Shape, square

Description automatically generated **ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

-----🙞🙜🕮🙞🙜-----

A picture containing text, vector graphics, clipart

Description automatically generated

**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**MÔN: ĐỒ HỌA MÁY HỌC**

**LỚP: CS105.N21**

**Đề Tài: 3D CITY**

**Sinh viên thực hiện:**

Phạm Đức Anh 20521076

**Giảng viên hướng dẫn:** Cáp Phạm Đình Thăng

**Ho Chi Minh City, 2023**

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 4](#_Toc138601983)

[**CHƯƠNG 1 : GIỐI THIỆU** 4](#_Toc138601984)

[**1.** **Tổng quan** 4](#_Toc138601985)

[**2.** **Lý do chọn đề tài** 5](#_Toc138601986)

[**CHƯƠNG 2: CÔNG CỤ HỔ TRỢ** 6](#_Toc138601987)

[**1.** **Blender** 6](#_Toc138601988)

[**2.** **Three js** 6](#_Toc138601989)

[**CHƯƠNG 3: CÁC BƯỚC TIẾN HÀNH** 7](#_Toc138601990)

[**3.1** **Xây dựng mô hình** 7](#_Toc138601991)

[**3.1.1** **Quy trình lập mô hình: Building** 7](#_Toc138601992)

[**3.1.2** **Quy trình lập mô hình: Road** 9](#_Toc138601993)

[**3.1.3** **Quy trình lập mô hình: Park** 10](#_Toc138601994)

[**3.1.4** **Hoàn thành mô hình City** 12](#_Toc138601995)

[**3.1.5** **Sử dụng mô hình Car low poly có sẵn** 13](#_Toc138601996)

[**3.2** **Xây dựng các chức năng** 14](#_Toc138601997)

[**3.2.1** **Đặt vị trí của xe** 14](#_Toc138601998)

[**3.2.2** **Xây dựng góc nhìn** 15](#_Toc138601999)

[**3.2.3** **Xây dựng chức năng điều khiển xe** 18](#_Toc138602000)

[**3.2.4** **Tạo nguồn sáng và điều chỉnh góc độ sáng** 20](#_Toc138602001)

[**CHƯƠNG 4: KẾT LUẬN** 23](#_Toc138602002)

[**1.Ưu điểm** 23](#_Toc138602003)

[**2. Nhược điểm** 24](#_Toc138602004)

[**3. Hướng phát triển** 25](#_Toc138602005)

[**TÀI LIỆU THAM KHẢO** 25](#_Toc138602006)

**MỤC LỤC HÌNH ẢNH**

[Hình 1: Blender 7](#_Toc138603419)

[Hình 2 : Three js 8](#_Toc138603420)

[Hình 3 : Modeling Process Building 9](#_Toc138603421)

[Hình 4 : Building after decoration 10](#_Toc138603422)

[Hình 5 : Modeling Process: Road 11](#_Toc138603423)

[Hình 6 : The road after decoration 11](#_Toc138603424)

[Hình 7 : Modeling Process Park 12](#_Toc138603425)

[Hình 8 : The park after decoration 13](#_Toc138603426)

[Hình 9 : The City 14](#_Toc138603427)

[Hình 10 : The Car low poly 14](#_Toc138603428)

[Hình 11 : The 3D city 16](#_Toc138603429)

[Hình 12 : The 3D city with camera.position.set(-8, 30, 24) 17](#_Toc138603430)

[Hình 13 : The 3D city with camera.position.set(-10, 20, 30) 18](#_Toc138603431)

[Hình 14 : The 3D city with camera.position.set(-Math.PI/6, 0, 0) 19](#_Toc138603432)

[Hình 15 : The Car is moving 20](#_Toc138603433)

[Hình 16 : The car is turning to the right 21](#_Toc138603434)

[Hình 17 : The car returns to its original position 21](#_Toc138603435)

[Hình 18 : The Brightness slider 22](#_Toc138603436)

[Hình 19 : The light source from left to right 23](#_Toc138603437)

[Hình 20 : The light source from right to left 24](#_Toc138603438)

# **LỜI MỞ ĐẦU**

Đồ họa máy tính là lĩnh vực trong công nghệ thông tin liên quan đến việc tạo ra, xử lý và hiển thị hình ảnh và đồ hoạ trên máy tính. Nó có nhiều ứng dụng trong thiết kế đồ họa, trò chơi điện tử, đồ họa khoa học và thiết kế kiến trúc. Công nghệ đồ họa máy tính phát triển nhanh chóng và bao gồm đồ họa 2D và 3D, thiết kế giao diện người dùng và các công nghệ hiển thị khác.

Trong đề tài lần này , em đã xây dựng một mô hình 3D về city bằng cách áp dụng những kiến thức đã được dạy trên lớp và tìm hiểu qua các nguồn khác nhau.Chúng em đã dành hết những khả năng, kiến thưucs và tâm huyết của mình để cùng làm nên đề tài này để đạt được kết quả tốt nhất.

Em cũng xin được gửi lời cảm ơn tới thầy Mai tiến dũng, một người thầy đã đặt nền móng, trao dồi những kiến thức đầu tiên về đồ họa máy tính . Thầy đã nhiệt tình giảng dạy , hỗ trợ chúng em trong thời gian qua để đạt được những điều tốt nhất. Em mong thầy sẽ xem xét đồ án và chỉ ra những sai sót, góp ý chỉnh sửa dể chúng em được tốt hơn. Em xin chân thành cảm ơn.

# **CHƯƠNG 1 : GIỚI THIỆU**

1. **Tổng quan**

3D City là một đề tài liên quan đến xây dựng mô hình thành phố trong không gian 3 chiều. Nó nhằm tạo ra một phiên bản số hóa chính xác của một thành phố, cho phép mô phỏng và hiển thị các yếu tố của môi trường đô thị. Để thực hiện đề tài này, cần thu thập dữ liệu từ nhiều nguồn, xử lý dữ liệu để tạo ra mô hình 3D, và tích hợp thông tin từ các hệ thống khác nhau. Mục tiêu của đề tài là cung cấp một hệ thống đa chiều cho nghiên cứu và quản lý thông tin về thành phố, giúp các nhà quản lý và nhà phát triển đô thị có cái nhìn trực quan và chân thực về thành phố.

1. **Lý do chọn đề tài**

**Mô phỏng thực tế**: Mô hình 3D City cho phép tái tạo một phiên bản số hóa chính xác của thành phố hoặc khu vực cụ thể. Điều này giúp tái tạo một môi trường sống thực tế và chân thực hơn, cho phép các nhà nghiên cứu, quản lý đô thị và nhà phát triển có cái nhìn rõ ràng về thành phố.

**Quản lý đô thị và phát triển**: Mô hình 3D City cung cấp một công cụ mạnh mẽ để quản lý thông tin đô thị và phát triển thành phố. Nó giúp nhà quản lý đô thị hiểu rõ hơn về cấu trúc đô thị, quy hoạch không gian và tình trạng hạ tầng, từ đó đưa ra quyết định thông minh về phát triển và cải thiện đô thị.

**Trực quan hóa dữ liệu**: Mô hình 3D City giúp trực quan hóa dữ liệu đô thị phức tạp. Thay vì phải xem thông tin trên bản đồ hoặc trong bảng số, mô hình 3D cho phép người dùng xem và tương tác với dữ liệu thành phố một cách trực quan và dễ dàng hơn. Điều này giúp người dùng hiểu rõ hơn về môi trường đô thị và các yếu tố liên quan.

**Nghiên cứu và phát triển công nghệ**: Đề tài 3D City cung cấp một cơ sở cho việc nghiên cứu và phát triển công nghệ liên quan đến mô phỏng đô thị. Nó khuyến khích sự phát triển của công nghệ 3D, xử lý hình ảnh, trực quan hóa và tích hợp dữ liệu, góp phần mở ra những cơ hội mới trong lĩnh vực này.

# **CHƯƠNG 2: CÔNG CỤ HỔ TRỢ**

1. **Blender**

Blender là một phần mềm 3D miễn phí và mã nguồn mở được sử dụng để tạo ra đồ họa máy tính, hoạt hình, và hiệu ứng hình ảnh chuyển động. Nó cung cấp một công cụ mạnh mẽ cho việc mô hình hóa, render, và tạo ra các hoạt ảnh 3D, cả trong lĩnh vực thiết kế đồ họa và lĩnh vực phim ảnh.

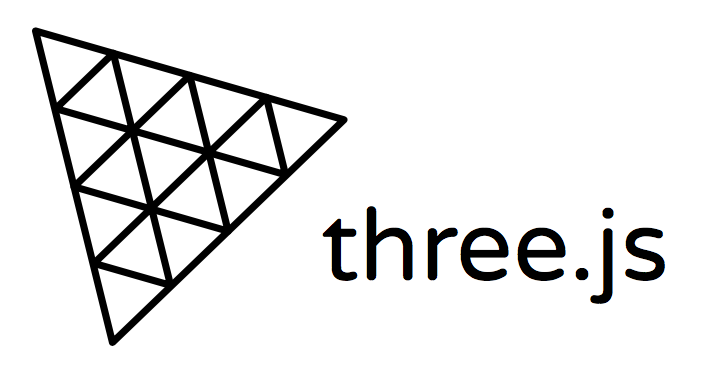


Hình 1: Blender

Blender hỗ trợ nhiều tính năng đa dạng, bao gồm tạo mô hình 3D, tạo và xử lý vật liệu, ánh sáng và cảnh quan, hoạt hình chuyển động, xử lý hình ảnh và video, và phân tích động. Nó cung cấp một giao diện người dùng trực quan và mạnh mẽ, cho phép người dùng tạo ra các hình ảnh và hiệu ứng chất lượng cao.

1. **Three js**

Three.js là một thư viện JavaScript mã nguồn mở và miễn phí được sử dụng để tạo và hiển thị đồ họa 3D trong trình duyệt web. Nó cung cấp một tập hợp các công cụ và tính năng mạnh mẽ để xây dựng và thao tác các đối tượng 3D, áp dụng các hiệu ứng ánh sáng và vật liệu, và tạo ra các hoạt ảnh chuyển động.



Hình 2 : Three js

Three.js sử dụng WebGL, một công nghệ đồ họa phía máy tính để xử lý đồ họa 3D, để tận dụng sức mạnh của GPU (đơn vị xử lý đồ họa) trong máy tính người hem. Nó giúp tạo ra các trải nghiệm đồ họa 3D phong phú và tương tác trên các trình duyệt web mà không cần sử dụng các plugin bên ngoài.

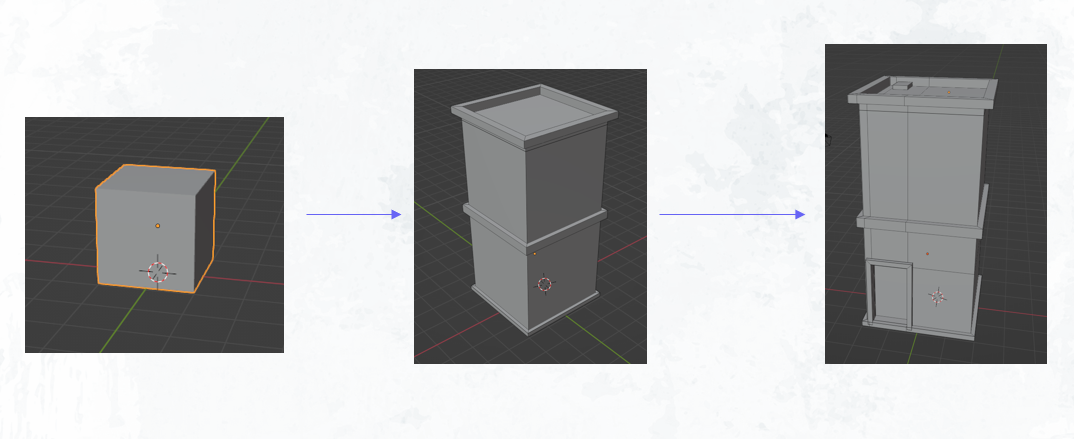
# **CHƯƠNG 3: CÁC BƯỚC TIẾN HÀNH**

## **Xây dựng mô hình**

* + 1. **Quy trình lập mô hình: Building**

Trên giao diện Blender, sẽ thấy một cửa sổ chính và một viewport hiển thị không gian làm việc 3D.

* Để tạo căn nhà, chúng ta sẽ bắt đầu bằng việc hem một hình hộp (Cube). Chọn menu “Add” hoặc sử dụng phím tắt Shift + A để mở danh sách các đối tượng có thể hem vào
* Trong danh sách, chọn “Mesh” và sau đó chọn “Cube”. Một hình hộp sẽ được hem vào không gian làm việc.
* Sử dụng công cụ biến đổi (Transform Tools) như Move Scale và Rotate để điều chỉnh kích thước và vị trí của hình hộp để tạo thành hình dạng cơ bản của ngôi nhà.
* Để tạo các cửa và cửa sổ, ta sử dụng công cụ kết cạnh (Edge Loops) và công cụ trích xuất (Extrude) để tạo các mặt và hình dạng phù hợp.
* Sử dụng công cụ tạo chấn động (Subdivision Surface) để làm mềm các cạnh và góc của ngôi nhà, tạo ra một diện mạo mượt mà hơn.



Hình 3 : Modeling Process Building

* Nếu thêm màu sắc và vật liệu cho ngôi nhà, có thể chọn ngôi nhà và chọn tab "Material trên bên phải của giao diện Blender. Tại đây, có thể tạo và chỉnh sửa các vật liệu để áp dụng cho các phần khác nhau của ngôi nhà.



Hình 4 : Building after decoration

* Cuối cùng, sử dụng công cụ ánh sáng (Lighting Tools) để thêm ánh sáng vào ngôi nhà và công cụ camera để thiết lập góc nhìn để hiển thị ngôi nhà một cách thích hợp.
* Khi đã hoàn thành xây dựng và trang trí ngôi nhà, bạn có thể lưu lại công việc của mình bằng cách chọn "File" và sau đó "Save" hoặc "Save As"
  + 1. **Quy trình lập mô hình: Road**

Mô hình đường được sử dụng ở đây gồm 2 phần là là Intersection (ngã tư) và Straight road (đường thẳng). Chúng ta sẽ xây dựng bằng blender như sau:

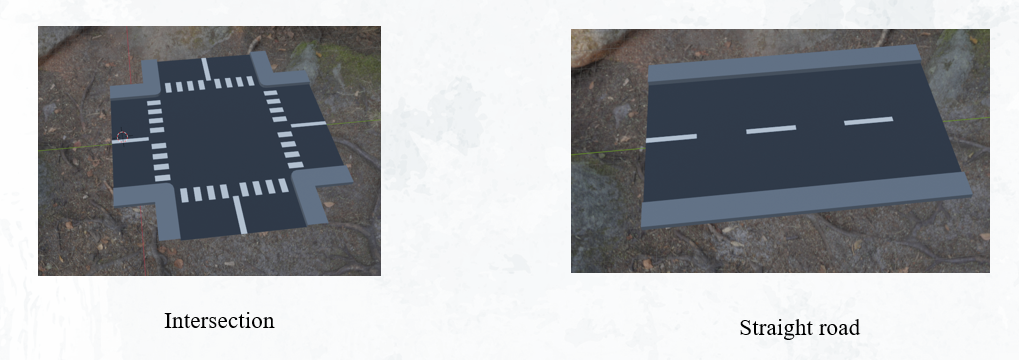
* Xóa đối tượng mặc định trong screen bằng cách chọn nó và nhấn phím "X" hoặc "Delete".
* Tạo một mặt phẳng làm đường bằng cách nhấn Shift + A và chọn "Mesh" sau đó "Plane".
* Thiết lập kích thước và chiều dài của mặt phẳng theo ý muốn bằng cách chọn mặt phẳng và điều chỉnh các giá trị trong bảng thuộc tính (Properties Panel) bên phải.

A white squares on a black background

Description automatically generated with low confidence

Hình 5 : Modeling Process: Road

* Tạo một intersection bằng cách sử dụng công cụ "Loop Cut" để tạo hai đường cắt ngang hoặc dọc trên mặt phẳng.
* Chọn một trong các đường cắt, sau đó nhấn "E" để kéo đường cắt theo hướng muốn tạo ngã tư. Làm tương tự với đường cắt còn lại để hoàn thiện ngã tư.
* Để làm cho mô hình đường thêm chi tiết, dụng các công cụ như "Subdivision Surface" để làm mịn các cạnh hoặc "Bevel" để bo tròn các cạnh.
* Cũng có thể thêm các chi tiết khác như vạch kẻ đường, biển báo hoặc cây cối để tạo ra một mô hình đường thực tế hơn.

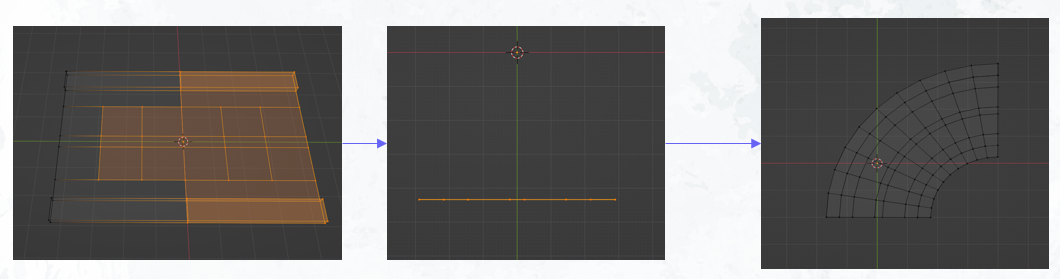


Hình 6 : The road after decoration

* Khi hoàn thành xây dựng, có thể tùy chỉnh vật liệu và ánh sáng cho mô hình đường bằng cách sử dụng bảng thuộc tính (Properties Panel) và bảng Node Editor trong Blender.
  + 1. **Quy trình lập mô hình: Park**

Mô hình park được xây dựng ở đây là một mô hình công viên mini để them vào model city tạo them cảnh quan đẹp. Chúng em sẽ xây dựng bằng blender như sau:

* Bắt đầu xây dựng mô hình bằng việc thêm các yếu tố cơ bản như mặt đất và đường đi. có thể sử dụng công cụ "Plane" để tạo một mặt phẳng làm mặt đất của công viên và sử dụng công cụ "Extrude" để tạo đường đi trong công viên.



Hình 7 : Modeling Process Park

* Sử dụng công cụ "Add" để thêm các yếu tố khác như cây cối, bụi cây, cỏ, và các loại hoa. Blender cung cấp một số đối tượng cơ bản như cube, sphere, cylinder, hay cone mà ta có thể sử dụng để tạo các yếu tố tự nhiên trong công viên.
* Để tạo cây cối, sử dụng một cylinder làm thân cây và một sphere làm chiếc lá. Cũng có thể sử dụng các công cụ như "Scale" và "Rotate" để tùy chỉnh hình dạng và vị trí của các yếu tố trong công viên.
* Để tạo hồ nước, sử dụng công cụ "Add" và chọn "Mesh" sau đó "Circle" để tạo một hình tròn, sau đó dùng công cụ "Extrude" để tạo chiều cao và hình dạng của hồ.
* Ngoài ra, có thể thêm các yếu tố khác như ghế, bàn, đèn chiếu sáng hoặc khu vui chơi trẻ em bằng cách sử dụng các hình dạng cơ bản và tùy chỉnh chúng theo ý muốn.

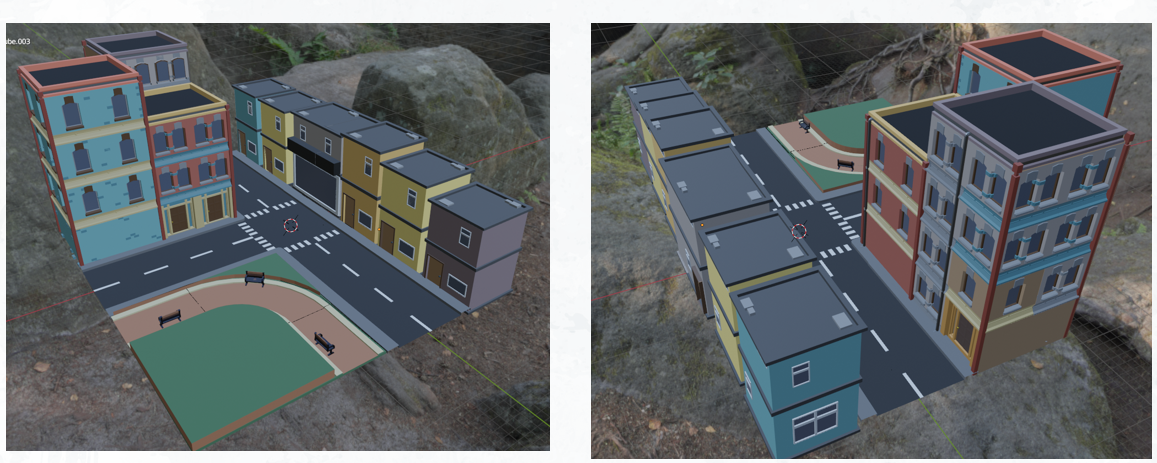
A picture containing screenshot, rectangle, design

Description automatically generated

Hình 8 : The park after decoration

* Khi hoàn thành xây dựng, có thể tùy chỉnh vật liệu và ánh sáng cho mô hình công viên bằng cách sử dụng bảng thuộc tính (Properties Panel) và bảng Node Editor trong Blender.
  + 1. **Hoàn thành mô hình City**

Sau khi xây dựng từng model riêng lẻ như building, road, park xong. Chúng em sẽ kết hợp, gắn các model nhỏ lại với nhau để hoàn thành model City hoàn chỉnh. Tạo nên một background thành phố tiêu chuẩn.



Hình 9 : The City

* + 1. **Sử dụng mô hình Car low poly có sẵn**

Ngoài việc xây dựng mô hình City bằng công cụ blender, thì để tang them tính trực quan cho đề tài, chúng em đã download một mô hình car low poly có sẵn về them vào mô hình thành phố để mang lại những lợi ích.

**A picture containing vehicle, wheel, transport, land vehicle

Description automatically generated**

Hình 10 : The Car low poly

* Tăng tính sinh động: Bằng cách thêm mô hình Car vào mô hình thành phố, tạo ra một cảnh quan sống động hơn. Xe cộ di chuyển trên đường và tạo ra sự cảm giác hoạt động của thành phố.
* Tăng tính thực tế: Mô hình Car giúp tái hiện thế giới thực và tăng tính thực tế của mô hình thành phố. Điều này có thể giúp ta hình dung và hiểu rõ hơn về cách giao thông diễn ra trong một thành phố thực tế.
* Tạo ra sự tương tác: Khi mô hình Car di chuyển trong mô hình thành phố, n có thể tạo ra các tương tác khác nhau, xe đi qua các ngã tư, hoặc đặt các biển báo giao thông. Điều này giúp n tạo ra các kịch bản hoặc trò chơi tương tác trong mô hình thành phố.
  1. **Xây dựng các chức năng** 
     1. **Đặt vị trí của xe**

Chúng em có 2 phiên bản `GLTFLoader` được tạo: `loader` và `loader2 tương ứng với City và Car. Đây là cách mô hình ô tô được đặt trong mô hình thành phố:

* Phiên bản `loader` tải mô hình "City ". Sau khi mô hình được tải thành công, cảnh của mô hình được gán cho biến `model`.
* Sau đó, `mô hình` được thu nhỏ xuống 0,5 ở tất cả các kích thước bằng cách sử dụng `model.scale.set(0,5, 0,5, 0,5)`. Thao tác này sẽ thay đổi kích thước mô hình ô tô thành kích thước nhỏ hơn.
* Model được định vị tại tọa độ (0, 0, 0) trong cảnh bằng cách sử dụng `model.position.set(0, 0, 0)`. Thao tác này đặt vị trí ban đầu của mô hình ô tô trong cảnh.
* Model` được thêm vào cảnh bằng cách sử dụng `scene.add(model)`. Thao tác này sẽ thêm mô hình ô tô vào cảnh 3D.
* Tương tự, phiên bản `loader2` tải mô hình "car low poly". Sau khi tải thành công, cảnh của mô hình được gán cho biến `model2`.
* Model2 được thu nhỏ xuống 0,75 ở tất cả các kích thước bằng cách sử dụng `model2.scale.set (0,75, 0,75, 0,75)`. Thao tác này thay đổi kích thước mô hình thành phố thành kích thước nhỏ hơn.
* Model2 được định vị tại tọa độ (0, 0,54, -15) trong cảnh bằng cách sử dụng `model2.position.set (0, 0,54, -15)`. Điều này đặt vị trí ban đầu của mô hình thành phố trong cảnh.
* Model2 được thêm vào cảnh bằng cách sử dụng `scene.add(model2)`. Thao tác này sẽ thêm mô hình thành phố vào cảnh 3D.

A picture containing LEGO, scale model, screenshot

Description automatically generated

Hình 11 : The 3D city

Bằng cách tải và thêm cả mô hình ô tô và thành phố vào cảnh, mô hình ô tô sẽ xuất hiện trong mô hình thành phố, tạo ra một cảnh tổng hợp trong đó ô tô được đặt trong môi trường thành phố.

* + 1. **Xây dựng góc nhìn**

Để điều chỉnh góc xem trong mô hình, bạn có thể thao tác xoay và xoay vị trí của máy ảnh. Trong mã được cung cấp, vị trí camera ban đầu được đặt bằng dòng sau:

```javascript

camera.position.set(-8, 30, 24);

```

A picture containing scale model, screenshot, LEGO, bus

Description automatically generated

Hình 12 : The 3D city with camera.position.set(-8, 30, 24)

Để thay đổi góc xem, bạn có thể sửa đổi các giá trị của `camera.position`. Vị trí máy ảnh được xác định là `(x, y, z)`, trong đó `x` biểu thị vị trí nằm ngang, `y` biểu thị vị trí thẳng đứng và `z` biểu thị vị trí độ sâu.

**Ví dụ,** để thay đổi vị trí của camera để có góc nhìn khác, bạn có thể cập nhật dòng `camera.position.set` như sau:

```javascript

camera.position.set(-10, 20, 30);

```

A picture containing scale model, LEGO

Description automatically generated

Hình 13 : The 3D city with camera.position.set(-10, 20, 30)

Thao tác này sẽ định vị máy ảnh ở các tọa độ `x = -10`, `y = 20` và `z = 30`, điều này sẽ thay đổi góc và phối cảnh mà cảnh 3D được xem.

Ngoài ra, bạn có thể điều chỉnh góc quay của máy ảnh để thay đổi hướng quay của máy ảnh. Bằng cách sửa đổi thuộc tính `xoay` của máy ảnh, bạn có thể thay đổi hướng xoay quanh các trục x, y và z.

**Ví dụ,** để xoay camera để nhìn ở một góc khác, bạn có thể cập nhật dòng `camera.rotation` như sau:

```javascript

camera.rotation.set(-Math.PI/6, 0, 0);

```

A picture containing screenshot, multimedia software

Description automatically generated

Hình 14 : The 3D city with camera.position.set(-Math.PI/6, 0, 0)

Thao tác này sẽ xoay máy ảnh theo `-Math.PI / 6` radian (30 độ) quanh trục x, dẫn đến chế độ xem của máy ảnh nghiêng xuống dưới.

Bằng cách thử nghiệm các giá trị khác nhau cho vị trí và góc quay của máy ảnh, bạn có thể điều chỉnh góc xem và phối cảnh của mô hình 3D trong cảnh.

* + 1. **Xây dựng chức năng điều khiển xe**

Chức năng điều khiển ô tô trong mô hình chịu trách nhiệm xử lý đầu vào của người dùng và cập nhật vị trí của ô tô tương ứng với hàm `updateCarPosition()`:

* Các biến: Biến `speed` xác định tốc độ di chuyển của ô tô và biến `rotationSpeed` điều khiển tốc độ quay của ô tô. Điều chỉnh các giá trị này nếu cần để đạt được chuyển động và xoay mong muốn.

A white car on a street

Description automatically generated with low confidence

Hình 15 : The Car is moving

* Kiểm soát chuyển động: Chức năng kiểm tra trạng thái của các phím cụ thể (`ArrowUp`, `ArrowDown`, `ArrowLeft`, `ArrowRight`) trong đối tượng `keys`, được cập nhật bởi trình xử lý sự kiện cho các sự kiện nhấn phím và nhấn phím. Nếu nhấn phím `ArrowUp`, ô tô sẽ di chuyển về phía trước bằng cách đặt thành phần `z` của `movementVector` thành `speed`. Nếu nhấn phím `ArrowDown`, ô tô sẽ di chuyển lùi lại bằng cách đặt thành phần `z` thành `-speed`. Tương tự, nếu nhấn phím `ArrowLeft`, ô tô sẽ quay sang trái và nếu nhấn phím `ArrowRight`, ô tô sẽ quay sang phải.

A picture containing vehicle, land vehicle, screenshot, car

Description automatically generated

Hình 16 : The car is turning to the right

* Áp dụng Xoay: `movementVector` sau đó được xoay dựa trên vòng quay hiện tại của ô tô (`model2.rotation.y`). Điều này đảm bảo rằng chiếc xe di chuyển đúng hướng dựa trên vòng quay của nó.
* Tính toán vị trí mới: Hàm tính toán vị trí mới của ô tô bằng cách thêm `movementVector` vào vị trí hiện tại của ô tô (`model2.position`). Phương thức `clone()` được sử dụng để tạo một bản sao của vectơ vị trí nhằm tránh sửa đổi vectơ gốc.
* Kiểm tra ranh giới: Vị trí mới được kiểm tra dựa trên các ranh giới đã xác định (`minX`, `maxX`, `minZ`, `maxZ`). Nếu vị trí mới nằm trong ranh giới, vị trí của ô tô được cập nhật lên vị trí mới bằng phương thức `copy()`. Nếu vị trí mới nằm ngoài ranh giới, vị trí của ô tô sẽ được đặt lại về vị trí ban đầu (`0, 0,54, -15`).

A picture containing window, screenshot, house, LEGO

Description automatically generated

Hình 17 : The car returns to its original position

Bằng cách tách logic điều khiển ô tô thành chức năng này, việc hiểu và duy trì mã trở nên dễ dàng hơn. Sau đó, chức năng này được gọi trong vòng lặp hoạt hình để liên tục cập nhật vị trí của ô tô dựa trên đầu vào của người dùng

* + 1. **Tạo nguồn sáng và điều chỉnh góc độ sáng**

Để tạo và điều khiển nguồn sáng trong mô hình 3D bằng Three.js, talàm theo các bước sau:

* Trong mô hình này , em sử dụng sử dụng lớp `DirectionalLight` cho nguồn sáng có hướng để có thể điều chỉnh được cả về nguồn sáng và góc độ chiếu sáng

```javascript

input {DirectionLight} from ‘three';

```

* Tạo một phiên bản của nguồn sáng và cấu hình các thuộc tính của nó, chẳng hạn như màu sắc và cường độ. Ta có thể định vị nguồn sáng trong cảnh bằng thuộc tính `position`.

A picture containing window, screenshot, building

Description automatically generated

Hình 18 : The Brightness slider

```javascript

const light = new DirectionalLight(0xffffff, 2.5); // Color: white, Intensity: 2.5

light.position.set(x, y, z); // Set the position of the light source in the scene

scene.add(light); // Add the light to the scene

```

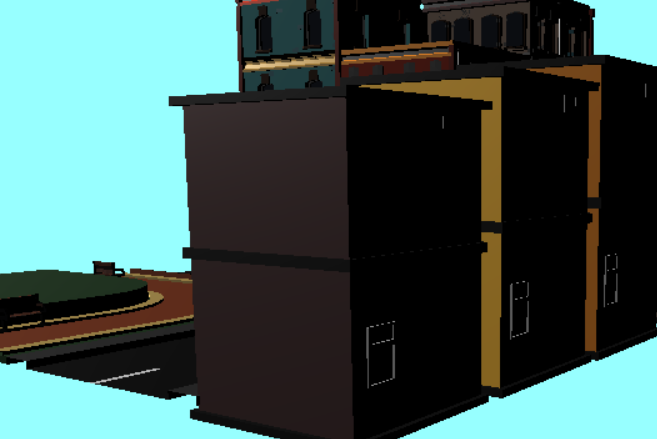
* Thay thế `x`, `y` và `z` bằng các giá trị vị trí mong muốn cho nguồn sáng.
* Nguồn sáng chiếu từ trái sang phải nên các bức tường phía bên trái những ngôi nhà sẽ nhận được ánh sáng.

A picture containing screenshot, window, pc game

Description automatically generated

Hình 19 : The light source from left to right

* Trong khi đó, các bức tường bên phải của những ngôi nhà sẽ không nhận được ánh sáng.



Hình 20 : The light source from right to left

* Để kiểm soát nguồn sáng, bạn có thể cập nhật vị trí của nó hoặc bất kỳ thuộc tính liên quan nào khác. **Ví dụ:** Ta có thể sử dụng thuộc tính `light.position` để thay đổi vị trí của đèn một cách linh hoạt.

```javascript

light.position.set(newX, newY, newZ); // Set new position values for the light source

```

* Thay thế `newX`, `newY` và `newZ` bằng các giá trị vị trí mong muốn mới.
* Khi kết xuất cảnh, đảm bảo bao gồm nguồn sáng trong đối tượng `scene` và gọi phương thức `render` của trình kết xuất với cảnh và máy ảnh làm đối số.

```javascript

scene.add(light); // Add the light to the scene (if not added already)

renderer.render(scene, camera); // Render the scene with the updated light position

``

Bằng cách cập nhật vị trí hoặc thuộc tính của nguồn sáng và kết xuất lại cảnh, bạn có thể kiểm soát hiệu ứng ánh sáng trong mô hình 3D của mình.

# **CHƯƠNG 4: KẾT LUẬN**

## **1.Ưu điểm**

* Tính linh hoạt: Mô hình cho phép dễ dàng tạo và kiểm soát ánh sáng trong cảnh 3D, cung cấp tính linh hoạt để điều chỉnh vị trí, màu sắc và cường độ của nguồn sáng.
* Kết xuất thực tế: Bằng cách định vị và định cấu hình chính xác nguồn sáng, mô hình có thể góp phần tạo hiệu ứng ánh sáng chân thực hơn trong cảnh 3D.
* Tính tương tác: Mô hình kết hợp đầu vào của người dùng để cập nhật liên tục vị trí của nguồn sáng, cho phép tương tác và hiệu ứng ánh sáng động.
* Khả năng tương thích: Mô hình sử dụng thư viện Three.js, đây là một khung được sử dụng rộng rãi và có nhiều tài liệu để tạo đồ họa web 3D. Điều này đảm bảo khả năng tương thích và quyền truy cập vào một cộng đồng lớn để được hỗ trợ và tài nguyên.

## **2. Nhược điểm**

* Mô hình chiếu sáng đơn giản hóa: Mô hình được thảo luận trong tài liệu dường như tập trung vào chiếu sáng định hướng bằng cách sử dụng lớp `DirectionalLight`. Mặc dù điều này cung cấp khả năng kiểm soát góc chiếu sáng, nhưng nó có thể không bao gồm các kỹ thuật chiếu sáng tiên tiến hơn như chiếu sáng toàn cầu hoặc đèn khu vực.
* Thiếu độ phức tạp: Tài liệu không trình bày chi tiết về các kỹ thuật chiếu sáng bổ sung, chẳng hạn như ánh sáng xung quanh, đèn điểm hoặc đèn sân khấu. Tùy thuộc vào các yêu cầu cụ thể của cảnh 3D, mô hình có thể cần được mở rộng để kết hợp các loại ánh sáng này.
* Cân nhắc về hiệu suất: Hiệu ứng ánh sáng thực tế có thể tốn kém về mặt tính toán, đặc biệt là trong các cảnh 3D phức tạp. Các nhà phát triển nên xem xét việc tối ưu hóa việc triển khai mô hình để đảm bảo kết xuất mượt mà và hiệu suất hiệu quả.

## **3. Hướng phát triển**

* Kỹ thuật chiếu sáng nâng cao: Để cung cấp các hiệu ứng ánh sáng nâng cao hơn, các nhà phát triển có thể khám phá việc kết hợp các mô hình chiếu sáng bổ sung như ánh sáng xung quanh, đèn điểm hoặc đèn sân khấu. Điều này sẽ yêu cầu mở rộng triển khai mô hình và điều chỉnh nó để xử lý nhiều nguồn sáng.
* Tích hợp với các mô hình đổ bóng và vật liệu: Ánh sáng và đổ bóng đi đôi với nhau để đạt được kết xuất chân thực. Mô hình này có thể được phát triển thêm để tích hợp với các mô hình tạo bóng, chẳng hạn như Phong hoặc kết xuất dựa trên vật lý (PBR) và các mô hình vật liệu để nâng cao chất lượng hình ảnh của cảnh 3D.
* Tối ưu hóa và cải thiện hiệu suất: Như đã đề cập trước đó, tối ưu hóa việc triển khai mô hình để xử lý các cảnh phức tạp một cách hiệu quả sẽ rất quan trọng. Điều này có thể liên quan đến các kỹ thuật như chọn lọc, quản lý mức độ chi tiết và tối ưu hóa trình đổ bóng để tối đa hóa hiệu suất trong khi vẫn duy trì chất lượng hình ảnh.

# **TÀI LIỆU THAM KHẢO**

1. Tài liệu về tham khảo về blender : <https://www.blender.org>
2. Tài liệu tham khảo về three is : <https://threejs.org>
3. Link hướng dẫn làm low poly Building: <https://www.youtube.com/watch?v=QHWAbVSkYic>