(\theta)$. Note A_l $Y_{l0}(\theta)$. is just 1 ID, and Y is just a function of theta; i.e: we discard all non m=0 (only use 3rd BF).
     - and $E_{lm} = \sqrt{(4\Pi)/(2l+1)}\, A_l L_{lm}$
     - All of the constant terms can be precomputed - more efficient!
       - Thus to compute irradiance we JUST compute the sum of the products between the coefficients and basis function!
   - You don't need much more than 3 orders. Only the samples of sub order 3 had significant value, after that every odd coeff is 0, every even is oscillating with diminishing size around 0. The first 3 samples approx 99%. e.g: 3rd order; or A0, A1, A2 is all we need; hence we only need 9 coeffs (1st order has 1 coeff, 2nd has 3 - one in each direction of l, 3rd 5…).
   - Normally we need 1M*1000 brute force upper hemisphere integs in each dir. With the 9 coeffs we JUST need 1M*9. This is amazing for diffuse shad
   - Doesn't do specular.
2. *Glossy BRDFs:*
   - Use Spherical Harmonic Reflection Maps. Glossy BRDFs require higher order SHs; order 4 <= l <= 10 usually. They are direction dependent. $B(\tilde{N}; \tilde{\omega}_o) = \int_\Omega L(\tilde{N}; \tilde{\omega}_i) \rho(\tilde{\omega}_i, \tilde{\omega}_o)(\tilde{\omega}_i \cdot \tilde{N})\, d\omega_i$
   - $B(\alpha, \beta; \theta_o, \phi_o) = \int_\Omega L(R_{\alpha,\beta}(\theta_i, \phi_i))\rho^{\wedge}(\theta_i, \theta_o, |\phi_o - \phi_i|)\, d\omega_i$. The term after the ρ is the isotropic BRDF - i.e: it will only have θ_o variation and not ϕ_o
     - And instead of using ω_i, ω_o which is about the local surface normal we use a diff parameterization where the incident and outgoing directions are about rotated coordinate systems. R means rotated coord system (about the reflection vector).
     - Reparameterization makes the function have nicer properties: the og parameterization has a diagonal full rank matrix of θ_i varies horizontally and θ_o vertically. But in the reparam, the variation only happens in one direction; θ_i so it is 1D / 1 rank matrix. Efficient!
     - Then in the reflected domain we calculate:
     - $B(\alpha, \beta; \theta, \phi_o) = \sum_{lm} L_{lm} C_{lm} D_{lm}(\alpha, \beta)$, θ_o, C_{lm} is the product of 1D BRDF and 2D lighting coefficient
     - $\varepsilon(\alpha, \beta) = \sum_{lm} C_{lm} D_{lm}(\alpha, \beta)$ - we can precompute the SH Coefficient Maps!
     - $B(\alpha, \beta; \theta, \phi_o) = \sum_{p=0 \text{ to } P} \sum_{q=-p \text{ to } p} B_{pq}(\alpha, \beta)\, Y_{pq}(\theta_o, \phi_o)$, B_{pq} is the coefficient map, Y_{pq} is the basis map

Precomputed Radiance Transfer:

- Helps games achieve more realistic global illumination with self shadowing and interreflection. Done in gaming with ambient occlusion - precompute for each vertex how much of the upper hemisphere from the vertex is occluded by the geometry itself.
- *Precomputation:* A global illumination simulator computes how a model shadows and scatters light onto itself.
  - Diffuse materials - result recorded as transfer vector of SH coefficients per vertex (BDRF * ambient occlusion)
  - Glossy materials - result recorded as a transfer matrix of SH coefficients per vertex
- *Run time* - Incident EM illumination is projected into SH basis •
  - Diffuse materials - model's transfer vector is dotted with lighting coefficients
  - Glossy materials – transfer matrix is applied to lighting coefficients followed by BRDF convolution.
- For glossy materials, the transfer function becomes a directionally dependent matrix instead of a vector for the diffuse case. Each row of the matrix encodes transfer in a different viewing direction.

Global Illumination

To compute $L_r(x, \omega_r)$ at light sources, we add an emissive term $L_e(x, \omega_r)$ to the rendering equation. To ensure conservation of energy, we need to ensure the BRDF satisfies $\Phi_r + \Phi_i = \Phi_e - \Phi_a$ (reflectance - incident = emitted - absorbed).

In context of GI, we write $L_r(x' \to x)$ for $L(x', \omega_r)$ where $|x - x'| = \omega_r$ (i.e. x is the location where are measuring from). Similarly, we write $f_r(x'' \to x' \to x)$ instead of $f_r(x', \omega_i, \omega_r)$, where x'' is the origin of the light,

Geometric Coupling Term: $G(x' \leftrightarrow x) = V(x' \leftrightarrow x) |\cos\theta| |\cos\theta'| / \|x - x'\|^2$ where θ' is the angle at which the light leaves the light source.

$L(x' \to x) = L_e(x' \to x) + \int_A f_r(x'' \to x' \to x)\, L(x'' \to x')\, G(x'' \leftrightarrow x')\, dA(x'')$
$= L_e(x' \to x) + \int_A f_r(x'' \to x' \to x)\, L_e(x'' \to x')\, G(x'' \leftrightarrow x')\, dA(x'')$
$+ \int_A \int_A f_r(x''' \to x'' \to x')\, L_e(x''' \to x'')\, G(x''' \leftrightarrow x'')$
$f_r(x'' \to x' \to x)\, L_e(x''' \to x'')\, G(x''' \leftrightarrow x'')\, dA(x''')$

+ …

Calculating lighting from infinite bounces is computationally infeasible, but because most surfaces reflect a small fraction of the light they receive, with every bounce, the magnitude of light transport falls off.

Russian Roulette Path Termination: We can achieve an unbiased estimate by randomly skipping the rest of the integral with probability q-1, but weighting the extra light contributions by 1/(1-q).

i.e. $F' = $ if $u > q$ then $(F - qc)/(1 - q)$ else c where $u \sim$ uniform [0,1] and c is some constant. $E(F') = (1 - q)\, (E(F) - qc)/(1 - q) + qc = E(F)$

We usually consider terminating only from the 3rd bounce onwards due to lower contributions (so higher variance is tolerable).

Path Sampling: Can sample light sources by area (i.e. pick points on a light source to trace towards) or according to the BRDF. For the former, need to convert solid angle probability to area, $p_A = p_\omega |\cos\theta_i| / \|x_i - x_{i-1}\|^2$

MC estimate for path is

$L_e(x_l \to x_{l-1}) / p_A(x_l) \prod_{i=1 \text{ to } l-1} f_r(x_{i+1} \to x_i \to x_{i-1}) |\cos\theta| / p_\omega(x_{i+1} - x_i)$

Bidirectional Path Tracing: Trace from both the light source and the camera (e.g. two bounces from camera, one from light). Then compute light contribution based on if the end samples are visible from each other.

Metropolis Light Transport: Mutate existing paths to obtain new paths. Mutations are accepted/rejected based on energy balance.

Metropolis Hastings Algorithm:

```
x = x0
for i = 1 to n
   x' = mutate(x)
   a  = accept(x, x')
   record(x , (1-a) * weight)
   record(x', a     * weight)
if (random() < a)
   x = x'
```

Detailed Balance: Let mutations be proposed with probability $T(x \to x')$ and mutations accepted with probability $a(x \to x')$. Then we require $f(x)\, T(x \to x')\, a(x \to x')$ $= f(x')\, T(x' \to x)\, a(x' \to x)$

So $a(x \to x')$ $= \min(1, f(x')\, T(x' \to x) / (f(x)\, T(x \to x')))$

Local perturbations are good for explorations of high importance paths, but we can get stuck in local minima! Every so often, it is good to do a global perturbation (proposal from a distribution) to maintain ergodicity.

Lens Perturbation: Slightly changing the direction from the camera. Useful for paths involving specular bounces immediately before hitting the camera. Effect is almost like image-space blurring.

Caustic Perturbation: Slightly changing the direction of a bounce on a specular surface from the light source. Useful for paths involving specular bounces followed by a final diffuse bounce before hitting the camera (e.g. depositing a caustic on that final diffuse surface).

Multi-Chain Perturbation: Lens perturbation followed by caustic perturbation. Useful for sampling caustics in a pool of water.

Start-up Bias: MLT assumes the original sample is drawn from the target distribution. We need many samples to achieve the target distribution, many of which will end up discarded. Of course, we could also draw from a proposal distribution and weight the samples but this is still sub-optimal.

Energy Redistribution Sampling: Combine an initial round of Monte Carlo path tracing with MLT-lens-perturbation-style redistribution of energy using a balanced energy flow filter - equal deposition flow.

Equal Deposition Flow: Select pixels in a local neighbourhood and deposit some of the sample's energy to it's neighbour. Image pixels in a neighbourhood will therefore all converge to similar energy.

Correlated Visibility Sampling: Similar to energy redistribution sampling but applied to visibility sampling for direct illumination from an EM. In the first pass, bidirectional sampling is used for un-occluded regions, and partially occluded pixels are marked. In the second pass, metropolis energy exchange is performed on the image plane for partially-occluded regions.

SOTA: Apply neural networks (originally MLPs to predict parameters of a filter, now CNNs) to do this second denoising pass.

Scattering

Volumetric Scattering: Light transport in participating media. Single scattering is sufficient for optically thin media, while multiple scattering is necessary when the media is optically dense.

Subsurface Scattering: Scattering inside a surface.

We can model the participating media as particles distributed in a volume of 3d space. Light can be absorbed or scattered when it hits a particle, and particles might emit their own light (e.g. fire).

Absorption: $L_o(x, \omega) - L_i(x, \omega) = dL_o(x, \omega) = -\sigma_a(x, \omega)\, L(x, -\omega)\, dt$ where σ_a is probability density of absorption per unit distance.

Out-scattering: $dL_o(x, \omega) = -\sigma_s(x, \omega)\, L(x, -\omega)\, dt$ where σ_s is the scattering coefficient, the probability density of out-scatter per unit distance.

Extinction: Can sum absorption and out-scattering into one coefficient σ_t, probability density of radiance reduction per unit distance.

Transport Coefficient: $T_r(x \to x') = \exp(-\int_{0 \text{ to } d} \sigma_t(x + t\omega, \omega)\, dt)$

Beer's Law: In homogeneous media, σ_t is constant, so $T_r(x \to x') = \exp(-\sigma d)$.

Emission: $dL_o(x, \omega) = Q(x, \omega)\, dt$ where $Q(x, \omega)$ is emitted radiance added per unit distance.

In-scattering: Described by volumetric analogue of BRDF, the phase function $p(\omega \to \omega')$. To conserve energy, we require $1/4\Pi \int_\Omega p(\omega \to \omega')\, d\omega' = 1$ (note the integral is spherical rather than hemispherical).

Total added radiance is the sum of emission and in-scattering,

$dL_o(x, \omega) = S(x, \omega)\, dt$ where $S(x, \omega) = Q(x, \omega) + \sigma_s(x, \omega) \int_\Omega L_i(x, -\omega')\, p(x, -\omega' \to \omega)\, d\omega'$

Volume Rendering Equation: $dL_o(x, \omega) = -\sigma_t(x, \omega)\, L_i(x, -\omega)\, dt + S(x, \omega)\, dt$

Phase functions, similarly to BRDFs, can be isotropic (in which case $p(\omega \to \omega') = 1/4\Pi$). If anisotropic, we can use the Henyey-Greenstein phase function $p_{HG}(\cos\theta) = (1 - g^2) / 4\Pi (1 + g^2 - 2g\cos\theta)^{\wedge}3/2$ where θ is the angle between ω and ω' and g is some constant in [-1, 1], where negative produces "backward scattering", positive produces "forward scattering" and at g=0 it simplifies to isotropic scattering.

Can measure scattering of media using lasers, and then try to fit parameters like g or the extinction/scattering coefficients σ_s/σ_a. Note albedo $\alpha = \sigma_s / \sigma_t$. Measurements usually aim to isolate single scattering to make fitting tractable.

Radiative Transport: $(\tilde{\omega} \cdot \nabla)\, L(x, \tilde{\omega}) = -\sigma_t L(x, \tilde{\omega}) + \sigma_s \int_{4\Pi} p(\tilde{\omega}, \tilde{\omega}')\, L(x, \tilde{\omega}')\, d\tilde{\omega}' + s(x, \tilde{\omega})$

Very costly computation for highly scattering materials. To optimise, can take advantage of how light diffuses (loses directionality) with multiple scattering. (Not really explained how in the lecture - MLPs can help, apparently).

Bidirectional Surface Scattering Distribution Function (BSSRDF): Generalises 4D BRDF to account for transport across material.

$S(x_i, \tilde{\omega}_i, x_o, \tilde{\omega}_o) = dL(x_o, \tilde{\omega}_o) / d\Phi(x_i, \tilde{\omega}_i)$

Diffuse BSSRDF: Removes dependence on angle of incidence/reflectance (only distance of transport).

The kernel therefore becomes radially symmetric $R_d(r) = (\tilde{n} \cdot E(x_o))/d\Phi_i(x_i)$.

Can model with the dipole diffusion model $S_d(x_i, \tilde{\omega}_i, x_o, \tilde{\omega}_o) \approx R_d(x_i, x_o)$

$R_d(r) = \alpha'/4\Pi\, (z_r(\sigma_{tr} + 1/d_r) \exp(-\sigma_{tr} d_r)/d_r^2 + z_v(\sigma_{tr} + 1/d_v) \exp(-\sigma_{tr} d_v)/d_v^2)$

Note $R_d(x_i, x_o)$ models isotropic Gaussian-like diffusion between two points as the sum of two Gaussian falloffs (assumes semi-infinite medium - i.e. thickness of surface not taken into account). Effectively, we want to treat the surface as if it contains an extra (real) light source, so to conserve energy, we need a "negative" (virtual) light source outside the surface. d_r and d_v are the distances to these real and virtual light sources, z_r and z_v are distances of the sources to the surface, α' is albedo and σ_{tr} is the extinction coefficient (note even the "real" light source under this terminology isn't actually a physical emitter of light - just an approximate point under the surface from where the scattered light appears to return from).

Single Scattering BRDF Approx: $f_r^{(1)}(x, \tilde{\omega}_i, \tilde{\omega}_o) = \alpha F\, p(\tilde{\omega}_i' \cdot \tilde{\omega}_o')/(|\tilde{n} \cdot \tilde{\omega}_i'| + |\tilde{n} \cdot \tilde{\omega}_o'|)$ where α is the single-scattering albedo, F is fresnel and p is the phase function (Henyey-Greenstein forward scatter) - here we just assume the point the light leaves the surface is the same at which it entered (i.e. it travelled no distance in the surface while being scattered).

Skin is unfortunately not entirely homogeneous. We can model diffusion through multi-layered media using the multipole model, which includes multiple pairs of real and virtual light sources.

Kubelka-Munk theory studies transport through two interfaces as a geometric series of events (i.e. we could enter the first layer and then scatter back out, or pass through the first layer, enter the second layer, scatter back to the first layer, and then either leave or scatter back into that second layer, etc…). We have $T_{12} = T_1 * T_2 + T_1 * R_2 * R_1 * T_2 + … = T_1T_2/(1 - R_2R_1)$

$R_{12} = R_1 + T_1R_2T_1 + … = R_1 + T_1R_2T_1/(1 - R_2R_1)$ where T_n/R_n is the amount of radiance transmitted through/reflected at the nth boundary and T_{12}/R_{12} is the bulk radiance transmitted/reflected overall.

For real-time rendering (i.e. no photon mapping/path tracing), can simulate diffusion in texture space. Need to also account for texture distortion (i.e. different texels will take up different amounts of the surface). Can also do the diffusion in screen-space but need masks for which materials should be blurred and separate diffuse/specular/depth buffers (deferred shading).

BRDF Measurement

Analytical Models are limited - only describe specific surfaces & params are hard to obtain. You can measure them as tabul. data & fit to models or BFs,

- Measure with Goniereflectometers - we interpolate missing measures ofc.
- There's databases recording it.

Common techniques for measuring BRDFs:

- Spherical sample of isotropic BRDFs: 1 photograph can handle all reflectance directions, with a lightsource rotation along 1 axis. These values are all recorded in a model database.
- Whatever materials can't be spherized you can make a cylindrical mount which you mount strips of anisotropic fabric. Each strip is a slightly different rotation of the sample
- Measurement setup for anisotropic reflections on brushed alum / metal: camera looks at object reflection on hemispheric half-silver dome (reflec/trans both 50%), light is moved on inside while cam measures reflec.
- Catadioptic measurements: use mirrors and cameras - place cam and light source at one end and focus on the parabolic mirror, and put the sample at focus of parabolic mirror, the rays converge and you get many measurements covered by parabolic mirror in 1 photo. Beam splitter - share same dir onto mirror and check incoming and outgoing mirror. Achieves low angle range. Cylindrical mirror - each radial line gives diff angle of incidence.
  - Beam splitter, cam, proj within a curved mirror which has a suspended mirror dome beneath the hole. light reflects in to the dome, hits the side, the material then goes back out. They project BFs - orthonormal basis functions Z_k, the coefficients in Z_k are recorded. You illuminate the setup with the basis illumin. condition and this optically calculates eqn.
    - The photograph then is an image of the coefficient of the basis func.
      - Quick - processing is done at speed of light, coeffs act as low pass filter on noise in the BRDF. Limited to glossy BRDFs, not v. shiny.
  - Can use ellipsoidal mirror too - by pixel by pixel sampling the BRDF
- LEDs going through small dome inserted acting as light source and photodiode. One LED is on, the rest are sensor mode. Each takes turns. Only order n measurements. Fixed sampling, need to interpolate others.

Data Driven:

- We refer to the sphere measurement idea from earlier. From it we have a database of 100s of material BRDFs.
- We simplify data further for analysis - parameterization using *Rusinkiewicz*
  - Reduces to a 1 rank matrix
  - Lots of the BRDF var happens at ω_h near 0 so take most samples there
  - We can then tabulate as 90 (θ_h) x 90 (θ_d) x 360 (ϕ_d) (or 180, $\phi_d = \phi_d + \Pi$)
    - Easy to use. You can halve bins with reciprocity! But 17Mb/12hr to get
  - If we linearize each BRDF in a long vector and do PCA, around 45 principal components are enough to have a very good reconstruction!
    - If you take the eigenvalues, and interpolate b/w eivecs, you can interpolate BRDFs to go to diff colours and rough / shinier metals
    - However BRDFs are non linear, if you edit them slightly / go in a non linear direction you can ruin the material appearance and make a non physical BRDF. You can locally linear proj it with about 10-15 coeffs to avoid that issue.
  - We can also express new BRDFs as lin combs of the 100 measured ones
    - Solve b = Pa; b = linearized BRDF (4M x 1; new data), P = matrix of BRDFs (4M x 100); a = unknowns (100*1). This makes the db smaller
    - But this is hugely overconstrained = many more knowns than unkn.
      - So we just take 800 random rows from P; b'=P'a, and P and b have 800 rows instead of 4 million. The model DB can fit new mats!
  - Optimal BRDF sampling: it's found that 5 views is enough for sufficient spherical smpls. Fitting based on prj to space span by 100 MERL BRDFs

Neural BRDFs: just use NeurNet (check the BRDF section). If you interp. 2 NNs weights, we can approx the BRDF for mixed materials / mats that are similar to both, or for blending. To fix specular being overweighted, use a ZNBRDF which linearises the inputs, with a 32x1 latent space which is what we interp. 675x1 & 675x1 -> 32x1 -> 100x1 -> 675x1. Surrounded by Enc & Dec.

Spatially Varying Reflectance Modelling

Every single pixel acts as its own 6D BRDF (surface pos, incident, exitant)
Ideal for planar surfaces. Measured via analytic, data-driven or statis. models.
We need to recover diffuse colour, specular colour, and specular roughness.

- We can scan this with a horizontal camera scan over document + light
  - Obtain diffuse & specular intensity maps, spec roughness (which we fit to ward BRDF model; dark = shiny & low roughness), translucency (with backlight), normals map (x, y gradients using a scanning system to find the local variation) -> constructs displacement map
- We need fewer images to sample plane w/ linear light sources. Compresses dyn ranges -> just 1 exposure. Only need simple 1D translation machinery.

Reflectance Traces for Each Pixel:

- We over time measure the reflectance from a pixel as the translation light goes down it. Diffuse peak td = when light coincides w/ sf norm. Specular peak tm = when light is aligned with the mirror reflection
- BRDF fitting: 1. Fit diffuse 2. Subtract diffuse 3. Estimate mean&var of spec 4. Look up specular parameters (in table - we have them)

Pocket Reflectrometry:

- BRDF Charts - we have squares representing diff BRDF colour samples
  - Diff common material - rubber, plastic, metals, fabrics…
- Using a moving light tube above the material sample with a BRDF chart next to it. Record it with a camera.
- The chart helps calibrate relative reflectivity of the pixel compared to the known reflectivities of the 24 samples in our BRDF chart.
  - To figure out light direction (we don't know at each timestep) to properly relate points on the chart to points on the sample (as we have different angles of incidence) we use *Dynamic Time Warping*. This allows us to shift two sample traces for maximum alignment.
  - Then we can just fit pixels to samples using the chart!
    - $r = da + sb$, $a(t) = \int_{\Omega^+} L(i)\alpha(i, o)(n \cdot i)\, di$, $b(t) = \int_{\Omega^+} L_t(i)\beta(i, o)\, (n \cdot i)\, di$
    - Then we fit: $\min[u_0, u_1, …, u_k] \|r - u_0 a - \sum_{[j=1 \text{ to } k]} u_j b_j\|$, $u_j \geq 0$, $b_j \in \Phi(r)$
    - a: diffuse BRDF with albedo d, b: specular BRDF with params s
    - Instead of direct estimation of s, estimate a linear combination of k exemplar BRDFs in BRDF chart that fits the observed data per-pixel
  - Works well for both bumpy and flat surfaces, for marble and metal!

Second Order Statistics of Reference:

- Instead of measuring 1000 outputs for specular (which is like a Gaussian), just measure variance σ^2 about mean μ. This is the second order statistic!
- For reflect vec & specular roughness the 2nd order stat is also σ^2 and μ
- Requires Light Stage. Illuminate sample with 1st order gradients (related to 1st order SHs, but mapped from 1 to 0 in (x,y,z) dir) then you estimate the reflection vector, and hence the surface normal which is the spec norm
  - Then if you do 2nd order grads - bright to dark to bright in various directions, can directly estim 2nd order spec component - roughness
  - Still need to polarize light for seperate specular and diffuse
  - In 1D, the moments of f(x): $\alpha = \int f(x)\, dx = L_0$, $\mu = \int x^*(f(x) / \alpha)dx = (1 / \alpha)\int x * f(x)\, dx = L_1 / L_0$, $\sigma^2 = \int (x - \mu)^2 * (f(x) / \alpha)\, dx = L_2 / L_0 - (L_1^2 / L_0^2)$
  - Sigma squared:
  - total energy α = – mean μ – variance σ
  - To get specular albedo, illuminate the sample in two polarization states, one that images the specular and one that does the diffuse. We get this integral: $L_0 = \int_\Omega f(\vec{\omega})\, d\vec{\omega}$. For sf normal we do linear gradient in x, y, z: $L_1 = \int_\Omega \vec{\omega} f(\vec{\omega})\, d\vec{\omega}$ = reflection vector. We can extract the sf normal fro the reflection vector. For the specular roughness we illuminate with 2nd order gradients $L_2 = \int_\Omega \vec{\omega}\vec{\omega}^{\wedge}T\, f(\vec{\omega})\, d\vec{\omega}$
    - These gradients need to be oriented about the local coord of the reflectance function. We rotate them to be correct for both an/isotropic using SHs. They have a steerable spherical basis (can be rotated over 3D Sphere), and we can capture the reflectance with SH patterns. We just need a fixed number of measurements related to the spherical harmonics and then and do the steering afterwards.
      - For the 2nd order, there's 5 SH BFs, and we do a linear comb of those to rotate the spherical harmonics
      - We estimate σ_x^2 and σ_y^2 - anisotropic specular roughness. We also extract anisotropy and the tangent and bitangent of it.
- With a purely flat sample, a flat screen is enough rather than light stage
  - Put the screen in the reflection direction of the camera w.r.t the sample, the screen is large enough to cover the refl. of planar sample's refl lobe
    - Then we use the screen to illum the sample with 0th, 1st, 2nd grads The computer screen is already polarized. Enough for isotropic refl.

Highly Specular Objects

- Hard to acquire their BRDFs in regular light stages - Niquist-Shannon.
- We can cover obj. with full sphere of light. LED arc makes continuous sphere of light through rotation. Long exposure photog used.
- Rotating the arc gives us a ball of light. Instead of doing a constant illumn as the arc rotates, we can program the LEDs to change their light values to change their brightness values to trace out the SH lighting patterns and not just const illum (i.e: 2nd order SH pattern bright-dark-bright). n orders work.
  - At higher orders / freqs - diffuse gets darker, specular remains similar. By 3rd order, the diffuse p much disappears. i.e: subtracting 3rd order from 0th order gives us a good diffuse seperation.
  - For specular roughness - take the ratio of two central bands of high freqs (e.g: 3rd and 5th order) then you take the 2 harmonics of spec freqs and align them with the specular lobe. Broad specular lobes have a stronger resp to the third order SH and a less strong to 5th; opposite for sharp.
  - We can then build up a lookup table via simulations
- To digitise we need more views - use 5 cameras and rotate sample

Mobile Surface Reflectrometry: we use a camera flash pair - this can sample the upper hemisphere with a handheld device. $\omega_i = \omega_r$, but this is still enough to recover the parameter maps to render all samples

- Because each photo views the sample at a different angle, we do computer vision to register all the views to a central view (from above; optical flow)
- We also need to know where the camera is looking from: Android API (getRotationMatrix), or 3D tracking (SLAM)
- Once you know incidence angle you can recover the normal map (weighted avg of multiple measurements). Diffuse albedo - use median. Specular albedo - get the integral of the diffuse subtracted values. Roughness - microfacet BRDF fit GGX

Deep Learning SVBRDFs:

- Take a flash photo. UNet with skip allows estimating spatially varying diffuse albedo, surface normal, specular albedo and roughness.
- Training done on novel rendering loss function instead of just L1 or L2 error on the reflectance maps
- Synthetic training data – artist and procedurally created appearance maps