MỤC LỤC

LỜI CẢM ƠN

Em xin bày tỏ lòng biết ơn sâu sắc nhất tới thầy giáo Trần Ngọc Thái, thầy đã tận tình hướng dẫn và giúp đỡ em trong suốt quá trình làm tốt nghiệp. Với sự chỉ bảo của thầy, em đã có những định hướng tốt trong công việc triển khai và thực hiện các yêu cầu trong quá trình làm luận án tốt nghiệp

Em xin chân thành cảm ơn sự dạy bảo và giúp đỡ của các thầy giáo, cô giáo Khoa Công Nghệ Thông Tin – Trường Đại học Dân Lập Hải Phòng đã trang bị những kiến thức cơ bản để em có thể hoàn thành tốt báo cáo tốt nghiệp này.

Em xin cảm ơn GS.TS.NGƯT Trần Hữu Nghị Hiệu trưởng trường Đại học Dân lập Hải Phòng, Ban Giám hiệu nhà trường, Bộ môn tin học, các phòng ban của nhà trường đã tạo điều kiện tốt nhất cho em trong quá trình học tập và làm tốt nghiệp.

Em xin chân thành cảm ơn !

Hải Phòng, tháng 12 năm 2012

Sinh viên

Đinh Văn Sơn

LỜI MỞ ĐẦU

Thực tế ảo là một thuật ngữ mới xuất hiện khoảng đầu thập kỷ 90, nhưng ở Mỹ và châu Âu thực tế ảo (Virtual Reality) đã và đang trở thành một công nghệ mũi nhọn nhờ khả năng ứng dụng rộng rãi trong mọi lĩnh vực (nghiên cứu và công nghiệp, giáo dục và đào tạo, thương mại và giải trí,..) và tiềm năng kinh tế, cũng như tính lưỡng dụng (trong dân dụng 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 Văn Miếu Quốc Tử Giám ảo mà ta có thể đi lại quan sát trong đó. Chính vì tầm quan trọng cũng như khả năng ứng dụng to lớn đó nên việc nghiên cứu về thực tại ảo là vô cùng cần thiết. Và trên cơ sở đó có thể xây dựng một ứng dụng thực tại ảo hoàn chỉnh.

Chính vì vậy mà em đã chọn đề tài tốt nghiệp: ” Tìm hiểu công nghệ thực tế ảo và ứng dụng”. Đồ án này gồm có phần mở đầu, kết luận và nội dung:

Chương 1: Tổng quan về thực tại ảo

Chương 2: Tìm hiểu công cụ 3dmax

Chương 2: Ngôn ngữ VRML

Chương 3: Ứng dụng VRML trong việc xây dựng mô hình tủ kính trưng bày cổ vật

DANH MỤC VIẾT TẮT

VR	Thực tế ảo(Virtual Reality)
VRML	Virtual Reality Modeling Language
BOF	Birds of a Feather
HTML	Hyper Text Markup Language
HMD	Headsight.

CHƯƠNG 1. TỔNG QUAN VỀ THỰC TẠI ẢO.

1.1. Thực tế ảo là gì ?

Thực tế ảo-Virtual Reality(VR) là một hệ thống mô phỏng trong đó đồ họa máy tính được sử dụng để tạo ra một thế giới "như thật". Hơn nữa, thế giới "nhân tạo" này không tĩnh tại, mà lại phản ứng, thay đổi theo ý muốn (tín hiệu vào) của người sử dụng (nhờ hành động, lời nói,..). Điều này xác định một đặc tính chính của VR, đó là tương tác thời gian thực (real-time interactivity). Thời gian thực ở đây có nghĩa là máy tính có khả năng nhận biết được tín hiệu vào của người sử dụng và thay đổi ngay lập tức thế giới ảo. Người sử dụng nhìn thấy sự vật thay đổi trên màn hình ngay theo ý muốn của họ và bị thu hút bởi sự mô phỏng này.

Điều này chúng ta có thể nhận thấy ngay khi quan sát trẻ nhỏ chơi video game. Theo báo Bild (Đức), có hai trẻ nhỏ ở Anh bị thu hút và mải mê chơi Nintendo đến nỗi ngay cả khi nhà chúng đang bị cháy cũng không hề hay biết! Tương tác và khả năng thu hút của VR góp phần lớn vào cảm giác đắm chìm (immersion), cảm giác trở thành một phần của hành động trên màn hình mà người sử dụng đang trải nghiệm. Nhưng VR còn đẩy cảm giác này "thật" hơn nữa nhờ tác động lên tất cả các kênh cảm giác của con người. Trong thực tế, người dùng không những nhìn thấy đối tượng đồ họa 3D nổi (như hình nổi ở trang cuối báo Hoa học trò đã đăng trước kia), điều khiển (xoay, di chuyển,..) được đối tượng trên màn hình (như trong game), mà còn sờ và cảm thấy chúng như có thật. Ngoài khả năng nhìn (thị giác), nghe (thính giác), sờ (xúc giác), các nhà nghiên cứu cũng đã nghiên cứu để tạo các cảm giác khác như ngửi (khứu giác), nếm (vị giác). Tuy nhiên hiện nay trong VR các cảm giác này cũng ít được sử dụng đến.

Từ các phân tích trên, chúng ta có thể thấy định nghĩa sau đây của C. Burdea và P. Coiffet về VR là tương đối chính xác: VR- Thực Tế Ảo là một hệ thống giao diện cấp cao giữa Người sử dụng và Máy tính. Hệ thống này mô phỏng các sự vật và hiện tượng theo thời gian thực và tương tác với người sử dụng qua tổng hợp các kênh cảm giác. Đó là ngũ giác gồm: thị giác, thính giác, xúc giác, khứu giác, vị giác.

1.2. Lịch sử phát triển của công nghệ thực tế ảo

Khái niệm thực tế ảo đã có trong nhiều thập niên nhưng nó chỉ thực sự được nhận thức vào đầu những năm 90. Vào giữa những năm 50 Morton Heilig (Mỹ) đã phát minh ra thiết bị mô phỏng SENSORAMA. Đó là 1 thiết bị điều khiển 1 người sử dụng gồm có : một màn hình thực thể kính, quạt, máy tạo mùi, loa âm thanh và 1 chiếc ghế có thể di chuyển được. Ông cũng phát minh ra màn hình truyền hình được gắn vào đầu để có thể xem phim 3D. Tuy là những sản phẩm phục vụ cho điện ảnh nhưng những khái niệm của Heilig đã trở thành tiền đề cho VR sau này.

Những kỹ sư của Công ty Philco là những người đầu tiên phát triển HMD vào 1961, gọi là Headsight. Cái mũ sắt bao gồm một màn ảnh và hệ thống theo dõi video đã những kỹ sư liên kết tới một hệ thống camera mạch đóng. Họ dự định sử dụng HMD trong các tình huống nguy hiểm - một người có thể quan sát một môi trường thực sự từ xa, điều chỉnh góc quay camera bằng cách quay đầu. Bell Laboratories đã sử dụng HMD cho những phi công lai máy bay trực thăng. Họ liên kết HMD với những camera hồng ngoại gắn bên ngoài máy bay giúp phi công có thể nhìn rõ ngay cả trong môi trường thiếu ánh sáng.

Vào 1965, một nhà khoa học máy tính có tên Ivan Sutherland hình dung điều mà ông ta gọi là "Ultimate Display". Sử dụng hiển thị này, một người có thể thấy một thế giới ảo hiện ra như thế giới vật lý thật. Điều này đã định hướng toàn bộ tầm nhìn về VR. Khái niệm của Suntherland bao gồm :

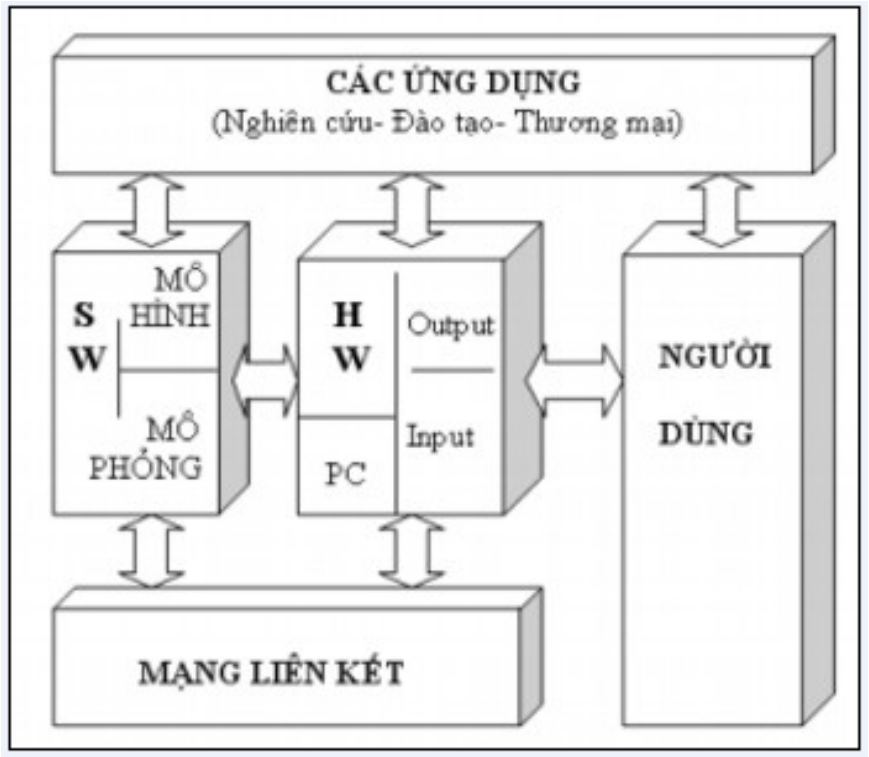
- Một thế giới ảo mà ta có thể quan sát thông qua một HMD
- Một máy tính để duy trì các mô hình trong thời gian thực
- Các khả năng cho người sử dụng để thao tác những đối tượng thực tế một cách trực quan nhất.

1.3. Các đặc tính chính của VR

Như trên đã trình bày, 2 đặc tính chính của VR là Tương tác và Đắm chìm, đây là hai "I" (Interactive, Immersion) mà nhiều người đã biết. Tuy nhiên VR cần có 1 đặc tính thứ 3 mà ít người để ý tới. VR không chỉ là một hệ thống tương tác Người - Máy tính, mà các ứng dụng của nó còn liên quan tới việc giải quyết các vấn đề thật trong kỹ thuật, y học, quân sự,...Các ứng dụng này do các nhà phát triển VR thiết kế, điều này phụ thuộc rất nhiều vào khả năng Tưởng tượng của con người, đó chính là đặc tính "I" (Imagination) thứ 3 của VR. Do đó có thể coi VR là tổng hợp của 3 yếu tố: Tương tác - Đắm chìm - Tưởng tượng, (3 I trong tiếng Anh: Interactive- Immersion- Imagination)

1.4. Các thành phần một hệ thống VR

Tổng quát một VR bao gồm những thành phần sau



Hình 1.1 Các thành phần của một hệ thông VR

1.4.1. Phần cứng (Hardware)

Phần cứng của một VR bao gồm:

- Máy tính (PC hay Workstation với cấu hình đồ họa mạnh).

- Các thiết bị đầu vào (Input devices): Bộ dò vị trí (position tracking) để xác định vị trí quan sát. Bộ giao diện định vị (Navigation interfaces) để di chuyển vị trí người sử dụng. Bộ giao diện cử chỉ (Gesture interfaces) như găng tay dữ liệu (data glove) để người sử dụng có thể điều khiển đối tượng.
- Các thiết bị đầu ra (Output devices): gồm hiển thị đồ họa (như màn hình, HDM,..) để nhìn được đối tượng 3D nổi. Thiết bị âm thanh (loa) để nghe được âm thanh vòm (như Hi-Fi, Surround,..). Bộ phản hồi cảm giác (Haptic feedback như găng tay,..) để tạo xúc giác khi sờ, nắm đối tượng. Bộ phản hồi xung lực (Force Feedback) để tạo lực tác động như khi đạp xe, đi đường xóc,...

1.4.2. Phần mềm (Software)

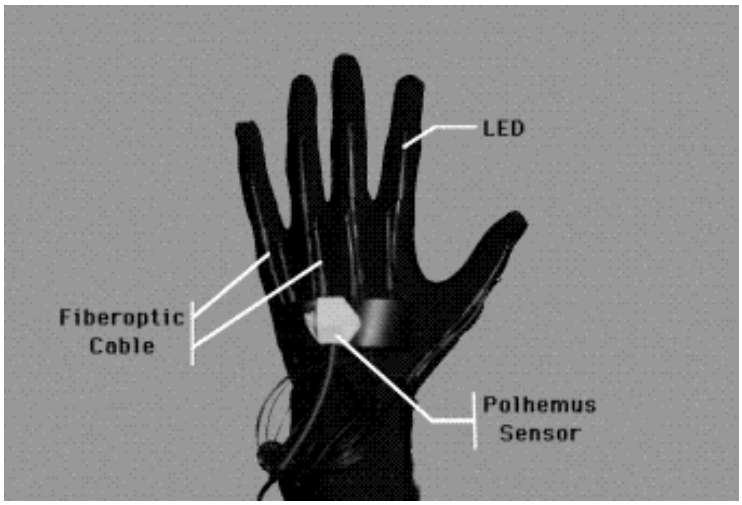
Phần mềm luôn là linh hồn của VR cũng như đối với bất cứ một hệ thống máy tính hiện đại nào. Về mặt nguyên tắc có thể dùng bất cứ ngôn ngữ lập trình hay phần mềm đồ họa nào để mô hình hóa (modelling) và mô phỏng (simulation) các đối tượng của VR. Ví dụ như các ngôn ngữ (có thể tìm miễn phí) OpenGL, C++, Java3D, VRML, X3D,... hay các phần mềm thương mại như WorldToolKit, PeopleShop,... Phần mềm của bất kỳ VR nào cũng phải bảo đảm 2 công dụng chính: Tạo hình vào Mô phỏng. Các đối tượng của VR được mô hình hóa nhờ chính phần mềm này hay chuyển sang từ các mô hình 3D (thiết kế nhờ các phần mềm CAD khác như AutoCAD, 3D Studio,..). Sau đó phần mềm VR phải có khả năng mô phỏng động học, động lực học, và mô phỏng ứng xử của đối tượng.

1.5. Các thiết bị cơ bản

1.5.1. Thiết bị định hướng và chuyển động

DataGloves

Thiết bị đo lường bàn tay phải cảm nhận được cả độ cong của các ngón tay và vị trí, sự định hướng của cổ tay trong thời gian thực. Thiết bị thương mại đầu tiên là DataGloves từ viện nghiên cứu VPL. DataGloves bao gồm 1 găng tay nylon nhẹ có các cảm biến quang học được gắn ở các ngón tay.

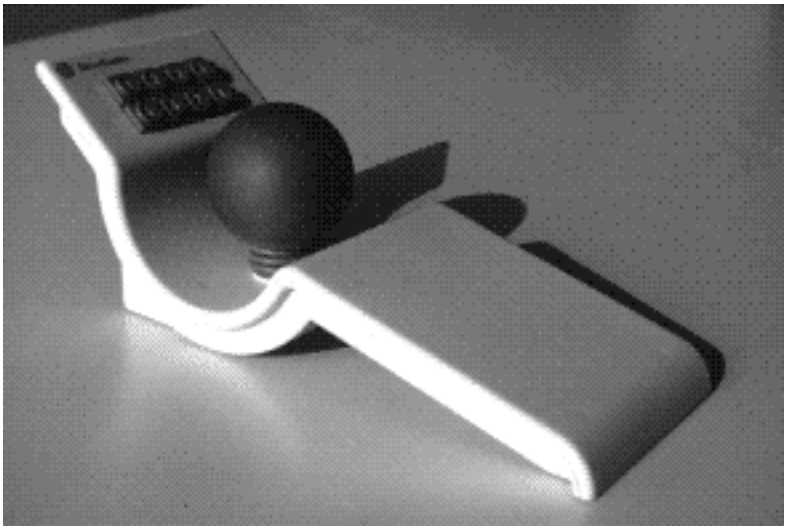


Hình 1.2 DataGloves

3D Mouse and SpaceBall

Hình 1.3 3D Mouse và SpaceBall

Chuột Logitech 3D dựa trên một mảng các vị trí siêu âm tham chiếu, đó là 1 cái kiềng gồm 3 loa siêu âm đặt ở 3 góc tam giác phát ra tín hiệu siêu thanh. Nó được sử dụng để theo dõi thiết bị thu, định hướng và chuyển động. Nó qui định thành phần của tỷ lệ gửi ra trong tất cả 6. mức tự do: X, Y, Z, Pitch, Yaw, và Roll.



Hình 1.4 Mouse

Shutter glasses

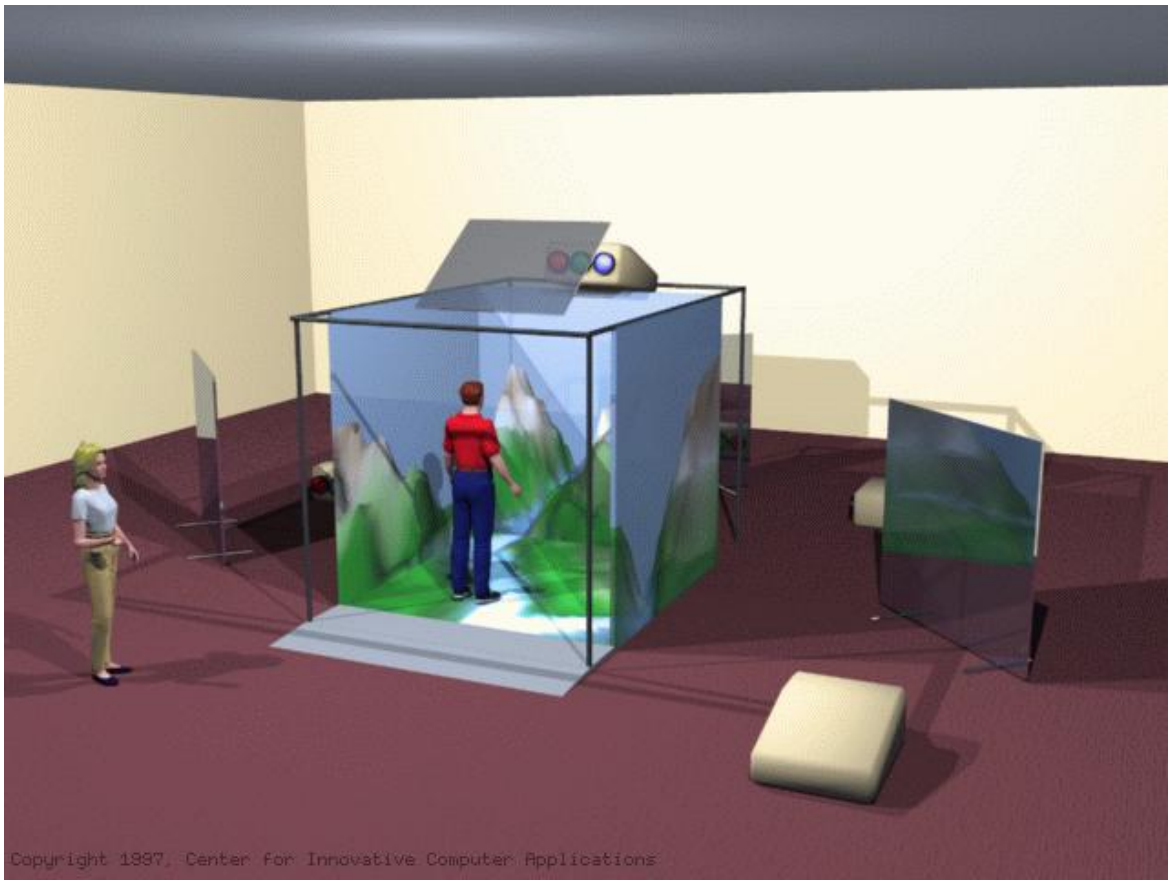
Hình 1.5 Shutter glasses

Head-Mounted Displays

Hình 1.6 Head-Mounted Displays

Cave

CAVE là 1 nhà hát có kích thước 10 X 10 X 9 được đặt bên trong 1 Phòng lớn hơn có kích thước 35 X 25 X 13. Phòng bên ngoài phải được chiếu sáng trong suốt quá trình sử dụng CAVE. Các bức tường của CAVE được tạo bởi các màn chiếu và sàn nhà cũng là một màn chiếu thẳng đứng. Máy chiếu độ phân giải cao hiển thị hình ảnh lên toàn bộ những màn ảnh khác bằng các tấm gương phản chiếu. Người dùng sẽ đi vào bên trong CAVE và đeo 1 chiếc kính đặc biệt để có thể nhìn thấy những hình ảnh 3 chiều mà CAVE hiển thị. Với những chiếc kính này người dùng có thể thấy các đối tượng thực sự nổi trong không khí và có thể đi lại xung quanh chúng. Điều này là hoàn toàn khả dĩ với các cảm biến điện tử. Khung của CAVE được làm từ i-nox không từ tính để có thể can thiệp một cách tốt nhất vào các cảm biến điện từ. Khi một người đi lại trong CAVE, chuyển động của họ được theo dõi bởi các cảm biến này và video sẽ điều chỉnh cho phù hợp. Máy tính sẽ kiểm soát việc này của CAVE cũng như cả khía cạnh âm thanh nữa. Có rất nhiều loa được đặt trong CAVE dưới nhiều góc độ giúp cho không chỉ có hình ảnh 3 chiều mà có cả âm thanh 3 chiều nữa.



Hình 1.7 Cave

1.5.2. Thiết bị tương tác và phản hồi

Các thiết bị này cảm nhận một số nhân tố sau của thiết bị khác gây ra: nhiệt độ, vận tốc di chuyển, sự chuyển động, áp lực và các ngoại lực khác.

CyberTouch

Hình 1.8 CyberTouch

CyberGrasp

Hình 1.9 CyberGrasp

1.6. Một số ứng dụng chính của VR

Tại các nước phát triển, chúng ta có thể nhận thấy VR được ứng dụng trong mọi lĩnh vực: Khoa học kỹ thuật, kiến trúc, quân sự, giải trí,... và đáp ứng mọi nhu cầu: Nghiên cứu- Giáo dục- Thương mại. Y học là lĩnh vực ứng dụng truyền thống của VR. Bên cạnh đó VR cũng được ứng dụng trong giáo dục, nghệ thuật, giải trí. Trong lĩnh vực quân sự, VR cũng được ứng dụng rất nhiều ở các nước phát triển. 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: VR ứng dụng trong sản xuất, VR ứng dụng trong ngành rôbốt, VR ứng dụng trong hiển thị thông tin (thăm dò dầu mỏ, hiển thị thông tin khối,....) VR có tiềm năng ứng dụng vô cùng lớn. Có thể nói tóm lại một điều: Mọi lĩnh vực "có thật " trong cuộc sống đều có thể ứng dụng "thực tế ảo" để nghiên cứu và phát triển hoàn thiện hơn.

1.6.1. Quân sự

Với việc phát triển của VR, các binh sĩ sẽ được huấn luyện 1 cách trực quan nhất các kĩ năng cân thiết như : lái máy bay, lái xe tăng, . .. trước khi tham gia công việc thực tế. Điều này vừa bảo đảm an toàn cho binh sĩ, vừa tiết kiệm được chi phí cho các khoa huấn luyện thực tế. Lầu Năm Góc vừa đưa ra quyết định sẽ đầu tư 36 triệu USD cho quân đội Mỹ để phát triển một game đặc biệt nhằm huấn luyện binh sĩ chống lại khủng bố dưới dạng chiến thuật thực tế ảo.Với hệ thống trò chơi đặc

biệt này, những binh sĩ có thể tập luyện những bài tập của mình ngay tại nhà nhằm chống lại những tình huống có thể phát sinh ra trong thực tế. Đây sẽ là một game rất sống động, có tình 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 1.10 Binh lính học nhảy dù bằng thực tế ảo

1.6.2. Giáo dục

Ở các nước phương Tây việc ở nhà học qua Internet không còn là điều mới mẻ nữa. Và công nghệ VR sẽ làm cho việc này trở nên thú vị hơn rất nhiều. Giống như một game MMORPG bạn điều khiển 1 nhân vật đại diện cho bạn đi lại trong 1 trường học ảo được xây dựng trên máy tính. Bạn có thể tham gia vào bất cứ lớp học ảo nào mà bạn thích, nói chuyện với nhưng thành viên khác trong lớp.

Hình 1.11 Cảnh trong một lớp học ảo

1.6.3. Xây dựng

Bạn muốn xây nhà. Bạn thuê một kiến trúc sư thiết kế cho ngôi nhà tương lai của bạn. Anh ta hoàn thành nó trên bản vẽ và liệu bạn có thể tưởng tượng ra nó thế nào không ?Có thể nhưng chắc là không thể chính xác được. Và khi hoàn thành thì chưa chắc nó đã đúng ý của bạn. Giờ đây ngôi nhà đó được xây dựng trên máy tinh, bạn có thể đi lại khắp nơi trong nhà, xem xét từng ngõ ngách nhỏ nhất.

1.6.4. Y học

Thực tại ảo giải quyết được 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 tránh gây rủi ro trong thực tế

Như vậy thực tại ảo có ứng dụng trong hầu hết các lĩnh vực của cuộc sống. Qua đó cũng nhận thấy được ý nghĩa to lớn của việc ứng dụng thực tại ảo, bởi những vấn đề khó khăn mà nếu không có thực tại ảo thì rất khó giải quyết hoặc hiệu quả không cao mà chi phí tốn kém.

CHƯƠNG 2. TÌM HIỂU VỀ CÔNG CỤ 3DMAX.

2.1. Tổng quan về 3dmax.

2.1.1. Giới thiệu.

3dmax là một trong những chương trình giúp bạn tạo ra và diễn hoạt các vật thể 3 chiều, cho phép thiết đặt khung cảnh mà trong đó ánh sáng, bóng đổ, sự phản chiếu, hiệu ứng mưa, sương mù, lửa,…được thiết đặt khi cần thiết..và cuối cùng cho phép xuất ra dưới các định dạng như phim, ảnh, các mô hình thực tế ảo…phục vụ cho việc tạo các pim hoạt hình, quảng cáo, thiết kế các nhân vật trong các trò chơi(game). Trong xây dựng và thiết kế phối cảnh nội ngoại thất. Phục vụ rất đắc lực trong việc thiết kế các mô hình hỗ trợ cho việc giảng dạy như:

Ví dụ: trong cơ khí: thiết kế và diễn hoạt sư hoạt động của một động cơ đốt trong. Một hệ thống lạnh. Người máy (robot)…

Trong giải phẫu học…ví dụ: sự chuyển động của máu trong cơ thể qua các mạch máu trở về tim…

2.1.2. Một số khái niệm cơ bản.

2.1.2.1. Ba chiều.

Dựng mô hình 3 chiều đồng nghĩa với sử dụng trục x, y, z. Ba trục này chi phối hình dạng, tỉ lệ, vị trí của đối tượng được tạo dựng nên.

Khi quan sát 1 hình chiếu đỉnh: nhìn vật thể từ trên xuống, trục x chạy từ trái qua phải ngang qua màn hình, trục y chạy từ trên xuống dưới dọc theo màn hình, trục z chạy từ xa đến gần người dùng.

2.1.2.2. Dựng mô hình.

Mặt phẳng dựng hình: mọi hoạt động tạo đối tượng 3 chiều đều diễn ra trên mặt phẳng lưới 2 chiều, có thể xem mỗi hình chiếu là 1 mặt phẳng, nguyên tắc kiến tạo và thao tác trong chương trình là phải nhận diện được mặt phẳng lưới mà bạn đang làn việc thao tác trên đó.

Cấu trúc mô hình: Đối tượng phức tạp trong 3dmax cấu thành từ nhiều hình thái cấu trúc, một vị trí trong không gian 2 chiều hoặc 3 chiều là hình thái đơn giản nhất của cấu trúc mô hình. Mặt – face là mặt phẳng bề mặt, xác định bởi 3 đỉnh, dùng để phủ lên đối tượng. Phức tạp hơn nữa là phần tử element, gồm nhiều mặt

liên kết với nhau hình thành nên 1 phần của đối tượng. nếu 2 mặt có chung cạnh và nằm trên cùng mặt phẳng, cạnh chung thường không hiển thị, còn bề mặt trông như có hình chữ nhật. Cuối cùng, đối tượng – object là tập hợp gồm nhiều phần tử tạo nên hình dạng phúc tạp.

Hình dạng: hình dạng – shape là đa giác đường viền 2 chiều khép kín làm cơ sở để tạo đối tượng 3D phức tạp hơn. Khi hở, chúng đóng vai trò là đường dẫn – path giúp tạo đối tượng 3D và hoạt cảnh.

Đối tượng ghép: Những đối tượng tạo trong 3Dmax được phủ bằng nhiều mặt gọi là đối tượng ghép. Một khi đã tạo đối tượng, có thể thao tác bất kỳ phần nào của đối tượng từ đỉnh mặt cho đến phần tử.

2.1.2.3. Hiển thị mô hình.

Khung nhìn: viewport dung để hiển thị 1 hay nhiều hình chiếu – view của đối tượng hoặc khung cảnh – scene trên màn hình. Ta có thể tạo ra khung nhìn riêng gọi là khung nhìn User hoặc bố trí camera để tạo khung nhìn camera.

Khung dây: khung dây hiển thị những cạnh bị che khuất: khung dây – wireframe là chế độ hiển thị đường nét, hiển thị tất cả các cạnh của đối tượng cho thấy chúng tác động ra sao đến những đối tượng xung quanh. Khung dây không hiển thị những cạnh bị che khuất: chế độ wireframe với đặc tính Backface Cull hoạt động chỉ hiển thị những cạnh không bị che khuất của đối tượng trông có vẻ 3 chiều hơn, giúp người dùng nhận diện từng đối tượng dễ dàng hơn.

Hộp giới hạn: Bounding box còn gọi là khung bao là khung bao quanh từng đối tượng phức tạp. Mục đích là nhằm tăng tốc độ hiển thị khung cảnh phức tạp.

2.1.2.4. Diễn họa.

Diễn họa – render là kỹ thuật áp màu, chất liệu, ánh sáng tối cho bề mặt hoặc mặt của mô hình 3 chiều. Có thể hiển thị ảnh diễn họa kết quả trên màn hình hoặc lưu vào đĩa với đủ định dạng tập tin.

2.1.2.1. Hoạt hình.

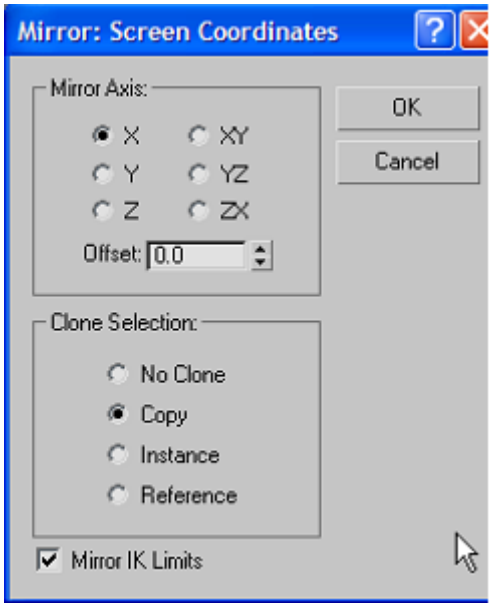
Khi trình chiếu 1 loạt ảnh liên tiếp nhau, mỗi ảnh có cảnh chỉ hơi khác nhau 1 chút thì sẽ cho cảm giác hình ảnh đang chuyển động. Đó là kỹ thuật hoạt hình.

2.2. Các phép biến đổi.

2.2.1. Các phép biến đổi mirror, align, array.

2.2.1.1. Lệnh Mirror.

Dùng lật đối xứng đối tượng, nếu sao chép (copy) thì sẽ tạo ra 1 đối tượng khác đối xứng với đối tượng hiện hành, tại trục mà nó được chọn. Hệ trục được sử dụng là hệ trục tọa độ thế giới.

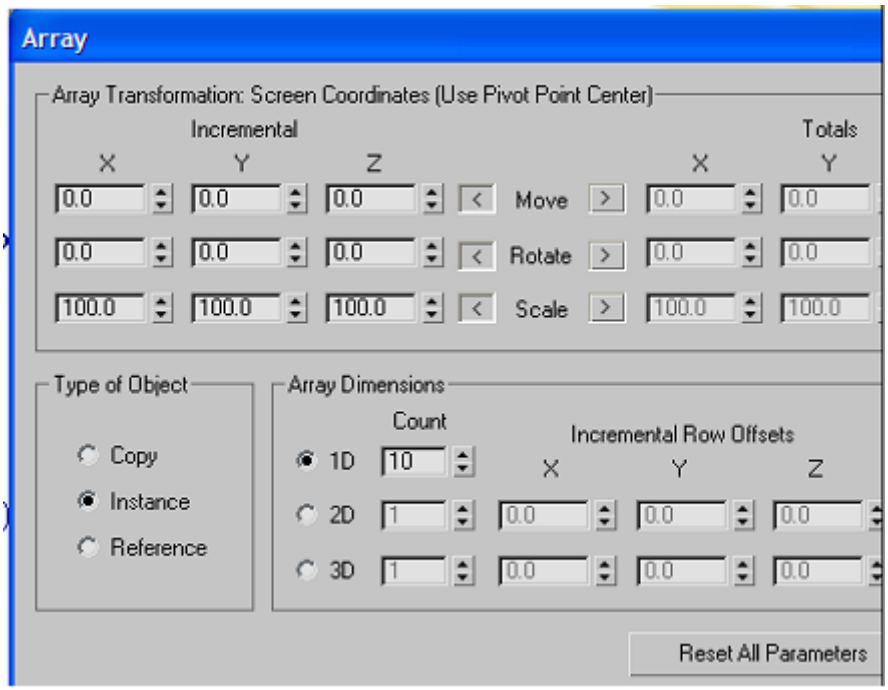


Hình 2.1: bảng lệnh Mirror.

2.2.1.2. Lệnh Array.

Cho phép tạo nhiều bản sao của đối tượng theo kiểu tròn hay ô lưới. Bạn có thể tạo dãy theo 1 chiều, 2 chiều, hay 3 chiều kích thước. Và cũng có thể xác lập các đối tượng tạo ra theo dãy đó là sao chép (copy), Instance hoặc References.

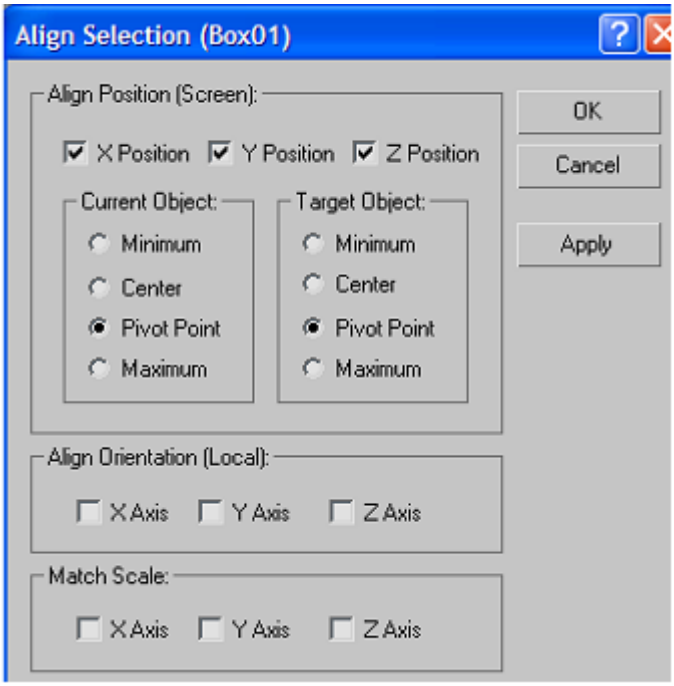
Cho phép điều chỉnh khoảng cách các thành phần trong dãy và được tính chính xác chi tiết khoảng cách của từng đối tượng (Incremental) hoặc tính theo tổng khoảng cách (Total) bằng cách chọn vào mũi tên nằm giữa incremental và total bạn có thể chuyển đổi giữa 2 cách tính.



Hình 2.2: bảng lệnh Array.

2.2.1.3. Lệnh Align.

Lệnh align dùng để gióng 1 đối tượng này theo đối tượng khác, có thể là gióng theo vị trí, góc xoay hoặc kích thước.



Hình 2.3: bảng lệnh Align.

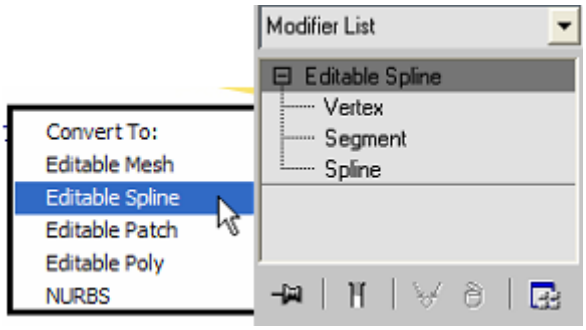
2.2.1. Các phép biến đổi hình dạng.

Các phép biến đổi hình dạng (shape Modifier) dùng thay đổi các hình dạng đường viền(spline shape). Có 3 phép biến đổi cơ bản: Edit spline, Extrude, lathe. Trong đó Extrude và lathe biến dạng hình 2D thành 3D, Edit mesh giúp tạo hiệu ứng đặc biệt trên đường viền.

2.2.1.1. Edit spline.

Dùng để chọn thay đổi toàn bộ đường viền, đoạn đỉnh trong hình dạng. Sau khi đã chọn hình dạng cần biến đổi, cần ấn định cấp đối tượng con (subobject) như:

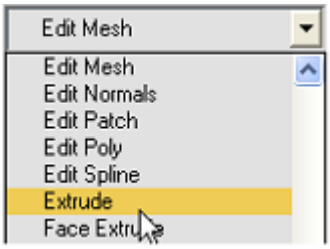
- Đường viền: Spline.
- Đoạn: Segment.
- Đỉnh: Vertex.



Hình 2.4: bảng Edit spline.

2.2.1.2. Extrude.

Phép này biến hình được chọn thành 3 chiều dựa vào giá trị độ dày Amount. Có 2 phương pháp kết xuất: patch và Mesh. Chọn Mesh cho bề mặt chuẩn, Patch nếu muốn hiệu chỉnh bề mặt.



Hình 2.5: bảng Extrude.

2.2.1.3. Lathe.

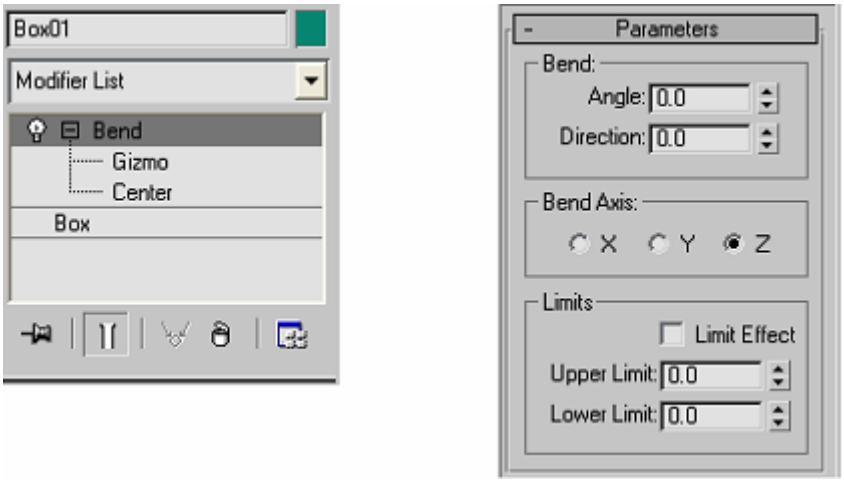
Tạo đối tượng 3 chiều từ hình dạng được chọn bằng cách quay đối tượng xung quanh trục xác định. Điều chỉnh trục quay dựa vào nhóm tùy chọn Align hoặc tự điều chỉnh trục quay bằng 1 lệnh biến ảnh như Move.

Hình 2.6: bảng lathe.

2.2.2. Các phép biến đổi hình học.

2.2.2.1. Uốn cong (Bend).

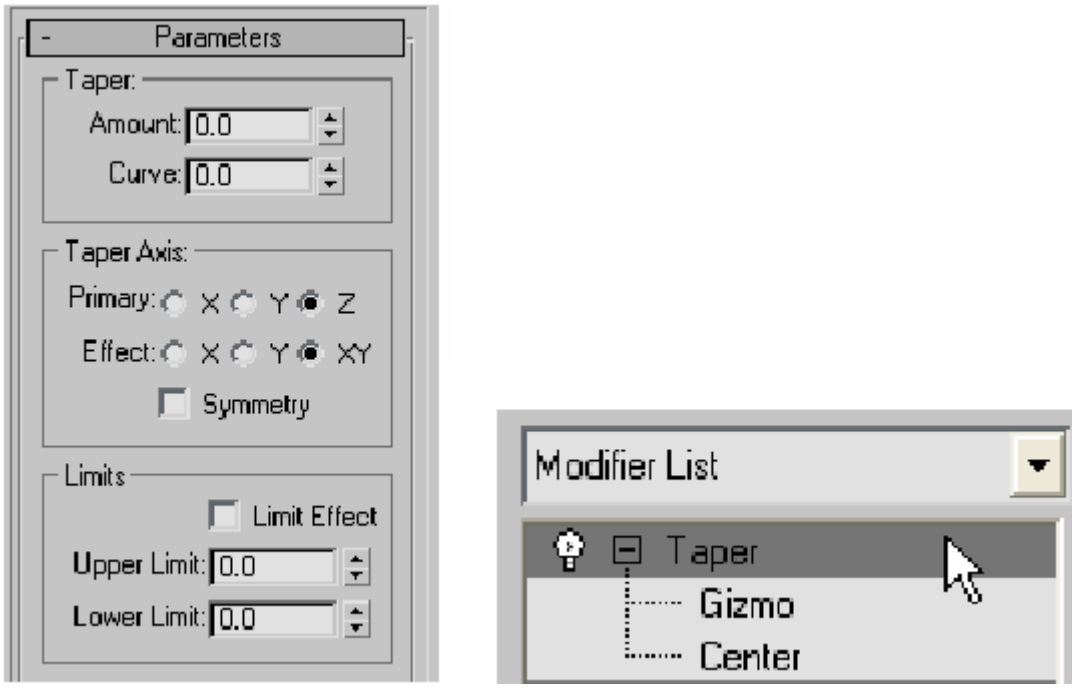
Tạo hiệu ứng uốn cong đồng dạng, có thể điều chỉnh góc uốn cong, hướng uốn cong, trục uốn cong và các giới hạn. giới hạn gồm có khoảng cách trên và dưới tâm điểm của vùng uốn cong.



Hình 2.7: bảng Bend.

2.2.2.2. Khuôn hình búp măng (Taper).

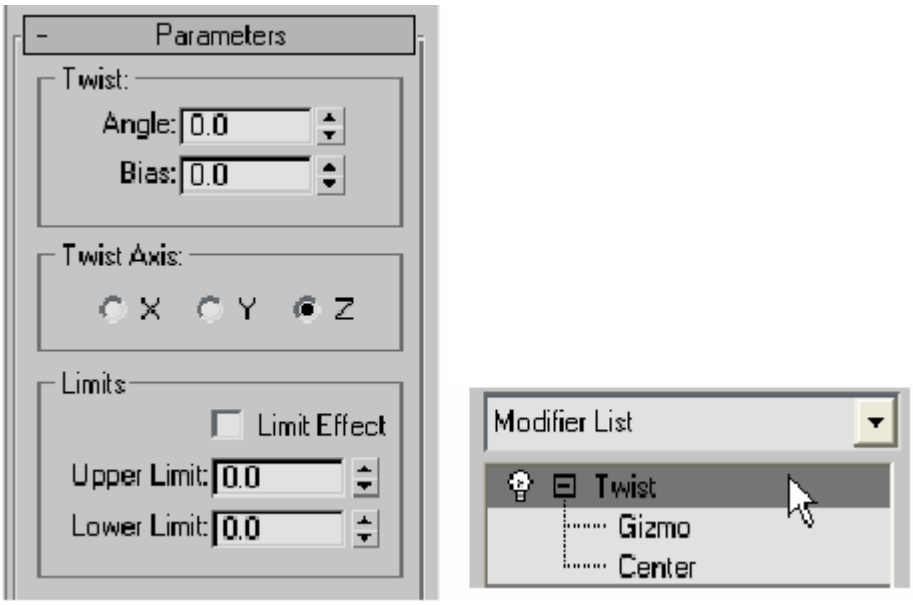
Tạo hiệu ứng khuôn hình búp măng bằng cách thu hẹp 1 đầu của đối tượng hình học, sau đó định mức độ thu hẹp và áp dụng đường cong.



Hình 2.8: bảng Taper.

2.2.2.3. Vặn xoắn (Twist).

Áp dụng hiệu ứng vặn xoắn trông như cái mở nút chai cho đối tượng. Đối tượng bị xoắn xung quanh trục đã chọn và theo góc xoắn xác định.



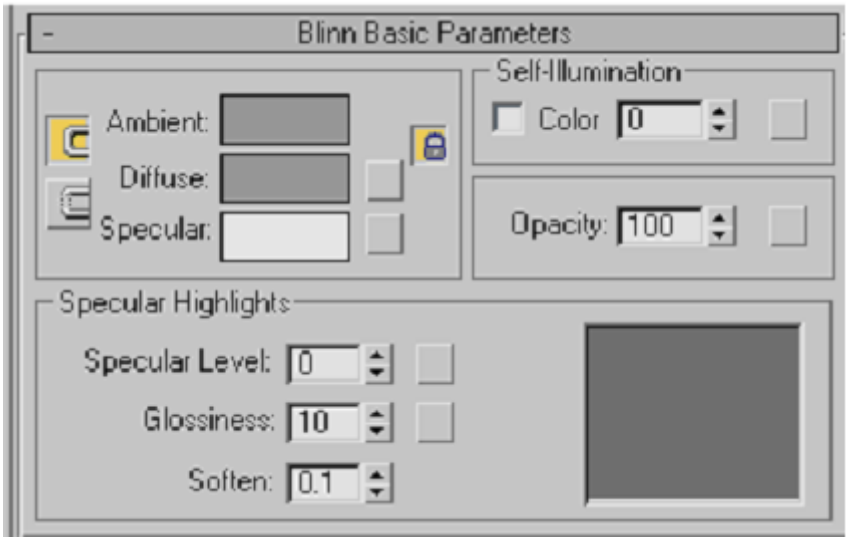
Hình 2.9: bảng twist.

2.3. Gán vật liệu – vật liệu.

2.3.1. Cửa sổ biên tập chất liệu.

Trên vật thể có các vùng chính sau:

- Ambient: vùng tối.
- Diffuse: vùng khuyết tán.
- Specular: vùng sáng chói.



Hình 2.10: bảng cửa sổ chất liệu.

2.3.2. Các kiểu vật liệu đơn giản.

2.3.2.1. Vật liệu có màu hoen ố hay sáng chói.

- Diffuse: màu vật liệu.
- Specular: màu hoen ố.
- Khai báo độ hoen ố rộng hẹp thông qua Specular Highlight.

2.3.2.2. Vật liệu màu chuyển tiếp.

- Diffuse = gradient.
- Color 1: màu đậm.
- Color 2: màu vừa.
- Color 3: màu nhạt.
- Color 2 Position = 0.5 (tỉ lệ màu chuyển tiếp).
- Gradient Type – kiểu chuyển tiếp:

+ Linear: thẳng.

+ Radial: tròn.

- Noise: tạo những thay đổi ngẫu nhiên được gán cho vật liệu như hạt, màu:

+ Amount: 0.5

+ Size: 0.1

2.3.2.3. Vật liệu có những gai lồi lõm.

- Maps
- Bump = noise
- Tại bảng Noise parameters ta thiết lập các thông số cho: Regular, practal, Turbulence.

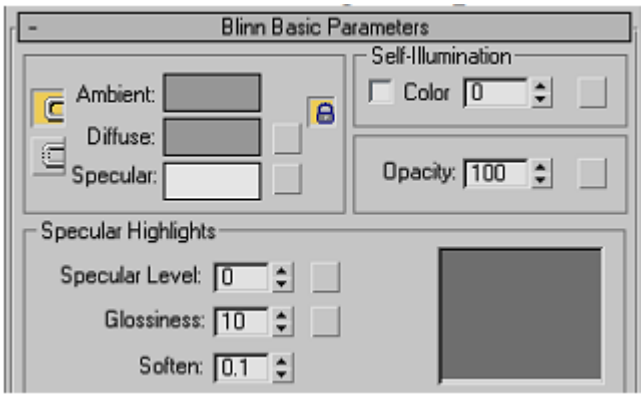
2.3.3. Thao tác với các ảnh bitmap.

Có 2 loại ảnh bitmap và metarial editor có thể xử lý đó là ảnh Bitmap và ảnh thủ tục.

2.3.3.1. Ảnh Bitmap.

- Được lưu với nhiều định dạng như: Jpeg, gif, png… và định dạng avi định dạng phim hoạt hình.
- Ảnh bitmap có thể áp dụng cho nhiều thuộc tính của chất liệu. Tất cả các màu của ảnh bitmap đều được sử dụng nhưng 1 vài trường hợp khác thì chỉ có 2 màu đen hoặc trắng hoặc giá trị bão hòa của ảnh bitmap mới được sử dụng.
- Mỗi thuộc tính khi áp ảnh bitmap có thể liệt kê mô tả như sau:

  + Ambient: rất ít khi được sử dụng nhưng khi được sử dụng ở ảnh bitmap thì ảnh đó sẽ xuất hiện khi đối tượng đó nằm trong bóng tối. Thường thì 2 kênh Embient và Deffuse khóa lại với nhau vì vậy ảnh bitmap vào Deffuse cũng là áp cho Ambient.



Hình 2.11: bảng thuộc tính ảnh bitmap

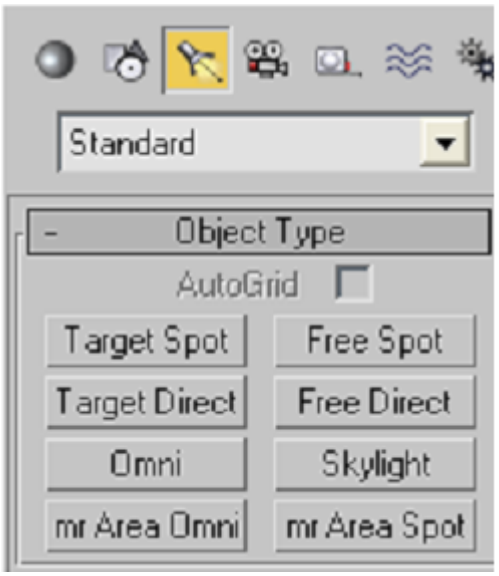
2.3.3.2. Thao tác.

- Chọn vào thanh cuộn bitmaps để mở thanh cuộn này. Chọn vào nút màu xám dài ngay thuộc tính Deffuse lấy vào 1 ảnh bitmap có sẵn trong thư viện, sau đó double chọn vào ô chất liệu ta sẽ có 1 cửa sổ với ô chất liệu.
- Ổ khóa bên trong phải đang khóa deffuse với emblient, chứng tỏ ảnh lấy vào được áp lên cho cả 2 thuộc tính này. Chọn vào khóa để bỏ sự ràng buộc và quan sát sự thay đổi trên ô đầu mẫu. Mọi thứ tối đi khi bỏ sự ràng buộc này.
- Chọn vào ô khóa lần nữa để trả lại như mặc định. Chọn và giữ chuột từ thanh bitmap của Deffuse thả xuống thanh none/map của Specular color, chọn lệnh Swap để hoán đổi 2 thanh này cho nhau, quan sát sự thay đổi.

2.4. Ánh sáng light và camera.

2.4.1. Ánh sáng light.

Có 3 loại nguồn sáng được sử dụng trong chương trình: điểm (Ommi),chùm (Target spot & Free spot0, xa (Targer Direct & free Direct).



Hình 2.12: bảng light.

2.4.1.1. Nguồn sáng điểm (Ommi).

- Là dạng nguồn sáng điểm, lan tỏa theo mọi hướng. Nguồn sáng Ommi được sử dụng tốt trong việc mô phỏng các loại ánh sáng không định hướng như: ánh sáng mặt trời, của bóng đèn…

2.4.1.2. Nguồn sáng chùm (Target spot và free spot).

- Là dạng nguồn sáng theo chùm, chiếu sáng từ 1 điểm theo hình nón, giống như nguồn sáng từ đền pin soi trong đêm tối hoặc nguồn chùm sáng trên sân khấu chiếu sáng từ 1 điểm và rọi sáng tập trung vào diễn viên, ca sĩ..
- Đây là dạng nguồn sáng thường rất được sử dụng để tạo các hiệu ứng ánh sáng trong chương trình

2.4.1.3. Nguồn sáng xa (Target Direct và free Direct).

- Là nguồn sáng xa, chỉ chiếu sáng dọc theo 1 trục và các tia sáng này thì song song với nhau. Có thể dùng dạng này để tạo ra hiệu ứng của ánh sáng tia laser..

2.4.2. Camera.

- Bởi vì camera tượng trưng cho con mắt của khán giả, nên vấn đề cực kỳ quan trọng ở đây là phải liên hệ những gì nhìn qua camera với khán giả sẽ cảm nhận nếu khán giả ở trong cảnh thực.
- Con người vốn có khả năng liên cảm mạnh mẽ về tốc độ, do đó người ta sẽ cắt nghĩa hoạt cảnh của bạn bằng cách so sánh với kinh nghiệm riêng của họ.

Hầu hết khán giả đều biết rõ theo kinh nghiệm như thế nào là bước đều, thế nào là chạy và ra sao là cuộc đua. và vì vậy, họ cũng rõ là tốc độ ống kính có thực hay không.

- Bởi vì cắt giảm tốc độ diễn cảnh có nghĩa là tăng thêm số khung nhìn và tăng thời gian diễn họa, nên hoạt cảnh máy tính thường có xu hướng bay và chạy xuyên qua cảnh.

2.5. Kết xuất tập tin.

2.5.1. Xuất tập tin ra mô hình.

- Xuất mô hình thành đoạn phim:
- Nhấn phím F10 để mở hộp thoại render.
- Chọn thẻ Common.
- Trong vùng xuất output, chọn mục Active Time Segment để chọn hết tất cả các frame hình.
- Mục output size: thiết lập chế độ kích thước khung hình xuất ra cho đoạn phim như 800 x 600 hoặc 320 x 240.
- Mục render output: định dạng tập tin cần xuất → chọn nút file chọn đường dẫn đặt tên và kiểu định dạng cho tập tin.
- Sau đố chọn vào nút render.

Lưu ý: trước khi xuất ra cần phải chuyển qua khung nhìn camera.

2.5.2. Xem trước khi xuất tập tin.

- Nhấn phím F9 (render last), lệnh render last lặp lại tại phần hoàn trả cuối cùng(dù là 1 render view, render region, render blowup, hoặc render selected) sử dụng viewport cuối cùng từ cái mà bạn đã lọc.

CHƯƠNG 3. NGÔN NGỮ VRML

3.1. Giới thiệu về VRML

3.1.1. VRML là gì ?

VRML (Virtual Reality Modeling Language) là ngôn ngữ mô hình hóa thực tế ảo, một định dạng tập tin được sử dụng trong việc mô tả các thế giới và các đối tượng đồ họa tương tác ba chiều. VRML được thiết kế dùng trong môi trường Internet, Intranet và các hệ thống máy khách cục bộ (local client). VRML còn được dự trù trở thành một chuẩn trao đổi đa năng cho đồ họa ba chiều tích hợp và truyền thông đa phương tiện. VRML có thể 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.

Ra đời phiên bản đầu tiên vào tháng 10 năm 1994 (VRML 1.0), cho đến nay VRML đã phát triển tới phiên bản 2.0 với các chức năng mạnh mẽ, nhanh chóng trở thành chuẩn phát triển cho nhiều chương trình đồ họa. VRML là ngôn ngữ Interner 3D dùng để phát triển đồ họa 3D trên Web, có cấu trúc chặt chẽ, với khả năng mạnh mẽ, giúp cho việc xây dựng các ứng dụng 3D một cách nhanh chóng và chân thực nhất.

3.1.2. Định nghĩa về VRML

Ngôn ngữ VRML là ngôn ngữ sử dụng mô hình phân cấp trong việc thể hiện các tương tác với các đối tượng của mô hình, VRML được sử dụng để phát triển những hình ảnh 3D và quang cảnh trên Web. Các file VRML có kích thức nhỏ, thường không quá 1Mb.

Ngôn ngữ mô hình hóa thực tại ảo VRML là một chuẩn không chính thức để mô tả thực tế ảo mà không phụ thuộc vào hệ điều hành thông qua Internet. Chỉ với một file text bạn có thể mô tả, tương tác, điều khiển một thế giới ảo mà không bị hạn chế nhiều.

VRML cho phép truyền đi trong mạng những hình ảnh 3D. Với kích thước khá nhỏ so với băng thông, phần lớn giới hạn trong khoảng 100 - 200Kb nên các file VRML được truyền đi một cách khá dễ dàng. Nếu HTML là định dạng văn bản

thì VRML là định dạng đối tượng 3D. Hiện nay VRML có lợi thế là sự đơn giản, hỗ trợ dịch vụ web3D.

3.1.3. Lịch sử ra đời và phát triển của VRML

VRML đã trở thành một ngôn ngữ chuẩn cho việc mô phỏng tương tác thế giới 3D tren Web. Với mục đích xây dựng định dạng chuẩn cho phép mô tả thế giới thực trên máy tính mà cho phép chạy trên môi trường web, VRML đã trở thành chuẩn ISO từ năm 1997.

VRML ra đời vào mùa xuân năm 1994 ở hội nghị WWW được tổ chức đầu tiên tại Gieneva, Thụy Sĩ. Tim Berners-Lee và Dave Raggett đã tổ chức ra phiên họp có tên là Birds of a Feather (BOF) để mô tả giao diện thực tế ảo trên WWW. Nhiều thành viên tham dự phiên họp BOF đã mô tả nhiều dự án thực hiện việc xây dựng các công cụ hiển thị đồ họa 3D cho phép có nhiều thao tác hữu ích trên Web. Những thành viên này đã nhất trí đồng ý sự cần thiết cho các công cụ này có một ngôn ngữ chung, phổ biến cho định dạng, xác định việc mô tả thế giới 3D và các siêu liên kết WWW. Vì thế cụm từ “the Virtual Reality Markup Language” ra đời, từ Markup sau đó đã được đổi thành Modelling để phản ánh bản chất tự nhiên của VRML.

Sau phiên họp BOF một thời gian ngắn thì tổ chức www-vrml mailing list được thành lập để tập trung vào xây dựng phiên bản VRML đầu tiên. Sự hưởng ứng lời mời của tổ chức này kéo dài đến một tuần và có trên một nghìn khách mời tham dự. Tại buổi họp chủ tịch Mark Pesce đã thông báo ý kiến của mình là đưa ra phiên bản phác thảo xây dựng VRML đã có sẵn ở hội nghị mùa xuân năm 1994 được tổ chức mới cách đó 5 tháng. Bản phác thảo của Mark Pesce đã có được sự đồng ý chung.

Vào tháng 3/ 1995 thì có một thảo luận trên www-vrml mailing list liên quan đến việc tạo ra những tương tác của người sử dụng với hoạt cảnh và tất cả mọi người đều đi đến thống nhất ý kiến những thứ mới đưa ra đó thực sự cần thiết cho VRML. Công ty Silicon Graphics cộng tác với hãng Sony Research và Mitra để đưa ra phiên bản mới cho VRML. Bản đệ trình của Silicon Graphics có tên là Moving Worlds đến tổ chức Request for Proposals cho việc xây dựng phiên bản mới

VRML, bản đệ trình này là một minh chứng cho sự cộng tác thành công của tất cả các thành viên của Silicon Graphics, Sony và Mitra. Năm 1996 tại New Orleans, phiên bản đầu tiên của VRML 2.0 được đưa ra.

Vào tháng 7/1996 tổ chức tiêu chuẩn quốc tế (ISO) đã thông nhất ý kiến phiên bản năm 1996 của VRML 2.0 như là một đề xuất mà sẽ được đưa ra xem xet vào tháng 4/1997. Sau khi bỏ phiếu về chuẩn ISO thì VRML97 được đưa ra như một chuẩn ISO vào năm 1997.

3.1.4. Đặc điểm cơ bản của VRML

Tiếu chuẩn cho việc xác định đối tượng 3D, quang cảnh và cho sự liên kết các mô hình với nhau là:

- Không phụ thuộc phần cứng: có thể chạy trên các máy tính do các nhà sản xuất khác nhau chế tạo.
- Có thể mở rộng: có thể chấp nhận các lệnh mới do người sử dụng thêm vào hoặc quy định.
- Thao tác được thế giới ảo thông qua môi trường Internet có băng thông thấp.

Cùng với VRML thì có thể xem thông tin trong mô hình 3D trên Internet. VRML được thiết kế dành riêng cho việc hiển thị thế giới 3D và không phải là sự mở rộng của HTML. HTML có khả năng hiển thị các đối tượng tĩnh và động, các đối tượng multimedia cùng với các siêu liên kết khác đến các media khác như là văn bản, âm thanh, phim và hình ảnh.

3.2. Các vấn đề cơ bản của VRML

3.2.1. Các thành phần cơ bản của VRML

Gồm các trình duyệt (Browses) và bộ soạn thảo dành riêng cho ngôn ngữ VRML. Các file chỉ có thể đọc nếu hệ thống có trình duyệt VRML.

Trình duyệt VRML cũng giống như trình duyệt Internet(Internet Explorer hay Fire Fox) và được tích hợp trong các trình duyệt này.

Bộ soạn thảo VRML cho phép người dùng gõ mã VRML. Hiện nay có nhiều bộ soạn thảo nhưng VRML Pad là khá thông dụng khi có thể cho xem trực tiếp kết quả mà không cần qua trình duyệt Internet.

3.2.2. Công cụ hiển thị VRML

Để hiển thị các file VRML thì ta sử dụng chương trình Cortona VRML Client của hãng Parallrl Graphics. Chương trình sẽ giúp bạn thuận tiện hơn khi xem cắc mô hình ảo trên máy tính một cách trực quan sinh động.

Yêu cầu trước khi cài đặt Cortona VRML Client:

- Hệ điều hành Microsoft® Windows® ME/2000/XP

- Trình duyệt Web Internet Explorer 6.0 trở lên, Netscape Navigator 8.0 trở lên, Mozilla Firefox 1.5 trở lên, Opera 8.5 trở lên

- CPU Pentium® II 300 MHz trở lên.

- RAM tối thiểu 64 MB.

- Độ phân giải màn hình tối thiểu 1024x768.

- Card đồ họa hỗ trợ 3D và cài đặt DirectX 9

Cortona VRML Client tương thích với hầu hết các trình duyệt như Internet Explorer, Netscape Browser, Mozilla, Mozilla Firefox và các công cụ văn phòng như Word, PowerPoint...

Tính năng của Cortona VRML ClientCortona VRML Client sẽ trình diễn toàn bộ mô hình 3D trên máy tính một cách hoàn hảo như khi người tạo ra đó. ứng dụng hoàn chỉnh trên toàn bộ hiệu ứng trên nhiều hệ thống như Flash, DirectX9, MPEG4... Khi bạn truy xuất vào một ứng dụng VRML, toàn bộ hình mô phỏng sẽ được trình diễn tương tác trên nền 3D dạng mở. Rất ấn tượng và bắt mắt.

3.2.3. Tập tin của VRML

Tập tin của VRML có thể tạo ra từ nhiều cách: dùng một trình soạn thảo văn bản như Nopepad để soạn thảo, sau đó lưu file với phần mở rộng là. wrl. Hoặc ta có thể soạn thảo trên một trình soạn thảo dành riêng cho VRML như VRML Pad và chạy trực tiếp ứng dụng.

Một file VRML gồm có các phần như: header, scene graph, prototype và event routing.

- Header: dùng để nhận dạng tập tin VRML và cách mã hóa. Header của file VRML bắt đầu bằng dấu #. Ngoài lần xuất hiện đầu tiên ra thì dấu # đánh dấu những gì theo sau nó là phần chú thích. File tiêu đề của VRML có dạng: #VRML V1.0 ascii dành riêng cho phiên bản VRML 1.0 và #VRML V2.0 utf-8 dành cho phiên bản 2.0.
- Scene Graph: chứa những node mô tả các đối tượng và các thuộc tính đi kèm. Nó gần như một cây phả hệ gồm các nhóm đối tượng.
- Prototype: cho phép một tập các nút kiểu VRML được mở rộng bởi người sử dụng. Các định danh kiểu này có thể được bao hàm trong file (mà chúng được sử dụng) hay định nghĩa ở bên ngoài (file đó).
- Event routing: một số nút có thể phát sinh những sự kiện đáp trả những thay đổi môi trường do tương tác phía người dùng. "Event routing" cho phép một sự kiện phát sinh được truyền đến các "đích"- những nút trong hệ thống, từ đó gây ra những thay đổi cho riêng nút đó và hệ thống

3.3. Tìm hiểu chi tiết về VRML

3.3.1. Xây dựng các đối tượng hình học cơ bản

Một thế giới VRML được cấu tạo nên từ các đối tượng hình học như: hình hộp, hình trụ, hình nón, hình cầu, văn bản.

Shape là thẻ mà các đối tượng trong nó đều được hiển thị. Trong node Shape có node con là geometry chỉ rõ dạng hình học được vẽ ra thế nào. Node geometry có các thuộc tính:

- Box - hình hộp

Ví dụ :

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

Các tham số

  - size X Y Z (chiều rộng, chiều cao, chiều sâu)

- Cone - hình nón

Ví dụ :

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

Các tham số

- height X : chiều cao của hình nón
- bottomRadius Y : bán kính của đáy
- bottom TRUE / FALSE : hiện / ẩn đáy
- side TRUE / FALSE : hiện / ẩn mặt bên

❖ Cylinder : vẽ hình trụ

Ví dụ :

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

Các tham số

- height X : chiều cao
- radius Y : bán kính mặt đáy
- bottom TRUE / FALSE : hiện / ẩn mặt đáy
- top TRUE / FALSE : hiện / ẩn mặt nắp
- side TRUE / FALSE : hiện / ẩn các mặt bên

- Sphere : vẽ hình cầu

Ví dụ :

```
Shape{
Geometry Cylinder{
Radius 1.0
}
}
```

Các tham số

Radius X : bán kính

3.3.2. Xây dựng một số hình phức tạp

- Text : văn bản

Cú pháp

```
Shape{
Geometry Text{
String [“Text”,”Shape”]
fontStyle FontStyle{
family “SERIF”
style “BOLD”}
}
}
```

Các tham số

- string [“Nội dung văn bản”]
- family SERIP / SANS / TYPEWRITER
- style BOLD / ITALIC / BOLDITALIC / PLAIN
- size X : chiều cao của chữ
- spacing Y :
- justify FIRST / BEGIN / MIDDLE / END
- horizontal TRUE / FALSE : trình bày ngang / dọc màn hình
- leftToRight TRUE / FALSE : trình bày từ phải / trái sang

topToBottom TRUE / FALSE : trình bày từ trên / dưới

Ví dụ

```
Shape{
Geometry Text{
String [“Hi all, Today is fine.”]
fontStyle FontStyle{
        family “TYPEWRITER”
        style “plain”
        horizontal TRUE
        justify “MIDDLE”
        leftToRight TRUE
        size 0.9
        spacing 1.0
        topToBottom TRUE
        }
Lenght [6]
maxExtent 6
}
}
```

- Xây dựng các đường thẳng trong hệ tọa độ ba chiều

Thẻ IndexedLineSet xác định một tập hợp các đường thẳng trong hệ tọa độ không gian ba chiều của thế giới VRML và tập hợp các màu tương ứng cho các đường thẳng đó.

Thẻ này có các thuộc tính:

- Coord và color: số thành phần trong trường color không nhất thiết phải bằng với số thành phần trong trường coord. Chẳng hạn khi colorPerVexter nhận giá trị là FALSE.
- CoordIndex: bao gồm một dãy chỉ số thứ tự cho các điểm ảnh tạo nên đường thẳng. Ví dụ colorIndex [0 1 2 3] có nghĩa là đường thẳng được tạo bởi điểm thứ nhất nối với điểm thứ hai, thứ hai nối với thứ ba, thứ ba nối với thứ tư, các điểm ảnh được xác định trong thẻ Coordinate. Một ví dụ khác coordIndex [0 1 -1 2 3] ở dãy kí hiệu -1

cho biết hình ảnh được tạo trong VRML gồm hai đường thẳng: một đường thẳng tạo bởi điểm thứ nhất nối với điểm thứ hai và một đường thẳng khác tạo bởi điểm thứ ba nối với thứ tư.

- ColorIndex: được sử dụng khi colorPerVexter nhận giá trị FALSE, khi colorPerIndex nhận giá trị TRUE thì có thể bỏ qua trường colorINdex trong thẻ IndexedLineSet bởi vì nếu colorPerIndex nhận giá trị TRUE thì bắt buộc phải thiết lập mầu cho mỗi điểm ảnh.
- ColorPerVexter: có giá trị boolean. Khi colorPerVexter nhận giá trị TRUE thì màu của đườn thẳng sẽ là màu trung bình cả hai màu tại hai điểm ảnh tạo nên đường thẳng.

Cú pháp

```
IndexedLineSet{
Coord coordinate[]
coordIndex
color Color[]
colorIndex[]
colorPerVexter TRUE
}
```

Ví dụ

```
Shape{
Appearance Appearance{}
Geometry IndexedLineSet{
Coord Coordinate{
Point[
        0.0 0.0 0.0,0.0 4.0 0.0, 2.0 0.0 0.0,]
}
Color Color{
Color[
1.0 1.0 1.0,1.0 0.0 0.0,0.0 0.0 1.0
]
}
coordIndex[
0 1 2 0
```

```
]
colorPerVexter TRUE
}
}
```

- Xây dựng khung bề mặt trong không gian

Thẻ IndexedFaceSet sẽ tạo lên bề mặt bằng cách kết hợp các điểm với nhau.

Cú pháp

```
IndexedFaceSet
{
        Coord Color[]
        coordIndex[]
        color color[]
        colorIndex[]
        colorPerVexter TRUE
        convex TRUE
        solid TRUE
}
```

- Coord, coordIndex, color, colorIndex, colorPerVexter có đặc điểm tương tự như đã nói ở trên. Chú ý là bề mặt luôn luôn được xác định bởi các đường thẳng khép kín vì thế không phải chỉ ra điểm đầu tiên lại một lần nữa trong trường coord.

Ví dụ

```
Coord Coordinate
{
Point [0 0 0,1 0 0,1 1 0,0 1 0]
}
```

Đã có bốn điểm xác định, khi liên kết các điểm này sẽ tạo ra hình vuông bằng cách sử dụng thuộc tính coordIndex[] và coordIndex[0 1 2 3] nói rằng điểm thứ nhất nối với điểm thứ hai, điểm thứ hai nối với điểm thứ ba, điểm thứ ba nối với điểm thứ tư và điểm thứ tư nối với điểm thứ nhất.

Convex định nghĩa bề mặt là lồi hay lõm, trường này có giá trị kiểu boolean. Bộ trình duyệt VRML chỉ vẽ bề mặt lồi, trong trường hợp vẽ

bề mặt lõm, bộ trình duyệt chia bề mặt đó thành các bề mặt lồi bé hơn để vẽ.

Ví dụ

```
Shape
{
        Appearance appearance{}
        Geometry IndexedFaceSet
        {
                Coord coordinate
                {
                        Point[0 0 0,1 1 0,0 1 0,1 1 0,1 0 0,1 0 1]
                }
        coordIndex[0 1 2 3 -1 4 5 6]
        color Color
        {
                Color[1 1 1,1 0 0]
        }
        colorIndex [0 1]
        colorPerVexter FALSE
        convex FALSE
        solid FALSE
}
}
```

- Xây dựng khung lưới và bản đồ trong không gian

Thẻ ElevationGrip cho phép xây dựng khung lưới được tạo bởi các điểm có độ cao xác định trong không gian. Thẻ này rất hữu ích cho việc xây dựng các mạng lưới hoặc địa hình.

Hình ảnh xây dựng được đặt trong không gian của mặt phẳng OXYZ. Điểm bắt đầu là gốc tọa độ, các điểm còn lại tạo nên lưới phải nằm theo hướng dương của các trục OX avf OZ.

Cú pháp

```
ElevationGrid
{
        xDimension 0
        xSpacing 0.0
        zDimension 0
        zSpacing 0.0
        height[]
                color color[]
                colorPerVexter TRUE
                convex TRUE
                solid TRUE
}
```

Tham số

- Color, colorPerVexter, convex, solid có tính chất tương tự như đã nói ở phần trên.
- xDimension chứa số điểm bên trong lưới nằm trên trục X.
- zDimension chứa số điểm bên trong lưới nằm trên trục Z.
- xSpacing là khoảng cách của hai điểm liên tiếp nhau theo hướng của trục X.
- zSpacing là khoảng cách của hai điểm liên tiếp nhau theo hướng của trục Z.
- height chứa một danh sách các giá trị độ cao của mỗi điểm trong lưới.

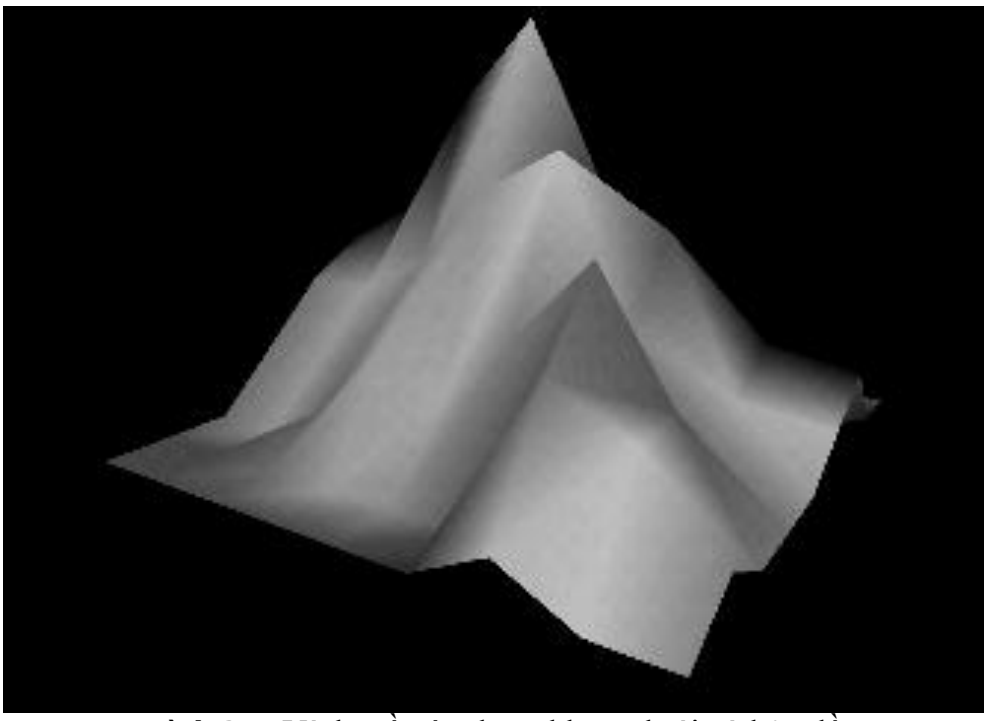
Các điểm này được tính theo thứ tự từ trái sang phải từ trên xuống dưới.

Ví dụ

```
Shape {
appearance Appearance {
material Material { }
}
geometry ElevationGrid {
xDimension  9
zDimension  9
xSpacing    1.0
```

```
zSpacing    1.0
solid       FALSE
height [
0.0, 0.0, 0.5, 1.0, 0.5, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 2.5, 0.5, 0.0, 0.0, 0.0,
0.0, 0.0, 0.5, 0.5, 3.0, 1.0, 0.5, 0.0, 1.0,
0.0, 0.0, 0.5, 2.0, 4.5, 2.5, 1.0, 1.5, 0.5,
1.0, 2.5, 3.0, 4.5, 5.5, 3.5, 3.0, 1.0, 0.0,
0.5, 2.0, 2.0, 2.5, 3.5, 4.0, 2.0, 0.5, 0.0,
0.0, 0.0, 0.5, 1.5, 1.0, 2.0, 3.0, 1.5, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2.0, 1.5, 0.5,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.0, 0.0,
]}}
```

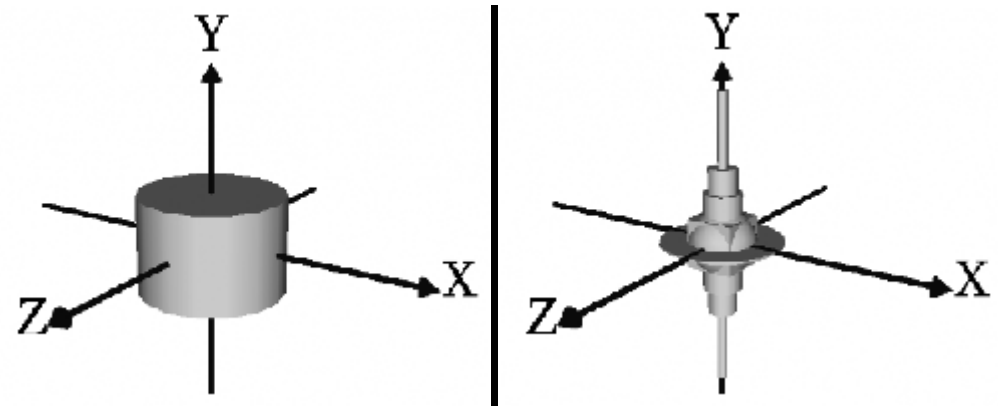
Kết quả



Hình 3.1 Ví dụ về xây dựng khung lưới và bản đồ

3.3.3. Các phép biến đổi trong VRML

Một file VRML xây dựng các thành phần của thế giới ảo. Mặc định mọi hình ảnh được xây dựng trong VRML đều được đặt ở tọa độ gốc. Biến đổi là tạo ra một hệ thống hệ trục tọa độ mới mà hệ truc này có vị trí tương đối so với hệ tọa độ mặc định. Có các phép biến đổi như: di chuyển, co giãn, xoay.



Hình 3.2 Trục XYZ với 1 hình cơ bản và 1 hình phức tạp

Nhóm Node Transform quản lý các phép biến đổi trong VRML như : dịch chuyển, co dãn, xoay. Node transform gồm có các node con: translation, rotation, scale, children.

- Phép di chuyển (Translation)

Ví dụ :

```
Transform {
translation 2.0 0.0 0.0
children [. . . ]
}
```

Tham số :

- translation X Y Z

- Phép xoay (Rotation)

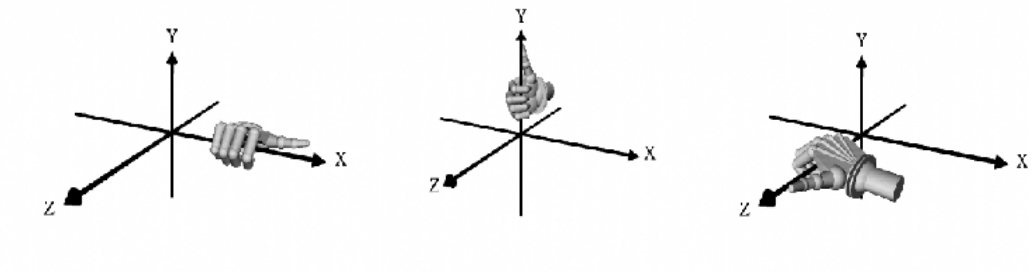
Ví dụ :

```
Transform {
rotation 0.0 0.0 1.0 0.52
children [. . . ]
}
```

Tham số :

- rotation X Y Z Góc
- Góc = radians = degrees / 180.0 * 3.141

Lưu ý: để dễ dàng trong việc xoay đối tượng theo đúng chiều ta có quy tắc bàn tay phải.



Hình 3.3 Quay theo trục X, Y, Z

Nắm bàn tay phải lại, choãi ngón cái ra theo như hình vẽ. Ngón tay cái chỉ phương của trục quay thì chiều từ cổ tay đến các ngón tay chi chiều dương khi quay đối tượng.

- Scale : co dãn đối tượng

Ví dụ :

```
Transform {
scale 0.5 0.5 0.5
children [. . . ]
}
```

Tham số :

- scale X Y Z

- Children : Các đối tượng đước vẽ trong này sẽ chịu ảnh hưởng của các phép biến đổi trong node Transform.

Vi dụ :

```
Transform {
translation -2.0 -1.0 0.0
children [
 Shape {
  appearance Appearance {
   material Material { }
  }
  geometry Cylinder {
   radius 0.3
   height 6.0
   top FALSE
  }
```

```
}
]
}
Shape {
        Geometry Box { size 1.0 1.0 1.0}
}
```

Như ví dụ trên thì hình trụ sẽ dịch chuyển so với tọa độ gốc theo trục X là -2.0 và theo trục Y là -1.0, còn hinh hộp thì nằm tại tọa độ gốc

3.3.4. Màu sắc trong VRML

Các đối tượng khi vẽ sẽ có mặc định là màu trắng và bạn có thể kiểm soát các thứ như : Shading color, Glow color, Transparency, Shininess, Ambient intensity. Màu sắc trong VRML được thể hiện qua 3 thông số là R G B(red-green-blue) với giá trị nằm trong khoảng từ 0.0 đến 1.0.

Node material nằm trong node appearance quản lý việc này. Node material có các node con là :

- diffuseColor : màu sắc khuyết tán
- emissiveColor : màu sắc của ánh sáng phát ra từ đối tượng
- transparency : độ trong suốt (0 : tối mờ, 1 : nét)
- ambientIntensity : xác định hàm lượng ánh sáng mà đối tượng phản chiếu
- specularColor : cho biết màu sắc của các pixel trên bề mặt của đối tượng
- shininess : cường độ sáng

Ví dụ :

```
Shape {
 appearance Appearance {
  material Material {
  diffuseColor 0.8 0.8 0.8
   emissiveColor 0.0 0.0 0.0
   transparency 0.0
   specularColor 0.71 0.70 0.56
   shininess 0.16
```

```
 ambientIntensity 0.4
}
}
geometry. . .
}
```

3.3.5. Nhóm node

Ta có thể nhóm các đối tượng đơn lẻ thành những đối tượng phức tạp. Việc đặt một nhóm này vào nhóm kia tạo thành một lược đồ cấu trúc các node. Một node trong nhóm có thể có bất kì các node con nào đặt bên trong trường children.

Có các node nhóm sau được hỗ trợ trong VRML: Anchor, Billboard, Group, Switch, Transform, Inline

- Anchor: cho phép xác định một tập hợp các đối tượng và cách liên kết đến một url, chẳng hạn như là một siêu liên kết đến thế giới VRML khác, đến một trang HTML hoặc đến một dữ liệu nào đó mà toàn bộ trình duyệt có thể đọc được. Khi kích vào một trong các đối tượng bên trong node Anchor thì toàn bộ trình duyệt sẽ đưa đến địa chỉ url. Khi đưa chuột qua đối tượng nào đó chứa trong node thì có thể nhìn thấy được địa chỉ url của nó.

  Ví dụ

```
Anchor{
Children[
Shape{
Geometry Sphere{}}]
url http://www.cuasoit.com
description “Toi trang web cua lop CT 901”
}
```

  Các thuộc tính

  - children: chứa tất cả các node bên trong nhóm
  - url: cho biết địa chỉ url được tìm kiếm. Có thể có nhiều vị trí đặt trong url, khi đó bộ trình duyệt sẽ tìm kiếm dữ liệu trong các vị trí này theo thứ tự ưu tiên giảm dần.

- parameter: cung cấp thông tin thêm vào cho bộ trình duyệt. Cho ví dụ có thể chỉ rõ là cửa sổ cần xem mà url hiển thị vào đó.
- description: đưa ra một xâu, xâu này sẽ hiển thị khi người dùng di chuột qua đối tượng chứa trong node Anchor.
- bboxCenter: xác định trung tâm của một hình hộp chữ nhật mà nó bao quanh tất cả các node trong nhóm. Giá trị của trường này là một điểm trong không gian.
- bboxSize: xác định kích thước của hình hộp chữ nhật bao quanh tất cả các node trong nhóm. Mặc định trường này có giá trị -1 -1 -1 nghĩa là không có hình hộp nào. Tất cả các giá trị thành phần của trường này là lớn hơn hoặc bằng 0.Nếu tất cả các node con không nằm trong hình hộp thì kết quả không xác định. Hai trường bboxCenter và bboxSize là tùy chọn.

- Group: tạo ra một tập hợp các vật thể như là một thực thể đơn. Node này có các trường: children, bboxCenter, bboxSize. Các trường này tương tự như của node Anchor. Tất cả các node con trong Group đều hiển thị.

  Ví dụ:

  ```
  Group{
  Children[]
  bboxCenter 0 0 0
  bboxSize -1 -1 -1
  }
  ```

- Switch: tạo ra một nhóm chuyển đổi. Chỉ có một node con trong nhóm được hiển thị. Và node này là do người dùng lựa chọn. Node con tuyệt đối được đánh số từ 0, không lựa chọn đánh là -1.

  Ví dụ:

  Các thuộc tính:

  - whichChoice
  - choice

- Transform: xác định hệ trục tọa độ mới cho các đối tượng trong cùng một nhóm. Node này đã được trình bày ở trên.

- Billboard: node này tạo ra một nhóm với hệ tọa độ dặc biệt. Tất cả các node trong nhóm đều được hiển thị. Hệ tọa độ này luôn được quay về đối diện với người dùng.

  Vi dụ:

  ```
  Billboard{
  axisOfRotation 0.0 1.0 0.0
  children [....]
  }
  ```

  Các thuộc tính:

  - axisOfRotationbboxCenter
  - bboxSize

- Inline: node này sẽ được trình bày ở phần sau

3.3.6. Một số phương pháp vẽ trong VRML

Trong VRML chúng ta có thể vẽ trực tiếp đối tượng. Tuy nhiên trong nhiều trường hợp, đối tượng được sử dụng nhiều lần thì việc vẽ trực tiếp sẽ rất mất thời gian, công sức và tăng dung lượng file lên đáng kể.

Có 2 cách thông dụng nhất để giải quyết việc này đó là:

- Inline: hàm Inline cho phép ta gọi trực tiếp một đối tượng bên ngoài file vào file hiện tại.

  Cú pháp:

  ```
  Inline {url ”Tên đường dẫn”}
  ```

  Vi dụ:

  ```
  Inline {url ”robo.wrl”}
  ```

- DEF (DEFine): cho phép ta định nghĩa một đối tượng hay một kiểu thuộc tính

  Vi dụ

  ```
  Shape {
   appearance Appearance {
    material DEF RedColor Material {
     diffuseColor 1.0 0.0 0.0
    }
  ```

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

Như ví dụ trên ta định nghĩa đối tượng RedColor kiểu Material. Sau đó muốn sử dụng lại ta gọi hàm bằng từ khóa USE.

3.3.7. Texture Mapping

Ta có thể mô hình mỗi chi tiết kết cấu nhỏ bé của một đối tượng. Thay vì phải mất thời gian để vẽ ta có thể lấy một bức ảnh dán lên bề mặt kết cấu như là dán decal. Kĩ thuật này được gọi là ánh xạ kết cấu.

Trong VRML có thể sử dụng một trong các định dạng ảnh sau đây:

- GIF: 8bit, có hỗ trợ trong suốt, không là sự lựa chọn tốt cho texture mapping.
- JPEG: từ 8-16 bit, không hỗ trợ trong suốt, là sự lựa chọn tốt cho texture mapping.
- PNG: từ 8-16 bit, hỗ trợ trong suốt trên từng điểm ảnh, là sự lựa chọn tốt nhất cho texture mapping.

3.3.8. Script

Thẻ script đưa ra khả năng linh hoạt, mềm dẻo để mở rộng gần như vô hạn các yêu cầu hành động của chúng ta cũng như hành động mong đợi phản hồi từ thế giới ảo của chúng ta. Thẻ script cho phép chúng ta xác định cách tương tác với thế giới ảo bằng việc sử dụng các ngôn ngữ lập trình như Java, Javascript hoặc VRML script mới đây được đưa ra bởi Silicon Graphics, Inc.

Trong một thể Script có thể có các trường hay các thuộc tính, nhưng không giống với các thẻ khác, trong thẻ script chúng ta có thể xác định các sự kiện mà thẻ này có thể nhận và gửi sự kiện. Một thẻ script sẽ xác định một hành động nào đó

mỗi lúc nó nhận một sự kiện, trong mỗi hành động thẻ script có thể tạo ra nhiều sự kiện. Hơn nữa thẻ script có thể được dụng để xây dựng các thủ tục khởi tạo (initiazation procedures) và dừng, tắt (shutdown procedures).

Một thẻ script có thể nhận bất kỳ các sự kiện, các giá trị mà các sự kiện này có được là các biến thuộc loại eventIn. Tên của các biến này là tên của các sự kiện mà thẻ script nhận sự kiện đó. Các biến này là chỉ có thể đọc nghĩa là chúng ta không thể thiết lập giá trị trực tiếp cho bất kỳ biến nào thuộc loại này. Một thẻ script cũng tạo ra bất kỳ các sự kiện nào, các giá trị mà các sự kiện này có được là các biến thuộc loại eventOut.

Cú pháp

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

- url cho biết chương trình script được xử lí. Các ngôn ngữ gần đây hỗ trợ cho việc xây dựng các chương trình script này là: JAVA, JavaScript, VRMLScript.
- Có hai cách mà có thể thay đổi thế giới ảo bằng việc sử dụng script: thứ nhất là sử dụng một biến eventOut và đặt sự kiện đến thẻ khác, thứ hai là truy cập trực tiếp các trường của thẻ khác, chú ý là cả hai cách này đều phải thực hiện trong thẻ script.

Điều khó khăn cho bộ trình duyệt là vừa phải vẽ thế giới ảo vừa phải xử lý các sự kiện ở cùng một thời gian.

CHƯƠNG 4. ỨNG DỤNG VRML TRONG VIỆC XÂY DỰNG MÔ HÌNH TỦ KÍNH TRƯNG BÀY CỔ VẬT

4.1. Bài toán.

Trên thế giới có rất nhiều tủ kính trưng bày cổ vật .Nhưng việc chiêm ngưỡng một cách cụ thể trên máy tính là rất khó. Vì thế với em đã sử dụng ngôn ngữ VRML xây dựng tủ trưng bày để mọi người có thể tham quan chiêm ngưỡng các tác phẩm nghệ thuật ngay tại máy tính của mình mà vẫn có cảm giác như thật vậy.

Mô hình được thể hiện trên nền một trang web nên mọi người ở nhiều nơi có thể tham gia trực tiếp vào mô hình mà không cần phải cài đặt gì nhiều.

4.2. Yêu cầu đặt ra và hướng giải quyết.

Về mặt thẩm mỹ:

- Mô hình thể hiện chân thực toàn cảnh của mô hình.
- Kết cấu và màu sắc như thật, bắt mắt.

Về mặt kỹ thuật:

- Người dùng có thể tham quan trực tiếp, quan sát tận bên trong tủ kính.
- Người dùng có thể tương tác với tủ kính như thật (mở đóng cửa, di chuyển cổ vật,…).
- Ứng dụng của mô hình gọn nhẹ, có thể truyền tải lên website một cách nhanh chóng.
- Thao tác trên mô hình một cách dễ dàng mà không cần phải học cách sử dụng.

4.3. Một số kỹ thuật xây dựng mô hình.

4.3.1. Kế thừa sử dụng mô hình gốc.

Ngôn ngữ VRML hỗ trợ các đối tượng hình học cơ bản như: hình hộp, hình cầu, hình trụ,… đến các đối tượng hình học phức tạp như: khung bề mặt trong không gian,… Vì thế, nhờ vào việc sử các đối tượng hình học này ta có thể dễ dàng xây được toàn bộ khung tủ kính. Tuy nhiên nếu chúng ta cứ sử dụng lập đi lập lại các đối tượng hình học giống nhau sẽ gây ra việc lãng phí tài nguyên lưu trữ cũng như làm chậm tiến trình tải cảnh của thế giới ảo. Vì vậy trong ứng dụng này đã sử dụng tối đa các kỹ thuật sử dụng lại.

Ví dụ:

- Sử dụng nút Inline: để xây dựng tòa nhà cần rất nhiều cửa sổ, vì vậy chỉ cần xây dựng cửa sổ một lần và lưu vào trong file "ban.wrl". Khi sử dụng, tại vị trí mới, ta chỉ cần dùng nút Inline để gọi lại như sau:

```
Transform {
        translation 86.2 10.02 2.8
        children [
                Inline    {url "ban.wrl"}
        ]
}
```

- Sử dụng DEF: trong tòa nhà còn sử dụng một hình hộp, ta có thể dùng hàm DEF để định nghĩa một box, sau đó khi sử dụng ta gọi nó tại vị trí mới.

```
DEF box1 Transform {
                translation 0 15 0
                children [
                        Shape {
                                appearance Appearance {
                                        material Material {
                                         diffuseColor 1 1 0.8888
                                        }
                                }
                                geometry Box { size 100 0.5 100 }
                        }
                ]
}

Transform {
                translation 108.9 30 0.1125
                rotation 0 0 1 -3.142
                children [
                        USE box1  ]
```

4.3.2. Sử dụng cảm biến.

Để ứng dụng trông thực hơn, ngoài khả năng cho phép người dùng quan sát, ứng dụng còn cho phép người dùng thao tác, điều khiển như việc: đóng mở cửa,di chuyển đồ vật,… Để làm được những điều này bắt buộc phải sử dụng các cảm biến được hỗ trợ trong VRML.

- *Dùng các cảm biến rê chuột để có thể tạo ra hiệu ứng đóng mở cửa bằng tay.*

  Ví dụ:

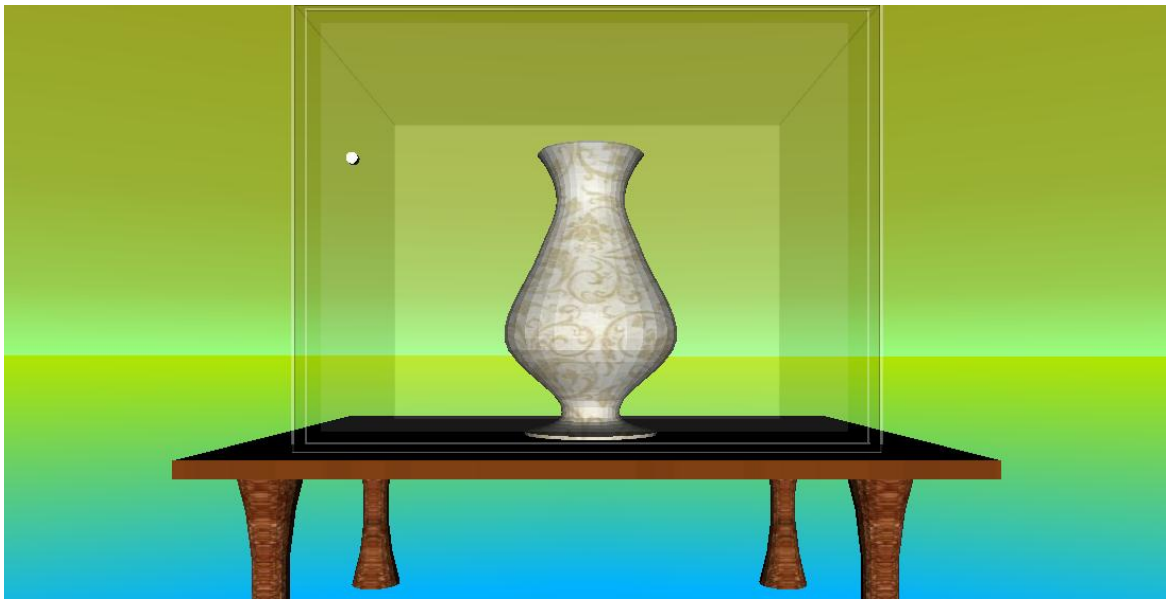
```
Transform {
        children [
            DEF SENSOR1 CylinderSensor {
                enabled FALSE
                minAngle -1.57
                maxAngle 1.57
            }
            USE Canhtrai
        ]
  ROUTE SENSOR1.rotation_changed TO Canhtrai.rotation
}
```

- *Dùng cảm biến để di chuyển đồ vật trên mặt bàn.*

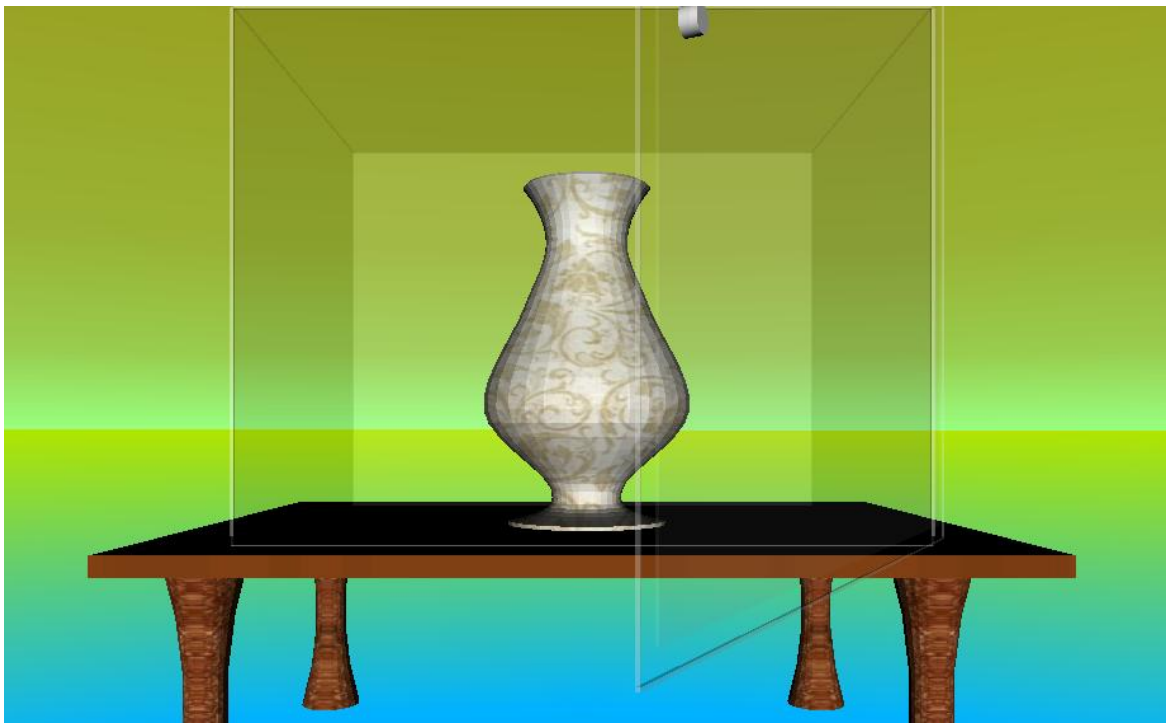
```
DEF T3 Transform {
 children [
        children [
      Shape {
       appearance Appearance {
        material Material {
         diffuseColor 0.03137 0.03137 0.5333          }
       }          geometry Box { size 7 7 7 }        }
  ] }
DEF PS3 PlaneSensor {
offset -30 51 5
minPosition -55 52
       maxPosition 45 100}
 ]}ROUTE PS3.translation_changed TO T3.set_translation
```

4.3.3. Kết quả đạt được.

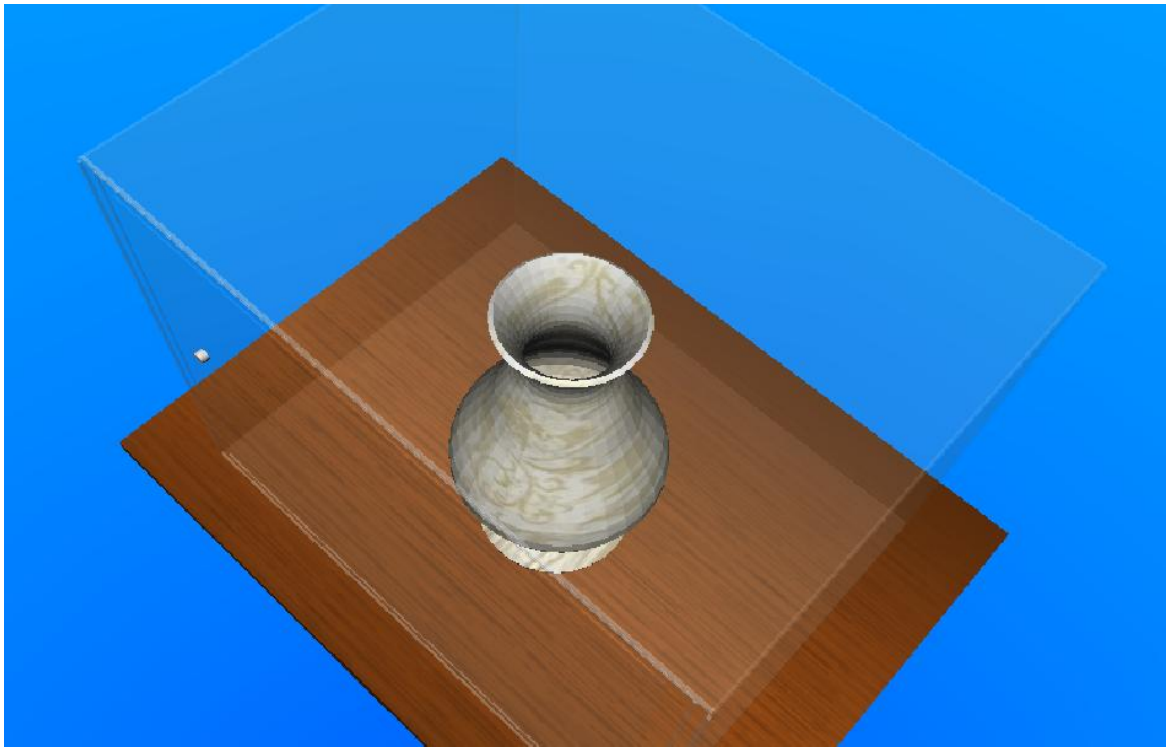
Từ việc áp dụng ngôn ngữ VRML ta thu được các kết quả sau



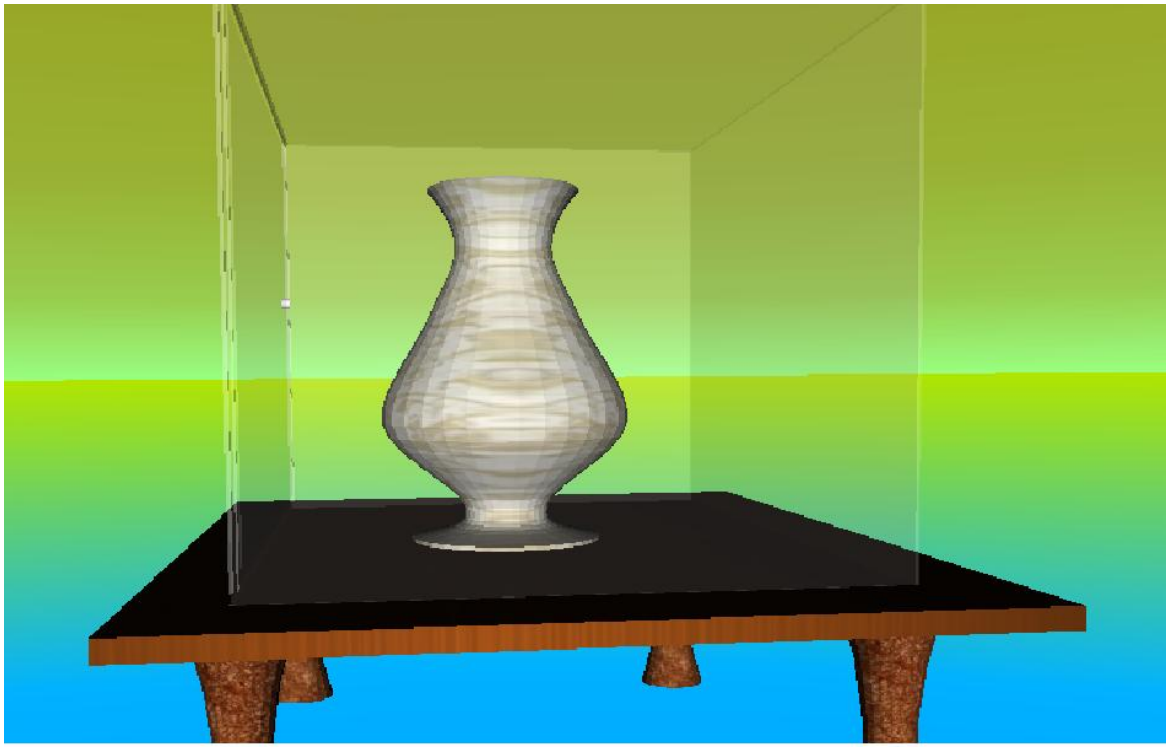
Hình 4.1 khung nhìn chính.



Hình 4.2 tủ kính khi mở.



Hình 4.3 khung hình nhìn từ trên xuống



Hình 4.4 khung hình nhìn từ bên trái.

KẾT LUẬN

Ngày nay với sự phát triển của khoa học kỹ thuật đã mang lại những hiệu quả to lớn đặc biệt là tin học. Ở trong bất kì lĩnh vực nào của cuộc sống tin học cũng có mặt và hiệu quả của nó là một điều mọi người mong đợi. Đặc biệt trong lĩnh vực thực tế ảo là lĩnh vực còn khá mới mẻ nhưng có tầm quan trọng rất lớn.

Cũng giống như các lĩnh vực khác của công nghệ thông tin, thực tế ảo có ngôn ngữ riêng của mình đó chính là VRML. Đồ án này nhằm mục đích tìm hiểu rõ hơn về ngôn ngữ này. Qua tìm hiểu, nghiên cứu cũng như tham khảo đề tài của khóa trên và một số mã nguồn mở trên mạng nên đề tài của em đã đạt được một số mục đích:

- Tìm hiểu tổng quan về thực tại ảo
- Tìm hiểu về ngôn ngữ thực tại ảo VRML
- Xây dựng được chương trình ứng dụng

Có thể nói rằng thực tại ảo đóng một vai trò hết sức quan trọng bởi những vấn đề khó khăn mà nếu không có thực tại ảo thì rất khó để giải quyết. Nhưng khi ứng dụng thực tại ảo vào thì vấn đề đó sẽ trở nên đơn giản hơn với chi phí và thời gian ít hơn.

Tuy nhiên vì thời gian làm bài của em có hạn nên đề tài còn rất nhiều thiếu sót. Mong các thầy cô và các bạn góp ý.

Em xin chân thành cảm ơn!

TÀI LIỆU THAM KHẢO

[1]. Lê Tấn Hùng, Huỳnh Quyết Thắng, Kỹ thuật đồ hoạ, Nhà xuất bản Khoa học và Kỹ thuật – 2000, 2002, 2004

[2]. http://tecfa.unige.ch/guides/vrml/pointers.html

[3]. http://www.lighthouse3d.com/vrml/tutorials.shtml

[4 VRML Audio Tutorial, http://www.dform.com/inquiry/tutorials/vrmlaudio/

[5]. Bài viết "Ứng dụng VRML trên các trình duyệt Web" của Phạm Lê Minh Định, http://tintuc.xalo.vn/00-110676726/ung_dung_VRML_tren_cac_trinh_duyet_Web.html?id=1bafe4b&o=206

[6]. Bài viết "Tạo tập tin VRML" của Hoài An, www.echip.com.vn/echiproot/html/2004/so114/taotaptinvrml.html

[7]. Các bài viết trên http://vi.wikipedia.org và các website Công nghệ thông tin.

[8]. How Virtual Reality Works, http://electronics.howstuffworks.com/gadgets/other-gadgets/virtual-reality8.htm

[9]. VRML Tutorial, http://www.lighthouse3d.com/vrml/tutorial/

[10]. The Virtual Reality Modeling Language Specification, http://graphcomp.com/info/specs/sgi/vrml/