Lecture 1: Introduction

Hawley, Adam

November 22, 2018

1 Admin

1.1 Exam Format

One 2 hour exam at the beginning of Spring Term.

- One question worth 20% (basic knowledge, shorter questions)
- Two questions worth 40% (more advanced, longer questions)

Note: Programming will NOT be examined

1.2 Reading

- Nielsen, “Visual Computing: Geometry, Graphics and Vision”, Charles River Media, 2005
- Prince, “Computer Vision”, Cambridge University Press

Lecture 2: Transformations

Adam Hawley

November 19, 2018

References: Nielsen 3.2.1 & 3.2.2

Contents

1 Conventions

- Matrices written in uppercase bold eg: $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$
- Matrices are also indexed (row, column) from top left starting at 1.
- Individual elements are written as lowercase:
$$a_{11} = 1, a_{12} = 2, a_{21} = 3, a_{22} = 4$$
- Vectors are simply matrices with a single row or column represented by a lower-case bold symbol:
$$\mathbf{a} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \; OR \;\; \mathbf{a} = \begin{bmatrix} x & y & z \end{bmatrix}$$

2 Transformations

2.1 Translation

Translation involves ***component-wise addition***.

For example:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

2.2 Scaling

Scaling involves ***multiplication by a scalar***.

For example:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = s \begin{bmatrix} x \\ y \end{bmatrix}$$

Or to have a separate dimensional scalars:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

2.3 Rotation

Rotation involves ***matrix multiplication***. Using the formula below to give a 2D rotation counter clockwise by angle θ:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation preserves distances:
$$||\mathbf{x}|| = ||\mathbf{Rx}||$$
$$||\mathbf{x}||^2 = x_1^2 + x_2^2 + ... + x_n^2 = \mathbf{x}^T\mathbf{x}$$
$$\mathbf{x}^T\mathbf{x} = (\mathbf{Rx})^T(\mathbf{Rx})$$

$$\begin{aligned}\mathbf{x}^T\mathbf{I}\mathbf{x} &= (\mathbf{x})^T(\mathbf{R}^T)(\mathbf{R}\mathbf{x}) \\ &= \mathbf{x}^T(\mathbf{R}^T\mathbf{R})\mathbf{x}\end{aligned}$$

And hence:

$$\mathbf{R}^T\mathbf{R} = \mathbf{I}$$
$$\mathbf{R}^T\mathbf{R}\mathbf{R}^{-1} = \mathbf{I}\mathbf{R}^{-1}$$
$$\mathbf{R}^T = \mathbf{R}^{-1}$$

So to get the inverse of a rotation you only need to transpose it.

2.4 Affine Transformations

All of the previously mentioned transformations have been **affine** transformations. These are transformations which can be expressed as matric multiplication and addition:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

For 2D, **A** is a 2x2 matrix and **b** a 2x1 vector.

For 3D, **A** is a 3x3 matrix and **b** a 3x1 vector.

3 Homogeneous Coordinates

3.1 Defining Homogenous Coordinates

Representing the affine transformations can be done by moving from Cartesian to homogeneous coordinates.

All 2D affine transformations can be then represented by a 3x3 matrix (or 4x4 for 3D)

$$\begin{bmatrix} x \\ y \end{bmatrix} \textit{becomes} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

3.2 Affine Transformations as Homogeneous Coordinates

The extra dimension allows uniform treatment of transformations:

Translation: $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Scaling: $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & 0 & s_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Rotation: $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

In general, any affine transformation:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$$

Can be expressed as:

$$\begin{bmatrix} y \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & \mathbf{A} & & \mathbf{b} \\ 0 & ... & 0 & 1 \end{array}\right] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$$

Note: Two homogeneous points are equal if they differ only by a scale factor.

Lecture 3: Photometric Image Formation Part 1

Hawley, Adam

December 14, 2018

Contents

1 Types of Sensor

There are many different types of sensors used in computer vision:

- Optical
  - CCD, Photodiodes, Photomultipliers
- Infra-red (thermal imaging cameras)
  - CCD (Cooled), Photodiodes
- Synthetic Aperture Radar (SAR)
  - Radar, Antenna
- Range Sensors
  - Laser & Photodiode
- MRI
  - Magnetic field gradients applied causing production of rotating magnetic field which can be measured.
- PET/CAT
  - Simulated radiation emission via magnetic field or radio isotope.

2 From Light to Images

2.1 Definitions

- ***Irradiance***: power incident on a surface (power per unit area).
- ***Radiance***: power travelling from a source (power per unit solid angle per unit projected source area).

2.2 Charge-Coupled Devices

The most common device for digitising image information is a charge-coupled device (CCD). They are made up of a square array of solid-state capacitors:

From the photo-electric effect photons of light knock out electrons from the upper plate and hence charge accumulates in the electron capture "*wells*". Using a shift register, capacitors transfer their contents to the appropriate neighbour. The final capacitor dumps the charge for each capacitor into the analogue-digital converter (ADC). The dump happens once for each capacitor until the contents of each one has been read.

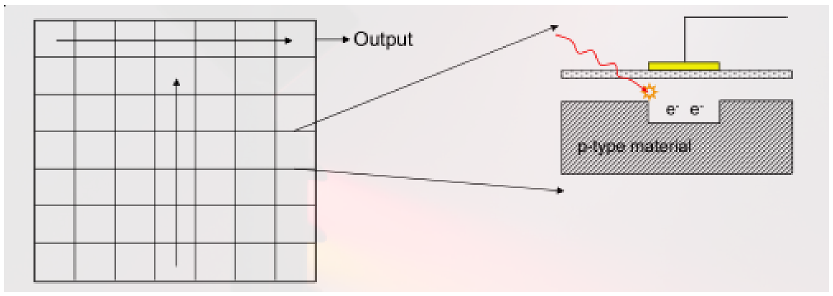


Figure 1: The flow of electrons inside a CCD

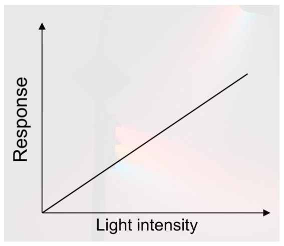


Figure 2: Ideal response of a CCD

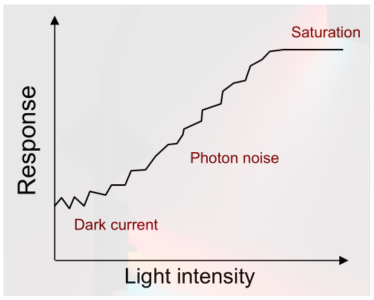


Figure 3: Typical response of a CCD

The ideal reading from a CCD would be as simple as "$output = gain \cdot input$". See figure 2. However, a typical reading from a CCD also includes bias and noise (as can be seen in figure 3). ("$output = gain \cdot input + bias + noise$") The noise mostly comes from the following sources:

- Dark Current - Thermal noise.
- Photon Noise - Quantum noise.
- Quantisation of pixels.
- Amplifier

3 Saturation

The electron capture wells inside the CCD have a finite capacity $\approx 700\epsilon^{-}/\mu m^{2}$. That means that there is roughly a maximum of 100,000 e^{-} per well in a typical CCD. When a well is full, the pixel is said to be saturated. CCDs have a small illumination range (this is what *HDR imaging* attempts to overcome). In an 8-bit image, a pixel brightness of 255 indicates saturation.

4 Noise

4.1 Signal to Noise Ratio

The level of noise relative to the signal strength is measured by the *signal to noise ratio*. This is often expressed in Decibels (10 times the logarithm of the ratio):

$$SNR(dB) = 10\log_{10}\frac{signalpower}{noisepower}$$

10dB is a tenfold increase in power, 3dB is a doubling of signal power.

4.2 Photon Noise (Shot noise)

- Light sources emit light in photons.
  - β photons/second on average
  - However the photons are emitted randomly.
- They follow a Poisson distribution of time t.
  - Therefore there is a mean intensity of $\mu = \beta t$
  - And a standard deviation of $\sigma = \sqrt{\beta t}$

Note: At mid-brightness levels the *Signal to Noise Ratio*(SNR) for photon noise is around 23dB.

4.3 Dark Current

Dark Current is a product of thermal energy from within the CCD. Electrons are displaced over time that are independent of light actually falling on the sensor (therefore this effect builds up with exposure). The effect also obviously increases at higher environmental temperatures. The manufacturing process means certain electron wells may have higher than average dark current (*hot pixels*). *Dark frame subtraction* can be used to remove an estimate of the mean pattern of dark current. However, dark current is also random and hence why it still affects CCD readings. The randomness of the dark current follows the Poisson distribution just as the photon noise did.

Dark current can be easily shown by taking a picture in complete darkness and inspecting the images to see that it is unlikely that all the pixels will be of absolute darkness.

The SNR for dark current in mid-light levels is approximately 38dB.

4.4 Quantisation noise

Pixels in photos are digitised to 2^B levels. There is a uniformly distributed error on values of $\pm\frac{1}{2}$. This distribution leads to the following properties:

- Mean: $\mu = 0$
- Standard Deviation: $\sigma = \frac{1}{\sqrt{12}}$
- $SNR = 10log_{10}(2^{B-1} \cdot \sqrt{12})$

$SNR = 26dB$ for 8-bit digitisation, which is better than the photon noise ratio. Photon noise is the limiting factor of CCDs.

5 CCDs and Colour

CCDs record photons of all wavelengths, hence they are all monochrome. A colour image requires red, green and blue values.

To get colour when using a CCD, there are three options:

1. Take 3 separate images with a different filter places over the lens. This technique can use filters to allow other frequencies to pass and is hence a multispectral camera. However this method can have issues with motion between frames.
2. Use a beamsplitter and 3 different CCDs, each with their own colour filter. Using the beamsplitter means that one can have instantaneous colour image capture but careful calibration is required to get exact correspondence between pixels of 3 cameras and it is 3 times more expensive.
3. Place filters over individual pixels in a mosaic pattern. This is the most common approach. There are more green than red/blue filters since human vision has higher resolution in green than in red/blue. The colours must be interpolated (the process of *demosaicing*). This method also reduces the effective resolution of the CCD.

6 Conclusions

- Measuring a colour image digitally is difficult
- CCDs are generally the best choice
- There are many sources of noise, namely:

  - Photon noise
  - Dark current
  - Quantisation noise

- No easy way to capture colour with a CCD
- Most common option is a Bayer mosaic
  - But this definitely introduces artefacts
  - (in fact — these can be used to detect forgeries)

Lecture 3: Photometric Image Formation Part 2

Hawley, Adam

November 14, 2018

Contents

1 Brightness

Light reaches the camera from a scene. The amount of light arriving from the scene depends on what is in the scene (strength of light sources, reflectance properties of objects in the scene etc). The pixel brightness reported in an image for a given scene radiance quantity depends on the *exposure* of the image. A camera can control *exposure* in a number of ways.

1.1 Formulae

Before discussing the several parameters which can be adjusted to increase total exposure, there are a couple of formulae to be explained and understood first.

- The amount of light captured by a lens is proportional to the area of the aperture:
  $$Area = \pi(\tfrac{D}{2})^2$$
  D is the diameter of the entrance pupil.
- The **f-number**(N) is the ratio of the focal length (f) to the diameter of the entrance pupil:
  $$N = \tfrac{f}{D}$$
  The higher the f-number, the less light that reaches the CCD (smaller diameter)

Substituting D gets:
$$Area = \pi(\tfrac{f}{2N})^2$$

- Area is proportional to the reciprocal square of the f-number
- Photographers have found it convenient to define a discrete set of f-numbers as "stops".
- Each stop represents a halving of area of the aperture relative to the previous stop.

$$\begin{aligned} A_1 &= 2 \cdot A_2 \\ \frac{\pi(d_1)^2}{4} &= 2 \cdot \frac{\pi(d_2)^2}{4} \\ (d_1)^2 &= 2 \cdot (d_2)^2 \\ d_2 = \sqrt{(d1)^2 \cdot \frac{1}{2}} &= (d_1) \cdot \frac{\sqrt{1}}{\sqrt{2}} = \frac{d_1}{\sqrt{2}} \end{aligned}$$

Therefore to halve the area of the aperture the f-number must be multiplied by $\sqrt{2} \approx 1.4$.

Hence, the set of stops is:
$$\tfrac{f}{1}, \tfrac{f}{1.4}, \tfrac{f}{2}, \tfrac{f}{2.8} \cdots$$

Each stop represents a halving of the light intensity from the previous stop. Hence aperture gives us one way to control how much light reaches the camera. However, as one changes the aperture, other complex effects are introduced.

As the f-number reduces (aperture gets larger, more light enters lens), “depth of field” gets smaller and sharp focus is only possible for a limited range of distances.

1.2 Aperture

Aperture is the first parameter which can change the overall light exposure onto the CCD. The *aperture* of a camera is the hole through which light travels. Aperture size determines the cone angle of a bundle of rays that come to focus in the image plane.

- Small aperture = less light and more highly collimated rays admitted, sharp focus at image plane.
- Large aperture = more light and uncollimated rays admitted, sharp focus only for rays with a certain focal length.

Aperture sizes are measured in units of *stops* with cameras offering a discrete set of aperture sizes.

1.3 Exposure

Exposure is the accumulated physical quality of light applied to the image-plane over a given time.

Product of the image irradiance (light arriving at the CCD) and time:

$$H = E \cdot t$$

Note: In practice, cameras apply a non-linear transformation to intensities so the discretised pixel values will be a function of H. However, this is ignored under the *“linear camera”* assumption.

Hence, another way to control how much light reaches the CCD is to change the exposure time.

However, increasing the exposure can cause motion blur and dark current noise (see Lecture 3 notes) will also increase with time.

1.4 Gain

The brightness of an image can also be increased by passing the analogue signal from the CCD through an amplifier. The signal gain of a camera is measured on the *“ISO”* scale. Increasing ISO from its default value of 100 corresponds to increasing signal gain. Modern CCDs can support incredibly high ISO values and hence operation in very low light. Remember, increasing the gain will not affect any saturated pixels, they will simply remain saturated.

1.5 Summary

To summarise, there are three main ways one can increase the brightness of an image:

- Increasing the aperture (smaller f-number). (Reduced depth of field)
- Longer exposure (motion blur is worsened).
- Increasing the gain through a large ISO (noise is also amplified)

2 High Dynamic Range Imaging

2.1 History

- **1850s**: Gustave Le Gray - Chemical process for combining two exposures.
- **1950s**: "Dodging and burning", during film development, make local adjustments to sensitivity.
- **1988-1993**: Technion, Israel — local tonemapping, first HDR video.
- **Modern HDR**: Global HDR **(1993)** — idea of combining images to compute a global luminance map then tonemap, Debevac et al. **(1997)** — the method explained below.

2.2 HDR — How is it done today?

High dynamic range imaging captures a sequence of images with different exposures. Combining the images captures a much wider dynamic range. The brightness of pixel i in image j is given by:

$$\begin{aligned} H_{ij} &= E_i \cdot t_j \\ \log(H_{ij}) &= \log(E_i) + \log(t_j) \end{aligned}$$

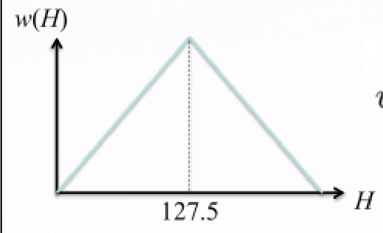
- t is varied for each image
- H are the known pixel values
- E are the unknown pixel radiances

Each image gives us an estimate of the radiance at a pixel.

$$\log(E_i) = \log(H_{ij}) - \log(t_j)$$

The pixel brightness tells us something about how reliable the estimate is:

- ≈ 255 If a pixel is very bright (close to being saturated then the estimate is unreliable.



$$w(H) = \begin{cases} H & \text{if } H < 127.5 \\ 255 - H & \text{if } H > 127.5 \end{cases}$$

Figure 1: Formula for calculating weights based on pixel value

- ≈ 0 If a pixel is very dark it will be noisy and unreliable.

We give more weight to pixel values near the middle of the range: To estimate the true radiance at every pixel we then simply take a weighted average of the estimates from each image:

$$\log(E_i) = \frac{\Sigma_{j=1}^{K}\omega(H_{ij})(\log(H_{ij}) - \log(t_j))}{\Sigma_{j=1}^{K}\omega(H_{ij})}$$

Saturated pixels are completely exluded. Radiance estimates for a pixel come from images where the exposure is appropriate for the scene radiance. For colour images, separate channels can be processed independently.

3 Tonemapping

3.1 Gamma Compression

Gamma compression is an effective global (same function applied to all pixels) technique.

$$I_{tonemapped} = cE^{\gamma}$$

- c is a constant scaling where $c > 0$.
- γ is a constant that determines the contrast of the image where $0 < \gamma < 1$.

3.2 Gradient-based Techniques

The best techniques are local (output colour only depends on the local region around the pixel). Gradient-based techniques try to preserve local gradient with less concern about overall brightness. This means that they solve for a set of brightness values that minimises error in gradient while remaining between max and min allowable values.

Lecture 5: About Images

Adam Hawley

November 19, 2018

Contents

1 Scenes & Images — The Basics

1.1 Intro

- Images represent a projection of the information from a scene in 2D.
- Scenes are real in vision, and created in graphics.

1.2 What is an image?

- An image is a grid of pixels characterised by image size and pixel values.
- Each grey-level image pixel has 8 bits so its value ranges from 0–255.
- Each colour pixel has 3 colour components: red, green and blue.

1.3 Histograms

Histograms represent the *the global statistical information* from the image, which may or may not correspond to a specific object. They count the frequency of each value in the image.

Histograms are represented as 1-D arrays for grey-level images and as 3-D arrays for colour images - one for each component.

Histogram stretching represents a mapping of the histogram aiming to improve contrast.

1.4 Graphical Objects and 3D

- Voxel representaions can be made by adding layers of images together, producing a volumetric (3D) image.
- A *graphical object* can be represented as a mesh which is a sequence of vertices joined by polygons (usually triangles).

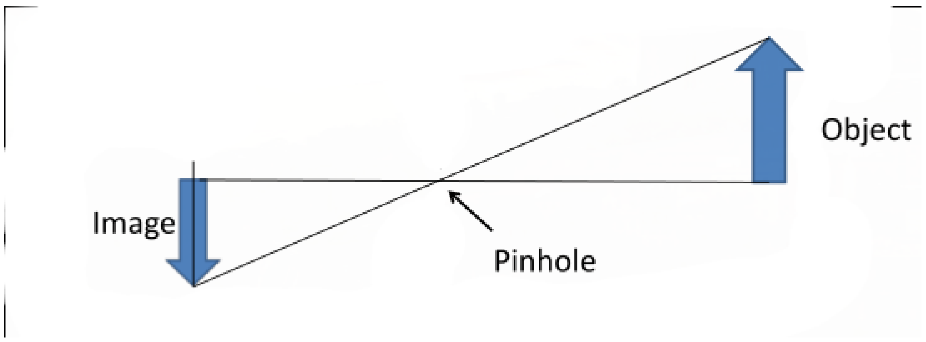
2 Image Production

2.1 Abstract View

Stepping back, one can see that image production has four important properties, the latter two of which will be discussed in greater detail later in the notes.

1. Images represent projections of real scenes as viewed by the human eye or as taken by a camera.
2. Images also represent projections of artificially generated scenes in computer graphics.
3. The theoretical model of vision we use is called the *Pinhole Camera*
4. The more accurate model of real life is known as the *Thin Lens Camera*

2.2 Pinhole Camera



The pinhole camera model allows us to use a simplified model of a camera as in figure 2.2. Properties of a pinhole camera:

- Continuous sharpness of depth.
- Very wide angle range.
- Exact perspective projection.

Note: Pinhole cameras can produce dark images because not enough light enters the camera.

Methods to improve pinhole images:

- Increase exposure time (however movement will blur the image).
- Increase the size of the pinhole (spread of light causes blurring).
- Use lenses to concentrate the light.

2.3 Thin Lens Camera

Real images have lenses which collect light over a finite aperture controlling the amount of light received. Lenses are charactised by the focal distance f that depends on *lens curvature* and *lens material*.

Note: we discuss single lenses however real cameras use a sequence of lenses.

3 The View Plane

The *View Plane* represents where the image is formed.

In practice the view plane could be any of the following:

- Photo-reactive chemicals.
- Charge-Coupled Device (CCDs) or CMOS sensors.
- Retina in the human eye.

Lecture 6: Projections

Adam Hawley

November 26, 2018

Recommended Reading:

- Prince, Chapter 14
- Nielsen 3.3
- Forsyth & Ponce Chapters 1 and 3

Contents

1 Viewing Systems

In real life we pick up objects, position them and then view them. In computer graphics objects are positioned in a fixed frame. The viewer moves to the appropriate position in order to achieve the desired view.

2 Perspective Projections

We assume that there exists a *Centre of Projection* (COP). Objects' distances from the **COP** cause the objects to appear larger or smaller on the intersecting viewplane.

Lines that are not parallel to the viewplane converge to a vanishing point. The *Principal Vanishing Point* exists for lines that are parallel to the principal axis. This can be clearly seen in the perspective projection of a cube, edges which are parallel to one another appear to converge.

3 Parallel/Orthographic Projections

Here the COP is **always** at infinity. This means that the viewplane is aligned with the axes and the *Direction of Projection* (DOP)

4 View Reference Coordinate System

4.1 Viewing Parameters

- Location of viewplane
- Window within viewplane
- Projection type
- Projection reference point (**prp**)

4.2 Viewplane Parameters

- View Reference Point (**vrp**)
- Viewplane Normal (**vpn**)
- View Up Vector (**vup**)

4.3 Window Parameters

- Width — u_{min} and u_{max}
- Height — v_{min} and v_{max}

4.4 Coordinate System in Different Projections

For **Perspective Projections** there is only one property:

- **prp** becomes **cop**

In **Parallel Projections** there are three properties:

- **cw** — The center of window
- **dop** — Direction of projection (**cw** – **prp**)
- The projection is *Orthographic* if **dop** and **vpn** are parallel

4.5 View Volumes for Graphics

The view volume for perspective projections is a truncated pyramid (frustrum) while the view volume for parallel perspective projections is a cuboid. The view volumes define a specific region of the scene which is viewed in the current image. The view volume is defined by the clipping planes, the projection rays used by the projection system and by the image rectangle. They are sometimes used in graphics for visualisation such as creating smoke or mist defined only in a specific region of the scene.

4.6 Focal Length in Perspective Projection

Focal length (ϕ) is used for two things:

- Modelling the effect of the distance to the focal plane
- Modelling the density of the receptors (e.g. CCD pixels)

$$x = \frac{\phi u}{\omega} \quad y = \frac{\phi v}{\omega}$$

In practice, the receptors may not be square. If this case just use different focal length parameters for x and y dimensions:

$$x = \frac{\phi_x u}{\omega} \quad y = \frac{\phi_y v}{\omega}$$

4.7 Offset

Currently we assume that (0,0) is where the principal ray strikes the image plane however if this is not the case (and a different coordinate system is used e.g (0,0) in the top left corner) then one can use an offset (δ) or a skew parameter (γ):

$$x = \frac{\phi_x u + \gamma v}{\omega} + \delta_x \quad y = \frac{\phi_y v}{\omega} + \delta_y$$

4.8 Intrinsic Parameters

The aforementioned focal lengths, skew and offset form the set of **intrinsic paramters**:

$$\phi_x, \phi_y, \gamma, \delta_x, \delta_y$$

It is convenient to store these in an **intrinsic matrix**: $\begin{bmatrix} \phi_x & \gamma & \delta_x \\ 0 & \phi_y & \delta_y \\ 0 & 0 & 1 \end{bmatrix}$

4.9 Extrinsic Parameters

If the camera itself is not centered at the origin of the world coordinate system which is often the case. If more than one camera is going to be used then it is very common for an arbitrary coordinates system to be used. To deal with this we express world coordinate points in the local coordinate system of the camera before they are passed to the projection model.

$$\begin{bmatrix} u' \\ v' \\ \omega' \end{bmatrix} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ \omega \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix}$$

or

$$P' = \Omega P + T$$

4.10 3D Transformations

Ω above is the product of 3D transformations where:

$$\Omega = R_x R_y R_z$$

The same principles apply but now to 4D-homogeneous space. Rotation is still defined with respect to a specific axis:

Axis	**Positive Direction**
x	$y \rightarrow z$
y	$z \rightarrow x$
z	$x \rightarrow y$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.11 Camera Calibration

How does one point in the scene relate to the observed point in the image. A 3D point of coordinates (X, Y, Z) from the scene is related to a 2D point (u, v) from the image by the equation:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_x & \gamma & \delta_x & 0 \\ 0 & \phi_y & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

Lecture 7: Multiple View Vision

Adam Hawley

December 26, 2018

Contents

1 Epipolar Geometry

Considering two pinhole cameras with different projection centres, in order to relate the two images one can use *Epipolar Geometry.*

- p_l & p_r are the vectors from the centres of projections O_l in the left and O_r in the right image to the corresponding projections P_l & P_r of the same 3D point P.
- h_l & h_r are called **epipolar lines** — they are located at the interection between the image plane and the plane formed by the points P, O_l & O_r. Each point P will have an epipolar line in each image plane.
- e_i & e_r are epipoles representing the intersection points of O_lO_r with the left and right image planes. They may be located outside the actual images.

To estimate the epipolar geometry, determine the mapping between corresponding points in the two images.

$$p_r = Rp_l + t$$
$$O_lO_r = t$$

The left and right images are connected by means of a matrix representing rotation $\mathbf{R}$ in the plane PO_lO_r and translation $\mathbf{t}$.

Epipolar geometry depends only on the cameras' internal parameters and relative pose.

$$\text{If } O_lO_r, P_rP, P_lP \text{ are coplanar } \implies \mathbf{p}_l \cdot [\mathbf{t} \times (\mathbf{R}\mathbf{p}_r)] = 0$$

This results in the Fundamental matrix.

$$\mathbf{p}_r^T\mathbf{E}\mathbf{p}_l \text{ where:}$$
$$E = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \mathbf{R}$$
$$rank(\mathbf{E}) = 2$$

We have a well defined geometry linking the epipolar vectors of right and left images by means of matrix $\mathbf{E}$.

2 Image Rectification

Image rectification is implemented by multiplying the image matrices with a geometric transformation matrix. Epipolar lines become parallel following image rectification.

3 Depth Maps

Depth maps represent the distance from the image plane to the actual 3D scene. In computer vision depth maps are estimated from the disparity between pairs of images. If images are aligned, the disparity represents a displacement along the epipolar lines.

In computer graphics, the depth map (also called *Z-Buffer*) is calculated as the distance from the 3D scene to the image plane.

3.1 Depth Maps for Photorealistic Effects

Depth maps can be used to apply effects to specific objects/regions of the image. For example, portrait modes can use depth to apply a blurring effect to everything in an image except for a face.

3.2 Estimating Depth Maps

Depth maps can be estimated from stereo images based on fields of disparities. Each disparity represents the correspondence of a block of pixels from one image into the other. Objects which are far away will be in identical places in both images (disparity is 0). Objects which are near the camera will have a displacement according to their depth.

3.3 Block Matching

Comparing the two images and finding the disparities is done using a method called Block matching. Block matching is done using the following procedure:

1. Calculate differences between the block of pixels, pixel-by-pixel, from left image and all blocks from a search region in the right image.
2. The search region is centred at the location of the original block of pixels.
3. Choose the block of pixels from the second image which is the most similar with the orignial block and define a vector as the difference of their locations.

The size of the block of pixels $B_x \times B_y$ should not be too small (which may lead to confusion) or too large (requiring large computational complexity and low disparity resolution). Block matching works well when we have well defined features or non-regular textures. Block matching does not work well where we have regions of constant colour or regular textures.

3.4 Estimating Depth Maps ii

When estimating the depth maps from real images, local errors emerge due to:

- Left and right images are affected by the noise of the image sensors and changes in light.

- Results in wrong matches and consequently in errrs in the depth map.

3.5 3D from Multiple Views

Several images are provided from various angles and the 3D scene is reconstructed as made up of voxels. Surfaces are visible only from some cameras. For this process, the cameras have to be calibrated. Usually, this calibration is done using chessboard patterns.

4 Space Carving

The 3D scene is formed using space carving. Space carving starts with a cube in the scene. If there is no projection onto any of the images for that voxel, then it is carved away.Otherwise — get all pixels where it projects. Use all of the images to estimate the colour of the voxel from the pixels' colour.

When a voxel is seen from several images a statistical estimation takes place (averaging or median statistics).

Floating voxels and uneven surfaces emerge due to wrong correspondences and are caused by:

- Confusions in colours due to the illumination and noise
- Errors in cameras' geometry due to wrong camera calibration.

5 Conclusions

- Stereo vision
- Epipolar geometry
- Image rectification
- Depth maps
- Multi-view vision for 3D scene estimation
- Space carving method

Lecture 8: Colour Perception

Adam Hawley

December 26, 2018

Contents

1 The Light Spectrum

1.1 Visible Light

Radiation in nature is characterised by wavelength and frequency. Visible light has a very narrow wavelength range of 350nm - 780nm (neighbouring to ultraviolet and infrared respectively).

1.2 Visualising Other Spectra

In order to visualise other spectra, we can use the appropriate sensors and lenses. For example, visualising infrared radiation has many uses including security and medicine.

Most of the time, we use *true colour* to represent radiation as it is what we use in photographs. However, when visualising other spectra which are not within the reach of true colour, one can use *pseudocolour* or *greylevel mapping* to represent the radiation.

- **Pseudocolour** is when the image uses colours re-assigned according to an LUT or function.
- Similarly, **greylevel mapping** represents the radiation using a function however the product of this function represents a grey level value and no colours are involved.

2 Hyperspectral Images

2.1 What Are Hyperspectral Images?

A *hyperspectral image* associates a vector of various wavelengths to a pixel. This results in volumetric images where one of the dimensions is the wavelength (spectral dimension). The information in each pixel depends on the capacity of the corresponding object surface to absorb and reflect radiation of that specific wavelength.

Hyperspectral imaging collects and processes information from a large range of wavelengths using specific sensors.

2.2 Appplications

Hyperspectral imaging can be used for a number of things:

- Identification of materials by measuring their spectral absorption and reflectance.
- Discovery of natural resources by using ground-penetrating images taken from aeroplanes and satellites.
- Discovery of concealed objects

- Assessing the quality of fruit and vegetables

3 Illumination

Light sources are characterised by their own spectral range $[\lambda_1 - \lambda_2]$. The spectral range interacts with scene colours which means that the output in the image depends on the light spectral range of the light source. For example, consider the difference between some photos with and without a flash or at different times of day.

4 CIE Colour Model

4.1 Outlining the CIE

Commision Internatinal de l'Eclairage (CIE). Three standard primaries x, y, z for representing all colours perceived by humans.

Where **C** is a colour to be matched:

$$\mathbf{C} = X.\mathbf{X} + Y.\mathbf{Y} + Z.\mathbf{Z}$$

This can be thought of as a 3D space. Chromacity values depend on dominant wavelength and saturation (not the luminance) through normalising (X, Y, Z):

$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = \frac{Z}{X+Y+Z}$$

Since $x + y + z = 1$ we only need (x, y) to specify the chromactiy value, if we want the luminance information then we just need (x, y) and Y since:

$$X = \frac{x}{y}Y \qquad Z = \frac{1-x-y}{y}Y$$

Aspects of the chromacity diagram:

- Interior corresponds to visible colours
- Pure colours appear along the curve
- **Illuminant C** is considered to be 'standard daylight'
- Luminance dependent colours do not appear (e.g brown)

4.2 Selecting Colours Using the CIE

The diagram allows us to determine the dominant wavelength and *excitiation purity* of light:

1. Match colour A using the 3 CIE primaries

2. Plot the position in the CIE chromacity diagram
3. The mix of two colours lie on a straight line between them

$$\begin{aligned}\text{dominant wavelength} &= B\\ \text{excitation purity} &= \frac{|AC|}{|BC|} \times 100\%\end{aligned}$$

Complementay are those that can be mixed to yield white light. Colours which have no dominant wave-length are referred to as non-spectral.

Lecture 9: Colour Image Representation

Adam Hawley

December 26, 2018

Contents

1 Colour Image Pipeline

- Bayer colour filter arrays — mosaics of colour
- Coordinated colour temperature transform (CCT) — colours are adjusted based on light source illiminating the scene — based on tables of chromacity for standard illumination
- Colour gamut – depends on a specific device
- White balance adjustment
- Demosaicing — interpolates the R,G,B components producing the full colour image

1.1 Bayer Mosaics

Bayer pattern sensors are composed of colour filtered sensors in a specific layout, using two green sensors for each red and blue sensor.

1.2 Demosaicing

Demosaicing is the process of turning the sensor output with 3 separate colour channels into one pixel per three colour values. I.e going from 2 greens, 1 red and 1 blue to one pixel which represents them all.

1.3 Colour Matching

Colour matching functions work by combining red, green and blue to define the colour accuracy. A particular colour (spectral wavelength) can be matched to filtered measurements of sensor values. The matching is performed as relative values of colour primaries. A different sensor could provide different colour values.

1.4 Colour Gamut

The colour gamut represents the colours which are reproduced by a specific media (the rest of the colours are clipped). The gamut restrics which colours can be visualised. Graphics artists should consider the display media gamut for the colours of his graphics.

1.5 CIE

The CIE chromacity diagram is obtained based on their perception. Hue is defined on the border. Colour saturation decreases as you go towards the centre (location of white).

2 Colour Systems

2.1 RGB

The 3 types of cone in the human eyes are sensitive to 3 wavelengths: 630nm (red), 530nm (green) and 450nm (blue).

$$C_\lambda = R\mathbf{R} + G\mathbf{G} + B\mathbf{B}$$

The values for **R,G,B** are defined by:

- Standard NTSC phosphor
- Monitor/phosphur manufacturers

For two monitors 1 and 2 with different gamuts we can compute:

$$\begin{bmatrix} Y \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Y_r & Y_g & Y_b \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.2 YIQ and CMY

2.2.1 YIQ

Composite signal for TV defined by NTSC. This colour system is based on CIE. The Y parameter carries luminance (similar to the CIE). I and Q contribute to hue and purity.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.2.2 CMY

Subtractive colour model used in printing. Cyan ink subtracts red component of light, magenta subtracts green component and yellow subtracts the blue component of light. The printing process uses 3 or 4 dots. (CMYK)

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.3 3D Polar Coordinates

These spaces use a cylindrical (3D polar) coordinate system to encode the following three psycho-visual coordinates:

- Hue (Dominant Colour Seen)
    - Wavelength of the pure colour observed in the signal.

  - Distinguishes red, yellow, green etc.
  - More than 400 hues can be seen by the human eye.

- Saturation (Degree of Dilution)
  - Inverse of the quantity of “white” present in the signal. A pure colour has 100% saturation, the white and grey have 0% saturation.
  - Distinguishes red from pink.
  - About 20 saturation levels are visible per hue.

- Brightness
  - Amount of light emitted.
  - Distinguishes the greylevels.
  - The human eye perceives about 100 levels.

2.4 Munsel

The Munsel colour system is a cylindrically mapped colour system.

- Hue: 100 Equally spaced hues around circle
- Saturation: Starts at 0 on the centre line and increases to 10-18 depending on the hue
- Brightness: 0 for black up to 10 for white

2.5 HSV

The HSV colour system is the most intuitive model for users.

- H = Hue (0.0°-360.0°)
- S = Saturation (Ratio from 0.0-1.0)
- V = Value (Intensity)(0.0-1.0)

The colour system is based on a hexcone shape. When S=1 and V=1 “akin to artists pure pigment”:

- Hue: Change H
- Tints (Adding white): Decrease S
- Tone (Adding black): Decrease S and V
- Shade: S=1, decrease V

2.6 HSL

- Hue
- Saturation
- Luminance

The HSL colour system uses a double cone representation. Hue is represented $0°$-$360°$. Saturation is the value 0-1 out horizontally from the centre. Luminance has a 0-1 value out vertically from the bottom.

3 Colour Image Histograms

For the HSL colour system, colour image histograms are generated as follows:

- The luminance is quantised into N+1 levels. (E.g 256)
- For each level (bin)

4 Colour Representation Characteristics

- **RGB**: Most widely used because of its simplicity.
- **HSV**: Most intuitive to a computer graphics artist.
- **CIE Lab**: Most accurate colour system for the human eye.
- **YIQ**: Best representation of image grey-level is the Y component.

5 Conclusions

- Colour image acquisition
- Bayer filters
- Colour gamuts
- RGB/CMY/YIQ colour spaces
- Munsel/HSV/HSL colour spaces
- Colour image histograms

Lecture 10: Light & Reflectance

Adam Hawley

December 10, 2018

Contents

1 Light and Reflectance

Images are produced by light falling on a camera sensor:

- The light must come from somewhere
- It may be reflected from surfaces before it reaches the camera

Hence we need models of:

- Light sources
- Reflectance of light from surfaces

Illuminating a surface gives rise to shading — very important for realism and the perception of 3D shape.

2 Importance of Shading

We perceive curved surfaces through their shading patterns. Shading arises when we add a light source to our scene and employ a reflectance model.

3 Image Formation Process

Factors that affect the amount of light reflected towards us:

- Object shape and in particular the angle of the surface; if the surface is perpendicular to the light from the source it is more likely to reflect light.
- Different materials; some materials can give very different reflections of light such as a snooker ball which gives one small concentration of bright reflection compared with velvet which gives a much more even amount of light reflected in a bigger region.
- Illumination conditions
- Viewpoint/direction, camera parameters.

The surface "reflectance properties" of an object are described by a **reflectance model** (or illumination model).

4 Distant Vs. Local Illumination

4.1 Distant Illumination

Light sources which are very far away (e.g the sun) are often assumed to be at an infinite distance. This allows for two simplifications:

- Direct constant within scene
- No attenuation

4.2 Local Illumination

Light sources situated within the scene (local) are more complex:

- Direction varies within the scene
- Objects closer to the source are illuminated more strongly than those futher away (there is greater ***light source attenuation***)

5 Light Source Attenuation

The physically correct model for attenuation of a light source follows the classical one over distance squared form:

$$f_{att} = \frac{1}{d_L^2}$$

Where d_L is the distance to the light source.

Even though this is physically correct, this sometimes gives too severe a fall off to be used in graphics. For this reason, alternative functions are sometimes used. For example:

$$f_{att} = \frac{1}{d_L}$$

6 Point Sources and Spotlights

The simplest light source type is a **point source**. Characterised by a 3D location (local) or 3D direction vector (distant) and an intensity or colour. Sometimes the intensity of a distant light source is encoded in the length of the direction vector.

A **spotlight** is a point source that only emits light in a constrained cone of angles. For example: studio “barn door” type spotlights.

7 Ambient & Distributed Sources

Ambient light is light that arrives equally from all directions. This makes it a good model for a cloudy day. It also leads to very *flat* images which lack shading.

Distributed sources emit light from an area. This means that regions may be illuminated by part of the source. It leads to soft shadows: penumbra around shadowed regions.

8 Environment Maps

In reality, light arrives at a point from all directions. One useful approximation iss a spherical **environment map**. Each pixel is like a distant point source with its own direction and colour.

9 The Reflectance Equation

See slides.

10 Local Vs. Global Reflectance

In practice, light bounces off surfaces onto other surfaces (perhaps many times) before reaching our eye. If it didn't, then any surface not directly illuminated would be invisible. Light reflecting from one surface to another is called **interreflection**. In this lecture, we consider only **local models of reflectance**. In this case, we treat each point separately and only consider illumination coming directly from a light source.

11 Reflectance Models

Designing realistic reflectance models is a large part of current computer graphics research. Many materials have complex reflectance properties:

- Human skin
- Fur
- Velvet
- Hair
- Brushed metals
- The underside of a CD!

A reflectance model describes what happens when light from a certain direction strikes a surface. When this happens light may be:

- Absorbed (As on a **Translucent Surface**)
- Scattered (As on a **Diffuse Surface**)
- Reflected (As on a **Specular Surface**)
- Transmitted into the surface

Or any combination of the above.

Almost all surface reflectance models are of the form:

$$I = f(n, s, v, P)$$

Where n, s, v are the three unit vectors describing direction: **n** - surface normal, **s** - light source & **v** - viewer. P generalises some other parmeters describing the material properties of the surface.

11.1 Ambient Reflection Model

The simplest reflectance model assumes completely ambient illumination:

$$I = L_a k_a$$

Where L_a is the strength of the light source and k_a is the ambient coefficient $(0 < k_a \leq 1)$. k_a is our first example of a material property. In genereal, we will have a value for each of the RGB channels.

This simple ambient model is not realistic. It is generally added to models because graphics programmers noticed their models looked too dark otherwise and the shading drop off at the edge of objects was too drastic. It ignores the fact that parts of an object may occlude others, reducing the amount of ambient light that reaches the surface.

(We will see a better model for ambient reflectance later)

11.2 The Lambertian Reflection Model

The simples model which accounts for changes in intensity due to changes in surfaqce orientatin is the **Lambertian Model**. This is based on the idea of a **perfecttly diffuse reflector**. The assumption is that if the surface is macroscopically rough, light will be scattered equally in all directions. Similarly, it may be that the light which enters the surface is scattered internally and re-emerges completely diffused. This model is good for materials like paper and chalk.

Where the angle between surface normal and light source, θ_i, is the **angle of incidence**, the Lambertian intensity is simply:

$$I = L_d k_d \cos(\theta_i)$$

Or in terms of vectors:

$$I = L_d k_d (\mathbf{n} \cdot \mathbf{s})$$

Note: there is no $\mathbf{v}$ since diffuse reflection is independant of viewing direction. In general, we assume that the light source is distant and hence that L is constant over the entire image.

11.3 Specular Reflection i: The Phong Model

The Lambertian model gives a dull, matte appearance. Many real surfaces are macroscopically smooth and appear shiny. Reflection from these sorts of surfaces is known as specular reflectance. A perfect specualor reflector only reflects light in the direction $\mathbf{r}$. Imperfect specular reflectors reflect light into a range of directions around $\mathbf{r}$. So strength of specular reflectance is determined by andle $\mathbf{r}$ and $\mathbf{v}$. $\mathbf{r}$ is the reflection of $\mathbf{s}$ about $\mathbf{n}$. So $\mathbf{r}$, $\mathbf{s}$ and $\mathbf{n}$ lie in the same plane. $\mathbf{r}$ can be found from $\mathbf{s}$ and $\mathbf{n}$:

$$\mathbf{r} = 2(\mathbf{n} \cdot \mathbf{s})\mathbf{n}) - \mathbf{s}$$

The Phong specular model is a simple phenomenological model of specular reflectance for non-perfect reflectors. The maximum reflectance is observed when $\theta_{spec} = 0$ where θ_{spec} is the angle between **r** and **v**. Rapid fall-off is determined by the **shininess coefficient**, η. The specular term is therefore given by:

$$I = L_s k_s \cos^{\eta} \theta_{spec}$$
$$I = L_s k_s (\mathbf{r} \cdot \mathbf{v})^{\eta}$$

11.4 Specular Reflection ii: The Blinn-Phong Model

A problem with computing the perfect reflection direction, **r**, is that **n** is different for every point on the surface, so we must recompute **r** for every polygon — very slow. There is a common simplification known as the ***Blinn-Phong Model***. Instead of **r**, use aa vector halfway between the viewer and the light source. Since we assume **s** is distant, it is fixed over the image so we only need to calculate **h** once per image:

$$\mathbf{h} = \frac{\mathbf{s} + \mathbf{v}}{\|\mathbf{s} + \mathbf{v}\|}$$

If **n**=**h**, we have perfect specular direction.

11.5 The Complete Phong Model

The full Phong model is simply the sum of the ambient, diffuse and specular terms, summing over all lights:

$$I = k_a L_a + \sum_{lights} (k_d(\mathbf{n}{\cdot}\mathbf{s})L_d + k_s(\mathbf{n}{\cdot}\mathbf{h})^{\eta} L_s)$$

12 Colour in Reflectance Models

In all of the reflectance model equations given so far, material properties (ambient, diffuse, specular coefficients and shininess) are **wavelength dependent**. The strength of the light source is also wavelength dependent (i.e light source has a colour). Some of the latest reflectance models evaluate intensity over the whole visible spectrum. In practice, we provide RGB values for light sources and ambient, diffuse and specular coefficients. Often:

- Ambient RGB = Diffuse RGB (Object Colour)
- Specular RGB = White

13 Summary

What you should know:

- Types of light source
- The rendering equation
- Local reflectance models
- The Phong model: Ambient, diffuse and specular reflectance

Lecture 11: Shape-from-Shading

Adam Hawley

December 13, 2018

Contents

1 Human Shape-from-Shading

Humans can perform shape-from-shading well. They make assumptions to help solve the problem.

Shape-from-shading is also a “top down” process. High level information about the scene is incorporated into the shape estimation process. Images are interpreted to be consistent with our prior knowledge of shapes.

2 The Geometry of Surface Reflectance

Remember: Surface reflectance is described in terms of 3 unit vectors, describing the direction of the surface normal, viewer and light source. This means that intensity (I) is a function of the three unit vectors:

$$I = f(\mathbf{n},\mathbf{s},\mathbf{v},\mathbf{P})$$

P here refers to the material properties of the surface.

3 Shape-from-Shading

Shape-from-shading aims to estimate a surface normal for each pixel in an image, a depth map can subsequently be estimated from the normals.

4 Finding Depth From Illumination

4.1 Unknown Illumination: The Bas Relief Ambiguity

With different models, the same image can be created by moving the angle of the light source. This means that we cannot be sure of the depth.

4.2 The Lambertian Reflection Model i

Remember: the Lambertian model describes perfectly diffuse reflectance. It assumes that light is scattered equally in all directions. Intensity is given by cosine of angle of incidence:

$$I = L_d k_d \cos\theta_i$$
$$I = L_d k_d (\mathbf{n} \cdot \mathbf{s})$$

k_d is the **diffuse coefficient** where $0 < k_d \leq 1$ representing the proportion of light that is diffusely reflected by the surface (the rest is absorbed).

In a greyscale image we have one intensity value for each pixel. A surface normal is a 3D vector, so it is described by 3 values (x, y and z). But its length is fixed to one. Hence, it has two degrees of freedom (the two angles of a spherical coordinate representation). This means that we are left trying to recover two unknowns from one measurement and hence the problem is ill-posed (under constrained).

To try and get around this issue we can make simplifications by assuming:

- Diffuse coefficient (k_d) = 1 — A perfect white diffuse reflector
- Light source direction is known and light source intensity (L_d) = 1

4.3 The Convex-Concave Ambiguity

Even with known light source direction, ambiguity remains.

SEE SLIDES FOR EXAMPLE

4.4 The Lambertian Reflection Model ii

Under the previously outlined assumptions we have:

$$I = \cos \theta_i$$

So we can recover the original angle between the surface normal and the light source:

$$\theta_i = \arccos I$$

In other words, the surface normal must lie on a cone whose opening angle is determined by the image brightness. The problem is transformed to choosing a position on the cone for the normal at each pixel.

4.5 Constraints

Surface normals at adjecent pixels are not independent. They must correspond to a real surface. Surface normals which satisfy this constraint are said to be *integrable.*

Other constraints often used include the following:

- **Smoothness** — surface normal direction should vary slowly across the surface, with no discontinuities.
- **Global convexity** — prefer convex interpretation to concave.

4.6 The Worthington & Hancock Algorithm

The Worthington and Hancock alogrithm uses the cone constraint in an iterative framework.

- Initialise normals to on-cone position.
- Enforce additional constraint (e.g apply smoothing) resulting in off-cone normals.
- Rotate normals back to closest on-cone positions.
- Repeat until convergence.

4.6.1 Initialisation

Worthington and Hancock use an itialisation in which the normals line up with the local negative image gradient. This has the effect of picking the darkest point on the cone. This idea is consistent with an assumption of convexity.

4.7 Deep Learning

It should be noted that deep research into the application of deep learning is being undertaken. Neural networks have produced models which can compute depth maps in milliseconds and using fewer assumptions than we are discussing. For example they are able to estimate the lighting intensity.

4.8 Shape-from-Shading Images Conclusions

- Shape-from-shading is still an unsolved problem.
- Fails on real-world images.
- Even on synthetic images, surface can be highly distorted.

A more practical alternative is to use **photometric stereo**. This involves taking multiple images from the same viewpoint but with varying illumination. Each image provides a partial constraint on the surface normal direction. They can all then be combined.

4.9 Photometric Stereo

Two light source directions gives us two cones and two points of intersection. Two points of intersection means that there is two possible normal directions. Three light sources uniquely determines the normal. However, because of noise, this may not be an exact solution. This means that in reality it is best to use more than three light sources and not compute it exactly but use something similar to a least squares solution.

One image gives the following equation for each pixel from the Lambertian model equation:

$$I = L_d k_d (n_x s_x + n_y s_y + n_z s_z)$$

We can factor the diffuse coefficient into the normal since these are our unknowns and light source intensity into the light source vector since we know both of these:

$$I = (k_d \mathbf{n}) \cdot (L_d \mathbf{s})$$

In other words, we have a linear equation in the scaled surface normal. The three unknowns are:

$$N = [\bar{n}_x \bar{n}_y \bar{n}_z]^T$$

Hence, we need at least three images. With three images, we have 3 simultaneous equations for each pixel:

$$\begin{aligned} I_1 &= \bar{s}_{1x}\bar{n}_x + \bar{s}_{1y}\bar{n}_y + \bar{s}_{1z}\bar{n}_z \\ I_2 &= \bar{s}_{2x}\bar{n}_x + \bar{s}_{2y}\bar{n}_y + \bar{s}_{2z}\bar{n}_z \\ I_3 &= \bar{s}_{3x}\bar{n}_x + \bar{s}_{3y}\bar{n}_y + \bar{s}_{3z}\bar{n}_z \end{aligned}$$

Subscripts represent image number, bar means the vector has been scaled by diffuse coefficient or light source intensity. I and s quantities are known, n are unknown. One can solve for n by rearranging and combining equations solving for:

$$N = [\bar{n}_x\bar{n}_y\bar{n}_z]^T = [k_d n_x k_d n_y k_d n_z]^T$$

We form a matrix of the scaled light source vectors. Assuming we have k images, this will be of dimension k times 3:

$$S = \begin{bmatrix} s_1^T \\ \vdots \\ s_k^T \end{bmatrix}$$

We form a vector of the image intensities over the k images:

$$I = \begin{bmatrix} I_1 \\ \vdots \\ I_k \end{bmatrix}$$

Then we minimise the following quantity:

$$||\mathbf{I} - \mathbf{SN}||^2$$

The vecot of unknowns $\mathbf{N}$ contains the surface normal scaled by the diffuse coefficients. We can solve for $\mathbf{N}$ using the *pseudoinverse* of $\mathbf{S}$:

$$\mathbf{N} = \mathbf{S}^+\mathbf{I} = (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T\mathbf{I}$$

For both the k image and 3 image version, we can separate out the normal and diffuse coefficient:

$$k_d = ||\mathbf{N}|| \qquad \mathbf{n} = \frac{\mathbf{N}}{||\mathbf{N}||}$$

4.10 Summary of Photometric Stereo

Weaknesses:

- Strong assumptions:
  - Lambertian Reflectance, specularities not modelled

  - Single point source lights (known directions)
  - No shadowing
- Estimates normals, not full 3D surface
- Photometric stereo requires multipple images with no motion

Strengths:

- Simplicity (linear model of image formation, geometric cone interpretation)
- Dense shape estimate (normal for every pixel in the image)
- Photometric stereo estimates both albedo and shappe, allowing **relighting** (rendering with novel light source directions)

5 Conclusions

What you should know:

- What shape from shading is
- Lambertian reflectance = cone constraint
- Worthington and Hancock algorithm
- Photometric stereo

Lecture 12: Image Filtering

Adam Hawley

December 17, 2018

Recommended reading:

- Forsyth & Ponce Chapter 7

Contents

1 Intoduction: Images as Functions

An image can be seen as a function which is discretised and quantised to [0-255] levels. The function depends on the scene represented, light properties, image acquisition and so on.

2 Image Subsampling

Image subsampling consists of removing rows and columns from the image.

Noise is often lost through subsampling because noise is pixel values that are vastly different to their neighbours which do not represent small details. These pixels are removed through subsampling. This method is used commonly in deep learning and neural networks because large images require high levels of processing power and time.

When you zoom in on subsampled images, blurring is created, features related to noise and small details are lost. This is because when you zoom in, pixels are duplicated to fill the new pixels which come from the increase in size.

3 Image Interpolation

3.1 What is Image Interpolation?

A better method for restoring the original image after subsampling is to use **image interpolation**. **Image interpolation** uses neighbouring pixels to estimate the original pixel value between the neighbours.

3.2 How is it done?

The simples example would be to replicate a neighbouring pixel. However, this is the same as zooming in on the image. Instead it can be better to take an average of the neighbouring pixels.

- **Bilinear interpolation** considers a weighted average of a 4x4 pixel neighbourhood.
- **Bicubic interpolation** considers a weighted average of an 8x8 pixel neighbourhood.
- **Adaptive interpolation** depends on image context.

4 Image Convolution

Image convulation uses a window of a neighbourhood. Kernel $w(x, y)$ with values for each window element. Calculates the output using the following formula:

$$g(x,y) = \sum_{s=-K}^{K} \sum_{t=-K}^{K} w(s,t) f(x+s, y+t)$$

Applying this formula assigns the output to the central element of the $2K + 1 \times 2K + 1$ window. Afterwards, the centre moves to the next neighbourhood according to the following:

$$s = 1, ..., image\_width - 1$$
$$t = 1, ..., image\_height - 1$$

5 Applications of Image Convolution

- Filtering for removing noise
- Sharpening images and image features such as object edges and texture
- Producing graphical artistic effects in images for *softening* and *sharpening* images, "fire" effects and embossing images.
- Colouring images as vectors.

6 Image Filtering

Image filtering can be usually applied simply by convolution with a filter mask.

6.1 The Mean (Averaging) Filter

Consider the convolution of an image with the follwing kernal for a 3×3 window:

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

The output is the average of the 9 pixels. This results in the removal of noise but also blurring of the image.

6.2 Gaussian Filter

The Gaussian filter represents a weighted averaging filter, where the weights are higher in the central region of the window according to the Gaussian equation:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Note: in graphics, G is discretly approximated.

7 Image Noise

Noise occurs naturally in images due to the camera sensors — increase camera sensitivity (ISO) — increase noise. Noise can be removed by filtering.

$$\hat{f}(x, y) = f(x, y) + n(x, y)$$

Where $\hat{f}$ is the noisy image, f is the original image and n is the noise itselve. We can test various filters by artificially adding noise to the image and filtering it afterwards.

7.1 The Mean Filter

Images are invariably noisy — for example when increasing ISO we achieve better detail and better night images — but also it adds noise to images.

Averaging corresponds to a low pass filter and removes some of the noise by smoothing. On the other hand averaging filters lead to blurring and can remove small details. It is also very effective in temporal image sequences.

Problems with the mean filter:

- Blurs edges and details in the image
- Not effective for impule noise (*Salt-and-Pepper*)

Salt-and-Pepper Noise:
Add extreme values to the image of black (0) and white (255) according to a certain percentage.

7.2 Median Filter

The **median filter** ranks the data inside the window and picks the middle value.

Note: since the median filter simply takes the middle value of the window, it does not need to be a square image.

8 Rank Order Statistics Filters

Rank order statistics filters rank the values from the window and pick the value of a certain rank. The most common rank order statistics filters used are minimum and maximum filters — these can be used for segmenting the image. (See section 9)

9 Maximum & Minimum Filters

9.1 Maximum Filter

The max filter takes the maximum value in the neighbourhood. This has the effect of enhancing bright areas.

9.2 Minimum Filter

The min filter takes the minimum value in the neighbourhood. This has the effect of introducing darkness in the image.

10 Conclusions

What you need to know:

- Image interpolation
- Image convolution
- Image neighbourhoods
- Mean filter
- Median filter
- Gaussian and salt-and-pepper noise
- Min and max filters

Lecture 13: Drawing Lines & Detecting Edges

Adam Hawley

December 18, 2018

1 Pixels

The pixel space is a rectangle on the x, y plane, bounded by $0 \leq x \leq x_{max}$ and $0 \leq y \leq y_{max}$ Each axis is divided into an integer number of pixels, N_x and N_y. The pixels therefore have width and height:

$$W = \frac{x_{max}}{N_x}$$
$$H = \frac{y_{max}}{N_y}$$

Pixels are reffered to using integer coordinates, that either refer to the locatin of their left hand corners or their centres. Knowing the W and H values allows the pixel to be defined. Assuming that W and H are equal, the pixels are square.

2 Line Rastersation

Straight lines are described by the equation:

$$y = mx + c$$

To draw a line we have to vary x between a given range of values and compute a point y at each location according to the line equation. **Rasterisation** is the process of converting the drawing to pixels in images (also called scan conversion).

3 The Digital Differential Analyser Algorithm

The Digital Differential Analyser Algorithm (DDA) is an algorithm used to do line rasterisation. Assuming that we have two endpoints:

$$p_0 = (x_0, y_0) \qquad\qquad p_1 = (x_1, y_1)$$

The point of intersection is given by the line rule where:

$$m = \frac{y_1 - y_0}{x_1 - x_0}$$

We can consider a point (x, y) on the line and then the next point is $(x+1, y+m)$ when the line is drawn to the right.

The drawback of the DDA is that it uses folat and rounding operations for each point which require increased computational power.

4 Bresenham's Algorithm

The DDA algorithm needs to carry out a floating point addition and a rounding operaiton for every iteration. Bresenham's algorithm operates only on integers requiriing only that the start ad end points of a line segment are rounded.

Firstly assuming that $0 \leq m \leq 1$ and pixel (i, j) is the current pixel. The value of y on the line $x = i + 1$ must lie between j and $j + 1$. Because the gradient is in the range 0-45° the next pixel must be either East $(i + 1, j)$ or North East $(i + 1, j + 1)$. We should also assume that pixels are identified by their centres.

At this point, we have a decision problem, and so must find a decision variable that can be tested to determine which pixel to colour. First rewrite the equation of a straight line:

$$\begin{aligned} y &= mx{+}c \\ mx - y + c &= 0 \\ \frac{\Delta y}{\Delta x}x - y + c &= 0 \\ F(x, y) = x\Delta y - y\Delta x + B &= 0 \end{aligned}$$

The useful properties of this function are as follows:

- $F(x, y) > 0 \implies$ point is above the line
- $F(x, y) < 0 \implies$ point is below the line

The function $F(x, y)$ clearly has the properties required of a decision variable. Now all we have to do is apply it. This will involve using rational numbers in the derivation even though we were supposed to be working with integers.

Consider an arbitrary pixel (x_p, y_p) that is on a line segment. We need to test a point to determine which of the possible next 2 pixels to colour. A suitable point is $(x_p + 1, y_p + \frac{1}{2})$ this will be on the vertical boundary between pixels E and NE and at the horizontal midpoint.

If a $F(x_p + 1, y_p + \frac{1}{2}) \leq 0$ the next pixel is E. Else, the next pixel is NE. This is still under the same assumption that that $0 \leq m \leq 1$.

Examining the next step we see it is not independent of the decision we have just made.

- If E was chosen then the next test will be $F(x_p + 2, y_p + \frac{1}{2})$
- Else is NE was chosen then the next test will be $F(x_p + 2, y_p + \frac{3}{2})$.

Now let $d_n = F(x_p + 1, y + \frac{1}{2})$ and substitute the values $x + 2, y + \frac{1}{2}$ and $y + \frac{3}{2}$ back into the definition of F and we have:

- If E was chosen then the next test will be $d_{n+1} = d_n + \Delta y$
- Else if NE was chosen the next test will be $d_{n+1} = d_n + (\Delta y - \Delta x)$

5 Edge Detection

Edges can be caused by:

- Boundaries of objects
- Depth discontinuity
- Illuminating discontinuity
- Shading discontinuity
- Margins of shadows
- Textures (regions of medium or high variation)
- Noise

5.1 Image Gradient

The gradient of an image:

$$\Delta f = [\frac{\delta f}{\delta x}, \frac{\delta f}{\delta y}]$$

Edge orientation:

$$\theta = tan^{-1}(\frac{\delta f}{\delta x}/\frac{\delta f}{\delta y})$$

Edge strength:

$$\|\Delta f\| = \sqrt{(\frac{\delta f}{\delta x})^2 + (\frac{\delta f}{\delta y})^2}$$

Lecture 14: Shadows & Interpolation Shading

Adam Hawley

December 22, 2018

Contents

1 Shadowing

There are two ways a point can be in a shadow: firstly, points can be in shadow because they face away from the light source or they can be in darkness caused by another point. The first is called an self/attached shadow while the second is a cast shadow.

1.1 Self-Shadowing

How can we test for self shadows? The check is based on whether the angle between the surface normal and the light source is greater than 90°:

$$\theta_i > 90^\circ \implies \text{Self shadow}$$

Equivalently, check whether the dot product is negative:

$$\mathbf{n} \cdot \mathbf{s} < 0 \implies \text{Self shadow}$$

This is a very efficient process as it depends on local information only. Normals are already computed for the reflectance calcultion, simply check dot product.

Self shadowing can be incorporated into the Lambertian model since Lambertian shading uses the same calculation. (The dot product between the surface normal and the light source) To make use of the pre-calculated dot product, simply clamp negative values to zero:

$$I = max(0, \mathbf{n} \cdot \mathbf{s})$$

Note: we did not consider this in either the shape-from-shading or photometric stereo algorithms in the previous lecture. So self shadows are not accounted for. "Add noise to the solution"

1.2 Cast Shadows

Case shadows are much more difficult to calculate. A point can be shadowed by a distant part of the surface. (They depend on global geometry) To check whether a point is in shadow, we must consider all other points on the surface.

If we set height values for every pixel (x, y) as $z(x, y)$. Then the cast shadow test can be written as:

$$\forall d \in \mathbb{R}^+, z(x,y) + ds_z > z(x + ds_x, y + ds_y) \implies \text{Not in cast shadow}$$

For general polygonal surfaces it is more complex: For every polygon or vertex, compute equation of line to light source. Then test whether the line intersects any other polygon in the scene e.g:

- Compute plane equation for every polygon
- Compute line/plane intersection
- Test whether intersection is inside or outside extent of polygon

There are efficient ways to calculate this.

2 Flat Shading

For each planar polygon we compute the surface normal. The surface normal is constant over the polygon and used to compute a single colour for the polygon. The whole polygon is then filled with this colour. For objects which consist of only planar surfaces, it can look fine. However it generally looks bad for polygonal approximations to curved surfaces.

Flat shading has a very bad effect with curved surfaces. **Mach banding** caused by **lateral inhibiion** means that the human eye effectively has a built in edge detector. Perception of change in intensity exaggerates actual change.

To smoothly vary the colour of a polygon across the surface we compute a surface normal for each vertex. The meaning of the normal direction to a point is not well defined so there are two options:

- If the surface is described by a parametric function (e.g a sphere or cylinder), just **derive an analytical expression for the surface normal** from the function.
- If the surface is an arbitrary polygonal mesh, compute **one normal per polygon and combine** these in some way to find vertex normals.

3 Face Normals

Typically, a surface is represented by a list of vertices and a list of faces. Faces are specified such that vertex ordering is clockwise or counterclockwise when viewed from outward facing side. To compute a normal to a face, use the cross product of two edge vectors:

$$\mathbf{n} = \mathbf{e}_1 \times \mathbf{e}_2$$

Remember: The length of $\mathbf{n}$ gives twice the area of the triangle. To find $\mathbf{n}$ as a unit vector simply divide it by its magnitude.

4 Vertex Normals

A vertex normal is a combination of the surface normals of the facets adjacent to the vertex.

To find a vertex normal, one can find the average of the surrounding facet normals. Often a weighted average is used — weights are areas of facets. Normals of larger facets contribute more to the vertex normal. If we use the unnormalised cross products of facet edges, we get this weighting for free. This method is fine for most purposes but there are more complex methods justified by differential geometry.

5 Interpolation Shading

There are two main approaches to use vertex normals for interpolation shading:

- **Gouraud**: calculate a colour for each vertex, **interpolate colours** along the polygon edge, then interpolate along scanlines within polygons.
- **Phong**: interpolate the vertex normals along the polygon edge, then **interpolate normals**

5.1 Drawbacks of Interpolation Shading

There are two main drawbacks of interpolation shading:

- The polygonss are still flat and so the silhouette of curved objects can visibly consist of straight lines.
- Shading is carried out after the projection transformation is applied. So if a perspective transformation is used the non-linear transformation of the Z axis may produce odd effects.

Both of these effects are reduced by using smaller polygons (i.e a higher resolution mesh).

Other possible problems include strange effects if the normals are set up wrongly. Mach banding may still be visible — although colours along polygon edge now match, the rate of change of colour may change drastically giving the appearance of an edge.

5.2 Gouraud Shading

Summary:

1. Each vertex has a surface normal and material properties, pass the reflectance model to compute a colour.
2. The colouring along the edge of each polygon is interpolated along the edge.
3. When the polygon is scan converted, interpolate between colours at edge along scanline.

5.2.1 Drawbacks of Gouraud Shading

The weakness of Gouraud shading is that it assumes the intensity changes smoothly within a polygon. Specular reflections cause large changes in intensity over a small distance. If the size of the specularity is small compared to the size of the polygon, Gouraud shading will miss it — this results in strange behaviour.

5.3 Phong Shading

In Phong shading we **interpolate surface normals**. The reflectance model will be evaluated independently for each pixel.

1. Interpolate normals along the polygon edges
2. Interpolate normals along scanlines, giving a normal for each pixel.
3. Calculate the colour intensity for each pixel.

5.3.1 Advantages & Disadvantages of Phong Shading

Advantages:

- Correct specular highlight
- Reduced Mach banding problems

Disadvantages:

- Expensive to compute

In modern GPUs something very similar to this happens in pixel or fragment shaders.

6 Conclusions

What you should know:

- Shadows: Cast vs. Self
- Face/Vertex normals and how to calculate them
- Purpose of interpolation shading
- Gouraud vs. Phong Shading

Lecture 15: Global Illumination & Image-Based Lighting

Adam Hawley

December 23, 2018

Contents

1 Introduction

Up until this lecture, we have only been modelling illumination in purely local terms. Reflected intensity is a function solely of light arriving at a point directly from light sources. **Global illumination** aims to model all illumination which has arrived at a point including light which has been reflected or refracted by other objects.

2 Global Interactions

The general approach to considering global interactions:

- Start with the light source and follow every ray of light as it travels through the environment.

Stopping under any of the following conditions:

- Light hits the eye point
- Light travels out of the environment
- Light has its energy reduced under an admissable limit due to absorption in objects.

This process is known as ***photon mapping***. This is computationally very expensive Certainly not practical for real time graphics but is used offline.

3 Types of Interactions

Diffusely radiated light reflected off another diffuse surface is called **Radiosity**. Specular radiated light reflected off another specular surface is called **Ray Tracing**.

4 Ray Tracing

4.1 Whitted Ray Tracing

The following outlines Whitted ray tracing:

- Traces light rays in the reverse direction of propogation from the eye back into the scene towards the light source.
- Spawn reflection rays and refraction rays for every hit point (recursive calculation).

Global illumination model (specular) and a local model. Considers diffuse-reflection interactions but not diffuse-diffuse interactions.

Whitted's illumination equation:

$$I = k_a L_a + \sum_{i=1}^{m} f_{att_i} L_i [(k_d(\mathbf{n} \cdot \mathbf{s}_i) + k_s(\mathbf{n} \cdot \mathbf{h}_i)^\eta)] + k_r I_r + k_t I_t$$

This equation consists of the standard Blinn-Phong model combined with two extra terms; describing reflected light and transmitted light. These two additional terms contain the recursive elements.

4.2 Problems with Ray Tracing

Ray tracing, in its current state has several disadvantages:

- Shadow rays usually not refracted
- Numerical precision — false shadows
- Simplistic model of ambient light
- Expensive to compute — lots of ray-surface intersection computations

Note: issues with numerical precision can be alleviated using multiple rays per pixel or sub-pixel jitter — gives soft shadows.

5 Image-Based Illumination

Ray tracing allows us to more realistically model how light is transferred around a scene. This gives a big improvement in realism. But modelling of the scene (objects+illumination) is a very laborious process. Humans perceive material properties and shape better under **natural illumination**. The easiest way to measure natural illumination is to take a photograph using a mirrored ball (light probe). The mirrored ball measures incident illuminatin from every direction. The next step is to resample the picture to "equirectangular mapping". This is called an **environment map**. The mapping has two dimensions: elevation (y) and azimuth ($\sim x$)

To convert (x, y) coordinates to angles:

$$\theta(x) = \pi \left(\frac{2x}{x_{max}} - 1 \right)$$
$$\phi(y) = \frac{y\pi}{y_{max}}$$

To convert angles to a direction (unit vector):

$$\mathbf{s}(x, y) = \begin{bmatrix} \sin(\phi(y)) \sin(\phi(x)) \\ \cos(\phi(y)) \\ \sin(\phi(y)) \cos(\phi(x)) \end{bmatrix}$$

Each pixel (x,y) in an environment map $I(x, y)$ corresponds to a light source with direction $\mathbf{s}$(x,y) and colour $I(x, y)$ To render with an environment map, we just sum over light sources (every pixel in environment map image!) as in Lecture 10.

6 Environment/Reflection Mapping

Very cheap approximation to appearance of mirror specular object under environment illumination. Reflect viewing direction about surface normal giving specular lighting direction:

$$\mathbf{s} = 2(\mathbf{n} \cdot \mathbf{v})\mathbf{n} - \mathbf{v}$$

Use this to look up colour in environment map and copy to pixel. Environment map stored as texture on reference object such as sphere or cube. Possible in real time at very low computational cost.

7 Occlusions

A point on a surface will not necessarily be able to see all of the environment. i.e Point may be shadowed from certain sources. Half will be excluded because of self shadowing. Only those in “upper hemisphere” considered.

To handle occlusions exactly, every direction from every surface point needs checking — like ray tracing with thousands of light sources.

Alternatively: **precompute** proportion of hemisphere that is occluded and use with environent lighting. This is **ambient occlusion**.

7.1 Ambient Occlusion

- Ambient occlusion = 1: Completely unoccluded — use all sources in upper hemishphere.
- Ambient occlusion = 0: Completely occluded — no light reaches point

The **bent normal** is the average unoccluded direction.

Ambientt occlusion is exact for an environment of perfectly ambient light. It gives rise to a different kind of shading that is important in human perception.

To use ambient occlusion with general environment lighting:

1. Render surface while ignoring occlusions
2. Downweight resulting intensities by ambient occlusion amount
3. Use bent normal instead of surface normal

The bent normal is used to describe the dominant direction from which light arrives. It can be unrepresentative when there are “*blockers*” in the scene.

8 Photon Mapping

Photon mapping approximates the full rendering equation. Considers all types of interaction and can support image-based lighting. Based on the *Monte Carlo* simulation:

1. Fire photon from light source.
2. When photon strikes a surface, randomly select a behaviour according to a probabilistic model (e.g. may be absorbed, may be reflected).
3. Count number of photons that strike each pixel.

The more photons we fire, the better the approximation. Can include arbitrarily complex models of photon behaviour when striking a surface (e.g. subsurface scattering through multiple layers of a surface). Can incluse wavelength-dependent effects.

9 Conclusions

What you should know:

- What is global illumination
- Recursive ray tracing
- Image-based lighting
- Ambient occlusion