Commonality Analysis for Family of Lighting Models

Sasha Soraine

October 17, 2019

1 Revision History

Date	Version	Notes
October 1, 2019	1.0	Original draft
October 17, 2019	1.1	Updates in Response to GitHub Issues

2 Reference Material

This section records information for easy reference.

2.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
rad	angle	radian

2.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the physics and calculus notation. The symbols are listed in alphabetical order.

symbol	unit	description
$\angle\theta_i$	rad	Angle of incidence between incident ray and surface normal.
$\angle\theta_r$	rad	Angle of reflection between reflected ray and surface normal.
θ_1	rad	Angle of incidence between incident ray and surface normal in material 1.
θ_2	rad	Angle of refraction between refracted ray and surface normal in material 2.
n_1	—	Refractive index of first material.
n_2	—	Refractive index of second material.
n_i	—	Refractive index of i-th material.
$\vec{L_i}$	—	Vector form of incident ray.
$\vec{L_r}$	—	Vector form of reflected ray.
$\vec{n}$	—	Vector form of surface normal.
$\vec{N}$	—	Unit vector of surface normal, $\vec{n}$.
$\vec{U_{L_i}}$	—	Unit vector of incident ray, L_i.
I	–	Intensity (Luminance) of Light Source.

2.3 Abbreviations and Acronyms

symbol	description
A	Assumption
AS	Scope Time Assumption
AB	Build Time Assumption
AR	Run Time Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
CA	Commonality Analysis
LM	Lighting Model
Family of Lighting Models	Family of Lighting Models
T	Theoretical Model
3D	Three Dimensional
API	Application Programming Interface

Contents

3 Introduction

An important part of computer graphics is modeling the lighting of objects in a virtual environment. Understanding and modeling how light interacts with objects of different materials is necessary to provide realistic lighting to virtual environments. Having an open source lighting model library that is reliable and robust would allow for new computer graphics programmers to more efficiently create scenes.

The following section provides an overview of the Commonality Analysis for a family of lighting models. This section explains the purpose of the document, scope of the family, characteristics of the intended reader, and organization of the document.

3.1 Purpose of Document

This document describes a family of lighting models to be used for computer graphics. This document is intended to be used as a reference to be provide the necessary information to verify the family of models, and implement the different family members.

This document captures the problem domain, theoretical models used to address the problem, the commonalities and variabilities between members of the family, and the requirements common to those members. It serves as a starting point to the design and implementation of a library of lighting models, and will be referenced in the creation of a verification and validation plan.
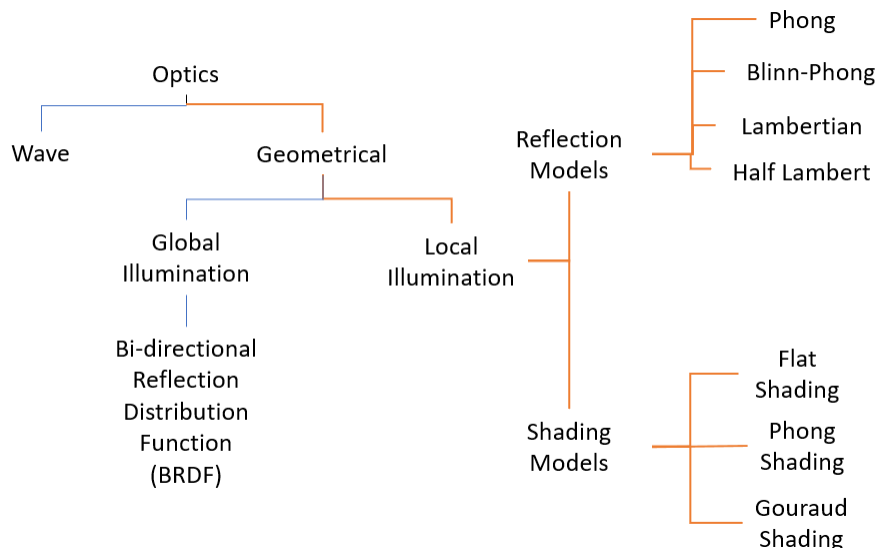
3.2 Scope of the Family

Figure 1: Problem Domain for Lighting Models of Computer Graphics.

The problem domain for lighting models in computer graphics is broad (Fig. 1). To simplify this problem we invoke assumptions AS1, AS2, AS4. This allows us to focus on the parts of the problem connected by the orange line.

The scope of the family includes geometrical optics simulation of light reflection for 3D material objects in a local illumination context.

3.3 Characteristics of Intended Reader

The intended readers of this document should have understanding of Grade 12 Physics (particularly Optics) and a undergraduate Level 2 understanding of Linear Algebra and Matrix operations.

3.4 Organization of Document

This document is organized in accordance with the CA template for scientific computing software provided by Dr. Smith for CAS 741. These templates are based on work by Smith (2006).

4 General System Description

This section identifies the interfaces between the system and its environment, describes the potential user characteristics and lists the potential system constraints.

4.1 Potential System Contexts

Figure 2 shows the high level system context. The circle represents the system user. The box is the library of lighting models system. Arrows are used to show the flow of data between the system and its environment.
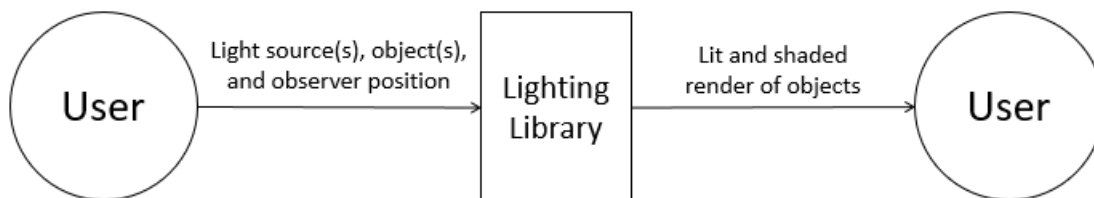

Figure 2: High level system context for Family of Lighting Models

- User Responsibilities:

  - Ensure that the problem they are looking to solve matches the assumptions made for this family.

- Provide information on the light source(s), object(s) in scene, and observer position.
- Declare shading model to use from preset options.
- Declare lighting model to use from present options.
- Ensure application programming interface use complies with the user guide.

- Family of Lighting Models Responsibilities:

  - Calculate the reflections of all light rays coming from the light source(s).
  - Determine which light rays (reflected or from source) reach the observer.
  - Render a lit environment based on selected shading and lighting model.
  - Update the calculations and render in response to changes in the input data.
  - Detect data type mismatch, such as a string of characters instead of a floating point number.

4.2 Potential User Characteristics

The end user of Family of Lighting Models should have an Computer Science/Software Engineering Undergraduate Level 3 understanding of Computer Graphics (such as through completing the SFWR ENG/CS 3GC3) and moderate experience with programming.

4.3 Potential System Constraints

There are no system constraints.

5 Commonalities

This section presents a high-level view of the problem. It captures terminology and definitions relevant to the problem, theoretical models that are common to all members of the family, goal statements of the family, data definitions that will be used to solve these problems, as well as commonalities in inputs, calculations, and outputs.

This section also provides a background overview which explains the motivation for this work.

5.1 Background Overview

Computer graphics and the techniques developed therein have become extremely useful in a variety of fields. Animation, games and media all rely on computer graphics to render semi-realistic images for their applications. As we move into an era of virtual and augmented reality we see that computer graphics has a place beyond the scope of entertainment. Training and rehabilitation simulations are extremely popular uses for computer graphics.

Modeling the interaction of light in realistic ways is difficult for computers. As such it becomes essential to have mostly accurate approximations of lights behaviours that can be used to render scenes in near real-time. On top of the need for efficient approximations, it's imperative to make the libraries and programs that handle lighting easy to use. It would create an incredibly high learning curve if users needed to understand all of the linear algebra and physics behind the approximations to mock up a scene. Existing paradigms for graphics coding offer APIs to try and make it more user friendly, but there is still a disproportionate focucs in graphics to manually fixing the scene and matrix manipulations.

The broad purpose of this work is to abstract the details of lighting in computer graphics. We aim for end-users to specify their objects and light sources, and have the system render a fully lit and shaded scene. This type of work would be useful in teaching lighting model concepts, as it would easily allow for switching between different models. More concretely the exercise of understanding the family of lighting models would allow for future students to review this creation process and have a deeper understanding of lighting in computer graphics. Outside of academia, it would similarly be useful for individuals working on small scale graphics problems, like prototyping a simulation environment. Work like this would allow them to not worry about the math involved in getting the environment lit adequately, and instead to focus on their more abstract problem.

5.2   Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements. Definitions are borrowed or synthesized from Lengyel (2003); Comninos (2005); Shreiner and Angel (2012).

Geometrical Optics: The study of lights as rays.

Ray: A view of light as a continuous beam, represented as a vector.

Reflection: The redirection caused by collision of light rays with objects.

Specular reflection: Perfect reflection of a light ray on a smooth/glossy surface. This is used to create highlights on objects, or create highly reflective objects like mirrors.

Diffuse (Lambertian) reflection: Reflected light ray is scattered in random directions due to microfacets in the surface. This type of reflection does not depend of the position of an observer.

Refraction: The redirection of light rays when passing through different materials.

Illumination: Process by which amount of light reaching a surface is determined.

Shading: methods used to determine the colour and intensity of light reflected towards viewer.

Lighting model: Method used to calculate the colour of a point on a surface. Incorporates a reflection model and a shading model.

Reflection model: How light reflects off a surface.

Shading model: How the normal vector is calculated for polygons in the scene.

Light source: The origin of light rays.

Ambient light: Light with no identifiable source or direction. It has equal intensity in every direction, and illuminates every part of an object uniformly.

Directional (or infinite) light: Light defined only by direction; light travels infinitely in that single direction with constant intensity.

Point light: Light defined only by a point; light travels uniformly in every direction from that point, with intensity decreasing with inverse square of distance from source.

Spotlight: Light defined by a point and direction; light travels infinitely in a single direction from the defined source point with intensity decreasing with inverse square of distance from source and increased radius of coverage.

Scene: A collection of objects with specific material properties.

Object: Set of polygons, edges, and vertices.

Polygon: Closed shape (with well defined interior and exterior).

Normal: A vector perpendicular to a surface, point, or pair of vectors.
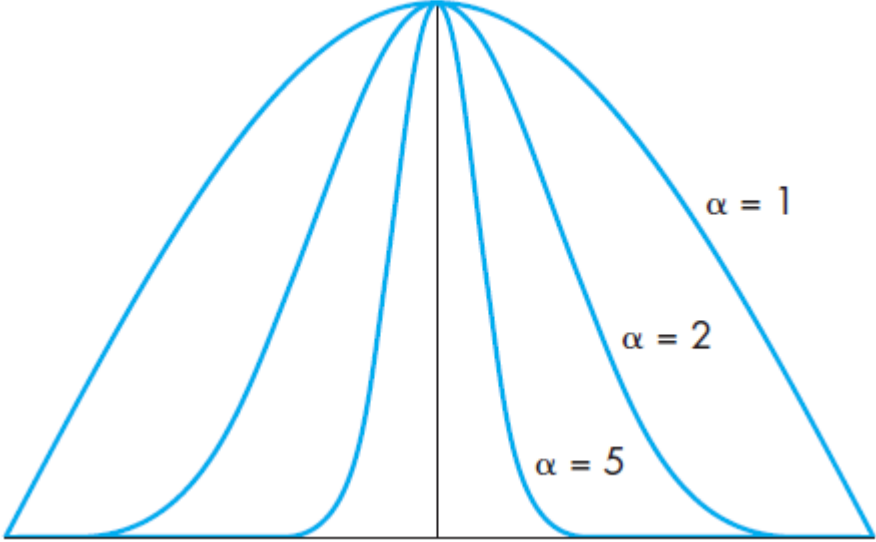
5.3   Data Definitions

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given.

Number	DD1
Label	**Dot Product of two n-dimensional vectors**
Symbol	$P \bullet Q$
SI Units	–
Equation	$P \bullet Q = \sum_1^n P_i Q_i$
Description	P and Q are two n-dimensional vectors. The dot product $(P \bullet Q)$ is the scalar sum of the products of each component. The dot product acts a measure of the difference between the directions in which P and Q point. P and Q are perpendicular when $P \bullet Q = 0$. This is used in T1 and GD1.
Sources	Lengyel (2003)
Ref. By	AB1, AB2

Number	DD2
Label	**Cross Product of two 3-dimensional vectors**
Symbol	$P \times Q$
SI Units	–
Equation	$P \times Q = \langle P_y Q_z - P_z Q_y, P_z Q_x - P_x Q_z, P_x Q_y - P_y Q_x \rangle$
Description	P and Q are two 3-dimensional vectors. The cross product $(P \times Q)$ is the unit vector normal to both P and Q. The calculation of the cross product's direction follows the *right hand rule* paradigm. This is used in T1 and GD1.
Sources	Lengyel (2003)
Ref. By	AB1, AB2

Number	DD3
Label	**Luminance for a Light Source**
Symbol	I_a
SI Units	–
Equation	$I_a = \left\{ \begin{array}{c} I_{ar} \\ I_{ag} \\ I_{ab} \end{array} \right\} = I_{L_i} \cdot k_a$
Description	I_{Li} is the intensity of the light source - it is uniform for ambient light and shouldn't rely on position. k_a is the coefficient of ambient reflection.
Sources	Shreiner and Angel (2012)
Ref. By	IM1, IM2, IM3, IM4
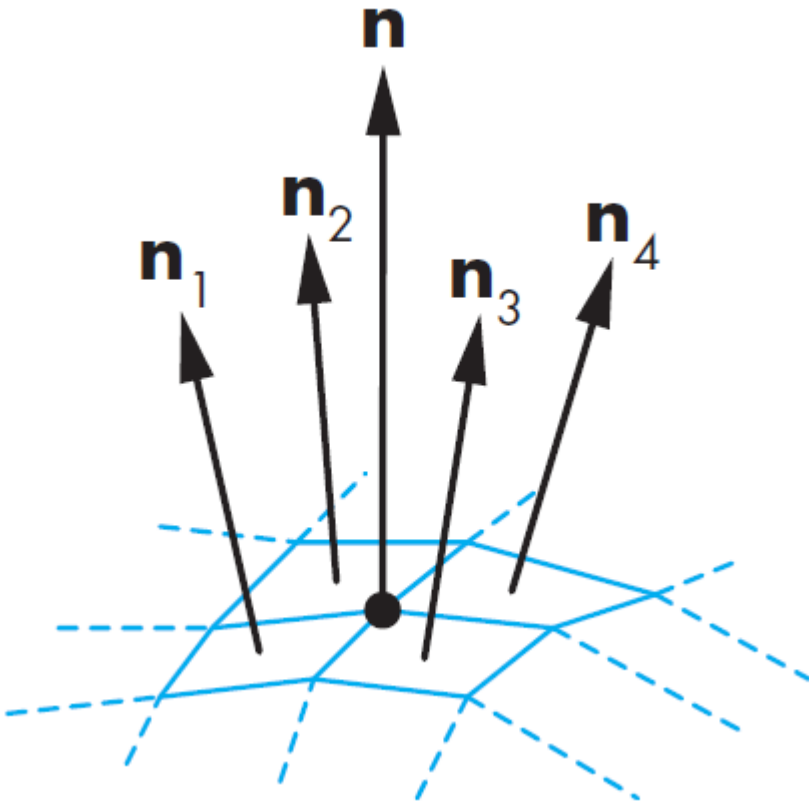
Number	DD4
Label	**Luminance for a Light Source**
Symbol	I_d
SI Units	–
Equation	$I_d = \left\{ \begin{array}{c} I_{dr} \\ I_{dg} \\ I_{db} \end{array} \right\} = k_d \cdot I_{L_i}(p)(\vec{U_i} \bullet \vec{N})L_d$
Description	p is the point where $\vec{L_i}$ hits the surface. $I_{L_i}(p)$ is the intensity of L_i at point p. k_d is the coefficient of diffuse reflection. $\vec{U_i}$ is the unit vector of L_i. $\vec{N}$ is the unit vector surface normal.
Sources	Shreiner and Angel (2012)
Ref. By	IM1, IM2, IM3, IM4
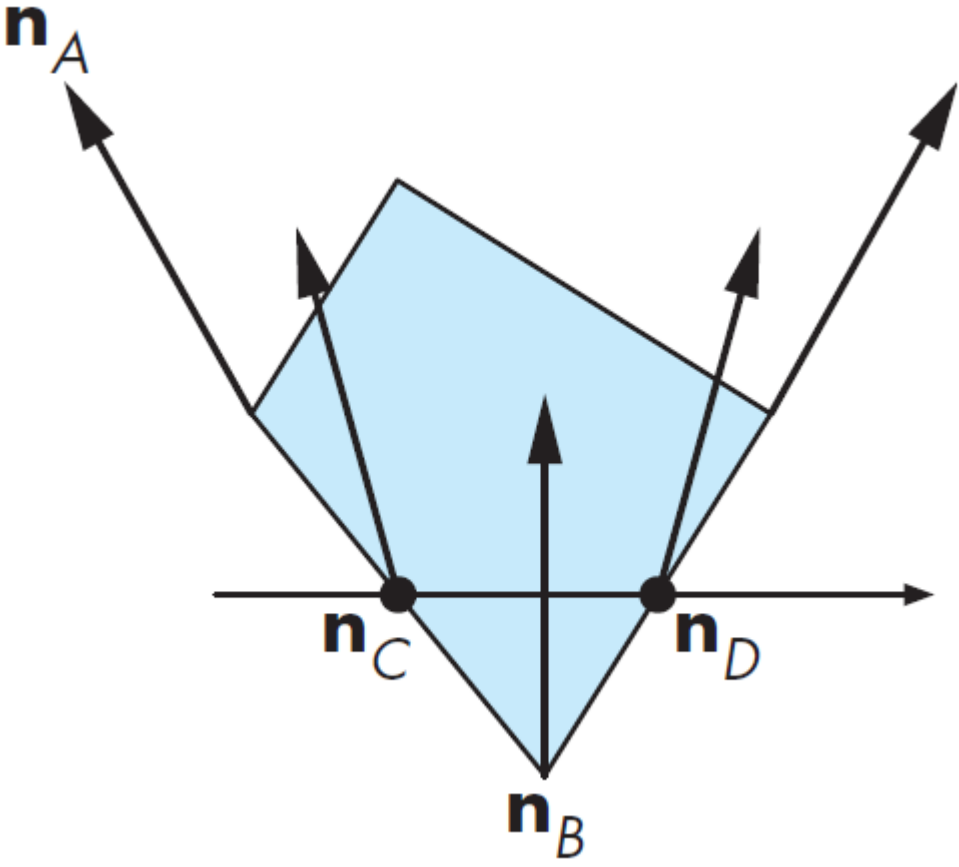
Number	DD5
Label	**Luminance for a Light Source**
Symbol	I_s
SI Units	–
Equation	$I_s = \left\{ \begin{array}{c} I_{sr} \\ I_{sg} \\ I_{sb} \end{array} \right\} = k_s L_s \max((\vec{U_r} \bullet \vec{v})^\alpha, 0)$
Description	p_0 is the position of the light source in Cartesian Coordinates.
	$\vec{U_r}$ is the unit vector of the reflected ray.
	$\vec{v}$ is the vector to the viewer.
	k_s is the specular coefficient of the material.
	α is the shininess coefficient to determine the size of the specular highlight.
Sources	Shreiner and Angel (2012)
Ref. By	IM3, IM4

Number	DD6
Label	**Luminance for a Light Source**
Symbol	I_T
SI Units	–
Equation	$I_T = I_a + I_d + I_s$
Description	p_0 is the position of the light source in Cartesian Coordinates.
Sources	Shreiner and Angel (2012)
Ref. By	IM3

Number	DD7
Label	**Intensity of illumination at point p from a point source, p_0**
Symbol	$i(p, p_0)$
SI Units	–
Equation	$i(p, p_0) = \frac{1}{|p - p_0|^2} I(p_0)$
Description	P and Q are two 3-dimensional vectors. The cross product $(P \times Q)$ is the unit vector normal to both P and Q. The calculation of the cross product's direction follows the *right hand rule* paradigm. This is used in T1 and GD1.
Sources	Lengyel (2003)
Ref. By	AB1, AB2

Number	DD8
Label	**Calculation of Normals for Flat Shading**
Symbol	$\vec{n}$
SI Units	–
Equation	$\vec{n} = \frac{\vec{P} \times \vec{Q}}{|\vec{P} \times \vec{Q}|}$
Description	Flat shading calculates the surface normal of a polygon in a mesh. Given two non-parallel vectors tangent to the surface of the polygon, $\vec{P}$ and $\vec{Q}$, the normal ($\vec{n}$) is the normalized cross-product of $\vec{P}$ and $\vec{Q}$
Sources	Shreiner and Angel (2012)
Ref. By	AB1, AB2

Number	DD9
Label	**Calculation of Normals for Gouraud Shading**
Symbol	$\vec{n_v}$
SI Units	–
Equation	$\vec{n_v} = \frac{\sum_{i=1}^{t_v} n_i}{|\sum_{i=1}^{t_v} n_i|}$
Description	Gouraud shading calculates the normal at the vertices of a polygon in a mesh. Given a vertex, v, the normal $(\vec{n_v})$ is the normalized average of the surface normals $(\vec{n_i})$ for all polygons that share that vertex. t_v is the number of polygons that share the vertex v.
Sources	Shreiner and Angel (2012)
Ref. By	AB1, AB2

Number	DD10
Label	**Calculation of Normals for Phong Shading**
Symbol	$\vec{n_v}$
SI Units	–
Equation	
Description	Phong shading linearly interpolates normals across each the surface of the polygon from the vertex normals. Surface normals are then normalized per pixel.
Sources	Shreiner and Angel (2012)
Ref. By	AB1, AB2
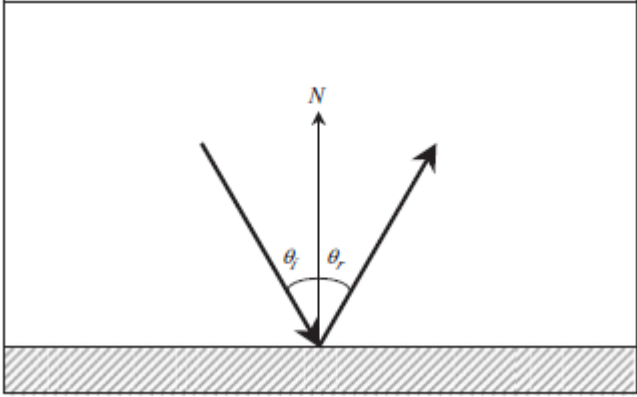
5.4 Goal Statements

Given some light source(s), some object(s) and their respective material properties, and an observer the goal statements are:

GS1: Render a fully lit and shaded scene of the objects based on the observer location.
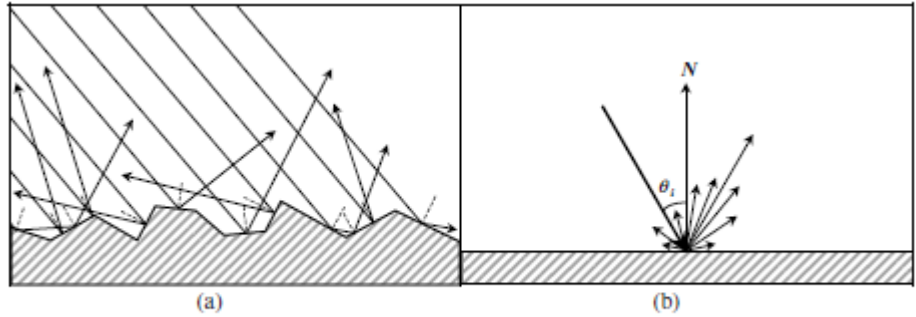
5.5 Theoretical Models

This section focuses on the general equations and laws that Family of Lighting Models is based on.

Number	T1
Label	**Law of Reflection**
Equation	$\angle\theta_i = \angle\theta_r$ or $\vec{L_i} \bullet \vec{N} = \vec{L_r} \bullet \vec{N}$
Description	When a ray of light reflects off a surface, the angle of incident is equal to the angle of reflection. This can also be written as the dot product of the incident ray and the normal is equal to the dot product of the reflected ray and the normal.
Source	Comninos (2005)
Ref. By	GD1

5.5.1 General Definitions

This section collects the laws and equations that will be used in building the instance models.

Number	GD1
Label	**Diffuse Reflection**
SI Units	-
Equation	$\angle\theta_i = \angle\theta_r$ or $\vec{L_i} \bullet \vec{N} = \vec{L_r} \bullet \vec{N}$
Description	Law of Reflection applied to rough surfaces. (a) The surface micro-facets. (b) Diffuse reflection.
Source	Comninos (2005)
Ref. By	DD1

6 Variabilities

The follow section outlines variabilities between family members. Section 6.1 covers the assumptions made to simplify/realise the problem. Section 6.2 captures variabilities in the implementation.

Before tackling those sections, we summarize here variabilities in the problem.

6.1 Assumptions

This section outlines the various assumptions made in defining this problem. These are divided based on their binding time. All assumptions are used to support the 1

- **Scope Time Bindings**:

  AS1: Light will be defined by Geometrical Optics principles (ray-based).

  AS2: Lighting will be handled at the local illumination level.

  AS3: Objects will not cast shadows on each other.

  AS4: Objects will not reflect light at each other.

  AS5: Objects will not emit light.

- **Build Time Bindings**:

  AB1: The virtual environment will be described by a 3D Caretsian Coordinate System using right-hand rules.

  AB2: Positions of light source(s), object(s), and the observer will be defined on a 3D Cartesian Coordinate System.

  AB3: There will only be a single observer in a scene.

  AB4: Objects will be represented by a geometric mesh of triangles.

  AB5: Object meshes will be closed and convex (there is a well defined interior and exterior).

  AB6: All objects are opaque; light will not experience refraction with any object.

- **Run-Time Bindings**: These were captured as inputs to the system.

6.2   Calculation

This section outlines variabilities in design details, capturing variabilities in: input, calculations, algorithms, and data structures.

Input variabilities:

Variability	Parameter of Variation	Constraints	Related
Model room size $\langle h, w, d \rangle$	Set of $\mathbb{R}$	–	AB2
Allowed values for position of light source $\langle x, y, z \rangle$	Set of $\mathbb{R}$	Must be in room boundaries. Light source cannot be inside of an object. Light source cannot be on top of another light source or on top of the observer.	AB2
Allowed values for position of object(s) $\langle x, y, z \rangle$	Set of $\mathbb{R}$	Object(s) must be in room boundaries. Objects must not overlap with each other or a light source	AB2
Allowed values for position of observer $[x, y, z]$	Set of $\mathbb{R}$	Observer must be in room boundaries.	AB2
Allowed values for colour of light $\langle r, g, b \rangle$	Set of $\mathbb{Z}$	0 - 256	–
Allowed values for colour of object $\langle r, g, b \rangle$	Set of $\mathbb{Z}$	0 - 256	–
Reflection coefficient of material	$k_a \in \mathbb{R}$	$0 \leq k_a \leq 1$	–
Specular coefficient of material	$k_s \in \mathbb{R}$	$0 \leq k_s \leq 1$	–
Diffuse coefficient of material	$k_d \in \mathbb{R}$		–
Shininess coefficient of material	$\alpha \in \mathbb{R}$	–	–
Type of light source	Set of $\{Point, Spot, Directional, Ambient\}$	–	–
Type of object(s)	Set of $\{Sphere, Cube, Torus, Teapot\}$	–	–
Type of shading	Set of $\{Flat, Gouraud, Phong\}$	–	–
Number of object(s)	Set of $\mathbb{R}$	–	–
Number of light(s)	Set of $\mathbb{R}$	–	–
Number of observer(s)	Set of $\mathbb{R}$	No. observers = 1	–
Input method	Single file with light, object and other data; multiple files (one for lights and one for objects)	–	–

Table 1: Parameters of Variation for Input.

Calculation variabilities:

Variability	Parameters of Variation	Constraints	Ref.
Interpolation of normals for shading	Set of {DD8, DD9, DD10}	–	AS1
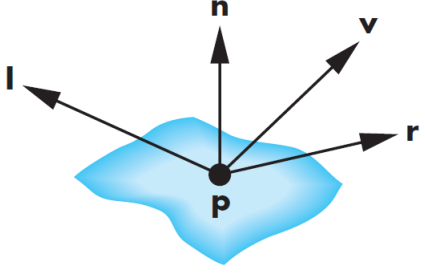
Table 2: Parameters of Variation in Calculations.

6.2.1 Instance Models

This section transforms the problem defined in Section 3.2 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 5.3 to replace the abstract symbols in the models identified in Sections 5.5 and 5.5.1.

Number	IM1
Label	**Lambertian (Diffuse) Reflection Model**
Input	$\vec{L_i}$, p_0, object
Output	$\vec{L_r}$ I_d
Description	L_i is the vector representation of the incident ray. U_i is the unit vector of L_i. $\vec{N}$ is the surface normal unit vector. p_0 is the point the incident ray intercepts the surface of an object. k_d is the coefficient of diffuse reflection. $I_{L_i p_0}$ is the intensity of the light at point P This IM is a concrete refinement on GD1 by modeling the interactions of a single light ray at a single point.
Sources	Comninos (2005); Lengyel (2003); Shreiner and Angel (2012)
Ref. By	IM2

17

Number	IM2
Label	**Half Lambert (Diffuse) Reflection Model**
Input	$\vec{L_i}$, p_0, object
Output	$\vec{L_r}$
	I_d
Description	L_i is the vector representation of the incident ray.
	U_i is the unit vector of L_i.
	$\vec{N}$ is the surface normal unit vector.
	p_0 is the point the incident ray intercepts the surface of an object.
	k_d is the coefficient of diffuse reflection.
	$I_{L_i p_0}$ is the intensity of the light at point P
	This IM is a concrete refinement on IM1 by halving the dot product of the $\vec{U_i}$ and $\vec{N}$ to create a softer shading.
Sources	Comninos (2005); Lengyel (2003); Shreiner and Angel (2012)
Ref. By	

Number	IM3
Label	**Phong Reflection Model**
Input	$\vec{n}$, p, v, $\vec{L_i}$, $I_{a_{L_i}}$, $I_{d_{L_i}}$, $I_{s_{L_i}}$
Output	$\vec{v}$, $\vec{L_r}$, $I_{a_{L_r}}$, $I_{d_{L_r}}$, $I_{s_{L_r}}$
Description	p is a point on the polygon surface.

v is the position of the viewer.

$\vec{L_i}$ is the incident ray hitting point p.

$\vec{n}$ is the surface normal at point p.

$I_{a_{L_i}}$ is the ambient intensity of the light described by L_i.

$I_{d_{L_i}}$ is the diffuse intensity of the light described by L_i.

$I_{s_{L_i}}$ is the specular intensity of the light described by L_i.

$\vec{v}$ is the vector in the direction of the viewer from p.

$\vec{L_r}$ is the perfectly reflected ray.

$I_{a_{L_r}}$ is the ambient intensity of the light described by L_r.

$I_{d_{L_r}}$ is the diffuse intensity of the light described by L_r.

$I_{s_{L_r}}$ is the specular intensity of the light described by L_r.

Sources	Comninos (2005); Lengyel (2003); Shreiner and Angel (2012)
Ref. By	IM4

Number	IM4
Label	**Blinn-Phong Reflection Model**
Input	$\vec{n}$, p, v, $\vec{L_i}$, $I_{a_{L_i}}$, $I_{d_{L_i}}$, $I_{s_{L_i}}$
Output	$\vec{v}$, $\vec{h}$, I_{a_h}, I_{d_h}, $I_{s_{L_r}}$
Description	p is a point on the polygon surface.
	v is the position of the viewer.
	$\vec{L_i}$ is the incident ray hitting point p.
	$\vec{n}$ is the surface normal at point p.
	$I_{a_{L_i}}$ is the ambient intensity of the light described by L_i.
	$I_{d_{L_i}}$ is the diffuse intensity of the light described by L_i.
	$I_{s_{L_i}}$ is the specular intensity of the light described by L_i.
	$\vec{v}$ is the vector in the direction of the viewer from p.
	$\vec{L_r}$ is the perfectly reflected ray.
	$I_{a_{L_r}}$ is the ambient intensity of the light described by L_r.
	$I_{d_{L_r}}$ is the diffuse intensity of the light described by L_r.
	$I_{s_{L_r}}$ is the specular intensity of the light described by L_r.
	$\vec{h}$ is the unit vector halfway between $\vec{L_i}$ and $\vec{v}$
	ψ is the halfway angle, the angle between the normal and halfway vectors.
	ϕ is the angle between the viewer and the reflected ray. It is exactly 2ψ.
Sources	Comninos (2005); Lengyel (2003); Shreiner and Angel (2012)
Ref. By	

6.3 Output

Variability	Parameter of Variation
Output file	Generated code file that when compiled renders a scene, executable file that shows the rendered scene

Table 3: Parameters of Variation for Output.

7 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

7.1 Family of Functional Requirements

R1: Input data for light source(s) and object(s) are correctly read from file.

R2: Input data meets constraints laid out in Table 1.

R3: Calculate the surface normals for object(s) based on shading model selected.

R4: Calculate the reflection vectors off of object(s) based on light position(s) and observer.

R5: Calculate the light intensity based on light position(s) and object material properties.

R6: Calculate the shaded colours for the faces of the object(s).

R7: Draw and render object(s) in scene.

R8: Generate a program that renders the specified scene.

R9: Verify generated code renders the scene as specified through the inputs.

7.2 Nonfunctional Requirements

R10: System responds with specific error message when user inputs contain errors (type mismatch, data outside of constraints).

R11: System responds with specific error message when system cannot read input files.

R12: System provides a default setup for shading and reflection models.

R13: System asks user if they would like to use the default setup when shading and reflection model information is missing from input.

R14: Users can render a default scene (one cube, one point light in the centre of the room, phong shading, and phong reflection model) faster than in OpenGL.

8 Likely Changes

LC1: Refractive materials incorporated into family.

LC2: More reflection models incorporated into family.

LC3: Variations on how normals are computed (e.g. partial derivatives) may be added to the family.

9 Traceability Matrices and Graphs

	AS1	AS2	AS3	AS4	AS5	AB1	AB2	AB3	AB4	AB5	AB6
T1	X				X	X					X
GD1	X		X		X	X					
IM1	X	X	X	X	X						
IM2	X	X	X	X	X						
IM3	X	X	X	X	X		X	X			
IM4	X	X	X	X	X		X	X			
DD1						X	X				
DD2						X	X				
DD3	X	X	X	X							
DD4		X			X						X
DD5		X			X						X
DD6		X	X	X	X						X
DD7	X	X				X	X				
DD8	X				X	X	X		X		
DD9	X	X	X		X	X	X		X	X	X
DD10	X	X	X		X	X	X		X	X	X
LC3	X					X	X		X	X	
LC2		X	X	X	X		X				X
LC1		X	X	X	X						X

Table 4: Traceability Matrix matching Scope and Build Time Assumptions to Theoretical Models, General Definitions, Instance Models, Data Definitions and Likely Changes.

References

Peter Comninos. *Mathematical and Computer Programming Techniques for Computer Graphics*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 1852339020.

Eric Lengyel. *Mathematics for 3D Game Programming and Computer Graphics, Second Edition*. Charles River Media, Inc., Rockland, MA, USA, 2003. ISBN 1584502770.

Dave Shreiner and Edward Angel. Interactive computer graphics: A top-down approach with shader-based opengl, 2012.

W. Spencer Smith. Systematic development of requirements documentation for general purpose scientific computing software. In *Proceedings of the 14th IEEE International Requirements Engineering Conference, RE 2006*, pages 209–218, Minneapolis / St. Paul, Minnesota, 2006. URL http://www.ifi.unizh.ch/req/events/RE06/.