Lightning and Shading

- **Lighting**
  - the process of computing the luminous intensity (i.e., outgoing light) at a particular 3-D point, usually on a surface.
  - The emission color is a color that does not come from any external source and therefore seems to be emitted by the material itself.
  - In the presence of light, the object will be brighter than can be accounted for by the light that illuminates it, and in that sense, it appears to glow. The emission color is usually black; that is, the object has no emission.
- The properties of a surface that determine how it interacts with light are referred to as the surface **material**.
- The material property that determines the size and sharpness of specular highlights is called **shininess**.
- **Specular Reflection**
  - Direct reflections of the light source off of a shiny surface Smooth surfaces
- **diffuse color**
  - tells how the material reflects light diffusely, and a specular color that tells how it reflects light specularly.
  - the basic color of the object.
- **specular color**
  - determines the color of specular highlights. The diffuse and specular colors can be the same.
- **Diffuse Reflection**
  - When light hits an object with a rough surface, it is reflected in all directions.
  - The larger the angle (up to 90 degrees), the larger the area the incident light is spread over
- **Ambient Color.**
  - The third color is the material's ambient color, which tells how the surface reflects ambient light.
- **Ambient light**
  - refers to a general level of illumination that does not come directly from a light source. It consists of light reflected and re-reflected so many times that it is no longer coming from any particular direction.
- **Ambient lighting**
  - Light reflected or scattered from other objects in the scene.
  - Environmental light
- **Point Light**
  - Emits light in all directions.
- **Directional Light**
  - Has parallel light rays, all from the same direction.
- **Shading**
  - the process of assigning colors to pixels.
  - referred to as the implementation of the illumination model at the pixel points or polygon surfaces of the graphics objects.
- **Why do we need Lightning and Shading?**
  - Sphere with shading. Shading has visual cues for humans (shape, light position, a viewer position, surface orientation, material properties, etc.)
- **What Causes Shading?**
  - Shading caused by different angles with light, camera at different points.
  - **Light attributes**: intensity, color, position, direction, shape
  - **Surface attributes**: color, reflectivity, transparency, etc.
  - **Interaction**: between lights and objects
- **Flat Shading**
  - is a single intensity is calculated for each polygon.
  - All points over the polygon's surface are then displayed with the same intensity value. Constant Shading can be useful for quickly displaying the general appearances of the curved surface.
- **Gouraud Shading**
  - renders a polygon surface by linear interpolating intensity value across the surface.
  - Intensity values for each polygon are coordinated with the value of adjacent polygons along the common edges, thus eliminating the intensity discontinuities that can occur in flat shading.
- **Phong Shading**
  - A more accurate method for rendering a polygon surface is to interpolate the normal vector and apply the illumination model to each surface point.
  - This method developed by Phong Bui Tuong is called Phong Shading or normal vector Interpolation Shading. It displays more realistic highlights on a surface and greatly reduces the Match-band effect.
  - **Three parameter model comprised of**
    - **Diffuse**: non-shiny illumination and shadows
    - **Specular**: shiny reflections
    - **Ambient**: background illumination

- **Illumination**
  - The eye works like a camera.
  - Sense the amount of light coming from different directions.

Game Development

- **Game**
  - is an activity or sport usually involving skill, knowledge, or chance, in which you follow fixed rules and try to win against an opponent or solve a puzzle.
  - is a structured form of play, usually undertaken for entertainment or fun and sometimes used as an educational tool.
- **Computer Games**
  - enable players to interact with a virtual environment for entertainment and fun. Many computer games are available, ranging from traditional card games to more advanced video games such as role-playing games and adventure games.
  - es, such as consoles, smartphones, tablets, virtual reality headsets, or personal computers. They can be played on the internet, local area networks, or offline.
  - is a workout for your mind disguised as fun. Studies have shown that playing video games may increase gray matter in the brain and boost brain connectivity. (Gray matter is associated with muscle control, memories, perception, and spatial navigation.)
- **Game Development**
  - is the complex process of crafting video games, from ideation to final product.
  - is an iterative process that involves designing, programming, and interactive video games.
  - is a collaborative effort that brings together experts with diverse skills and knowledge, such as game designers, programmers, artists, musicians, writers, and testers.
- **Fundamentals of Game Development**
  - **Conceptualization**: This is where game ideas are born. They could be a unique gameplay mechanic, an interesting story, or a visual style.
  - **Art and Graphics**: Visuals are crucial in creating immersive gaming experiences. Artists and graphic designers create characters, environments, animations, and other visual elements that make up the game world.
  - **Publishing and Distribution**: After development, the game needs to be distributed. releasing published This it on and involves digital storefronts such as Steam, console marketplaces, or mobile app stores.
  - **Design**: Game design involves defining the game's rules, mechanics, and overall structure.
  - **Sound Design**: Sound effects, music, and voice acting enhance a game's atmosphere and immersion.
  - **Programming**: Programming is the technical implementation of the game design.
  - **Testing and Quality Assurance**: Testing and Quality Assurance are important parts of game development. They help identify and fix bugs, balance gameplay, and ensure a good player experience.
- **Platforms**
  - **Personal Computers**: Games can be accessed through desktop and laptop computers. Games are often released on digital distribution platforms such as Steam, Epic Games Store, GOG (Good Old Games), and Itch.io.
  - **Console**: Console games are specifically designed to be played on gaming consoles like PlayStation, Xbox, and Nintendo Switch and are typically distributed to users through the respective console's online stores.
  - **Mobile**: Download games for smartphones and tablets directly from app stores.
  - **Web**: Publishers offer browser-based games that can be played on gaming portals or the publisher's website. ▪
  - **VR/AR**: With the rise of virtual reality (VR) and augmented reality (AR) technology, some publishers release games for VR headsets, such as the Oculus Rift, HTC Vive, and PlayStation VR.
  - **Streaming Services**: Streaming services such as Google Stadia, Xbox Cloud Gaming, and NVIDIA GeForce Now allow users to play games online without downloading.
- **Game Development Process**
  - **Ideation and Conceptualization**: The game development process begins with generating ideas and creating a compelling game concept. This phase involves brainstorming, market research, and the establishment of the game's core pillars, such as its genre, theme, and target audience.
  - **Game Design**: Once the initial concept of a game is established, the game

design phase begins. This phase is a crucial step in which the game's mechanics, systems, and features are planned in detail.
    - **Asset Creation**: The game's visual and audio assets are created concurrently with the design phase. Artists, 3D modelers, animators, and sound designers work together to bring the game world to life, crafting characters, environments, special effects, and a captivating soundtrack that enhances player immersion.
    - **Prototyping**: After the design is complete, the next step is to create prototypes for testing and iterating on the game's mechanics and systems.
    - **Programming and Implementation**: The game's core functionality is built during this phase, where programmers translate the design vision into a fully functional interactive experience.
    - **Testing and Optimization**: Rigorous testing is conducted throughout development to ensure the game's stability, performance, and overall quality.
    - **Release and Post-Launch Support**: After thorough testing and polishing, the game is ready for release and post-launch support. This phase involves distribution, marketing, and continuous support, such as bug fixes, updates, and additional content, to ensure the game's longevity and keep players engaged.
- **2D & 3D Design**
    - Graphics
    - Gameplay Mechanics
    - Level Design
    - User Interface
    - Animation
    - Sound and Music
    - Platform
- **Game Engine and Tools**
    - A game engine is a software framework designed to facilitate the creation and development of video games.
    - Game engines typically include features such as rendering engines for graphics, physics engines for realistic interactions, scripting or programming interfaces for game logic, audio systems, animation tools, and more.
    - Game engines abstract many low-level programming and technical complexities, allowing developers to focus on game design and gameplay mechanics.
    - Engines are often tailored for specific games or platforms, yet many boast a versatility that assures support for various genres and target platforms.
- **Game Development Tools**
    - 3D Modeling and Animation Software
    - Texture Editors
    - Sound and Music Editors
    - Integrated Development Environments
    - Version Control Systems
- **Game Design Principles**
    - **User-Centric Design**:
        - Player-centered design, also known as user-centric design, is an approach to game development that prioritizes the player's needs, preferences, and experiences.
        - This design philosophy centers around understanding the target audience and crafting a game experience that caters to their expectations, desires, and abilities.
    - **Balanced Game Mechanics**:
        - Balanced game mechanics are essential for game development, ensuring fair, challenging, and enjoyable gameplay. ▪
        - These mechanics balance different aspects of the game, such as player abilities, resources, challenges, and objectives.
    - **Iterative Development**:
        - Iterative development is an approach where a game is built incrementally through cycles or iterations. ▪
        - Instead of attempting to create the entire game in a single, linear process, developers work on small, manageable portions of the game at a time, continually refining and improving it based on feedback and testing.
    - **Emergent Gameplay**:
        - Emergent gameplay is a fundamental concept in game design that refers to the emergence of unscripted, player driven experiences within a game world.
        - It occurs when complex interactions between game systems, mechanics, and player actions give rise to unexpected

outcomes and behaviors that the developers did not explicitly design.

- **Types of Computer Games**
  - **Card Games.**
    - A card game is played for pleasure or gambling (or both) with one or more decks of playing cards. Games using playing cards exploit the fact that cards are individually identifiable from one side only so that each player knows only the cards he holds, not those held by anyone else
  - **Board Games**
    - A board game is played by moving pieces on a special board.
    - a strategy game (such as chess, checkers, or backgammon) played by moving pieces on a board.
  - **Puzzles**
    - Puzzle games concentrate on completion, which requires players to solve a logic puzzle or navigate a challenge to progress to the next, more difficult challenge.
  - **Maze Games**
    - A maze is a puzzle game where a player moves in complex and branched passages to find a particular target or location.
  - **Fighting Games**
    - Fighting games are a type of action game where two (in one-
    - on-one fighting games) or more (in platform fighters) on-screen characters fight each other.
  - **Action Games**
    - The action game is a video game genre that emphasizes physical challenges, including hand–eye coordination and reaction time. The genre includes various sub-genres, such as fighting games, beat them ups, shooter games, and platform games, widely considered the most important action games.
  - **Adventure Games**
    - Adventure games focus on puzzle solving within a narrative framework, generally with few or no action elements.
  - **Role Playing Games.**
    - The role-playing game, a game in which players assume the roles of fantasy characters.
  - **Sports Games**
    - A sports game is a video game genre that simulates sports practice. Most sports have been recreated with a game, including team sports, track and field, extreme sports, and combat sports.
  - **Simulation Games**
    - Simulation games are a genre of games that are designed to mimic activities you'd see in the real world. The purpose of the game may be to teach you something.

Introduction to OpenGL

- **OpenGL**
  - (Open Graphics Library) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics.
  - It provides a set of functions to interact with the graphics hardware to render high-quality images.
  - OpenGL operates as **a state machine**: you set various parameters (such as color, texture, etc.), and then issue **rendering commands** to draw geometric primitives (points, lines, triangles) or render complex objects by assembling these primitives.
  - It **seamlessly** operates on a **multitude of operating** systems, including Windows, macOS, Linux, and mobile platforms like Android and iOS.
  - OpenGL has faced **competition** from other graphics APIs like DirectX and Vulkan, each with its strengths and target platforms.
- **What is the role of OpenGL in graphics programming?**
  - OpenGL is a crucial tool for **rendering** 2D and 3D graphics on a computer screen.
  - OpenGL acts as a **rendering engine**, providing functions and operations for creating images from geometric shapes and data.
  - One of the primary roles of OpenGL is to abstract the complexities of various graphics **hardware architectures**. It does so with utmost efficiency,

providing a unified interface for developers to interact with the GPU.
    - The **rendering pipeline**, as defined by OpenGL, comprises several sequential stages that graphics data undergoes to culminate in the final image. These stages include vertex processing, primitive assembly, rasterization, fragment processing, and framebuffer operations.
    - OpenGL supports rendering basic **geometric primitives** such as points, lines, and triangles. These primitives serve as building blocks for creating complex 3D scenes.
    - OpenGL supports a myriad of **rendering techniques**, each capable of creating visually stunning and realistic graphics. From lighting and shading to texturing, blending, and shadow mapping.
- **OpenGL Architecture**
    - OpenGL is a **cross-platform** API used to render 2D and 3D graphics in computer programs.
    - **Components**: API, Shaders, Rendering Pipeline
- **OpenGL Components**
    - The **OpenGL API (Application Programming Interface)**
        - is a collection of functions and commands developers can use to interact with the graphics hardware.
    - **Shaders**
        - **Shaders**, these small but powerful programs, are written in specialized languages like GLSL (OpenGL Shading Language) and run directly on the GPU (Graphics Processing Unit), ensuring efficient and direct execution.
        - **Vertex shaders** manipulate vertex data (e.g., position, color) before it is rasterized.
        - **Fragment shaders** determine the final color of pixels on the screen.
    - **Rendering Pipeline**
        - OpenGL's core is the rendering pipeline, which **transforms** geometric data into rendered images on the screen.