

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO BÀI TẬP LỚN
MÔN: CÔNG NGHỆ THỰC TẾ**

Tên tài: Mô phỏng và chế tạo giao thông

**Giảng viên HD : Th.S. Võ Văn Huy
Lớp : Đại học Kỹ thuật phần mềm 1 – K11
Sinh viên TH : Nhóm 03**

LỜI NÓI ĐẦU

Thực tế là một thuật ngữ mới xuất hiện khoảng 20 năm trước, nhưng ở Mỹ và châu Âu thực tế là (Virtual Reality) đã và đang trở thành một công nghệ mới như thế nào? khi nào? ứng dụng rộng rãi trong môi trường (nghiên cứu và công nghiệp, giáo dục và đào tạo, thể thao, giải trí,...) và tiềm năng kinh tế, ứng dụng tính ứng dụng (trong dân chúng và quân sự) của nó. Tại Việt Nam, tuy là một lĩnh vực mới nhưng đã có những công trình rất hữu ích như: tái hiện lại con Sao La hay một Vườn Miêu Quê Tả Giám mà ta có thể đi lại quan sát trong đó.

Chính vì tầm quan trọng của nó khi nào? ứng dụng to lớn đó nên việc nghiên cứu và thực tế là vô cùng cần thiết. Và trên cơ sở đó có thể xây dựng một ứng dụng thực tế là hoàn chỉnh.

Với môn công nghệ thực tế là chúng em đã học, mang lại cho chúng em nhiều kiến thức bổ ích, hiểu rõ về công nghệ thực tế là của các ứng dụng rộng rãi của nó, nắm bắt các kiến thức cơ bản về ngôn ngữ mô phỏng thực tế là VRML. Vì hiểu rõ môn học và nắm vững kiến thức ngôn ngữ VRML chúng em đã thực hiện đề tài “Mô phỏng và chương trình giao thông”. Bài tập lớn này gồm có phần mở đầu, kết luận và nội dung:

Chương 1: Khảo sát, xây dựng kịch bản

Chương 2: Các kỹ thuật sử dụng trong VRML

Chương 3: Kết quả mô phỏng và chương trình giao thông

Chúng em xin gửi lời cảm ơn chân thành nhất tới thầy V. Huy - người đã truyền đạt cho chúng em những kiến thức về môn học và chúng em hoàn thành bài báo cáo này.

Nhóm 3

CHƯƠNG 1: KHẢO SÁT, XÂY DỰNG KỊCH BẢN

Đề tài: Mô phỏng và chương trình giao thông

1.1.1 Yêu cầu bài toán

Với những công nghệ to lớn mà ngôn ngữ thực tế là đem lại và nhóm mô phỏng lại các tình huống xảy ra tại nút giao thông tại các ngã tư hiện không có đèn tín hiệu: tại nút ngã 2 xe ô tô, ngã 2 xe ô tô và xe máy và tại nút ngã 2 xe máy với nhau. Việc mô phỏng giúp cho chúng ta cân nhắc kỹ việc di chuyển phương tiện một cách cẩn thận, vì trường hợp xảy ra nào cũng có thể xảy ra khi tham gia giao thông.

Tóm tắt về đề tài trên, nhóm chúng tôi đã bắt tay vào tìm hiểu và xây dựng mô hình: “**Mô phỏng và chương trình giao thông**”.

Mở đầu và chương 1



Màu và chiều s? 2

Màu và chiều s? 3

Hình ảnh ???c tham khảo qua trang tìm kiếm: <https://www.google.com.vn> với từ khóa: “Va chạm giao thông”

1.3 Các phần mềm hỗ trợ

- VRMLPAD 3.0
- Autodesk 3ds Max 2016
- Cortona 3D
- Adobe Photoshop CC

CHƯƠNG 2: CÁC KỸ THUẬT SỬ DỤNG TRONG VRML

2.1. Các thành phần cơ bản của môi trường VR

Tổng quát môi trường VR bao gồm những thành phần sau:

Hình 1.1 Các thành phần cơ bản của môi trường VR

2.1.1 Phần cứng (Hardware)

Phần cứng của môi trường VR bao gồm:

2.1.2 Phần mềm (Software)

Phần mềm luôn là linh hồn của VR cũng như vậy với bất kỳ môi trường máy tính hiện nay. Với môi trường nguyên tắc có thể dùng bất kỳ ngôn ngữ lập trình hay phần mềm nào để mô hình hóa (modelling) và mô phỏng (simulation) các đối tượng của VR.

Ví dụ như các ngôn ngữ (miễn phí) OpenGL, C++, Java3D, VRML, X3D,... hay các phần mềm thương mại như WorldToolKit, PeopleShop,... Phần mềm của bất kỳ VR nào cũng phải dựa trên 2 công nghệ chính: Tạo hình và Mô phỏng. Các đối tượng của VR được mô hình hóa như chính phần mềm này hay chuyển sang tạo các mô hình 3D (thiết kế như các phần mềm CAD khác như AutoCAD, 3D Studio,...). Ngoài ra, phần mềm VR còn có khả năng mô phỏng động học, tĩnh học, và mô phỏng phần cứng của đối tượng.

2.2. Giới thiệu các ngôn ngữ VRML

2.2.1. Định nghĩa VRML

VRML (Virtual Reality Modeling Language) là ngôn ngữ mô hình hóa thực tại ảo, một định dạng tệp tin được sử dụng trong việc mô tả thế giới thực và các đối tượng ảo dựa trên tác ba chiều, sử dụng mô hình phân cấp trong việc thể hiện tác động với các đối tượng của mô hình, được thiết kế dùng trong môi trường Internet, Intranet và các hệ thống máy khách cục bộ (local client) mà không phải được vào hệ thống hành.

Các ứng dụng 3D của VRML có thể truy cập một cách dễ dàng trên mạng với kích thước khá nhỏ so với băng thông, phần lớn giới hạn trong khoảng 100 - 200KB. Nếu HTML là định dạng văn bản thì VRML là định dạng đối tượng 3D có thể tác động và biến đổi khi cần thiết.

Hiện nay, VRML có lợi thế là sử dụng giao thức, hỗ trợ dịch vụ Web3D, có cấu trúc chặt chẽ, với khả năng mạnh mẽ, giúp cho việc xây dựng các ứng dụng ảo dựa ba chiều một cách nhanh chóng và thuận tiện.

VRML là một trong những chuẩn trao đổi đa năng cho dữ liệu ba chiều tích hợp và truy cập thông tin đa phương tiện, được sử dụng trong rất nhiều lĩnh vực ứng dụng, chẳng hạn như trực quan hóa các khái niệm khoa học và kỹ thuật, trình diễn đa phương tiện, giải trí và giáo dục, hỗ trợ web và chia sẻ các thế giới ảo. Với mục đích xây dựng định dạng chuẩn cho phép mô tả thế giới thực trên máy tính và cho phép chạy trên môi trường web, VRML đã trở thành chuẩn ISO từ năm 1997.

2.2.2. Các tiêu chuẩn

Tiêu chuẩn cho việc xác định đối tượng 3D, quang cảnh và cho sự liên kết các mô hình với nhau là: VRML được thiết kế dành riêng cho việc thể hiện thế giới 3D và không phải là sự mở rộng của HTML.

2.2.3. Các đặc tính chính của VR

Đặc tính chính của VR là Tương tác (Interactive) và Immersion (Immersion). Tuy nhiên, VR không chỉ là một hệ thống tương tác Ngõ - Máy, mà các ứng dụng của nó còn liên quan tới việc quy định các vấn đề thể hiện trong kỹ thuật, ý thức, quân sự,... Các ứng dụng này phải được rất nhiều vào khả năng tưởng tượng (Imagination) của con người. Do đó có thể coi VR là tổng hợp của 3 yếu tố: Tương tác - Immersion - Tưởng tượng (“3I” trong tiếng Anh: Interactive – Immersion - Imagination).

Hình 1.2 Các đặc tính chính của VR

2.2.4. Một số ứng dụng chính của VR

Từ các nhu cầu phát triển, VR được ứng dụng trong nhiều lĩnh vực: Khoa học kỹ thuật, kiến trúc, quân sự, giải trí,... và đáp ứng nhiều nhu cầu: Nghiên cứu - Giáo dục - Thể thao. Ý thức là lĩnh vực ứng dụng truy cập thế giới của VR.

Ngoài ra, VR cũng đã được ứng dụng trong giáo dục, nghệ thuật, giải trí. Được biết trong lĩnh vực quân sự, VR đã được ứng dụng rất nhiều ở các nước phát triển hiện nay. Bên cạnh các ứng dụng truyền thống ở trên, cũng có một số ứng dụng mới nổi lên trong thời gian gần đây của VR như: ứng dụng trong săn bắn, ứng dụng trong ngành robot, ứng dụng trong huấn luyện thông tin (thậm chí dò dẫm mìn, huấn luyện thông tin khẩn cấp, ...). VR có tiềm năng ứng dụng vô cùng lớn, hứa hẹn các lĩnh vực “có thể” trong cuộc sống nếu có thể ứng dụng “thực tiễn” nghiên cứu và phát triển hoàn thiện hơn.

- Trong y học

Thực tiễn cho thấy quy trình rất nhiều vấn đề trong y học: cung cấp môi trường thực hành cho nghiên cứu và học tập, rất hữu ích trong việc mô phỏng các ca phẫu thuật nhằm giảm thiểu rủi ro trong thực tế.

Hình 1.3 Một ca phẫu thuật trong thực tiễn

Vì sự trợ giúp của thực tiễn, ngày nay, con người không những có thể xem được hình ảnh trực quan của các thiết bị y tế săn bắn mà thậm chí người ta còn có thể dùng hay thay thế các thiết bị của các thiết bị y tế. Việc này nhằm giúp cho các nhà khoa học và các kỹ sư thu thập dữ liệu trong việc tạo ra một sản phẩm hợp ý muốn mà không cần tốn nhiều chi phí.

Hình 1.4 Các kỹ sư đang thay thế các thiết bị cho chiếc xe hơi

Trong nhiều năm trở lại đây, thực tiễn đã được xây dựng mô hình của các dự án kiến trúc trước khi các dự án này đưa vào thực tế, nhằm giúp cho người sử dụng có cái nhìn tổng quan và chi tiết về các dự án đó. Bên cạnh đó, thực tiễn cũng được tái hiện lại các công trình kiến trúc cũ, nhằm lưu giữ lại các di sản văn hóa...

Hình 1.5 Một góc của Viện Bảo tàng T. Giám trên mô hình 3D

- Trong quân sự

Vì việc phát triển của VR, các binh sĩ sẽ được huấn luyện một cách trực quan nhất các kỹ năng cần thiết như: lái máy bay, lái xe tăng,... trước khi tham gia công việc thực tế. Việc này vừa bảo đảm an toàn cho binh sĩ, vừa tiết kiệm được chi phí cho các khóa huấn luyện thực tế. Quân đội Mỹ đã phát triển một game được biết đến như huấn luyện binh sĩ chiến đấu khốc liệt để dùng để huấn luyện thực tiễn (Hình 1.6). Đây là một game rất sống động, có tính hành động cao với môi trường và bối cảnh bám sát với thực tế. Những người lính sẽ phải vận dụng tất cả những kỹ năng đã được rèn giũa trong quân đội.

Hình 1.6 Ứng dụng của VR trong quân sự

Các nước phương Tây vì qua Internet không còn là vùng đất mới. Công nghệ VR sẽ làm cho việc này trở nên thú vị hơn rất nhiều. Người học có thể đi vào khiếm khuyết nhân vật để đi đến cho mình vị trí trong một trường học để xây dựng trên máy tính. Người học cũng có thể tham gia vào bất cứ lớp học nào mà họ thích và nói chuyện với những thành viên khác trong lớp.

Hình 1.7 Cảnh sinh hoạt trong lớp học

Game thực tiễn hiện nay đã trở thành một ngành công nghiệp thu được nhiều lợi nhuận. Vì vậy ta hiện nay thì game thực tiễn của được biết đến nhiều song một số nước phát triển thì đây là một ngành giải trí thu lợi nhuận khổng lồ, ví dụ các nước Mỹ, Nhật, Anh, ...

Hình 1.8 Game Nintendo Wii

Như vậy thực tiễn có ứng dụng trong hứa hẹn các lĩnh vực của cuộc sống. Qua đó cũng nhận thấy được ý nghĩa to lớn của việc ứng dụng thực tiễn. Với những vấn đề khó khăn, nếu không có thực tiễn

Cho thì rất khó gì? quy? t? ho?c hi?u qu? không cao mà chi phí t?n kém.

2.3. T?p tin c?a VRML

T?p tin c?a VRML có ph?n m? r?ng là “.wrl” v?i các ph?n nh? sau:

2.4. Các nút trong VRML

T?p tin VRML ???c xây d?ng d?a trên t?p các ??i t???ng nh?m ???n các m?c ?ích khác nhau. Thông th???ng các ??i t???ng có các thu?c tính v?t lý c?a mình nh? hình d?ng, màu s?c, t?a ?? ?i?m, ... ?? mô t? cho các ??i t???ng c?a th? gi?i th?t, VRML s? d?ng thu?t ng? “Nút - Node” ?? bi?u di?n chúng.

Nút là kh?i c? s? c?a t?p tin VRML dùng ?? mô t? nh?ng ??i t???ng mà thu?c tính c?a chúng ???c ???nh ngh?a trong nút ?ó. Nút có th? là các ??i t???ng hình h?c nh? hình h?p, hình nón, hình tr? ... hay các ??i t???ng khác nh? màu s?c, ánh sáng, âm thanh. S? t?n t?i c?a nút trong t?p tin VRML có th? là m?t c?u trúc c? b?n ???ng ???n l? ho?c có th? ch?a nhi?u các nút có liên h? v?i nhau.

D? li?u c?a nút ???c l?u gi? b?i các tr???ng (Field) trong nút, tuy nhiên ta có th? khai báo ch? m?t nút trong file nh?ng không th? ch? ??a ra m?t tr???ng ???n l? mà b?t bu?c ph?i ?? trong m?t nút nào ?ó. V? m?t khía c?nh nào ?ó nút t???ng ???ng v?i m?t l?p (class) trong các ngôn ng? l?p trình h???ng ??i t???ng (Java). VRML bao g?m 54 nút khác nhau và ???c phân lo?i làm 9 nhóm chính d?a trên ch?c n?ng và các hàm c?a các nút.

Bao g?m:

Anchor

Billboard

Collision

Group

Transform.

CylinderSensor

PlaneSensor

ProximitySensor

SphereSensor

TimeSensor

TouchSensor

VisibilitySensor.

Box

Cone

Cylinder

ElevationGrid

Extrusion

IndexedFaceSet

IndexedLineSet

PointSet

Sphere

Text.

Inline

LOD

Switch.

Color

Coordinate
Normal
TextureCoordinate.
Appearance
FontStyle
ImageTexture
Material
MovieTexture
PixelTexture
TextureTransform.
ColorInterpolator
CoordinateInterpolator
NormalInterpolator
OrientationInterpolator
PositionInterpolator
ScalarInterpolator.
Background
Fog
NavigationInfo
Viewpoint.

Tên các nút trong VRML thường bắt đầu bằng chữ in hoa và có thể là một trong các tên chuẩn do VRML cung cấp, các trường của nút thường bắt đầu là chữ thường, mỗi loại nút có các trường khác nhau. Giá trị của trường có thể là các giá trị thực hoặc các biến giá trị thực hoặc có thể là một nút con. Có thể hình dung các nút như các lớp trong lớp trình hướng dẫn. VRML không cho phép định nghĩa thêm các nút mới mà chỉ định dùng các nút con trong chín nhóm nút đã nêu.

Nút **WorldInfo** là nút chứa thông tin chung về thế giới như tiêu đề của thế giới hay một chuỗi thông tin về tác giả hoặc về nội dung của tệp tin, nút này không ảnh hưởng đến hình ảnh cũng như các sự kiện trong thế giới (nút này có thể không định trong tệp tin VRML). Các trường của nút này định nghĩa tên hoặc tiêu đề của thế giới và trình duyệt có thể hiển thị cho người dùng hoặc phục vụ cho các công cụ tìm kiếm.

Sau đây là một ví dụ của nút **WorldInfo**:

```
WorldInfo {  
    title "Hello VRML"  
    info ["Virtual Reality Modeling Language"  
        "Vu Thi Mai"]  
}
```

Nút **Sharp** là nút con sử dụng để chứa các đối tượng hình học thông qua "Geometry nodes" và các thuộc tính của thế giới hiển thị của đối tượng hình học đó qua "Appearance nodes". Vì vậy có nghĩa nếu ta muốn tạo ra một đối tượng nào thì nút hình dáng của đối tượng đó phải được tạo ra trước.

Ví dụ:


```
Shape {
    appearance Appearance {
        material Material { diffuseColor 0.8941 0.7216 0.6}
    }
    geometry Box {
size 2 2 2
    }
}
```

2.4.1. Các ??i t??ng hình h?c c? b?n

2.4.1.1. Hình h?p (Box)

Ví d?:

```
Shape {
geometry Box{ size 2.0 2.0 2.0 }
}
```

Tham s?:

2.4.1.2. Hình nón (Cone)

Ví d?:

```
Shape {
geometry Cone {
height 2.0
bottomRadius 1.0
bottom TRUE
side TRUE
}
}
```

Các tham s?:

2.4.1.3. Hình c?u (Shpere)

Ví d?:

```
Shape {
geometry Sphere{ radius 1.0 }
}
```

Các tham s?:

2.4.1.4. Hình tr? (Cylinder)

Ví d?:

```
Shape {
  geometry Cylinder{
    height 2.0
    radius 1.0
    bottom TRUE
    top TRUE
    side TRUE
  }
}
```

Các tham số:

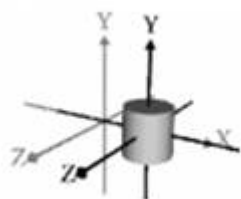
2.4.2. Các phép biến đổi

Một cảnh, một vật thể và hình ảnh được xây dựng trong VRML đều được kết nối với nhau và các phép biến đổi được dùng để tạo ra một hình ảnh thực tế mà người mà hình ảnh này có vị trí tương đối với hình ảnh thực tế. Có các phép biến đổi như: dịch chuyển, co giãn, quay. Ngoài ra, nếu các hình ảnh được xây dựng trùng lên nhau như hình 2.2 có thể giúp chúng ta tạo ra một hình ảnh phức tạp mà không cần phải dùng đến nút **ElevationGrip**.

Nút **Transform** quản lý các phép biến đổi trong VRML như: dịch chuyển, co giãn, quay.

Nút **Transform** gồm có các trường con:

2.4.2.1. Phép dịch chuyển (Translation)



Cú pháp:

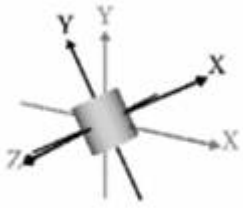
```
Transform {
  translation X Y Z
  children []
}
```

Các tham số:

Ví dụ:

```
Transform {
  translation 1 0 1
  children [
    Shape {
      geometry Sphere{ radius 1.0 }
    }
  ]
}
```

2.4.2.2. Phép quay (Rotation)



Cú pháp:

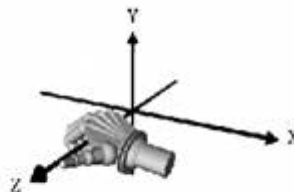
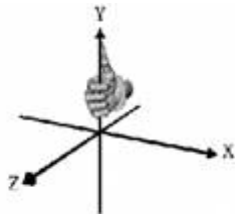
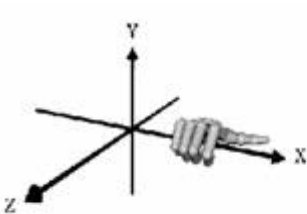
```
Transform {  
    rotation X Y Z G  
    children []  
}
```

Các tham số:

Ví dụ:

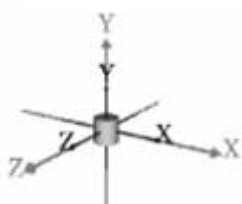
```
Transform {  
    rotation 1 0 0 1.57  
    children [  
Shape {  
geometry Box{ size 2 1 0.5}  
}  
]  
}
```

Lưu ý: quy ước, chiều quay theo chiều kim đồng hồ xác định bởi quy tắc bàn tay phải.



(Nếu bàn tay phải trái, choãi ngón cái ra theo như hình vẽ. Ngón tay cái chỉ phương của trục quay thì chiều tay phải các ngón tay chỉ chiều quay theo chiều kim đồng hồ khi quay theo chiều kim đồng hồ).

2.4.2.3. Phép co giãn (Scale)



Cú pháp:

```
Transform {  
    Scale x y z  
    children []  
}
```

Các tham số:

Ví dụ:

```
Transform {  
    Scale 3 2 0.02  
    children [  
Shape {  
geometry Sphere{ radius 1.0 }  
}  
]  
}
```

2.4.3. Màu sắc và hình ảnh

Các đối tượng trong thế giới ảo khi được tạo ra sẽ có màu mặc định là màu trắng nhưng chúng ta có thể thay đổi chúng như vào nút **Appearance** được cung cấp sẵn trong VRML, nút này được sử dụng để quy định các thành phần xuất hiện trên đối tượng như màu sắc, hình ảnh, video,... Nút **Appearance** bao gồm các nút con như: **Material**, **ImageTexture**, **MovieTexture**...

2.4.3.1. Màu sắc trong VRML (Material)

Nút **Material** có chức năng tạo màu cho các đối tượng trong VRML. Màu sắc trong VRML được thể hiện qua 3 tham số là R G B (Red-Green-Blue) với giá trị nằm trong khoảng từ 0.0 đến 1.0. Nút **Material** có các thuộc tính sau:

Ví dụ:

```

Shape {
  appearance Appearance {
    material Material {
      diffuseColor 1.0 0.5 0.0
      emissiveColor 0.0 0.0 0.0
      transparency 0.5
      specularColor 0.5 0.5 0.5
      shininess 0.3
      ambientIntensity 0.4
    }
  }
  geometry Box { size 1.0 1.0 1.0}
}

```

2.4.3.2. Vẽ trên hình ảnh (PixelFormat)

Ngoài việc tô màu cho mặt hình ảnh nào đó, trong VRML còn cho phép ta vẽ hình trên bề mặt hình ảnh đó, hình vẽ được tạo ra từ tập hợp các điểm ảnh. Nút PixelTexture sẽ giúp ta thực hiện công việc này.

Ví dụ:

```

Shape {
  appearance Appearance {
    material Material {}
    texture PixelTexture {
      image 2 4 3 0xFF0000 0xFF00 0 0 0 0 0xFFFF 0xFFFF00
      repeatS TRUE
      repeatT TRUE
    }
  }
  geometry Cylinder {height 5}
}

```

Các thuộc tính:

Trong ví dụ trên ta vẽ mặt bên hình trụ 8 điểm ảnh (mỗi điểm ảnh có các màu sẽ được chèn ra bởi các số thập lục phân trong ví dụ ở trên), sau đó vẽ hình ảnh kéo giãn ra bề mặt với kích thước của hình trụ và dán lên hình trụ đó.

2.4.3.3. Dán hình ảnh (ImageTexture)

Thay vì ph?i m?t th?i gian ?? v? hình lên ??i t??ng nh? trên ta có th? l?y m?t b?c ?nh có s?n dán lên b? m?t ??i t??ng nh? là dán decal. K? thu?t này ???c g?i là ánh x? k?t c?u (texture mapping).

Trong VRML có th? s? d?ng m?t trong các ??nh d?ng ?nh sau ?ây:

Ví d?:

```
Shape {
  appearance Appearance {
    material Material {}
    texture ImageTexture {
url “TV.jpg”
repeatS TRUE
repeatT TRUE
}
}
  geometry Box { size 3 4 5}
}
```

Các thu?c tính:

? ví d? này chúng ta l?y hình ?nh “hinha.jpg” t? bên ngoài thông qua thu?c tính **url** dán lên ??i t??ng, các thu?c tính **repeatS** và **repeatT** t??ng t? nh? nút **PixelTexture**.

2.4.3.4. Chi?u phim lên b? m?t ??i t??ng (MovieTexture)

Ngoài vi?c cho phép v? hay dán hình lên ??i t??ng, VRML còn h? tr? vi?c chi?u m?t t?p tin video trên ??i t??ng.

VRML ch? h? tr? ??nh d?ng video MPEG (bao g?m MPEG1-Systems (âm thanh và hình ?nh) và MPEG1-Video (ch? có hình ?nh)).

Cú pháp:

```
MovieTexture {
  loop    FALSE
  speed   1.0
startTime 0
  stopTime 0
  url     []
  repeatS  TRUE
  repeatT  TRUE
}
```

Các thu?c tính:

Ví d?:

```

Shape {
  appearance Appearance {
    material Material {}
    texture MovieTexture {
      url "phim.mp4"
      loop TRUE
    }
  }
  geometry Cylinder {height 5}
}

```

2.4.4. Âm thanh, ánh sáng, camera và phong cảnh

2.4.4.1. Âm thanh

?? làm cho th? gi?i ?o tr? nên sinh ??ng và h?p d?n h?n, chúng ta nên thêm âm thanh vào ?ó. Ta có th? t?o ra âm thanh n?n, ti?ng chuông c?a, âm thanh khi m? c?a ho?c b?t c? âm thanh nào mà mình mu?n có trong th? gi?i ?o. T?t c? nh?ng ?i?u ?ó ???c th?c hi?n b?i hai nút do VRML cung c?p là nút **Sound** và nút **AudioClip**. Có th? hi?u ??n gi?n nh? sau: nút **Sound** ???c dùng ?? xác ??nh v? trí phát ra âm thanh trong th? gi?i ?o và âm thanh ?ó ???c xác ??nh b?i nút **AudioClip**.

2.4.4.2. Ánh sáng

Khi quan sát m?t v?t th? b?t k?, các v?t th? ?ó ch? ???c chi?u sáng b?i m?t ánh sáng xung quanh. Đây là m?t ánh sáng ??c bi?t ???c t?o ra b?i trình duy?t. Tuy nhiên, chúng ta c?ng có th? ?i?u khi?n ánh sáng này thông qua nút **NavigationInfo**, đây là m?t nút ???c s? d?ng ?? xác ??nh các thu?c tính c?a ng??i s? d?ng. Tuy nhiên ?? có ???c m?t th? gi?i ?o trông th?t h?n thì chúng ta nên t?o ra các ngu?n sáng nh? trong th? gi?i th?t, ví d? nh?: ánh sáng m?t tr?i, ánh sáng c?a ?èn ?i?n,... Và VRML h? tr? cho chúng ta các lo?i ánh sáng ?ó thông các nút **DirectionalLight**, **PointLight**, **SpotLight**.

2.4.4.3. Camera

Khi ta m? m?t t?p tin VRML thì trình duy?t s? h? tr? m?c ??nh cho chúng ta m?t góc nhìn, ?? tùy bi?n góc nhìn ?ó ta có th? s? d?ng nút **Viewpoint**. Nút **Viewpoint** có ch?c n?ng xác ??nh v? trí c?a ng??i dùng trong th? gi?i và các thông s? c?a góc nhìn. V?i nút này chúng ta có th? tùy bi?n v? trí và các ??c tính quan sát c?a ng??i dùng khi th? gi?i ???c kh?i ??ng lên, vì?c ?ó s? giúp cho ng??i xem nh? ?ang ? trong th? gi?i th?c. Nút này có các thu?c tính nh? sau:

Ví d?:

```
Viewpoint {
  fieldOfView 0.785398
  position 0 0 10
  orientation 0 0 1 0
  description "Góc nhìn s? 1"
  jump TRUE
}
```

2.4.4.4. Phong c?nh n?n và n?i tr??ng

Vi?c xây d?ng m?t th? gi?i ?o g?m t?t c? các ??i t??ng gi?ng nh? v?i th? gi?i th?c là chuy?n không th? vì ta không có ?? th?i gian ?? th?c hi?n vi?c ?ó. Vì th? chúng ta ch? có th? xây d?ng m?t th? gi?i ?o gi?ng m?t ph?n (khu v?c) nào ?ó c?a th? gi?i th?c. V?i các khu v?c xung quanh, n?u chúng ta ?? tr?ng các khu v?c này thì th? gi?i c?a chúng ta s? không gi?ng nh? th?c ???c. ?ây th?t s? là m?t v?n ?? khó kh?n, và VRML cung c?p cho chúng ta m?t công c? giúp che ph? các ph?n còn l?i ?ó ?? th? gi?i c?a chúng ta trông th?c h?n. Nút **Background** ???c cung c?p nh?m mô t? các ???ng chân tr?i c?a b?n, nó cho phép chúng ta xác ??nh b?u tr?i, m?t ??t và các hình ?nh xung quanh th? gi?i c?a chúng ta.

2.4.5. Nhóm ??i t??ng

2.4.5.1. Anchor

Anchor: nút này có ch?c n?ng xác ??nh m?t t?p h?p các ??i t??ng và g?n m?t siêu liên k?t ??n m?t url, ch?ng h?n nh? là m?t siêu liên k?t ??n th? gi?i VRML khác, ??n m?t trang HTML ho?c ??n m?t d? li?u nào ?ó mà trình duy?t có th? ??c ???c. Khi kích chu?t vào m?t trong các ??i t??ng bên trong nút Anchor thì toàn b? trình duy?t s? ??a ??n ??a ch? url. Khi ??a chu?t lên m?t ??i t??ng nào ?ó trong nút này thì có th? nhìn th?y ???c dòng ghi chú c?a nó. T?t c? các nút trong nút **Anchor** ??u hi?n th?.

Ví d?:

```
Anchor {
  children [
    Shape {
      geometry Sphere {}
    }
  ]
  url "http://ninkuhack.blogspot.com/"
  description "Blog c?a tôi!"
  parameter ["target=my_frame" ]
  bboxCenter 0 0 0
  bboxSize -1 -1 -1
}
```

Các thu?c tính:

2.4.5.2. Group

Group: tạo ra một tập hợp các đối tượng nh? là một thực thể duy nhất. Nút này có các trường: **children**, **bboxCenter**, **bboxSize**. Các trường này tương tự như của nút **Anchor**. Tất cả các nút con nằm trong nút **Group** đều hiển thị.

Ví dụ:

```
Group {  
  children [...]  
  bboxCenter 0 0 0  
  bboxSize -1 -1 -1  
}
```

2.4.5.3. Switch

Switch: tạo ra một nhóm chuyển đổi gồm tập hợp các nút con, chỉ có một nút con trong nhóm được hiển thị và nút này là do người dùng lựa chọn. Các nút con được ánh xạ theo giá trị 0, nếu trường **whichChoice** có giá trị -1 tức là không chọn nút con nào.

Ví dụ:

```
Switch {  
  whichChoice -1  
  choice [...]  
}
```

Các thuộc tính:

2.4.5.4. Transform

Transform: tạo ra một nhóm các đối tượng và tất cả chúng đều hoạt động theo một mô hình.

2.4.5.5. Inline

Inline: tạo ra một nhóm các đối tượng được biết đến là một tập tin VRML khác (được chỉ ra sau **url**). Tất cả các đối tượng trong tập tin này sẽ được hiển thị. Nút này thường được sử dụng để gọi trực tiếp một đối tượng bên ngoài vào để hiển thị.

Ví dụ:

```
Inline {  
  url ["cuaso.wrl"]  
  bboxCenter 0 0 0  
  bboxSize -1 -1 -1  
}
```

Các thuộc tính:

2.4.6. Tái s? d?ng ??i t??ng

2.4.6.1. Inline

Inline: s? d?ng nút **Inline** ?? g?i tr?c ti?p m?t ??i t??ng bên ngoài t?p tin vào t?p tin hi?n t?i. Nút này ?ã ???c trình bày ? trên.

2.4.6.2. DEF

DEF (DEFine): cho phép ta ??nh ngh?a m?t ??i t??ng hay m?t ki?u thu?c tính.

Ví d?:

T?o ??i t??ng

```
Shape {  
  appearance Appearance {  
    material DEF RedColor Material {  
      diffuseColor 1.0 0.0 0.0  
    }  
  }  
  geometry Box { size 5 5 5 }  
}
```

S? d?ng:

```
Shape {  
  appearance Appearance {  
    material USE RedColor  
  }  
  geometry Sphere { }  
}
```

Nh? ví d? trên ta ??nh ngh?a ??i t??ng RedColor ki?u **Material**. Sau ?ó mu?n s? d?ng l?i ta ch? g?i hàm b?ng t? khóa **USE**.

2.4.7. N?i suy và c?m bi?n

2.4.7.1. Các nút n?i suy (Interpolators nodes)

Các nút thu?c nhóm nút **Interpolators** có ch?c n?ng gi? các giá tr? xen vào các tr??ng khi có s? ki?n thay ??i giá tr? các tr??ng x?y ra (tr? các tr??ng có giá tr? ki?u logic). Các nút thu?c nhóm nút này có cú pháp gi?ng nhau (có cùng các tr??ng) ch? khác nhau v? ki?u d? li?u.

Trong VRML cung c?p s?n các nút **Interpolators** sau:

M?i nút ??u có cú pháp nh? sau:

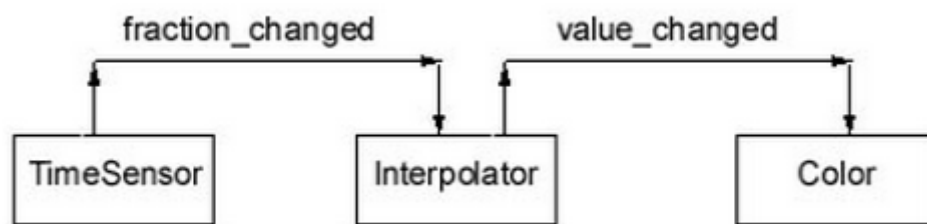
```

...Interpolator {
key [...]
keyValue [...]
}

```

Trong đó **key** là tập các giá trị đầu vào còn **keyValue** là tập các giá trị đầu ra. Trong các nút **Interpolators** đầu có một số kiến đầu vào là **set_fraction** và số kiến đầu ra là **value_changed**, hai số kiến này liên quan với nhau tức là khi các nút này khi nhận được một số kiến thì nó cũng tạo ra một số kiến. Số kiến **set_fraction** xác định một giá trị **key** và số kiến đầu ra xác định một **keyValue** liên quan với giá trị **key**.

Các nút **Interpolators** thường liên kết với **TimeSensor** để tạo ra các hoạt động. Xem hình vẽ dưới đây để hiểu rõ hơn về các nút **Interpolators**.



Như trong hình vẽ là một mô hình về nút **ColorInterpolator** (các nút khác cũng tương tự như vậy). Vậy khi nút **TimeSensor** được kích hoạt thì nó sẽ gửi các giá trị liên tiếp cho các nút **Interpolators** và các nút này cũng sẽ gửi liên tiếp các giá trị cho một nút khác, theo hình vẽ thì nút **ColorInterpolators** sẽ gửi liên tiếp các giá trị màu sắc theo thời gian nào đó trong thời gian.

Ví dụ:

```

OrientationInterpolator {
    key [ 0.0, 0.50, 1.0 ]
    keyValue [
        0.0 1.0 0.0 0.0,
        0.0 1.0 0.0 3.14,
        0.0 1.0 0.0 6.28
    ]
}

```

Trong ví dụ này nếu nút này nhận một giá trị là 0.0 thì nó sẽ gửi ra giá trị (0.0 1.0 0.0 0.0) và nếu nhận giá trị từ khoảng 0.0 đến 0.5 thì nó sẽ tính toán và gửi ra giá trị nằm trong khoảng 0.0 đến 3.14 (chính xác là vậy nếu nó nhận được giá trị là 0,3 thì nó sẽ gửi ra giá trị là 1,88), và tương tự với các giá trị khác.

2.4.7.2. Nhúng mã vào VRML (Script)

Nút **Script** mở ra khả năng linh hoạt, mở rộng cho phép thực hiện những công việc mà chúng ta cũng như hành động mong đợi phần mềm thực hiện. Nút **Script** cho phép chúng ta xác định việc thực hiện tác vụ thực hiện theo những cách sử dụng các ngôn ngữ lập trình như Java, Javascript hoặc VRMLScript (được đưa ra bởi Silicon Graphics, Inc).

Trong nút **Script** có thể có các trigger hay các thuộc tính, nhưng không giống với các nút khác, trong nút **Script**, chúng ta có thể xác định các sự kiện mà trigger này có thể nhận và gửi. Một nút **Script** sẽ thực hiện một hành động nào đó mỗi khi nó nhận một sự kiện, trong mỗi hành động đó nút **Script** có thể tạo ra nhiều sự kiện. Hơn nữa nút **Script** có thể thực hiện xây dựng các thủ tục khởi tạo (initiazation procedures), đóng và tắt (shutdown procedures) các tiến trình trong thế giới ảo.

Một nút **Script** có thể nhận nhiều sự kiện, các giá trị mà các sự kiện này tạo ra là các biến thuộc loại **eventIn**. Tên của các biến này là tên của các sự kiện mà nút **Script** tiếp nhận. Các biến này là chỉ có thể đọc, nghĩa là chúng ta không thể thay thế giá trị trực tiếp cho bất kỳ biến nào thuộc loại này. Một nút **Script** cũng có thể tạo ra nhiều sự kiện, các giá trị mà các sự kiện này tạo ra là các biến thuộc loại **eventOut**.

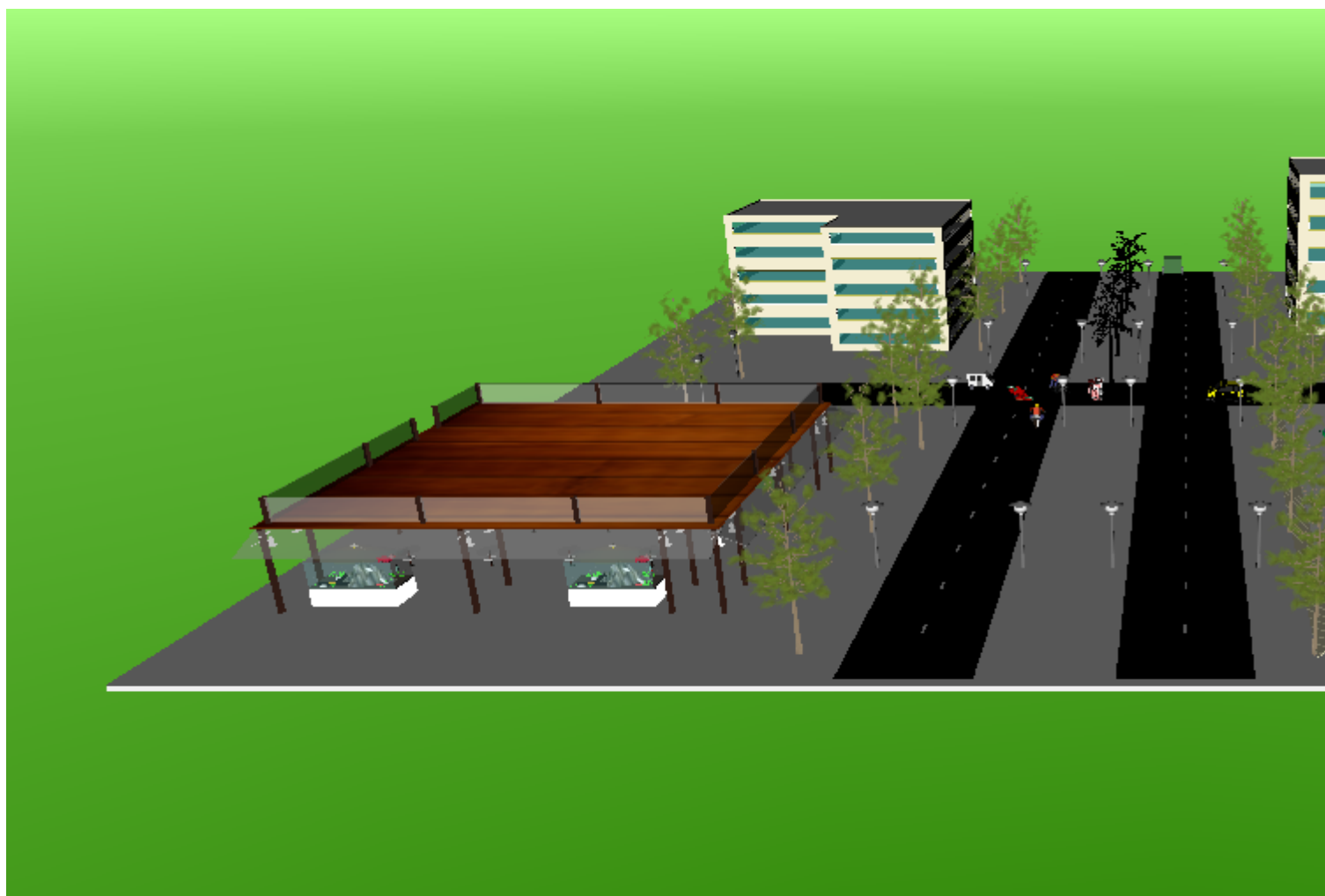
Cú pháp:

```
Script
{
url []
directOutput FALSE
mustEvaluate FALSE
eventIn Datatype EventName
eventOut Datatype EventName
field Datatype FieldName InitialValue
}
```

Các tham số:

CHƯƠNG 3: HÌNH THÀNH SẢN PHẨM SAU KHI MÔ PHỎNG MÔ HÌNH VÀ CHẾM GIAO THÔNG

3.1. Hình thành từng quan mô hình



Hình 3.1.1 T?ng quan mô hình nhìn t? tr??c



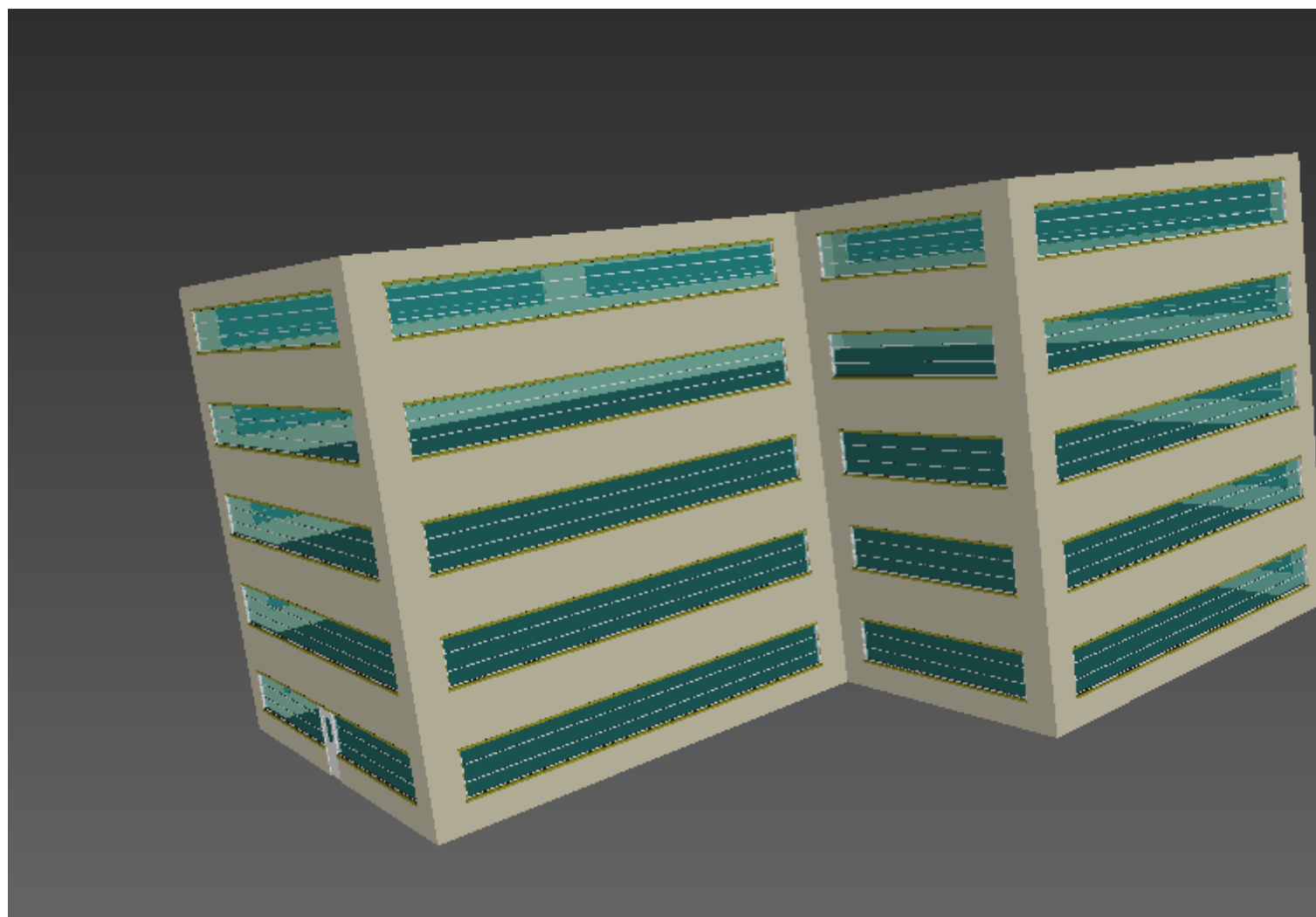
Hình 3.1.2 T?ng quan mô hình nhìn t? phía sau

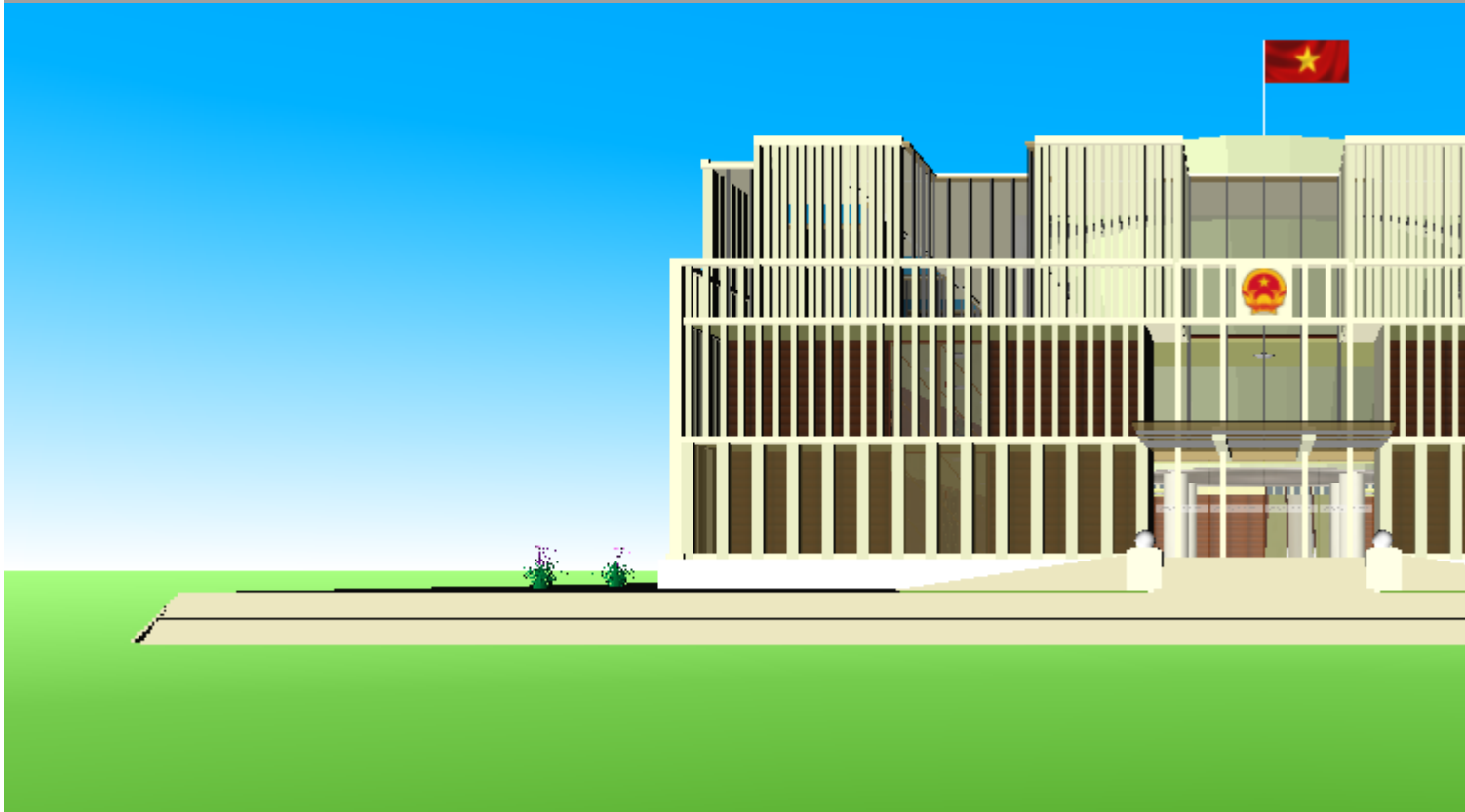


Hình 3.1.2 T?ng quan mô hình nhìn t? trên xu?ng

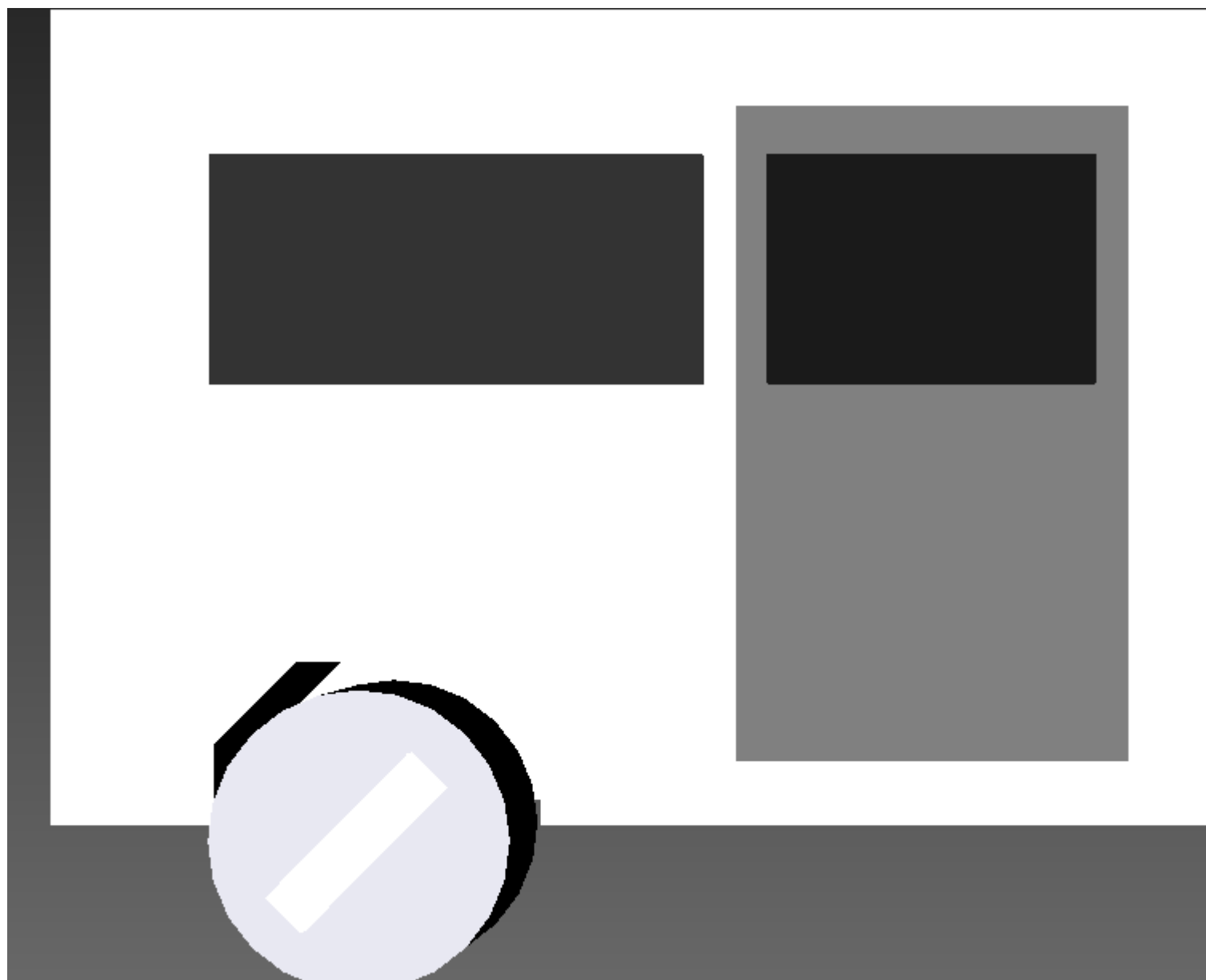
3.2. Các thành ph?n trong mô hình

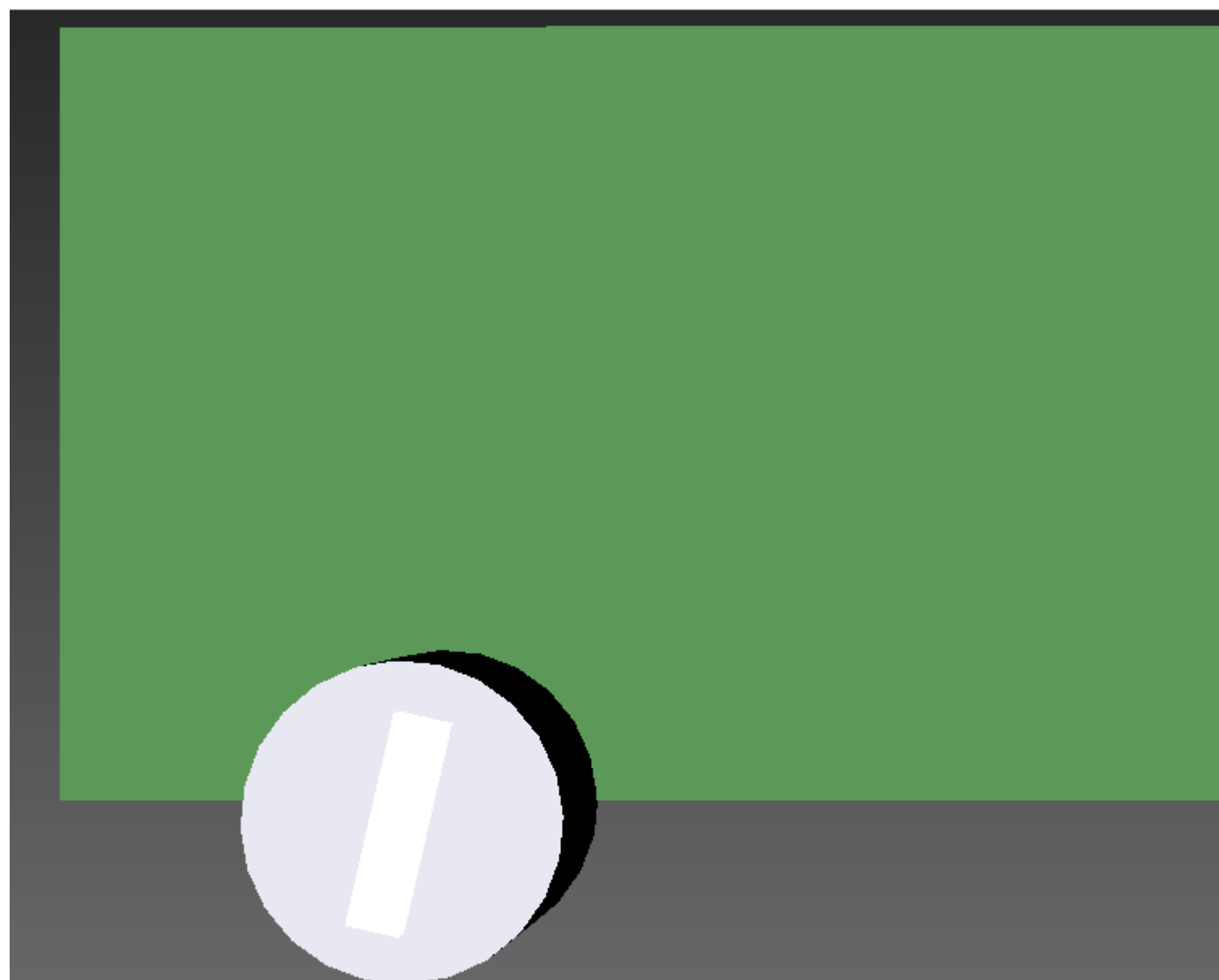
3.2.1. T?ng quan mô hình và ch?m

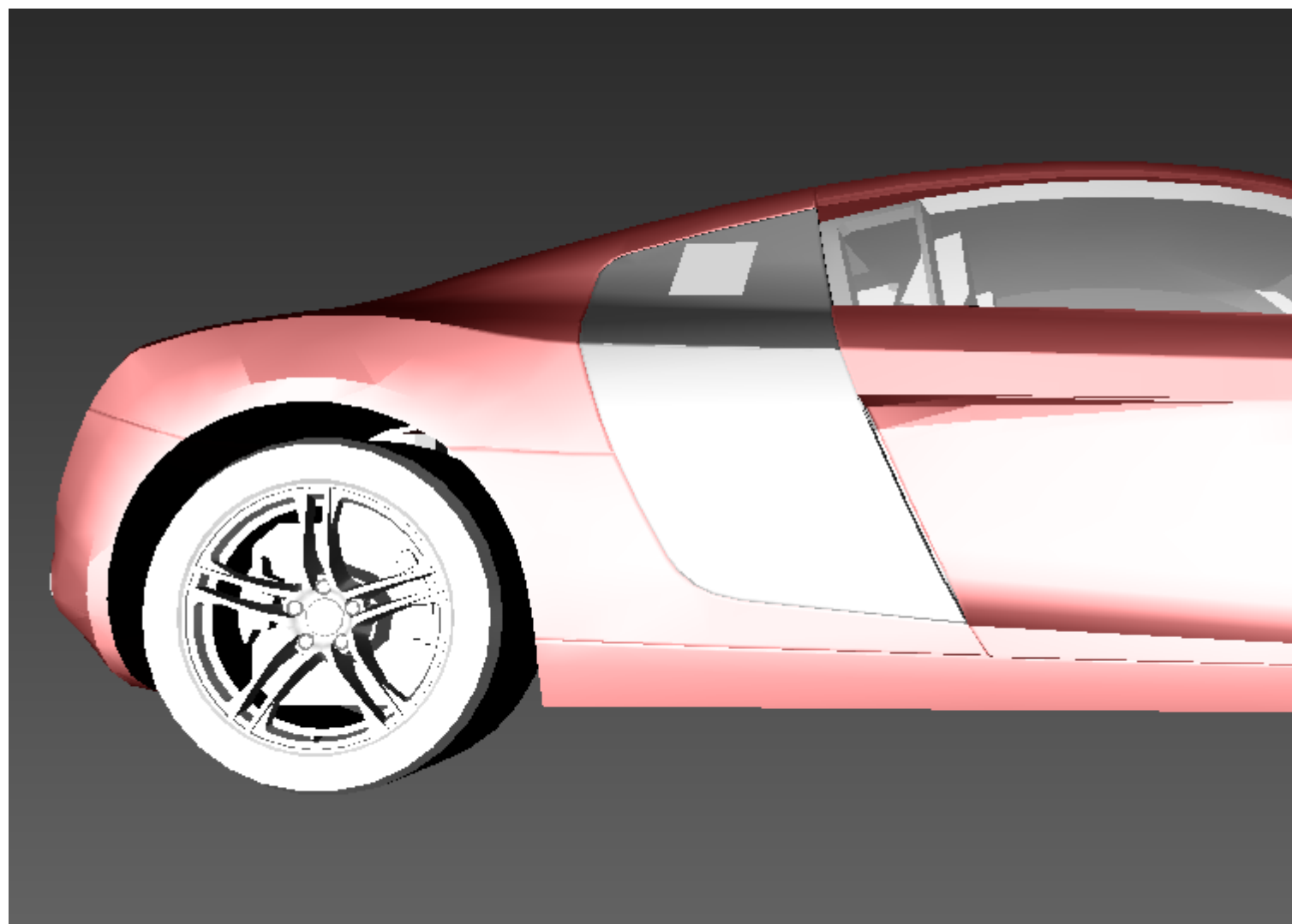


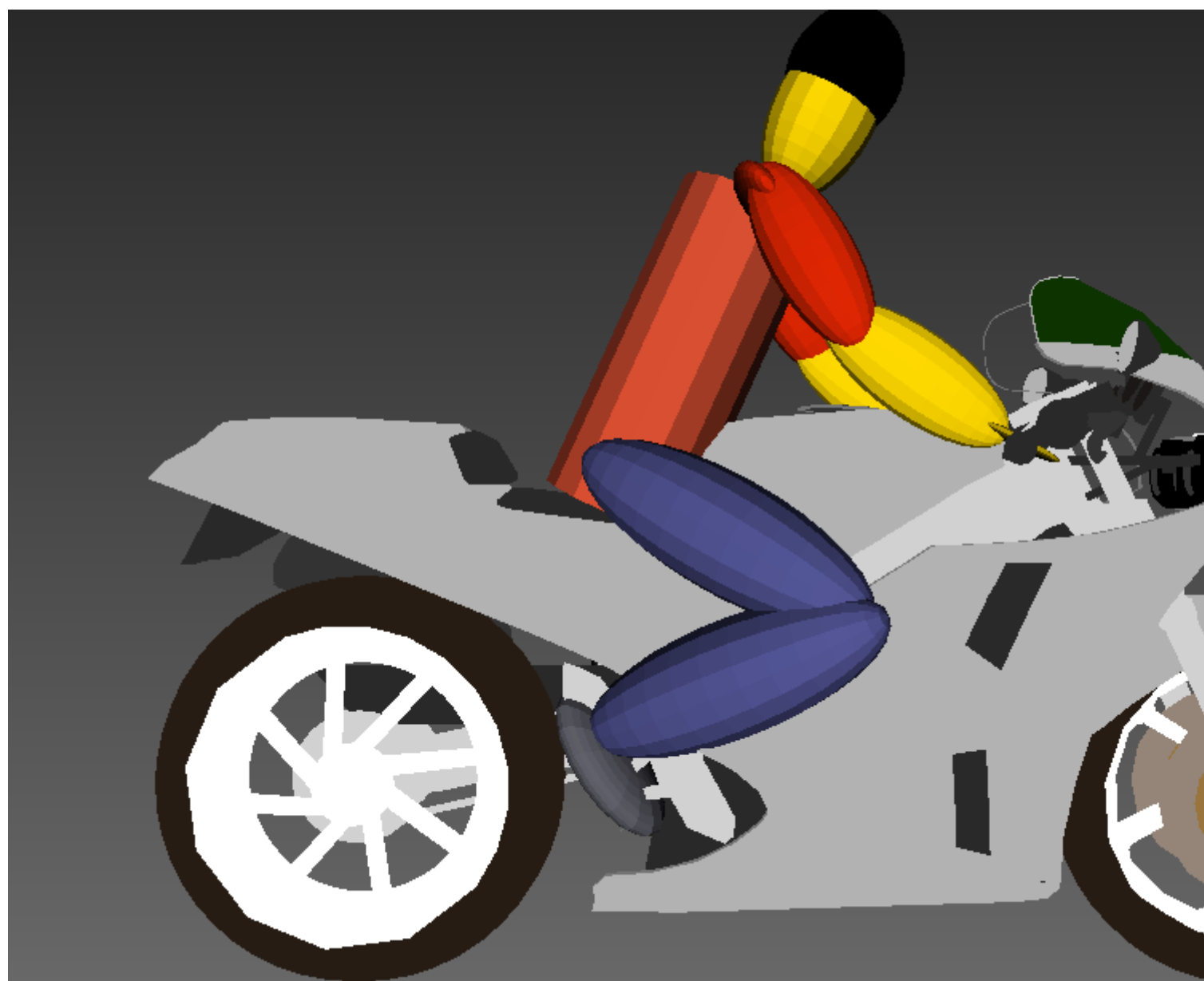




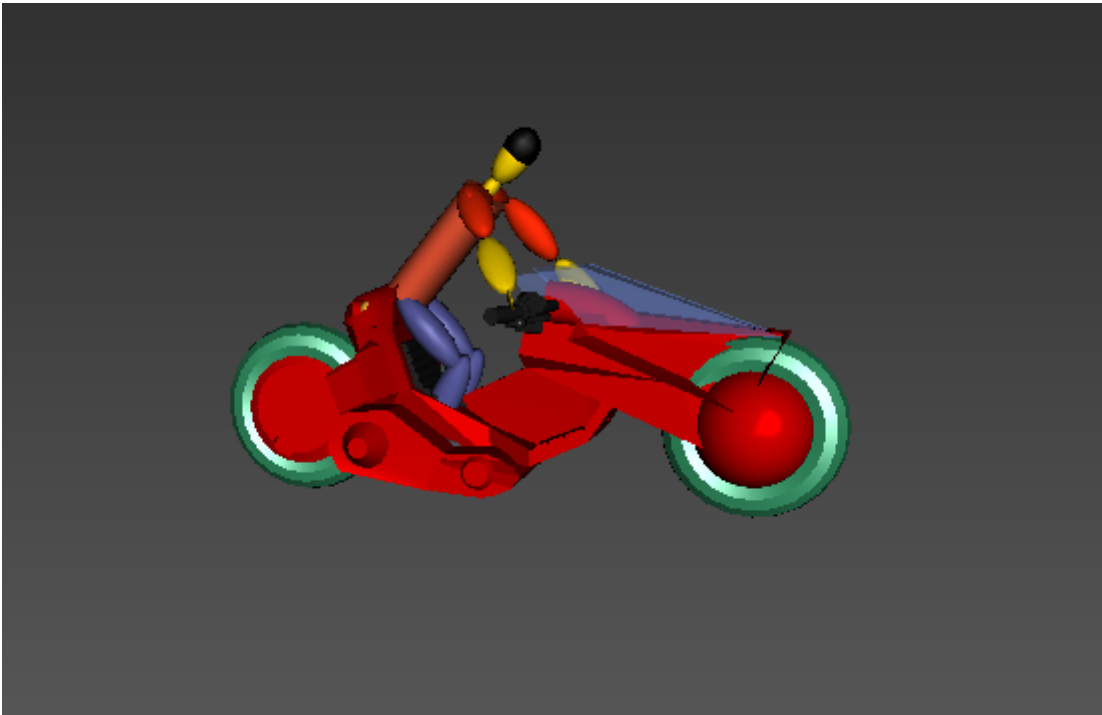












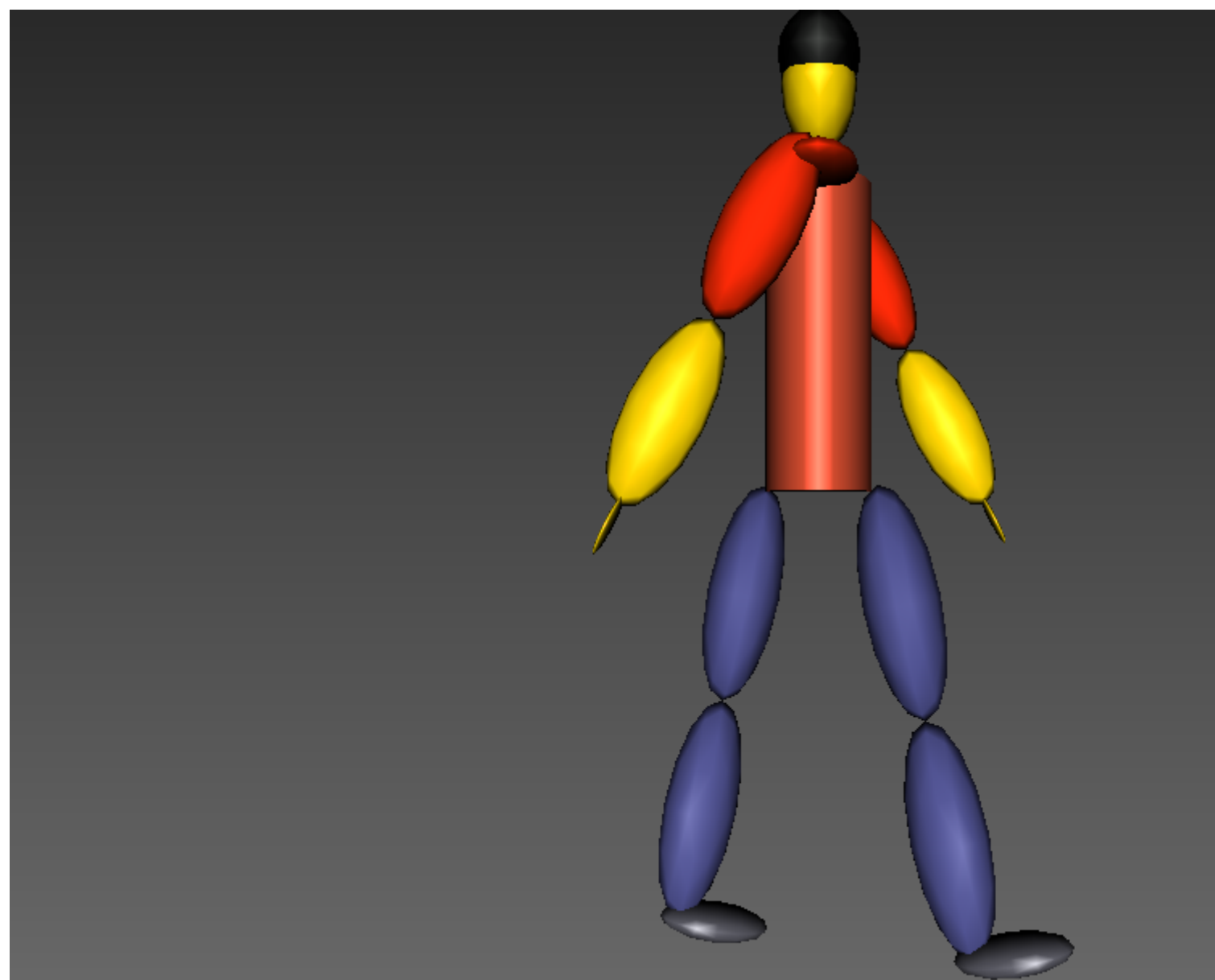
3.2.2. M?t s? v?t th? trong mô hình











TÀI LIỆU THAM KHẢO

1. Giáo trình ebook VRML, trường đại học công nghệ Hà Nội

[1] <http://123doc.org/document/100256-do-hoa-may-tinh-va-hien-thuc-ao.htm>

[2] <http://www.ebook.edu.vn/?page=1.39&view=21190>

[3] <http://khotailieu.com/luan-van-do-an-bao-cau/van-hoa-nghe-thuat/thiet-ke-do-hoa/do-hoa-may-tinh-va-hien-thuc-ao.html>

[4] <http://text.123doc.org/document/7500-ngon-ngu-mo-hinh-hoa-thuc-tai-vrml.htm>

[5] <http://hoidapit.com.vn/Questions/ViewQuestions/1048/chia-se-bo-tai-lieu-hoc-thuc-tai-ao-bang-ngon-ngu-vrml.html>

[6] <http://www.3dcadbrowser.com/>

[7] <http://www.lighthouse3d.com/vrml/tutorial/>

[8] <http://vrmlworks.crispen.org/models.html>